KLASIFIKASI DENGAN METODE NEAREST CENTROID CLASSIFIER DENGAN OUTLIER REMOVAL

TESIS

Untuk memenuhi sebagian persyaratan
memperoleh gelar Magister Komputer

Disusun oleh:
Aditya Hari Bawono
NIM: 176150100111042

PROGRAM STUDI MAGISTER ILMU KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2020

DAFTAR ISI

BAB 1 PENDAHULUAN

1.1 Latar belakang

Klasifikasi adalah sebagian pengolahan data yang telah menarik perhatian akademisi dan industri. Klasifikasi adalah upaya untuk memberikan label terhadap suatu data berdasarkan data yang telah dilabeli sebelumnya (Ougiaroglou, 2015). Dalam dunia nyata, klasifikasi digunakan untuk membedakan satu obyek dengan obyek lainnya. Beberapa topik riset klasifikasi yang populer adalah klasifikasi wajah (Shen, Yang, Li, Zhang, & Tao, 2014) (Shen, Yang, Li, Zhang, & Shen, 2016), klasifikasi teks (Lazhar, 2018) (Uchida, Toriumi, & Sakaki, 2017), dan klasifikasi data statistika (Xie, Gao, Xie, Liu, & Grant, 2016) (Paulheim & Meusel, 2015) (Ertuğrul & Tağluk, 2017).

Faktor yang mempengaruhi hasil prediksi adalah pilihan metode klasifikasi (*classifier*), *preprocessing*, dan data yang digunakan untuk proses pelatihan (data latih). Pemilihan metode akan sangat menentukan hasil klasifikasi, karena menurut teori "*no free lunch theorem*" tidak ada metode yang terbaik untuk semua kasus atau *data set* (H. Wolpert & G. Macready, 1997). Dalam ranah penelitian, dianjurkan untuk mencoba beberapa metode dan modifikasinya untuk mengetahui metode yang terbaik dalam *data set* tertentu. *Preprocessing* adalah proses pengolahan data latih sebelum diproses *classifier* (Han, Kamber, & Pei, 2012). Dampak dari *preprocessing* adalah membantu *classifier* dalam memproses data, memilih fitur yang berpengaruh, dan menormalisir nilai dari keseluruhan data (Uysal & Gunal, 2014). Data latih digunakan untuk dasar pembentukan model oleh *classifier* yang akan digunakan untuk penentuan label atau kelas (Ougiaroglou, 2015). Ketepatan, validitas, konsistensi, relevansi, kelengkapan, dan kebenaran data adalah faktor-faktor yang menjadi tolak ukur kualitas data, dan akan menentukan metode *preprocessing* yang akan dilakukan serta *classifier* yang cocok dengan data latih tersebut. Pemilihan tiga faktor tersebut akan menentukan hasil prediksi.

Data pada sebagian kasus di dunia nyata pada permasalahan prediksi tidak selalu ideal (Krishnan, Wang, Wu, Franklin, & Goldberg, 2016). Beberapa *data set* memiliki nilai yang tidak lengkap atau hilang (*missing value*), kesalahan label (*mislabel*), dan ketidaksesuaian nilai (*misvalue*). *Missing value* menyebabkan algoritme tidak dapat melakukan perhitungan yang disebabkan ketiadaan nilai dalam suatu data. Dalam mengatasi *missing value*, terdapat beberapa solusi seperti menghapus data maupun memberikan nilai prediktif pada *missing value* (Sen, Das, & Chatterjee, 2016). *Mislabel* adalah masalah yang serius untuk klasifikasi karena label atau kelas digunakan dalam menentukan pembentukan model serta pengujian (Xiao, Xia, Yang, Huang, & Wang, 2015). Sedangkan

misvalue adalah kesalahan nilai pada data. *Misvalue* dapat menyebabkan kesalahan perhitungan hingga kesalahan algoritme dalam memperlakukan data. Masalah-masalah tersebut dapat disebabkan pada proses generasi data maupun pada proses pengambilan data dapat terjadi kesalahan pengukuran, kesalahan pengkodean, kesalahan manusia, kesalahan alat, dan bias (Nazari, 2018). Gabungan dari masalah-masalah *misvalue* dan *mislabel* tersebut dapat disebut sebagai *outlier*.

Outlier adalah suatu hasil observasi yang berbeda sifat dengan suatu mayoritas hasil observasi yang lain. *Outlier* dapat disebabkkan oleh kesalahan maupun bias manusia, kesalahan mesin dalam pengambilan data, hingga suatu kejadian janggal. *Outlier* mengakibatkan kecurigaan terhadap mekanisme data tersebut dihasilkan pada suatu waktu (Hawkins, 1980). Keberadaan *outlier* menghasilkan pola yang berbeda pada sebaran data dan rentan mengurangi kemampuan *classifier* dalam mengenali pola (Tallón-Ballesteros & Riquelme, 2014). Dalam kasus lain, *outlier* dapat mengakibatkan kesalahan klasifikasi yang hasilnya akan berbahaya jika diterapkan pada dunia nyata seperti dunia medis dan keuangan (Pasillas-Díaz & Ratté, 2016). Diantara penelitian-penelitian perbandingan, disebutkan bahwa keberadaan *outlier* dapat mengurangi akurasi prediksi sebuah metode (Pelletier, Valero, Inglada, & Dedieu, 2017).

Terdapat penelitian-penelitian sebelumnya yang mencoba menyelesaikan permasalahan *outlier* pada klasifikasi. Penelitian terhadap pengaruh *outlier* pada hasil prediksi telah dilakukan (Estaghvirou, Ogutu, & Piepho, 2014). Pada penelitian tersebut evaluasi terhadap akurasi *classifier* pada prediksi gen tumbuhan. Metode yang digunakan adalah *outlier removal* berbasis statistika, yaitu *variance* dan *covariance matrix* yang menghasilkan kesimpulan bahwa *outlier* memiliki pengaruh yang signifikan untuk prediksi gen dan pengaruhnya makin besar ketika ukuran sampel kecil dan variasi genetis kecil. Sedangkan penelitian pengaruh *outlier* pada perkiraan usaha pengembangan perangkat lunak telah dilakukan (Ono, Tsunoda, Monden, & Matsumoto, 2016). Pada penelitian tersebut *outlier* sengaja ditempatkan pada data set dengan persentase tertentu untuk mengetahui besaran pengaruhnya. Metode *outlier removal* yang digunakan adalah *Cook's Distance* dan menghasilkan kesimpulan bahwa terdapat cara lain selain menghapus *outlier* yaitu menambahkan data lain yang relevan. Penelitian pengaruh *outlier* terhadap prediksi kebangkrutan telah dilakukan oleh (Nyitrai & Miklós, 2018). Pada kasus tersebut, *ratio type financial indicator* adalah variable yang paling populer dalam prediksi kebangkrutan, namun variable tersebut sering terjadi penyimpangan distribusi yang disebabkan oleh *outlier*. Algoritme yang digunakan adalah CHAID yang merupakan algoritme berbasis pohon keputusan dan membuktikan bahwa algoritme berbasis pohon

keputusan CHAID memiliki performa yang lebih baik daripada *linear model* maupun *multilayer perceptron* dalam klasifikasi data yang menyimpang.

Penelitian sebelumnya menyatakan bahwa terdapat peningkatan akurasi setelah *outlier* dihilangkan. Penelitian pengaruh *outlier* terhadap metode klasifikasi *Support Vector Machine* (SVM) dan *Random Forest* (RF), dan telah menghasilkan peningkatan akurasi setelah *outlier* dihilangkan (Sharma, Haleem, & Ahmad, 2015). Penelitian terkait ingin membuktikan bahwa *outlier removal* dapat meningkatkan performa *classifier*. Metode *outlier removal* yang digunakan adalah *categories vector* dan dilakukan pada tahap *preprocessing*. Hasil dari penelitian tersebut membuktikan terdapat peningkatan akurasi setelah diujikan pada algoritme SVM dan RF. Penelitian pengaruh penghilangan *outlier* pada data citra satelit telah dilakukan oleh (Pelletier et al., 2017). Pada kasus citra satelit, data referensi kekurangan aktualitas waktu yang menyebabkan *mislabel* pada pengumpulan data. Metode yang digunakan adalah *learning RF model* yang menghilangkan *outlier* secara berangsur-angsur. Hasil yang didapatkan adalah *learning RF model* cocok untuk mendeteksi data yang *mislabel* dan melakukan klasifikasi. Penelitian tentang *outlier* dengan *clustering* telah dilakukan. Metode *k-Means with outlier detection* (KMOR) melakukan pengelompokan data sekaligus mendeteksi *outlier* pada proses yang bersamaan (*one stage*) (Gan & Ng, 2017b). Hasil dari metode KMOR telah melampaui kecepatan dan akurasi metode lain dalam pengelompokan data.

Berdasarkan pada penelitian yang telah dilakukan sebelumnya terdapat beberapa kekurangan pada penyelesaian permasalahan yang berkaitan dengan *outlier*. Penelitian-penelitian sebelumnya menggunakan metode deteksi *outlier* sebagai *preprocessing* sebelum memulai klasifikasi. Tahapan tersebut biasa disebut *multi-stage process*. *Multi-stage process* melakukan beberapa proses secara berulang-ulang dalam perhitungan antar titik data untuk menyesuaikan data sesuai kebutuhan. Proses komputasi yang besar dilakukan pada *multi-stage processing* (Gan & Ng, 2017a). Hal ini disebabkan proses *outlier removal* saja memiliki waktu yang lama (Domingues, Filippone, Michiardi, & Zouaoui, 2018) (Campos et al., 2016). Selain itu, kekurangan lainnya adalah perbedaan akurasi sebelum dan sesudah data dilakukan *preprocessing* belum tentu signifikan (Sharma et al., 2015). Penelitian sebelumnya berfokus pada *outlier removal* sebagai *preprocessing*. Peneliti-peneliti sebelumnya banyak berfokus kasus *outlier*. Akan tetapi kebanyakan peneliti berfokus pada kasus *outlier* untuk permasalahan *clustering* (Gan & Ng, 2017) (Jiang, Liu, Du, & Sui, 2016). Berdasarkan pengamatan terhadap penelitian sebelumnya, penelitian-penelitian dengan *oulier removal* pada kasus *one-stage classifier* belum pernah dilakukan.

Pada kasus klasifikasi terdapat berbagai jenis klasifikasi dalam kasus *machine learning* yaitu klasifikasi berbasis probabilitas, klasifikasi berbasis jarak, dan

klasifikasi berbasis pohon keputusan. Klasifikasi berbasis probabilitas memiliki hasil yang cukup akurat dan efisien dalam komputasi, namun membutuhkan data yang besar untuk mendapatkan akurasi yang baik. Salah satu metode klasifikasi berbasis probabilitas adalah *Naïve Bayes classifier* (Netti, 2015). Pada klasifikasi berbasis jarak, jarak antar titik data akan dihitung, dibandingkan, dan dicari yang terdekat sehingga waktu komputasi menjadi tinggi, namun akurasi yang tinggi akan dihasilkan dengan pemilihan parameter yang tepat. Beberapa metode klasifikasi berbasis jarak adalah *Support Vector Machine* (SVM) (Wang, Huang, & Cheng, 2014) dan *k-Nearest Neighbor* (*k*-NN) (Ekaristio, Soebroto, & Supianto, 2015). Pada klasifikasi berbasis pohon keputusan, dapat memproses data diskrit dan kontinyu sekaligus dan tangguh menghadapi noise. Kekurangan metode klasifikasi berbasis pohon keputusan adalah dalam pembentukan pohon keputusan membutuhkan memori yang besar, menurun akurasinya jika diterapkan pada *data set* kecil, dan jika diterapkan pada *data set* berukuran besar membutuhkan waktu komputasi yang tinggi pada pembentukan pohon keputusan. Beberapa metode klasifikasi berbasis pohon keputusan adalah C4.5, ID3, dan RF.

Pada kasus klasifikasi, indikator klasifikasi yang baik adalah hasil prediksi yang akurat dan efisiensi waktu (Dogan & Tanrikulu, 2013). Metode klasifikasi seperti SVM, *k*-NN, Naïve Bayes, dan RF selain sensitif terhadap *outlier*, metode-metode tersebut rentan memiliki kompleksitas yang tinggi karena melakukan perhitungan untuk seluruh antar titik data dan antar fitur (Mullick, Datta, & Das, 2018). Beberapa metode seperti *k*-NN dan SVM membutuhkan parameter yang cocok untuk *data set* tertentu untuk menghasilkan akurasi yang baik. Penentuan parameter membutuhkan waktu dan percobaan yang dilakukan berkali-kali (Ning, Chen, Zhou, & Wen, 2018).

Salah satu metode klasifikasi yang memanfaatkan *centroid* adalah *Nearest Prototype Classifier* atau *Nearest Centroid Classifier* (NCC) (Bezdek & Kuncheva, 2001). Beberapa penelitian yang menggunakan metode NCC antara lain (Tibshirani, Hastie, Narasimhan, & Chu, 2003) yang menambahkan *feature selection*, (Dabney, 2005) digunakan untuk klasifikasi DNA, (Praveen, Kousalya, & Kumar, 2016) digunakan untuk *vehicle routing problem*, (Liu et al., 2017) menggunakan *centroid*-based *classifier* untuk klasifikasi teks, (Setiawan, Djanali, & Ahmad, 2018) menggunakan *centroid* based *classifier* untuk deteksi intrusi, dan (Tamatjita & Mahastama, 2017) digunakan untuk klasifikasi genre musik. Penelitian-penelitian tersebut membuktikan performa NCC handal dalam klasifikasi. Akan tetapi, pada penelitian-penelitian tersebut masih belum terdapat penelitian yang menggunakan NCC dengan kemampuan *outlier removal*. Untuk menangani kekurangan tersebut dibutuhkan metode untuk melakukan *outlier removal* dan klasifikasi secara sekaligus (*one stage*).

Berdasarkan kajian penelitian yang telah dilakukan, pada penelitian ini diusulkan sebuah metode klasifikasi dengan *outlier removal* secara *one stage*. Ide awal dari penelitian ini adalah menggunakan metode KMOR. Namun, pada penelitian sebelumnya KMOR digunakan untuk kasus *clustering*. Pada penelitian ini KMOR akan diadopsi untuk kasus klasifikasi. Akan tetapi, efektifitas dari pendekatan ini belum diketahui hasilnya. Inti dari KMOR adalah penggunaan *centroid* atau titik pusat cluster, maka dibutuhkan metode klasifikasi yang menggunakan *centroid* yang sama dengan KMOR untuk mempertahankan prinsip *one-stage processing*. Penelitian ini diharapkan dapat menggabungkan antara metode KMOR dengan metode NCC yang selanjutnya disebut *Nearest Centroid Classifier with Outlier Removal* (NCCOR) untuk mendapatkan solusi klasifikasi yang lebih baik dari metode klasifikasi yang sudah ada dengan cara menghilangkan *outlier* dalam data.

Berdasarkan kajian dari penelitian sebelumnya, penelitian ini berfokus pada klasifikasi dengan menghilangkan *outlier*. Dengan demikian, judul dari penelitian ini adalah **Klasifikasi dengan Metode *Nearest Centroid Classifier* dengan *Outlier Removal***.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah diuraikan, maka masalah yang dirumuskan adalah sebagai berikut.

1. Bagaimana tingkat efektifitas pendekatan *outlier removal* pada akurasi metode usulan NCCOR?
2. Bagaimana parameter optimal pada metode usulan NCCOR?
3. Bagaimana perbandingan akurasi metode usulan NCCOR jika dibandingkan dengan metode pembanding (NCC, *k*-NN, NB)?
4. Bagaimana perbandingan waktu komputasi metode usulan NCCOR jika dibandingkan dengan metode pembanding (NCC, *k*-NN, NB)?

1.3 Tujuan

Tujuan yang ingin dicapai pada penelitian ini adalah sebagai berikut.

1. Mengetahui efektifitas pendekatan *outlier removal* pada akurasi metode usulan NCCOR.
2. Mengetahui parameter optimal pada metode usulan NCCOR.
3. Mengetahui perbandingan akurasi metode usulan NCCOR jika dibandingkan dengan metode pembanding.
4. Mengetahui perbandingan waktu komputasi metode usulan NCCOR jika dibandingkan dengan metode pembanding.

1.4 Manfaat

Penelitian ini dapat memberikan manfaat untuk penelitian lain yang membutuhkan metode untuk melakukan klasifikasi yang tahan terhadap keberadaan *outlier* dalam data serta memiliki performa yang baik.

1.5 Batasan masalah

Batasan masalah pada penelitian ini adalah sebagai berikut:

1. Pada *data set*, tidak ada perlakuan khusus terhadap *missing value*.
2. *Data set* yang digunakan hanya yang bertipe *numeric*.

1.6 Sistematika penelitian

Adapun sistematika penelitian dimulai dengan bab pendahuluan hingga metodologi. Rincian sistematika penelitian ini adalah sebagai berikut.

1. **BAB 1 Pendahuluan**

   Dalam bab ini akan dipaparkan latar belakang penelitian mengenai klasifikasi dengan penanganan *outlier* pada *data set*. Penelitian-penelitian terdahulu seperti penelitian pengaruh *outlier* pada klasifikasi, penelitian mengenai metode *outlier removal*, serta penelitian mengenai metode klasifikasi digunakan sebagai landasan pemikiran untuk solusi dan metode usulan. Masalah yang dipaparkan adalah kehandalan klasifikasi dengan tolak ukur akurasi dan waktu komputasi. Masalah diselesaikan dengan batasan masalah tertentu yang berada di dalam fokus penelitian.

2. **BAB 2 Landasan kepustakaan**

   Pada bab ini akan dipaparkan secara detail tentang dasar teori penelitian sebelumnya hingga terbentuk ide tentang metode usulan. Landasan kepustakaan yang dibahas meliputi *outlier*, klasifikasi, metode-metode terkait, serta metode evaluasi.

3. **BAB 3 Metodologi**

   Bab ini menjelaskan tentang tata cara penelitian ini dilakukan. Mulai alur penelitian dengan metode yang diajukan, *data set* yang digunakan dalam penelitian, metode pembanding yang digunakan, hingga skenario pengujian yang meliputi cara dan dokumentasi hasil pengujian.

4. **BAB 4 Perancangan**

   Bab ini menjelaskan tentang perancangan metode yang diajukan. Perancangan metode yang diajukan meliputi perhitungan manual dari pelatihan hingga pengujian. Perhitungan manual metode evaluasi dibahas pada bab ini setelah perhitungan manual metode yang diajukan menunjukkan hasilnya.

5. **BAB 5 Pengujian dan Pembahasan**

   Bab ini berisi pemaparan hasil pengujian dari semua metode dan semua konfigurasi parameter. Hasil pengujian dibahas berdasarkan rumusan masalah. Hasil yang dibahas adalah akurasi dan waktu komputasi.

6. **BAB 6 Kesimpulan dan Saran**

   Bab ini berisi kesimpulan dan saran yang berhubungan dengan penelitian. Kesimpulan berisi jawaban dari rumusan masalah yang telah dipaparkan sebelumnya. Saran adalah masukan yang masih berhubungan dengan penelitian ini namun berada pada ruang lingkup yang lain.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Terdapat beberapa penelitian yang menjadikan dasar penelitian ini dan juga penelitian yang menjadi acuan dalam pengembagan metode yang diusulkan. Beberapa penelitian sebelumnya tersebut terkait dengan outlier, metode acuan untuk outlier removal dengan pendekatan centroid, serta perlunya tahapan pre-processing. Penelitian-penelitian tersebut dijabarkan dalam paragraf berikut:

Penelitian (Pelletier et al., 2017) membuktikan bahwa *outlier* berdampak pada klasifikasi. Penelitian tersebut menggunakan data penginderaan jauh yang memiliki data mislabel. Metode yang digunakan pada penelitian ini adalah LOF dan *RFTree* untuk *preprocessing* deteksi *outlier* dengan *k*-NN sebagai metode klasifikasi. Fokus pada penelitian ini adalah pemilihan metode deteksi *outlier*, serta membuktikan bahwa *outlier* berdampak pada klasifikasi.

Penelitian (Sharma et al., 2015) mengusulkan metode *outlier removal* untuk meningkatkan performa klasifikasi metode *Support Vector Classifier* (SVC) dan *Random Forest*. Metode *outlier removal* yang diajukan membagi data menjadi dua bagian yaitu *generator set* dan *analysis set* yang akan dibandingkan pada setiap iterasi. Jika terdapat penurunan akurasi pada suatu iterasi, maka titik data yang tidak memenuhi kriteria akan dihilangkan pada *generator set*. Proses berhenti ketika tidak ada perbedaan akurasi pada tiap iterasi. Terdapat peningkatan sebesar 3 pada akurasi metode *Random Forest* dan peningkatan 4 akurasi pada metode SVC.

Penelitian (Gan & Ng, 2017c) mengusulkan metode untuk melakukan *clustering* sekaligus *outlier removal* dalam satu waktu (one stage). *Clustering*, terutama metode k-Means adalah salah satu metode yang populer, namun memiliki kelemahan karena sensitif terhadap *outlier* maupun *noisy data*. Kriteria *outlier* yang diajukan pada penelitian tersebut adalah titik data yang berada jauh dengan *cluster* maupun mayoritas titik data lainnya. Metode yang diajukan memiliki satu tambahan *cluster* untuk menampung *outlier*, yang dinamakan KMOR. KMOR melakukan *clustering* dengan *outlier removal* sekaligus pada setiap iterasi hingga konvergen. Hasil yang didapat adalah peningkatan kualitas *clustering* dan percepatan waktu komputasi yang signifikan.

Penelitian (Levner, 2005) menggunakan NCC dengan merujuk pada masalah dimensionalitas dan mengajukan *feature selection* sebagai metode *preprocessing*. Metode yang diajukan adalah *Boosting*. Hasil yang diperoleh adalah terdapat penambahan akurasi sebesar 11.2 namun terjadi peningkatan waktu komputasi dari 1.19 ms menjadi 543.42 ms. Hasil dari penelitian terkait

dapat menjadi pertimbangan untuk pemilihan metode *preprocessing*, karena berdampak besar pada waktu komputasi.

Perbandingan metode klasifikasi NCC dan *k*-NN dilakukan pada (Tamatjita & Mahastama, 2017). Studi kasus atau data set yang digunakan adalah data *wavelength* untuk klasifikasi genre musik. Penelitian dilakukan dengan modifikasi data set dengan jumlah kelas yang berbeda yaitu, tiga, enam, sembilan, dan dua belas kelas. Hasil penelitian tersebut menemukan bahwa semakin sedikit jumlah kelas pada NCC maka semakin baik tingkat akurasinya. Pada data set dengan tiga kelas, NCC dapat mencapai 96.67 akurasi, namun menurun pada 6 kelas sebesar 70, Sembilan kelas sebesar 53.33, dan paling rendah pada dua belas kelas yaitu sebesar 33.33. Hal ini disebabkan pada data *wavelength*, *genre* musik bisa berdekatan antar satu dengan lainnya sehingga makin banyak kelas, makin mungkin terjadi irisan antar *centroid*. Untuk metode *k*-NN sebagai pembanding, performa yang dihasilkan tidak lebih baik daripada NCC, yaitu sekitar 22.5 hingga 26.7 saja. NCC dapat menjadi pertimbangan metode klasifikasi merujuk pada penelitian terkait.

Berdasarkan kajian pustaka yang telah dilakukan, maka dapat disimpulkan bahwa *outlier* berpengaruh terhadap akurasi metode klasifikasi (Pelletier et al., 2017). Penghilangan *outlier* dapat meningkatkan akurasi metode klasifikasi (Sharma et al., 2015). Namun, pada metode *outlier removal* sebagai *preprocessing* membutuhkan waktu yang cukup besar (Levner, 2005). Untuk mengatasi waktu komputasi yang besar dibutuhkan metode *one-stage process* yang menjalankan proses *outlier removal* dan *clustering* maupun klasifikasi secara bersamaan (Gan & Ng, 2017c). Namun untuk menggunakan metode tersebut dibutuhkan metode klasifikasi berbasis *centroid* (Levner, 2005) (Tamatjita & Mahastama, 2017). Dengan berbagai pertimbangan kajian pustaka, maka diusulkan metode klasifikasi *one-stage* dengan harapan dapat meningkatkan akurasi klasifikasi dengan waktu yang cukup singkat.

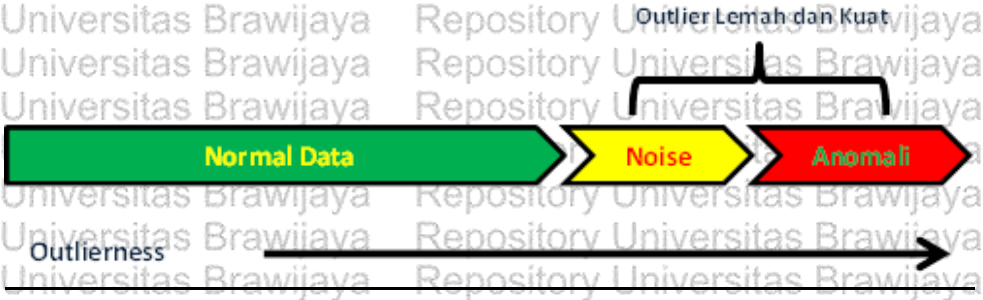
2.2 Outlier

Terdapat beberapa definisi *outlier*. *Outlier* adalah sampel yang menyimpang dari anggota lain (Grubbs, 1969). Observasi (atau subset dari pengamatan) yang tampaknya konsisten dengan sisa dari kumpulan data tersebut (Barnett and Lewis, 1994). *Outlier* adalah data yang ekstrim, sebuah titik yang termasuk dalam kelas A tetapi sebenarnya terletak di dalam kelas B sehingga klasifikasi sebenarnya dari titik tersebut diluar dugaan pengamat (John, 1995). *Outlier* dapat disebut sebagai ketidaknormalan, ketidaksesuaian, penyimpangan, atau *anomali* dalam *data mining* dan statistika. Pada proses mendapatkan data, data diperoleh dari satu atau lebih pengamat atau sistem. Ketika terdapat proses yang

tidak biasa, kejadian tersebut dapat menghasilkan *outlier*. Namun, beberapa *outlier* sering terdapat informasi yang berguna. Beberapa contoh *outlier* sebagai informasi yang berguna adalah system deteksi intrusi, kecurangan kartu kredit, *diagnosis* medis, dan *geo-information* (Aggarwal, 2017). Contoh *outlier* dalam data dua dimensi akan ditunjukkan oleh Gambar 2.1.

Pada Gambar 2.1, ditunjukkan bahwa terdapat observasi yang berjauhan dengan observasi lain pada umumnya terdapat titik data dengan kelas *a* dan *b* berwarna hitam yang jauh dengan mayoritas titik data dengan kelas yang bersangkutan. Jarak antara observasi normal dengan *outlier* dapat berbeda-beda, namun dapat dibedakan sebagai *weak* dan *strong outlier*. Tingkatan *outlier* berbeda-beda tergantung kebutuhan analisis dan peneliti. Perlakuan yang diberikan dapat berupa penghilangan, *scoring*, maupun analisis mendalam terhadap *outlier* itu sendiri. Spektrum data ditunjukkan oleh Gambar 2.2.

Gambar 2.1 Contoh *outlier* dalam data dua dimensi



Gambar 2.2 Spektrum data

Beberapa penelitian terkait efek *outlier* terhadap *data mining* telah dilakukan. Pada penelitian yang dilakukan (Aparna & Nair, 2016) terjadi peningkatan akurasi dan waktu komputasi setelah *outlier* dihilangkan. Pada penelitian (Jiang et al., 2016), pemilihan *centroid* awal pada kasus *clustering* amat menentukan performa. Hasil yang buruk didapatkan jika *centroid* awal yang dipilih adalah *outlier*. Sedangkan pada penelitian (Gan & Ng, 2017c), *outlier* dihilangkan bersamaan dengan proses *clustering* dan menghasilkan performa *clustering* yang baik serta waktu komputasi yang cepat. Dengan demikian dapat disimpulkan bahwa penanganan terhadap *outlier* menghasilkan peningkatan performa pada proses *data mining*.

2.3 Metode deteksi *outlier*

Pada subbab ini akan dijelaskan mengenai metode deteksi *outlier* berbasis statistika, jarak, densitas, dan *clustering*. Tujuan dari metode-metode tersebut adalah mendeteksi *outlier*, namun memiliki dasar pemikiran serta cara yang berbeda. Metode deteksi *outlier* berbasis statistika adalah dasar dalam ranah deteksi *outlier*. Metode lain merujuk pada metode deteksi *outlier* berbasis statistika. Metode deteksi *outlier* berbasis jarak membandingkan jarak antar titik data, sehingga dapat diketahui jarak data yang terjauh dan kemungkinan adalah *outlier*. Metode deteksi *outlier* berbasis densitas melihat kedekatan antar sekumpulan titik data. Sedangkan pada metode deteksi *outlier* berbasis *clustering*, mengelompokkan titik-titik data terhadap suatu cluster. Titik data yang berjauhan dengan semua atau mayoritas cluster dapat dianggap sebagai *outlier*. Keterangan lebih lengkap disajikan pada subbab 2.3.1, subbab 2.3.2, subbab 2.3.3, dan subbab 2.3.4.

2.3.1 Metode deteksi *outlier* berbasis statistika

Metode pendeteksian *outlier* berbasis statistik bergantung pada pendekatan statistik yang mengasumsikan distribusi atau model probabilitas agar sesuai dengan *data set*. Pada distribusi data yang diasumsikan sesuai dengan *data set*, *outlier* adalah titik-titik data yang tidak sesuai dengan model data yang ada. (Hawkins, 1980)

Metode deteksi *outlier* berbasis statistik dapat secara luas diklasifikasikan ke dalam dua kategori, yaitu, metode parametrik dan metode non-parametrik. Perbedaan utama antara kedua kategori metode ini terletak pada asumsi distribusi data. Metode parametrik mengasumsikan distribusi data dan memperkirakan parameter distribusi dari data yang diberikan sedangkan metode non-parametrik tidak mengasumsikan distribusi data (Barnett & Lewis, 1996).

Salah satu contoh metode *outlier detection* yang populer adalah metode *Box Plot*. Pada metode *Box Plot* terdapat lima nilai yaitu maksimum, minimum, *upper quartile*, *lower quartile*, dan *median*. Metode *Box Plot* pada umumnya mendefinisikan *outlier* dengan titik data pada 1.5 *inter-quartile range* (IQR), dengan nilai IQR adalah selisih antara nilai *upper* dan *lower quartile*. Berikut ini adalah contoh visualisasi *Box Plot* yang ditunjukkan pada Gambar 2.3.

Metode pendeteksian *outlier* berbasis statistik menampilkan beberapa keuntungan yaitu metode tersebut benar secara matematis dan jika diberi model probabilistik akan menjadi sangat efisien dan memungkinkan untuk mengungkap nilai dari *outlier* yang ditemukan. Selain itu, model yang dibangun, sering disajikan dalam bentuk yang ringkas, memungkinkan untuk mendeteksi *outlier* tanpa menyimpan *data set* asli yang biasanya berukuran besar.

Gambar 2.3 Visualisasi metode *box plot*

Namun, metode deteksi *outlier* berbasis statistik, khususnya metode parametrik, mengalami beberapa kelemahan. Pertama, metode tersebut pada umumnya tidak diterapkan dalam *data set* multi-dimensi karena sebagian besar model distribusi biasanya berlaku untuk data *univariate*. Dengan demikian, metode tersebut tidak cocok bahkan untuk *data set* multi-dimensi. Kelemahan tersebut sangat membatasi penerapan metode deteksi *outlier* karena dalam sebagian besar kasus memiliki data berdimensi tinggi. Pada akhirnya, kualitas hasil deteksi tidak dapat dijamin karena metode statistika sangat tergantung pada distribusi yang dipilih agar sesuai dengan data. Tidak ada jaminan bahwa data yang diperiksa sesuai dengan distribusi yang diasumsikan jika tidak ada perkiraan kepadatan distribusi berdasarkan data empiris. Dari pembahasan di atas, dapat disimpulkan bahwa metode statistika terbatas pada data dunia nyata yang pada umumnya berukuran besar.

Penelitian (X. Sun, Wu, Zhang, & Wang, 2019) adalah salah satu penelitian yang menggunakan deteksi *outlier* berbasis statistika. Penelitian ini menggunakan metode *box-plot* yang ditransformasi dengan metode *median kriging*. Metode *median kriging* dapat mengidentifikasi *local* dan *global outlier*. Metode *median kriging* efektif dalam mengatasi data dengan jumlah *outlier* yang besar, namun kurang efektif untuk data dengan *outlier* yang sedikit. Transformasi data menggunakan *median kriging* memecahkan permasalahan dimensionalitas namun hanya mendapat hasil yang baik untuk beberapa data set saja.

2.3.2 Metode deteksi *outlier* berbasis jarak

Metode untuk mendefinisikan *outlier* dari perspektif jarak telah banyak dilakukan. Metode berbasis jarak didefinisikan berdasarkan konsep *local neighborhood* atau *k-Nearest Neighbor* (*k*-NN) dari titik-titik data. Metode *outlier* berbasis jarak tidak berdasar asumsi distribusi data. Selain itu, metode berbasis jarak lebih baik diterapkan pada data berdimensi banyak dan dapat dihitung jauh lebih efisien daripada berbasis statistik.

Dalam metode berbasis jarak, jarak antar titik data perlu dihitung. Fungsi seperti jarak *Manhattan* atau *Euclidean* dapat digunakan untuk mengukur jarak antar titik. Normalisasi dilakukan untuk menormalkan berbagai skala fitur data.

$$Euclidean\ d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + \cdots + (q_n - p_n)^2} \quad (1)$$

$$Manhattan\ d(p,q) = d(q,p) = |q_1 - p_1| + \cdots + |q_n - p_n| \quad (2)$$

Jika merujuk pada penelitian (Bakar & Mohemad, 2006), langkah-langkah dalam metode *outlier* berbasis jarak pada umumnya adalah sebagai berikut:

1. Lakukan pengumpulan data.
2. Hitung jarak antar titik data menggunakan metode *Manhattan* atau *Euclidean* d_1.
3. Identifikasi nilai jarak maksimal pada data d_2.
4. Tentukan threshold nilai jarak d_3.
5. Bandingkan d_3 dengan d_1. Pada fase ini, nilai parameter p dapat ditentukan membandingkan d_3 dengan d_1, dengan nilai p berada diantara d_3 dengan d_1.
6. Tentukan threshold t yang menunjukkan besaran ruang pencarian.
7. Bandingkan p dan t.
8. Pengujian data. Pada fase ini *outlier* dapat diidentifikasi.
9. Analisis dan perbandingan output.

Keuntungan dari metode berbasis jarak adalah metode berbasis jarak adalah non-parametrik dan tidak bergantung pada distribusi yang diasumsikan untuk

menyesuaikan data. Definisi jarak untuk *outlier* cukup mudah dipahami dan diimplementasikan. Kelemahan mereka adalah tidak efektif dalam ruang dimensi tinggi karena *curse of dimensionality* meskipun jarak *Euclidean* telah ditemukan. Selain itu, lingkungan dan pencarian *k*-NN dalam data dimensi tinggi adalah tugas yang tidak mudah dan membutuhkan komputasi yang tinggi. Algoritme yang didasarkan pada *nested-loop*, biasanya membutuhkan $O(N^2)$ pada perhitungan jarak. Penskalaan kuadratik berarti bahwa akan sangat sulit untuk mencari *outlier* saat data yang diatasi menjadi besar.

2.3.3 Metode deteksi *outlier* berbasis densitas

Asumsi dasar dari metode berbasis densitas adalah kepadatan titik data yang valid sama dengan kepadatan disekitar tetangga terdekatnya, namun kepadatan disekitar *outlier* jelas berbeda dengan tetangga disekitarnya. Metode berbasis densitas menggunakan mekanisme yang lebih kompleks untuk memodelkan tingkat *outlier* suatu titik data dari pada metode berbasis jarak. Pada umumnya melibatkan penyelidikan *local density* dari titik yang sedang dipelajari dan *local density* tetangga terdekatnya. Dengan demikian ukuran *outlier* titik data adalah relatif dalam arti bahwa pada umumnya rasio kepadatan suatu titik terhadap kepadatan rata-rata tetangga terdekatnya. Metode berbasis densitas memiliki kemampuan pemodelan yang lebih kuat terhadap *outlier* tetapi memerlukan komputasi yang tinggi.

Salah satu metode deteksi *outlier* berbasis densitas adalah *Local Outlier Factor* (LOF) (Breunig, Kriegel, Ng, & Sander, 2000). LOF menggunakan jarak yang dikonversi menjadi kerapatan. Kerapatan didapatkan dengan mencari jarak tetangga sebanyak *k*. Titik data dengan jarak tetangga yang tinggi akan ditetapkan sebagai *outlier*.

Metode deteksi *outlier* berbasis densitas umumnya lebih efektif daripada metode berbasis jarak. Namun, untuk meningkatkan efektivitas metode berbasis densitas lebih rumit dan komputasi tinggi.

2.3.4 Metode deteksi *outlier* berbasis *clustering*

Beberapa metode data mining menganggap *outlier* sebagai sampingan dari *clustering* itu sendiri dan mendefinisikan *outlier* sebagai titik yang tidak terletak atau terletak berjauhan dari cluster lainnya. Dengan demikian, teknik *clustering* secara implisit mendefinisikan *outlier* sebagai *noise* dari cluster. Sejauh ini, ada beberapa penelitian tentang *clustering*, dan beberapa dari penelitian tersebut dilengkapi dengan beberapa mekanisme untuk mengurangi efek buruk *outlier*, seperti CLARANS, DBSCAN, BIRCH, WaveCluster. Beberapa tahun ini, penelitian

tentang teknik *clustering* yang disesuaikan dengan untuk data berdimensi tinggi telah dilakukan.

Salah satu penelitian *outlier* detection berbasis *clustering* adalah (Xiong, Pandey, Steinbach, & Kumar, 2006). Pada penelitian tersebut, dilakukan sorting berdasar *similarity*, sama seperti *k*-Means. Deteksi *outlier* berbasis *clustering* membutuhkan parameter k yang dijadikan jumlah cluster untuk proses *clustering*. Pada penelitian tersebut, akurasi terbaik didapatkan ketika nilai k hamper sama atau mendekati jumlah kelas, namun tidak menutup kemungkinan jumlah k yang berbeda dengan jumlah kelas menghasilkan performa yang lebih baik.

Deteksi *outlier* menggunakan analisis *clustering* cukup intuitif dan konsisten dengan persepsi manusia tentang *outlier*. Selain itu, *clustering* adalah area penelitian solid dan terdapat banyak algoritme *clustering* yang dapat dipilih untuk mendeteksi *outlier*. Namun demikian, beberapa peneliti berpendapat bahwa, algoritme *clustering* seharusnya tidak dianggap sebagai metode pendeteksian *outlier*, karena tujuan mereka hanya untuk mengelompokkan objek dalam *data set* sehingga fungsi *clustering* dapat dioptimalkan. Tujuannya untuk menghilangkan *outlier* dalam dataset menggunakan algoritme *clustering* hanya untuk meredam efek buruk mereka pada hasil akhir *clustering*. Namun tidak dapat dipungkiri bahwa metode deteksi *outlier* berbasis *clustering* dapat dan mencapai hasil yang relatif baik dibandingkan metode yang lain.

Salah satu pemanfaatan *outlier removal* pada kasus *clustering* telah dilakukan (Gan & Ng, 2017c). *k*-Means *with outlier removal* atau KMOR adalah metode pengelompokan dengan penghilangan *outlier* dalam prosesnya. KMOR memodifikasi algoritme *k*-Means dengan menambahkan 1 kelompok tambahan *(k+1)* untuk menampung *outlier*. Langkah-langkah KMOR hampir sama dengan algoritme *k*-Means, namun terdapat fase *outlier removal* pada setiap perulangannya. Tahapan metode KMOR adalah sebagai berikut:

1. Memilih titik data sebanyak *k* untuk dijadikan titik pusat kelompok awal $Z = \{z_1, z_2, z_3, \dots, z_k\}$.
2. Melakukan cek terhadap tiap titik data pada *data set* $X = \{x_1, x_2, x_3, \dots, x_n\}$ terhadap tiap titik pusat kelompok Z.
3. Masukkan hasil pengelompokan awal pada U. U adalah matriks biner yang menampung hasil pengelompokkan dengan kolom k+1 dan baris n.
4. Cari jarak rata-rata tiap titik data pada kelompok yang bersangkutan menjadi nilai d_{avg}.
5. Lakukan pembaruan nilai titik pusat Z berdasarkan rata-rata nilai fitur titik data yang terdapat pada kelompok yang bersangkutan.

6. Lakukan cek terhadap tiap titik data pada *data set* $X = \{x_1, x_2, x_3, \ldots, x_n\}$ terhadap tiap titik pusat kelompok Z yang baru dan dengan syarat jika jarak suatu titik data berada $> \gamma * d_{avg}$, maka titik data tersebut dinyatakan sebagai *outlier* dan akan dimasukkan pada kelompok *outlier*.
7. Ulangi langkah 3, 4, dan 5 hingga menemui syarat pemberhentian. Syarat pemberhentian ada dua, yaitu tidak ada titik data yang berpindah kelompok dan mencapai batas perulangan yang ditentukan.

Pada penelitian yang telah dilakukan, KMOR membuktikan performa yang baik dalam waktu proses dan kualitas pengelompokan. Performa KMOR pada penelitian sebelumnya ditunjukkan oleh Tabel 2.1.

Tabel 2.1 Tabel perbandingan performa algoritme KMOR

	KMOR	ODC	*k*-Means	NEO *k*-Means
R	0.91	0.87	0.32	0.34
Me	0.05	0.16	0.95	0.94
Waktu Proses	0.003s	0.003s	0.019s	0.015

Sumber: (Gan & Ng, 2017c)

R adalah ukuran ketepatan *clustering*, dan Me adalah ukuran performa algoritme untuk deteksi *outlier*. Semakin tinggi R dan semakin rendah Me menandakan hasil yang baik. Terbukti KMOR dapat mengungguli performa algoritme lain dalam melakukan *clustering*. Dalam hitungan waktu proses KMOR termasuk yang paling cepat dibandingkan metode-metode pembanding.

Performa KMOR sudah terbukti, namun KMOR adalah modifikasi dari metode *k*-Means yang merupakan metode untuk *clustering*. KMOR belum digunakan untuk kasus klasifikasi. Namun, ide tentang klasifikasi dengan *outlier removal* dapat terinspirasi dari KMOR.

2.4 Metode klasifikasi

Pada subbab ini menerangkan mengenai metode klasifikasi yang digunakan, yaitu *Naïve Bayes, k-Nearest Neighbor, dan Nearest Centroid Classifier.*

2.4.1 Metode *Naïve Bayes* (NB)

Naïve Bayes adalah metode klasifikasi berbasis statistika. NB dapat memprediksi kelas suatu titik data berdasarkan kemungkinan sifat-sifat titik data pada kelas tertentu. NB mengasumsikan nilai tiap fitur adalah independen. Untuk masalah klasifikasi, tentukan $P(C|X)$ dengan X berada pada kelas C, yaitu kemungkinan hipotesis C pada data X. Dengan kata lain, NB mencari kemungkinan deskripsi fitur X yang terdapat pada kelas C. cara untuk mengukur

kemungkinan dapat diketahui dengan teorema *Bayes*. Teorema *Bayes* akan ditunjukkan oleh rumus sebagai berikut.

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)} \quad (3)$$

Dimana:

1. $P(C|X)$ adalah *posterior probability* yaitu kemungkinan terjadi sesuatu setelah kejadian terjadi dengan C adalah kelas dan X adalah fitur.
2. $P(C)$ adalah *prior probability* dari kelas C, yaitu kemungkinan dipilihnya kelas tersebut sebelum kejadian.
3. $P(X|C)$ adalah kemungkinan munculnya nilai X pada kelas C.
4. Sedangkan $P(X)$ adalah kemungkinan munculnya nilai X.

Kelebihan dari metode NB adalah mudah dan sederhana dalam implementasi, serta cepat dalam klasifikasi. Pada data set dengan titik data yang unik satu sama lain, NB memiliki kemampuan prediksi yang baik sekali, dengan demikian performa NB lebih baik pada data set kategori. Sedangkan NB memiliki kelemahan jika pada data uji tidak terdapat data yang ada pada data latih, maka kemungkinan pemilihan kelas yang benar akan dianggap nol. Dalam kasus lain, untuk meningkatkan performa NB dibutuhkan data set yang besar.

Meskipun NB adalah metode yang sederhana dan dasar dalam hal klasifikasi, namun masih digunakan hingga sekarang dengan berbagai macam modifikasi. Penelitian (N. Sun, Sun, Lin, & Wu, 2018) menggunakan metode NB dengan modifikasi *pruning class* untuk *big data*. Penelitian tersebut menghasilkan peningkatan dalam kecepatan waktu komputasi.

2.4.2 Metode *k-Nearest Neighbor* (*k*-NN)

Metode *k-Nearest Neighbor* adalah metode klasifikasi berbasis analogi. Analogi yang dimaksud adalah *k*-NN membandingkan data uji dengan data latih yang sama atau mendekati kemiripannya. Langkah klasifikasi metode *k*-NN yaitu lakukan perhitungan jarak terhadap semua titik data latih, pilih titik data latih terdekat sebanyak *k*, dan pilih label yang terbanyak dari hasil *voting* data latih k.

Kedekatan dalam *k*-NN didefinisikan dengan ukuran jarak, *Euclidean distance* (1) maupun *Manhattan distance* (2) diantaranya.

Tahapan metode *k*-NN adalah sebagai berikut:

1. Masukkan data, anggap data latih sebagai Tr dan data uji sebagai Ts.
2. Tentukan nilai k.
3. Hitung jarak antara data uji Ts ke-j dengan
   a. Data latih Tr ke-i hingga i=n, n adalah banyaknya data latih.

b. Lakukan pengurutan dari jarak terdekat hingga terjauh.
c. Dapatkan data latih dengan jarak terdekat sebanyak *k*.
d. Dapatkan kelas yang paling sering muncul diantara *k* data latih tersebut.
e. Tentukan kelas data uji *Ts ke-j* tersebut berdasar hasil langkah 3d.

4. Ulangi langkah 3 hingga data uji *Ts ke-m*, m adalah banyaknya data uji.

K adalah parameter jumlah tetangga terdekat yang akan dipilih. Pemilihan parameter *k* sangat menentukan hasil klasifikasi *k*-NN. Ketika *k=1*, maka label tetangga terdekat akan dipilih langsung tanpa melalui *voting*. Namun jika *k>1*, maka harus terdapat proses voting dengan label terbanyak yang akan dipilih. Pemilihan *k* juga menentukan kecepatan waktu komputasi. Semakin besar *k* yang dipilih maka semakin besar waktu komputasi. Namun ketika nilai *k* relatif kecil, pengaruh *outlier* akan makin besar jika terdapat *outlier* dalam tetangga terpilih.

k-NN menggunakan jarak sebagai indikator klasifikasi, dengan demikian metode untuk menghitung jarak menjadi penting untuk diperhatikan. Jarak didapatkan dari perhitungan beda nilai antar fitur. Fitur yang tidak relevan atau *noisy* dapat menurunkan akurasi. Solusi untuk mengatasi fitur yang tidak relevan adalah pembobotan atau seleksi. Pemilihan persamaan jarak dapat menjadi pembeda dalam mencapai hasil akurasi.

Waktu komputasi *k*-NN cukup tinggi. Hal ini disebabkan perhitungan jarak antar titik data dan proses pengurutan jarak. Dalam menangani proses pengurutan jarak, dapat digunakan metode *Partial Sorting* dengan mengurutkan titik data sebanyak *k* saja. Sedangkan untuk mengatasi perhitungan jarak titik data dapat dilakukan *pruning* atau *outlier removal* yaitu menghapus atau menghiraukan titik data yang tidak berpengaruh atau jauh terhadap titik data uji.

Penelitian dalam meningkatkan kemampuan *k*-NN dapat dilakukan dengan menambahkan *preprocessing weighted feature selection* (Bugata & Drotár, 2019). Dasar pemikiran metode ini adalah tidak semua fitur relevan untuk digunakan dalam analisis. Fitur yang tidak relevan dapat memberikan kesalahan prediksi. Sedangkan semakin banyak fitur maka semakin lama waktu komputasi yang dibutuhkan. Metode yang diajukan menggunakan pembobotan fitur menggunakan *gradient descent*. Dengan skala bobot 0 hingga 1, jika terdapat fitur dengan bobot 0 maka fitur tersebut tidak akan diikutkan dalam perhitungan. Hasil dari penelitian tersebut menghasilkan peningkatan akurasi yang cukup baik.

2.4.3 Metode *Nearest Centroid Classifier* (NCC)

NCC adalah metode klasifikasi berbasis *centroid* (Tamatjita & Mahastama, 2017). NCC adalah metode klasifikasi yang menetapkan kelas pada suatu titik data berdasarkan *centroid* atau pusat data dari suatu kelas yang terdekat dengan

titik data tersebut. *Centroid* didapatkan dari rata-rata nilai fitur pada data latih pada masing-masing kelas yang bersangkutan.

Keunggulan NCC adalah tidak memiliki parameter, jadi hasil yang didapatkan tidak bergantung pada konfigurasi parameter, namun pada sebaran data. Metode untuk mendapatkan jarak pada penelitian ini menggunakan *Euclidean distance*. Adapun tahapan pada metode NCC adalah sebagai berikut:

1. Diketahui data latih X dengan titik data sebanyak n dan fitur sebanyak m $X = \{x_1, x_2, x_3, \dots, x_n\}$, contoh $x_1 = \{x_{11}, x_{12}, x_{13}, \dots, x_{1m}\}$, maka ambil rata-rata nilai tiap fitur untuk tiap kelas dan jadikan nilai titik pusat kelas yang bersangkutan pada $Z = \{z_1, z_2, z_3, \dots, z_k\}$.
2. Lakukan uji terhadap data uji $Y = \{y_1, y_2, y_3, \dots, y_h\}$ dengan titik data sebanyak h terhadap tiap titik pusat kelas Z, berikan label menurut pusat Z yang paling dekat dengan titik data uji yang bersangkutan.

Meskipun NCC dan *k*-NN sama-sama menggunakan jarak, namun terdapat perbedaan pada cara pelabelan. Pada metode *k*-NN, semua titik data akan diukur jaraknya dan pada titik data sebanyak k akan dilakukan *voting* terhadap kelas untuk kemudian ditetapkan pada titik data uji. Sedangkan metode NCC jarak digunakan untuk menentukan titik *centroid* terdekat dengan titik data uji.

2.5 *Feature scaling*

Feature scaling adalah proses untuk menskalakan nilai setiap fitur. Tujuan dari proses *feature scaling* adalah untuk menormalisasi setiap fitur yang umumnya dengan rentang *[0,1]* (Aksoy & Haralick, 2001). Tujuan dari *feature scaling* adalah menyamakan rentang tiap fitur supaya tidak ada fitur yang bobot nilainya terlalu timpang dan mempengaruhi perhitungan jarak secara signifikan. Salah satu dari metode penskalaan adalah *linear scaling to unit range* atau *min-max normalization*. *Min-max normalization* menskalakan nilai fitur berdasarkan nilai minimum dan maksimum. *Min-max normalization* menskalakan nilai tiap fitur menjadi 0 hingga 1 dengan 0 adalah nilai minimum pada suatu fitur dan 1 adalah nilai maksimal pada suatu fitur. Persamaan *Min-max normalization* akan ditunjukkan pada persamaan 4.

$$z' = \frac{z - min(z)}{max(z) - min(z)} \quad (4)$$

z' adalah nilai fitur yang akan diskalakan. z adalah nilai fitur sebelum diskalakan. *Min(z)* adalah nilai minimum pada fitur yang bersangkutan dan *max(z)* adalah nilai maksimum pada fitur yang bersangkutan.

2.6 *Sparsity*

Salah satu faktor penting yang mempengaruhi kinerja metode klasifikasi adalah derajat Sparsity data set tersebut. Sparsity adalah derajat persebaran antar titik data dalam satu data set (Phan, Nguyen, & Horiguchi, 2008). Perhitungan sparsity dapat diketahui dengan rumus sebagai berikut:

$$S_d = 1 - \frac{\sum_i^n N_i}{n * |V|} \quad (5)$$

Dengan keterangan N_i adalah nilai fitur dalam titik data i. n adalah jumlah titik data pada data set. Dan V adalah jumlah fitur. Pada penelitian (Saif, Fernandez, He, & Alani, 2013), rumus tersebut digunakan untuk mengetahui sparsity data set text Twitter. Untuk menyesuaikan dengan kebutuhan penelitian dengan data set numerik dengan rentang yang berbeda-beda, maka rumus tersebut disesuaikan dengan normalisasi data.

2.7 Metode pengujian *F-Measure*

Pada dasarnya, performa metode klasifikasi didefinisikan dengan seberapa banyak sampel yang diklasifikasi dengan benar pada data uji, dalam hal ini ukuran tersebut dinamakan akurasi. Salah satu metode untuk melakukan pengecekan performa klasifikasi adalah dengan membuat *confusion matrix*. *Confusion matrix* adalah *matrix* yang berukuran L *x* L dan terdapat perbandingan hasil sebenarnya (*actual label*) dan hasil klasifikasi (*predicted label*). Dalam kasus klasifikasi, pengukuran yang digunakan adalah *F-measure* (Gottlieb, Kontorovich, & Nisnevitch, 2014). Contoh *confusion matrix* akan ditunjukkan pada Tabel 2.2.

Tabel 2.2 *Confusion Matrix* dengan 2 kelas *YES* dan *NO*

	Predicted YES	***Predicted NO***
Actual YES	*True Positive (TP)*	*False Positive (FP)*
Actual NO	*False Negative (FN)*	*True Negative (TN)*

Keterangan:

1. *True Positive* (TP): kasus yang predicted *YES* dan pada *actual YES*.
2. *True Negative* (TN): kasus yang predicted *NO* dan pada *actual NO*.
3. *False Positive* (FP): kasus yang predicted *YES*, namun pada *actual NO*.
4. *False Negative* (FN): kasus yang predicted *NO*, namun pada *actual YES*.

Penelitian ini menggunakan pengukuran *F-measure* dan akurasi untuk mengukur seberapa sering prediktor melakukan klasifikasi dengan benar. *F-*

measure adalah rerata antara *precision* dan *recall*. *Presicion* ditunjukkan pada Persamaan (7), *Recall* ditunjukkan pada Persamaan (8), *F-measure* ditunjukkan Persamaan (9), dan akurasi ditunjukkan pada Persamaan (10).

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

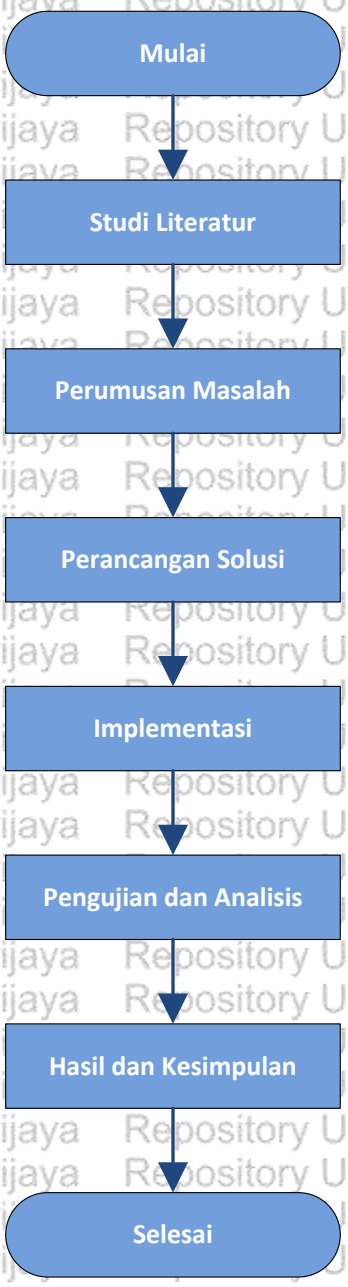
$$F - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \quad (8)$$

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (9)$$

BAB 3 METODOLOGI

3.1 Metode penelitian

Penelitian ini dijalankan dengan melakukan implementasi metode yang diusulkan dengan tujuan meningkatkan performa metode klasifikasi pada data dengan menghilangkan *outlier* sebagai fokus penelitian. Tujuan dari penelitian ini adalah mengetahui performa metode yang diusulkan dalam melakukan klasifikasi. Alur metode penelitian yang diajukan ditunjukkan pada Gambar 3.1.



Gambar 3.1 Alur metode penelitian

Tahapan yang dilakukan dalam penelitian ini adalah studi literatur, perumusan masalah, perancangan solusi, implementasi, pengujian dan analisis, diakhiri dengan penyampaian hasil dan kesimpulan. Studi literatur yang dilakukan difokuskan terhadap pengaruh *outlier* dalam klasifikasi dan penelitian yang terkait. Lalu, masalah dirumuskan dengan merujuk pada studi literatur. Uraian masalah harus dapat diukur dan diuji. Setelah masalah dirumuskan, maka solusi dibentuk dengan merancang metode usulan yang diharapkan dapat menyelesaikan masalah. Implementasi dilakukan dengan merujuk pada rancangan solusi atau metode usulan. Setelah proses implementasi selesai, pengujian dilakukan dengan perangkat lunak yang telah jadi. Hasil pengujian akan dianalisis untuk menjawab rumusan masalah. Hasil dan kesimpulan dapat dipaparkan sebagai akhir dari proses riset.

3.2 Studi literatur

Studi literatur yang dilakukan adalah mengumpulkan dasar teori mengenai permasalahan pengaruh *outlier* pada performa klasifikasi serta metode-metode yang digunakan dalam penelitian yang bersangkutan. Terdapat beberapa penelitian dengan metode, kasus, *data set*, dan instrumen uji yang berbeda-beda, namun dapat digunakan sebagai landasan pemikiran penelitian ini. Metode yang telah dilakukan antara lain *k*-NN dan *Naïve Bayes* yang kemudian akan menjadi metode pembanding. Sedangkan metode usulan pada penelitian ini adalah NCCOR yang berasal dari metode NCC dengan KMOR yang memiliki kemampuan klasifikasi dan *outlier removal* dalam satu proses. Semua metode akan dibandingkan dengan metode evaluasi akurasi untuk mengetahui seberapa akurat klasifikasi metode yang bersangkutan dan perbandingan waktu.

3.3 Data

Data pada penelitian ini adalah data yang bersumber dari pusat *repository data set* yang populer yaitu UCI https://archive.ics.uci.edu/ dengan penyesuaian tambahan informasi mengenai *outlier* pada http://odds.cs.stonybrook.edu/. *Data set* yang akan digunakan memiliki sifat yang berbeda-beda dalam ukuran jumlah titik data, fitur, dan *outlier*. Penggunaan data dengan sifat yang berbeda diharapkan memberikan informasi tentang perbedaan hasil klasifikasi. Data yang digunakan adalah data numerik dengan nilai fitur yang berbeda-beda. Dengan demikian diperlukan penskalaan nilai. Penskalaan nilai yang digunakan adalah *Min-Max Normalization* yang merujuk pada persamaan (8).

Karakteristik *data set* dapat berdasarkan pada persentase *outlier* dan distribusi kelas. Karakteristik data set dengan persentase *outlier* tinggi

berdasarkan rata-rata persentase dari keseluruhan *data set* pada repositori ODDS yaitu 8.6. Persentase diatas 8.6 akan diberikan karakteristik persentase *outlier* tinggi, sedangkan jika dibawah 8.6 akan diberikan karakteristik persentase *outlier* rendah. Sedangkan untuk keseimbangan data, suatu data set dikarakteristikkan sebagai data set *imbalance* jika terdapat satu atau lebih kelas dengan populasi kurang dari 10 dengan kelas lainnya. Sparsity dihitung dengan Persamaan (5) dengan rentang 0 hingga 1. Jika sparsity mendekati 0 maka titik data makin berdekatan satu sama lain, demikian sebaliknya.

Data yang akan digunakan pada penelitian kali ini akan ditunjukkan pada Tabel 3.1.

Tabel 3.1 Daftar *data set* yang digunakan

Data set	Titik Data	Fitur	Jumlah Kelas	Distribusi Kelas	*Outlier*	Karakteristik Data set
Breast Cancer	679	9	**2**	'2' (458) '4' (241)	35	Persentase *outlier* tinggi Data set *balance* *Sparsity* = 0.755631502
Ecoli	336	7	**8**	'cp' (143) 'im' (77) 'imS' (2) 'imL' (2) 'imU' (35) 'om' (20) 'omL' (5) 'pp' (52)	2.6	Persentase *outlier* rendah Data set *imbalance* *Sparsity* = 0.6347028
Lymphography	148	18	**4**	'3' (61) '2' (81) '4' (4) '1' (2)	4.1	Persentase *outlier* rendah Data set *imbalance* *Sparsity* = 0.5373992413
Vertebral	240	6	**3**	'DH' (60) 'SL' (150) 'NO' (100)	12.5	Persentase *outlier* tinggi Data set *balance* *Sparsity* = 0.67061456
Wine	178	13	**3**	'1' (59) '2' (71) '3' (48)	7.7	Persentase *outlier* rendah Data set *balance* *Sparsity* = 0.591508663
Yeast	1364	8	**10**	'MIT' (244) 'NUC' (429) 'CYT' (463) 'ME1' (44) 'EXC' (35) 'ME2' (51) 'ME3' (163) 'VAC' (30) 'POX' (20) 'ERL' (5)	4.7	Persentase *outlier* rendah Data set *imbalance* *Sparsity* = 0.69108627197

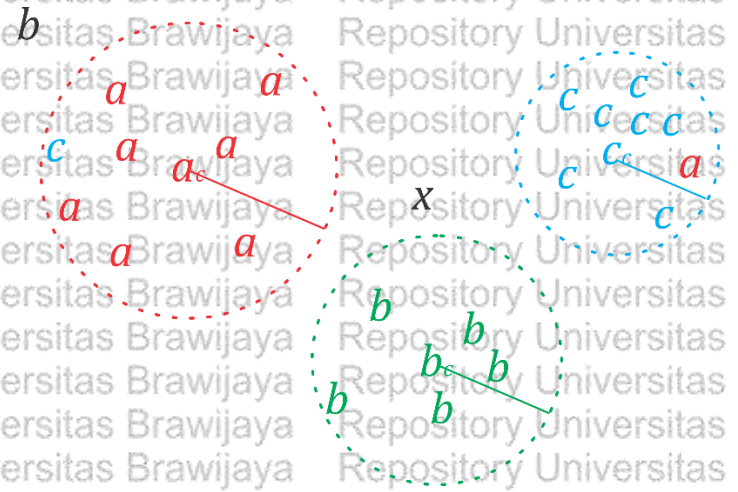
3.4 Skenario Implementasi

Implementasi metode usulan NCCOR meliputi input data set, proses NCCOR, dan menampilkan hasil klasifikasi. Langkah-langkah metode usulan NCCOR adalah sebagai berikut:

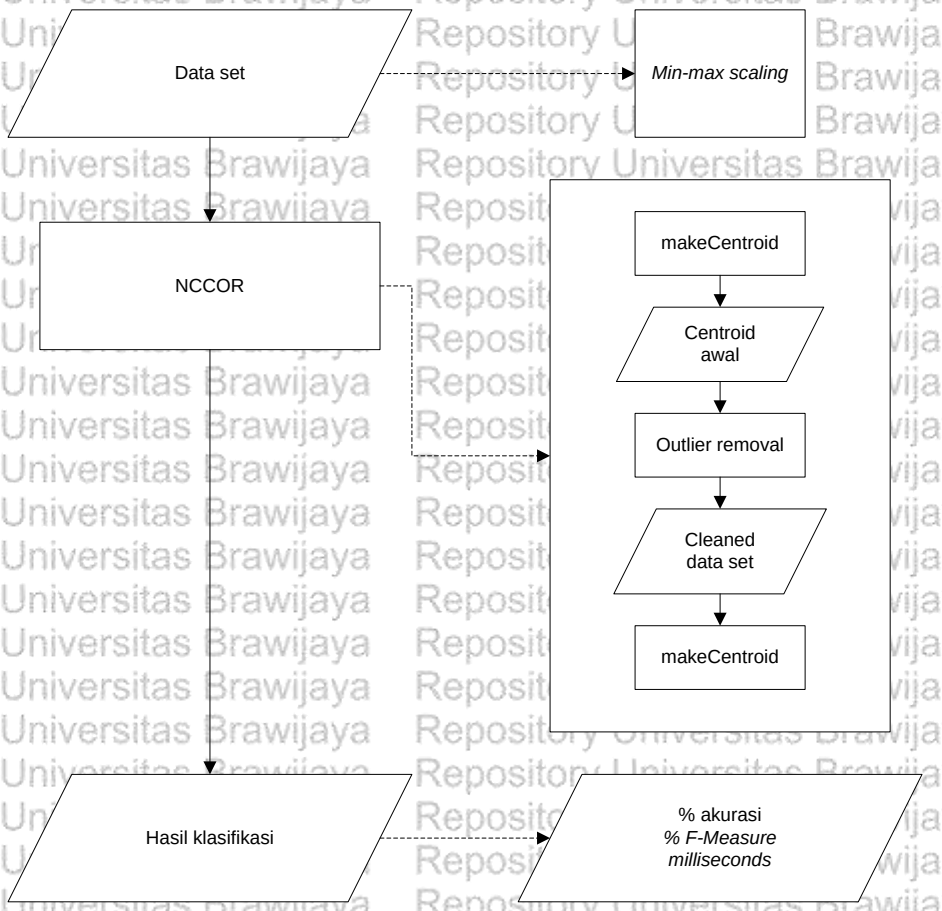
1. Pertama, input data set yang diinginkan.
2. Data set yang diinput akan diskalakan dengan metode *min-max*.
3. Lalu, pada proses NCCOR akan dilakukan pembentukan centroid awal.
4. Centroid awal lalu digunakan untuk patokan *outlier removal*.
5. Hilangkan titik data dengan patokan centroid awal dengan menggunakan $\gamma * d_{avg}$ sebagai *threshold radius* untuk membedakan titik data *outlier* dan bukan *outlier*.
6. Hentikan langkah 5 ketika telah mencapai jumlah *outlier* yang boleh dihilangkan $0 < n_0 \leq n$ atau sudah tidak ada data set yang berada diluar threshold radius.
7. Bentuk centroid baru dengan data set yang telah dibersihkan.
8. Klasifikasikan titik data uji menggunakan centroid baru sebagai classifier.
9. Tampilkan hasil berupa *F-measure*, akurasi, dan waktu komputasi.

Gambaran umum dari metode usulan NCCOR akan ditunjukkan Gambar 3.2 dan ilustrasi metode NCCOR akan ditunjukkan oleh gambar 3.3.

Pada Gambar 3.3, terdapat 3 jenis kelas yaitu *a, b, c*. Untuk setiap jenis kelas ikan dibentuk *centroid* yang ditunjukkan pada a_c, b_c, dan c_c. Pada a_c, b_c, dan c_c, ditarik garis lurus pada gambar menunjukkan *threshold radius* $\gamma * d_{avg}$ yang membentuk garis putus-putus menunjukkan *threshold class* yang digunakan untuk membedakan antara *outlier* dan bukan *outlier*. Semua titik data yang berada diluar garis putus-putus dan yang lebih dekat dengan kelas lain akan dihilangkan. Setelah itu akan dibentuk centroid baru yang akan digunakan untuk menentukan kelas titik data uji *x*.



Gambar 3.2 Gambaran umum metode NCCOR



Gambar 3.3 Gambaran umum skenario implementasi metode usulan NCCOR

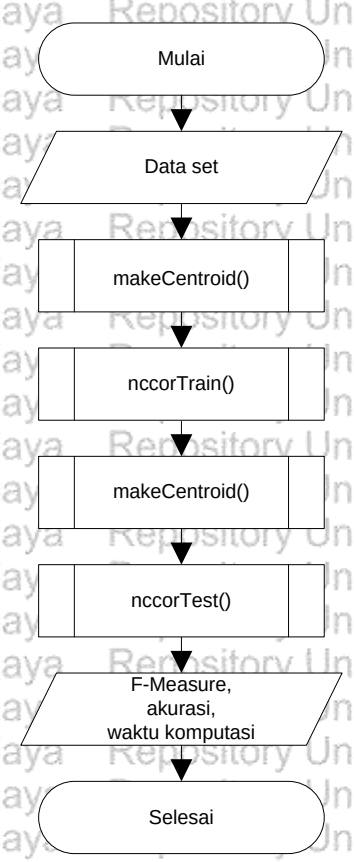
3.5 Skenario Pengujian

Pengujian dilakukan untuk mengetahui performa metode yang baik dalam klasifikasi data yang memiliki *outlier*. Metode yang akan digunakan dalam pengujian adalah NB, *k*-NN, NCC, dan NCCOR. Skenario yang akan dilakukan adalah mengukur akurasi dan waktu komputasi metode usulan dengan metode pembanding terhadap *data set* uji. Pada pengujian akurasi, metode *k*-NN, akan diujikan dengan parameter k = {1, 2, 3, …, n} dengan n adalah banyaknya data latih dan diambil hasil terbaik, sedangkan untuk metode NCCOR akan diujikan dengan parameter y = {0.5, 1, 1.5, 2, 2.5, 3} dan n_0 = {0.2, 0.4, 0.6, 0.8} . Pada waktu komputasi, akan diujikan dengan satuan milidetik untuk setiap metode.

BAB 4 PERANCANGAN

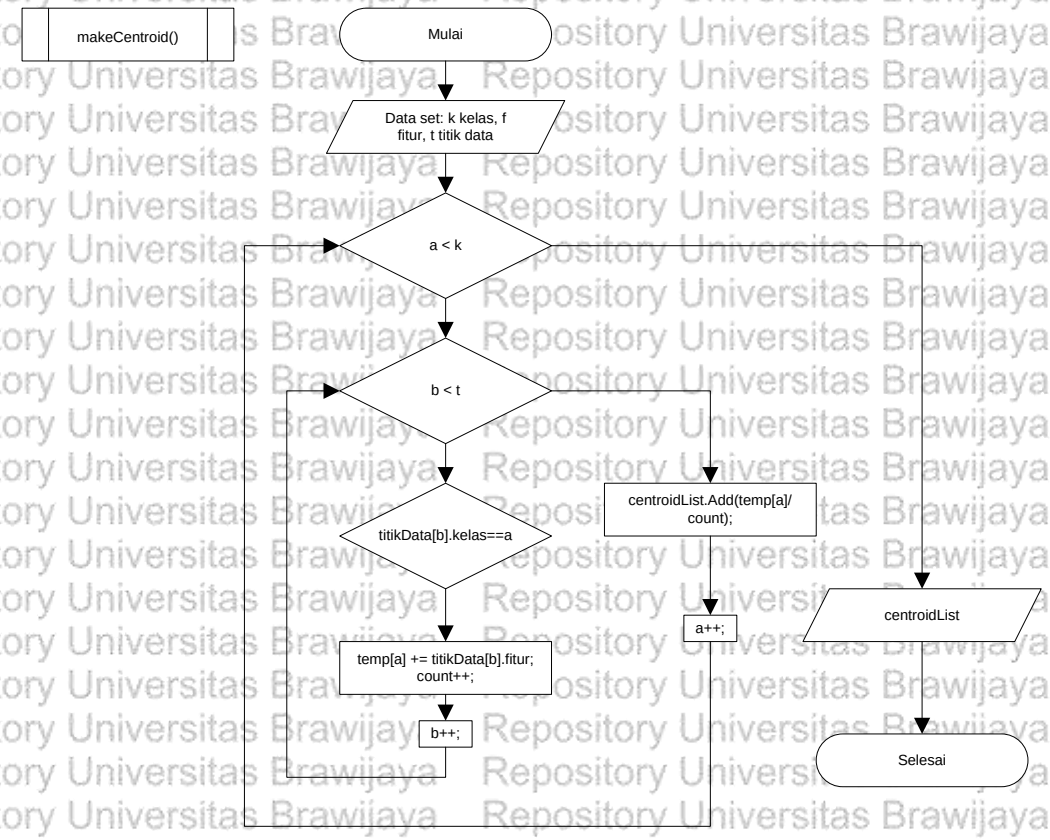
4.1 Perancangan Metode

Metode usulan NCCOR adalah metode yang memiliki ide yang sama dengan KMOR namun disesuaikan dengan metode NCC untuk klasifikasi. Perbedaan NCCOR dengan KMOR adalah pada metode usulan telah diketahui jumlah kelas jadi tidak perlu melakukan set parameter k untuk jumlah kelompok seperti pada KMOR. Gambaran umum perancangan metode NCCOR dijelaskan Gambar 4.1.

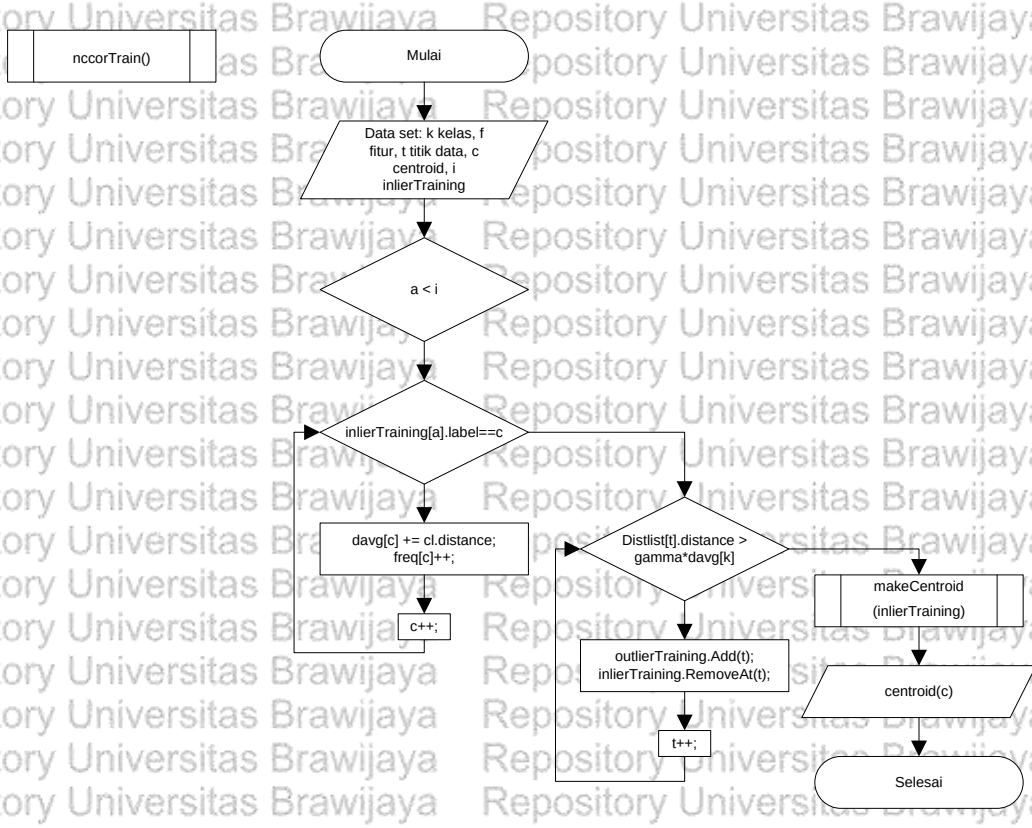


Gambar 4.1 Diagram perancangan metode NCCOR

Adapun tahapan pada metode NCCOR adalah membuat *centroid* terlebih dahulu yang ditunjukkan pada diagram alir `makeCentroid` Gambar 4.2 dan Gambar 4.4. Pada Gambar 4.2, *method* `makeCentroid` adalah prosedur untuk membentuk *centroid* yang digunakan untuk *outlier removal* dan *classifier* pada *method* `nccorTrain`. *Centroid* dibentuk dengan mengambil rata-rata seluruh fitur pada data dengan kelas tertentu. Nilai rata-rata tersebut akan dijadikan *centroid* untuk kelas tertentu dan disimpan pada `centroidList`. Proses *training* metode NCCOR dapat dilihat pada diagram alir Gambar 4.3 dan *pseudocode* Gambar 4.5.



Gambar 4.2 Diagram alir *method* `makeCentroid()`



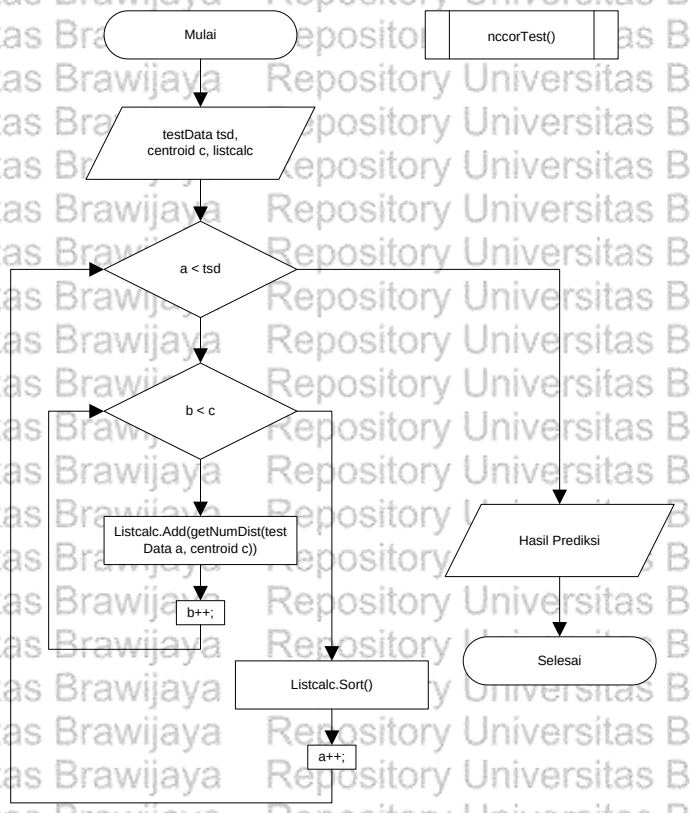
Gambar 4.3 Diagram alir *method* `nccorTrain()`

```
method makeCentroid(centroidList, centroid, dataSource)
1   BEGIN
2   for (a = 0; a < labels.Count; a++)
3       centroid.label = labels[a];
4       double[] dotemp = new double[dataSource[a].feature.Length];
5       for (int b = 0; b < dataSource[a].feature.Length; b++)
6           List<double> temp = new List<double>();
7           for (int c = 0; c < dataSource.Count; c++)
8               if (labels[a] == dataSource[c].label)
9                   temp.Add(dataSource[c].feature[b]);
10              end if
11          end for
12          dotemp[b] = temp.Average();
13      end for
14      centroid.feature = dotemp;
15      centroidList.Add(centroid);
16  end for
17  END
```

Gambar 4.4 *Pseudocode method* makeCentroid

```
method nccorTrain(trainingData trd, gamma, n0)
1   BEGIN
2   makeCentroid(cenlist, cen, trd);
3   double[] countLabel = new double[labels.Count];
4   for (int a = 0; a < inlierTraining.Count; a++)
5       cl.label = inlierTraining[a].label;
6       int index = labels.IndexOf(inlierTraining[a].label);
7       double dstnc = getNumDist(inlierTraining, a, cenlist, index);
8       cl.distance = dstnc;
9       davg[index] += dstnc;
10      freq[index]++;
11      distlist.Add(cl);
12  end for
13  for (int a = 0; a < davg.Length; a++)
14      davg[a] = Math.Round(davg[a] / freq[a], 5);
15  end for
16  for (int a = distlist.Count - 1; a >= 0; a--)
17      int ind = labels.IndexOf(distlist[a].label);
18      if (limit > (maxOut * trd.Count))
19          break;
20      end if
21      if (distlist[a].distance>gamma*davg[ind]&&countLabel[ind]>1)
22          cen.label = inlierTraining[a].label;
23          cen.feature = inlierTraining[a].feature;
24          cen.index = inlierTraining[a].index;
25          countLabel[ind]--;
26          limit++;
27          outlierTraining.Add(cen);
28          inlierTraining.RemoveAt(a);
29      end if
30  end for
31  makeCentroid(cenlist, cen, inlierTraining);
32  END
```

Gambar 4.5 *Pseudocode method* nccorTrain



Gambar 4.6 Diagram alir *method* nccorTest

```
method nccorTest(testingData tsd)
1   BEGIN
2   for (int a = 0; a < tsd.Count; a++)
3       listcalc = new List<calculation>();
4       for (int b = 0; b < cenlist.Count; b++)
5           cl.distance = getNumDist(tsd, a, cenlist, b);
6           cl.label = cenlist[b].label;
7           listcalc.Add(cl);
8       end for
9       listcalc.Sort
10      (delegate (calculation p1, calculation p2)
11                  return p1.distance.CompareTo(p2.distance););
12      ev.index = a;
13      ev.label = tsd[a].label;
14      ev.predicted = listcalc[0].label;
15      eval.Add(ev);
16  end for
17  int match = 0;
18  for (int a = 0; a < eval.Count; a++)
19      if (eval[a].label == eval[a].predicted)
20          match++;
21      end if
22  end for
23  END
```

Gambar 4.7 *Pseudocode method* nccorTest

Prosedur `nccorTrain` menggunakan *method* `makeCentroid` sebagai awal dan akhir proses. Selanjutnya bandingkan *centroid* awal dengan `trainingData`, jika melebihi *threshold radius*, maka masukkan data pada `outlierTraining` dan jika tidak, maka masukkan data pada `inlierTraining`. Buat *centroid* baru setelah semua data telah dibandingkan. *Centroid* baru akan dijadikan *classifier* pada proses selanjutnya yaitu `nccorTest` yang ditunjukkan pada Gambar 4.7. *Method* `nccorTest` adalah *method* yang melakukan operasi perhitungan jarak antar *centroid* dengan `testingData` dan menentukan *centroid* yang terdekat untuk ditetapkan sebagai kelas dan disimpan pada *eval* untuk dibandingkan akurasinya. *Method* `nccorTest` adalah proses akhir dari metode NCCOR.

Kesamaan metode NCCOR dengan KMOR adalah pada penggunaan parameter γ dan n_0. γ digunakan untuk mengatur ukuran ambang batas *outlier* dan n_0 mengatur jumlah *outlier* yang dihilangkan. Jumlah *outlier* yang boleh dihilangkan adalah $0 < n_0 \leq n$, parameter ini mencegah kemungkinan metode NCCOR untuk menganggap sebagian besar atau seluruh titik data sebagai *outlier*. Perhitungan manual metode NCCOR dijelaskan pada subbab 4.2.

4.2 Perhitungan Manual

Pada bagian 4.2 akan dijelaskan perancangan terkait metode yang akan digunakan untuk penelitian ini. Perancangan metode usulan terdiri dari pembentukan *centroid* awal, penghilangan *outlier*, pembaruan *centroid*, dan klasifikasi, dan evaluasi.

4.2.1 Pembentukan *centroid* awal

Pada metode yang diajukan, langkah awal yang akan dilakukan adalah pembentukan *centroid* awal. *Centroid* awal dibentuk dengan menggunakan data yang telah diketahui kelas atau labelnya. Kemudian, cari titik pusat *cluster* atau *centroid* dengan cara menghitung rata-rata nilai fitur untuk seluruh titik data dengan kelas-kelas yang bersangkutan. Proses ini merujuk pada *pseudocode makeCentroid* pada Gambar 3.2. Contoh data yang digunakan adalah potongan *data set Iris* dengan 15 titik data, 4 fitur, dan 3 kelas yang terdiri dari *Iris-setosa* (1), *Iris-versicolor* (2), dan *Iris-virginica* (3). Tabel contoh data akan ditunjukkan pada Tabel 4.1.

Centroid dari data pada Tabel contoh untuk tiap kelas dapat diperoleh dengan menghitung rata-rata terhadap tiap titik data yang bersangkutan. Jika terdapat 3 kelas, maka akan terdapat 3 *centroid*. Tujuan pembentukan *centroid* awal adalah sebagai acuan untuk proses *outlier removal*. *Centroid* awal yang terbentuk akan ditunjukkan oleh Tabel 4.2.

Tabel 4.1 Tabel contoh data

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Kelas
1	0.083	0.458	0.085	0.042	*Iris-setosa*
2	0.194	0.667	0.068	0.042	*Iris-setosa*
3	0.306	0.792	0.119	0.125	*Iris-setosa*
4	0.083	0.583	0.068	0.083	*Iris-setosa*
5	0.194	0.583	0.085	0.042	*Iris-setosa*
6	0.306	0.417	0.593	0.583	*Iris-versicolor*
7	0.472	0.583	0.593	0.625	*Iris-versicolor*
8	0.667	0.458	0.627	0.583	*Iris-versicolor*
9	0.556	0.125	0.576	0.5	*Iris-versicolor*
10	0.361	0.417	0.525	0.5	*Iris-versicolor*
11	0.667	0.542	0.797	1	*Iris-virginica*
12	0.667	0.417	0.712	0.917	*Iris-virginica*
13	0.556	0.208	0.678	0.75	*Iris-virginica*
14	0.611	0.417	0.712	0.792	*Iris-virginica*
15	0.528	0.583	0.746	0.917	*Iris-virginica*

Tabel 4.2 Tabel *centroid* awal

Centroid	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Kelas
1	0.172	0.6166	0.085	0.0668	*Iris-setosa*
2	0.4724	0.4	0.5828	0.5582	*Iris-versicolor*
3	0.6058	0.4334	0.729	0.8752	*Iris-virginica*

Setelah diketahui *centroid* awal, maka proses *outlier removal* dapat dilakukan. *Centroid* awal belum tentu sama dengan *centroid* yang digunakan sebagai *classifier*. Jika terdapat *outlier* dalam data, maka *centroid* akan berubah.

4.2.2 Penghilangan *outlier*

Pada tahapan penghilangan *outlier*, terdapat proses perhitungan nilai d_{avg}. d_{avg} didapatkan dengan cara menghitung rata-rata jarak tiap titik data dengan *centroid* kelas yang sama. Terdapat tiga kelas dalam contoh kasus, maka akan terdapat tiga d_{avg} sesuai kelas yang bersangkutan. Proses ini merujuk pada *pseudocode nccorTrain* pada Gambar 3.3. Contoh perhitungan akan ditunjukkan pada Tabel 4.3.

Tabel 4.3 Contoh perhitungan jarak *centroid* dengan titik data

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	
Centroid 1	0.172	0.6166	0.085	0.0668	**Jarak Euclidean**
Titik Data 3	0.306	0.792	0.119	0.125	
	0.134^2	0.1754^2	0.034^2	0.0582^2	0.230791

Tabel 4.4 Tabel hasil penentuan *outlier*

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Jarak	is > $\gamma * d_{avg}$
C1	0.172	0.6166	0.085	0.0668	d_{avg}	0.124441
1	0.007921	0.025154	0	0.000615	0.183548	Tidak
2	0.000484	0.00254	0.000289	0.000615	0.062675	Tidak
3	0.017956	0.030765	0.001156	0.003387	0.230791	Ya
4	0.007921	0.001129	0.000289	0.000262	0.097987	Tidak
5	0.000484	0.001129	0	0.000615	0.047202	Tidak
C2	0.1664	0.017	0.0102	0.0248	d_{avg}	0.201298
6	0.027689	0.000289	0.000104	0.000615	0.169402	Tidak
7	1.6E-07	0.033489	0.000104	0.004462	0.195078	Tidak
8	0.037869	0.003364	0.001954	0.000615	0.209289	Ya
9	0.006989	0.075625	4.62E-05	0.003387	0.293338	Ya
10	0.01241	0.000289	0.003341	0.003387	0.139381	Tidak
C3	0.6058	0.4334	0.729	0.8752	d_{avg}	0.159108
11	0.003745	0.011794	0.004624	0.015575	0.189046	Ya
12	0.003745	0.000269	0.000289	0.001747	0.077786	Tidak
13	0.00248	0.050805	0.002601	0.015675	0.267509	Ya
14	2.7E-05	0.000269	0.000289	0.006922	0.086644	Tidak
15	0.006053	0.02238	0.000289	0.001747	0.174554	Ya

Lakukan langkah pada Tabel 4.3 untuk setiap titik data yang ada. Gunakan parameter γ untuk menentukan titik data yang bersangkutan sebagai *outlier* atau tidak. Jika nilai $\gamma = 1$, *i* adalah banyak titik data, dan *c* adalah banyak kelas maka $jarak(titikdata_i, centroid_c) > 1 * d_{avg}$ adalah *outlier*. Proses ini merujuk pada *pseudocode nccorTrain* pada Gambar 3.3. Proses penentuan *outlier* akan ditunjukkan Tabel 4.4. Setelah proses penentuan *outlier* dilakukan maka hapus semua titik data yang diidentifikasi sebagai *outlier*. Sisa titik data yang akan diproses selanjutnya akan ditunjukkan Tabel 4.5.

Tabel 4.5 Titik data yang terpilih

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Kelas
1	0.083	0.458	0.085	0.042	*Iris-setosa*
2	0.194	0.667	0.068	0.042	*Iris-setosa*
4	0.083	0.583	0.068	0.083	*Iris-setosa*
5	0.194	0.583	0.085	0.042	*Iris-setosa*
6	0.306	0.417	0.593	0.583	*Iris-versicolor*
7	0.472	0.583	0.593	0.625	*Iris-versicolor*
10	0.361	0.417	0.525	0.5	*Iris-versicolor*
12	0.667	0.417	0.712	0.917	*Iris-virginica*
14	0.611	0.417	0.712	0.792	*Iris-virginica*

4.2.3 Pembaruan *centroid*

Lakukan proses penghitungan *centroid* ulang dengan titik data yang tersisa. *Centroid* baru akan ditunjukkan Tabel 4.6.

Tabel 4.6 *Centroid* baru

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Kelas
1	0.1385	0.57275	0.0765	0.05225	*Iris-setosa*
2	0.379667	0.472333	0.570333	0.569333	*Iris-versicolor*
3	0.639	0.417	0.712	0.8545	*Iris-virginica*

4.2.4 Klasifikasi

Setelah *centroid* baru sebagai klasifier telah diketahui, maka proses klasifikasi dapat dilakukan. Klasifikasi dilakukan dengan membandingkan jarak antar titik data uji dengan tiap *centroid*. *Centroid* dengan jarak terdekat akan menjadi patokan untuk penentuan kelas. Data uji yang digunakan sebagai contoh akan ditunjukkan oleh Tabel 4.7. Proses ini merujuk pada *pseudocode nccorTest* pada Gambar 3.4.

Tabel 4.7 Contoh data uji

No	Fitur 1	Fitur 2	Fitur 3	Fitur 4	Kelas Aktual
1	0.306	0.583	0.119	0.042	*Iris-setosa*
2	0.444	0.5	0.644	0.908	*Iris-versicolor*
3	0.944	0.417	0.864	0.917	*Iris-virginica*

Kemudian lakukan perhitungan jarak antar titik data dengan tiap *centroid* lalu bandingkan hasilnya. Perbandingan jarak antar titik data dengan *centroid* akan ditunjukkan Tabel 4.8.

Tabel 4.8 Perbandingan jarak antar titik data dengan *centroid*

No data uji	Jarak *centroid* 1	Jarak *centroid* 2	Jarak *centroid* 3	Kelas Terprediksi	Kelas Aktual
1	0.173415	0.706722	1.072497	Iris-setosa	Iris-setosa
2	1.073772	0.353591	0.228911	Iris-virginica	Iris-versicolor
3	1.428649	0.727081	0.346461	Iris-virginica	Iris-virginica

4.2.5 Evaluasi

Setelah proses klasifikasi selesai dilakukan, maka hasil yang telah didapatkan akan dievaluasi untuk mengetahui tingkat keakuratan metode klasifikasi. Metode yang digunakan untuk evaluasi adalah akurasi. Untuk mengetahui akurasi dibutuhkan *confusion matrix* terlebih dahulu. *Confusion matrix* dapat dibentuk berdasarkan kelas terprediksi dan kelas aktual yang didapat pada proses klasifikasi. Akurasi didapatkan dengan menjumlahkan hasil yang benar dengan data keseluruhan. *Confusion matrix* beserta akurasi akan ditunjukkan oleh Tabel 4.9.

Tabel 4.9 Confusion matrix

		Truth data			*Classification overall*	*Producer accuracy (precision)*
		setosa	*versicolor*	*virginica*		
Classifier results	*setosa*	1	0	0	1	100
	versicolor	0	0	0	0	0
	virginica	0	1	1	2	50
Truth overall		1	1	1	3	
User accuracy (recall)		100	0	100	*Overall accuracy*	**(1+0+1)/3 = 66.667**

4.3 Perancangan pengujian dan evaluasi

Pada bagian ini akan dijelaskan skenario pengujian untuk setiap metode. Proses pengujian penting dilakukan untuk mengetahui metode terbaik untuk data set yang diujikan. Proses perancangan pengujian mencakup pengujian parameter metode yang diajukan NCCOR dan perbandingan dengan metode terdahulu NCC. Selain itu dilakukan perbandingan dengan metode-metode pembanding, antara lain: *k*-NN dan NB yang telah dilakukan outlier removal terlebih dahulu

4.3.1 Pengujian metode NCC

Pengujian metode NCC diujikan pada data set yang disiapkan. Tabel yang digunakan bertujuan untuk merekam hasil *F-measure*, akurasi, dan waktu komputasi. Tabel pengujian metode NCC ditunjukkan pada Tabel 4.10.

Tabel 4.10 Tabel Pengujian metode NCC

Data set	*F-measure*	Akurasi	Waktu Komputasi (ms)
Data set 1			
Data set 2			
...			
Data set-n			

Hasil yang didapat pada Tabel 4.10 akan dijadikan landasan sebagai perbandingan dengan metode usulan NCCOR dari segi *F-measure* dan waktu komputasi untuk mengetahui peningkatan performa pendekatan *outlier removal* pada metode NCC.

4.3.2 Pengujian metode usulan NCCOR

Pengujian dilakukan untuk mengetahui performa metode usulan dalam klasifikasi data yang memiliki *outlier*. Skenario yang akan dilakukan adalah mengukur akurasi dan waktu komputasi metode usulan dengan metode pembanding terhadap *data set* uji. Metode NCCOR akan diujikan dengan parameter γ = {0.5, 1, 1.5, 2, 2.5, 3} dan n_0 = {0.2, 0.4, 0.6, 0.8} . Pada waktu komputasi, akan diujikan dengan satuan milidetik untuk setiap metode. Bentuk table yang akan digunakan dalam mencatat hasil pengujian akan ditunjukkan pada Tabel 4.11 sebagai Tabel Hasil Pengujian metode NCCOR.

Tabel 4.11 Tabel Hasil Pengujian NCCOR

Nama Data set					
γ	n_0	Data points Removed	*F-measure*	Akurasi	Waktu Komputasi (ms)
1	0.1				
1	0.2				
1	0.3				
...	...				
3	1				

4.3.3 Pengujian metode pembanding dengan outlier removal

Metode *Naïve Bayes* adalah salah satu metode yang menggunakan prinsip probabilistic atau kemungkinan. Pada pengujian metode *Naïve Bayes*, tidak terdapat setting parameter. Tabel Hasil pengujian metode *Naïve Bayes* sama seperti metode NCC yaitu hasil klasifikasi *F-measure*, akurasi, dan waktu komputasi untuk setiap data set. Tabel hasil pengujian metode *Naïve bayes* ditunjukkan Tabel 4.12.

Tabel 4.12 Tabel pengujian metode *Naïve Bayes*

Data set	*F-measure*	Akurasi	Waktu Komputasi (ms)
Data set 1			
Data set 2			
...			
Data set-n			

Berbeda dengan metode *Naïve Bayes,* metode *k-Nearest Neighbor (k*-NN*)* adalah salah satu metode yang menggunakan jarak dalam menentukan kedekatan antar titik data. Pada pengujian metode Naïve Bayes, terdapat setting parameter *k*. Pada pengujian akurasi, metode *k*-NN, akan diujikan dengan parameter k = {1, 2, 3, …, n} dengan n adalah banyaknya data latih terdekat dan diambil hasil terbaik. Tabel hasil pengujian metode *k*-NN ditunjukkan Tabel 4.13.

Tabel 4.13 Tabel Hasil Pengujian *k*-NN

Nama Dataset			
k	*F-measure*	Akurasi	Waktu Komputasi (ms)
1			
2			
…			
n			

4.3.4 Perbandingan hasil pengujian tiap metode

Dalam usaha analisis hasil pengujian, dibutuhkan perbandingan hasil klasifikasi tiap metode. Dalam Tabel 4.14, Tabel 4.15, dan 4.16 hasil terbaik dari setiap metode akan ditampilkan untuk dapat dibandingkan hasilnya. Tabel perbandingan akurasi dan *F-measure* akan ditampilkan bersama dengan grafik. Sedangkan Tabel perbandingan waktu komputasi akan ditampilkan dengan keterangan data set yang tersedia.

Tabel 4.14 Tabel perbandingan hasil Akurasi

Data set	Sparsity	Akurasi			
		NCCOR	NCC	*k*-NNOR	NBOR
Data set 1					
Data set 2					
…					
Data set-n					

Tabel 4.15 Tabel perbandingan hasil *F-measure*

Data set	Titik Data	*F-measure*			
		NCCOR	NCC	*k*-NNOR	NBOR
Data set 1					
Data set 2					
…					
Data set-n					

Tabel 4.16 Tabel Hasil Perbandingan Waktu Komputasi

Data set	Titik Data	Fitur	Waktu Komputasi (ms)			
			NCCOR	NCC	*k*-NNOR	NBOR
Data set 1						
Data set 2						
…						
Data set-n						

BAB 5 HASIL DAN PEMBAHASAN

5.1 Hasil pengujian

Pada subbab 5.1 adalah paparan hasil pengujian klasifikasi untuk semua metode pada semua data set yang diujikan. Hasil pengujian meliputi akurasi dan waktu komputasi.

5.1.1 Hasil pengujian metode Nearest *Centroid Classifier*

Pengujian metode NCC bertujuan mengetahui performa metode asal sebelum *outlier removal*. Hasil pengujian metode NCC ditunjukkan Tabel 5.1.

Tabel 5.1 Hasil pengujian metode Nearest *Centroid Classifier*

Data set	*F-measure*	Akurasi	Waktu (ms)
Breast cancer	93.56	94.29	0.939
Ecoli	63.57	84.16	0.7664
Lymphography	78.17	80	0.4623
Vertebral	71.99	73.12	0.3922
Wine	**95.85**	**96.3**	0.5065
Yeast	52.64	47.76	5.2945

Tabel 5.1 menunjukkan bahwa metode NCC memiliki performa yang berbeda untuk setiap data set. Akurasi tertinggi diraih pada klasifikasi data set *Wine* yaitu 96.3 dan *F-measure* sebesar 95.85, sedangkan akurasi terendah diraih pada saat klasifikasi data set *Yeast*, yaitu sebesar 47.76 dan *F-measure* sebesar 52.64. Untuk waktu komputasi, metode NCC melakukan klasifikasi dengan waktu yang relatif cepat. Rentang capaian waktu komputasi metode NCC antara 0.3922 hingga 5.2945 ms.

5.1.2 Hasil pengujian metode pembanding dengan Outlier Removal

Metode pembanding yang diujikan pertama adalah metode Naïve Bayes dengan perlakuan outlier removal sebelumnya. Pada Tabel 5.2 didapatkan hasil berupa akurasi dan waktu komputasi klasifikasi dengan metode *Naïve Bayes*. Untuk akurasi, *Naïve Bayes* dapat melakukan performa terbaik pada data set *Wine* sebesar 98.15 dan *F-measure* sebesar 97.78, diikuti dengan akurasi 92.29 dan *F-measure* 95.37 pada data set *Breast cancer*. Namun metode *Naïve Bayes* melakukan performa klasifikasi yang buruk pada data set *Ecoli* (akurasi 13.86 dan *F-measure* 3.04), *Lymphography* (*F-measure* 13.67 dan akurasi 37.78), dan *Yeast* (Akurasi 31.17 dan *F-measure* 4.75). Waktu komputasi yang dicapai metode NB antara 0.5 hingga 13.8597 ms.

Tabel 5.2 Hasil pengujian metode *Naïve Bayes*

Data set	*F-measure*	Akurasi	Waktu (ms)
Breast cancer	92.29	92.38	3.1052
Ecoli	3.41	15.84	5.2785
Lymphography	14.80	42.22	2.7731
Vertebral	75.68	79.57	1.1342
Wine	97.78	98.15	1.2952
Yeast	0.09	0.45	27.06

Untuk metode pembanding kedua yang diujikan adalah metode k-NN dengan perlakuan *outlier removal* sebelumnya. Performa metode *k*-NN cukup bervariasi. Performa terbaik didapat pada klasifikasi data set *Breast Cancer* dengan nilai 97.62 *F-measure* dan 97.31 Akurasi pada k = 3 disusul dengan data set *Wine* dengan akurasi 96.30 dan *F-measure* 95.95 pada k = 3. Performa klasifikasi terburuk didapat pada data set *Yeast* dengan akurasi 58.74 dan *F-measure* 60.26. Untuk pengukuran waktu komputasi, metode k-*Nearest Neighbor* jauh lebih lama dibandingkan metode lainnya. Waktu terlama pada saat memproses data set *Yeast* dengan waktu 270.67 ms, sedangkan waktu tercepat diperolah pada saat memroses data set *Lymphography* sebesar 4 ms. Tabel 5.3, Tabel 5.4, dan Tabel 5.5 akan menunjukkan hasil klasifikasi metode k-*Nearest Neighbor*. Pengujian dilakukan dengan parameter k = 1 hingga k = 15.

Tabel 5.3 Hasil klasifikasi metode *k*-NN pada data set *Breast* dan *Ecoli*

	Breast Cancer			*Ecoli*		
k	*F-measure*	Akurasi	Waktu(ms)	*F-measure*	Akurasi	Waktu(ms)
1	97.31	97.62	47.7588	60.57	84.16	9.0467
2	97.31	97.62	46.7263	60.57	84.16	9.6018
3	**97.31**	**97.62**	**56.3311**	60.50	87.13	10.5633
4	97.31	97.62	51.7081	62.85	89.11	11.0679
5	96.76	97.14	54.752	59.45	86.14	10.6364
6	96.76	97.14	61.1982	60.50	87.13	12.7386
7	96.76	97.14	59.9716	59.45	86.14	13.0758
8	96.76	97.14	61.3622	58.98	85.15	13.4146
9	96.22	96.67	73.0041	60.14	86.14	14.5573
10	96.22	96.67	69.2203	50.82	84.16	14.5187
11	96.22	96.67	70.0591	51.54	86.14	16.932
12	96.22	96.67	76.3472	51.54	86.14	14.7269
13	96.22	96.67	86.8985	51.54	86.14	16.4825
14	96.22	96.67	79.8377	52.44	86.14	17.5744
15	96.22	96.67	77.9969	52.44	86.14	19.2685

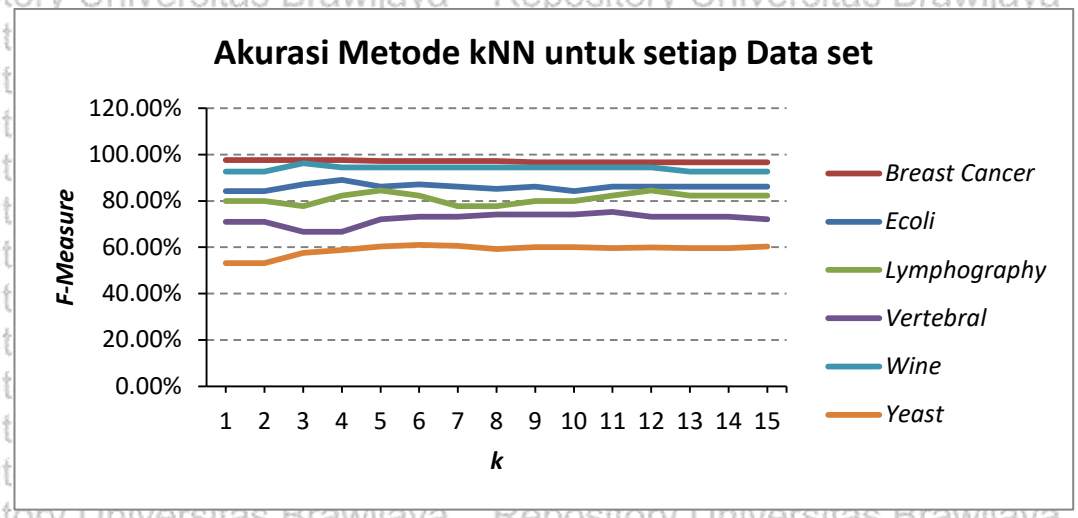
Tabel 5.4 Hasil klasifikasi metode *k*-NN data set *Lymphography* dan *Vertebral*

k	*Lymphography*			*Vertebral*		
	F-measure	**Akurasi**	**Waktu(ms)**	***F-measure***	**Akurasi**	**Waktu(ms)**
1	57.04	80.00	4.9662	68.29	70.97	7.4635
2	57.04	80.00	5.7222	68.29	70.97	8.5654
3	39.74	77.78	5.4779	62.73	66.67	8.386
4	63.48	82.22	4.0186	64.90	66.67	10.025
5	42.52	84.44	3.8908	70.49	72.04	9.831
6	41.33	82.22	3.933	72.91	73.12	10.8144
7	39.04	77.78	4.1047	72.08	73.12	10.5059
8	39.04	77.78	4.1614	73.38	74.19	10.8297
9	40.30	80.00	4.8251	73.56	74.19	11.5604
10	40.15	80.00	4.5256	73.21	74.19	11.4201
11	41.38	82.22	4.6309	73.80	75.27	12.3955
12	42.62	84.44	4.9745	71.31	73.12	12.8088
13	41.38	82.22	4.6859	70.86	73.12	12.9624
14	41.58	82.22	6.5819	70.62	73.12	15.0259
15	41.38	82.22	4.9225	68.70	72.04	17.0724

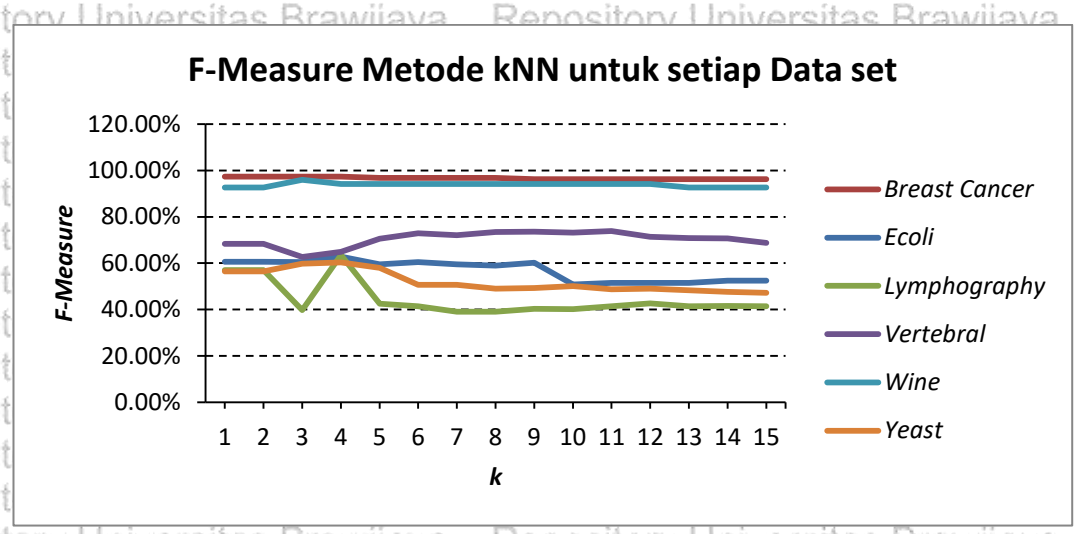
Tabel 5.5 Hasil klasifikasi metode *k*-NN pada data set *Wine* dan *Yeast*

k	*Wine*			*Yeast*		
	F-measure	**Akurasi**	**Waktu(ms)**	***F-measure***	**Akurasi**	**Waktu(ms)**
1	92.69	92.59	5.0007	56.40	53.14	237.0105
2	92.69	92.59	5.7088	56.40	53.14	265.2544
3	95.95	96.30	5.0871	59.81	57.62	275.8495
4	94.23	94.44	5.376	60.26	58.74	270.6741
5	94.23	94.44	6.0921	57.99	60.31	252.0318
6	94.23	94.44	5.5214	50.61	60.99	255.8692
7	94.23	94.44	5.884	50.59	60.54	265.8864
8	94.23	94.44	5.8598	48.99	59.19	268.7219
9	94.23	94.44	5.9863	49.28	60.09	291.0682
10	94.23	94.44	6.3551	50.15	60.09	310.951
11	94.23	94.44	6.3478	48.71	59.64	308.652
12	94.23	94.44	6.2876	48.94	59.87	393.8348
13	92.69	92.59	6.3864	48.33	59.64	335.54
14	92.69	92.59	6.8078	47.61	59.64	356.6562
15	92.69	92.59	7.3978	47.22	60.31	417.8418

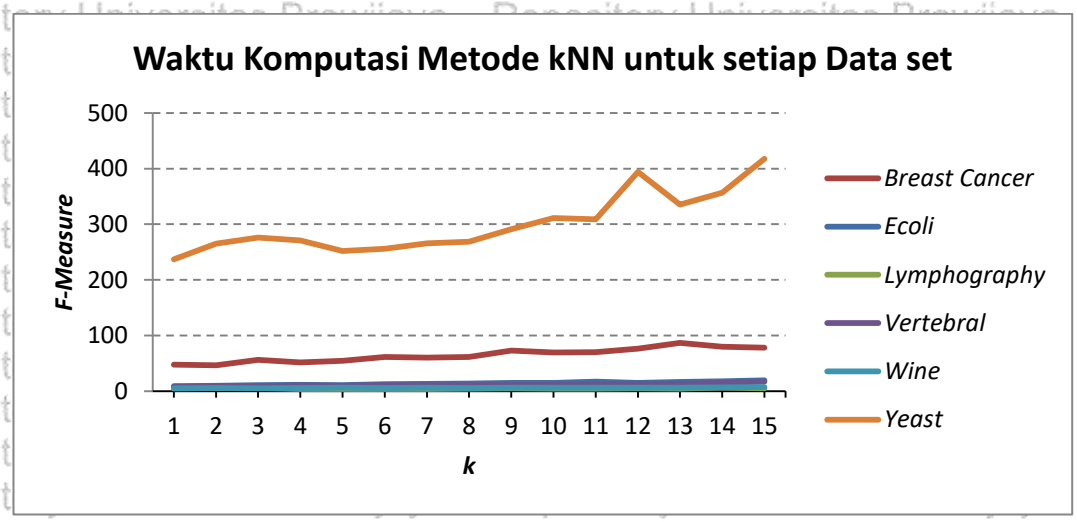
Untuk memperjelas hasil pengujian, Gambar 5.1 menggambarkan grafik Akurasi metode *k*-NN untuk setiap data set, Gambar 5.2 menggambarkan grafik *F-measure* untuk setiap data set, dan Gambar 5.3 menggambarkan grafik waktu komputasi untuk setiap data set.



Gambar 5.1 Akurasi metode *k*-NN untuk setiap Data set



Gambar 5.2 *F-measure* metode *k*-NN untuk setiap Data set



Gambar 5.3 Waktu komputasi metode *k*-NN untuk setiap Data set

5.1.3 Hasil pengujian metode Nearest *Centroid Classifier* with *Outlier Removal*

Pada subbab 5.1.4 ini akan dipaparkan hasil pengujian metode Nearest *Centroid Classifier* with *Outlier Removal* sebagai metode usulan.

Metode NCCOR dapat mencapai akurasi 96.19 hingga 97.14 pada data set *Breast cancer*, dengan konfigurasi terbaik n_0 = 0.1 dan γ = 1.5, demikian dengan *F-measure* dapat mencapai nilai 95.71 hingga 96.81 pada parameter yang sama. Untuk waktu komputasi berada pada rentang 2.1 hingga 5.1 ms.

Pada data set *Ecoli*, metode NCCOR dapat mencapai akurasi 83.17 hingga 86.14 dengan konfigurasi terbaik n_0 = 0.2 dan γ = 1, demikian dengan *F-measure* dapat mencapai 62.38 hingga 64.58 pada parameter yang sama. Untuk waktu komputasi mencapai 2.4 hingga 4.4 ms. Tabel 5.6 memaparkan hasil klasifikasi metode NCCOR pada data set *Breast Cancer* dan *Ecoli*.

Metode NCCOR dapat mencapai akurasi 80 hingga 84.44 dengan konfigurasi terbaik n_0 = 0.5 dan γ = 1.5 pada data set *Lymphography*. *F-measure* dapat mencapai 73.68 hingga 80.15 pada konfigurasi n_0 = 0.5 dan γ = 1. Untuk waktu komputasi mencapai 1.4 hingga 2.4 ms.

Saat mengklasifikasi data set *Vertebral*, metode NCCOR dapat mencapai akurasi 72.04 hingga 77.42 dengan konfigurasi terbaik n_0 = 0.5 dan γ = 1.5, demikian dengan *F-measure* dapat mencapai 69.05 hingga 76.02 pada konfigurasi n_0 = 0.4 dan γ = 1. Waktu komputasi mencapai sekitar 0.94 hingga 1.8 ms. Tabel 5.7 memaparkan hasil klasifikasi metode NCCOR pada data set *Lymphography* dan *Vertebral*.

Akurasi yang dicapai metode NCCOR pada saat klasifikasi *data set Wine* adalah 96.3 hingga 98.15 dengan konfigurasi terbaik n_0 = 0.2 dan γ = 1, demikian dengan *F-measure* dapat mencapai 95.98 hingga 97.76 pada konfigurasi yang sama. Untuk waktu komputasi mencapai sekitar 1 hingga 1.7 ms.

Pada data set *Yeast*, metode NCCOR dapat mencapai akurasi 51.35 hingga 52.91 dengan konfigurasi terbaik n_0 = 0.2 dan γ = 1, demikian dengan *F-measure* dapat mencapai 58.92 pada n_0 = 0.2 dan γ = 1. Untuk waktu komputasi berada pada rentang 16.4 hingga 27.8 ms. Tabel 5.8 memaparkan hasil klasifikasi metode NCCOR pada data set *Wine* dan *Yeast*.

Performa terbaik metode NCCOR terbaik didapat pada klasifikasi data set *Wine* (97.76 *F-measure* dan 98.15 Akurasi) disusul dengan data set *Breast Cancer* (akurasi 97.14 dan *F-measure* 97.14). Performa klasifikasi terburuk didapat pada data set *Yeast* dengan akurasi 52.47 dan *F-measure* 58.92. Secara pengukuran akurasi dan *F-measure* metode NCCOR lebih baik dibanding metode-metode pembanding. Untuk pengukuran waktu komputasi, metode NCCOR tidak berbeda jauh dengan metode NCC dan jauh lebih baik dibanding metode *k*-NN.

Tabel 5.6 Hasil klasifikasi metode NCCOR pada data set *Breast Cancer* dan *Ecoli*

Breast Cancer						*Ecoli*			
γ	n_0	*Removal*	*F-measure*	Akurasi	Waktu(ms)	*Removal*	*F-measure*	Akurasi	Waktu (ms)
1	0.1	49	96.26	96.67	2.209	24	62.54	83.17	3.1075
1	**0.2**	**98**	**95.71**	**96.19**	3.8337	**48**	**64.58**	**86.14**	2.4859
1	0.3	147	96.26	96.67	2.2651	71	63.87	84.16	2.5175
1	0.4	180	96.26	96.67	3.2267	89	63.87	84.16	2.6211
1	0.5	180	96.26	96.67	2.2448	89	63.87	84.16	2.6009
1	0.6	180	96.26	96.67	2.2717	89	63.87	84.16	2.5343
1	0.7	180	96.26	96.67	2.4283	89	63.87	84.16	2.6523
1	0.8	180	96.26	96.67	2.2441	89	63.87	84.16	2.5379
1	0.9	180	96.26	96.67	2.2799	89	63.87	84.16	2.5167
1	1	180	96.26	96.67	2.3507	89	63.87	84.16	2.7135
1.5	0.1	37	96.81	97.14	2.9875	**24**	**62.38**	**84.16**	3.3937
1.5	0.2	37	96.81	97.14	5.161	26	62.38	84.16	2.8738
1.5	0.3	37	96.81	97.14	2.5206	26	62.38	84.16	2.5262
1.5	0.4	37	96.81	97.14	2.3803	26	62.38	84.16	2.6672
1.5	0.5	37	96.81	97.14	2.3441	26	62.38	84.16	2.5682
1.5	0.6	37	96.81	97.14	2.38	26	62.38	84.16	2.5449
1.5	0.7	37	96.81	97.14	3.5285	26	62.38	84.16	3.8879
1.5	0.8	37	96.81	97.14	2.344	26	62.38	84.16	2.511
1.5	0.9	37	96.81	97.14	2.3893	26	62.38	84.16	4.4116
1.5	1	37	96.81	97.14	2.2314	26	62.38	84.16	2.4995
2	0.1	22	96.81	97.14	2.441	12	63.36	85.15	2.5153
2	0.2	22	96.81	97.14	2.2307	12	63.36	85.15	2.4938
2	0.3	22	96.81	97.14	2.2471	12	63.36	85.15	3.7315
2	0.4	22	96.81	97.14	2.329	12	63.36	85.15	3.3787
2	0.5	22	96.81	97.14	2.2263	12	63.36	85.15	3.8929
2	0.6	22	96.81	97.14	2.3009	12	63.36	85.15	2.5181
2	0.7	22	96.81	97.14	2.581	12	63.36	85.15	2.5233
2	0.8	22	96.81	97.14	3.6091	12	63.36	85.15	2.5318
2	0.9	22	96.81	97.14	2.3049	12	63.36	85.15	2.6735
2	1	22	96.81	97.14	2.2931	12	63.36	85.15	2.4984
2.5	0.1	15	96.81	97.14	2.5228	5	62.82	84.16	2.4666
2.5	0.2	15	96.81	97.14	2.425	5	62.82	84.16	2.5206
2.5	0.3	15	96.81	97.14	2.5041	5	62.82	84.16	2.643
2.5	0.4	15	96.81	97.14	2.5979	5	62.82	84.16	2.5813
2.5	0.5	15	96.81	97.14	2.9553	5	62.82	84.16	2.686
2.5	0.6	15	96.81	97.14	2.2818	5	62.82	84.16	2.4652
2.5	0.7	15	96.81	97.14	2.6494	5	62.82	84.16	2.8807
2.5	0.8	15	96.81	97.14	2.2502	5	62.82	84.16	3.5941
2.5	0.9	15	96.81	97.14	2.4551	5	62.82	84.16	2.8443
2.5	1	15	96.81	97.14	2.8729	5	62.82	84.16	2.8185
3	0.1	11	96.26	96.67	2.2493	3	63.5	85.15	2.5661
3	0.2	11	96.26	96.67	2.2195	3	63.5	85.15	3.5498
3	0.3	11	96.26	96.67	3.5047	3	63.5	85.15	2.4318
3	0.4	11	96.26	96.67	3.2845	3	63.5	85.15	2.4588
3	0.5	11	96.26	96.67	2.7693	3	63.5	85.15	3.0953
3	0.6	11	96.26	96.67	2.5145	3	63.5	85.15	2.4253
3	0.7	11	96.26	96.67	2.207	3	63.5	85.15	3.1815
3	0.8	11	96.26	96.67	2.1834	3	63.5	85.15	3.3327
3	0.9	11	96.26	96.67	2.3814	3	63.5	85.15	2.6233
3	1	11	96.26	96.67	2.4683	3	63.5	85.15	2.4808

Tabel 5.7 Hasil klasifikasi metode NCCOR data set *Lymphography* dan *Vertebral*

		Lymphography				*Vertebral*			
γ	n_0	*Removal*	*F-measure*	Akurasi	Waktu (ms)	*Removal*	*F-measure*	Akurasi	Waktu (ms)
1	0.1	11	75.5	84.44	1.6928	22	69.92	72.04	1.5593
1	0.2	21	74.71	82.22	1.6029	44	71.32	73.12	0.9794
1	**0.3**	**31**	**73.68**	**80**	1.5101	66	72.24	74.19	1.0447
1	0.4	42	73.68	80	2.1654	**87**	**76.02**	**77.42**	0.9967
1	**0.5**	**46**	**80.15**	**82.22**	1.5598	99	75.83	77.42	1.0295
1	0.6	46	80.15	82.22	1.7423	99	75.83	77.42	0.967
1	0.7	46	80.15	82.22	1.6184	99	75.83	77.42	1.0171
1	0.8	46	80.15	82.22	1.9516	99	75.83	77.42	1.2314
1	0.9	46	80.15	82.22	1.5457	99	75.83	77.42	1.5136
1	1	46	80.15	82.22	1.5067	99	75.83	77.42	1.0245
1.5	0.1	4	76.36	84.44	1.5141	22	72.24	74.19	1.5729
1.5	0.2	4	76.36	84.44	2.4984	31	69.5	72.04	0.9952
1.5	0.3	4	76.36	84.44	1.5745	31	69.5	72.04	1.3798
1.5	0.4	4	76.36	84.44	2.0524	31	69.5	72.04	0.9911
1.5	0.5	4	76.36	84.44	1.5363	31	69.5	72.04	0.9711
1.5	0.6	4	76.36	84.44	1.5055	31	69.5	72.04	1.0507
1.5	0.7	4	76.36	84.44	1.4928	31	69.5	72.04	0.9681
1.5	0.8	4	76.36	84.44	1.5922	31	69.5	72.04	1.1198
1.5	0.9	4	76.36	84.44	1.5292	31	69.5	72.04	0.971
1.5	1	4	76.36	84.44	1.5228	31	69.5	72.04	1.3509
2	0.1	0	76.06	84.44	1.5195	7	73.19	75.27	0.9702
2	0.2	0	76.06	84.44	1.9175	7	73.19	75.27	0.9895
2	0.3	0	76.06	84.44	1.5067	7	73.19	75.27	0.98
2	0.4	0	76.06	84.44	1.65	7	73.19	75.27	1.0566
2	0.5	0	76.06	84.44	2.017	7	73.19	75.27	1.0172
2	0.6	0	76.06	84.44	1.5238	7	73.19	75.27	1.1515
2	0.7	0	76.06	84.44	2.1677	7	73.19	75.27	1.7129
2	0.8	0	76.06	84.44	1.5454	7	73.19	75.27	1.0161
2	0.9	0	76.06	84.44	1.506	7	73.19	75.27	0.9873
2	1	0	76.06	84.44	1.7109	7	73.19	75.27	0.9849
2.5	0.1	0	76.06	84.44	1.5322	0	71.32	73.12	0.9796
2.5	0.2	0	76.06	84.44	1.5075	0	71.32	73.12	0.9841
2.5	0.3	0	76.06	84.44	1.519	0	71.32	73.12	1.2144
2.5	0.4	0	76.06	84.44	1.5566	0	71.32	73.12	1.1114
2.5	0.5	0	76.06	84.44	1.5467	0	71.32	73.12	1.1763
2.5	0.6	0	76.06	84.44	1.954	0	71.32	73.12	1.1668
2.5	0.7	0	76.06	84.44	1.5351	0	71.32	73.12	1.0042
2.5	0.8	0	76.06	84.44	1.5252	0	71.32	73.12	0.9784
2.5	0.9	0	76.06	84.44	1.528	0	71.32	73.12	1.133
2.5	1	0	76.06	84.44	1.686	0	71.32	73.12	1.1708
3	0.1	0	76.06	84.44	1.496	0	71.32	73.12	0.9751
3	0.2	0	76.06	84.44	1.6148	0	71.32	73.12	1.0395
3	0.3	0	76.06	84.44	2.0885	0	71.32	73.12	0.9599
3	0.4	0	76.06	84.44	1.5454	0	71.32	73.12	0.957
3	0.5	0	76.06	84.44	2.29	0	71.32	73.12	1.336
3	0.6	0	76.06	84.44	2.3281	0	71.32	73.12	1.8002
3	0.7	0	76.06	84.44	1.5893	0	71.32	73.12	1.5291
3	0.8	0	76.06	84.44	1.4757	0	71.32	73.12	1.1583
3	0.9	0	76.06	84.44	1.4162	0	71.32	73.12	1.0018
3	1	0	76.06	84.44	1.4485	0	71.32	73.12	0.9457

Tabel 5.8 Hasil klasifikasi metode NCCOR pada data set *Wine* dan *Yeast*

		Wine				*Yeast*			
γ	n_0	*Removal*	*F-measure*	Akurasi	Waktu (ms)	*Removal*	*F-measure*	Akurasi	Waktu (ms)
1	0.1	13	95.98	96.3	1.144	104	56.89	51.79	23.871
1	0.2	25	97.76	98.15	1.4287	208	56.59	52.02	21.335
1	0.3	38	97.76	98.15	1.1172	**312**	**58.92**	**52.47**	18.86
1	0.4	50	95.98	96.3	1.4704	416	57.48	52.91	17.667
1	0.5	54	95.98	96.3	1.1163	421	57.5	52.91	16.964
1	**0.6**	**54**	**95.98**	**96.3**	1.1043	421	57.5	52.91	18.245
1	0.7	54	95.98	96.3	1.1114	421	57.5	52.91	18.061
1	0.8	54	95.98	96.3	1.1642	421	57.5	52.91	17.911
1	0.9	54	95.98	96.3	1.1298	421	57.5	52.91	18.813
1	1	54	95.98	96.3	1.3131	421	57.5	52.91	19.611
1.5	0.1	7	95.98	96.3	1.1187	104	55.2	51.35	24.297
1.5	0.2	7	95.98	96.3	1.1756	108	54.74	51.35	17.394
1.5	0.3	7	95.98	96.3	1.1357	108	54.74	51.35	22.126
1.5	0.4	7	95.98	96.3	1.5798	108	54.74	51.35	17.441
1.5	0.5	7	95.98	96.3	1.113	108	54.74	51.35	18.032
1.5	0.6	7	95.98	96.3	1.1132	108	54.74	51.35	17.34
1.5	0.7	7	95.98	96.3	1.1288	108	54.74	51.35	19.26
1.5	0.8	7	95.98	96.3	1.1016	108	54.74	51.35	18.431
1.5	0.9	7	95.98	96.3	1.1023	108	54.74	51.35	17.665
1.5	1	7	95.98	96.3	1.6253	108	54.74	51.35	17.519
2	0.1	0	95.98	96.3	1.4424	37	55.55	51.57	23.341
2	0.2	0	95.98	96.3	1.2864	37	55.55	51.57	19.028
2	0.3	0	95.98	96.3	1.1185	37	55.55	51.57	17.421
2	0.4	0	95.98	96.3	1.2621	37	55.55	51.57	18.954
2	0.5	0	95.98	96.3	1.7201	37	55.55	51.57	16.873
2	0.6	0	95.98	96.3	1.094	37	55.55	51.57	17.398
2	0.7	0	95.98	96.3	1.1082	37	55.55	51.57	18.186
2	0.8	0	95.98	96.3	1.1503	37	55.55	51.57	17.894
2	0.9	0	95.98	96.3	1.6866	37	55.55	51.57	23.813
2	1	0	95.98	96.3	1.7682	37	55.55	51.57	17.315
2.5	0.1	0	95.98	96.3	1.0894	**20**	**54.88**	**51.79**	17.464
2.5	0.2	0	95.98	96.3	1.3189	20	54.88	51.79	18.708
2.5	0.3	0	95.98	96.3	1.0782	20	54.88	51.79	19.553
2.5	0.4	0	95.98	96.3	1.1391	20	54.88	51.79	19.178
2.5	0.5	0	95.98	96.3	1.7603	20	54.88	51.79	18.973
2.5	0.6	0	95.98	96.3	1.0992	20	54.88	51.79	19.51
2.5	0.7	0	95.98	96.3	1.1094	20	54.88	51.79	18.236
2.5	0.8	0	95.98	96.3	1.1487	20	54.88	51.79	17.254
2.5	0.9	0	95.98	96.3	1.138	20	54.88	51.79	27.444
2.5	1	0	95.98	96.3	1.2867	20	54.88	51.79	18.09
3	0.1	0	95.98	96.3	1.1364	13	54.99	51.57	18.028
3	0.2	0	95.98	96.3	1.1043	13	54.99	51.57	19.29
3	0.3	0	95.98	96.3	1.2523	13	54.99	51.57	18.386
3	0.4	0	95.98	96.3	1.0686	13	54.99	51.57	19.381
3	0.5	0	95.98	96.3	1.078	13	54.99	51.57	17.247
3	0.6	0	95.98	96.3	1.084	13	54.99	51.57	17.777
3	0.7	0	95.98	96.3	1.212	13	54.99	51.57	18.492
3	0.8	0	95.98	96.3	1.2698	13	54.99	51.57	17.936
3	0.9	0	95.98	96.3	1.1431	13	54.99	51.57	19.331
3	1	0	95.98	96.3	1.1043	13	54.99	51.57	17.889

Tabel 5.9 Hasil pengujian metode NCCOR dengan konfigurasi terbaik

Data sets	γ	n_0	*F-measure*	Akurasi	Waktu (ms)
Breast cancer	1.5	0.1	96.81	97.14	2.9875
Ecoli	1	0.2	64.58	86.14	2.4859
Lymphography	1.5	0.5	80.15	84.44	1.5363
Vertebral	1.5	0.5	76.02	77.42	0.9711
Wine	1	0.2	97.76	98.15	1.4287
Yeast	1	0.2	58.92	52.91	21.335

5.1.4 Perbandingan Hasil Pengujian

Setelah pengujian dilakukan, maka hasil dapat dibandingkan antara satu metode dengan yang lain. Merujuk dari Subbab 5.1.1, Subbab 5.1.2, dan Subbab 5.1.3 maka dapat dirangkum menjadi Tabel perbandingan akurasi dan F-Measure yang ditunjukkan pada Tabel 5.10, sedangkan waktu komputasi ditunjukkan Tabel 5.11. Tabel 5.10 dan Tabel 5.11 menjadi rujukan untuk Subbab berikutnya.

Tabel 5.10 Perbandingan *F-measure*

Data set	Tingkat Outlier	Imbalance?	*Sparsity*	*F-measure*			
				NCC	NCCOR	*k*-NN	NB
Breast Cancer	Tinggi	Tidak	0.75	93.56	96.81	**97.31**	92.29
Ecoli	Rendah	Ya	0.63	63.57	**64.58**	62.85	3.41
Lymphography	Rendah	Ya	0.53	78.17	**80.15**	63.48	14.8
Vertebral	Tinggi	Tidak	0.67	71.99	**76.02**	73.8	75.68
Wine	Rendah	Tidak	0.59	95.85	**97.76**	95.95	97.78
Yeast	Rendah	Ya	0.69	52.64	58.92	**60.26**	0.09

Pada Tabel 5.10 terdapat keterangan karakteristik data set dan *F-measure* metode-metode yang telah diujikan. Karakteristik yang disebut adalah tingkat *outlier*, termasuk *imbalance* atau tidak, dan derajat *Sparsity*. Sedangkan pada Tabel 5.11, terdapat jumlah titik data, fitur, dan waktu komputasi metode yang diujikan.

Tabel 5.11 Perbandingan waktu komputasi

Data set	Titik Data	Jumlah Fitur	Waktu Komputasi			
			NCC	NCCOR	*k*-NN	NB
Breast Cancer	679	9	**0.939**	2.9875	47.6133	3.1052
Ecoli	336	7	**0.7664**	2.4859	16.0183	5.2785
Lymphography	148	18	**0.4623**	1.5363	6.1356	2.7731
Vertebral	240	6	**0.3922**	0.9711	14.9248	1.1342
Wine	178	13	**0.5065**	1.4287	7.3873	1.2952
Yeast	1364	8	**5.2945**	21.335	275.4713	27.06

5.2 Pembahasan

Pada subbab pembahasan dipaparkan bahasan mengenai jawaban permasalahan penelitian.

5.2.1 Tingkat efektifitas *outlier removal* pada metode usulan NCCOR

Pada Tabel 5.11 terdapat hasil dengan nilai diatas 0.8 pada *F-measure* data set *Yeast* dan Akurasi data set *Lymphography*. Untuk data set lain dengan pengukuran berbeda hanya berada di kisaran 0.65 hingga 0.2. Dengan demikian dapat ditarik kesimpulan bahwa tidak ada korelasi linier terhadap pengaruh *outlier* terhadap *F-measure* dan Akurasi. Namun efektifitas *outlier removal* dapat diuji dengan membandingkan antara NCC sebagai metode tanpa *outlier removal*.

Tingkat efektifitas adalah tingkatan seberapa efektif suatu metode dalam menghasilkan perubahan. Untuk mengetahui tingkat efektifitas *outlier removal* pada metode NCCOR perlu terdapat perbandingan dengan metode NCC sebagai metode awalan. Jika merujuk pada Tabel 5.1, Tabel 5.4, Tabel 5.5, Tabel 5.6, Tabel 5.7, dan Tabel 5.8, maka didapatkan analisis pada Tabel 5.11 dengan kolom F mewakili *F-measure*, A mewakili Akurasi, T mewakili waktu, dan R mewakili banyak data yang dihapus.

Tabel 5.12 Tabel Perbandingan antara metode NCC dan NCCOR

Data set	NCC			NCCOR						NCCOR-NCC		
	F	A	T	γ	n_0	R	F	A	T	dF	dA	dT
Breast cancer	93.56	94.29	0.93	1.5	0.1	37	96.81	97.14	2.98	3.25	2.85	-2.05
				1	0.2	98	95.71	96.19	3.83	2.15	1.9	-2.9
Ecoli	63.57	84.16	0.76	1	0.2	48	64.58	86.14	2.48	1.01	1.98	-1.72
				1.5	0.1	24	62.38	84.16	2.49	-1.19	0	-1.74
Lymphography	78.17	80	0.46	1	0.5	46	80.15	82.22	1.53	1.98	2.22	-1.07
				1	0.3	31	73.68	80	1.51	-4.49	0	-1.05
Vertebral	71.99	73.12	0.39	1	0.4	87	76.02	77.42	0.97	4.03	4.3	-0.58
				1.5	0.5	31	69.5	72.04	0.99	-2.49	-1.08	-0.6
Wine	95.85	96.3	0.50	1	0.2	25	97.76	98.15	1.42	1.91	1.85	-0.92
				1	0.6	54	95.98	96.3	1.10	0.13	0	-0.604
Yeast	52.64	47.76	5.29	1	0.3	312	58.92	52.47	21.33	6.28	4.71	-16.04
				2.5	0.1	20	54.88	51.79	17.46	2.24	4.03	-12.17

Tabel 5.10 menunjukkan bahwa terdapat perbedaan performa antara NCCOR dan NCC. Pada kolom dF, dA, dan dT ditunjukkan beda performa kedua metode tersebut, jika bernilai negatif maka lebih baik NCC, jika positif maka lebih baik NCCOR. Selisih performa kedua metode tersebut bervariasi untuk setiap data set. Pada data set *Breast cancer*, *Wine*, dan *Yeast* NCCOR unggul pada semua konfigurasi. Pada data set *Ecoli*, NCCOR unggul pada konfigurasi terbaik sebesar 1.01, namun kalah pada konfigurasi terburuk sebesar -1.19. Pada data

set *Lymphography* NCCOR unggul sebesar 1.98 pada konfigurasi terbaik namun kalah pada konfigurasi terburuk sebesar 4.49. Konfigurasi terbaik NCCOR unggul pada data set *Vertebral* sebesar 4.03 dan diungguli NCC pada konfigurasi terburuk dengan beda 2.49. Untuk perbedaan waktu komputasi, NCCOR hanya selisih rata-rata sebesar 3.73 ms dengan keunggulan pada NCC, namun pada data set *Yeast* terdapat perbedaan sebesar 16 ms dengan keunggulan pada NCC.

Jika merujuk pada Tabel 5.12, tidak selalu hasil NCCOR lebih baik dari NCC. Untuk beberapa konfigurasi parameter NCC justru lebih baik dari NCCOR. Konfigurasi parameter yang terbaik pun berbeda untuk setiap data set. Dapat diambil kesimpulan bahwa NCCOR dapat melakukan performa dengan baik jika mendapatkan konfigurasi parameter yang cocok untuk setiap data set.

5.2.2 Parameter optimal pada metode usulan NCCOR

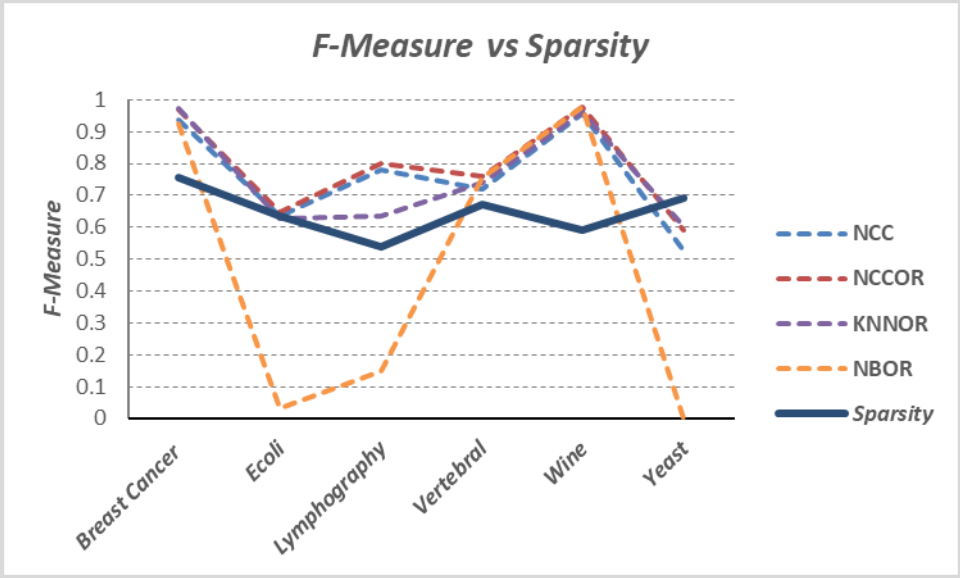
Parameter optimal adalah konfigurasi parameter dengan hasil yang terbaik ditinjau dari nilai *F-measure*, akurasi, dan waktu. Parameter optimal didapat dengan merujuk pada Tabel 5.10. Parameter optimal pada data set *Breast cancer* adalah $\gamma = 1.5$ dan $n_0 = 0.1$. Parameter optimal pada data set *Ecoli* adalah $\gamma = 1$ dan $n_0 = 0.2$. Parameter optimal pada data set *Lymphography* adalah $\gamma = 1$ dan $n_0 = 0.5$. Parameter optimal pada data set *Vertebral* adalah $\gamma = 1$ dan $n_0 = 0.5$. Parameter optimal pada data set *Wine* adalah $\gamma = 1$ dan $n_0 = 0.2$. Parameter optimal pada data set *Yeast* adalah $\gamma = 1$ dan $n_0 = 0.2$. Menurut hasil pengujian, parameter optimal berbeda untuk setiap data set, jadi untuk mendapatkan hasil terbaik harus melakukan konfigurasi parameter dan dibandingkan.

5.2.3 Perbandingan performa klasifikasi metode NCCOR dengan metode pembanding

Merujuk pada Tabel 5.10, ukuran *F-measure* antara metode pembanding dan NCCOR menunjukkan bahwa NCCOR memiliki F-measure yang baik jika dibanding metode pembanding. Pada data set *Breast cancer* semua metode dapat mencapai, performa diatas 90, namun NCCOR beda tipis dibanding *k*-NN. Untuk data set *Ecoli*, NCCOR unggul tipis dengan *F-measure* 64.58. Untuk data set *Lymphography*, NCCOR unggul tipis dengan 80.15 *F-measure* selisih cukup besar dengan *k*-NN, namun cukup jauh mengungguli NB. Pada data set *Vertebral*, NB mengungguli semua metode dengan mencapai 80.65 akurasi.. Pada data set *Yeast*, *k*-NN dapat memperoleh 60.26 *F-Measure*, mengungguli metode lainnya. NCCOR terbukti dapat mencapai hasil *F-measure* yang baik, namun masih diungguli oleh *k*-NN pada data set tertentu dengan sparsity yang tinggi. Hal ini menunjukkan bahwa NCCOR lebih cocok untuk data set dengan sparsity rendah karena NCCOR menggunakan centroid yang cocok digunakan jika datanya

terkumpul. Untuk performa pada data set imbalance, dapat diamati pada Tabel 5.10 bahwa yang paling terpengaruh adalah metode *Naïve Bayes*. NCCOR tidak terpengaruh karena yang penting untuk NCCOR adalah data yang dapat mewakili untuk membuat centroid yang tepat, jumlah data tidak menjadi masalah.

Analisis karakteristik data set dilakukan dengan menghitung derajat Sparsity masing-masing data set. Grafik perbandingan F-measure dengan Sparsity untuk tiap data set ditunjukkan Gambar 5.6.



Gambar 5.4 Grafik perbandingan *F-measure* dengan *Sparsity*

Pada Gambar 5.6 dapat diteliti bahwa data set Lymphography memiliki derajat Sparsity terendah yang artinya data set tersebut lebih berkelompok-kelompok. Data set Breast Cancer memiliki Sparsity tertinggi yang artinya data set tersebut memiliki data yang lebih tersebar. Sedangkan hubungan Sparsity dengan F-measure masing-masing metode adalah mayoritas metode yang digunakan yaitu NCC, NCCOR, *k*-NN, *k*-NNOR cocok digunakan untuk data set yang sparse. Namun, tidak demikian dengan data set NB dan NBOR yang tidak terpengaruh dengan derajat Sparsity. Metode NB dan NBOR mengalami performa yang buruk pada data set imbalance yaitu Lymphography, Ecoli, dan Yeast. Buruknya performa tersebut disebabkan metode NB dan NBOR memerlukan data set yang banyak dan seimbang untuk melakukan klasifikasi dengan efektif.

5.2.4 Perbandingan waktu komputasi metode NCCOR dengan metode pembanding

Waktu komputasi bergantung pada jumlah titik data dan fitur pada suatu data set. Merujuk pada Tabel 3.1 yang berisi tentang jumlah titik data dan fitur, maka kita dapat membentuk Gambar 5.3 yang berisi grafik perbandingan waktu komputasi tiap metode untuk setiap data set dengan ukuran yang berbeda-beda dengan detail pada Tabel 5.13.

Tabel 5.13 Perbandingan Waktu Komputasi antara metode NCCOR dengan metode pembanding

Data set	Titik Data	Fitur	Waktu Komputasi (ms)			
			NCC	NCCOR	k-NN	NB
Breast Cancer	679	9	0.939	2.9875	47.6133	3.1052
Ecoli	336	7	0.7664	2.4859	16.0183	5.2785
Lymphography	148	18	0.4623	1.5363	6.1356	2.7731
Vertebral	240	6	0.3922	0.9711	14.9248	1.1342
Wine	178	13	0.5065	1.4287	7.3873	1.2952
Yeast	1364	8	5.2945	21.335	275.4713	27.06
Korelasi waktu komputasi dengan ukuran data			0.9290	0.9245	0.94238	0.90080

Pada Tabel 5.13 dijelaskan dapat diketahui terdapat perbedaan waktu komputasi untuk setiap metode yang digunakan seperti yang telah dipaparkan sebelumnya. Tabel 5.13 memberikan tambahan penjelasan bahwa terdapat korelasi positif diatas 0.9 untuk semua metode antara ukuran data dengan waktu komputasi, hal ini membuktikan bahwa semakin besar jumlah data, maka semakin lama pula waktu komputasi yang diperlukan.

Pada Tabel 5.13 dapat diamati bahwa NCCOR cukup cepat dalam proses klasifikasi. Kecepatan komputasi tidak berbeda jauh dengan NCC, NB, dan NBOR namun dapat menghasilkan performa klasifikasi yang lebih baik. Waktu komputasi tercepat dihasilkan oleh NCC dan yang terlama oleh *k*-NN. Merujuk pada Tabel 5.3, Tabel 5.4, dan Tabel 5.5, waktu komputasi metode *k*-NN makin lama seiring dengan bertambahnya nilai parameter *k*, namun proses outlier removal pada k-NNOR berhasil mengurangi waktu komputasi dengan rata-rata 9.5 ms, hal ini disebabkan titik data yang dihitung berkurang dan mengakibatkan waktu komputasi yang lebih cepat. Sedangkan waktu komputasi NCCOR relatif stabil untuk semua konfigurasi parameter.