

**OPTIMASI METODE ADAPTIVE FUZZY K-NEAREST
NEIGHBOR DENGAN PARTICLE SWARM OPTIMIZATION
UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI
KELUARGA**

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

UNIVERSITAS BRAWIJAYA



Disusun Oleh:

IKA RETNO PUTRI

NIM. 125150209111006

KEMENTERIAN RISET TEKNOLOGI DAN PENDIDIKAN TINGGI

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2015

**OPTIMASI METODE ADAPTIVE FUZZY K-NEAREST
NEIGHBOR DENGAN PARTICLE SWARM OPTIMIZATION
UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI
KELUARGA**

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

UNIVERSITAS BRAWIJAYA



Disusun Oleh:

IKA RETNO PUTRI

NIM. 125150209111006

KEMENTERIAN RISET TEKNOLOGI DAN PENDIDIKAN TINGGI

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2015

LEMBAR PERSETUJUAN

OPTIMASI METODE *ADAPTIVE FUZZY K-NEAREST NEIGHBOR* DENGAN *PARTICLE SWARM OPTIMIZATION* UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer



Disusun Oleh:

IKA RETNO PUTRI

NIM.125150209111006

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I,

Dosen Pembimbing II,

Imam Cholissodin, S.Si., M.Kom.
NIK. 850719 16 11 0422

Budi Darma S, S.Kom., M.Cs.
NIK. 841015 06 11 0090

LEMBAR PENGESAHAN

OPTIMASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR DENGAN PARTICLE SWARM OPTIMIZATION UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA

SKRIPSI

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh:

IKA RETNO PUTRI
NIM. 125150209111006

Skripsi ini telah diuji dan dinyatakan lulus pada

Tanggal 16 Januari 2015

Penguji I,

Candra Dewi, S.Kom., M.Sc.
NIP. 19771114 200312 2 001

Penguji II,

Rekyan Regasari MP, S.T.,M.T.
NIK. 770414 06 1 2 0253

Penguji III,

Indriati, S.T.,M.Kom.
NIK. 831013 06 1 2 0035

Mengetahui,

Ketua Program Studi Informatika / Ilmu Komputer,

Drs. Marij., M.T.
NIP. 19670801 199203 1 001

PERNYATAAN ORISINALITAS SKRIPSI

Saya yang bertanda tangan di bawah ini:

Nama : Ika Retno Putri

NIM : 125150209111006

Program Studi : Teknik Informatika

Judul : **OPTIMASI METODE ADAPTIVE FUZZY K-NEAREST
NEIGHBOR DENGAN PARTICLE SWARM OPTIMIZATION
UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI
KELUARGA**

Dengan ini menyatakan bahwa saya dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata didalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku. (UU No. 20 Tahun 2003, Pasal 25 Ayat 2 dan Pasal 70).

Malang, Januari 2015

Mahasiswa

Ika Retno Putri

NIM 125150209111006

KATA PENGANTAR

Segala puji bagi Allah SWT yang telah melimpahkan rahmat dan karunia-Nya kepada seluruh umat-Nya sehingga penulis dapat menyelesaikan Skripsi yang berjudul **“OPTIMASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR DENGAN PARTICLE SWARM OPTIMIZATION UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA”** dengan baik. Sholawat serta salam kami haturkan kepada junjungan kami Nabi Muhammad SAW, semoga kami selalu istiqomah dalam meneladaninya. Skripsi ini merupakan serangkaian mata kuliah dan syarat kelulusan yang harus ditempuh oleh mahasiswa sebagai pengaplikasian dari beberapa mata kuliah yang telah di pelajari di bangku kuliah.

Tidak terlupakan penulis juga mengucapkan terimakasih yang sebesar besarnya kepada pihak-pihak yang membantu terselesaikannya laporan Skripsi ini, ucapan tersebut kami berikan kepada:

1. Syukur Alhamdulillah kepada Allah SWT, atas syukur dan nikmat kesehatan yang telah diberikan.
2. Ir. Sutrisno, M.T., Ir. Heru Nurwarsito, M.kom., Himawat Aryadita, S.T., M.Sc., dan Edy Santoso, S.Si., M.Kom., selaku Ketua, Wakil Ketua 1, Wakil Ketua 2, Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
3. Drs. Marji, M.T. dan Issa Arwani, S.Kom., M.Sc., selaku Ketua dan Sekertaris Program Studi Teknik Informatika Universitas Brawijaya.
4. Imam Cholissodin, S.Si., M.Kom. dan Budi Darma Setiawan, S.Kom., M.Cs., selaku dosen pembimbing selama pelaksanaan skripsi.
5. Bapak dan Ibu Dosen yang telah memberikan ilmu dan pengetahuan serta wawasan baru kepada kami sebagai anak didik.
6. Seluruh Civitas Akademika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya yang telah banyak memberikan bantuan dan dukungan selama menempuh studi dan menyelesaikan skripsi ini.

7. Ayahanda dan Ibunda tercinta, serta keluarga besar dirumah, terimakasih atas curahan doa, kasih sayang yang tulus serta dukungan yang telah diberikan.
8. Kakak tercinta Luvita Yuni Setyarini dan Hilmi Ridwan Firdausi yang selalu memberi wejangan, support, motivasi, bantuan, serta doa yang telah diberikan.
9. Anditya Adi Prasetyo beserta keluarga besarnya yang selalu mendoakan, mensupport dan memberikan motivasi.
10. Sahabat-sahabat tercinta Diah Angraeni Pitaloka, Kiki Aprilia Puspitaningrum, Andi Andriyan Krisnayanti, Humairah Fauziah, Afe Faleniko dan Nanda Yulimawanto trimakasih atas bantuan dan suportnya.
11. Teman-teman Galang, Titis, Tya, Dody, Dian Arya, Dian Rifia, Iyan, Pratiwi, Vinda, Shery, Ilvi, Cahya, Ifa, Meliza, Erisa dan seluruh angkatan SAP 2012 Teknik Informatika, terima kasih atas dukungannya dan berbagai pihak yang tidak dapat disebut satu persatu yang telah memberikan bantuan dan dukungannya.
12. Teman-teman dan adik-adik kavlas Tika, Dian, Novella, Asfa Durri, Nea P Wijaya, Hanifa Muslima, Eka Mutiarazani, Desty Yuandini, Savrida Nurahmi, trimakasih atas bantuan dan motivasi yang diberikan

Penulis yakin bahwa Skripsi ini masih sangat jauh dari sempurna. Saran dan kritik yang membangun dapat disampaikan melalui email penulis irputri66@gmail.com. Harapannya Skripsi ini dapat memberikan manfaat bagi penulis dan para pembaca.

Malang, Januari 2015

Penulis

ABSTRAK

Ika Retno Putri. 2014. Optimasi Metode Adaptive Fuzzy k-Nearest Neighbor dengan Particle Swarm Optimization untuk Klasifikasi Status Sosial Ekonomi Keluarga. Skripsi Program Studi Informatika / Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer. Pembimbing: Imam Cholissodin, S.Si., M.Kom. dan Budi Darma S., S.Kom., M.Cs.

Penelitian ini bertujuan untuk klasifikasi status sosial ekonomi keluarga pada 3 kelas yaitu kelas atas, menengah, dan bawah dilihat dari aspek pendidikan, pekerjaan, pendapatan, pemilikan, dan jenis tempat tinggal dengan menggunakan metode *adaptive fuzzy k-nearest neighbor* (FKNN) dengan *particle swarm optimization* (PSO), dimana metode FKNN digunakan pada proses PSO yaitu untuk menghitung nilai *fitness* partikel dan pada pengujian hasil implementasi metode *adaptive* FKNN berdasarkan PSO. Adanya *time variant* pada proses PSO yaitu *time varying acceleration coefficients* (TVAC) dan *time varying inertia weight* (TVIW), sehingga metode ini disebut dengan metode *time variant particle swarm optimization-fuzzy k-nearest neighbor* (TVPSO-FKNN). TVPSO-FKNN digunakan untuk mengoptimalkan kinerja proses klasifikasi FKNN, dimana hasil dari proses TVPSO-FKNN adalah nilai m , k , dan fitur yang optimal untuk klasifikasi FKNN. Dari 5 skenario pengujian didapatkan nilai rata-rata akurasi mencapai 91.31 % pada pengujian jumlah iterasi. Metode TVPSO-FKNN terbukti optimal untuk meningkatkan kinerja proses klasifikasi FKNN pada klasifikasi 3 kelas.

Kata kunci: *Fuzzy k-nearest neighbor, time variant, particle swarm optimization, klasifikasi status sosial ekonomi keluarga.*

ABSTRACT

Ika Retno Putri. 2014. Optimization of An Adaptive Fuzzy k-Nearest Neighbor Method based on Particle Swarm Optimization for the classification of socioeconomic status. Undergraduate Thesis. Informatics / Computer Science Department, Information Technology and Computer Science Program (PTIIK). Advisor: Imam Cholissodin, S.Si., M.Kom. dan Budi Darma S., S.Kom., M.Cs.

Research is aimed for the classification of socioeconomic status in class families at 3 classes, which is upper class, intermediate class, and lower class in terms of aspects of education, work, income, ownership of property, and the type of dwelling place by using an adaptive fuzzy k-nearest neighbor (FKNN) method based on particle swarm optimization (PSO), where an adaptive FKNN method used in the process of PSO for calculate the value of fitness of particles and in testing the result of the implementation of an adaptive FKNN method based on PSO. The presence of time variant of PSO process is time varying acceleration coefficients (TVAC) and time varying inertia weight (TVIW), so this method named time variant particle swarm optimization-fuzzy k-nearest neighbor (TVPSO-FKNN). TVPSO-FKNN used to optimize FKNN to performing the process of classification, where optimal value m , k , and features for the classification of FKNN. In the 5 scenarios testing obtained average value of accuracy reached 91.31 % in testing the number of iterations. A method of TVPSO-FKNN proven optimal for improving the performance of the process of FKNN classification in 3 class classification.

Keyword: Fuzzy k-nearest neighbor, time variant, particle swarm optimization, classification of socioeconomic status in class families.

DAFTAR ISI

LEMBAR PERSETUJUAN	ii
LEMBAR PENGESAHAN	iii
PERNYATAAN.....	iv
ORISINALITAS SKRIPSI.....	iv
KATA PENGANTAR.....	v
ABSTRAK	vii
ABSTRACT	viii
DAFTAR ISI.....	ix
DAFTAR TABEL	xiv
DAFTAR GAMBAR.....	xvii
DAFTAR SOURCE CODE	xix
DAFTAR LAMPIRAN	xx
BAB I	1
PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	5
1.3 Batasan Masalah.....	5
1.4 Tujuan Penelitian	6
1.5 Manfaat Penelitian	6
1.6 Sistematika Penulisan	6
BAB II	8
KAJIAN PUSTAKA DAN DASAR TEORI.....	8
2.1 Kajian Pustaka.....	8
2.2 Status Sosial Ekonomi Keluarga.....	13
2.2.1 Definisi Status Sosial Ekonomi Keluarga	13
2.1.2 Klasifikasi Status Sosial Ekonomi Keluarga.....	15
2.3 Logika Fuzzy	18
2.4 Data Mining	20
2.5 Klasifikasi	23
2.6 Jenis Data dan Cara Konversi Data.....	24

2.6.1 Data Berdasarkan Sumbernya	24
2.6.2 Data Berdasarkan Bentuk dan Sifatnya.....	24
2.6.3 Konversi Data Kualitatif ke Kuantitatif dan Teknik Pengolahan Data	26
2.7 Normalisasi Data.....	28
2.8 <i>k- Nearest Neighbor</i> (k-NN)	29
2.9 <i>Fuzzy k-Nearest Neighbor</i> (FKNN)	31
2.10 <i>Particle Swarm Optimization</i> (PSO).....	33
2.10.1 Definisi <i>Particle Swarm Optimization</i> (PSO).....	33
2.10.2 Algoritma PSO	35
2.10.3 Implementasi <i>Particle Swarm Optimization</i> (PSO)	36
2.11 <i>Time Variant Particle Swarm Optimization</i> (TVPSO)	38
2.12 Langkah-Langkah Model TVPSO-FKNN Berdasarkan Algoritma PSO Serial	41
2.13 <i>K-Fold Cross Validation</i> (CV).....	44
2.14 Akurasi	45
2.15 <i>Receiver Operating Characteristic</i> (ROC) <i>Curve</i>	49
2.16 <i>Area Under ROC Curve</i> (AUC).....	50
BAB III.....	52
METODOLOGI PENELITIAN DAN PERANCANGAN	52
3.1 Metodologi Penelitian	52
3.1.1 Mempelajari Literatur	53
3.1.2 Pengumpulan Data	54
3.1.3 Analisis dan Perancangan Sistem.....	55
3.1.4 Implementasi Sistem	56
3.1.5 Pengujian dan Analisis Sistem	56
3.1.6 Evaluasi dan Analisis Hasil.....	56
3.2 Perancangan Sistem	57
3.2.1 Diskripsi Sistem	58
3.2.2 Perancangan Perangkat Lunak	58
3.2.2.1 Proses Normalisasi Data Kali Bobot Atribut.....	60
3.2.2.2 Adaptive Fuzzy K-Nearest Neighbor Berdasarkan Pada Particle Swarm Optimization.....	62

A. Proses Membangkitkan Partikel.....	64
B. Proses <i>K-Fold Cross Validation</i>	65
C. Proses Klasifikasi <i>Fuzzy k-Nearest Neighbor</i> (FKNN) dan menghitung <i>Fitness</i>	67
D. Proses <i>Update</i> Nilai <i>Velocity</i> dan Posisi	71
E. Proses <i>Update Pfit</i> dan <i>Pbest</i>	73
F. Proses <i>Update Gfit</i> dan <i>Gbest</i>	73
3.2.2.3 Pengujian dengan <i>K-Fold Cross Validation</i>	74
3.2.3 Perhitungan Manual	76
3.2.3.1 Normalisasi Data	77
3.2.3.2 Encode Partikel.....	79
3.2.3.3 Train Klasifikasi FKNN	80
A. <i>Fold #1</i> Partikel 1.....	81
B. <i>Fold #1</i> Partikel 2.....	87
C. <i>Fold #1</i> Partikel 3.....	90
3.2.3.4 Metode Adaptive Fuzzy K-Nearest Neighbor Berdasarkan Pada Particle Swarm Optimization.....	93
A. Menghitung Nilai AUC dan Nilai <i>Fitness</i> Tiap Partikel.....	93
B. Inisialisasi Parameter	94
C. Proses TVPSO <i>Fold #1</i> Iterasi 1	95
D. Proses TVPSO <i>Fold #1</i> Iterasi 2	98
3.2.4 Perancangan <i>User Interface</i>	107
3.2.5 Perancangan Uji Coba dan Evaluasi	111
3.2.5.1 Pengujian Pengaruh Interval Nilai <i>m</i> Terhadap Rata-rata Akurasi.....	111
3.2.5.2 Pengujian Pengaruh Interval Nilai <i>k</i> Terhadap Rata-rata Akurasi.....	112
3.2.5.3 Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi.....	112
3.2.5.4 Pengujian Pengaruh Banyaknya Iterasi Terhadap Rata-rata Akurasi.....	113
3.2.5.5 Pengujian Pengaruh Nilai <i>K</i> pada <i>K-Fold CV</i> Terhadap Rata-rata Akurasi.....	113
3.2.5.6 Pengujian untuk Mengukur Konvergensi pada TVPSO.....	114
BAB IV	115

IMPLEMENTASI	115
4.1 Lingkungan Implementasi.....	115
4.1.1 Lingkungan Perangkat Keras	115
4.1.2 Lingkungan Perangkat Lunak	115
4.2 Implementasi Program	116
4.2.1 Proses Normalisasi Dikalikan Bobot Atribut.....	116
4.2.1.1 Proses Load Data dan Bobot Atribut.....	116
4.2.1.2 Proses Normalisasi	117
4.2.1.3 Proses Normalisasi Kali Bobot Atribut	119
4.2.2 Proses TVPSO-FKNN	120
4.2.2.1 Proses Membangkitkan Partikel	120
4.2.2.2 Proses K-Fold Cross Validation	122
4.2.2.3 Proses Inisialisasi Parameter	124
4.2.2.4 Proses Train FKNN	125
4.2.2.5 Proses Hitung Area Under ROC Curve.....	129
4.2.2.6 Proses Hitung Nilai Fitness	130
4.2.2.7 Proses Update Velocity dan Posisi	131
4.2.2.8 Proses Update Pbest dan Gbest	133
4.2.3 Proses Uji dengan <i>K-fold Cross Validation</i>	134
4.2.4 Proses Menampilkan Hasil Akurasi dari Uji dengan <i>K-fold Cross Validation</i>	136
4.3 Implementasi <i>Interface</i>	137
4.3.1 <i>Form</i> Hasil Normalisasi Kali Bobot Atribut atau Fitur	137
4.3.2 <i>Form</i> Hasil Membangkitkan Partikel.....	138
4.3.3 <i>Form</i> Hasil Proses TVPSO-FKNN	139
4.3.4 <i>Form</i> Hasil Proses Uji Hasil TVPSO-FKNN	140
4.3.5 <i>Form</i> Hasil Akurasi.....	141
BAB V.....	142
PENGUJIAN DAN ANALISIS	142
5.1 Pengujian.....	142
5.2 Hasil Pengujian	142
5.2.1 Pengujian Pengaruh Interval Nilai <i>m</i> Terhadap Rata-rata Akurasi	143

5.2.2 Pengujian Pengaruh Interval Nilai k Terhadap Rata-rata Akurasi .	143
5.2.3 Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi	144
5.2.4 Pengujian Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi...	145
5.2.5 Pengujian Pengaruh Nilai K Terhadap Rata-rata Akurasi.....	145
5.2.6 Pengujian untuk Mengetahui Konvergensi pada TVPSO.....	146
5.3 Analisis Hasil Pengujian	147
5.3.1 Analisis Hasil Pengujian Pengaruh Nilai Interval m Terhadap Rata-rata Akurasi	147
5.3.2 Analisis Hasil Pengujian Pengaruh Nilai Interval k Terhadap Rata-rata Akurasi	148
5.3.3 Analisis Hasil Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi	149
5.3.4 Analisis Hasil Pengujian Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi	150
5.3.5 Analisis Hasil Pengujian Pengaruh Nilai K Terhadap Rata-rata Akurasi	151
5.3.6 Analisis Hasil Pengujian Mengukur Konvergensi pada TVPSO...	152
BAB VI	154
PENUTUP.....	154
6.1 Kesimpulan	154
6.2 Saran.....	155
DAFTAR PUSTAKA.....	DP-1
LAMPIRAN.....	L-1

DAFTAR TABEL

Tabel 2.1 <i>Confusion Matrix</i> untuk 2 Kelas (Makhtar, 2011)	46
Tabel 2.2 <i>Confusion Matrix</i> untuk 3 Kelas (Hermaduanti dan Dewi, 2008)	47
Tabel 2.3 Tabel Kategori Pengklasifikasi Berdasarkan Nilai AUC (Mohanty at all, 2011).....	51
Tabel 3.1 Bobot Atribut.....	76
Tabel 3.2 Sampel <i>Dataset</i>	77
Tabel 3.3 <i>Range</i> Normalisasi	78
Tabel 3.4 Hasil Normalisasi	78
Tabel 3.5 Hasil Normalisasi Dikalikan Bobot Atribut	79
Tabel 3.6 Inisialisasi <i>Encode</i> Partikel untuk <i>Fold</i> #1.....	80
Tabel 3.7 Inisialisasi <i>Encode</i> Partikel untuk <i>Fold</i> #2.....	80
Tabel 3.8 Inisialisasi <i>Encode</i> Partikel untuk <i>Fold</i> #3.....	80
Tabel 3.9 <i>Data Training Fold</i> #1 Partikel 1 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut	81
Tabel 3.10 <i>Data Testing Fold</i> #1 Partikel 1 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut	81
Tabel 3.11 Nilai Hasil dari Perhitungan Jarak <i>Fold</i> #1 Partikel 1 dengan <i>Euclidean Distance</i> Sebelum Diurutkan.....	82
Tabel 3.12 Nilai Hasil dari Perhitungan Jarak <i>Fold</i> #1 Partikel 1 dengan <i>Euclidean Distance</i> Setelah Diurutkan.....	83
Tabel 3.13 Nilai Keanggotaan <i>Fold</i> #1 Kelas ke- <i>i</i> Terhadap Tetangga ke- <i>j</i>	85
Tabel 3.14 Hasil FKNN <i>Fold</i> #1 Partikel 1.....	86
Tabel 3.15 <i>Confusion Matrix</i>	86
Tabel 3.16 Hasil <i>Confusion Matrix</i> Partikel 1.....	87
Tabel 3.17 <i>Data Training Fold</i> #1 Partikel 2 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut	88
Tabel 3.18 <i>Data Testing Fold</i> #1 Partikel 2 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut	88
Tabel 3.19 Nilai Hasil dari Perhitungan Jarak <i>Fold</i> #1 Partikel 2 dengan <i>Euclidean Distance</i> Setelah Diurutkan.....	88
Tabel 3.20 Hasil FKNN <i>Fold</i> #1 Partikel 2.....	89
Tabel 3.21 Hasil <i>Confusion Matrix</i> Partikel 2.....	90

Tabel 3.22 <i>Data Training Fold #1 Partikel 3 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut</i>	90
Tabel 3.23 <i>Data Testing Fold #1 Partikel 3 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut</i>	91
Tabel 3.24 Nilai Hasil dari Perhitungan Jarak <i>Fold #1 Partikel 3 dengan Euclidean Distance</i> setelah Diurutkan	91
Tabel 3.25 Hasil FKNN <i>Fold #1 partikel 3</i>	92
Tabel 3.26 Hasil <i>Confusion Matrix</i> Partikel 3	92
Tabel 3.27 Hasil Nilai <i>Fitness</i> Partikel pada <i>Fold #1</i>	94
Tabel 3.28 Posisi Awal, <i>Pbest</i> dan <i>Gbest</i> Tiap Partikel pada <i>Fold #1</i>	95
Tabel 3.29 Nilai <i>Velocity</i> Iterasi 1	96
Tabel 3.30 Nilai <i>Sigmoid</i> Iterasi 1	96
Tabel 3.31 Nilai Random Iterasi 1	97
Tabel 3.32 Hasil <i>Update</i> Posisi Partikel pada Iterasi 1 atau $X_{ij}(1)$	97
Tabel 3.33 Hasil Nilai <i>Fitness</i> Partikel pada <i>Fold #1</i> Iterasi 1	98
Tabel 3.34 <i>Pbest</i> dan <i>Gbest</i> Partikel pada <i>Fold #1</i> Iterasi 1	98
Tabel 3.35 Nilai <i>Velocity</i> Iterasi 2	99
Tabel 3.36 Nilai <i>Sigmoid</i> Iterasi 2	99
Tabel 3.37 Nilai Random Iterasi 2	99
Tabel 3.38 Hasil <i>Update</i> Posisi Partikel pada Iterasi 2 atau $X_{ij}(2)$	99
Tabel 3.39 Hasil Nilai <i>Fitness</i> Partikel pada <i>Fold #1</i> Iterasi 2	100
Tabel 3.40 <i>Pbest</i> dan <i>Gbest</i> Partikel pada <i>Fold #1</i> Iterasi 2	100
Tabel 3.41 <i>Pbest</i> dan <i>Gbest</i> Partikel pada <i>Fold #2</i> Iterasi 2	100
Tabel 3.42 <i>Pbest</i> dan <i>Gbest</i> Partikel pada <i>Fold #3</i> Iterasi 2	101
Tabel 3.43 Hasil TVPSO-FKNN dengan 3-Fold CV	101
Tabel 3.44 Hasil FKNN dan <i>Confusion Matrix Fold 1</i> Partikel <i>Fold #1</i>	101
Tabel 3.45 Hasil FKNN dan <i>Confusion Matrix Fold 2</i> Partikel <i>Fold #1</i>	102
Tabel 3.46 Hasil FKNN dan <i>Confusion Matrix Fold 3</i> Partikel <i>Fold #1</i>	102
Tabel 3.47 Hasil Akurasi 3 –Fold CV dengan Partikel Terbaik <i>Gbest Fold #1</i>	103
Tabel 3.48 Hasil FKNN dan <i>Confusion Matrix Fold 1</i> Partikel <i>Fold #2</i>	103
Tabel 3.49 Hasil FKNN dan <i>Confusion Matrix Fold 2</i> Partikel <i>Fold #2</i>	104
Tabel 3.50 Hasil FKNN dan <i>Confusion Matrix Fold 3</i> Partikel <i>Fold #2</i>	104
Tabel 3.51 Hasil Akurasi 3 –Fold CV dengan Partikel Terbaik <i>Gbest Fold #2</i>	105
Tabel 3.52 Hasil FKNN dan <i>Confusion Matrix Fold 1</i> Partikel <i>Fold #3</i>	105

Tabel 3.53 Hasil FKNN dan <i>Confusion Matrix Fold 2</i> Partikel <i>Fold #3</i>	105
Tabel 3.54 Hasil FKNN dan <i>Confusion Matrix Fold 3</i> Partikel <i>Fold #3</i>	106
Tabel 3.55 Hasil Akurasi 3 – <i>Fold CV</i> dengan Partikel Terbaik <i>Gbest Fold #3</i>	106
Tabel 3.56 Hasil Rata-rata Akurasi 3 Partikel Terbaik (<i>Gbest</i>)	107
Tabel 3.57 Pengaruh Nilai <i>m</i> Terhadap Rata-rata Akurasi.....	111
Tabel 3.58 Pengaruh Nilai <i>k</i> Terhadap Rata-rata Akurasi.....	112
Tabel 3.59 Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi	112
Tabel 3.60 Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi	113
Tabel 3.61 Pengaruh Nilai <i>K</i> Terhadap Rata-rata Akurasi.....	114
Tabel 3.62 Hasil Mengukur Konvergensi pada TVPSO	114
Tabel 5.1 Pengujian Pengaruh Nilai <i>m</i> terhadap Rata-rata Akurasi.....	143
Tabel 5.2 Pengaruh Nilai <i>k</i> Terhadap Rata-rata Akurasi.....	144
Tabel 5.3 Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi	144
Tabel 5.4 Pengaruh Jumlah Iterasi Terhadap Akurasi.....	145
Tabel 5.5 Pengaruh Nilai <i>K</i> Terhadap Rata-rata Akurasi.....	146
Tabel 5.6 Hasil Mengukur Konvergensi pada TVPSO	146



DAFTAR GAMBAR

Gambar 2.1 Perbandingan <i>F-measure</i> Pada Metode Pencarian Jarak Untuk Kategori <i>Action</i>	11
Gambar 2.2 Tahapan dalam KDD	22
Gambar 2.3 <i>Pseudo code</i> Algoritma PSO	36
Gambar 2.4 Bentuk dari <i>Confusion Matrix</i>	45
Gambar 2.5 Perbandingan Bentuk-bentuk Kurva ROC untuk Kepentingan Diagnostik.....	50
Gambar 2.6 Perbandingan Dua Buah Kurva ROC A dan B untuk Membedakan Nilai AUC.....	50
Gambar 3.1 Diagram Alir Penelitian	53
Gambar 3.2 Diagram Alir Tahap Perancangan Sistem.....	57
Gambar 3.3 Diagram Alir Sistem	59
Gambar 3.4 Diagram Alir Proses Normalisasi Dikalikan Bobot Atribut	61
Gambar 3.5 Diagram Alir Proses TVPSO-FKNN.....	62
Gambar 3.6 Diagram Alir Proses Membangkitkan Partikel.....	65
Gambar 3.7 Diagram Alir Proses <i>K-Fold Cross Validation</i>	66
Gambar 3.8 Diagram Alir Proses FKNN.....	68
Gambar 3.9 Diagram Alir Proses Hitung Jarak dengan <i>Euclidean Distance</i>	69
Gambar 3.10 Diagram Alir Proses Hitung Nilai <i>Fitness</i>	70
Gambar 3.11 Diagram Alir Proses <i>Update</i> Nilai <i>Velocity</i> dan Posisi	72
Gambar 3.12 Diagram Alir Proses <i>Update Pfit</i> dan <i>Pbest</i>	73
Gambar 3.13 Diagram Alir Proses <i>Update Gfit</i> dan <i>Gbest</i>	74
Gambar 3.14 Diagram Alir Proses Pengujian dengan <i>K-Fold Cross Validation</i>	75
Gambar 3.15 Kurva ROC	93
Gambar 3.16 <i>Interface</i> Proses Normalisasi Dikalikan Bobot Atribut	108
Gambar 3.17 <i>Interface</i> Proses Membangkitkan Partikel.....	108
Gambar 3.18 <i>Interface</i> Proses TVPSO-FKNN.....	109
Gambar 3.19 <i>Interface</i> Proses Uji Hasil TVPSO-FKNN	110
Gambar 3.20 <i>Interface</i> Hasil Akurasi.....	110

Gambar 4.1	Tampilan Utama Tab 1 <i>Form</i> Hasil Hitung Normalisasi Kali Bobot Atribut atau Fitur	138
Gambar 4.2	Tampilan Tab 2 <i>Form Input</i> Parameter dan Hasil Membangkitkan Partikel.....	139
Gambar 4.3	Tampilan Tab 3 <i>Form</i> Hasil Proses TVPSO-FKNN	140
Gambar 4.4	Tampilan Tab 4 <i>Form</i> Hasil Proses Uji Hasil TVPSO-FKNN Hitung Akurasi.....	141
Gambar 4.5	Tampilan Tab 5 <i>Form</i> Hasil Akurasi	141
Gambar 5.1	Grafik hasil rata-rata akurasi pengujian nilai interval m	148
Gambar 5.2	Grafik hasil rata-rata akurasi pengujian nilai interval k	149
Gambar 5.3	Grafik hasil rata-rata akurasi pengujian jumlah partikel	150
Gambar 5.4	Grafik hasil rata-rata akurasi pengujian jumlah iterasi	151
Gambar 5.5	Grafik hasil rata-rata akurasi pengujian nilai K	152
Gambar 5.6	Grafik hasil rata-rata akurasi pengujian nilai K	153



DAFTAR SOURCE CODE

Source Code 4.1	Listing Code Proses Load Data dan Bobot Atribut.....	117
Source Code 4.2	Listing Code Proses Normalisasi.	118
Source Code 4.3	Listing Code Proses Normalisasi Dikalikan Bobot Atribut.	120
Source Code 4.4	Listing Code Proses Membangkitkan Partikel.....	121
Source Code 4.5	Listing Code Proses K-Fold Cross Validation.....	124
Source Code 4.6	Listing Code Proses Inisialisasi Parameter.	124
Source Code 4.7	Listing Code Proses Train FKNN.....	128
Source Code 4.8	Listing Code Proses Hitung Area Under ROC Curve.....	130
Source Code 4.9	Listing Code Proses Hitung Nilai Fitness.	131
Source Code 4.10	Listing Code Proses Update Velocity dan Posisi.....	132
Source Code 4.11	Listing Code Proses Update Pbest dan Gbest.	133
Source Code 4.12	Listing Code Proses Uji Hasil FKNN dan Hitung Akurasi.	135
Source Code 4.13	Listing Code Proses Menampilkan Hasil Akurasi.	136

DAFTAR LAMPIRAN

Lampiran 1. Form Survey	L-1
Lampiran 2. Surat Keterangan dari RT	L-3
Lampiran 3. Data KK	L-4
Lampiran 4. Sampel <i>Dataset</i> pada Perhitungan Manual	L-9
Lampiran 5. <i>Range</i> Normalisasi pada Perhitungan Manual	L-10
Lampiran 6. Hasil Normalisasi pada Perhitungan Manual	L-10
Lampiran 7. Hasil Normalisasi Dikalikan Bobot pada Perhitungan Manual	L-11
Lampiran 8. Keterangan Nilai Atribut	L-12
Lampiran 9. Detail Hasil Pengujian Pengaruh Interval m dengan 121 Data....	L-13
Lampiran 10. Detail Hasil Pengujian Pengaruh Interval k dengan 121 Data...	L-20
Lampiran 11. Detail Hasil Pengujian Pengaruh Jumlah Partikel dengan 121 Data	L-27
Lampiran 12. Detail Hasil Pengujian Pengaruh Jumlah Iterasi dengan 121 Data .	L-34
Lampiran 13. Detail Hasil Pengujian Pengaruh Nilai K dengan 121 Data	L-39
Lampiran 14. Detail Hasil Pengujian Diluar Skenario dengan 20 Partikel dan 100 Iterasi pada 121 Data	L-45
Lampiran 15. Data Hasil Pengujian Konvergensi	L-46



BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi yang berkembang saat ini telah memasuki berbagai aspek kehidupan di dunia. Salah satunya dalam bidang komputer seperti sistem informasi, aplikasi penghitung, aplikasi *mobile*, pengolahan data dan lain sebagainya. Dimana sistem yang terkomputerisasi saat ini telah banyak dikembangkan sebagai alternatif untuk membantu menyelesaikan permasalahan yang ada atau membantu dalam proses pengambilan keputusan guna meningkatkan kinerja suatu organisasi atau instansi khususnya yang baru berkembang. Karena dengan menggunakan teknologi komputer dirasa akan lebih efisien dan efektif dalam proses pengolahan dan penyajian data untuk memberikan informasi yang lebih akurat.

Perkembangan teknologi komputer tersebut telah diarahkan pada banyak instansi pemerintah, salah satunya adalah Kelurahan Desa Jugo Kecamatan Kesamben Kabupaten Blitar. Informasi yang telah diperoleh dari hasil survey wawancara terhadap Kepala Desa Jugo diketahui bahwa faktanya pada Kelurahan Jugo pelayanannya terkait dengan pendataan dan klasifikasi status sosial ekonomi warganya masih dilakukan secara manual atau belum dilakukan secara otomatis, sehingga prosesnya kurang efisien yaitu lama, dimana jumlah warganya kurang lebih 2363 KK (Kepala Keluarga) pada tahun 2012.

Kemudian proses klasifikasi untuk penentuan status sosial ekonomi warganya dilakukan hanya pada 1 kelas dan proses tersebut dilakukan saat diperlukan saja. Misalnya apabila ada bantuan dari pihak pemerintah pihak kelurahan melakukan pendataan dan klasifikasi pada warganya yang tergolong kelas menengah ke bawah, sedangkan apabila pihak kelurahan akan mencari dana bantuan untuk pembangunan desa, maka pihak kelurahan melakukan pendataan dan mengklasifikasikan warganya yang tergolong kelas menengah ke atas. Sehingga pihak kelurahan tidak dapat mengetahui secara pasti dan akurat mengenai warganya yang tergolong kelas atas, menengah, atau bawah.

Berkaitan dengan penggolongan di atas, guna meningkatkan kinerja pemerintah dalam mengontrol kesejahteraan masyarakatnya yang kenyataannya mengalami banyak kendala, diantaranya adalah pengontrolan yang masih dilakukan dalam waktu yang tidak tentu yaitu terkadang sampai beberapa tahun baru dilakukan. Hal ini terjadi karena belum adanya sistem yang mendukung kinerja perangkat desa agar lebih efektif dan efisien dalam menggolong-golongkan tiap-tiap keluarga. Kelurahan Jugo saat ini masih melakukan dokumentasi tidak secara mendetail yaitu hanya berdasarkan pada prosentase saja. Sehingga tingkat kesejahteraan warganya tidak terkontrol dengan baik, misalnya warga yang memiliki persawahan hanya berapa persen dari keseluruhan warganya.

Selain itu Kelurahan Desa Jugo juga wajib memberikan pelayanan contohnya dalam membuat surat keterangan atau surat pengajuan seperti PKH (Program Keluarga Harapan), Jamkesmas (Jaminan Kesehatan Masyarakat), Jamkesda (Jaminan Kesehatan Daerah), dan lain sebagainya yang berkaitan dengan status sosial ekonomi warga masyarakatnya. Namun belum ada dasar yang pasti dan sesuai untuk membantu pihak kelurahan dalam meningkatkan pelayanannya terkait pembuatan surat keterangan atau surat pengajuan tersebut, dimana terkadang ada warga yang sebenarnya tergolong sosial ekonominya menengah atas meminta dibuatkan surat keterangan tidak mampu untuk kepentingan tertentu. Disini perangkat desa tidak memiliki dasar yang pasti sehingga tidak sesuai dengan kenyataannya. Maka dari itu untuk meningkatkan kinerja pemerintahan yang lebih efektif dan efisien dalam melayani masyarakatnya, khususnya untuk proses klasifikasi status sosial ekonomi warganya dibutuhkan suatu aplikasi atau sistem yang terkomputerisasi.

Klasifikasi merupakan salah satu alternatif yang digunakan untuk mengelompokkan sebuah objek ke dalam satu atau lebih kelompok berdasarkan variabel yang diamati. Banyak sekali metode klasifikasi yang ada diantaranya adalah Fuzzy K-Nearest Neighbour (FKNN), K-Nearest Neighbour (K-NN), Adaptive K-Nearest Neighbour (AKNN), Support Vector Machine (SVM), Backpropagation dan lain sebagainya.

Penelitian sebelumnya adalah penelitian yang menggunakan metode *k-nearest neighbor* (K-NN) untuk mengklasifikasikan data yang berupa dokumen *e-*

complaint (Baharsyah, 2014). Dari hasil pengujian tersebut dihasilkan nilai akurasi tertinggi sebesar 78.0952% dan tanpa *stemming* sebesar 73.3333%. Selain itu adalah penelitian yang mengembangkan metode K-NN dengan mengkombinasikan dengan logika *fuzzy* (FKNN) pada klasifikasi jenis mangga berdasarkan tekstur daun (Mubarok, 2013) dengan hasil rata-rata akurasi 81.1574%, selain itu untuk klasifikasi *genre* film dengan data dokumen text judul dan sinopsis film (Prasetyo, 2014) menghasilkan akurasi tertinggi dari pemberian nilai k pada rentang 10 sampai 20 dan penggunaan metode *cosine similarity* menghasilkan tingkat akurasi yang lebih baik dibandingkan dengan *euclidean distance*. Penelitian lain yang membandingkan kedua metode klasifikasi K-NN dan FKNN untuk diagnosis penyakit diabetes mellitus (Selly, 2013), dari pengujian yang dilakukan nilai akurasi maksimum diperoleh FKNN mencapai 98% pada k=15 sedangkan K-NN hanya mencapai 96% pada k=11. Sehingga terbukti bahwa FKNN lebih unggul dibandingkan K-NN. Hal tersebut dikarenakan metode tersebut mampu mempertimbangkan sifat ambigu dari tetangganya agar tetangga yang ambigu tidak melakukan peranan penting dalam klasifikasi. Penggunaan nilai derajat keanggotaan pada setiap kelas akan memberikan kekuatan suatu *instance* berada pada suatu kelas, selain itu pemberian nilai *membership* digunakan sebagai fungsi pola jarak atau kesamaan dari sejumlah himpunan K-NN dan pemberian nilai keanggotaan tetangga pada kelas tertentu sehingga *data testing* yang akan diklasifikasikan akan memiliki nilai keanggotaan pada semua kelas.

Penelitian yang lain yaitu model *diffusion decision making* untuk klasifikasi algoritma *adaptive k-nearest neighbor* (Yung-Kyun at all, 2012). Klasifikasi dilakukan pada 2 kelas *convidence* dan tidak *convidence* pada dua proses Poisson menghasilkan maksimum rata-rata kinerja sampai 0.721 pada k = 3, yang mana jauh lebih kecil dari akurasi keseluruhan, karena jarak informasi yang rendah untuk dimensi ruang yang tinggi. Potensi manfaat dari penelitian ini adalah untuk meningkatkan kinerja klasifikasi berdasarkan komputasi paralel pada *dataset* yang sangat besar. Pada penelitian yang lain yaitu penggunaan metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Parallel Particle Swarm Optimization* untuk prediksi kebangkrutan, klasifikasinya terdiri dari dua kelas yaitu bangkrut dan tidak bangkrut (Chen at all, 2011). Penelitian tersebut bertujuan untuk mengeksplorasi

seluruh potensi dari metode FKNN dengan menentukan k dan m dengan memanfaatkan maksimum akurasi klasifikasi untuk prediksi kebangkrutan secara otomatis dengan menggunakan pendekatan TVPSO. Dari proses pengujian dihasilkan tingkat akurasi PTVPSO-FKNN lebih baik dibandingkan MLP, KNN, dan SVM yaitu 81.67%, selain itu penggunaan model seleksi fitur dapat memperbaiki rata-rata akurasi lebih baik dibandingkan tanpa seleksi fitur, pada pengujian dengan membandingkan antara model paralel dan seri pada *platform multi-core* yang menunjukkan bahwa dengan paralel kinerja CPU jauh lebih cepat dibanding seri. Selain itu model paralel ini hanya dapat digunakan pada sebuah komputer atau laptop dengan spesifikasi prosesor *quad-core*.

Berdasarkan dari latar belakang di atas dijelaskan bahwa FKNN merupakan metode yang telah terbukti dapat meningkatkan akurasi yang lebih baik, karena metode tersebut mampu mempertimbangkan sifat ambigu dari tetangganya agar tetangga yang ambigu tidak melakukan peranan penting dalam klasifikasi, hal tersebut juga dipengaruhi oleh penggunaan derajat keanggotaan pada setiap kelas untuk memberikan kepercayaan atau kekuatan suatu *instance* berada pada suatu kelas (Mubarok, 2013). Untuk meningkatkan kinerja klasifikasi telah banyak dilakukan penelitian dengan mengkombinasikan metode klasifikasi dengan aturan *adaptive* diantaranya adalah penggunaan model *diffusion decision making* (Yung-Kyun at all, 2012) dan berdasarkan pada metode *particle swarm optimization* (PSO) (Chen at all, 2011). PSO dirasa lebih cocok digunakan dalam mengoptimalkan kinerja klasifikasi pada metode FKNN, karena telah terbukti dapat mengeksplorasi seluruh potensi dari metode FKNN dengan menentukan k dan m dengan memanfaatkan maksimum akurasi klasifikasi untuk prediksi kebangkrutan secara otomatis dengan menggunakan pendekatan TVPSO yang juga dapat membantu meningkatkan kinerja klasifikasi secara optimal (Chen at all, 2011). Sehingga penulis mengambil judul “**Optimasi Metode Adaptive Fuzzy K-Nearest Neighbor dengan Particle Swarm Optimization untuk Klasifikasi Status Sosial Ekonomi Keluarga**”. Dimana implementasi tersebut dijalankan untuk mengklasifikasikan status sosial ekonomi keluarga ke dalam kelas atas, menengah, atau bawah. Dimana metode ini memadukan metode K-NN dengan aturan fuzzy dan aturan *adaptive* berdasarkan pendekatan PSO untuk mengoptimalkan kinerja pada saat proses

klasifikasi. Harapannya sistem ini dapat membantu mengatasi permasalahan yang ada khususnya pada proses klasifikasi data keluarga sehingga dapat meningkatkan kinerja perangkat desa di Kelurahan Desa Jugo.

1.2 Rumusan Masalah

Adapun Rumusan Masalah yang ada pada penelitian ini berdasarkan latar belakang diatas adalah:

1. Bagaimana implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga di Dusun Jugo Desa Jugo Kecamatan Kesamben Kabupaten Blitar?
2. Bagaimana tingkat akurasi hasil klasifikasi dengan metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization*?

1.3 Batasan Masalah

Dalam menghindari pembahasan yang tidak terfokus pada tujuan maka penulis membuat batasan-batasan ruang lingkup sebagai berikut:

1. Aplikasi ini dibuat hanya untuk klasifikasi status sosial ekonomi keluarga dengan data yang diambil dari hasil *survey* 4 RT di Dusun Jugo Desa Jugo Kecamatan Kesamben Kabupaten Blitar yaitu RT 001, 003, 004, dan 005.
2. Adapun data yang digunakan sebanyak 121 data (berdasarkan hasil wawancara dan diskusi dengan Kepala Desa Jugo dihasilkan 24 data yang tergolong kelas atas, 52 kelas menengah, dan 45 kelas bawah). Atribut yang digunakan 9 yang berisi data kuantitatif dan 5 yang berisi data kualitatif.
3. Metode yang digunakan adalah *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization*.
4. Hasil klasifikasi dari penerapan metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization* digunakan untuk menentukan keluarga yang tergolong pada status sosial ekonomi kelas atas, menengah dan bawah.
5. Pengujian hanya fokus pada pengujian metode, tidak pada pengujian kinerja hardware.

1.4 Tujuan Penelitian

Adapun tujuan yang ada pada penelitian ini dilihat dari rumusan masalahnya adalah:

1. Mengimplementasikan metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga di Dusun Jugo Desa Jugo Kecamatan Kesamben Kabupaten Blitar.
2. Mengukur tingkat akurasi hasil klasifikasi dengan metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization*.

1.5 Manfaat Penelitian

Harapannya dalam penelitian ini dapat memberikan manfaat bagi semua pihak yang terkait. Adapun manfaat yang diperoleh dari pengimplementasian metode *Adaptive Fuzzy K-Nearest Neighbor* dengan *Particle Swarm Optimization* ini adalah:

1. Membantu meningkatkan kinerja perangkat desa terkait proses klasifikasi agar lebih efektif dan efisien yaitu hasilnya lebih akurat, obyektif dan proses klasifikasi berjalan lebih cepat, sehingga dapat membantu dalam mengontrol kesejahteraan masyarakat di desa setempat.
2. Dapat digunakan sebagai salah satu alternatif yang memudahkan untuk penyaluran dana bantuan dari pemerintah agar tepat sasaran. Selain itu memudahkan dalam pencarian bantuan apabila ada perbaikan fasilitas desa oleh pihak kelurahan.
3. Mempermudah suatu keluarga jika ingin mengajukan atau mendaftarkan diri dan membuat surat keterangan bagi warga masyarakat yang membutuhkan terkait dengan status sosial ekonomi keluarganya yang tepat dan sesuai.
4. Membantu pemerintah untuk meningkatkan kepercayaan warga masyarakatnya yaitu untuk mengetahui pendataan dan penyaluran bantuan yang lebih transparan dari hasil klasifikasi yang lebih akurat.

1.6 Sistematika Penulisan

Dalam penulisan skripsi ini terdiri dari 6 bab dan beberapa sub bagian. Untuk memudahkan dalam penulisan dan pembahasan pada tiap-tiap bab agar

sesuai dengan rumusan masalah yang ada maka digunakan sistematika penulisan seperti dibawah ini.

BAB 1: PENDAHULUAN

Didalam bab pendahuluan berisi pembahasan secara umum mengenai permasalahan yang akan diangkat mencakup latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

BAB 2: KAJIAN PUSTAKA DAN DASAR TEORI

Didalam bab kajian pustaka dan dasar teori berisi mengenai penjelasan-penjelasan teori-teori dan konsep dasar yang berkaitan dengan implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga.

BAB 3: METODOLOGI PENELITIAN DAN PERANCANGAN

Didalam bab metodologi penelitian dan perancangan berisi mengenai penjelasan metode yang digunakan untuk menyelesaikan permasalahan yang diangkat dalam penelitian skripsi ini.

BAB 4: IMPLEMENTASI

Didalam bab implementasi menguraikan tentang gambaran serta proses implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga, dimana pada implementasi ini juga disertai dengan *source code* dan *screenshot* aplikasi yang dibuat.

BAB 5: PENGUJIAN DAN ANALISIS

Didalam bab pengujian berisi mengenai hasil pengujian dari implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga yang telah di buat. Selanjutnya menganalisa dari hasil pengujian tersebut untuk mengetahui tingkat akurasi penerapan metodenya.

BAB 6: PENUTUP

Didalam bab penutup berisi mengenai uraian tentang kesimpulan dan saran dari implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk klasifikasi status sosial ekonomi keluarga.

BAB II

KAJIAN PUSTAKA DAN DASAR TEORI

Pada bab ini berisi mengenai kajian pustaka dan dasar teori yang digunakan dalam skripsi ini. Kajian pustaka membahas mengenai penelitian-penelitian yang sebelumnya telah ada yang ada kaitannya dengan penyusunan skripsi ini. Sedangkan untuk dasar teori disusun berdasarkan dari latar belakang dan rumusan terkait apapun teori untuk mendasari skripsi ini diantaranya adalah mengenai klasifikasi status sosial ekonomi keluarga, logika *fuzzy*, *data mining*, klasifikasi, konversi data kualitatif ke kuantitatif, normalisasi data, metode *k-nearest neighbor* (K-NN), metode *fuzzy k-nearest neighbor* (FKNN). Teori selanjutnya adalah menjabarkan tentang metode *particle swarm optimization* (PSO), metode *time variant particle swarm optimization* (TVPSO), mengenai langkah langkah kombinasi model TVPSO dengan FKNN berdasarkan algoritma PSO serial, dasar teori untuk pengujiannya menggunakan metode *k-fold cross validation*, *Accuracy*, dan *area under the receiver operating characteristic* (ROC) curve (AUC).

2.1 Kajian Pustaka

Pada kajian pustaka ini membahas mengenai penelitian sebelumnya yang berkaitan dengan skripsi ini. Diantaranya pada penelitian sebelumnya adalah penelitian yang mengklasifikasikan data yang berupa dokumen *e-complaint* dengan menggunakan metode *k-nearest neighbor* (K-NN) dan metode *cosine similarity* untuk menghitung jarak atau nilai kemiripannya dengan jumlah data keseluruhan sebanyak 175 data. Metode tersebut digunakan untuk mengetahui tingkat kesopanan *e-complaint* di Universitas Brawijaya yang pelabelan kesopannya berlabel positif dan 4 level kedalam *sentiment* analisis yang berlabel negatif yaitu level 0, level 1, level 2, dan level 3. Dimana pengujiannya dilakukan sebanyak 5 kali pengujian, 4 kali skenario dengan *stemming* dan tanpa *stemming* pada 1 skenario serta menggunakan 21 data uji dengan data latih dan nilai *k* yang berbeda. Dari hasil pengujian dihasilkan nilai akurasi tertinggi sebesar 78.0952% pada skenario 1 dan tanpa *stemming* sebesar 73.3333%. Sehingga dapat disimpulkan bahwa dengan *stemming* akurasi yang dihasilkan jauh lebih baik dibandingkan

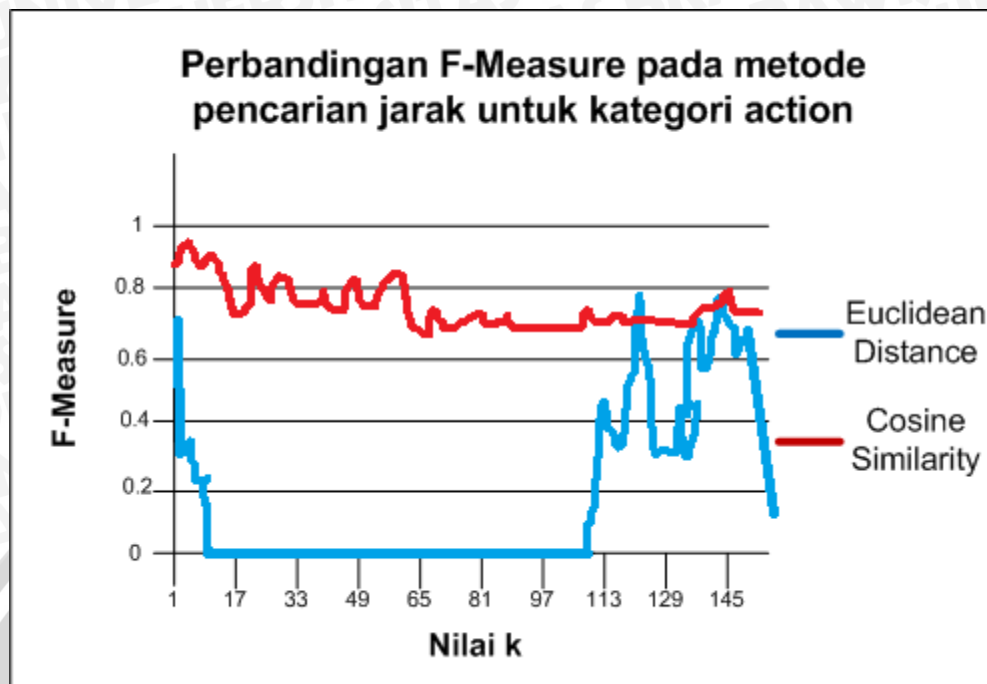
tanpa *stemming*. Namun pada penelitian tersebut tidak dapat menangani jumlah data latih yang lebih kecil dibandingkan data uji sehingga distribusi kurang merata dan mengakibatkan sedikitnya kesamaan kata antara data uji dengan data latih (Baharsyah, 2014).

Penelitian yang lain yaitu model *diffusion decision making* untuk klasifikasi algoritma *adaptive k-nearest neighbor*, dimana model keputusan difusi sebagai strategi optimasi dengan menggunakan aturan *adaptive* yang dibangun untuk menentukan nilai k yang sesuai pada klasifikasi *k-nearest neighbor* dengan memanfaatkan *sequential probability ratio test* (SPRT) dan analisis Bayesian. Pertama 3 kriteria yang berbeda diperoleh berdasarkan uji statistik SPRT. Kedua, berdasarkan analisis Bayesian, dimana 2 probabilitas diperoleh dari kasus yang fungsi *density*-nya lebih besar dari yang lain. Klasifikasi dilakukan pada 2 kelas *convidence* dan tidak *convidence* pada dua proses Poisson. Percobaan dilakukan baik dengan *dataset* sintetis dan *dataset* nyata yang menunjukkan efektivitas dari kriteria klasifikasi tersebut. Sehingga dapat menampilkan hasil rata-rata akurasi dari metode klasifikasi K-NN dan 5 aturan *adaptive* yaitu DV, CDV, PV, DN, dan PN berdasarkan pada pendekatan probabilitas Bayesian dan SPRT yang berhubungan dengan rata-rata nilai tetangga terdekat yang digunakan. Klasifikasi K-NN pada ruang data asli menunjukkan maksimum rata-rata kinerja sampai 0.721 pada $k = 3$, yang mana jauh lebih kecil dari akurasi keseluruhan, karena jarak informasi yang rendah untuk dimensi ruang yang tinggi. Dari beberapa pengujian yang dilakukan dapat disimpulkan bahwa klasifikasi K-NN dalam batas *asymptotic* setara pada *diffusion decision model* untuk membuat keputusan, namun hubungan keduanya belum pernah diteliti sebelumnya. Pada model keputusan difusi level *convidence* berperan pada penentuan penilaian antara kecepatan dan akurasi, selain itu juga untuk mengadaptasikan jumlah tetangga terdekat dalam membuat keputusan klasifikasi yang diterapkan pada klasifikasi K-NN. Potensi manfaat dari penelitian ini adalah untuk meningkatkan kinerja klasifikasi berdasarkan komputasi paralel pada *dataset* yang sangat besar (Yung-Kyun et al., 2012).

Selanjutnya yaitu penelitian yang mengembangkan metode K-NN dengan mengkombinasikan logika fuzzy pada klasifikasi jenis mangga berdasarkan tekstur daun. Adapun data yang digunakan adalah *dataset* numerik citra jenis mangga

gadung dan curut berdasarkan tekstur daun. Metode yang digunakan adalah metode *fuzzy k-nearest neighbor* (FKNN), dimana metode tersebut mampu mempertimbangkan sifat ambigu dari tetangganya agar tetangga yang ambigu tidak melakukan peranan penting dalam klasifikasi. Penggunaan nilai derajat keanggotaan pada setiap kelas akan memberikan kekuatan suatu *instance* berada pada suatu kelas. Pengujian dilakukan sebanyak 25 kali dengan menggunakan *k-fold cross validation* yang dilakukan sebanyak 4 kali yaitu 4 *fold* dengan hasil rata-rata keseluruhan akurasi 81.1574% (Mubarok, 2013).

Penelitian yang lain dengan metode yang sama pada klasifikasi *genre* film menggunakan data dokumen *text* judul dan sinopsis film dengan jumlah data yang lebih dari 200 data yang diambil dari sebuah situs yang memuat informasi mengenai film dalam skala internasional, yaitu <http://www.imdb.com> (Prasetyo, 2014). Dimana pada metode FKNN pemberian nilai *membership* digunakan sebagai fungsi pola jarak atau kesamaan dari sejumlah himpunan K-NN dan pemberian nilai keanggotaan tetangga pada kelas tertentu sehingga *data testing* yang akan diklasifikasikan akan memiliki nilai keanggotaan pada masing-masing kelas sebagai penentu kelas prediksi. Hanya saja untuk perhitungan nilai jarak pada penelitian ini menggunakan 2 metode yaitu *euclidean distance* dan *cosine similarity* untuk dibandingkan mana yang menghasilkan tingkat akurasi yang lebih baik. Pengujian untuk mengetahui tingkat akurasi hasil klasifikasi pada penelitian ini diukur berdasarkan *recall*, *precision*, dan *F-measure*, dimana akurasi tertinggi dihasilkan dari pemberian nilai k pada rentang 10 sampai 20 dan penggunaan metode *cosine similarity* menghasilkan tingkat akurasi yang lebih baik dibandingkan dengan *euclidean distance* yang dapat dilihat pada Gambar 2.1.



Gambar 2.1 Perbandingan *F-measure* Pada Metode Pencarian Jarak Untuk Kategori *Action*.

Sumber: (Prasetyo, 2014)

Pada penelitian (Prasetyo, 2014) dijelaskan bahwa mengapa metode *euclidean distance* kurang baik, yaitu karena metode *cosine similarity* mempunyai konsep normalisasi panjang vektor data, sedangkan *euclidean distance* tidak. Sehingga jika data training dan data testing akan dicari jaraknya memiliki panjang vektor data yang berbeda, maka hasil pencarian jaraknya menjadi kurang akurat. Namun penggunaan nilai *k* yang terlalu kecil atau terlalu besar mengakibatkan tingkat akurasi yang kurang baik, dimana nilai *k* yang terlalu kecil menyebabkan data terdekat hanya terfokus pada kategori tertentu sedangkan nilai *k* yang terlalu besar menyebabkan data yang memiliki tingkat relevansi rendah akan ikut diambil (Prasetyo, 2014).

Selain itu ada penelitian lain yang juga berkaitan dengan skripsi ini yaitu (Selly, 2013) membandingkan metode klasifikasi K-NN dan FKNN untuk diagnosis penyakit diabetes melitus, setelah proses *missing value* data yang digunakan adalah 290 data (154 diabetes data negatif dan 136 data positif diabetes) dengan 6 atribut. Data tersebut diambil dari repository *database* UCI Indian Pima (<http://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes>) yang terdiri dari

768 data diagnosis DM dan 8 atribut data asli. Dari pengujian yang dilakukan nilai akurasi maksimum diperoleh FKNN mencapai 98% pada $k=15$ sedangkan K-NN hanya mencapai 96% pada $k=11$. Sehingga terbukti bahwa FKNN lebih unggul dibandingkan K-NN, dimana pada nilai k yang besar pada K-NN menyebabkan penurunan akurasi karena terdapat *noise* yang sangat besar, semakin besar nilai k maka semakin banyak kemungkinan kelas diagnosis, sehingga diagnosis menjadi salah. Sebaliknya pada FKNN, nilai k yang besar membuat akurasi lebih stabil karena adanya pengaruh nilai derajat keanggotaan tetangga terdekat. Pada penelitian ini nilai bobot yang paling optimal pada perhitungan FKNN adalah 2, dimana semakin tinggi nilai m yang digunakan, maka semakin rendah nilai derajat keanggotaan yang didapat sehingga berpengaruh pada penentuan hasil kelas prediksi (Selly, 2013).

Penelitian (Chen at all, 2011) yang menggunakan metode *Adaptive Fuzzy k-Nearest Neighbor* berdasarkan pada *Parallel Particle Swarm Optimization* untuk prediksi kebangkrutan. Adapun data yang digunakan adalah *dataset* Wieslaw dengan total data 240 kasus (112 dari Polish perusahaan yang bangkrut dan 128 dari perusahaan yang tidak bangkrut antara tahun 1997 dan 2001). Klasifikasinya terdiri dari dua kelas yaitu bangkrut dan tidak bangkrut. Dimana hal tersebut dilatar belakangi karena belum ada yang mendasari teori ekonomi untuk kebangkrutan sehingga dicari model yang tepat untuk membantu memprediksi kebangkrutan yang lebih akurat. Penelitian tersebut bertujuan untuk mengeksplorasi seluruh potensi dari metode FKNN dengan menentukan k dan m dengan memanfaatkan maksimum akurasi klasifikasi untuk prediksi kebangkrutan secara otomatis dengan menggunakan pendekatan TVPSO. Adapun data yang digunakan adalah *dataset* Wieslaw dengan total data 240 kasus (112 dari Polish perusahaan yang bangkrut dan 128 dari perusahaan yang tidak bangkrut antara tahun 1997 dan 2001). Untuk perhitungan nilai jarak pada klasifikasi FKNN digunakan metode *euclidean distance*. Pengujian dilakukan dengan membandingkan metode *Parallel Time Variant Particle Swarm Optimization* (PTVPSO-FKNN) dengan 3 metode yang lain yaitu KNN, MLP dan SVM dengan pengujian Type I error, Type II error, ACC dan AUC. Pengujian dilakukan 3 tahapan, dimana tahapan pertama pengujian klasifikasi pada seluruh ruang fitur asli yaitu 30 fitur rasio keuangan dengan

memberikan nilai parameter k dan m berbeda tiap fold pada 10 fold yang menghasilkan tingkat akurasi PTVPSO-FKNN lebih baik dibandingkan MLP, KNN, dan SVM yaitu 81.67%. Selanjutnya pengujian klasifikasi dengan model PTVPSO-FKNN dengan seleksi fitur dapat memperbaiki rata-rata akurasi lebih baik dibandingkan tanpa seleksi fitur dimana berturut turut ACC diperbaiki 2.5%, AUC 2.55%, Type I error 1.71%, dan Type II error 3.38%. Pengujian terakhir yaitu membandingkan antara model paralel dan seri TVPSO-FKNN pada *platform multi-core* yang menunjukkan bahwa dengan paralel kinerja CPU jauh lebih cepat dibanding seri yaitu batas atas untuk paralel 1173.80 detik dan seri 3826.96 detik, sedangkan batas bawah untuk paralel 1127.12 detik dan seri 3766.06 detik. Jadi model PTVPSO-FKNN dapat digunakan sebagai alternatif yang menjanjikan dalam membuat keputusan keuangan sebagai sistem peringatan dini. Namun model tersebut tidak dapat bekerja secara efisien pada data ditangan dan belum jelas bahwa algoritma paralel akan lebih baik ketika diterapkan pada data keuangan dengan kasus yang lebih besar, sehingga kedepannya perlu dievaluasi lagi untuk *dataset* yang lebih besar. Selain itu model paralel ini hanya dapat digunakan pada sebuah komputer atau laptop dengan spesifikasi prosesor *quad-core* (Chen at all, 2011).

Pada skripsi ini berdasarkan dari penelitian di atas, penulis tertarik untuk membuat sebuah sistem untuk klasifikasi status sosial ekonomi keluarga dengan menggunakan metode *Fuzzy k-Nearest Neighbor* dengan menggunakan metode *euclidean distance* untuk pencarian pola jarak serta menggunakan metode *Particle Swarm Optimization* untuk optimasi kinerja proses klasifikasi dari metode FKNN dengan menentukan nilai k , m , dan memilih fitur terbaik dengan memanfaatkan maksimum akurasi klasifikasi.

2.2 Status Sosial Ekonomi Keluarga

2.2.1 Definisi Status Sosial Ekonomi Keluarga

Status sosial ekonomi keluarga merupakan salah satu ukuran penempatan anggota masyarakat atau keluarga dalam stratafikasi sosial saat ini yang menempatkan seseorang atau keluarga pada posisi tertentu di dalam masyarakat (Basrowi dan Juariyah, 2010). Dimana status sosial ekonomi ini berkenaan dengan ukuran rata-rata yang berlaku umum bagi setiap orang atau individu mengenai

pendidikan, pekerjaan, kepemilikan barang-barang, kekayaan, kebiasaan dalam kehidupannya sehari-hari, dan status kepemilikan terhadap barang-barang yang dimiliki. Menurut (Basrowi dan Juariyah, 2010) aspek status sosial ekonomi Desa dan peluang kerja berkaitan dengan masalah kesejahteraan masyarakat Desa. Kecukupan pangan dan keperluan ekonomi bagi masyarakat baru terjangkau bila pendapatan rumah tangga cukup untuk menutupi keperluan rumah tangga dan pengembangan usahanya. Menurutnya pula ada beberapa ciri-ciri status sosial ekonomi yaitu sebagai berikut.

1. Lebih berpendidikan
2. Mempunyai status sosial yang ditandai dengan tingkat kehidupan, kesehatan, pekerjaan, dan pengenalan diri terhadap lingkungan.
3. Mempunyai tingkat mobilitas ke atas lebih besar.
4. Mempunyai ladang yang luas.
5. Lebih berorientasi pada ekonomi komersial produk.
6. Mempunyai sikap yang lebih berkenaan dengan kredit.
7. Pekerjaan lebih spesifik.

Di dalam penelitian (Saifi and Mehmood, 2011) menggunakan 6 indikator untuk mengukur status sosial ekonomi seseorang yaitu pendidikan, pekerjaan, pendapatan, material yang dimiliki, pelayan, dan sarana transportasi. Dengan membuat kuesioner sebanyak 28 pertanyaan untuk mendapatkan data terkait indikator yang digunakan.

Hasil penelitian yang dihasilkan oleh (Saifi and Mehmood, 2011) dengan menggunakan 6 indikator dan 28 pertanyaan untuk melihat pengaruh status sosial ekonomi orang tua terhadap prestasi anaknya dapat diambil kesimpulan sebagai berikut:

1. Indikator pertama adalah pendidikan orang tua, dimana siswa dari orang tua yang berpendidikan memiliki nilai maksimum dengan prosentase 68%
2. Indikator kedua adalah pekerjaan orang tua terutama ayah dari analisa data dihasilkan bahwa anak-anak dari pegawai pemerintah dijamin lebih dengan prosentase 60.02% dibandingkan dengan pekerja swasta. Selain pekerjaan ayah adalah pekerjaan ibu, dimana profesi ibu juga berpengaruh terhadap

prestasi anak prosentase maksimum 64.5% bagi siswa yang ibunya bekerja di pemerintahan.

3. Indikator yang ketiga adalah pendapatan yang juga merupakan indikator terbesar dimana pendapatan dapat memprediksi stabilitas mental dan fisik kelurga. Nilai prosentase maksimum 62.09% adalah siswa yang pendapatan orang tuanya kurang lebih diatas 2 juta rupiah, dimana hal tersebut menggambarkan bahwa stabilitas ekonomi keluarga menyebabkan kinerja siswa atau anak disekolah lebih baik.
4. Indikator yang keempat adalah material yang dimiliki, disini dilihat dari beberapa hal alat penunjang belajar anak. Dilihat dari pendapatan orang tua yang tinggi juga memberikan material penunjang bagi anak seperti sarana teknologi informasi komputer, laptop, *gadget*, dan internet. Nilai prosentase 61.32% siswa yang memiliki kompter dirumah dan 60.13% yang menggunakan internet, sehingga sarana tersebut merupakan sumber yang penting, lebih cepat, dan murah untuk mendapatkan pengetahuan.
5. Indikator selanjutnya adalah pelayan yang dimiliki. Nilai prosentase maksimum 69.02% adalah siswa yang memiliki pelayan dirumah, dimana siswa yang memiliki pelayan dirumah lebih terfokus untuk belajar sehingga dapat meningkatkan prestasi belajar anak.
6. Indikator yang terakhir adalah sarana transportasi pada penelitian menunjukkan prosentasi nilai maksimum 62.23% adalah para pelajar yang kuliah dengan membawa mobil karena lebih aman dan nyaman, selain itu juga hanya anak yang membawa mobil saat kuliah yang pendapatan orang tuanya tinggi.

2.1.2 Klasifikasi Status Sosial Ekonomi Keluarga

Kelas sosial ekonomi adalah status atau kedudukan seseorang di masyarakat, dimana berdasarkan pada perbedaan masyarakat kedalam kelas-kelas secara vertical, yang diwujutkan dengan adanya tingkatan masyarakat dari yang tinggi ke yang rendah dengan pengelompokan menurut kekayaan. Menurut (Wardani, 2012) Klasifikasi status sosial ekonomi ada 2 yaitu:

1. Status sosial ekonomi atas

Status sosial atau kedudukan seseorang di masyarakat yang diperoleh dari penggolongan berdasarkan kekayaan, dimana harta yang dimiliki diatas rata-rata masyarakat pada umumnya dan dapat memenuhi kebutuhannya sehari-hari dengan baik. Adapun ciri-cirinya adalah:

- a. Tinggal di rumah-rumah mewah dengan pagar yang tinggi dan berbagai model yang modern dengan status hak milik.
 - b. Tanggungan keluarga tidak lebih dari 5 orang atau pencari nafkah masih produktif yang berusia dibawah 60 tahun dan tidak memiliki penyakit atau sakit.
 - c. Kepala rumah tangga bekerja dan biasanya menduduki tingkat professional ke atas.
 - d. Memiliki modal usaha.
2. Status sosial ekonomi bawah

Status sosial atau kedudukan seseorang di masyarakat yang diperoleh dari penggolongan berdasarkan kekayaan, dimana harta yang dimiliki dibawah rata-rata masyarakat pada umumnya serta tidak mampu memenuhi kebutuhannya sehari-hari. Adapun ciri-cirinya adalah:

- a. Tinggal di rumah kontrakan atau numpang atau rumah sendiri namun kondisinya sangat memprihatinkan seperti masih berdinding kayu atau bamboo serta berlantai tanah.
- b. Tanggungan keluarga lebih dari 5 orang atau pencari nafkah sudah tidak produktif yaitu berusia diatas 60 tahun dan sakit-sakitan atau memiliki penyakit.
- c. Kepala rumah tangga atau ayah menganggur dan hidup dari bantuan orang lain atau saudara dan bekerja sebagai buruh atau pekerja rendahan seperti pembantu rumah tangga, pemulung, dan lain sebagainya.
- d. Tidak memiliki modal usaha.

Selain itu (Wardani, 2012) juga menyebutkan bahwa status sosial ekonomi seseorang dapat diukur dari beberapa aspek diantaranya adalah:

1. Pekerjaan
 - a. Pekerjaan yang tergolong berstatus tinggi yaitu tenaga ahli teknik, tenaga ahli dalam bidang kesehatan, pemimpin pada suatu instansi baik

pemerintah ataupun swasta, pegawai negeri sipil golongan 3 dan 4, pengusaha perusahaan besar, dan lain sebagainya.

- b. Pekerjaan yang tergolong berstatus sedang yaitu pekerjaan dibidang penjualan dan jasa, pengusaha perusahaan sedang, pegawai negeri sipil golongan 1 dan 2, petani dengan sawah milik sendiri, dan lain sebagainya.
 - c. Pekerjaan yang berstatus rendah yaitu buruh, petani yang bekerja mengelola sawah orang lain, bengkel kecil, kuli bangunan, dan lain sebagainya.
2. Pendidikan
 - a. Pendidikan yang tergolong berstatus tinggi yaitu Sarjana S1, Magister S2, dan Doktor S3
 - b. Pendidikan yang tergolong berstatus sedang yaitu SLTP atau SMP, SLTA atau SMA, Diploma I, II, dan III.
 - c. Pendidikan yang tergolong berstatus rendah yaitu Tidak bersekolah, Sekolah Rakyat, TK, dan SD
 3. Pendapatan
 - a. Golongan pendapatan tinggi rata-rata lebih dari Rp. 2.500.000,00 per bulan
 - b. Golongan pendapatan sedang rata-rata antara Rp. 1.500.000,00 sampai dengan Rp. 2.500.000,00 per bulan
 - c. Golongan pendapatan rendah rata-rata dibawah Rp. 1.500.000,00 per bulan.

4. Pemilikan

Selain pekerjaan, pendidikan dan pendapatan yang menjadi ukuran status sosial ekonomi seseorang, masih ada lagi yaitu pemilikan barang-barang yang berharga seperti rumah dan tanah.

- a. Apabila seseorang memiliki rumah sendiri yang mewah; tanah, sawah, ladang yang luas lebih dari 10 are; sepeda motor lebih dari 1; mobil; komputer, laptop, televisi, HP, *gadget*, barang-barang bermerek lebih dari 1 biasanya tergolong orang yang mampu dan golongan atas.

- b. Sedangkan seseorang yang berada pada golongan sedang adalah seseorang yang memiliki rumah sendiri yang sederhana, menempati rumah Dinas, memiliki sepeda motor hanya 1, komputer, televisi, HP yang terbatas.
- c. Apabila seseorang yang tinggal dirumah orang atau numpang, kontrak, sepeda atau bahkan tidak memiliki, radio saja atau bahkan tidak punya, dapat digolongkan bawah.

5. Jenis Tempat Tinggal

- a. Status rumah yang ditempati, rumah sendiri, rumah Dinas, menyewa, menumpang pada saudara atau orang lain. Seseorang yang tergolong atas biasanya tinggal dirumah sendiri dengan status hak milik, sedangkan yang tinggal dirumah sendiri dengan status hak guna bangunan, rumah Dinas dan menyewa merupakan golongan sedang, tetapi bagi seseorang yang status tempat tinggalnya menumpang dengan saudara atau orang lain biasanya tergolong bawah.
- b. Kondisi fisik bangunan dapat berupa permanen, kayu dan bamboo. Keluarga yang keadaan sosial ekonominya diatas pada umumnya tinggal dirumah permanen, sedangkan keluarga yang tergolong sosial ekonominya menengah kebawah tinggal dirumah semi permanen atau bahkan tidak permanen.
- c. Besarnya rumah yang ditempati, semakin luas rumah yang ditempati maka semakin tinggi tingkat sosial ekonominya. Rumah dengan ukuran besar, permanen, dan milik sendiri dapat menunjukkan kondisi sosial ekonominya tinggi berbeda dengan rumah dengan ukuran kecil, semi permanen atau tidak permanen dan menyewa menunjukkan bahwa kondisi sosial ekonominya rendah.

2.3 Logika Fuzzy

Logika *fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang *input* kedalam suatu ruang *output* (Dewi dan Purnomo, 2004). Menurut buku yang disusun oleh (Dewi dan Purnomo, 2004) ada beberapa alasan mengapa banyak orang yang menggunakan logika *fuzzy* diantaranya adalah:

1. Konsep matematis yang mendasari penalaran *fuzzy* sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat *fleksible* sehingga dapat dikombinasikan dengan beberapa metode yang lain.
3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat.
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi *nonlinier* yang kompleks.
5. Logika *fuzzy* dapat mengimplementasikan pengetahuan dan pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
6. Logika *fuzzy* juga dapat bekerjasama dengan teknik-teknik kendali secara konvensional.
7. Logika *fuzzy* didasarkan pada bahasa alami.

Himpunan *fuzzy* memiliki 2 atribut yaitu atribut linguistik dan numeris. Atribut linguistik adalah penamaan suatu grup yang mewakili suatu kondisi dengan menggunakan bahasa alami, sedangkan atribut numeris adalah suatu nilai angka yang menunjukkan ukuran dari suatu variabel (Dewi dan Purnomo, 2004). Ada beberapa hal yang harus diketahui dalam memahami sistem *fuzzy* diantaranya adalah:

1. Variabel *fuzzy* yang merupakan variabel yang akan dibahas atau objek dalam suatu sistem *fuzzy*.
2. Himpunan *fuzzy* yang merupakan suatu grup yang mewakili suatu keadaan dalam suatu variabel *fuzzy*.
3. Semesta pembicaraan adalah keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel *fuzzy*. Nilai ini dapat berupa bilangan positif maupun negatif.
4. Domain himpunan *fuzzy* adalah seluruh nilai yang diperbolehkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*.

Logika *fuzzy* merupakan salah satu komponen pembentuk *soft computing* yang pertama kali diperkenalkan oleh Prof. Lotfi A. Zadeh pada tahun 1965. Teori himpunan *fuzzy* digunakan sebagai dasar logika *fuzzy*, dimana didalam teori tersebut terdapat derajat keanggotaan yang memiliki peranan penting sebagai penentu keberadaan elemen dalam suatu himpunan. Nilai derajat keanggotaan atau *membership function* merupakan ciri utama dari penalaran dengan menggunakan

logika *fuzzy* (Prasetyo, 2014). Derajat keanggotaan adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya yang memiliki interval antara 0 sampai 1. Salah satu solusi untuk mendapatkan nilai tersebut yaitu dengan melalui pendekatan fungsi diantaranya adalah fungsi representasi linier yang digambarkan sebagai satu garis lurus, representasi kurva segitiga yang digambarkan gabungan antara 2 garis (linier), representasi kurva trapesium yang digambarkan seperti kurva segitiga namun ada beberapa titik yang memiliki nilai keanggotaan 1, representasi kurva bentuk bahu yang digambarkan dimana daerah yang terletak di tengah-tengah suatu variabel yang direpresentasikan dalam bentuk segitiga dengan sisi kanan dan kiri akan naik turun, representasi kurva S atau sigmoid yaitu berhubungan dengan kenaikan dan penurunan permukaan yang tidak linier seperti huruf S, representasi kurva bentuk lonceng yaitu kurva yang terbagi dalam 3 kelas yaitu himpunan fuzzy π , beta, dan gauss yang perbedaannya terletak pada gradiennya, koordinat keanggotaan yaitu himpunan fuzzy yang berisi urutan berpasangan dan berisi nilai domain dan kebenaran nilai keanggotaannya dalam bentuk skalar atau derajat, dimana skalar adalah nilai yang digambar dari domain himpunan fuzzy sedangkan derajat merupakan derajat keanggotaan himpunan fuzzynya (Dewi dan Purnomo, 2004).

2.4 Data Mining

Istilah *data mining* digunakan untuk menguraikan penemuan pengetahuan didalam *database*. *Data mining* itu sendiri didefinisikan sebagai proses ekstraksi dan identifikasi informasi atau pola yang penting dan pengetahuan dalam basis data dengan ukuran yang besar dengan menggunakan teknik statistik, matematik, kecerdasan buatan, dan *machine learning* (Amri, 2013). *Data mining* dapat didefinisikan juga sebagai penguraian secara implisit dari sekumpulan data yang tidak diketahui sebelumnya dan informasinya memiliki potensi kegunaan serta merupakan penggalian dan analisis dengan menggunakan piranti otomatis atau semi otomatis dari sejumlah besar data yang bertujuan untuk menemukan bentuk yang bermanfaat (Puspitaningrum dkk, 2012).

Adapun beberapa karakteristik umum yang dimiliki data yang akan di mining adalah memiliki ukuran data yang besar, data tidak lengkap sehingga perlu

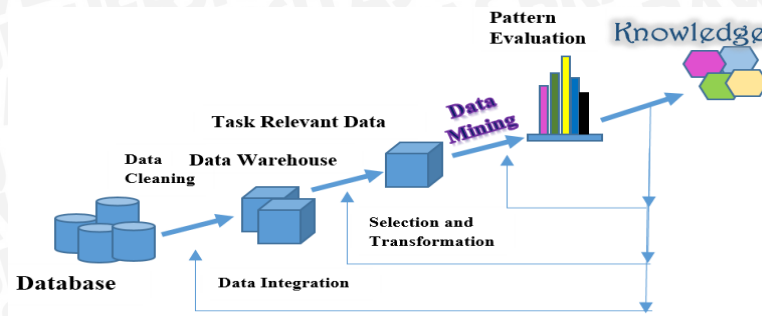
dilakukan proses *cleaning*, struktur data kompleks dengan banyak dimensi, dan merupakan kumpulan data yang heterogen dan tersebar (Puspitaningrum dkk, 2012).

Tahapan *data mining* terdiri dari tiga langkah utama yaitu *data preparation*, penggunaan algoritma data mining, dan tahap analisa data. Tahap *data preparation* merupakan langkah untuk memilih data, membersihkan, dan melakukan *preprocessed* dengan mengikuti pedoman dari ahli domain yang menangkap dan mengintegrasikan data internal dan eksternal ke tinjauan organisasi secara menyeluruh. Tahap penggunaan algoritma *data mining* merupakan langkah menggunakan algoritma *data mining* untuk memberi kemudahan dalam melakukan identifikasi data dan mengintegrasikan keseluruhan data yang telah ditemukan. Tahap yang terakhir adalah tahap analisa data yaitu langkah mengevaluasi hasil dari *data mining* untuk melihat apakah *knowledge* domain ditemukan dalam bentuk *rule* yang telah diekstrak dari jaringan (Asep dan Sugeng, 2013).

Terdapat dua tujuan umum dalam proses *data mining* yaitu deskriptif *data mining* dan prediktif *data mining*. Deskriptif *data mining* merupakan deskripsi konsep atau *task relevan* data dalam bentuk yang ringkas, informatif, dan diskriminatif, contohnya adalah *clustering*, *association rule discovery*, dan *sequential pattern discovery*. Berdasarkan analisis data prediktif *data mining* dibuat model untuk kemudian dijadikan sebagai alur prediksi *trend* dan data-data yang tidak diketahui nilainya, contohnya adalah *classification*, *regression*, dan *deviation detection* (Puspitaningrum dkk, 2012).

Data mining merupakan inti dari proses *Knowledge Discovery in Database* (KDD). KDD merupakan proses yang terorganisir yang digunakan untuk mengidentifikasi pola yang valid, baru, berguna, dan mudah dimengerti dari *data set* yang besar dan kompleks (Asep dan Sugeng, 2013).

Knowledge Discovery in Database (KDD) adalah proses menemukan informasi yang berguna dengan pola-pola yang ada dalam data. KDD ini adalah sebuah proses yang terdiri dari serangkaian proses iteratif yang terurut dan *data mining* merupakan salah satu langkah dalam proses KDD tersebut. Untuk tahapan proses KDD dapat dilihat pada Gambar 2.2.



Gambar 2.2 Tahapan dalam KDD
Sumber: (Han and Kember, 2001)

Tahapan tahapan proses KDD menurut Han dan Kember yang dikutip oleh (Puspitaningrum dkk, 2012) adalah sebagai berikut:

1. **Pembersihan Data (Data Cleaning)** yang merupakan pembersihan terhadap data yang dilakukan untuk menghilangkan data yang tidak konsisten dan mengandung *noise*.
2. **Integrasi Data (Data Integration)** yang merupakan proses integrasi data yang dilakukan untuk menggabungkan data dari berbagai sumber.
3. **Seleksi Data dan Transformasi Data (Selection and Transformation)** dimana proses seleksi data yaitu mengambil data yang relevan digunakan untuk proses analisis, sedangkan proses transformasi adalah proses mentransformasikan atau menggabungkan data ke dalam bentuk yang tepat untuk di mining dengan cara melakukan peringkasan atau operasi agregasi.
4. **Data Mining** yang merupakan proses yang penting dimana pada proses ini metode-metode cerdas diaplikasikan untuk mengekstrak pola-pola dalam data.
5. **Evaluasi Pola (Pattern Evaluation)** dimana proses ini diperlukan untuk mengidentifikasi beberapa pola-pola yang menarik dalam merepresentasikan pengetahuan.
6. **Presentasi Pengetahuan (Knowledge)** merupakan penggunaan visualisasi dan teknik representasi untuk menunjukkan pengetahuan hasil penggalian gunung data kepada pengguna.

2.5 Klasifikasi

Klasifikasi adalah suatu fungsionalitas *data mining* bagaimana menempatkan beberapa objek ke salah satu dari beberapa kategori yang sebelumnya telah ditetapkan (*supervised learning*). Misalnya pada aplikasi pengklasifikasian *genre* film berdasarkan judul film yang ditetapkan pada beberapa kategori yaitu *action*, *drama*, *romance*, *horror*, dan komedi atau mengklasifikasikan jenis mangga pada kategori mangga gadung atau curut berdasarkan tekstur daunnya. Klasifikasi dapat didefinisikan juga sebagai proses menemukan model untuk menjelaskan kelas data yang tujuannya untuk memperkirakan kelas dari suatu objek yang labelnya belum diketahui sebelumnya. Model tersebut dapat berupa aturan “jika-maka”, berupa *decision tree*, formula matematis atau *neural network* (Asep dan Sugeng, 2013).

Teknik klasifikasi merupakan pendekatan secara sistematis untuk membuat model klasifikasi dari inputan *dataset*. Adapun beberapa contoh teknik klasifikasi diantaranya adalah Naive Bayes, Decision Tree based Methods, Rule-based Methods, Support Vector Machines, Neural Networks, K-Nearest Neighbor (K-NN), banyak juga yang telah dikembangkan dengan mengkombinasi 2 atau lebih teknik klasifikasi seperti Fuzzy K-Nearest Neighbor, Adaptive K-Nearest Neighbor dan lain sebagainya. Pada setiap teknik klasifikasi tersebut pastinya memiliki algoritma pembelajaran berbeda-beda untuk mengidentifikasi model yang memberikan hubungan yang pas dan sesuai dari data input antara himpunan atribut dengan label kelas. Kunci utama dari algoritma pembelajaran yaitu membangun model sesuai dengan data input dengan kemampuan generalisasi yang baik, dimana model secara akurat memprediksi label kelas yang tidak diketahui sebelumnya.

Terdapat dua *fase* pada proses klasifikasi yaitu *fase learning* dan *fase test*. *Fase learning* merupakan *fase* membentuk model prakiraan dari sebagian data yang telah diketahui kelas datanya, sedangkan *fase test* adalah *fase* menguji akurasi model untuk memprediksi kelas data yang belum diketahui menggunakan model yang sudah dibentuk dengan data yang lainnya. Pendekatan umum untuk membangun model klasifikasi pertama menyiapkan *data training* atau bisa disebut *training set* berisi *record* data yang memiliki label kelas yang diketahui. Kemudian membuat model klasifikasi dengan menggunakan *training set* untuk diaplikasikan

ke *data testing* atau *test set* yang berisi *record-record* data yang label kelasnya tidak diketahui (Anonim, 2013).

2.6 Jenis Data dan Cara Konversi Data

Setelah melewati proses pengumpulan data, jenis data yang dikumpulkan akan sangat mempengaruhi dalam proses analisa data untuk menentukan jenis analisa apa yang akan digunakan agar data dapat diolah. Sebelum menganalisa suatu data sebaiknya kita harus mengenali jenis data apa yang akan dianalisa dan diolah. Adapun jenis data yang ada dalam suatu penelitian akan dijelaskan sebagai berikut.

2.6.1 Data Berdasarkan Sumbernya

Berdasarkan pada sumbernya data dapat dibedakan dalam 2 jenis, yaitu data primer dan data sekunder (Aedi, 2010).

1. Data primer adalah jenis data yang didapatkan secara langsung oleh peneliti dari sumber utama. Data ini memiliki sifat yang *up to date* karena merupakan data asli dan baru. Sedangkan untuk mendapatkannya peneliti harus mengumpulkannya secara langsung misalnya dengan melakukan observasi, wawancara, dan menyebarkan angket atau kuesioner.
2. Data sekunder adalah jenis data yang didapatkan tidak secara langsung oleh peneliti, namun didapatkan dari berbagai sumber yang telah ada atau bisa dikatakan bahwa peneliti merupakan tangan kedua. Untuk mendapatkannya peneliti dapat memperoleh dari berbagai sumber seperti Biro Pusat Statistik (BPS), buku, laporan, jurnal, dan lain sebagainya.

2.6.2 Data Berdasarkan Bentuk dan Sifatnya

Berdasarkan pada bentuk dan sifatnya data dapat di bedakan dalam 2 jenis juga, yaitu data kualitatif dan data kuantitatif. Biasanya data kualitatif berbentuk kata-kata atau kalimat sedangkan data kuantitatif berbentuk angka. Berdasarkan pada cara untuk mendapatkannya data kuantitatif dibedakan menjadi 2 yaitu diskrit dan kontinum, sedangkan berdasarkan pada sifatnya data kuantitatif terdiri dari data nominal, ordinal, interval, dan rasio (Aedi, 2010).

1. Data kualitatif adalah data yang kebanyakan berbentuk kata-kata bukan berbentuk angka. Adapun cara untuk memperolehnya dapat dilakukan dalam berbagai macam cara diantaranya adalah wawancara, analisis dokumen, diskusi terfokus, atau observasi yang telah tertuang dalam catatan lapangan. Bentuk yang lain misalnya gambar yang diperoleh dari pemotretan atau rekaman video. Dalam pengolahan data kualitatif ada yang langsung menggunakannya namun ada juga yang menggunakan dengan mengkonversi kedalam data kuantitatif sebelum digunakan.
2. Data kuantitatif merupakan data yang kebanyakan berupa angka atau bilangan, dimana dalam mengolah dan menganalisanya menggunakan teknik matematika atau statistika. Sedangkan berdasarkan proses untuk mendapatkannya data kuantitatif dapat dibedakan ke dalam 2 bentuk yaitu (Aedi, 2010):
 - a. Data diskrit adalah data yang diperoleh dengan cara membilang, misalnya jumlah ternak yang dimiliki adalah 30. Karena diperoleh dengan cara membilang maka data diskrit pasti berupa bilangan bulat bukan pecahan.
 - b. Data kontinum adalah data yang diperoleh dari hasil pengukuran. Adapun bentuknya bilangan kontinum dapat berupa bilangan bulat atau pecahan tergantung pada jenis skala pengukuran yang digunakan, misalnya luas tanah yang dimiliki Pak Efendi 3 are.

Berdasarkan pada tipe skala pengukurannya, data kuantitatif dapat dibedakan menjadi 4 jenis yang memiliki sifat berbeda yaitu (Aedi, 2010):

 - a. Data nominal atau biasa disebut data kategori yaitu data yang diperoleh melalui pengelompokan objek berdasarkan pada kategori tertentu. Walaupun data nominal ini berbentuk angka namun tidak dapat diukur secara matematis sehingga tidak dapat dibandingkan. Contohnya adalah keadaan rumah terdiri dari 3 kategori, yaitu angka (1) untuk memprihatinkan, angka (2) untuk sederhana, dan angka (3) untuk bagus.
 - b. Data ordinal adalah data yang diperoleh dari suatu objek atau kategori yang telah disusun secara berjenjang berdasarkan besarnya yang memiliki tingkatan tertentu yang dapat diurutkan dari yang terendah

sampai tertinggi atau sebaliknya. Contohnya adalah pendidikan yang disusun dalam urutan dari terendah sampai tertinggi, yaitu angka (1) untuk SD, angka (2) untuk SMP, angka (3) untuk SMA, angka (4) untuk Diploma, angka (5) untuk S1, angka (6) untuk S2, dan angka (7) untuk S3.

- c. Data interval adalah data yang diperoleh dari hasil pengukuran yang dapat diurutkan atas dasar kriteria tertentu dan menunjukkan semua sifat yang dimiliki oleh data ordinal. Adapun perbedaan antara data interval dan ordinal adalah data interval memiliki rentang yang sama antara data yang telah diurutkan. Selain dapat dilakukan operasi matematis penjumlahan dan pengurangan, data interval tidak memiliki angka Nol mutlak. Contohnya adalah pekerjaan dengan memberikan skor pada tiap jenis pekerjaan berdasarkan tingkatan pendapatan yang diperoleh, yaitu skor (1) untuk petani, skor (2) untuk wiraswasta, skor (3) untuk wirausaha, dan skor (4) untuk PNS dan Pegawai BUMN.
- d. Data rasio adalah data yang menghimpun dari semua sifat sebelumnya yaitu data nominal, data ordinal, dan data interval. Data rasio merupakan data angka yang berarti sesungguhnya karena dilengkapi titik Nol absolut (mutlak), sehingga dapat diterapkan kesemua bentuk perhitungan matematis. Contohnya adalah luas sawah yang dimiliki ibu mistiniati memiliki luas 33 are atau $33 \times 100 \text{ m}^2$.

2.6.3 Konversi Data Kualitatif ke Kuantitatif dan Teknik Pengolahan Data

Pada setiap penelitian pasti terdapat data yang akan diolah dan digunakan. Setelah mengenali jenis data apa yang diperoleh peneliti harus menganalisa data seperti apa yang akan diolah dan digunakan. Permasalahan yang biasanya muncul dalam penelitian adalah bagaimana cara menganalisa data yang berbentuk kualitatif dan kuantitatif pada penelitian kuantitatif. Untuk data kuantitatif pasti tidak akan bermasalah karena datanya berupa angka sehingga dapat langsung digunakan, namun bagaimana pada data kualitatif, data kualitatif pasti harus dikonversi ke dalam data kuantitatif.

Adapun salah satu cara yang dapat dilakukan untuk mengkonversikan data kualitatif ke dalam data kuantitatif yaitu dengan cara merubah data kualitatif yang berbentuk kata menjadi data kuantitatif kontinum bisa dikonversi ke data nominal, ordinal, interval, atau rasio tergantung pada data yang bagaimana yang dibutuhkan misalnya pada data kualitatif pekerjaan diantaranya adalah petani, wiraswasta, wirausaha, PNS, Pegawai BUMN. Kemudian dikonversi ke dalam data kuantitatif dengan cara mengintervalkan yaitu untuk petani diberikan skor 1, wiraswasta skor 2, wirausaha skor 3, PNS skor 4, pegawai BUMN skor 5, dan lain sebagainya.

Pengolahan data adalah bagian terpenting dalam suatu penelitian karena dengan pengolahan data, data tersebut memiliki makna yang berguna dalam memecahkan masalah penelitian. Adapun langkah langkah pengolahan data secara garis besar dapat dibedakan menjadi 4 yaitu penyusunan data, klasifikasi data, pengolahan data, dan interpretasi hasil pengolahan data (Aedi, 2010).

1. Penyusunan data yaitu proses mengumpulkan dan mengecek semua data yang dibutuhkan apakah sudah terekap semua dimaksudkan untuk menguji hipotesis penelitian. Dimana penyusunan data harus dipilih data yang ada hubungannya dengan penelitian dan benar-benar otentik.
2. Klasifikasi data adalah usaha menggolongkan, mengelompokkan, dan memilih data berdasarkan pada kategori tertentu yang telah ditentukan oleh peneliti tujuannya untuk memudahkan pengujian hipotesis.
3. Pengolahan data adalah perlakuan untuk menguji hipotesis yang telah dirumuskan. Walaupun semua penelitian tidak harus berhipotesis namun wajib memiliki rumusan masalah sehingga jenis data akan menentukan apakah peneliti menggunakan teknik kualitatif atau kuantitatif.
4. Interpretasi hasil pengolahan data yaitu tahap menerangkan setelah peneliti selesai menganalisa datanya dengan cermat yang selanjutnya diinterpretasikan hasilnya untuk menarik suatu kesimpulan.

Penelitian ini merupakan penelitian kuantitatif sehingga pengolahan datanyapun menggunakan teknik pengolahan data kuantitatif. Karena penelitian kuantitatif menggunakan data berupa angka sehingga data kuantitatif memerlukan adanya perhitungan secara matematis misalnya diolah dan dianalisis dengan statistik. Perhitungan statistik terdapat 2 macam yaitu statistik deskriptif untuk

mendeskriskan variabel penelitian melalui pengukuran, sedangkan statistik inferensial untuk menguji hipotesis dan membuat generalisasi. Pengolahan statistik digunakan untuk mengolah data kuantitatif agar data mempunyai arti dengan teknik distribusi frekuensi (sebaran frekuensi) dan ukuran memusat (mean, median, modus) (Aedi, 2010).

Kegiatan awal yang dilakukan dalam pengolahan data kuantitatif pertama adalah *editing* yaitu proses memeriksa data yang telah terkumpul yang meliputi kelengkapan isi, bacaan tulisan, kejelasan jawaban, relevansi jawaban, keseragaman satuan data yang digunakan, dan sebagainya. Kedua adalah *coding* yaitu kegiatan memberikan kode pada setiap data yang terkumpul disertai instrument penelitian yang bertujuan untuk memudahkan dalam menganalisa dan menafsirkan data. Ketiga adalah *tabulating* yaitu memasukkan data yang sudah dikelompokkan kedalam table-table agar mudah dipahami (Aedi, 2010).

2.7 Normalisasi Data

Normalisasi adalah suatu teknik yang dilakukan untuk mengelompokkan data atau nilai atribut sesuai dengan skala tertentu. Normalisasi dilakukan untuk menghindari beberapa atribut yang memiliki nilai besar mempengaruhi atribut lain yang diukur dengan skala yang lebih kecil (Yessivirna, 2013) atau mengecilkan range data gambar misalnya range piksel. Ada beberapa metode normalisasi diantaranya adalah metode *Z-Score standardization* (z') pada data kontinu dapat dilihat pada Persamaan 2-1 (Yessivirna, 2013), dimana normalisasi berdasarkan *mean* (nilai rata-rata) dan *standard deviation* dari data.

$$z' = \frac{z - \text{mean}(z)}{\text{STD}(z)} \quad 2-1$$

Dimana z' adalah nilai baru, z adalah nilai awal, $\text{mean}(z)$ adalah nilai rata-rata, sedangkan $\text{STD}(z)$ adalah nilai standart deviasi dari data. Metode ini dapat digunakan jika peneliti tidak mengetahui nilai aktual minimum dan maksimum dari suatu data

Metode *min-max normalization* (x') untuk atribut campuran antara numerik dan kategorikal dapat dilihat pada Persamaan 2-2 (Leidiyana, 2013).

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad 2-2$$

Dimana x' adalah nilai baru rentang nilai antara 0 sampai 1, x adalah nilai awal, $\min(x)$ adalah nilai minimum variabel x dan $\max(x)$ nilai maksimum x .

Metode aktivasi *sigmoid biner* (s') untuk menyelesaikan permasalahan dunia nyata yang kompleks dan biasanya bersifat non-linier dengan mentransformasikan data pada interval $[0.1, 0.9]$ karena fungsi ini tidak pernah mencapai 0 atau 1 dapat dilihat pada Persamaan 2-3, namun jika aktivasi yang digunakan adalah fungsi *sigmoid bipolar* (sp') data dinormalisasikan ke interval $[-1, 1]$ terlebih dahulu dapat dilihat pada Persamaan 2-4 (Mardianto dan Pratiwi, 2008).

$$s' = \frac{0.8(s - a)}{b - a} + 0.1 \quad 2-3$$

$$sp' = \frac{1 - e^{-s}}{1 + e^{-s}} \quad 2-4$$

Dimana s' dan sp' adalah nilai baru rentang nilai antara 0 sampai 1 untuk *sigmoid biner* tetapi lebih baik ditransformasi ke interval yang lebih kecil yaitu 0.1 sampai 0.9, hal ini dilakukan untuk menghindari nilai 0 dan rentang -1 sampai 1 untuk *sigmoid bipolar*, s adalah nilai awal atribut yang akan dinormalisasi, a adalah nilai minimum atribut dan b adalah nilai maksimum atribut dan e adalah nilai eksponensial.

2.8 *k- Nearest Neighbor* (k-NN)

Algoritma *k-nearest neighbor* (k-NN) adalah salah satu metode yang sering digunakan yang merupakan *instance-based learning* dan tergolong *supervised learning*, dimana proses klasifikasi untuk *record* baru yang belum diklasifikasi dilakukan dengan menyimpan *data training* terlebih dahulu kemudian hasilnya dapat ditemukan dengan membandingkan tingkat kemiripan yang paling banyak terhadap *data training*. Pada algoritma ini yang harus dipertimbangkan adalah nilai k , karena pemilihan nilai k yang terlalu kecil menyebabkan sensitif terhadap *noise*, k terlalu besar juga dapat menyebabkan *neighborhood* dapat mencakup titik-titik dari kelas lain, sehingga dilakukan pemilihan dengan meminimalkan estimasi *error* pengklasifikasian (Muflikhah, 2012). Adapun tujuannya adalah mengklasifikasikan objek baru berdasarkan atribut dan data training dengan menggunakan klasifikasi ketetanggaan sebagai prediksi terhadap data baru.

Secara umum algoritma k-NN memiliki prosedur yaitu pertama menentukan parameter k . Kemudian menghitung jarak antara data yang akan dievaluasi dengan semua pelatihan dan urutkan jarak yang terbentuk dengan urutan naik. Selanjutnya tentukan jarak terdekat atau jarak minimum sampai urutan k , pasangkan kelas yang bersesuaian, dan cari jumlah kelas dari tetangga yang terdekat dan tetapkan kelas tersebut sebagai kelas data yang akan dievaluasi (Riyanda, 2013).

Pada penelitian yang dilakukan oleh (Baharsyah, 2014) yang mengklasifikasikan dokumen *e-complaint* dengan menggunakan metode *k-nearest neighbor* (k-NN), menjelaskan bahwa k-NN adalah sebuah algoritma terbaik yang digunakan untuk mengkategorikan teks. Adapun tahapan metode k-NN adalah pertama yaitu hasil dari pembobotan sebagai inputannya yang kemudian dihitung nilai kemiripan antar dokumen, selanjutnya menentukan nilai k dan menentukan kelas dokumen, yang terakhir menyimpan hasil klasifikasi (Baharsyah, 2014).

Untuk mendapatkan hasil klasifikasi dengan menggunakan metode *k-nearest neighbor* hal pertama yang dilakukan adalah mencari nilai jarak yaitu mengukur seberapa dekat jarak antara *data training* dengan *data testing*. Metode yang dapat digunakan untuk menentukan jarak dapat dengan menggunakan metode *euclidean distance* atau *cosine similarity*. Adapun rumus dari metode *euclidean distance* dapat dilihat pada Persamaan 2-5 (Selly, 2013) dan *cosine similarity* pada Persamaan 2-6 di bawah ini (Prasetyo, 2014).

$$\|x - x_j\| = \left(\sum_{l=1}^n |N_l - N_l^j|^2 \right)^{\frac{1}{2}} \quad 2-5$$

Dimana $norm\ x$ dikurangi x_j ($\|x - x_j\|$) adalah nilai jarak *euclidean* antara data *testing* x dan tetangga terdekat atau data training ke- j (x_j). Sedangkan N_l adalah nilai bobot data testing pada atribut ke- l dan N_l^j nilai bobot data training ke- j pada atribut ke- l . Sedangkan n adalah jumlah keseluruhan atribut dan l adalah index ke- l .

$$sim(q, d) = \frac{\sum w_d \cdot w_q}{\sqrt{\sum w_d^2} \cdot \sqrt{\sum w_q^2}} \quad 2-6$$

Dimana $sim(q, d)$ adalah nilai kemiripan atau *similarity* antara data *testing* dengan data *training*, dimana d adalah data *training* sedangkan q adalah data

testing. Sedangkan w_d adalah nilai bobot *data training* dan w_q adalah nilai bobot *data testing*.

2.9 Fuzzy k-Nearest Neighbor (FKNN)

Algoritma *fuzzy k-nearest neighbor* dasarnya adalah menetapkan nilai keanggotaan atau *membership* sebagai fungsi pola jarak atau kesamaan dari sejumlah himpunan k-NN dan pemberian nilai keanggotaan tetangga pada kelas tertentu (Prasetyo, 2014). Perbedaan antara metode k-NN dengan FKNN adalah dimana k-NN melakukan prediksi secara tegas pada *data testing* berdasarkan perbandingan k tetangga terdekat, sedangkan FKNN yang pertama diperkenalkan oleh Keller at all pada tahun 1985 yaitu dengan mengembangkan k-NN yang digabung dengan teori *fuzzy* dalam mendefinisikan pemberian label kelas pada *data testing* yang diprediksi namun tidak secara tegas (Shofa, 2013).

Teori himpunan *fuzzy* yang digunakan mengeneralisasi teori k-NN klasik dengan mendefinisikan nilai keanggotaan sebuah data pada masing-masing kelas, seperti halnya pada teori *fuzzy* sebuah data memiliki nilai keanggotaan pada tiap kelas maksudnya adalah sebuah data bisa dimiliki oleh kelas yang berbeda dengan nilai derajat keanggotaan dalam interval 0 sampai 1 (Mubarok, 2013). Sehingga pada algoritma FKNN *data testing* yang diklasifikasikan memiliki nilai keanggotaan pada semua kelas namun klasifikasi dilakukan dengan memilih nilai keanggotaan kelas pada *data testing* x , $(u_1(x))$ yang paling tinggi (Prasetyo, 2014).

Untuk menentukan nilai keanggotaan kelas ke- i pada tetangga ke- j dihitung dengan menggunakan Persamaan 2-7 (Selly, 2013).

$$u_{ij} = \begin{cases} 0.51 + \left(\frac{n_j}{K}\right) \times 0.49, & \text{kelas } j = i \\ \left(\frac{n_j}{K}\right) \times 0.49, & \text{kelas } j \neq i \end{cases} \quad 2-7$$

Dimana u_{ij} adalah nilai keanggotaan tetangga ke- j pada kelas ke- i dan n_j adalah jumlah data yang termasuk kelas j pada *data training*. Sedangkan i adalah kelas status sosial ekonomi, j adalah kelas tetangga dan K adalah jumlah *data training*.

Untuk menentukan nilai keanggotaan kelas pada *data testing* x dengan menggunakan metode *cosine similarity* dihitung dengan menggunakan Persamaan 2-8 (Prasetyo, 2014).

$$u_i(x) = \frac{\sum_{j=1}^k u_{ij}(\text{sim}(q, d))}{\sum_{j=1}^k (\text{sim}(q, d))} \quad 2-8$$

Dimana $u_i(x)$ adalah nilai keanggotaan kelas pada *data testing* x , u_{ij} adalah nilai keanggotaan kelas ke- i pada tetangga ke- j yang telah dihitung menggunakan Persamaan 2-7, dan $\text{sim}(q, d)$ adalah nilai kemiripan atau *similarity* antara *data testing* q dengan *data training* d yang telah dihitung dengan menggunakan Persamaan 2-6.

Sedangkan untuk menentukan nilai keanggotaan kelas pada *data testing* x dengan menggunakan metode *Euclidean distance* antara *data testing* x dan tetangga terdekat x_j ke- j dihitung dengan menggunakan Persamaan 2-9 (Chen at all, 2011).

$$u_i(x) = \frac{\sum_{j=1}^k u_{ij} \left(\frac{1}{\|x - x_j\|^{\frac{2}{m-1}}} \right)}{\sum_{j=1}^k \left(\frac{1}{\|x - x_j\|^{\frac{2}{m-1}}} \right)} \quad 2-9$$

Dimana $u_i(x)$ adalah nilai keanggotaan *data testing* x pada kelas ke- i , $i = 1, 2, \dots, c$ dan $j = 1, 2, \dots, k$ dengan c adalah nilai kelas dan k adalah nilai tetangga terdekat. Parameter kekuatan *fuzzy* m adalah bobot pangkat yang digunakan untuk menentukan seberapa banyak jarak tertimbang ketika mengkalkulasi kontribusi tiap tetangga sebagai nilai keanggotaan dan nilai tersebut biasanya sebagai berikut $m \in (1, +\infty)$. Sedangkan *norm* x dikurangi x_j ($\|x - x_j\|$) adalah jarak Euclidean pada Persamaan 2-5 dan u_{ij} nilai keanggotaan tetangga ke- j terhadap kelas ke i pada Persamaan 2-7.

2.10 Particle Swarm Optimization (PSO)

2.10.1 Definisi Particle Swarm Optimization (PSO)

Algoritma *particle swarm optimization* (PSO) pertama dikenalkan oleh Kennedy dan Eberhart pada tahun 1995 (Dhanasaputra dan Santosa, 2012). PSO memiliki banyak kesamaan dengan teknik-teknik *evolutionary computation* yang lain, misalnya *genetic algorithms* (GA) dan *evolutionary strategies* (ES). PSO maupun GA dimulai dengan sebuah populasi yang terdiri dari berbagai macam individu yang dibangkitkan secara acak yang selanjutnya dilakukan pencarian solusi terbaik dengan memperbaiki individu untuk beberapa generasi tertentu. PSO memiliki *memory* untuk menyimpan solusi terbaik dan tidak menggunakan operator-operator evolusi rekombinasi dan mutasi, namun sebaliknya untuk GA. Selain itu setiap partikel PSO tidak pernah mati dan sebaliknya untuk GA bisa mati dan digantikan dengan individu baru. Posisi dan kecepatan terbang partikel pada PSO di-update pada setiap iterasi sehingga partikel tersebut dapat menghasilkan solusi baru yang lebih baik (Suyanto, 2010).

Algoritma ini merupakan teori yang diinspirasi oleh tingkah laku sosial sekawanan burung atau ikan. Algoritma ini dapat dianalogikan sebagai berikut misalnya ada sekawanan burung atau ikan yang mencari makanan secara random pada suatu daerah, dimana hanya ada sebuah makanan di daerah tersebut dan sekawanan burung atau ikan tersebut tidak tahu dimana keberadaan makanan tersebut. Namun mereka tahu makanan pada setiap iterasi, sehingga strategi yang dapat digunakan oleh mereka adalah mencari makanan dengan mengikuti burung yang keberadaannya lebih dekat dengan makanan tersebut (Mohammad dkk, 2011).

Berdasarkan analogi perilaku burung yaitu burung lebih suka terbang bersama namun seekor burung tidak akan berada dekat dengan burung yang lain, sekawanan burung akan mengikuti arah rata-rata keseluruhan burung bersama, dan memposisikan diri menjaga jarak agar tidak terlalu jauh dari kawanannya. Hal tersebut digunakan untuk mengembangkan PSO dengan model dimana pertama ketika seekor burung mendekati target atau makanan (bisa minimal atau maksimal) secara cepat memberikan informasi kepada temannya yang lain, kemudian secara tidak langsung burung yang lain akan mengikuti arah menuju ke makanan, dimana setiap burung memiliki pikiran dalam memorinya tentang apa yang telah dilewati

pada waktu sebelumnya. Sehingga model disimulasikan dengan dimensi tertentu dengan sejumlah iterasi sehingga setiap iterasi posisi partikel akan semakin mengarah kepada target yang dituju (minimalisasi atau maksimalisasi fungsi). Hal tersebut dilakukan hingga maksimum iterasi dicapai atau dengan kriteria penghentian yang lain (Santosa, 2014).

Pada PSO standar, suatu partikel atau solusi tersusun oleh tiga vektor dan dua nilai *fitness*. Vektor pertama yaitu X yaitu vektor yang menyimpan posisi partikel saat ini diruang pencarian, $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$. Vektor P yaitu vektor untuk menyimpan posisi dari solusi terbaik sejauh ini yang ditemukan oleh partikel tersebut, $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})$. Vektor V yaitu vektor yang berisi *gradient* atau arah yang menyatakan kemana partikel akan terbang, $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$. Dua nilai *fitness* yaitu *x-fitness* yang menyimpan nilai *fitness* dari *x-vector* dan *p-fitness* yang menyimpan nilai *fitness* dari *p-vector*. Pertanyaannya adalah bagaimana partikel terbang dari satu posisi ke posisi yang lain, hal ini dilakukan dengan cara sederhana yaitu menambahkan *v-vector* ke *x-vector* untuk mendapatkan *x-vector* yang baru dengan rumus $X_i' = X_i + V_i$. Begitu partikel mendapatkan partikel yang baru maka dia akan mengevaluasi posisi barunya. Jika *x-fitness* lebih baik daripada *p-fitness*, maka $P_i = X_i$ dan *p-fitness* = *x-fitness* (Suyanto, 2010).

Algoritma PSO ini diawali dengan inisialisasi sekawanan partikel, dimana tiap partikel direpresentasikan dengan vector berukuran *d*-dimensi. Kemudian posisi setiap partikel dievaluasi dengan fungsi *fitness* yang telah ditentukan sebelumnya. Kecepatan tiap partikel dapat berubah-ubah tergantung pada posisinya dan terdapat pertukaran informasi antar partikel juga, dimana tiap partikel akan merubah kecepatan terbangnya dengan mempertimbangkan posisi terbaik yang pernah dilaluinya, selain itu kecepatan juga dipengaruhi oleh posisi terbaik diantara seluruh partikel. Adapun langkah langkah algoritma PSO adalah sebagai berikut (Dhanasaputra dan Santosa, 2012):

1. Inisialisasi yaitu menentukan jumlah populasi
2. Membangkitkan populasi secara acak
3. Membangkitkan kecepatan awal
4. Mengevaluasi fungsi *fitness*
5. Menyimpan partikel yang memiliki nilai *fitness* paling optimal

6. Selama *terminating condition* belum terpenuhi lakukan langkah 6-10
7. Menghitung kecepatan setiap partikel
8. Menghitung posisi baru setiap partikel
9. Mengevaluasi fungsi *fitness* yang baru
10. Mengupdate solusi optimal
11. Mengupdate partikel terbaik dan yang terakhir yaitu menemukan solusi pada partikel terbaik.

2.10.2 Algoritma PSO

PSO sudah dijelaskan juga sebelumnya bahwa PSO ini dimulai dengan sekumpulan partikel yang dibangkitkan secara acak kemudian setiap partikel dievaluasi kualitasnya dengan menggunakan fungsi *fitness*, selanjutnya partikel-partikel akan mengikuti partikel yang optimum. Pada tiap generasi, partikel di-update mengikuti dua nilai yang terbaik yaitu *fitness* terbaik yang dicapai oleh satu partikel saat ini yang dilambangkan dengan p dan disimpan di memori, yang kedua adalah nilai *fitness* terbaik yang dicapai oleh semua partikel dalam topologi ketetanggaan yang dilambangkan dengan g . Setelah menemukan dua nilai terbaik, suatu partikel i pada posisi X_i meng-update vektor *velocity* dan kemudian meng-update posisinya menggunakan persamaan berikut (Suyanto, 2010):

$$v_{id}' = v_{id} + \varphi_1 * r * (p_{id} - x_{id}) + \varphi_2 * r * (p_{gd} - x_{id}) \quad 2-10$$

$$x_{id}' = x_{id} + v_{id} \quad 2-11$$

dimana i adalah partikel ke- i , d adalah dimensi ke- d , φ_1 adalah laju belajar (*learning rate*) untuk komponen *cognition* (kecerdasan individu), φ_2 adalah laju belajar untuk komponen *social* (hubungan sosial antar individu), p adalah vektor nilai *fitness* terbaik yang dihasilkan sejauh ini, g adalah indeks dari partikel dengan *fitness* terbaik di dalam topologi ketetanggaan, dan r adalah bilangan acak dalam interval $[0, 1]$. *Velocity* partikel pada setiap dimensi dibatasi pada suatu *velocity* maksimum V_{max} . Jika percepatan akan mengakibatkan *velocity* pada suatu dimensi melebihi V_{max} , maka *velocity* pada dimensi tersebut dianggap sama dengan V_{max} . Nilai batas V_{max} ditentukan oleh user. Pada saat inisialisasi nilai-nilai pada vektor *velocity* dibangkitkan secara acak dalam interval $[-V_{max}, V_{max}]$, dimana V_{max} adalah nilai

maksimum yang dapat diberikan pada v_{id} . Seperti terlihat pada *pseudo code* algoritma PSO pada Gambar 2.3 (Suyanto, 2010).

```

for setiap partikel
    Inisialisasi partikel
end
repeat
    for setiap partikel
        Hitung nilai fitness
        if nilai fitness baru lebih baik daripada nilai fitness lama
            Update nilai fitness dari partikel tersebut
        end
    end
    Pilih partikel dengan nilai fitness terbaik diantara semua partikel
    tetangganya dan simpan nilai fitness terbaik tersebut
    for setiap partikel
        Hitung velocity partikel menggunakan rumus pada persamaan 2-10
        Update posisi partikel menggunakan rumus pada persamaan 2-11
    end
until (kriteria berhenti = true)
    
```

Gambar 2.3 *Pseudo code* Algoritma PSO

Sumber: (Suyanto, 2010)

2.10.3 Implementasi Particle Swarm Optimization (PSO)

Sesuai algoritma PSO diatas untuk memahaminya misalnya ada contoh masalah maksimasi fungsi satu dimensi $f(x)$ yang bernilai kontinu, dimana dalam masalah tersebut digunakan 5 partikel yang diberi nama p_1 sampai p_5 . Pada awalnya partikel dibangkitkan secara acak, kemudian dipilih nilai *fitness*-nya yang paling tinggi (terbaik). Selanjutnya keempat partikel yang lain akan mengubah kecepatan dan posisinya relatif terhadap partikel terbaik tersebut, sehingga kelima partikel tersebut berada pada posisi terbaik yang menjadi solusi (nilai maksimum) yang dicari (Suyanto, 2010). Pada implemantasi yang lain yang ditulis oleh (Santosa, 2014) misalkan terdapat fungsi min $f(x)$ dimana $X^{(B)} \leq X \leq X^{(A)}$ dimana $X^{(B)}$

adalah batas bawah dan $X^{(A)}$ adalah batas atas dari X . Sehingga prosedur PSO dapat dijabarkan sebagai berikut (Santosa, 2014):

1. Diasumsikan bahwa ukuran kawanan (jumlah partikel) adalah N . Untuk mengurangi jumlah evaluasi fungsi yang diperlukan agar menemukan solusi sebaiknya N tidak terlalu besar dan juga tidak terlalu kecil agar banyak kemungkinan mendapatkan posisi yang optimal, dimana ukuran terlalu kecil kemungkinannya sedikit untuk menemukan posisi partikel yang baik sedangkan terlalu besar akan membuat perhitungan menjadi panjang. Disarankan menggunakan ukuran kawanan sekitar 30 partikel karena cukup kecil untuk efisiensi waktu dan sudah cukup besar untuk menghasilkan solusi yang baik (mendekati optimum global) (Suyanto, 2010).
2. Membangkitkan populasi awal X dengan rentang $X^{(B)}$ dan $X^{(A)}$ secara random sehingga didapat $X_1, X_2, \dots, X_N$. Setelah itu misalnya partikel i dan kecepatan pada iterasi j dinotasikan sebagai X_{ij} dan V_{ij} , sehingga partikel-partikel awal akan menjadi $X_{i1}, X_{i2}, \dots, X_{ij}$. Vektor X_{ij} , ($i=1, 2, \dots, N$) disebut partikel atau vektor posisi dari partikel (seperti kromosom pada algoritma genetika). Evaluasi nilai fungsi tujuan untuk setiap partikel dinyatakan dengan $f(P_{1j}), f(P_{2j}), \dots, f(P_{Nj})$, dimana P_{ij} adalah vektor untuk menyimpan posisi dari solusi terbaik oleh partikel tersebut sejauh ini pada partikel ke- i dan iterasi ke- j .
3. Selanjutnya menghitung kecepatan dari semua partikel, dimana semua partikel bergerak menuju titik optimal dengan suatu kecepatan. Awalnya semua kecepatan dari partikel diasumsikan sama dengan nol. Set iterasi $j=1$.
4. Pada iterasi ke- j , temukan 2 parameter penting untuk setiap partikel i yaitu:
 - a. Nilai terbaik sejauh ini dari X_{ij} (posisi partikel i pada iterasi j) dan nyatakan sebagai $P_{best,i}$ dengan nilai fungsi obyektif paling rendah (kasus minimasi), $f(X_{ij})$ yang ditemukan sebuah partikel i pada semua iterasi sebelumnya. Nilai terbaik untuk semua partikel X_{ij} yang ditemukan sampai iterasi ke- j adalah G_{best} , dengan nilai fungsi tujuan paling kecil atau minimum diantara semua partikel untuk semua iterasi sebelumnya $f(X_{ij})$.

- b. Hitung kecepatan partikel i pada iterasi ke- j dengan rumus yang sama dengan Persamaan 2-10 sebagai berikut:

$$V_i(j) = V_i(j-1) + c_1 r_1 [P_{best,i} - x_i(j-1)] + c_2 r_2 [G_{best} - x_i(j-1)], i = 1, 2, \dots, N \quad 2-12$$

Dimana c_1 dan c_2 masing-masing adalah *learning rates* untuk kemampuan individu (*cognitive*) dan pengaruh sosial (*group*), dan r_1 dan r_2 bilangan random yang berdistribusi *uniform* dalam interval $[0, 1]$. Jadi parameter c_1 dan c_2 menunjukkan bobot dari memori (posisi) sebuah partikel terhadap memori (posisi) dari kelompok (*swarm*). Nilai c_1 dan c_2 biasanya adalah 2 sehingga perkalian $c_1 r_1$ dan $c_2 r_2$ memastikan bahwa partikel-partikel akan mendekati target sekitar setengah selisihnya.

- c. Hitung posisi atau posisi partikel i pada iterasi ke- j sama dengan Persamaan 2-11 dengan cara berikut:

$$X_i(j) = X_i(j-1) + V_i(j); i = 1, 2, \dots, N \quad 2-13$$

Untuk evaluasi nilai fungsi tujuan bagi setiap partikel dinyatakan sebagai $f(P_{1j}), f(P_{2j}), \dots, f(P_{Nj})$.

5. Cek apakah solusi yang sekarang sudah konvergen atau belum. Jika posisi semua partikel menuju ke satu nilai yang sama, maka ini disebut konvergen. Jika belum maka ulangi langkah 4 dengan memperbaharui iterasi $j = j + 1$, dengan cara menghitung nilai baru dari $P_{best,i}$ dan G_{best} . Proses iterasi ini dilanjutkan sampai semua partikel menuju ke satu titik solusi yang sama, biasanya akan ditentukan dengan kriteria penghentian (*stopping criteria*), misalnya jumlah selisih solusi sekarang dengan solusi sebelumnya sudah sangat kecil, selain itu dengan pemberhentian apabila nilai *fitness* saat ini sudah sama dengan *fitness* terbaik yang tersimpan sebelumnya.

2.11 Time Variant Particle Swarm Optimization (TVPSO)

Time variant yang digunakan adalah *time varying acceleration coefficients* (TVAC) dan *time varying inertia weight* (TVIW). *Time variant* inilah yang digunakan untuk mengontrol kemampuan PSO dalam lokal pencarian secara efisien dan konvergensi ke solusi optimum global (Ratnaweera, 2004). Adapun nilai minimum dan maksimum w (TVIW) yang digunakan adalah 0.4 dan 0.9 nilai ini

terbukti dapat meningkatkan solusi optimum pada pencarian banyak masalah, sedangkan nilai range c_1 dan c_2 (TVAC) yang digunakan adalah [2.5, 0.5] dan [0.5, 2.5] karena terbukti optimal (Ratnaweera, 2004). Seperti yang telah dijelaskan sebelumnya bahwa PSO merupakan algoritma optimasi dimana pada PSO tiap individu sebagai sebuah partikel pada tempat d dimensi dan tiap partikel memiliki sebuah posisi dan kecepatan yang merupakan dasar dari algoritma TVPSO. Vektor posisi dari partikel ke- i direpresentasikan sebagai $X_i = (x_{i,1}, x_{i,2}, \dots, x_{i,d})$, kecepatan direpresentasikan sebagai $V_i = (v_{i,1}, v_{i,2}, \dots, v_{i,d})$ dan n adalah jumlah iterasi. Kecepatan dan posisi sama dengan Persamaan 2-10, 2-11, 2-12, 2-13 hanya saja di perbaharui sebagai berikut dengan menambahkan bobot *inertia* w (Chen at all, 2011):

$$v_{i,j}^{n+1} = w \times v_{i,j}^n + c_1 + r_1(p_{i,j}^n - x_{i,j}^n) + c_2 + r_2(p_{g,j}^n - x_{i,j}^n) \quad 2-14$$

$$x_{i,j}^{n+1} = x_{i,j}^n + v_{i,j}^{n+1}, j=1,2,\dots,d \quad 2-15$$

dimana vektor ($P_i = p_{i,1}, p_{i,2}, \dots, p_{i,d}$) merepresentasikan posisi terbaik sebelumnya dari partikel ke- i yang memberi nilai *fitness* terbaik, yang biasanya di sebut *personal best position* (*pbest*). ($P_g = p_{g,1}, p_{g,2}, \dots, p_{g,d}$) adalah partikel terbaik diantara semua partikel dalam populasi, yang biasanya disebut *global best position* (*gbest*). r_1 dan r_2 adalah angka random yang dihasilkan secara seragam dalam *range* antara 0 sampai 1. Kecepatan $v_{i,j}$ pembatas ke *range* $[-v_{max}, v_{max}]$ (Chen at all, 2011).

Variabel v_{max} menentukan perubahan maksimum yang bisa dilakukan oleh suatu partikel dalam satu iterasi, v_{max} biasanya diset sama dengan rentang nilai partikel. Misalnya suatu partikel i berada pada posisi $X_i = (x_1, x_2, x_3)$ dengan x_1 berada dalam rentang $[-10, 10]$, x_2 dalam rentang $[-5, 5]$, dan x_3 dalam rentang $[-3, 3]$, sehingga untuk partikel tersebut V_{max} yang digunakan untuk x_1 adalah 20 (dari 10-(-10)), V_{max} yang digunakan untuk x_2 adalah 10, V_{max} yang digunakan untuk x_3 adalah 6 (Suyanto, 2010). Dalam buku tersebut juga dijelaskan bahwa batasan $[X_{min}, X_{max}]$, ketika partikel mencapai X_{max} maka ubah *velocity*-nya menjadi 0.

Rumus peng-update-an *velocity* yang standar cenderung terjebak pada optimum lokal, sehingga penambahan bobot *inertia* perlu ditambahkan untuk meningkatkan

performansi PSO. Pada awalnya w bernilai 1 dan selanjutnya diturunkan secara bertahap pada setiap iterasi. Ketika w bernilai 1, proses pencarian terjadi secara global ke segala arah. Sedangkan ketika w diturunkan, pencarian menjadi lebih fokus ke suatu area tertentu yang potensial (Suyanto, 2010). Bobot *inertia* w diperbaharui seperti pada persamaan berikut (Chen at all, 2011):

$$w = w_{\min} + (w_{\max} - w_{\min}) \frac{(t_{\max} - t)}{t_{\max}} \quad 2-16$$

dimana $w_{\max}$ dan $w_{\min}$ adalah nilai maksimum dan minimum bobot *inertia* w , t adalah iterasi awal dari algoritma dan $t_{\max}$ adalah nilai maksimum iterasi. Pada Persamaan 2-16 juga disebut sebagai *Time varying inertia weight* (TVIW), yang akan disatukan ke TVPSO. c_1 dan c_2 adalah koefisien percepatan untuk keseimbangan yang lebih baik antara global eksplorasi yaitu *fitness* terbaik yang dicapai oleh semua partikel dalam topologi ketetanggaan dan lokal eksploitasi yaitu *fitness* terbaik yang dicapai oleh satu partikel saat ini, *Time varying acceleration coefficients* (TVAC). Konsep ini akan diadopsi untuk solusi pencarian yang lebih baik. Inti dari TVAC adalah c_1 menurunkan dari nilai inisial c_{1i} sampai c_{1f} , saat c_2 menaikkan dari c_{2i} sampai c_{2f} berdasarkan persamaan TVAC secara matematika sebagai berikut (Chen at all, 2011):

$$c_1 = (c_{1f} - c_{1i}) \frac{t}{t_{\max}} + c_{1i} \quad 2-17$$

$$c_2 = (c_{2f} - c_{2i}) \frac{t}{t_{\max}} + c_{2i} \quad 2-18$$

dimana c_{1f} , c_{1i} , c_{2f} , c_{2i} adalah konstanta, t adalah iterasi awal dari algoritma dan $t_{\max}$ adalah nilai maksimum iterasi. Untuk *binary* PSO, dimana versi PSO yang dikenalkan oleh Kennedy dan Eberhart memiliki satu ciri-ciri tersendiri yang dipergunakan untuk bertindak sebagai alat seleksi fitur. Pada *binary* PSO, sebuah fungsi *sigmoid* digunakan untuk mentransformasi kecepatan dari ruang kontinu untuk ruang probabilitas:

$$\text{sig}(v_{i,j}) = \frac{1}{1 + \exp(-v_{i,j})}, j = 1, 2, \dots, d \quad 2-19$$

Kecepatan yang diperbaharui pada Persamaan 2-14 tetap tidak berubah kecuali yang $x_{i,j}$, $p_{i,j}$, dan $p_{g,j} \in \{0,1\}$, dan agar memastikan bahwa bit dapat mentransfer antara 1 dan 0 dengan probabilitas positif, v_{max} telah dikenalkan untuk membatasi $v_{i,j}$. Posisi partikel baru diperbaharui berdasarkan aturan sebagai berikut:

$$x_{i,j}^{n+1} = \begin{cases} 1, & \text{if } rnd < sig(v_{i,j}) \\ 0, & \text{if } rnd \geq sig(v_{i,j}) \end{cases}, j = 1, 2, \dots, d \quad 2-20$$

dimana $sig(v_{i,j})$ kalkulasi dari Persamaan 2-19, dan rnd adalah angka random yang seragam dalam *range* 0 sampai 1. (Chen at all, 2011).

2.12 Langkah-Langkah Model TVPSO-FKNN Berdasarkan Algoritma PSO Serial

Berdasarkan pada *flowchart* model TVPSO-FKNN untuk prediksi kebangkrutan dapat dilihat langkah-langkah algoritma tersebut pada PSO seri adalah sebagai berikut (Chen at all, 2011):

1. Membangkitkan partikel atau *encode* partikel dengan dimensi $n+2$. Pertama 2 dimensi k dan m yang nilainya kontinu. Sisanya dimensi n adalah fitur penutup Boolean yang direpresentasikan dengan nilai yang berlainan, 1 mengindikasikan bahwa fitur terpilih dan 0 merepresentasikan fitur yang terbuang.
2. Menginisialisasi individu atau partikel dari populasi dengan *feasible random number*. Sementara itu, menentukan parameter PSO yang mencakup batas bawah dan batas atas dari *velocity*, ukuran dari partikel, angka iterasi, dan sebagainya.
3. *Train* model FKNN dengan vektor seleksi fitur yang memiliki nilai 1 pada tiap kelas pada langkah 2.
4. Menghitung nilai *fitness*, dengan menyimpan optimal *global fitness* (g_{fit}), optimal *personal fitness* (p_{fit}), *global* partikel optimal (g_{best}), dan *personal position* optimal (p_{best}). Dimana tinggi nilai AUC pengklasifikasi yang lebih baik akan dipilih, karena partikel dengan nilai AUC tinggi dan nilai dari fitur terpilih kecil dapat memproduksi sebuah nilai *fitness* yang tinggi. Karena itu, kita mengambil kedua dari keseluruhan dengan pertimbangan

didalam gambaran fungsi *fitness*, nilai *fitness* dikalkulasikan dengan mengikuti fungsi sebagai berikut:

$$\begin{cases} f_1 = AUC \\ f_2 = \left(1 - \frac{\sum_{j=1}^n f_{ti}}{n} \right) \\ f = \alpha \times f_1 + \beta \times f_2 \end{cases} \quad 2-21$$

Dimana:

- Variabel AUC (*Area Under ROC Curve*) didalam fungsi pertama *sub-objective* f_1 merepresentasikan area dibawah kurva ROC diperoleh dari pengklasifikasi FKNN melalui *K-fold cross-validation* (CV), dimana $K=5$. Catatan yang ada *5-fold CV* digunakan untuk menentukan parameter yang optimal (mencakup k dan m) yang berbeda dari putaran luar *10-fold CV* yang digunakan untuk mengerjakan estimasi kinerja.
 - Didalam fungsi *sub-objective* f_2 , f_{ti} adalah nilai dari penutup fitur ('1' merepresentasikan fitur yang dipilih dan '0' mengindikasikan bahwa fitur terbuang), n adalah total nilai dari fitur, dan j adalah dimensi.
 - Penjumlahan bobot dari dua fungsi *sub-objective* dipilih sebagai *final* fungsi *sub-objective*. Di dalam fungsi f , variabel α adalah bobot dari akurasi klasifikasi FKNN, β mengindikasikan bobot untuk memilih fitur. Bobot akan didapatkan biasanya pada sebuah nilai yang tepat tergantung atas pentingnya dari fungsi *sub-objective*. Karena kinerja klasifikasi lebih tergantung pada akurasi klasifikasi, karena itu nilai α di set jauh lebih besar dari pada β . Berdasarkan pada percobaan pendahuluan, nilai dari α dan β terambil sebagai 0.85 dan 0.15 secara berturut-turut. Setelah nilai *fitness* didapatkan *global optimal fitness* tersimpan sebagai g_{fit} , *personal optimal fitness* sebagai p_{fit} , *global optimal particle* sebagai g_{best} dan *personal optimal partikel* sebagai p_{best} .
- Meningkatkan nilai iterasi dengan mengeset $i = i + 1$
 - Meningkatkan nilai populasi dengan memperbaharui posisi dan kecepatan dari k , m berdasarkan Persamaan 2-14 dan 2-15 untuk optimasi parameter

dan untuk seleksi fitur berdasarkan Persamaan 2-14, Persamaan 2-19 dan 2-20 pada tiap partikel.

7. Mencoba model klasifikasi FKNN dengan vektor seleksi fitur yang diperoleh pada langkah 6 dan menghitung nilai *fitness* dari tiap partikel berdasarkan pada Persamaan 2-21. Peringatan yang PSO gunakan untuk mengoptimasi tugas-tugas dimana tetangga terdekat k menjadi optimal dengan nilai *integer*. Oleh karena itu sebuah langkah ekstra diambil sebagai rentetan pengkodean nilai k pada nilai *integer* terdekat sebelum partikel dievaluasi.
8. Memperbaharui *personal optimal fitness* (*pf_{it}*) dan *personal optimal position* (*pb_{est}*) dari membandingkan nilai *fitness* saat ini dengan *pf_{it}* disimpan didalam memori. Jika *fitness* saat ini didominasi oleh *pf_{it}* yang tersimpan didalam memori, kemudian menjaga *pf_{it}* dan *pb_{est}* didalam memori. Jika sebaliknya, mengganti *pf_{it}* dan *pb_{est}* didalam memori dengan nilai *fitness* saat ini dan posisi partikel.
9. Dijalankan jika ukuran dari populasi tercapai atau memenuhi, kemudian lanjut ke langkah 10, jika sebaliknya kembali ke langkah 6.
10. Memperbaharui *global optimal fitness* (*g_{fit}*) dan *global optimal partikel* (*g_{best}*) dari membandingkan *g_{fit}* dengan *pf_{it}* optimal dari keseluruhan populasi, jika *pf_{it}* optimal saat ini mendominasi dengan *g_{fit}* tersimpan didalam memori, kemudian menjaga *g_{fit}* dan *g_{best}* didalam memori. Jika sebaliknya ganti *g_{fit}* dan *g_{best}* didalam memori dengan optimal *g_{fit}* dan optimal *g_{best}* saat ini dari keseluruhan populasi.
11. Jika memberhentikan kriteria cukup, kemudian lanjut ke langkah 12. Jika sebaliknya kembali ke langkah 5. Pemberhentian kriteria dilakukan pada nilai iterasi yang mencapai nilai maksimum dari iterasi atau nilai *g_{fit}* tidak meningkat setelah 100 iterasi berturut-turut.
12. Mengambil optimal k , m dan subset fitur yang optimal dari partikel terbaik (*g_{best}*). Selanjutnya dilakukan pengujian pertama yaitu mengeset $K=K+1$, mencoba FKNN dengan (k , m) terbaik dan bagian fitur optimal sebagai inputan pada $K-1$ bagian *training*. Kemudian memprediksi label di sisa 1 bagian uji dengan fitur terseleksi. Jika *K-fold cross validation* berhenti maka

menghitung rata-rata nilai akurasi, sebaliknya jika tidak maka kembali ke awal yaitu mengeset nilai K ulang.

2.13 *K-Fold Cross Validation* (CV)

Dalam mengestimasi tingkat kesalahan terdapat beberapa teknik yang dapat digunakan, salah satunya adalah *K-fold cross validation*. Teknik ini bekerja dengan mengelompokkan antara *data training* dan *data testing* yang saling terpisah dan tidak ada irisan, kemudian akan dilakukan proses pengujian yang dilakukan sebanyak K kali, selanjutnya dirata-rata untuk menghasilkan sebuah nilai (Mubarok, 2013). Teknik ini merupakan pengujian standar yang dilakukan untuk memprediksi *error rate*, dimana tingkat kesalahan pada iterasi yang berbeda dalam pengujian akan dihitung rata-ratanya untuk menghasilkan *error rate* tersebut secara keseluruhan (Hastuti, 2012). Adapun langkah-langkah teknik *K-fold cross validation* adalah sebagai berikut (Mubarok, 2013):

1. Membagi data yang ada menjadi K kelompok
2. Untuk setiap K , buat sejumlah T himpunan data yang memuat semua *data training* kecuali yang berada dikelompok ke- K
3. Kerjakan algoritma yang dimiliki dengan sejumlah *data training*
4. Uji algoritma dengan menggunakan data pada kelompok K sebagai *data testing*
5. Lakukan pencatatan hasil akurasi dari algoritma.

Teknik ini sangat tepat digunakan untuk menentukan nilai yang tepat untuk K . Teknik ini digunakan untuk menghindari *overlapping* (tumpah tindih atau berlebihan) pada *data testing*, karena tidak membutuhkan waktu yang banyak untuk membuat data uji yang ada. Keuntungannya adalah bahwa semua elemen pada basis data digunakan untuk pelatihan sekaligus pengujian. Ada beberapa percobaan untuk menentukan nilai K yang tepat dan ternyata nilai K sebanyak 10 merupakan pilihan terbaik untuk menghasilkan validasi yang akurat, dimana *10-fold cross validation* yaitu mengulang pengujian dengan membagi *data testing* dan *data training* sebanyak 10 kali dan hasil pengukurannya adalah nilai rata-rata dari 10 kali pengujian. Telah terbukti pada penelitian (Mubarok, 2013) tersebut dengan

menggunakan 4, 5, 6, dan 10 untuk nilai K , dihasilkan rata-rata nilai akurasi tertinggi diperoleh dari nilai K sebesar 10.

2.14 Akurasi

Tujuan evaluasi percobaan pada klasifikasi yaitu untuk mengukur keefektifan apakah sistem mengklasifikasi secara benar (Prasetyo, 2014). Akurasi adalah nilai seberapa dekat atau seberapa miripnya nilai hasil pengukuran dengan nilai sebenarnya adapun persamaannya adalah pada Persamaan 2-22 (Selly, 2013).

$$Akurasi = \frac{\sum \text{data testing benar}}{\sum \text{jumlah total data testing}} \times 100\% \quad 2-22$$

Selain dengan menggunakan persamaan diatas evaluasi biasanya juga membutuhkan sebuah matriks yaitu *confusion matrix* untuk memperkirakan objek yang benar dan salah (Fawcett, 2006). Misalkan terdapat suatu permasalahan klasifikasi dengan 2 kelas, setiap pasang data dianggap memetakan satu set elemen $\{p, n\}$ sebagai label kelas positif dan label kelas negatif. Model pengklasifikasi memetakan pasangan data tersebut ke kelas yang diprediksikan. Untuk membedakan kelas sebenarnya dan kelas yang diprediksi, kelas yang diprediksi disimbolkan dengan $\{Y, N\}$. Pemetaan akan menghasilkan empat keluaran yaitu *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). TP merupakan data positif yang terindikasi benar, FP adalah data negatif yang terindikasi salah, TN adalah data negatif yang terindikasi benar, dan FN adalah data positif yang terindikasi salah. Keempat nilai tersebut membentuk sebuah matriks yang disebut *confusion matrix* (Fawcett, 2006).

		True class	
		p	n
Hypothesized class	Y	True Positives (TP)	False Positives (FP)
	N	False Negatives (FN)	True Negatives (TN)
Column Totals:		P	N

Gambar 2.4 Bentuk dari *Confusion Matrix*
Sumber: (Fawcett, 2006)

Beberapa parameter pengukur kinerja ditunjukkan dengan Persamaan 2-23 sampai 2-27.

$$f_p \text{ rate} = \frac{FP}{TN + FP} \quad 2-23$$

$$t_p \text{ rate} = \frac{TP}{TP + FN} \quad 2-24$$

$$\text{precision} = \frac{TP}{TP + FP} \quad 2-25$$

$$\text{recall} = \frac{TP}{TP + FN} \quad 2-26$$

$$\text{accuracy (ACC)} = \frac{TP + TN}{TP + TN + FP + FN} \quad 2-27$$

Untuk kurva ROC *sensitifity* di hitung dengan menggunakan persamaan t_p *rate* atau *recall*, sedangkan untuk nilai *specificity* digunakan rumus $1 - f_p$ *rate* atau *true negatives* dibagi dengan *false positive* ditambah *true negatives* (Hastuti, 2012). Selain pada gambar diatas pada paper (Hastuti, 2012) juga menggambarkan bentuk *confusion matrix* seperti pada Tabel 2.1.

Tabel 2.1 *Confusion Matrix* untuk 2 Kelas (Makhtar, 2011)

PREDICTED	ACTUAL CLASS	
	Class = Yes	Class = No
Class = Yes	A (True Positive-TP)	B (False Positive-FP)
Class = No	C (False Negative- FN)	D (True Negative-TN)

Dimana kelas yang diamati ditampilkan dibagian atas matriks dan kelas yang diprediksi disisi kiri. Setiap sel berisi angka yang menunjukkan berapa banyak kasus yang sebenarnya dari kelas yang diamati untuk diprediksi. Parameter A merupakan jumlah data yang berhasil di klasifikasikan, parameter B adalah jumlah data yang bukan kelas tetapi sistem mengklasifikasikan ke dalam kelas tersebut, parameter C adalah jumlah data yang mempunyai kelas tetapi sistem tidak mengkalsifikasikannya kedalam kelas tersebut, dan parameter D adalah jumlah data yang tidak termasuk pada kelas dan sistem juga tidak mengklasifikasikannya kedalam kelas tersebut (Prasetyo, 2014).

Sedangkan untuk menghitung nilai *sensitifity* dan *specificity* untuk kurva ROC dengan 3 kelas yaitu berdasarkan *confusion matrix* dibawah ini.

Tabel 2.2 *Confusion Matrix* untuk 3 Kelas (Hermaduanti dan Dewi, 2008)

Kelas Aktual	Kelas Prediksi		
	Kelas 1	Kelas 2	Kelas 3
Kelas 1	$C_{1,1}$	$C_{1,2}$	$C_{1,3}$
Kelas 2	$C_{2,1}$	$C_{2,2}$	$C_{2,3}$
Kelas 3	$C_{3,1}$	$C_{3,2}$	$C_{3,3}$

Dimana $C_{1,1}$ adalah *True Positive* kelas 1, $C_{1,2}$ adalah kelas 1 masuk kelas 2, $C_{1,3}$ adalah kelas 1 masuk kelas 3, $C_{2,1}$ adalah kelas 2 masuk kelas 1, $C_{2,2}$ adalah *True Positive* kelas 2, $C_{2,3}$ adalah kelas 2 masuk kelas 3, $C_{3,1}$ adalah kelas 3 masuk kelas 1, $C_{3,2}$ adalah kelas 3 masuk kelas 2, dan $C_{3,3}$ adalah *True Positive* kelas 3.

Secara prinsip sama dengan *confusion matrix* 2 kelas, hanya saja TP yang digunakan adalah Persamaan 2-28, TN pada Persamaan 2-29, FP pada Persamaan 2-30, dan FN pada Persamaan 2-31 untuk rumus *sensitifity* atau TPR, *specificity* atau TNR, dan FPR sama dengan pada *confusion matrix* 2 kelas. Misalnya ada 3 kelas yaitu kelas “atas”, “menengah”, dan “bawah”, dimana n jumlah kelas, i adalah kelas aktual dan j adalah kelas prediksi C adalah *confusion matrix*.

a. TP (*True Positive*)

Apabila kelas aktual dan sistem sama menghasilkan kelas positif, jika:

- Kelas aktual “Atas” sistem juga memutuskan kelas “Atas”.
- Kelas aktual “Menengah” sistem juga memutuskan kelas “Menengah”.
- Kelas aktual “Bawah” sistem juga memutuskan kelas “Bawah”.

Adapun persamaannya terlihat pada Persamaan 2-28 (Hermaduanti dan Dewi, 2008).

$$TP = C_{1,1} + C_{2,2} + C_{3,3} \quad 2-28$$

b. TN (*True Negative*)

Apabila kelas aktual dan sistem sama menghasilkan kelas negatif, jika:

- Kelas aktual tidak kelas “Atas” sistem juga memutuskan kelas tidak kelas “Atas”.
- Kelas aktual tidak kelas “Menengah” sistem juga memutuskan kelas tidak kelas “Menengah”.

- Kelas aktual tidak kelas “Bawah” sistem juga memutuskan kelas tidak kelas “Bawah”.

Adapun persamaannya terlihat pada Persamaan 2-29 (Hermaduanty dan Dewi, 2008).

$$TN = (C_{1,1} + C_{2,2}) + (C_{1,1} + C_{3,3}) + (C_{2,2} + C_{3,3}) \quad 2-29$$

c. FP (*False Positive*)

Apabila kelas aktualnya negatif, tetapi sistem memutuskan kelas positif, jika:

- Kelas aktual tidak kelas “Atas”, tetapi sistem memutuskan kelas “Atas”.
- Kelas aktual tidak kelas “Menengah”, tetapi sistem memutuskan kelas “Menengah”.
- Kelas aktual tidak kelas “Bawah”, tetapi sistem memutuskan kelas “Bawah”.

Adapun persamaannya terlihat pada Persamaan 2-30 (Hermaduanty dan Dewi, 2008).

$$FP = (C_{2,1} + C_{3,1}) + (C_{1,2} + C_{3,2}) + (C_{1,3} + C_{2,3}) \quad 2-30$$

d. FN (*False Negative*)

Apabila kelas aktualnya positif, tetapi sistem memutuskan kelas negatif, jika:

- Kelas aktual “Atas”, tetapi sistem memutuskan tidak kelas “Atas”.
- Kelas aktual “Menengah”, tetapi sistem memutuskan tidak kelas “Menengah”.
- Kelas aktual “Bawah”, tetapi sistem memutuskan tidak kelas “Bawah”.

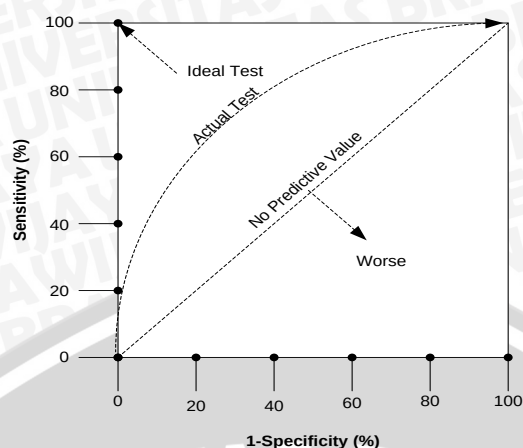
Adapun persamaannya terlihat pada Persamaan 2-31 (Hermaduanty dan Dewi, 2008).

$$FN = (C_{1,2} + C_{1,3}) + (C_{2,1} + C_{2,3}) + (C_{3,1} + C_{3,2}) \quad 2-31$$

2.15 Receiver Operating Characteristic (ROC) Curve

Kinerja suatu sistem verifikasi dapat diukur dengan menggunakan nilai kesalahan yang terjadi dan dapat pula diukur dari seberapa suksesnya suatu sistem (*specificity*). Salah satu metode yang dapat digunakan untuk menghitung nilai kesalahan dan nilai kesuksesan suatu sistem adalah ROC dan untuk nilai kerjanya dapat diukur dengan menggunakan perhitungan daerah dibawah kurva ROC atau biasa disebut dengan AUC. ROC adalah sebuah metode yang digunakan untuk menguji kemampuan sistem prediksi berdasarkan tabel kontingensi dengan menampilkan *skill* sistem prediksi dengan membandingkan *hit rate* dan *false alarm rate* (Kadarsah, 2010). Kurva ROC saat ini telah banyak digunakan untuk analisis pada bidang *machine learning* dan *data mining* (Fawcett, 2006). Kurva ROC paling sering digunakan untuk mengevaluasi pengklasifikasi karena mempunyai kemampuan evaluasi secara menyeluruh dan cukup baik (Cheng at all, 2010). Berkaitan dengan pembahasan sebelumnya mengenai *confusion matrix* kurva ROC merupakan grafik perbandingan antara *sensitivity* (*tp rate* (TPR)) pada sumbu vertical yaitu proporsi data positif yang terindikasi dengan benar dan $1 - \text{specificity}$ (*fp rate* (FPR)) pada sumbu horizontal yaitu proporsi data negatif yang terindikasi salah sebagai positif pada suatu model klasifikasi (Fawcett, 2006).

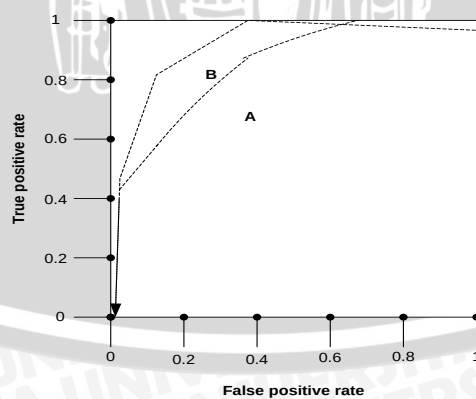
Perbandingan bentuk-bentuk kurva ROC menunjukkan *trade off* antara tingkat dimana suatu model dapat mengenali data negatif sebagai data positif. Seperti yang terlihat pada Gambar 2.5, koordinat (0,1) pada kurva tersebut mewakili nilai *tp rate* 100% yang tidak terdapat *false* negatif atau *fp rate* 100% yang tidak terdapat *false* positif, pada koordinat tersebut bisa juga dikatakan klasifikasi yang sempurna. Untuk kebanyakan prosedur analisis antara titik ideal dan garis *no predictive value* juga merupakan hubungan antara TPR dan FPR (Sparwls, 1995).



Gambar 2.5 Perbandingan Bentuk-bentuk Kurva ROC untuk Kepentingan Diagnostik
Sumber: (Sparwls, 1995)

2.16 Area Under ROC Curve (AUC)

AUC merupakan metode yang umum digunakan untuk merepresentasikan grafis yang menentukan klasifikasi mana yang lebih baik, yaitu menghitung nilai kinerja dari pengklasifikasi dengan menghitung luas daerah dibawah kurva ROC, dimana kurva ROC merupakan grafis dua dimensi dari kinerja suatu pengklasifikasi (Hastuti, 2012). Karena AUC adalah luas kurva dari suatu persegi empat, maka nilainya selalu berada diantara 0 dan 1. Untuk suatu kurva ROC yang memenuhi letaknya selalu berada pada daerah sebelah atas dari garis diagonal (0, 0) dan (1, 1) (Hastuti, 2012).



Gambar 2.6 Perbandingan Dua Buah Kurva ROC A dan B untuk Membedakan Nilai AUC
Sumber: (Fawcett, 2006)

Pada Gambar 2.6 ditunjukkan bahwa contoh dari dua buah kurva ROC, yaitu A dan B, karena pengklasifikasi B luas daerahnya lebih besar daripada pengklasifikasi A, maka pengklasifikasi B memiliki kinerja yang lebih baik dari pengklasifikasi A.

Tabel 2.3 Tabel Kategori Pengklasifikasi Berdasarkan Nilai AUC (Mohanty at all, 2011)

Nilai AUC	Diklasifikasikan sebagai
0.90 - 1.00	Excellent / Paling baik
0.80 - 0.90	Good / Baik
0.70 - 0.80	Fair / Sama
0.60 - 0.70	Poor / Rendah
0.50 - 0.60	Fail / Gagal

Pada Tabel 2.3 jika nilai AUC diatas 0.90 kinerja sistem diklasifikasikan sebagai ‘*Excellent*’ atau ‘Sangat baik’, jika nilai AUC berada antara 0.80 dan 0.90 kinerja sistem adalah ‘*Good*’ atau ‘Baik’, untuk nilai AUC yang berada antara 0.70 dan 0.80 kinerja sistem adalah ‘*Fair*’ atau ‘Sama’, nilai AUC yang berada antara 0.60 dan 0.70 kinerja sistem dikatakan ‘*Poor*’ atau ‘Rendah’, dan apabila nilai AUC dibawah 0.60 maka sistem dianggap gagal mengklasifikasi objek.

BAB III

METODOLOGI PENELITIAN DAN PERANCANGAN

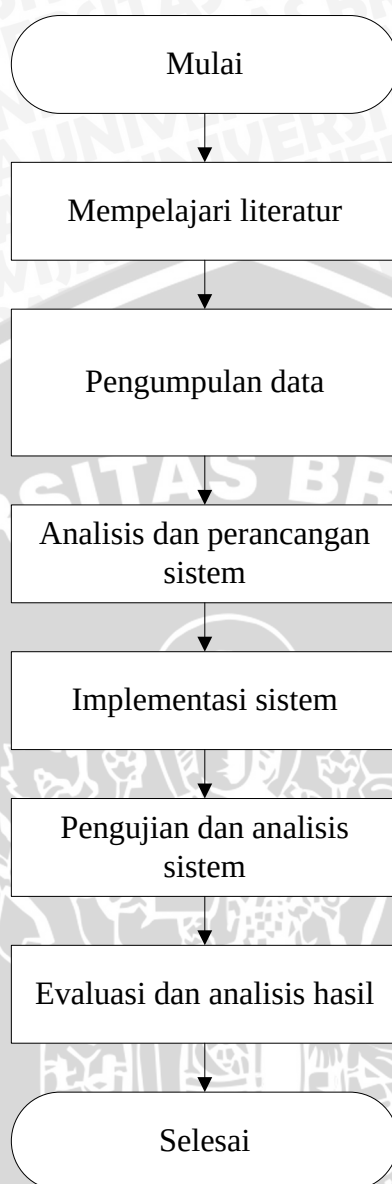
Pada bab metodologi penelitian dan perancangan akan dibahas mengenai metode yang digunakan dan perancangan yang akan dibuat, meliputi langkah-langkah penelitian hingga gambaran penggunaan metode pada implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga.

3.1 Metodologi Penelitian

Adapun metodologi yang digunakan dalam penelitian ini dilakukan dengan tahapan-tahapan sebagai berikut:

1. Mempelajari literatur terkait dengan penelitian yaitu metode *K-Nearest Neighbor*, *Fuzzy K-Nearest Neighbor*, dan *Particle Swarm Optimizaion*.
2. Mengumpulkan dan menganalisis data penelitian yang berupa *dataset* hasil observasi dan *survey* untuk digunakan sebagai *data training* dan *data testing*.
3. Melakukan analisis dan perancangan sistem implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga.
4. Mengimplementasikan hasil dari perancangan yang telah dibuat sebelumnya menjadi sistem klasifikasi yang terkomputerisasi.
5. Melakukan pengujian dan analisis terhadap sistem yang dibuat.
6. Mengevaluasi dan menganalisis tingkat akurasi dari sistem berdasarkan pada hasil *output* yang dihasilkan.

Adapun tahapan dalam penelitian dapat digambarkan dalam bentuk diagram alir pada Gambar 3.1.



Gambar 3.1 Diagram Alir Penelitian

3.1.1 Mempelajari Literatur

Dalam merealisasikan tujuan dan menyelesaikan permasalahan dalam penelitian ini dibutuhkan mempelajari literatur. Pembelajaran ini meliputi teori-teori yang digunakan dalam penelitian misalnya pada pengolahan *dataset* mulai dari data asli yang belum siap digunakan hingga siap digunakan dengan mempelajari metode konversi data dan normalisasi. Kemudian mempelajari mengenai himpunan *fuzzy*, metode *K-Nearest Neighbor*, metode *Fuzzy K-Nearest Neighbor*, dan metode *Particle Swarm Optimizaion* sebagai dasar dari penelitian yang diperoleh dari buku, jurnal penelitian, dan *browsing* dari Internet, serta beberapa literatur lain yang

mendukung yang telah dijabarkan pada bab sebelumnya yaitu bab 2. Kemudian *dataset* penelitian diproses dengan menggunakan metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimizaion* sehingga dapat dianalisis, dan selanjutnya dapat diimplementasikan ke dalam program aplikasi.

3.1.2 Pengumpulan Data

Pengumpulan data dilakukan dengan observasi, wawancara, dan *survey* kepada beberapa keluarga yang bersangkutan yaitu dengan memberikan *form survey* terkait status sosial ekonomi secara *door to door* pada 4 RT yang mengetahui ketua RT dan Kepala Desa setempat. Adapun data diambil dari RT 001, 003, 004, dan 005 pada Dusun Jugo Desa Jugo. Informasi didapatkan tidak hanya langsung dari pihak terkait yaitu kepala keluarga namun juga dari anggota keluarga, ketua RT dan Kepala Desa.

Dari observasi, wawancara, dan *survey* 4 RT tersebut telah didapatkan data sebanyak 121 data keluarga. Kemudian data tersebut dipindahkan ke dalam bentuk *dataset* sesuai dari *form survey*. Selanjutnya adalah menganalisis data terkait atribut yang akan digunakan dan data yang tidak diperlukan. Dari hasil analisis tersebut didapatkan 14 atribut yang akan digunakan dari data hasil observasi, wawancara, dan *survey*, yaitu 9 data kuantitatif dan 5 data kualitatif.

Setelah menganalisis data secara keseluruhan ditemukan beberapa permasalahan terkait pertanyaan pada *form* yaitu data tidak ada nilainya atau *missing value* hal ini disebabkan karena untuk tiap-tiap keluarga memiliki data yang berbeda misalnya terkait dengan kepemilikan tanah, motor, mobil, dan lain sebagainya. Untuk mengatasinya yaitu mengisi angka 0 apabila data kuantitatif, angka 0 dimaksudkan keluarga tersebut tidak memiliki *item* tersebut, sedangkan untuk data kualitatif diisi dengan istilah yang sesuai misalnya pada atribut usaha lain karena ada beberapa keluarga yang tidak memiliki pekerjaan lain dan mengosongkannya maka diisi dengan istilah “tidak ada”. Selanjutnya adalah proses konversi data dari data kualitatif ke data kuantitatif. Hal ini dilakukan karena pada penelitian ini digunakan data kuantitatif untuk implementasi metode. Adapun caranya adalah mengintervalkan yaitu memberi skor pada tiap isi atribut misalnya pada atribut pendidikan memiliki isi tidak tamat SD, SD, SMP, SMA, Diploma, S1,

S2, dan S3. Skor yang diberikan pada tidak tamat SD dan SD yaitu 0. Kemudian SMP, SMA, dan DIPLOMA skor 1. Sedangkan untuk S1, S2, dan S3 skor 2. Dalam menentukan skor penulis memberinya berdasarkan dasar dari literatur yang dibaca berdasarkan tingkat pendidikan.

Dari proses analisis diperoleh data sebanyak 121 data (24 data status sosial ekonomi atas, 52 menengah, dan 45 bawah) dengan jumlah atribut 14 yang sudah berisi data kuantitatif semua. Dalam menentukan data yang akan digunakan, selanjutnya data tersebut akan di normalisasi. Hasil dari normalisasi kemudian dikalikan dengan bobot tiap-tiap atribut. Kemudian untuk digunakan pada implementasi metode data diproses dengan metode *K-fold cross validation* untuk mendapatkan *data testing* dan *data training*.

3.1.3 Analisis dan Perancangan Sistem

Analisis dan perancangan sistem membahas mengenai semua hal terkait yang diperlukan dalam proses pembuatan sistem yaitu mengenai deskripsi sistem, perancangan program aplikasi, perhitungan manual, desain interface, dan perancangan pengujian dan evaluasi. Secara umum sistem yang akan dibangun adalah sebuah sistem perangkat lunak untuk klasifikasi status sosial ekonomi keluarga dengan menggunakan metode *adaptive fuzzy k-nearest neighbor* (FKNN) berdasarkan *particle swarm optimization* (PSO), metode ini digunakan dengan tujuan untuk mengoptimalkan kinerja klasifikasi.

Pada tahap analisis, pertama menyiapkan data kuantitatif atau numerik yang akan digunakan untuk proses klasifikasi FKNN yaitu data keluarga hasil *survey* sebanyak 121 data keluarga. Selanjutnya membuat model pemecahan masalah dengan membuat model metode yang akan digunakan dalam sebuah diagram alir, agar permasalahan mudah untuk dipecahkan. Selanjutnya menganalisis kebutuhan pada proses penelitian, yaitu kebutuhan perangkat lunak dan perangkat keras.

Kebutuhan perangkat lunak yang digunakan meliputi *operating system windows 8 64 bit*, *microsoft office 2013*, *visual studio 2010*, bahasa pemrograman *C#*. Sedangkan kebutuhan perangkat kerasnya adalah *Intel® Core™2 Duo 2.00 GHz*, *RAM 4.00 GB*, *hardisk 320 GB*, dan *Monitor 14"*.

Tahap perancangan sistem merupakan tahap pembuatan alur kerja pemecahan permasalahan klasifikasi menggunakan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization*, penggunaan parameter dan variabel, penggunaan data kedalam algoritma dan formula matematika, desain aplikasi yang menampilkan proses penyelesaian dan hasil akurasi klasifikasi serta desain pengujian dan evaluasi sistem sesuai dengan parameter dan variabel yang berpengaruh terhadap kinerja sistem.

3.1.4 Implementasi Sistem

Implementasi adalah proses pembuatan program aplikasi sesuai dengan analisis dan perancangan sistem yang dilakukan pada tahap sebelumnya. Dimana pemecahan permasalahan klasifikasi diimplementasikan kedalam program aplikasi dengan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization* yang selanjutnya dioperasikan untuk diuji dan dianalisis pada data yang ada.

3.1.5 Pengujian dan Analisis Sistem

Pada tahap ini permasalahan klasifikasi yang telah diimplementasikan dalam bentuk program aplikasi akan diuji tingkat keberhasilannya, kemudian hasil dari uji coba tersebut dievaluasi agar sesuai dengan tujuan yang diharapkan. Adapun pengujiannya meliputi pengujian parameter dan variabel yaitu kombinasi interval untuk parameter m , k , dan jumlah partikel yang digunakan. Pengujian data dengan *K-fold cross validation* dengan kombinasi nilai K , kemudian pengujian dengan banyaknya iterasi yang akan digunakan.

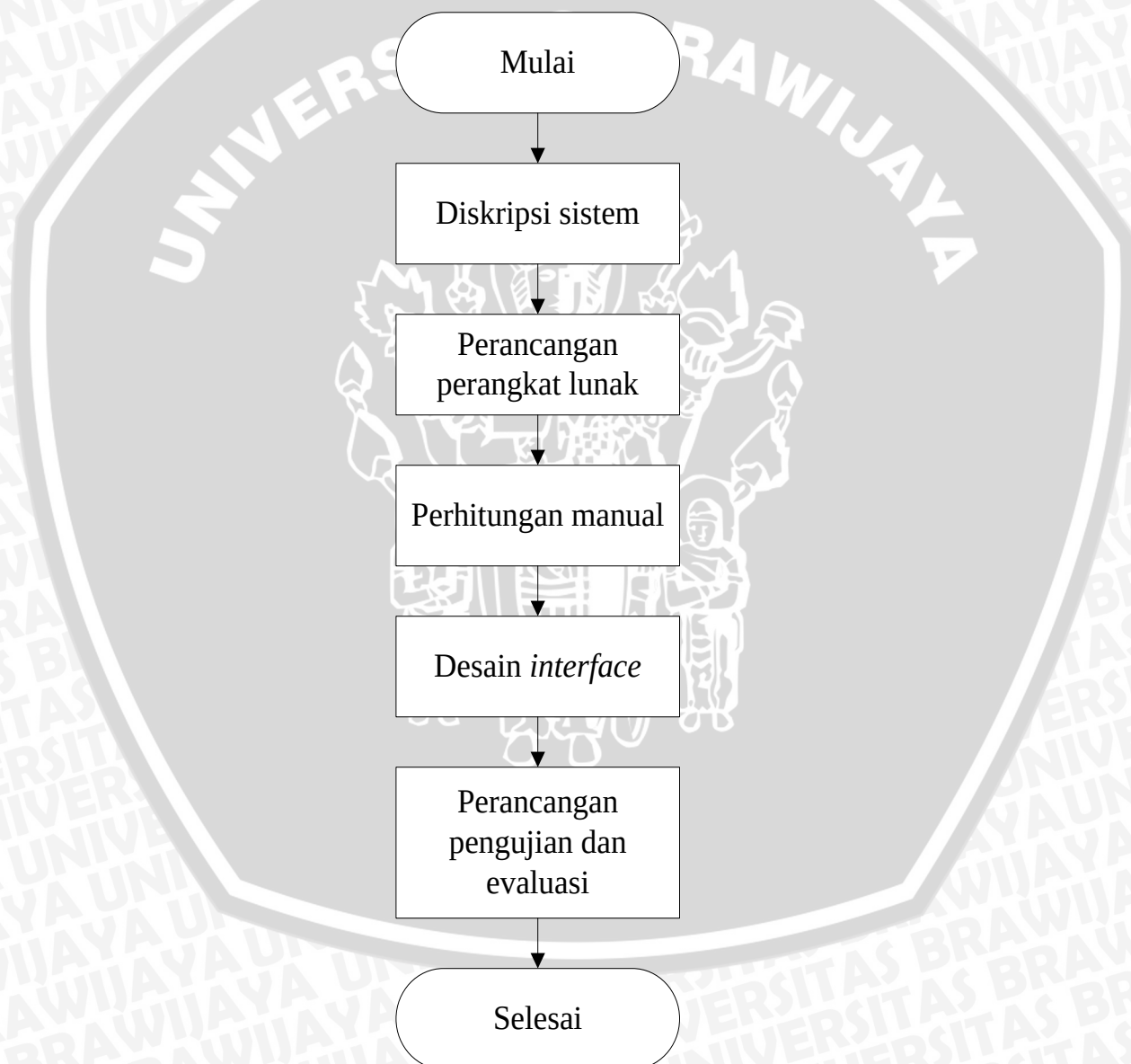
3.1.6 Evaluasi dan Analisis Hasil

Tahap evaluasi dilakukan setelah proses pengujian sistem, evaluasi dilakukan untuk mempertimbangkan tingkat keberhasilan program aplikasi implementasi metode dalam menyelesaikan permasalahan klasifikasi dan apakah kinerja klasifikasi lebih optimal sesuai dengan tujuan penelitian ini. Kemudian adalah tahap akhir dari seluruh rangkaian proses yaitu proses penarikan kesimpulan terhadap hasil yang diperoleh dari proses pengujian dan analisis yang telah

dievaluasi. Selanjutnya kesimpulan dan saran dapat digunakan oleh pembaca sebagai acuan untuk perbaikan serta pengembangan aplikasi kedepannya.

3.2 Perancangan Sistem

Pada permasalahan klasifikasi, adapun tahapan perancangan sistemnya terdiri dari diskripsi umum sistem, perancangan program aplikasi, perhitungan manual, desain *interface*, dan perancangan pengujian dan evaluasi. Gambar 3.2 merupakan model perancangan sistem yang dilakukan pada proses penelitian.



Gambar 3.2 Diagram Alir Tahap Perancangan Sistem

3.2.1 Diskripsi Sistem

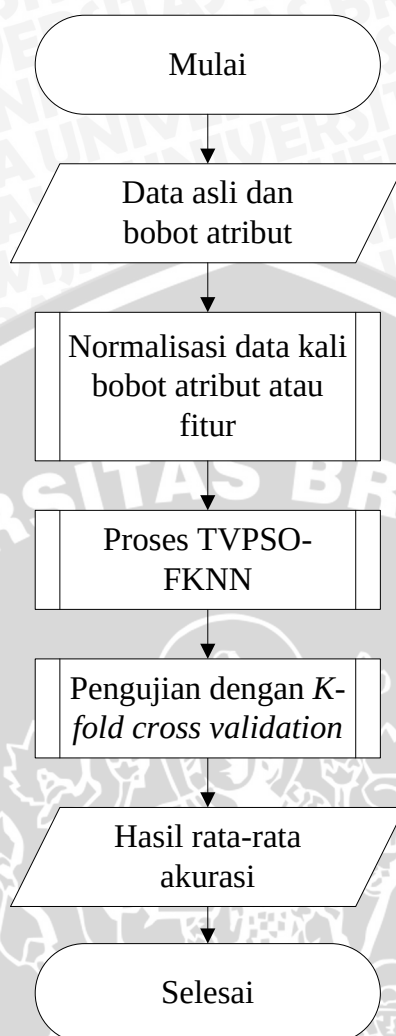
Penelitian ini dilakukan untuk menerapkan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization* pada permasalahan klasifikasi tiga kelas ke dalam sebuah program aplikasi. Permasalahan klasifikasi adalah permasalahan mengelompokkan beberapa data kedalam beberapa kelas dengan menyiapkan *data training* dan *data testing*, dimana metode *fuzzy k-nearest neighbor* digunakan untuk proses klasifikasi yaitu menghitung jarak kedekatan antara *data testing* dengan *data training* dan menghitung nilai keanggotaan *data testing* terhadap tiap-tiap kelas. Sehingga dapat dihasilkan hasil akurasi dari klasifikasi.

Metode *adaptive* pada penelitian ini digunakan untuk mengoptimalkan kinerja proses klasifikasi yaitu untuk memilih nilai m , k , dan fitur terbaik yang dapat menghasilkan akurasi klasifikasi yang lebih baik. Sehingga pada penelitian ini mengadaptifkan metode *fuzzy k-nearest neighbor* kedalam metode *particle swarm optimization*.

3.2.2 Perancangan Perangkat Lunak

Tahap perancangan perangkat lunak pada proses penerapan metode *adaptive fuzzy k-nearest neighbor* (FKNN) berdasarkan *particle swarm optimization* (PSO) untuk permasalahan klasifikasi yang dijalankan pada program aplikasi digambarkan dalam diagram alir sesuai dengan analisis yang dilakukan sebelumnya.

Pada prosesnya terdapat beberapa parameter yang digunakan. Adapun parameter yang digunakan pada klasifikasi FKNN adalah m , dan k . Sedangkan parameter pada PSO adalah dimensi (m , k , dan fitur), *velocity*, posisi, *pbest*, dan *gbest*. Diagram alir sistem yaitu tahapan proses sistem dengan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *time variant particle swarm optimization* (TVPSO-FKNN) ditunjukkan pada Gambar 3.3.

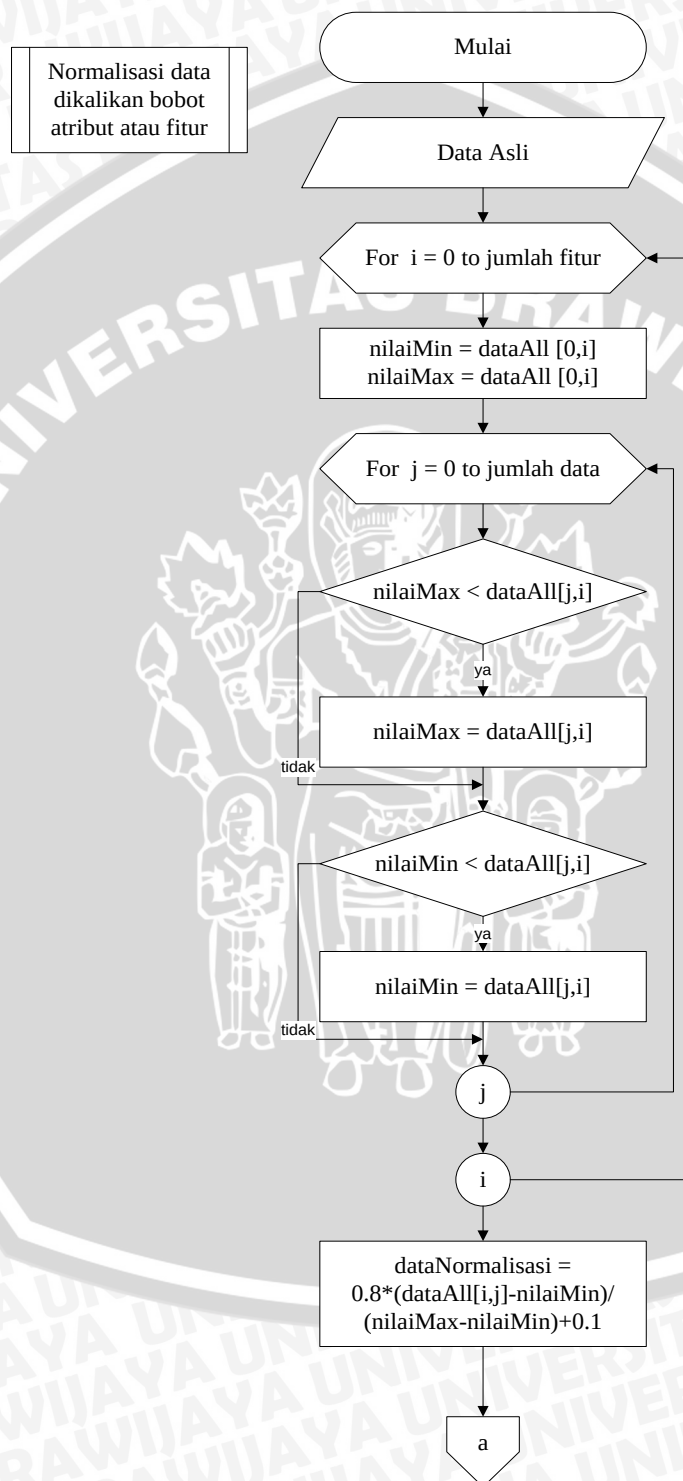


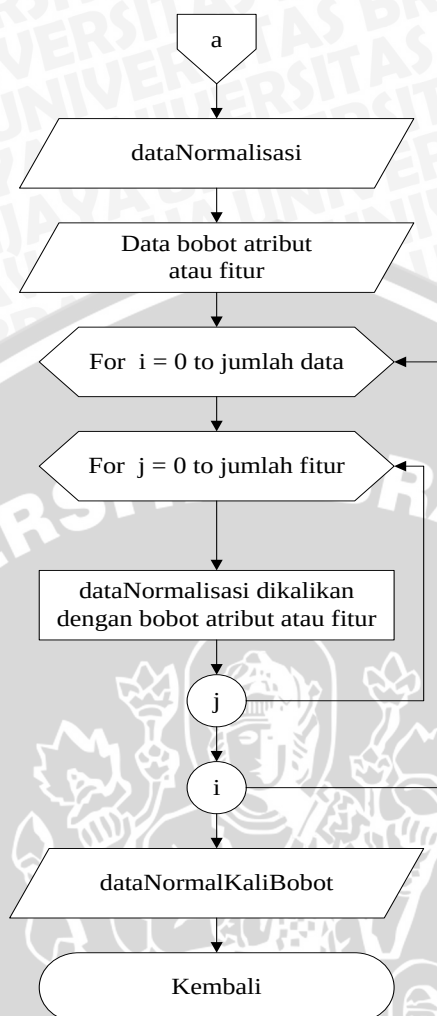
Gambar 3.3 Diagram Alir Sistem

Proses yang dilakukan seperti terlihat pada diagram alir sistem diatas pertama adalah *input* data asli dan bobot atribut, yang kemudian data asli tersebut dinormalisasikan, selanjutnya data setelah dinormalisasikan dikalikan dengan bobot tiap atribut atau fitur. Data yang telah dinormalisasikan dan dikalikan dengan bobot atribut atau fitur kemudian diproses pada proses TVPSO-FKNN hingga mendapatkan partikel terbaik yang berisi nilai m , k , dan fitur terpilihnya. Pada sistem ini tahap akhir yang dilakukan adalah menguji hasil dari proses TVPSO-FKNN dengan *K-fold cross validation* hingga mendapatkan hasil rata-rata akurasi. Pada setiap proses yang diilustrasikan pada Gambar 3.3 terdapat proses *predefined* yang akan dijelaskan lebih rinci pada sub bab selanjutnya.

3.2.2.1 Proses Normalisasi Data Kali Bobot Atribut

Secara garis besar proses normalisasi data dikalikan bobot atribut ditunjukkan pada Gambar 3.4.





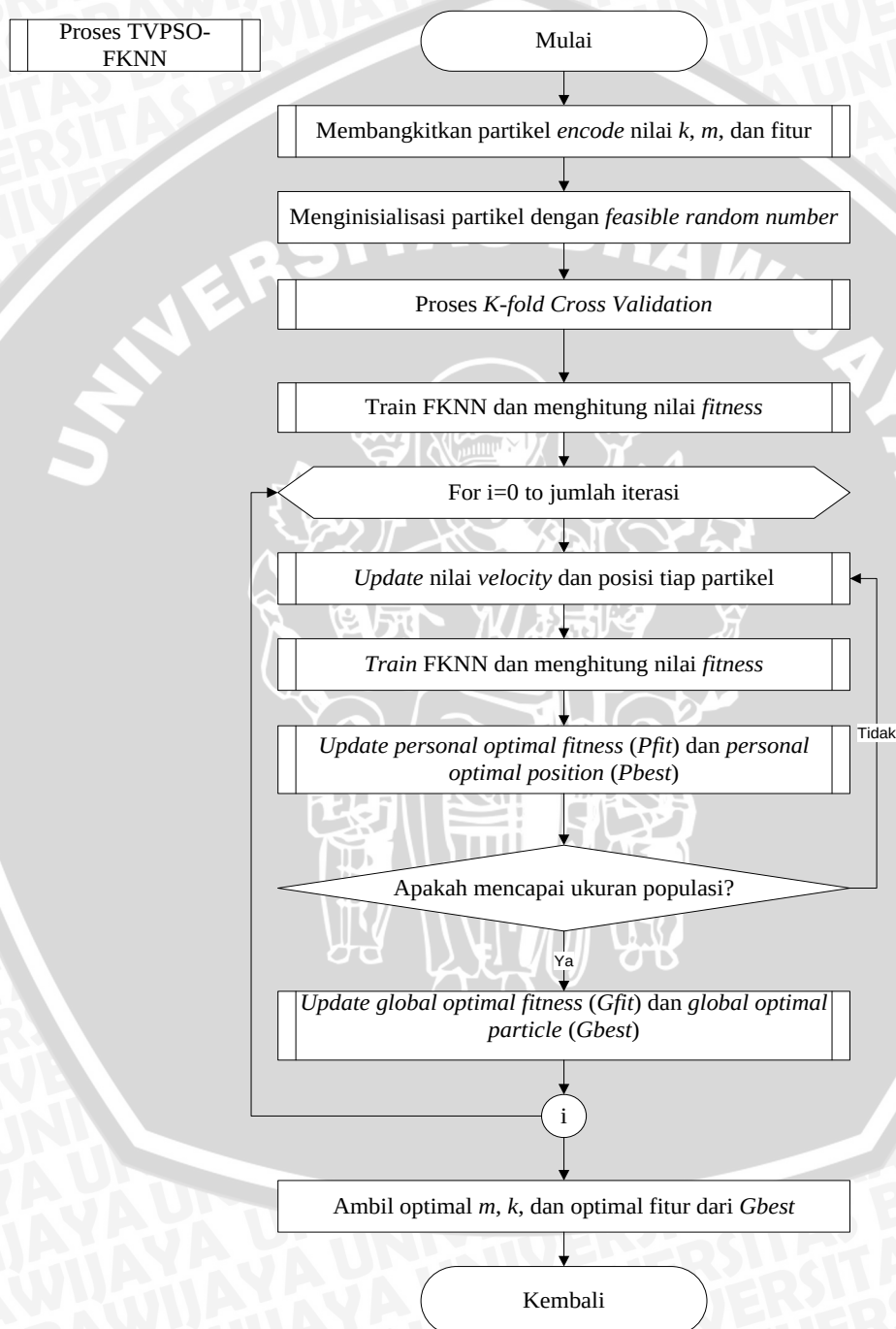
Gambar 3.4 Diagram Alir Proses Normalisasi Dikalikan Bobot Atribut

Adapun proses normalisasi kali bobot atribut yang ditunjukkan pada Gambar 3.4 dapat dijabarkan sebagai berikut:

1. Inputkan data asli yang akan dinormalisasikan
2. Mencari nilai minimum dan nilai maksimum dari keseluruhan data pada tiap-tiap fitur
3. Hitung normalisasi dengan Persamaan 2-3.
4. Setelah mendapatkan data hasil normalisasi selanjutnya mengalikan data hasil normalisasi dengan bobot atributnya dengan persamaan berikut.

3.2.2.2 Adaptive Fuzzy K-Nearest Neighbor Berdasarkan Pada Particle Swarm Optimization

Secara garis besar, proses metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization* diilustrasikan pada Gambar 3.5 berikut.



Gambar 3.5 Diagram Alir Proses TVPSO-FKNN

Secara lebih rinci Gambar 3.5 dapat dijabarkan sebagai berikut:

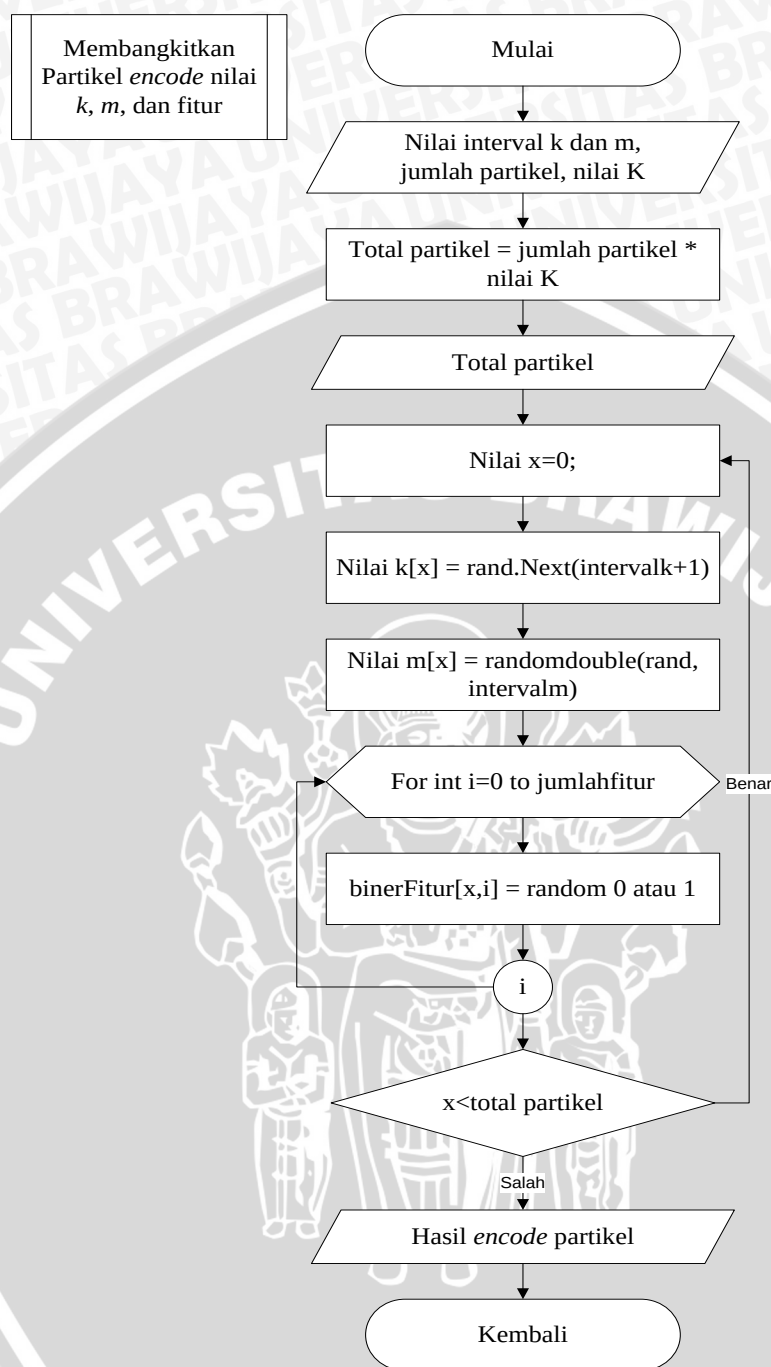
1. Membangkitkan partikel dengan *encode* nilai m , k , dan fitur. Pertama 2 dimensi yaitu k dan m yang nilainya kontinu. Sisanya dimensi n adalah fitur penutup Boolean yang direpresentasikan dengan nilai yang berlainan, 1 mengindikasikan bahwa fitur terpilih dan 0 merepresentasikan fitur yang terbuang.
2. Menginisialisasi individu atau partikel dari populasi dengan *feasible random number*. Sementara itu, menentukan parameter PSO yang mencakup ukuran dari partikel, angka iterasi, dan sebagainya.
3. Proses *K-fold cross validation* merupakan proses mempersiapkan data untuk selanjutnya di gunakan untuk *train* FKNN. Proses ini dilakukan untuk membagi data sebanyak K dan membangkitkan populasi yang berisi n partikel sebanyak K juga.
4. *Train* model FKNN dengan vektor seleksi fitur yang memiliki nilai 1 pada tiap fitur pada langkah 2 dan menghitung nilai *fitness*, dengan menyimpan optimal *global fitness* (g_{fit}), optimal *personal fitness* (p_{fit}), *global* partikel optimal (g_{best}), dan *personal position* optimal (p_{best}). Dimana nilai *fitness* dikalkulasikan dengan mengikuti fungsi Persamaan 2-21.
5. Membangkitkan iterasi dengan mengeset $i = i + 1$
6. Membangkitkan populasi dengan memperbaharui posisi dan kecepatan dari k , m , dan seleksi fitur berdasarkan Persamaan 2-14, Persamaan 2-19 dan 2-20 pada tiap partikel.
7. Mencoba model klasifikasi FKNN dengan vektor seleksi fitur yang diperoleh pada langkah 6 dan menghitung nilai *fitness* dari tiap partikel berdasarkan pada Persamaan 2-21.
8. Memperbaharui *personal optimal fitness* (p_{fit}) dan *personal optimal position* (p_{best}) dari membandingkan nilai *fitness* saat ini dengan p_{fit} disimpan didalam memori. Jika *fitness* saat ini didominasi oleh p_{fit} yang tersimpan didalam memori, kemudian menjaga p_{fit} dan p_{best} di dalam

memori. Jika sebaliknya, mengganti $pfit$ dan $pbest$ didalam memori dengan nilai $fitness$ saat ini dan posisi partikel.

9. Dijalankan jika ukuran dari populasi tercapai atau memenuhi, kemudian lanjut ke langkah 10, jika sebaliknya kembali ke langkah 6.
10. Memperbaharui *global optimal fitness* ($gfit$) dan *global optimal partikel* ($gbest$) dari membandingkan $gfit$ dengan $pfit$ optimal dari keseluruhan populasi, jika $pfit$ optimal saat ini mendominasi dengan $gfit$ tersimpan didalam memori, kemudian menjaga $gfit$ dan $gbest$ didalam memori. Jika sebaliknya ganti $gfit$ dan $gbest$ didalam memori dengan optimal $gfit$ dan optimal $gbest$ saat ini dari keseluruhan populasi.
11. Jika memberhentikan kriteria cukup, kemudian lanjut ke langkah 12. Jika sebaliknya kembali ke langkah 5. Pemberhentian kriteria dilakukan pada nilai iterasi yang mencapai nilai maksimum dari iterasi.
12. Mengambil optimal k , m dan subset fitur yang optimal dari partikel terbaik ($gbest$) yang selanjutnya akan dilakukan pengujian dengan *K-fold cross validation* untuk mendapatkan nilai akurasi tiap-tiap *fold* dengan partikel terpilih yang kemudian dirata-rata hasilnya untuk masing-masing partikel.

A. Proses Membangkitkan Partikel

Proses membangkitkan partikel yaitu proses *encode* partikel dengan menentukan nilai m dan k berdasarkan interval yang telah diinputkan. Kemudian *encode* fitur dengan nilai 1 dan 0, dimana nilai 1 dimaksudkan bahwa fitur tersebut terpilih sedangkan 0 menyatakan fitur tidak terpilih. *Encode* partikel dilakukan sebanyak n partikel kali nilai K . Untuk lebih jelasnya proses tersebut dapat dilihat pada Gambar 3.6.

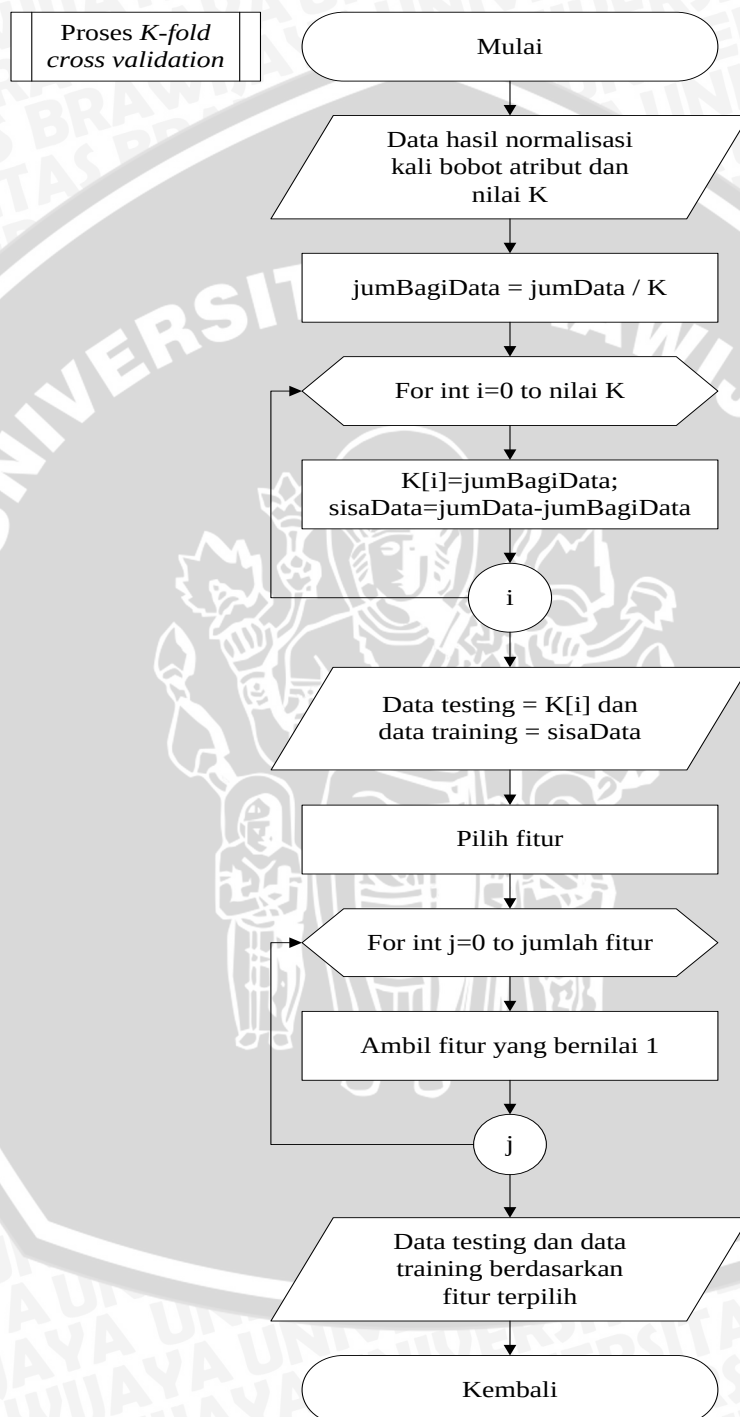


Gambar 3.6 Diagram Alir Proses Membangkitkan Partikel

B. Proses *K-Fold Cross Validation*

Proses *K-fold cross validation* dilakukan untuk mengestimasi tingkat kesalahan dengan cara mengelompokkan antara *data training* dengan *data testing* dari keseluruhan data. Adapun langkah langkahnya adalah pertama menginputkan data hasil normalisasi kali bobot atribut, kemudian set nilai *K*, kemudian membagi

data menjadi K kelompok. Data yang berada pada kelompok K merupakan *data testing*, untuk *data training* diambil dari semua data kecuali data yang berada pada kelompok ke- K . Secara garis besar dapat dilihat pada Gambar 3.7.

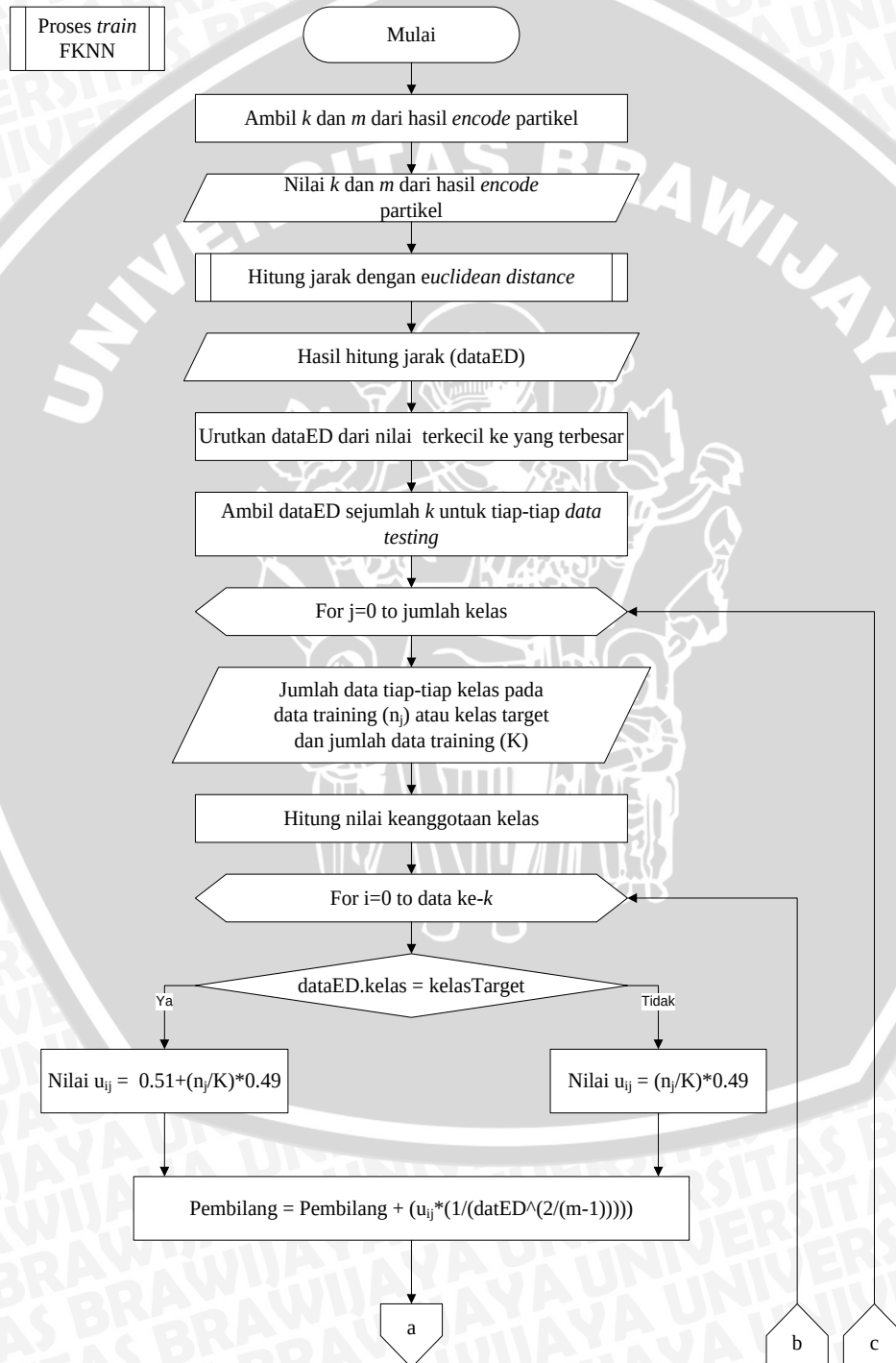


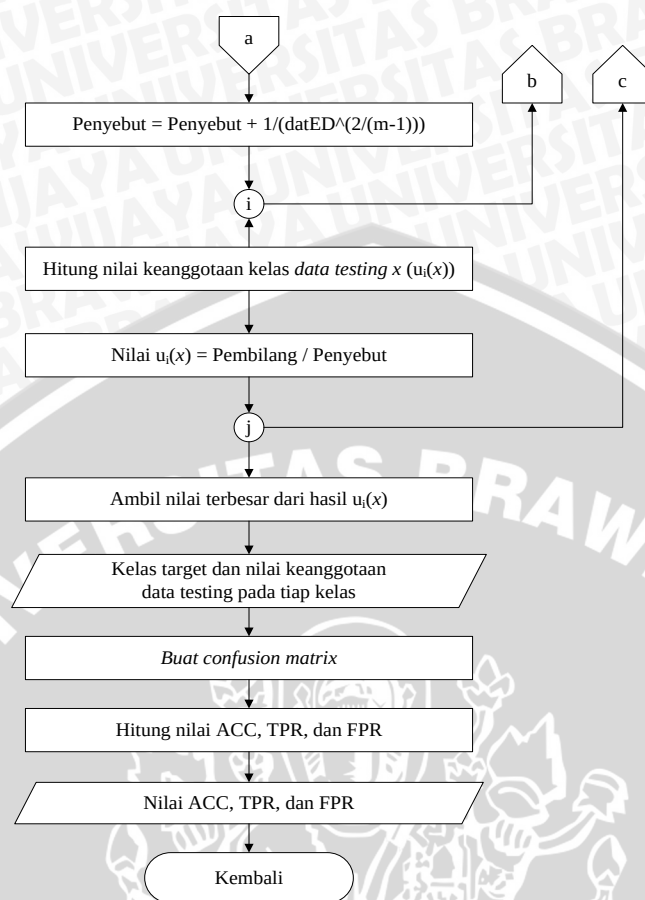
Gambar 3.7 Diagram Alir Proses K -Fold Cross Validation

C. Proses Klasifikasi *Fuzzy k-Nearest Neighbor* (FKNN) dan menghitung

Fitness

Pada proses ini, sebelumnya mengambil nilai m dan k hasil *encode* partikel kemudian menghitung jarak sampai mendapatkan kelas target dan nilai keanggotaan untuk *data testing* x . Secara garis besar dapat dilihat pada Gambar 3.8.



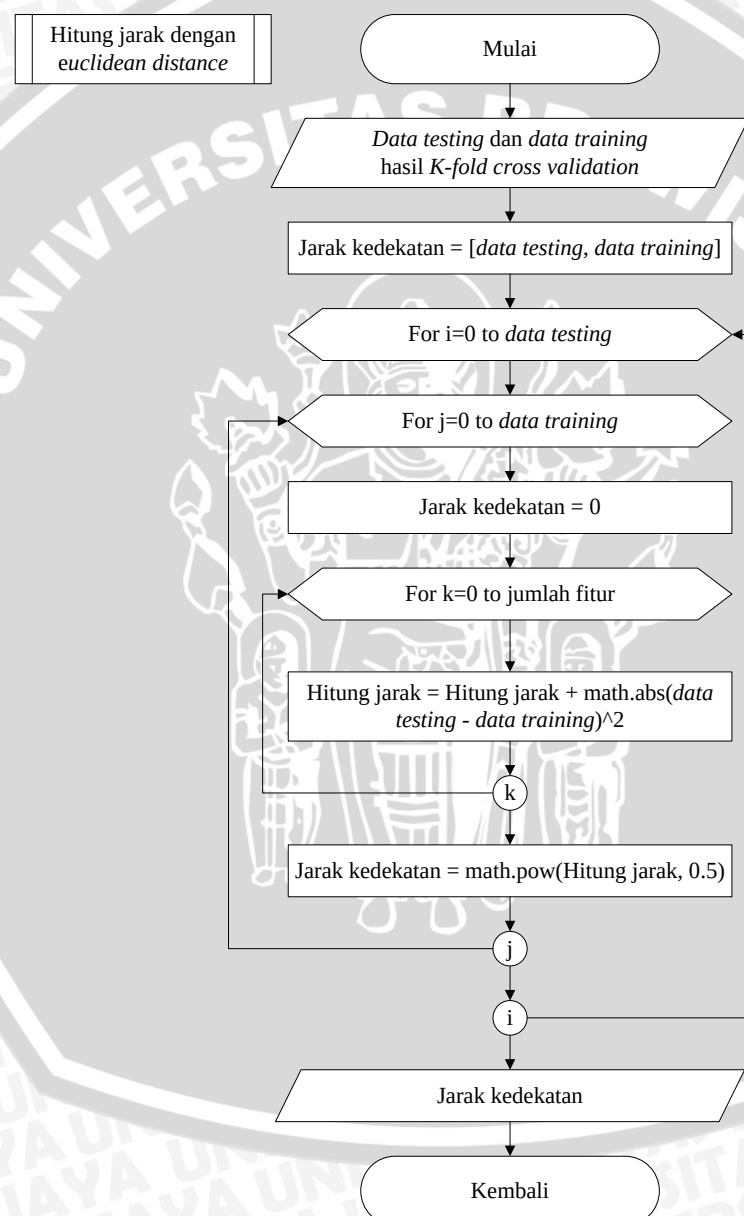


Gambar 3.8 Diagram Alir Proses FKNN

Data hasil *K-fold cross validation* yaitu *data testing* dan *data training* dengan fitur terpilih digunakan sebagai inputan untuk proses hitung jarak dengan *euclidean distance* yaitu jarak antara *data testing* dengan *data training* yang kemudian hasilnya digunakan dalam proses FKNN. Secara rinci langkah-langkahnya adalah sebagai berikut:

1. Ambil nilai *k* dan *m* hasil dari encode partikel.
2. Hitung jarak dengan *euclidean distance* dengan Persamaan 2-5 prosesnya secara garis besar terlihat pada Gambar 3.9.
3. Urutkan hasil penghitungan jarak dengan *euclidean distance* yaitu dari nilai terkecil ke nilai terbesar.
4. Ambil data hasil pengurutan sejumlah *k* untuk tiap-tiap *data testing*.
5. Hitung jumlah data untuk tiap-tiap kelas pada *data training* (n_j) dan hitung jumlah *data training* (*K*).
6. Hitung nilai keanggotaan tiap kelas dengan Persamaan 2-7.

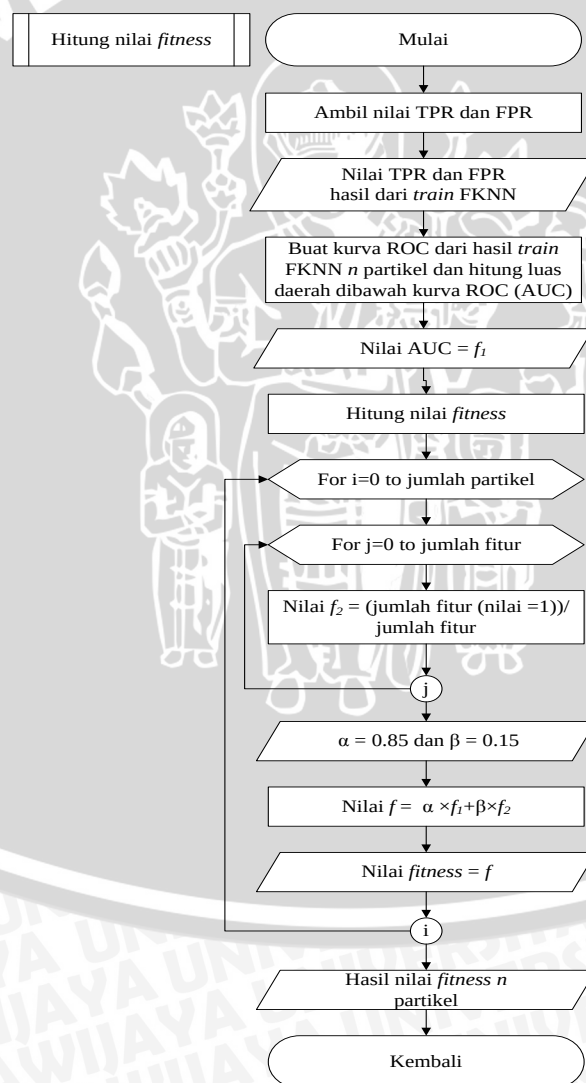
7. Hasil dari nilai kenggotaan kelas digunakan untuk menghitung nilai keanggotaan kelas pada data testing x dengan Persamaan 2-9.
8. Hasilnya adalah kelas target dari *data testing* dan mengambil nilai keanggotaan yang terbesar dari *data testing* terhadap tiap-tiap kelas.
9. Buat *confusion matrix*
10. Hitung nilai *accuracy* (ACC), *sensitivity* (TPR), dan *1-specificity* (FPR)



Gambar 3.9 Diagram Alir Proses Hitung Jarak dengan *Euclidean Distance*

Setelah mendapatkan nilai TPR dan FPR dari proses FKNN, selanjutnya menghitung nilai *fitness* untuk tiap-tiap partikel. Perhitungan nilai *fitness* dimulai dengan membuat kurva ROC kemudian dihitung luas dibawahnya untuk mendapatkan nilai AUC. Nilai AUC sama dengan nilai f_1 , sedangkan nilai f_2 dihasilkan dari menghitung jumlah fitur yang bernilai 1 dibagi jumlah keseluruhan fitur. Nilai *fitness* ini digunakan untuk mendapatkan solusi yang optimal dari partikel terbaik untuk optimasi kinerja proses klasifikasi FKNN, sehingga nilai *fitness* diperoleh dari menguji partikel yang telah dibangkitkan dengan train FKNN. Kemudian nilai *fitness* didapatkan dari Persamaan 2-21.

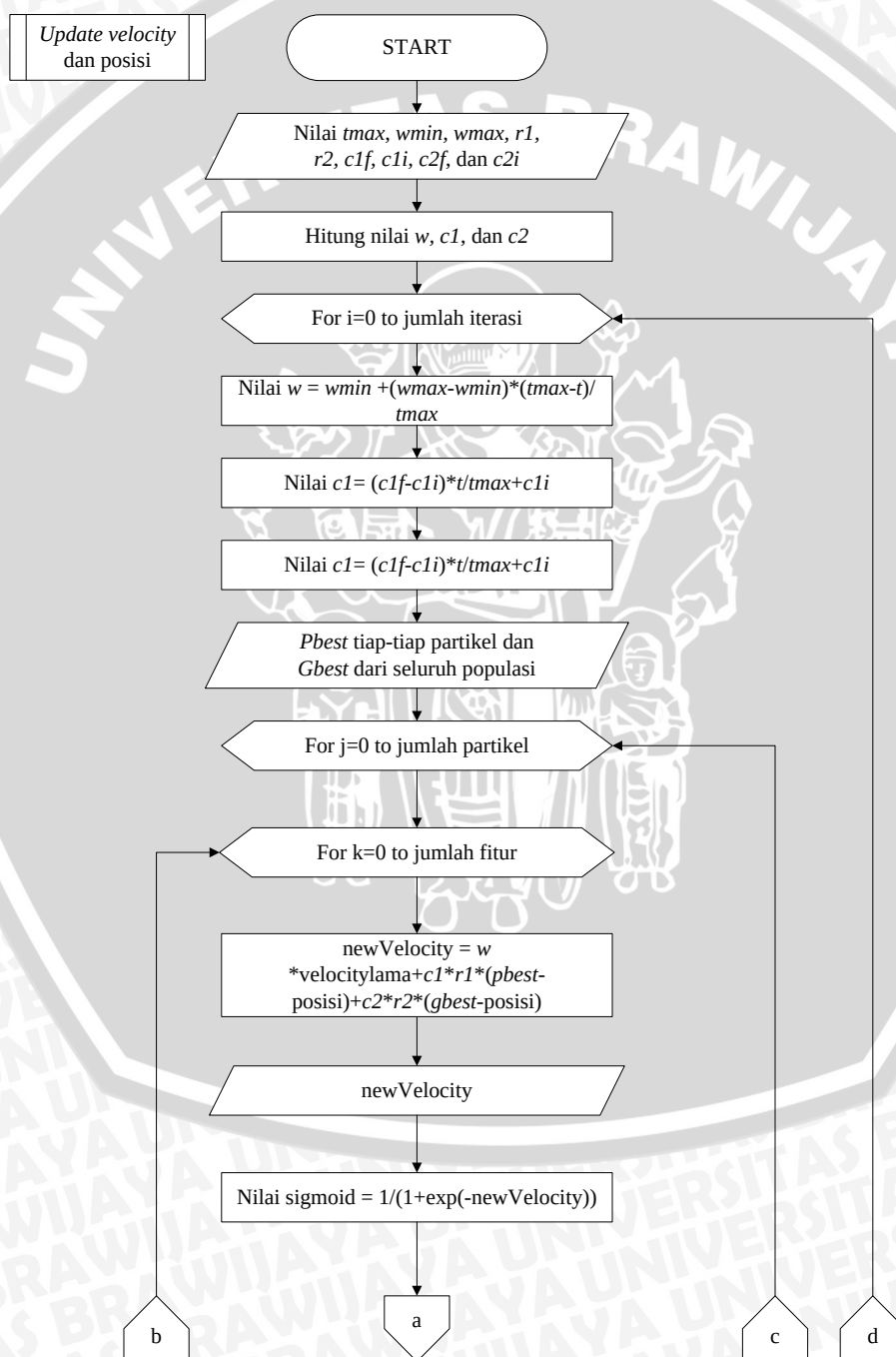
Secara garis besar proses perhitungan *fitness* dapat dilihat pada Gambar 3.10.

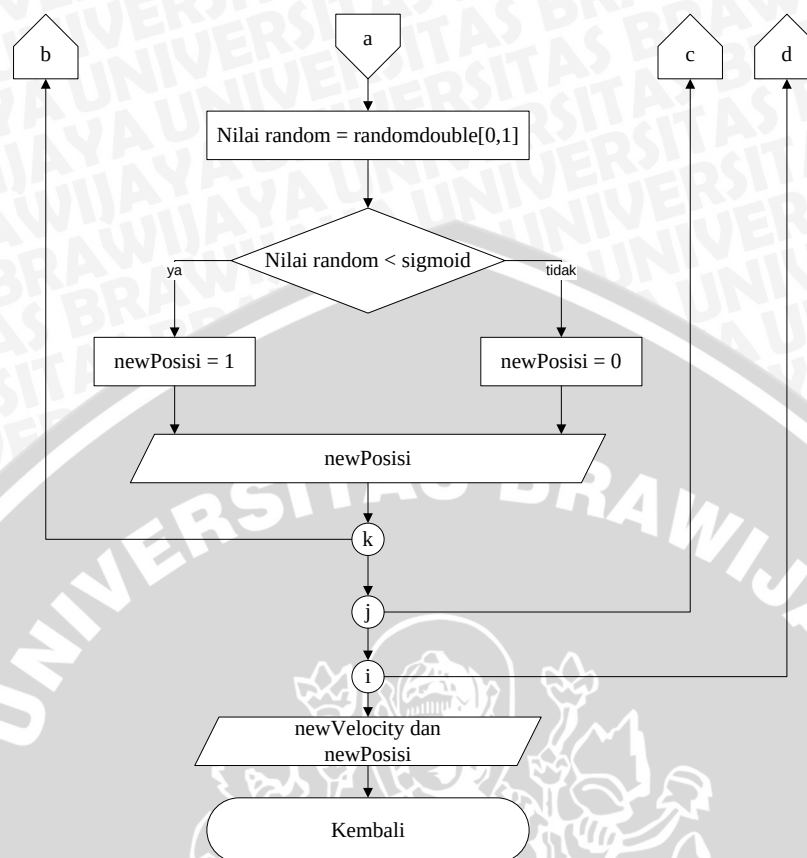


Gambar 3.10 Diagram Alir Proses Hitung Nilai *Fitness*

D. Proses Update Nilai Velocity dan Posisi

Proses *update velocity* diawali dengan menginisialisasi dan menghitung nilai w , c_1 dan c_2 . Kemudian ambil $pbest$ tiap-tiap partikel dan $gbest$ dari seluruh populasi. Selanjutnya *update velocity*, hitung nilai *sigmoid*, membangkitkan nilai random $[0, 1]$, dan mengupdate nilai posisi 1 atau 0. Secara garis besar dapat dilihat pada Gambar 3.11.



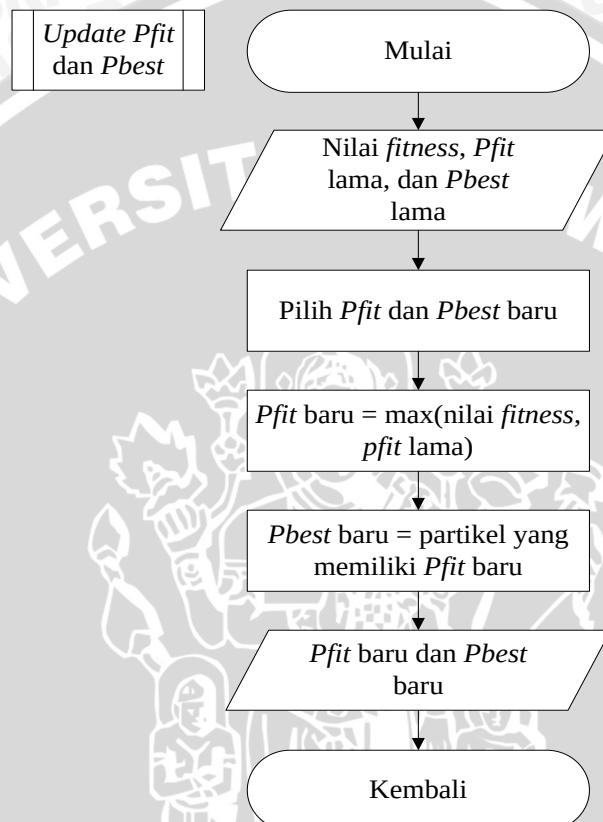


Gambar 3.11 Diagram Alir Proses *Update* Nilai *Velocity* dan Posisi
Secara lebih rinci proses pada Gambar 3.11 dijabarkan sebagai berikut:

1. Pertama inialisasi nilai maksimum iterasi (t_{max}), kemudian nilai minimum dan maksimum w yang masing masing adalah $w_{min} = 0.4$ dan $w_{max} = 0.9$. Selanjutnya inialisasi nilai r_1 dan r_2 adalah 0.5, nilai c_{1f} dan c_{1i} adalah 0.5 dan 2.5, untuk nilai c_{2f} dan c_{2i} adalah 2.5 dan 0.5.
2. Kemudian hitung nilai w dengan Persamaan 2-16
3. Kemudian hitung nilai c_1 dengan Persamaan 2-17 dan c_2 pada Persamaan 2-18
4. Ambil $pbest$ tiap-tiap partikel dan $gbest$ dari seluruh populasi
5. Kemudian *update velocity* dengan Persamaan 2-14
6. Hitung nilai *sigmoid* dengan Persamaan 2-19 dan bangkitkan nilai random $[0, 1]$ untuk semua partikel.
7. Selanjutnya *update* posisi dengan nilai biner yaitu 0 atau 1 sesuai dengan Persamaan 2-20.
8. Mendapatkan *velocity* dan posisi baru.

E. Proses Update Pfit dan Pbest

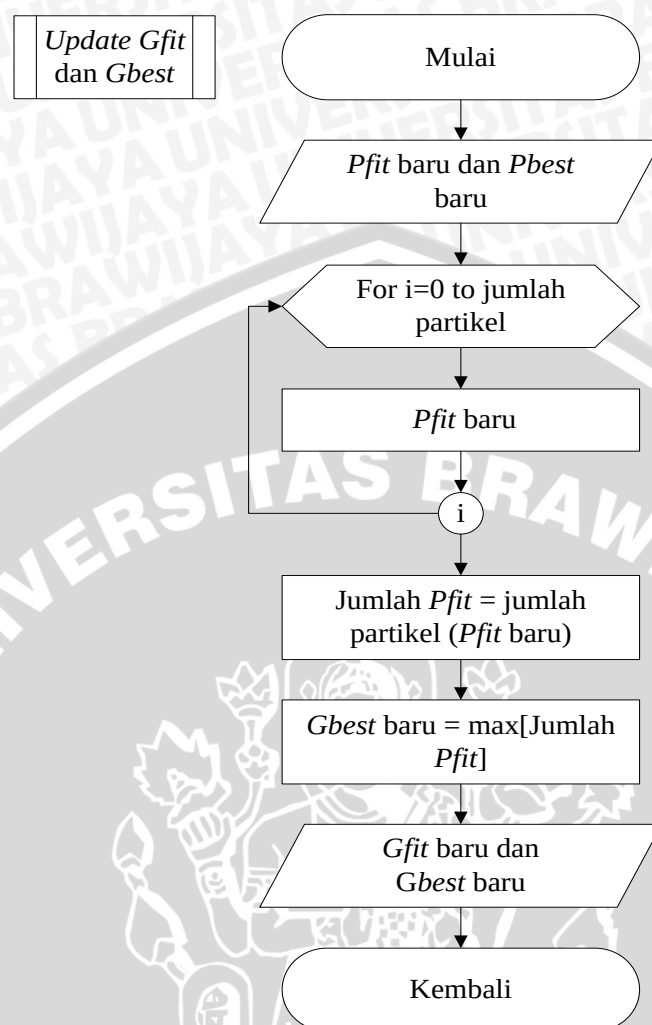
Proses *update pfit* adalah proses mencari nilai maksimum antara nilai *fitness* baru dengan *pfit* lama. Kemudian *pfit* baru adalah nilai *fitness* dari *pbest*. Sehingga dengan mendapatkan nilai *pfit* baru sama dengan mendapatkan *pbest*. Secara garis besar dapat dilihat pada Gambar 3.12.



Gambar 3.12 Diagram Alir Proses *Update Pfit dan Pbest*

F. Proses Update Gfit dan Gbest

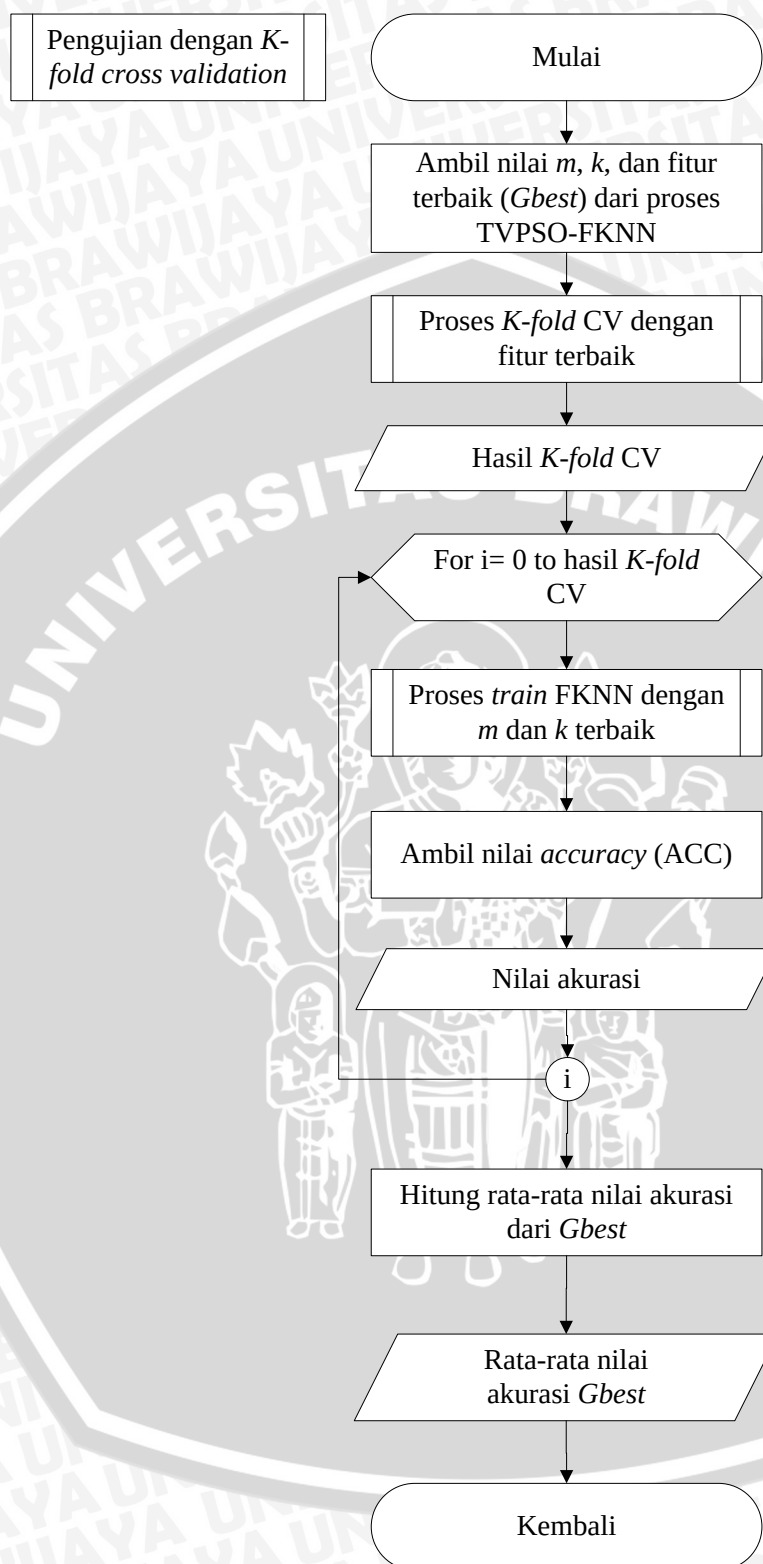
Proses *update gfit* adalah proses mencari nilai maksimum *pfit* baru dari seluruh partikel, dimana *gbest* didapatkan dari partikel yang memiliki *gfit*. Secara garis besar prosesnya dapat dilihat pada Gambar 3.13.



Gambar 3.13 Diagram Alir Proses *Update Gfit* dan *Gbest*

3.2.2.3 Pengujian dengan *K-Fold Cross Validation*

Pengujian dilakukan untuk menguji nilai akurasi dari hasil proses TVPSO-FKNN dengan melakukan *train* FKNN kembali dengan *K-fold cross validation*. Tahapannya adalah pertama ambil kembali data hasil normalisasi kali bobot atribut, mengeset nilai *K*, kemudian membagi data menjadi *K* kelompok, setelah mendapatkan *data testing* dan *data training fold ke-K*, kemudian *train* FKNN kembali dengan *k*, *m*, dan fitur terbaik dari *gbest* dari hasil TVPSO-FKNN, mendapatkan nilai akurasi, ulangi hingga *K-fold* terpenuhi. Kemudian hitung rata-rata nilai akurasinya. Secara garis besar proses tersebut dapat dilihat pada Gambar 3.14.



Gambar 3.14 Diagram Alir Proses Pengujian dengan *K-Fold Cross Validation*

3.2.3 Perhitungan Manual

Tabel diatas merupakan tabel bobot kriteria, dimana bobot kriteria tersebut didapatkan dari pihak kelurahan yaitu Kepala Desa Jugo. Bobot kriteria ini diberikan karena dalam penilaian status sosial ekonomi keluarga tiap atribut yang digunakan memiliki bobot masing-masing, terdapat atribut yang memang bobotnya tinggi untuk digunakan sebagai penilain namun ada juga yang hanya kecil bobotnya dalam penilaian, misalnya pekerjaan, pendapatan, memiliki bobot tertinggi karena memang atribut tersebut yang lebih menjadi prioritas penilaian. Adapun bobot masing masing atribut dapat dilihat pada Tabel 3.1.

Tabel 3.1 Bobot Atribut

No	Nama Atribut	Label	Bobot	No	Nama Atribut	Label	Bobot
1	Pendidikan	F1	0.0357	8	Luas Rumah (m ²)	F8	0.0357
2	Pekerjaan	F2	0.1429	9	Jumlah Mobil	F9	0.1071
3	Usaha lain	F3	0.0357	10	Jumlah Motor	F10	0.0714
4	Pendapatan	F4	0.1429	11	Luas Tanah/ Sawah/ Ladang (are)	F11	0.0714
5	Pengeluaran	F5	0.0357	12	Jumlah Sapi	F12	0.0714
6	Status Rumah	F6	0.0714	13	Jumlah Kambing	F13	0.0429
7	Keadaan Rumah	F7	0.1071	14	Jumlah Unggas	F14	0.0286

Pada tabel diatas terdapat 4 kolom, dimana kolom “No” sebagai index, kolom “Nama Atribut” sebagai identitas, kolom “Label” untuk memudahkan dalam proses selanjutnya untuk mewakili tiap tiap atribut, dan kolom “Bobot” merupakan bobot masing masing atribut.

Pada perhitungan manual ini untuk sampel data yang digunakan diambil dari *dataset* hasil observasi yang sudah diolah sebanyak 12 data dengan 14 atribut, dimana 8 data digunakan sebagai *data training* dan 4 sebagai *data testing* dengan menggunakan metode *K-Fold Cross Validation*. Adapun keseluruhan data seperti pada Tabel 3.2.

Tabel 3.2 Sampel *Dataset*

No	KK	F1	F2	F3	F4		F11	F12	F13	F14	Kelas
1	60	2	4	0	8,687,000		0	0	0	0	A
2	90	1	3	3	3,500,000		7.5	0	4	0	A
3	81	0	1	0	200,000		0	0	0	7	B
4	24	1	3	3	2,000,000		5	0	3	0	M
5	21	1	4	3	6,000,000		2	0	0	0	A
6	109	1	2	0	2,000,000		0	0	0	0	M
7	14	0	3	3	1,500,000		14	2	0	0	M
8	55	0	1	0	500,000		0	0	0	10	B
9	30	1	3	3	10,000,000		25	0	0	0	A
10	32	0	2	0	2,000,000		7	0	0	0	M
11	80	0	1	0	500,000		0	0	0	10	B
12	115	0	1	0	300,000		0	0	2	0	B

Pada Tabel 3.2 yaitu tabel data sampel terdapat kolom “No” sebagai index, kolom “KK” sebagai identitas, kolom F1 sampai F14 merupakan label yang mewakili tiap tiap atribut dimana berisi nilai dari tiap-tiap atribut, dan kolom “Kelas” merupakan kelas tiap data. Kelas A menyatakan kelas atas, kelas M menyatakan kelas menengah, dan kelas B menyatakan kelas bawah. Detailnya ada pada Lampiran 4.

3.2.3.1 Normalisasi Data

Pada penelitian ini digunakan *sigmoid biner* dengan interval [0.1, 0.9] untuk menormalisasikan data. Hal ini digunakan untuk menghindari hasil bernilai nol, sehingga hasil minimum bernilai 0.1 dan hasil maksimum bernilai 0.9. Dimana nilai *range* diambil dari nilai maksimum dikurangi nilai minimum pada keseluruhan

data. Berikut adalah contoh perhitungannya, misalnya diambil 1 data dari *data training* dengan atribut F1 yaitu nomer 1 yang bernilai 2. Detailnya ada pada Lampiran 5.

Tabel 3.3 Range Normalisasi

	F1	F2	F3	F4		F10	F11	F12	F13	F14
max	2	4	3	10000000		3	25	2	4	10
min	0	1	0	200000		0	0	0	0	0
range	2	3	3	9800000		3	25	2	4	10

$$s' = \frac{0.8(s - a)}{b - a} + 0.1$$

Dimana s' adalah nilai hasil normalisasi, s adalah data awal, a adalah nilai minimum, b adalah nilai maksimum.

$$s' = \frac{0.8(2 - 0)}{2 - 0} + 0.1 = \frac{0.8(2)}{2} + 0.1 = \frac{1.6}{2} + 0.1 = 0.8 + 0.1 = 0.9$$

Tabel 3.4 Hasil Normalisasi

No	KK	F1	F2	F3		F12	F13	F14	Kelas
1	60	0.9000	0.9000	0.1000		0.1000	0.1000	0.1000	A
2	90	0.5000	0.6333	0.9000		0.1000	0.9000	0.1000	A
3	81	0.1000	0.1000	0.1000		0.1000	0.1000	0.6600	B
4	24	0.5000	0.6333	0.9000		0.1000	0.7000	0.1000	M
5	21	0.5000	0.9000	0.9000		0.1000	0.1000	0.1000	A
6	109	0.5000	0.3667	0.1000		0.1000	0.1000	0.1000	M
7	14	0.1000	0.6333	0.9000		0.9000	0.1000	0.1000	M
8	55	0.1000	0.1000	0.1000		0.1000	0.1000	0.9000	B
9	30	0.5000	0.6333	0.9000		0.1000	0.1000	0.1000	A
10	32	0.1000	0.3667	0.1000		0.1000	0.1000	0.1000	M
11	80	0.1000	0.1000	0.1000		0.1000	0.1000	0.9000	B
12	115	0.1000	0.1000	0.1000		0.1000	0.5000	0.1000	B

Pada Tabel 3.4 merupakan tabel hasil normalisasi dengan menggunakan *sigmoid biner* dengan interval [0.1, 0.9]. Kemudian menghitung nilai hasil normalisasi dengan mengalikan bobot tiap atribut. Misalkan diambil dari data

setelah normalisasi untuk *data training* nomer 1 atribut F1 dikalikan dengan bobot atribut F1 sebesar 0.0357, dimana x' adalah nilai hasil normalisasi kali bobot, s' hasil normalisasi, dan w adalah bobot atribut. Detailnya ada pada Lampiran 6.

$$x' = s' \times w = 0.9 \times 0.0357 = 0.0321$$

Tabel 3.5 Hasil Normalisasi Dikalikan Bobot Atribut

No	KK	F1	F2	F3		F12	F13	F14	Kelas
1	60	0.0321	0.1286	0.0036		0.0071	0.0043	0.0029	A
2	90	0.0179	0.0905	0.0321		0.0071	0.0386	0.0029	A
3	81	0.0036	0.0143	0.0036		0.0071	0.0043	0.0189	B
4	24	0.0179	0.0905	0.0321		0.0071	0.0300	0.0029	M
5	21	0.0179	0.1286	0.0321		0.0071	0.0043	0.0029	A
6	109	0.0179	0.0524	0.0036		0.0071	0.0043	0.0029	M
7	14	0.0036	0.0905	0.0321		0.0643	0.0043	0.0029	M
8	55	0.0036	0.0143	0.0036		0.0071	0.0043	0.0257	B
9	30	0.0179	0.0905	0.0321		0.0071	0.0043	0.0029	A
10	32	0.0036	0.0524	0.0036		0.0071	0.0043	0.0029	M
11	80	0.0036	0.0143	0.0036		0.0071	0.0043	0.0257	B
12	115	0.0036	0.0143	0.0036		0.0071	0.0214	0.0029	B

Tabel 3.5 merupakan tabel yang berisi hasil perkalian normalisasi dengan bobot atribut. Detailnya ada pada Lampiran 7.

3.2.3.2 Encode Partikel

Encode partikel adalah membangkitkan partikel sebanyak n partikel, dimana pada partikel tersebut diinisialisasi dengan *feasible random numbers* dan dimensi dibangkitkan dengan $d + 2$ dimensi yaitu 2 untuk k dan m yang bernilai kontinu yang nilainya telah ditentukan dalam sebuah interval misalnya pada perhitungan ini untuk nilai m berada pada interval $[1, 3]$, k berada pada interval $[1, 4]$ dan d dimensi untuk seleksi fitur dengan nilai 0 atau 1 pada tiap partikel, dimana fitur yang bernilai 1 yang akan dipilih untuk di lakukan proses *train* model FKNN. Misalnya disiapkan 3 partikel untuk tiap *fold*, dimana pada perhitungan manual ini ada 3 *fold*. Adapun hasil *encode* partikel seperti pada Tabel 3.6, dimana n adalah

jumlah partikel, k adalah jumlah tetangga terdekat, m adalah bobot pangkat nilai keanggotaan, dan d jumlah dimensi. *Fold #K* merupakan inisialisasi *K-fold* yang dijalankan pada proses TVPSO-FKNN.

Tabel 3.6 Inisialisasi *Encode* Partikel untuk *Fold #1*

Partikel	m	k	Xij													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0	1	1	0	1	1	0	1	1	1	0	0	1	0
P2	2	3	1	0	1	0	1	0	0	1	1	0	1	1	1	1
P3	2.5	4	0	1	1	1	1	1	0	1	1	0	0	0	1	0

Tabel 3.7 Inisialisasi *Encode* Partikel untuk *Fold #2*

Partikel	m	k	Xij													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4	3	1	1	1	0	1	0	0	1	1	1	0	0	1	1
P2	1.9	4	0	0	1	0	0	0	0	1	1	0	0	0	1	0
P3	1.2	2	0	1	0	1	1	0	0	1	1	0	0	0	0	0

Tabel 3.8 Inisialisasi *Encode* Partikel untuk *Fold #3*

Partikel	m	k	Xij=Pbest													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	2.4	3	0	1	1	0	1	0	0	1	1	1	0	0	1	0
P2	2.6	2	1	0	1	0	0	0	0	1	1	1	0	0	1	0
P3	1.3	4	1	1	0	1	1	0	0	1	1	0	1	0	0	0

3.2.3.3 Train Klasifikasi FKNN

Train Klasifikasi FKNN pada proses ini dengan menggunakan *K-fold Cross Validation* yaitu membagi data sampel menjadi K kelompok data, dimana pada tiap K kelompok data akan digunakan menjadi *data testing* dan yang lainnya menjadi *data training*, kemudian hal tersebut akan berulang sampai K kali. Misalnya pada perhitungan manual ini digunakan data sampel sebanyak 12 data kemudian memilih $K = 3$ jadi tiap kelompok data berisi 4 data, sehingga *data testing* yang digunakan adalah 4 *data testing* dan 8 data yang lain akan menjadi *data training*, hal tersebut

akan berulang hingga K kali. Dimana hasil tiap *fold* di *train* FKNN dimana fitur yang digunakan adalah fitur yang bernilai 1 sesuai hasil *encode*.

A. *Fold* #1 Partikel 1

Pada *fold* #1 partikel 1 proses train FKNN digunakan nilai $m = 1.5$ dan $k = 2$ dan fitur yang digunakan adalah fitur F2, F3, F5, F6, F8, F9, F10, F13, sehingga data yang digunakan seperti terlihat pada Tabel 3.9 dan Tabel 3.10.

Tabel 3.9 Data Training *Fold* #1 Partikel 1 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut

No	KK	F2	F3	F5	F6	F8	F9	F10	F13	Kelas
1	21	0.1286	0.0321	0.0171	0.0071	0.0216	0.0107	0.0452	0.0043	A
2	109	0.0524	0.0036	0.0081	0.0071	0.0216	0.0107	0.0262	0.0043	M
3	14	0.0905	0.0321	0.0081	0.0071	0.0156	0.0107	0.0071	0.0043	M
4	55	0.0143	0.0036	0.0051	0.0071	0.0106	0.0107	0.0071	0.0043	B
5	30	0.0905	0.0321	0.0111	0.0071	0.0321	0.0107	0.0262	0.0043	A
6	32	0.0524	0.0036	0.0081	0.0071	0.0258	0.0964	0.0643	0.0043	M
7	80	0.0143	0.0036	0.0051	0.0071	0.0036	0.0107	0.0262	0.0043	B
8	115	0.0143	0.0036	0.0039	0.0071	0.0057	0.0107	0.0071	0.0214	B

Tabel 3.10 Data Testing *Fold* #1 Partikel 1 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut

No	KK	F2	F3	F5	F6	F8	F9	F10	F13	Kelas
1	60	0.1286	0.0036	0.0321	0.0071	0.0216	0.0964	0.0643	0.0043	A
2	90	0.0905	0.0321	0.0111	0.0071	0.0216	0.0107	0.0452	0.0386	A
3	81	0.0143	0.0036	0.0036	0.0071	0.0036	0.0107	0.0071	0.0043	B
4	24	0.0905	0.0321	0.0057	0.0071	0.0131	0.0107	0.0071	0.0300	M

Menghitung jarak kedekatan antara *data testing* terhadap *data training* dengan menggunakan metode *euclidean distance*. Misalnya kita hitung jarak antara *data testing* 1 terhadap *data training* 1 seperti dibawah ini.

$$\|x - x_j\| = \left(\sum_{l=1}^n |N_l - N_l^j|^2 \right)^{\frac{1}{2}}$$

Diketahui *norm* x dikurangi x_j ($\|x - x_j\|$) adalah nilai jarak *euclidean* antara data *testing* x dan tetangga terdekat atau data training ke- j (x_j), N_l adalah nilai bobot data *testing* pada atribut ke- l dan N_l^j nilai bobot data training ke- j pada atribut ke- l . Sedangkan n adalah jumlah keseluruhan atribut dan l adalah index ke- l .

$$\|x - x_j\| = \sqrt{\sum_{l=1}^n |N_l - N_l^j|^2}$$

$$\|x - x_j\| = \sqrt{((|0.1286 - 0.1286|^2) + (|0.0036 - 0.0321|^2) + (|0.0321 - 0.0171|^2) + (|0.0071 - 0.0071|^2) + (|0.0216 - 0.0216|^2) + (|0.0964 - 0.0107|^2) + (|0.0643 - 0.0452|^2) + (|0.0043 - 0.0043|^2))}$$

$$\|x - x_j\| = \sqrt{(0^2) + (0.0285^2) + (0.015^2) + (0^2) + (0^2) + (0.0857^2) + (0.0191^2) + (0^2)}$$

$$\|x - x_j\| = \sqrt{0 + 0.00081 + 0.00023 + 0 + 0 + 0.0074 + 0.000365 + 0}$$

$$\|x - x_j\| = \sqrt{0.0088} = 0.0936$$

Tabel 3.11 Nilai Hasil dari Perhitungan Jarak *Fold* #1 Partikel 1 dengan *Euclidean Distance* Sebelum Diurutkan

No	Data Training	Data Testing ke-1 kelas A	Data Testing ke-2 kelas A	Data Testing ke-3 kelas B	Data Testing ke-4 kelas M
1	Ke- 1 kelas A	0.0936	0.0516	0.1258	0.0614
2	Ke-2 kelas M	0.1232	0.0618	0.0465	0.0580
3	Ke-3 kelas M	0.1162	0.0517	0.0824	0.0260
4	Ke-4 kelas B	0.1566	0.0970	0.0071	0.0854
5	Ke-5 kelas A	0.1078	0.0406	0.0886	0.0376
6	Ke-6 kelas M	0.0800	0.1057	0.1122	0.1171
7	Ke-7 kelas B	0.1514	0.0923	0.0191	0.0880
8	Ke-8 kelas B	0.1582	0.0931	0.0173	0.0822

Tabel 3.11 merupakan hasil dari penghitungan jarak antara *data testing* dengan *data training* dengan menggunakan metode *euclidean distance*, pada kolom

pertama merupakan kolom *data training* ke-1 sampai ke-8 dengan kelas aktualnya kemudian kolom selanjutnya adalah kolom *data testing* ke-1 sampai ke-4 dengan kelas aktualnya nilainya seperti terlihat pada tabel tersebut. Kemudian diurutkan seperti pada Tabel 3.12.

Tabel 3.12 Nilai Hasil dari Perhitungan Jarak *Fold* #1 Partikel 1 dengan *Euclidean Distance* Setelah Diurutkan

No	Data Training	Data Testing ke-1 kelas A	No	Data Training	Data Testing ke-2 kelas A
1	Ke-6 kelas M	0.0800	1	Ke-5 kelas A	0.0406
2	Ke-1 kelas A	0.0936	2	Ke-1 kelas A	0.0516
3	Ke-5 kelas A	0.1078	3	Ke-3 kelas M	0.0517
4	Ke-3 kelas M	0.1162	4	Ke-2 kelas M	0.0618
5	Ke-2 kelas M	0.1232	5	Ke-7 kelas B	0.0923
6	Ke-7 kelas B	0.1514	6	Ke-8 kelas B	0.0931
7	Ke-4 kelas B	0.1566	7	Ke-4 kelas B	0.0970
8	Ke-8 kelas B	0.1582	8	Ke-6 kelas M	0.1057

No	Data Training	Data Testing ke-3 kelas B	No	Data Training	Data Testing ke-4 kelas M
1	Ke-4 kelas B	0.0071	1	Ke-3 kelas M	0.0260
2	Ke-8 kelas B	0.0173	2	Ke-5 kelas A	0.0376
3	Ke-7 kelas B	0.0191	3	Ke-2 kelas M	0.0580
4	Ke-2 kelas M	0.0465	4	Ke-1 kelas A	0.0614
5	Ke-3 kelas M	0.0824	5	Ke-8 kelas B	0.0822
6	Ke-5 kelas A	0.0886	6	Ke-4 kelas B	0.0854
7	Ke-6 kelas M	0.1122	7	Ke-7 kelas B	0.0880
8	Ke-1 kelas A	0.1258	8	Ke-6 kelas M	0.1171

Selanjutnya menghitung nilai derajat keanggotaan tiap-tiap kelas terhadap kelas yang lain misalnya kelas i pada tetangga ke- j , misalnya diketahui sebagai berikut dibawah ini.

Data training	$n1$	$n2$	$n3$	m	k
8	2	3	3	1.5	2

$n1$ = jumlah *data training* pada kelas 1 (kelas A atau kelas Atas)

$n2$ = jumlah *data training* pada kelas 2 (kelas M atau kelas Menengah)

$n3$ = jumlah *data training* pada kelas 3 (kelas B atau kelas Bawah)

m = bobot pangkat yang nilainya $m \in (1, +\infty)$ (nilai m tidak boleh bernilai 1 atau + tak hingga), bobot ini nanti digunakan untuk mencari nilai keanggotaan kelas pada *data testing*.

k = jumlah k tetangga terdekat

$$u_{ij} = \begin{cases} 0,51 + \left(\frac{n_j}{K}\right) * 0,49, & \text{kelas } j = i \\ \left(\frac{n_j}{K}\right) * 0,49, & \text{kelas } j \neq i \end{cases}$$

Dimana:

u_{ij} adalah nilai keanggotaan tetangga ke- j pada kelas ke- i

n_j adalah jumlah data yang termasuk kelas j pada *data training*

i adalah kelas status sosial ekonomi

j adalah kelas tetangga

K adalah jumlah *data training*.

$u_{1(1)} = 0.51 + \left(\frac{2}{8}\right) * 0.49$ $= 0.51 + 0.25 * 0.49$ $= 0.51 + 0.1225$ $= 0.6325$	$u_{1(2)} = \left(\frac{3}{8}\right) * 0.49$ $= 0.38 * 0.49$ $= 0.1838$	$u_{1(3)} = \left(\frac{3}{8}\right) * 0.49$ $= 0.38 * 0.49$ $= 0.1838$
$u_{2(1)} = \left(\frac{2}{8}\right) * 0.49$ $= 0.25 * 0.49$ $= 0.1225$	$u_{2(2)} = 0.51 + \left(\frac{3}{8}\right) * 0.49$ $= 0.51 + 0.38 * 0.49$ $= 0.51 + 0.1838$ $= 0.6938$	$u_{2(3)} = \left(\frac{3}{8}\right) * 0.49$ $= 0.38 * 0.49$ $= 0.1838$
$u_{3(1)} = \left(\frac{2}{8}\right) * 0.49$	$u_{3(2)} = \left(\frac{3}{8}\right) * 0.49$	$u_{3(3)} = 0.51 + \left(\frac{3}{8}\right) * 0.49$ $= 0.51 + 0.38 * 0.49$

$= 0.25 * 0.49$	$= 0.5 * 0.38$	$= 0.51 + 0.1838$
$= 0.1225$	$= 0.1838$	$= 0.6938$

Tabel 3.13 Nilai Keanggotaan *Fold* #1 Kelas ke-*i* Terhadap Tetangga ke-*j*

No	$u_{i,j}$	Nilai $u_{i,j}$
1	$u_{1,1}$	0.6325
2	$u_{2,1}$	0.1225
3	$u_{3,1}$	0.1225
4	$u_{1,2}$	0.1838
5	$u_{2,2}$	0.6938
6	$u_{3,2}$	0.1838
7	$u_{1,3}$	0.1838
8	$u_{2,3}$	0.1838
9	$u_{3,3}$	0.6938

Tabel 3.13 adalah tabel derajat keanggotaan kelas ke-*i* pada tetangga ke-*j*, dimana kolom ke-2 menyatakan identitas dan kolom ke-3 adalah nilainya. Selanjutnya menghitung nilai keanggotaan kelas pada *data testing* dengan *euclidean distance*. Misalkan menghitung nilai keanggotaan kelas A untuk *data testing* 1

$$u_i(x) = \frac{\sum_{j=1}^k u_{ij} \left(\frac{1}{\|x - x_j\|^{\frac{2}{(m-1)}}} \right)}{\sum_{j=1}^k \left(\frac{1}{\|x - x_j\|^{\frac{2}{(m-1)}}} \right)}$$

$$u_i(x) = \frac{\left(0.1225 \left(\frac{1}{0.0800^{\frac{2}{1.5-1}}} \right) \right) + \left(0.6325 \left(\frac{1}{0.0936^{\frac{2}{1.5-1}}} \right) \right)}{\left(\frac{1}{0.0800^{\frac{2}{1.5-1}}} \right) + \left(\frac{1}{0.0936^{\frac{2}{1.5-1}}} \right)}$$

$$u_i(x) = \frac{\left(0.1225\left(\frac{1}{0.000041}\right)\right) + \left(0.6325\left(\frac{1}{0.000077}\right)\right)}{\left(\left(\frac{1}{0.000041}\right) + \left(\frac{1}{0.000077}\right)\right)}$$

$$u_i(x) = \frac{((0.1225(24390.24)) + (0.6325(12987.01)))}{24390.24 + 12987.01}$$

$$u_i(x) = \frac{2987.80 + 8214.28}{37377.25} = \frac{11202.08}{37377.25} = 0.2997 = 0.30$$

Tabel 3.14 Hasil FKNN *Fold* #1 Partikel 1

Data Testing	kelas A	kelas M	kelas B	jumlah
Ke-1 kelas A	0.30	0.52	0.18	1.00
Ke-2 kelas A	0.63	0.18	0.18	1.00
Ke-3 kelas B	0.12	0.18	0.69	1.00
Ke-4 kelas M	0.22	0.60	0.18	1.00

Tabel 3.14 merupakan hasil dari proses *train* FKNN dimana hasil dari *data testing* 1 sampai 4 terhadap semua kelas dengan melihat nilai derajat keanggotaannya yang paling besar. Dari hasil pada Tabel 3.14 kita dapat membuat tabel *confusion matrix* seperti terlihat pada Tabel 3.16 untuk menghitung nilai *sensitifity* dan *specificity* yaitu mendapatkan nilai TPR (*True Positive Rate*) dan FPR (*False Positive Rate*) sebelum membuat kurva ROC, setelah membuat kurva ROC selanjutnya mencari nilai *Area Under ROC Curve* (AUC) yang didapatkan dari menghitung luas daerah dibawah kurva ROC.

Tabel 3.15 Confusion Matrix

Kelas Aktual	Kelas Prediksi		
	Kelas 1	Kelas 2	Kelas 3
Kelas 1	C _{1,1}	C _{1,2}	C _{1,3}
Kelas 2	C _{2,1}	C _{2,2}	C _{2,3}
Kelas 3	C _{3,1}	C _{3,2}	C _{3,3}

Dimana,

$C_{1,1}$ = <i>True Positive</i> kelas 1	$C_{1,2}$ = kelas 1 masuk kelas 2	$C_{1,3}$ = kelas 1 masuk kelas 3
$C_{2,1}$ = kelas 2 masuk kelas 1	$C_{2,2}$ = <i>True Positive</i> kelas 2	$C_{2,3}$ = kelas 2 masuk kelas 3
$C_{3,1}$ = kelas 3 masuk kelas 1	$C_{3,2}$ = kelas 3 masuk kelas 2	$C_{3,3}$ = <i>True Positive</i> kelas 3

Tabel 3.16 Hasil *Confusion Matrix* Partikel 1

Kelas Aktual	Kelas Prediksi		
	Kelas 1	Kelas 2	Kelas 3
Kelas 1	1	1	0
Kelas 2	0	1	0
Kelas 3	0	0	1

- $TP = C_{1,1} + C_{2,2} + C_{3,3} = 1 + 1 + 1 = 3$
- $TN = (C_{1,1} + C_{2,2}) + (C_{1,1} + C_{3,3}) + (C_{2,2} + C_{3,3}) = 1 + 1 + 1 + 1 + 1 + 1 = 6$
- $FP = (C_{2,1} + C_{3,1}) + (C_{1,2} + C_{3,2}) + (C_{1,3} + C_{2,3}) = 0 + 0 + 1 + 0 + 0 + 0 = 1$
- $FN = (C_{1,2} + C_{1,3}) + (C_{2,1} + C_{2,3}) + (C_{3,1} + C_{3,2}) = 1 + 0 + 0 + 0 + 0 + 0 = 1$

Setelah diketahui nilai TP, TN, FN, dan FP maka selanjutnya menghitung nilai *Sensitifity* atau TPR, menghitung nilai *specificity* atau TNR, dan menghitung nilai FPR.

$$t_{p\text{rate}} = \frac{TP}{P} = \frac{TP}{TP + FN} = \frac{3}{3 + 1} = \frac{3}{4} = 0.75$$

$$t_{n\text{rate}} = \frac{TN}{N} = \frac{TN}{TN + FP} = \frac{6}{6 + 1} = \frac{6}{7} = 0.8571$$

Sehingga nilai FPR = $1 - \text{TNR} = 1 - 0.8571 = 0.1429$

B. Fold #1 Partikel 2

Pada *fold* #1 partikel 2 ini masih sama untuk penggunaan datanya hanya saja untuk m , k dan fitur yang digunakan berbeda dari sebelumnya, dimana $m = 2$, $k = 3$, serta fitur yang digunakan adalah F1, F3, F5, F8, F9, F11, F12, F13, F14, sehingga data yang digunakan seperti terlihat pada Tabel 3.17 dan Tabel 3.18.

Tabel 3.17 Data Training Fold #1 Partikel 2 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut

No	KK	F1	F3	F5	F8	F9	F11	F12	F13	F14	Kelas
1	21	0.0179	0.0321	0.0171	0.0216	0.0107	0.0117	0.0071	0.0043	0.0029	A
2	109	0.0179	0.0036	0.0081	0.0216	0.0107	0.0071	0.0071	0.0043	0.0029	M
3	14	0.0036	0.0321	0.0081	0.0156	0.0107	0.0391	0.0643	0.0043	0.0029	M
4	55	0.0036	0.0036	0.0051	0.0106	0.0107	0.0071	0.0071	0.0043	0.0257	B
5	30	0.0179	0.0321	0.0111	0.0321	0.0107	0.0643	0.0071	0.0043	0.0029	A
6	32	0.0036	0.0036	0.0081	0.0258	0.0964	0.0231	0.0071	0.0043	0.0029	M
7	80	0.0036	0.0036	0.0051	0.0036	0.0107	0.0071	0.0071	0.0043	0.0257	B
8	115	0.0036	0.0036	0.0039	0.0057	0.0107	0.0071	0.0071	0.0214	0.0029	B

Tabel 3.18 Data Testing Fold #1 Partikel 2 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut

No	KK	F1	F3	F5	F8	F9	F11	F12	F13	F14	Kelas
1	60	0.0321	0.0036	0.0321	0.0216	0.0964	0.0071	0.0071	0.0043	0.0029	A
2	90	0.0179	0.0321	0.0111	0.0216	0.0107	0.0243	0.0071	0.0386	0.0029	A
3	81	0.0036	0.0036	0.0036	0.0036	0.0107	0.0071	0.0071	0.0043	0.0189	B
4	24	0.0179	0.0321	0.0057	0.0131	0.0107	0.0186	0.0071	0.0300	0.0029	M

Proses KNN yang perhitungannya sama dengan perhitungan pada *fold* #1 partikel 1 yaitu mencari jarak *euclidean distance* antara *data testing* dengan *data training* dimana hasilnya setelah diurutkan seperti pada Tabel 3.19.

Tabel 3.19 Nilai Hasil dari Perhitungan Jarak Fold #1 Partikel 2 dengan *Euclidean Distance* Setelah Diurutkan

No	Data Training	Data Testing ke-1 kelas A	No	Data Training	Data Testing ke-2 kelas A
1	Ke-6 kelas M	0.0409	1	Ke-1 kelas A	0.0370
2	Ke-2 kelas M	0.0902	2	Ke-8 kelas B	0.0437
3	Ke-1 kelas A	0.0928	3	Ke-2 kelas M	0.0479
4	Ke-8 kelas B	0.0975	4	Ke-5 kelas A	0.0537

5	Ke-4 kelas B	0.0977	5	Ke-4 kelas B	0.0563
6	Ke-7 kelas B	0.0987	6	Ke-7 kelas B	0.0581
7	Ke-5 kelas A	0.1104	7	Ke-3 kelas M	0.0701
8	Ke-3 kelas M	0.1178	8	Ke-6 kelas M	0.0978

No	Data Training	Data Testing ke-3 kelas B	No	Data Training	Data Testing ke-4 kelas M
1	Ke-7 kelas B	0.0070	1	Ke-1 kelas A	0.0302
2	Ke-4 kelas B	0.0099	2	Ke-8 kelas B	0.0358
3	Ke-8 kelas B	0.0235	3	Ke-2 kelas M	0.0411
4	Ke-2 kelas M	0.0284	4	Ke-4 kelas B	0.0484
5	Ke-1 kelas A	0.0425	5	Ke-7 kelas B	0.0493
6	Ke-5 kelas A	0.0736	6	Ke-5 kelas A	0.0561
7	Ke-3 kelas M	0.0743	7	Ke-3 kelas M	0.0676
8	Ke-6 kelas M	0.0915	8	Ke-6 kelas M	0.0960

Selanjutnya adalah proses FKNN yaitu mencari nilai derajat keanggotaan kelas ke- i terhadap tetangga ke- j , kemudian mencari nilai FKNN seperti pada perhitungan pada *fold* #1 partikel 1 sebelumnya. Dengan diketahui jumlah *data training* jumlah data kelas ke-1, ke-2, dan ke-3 yang akan digunakan pada *fold* #1 partikel 2 ini sama dengan pada *fold* #1 partikel 1, sehingga hasilnya pun sama untuk nilai derajat keanggotaan kelas ke- i terhadap tetangga ke- j , namun untuk nilai m dan k yang digunakan berbeda sesuai hasil *encode* awal dimana m yang digunakan pada *fold* #1 partikel 2 ini adalah 2 dan k adalah 3. Adapun hasil FKNN *fold* #1 partikel 2 seperti pada Tabel 3.20.

Tabel 3.20 Hasil FKNN *Fold* #1 Partikel 2

	kelas A	kelas M	kelas B	jumlah
data uji1/A	0.193	0.623	0.184	1.00
data uji2/A	0.343	0.315	0.342	1.00
data uji3/B	0.123	0.184	0.694	1.00
data uji 4/M	0.349	0.306	0.345	1.00

Selanjutnya adalah membuat tabel *confusion matrix* dari hasil klasifikasi diatas pada Tabel 3.21 untuk mencari TP, TN, FP, dan FN kemudian menghitung nilai *sensitifity* atau nilai TPR, mencari nilai *specificity* atau TNR, dan FPR seperti perhitungan pada *fold* 1 partikel 1.

Tabel 3.21 Hasil *Confusion Matrix* Partikel 2

Kelas Aktual	Kelas Prediksi		
	Kelas 1	Kelas 2	Kelas 3
Kelas 1	1	1	0
Kelas 2	1	0	0
Kelas 3	0	0	1

Sehingga dihasilkan nilai *sensitifity* atau TPRnya adalah 0.50, sedangkan nilai *specificity* 0.67, dan nilai FPRnya 0.33.

C. *Fold* #1 Partikel 3

Pada *fold* #1 partikel 3 ini masih sama untuk penggunaan datanya hanya saja untuk m , k dan fitur yang digunakan berbeda dari sebelumnya, dimana $m = 2.5$, $k = 4$, serta fitur yang digunakan adalah F2, F3, F4, F5, F6, F8, F9, F13, sehingga data yang digunakan seperti terlihat pada Tabel 3.22 dan Tabel 3.23.

Tabel 3.22 *Data Training Fold #1* Partikel 3 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut

No	KK	F2	F3	F4	F5	F6	F8	F9	F13	Kelas
1	21	0.1286	0.0321	0.0819	0.0171	0.0071	0.0216	0.0107	0.0043	A
2	109	0.0524	0.0036	0.0353	0.0081	0.0071	0.0216	0.0107	0.0043	M
3	14	0.0905	0.0321	0.0294	0.0081	0.0071	0.0156	0.0107	0.0043	M
4	55	0.0143	0.0036	0.0178	0.0051	0.0071	0.0106	0.0107	0.0043	B
5	30	0.0905	0.0321	0.1286	0.0111	0.0071	0.0321	0.0107	0.0043	A
6	32	0.0524	0.0036	0.0353	0.0081	0.0071	0.0258	0.0964	0.0043	M
7	80	0.0143	0.0036	0.0178	0.0051	0.0071	0.0036	0.0107	0.0043	B
8	115	0.0143	0.0036	0.0155	0.0039	0.0071	0.0057	0.0107	0.0214	B

Tabel 3.23 *Data Testing Fold #1 Partikel 3 Setelah Dinormalisasi dan Dikalikan dengan Bobot Atribut*

No	KK	F2	F3	F4	F5	F6	F8	F9	F13	Kelas
1	60	0.1286	0.0036	0.1133	0.0321	0.0071	0.0216	0.0964	0.0043	A
2	90	0.0905	0.0321	0.0528	0.0111	0.0071	0.0216	0.0107	0.0386	A
3	81	0.0143	0.0036	0.0143	0.0036	0.0071	0.0036	0.0107	0.0043	B
4	24	0.0905	0.0321	0.0353	0.0057	0.0071	0.0131	0.0107	0.0300	M

Proses KNN yang perhitungannya sama dengan perhitungan pada *fold* #1 partikel 1 yaitu mencari jarak *euclidean distance* antara *data testing* dengan *data training* dimana hasilnya setelah diurutkan seperti pada Tabel 3.24.

Tabel 3.24 Nilai Hasil dari Perhitungan Jarak *Fold* #1 Partikel 3 dengan *Euclidean Distance* setelah Diurutkan

No	Data Training	Data Testing ke-1 kelas A	No	Data Training	Data Testing ke-2 kelas A
1	Ke-1 kelas A	0.0968	1	Ke-3 kelas M	0.0420
2	Ke-5 kelas A	0.1020	2	Ke-1 kelas A	0.0593
3	Ke-6 kelas M	0.1117	3	Ke-2 kelas M	0.0613
4	Ke-3 kelas M	0.1314	4	Ke-5 kelas A	0.0839
5	Ke-2 kelas M	0.1408	5	Ke-8 kelas B	0.0928
6	Ke-4 kelas B	0.1743	6	Ke-4 kelas B	0.0958
7	Ke-7 kelas B	0.1749	7	Ke-7 kelas B	0.0969
8	Ke-8 kelas B	0.1770	8	Ke-6 kelas M	0.1055

No	Data Training	Data Testing ke-3 kelas B	No	Data Training	Data Testing ke-4 kelas M
1	Ke-7 kelas B	0.0038	1	Ke-3 kelas M	0.0266
2	Ke-4 kelas B	0.0080	2	Ke-2 kelas M	0.0548
3	Ke-8 kelas B	0.0173	3	Ke-1 kelas A	0.0670
4	Ke-2 kelas M	0.0473	4	Ke-8 kelas B	0.0845
5	Ke-3 kelas M	0.0838	5	Ke-4 kelas B	0.0872
6	Ke-6 kelas M	0.0988	6	Ke-7 kelas B	0.0876

7	Ke-1 kelas A	0.1377	7	Ke-5 kelas A	0.0988
8	Ke-5 kelas A	0.1434	8	Ke-6 kelas M	0.1022

Selanjutnya adalah proses FKNN yaitu mencari nilai derajat keanggotaan kelas ke- i terhadap tetangga ke- j , kemudian mencari nilai FKNN seperti pada perhitungan pada *fold* #1 partikel 1 sebelumnya. Dengan diketahui jumlah *data training* jumlah data kelas ke-1, ke-2, dan ke-3 yang akan digunakan pada *fold* #1 partikel 3 ini sama dengan pada *fold* #1 partikel 1, sehingga hasilnya pun sama untuk nilai derajat keanggotaan kelas ke- i terhadap tetangga ke- j , namun untuk nilai m dan k yang digunakan berbeda sesuai hasil *encode* awal dimana m yang digunakan pada *fold* #1 partikel 3 ini adalah 2 dan k adalah 3. Adapun hasil FKNN *fold* #1 partikel 3 seperti pada Tabel 3.25.

Tabel 3.25 Hasil FKNN *Fold* #1 partikel 3

	kelas A	kelas M	kelas B	jumlah
data uji1/A	0.410	0.406	0.184	1.00
data uji2/A	0.322	0.494	0.184	1.00
data uji3/B	0.123	0.195	0.682	1.00
data uji 4/M	0.201	0.557	0.242	1.00

Selanjutnya adalah membuat tabel *confusion matrix* dari hasil klasifikasi diatas pada Tabel 3.26 untuk mencari TP, TN, FP, dan FN kemudian menghitung nilai *sensitifity* atau nilai TPR, mencari nilai *specificity* atau TNR, dan FPR seperti perhitungan pada *fold* #1 partikel 1.

Tabel 3.26 Hasil *Confusion Matrix* Partikel 3

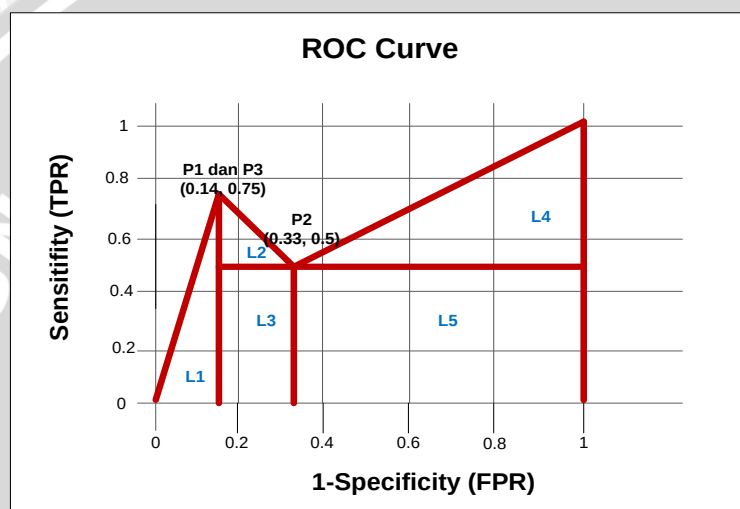
Kelas Aktual	Kelas Prediksi		
	Kelas 1	Kelas 2	Kelas 3
Kelas 1	1	1	0
Kelas 2	0	1	0
Kelas 3	0	0	1

Sehingga dihasilkan nilai *sensitifity* atau TPRnya adalah 0.75, sedangkan nilai *specificity* 0.8571, dan nilai FPRnya 0.1429.

3.2.3.4 Metode Adaptive Fuzzy K-Nearest Neighbor Berdasarkan Pada Particle Swarm Optimization

Setelah didapatkan nilai TPR dan FPR dari masing masing hasil *train* FKNN dengan *K-Fold Cross Validation fold #1*, proses selanjutnya yaitu membuat kurva ROC dan menghitung nilai *Area Under ROC Curve* (AUC) dan menghitung nilai *fitness* dari partikel yang di *encode* untuk *fold #1*.

A. Menghitung Nilai AUC dan Nilai Fitness Tiap Partikel



Gambar 3.15 Kurva ROC

Kemudian menghitung nilai dibawah kurva ROC atau AUC. Adapun nilai AUC dari ketiga partikel yang sudah di-*train* pada *fold #1* adalah sebagai berikut.

$$AUC = L1 + L2 + L3 + L4 + L5$$

$$AUC = (0.5 * 0.14 * 0.75) + (0.5 * 0.19 * 0.25) + (0.19 * 0.5) + (0.5 * 0.67 * 0.5) + (0.67 * 0.5)$$

$$AUC = 0.0525 + 0.02375 + 0.095 + 0.1675 + 0.335 = 0.67375 = 0.67$$

Setelah mendapatkan nilai AUC selanjutnya adalah menghitung nilai *fitness* tiap partikel dimana diketahui α dan β (0.85 dan 0.15), dimana α adalah ketetapan nilai akurasi klasifikasi FKNN, β adalah ketetapan nilai yang terindikasi untuk seleksi fitur, $f_1 = AUC$, dan f_2 adalah fitur yang bernilai 1 dibagi dengan jumlah fitur, misalnya menghitung nilai *fitness* partikel 1 pada *fold #1* pada Tabel 3.6 sebagai berikut.

$$f_2 = 1 - \frac{\sum_{j=1}^n f_{ti}}{n} = 1 - \frac{8}{14} = 1 - 0.57 = 0.43$$

$$fP1 = \alpha \times f_1 + \beta \times f_2 = 0.85 \times 0.67 + 0.15 \times 0.43 = 0.5695 + 0.0645 \\ = 0.6363 = 0.64$$

Tabel 3.27 Hasil Nilai *Fitness* Partikel pada *Fold* #1

Partikel	m	k	X _{ij}														f ₂	f ₁	f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14			
P1	1.5	2	0	1	1	0	1	1	0	1	1	1	0	0	1	0	0.43	0.67	0.64
P2	2	3	1	0	1	0	1	0	0	1	1	0	1	1	1	1	0.36	0.67	0.63
P3	2.5	4	0	1	1	1	1	1	0	1	1	0	0	0	1	0	0.43	0.67	0.64

Data yang diblok warna merah adalah *global best fitness* atau *fitness* terbaik dari keseluruhan populasi.

B. Inisialisasi Parameter

Proses inisialisasi adalah proses menentukan nilai semua parameter yang akan digunakan t_{max} adalah maksimum iterasi, r_1 , r_2 adalah nilai *random*, w_{min} adalah minimum bobot *inertia*, w_{max} adalah maksimum bobot *inertia*, c_{1f} , c_{1i} , c_{2f} dan c_{2i} adalah konstanta. Dimana diketahui nilai parameter sebagai berikut sesuai pada jurnal (Chen at all, 2011):

$$\begin{array}{llllll} t_{max} = 2 & r_1 = 0.5 & r_2 = 0.5 & \text{fitur} = 14 & w_{min} = 0.4 \\ w_{max} = 0.9 & c_{1f} = 0.5 & c_{1i} = 2.5 & c_{2f} = 2.5 & c_{2i} = 0.5 \end{array}$$

untuk nilai w pada tiap iterasi adalah sebagai berikut:

a. Iterasi 1

$$w = w_{min} + (w_{max} - w_{min}) \frac{(t_{max} - t)}{t_{max}}$$

$$w_1 = 0.4 + (0.9 - 0.4) \frac{(2 - 1)}{2} = 0.4 + \left(0.5 \times \frac{1}{2}\right) = 0.4 + 0.25 = 0.65$$

b. Iterasi 2

$$w = w_{min} + (w_{max} - w_{min}) \frac{(t_{max} - t)}{t_{max}}$$

$$w_2 = 0.4 + (0.9 - 0.4) \frac{(2 - 2)}{2} = 0.4 + \left(0.5 \times \frac{0}{2}\right) = 0.4 + 0 = 0.4$$

untuk nilai c_1 dan c_2 untuk tiap iterasi adalah sebagai berikut:

a. Iterasi 1

$$c_1 = (c_{1f} - c_{1i}) \frac{t}{t_{max}} + c_{1i}$$

$$= (0.5 - 2.5) * \frac{1}{2} + 2.5 = -2 * \frac{1}{2} + 2.5 = 1.5$$

$$c_2 = (c_{2f} - c_{2i}) \frac{t}{t_{max}} + c_{2i}$$

$$= (2.5 - 0.5) \frac{1}{2} + 0.5 = 2 * \frac{1}{2} + 0.5 = 1.5$$

b. Iterasi 2

$$c_1 = (c_{1f} - c_{1i}) \frac{t}{t_{max}} + c_{1i}$$

$$= (0.5 - 2.5) * \frac{2}{2} + 2.5 = -2 * \frac{2}{2} + 2.5 = 0.5$$

$$c_2 = (c_{2f} - c_{2i}) \frac{t}{t_{max}} + c_{2i}$$

$$= (2.5 - 0.5) \frac{2}{2} + 0.5 = 2 * \frac{2}{2} + 0.5 = 2.5$$

C. Proses TVPSO Fold #1 Iterasi 1

Selanjutnya adalah mengupdate nilai *velocity* dan posisi tiap partikel pada iterasi yang telah dibangkitkan karena masih dalam iterasi awal maka untuk *velocity* awal belum ada pergerakan maka nilainya adalah 0 dan nilai posisi terbaik tiap partikel sama dengan posisi awal. Global optimal partikel yaitu partikel yang memiliki nilai *fitness* tertinggi disini bisa dilihat bahwa *global best* partikel atau *gbest* adalah partikel 1 (P1) seperti pada Tabel 3.28. Dimana *pfit* dan *pbest* sementara disimpan didalam *memory* untuk nantinya diupdate kembali pada tiap iterasi dengan membandingkan nilai *fitness* terbaru, selain itu nilai *velocity* sebelumnya juga disimpan untuk perhitungan *update velocity* selanjutnya.

Tabel 3.28 Posisi Awal, *Pbest* dan *Gbest* Tiap Partikel pada Fold #1

Partikel	m	k	Pbest = X _{ij}														f2	f1	f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14			
P1	1.5	2	0	1	1	0	1	1	0	1	1	1	0	0	1	0	0.43	0.67	0.64
P2	2	3	1	0	1	0	1	0	0	1	1	0	1	1	1	1	0.36	0.67	0.63
P3	2.5	4	0	1	1	1	1	1	0	1	1	0	0	0	1	0	0.43	0.67	0.64

Pada iterasi pertama kita mulai hitung nilai $V_{ij}(1)$ yaitu update nilai *velocity* untuk tiap partikel dimana diketahui nilai V_{ij} awal adalah 0 dan nilai X_{ij} sesuai hasil *encode* dan nilai $pbest$ sama dengan X_{ij} , untuk iterasi awal pasti nilai *velocity*nya masih 0 karena partikel belum ada perlakuan. Berikut perhitungannya, misalnya kita *update velocity* untuk P1 dengan X_{ij} FI atau untuk partikel ke-1 pada dimensi 1 seperti dibawah ini.

$$v_{i,j}^{n+1} = w \times v_{i,j}^n + c_1 \times r_1 (P_{i,j}^n - x_{i,j}^n) + c_2 \times r_2 (P_{g,j}^n - x_{i,j}^n)$$

$$v_{i,j}^{n+1} = 0.65 \times 0 + 1.5 \times 0.5 (0 - 0) + 1.5 \times 0.5 (0 - 0)$$

$$v_{i,j}^{n+1} = 0 + 0 + 0 = 0.00$$

Tabel 3.29 Nilai *Velocity* Iterasi 1

Partikel	m	k	$V_{ij}(1)$													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P2	2	3	-0.75	0.75	0.00	0.00	0.00	0.75	0.00	0.00	0.00	0.75	-0.75	-0.75	0.00	-0.75
P3	2.5	4	0.00	0.00	0.00	-0.75	0.00	0.00	0.00	0.00	0.00	0.75	0.00	0.00	0.00	0.00

Tabel 3.29 merupakan tabel hasil *update* nilai *velocity* untuk tiap-tiap dimensi partikel, kemudian menghitung nilai *sigmoid* karena pada proses PSO disini menggunakan binary PSO untuk mentransformasi kecepatan dari ruang kontinu untuk ruang probabilitas. Berikut perhitungannya misalnya kita ambil pada *velocity* partikel ke 1 pada dimensi 1 diatas untuk dicari nilai *sigmoid*nya.

$$\text{sig}(v_{i,j}) = \frac{1}{1 + \exp(-v_{i,j})}, j = 1, 2, \dots, d$$

$$\text{sig}(v_{i,j}) = \frac{1}{1 + \exp(-0)} = \frac{1}{1 + 1} = \frac{1}{2} = 0.5$$

Tabel 3.30 Nilai *Sigmoid* Iterasi 1

Partikel	m	k	sig 1													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
P2	2	3	0.32	0.68	0.50	0.50	0.50	0.68	0.50	0.50	0.50	0.68	0.32	0.32	0.50	0.32
P3	2.5	4	0.50	0.50	0.50	0.32	0.50	0.50	0.50	0.50	0.50	0.68	0.50	0.50	0.50	0.50

Tabel 3.30 merupakan tabel hasil menghitung nilai *sigmoid* pada iterasi 1 untuk tiap-tiap dimensi partikel. Kemudian membangkitkan nilai random antara 0 sampai 1 sebagai pembanding menentukan posisi terbaru tiap-tiap dimensi partikel. Nilai Random yang dibangkitkan terlihat pada Tabel 3.31.

Tabel 3.31 Nilai Random Iterasi 1

Partikel	m	k	rnd 1													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0.24	0.94	0.94	0.10	0.71	0.66	0.90	0.66	0.47	0.08	0.33	0.57	0.08	0.39
P2	2	3	0.83	0.84	0.43	0.37	0.71	0.25	0.57	0.30	0.02	0.71	0.58	0.08	0.73	0.92
P3	2.5	4	0.93	0.10	0.61	0.92	0.37	0.70	0.47	0.59	0.46	0.52	0.34	0.42	0.78	0.30

Selanjutnya mengupdate posisi pada iterasi 1 atau $X_{ij}(1)$ dengan persamaan update posisi pada *binary* PSO seperti dibawah ini, misalnya kita bandingkan antara nilai *sigmoid* pada partikel 1 dimensi 1 dengan nilai random pada partikel 1 dimensi 1 untuk menentukan posisi partikel bernilai 1 atau 0. Hasilnya seperti pada Tabel 3.32.

$$x_{i,j}^{n+1} = \begin{cases} 1, & \text{if } rnd < sig(v_{i,j}) \\ 0, & \text{if } rnd \geq sig(v_{i,j}) \end{cases}, j = 1, 2, \dots, d$$

$$x_{i,j}^{n+1} = 0.24 < 0.50 = 1$$

Tabel 3.32 Hasil Update Posisi Partikel pada Iterasi 1 atau $X_{ij}(1)$

Partikel	m	k	$X_{ij}(1)$													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	1	0	0	1	0	0	0	0	1	1	1	0	1	1
P2	2	3	0	0	1	1	0	1	0	1	1	0	0	1	0	0
P3	2.5	4	0	1	0	0	1	0	1	0	1	1	1	1	0	1

Selanjutnya adalah *train* FKNN kembali dan menghitung nilai AUC untuk mendapatkan nilai *fitness* dari posisi terbaru dari iterasi 1. Dimana untuk *train* FKNN digunakan fitur yang terpilih pada iterasi 1 masih pada *fold* #1. Sehingga dihasilkan nilai AUCnya adalah 0.43 dan nilai *fitness* untuk tiap tiap partikel adalah sebagai berikut terlihat pada Tabel 3.33.

Tabel 3.33 Hasil Nilai *Fitness* Partikel pada *Fold* #1 Iterasi 1

Partikel	m	k	$X_{ij}(1)$														f2	f1	f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14			
P1	1.5	2	1	0	0	1	0	0	0	0	1	1	1	0	1	1	0.50	0.43	0.44
P2	2	3	0	0	1	1	0	1	0	1	1	0	0	1	0	0	0.57	0.43	0.45
P3	2.5	4	0	1	0	0	1	0	1	0	1	1	1	1	0	1	0.43	0.43	0.43

Kemudian mengupdate kembali *pfit* dan *pbest* dengan membandingkan nilai *fitness* tiap partikel yang disimpan didalam *memory* dengan nilai *fitness* terbaru untuk memilih *pbest*. Dari membandingkan nilai *pfit* di *memory* dengan *pfit* terbaru, yaitu memilih nilai *fitness* yang tertinggi. Sehingga di dapatkan *pbest* sebagai berikut yang kemudian disimpan kembali untuk diperbaharui kembali hingga iterasi terakhir. Sedangkan *gbest* adalah partikel yang memiliki *gfit* atau global *fitness* terbaik pada populasi disini terlihat pada data *fitness* yang diblok warna merah seperti terlihat pada Tabel 3.34.

Tabel 3.34 *Pbest* dan *Gbest* Partikel pada *Fold* #1 Iterasi 1

Partikel	m	k	<i>Pbest</i> (1)														f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	
P1	1.5	2	0	1	1	0	1	1	0	1	1	1	0	0	1	0	0.64
P2	2	3	1	0	1	0	1	0	0	1	1	0	1	1	1	1	0.63
P3	2.5	4	0	1	1	1	1	1	0	1	1	0	0	0	1	0	0.64

D. Proses TVPSO *Fold* #1 Iterasi 2

Pada iterasi 2 ini kita gunakan *pbest*, *gbest* dan *velocity* yang tersimpan terakhir didalam *memory* yang telah di perbaharui pada iterasi sebelumnya untuk memperbaharui lagi *velocity* dan posisi tiap partikel. Adapun prosesnya sama dengan proses pada iterasi sebelumnya. Sehingga didapatkan *velocity* baru, *sigmoid* baru dan untuk nilai random juga di perbaharui kembali pada tiap iterasi. Berikut hasilnya terlihat pada Tabel 3.35 sampai 3.38.

Tabel 3.35 Nilai *Velocity* Iterasi 2

Partikel	m	k	$V_{ij}(2)$													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	-1.50	1.50	1.50	-1.50	1.50	1.50	0.00	1.50	0.00	0.00	-1.50	0.00	0.00	-1.50
P2	2	3	-0.05	1.55	0.00	-1.50	1.50	0.05	0.00	0.00	0.00	1.55	-0.05	-1.55	1.50	-0.05
P3	2.5	4	0.00	0.00	1.50	-0.05	0.00	1.50	-1.50	1.50	0.00	0.05	-1.50	-1.50	1.50	-1.50

Tabel 3.36 Nilai *Sigmoid* Iterasi 2

Partikel	m	k	sig 2													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0.18	0.82	0.82	0.18	0.82	0.82	0.50	0.82	0.50	0.50	0.18	0.50	0.50	0.18
P2	2	3	0.49	0.82	0.50	0.18	0.82	0.51	0.50	0.50	0.50	0.82	0.49	0.18	0.82	0.49
P3	2.5	4	0.50	0.50	0.82	0.49	0.50	0.82	0.18	0.82	0.50	0.51	0.18	0.18	0.82	0.18

Tabel 3.37 Nilai *Random* Iterasi 2

Partikel	m	k	rnd 2													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0.87	0.59	0.63	0.56	0.30	0.94	0.18	0.90	0.22	0.63	0.66	0.81	0.66	0.71
P2	2	3	0.05	0.69	0.22	0.24	0.27	0.19	0.01	0.54	0.57	0.13	0.43	0.54	0.15	0.76
P3	2.5	4	0.12	0.69	0.24	0.58	0.57	0.94	0.29	0.96	0.14	0.18	0.00	0.51	0.97	0.86

Tabel 3.38 Hasil *Update* Posisi Partikel pada Iterasi 2 atau $X_{ij}(2)$

Partikel	m	k	$X_{ij}(2)$													
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5	2	0	1	1	0	1	0	1	0	1	0	0	0	0	0
P2	2	3	1	1	1	0	1	1	1	0	0	1	1	0	1	0
P3	2.5	4	1	0	1	0	0	0	0	0	1	1	1	0	0	0

Selanjutnya adalah *train* FKNN kembali dan menghitung nilai AUC untuk mendapatkan nilai *fitness* dari posisi terbaru dari iterasi 2. Dimana untuk *train* FKNN digunakan fitur yang terpilih pada iterasi 2 masih pada *fold* #1. Sehingga dihasilkan nilai AUCnya adalah 0.68 dan nilai *fitness* untuk tiap tiap partikel adalah sebagai berikut seperti terlihat pada Tabel 3.39.

Tabel 3.39 Hasil Nilai *Fitness* Partikel pada *Fold* #1 Iterasi 2

Partikel	M	k	X _{ij} (2)														f2	f1	f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14			
P1	1.5	2	0	1	1	0	1	0	1	0	1	0	0	0	0	0	0.64	0.68	0.67
P2	2	3	1	1	1	0	1	1	1	0	0	1	1	0	1	0	0.36	0.68	0.63
P3	2.5	4	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0.64	0.68	0.67

Kemudian mengupdate kembali *pfit* dan *pbest* dengan membandingkan nilai *fitness* tiap partikel yang disimpan didalam *memory* dengan nilai *fitness* terbaru untuk memilih *pbest*. Dari membandingkan nilai *pfit* di *memory* dengan *pfit* terbaru, yaitu memilih nilai *fitness* yang tertinggi. Sehingga di dapatkan *pbest* sebagai berikut yang kemudian disimpan kembali untuk diperbaharui kembali hingga iterasi terakhir. Sedangkan *gbest* adalah partikel yang memiliki *gfit* atau global *fitness* terbaik pada populasi disini terlihat pada data *fitness* yang diblok warna merah seperti terlihat pada Tabel 3.40.

Tabel 3.40 *Pbest* dan *Gbest* Partikel pada *Fold* #1 Iterasi 2

Partikel	m	k	Pbest(2)														f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	
P1	1.5	2	0	1	1	0	1	0	1	0	1	0	0	0	0	0	0.67
P2	2	3	1	1	1	0	1	1	1	0	0	1	1	0	1	0	0.63
P3	2.5	4	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0.67

Sehingga pada *fold* #1 terpilih kombinasi nilai *m*, *k* dan fitur optimal yaitu *m* = 1.5, *k* = 2 dan fitur terpilihnya adalah F2, F3, F5, F7, F9. Kemudian proses ini diulangi sampai *fold* #ke-*K* dimana pada percobaan ini digunakan *fold* sebanyak 3, adapun hasil dari *fold* #2 dan #3 adalah sebagai berikut seperti terlihat pada Tabel 3.41 dan Tabel 3.42.

Tabel 3.41 *Pbest* dan *Gbest* Partikel pada *Fold* #2 Iterasi 2

Partikel	m	k	Pbest(2)														f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	
P1	1.4	3	0	0	0	0	0	1	1	1	1	0	1	1	1	1	0.81
P2	1.9	4	0	1	0	1	0	1	1	1	0	0	0	1	1	1	0.81
P3	1.2	2	0	0	1	1	0	1	1	1	0	0	0	1	1	0	0.82

Tabel 3.42 *Pbest* dan *Gbest* Partikel pada *Fold* #3 Iterasi 2

Partikel	m	k	Pbest(2)														f
			F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	
P1	2.4	3	0	0	1	1	0	0	0	1	0	0	0	0	1	0	0.74
P2	2.6	2	1	0	1	0	1	1	0	1	0	0	0	0	0	1	0.72
P3	1.3	4	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0.76

Pada *fold* #2 terpilih nilai $m = 1.2$, $k = 2$ dan fitur terpilihnya F3, F4, F6, F7, F8, F12, F13 dan untuk *fold* #3 terpilih nilai $m = 1.3$, $k = 4$ dan fitur terpilihnya adalah F8, F12, F13. Secara garis besar didapatkan hasil sebagai berikut seperti terlihat pada Tabel 3.43.

Tabel 3.43 Hasil TVPSO-FKNN dengan 3-Fold CV

Gbest	m	k	fitur	fitness	Keterangan (Asal Gbest)
Fold #1	1.5	2	F2, F3, F5, F7, F9	0.67	Fold 1 Partikel 1 Iterasi 2
Fold #2	1.2	2	F3, F4, F6, F7, F8, F12, F13	0.82	Fold 2 Partikel 3 Iterasi 1
Fold #3	1.3	4	F8, F12, F13	0.76	Fold 3 Partikel 3 Iterasi 2

3.2.3.5 Train FKNN via 3-fold CV dengan seleksi fitur, m , dan k terpilih

Proses terakhir adalah menghitung nilai akurasi FKNN dari 3 *fold* yang sebelumnya dengan menggunakan seleksi fitur, nilai m , dan k yang terpilih dari masing-masing partikel terbaik dari tiap *fold* pada proses TVPSO-FKNN, berikut hasilnya terlihat pada Tabel 3.44 sampai 3.46 untuk pengujian *Fold* # 1, pada Tabel 3.48 sampai 3.50 untuk pengujian *Fold* # 2, dan pada Tabel 3.52 sampai 3.54 untuk pengujian *Fold* # 3.

Tabel 3.44 Hasil FKNN dan *Confusion Matrix* Fold 1 Partikel Fold #1

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
						A	M	B
Data uji1/A	0.37	0.44	0.18	1.00	A	0	2	0
Data uji2/A	0.12	0.69	0.18	1.00	M	0	1	0
Data uji3/B	0.12	0.18	0.69	1.00	B	0	0	1
Data uji 4/M	0.12	0.69	0.18	1.00				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #1 pada *Fold* 1 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{2 + 4}{2 + 2 + 4 + 2} \times 100\% = \frac{6}{10} \times 100\% = 60\%$$

Tabel 3.45 Hasil FKNN dan *Confusion Matrix Fold 2 Partikel Fold #1*

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
						A	M	B
Data uji1/A	0.694	0.123	0.184	1	A	1	0	0
Data uji2/M	0.438	0.378	0.184	1	M	1	1	0
Data uji3/M	0.332	0.484	0.184	1	B	0	0	1
Data uji 4/B	0.184	0.123	0.694	1				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #1 pada *Fold* 2 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{3 + 6}{3 + 1 + 6 + 1} \times 100\% = \frac{9}{11} \times 100\% = 82\%$$

Tabel 3.46 Hasil FKNN dan *Confusion Matrix Fold 3 Partikel Fold #1*

	Kelas A	Kelas M	Kelas B	Jumlah	kelas aktual	kelas prediksi		
						A	M	B
Data uji1/A	0.69	0.18	0.12	1.00	A	1	0	0
Data uji2/M	0.41	0.47	0.12	1.00	M	0	1	0
Data uji3/B	0.18	0.18	0.63	1.00	B	0	0	2
Data uji 4/B	0.18	0.18	0.63	1.00				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #1 pada *Fold* 3 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{4 + 8}{4 + 0 + 8 + 0} \times 100\% = \frac{12}{12} \times 100\% = 100\%$$

Tabel 3.47 Hasil Akurasi 3 –*Fold* CV dengan Partikel Terbaik *Gbest Fold* #1

Fold	Akurasi
1	60%
2	82%
3	100%
Rata-Rata Akurasi	80.6%

Tabel 3.48 Hasil FKNN dan *Confusion Matrix Fold* 1 Partikel *Fold* #2

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	kelas prediksi		
						A	M	B
Data uji1/A	0.63	0.18	0.18	1.00				
Data uji2/A	0.12	0.69	0.18	1.00	A	1	1	0
Data uji3/B	0.12	0.18	0.69	1.00	M	0	1	0
Data uji 4/M	0.12	0.69	0.18	1.00	B	0	0	1

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #2 pada *Fold* 1 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{3 + 6}{3 + 1 + 6 + 1} \times 100\% = \frac{9}{11} \times 100\% = 82\%$$

Tabel 3.49 Hasil FKNN dan *Confusion Matrix Fold 2 Partikel Fold #2*

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
Data uji1/A	0.69	0.12	0.18	1.00	A	1	0	0
Data uji2/M	0.18	0.63	0.18	1.00	M	0	2	0
Data uji3/M	0.18	0.63	0.18	1.00	B	0	1	0
Data uji 4/B	0.18	0.63	0.18	1.00				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold #2* pada *Fold 2* adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{3 + 6}{3 + 1 + 6 + 1} \times 100\% = \frac{9}{11} \times 100\% = 82\%$$

Tabel 3.50 Hasil FKNN dan *Confusion Matrix Fold 3 Partikel Fold #2*

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
Data uji1/A	0.69	0.12	0.18	1.00	A	1	0	0
Data uji2/M	0.18	0.63	0.18	1.00	M	0	1	0
Data uji3/B	0.18	0.12	0.69	1.00	B	0	0	2
Data uji 4/B	0.18	0.12	0.69	1.00				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold #2* pada *Fold 3* adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{4 + 8}{4 + 0 + 8 + 0} \times 100\% = \frac{12}{12} \times 100\% = 100\%$$

Tabel 3.51 Hasil Akurasi 3 –Fold CV dengan Partikel Terbaik Gbest Fold #2

Fold	Akurasi
1	82%
2	82%
3	100%
Rata-Rata Akurasi	88%

Tabel 3.52 Hasil FKNN dan Confusion Matrix Fold 1 Partikel Fold #3

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
						A	M	B
Data uji1/A	0.377	0.439	0.184	1	A	0	1	1
Data uji2/A	0.155	0.247	0.599	1	M	0	0	1
Data uji3/B	0.123	0.184	0.694	1	B	0	0	1
Data uji 4/M	0.124	0.185	0.691	1				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN fold #3 pada Fold 1 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{1 + 2}{1 + 3 + 2 + 3} \times 100\% = \frac{3}{9} \times 100\% = 33.33\%$$

Tabel 3.53 Hasil FKNN dan Confusion Matrix Fold 2 Partikel Fold #3

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
						A	M	B
Data uji1/A	0.694	0.184	0.123	1	A	1	0	0
Data uji2/M	0.694	0.184	0.123	1	M	1	0	1
Data uji3/M	0.321	0.312	0.368	1	B	0	0	1
Data uji 4/B	0.196	0.185	0.619	1				

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #3 pada *Fold* 2 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{2 + 4}{2 + 2 + 4 + 2} \times 100\% = \frac{6}{10} \times 100\% = 60\%$$

Tabel 3.54 Hasil FKNN dan *Confusion Matrix Fold* 3 Partikel *Fold* #3

	Kelas A	Kelas M	Kelas B	Jumlah	Kelas aktual	Kelas prediksi		
Data uji1/A	0.523	0.353	0.124	1	A	A	M	B
Data uji2/M	0.524	0.354	0.123	1		1	0	0
Data uji3/B	0.184	0.184	0.632	1		1	0	0
Data uji 4/B	0.187	0.64	0.172	1		0	1	1

Adapun nilai akurasi dengan menggunakan seleksi fitur, m dan k terpilih dari proses TVPSO-FKNN *fold* #3 pada *Fold* 3 adalah sebagai berikut.

$$accuracy = \frac{TP + TN}{P + N}$$

$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$accuracy = \frac{2 + 4}{2 + 2 + 4 + 2} \times 100\% = \frac{6}{10} \times 100\% = 60\%$$

Tabel 3.55 Hasil Akurasi 3 –*Fold* CV dengan Partikel Terbaik *Gbest Fold* #3

Fold	Akurasi
1	33.33%
2	60%
3	60%
Rata-Rata Akurasi	51.11%

Tabel 3.56 Hasil Rata-rata Akurasi 3 Partikel Terbaik (*Gbest*)

Gbest	Akurasi
Fold #1	80.6%
Fold #2	88%
Fold #3	51.11%

Dapat disimpulkan bahwa pengujian rata-rata akurasi dengan 3-fold CV didapatkan rata-rata akurasi terbaik dengan menggunakan partikel terbaik *fold #2* dengan nilai rata-rata akurasi sebesar 88%.

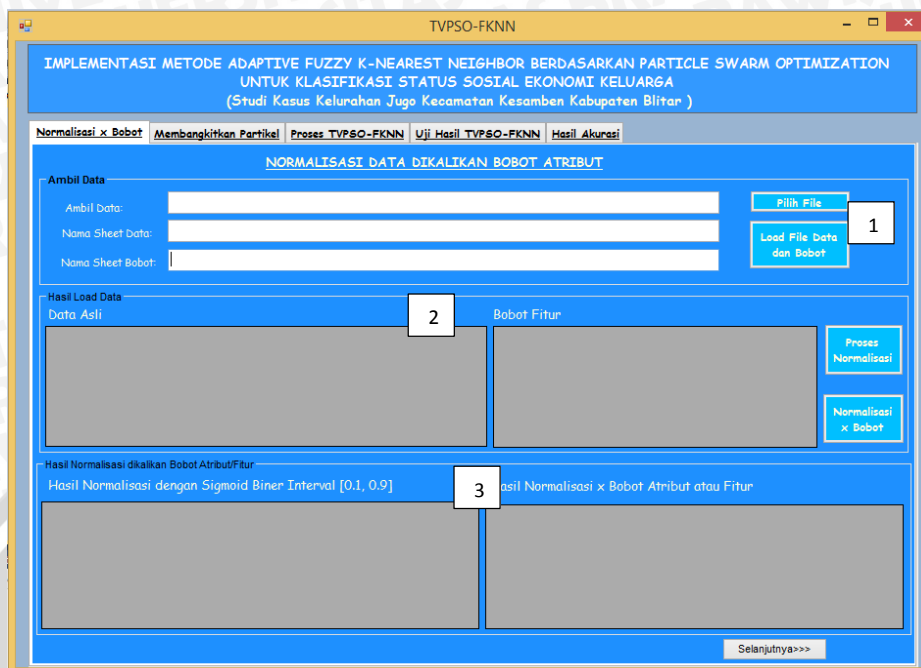
3.2.4 Perancangan *User Interface*

Pada sub bab perancangan *user interface* permasalahan klasifikasi menggunakan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization* yang terdiri dari proses normalisasi data dikalikan bobot atribut, kemudian proses membangkitkan partikel, proses TVPSO-FKNN, menguji hasil TVPSO-FKNN untuk dilakukan proses *train* FKNN, terakhir adalah proses perhitungan hasil akurasi.

- a. Proses normalisasi kali bobot atribut

Dari Gambar 3.16 berikut penjelasannya:

1. *Groupbox* ambil data berisi proses pengambilan data (*load data*) dan pengambilan bobot tiap atribut atau fitur. Adapun isinya adalah pilih lokasi *file* dan nama *sheet* yang akan *diload* dengan menginputkan pada *textbox*. Kemudian *button load file* digunakan untuk mengeksekusi proses ambil data dan bobot tiap atribut atau fitur.
2. *Groupbox* hasil *load data* berisi *datagridview* untuk menampilkan hasil *load data* dan hasil *load bobot*. Serta berisi *button* proses normalisasi yaitu untuk mengeksekusi proses normalisasi dan *button* normalisasi kali bobot merupakan eksekusi proses hasil normalisasi dikalikan bobot atribut.
3. *Groupbox* hasil normalisasi dikalikan bobot atribut berisi *datagridview* untuk menampilkan hasil normalisasi data dan hasil normalisasi data dikalikan bobot atribut atau fitur.
4. *Button* selanjutnya bertujuan untuk masuk ke tab proses selanjutnya



Gambar 3.16 Interface Proses Normalisasi Dikalikan Bobot Atribut

b. Proses membangkitkan partikel



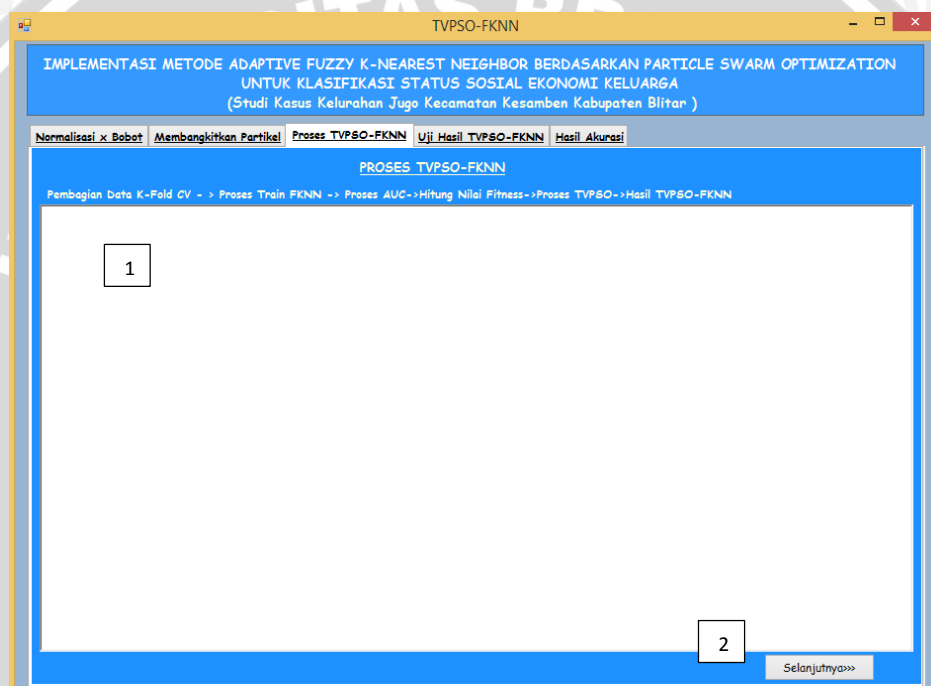
Gambar 3.17 Interface Proses Membangkitkan Partikel

1. *Groupbox* input data dan eksekusi proses membangkitkan partikel. Adapun isinya adalah *input* jumlah partikel, *input* nilai *K*, *input* nilai

interval m dan k , dan *input* jumlah iterasi, serta *button* proses membangkitkan partikel yaitu untuk mengeksekusi proses membangkitkan partikel.

2. *Datagridview* untuk menampilkan hasil membangkitkan partikel.
3. *Button* proses TVPSO-FKNN yaitu untuk mengeksekusi proses TVPSO-FKNN untuk menghasilkan partikel terbaik dimana nilai m , k dan fitur terbaik yang akan digunakan untuk uji dengan *train* FKNN.

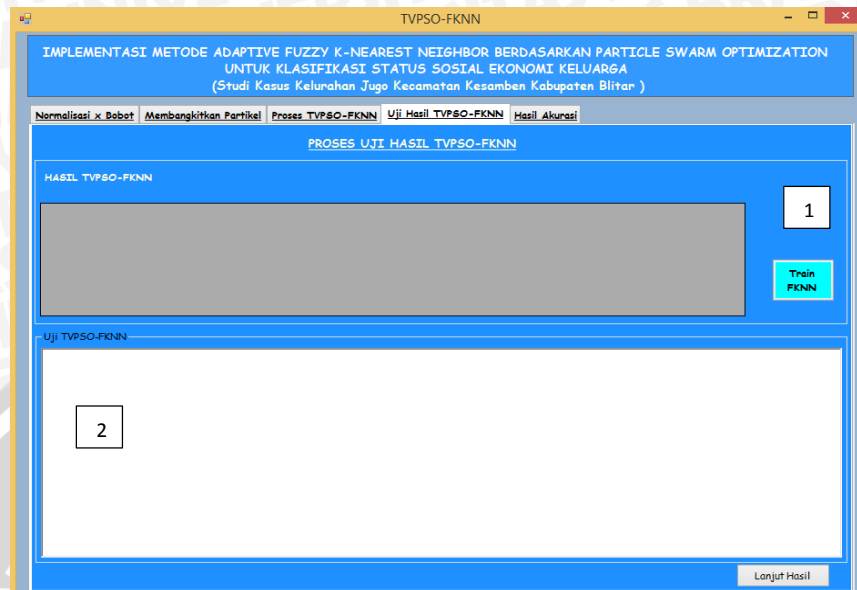
c. Proses TVPSO-FKNN



Gambar 3.18 Interface Proses TVPSO-FKNN

1. *Rich textbox* yang digunakan sebagai tempat hasil dari proses TVPSO-FKNN dimana berisi proses pembagian data *K-Fold CV*, proses *train* FKNN, proses perhitungan nilai AUC, proses menghitung nilai *fitness*, proses TVPSO yaitu *update velocity* dan posisi baru, sampai mendapatkan hasil TVPSO-FKNN yaitu partikel terbaik yang berisi nilai m , k , dan fitur terbaik.
2. *Button* selanjutnya bertujuan untuk masuk ke tab proses selanjutnya

d. Proses uji hasil TVPSO-FKNN



Gambar 3.19 Interface Proses Uji Hasil TVPSO-FKNN

1. *Groupbox* yang berisi *datagridview* untuk menampilkan hasil TVPSO-FKNN yaitu semua partikel terbaik dan *button* proses *train* FKNN untuk mengeksekusi proses klasifikasi FKNN dengan menggunakan partikel terbaik yang berisi nilai m , k , dan fitur terbaik dari proses TVPSO-FKNN..
2. *Richtextbox* yang digunakan sebagai tempat hasil dari proses FKNN.
3. *Button* lanjut hasil bertujuan untuk masuk ke tab hasil akurasi.

e. Hitung hasil akurasi



Gambar 3.20 Interface Hasil Akurasi

1. *Datagridview* untuk menampilkan hasil rata-rata akurasi dari tiap-tiap partikel TVPSO-FKNN yaitu semua partikel terbaik (*Gbest*).

3.2.5 Perancangan Uji Coba dan Evaluasi

Pada tahap ini adalah membuat rancangan uji coba dan evaluasi pada sistem. Pengujian ini dilakukan untuk mengetahui tingkat akurasi klasifikasi menggunakan metode *adaptive fuzzy k-nearest neighbor* berdasarkan *particle swarm optimization*. Adapun pengujian yang dilakukan adalah sebagai berikut:

1. Pengujian untuk mengetahui pengaruh interval nilai m dan k yang digunakan.
2. Pengujian untuk mengetahui pengaruh jumlah partikel yang digunakan.
3. Pengujian untuk mengetahui pengaruh banyaknya iterasi yang akan dilakukan.
4. Pengujian untuk mengetahui pengaruh nilai K pada *K-fold cross validation* dengan 121 data yang akan digunakan.
5. Pengujian untuk mengukur konvergensi pada TVPSO.

3.2.5.1 Pengujian Pengaruh Interval Nilai m Terhadap Rata-rata Akurasi

Pengujian pengaruh nilai interval m terhadap rata-rata akurasi sistem dilakukan untuk melihat pada kombinasi interval m berapa mendapatkan akurasi terbaik. Interval untuk nilai m yaitu interval [1, 2], [2, 4], [4, 6], [6, 8], dan [8, 10]. Nilai m yang digunakan adalah nilai diantara interval tersebut. Adapun perancangannya ditunjukkan pada Tabel 3.56. Nilai *default* yang digunakan yaitu menggunakan nilai interval k [1, 10], untuk nilai K adalah 5, jumlah partikel adalah 8, serta banyak iterasinya 100.

Tabel 3.57 Pengaruh Nilai m Terhadap Rata-rata Akurasi

No	Interval m	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	[1, 2]				
2	[2, 4]				
3	[4, 6]				
4	[6, 8]				

5	[8, 10]				
---	---------	--	--	--	--

3.2.5.2 Pengujian Pengaruh Interval Nilai k Terhadap Rata-rata Akurasi

Pengujian pengaruh nilai k terhadap rata-rata akurasi sistem dilakukan untuk melihat pada kombinasi interval k berapa mendapatkan akurasi terbaik. Interval untuk nilai k yaitu interval [1, 10], [11, 20], [21, 40], [41, 60], dan [61, 80]. Nilai k yang digunakan adalah nilai diantara interval tersebut. Adapun perancangannya ditunjukkan pada Tabel 3.57. Nilai *default* yang digunakan yaitu menggunakan nilai interval m dari pengujian m yang rata-rata akurasinya tertinggi, untuk nilai K adalah 5, jumlah partikel adalah 8, serta banyak iterasinya 100.

Tabel 3.58 Pengaruh Nilai k Terhadap Rata-rata Akurasi

No	Interval k	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	[1, 10]				
2	[11, 20]				
3	[21, 40]				
4	[41, 60]				
5	[61, 80]				

3.2.5.3 Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi

Pengaruh jumlah partikel terhadap rata-rata akurasi dilakukan untuk mengetahui pemberian jumlah partikel berapa akan mendapatkan partikel terbaik yang akan digunakan untuk klasifikasi dan seberapa besar rata-rata akurasi yang didapatkan. Kombinasi jumlah partikel yang akan digunakan adalah 4, 5, 6, 8, dan 10. Adapun rancangannya ditunjukkan pada Tabel 3.58. Nilai *default* yang digunakan yaitu untuk nilai interval k dan nilai interval m adalah nilai yang memiliki rata-rata akurasi tertinggi pada pengujian m dan k , untuk nilai K adalah 5, serta banyak iterasinya 100.

Tabel 3.59 Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi

No	Jumlah Partikel	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	4				

2	5				
3	6				
4	8				
5	10				

3.2.5.4 Pengujian Pengaruh Banyaknya Iterasi Terhadap Rata-rata Akurasi

Pengaruh banyaknya iterasi terhadap akurasi dilakukan untuk mengetahui banyaknya iterasi berapa akan mendapatkan partikel terbaik yang akan digunakan untuk klasifikasi dan seberapa besar akurasi yang didapatkan. Kombinasi banyaknya iterasi yang akan digunakan adalah 10, 15, 20, 25, dan 30. Adapun rancangannya ditunjukkan pada Tabel 3.59. Nilai *default* yang digunakan yaitu untuk nilai interval k , nilai interval m , dan jumlah partikel adalah nilai k , m , dan jumlah partikel yang memiliki rata-rata akurasi tertinggi pada pengujian sebelumnya, sedangkan nilai K adalah 5.

Tabel 3.60 Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi

No	Banyak Iterasi	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	10				
2	50				
3	100				
4	150				
5	250				

3.2.5.5 Pengujian Pengaruh Nilai K pada K -Fold CV Terhadap Rata-rata Akurasi

Pengaruh nilai K yang pada proses *K-fold cross validation* terhadap rata-rata akurasi digunakan untuk mengetahui nilai K berapa yang dapat menghasilkan rata-rata akurasi terbaik. Adapun kombinasi nilai K yang akan digunakan adalah 4, 5, 6, 8, dan 10. Adapun rancangannya ditunjukkan pada Tabel 3.60. Nilai *default* yang digunakan yaitu untuk nilai interval k , nilai interval m , jumlah partikel, dan

banyak iterasinya adalah nilai yang memiliki rata-rata akurasi tertinggi pada pengujian m , k , jumlah partikel, dan banyak iterasi.

Tabel 3.61 Pengaruh Nilai K Terhadap Rata-rata Akurasi

No	Nilai K pada K-Fold CV	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	4				
2	5				
3	6				
4	8				
5	10				

3.2.5.6 Pengujian untuk Mengukur Konvergensi pada TVPSO

Pengujian untuk mengukur konvergensi pada TVPSO digunakan untuk mengetahui pada iterasi seberapa pencarian solusi pada TVPSO berada pada titik konvergensi. Adapun rancangannya ditunjukkan pada Tabel 3.61. Nilai *default* yang digunakan yaitu untuk nilai interval k , nilai interval m , jumlah partikel, banyak iterasi, dan nilai K adalah nilai yang memiliki rata-rata akurasi tertinggi pada pengujian m , k , jumlah partikel, banyak iterasi, dan pengujian K .

Tabel 3.62 Hasil Mengukur Konvergensi pada TVPSO

No	Fold	Jumlah Iterasi	Nilai <i>Fitness</i> Terbaik
1	#1		
2	#2		
.	.		
.	.		
K	#K		

BAB IV

IMPLEMENTASI

Pada bab implementasi akan dibahas mengenai implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga berdasarkan perancangan yang telah dijelaskan pada bab sebelumnya.

4.1 Lingkungan Implementasi

Adapun lingkungan implementasinya yaitu lingkungan perangkat keras (*hardware*) dan perangkat lunak (*software*). Selanjutnya implementasi program dan implementasi *interface*.

4.1.1 Lingkungan Perangkat Keras

Untuk spesifikasi perangkat keras yang digunakan dalam implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga adalah:

1. Prosesor Intel® Core™2 Duo 2.00 GHz
2. RAM 4.00 GB
3. Hardisk 320 GB
4. Monitor 14"
5. Mouse
6. Keyboard

4.1.2 Lingkungan Perangkat Lunak

Untuk spesifikasi perangkat lunak yang digunakan dalam implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga adalah:

1. Operating system windows 8 64 bit
2. Microsoft office 2013
3. Visual studio 2010
4. Bahasa pemrograman C#.
5. Microsoft visio 2007

4.2 Implementasi Program

Pada sub-bab implementasi program akan dibahas implementasi berdasarkan perancangan sistem pada bab sebelumnya yaitu implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga. Implementasi pembuatan program aplikasi dibuat dengan menggunakan bahasa pemrograman C#. Program aplikasi yang dibuat memiliki 3 proses utama yaitu proses normalisasi dikalikan bobot atribut, proses TVPSO-FKNN (*Time Variant Particle Swarm Optimization-Fuzzy K-Nearest Neighbor*), dan proses uji hasil TVPSO-FKNN.

4.2.1 Proses Normalisasi Dikalikan Bobot Atribut

Proses normalisasi dikalikan bobot atribut merupakan proses menormalisasikan data agar seluruh data berada pada interval yang sama dan untuk mengecilkan data yang rentang nilainya terlalu besar, kemudian data hasil normalisasi dikalikan dengan bobot atribut atau fitur. Adapun sub proses yang dilakukan diantaranya adalah proses *load* data dan bobot, kemudian proses normalisasi, proses terakhir yang dilakukan adalah mengalikan hasil normalisasi dengan bobot atribut atau fiturnya.

4.2.1.1 Proses Load Data dan Bobot Atribut

Proses *load* data merupakan proses mengambil *file dataset* yang diambil dalam bentuk *file excel* dengan format *.xls* yang datanya akan digunakan untuk proses selanjutnya. Pada *Source Code* 4.1 adalah *source code* proses *load* data.

```

1 private void loadfile_Click(object sender, EventArgs e)
2 {
3     string pathConn = "Provider=Microsoft.Jet.OLEDB.4.0;Data
4 Source=" + txtPath.Text + ";Extended Properties=\\Excel
5 8.0;HDR=Yes;";
6     OleDbConnection conn = new OleDbConnection(pathConn);
7     var myDataAdapter = new OleDbDataAdapter("SELECT * FROM [" +
8 txtname.Text + "$]", conn);
9     var dtnew = new DataTable();
10    myDataAdapter.Fill(dtnew);
11    dgvDataasli.DataSource = dtnew;
12    myDataAdapter = new OleDbDataAdapter("SELECT * FROM [" +
13 bobot.Text + "$]", conn);
14    dtnew = new DataTable();
15    myDataAdapter.Fill(dtnew);
16    dgvbobot.DataSource = dtnew;

```

17	}
----	---

Source Code 4.1 Listing Code Proses Load Data dan Bobot Atribut.

Penjelasan Source Code 4.1 adalah:

1. Baris 3-6 merupakan data koneksi antara program dan data excel.
2. Baris 7-10 dan 12-15 merupakan proses pemanggilan data dari excel ke program.
3. Baris 11 dan 16 merupakan proses membaca data dan menampilkan data ke dalam *datagridview*.

4.2.1.2 Proses Normalisasi

Proses normalisasi adalah proses menormalisasikan data hasil *load file* agar berada pada interval yang sama, pada penelitian ini menggunakan interval 0.1 sampai 0.9. Source code proses normalisasi ditunjukkan pada Source Code 4.2.

```

1 private void proses_normalisasi_Click(object sender, EventArgs e)
2 {
3     jumData = dgvDataasli.Rows.Count - 1;
4     jumFitur = dgvDataasli.ColumnCount - 3;
5     dataAll = new double[jumData, jumFitur];
6     dataIdentitas = new string[jumData];
7     dataClass = new string[jumData];
8     for (int i = 0; i < jumData; i++)
9     {
10        dataIdentitas[i] = dgvDataasli[0, i].Value.ToString();
11        dataClass[i] = dgvDataasli[jumFitur + 1, i].Value.ToString();
12        for (int j = 0; j < jumFitur; j++)
13        {
14            dataAll[i, j] =
15                Convert.ToDouble(dgvDataasli[j+1,i].Value.ToString());
16        }
17        double[] tempMax = new double[jumFitur];
18        double[] tempMin = new double[jumFitur];
19        double valMin, valMax;
20        for (int i = 0; i < jumFitur; i++)
21        {
22            valMin = dataAll[0, i];
23            valMax = dataAll[0, i];
24            for (int j = 0; j < jumData; j++)
25            {
26                if (valMax < dataAll[j, i])
27                {
28                    valMax = dataAll[j, i];
29                }
30                if (valMin > dataAll[j, i])
31                {
32                    valMin = dataAll[j, i];
33                }
34            }
35            tempMax[i] = valMax;
36            tempMin[i] = valMin;

```

```

36     }
37     dataNormalisasi = new double[jumData, jumFitur];
38     double tempNormal = 0;
39     for (int i = 0; i < jumData; i++)
40     {
41         for (int j = 0; j < jumFitur; j++)
42         {
43             tempNormal = ((0.8 * (dataAll[i, j] - tempMin[j])) /
44             (tempMax[j] - tempMin[j])) + 0.1;
45             if (!double.IsNaN(tempNormal))
46             {
47                 dataNormalisasi[i, j] = tempNormal;
48             }
49             else
50             {
51                 dataNormalisasi[i, j] = 0.1;
52             }
53             //menampilkan data normalisasi
54             dgvHasilNormalisasi.ColumnCount = jumFitur+1;
55             dgvHasilNormalisasi.RowCount = jumData;
56             //Menampilkan Fitur
57             dgvHasilNormalisasi.Columns[0].Name = "KK";
58             dgvHasilNormalisasi.Columns[0].Width = 50;
59             x = 0;
60             while (x < jumFitur)
61             {
62                 dgvHasilNormalisasi.Columns[x+1].Name = "F" + (x +
63                 1).ToString();
64                 dgvHasilNormalisasi.Columns[x+1].Width = 50;
65                 x++;
66             }
67             //Menampilkan Data normalisasi
68             for (int i = 0; i < jumData; i++)
69             {
70                 dgvHasilNormalisasi[0, i].Value = dataIdentitas[i];
71             }
72             for (int i = 0; i < jumData; i++)
73             {
74                 for (int j = 0; j < jumFitur; j++)
75                 {
76                     dgvHasilNormalisasi[j+1, i].Value = dataNormalisasi[i, j];
77                 }
78             }

```

Source Code 4.2 Listing Code Proses Normalisasi.

Penjelasan *Source Code 4.2* adalah:

1. Baris 3-7, 17-19, dan 37-38 merupakan inisialisasi variabel.
2. Baris 8-16 adalah mengambil data dari *datagridview* hasil *load* data.
3. Baris 20-36 adalah proses mencari nilai minimum dan maksimum pada tiap-tiap fitur.
4. 39-52 adalah proses hitung normalisasi.
5. 53-78 adalah proses menampilkan data hasil normalisasi.

4.2.1.3 Proses Normalisasi Kali Bobot Atribut

Proses normalisasi kali bobot atribut adalah proses mengalikan data hasil normalisasi dengan bobot masing-masing atribut atau fitur. Hal ini dilakukan karena pemberian kelas aktual diberikan berdasarkan pada bobot tiap-tiap atribut atau fitur. *Source code* proses normalisasi kali bobot atribut ditunjukkan pada *Source Code 4.3*.

```

1  private void normalisasixbobot_Click(object sender, EventArgs e)
2  {
3      bobotfitur = new double[jumFitur];
4      dataFitur = new string[jumFitur];
5      dataNormalKaliBobot = new double[jumData, jumFitur];
6      //mengambil nilai bobot
7      double tempTot = 0;
8      for (int i = 0; i < jumFitur; i++)
9      {
10         dataFitur[i] = dgvbobot.Columns[i + 1].HeaderText;
11         tempTot += Convert.ToDouble(dgvbobot[i+1,0].Value.ToString());
12     }
13     for (int i = 0; i < jumFitur; i++)
14     {
15         bobotfitur[i]=
16         Convert.ToDouble(dgvbobot[i+1,0].Value.ToString())/tempTot;
17     }
18     //Data Hasil Normalisasi kali bobot
19     for (int i = 0; i < jumData; i++)
20     {
21         for (int j = 0; j < jumFitur; j++)
22         {
23             dataNormalKaliBobot[i,j] = dataNormalisasi[i,j]*bobotfitur[j];
24         }
25         //menampilkan data normalisasi kali bobot
26         dgvhasilkalibobot.ColumnCount = jumFitur+1;
27         dgvhasilkalibobot.RowCount = jumData;
28         //menampilkan fitur
29         dgvhasilkalibobot.Columns[0].Name = "KK";
30         dgvhasilkalibobot.Columns[0].Width = 50;
31         x = 0;
32         while (x < jumFitur)
33         {
34             dgvhasilkalibobot.Columns[x+1].Name = "F" + (x+1).ToString();
35             dgvhasilkalibobot.Columns[x + 1].Width = 50;
36             x++;
37         }
38         for (int i = 0; i < jumData; i++)
39         {
40             dgvhasilkalibobot[0, i].Value = dataIdentitas[i];
41         }
42         //Menampilkan Data normalisasi kali bobot
43         for (int i = 0; i < jumData; i++)
44         {
45             for (int j = 0; j < jumFitur; j++)
46             {
47                 dgvhasilkalibobot[j+1, i].Value = dataNormalKaliBobot[i, j];

```

48	}
49	}}

Source Code 4.3 Listing Code Proses Normalisasi Dikalikan Bobot Atribut.

Penjelasan Source Code 4.3 adalah:

1. Baris 3-7 merupakan inisialisasi variabel.
2. Baris 8-17 adalah proses mengambil nilai bobot tiap-tiap atribut atau fitur.
3. Baris 19-24 adalah proses hitung hasil normalisasi kali bobot atribut atau fitur.
4. Baris 26-49 adalah menampilkan data hasil normalisasi kali bobot atribut.

4.2.2 Proses TVPSO-FKNN

Pada proses TVPSO-FKNN terdapat beberapa sub proses di dalamnya yaitu proses membangkitkan partikel, proses *K-fold cross validation*, proses inisialisasi parameter, proses *train* FKNN, proses hitung *area under ROC curve*, proses hitung *fitness*, proses *update velocity* dan posisi, serta proses *update pbest* dan *gbest*.

4.2.2.1 Proses Membangkitkan Partikel

Proses membangkitkan partikel adalah proses awal dari metode TVPSO-FKNN yaitu proses meng-*encode* nilai *m* dan *k* secara random berdasarkan interval yang diinputkan, sedangkan *encode* fitur dengan angka random 0 atau 1. Dimana hasil *encode* inilah yang akan digunakan pada proses selanjutnya. *Source code* proses membangkitkan partikel ditunjukkan pada Source Code 4.4.

1	<code>private void prosesencodepartikel_Click(object sender, EventArgs e)</code>
2	<code>{</code>
3	<code> dgvhasilencode.DataSource = null;</code>
4	<code> dgvhasilencode.Rows.Clear();</code>
5	<code> kFold = Convert.ToInt32(txt_K.Text);</code>
6	<code> jumPartikel = Convert.ToInt32(txtjumpartikel.Text);</code>
7	<code> intervalK1 = Convert.ToInt32(txtbk.Text);</code>
8	<code> intervalK2 = Convert.ToInt32(txtak.Text);</code>
9	<code> intervalM1 = Convert.ToDouble(txtbm.Text);</code>
10	<code> intervalM2 = Convert.ToDouble(txtam.Text);</code>
11	<code> totalPartikel = jumPartikel * kFold;</code>
12	<code> valM = new double[totalPartikel];</code>
13	<code> valK = new int[totalPartikel];</code>
14	<code> binerAll = new int[totalPartikel, jumFitur];</code>
15	<code> x = 0;</code>
16	<code> double tempRandomM = 0;</code>
17	<code> int tempRandomK = 0;</code>
18	<code> Random rand = new Random();</code>
19	<code> while (x < totalPartikel)</code>
20	<code> {</code>

```

21 tempRandomK = rand.Next(intervalK1, intervalK2 + 1);
22 tempRandomM = randomDouble(rand, intervalM1, intervalM2);
23 valM[x] = Math.Round (tempRandomM,4);
24 valK[x] = tempRandomK;
25 for (int i = 0; i < jumFitur; i++)
26 {
27     binerAll[x, i] = rand.Next(0, 2);
28 }
29 x++;
30 }
31 dgvhasilencode.ColumnCount = jumFitur + 3; ;
32 //menampilkan fitur
33 dgvhasilencode.Columns[0].Name = "Partikel";
34 dgvhasilencode.Columns[0].Width = 50;
35 dgvhasilencode.Columns[1].Name = "m";
36 dgvhasilencode.Columns[1].Width = 50;
37 dgvhasilencode.Columns[2].Name = "k";
38 dgvhasilencode.Columns[2].Width = 50;
39 x = 0;
40 while (x < jumFitur)
41 {
42     dgvhasilencode.Columns[x + 3].Name = "F" + (x + 1).ToString();
43     dgvhasilencode.Columns[x + 3].Width = 50;
44     x++;
45 }
46 x = 0;
47 int tempPart = 0;
48 int tempK = 0;
49 row = new string[] { "Fold 1" };
50 dgvhasilencode.Rows.Add(row);
51 for (int i = 0; i < totalPartikel; i++)
52 {
53     if (tempPart == jumPartikel)
54     {
55         tempK++;
56         row = new string[] { "Fold" + (tempK + 1).ToString() };
57         dgvhasilencode.Rows.Add(row);
58         tempPart = 0;
59     }
60     row = new string[3 + jumFitur];
61     row[0] = "P" + (tempPart + 1).ToString();
62     row[1] = valM[i].ToString();
63     row[2] = valK[i].ToString();
64     x = 3;
65     for (int j = 0; j < jumFitur; j++)
66     {
67         row[x] = binerAll[i, j].ToString();
68         x++;
69     }
70     dgvhasilencode.Rows.Add(row);
71     x = 0;
72     tempPart++;
73 }

```

Source Code 4.4 Listing Code Proses Membangkitkan Partikel.

Penjelasan Source Code 4.4 adalah:

1. Baris 3-4 merupakan proses me-refresh-kan kembali *datagridview*.

2. Baris 5-10 mengambil data dari inputan.
3. Baris 11 menghitung total partikel.
4. Baris 12-17 adalah inisialisasi variabel.
5. Baris 18 membuat nilai random.
6. Baris 19-30 adalah mengambil dan menyimpan nilai random.
7. Baris 32-73 adalah menampilkan hasil membangkitkan partikel.

4.2.2.2 Proses K-Fold Cross Validation

Proses *K-fold cross validation* merupakan proses pembagian data sebanyak *K* untuk diujikan, dimana jika 1 bagian akan menjadi *data testing* dan bagian yang lain akan menjadi *data training*, hal tersebut dilakukan hingga semua bagian menjadi *data testing*. *Source code* proses *K-fold cross validation* ditunjukkan pada *Source Code 4.5*.

```

1  ArrayList dataKelasA= new ArrayList();
2  ArrayList dataKelasM=new ArrayList();
3  ArrayList dataKelasB=new ArrayList();
4  for (int i = 0; i < jumData; i++)
5  {
6      string cekKelas = dgvDataasli["Kelas",i].Value.ToString();
7      if (cekKelas == "A")
8      {
9          dataKelasA.Add(i);
10     }
11     if (cekKelas == "M")
12     {
13         dataKelasM.Add(i);
14     }
15     if (cekKelas == "B")
16     {
17         dataKelasB.Add(i);
18     }
19     Random rand1 = new Random();
20     jumDataPembagianKFold = new int[kFold];
21     int tempJumBagiData = 0;
22     tempJumBagiData = jumData / kFold;
23     int tempSisaData = jumData;
24     for (int i = 0; i < kFold; i++)
25     {
26         jumDataPembagianKFold[i] = tempJumBagiData;
27         tempSisaData = tempSisaData - tempJumBagiData;
28     }
29     if (tempSisaData > 0)
30     {
31         x = 0;
32         while (tempSisaData > 0)
33         {
34             jumDataPembagianKFold[x] += 1; ;
35             tempSisaData--;

```

```
36     x++;
37     if (x == kFold - 1)
38     {
39         x = 0;
40     }
41     dataPembagianKFold = new int[kFold][];
42     x = 0;
43     int cekIndeks = 0;
44     int terpilih = 0;
45     int tempClassStop = 0;
46     for (int i = 0; i < kFold; i++)
47     {
48         dataPembagianKFold[i] = new int[jumDataPembagianKFold[i]];
49         //Menampilkan
50         rtbPembagianDataKFold.Text += "Data yang terpilih untuk K-Fold #"
51 + (i + 1).ToString() + "\n";
52         ArrayList dataTerpilih = new ArrayList();
53         x = 0;
54         while (x < jumDataPembagianKFold[i])
55         {
56             for (int j = tempClassStop; j < 3; j++)
57             {
58                 if (x == jumDataPembagianKFold[i])
59                 {
60                     tempClassStop = j;
61                     j = 3;
62                 }
63                 else
64                 {
65                     if (j == 0)
66                     {
67                         if (dataKelasA.Count != 0)
68                         {
69                             cekIndeks = rand1.Next(0, dataKelasA.Count);
70                             terpilih = Convert.ToInt32(dataKelasA[cekIndeks]);
71                             dataTerpilih.Add(terpilih);
72                             dataKelasA.RemoveAt(cekIndeks);
73                             x++;
74                         }
75                     }
76                     if (j == 1)
77                     {
78                         if (dataKelasM.Count != 0)
79                         {
80                             cekIndeks = rand1.Next(0, dataKelasM.Count);
81                             terpilih = Convert.ToInt32(dataKelasM[cekIndeks]);
82                             dataTerpilih.Add(terpilih);
83                             dataKelasM.RemoveAt(cekIndeks);
84                             x++;
85                         }
86                     }
87                     if (j == 2)
88                     {
89                         if (dataKelasB.Count != 0)
90                         {
91                             cekIndeks = rand1.Next(0, dataKelasB.Count);
92                             terpilih = Convert.ToInt32(dataKelasB[cekIndeks]);
93                             dataTerpilih.Add(terpilih);
94                             dataKelasB.RemoveAt(cekIndeks);
95                             x++;
96                         }
97                     }
98                 }
99             }
100         }
101     }
102 }
```

```

95     tempClassStop = 0;
96     }}
97     for (int j = 0; j < jumDataPembagianKFold[i]; j++)
98     {
99         dataPembagianKFold[i][j] = Convert.ToInt32(dataTerpilih[j]);
100         rtbPembagianDataKFold.Text +=
101         dataIdentitas[Convert.ToInt32(dataTerpilih[j])] + "-";
102     }}

```

Source Code 4.5 Listing Code Proses K-Fold Cross Validation.

Penjelasan Source Code 4.5 adalah:

1. Baris 1-3 dan 52 variabel arraylist untuk menyimpan hasil sementara.
2. Baris 4-18 menghitung data untuk tiap-tiap kelas.
3. Baris 20-23 dan 41-45 inialisasi variabel.
4. Baris 24-40 adalah proses pembagian data sebanyak K yang diinisialisasi oleh variabel $kFold$.
5. Baris 46-96 mengambil data secara random.
6. Baris 97-102 adalah menampilkan data yang terpilih untuk masing-masing K -fold (hasil K -fold) berdasarkan data identitas.

4.2.2.3 Proses Inisialisasi Parameter

Proses inisialisasi parameter adalah proses menginisialisasi dengan *input*-an nilai t_{max} , C_{if} , C_{1i} , C_{2f} , C_{2i} , W_{min} , W_{max} untuk menghitung nilai tiap-tiap parameter yang digunakan pada proses TVPSO-FKNN yaitu nilai w , c_1 , dan c_2 pada tiap-tiap iterasi. Source code proses inisialisasi parameter ditunjukkan pada Source Code 4.6.

```

1  for (int i = 0; i < tMax; i++)
2  {
3      w[i] = wmin + ((wMax - wmin) *
4      (Convert.ToDouble(Convert.ToDouble(tMax - (i + 1)) /
5      Convert.ToDouble(tMax))));
6      c1[i] = ((c1f - c1i) * Convert.ToDouble(Convert.ToDouble(i + 1) /
7      Convert.ToDouble(tMax))) + c1i;
8      c2[i] = ((c2f - c2i) * Convert.ToDouble(Convert.ToDouble(i + 1) /
9      Convert.ToDouble(tMax))) + c2i;
10     rtbPembagianDataKFold.Text += "\nw - " + (i + 1).ToString() + " = "
11     + w[i].ToString();
12     rtbPembagianDataKFold.Text += "\nc1 - " + (i + 1).ToString() + " = "
13     + c1[i].ToString();
14     rtbPembagianDataKFold.Text += "\nc2 - " + (i + 1).ToString() + " = "
15     + c2[i].ToString();
16 }

```

Source Code 4.6 Listing Code Proses Inisialisasi Parameter.

Penjelasan *Source Code* 4.6 adalah:

1. Baris 1 merupakan awal proses *looping*.
2. Baris 3-9 proses hitung nilai parameter.
3. Baris 10-16 adalah menampilkan hasil menghitung nilai parameter.

4.2.2.4 Proses *Train* FKNN

Proses *train* FKNN adalah proses menjalankan metode klasifikasi dengan menggunakan parameter m , k , dan fitur yang bernilai 1 dari hasil membangkitkan partikel hingga mendapatkan nilai TPR, TNR, dan FPR untuk tiap-tiap partikel.

Source code proses *train* FKNN ditunjukkan pada *Source Code* 4.7.

```

1  //Pemilihan Fitur
2  for (int i = 0; i < jumFitur; i++)
3  {
4      if (binerFitur[tempIndeksPart, i] == 1)
5      {
6          tempFiturTerpilih.Add(i);
7      }
8      //Menyimpan Jumlah fitur terpilih untuk masing2 partikel
9      jumFiturTerpilih[tempFold, tempIndeksPart] =
10     tempFiturTerpilih.Count;
11     for (int i = 0; i < dataPembagianKFold.GetLength(0); i++)
12     {
13         for (int j = 0; j < dataPembagianKFold[i].Length; j++)
14         {
15             if (i == tempFold)
16             {
17                 tempDataTerpilih = dataPembagianKFold[i][j];
18                 tempDataTesting.Add(tempDataTerpilih);
19             }
20             else
21             {
22                 tempDataTerpilih = dataPembagianKFold[i][j];
23                 tempDataTraining.Add(tempDataTerpilih);
24             }
25         }
26     }
27     //proses hitung jarak
28     for (int i = 0; i < tempJumDatTest; i++)
29     {
30         for (int j = 0; j < tempJumDatTrain; j++)
31         {
32             tempResultJarakKedekatan = 0;
33             for (int k = 0; k < tempJumFitur; k++)
34             {
35                 tempHitungJarakKedekatan =
36                 Math.Abs(dataNormalKaliBobot[Convert.ToInt32(tempDataTesting[i]),
37                 Convert.ToInt32(tempFiturTerpilih[k])] -
38                 dataNormalKaliBobot[Convert.ToInt32(tempDataTraining[j]),
39                 Convert.ToInt32(tempFiturTerpilih[k])]);
40                 tempHitungJarakKedekatan = tempHitungJarakKedekatan *
41                 tempHitungJarakKedekatan;
42                 tempResultJarakKedekatan += tempHitungJarakKedekatan;
43             }
44         }
45     }

```

```

42     tempResultJarakKedekatan = Math.Pow(tempResultJarakKedekatan,
43     0.5);
44     if (tempResultJarakKedekatan == 0)
45     {
46         tempResultJarakKedekatan = 0.0001;
47     }
48     jarakKedekatan[i, j] = tempResultJarakKedekatan;
49 }
50 //PROSES SORTING
51 for (int i = 0; i < tempJumDatTest; i++)
52 {
53     for (int j = 0; j < tempJumDatTrain; j++)
54     {
55         tempMin = -1;
56         for (int k = 0; k < tempJumDatTrain; k++)
57         {
58             strTemp = jarakKedekatan[i, k];
59             if (strTemp != -1)
60             {
61                 if (tempMin == -1)
62                 {
63                     indeksTerpilih = k;
64                     tempMin = strTemp;
65                 }
66                 if (Convert.ToDouble(jarakKedekatan[i, k]) < tempMin)
67                 {
68                     tempMin = Convert.ToDouble(jarakKedekatan[i, k]);
69                     indeksTerpilih = k;
70                 }
71             }
72             jarakKedekatanSorting[i, j] = jarakKedekatan[i, indeksTerpilih];
73             jarakKedekatan[i, indeksTerpilih] = -1;
74             tempIndeksSorting[i, j] = indeksTerpilih;
75         }
76     }
77     // proses hitung keanggotaan kelas
78     for (int i = 0; i < 3; i++)
79     {
80         for (int j = 0; j < 3; j++)
81         {
82             double tempHasil = 0;
83             if (i == j)
84             {
85                 tempHasil = 0.51 + ((Convert.ToDouble(jumDatTrainClass[i]) /
86                 Convert.ToDouble(tempJumDatTrain)) * 0.49);
87             }
88             else
89             {
90                 tempHasil = ((Convert.ToDouble(jumDatTrainClass[i]) /
91                 Convert.ToDouble(tempJumDatTrain)) * 0.49);
92             }
93             derajatKeanggotaan[i, j] = tempHasil;
94         }
95     }
96     //proses hitung keanggotaan data testing
97     for (int i = 0; i < 3; i++)
98     {
99         for (int j = 0; j < tempJumDatTest; j++)
100         {
101             int tempi = i;
102             int tempj = j;
103             double tempHasil = 0;

```

```

101     double tempPembilang = 0, tempPembagi = 0;
102     for (int k = 0; k < valK[tempPart]; k++)
103     {
104         If
105         (dataClass[Convert.ToInt32(tempDataTraining[tempIndeksSorting[j],
106 k]]]) == "A")
107         {
108             tempj = 0;
109         }
110         if (dataClass[Convert.ToInt32(tempDataTraining[tempIndeksSorting[j],
111 k]]]) == "M")
112         {
113             tempj = 1;
114         }
115         if
116         (dataClass[Convert.ToInt32(tempDataTraining[tempIndeksSorting[j],
117 k]]]) == "B")
118         {
119             tempj = 2;
120         }
121         tempPembilang += derajatKeanggotaan[tempi, tempj] * (Math.Pow(1
122 / jarakKedekatanSorting[j, k], (2 / (valM[tempPart] - 1))));
123         tempPembagi += (Math.Pow(1 / jarakKedekatanSorting[j, k], (2 /
124 (valM[tempPart] - 1))));
125     }
126     tempHasil = tempPembilang / tempPembagi;
127     hasilFKNN[i, j] = tempHasil;
128     }}
129 //CONFUSION MATRIX
130 int[,] hasilConfusionMatrix = new int[3, 3];
131 for (int i = 0; i < tempJumDatTest; i++)
132 {
133     double tempMax = 0;
134     int tempIndeks = 0;
135     for (int j = 0; j < 3; j++)
136     {
137         if (hasilFKNN[j, i] > tempMax)
138         {
139             tempMax = hasilFKNN[j, i];
140             tempIndeks = j;
141         }
142     }
143     int tempj = 0;
144     if (dataClass[Convert.ToInt32(tempDataTesting[i])] == "A")
145     {
146         tempj = 0;
147     }
148     if (dataClass[Convert.ToInt32(tempDataTesting[i])] == "M")
149     {
150         tempj = 1;
151     }
152     if (dataClass[Convert.ToInt32(tempDataTesting[i])] == "B")
153     {
154         tempj = 2;
155     }
156     hasilConfusionMatrix[tempj, tempIndeks] += 1;
157 }
158 // Menampilkan CONFUSION MATRIX
159 for (int i = 0; i < 3; i++)
160 {

```

```

160     for (int j = 0; j < 3; j++)
161     {
162         rtbPembagianDataKFold.Text += hasilConfusionMatrix[i, j] + "\t";
163     }
164     rtbPembagianDataKFold.Text += "\n";
165 }
166 int tp = 0, tn = 0, fp = 0, fn = 0;
167 // hitung TP, TN, FP, FN
168 tp = hasilConfusionMatrix[0, 0] + hasilConfusionMatrix[1, 1] +
169     hasilConfusionMatrix[2, 2];
170 tn = (hasilConfusionMatrix[0, 0] + hasilConfusionMatrix[1, 1]) +
171     (hasilConfusionMatrix[0, 0] + hasilConfusionMatrix[2, 2]) +
172     (hasilConfusionMatrix[1, 1] + hasilConfusionMatrix[2, 2]);
173 fp = (hasilConfusionMatrix[1, 0] + hasilConfusionMatrix[2, 0]) +
174     (hasilConfusionMatrix[0, 1] + hasilConfusionMatrix[2, 1]) +
175     (hasilConfusionMatrix[0, 2] + hasilConfusionMatrix[1, 2]);
176 fn = (hasilConfusionMatrix[0, 1] + hasilConfusionMatrix[0, 2]) +
177     (hasilConfusionMatrix[1, 0] + hasilConfusionMatrix[1, 2]) +
178     (hasilConfusionMatrix[2, 0] + hasilConfusionMatrix[2, 1]);
179 double tpRate = 0, tnRate = 0, fpr = 0;
180 // hitung TPR, TNR, FPR
181 tpRate = Convert.ToDouble(tp) / Convert.ToInt32(tp + fn);
182 tnRate = Convert.ToDouble(tn) / Convert.ToInt32(tn + fp);
183 fpr = 1 - tnRate;
184 valTpRate[tempFold, tempIndeksPart] = tpRate;
185 valFpRate[tempFold, tempIndeksPart] = fpr;
186 valX.Add(fpr);
187 valY.Add(tpRate);
188 tempPart++;
189 tempIndeksPart++;
190 }

```

Source Code 4.7 Listing Code Proses Train FKNN.

Penjelasan *Source Code 4.7* adalah:

1. Baris 2-7 proses memilih fitur.
2. Baris 166 dan 179 inialisasi variabel dengan memberikan nilainya sama dengan 0.
3. Baris 9-24 proses menyimpan jumlah fitur terpilih untuk tiap-tiap partikel.
4. Baris 26-49 adalah proses hitung jarak.
5. Baris 51-74 adalah proses *shorting* hitung jarak.
6. Baris 76-92 adalah proses hitung keanggotaan kelas.
7. Baris 94-128 adalah proses hitung keanggotaan *data testing* terhadap masing-masing kelas.
8. Baris 130-156 adalah proses hitung dan membuat *confusion matrix*.
9. Baris 158-165 adalah proses menampilkan hasil *confusion matrix*.
10. Baris 169-178 adalah proses menghitung nilai TP, TN, FP, dan FN.

11. Baris 181-183 adalah hitung nilai TPR, TNR, dan FPR.
12. Baris 184-187 adalah proses menyimpan hasil TPR, dan FPR untuk semua partikel.

4.2.2.5 Proses Hitung Area Under ROC Curve

Proses Menghitung nilai *area under ROC curve* merupakan proses menghitung nilai daerah di bawah kurva antara nilai TPR (y) dan FPR (x) tiap-tiap partikel pada 1 *fold*, dimana misalnya *K-fold* dengan nilai $K = 3$ maka akan ada 3 *fold* pembagian yaitu *fold* #1, *fold* #2, dan *fold* #3. *Source code* proses hitung *area under ROC Curve* ditunjukkan pada *Source Code* 4.8.

```

1 public void hitungKurvaROC(ArrayList x, ArrayList y)
2 {
3     ArrayList newX = new ArrayList();
4     ArrayList newY = new ArrayList();
5     ArrayList tempIndeks = new ArrayList();
6     double tempMin = -1;
7     double strTemp = 0;
8     int indeksTerpilih = 0;
9     for (int i = 0; i < x.Count; i++)
10    {
11        tempMin = -1;
12        for (int j = 0; j < x.Count; j++)
13        {
14            strTemp = Convert.ToDouble(x[j]);
15            if (strTemp != -1)
16            {
17                if (tempMin == -1)
18                {
19                    indeksTerpilih = j;
20                    tempMin = strTemp;
21                }
22                if (Convert.ToDouble(x[j]) < tempMin)
23                {
24                    tempMin = Convert.ToDouble(x[j]);
25                    indeksTerpilih = j;
26                }
27            }
28            tempIndeks.Add(indeksTerpilih);
29            newX.Add(x[indeksTerpilih]);
30            x[indeksTerpilih] = -1;
31        }
32        for (int i = 0; i < x.Count; i++)
33        {
34            newY.Add(y[Convert.ToInt32(tempIndeks[i])]);
35            rtbPembagianDataKFold.Text += "\n" + tempIndeks[i].ToString()
36            + "\tx = " + newX[i].ToString() + "\ty = " + newY[i].ToString();
37        }
38        int tempCek = 0;
39        auc = 0;
40        while (tempCek < x.Count)
41        {
42            double valX1 = 0;

```

```

42     double valY1 = 0;
43     double valX2 = 0;
44     double valY2 = 0;
45     double tempHasil = 0;
46     if (tempCek == 0)
47     {
48         valX1 = Convert.ToDouble(newX[tempCek]);
49         valY1 = Convert.ToDouble(newY[tempCek]);
50         auc += 0.5 * (valX1 * valY1);
51         tempHasil = 0.5 * (valX1 * valY1);
52     }
53     else
54     {
55         if (tempCek == x.Count - 1)
56         {
57             valX1 = Convert.ToDouble(newX[tempCek - 1]);
58             valY1 = Convert.ToDouble(newY[tempCek - 1]);
59             valX2 = Convert.ToDouble(newX[tempCek]);
60             valY2 = Convert.ToDouble(newY[tempCek]);
61             auc += 0.5 * Math.Abs(valX2 - valX1) * Math.Abs(valY2 -
62 valY1);
63             auc += Math.Abs(valX2 - valX1) * valY1;
64         }
65         else
66         {
67             valX1 = Convert.ToDouble(newX[tempCek - 1]);
68             valY1 = Convert.ToDouble(newY[tempCek - 1]);
69             valX2 = Convert.ToDouble(newX[tempCek]);
70             valY2 = Convert.ToDouble(newY[tempCek]);
71             auc += 0.5 * Math.Abs(valX2 - valX1) * Math.Abs(valY2 -
72 valY1);
73             auc += Math.Abs(valX2 - valX1) * valY2;
74             tempCek++;
75         }
76         rtbPembagianDataKFold.Text += "\n\nHasil AUC = " +
77         auc.ToString() + "\n";
78     }

```

Source Code 4.8 Listing Code Proses Hitung Area Under ROC Curve.

Penjelasan Source Code 4.8 adalah:

1. Baris 3-8 dan 37-38 merupakan inisialisasi variabel.
2. Baris 9-30 adalah mengurutkan titik x dan y berdasarkan nilai x.
3. Baris 31-36 adalah menampilkan nilai x dan y yang telah diurutkan.
4. Baris 39-74 menghitung nilai AUC.
5. Baris 75-77 menampilkan hasil AUC.

4.2.2.6 Proses Hitung Nilai *Fitness*

Proses hitung nilai *fitness* adalah proses menghitung jumlah antara nilai hasil AUC atau f_1 dan f_2 atau nilai fitur dikalikan bobot masing masing, dimana bobot untuk nilai AUC adalah 0.85 dan bobot fitur 0.15 untuk tiap-tiap partikel. Nilai

fitness ini yang nantinya digunakan untuk meng-update *pbest* dan *gbest*. *Source code* proses hitung nilai *fitness* ditunjukkan pada *Source Code 4.9*.

```

1  while (tempPart < tempJumPart)
2  {
3      f2 = 1 - (Convert.ToDouble(jumFiturTerpilih[tempFold,
4      tempIndeksPart]) / Convert.ToDouble(jumFitur));
5      f1 = auc;
6      newFitness[tempIndeksPart] = (0.85 * f1) + (0.15 * f2);
7      if (tempMaxFitness < newFitness[tempIndeksPart])
8      {
9          tempMaxFitness = newFitness[tempIndeksPart];
10         tempIndeksTerpilih = tempIndeksPart;
11     }
12     tempPart++;
13     tempIndeksPart++;
14 }

```

Source Code 4.9 Listing Code Proses Hitung Nilai *Fitness*.

Penjelasan *Source Code 4.9* adalah:

1. Baris 3-4 merupakan proses hitung *f2*.
2. Baris 5 ambil nilai AUC pada proses sebelumnya.
3. Baris 6-14 hitung nilai *fitness*.

4.2.2.7 Proses Update Velocity dan Posisi

Proses *update velocity* dan posisi merupakan rangkaian dari implementasi metode TVPSO-FKNN proses ini dilakukan untuk mencari kemungkinan posisi terbaik hingga sebanyak iterasi. *Update velocity* dilakukan dengan menentukan *pbest* dan *gbest* yang tersimpan sebelumnya, selanjutnya hitung nilai *sigmoid* dari hasil *update velocity* dan membangkitkan nilai random antara 0 sampai 1 sebanyak partikel kali jumlah fiturnya untuk proses *update* posisinya. *Update* posisi didapatkan dari membandingkan nilai *sigmoid* dan nilai random yang dibangkitkan jika random lebih kecil dari nilai *sigmoid* maka posisi baru akan bernilai 1, jika sebaliknya akan bernilai 0. *Source code* proses *update velocity* dan posisi ditunjukkan pada *Source Code 4.10*.

```

1  if (x < tMax)
2  {
3      //hitung velocity
4      for (int i = 0; i < jumPartikel; i++)
5      {
6          for (int j = 0; j < jumFitur; j++)
7          {

```

```

8      velocity[i, j] = (w[x] * oldVelocity[i, j]) + ((c1[x] * r1) *
9      (pBest[i, j] - binerFitur[i, j])) + ((c2[x] * r2) *
10     (pBest[tempIndeksTerpilih, j] - binerFitur[i, j]));
11     rtbPembagianDataKFold.Text += "\n Pbest - " + (j + 1).ToString()
12     + " = " + binerFitur[i, j].ToString() + "\t" + "\tPosisi - " + (j +
13     1).ToString() + " = " + tempSimpanBiner[i, j].ToString() + "\n";
14     sigmoid[i, j] = (1 / (1 + Math.Exp(-velocity[i, j])));
15     rtbPembagianDataKFold.Text += "\nVelocity - " + (j +
16     1).ToString() + " = " + velocity[i, j].ToString();
17     }}
18     //menampilkan nilai sigmoid
19     for (int i = 0; i < jumPartikel; i++)
20     {
21         for (int j = 0; j < jumFitur; j++)
22         {
23             rtbPembagianDataKFold.Text += "\nSigmoid - " + (j +
24             1).ToString() + " = " + sigmoid[i, j].ToString();
25             oldVelocity[i, j] = velocity[i, j];
26         }}
27     // Random number
28     Random r = new Random();
29     for (int i = 0; i < jumPartikel; i++)
30     {
31         for (int j = 0; j < jumFitur; j++)
32         {
33             randomIter[i, j] = r.NextDouble();
34         }}
35     // New Posisi
36     for (int i = 0; i < jumPartikel; i++)
37     {
38         tempFiturTerpilih = new ArrayList();
39         for (int j = 0; j < jumFitur; j++)
40         {
41             if (randomIter[i, j] < sigmoid[i, j])
42             {
43                 newPosisi[i, j] = 1;
44             }
45             else
46             {
47                 newPosisi[i, j] = 0;
48             }
49             if (newPosisi[i, j] == 1)
50             {
51                 tempFiturTerpilih.Add(i);
52             }}

```

Source Code 4.10 Listing Code Proses Update Velocity dan Posisi.

Penjelasan Source Code 4.10 adalah:

1. Baris 4-6 merupakan inisialisasi untuk *looping*.
2. Baris 8-10 untuk hitung *velocity* baru
3. Baris 11-13 menampilkan *pbest* dan posisi.
4. Baris 14 adalah untuk hitung nilai *sigmoid*.
5. Baris 15-16 digunakan untuk menampilkan *velocity* baru.

6. Baris 23-24 untuk menampilkan nilai *sigmoid*.
7. Baris 25 untuk menyimpan *velocity* baru
8. Baris 28-34 untuk membuat angka random sebanyak jumlah partikel kali jumlah fitur.
9. Baris 36-52 untuk mengupdate posisi baru

4.2.2.8 Proses Update Pbest dan Gbest

Proses *update pbest* dan *gbest* merupakan proses meng-update posisi partikel terbaik tiap-tiap partikel berdasarkan pada nilai *pfit* tertinggi. Sedangkan *update gbest* adalah mencari partikel terbaik dalam populasi dengan mencari nilai *gfit* tertinggi. *Source code* proses *update pbest* dan *gbest* ditunjukkan pada *Source Code 4.11*.

```

1  while (tempPart < tempJumPart)
2  {
3      if (newFitness[tempIndeksPart] >=
4      bestFitnessIterasi[tempIndeksPart])
5      {
6          for (int i = 0; i < jumFitur; i++)
7          {
8              pBest[tempIndeksPart, i] = binerFitur[tempIndeksPart, i];
9          }
10         bestFitnessIterasi[tempIndeksPart] = newFitness[tempIndeksPart];
11     }
12     if (tempMaxFitness < bestFitnessIterasi[tempIndeksPart])
13     {
14         tempMaxFitness = bestFitnessIterasi[tempIndeksPart];
15         tempIndeksTerpilih = tempIndeksPart;
16     }
17     tempPart++;
18     tempIndeksPart++;
19 }
20 tempBestK = valK[(tempPart-jumPartikel) + tempIndeksTerpilih];
21 tempBestM = valM[(tempPart-jumPartikel) + tempIndeksTerpilih];
22 rtbPembagianDataKFold.Text += "\nBest Fitness Indeks = " +
23 tempIndeksTerpilih.ToString() + "\t";
24 rtbPembagianDataKFold.Text += "\nBest Fitness (Gfit) = " +
25 tempMaxFitness.ToString() + "\n";
26 for (int j = 0; j < jumFitur; j++)
27 {
28     rtbPembagianDataKFold.Text += "\nG Best - " + (j + 1).ToString() +
29     " = " + pBest[tempIndeksTerpilih, j].ToString();
30 }

```

Source Code 4.11 Listing Code Proses Update Pbest dan Gbest.

Penjelasan *Source Code* 4.11 adalah:

1. Baris 1-19 merupakan proses *update pbest* dan *gbest*.
2. Baris 20-21 merupakan proses menyimpan nilai *m* dan *k* dari *gbest*.
3. Baris 22-23 merupakan proses menampilkan *index best fitness (index fitness gbest)*.
4. Baris 24-25 adalah untuk menampilkan nilai *best fitness (gfit)*.
5. Baris 26-30 untuk menampilkan *gbest*.

4.2.3 Proses Uji dengan *K-fold Cross Validation*

Proses uji dengan *K-fold cross validation* merupakan proses menguji hasil TVPSO-FKNN yaitu *gbest* masing-masing *fold* untuk di-*train* FKNN kembali dengan menggunakan *K-fold* sesuai *input-an K* diawal pada proses membangkitkan partikel. *Source code* proses uji dengan *K-fold cross validation* ditunjukkan pada *Source Code* 4.12.

```

1 private void trainUjiHasil_Click(object sender, EventArgs e)
2 {
3     rtbUjiTvpsofknn.Text = "";
4     int tempFold = 0;
5     int tempFoldData = 0;
6     avgAkurasi = new double[kFold];
7     while (tempFold < kFold)
8     {
9         rtbUjiTvpsofknn.Text += "Fold # " + (tempFold + 1).ToString();
10        tempFoldData = 0;
11        double tempAkurasi = 0;
12        while (tempFoldData < kFold)
13        {
14            tempFiturTerpilih = new ArrayList();
15            tempDataTesting = new ArrayList();
16            tempDataTraining = new ArrayList();
17            int tempDataTerpilih = 0;
18            //TAMPIL
19            rtbUjiTvpsofknn.Text += "\n\nFitur Terpilih K-fold " +
20            (tempFoldData + 1).ToString() + "\n";
21            //Pemilihan Fitur
22            for (int i = 0; i < jumFitur; i++)
23            {
24                if (bestPartikel[tempFold, i] == 1)
25                {
26                    tempFiturTerpilih.Add(i);
27                    rtbUjiTvpsofknn.Text += (i + 1).ToString() + "-";
28                }
29            }
30            for (int i = 0; i < dataPembagianKFold.GetLength(0); i++)

```

```

30     {
31         for (int j = 0; j < dataPembagianKFold[i].Length; j++)
32         {
33             if (i == tempFoldData)
34             {
35                 tempDataTerpilih = dataPembagianKFold[i][j];
36                 tempDataTesting.Add(tempDataTerpilih);
37             }
38             else
39             {
40                 tempDataTerpilih = dataPembagianKFold[i][j];
41                 tempDataTraining.Add(tempDataTerpilih);
42             }
43             // Train FKNN dengan Kfold sampai mendapatkan nilai FPR code sama
44             // dengan pada proses TVPSO-FKNN hanya saja ada tambahan code proses
45             // penghitungan accuracy (acc)
46             acc = Convert.ToDouble(tp + tn) / Convert.ToDouble(tp + tn + fp +
47             fn);
48             tempAkurasi += acc;
49             rtbUjiTvpsoknn.Text += "\n\nHasil KFold - "
50             +(tempFold+1).ToString() + " Part - " + (tempFoldData+1).ToString();
51             rtbUjiTvpsoknn.Text += "K fold - " + (tempFold + 1).ToString();
52             rtbUjiTvpsoknn.Text += "\ntp : " + tp.ToString() + "\n";
53             rtbUjiTvpsoknn.Text += "tn : " + tn.ToString() + "\n";
54             rtbUjiTvpsoknn.Text += "fp : " + fp.ToString() + "\n";
55             rtbUjiTvpsoknn.Text += "fn : " + fn.ToString() + "\n";
56             rtbUjiTvpsoknn.Text += "tp Rate : " + tpRate.ToString() + "\n";
57             rtbUjiTvpsoknn.Text += "fp Rate : " + fpr.ToString() + "\n";
58             rtbUjiTvpsoknn.Text += "acc : " + acc.ToString() + "\n";
59             tempFoldData++;
60         }
61         tempAkurasi = (tempAkurasi/Convert.ToDouble(kFold)) * 100;
62         avgAkurasi[tempFold] = tempAkurasi;
63         tempFold++;
64     }
65     //Tampilkan Rata-Rata Akurasi
66     rtbUjiTvpsoknn.Text += "\n\nRata-Rata Akurasi dengan " +
67     (kFold).ToString() + " -Fold CV ";
68     for (int i = 0; i < kFold; i++)
69     {
70         rtbUjiTvpsoknn.Text += "Fold # " + (i + 1).ToString() + " = " +
71         avgAkurasi[i].ToString()+"%\n";
72     }

```

Source Code 4.12 Listing Code Proses Uji Hasil FKNN dan Hitung Akurasi.

Penjelasan Source Code 4.12 adalah:

1. Baris 3-6 merupakan proses inisialisasi variabel.
2. Baris 7-42 adalah untuk menampilkan fitur terpilih untuk tiap-tiap *K-fold*.

3. Baris 46-47 adalah proses hitung nilai *accuracy* (ACC).
4. Baris 48-60 adalah proses menampilkan hasil.
5. Baris 61-63 adalah proses menyimpan nilai akurasi.
6. Baris 66-72 adalah untuk menampilkan hasil nilai akurasi tiap-tiap *fold*.

4.2.4 Proses Menampilkan Hasil Akurasi dari Uji dengan *K-fold Cross Validation*

Proses menampilkan hasil akurasi dari uji dengan *K-fold cross validation* adalah proses menampilkan hasil akurasi ke dalam *datagridview*. *Source code* proses menampilkan hasil akurasi ini ditunjukkan pada *Source Code 4.13*.

```

1  tabControl1.SelectedIndex = 4;
2  dataGridView22.Rows.Clear();
3  int jumF = 6;
4  dataGridView22.ColumnCount = jumF;
5  for (int i = 0; i < kFold; i++)
6  {
7      row = new string[jumF];
8      row[0] = (i + 1).ToString();
9      row[1] = "Fold #" + (i + 1).ToString();
10     row[2] = bestValM[i].ToString();
11     row[3] = bestValK[i].ToString();
12     for (int j = 0; j < jumFitur; j++)
13     {
14         if (bestPartikel[i, j] == 1)
15         {
16             tempFiturTerpilih.Add(j);
17             row[4] += (j + 1).ToString() + "-";
18         }
19         row[5] = Math.Round(avgAkurasi[i, 4].ToString()+"%";
20             dataGridView22.Rows.Add(row);
21     }

```

Source Code 4.13 Listing Code Proses Menampilkan Hasil Akurasi.

Penjelasan *Source Code 4.13* adalah:

1. Baris 5 merupakan awal dari proses *looping*.
2. Baris 7 inialisasi baris atau *row* dari *datagridview*.
3. Baris 8-20 merupakan proses menampilkan hasil pada *datagridview* dengan menempatkan nilai pada *row* berdasarkan *index* kolom.

4.3 Implementasi Interface

Pada sub bab implementasi *interface* merupakan hasil dari perancangan *user interface* pada bab sebelumnya. Adapun tampilan *interface*-nya terdiri dari *form* normalisasi dikalikan bobot atribut, *form* membangkitkan partikel, *form* proses TVPSO-FKNN, *form* uji hasil TVPSO-FKNN, dan *form* hasil akurasi.

4.3.1 Form Hasil Normalisasi Kali Bobot Atribut atau Fitur

Pada *form* hasil normalisasi kali bobot atribut atau fitur merupakan *interface* dimana proses perhitungan penerapan metode proses normalisasi dikalikan bobot atribut atau fitur. Adapun tahapan prosesnya adalah sebagai berikut:

1. Pertama adalah pilih *file* yang akan di-load yaitu *file excel* yang berformat *.xls* yang berisi *sheet data* dan *sheet bobot* dan *inputkan* nama dari *sheet data* dan *sheet bobot* dan klik pada *button load file data* dan *bobot*.
2. Selanjutnya hasil *load* akan muncul pada *datagridview* data asli dan bobot fitur.
3. Kemudian klik *button* proses normalisasi untuk menjalankan proses normalisasi hingga hasilnya muncul pada *datagridview* hasil normalisasi.
4. Proses terakhir pada *form* tab ini yaitu klik *button* normalisasi kali bobot untuk menjalankan proses normalisasi kali bobot fitur hingga hasilnya muncul pada *datagridview* hasil normalisasi kali bobot.
5. *Button* selanjutnya merupakan *button link* untuk menuju proses selanjutnya yaitu tab membangkitkan partikel.

Adapun *interface form* hasil normalisasi kali bobot atribut atau fitur ditunjukkan pada Gambar 4.1.

IMPLEMENTASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR BERDASARKAN PARTICLE SWARM OPTIMIZATION UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA (Studi Kasus Kelurahan Jugo Kecamatan Kesamben Kabupaten Blitar)

Normalisasi x Bobot | Membangkitkan Partikel | Proses TVPSO-FKNN | Uji Hasil TVPSO-FKNN | Hasil Akurasi

NORMALISASI DATA DIKALIKAN BOBOT ATRIBUT

Ambil Data

Ambil Data: F:\SAP\thvps-fknn Random Data - Copy\121data.xls Pilih File

Nama Sheet Data: 121data Load File Data dan Bobot

Nama Sheet Bobot: bobot

Hasil Load Data

Data Asli

No	KK	F1	F2
1	1	1	2
2	2	2	4
3	3	0	2
4	4	1	2

Bobot Fitur

Fitur	Pendidikan (F1)	Pekerjaan (F2)
Bobot	5	20

Proses Normalisasi Normalisasi x Bobot

Hasil Normalisasi dikalikan Bobot Atribut/Fitur

Hasil Normalisasi dengan Sigmoid Biner Interval [0.1, 0.9]

No	KK	F1	F2	F3	F4	F5	F6
1	1	0.5	0.5	0.1	0.2561	0.34	0.9
2	2	0.9	0.9	0.1	0.3732	0.42	0.9
3	3	0.1	0.5	0.1	0.2171	0.34	0.5
4	4	0.5	0.5	0.6333	0.178	0.26	0.9

Hasil Normalisasi x Bobot Atribut atau Fitur

No	KK	F1	F2	F3	F4	F5	F6
1	1	0.0179	0.0714	0.0036	0.0366	0.0121	0.0
2	2	0.0321	0.1286	0.0036	0.0533	0.015	0.0
3	3	0.0036	0.0714	0.0036	0.031	0.0121	0.0
4	4	0.0179	0.0714	0.0226	0.0254	0.0093	0.0

Selanjutnya>>>

Gambar 4.1 Tampilan Utama Tab 1 *Form* Hasil Hitung Normalisasi Kali Bobot Atribut atau Fitur

4.3.2 *Form* Hasil Membangkitkan Partikel

Pada *form* hasil membangkitkan partikel merupakan *interface* dari proses membangkitkan partikel dimana pada proses ini tahapannya adalah sebagai berikut:

1. *Input*-kan nilai jumlah partikel, nilai K , nilai interval m , interval k , dan jumlah iterasi pada *textbox* yang telah disediakan.
2. Kemudian klik *button* proses membangkitkan partikel untuk menjalankan proses pembangkitan parameter hingga hasilnya muncul pada *datagridview* hasil inisialisasi *encode* partikel.
3. *Button* proses TVPSO-FKNN merupakan *button* selain digunakan untuk menuju ke tab selanjutnya yaitu tab proses TVPSO-FKNN juga untuk menjalankan proses TVPSO-FKNN.

Adapun *interface form* hasil membangkitkan partikel ditunjukkan pada Gambar 4.2.

IMPLEMENTASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR BERDASARKAN PARTICLE SWARM OPTIMIZATION
UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA
(Studi Kasus Kelurahan Jugo Kecamatan Kesamben Kabupaten Blitar)

Normalisasi x Bobot | Membangkitkan Partikel | Proses TVPSO-FKNN | Uji Hasil TVPSO-FKNN | Hasil Akurasi

PROSES MEMBANGKITKAN PARTIKEL

Input Jumlah Partikel: 5
Input Nilai K: 8
Input Interval m: 1 - 4
Input Interval k: 1 - 10
Input Jumlah Iterasi: 10

Proses Membangkitkan Partikel

INISIALISASI ENCODE PARTIKEL

Partikel	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13
Fold 1															
P1	3.6142	6	1	1	1	1	0	1	0	0	0	1	1	0	0
P2	1.9802	1	1	1	0	0	0	1	1	1	1	0	1	1	0
P3	3.1185	3	1	0	0	0	1	1	1	0	0	0	0	0	1
P4	2.1738	5	1	1	0	1	0	0	1	1	0	0	0	1	1
P5	3.5245	1	1	0	0	0	1	1	1	1	1	1	1	1	0
Fold2															
P1	2.4797	9	1	0	0	1	0	1	0	1	0	1	0	1	1
P2	1.2128	4	0	1	1	1	1	0	1	0	1	0	1	1	0
P3	2.6853	3	0	1	0	1	0	0	0	1	1	1	1	0	0
P4	3.1188	2	0	1	0	1	1	0	1	0	1	0	1	0	1

Proses TVPSO-FKNN

Gambar 4.2 Tampilan Tab 2 *Form Input* Parameter dan Hasil Membangkitkan Partikel

4.3.3 *Form Hasil Proses TVPSO-FKNN*

Pada *form* hasil proses TVPSO-FKNN merupakan *interface* untuk menampilkan proses sekaligus hasil dari proses TVPSO-FKNN. Proses tersebut ditampilkan pada 1 *richtextbox* saja. Pada hasil proses TVPSO-FKNN merupakan serangkaian proses mulai dari proses pembagian data dengan *K-Fold CV*, kemudian proses *train* FKNN, menghitung AUC, Menghitung nilai *fitness*, proses TVPSO hingga mendapatkan solusi yaitu partikel terbaik yang berisi parameter *m*, *k* dan fitur terpilih dengan *fitness* tertinggi (*gbest*). Kemudian *button* selanjutnya selain digunakan untuk menuju tab selanjutnya yaitu juga merupakan tab uji hasil TVPSO-FKNN untuk menjalankan proses penyimpanan dan menampilkan hasil dari proses TVPSO-FKNN pada *datagridview* hasil TVPSO-FKNN pada tab selanjutnya. Adapun *interface form* hasil proses TVPSO-FKNN ditunjukkan pada Gambar 4.3.

**IMPLEMENTASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR BERDASARKAN PARTICLE SWARM OPTIMIZATION
UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA
(Studi Kasus Kelurahan Jugo Kecamatan Kesamben Kabupaten Blitar)**

Normalisasi x Bobot Membangkitkan Partikel **Proses TVPSO-FKNN** Uji Hasil TVPSO-FKNN Hasil Akurasi

PROSES TVPSO-FKNN

Pembagian Data K-Fold CV -> Proses Train FKNN -> Proses AUC->Hitung Nilai Fitness->Proses TVPSO->Hasil TVPSO-FKNN

Data yang terpilih untuk K-Fold #1
18-74-6-21-106-37-60-48-9-101-23-107-27-100-113-40-
Data yang terpilih untuk K-Fold #2
36-77-2-38-57-51-33-86-93-16-111-92-1-75-104-
Data yang terpilih untuk K-Fold #3
112-89-72-42-80-19-91-31-105-59-66-68-71-43-94-
Data yang terpilih untuk K-Fold #4
20-87-62-24-5-119-67-28-90-109-50-30-69-82-41-
Data yang terpilih untuk K-Fold #5
99-56-103-3-49-13-84-58-108-12-79-44-34-110-115-
Data yang terpilih untuk K-Fold #6
98-83-61-116-10-7-14-85-64-81-39-70-117-54-35-
Data yang terpilih untuk K-Fold #7
114-22-53-52-65-15-120-88-97-78-29-45-8-25-47-
Data yang terpilih untuk K-Fold #8
95-17-32-73-11-59-118-4-96-46-102-76-121-26-63-
=====

INISIALISASI PARAMETER
=====

Fold = #1
0 x = 0.066666666666667 y = 0.875
1 x = 0.103448275862069 y = 0.8125
3 x = 0.142857142857143 y = 0.75
2 x = 0.185185185185185 y = 0.6875

Selanjutnya>>

Gambar 4.3 Tampilan Tab 3 *Form* Hasil Proses TVPSO-FKNN

4.3.4 *Form* Hasil Proses Uji Hasil TVPSO-FKNN

Pada *form* hasil proses uji hasil TVPSO-FKNN merupakan *interface* untuk melakukan proses uji dari hasil proses TVPSO-FKNN, adapun tahapan pada proses ini adalah sebagai berikut:

1. Terdapat *datagridview* hasil dari proses sebelumnya yaitu proses TVPSO-FKNN.
2. Hasil tersebut yang akan digunakan pada proses uji ini dengan menekan *button train* FKNN maka proses uji hasil tersebut dijalankan hingga mendapatkan hasil yang ditampilkan pada 1 *richtextbox* uji-TVPSO-FKNN.
3. *Button* lanjut hasil merupakan *button* yang digunakan untuk menuju pada tab selanjutnya yaitu tab hasil akurasi, selain itu juga untuk menjalankan proses menampilkan hasil akurasi ke dalam *datagridview* pada tab hasil akurasi.

Adapun *interface form* hasil proses uji hasil TVPSO-FKNN ditunjukkan pada Gambar 4.4.

IMPLEMENTASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR BERDASARKAN PARTICLE SWARM OPTIMIZATION UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA (Studi Kasus Kelurahan Jugo Kecamatan Kesamben Kabupaten Blitar)

Normalisasi x Bobot | Membangkitkan Partikel | Proses TVPSO-FKNN | **Uji Hasil TVPSO-FKNN** | Hasil Akurasi

PROSES UJI HASIL TVPSO-FKNN

HASIL TVPSO-FKNN

	Fold	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
Fold#1	3.1185	3	1	1	0	1	0	0	1	1	0	0	0	0
Fold#2	2.4797	9	1	0	0	1	1	0	1	1	0	1	0	0
Fold#3	1.6595	6	0	0	1	1	0	0	0	0	1	0	1	0
Fold#4	2.1043	1	0	1	0	1	0	1	0	0	1	0	0	0

Uji TVPSO-FKNN

Fold # 1

Fitur Terpilih K-fold 1
1-2-4-7-8-
Data Testing Terpilih =
18-74-6-21-106-37-60-48-9-101-23-107-27-100-113-40-
Data Training Terpilih =
36-77-2-38-57-51-33-86-93-16-111-92-1-75-104-112-89-72-42-80-19-91-31-105-59-66-68-71-43-94-20-87-62-24-5-119-67-28-90-109-50-30-69-82-41-99-56-103-3-49-13-84-58-108-12-79-44-34-110-115-98-83-61-116-10-7-14-85-64-81-39-70-117-54-35-114-22-53-52-65-15-120-88-97-78-29-45-8-25-47-95-17-32-73-11-85-118-4-96-46-102-76-121-26-63-

Diketahui Nilai :
Data Training = 105
n1 = 18
n2 = 47
n3 = 40

Train FKNN

Lanjut Hasil

Gambar 4.4 Tampilan Tab 4 *Form* Hasil Proses Uji Hasil TVPSO-FKNN Hitung Akurasi

4.3.5 *Form* Hasil Akurasi

Pada *form* hasil akurasi merupakan *interface* terakhir yaitu menampilkan hasil akurasi dari pengujian sebelumnya yang ditampilkan oleh *datagridview* hasil akurasi. Adapun *interface form* hasil akurasi ditunjukkan pada Gambar 4.5.

IMPLEMENTASI METODE ADAPTIVE FUZZY K-NEAREST NEIGHBOR BERDASARKAN PARTICLE SWARM OPTIMIZATION UNTUK KLASIFIKASI STATUS SOSIAL EKONOMI KELUARGA (Studi Kasus Kelurahan Jugo Kecamatan Kesamben Kabupaten Blitar)

Normalisasi x Bobot | Membangkitkan Partikel | Proses TVPSO-FKNN | Uji Hasil TVPSO-FKNN | **Hasil Akurasi**

Hasil rata-rata akurasi dari tiap-tiap Partikel Terbaik (Gbest)

No	Gbest	Nilai m	Nilai k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	3.1185	3	1-2-4-7-8-	83.8576%
2	Fold #2	2.4797	9	1-4-5-7-8-10-	83.8104%
3	Fold #3	1.6595	6	3-4-9-11-	84.8308%
4	Fold #4	2.1043	1	2-4-6-9-	83.35%
5	Fold #5	3.808	6	3-5-7-	77.0481%
6	Fold #6	3.5095	3	2-3-4-	84.4637%
7	Fold #7	2.928	6	7-8-14-	62.1762%
8	Fold #8	1.1112	4	3-4-5-11-12-	83.6208%

Gambar 4.5 Tampilan Tab 5 *Form* Hasil Akurasi

BAB V

PENGUJIAN DAN ANALISIS

Pada bab pengujian dan analisis akan dibahas mengenai analisis dan pengujian sistem yang telah diimplementasikan pada tahap sebelumnya. Analisis pengujian meliputi pengaruh interval nilai m dan k yang digunakan, pengaruh jumlah partikel yang digunakan, pengaruh nilai K yang akan digunakan pada *K-fold cross validation* dengan 121 data yang diambil secara random, dan pengaruh banyaknya iterasi yang akan dilakukan.

5.1 Pengujian

Pada pengujian ini digunakan data sebanyak 121 data keluarga yang terdiri dari kelas atas sebanyak 24 data, kelas menengah 52 data, dan kelas bawah sebanyak 45 data. Dimana pada pengujian ini data diambil secara random dengan tahapan mengelompokkan ke masing-masing kelas kemudian diambil secara random berturut-turut mengambil 1 persatu dari masing-masing kelas sebanyak nilai K hal ini dilakukan agar penyebaran data tiap *K-fold* dapat merata. Pengujian ini dilakukan sebanyak 5 model pengujian yaitu:

1. Pengujian untuk mengetahui pengaruh interval nilai m yang digunakan.
2. Pengujian untuk mengetahui pengaruh interval k yang digunakan.
3. Pengujian untuk mengetahui pengaruh jumlah partikel yang digunakan.
4. Pengujian untuk mengetahui pengaruh banyaknya iterasi yang akan dilakukan.
5. Pengujian untuk mengetahui pengaruh nilai K pada *K-fold cross validation* dengan 121 data yang diambil secara random.
6. Pengujian untuk mengukur konvergensi pada PSO.

5.2 Hasil Pengujian

Terdapat 5 hasil pengujian yaitu hasil pengujian pengaruh interval nilai m dan k yang digunakan, pengujian pengaruh jumlah partikel yang digunakan, pengujian pengaruh nilai K pada *K-fold cross validation* dengan 121 data yang

diambil secara random, dan yang terakhir adalah pengujian pengaruh banyaknya iterasi yang akan dilakukan.

5.2.1 Pengujian Pengaruh Interval Nilai m Terhadap Rata-rata Akurasi

Pengujian interval nilai m ini dilakukan untuk mengetahui pada interval ke berapa nilai m dapat menghasilkan rata-rata akurasi tertinggi. Pada pengujian ini nilai *default* yang digunakan yaitu menggunakan nilai interval k [1, 10] sedangkan untuk nilai K adalah 5, jumlah partikel adalah 8, dan banyak iterasinya adalah 100. Pada pengujian ini terdapat 5 kali pengujian dengan interval nilai m yang berbeda. Adapun hasilnya dapat dilihat pada Tabel 5.1.

Tabel 5.1 Pengujian Pengaruh Nilai m terhadap Rata-rata Akurasi

No	Interval m	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	[1, 2]	1.2208	7	4-11	89.50%
2	[2, 4]	2.5858	6	4-5-11-14	88.78%
3	[4, 6]	5.7911	5	4-11	89.39%
4	[6, 8]	6.8878	8	2-4-8	85.59%
5	[8, 10]	8.0911	8	4-7	84.51%

Pada Tabel 5.1 dapat dilihat bahwa hasil rata-rata akurasi tertinggi adalah pada interval 1 sampai 2, dimana dengan nilai $m = 1.2208$ dan $k = 7$ serta fitur terpilihnya 4 dan 11 menghasilkan nilai rata-rata akurasi sebesar 89.50%. Hasil pengujian secara detail dapat dilihat pada Lampiran 9.

5.2.2 Pengujian Pengaruh Interval Nilai k Terhadap Rata-rata Akurasi

Pengujian interval nilai k ini dilakukan untuk mengetahui pada interval ke berapa nilai k dapat menghasilkan rata-rata akurasi tertinggi. Pada pengujian ini nilai *default* yang digunakan yaitu menggunakan nilai interval m dari pengujian m yang rata-rata akurasinya tertinggi yaitu interval [1, 2], sedangkan untuk nilai K adalah 5, jumlah partikel adalah 8, dan banyak iterasinya 100. Pada pengujian ini terdapat 5 kali pengujian dengan interval nilai k yang berbeda. Adapun hasilnya dapat dilihat pada Tabel 5.2.

Tabel 5.2 Pengaruh Nilai k Terhadap Rata-rata Akurasi

No	Interval k	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	[1, 10]	1.8302	4	4-11	90.17%
2	[11, 20]	1.9225	11	2-4-12	85.76%
3	[21, 40]	1.2541	26	3-4-9-13	85.05%
4	[41, 60]	1.8766	49	2-4	84.47%
5	[61, 80]	1.8091	78	4-5-9-11	83.05%

Pada Tabel 5.2 dapat dilihat bahwa hasil rata-rata akurasi tertinggi adalah pada interval 1 sampai 10, dimana dengan nilai $m = 1.8302$ dan $k = 4$ serta fitur terpilihnya 4 dan 11 menghasilkan nilai rata-rata akurasi sebesar 90.17%. Hasil pengujian secara detail dapat dilihat pada Lampiran 10.

5.2.3 Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi

Pengujian jumlah partikel ini dilakukan untuk mengetahui pada jumlah partikel berapa dapat menghasilkan rata-rata akurasi tertinggi. Pada pengujian ini nilai *default* yang digunakan yaitu untuk nilai interval k dan nilai interval m adalah nilai yang memiliki rata-rata akurasi tertinggi pada pengujian m dan k yaitu [1, 2] dan [1, 10], untuk nilai K adalah 5, serta banyak iterasinya 100. Adapun hasilnya dapat dilihat pada Tabel 5.3.

Tabel 5.3 Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi

No	Jumlah Partikel	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	4	1.4231	1	4-7-8-11	89.46%
2	5	1.3364	9	4-5-11-14	88.27%
3	6	1.7833	10	4-8-11-12	87.07%
4	8	1.0892	2	2-4-12	85.82%
5	10	1.2518	3	1-4-11	84.99%

Pada Tabel 5.3 dapat dilihat bahwa hasil rata-rata akurasi tertinggi adalah pada jumlah partikel = 4 dimana dengan nilai $m = 1.4231$ dan $k = 1$ serta fitur terpilihnya 4, 7, 8, dan 11 menghasilkan nilai rata-rata akurasi sebesar 89.46%. Hasil pengujian secara detail dapat dilihat pada Lampiran 11.

5.2.4 Pengujian Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi

Pengujian jumlah iterasi ini dilakukan untuk mengetahui pada jumlah iterasi berapa dapat menghasilkan rata-rata akurasi tertinggi. Pada pengujian ini nilai *default* yang digunakan yaitu untuk nilai interval m , nilai interval k , dan jumlah partikel adalah nilai m , k , dan jumlah partikel yang memiliki rata-rata akurasi tertinggi pada pengujian sebelumnya yaitu [1, 2], [1, 10], dan 4. Nilai K yang digunakan adalah 5. Pada pengujian ini terdapat 5 kali pengujian dengan jumlah iterasi yang berbeda. Adapun hasilnya dapat dilihat pada Tabel 5.4.

Tabel 5.4 Pengaruh Jumlah Iterasi Terhadap Akurasi

No	Banyak Iterasi	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	10	1.4373	7	2-5-6-7-9-13	81.24%
2	50	1.8964	3	1-4-11	84.54%
3	100	1.8753	2	2-4-6-10	86.30%
4	150	1.6741	7	4-7-8-11	89.53%
5	250	1.1843	4	4-8-9-11-14	91.31%

Pada Tabel 5.4 dapat dilihat bahwa hasil rata-rata akurasi tertinggi adalah pada jumlah iterasi = 250 dimana dengan nilai $m = 1.1843$ dan $k = 4$ serta fitur terpilihnya 4, 8, 9, 11, dan 14 menghasilkan nilai rata-rata akurasi sebesar 91.31%. Hasil pengujian secara detail dapat dilihat pada Lampiran 12.

5.2.5 Pengujian Pengaruh Nilai K Terhadap Rata-rata Akurasi

Pengujian nilai K ini dilakukan untuk mengetahui pada nilai K berapa dapat menghasilkan rata-rata akurasi tertinggi. Pada pengujian ini nilai *default* yang digunakan yaitu untuk nilai interval m , nilai interval k , jumlah partikel, dan banyaknya iterasi adalah nilai yang memiliki rata-rata akurasi tertinggi pada

pengujian m , k , jumlah partikel, dan banyak iterasi yaitu [1, 2], [1, 10], 4, dan 250. Pada pengujian ini terdapat 5 kali pengujian dengan nilai K yang berbeda. Adapun hasilnya dapat dilihat pada Tabel 5.5.

Tabel 5.5 Pengaruh Nilai K Terhadap Rata-rata Akurasi

No	Nilai K pada K-Fold CV	m	k	Fitur Terpilih	Rata-rata Akurasi Tertinggi (%)
1	4	1.8451	9	2-4-6-7	85.50%
2	5	1.1372	6	4-7	85.15%
3	6	1.1213	5	2-4-7-10	86.38%
4	8	1.3005	2	4-5-8-9-11	89.49%
5	10	1.5315	7	4-11	84.94%

Pada Tabel 5.5 dapat dilihat bahwa hasil rata-rata akurasi tertinggi adalah pada nilai $K = 8$ dimana dengan nilai $m = 1.3005$ dan $k = 2$ serta fitur terpilihnya 4, 5, 8, 9, dan 11 menghasilkan nilai rata-rata akurasi sebesar 89.49%. Hasil pengujian secara detail dapat dilihat pada Lampiran 13.

5.2.6 Pengujian untuk Mengetahui Konvergensi pada TVPSO

Pengujian untuk mengukur konvergensi pada TVPSO digunakan untuk mengetahui pada iterasi ke berapa pencarian solusi pada TVPSO berada pada titik konvergensi. Nilai *default* yang digunakan yaitu untuk nilai interval k , nilai interval m , jumlah partikel, banyak iterasi, dan nilai K adalah nilai yang memiliki rata-rata akurasi tertinggi pada pengujian m , k , jumlah partikel, banyak iterasi, dan pengujian K yaitu [1, 2], [1, 10], 4, 250, dan 8. Adapun hasilnya dapat dilihat pada Tabel 5.6.

Tabel 5.6 Hasil Mengukur Konvergensi pada TVPSO

No	Fold	Jumlah Iterasi	Nilai <i>Fitness</i> Terbaik
1	#1	103	0.884997
2	#2	193	0.932549
3	#3	28	0.870119
4	#4	122	0.882632
5	#5	29	0.844886
6	#6	227	0.812743

7	#7	56	0.91119
8	#8	26	0.529179

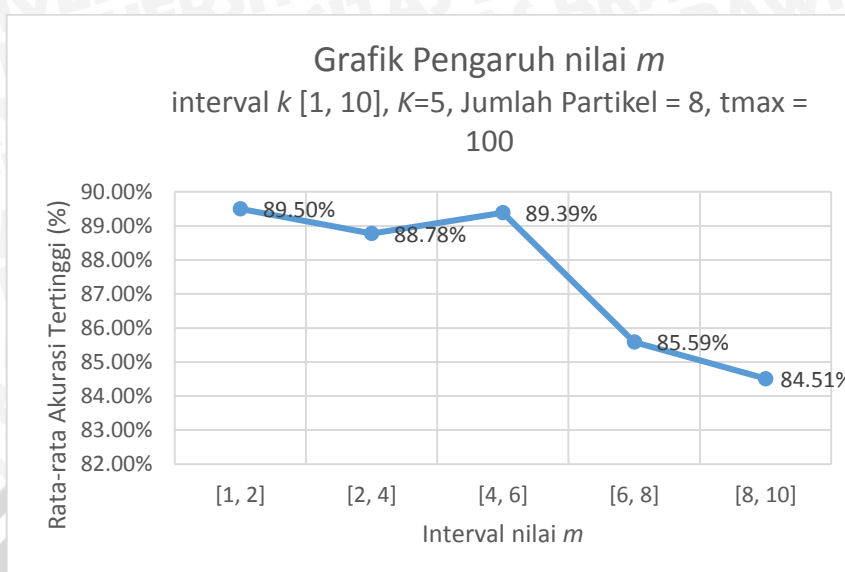
Pada Tabel 5.6 dapat dilihat bahwa nilai *fitness* terbaik dari partikel terbaik sebanyak 8 yaitu 0.9325 konvergensi pada iterasi ke-193 dan pada *fold* #2, selain itu pada *fold* #7 dengan nilai *fitness* terbaiknya memiliki selisih sedikit dengan *fold* #2 yaitu 0.9112 yang mengalami konvergensi pada iterasi ke-56. Hasil pengujian secara detail dapat dilihat pada Lampiran 15.

5.3 Analisis Hasil Pengujian

Analisis hasil pengujian merupakan analisis dari hasil pengujian sebelumnya. Analisis yang dilakukan sebanyak pengujian yang dilakukan yaitu analisis pengujian pengaruh interval nilai m dan k yang digunakan, analisis pengujian pengaruh jumlah partikel yang digunakan, analisis pengujian pengaruh banyaknya iterasi yang akan dilakukan, analisis pengujian pengaruh nilai K pada *K-fold cross validation* dengan 121 data yang diambil secara random, dan yang terakhir adalah analisis pengujian dengan mengukur konvergensi pada PSO.

5.3.1 Analisis Hasil Pengujian Pengaruh Nilai Interval m Terhadap Rata-rata Akurasi

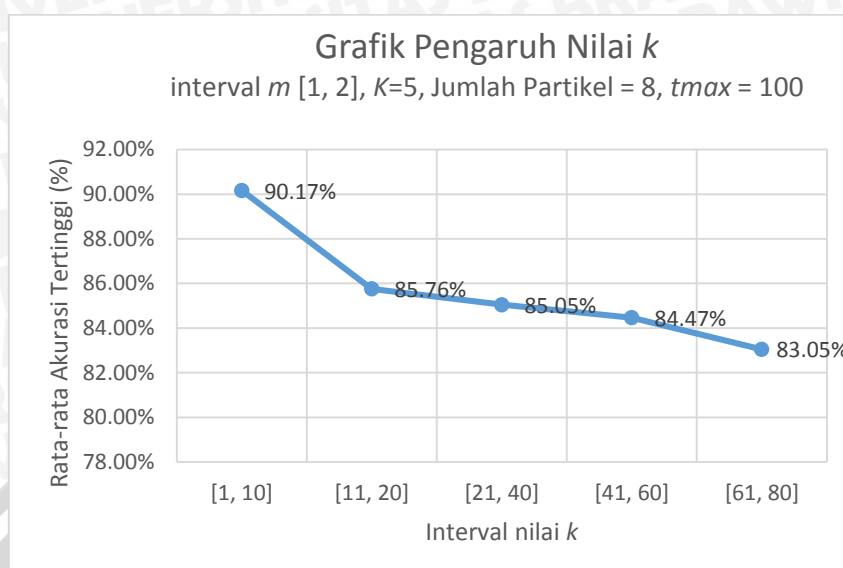
Berdasarkan pada pengujian terhadap pengaruh nilai interval m yang ditunjukkan pada Gambar 5.1, terlihat bahwa nilai interval $[1, 2]$ memberikan nilai rata-rata akurasi tertinggi yaitu 89.50% dengan nilai m terpilih adalah 1.2208, nilai k terpilih adalah 7, dan fitur terpilihnya adalah 4 dan 11. Pada pengujian nilai interval m , nilai m pada proses klasifikasi FKNN tidak berpengaruh secara langsung terhadap hasil akurasi, namun berpengaruh pada nilai derajat keanggotaan tiap-tiap *data testing* terhadap masing-masing kelas. Variabel m merupakan bobot pangkat pada FKNN yang digunakan untuk menentukan seberapa besar jarak tertimbang ketika menghitung kontribusi tiap tetangga pada nilai keanggotaan. Nilai m yang semakin besar akan membuat nilai keanggotaan semakin rendah, sehingga berpengaruh pada penentuan hasil kelas prediksi, dimana kelas prediksi tersebut yang mempengaruhi akurasi. Hal ini dikarenakan nilai m menentukan berapa banyak bobot jarak antara masing-masing tetangga ke nilai keanggotaan.



Gambar 5.1 Grafik hasil rata-rata akurasi pengujian nilai interval m

5.3.2 Analisis Hasil Pengujian Pengaruh Nilai Interval k Terhadap Rata-rata Akurasi

Berdasarkan pada pengujian terhadap pengaruh nilai interval k yang ditunjukkan pada Gambar 5.2, terlihat bahwa nilai interval antara 1 sampai 10 memberikan nilai rata-rata akurasi tertinggi yaitu 90.17% dengan nilai m terpilih adalah 1.8302, nilai k terpilih adalah 4, dan fitur terpilihnya adalah 4 dan 11. Pada pengujian nilai interval k , nilai k sangat berpengaruh pada hasil akurasi karena nilai derajat keanggotaan pada klasifikasi FKNN berdasarkan pada jumlah tetangga pada rentang k , selain itu karena adanya jarak kedekatan terhadap tetangga terdekatnya. Nilai k yang kecil atau besar tidak terlalu menurunkan akurasi karena adanya pengaruh nilai keanggotaan tetangga terdekat ke- j terhadap kelas ke- i pada klasifikasi FKNN, dimana nilai keanggotaan yang bernilai maksimum yang digunakan sebagai penentu kelas prediksi. Selain itu kemungkinan dikarenakan adanya proses random data yang dilakukan agar sebaran data tiap-tiap kelas pada tiap K -fold merata, sehingga nilai keanggotaan tetangga ke- j pada kelas ke- i seimbang. Pada penelitian ini nilai k antara 1 sampai 20 cenderung menghasilkan nilai akurasi lebih baik dibandingkan dengan nilai k yang lebih dari 20, walaupun tidak ada penurunan akurasi yang signifikan dengan menggunakan nilai k yang lebih dari 20.

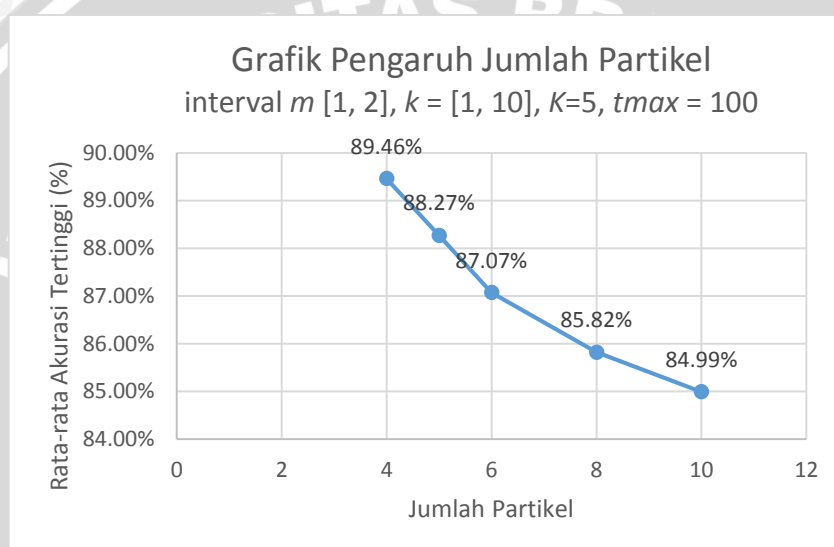


Gambar 5.2 Grafik hasil rata-rata akurasi pengujian nilai interval k

5.3.3 Analisis Hasil Pengujian Pengaruh Jumlah Partikel Terhadap Rata-rata Akurasi

Berdasarkan pada pengujian terhadap pengaruh jumlah partikel yang ditunjukkan pada Gambar 5.3, terlihat bahwa dengan menggunakan jumlah partikel sebanyak 4 memberikan nilai rata-rata akurasi tertinggi yaitu 89.46% dengan nilai m terpilih adalah 1.4231, nilai k terpilih adalah 1, dan fitur terpilihnya adalah 4, 7, 8, dan 11. Pada pengujian jumlah partikel, jumlah partikel sangat berpengaruh terhadap nilai akurasi, karena jumlah partikel berpengaruh pada ruang pemilihan solusi pada TVPSO, dimana solusi tersebut merupakan partikel terbaik untuk mendapatkan nilai akurasi yang optimal, selain itu banyaknya partikel juga merupakan penentu banyaknya titik-titik pada kurva ROC. Semakin banyak partikel yang dihasilkan semakin banyak pula ruang pemilihan, sehingga pilihan solusi terbaik lebih banyak. Namun terlalu banyak partikel juga tidak baik dalam pemilihan solusi terbaiknya, karena apabila ruang pencarian atau iterasi maksimumnya tidak sebanding dengan jumlah partikel maka hasil solusi yang didapatkan kurang optimal dan rata-rata akurasi cenderung rendah, misalnya pada 10 partikel dengan iterasi maksimumnya hanya 100 diperoleh g_{best} yang menghasilkan rata-rata akurasi terendah, hal tersebut terbukti juga pada pengujian diluar skenario dengan menggunakan 20 partikel, $K=8$, interval $m=[1,2]$, interval

$k=[1, 10]$, dan iterasi hanya 100 ditunjukkan hasil akurasi tertinggi hanya 76, 93% terlihat pada Lampiran 14. Sehingga semakin banyak partikel semakin banyak pula iterasinya. Banyak iterasi 10 atau lebih kali lipat dari jumlah partikelnya misalnya dengan 10 partikel maka banyak iterasinya 100 atau lebih dari 100 dengan begitu hasilnya akan lebih baik seperti terlihat pada Lampiran 14 dengan pengujian diluar skenario menggunakan 15 partikel dengan 200 iterasi. dari pengujian ini dengan penggunaan iterasi maksimum 100 jumlah partikel yang optimal adalah 4, selain itu terlalu banyak partikel akan lebih memperlambat proses komputasi.

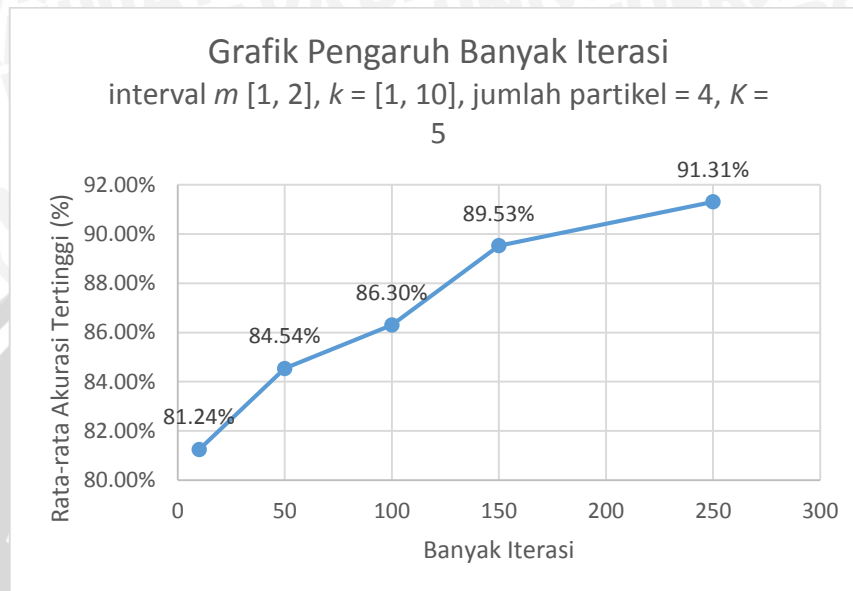


Gambar 5.3 Grafik hasil rata-rata akurasi pengujian jumlah partikel

5.3.4 Analisis Hasil Pengujian Pengaruh Jumlah Iterasi Terhadap Rata-rata Akurasi

Berdasarkan pada pengujian terhadap pengaruh jumlah iterasi yang ditunjukkan pada Gambar 5.4, terlihat bahwa nilai iterasi 250 memberikan nilai rata-rata akurasi tertinggi yaitu 91.31% dengan nilai m terpilih adalah 1.8451, nilai k terpilih adalah 4, dan fitur terpilihnya adalah 4, 8, 9, 11, dan 14. Pada pengujian jumlah iterasi sangat mempengaruhi hasil rata-rata akurasi, karena kemungkinan solusi terbaik yang dihasilkan pada proses TVPSO-FKNN dicari sampai banyak iterasi yang di-*input*-kan. Sehingga dengan mendapatkan solusi terbaik dapat dipastikan memberikan nilai rata-rata akurasi yang baik juga. Semakin banyak iterasi akan memberikan hasil solusi yang jauh lebih baik dan lebih optimal, hal ini

dikarenakan dengan banyaknya iterasi maka banyak pula ruang pencarian solusi yaitu partikel terbaiknya (*gbest*), tetapi terlalu banyak iterasi mengakibatkan proses komputasi akan berjalan lebih lama.

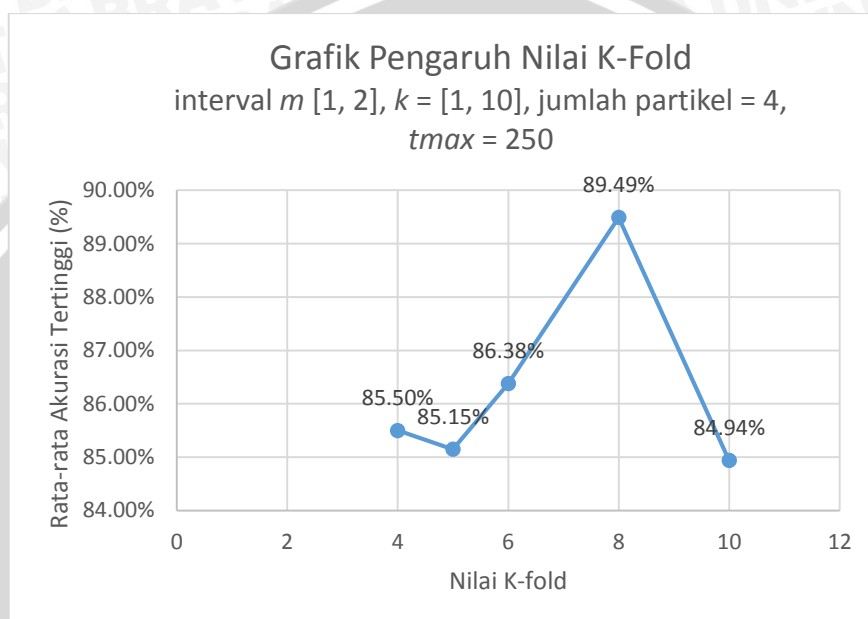


Gambar 5.4 Grafik hasil rata-rata akurasi pengujian jumlah iterasi

5.3.5 Analisis Hasil Pengujian Pengaruh Nilai K Terhadap Rata-rata Akurasi

Berdasarkan pada pengujian terhadap pengaruh nilai K yang ditunjukkan pada Gambar 5.5, terlihat bahwa nilai $K = 8$ memberikan nilai rata-rata akurasi tertinggi yaitu 89.49% dengan nilai m terpilih adalah 1.3005, nilai k terpilih adalah 2, dan fitur terpilihnya adalah 4, 5, 8, 9, dan 11. Pada pengujian nilai K dengan melakukan random data yang dilakukan agar penyebaran data tiap-tiap kelas lebih merata pada tiap-tiap bagian K -fold, sehingga karena sebaran datanya lebih merata maka hal ini dapat meningkatkan nilai akurasi pada proses klasifikasi FKNN, selain itu proses K -fold dilakukan pada pengujian ini untuk mengoptimalkan seluruh data, karena semua data dapat menjadi *data testing* dan *data training*. Nilai K yang digunakan pada proses TVPSO-FKNN mempengaruhi banyaknya solusi yang dihasilkan. Sehingga semakin banyak nilai K , semakin banyak pula solusi yang didapatkan yaitu banyaknya *gbest* atau partikel terbaik yang kemungkinan dapat menghasilkan nilai rata-rata akurasi yang lebih beragam, sehingga dapat diambil

gbest yang menghasilkan nilai rata-rata akurasi yang tertinggi. Sedangkan pada proses uji hasil TVPSO-FKNN, nilai K yang semakin banyak akan sangat mempengaruhi nilai rata-rata akurasi, bahkan dapat menurunkan nilai akurasi dimana apabila akurasi tiap *fold* terdapat nilai akurasi yang kecil maka rata-rata akurasi akan menjadi lebih kecil.

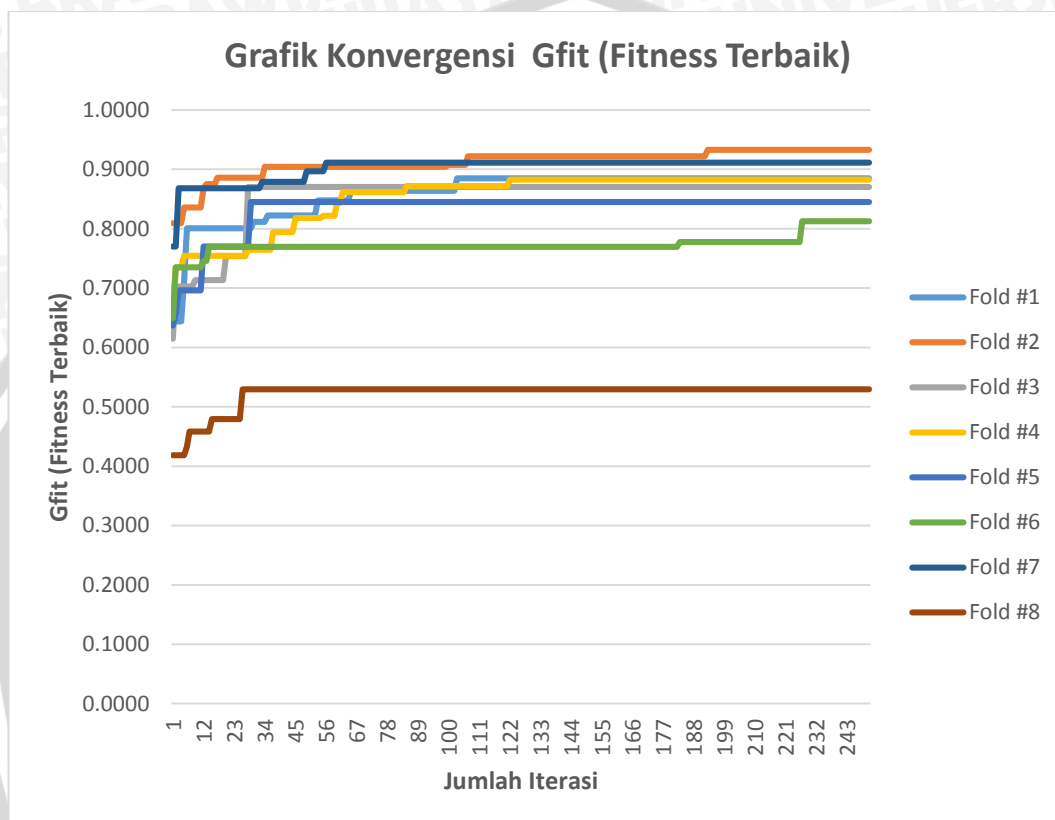


Gambar 5.5 Grafik hasil rata-rata akurasi pengujian nilai K

5.3.6 Analisis Hasil Pengujian Mengukur Konvergensi pada TVPSO

Berdasarkan pada pengujian terhadap konvergensi TVPSO, dapat dilihat bahwa nilai *fitness* terbaik (G_{fit}) dari partikel terbaik (G_{best}) sebanyak 8 yaitu 0.9325 konvergensi pada iterasi ke-193 dan pada *fold* #2, selain itu pada *fold* #7 dengan nilai *fitness* terbaiknya memiliki selisih sedikit dengan *fold* #2 yaitu 0.9112 yang mengalami konvergensi pada iterasi ke-56. Dari grafik yang ditunjukkan pada Gambar 5.6, hasil konvergensi pada TVPSO dalam berevolusi untuk mencari solusi terbaik sangat bagus. Dimana konvergensi tercepat dicapai setelah iterasi ke-28 dan hasil terbaik berdasarkan nilai *fitness* tertinggi didapatkan pada iterasi ke-56 dan ke-193. Maka TVPSO telah mencapai konvergensi ke solusi optimum global dengan pemberhentian kriteria pencarian solusi tercepat setelah iterasi ke-28. Namun untuk mendapatkan hasil optimal yang tercepat diperoleh pada iterasi ke-56 dengan maksimum iterasi 250. Sehingga TVPSO-FKNN telah berhasil

meningkatkan efektifitas dan efisiensi dalam pencarian solusi yang optimal yaitu nilai m , k , dan fitur terbaik pada kinerja proses klasifikasi FKNN dengan hasil akurasi yang lebih baik. Hal ini juga terlihat pada kenaikan pada nilai *fitness* yang dicapai stabil setelah iterasi ke-28.



Gambar 5.6 Grafik hasil pengujian konvergensi pada TVPSO

BAB VI

PENUTUP

Pada bab penutup akan dibahas mengenai kesimpulan dan saran dari implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga. Berikut kesimpulan dan sarannya.

6.1 Kesimpulan

Berdasarkan hasil penelitian implementasi metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* untuk mengklasifikasikan status sosial ekonomi keluarga, dapat diambil kesimpulan sebagai berikut:

1. Metode *Adaptive Fuzzy K-Nearest Neighbor* berdasarkan *Particle Swarm Optimization* dapat diterapkan pada klasifikasi status sosial ekonomi keluarga dengan kinerja klasifikasi yang lebih optimal. Adapun langkah-langkah dalam implementasi metode ini secara garis besar adalah:
 - Pertama adalah melakukan proses normalisasi pada *dataset*, kemudian setelah itu hasil dari normalisasi dikalikan bobot atribut atau fitur.
 - Kedua adalah membangkitkan partikel dan proses TVPSO-FKNN sampai mendapatkan partikel terbaik yang berisi solusi yaitu nilai m , k , dan fitur yang terpilih.
 - Proses ketiga adalah uji hasil dari TVPSO-FKNN yaitu *gbest* tiap-tiap *fold* sebanyak K , diujikan dengan *train* FKNN menggunakan *K-fold cross validation* juga sebanyak K sesuai yang digunakan pada proses TVPSO-FKNN. Nilai m , k dan fitur yang digunakan adalah yang terdapat pada *gbest* yang merupakan nilai m , k , dan fitur terpilih hingga mendapatkan nilai rata-rata akurasi tiap-tiap *gbest* dari beberapa *gbest* yang terpilih ambil yang memiliki rata-rata akurasi tertinggi untuk dianalisis.
2. Nilai rata-rata akurasi maksimum diperoleh pada pengujian jumlah iterasi 250 yaitu 91.31% dengan nilai m terpilih yaitu 1.8451, nilai k terpilih adalah

4, dan fitur terpilihnya adalah fitur ke 4, 8, 9, 11, dan 14. Pada pengujian tersebut menggunakan interval $m = [1, 2]$, interval $k = [1, 10]$, jumlah partikel = 4, dan $K = 5$. Pada pengujian konvergensi didapatkan hasil bahwa penggunaan metode TVPSO telah mengalami konvergensi dalam pencarian solusi terbaiknya, dilihat dari *fitness* terbaik tiap-tiap partikel terbaik. Salah satu yang terbaik telah mengalami konvergensi pada iterasi ke 56 dari iterasi maksimum 250. Pada kasus ini partikel terbaik yang didapatkan dari proses TVPSO-FKNN belum tentu menghasilkan nilai rata-rata akurasi tertinggi, karena nilai pada beberapa fitur untuk kelas satu dengan kelas yang lain memiliki nilai yang selisihnya sangat kecil bahkan sama.

6.2 Saran

Adapun saran yang diberikan oleh penulis untuk penelitian lebih lanjut adalah:

1. Pada penelitian selanjutnya terkait dengan perkembangan data agar metode ini dapat diimplementasikan dengan menggunakan *database* untuk data yang lebih besar karena pada kenyataannya kasus ini memiliki data yang lebih besar untuk cakupan lembaga yang lebih tinggi, dimana pada penelitian ini masih digunakan *load* data dari file *excel* dengan 121 data.
2. Pada penelitian selanjutnya agar melakukan penelitian dengan pengujian pada parameter bobot inertia w terkait penggunaan nilai w_{min} dan w_{max} , kemudian pada parameter koefisien percepatan c_1 dan c_2 terkait penggunaan nilai konstanta c_{1f} , c_{1i} , c_{2f} , dan c_{2i} . Selain itu pengujian pada nilai bobot α dan β untuk mendapatkan nilai yang sesuai pada algoritma TVPSO.
3. Pada penelitian ini belum digali lebih dalam mengenai pengaruh *imbalance class* pada metode FKNN atau PSO, dimana ketidakseimbangan porsi data antara kelas satu dengan kelas yang lain pada *data training*. Sehingga penelitian lebih lanjut agar dapat menggunakan metode yang lain untuk mengatasi permasalahan tersebut dengan cakupan data yang lebih besar. Misalnya menggunakan metode *Fuzzy Support Vector Machines* (FSVM), karena metode FSVM pada sebuah penelitian yang lain digunakan karena terbukti berhasil digunakan pada klasifikasi *multi-class* dengan *imbalance class*.

4. Pada penelitian ini nilai bobot atribut yang digunakan merupakan nilai yang diperoleh berdasarkan nilai subyektif dari pakar yaitu kepala desa Kelurahan Jugo, sehingga bobot tersebut belum dapat dipastikan tingkat konsistensinya dan apakah bobot tersebut sudah tepat. Untuk mengurangi subyektifitas penilaian dari pakar agar dilakukan pengujian bobot berdasarkan dari beberapa pakar dan menggunakan sebuah metode untuk mengukur tingkat konsistensi dari bobot yang digunakan. Misalnya dengan menggunakan metode *Analytical Hierarchy Proses* (AHP), karena metode ini dapat digunakan sebagai solusi untuk mengatasi permasalahan multi kriteria dengan pengukuran konsistensi dalam pembuatan keputusan yaitu menentukan bobot kriteria status sosial ekonomi keluarga yang multi kriteria agar sesuai dengan kriteria yang dipentingkan.



DAFTAR PUSTAKA

- Aedi, Nur. 2010, "Pengolahan dan Analisis Data Hasil Penelitian", Bahan Belajar Mandiri, Fakultas Ilmu Pendidikan, Universitas Pendidikan Indonesia, http://file.upi.edu/Direktori/DUALMODES/PENELITIAN_PENDIDIKA_N/BBM_8.pdf.
- Amri, M., Khoiril. 2013, "Penerapan Data Mining untuk Menentukan Kriteria Calon Nasabah Potensial pada AJB BumiPutera 1912 Palembang", Skripsi Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Bina Darma, <http://eprints.binadarma.ac.id/255/1/Penerapan%20Data%20Mining%20U ntuk%20Menentukan%20Kriteria%20Calon%20Nasabah%20Potensial%2 0Pada%20AJB%20Bumiputera%201912%20Palembang.pdf>.
- Anonim. 2013, "Bab IV Klasifikasi", Materi Data Mining, Modul Kuliah, <http://www.biomaterial.lipi.go.id/ilkomers2010materi/SEMESTER%20IV /DAMING/Referensi/modul%20kuliah%20PDF/Bab%204%20Klasifikasi. pdf>.
- Asep, Saefulloh, dan Sugeng, Santoso. 2013, "Penerapan Metode Klasifikasi Data Mining untuk Prediksi Kelulusan Tepat Waktu", Penelitian Dosen Sekolah Tinggi Manajemen dan Ilmu Komputer Raharja, 462 Sistem Informasi, http://hibah.ilearning.me/wpcontent/uploads/sites/381/2013/11/AsepSaeful lah_STMIKRaharja_PDP.pdf.
- Baharsyah, Ikram. 2014, "Klasifikasi Deep Sentiment Analisis E-Complaint Universitas Brawijaya Menggunakan Metode K-Nearest Neighbour", Prodi Teknik Informatika. Fakultas PTIIK, Universitas Brawijaya, Malang.
- Basrowi, dan Juariyah , Siti. 2010, "Analisis Kondisi Sosial Ekonomi dan Tingkat Pendidikan Masyarakat Desa Srigading, Kecamatan Labuhan Maringgai, Kabupaten Lampung Timur", Jurnal Ekonomi dan Pendidikan, Volume 7 Nomer 1, Dosen Pendidikan IPS FKIP Unila, <http://journal.uny.ac.id/index.php/jep/article/view/577>.

Chen, Hui, Ling, at all. 2011, "An Adaptive Fuzzy K-Nearest Neighbor Method Based on Parallel Particle Swarm Optimation for Bankruptcy Prediction", Part 1 LNAI 6634 Page 249-264, Springer-Verlag Berlin Heidelberg, <http://www.sujingwang.name/publication/pakdd11.pdf>.

Cheng, H.D., at all. 2010, "Automated Breast Cancer Detection and Clasification Using Ultrasound Image: A Survey", Pattern Recognition, Vol. 43, hal. 299-317.

Dewi, Sri, Kusuma, dan Purnomo, Hari. 2004, "Aplikasi Logika Fuzzy untuk Pendukung Keputusan", Edisi 1, Graha Ilmu, Yogyakarta.

Dhanasaputra, Nalendra, dan Santosa, Budi. 2012, "Pengembangan Algoritma Cat Swarm Optimization (CSO) untuk Klasifikasi", Jurusan Teknik Industri, Institut Teknologi Sepuluh Nopember (ITS) Surabaya, <http://dosen.narotama.ac.id/wpcontent/uploads/2012/03/PENGEMBANGAN-ALGORITMA-CAT-SWARM-OPTIMIZATION-CSO-UNTUK-KLASIFIKASI.pdf>.

Fawcett, Tom. 2006, "An Introduction to ROC Analysis", Pattern Recognition Letters 27 Page 861-874, Elseiver BV, Institutr for the Study of Learning and Expertise, 2164 Staunton Court, Pulo Alto, CA 94306, USA.

Han J and Kamber M. 2001, "Data Mining Concepts & Techniques", USA: Simon Frase University Academic Press.

Hastuti, Khafiizh. 2012, "Analisis Komparasi Algoritma Klasifikasi Data Mining untuk Prediksi Mahasiswa Non Aktif", Semantik, ISBN 979-26-0255-0 INFRM 241-248, Semarang.

Hermaduant, Ninki dan Dewi, Sri, Kusuma. 2008. *Sistem Pendukung Keputusan Berbasis SMS untuk Menentukan Status Gizi dengan Metode K-Nearest Neighbor*. ISSN 1907-5022 Hal: E49 sampai E55. Seminar Nasional Aplikasi Teknologi Informasi (SNATI). Yogyakarta.

Kadarsah. 2010, "Aplikasi ROC untuk Uji Keandalan Model HYBMG", Jurnal Meteorologi dan Geofisika Volume 11 Nomer 1 halaman 32-42.

Leidiyana, Henny. 2013, "Penerapan Algoritma K-Nearest Neighbor untuk Penentuan Resiko Kredit Kepemilikan Kendaraan Bermotor", Jpiksel 1(1): 65-76, Program Pasca Sarjana Magister Ilmu Komputer, STMIK Nusa Mandiri.

Makhtar, M., Neagu, D.C., Ridley, M.J. 2011, "Comparing Multi-class Classifiers: On the Similarity of Confusion Matrices for Predictive Toxicology Applications", In H.Yin, W.Wang, and V.Rayward-Smith (Eds), LNCS 6936, pp.252-261, Springer-Verlag Berlin Heidelberg.

Mardianto, Is, dan Pratiwi, Dian. 2008, "Sistem Deteksi Penyakit Pengeroposan Tulang dengan Metode Jaringan Saraf Tiruan Backpropagation dan Representasi Ciri Dalam Ruang Eigen", CommIT Vol 2 No 1 Hal 69-80, Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Trisakti.

Mohammad, Alkis, Fuady, Fariza, Arna, dan Prasetyaningrum, Ira. 2011, "Aplikasi GIS Berbasis J2ME Pencarian Jalur Terpendek Menggunakan Algoritma Particle Swarm Optimization (PSO) Di Kabupaten Bangkalan", Seminar Industri Elektronika, ISBN: 978-979-8689-14-7 Page 320-324, Politeknik Elektronika Negeri Surabaya, ITS,
[http://repo.eepisits.edu/1475/1/\[FH1027\]_pp.320324_Aplikasi_GIS_Berbasis_J2ME_Pencarian_Jalur_Terpendek.pdf](http://repo.eepisits.edu/1475/1/[FH1027]_pp.320324_Aplikasi_GIS_Berbasis_J2ME_Pencarian_Jalur_Terpendek.pdf).

Mohanty, A.K., et all. 2011, "Calssifying Benign and Malignant Mass using GLCM and GLRLM based Texture Features from Mammogram", International Jurnal of Engineering Research and Applications (IJERA), Vol. 1, page 687-693.

Mubarok, Ahmad, Zaki. 2013, "Penerapan Metode Fuzzy KNN pada Klasifikasi Jenis Mangga Berdasarkan Tekstur Daun", Prodi Teknik Informatika, Fakultas PTIIK, Universitas Brawijaya, Malang.

Muflikhah, Lailil. 2012, "Bab Klasifikasi K-Nearest Neighbor (KNN)", Materi Mata Muliah Data Mining, Prodi Informatika, Fakultas PTIIK, Universitas Brawijaya, Malang.

Prasetyo, Dimas. 2014, "Klasifikasi Genre Film Berdasarkan Judul dan Sinopsis Menggunakan Fuzzy K-Nearest Neighbour (Fuzzy K-NN)", Prodi Ilmu Komputer, Fakultas PTIIK, Universitas Brawijaya, Malang.

Puspitaningrum, Kiki, Aprilia, P., Diah, Anggraeni, dan P, Ika, Retno. 2012, "Pemilihan Judul Film Menggunakan Metode Sequential pattern Mining Pada Sistem Informasi Penyewaan VCD/DVD", Laporan Project Akhir Mata Kuliah Data Mining, Prodi Informatika, Fakultas PTIIK, Universitas Brawijaya Malang.

Ratnaweera, Asanga at all. 2004, "Self Organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients", Transactions on Evolutionary Computation, Vol.8, No.3, IEEE.

Riyanda, Fitra. 2013, "Klasifikasi Data Otomotif Menggunakan SVM Light", Proyek Mata Kuliah Data Mining Lanjut, Jurusan Informatika, Fakultas MIPA, Universitas Syiah Kuala Darussalam Banda Aceh.

Saifi, Saifullah, and Mehmood, Tariq. 2011, "Effects of Socioeconomic Status on Students Achievement", International Journal of Social Sciences and Education, Volume: 1 Issue: 2 Page: 119-128.

Santoso, Budi. 2014, "Tutorial Particle Swarm Optimization", Materi Kuliah, Dosen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya, <http://oc.its.ac.id/ambilfile.php?idp=1221>.

Selly, Meristika, Yanita. 2013, "Perbandingan K-Nearest Neighbor dan Fuzzy K-Nearest Neighbor pada Diagnosis Penyakit Diabetes Melitus", Prodi Ilmu Komputer, Fakultas PTIIK, Universitas Brawijaya, Malang.

Shofa, Rahmi, Amiratus. 2013, "Penerapan Metode Fuzzy K-Nearest Neighbor (FK-NN) untuk Menentukan Kualitas Hasil Rendaman Tanaman Tebu", Prodi Informatika, Fakultas PTIIK, Universitas Brawijaya, Malang.

Sparwls, P. 1995. "The Physical Principles of Medical Imaging", 2nd edition, Medical Physics Pub.Corp.

Suyanto. 2010, "Algoritma Optimasi Deterministik atau Probabilitik", Edisi 1, Graha Ilmu, Yogyakarta.

Wardani, Lidya. 2012, "Hubungan Status Sosial Ekonomi Orangtua dengan Prestasi Belajar Siswa", BAB II Tinjauan Pustaka,
<http://digilib.unimed.ac.id/public/UNIMED-Undergraduate-22748-BAB%20II.pdf>.

Yan Cai at all. 2013, "Transformer Fault Diagnosis Based On Hierarchical Fuzzy Support Vector Machines", Vol.11, No.10, Page 5842-5850, eISSN: 2097-278X, TELKOMNIKA.

Yang Jiaqin and Shi Ping. 2002, "Applying An Analytic Hierarchy Process in Firm's Overall Performance Evaluation: A Case Study In China", ISSN: 1083-4346, Page 30-46, International Journal of Business, 7(1).

Yessivirna, Riska. 2013, "Klasifikasi Suara Berdasarkan Gender (Jenis Kelamin) dengan Metode K-Nearest Neighbor (KNN)",_Prodi Ilmu Komputer, Fakultas PTIIK, Universitas Brawijaya, Malang.

Yung-Kyun, Chongwoo Frank, and D., Daniel. 2012, "Diffusion Decision Making for Adaptive k-Nearest Neighbour Classification",
http://books.nips.cc/papers/files/nips25/NIPS2012_0954.pdf.

LAMPIRAN

Lampiran 1. Form Survey



**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU
KOMPUTER**

JL. Veteran No. 8, Malang, 65145, Indonesia
Telp. : +62-341-577911 ; Fax : +62-341-577911
<http://ptiik.ub.ac.id> Email : ptiik@ub.ac.id

FORM SURVEY DATA KELUARGA

Nama Kepala Keluarga	:	
Umur	:	
Pendidikan	:	
Pekerjaan	:	
Rata-rata Penghasilan	:	/bulan
Nama Istri	:	
Umur	:	
Pendidikan	:	
Pekerjaan	:	
Rata-rata Penghasilan	:	/bulan
Jumlah Anak	:	SD : SMP : SMA : PT : Bekerja :
Biaya Sekolah Anak	:	/bulan /tahun
Status Rumah	:	a. Milik Sendiri b. Kontrak c. Tinggal dengan Saudara d. Lain-lain ()
Jumlah Kendaraan Mobil	:	
Jumlah Kendaraan Motor	:	
Jumlah Kendaraan Lain	:	
Kadaan Rumah	:	
Luas Rumah	:	
Luas Tanah yang Dimiliki	:	

Status Listrik : a. Milik Sendiri
b. Numpang

Rata-rata Tagihan Listrik : /bulan. Daya : Watt

Rata-rata Tagihan Air : /bulan

Tagihan PBB terakhir :

Usaha Lain :

Rata-rata Hasil Usaha : /bulan

Rata-rata Pengeluaran : /bulan /tahun

Jumlah Ternak : Sapi :
Kambing :
Ayam :
Burung :
Lain-lain : ()

Keterangan Lain :

Kepala Rumah Tangga

Lampiran 2. Surat Keterangan dari RT



**KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS BRAWIJAYA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU
KOMPUTER**

JL. Veteran No. 8, Malang, 65145, Indonesia
Telp. : +62-341-577911 ; Fax : +62-341-577911
<http://ptiik.ub.ac.id> Email : ptiik@ub.ac.id

SURAT PERNYATAAN SURVEY

Saya yang bertanda tangan di bawah ini:

Nama : Ika Retno Putri

NIM : 125150209111006

Mahasiswa Universitas Brawijaya. Telah benar-benar melakukan observasi dengan melakukan survey data keluarga pada 4 RT di RW 001 Dusun Jugo Desa Jugo Kecamatan Kesamben Kabupaten Blitar pada tanggal 4 sampai 6 Maret 2014 yaitu:

1. RT 001 sebanyak 40 KK
2. RT 003 sebanyak 22 KK
3. RT 004 sebanyak 25 KK
4. RT 005 sebanyak 34 KK

Surat pernyataan ini dibuat sebagai bukti tertulis guna membantu terselesaikannya penelitian skripsi yang dilakukan nama tersebut diatas.

Demikian surat ini dibuat semoga dapat dipergunakan sebagaimana mestinya.

Jugo, 6 Maret 2014

Ika Retno Putri

Mengetahui,

Ketua RT 001 RW 001

RUKUN TETANGGA 1
RUKUN WARGA 1
Bpk. Mesiyo JUGO

Ketua RT 004 RW 001
RUKUN TETANGGA 4
RUKUN WARGA 1
DUSUN JUGO
Bpk. Ali Susanto

Ketua RT 003 RW 001

RUKUN TETANGGA 3
RUKUN WARGA 1
Bpk. Tayip
DUSUN JUGO

Ketua RT 005 RW 001
RUKUN TETANGGA 5
RUKUN WARGA 1
DUSUN JUGO
Bpk. Gunawan

Lampiran 3. Data KK

DATA KELUARGA																
KK	Pendidikan	Pekerjaan	Usaha Lain	Pendapatan	Pengeluaran	Status Rumah	Keadaan Rumah	Luas Rumah (m2)	Jumlah Mobil	Jumlah Motor	Luas Tanah/Sawah/Ladang (are)	Sapi	Kambing	Unggas	Kelas	
1	1	2	0	2,000,000	1,500,000	2	2	63	0	2	26	0	1	0	M	
2	2	4	0	3,500,000	2,000,000	2	2	72	0	1	0	0	0	0	A	
3	0	2	0	1,500,000	1,500,000	1	1	36	0	2	0	0	0	0	M	
4	1	2	2	1,000,000	1,000,000	2	1	60.5	0	1	0	0	0	6	B	
5	0	1	0	750,000	750,000	2	1	60	0	1	0	0	0	0	B	
6	0	1	0	1,000,000	1,000,000	2	0	30	0	1	0	0	5	10	B	
7	0	2	0	1,000,000	1,000,000	2	0	50	0	2	0	0	0	0	B	
8	1	2	0	1,000,000	800,000	2	1	72	0	1	0	0	0	0	B	
9	0	1	0	500,000	300,000	2	1	50	0	1	0	0	0	0	B	
10	0	2	0	700,000	500,000	2	1	200	0	1	0	0	0	0	M	
11	1	2	0	750,000	500,000	2	1	150	0	1	0	0	7	0	M	
12	0	3	0	1,500,000	900,000	2	1	350	0	1	19	0	0	0	M	
13	1	3	0	3,000,000	1,500,000	2	1	150	0	2	0	0	0	0	A	
14	0	3	3	1,500,000	1,000,000	2	1	72	0	0	14	2	0	0	M	
15	0	3	3	1,500,000	1,000,000	2	1	450	0	2	5.5	0	6	0	M	
16	0	3	3	1,000,000	800,000	2	2	77.5	0	1	100	0	10	0	M	
17	0	1	0	500,000	500,000	1	0	100	0	0	0	0	0	0	B	
18	1	3	0	2,000,000	1,000,000	2	2	66	0	1	10	0	0	0	A	
19	1	3	0	5,000,000	2,500,000	2	2	66	0	1	0	0	0	0	A	
20	0	1	0	2,350,000	1,200,000	2	1	250	0	1	0	0	0	0	M	
21	1	4	3	6,000,000	2,500,000	2	2	100	0	2	2	0	0	0	A	
22	1	3	0	1,500,000	1,000,000	2	1	120	0	1	2	0	0	20	M	

23	1	2	0	1,500,000	1,500,000	2	1	72	0	1	17	0	0	0	M
24	1	3	3	2,000,000	600,000	2	1	60	0	0	5	0	3	0	M
25	0	3	0	1,000,000	1,000,000	2	1	54	0	2	2	0	4	0	M
26	1	2	0	1,000,000	1,000,000	2	2	250	1	1	5	0	0	0	M
27	1	3	0	4,000,000	1,000,000	2	2	200	1	1	10	0	0	0	A
28	0	1	0	1,000,000	1,000,000	2	1	48	0	2	0	0	0	0	B
29	0	1	0	500,000	500,000	2	1	150	0	0	0	0	0	0	B
30	1	3	3	10,000,000	1,500,000	2	2	150	0	1	25	0	0	0	A
31	0	1	0	200,000	450,000	1	0	15	0	0	0	0	0	0	B
32	0	2	0	2,000,000	1,000,000	2	1	120	1	3	7	0	0	0	M
33	1	3	2	1,500,000	1,000,000	2	2	96	0	1	0	0	0	0	M
34	0	1	0	500,000	500,000	2	1	24	0	0	0	0	3	0	B
35	1	2	1	1,000,000	500,000	2	1	72	0	1	1.12	0	0	0	M
36	1	2	3	1,600,000	1,000,000	2	1	300	0	0	0	0	0	0	M
37	1	1	2	500,000	300,000	1	1	150	0	1	0	0	6	21	B
38	0	3	3	500,000	400,000	2	1	150	0	2	33	2	5	0	M
39	1	3	0	700,000	500,000	2	1	250	0	1	10	0	0	0	M
40	2	3	0	10,250,000	3,000,000	2	2	200	0	1	0	0	0	0	A
41	1	2	3	3,000,000	2,000,000	2	2	72	1	1	15	0	6	0	A
42	1	3	0	2,450,000	2,000,000	2	1	650	0	2	0	0	0	7	M
43	1	1	0	1,000,000	700,000	2	1	50	0	1	0	0	4	3	B
44	1	3	2	1,500,000	1,500,000	2	1	45	0	1	2	0	0	0	M
45	1	3	2	2,000,000	1,000,000	2	1	200	0	2	14	0	0	0	M
46	1	3	3	3,000,000	1,000,000	2	1	54	0	1	0	0	0	0	M
47	1	2	0	300,000	300,000	2	0	20	0	0	0	0	0	0	B
48	2	2	0	2,000,000	1,500,000	1	1	40	0	1	0	0	0	0	M

49	1	1	0	400,000	500,000	2	1	40	0	0	0	0	0	0	B
50	1	1	0	500,000	500,000	1	1	300	0	0	0	0	0	0	B
51	1	4	0	3,500,000	1,500,000	2	1	60	0	2	0	0	0	0	A
52	1	2	0	1,200,000	1,200,000	2	1	60	0	2	0	0	0	0	M
53	1	2	0	1,000,000	1,000,000	1	1	20	0	1	0	0	0	0	B
54	0	1	0	800,000	800,000	2	1	40	0	0	0	0	0	0	B
55	0	1	0	500,000	500,000	2	1	48	0	0	0	0	0	10	B
56	1	2	0	1,000,000	1,000,000	2	1	100	0	0	0	0	0	0	B
57	1	2	0	1,000,000	1,000,000	2	1	40	0	0	0	0	0	0	B
58	1	3	0	500,000	500,000	2	0	15	0	0	0	0	0	0	B
59	1	3	0	1,500,000	1,500,000	2	1	64	0	1	0	0	10	50	M
60	2	4	0	8,687,000	5,000,000	2	2	100	1	3	0	0	0	0	A
61	1	2	0	1,500,000	500,000	2	1	66	0	1	0	0	0	0	M
62	1	4	0	3,500,000	1,000,000	2	2	130	0	3	0	0	0	50	A
63	1	1	0	2,400,000	1,000,000	2	1	20	0	1	0	0	2	3	M
64	0	3	0	1,000,000	600,000	2	2	80	0	1	33	0	3	0	M
65	1	2	0	800,000	500,000	2	1	40	0	1	0	0	2	0	B
66	0	1	0	1,000,000	500,000	2	1	100	0	0	0	0	0	0	B
67	0	2	0	2,000,000	1,000,000	2	1	150	0	0	0	0	0	0	M
68	1	3	2	2,000,000	1,500,000	2	1	150	0	1	20	0	0	0	A
69	1	2	0	2,000,000	2,000,000	2	1	500	0	1	0	0	0	0	M
70	1	3	0	1,000,000	500,000	1	1	72	0	1	0	0	0	0	B
71	1	3	0	500,000	750,000	2	1	72	0	1	55	0	8	0	M
72	0	3	0	3,000,000	500,000	2	1	200	0	0	0	2	35	28	A
73	1	2	0	1,000,000	500,000	2	1	200	0	1	0	0	0	0	B
74	0	3	0	1,500,000	1,500,000	2	2	200	0	1	21	1	3	8	M

75	0	1	0	200,000	200,000	2	0	16	0	1	0	0	0	0	B
76	0	3	0	500,000	500,000	2	1	150	0	1	2	0	0	0	M
77	0	1	0	500,000	500,000	2	0	18	0	0	0	0	2	5	B
78	0	2	0	2,000,000	2,000,000	1	1	150	0	1	0	0	0	0	M
79	0	1	0	1,000,000	1,000,000	2	1	60	0	0	0	0	0	0	B
80	0	1	0	500,000	500,000	2	0	15	0	1	0	0	0	10	B
81	0	1	0	200,000	250,000	2	0	15	0	0	0	0	0	7	B
82	0	1	0	1,000,000	500,000	2	1	500	0	1	0	2	0	30	B
83	0	0	0	0	0	0	1	500	0	0	0	0	0	0	B
84	1	3	3	2,000,000	1,000,000	2	1	150	0	2	3	0	0	0	M
85	0	0	0	0	0	2	1	250	0	0	0	0	0	0	B
86	1	2	1	700,000	500,000	2	1	75	0	1	0	0	0	0	B
87	2	2	2	1,000,000	500,000	1	1	250	0	1	0	0	0	0	B
88	1	3	0	3,000,000	1,000,000	2	1	72	0	2	6	0	0	0	M
89	0	1	0	500,000	500,000	0	1	72	0	0	0	0	0	0	B
90	1	3	3	3,500,000	1,500,000	2	1	100	0	2	7.5	0	4	0	A
91	1	2	0	1,500,000	1,000,000	2	1	1200	0	1	0	0	2	0	M
92	1	3	3	4,000,000	2,000,000	2	1	54	0	0	31	0	0	0	A
93	1	3	1	4,000,000	1,000,000	2	1	300	0	1	0	0	0	0	A
94	1	4	0	3,000,000	2,500,000	2	2	140	0	2	0	0	0	0	A
95	1	2	3	2,000,000	1,700,000	2	1	900	0	1	4	0	0	0	M
96	1	2	0	2,000,000	1,500,000	2	1	250	0	2	0	0	0	0	M
97	1	1	0	500,000	500,000	1	1	250	0	1	0	2	0	0	B
98	1	2	0	1,500,000	500,000	2	2	68	0	1	0	0	0	0	M
99	1	2	0	2,000,000	1,800,000	2	1	120	0	2	0.6	0	0	0	M
100	1	3	0	1,000,000	500,000	2	1	72	0	2	35	0	0	0	M

101	1	3	0	1,000,000	1,750,000	2	2	600	0	2	42	0	0	0	A
102	1	3	2	1,000,000	750,000	2	2	72	0	1	28	0	0	0	M
103	1	3	3	3,500,000	3,000,000	2	2	550	1	3	31	0	7	120	A
104	2	4	3	2,500,000	2,000,000	2	2	600	1	1	19	0	0	700	A
105	1	2	2	3,200,000	1,700,000	2	2	200	1	1	0	0	0	10	A
106	1	2	0	1,000,000	1,000,000	2	1	150	0	2	0	0	0	0	M
107	0	1	0	500,000	450,000	2	1	150	0	0	0	0	0	0	B
108	2	4	0	3,000,000	2,500,000	2	2	200	0	2	0	0	0	0	A
109	1	2	0	2,000,000	1,000,000	2	1	100	0	1	0	0	0	0	M
110	1	3	0	2,500,000	1,500,000	2	2	200	0	1	0	0	0	0	M
111	1	3	1	800,000	800,000	1	1	36	0	1	0	0	0	0	B
112	1	0	0	1,500,000	1,000,000	2	1	200	0	0	0	0	0	0	M
113	0	0	1	100,000	0	0	0	30	0	0	0	0	0	10	B
114	0	1	1	250,000	600,000	2	0	30	0	0	0	0	0	15	B
115	0	1	0	300,000	300,000	2	0	25	0	0	0	0	2	0	B
116	0	3	2	1,250,000	1,000,000	2	1	50	0	0	0	0	0	0	B
117	0	3	0	800,000	800,000	2	1	72	0	1	2	0	0	0	M
118	0	3	0	500,000	500,000	2	1	250	0	0	10	0	0	0	M
119	1	3	0	2,500,000	2,500,000	2	2	300	2	1	5	0	0	0	A
120	0	1	0	700,000	700,000	1	0	200	0	0	0	0	0	0	B
121	0	3	1	700,000	500,000	2	1	66	0	0	6	0	2	0	M

Bobot Atribut

No	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	
Atribut	Pendidikan	Pekerjaan	Usaha Lain	Pendapatan	Pengeluaran	Status Rumah	Keadaan Rumah	Luas Rumah (m2)	Jumlah Mobil	Jumlah Motor	Luas Tanah/ Sawah/ Ladang (are)	Sapi	Kambing	Unggas	Jumlah
Bobot awal	5	20	5	20	5	10	15	5	15	10	10	10	6	4	140
Bobot setelah normalisasi	0.0357	0.1429	0.0357	0.1429	0.0357	0.0714	0.1071	0.0357	0.1071	0.0714	0.0714	0.0714	0.0429	0.0286	1

Jugo, 18 Mei 2014
Mengetahui
Kepala Desa Jugo



Lampiran 4. Sampel Dataset pada Perhitungan Manual

No	KK	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	Kelas
1	60	2	4	0	8,687,000	5,000,000	2	2	100	1	3	0	0	0	0	A
2	90	1	3	3	3,500,000	1,500,000	2	1	100	0	2	7.5	0	4	0	A
3	81	0	1	0	200,000	250,000	2	0	15	0	0	0	0	0	7	B
4	24	1	3	3	2,000,000	600,000	2	1	60	0	0	5	0	3	0	M
5	21	1	4	3	6,000,000	2,500,000	2	2	100	0	2	2	0	0	0	A
6	109	1	2	0	2,000,000	1,000,000	2	1	100	0	1	0	0	0	0	M
7	14	0	3	3	1,500,000	1,000,000	2	1	72	0	0	14	2	0	0	M

8	55	0	1	0	500,000	500,000	2	1	48	0	0	0	0	0	10	B
9	30	1	3	3	10,000,000	1,500,000	2	2	150	0	1	25	0	0	0	A
10	32	0	2	0	2,000,000	1,000,000	2	1	120	1	3	7	0	0	0	M
11	80	0	1	0	500,000	500,000	2	0	15	0	1	0	0	0	10	B
12	115	0	1	0	300,000	300,000	2	0	25	0	0	0	0	2	0	B

Lampiran 5. Range Normalisasi pada Perhitungan Manual

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
max	2	4	3	10000000	5000000	2	2	150	1	3	25	2	4	10
min	0	1	0	200000	250000	2	0	15	0	0	0	0	0	0
range	2	3	3	9800000	4750000	0	2	135	1	3	25	2	4	10

Lampiran 6. Hasil Normalisasi pada Perhitungan Manual

No	KK	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	Kelas
1	60	0.9000	0.9000	0.1000	0.7928	0.9000	0.1000	0.9000	0.6037	0.9000	0.9000	0.1000	0.1000	0.1000	0.1000	A
2	90	0.5000	0.6333	0.9000	0.3694	0.3105	0.1000	0.5000	0.6037	0.1000	0.6333	0.3400	0.1000	0.9000	0.1000	A
3	81	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.6600	B
4	24	0.5000	0.6333	0.9000	0.2469	0.1589	0.1000	0.5000	0.3667	0.1000	0.1000	0.2600	0.1000	0.7000	0.1000	M
5	21	0.5000	0.9000	0.9000	0.5735	0.4789	0.1000	0.9000	0.6037	0.1000	0.6333	0.1640	0.1000	0.1000	0.1000	A
6	109	0.5000	0.3667	0.1000	0.2469	0.2263	0.1000	0.5000	0.6037	0.1000	0.3667	0.1000	0.1000	0.1000	0.1000	M
7	14	0.1000	0.6333	0.9000	0.2061	0.2263	0.1000	0.5000	0.4378	0.1000	0.1000	0.5480	0.9000	0.1000	0.1000	M
8	55	0.1000	0.1000	0.1000	0.1245	0.1421	0.1000	0.5000	0.2956	0.1000	0.1000	0.1000	0.1000	0.1000	0.9000	B

9	30	0.5000	0.6333	0.9000	0.9000	0.3105	0.1000	0.9000	0.9000	0.1000	0.3667	0.9000	0.1000	0.1000	0.1000	A
10	32	0.1000	0.3667	0.1000	0.2469	0.2263	0.1000	0.5000	0.7222	0.9000	0.9000	0.3240	0.1000	0.1000	0.1000	M
11	80	0.1000	0.1000	0.1000	0.1245	0.1421	0.1000	0.1000	0.1000	0.1000	0.3667	0.1000	0.1000	0.1000	0.9000	B
12	115	0.1000	0.1000	0.1000	0.1082	0.1084	0.1000	0.1000	0.1593	0.1000	0.1000	0.1000	0.1000	0.5000	0.1000	B

Lampiran 7. Hasil Normalisasi Dikalikan Bobot pada Perhitungan Manual

No	KK	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	Kelas
1	60	0.0321	0.1286	0.0036	0.1133	0.0321	0.0071	0.0964	0.0216	0.0964	0.0643	0.0071	0.0071	0.0043	0.0029	A
2	90	0.0179	0.0905	0.0321	0.0528	0.0111	0.0071	0.0536	0.0216	0.0107	0.0452	0.0243	0.0071	0.0386	0.0029	A
3	81	0.0036	0.0143	0.0036	0.0143	0.0036	0.0071	0.0107	0.0036	0.0107	0.0071	0.0071	0.0071	0.0043	0.0189	B
4	24	0.0179	0.0905	0.0321	0.0353	0.0057	0.0071	0.0536	0.0131	0.0107	0.0071	0.0186	0.0071	0.0300	0.0029	M
5	21	0.0179	0.1286	0.0321	0.0819	0.0171	0.0071	0.0964	0.0216	0.0107	0.0452	0.0117	0.0071	0.0043	0.0029	A
6	109	0.0179	0.0524	0.0036	0.0353	0.0081	0.0071	0.0536	0.0216	0.0107	0.0262	0.0071	0.0071	0.0043	0.0029	M
7	14	0.0036	0.0905	0.0321	0.0294	0.0081	0.0071	0.0536	0.0156	0.0107	0.0071	0.0391	0.0643	0.0043	0.0029	M
8	55	0.0036	0.0143	0.0036	0.0178	0.0051	0.0071	0.0536	0.0106	0.0107	0.0071	0.0071	0.0071	0.0043	0.0257	B
9	30	0.0179	0.0905	0.0321	0.1286	0.0111	0.0071	0.0964	0.0321	0.0107	0.0262	0.0643	0.0071	0.0043	0.0029	A
10	32	0.0036	0.0524	0.0036	0.0353	0.0081	0.0071	0.0536	0.0258	0.0964	0.0643	0.0231	0.0071	0.0043	0.0029	M
11	80	0.0036	0.0143	0.0036	0.0178	0.0051	0.0071	0.0107	0.0036	0.0107	0.0262	0.0071	0.0071	0.0043	0.0257	B
12	115	0.0036	0.0143	0.0036	0.0155	0.0039	0.0071	0.0107	0.0057	0.0107	0.0071	0.0071	0.0071	0.0214	0.0029	B

Lampiran 8. Keterangan Nilai Atribut

No	Pendidikan	
	Nama	Nilai
1	TIDAK TAMAT SD	0
2	SD	0
3	SMP	1
4	SMA	1
5	DIPLOMA	1
6	S1	2
7	S2	2
8	S3	2

No	Status rumah	
	Nama	Nilai
1	TINGGAL DENGAN SAUDARA	0
2	TINGGAL DENGAN ANAK	0
3	TINGGAL DENGAN ORTU	1
4	RUMAH JURU KUNCI	1
5	MENEMPATI RUMAH ORANG	1
6	MILIK SENDIRI	2

No	Pekerjaan	
	Nama	Nilai
1	TIDAK BEKERJA	0
2	IBU RUMAH TANGGA	0
3	BURUH	1
4	TAMBAL BAN	1
5	BURUH TANI	1
6	KULI BANGUNAN	1
7	DAGANG KECIL	1
8	SWASTA	2
9	TUKANG KAYU	2
10	WIRASWASTA	2
11	PENJAHIT	2
12	BENGKEL	2
13	TUKANG BANGUNAN	2
14	INDUSTRI	2

No	Usaha lain	
	Nama	Nilai
1	TIDAK ADA	0
2	OJEK	1
3	KULI	1
4	BURUH	1
5	SOPIR	2
6	TOKO	2
7	MENJAHIT	2
8	BAJAK SAWAH	2
9	TERNAK	3
10	BERTANI	3
11	DAGANG	3

No	Keadaan rumah	
	Nama	Nilai
1	MEMPRIHATINKAN	0
2	SEDERHANA	1
3	BAGUS	2

No	Pekerjaan	
	Nama	Nilai
1	GURU SWASTA	2
2	KARYAWAN SWASTA	3
3	PETANI	3
4	PAMONG DESA	3
5	KARYAWAN HONORER	3
6	PETERNAK	3
7	PENSIUNAN	3
8	DAGANG	3
9	MEKANIK	3
10	WIRUSAHA	3
11	PNS	4
12	TNI	4
13	KARYAWAN BUMN	4
14	GURU PNS	4

Lampiran 9. Detail Hasil Pengujian Pengaruh Interval m dengan 121 Data

- Nilai Interval [1,2]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8918	1	0	1	0	0	0	1	1	0	1	0	0	0	0	0
P2	1.7752	9	0	1	1	0	1	1	0	1	0	0	1	1	1	0
P3	1.032	4	0	0	1	0	0	0	1	1	0	0	0	0	1	1
P4	1.3315	9	0	1	1	1	1	1	1	1	1	1	0	1	0	1
P5	1.6651	9	1	0	0	0	1	1	0	1	0	1	0	1	0	0
P6	1.3724	3	1	0	1	1	1	1	0	1	0	1	0	1	0	1
P7	1.6477	9	0	1	0	0	1	1	1	0	0	1	0	0	1	0
P8	1.6058	4	0	1	0	1	1	1	1	0	1	1	0	1	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2992	5	1	0	0	1	1	1	1	0	1	0	1	1	0	0
P2	1.9253	9	1	1	0	1	0	0	1	1	1	1	1	1	0	1
P3	1.4882	3	0	0	0	0	0	1	1	0	1	0	1	0	0	0
P4	1.4379	9	1	0	0	0	0	1	1	1	1	1	1	1	0	1
P5	1.0248	6	0	0	1	0	1	1	1	1	0	1	1	0	0	0
P6	1.7045	3	0	0	0	0	0	1	0	1	0	1	1	1	1	1
P7	1.9084	4	1	1	0	0	0	1	0	0	0	1	0	1	1	1
P8	1.3815	5	0	0	1	1	1	0	1	0	0	0	1	0	0	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5867	8	0	1	0	0	1	0	0	0	1	1	1	1	1	0
P2	1.5439	7	0	0	0	1	1	1	0	0	0	0	0	0	1	0
P3	1.0728	1	1	0	0	1	1	0	0	0	1	1	1	1	1	1
P4	1.6444	5	1	0	0	1	0	1	1	1	1	1	1	1	1	0
P5	1.745	4	1	1	0	0	0	1	0	1	1	1	0	0	1	0
P6	1.619	9	1	1	1	1	1	0	1	1	1	0	0	1	1	0
P7	1.6837	5	0	0	1	1	1	0	0	1	0	0	0	0	1	1
P8	1.6397	2	1	0	0	0	0	1	1	1	1	0	1	0	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2835	4	0	0	1	1	0	0	1	1	0	1	0	1	0	1
P2	1.9234	9	0	0	1	0	0	0	0	1	0	0	1	0	0	0
P3	1.9038	4	1	1	0	1	1	1	0	1	0	1	0	1	0	0
P4	1.2208	7	0	1	1	0	1	0	0	0	0	0	0	0	0	0
P5	1.8524	2	0	1	1	0	1	0	0	1	0	1	0	0	1	1
P6	1.2541	4	0	1	0	1	0	0	0	1	1	1	1	1	0	0
P7	1.8206	7	1	0	0	1	0	0	1	0	1	0	1	1	1	1
P8	1.3218	2	1	1	1	0	0	0	0	0	0	1	1	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0394	2	0	0	1	1	1	1	1	0	0	0	1	0	0	1
P2	1.9539	1	1	0	1	1	0	0	1	0	1	1	0	1	1	1
P3	1.0694	6	0	1	1	1	0	1	1	0	0	1	0	0	1	0

P4	1.6666	8	1	1	0	0	0	1	1	1	1	0	1	1	0	0
P5	1.2409	2	1	1	1	1	0	0	1	0	1	0	0	1	1	0
P6	1.6546	7	0	1	1	0	1	0	0	1	0	1	1	1	1	1
P7	1.9312	8	1	0	1	1	0	0	1	0	1	1	1	1	0	0
P8	1.5244	1	1	0	1	1	0	1	0	0	1	0	0	1	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.6477	9	0	1	0	0	0	0	0	0	0	1	0	0	0	0
Fold#2	1.9084	4	0	0	0	0	1	1	0	1	0	0	1	0	0	0
Fold#3	1.745	4	0	0	0	0	1	1	0	1	0	0	0	0	0	0
Fold#4	1.2208	7	0	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#5	1.0694	6	1	0	0	0	1	0	0	0	0	0	1	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.6477	9	2-10	65.32%
2	Fold #2	1.9084	4	5-6-8-11	75.17%
3	Fold #3	1.745	4	5-6-8	76.44%
4	Fold #4	1.2208	7	4-11	89.50%
5	Fold #5	1.0694	6	1-5-11	76.50%

- Nilai Interval [2,4]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	2.4597	6	1	1	0	0	0	0	1	1	1	0	0	0	0	0
P2	3.9265	10	1	1	1	0	1	0	0	0	0	1	1	0	1	0
P3	2.5858	6	1	0	0	1	1	1	0	0	1	0	0	1	0	1
P4	3.454	8	1	1	1	1	1	0	0	1	1	1	0	0	0	0
P5	2.8438	1	1	1	0	1	1	0	0	0	1	0	0	0	1	0
P6	2.4723	1	0	0	1	0	0	1	1	1	1	0	1	1	1	0
P7	3.0083	2	0	0	1	1	0	1	1	0	0	1	0	0	1	1
P8	2.5929	3	1	1	1	0	1	1	0	0	0	1	0	0	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	3.0048	4	0	1	0	1	1	0	1	0	0	0	0	1	0	0
P2	3.7373	10	0	1	0	1	0	0	1	1	0	1	0	1	1	0
P3	3.6296	9	0	1	0	1	1	1	1	0	0	1	1	0	0	0
P4	3.9733	10	1	0	0	1	0	1	0	0	0	0	1	1	0	1
P5	3.5276	7	0	0	0	0	0	0	1	0	1	1	0	1	0	0
P6	3.2733	4	1	0	0	0	0	1	1	1	0	1	1	1	0	1
P7	2.6244	2	1	0	0	1	0	0	1	1	0	1	0	0	0	1
P8	3.7816	9	1	1	0	1	1	0	1	1	1	0	0	0	0	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	2.3213	3	1	0	1	0	0	0	0	0	1	0	0	0	1	1
P2	3.1007	6	0	1	1	0	0	0	1	1	0	1	0	1	0	0

P3	3.9345	4	1	1	0	1	0	1	1	1	1	0	0	0	0	0
P4	2.2485	8	1	0	0	0	1	1	1	0	1	1	1	1	0	1
P5	2.7083	8	0	0	1	1	0	1	0	1	0	0	1	0	0	0
P6	3.9966	7	1	0	0	0	1	1	0	1	0	1	0	0	1	0
P7	3.9675	5	0	1	0	1	1	1	1	0	0	1	0	1	0	0
P8	2.6836	10	0	1	1	1	0	1	0	0	1	1	1	0	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	3.9141	9	1	1	1	1	0	1	0	1	0	1	1	0	0	1
P2	3.4445	5	0	0	0	1	0	0	1	1	0	1	1	0	0	1
P3	2.5759	10	0	0	1	0	1	0	0	1	1	1	1	1	1	0
P4	2.4358	9	0	1	0	0	1	1	1	0	0	0	0	0	1	0
P5	3.4655	5	1	1	0	0	1	0	0	1	1	1	0	0	1	1
P6	3.917	8	1	1	1	1	0	1	0	0	1	1	0	0	0	1
P7	3.2747	9	1	1	1	0	0	1	0	0	1	1	0	0	1	0
P8	2.7416	5	1	0	0	1	0	1	0	0	0	1	0	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	2.0239	7	1	0	1	1	0	0	1	0	0	0	1	1	0	0
P2	3.2166	3	0	1	1	1	0	1	0	1	0	1	0	0	1	1
P3	3.561	4	1	0	1	0	0	1	0	0	1	0	1	0	0	1
P4	3.8738	8	1	1	1	0	0	0	1	1	0	0	0	1	0	0
P5	3.42	7	1	0	1	0	1	1	0	1	1	0	1	0	0	0
P6	3.391	6	1	0	1	1	0	0	1	1	1	1	0	0	0	1
P7	2.9915	3	1	1	0	0	0	1	0	1	0	1	1	1	0	0
P8	2.5252	8	1	1	1	1	0	1	1	1	0	1	0	1	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	2.5858	6	0	0	0	1	1	0	0	0	0	0	1	0	0	1
Fold#2	3.0048	4	0	0	0	1	0	0	0	0	1	0	0	0	1	0
Fold#3	2.3213	3	1	0	1	0	0	0	1	0	0	1	0	0	0	0
Fold#4	3.9141	9	0	0	0	0	0	0	0	0	0	1	1	0	0	1
Fold#5	2.5252	8	0	1	0	0	0	0	1	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	2.5858	6	4-5-11-14	88.78%
2	Fold #2	3.0048	4	4-9-13	80.58%
3	Fold #3	2.3213	3	1-3-7-10	69.82%
4	Fold #4	3.9141	9	10-11-14	67.38%
5	Fold #5	2.5252	8	2-7	75.64%

- Nilai Interval [4,6]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	5.3736	1	1	1	0	1	0	0	0	0	1	1	0	0	1	1

P2	5.1448	3	1	1	1	0	1	1	0	1	0	1	0	1	0	1
P3	5.8009	10	1	1	0	0	0	0	1	0	1	1	0	0	0	0
P4	5.2014	5	0	1	0	0	1	1	0	0	0	1	1	1	0	0
P5	4.5288	8	1	0	0	1	1	0	0	0	1	1	0	1	0	0
P6	5.7911	5	0	1	0	0	0	0	1	1	1	1	1	1	1	0
P7	4.9497	10	0	1	1	0	1	1	1	0	1	1	1	0	1	0
P8	4.207	9	0	1	1	0	0	0	1	0	0	1	1	1	0	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	4.9331	9	1	1	1	0	0	0	1	0	1	1	1	0	1	1
P2	4.771	9	0	1	0	1	1	0	0	1	1	1	1	0	1	0
P3	4.3387	1	0	0	0	0	1	0	1	1	1	0	1	0	0	0
P4	4.4809	3	1	0	1	0	0	0	1	0	0	1	0	1	0	1
P5	4.99	5	1	1	0	1	1	1	1	1	1	1	0	1	1	1
P6	5.0066	6	0	1	1	0	1	1	0	1	1	1	1	0	0	0
P7	4.498	5	1	0	0	1	0	1	1	1	1	1	0	1	1	0
P8	5.1055	8	0	1	0	1	1	0	1	0	1	0	0	1	0	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	5.3699	7	0	1	1	0	0	1	0	1	0	1	0	1	0	1
P2	4.8796	10	0	0	1	0	0	1	0	1	0	0	0	1	1	1
P3	4.8952	9	0	1	0	0	0	0	0	0	0	0	1	0	1	0
P4	5.9121	3	0	1	1	1	1	1	1	0	0	0	1	0	1	1
P5	5.0028	10	1	0	0	0	0	1	0	1	0	1	1	1	0	0
P6	4.0314	2	1	0	0	0	1	1	1	1	0	0	1	0	1	1
P7	5.2847	7	0	0	0	1	0	1	1	1	0	0	1	0	1	1
P8	4.1947	1	0	1	0	1	1	1	1	1	0	1	1	1	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	4.3104	2	1	1	0	0	0	1	0	1	0	0	1	1	0	1
P2	5.0966	7	0	1	0	1	1	0	1	1	0	0	0	0	0	1
P3	4.6472	7	1	1	0	0	0	1	1	0	1	0	0	0	0	1
P4	4.6688	10	0	0	1	0	1	1	1	1	0	1	1	1	0	1
P5	4.7575	9	1	0	1	0	1	1	0	0	1	1	1	0	1	1
P6	5.2383	6	0	0	0	1	0	0	0	0	0	1	1	1	0	0
P7	4.3958	4	1	1	1	1	1	1	1	0	0	1	1	0	1	1
P8	4.7163	7	0	0	1	0	0	0	0	0	0	1	0	0	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	5.4076	9	0	1	1	0	0	1	1	1	0	1	0	1	1	1
P2	5.0724	5	1	1	1	0	0	0	1	0	1	0	1	0	1	1
P3	5.7634	8	1	1	0	0	0	1	0	0	0	0	1	1	1	1
P4	4.4055	6	0	1	0	0	1	1	0	1	1	0	1	1	0	0
P5	5.489	10	0	0	1	1	1	0	0	0	0	0	1	0	1	0
P6	5.9511	6	1	0	0	1	0	1	0	0	0	0	1	1	1	0
P7	4.4439	9	0	1	1	0	1	1	0	1	0	1	1	1	1	0
P8	5.2755	3	1	1	0	0	1	0	0	0	0	0	0	0	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	5.7911	5	0	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#2	4.4809	3	0	1	0	1	0	0	0	0	0	0	0	1	1	0
Fold#3	5.3699	7	0	0	0	0	0	0	1	1	0	0	0	0	0	0
Fold#4	4.6688	10	0	0	0	1	1	0	0	0	0	0	1	0	0	0
Fold#5	5.2755	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	5.7911	5	4-11	89.39%
2	Fold #2	4.4809	3	2-4-12-13	81.97%
3	Fold #3	5.3699	7	7-8	66.10%
4	Fold #4	4.6688	10	4-5-11	87.56%
5	Fold #5	5.2755	3	6	52.61%

- Nilai Interval [6,8]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	6.7309	5	0	1	1	1	1	0	0	1	0	1	1	0	0	0
P2	6.573	8	1	0	0	1	0	1	1	0	1	0	1	0	1	1
P3	7.3017	5	0	0	1	0	1	0	0	0	1	1	0	0	0	0
P4	7.163	9	0	1	0	1	1	0	0	0	0	0	0	0	1	0
P5	6.9125	8	0	1	0	1	0	1	0	1	1	1	0	0	0	0
P6	6.5174	10	1	1	1	1	1	1	0	1	0	0	1	1	1	1
P7	7.0064	8	0	1	0	0	1	0	0	1	1	0	1	1	0	1
P8	6.1596	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	7.7951	9	0	1	0	1	0	1	0	1	1	0	0	0	1	0
P2	7.1551	5	0	1	0	0	0	0	0	0	0	1	0	1	0	0
P3	7.3641	8	1	1	1	0	1	0	0	1	0	0	1	1	1	0
P4	7.1347	6	1	0	1	0	1	1	1	1	1	1	0	0	0	1
P5	6.5239	2	1	1	1	1	1	0	0	1	1	1	1	0	0	1
P6	7.8557	7	0	1	0	0	1	1	1	1	0	1	1	1	0	1
P7	7.8763	3	1	1	0	0	1	1	0	0	1	1	0	0	0	1
P8	7.7285	5	0	0	0	1	0	1	1	1	1	1	1	1	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	6.0751	8	0	1	0	1	1	1	0	1	1	0	0	1	1	1
P2	7.272	7	0	1	0	1	1	1	1	0	0	0	0	1	0	0
P3	7.0013	3	0	0	1	1	1	1	0	1	0	0	1	1	0	1
P4	7.954	4	0	1	0	0	0	0	0	1	1	1	1	0	0	1
P5	6.594	5	1	1	1	0	1	0	0	1	0	1	1	0	1	1
P6	7.1876	9	1	0	0	0	0	0	0	1	1	0	1	1	0	1
P7	6.4899	7	1	1	1	0	1	0	0	0	1	0	1	0	1	0

P8	6.0553	9	0	1	1	1	0	0	1	1	1	0	0	0	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	7.067	6	0	0	0	0	1	1	0	1	0	0	0	0	0	0
P2	7.5814	3	0	1	1	0	0	0	1	1	1	0	1	0	1	1
P3	6.5635	9	1	0	0	0	1	0	0	0	1	1	0	0	0	1
P4	6.8187	6	1	1	0	1	0	0	0	0	1	1	1	1	1	1
P5	6.9835	1	0	0	0	0	1	0	0	1	1	0	0	1	1	0
P6	7.523	7	1	0	1	0	1	0	0	0	0	0	1	0	1	0
P7	6.8878	8	1	1	0	1	1	0	1	1	1	0	0	0	0	1
P8	6.0872	5	0	1	0	0	1	0	1	0	1	1	0	1	0	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	6.2326	3	0	1	1	1	0	1	0	0	1	1	0	1	0	0
P2	6.6283	4	1	1	0	1	0	0	1	1	0	1	0	1	0	1
P3	7.4064	4	1	0	1	0	1	1	1	0	1	1	1	1	1	0
P4	7.1655	1	0	1	1	1	1	1	1	1	0	0	1	1	0	0
P5	7.6899	10	1	1	1	0	0	0	0	0	0	1	1	0	1	1
P6	7.901	1	1	1	0	0	0	1	1	0	0	1	0	1	1	0
P7	6.8222	10	1	1	0	1	0	1	0	0	0	1	1	0	1	0
P8	6.8495	3	0	1	1	1	0	1	1	1	1	0	1	1	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	7.0064	8	0	1	0	0	0	0	0	1	0	1	0	0	0	0
Fold#2	7.7951	9	0	1	0	1	0	0	0	1	0	0	0	0	0	0
Fold#3	7.0013	3	1	1	0	0	1	0	0	0	0	0	0	0	0	0
Fold#4	6.8878	8	0	1	0	1	0	0	0	1	0	0	0	0	0	0
Fold#5	6.6283	4	0	0	0	0	1	0	0	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	7.0064	8	2-8-10	77.22%
2	Fold #2	7.7951	9	2-4-8	84.30%
3	Fold #3	7.0013	3	1-2-5	75.73%
4	Fold #4	6.8878	8	2-4-8	85.59%
5	Fold #5	6.6283	4	5	62.19%

- Nilai interval [8,10]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	9.1754	9	0	0	1	1	0	0	0	0	0	1	1	1	1	1
P2	9.5512	10	0	1	1	0	0	0	1	1	1	0	1	1	1	1
P3	9.3075	4	0	0	0	0	0	1	1	1	0	0	0	1	0	0
P4	9.4707	10	1	0	0	1	1	1	0	0	1	0	0	1	0	0
P5	9.8057	6	0	1	0	0	0	1	0	0	0	0	1	0	1	1
P6	8.3956	4	0	0	0	0	1	1	1	1	0	0	1	0	1	1

P7	8.1719	6	1	0	0	0	1	1	0	1	0	1	1	1	1	0
P8	9.5034	4	1	0	0	1	0	0	0	1	0	1	0	0	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	8.4069	7	1	1	0	0	1	0	0	1	1	0	0	1	0	0
P2	8.0945	10	1	1	0	1	0	1	1	0	0	1	0	0	0	0
P3	9.91	1	0	0	0	0	1	1	0	1	1	1	1	0	0	1
P4	9.8221	4	1	0	0	1	1	0	1	1	1	1	0	1	0	1
P5	9.7948	6	0	0	0	1	0	1	0	1	1	1	1	1	1	1
P6	8.6599	3	0	0	1	0	1	0	0	0	1	0	0	1	1	0
P7	9.6754	5	0	1	0	1	0	1	1	1	1	1	0	0	0	0
P8	8.4887	8	1	0	0	0	0	0	0	0	0	0	0	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	8.735	3	0	1	0	0	1	0	0	1	0	1	0	1	1	1
P2	8.642	10	1	1	1	1	0	0	0	1	1	1	1	0	0	0
P3	8.4174	1	0	1	0	0	1	1	1	0	1	0	1	1	0	0
P4	8.366	2	0	1	1	1	1	1	0	1	1	1	0	1	0	0
P5	8.0911	8	1	0	1	0	1	1	1	1	0	1	1	1	1	1
P6	8.6542	6	1	1	1	0	0	0	1	1	1	0	1	1	0	1
P7	9.8584	10	1	1	0	0	1	0	0	1	0	1	1	0	0	1
P8	9.7667	10	0	0	1	1	0	0	1	1	0	0	1	1	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	8.835	7	1	1	0	0	1	0	1	1	0	0	1	0	1	0
P2	8.0116	6	0	1	1	0	0	0	1	0	1	1	1	1	1	1
P3	9.3535	5	0	0	0	0	0	1	0	1	0	0	0	0	1	0
P4	8.0528	10	0	1	0	0	1	0	0	1	0	1	0	0	1	0
P5	9.0587	3	1	0	0	0	0	0	1	0	1	0	1	0	1	0
P6	9.3102	7	1	1	0	0	1	0	1	1	1	1	0	1	1	0
P7	8.7764	10	1	1	0	0	0	0	1	0	1	1	1	1	1	0
P8	9.2603	10	0	0	0	0	1	0	1	1	0	0	1	0	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	8.5962	5	0	0	0	1	1	0	0	0	1	1	0	1	0	1
P2	8.9778	3	1	1	0	1	1	1	1	1	0	1	1	1	1	1
P3	8.1006	7	0	1	1	0	0	0	1	1	1	0	1	1	0	1
P4	8.9033	3	0	0	0	1	1	1	0	0	1	1	1	0	0	1
P5	9.6655	3	0	1	1	1	0	0	0	1	0	0	1	1	0	1
P6	8.4849	4	1	0	1	1	0	1	0	0	0	0	0	0	1	1
P7	8.5551	7	1	1	1	0	0	0	0	1	1	1	1	0	0	1
P8	8.5638	3	0	0	1	1	1	1	0	1	1	1	1	0	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	8.1719	6	1	0	1	0	0	0	1	1	0	0	0	0	0	0
Fold#2	8.0945	10	0	1	0	0	0	0	0	0	0	0	0	1	0	0
Fold#3	8.0911	8	0	0	0	1	0	0	1	0	0	0	0	0	0	0

Fold#4	8.835	7	0	1	0	0	1	0	0	0	0	1	0	0	0	0
Fold#5	8.9033	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	8.1719	6	1-3-7-8	69.22%
2	Fold #2	8.0945	10	2-12	79.02%
3	Fold #3	8.0911	8	4-7	84.51%
4	Fold #4	8.835	7	2-5-10	73.57%
5	Fold #5	8.9033	3	6	55.92%

Lampiran 10. Detail Hasil Pengujian Pengaruh Interval k dengan 121 Data

- Nilai interval [1,10]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.775	5	0	0	0	0	0	0	0	0	1	0	1	1	1	0
P2	1.9677	6	0	0	1	0	1	1	1	0	1	0	0	0	0	0
P3	1.0693	5	1	0	0	1	1	1	1	1	0	0	0	0	1	0
P4	1.6242	4	0	0	1	1	0	1	1	1	1	1	0	0	1	1
P5	1.4347	2	1	0	1	1	0	1	0	0	1	1	0	1	0	0
P6	1.8511	7	1	1	0	1	1	1	1	1	0	1	0	0	0	1
P7	1.3249	10	0	1	1	0	0	1	0	0	0	0	0	1	1	0
P8	1.4464	3	0	1	1	0	1	0	1	0	1	0	0	1	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9973	2	1	1	1	0	0	0	1	1	0	1	0	0	0	0
P2	1.2539	10	0	1	0	1	1	1	1	1	0	1	1	1	1	1
P3	1.5323	10	0	0	1	1	1	1	0	1	1	1	1	0	1	0
P4	1.4563	10	0	1	0	1	1	0	0	1	0	1	0	1	0	1
P5	1.905	5	1	1	0	1	1	0	1	0	1	0	1	0	0	0
P6	1.7939	9	1	0	0	1	0	0	1	1	0	1	0	1	0	0
P7	1.8759	5	0	1	0	0	0	0	1	0	0	0	1	1	0	1
P8	1.9309	7	0	1	1	0	0	1	1	0	0	0	0	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3138	10	1	0	0	1	1	1	1	0	0	0	0	0	0	0
P2	1.8442	4	0	1	1	1	1	0	0	0	1	0	1	1	0	0
P3	1.31	8	0	1	1	0	0	1	1	0	1	0	1	0	1	1
P4	1.0735	10	0	0	1	0	0	1	1	1	1	0	1	1	1	0
P5	1.2704	3	0	0	1	1	0	1	1	0	1	0	0	0	0	0
P6	1.0649	8	0	1	0	1	0	1	0	0	1	0	1	0	0	0
P7	1.7344	5	0	1	1	1	1	0	1	0	1	1	0	1	1	0
P8	1.6811	3	0	1	1	1	1	0	1	1	1	1	1	1	0	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4968	10	1	0	0	1	0	0	0	0	1	0	0	1	0	0
P2	1.5067	3	0	1	1	0	0	0	0	1	1	0	1	1	0	0

P3	1.4483	9	1	1	1	0	0	0	1	0	1	1	0	0	0	0
P4	1.8302	4	1	0	1	0	0	1	0	0	0	0	0	1	1	0
P5	1.7056	9	0	1	0	0	0	1	0	1	0	1	1	1	0	1
P6	1.3996	8	1	1	0	1	1	0	0	0	1	1	0	0	0	1
P7	1.265	6	1	0	0	0	0	1	1	1	0	1	0	1	1	0
P8	1.7122	10	0	1	1	1	1	0	0	1	1	1	1	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3854	3	0	1	0	0	1	0	0	0	0	1	1	1	0	0
P2	1.8459	4	1	0	1	1	0	1	0	1	0	0	1	0	0	0
P3	1.0351	3	1	1	1	0	1	1	0	0	1	1	1	0	0	0
P4	1.4178	5	1	1	0	0	1	1	0	1	0	1	1	0	1	1
P5	1.9716	7	0	1	0	1	1	0	0	1	0	0	0	1	1	0
P6	1.1569	3	1	0	1	1	0	0	0	0	1	0	1	1	0	1
P7	1.7223	5	1	1	1	1	1	1	1	0	1	0	1	0	0	1
P8	1.1073	8	1	1	1	0	0	1	1	0	0	0	0	1	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.0693	5	0	0	0	0	1	0	0	1	0	0	1	0	0	0
Fold#2	1.8759	5	0	1	0	0	0	0	1	0	0	0	0	1	0	0
Fold#3	1.31	8	1	0	0	0	1	0	0	0	0	0	1	0	1	0
Fold#4	1.8302	4	0	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#5	1.1569	3	0	1	0	0	0	1	0	0	1	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.0693	5	5-8-11	77.88%
2	Fold #2	1.8759	5	2-7-12	79.34%
3	Fold #3	1.31	8	1-5-11-13	80.48%
4	Fold #4	1.8302	4	4-11	90.17%
5	Fold #5	1.1569	3	2-6-9	80.48%

- Nilai interval [11,20]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3029	13	1	1	0	1	1	0	0	1	0	0	0	0	1	0
P2	1.2358	14	0	1	1	0	0	1	0	0	1	1	0	1	1	0
P3	1.1649	18	1	0	0	1	0	0	0	0	1	1	0	0	1	1
P4	1.0525	12	1	0	1	1	0	0	0	0	1	0	1	0	0	0
P5	1.0398	19	1	0	0	0	1	1	0	0	1	0	0	0	0	1
P6	1.2137	16	1	1	1	1	0	0	1	1	0	0	1	0	0	0
P7	1.3993	12	0	0	1	1	0	1	0	1	0	1	1	1	0	1
P8	1.2475	16	1	0	0	0	0	0	1	0	0	1	0	0	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7838	19	0	1	0	0	1	1	1	1	1	0	0	1	0	1

P2	1.6037	16	1	1	0	0	1	1	1	1	0	0	0	1	1	1
P3	1.1223	20	1	1	1	0	0	0	0	1	1	1	1	0	1	0
P4	1.1927	11	0	0	1	0	1	1	0	0	1	0	0	1	1	0
P5	1.9376	12	0	0	1	0	0	0	0	1	0	1	0	0	0	0
P6	1.9962	20	0	0	1	0	0	1	1	0	1	0	0	1	0	1
P7	1.5577	11	0	1	0	0	0	1	1	1	0	1	0	1	0	0
P8	1.4466	11	1	0	0	0	0	1	1	0	1	1	1	1	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4354	16	1	1	1	1	1	0	1	0	0	1	1	0	0	1
P2	1.7849	18	0	1	0	1	0	1	0	0	1	0	1	1	0	1
P3	1.0282	19	0	1	1	0	0	0	0	0	1	1	0	1	1	0
P4	1.9225	11	1	1	1	0	1	1	0	1	1	0	1	1	1	1
P5	1.2881	14	1	1	1	1	1	0	1	1	0	0	0	0	1	0
P6	1.5631	15	1	1	1	0	0	1	0	1	1	0	1	1	1	0
P7	1.686	14	1	1	1	0	0	0	0	1	0	1	1	0	0	1
P8	1.3997	13	0	0	0	1	0	0	0	0	1	1	1	1	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0058	14	0	1	1	1	1	0	0	0	1	0	0	0	0	1
P2	1.7352	19	1	0	1	0	0	0	1	1	1	1	0	0	1	0
P3	1.509	19	1	1	0	0	0	0	1	1	0	1	0	0	0	1
P4	1.0074	18	0	0	0	1	1	0	1	1	0	0	0	1	0	1
P5	1.485	20	0	1	1	1	0	0	1	0	0	0	0	1	0	0
P6	1.1203	15	0	0	0	0	0	0	1	0	1	1	0	1	0	1
P7	1.4807	16	1	1	1	0	1	1	1	1	1	1	0	1	1	1
P8	1.7938	16	1	0	1	1	0	1	0	1	0	0	1	1	0	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4509	13	1	1	0	0	1	1	0	0	1	1	1	0	0	1
P2	1.2069	15	0	0	0	0	0	1	0	1	0	1	0	0	0	0
P3	1.1301	14	0	1	0	0	1	0	0	1	1	1	0	1	0	1
P4	1.346	19	1	1	0	0	0	0	0	1	1	0	0	1	1	0
P5	1.5837	11	0	0	0	0	0	0	1	0	0	1	0	1	1	0
P6	1.5469	19	0	0	0	0	1	0	1	0	0	1	1	0	1	1
P7	1.5377	12	1	1	0	1	1	0	0	1	1	0	1	1	1	1
P8	1.1825	13	0	1	1	0	1	1	1	1	1	1	0	1	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.0398	19	0	0	0	1	0	1	0	0	1	0	0	0	0	0
Fold#2	1.1927	11	1	0	0	1	0	0	0	0	1	0	0	0	0	0
Fold#3	1.9225	11	0	1	0	1	0	0	0	0	0	0	0	1	0	0
Fold#4	1.0058	14	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Fold#5	1.1301	14	0	0	0	0	0	0	0	0	0	0	1	0	0	1

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.0398	19	4-6-9	80.17%
2	Fold #2	1.1927	11	1-4-9	80.92%
3	Fold #3	1.9225	11	2-4-12	85.76%
4	Fold #4	1.0058	14	3	25.36%
5	Fold #5	1.1301	14	11-14	65.53%

- Nilai interval [21,40]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0314	21	0	0	1	0	1	1	1	0	0	1	0	0	0	1
P2	1.7295	26	1	0	0	1	0	0	0	0	0	1	0	1	0	0
P3	1.0165	32	0	0	1	1	0	0	0	1	0	1	0	1	0	0
P4	1.2669	36	0	0	1	0	1	0	1	0	0	1	1	1	0	1
P5	1.0534	23	0	0	0	1	0	1	0	0	1	1	1	0	1	0
P6	1.1379	22	1	0	1	0	0	0	0	0	0	1	1	1	1	0
P7	1.1685	28	1	0	1	1	0	1	0	1	1	0	1	0	0	1
P8	1.8958	32	1	1	1	0	0	0	1	1	0	1	1	0	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5789	36	0	1	0	0	0	0	1	0	1	0	0	0	1	1
P2	1.5876	22	0	1	0	1	0	0	1	1	1	1	0	1	0	1
P3	1.36	34	1	1	1	1	0	0	0	0	1	1	0	0	0	1
P4	1.3736	28	1	1	1	0	1	1	0	0	1	1	0	1	0	0
P5	1.8281	36	0	1	1	1	0	1	0	1	0	1	1	0	1	1
P6	1.2304	34	1	1	1	0	0	0	1	1	1	1	1	1	0	1
P7	1.6863	23	0	1	1	0	0	1	0	0	1	0	0	1	1	1
P8	1.9652	24	1	0	0	0	0	0	1	0	1	1	1	0	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2949	33	1	0	0	0	0	0	0	1	1	1	0	1	0	1
P2	1.0108	34	1	1	0	0	1	1	0	1	1	1	0	0	1	0
P3	1.7514	29	1	1	0	1	0	1	1	1	1	0	0	0	1	0
P4	1.9874	36	0	1	1	1	1	1	0	1	1	1	1	0	1	0
P5	1.6134	37	0	1	0	1	1	0	1	1	1	1	0	1	1	0
P6	1.6446	36	1	1	1	0	0	1	1	1	0	0	1	1	1	0
P7	1.7486	36	0	1	1	0	1	0	0	0	0	1	1	0	0	1
P8	1.6889	24	1	0	1	0	0	1	0	1	1	0	0	0	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5605	39	1	0	0	1	0	0	1	1	0	1	1	1	1	0
P2	1.6333	33	0	0	0	1	1	0	0	1	1	0	0	1	0	1
P3	1.7921	21	1	0	0	1	1	1	0	0	0	0	1	1	0	0
P4	1.4563	25	1	1	0	0	0	0	1	0	1	1	1	0	0	0
P5	1.403	29	1	1	1	0	1	1	0	1	1	0	1	1	1	1
P6	1.2622	33	1	1	1	1	1	1	1	1	1	0	1	1	1	0

P7	1.9492	23	1	1	1	0	0	1	0	1	0	0	0	0	0	0
P8	1.09	37	1	1	0	0	0	1	1	1	0	0	1	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4419	26	1	1	1	1	1	1	1	1	0	1	0	1	1	0
P2	1.3596	29	0	1	1	1	1	1	1	1	0	0	1	0	0	1
P3	1.2067	26	1	1	0	1	0	1	0	0	1	1	1	0	1	1
P4	1.1776	23	1	0	1	1	0	0	1	0	0	0	1	1	0	1
P5	1.7458	38	1	0	1	1	1	1	0	0	1	1	1	0	0	0
P6	1.2541	26	0	0	0	1	0	0	0	0	0	0	0	0	0	1
P7	1.3092	26	1	0	1	1	0	0	0	0	0	0	0	1	1	0
P8	1.0256	39	1	1	0	1	0	0	1	0	0	0	0	1	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.7295	26	0	0	0	0	1	0	0	0	0	0	1	0	0	0
Fold#2	1.36	34	0	1	1	0	0	0	1	0	0	0	0	0	0	0
Fold#3	1.9874	36	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Fold#4	1.9492	23	0	1	0	0	0	0	1	0	0	0	0	0	0	0
Fold#5	1.2541	26	0	0	1	1	0	0	0	0	1	0	0	0	1	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.7295	26	5-11	77.22%
2	Fold #2	1.36	34	2-3-7	79.74%
3	Fold #3	1.9874	36	5	68.64%
4	Fold #4	1.9492	23	2-7	77.07%
5	Fold #5	1.2541	26	3-4-9-13	85.05%

- Nilai interval [41,60]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0364	56	0	0	1	0	0	0	1	1	1	0	0	0	1	1
P2	1.1164	43	1	0	0	0	0	1	0	0	0	0	1	1	0	0
P3	1.0151	54	0	0	0	1	0	0	0	1	0	1	0	0	1	0
P4	1.2424	51	1	0	1	0	1	1	1	1	0	1	1	1	1	1
P5	1.6014	48	1	0	0	0	0	0	1	1	1	0	1	0	1	1
P6	1.2939	51	1	0	0	0	1	1	1	0	1	0	1	1	0	0
P7	1.5431	43	0	0	1	1	1	0	1	0	0	0	0	0	1	1
P8	1.1566	59	0	1	0	0	1	0	0	1	1	1	1	1	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5821	46	0	0	0	0	1	1	0	1	1	1	1	0	0	0
P2	1.9859	56	1	1	0	1	0	1	1	1	0	1	0	0	0	1
P3	1.8853	54	1	1	1	0	0	1	0	0	0	0	1	1	1	1
P4	1.1969	51	0	1	0	0	0	1	0	0	0	0	1	0	1	0
P5	1.6101	53	0	1	1	1	1	0	1	1	0	0	1	1	0	0

P6	1.829	44	0	1	1	0	1	0	1	0	0	0	1	0	1	0
P7	1.8766	49	1	0	1	1	1	1	1	1	0	0	1	1	0	1
P8	1.3525	42	1	0	1	0	0	1	0	1	1	0	1	0	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2422	46	1	1	0	0	0	0	1	1	0	0	1	0	1	1
P2	1.6602	46	0	1	0	1	0	0	1	0	1	0	1	1	0	0
P3	1.04	43	0	1	0	0	1	1	0	1	0	1	1	0	1	0
P4	1.1066	42	0	1	1	0	0	1	1	0	0	1	0	0	0	0
P5	1.945	52	0	1	1	1	1	1	0	1	0	0	0	1	1	0
P6	1.9537	51	0	1	1	0	1	0	1	0	1	1	0	1	1	1
P7	1.9378	49	0	1	1	0	0	0	0	0	1	0	1	1	0	0
P8	1.6923	55	0	0	0	0	1	0	0	0	1	1	0	1	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6604	41	0	0	0	1	1	0	1	1	0	0	0	0	1	0
P2	1.5878	41	0	1	1	0	1	1	0	1	0	1	0	1	1	1
P3	1.5943	53	0	1	1	1	0	1	0	1	0	0	0	0	0	0
P4	1.5346	50	0	1	1	0	1	1	1	0	0	0	0	1	0	0
P5	1.6726	60	1	1	0	1	1	0	0	0	0	1	1	0	1	1
P6	1.2663	59	0	1	0	0	0	1	0	0	1	1	1	1	1	0
P7	1.1473	55	0	0	1	0	0	1	1	0	0	0	1	1	1	1
P8	1.5606	43	0	0	1	0	0	1	1	0	1	1	1	0	0	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1906	55	0	1	0	1	1	1	0	1	1	1	0	1	1	0
P2	1.3016	52	0	0	1	0	1	0	1	1	1	1	1	0	1	1
P3	1.4907	41	1	0	1	0	0	1	1	1	1	0	0	0	0	1
P4	1.4668	44	0	1	1	0	1	0	1	1	1	0	1	0	0	1
P5	1.1022	54	1	1	0	1	1	0	1	1	0	0	0	1	0	1
P6	1.9403	51	1	0	0	0	0	1	1	0	1	1	1	1	1	1
P7	1.2065	51	0	0	1	0	1	1	1	1	1	1	1	1	1	1
P8	1.1778	51	1	1	1	1	0	0	1	1	0	1	1	1	1	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.2939	51	0	0	0	0	1	0	0	1	1	0	0	0	0	0
Fold#2	1.8766	49	0	1	0	1	0	0	0	0	0	0	0	0	0	0
Fold#3	1.1066	42	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Fold#4	1.5606	43	0	1	0	0	0	0	0	0	1	0	0	0	0	0
Fold#5	1.1022	54	0	0	0	0	1	1	1	0	0	1	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.2939	51	5-8-9	75.00%
2	Fold #2	1.8766	49	2-4	84.47%
3	Fold #3	1.1066	42	10	67.20%
4	Fold #4	1.5606	43	2-9	79.72%

5	Fold #5	1.1022	54	5-6-7-10	78.64%
---	---------	--------	----	----------	--------

- Nilai interval [61,80]

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7329	72	1	0	1	1	1	0	1	0	0	0	0	0	0	1
P2	1.7801	61	0	0	0	1	0	0	0	1	0	0	0	0	1	1
P3	1.9241	71	1	0	1	0	1	0	1	0	1	0	0	1	1	0
P4	1.2863	77	1	1	0	0	0	1	0	0	1	1	1	0	0	1
P5	1.7526	64	1	0	0	0	1	1	0	0	1	0	1	0	1	1
P6	1.799	66	1	0	0	0	1	0	1	1	0	0	0	1	1	0
P7	1.1524	79	1	1	0	0	0	1	1	1	0	1	0	0	0	1
P8	1.2954	75	1	1	1	1	0	1	0	0	1	0	1	0	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5317	73	0	1	0	0	1	0	1	1	0	1	0	1	0	0
P2	1.1873	79	1	1	0	1	1	0	0	0	1	1	0	1	1	1
P3	1.7401	80	0	1	0	0	1	1	1	0	1	1	1	1	1	0
P4	1.4629	63	1	1	0	0	0	1	1	0	1	0	0	1	1	0
P5	1.8091	78	1	0	1	0	0	1	1	0	0	1	1	1	1	0
P6	1.2951	69	1	1	0	0	0	1	0	0	1	0	1	0	0	0
P7	1.7319	75	1	1	1	0	1	1	0	1	0	0	1	1	1	1
P8	1.2608	73	1	0	1	1	0	1	1	1	1	0	0	0	0	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8833	79	0	0	1	1	1	1	1	1	1	0	0	1	1	1
P2	1.0001	74	0	0	1	1	0	1	1	0	0	0	0	1	1	0
P3	1.8275	74	0	0	0	1	1	1	1	1	1	1	0	0	0	1
P4	1.1772	75	0	0	1	1	1	1	0	0	1	1	1	1	1	0
P5	1.3315	68	0	0	1	1	0	0	0	1	1	1	0	1	1	1
P6	1.5519	80	0	0	0	0	1	1	0	0	1	0	1	0	0	0
P7	1.56	67	0	1	1	0	1	1	1	0	0	1	0	0	1	1
P8	1.082	66	1	0	0	0	1	0	1	1	0	0	0	1	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5399	71	0	1	0	0	0	1	1	1	0	1	1	1	0	0
P2	1.708	76	0	1	1	0	1	0	1	0	0	1	1	1	0	1
P3	1.5904	66	0	1	1	1	1	0	0	1	1	1	1	0	0	1
P4	1.2212	65	0	1	1	0	0	1	1	0	0	1	1	0	0	1
P5	1.4224	62	0	0	0	1	0	1	1	1	1	1	0	0	0	0
P6	1.3866	63	1	1	0	1	0	0	1	1	0	0	1	1	0	1
P7	1.736	70	0	0	1	1	0	1	1	0	1	1	0	0	0	0
P8	1.675	61	1	1	1	1	0	0	0	0	1	1	1	1	0	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9978	74	0	1	0	1	1	0	0	0	1	1	0	0	1	1
P2	1.0615	77	0	1	0	1	0	1	0	0	0	0	1	1	1	1

P3	1.6164	64	0	0	1	0	0	0	1	0	0	1	0	1	1	1
P4	1.2889	75	1	0	0	0	1	1	1	1	0	1	1	0	0	0
P5	1.6837	63	1	1	0	0	1	1	1	0	0	1	0	1	1	1
P6	1.2556	66	1	1	1	1	0	1	1	0	1	1	1	0	1	0
P7	1.0005	80	0	0	1	0	1	0	0	0	1	0	0	1	1	0
P8	1.857	80	1	0	1	1	1	1	0	1	1	1	1	0	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.9241	71	0	0	0	0	0	0	0	1	0	0	0	0	0	1
Fold#2	1.8091	78	0	0	0	1	1	0	0	0	1	0	1	0	0	0
Fold#3	1.0001	74	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Fold#4	1.5399	71	0	0	0	0	0	1	1	0	0	0	0	0	0	0
Fold#5	1.9978	74	0	0	0	0	0	0	0	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.9241	71	8-14	59.16%
2	Fold #2	1.8091	78	4-5-9-11	83.05%
3	Fold #3	1.0001	74	7	25.36%
4	Fold #4	1.5399	71	6-7	63.20%
5	Fold #5	1.9978	74	6	52.61%

Lampiran 11. Detail Hasil Pengujian Pengaruh Jumlah Partikel dengan 121 Data

• Jumlah 4

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6999	9	1	0	1	1	0	0	0	1	1	1	1	0	1	1
P2	1.52	10	1	1	0	1	1	0	0	0	1	1	0	1	1	1
P3	1.9875	4	0	0	1	0	1	0	0	1	0	0	1	0	1	1
P4	1.3212	3	0	1	0	1	1	0	0	0	1	1	1	1	0	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2307	3	1	1	0	0	1	1	1	1	1	1	0	0	1	1
P2	1.0069	8	1	1	1	1	0	1	0	1	0	0	0	0	0	1
P3	1.1828	6	0	1	1	1	1	1	0	1	0	0	0	0	0	1
P4	1.1085	7	1	1	1	0	1	1	1	1	0	0	0	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0425	7	0	1	1	0	0	1	0	1	1	1	0	1	1	0
P2	1.201	9	1	0	0	0	1	0	0	1	1	1	0	1	0	1
P3	1.4231	1	1	1	0	0	1	1	0	1	0	0	1	1	0	1
P4	1.2811	1	0	0	1	0	0	0	0	1	1	0	0	0	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8101	7	1	0	0	1	0	1	0	0	0	0	1	1	0	1
P2	1.7267	5	1	0	0	0	1	1	0	1	1	0	1	1	0	0
P3	1.69	4	0	0	0	0	1	0	0	0	1	1	0	1	0	1

P4	1.9555	4	1	1	0	1	0	1	0	0	1	0	0	1	0	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0347	10	0	0	0	0	1	0	1	1	1	1	0	0	0	0
P2	1.788	5	1	1	1	1	1	1	0	1	0	1	0	1	0	0
P3	1.3072	3	1	0	1	1	1	1	0	1	1	1	1	0	1	1
P4	1.7236	9	1	0	1	1	1	0	0	0	1	0	0	1	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.6999	9	0	0	0	1	0	0	1	0	0	0	0	0	0	0
Fold#2	1.1828	6	0	1	0	0	1	0	0	0	0	0	0	0	0	0
Fold#3	1.4231	1	0	0	0	1	0	0	1	1	0	0	1	0	0	0
Fold#4	1.69	4	0	1	0	0	0	1	0	0	1	0	0	0	0	0
Fold#5	1.0347	10	0	1	0	1	0	0	0	1	0	0	1	0	1	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.6999	9	4-7	85.14%
2	Fold #2	1.1828	6	2-5	78.53%
3	Fold #3	1.4231	1	4-7-8-11	89.46%
4	Fold #4	1.69	4	2-6-9	79.75%
5	Fold #5	1.0347	10	2-4-8-11-13	84.38%

• Jumlah 5

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.165	2	1	0	1	1	1	1	0	0	1	0	0	1	1	0
P2	1.9875	5	0	1	0	1	1	0	0	1	1	1	1	0	1	1
P3	1.9894	10	0	0	1	0	0	0	1	1	0	1	1	0	1	1
P4	1.8177	3	1	1	1	1	1	0	0	0	0	0	0	0	1	1
P5	1.1377	2	0	0	1	1	1	0	1	1	1	0	1	0	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9578	1	0	0	1	0	0	0	0	1	1	1	0	0	0	1
P2	1.258	4	1	1	0	0	1	1	0	0	0	1	1	1	0	0
P3	1.3364	9	0	0	1	0	0	1	0	1	1	1	1	0	1	0
P4	1.8092	7	1	0	1	1	1	1	0	1	1	0	0	0	0	1
P5	1.4754	5	1	1	0	1	0	1	1	1	1	0	0	1	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9264	5	0	0	1	1	1	1	1	0	0	1	1	0	1	1
P2	1.3368	2	1	0	0	0	1	1	1	1	1	1	0	0	0	0
P3	1.5913	2	1	0	0	0	1	0	1	1	0	1	1	0	0	1
P4	1.8759	10	0	0	0	0	1	0	1	0	1	1	0	1	1	0
P5	1.78	4	1	0	0	0	0	1	1	0	1	1	1	0	0	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6319	8	1	0	1	0	0	1	1	1	0	0	1	1	0	0

P2	1.7455	10	0	1	1	0	1	1	0	1	1	0	1	0	0	1
P3	1.4253	1	0	1	1	0	0	0	1	1	0	1	0	0	1	1
P4	1.5789	1	1	0	1	0	0	1	1	0	1	0	0	1	0	1
P5	1.6079	3	1	0	1	0	0	1	0	0	0	0	1	0	0	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7673	8	0	0	0	0	1	1	0	1	1	0	0	0	1	1
P2	1.6987	6	0	1	0	0	0	1	0	0	1	1	0	0	1	1
P3	1.8224	1	1	1	1	0	0	0	1	0	0	0	0	1	0	0
P4	1.45	3	1	0	1	1	1	0	1	0	1	0	0	0	1	0
P5	1.9317	7	0	0	0	0	1	0	0	0	1	0	0	1	1	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.165	2	0	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#2	1.3364	9	0	0	0	1	1	0	0	0	0	0	1	0	0	1
Fold#3	1.3368	2	0	0	0	0	1	1	0	0	0	0	0	0	0	0
Fold#4	1.5789	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Fold#5	1.45	3	0	1	1	0	0	0	1	1	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.165	2	4-11	88.27%
2	Fold #2	1.3364	9	4-5-11-14	88.27%
3	Fold #3	1.3368	2	5-6	70.82%
4	Fold #4	1.5789	1	3-4	81.99%
5	Fold #5	1.45	3	2-3-7-8	71.41%

• Jumlah 6

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9628	6	0	1	0	1	1	1	1	1	1	1	0	0	1	1
P2	1.095	2	0	1	0	0	1	1	0	0	1	1	0	0	0	0
P3	1.3467	6	0	0	1	1	1	0	1	0	1	0	1	1	0	1
P4	1.7672	1	1	1	1	1	1	0	0	1	0	0	0	0	1	1
P5	1.444	5	0	1	0	0	1	0	1	1	0	0	0	0	0	1
P6	1.1108	2	1	0	0	0	1	1	0	1	0	1	0	1	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3419	8	1	0	0	0	0	1	1	0	0	1	1	1	0	1
P2	1.8485	8	1	0	0	0	0	0	0	0	0	1	0	1	0	0
P3	1.3637	9	0	1	0	0	0	1	1	1	0	0	0	0	0	1
P4	1.6412	5	0	0	1	0	0	1	0	1	1	0	1	1	1	0
P5	1.1732	2	1	0	1	0	0	1	1	1	0	1	1	1	1	0
P6	1.6634	1	0	0	1	1	0	1	0	1	1	1	0	0	0	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1177	2	0	0	1	1	0	1	0	1	1	1	1	1	1	0

P2	1.5468	1	0	0	1	0	1	1	1	1	0	1	1	1	0	1
P3	1.252	10	0	1	1	1	0	0	1	0	1	1	1	1	0	0
P4	1.2316	2	1	1	1	1	0	1	0	1	0	1	1	0	0	1
P5	1.0895	3	1	1	0	1	1	0	0	1	1	0	0	1	1	0
P6	1.7833	10	1	1	0	0	1	0	1	1	0	0	0	1	0	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.039	2	1	1	1	1	0	0	1	0	1	1	0	0	0	1
P2	1.2057	8	1	1	1	0	1	1	0	1	1	1	0	1	1	0
P3	1.361	6	0	0	0	0	0	0	1	0	0	1	1	1	0	1
P4	1.699	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1
P5	1.2897	4	0	1	0	0	1	1	0	0	0	0	0	1	0	1
P6	1.1929	5	1	1	0	1	0	0	1	0	0	0	1	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1303	3	0	0	0	0	0	1	0	1	0	1	0	0	0	0
P2	1.87	8	1	1	1	0	1	1	1	0	0	1	0	0	1	0
P3	1.9891	6	0	0	0	1	0	0	1	1	1	0	0	1	1	0
P4	1.8534	2	0	1	1	0	1	1	0	1	1	1	0	1	1	1
P5	1.6621	9	0	0	0	0	1	0	1	1	1	0	1	0	0	0
P6	1.6061	5	1	0	0	0	1	0	0	0	1	1	0	0	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.7672	1	1	0	0	0	0	0	1	0	0	0	0	0	0	1
Fold#2	1.3419	8	0	0	0	1	0	0	0	1	0	0	0	0	0	0
Fold#3	1.7833	10	0	0	0	1	0	0	0	1	0	0	1	1	0	0
Fold#4	1.699	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0
Fold#5	1.6621	9	0	0	0	1	0	0	0	0	0	0	0	1	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.7672	1	1-7-14	67.22%
2	Fold #2	1.3419	8	4-8	77.43%
3	Fold #3	1.7833	10	4-8-11-12	87.07%
4	Fold #4	1.699	1	5-11	77.28%
5	Fold #5	1.6621	9	4-12	79.14%

- Jumlah 8

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9812	6	1	0	1	0	0	1	1	1	0	1	1	1	0	1
P2	1.0892	2	1	0	0	1	1	1	0	0	0	1	1	0	1	0
P3	1.4476	2	1	1	1	1	1	1	0	1	1	1	1	0	0	0
P4	1.0911	7	0	1	1	1	0	1	0	0	0	1	1	0	0	0
P5	1.2327	10	1	0	1	0	1	0	0	0	0	1	0	1	1	0
P6	1.2521	1	0	0	0	1	1	1	0	0	1	0	1	0	1	1

P7	1.4256	3	1	1	0	1	0	0	0	0	1	0	1	0	1	0
P8	1.9791	8	1	1	0	1	0	0	1	0	0	1	1	0	0	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2162	8	0	1	0	1	1	0	1	1	0	0	0	0	0	1
P2	1.2464	5	1	1	1	0	1	1	0	1	1	0	1	1	0	1
P3	1.2796	8	0	0	0	0	0	1	0	0	1	0	1	0	1	0
P4	1.2919	3	0	1	1	0	1	1	0	1	0	0	1	1	1	0
P5	1.3981	6	0	0	0	1	1	0	0	1	1	1	0	1	0	0
P6	1.6767	10	1	0	1	1	0	0	1	1	1	0	1	1	1	0
P7	1.6495	3	1	0	1	1	0	1	1	1	0	1	1	0	0	1
P8	1.1614	9	1	0	1	1	1	1	0	1	0	0	1	0	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3613	1	0	1	0	1	1	0	0	0	1	1	0	0	1	1
P2	1.1591	5	1	1	0	1	0	1	0	1	1	1	0	0	1	0
P3	1.8816	3	1	0	1	1	1	1	1	0	1	1	0	1	0	0
P4	1.8292	9	1	1	1	0	1	1	1	1	1	0	0	1	1	0
P5	1.3452	6	0	1	0	1	1	1	1	1	1	1	1	1	1	0
P6	1.5127	7	1	0	0	0	0	0	0	0	0	0	1	0	1	1
P7	1.525	5	0	1	1	1	0	0	0	1	0	0	0	1	1	0
P8	1.49	1	1	1	1	0	0	0	1	0	0	1	1	1	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0428	8	0	0	1	1	1	1	0	1	0	0	0	0	0	1
P2	1.1925	6	0	0	1	0	0	1	0	1	1	0	1	0	0	1
P3	1.8753	1	0	0	1	1	0	0	1	1	0	1	0	1	0	1
P4	1.6694	2	0	0	0	0	0	1	1	0	1	1	0	0	0	0
P5	1.7443	5	0	0	1	1	1	1	1	1	1	0	0	0	1	0
P6	1.1047	5	0	1	1	0	0	1	0	1	0	0	1	0	0	1
P7	1.788	3	0	0	1	0	1	0	1	1	1	0	1	0	0	1
P8	1.3947	6	0	1	1	1	1	0	0	0	1	1	1	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7938	8	1	0	1	0	1	0	0	1	0	0	1	1	1	0
P2	1.4092	5	1	1	1	0	1	1	0	0	1	0	0	0	1	1
P3	1.2163	9	1	0	0	0	0	0	1	0	1	0	0	0	1	1
P4	1.5574	7	0	0	0	0	0	1	0	0	0	0	1	0	0	0
P5	1.2837	4	0	0	0	1	1	0	0	1	1	1	0	1	0	1
P6	1.1989	5	1	1	1	0	1	1	1	0	0	0	0	1	0	1
P7	1.7833	2	1	0	1	0	0	1	1	0	1	1	0	0	0	0
P8	1.0727	8	0	0	0	0	0	0	1	1	0	0	1	0	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.0892	2	0	1	0	1	0	0	0	0	0	0	0	1	1	0
Fold#2	1.3981	6	1	0	1	1	0	0	0	0	1	0	0	0	0	0
Fold#3	1.8292	9	0	0	0	0	1	0	1	0	0	0	0	0	0	0

Fold#4	1.1047	5	0	0	1	0	0	0	0	0	0	1	0	0	0	0
Fold#5	1.2163	9	0	0	0	1	1	0	0	0	1	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.0892	2	2-4-12-13	85.82%
2	Fold #2	1.3981	6	1-3-4-9	81.83%
3	Fold #3	1.8292	9	5-7	78.48%
4	Fold #4	1.1047	5	3-10	61.39%
5	Fold #5	1.2163	9	4-5-9	83.25%

• Jumlah 10

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5731	1	0	1	0	1	0	0	0	1	1	1	1	1	1	1
P2	1.6959	5	1	0	0	1	1	0	0	0	1	0	0	1	0	0
P3	1.5196	8	0	0	1	1	1	1	0	0	1	1	0	0	0	1
P4	1.7654	7	0	0	0	1	1	0	1	0	1	0	0	1	1	0
P5	1.8142	8	0	0	1	1	0	1	0	1	0	0	1	0	1	1
P6	1.6538	8	1	1	1	1	1	1	1	1	0	0	1	1	1	0
P7	1.2518	3	1	0	0	0	0	0	1	0	0	0	1	1	0	1
P8	1.0851	1	0	1	0	0	1	0	0	0	1	1	0	1	0	1
P9	1.9568	4	0	1	0	1	0	1	0	1	1	0	1	0	1	0
P10	1.477	4	0	1	0	1	0	1	0	1	0	0	0	0	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5361	1	0	1	1	1	1	0	1	1	0	0	1	1	0	1
P2	1.6964	4	0	1	0	1	0	0	1	0	0	1	0	0	0	0
P3	1.7734	8	0	0	0	0	0	1	1	0	1	1	1	1	0	1
P4	1.6643	2	1	1	1	0	1	0	0	1	0	0	1	1	0	0
P5	1.1952	4	1	1	0	0	0	0	0	1	1	0	1	1	1	0
P6	1.2988	8	0	1	1	1	0	1	1	1	1	1	0	0	0	0
P7	1.5324	9	0	0	0	1	1	1	0	0	0	0	0	1	1	0
P8	1.8298	10	1	0	0	1	1	1	0	0	1	1	1	1	0	0
P9	1.7701	8	0	0	1	0	0	0	1	1	0	0	0	1	0	0
P10	1.5657	1	1	0	0	1	1	0	1	1	0	0	1	1	0	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2018	1	0	1	0	1	1	1	1	1	0	1	0	0	0	0
P2	1.7574	8	1	0	1	0	1	1	0	1	0	1	1	0	0	1
P3	1.569	6	0	0	1	1	1	1	1	0	1	1	0	1	0	1
P4	1.2608	3	0	1	0	0	1	1	0	1	0	0	0	1	1	0
P5	1.6587	1	0	1	1	0	1	0	0	0	0	1	0	0	0	0
P6	1.7867	9	1	1	1	0	1	0	1	0	1	0	0	0	0	1
P7	1.0741	3	1	1	1	0	1	1	1	0	1	0	1	0	1	0
P8	1.2534	1	1	0	0	0	1	1	0	1	1	1	1	1	1	1

P9	1.3089	9	1	0	0	1	0	0	0	1	1	1	1	0	1	1
P10	1.468	9	1	0	1	1	1	0	0	0	1	0	1	1	0	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0528	6	0	1	0	0	1	0	1	0	0	0	0	1	0	1
P2	1.1785	1	1	0	1	1	0	1	1	1	1	0	0	1	1	0
P3	1.7018	6	0	1	1	1	0	1	1	1	0	0	1	0	1	0
P4	1.8402	1	0	0	0	0	1	1	1	0	0	1	0	1	0	0
P5	1.9912	5	1	1	1	1	1	0	1	0	1	1	1	1	0	0
P6	1.473	9	0	0	1	0	0	0	0	0	1	1	1	0	1	1
P7	1.9947	5	1	1	1	0	1	1	0	0	0	0	0	1	0	1
P8	1.8758	4	0	1	1	1	0	1	1	0	1	1	1	0	0	0
P9	1.7992	5	1	1	0	0	1	1	1	1	0	1	1	1	1	0
P10	1.9651	2	0	1	0	1	0	1	1	1	1	0	1	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0519	7	1	1	0	0	1	1	0	0	1	1	1	1	0	0
P2	1.2026	4	1	1	0	0	0	1	1	1	0	0	1	0	0	0
P3	1.4012	10	0	1	0	0	0	1	0	0	1	0	0	0	1	0
P4	1.6308	9	1	0	0	1	1	0	1	0	0	0	0	0	0	1
P5	1.7454	2	1	0	1	0	1	0	0	1	1	1	1	0	0	0
P6	1.9134	2	0	0	0	1	1	0	1	1	1	0	0	1	0	0
P7	1.6818	2	1	0	0	1	1	1	1	1	0	1	0	0	1	1
P8	1.9368	6	0	1	1	0	1	0	0	1	1	1	0	1	1	0
P9	1.4874	5	1	0	0	0	1	1	0	1	0	1	1	0	1	1
P10	1.9606	2	1	0	1	1	0	0	0	1	0	1	1	1	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.2518	3	1	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#2	1.1952	4	0	0	0	1	0	0	1	0	0	1	0	0	0	0
Fold#3	1.3089	9	0	1	0	0	0	0	0	0	0	1	0	0	0	1
Fold#4	1.8758	4	0	0	1	0	1	0	0	0	0	0	0	0	0	0
Fold#5	1.4012	10	0	0	0	0	0	0	1	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.2518	3	1-4-11	84.99%
2	Fold #2	1.1952	4	4-7-10	84.46%
3	Fold #3	1.3089	9	2-10-14	76.47%
4	Fold #4	1.8758	4	3-5	74.12%
5	Fold #5	1.4012	10	6	60.98%

Lampiran 12. Detail Hasil Pengujian Pengaruh Jumlah Iterasi dengan 121 Data

- Jumlah iterasi 10

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4104	1	1	0	1	0	0	0	1	1	1	1	0	1	0	1
P2	1.6515	3	1	1	0	0	1	0	1	0	0	1	1	1	0	1
P3	1.0138	6	0	0	0	0	0	1	0	0	1	0	1	1	1	1
P4	1.3301	1	1	0	0	0	0	1	1	1	0	1	1	0	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7587	6	0	0	0	0	0	1	0	1	0	1	1	0	0	0
P2	1.658	5	1	1	1	0	1	0	1	1	1	1	0	0	1	1
P3	1.8107	7	1	0	0	1	0	1	1	1	1	0	1	0	0	0
P4	1.9104	4	0	0	1	1	1	0	0	1	0	1	1	1	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.621	1	0	1	0	0	1	1	0	1	1	0	0	0	0	0
P2	1.3714	3	0	0	0	1	0	1	1	1	0	0	0	1	0	0
P3	1.5878	2	0	0	0	1	1	0	0	0	1	1	0	1	1	1
P4	1.8928	8	1	0	1	0	0	0	0	0	1	1	1	0	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3708	8	1	1	0	1	0	1	1	1	0	1	1	1	0	0
P2	1.4506	7	1	0	0	1	1	1	1	1	1	1	1	1	0	1
P3	1.4373	7	1	1	1	0	0	1	1	1	1	1	0	0	1	1
P4	1.8195	10	0	0	1	0	0	1	1	0	0	1	1	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6382	5	0	0	1	1	0	1	0	1	1	0	0	0	0	0
P2	1.2951	2	1	1	1	1	1	0	0	0	0	1	1	1	1	1
P3	1.7034	2	0	1	0	0	1	0	1	1	0	1	1	0	0	1
P4	1.1267	10	0	1	1	1	0	0	1	1	0	1	0	0	1	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.6515	3	1	0	0	0	1	1	1	0	0	0	1	1	0	0
Fold#2	1.9104	4	1	1	0	0	0	0	1	0	0	0	1	0	0	0
Fold#3	1.3714	3	1	0	0	0	1	0	0	0	0	0	0	1	0	0
Fold#4	1.4373	7	0	1	0	0	1	1	1	0	1	0	0	0	1	0
Fold#5	1.6382	5	0	0	0	1	0	0	0	0	1	1	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.6515	3	1-5-6-7-11-12	72.10%
2	Fold #2	1.9104	4	1-2-7-11	71.65%
3	Fold #3	1.3714	3	1-5-12	61.72%
4	Fold #4	1.4373	7	2-5-6-7-9-13	81.24%
5	Fold #5	1.6382	5	4-9-10	77.08%

- Jumlah iterasi 50

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7709	7	0	1	0	1	1	0	0	1	1	0	1	1	0	1
P2	1.4646	7	0	1	1	1	0	0	1	1	1	0	0	0	1	1
P3	1.59	3	0	1	0	0	1	1	0	0	1	1	1	0	1	1
P4	1.176	6	1	0	0	0	1	0	0	1	1	1	0	0	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2908	6	1	0	1	1	1	0	1	0	0	1	0	1	1	1
P2	1.8329	6	1	1	1	1	0	0	1	0	0	0	0	0	1	1
P3	1.8729	8	0	0	0	1	0	1	0	0	1	1	1	0	1	0
P4	1.2748	4	0	0	1	1	1	1	0	1	0	1	0	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4218	4	1	0	1	0	0	1	1	1	0	1	1	0	0	0
P2	1.0926	2	1	0	1	1	0	0	1	0	0	1	0	1	1	1
P3	1.0977	4	1	0	0	1	1	0	0	1	1	1	0	0	0	1
P4	1.0123	4	1	0	1	1	1	1	0	1	0	1	1	1	0	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5721	3	0	1	1	0	0	0	0	1	0	0	0	1	0	1
P2	1.7028	10	0	0	0	1	0	1	0	1	0	1	0	1	0	1
P3	1.4931	3	1	1	0	0	0	0	1	0	1	1	1	1	1	0
P4	1.6934	7	1	0	0	1	0	1	0	1	0	1	1	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6339	4	0	1	0	1	0	1	1	1	1	1	1	0	0	1
P2	1.8964	3	1	1	0	0	0	0	0	1	0	1	1	0	1	0
P3	1.9896	2	1	1	0	0	0	0	0	0	0	0	0	0	1	0
P4	1.7238	4	1	1	1	0	0	1	0	1	0	0	0	1	1	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.4646	7	0	1	0	0	1	0	0	0	0	1	0	0	0	0
Fold#2	1.8729	8	0	0	0	0	1	1	0	1	0	0	0	0	0	0
Fold#3	1.0123	4	1	0	0	0	0	0	0	0	1	0	0	0	0	0
Fold#4	1.7028	10	0	0	0	0	1	0	0	1	0	1	1	0	1	0
Fold#5	1.8964	3	1	0	0	1	0	0	0	0	0	0	1	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.4646	7	2-5-10	79.27%
2	Fold #2	1.8729	8	5-6-8	73.61%
3	Fold #3	1.0123	4	1-9	26.26%
4	Fold #4	1.7028	10	5-8-10-11-13	70.67%
5	Fold #5	1.8964	3	1-4-11	84.54%

- Jumlah iterasi 100

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9053	10	0	1	1	1	1	0	1	0	1	1	0	1	0	1
P2	1.101	6	1	1	1	1	0	0	1	1	1	1	1	0	1	0
P3	1.7303	2	1	1	0	1	1	1	1	0	1	0	1	1	1	1
P4	1.561	4	0	0	0	0	0	0	0	0	1	0	1	0	1	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5769	6	1	1	1	1	1	1	1	1	0	0	0	0	0	0
P2	1.9129	2	0	0	0	1	1	1	0	1	0	1	0	0	0	0
P3	1.0236	7	1	0	0	1	0	1	1	0	0	1	0	1	1	0
P4	1.5079	6	0	1	0	1	0	1	1	1	0	1	0	1	1	0
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6449	8	1	0	0	0	1	0	1	0	0	0	0	1	1	0
P2	1.9457	2	0	1	0	0	0	0	1	0	0	1	1	0	1	0
P3	1.2415	3	1	0	0	0	0	0	1	1	0	1	0	0	0	0
P4	1.8153	8	1	0	0	0	0	1	0	1	1	0	0	1	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0744	9	1	0	0	0	0	1	1	0	1	0	1	1	1	0
P2	1.1363	8	1	1	0	0	1	0	0	0	0	0	1	1	1	1
P3	1.4173	4	1	0	1	0	1	0	0	0	0	0	0	1	1	0
P4	1.5754	8	0	0	1	1	0	0	0	1	1	1	0	1	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8753	2	1	1	0	1	0	0	0	1	0	0	1	1	1	0
P2	1.8169	7	1	0	0	1	1	1	0	0	1	1	0	0	0	0
P3	1.4647	2	0	1	1	0	0	0	0	0	1	0	0	0	1	1
P4	1.4885	3	0	0	0	1	1	0	1	0	0	0	1	1	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.101	6	1	0	0	1	0	0	0	1	1	0	0	0	0	0
Fold#2	1.5079	6	0	0	0	1	0	1	1	0	0	0	0	0	0	0
Fold#3	1.6449	8	0	1	1	0	0	0	0	0	0	0	0	0	0	1
Fold#4	1.5754	8	0	0	0	0	1	0	0	1	0	0	0	0	0	0
Fold#5	1.8753	2	0	1	0	1	0	1	0	0	0	1	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.101	6	1-4-8-9	79.14%
2	Fold #2	1.5079	6	4-6-7	82.60%
3	Fold #3	1.6449	8	2-3-14	74.31%
4	Fold #4	1.5754	8	5-8	73.43%
5	Fold #5	1.8753	2	2-4-6-10	86.30%

- Jumlah iterasi 150

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6741	7	0	1	1	1	1	1	0	1	0	1	1	0	0	1
P2	1.8961	3	0	0	0	0	0	0	1	0	1	0	0	0	1	1
P3	1.1593	9	0	1	0	1	1	0	1	0	1	0	1	1	1	0
P4	1.7812	3	0	1	0	1	0	0	0	0	0	0	1	1	1	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3388	10	0	1	1	0	0	0	0	1	1	0	1	0	0	1
P2	1.6799	7	0	1	0	1	1	0	1	1	1	0	1	0	1	1
P3	1.8617	5	0	0	1	1	0	1	1	1	1	0	1	1	1	0
P4	1.4375	8	1	1	1	0	0	1	1	1	1	0	1	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8142	1	1	1	1	0	0	1	0	0	1	0	0	0	1	1
P2	1.3435	3	0	0	1	1	0	0	1	0	0	0	1	0	1	1
P3	1.1696	8	0	1	0	0	0	1	0	0	1	0	0	1	1	0
P4	1.9733	1	1	1	1	1	1	0	0	0	1	1	1	0	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2004	3	1	0	0	1	1	0	1	1	1	1	0	1	0	0
P2	1.0724	7	1	1	1	0	0	1	0	0	0	1	0	1	1	0
P3	1.4528	2	0	1	0	1	0	0	0	0	1	1	0	0	1	0
P4	1.7405	1	1	1	0	1	0	1	0	1	1	0	0	1	0	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6244	2	1	1	1	0	0	0	1	1	1	0	1	1	1	0
P2	1.9728	3	1	0	1	1	0	0	1	0	1	0	1	0	0	0
P3	1.3748	4	0	0	1	1	1	1	1	1	1	1	0	0	1	0
P4	1.6892	9	1	0	0	1	0	0	1	0	1	1	0	1	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.6741	7	0	0	0	1	0	0	1	1	0	0	1	0	0	0
Fold#2	1.3388	10	0	0	0	0	1	0	0	0	0	0	1	0	0	0
Fold#3	1.3435	3	0	0	0	1	0	0	0	0	1	0	1	0	0	0
Fold#4	1.0724	7	1	0	0	0	0	1	0	0	0	0	0	0	0	0
Fold#5	1.6244	2	0	1	0	1	0	0	0	0	1	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.6741	7	4-7-8-11	89.53%
2	Fold #2	1.3388	10	5-11	80.40%
3	Fold #3	1.3435	3	4-9-11	88.20%
4	Fold #4	1.0724	7	1-6	55.97%
5	Fold #5	1.6244	2	2-4-9	86.90%

- Jumlah iterasi 250

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1074	10	0	1	0	1	1	0	1	0	0	0	1	0	0	0
P2	1.666	3	0	0	0	0	1	0	0	1	1	1	1	0	0	0
P3	1.835	5	0	0	1	1	0	1	0	1	0	1	0	1	1	0
P4	1.7448	3	1	1	1	1	0	1	0	1	0	1	0	0	0	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3796	8	1	1	1	0	1	0	0	0	1	1	1	0	1	1
P2	1.1774	10	0	0	1	1	0	1	1	1	1	0	0	0	1	0
P3	1.3219	3	1	0	1	1	1	1	0	0	0	1	1	1	1	0
P4	1.8535	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7823	5	0	1	0	1	1	0	1	0	1	1	0	1	1	0
P2	1.1843	4	0	1	1	1	0	1	0	1	1	1	0	1	0	1
P3	1.3474	2	0	0	0	1	0	0	0	1	0	0	0	0	1	0
P4	1.0646	4	1	0	1	0	0	1	1	0	0	0	1	1	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3915	1	1	1	1	1	1	0	1	1	0	1	0	1	1	1
P2	1.4098	1	0	0	0	1	0	1	1	0	1	1	0	1	0	0
P3	1.6256	9	0	1	1	1	1	1	0	0	0	1	1	1	0	1
P4	1.6627	7	0	1	1	0	1	0	0	1	0	0	1	0	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.367	6	0	1	1	1	0	0	1	1	0	1	1	0	1	0
P2	1.0576	3	0	1	0	1	1	0	1	0	1	1	1	1	1	0
P3	1.3934	3	0	0	1	0	1	1	0	0	1	0	0	0	1	1
P4	1.7882	10	1	1	1	0	1	1	1	1	1	0	0	1	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.7448	3	0	0	0	1	1	0	0	0	0	0	1	0	0	0
Fold#2	1.1774	10	0	0	0	1	1	0	0	0	0	0	0	0	0	0
Fold#3	1.1843	4	0	0	0	1	0	0	0	1	1	0	1	0	0	1
Fold#4	1.4098	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0
Fold#5	1.3934	3	1	0	0	0	1	0	1	0	0	0	0	1	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.7448	3	4-5-11	89.47%
2	Fold #2	1.1774	10	4-5	84.49%
3	Fold #3	1.1843	4	4-8-9-11-14	91.31%
4	Fold #4	1.4098	1	4-6-8	84.50%
5	Fold #5	1.3934	3	1-5-7-12	72.12%

Lampiran 13. Detail Hasil Pengujian Pengaruh Nilai K dengan 121 Data

- Jumlah 4

Fold 1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5136	3	1	1	0	1	1	1	0	1	0	0	0	1	0	0
P2	1.0583	7	1	1	0	0	0	0	1	1	1	1	1	1	1	1
P3	1.875	4	0	0	0	0	0	0	1	0	0	1	0	1	0	0
P4	1.8451	9	0	1	1	1	0	1	0	1	1	0	0	0	0	1
Fold 2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4797	3	1	1	0	0	0	0	1	1	0	1	0	1	1	0
P2	1.3623	4	0	1	0	1	1	0	1	0	1	1	1	1	0	1
P3	1.896	1	0	1	1	1	1	1	1	1	1	0	1	0	1	1
P4	1.3582	6	1	1	1	1	0	0	1	0	0	1	0	0	1	0
Fold 3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5614	7	1	0	0	0	1	1	1	0	1	0	0	0	1	0
P2	1.8436	8	0	0	1	0	1	1	1	1	0	0	1	1	0	1
P3	1.3777	1	1	0	1	1	1	0	1	0	0	1	1	1	0	0
P4	1.8726	8	0	1	1	1	1	0	1	0	0	0	1	1	0	1
Fold 4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0947	10	0	1	1	0	0	0	0	0	1	0	0	1	1	1
P2	1.7873	8	0	1	1	1	0	0	1	0	0	0	0	0	0	1
P3	1.5451	3	1	1	0	0	0	0	0	1	0	1	0	0	0	0
P4	1.4511	5	1	0	0	0	0	0	1	0	0	1	0	0	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.8451	9	0	1	0	1	0	1	1	0	0	0	0	0	0	0
Fold#2	1.3623	4	0	1	0	0	0	0	1	0	0	0	0	0	1	0
Fold#3	1.3777	1	0	1	0	1	0	0	0	0	1	0	0	0	1	0
Fold#4	1.0947	10	0	0	0	1	0	1	0	0	1	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.8451	9	2-4-6-7	85.50%
2	Fold #2	1.3623	4	2-7-13	79.11%
3	Fold #3	1.3777	1	2-4-9-13	82.43%
4	Fold #4	1.0947	10	4-6-9	79.14%

- Jumlah 5

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6402	2	0	0	0	0	0	0	0	1	0	0	0	1	0	0
P2	1.0533	8	0	0	0	1	0	1	1	0	0	0	1	0	1	1

P3	1.1372	6	1	1	0	0	1	0	1	1	0	0	0	1	1	1
P4	1.8077	5	1	1	0	1	0	0	1	1	0	0	1	1	0	0
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8092	1	1	1	1	1	0	0	0	1	1	0	1	1	0	0
P2	1.634	4	1	0	1	1	1	1	1	1	0	1	0	1	0	1
P3	1.9171	5	0	1	1	1	1	1	1	0	1	0	0	1	1	1
P4	1.0342	3	1	0	0	0	0	0	0	1	0	0	1	1	0	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8153	6	1	1	0	1	0	1	1	1	1	0	0	0	1	0
P2	1.169	8	0	0	1	0	1	0	0	1	0	1	1	0	0	0
P3	1.0925	10	1	0	0	0	0	0	1	0	0	1	1	0	0	1
P4	1.9348	5	1	0	0	0	1	0	0	1	0	1	0	0	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5312	3	0	0	1	0	0	1	1	1	0	1	1	1	0	0
P2	1.652	8	1	1	1	0	1	1	0	0	0	0	1	1	0	0
P3	1.8005	3	0	0	1	0	1	0	1	1	1	0	1	0	1	0
P4	1.9413	4	1	1	1	0	1	0	1	1	1	1	0	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0074	8	0	1	1	0	1	1	1	0	1	1	0	1	1	0
P2	1.8737	4	0	1	1	0	0	0	1	0	1	1	0	1	0	0
P3	1.5445	1	0	1	0	1	1	0	0	0	1	1	0	0	1	0
P4	1.5944	2	0	0	1	0	0	0	1	0	1	1	1	1	0	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.1372	6	0	0	0	1	0	0	1	0	0	0	0	0	0	0
Fold#2	1.9171	5	0	0	0	1	0	1	0	0	0	1	1	0	0	1
Fold#3	1.169	8	0	1	0	0	0	1	0	0	0	0	0	0	0	0
Fold#4	1.652	8	0	1	0	0	0	0	0	0	0	0	0	1	0	0
Fold#5	1.5445	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.1372	6	4-7	85.18%
2	Fold #2	1.9171	5	4-6-10-11-14	83.86%
3	Fold #3	1.169	8	2-6	78.91%
4	Fold #4	1.652	8	2-12	77.60%
5	Fold #5	1.5445	1	9	26.21%

• Jumlah 6

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8407	9	0	1	0	0	1	0	1	0	1	1	1	1	1	1
P2	1.716	3	1	1	0	1	0	0	1	0	1	0	1	1	0	1
P3	1.2877	1	0	1	0	1	0	1	0	0	1	1	1	0	0	1
P4	1.8544	3	1	0	1	0	0	0	0	0	1	0	0	1	1	1

Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1213	5	1	1	0	0	1	0	1	1	1	1	1	1	0	1
P2	1.038	1	1	1	0	1	1	1	1	1	0	1	1	1	1	0
P3	1.7741	8	0	1	0	1	1	0	0	0	1	1	1	1	0	1
P4	1.2087	7	0	0	1	1	1	0	0	0	1	0	1	1	0	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2784	2	1	0	0	1	1	1	1	1	1	1	0	1	0	0
P2	1.3546	6	0	1	1	1	0	1	0	1	0	1	1	0	0	0
P3	1.8829	10	1	0	1	0	1	1	1	1	1	0	0	1	0	0
P4	1.2395	2	1	1	1	1	1	1	1	0	0	0	0	1	1	0
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6043	4	1	1	0	1	1	0	1	1	0	0	1	0	1	0
P2	1.1771	10	1	1	0	1	1	0	0	0	1	0	0	1	0	0
P3	1.315	1	1	1	1	0	1	0	1	1	1	0	1	1	1	1
P4	1.7706	6	0	0	1	1	0	0	0	1	1	0	0	0	0	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.9543	2	0	1	0	1	0	0	0	0	0	1	1	0	0	0
P2	1.9671	7	1	0	0	0	0	0	0	0	1	0	1	1	0	0
P3	1.6854	6	1	1	1	1	0	1	0	1	0	0	1	0	1	1
P4	1.168	2	0	0	0	1	1	1	1	1	0	1	0	1	1	0
Fold6	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8804	10	0	0	0	1	1	1	1	0	1	1	0	0	1	0
P2	1.9097	10	0	1	1	0	0	0	1	0	0	1	1	0	0	1
P3	1.7864	3	0	1	0	0	0	0	0	1	0	1	1	0	1	0
P4	1.3203	3	1	1	1	1	1	1	1	0	0	0	1	0	0	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.2877	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0
Fold#2	1.1213	5	0	1	0	1	0	0	1	0	0	1	0	0	0	0
Fold#3	1.2784	2	0	0	0	1	0	0	0	0	0	1	0	1	0	0
Fold#4	1.6043	4	0	0	0	0	0	0	1	0	0	1	0	0	0	0
Fold#5	1.168	2	0	1	0	0	1	0	1	0	0	1	0	0	0	0
Fold#6	1.8804	10	0	0	0	0	0	0	1	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.2877	1	2-13	65.25%
2	Fold #2	1.1213	5	2-4-7-10	86.38%
3	Fold #3	1.2784	2	4-10-12	74.24%
4	Fold #4	1.6043	4	7-10	73.67%
5	Fold #5	1.168	2	2-5-7-10	82.48%
6	Fold #6	1.8804	10	7	70.20%

- Jumlah 8

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6457	2	1	1	0	1	1	0	1	0	0	1	0	0	1	0
P2	1.6046	2	0	1	1	0	0	0	1	1	0	1	0	1	1	0
P3	1.3464	10	1	1	1	1	1	1	0	1	0	0	1	0	0	0
P4	1.4827	3	1	1	1	1	0	0	0	0	1	0	1	1	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1823	2	1	1	0	1	0	0	0	1	1	0	0	0	0	0
P2	1.6974	5	0	1	0	0	0	0	1	1	1	1	1	0	0	0
P3	1.142	5	0	1	1	0	1	0	0	0	1	0	0	0	1	0
P4	1.7097	7	0	1	1	0	1	0	1	1	1	1	0	1	1	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6403	5	1	1	0	1	1	1	1	0	1	0	0	0	1	1
P2	1.3681	9	0	0	0	0	1	0	0	1	1	1	0	0	1	0
P3	1.7309	4	1	0	1	1	0	1	0	0	1	0	0	1	1	1
P4	1.3252	8	1	0	1	1	1	0	0	0	1	0	1	1	1	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5402	2	1	1	0	1	1	1	0	0	1	1	0	0	1	1
P2	1.971	4	0	0	0	1	1	1	1	1	0	0	0	0	0	1
P3	1.3217	6	0	1	0	0	1	0	0	1	1	0	1	0	1	1
P4	1.8335	8	1	1	0	0	0	0	1	1	1	1	0	1	1	0
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3005	2	0	0	1	1	1	0	0	0	1	1	0	0	1	1
P2	1.6004	10	1	1	0	0	0	0	0	0	0	0	0	0	0	1
P3	1.4965	4	0	1	1	1	1	1	0	1	1	0	1	1	1	1
P4	1.0917	2	1	0	0	1	0	0	0	0	1	1	1	1	0	0
Fold6	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.1522	10	0	1	1	0	0	0	1	1	0	1	1	1	1	0
P2	1.2688	2	1	0	0	1	0	0	0	0	0	0	1	0	0	1
P3	1.2661	4	0	0	1	0	1	1	1	0	1	1	1	0	0	0
P4	1.6414	8	1	1	1	0	1	0	1	1	0	1	1	1	0	1
Fold7	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8567	5	0	0	1	0	1	1	0	1	0	0	1	1	1	0
P2	1.4082	9	1	0	0	1	1	0	0	1	1	1	0	0	0	0
P3	1.2936	10	0	1	0	0	0	1	1	1	0	1	1	1	1	1
P4	1.168	9	1	0	0	0	0	1	0	0	1	1	0	0	1	0
Fold8	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.997	9	0	0	1	0	0	1	0	1	0	1	1	0	0	1
P2	1.8913	3	1	1	0	1	1	1	0	0	1	0	0	0	0	1
P3	1.8841	10	1	1	1	0	0	1	0	0	0	0	0	0	0	1
P4	1.3075	7	1	0	0	0	1	0	1	0	0	1	1	0	1	1

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.6457	2	0	0	1	1	0	0	1	0	1	0	0	0	0	0
Fold#2	1.1823	2	0	1	0	1	0	0	0	0	1	0	1	0	0	0
Fold#3	1.6403	5	0	0	0	1	0	0	0	0	0	1	0	0	0	0
Fold#4	1.5402	2	1	0	0	1	0	0	0	1	1	0	1	0	1	0
Fold#5	1.3005	2	0	0	0	1	1	0	0	1	1	0	1	0	0	0
Fold#6	1.6414	8	0	1	0	1	0	0	0	0	0	1	0	0	0	0
Fold#7	1.4082	9	0	1	0	1	0	0	0	0	0	0	1	0	1	0
Fold#8	1.3075	7	0	1	0	0	0	0	0	0	0	0	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.6457	2	3-4-7-9	74.66%
2	Fold #2	1.1823	2	2-4-9-11	81.50%
3	Fold #3	1.6403	5	4-10	82.94%
4	Fold #4	1.5402	2	1-4-8-9-11-13	85.04%
5	Fold #5	1.3005	2	4-5-8-9-11	89.49%
6	Fold #6	1.6414	8	2-4-10	86.62%
7	Fold #7	1.4082	9	2-4-11-13	83.91%
8	Fold #8	1.3075	7	2	78.55%

• Jumlah 10

Fold1	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2499	5	1	1	0	0	0	0	0	1	0	0	1	0	1	0
P2	1.9999	3	1	1	0	1	1	0	1	1	1	0	0	0	0	1
P3	1.4311	10	0	0	1	1	0	0	1	0	1	0	1	1	1	0
P4	1.0652	5	1	0	1	1	1	1	0	0	0	0	1	0	0	1
Fold2	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.5443	5	0	0	1	0	0	1	1	0	0	0	0	0	0	1
P2	1.4473	3	1	0	1	1	1	0	1	1	1	0	0	1	1	0
P3	1.8967	8	1	0	1	0	0	1	1	1	1	0	1	0	1	0
P4	1.2208	9	0	0	1	0	1	1	1	1	1	0	1	0	0	1
Fold3	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.7582	5	1	0	1	0	0	0	1	1	0	0	0	0	1	0
P2	1.6824	2	1	0	1	1	1	1	0	0	1	1	1	0	0	1
P3	1.5876	7	0	1	1	0	1	1	0	1	0	1	1	1	0	1
P4	1.1308	4	1	0	1	1	0	1	1	0	0	0	1	0	0	1
Fold4	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.8597	6	0	1	0	1	1	1	0	1	0	1	1	1	0	0
P2	1.9382	8	0	1	0	0	0	1	0	0	1	1	0	0	0	1
P3	1.5315	7	0	1	1	1	0	1	0	0	1	1	1	0	1	0
P4	1.96	10	1	1	1	1	0	0	1	1	0	1	1	0	1	1
Fold5	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14

P1	1.3346	6	0	0	1	0	1	1	1	1	0	0	0	0	1	0
P2	1.833	5	1	0	1	1	1	1	0	1	0	1	1	1	1	0
P3	1.9537	7	0	0	0	1	0	1	0	1	0	0	1	0	0	0
P4	1.0226	8	0	0	1	0	0	1	0	0	1	1	0	1	0	1
Fold6	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.2527	10	1	0	1	0	0	1	0	1	1	0	0	1	0	0
P2	1.558	9	0	1	0	1	0	0	1	0	0	1	1	1	0	1
P3	1.1121	3	0	0	1	1	0	0	1	1	1	1	0	0	0	1
P4	1.664	10	1	1	1	1	1	0	1	0	1	1	0	0	1	1
Fold7	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.3938	7	1	0	1	0	1	0	0	1	1	1	0	1	1	0
P2	1.5873	10	1	0	1	1	0	0	0	1	0	0	0	1	0	1
P3	1.8789	9	1	1	0	0	1	1	0	1	1	1	0	1	1	0
P4	1.1483	4	1	1	1	0	1	0	0	0	1	1	1	0	1	0
Fold8	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.0996	9	1	0	1	1	0	1	1	1	0	1	0	0	0	1
P2	1.3613	1	1	0	0	0	1	0	0	1	1	1	0	0	0	1
P3	1.3273	1	1	0	1	1	0	1	1	1	1	1	0	1	0	1
P4	1.2507	2	1	0	1	1	0	0	0	1	0	1	0	0	1	1
Fold9	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.6363	8	1	1	1	0	1	1	0	0	0	0	0	0	0	0
P2	1.675	10	1	1	0	1	0	0	0	0	1	0	0	0	0	1
P3	1.112	3	0	1	1	0	0	1	1	0	0	0	1	0	1	1
P4	1.2031	5	1	1	1	1	0	1	1	1	1	0	0	0	0	1
Fold10	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
P1	1.4456	2	1	0	0	1	0	1	0	1	0	0	1	1	0	0
P2	1.9132	4	1	0	1	1	1	1	1	0	0	0	1	0	1	0
P3	1.4972	9	0	0	0	1	1	1	1	0	0	1	0	0	0	1
P4	1.4635	4	1	0	0	0	0	0	1	0	1	0	1	0	1	0

Gbest	m	k	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14
Fold#1	1.9999	3	0	1	0	0	0	0	0	0	1	0	0	0	0	0
Fold#2	1.4473	3	0	1	0	0	0	0	0	0	0	0	1	0	1	0
Fold#3	1.7582	5	1	0	0	1	1	0	0	0	0	0	1	0	0	0
Fold#4	1.5315	7	0	0	0	1	0	0	0	0	0	0	1	0	0	0
Fold#5	1.833	5	0	1	0	1	0	0	0	0	0	0	0	0	0	0
Fold#6	1.2527	10	1	0	0	0	0	0	0	0	0	0	0	1	0	0
Fold#7	1.1483	4	0	0	0	1	0	1	0	0	0	0	0	0	0	0
Fold#8	1.3273	1	0	0	0	0	1	1	0	0	0	0	0	1	1	0
Fold#9	1.6363	8	0	0	0	1	0	0	0	0	0	0	0	0	0	1
Fold#10	1.4635	4	0	0	0	0	0	1	1	0	1	1	0	0	0	0

No	Gbest	m	k	Fitur Terpilih	Rata-rata Akurasi
1	Fold #1	1.9999	3	2-9	69.51%
2	Fold #2	1.4473	3	2-11-13	71.32%
3	Fold #3	1.7582	5	1-4-5-11	83.72%
4	Fold #4	1.5315	7	4-11	84.94%
5	Fold #5	1.833	5	2-4	82.83%
6	Fold #6	1.2527	10	1-12	60.84%
7	Fold #7	1.1483	4	4-6	81.11%
8	Fold #8	1.3273	1	5-6-12-13	68.05%
9	Fold #9	1.6363	8	4-14	77.13%
10	Fold #10	1.4635	4	6-7-9-10	78.29%

Lampiran 14. Detail Hasil Pengujian Diluar Skenario dengan 20 Partikel dan 100 Iterasi pada 121 Data

- Hasil coba dengan 20 partikel, K=8, interval m [1, 2], interval k [1, 10], dan iterasi =100

No	Gbest	m	k	Fitur	Rata-rata akurasi
1	Fold #1	4.3163	8	1-9	64.134%
2	Fold #2	5.7806	3	2	76.5631%
3	Fold #3	4.5852	3	8	51.7491%
4	Fold #4	5.3578	7	7-10	74.9955%
5	Fold #5	5.329	5	3-5	66.3046%
6	Fold #6	5.7673	2	3-8	51.585%
7	Fold #7	4.5765	3	11-13	58.7862%
8	Fold #8	4.4799	9	5-7	76.9334%

- Hasil coba dengan 15 partikel, K=8, interval m [1, 2], interval k [1, 10], dan iterasi =200

No	Gbest	m	k	Fitur	Rata-rata akurasi
1	Fold #1	5.1203	3	7	69.8493%
2	Fold #2	5.0579	6	4-13	79.6819%
3	Fold #3	4.9559	8	5	66.3527%
4	Fold #4	4.5789	1	2-7-8	71.1344%
5	Fold #5	5.8223	10	4-8-14	81.127%
6	Fold #6	4.7607	8	8	59.9502%
7	Fold #7	5.6036	2	2	76.8145%
8	Fold #8	4.2964	4	4-11	88.1446%

Lampiran 15. Data Hasil Pengujian Konvergensi

	Fold #1	Fold #2	Fold #3	Fold #4	Fold #5	Fold #6	Fold #7	Fold #8
1	0.6391	0.8094	0.6148	0.6606	0.6366	0.6489	0.7699	0.4185
2	0.6436	0.8094	0.7024	0.7348	0.6540	0.7349	0.7699	0.4185
3	0.6436	0.8094	0.7024	0.7348	0.6961	0.7349	0.8683	0.4185
4	0.6436	0.8094	0.7024	0.7348	0.6961	0.7349	0.8683	0.4185
5	0.7107	0.8358	0.7024	0.7545	0.6961	0.7349	0.8683	0.4185
6	0.8006	0.8358	0.7024	0.7545	0.6961	0.7349	0.8683	0.4329
7	0.8006	0.8358	0.7024	0.7545	0.6961	0.7349	0.8683	0.4584
8	0.8006	0.8358	0.7024	0.7545	0.6961	0.7349	0.8683	0.4584
9	0.8006	0.8358	0.7130	0.7545	0.6961	0.7349	0.8683	0.4584
10	0.8006	0.8358	0.7130	0.7545	0.6961	0.7349	0.8683	0.4584
11	0.8006	0.8358	0.7130	0.7545	0.6961	0.7349	0.8683	0.4584
12	0.8006	0.8679	0.7130	0.7545	0.7697	0.7455	0.8683	0.4584
13	0.8006	0.8750	0.7130	0.7545	0.7697	0.7455	0.8683	0.4584
14	0.8006	0.8750	0.7130	0.7545	0.7697	0.7694	0.8683	0.4584
15	0.8006	0.8750	0.7130	0.7545	0.7697	0.7694	0.8683	0.4796
16	0.8006	0.8750	0.7130	0.7545	0.7697	0.7694	0.8683	0.4796
17	0.8006	0.8858	0.7130	0.7545	0.7697	0.7694	0.8683	0.4796
18	0.8006	0.8858	0.7130	0.7545	0.7697	0.7694	0.8683	0.4796
19	0.8006	0.8858	0.7130	0.7545	0.7697	0.7694	0.8683	0.4796
20	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
21	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
22	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
23	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
24	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
25	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.4796
26	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.5292
27	0.8006	0.8858	0.7540	0.7545	0.7697	0.7694	0.8683	0.5292
28	0.8006	0.8858	0.8701	0.7647	0.7697	0.7694	0.8683	0.5292
29	0.8006	0.8858	0.8701	0.7647	0.8449	0.7694	0.8683	0.5292
30	0.8113	0.8858	0.8701	0.7647	0.8449	0.7694	0.8683	0.5292
31	0.8113	0.8858	0.8701	0.7647	0.8449	0.7694	0.8683	0.5292
32	0.8113	0.8858	0.8701	0.7647	0.8449	0.7694	0.8683	0.5292
33	0.8113	0.8858	0.8701	0.7647	0.8449	0.7694	0.8790	0.5292
34	0.8113	0.9041	0.8701	0.7647	0.8449	0.7694	0.8790	0.5292
35	0.8220	0.9041	0.8701	0.7647	0.8449	0.7694	0.8790	0.5292
36	0.8220	0.9041	0.8701	0.7647	0.8449	0.7694	0.8790	0.5292
37	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
38	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
39	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
40	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292

41	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
42	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
43	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
44	0.8220	0.9041	0.8701	0.7943	0.8449	0.7694	0.8790	0.5292
45	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8790	0.5292
46	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8790	0.5292
47	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8790	0.5292
48	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8790	0.5292
49	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
50	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
51	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
52	0.8220	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
53	0.8473	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
54	0.8473	0.9041	0.8701	0.8177	0.8449	0.7694	0.8967	0.5292
55	0.8473	0.9041	0.8701	0.8218	0.8449	0.7694	0.8967	0.5292
56	0.8473	0.9041	0.8701	0.8218	0.8449	0.7694	0.9112	0.5292
57	0.8473	0.9041	0.8701	0.8218	0.8449	0.7694	0.9112	0.5292
58	0.8473	0.9041	0.8701	0.8218	0.8449	0.7694	0.9112	0.5292
59	0.8473	0.9041	0.8701	0.8218	0.8449	0.7694	0.9112	0.5292
60	0.8473	0.9041	0.8701	0.8434	0.8449	0.7694	0.9112	0.5292
61	0.8473	0.9041	0.8701	0.8434	0.8449	0.7694	0.9112	0.5292
62	0.8473	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
63	0.8473	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
64	0.8473	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
65	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
66	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
67	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
68	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
69	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
70	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
71	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
72	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
73	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
74	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
75	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
76	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
77	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
78	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
79	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
80	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
81	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
82	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292
83	0.8635	0.9041	0.8701	0.8614	0.8449	0.7694	0.9112	0.5292

L-48

L-49

L-50

L-51