TESIS

ANALISIS KOMPARASI ALGORITMA *SUPPORT VECTOR MACHINE* DAN *RANDOM FOREST* UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS

(Studi Kasus: Universitas Darussalam Gontor)

Disusun oleh:

Nama : Widya Kurniawan
NIM : 19.51.1225
Konsentrasi : Informatics Technopreneurship

PROGRAM STUDI S2 TEKNIK INFORMATIKA
PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA
YOGYAKARTA
2021

TESIS

ANALISIS KOMPARASI ALGORITMA *SUPPORT VECTOR MACHINE* DAN *RANDOM FOREST* UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS

(Studi Kasus: Universitas Darussalam Gontor)

COMPARISON ANALYSIS OF *SUPPORT VECTOR MACHINE* AND *RANDOM FOREST* ALGORITHM FOR UNIVERSITY STUDENT VIOLATIONS CLASSIFICATION

(Case Study: University Of Darussalam Gontor)

Diajukan untuk memenuhi salah satu syarat memperoleh derajat Magister

Disusun oleh:

Nama	**: Widya Kurniawan**
NIM	**: 19.51.1225**
Konsentrasi	**: Informatics Technopreneurship**

PROGRAM STUDI S2 TEKNIK INFORMATIKA

PROGRAM PASCASARJANA UNIVERSITAS AMIKOM YOGYAKARTA

YOGYAKARTA

2021

HALAMAN PENGESAHAN

ANALISIS KOMPARASI ALGORITMA *SUPPORT VECTOR MACHINE* DAN *RANDOM FOREST* UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS
(Studi Kasus: Universitas Darussalam Gontor)

COMPARISON ANALYSIS OF *SUPPORT VECTOR MACHINE* AND *RANDOM FOREST* ALGORITHM FOR UNIVERSITY STUDENT VIOLATIONS CLASSIFICATION
(Case Study: University Of Darussalam Gontor)

Dipersiapkan dan Disusun oleh

Widya Kurniawan

19.51.1225

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Rabu, 5 Mei 2021

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 5 Mei 2021
Rektor

Prof. Dr. M. Suyanto, M.M.
NIK. 190302001

HALAMAN PERSETUJUAN

ANALISIS KOMPARASI ALGORITMA *SUPPORT VECTOR MACHINE* DAN *RANDOM FOREST* UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS
(Studi Kasus: Universitas Darussalam Gontor)

COMPARISON ANALYSIS OF *SUPPORT VECTOR MACHINE* AND *RANDOM FOREST* ALGORITHM FOR UNIVERSITY STUDENT VIOLATIONS CLASSIFICATION
(Case Study: University Of Darussalam Gontor)

Dipersiapkan dan Disusun oleh

Widya Kurniawan

19.51.1225

Telah Diujikan dan Dipertahankan dalam Sidang Ujian Tesis
Program Studi S2 Teknik Informatika
Program Pascasarjana Universitas AMIKOM Yogyakarta
pada hari Rabu, 5 Mei 2021

Pembimbing Utama	**Anggota Tim Penguji**
Dr.Kusrini, M.Kom	**Dr. Arief Setyanto, S.Si., M.T.**
NIK. 190302106	**NIK. 190302036**
	Alva.H.Muhammad,S.T.,M.Eng.,Ph.D.
Pembimbing Pendamping	**NIK. 190302493**
Drs.Asro Nasiri, M.Kom	**Dr.Kusrini, M.Kom**
NIK. 190302152	**NIK. 190302106**

Tesis ini telah diterima sebagai salah satu persyaratan
untuk memperoleh gelar Magister Komputer

Yogyakarta, 5 Mei 2021
Direktur Program Pascasarjana

Dr. Kusrini, M.Kom.
NIK. 190302106

HALAMAN PERNYATAAN KEASLIAN TESIS

HALAMAN PERNYATAAN KEASLIAN TESIS

Yang bertandatangan di bawah ini,

Nama mahasiswa : Widya Kurniawan
NIM : 19.51.1225
Konsentrasi : Informatics Technoprenecurship

Menyatakan bahwa Tesis dengan judul berikut:
Analisis Komparasi Algoritma *Support Vector Machine* Dan *Random Forest* Untuk klasifikasi Tingkat Pelanggaran Mahasiswa Di Universitas (Studi Kasus : Universitas Darussalam Gontor)

Dosen Pembimbing Utama : Dr.Kusrini, M.Kom
Dosen Pembimbing Pendamping : Drs.Asro Nasiri, M.Kom

1. Karya tulis ini adalah benar-benar ASLI dan BELUM PERNAH diajukan untuk mendapatkan gelar akademik, baik di Universitas AMIKOM Yogyakarta maupun di Perguruan Tinggi lainnya
2. Karya tulis ini merupakan gagasan, rumusan dan penelitian SAYA sendiri, tanpa bantuan pihak lain kecuali arahan dari Tim Dosen Pembimbing
3. Dalam karya tulis ini tidak terdapat karya atau pendapat orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan disebutkan dalam Daftar Pustaka pada karya tulis ini
4. Perangkat lunak yang digunakan dalam penelitian ini sepenuhnya menjadi tanggung jawab SAYA, bukan tanggung jawab Universitas AMIKOM Yogyakarta
5. Pernyataan ini SAYA buat dengan sesungguhnya, apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka SAYA bersedia menerima SANKSI AKADEMIK dengan pencabutan gelar yang sudah diperoleh, serta sanksi lainnya sesuai dengan norma yang berlaku di Perguruan Tinggi

Yogyakarta, 5 Mei 2021
Yang Menyatakan,

Widya Kurniawan

HALAMAN PERSEMBAHAN

Sujud syukurku kusembahkan kepadaMu ya Allah, Tuhan Yang Maha Agung dan Maha Tinggi. Atas takdirmu saya bisa menjadi pribadi yang berpikir, berilmu, beriman dan bersabar. Semoga keberhasilan ini menjadi satu langkah awal untuk masa depanku, dalam meraih cita-cita saya.

Dengan ini saya persembahkan karya ini untuk Ibu, Terima kasih atas kasih sayang berlimpah dari mulai saya lahir, yang telah merawat dan mendidik hingga bisa pada titik ini. Lalu teruntuk Ayah, terima kasih juga atas limpahan doa yang tak terbatas. Serta segala hal yang telah Ayah lakukan, semua yang terbaik.

Terima kasih juga yang tak terhingga untuk para dosen pembimbing, Ibu Dr. Kusrini,M.Kom.selaku pembimbing pertama dan Bapak Drs. Asro Nasiri, M.Kom. sebagai pembimbing kedua, yang dengan sabar membimbing saya selama mengerjakan Tesis.

Ucapan terima kasih ini saya persembahkan juga untuk seluruh teman-teman saya di Magister Teknik Informatika 2019. Terima kasih untuk tawa yang setiap hari kita miliki, dan atas solidaritas yang luar biasa. Sehingga masa kuliah selama 2 tahun ini menjadi lebih berarti.

Untuk semua pihak yang saya sebutkan, terima kasih atas semuanya. Semoga Tuhan senantiasa membalas setiap kebaikan kalian. Serta kehidupan kalian semua juga dimudahkan dan diberkahi selalu oleh Allah SWT.

Saya menyadari bahwa hasil karya tesis ini masih jauh dari kata sempurna, tetapi saya harap isinya tetap memberi manfaat sebagai ilmu dan pengetahuan bagi para pembacanya.

HALAMAN MOTTO

MOTTO :

"Dan bahwasanya seorang manusia tiada memperoleh selain apa yang telah diusahakannya" (An Najm : 39).

"Barang siapa yang mempelajari ilmu pengetahuan yang seharusnya yang ditunjukan untuk mencari ridho Allah bahkan hanya untuk mendapatkan kedudukan/kekayaan duniawi maka ia tidak akan mendapatkan baunya surga nanti pada hari kiamat (HR. Abu Hurairah radhiallau anhu)"

" Sesungguhnya Allah tidak akan merubah keadaan suatu kaum sehingga mereka merubah keadaan yang ada pada diri mereka sendiri " (QS. Ar Ra'ad :11).

KATA PENGANTAR

Puji syukur penulis Panjatkan kepada Tuhan Yang Maha Esa atas segala berkat dan kasih karunia-Nya sehingga Tesis ini dapat diselesaikan dengan baik dan tepat pada waktunya.

Tesis ini ditulis dalam rangka memenuhi syarat untuk mencapai gelar Magister Komputer pada Program Studi Teknik Informatika, Sekolah Pascasarjana Universitas AMIKOM, Yogyakarta.

Adapun judul proposal penelitian ini adalah: " Analisis Komparasi Algoritma *Support Vector Machine* Dan *Random Forest* Untuk Klasifikasi Tingkat Pelanggaran Mahasiswa Di Universitas (Studi Kasus: Universitas Darussalam Gontor)". Di dalam menyelesaikan Tesis ini, penulis banyak memperoleh bantuan baik berupa pengajaran, bimbingan dan arahan dari berbagai pihak.

Oleh karena itu Penulis menyampaikan ucapan terima kasih dan penghargaan setinggi-tingginya kepada yang terhormat para pembimbing. Dimana di tengah-tengah kesibukannya masih tetap meluangkan waktunya untuk memberikan bimbingan, petunjuk, dan mendorong semangat penulis untuk menyelesaikan penulisan Tesis ini. Perkenankanlah juga, penulis menyampaikan ucapan terima kasih kepada semua pihak yang terlibat dalam penyelesaian studi ini, kepada:

1. Rektor Universitas AMIKOM Yogyakarta, Bapak Prof. Dr. M. Suyanto, M.M. atas kesempatan dan fasilitas yang diberikan kepada penulis untuk mengikuti dan menyelesaikan pendidikan.
2. Dr.Kusrini, M.Kom., sebagai Ketua Program studi Magister Teknik Informatika sekaligus sebagai Pembimbing utama penulis, yang telah

meluangkan waktunya dan dengan penuh perhatian memberikan dorongan, bimbingan, saran kepada penulis. .

3. Drs. Asro Nasiri, M.Kom., sebagai Komisi Pembimbing yang telah meluangkan waktunya dan dengan penuh perhatian memberikan dorongan, bimbingan, saran dan masukan yang sangat penting.
4. Orang Tua tercinta yang mendidik dengan penuh rasa kasih sayang dan senantiasa memberi semangat dan dorongan kepada penulis.
5. Kepada Adek Debi, yang Penulis sayangi, atas pengertiannya serta memberikan Doa dan semangat kepada penulis dalam menyelesaikan Tesis.
6. Teman baik saya yang luar biasa, dalam memberi dukungan dan bantuan. Saudara Fathin Muhammad Wasmanson yang selama ini sudah menjadi tempat saya berlari dan berkonsultasi ketika saya merasa sulit dalam mengerjakan Tesis.
7. Kepada Rekan-rekan mahasiswa pascasarjana, dan rekan-rekan kerja saya yang tidak dapat disebutkan satu persatu.

Akhirnya penulis berharap semoga Tesis ini dapat bermanfaat dan permintaan maaf yang tulus jika seandainya dalam penulisan ini terdapat kekurangan dan kekeliruan, penulis juga menerima kritik dan saran yang bersifat membangun demi menyempurnakan penulisan tesis ini.

Yogyakarta, 5 Mei 2021

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

INTISARI

Universitas Darussalam (UNIDA) Gontor merupakan perguruan tinggi pesantren. Sistem pesantren mewajibkan mahasiswanya tinggal didalam kampus dengan ber-asrama. Meningkatnya pelanggaran ditunjukan dengan banyaknya data rekap pelanggaran, semester genap tahun 2019/2020 terdapat 1.478 kasus pelanggaran. Pada penelitian ini teknik data mining dilakukan dengan membandingkan antar algoritma untuk mengklasifikasikan data pelanggaran menggunakan algoritma *Support Vector Machine* dan *Random Forest*. Penelitian dilakukan untuk membantu sistem dalam pengklasifikasian pelanggaran yang baru dimasukan, tanpa harus memprediksi kategorinya.

Penelitian ini menggunakan dua data dari sumber sama dengan kuantitas dan kualitas berbeda. Digunakan 9 alur tahapan pada penelitian, yaitu membangun dataset, *Text Preprocessing* yang didalamnya memiliki 3 tahapan, dilanjut *Feature Selection, Text Representation,* dan *Application of Text Mining Techniques*. Proses perhitungan dilihat dari performa akurasinya antara algoritma biasa dengan dilakukan optimalisasi *stemming*. Pada SVM ditambah optimalisasi menggunakan OvO. Uji coba dengan *K-fold Validation*, tahap evaluasi menggunakan *confussion matrix*, dan terakhir perbandingan akurasi dari semua proses yang dilakukan dengan kedua data yang digunakan.

Ketepatan Algoritma SVM dalam mengklasifikasikan dataset pertama dan kedua, dengan metode *stemming* menghasilkan akurasi 98,9% dan 99,5% lebih tinggi dari RF yang hanya 98,6% dan 98,2%. Uji coba dengan metode *K-fold Validation* SVM menghasilkan akurasi yang sama 99,5%, lebih tinggi dari hasil RF yang hanya menghasilkan 98,9% dan 98,8%. Evaluasi dengan metode *Confuss Matrix* SVM menghasilkan akurasi sebesar 98,9% dan 99,5%, lebih tinggi dari hasil pengklasifikasian RF yang hanya 98,6% dan 99,1%. Hasil perbandingan kinerja kedua algoritma dengan 3 metode menunjukkan bahwa, data pertama dan kedua nilai rata-rata dari algoritma SVM lebih baik dari algoritma RF.

Kata kunci: *Support Vector Machine*, *Random Forest*, pelanggaran, *stemming*

ABSTRACT

Darussalam University (UNIDA) Gontor is a boarding school. The pesantren system requires students to live on campus by boarding. The increase in violations is indicated by the large number of violation recap data, even semester 2019/2020 there are 1,478 cases of violations. In this study, data mining techniques were carried out by comparing between algorithms to classify violation data using the Support Vector Machine and Random Forest algorithms. Research was conducted to assist the system in classifying newly entered violations, without having to predict their categories.

This study uses two data from the same source with different quantity and quality. 9 stages were used in the research, namely building a dataset, Text Preprocessing which included 3 stages, followed by Feature Selection, Text Representation, and Application of Text Mining Techniques. The calculation process is seen from the accuracy performance between ordinary algorithms and stemming optimization. In the SVM plus optimization using OvO. Tests with K-fold Validation, the evaluation stage using a configuration matrix, and finally the comparison of the accuracy of all processes carried out with the two data used.

The accuracy of the SVM algorithm in classifying the first and second datasets, with the stemming method produces an accuracy of 98.9% and 99.5% higher than RF which is only 98.6% and 98.2%. The trial with the K-fold Validation SVM method produced the same accuracy of 99.5%, higher than the RF results which only produced 98.9% and 98.8%. Evaluation using the Confuss Matrix SVM method produces an accuracy of 98.9% and 99.5%, higher than the RF classification results which are only 98.6% and 99.1%. The results of the comparison of the performance of the two algorithms with 3 methods show that, the first and second data, the average value of the SVM algorithm is better than the RF algorithm.

Keyword: Support Vector Machine, Random Forest, violation, stemming

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Universitas Darussalam (UNIDA) Gontor merupakan perguruan tinggi pesantren yang berada di bawah naungan Pondok Modern Darussalam Gontor. Universitas atau Perguruan Tinggi Pesantren hakikatnya merupakan suatu perguruan tinggi yang cikal bakalnya berasal dari pesantren dimaksudkan sebagai kelanjutan setelah menyelesaikan pendidikan di pondok pesantren yang sederajat dengan SMP dan SMA. UNIDA menyebut para pelajarnya dengan "mahasiswa santri", yang mana dosennya berada di dalamnya dan tinggal bersama mahasiswa santri sehingga Interaksi antara dosen dan mahasiswa santri dapat berlangsung lebih intensif dan dapat menanamkan nilai-nilai keilmuan selama 24 jam.

Sistem pesantren yang diterapkan oleh UNIDA mewajibkan mahasiswanya untuk tinggal didalam kampus dengan sistem ber-asrama. Mahasiswa yang banyak didalamnya memunculkan milieu belajar bervariasi dan perlu pengawalan, khususnya dalam disiplin. Untuk mendukung proses disiplin, harus menerapkan strategi supaya mahasiswa dapat berdisiplin secara efektif. Hal ini dilakukan dalam bentuk pengelolaan pengajaran dalam kehidupan sehari-hari (Maya Nurfitriyanti, 2014). Disiplin tidak lepas dari pelanggaran sehingga terbuatnya peraturan didalamnya, dan akan menimbulkan hal bertentangan dengan beberapa sifat mahasiswa, dengan adanya disiplin berarti menjadi hukuman atau latihan yang membetulkan serta kontrol yang memperkuat ketaatan (Binning, dkk., 2019).

Meningkatnya pelanggaran di UNIDA ditunjukan dengan banyaknya poin pelanggaran yang didapat oleh siswa. Data rekap pelanggaran diperoleh dari *https://simpel.unida.gontor.ac.id* untuk semester genap tahun 2019/2020 terdapat 1.483 kasus dengan rincian pelanggaran dimana bulan Oktober 2019 tercatat 36 pelanggaran, bulan November 2019 tercatat 8 pelanggaran, Desember 2019 tercatat 225 pelanggaran, pada Januari 2020 tercatat 181 pelanggaran, bulan Februari 2020 tercatat 383, dan bulan Maret 2020 tercatat 317, April 2020 tercatat 35, Mei 2020 tercatat 40, Juni tercatat 45, Juli 2020 tercatat 57, Agustus 2020 tercatat 26 kasus, September 2020 tercatat 43 kasus, Oktober 2020 tercatat 16, November 2020 tercatat 61 kasus, Desember tercatat 10 pelanggaran. Komposisi jenis pelanggaran secara umum yang paling sering dilakukan tercatat alpa absen wajib, terlambat datang ke kampus, pulang ke asrama tengah malam, keluar kampus tanpa izin, menyalahgunakan smartphone, berhubungan dengan lawan jenis, mencuri, memalsukan tandatangan atau dokumen, dan berkelahi, dan masih banyak jenis pelanggaran lain yang dilakukan.

Penelitian ini fokus pada pemanfaatan data mining untuk mengklasifikasikan jenis tingkat pelanggaran yang terjadi. Untuk menyelesaikan masalah yang dihadapi, teknik data mining yang dipilih adalah dengan cara mengklasifikasi mahasiswa yang memiliki catatan pelanggaran. Teknik data mining dapat digunakan untuk memprediksi berdasarkan data yang telah tersedia dimasa lalu (Aries Saifudin, 2018).

Untuk melakukan klasifiaksi tingkat pelanggaran mahasiswa di UNIDA, dapat dilakukan dengan cara membandingkan antar algoritma klasifikasi untuk

menganalisa dan mengklasifikasikan data pelanggaran historikal mahasiswa, namun pada Tesis ini penulis hanya akan menggunakan algoritma *Support Vector Machine* dan *Random Forest* untuk mengolah, mengklasifikasikan, serta memining knowledge dari dataset pelanggaran mahasiswa. Beberapa penelitian terdahulu telah mengkaji kinerja algoritma yang akan dibahas dan telah diterapkan dibeberapa bidang. Pada bidang pendidikan diantaranya penelitian untuk mengetahui lama studi mahasiswa yang membandingkan metode klasifikasi *Random Forest* dan Support Vector Machine. Analisis data menggunakan analisis deskriptif untuk mengetahui gambaran umum data dan perbandingan hasil analisis klasifikasi *Support Vector Machine* (SVM) dan *Random Forest*. Untuk metode SVM, peneliti menggunakan kernel RBF dan Sigmoid. Akurasi SVM kernel sigmoid dengan parameter optimum C= 10 dan gamma= 1 adalah sebesar 68%. Lalu tingkat akurasi untuk metode *Random Forest* dengan nilai optimum m= 2 dan k=500 sebesar 80%. Semua nilai akurasi tersebut didapat menggunakan analisis pada data asli yang berupa imbalance data (Syauqi Amri Yahya, 2018). Untuk mengetahui tingkat kelulusan mahasiswa diperlukan informasi yang cepat, tepat, dan akurat, maka dilakukan penelitian dengan menggunakan data mining menggunakan komparasi algoritma C4.5, *naive bayes*,dan *Random Forest*. Penelitian ini dilakukan dengan membagi data testing dan data training dengan pengujian menggunakan k-fold cross validation. Data diolah menggunakan rapid miner, Jika dibandingkan dengan nilai akurasi algoritma *naive bayes* dan algoritma *Random Forest* nilai akurasi dengan menggunakan algoritma klasifikasi C4.5 adalah yang terbesar (Ibnu.A, Taransa.A. Tutupoly, Ade.S, 2018). Pada penelitian terkait seleksi calon mahasiswa

pada penerimaan mahasiswa baru, teknik data mining dan machine learning dapat digunakan untuk memprediksi berdasarkan data-data masa lalu. Metode data mining yang digunakan untuk memprediksi adalah klasifikasi, Penelitian ini dilakukan dengan cara menerapkan dataset ketepatan waktu kelulusan mahasiswa. Pada penelitian ini diusulkan model prediksi dengan 10 algoritma klasifikasi. Untuk menguji model yang diusulkan digunakan teknik validasi 10-fold cross validation. Pengukuran kinerja model dilakukan dengan dilakukan menggunakan confusion matrix. Model disimulasikan menggunakan software RapidMiner Studio 6.0.001 starter edition dan dataset mahasiswa yang telah dikumpulkan. Dari hasil pengukuran diperoleh model terbaik yaitu *Support Vector Machine* (SVM) dengan akurasi 65.00%. Tetapi akurasi ini masih jauh dari nilai excellent (sangat baik) (Aries Saifudin, 2018). Dalam bidang keuangan digunakan untuk pengujian prediksi kelancaran piutang pelanggan menggunakan dataset transaksi piutang perusahaan swasta selama tiga tahun, dengan membandingkan akurasi algoritma *Support Vector Machine* (SVM) dan *Random Forest*. Pada penelitian ini penulis menggunakan aplikasi Weka 3.8, proses pengumpulan data dan preparasi dihasilkan 10.230 baris data, kemudian dilakukan labeling dengan menggunakan metode expert judgement, Dilanjutkan dengan pengujian FS menggunakan PCA dengan kombinasi parameter variance covered dari 0,95 sampai 1. Setiap dataset baru yang terbentuk dari algoritma PCA akan diuji akurasinya menggunakan *Random Forest* dan SVM. *Random Forest*, menghasilkan akurasinya adalah ACC 83,21% dan AUC 94,9%. Sedangkan akurasi dari algoritma SVM menghasilkan rata-rata ACC 75%-an dan AUC 81%-an (Ferry.I dan Febriliyan.S, 2018). Dalam

bidang sosial penelitian ini menerapkan metode klasifikasi untuk mengetahui tingkah laku bully pada aplikasi whatsapp, penelitian ini mengklasifikasikan algoritma K-Nearest Neighbor, Naïve Bayes Classifier dan Support Vector Machine. Metode penelitian yang digunakan pada penelitian ini menggunakan model KDD (Knowledge Discovery in Database). Data yang digunakan dari grup WhatsApp, Hasil pengujian menunjukkan bahwa akurasi yang dimiliki oleh algoritma k-NN, NBC dan SVM yaitu masing-masing sebesar 81.32%, 78.95% dan 81.58%. Pengujian tersebut dapat diketahui bahwa algoritma SVM lebih baik dibanding kedua algoritma lainnya (Irwansyah.S dan Didi.R, 2019). Penelitian Dalam bidang medis juga menerapkan untuk memprediksi serangan epilepsi. Penelitian ini menggunakan dua metode klasifikasi yaitu *Random Forest* (RF) dan *Support Vector Machine* (SVM). Variabel yang digunakan dalam penelitian ini adalah sinyal EEG yang telah melewati proses DWT dan melakukan Pemisahan Data Training dan Testing setelah proses preprocessing, maka akan diperoleh tiga dataset. Masing-masing dataset yang diperoleh kemudian dibagi menjadi dua yakni data training dan data testing. Hasil perbandingan kinerja kedua metode menunjukkan bahwa data training nilai rata-rata dari metode *Random Forest* lebih baik dari metode SVM. Sedangkan untuk data testing nilai rata-rata dari metode SVM lebih baik dari pada *Random Forest* (Muhammad.I.Fachruddin, 2015).

Pemilihan penggunaan algoritma *Support Vector Machine* dan *Random Forest* pada penelitian ini didasarkan pada beberapa alasan, yaitu: Selain kedua algoritma tersebut sama-sama mudah diimplementasikan dan sama-sama dapat memberikan hasil yang baik dalam kasus klasifikasi, kedua algoritma tersebut juga

mempunyai beberapa keunggulan masing-masing. Berdasarkan uraian pada latar belakang dapat diidentifikasikan bahwa penyebab mahasiswa banyak melakukan pelanggaran adalah pengaruh dari tingkat pelanggaran yang tinggi sehingga menjadikan pola pikir mahasiswa, bahwa melanggar merupakan hal yang biasa. Data dari hasil penelitian ini akan digunakan sebagai data pelatihan untuk mengklasifikasikan perilaku mahasiswa yang melakukan pelanggaran nantinya. Adanya penelitian ini dapat membantu sistem dalam melakukan mengklasifikasikan pelanggaran mahasiswa yang baru melakukan pelanggaran tanpa harus mengira-mengira kategorinya, yang nantinya dosen dan pihak kampus dapat mengetahui jenis tingkatan masalah mahasiswa yang sedang dihadapinya didalam kampus.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan sebelumnya,maka rumusan masalah dari penelitian ini adalah :

a. Seberapa besar ketepatan metode klasifikasi menggunakan algoritma *Support Vector Machine* jika dibandingkan dengan *Random Forest* pada penggunaan data pertama dan kedua, menggunakan metode tanpa *stemming* dengan *stemming*, dalam mengklasifikasikan data pelanggaran ?
b. Seberapa besar ketepatan algoritma *Support Vector Machine* jika dibandingkan dengan *Random Forest* pada penggunaan data pertama dan kedua, menggunakan metode tanpa *stemming* dengan *stemming*, dalam mengklasifikasikan data pelanggaran?

c. Berapa ketepatan algoritma *Support Vector Machine* jika dibandingkan dengan *Random Forest* pada penggunaan data pertama dan kedua, menggunakan metode tanpa *stemming* dengan *stemming*, dalam mengklasifikasikan data pelanggaran?

d. Dari penggunaan dua algoritma yang digunakan, *Support Vector Machine* dan *Random Forest* dengan penggunaan tiga metode uji coba berbeda yaitu *stemming, K-fold Cross Validation,* dan *Confuss Matrix.* Algoritma manakah yang memiliki nilai akurasi paling baik?

1.3. Batasan Masalah

Untuk menghindari adanya penyimpangan dan pelebaran pembahasan masalah, serta topik lebih terarah dan terfokus pada tujuan yang ingin dicapai dalam penyusunan penelitian, maka peneliti memberikan batasan masalah, yaitu :

a. Data yang digunakan sebagai acuan klasifikasi adalah data mahasiswa UNIDA dari semester 1 sampai 9 keatas yang melakukan pelanggaran dari bulan Oktober 2019 sampai bulan Desember 2020.

b. Metode yang dibandingkan adalah metode *Support Vector Machine* dan *Random Forest.*

c. Pelanggaran yang dianalisis untuk diklasifikasikan terdiri dari 3 jenis yaitu pelanggaran ringan, pelanggaran sedang, dan pelanggaran berat.

d. Untuk semua variabel yang digunakan merupakan data hasil rata-rata yang diperoleh langsung dari website *https://simpel.unida.gontor.ac.id*

1.4. Tujuan Penelitian

Berdasar pada rumusan masalah yang tertera di atas, maka tujuan peneliti melakukan penelitian ini ialah :

a. Membandingkan algoritma *Support Vector Machine* dan *Random Forest* berdasarkan 3 tahapan yaitu *stemming, K-fold,* dan *Confussion Matrix*
b. Mengidentifikasi algoritma terbaik dari segi klasifikasi di antara dua pilihan algoritma klasifikasi *Support Vector Machine* dan *Random Forest*.

1.5. Manfaat Penelitian

Manfaat yang diperoleh dari dilakukannya penelitian ini antara lain :

a. Bagi akademik :

Hasil dari penelitian ini dapat dimanfaatkan sebagai berikut :

1. Memberikan informasi untuk menganalisa kelengkapan data pelanggaran mahasiswa yang telah tercatat didalam sistem.
2. Membantu perguruan tinggi dalam mengklasifikasikan jenis pelanggaran mahasiswa untuk bisa mengambil keputusan disiplin yang akan diterapkan kepada mahasiswa.

b. Bagi masyarakat dan ilmu pengetahuan :

Hasil dari penelitian ini dapat dimanfaatkan sebagai berikut :

1. Memberikan pertimbangan bahwa teknik atau model klasifikasi yang digunakan pada penelitian ini dapat dimanfaatkan tidak hanya fokus pada bidang pendidikan tapi juga berperan pada bidang keuangan, medis, bisnis, dan bidang lainnya.

c. Bagi peneliti :

Manfaat yang dapat digunakan dari penelitian ini sebagai berikut :

1. Memberikan kontribusi keilmuan pada penelitian bidang klasifikasi data mining khususnya untuk prediksi tingkat pelanggaran mahasiswa.

2. Bisa mengetahui perbandingan tingkat akurasi dan kecepatan terhadap algoritma yang digunakan.
3. Memberikan prediksi digunakan untuk rujukan penelitian selanjutnya dengan metode yang sama ataupun dengan cara memaksimalkan penggunaan algoritma lainnya

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Sebelum melakukan sebuah penelitian, penulis terlebih dahulu melakukan tinjauan pustaka dari peneliti terdahulu, karena pada penelitian terdahulu merupakan acuan dalam penulisan penelitian sehingga penulis dapat memperbanyak teori yang akan digunakan. Dari penelitian terdahulu, penulis tidak menemukan penelitian dengan judul yang sama seperti judul penelitian penulis. Namun penulis mengangkat beberapa metode penelitian sebagai referensi dalam menganalisa sebuah metode dan memperkaya bahan kajian pada penelitian penulis yang berkaitan dengan algoritma Support Vector Machine, dan *Random Forest*. Berikut adalah beberapa penelitian yang terkait dengan masalah tersebut :

A. Syauqi Amri Yahya (2018)

Pada penelitian yang dilakukan oleh Syauqi Amri Yahya "Klasifikasi Ketepatan Lama Studi Mahasiswa Menggunakan Metode *Support Vector Machine* Dan *Random Forest (Studi Kasus : Data Lama Studi Alumni Universitas Islam Indonesia Tahun Kelulusan 2000-2017)*"(2019). Penelitian ini dibuat untuk mengetahui lama Studi Mahasiswa di Universitas Islam Indonesia dengan membandingkan metode klasifikasi *Random Forest dan Support Vector Machine.* Ketepatan lama studi alumni dibagi menjadi 2, yaitu tepat waktu dan tidak tepat waktu. Alumni dikatakan tepat waktu jika lulus dengan menempuh studi maksimal 4 tahun lamanya, sedangkan dikatakan tidak tepat waktu jika alumni menyelesaikan masa studinya lebih

dari 4 tahun. Variabel penelitian yang digunakan dalam penelitian ini adalah sebanyak 10 variabel. Untuk analisis data menggunakan analisis deskriptif untuk mengetahui gambaran umum data dan perbandingan hasil analisis klasifikasi.

Untuk metode SVM, peneliti menggunakan kernel RBF dan Sigmoid dengan parameter C =0.1, 1, 5, 10, 50 dan *gamma* = 1, 2, 3, 4. metode SVM yang digunakan dalam penelitian ini adalah menggunakan *Kernel Radian Basis Function* (RBF). Metode RBF ini memerlukan nilai parameter *cost* (C) dan *gamma* yang nilailnya ditentukan oleh peneliti. Untuk metode *Random Forest*, Analisis klasifikasi *Random Forest* dilakukan dengan menentukan berapa banyak pohon yang akan terbentu (*ntree*) dan menentukan berapa banyak *random sample* yang diambil untuk setiap percobaan (*mtry*).peneliti menggunakan pohon sebanyak 25,50,100,500, dan 1000. Proses untuk data *testing* yang dilakukan akan menghasilkan tabel *confusionan matrix*, yang mana dengan itu bisa terlihat tingkat akurasi dari masing-masing metode klasifikasi.

B. Ibnu Alfarobi, Taransa Agasya Tutupoly, dan Ade Suryanto (2018)

Pada penelitian yang dilakukan oleh Ibnu Alfarobi, Taransa Agasya Tutupoly, dan Ade Suryanto yang berjudul "Komparasi Algoritma C4.5, *Naive Beyes*, dan *Random Forest* Untuk Klasifikasi Data Kelulusan Mahasiswa Jakarta" penelitian digunakan untuk mengetahui tingkat kelulusan mahasiswa dalam institusi di Jakarta dengan membagi data testing dan data training untuk mengkomparasikan algoritma C4.5, *Naive Bayes*,

dan *Random Forest* dalam penentuan klasifikasi data kelulusan mahasiswa. Untuk pengujiannya menggunakan *k-fold cross validation* dengan perbandingan antara *data testing* dan *data training* 10 : 90, 20 : 80, 30 : 70 dan mengulang pengujian tersebut beberapa kali.

Hasil penelitian menunjukkan bahwa secara keseluruhan algoritma C4.5 mempunyai akurasi paling besar jika dibandingkan dengan algoritma lainnya dengan tingkat akurasi sebesar 85.34% pada eksperimen pertama dan 89.06% pada ekperimen ketiga. Sedangkan pengukuran dengan menggunakan ROC curve, algoritma *Naive Bayes* menjadi algoritma yang mempunyai tingkat akurasi teringgi dibandingkan dengan algoritma C4.5 dan *Random Forest*.

C. Ferry Irawan dan Febriliyan Samopa (2018)

Pada penelitian yang dilakukan oleh Ferry Irawan dan Febriliyan Samopa yang berjudul "A Comparative Assessment of *Random Forest* and SVM Algorithms, Using Combination of Principal Component Analysis and SMOTE For Accounts Receivable Seamless Prediction, case study company X in Surabaya (2018)". Dalam penelitian ini membahas perbandingan akurasi algoritma *Support Vector Machine* (SVM) dan *Random Forest* untuk pengujian prediksi kelancaran piutang pelanggan di suatu perusahaan swasta selama tiga tahun. Pada penelitian ini penulis menggunakan aplikasi Weka 3.8, proses pengumpulan data dan preparasi dihasilkan 10.230 baris data, kemudian dilakukan labeling dengan menggunakan metode expert judgement dan didapatkan kondisi dataset

tidak balance, Dilanjutkan dengan pengujian FS menggunakan PCA dengan kombinasi parameter *variance covered* dari 0,95 sampai 1.

Setiap dataset baru yang terbentuk dari algoritma PCA akan diuji akurasinya menggunakan *Random Forest* dan SVM dari percobaan beberapa kali menggunakan *Random Forest*, algoritma ini secara konsisten mampu menghasilkan akurasi yang tinggi dan akurasi terbaiknya adalah ACC 83,21% dan AUC 94,9%. Sedangkan akurasi dari algoritma SVM banyak dipengaruhi oleh pemilihan parameter "kernel type", dimana parameter "kernel type" *polynomial* dan RBF menghasilkan rata-rata ACC 75%-an dan AUC 81%-an.

D. Irwansyah Saputra Dan Didi Rosiyadi (2019)

Pada penelitian yang dilakukan oleh Irwansyah Saputra Dan Didi Rosiyadi yang berjudul " Perbandingan Kinerja Algoritma *K-Nearest Neighbor, Naïve Bayes Classifier,* dan *Support Vector Machine* dalam Klasifikasi Tingkah Laku Bully pada Aplikasi Whatsapp (2019)". Penelitian ini menerapkan metode klasifikasi untuk mengetahui tingkah laku bully pada aplikasi whatsapp, penelitian ini mengklasifikasikan algoritma *K-Nearest Neighbor, Naïve Bayes Classifier* dan *Support Vector Machine*. Metode yang digunakan dalam analisis data, pelabelan dan klasifikasi hingga jutaan *tweet*, dan algoritma pemodelan yang dibangun *Random Forest classifier* dapat membedakan antara pengguna normal, agresif dan bullying dengan akurasi tinggi.

Metode penelitian yang digunakan pada penelitian ini menggunakan model KDD (*Knowledge Discovery in Database*). Penelitian ini bertujuan untuk mengklasifikasikan data uji kedalam kelas yang sudah ditentukan, yaitu *bully* dan tidak *bully*. Data yang digunakan dari grup WhatsApp, Data yang digunakan untuk proses validasi terdiri dari dua jenis yaitu data dengan didapatkan kelas *bully* sebanyak 204 *record* dan data dengan kelas tidak *bully* sebanyak 176 *record*. Hasil pengujian menunjukkan bahwa akurasi yang dimiliki oleh algoritma k-NN, NBC dan SVM yaitu masing-masing sebesar 81.32%, 78.95% dan 81.58%. Pengujian dapat diketahui bahwa algoritma SVM lebih baik dibanding kedua algoritma lainnya.

E. Aries Saifudin (2018)

Pada penelitian yang dilakukan oleh Aries Saifudin yang berjudul "Metode Data Mining Untuk Seleksi Calon Mahasiswa Pada Penerimaan Mahasiswa Baru Di Universitas Pamulang (2018)". Penelitian ini dilakukan untuk seleksi calon mahasiswa pada penerimaan mahasiswa baru. Tapi dengan melihat mahasiswa terdahulu banyak yang keluar di tiap semester, sehingga menyebabkan rasio jumlah mahasiswa baru dengan jumlah yang lulus tidak seimbang. Selain itu banyak mahasiswa yang tidak lulus tepat waktu, hal ini mengakibatkan rasio dosen dan mahasiswa tidak seimbang.

Teknik data mining dan machine learning dapat digunakan untuk memprediksi berdasarkan data-data masa lalu. Metode *data mining* yang digunakan untuk memprediksi adalah klasifikasi. Penelitian ini dilaksanakan dengan cara menganalisa model yang diusulkan dengan

menerapkan pada dataset ketepatan waktu kelulusan mahasiswa. Data yang dikumpulkan adalah data alumni dan mahasiswa yang masa studinya lebih dari 4 tahun (8 semester). Pada penelitian diusulkan model prediksi ketepatan waktu lulusan mahasiswa menggunakan beberapa teknik data mining, yaitu 10 algoritma klasifikasi. Untuk menguji model yang diusulkan digunakan teknik validasi *10-fold cross validation*. Pengukuran kinerja model dilakukan dengan dilakukan menggunakan *confusion matrix*. *Confusion matrix* diperoleh dari proses validasi menggunakan *10-fold cross validation*, sehingga model yang terbentuk dapat langsung diuji dengan melakukan 10 kali pengujian. Model disimulasikan menggunakan software RapidMiner Studio 6.0.001 starter edition dan dataset mahasiswa yang telah dikumpulkan. Dari hasil pengukuran diperoleh model terbaik yaitu *Support Vector Machine* (SVM) dengan akurasi 65.00%. Tetapi akurasi ini masih jauh dari nilai *excellent* (sangat baik).

F. Mohammed Hamza Momade, Shamsuddin Shahid, Mohd Rosli bin Hainin, Mohamed Salem Nashwan & Abdulhakim Tahir Umar (2020)

Pada penelitian yang dilakukan oleh Mohammed Hamza Momade dkk yang berjudul "Modelling labour productivity using SVM and RF: a comparative study on classifiers performance (2020)". Penelitian dilakukan untuk mengusulkan pendekatan berbasis data untuk persiapan produktivitas tenaga kerja konstruksi atau biasa disebut CLP (*Construction Labour Productivity*) yang menjadi model dari pengaruh faktor tenaga kerja. Penelitian ini menggunakan dua metode klasifikasi berbasis *machine*

learning, yaitu *Support Vector Machine* (SVM) dan *Random Forest* (RF) nantinya digunakan untuk pemodelan dalam CLP. Faktor paling berpengaruh yang diidentifikasi adalah kurangnya tenaga kerja, pengalaman kerja, kategori pekerjaan, pendidikan / pelatihan, kebangsaan, keterampilan, usia dan status perkawinan. Data didapatkan dengan cara menyebar kuisioner yang disiapkan untuk mendapatkan respons dari pekerja konstruksi berdasarkan faktor-faktor berpengaruh yang diidentifikasi. Data dikumpulkan dengan total 220 pekerja konstruksi yang mewakili tiga kebangsaan yang berbeda seperti 41 Bangladesh, 85 Malaysia dan 94 Indonesia.

Untuk mengatasi kesulitan ini, sebuah data Metode clustering yang dikenal sebagai Jenks optimization, digunakan dalam penelitian ini untuk klasifikasi faktor indeks kepentingan. Metode Optimasi Jenks digunakan untuk membangun kelas dan untuk mengurutkan faktor berdasarkan indeks kepentingannya. Perbandingan model kinerja yang dilakukan dengan hasil yang lebih tinggi tingkat akurasi prediksinya untuk model CLP adalah menggunakan SVM dibandingkan dengan RF. SVM juga menunjukkan tingkat yang lebih tinggi akurasi dibandingkan dengan RF dalam hal relatif yang diperkirakan frekuensi. Oleh karena itu, model SVM dapat digunakan untuk prediksi produktivitas tenaga kerja konstruksi.

2.2. Keaslian Penelitian

Tabel 2. 1 Matriks literatur review ANALISIS KOMPARASI ALGORITMA SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS (Studi Kasus: Universitas Darussalam Gontor)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
1.	Klasifikasi Ketepatan Lama Studi Mahasiswa Menggunakan Metode *Support Vector Machine* Dan *Random Forest (Studi Kasus : Data Lama Studi Alumni Universitas Islam Indonesia Tahun Kelulusan 2000-2017)*	Syauqi Amri Yahya (2018)	Dibuat untuk mengetahui lama Studi Mahasiswa di Universitas Islam Indonesia dengan membandingkan metode klasifikasi *Random Forest dan Support Vector Machine*	Metode yang digunakan adalah *Kernel Radian Basis Function* (RBF). Metode ini memerlukan parameter *cost* (C) dan *gamma* yang nilailnya ditentukan oleh peneliti. Klasifikasi metode *Random Forest*, dilakukan dengan banyak pohon yang terbentuk (*ntree*). Data *testing* dilakukan, dengan *confusion matrix*	Pada penelitian belum menggunakan metode uji coba validasi, uji coba hanya menggunakan bantuan dari software RStudio Saran untuk penelitian ini agar mendapatkan data yang akurat, agar melakukan uji coba validasi menggunakan *k-fold validation*.	Metode uji coba yang akan digunakan pada penelitian ini menggunakan uji coba validasi *k-fold validation* yang akan di uji sebanyak 10 kali ujicoba. Untuk uji coba data testing klasifikasi pada SVM menggunakan kernel *Polynomial*.
2.	Komparasi Algoritma C4.5, *Naive Beyes*, dan *Random Forest* Untuk Klasifikasi Data Kelulusan Mahasiswa Jakarta	Ibnu Alfarobi, Taransa Agasya Tutupoly, dan Ade Suryanto (2018)	Untuk mengetahui tingkat kelulusan mahasiswa di Jakarta dengan membagi data testing dan data training untuk mengkomparasi algoritma C4.5, *Naive Bayes*, dan *Random Forest*.	Data yang digunakan sebanyak 379 data. Pengujian menggunakan *k-fold cross validation*. Pengukuran klasifikasi dengan menggunakan ROC *curve*,	Belum adanya proses *Text Representation* untuk menghitung bobot kalimat yang penting. Saran untuk penelitian ini agar menggunakan operator optimasi seperti *Particle Swarm Optimization* (PSO).	Metode penghitungan bobot kalimat pada data penelitian ini menggunakan metode klasifikasi TF-IDF (*Term Frequency – Inverse Document Frequency*).

Tabel 2.1 Matriks literatur review ANALISIS KOMPARASI ALGORITMA SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS (Studi Kasus: Universitas Darussalam Gontor) (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
3.	A Comparative Assessment of *Random Forest* and SVM Algorithms, Using Combination of Principal Component Analysis and SMOTE For Accounts Receivable Seamless Prediction, case study company X in Surabaya	Ferry Irawan dan Febriliyan Samopa (2018)	Membandingkan akurasi algoritma *Support Vector Machine* (SVM) dan *Random Forest* untuk pengujian prediksi kelancaran piutang pelanggan di suatu perusahaan swasta selama tiga tahun.	Proses pengumpulan data dan preparasi dihasilkan 10.230 data, labeling dilakukan dengan metode expert judgement, motede klasifikasi menggunakan *feature selection*. Sedangkan akurasi dari algoritma SVM menggunakan kernel *polynomialrix*	Belum ada metode untuk pengolahan data training, maka disarankan untuk menghasilkan rules yang lebih akurat pada penelitian ini perlu melakukan metode *Text Preprocessing* dan *Text Representation.*	Tahapan pada penelitian yang akan dilakukan mempunyai 5 tahapan yaitu membangun dataset, *Text Preprocessing*, *Feature Selection*, *Text Representation*, *Application of Text Mining Techniques.*
4.	Perbandingan Kinerja Algoritma K-Nearest Neighbor, Naïve Bayes Classifier, dan *Support Vector Machine* dalam Klasifikasi Tingkah Laku Bully pada Aplikasi Whatsapp	Irwansyah Saputra Dan Didi Rosiyadi (2019)	Menerapkan metode klasifikasi untuk mengetahui tingkah laku bully pada aplikasi whatsapp, mengklasifikasikan algoritma *K-Nearest Neighbor*, *Naïve Bayes Classifier* dan *Support Vector Machine*.	Metode pada penelitian ini menggunakan model KDD (*Knowledge Discovery in Database*). Dataset yang digunakan terdiri dari atribut teks dan kelas yang akan dilabeli menggunakan teknik *crowdsourcing* Label terdiri dari dua jenis, yaitu bully dan tidak bully.	Belum adanya proses *Text Representation* untuk menghitung bobot kalimat yang penting.	Metode penghitungan bobot kalimat pada data penelitian ini menggunakan metode klasifikasi TF-IDF (*Term Frequency – Inverse Document Frequency*). Proses konversi data menjadi matrix dan pembobotan ini dibantu dengan menggunakan *liblary python sklearn tfid vectorize.*

Tabel 2.1 Matriks literatur review ANALISIS KOMPARASI ALGORITMA SUPPORT VECTOR MACHINE DAN RANDOM FOREST UNTUK KLASIFIKASI TINGKAT PELANGGARAN MAHASISWA DI UNIVERSITAS (Studi Kasus: Universitas Darussalam Gontor) (Lanjutan)

No	Judul	Peneliti, Media Publikasi, dan Tahun	Tujuan Penelitian	Kesimpulan	Saran atau Kelemahan	Perbandingan
5.	Metode Data Mining Untuk Seleksi Calon Mahasiswa Pada Penerimaan Mahasiswa Baru Di Universitas Pamulang	Aries Saifudin (2018)	Penelitian ini dilakukan untuk seleksi calon mahasiswa pada penerimaan mahasiswa baru.	Pengujian model yang digunakan teknik validasi *10-fold cross validation*. Pengukuran kinerja model dilakukan dengan dilakukan menggunakan *confusion matrix*. Dari hasil pengukuran diperoleh model terbaik yaitu *Support Vector Machine* (SVM) dengan akurasi 65.00%.	Data set hanya diolah dalam bentuk CSV. Belum terbagi antara data trainig dan data testing sehingga Belum diterapkan metode untuk mengolah dataset, disarankan untuk menghasilkan rules yang lebih akurat pada penelitian ini perlu melakukan metode *Text Preprocessing* dan *Text Representation*.	Untuk mendapat data training dan data testing yang akurat maka pada penelitian ini dilakukan 5 tahapan yaitu membangun dataset, *Text Preprocessing*, *Feature Selection*, *Text Representation*, *Application of Text Mining Techniques*.
6.	Modelling labour productivity using SVM and RF: a comparative study on classifiers performance .	Mohammed Hamza Momade, Shamsuddin Shahid, Mohd Rosli bin Hainin, Mohamed Salem Nashwan & Abdulhakim Tahir Umar (2020)	Penelitian ini dilakukan untuk mengusulkan pendekatan berbasis data untuk persiapan produktivitas tenaga kerja konstruksi yang mana menjadi model dari pengaruh faktor tenaga kerja.	Penelitian ini menggunakan SVM dan RF. Data didapatkan dengan menyebar kuisioner. Metode klasifikasi yg digunakan adalah Metode Optimasi Jenks. SVM menunjukkan tingkat yang lebih tinggi akurasi dibanding dengan RF.	Data sampel yang digunakan pada penelitian ini sejumlah 220 peserta yang diambil dari kuisioner jadi data dirasa kurang banyak. Belum terbagi antara data trainig dan data testing sehingga Belum diterapkan metode untuk mengolah dataset. Uji coba belum menggunakan tools software.	Data sampel yang digunakan pada penelitian ini sejumlah 983 data. Uji coba pada penelitian ini pemrosesan data menggunakan aplikasi *google collabs* dan aplikasi *pycharm*, *google collabs* .

2.3. Landasan Teori

2.3.1 Data Mining

Data mining merupakan proses untuk menemukan hubungan dalam data yang tidak diketahui oleh pengguna dan menyajikannya dengan cara yang dapat dipahami, sehingga hubungan tersebut dapat menjadi dasar pengambilan keputusan (Jr.McLeod.R dan Schell, 2007). Data mining juga sebuah proses untuk menggali nilai tambah berupa informasi yang selama ini tidak diketahui secara manual dari suatu basis data dengan cara melakukan penggalian pola-pola dari data, dengan tujuan untuk bisa melakukan manipulasi data menjadi informasi berharga yang diperoleh dengan cara mengenali pola penting atau menarik dari data yang terdapat dalam basis data (Suyadi, 2017).

Salah satu metode data mining adalah dengan klasifikasi. Metode klasifikasi memiliki beberapa algoritma dan setiap algoritma klasifikasi pada data mining pastinya memiliki kelebihan dan kekurangan, sehingga penelitian yang dilakukan juga digunakan untuk menganalisa perbandingan diantara algoritma tersebut. Pengukuran kinerja algoritma data mining dapat dilakukan berdasarkan kriteria antara lain melalui keakuratan, kesempurnaan, konsistensi, kecepatan, dapat dipercaya dan interpretabilitas (Sari Dewi, 2016).

2.3.2 Text Mining

Text mining merupakan teknologi baru yang digunakan untuk data dalam jumlah besar yang selalu bertambah sehingga data teks yang tidak terstruktur tersebut dapat dianalisis. *Text mining* bekerja pada data yang tidak terstruktur atau semi testruktur seperti email, dokumen text lengkap, html dan lain-lain. Sehingga

text mining dianggap sebagai sebuah penemuan teknologi baru dari informasi yang belum diketahui dengan mengekstrak informasi dari sumber yang tertulis (Syaifudin.K, Hasbi.Y, dan Moch. A. Mukid, 2016).

Kategori yang termasuk didalam teknik text mining salah satunya adalah analisis tingkah laku, yaitu proses untuk memahami, mengekstrak, dan mengolah data tekstual secara otomatis, atau merupakan studi komputasi pendapat dan tingkah laku yang dinyakan dalam bentuk teks. Ada beberapa algoritma atau metode yang di gunakan untuk analisis tingkah laku, antara lain *Naïve Bayes* (NB), *Decision Tree*, dan *clustering K-Means* (Setyo Budi, 2017). Didalam penelitian ini algoritma yang digunakan adalah algoritma C4.5, *Naive Bayes*, dan *Random Forest* dengan metode TF-IDF (*Term Frequency – Inverse Document Frequency*). Metode ini adalah suatu metode algoritma yang berguna untuk menghitung bobot setiap kata yang umum digunakan. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat.

2.3.3 Klasifikasi

Klasifikasi merupakan kegiatan yang mengelompokkan suatu benda yang memiliki beberapa ciri yang sama dan memisahkannya dengan benda yang tidak sama. Pada dasarnya di dunia perpustakaan dikenal ada 2 (dua) jenis kegiatan klasifikasi yaitu Klasifikasi Fundamental (*Fundamental Classification*) dan Klasifikasi Artifisial (*Artificial Classification*). Klasifikasi Fundamental (*Fundamental Classification*) adalah klasifikasi bahan pustaka yang diambil berdasarkan subyek/isi buku, karena pemakai perpustakaan lebih banyak mencari informasi tentang subyek tertentu. Klasifikasi Artifisial (*Artificial Classification*),

yaitu klasifikasi bahan pustaka yang diambil berdasarkan ciri-ciri yang ada pada bahan pustaka (Suyadi, 2017).

Dalam klasifikasi terdapat dua pekerjaan utama yang harus dilakukan, yaitu pembangunan model sebagai prototipe yang hasilnya disimpan sebagai memori dan penggunaan model untuk melakukan pengenalan/klasifikasi/prediksi pada suatu objek, agar data lain diketahui berasal dari kelas mana objek datanya didapatkan (Dwi.L.Arisandy, Victor.W, dan Yeni.D.Rahayu, 2016).

2.3.4 *Random Forest*

Random Forest merupakan sebuah metode klasifikasi pengembangan dari *Decision Tree*. Sesuai dengan namanya, metode ini menciptakan sebuah hutan (*forest*) dengan sejumlah pohon (*tree*). Analogi yang bisa digunakan dalam penyebutan *Random Forest* kerena semakin banyak pohon (*tree*) pada hutan (*forest*) maka akan semakin kuat hutan terlihat. Jika diimplementasikan pada sebuah kasus, apabila semakin banyak pohon (*tree*), maka akan semakin besar pula akurasi yang didapatkan (Reinardus.A.Haristu, 2019). Banyaknya pohon yang akan dibentuk sangat berpengaruh terhadap tingkat akurasi hasil klasifikasi. Semakin banyak pohon, semakin akurat hasil klasifikasinya. Selain itu juga RF dapat menangani input variabel yang besar, menyeimbangkan *error* dalam *unbalanced dataset*. Pohon keputusan dimulai dengan cara menghitung nilai *entropy* sebagai penentu tingkat ketidak murnian atribut dan nilai *information gain*. Untuk menghitung nilai *entropy* digunakan rumus seperti pada persamaan 1, sedangkan nilai *information gain* menggunakan persamaan 2 (Yusuf S. Nugroho dan Nova Emiliyawati, 2017).

$$Entrophy = -\sum \quad i\, p(c|Y)\, log2p\, (c|Y)$$

Dimana Y adalah himpunan kasus dan p(c|Y) merupakan proporsi nilai Y terhadap kelas c.

$$Information\ Gain\ (Y, a)$$

$$= Entropy\ (Y) - \sum \quad v \epsilon Values(a) \frac{|Yv|}{|Ya|} Entropy\ (Yv)$$

Dimana *Values(a)* merupakan semua nilai yang mungkin dalam himpunan kasus *a*. *Yv* adalah subkelas dari *Y* dengankelas *v* yang berhubungan dengan kelas *a*. *Y* adalah semua nilai yang sesuai dengan *a*.

Untuk penerapan algoritma *Random Forest*, diperlukan algoritma untuk membangun *tree*. Salah satu algoritma yang digunakan adalah penggunaan algoritma *Classification and Regression Tree* (CART). CART membagi pohon keputusan dengan teknik *binary tree* dan dapat diterapkan pada variabel numerik dan kategoris sekaligus dibuktikan, bahwa CART cocok diterapkan untuk data dengan variabel yang banyak dan kompleks (Fransiska.A.Kurniawan dan Angelina.P.Kurniati, 2011).

2.3.5 *Support Vector Machine* (SVM)

SVM merupakan salah satu dari metode yang dikembangkan untuk mengatasi permasalah yang tidak bisa diselesaikan dengan metode statistika klasik, terutama pada kasus klasifikasi dan prediksi. SVM salah satu teknik yang relatif baru dibandingkan dengan teknik lain, tetapi memiliki performansi yang lebih baik di berbagai bidang aplikasi seperti *bioinformatics*, pengenalan tulisan tangan, klasifikasi teks dan lain sebagainya (Bendi.V.Ramana1, Prof.

M.Surendra.P.Babu, Prof. N. B. Venkateswarlu, 2011). Dalam SVM, untuk memisahkan data terhadap kelasnya, SVM membangun sebuah *hyperplane* (bidang pemisah). *Hyperplane* (bidang pemisah) yang baik, bukan hanya *hyperplane* yang bisa digunakan untuk memisahkan data, akan tetapi *hyperplane* yang baik adalah *hyperplane* yang memiliki batasan (*margin*) yang paling besar (Munawarah dan Raudlatul, 2016).

SVM dikembangkan dengan prinsip *linier classifier*. Namun dalam kasus nyata sering dijumpai data yang tidak linier sehingga dikembangkan SVM untuk kasus nonlinier dengan memasukkan konsep kernel. Dengan begitu, ada jaminan bahwa klasifikasi menggunakan SVM akan menghasilkan pemetaaan yang sangat akurat (Muhammad.I.Fachruddin, 2015). Proses pembelajaran SVM adalah untuk menentukan *support vector*, kita hanya cukup mengetahui fungsi kernel yang dipakai, dan tidak perlu mengetahui wujud dari fungsi *non-linear*.Persamaan SVM :

$$f(x) = w^t \Phi(x) + b$$

Yang mana :

$\boldsymbol{b}$ = Bias

$\boldsymbol{x} = (\boldsymbol{x1}, \boldsymbol{x2}, \ldots\ldots\boldsymbol{xD})\boldsymbol{T}$ = Variabel Input

$\boldsymbol{w} = (\boldsymbol{w0}, \boldsymbol{w1}, \ldots\ldots\boldsymbol{wD})\boldsymbol{T}$ = Parameter Bobot

$\phi(\boldsymbol{x})$ = Fungsi Transformasi fitur

SVM yang merupakan algoritma yang memiliki kelas metode kernel, yang berakar pada teori belajar statistik. Kernel berfungsi sebagai dasar pembelajaran semua algoritma, algoritma ini bertujuan umum masalah fungsi kernel tertentu.

Karena mesin *linear* hanya dapat mengklasifikasi data dalam *linear* ruang fitur terpisah. Fungsi peran kernel untuk mendorong sebuah ruang fitur oleh implisit pemetaan data pelatihan kedalam ruang dimensi yang lebih tinggi dimana data adalah *linear* terpisah. Tujuan dari SVM adalah untuk merancang cara pembelajaran komputasi yang efisien dalam pemisahan *hyperplane* didalam ruang fitur berdimensi tinggi (Ivo Colanus Rally Drajana, 2017). Dalam algoritma SVM ada *trik kernel* dimana ada SVM *linear* dan SVM *nonlinear*. Dimana SVM adalah *hyperplane linear* yang bekerja hanya pada data yang hanya dapat dipisahkan dengan cara*linear*. SVM *nonlinear* yaitu data yang berdistribusi pada kelas yang tidak *linear* sering digunakan pendekatan kernel pada fitur awal data set. Dimana kernel dapat diartikan sebagai suatu fungsi yang memetakan fitur data yang memiliki dimensi awal rendah fitur lainya yang berdimensi lebih tinggi bahkan jauh lebih tinggi. Masalah data yang sifatnya tidak *linear*, kita memerlukan penggunaan fungsi kernel (Eko Prasetyo, 2012). Kernel yang akan digunakan pada penelitian ini menggunakan Fungsi kernel *Polynomial:*

$$K\,(x, y) = (x.\,y + c)^d$$

Dimana :

$(\boldsymbol{x}, \boldsymbol{y})$ = Vektor Input

$\boldsymbol{c}$ = Variabel

$\boldsymbol{d}$ = Derajat *polynomial*

Melihat dari konsep metode *Support Vector Machine* ini, muncul pemikiran apakah metode ini dapat digunakan untuk menganalisis seseorang yang akan

melanggar atau tidak. Hasil prediksi inilah yang menjadi bahan penelitian dengan menggunakan metode *machine learning,* yaitu metode *Support Vector Machine*.

2.3.6 Multiclass Classification Support Vector Machine

Merupakan proses klasifikasi yang memiliki jumlah kelas lebih dari 2. Sementara *Support Vector Machine* (SVM) yang standar dibuat untuk proses *two class classification*. Seiring dengan berkembangnya dalam penggunaan algoritma tersebut, *Support Vector Machine* (SVM) mengembangkan metode pengklasifikasian dengan masalah lebih dari 2 kelas, metode tersebut yaitu:

1. Metode *One-vs-The Rest*, yaitu membangun *K* buah *Support Vector Machine* (SVM), dimana model ke-k, yaitu *yk(x)*, dilatih dengan menggunakan data dari kelas *Ck* sebagai sampel positif (+1) dan data dari kelas yang lain sebagai sampel negatip (-1). Seperti contoh: *y1(x)* akan memisahkan antara kelas 0 dan kelas – kelas lainnya (1,2,3,4).
2. Metode *One-vs-One*, yaitu membangun *K(K-1)/2* buah *Support Vector Machine* (SVM) yang merupakan semua kemungkinan pasangan kelas, selanjutnya suatu data penguji akan diklasifikasikan ke kelas yang menang paling banyak. sebagai contoh: *y1(x)* akan memisahkan kelas 0 dan kelas 1, *y2(x)* akan memisahkan kelas 1 dan kelas 2, dst

2.3.7 Term Frequency Inverse Document Frequency (TF-IDF)

Algoritma *Term Frequency Inverse-Document Frequency* merupakan suatu algoritma yang menggabungkan antara *Term frequency* dengan *Inverse Document Frequency*. *Term frequency* yaitu jumlah kemunculan sebuah term pada sebuah

dokumen. *Inverse Document Frequency* yaitu pengurangan dominasi term yang sering muncul diberbagai dokumen, dengan memperhitungkan kebalikan frekuensi dokumen yang mengandung suatu kata (Musfiroh.N, Hamdani, Inda.F.Astuti, 2013). TF-IDF merupakan metode untuk menghitung bobot setiap kata yang paling umum digunakan pada information retrieval. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat (Abdul.A.Maarif, 2015).

Metode ini menggabungkan dua konsep untuk perhitungan bobot, yaitu frekuensi kemunculan sebuah kata di dalam sebuah dokumen tertentu dan *inverse* frekuensi dokumen yang mengandung kata tersebut. Frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata itu di dalam dokumen tersebut. Frekuensi dokumen yang mengandung kata tersebut menunjukkan seberapa umum kata tersebut. Sehingga bobot hubungan antara sebuah kata dan sebuah dokumen akan tinggi apabila frekuensi kata tersebut tinggi di dalam dokumen dan frekuensi keseluruhan dokumen yang mengandung kata tersebut yang rendah pada kumpulan dokumen (Musfiroh Nurjannah, Hamdani, dan Inda Fitri Astuti 2013). Rumus untuk TF-IDF:

$$tf = 0{,}5 + 0{,}5\ X \frac{tf}{(tf)}$$

$$idf = log(\frac{D}{df})$$

$$W_{d,t} = tf_{d,t}\ X\ IDF_{d,t}$$

Keterangan:

tf = banyaknya kata yang dicari pada sebuah dokumen

max (*tf*) = jumlah kemunculan terbanyak *term*

pada dokumen yang sama.

Nilai D = total dokumen

df_t = jumlah dokumen yang mengandung *term* t.

IDF = Inversed Document Frequency (log2

(D/df))

d = dokumen ke-d

t = kata ke-t dari kata kunci

W = bobot dokumen ke-d terhadap kata ke-t

TF-IDF juga merupakan sebuah ukuran statistik yang digunakan untuk mengevaluasi seberapa penting sebuah kata di dalam sebuah dokumen atau dalam sekelompok kata. Untuk dokumen tunggal tiap kalimat dianggap sebagai dokumen. Frekuensi kemunculan kata di dalam dokumen yang diberikan menunjukkan seberapa penting kata itu di dalam dokumen tersebut. Frekuensi dokumen yang mengandung kata tersebut menunjukkan seberapa umum kata tersebut. Bobot kata semakin besar jika sering muncul dalam suatu dokumen dan semakin kecil jika muncul dalam banyak dokumen (Auliaguntary.A. Putra dan Agung, 2016).

2.3.8 K-Fold Cross Validation

K-Fold Cross Validation merupakan salah satu metode yang digunakan untuk mengetahui rata-rata keberhasilan dari suatu sistem dengan cara melakukan perulangan dengan mengacak atribut masukan sehingga sistem tersebut teruji untuk beberapa atribut input yang acak (Pandie, Emerensye S. Y, 2012). *K-Fold Cross Validation* mengulang k-kali untuk membagi sebuah himpunan contoh

secara acak menjadi k subset yang saling bebas, setiap ulangan disisakan satu subset untuk pengujian dan subset lainnya untuk pelatihan (Trevor.H., Robert.T., dan Friedman.J , 2009). Dengan K=5 atau 10 digunakan untuk memperkirakan tingkat kesalahan yang terjadi, sebab data training pada setiap *fold* cukup berbeda dengan data training yang asli. Secara keseluruhan, 5 atau 10-*fold cross* validation sama-sama direkomendasikan dan disepakati. Menghitung nilai akurasi dapat dilakukan dengan menggunakan persamaan (Sandi.F. Rodiyansyah dan Edi.W, 2013).

K-Fold Cross Validation mengulang k-kali untuk membagi sebuah himpunan contoh secara acak menjadi k subset yang saling bebas, setiap ulangan disisakan satu subset untuk pengujian dan subset lainnya untuk pelatihan. Dengan K=5 atau 10 dapat digunakan untuk memperkirakan tingkat kesalahan yang terjadi, sebab data *training* pada setiap *fold* cukup berbeda dengan data training yang asli. Secara keseluruhan, 5 atau 10-fold cross validation sama-sama direkomendasikan dan disepakati bersama. Menghitung nilai akurasinya dapat dilakukan dengan menggunakan persamaan (Sandi.F. Rodiyansyah dan Edi.W, 2013) :

$$Akurasi = \frac{Jumlah\ klasifikasi\ benar}{Jumlah\ data\ uji}\ X100\%$$

Dengan percobaan *K* kali dari model, model dapat diketahui performanya. Dengan cara kerja membagi dataset menjadi *K* set data, atau data *training* dan *test* yang berbeda sehingga semua bagian dataset akan melewati proses *training* dan *test* secara bergantian.

2.3.9 Phyton

Merupakan bahasa pemrograman dinamis yang mendukung pemrograman berbasis objek. Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Prinsipnya Python diperoleh dan dipergunakan secara bebas (freeware), bahkan untuk kepentingan komersial. Lisensi Python tidak bertentangan, secara definisi *Open Source* maupun *General Public Licens*e (GPL). Lengkap dengan source codenya, debugger dan profiler, antarmuka yang terkandung di dalamnya untuk pelayanan antarmuka, fungsi sistem, GUI (antar muka pengguna grafis), dan basis datanya (Therzian.R.Perkasa, Helmy. W, dan Pauladie.S, 2014).

Hal yang membedakan Python dengan bahasa lain adalah dalam hal aturan penulisan kode program. Bahasa Python mendukung hampir di semua sistem operasi, bahkan untuk sistem operasi Linux, hampir semua distronya sudah menyertakan Python di dalamnya. Dengan kode yang simpel dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat (R.H. Sianipar dan Hamzan.W., 2015).

BAB III

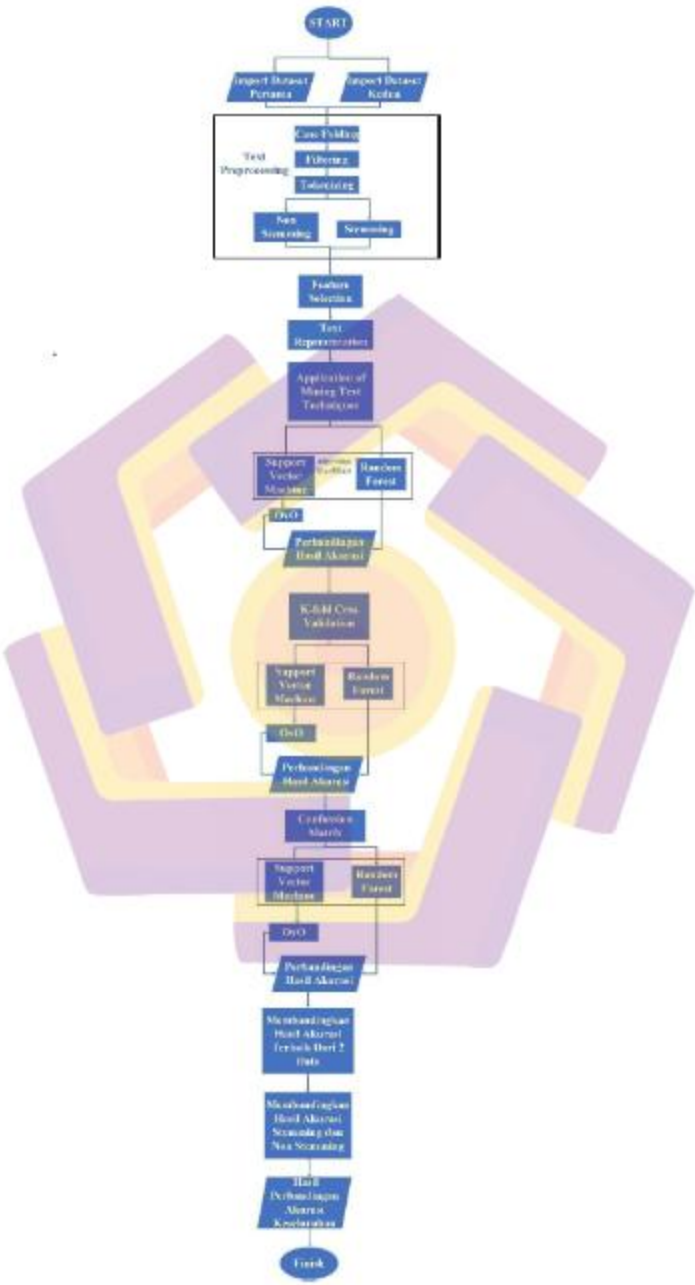
METODE PENELITIAN

Metodologi merupakan proses penguraian dari konsep menuju pembahasan bagian dalam yang lebih sederhana, sehingga susunan logikanya menjadi jelas. Metode untuk menguji dan menilai sistem kompleks dengan membagi unsur-unsur yang lebih sederhana sehingga hubungan antar unsur menjadi jelas (Natalina Br Sitepu, 2019). Metode penelitian yang digunakan mengkomparasikan 2 algoritma klasifikasi, menggunakan dua data dari sumber yang sama tapi dengan kuantitas dan kualitas berbeda yang telah memiliki label dan model, Pada data pertama berisi data yang masih lengkap sesuai dengan data bawaan dari sistem, data yang kedua berisi data yang telah dibersihkan dari data-data yang cacat atau tidak lengkap. Serta satu data pelanggaran baru yang belum memiliki label dan model.

Untuk membandingkan hasil analisa dilakukan pembedaan pada tahap *preprocessing*, yaitu dengan menyertakan tahap *stemming* dan dengan tidak menyertakan tahap *stemming*. Proses evaluasi kinerja model atau algoritma menggunakan uji coba validasi *k-fold validation* dari setiap masing masing algoritma. Namun kinerja suatu sistem yang melakukan klasifikasi tidak akan selalu bisa 100% benar. Oleh karena itu, sistem harus diukur kinerjanya. dengan menggunakan *confusion matrix* (Prasetyo, 2014). Untuk struktur penelitian tersusun seperti penjelasan dibawah ini :

3.1. Jenis, Sifat, dan Pendekatan Penelitian

Pada penelitian ini, eksperimen yang dilakukan bertujuan untuk mengetahui tingkat akurasi yang terbaik diantara dua algoritma. Untuk mengolah data yang ada, akan digunakan metode yang terlihat pada gambar 3.1 dibawah berikut:



Gambar 3. 1 Metode Penelitian

Untuk mengetahui hasil akurasi pengklasifikasian pelanggaran mahasiswa dalam penelitian ini menggunakan beberapa alur tahapan yaitu membangun dataset, yang mana penelitian ini menggunakan dua data dari sumber yang sama dengan kualitas data yang berbeda. Dilanjut dengan proses data training yaitu *Text Preprocessing* yang didalamnya memiliki 3 tahapan yaitu *case folding*, *filtering*, *tokenizing*, dan *stemming*, namun pada penelitian penulis ingin melakukan percobaan antara proses klasifikasi yang dilakukan *stemming* dengan yang tidak dilakukan *stemming*, untuk mengetahui perbandingan nilai akurasi yang didapat dari setiap algoritma. Dilanjutkan dengan masing-masing dataset dengan tahapan *Feature Selection*, *Text Representation*, *Application of Text Mining Techniques*.

Proses uji pertama adalah perhitungan dua algoritma yang digunakan yaitu *Support Vector Machine* dan *Random Forest*, yang mana dari kedua algotitma ini akan dilihat perbedaan performa akurasinya. Pada *Support Vector Machine* karena memiliki kelas lebih dari 2, maka ditambah optimalisasinya menggunakan metode OvO. Uji coba berikutnya dengan metode *K-fold Validation*. Serta dilanjutkan pada tahap evaluasi menggunakan *confussion matrix*. Setiap akhir dari uji coba masing-masing metode dilakukan yang namanya perbandingan hasil akurasi, dari data pertama dan data kedua, baik yang dilakukan *stemming* dan yang tidak dilakukan *stemming*. Tahap akhir perbandingan akurasi dari semua proses yang dilakukan denga kedua data yang digunakan. Berikut penjelasan dari masing-masing proses :

1. Membangun dataset

   a. Pengumpulan Data

      Pengambilan data dari website *https://simpel.unida.gontor.ac.id*. Data yang dikumpulkan diklasifikasikan atas 3 jenis pelanggaran yaitu ringan, sedang, dan berat.

   b. Labelisasi Data

      Tahap ini data yang telah terkumpul akan dilabeli berdasarkan 3 jenis pelanggaran yaitu ringan, sedang, dan berat. Labelisasi digunakan sebagai pembanding dan acuan dari data yang telah memiliki model. Kemudian data yang telah terlabeli divalidasi oleh sistem yaitu *https://simpel.unida.gontor.ac.id*.

2. *Text Preprocessing*

   Dataset yang sudah ada akan diubah ekstensinya dalam bentuk *comma separate value* (.csv). Data yang telah diubah tersebut kemudian dipanggil kedalam program dan kemudian dipisahkan antara fitur dan data target atau labelnya.

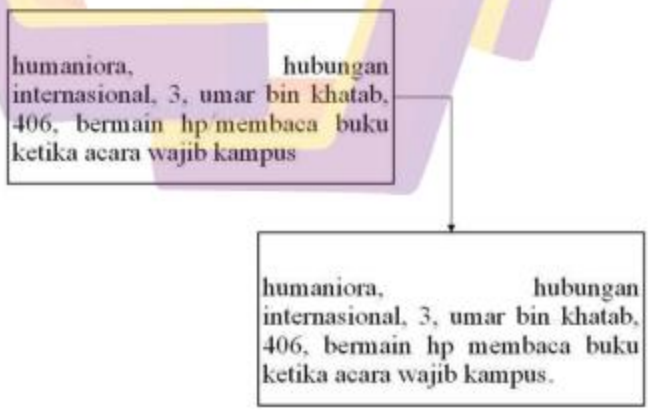
   a. *Case Folding*

      Tahap ini dataset yang memiliki karakteristik kapitalisasi hurup yang tidak beraturan akan diubah secara keseluruhan menjadi huruf kecil. Contoh penerapanya yaitu dapat dilihat pada Gambar 3.2

Gambar 3. 2 Proses *Case Folding*

b. *Filtering*

Tahap ini merupakan tahap untuk menghapus kata, simbol, dan angka, karena komponen tersebut tidak diperlukan. Hal ini membuat data lebih mudah untuk diproses dan membuat data bersih, contoh sederhana dari proses ini dapat dilihat pada Gambar 3.3.



Gambar 3. 3 Proses *Filtering*

c. *Tokenizing*

Dataset yang telah diubah menjadi kumpulan data berhuruf kecil, dan telah bersih dari kata yang tidak diperlukan, proses berikutnya adalah data yang berupa kalimat akan dipotong atau dipisah – pisahkan berdasarkan tiap kata yang menyusunnya, contoh proses *tokenizing* dapat dilihat pada Gambar 3.4

Gambar 3. 4 Proses *Tokenizing*

d. *Stemming*

Stemming merupakan tahap dalam mengembalikan kata yang telah dilakukan pemfilteran sehingga menjadi kata asalnya. Pada penelitian ini metode *stemming* yang digunakan adalah metode *stemming* untuk bahasa Indonesia yaitu algoritma Nazief – Adriani. Pada metode algoritma Nazief – Adriani ini memiliki tiga komponen, yaitu: pengelompokan imbuhan, urutan penggunaan aturan (*rule*) dan kamus (*dictionary*). Kamus akan dicek setiap penerapan aturan stemming berhasil diidentifikasi, dan apabila *stemming*

berhasil menemukan akar kata maka algoritma akan mengembalikan kata dalam kamus dan algoritma berhenti dengan langkah-langkah :

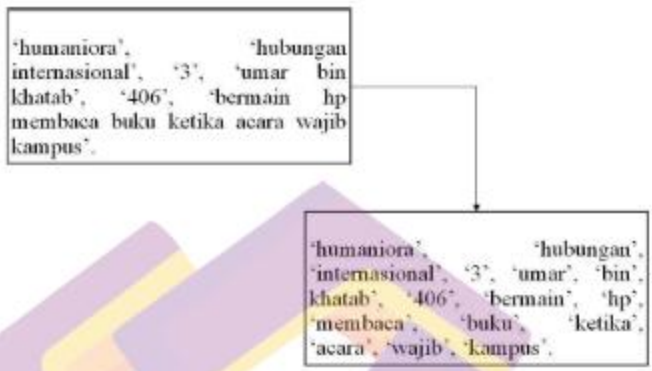
1. Kata yang akan dilakukan *stemm* dicari dalam kamus. Jika ditemukan kata yang dimaksud, maka dianggap kata tersebut adalah akar kata sehingga kata tersebut direturn dan algoritma stop di sini.
2. Hilangkan imbuhan infleksi ("-lah", "kah","-ku","-mu" dan "-nya"). Jika berhasil dan jika akhiran adalah partikel ("-lah" atau "-kah") dilanjut dengan menghilangkan imbuhan possesive ("-ku", "-mu" dan "nya").
3. Hilangkan imbuhan derivasi ("-i" atau "an"). Jika berhasil, lanjutkan ke langkah 4, jika tidak lakukan hal berikut ini:
    a. Jika "-an" dibuang dan huruf terakhir dari kata adalah "-k", maka "-k" juga dibuang dan pergi ke langkah 4.
    b. Penghilangan akhiran "-i", "-an" dan "kan" dibatalkan.
4. Penghilangan awalan dengan berbagai variasi.

Jika semua langkah telah ditempuh dan tidak berhasil, maka kembalikan kata asli yang belum distemm (Oppie Rezalina, 2016). Hal ini dikarenakan bahasa yang digunakan dalam data penelitian menggunakan bahasa indonesia.

5. *Feature Selection*

Proses ini merupakan proses untuk membuat data tekstual menjadi lebih bersih atau lebih sedikit sehingga membuat hasil *text mining* lebih baik dan efektif untuk digunakan. Proses yang dilakukan pada tahapan ini adalah *filtering* (*stopword removal*) yaitu menghilangkan kata-kata yang dianggap

tidak penting atau tidak menggambarkan isi dari sebuah kalimat, proses ini dapat kita lihat secara sederhana pada Gambar 3.5.



Gambar 3. 5 Proses *Feature Selection*

6. *Text Representation*

Merupakan tahapan merubah data tekstual menjadi representasi yang lebih mudah untuk diproses. Data akan diubah kedalam bentuk vektor, yaitu bentuk *array* 1 dimensi atau matrik 1 dimensi. Proses *text representation* secara sederhana dapat dilihat pada Gambar 3.6

Data	ringan	ringan	ringan	sedang	ringan	ringan	sedang	ringan	sedang
1	1	1	0	0	0	0	0	0	0
2	0	0	1	1	0	0	0	0	0
3	0	0	0	0	1	1	1	0	0
4	0	0	0	0	0	0	0	1	1

Gambar 3. 6 Proses *Text Representation*

7. *Application of Text Mining Techniques*

Data yang telah dilakukan pengolahan atau dibersihkan, pada proses berikutnya akan masuk kedalam proses penerapan algoritma *Support Vector Machine* untuk dilakukan klasifikasi yang mana 80% data akan melewati tahapan proses *training*. Dalam tahapan training data, data yang lebih dari 2 class maka akan masuk kedalam proses metode OvO (*one versus one strategy*), yaitu metode *Support Vector Machine* (SVM) untuk mengklasifikasikan data yang memiliki lebih dari 2 kelas, kemudian didukung dengan menggunakan *kernel linier*. Metode OvO mendekomposisi *K*-kelas menjadi *K(K-1)/2* biner, hal ini dilakukan untuk membandingkan satu persatu kelas dengan kelas lainnya. Setiap perbandingan tersebut dikerjakan oleh sebuah model secara keseluruhan yang mana terdapat *K(K-1)/2* sub model untuk pengklasifikasian.

Dalam klasifikasi One-vs-One, untuk dataset instans kelas-K, kita harus menghasilkan model pengklasifikasi biner K * (K-1) / 2. Dengan menggunakan pendekatan klasifikasi ini, dapat membagi kumpulan data utama menjadi satu kumpulan data untuk setiap kelas yang berlawanan dengan setiap kelas lainnya Dari penjelasan diatas dirincikan pada penelitian ini melakukan pengklasifikasian pelanggaran berdasarkan tingkatannya yaitu ringan, sedang, dan berat (K=3), sehingga perhitungannya adalah 3(3-1)/2 = 3 kumpulan data klasifikasi biner sebagai berikut:

- Pengklasifikasi 1: Ringan vs. Sedang
- Pengklasifikasi 2: Ringan vs. Berat

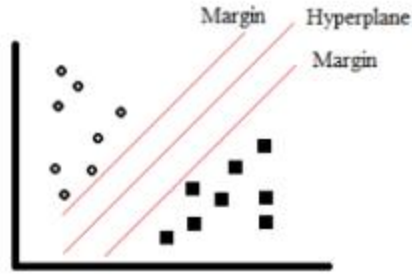
- Pengklasifikasi 3: Sedang vs. Berat

Setiap pengklasifikasi biner memprediksi satu label kelas. Saat kita memasukkan data uji ke pengklasifikasi, maka model dengan jumlah mayoritas disimpulkan sebagai hasilnya, sehingga terdapat 3 kelas biner dan kelas ini dapat dilihat pada Tabel 3.1

Tabel 3. 1 Dekomposisi kelas menggunakan OvO

3(3-1)/2=3	Kelas +1	Kelas -1
Kelas	1	2
Kelas	1	3
Kelas	2	3

Dari ke 3 kelas diatas akan menghasilkan klasifikasi sehingga untuk mencari hasil klasifikasi sebenarnya dipilih hasil yang paling banyak keluar. *Support Vector Machine* (SVM) merupakan algoritma untuk menemukan *hyperplane* yang paling terbaik guna memisahkan 2 buah objek, untuk mencari *hyperplane* tersebut menggunakan rumus $y(x) = wTx + wo = 0$ dan *margin* dengan rumus $wTx + wo = 1$ dan $wTx + wo = -1$ yang mana x merupakan *input* data dan wo adalah bias. Jadi *hyperplane* merupakan *decision boundary* yang memisahkan kelas -1 dengan kelas +1, pemetaan data dan pemisahannya digambarkan secara sederhana pada Gambar 3.7



Gambar 3. 7 Pemetaan *Support Vector Machine*

8. Uji coba dan Evaluasi

Model yang telah terbentuk dari proses *training* model akan dilakukan uji coba. Proses uji coba ini dilakukan dengan 2 cara yaitu dengan menggunakan data *testing* yang telah dibagi pada proses *training* dan menggunakan *K-Fold Cross Validation*. *Cross-validation* sebagai metode statistik yang digunakan untuk mengevaluasi kinerja model atau algoritma, dimana data dipisahkan menjadi dua subset yaitu data proses pembelajaran dan data validasi / evaluasi. Model atau algoritma dilatih oleh subset pembelajaran dan divalidasi oleh subset validasi. Selanjutnya pemilihan jenis *Cross-validation* dapat didasarkan pada ukuran dataset. Biasanya *K-Fold Cross Validation* digunakan karena dapat mengurangi waktu komputasi dengan tetap menjaga keakuratan estimasi.

Pada proses pertama, pengujian dilakukan dengan 20% dataset yang telah dipilih dari proses *training*, langsung diklasifikasikan kedalam model, sehingga model akan secara langsung menghasilkan hasil klasifikasinya dan dapat dihitung akurasi keefektifan dari model tersebut. Proses kedua, *K-Fold Cross Validation* bekerja dengan membagi data menjadi *K* bagian, sehingga

akan terjadi *K* kali pengujian dengan 1 bagian menjadi bagian data *test* dan sisanya menjadi data *training*.

Dalam penelitian ini proses *K – Fold Cross Validation* akan dilakukan sebanyak 10 Kali. Proses 10 *K – Fold Cross Validation* merupakan *K – Fold Cross Validation* yang direkomendasikan untuk pemilihan model terbaik pada penelitian ini karena bisa memberikan estimasi akurasi yang lebih baik dibandingkan dengan *Cross Validation* biasa seperti, *leave-one-out Cross Validation* dan *bootstrap*. Dalam 10 *K – Fold Cross Validation*, data dibagi menjadi 10 fold berukuran yang kira-kira sama, sehingga kita memiliki 10 subset data untuk mengevaluasi kinerja model atau algoritma. Untuk masing-masing dari 10 subset data tersebut, *Cross Validation* akan menggunakan 9 fold untuk pelatihan dan 1 fold untuk pengujian,

Data yang akan diuji dibagi menjadi sepuluh subset, seperti berikut : *fold* 1, *fold* 2, *fold* 3, *fold* 4, *fold* 5, *fold* 6, *fold* 7, *fold* 8, *fold* 9, dan *fold* 10. Hal ini dilakukan untuk mencari hasil terbaik. Pelatihan dilakukan secara berulang sebanyak sepuluh kali, setiap pengulangan sembilan *fold* akan dijadikan data latih dan satu *fold* untuk data uji. Proses ini dilakukan sampai semua *fold* pernah berperan sebagai data latih dan data uji. Berikut cara kerja k-fold cross validation pada penelitian ini :

a. Total *instance* dibagi menjadi N bagian.
b. Pada *fold* ke-1 merupakan bagian pertama yang menjadi data uji (*testing data*) dan sisanya menjadi data latih (*training data*). Tahap selanjutnya, menghitung akurasi berdasarkan porsi data tersebut.

c. *Fold* ke-2 merupakan bagian kedua yang menjadi data uji (*testing data*) dan sisanya menjadi data latih (*training data*). Tahap selanjutnya, menghitung akurasi berdasarkan porsi data tersebut.

d. Demikian seterusnya hingga mencapai *fold* ke-10. Menghitung rata-rata akurasi dari K buah akurasi diatas. Rata-rata akurasi akan menjadi akurasi akhir.

Proses pada pembagian data latih dan data uji dalam proses 10-*fold cross validation* dapat dilihat dari Tabel 3.2 dibawah ini

Tabel 3. 2 Pengujian menggunakan *K-Fold Cross Validation*

No.	**Pembagian Dataset**									
1	1	2	3	4	5	6	7	8	9	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	7	8	9	10
4	1	2	3	4	5	6	7	8	9	10
5	1	2	3	4	5	6	7	8	9	10
6	1	2	3	4	5	6	7	8	9	10
7	1	2	3	4	5	6	7	8	9	10
8	1	2	3	4	5	6	7	8	9	10
9	1	2	3	4	5	6	7	8	9	10
10	1	2	3	4	5	6	7	8	9	10

Keterangan	
	Nomor Pengujian
	Data Test
	Data Training

Proses selanjutnya yaitu evaluasi, evaluasi pada penelitian ini menggunakan metode *Confusion Matrix*. *Confusion matrix* dapat diartikan sebagai suatu alat yang memiliki fungsi untuk melakukan analisis apakah classifier tersebut baik dalam mengenali tuple dari kelas yang berbeda (Han & Kamber, 2011). Pada pengukuran kinerja menggunakan *confusion matrix*, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Nilai *True Negative* (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan *False Positive* (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, *True Positive* (TP) merupakan data positif yang terdeteksi benar. *False Negative* (FN) merupakan kebalikan dari *True Positive*, sehingga data posifit, namun terdeteksi sebagai data negatif. Pada jenis klasifikasi *binary* yang hanya memiliki 2 keluaran kelas, *confusion matrix* dapat disajikan seperti pada Tabel 3.3

Tabel 3. 3 Klasifikasi *multiclass* pada *confusion matrix*

Kelas	Klasifikasi Positif	Klasifikasi Negatif
Positif	TP (True Positive)	FN (False Negative)
Negatif	FP (False Positive)	TN (True Nagitive)

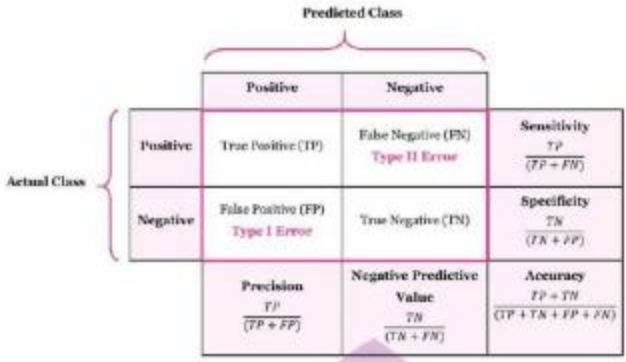
Dari tabel tersebut, ada empat macam kejadian yang bisa dijelaskan sebagai berikut :

a. **TP** (True Positive) = Jumlah data aktual yang sebenarnya positif diprediksi positif.
b. **TN** (True Negative) = Jumlah data aktual yang sebenarnya negatif diprediksi negative.
c. **FP** (False Positive) = Jumlah data aktual yang sebenarnya negatif diprediksi positif.
d. **FN** (False Negative) = Jumlah data aktual yang sebenarnya positif diprediksi negatif

Dan dari empat macam kejadian tersebut, nantinya akan digunakan sebagai acuan untuk metrik evaluasi sebagai berikut :

a. Akurasi/accuracy = TP / Jumlah Set Data.
b. Error Rate = (FP + FN) / Jumlah Set Data.
c. Precission = TP / (TP + FP).
d. Recall = TP / (TP + FN.
e. Spesifisitas = TN / (TN + FP)

Nilai dari *True - Positive* dan *True - Negative* memberi informasi ketika *classifier* melakukan klasifikasi data yang nilainya benar, sedangkan *False - Positive* dan *False - Negative* memberikan informasi ketika classifier melakukan klasifikasi yang salah dalam melakukan klasifikasi data (Mercury.F.Fibrianda dan Adhitya B., 2109). Ilustrasi untuk metode confusion matrix dapat dilihat pada Gambar 3.8



Gambar 3. 8 *Confusion Matrix*

Keluaran yang akan dihasilkan dari uji coba ini adalah *precission*, *recall* dan *accuracy* dari pengklasifikasian sehingga dapat diketahui seberapa baik model yang telah dibuat. *Accuracy* menggambarkan seberapa akurat model dapat mengklasifikasikan dengan benar. *Accuracy* merupakan rasio prediksi benar (positif dan negatif) dari keseluruhan data. Maka, *accuracy* merupakan tingkat kedekatan nilai prediksi dengan nilai aktual (sebenarnya). Pada *precision* menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model. *Precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif. Maka dilihat dari semua kelas positif yang telah di prediksi dengan benar, ada berapa banyak data yang benar-benar dianggap positif. *Recall* menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi. Maka, *recall* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif.

3.2. Metode Pengumpulan Data

Berdasarkan sumber data yang tersedia, data pada penelitian ini diperlukan sebagai landasan untuk menentukan teknik serta langkah pengumpulan data. Data yang digunakan untuk model penelitian diperoleh atau dikumpulkan oleh peneliti secara langsung dari sumber datanya yaitu pada Bidang Non Akademik UNIDA GONTOR dengan beberapa macam teknik, seperti observasi, wawancara, serta pengambilan data dari website *https://simpel.unida.gontor.ac.id*. Data yang diperlukan dalam penelitian ini adalah berupa data riwayat jenis pelanggaran mahasiswa yang berpengaruh terhadap pelanggaran mahasiswa.

3.3. Metode Analisis Data

Metode analisis data merupakan fase proses penelitian di mana data yang dikumpulkan diproses untuk menanggapi perumusan masalah. Setelah data-data yang penulis perlukan terkumpul, maka langkah selanjutnya adalah menganalisis data. Analisis data yang penulis gunakan pada penelitian ini menggunakan analisis kuantitatif. Analisis data kuantitatif yang digunakan pada penelitian ini untuk menyederhanakan data dalam bentuk yang lebih mudah dibaca dan diinterpretasikan. Berikut langkah-langkah sederhana untuk metode analisis penelitian ini :

a. Manajemen Data

   Fase manajemen data kuantitatif terdiri dari mentransformasikan set data mentah menjadi yang lebih matang, Hal ni dilakukan dengan cara menghapus data yang tidak penting atau disebut eliminasi. Eliminasi berarti bahwa peneliti "membersihkan" data mentah yang tidak relevan untuk diproses. Setelah menyisakan data yang penting atau matang, peneliti

melakukan manajemen data dalam bentuk file yang siap untuk dimasukkan. Dalam fase ini, peneliti juga memeriksa kualitas data, misalnya jika data hilang atau eror.

b. Entry Data

Input data kedalam sistem dilakukan untuk pemrosesan yang akan diterapkan. Saat memasukkan data, data dari alat penelitian seperti kuesioner dikirimkan ke perangkat lunak analisis. Jika data dalam bentuk kumpulan data, dalam arti bahwa peneliti tidak mengumpulkan data mentah mereka dengan kuesioner atau angket, data tersebut harus dihapus atau di filter. Pada penelitian ini penulis membuat uji coba dengan membandingkan dua data yang kualitas dan kuantitasnya berbeda akan tetapi pada dua data ini telah diberi label yang nantinya akan menjadi model untuk *training* dan *testing*. Serta data Ketiga merupakan data yang diperoleh atau dikumpulkan peneliti dari sumber yang sama akan tetapi datanya belum diberi label.

c. Implementasi Statistik

Setelah memasukkan data dalam perangkat lunak statistik, teknik statistik dapat diterapkan. Tahapan teknis menganalisis data statistik harus dikuasai oleh peneliti atau pemroses data. Analisis yang digunakan pada penelitian ini dibantu dengan penggunaan *tools* bahasa pemrograman *phyton. Output* dari analisis statistik melalui *phyton* dapat menunjukan struktur program dan aturannya.

3.4. Alur Penelitian

Gambar 3. 9 Alur Penelitian

Alur merupakan suatu penjabaran secara deskriptif tentang hal-hal yang akan ditulis, yang secara garis besar terdiri dari Bagian Awal, Bagian Isi dan Bagian akhir. Dalam prosedur format penelitian ini terdapat 3 hal utama yang menjadi unsur pembuatan karya tulis ini, yaitu bagian Awal yang meliputi studi pustaka yang mengkaji bahan-bahan yang akan digunakan untuk penelitian ini yang diambi dari buku bacaan, jurnal, literatur serta sumber lain yang terpercaya, setelah bahan-bahan yang dikumpulkan telah didapatkan masuk ke bagian pengumpulan data, data yang digunakan pada penelitian ini diambil dari website milik UNIDA sendiri.

Pada bagian kedua atau isi bagian ini merupakan paling banyak memiliki proses yang dijalankan yang mana didalamnya terdapat analisis dan perancangan dari penelitian yang akan dilakukan, dengan melakukan hal ini alur penelitian akan lebih terperinci dan mudah dipahami. Dilanjutkan dengan implementasi sistem yang akan dilakukan jika system yang dirancang telah disetujui termasuk program yang telah dibuat pada tahap perancangan sistem agar siap untuk dioperasikan. Untuk tahap implementasi ini peneliti melakukan uji coba menggunakan bahasa pemrograman *phyton*. Bahasa pemrograman *phyton* dipilih untuk pada penelitan ini

karena memiliki beberapa kelebihan tersendiri dibandingkan dengan bahasa pemrograman lainnya seperti php dan juga *javascript*.

Kelebihan dari bahasa pemrograman *python* yang pertama adalah sangat mudah untuk dipelajari, karena bahasa pemrograman ini memiliki sintaks yang cukup sederhana, sehingga sangat mudah untuk dipahami sekaligus dipelajari. Kelebihan kedua dari bahasa pemrograman Python ini adalah sangat mudah diaplikasikan dan diimplementasikan pada saat pembuatan website, Aplikasi Android, Software, hingga video game, karena *python* ini sangat kompatibel dengan beberapa sistem untuk mengembangkan software. kelebihan yang berikutnya dari bahasa pemrograman *python* adalah sangat fleksibel dan dapat digunakan pada beberapa sistem operasi seperti Windows, Unix, Mac OS, Ubuntu, dan sistem operasi lainnya. Pada penggunaan pemrograman *phyton* ada beberapa library yang akan digunakan pada penelitian ini, beberapa diataranya seperti :

a. *Pandas*

   Merupakan pustaka pembelajaran mesin dengan Python yang menyediakan struktur data tingkat tinggi dan beragam alat untuk melakukan analisis data. Salah satu fitur unggulan dari perpustakaan ini adalah kemampuannya untuk menerjemahkan operasi yang kompleks dengan mengandalkan data dari satu atau dua perintah saja.

b. *Scikit-learn*

   Pustaka ini berisi sejumlah algoritma untuk menerapkan pembelajaran mesin standar dan tugas-tugas penambangan data seperti mengurangi dimensi, klasifikasi, regresi, pengelompokan data, dan pemilihan model.

c. *NumPy*

Library Python yang fokus pada *scientific computing*. NumPy memiliki kemampuan untuk membentuk objek N-dimensional *array*, yang mirip dengan *list* pada Python. Keunggulan NumPy array dibandingkan dengan *list* pada Python adalah konsumsi *memory* yang lebih kecil serta *runtime* yang lebih cepat. NumPy juga memudahkan kita pada Aljabar Linear, terutama operasi pada Vector (1-d *array*) dan Matrix (2-d *array*).

d. *Matplotlib*

Merupakan library Python yang fokus pada visualisasi data seperti membuat *plot* grafik. Matplotlib pertama kali diciptakan oleh John D. Hunter dan sekarang telah dikelola oleh tim developer yang besar. Awalnya matplotlib dirancang untuk menghasilkan *plot* grafik yang sesuai pada publikasi jurnal atau artikel ilmiah. Matplotlib dapat digunakan dalam skrip Python, Python dan IPython *shell,* server aplikasi web, dan beberapa toolkit *graphical user interface* (GUI) lainnya.

e. Sastrawi

Python Sastrawi adalah pengembangan dari proyek PHP Sastrawi. *Python Sastrawi* merupakan library sederhana yang dapat mengubah kata berimbuhan bahasa Indonesia menjadi bentuk dasarnya. Sastrawi juga dapat diinstal melalui "pip".

f. NLTK

Natural Language Toolkit atau disingkat NLTK, adalah *libray* python untuk bekerja dengan permodelan teks. NLTK menyediakan alat yang baik guna mempersiapkan teks sebelum digunakan pada *machine learning* atau

algoritma *deep learning*. Cara termudah untuk menginstall NLTK adalah menggunakan "pip" pada *command line/terminal*.

Dilanjut pada tahap uji coba dengan tujuan program yang telah dibuat tidak memiliki kesalahan, selain itu juga dilakukan analisa untuk mengatahui kekurangan dari program yang telah dibuat. Terakhir pada bagian akhir yaitu tahapan penulisan laporan merupakan tahap akhir dari proses penelitian dimana peneliti menyampaikan dan mengkomunikasikan kepada audiens hasil akhir risetnya melalui tulisan. Dengan pengertian ini kita mengetahui bahwa penulisan laporan penelitian merupakan bagian penting dari rangkaian penelitian itu sendiri.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

Pada tahapan selanjutnya bab ke empat ini, penulis menyajikan hasil dan pembahasan penelitian mengenai hasil dari aplikasi *executable* yang telah dibuat dan dilakukan uji coba. Berikut merupakan hasil dan pembahasan yang terdiri dari tahapan aplikasi, pengujian, pembahasan, dan keakurasian sistem dengan metode yang telah dilakukan penerapannya. Penulis juga melakukan perbandingan terhadap kedua macam teknik klasifikasi tersebut melalui pengukuran akurasinya.

4.1 Hasil

Pada pengujian sistem diperlukan beberapa metode sehingga mendapatkan perbandingan hasil yang lebih efisien dalam melakukan analisa akurasi. Penulis melakukan tahapan yang mana pada bagian ini merupakan hasil dari proses membangun data, proses *text preprocessing*, *feature selection*, *text representation* dan kasifikasi dengan menggunakan *Support Vector Machine* dan *Random Forest*.

4.1.1 Membangun Dataset

Dalam penelitian ini penulis menggunakan data set dari platform sistem informasi UNIDA Gontor yang mendata seluruh kegiatan non akademik yang terdapat didalam kampus yaitu *https://simpel.unida.gontor.ac.id* untuk dari bulan Oktober 2019 sampai bulan Desember 2020, mengingat UNIDA merupakan universitas dengan sistem pesantren yang mana seluruh kegiatan non akademik harus terekam segala bentuk aktifitasnya. Beberapa data kegiatan non akademik yang terdapat pada *https://simpel.unida.gontor.ac.id* seperti data asrama untuk mahasiswa, data asal mahasiswa, data perizinan, dan data pelanggaran mahasiswa.

Pada penelitian ini penulis menggunakan uji coba pada data pelanggaran mahasiswa, karena bertujuan menganalisis seberapa kuat akurasi data-data pelanggaran yang dimiliki sistem yang ada dengan uji coba algoritma yang telah ditentukan serta melakukan klasifikasi terhadap pelanggaran-pelanggaran yang telah tercatat dalam sistem.

Uji coba ini penulis menggunakan dua data dari sumber yang sama tapi dengan kuantitas dan kualitas yang berbeda karena dari dua data tersebut yang membedakan adalah data pertama yang masih lengkap sesuai dengan data bawaan dari sistem, sedangkan data yang kedua berisi data yang telah dibersihkan dari data-data yang cacat atau tidak lengkap. Pada Dua data ini telah memiliki label yang nantinya akan menjadi model yang akan masuk kedalam proses *training* dan proses *testing*, dan akan ditambah satu data lagi berisikan data pelanggaran yang belum memiliki label, yang mana data ini belum dikenal modelnya oleh sistem, yang nantinya akan hanya digunakan untuk uji coba model klasifikasi.

Data didapat dengan melakukan export data dari *https://simpel.unida.gontor.ac.id* menjadi format .xls (*Extensible Stylesheet Language*) data tersebut akan di *copy* dan di *paste* ke *spreadsheet*. Kemudian data akan dilabeli sesuai dengan jenis pelanggaran yang tersirat dalam data. Data pertama yaitu data yang isinya belum diperbaiki yang memiliki jumlah data sebanyak 1.483, sedangkan data kedua berupa data yang sudah diperbaiki yang mana setelah dilakukan optimalisasi menjadi 1117 data. selanjutnya data akan diacak dan diubah kedalam bentuk .csv (*comma separate value*) agar dapat diproses mudah oleh sistem.

Data yang telah siap pakai akan masuk kedalam tahap proses inti yaitu proses *text preprocessing*. Dalam penelitian ini pemrosesan data menggunakan aplikasi *google collabs* dan aplikasi *pycharm*. Untuk *google collabs* sendiri digunakan penulis untuk proses training data yang memerlukan pemrosesan yang berat dan banyak agar uji coba dapat dilakukan dimana saja, sedangkan untuk penggunaan aplikasi *pycharm* digunakan untuk pengolahan berbasis offline. Kedua data ini nantinya akan dilihat seberapa besar perbedaan tingkat keakurasiannya jika diterapkan kedalam metode algoritma *Random Forest* (RF) dan *Support Vector Machine* (SVM).

4.1.2 *Text Preprocessing*

4.1.2.1 *Case Folding*

Proses pertama pada *preprocessing* merupakan tahapan dimana data diimport dan dibaca oleh sistem, setelahnya dibagi menjadi dua bagian yaitu kolom pertama pada data sebagai labels dan kolom ke 2 sebagai data. Bagian data merupakan bagian inti pengolahan dalam penelitian ini sehingga proses akan difokuskan kedalam bagian fitur ini. Data yang memiliki macam variasi ini harus diseleksi dengan baik dari segi susunan katanya hingga bentuk katanya sendiri. Penyeleksian data awal yang dilakukan adalah dengan cara mengubah semua huruf kapital menjadi huruf kecil semua tanpa terkecuali. Seperti pada contoh kalimat di bawah yang mana pada kalimat "PENDIDIKAN AGAMA ISLAM" yang mana semua hurufnya kapital, maka perlu diubah menjadi huruf kecil semua karena pada mesin bisa dianggap beda artian maka perlu dilakukan *case folding* agar makna bisa dipahami mesin. Proses ini menggunakan salah satu fitur dalam

python yaitu fitur *lower case* yang dituliskan dengan *syntax lower*(), berikut syntaks dari implementasi penggunaan *case folding* pada gambar 4.1

```
//implementasi penggunaan case folidng

# Untuk Membuat huruf KECIL SEMUA / LOWERCASE
fitur_ekstraksi0 = []
for cuitan in range(0, len(fitur)):
    tmp = str(fitur[cuitan]).lower()
    fitur_ekstraksi0.append(tmp)

print(*fitur_ekstraksi0[:10], sep='\n')
print('\n')
```

Gambar 4. 1 Syntak dari implementasi *case folding*

Hasil perbandingan dari proses sebelum *case folding* dan setelah *case folding* dapat dilihat dari tabel 4.1.

Tabel 4. 1 Proses *Case Folding*

Sebelum *Case Folding*	Setelah *Case Folding*
PENDIDIKAN AGAMA ISLAM, 10 ,ALI BIN ABI THALIB, 421, Mengoleksi Kontak Perempuan yang tidak dikenal	pendidikan agama islam, 10 ,ali bin abi thalib, 421, mengoleksi kontak perempuan yang tidak dikenal

4.1.2.2 *Filtering*

Tahapan proses selanjutnya adalah proses *filtering* yang merupakan proses penyeleksian pada tahap kedua dimana pada proses ini tahapan untuk menghilangkan karakter-karakter yang tidak diperlukan. Langkah pertama pada

proses adalah data akan dibersihkan dari semua karakter angka dan numerik. Angka pada data disini merupakan data kamar dan data semester, jadi tidak dihapus karena penting, yang mana ada keunikan dalam datanya dan angka diperlukan untuk korelasi data satu dengan yang lain seperti data prodi dengan data semester. Tahap akhir *filtering* adalah pembuangan karakter huruf yang tidak diperlukan seperti menghilangkan kata yang hanya satu karakter, menghilangkan spasi ganda, dan lainnya, seperti tanda koma pada contoh dibawah, yang mana menjadikan data lebih dari satu, maka harus dihilangkan agar data menjadi satu. Berikut syntaks dari implementasi penggunaan *filtering* pada gambar

```
//implementasi penggunaan filtering

fitur_ekstraksi2 = []
for cuitan in range(0, len(fitur_ekstraksi0)):
    tmp = fitur_ekstraksi0[cuitan].translate(str.maketrans(' ',' ', string.punctuation)) # membuang
karakter
    fitur_ekstraksi2.append(tmp)

print("fitur_ekstraksi2[910], sep='\n')
print("\n")

fitur_ekstraksi3 = []
# fitur_ekstraksi = []
for cuitan in range(0, len(fitur_ekstraksi2)):
    tmp = re.sub(r'\W', ' ',str(fitur_ekstraksi2[cuitan])) # membuang karakter khusus selain angka dan
huruf
    tmp = re.sub(r'\s+[a-zA-Z]\s+', ' ',str(fitur_ekstraksi2[cuitan])) # membuang kata yang hanya satu
huruf
    tmp = re.sub(r'\^[a-zA-Z]\s+', ' ',str(fitur_ekstraksi2[cuitan])) # membuang kata yang hanya satu
huruf dari awal
    tmp = re.sub(r'\s+', ' ',str(fitur_ekstraksi2[cuitan])) # menggantti spasi ganda dengan spasi tunggal
    fitur_ekstraksi3.append(tmp)
```

Gambar 4. 2 Syntak dari implementasi *filtering*

Hasil perbandingan dari proses sebelum *filtering* dengan setelah dilakukan *filtering* dapat dilihat pada Tabel 4.2

Tabel 4. 2 Proses *Filtering*

Sebelum *Filtering*	Setelah *Filtering*
pendidikan agama islam, 10 ,ali bin abi thalib, 421, mengoleksi kontak perempuan yang tidak dikenal	pendidikan agama islam 10 ali bin abi thalib 421 mengoleksi kontak perempuan yang tidak dikenal

4.1.2.3 *Tokenizing*

Tahapan *tokenizing* merupakan proses algoritma untuk memisahkan kalimat menjadi kata – perkata, proses ini menggunakan *natural language toolkit*. Tahapan *tokenizing* terfokus pada spasi pada data sehingga kata yang tidak sengaja, tidak dipisahkan oleh spasi dan dianggap sebagai satu kata saja. Tujuan dari proses *tokenizing* ini untuk membuat data menjadi *array* agar dapat diproses lebih mudah dalam proses selanjutnya. Berikut syntaks dari implementasi penggunaan *tokenizing* pada gambar 4.3

```
//implementasi penggunaan tokenizing

from nltk.tokenize import word_tokenize
|
fitur_ekstraksi5 = []
for cuitan in range(0, len(fitur_ekstraksi4)):
    tmp = word_tokenize(str(fitur_ekstraksi4[cuitan]))
    fitur_ekstraksi5.append(tmp)

print(*fitur_ekstraksi5[:10], sep='\n')
print('\n')
```

Gambar 4. 3 Syntak dari implementasi *tokenizing*

Hasil perbandingan dari proses sebelum *tokenizing* dan setelah *tokenizing* dilihat pada Tabel 4.3

Tabel 4. 3 Proses *Tokenizing*

Sebelum *Tokenizing*	Setelah *Tokenizing*
pendidikan agama islam 10 ali bin abi thalib 421 menyimpan video mengoleksi kontak perempuan yang tidak dikenal	'pendidikan','agama','islam', '10', 'ali', 'bin', 'abi', 'thalib','421','mengoleksi','kontak','perempuan', 'yang', 'tidak', 'dikenal'

4.1.2.4 *Stemming*

Pengklasifikasian data yang lebih baik apabila yang memiliki imbuhan dihilangkan, pada awal maupun akhir pada kata dibuat menjadi bentuk awal atau kata dasar. Proses *stemming* ini, merupakan proses yang digunakan untuk algoritma *stemming* bahasa Indonesia. Metode *stemming* yang digunakan dalam proses ini adalah metode *stemming* yang dibuat oleh Nazief – Adriani dengan menggunakan *liblary* python Sastrawi. Seperti yang telah dijelaskan di bab 3 pada metode penelitian metode *stemming* Nazief – Adriani memiliki tambahan bentuk awalan yang ditentukan dengan langkah berikut :

1. Jika awalannya adalah: "di-", "ke-", atau "se-" maka tipe awalannya secara berturut-turut adalah "di-", "ke-", atau "se-".

2. Jika awalannya adalah "te-", "me-", "be-", atau "pe-" maka dibutuhkan sebuah proses tambahan untuk menentukan tipe awalannya.
3. Jika dua karakter pertama bukan "di-", "ke-", "se-", "te-", "be-", "me-", atau "pe-" maka berhenti.
4. Jika tipe awalan adalah "none" maka berhenti. Jika tipe awalan adalah bukan "none" maka dihapus jika ditemukan.

Contoh aturan tambahan bentuk awalan yang terdapat pada pembentuk kata dasar :

1. Awalan PE-

   Pe + konsonan (j,d,c,z) menjadi "pen"

   Contoh :

   a. Pe + didik = pendidik

Contoh pada kalimat 'pendidikan' apabila dipecah katanya akan memiliki arti yang berbeda-beda seperti antara kalimat 'pen-didik-an', 'pen-didik', dan 'didik-an' memiliki makna yang berbeda maka perlu disamakan dengan stemming menjadi 'didik', jadi kalimat diseragamkan agar tidak banyak model, karena jika kebanyakan model menimbulkan data yang tidak sama, banyak model bisa dibilang bagus akan tetapi dari segi performa kecepatan klasifikasi akan lebih lambat. Menggunakan stemming menjadikan lebih cepat karena beberapa kata yang harusnya bisa jadi banyak model diseragamkan menjadi satu kata. *Stemming* yang dibuat oleh penulis menggunakan bantuan *library* Sastrawi yang sudah

disediakan github *open sources*. Berikut adalah gambar Syntaks dari implementasi penggunaan *Stemming* dengan yang tidak dilakukan *Stemming* :

```
//implementasi penggunaan steemming

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

#DEKLARASI STEMMING
factory = StemmerFactory()
stemmer = factory.create_stemmer()

# MENGAKTIFKAN STEMMING
fitur_ekstraksi4 = []
for cuitan in range(0, len(fitur_ekstraksi3)):
    tmp = stemmer.stem(str(fitur_ekstraksi3[cuitan]))
    fitur_ekstraksi4.append(tmp)
```

Gambar 4. 4 Syntak dari implementasi Stemming

```
//implementasi penggunaan Unstemming

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

#DEKLARASI STEMMING
factory = StemmerFactory()
stemmer = factory.create_stemmer()

    #UNCOMMENT UNTUK MENGAKTIFKAN STEMMING
    # fitur_ekstraksi4 = []
    # for cuitan in range(0, len(fitur_ekstraksi3)):
    #     tmp = stemmer.stem(str(fitur_ekstraksi3[cuitan]))
    #     fitur_ekstraksi4.append(tmp)
```

Gambar 4. 5 Syntak dari implementasi dengan tidak dilakukan Stemming

Berikut contoh hasil *stemming* dapat dilihat pada Tabel 4.4

Tabel 4. 4 Proses *Stemming*

Tidak dilakukan *Stemming*	Dilakukan *Stemming*
'pendidikan', 'agama', 'islam', '10', 'ali', 'bin', 'abi', 'thalib', '421', 'mengoleksi', 'kontak', 'perempuan', 'yang', 'tidak', 'dikenal'	'didik', 'agama', 'islam', '10', 'ali', 'bin','abi', 'thalib','421','koleksi', 'kontak','perempuan','yang', 'tidak', 'dikenal'

4.1.3 *Feature Selection*

Merupakan proses *stopword removal* sebagai pembersihan data yang memiliki kata bermakna umum atau tidak bermakna seperti kata sambung, kata tanya, dan kata benda. Diterapkan tahap ini bertujuan untuk mengurangi data sehingga dapat mempercepat proses berikutnya. Daftar *stopword* dibuat sesuai dengan kebutuhan penelitian, dalam penelitian jumlah *stopword* yang tersedia sebanyak 359 buah terdiri dari *stopword* bahasa Indonesia, yang mana *stopword* ini berisi kalimat-kalimat yang tidak diperlukan pada data yang digunakan.

Apabila ada kata pada data yang digunakan, maka sistem akan menghapus kata yang ada pada *stopword.* Sehingga lebih efektif apabila *stopword* bahasa indonesia dimasukan kedalam proses ini. Setelah proses ini berjalan dari data kata yang tersisa dan menjadi fitur unik dari data tersebut, sehingga setiap data memiliki fiturnya sendiri dan fitur ini yang menjadi patokan dari pengklasifikasian, Berikut syntaks dari implementasi penggunaan *Feature Selection* pada gambar 4.6

```
//implementasi penggunaan feature selection

# Bagian Stopword Removal
# ----------------------------------------------------------
stopsunda1 = open('stopwords_id.txt', 'r')
stopsunda2 = stopsunda1.read()
stopsunda = word_tokenize(stopsunda2)
fitur_ekstraksi = []
def swr(a, b):
    filtered_sentence = []
    for w in a:
        if w not in b:
            filtered_sentence.append(w)
    return filtered_sentence
for cuitan in range(0, len(fitur_ekstraksi5)):
    tmp = swr(fitur_ekstraksi5[cuitan], stopsunda)
    fitur_ekstraksi.append(tmp)
print(*fitur_ekstraksi[:10], sep='\n')
print('\n')
def identity_tokenizer(text):
    return text
```

Gambar 4. 6 Syntak dari implementasi *Feature Selection*

Hasil perbandingan dari proses sebelum *Feature Selection* dan setelah *Feature Selection* proses ini dapat dilihat pada Tabel 4.5

Tabel 4. 5 Proses Feature Selection

Sebelum *Feature Selection*	Setelah *Feature Selection*
'didik', 'agama', 'islam', '10', 'ali', 'bin', 'abi', 'thalib', '421', 'koleksi', 'kontak', 'perempuan', 'yang', 'tidak', 'dikenal'	'didik', 'agama', 'islam', '10', 'ali', 'bin', 'abi', 'thalib', '421','koleksi', 'kontak', 'perempuan', 'kenal'

4.1.4 *Text Representation*

Proses pengklasifikasian yang mudah dilakukan mengubah data menjadi bentuk vektor atau matrik 1 dimensi. Sehingga berbeda dengan BOW (*bag of word*) yang mana fungsinya menghitung frekuensi kata, membuat kata paling sering muncul menjadi fitur dari data tersebut, kebalikan dari TF – IDF (*Term Frequency – Inverse Document Frequency*) yang berfungsi merepresentasikan data agar bisa dibaca oleh mesin, karena mesin hanya mengetahui bahasa biner agar lebih cepat dalam pengolahan algoritma. Semakin sedikit kemunculan kata yang terdapat pada data maka itu adalah fitur data tersebut, cara kerja dari TF – IDF menggunakan pembobotan terhadap hubungan suatu kata pada datanya. Proses konversi data menjadi matrix dan pembobotan kata ini dilengkapi dengan menggunakan *liblary* python *sklearn tfifdvectorize*. Gambaran pada proses ini digambarkan pada tabel 4.6 dibawah ini

Tabel 4. 6 Tabel TF-IDF

Ex	Term Frequency				DF	IDF	TF-IDF			
	Data 1	Data 2	...	Data-400			Data 1	Data 2	...	Data-400
Video	0	0	...	0	12	Log(400/12) = 1,079	0	0	...	0
Porno	0	1	...	0	9	Log(400/9) = 0,954	0	0,954	...	0
...	...	...	...	...	...	...	...	...	...	...
Pacaran	1	0	...	0	19	Log(400/19) = 1,278	1,278	0	...	0

Dijelaskan bahwa dari data yang ada, memiliki berbagai macam fitur yang mana pada tabel ini hanya diambil beberapa sample fitur. Dari data yang akan dijadikan gambaran kasus perhitungan bobot fitur dengan menggunakan metode TF-IDF menggunakan tabel yang digambarkan diatas. Contoh fitur yang digunakan adalah video porno pacaran, sehingga didapatkan fitur video, porno, dan pacaran,

yang mana fitur ini hanya sebagai contoh perwakilan data lain yang memiliki jumlah data banyak dan pada contoh ditabel hanya menggunakan 400 data. Pada kolom *Term Frequency* (TF) memiliki beberapa kolom data, dari data 1 sampai data ke 400. Pada data 1 bisa dihat hanya ditemukan satu fitur pacaran. Pada data 2 juga ditemukan hanya satu fitur porno, begitu seterusnya sampai ke data 400.

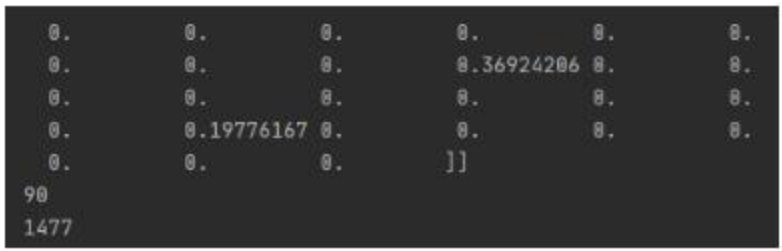
Pada tabel DF menjelaskan *document frequency* merupakan total fitur dari data 1 sampai data ke 400. Ditunjukan bahwa fitur "video" memiliki total 12 fitur dari 400 data, sedangkan fitur "porno"memiliki 9 buah dari 400 data, yang terakhir fitur "pacaran" memiliki 19 buah dari 400 kata. Sedangkan untuk *Inverse Document Frequency* (IDF) dihitung dengan formula seperti pada tabel dengan rumus IDF_j = *Log*(D/df_j) yang dicontohkan " Log(400/12) = 1,079" Dimana D adalah jumlah semua data dalam koleksi yang memiliki total 400 data, sedangkan dfj adalah jumlah fitur yang mengandung term (tj) dangan jumlah total pada fitur video berjumlah 12, fitur porno berjumlah 9, dan fitur pacaran berjumlah 19.

Untuk tabel TF-IDF sendiri merupakan hasil dari banyak fitur yang muncul dari data yang ada pada tabel TF dikalikan dengan hasil yang ada pada tabel IDF. Sehingga jika dijelaskan dengan contoh tabel, pada data 2 di tabel *Term Frequency* (TF) fitur porno ditemukan 1 didalamnya dan pada *Inverse Document Frequency* (IDF) fitur porno memiliki nilai 0,954 maka perhitungan didalam tabel TF-IDF adalah jumlah fitur porno di data 2 yang berjumlah 1 dikalikan dengan nilai porno yang ada IDF 0,954. Jadi nilai TF-IDF pada data 2 di fitur porno adalah 1 * 0,954 = 0,954, dan seperti itu proses selanjutnya. Untuk mendapatkan nilai perhitungan berikut implementasi perhitungan probabilitas kata pada gambar 4.7 :

```
# DATA TRAIN DAN DATA TEST DIGABUNG BUAT DI JADIIN FORMAT TFIDF
for cuitan in range(0, len(fitur_ekstraksinext2)):
    tmp = fitur_ekstraksinext2[cuitan]
    fitur_ekstraksinext.append(tmp)

def identity_tokenizer(text):
    return text

# BAGIAN TFIDF
from sklearn.feature_extraction.text import TfidfVectorizer
vektor_kata = TfidfVectorizer(tokenizer=identity_tokenizer,
lowercase=False)
fitur_ekstraksisiap =
vektor_kata.fit_transform(fitur_ekstraksinext).toarray()
print(fitur_ekstraksisiap[:1])

# BAGIAN DATA TRAIN SAMA DATA TEST DIPISAH KEMBALI
fitur_ekstraksi = []
for cuitan in range(0, len(callbackvalue)):
    tmp = fitur_ekstraksisiap[cuitan]
    fitur_ekstraksi.append(tmp)

fitur_ekstraksitest = []
for cuitan in range(len(callbackvalue), len(fitur_ekstraksisiap)):
    tmp = fitur_ekstraksisiap[cuitan]
    fitur_ekstraksitest.append(tmp)
print(len(fitur_ekstraksitest))
print(len(fitur_ekstraksi))
```

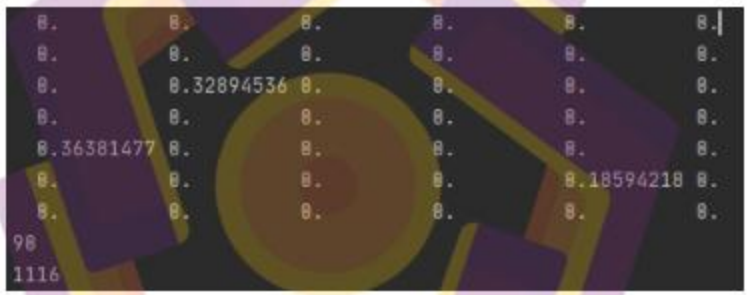
Gambar 4. 7 implementasi proses (TF-IDF)

Dikerenakan pada penelitian menggunakan 2 data yang berbeda kuantitas dan kualitasnya maka hasil pembobotannya pun juga akan berbeda. Dapat dilihat Pada data pertama dengan hasil pembobotan, ditemukan hasil uji coba dengan rincian 90 data untuk hasil dari testing, dan 1477 data untuk hasil training ditunjukan hasilnya pada gambar 4. 8 dibawah ini



Gambar 4. 8 hasil prediksi TF-IDF pada data pertama

Sedangkan pada data kedua dengan hasil prediksi pembobotan , ditemukan hasil uji coba dengan rincian 90 data untuk hasil dari testing, dan 1116 data untuk hasil dari testing yang ditunjukan hasilnya pada gambar 4. 9 dibawah ini



Gambar 4. 9 hasil prediksi TF-IDF pada data kedua

4.1.5 *Application of Text Mining Techniques*

Setelah proses *preprocesing* dilakukan, maka data masuk kedalam proses *training* dataset oleh *Support Vector Machine* dan *Random Forest*. Sebelum dilatih, data dibagi menjadi 2 bagian yaitu data *train* dan data *test* dengan rasio 80 : 20 secara acak. 80% data ini akan dipelajari oleh sistem, hasil pembelajaran data ini yang dapat membuat model yang memprediksi kelas dari suatu data inputan baru.

Pada saat proses *training*, algoritma *Support Vector Machine* (SVM) dibantu dengan metode klasifikasi *multiclass* yaitu metode OvO (*One versus One*

Classification), dikarenakan pada dasarnya algoritma *Support Vector Machine* ini ditujukan hanya untuk klasifikasi 2 kelas saja. Dalam proses OvO ini proses *training* akan menjadi 3 kali, karena objek pengklasifikasian dalam penelitian ini adalah 3 kelas dan semua kelas akan dipertemukan satu per satu. Hasil dari OvO ini adalah hasil terbanyak dari prediksi 3 kali *test* tersebut.

Untuk proses *training* pada *Random Forest* jumlah data menentukan tingkat resiko yang lebih, saat membangun sebuah data *training*, maka aspek yang penting untuk dipertimbangkan adalah pemilihan data. Artinya, jumlah data harus menyesuaikan data *training* selama proses klasifikasi. Dari data yang banyak ini akan dibuatkan *cluster-cluster* yang berbeda menurut klasifikasinya, data tersebut nanti nya akan diklasifikasikan oleh mesin sesuai dengan model yang ditentukan. Hasil prediksi dari setiap *cluster* yang terbanyak akan dijadikan hasil prediksi akhir. Semakin banyak data yang dimiliki maka akan semakin besar pula hasil akurasi yang didapatkan. Proses diatas menunjukan bahwa yang membedakan antara data training dan data test adalah dari segi kuantitas atau ukurannya datanya, yang mana ketika dilakukan training data yang terproses sebesar 80% dan ketika melakukan testing sebesar 20% dan ditemukan perbandingan perbedaannya 80 : 20.

4.2 Uji Coba dan Analisis

Pada tahap ini dijelaskan mengenai hasil dan pembahasan aplikasi yang telah dibuat. Berikut adalah hasil dan pembahasan yang terdiri dari hasil aplikasi *executable*, pengujian, pembahasan, dan keakurasian sistem dengan metode yang telah diterapkan. Tahapan dari uji coba ini berdasarkan jumlah data, yang mana pada penelitian ini menggunakan dua jenis data, data pertama berisikan data yang

masih lengkap sesuai dengan data bawaan dari sistem, sedangkan data yang kedua berisi data yang telah dibersihkan dari data-data yang cacat atau tidak lengkap.

Pada kedua data ini juga akan dilakukan analisis keakurasian dengan metode-metode yang ditentukan. Pemilihan metode ini dilihat dalam penggunaan algoritma dari segi ke akurasiannya, seperti *stemming* dijadikan bahan uji coba karena metode ini dapat mengubah makna dari suatu kata, penggunaan *K-fold validation* untuk melakukan evaluasi model, apakah model tersebut stabil atau tidak dalam pengklasifikasian. Penggunaan *Confussion Matrix* untuk menjelaskan performa dari suatu algoritma pengklasifikasian, yang mana keseluruhan penggunaan metode ini bertujuan untuk mendapatkan nilai akurasi yang lebih detail dan lebih kuat.

4.2.1 Uji Coba Data Pertama

Uji coba pertama menggunakan data pertama yaitu data data yang masih lengkap sesuai dengan data bawaan dari sistem dan telah memiliki label dan memiliki model berikut hasil uji coba berdasarkan metode yang telah ditentukan :

4.2.1.1 Stemming

Pada penjelasan awal akan dibahas bagaimana pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi. Uji coba pertama dengan uji coba dengan data *trainig* dan data *testing* yang belum dipisah. Percobaan awal antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming*. Pada uji coba ini menggunakan aplikasi *phyton*, ditemukan nilai akurasi pada *Support Vector Machine* (SVM) yang tidak menggunakan metode *stemming*

memiliki nilai 0,972 (97,2%). Algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,972 (97,2%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming* pada data pertama memiliki tingkat akurasi yang sama yaitu sebesar 0,972 (97,2%).

Berikutnya penggunaan *Support Vector Machine* dibantu dengan metode OvO (*One versus One Classification*), antara OvO dengan metode yang biasa, dengan OvO penggunaan metode *stemming*. Pada pengujian ini nilai akurasi pada metode OvO yang tidak menggunakan metode *stemming* memiliki nilai 0,986 (98,6%). metode OvO yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,989 (98,9%). Kesimpulan dari hasil perbandingan ini bahwa antara metode OvO dengan metode yang biasa, dengan metode OvO dengan metode *stemming* pada data pertama memiliki tingkat akurasi berbeda, yang mana metode Ovo dengan stemming lebih besar nilai akurasinya.

Dilanjutkan dengan pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi antara algoritma *Random Forest* (RF) dengan metode yang biasa, dengan algoritma *Random Forest* (RF) dengan penggunaan metode *stemming*. Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,983 (98,3%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,986 (98,6%). Kesimpulan dari hasil perbandingan ini adalah *Random Forest* (RF yang biasa, dengan penggunaan metode *stemming* pada data pertama

memiliki tingkat akurasi yang berbeda yaitu 0,983 (98,3%) : 0,986 (98,6%). Tingkat akurasi yang lebih tinggi didapatkan oleh algoritma *Random Forest* (RF) dengan menggunakan metode *stemming*.

Uji coba kedua dengan uji coba dengan data *trainig* dan data *testing* yang sudah dipisah, yang mana percobaan kedua ini melakukan pengujian dengan data baru yang berisi data pelanggaran yang belum memiliki label. Pada uji coba ini dilihat apakah model bisa digunakan atau tidaknya, maka dengan model yang sudah dibuat di uji coba ke dataset yang baru. Data yang baru nantinya akan diprediksi secara otomatis. Pada data pertama yang sebanyak 1.483 setelah diprediksi secara otomatis, apabila tidak bekerja akurasinya dengan baik berarti model kurang bisa digunakan untuk model permasalahan penelitian ini, namun apabila prediksi dan akurasinya baik pada dataset ini maka model ini bisa digunakan untuk permasalahan pada penelitian.

Pertama uji coba antara algoritma *Support Vector Machine* (SVM) metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) penggunaan metode *stemming*. Ditemukan nilai akurasi pada *Support Vector Machine* (SVM) yang tidak menggunakan metode *stemming* memiliki nilai 0,90 (90%). Algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,80 (80%), pada data pertama memiliki tingkat akurasi yang di ungguli oleh *Support Vector Machine* (SVM) yang tidak menggunakan metode *stemming*

Berikutnya penggunaan *Support Vector Machine* yang dibantu dengan metode pengklasifikasian multikelas OvO (*One versus One Classification*), antara

OvO dengan metode yang biasa, dengan OvO penggunaan metode *stemming*. Pada pengujian ini Ditemukan nilai akurasi pada metode OvO yang tidak menggunakan metode *stemming* memiliki nilai 0,988 (98,8%). metode OvO yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,988 (98,8%). Kesimpulan dari hasil perbandingan ini bahwa antara metode OvO dengan metode yang biasa, dengan metode OvO dengan penggunaan metode *stemming* pada data pertama memiliki tingkat akurasi yang sama.

Dilanjutkan dengan pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi antara algoritma *Random Forest* (RF) dengan metode yang biasa, dengan algoritma *Random Forest* (RF) dengan penggunaan metode *stemming*. Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,888 (88,8%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,91 (91%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Random Forest* (RF) metode *stemming*, dengan algoritma *Random Forest* (RF) metode biasa pada data kedua memiliki tingkat akurasi yang berbeda yaitu 0,888 (88,8%) : 0,911 (91,1%). Tingkat akurasi yang lebih tinggi didapatkan oleh algoritma *Random Forest* (RF) dengan menggunakan metode *stemming*.

4.2.1.2 K-fold cross validation

Penjelasan berikutnya dibahas pengaruh penggunaan *K-fold cross validation* terhadap perbandingan nilai akurasi. Kfold digunakan pada penelitian ini untuk melakukan testing data secara keseluruhan, karena data yang digunakan tidak terlalu banyak maka digunakan untuk mengukur kestabilan data, peneliti

menggunakan k-fold dengan tujuan apabila data nya tidak terlalu banyak tapi stabil, maka hasil datanya cocok. Uji coba pertama penggunaan *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM). ditemukan nilai akurasi pada penggunaan *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM) memiliki nilai 0,993 (99,3%).

Selanjutnya penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*), Pada uji coba ditemukan nilai akurasi pada penggunaan K-fold validation untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO (*One versus One Classification*) memiliki nilai akurasi sebesar 0,995 (99, 5%).

Pengaruh penggunaan *K-fold validation* terhadap nilai akurasi algoritma *Random Forest* (RF). Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *K-fold validation* sebesar 0,990 (99%). Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM), juga yang dibantu dengan metode OvO (*One versus One Classification*), penggunaan *K-fold validation* dengan algoritma *Random Forest* (RF), memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,993 (99,3%) : 0,995 (99,5%) : 0,989 (98,9%). Dapat dilihat rincian setiap proses pada tabel 4.7

Tabel 4. 7 Hasil pengujian dengan *K - Fold Cross Validation*

K-fold validation	Akurasi dengan SVM	Akurasi dengan OvO	Akurasi dengan RF
1	99,3%	98,6%	98,6%
2	97,9%	99,3%	98,6%
3	99,3%	100%	98,6%
4	100%	100%	99,3%
5	99,3%	99,3%	99,3%
6	100%	100%	100%
7	100%	100%	100%
8	99,3%	99,3%	98,6%
9	99,3%	99,3%	99,3%
10	98,6%	99,3%	97,9%
Rata-rata	99,3%	99,5%	99%

Uji coba kedua *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM) ditambah metode *stemming*. ditemukan nilai akurasi pada uji coba ini memiliki nilai 0,994 (99, 4%).Selanjutnya penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM) yang ditambah OvO (*One versus One Classification*) yang ditambahkan metode *stemming*, Pada uji coba ini ditemukan nilai akurasi sebesar 0,995 (99, 5%).

Pengaruh penggunaan *K-fold validation* terhadap nilai akurasi algoritma *Random Forest* (RF) yang ditambahkan metode *stemming* ditemukan sebesar 0,989 (98,9%). Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *K-fold validation* yang ditambahkan metode *stemming* untuk algoritma *Support*

Vector Machine (SVM), juga yang dibantu dengan metode OvO (*One versus One Classification*), juga algoritma *Random Forest* (RF), memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,994 (99, 4%) : 0,995 (99,5%) : 0,989 (98,9%). dilihat rincian setiap proses pada tabel 4.8

Tabel 4. 8 Hasil pengujian dengan *K - Fold Cross Validation* dengan *Stemming*

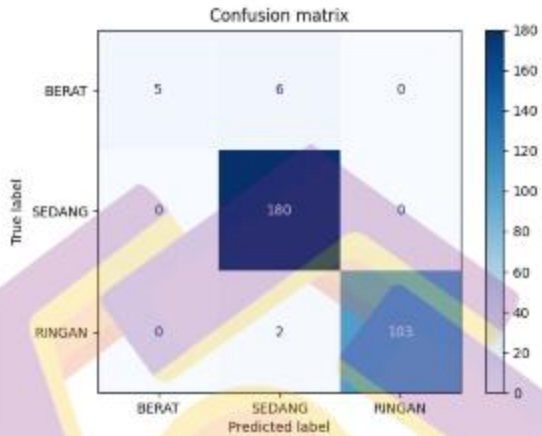
K-fold validation	**Akurasi dengan SVM**	**Akurasi dengan OvO**	**Akurasi dengan RF**
1	100%	99,3%	99,3%
2	98,6%	99,3%	97,9%
3	99,3%	100%	98,6%
4	100%	100%	99,3%
5	99,3%	99,3%	99,3%
6	100%	100%	99,3%
7	100%	100%	100%
8	99,3%	99,3%	98,6%
9	99,3%	99,3%	99,3%
10	98,6%	99,3%	97,9%
Rata-rata	99, 4%	99,5%	98,9%

Diambil kesimpulan setelah dilakukan uji coba dengan hasil akurasi yang didapat, bahwa model algoritma *Support Vector Machine* (SVM) dan *Random Forest* (RF) dengan metode biasa lebih stabil dalam mengklasifikasikan dataset yang digunakan daripada yang menggunakan metode *Stemming*

4.2.1.3 Confussion Matrix

Proses berikutnya akan dibahas bagaimana pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi. Uji coba pertama penggunaan

Confussion Matrix untuk algoritma *Support Vector Machine* (SVM), yang hasil implementasinya pada dapat dilihat pada gambar 4.10



Gambar 4. 10 Hasil pengujian dengan *confusion matrix* SVM

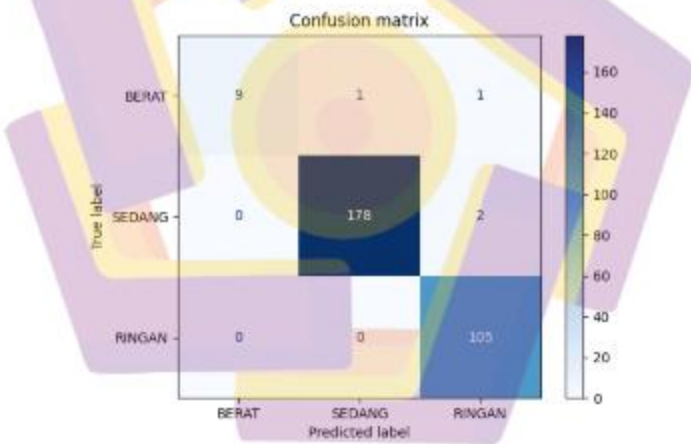
Pada gambar diatas dijelaskan bahwa terdapat 3 *class* yang terdiri dari BERAT, SEDANG, RINGAN. Pada *class* BERAT dilihat memiliki 5 data yang akurasi prediksinya benar, dan 6 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG. Pada *class* SEDANG tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun RINGAN, ditemukan 180 data yang akurasi prediksinya benar. Pada class RINGAN ditemukan 2 data salah data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG dan memiliki 103 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) memiliki nilai 0,972 (97,2%) hasil dilihat pada tabel 4.9

Tabel 4. 9 Evaluasi *confusion matrix SVM*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,45	0,62	11
Ringan	0,96	1	0,98	180
Sedang	1	0,98	0,99	105
Akurasi	0,97			296

Uji coba yang kedua penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*), Pada uji coba ini hasil implementasi dapat diihat pada gambar 4.11



Gambar 4. 11 Hasil pengujian dengan *confusion matrix SVM OvO*

Pada *class* BERAT dilihat memiliki 9 data yang akurasi prediksinya benar, 1 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG, dan 1 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada c*lass* SEDANG ditemukan 1 data salah, dianggap prediksinya bersinggungan

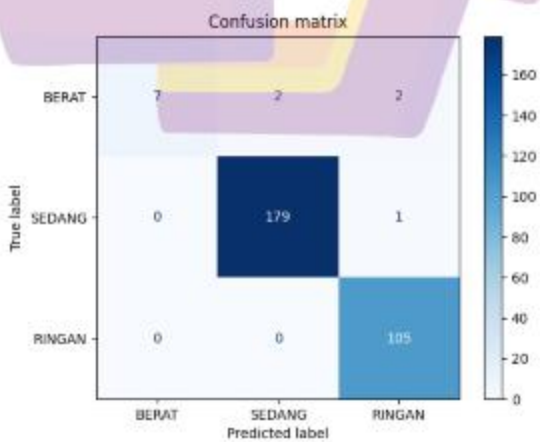
dengan data RINGAN, ditemukan 178 data yang akurasi prediksinya benar. Pada c*lass* RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 105 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO (*One versus One Classification*) memiliki nilai akurasi sebesar 0,986 (98,6%) hasil dilihat pada tabel 4.10

Tabel 4. 10 Evaluasi *confusion matrix SVM OvO*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,82	0,90	11
Ringan	0,99	0,99	0,99	180
Sedang	0,98	1	0,99	105
Akurasi		0,986		296

Berikutnya pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi algoritma *Random Forest* (RF). Pada uji coba ini menggunakan aplikasi *phyton*, yang hasil implementasinya pada dapat diihat pada gambar 4.12



Gambar 4. 12 Hasil pengujian dengan *confusion matrix* SVM

Pada *class* BERAT dilihat memiliki 7 data yang akurasi prediksinya benar, 2 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG, dan 2 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada c*lass* SEDANG ditemukan 2 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN dan ditemukan 179 data yang akurasi prediksinya benar. Pada c*lass* RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 105 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *Confussion Matrix* sebesar 0,983 (98,3%). hasil dilihat pada tabel 4.11

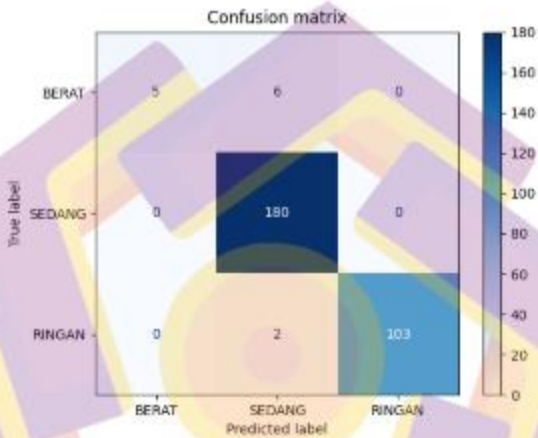
Tabel 4. 11 Evaluasi RF

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,64	0,78	11
Ringan	0,99	0,99	0,99	180
Sedang	0,98	1	0,99	105
Akurasi		0,983		296

Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*), penggunaan *Confussion Matrix* dengan algoritma *Random Forest* (RF), memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,972 (97,2%) : 0,986 (98,6%) : 0,983 (98,3%). Tingkat akurasi yang lebih tinggi didapatkan oleh penggunaan *Confussion Matrix*

untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*).

Uji coba kedua *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) ditambah metode *stemming*. Hasil implementasinya pada dapat dilihat pada gambar 4.13



Gambar 4. 13 Hasil pengujian dengan *confusion matrix SVM OvO*

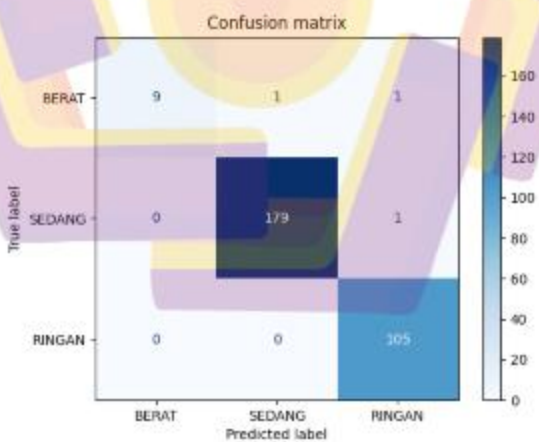
Dijelaskan bahwa terdapat 3 *class* yang terdiri dari BERAT, SEDANG, RINGAN. Pada *class* BERAT memiliki 5 data yang akurasi prediksinya benar, dan 6 data salah, dianggap prediksinya bersinggungan dengan data SEDANG. Pada *class* SEDANG tidak ditemukan kesalahan bersinggungan dengan class SEDANG maupun RINGAN, ditemukan 180 data akurasi prediksinya benar. Pada class RINGAN ditemukan 2 data salah, dianggap prediksinya bersinggungan dengan data SEDANG dan memiliki 103 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada uji coba ini memiliki nilai 0,972 (97,2%) hasil dilihat pada tabel 4.12

Tabel 4. 12 Evaluasi *confusion matrix SVM* metode *stemming*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,45	0,62	11
Ringan	0,96	1	0,98	180
Sedang	1	0,98	0,99	105
Akurasi		0,972		296

Uji coba yang berikutnya penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*) yang dilakukan metode *stemming*, hasil implementasinya pada dapat diihat pada gambar 4.14



Gambar 4. 14 Hasil pengujian dengan *confusion matrix SVM OvO*

Pada *class* BERAT dilihat memiliki 9 data yang akurasi prediksinya benar, 1 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG, dan 1 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada c*lass* SEDANG ditemukan 1 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN dan ditemukan 179 data yang akurasi prediksinya benar. Pada c*lass* RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 105 data yang akurasi prediksinya benar.

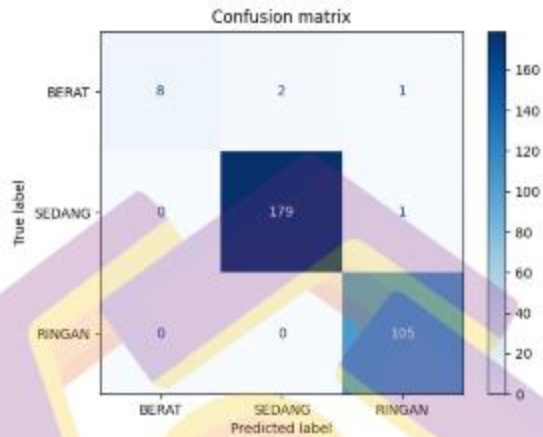
Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO (*One versus One Classification*) memiliki nilai akurasi sebesar 0,989 (98,9%) hasil dilihat pada tabel 4.13

Tabel 4. 13 Evaluasi *confusion matrix SVM OvO* metode *stemming*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,82	0,90	11
Ringan	0,99	0,99	0,99	180
Sedang	0,98	1	0,99	105
Akurasi		0,989		296

Berikutnya pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi algoritma *Random Forest* (RF) yang dilakukan metode *stemming*. Pada uji coba

ini menggunakan aplikasi *phyton*, yang hasil implementasinya pada dapat diihat pada gambar 4.15



Gambar 4. 15 Hasil pengujian dengan *confusion matrix RF*

Pada *class* BERAT dilihat memiliki 8 data yang akurasi prediksinya benar, 2 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG, dan 1 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada *class* SEDANG dan ditemukan 179 data yang akurasi prediksinya benar. Pada *class* RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 105 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *Confussion Matrix* sebesar 0,986 (98,6%). hasil dilihat pada tabel 4.14

Tabel 4. 14 Evaluasi RF metode *stemming*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,73	0,84	11
Ringan	0,99	0,99	0,99	180
Sedang	0,98	1	0,99	105
Akurasi	0,986			296

Kesimpulan dari hasil perbandingan uji coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), dengan dibantu dengan metode OvO (*One versus One Classification*), serta *Random Forest* (RF) metode biasa, memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,972 (97,2%) : 0,986 (98, 6%) : 0,983 (98,3%). Tingkat akurasi yang lebih tinggi didapatkan oleh penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*).

Sedangkan perbandingan uji kedua coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), dengan dibantu dengan metode OvO (*One versus One Classification*), serta *Random Forest* (RF) dengan metode Stemming, memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,972 (97,2%) : 0,989 (98,9%) : 0,986 (98,6%). Tingkat akurasi yang lebih tinggi didapatkan sama dengan metode biasa.

Diambil kesimpulan setelah dilakukan uji coba dengan hasil akurasi yang didapat, bahwa model algoritma *Support Vector Machine* (SVM) dan *Random Forest* (RF) dengan metode *Stemming* lebih stabil dalam mengklasifikasikan dataset yang digunakan daripada yang menggunakan metode biasa.

4.2.2 Uji Coba Data Kedua

Setelah Uji coba pada data pertama telah dilakukan, maka pada selanjutnya uji coba menggunakan data kedua yaitu data yang telah dibersihkan dari data-data yang cacat atau tidak lengkap. berikut hasil uji coba berdasarkan metode yang telah ditentukan :

4.2.2.1 Stemming

Pada penjelasan awal akan dibahas bagaimana pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi. Uji coba pertama dengan uji coba dengan data *trainig* dan data *testing* yang belum dipisah. Percobaan awal antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming*. Pada uji coba ini menggunakan aplikasi *phyton*, ditemukan nilai akurasi pada *Support Vector Machine* (SVM) yang tidak menggunakan metode *stemming* memiliki nilai 0,986 (98,6%). Algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming* ditemukan nilai akurasi yang sama sebesar 0,986 (98,6%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming* pada data pertama memiliki tingkat akurasi yang sama yaitu sebesar 0,986 (98,6%).

Berikutnya penggunaan *Support Vector Machine* yang dibantu dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*), antara OvO dengan metode yang biasa, dengan OvO penggunaan metode *stemming*. Pada pengujian ini Ditemukan nilai akurasi pada metode OvO

yang tidak menggunakan metode *stemming* memiliki nilai 0,995 (99,5%). metode OvO yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,989 0,995 (99,5%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming* pada data pertama memiliki tingkat akurasi yang sama yaitu sebesar 0,995 (99,5%).

Dilanjutkan dengan pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi antara algoritma *Random Forest* (RF) metode biasa, dengan algoritma *Random Forest* (RF) dengan penggunaan metode *stemming*. Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,991 (99,1%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,982 (98,2%). Kesimpulan dari hasil perbandingan ini adalah *Random Forest* (RF) yang biasa, dengan penggunaan metode *stemming* pada data kedua memiliki tingkat akurasi yang berbeda yaitu 0,991 (99,1%) : 0,982 (98,2%). Tingkat akurasi lebih tinggi didapatkan oleh algoritma *Random Forest* (RF) dengan menggunakan metode biasa tanpa *Stemming*.

Uji coba kedua dengan uji coba dengan data *trainig* dan data *testing* yang sudah dipisah, yang mana percobaan kedua ini melakukan pengujian dengan data baru yang berisi data pelanggaran yang belum memiliki label. Pada uji coba ini dilihat apakah model bisa digunakan atau tidaknya, maka dengan model yang sudah dibuat di uji coba ke dataset yang baru. Data yang baru nantinya akan diprediksi secara otomatis. Pada data kedua yang sebanyak 1.117 setelah

diprediksi secara otomatis, apabila tidak bekerja akurasinya dengan baik berarti model kurang bisa digunakan untuk model permasalahan penelitian ini, namun apabila prediksi dan akurasinya baik pada dataset ini maka model ini bisa digunakan untuk permasalahan pada penelitian.

Pertama uji coba antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming*. Pada uji coba ini ditemukan nilai akurasi pada *Support Vector Machine* (SVM) yang tidak menggunakan metode *stemming* memiliki nilai 0,800 (80%). Algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,800 (80%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Support Vector Machine* (SVM) dengan metode yang biasa, dengan algoritma *Support Vector Machine* (SVM) dengan penggunaan metode *stemming* pada data kedua memiliki tingkat akurasi yang sama yaitu sebesar 0,80 (80%).

Berikutnya penggunaan *Support Vector Machine* yang dibantu dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*), antara OvO dengan metode yang biasa, dengan OvO penggunaan metode *stemming*. Pada pengujian ini Ditemukan nilai akurasi pada metode OvO yang tidak menggunakan metode *stemming* memiliki nilai 0,866 (86,6%). metode OvO yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,988 0,866 (86,6%). Kesimpulan dari hasil perbandingan ini bahwa antara metode OvO yang biasa, dengan metode OvO dengan penggunaan metode *stemming* pada data kedua memiliki tingkat akurasi yang sama yaitu sebesar 0,86 (86%).

Dilanjutkan dengan pengaruh penggunaan *stemming* terhadap perbandingan nilai akurasi antara algoritma *Random Forest* (RF) dengan metode yang biasa, dengan algoritma *Random Forest* (RF) dengan penggunaan metode *stemming*. Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,811 (81,1%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,833 (83,3%). Kesimpulan dari hasil perbandingan ini bahwa antara algoritma *Random Forest* (RF) metode *stemming*, dengan algoritma *Random Forest* (RF) metode biasa pada data kedua memiliki tingkat akurasi yang berbeda yaitu 0,811 (81,1%) 0,833 (83,3%). Tingkat akurasi yang lebih tinggi didapatkan oleh algoritma *Random Forest* (RF) dengan menggunakan metode *stemming*.

4.2.2.2 K-Fold

Penjelasan berikutnya dibahas pengaruh penggunaan *K-fold cross validation* terhadap perbandingan nilai akurasi. Uji coba pertama penggunaan *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM). ditemukan nilai akurasi pada penggunaan *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM) memiliki nilai 0,993 (99,3%).

Selanjutnya penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*), Pada uji coba ditemukan nilai akurasi pada penggunaan K-fold validation untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO (*One versus One Classification*) memiliki nilai akurasi sebesar 0,994 (99, 4%).

Pengaruh penggunaan *K-fold validation* terhadap nilai akurasi algoritma *Random Forest* (RF). Pada uji coba ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *K-fold validation* sebesar 0,988 (98,8%). Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM), juga yang dibantu dengan metode OvO (*One versus One Classification*), penggunaan *K-fold validation* dengan algoritma *Random Forest* (RF), memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,993 (99,3%) : 0,994 (99,4%) : 0,988 (98,8%). Dapat dilihat rincian setiap proses pada tabel 4.15

Tabel 4. 15 Hasil pengujian dengan *K - Fold Cross Validation*

K-fold validation	**Akurasi dengan SVM**	**Akurasi dengan OvO**	**Akurasi dengan RF**
1	99,1%	100%	98,2%
2	97,3%	98,2%	98,2%
3	100%	100%	100%
4	99,1%	100%	99,1%
5	99,1%	98,2%	98,2%
6	99,1%	99,1%	98,2%
7	100%	100%	100%
8	100%	100%	99 %
9	100%	99 %	99 %
10	100%	100%	98,1%
Rata-rata	99,3%	99,4%	98,8%

Uji coba kedua *K-fold cross validation* untuk algoritma *Support Vector Machine* (SVM) ditambah metode *stemming*. ditemukan nilai akurasi pada uji coba ini memiliki nilai 0,994 (99, 4%).Selanjutnya penggunaan *K-fold validation* untuk algoritma *Support Vector Machine* (SVM) yang ditambah OvO (*One versus One Classification*) yang ditambahkan metode *stemming*, Pada uji coba ini ditemukan nilai akurasi sebesar 0,995 (99, 5%).

Pengaruh penggunaan *K-fold validation* terhadap nilai akurasi algoritma *Random Forest* (RF) yang ditambahkan metode *stemming* ditemukan sebesar 0,988 (98,8%). Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *K-fold validation* yang ditambahkan metode *stemming* untuk algoritma *Support Vector Machine* (SVM), juga yang dibantu dengan metode OvO (*One versus One Classification*), algoritma *Random Forest* (RF), memiliki tingkat akurasi berbeda dengan perbandingan yaitu 0,994 (99, 4%) : 0,995 (99,5%) : 0,998 (99,8%). Dilihat rincian setiap proses pada tabel 4.16

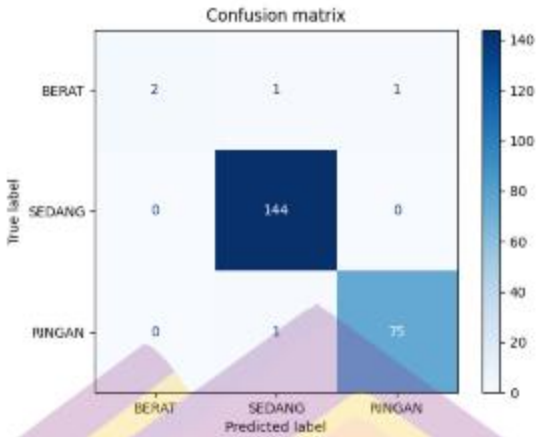
Tabel 4. 16 Hasil pengujian dengan *K - Fold Cross Validation* dengan *Stemming*

K-fold validation	**Akurasi dengan SVM**	**Akurasi dengan OvO**	**Akurasi dengan RF**
1	99,1%	100%	97,3%
2	97,3%	98,2%	98,2%
3	100%	100%	100%
4	99,1%	100%	99,1%
5	100%	99,1%	98,2%
6	99,1%	99,1%	98,2%
7	100%	100%	100%
8	100%	99 %	99 %
9	100%	100%	99 %
10	100%	100%	99 %
Rata-rata	99, 4%	99,5%	98,8%

Diambil kesimpulan setelah dilakukan uji coba, bahwa model algoritma *Support Vector Machine* (SVM) dan *Random Forest* (RF) dengan metode *Stemming* lebih stabil dalam mengklasifikasikan dataset yang digunakan daripada yang menggunakan metode biasa meski kenaikan akurasinya tidak terlalu banyak.

4.2.2.3 Confussion Matrix

Proses berikutnya membahas bagaimana pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi data kedua. Uji coba pertama penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), yang hasil implementasinya pada dapat dilihat pada gambar 4.16



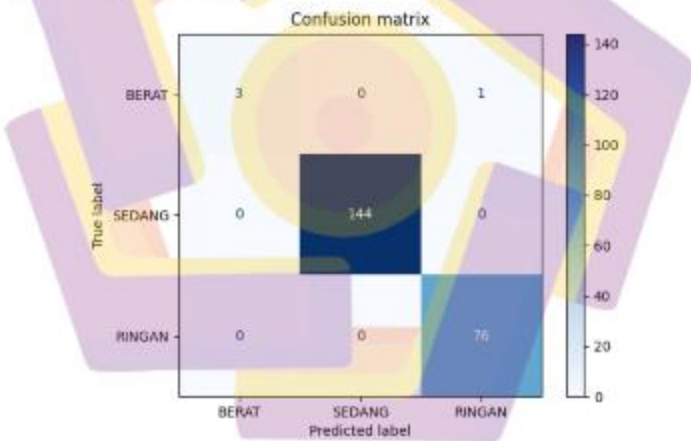
Gambar 4. 16 Hasil pengujian dengan confusion matrix SVM

Pada gambar diatas dijelaskan bahwa terdapat 3 *class* yang terdiri dari BERAT, SEDANG, RINGAN. Pada *class* BERAT dilihat memiliki 2 data yang akurasi prediksinya benar. 1 data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG, 1 data salah, yang dianggap prediksinya bersinanggungan dengan data RINGAN. Pada *class* SEDANG tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun RINGAN, ditemukan 144 data yang akurasi prediksinya benar. Pada class RINGAN ditemukan 1 data salah data salah, yang dianggap prediksinya bersinggungan dengan data SEDANG dan memiliki 75 data yang akurasi prediksinya benar. Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) memiliki nilai 0,986 (98,6%) hasil dilihat pada tabel 4.17

Tabel 4. 17 Evaluasi *confusion matrix SVM*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,50	0,67	11
Ringan	0,99	1	0,99	180
Sedang	0,99	0,99	0,99	105
Akurasi	0,986			296

Berikutnya penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*), yang hasil implementasinya pada dapat diihat pada gambar 4.17



Gambar 4. 17 Hasil pengujian dengan *confusion matrix SVM OvO*

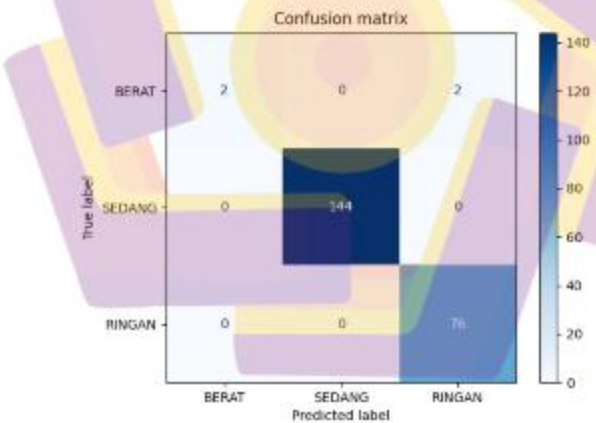
Pada *class* BERAT dilihat memiliki 3 data yang akurasi prediksinya benar, dan 1 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada *class* SEDANG ditemukan 144 data yang akurasi prediksinya benar. Pada *class* RINGAN tidak ditemukan kesalahan yang bersinggungan dengan class SEDANG maupun BERAT, ditemukan 76 data yang akurasi prediksinya benar.

Pada pemodelan analisis, ditemukan nilai akurasi penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO memiliki nilai akurasi sebesar 0,995 (99,5%) hasil dilihat pada tabel 4.18

Tabel 4. 18 Evaluasi *confusion matrix SVM OvO* data kedua

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,75	0,86	4
Ringan	1	1	1	144
Sedang	0,99	1	0,99	76
Akurasi	0,995			224

Berikutnya pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi algoritma *Random Forest* (RF). Hasil implementasi diihat pada gambar 4.18



Gambar 4. 18 Hasil pengujian dengan *confusion matrix RF*

Pada *class* BERAT dilihat memiliki 2 data yang akurasi prediksinya benar, dan 2 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada c*lass* SEDANG ditemukan 144 data yang akurasi prediksinya benar. Pada c*lass*

RINGAN tidak ditemukan kesalahan yang bersinggungan dengan class SEDANG maupun BERAT, ditemukan 76 data yang akurasi prediksinya benar.

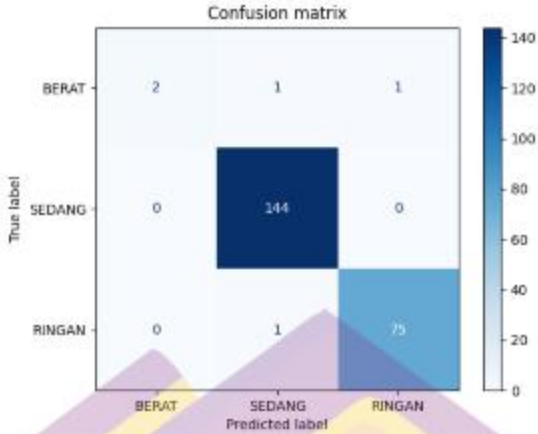
Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *Confussion Matrix* sebesar 0,991 (99,1%). hasil dilihat pada tabel 4.19

Tabel 4. 19 Evaluasi RF data kedua

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,50	0,67	11
Ringan	1	1	1	144
Sedang	0,97	1	0,99	76
Akurasi	0,991			224

Kesimpulan dari hasil perbandingan 3 uji coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*), penggunaan *Confussion Matrix* dengan algoritma *Random Forest* (RF), memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,986 (97,2%) : 0,995 (99,5%) : 0,991 (99,1%). Tingkat akurasi yang lebih tinggi didapatkan oleh penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*).

Uji coba kedua *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) ditambah metode *stemming*. Implementasi dapat dilihat pada gambar 4.19



Gambar 4. 19 Hasil pengujian *confusion matrix* SVM dengan *Stemming*

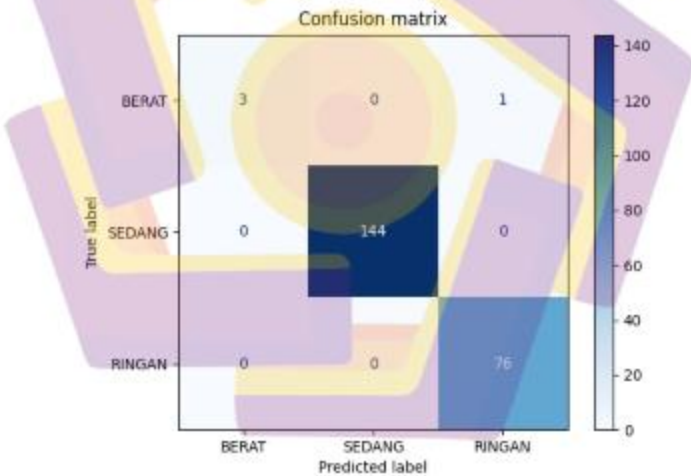
Dijelaskan bahwa terdapat 3 *class* yang terdiri dari BERAT, SEDANG, RINGAN. Pada *class* BERAT memiliki 2 data yang akurasi prediksinya benar. 1 data salah, dianggap prediksinya bersinggungan dengan data SEDANG. 1 data salah, dianggap prediksinya bersinggungan dengan data RINGAN. Pada *class* SEDANG tidak ditemukan kesalahan bersinggungan dengan class SEDANG maupun RINGAN, ditemukan 144 data akurasi prediksinya benar. Pada class RINGAN ditemukan 1 data salah, dianggap prediksinya bersinggungan dengan data SEDANG dan memiliki 75 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada uji coba ini memiliki nilai 0,986 (98,6%) hasil dilihat pada tabel 4.20

Tabel 4. 20 Evaluasi *confusion matrix SVM* metode *stemming*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,50	0,67	4
Ringan	0,99	1	0,99	144
Sedang	0,99	0,99	0,99	76
Akurasi	0,986			296

Uji coba yang berikutnya penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode pengklasifikasian multikelas yaitu metode OvO (*One versus One Classification*) yang dilakukan metode *stemming*, hasil implementasinya pada dapat diihat pada gambar 4.20



Gambar 4. 20 Hasil pengujian *confusion matrix SVM OvO* dengan *Stemming*

Pada *class* BERAT dilihat memiliki 3 data yang akurasi prediksinya benar, dan 1 data salah yang dianggap prediksinya bersinggungan dengan data RINGAN .Pada c*lass* SEDANG ditemukan 144 data yang akurasi prediksinya benar. Pada

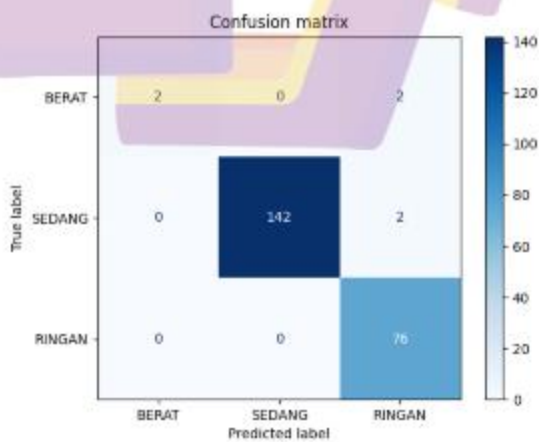
class RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 76 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) dengan metode OvO (*One versus One Classification*) memiliki nilai akurasi sebesar 0,995 (99,5%) hasil dilihat pada tabel 4.21

Tabel 4. 21 Evaluasi *confusion matrix SVM OvO* metode *stemming*

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,75	0,86	4
Ringan	1	1	1	144
Sedang	0,99	1	0,99	76
Akurasi	0,989			224

Berikutnya pengaruh penggunaan *Confussion Matrix* terhadap nilai akurasi algoritma *Random Forest* (RF) yang dilakukan metode *stemming*., yang hasil implementasinya pada dapat diihat pada gambar 4.21



Gambar 4. 21 Hasil pengujian *confusion matrix RF* dengan *Stemming*

Pada *class* BERAT dilihat memiliki 2 data yang akurasi prediksinya benar, dan 2 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN . Pada c*lass* SEDANG dan ditemukan 142 data yang akurasi prediksinya benar, dan 2 data salah, yang dianggap prediksinya bersinggungan dengan data RINGAN. Pada c*lass* RINGAN tidak ditemukan kesalahan yang mana bersinggungan dengan class SEDANG maupun BERAT, ditemukan 76 data yang akurasi prediksinya benar.

Setelah dilakukan pemodelan analisis, ditemukan nilai akurasi pada *Random Forest* (RF) dengan menggunakan *Confussion Matrix* sebesar 0,982 (98,2%). hasil dilihat pada tabel 4.22

Tabel 4. 22 Evaluasi RF metode *stemming* data kedua

Kelas	Pricision	Recall	F1-Score	Data
Berat	1	0,50	0,67	4
Ringan	1	0,99	0,99	144
Sedang	0,95	1	0,97	76
Akurasi		0,982		224

Kesimpulan dari hasil perbandingan uji coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), dengan dibantu dengan metode OvO (*One versus One Classification*), serta *Random Forest* (RF) metode biasa, memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,986(98,6%) : 0,995 (99, 5%) : 0,991 (99,1%). Tingkat akurasi yang lebih tinggi didapatkan oleh penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM) yang dibantu dengan metode OvO (*One versus One Classification*).

Sedangkan perbandingan uji kedua coba antara penggunaan *Confussion Matrix* untuk algoritma *Support Vector Machine* (SVM), dengan dibantu dengan metode OvO (*One versus One Classification*), serta *Random Forest* (RF) dengan metode Stemming, memiliki tingkat akurasi yang berbeda dengan perbandingan yaitu 0,986 (98,6%) : 0,995 (99,5%) : 0,982 (98,2%). Tingkat akurasi yang lebih tinggi didapatkan sama dengan metode biasa.

Diambil kesimpulan setelah dilakukan uji coba dengan hasil akurasi yang didapat, bahwa model algoritma *Support Vector Machine* (SVM) dan *Random Forest* (RF) dengan metode biasa lebih stabil dalam mengklasifikasikan dataset yang digunakan daripada yang menggunakan metode *Stemming*.

4.2.3 Analisis Perbandingan

4.2.3.1 Analisis Perbandingan Data Pertama

Setelah hasil proses uji coba menggunakan 3 metode diatas, diambil perbandingan dari data pertama antara penggunaan metode tidak dilakukan *stemming* dengan penggunaan metode dilakukan *stemming*, yang mana dalam uji coba algoritma *Support Vector Machine* (SVM) biasa memiliki nilai akurasi sebesar 0,972 (97,2%) dan hasil nilai akurasi yang sama didapatkan setelah dilakukan uji coba dan perbandingan dengan algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming*.

Selanjutnya pada *Support Vector Machine* (SVM) dibantu dengan metode menggunakan metode pengklasifikasian multiclass OvO (*One versus One Classification*) antara penggunaan metode tidak dilakukan *stemming* dengan penggunaan metode dilakukan *stemming*, bisa dilihat nilai akurasi pada *Support*

Vector Machine (SVM) yang di optimalisasi dengan OvO (*One versus One Classification*) metode biasa dapat meningkat menjadi 0,986 (98,6%). Untuk *Support Vector Machine* (SVM) yang di optimalisasi dengan OvO (*One versus One Classification*) metode *stemming* meningkat menjadi 0,989 (98,9%)

Dalam uji coba algoritma *Random Forest* (RF) biasa memiliki nilai akurasi 0,983 (98,3%), sedangkan setalah dilakukan uji coba dengan algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ternyata hasil akurasi yang diperoleh menjadi 0,986 (98,6%). Diambil kesimpulan metode *stemming* pada algoritma *Support Vector Machine* (SVM) memiliki kenaikan dalam nilai akurasi, apalagi setelah dibantu dengan metode OvO, sehingga berpengaruh secara signifikan terhadap dataset pertama, berbeda dengan algoritma *Random Forest* (RF) yang tidak terlalu banyak meningkat nilai akurasinya, sehingga tidak terlalu berpengaruh secara signifikan.

Pada uji coba dengan data baru yang berisi data pelanggaran yang belum memiliki label, untuk menguji data *trainig* dan data *testing* yang dipisah. Ditemukan *Support Vector Machine* (SVM) biasa memiliki nilai akurasi sebesar 0,900 (90%) dan hasil nilai *Support Vector Machine* (SVM) yang dilakukan *stemming* memiliki nilai akurasi sebesar 0,810 (81%). Berikutnya penggunaan *Support Vector Machine* yang dibantu dengan OvO dengan metode yang biasa dibandingkan dengan OvO penggunaan metode *stemming*. Pada pengujian ini ditemukan nilai akurasi yang sama yaitu 0,988 (98,8%).

Dilanjutkan dengan perbandingan nilai akurasi antara algoritma *Random Forest* (RF) dengan metode yang biasa, dengan algoritma *Random Forest* (RF)

dengan penggunaan metode *stemming*. Ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,888 (88,8%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,911 (91,1%). Maka diambil kesimpulan pada algoritma *Support Vector Machine* (SVM) mendapat nilai akurasi yang baik ketika tidak dilakukan *stemming*, akan tetapi menurun nilai akurasinya setelah dilakukan *stemming*. Pada *Random Forest* (RF) mendapatkan peningkatan akurasi setelah dilakukan *stemming*, jadi pada uji coba ini algortitma *Random Forest* (RF) lebih baik akurasinya dibandingkan *Support Vector Machine* (SVM) dalam uji coba ini.

Pada penggunaan metode *K-fold* tanpa *stemming*, algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,993 (99,3%), setelah dibantu dengan metode menggunakan metode pengklasifikasian multiclass OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,990 (99%). Penggunaan metode *K-fold* dengan *stemming* algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,994 (99,4%), setelah dibantu dengan metode menggunakan metode pengklasifikasian multiclass OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,989 (98,9%). Maka diambil kesimpulan bahwa algoritma *Support Vector Machine* (SVM) dalam penggunaan *K-fold* antara dilakukan *stemming* dengan yang tidak dilakukan *stemming* terdapat perbedaan bahwa pada *Support Vector Machine* (SVM) tidak

terjadi kenaikan akurasi yang banyak akan tetapi mendapatkan model lebih stabil dalam mengklasifikasikan dataset pertama dari pada algoritma *Random Forest* (RF) yang mengalami penurunan akurasi setalah dilakukan *stemming*.

Pada metode *Confussion Matrix* tanpa *stemming*, algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,972 (97,2%), setelah dibantu dengan metode menggunakan metode pengklasifikasian *multiclass* yaitu metode OvO (*One versus One Classification*) mendapatkan nilai 0,986 (98,6%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,983 (98,3%). Penggunaan *Confussion Matrix* dengan *stemming* algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,972 (97,2%), setelah dibantu dengan metode menggunakan metode pengklasifikasian *multiclass* yaitu metode OvO (*One versus One Classification*) mendapatkan nilai 0,989 (98,9%), sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,986 (98,6%). Maka diambil kesimpulan bahwa dua algoritma dalam penggunaan metode *Confussion Matrix* antara dilakukan *stemming* dengan yang tidak dilakukan terdapat persamaan bahwa pada *Support Vector Machine* (SVM) dan *Random Forest* (RF) terjadi kenaikan akurasi yang tidak banyak akan tetapi mendapatkan model lebih stabil dalam mengklasifikasikan dataset pertama setelah dilakukan pengunaan *stemming*.

4.2.3.2 Analisis Perbandingan Data Kedua

Setelah hasil proses uji coba pada data pertama, selanjutnya pada data kedua diambil perbandingan dengan metode tidak dilakukan *stemming* dengan penggunaan metode dilakukan *stemming*, yang mana dalam uji coba algoritma

Support Vector Machine (SVM) biasa memiliki nilai akurasi sebesar 0,986 (98,6%) dan hasil nilai akurasi yang sama didapatkan setelah dilakukan uji coba dan perbandingan dengan algoritma *Support Vector Machine* (SVM) yang menggunakan metode *stemming*.

Selanjutnya pada *Support Vector Machine* (SVM) dibantu dengan metode menggunakan metode pengklasifikasian multiclass OvO (*One versus One Classification*) antara penggunaan metode tidak dilakukan *stemming* dengan penggunaan metode dilakukan *stemming*, bisa dilihat nilai akurasi pada *Support Vector Machine* (SVM) yang di optimalisasi dengan OvO (*One versus One Classification*) metode biasa dan metode *stemming* sama-sama meningkat menjadi 0,995 (99,5%)

Dalam uji coba algoritma *Random Forest* (RF) biasa memiliki nilai akurasi 0,991 (99,1%), sedangkan setalah dilakukan uji coba dengan algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ternyata hasil akurasi yang diperoleh menjadi 0,982(98,2%). Diambil kesimpulan metode *stemming* pada algoritma *Support Vector Machine* (SVM) memiliki kenaikan dalam nilai akurasi, apalagi setelah dibantu dengan metode OvO, sehingga berpengaruh secara signifikan terhadap dataset pertama, berbeda dengan algoritma *Random Forest* (RF) yang malah menurun nilai akurasinya, sehingga tidak terlalu berpengaruh secara signifikan.

Pada uji coba dengan data baru yang berisi data pelanggaran yang belum memiliki label, untuk menguji data *trainig* dan data *testing* yang dipisah. Ditemukan *Support Vector Machine* (SVM) biasa memiliki nilai akurasi sebesar

0,800 (80%) dan hasil nilai *Support Vector Machine* (SVM) yang dilakukan *stemming* memiliki nilai akurasi sebesar 0,800 (80%). Berikutnya penggunaan *Support Vector Machine* yang dibantu dengan OvO dengan metode yang biasa dibandingkan dengan OvO penggunaan metode *stemming*. Pada pengujian ini ditemukan nilai akurasi yang sama yaitu 0,866 (86,6%).

Dilanjutkan dengan perbandingan nilai akurasi antara algoritma *Random Forest* (RF) dengan metode yang biasa, dengan algoritma *Random Forest* (RF) dengan penggunaan metode *stemming*. Ditemukan nilai akurasi pada *Random Forest* (RF) yang tidak menggunakan metode *stemming* memiliki nilai 0,811 (81,1%). Algoritma *Random Forest* (RF) yang menggunakan metode *stemming* ditemukan nilai akurasi sebesar 0,833 (83,3%). Maka diambil kesimpulan pada algoritma *Support Vector Machine* (SVM) mendapat nilai akurasi yang baik ketika dilakukan *stemming*, sama halnya Pada *Random Forest* (RF) mendapatkan peningkatan akurasi setelah dilakukan *stemming*, jadi pada uji coba ini algortitma *Random Forest* (RF) lebih baik akurasinya dibandingkan *Support Vector Machine* (SVM) dalam uji coba ini karena menunjukan peningkatan akurasi yang signifikan.

Pada penggunaan metode *K-fold* tanpa *stemming*, algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,993 (99,3%), setelah dibantu dengan metode OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,988 (98,8%). Penggunaan metode *K-fold* dengan *stemming* algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,994 (99,4%), setelah

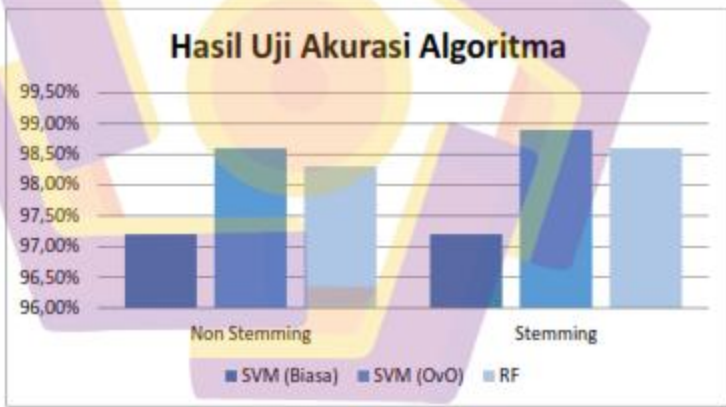
dibantu dengan metode menggunakan metode pengklasifikasian multiclass OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,989 (98,9%). Maka diambil kesimpulan dari 2 algoritma ini dalam penggunaan *K-fold* antara dilakukan *stemming* dengan yang tidak dilakukan *stemming* terdapat persamaan antara *Support Vector Machine* (SVM) dan *Random Forest* (RF) terjadi kenaikan akurasi yang tidak terlalu banyak akan tetapi mendapatkan model lebih stabil dalam mengklasifikasikan dataset kedua.

Pada metode *Confussion Matrix* tanpa *stemming*, algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,986 (98,6%), setelah dibantu dengan metode menggunakan metode pengklasifikasian *multiclass* yaitu metode OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%). Sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,991 (99,1%). Penggunaan *Confussion Matrix* dengan *stemming* algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi sebesar 0,989 (98,9%), setelah dibantu dengan metode pengklasifikasian *multiclass* yaitu metode OvO (*One versus One Classification*) mendapatkan nilai 0,995 (99,5%), sedangkan pada algoritma *Random Forest* (RF) mendapatkan nilai 0,982 (98,2%). Maka diambil kesimpulan bahwa dalam penggunaan metode *Confussion Matrix* antara dilakukan *stemming* dengan yang tidak pada *Support Vector Machine* (SVM) terjadi kenaikan akurasi berbeda dengan *Random Forest* (RF) yang mengalami penurunan nilai akurasi setelah dilakukan *stemming*. Jadi pada uji coba

confussion matrix algoritma *Support Vector Machine* (SVM) memiliki nilai akurasi yang stabil, baik dilakukan stemming dan tidak dilakukan *stemming*.

4.2.3.3 Analisis Perbandingan Data Keseluruhan

Hasil akurasi pada data pelanggaran pertama di Univeristas Darussalam (UNIDA) Gontor dengan menggunakan algoritma *Support Vector Machine* (SVM) yang biasa, kurang efektif apabila tidak dilakukan proses Stemming dibanding dengan penggunaan algoritma *Random Forest* (RF) yang tidak dilakukan proses Stemming, dalam kondisi parameter default, hal ini dapat dilihat dari perbandingan akurasinya pada data pertama yaitu pada penggunaan uji coba stemming pada gambar 4.22

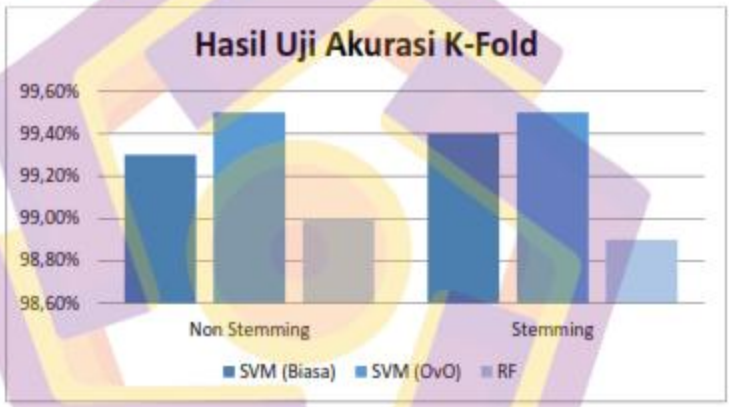


Gambar 4. 22 Perbandingan Uji Coba Algoritma

Dijelaskan uji coba algoritma klasifikasi jika di uji pada penggunaan *stemming*, antara algoritma *Support Vector Machine* (SVM) yang tidak di *stemming* dengan yang dilakukan *stemming* memiliki nilai akurasi yang sama yaitu 97,2% jadi tak ada perubahan akurasi, akan tetapi setekah dibantu dengan metode OvO nilai akurasinya naik menjadi 98,6% dan jika dilakukan stemming

terdapat kenaikan nilai akurasi bertambah menjadi 98,9%. Begitu halnya dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya hanya 98,3%, akan tetapi jika sudah di lakukan *stemming* nilai akurasinya bertambah menjadi 98,6%, menunjukan bahwa pengaruh penggunaan *stemming* sangat signifikan hasilnya dalam meningkatkan akurasi,

Pada uji coba *K-fold Cross Validation* penggunaan *stemming* juga memberikan pengaruh, dilahat pada gambar 4.23 dibawah

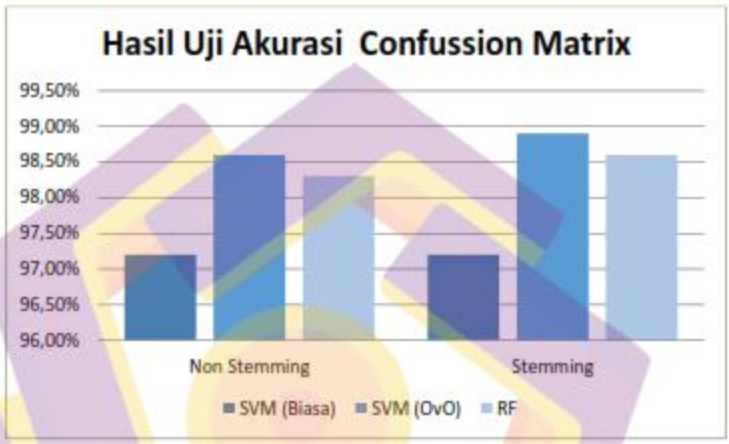


Gambar 4. 23 Perbadingan uji akurasi dengan *K-fold Cross Validation*

Dijelaskan pada uji coba K-Fold diatas antara algoritma *Support Vector Machine* (SVM) yang tidak di *stemming* dengan yang dilakukan *stemming* memiliki nilai akurasi yang berbeda yaitu 99,3% dengan 99,4%, namun ketika dilakukan akan tetapi setelah dibantu dengan metode OvO meningkat nilai akurasinya menjadi 99,5% baik yang tidak di *stemming* dengan yang dilakukan *stemming*. Berbeda dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya 99% turun akurasinya menjadi 98,9% setelah dilakukan *stemming*. Maka pada uji coba ini penggunaan *stemming* memberi hasil yang

efisien pada *Support Vector Machine* (SVM) namun kurang efisien untuk *Random Forest* (RF).

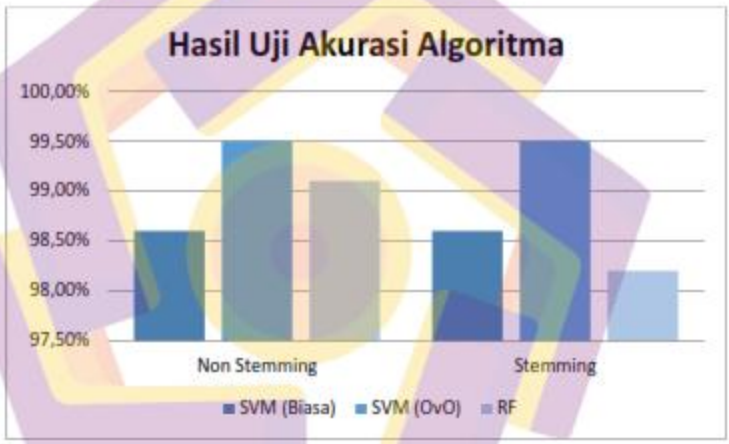
Pada uji coba *Confussion Matrix* penggunaan *stemming* juga memberikan pengaruh, dilahat pada gambar 4.24 dibawah



Gambar 4. 24 Perbadingan uji akurasi dengan *Confussion Matrix*

Dijelaskan uji coba *Confussion Matrix* jika di uji pada penggunaan *stemming*, antara algoritma *Support Vector Machine* (SVM) yang tidak di *stemming* dengan yang dilakukan *stemming* memiliki nilai akurasi yang sama yaitu 97,2% jadi tak ada perubahan akurasi, akan tetapi jika dibantu dengan metode OvO nilai akurasinya naik menjadi 98,6% dan jika dilakukan stemming terdapat kenaikan nilai akurasi bertambah menjadi 98,9%. Begitu halnya dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya hanya 98,3%, akan tetapi jika sudah di lakukan *stemming* nilai akurasinya bertambah menjadi 98,6%, menunjukan bahwa pengaruh penggunaan *stemming* sangat signifikan hasilnya dalam meningkatkan akurasi

Pada hasil akurasi data pelanggaran kedua di Univeristas Darussalam (UNIDA) Gontor dengan menggunakan algoritma *Support Vector Machine* (SVM) yang biasa, juga kurang efektif apabila tidak dilakukan proses Stemming dibanding dengan penggunaan algoritma *Random Forest* (RF) yang tidak dilakukan proses Stemming, dalam kondisi parameter default, hal ini dapat dilihat dari perbandingan akurasinya pada data pertama yaitu pada penggunaan uji coba stemming pada gambar 4.25

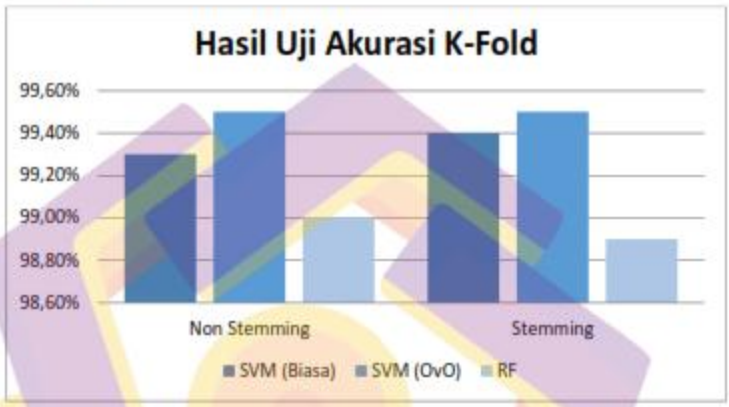


Gambar 4. 25 Perbandingan Uji Coba Algoritma

Dijelaskan uji coba algoritma klasifikasi jika di uji pada penggunaan *stemming, Support Vector Machine* (SVM) yang tidak di *stemming* dengan yang dilakukan *stemming* memiliki nilai akurasi yang sama yaitu 98,6%, namun ketika dibantu dengan metode OvO meningkat nilai akurasinya menjadi 99,5% baik yang tidak di *stemming* dengan yang dilakukan *stemming*. Berbeda dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya 99,1% turun akurasinya menjadi 98,2% setelah dilakukan *stemming*. Maka pada uji

coba ini penggunaan *stemming* memberi hasil yang efisien pada *Support Vector Machine* (SVM) namun kurang efisien untuk *Random Forest* (RF).

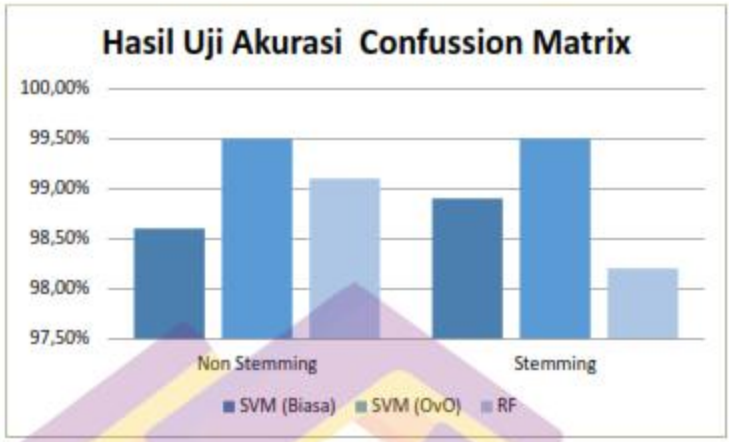
Pada uji coba K-fold Cross Validation penggunaan *stemming* juga memberikan pengaruh, dilahat pada gambar 4.26 dibawah



Gambar 4. 26 Perbadingan uji akurasi dengan *K-fold Cross Validation*

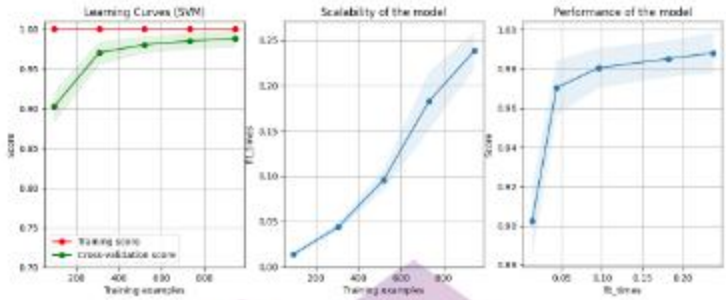
Dijelaskan pada uji coba K-Fold diatas antara algoritma *Support Vector Machine* (SVM) yang tidak di *stemming* yaitu 99,3% dengan yang dilakukan *stemming* memiliki nilai akurasi yaitu 99,4% jadi terdapat peningkatan akurasi, akan tetapi jika dibantu dengan metode OvO baik yang dilakukan *stemming* dengan yang tidak nilai akurasinya naik menjadi 99,5%. Begitu halnya dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya hanya 98,8%, akan tetapi jika sudah di lakukan *stemming* nilai akurasinya bertambah menjadi 98,9%, menunjukan bahwa pengaruh penggunaan *stemming* sangat signifikan hasilnya dalam meningkatkan akurasi

Pada uji coba *Confussion Matrix* penggunaan *stemming* juga memberikan pengaruh, dilahat pada gambar dibawah 4.27



Gambar 4. 27 Perbadingan uji akurasi dengan *Confussion Matrix*

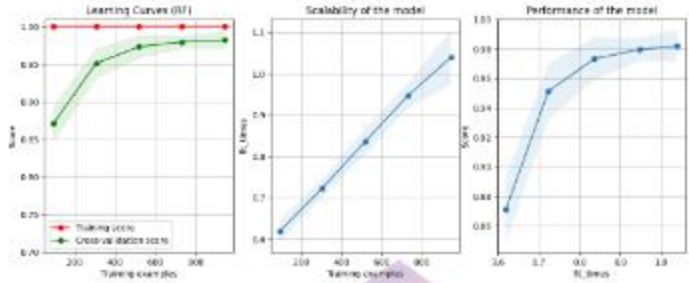
Dijelaskan uji coba *Confussion Matrix* jika di uji pada penggunaan *stemming*, antara algoritma *Support Vector Machine* (SVM) yang tidak di *stemming* dengan yang dilakukan *stemming* memiliki nilai akurasi yang berbeda yaitu 98,6% dengan 98,9%, namun ketika dibantu dengan metode OvO meningkat nilai akurasinya menjadi 99,5% baik yang tidak di *stemming* dengan yang dilakukan *stemming*. Berbeda dengan *Random Forest* (RF) yang mana sebelum dilakukan *stemming* akurasinya 99,1% turun akurasinya menjadi 98,2% setelah dilakukan *stemming*. Maka pada uji coba ini penggunaan *stemming* memberi hasil yang efisien pada *Support Vector Machine* (SVM) namun kurang efisien untuk *Random Forest* (RF).



Gambar 4. 28 Kurva Efektifitas dari SVM data pertama

Pada *learning curve* diketahui akurasi dari proses *training score* tanpa *cross validation* relatif stabil dari data ke - 1 sampai data terakhir, berbeda dengan proses *training* dengan menggunakan *cross validation* yang memulai dengan hasil yang kurang maksimal dari data ke- 1 tetapi, terus meningkat seiring dengan bertambahnya data yang lebih dari 800 saat *training*. Untuk kurva *scalability of the model*, model membutuhkan waktu 0,24 menit untuk melakukan *training* seluruh data yang jumlahnya lebih dari 800. Pada kurva *performance of the model* dilihat bahwa hasil akurasi yang didapat semakin meningkat seiring dengan lamanya *training* yang dilakukan terhadap data sehingga mendapat akurasi diatas 0,98 (98%) dengan waktu diatas 0,20 menit.

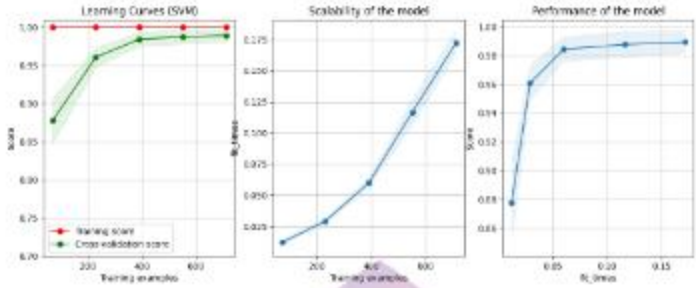
Pada uji coba selanjutnya dengan algoritma *Random Forest* (RF) dengan tiga analisis performa model dapat dilihat pada gambar 4.29



Gambar 4. 29 Kurva Kurva Efektifitas dari RF data pertama

Dari *learning curve* milik *Random Forest* (RF) dapat diketahui akurasi dari proses *training score* tanpa *cross validation* relatif stabil dari data ke - 1 sampai data terakhir, berbeda dengan proses *training* dengan menggunakan *cross validation* yang memulai dengan hasil yang kurang maksimal tetapi, terus meningkat seiring dengan bertambahnya data yang lebih dari 800 saat *training*. Sedangkan pada kurva *scalability of the model*, model membutuhkan waktu hampir 1 menit lebih untuk men – *training* seluruh data. Dari kurva *performance of the model* dapat diambil kesimpulan bahwa hasil yang didapat semakin meningkat seiring dengan lamanya *training* yang dilakukan terhadap data sehingga mendapat akurasi 0,98 (98%) dengan waktu diatas 1 menit.

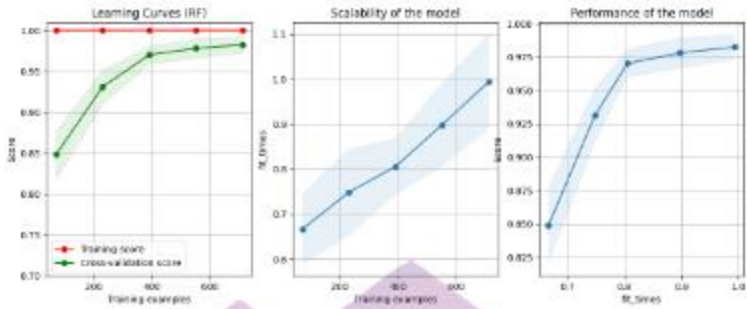
Pada uji coba selanjutnya yaitu menggunakan data kedua, menggunakan algoritma *Support Vector Machine* (SVM) dengan tiga analisis performa model dapat dilihat dari secara berturut – turut pada gambar 4.30 di bawah ini.



Gambar 4. 30 Kurva Efektifitas dari SVM data kedua

Dari *learning curve* dapat diketahui akurasi dari proses *training score* tanpa *cross validation* relatif stabil dari data ke awal sampai data terakhir, berbeda dengan proses *training* dengan menggunakan *cross validation* yang memulai dengan hasil dibawah nilai 0,90 terus meningkat seiring menjadi diatas 0,95 dengan bertambahnya data saat *training*. Sedangkan dari kurva *scalability of the model*, model membutuhkan waktu hampir 0,17 menit untuk men – *training* seluruh data diatas 600. Dan dari kurva *performance of the model* dapat diambil kesimpulan bahwa hasil yang didapat dengan waktu 0 menit hingga 0,15 menit bisa mendapatkan akurasi diatas 0,98 semakin meningkat seiring dengan lamanya *training* yang dilakukan terhadap data.

Selanjutnya menggunakan algoritma *Random Forest* (RF) dengan ketiga analisis performa model dapat dilihat dari secara berturut – turut pada gambar 4.31



Gambar 4. 31 Kurva Efektifitas dari RF data kedua

Dapat diketahui akurasi dari proses *training* tanpa *cross validation* relatif stabil dari data ke - 1 sampai data terakhir, berbeda dengan proses *training* dengan menggunakan *cross validation*, memulai dengan hasil pada nilai 0,90 terus meningkat seiring menjadi diatas 0,95 dari data lebih dari 600 tetapi, seiring dengan bertambahnya data saat *training*. Sedangkan dari kurva *scalability of the model*, model membutuhkan waktu hampir 1 menit untuk men – *training* seluruh data. Dari kurva *performance of the model* dapat diambil kesimpulan bahwa hasil yang didapat semakin meningkat seiring dengan lamanya *training* yang dilakukan terhadap data, sehingga mendapat akurasi 0,98 (98%) dengan waktu 1 menit.

Setelah diambil kesimpulan dari dua data yang berbeda dengan 3 metode diatas, menunjukan bahwa algoritma *Support Vector Machine* (SVM) tidak lebih baik dalam mengklasifikasikan dari pada *Random Forest* (RF), akan tetapi setelah dibantu dengan metode menggunakan metode OvO, *Support Vector Machine* (SVM) menjadi lebih baik dari pada *Random Forest* (RF) pada dateset pertama dan pada dataset kedua juga ditemukan hasil yang sama.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan hasil analisis yang telah dilakukan, diperoleh dari pembuatan model untuk pengklasifikasian data pelanggaran di Universitas Darussalam (UNIDA) Gontor ini dapat diambil beberapa kesimpulan sebagai berikut :

1. Didapatkan akurasi pada metode klasifikasi Algoritma *Support Vector Machine* mendapatkan hasil terbaik dari penggunaan *Random Forest* dalam mengklasifikasikan dataset kedua, hal ini ditunjukkan dengan tingkat akurasinya pada metode tanpa *stemming* menghasilkan *Support Vector Machine* memiliki akurasi 98,6% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 98,3%. Untuk penggunaan *stemming Support Vector Machine* menghasilkan akurasi 97,2% dibandingkan dengan *Random Forest* dengan akurasi lebih tinggi yaitu 98,6%. Pada data kedua *Support Vector Machine naik* menjadi 99,5% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 99,1%, sedangkan dengan metode *stemming, Support Vector Machine* memiliki akurasi 99,5% dibandingkan dengan *Random Forest* dengan akurasi 98,2% Pada data kedua menghasilkan nilai akurasi yang lebih baik dengan metode *stemming*.
2. Didapatkan akurasi pada metode klasifikasi Algoritma *Support Vector Machine* mendapatkan hasil terbaik dari penggunaan *Random Forest* dalam mengklasifikasikan dataset pertama, hal ini ditunjukkan dengan tingkat akurasinya pada metode *K-fold Cross Validation* tanpa *stemming*

menghasilkan akurasi *Support Vector Machine* 99,5% dibandingkan dengan *Random Forest* dengan akurasi 99%, sedangkan dengan metode *stemming*, *Support Vector Machine* menghasilkan akurasi 99,5% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 98,9%. Pada data kedua mengalami penurunan, yang didapatkan akurasi tanpa *stemming*, *Support Vector Machine* menghasilkan 98,6% dibandingkan dengan *Random Forest* dengan akurasi 98,3%, sedangkan dengan metode *stemming*, *Support Vector Machine* menghasilkan akurasi yaitu 98,9% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 98,6%. Pada data pertama menghasilkan nilai akurasi yang lebih baik dengan metode tanpa *stemming*.

3. Didapatkan akurasi pada metode klasifikasi Algoritma *Support Vector Machine* mendapatkan hasil terbaik dari penggunaan *Random Forest* dalam mengklasifikasikan dataset kedua, hal ini ditunjukkan dengan tingkat akurasinya pada metode *Confuss Matrix* tanpa *stemming*, *Support Vector Machine* menghasilkan akurasi 98,6% dibandingkan dengan *Random Forest* yang lebih unggul dengan akurasi 98,3%. Pada metode *stemming*, *Support Vector Machine* menghasilkan akurasi 98,9% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 98,6%. Pada data kedua menghasilkan nilai akurasi yang lebih baik pada metode tanpa *stemming*. *Support Vector Machine* naik menghasilkan akurasi 99,5% dibandingkan dengan *Random Forest* yang hanya memiliki akurasi 99,1%, sedangkan dengan metode *stemming*, *Support Vector Machine* menghasilkan akurasi yaitu 99,5% dibandingkan dengan *Random Forest* yang hanya memiliki

akurasi 98,2%. Pada data kedua menghasilkan nilai akurasi yang lebih baik dengan metode *stemming*.

4. Hasil perbandingan kinerja kedua algoritma dengan 3 metode menunjukkan bahwa, pada data pertama dan kedua nilai rata-rata dari algoritma *Support Vector Machine* lebih baik dari algoritma *Random Forest*.

5.2. Saran

Dari hasil penelitian, penulis menyadari bahwa penelitian ini masih jauh dari kata sempurna, sehingga perlu ada adanya pengembangan lebih lanjut untuk menghasilkan hasil yang lebih maksimal, beberapa saran yang dapat diambil adalah sebagai berikut :

1. Pada penelitian selanjutnya, penelitian dapat dikembangkan dengan menggunakan algoritma lain, seperti, C.45, Naïve Bayes, KNN (*K – Nearest Neighbour*) dan lainnya.
2. Pada penelitian berikutnya, penggunaan metode pada algoritma *Support Vector Machine* (SVM) tidak hanya *One Vs One* (OvO) saja tapi juga bisa menggunakan metode *One Vs All* (OvA) atau *One Vs Rest* (OvR).
3. Penggunaan dataset selanjutnya agar diperbanyak jumlah datanya dan perlu dilakukan pengecekan data, yang terdapat pada sistem agar data yang akan di uji tidak cacat atau datanya tidak lengkap isinya.
4. Pada penelitian selanjutnya pada proses *text preprocessing* agar dioptimalkan lagi secara keseluruhan.

DAFTAR PUSTAKA

PUSTAKA BUKU

Jr.McLeod.R dan Schell. G.P., Inc 2007, *Management Information System. 10th ed,* Pearson Education, University of Texas at Austin .

Sianipar. R.H, Wadi. H. 2015. *Pemrograman Python (Teori Dan Implementasi).* Penerbit Informatika. Kota Yogyakarta.

Trevor.H., Robert.T., dan Friedman.J., 2009. *The Elements of Statistical Learning Data Mining, Inference, and Prediction*. California : Springer.

Prasetyo, E., 2014. *Data Mining-Mengolah Data Menjadi InformasiMenggunakan Matlab*. Yogyakarta: Penerbit Andi.

PUSTAKA MAJALAH, JURNAL ILMIAH ATAU PROSIDING

M.Nurfitriyanti, 2014, *Pengaruh Kreativitas Dan Kedisiplinan Mahasiswa*, vol. 4, no. 3, pp. 219–26, Universitas Indraprasta PGRI.

Binning, Kevin R., Cook, Jonathan E., Purdie-greenaway, Valerie., Garcia, Julio., Chen, Susie., Apfel, Nancy., Sherman, David K. Cohen., 2019 , *Bolstering trust and reducing discipline incidents at a diversemiddle school : How self-a ffi rmation a ff ects behavioral conduct during the transition to adolescence*, vol. 75, pp. 74–88,University of Pittsburgh, Department of Psychology, Learning Research and Development Center, United States of America.

Saefudin.A., 2018, *Metode Data Mining Untuk Seleksi Calon Mahasiswa Pada Penerimaan Mahasiswa Baru Di Universitas Pamulang,* Vol. 10, no.1, Pp.25-36.

Ashraf.A, Anwer.S, and Ghufran Khan.M, 2018, *A Comparative Study of Predicting Student ' s Performance by use of Data Mining Techniques,* National University of Computer & Emerging Sciences, pp. 122–36, Department of Computer Sciences, Pakistan.

Alfarobi.I, Tutopoly.T.A, dan Suryanto.A, 2018 "Komparasi Algoritma C4.5, Naive Bayes, Dan *Random Forest* Untuk Klasifikasi Data Kelulusan

Mahasiswa Jakarta", *Jurnal Manajemen Dan Teknologi Pendidikan*, vol. 4, no. Jakarta ISSN : 2477-1724, pp. 1–14.

Irawan.F. dan Samopa.F., *2018, A Comparative Assessment of Random Forest and SVM Algorithms, UsingCombination of Principal Component Analysis and SMOTE For Accounts Receivable Seamless Prediction, case study company X in Surabaya*, Proceedings of The 2nd International Seminar Of Contemporary Research OnBusiness & Management, Universitas Warmadewa Bali, July 30th, 2018.

Irwansyah,I., Rosiyadi.D., 2019, *Perbandingan Kinerja Algoritma K-Nearest Neighbor, Naïve Bayes Classifier dan Support Vector Machine dalam Klasifikasi Tingkah Laku Bully pada Aplikasi Whatsapp, Vol. 12, no.2, Pp.101-111.*

Drajana.R.I. Colanus , 2017, *Metode Support Vector Machine Dan Forward Selection Prediksi Pembayaran Pembelian Bahan Baku Kopra, ILKOM* Jurnal Ilmiah Volume 9 Nomor 2 .

Momade.M.Hamza, Shahid.S., Rosli.M.bin Hainin, Nashwan.M.Salem, and Umar.A.Tahir , 2020, *Modelling labour productivity using SVM and RF: a comparative study on classifiers performance*, Universiti Teknologi Malaysia (UTM), Johor Bahru, Malaysia.

Suyadi, 2017, *Analisis Perbandingan Algoritma Decision Tree (C4 . 5) Dan K-Naive Bayes Untuk Mengklasifikasi Penerimaan Mahasiswa Baru Tingkat Universitas*, vol. 2, no. 1, pp. 59–68.

Sari. D., 2016, *Komparasi 5 Metode Algoritma Klasifikasi Data Mining Pada Prediksi Keberhasilan Pemasaran Produk Layanan Perbankan*, vol. XIII, no. 1, pp. 60–6.

Karyadi.S., Yasin.H., and Mukid.M. Abdul., 2016, "*Analisis Kecenderungan Informasi Dengan Menggunakan Metode Text Mining* (Studi Kasus: Akun twitter @detikcom)", *Jurnal Gaussian*, vol. 5, pp. 763–70.

Budi.S., c, *Text Mining Untuk Analisis Sentimen Review Film Menggunakan Algoritma K-Means*, vol. 16, no. 1, pp. 1–8.

Arisandy.D. Luki, Wahanggara.V., dan Rahayu.Y. Dwi, 2016, *Analisis perbandingan algoritma naive bayes dan algoritma c4.5 untuk klasifikasi multi data*, no. 1310651061.

Nugroho.Y.Sulistyo dan Emiliyawati.N., 2017, *Sistem Klasifikasi Variabel Tingkat Penerimaan Konsumen Terhadap Mobil Menggunakan Metode Random Forest.* Jurnal Teknik Elektro Vol. 9 No. 1.

Kurniawan.A.Fransiska dan Kurniati.A. Prima, 2011, Regression Tree, *Analisis Dan Implementasi Random Forest Dan Classification Dan Regression Tree (Cart) Untuk Klasifikasi Pada Misuse Intrusion Detection System.*

Ramana, B. V., Babu, S. P., & Venkateswarlu, N. B. 2011. *A Critical Study Of Selected Classification Algorithms For Liver Disease Diagnosis*. International Journal Of Database Management Systems , Vol. 3.

Munawarah, Raudlatul. 2016. *Implementasi Metode Support Vector Machine untuk Mendiagnosa Penyakit Hepatitis*. Program S-1 Ilmu Komputer, Universitas Lambung Mangkurat: Banjarbaru.

Nurjannah.M., Hamdani , S. P., & Astuti, I. Fitri. 2013. *Penerapan Algoritma Term Frequency-Inverse Document Frequency (Tf-Idf) Untuk Text Mining*, Vol. 8.

Ma'arif, Aziz.A., 2015. Penerapan Algoritma Tf-Idf Untuk Pencarian Karya Ilmiah. Jurnal. Jurusan Teknik Informatika. Fakultas Ilmu Komputer.,Universitas Dian Nuswantoro, Semarang.

Nurjannah.M., Hamdani,& Astuti.F. Inda, 2013. *Penerapan Algoritma Term Frequency-Inverse Document Frequency (Tf-Idf) Untuk Text Mining, Jurnal Informatika Mulawarman* , Vol.8, No.3, Pp.110.

Pandie, Emerensye S. Y. 2012. *Implementasi Algoritma Data mining K-Nearest Neighbour (KNN) Dalam Pengambilan Keputusan Pengajuan Kredit*. Jurusan Ilmu Komputer, Fakultas Sains dan Teknik, Universitas Nusa Cendana, Kupang

Perkasa.T.Richard , Widyantara.H.,& Susanto.P., 2014. *Rancang Bangun Pendeteksi Gerak Menggunakan Metode Image Subtraction Pada Single Board Computer (Sbc)*, Vol.3, No.2, Pp.90-97.

Fibrianda.F.Mercury dan Bhawiyuga, Adhitya, 2018, "*Analisis Perbandingan Akurasi Deteksi Serangan Pada Jaringan Komputer Dengan Metode Naïve Bayes Dan Support Vector Machine (SVM)*," Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer 2, no. 9 : 3112–23.

Rezalina, Oppie. 2016. *perbandingan algoritma stemming nazief & adriani, porter dan arifin setiono untuk dokumen teks bahasa indonesia*. Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah, Jember.

PUSTAKA LAPORAN PENELITIAN

Yahya.S.Amri, 2018, *Klasifikasi Ketepatan Lama Studi Mahasiswa Menggunakan Metode Support Vector Machine Dan Random Forest (Studi Kasus : Data Lama Studi Alumni Universitas Islam Indonesia Tahun Kelulusan 2000-2017)*, Universitas Islam Indonesia, Yogyakarta.

Fachruddin.M.Idrus, 2018, *Perbandingan Metode Random Forest Classification Dan Support Vector Machine Untuk Deteksi Epilepsi Menggunakan Data Rekaman Electroencephalograph (EEG)*, Skripsi, Institut Teknologi Sepuluh Nopember, Surabaya.

Putra, Arif. A.Auliaguntary, 2016, *Implementasi Text Summarization Menggunakan Metode Vector Space Model pada Artikel Berita Bahasa Indonesia*, Skripsi. Jurusan Teknik Informatika, Fakultas Teknik dan Ilmu Komputer, Universitas Komputer Indonesia, Bandung.

Pandie, Emerensye S. Y. 2012. *Implementasi Algoritma Data mining K-Nearest Neighbour (KNN) Dalam Pengambilan Keputusan Pengajuan Kredit*. Jurusan Ilmu Komputer, Fakultas Sains dan Teknik, Universitas Nusa Cendana, Kupang

Rodiyansyah, Fajar.S., dan Winarko.E., 2013, *Klasifikasi Posting Twitter Kemacetan Lalu Lintas Kota Bandung Menggunakan Naive Bayesian Classification*, Universitas Pendidikan Indonesia, Yogyakarta.

Haristu.A.Reinardus, 2019, *Penerapan Metode Random Forest Untuk Prediksi Win Ratio Pemain Player Unkown Battleground* , Universitas Sanata Dharma, Yogyakarta.

Sitepu.Br.Natalina, 2019, *Analisis Algoritma Decision Tree dengan Algoritma Random Forest pada Discretize By Frequency*, Universitas Sumatera Utara, Medan.

PUSTAKA ELEKTRONIK

Utami, E.; Istiyanto, J.E.; Hartati, S.; Marsono; Ashari, A., 25 November 2009, Developing Transliteration Pattern of Latin Character Text Document Algorithm Based on Linguistics Knowledge of Writing Javanese Script, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5417267