Tesis-Sidang Akhir - EF235401

DETEKSI HATESPEECH MENGGUNAKAN NEURAL NETWORK – NAÏVE BAYES CLASSIFIER DARI DATASET TWITTER

REZA WAHYU RAMADHAN
6025222007

Dosen Pembimbing
Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D

PROGRAM MAGISTER
BIDANG KEAHLIAN KOMPUTASI BERBASIS JARINGAN
PROGRAM STUDI S2 TEKNIK INFORMATIKA
DEPARTEMEN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2024

Tesis-Sidang Akhir - EF235401

DETEKSI HATESPEECH MENGGUNAKAN NEURAL NETWORK – NAÏVE BAYES CLASSIFIER DARI DATASET TWITTER

REZA WAHYU RAMADHAN
6025222007

Dosen Pembimbing
Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D

PROGRAM MAGISTER
BIDANG KEAHLIAN KOMPUTASI BERBASIS JARINGAN
PROGRAM STUDI S2 TEKNIK INFORMATIKA
DEPARTEMEN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2024

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN THESIS

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom.)
di
Institut Teknologi Sepuluh Nopember

Oleh:
Reza Wahyu Ramadhan
NRP. 6025222007

Tanggal Ujian: 16 Juli 2024
Periode Wisuda:

Disetujui oleh:

1 Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D
NIP. 198106202005011003
(Pembimbing 1) .....

2 Hudan Studiawan, S.Kom., M.Kom., Ph.D.
NIP. 198705112012121003
(Penguji 1) .....

3 Bagus Jati Santoso, S.Kom, Ph.D
NIP. 198611252018031001
(Penguji 2) .....

4 Dr. Baskoro Adi Pratomo, S.Kom., M.Kom.
NIP. 198702182014041001
(Penguji 3) .....

Kepala Departemen Teknik Informatika

Prof. Dr. Eng. Nur Ulfa Maulidevi, S.Kom., M.Kom.
NIP. 197512202001122002

[Halaman ini sengaja dikosongkan]

PERNYATAAN ORISINALITAS

Yang bertanda tangan dibawah ini

Nama Mahasiswa (NRP)	:	Reza Wahyu Ramadhan (6025222007)
Doeen Pembimbing 1 (NIP)	:	Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D (198106202005011003)
Program Studi	:	S2 Teknik Informatika
Departemen	:	Departemen Teknik Informatika
Fakultas	:	Fakultas Teknologi Elektro dan Informatika Cerdas

Dengan ini menyatakan bahwa Tesis yang berjudul "**Deteksi Hatespeech Menggunakan Neural Network – Naïve Bayes Classifier Dari Dataset Twitter**" adalah hasil karya sendiri, bersifat orisinil, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Apabila di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

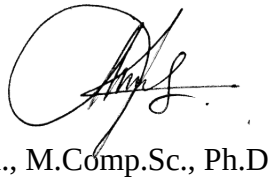
Surabaya, 25 Juni 2024

Reza Wahyu Ramadhan

NRP 6025222007

Mengetahui

Dosen Pembimbing 1



Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D

NIP. 198106202005011003

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur yang sedalam-dalamnya penulis panjatkan kehadirat Tuhan Yang Maha Esa atas segala berkat dan limpahan rahmat-Nya sehingga penulis dapat menyelesaikan laporan penelitian tesis dengan judul "**Deteksi Hatespeech Menggunakan Neural Network – Naïve Bayes Classifier Dari Dataset Twitter**"

Tujuan dari penulisan tesis ini adalah untuk memenuhi syarat dalam mencapai derajat Magister Komputer pada Program Studi Pasca Sarjana Institut Teknologi Sepuluh Nopember Surabaya.

Di dalam proses penulisan tesis ini, penulis banyak mendapatkan bimbingan dan dukungan dari berbagai pihak sehingga penulisan tesis ini dapat terselesaikan tepat waktu. Oleh karena itu, ucapan terimakasih yang sebesarbesarnya dan penghargaan setinggi-tingginya penulis sampaikan kepada :

1. Bapak Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D selaku dosen pembimbing
2. Bapak Dr. Ahmad Saikhu, S.Si,MT selaku kepala program studi pasca sarjana Institut Teknologi Sepuluh Nopember Surabaya
3. Ibu Dr. Eng. Chastine Fatichah, S.Kom,, M.Kom selaku kepala departemen Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya
4. Dr I Ketut Eddy Purnama ST MT selaku Dekan Fakultas Teknologi Elektro dan Informatika Cerdas Institut Teknologi Sepuluh Nopember Surabaya
5. Bambang Pramujati, S.T., M.SC.ENG., PH.D selaku rector Institut Teknologi Sepuluh Nopember Surabaya

Penulis menyadari bahwa tesis ini masih jauh dari sempurna. Untuk itu saran beserta kritikan yang membangun sangat diharapkan. Semoga karya ini dapat bermanfaat bagi kita semua.

Surabaya, 25 Juni 2024

Reza Wahyu Ramadhan

[Halaman ini sengaja dikosongkan]

Deteksi Hatespeech Menggunakan Neural Network – Naïve Bayes Classifier Dari Dataset Twitter

Nama Mahasiswa	:	Reza Wahyu Ramadhan
NRP	:	6025222007
Pembimbing	:	Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D.

ABSTRAK

Indonesia merupakan salah satu pengguna media sosial terbesar di dunia, dengan Twitter/X menjadi platform yang sangat populer dengan Indonesia menempati peringkat keenam sebagai pengguna terbesar *Twitter/X* di dunia. Namun, penggunaan media sosial yang masif ini juga memunculkan tantangan signifikan terkait ujaran kebencian. Ujaran kebencian di media sosial menjadi masalah yang mengkhawatirkan karena dapat menyebar dengan cepat dan mempengaruhi audiens secara luas yang menyebabkan tindakan kekerasan di dunia nyata. Tetapi masih cukup minim penelitian dilakukan untuk membentuk sebuah model yang lebih baik pada Bahasa Indonesia.

Naive Bayes (NB) adalah salah satu model tradisional yang umum digunakan untuk melakukan deteksi terhadap teks. Model NB adalah sebuah model dengan basis probabilistik dengan asumsi independensi antar fitur, sedangkan pada klasifikasi data berbentuk teks hubungan antar fitur akan menjadi penting satu dengan yang lainnya. Model Artificial Neural Network (ANN) dapat digunakan untuk mengatasi kelemahan dari model NB. ANN mampu menangkap dan memodelkan hubungan non-linear antara fitur-fitur dalam data. Dengan menggunakan ANN untuk ekstraksi fitur, representasi fitur akan lebih baik, sehingga memungkinkan NB untuk bekerja dengan fitur-fitur yang lebih informatif dan relevan, meningkatkan performa dari model NB.

Dengan melakukan implementasi ekstraksi fitur ANN terhadap NB (ANN-NB) pada penelitian ini berhasil membuktikan peningkatan model NB baik pada skenario klasik untuk membagi dataset menjadi 70:30 dan juga pada skenario *data stream mining.* Dengan nilai akurasi tertinggi didapat di angka 97,6% yang diukur menggunakan *confusion matrix*.

Kata Kunci : Analisis Sentimen, Naïve Bayes Classifier, Neural Network, Adaptif Model

[Halaman ini sengaja dikosongkan]

Hatespeech Detection Using Neural Network – Naïve Bayes Classifier with Twitter Dataset

By	:	Reza Wahyu Ramadhan
Student Identity Number	:	6025222007
Supervisor	:	Ir. Ary Mazharuddin Shiddiqi, S.Kom., M.Comp.Sc., Ph.D.

ABSTRACT

Indonesia is one of the largest social media users in the world, with Twitter/X being a particularly popular platform, ranking sixth in the world for the number of users. However, this massive use of social media also presents significant challenges related to hate speech. Hate speech on social media is a concerning issue because it can spread quickly and widely influence audiences, potentially leading to real-world violence. Despite this, there is still a lack of research aimed at developing better models for the Indonesian language.

Naive Bayes (NB) is one of the traditional models commonly used for text detection. NB is a probabilistic model based on the assumption of feature independence. However, in text data classification, the relationships between features are important. The Artificial Neural Network (ANN) model can be used to overcome the weaknesses of the NB model. ANN can capture and model non-linear relationships between features in the data. By using ANN for feature extraction, feature representation becomes more refined, allowing NB to work with more informative and relevant features, thereby enhancing the performance of the NB model.

Implementing ANN feature extraction on NB (ANN-NB) in this study has proven to improve the NB model's performance in both classical scenarios of splitting the dataset into 70:30 and in data stream mining scenarios. The highest accuracy achieved was 97.6% which measured by confusion matrix.

Keyword : Sentiment Analysis, Naïve Bayes Classifier, Neural Network, Adaptive Model

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

BAB 1
PENDAHULUAN

1.1. Latar Belakang

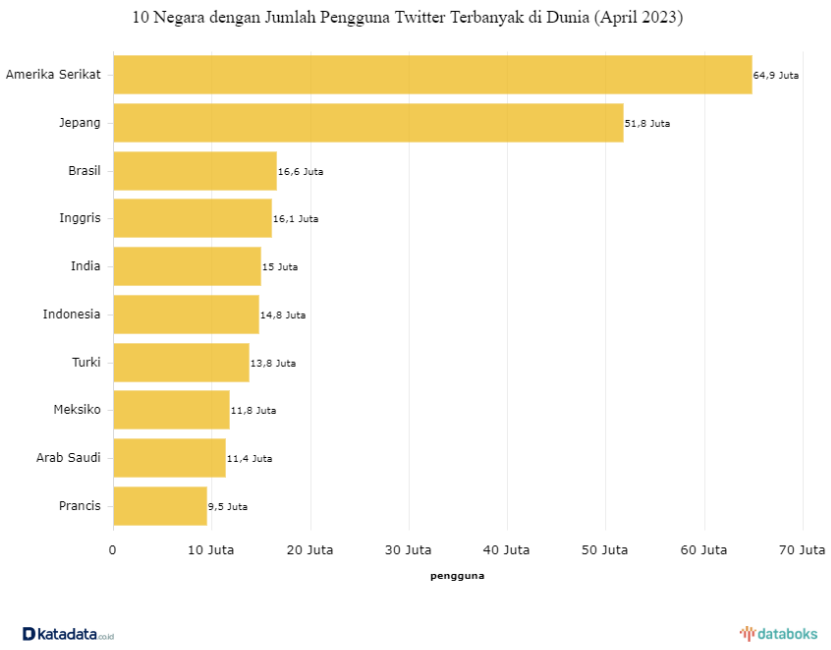
Teknologi telah digunakan secara masif oleh masyarakat di era digital seperti saat ini, salah satu cerminan dari penggunaan teknologi dapat dilihat dari pergeseran yang signifikan dalam cara berinteraksi dan berkomunikasi. Dengan semakin meluasnya penggunaan media sosial, platform *e-commerce*, dan situs berita *online*. Kompleksitas dan kemajuan pesat dalam perkembangan media, terutama media *online* yang mencakup berbagai *platform* mulai dari media berita hingga media sosial. *Platform* media sosial yang sangat beragam, seperti *Facebook*, *Twitter/X*, *Path*, *Instagram*, *Google+*, *Tumblr*, *Linkedin*, dan sebagainya [1].

Salah satu media sosial populer yang digunakan pengguna internet adalah *Twitter/X*. Pada Gambar 1. 1 ditunjukkan bahwa Indonesia merupakan pengguna *Twitter* ke 6 terbanyak dibawah Amerika Serikat, Jepang, Brasil, Inggris dan India. *Twitter/X* merupakan media sosial yang dapat digunakan secara gratis oleh penggunanya, dan sering dimanfaatkan untuk menuliskan pendapat atau opininya tentang suatu kasus atau berita. Pada *platform Twitter/X* setiap postingan disebut sebagai *tweet*. *Tweet* memiliki format berupa tulisan, gambar maupun tulisan dan gambar. *Platform* ini juga memberikan perlindungan privasi terhadap penggunanya sehingga pengguna dapat menyembunyikan identitas asli mereka ketika sedang menggunakannya.

Di Indonesia memiliki banyak ragam jenis pengguna *platform Twitter/X* seperti membuat akun untuk sharing tentang sejarah, atau akun dengan tujuan *review* dan memberikan pengetahuan tentang senjata api, akun pribadi, atau bahkan akun untuk melakukan informasi atau mungkin disinformasi tentang perpolitikan. Dengan banyaknya informasi yang ada pada *platform* ini, pengguna biasanya akan berselancar di media sosial untuk membaca atau mencari informasi tentang sesuatu.

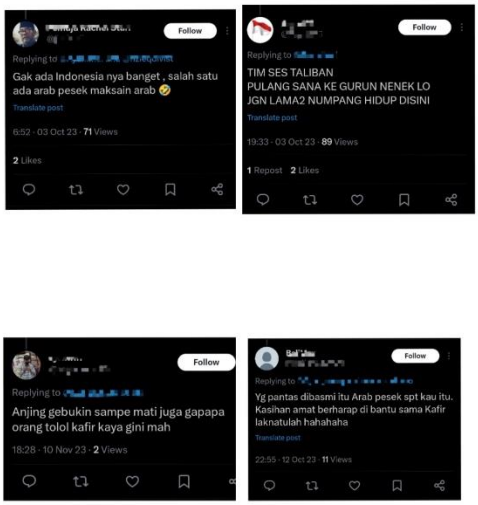
Dengan banyaknya kelebihan dan kemudahan yang dimiliki oleh *platform Twitter/X* ada beberapa pihak yang menyalagunakan *Twitter/X* untuk kepentingannya sendiri atau kelompok. Seperti melakukan *blackmail*, *doxing*, penipuan, memberikan *tweet* iklan yang tidak ingin dilihat oleh pengguna lain.

Selain itu akhir-akhir ini cukup lazim ditemukan *tweet* yang bersifat menyerang baik itu pribadi, ras, agama, dan budaya tertentu atau *hate speech*. Penyebaran *tweet* yang bersifat *hate speech* ini bahkan terkadang didasarkan oleh misinterpretasi informasi, atau berujung pada *hoax*.



Gambar 1. 1 Data Pengguna Twitter (https://databoks.katadata.co.id/)

Hate Speech didefinisikan sebagai ekspresi kata-kata atau ucapan yang mengandung rasa benci. Pada masa lampau, ekspresi kebencian sering kali diungkapkan secara langsung kepada individu yang menjadi target, tetapi dengan kemajuan waktu dan perkembangan Teknologi Informasi, *Hate Speech* dapat tersebar melalui berbagai media [1]. Dengan pesatnya penggunaan internet dan media sosial, muncul istilah Istilah '*online hate speech*' yang secara umum merujuk pada ungkapan diskriminatif yang dibagikan melalui internet dan ditujukan kepada kelompok-kelompok yang secara historis dianggap terpinggirkan berdasarkan karakteristik bawaan mereka [2]. Gambar 1. 2 menunjukkan contoh *hatespeech* yang ada di Indonesia.



Gambar 1. 2 Contoh Hatespeech di Indonesia

Komisi Nasional Hak Asasi Manusia (Komnas HAM) menjelaskan bahwa ujaran kebencian dapat ditujukan pada seseorang atau kelompok berdasarkan beberapa kategori bersifat inheren seperti agama, ras, atau gender. Selain itu, ujaran kebencian juga dapat dibagi menjadi beberapa tingkatan, seperti ujaran kebencian yang lemah, sedang, dan kuat. Target ujaran kebencian, kategori, dan tingkatnya perlu dideteksi untuk membantu lembaga tersebut dalam memutuskan kasus ujaran kebencian mana yang perlu diprioritaskan untuk ditangani [3]. Dan diperkuat di undang-undang pada pasal 27 ayat 2 berbunyi Setiap orang dengan sengaja dan tanpa hak menyebarkan informasi yang ditujukan untuk menimbulkan rasa kebencian atau permusuhan individu dan/atau kelompok masyarakat tertentu berdasarkan atas suku, agama, ras, dan antargolongan (SARA) [4].

Masalah penyebaran *hatespeech* di media sosial Indonesia merupakan permasalahan besar. Hal ini disebabkan oleh jumlah pengguna media sosial yang besar dan sering disalahgunakan untuk menyebarkan *hatespeech*. Membangun model sistem deteksi *hatespeech* berbahasa Indonesia lebih sulit dibandingkan dengan bahasa lain, Peneliti akan menghadapi kesulitan dalam mengurangi bias annotasi (karena tugas annotasi *hatespeech* bersifat sangat subjektif) untuk

mendapatkan *dataset* yang valid, terutama di Indonesia yang memiliki banyak etnis. Dibandingkan dengan negara-negara dengan etnis lebih sedikit, mengurangi bias annotasi di Indonesia mungkin lebih sulit karena perbedaan budaya etnis yang memengaruhi tingkat sensitivitas mereka dalam melakukan annotasi data [5].

Dalam membangun sebuah model klasifikasi terdapat beberapa metode yang cukup sering digunakan seperti Naïve Bayes (NB), *Support Vector Machine* (SVM) maupun menggunakan algoritma syaraf buatan seperti *Artificial Neural Network* (ANN). Penelitian [6] menunjukkan bagaimana hasil NB dalam mengklasifikasi *hatespeech* berbahasa Indonesia di *Twitter/X* dan menunjukkan bahwa NB dapat melakukan klasifikasi *hatespeech* dengan akurasi hingga 96% dengan menggunakan *Information Gain* pada tahap *pre-processing* data. Algorima NB juga memiliki kelebihan berupa hanya membutuhkan sumber daya komputasi yang kecil dalam melakukan pembangunan model dan melakukan klasifikasi. Tetapi Algoritma NB memiliki kekurangan yaitu bergantung pada asumsi bahwa fitur-fitur yang digunakan adalah independen satu sama lain. Meskipun asumsi ini seringkali tidak realistis di dunia nyata, NB masih dapat memberikan hasil yang baik, terutama pada dataset yang cukup besar. Namun, jika asumsi ini jelas tidak terpenuhi dalam suatu dataset, performa NB dapat terpengaruh.

Penelitian ini bertujuan untuk mengatasi kekurangan NB dengan melakukan pendekatan *artificial neural network* (ANN), yang memiliki kelebihan dalam membangun model dengan dapat mempelajari representasi fitur secara otomatis dari data sehingga memungkinkan untuk menangkap pola dan hubungan yang kompleks dan abstrak. Kemampuan ANN dalam membangun sebuah model dapat digabungkan dengan tahapan klasifikasi yang ada pada NB.

1.2. Perumusan Masalah

Rumusan masalah yang diangkat dalam penelitian ini adalah sebagai berikut:

1. Bagaimana membangun model klasifikasi menggunakan pendekatan ANN di dalam klasifikasi teks menggunakan NB *classifier*?
2. Bagaimana hasil implementasi ANN terhadap klasifikasi teks menggunakan algoritma NB?

3. Bagaimana optimasi penggunaan sumber daya komputasi algoritma NB dengan pendekatan ANN menggunakan *framework resource-aware*?

1.3. Tujuan Penelitian

Tujuan penelitian ini adalah untuk mengatasi kekurangan pada algoritma NB dalam membangun sebuah model tanpa memperhitungkan hubungan antar fitur didalam model klasifikasinya, dengan mengimplementasikan pendekatan ANN dalam tahap pembelajaran dari model klasifikasi dari NB.

1.4. Manfaat Penelitian

Penelitian ini memiliki manfaat dalam meningkatkan akurasi algoritma NB melalui penerapan pendekatan *artificial neural network* (ANN) dalam pembangunan model klasifikasi. Dengan model dari ANN, diharapkan dapat tercapai peningkatan kinerja model, yang memiliki dampak positif terutama dalam konteks pengambilan keputusan berdasarkan output klasifikasi. Selain itu, penelitian ini juga melakukan analisis terhadap penggunaan sumber daya komputer.

1.5. Kontribusi Penelitian

Pada penelitian ini mengusulkan pendekatan ANN dalam pembangunan model klasifikasi algoritma NB, dengan harapan fitur-fitur hasil pembelajaran dengan pendekatan ANN dapat meningkatkan performa dari klasifikasi menggunakan NB. Dan memberikan kontribusi dengan analisis dari penggunaan sumber daya komputer yang digunakan.

[Halaman Sengaja Dikosongkan]

BAB 2
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

2.1.1. Hate Speech

Penelitian [7] ditemukan bahwa ISIS secara sistematis menggunakan justifikasi religius untuk menjustifikasi kekerasan terhadap non-muslim tertulis didalam artikel "The Kafir's Blood is Halal for You, So Shed It" yang diterbikan di majalah rumiyah. Pada penelitian ini juga menyebutkan bahwa ujaran kebencian yang disebarkan melalui platform digital dapat menyebar dengan cepat dan mempengaruhi audiens yang luas sehingga menimbulkan prasangka dan tindakan kekerasan yang nyata.

Selanjutnya pada studi [8] menekankan berbagai tantangan yang membuat deteksi ujaran kebencian menjadi sulit. Penelitian menemukan bahwa banyak pengguna media sosial di Indonesia menggunakan bahasa informal dan campuran, termasuk dialek lokal dan bahasa asing, yang memperumit deteksi. Selain itu, pengguna sering kali mengubah bentuk kata kasar untuk menghindari deteksi otomatis, misalnya dengan menghapus vokal atau mengganti karakter tertentu. Pada penelitian ini juga menunjukkan pentingnya normalisasi data dan penyeimbangan dataset untuk meningkatkan akurasi klasifikasi. Hasil penelitian menunjukkan bahwa penggunaan teknik machine learning seperti Naive Bayes dengan fitur n-gram kata memberikan hasil yang cukup baik, namun klasifikasi menjadi lebih rumit ketika harus membedakan antara bahasa kasar yang ofensif dan yang tidak ofensif.

Penelitian [9] menjabarkan bahwa *online toxicity* di Indonesia seringkali menggunakan hatespeech dalam konteks politik, ideologis dan agama, serta untuk *cyberbullying*. Penelitian juga mengidentifikasikan *hate speech* sebagai bentuk komunikasi yang merendahkan, menghina, atau melecehkan kelompok atau individu berdasarkan karakteristik seperti etnis, agama, atau gender.

2.1.2. Klasifikasi Naïve Bayes

Pada penelitian [10] mengusulkan untuk melakukan tuning dan pembobotan *attribute* untuk meningkatkan akurasi dari algoritma NB dengan nama *Fine Tuning Attribute Weigted Naïve Bayes* (FTAWNB). Peneltian ini dilakukan dengan membandingkan performa FTAWNB dengan algoritma *state-of-the-art* dari algoritma NB. Penelitian dilakukan dengan membandingkan hasil dari beberapa dataset yang ada, dan didapatkan hasil bahwa FTAWNB superior dibandingkan dengan *state-of-the-art* dari algoritma NB. Dengan hasil FTAWNB lebih baik di 18 dataset jika dibandingkan *Boosted Naïve Bayes* (BNB), dan hanya lebih buruk di dua dataset. FTAWNB juga lebih baik di 23 dataset jika dibandingkan dengan Naïve Bayes. Tetapi pada penelitian ini tidak menggunakan dataset dengan bentuk teks dalam melakukan pengujian terhadap model FTAWNB.

Penelitian [11] mengusulkan bagaimana mengatasi keterbatasan algorita NB yang memiliki karakteristik independensi antar fitur menjadi dependen antar fitur berdasarkan label yang diberikan dengan nama algoritma *Multilabel Naïve Bayes Label Dependence* (MLNB-LD). Penelitian dilakukan dengan mengujikan MLNB-LD dengan beberapa algoritma *multilabel* yang lain seperti *Multilabel Naïve Bayes* (MLNB), Multilabel Decission Tree (MLDT), dan Multilabel kernel extreme learning machine (ML-kELM), dan didapatkan hasil bahwa dari 14 *dataset* yang diujikan MLNB-LD accuracy MLNB-LD hanya 2 kali lebih rendah dengan rata-rata peringkat akurasi di 1,214 dari 14 pengujian.

Penelitian [12] mengusulkan untuk menggunakan *deep feature extraction* dengan ANN terhadap klasifikasi dari NB, dengan pengurangan dimensi menggunakan *Black-Jacobi* dalam prediksi *churn*. Penelitian dilakukan dengan membandingkan dengan algoritma klasifikasi yang lain dan didapatkan hasil dengan melakukan *deep feature extraction* terhadap NB hasil akurasi NB meningkat menjadi 93%. Dan model yang ditawarkan juga memiliki performa akurasi yang lebih baik dibandingkan dengan *Deep Neural Network* (DNN), *Random Forest* (RF), *K-Nearest Neighbor* (KNN), dan *Decission Tree* (DT).

Pada penelitian [13] mengusulkan dengan menambahkan *Markov Random Field* (MRF) dalam melakukan klasifikasi email spam dan tidak. Pada penelitian ini klasifikasi NB mengukur efektivitas dan konfigurasi nilai dalam dataset dan

kemudian melakukan proses klasifikasi probabilistik yang sederhana. Teknik klasifikasi NB menggunakan prinsip Bayes untuk menentukan apakah sebuah email adalah spam atau tidak. Selanjutnya, Markov Random Field yang memodelkan kebiasaan prediktif Spam dianalisis. Untuk menciptakan vektor fungsi dari dokumen yang masuk (email), dokumen harus diuraikan menjadi fitur-fitur. Bobot dapat diberikan kepada vektor-vektor fitur ini untuk memungkinkan algoritma pembelajaran memperhitungkan dependensi antar kata, untuk mengatasi independensi fitur yang menjadi kekurangan dari algoritma NB. Hasil dari penelitian didapatkan bahwa penggunaan komputasi tetap rendah walaupun dengan melakukan penggabungan dari 2 algoritma yang ada dan penggabungan algoritma ini juga memberikan kemampuan untuk algoritma NB dalam mengklasifikan *email spam* lebih banyak.

Penelitian [14] mengusulkan untuk menambahkan *attribute value frequency-based instance weighting filter* untuk mempelajari bobot setiap *instance*. Pendekatan ini didorong oleh beberapa pengamatan yaitu; Frekuensi setiap nilai atribut mengandung informasi penting yang dapat digunakan untuk menentukan bobot instance pelatihan. dan Bobot setiap instance pelatihan memiliki korelasi positif dengan vektor frekuensi nilai atribut dan vektor jumlah nilai atribut dari seluruh dataset pelatihan. Hasil penelitian didapatkan bahwa model yang diusulkan memiliki hasil yang lebih baik dengan *two-tailed t-tests*.

2.1.3. Klasifikasi Neural Network

Penelitian [15] mengusulkan untuk menambahkan *label based attention* terhadap algoritma *Hierarchical multi-label text classification* (HMTC), Tujuan dari modul ini adalah untuk mempelajari hubungan laten antara setiap label dan komponennya. Label pada tingkat *h* direpresentasikan dengan sekumpulan vektor (embedding label). Hasil penelitian didapatkan bahwa implementasi *label based attention* terhadap HMTC dapat meningkatkan akurasi dari algoritma HMTC, dengan akurasi terbaik yang didapatkan 82% dibandingkan 78% jika algoritma HMTC tanpa mengimplementasikan *label based attention*.

Penelitian [16] mengusulkan sebuah metode Frog-GNN sebuah metode *Natural Language Processing* (NLP), dengan tujuan untuk membangun sebuah model yang memiliki sedikit dataset dan memiliki performa yang baik jika

dibandingkan dengan algoritma NLP yang lain dengan dataset yang kecil.Frog-GNN menggunakan basis BERT dalam melakukan pembobotan terhadap fitur dari text. Dari penelitian didapatkan hasil bahwa Frog-GNN menghasilkan performa yang lebih baik dari algoritma *state-of-the-art* dari algoritma *few-shot* yang lain, dengan akurasi sebesar 88,88% dengan *5 way- 1 shot* atau 94,28% jika menggunakan *5 way – 5 shot*.

Pada penelitian [17] mengusulkan *graph-based pipeline framework* yang disebut TextFCG untuk klasifikasi teks induktif. Pada penelitian dibuat *single graph* untuk setiap teks masukan untuk mengintegrasikan hubungan kontekstual dengan tepat. Selain itu, metode TextFCG berfokus pada kata-kata dan hubungan dari setiap teks untuk mengintegrasikan berbagai informasi secara harmonis, yang dapat diterapkan pada pembelajaran induktif. Dari penelitian didapatkan rata-rata akurasi terbaik yang dimiliki oleh TextFCG 98,22 % pada dataset R8.

2.2. Dasar Teori

2.2.1 Data Mining

Text mining atau penambangan teks adalah ddisiplin yang melibatkan analisis komputasional terhadap dokumen teks untuk mengidentifikasi pola, hubungan, dan informasi yang berharga [18]. Dalam era di mana data teks terus berkembang secara eksponensial, *text mining* menjadi kunci untuk memahami dan memanfaatkan informasi yang terkandung dalam dokumen, artikel, sosial media, dan sumber teks lainnya.

Tujuan utama dari *text mining* adalah menemukan informasi baru dan berharga yang mungkin tidak terdeteksi oleh manusia dalam data teks yang besar dan kompleks. Dengan menganalisis pola-pola kata, hubungan, dan tren dalam teks, *text mining* membantu dalam pengambilan keputusan, pemahaman sentimen publik, dan identifikasi informasi penting.

Proses *text mining* melibatkan beberapa tahap seperti pemrosesan teks, ekstraksi fitur, pengelompokan (*clustering*), klasifikasi, dan analisis sentimen. Pada tahap pemrosesan teks, dokumen-dokumen diubah menjadi representasi yang dapat dimengerti oleh mesin, seperti vektor kata-kata. Selanjutnya, melalui ekstraksi fitur, informasi yang relevan diidentifikasi dan diambil dari teks. Tahap pengelompokan dan klasifikasi memungkinkan pengelompokan dokumen berdasarkan kesamaan

atau pembedaan tertentu, sementara analisis sentimen dapat mengungkapkan sentimen atau pandangan yang terkandung dalam teks.

2.2.2 Data Stream Mining

Data stream mining adalah bidang penelitian yang berkaitan dengan analisis data yang berasal dari aliran data kontinu, seperti data sensor, log transaksi, atau arus media sosial. Seiring dengan pertumbuhan pesat data *real-time*, *data stream mining* menjadi semakin penting dalam mengekstrak pola, tren, dan informasi berharga dari aliran data yang terus bergerak.

Dibandingkan dengan data statis, data stream memiliki karakteristik unik seperti tingginya laju penghasilan data, sifat non-stasioner, dan keterbatasan waktu pemrosesan. Oleh karena itu, *data stream mining* melibatkan pendekatan yang berbeda dalam menghadapi tantangan ini. [19] membahas cara-cara untuk menangani data stream, termasuk masalah algoritma dan strategi pengelolaan memori yang efisien.

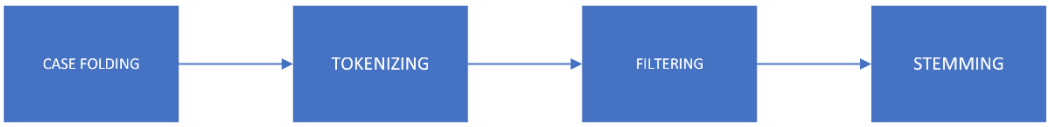
Penelitian oleh Aggarwal [20] Mengeksplorasi berbagai algoritma yang dirancang khusus untuk menangani data stream. Aggarwal menyoroti pentingnya algoritma yang adaptif dan efisien dalam menghadapi ketidakpastian dan perubahan dalam data stream. Algoritma-algoritma ini mencakup teknik sampling, summarization, dan pengelolaan konsep.

2.2.3 Text Preprocessing

Text preprocessing adalah tahap penting dalam pengolahan bahasa alami (*Natural Language Processing* atau NLP). NLP adalah cabang kecerdasan buatan yang berfokus pada interaksi antara komputer dan bahasa manusia. Dalam rangka meningkatkan kinerja algoritma NLP, langkah-langkah *preprocessing* diterapkan pada data teks untuk membersihkan, memformat, dan menyederhanakan informasi agar dapat diolah lebih efektif oleh model-model NLP.

Tujuan utama dari *text preprocessing* dalam konteks NLP adalah menciptakan data teks yang bersih, terstruktur, dan representatif. Langkah-langkah preprocessing membantu menghilangkan *noise*, menyatukan format, dan memfasilitasi pemahaman konten oleh model-model NLP. Proses ini membantu

menghadapi variasi dalam gaya penulisan, ejaan, dan struktur kalimat yang dapat mempengaruhi kinerja model.



Gambar 2. 1 Proses Text Preprocessing

Pada Gambar 2. 1 ditunjukkan bagaimana alur dari text preprocessing yang umum dilakukan, dimulai dengan melakukan *case-folding, tokenizing, filtering* dan *stemming.*

1. *Case Folding*

Tahap *case folding* adalah salah satu langkah dalam proses normalisasi teks yang umumnya dilakukan dalam text preprocessing. *Case folding* melibatkan konversi semua huruf dalam sebuah teks ke dalam bentuk yang seragam, seperti huruf kecil (lowercase) atau huruf besar (uppercase). Tujuannya adalah untuk mengatasi variasi gaya penulisan dan memastikan bahwa kata-kata dengan huruf besar atau kecil tetap diperlakukan secara seragam selama analisis teks.

Penerapan *case folding* tergantung pada tugas atau analisis spesifik yang sedang dilakukan. Dalam kebanyakan kasus, *lowercasing* lebih umum digunakan karena membantu meminimalkan ambiguitas dan memastikan konsistensi data. Namun, dalam beberapa konteks, seperti analisis nama entitas atau akronim, *uppercase folding* mungkin lebih sesuai. Pilihan antara kedua metode ini tergantung pada karakteristik khusus dari dataset dan tujuan analisis teks yang dilakukan.

2. *Tokenizing*

Tokenizing adalah tahap dalam *text preprocessing* yang melibatkan pemecahan teks menjadi unit-unit kecil yang disebut token. Token bisa berupa kata-kata, frasa, atau subword, tergantung pada tingkat granularitas yang diinginkan. Proses *tokenizing* bertujuan untuk membuat representasi yang lebih terstruktur dari teks sehingga dapat diolah lebih lanjut oleh model-model NLP.

Tahap *tokenizing* ini membantu model NLP untuk memahami struktur dan makna dalam teks dengan lebih baik. Algoritma *tokenizing* dapat dilakukan dengan

berbagai cara, termasuk menggunakan aturan gramatikal, pemisahan berdasarkan spasi, atau menggunakan model pembelajaran mesin seperti *tokenizers* yang telah dilatih sebelumnya. Pilihan metode *tokenizing* tergantung pada bahasa teks, jenis teks, dan tugas NLP yang sedang dilakukan.

3. Filtering

Tahap *filtering* dalam *text preprocessing* adalah proses menghapus atau memfilter elemen-elemen tertentu dari teks yang dianggap tidak relevan atau tidak diinginkan untuk analisis lebih lanjut. *Filtering* membantu membersihkan dan menyederhanakan teks agar fokus pada informasi yang paling penting.

Tahap filtering ini dapat diadaptasi sesuai dengan kebutuhan dan tujuan analisis teks. Pemilihan elemen yang dihapus atau dipertahankan sangat tergantung pada jenis tugas NLP yang sedang dilakukan dan karakteristik spesifik dari dataset teks.

4. Stemming

Tahap *stemming* dalam *text preprocessing* adalah proses mengubah kata-kata ke dalam bentuk dasar atau kata dasar dengan menghapus afiks (akhiran atau awalan). Tujuan dari stemming adalah untuk mengurangi variasi kata sehingga kata-kata yang memiliki akar atau makna yang sama diwakili dengan bentuk yang seragam.

Proses *stemming* sering digunakan dalam analisis teks, terutama ketika fokus pada makna kata dan mengabaikan variasi bentuk kata yang lain. Meskipun stemming dapat membantu memperbaiki kepadatan kata dan mengurangi dimensi dalam analisis, perlu diingat bahwa hasil stemming tidak selalu kata yang valid atau umum digunakan dalam bahasa sehari-hari. Beberapa kata hasil stemming mungkin tidak bermakna atau tidak terdengar alami, dan hal ini perlu diperhatikan dalam konteks penggunaannya.

2.2.4 Teorema *Bayes*

Teorema *Bayes*, yang dinamai sesuai dengan matematikawan Inggris Thomas Bayes, merupakan konsep fundamental dalam statistika yang memberikan cara untuk memperbarui keyakinan atau probabilitas tentang suatu peristiwa setelah mempertimbangkan bukti baru. Dengan kata lain, teorema *Bayes* memungkinkan untuk menggabungkan informasi sebelumnya dengan data baru untuk memperoleh

pemahaman yang lebih baik tentang suatu fenomena. Sebagai alat analisis yang kuat, teorema *Bayes* memiliki aplikasi luas dalam berbagai bidang, termasuk ilmu komputer, kecerdasan buatan, dan ilmu data.

Dalam teorema Bayes, perhitungan *posterior probability*, yang merupakan probabilitas suatu peristiwa setelah mempertimbangkan bukti baru, didasarkan pada *prior probability*, yaitu keyakinan awal kita tentang peristiwa tersebut, serta *likelihood*, yang menyatakan sejauh mana bukti mendukung peristiwa tersebut seperti yang dituliskan dalam Persamaan (2.1).

$$posterior\ probability = \frac{conditional\ probability * prior\ probability}{evidence} \quad (2.\ 1)$$

Dalam prakteknya, teorema *Bayes* sering digunakan dalam pengambilan keputusan, pengenalan pola, dan peramalan, di mana dapat menggabungkan pengetahuan sebelumnya dengan informasi yang baru diperoleh untuk meningkatkan ketepatan prediksi. Meskipun teorema ini memiliki dasar matematika yang kuat, interpretasinya juga memerlukan pemahaman kontekstual yang baik. Oleh karena itu, pemahaman konsep dasar teorema Bayes serta kemampuan untuk menerapkannya dengan bijak sangat penting dalam pemrosesan informasi yang berhubungan dengan ketidakpastian.

2.2.5 Naïve Bayes Classifier

Klasifikasi Naive Bayes adalah salah satu metode klasifikasi dalam bidang pembelajaran mesin yang didasarkan pada Teorema Bayes dengan asumsi bahwa fitur-fitur yang digunakan untuk melakukan klasifikasi adalah independen secara kondisional. Metode ini sering digunakan dalam analisis teks, klasifikasi dokumen, dan pengenalan pola. Dalam konteks klasifikasi, Naive Bayes memanfaatkan probabilitas untuk memprediksi kelas suatu instansi berdasarkan nilai fitur-fiturnya. Secara matematis didefinisikan seperti dalam Persamaan (2.2).

$$P(\omega j) = \frac{P(x_i|\omega_j) \cdot P(wj)}{P(x_j)} \quad (2.\ 2)$$

$$(2.\ 3)$$

Dengan:

x_i = Fitur dari sampel i

ωj = Notasi untuk kelas j

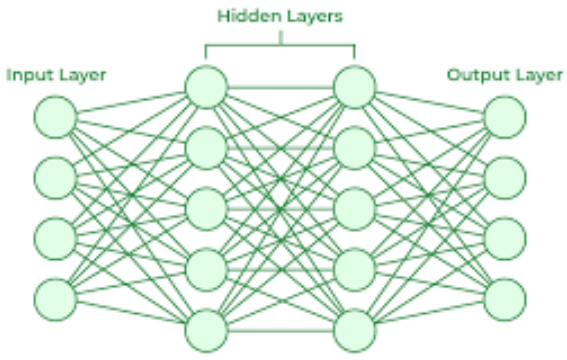
$P(x_i|\omega_j)$ = Probabilitas dari sampel xi terhadap kelas wj

Proses klasifikasi dimulai dengan menghitung probabilitas prior untuk setiap kelas, kemudian menghitung probabilitas likelihood dari fitur-fitur yang diamati untuk masing-masing kelas. Dengan menggunakan Teorema Bayes, probabilitas posterior dapat dihitung, dan instansi dapat diklasifikasikan ke dalam kelas dengan probabilitas tertinggi, secara matematis dapat dituliskan seperti Persamaan (2.3)

$$predicted\ class\ label \leftarrow argmax\ P(x_i|\omega_j) \quad (2.3)$$

2.2.6 Neural Network

Artificial Neural Network atau yang sering disebut Neural Network (NN) adalah suatu sistem pengolahan informasi yang meniru karakteristik kinerja jaringan saraf biologis. Neural Network bukan bagian dari komputasi keras tradisional yang mengutamakan presisi dan kepastian, melainkan merupakan bagian dari komputasi lunak yang lebih toleran terhadap ketidakpastian, ketidakpresisian, dan kebenaran parsial, dengan fokus pada pendekatan berpikir. Tujuan utama dari komputasi lunak adalah memanfaatkan toleransi tersebut untuk mencapai daya tarik, ketahanan, kecerdasan mesin tingkat tinggi, dan biaya rendah dalam aplikasi.



Gambar 2. 2 Struktur Jaringan Neural Network (https://www.geeksforgeeks.org)

Seperti pada Gambar 2. 2 Neural network terdiri dari unit pemrosesan sederhana yang disebut *neuron*, yang diorganisir dalam lapisan-lapisan untuk membentuk arsitektur jaringan. *Neuron-neuron* ini saling terhubung dan mampu

mengenali pola dan fitur dari data masukan. Konsep dasar *neural network* adalah untuk melakukan pembelajaran dari data dengan menyesuaikan bobot dan bias koneksi antar *neuron*. Jaringan saraf terbagi menjadi beberapa jenis, dengan yang paling umum adalah *feedforward neural network* dan *recurrent neural network*. *Feedforward neural network memiliki* arah aliran informasi yang satu arah, dari input ke output, tanpa adanya siklus. Sementara itu, *recurrent neural network* memiliki hubungan siklik di antara neuron-neuron, memungkinkan mereka untuk memproses informasi yang melibatkan urutan atau konteks waktu. Keduanya digunakan dalam berbagai aplikasi, seperti pengenalan gambar, pengolahan bahasa alami, dan prediksi rangkaian waktu.

Proses pelatihan neural network melibatkan *backpropagation*, yang mana model menyesuaikan bobot dan biasnya berdasarkan perbedaan antara prediksi yang dihasilkan dan label yang sebenarnya. Proses ini memerlukan fungsi kerugian yang mengukur sejauh mana model mendekati hasil yang diinginkan. Keberhasilan *neural network* juga sangat bergantung pada jumlah data pelatihan yang memadai, serta parameter seperti tingkat pembelajaran dan arsitektur jaringan.

2.2.7 Confusion Matrix

Confusion matrix merupakan teknik yang umum digunakan dalam *machine learning* untuk menilai kinerja suatu model. Matriks ini terdiri dari empat elemen utama: *True Positive* (TP), *False Positive* (FP), *True Negative* (TN), dan *False Negative* (FN). TP adalah jumlah prediksi positif yang benar, sedangkan FP adalah jumlah prediksi positif yang salah. TN mengacu pada jumlah prediksi negatif yang benar, dan FN adalah jumlah prediksi negatif yang salah. Dari analisis keempat elemen ini, mampu mendapatkan gambaran yang komprehensif tentang bagaimana model melakukan klasifikasi.

Penggunaan *confusion matrix* memungkinkan untuk menghitung beberapa matrik performa seperti akurasi, *precision, recall, dan F1-score*. Akurasi adalah rasio dari semua prediksi yang benar terhadap total prediksi yang dibuat oleh model secara metematis dapat dituliskan seperti persamaan (2. 4). *Precision* mengukur seberapa banyak prediksi positif yang benar dibandingkan dengan semua prediksi positif yang dibuat atau secara matematis dapat dihitung seperti persamaan (2. 5). *Recall*, juga dikenal sebagai sensitivitas, adalah ukuran dari kemampuan model

untuk mendeteksi semua sampel positif yang sebenarnya secara matematis dapat dihitung menggunakan persamaan (2. 6). *F1-score* adalah rata-rata harmonis dari presisi dan recall, memberikan keseimbangan antara keduanya, atau dapat dihitung menggunakan persamaan (2. 7)

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (2.\ 4)$$

$$precission = \frac{TP}{TP+FP} \quad (2.\ 5)$$

$$recall = \frac{TP}{TP+FN} \quad (2.\ 6)$$

$$F1\ Score = \frac{2*Precision*Recall}{Precision+Recall} \quad (2.\ 7)$$

2.3 Research Gap

Penelitian-penelitian yang telah dijelaskan memberikan gambaran yang cukup komprehensif tentang berbagai pendekatan untuk meningkatkan kinerja algoritma klasifikasi *Naive Bayes* (NB). Terutama, beberapa penelitian terakhir menyoroti peran penting dari pendekatan *Neural Network* (NN) dalam mengatasi keterbatasan NB dan meningkatkan akurasi prediksi.

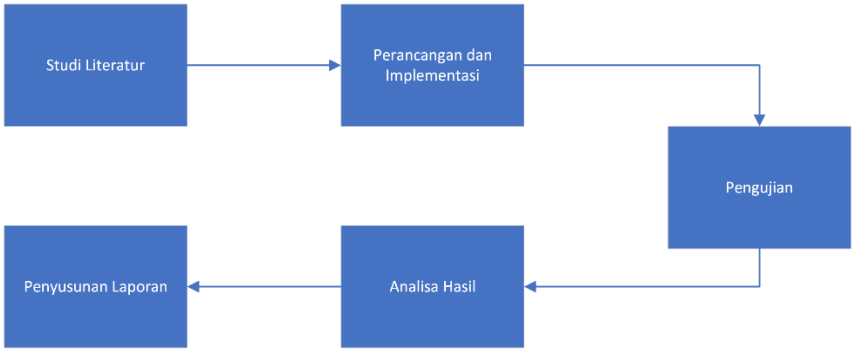
Salah satunya adalah penelitian [12], yang mengusulkan penerapan *deep feature extraction* dengan *Artificial Neural Network* (ANN) terhadap klasifikasi NB. Dengan menggunakan ANN, penelitian ini berhasil meningkatkan akurasi NB hingga mencapai 93%. Hasil ini mengindikasikan bahwa ekstraksi fitur mendalam dengan menggunakan *neural network* dapat memberikan kontribusi signifikan terhadap peningkatan performa NB dalam tugas klasifikasi, bahkan melebihi beberapa model klasifikasi lainnya seperti *Deep Neural Network* (DNN), *Random Forest* (RF), *K-Nearest Neighbor* (KNN), dan *Decision Tree* (DT).

Penelitian [15] juga menunjukkan potensi penerapan *Neural Network* dalam meningkatkan akurasi algoritma klasifikasi. Dalam konteks *Hierarchical Multi-Label Text Classification* (HMTC), penambahan *label-based attention* terhadap HMTC berhasil meningkatkan akurasi hingga mencapai 82%. Ini menunjukkan bahwa penggunaan mekanisme perhatian berbasis label dapat memberikan informasi tambahan yang berguna untuk meningkatkan kemampuan klasifikasi, terutama pada tugas-tugas yang melibatkan struktur hierarki dan multi-label.

Umumnya sering ditemukan ekstraksi fitur lain seperti menggunakan *Principal Component Analysis* (PCA) pada klasifikasi *naïve bayes* berbasis teks, tetapi PCA tidak mampu untuk menangkap representasi dari fitur *non linier* dari data teks, karena sifat PCA cenderung fokus pada variasi terbesar dari data, di sisi lain dengan ekstraksi fitur dari NN, fitur-fitur diekstraksi secara otomatis melalui lapisan-lapisan neuron. Meskipun demikian, terdapat kebutuhan untuk lebih mendalam dalam mengeksplorasi integrasi antara pendekatan *Neural Network* dan *Naive Bayes*, khususnya dalam dataset berbasis data teks. Penelitian ini bertujuan untuk mengeksplorasi sejauh mana penggabungan elemen-elemen *Neural Network* dapat memberikan kontribusi pada peningkatan akurasi dan kinerja NB pada dataset berbasis teks yang kompleks. Dengan lebih memahami interaksi antara NB dan NN, mungkin dapat ditemukan inovasi yang lebih lanjut untuk memperbaiki kelemahan dan memanfaatkan kekuatan keduanya secara bersamaan.

BAB 3
METODE PENELITIAN

Bab ini akan menjelaskan mengenai metode penelitian yang diterapkan dalam studi ini. Rincian mengenai alur metode penelitian dapat ditemukan dalam Gambar 3. 1. Langkah-langkahnya meliputi studi literatur sebagai awal, dilanjutkan dengan perancangan dan implementasi, pengujian, analisis hasil, dan akhirnya penyusunan laporan.



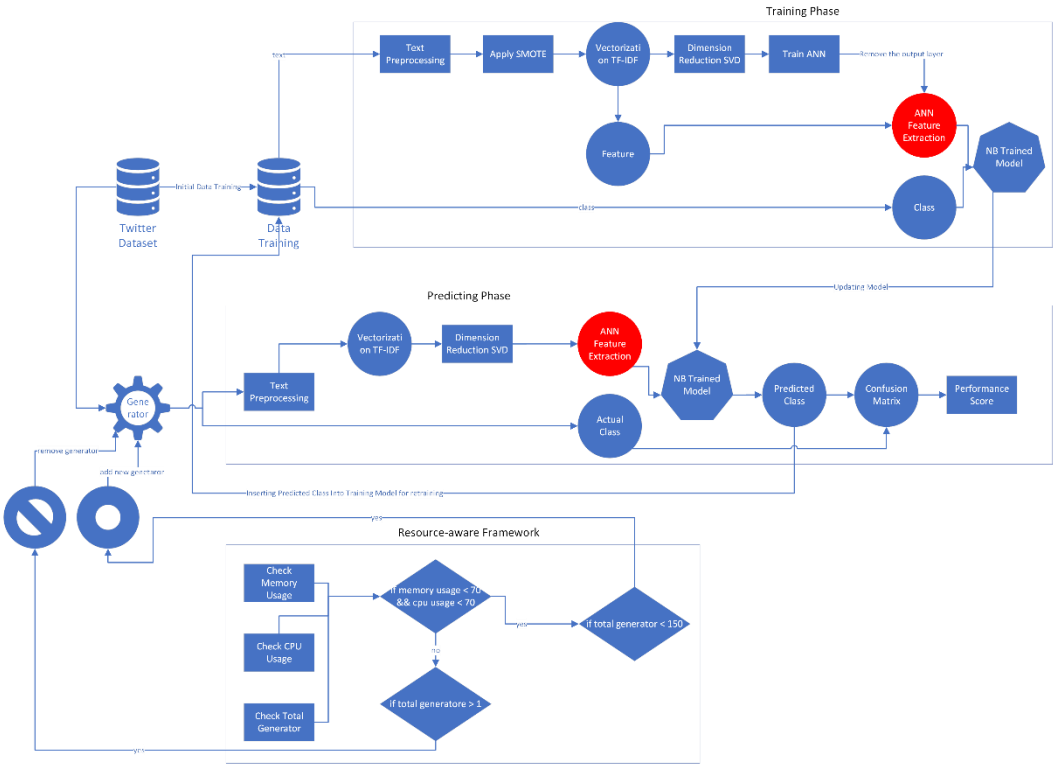
Gambar 3. 1 Alur Penelitian

3.1. Studi Literatur

Penelitian ini dimulai dengan melakukan kajian literatur yang terfokus pada topik penelitian yang dipilih. Referensi yang diakses dan digunakan dalam penelitian berasal dari jurnal-jurnal yang memiliki keterkaitan dengan aspek-aspek tertentu, seperti naive bayes, analisis sentimen, neural network, dan klasifikasi teks. Proses pengumpulan literatur ini melibatkan eksplorasi berbagai sumber informasi ilmiah yang relevan untuk memastikan pemahaman yang mendalam terhadap kerangka konseptual dan landasan teori yang melandasi penelitian ini. Dengan demikian, literatur yang dipilih menjadi landasan untuk merinci dan mengembangkan metodologi penelitian serta menyelidiki isu-isu kunci yang terkait dengan topik penelitian.

3.2. Perancangan dan Implementasi

Gambar 3.2 memvisualisasikan alur perancangan penelitian yang akan diimplementasikan, dengan fokus pada simulasi menggunakan metode data stream mining. Tahapan awal melibatkan pembagian dataset Twitter yang telah terkumpul melalui proses *splitting*. 70% data yang telah di-*split* akan menjadi dataset latih, kemudian data latih akan masuk ke tahap *preprocessing*. Hasil *preprocessing* akan dilakukan proses penyeimbangan data antara kelas *hate speech* dan *non hate speech* menggunakan algoritma SMOTE. Selanjutnya untuk memastikan bahwa ukuran data tidak melebihi kapasitas memori *device* yang digunakan maka dilakukan reduksi dimensi data menggunakan SVD. Kemudian akan dilakukan proses *training* pada model ANN, setelah proses *training* selesai layer output pada ANN akan dihilangkan yang kemudian akan menjadi *feature extractor* untuk proses *training* dan *testing* pada model NB.



Gambar 3. 2 Rancangan Model Penelitian

Pada tahap simulasi, generator akan menggunakan 30% sisa data dari dataset sebagai data uji, dengan melakukan pemilihan data secara acak sebagian data dalam 1 siklus simulasi. Ekstrsi fitur akan dilakukan dengan menggunakan model ANN yang sudah dilatih, Kemudian setelah mendapatkan hasil prediksi, penelitian ini akan menguji performa model menggunakan metode *confusion matrix*. Fitur yang telah diekstraksi dengan menggunakan neural network dan hasil prediksi akan menjadi data pelatihan untuk tahap selanjutnya. Dalam fase ini, model akan menjalani proses training ulang secara konkuren selama masa *streaming* masih berlangsung. Selain mengevaluasi akurasi model, penelitian ini juga akan melakukan review terhadap penggunaan sumber daya sistem, dengan fokus pada penggunaan RAM dan CPU selama proses pengujian. Dengan melakukan pendekatan *resource-aware framework* total generator *datastream* akan dikontrol melalui penggunaan CPU dan RAM pada perangkat.

Penelitian ini tidak hanya bertujuan untuk mengukur sejauh mana akurasi model, tetapi juga untuk melakukan evaluasi performa terhadap sumber daya yang digunakan selama proses pengujian. Dengan memantau penggunaan RAM dan CPU, penelitian ini berusaha untuk memahami dampak pengoperasian sistem terhadap sumber daya komputasi. Pendekatan ini membantu menyediakan pandangan yang komprehensif terhadap efisiensi dan efektivitas model yang diusulkan dalam konteks aplikasinya pada *data stream mining*, dengan mempertimbangkan keterbatasan sumber daya yang mungkin ada.

3.2.1 Data Explanatory

Penelitian ini memanfaatkan beberapa dataset twitter berbahasa Indonesia. Setelah dataset terkumpul, selanjutnya akan dilakukan klasifikasi terhadap sentimen berupa *hatespeech*, yaitu membedakan antara *hatespeech* atau bukan. Sebagai contoh, struktur dataset dengan klasifikasi bukan hatespeech dapat dilihat pada Tabel 3. 1 sedangkan dataset dengan klasifikasi hatespeech dapat diliihat pada Tabel 3. 2

Tabel 3. *1* Contoh Dataset Non Hatespeech

Teks	**Klasifikasi**
Itu yang ngomong jangan pilih Ahok Djarot pernah ngerasain banjir ga sih? :(	NON-HS
jadi yg nyebar hoax ketahuan dong ada akun twitter nya lho	NON-HS
Ke Bidara beli mangga. Pulang nye liwat Slipi. Jangan lupa ingetin tetangga. Untuk milih Agus sylvi. #7HariLagiMilihAHY #AgusBersamaUlama	NON-HS
#DebatFinalPilkadaJKT aduh... Kecewa pertanyaan paslon 1....	NON-HS
Ahok gagah kaya gatot kaca! #DebatFinalPilkadaJKT	NON-HS
Paslon 3 Anies Sandiaga Uno 15 Feb 2017 Utk Jakarta Indonesia Raya OK OCE	NON-HS
Suporternya mulai kampungan paslon masih ngomong jg ..... #DebatFinalPilkadaJKT	NON-HS
kalau kamu diputuskan pacar tanpa sebab mungkin program OKE OCE bisa jadi solusi nya #DebatFinalPilkadaJKT	NON-HS
Melibatkan yang paham pada bidangnya untuk duduk bersama memecahkan solusi. Cara kepemimpinan yg cerdas ! #DebatFinalPilkadaJKT	NON-HS
Bangga masih memiliki putra bangsa seperti pak Basuki dan pak Jarot.	NON-HS

Tabel 3. *2* Contoh Dataset Hatespeech

Teks	**Klasifikasi**
Ini tuh kesalahan Jokowi dan Ahok si babi ateis. https:\/\/t.co\/v2GRcUeStI	HS
ketutunan cina dari luar kota dan pulau mulai kejakarta mau coblos ahok dengan KTP aspal dari kamboja	HS
Ahok : program paslon no. 1 ini programnya ngambang | yaiyalah	HS

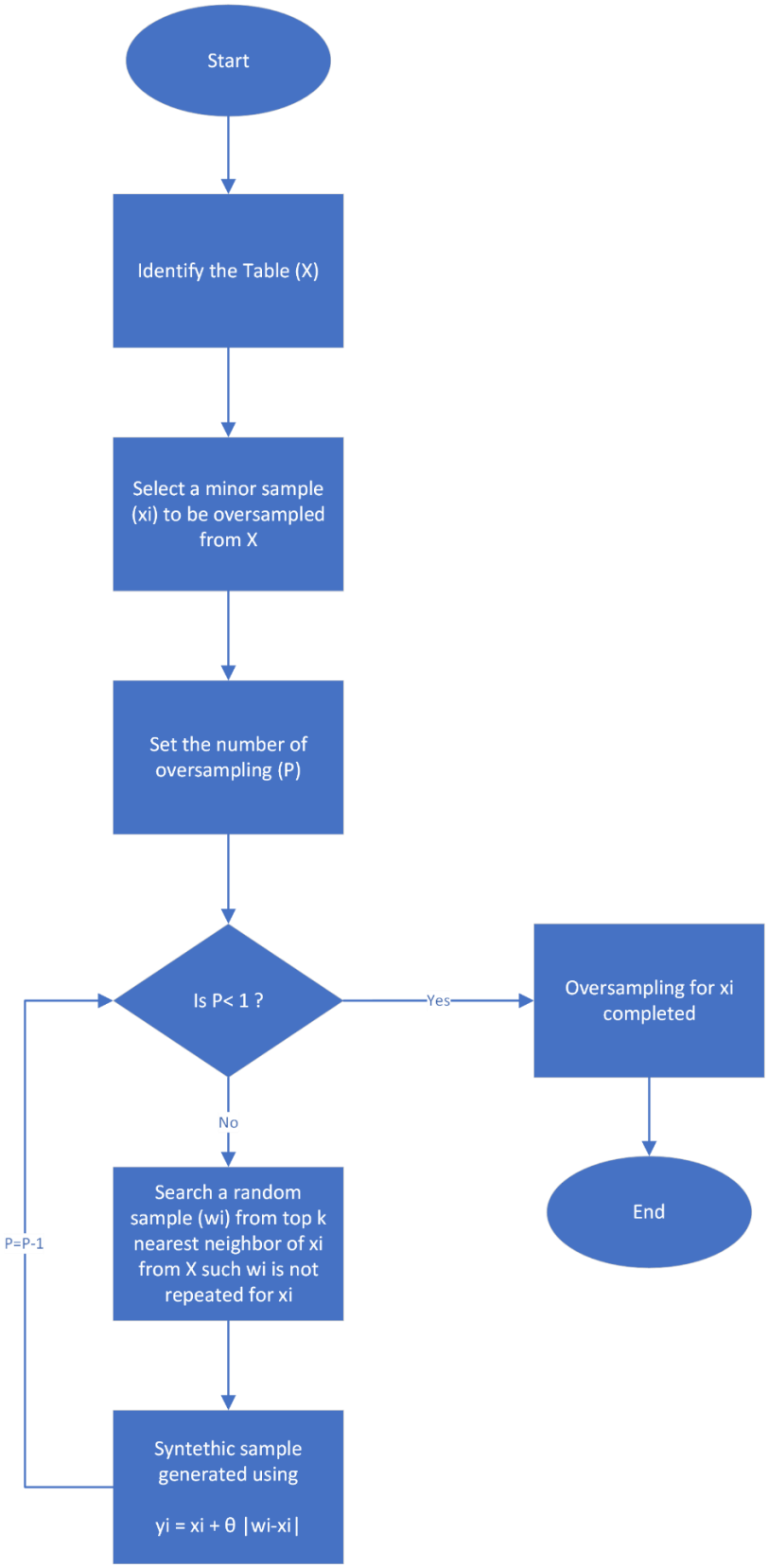
pakkk gak ada program diaa cma ada celaan #DebatFinalPilkadaJKT	
Huuu sylvi tak tahu apa-apa asal ngoceh #NobarAhokDjarot #DebatFinalPilkadaJKT	HS
Sylvi keliatan bloonnya #DebatFinalPilkadaJKT	HS
Teman Ahok emank TAI. apalagi pemimpinnya kek BABI! @temanahok	HS
ahmad dhani itu borok, sampah, babi, anjing. Saya adalah orang pertama yg sangat muak liat muka dhani dan saya berbangga.	HS
SETUJU SAYA,KLAU AHOK DAPAT GELAR SANTRI KEHORMATAN.... TAPI SANTRI KEHORMATAN BABI.... https://t.co/uD16d7Gwhd	HS
#MataNajwaDebatJakarta Anies anda SADIS. Topengmu terbuka lebar malam ini. Biar warga DKI yg menilai...apa yg anda katakan adalah PENGECUT	HS HS
Anies congornya aja gede. Realisasinya gak ada. Janji terus dari awal bisanya paling korupsi. Balik jadi dosen aja bangsat.	HS

Pada contoh yang diberikan, Tabel 3.1 dan Tabel 3.2 menampilkan masing-masing 10 data dalam kelas labelnya. Namun, dalam kenyataannya, data cenderung tidak seimbang antar kelas klasifikasi atau bersifat *imbalanced*. Untuk mengatasi permasalahan ketidakseimbangan data ini, penelitian ini akan menerapkan Teknik *sampling* agar data menjadi seimbang dengan menggunakan algoritma SMOTE.

Synthetic Minority Over-sampling Technique (SMOTE) adalah suatu metode *oversampling* yang menghasilkan sampel sintetis untuk kelas minoritas, sehingga menciptakan keseimbangan antara kelas mayoritas dan minoritas. Dengan menerapkan SMOTE, diharapkan dapat meningkatkan kinerja model klasifikasi pada data yang tidak seimbang.

SMOTE merupakan suatu teknik yang digunakan untuk menangani permasalahan ketidakseimbangan kelas dalam dataset. Terutama diterapkan pada

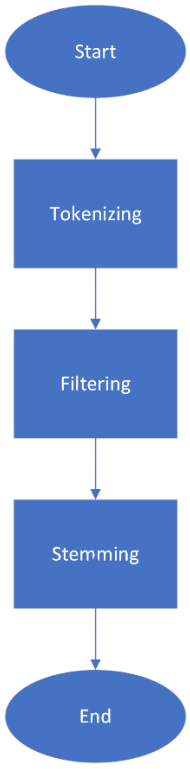
tugas-tugas klasifikasi di mana kelas minoritas memiliki jumlah sampel yang signifikan lebih sedikit daripada kelas mayoritas. Algoritma ini bekerja dengan cara menciptakan sampel sintetis baru untuk kelas minoritas, sehingga menciptakan distribusi yang lebih seimbang antara kelas positif dan negatif. Langkah-langkah SMOTE melibatkan pemilihan sampel dari kelas minoritas, penemuan tetangga terdekat dari sampel tersebut, dan pembuatan sampel sintetis dengan menambahkan sebagian perbedaan antara sampel dan tetangga ke sampel asli. Penggunaan SMOTE membantu model klasifikasi, termasuk neural network, untuk belajar dengan lebih baik dari kelas minoritas dan meningkatkan kemampuannya untuk melakukan prediksi yang akurat pada data yang tidak seimbang. Penerapan SMOTE dapat meningkatkan generalisasi model, mencegah overfitting pada kelas mayoritas, dan meningkatkan sensitivitas terhadap sinyal atau pola pada kelas minoritas.



Gambar 3. 3 Flowchart Algoritma SMOTE

3.2.2 Preprocessing Data

Proses persiapan data pada *text mining* melibatkan serangkaian langkah-langkah untuk membersihkan dan menyiapkan data teks sebelum dilakukan analisis. Langkah-langkah ini memiliki peran penting dalam meningkatkan kualitas dan relevansi informasi yang dapat diekstrak dari teks tersebut. Seperti yang tergambarkan dalam Gambar 3. 4, mengilustrasikan secara visual rangkaian tahap *preprocessing data*. *Flowchart* ini memberikan gambaran tentang alur langkah-langkah yang dilakukan, dimulai dari *input* data teks hingga hasil *output* yang telah melalui proses pembersihan dan penyesuaian. Tahapan ini berfungsi sebagai landasan untuk memastikan bahwa data teks siap digunakan dalam analisis lebih lanjut dengan meminimalkan *noise* dan meningkatkan relevansi informasi yang dapat diperoleh.



Gambar 3. 4 Flowchart Text Preprocessing

Tahapan *text preprocessing* dimulai dengan tahap *tokenizing* yaitu memisahkan kata atau frasa dari data teks yang diberikan. Tahapan ini juga

dilakukan dengan menghilangkan tanda baca dan melakukan *lowercasing* data teks yang diberikan. Contoh *tokenizing* dapat dilihat pada

Tabel 3. *3* Contoh Proses Tokenizing

Data awal	**Hasil *Tokenizing***
keturunan cina dari luar kota dan pulau mulai kejakarta mau coblos ahok dengan KTP aspal dari kamboja	['keturunan', 'cina', 'dari', 'luar', 'kota', 'dan', 'pulau', 'mulai', 'kejakarta', 'mau', 'coblos', 'ahok', 'dengan', 'ktp', 'aspal', 'dari', 'kamboja']
Itu yang ngomong jangan pilih Ahok Djarot pernah merasakan banjir ga sih? :(	['itu', 'yang', 'ngomong', 'jangan', 'pilih', 'ahok', 'djarot', 'pernah', 'merasakan', 'banjir', 'ga', 'sih']

Setelah proses tokenisasi selesai, langkah selanjutnya adalah melakukan tahap *filtering* dalam sistem, yaitu menghapus *stopwords* atau kata-kata yang dianggap kurang relevan. Dalam konteks penelitian ini, dataset *stopwords* yang telah dipersiapkan pada *library* python sastrawi akan digunakan sebagai acuan untuk mengidentifikasi dan menghilangkan kata-kata yang dianggap kurang penting. Tabel 3. 4 memperlihatkan hasil *filtering* dari dataset yang sudah didapatkan pada tahap *tokenizing*.

Tabel 3. *4* Contoh Proses Filtering

Data *Tokenizing*	**Hasil *Filtering***
['keturunan', 'cina', 'dari', 'luar', 'kota', 'dan', 'pulau', 'mulai', 'kejakarta', 'mau', 'coblos', 'ahok', 'dengan', 'ktp', 'aspal', 'dari', 'kamboja']	['keturunan', 'cina', 'kota', 'pulau', 'kejakarta', 'coblos', 'ahok', 'ktp', 'aspal', 'kamboja']

['itu', 'yang', 'ngomong', 'jangan', 'pilih', 'ahok', 'djarot', 'pernah', 'merasakan', 'banjir', 'ga', 'sih']	['ngomong', 'pilih', 'ahok', 'djarot', 'merasakan', 'banjir', 'ga', 'sih']

Setelah tahap *filtering* selesai, langkah selanjutnya adalah melakukan proses *stemming*. Proses *stemming* bertujuan untuk mengubah teks yang telah dibersihkan dari *stopwords* menjadi bentuk kata dasar menggunakan algoritma nazief-andriani. Dalam proses ini, dilakukan penghapusan sufiks, infiks, dan prefiks dari kata-kata awal. Contoh dari proses filtering dapat dilihat pada Tabel 3.

Tabel 3. *5* Contoh Proses Stemming

Data *Filtering*	**Hasil *Stemming***
['keturunan', 'cina', 'kota', 'pulau', 'kejakarta', 'coblos', 'ahok', 'ktp', 'aspal', 'kamboja']	['turun', 'cina', 'kota', 'pulau', 'kejakarta', 'coblos', 'ahok', 'ktp', 'aspal', 'kamboja']
['ngomong', 'pilih', 'ahok', 'djarot', 'merasakan', 'banjir', 'ga', 'sih']	['ngomong', 'pilih', 'ahok', 'djarot', 'rasa', 'banjir', 'ga', 'sih']

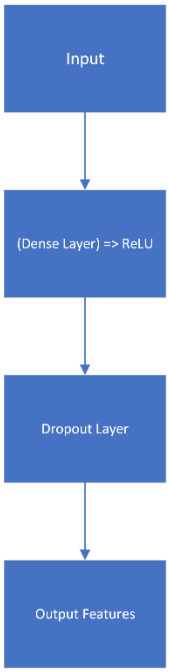
Secara singkat, tabel menyajikan transformasi data teks awal dari dataset menjadi dataset yang telah melalui proses *text preprocessing*. Proses tersebut mencakup langkah-langkah seperti *tokenization*, *stopword removal*, dan *stemming*. Data teks awal yang mungkin mengandung berbagai variasi kata dan struktur diubah menjadi bentuk yang lebih terstruktur, konsisten, dan siap untuk digunakan dalam analisis lebih lanjut. Tabel tersebut memberikan gambaran tentang bagaimana proses *preprocessing* secara signifikan dapat mempengaruhi bentuk dan sifat data teks, meningkatkan kualitas serta relevansi informasi yang dapat diekstrak.

Tabel 3. *6* Contoh Text Preprocessing

Data awal	**Hasil *Preprocessing***

keturunan cina dari luar kota dan pulau mulai kejakarta mau coblos ahok dengan KTP aspal dari kamboja	turun cina kota pulau kejakarta coblos ahok ktp aspal kamboja
Itu yang ngomong jangan pilih Ahok Djarot pernah merasakan banjir ga sih? :(	ngomong pilih ahok djarot rasa banjir ga sih

3.2.3 Modelling



Gambar 3. 5 Arsitektur ANN

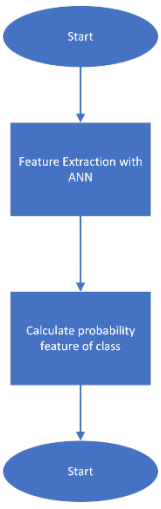
Dalam penelitian ini, ekstraksi fitur menggunakan *Artificial Neural Network* (ANN) menjadi tahapan kritis dalam membangun model klasifikasi *Naïve Bayes* untuk data teks. Fokus awal terletak pada pemilihan arsitektur ANN yang optimal, dengan pertimbangan jumlah *layer*, jumlah *neuron* dalam setiap *layer*, dan jenis fungsi aktivasi. Desain arsitektur yang tepat dianggap krusial karena memiliki dampak langsung pada kinerja keseluruhan model.

Langkah berikutnya adalah pemisahan dataset menjadi data latih dan data uji, dengan memperhatikan bahwa dataset yang telah melalui tahap *preprocessing*

merupakan faktor penting untuk evaluasi yang akurat terhadap performa model. Proses ekstraksi fitur kemudian dilakukan pada *dense layer* menggunakan fungsi aktivasi ReLU dalam ANN seperti yang dapat dilihat pada Gambar 3. 5. Asumsi yang mendasari pendekatan ini adalah bahwa ekstraksi fitur yang dilakukan oleh ANN dapat memberikan representasi fitur yang sudah dikenali polanya sehingga dapat meningkatkan akurasi model klasifikasi *Naïve Bayes*.

Dalam konteks ini, parameter *training* seperti *learning rate* dan jumlah *epoch* juga menjadi pertimbangan penting. Pengaturan parameter ini memiliki potensi untuk memengaruhi konvergensi model, yang pada gilirannya dapat memengaruhi hasil akhir eksperimen. Dengan merinci proses ini, diharapkan bahwa pendekatan ekstraksi fitur menggunakan ANN akan memberikan kontribusi positif dan membantu dalam meningkatkan akurasi serta kinerja keseluruhan dari model klasifikasi *Naïve Bayes* yang dikembangkan dalam penelitian ini.

Setelah proses ekstraksi fitur menggunakan algoritma ANN, fitur akan ditraining menggunakan algoritma *Naïve Bayes*, proses *modelling* dapat dilihat pada Gambar 3. 6



Gambar 3. 6 Proses Training Model Klasifikasi

Dari hasil *preprocessing* pada Tabel 3. *6*, output tersebut akan dilakukan perhitungan dengan TF-IDF untuk membentuk vektor input dari algoritma NN. Dapat dilihat pada Tabel 3. 7 dan Tabel 3. 8. Hasil dari TF-IDF yang berbentuk vektor akan menjadi input untuk training dan ekstraksi fitur oleh algoritma NN.

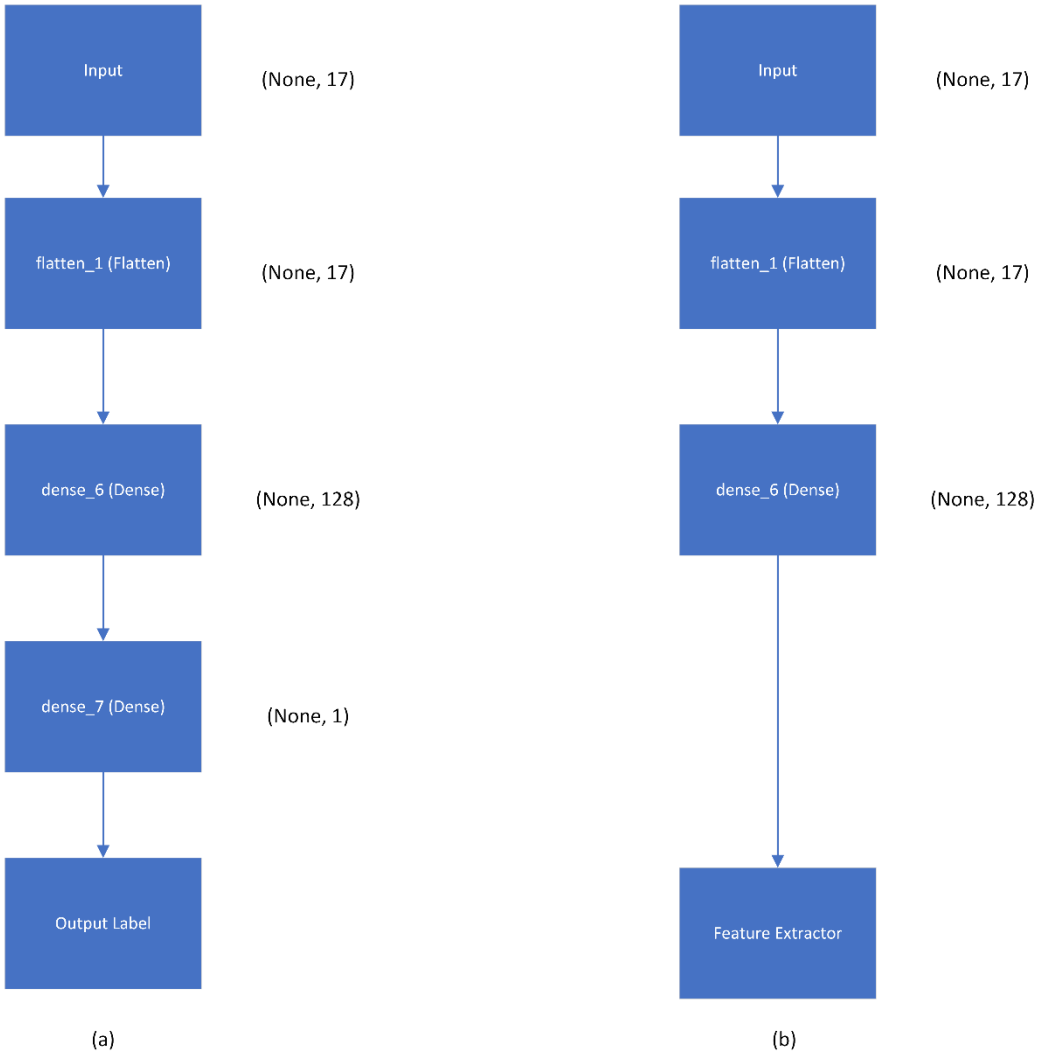
Tabel 3. 7 Hasil TF-IDF (1)

	ahok	aspal	banjir	cina	coblos	djarot	ga	kamboja
1	0,231	0,324	0	0,324	0,324	0	0	0,324
2	0,260	0	0,365	0	0	0,365	0,365	0

Tabel 3. 8 Contoh TF-IDF (2)

	kota	ktp	ngomong	pilih	pulau	rasa	sih	turun	Kejakarta
1	0,324	0,324	0	0	0,324	0	0	0,324	0,324
2	0	0	0,365	0,365	0	0,365	0,365	0	0

Karena output dari *preprocess* data adalah vektor yang seimbang antar kelas, maka tidak diperlukan *padding* pada fitur, sehingga ekstraksi fitur akan dapat langsung diproses untuk ekstraksi fitur menggunakan NN. Untuk mempersiapkan fitur sebelum terkoneksi di *dense layer* akan digunakan *flatten layer*. Konfigurasi NN akan menggunakan *optimizer* adam, dengan perhitungan *loss* menggunakan *binary crossentropy*, metrik *accuracy* dan menggunakan 10 *epoch*. Hasil dari training dan ekstraksi fitur dapat dilihat pada Gambar 3. 7. Pada proses ekstraksi fitur layer output akan dihilangkan, seperti yang dapat dilhat pada Gambar 3. 7 (b). Hasil dari ekstraksi fitur didapatkan 128 fitur dengan bentuk vektor yang akan menjadi input untuk model Naïve Bayes dapat dilihat pada Lampiran 1 Contoh Output Ekstraksi Fitur. Hasil dari ekstraksi fitur akan dihitung probabilitas dari setiap kelas, *likelihood* dan *prior probability* dari setiap fitur. Hasil dari perhitungan probabilitas dari setiap fitur dapat dilihat pada Lampiran 2 Contoh hasil probabilitas fitur



Gambar 3. 7 (a) Arsitektur ANN original (b) Menghilangkan layer output untuk ekstraksi fitur

3.2.4 Naïve Bayes Classifier

Setelah pembentukan model klasifikasi, langkah selanjutnya adalah melakukan uji validasi menggunakan algoritma *Naive Bayes*. Proses klasifikasi ini diawali dengan tahap ekstraksi fitur, di mana algoritma *Artificial Neural Network* (ANN) digunakan untuk mengidentifikasi dan mengekstrak fitur-fitur yang signifikan dari data. ANN memiliki kemampuan untuk memahami pola kompleks dan merepresentasikan data dengan cara yang mendalam, memungkinkan ekstraksi fitur yang lebih baik untuk digunakan dalam proses klasifikasi.

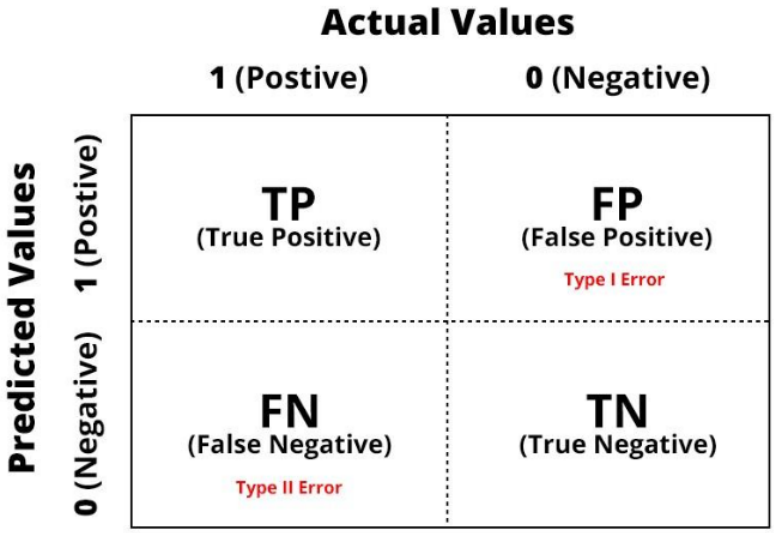
Setelah tahap ekstraksi fitur selesai, prediksi dilakukan menggunakan algoritma *Naive Bayes*. Algoritma ini dikenal karena pendekatannya yang sederhana dalam mengklasifikasikan data. Proses perhitungan Naive Bayes sangat bergantung pada *posterior probability*, yang diperoleh melalui perbandingan antara *likelihood* dan *prior probability*. Hasil klasifikasi diperoleh dari kategori dengan *posterior probability* terbesar, menunjukkan kecenderungan kelas yang paling mungkin.

ANN memberikan kontribusi dengan kemampuannya dalam ekstraksi fitur yang kompleks dan mendalam, sementara *Naive Bayes* menangani perhitungan *posterior probability* untuk memberikan hasil klasifikasi yang akurat. Implementasi kedua algoritma ini diharapkan dapat menciptakan model klasifikasi yang robust dan adaptif terhadap data teks yang kompleks.

Dalam konteks ini, fokus pada hasil klasifikasi Naive Bayes bergantung pada *posterior probability* terbesar, memastikan bahwa model memilih kelas yang paling mungkin secara probabilistik. Proses ini menandai tahap akhir dari perancangan model klasifikasi yang menggabungkan keunggulan dari ANN dalam ekstraksi fitur dan keandalan *Naive Bayes* dalam melakukan prediksi.

3.2.5 Evaluasi

Evaluasi model akan dilakukan dengan menggunakan Teknik *confision matrix*, Evaluasi menggunakan *confusion matrix* akan menghasilkan matriks performa berupa TP, TN, FP, dan FN seperti pada Gambar 3. 8. Yang kemudian dapat diukur performa klasifikasinya sehingga medapatkan nilai akurasi, *precision*, *recall* dan *f1-score*.



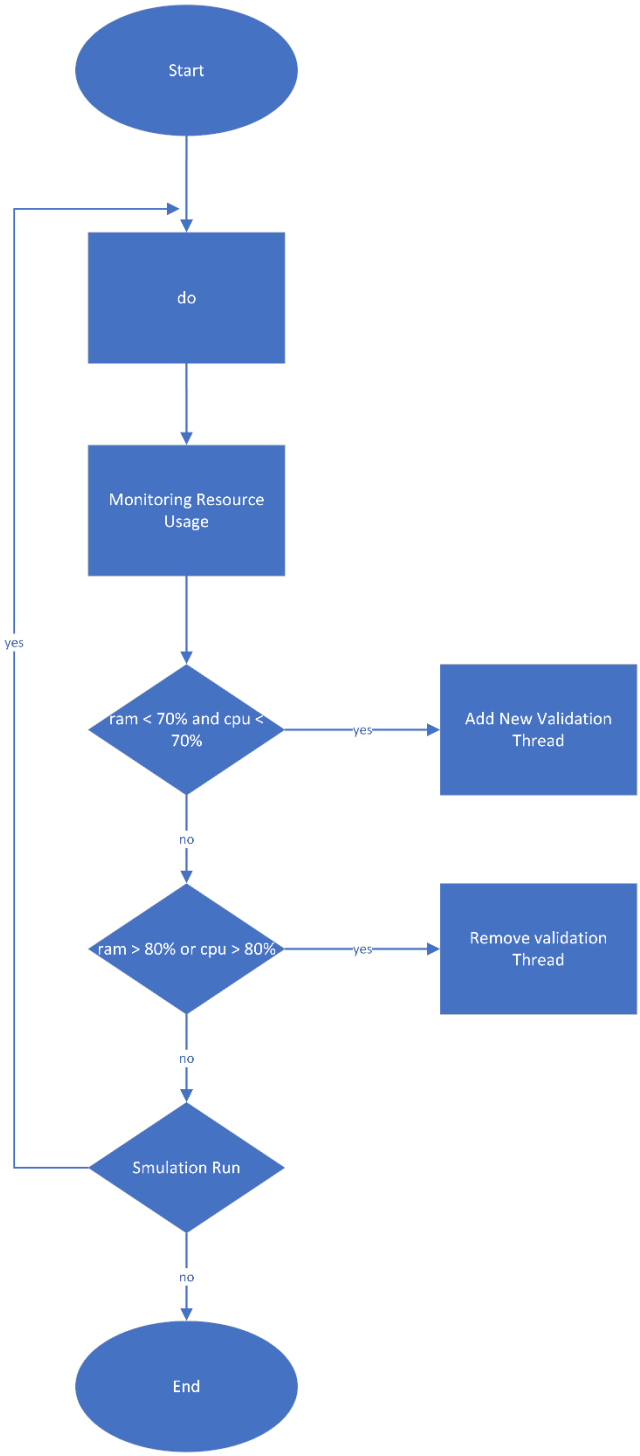
Gambar 3. 8 *Confusion matrix*

3.2.5. Resource-aware framework

Data stream merupakan proses *generating data*, dan proses yang dilakukan secara terus menerus, resource akan digunakan terus menerus selama proses dilakukan, sehingga sumber daya komputasi akan dikonsumsi selama simulasi dijalankan. Penggunaan *resource-aware* ditujukan untuk memastikan bahwa sumber daya akan tersedia selama simulasi dan menghindari *bottleneck* ketika simulasi dijalankan.

Pada penelitian ini, proses simulasi akan menggunakan *multithreading*, dengan *training data* dan validasi data akan dilakukan pada thread yang berbeda, sehingga akan proses pelatihan ulang model klasifikasi dan validasi data akan terjadi bersamaan, *resource-aware framework* akan digunakan untuk *monitoring* penggunaan CPU dan RAM, jika penggunaan CPU dan RAM turun di bawah 70%, *framework* secara otomatis akan menambahkan *thread* ke dalam pipa pemrosesan. Skalabilitas yang adaptif ini memastikan bahwa sistem dapat memanfaatkan sepenuhnya sumber daya yang tersedia, menjaga kinerja optimal bahkan selama beban data puncak. Dengan menyesuaikan jumlah *thread* berdasarkan sumber daya *real-time*, framework ini meningkatkan responsivitas dan throughput dalam pemrosesan aliran data teks.

Sebaliknya, ketika penggunaan CPU dan RAM melebihi 80%, framework yang memperhatikan sumber daya dirancang untuk mengurangi jumlah *thread*. Pengurangan ini mencegah *resource exhaustion* dan menghindari potensi ketidakstabilan sistem. Diagram alir dapat dilihat pada Gambar 3. 9.



Gambar 3. 9 *Flowchart resource-aware framework*

3.3. Pengujian dan Analisa Hasil

Dalam tahap ini, akan dilakukan perbandingan hasil uji dari model yang diusulkan dengan model dasar dalam melakukan klasifikasi, selain itu akan dilakukan juga dengan beberapa metode yang sudah ada sebelumnya.

3.4. Penyusunan Laporan

Tabel 3. *9* Jadwal Kegiatan Penelitian

No	Kegiatan	Bulan														
		I				II				III				IV		
1	Studi literatur															
2	Perancangan Metode															
3	Implementasi Metode															
4	Uji Coba dan Analisa Hasil															
5	Penyusunan Laporan															

Dalam fase penyusunan laporan, dilakukan pelaporan hasil penelitian dari setiap tahapan yang telah dilalui. Maksud dari tahap ini adalah untuk menghasilkan dokumentasi tertulis yang merinci seluruh aspek penelitian yang telah dilakukan. Tujuan utama penyusunan laporan ini adalah memberikan gambaran menyeluruh mengenai metodologi, temuan, dan analisis yang muncul dari penelitian tersebut. Laporan ini akan mencakup deskripsi rinci mengenai langkah-langkah yang diambil, metodologi yang diterapkan, serta interpretasi hasil penelitian.

Jadwal penelitian yang telah dilakukan dapat diakses dan dipahami melalui Tabel 3. 9. Tabel ini memberikan pandangan kronologis mengenai waktu pelaksanaan setiap tahap penelitian, membantu menggambarkan kerangka waktu yang telah diikuti. Pemahaman yang baik terhadap jadwal penelitian menjadi penting untuk mengevaluasi efisiensi dan konsistensi pelaksanaan, serta memastikan bahwa setiap tahapan dapat dilaksanakan sesuai dengan target waktu yang telah ditetapkan. Oleh karena itu, penyusunan laporan bukan hanya merupakan akhir dari proses penelitian, tetapi juga merupakan penegasan formal yang mengukuhkan seluruh perjalanan penelitian dari awal hingga akhir

[Halaman ini sengaja dikosongkan]..

BAB 4
HASIL DAN PEMBAHASAN

4.1. Dataset

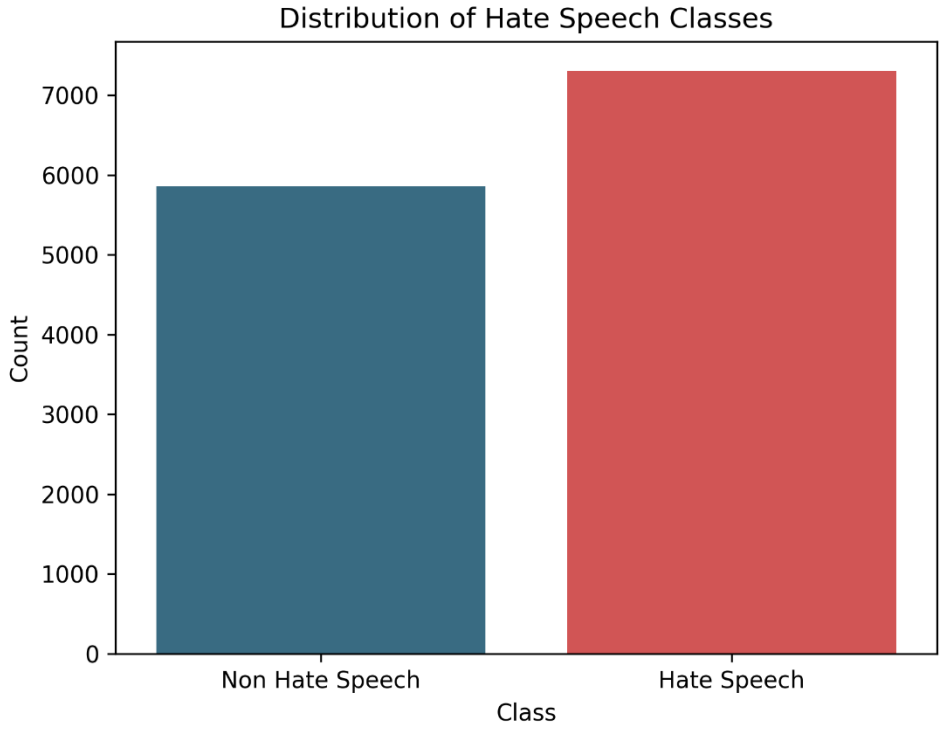
Pada penelitian ini, digunakan 3 dataset berbeda yang semua dataset merupakan dataset dari twitter berbahasa Indonesia yang dapat diakses secara publik melalui website kaggle.

Dataset pertama yang didapat dari peneltiian [21] memiliki 13085 *tweet* yang terbagi kepada 12 kelas, seperti kelas *hate speech*, *abusive*, *hate speech on individual*, dan lain-lain. Untuk menyederhanakan penulisan dataset ini akan dinamakan dataset 1. Selain itu pada penelitian ini dilakukan penyederhanaan terhadap kelas dengan membagi kelas hanya kepada 2 kelas, yaitu *hate speech* dan *non hate speech*. Implementasi penyederhanaan kelas dari dataset 1 dapat dilihat pada Gambar 4. 1

```
For each column in row:
     If column is not 'Tweet' AND value in row for this column is 1:
            Return 1
Return 0
```

Gambar 4. 1 Implementasi penyederhanaan kelas

Setelah dilakukan penyederhanaan kelas, didapatkan 7309 *tweet* yang dikategorikan sebagai *hate speech* dan 5860 *tweet* sebagai kategori *non hate* speech. Gambar menunjukkan bagaimana sebaran data dari *tweet* yang termasuk kedalam kelas *non hate speech* dan *hate speech*. Distribusi dataset 1 dapat dilihat pada Gambar 4. 2.



Gambar 4. 2 Distribusi dataset 1

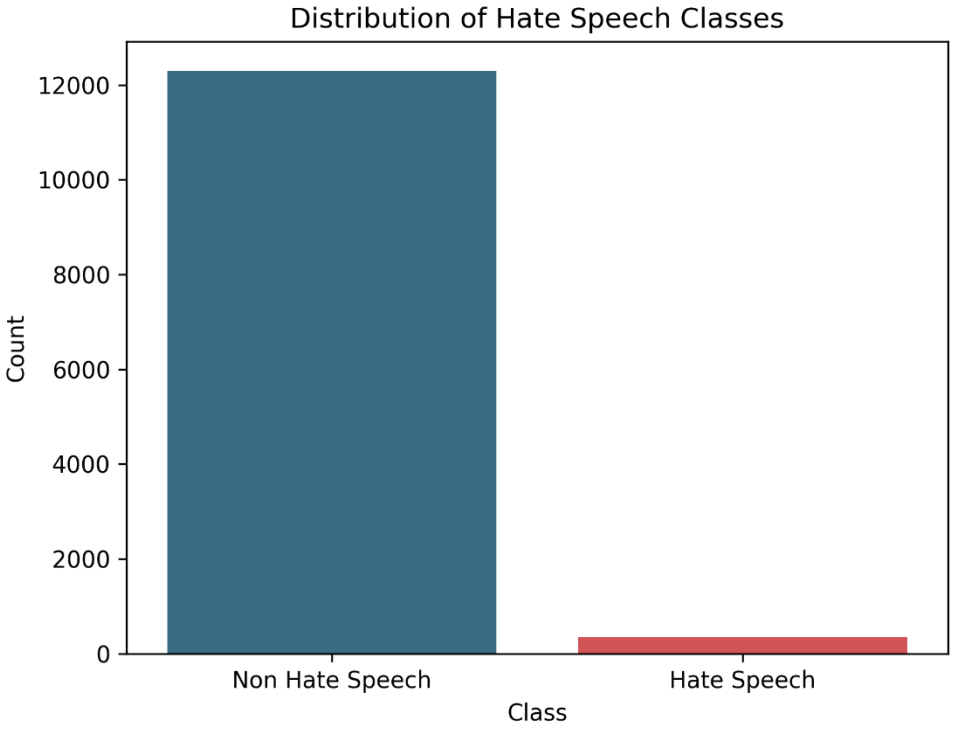
Dataset kedua yang digunakan pada penelitian ini didapat melalui Kaggle dengan judul "*Indonesia Twitter Comment Labeled with ITE Law*", yang pada penelitian ini akan disederhanakan penyebutannya sebagai dataset 2. Dataset ini memiliki 7 kelas dengan 3 kelas yang akan digolongkan sebagai *non hate speech* yaitu kelas *positive sentiment, neutral sentiment* dan *negative sentiment*. Sedangkan 4 kelas yang tergolong sebagai *hate speech* yaitu menghina pemerintahan sesuai dengan pasal 207 KUHP yang berbunyi "Barang siapa dengan sengaja dimuka umum, dengan lisan atau tulisan menghina kekuasaan yang ada di Negara Indonesia atau sesuatu majelis umum yang ada di sana, dihukum penjara selama-lamanya satu tahun enam bulan atau denda sebanyak-banyaknya Rp4.500.000" dan pasal 208 KUHP yang berbunyi "Barang siapa menyiapkan, mempertontonkan atau menempelkan tulisan atau gambar yang isinya penghinaan bagi sesuatu kekuasaan yang ada di Negara Indonesia atau bagi sesuatu mejelis umum yang ada di sana, dengan niat supaya isi yang menghina itu diketahui oleh orang banyak atau lebih diketahui oleh orang banyak, di hukum penjara paling lama 4 bulan atau denda

sebanyak Rp 4.500.000”. Kemudian pencemaran nama baik sesuai dengan pasal 27 ayat (3) UU ITE berbunyi sebagai berikut: “Setiap Orang dengan sengaja dan tanpa hak mendistribusikan dan/atau mentransmisikan dan/atau membuat dapat diaksesnya Informasi Elektronik dan/atau Dokumen Elektronik yang memiliki muatan penghinaan dan/atau pencemaran nama baik.”. Kemudian mengancam orang lain sesuai p asal 29 UU ITE, bunyi pasalnya sebagai berikut: “Setiap Orang dengan sengaja dan tanpa hak mengirimkan Informasi Elektronik dan/atau Dokumen Elektronik yang berisi ancaman kekerasan atau menakut-nakuti yang ditujukan secara pribadi.”. dan yang terakhir ucapan berbau sara sesuai pasal 28 ayat (2) UU ITE: “Setiap Orang dengan sengaja dan tanpa hak menyebarkan informasi yang ditujukan untuk menimbulkan rasa kebencian atau permusuhan individu dan/atau kelompok masyarakat tertentu berdasarkan atas suku, agama, ras, dan antargolongan (SARA).”. Implementasi penyederhanaan kelas klasifikasi dapat dilihat pada Gambar 4. 3.

```
If the value in row['sentimen'] is not 0, 1, or 2:
    Return 1
Else:
    Return 0
```

Gambar 4. 3 Implementasi penyederhanaan kelas pada dataset 2

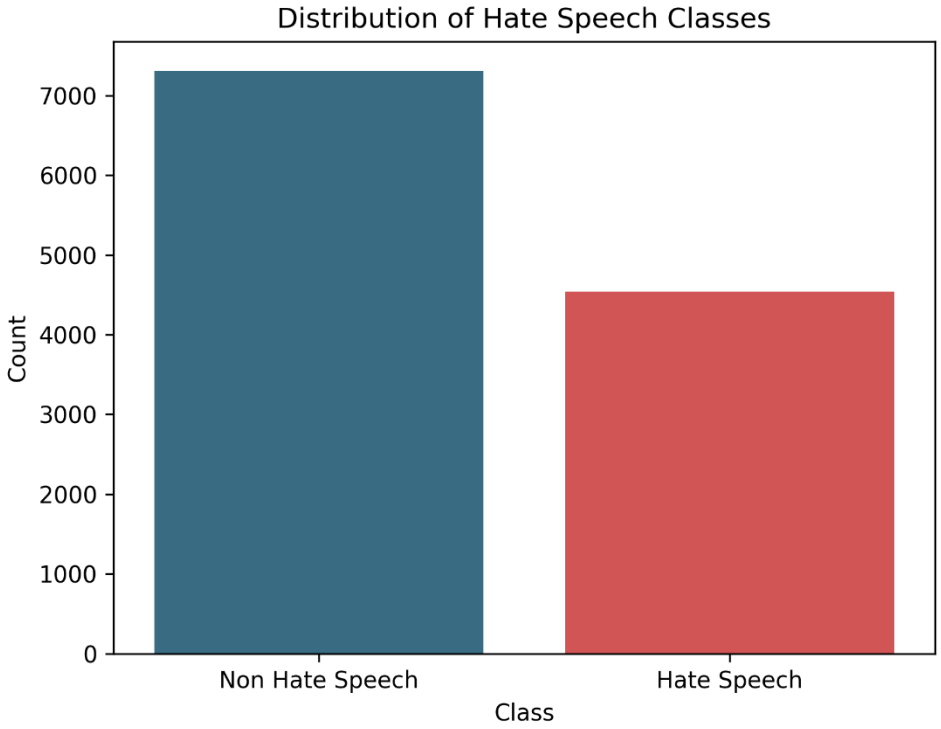
Setelah penyederhanaan kelas didapatkan total 12647 *tweet* dengan 12307 *tweet* dikategorikan sebagai *non hate speech*, dan 340 *tweet* dikategorikan sebagai *hate speech*. Distribusi dataset 2 dapat dilihat pada Gambar 4. 4.



Gambar 4. 4 Distribusi dataset 2

Dataset terakhir pada penelitian ini menggunakan dataset dari [22]. Pada dataset ini memiliki beberapa bahasa yang dikumpulkan, tetapi pada penelitian ini data *training* dan validasi yang digunakan hanya dengan dataset berbahasa Indonesia, yang pada penelitian ini akan disebutkan sebagai dataset 3.

Pada dataset 3, kelas klasifikasi sudah berbentuk *hate speech* dan *non hate speech*. Total dari dataset 3 berjumlah 11852 *tweet*, dengan 7313 *tweet* dikategorikan sebagai *hate speech* dan 4539 *tweet* dikategorikan sebagai *non hate speech*. Distribusi dari dataset 3 dapat dilihat pada Gambar 4. 5.



Gambar 4. 5 Distribusi dataset 3

4.2. Text Preprocessing

Tahapan *preprocessing* dilakukan untuk membersihkan data dari kata-kata yang tidak memiliki makna dalam kalimat sebelum dilakukan ekstraksi fitur. Pada penelitian ini ada 4 suh tahap *preprocessing* yang diimplementasikan.

4.1.1. Lowercasing

Pada tahapan ini, semua kata didalam kalimat akan dikonversikan menjadi huruf kecil. Implementasi *lowercasing* pada penelitian ini dapat dilihat pada Gambar 4. 6

```
Function lowercase(string):
    Return the lowercase version of string
```

Gambar 4. 6 Implementasi *lowercasing*

4.1.2. Punctuation removal

Pada tahapan ini dilakukan *special character* dan tanda baca akan dihapus karena token ini dianggap tidak bermakna. Implementasi *punctuation removal* dapat dilihat pada Gambar 4. 7

```
Function removeSpecialCharandExtraSpace(string):
    Remove all digits from the string
    Remove all punctuation from the string
    Remove leading and trailing whitespace from the string
    Return the cleaned string
```

Gambar 4. 7 Implementasi *punctuation removal*

4.1.3. Filtering

Pada tahapan ini kata-kata yang tidak memiliki makna seperti kata ganti orang, konjungsi, dan sebagainya akan dihapus, penelitian ini menggunakan *stopword* dari library python sastrawi. Implementasi *filtering* dapat dilihat pada Gambar 4. 8

```
Function filtering(string):
    { created stopword factory }
    tokens <- split string into array by space
    FOR EACH token IN tokens:
        IF token is not in stop word:
            Append token to cleaned tokens list

    cleaned_string <- Join the cleaned tokens with spaces
    Return cleaned_string
```

Gambar 4. 8 Implementasi *filtering*

4.1.4. Stemming

Tahapan terakhir yang digunakan pada peneltiain ini dalam tahap *preprocessing* adalah stemming, yaitu mengubah bentuk kata menjadi kata dasar. Implementasi dari proses *stemming* dapat dilihat pada Gambar 4. 9.

```
Function stemming(word):
    { Create dictionary }
    IF word is in the dictionary:
        Return word
    ELSE:
        FOR EACH rule in prefix rules:
            Apply rule to word to remove prefix
            IF resulting word is in the dictionary:
                Return resulting word

        FOR EACH rule in suffix rules:
            Apply rule to word to remove suffix
            IF resulting word is in the dictionary:
                Return resulting word

        Return original word (if no valid stem found)
```

Gambar 4. 9 Implementasi *stemming*

4.3. Oversampling

Dengan dataset yang tidak seimbang, performa dari algoritma sering kali akan bias terhadap kelas yang lebih besar, sehingga dibutuhkan algoritma untuk menyeimbangkan antar kelas. Pada penelitian ini menggunakan SMOTE untuk menyeimbangkan distribusi dari data. Implementasi SMOTE dapat dilihat pada Gambar 4. 10

```
Function SMOTE(data, N, k):
    # data: The minority class samples
    # N: Amount of SMOTE (e.g., 100 means 100%)
    # k: Number of nearest neighbors

    Initialize new_samples as an empty list
    num_samples <- Number of samples in data

    IF N < 100:
        Randomly select a percentage of the samples from data
        N <- 100

    N <- N / 100

    FOR EACH sample in data:
        Find k nearest neighbors of sample
        FOR i from 1 to N:
            neighbor <- Randomly select one of the k nearest neighbors
            synthetic_sample <- Create synthetic sample:
                FOR EACH feature:
                    difference <- neighbor[feature] - sample[feature]
                    gap <- Random number between 0 and 1
                    synthetic_sample[feature] <- sample[feature] + gap * dif-
ference
            Add synthetic_sample to new_samples

    Return data concatenated with new_samples
```

Gambar 4. 10 Implementasi SMOTE

4.4. Ekstraksi Fitur TF-IDF

Pada penelitian ini menggunakan TF-IDF pada kalimat sehingga didapatkan bentuk vektor dari kalimat yang ada. Implementasi TF-IDF pada penelitian ini dapat dilihat pada Gambar 4. 11

```
Function TFIDF(documents):
    # documents: List of documents, where each document is a list of words

    Initialize term_frequencies as an empty dictionary
    Initialize document_frequencies as an empty dictionary
    Initialize tfidf_values as an empty dictionary

    total_documents <- Number of documents

    # Calculate Term Frequencies (TF)
    FOR EACH document in documents:
        Initialize term_count as an empty dictionary
        total_terms <- Number of terms in document

        FOR EACH term in document:
            IF term is not in term_count:
                term_count[term] <- 0
            term_count[term] <- term_count[term] + 1

            IF term is not in document_frequencies:
                document_frequencies[term] <- 0
            document_frequencies[term] <- document_frequencies[term] + 1

        term_frequencies[document] <- {}
        FOR EACH term in term_count:
            term_frequencies[document][term] <- term_count[term] / total_terms

    # Calculate Inverse Document Frequencies (IDF)
    FOR EACH term in document_frequencies:
        idf[term] <- log(total_documents / document_frequencies[term])

    # Calculate TF-IDF
    FOR EACH document in term_frequencies:
        tfidf_values[document] <- {}
        FOR EACH term in term_frequencies[document]:
            tfidf_values[document][term] <- term_frequencies[document][term] *
idf[term]

    Return tfidf_values
```

Gambar 4. 11 Implementasi TF-IDF

Pada Tabel 4. 1, Tabel 4. 2 dan Tabel 4. 3 menunjukkan 10 kata urutan kata yang penting dari masing-masing dataset 1, dataset 2 dan dataset 3. Kata "user" memiliki nilai TF-IDF yang jauh lebih tinggi dibanding dengan kata lain

dalam korpus dataset 1, sedangkan kata “aku” merupakan kata dengan nilai TF-IDF tertinggi dari dataset 2 dan kata “user” juga memiliki nilai TF-IDF tertinggi pada dataset 3. Nilai TF-IDF yang tinggi mengindikasikan nilai keunikan dan pentingnya kata tersebut didalam korpus.

Tabel 4. 1 Top 10 nilai TF-IDF pada dataset 1

Term	**Rank**
user	1115.0755575670519
yg	279.4838603369785
url	158.652779563535
rt	158.42234071667775
jadi	156.34771635637549
jokowi	155.3884397732378
orang	142.14792193163913
presiden	133.28151667176195
sama	122.77685677130029
aja	122.24831721847342

Tabel 4. 2 Top 10 nilai TF-IDF pada dataset 2

Term	**Rank**
aku	588.5167537383571
kamu	361.9331923134265
yg	221.81428445874752
apa	195.63338960907114
orang	177.5784223503773
tak	171.70565176095946
jadi	149.38505370899563
ni	134.06959743257997
sama	134.0406916904683
mau	128.0127909374566

Tabel 4. 3 Top 10 nilai TF-IDF pada dataset 3

Term	**Rank**
user	1001.6490763947838

yg	252.91474071711903
url	142.87679011185355
rt	140.46760715791046
jadi	140.2739206563078
jokowi	139.9575553455323
orang	129.37600150562625
presiden	120.1645836658049
aja	111.23514528097536
sama	110.32447345003672

4.5. *Singular Value Decomposition* (SVD)

Pada tahap ini, dilakukan pengurangan dimensi menggunakan teknik SVD untuk mengolah data TF-IDF yang telah diperoleh sebelumnya. Dalam penelitian ini, digunakan metode *Truncated* SVD dengan jumlah komponen sebanyak 512. Proses ini akan mengurangi jumlah fitur secara signifikan dari ekstraksi TF-IDF, namun tetap mempertahankan informasi penting dari teks. Dengan demikian, SVD akan mengurangi kompleksitas data tanpa mengorbankan informasi yang terkandung dalam teks. Penggunaan SVD juga digunakan untuk mengantisipasi *exhausted* pada RAM selama pengujian *datastream mining*. Pada Gambar 4. 12 menunjukkan pseudocode pada algoritma SVD.

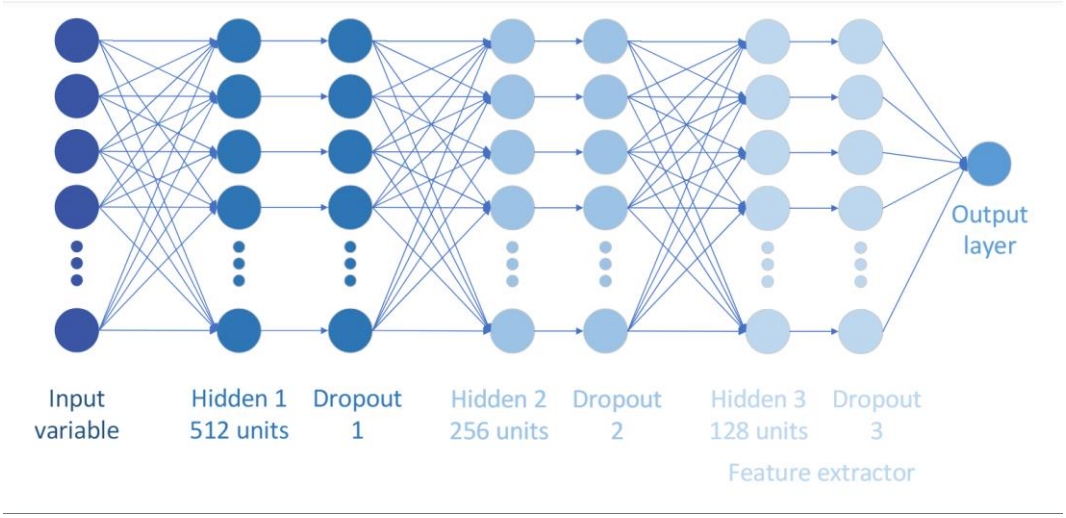
Input:
A: a real m x n matrix, with m <= n
k: desired rank of truncated SVD
p: parameter for oversampling dimension
ℓ = k + p (dimension of the approximate column space)
q: exponent of the power method
Output:
Approximate rank-k SVD of A

1. Generate an n x ℓ random matrix Ω
2. Compute Y = (AA^T)^q AΩ
3. Compute Q whose columns are an orthonormal basis of Y
4. Compute the SVD of Q^T A = $\hat{W}\ \hat{\Sigma}\ \hat{V}^T$
5. Assign $\tilde{U}_k = Q\ \hat{W}_k$
6. Extract the leading k singular vectors and singular values from $\tilde{U}_k$, $\hat{\Sigma}_k$, and $\hat{V}_k$

Gambar 4. 12 Pseudocode SVD

4.6. Ekstraksi Fitur ANN

Setelah nilai TF-IDF yang sudah tereduksi, selanjutnya vektor data dari dataset teks yang ada akan dilakukan proses *training* menggunakan ANN. Arsitektur ANN yang digunakan pada penelitian ini dapat dilihat pada Gambar 4. 13. Pada penelitian ini menggunakan 3 *hidden layer* yang masing masing memiliki 512 unit, 256 unit dan 128 unit, yang masing-masing diikuti oleh *dropout layer* yang masing-masing memiliki *dropout rates* 0,3, 0,2 dan 0,3. Serta memiliki extraksi fitur pada hidden layer terakhir yang memiliki 128 unit dengan fungsi aktifasi relu. Pada proses *training* ANN akan menggunakan *adam optimizer*.



Gambar 4. 13 Arsitektur ANN

Dengan total 128 unit pada layer ekstraksi fitur, maka akan ada 128 fitur dari setiap data dalam dataset. Contoh hasil ekstraksi fitur dapat dilihat pada Tabel 4. 4

Tabel 4. 4 Contoh ekstraksi fitur menggunakan ANN

Tweet	Ekstraksi Fitur ANN
barusan liat tulis belakang truk rela injek kopling kamu shopping	0.67929775, 0.0, 0.0, 0.0, 1.0087861, 0.80778384, 0.0, 0.0, 0.0, 1.9249076, 0.0, 1.4751441, 1.5883174, 0.0, 0.0, 0.0, 1.7242988, 1.2726113, 0.4215082, 0.0, 1.2467245, 0.96970975, 0.0, 0.0, 0.61261773, 0.6630342, 0.0, 1.6141379, 0.0,

	1.1764036, 1.5394752, 0.0, 1.8061445, 0.0, 0.0, 1.5098492, 1.9224081, 0.0, 0.0, 1.1770362, 0.0, 1.588235, 0.0, 1.0205113, 0.0, 0.0, 0.0, 0.0, 0.0, 1.227288, 1.4818089, 0.0, 0.0, 0.0, 0.0, 0.9692058, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07007249, 1.2538552, 0.0, 0.0, 0.0, 0.5735265, 0.9233314, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.6920248, 0.0, 0.0, 1.2803085, 0.92873114, 0.0, 1.1880606, 0.9905105, 0.0, 0.0, 1.9002733, 0.0, 2.070647, 1.4696475, 1.193936, 0.0, 0.0, 1.6624821, 1.9635105, 0.0, 0.0, 0.0, 0.0, 2.2184143, 0.0, 0.0, 0.0, 1.3003988, 0.0, 0.0, 1.107311, 0.0, 1.5060714, 0.0, 1.1941218, 1.0364305, 0.0, 1.9501091, 0.0, 1.3037797, 0.0, 1.2477934, 1.3882526, 0.0, 0.0, 0.9623589, 0.0, 1.3680141, 1.6933388, 1.8668189, 0.0, 1.1697253

4.7. Klasifikasi

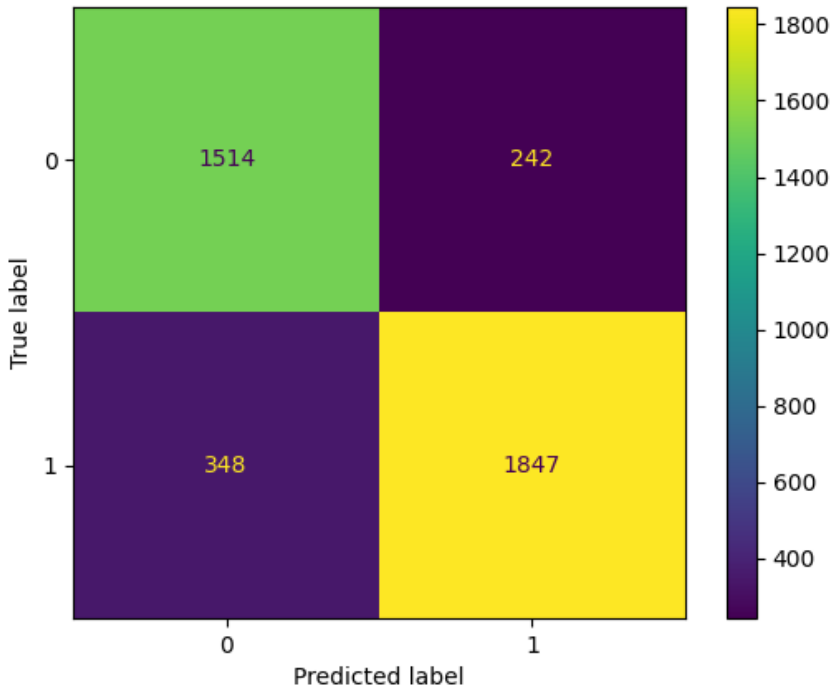
Metode yang digunakan dalam melakukan klasifikasi dalam penelitian ini adalah Naïve Bayes. Dengan keunggulan Naïve Bayes dalam komputasi yang ringan, penelitian ini mengharapkan masing-masing keunggulan dari algoritma mampu didapatkan dengan penggabungan 2 metode ini.

4.8. Analisis performa

Untuk membuktikan bagaimana pengaruh dari ekstraksi fitur menggunakan ANN terhadap algoritma Naïve Bayes, pada penelitian ini menggunakan 2 pendekatan dalam melakukan pengujian, pendekatan pertama yaitu dengan pendekatan klasifikasi klasik dengan membagi dataset menjadi 2 bagian yaitu data *training* dan data *testing*. Dan pendekatan kedua menggunakan pendekatan *data stream*, untuk melihat bagaimana pengaruh ANN yang memiliki komputasi lebih tinggi dibandingkan dengan algoritma Naïve Bayes klasik maka diterapkan *resource-aware framework*. Kedua pendekatan akan diuji performa dengan menggunakan 4 matrik pengujian yaitu *accuracy*, *recall*, *Precision*, dan *f1-score*.

4.1.1. Klasifikasi Klasik

Pada pendekatan pertama, dataset akan dibagi menjadi 2 bagian, yaitu data *training* dan data *testing*, dengan masing-masing besaran data 70% dan 30% dari total dataset yang ada.

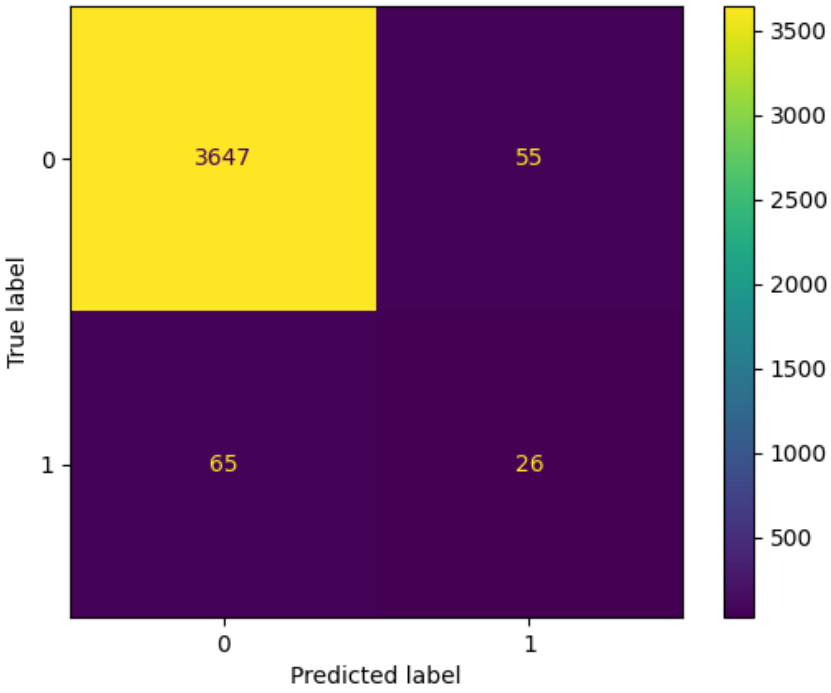


Gambar 4. 14 *Confusion matrix* dataset 1

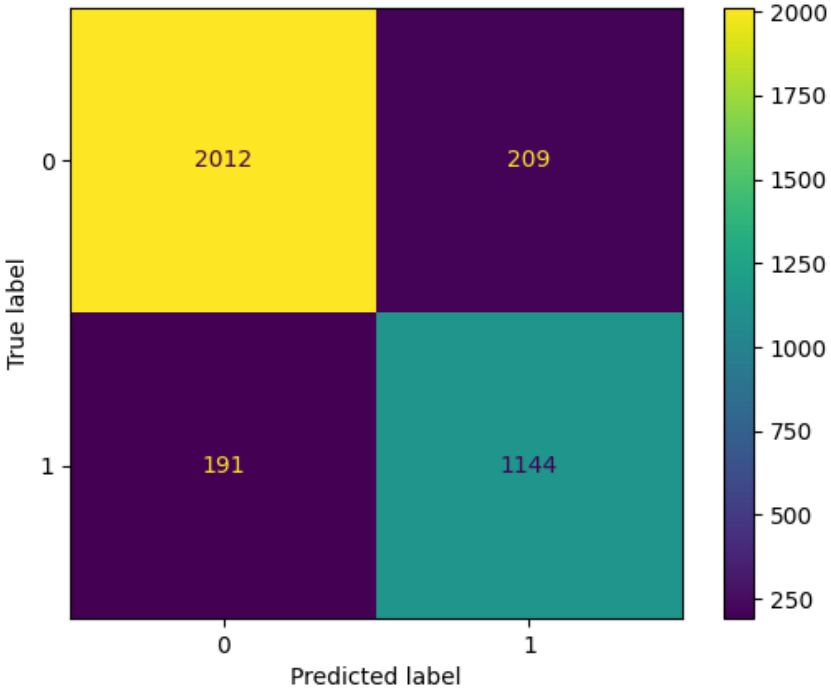
Pada Gambar 4. 14 menunjukkan bagaimana *confusion matrix* dari algoritma ANN-FTNB untuk dataset 1. Dari *confusion matrix* tersebut dapat dihitung matriks performanya yang menghasilkan akurasi sebesar 85%, *f1-score* sebesar 86,2%, *recall* sebesar 84,1% dan *Precision* sebesar 88,4%.

Kemudian pada Gambar 4. 15 menunjukkan *confusion matrix* untuk dataset 2. Dari *confusion matrix* yang ada, dapat dihitung performa ANN-NB untuk dataset 2 dengan akurasi 96,8%, *f1-score* sebesar 30,2%, *recall* sebesar 28,6% dan *Precision* sebesar 32,1%.

Yang terakhir pada Gambar 4. 16 menunjukkan *confusion matrix* untuk dataset 3. Dari *confusion matrix* tersebut dihasilkan akurasi sebesar 88,8%, *f1-score* sebesar 85,1%, *recall* sebesar 85,7%, dan *Precision* sebesar 84,6%.



Gambar 4. 15 *Confusion matrix* dataset 2



Gambar 4. 16 *Confusion matrix* dataset 3

Dari 3 hasil matriks performa diatas, dapat disimpulkan ANN-FTNB memiliki performa yang baik pada dataset 1 dan 3 dengan nilai matriks diatas 80%, tetapi memiliki performa yang relatif buruk untuk dataset 2, walaupun memiliki nilai akurasi yang sangat tinggi dengan nilai 97,6%, model memiliki *Precision* dan *recall* yang sangat rendah, menunjukkan banyak kesalahan pada *false-negative*.

Untuk membuktikan performa algoritma ANN-FTNB, penelitian ini mencoba membandingkan performa dari model usulan dengan model asli dari NB dan FTNB, selain itu performa model juga akan dibandingkan dengan beberapa model popular lainnya seperti *Support Vector Machine*(SVM), *Decission Tree* (DT), *Logistic Regression* (LR), *eXtreme Gradient Boosting* (XGBoost), dan *Artificial Neural Network* (ANN). Matriks perbandingan performa untuk dataset 1 dapat dilihat pada Tabel 4. 5, kemudian untuk dataset 2 dapat dilihat pada Tabel 4. 6, dan untuk dataset 3 dapat dilihat pada Tabel 4. 7

Tabel 4. 5 Perbandingan performa dari model pada dataset 1

Model	**Akurasi**	***Precision***	***Recall***	***F1-Score***
ANN-FTNB	**0,851**	**0,884**	**0,841**	**0,862**
NB	0,644	0,709	0,610	0,655
FTNB	0,646	0,700	0,634	0,665
SVM	0,842	0,897	0,808	0,850
DT	0,689	0,733	0,691	0,712
LR	0,835	0,881	0,813	0,846
XG boost	0,818	0,853	0,812	0,832
ANN	0,847	0,890	0,826	0,857

Tabel 4. 6 Perbandingan performa dari model pada dataset 2

Model	**Akurasi**	***Precision***	***Recall***	***F1-Score***
ANN-FTNB	0,968	0,321	0,286	0,302
NB	0,653	0,036	0,516	0,067
FTNB	0,659	0,036	0,516	0,068

Model	**Akurasi**	***Precision***	***Recall***	***F1-Score***
SVM	0,880	0,124	0,659	0,209
DT	0,914	0,073	0,220	0,110
LR	0,883	0,122	0,626	0,204
XG boost	0,972	0,318	0,154	0,207
ANN	0,968	0,299	0,253	0,274

Tabel 4. 7 Perbandingan performa dari model pada dataset 3

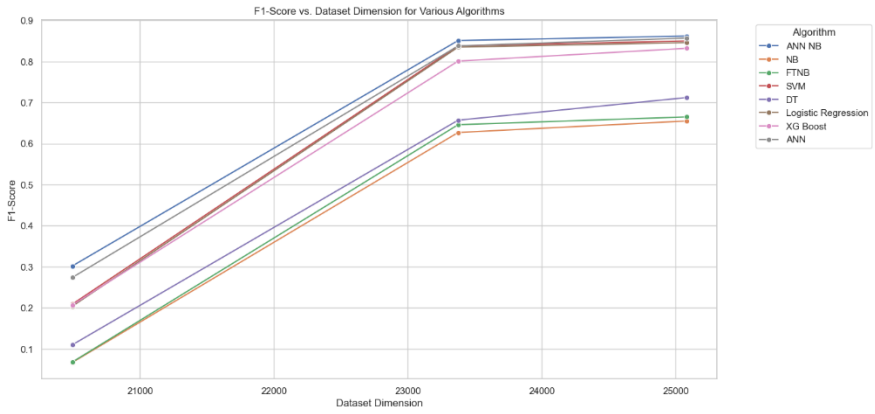
Model	**Akurasi**	***Precision***	***Recall***	***F1-Score***
ANN-FTNB	**0,888**	**0,846**	**0,857**	**0,851**
NB	0,713	0,611	0,644	0,627
FTNB	0,688	0,563	0,757	0,646
SVM	0,877	0,83	0,846	0,838
DT	0,733	0,635	0,681	0,657
LR	0,874	0,822	0,849	0,835
XG boost	0,854	0,821	0,783	0,801
ANN	0,881	0,854	0,823	0,838

Performa model di dataset 1 yang dapat dilihat pada Tabel 4. 5 menunjukkan ANN-FTNB memilki performa yang sangat baik, dengan akurasi sebesar 85,1%, nilai *recall* sebesar 84,1% dan *f1-score* 87,3% yang merupakan performa terbaik dibandingkan dengan model lain. Nilai *f1-score* tertinggi juga menunjukkan bahwa model mampu memberikan prediksi yang akurat dan seimbang antara *precision* dan *recall*. Meskipun SVM dan ANN memiliki nilai *Precision* yang sedikit lebih tinggi, ANN-FTNB unggul dalam keseimbangan model dalam melakukan klasifikasi.

Pada dataset 2 yang dapat dilihat di Tabel 4. 6 semua model menunjukkan performa yang kurang baik dikarenakan dataset yang sangat tidak *balance*, tetapi performa dari ANN-FTNB memiliki nilai akurasi. *precision,* dan *f1-score* terbaik dari model lainnya. Dengan nilai akurasi sebesar 96,8%, *precision* sebesar 32,1%,

dan *f1-score* 30,2%. Dengan memiliki nilai *f1-score* terbesar menunjukkan bahwa ANN-FTNB memiliki keseimbangan dalam klasifikasi terhadap kelas HS dan NON-HS.

Sedangkan hasil performa dengan dataset 3 yang dapat dilihat pada Tabel 4. 7 kembali menunjukkan performa yang seimbang dalam klasifikasi oleh ANN-FTNB yang ditunjukkan dengan memiliki nilai *f1-score* terbaik. Untuk melihat secara histogram perbandingan dari nilai *f1-score* dapat dilihat pada Gambar 4. 17.



Gambar 4. 17 Perbandingan *f1-score* dengan dimensi data

Pada Gambar 4. 17 menunjukkan bagaimana pendekatan ANN pada ekstraksi fitur mampu meningkatkan performa dari model NB. ANN-FTNB mampu secara konsisten memiliki nilai *f1-score* yang tinggi dibandingkan dengan model lain bahkan ketika dataset dengan jumlah fitur yang kecil. Dengan meningkatnya jumlah fitur juga menunjukkan bagaimana model ANN-FTNB mampu menghindari *overfitting* dan penurunan performa.

4.1.2. Datastream

Eksperimen kedua pada penelitian ini dilakukan dengan menggunakan pendekatan *datastream* mining. Pada eksperimen ini menggunakan dataset 1 karena memiliki jumlah data yang banyak. Selanjutnya seperti pada Gambar 3. 2 dataset akan dibagi menjadi 2 bagian sebesar 70% dan 30% untuk menjadi masing-masing data *training* dan data *testing*. Selanjutnya data training akan dilatih untuk menjadi model inisiator, sebelum data *testing* dimasukkan kedalam *datastream generator* untuk melakukan *randomize* dari data yang akan diuji.

Jumlah sampel yang diambil dari data *testing* adalah 100 – 200 data dalam sekali proses pengujian. Setelah data selesai diuji maka hasil pengujian akan dilakukan pengukuran performa menggunakan akurasi, *Precision, recall,* dan *f1-score*. Selanjutnya hasil klasifikasi dan data uji akan dimasukkan kedalam data *training* yang kemudian model akan dilakukan proses *training* ulang untuk melakukan pengujian selanjutnya.

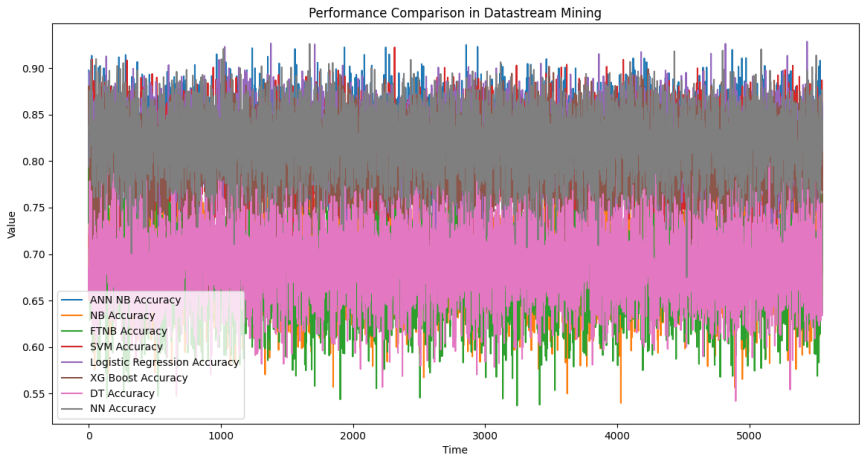
Untuk mengatasi keterbatasan data *testing*, setelah semua data *testing* digunakan maka pada iterasi di detik selanjutnya data *testing* akan digantukan dengan data augmentasi dari data *testing* yang ada. Proses augmentasi data dilakukan dengan memilih secara acak salah satu dari perubahan kata dengan kata sinonim, menambahkan kata pada kalimat, menghapus kata pada kalimat atau perubahan posisi kata pada kalimat.

Pada penelitian ini model akan dilatih ulang setiap 10 menit, dan proses streaming data akan dilakukan selama 30 menit. Untuk menghindari *error* karena *memory* yang tidak mencukupi maka dilakukan proses dekomposisi dari data training dengan menggunakan algoritma SVD.

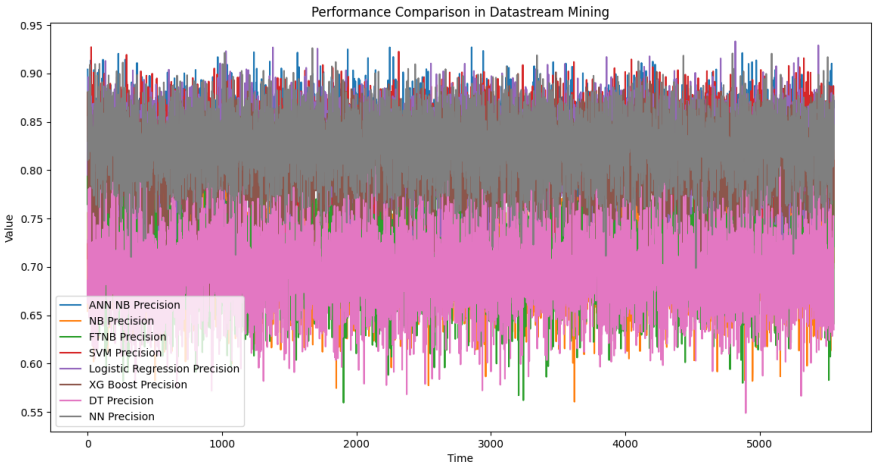
Untuk melihat bagaimana penggunaan *resource* komputasi dari setiap algoritma pada penelitian ini juga mengimplementasikan *resource-aware framework*. Dengan menggunakan *thread* untuk proses validasi data, dan *training*. *Thread* untuk proses validasi akan terus ditambah sebelum penggunaan RAM atau CPU diatas 70%. Tetapi jika CPU atau RAM komputer ada diatas 70% maka *thread* validasi akan dikurangi. Implementasi pada penelitian ini dapat dilihat pada Gambar 4. 18.

```python
def record_performance(stop_event):
    global performance, z
    threadnum = 2
    while not stop_event.is_set():
        cpu = psutil.cpu_percent()
        memory = psutil.virtual_memory().percent

        # Check if we should add a validation thread
        if cpu < 70 and memory < 70 and len(validation_events) < max_validation_threads:
            valid_event = threading.Event()
            validation_events.append(valid_event)
            threadnum+=1
            t = threading.Thread(target=validation, args=(stop_event,valid_event, threadnum))
            t.start()
        # Reduce to 1 thread if over threshold and more than 1 is running
        elif (cpu > 70 or memory > 70) and len(validation_events) > 1:
            while len(validation_events) > 1 and (cpu > 70 or memory > 70):
                thread_to_stop = validation_events.pop(1)
                thread_to_stop.set()

        total_thread.append(len(validation_events))
        performance['cpu'].append(cpu)
        performance['memory'].append(memory)
        print(f'CPU : {cpu} --- MEMORY : {memory} --- TOTAL THREAD : {len(validation_events)}')
        time.sleep(1)  # Check every 1 seconds
```

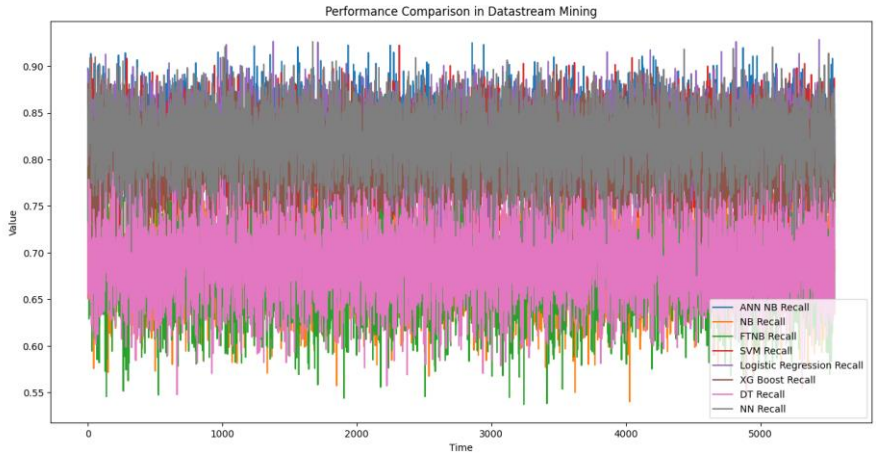
Gambar 4. 18 Implementasi *resource-aware framework*



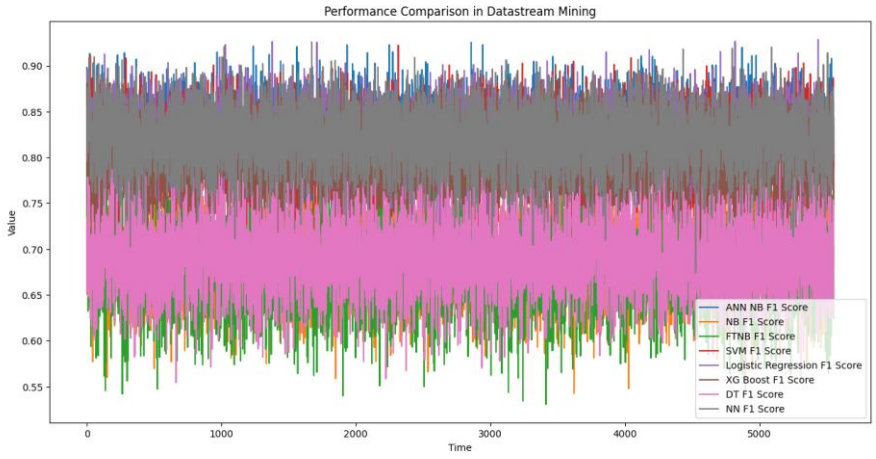
Gambar 4. 19 Perbandingan akurasi



Gambar 4. 20 Perbandingan *precision*



Gambar 4. 21 Perbandingan *recall*



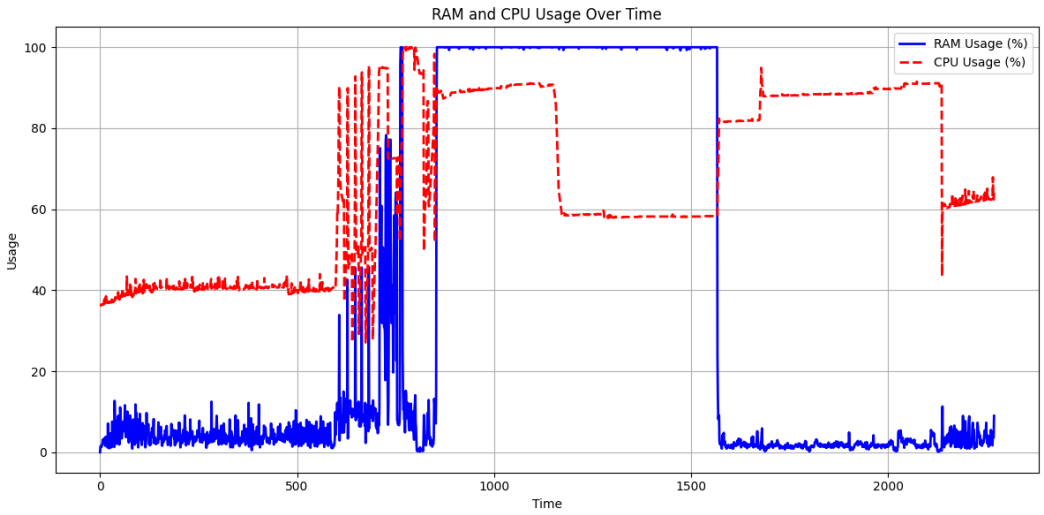
Gambar 4. 22 Perbandingan *F1-score*

Pada Gambar 4. 19, Gambar 4. 20, Gambar 4. 21 dan Gambar 4. 22 menunjukkan secara berturut turut perbandingan akurasi, *precision*, *recall*, dan *f1-score* dari model yang digunakan. Pada semua matriks secara sekilas terlihat performa ANN-FTNB lebih baik jika dibandingkan dengan model pembanding, dan terlihat jauh lebih baik dibandingkan dengan model NB dan FTNB.

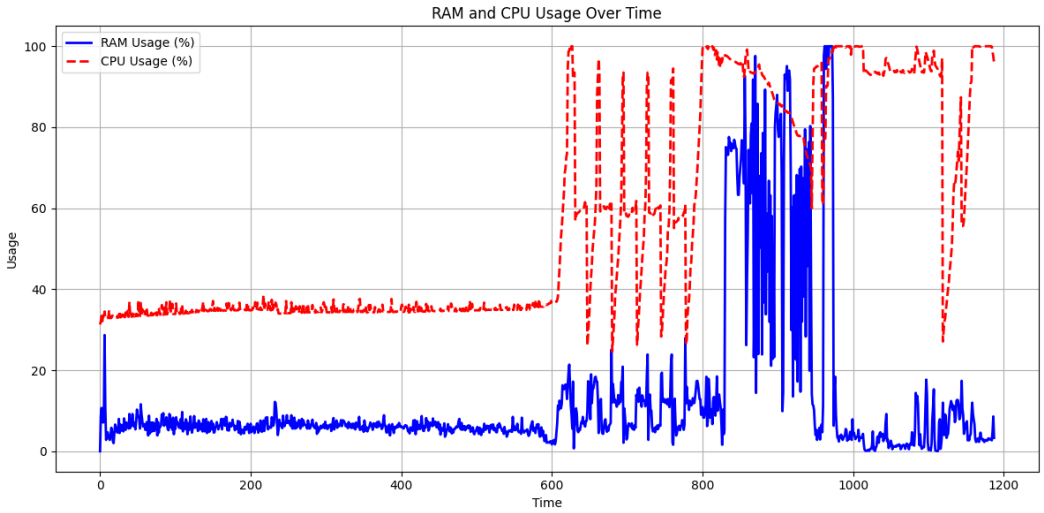
Tabel 4. 8 Rata-rata dan standar deviasi performa pada *datastream*

Model	**Rata-rata**				**Standar Deviasi**			
	acc	**prec**	**recall**	**f1**	**acc**	**prec**	**recall**	**f1**
ANN-FTNB	**0,827**	**0,828**	**0,832**	**0,827**	**0,031**	**0,031**	**0,031**	**0,031**
NB	0,689	0,689	0,710	0,689	0,038	0,038	0,037	0,038
FTNB	0,680	0,678	0,713	0,680	0,038	0,039	0,037	0,038
SVM	0,812	0,812	0,830	0,812	0,032	0,032	0,028	0,032
DT	0,690	0,691	0,694	0,690	0,038	0,038	0,038	0,038
LR	0,819	0,819	0,824	0,819	0,032	0,032	0,031	0,032
XGB	0,801	0,801	0,804	0,801	0,033	0,033	0,033	0,033
ANN	0,821	0,821	0,826	0,821	0,032	0,032	0,031	0,032

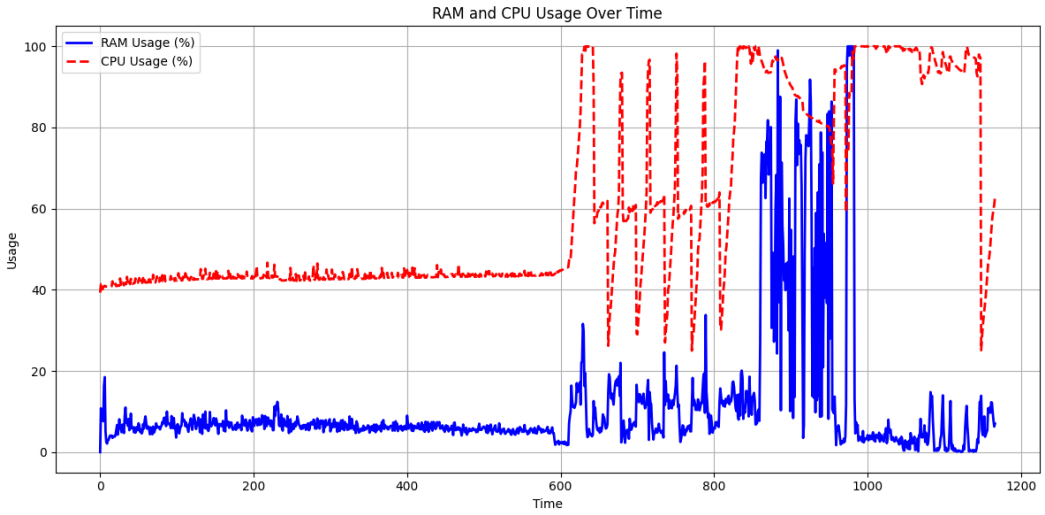
Pada Tabel 4. 8 menunjukkan rata-rata dan standar deviasi dari setiap matriks performa, dapat dilihat rata-rata baik akurasi, *precision*, *recall* dan *f1-score* model ANN-FTNB memiliki nilai rata-rata terbaik dibanding model lainnya. Untuk melihat persebaran nilai performa, maka dihitung nilai standar deviasi dari setiap matriks performa, dan didapatkan bahwa nilai standar deviasi dari setiap matriks performa ANN-FTNB memiliki nilai yang kecil yang menandakan sebaran data akurasi, *precision*, *recall* dan *f1-score* terpusat pada nilai rata-rata.



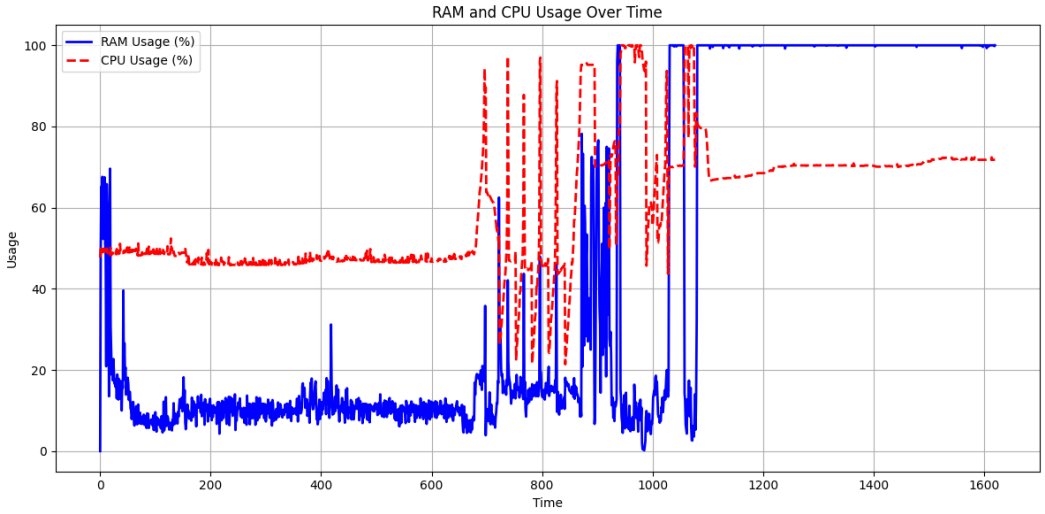
Gambar 4. 23 *Resource usage* ANN-FTNB



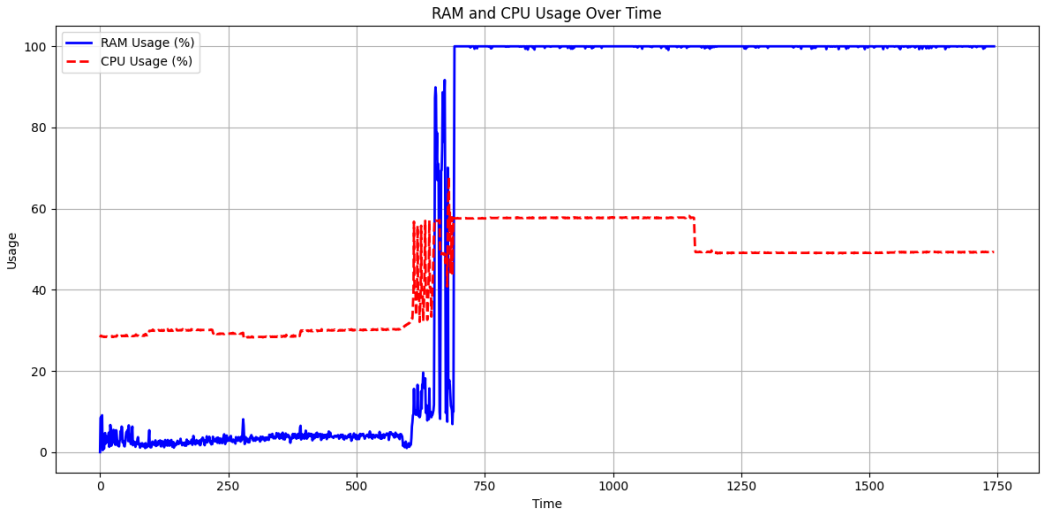
Gambar 4. 24 *Resource usage* NB



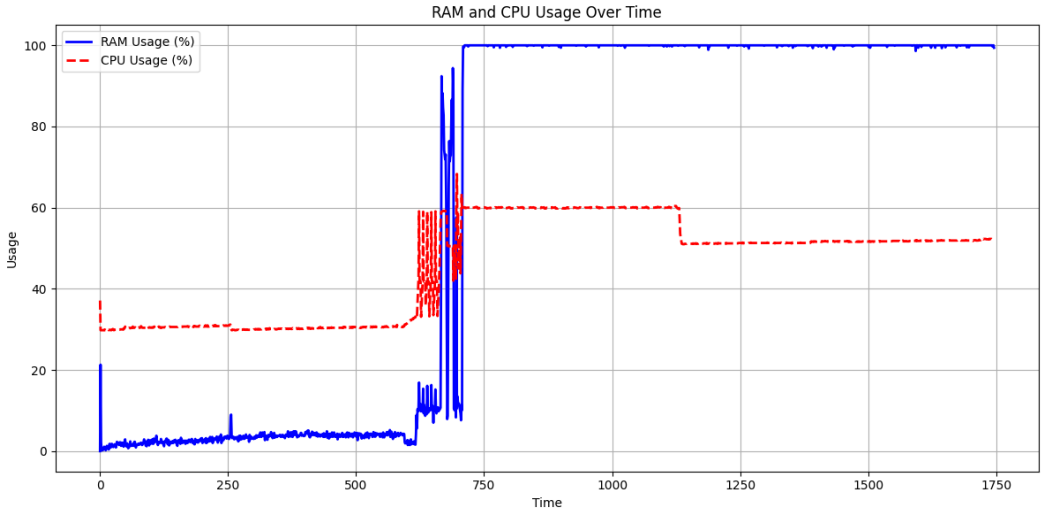
Gambar 4. 25 *Resource usage* FTNB



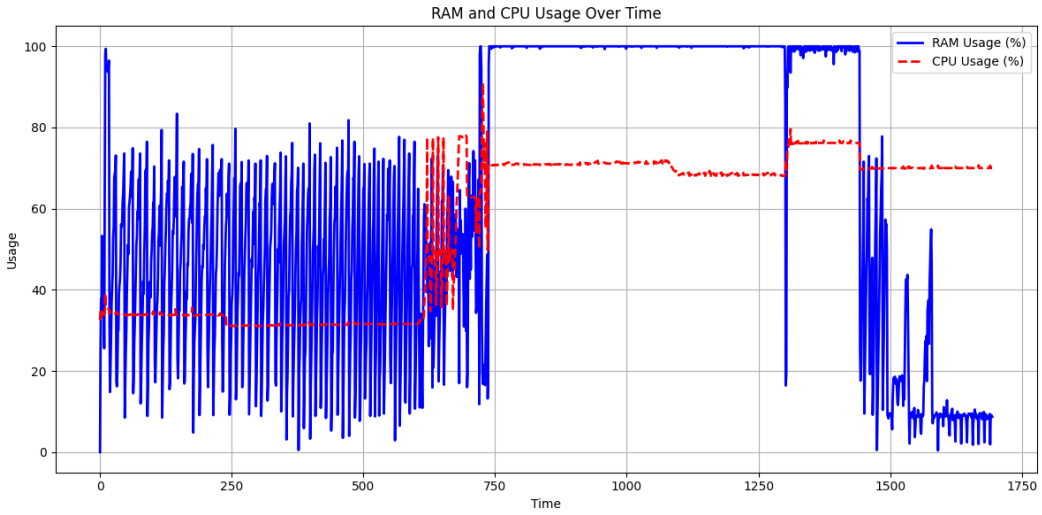
Gambar 4. 26 *Resource usage* SVM



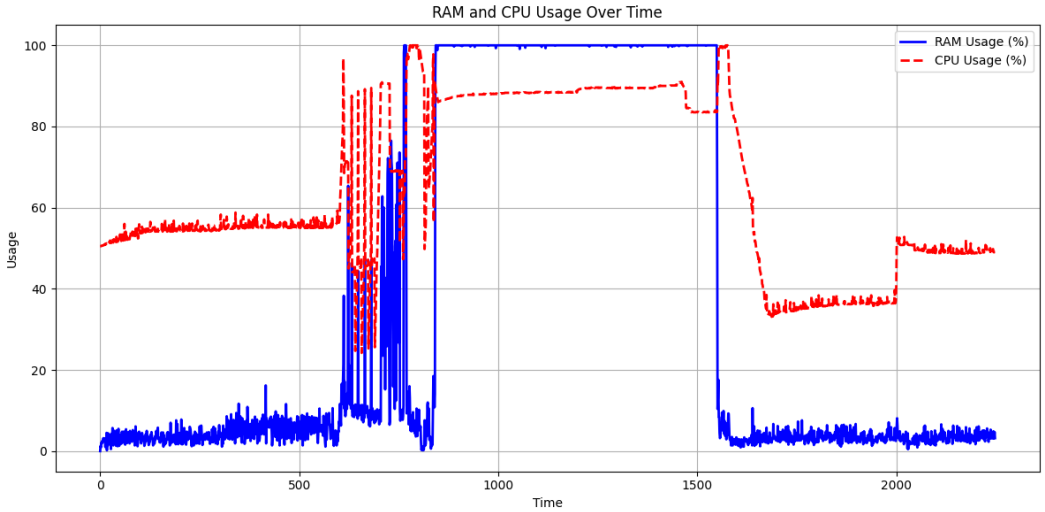
Gambar 4. 27 *Resource usage* DT



Gambar 4. 28 *Resource usage Logistic Regression*



Gambar 4. 29 *Resource usage XG Boost*



Gambar 4. 30 *Resource usage Artificial Neural Network*

Pada Gambar 4. 23 menunjukkan bagaimana penggunaan memori dan CPU dari algoritma ANN-FTNB, dari dapat dapat dilihat bahwa terdapat lonjakan yang tinggi penggunaan CPU secara periodik, hal ini disebabkan karena proses *training* dari algoritma ANN memilliki proses komputasi yang tinggi. Tetapi setelah proses *training* dapat dilihat penggunaan RAM menurun secara signifikan. Penggunaan memori dari awal proses *datastream* terlihat sangat kecil sebelum akhirnya ada lonjakan signifikan bahkan menyentuh 100% pada saat fase *retraining* pada model, Tetapi secara kesuluruhan penggunaan memori terlihat cukup rendah dan stabil dibawah 20%.

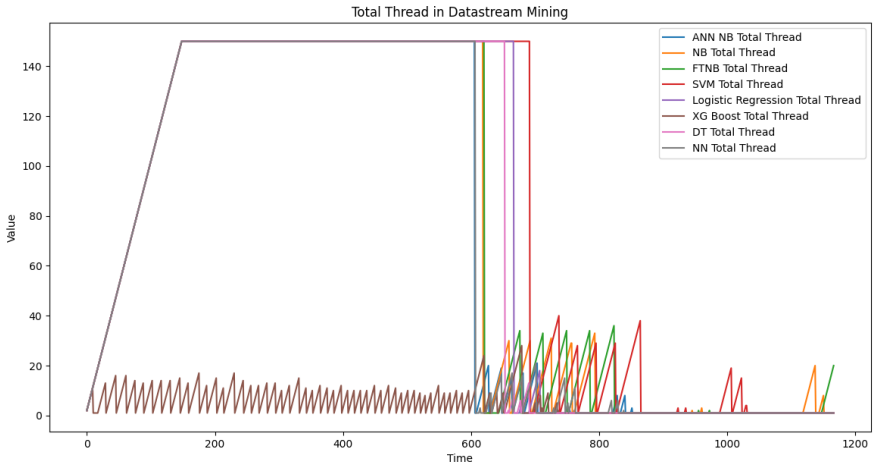
Selajutnya pada Gambar 4. 24 dan Gambar 4. 25 menunjukkan masing-masing penggunaan memori dan CPU dari algortima NB dan FTNB, pada 2 model terlihat bahwa penggunaan *resource* cukup stabil hanya terdapat lonjakan yang sangat signifikan di tengah fase *datastream*, bahkan penggunaan CPU mencapai 100 % yang terjadi pada saat *retraining* model. Tetapi setelah terjadi lonjakan signifikan pada fase awal, terlihat penggunaan memori dan CPU sangat stabil masing-masing di sekitar 10% dan 40%.

Kemudian pada Gambar 4. 26, Gambar 4. 27, Gambar 4. 28 masing-masing menunjukkan bagaimana penggunaan resource berturut-turut menggunakan algoritma SVM, DT, dan LR. Penggunaan memori dan CPU terlihat cukup mirip, dengan terjadi lonjakan di pertengahan fase *datastream*, yang disebabkan oleh pelatihan ulang yang dilakukan pada model dengan mengimplementasikan SMOTE. Yang kemudian setelah proses *retraining* selesai mengakibatkan memori stagnan di 100%.

Pada Gambar 4. 29 menunjukkan penggunaan *resource* dari algoritma *XG Boost*. Terlihat pada bagan, penggunaan ram terjadi cukup bervariasi, bahkan pada saat hanya thread validasi yang dijalankan lonjakan ram terjadi cukup fluktuatif, kemudian terdapat beberapa bagian yang cukup konsisten penggunaan ram ketika proses pelatihan ulang pada model. Tetapi penggunaan CPU terlihat cukup stabil, dan terlihat linier dengan penggunaan RAM.

Terakhir pada Gambar 4. 30 menunjukkan penggunaan *resource* dari algoritma ANN. Bagan terlihat sedikit mirip dengan ANN-FTNB yang terjadi lonjakan signifikan di pertengahan fase *datastream*, yang diakibatkan adanya proses *retraining* terhadap model yang digunakan.

Dari hasil analisis dapat disimpulkan bahwa ANN-FTNB, XG Boost dan ANN memiliki penggunaan *resource* yang tinggi dan cukup fluktuatif sepanjang proses *streaming* dijalankan, tetapi juga linier dengan performa ketiga algoritma ini yang jauh dibandingkan algoritma yang lainnya sehingga kedua algoritma ini dapat diandalkan untuk melakukan klasifikasi terus menerus. Dengan kemampuan pengenalan fitur pada algoritma ANN, terlihat mampu meningkatkan performa model FTNB khususnya pada *datastream mining*.



Gambar 4. 31 Total *thread* yang dijalankan selama masa *datastream mining*

Pada Gambar 4. 31 menunjukkan total *thread* yang dijalankan sistem selama periode *datastream*. Terlihat hampir semua algoritma secara signifikan menambah jumlah *thread* validasi, sebelum dilakukan proses *training* ulang pada model. Dengan algoritma *XG Boost* yang paling awal terjadi penurunan jumlah *thread*, menunjukkan *XG Boost* membutuhkan komputasi yang besar, di fase pertengahan terlihat ANN-FTNB dan ANN yang mengalami penurunan jumlah *thread*, yang kemudian tetap memiliki jumlah *thread* yang lebih banyak dari *XG Boost* menandakan bahwa model ANN-FTNB dapat bekerja dengan cukup baik untuk tetap menjaga performa model klasifikasi dan juga menjaga komputasi sehingga tetap mengoptimalkan penggunaan *resource* dibandingkan model *XG Boost*.

Selain itu, algoritma lain seperti NB, SVM dan *Logistic Regression* menunjukkan pola penggunaan thread yang berbeda. Pada SVM, menunjukkan fluktuasi yang cukup signifikan diakhir fase datastream, yang disebabkan oleh besarnya ukuran data. NB dan *Logistic Regression* menunjukkan penggunaan thread yang lebih stabil dan konsisten, yang menunjukkan efisiensi dalam komputasi mereka atau kebutuhan yang lebih rendah untuk *thread* tambahan selama validasi.

BAB 5
KESIMPULAN DAN SARAN

5. 1. Kesimpulan

Kesimpulan dari penerapan ANN pada ekstraksi fitur terhadap model NB pada klasifikasi *hate speech* dalam Bahasa Indonesia adalah sebagai berikut:

1. Penggunaan ANN pada ekstraksi fitur data teks sangat mempengaruhi performa dari model klasifikasi NB. Ekstraksi fitur oleh ANN yang dapat mengekstraksi fitur yang relevan, sehingga mengurangi dimensi fitur dan meningkatkan performa NB.
2. Penelitian ini membuktikan bahwa ANN-FTNB mampu secara konsisten mengungguli model lain yang diujikan pada 3 dataset yang digunakan, hal itu ditunjukkan dengan ANN-FTNB memiliki nilai *f1-score* terbaik pada 3 dataset yang yang diuji dengan eksprimen klasik yang menunjukkan model mampu melakukan klasifikasi secara seimbang terhadap kelas. Kemudian pada eksperimen *datastream mining* model ANN-FTNB juga mampu mengungguli model lain pada setiap komponen pengujian seperti akurasi, *precision, recall,* dan *f1-score*.
3. Proses *training* dengan 3 *hidden layer* mengakibatkan model NB mengkonsumsi *resource* lebih banyak dari NB klasik. Pada eksperimen *datastream mining* terlihat model ANN-FTNB mengalami penurunan performa di tengah fase *streaming* yang diakibatkan oleh dilakukannya proses *retraining* terhadap model klasifikasi. Hal ini dapat dianggap *trade-off* dari model ANN-FTNB untuk meningkatkan model NB maupun FTNB

5. 2. Saran

Berdasarkan hasil penelitian implementasi algoritma ANN pada ekstraksi fitur terhadap algoritma NB untuk dataset teks Twitter, didapatkan saran untuk penelitian lanjutan sebagai berikut :

1. Berdasarkan hasil ekstraksi fitur didapatkan beberapa fitur dengan nilai 0 yang akan mempengaruhi performa dari NB, untuk menghindari kondisi ini dapat mengimplementasikan *laplace smoothing*.
2. Melakukan riset untuk membuat arsitektur baku pada ANN, sehingga didapatkan model yang lebih baik untuk ekstraksi fitur yang mampu menunjang performa NB.
3. Implementasi SMOTE pada *datastream mining* terlihat sangat mempengaruhi komputasi dari algoritma, pada penelitian selanjutnya disarankan menggunakan teknik undersampling untuk menghindari tingginya penggunaan memori pada saat proses *datastream mining*.

DAFTAR PUSTAKA

[1] G. A. Buntoro, "Analisis Sentimen Hatespeech Pada Twitter Dengan Metode Naïve Bayes Classifier Dan Support Vector Machine," *Jurnal Dinamika Informatika,* vol. 5, no. 2, 2016.

[2] E. Nave and L. Lane, "Countering online hate speech How does human rights due diligence impact terms of service," *Computer Law & Security Review: The International Journal of Technology Law and Practice,* 2023.

[3] Komnas HAM, Buku saku penanganan ujaran kebencian (hate speech)., Komisi Nasional Hak Asasi Manusia Republik Indonesia, 2015, 2015.

[4] Indonesia, Undang-Undang No 11 Tahun 2008 Tentang Informasi dan Transaksi Elektronik.

[5] M. O. Ibrohim and I. Budi, "Hate speech and abusive language detection in Indonesian social media: Progress and challenges," *Heliyon,* 2023.

[6] I. Y. A. Sari and P. P. Adikara, "Klasifikasi Hate Speech Berbahasa Indonesia di Twitter Menggunakan Naive Bayes dan Seleksi Fitur Information Gain dengan Normalisasi Kata," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 3, no. 5, pp. 4914-4922, 2019.

[7] A. Fanani, S. Setiawan, O. Purwati and M. Maisarah, "ISIS grammar of persuasion of hatred in the article ‘The Kafir's blood is halal for you, so shed it’ published in the Rumiyah magazine," *Heliyon,* 2020.

[8] M. O. Ibrohim and I. Budi, "A Dataset and Preliminaries Study for Abusive Language Detection," in *3rd International Conference on Computer Science and Computational Intelligence 2018,* 2018.

[9] A. Alamsyah and Y. Sagama, "Empowering Indonesian internet users: An approach to counter online," *Intelligent Systems with Applications,* 2024.

[10] H. Zhang and L. Jiang, "Fine Tuning Attribute Weighted Naive Bayes," *Neurocomputing,* pp. 402-411, 2022.

[11] H.-C. Kim, J.-H. Park, D.-W. Kim and J. Lee, "Multilabel naïve Bayes classification considering label dependence," *Pattern Recognition Letters,* 2020.

[12] S. A. Panimalar and A. Krisnakumar, "Customer churn prediction model in cloud environment using DFE-WUNB: ANN deep feature extraction with Weight Updated Tuned Naïve Bayes classification with Block-Jacobi SVD

dimensionality reduction," *Engineering Applications of Artificial Intelligence*, 2023.

[13] S. J. S. Daisy and A. R. Begum, "Smart material to build mail spam filtering technique using Naive Bayes," in *Materials Today: Proceedings*, 2021.

[14] H. Zhang, L. Jiang and L. Yu, "Attribute and instance weighted naive Bayes," *Pattern Recognition*, 2021.

[15] X. Zhang, J. Xu, C. Soh and L. Chen, "LA-HCN: Label-based Attention for Hierarchical Multi-label Text," *Expert Systems With Applications*, 2022.

[16] S. Xu and Y. Xiang, "Frog-GNN: Multi-perspective aggregation based graph neural network for," *Expert Systems With Applications*, 2021.

[17] Y. Wang, C. Wang, J. Zhan, W. Ma and Y. Jiang, "Text FCG: Fusing Contextual Information via Graph Learning for text," *Expert Systems With Applications*, 2023.

[18] C. D. Manning, P. Raghavan and H. Schütze, An Introduction to Information Retrieval, Cambridgeshire: Cambridge University Press Cambridge, England, 2009.

[19] J. Leskovec, A. Rajaraman and J. D. Ullman, Mining of Massive Datasets, 2011.

[20] C. C. Aggarwal, Data Streams: Models and Algorithms, 2007.

[21] M. O. Ibrohim and I. Budi, "Multi-label Hate Speech and Abusive Language Detection in Indonesian Twitter," in *ALW3: 3rd Workshop on Abusive Language Online*, 2019.

[22] W. H. Moosa and N. , *Multi-lingual HateSpeech Dataset*, Kaggle, 2022.

[23] T. H. J. Hidayat, Y. Ruldeviyani, A. . R. Aditama, G. R. Madya, A. W. Nugraha and M. W. Adisaputra, "Sentiment analysis of twitter data related to Rinca Island development using Doc2Vec and SVM and logistic regression as classifier," in *Procedia Computer Science*, 2022.

BIOGRAFI PENULIS

Penulis tesis ini bernama Reza Wahyu Ramadhan yang dilahirkan di Lhokseumawe tanggal 16 Januari 1998. Beralamat di Jln Bakti II No, 50, Geuceu Komplek, Banda Raya, Provinsi Aceh. Lahir dari pasangan suami istri Bapak Ridwan Hadi dan Ibu Ermedah yang merupakan anak pertama dari 4 bersaudara. Penulis tesis ini menyelesaikan pendidikan dasar di MIN Kuta Blang dari tahun 2004 hingga 2010, setelah itu penulis melajutkan pendidikan menengah pertama di SMPN 1 Lhokseumawe dari tahun 2010 hingga 2013, kemudian melanjutkan ke pendidikan menengah atas di SMAN 1 Banda Aceh dari tahun 2013 hingga tahun 2016. Setelah lulus SMA penulis melanjutkan pendidikan ke perguruan tinggi di Institut Teknologi PLN dari tahun 2016 dan lulus pada tahun 2020 sebagai sarjana Teknik dari program studi Teknik Informatika. Setelah lulus perguruan tinggi penulis memilih untuk bekerja sebagai Backend Engineer pada beberapa Perusahaan dan instansi pemerintahan sebelum akhirnya memutuskan untuk melanjutkan ke Pendidikan pasca sarjana di Institut Teknologi Sepuluh Nopember pada program studi yang linier di Teknik Informatika. Setelah menempuh pembelajaran di ITS penulis akhirnya dapat meyelesaikan tesis sebagai syarat kelulusan pasca sarjana di ITS. Akhir kata penulis mengucapkan rasa syukur yang sebesar-besarnya atas penyelesaian Tesis yang berjudul "**Deteksi Hatespeech Menggunakan Neural Network – Naïve Bayes Classifier Dari Dataset Twitter**"

Lampiran 1

Contoh Output Ekstraksi Fitur

Fitur	Dokumen 1	Dokumen 2
0	0.06501728	0.0
1	0.0	0.21906151
2	0.0	0.057064515
3	0.0	0.0
4	0.0	0.06886956
5	0.0	0.20532218
6	0.018402709	0.0
7	0.0	0.0
8	0.035594333	0.026770052
9	0.0	0.0
10	0.0	0.0
11	0.09214199	0.12729685
12	0.0	0.1267047
13	0.12838323	0.017483464
14	0.14399034	0.103613645
15	0.0	0.10915665
16	0.0	0.0
17	0.0	0.07939072
18	0.0	0.0
19	0.20810409	0.23840177
20	0.0	0.0
21	0.0	0.0
22	0.0	0.0
23	0.0	0.137169
24	0.0	0.0
25	0.0	0.2312441
26	0.0	0.0
27	0.1335651	0.0
28	0.0	0.0
29	0.0	0.09377981
30	0.0	0.0
31	0.049096815	0.12942027
32	0.058739718	0.134391
33	0.0	0.0
34	0.0	0.0
35	0.048728608	0.06665516

Fitur	Dokumen 1	Dokumen 2
36	0.05566495	0.0
37	0.0	0.0
38	0.0	0.0
39	0.0	0.0
40	0.09908116	0.055319775
41	0.09659823	0.067456
42	0.0	0.1471075
43	0.0	0.25576425
44	0.15897305	0.0
45	0.061876927	0.049749095
46	0.0	0.0
47	0.0	0.0019011497
48	0.042215127	0.11316428
49	0.051918957	0.11888262
50	0.07347057	0.08389341
51	0.0	0.0
52	0.1401445	0.0
53	0.13251701	0.15776396
54	0.0	0.0
55	0.118723884	0.18305591
56	0.08895901	0.15199542
57	0.18162611	0.07199058
58	0.028539404	0.0
59	0.0	0.10209632
60	0.037560888	0.042771127
61	0.0	0.0
62	0.17474121	0.12744676
63	0.0	0.0
64	0.23677915	0.048920844
65	0.0	0.0
66	0.0665769	0.0
67	0.0846447	0.07075926
68	0.22624779	0.26629734
69	0.05555314	0.19121413
70	0.0	0.0
71	0.25465104	0.2626794
72	0.09953296	0.0
73	0.0	0.0
74	0.0	0.03454859

Fitur	Dokumen 1	Dokumen 2
75	0.21124056	0.2062549
76	0.036188606	0.0
77	0.08482724	0.06449176
78	0.0	0.0
79	0.0	0.10256445
80	0.0	0.16520867
81	0.14710072	0.0
82	0.017262535	0.20207609
83	0.15038736	0.05581898
84	0.0	0.0
85	0.0	0.0
86	0.05176752	0.05020741
87	0.0	0.0
88	0.0	0.0
89	0.018600361	0.064715475
90	0.0	0.0
91	0.0	0.0
92	0.0	0.0
93	0.0	0.19255377
94	0.0	0.0
95	0.070400104	0.057731014
96	0.19342889	0.117617086
97	0.1394305	0.03821778
98	0.0	0.0
99	0.11087524	0.008766951
100	0.052095816	0.0
101	0.0	0.1343294
102	0.0	0.0
103	0.21443486	0.04931827
104	0.0	0.036186047
105	0.0	0.0
106	0.0	0.1352508
107	0.06876501	0.0
108	0.18610016	0.0
109	0.24969815	0.019885374
110	0.0	0.0
111	0.15926027	0.0
112	0.0	0.0
113	0.0	0.0

Fitur	Dokumen 1	Dokumen 2
114	0.11608879	0.0
115	0.16457252	0.1423041
116	0.0052432977	0.15697157
117	0.09573625	0.24510859
118	0.10160379	0.0
119	0.0	0.0
120	0.0	0.0
121	0.0	0.0
122	0.0	0.0
123	0.0	0.06772404
124	0.0	0.17943172
125	0.08795334	0.24572192
126	0.05548799	0.0
127	0.07744167	0.0

Lampiran 2

Contoh hasil probabilitas fitur

Fitur	Kelas Positive	Kelas Negative
0	0.00792338960703814	0.0073792553398163655
1	0.00743968171577071	0.008995766148190726
2	0.00743968171577071	0.007800348963945506
3	0.00743968171577071	0.0073792553398163655
4	0.00743968171577071	0.007887461415258784
5	0.00743968171577071	0.008894380105055999
6	0.007576592011164636	0.0073792553398163655
7	0.00743968171577071	0.0073792553398163655
8	0.007704492223902291	0.007576798385802196
9	0.00743968171577071	0.0073792553398163655
10	0.00743968171577071	0.0073792553398163655
11	0.008125188818905471	0.00831861130054981
12	0.00743968171577071	0.00831424162103998
13	0.008394812115437808	0.007508270281287441
14	0.00851092399951106	0.008143846881767468
15	0.00743968171577071	0.008184750156446381
16	0.00743968171577071	0.0073792553398163655
17	0.00743968171577071	0.007965099730861873
18	0.00743968171577071	0.0073792553398163655
19	0.008987909900955568	0.009138482878477109
20	0.00743968171577071	0.0073792553398163655
21	0.00743968171577071	0.0073792553398163655
22	0.00743968171577071	0.0073792553398163655
23	0.00743968171577071	0.008391460441252396
24	0.00743968171577071	0.0073792553398163655
25	0.00743968171577071	0.009085664615189365
26	0.00743968171577071	0.0073792553398163655
27	0.008433363533578006	0.0073792553398163655
28	0.00743968171577071	0.0073792553398163655
29	0.00743968171577071	0.008071280501850507
30	0.00743968171577071	0.0073792553398163655
31	0.007804946394771914	0.008334280525496366
32	0.007876686522135072	0.008370960806602837
33	0.00743968171577071	0.0073792553398163655
34	0.00743968171577071	0.0073792553398163655
35	0.007802207046729949	0.007871120777768205

Fitur	Kelas Positive	Kelas Negative
36	0.007853811219941319	0.0073792553398163655
37	0.0074396817157707 1	0.0073792553398163655
38	0.0074396817157707 1	0.0073792553398163655
39	0.0074396817157707 1	0.0073792553398163655
40	0.008176814000067216	0.007787474084114423
41	0.008158341803526795	0.007877030384721996
42	0.0074396817157707 1	0.008464799121530146
43	0.0074396817157707 1	0.009266605017749585
44	0.008622390632202102	0.0073792553398163655
45	0.007900026355010038	0.007746366614693576
46	0.0074396817157707 1	0.0073792553398163655
47	0.0074396817157707 1	0.007393284409259042
48	0.007753748827945855	0.008214323481894194
49	0.007825942233710732	0.00825652053973606
50	0.007986279373140876	0.007998326239891437
51	0.0074396817157707 1	0.0073792553398163655
52	0.008482312168829147	0.0073792553398163655
53	0.008425566091899359	0.008543435869150887
54	0.0074396817157707 1	0.0073792553398163655
55	0.008322949626463915	0.008730071622631156
56	0.008101508424428526	0.008500868357843437
57	0.008790922174885201	0.007910492207024443
58	0.007652005799919786	0.0073792553398163655
59	0.0074396817157707 1	0.008132650148438293
60	0.007719122764735111	0.007694874407674203
61	0.0074396817157707 1	0.0073792553398163655
62	0.00873970069003648	0.008319717492851302
63	0.0074396817157707 1	0.0073792553398163655
64	0.00920124325360217	0.0077402547372886375
65	0.0074396817157707 1	0.0073792553398163655
66	0.007934992647243068	0.0073792553398163655
67	0.008069411324891203	0.00790140598084622
68	0.009122893243486408	0.009344331410890796
69	0.007852979382708099	0.008790273224713716
70	0.0074396817157707 1	0.0073792553398163655
71	0.009334204400761362	0.009317633690609225
72	0.008180175272128422	0.0073792553398163655
73	0.0074396817157707 1	0.0073792553398163655
74	0.0074396817157707 1	0.007634198220504843

Fitur	Kelas Positive	Kelas Negative
75	0.009011244246265657	0.008901262908328314
76	0.007708913427433717	0.0073792553398163655
77	0.008070769358623052	0.007855156531554759
78	0.0074396817157707 1	0.0073792553398163655
79	0.0074396817157707 1	0.008136104580272037
80	0.0074396817157707 1	0.00859837228177298
81	0.008534064229142416	0.0073792553398163655
82	0.007568109483040727	0.00887042642343086
83	0.00855851582048921	0.007791157836429749
84	0.0074396817157707 1	0.0073792553398163655
85	0.0074396817157707 1	0.0073792553398163655
86	0.007824815592297805	0.0077497486382141455
87	0.0074396817157707 1	0.0073792553398163655
88	0.0074396817157707 1	0.0073792553398163655
89	0.007578062484546489	0.007856807353127545
90	0.0074396817157707 1	0.0073792553398163655
91	0.0074396817157707 1	0.0073792553398163655
92	0.0074396817157707 1	0.0073792553398163655
93	0.0074396817157707 1	0.008800158801283153
94	0.0074396817157707 1	0.0073792553398163655
95	0.007963436081742228	0.007805267231267684
96	0.008878731083948129	0.008247181846607122
97	0.008477000206029527	0.007661274091415127
98	0.0074396817157707 1	0.0073792553398163655
99	0.008264558222380548	0.007443948910088011
100	0.007827258002107053	0.0073792553398163655
101	0.0074396817157707 1	0.008370506234140181
102	0.0074396817157707 1	0.0073792553398163655
103	0.009035008838833979	0.0077431874389113945
104	0.0074396817157707 1	0.007646281419617971
105	0.0074396817157707 1	0.0073792553398163655
106	0.0074396817157707 1	0.008377305578172253
107	0.007951271480731693	0.0073792553398163655
108	0.008824207637082814	0.0073792553398163655
109	0.009297356455790443	0.007525994593810412
110	0.0074396817157707 1	0.0073792553398163655
111	0.00862452745670671	0.0073792553398163655
112	0.0074396817157707 1	0.0073792553398163655
113	0.0074396817157707 1	0.0073792553398163655

Fitur	Kelas Positive	Kelas Negative
114	0.008303345384090605	0.0073792553398163655
115	0.008664048898941084	0.008429353575349461
116	0.00747869018154911	0.008537588664198199
117	0.0081519289465252	0.009187974207940408
118	0.008195581582801812	0.0073792553398163655
119	0.00743968171577071	0.0073792553398163655
120	0.00743968171577071	0.0073792553398163655
121	0.00743968171577071	0.0073792553398163655
122	0.00743968171577071	0.0073792553398163655
123	0.00743968171577071	0.007879008335727881
124	0.00743968171577071	0.00870332782933027
125	0.008094026545445198	0.009192500139861814
126	0.0078524947032407	0.0073792553398163655
127	0.008015823092112578	0.0073792553398163655