KLASIFIKASI PENYAKIT KAMBING DENGAN MENGGUNAKAN ALGORITME *SUPPORT VECTOR MACHINE* (SVM)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Ardiza Dwi Septian
NIM: 135150207111033

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ALGORITME *SUPPORT VECTOR MACHINE* (SVM) PADA KLASIFIKASI PENYAKIT KAMBING

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Ardiza Dwi Septian
NIM: 135150207111033

Skripsi ini telah diuji dan dinyatakan lulus pada
1 Agustus 2018
Telah diperiksa dan disetujiu oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Lailil Muflikhah, S.Kom, M.Sc	Randy Cahya Wihandika, S.ST., M.Kom
NIP: 19831013 201504 2 002	NIK: 201405 880206 1

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan 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 disitasi dalam naskah ini dan disebutkan dalam 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, 19 Juli 2018

Ardiza Dwi Septian

NIM: 135150207111033

KATA PENGANTAR

Puji Syukur penulis panjatkan kehadirat Allah SWT karena atas karunia serta hidayah-Nya yang telah diberikan sehingga penulis dapat menyelesaikan skripsi yang berjudul **"Klasifikasi Penyakit Kambing Dengan Menggunakan Algoritme *Support Vector Machine* (SVM)".** Penyusunan skripsi ini tidak semata-mata hasil kerja individu penulis melainkan berkat bimbingan, doa dan dorongan yang telah diberikan oleh pihak – pihak yang membantu, baik secara materi ataupun non materi. Maka pada kesempatan ini penulis ingin mengucapkan banyak terima kasih kepada semua pihak, diantaranya:

1. Lailil Muflikhah, S.Kom, M.Sc dan Randy Cahya Wihandika, S.ST., M.Kom selaku dosen pembimbing yang telah meluangkan waktunya untuk membantu dan membimbing penulis dalam pengerjaan skripsi ini.
2. Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D, Heru Nurwasito, Ir., M.Kom, Marji, Drs., M.T, Edy Santoso, S.Si, M.Kom selaku Dekan, Wakil Dekan I, Wakil Dekan II, Wakil Dekan III Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Ibu Dosen Fakultas Ilmu Komputer yang telah memberikan ilmunya selama saya menempuh pendidikan dan menyelesaikan skripsi ini.
4. Kedua orang tua penulis, Sudiono dan Endang Sukriyani yang telah tulus memberikan dukungan penulis menyelesaikan skripsi serta saudara – saudara penulis Septia dan Sulestiawan, Iqbal dan Lestia yang selalu menghibur penulis.
5. Pendamping hidup sementara penulis Deria Maharani yang telah setia memberikan doa dan motivasi untuk penulis.
6. Sahabat karib penulis Alvan, Yobel, Bayu, Yogi, Adit, Rezha, Ikhsan, Verio, Oky, Yossi, Bima, Uzha, Qori, Yoga,Fifi dan Aras dan yang lain teima kasih telah mendukung dalam proses penyelesaian skripsi ini dan proses kuliah di Fakultas Ilmu Komputer.
7. Seluruh teman-teman Informatika UB angkatan 2013 serta semua pihak yang tidak dapat saya sebutkan satu persatu yang telah membantu dan mendukung penulis selama proses pendidikan sampai terselesaikannya skripsi ini baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa skripsi ini masih mempunyai kekurangan, oleh karena itu segala bentuk kritik dan saran yang membangun dapat disampaikan melalui email penulis ardizaseptian@gmail.com

Malang, 19 Juli 2018

Penulis

ABSTRAK

Ardiza Dwi Septian. 2018. Klasifikasi Penyakit Kambing Dengan Menggunakan Algoritme *Support Vector Machine* (SVM). Fakultas Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing: Lailil Muflikhah, S.Kom, M.Sc dan Randy Cahya Wihandika, S.ST., M.Kom.

Masyarakat pedesaan di Indonesia sangat akrab dengan ternak hewan kambing karena modal untuk memelihara kambing relatif lebih kecil dan dapat berkembang biak dengan cepat. Faktor yang harus diperhatikan dalam memelihara kambing adalah kesehatan kambing itu sendiri. Kesehatan menjadi penting karena dapat menyebabkan kerugian akibat kambing yang tidak sehat. Ketika gejala penyakit timbul maka sebaiknya segera dilakukan tindakan penanganan dini salah satunya dengan mendiagnosis penyakit kambing itu sendiri. Tetapi pengetahuan dalam mendiagnosis penyakit tidak banyak diketahui oleh peternak kambing menjadikan penanganan penyakit kambing menjadi sulit. Oleh karena itu diperlukan sebuah sistem untuk membantu mengklasifikasi penyakit kambing. Penelitian klasifikasi penyakit kambing ini menggunakan algoritme *Support Vector Machine* dengan strategi *One Against All.* Data yang digunakan dalam penelitian ini sebanyak 148 data dengan 11 kelas penyakit kambing diantaranya Cacingan*, Endometritis,* Kelumpuhan, Kembung, Keracunan*, Masistis, Myasis, Orf, Pink Eye, Pneumonia dan Scabies*. Hasil akurasi yang didapatkan pada sistem ini adalah 90% dengan menggunakan parameter terbaik yaitu k-fold cross validation 10, λ= 0.1, C = 0,1, iterasi = 500 dan σ = 1.

Kata Kunci: *Klasifikasi, Penyakit Kambing, Support Vector Machine, One-Against-All*

ABSTRACT

Ardiza Dwi Septian. 2018. Classsification of Goat Disease With Algoritme *Support Vector Machine* (SVM). Faculty of Computer Science, Brawijaya University, Malang. Lecturer: Lailil Muflikhah, S.Kom, M.Sc dan Randy Cahya Wihandika, S.ST., M.Kom.

Indonesian's rural communities really familiar with goat cattling, that's because the fund for it's nurturing more cheaper and they breed faster.The main factor to nurturing is the health of the goat itself if the goat get sick, it will become disadvantage for them. So that's why health issues become the main factor.If there's disease indications exist, the early handling must be done soon. A disease diagnose is first thing to do.But, the awareness to diagnose the disease are still unknown. That's make the cattleman feel uneasy to handle it.Therefore, it need a system to help them to clasify the disease.This goat disease's research used algoritme *Support Vector Machine* with one againts all strategy. The data that used are 148 datas with 11 disease *classes, there are wormy, endometritis, paralizing, bloated, poisoning, Masistis, Myasis, Orf, Pink Eye, Pneumonia and Scabies*. The accuracy result that get from this system is 90% with using the best parameter that called k-fold cross validation 10 , λ= 0.1, C = 0,1, iterasi = 500 and σ = 1.

Keywords *: Classification, Goat Disease, Support Vector Machine, One Against All*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

BAB 1 PENDAHULUAN

1.1 Latar belakang

Masyarakat Indonesia terutama masyarakat pedesaan sangat akrab dengan ternak hewan kambing karena modal untuk memelihara kambing relatif lebih kecil daripada beternak sapi. Dengan keuntungan yang lebih seperti kemampuan kambing yang berkembang biak satu sampai tiga ekor anak kambing dalam satu kali proses berkembang biak (Susanto & Sitanggang, 2015). Menurut Direktorat Jendral Peternakan dan Kesehatan Hewan, populasi kambing di tahun 2016 yang ada di Jawa Timur mencapai 3.267.954 ekor. Dari data tersebut memberikan bukti bahwa Jawa Timur menjadi provinsi yang mempunyai potensi dalam ternak kambing yang sangat besar.

Dalam berternak kambing, faktor kesehatan kambing merupakan faktor sangat penting. Pemberian antibiotik dan obat cacing adalah bagian untuk menjaga kesehatan kambing, pemberian vaksin yang tepat biasanya dilakukan oleh pakar atau ahli (Mulyono & Sarwono, 2014). Kesehatan pada hewan ternak sangat penting karena dapat menyebabkan kerugian bagi peternak akibat gangguan pertumbuhan pada ternak, umur induk yang terlambat beranak pertama kali, masa reproduksi yang terganggu, efisiensi pemberian pakan ternak rendah dan sampai terjadi kematian pada ternak (Bahri, Adjid & April, 2013). Ketika gejala-gejala penyakit pada kambing timbul maka sebaiknya segera dilakukan tindakan penanganan secara dini agar penyakit kambing tidak menjadi parah. Tetapi pengetahuan dalam mendiagnosis penyakit tidak banyak diketahui oleh peternak kambing di desa membuat tindakan dalam penanganan penyakit kambing menjadi sulit (Ferdiansyah W.R, 2017). Dari permasalahan tersebut dibutuhkan sebuah sistem yang akan membantu peternak kambing dalam mengklasifikasi penyakit kambing.

Klasifikasi adalah proses yang ditujukan untuk menentukan objek kedalam suatu kelas atau kategori yang telah ditentukan sebelumnya. *Support Vector Machine* (SVM) merupakan metode statistik untuk diterapkan dalam klasifikasi. Konsep dasar dari Algoritme SVM yaitu *linier classifier* dan telah dikembangkan untuk mengatasi masalah yang non-linier dengan menggunakan fungsi kernel trick (Nugroho, 2003). Dengan menggunakan *Support Vector Machine* ada beberapa kelebihan yaitu menentukan jarak dengan support vector menjadikan komputasi lebih cepat (Octaviani, 2014).

Penelitian terkait dengan penyakit kambing yang dilakukan sebelumnya yaitu penelitian yang menggunakan metode *Certainty Factor.* Output sistem ini adalah diagnosa penyakit yang menginfeksi kambing dan solusi yang diberikan terhadap kambing yang terinfeksi penyakit tersebut. Kelebihan dari penelitian ini yaitu

memiliki 56 jenis gejala yang menjadi inputan. Kelemahan dari penelitian ini yaitu jenis penyakit hanya ada 10 jenis. Nilai akurasi yang di dapat dari penelitian ini 80% dan 84% dari dua kali percobaan (Orisa, Santoso & Setyawati,2014).

Penelitian lainnya tentang pengklasifikasian penyakit tuberkulosis dengan pendekatan metode *Support Vector Machine (SVM).* Dalam penelitian tersebut terdapat beberapa variable yang sangat berpengaruh terhadap penyakit TB paru sehingga menjadikan tingkat akurasi dalam klasifikasi menggunakan *Support Vector Machine* menjadi tinggi. Pada penelitian ini didapatkan nilai akurasi sebesar 98% (Darsyah,2014).

Berdasarkan adanya permasalahan tersebut maka penulis membangun sebuah sistem *"Klasifikasi Penyakit Kambing Dengan Menggunakan Algoritme Support Vector Machine (SVM)"*. Penerapan algoritme *Support Vector Machine* pada penelitian ini menggunakan kernel trick *Gaussian RBF* karena data pada penelitian ini bersifat non-linier dimana terdapat beberapa kelas penyakit yang memiliki gejala-gejala penyakit yang mirip dengan kelas penyakit lainnya. Pada permasalahan klasifikasi ini menggunakan data non-linier dengan fitur gejala sebanyak 32 gejala dan 11 kelas penyakit yaitu *Cacingan, Endometritis, Kelumpuhan, Kembung, Keracunan, Masistis, Myasis, Orf, Pink Eye, Pneumonia dan Scabies*. Penelitian ini diharapkan akan mendapatkan hasil yang tepat dan akurat untuk pengklasifikasian penyakit kambing berdasarkan gejala-gejala yang muncul.

1.2 Rumusan masalah

Dari uraian latar belakang yang ada, maka didapatkan rumusan masalah dalam penelitian ini, diantaranya sebagai berikut:

1. Bagaimana *Algoritme Support Vector Machine* dapat diimplementasikan dalam pengklasifikasian penyakit kambing?
2. Bagaimana tingkat akurasi yang dihasilkan dari *Algoritme Support Vector Machine* dalam pengklasifikasian penyakit kambing?

1.3 Tujuan

Dari penjelasan pada latar belakang diatas, penulis juga ingin mencapai tujuan antara lain:

1. Menerapkan *Algoritme Support Vector Machine* untuk pengklasifikasian penyakit kambing.
2. Menguji tingkat akurasi klasifikasi penyakit pada kambing menggunakan *Algoritme Support Vector Machine.*

1.4 Manfaat

Manfaat penelitian klasifikasi penyakit pada kambing dengan menggunakan *Algoritme Support Vector Machine (SVM)* adalah sebagai berikut:

1. Bagi Penulis
   - Penelitian sebagai salah satu bentuk dukungan penulis dalam ikut serta mengembangkan teknologi.
   - Memperoleh pengalaman untuk menciptakan sistem cerdas yang bermanfaat untuk masyarakat.
   - Dapat memahami ilmu tentang penyakit kambing dan Algoritme *Support Vector Machine*
2. Bagi Masyarakat Umum
   - Sistem ini dapat membantu untuk pengklasifikasian penyakit kambing berdasarkan gejala – gejala yang timbul.
   - Membantu dokter hewan untuk mengklasifikasi penyakit kambing dengan gejala yang timbul.
3. Bagi Universitas Brawijaya
   - Penelitian ini dapat digunakan sebagai referensi mahasiswa Universitas Brawijaya untuk melakukan penelitian terkait selanjutnya.
   - Sebagai tolak ukur kualitas yang dimiliki oleh mahasiswa untuk dapat meningkatkan mutu pendidikan selanjutnya.

1.5 Batasan masalah

Pada penelitian klasifikasi penyakit kambing ini memiliki batasan masalah penelitian diantaranya adalah sebagai berikut:

1. Data yang dipakai untuk penelitian diperoleh dari penelitian sebelumnya. Penelitian sebelumnya mengambil data dari Dinas Peternakan Provinsi Jawa Timur, UPT PT dan HMT Kabupaten Jember oleh peneliti sebelumnya.
2. Jenis kambing yang diteliti adalah kambing kacang, kambing ettawa, dan kambing peranakan etawa (PE).
3. Penyakit kambing yang ada pada penelitian ini adalah pneumonia, cacingan, endometritis, keracunan, mastitis, myasis, kembung, orf, kelumpuhan, pink eye dan scabies.

1.6 Sistematika pembahasan

Dalam penyusunan skripsi ini memiliki beberapa bab dengan menggunakan kerangka pembahasan yang tersusun sebagai berikut:

BAB 1 PENDAHULUAN

Pada bab pendahuluan ini berisi latar belakang, rumusan masalah, tujuan, batasan masalah dan manfaat penelitian tentang klasifikasi penyakit kambing dengan menggunakan Algoritme *Support Vector Machine* (SVM).

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan digunakan untuk menjelaskan dasar teori yang digunakan dalam penelitian klasifikasi penyakit pada kambing. Landasan kepustakaan nantinya akan menjelaskan tentang keterkaitan objek dan algoritme dengan penelitian sebelumnya.

BAB 3 METODOLOGI PENELITIAN

Pada bab Metodologi Penelitian berisi tentang penguraian langkah kerja dan metode yang dipakai dalam membangun sistem klasifikasi penyakit kambing.

BAB 4 PERANCANGAN

Menganalisa kebutuhan dan perancangan sistem yang akan dibangun dalam klasifikasi penyakit pada kambing dengan Algoritme *Support Vector Machine*.

BAB 5 IMPLEMENTASI

Mengimplementasikan dan penjelasan sistem yang akan dibangun dalam klasifikasi penyakit pada kambing dengan Algoritme *Support Vector Machine.*

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini menjelaskan proses pengujian serta analisa hasil pengujian sistem klasifikasi penyakit pada kambing yang dibangun. Pengujian akan dilakukan tahap pengujian parameter sistem terhadap aplikasi klasifikasi penyakit kambing denan Algoritme *Support Vector Machine.*

BAB 7 PENUTUP

Bab penutup berisi kesimpulan dari hasil penelitian klasifikasi penyakit pada kambing dan berisi saran sebagai pedoman untuk penelitian selajutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian pustaka

Pada penelitian ini didasarkan pada penelitian sebelumnya sudah dilakukan berkaitan dengan penyakit kambing dan penggunaan *Algoritme Support Vector Machine* (SVM). Sumber kajian pustaka pada penelitian ini didapat dari publikasi penelitian terdahulu dalam bentuk paper, jurnal dan skripsi.

Penelitian sebelumnya yang terkait degan penyakit kambing menggunakan metode *Certainty Factor* dimana sistem pakar digunakan untuk mendiagnosis penyakit pada kambing dengan menggunakan metode Certainty Factor. Input dari penelitian ini terdiri dari 56 gejala. Terdapat 10 output penyakit kambing hasil diagnosis penyakit kambing dari sistem pakar ini serta solusi penanganan terhadap penyakit tersebut. Kelebihan dari penelitian ini yaitu memiliki 56 jenis gejala yang menjadi inputan. Kelemahan dari penelitian ini yaitu jenis penyakit hanya ada 10 jenis. Hasil akurasi dari penelitian ini yaitu dari pakar 1 mendapatkan 84% dan dari pakar 2 mendapatkan 80% (Orisa, Santoso, & Setyawati, 2014).

Penelitian tentang pengklasifikasian penyakit tuberkulosis dengan pendekatan metode *Support Vector Machine (SVM)* yang menjelaskan tentang penyakit tuberculosis yang menyerang paru-paru dapat menular antar manusia lainnya. Ada dua variable yang dipakai untuk penelitian tersebut yaitu variable dependen dan independen. Pada variable dependen seseorang dinyatakan terkena penyakit tuberkulosis apabila telah dikonfirmasi oleh hasil pemeriksaaan oleh dokter. Variabel Independen terdapat 5 faktor yang digunakan dalam penelitian antara lain variabel Pendidikan, kebiasaan merokok, pekerjaan, status sosial ekonomi dan kebiasaan konsumsi alkohol. Hasil tangkat akurasi yang didapat pada penelitian ini yaitu 98% (Darsyah,2014). Dari uraian dan penjelasan diatas maka ditunjukkan dalam Tabel 2.1.

2.2 Jenis-jenis penyakit kambing

Jenis - jenis penyakit yang sering menyerang kambing diantaranya adalah sebagai berikut:

1. *Abses*

   Abses merupakan penyakit infeksi penimbunan nanah karena bakteri. bakteri akan masuk ke dalam jaringan yang sehat, dan akan terjadi infeksi dan menimbulkan nanah (Spesialis.info, 2016).

2. *Ascariasis*

   Penyakit ini disebabkan oleh cacing *ascariasis lumbricoides.* Cacing *ascariasis lumbricoides* adalah cacing gelang besar yang panjangnya mencaoai 40 cm dan setebal pensil (MedKes, 2016).

   Beberapa gejala klinisnya adalah demam, batuk kering, mual, diare berdarah dan gizi buruk, terutama pada anak-kambing (MedKes, 2016).

Tabel 2.1 Kajian Pustaka

No.	Penulis	Obyek (Input)	Metode (Proses)	Hasil (Output)	Akurasi
1.	(Orisa, Santoso, & Setyawati, 2014)	Diagnosa penyakit kambing. Inputan terdiri 56 jenis gejala penyakit, yaitu: • Perut keras dan membesar pada bagian kiri • Kambing lemah lesu • Kambing mengeluarkan lidah • Gelisah • Kekurangan zat cair • Sulit/berhenti mengunyah • Susah bernafas • Kambing sulit berdiri • Dst…	*Certainty Factor* Langkah-langkah: 1. Kebenaran gejala dan nilai kepastian gejala klinis. 2. Mengecek aturan setiap gejala inputan. 3. Menambahkan konklusi pada *working memory.* 4. Mulai menghitung nilai kepastian. 5. Mendapatkan jenis penyakit dengan bobot kepastian tertinggi.	Output dari sistem ini yaitu berupa diagnosa 10 penyakit kambing beserta solusi penanganan penyakit kambing dari pakar.	Pakar 1 = 84%, Pakar 2 = 80%
2..	(Darsyah, 2014)	Diagnosa Penyakit Tuberkulosis. Inputan sitem diambil dari data dan variable independen diantaranya : - Pendidikan, - Status sosial, - Merokok, - Konsumsi alkohol - Pekerjaan	*Support Vector Machine*	Sistem akan mendapatkan hasil klasifikasi pasien terserang TBC atau tidak.	98%

3. *Bovine Ephemeral Fever* (BEF)

   Penyakit ini merupakan penyakit yang bersifat ringan gejalanya seperti demam tinggi, otot sakit hingga pincang. Hewan akan cepat sembuh bila tidak terjadi komplikasi.

4. *Bloat* (Kembung)

   Penyakit kembung disebabkan karena gas yang ada di *rumenoreticulum* tidak dapat keluar sehingga menyebabkan jumlah gas menjadi berlebih (Situs-Peternakan, 2016).

5. *Endometris*

   Leleran nanah yang keluar dari uterus kambing betina yang disebabkan oleh *partus*. Salah satu gejalanya yaitu adanya leleran vaginal berwarna putih/putih kekuningan yang akan meningkat pada saat estrus (Dispertanakbatang, 2013).

6. *Enteritis*

   Suatu kondisi medis yang ditandai dengan terjadinya peradangan. Salah satu gejalanya adalah diare yang disertai darah dan kadang kadang dapat menyebabkan disentri.

7. *Masitis*

   Penyakit mastitis adalah penyakit kambing yang menyerang kelenjar susu kambing betina.Gejala pada kambing yang terinfeksi mastitis adalah ambing bengkak dan terasa panas, kambing demam dan nafsu makan berkurang (Suparman, 2014).

8. *Omphalitis*

   Pembengkakan pada pusar anak kambing akibat kurang higenis saat pelepasan tali pusar pada anak kambing. Anak kambing yang baru lahir sebaiknya segera dibersihkan dan diberi betadin pada pusarnya.

9. *Penumonia*

   Penyebab dari penyakit yang menyerang saluran pernafasan ini adalah bakteri *pneumoniace.* Gejala yang timbul adalah kambing batuk parah sampai sulit bernafas, demam tinggi, kambing lemah karena nafsu makan berkurang bahkan sampai kurus (Suparman, 2014).

10. *Retensio Secundinae*

    Plasenta kambing betina tidak dapat terlepas secara alami sehingga menjuntai keluar. Gejalanya yang dapat dilihat adalah selaput fetus yang menggantung diluar alat kelamin, bibir vulva membengkak, kemerahan, dan pada mukosa terdapat bintik-bintik merah.

11. *Scabies*

    Penyakit kulit gatal yang disebabkan bakteri *sarcoptes scabei, psomoptes communis var. ovis, chariopteso ovis.* Kambing yang terkena penyakit scabies merasakan gatal dan sakit sehingga tampak gelisah, sulit untuk beristirahat, nafsu makan berkurang sampai mengganggu pertumbuhan kambing (Suparman, 2014).

12. ORF

    Penyakit orf adalah penyakit menular bahkan dapat menular pada manusia (zoonosis). Penyakit orf atau dapat disebut keropeng disebabkan oleh virus *parapoxvirus* (Susanto & Sitanggang, 2015).

13. *Malignant Catarrhal Fever* (MCF)

    Penyakit ini bisa digolongkan pada penyakit yang bersifat fatal. Penyakit ini disebabkan karena infeksi *alcelaphine herpesvirus-1* atau *ovine herpesvirus-2.*

14. *Myasis*

    Penyakit myasis disebabkan dari infeksi pada kulit kambing yang terluka. Infeksi luka terjadi karena keadaan luas kandang yang sempit dan tidak bersih atau bekas pendarahan saat kambing melahirkan dan tidak ditangani dengan pembersihan, bahkan terjadi pada anak kambing yang baru lahir dan tidak diberi antiseptik pada bagian pusar (Susanto & Sitanggang, 2015).

15. *Pink Eyes*

    Penyakit ini disebabkan oleh bakteri dan virus, menyebabkan mata kambing berair, kemerahan pada bagian yang putih dan kelopaknya, lalu matanya bengkak dan lama kelamaan kornea matanya menjadi keruh atau tertutup lapisan putih (Pudjiatmoko, Syibili, & Nurtanto, 2014).

2.3 Kecerdasan Buatan

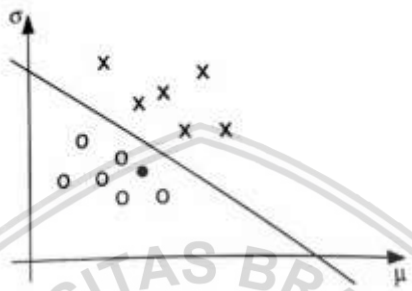
Ilmu pengetahuan yang menggunakan mesin untuk merepresentasikan kecerdasan makhuk hidup seperti pada bagian otak manusia sehingga mampu menyelesaikan permasalahan yang ada (Nugraha &Winiarti,2014). Kecerdasan buatan diciptakan untuk membantu meringankan pekerjaan manusia seperti prediksi ramalan cuaca dan diagnosa penyakit. Ruang lingkup kecerdasan buatan diantaranya yaitu sistem pakar*, computer vision,* robotika dan lainya.

2.4 Klasifikasi

Klasifikasi merupakan pengelompokan data berdasarkan kelas yang ada sebelumnya menjadi kelas-kelas tertantu. Klasifikasi dapat digambarkan pada Gambar 2.1 dimana ada dua kelompok data yang dipisahkan oleh garis. Garis yang memisahkan dua kelompok data tersebut dinamakan *decision line.* Apabila ada data baru disimbolkan dengan titik hitam maka garis inilah yang menjawab terkait

dengan posisi data tersebut. Pada Gambar 2.1 data baru dapat dilihat pada bagian bawah garis sehingga data tersebut masuk kedalam kelas lingkaran. Data merupakan kumpulan banyak fitur yang dapat mendeskripsikan objek. Set data merupakan kumpulan dari beberapa data. Persamaan yang digunakan apabila ada sebuah objek x dengan mempunyai fitur sebanyak n dapat dinyatakan dengan persamaan 2.1 (Prasetyo,2014):

$x = [x1, x2, x3, \dots xn]$ (2.1)



Gambar 2.1 Klasifikasi

Sumber: (Prasetyo,2014)

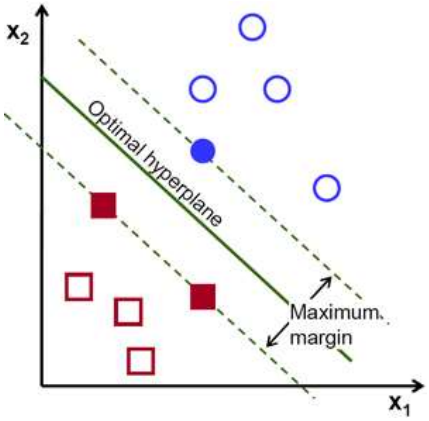
Klasifikasi mempunyai tugas utama yaitu merancang model tersebut dan menggunakan model tersebut (Prasetyo,2014). Dalam merancang pemodelan klasifikasi dan memecahkan masalah pengklasifikasian yang ada dengan cara melihat data latih dan selanjutnya disimpan menjadi data memori. Menggunakan model yang sudah dirancang untuk mengenal data baru sehingga dapat diketahui kelas pada data baru sesuai dengan model data yang sudah disimpan. Dalam pembangunan model dibutuhkan *learning algorithm* seperti *K-Nearest Neighbor, Artificial Neural Network, Support Vector Machine* dan algoritme lainnya (Prasetyo, 2014).

Secara umum klasifikasi terdapat dua proses daiantaranya (Munawarah,2016):

a) Proses *training* adalah suatu proses pembuatan model klasifikasi dengan memakai data training set.
b) Proses *testing* adalah proses pengujian model sesuai dengan model training sehingga dapat diketahui kelasnya.

2.5 *Support Vector Machine* (SVM)

Support Vector Machine adalah sebuah metode yang di pakai untuk melakukan pengklasifikasian dan regresi berdasarkan pembelajaran statistik (Permana,2016). Kelebihan dari *Support Vector Machine* adalah tidak semua data latih terlibat dalam iterasi pelatihannya. Konsep dasar pada *Support Vector Machine* yaitu memaksimalkan batas keputusan (*hyperplane*) untuk menemukan *hyperplane* yang baik agar dapat memisahkan kelas data dalam *input space*.



Gambar 2.2 Margin *Hyperplane*

Sumber : (Prasetyo, 2014)

Hyperplane adalah garis batas keputusan terbaik untuk memisahkan dua kelas dengan cara menghitung margin *hyperplane* maksimal dan mendapatkan titik maksimal. *Margin* adalah jarak terdekat antara data pada kelas dengan garis *hyperplane*. Pada Gambar 2.2 digambarkan *hyperplane* terbaik karena letak garis yang berada pada dua kelas berbeda. Pada data yang tepat dilawati oleh garis putus-putus dapat dinamakan *support vector*. Pada intinya proses pelatihan pada SVM adalah untuk mendapatkan lokasi *hyperplane* yang terbaik (Prasetyo, 2014). *Hyperplane* klasifikasi linier *Support Vector Machine* dituliskan dalam persamaan 2.2.

$$(\boldsymbol{w}.\boldsymbol{x}_i) + \boldsymbol{b} = \mathbf{0} \qquad (2.2)$$

Dataset memiliki nilai data dan memiliki bentuk inialisasi x_i sedangkan tipe kelas pada dataset memiliki bentuk inisialisasi y_i. *Hyperplane* dalam metode ini memisahkan du akelas dengan nilai pemisahan kelas yaitu 1 dan -1 dengan Persamaan 2.3 dan Persamaan 2.4.

$$(w.x_i + b) \geq 1, y_i = 1 \qquad (2.3)$$

$$(w.x_i + b) \leq 1, y_i = -1 \qquad (2.4)$$

Keterangan :

w = Parameter untuk bobot *support vector*

b = Parameter untuk bias

x_i = Data ke-i

y_i = Kelas pada data ke-i

Paramter support vector adalah sebuah garis vector tegak lurus terhadap titik pusat dan *hyperplane*. Nilai parameter bias yaitu nilai dari garis kordinat yang relatif terhadap titik pada koordinatnya. Untuk menadapatkan nilai parameter w dan b dijelaskan dalam persamaan 2.5 dan persamaan 2.6.

$$b = -\frac{1}{2}(w.x^{+} + w.x^{-}) \quad (2.5)$$

$$w = \sum_{i=1}^{n} a_i y_i K(x_i.x) \quad (2.6)$$

Keterangan :

b = Parameter untuk bias

$w.x+$ = Nilai bobot kelas data bernilai positif

$w.x-$ = Nilai bobot kelas data bernilai negatif

w = Parameter untuk bobot *support vector*

α_i = Parameter nilai *alpha*

y_i = Kelas pada data ke-i

$K(x_i . x)$ = Hasil nilai *kernel*

Setiap data yang berhubungan dngan αi dan bernilai positif dapat disebut support vectornya karena parameter nilai αi yaitu nilai bobot bernilai positif (Nugroho, 2003). Persamaan 2.7 dan persamaan 2.8 berikut adalah persamaan yang digunakan untuk hasil dari pengklasifikasian data uji yang digunakan.

$$f(x) = (w.x + b) \text{ atau } f(x) = \sum_{i=0}^{m} a_i y_i K(x_i.x) + b \quad (2.7)$$

$$\text{Fungsi klasifikasi} = sign(f(x)) \quad (2.8)$$

Keterangan :

α_i = Parameter nilai *alpha*

y_i = Kelas pada data ke-i

m = Jumlah data

$K(x_i . x)$= Hasil nilai *kernel*

b = Parameter untuk bias

Ada dua fungsi hasil dari klasifikasi yaitu apabila nilai $f(x)$ positif maka nilai $sign(f(x))$ adalah 1 dan apabila nilai $f(x)$ negative maka nilai $sign(f(x))$ yaitu -1. Dari fungsi hasil pengklasifikasian tersebut data dapat digolongkan sesuai dengan kelasnya. Data uji bernilai 1 termasuk kedalam data kelas 1 dan data uji bernialai -1 termasuk kedalam data pada kelas 2.

2.5.1 Support Vector Machine Nonlinier

Support Vector Machine dalam penggunaanya sering digunakan dalam kasus dengan data-data linier yang hanya ada dua kelas. Namun pada saat ini banyak kasus yang bersifat non-linier. Sehingga digunakan cara untuk mengubah dari sifat non-Iinier menjadi kedalam sifat linier. Dalam metode *Support Vector Machine* untuk mentransformasi data non linier ke dalam data linier dibutuhkan kernel. Fungsi kernel atau yang disebut *kernel trick* adalah pengelompokan data dari dimensi yang rendah ke dimensi yang tinggi (Prasetyo, 2012). Berikut beberapa fungsi kernel yang digunakan pada metode *Support Vector Machine* non-linier yang ditujukan pada Tabel 2.2:

Tabel 2. 2 Fungsi Kernel

Nama Kernel	Fungsi Kernel
Linier	$K(x,y)=x.y$
Polynomial	$K(x,y)=(x.y+c)^d$
Gaussian RBF	$K(x,y) = \exp(-\frac{\|\|x-y\|\|^2}{2.\sigma^2})$
Sigmoid	$K(x,y)=\tanh(\sigma(x.y)+c)$

Sumber: (Prasetyo,2012)

2.5.2 Sequential Training *Support Vector Machine*

Sequential Training *Support Vector Machine* digunakan dalam melakukan proses training. Selain itu untuk melakukan proses training ada berbagai macam proses training lainnya yaitu *Quadratic Programming* yang dapt dikatakan algotitme yang kompleks namun membutuhkan waktu yang lama untuk proses trainingnya dan proses *Sequential Minimal Optimization* yang termasuk dalam pengambangan dari algoritme *Quadratic Programming*. Algoritme *Sequential Training* memiliki konsep yang mudah dipahami dan waktu yang yang lebih minimal (Mase.J,2018). Langkah-langkah Algoritme yang ada pada *Sequential Training Support Machine* adalah sebagai berikut:

1. Inisialisasi nilai parameter yang ada seperti C, $\lambda, \gamma, \varepsilon$.
2. Menghitung nilai matriks *hessian* dengan rumus pada persamaan 2.9.

$$D_{ij} = y_i y_j (K(xixj) + \lambda^2 \quad (2.9)$$

Keterangan :

xi = Data ke-i

xj = Data ke-j

yi = Kelas pada data ke-i

yj = Kelas pada data ke-j

n = Jumlah data

$K(xi\ ,\ xj\ )$= Hasil nilai *kernel*

3. Perulangan pada tahap ketiga sampai menemukan nilai iterasi maksimum atau mencapai nilai $max\ (|\delta\alpha i\ |) < \varepsilon\ (epsilon)$.

$$E_i = \sum_{j=1}^{1} a_i D_{ij} \quad (2.10)$$

$$\delta a_i = \min\{max[\gamma(1 - E_i), -a_i], C - a_i\} \quad (2.11)$$

$$a_i = a_i + \delta a_i \quad (2.12)$$

Keterangan :

α_i = Parameter *alpha*

Dij = Nilai hasil matriks *Hessian*

C = Parameter C

$\delta\alpha i$ = Parameter delta alfa ke-i

4. Tahap terakhir yaitu mendapatkan nilai *Support Vector* (SV), SV = ($\alpha_i >$ *TresholdSV*). Nilai *Trashold*SV dilakukan berulang untuk mendapatkan nilai *TresholdSV* ≥ 0.

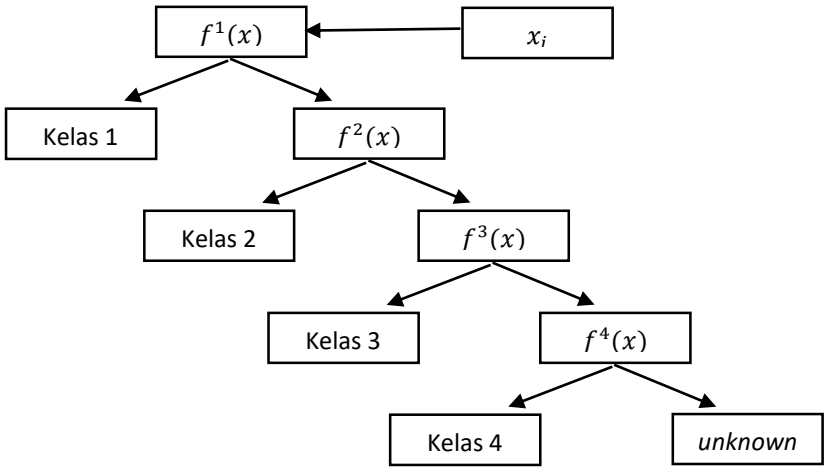
2.5.3 *One Against All*

Metode yang digunakan untuk permasalahan yang *multi-class* yaitu dengan *One Against All* (OAA) dengan membuat jumlah kelas dalam model SVM biner. Pada setiap model klasfikasi ke- i akan dlatih dengan seluruh data dalam menyelesaikan solusi permasalahan. Contoh permasalahan klasifikasi dengan empat kelas, maka dalam proses latih digunakan empat model SVM biner yang ditunjukkan Tabel 2.3 dan penerapannya pada Gambar 2.3

Tabel 2.3 Contoh SVM degan *One Against All*

$yi = 1$	$yj = -1$	Hipotesis
Kelas 1	Bukan kelas 1	$f^1(x) = (w^1)x + b^1$
Kelas 2	Bukan kelas 2	$f^2(x) = (w^2)x + b^2$
Kelas 3	Bukan kelas 3	$f^3(x) = (w^3)x + b^3$
Kelas 4	Bukan kelas 4	$f^4(x) = (w^4)x + b^4$

Sumber : (Sembiring, 2007)



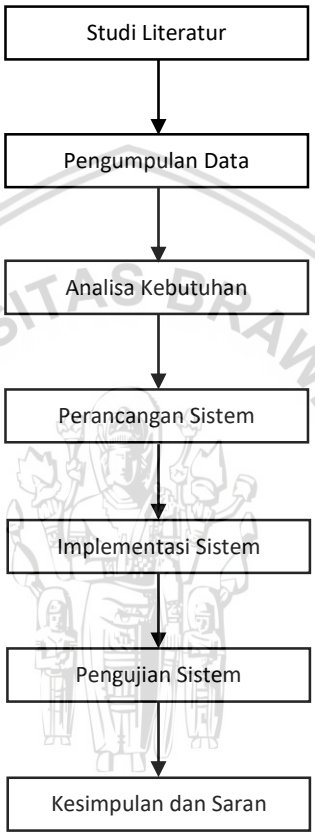
Gambar 2. 3 Contoh Klasifikasi dengan *One Against All*

2.6 K-Fold Cross Validation

K-Fold cross validation adalah metode pengujian hasil akurasi sistem dengan membagi data sebanyak nilai k. Data latih dan data uji akan dibagi sesuai dengan jumlah *m fold* sebagai data uji dan *k-m fold* sebagai data latih (Haryanto, D.J, 2018). Untuk mendapatkan nilai akurasi yaitu dengan menghitung rata – rata nilai akurasi setiap iterasi dan dibagi dengan jumlah nilai *k-fold.*

BAB 3 METODOLOGI

Metodologi penelitian pada pengklasifikasian ini melalui beberapa tahapan diantaranya studi literatur, pengumpulan data penelitian, analisa kebutuhan pada sistem, perancangan pada sistem, pengimplementasian pada sistem, pengujian pada sistem serta kesimpulan dan saran. Tahapan metodologi penelitian disajikan dalam diagram pada Gambar 3.1.



Gambar 3. 1 Diagram Metodologi Penelitian

3.1 Studi literatur

Studi literatur adalah metode untuk mendapatkan teori pendukung penelitian dan menjadi bahan dasar dalam penelitian ini. Literatur tersebut dapat diperoleh dari buku, jurnal, ebook, dan dokumentasi projek. Bagian studi literatur ini mencakup teori diantaranya sebagai berikut:

1. Metode *Support Vector Machine*
2. Penyakit pada kambing

3.2 Pengumpulan data

Dalam tahap pengumpulan data dilakukan untuk digunakan untuk penelitian. Penelitian ini menggunakan data sekunder yang telah dikumpulkan oleh peneliti sebelumnya. Peneliti sebelumnya melakukan observasi dan wawancara di Dinas Peternakan Provinsi Jawa Timur (UPT PT dan HMT Kabupaten Jember) untuk mendapatkan data yang nantinya digunakan sebagai data training metode *Support Vector Machine (SVM).* Data yang digunakan dalam penelitian ini sebanyak 148 data penyakit kambing yang terbagi menjadi 11 kelas penyakit dan 32 gejala. Sebelas penyakit kambing tersebut adalah *Cacingan, Endometritis, Kelumpuhan, Kembung, Keracunan, Masistis, Myasis, Orf, Pink Eye, Pneumonia* dan *Scabies.*

3.3 Analisa kebutuhan

Analisa kebutuhan menjelaskan mengenai haI-haI yang dibutuhkan dalam menunjang penelitian ini. Beberapa kebutuhan yang akan digunakan penelitian ini yaitu kebutuhan pada perangkat keras, kebutuhan perangkat lunak, dan kebutuhan data klasifikasi penyakit kambing.

1. Kebutuhan Perangkat Keras
   - Laptop dengan Processor Intel® Core™ i7-3612QM CPU @ 2.10GHz 2.1 GHz
   - RAM 8 GB
2. Kebutuhan Perangkat Lunak
   - Sistem operasi Windows 10
   - NetBeans IDE 8.2
   - Web browser: Google Chrome
   - Bahasa Pemrograman: *PHP, HTML, CSS* dan *JavaScript*
3. Kebutuhan Data
   - Data sekunder dari penelitian sebelumnya mengenai klasifikasi penyakit kambing dari Dinas Peternakan Provinsi Jawa Timur (UPT PT dan HMT Kabupaten Jember)

3.4 Perancangan sistem

Perancangan sistem adalah segala perancangan proses yang ada pada sistem yang dibangun dari analisis data hingga penerapan metode yang akan dipakai. Perancangan pada sistem ini dijelaskan pada flowchart di setiap langkah proses pada sistem. Flowchart menjelaskan proses mulai mengambil data dan di proses dengam metode SVM, alur proses SVM dan alur perhitungan SVM

3.5 Implementasi sistem

Tahap Implementasi sistem merupakan tahap mulai membangun sistem yang sudah dirancang sesuai dengan penerapan metode klasifikasi *Support Vector Machine*. Fase proses yang diterapkan dalam sistem antara lain:

1. Implementasi antarmuka sistem.
2. Implementasi algoritma klasifikasi yang digunakan yaitu *Support Vector Machine* kedalam bahasa pemrograman *Java.*
3. Output sistem yang berupa hasil klasifikasi penyakit pada kambing.

3.6 Pengujian sistem

Pada proses pengujian ini untuk mengetahui keberhasilan dari sistem klasifikasi sehingga mendapatkan nilai tingkat akurasi dari sistem yang berjalan. Pengujian dilakukan dengan mengolah data pada system klasifikasi untuk diuji. Sistem akan mendapatkan tingkat akurasi terhadap klasifikasi penyakit kambing setelah semua proses dijalankan sesuai dengan kebutuhan dan rancangan sebelumnya.

3.7 Kesimpulan dan Saran

Setelah melakukan tahap-tahap yang telah diuraikan diatas seperti perancangan, implementasi, dan pengujian sistem, maka dapat ditarik sebuah kesimpulan yang dapat menjawab rumusan masalah berdasar hasil yang telah didapatkan dalam penelitian. Selain itu juga dapat menghsilkan saran untuk pengembangan sistem yang selanjutnya agar lebih baik.

BAB 4 PERANCANGAN

4.1 Formula Permasalahan

Permasalahan utama pada penelitian ini yaitu mengklasifikasikan penyakit kambing dengan terdapat 11 penyakit kambing antara lain yaitu penyakit *Cacingan, Endometritis, Kelumpuhan, Kembung, Keracunan, Masistis, Myasis, Orf, Pink Eye, Pneumonia* dan *Scabies.* Klasifikasi dilakukan menggunakan algoritme SVM dimana kelas yang mempunyai nilai klasifikasi paling besar akan menentukan penyakit yang diderita oleh kambing tersebut. Terdapat sebanyak 32 parameter gejala penyakit pada kambing yang dipakai dalam penelitian.

Untuk menentukan kelas data latih dalam setiap level dipakai konsep *One Againts All*. Selanjutnya dilakukan tahapan proses pada SVM, yaitu menghitung nilai *kernel* dan *sequential training* SVM. Dari hasil perhitungan tersebut didapatkan nilai α_1 serta nilai *bias* pada setiap level. Strategi *One-Against-All* digunakan dalam menentukan kelas data uji.

4.1.1 Deskripsi Data

Pada penelitian ini data didapatkan dari penelitian sebelumnya yang dilakukan langsung oleh peneliti sebelumnya pada Dinas Peternakan Provinsi Jawa Timur (UPT PT dan HMT Kabupaten Jember). Data penyakit kambing yang didapat berisi 32 gejala penyakit kambing sebagai parameter serta 11 kelas penyakit kambing. Pada proses perhitungan manual SVM diambil sebanyak 22 data latih dan 11 data uji. Data latih yang digunakan dibagi dalam 11 kelas penyakit kambing antara lain *Cacingan, Endometritis, Kelumpuhan, Kembung, Keracunan, Masistis, Myasis, Orf, Pink Eye, Pneumonia* dan *Scabies.* Data yang digunakan sebanyak 148 data dengan 32 gejala penyakit kambing. Gejala-gejala penyakit kambing tersebut yaitu :

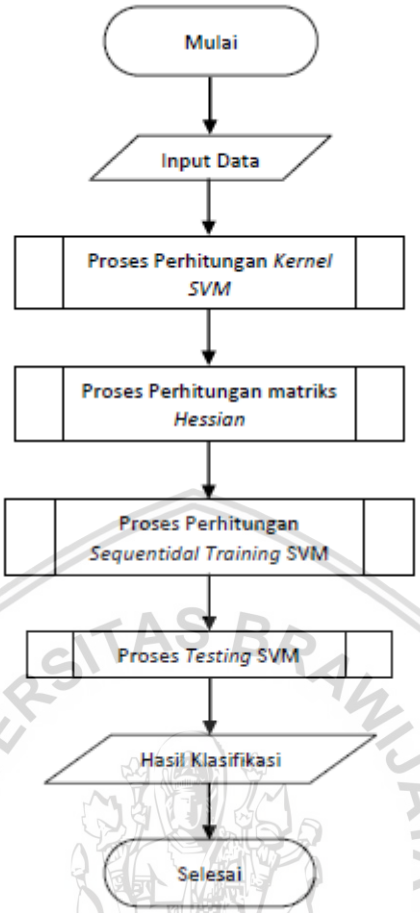
Tabel 4.1 Gejala – Gejala Penyakit Kambing

No	Kode Gejala	Nama Gejala
1	G001	Ambing bengkak berwarna kemerahan
2	G002	Bengkak di sekitar luka
3	G003	Bulu kusam terasa kasar
4	G004	Bulu rontok
5	G005	Bulu rontok pada bagian terinfeksi
6	G006	Demam
7	G007	Depresi
8	G008	Diare
9	G009	Diare berdarah
10	G010	Gatal-gatal

11	G011	Kaku saat berjalan
12	G012	Kejang-kejang
13	G013	Keluar belatung dari kulit luka
14	G014	Keluar ingus
15	G015	Keluar lendir pada vulva
16	G016	Keropeng di mulut
17	G017	Kornea keruh
18	G018	Kulit kasar dan bersisik
19	G019	Kurus
20	G020	Lemah lesu
21	G021	Mata merah
22	G022	Menggosokkan kulit ke dinding kandang
23	G023	Mulut berbusa
24	G024	Nafas berbau busuk
25	G025	Nafsu makan berkurang
26	G026	Perubahan warna susu dan terdapat gumpalan pada susu
27	G027	Perut sebelah kiri membesar dan terasa sakit
28	G028	Produksi susu menurun
29	G029	Sesak nafas
30	G030	Tidak dapat berdiri
31	G031	Tumbuh bintil-bintil kecil pada telinga
32	G032	Vulva berbau busuk

4.2 Perancangan Proses Algoritme *Support Vector Machine*

Pada proses Algoritme *Support Vector Machine* akan menjelaskan mengenai alur penyelesaian masalah pada metode *Support Vector Machine*. Tahap awal adalah memilih data inputan, selanjutnya menghitung kernel SVM, menghitung matriks *hessian* , proses training data, testing data dengan SVM dan didapatkan hasil klasifikasi. Tahapan alur algoritme SVM akan dijelaskan pada Gambar 4.1



Gambar 4.1 Proses Alur *Support Vector Machine*

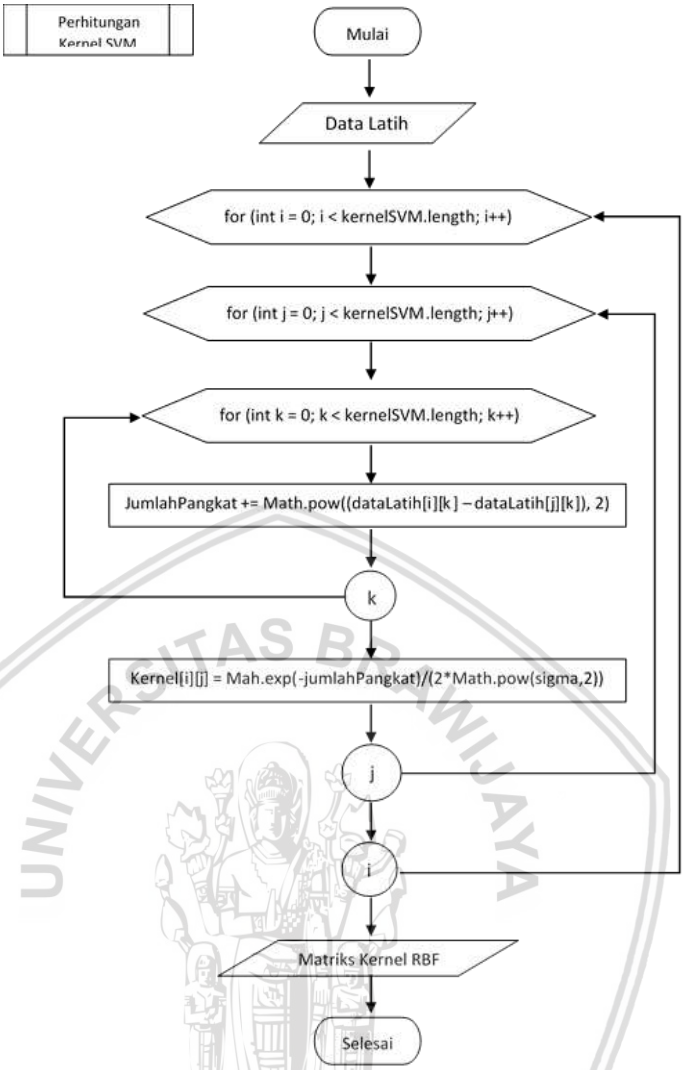
Alur proses algoritma SVM sesuai Gambar 4.1 adalah :

1. Input data
2. Menghitung *Kernel RBF*
3. Menghitung data latih dengan *Sequential Training* SVM
4. Menghitung *testing* SVM dalam menentukan kelas level.
5. Hasil klasifikasi yang dikeluarkan dari pengujian data uji.

4.2.1 Proses Perhitungan Kernel *Support Vector Machine*

Perhitungan kernel SVM dilakukan dengan mengolah data latih dengan kernel RBF. Penjelasan alur perhitungan kernel *Support Vector Machine* Pada Gambar 4.2 adalah sebagai berikut :

1. Mendapatkan data latih
2. Perulangan pada jumlah data latih
3. Proses menghitung kernel
4. Output nilai matriks Kernel RBF



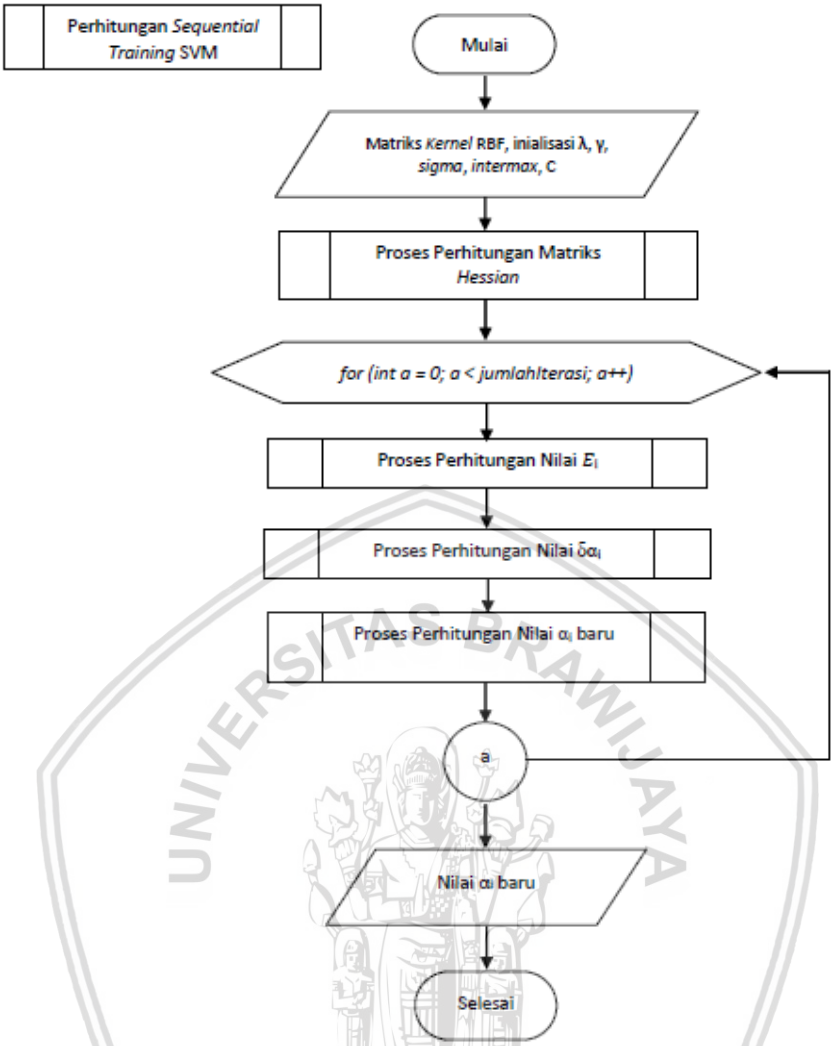
Gambar 4.2 Proses Perhitungan Kernel SVM

4.2.2 Proses Perhitungan Sequential Training SVM

Untuk mendapatkan garis *hyperplane* yang baik dilakukan perhitungan Sequential Training SVM. Alur proses dari perhitungan Sequential Training SVM akan dijelaskan pada Gambar 4.3

Penjelasan dari Gambar 4.3 adalah :

1. Input nilai matriks kernel RBF,λ lambda,γ gamma, sigma, iterasi maksimum dan complexity
2. Melakukan perhitungan pada matriks hessian
3. Menghitung nilai Ei
4. Menghitung nilai δαi
5. Proses menentukan αi yang baru
6. Jika mencapai nilai Itermax maka iterasi akan berhenti
7. Output adalah nilai αi

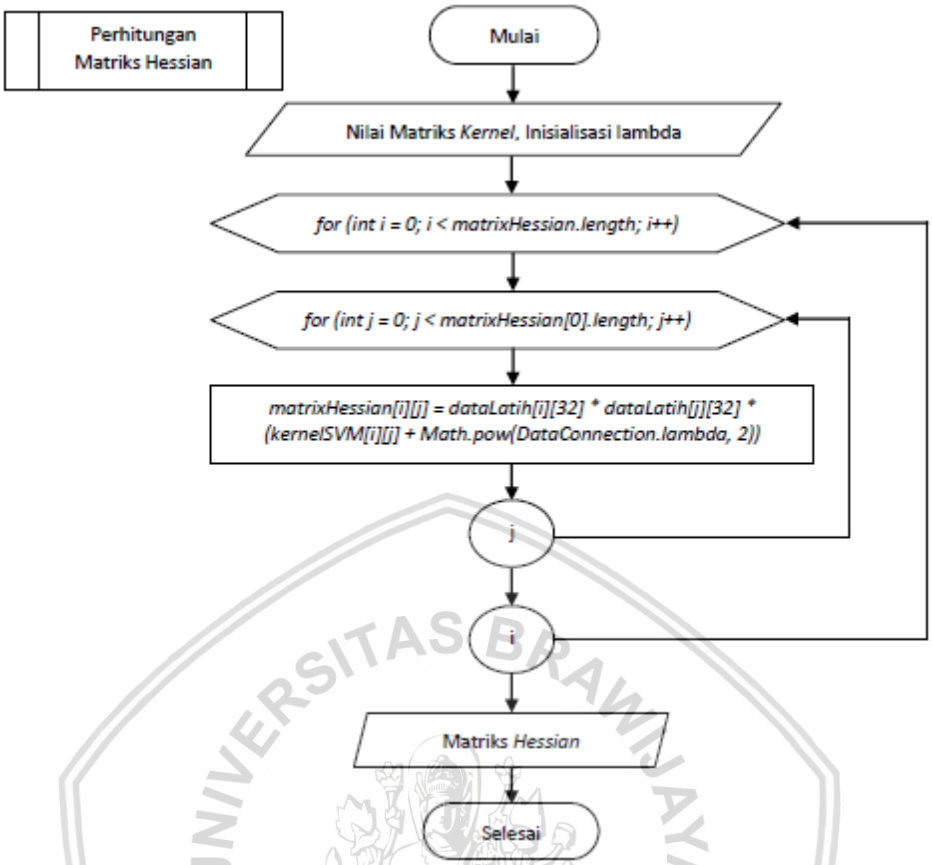


Gambar 4.3 Proses Perhitungan Sequential Training *Support Vector Machine*

4.2.3 Proses Perhitungan Matriks Hessian

Pada proses perhitungan Matriks Hessian akan ditunjukan pada Gambar 4.3 dan alur penjelasan perhitungan Matriks Hessian sebagai berikut :

1. Input matriks *kernel* , dan parameter λ (lambda)
2. Mulai menghitung matriks hessian
3. Perulangan terhadap data latih
4. Hasil matriks hessian

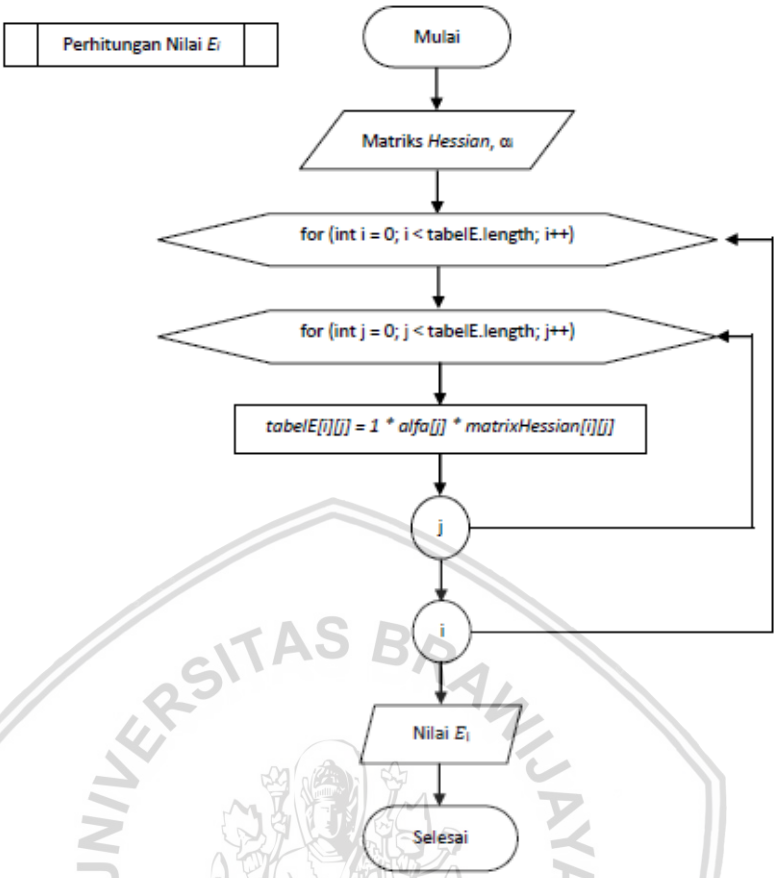


Gambar 4. 4 Proses Perhitungan Matriks Hessian

4.2.4 Proses Perhitungan Nilai Ei

Proses pada perhitungan Ei ditunjukan pada Gambar 4.4. Berikut adalah penjelasan alur proses perhitungan Ei :

1. Input nilai pada matriks hessian dan αi
2. Proses menghitung nilai Ei
3. Perulangan pada data latih dan iterasi untuk menghitung nilai Ei
4. Hasil berupa nilai Ei

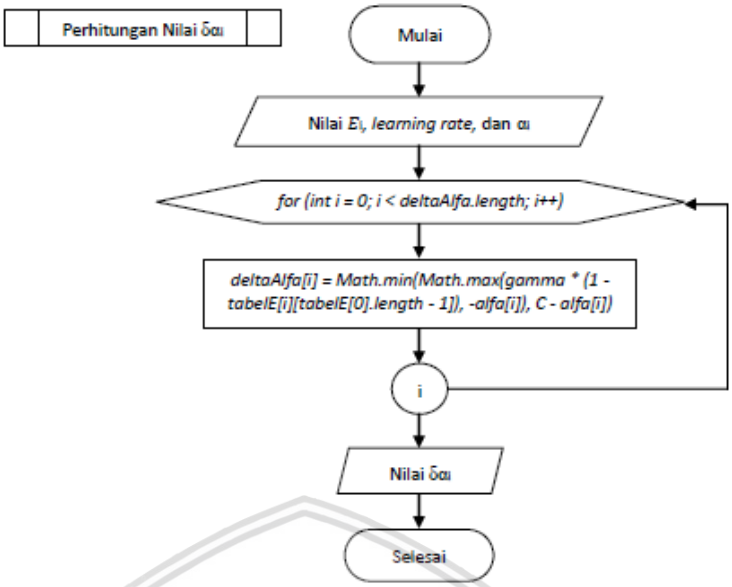


Gambar 4.5 Proses Perhitungan Nilai Ei

4.2.5 Proses Perhitungan Nilai δαi

Pada Gambar 4.5 akan ditunjukan proes alur menghitung nilai δαi beserta penjelasan alur proses menghitung nilai δαi sebagai berikut :

1. Input nilai Ei, *learning rate,* dan nilai αi
2. Proses menghitung nilai δαi
3. Melakukan perulangan data latih sejumlah data latih
4. Hasil berupa nilai δαi

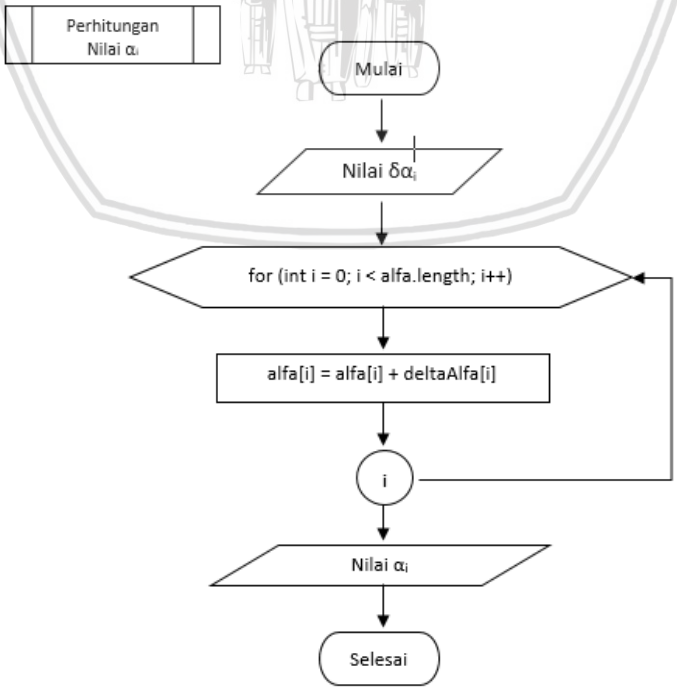


Gambar 4.6 Proses Perhitungan Nilai δαi

4.2.6 Proses Perhitungan Nilai αi

Proses perhitungan nilai αi akan dijelasan alur proses perhitungan nilai αi pada Gambar 4.7 sebagai berikut:

1. Input nilai δαi
2. Perhitungan pada nilai αi yang baru
3. Proses perulangan sesuai jumlah data latih
4. Hasil keluaran nilai αi yang baru

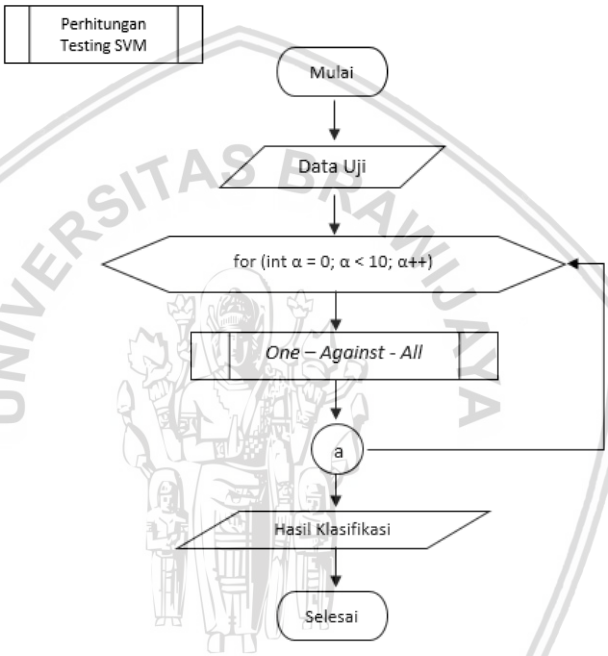


Gambar 4.7 Proses Perhitungan Nilai α_i

4.2.7 Proses Testing *Support Vector Machine*

Penjelasan proses Testing *Support Vector Machine* beserta alur proses dan perhitungan pada Gambar 4.8 adalah sebagai berikut :

1. Input nilai data uji
2. Pengujian nilai pada data latih dan didapatkan hasil klasifikasai penyakit kambing.
3. Menghitung fungsi sign(f(x)) setiap data uji yang akan diuji untuk mendaptkan hasil klasifikasi.
4. Hasil output klasifikasi dari data uji jika bernilai negatif maka dilanjutkan pada level selanjutnya menggunakan strategi *One-Against-All.*



Gambar 4.8 Proses Testing SVM

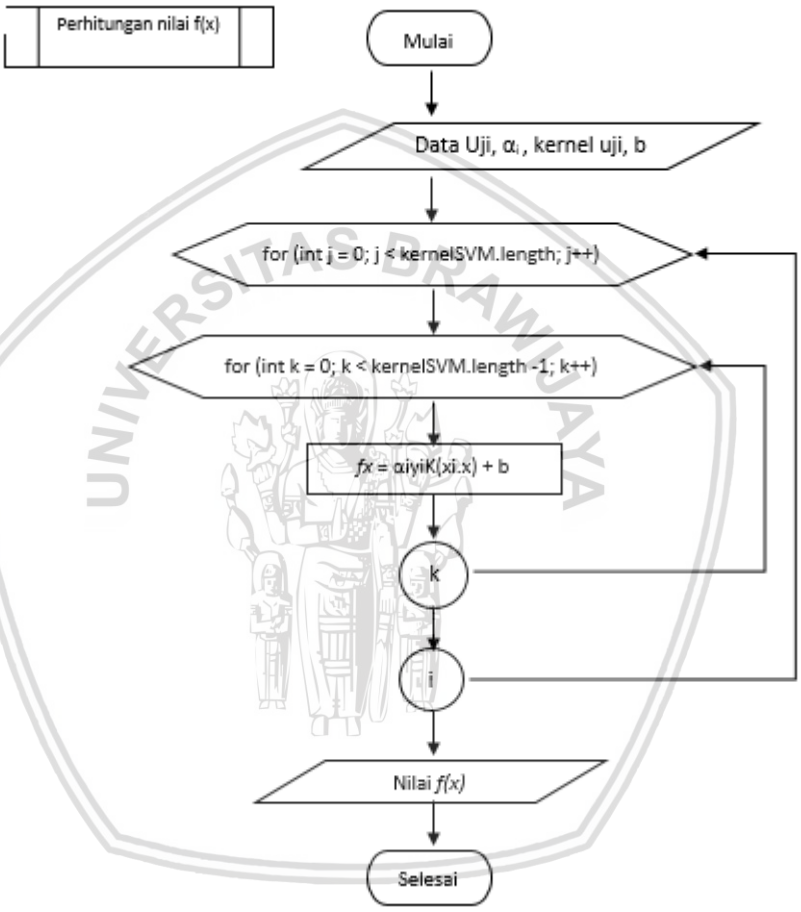
4.2.8 One-Againts-All

One Againts All adalah sebuah strategi klasifikasi dalam menyelesaikan permasalahan pada multi class dalam metode *Support Vector Machine*. Pada penelitian ini strategi One Againts All digunakan untuk membantu mengatasi masalah data penyakit kambing dengan pengklasifikasian 11 kelas penyakit kambing. Strategi ini membuat 10 level yang pada level pertama kelas 1 sebagai kelas positif satu (+1) dan kelas selanjutnya yaitu kelas 2,3,4,5,6,7,8,9,10 dan 11 sebagai kelas negative satu (-1). Pada level dua, kelas 2 menjadi kelas positif satu (+1) dan kelas selanjutnya 3,4,5,6,7,8,9,10 dan 11 sebagai kelas negative satu (-1). Sampai pada level sepuluh memisahkan kelas 10 sebagai kelas positif satu (+1) dan kelas 11 sebagai kelas negative satu (-1). Untuk menentukan kelas data pada data uji maka data akan di proses dimana saat data uji berupa nilai negative maka dilanjutkan pada level selanjutnya dan jika nilai yang didapat postif maka selesai.

4.2.9 Proses Perhitungan Nilai f(x)

Proses pada Perhitungan Nilai f(x) akan dijelaskan alur proses perhitungan Nilai f(x) adalah sebgai berikiu :

1. Input nilai data uji yang sudah terilih, nilai αi terbaru, nilai K(Xi,Xj) terhadap data testing dan nilai bias
2. Menghitung nilai f(x) dengan persamaan f(x) = αi*yi*K(Xi,Xj) dimana nilai yi adalah nilai kelas data uji.
3. Output berupa nilai f(x) data uji



Gambar 4.9 Proses perhitungan nilai *f(x)*

4.3 Perhitungan Manual *Support Vector Machine*

Pada perhitungan manual *Support Vector Machine* ini menggunakan 22 data latih dan 11 data uji. Awal dari proses ini yaitu menghitung nilai kernel yang ada pada data latih. Data Latih dapat dilihat pada Table 4.2. Data pada tahap perhitungan manulalisasi merupakan data yang ada pada setiap kelas dalam penelitian. Data yang digunakan memiliki 11 kelas, maka digunakan strategi *One-Againts-All*.

4.3.1 Perhitungan Manual Kernel *Support Vector Machine*

Kernel pada perhitungan manual penelitian ini yaitu kernel RBF.Pada Tabel 4.2 merupakan data pertama di level 1 pada manualisasi dan hasil perhitungan kernel RBF ditujukkan pada Tabel 4.4.

Tabel 4.2 Data Latih Penyakit Kambing Pada Level 1

No.	Data ke-	GEJALA										Kelas		Kelas Level 1
		G1	G2	G3	G4	G5	G6	G7	G8	G...	G32			
1	1	0	0	0	1	0	0	0	1	...	0	Cacingan	1	1
2	2	1	0	0	1	0	0	1	1	...	0	Cacingan	1	1
3	17	0	0	0	0	0	0	0	0	...	1	Endometritis	2	-1
4	18	0	0	0	0	0	0	0	0	...	1	Endometritis	2	-1
5	22	0	0	0	0	0	0	0	0	...	0	Kelumpuhan	3	-1
6	27	0	0	0	0	0	0	0	0	...	0	Kelumpuhan	3	-1
7	33	1	0	0	0	0	1	1	1	...	0	Kembung	4	-1
8	42	0	0	0	0	0	1	0	1	...	0	Kembung	4	-1
9	50	0	0	0	0	0	0	0	1	...	0	Keracunan	5	-1
10	54	0	0	0	0	0	0	0	0	...	0	Keracunan	5	-1
11	59	1	1	0	0	0	1	1	0	...	0	Mastitis	6	-1
12	65	1	0	0	0	0	1	0	0	...	0	Mastitis	6	-1
13	74	1	0	0	0	1	1	0	0	...	0	Myasis	7	-1
14	80	1	1	0	0	0	1	0	0	...	0	Myasis	7	-1
15	88	0	1	0	0	1	0	0	0	...	0	Orf	8	-1
16	92	0	0	0	0	1	0	0	0	...	0	Orf	8	-1
17	107	1	1	0	0	0	0	1	0	...	0	Pink Eye	9	-1
18	115	0	0	0	0	0	1	1	0	...	0	Pink Eye	9	-1
19	121	0	0	0	0	0	1	0	0	...	0	Pneumonia	10	-1
20	125	0	0	0	0	0	1	0	0	...	0	Pneumonia	10	-1
21	131	0	0	0	1	1	0	1	0	...	0	Scabies	11	-1
22	145	0	0	0	0	1	0	0	0	...	0	Scabies	11	-1

Tabel 4.3 Data Latih ke-1 Pada Level 1

No.	Data ke-	GEJALA										Kelas		Kelas Level 1
		G1	G2	G3	G4	G5	G6	G7	G8	G...	G32			
1	1	0	0	0	1	0	0	0	1	...	0	Cacingan	1	1

Tahapan dalam menghitung nilai pada kernel RBF adalah sebagai berikut :

1. Inisialisai parameter σ = 1
2. Proses perhitungan kernel RBF seperti contoh perhitungan kernel RBF dengan data ke-1.

$$K(x, x') = \exp(-\frac{||x - x'||^2}{2.\sigma^2})$$

= EXP(-(((0-0)^2+(0-0)^2+(0-0)^2+(1-1)^2+(0-0)^2+(0-0)^2+(0-0)^2+(1-1)^2+(0-0)^2+(0-0)^2+(0-0)^2+(0-0)^2+(0-0)^2+(0-0)^2+(1-1)^2+(0-0)^2+(0-0)^2+(0-0)^2+(0-0)^2+(1-1)^2+(1-1)^2+(0-0)^2+(0-0)^2+(0-0)^2+(1-1)^2+(1-1)^2+(0-0)^2+(1-1)^2+(1-1)^2+(0-0)^2+(0-0)^2+(0-0)^2+(0-0)^2))/2*1^2)
= 1

Tabel 4.4 Hasil Perhitungan Kernel RBF Level 1

k(x,y)	1	2	3	.....	22
1	1	0.082084998624	0.004086771438		0.000911881966
2	0.082084998624	1	0.002478752177		0.000553084370
3	0.004086771438	0.002478752177	1		0.011108996538
4	0.002478752177	0.001503439193	0.606530659713		0.006737946999
5	0.011108996538	0.006737946999	0.135335283237		0.011108996538
6	0.006737946999	0.004086771438	0.082084998624		0.006737946999
7	0.011108996538	0.018315638889	0.006737946999		0.000553084370
8	0.030197383422	0.006737946999	0.018315638889		0.001503439193
9	0.004086771438	0.006737946999	0.006737946999		0.000553084370
10	0.004086771438	0.006737946999	0.006737946999		0.000553084370
11	0.004086771438	0.018315638889	0.018315638889		0.001503439193
12	0.006737946999	0.004086771438	0.030197383422		0.002478752177
13	0.000553084370	0.000911881966	0.006737946999		0.030197383422
14	0.000335462628	0.000553084370	0.004086771438		0.018315638889
15	0.002478752177	0.001503439193	0.082084998624		0.049787068368
16	0.002478752177	0.001503439193	0.082084998624		0.135335283237
17	0.001503439193	0.006737946999	0.018315638889		0.001503439193
18	0.002478752177	0.004086771438	0.082084998624		0.006737946999
19	0.011108996538	0.002478752177	0.049787068368		0.004086771438
20	0.006737946999	0.004086771438	0.082084998624		0.006737946999
21	0.002478752177	0.004086771438	0.004086771438		0.135335283237
22	0.000911881966	0.000553084370	0.011108996538		1

4.3.2 Perhitungan Manual Matriks *Hessian*

Proses menghitung matriks hessian didapatkan setelah mendapatkan nilai perhitungan kernel RBF. Berikut adalah penjelasan tentang perhitungan matriks *hessian.*

1. Inisialisasi parameter λ = 0,5
2. Menghitung matriks hessian dengan rumus :

$$D_{ij} = y_i y_j (K(x_i, x_j)) + \lambda^2$$

y_i = 1 untuk kelas data pertama

y_j = 1 untuk kelas data kedua.

(xi, xj) = hasil nilai kernel.

Dibawah akan ditunjukkan perhitungan data ke-1 sebagai berikut :

$$D_{11} = y_1 y_1 (K(x_1, x_1)) + \lambda^2$$
$$= 1.1\ (1 + 0{,}5^2)$$
$$= 1{,}25$$

Berikut adalah hasil matriks *hessian* pada data latih level 1 yang ditunjjukan pada Tabel 4.5.

Tabel 4.5 Hasil Perhitungan Matriks Hessian Level 1

Dij	1	2	3	...	22
1	1.25	0.332084998624	-0.254086771438	...	-0.250911881966
2	0.332084998624	1.25	-0.252478752177	...	-0.250553084370
3	-0.254086771438	-0.252478752177	1.25	...	0.261108996538
4	-0.252478752177	-0.251503439193	0.856530659713	...	0.256737946999
5	-0.261108996538	-0.256737946999	0.385335283237	...	0.261108996538
6	-0.256737946999	-0.254086771438	0.332084998624	...	0.256737946999
7	-0.261108996538	-0.268315638889	0.256737946999	...	0.250553084370
8	-0.280197383422	-0.256737946999	0.268315638889	...	0.251503439193
9	-0.254086771438	-0.256737946999	0.256737946999	...	0.250553084370
10	-0.254086771438	-0.256737946999	0.256737946999	...	0.250553084370
11	-0.254086771438	-0.268315638889	0.268315638889	...	0.251503439193
12	-0.256737946999	-0.254086771438	0.280197383422	...	0.252478752177
13	-0.250553084370	-0.250911881966	0.256737946999	...	0.280197383422
14	-0.250335462628	-0.250553084370	0.254086771438	...	0.268315638889
15	-0.252478752177	-0.251503439193	0.332084998624	...	0.299787068368
16	-0.252478752177	-0.251503439193	0.332084998624	...	0.385335283237
17	-0.251503439193	-0.256737946999	0.268315638889	...	0.251503439193
18	-0.252478752177	-0.254086771438	0.332084998624	...	0.256737946999
19	-0.261108996538	-0.252478752177	0.299787068368	...	0.254086771438
20	-0.256737946999	-0.254086771438	0.332084998624	...	0.256737946999
21	-0.252478752177	-0.254086771438	0.254086771438	...	0.385335283237
22	-0.250911881966	-0.250553084370	0.261108996538	...	1.25

4.3.3 Perhitungan Manual Sequential Training *Support Vector Machine*

Pada proses ini terdapat perulangan pada nilai *Ei*, $\delta\alpha i$ dan αi. Perulangan dilakukan sampai mencapai iterasi maksimal yang ditentukan pada inputan di awal

sistem. Berikut adalah tahapan pada perhitungan *Sequential Training Support Vector Machine.*

1. Perhitungan manualisaasi penelitian ini menggunakan beberapa inputan parameter seperti nilai Complexity (C) =1, nilai γ = 0,008 dan iterasi maksimal = 2. Nilai γ = 0,008 didapatkan dari perhitungan konstanta yang dibagi nilai max diagonal pada matriks *hessian.*

$$\gamma = \frac{0.01}{max.D_{ij}}$$

Iterasi 1 :

a. Perhitungan nilai Ei dengan inialisasi awal parameter $\alpha i = 0$, berikut adalah perhitungan Ei pada data pertama sebagai berikut :

$$E_i = \sum_{j=1}^{1} \alpha_i D_{ij}$$

=0*1,25) + (0*0,332084998624) + (0*-0,254086771438) + (0*-0,252478752) +(0*-0,2611089965)+(0*-0,256737946)+(0*-0,2611089965)+(0*-0,2801973834)+(0*-0,254086771438)+(0*-0,2611089965) + (0*-0,256737946) +(0*-0,322084099)+(0*-0,2611089965)+(0*-0,256737946)+(0*-0,322084099) + (0*-0,280195583) + (0*-0,315335280) + (0*-0,315335280)) = 0

Tabel 4.6 berikut menunjukan hasil dari perhitungan nilai Ei pada Level 1.

Tabel 4. 6 Hasil Perhitungan Nilai Ei Level 1 Pada Iterasi 1

Dij	1	2	3	...	22	Ei
1	0	0	0	...	0	0
2	0	0	0	...	0	0
3	0	0	0	...	0	0
4	0	0	0	...	0	0
5	0	0	0	...	0	0
6	0	0	0	...	0	0
7	0	0	0	...	0	0
8	0	0	0	...	0	0
9	0	0	0	...	0	0
10	0	0	0	...	0	0
11	0	0	0	...	0	0
12	0	0	0	...	0	0
13	0	0	0	...	0	0
14	0	0	0	...	0	0
15	0	0	0	..	0	0
16	0	0	0	...	0	0
17	0	0	0	...	0	0
18	0	0	0	...	0	0

19	0	0	0	...	0	0
20	0	0	0	...	0	0
21	0	0	0	...	0	0
22	0	0	0	...	0	0

b. Proses selanjutnya perhitungan $\delta\alpha i$ pada data ke-1

$$\delta a_i = \min\{max[\gamma(1 - E_i), -a_i], C - a_i\}$$

$$\delta a_i = \min\{max[0{,}008(1 - 0), -0], 1 - 0\}$$

= O,OO8

Tabel 4.7 berikut menunjukan hasil perhitungan nilai δαi pada level 1

Tabel 4.7 Hasil Perhitungan Nilai δαi Level 1 Pada Iterasi 1

	1	2	3	...	22
$\delta\alpha_1$	0.008	0.008	0.008	...	0.008

c. Proses selanjutnya perhitungan α_i baru pada data ke-1

$$\alpha i = \alpha i + \delta\alpha i$$

= 0 + O,008 = O,008

Tabel 4.8 berikut menunjukkan hasil perhitungan α_i baru untuk level

Tabel 4.8 Hasil Perhitungan Nilai αi Level 1 Pada Iterasi 1

	1	2	3	...	22
α_1	0.008	0.008	0.008	...	0.008

d. Proses selanjutnya adalah tahapan iterasi ke 2. Nilai αi yang digunakan pada iterasi kedua adalah nilai αi terbaru yang ditunjukan pada Tabel 4.8. Proses perhitungan nilai *Ei* level 1 akan ditunjukan pada Tabel 4.9.

Tabel 4.9 Hasil Perhitungan Nilai Ei Level 1 Pada Iterasi 2

$\alpha_i D_{ij}$	1	2	....	22	Ei
1	0.01	0.002656679989	...	-0.002007295056	-0.028269583442
2	0.002656679989	0.01	...	-0.002004424675	-0.028161245952
3	-0.002032694172	-0.002019830017	...	0.002088871972	0.054615128867
4	-0.002019830017	-0.002012027514	...	0.002053903576	0.053240881478
5	-0.002088871972	-0.002053903576	...	0.002088871972	0.057990881490
6	-0.002053903576	-0.002032694172	...	0.002053903576	0.054395108328
7	-0.002088871972	-0.002146525111	...	0.002004424675	0.047045233734
8	-0.002241579067	-0.002053903576	...	0.002012027514	0.051562961359

9	-0.002032694172	-0.002053903576	...	0.002004424675	0.046336668712
10	-0.002032694172	-0.002053903576	....	0.002004424675	0.046039070040
11	-0.002032694172	-0.002146525111	...	0.002012027514	0.049562977908
12	-0.002053903576	-0.002032694172	...	0.002019830017	0.049661761738
13	-0.002004424675	-0.002007295056	...	0.002241579067	0.047797599629
14	-0.002002683701	-0.002004424675	...	0.002146525111	0.047307226055
15	-0.002019830017	-0.002012027514	...	0.002398296547	0.051656159706
16	-0.002019830017	-0.002012027514	...	0.003082682266	0.052439609765
17	-0.002012027514	-0.002053903576	...	0.002012027514	0.048249899851
18	-0.002019830017	-0.002032694172	...	0.002053903576	0.051844733454
19	-0.002088871972	-0.002019830017	...	0.002032694172	0.057399288653
20	-0.002053903576	-0.002032694172	...	0.002053903576	0.059554254109
21	-0.002019830017	-0.002032694172	...	0.003082682266	0.046171679081
22	-0.002007295056	-0.002004424675	...	0.01	0.047435284530

e. Proses selanjutnya adalah perhitungan nilai $\delta\alpha i$. Pada Tabel 4.10 ditunjukkan hasil perhitungan nilai $\delta\alpha i$ level 1 pada iterasi kedua.

Tabel 4.10 Hasil Perhitungan Nilai $\delta\alpha i$ Level 1 Pada Iterasi 2

	1	2	...	22
$\delta\alpha_2$	0.008226156668	0.008225289968	...	0.007620517724

f. Proses selanjutnya perhitungan nilai αi baru. Pada Tabel 4.11 ditunjukkan hasil perhitungan nilai αi baru level 1 pada iterasi kedua.

Tabel 4.11 Hasil Perhitungan Nilai αi Level 1 Pada Iterasi 2

	1	2	...	22
α_1	0.01622615666753	0.0162252899676147	...	0,015620517723

4.3.4 Perhitungan Nilai w dan b

Tahapan proses selanjutnya yaitu perhitungan posisi bidang normal atau nilai *w* dan *b* sebagai posisi bidang relatif pada koordinat. Pada Tabel 4.12 ditunjukkan hasil perhitungan $K(xi.x^{+})$ untuk kelas positif dan $K(xi.x^{-})$ untuk kelas negatif yang diperoleh dari perhitungan nilai *kernel* dan diambil dari nilai αi terbesar dari semua kelas.

Tabel 4.11 Hasil Perhitungan $K(xi,x+)$ dan $K(xi,x-)$ Level 1

ID	$K(x_i,x_+)$	$K(x_i,x_-)$
1	1	0.011108996538
2	0.082084998624	0.018315638889

3	0.004086771438	0.006737946999
4	0.002478752177	0.004086771438
5	0.011108996538	0.018315638889
6	0.006737946999	0.011108996538
7	0.011108996538	1
8	0.030197383422	0.135335283237
9	0.004086771438	0.002478752177
10	0.004086771438	0.002478752177
11	0.004086771438	0.049787068368
12	0.006737946999	0.030197383422
13	0.000553084370	0.006737946999
14	0.000335462628	0.004086771438
15	0.002478752177	0.004086771438
16	0.002478752177	0.004086771438
17	0.001503439193	0.018315638889
18	0.002478752177	0.030197383422
19	0.011108996538	0.049787068368
20	0.006737946999	0.030197383422
21	0.002478752177	0.001503439193
22	0.000911881966	0.000553084370

Setelah nilai $K(xi.x^{+})$ dan $K(xi.x^{-})$ didapatkan dari setiap kelas yang ada, proses selanjutnya adalah menghitung nilai *w*. Nilai $w.x^{+}$ adalah nilai *support vector* kelas positif dan $w.x^{-}$ nilai *support vector* kelas negatif. Untuk menghitung nilai $w.x^{+}$ dan $w.x^{-}$ digunakan rumus :

$$w.x^{+} = \sum_{i=1}^{n} a_i y_i K(x_i.x^{+})$$

$$w.x^{+} = ((1x1x0{,}016226156) + (0.082084998624x1x0{,}016225289) + (0{,}004087068x(-1)x0{,}016239717) + (0{,}002478639x(-1)0{,}016226156) + (0{,}011187383x(-1)x0{,}016226156) + (0{,}006735639x(-1)x0{,}016226156) + (0{,}011084999x(-1)x0{,}016226156) + (0{,}0301987068x(-1)x0{,}016226156) + (0{,}004086441x(-1)x0{,}016226156) + (0{,}004086016x(-1)x0{,}016226156) + (0{,}082084999x(-1)x0{,}016226156) + (0{,}002084899x(-1)x0{,}016226156) + (0{,}22413016x(-1)x0{,}016226156) + (0{,}0065335283x(-1)x0{,}016226156) + (0{,}002484999x(-1)x0{,}016226156) + (0{,}00097383x(-1)x0{,}016226156))$$

$$w.x^{-} = \sum_{i=1}^{n} a_i y_i K(x_i.x^{-})$$

$$w.x^{-} = ((0{,}011084999x1x0{,}016226156) + (0{,}0183383x1x0{,}016226156) + (0{,}0067383x(-1)x0{,}016226156) + (0{,}011108997x(-1)x0{,}016226156) + (0{,}018315639x(-1)x0{,}016226156) + (0{,}004068997x(-1)x0{,}016226156) + (1x(-1)x0{,}016226156) + (0{,}0135384999x(-1)x0{,}016226156) + (0{,}02478499$$

9x(−1)x0,016226156)+(0,024787068x(−1)x0,016226156)+(0,048315639x(−1)x0,016226156)+(0,02471639x(−1)x0,016226156)+(0,0249787068x(−1)x0,016226156)+(0,0247084999x(−1)x0,016226156)+(0,024315639x(−1)x0,016226156)+(0,024315639x(−1)x0,016226156)+(0,015208499x(−1)x0,016226156)+(0,000584999x(−1)x0,016226156))

Tabel 4.13 menunjukan hasil perhitungan nilai $w.x+$ dan $w.x-$ level 1.

Tabel 4.13 Hasil Perhitungan Nilai $w.x+$ dan Nilai $w.x-$ Level 1

ID	w.x^+	$w.x^-$
1	0.016226156668	0.000180256318
2	0.001331852905	0.000297176552
3	-0.000063602747	-0.000104863201
4	-0.000038604267	-0.000063647677
5	-0.000172590181	-0.000284553102
6	-0.000104875061	-0.000172909744
7	-0.000173562942	-0.015623638130
8	-0.000470701603	-0.002109538228
9	-0.000063873404	-0.000038741178
10	-0.000063883134	-0.000038747079
11	-0.000063767923	-0.000776852331
12	-0.000105130205	-0.000471160893
13	-0.000008637861	-0.000105230690
14	-0.000005240444	-0.000063841672
15	-0.000038635692	-0.000063699488
16	-0.000038620156	-0.000063673873
17	-0.000023474701	-0.000285980400
18	-0.000038631953	-0.000470633532
19	-0.000172642757	-0.000773731155
20	-0.000104713125	-0.000469291668
21	-0.000038521806	-0.000023364656
22	-0.000014171380	-0.000008595377
Σ	0.015754128231	-0.021535261206

Tahapan akhir manualisasi pada penelitian ini yaitu mnenghitung nilai *b* level 1. Untuk mendapatkan nilai *b* digunakan rumus perhitungan :

$$b = -\frac{1}{2}\sum(w.x^+ + w.x^-)$$

b = -0,5 (0,015754128231 + (-0,021535261206)) = 0.002890566487

Setelah perhitungan pada level 1 selesai maka dilanjutkan dengan menghitung proses level 2 dan seterusnya hingga 10 level.

4.3.5 Perhitungan Manual Testing SVM

Proses *testing* pada manualisasi dilakukan untuk menghasilkan nilai f(x) dan menentukan hasil klasifikasi. Pada Tabel 4.14 dijelaskan mengenai data uji yang dipakai pada manualisasi yaitu dengan 11 kelas data uji dari setiap penyakit.

Tabel 4.12 Data Uji

No.	Data ke-	Gejala-Gejala															Kelas
		G1	G2	G3	G4	G5	G6	G7	G8	G9	…	G28	G29	G30	G31	G32	
1	13	1	0	1	1	0	0	1	1	1	….	0	0	0	0	0	Cacingan
2	20	0	0	0	0	0	0	0	0	0	…	0	0	0	0	1	Endometritis
3	30	0	0	0	0	0	0	0	0	0	…	0	0	1	0	0	Kelumpuhan
4	40	1	0	0	0	0	1	1	1	0	…	0	1	0	0	0	Kembung
5	47	0	0	0	0	0	0	0	1	1	…	0	1	0	0	0	Keracunan
6	60	1	1	0	0	0	1	1	0	0	…	0	0	0	0	0	Mastitis
7	70	1	1	0	0	1	0	0	0	0	…	0	0	0	0	0	Myasis
8	90	0	1	0	0	1	0	0	0	0	…	0	0	0	0	0	Orf
9	110	1	0	0	0	0	0	1	0	0	…	0	0	0	0	0	Pink Eye
10	120	0	0	0	0	0	1	0	0	0	…	0	1	0	0	0	Pneumonia
11	140	0	0	0	1	1	0	0	0	0	…	1	0	0	0	0	Scabies

Pada proses *testing*, digunakan data ke 13 untuk perhitungannya diawali dengan perhitungan nilai *kernel* RBF data ke 13 dan semua data latih. Dilanjutkan dengan menghitung nilai $\alpha_i y_i K(xi.x)$. Tabel 4.15 menunjukkan hasil dari perhitungan nilai $\alpha iyiK(xi.x)$ dengan data uji ke 13.

Tabel 4.13 Hasil Perhitungan Kernel dan $\alpha iyiK(xi.x)$ Data Uji 1

ID	$K(x_i, x)$	$\alpha_i y_i K(x_i, x)$
1	0,135335283237	0,002200960964
2	0,367879441171	0,005974348130
3	0,049787068368	-0,000778655681
4	0,018315638889	-0,000286800228
5	0,030197383422	-0,000471706096
6	0,049787068368	-0,000778888812
7	0,030197383422	-0,000473414361
8	0,049787068368	-0,000777234414
9	0,135335283237	-0,002111863619
10	0,030197383422	-0,000472487058
11	0,223130160148	-0,003484728287
12	0,223130160148	-0,003485949944
13	0,223130160148	-0,003487599672
14	0,135335283237	-0,002118474836
15	0,030197383422	-0,000472286499

16	0,082084998624	-0,001285550840
17	0,135335283237	-0,002095991619
18	0,135335283237	-0,002095991619
		-0,016502314490

Proses akhir yaitu menghitung nilai variabel *f(x)* untuk mendapatkan hasil dari fungsi klasifikasi.

$$f(x) = \sum_{i=1}^{m} a_i y_i x_i . x + b$$

= 0,003838146657 + 0,002890566487

= 0,006728713145

Fungsi klasifikasi = sign(0,006728713145) = 1

Nilai b pada perhitungan nilai f(x) didapat nilai b pada level sebelumnya. Hasil dari fungsi klasifikasi adalah 1 maka hasil klasifikasi berhenti di level 1 yang masuk kedalam kelas penyakit kambing Cacingan.

4.3.6 Perhitungan Manual Nilai Akurasi

Untuk mengukur tingat keberhasilan sistem yang telah dibangun maka dilakukan dengan membandingkan hasil penentuan dari nilai pakar dengan hasil perhitungan manualisasi. Untuk menghitung nilai akurasi digunakan rumus akurasi sebagai berikut :

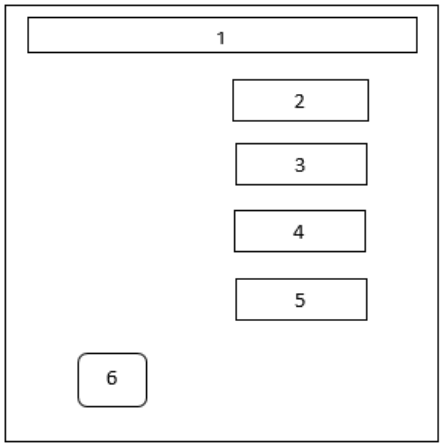
$$Akurasi = \frac{jumlah\ data\ testing\ manual}{jumlah\ seluruh\ data\ testing} x\ 100\%\ =\ldots\%$$

4.4 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk mempermudah interaksi pengguna dengan sistem klasifikasi penyakit kambing menggunakan metode *Support Vector Machine*. Perancangan antarmuka pada penelitian ini terdiri dari perancangan halaman awal sistem, halaman SVM level 1, halaman SVM level 2 dan halaman hasil sistem.

4.4.1 Perancangan Halaman Awal Sistem

Pada perancangan untuk halaman awal sistem ini dibangun untuk menginputkan nilai parameter yang digunakan. Tampilan halaman awal pada sistem digambarkan pada Gambar 4.6.



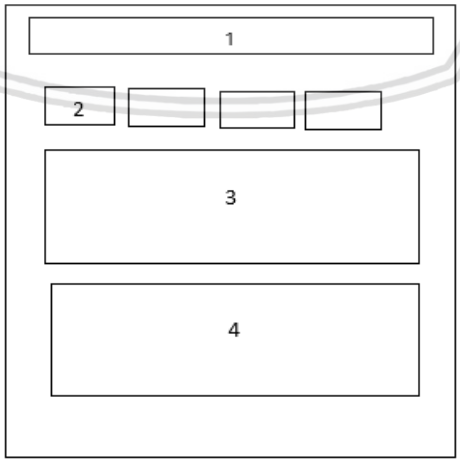
Gambar 4.10 Perancangan Halaman Pilihan Data

Berikut adalah keterangan pada Gambar 4.6 :

1. Judul Sistem
2. Inputan parameter K-Fold
3. Inputan Parameter Lambda
4. Inputan parameter Complexity
5. Inputan parameter Iterasi
6. Button Hitung

4.4.2 Perancangan Halaman SVM Level 1

Perancangan halaman SVM level 1 menampilkan hasil perhitungan klasifikasi *Support Vector Machine* pada level 1. Gambar 4.7 menampilkan model rancangan halaman SVM level 1.



Gambar 4.11 Perancangan Halaman SVM Level 1

Keterangan :

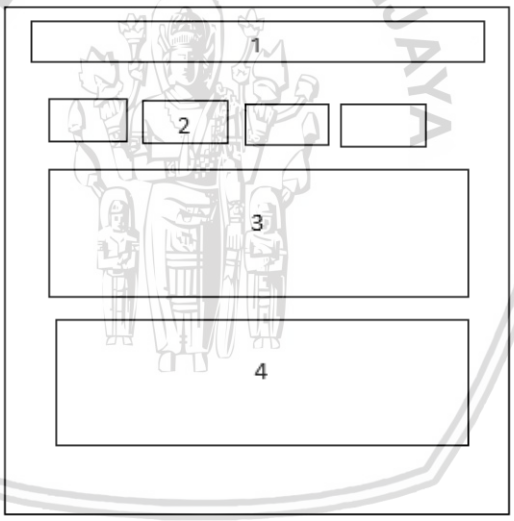
1. Judul Sistem
2. *Tab* Level SVM
3. *Table* perhitungan SVM
4. Hasil nilai bobot tiap level

4.4.3 Perancangan Halaman SVM Level 2

Perancangan halaman SVM level 2 menampilkan hasil perhitungan klasifikasi *Support Vector Machine* pada level 2. Gambar 4.8 menampilkan model rancangan halaman SVM level 2.

Berikut keterangan Gambar 4.8:

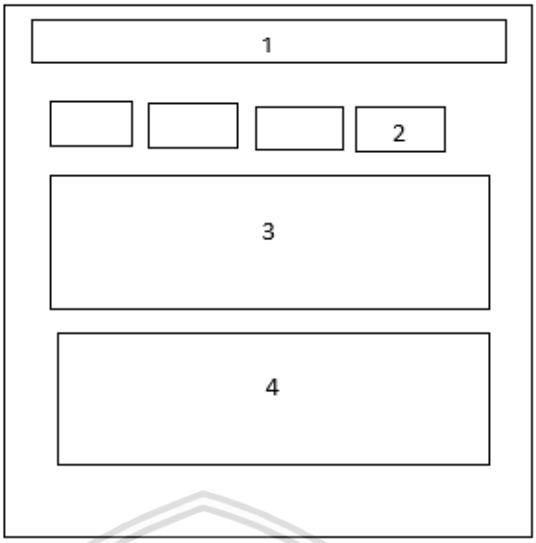
1. Judul Sistem
2. Tab Level SVM
3. Table perhitungn SVM
4. Hasil nilai bobot tiap level



Gambar 4.12 Perancangan Halaman SVM Level 2

4.4.4 Perancangan Halaman Hasil Sistem

Perancangan halaman hasil sistem menampilkan hasil akhir perhitungan data uji klasifikasi *Support Vector Machine*. Gambar 4.9 menampilkan model rancangan halaman hasil akhir pengujian.



Gambar 4.13 Perancangan Halaman Hasil Sistem

Keterangan:

1. Judul Sistem
2. Tab Hasil akhir
3. Hasil perhitungan data uji
4. Nilai Akurasi

4.5 Perancangan Pengujian Sistem

Perancangan pengujian sistem digunakan untuk menguji sistem yang telah dibangun. Sistem dapat dikatakan baik jika memiliki nilai error yang kecil. Berikut adalah beberapa proses pengujian yang ada pada penelitian ini:

1. Pengujian parameter λ (*lambda*)
2. Pengujian parameter C (*complexity*)
3. Pengujian parameter σ (*sigma*)
4. Pengujian jumlah iterasi

4.5.1 Pengujian Pada Nilai Lambda

Pengujian parameter Lambda berjutuan untuk mengetahui nilai parameter Lambda terbaik sehingga didapatkan nilai akurasi yang terbaik. Pada pengujian Lambda ini digunakan nilai parameter lambda 0.1, 0.5, 1, 10, 50, 100, 500, 1000. Tabel 4.16 menunjukkan nilai hasil pengujian parameter lambda yang akan dilakukan.

Tabel 4.14 Pengujian Nilai Lambda

Nilai Lambda	Rata – Rata Nilai Akurasi (%)
0,1	
0,5	
1	
10	
50	
100	
500	
1000	

4.5.2 Pengujian Pada Nilai C (Complexity)

Pengujian parameter *complexity* bertujuan untuk mengetahui nilai parameter C terbaik dan pengaruhnya pada nilai akurasi sistem. Pada pengujian Complexity ini digunakan nilai parameter 0.1, 0.5, 1, 10, 20, 30, 40 dan 50. Tabel 4.17 menunjukkan nilai hasil pengujian parameter lambda yang akan dilakukan.

Tabel 4.17 Pengujian Nilai Complexity

Nilai Complexity	Rata – Rata Nilai Akurasi (%)
0,1	
0,5	
1	
10	
20	
30	
40	
50	

4.5.4 Pengujian Pada Nilai σ Sigma

Pengujian parameter *Sigma* bertujuan untuk mengetahui nilai parameter terbaik dan pengaruhnya pada nilai akurasi sistem. Pada pengujian Sigma ini digunakan nilai parameter 0.01, 0.1, 0.5, 1, 1.5, 2, 3 dan 5. Tabel 4.18 menunjukkan nilai hasil pengujian parameter Sigma yang akan dilakukan.

Tabel 4.15 Pengujian Nilai Sigma

Nilai Sigma	Rata – Rata Nilai Akurasi (%)
0,01	
0,1	
0,5	
1	
1,5	
2	
3	
5	

4.5.5 Pengujian Pada Jumlah Iterasi

Pengujian Jumlah Iterai bertujuan untuk mengetahui nilai parameter iterasi terbaik dan pengaruhnya pada nilai akurasi sistem. Pada pengujian Iterasi ini digunakan nilai parameter 1, 10, 50, 100, 500,1000, 2000 dan 5000. Tabel 4.19 menunjukkan nilai hasil pengujian Jumlah Iterasi yang akan dilakukan.

Tabel 4.19 Pengujian Jumlah Iterasi

Jumlah Iterasi	Rata – Rata Nilai Akurasi (%)
1	
10	
50	
100	
500	
1000	
2000	
5000	

BAB 5 IMPLEMENTASI

5.1 Spesifikasi Sistem

Terdapat dua spesifikasi sistem yang diperlukan dalam peneliatian ini yaitu spesifikasi untuk perangkat keras dan spesifikasi untuk perangkat lunak. Spesifikasi perangkat keras yaitu laptop yang digunakan untuk membuat program dan spesifikasi perangkat lunak yaitu sistem operasi yang dipakai.

5.1.1 Spesifikasi Perangkat Keras

Dalam membuat sistem klasifikasi penyakit kambing digunakan perangkat laptop dengan spesifikasi yang akan ditampilkan Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keraas

Nama Komponen	Spesifikasi
Prosesor	Intel® Core™ i7-3612QM CPU @ 2.10GHz 2.1 GHz
Memori	8,00 GB
Kartu Grafis	NVIDIA GeForce GT 640M
Hardisk	1 TB

5.1.2 Spesifikasi Perangkat Lunak

Dalam pembuatan sistem klasifikasi penyakit kambing ini didukung dengan sistem operasi dan Bahasa pemograman yang akan ditampilkan Tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Sistem Operasi	Microsoft Windows 10 64-bit
Bahasa Pemrograman	Java

5.2 Batasan Implementasi Sistem

Batasan Implementasi pada sistem klasifikasi penyakit kambing adalah sebagai berikut :

1. Bahasa pemrograman yang dibangun untuk membuat klasifikasi penyakit kambing ini adalah pemograman *Java.*
2. Data penyakit kambing yang digunakan diambil dari file pada Microsoft Excel.
3. Metode yang digunakan pada klasifikasi penyakit kambing yaitu *Support Vector Machine.*
4. Inputan pada sistem merupakan data untuk penyakit kambing.
5. Output sistem yaitu hasil klsafikasi penyakit kambing berdasarkan data penyakit kambing.

5.3 Implementasi Algoritme

Implementasi algoritme klasifikasi penyakit kambing dengan metode *Support Vector Machine* ini menggunakan *file Microsoft Excel* untuk mengambil nilai data latih dan data uji.

5.3.1 Implementasi Algoritme *Kernel* SVM

Langkah pertama dalam melakukan perhitungan dengan metode SVM adalah menghitung nilai kernel. Kernel yang dipakai adalah kernel RBF dan hasil dari perhitungan didapatkan nilai matriks. Kode program perhitungan kernel RBF ditujukan Kode Program 5.1.

```
1.  public double[][] hitungKernel(double[][] dataLatih) {
2.      double[][] kernel = new
3.  double[dataLatih.length][dataLatih.length];
4.      double kernelTemp;
5.      double tao = 1;
6.      for (int i = 0; i < dataLatih.length; i++) {
7.        for (int j = 0; j < dataLatih.length; j++) {
8.            kernelTemp = 0;
9.            for (int k = 0; k < dataLatih[j].length - 1; k++) {
10.                kernelTemp += Math.pow((dataLatih[i][k] -
11. dataLatih[j][k]), 2);
12.               }
13.          kernelTemp = -1 * (kernelTemp / (2 * (Math.pow(tao, 2))));
14.          kernelTemp = Math.exp(kernelTemp);
15.          kernel[i][j] = kernelTemp;
16.        }
17.       }
18. return kernel;
19.    }
```

Kode Program 5.1 Algoritme Kernel SVM

Berikut penjelelasan dari Kode Program 5.1

1. Pada baris 1 merupakan deklarasi untuk fungsi method hitungKernel dengan menggunakan data latih.
2. Pada baris 6 merupakan kode untuk perulangan sebanyak jumlah iterasi.
3. Pada baris 7 merupakan kode untuk perulangan data ke-j dan ke-k sepanjang banyaknya data.
4. Pada baris 10 – 15 merupakan kode untuk rumus menghitung nilai kernel RBF.
5. Pada baris 18 merupakan kode untuk mengembalikan nilai hasil kernel.

5.3.2 Implementasi Algoritme *Matriks Hessian*

Perhitungan Matriks Hessian didapatkan setelah perhitungan nilai kernel. Hasil dari perhitungan Matriks Hessian akan digunakan untuk perhitungan selanjutnya yaitu Sequential Training *Support Vector Machine*. Kode program algoritme Matriks Hessian ditujukan Kode Program 5.2.

```
1.    public double[][] hitungMatriksHessian(double lambda, double[][]
2.  dataLatih, double[][] kernel) {
3.          double[][] hessian = new
4.  double[kernel.length][kernel.length];
5.          double levelKelas[][] = new double[dataLatih.length][1];
6.
7.          for (int i = 0; i < dataLatih.length; i++) {
8.              levelKelas[i][0] = (double)
9.  dataLatih[i][dataLatih[i].length - 1];
10.         }
11.
12.         for (int i = 0; i < kernel.length; i++) {
13.             for (int j = 0; j < kernel.length; j++) {
14.                 hessian[i][j] = (levelKelas[i][0] *
15. levelKelas[j][0] * (kernel[i][j] + (Math.pow(lambda, 2))));
16.             }
17.         }
18.         return hessian;
19.     }
```

Kode Program 5.2 Algoritme *Matriks Hessian*

Berikut adalah penjelasan Kode Program 5.2.

1. Pada baris 1 merupakan kode untuk deklarasi fungsi method dan atribut.
2. Pada baris 3 – 4 merupakan deklarasi array hessian untuk menyimpan hasil perhitungan hessian.
3. Pada baris 5 merupakan kode deklarasi array level kelas untuk menyimpan kelas dari data latih.
4. Pada baris 7 – 10 merupaakan kode untuk inisialisasi array kelas.
5. Pada baris 12 - 17 merupakan kode untuk perulangan dan rumus untuk menghitung nilai matriks hessian.
6. Pada baris 18 merupakan kode untuk mengembalikan nilai hasil matriks hessian.

5.3.3 Implementasi Algoritme Sequential Training SVM

Pada perhitungan algoritme *Sequential Training SVM* terdapat beberapa tahap yaitu pertama menghitung nilai Ei setiap data latih yang digunakan, kedua menghitung nilai δαi dan yang terakhir yaitu menghitung nilai αi baru.

5.3.3.1 Implementasi Algoritme Nilai Ei

Menghitung Nilai Ei adalah proses awal dar Sequential Training SVM. Untuk mendapatkan Nilai Ei dilakukan perkalian nilai αi dengan matriks hessian. Kode program Algoritma Nilai Ei ditujukkan pada Kode Program 5.3

```
1.    public double[][] hitungEpsilon(double alpha, double[][]
2.  hessian) {
3.      double[][] epsilon = new double[hessian.length][1];
4.      double[][] e = new double[hessian.length][hessian[0].length];
5.
6.      for (int i = 0; i < hessian.length; i++) {
7.          epsilon[i][0] = 0;
8.          for (int j = 0; j < hessian[i].length; j++) {
9.              e[i][j] = 1 * alpha * hessian[i][j];
10.             epsilon[i][0] += e[i][j];
11.           }
12.       }
13.      return epsilon;
14.    }
```

Kode Program 5.3 Algoritme Nilai *Ei*

Berikut adalah penjelasan untuk Kode Program 5.3

1. Pada baris 1 merupakan kode untuk deklarasi fungsi method dan atribut.
2. Pada baris 3 – 5 merupakan deklarasi array epsilon untuk menyimpan hasil nilai epsilon.
3. Pada baris 7 – 13 merupakan kode untuk rumus menghitung nilai epsilon.
4. Pada baris 14 merupakan kode untuk mengembalikan nilai hasil epsilon.

5.3.3.2 Implementasi Algoritme Nilai δαi

Pada perhitungan algoritme nilai δαi didapatkan dari nilai minimum yang ada pada nilai delta alpha ke-i, nilai *C* dan jumlah nilai Ei. Kode program algoritme nilai δαi ditujukan Kode Program 5.4

```
1.  public double[][] hitungDeltaAlpha(double learningRate, double
2.  alpha, double c, double[][] epsilon, int hesianLength) {
3.       double[][] deltaAlpha = new double[1][hesianLength];
4.
5.       for (int i = 0; i < hesianLength; i++) {
6.           deltaAlpha[0][i] = Math.min(Math.max((learningRate * (1 -
7.  epsilon[i][0])), (-1 * alpha)), (c - alpha));
8.          }
9.
10.      return deltaAlpha;
11.     }
```

Kode Program 5.4 Algoritme Nilai δαi

Berikut adalah penjelasan untuk Kode Program 5.4

1. Pada baris 1 merupakan kode untuk deklarasi fungsi method dan atribut.
2. Pada baris 3 merupakan deklarasi array deltaAlpha untuk menyimpan hasil nilai deltaAlpha.
3. Pada baris 5-7 merupakan kode untuk rumus menghitng nilai deltaAlpha.
4. Pada baris 10 merupakan kode untuk mengembalikan nilai hasil deltaAlpha.

5.3.3.3 Implementasi Algoritme Nilai αi

Perhitungan pada nilai αi baru dihasilkan dari nilai δαi baru dijumlah dengan nilai αi awal. Kode program algoritme nilai αi ditujukan Kode Program 5.5

```
1.  public double[][] updateAlpha(double alpha, double[][] deltaAlpha)
2.  {
3.       double[][] newAlpha = new double[1][deltaAlpha[0].length];
4.
5.       for (int i = 0; i < deltaAlpha[0].length; i++) {
6.            newAlpha[0][i] = alpha + deltaAlpha[0][i];
7.        }
8.      return newAlpha;
9.     }
```

Kode Program 5.5 Algoritme Nilai αi

Berikut adalah penjelasan untuk Kode Program 5.5

1. Pada baris 1 merupakan kode program untuk deklarasi fungsi method dan atribut.
2. Pada baris 3 merupakan kode untuk deklarsi array newAlpha untuk menyimpan nilai hasil newAlpha.
3. Pada baris 5 - 6 merupakan kode rumus untuk menghitung nilai newAlpha.
4. Pada baris 8 merupakan kode untuk mengembalikan nilai hasil newAlpha.

5.3.4 Implementasi Algoritme Nilai b

Perhitungan pada nilai b dihasilkan dari total nilai wx^+ dijumlahkan dengan nilai total wx^- yang didapat menggunakan perhitungan kernel setiap kelas positif dan negative dengan menggunakan nilai alpha baru pada masing – masing kelas. Kode program perhitungan nilai b ditujukan Kode Program 5.6

```
1.      public double hitungBobot(double[][] dataLatih, double[][]
2.  newAlpha) {
3.      double levelKelas[][] = new double[dataLatih.length][1];
4.      double[][] w = new double[hessian.length][2];
5.      double w1Total = 0;
6.      double w2Total = 0;
7.
8.      for (int i = 0; i < dataLatih.length; i++) {
9.          levelKelas[i][0] = (double)
10. dataLatih[i][dataLatih[i].length - 1];
11.         }
12.
13.     for (int j = 0; j < w[0].length; j++) {
14.         if (j == 0) {
15.            for (int k = 0; k < newAlpha[0].length; k++) {
16.                w[k][j] = newAlpha[0][k] * levelKelas[k][0] *
17. kPositive[k][0];
18.                w1Total += w[k][j];
19.                 }
20.         } else {
21.     for (int k = 0; k < newAlpha[0].length; k++) {
22.       w[k][j] = newAlpha[0][k] * levelKelas[k][0] *
23.
24.
25.
26.
```

```
kNegative[k][0];
                w2Total += w[k][j];
            }
          }
        }

  double bias = -0.5 * (w1Total + w2Total);
  return bias;
    }
```

Kode Program 5.6 Algoritme Perhitungan Nilai b

Berikut adalah penjelasan untuk Kode Program 5.6

1. Pada baris 1 merupakan kode program untuk deklarasi fungsi method dan atribut.
2. Pada baris 3 – 4 merupakan deklarasi array w untuk menyimpan nilai w.
3. Pada baris 13 – 18 merupakan kode program untuk rumus menghitung nilai wx^+.
4. Pada baris 21 – 27 kode program untuk rumus menghitung nilai wx^-.
5. Pada baris 32 merupakan kode program untuk menghitung nilai bias.
6. Pada baris 33 merupakan kode untuk mengembalikan nilai hasil bias.

5.3.5 Implemetasi Algoritme Perhitungan Nilai f(x)

Perhitungan nilai f(x) dijadikan penentu hasil klasifikasi dengan menjumlahkan nilai bias dan perhitungan pada data uji. Terdapt dua hasil pada perhitungan nilai f(x) yang didapatkan yaitu nilai positif atau nilai negative yang menentukan data uji tersebut masuk kedalam kelas positif atau kelas negative. Jika nilai f(x) positif maka masuk kedalam kelas 1 dan jika nilai f(x) negative maka masuk kelas -1. Kode program algoritma perhitungan nilai f(x) ditujukan Kode Program 5.7.

```
public int kernelUji(int idx, double[][] dataUji) {
    double[][] kernelUji = new double[dataLatih.length][1];
    double[][] alphaUji = new double[dataLatih.length][1];
    double kernelTemp;
    double tao = 1;
    double alphaTotal = 0;
    for (int i = 0; i < dataLatih.length; i++) {
         kernelTemp = 0;
         for (int k = 0; k < dataLatih[i].length - 1; k++) {
             kernelTemp += Math.pow((dataLatih[i][k] -
dataUji[idx][k]), 2);
            }
       kernelTemp = -1 * (kernelTemp / (2 * (Math.pow(tao, 2))));
            kernelTemp = Math.exp(kernelTemp);
            kernelUji[i][0] = kernelTemp;
            alphaUji[i][0] = kernelUji[i][0] * newAlpha[0][i] *
dataLatih[i][dataLatih[i].length - 1];
            alphaTotal += alphaUji[i][0];
        }
        double fx = alphaTotal + bias;
        return (fx < 0 ? -1 : 1);
    }
}
```

Kode Program 5.7 Algoritme Perhitungan Nilai f(x)

Berikut adalah penjelasan untuk Kode Program 5.7

1. Pada baris 1 merupakan kode program untuk deklarasi fungsi method dan atribut.
2. Pada baris 2 merupakan kode untuk deklarasi array kernelUji.
3. Pada baris 3 merupakan kode untuk deklarasi array alphaUji.
4. Pada baris 7-11 merupakan kode untuk menghitung nilai $K(x_i, x)$.
5. Pada baris 13-18 merupakan kode untuk menghitung nilai $\alpha_i y_i K(x_i, x)$.
6. Pada baris 20 merupakan kode untuk mendapatkan nilai f(x).
7. Pada baris 21 merupakan kode untuk mengembalikan nilai hasil f(x).

5.4 Implementasi Antarmuka Sistem

Antarmuka sistem yang dibangun merupakan penghubung antara pengguna dengan aplikasi yang dibangun secara langsung. Ada beberapa bagian utama dari antarmuka sistem klasifikasi penyakit kambing diantaranya halaman awal sistem, halaman SVM Level 1 sampai 10 dan halaman hasil klasifikasi sistem.

5.4.1 Implementasi Antarmuka Halaman Awal Sistem

Antarmuka halaman awal sistem digunakan untuk menampilkan dan menginputkan parameter yang akan digunakan untuk proses perhitungan menggunakan algoritma *Support Vector Machine*. Data latih dan data uji pada penelitian ini menggunakan format .xls. Tampilan implementasi halaman awal sistem ditampilkan pada Gambar 5.1.

Pada halaman awal sistem terdapat beberapa inputan parameter sistem diantaranya yaitu nilai k-fold yang akan dipakai, nilai parameter lambda, nilai parameter complexity, nilai parameter iterasi maksimum, dan nilai parameter sisgma. Terdapat dua tombol pada halaman awal sistem yaitu tombol Mulai Hitung dan tombol Default Parameter. Tombol Mulai Hitung pada halaman awal sistem digunakan untuk menjalankan proses hitung. Tombol Default Parameter adalah tombol yang digunakan untuk mengisi nilai parameter default dari sistem.

5.4.2 Implementasi Antarmuka Halaman SVM Level 1

Pada tampilan halaman SVM Level 1 ini akan menunjukkan semua perhitungan training menggunakan *Support Vector Machine* pada Level 1. Perhitungan ini terdiri dari perhitungan kernel RBF, matriks hessian dan Sequential training SVM. Tampilan halaman pilih data ditampilkan pada Gambar 5.2

Gambar 5. 1 Halaman Awal Sistem

Gambar 5.2 Halaman *Support Vector Machine* Level 1

5.4.3 Implementasi Antarmuka Halaman SVM Level 2

Pada tampilan halaman SVM Level 2 ini akan menunjukkan semua perhitungan training menggunakan *Support Vector Machine* pada Level 2. Perhitungan ini terdiri dari perhitungan kernel RBF, matriks hessian dan Sequential training SVM. Tampilan halaman pilih data ditampilkan pada Gambar 5.3

Gambar 5.3 Halaman *Support Vector Machine* Level 2

5.4.4 Implementasi Antarmuka Halaman SVM Level 3

Pada tampilan halaman SVM Level 3 ini akan menunjukkan semua perhitungan training menggunakan *Support Vector Machine* pada Level 3. Perhitungan ini terdiri dari perhitungan kernel RBF, matriks hessian dan Sequential training SVM. Tampilan halaman pilih data ditampilkan pada Gambar 5.4. Halaman tampilan SVM Level 4 sampai 10 sama dengan tampilan halaman SVM Level 2 dan 3 degan hasil perhitungan yang berbeda di setiap levelnya karena jumlah data yang berbeda di setiap level.

Gambar 5.4 Halaman *Support Vector Machine* Level 3

5.4 5 Implementasi Antarmuka Halaman Hasil Klasifikasi Sistem

Tampilan halaman hasil klasifikasi sistem adalah halaman yang menampilkan hasil pengklasifikasian sistem dengan menggunakan algoritme *Support Vector Machine* yang menempilkan hasil data sesuai dengan kelas penyakit dan hasil akurasi sistem. Tampilan halaman pilih data ditampilkan pada Gambar 5.5

Gambar 5.5 Halaman Hasil Klasifikasi Sistem

BAB 6 PENGUJIAN

6.1 Pengujian Variabel K-Fold

Pengujian parameter k-fold akan memberikan niai parameter k-fold terbaik dengan membagi data sesuai nilai *k-fold* yang ditentukan. Pada penelitian ini penulis menggunakan nilai *k-fold cross validation* terbaik yaitu *10-fold cross validation*. Dengan menggunakan nilai k-fold 10 maka satu dari sepuluh dijadikan sebagai data uji dan yang lain menjadi data latih, proses ini akan terus berkelanjutan hingga seluruh data telah menjadi data latih dan data uji.

6.2 Pengujian Jumlah Iterasi

Pengujian terhadap jumlah iterasi akan menjelaskan hasil dari julah iterasi yang digunakan dan analisis terhadap skenario jumlah iterasi.

6.2.1 Skenario Pengujian Pada Jumlah Iterasi

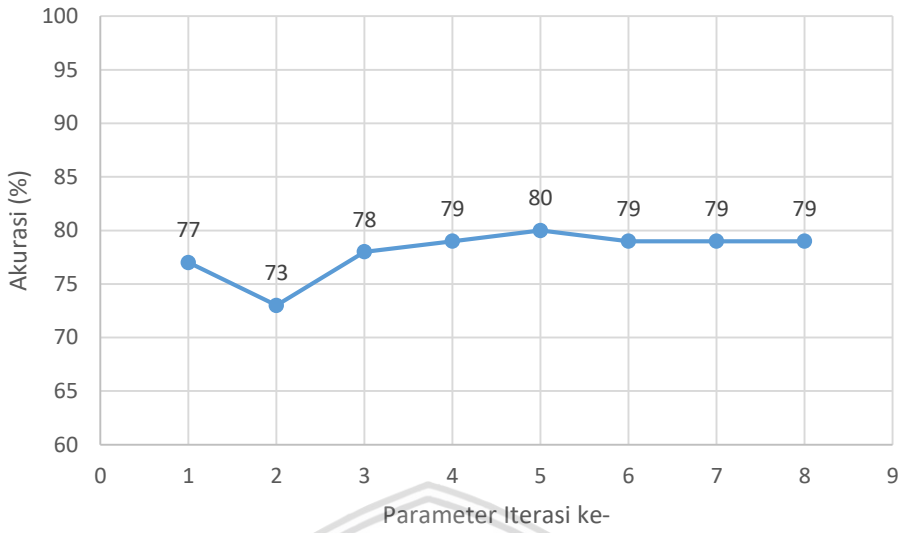
Pengujian terhadap jumlah iterasi dilakukan untuk melihat skenario hasil akurasi terbaik. Jumlah iterasi yang di uji adalah 1, 10, 50, 100, 500,1000, 2000 dan 5000. Parameter yang digunakan adalah λ = 0,5, C = 1, σ = 1, dengan menggunakan *10-fold cross validation*. Hasil pengujian skenario jumlah iterasi dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Jumlah Iterasi

Jumlah Iterasi	Rata – Rata Nilai Akurasi (%)
1	77
10	73
50	78
100	79
500	80
1000	79
2000	79
5000	79

6.2.2 Analisis Pengujian Jumlah Iterasi

Berdasarkan hasil akurasi yang ada pada Tabel 6.1 didapatkan hasil kesimpulan bahwa nilai rata-rata akurasi tertinggi yaitu 80% dengan nilai jumlah iterasi 500. Iterasi dalam pengujian ini didapatkan suatu kondisi dimana iterasi berhenti sebelum mencapai nilai itermax karena nilai alpha yang sudah konvergen. Penurunan nilai akurasi pada jumlah iterasi yang lebih besar dikarenakan ketidakseimbangan nilai support vector dan terdapat data yang jauh terhadap *hyperplane* (Hasanah, et.al, 2016).



Gambar 6.1 Grafik Hasil Pengujian Iterasi

6.3 Pengujian Parameter λ (Lambda)

Pengujian terhadap nilai parameter lambda akan menjelaskan hasil dari nilai lambda yang digunakan dan analisis terhadap skenario pengujian nilai lambda.

6.3.1 Skenario Pengujian Parameter λ (Lambda)

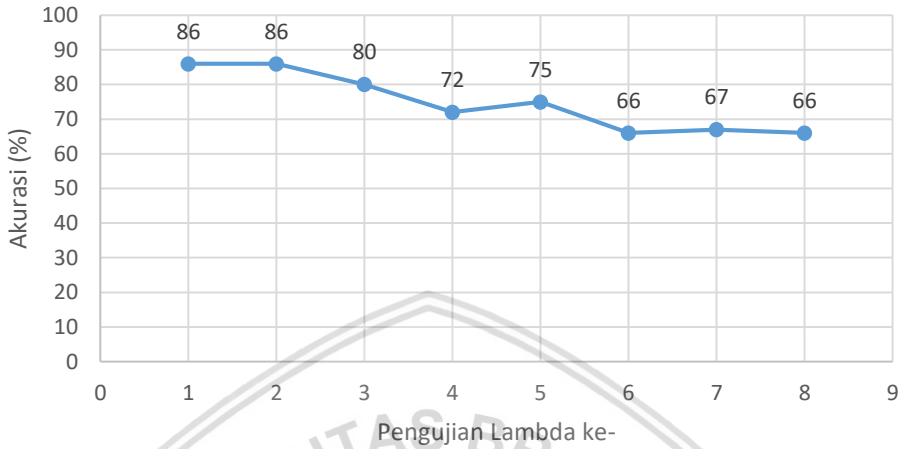
Pengujian terhadap parameter lambda dilakaukan untuk melihat nilai akurasi terbaik berdasarkan nilai lambda yang diuji pada sistem. Nilai lambda yang diuji yaitu 0.1, 0.5, 1, 10, 50, 100, 500, 1000 dan parameter yang digunakan yaitu iterasi = 500, C= 1, σ= 1 dengan nilai k-fold 10. Hasil skenario pengujian parameter lambda dapat dilihat pada Tabel 6.2

Tabel 6.2 Hasil Pengujian Jumlah Lambda

Nilai Lambda	Rata – Rata Nilai Akurasi (%)
0,1	86
0,5	86
1	80
10	72
50	75
100	66
500	67
1000	66

6.3.2 Analisis Pengujian Parameter λ (Lambda)

Berdasarkan Tabel 6.2 dapat disimpulkan bahwa nilai rata – rata akurasi tertinggi adalah 86% pada parameter λ = 0,1. Hasil pengujian pada Tabel 6.4 dijelaskan dalam bentuk grafik pada Gambar 6.2.



Gambar 6.2 Grafik Hasil Pengujian Nilai Lambda

Jika dilihat dari grafik hasil pengujian nilai Lambda dimana parameter yang besar tidak memberikan akurasi lebih baik. Nilai parameter Lambda memiliki pengaruh terhadap jarak *margin* pada *hyperplane* dimana nilai parameter lambda kecil maka jarak *margin* akan mengecil dan nilai *hyperplane* menjadi baik karena perpotongan antar *margin* berpengaruh terhadap garis *hyperplane* (Hasanah,et al., 2016).

6.4 Pengujian Parameter C (Complexity)

Pegujian terhadap nilai parameter complexity menjelaskan hasil dari nilai complexity yang digunakan dan analisis terhadap skenario pengujian nilai complexity.

6.4.1 Skenario Pengujian Pada Parameter C (Complexity)

Pengujian complexity dilakukan untuk melihat nilai complexity dengan akurasi terbaik. Nilai complexity yang digunakan dalam pengujian ini adalah 0.1, 0.5, 1, 10, 20, 30, 40 dan 50. Parameter yang digunakan dalam pengujian ini yaitu λ = 0.1, σ = 1, iterasi = 500 dan k-fold 10 Hasil pengujian terhadap nilai complexity dapat dilihat pada Tabel 6.3.

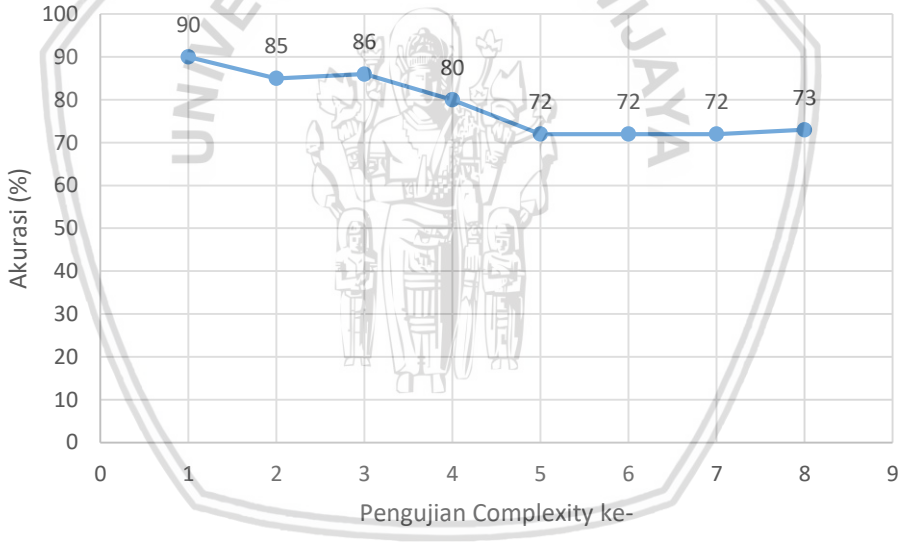
Tabel 6.3 Hasil Pengujian Nilai Complexity

Nilai Complexity	Rata – Rata Nilai Akurasi (%)
0,1	90
0,5	85

1	86
10	80
20	72
30	72
40	72
50	73

6.4.2 Analisis Pengujian Parameter C (Complexity)

Berdasarkan hasil data pada Tabel 6.3 dapat disimpulkan bahwa nilai rata-rata tertinggi adalah 90% untuk nilai C =0,1. Pengujian pada nilai complexity bertujuan untuk mengurangi nilai error. Semakin kecil nilai complexity maka kemungkinan nilai error juga kecil. Nilai parameter complexity juga berpengaruh terhadap waktu komputasi, jika nilai complexity besar maka waktu komputasi pada proses perhitungan data training menjadi lama (Hasanah, et.al, 2016). Grafik hasil pengujian parameter complexity ditunjukkan pada Gambar 6.3.



Gambar 6.3 Grafik Hasil Pengujian Nilai Complexity

6.5 Pengujian Parameter σ Sigma

Pengujian terhadap nilai parameter sigma akan menjelaskan hasil dari sigma yang dihunakan dan analisis terhadap skenario pengujian nilai sigma.

6.5.1 Skenario Pengujian Parameter σ Sigma

Pengujian parameter sigma dilakukan untuk melihat skenario nilai sigma mana yang memberikan nilai akurasi terbaik.Nilai sigma yang digunakan dalam pengujian ini adalah 0.01, 0.1, 0.5, 1, 1.5, 2, 3 dan 5. Parameter yang digunakan dalam pengujian parameter sigma yaitu λ= 0.1, C = 0,1, iterasi = 500, dan nilai k-

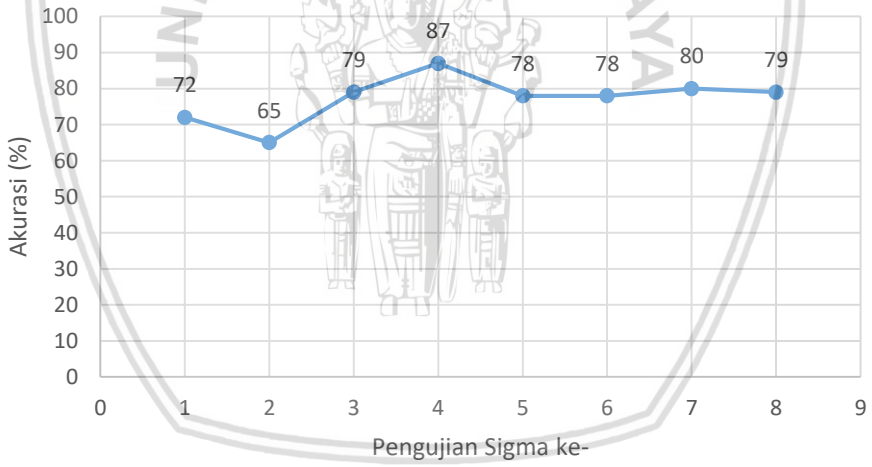
fold yaitu 10. Untuk hasil pengujian terhadap nilai parameter sigma dapat dilihat pada Tabel 6.4.

Tabel 6.4 Hasil Pengujian Nilai Sigma

Nilai Sigma	Rata – Rata Nilai Akurasi (%)
0,01	72
0,1	65
0,5	79
1	87
1,5	78
2	78
3	80
5	79

6.5.2 Analisis Pengujian Parameter σ Sigma

Grafik hasil pengujian parameter nilai sigma dapat dilihat pada Gambar 6.4.



Gambar 6.4 Grafik Hasil Pengujian Nilai Sigma

Berdasarkan hasil data pada Tabel 6.4 dapat disimpulkan bahwa nilai rata – rata akurasi tertinggi adalah 87 pada nilai Sigma = 1. Pada kernel RBF parameter nilai Sigma digunakan untuk menemukan nilai optimal dalam setiap dataset. Apabila nilai optimal telah ditemukan, didapatkan hasil akurasi yang besar dan ketika nilai sigma selanjutnya bertambah nilai akurasi yang di dapatkan menurun dan tidak stabil(Diani,et.al, 2017).

BAB 7 PENUTUP

Pada bab penutup ini akan membahas tentang kesimpulan dan saran dari sistem klasifikasi penyakit kambing dengan menggunakan algoritme *Support Vector Machine (SVM)* yang telah dibangun. Penulisan kesimpulan dan saran diharapkan membantu untuk pengembangan penelitian selanjutnya.

7.1 Kesimpulan

Kesimpulan yang didapat dari penelitian pengklasifikasian penakit kambing dengan menggunakan metode *Support Vector Machine* (SVM) adalah :

1. Algoritme *Support Vector Machine* (SVM) dapat diimplementasikan dalam pengklasifikasian penyakit kambing dengan menggunakan dataset sebanyak 148 data dengan 11 kelas penyakit dan 32 gejala penyakit kambing.
2. Langkah awal untuk pengklasifikasian penyakit kambing adalah menentukan data penyakit kambing selanjutnya memilih data latih dan data uji secara random dengan menggunakan perbandingan rasio data. Langkah berikutnya perhitungan *kernel* RBF dan *sequential training* SVM. Dari hasil perhitungan tersebut didapatkan nilai alpha dan nilai bias yang akan digunakan dalam *testing* SVM. Dari hasil testing didapatkan hasil pengklasifikasian penyakit kambing dan nilai akurasi dari sistem.
3. Tingkat akurasi sistem klasifikasi penyakit kambing dengan metode *Support Vector Machine* (SVM) dengan menggunakan parameter terbaik yaitu k-fold cross validation 10, λ= 0.1, C = 0,1, iterasi = 500 dan σ = 1 didapatkan hasil pengujian yaitu 90 %.

7.2 Saran

Pada penelitian klasifikasi penyakit kambing dengan menggunakan metode *Support Vector Machine* ini masih memeiliki beberapa kelemahan. Adapun saran yang penulis berikan sebagai pengembangan penelitian selanjutnya yaitu :

1. Jumlah dataset setiap kelas penyakit lebih seimbang agar pengujian dapat dilakukan dengan lebih baik.
2. Perbandingan kernel ditambahkan untuk mengetahui perbedaan hasil akurasi setiap kernel pada *Support Vector Machine.*
3. Penggunaan strategi lain pada studi kasus *multi-class* SVM selain *One Againts All*.

DAFTAR PUSTAKA

Bahri, S., Adjid, R.M.A, & Wardhana.A,(2012) Manajemen Kesehatan Dalam Usaha Ternak Kambing. Balai Penelitian Veteriner Bogor.

Darsyah, M. Y., (2014). Klasifikasi Tuberkulosis dengan Pendekatan Metode *Support Vector Machine* (SVM). *Statistika,* Volume 2.

Diani, R., Wisesty, U. N. & Aditsania, A., (2017). Analisis Pengaruh Kernel *Support Vector Machine* (SVM) pada Klasifikasi Data Microarray untuk Deteksi Kanker. Ind. Journal On Computing, Volume 2.

Ferdiansyah, W. R. (2016). *Sistem Pakar Diagnosis Penyakit Pada Kambing Menggunakan Metode Naive Bayes - Certainty Factor.* Malang: Universitas Brawijaya.

Haryanto, D.J. (2018) Analisis Sentimen Review Barang Berbahasa Indonesia Dengan Metode *Support Vector Machine* dan Query Expansion. Malang : Universitas Brawijaya

Hasanah, U., Pratama, A. & Cholissodin, I., 2016. Perbandingan Metode SVM, Fuzzy-KNN dan BDT-SVM Untuk Klasifikasi Detak Jantung Hasil Elektrokardiografi. Teknologi Informasi dan Ilmu Komputer, Volume 3.

Mase. J. (2018). Penerapan Algoritme *Support Vector Machine* (SVM) Pada Pengklasifikasian Penyakit Kucing. Malang : Universitas Brawijaya

MedKes. (2016, Desember Selasa). *Penyebab, Gejala, dan Pengobatan Askariasis*. Retrieved from MedKes: www.medkes.com

Mulyono, S., & Sarwono, B. (2014). *Penggemukan Kambing Potong.* Jakarta: Penerbit Swadaya.

Munawarah, R., Soesanto, O. & Faisal, M. R., 2016. Penerapan Metode *Support Vector Machine* Pada Diagnosa Hepatitis. *Ilmu Komputer (KLIK),* Volume 4.

Nugraha, D. & Winiarti, S., 2014. Pengembangan Media Pembelajaran Sistem Pelacakan Pada Mata Kuliah Kecerdasan Buatan Berbasis Multimedia. Sarjana Teknik Informatika, Volume 2.

Octaviani, P. A., Wilandari, Y. & Inspriyanti, D., 2014. Penerapan Metode Klasifikasi *Support Vector Machine* (SVM) Pada Data Akreditasi Sekolah Dasar (SD) di Kabupaten Magelang. *Gaussian,* Volume 3, pp. 811-820.

Orisa, M., Santoso, P. B., & Setyawati, O. (2014). Sistem Pakar Diagnosis Penyakit Kambing Berbasis Web Menggunakan Metode Certainty Factor. *EECCIS*, 151-156.

Permana, R. A., 2016. Seleksi Atribut Pada Metode *Support Vector Machine* Untuk Menentukan Kelulusan Mahasiswa E-Learning. Evolusi, Volume 4.

Prasetyo, E., 2012. Data Mining KOnsep dan Aplikasi Menggunakan MATLAB. Yogyakarta: Andi Offset

Prasetyo, E., 2014. *Data Mining Mengolah Data Menjadi Informasi Menggunakan MATLAB.* Yogyakarta: Andi Offset.

Setiawan, S. A. (2015). *Sistem Pakar Untuk Mendiagnosis Penyakit Pada Kambing Dengan Metode Forward Chaining.* Semarang: Universitas Dian Nuswantoro.

Suparman. (2014). *Beternak Kambing.* Semarang: Azka Press.

Susanto, A., & Sitanggang, M. (2015). *Mengatasi Permasalahan Praktis Beternak Kambing.* Jakarta: PT. Agro Media Pustaka.