

**PENGARUH KERNEL *POLYNOMIAL* TERHADAP DATA
NON-LINEAR DALAM IDENTIFIKASI *HATESPEECH* BERBAHASA
INDONESIA PADA KOMENTAR INSTAGRAM MENGGUNAKAN
METODE *SUPPORT VECTOR MACHINE***

TUGAS AKHIR

Sebagai syarat untuk memperoleh gelar sarjana S-1
Program Studi Informatika, Jurusan Informatika, Fakultas Teknik Industri, Universitas
Pembangunan Nasional “Veteran” Yogyakarta”



Disusun oleh :
Farisa Yumna Puspitaningrum HP
123180103

**PROGRAM STUDI INFORMATIKA
JURUSAN INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”
YOGYAKARTA
2023**

HALAMAN PENGESAHAN PEMBIMBING

PENGARUH KERNEL *POLYNOMIAL* TERHADAP DATA *NON-LINEAR* DALAM IDENTIFIKASI *HATESPEECH* BERBAHASA INDONESIA PADA KOMENTAR INSTAGRAM MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

Disusun oleh:

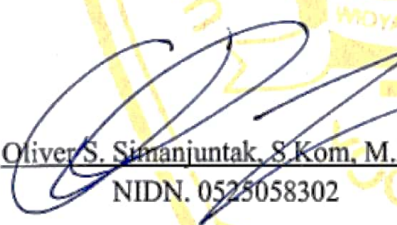
Farisa Yumna Puspitaningrum HP


123180103

Telah diuji dan dinyatakan lulus oleh pembimbing
pada tanggal: 20 September 2023

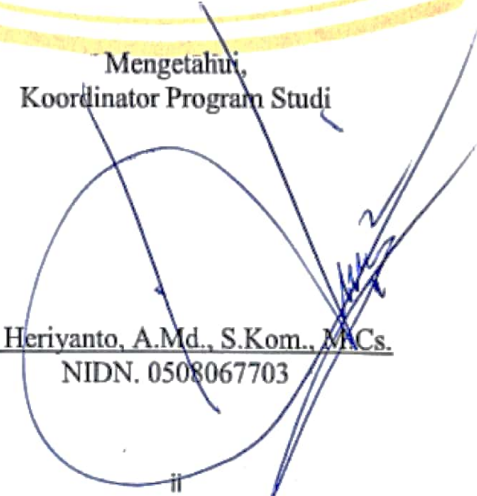
Menyetujui,
Pembimbing I

Pembimbing II


Oliver S. Simanjuntak, S.Kom, M.Eng.
NIDN. 0525058302


Mangaras Yanu F., S.T, M.Eng.
NIDN. 0521018201

Mengetahui,
Koordinator Program Studi


Dr. Heriyanto, A.Md., S.Kom., M.Cs.
NIDN. 0508067703

HALAMAN PENGESAHAN PENGUJI

PENGARUH KERNEL *POLYNOMIAL* TERHADAP DATA *NON-LINEAR* DALAM IDENTIFIKASI *HATESPEECH* BERBAHASA INDONESIA PADA KOMENTAR INSTAGRAM MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

Disusun oleh:

Farisa Yumna Puspitaningrum HP

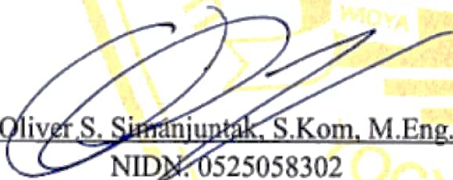
123180103


Telah diuji dan dinyatakan lulus
pada tanggal: 20 September 2023

Menyetujui,

Penguji I

Penguji II



Oliver S. Simanjuntak, S.Kom, M.Eng.
NIDN. 0525058302


Mangaras Yanu F., S.T., M.Eng.
NIDN. 0521018201

Penguji III

Penguji IV


Budi Santosa, S.Si., M.T.
NIDN. 0510097001


Wilis Kaswidjanti, S.Si., M.Kom.
NIDN. 0513047601

PERNYATAAN KARYA ASLI TUGAS AKHIR

Sebagai mahasiswa Program Studi Informatika Fakultas Teknologi Industri Universitas Pembangunan Nasional "Veteran" Yogyakarta, yang bertanda tangan dibawah ini, saya :

Nama : Farisa Yumna Puspitaningrum HP

NIM : 123180103

Menyatakan bahwa karya ilmiah saya yang berjudul :

PENGARUH KERNEL *POLYNOMIAL* TERHADAP DATA *NON-LINEAR* DALAM IDENTIFIKASI *HATESPEECH* BERBAHASA INDONESIA MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

merupakan karya asli saya dan belum pernah dipublikasikan dimanapun. Apabila di kemudian hari, karya saya disinyalir bukan merupakan karya asli saya, maka saya bersedia menerima konsekuensi apa pun yang diberikan Program Studi Informatika Fakultas Teknologi Industri Universitas Pembangunan Nasional "Veteran" Yogyakarta kepada saya.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Yogyakarta

Pada tanggal : 29 September 2023

Yang menyatakan



Farisa Yumna Puspitaningrum HP
NIM. 123180103

PERNYATAAN BEBAS PLAGIASI

Saya yang bertanda tangan di bawah ini,

Nama : Farisa Yumna Puspitaningrum HP
NIM : 123180103
Fakultas/Prodi : Teknik Industri/Informatika

dengan ini saya menyatakan bahwa judul Tugas Akhir

PENGARUH KERNEL *POLYNOMIAL* TERHADAP DATA *NON-LINEAR* DALAM IDENTIFIKASI *HATESPEECH* BERBAHASA INDONESIA MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

adalah hasil kerja keras saya sendiri dan benar bebas dari plagiasi kecuali cuplikan serta ringkasan yang terdapat didalamnya telah saya jelaskan sumbernya (Sitasi) dengan jelas. Apabila pernyataan ini terbukti tidak benar maka saya akan bersedia menerima sanksi sesuai peraturan Mendiknas RI No 17 tahun 2010 dan Peraturan Perundang-undangan yang berlaku.

Demikian surat pernyataan ini saya buat dengan penuh tanggung jawab.

Dibuat di : Yogyakarta
Pada tanggal : 29 September 2023

Yang menyatakan



Farisa Yumna Puspitaningrum HP
NIM. 123180103

ABSTRAK

Support Vector Machine (SVM) merupakan metode pembelajaran mesin *supervised learning* yang telah terbukti sebagai salah satu algoritma pembelajaran yang paling kuat untuk kategorisasi teks, tetapi karena pada prinsipnya bekerja secara *linear* sehingga untuk mengatasi masalah data *non-linear* dikembangkan fungsi kernel. Terdapat beberapa kernel *non-linear* salah satunya seperti kernel *polynomial*. Kernel *polynomial* memiliki parameter *degree* yang dapat disesuaikan sehingga mendapatkan hasil yang optimal. Penelitian ini melakukan analisis pengaruh kernel *polynomial* terhadap data *non-linear* yang diimplementasikan dalam identifikasi *hatespeech* berbahasa Indonesia pada komentar Instagram. Untuk mengetahui data yang dipakai berbentuk *linear* atau *non-linear*, maka digunakan teknik reduksi *Principal Component Analysis* (PCA) untuk memvisualisasikan data. Kernel yang digunakan yaitu kernel *polynomial*, *linear*, dan RBF.

Hasil pengujian menunjukkan bahwa metode *Support Vector Machine* menggunakan kernel *polynomial* memiliki performa yang lebih baik dibandingkan kernel *linear* dan kernel RBF dalam melakukan identifikasi data *non-linear* berjumlah 918 komentar yang terdiri dari 463 *non-hatespeech* dan 455 *hatespeech* dengan proses *split* data *training* dan *testing* 80:20, serta melalui tujuh proses *preprocessing* yaitu *case folding*, *cleaning*, *remove repetition character*, *tokenizing*, *normalisasi*, *stemming*, dan *stopword removal*. Selain itu, penentuan nilai parameter *C*, *degree*, dan *gamma* juga memiliki pengaruh dalam meningkatkan performa pada model. Performa terbaik kernel *polynomial* diperoleh dengan nilai parameter *C*=0.1, *degree*=1, dan *gamma*=0.01 yaitu akurasi 83.15%, presisi 83.45% dan *recall* sebesar 83.15%. Sedangkan kernel RBF diperoleh performa terbaik pada nilai parameter *C*=2 dan *gamma*=0.001 yaitu akurasi 80.43%, presisi 80.51%, *recall* 80.43%, dan kernel *linear* performa terbaik didapatkan dengan nilai parameter *C*=0.01 yaitu akurasi, presisi, dan *recall* sebesar 77.17%.

Kata kunci: analisis sentimen, ujaran kebencian, Instagram, data *non-linear*, kernel *polynomial*, *Support Vector Machine*, *Principal Component Analysis*

ABSTRACT

Support Vector Machine (SVM) is a supervised learning machine learning method that has proven to be one of the most powerful learning algorithms for text categorization, but because it works linearly in principle so to overcome the problem of non-linear data, a kernel function was developed. There are several non-linear kernels, one of which is the polynomial kernel. The polynomial kernel has degree parameters that can be adjusted to get optimal results. This research analyzes the effect of the polynomial kernel on non-linear data implemented in the identification of Indonesian-language hatespeech on Instagram comments. To find out the data used is linear or non-linear, the Principal Component Analysis (PCA) reduction technique is used to visualize the data. The kernels used are polynomial, linear, and RBF kernels.

The test results show that the Support Vector Machine method using the polynomial kernel has better performance than the linear kernel and RBF kernel in identifying non-linear data totaling 918 comments consisting of 463 non-hatespeech and 455 hatespeech with a split training and testing data process of 80:20, and through seven preprocessing processes namely case folding, cleaning, removing repetition characters, tokenizing, normalizing, stemming, and stopword removal. In addition, determining the value of the parameters C , degree, and gamma also has an influence in improving the performance of the model. The best performance of the polynomial kernel is obtained with a parameter value of $C = 0.1$, degree = 1, and gamma = 0.01, namely 83.15% accuracy, 83.45% precision and 83.15% recall. While the RBF kernel obtained the best performance at parameter value $C = 2$ and gamma = 0.001, namely accuracy 80.43%, precision 80.51%, recall 80.43%, and linear kernel the best performance is obtained with parameter value $C = 0.01$, namely accuracy, precision, and recall of 77.17%.

Keywords: sentiment analysis, hatespeech, Instagram, non-linear data, kernel polynomial, Support Vector Machine, Principal Component Analysis

KATA PENGANTAR

Puji syukur atas kehadiran Allah SWT, yang telah memberikan rahmat dan hidayah-Nya dalam memberikan kemudahan serta kekuatan sehingga tugas akhir dengan judul “Pengaruh Kernel *Polynomial* Terhadap Data *Non-Linear* Dalam Identifikasi *Hatespeech* Berbahasa Indonesia Pada Komentar Instagram Menggunakan Metode *Support Vector Machine*” dapat terselesaikan dengan baik. Tugas akhir ini merupakan salah satu syarat yang harus ditempuh untuk menyelesaikan Pendidikan pada jenjang Sastra Satu (S1) pada Program Studi Informatika Universitas Pembangunan Nasional “Veteran” Yogyakarta.

Tugas akhir ini dapat diselesaikan atas dukungan, bimbingan dan bantuan dari semua pihak baik secara lahir dan batin. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada :

1. Allah SWT yang telah memberikan kesehatan, petunjuk, dan kelancaran selama menyelesaikan jenjang Pendidikan S1 ini.
2. Kedua orang tua, Bapak dan Ibu serta keluarga yang selalu memberikan doa dan semangat tanpa henti
3. Bapak Oliver Samuel Simanjuntak, S.Kom., M.Eng., selaku dosen pembimbing I dan Bapak Mangaras Yanu Florestiyanto, S.T., M.Eng selaku dosen pembimbing II yang telah memberikan bimbingan, masukan dan arahan dalam menyusun tugas akhir ini.
4. Bapak Budi Santosa, S.Si., M.T. dan Ibu Wilis Kaswidjanti, S.Si, M.Kom, selaku dosen penguji yang telah memberikan saran dan masukan dalam menyelesaikan tugas akhir ini.
5. Seluruh dosen dan staff Program Studi Informatika Universitas Pembangunan Nasional “Veteran” Yogyakarta atas segala bantuan yang telah diberikan.
6. Teman-teman kelas C terutama Siti, Atan, Alya, Ina, teman-teman IF 2018, serta kakak tingkat yang tidak dapat disebutkan satu persatu atas dukungan, doa, dan bantuan yang diberikan.
7. Treasure yang telah menjadi hiburan disaat rasa lelah muncul selama menyelesaikan tugas akhir ini, terutama Jihoon yang telah memberikan kata-kata penenang dan penyemangat.

Tugas akhir ini masih jauh dari kata sempurna, oleh karena itu penulis mengharapkan adanya saran, tanggapan, dan kritik yang bersifat membangun dalam upaya pembelajaran lebih lanjut. Semoga tugas akhir ini dapat bermanfaat bagi semua pihak, terutama bagi penulis sendiri.

Yogyakarta, 29 September 2023

Penulis

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PENGESAHAN PEMBIMBING	ii
HALAMAN PENGESAHAN PENGUJI.....	iii
PERNYATAAN KARYA ASLI TUGAS AKHIR	iv
PERNYATAAN BEBAS PLAGIASI	v
ABSTRAK	vi
ABSTRACT	vii
KATA PENGANTAR.....	viii
DAFTAR ISI	ix
DAFTAR TABEL	xi
DAFTAR GAMBAR.....	xiii
DAFTAR MODUL PROGRAM.....	xiv
BAB I.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan Penelitian.....	2
1.5 Manfaat Penelitian.....	3
1.6 Metode Penelitian	3
1.6.1 Metodologi Penelitian	3
1.6.2 Pengembangan Sistem.....	3
1.7 Sistematika Penelitian	3
BAB II	5
2.1 Analisis Sentimen.....	5
2.2 <i>Hatespeech</i>	5
2.3 Instagram	5
2.4 <i>Text mining</i>	5
2.5 <i>Preprocessing</i>	6
2.6 <i>Term Frequency Inverse Document Frequency (TF-IDF)</i>	7
2.7 <i>Principal Component Analysis (PCA)</i>	7
2.8 <i>Support Vector Machine (SVM)</i>	8
2.9 <i>Confusion matrix</i>	11
2.10 <i>State of the art</i>	12

BAB III.....	17
3.1 Metodologi Penelitian	17
3.1.1 Analisa Masalah	17
3.1.2 Studi Literatur.....	18
3.1.3 Pengambilan Data.....	18
3.1.4 Pelabelan Data	18
3.1.5 <i>Preprocessing</i>	18
3.1.6 <i>Term Frequency Inverse Document Frequency (TF-IDF)</i>	24
3.1.7 <i>Principal Component Analysis (PCA)</i>	26
3.1.8 Pembuatan Model.....	31
3.1.9 Pengujian Model.....	42
3.2 Metode Pengembangan Sistem.....	42
BAB IV	50
4.1 Hasil Penelitian.....	50
4.1.1 Pengambilan Data.....	50
4.1.2 Preprocessing.....	51
4.1.3 Split Data	52
4.1.4 Pembobotan Kata TF-IDF	53
4.1.5 Pembuatan Visualisasi Principal Component Analysis.....	53
4.1.6 Pembuatan Model	54
4.1.7 Pengujian Model.....	54
4.1.8 Implementasi Sistem	57
4.1.9 Pengujian Sistem	60
4.2 Pembahasan	60
BAB V	63
5.1 Kesimpulan.....	63
5.2 Saran	63
DAFTAR PUSTAKA.....	64
LAMPIRAN	68

DAFTAR TABEL

Tabel 2.1 <i>Confusion Matrix</i>	11
Tabel 2.2 <i>State of The Art</i>	14
Tabel 2.3 <i>State of The Art</i> (Lanjutan).....	15
Tabel 2.4 <i>State of The Art</i> (Lanjutan).....	16
Tabel 3.1 Hasil Label Manual	18
Tabel 3.2 Hasil <i>Case Folding</i>	19
Tabel 3.3 Hasil <i>Cleaning</i>	20
Tabel 3.4 Hasil <i>Remove Repetition Character</i>	20
Tabel 3.5 Hasil <i>Tokenizing</i>	21
Tabel 3.6 Hasil Normalisasi	22
Tabel 3.7 Hasil <i>Stemming</i>	23
Tabel 3.8 Hasil <i>Stopword Removal</i>	23
Tabel 3.9 Data Hasil <i>Preprocessing</i>	24
Tabel 3.10 Contoh Perhitungan TF	25
Tabel 3.11 Contoh Perhitungan TF (Lanjutan)	26
Tabel 3.12 Contoh Hasil TF-IDF	26
Tabel 3.13 Contoh Hasil Perhitungan <i>Mean (x)</i>	27
Tabel 3.14 Contoh Hasil Normalisasi Data	28
Tabel 3.15 Contoh Hasil Perhitungan <i>Covarian Matrix</i>	28
Tabel 3.16 Contoh Hasil Perhitungan <i>Eigenvalue</i>	28
Tabel 3.17 Contoh Hasil Perhitungan <i>Eigenvalue</i> (Lanjutan).....	29
Tabel 3.18 Contoh Hasil Perhitungan <i>Eigenvector</i>	29
Tabel 3.19 Contoh Hasil <i>Eigenvector</i> yang telah diurutkan	29
Tabel 3.20 Contoh Hasil <i>Eigenvector n_component = 2</i>	30
Tabel 3.21 Matriks K Kernel <i>Linear</i>	32
Tabel 3.22 Matriks Hessian Kernel <i>Linear</i>	32
Tabel 3.23 Hasil Perhitungan Nilai <i>Error Kernel Linear</i>	32
Tabel 3.24 Hasil Perhitungan δai Kernel <i>Linear</i>	32
Tabel 3.25 Hasil Perhitungan ai Kernel <i>Linear</i>	33
Tabel 3.26 Hasil Iterasi Perhitungan Nilai <i>Error Kernel Linear</i>	33
Tabel 3.27 Hasil Iterasi Perhitungan δai Kernel <i>Linear</i>	33
Tabel 3.28 Hasil Iterasi Perhitungan ai Kernel <i>Linear</i>	33
Tabel 3.29 Hasil Kernelisasi Seluruh Data Uji Kernel <i>Linear</i>	34
Tabel 3.30 Matriks K Kernel <i>Polynomial</i>	35
Tabel 3.31 Matriks Hessian Kernel <i>Polynomial</i>	36
Tabel 3.32 Hasil Perhitungan Nilai <i>Error Kernel Polynomial</i>	36
Tabel 3.33 Hasil Perhitungan δai Kernel <i>Polynomial</i>	36
Tabel 3.34 Hasil Perhitungan ai Kernel <i>Polynomial</i>	36
Tabel 3.35 Hasil Iterasi Perhitungan Nilai <i>Error Kernel Polynomial</i>	37
Tabel 3.36 Hasil Iterasi Perhitungan δai Kernel <i>Polynomial</i>	37
Tabel 3.37 Hasil Iterasi Perhitungan ai Kernel <i>Polynomial</i>	37
Tabel 3.38 Hasil Kernelisasi Seluruh Data Uji Kernel <i>Polynomial</i>	38
Tabel 3.39 Matriks K Kernel RBF	39
Tabel 3.40 Matriks Hessian Kernel RBF	39
Tabel 3.41 Hasil Perhitungan Nilai <i>Error Kernel RBF</i>	39
Tabel 3.42 Hasil Perhitungan δai Kernel RBF.....	40
Tabel 3.43 Hasil Perhitungan ai Kernel RBF.....	40

Tabel 3.44 Hasil Iterasi Perhitungan Nilai <i>Error</i> Kernel RBF.....	40
Tabel 3.45 Hasil Iterasi Perhitungan <i>δai</i> Kernel RBF.....	40
Tabel 3.46 Hasil Iterasi Perhitungan <i>ai</i> Kernel RBF.....	40
Tabel 3.47 Hasil Kernelisasi Seluruh Data Uji Kernel RBF.....	41
Tabel 3.48 Pengujian Model.....	42
Tabel 3.49 Pengujian Actual Model SVM Kenel <i>Linear</i>	42
Tabel 3.50 Pengujian Actual Model SVM Kenel <i>Polynomial</i>	42
Tabel 3.51 Pengujian Actual Model SVM Kenel RBF	42
Tabel 3.52 Kebutuhan Non Fungsional.....	43
Tabel 3.53 Kebutuhan Fungsional.....	43
Tabel 3.54 Pengujian <i>Blackbox</i>	48
Tabel 3.54 Pengujian <i>Blackbox</i> (Lanjutan)	49
Tabel 4.1 Hasil Pengujian <i>Confussion Matrix</i> SVM Kernel <i>Linear</i>	55
Tabel 4.2 Hasil Pengujian Actual Model SVM Kernel <i>Linear</i>	55
Tabel 4.3 Hasil Pengujian <i>Confussion Matrix</i> SVM Kernel <i>Polynomial</i>	55
Tabel 4.4 Hasil Pengujian Actual Model SVM Kernel <i>Polynomial</i>	56
Tabel 4.5 Hasil Pengujian <i>Confussion Matrix</i> SVM Kernel RBF.....	56
Tabel 4.6 Hasil Pengujian Actual Model SVM Kernel RBF.....	57
Tabel 4.7 Hasil Pengujian Model	57
Tabel 4.8 Hasil Pengujian <i>Blackbox</i>	60
Tabel 4.9 Hasil Percobaan Nilai Parameter Pada Kernel <i>Polynomial</i>	61
Tabel 4.9 Hasil Percobaan Nilai Parameter Pada Kernel RBF.....	62
Tabel 4.11 Hasil Percobaan Nilai Parameter Pada Kernel <i>Linear</i>	62

DAFTAR GAMBAR

Gambar 2.1 (a) <i>hyperplane</i> data <i>linear</i> , (b) <i>hyperplane</i> data <i>nonlinear</i>	9
Gambar 3.1 Tahapan Metodologi Penelitian.....	17
Gambar 3.2 <i>Flowchart Scraping Data</i>	18
Gambar 3.3 <i>Flowchart Tahap Preprocessing</i>	19
Gambar 3.4 <i>Flowchart Case Folding</i>	19
Gambar 3.5 <i>Flowchart Cleaning</i>	20
Gambar 3.6 <i>Flowchart Remove Repetition Character</i>	21
Gambar 3.7 <i>Flowchart Tokenizing</i>	21
Gambar 3.8 <i>Flowchart Normalisasi</i>	22
Gambar 3.9 <i>Flowchart Stemming</i>	23
Gambar 3.10 <i>Flowchart Stopword Removal</i>	24
Gambar 3.11 <i>Flowchart TF-IDF</i>	25
Gambar 3.12 <i>Flowchart Principal Component Analysis</i>	27
Gambar 3.13 Grafik Contoh Hasil <i>Principal Component Analysis (PCA)</i>	30
Gambar 3.14 <i>Flowchart Model SVM Kernel Linear</i>	31
Gambar 3.15 <i>Flowchart Model SVM Kernel Polynomial</i>	35
Gambar 3.16 <i>Flowchart Model SVM Kernel RBF</i>	38
Gambar 3.17 Metode Prototyping	42
Gambar 3.18 Arsitektur Sistem	44
Gambar 3.19 DFD Level 0	44
Gambar 3.20 DFD Level 1	45
Gambar 3.21 Rancangan Halaman <i>Dashboard</i>	46
Gambar 3.22 Rancangan Halaman Dataset	46
Gambar 3.23 Rancangan Halaman <i>Preprocessing</i>	47
Gambar 3.24 Rancangan Halaman <i>Detection</i>	47
Gambar 3.25 Rancangan Halaman Statistika	48
Gambar 4.1 Visualisasi Data	54
Gambar 4.2 Halaman <i>Dashboard</i>	57
Gambar 4.3 Halaman Dataset.....	58
Gambar 4.4 Halaman <i>Preprocessing</i>	58
Gambar 4.5 Halaman <i>Detection</i>	59
Gambar 4.6 Halaman Statistika	60

DAFTAR MODUL PROGRAM

Modul Program 4.1 Pengambilan Data	50
Modul Program 4.2 <i>Case Folding</i>	51
Modul Program 4.3 <i>Cleaning</i>	51
Modul Program 4.4 <i>Remove Repetition Character</i>	51
Modul Program 4.5 <i>Tokenizing</i>	52
Modul Program 4.6 Normalisasi	52
Modul Program 4.7 <i>Stemming</i>	52
Modul Program 4.8 <i>Stopword Removal</i>	52
Modul Program 4.9 <i>Split Data</i>	53
Modul Program 4.10 TF-IDF	53
Modul Program 4.11 <i>Principal Component Analysis</i>	53
Modul Program 4.12 Pembuatan Model.....	54

BAB I

PENDAHULUAN

1.1 Latar Belakang

Support Vector Machine (SVM) merupakan metode pembelajaran mesin *supervised learning* yang dapat mengkorelasikan antar kata satu dengan yang lain, serta berusaha untuk menemukan pemisah kata terbaik dengan menggunakan fungsi *threshold*, sehingga tidak bergantung pada keberadaan jumlah fitur dalam data (Abro et al., 2020). *Support Vector Machine* telah terbukti sebagai salah satu algoritma pembelajaran yang paling kuat untuk kategorisasi teks (Rahat et al., 2020). Namun, karena *Support Vector Machine* pada prinsipnya bekerja secara *linear* sehingga *Support Vector Machine* memiliki kelemahan dalam data yang berbentuk *non-linear*. Untuk mengatasi masalah data yang berbentuk *non-linear* maka dikembangkan fungsi kernel (Pratama et al., 2018). Fungsi kernel atau disebut juga kernel *trick* akan mencari pemisah dengan cara dataset ditransformasikan ke ruang vektor yang berdimensi lebih tinggi sehingga dapat dipisahkan secara *linier* oleh *hyperplane* (Setiyono & Pardede, 2019). Menurut Karatzoglou et al. (2004) kernel yang sering digunakan yaitu kernel *linear*, *sigmoid*, *Radial Bias Function* (RBF), dan *polynomial* (Hilmiyah, 2017).

Penelitian yang menggunakan metode *Support Vector Machine* pernah dilakukan oleh Husni (2022). Penelitian ini menggunakan metode *Lexicon Based* dan *Support Vector Machine* dalam melakukan analisis sentimen mengenai pariwisata di Yogyakarta. Perpaduan antara metode *Lexicon Based* dan *Support Vector Machine* menghasilkan akurasi yang lebih baik yaitu sebesar 73.86%, sedangkan jika hanya menggunakan *Support Vector Machine* memperoleh akurasi 70.89%. Namun, kernel yang digunakan pada penelitian ini adalah kernel *linear* sedangkan data yang dipakai merupakan data *non-linear* sehingga menyebabkan *hyperplane linear* dari *Support Vector Machine* diperkirakan tidak mampu memisahkan data secara efektif karena *hyperplane linear* hanya bekerja pada data yang dapat dipisahkan secara *linear* (Drajana, 2017).

Penelitian lain yang menggunakan *Support Vector Machine* juga pernah dilakukan oleh Ikanovrianti (2021) dengan menambahkan metode *Query Expansion* (QE) dalam melakukan analisis sentimen pada komentar pendek tentang idol grup. Penelitian ini menghasilkan akurasi sebesar 74% dengan menggunakan kernel *linear* sebagai kernel *Support Vector Machine*. Namun karena pada penelitian ini menggunakan data *non-linear* sehingga dibutuhkan garis pemisah *non-linear*, yang mana dibandingkan menyesuaikan kurva *non-linear* ke data, *Support Vector Machine* menggunakan fungsi kernel untuk memetakan data ke dalam ruang berbeda dimana *hyperplane* dapat digunakan untuk partisi, salah satunya seperti kernel *polynomial* (Bhavsar & Panchal, 2012).

Penelitian Vogel & Meghana (2021) juga menggunakan *Support Vector Machine* dengan membandingkan metode Bi-LSTM dan *Support Vector Machine* untuk mendeteksi pengguna Twitter yang menyebarkan ujaran kebencian secara berulang. *Support Vector Machine* menghasilkan kinerja yang sedikit lebih baik dari Bi-LSTM yaitu akurasi keseluruhan masing-masing mencapai 64% untuk korpus Inggris dengan menggunakan kernel *linear* dan 75% pada korpus Spanyol menggunakan kernel RBF.

Kernel *polynomial* dalam melakukan analisis, memiliki parameter d atau *degree* yang dapat disesuaikan sehingga menghasilkan hasil yang optimal (Diani, 2017). Kernel *polynomial* juga merupakan kernel *non-linear* yang cocok untuk masalah data *training* yang dinormalisasi (Ningrum, 2018). Menurut Liang & Kusnadi (2021) untuk mendapatkan akurasi yang baik *Support Vector Machine* membutuhkan data yang dinormalisasi. Oleh karena itu, pada penelitian ini akan dilakukan analisis pengaruh kernel *polynomial* terhadap data *non-linear* dalam identifikasi *hatespeech* berbahasa Indonesia pada komentar Instagram. Untuk mengetahui data yang digunakan berbentuk *linear* atau *non-linear*, penelitian ini menggunakan *Principal Component Analysis* (PCA) untuk memvisualisasikan data. Selain itu, pada penelitian ini juga akan menerapkan normalisasi data pada tahap *preprocessing*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan di atas, maka rumusan masalah dalam penelitian ini yaitu :

1. Pengaruh kernel *polynomial* pada *Support Vector Machine* (SVM) dalam mengidentifikasi data *non-linear*.
2. Evaluasi performa metode *Support Vector Machine* (SVM) dalam mengidentifikasi *hatespeech* berbahasa Indonesia pada komentar Instagram menggunakan kernel *polynomial*, kernel *linear*, dan kernel RBF.

1.3 Batasan Masalah

Berdasarkan rumusan masalah tersebut, terdapat beberapa batasan masalah dalam penelitian ini yaitu :

1. Data yang digunakan adalah komentar Instagram dalam bentuk kalimat teks berbahasa Indonesia tahun 2019 s/d 2022 yang diambil dengan cara *scraping*
2. Pelabelan dilakukan secara manual
3. Jumlah data yang digunakan sebanyak 918 komentar
4. Pada penelitian ini mengidentifikasi komentar Instagram menjadi 2 kelas yaitu *hatespeech* dan *non-hate speech*
5. Kernel yang digunakan yaitu kernel *linear*, *polynomial*, dan RBF
6. Data yang digunakan adalah data *non-linear*

1.4 Tujuan Penelitian

Penelitian ini dilakukan dengan tujuan sebagai berikut:

1. Mengetahui pengaruh penggunaan kernel *trick polynomial Support Vector Machine* (SVM) dalam mengidentifikasi data *non-linear*
2. Mengetahui performa kernel *linear*, kernel *polynomial*, dan kernel RBF *Support Vector Machine* (SVM) dalam mengidentifikasi *hatespeech* berbahasa Indonesia pada komentar Instagram

1.5 Manfaat Penelitian

Dengan adanya penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Dapat mengetahui hasil akurasi, *presisi*, dan *recall* dalam mengidentifikasi *hate speech* Berbahasa Indonesia pada komentar Instagram menggunakan kernel *linear*, kernel *polynomial*, dan kernel RBF pada *Support Vector Machine* (SVM)
2. Dapat mempermudah *user* dalam mengidentifikasi komentar *hatespeech*
3. Dapat dijadikan referensi untuk penelitian selanjutnya yang akan menggunakan data *non-linear*

1.6 Metode Penelitian

Pada bagian ini akan dituliskan tahapan-tahapan penelitian serta metode yang digunakan pada sub bab – sub bab berikutnya.

1.6.1 Metodologi Penelitian

Tahapan-tahapan metodologi penelitian yang digunakan dalam penelitian ini yaitu sebagai berikut:

1. Analisa Masalah
2. Studi Literatur
3. Pengambilan data
4. Pelabelan data
5. Data *preprocessing*
6. Pembobotan kata (TF-IDF)
7. Visualisasi data (PCA)
8. Pembuatan Model
9. Pengujian Model

1.6.2 Pengembangan Sistem

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode pengembangan *prototyping*. Metode *prototyping* merupakan salah satu metode yang dapat memberikan pendekatan terbaik dan lebih umum untuk digunakan. Ketika kebutuhan yang dimiliki kurang jelas, metode ini juga dapat membantu seorang developer dan client untuk lebih memahami apa yang akan dibangun (Pressman, 2005). Beberapa tahapan dalam metode ini yaitu sebagai berikut:

1. *Communication*
2. *Quick plan and modeling quick design*
3. *Construction of prototype*
4. *Deployment delivery and feedback*

1.7 Sistematika Penelitian

Sistematika penulisan laporan penelitian ini adalah sebagai berikut :

BAB I PENDAHULUAN

Bab ini membahas tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

BAB II TINJAUAN LITERATUR

Bab tinjauan literatur ini berisi tentang landasan teori yang berkaitan dengan objek masalah, metode dan karya ilmiah yang sudah ada sebagai bahan referensi dan pondasi untuk memperkuat argumentasi dalam penelitian ini.

BAB III METODE PENELITIAN

Pada bab metode penelitian menjelaskan tentang tahapan-tahapan metode penelitian yang dilakukan pada penelitian ini.

BAB IV HASIL DAN PEMBAHASAN

Bab ini membahas mengenai hasil dan pembahasan dari pengujian dan implementasi bab sebelumnya.

BAB V KESIMPULAN DAN SARAN

Bab ini berisi tentang kesimpulan dan saran berdasarkan penelitian yang sudah dilakukan untuk penelitian selanjutnya agar lebih baik.

BAB II

TINJAUAN LITERATUR

2.1 Analisis Sentimen

Analisis sentimen (*opinion mining*) merupakan salah satu cabang dari *text mining* yang mulai meruak sekitar tahun 2003. Analisis sentimen adalah proses mengekstraksi opini atau komentar dari suatu teks mengenai topik tertentu (Kaur et al., 2017). Menurut Cutlip dan Center dalam Resphati (2006), opini dapat dianggap sebagai bentuk ekspresi dari suatu sikap yang mencerminkan pandangan atau pendapat seseorang terhadap suatu masalah atau persoalan tertentu. Penerapan analisis sentimen dapat digunakan pada berbagai opini di komentar media sosial, *web*, maupun *blog* seperti opini kebencian, opini umum, politik, film, merek produk, dan lain-lain (Cindo et al., 2019).

2.2 Hatespeech

Hatespeech adalah ujaran yang mengandung makna merendahkan seseorang atau kelompok dalam berbagai sudut pandang seperti agama, ras, suku, jenis kelamin, kebangsaan, orientasi seksual, warna kulit, dan lain-lain (Setiawan et al., 2021). *Hate speech* berisi ujaran tidak sopan dan tidak bijaksana yang bukan dimaksudkan untuk mengoreksi pendapat atau perilaku seseorang. Ujaran kebencian dapat menimbulkan permusuhan serta kebencian yang dapat merusak hubungan antar manusia, agama, ras dan suku serta dapat berdampak pada mental/psikologi seseorang.

2.3 Instagram

Instagram merupakan salah satu media sosial yang memungkinkan setiap pengguna dapat mengambil dan membagikan momen lewat foto/video secara cepat menggunakan koneksi internet. Serta memiliki fitur komentar yang dapat dikomentari oleh pengguna instagram lainnya difoto/video tersebut. Dilansir dari Wikipedia.com Instagram dirancang oleh Kevin Systrom dan Mike Krieger yang rilis pada tahun 2010. Tetapi, pada 9 april 2012 Instagram telah resmi diambil alih oleh Facebook. Instagram dapat digunakan di perangkat IOS, Android dan Windows 10 serta dapat diakses juga melalui *web* instagram.com. Instagram memiliki efektifitas paling tinggi dibandingkan dengan Facebook dan Twitter (Rachmat et al., 2019). Pengguna aktif Instagram di seluruh dunia pada April 2022 menurut data We Are Social yaitu mencapai 1,45 miliar orang, dengan Indonesia menduduki urutan ke-4 dibawah Amerika Serikat, Brasil, dan India.

2.4 Text mining

Text mining merupakan teknik yang dapat dilakukan untuk menangani masalah *clustering*, klasifikasi, *information retrival* dan *information extraction* (Manalu, 2014). Data yang digunakan adalah sekumpulan *text* yang memiliki format tidak terstruktur atau semi terstruktur. *Text mining* bertujuan untuk mengolah teks menjadi sebuah informasi yang berkualitas yang dilakukan dengan menemukan pola melalui pola *statistic*, pemodelan bahasa *statistic*, dan pemodelan topik (Han et al., 2011).

2.5 *Preprocessing*

Preprocessing merupakan proses awal pada *text mining* untuk mengubah data mentah yang merupakan data tidak struktur menjadi data terstruktur (Kurniawan et al., 2017) sehingga dapat memudahkan dalam melakukan prediksi/klasifikasi. Tujuan lain dari *preprocessing* juga untuk membersihkan/menghilangkan *noise* dan mengambil fitur atau parameter penting pada dokumen teks (Pardede et al., 2020) serta melakukan penyeragaman kata sehingga kata tersebut siap untuk diekstraksi ketahap berikutnya. Tahapan *preprocessing* pada penelitian ini yaitu:

a. *Case Folding*

Case folding adalah tahap *preprocessing* yang mengubah semua huruf pada data *input* menjadi huruf kecil (*lowercase*). *Case folding* bertujuan untuk menyeragamkan bentuk kata-kata pada data *input* sehingga mempermudah pencarian kata (Susilowati et al., 2015).

b. *Cleaning*

Cleaning merupakan tahap pembersihan *noise* pada data *input*. *Noise* yang dimaksud yaitu tanda baca, *link url*, angka, dan *hashtag* pada data komentar.

c. *Remove Repetition Character*

Remove Repetition Character dalam pemrosesan teks bertujuan untuk menghapus karakter berulang yang tidak memberikan informasi tambahan dalam teks. Menghapus repetisi karakter dari teks dapat membantu menyederhanakan teks dan mengurangi varian yang tidak perlu dalam representasi teks sehingga dapat meringankan beban pada *database* serta dapat menghemat waktu dalam melakukan klasifikasi.

d. *Tokenizing*

Tokenizing adalah memisahkan kalimat menjadi kata perkata/frasa sehingga membuatnya lebih mudah untuk dianalisa. Kata perkata tersebut disebut dengan token.

e. *Normalisasi*

Normalisasi merupakan proses mengubah kata yang tidak baku menjadi kata baku. Normalisasi memperbaiki kesalahan ejaan yang ada pada kata data *input* sehingga kata yang memiliki makna yang sama dapat menjadi setara (Hendriyanto et al., 2022).

f. *Stemming*

Stemming merupakan proses mengubah kata-kata berimbuhan menjadi sebuah kata dasar. Proses *stemming* bertujuan untuk meringankan beban pada *database* serta dapat menghemat waktu dalam melakukan klasifikasi. Jika sebuah imbuhan tidak dihilangkan maka akan banyak berbagai macam kata yang memiliki kata dasar yang sama tersimpan di *database*. Di dalam teks berbahasa Inggris, proses yang paling penting yaitu penghilangan *sufiks*. Sedangkan dalam teks berbahasa Indonesia kata imbuhan *prefiks* dan *sufiks* dihilangkan juga. *Prefiks* adalah imbuhan yang terletak didepan kata dasar. Sedangkan *sufiks* merupakan imbuhan yang terletak diakhir/dibelakang kata dasar. *Stemmer* sastrawi merupakan pengembangan dari *stemmer* nazief adriani. Aturan dalam *stemmer* sastrawi (Mustikasari et al., 2021) yaitu :

1. Dilakukan pemeriksaan kata yang akan di-*stemming* dalam kamus *stemming*. Jika kata tersebut ditemukan, maka proses akan dihentikan.

2. Jika kata tidak ditemukan maka kata tersebut merupakan kata imbuhan (*affix*), sehingga dihilangkan kata *sufiks* seperti –lah, -kah, -ku, -mu, -nya, -tah atau –pun.
3. Menghilangkan *afiks* turunan seperti –I, -ka, -an, kemudian menghapus be-, di-, ke-, me-, pe-, se-, dan te-.
4. Apabila hasil kata dasar dari langkah-langkah sebelumnya tidak ditemukan dalam kamus *stemming*, maka dilakukan pemeriksaan pada tabel ambigu.
5. Jika semua langkah telah dilakukan tetapi kata tetap tidak terdapat pada kamus, maka algoritma akan mengembalikan kata tersebut ke kata aslinya.

g. Stopword removal

Stopword removal adalah proses menghapus kata. Tujuannya untuk menghilangkan kata-kata yang tidak penting/berpengaruh terhadap kalimat (Pardede et al., 2020). Contoh *stopword* dalam bahasa Indonesia seperti “yang”, “di”, “dan”, “dari”, dan lain-lain.

2.6 Term Frequency Inverse Document Frequency (TF-IDF)

Pembobotan *Term Frequency Inverse Document Frequency* (TF-IDF) merupakan metode yang umum digunakan dalam menentukan hubungan kata (*term*) terhadap dokumen dengan memberikan bobot atau nilai pada masing-masing kata. *Term Frequency Inverse Document Frequency* merupakan gabungan dari konsep *Term Frequency* dan *Inverse Document Frequency* (Arifin et al., 2021). *Term Frequency* merupakan perhitungan bobot pada sebuah kata (*term*) berdasarkan jumlah kemunculan kata pada sebuah dokumen. Semakin besar nilai pembobotan *Term Frequency* di dalam suatu dokumen maka frekuensi kemunculan kata (*term*) semakin banyak. Sedangkan *Inverse Document Frequency* (IDF) merupakan perhitungan bobot berdasarkan frekuensi kemunculan kata pada dokumen lain. Semakin tinggi nilai *Inverse Document Frequency* pada suatu *term* maka semakin penting pula kata (*term*) tersebut karena tidak banyak digunakan di dokumen lain (Mandala, 2006).

Persamaan yang digunakan dalam perhitungan *Term Frequency Inverse Document Frequency* (TF-IDF) sebagai berikut:

$$W_t = TF_t \times IDF_t \dots \dots \dots (2.1)$$

$$W_t = (t, d) \times \log \frac{n}{df(t)} \dots \dots \dots (2.2)$$

Keterangan:

W_t : nilai *Term Frequency Inverse Document Frequency*

TF_t : nilai *Term Frequency* (TF)

IDF_t : nilai *Inverse Document Frequency* (IDF)

(t, d) : jumlah kemunculan term t pada dokumen d

n : jumlah dokumen secara keseluruhan

$df(t)$: jumlah dokumen yang mengandung term t

2.7 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) merupakan salah satu teknik reduksi dimensi yang dapat digunakan untuk visualisasi data. *Principal Component Analysis* (PCA) ditemukan oleh Karl Pearson pada tahun 1901. *Principal Component Analysis* dapat membantu mengurangi dimensi data yang tinggi menjadi dimensi yang rendah.

Perhitungan algoritma *Principal Component Analysis* (PCA) dirumuskan sebagai berikut (Hediyati & Suartana, 2021):

1. Mencari *mean* dari setiap *term*

$$\bar{x} = \frac{\sum x_i}{n} \dots \dots \dots (2.17)$$

2. Normalisasi data

$$X_{1new} = (x_i - \bar{x}) \dots \dots \dots (2.18)$$

3. Menghitung *covarian matrix*

$$C_x = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n-1} \dots \dots \dots (2.19)$$

4. Mencari *eigenvalue* dan *eigenvector*

$$C_x V_m = \lambda_m V_m \dots \dots \dots (2.20)$$

5. Mengurutkan *eigenvector* berdasarkan *eigenvalue* tertinggi

6. *Principal Components*

$$PC = X_{1new} \times V \dots \dots \dots (2.21)$$

Keterangan:

C_x : matriks kovarian

λ : nilai *Eigen Value*

\bar{x} : nilai rata-rata

PC : *Principal Component*

V : nilai *Eigen Vector*

2.8 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) merupakan salah satu metode *machine learning* yang pertama kali diperkenalkan oleh Vapnik pada tahun 1992. *Support Vector Machine* bekerja menggunakan prinsip *Structural Risk Minimization* (SRM) yang bertujuan untuk menemukan *hyperplane* terbaik dalam memisahkan dua buah *class* pada *input space* (Nugroho et al., 2003). *Hyperplane* merupakan sebuah bidang atau garis yang terdapat dalam ruang fitur. *Support Vector Machine* berusaha menemukan *hyperplane* yang memiliki *margin* terbesar, yaitu jarak terdekat antara data dari kedua kelas dengan *hyperplane* tersebut. *Support Vector Machine* telah terbukti sebagai salah satu algoritma pembelajaran yang paling kuat untuk kategorisasi teks (Rahat et al., 2020).

Karakteristik *Support Vector Machine* sebagaimana telah dijelaskan pada bagian sebelumnya, dirangkumkan sebagai berikut (Nugroho et al, 2003):

1. Secara prinsip *Support Vector Machine* adalah *linear classifier*.
2. *Pattern recognition* dilakukan dengan mentransformasikan data pada *input space* ke ruang yang berdimensi lebih tinggi, dan optimisasi dilakukan pada ruang *vector* yang baru tersebut. Hal ini membedakan *Support Vector Machine* dari solusi *pattern recognition* pada umumnya, yang melakukan optimisasi parameter pada ruang hasil transformasi yang berdimensi lebih rendah daripada dimensi *input space*.
3. Menerapkan strategi *Structural Risk Minimization* (SRM).
4. Prinsip kerja *Support Vector Machine* pada dasarnya hanya mampu menangani klasifikasi dua kelas.

Secara umum rumus yang digunakan pada *Support Vector Machine* pada proses klasifikasi data adalah :

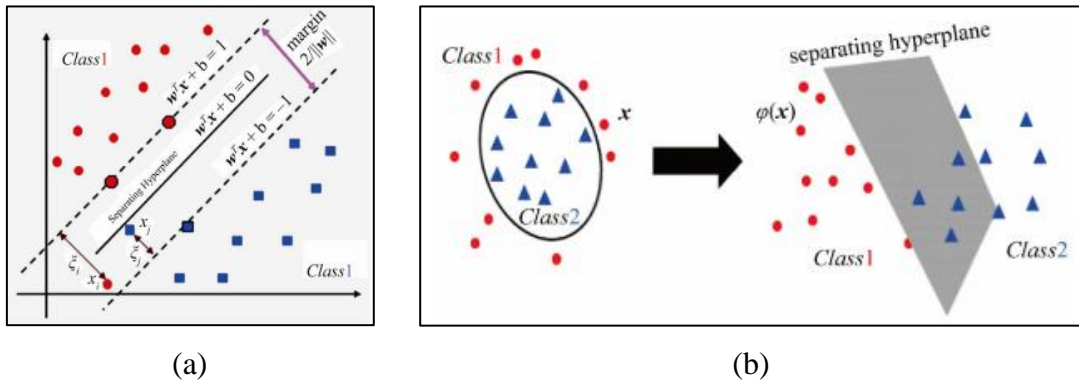
$$f(x) = w \cdot x + b \dots \dots \dots (2.3)$$

atau

$$f(x) = \sum_{i=1}^m a_i y_i K(x, x_i) + b \dots \dots \dots (2.4)$$

Keterangan :

- w : Parameter hyperlane yang dicari (garis yang tegak lurus antara garis hyperlane dan titik support vector)
- x : Titik data masukan *Support Vector Machine*
- a_i : Nilai bobot setiap titik data
- y_i : Kelas setiap data
- $K(x, x_i)$: Fungsi kernel
- b : Parameter *hyperlane* yang dicari (nilai bias)



Gambar 2.1 (a) *hyperplane* data linear, (b) *hyperplane* data nonlinear

Sumber : (Mahmoodi et al., 2011)

Dalam konteks pengklasifikasian data, permasalahan data *nonlinear* terjadi karena data tidak dapat dibagi secara linear menggunakan garis lurus (Sianturi et al., 2017). Untuk mengatasi masalah data *non-linear* maka dikembangkan fungsi kernel yang akan mentransformasikan data kedalam dimensi yang lebih tinggi. Kernel diperkenalkan oleh Aronszajn tahun 1950 (Nugroho et al., 2003). Fungsi kernel yang umum digunakan adalah sebagai berikut:

a. Kernel Linier

$$K(x_i, x_n) = x_i^T \cdot x_n \dots \dots \dots (2.5)$$

b. Kernel Polynomial

$$K(x_i, x) = (x_i^T x + 1)^d \dots \dots \dots (2.6)$$

c. Kernel Radial Basis Function (RBF)

$$K(x_i, x) = \exp \left(-\frac{1}{2\sigma^2} ||x_i - x||^2 \right) \dots \dots \dots (2.7)$$

d. Kernel Sigmoid

$$K(x_i, x) = \tanh(\alpha x_i^T \cdot x + \beta) \dots \dots \dots (2.8)$$

Pada penelitian ini digunakan algoritma *sequential learning* untuk memproses data latih dari *Support Vector Machine* yang dikenal sebagai algoritma sederhana dan menggunakan waktu yang singkat dibandingkan dengan algoritma lainnya (Vijayakumar, 1999). Langkah langkah untuk pelatihan data dapat dilihat sebagai berikut :

1. Melakukan perhitungan kernel dan inisialisasi terhadap parameter yang akan digunakan

γ : *Gamma / Learning rate*, yang berfungsi untuk mengontrol kecepatan

a_i : Alfa, yang berfungsi untuk mencari *support vector*

C : Variabel *slack*

d : *degree*, berfungsi untuk mengontrol fleksibilitas

2. Menghitung matriks Hessian

$$D_{ij} = y_i y_j (K(x_i x_j)) + \lambda^2 \dots \dots \dots (2.9)$$

Keterangan :

$K(x_i, x_j)$: Fungsi kernel

x_i : Data ke -i

x_j : Data ke-j

D_{ij} : Elemen matriks ke-ij

y_i : Kelas data ke-i

y_j : Kelas data ke-j

λ^2 : Batas teoritis yang diturunkan

3. Menghitung nilai error

$$E_i = \sum_{j=1}^n a_j D_{ij} \dots \dots \dots (2.10)$$

Keterangan :

E_i : Nilai error

4. Menghitung delta a_i

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

Keterangan :

δa_i : Delta a_i

5. Menghitung a_i baru

$$a_i \text{ baru} = a_i + \delta a_i \dots \dots \dots (2.12)$$

Ulangi iterasi pada langkah 3 sampai 5 hingga mencapai batas maksimum iterasi yang telah ditentukan.

6. Menghitung nilai $w \cdot x^+$ dan $w \cdot x^-$ untuk mendapatkan nilai bias

$$w \cdot x^+ = a_i Y_i K(w \cdot x^+) \dots \dots \dots (2.13)$$

$$w \cdot x^- = a_i Y_i K(w \cdot x^-) \dots \dots \dots (2.14)$$

$$b = -\frac{1}{2}(w \cdot x^+ + w \cdot x^-) \dots \dots \dots (2.15)$$

Keterangan :

$w \cdot x^+$: Nilai kernel data dengan kelas positif yang memiliki nilai a tertinggi

$w \cdot x^-$: Nilai kernel data dengan kelas negatif yang memiliki nilai a tertinggi

- b : Nilai bias
7. Menghitung nilai keputusan
 $f(x) = \sum_{i=0}^m \text{sign}(a_i Y_i K(x, x_i) + b) \dots\dots\dots (2.16)$
 Keterangan :
 $f(x)$: Fungsi untuk mendapatkan nilai keputusan

2.9 Confusion matrix

Pengujian dilakukan dengan menggunakan teknik *confusion matrix*. *Confusion matrix* bertujuan untuk menilai kinerja dari pengklasifikasian yang dibangun. *Performance* dari klasifikasi dievaluasi dengan menghitung *true positives* (TP), *false positives* (FP), *true negatives* (TN), dan *false negatives* (FN) (Abro et al., 2020). Tabel *confusion matrix* dapat dilihat sebagai berikut:

Tabel 2.1 Confusion Matrix

	Predicted Yes	Predicted No
Actual Positive	TP	FN
Actual Negatif	FP	TN

Keterangan:

True Positive (TP) = data *non-hatespeech* yang terdeteksi dengan benar sebagai data *non-hatespeech*

False Positive (FP) = data *hatespeech* tetapi terdeteksi sebagai data *non-hatespeech*

True Negative (TN) = data *hatespeech* yang terdeteksi dengan benar sebagai data *hatespeech*

False Negative (FN) = data *non-hatespeech* tetapi terdeteksi sebagai data *hatespeech*

Pengujian pada penelitian ini akan menghitung nilai akurasi, presisi, dan *recall*.

a. Akurasi

Akurasi digunakan untuk menghitung tingkat ketepatan antara nilai prediksi dan nilai sebenarnya yaitu membandingkan jumlah seluruh data yang terdeteksi dengan benar dengan jumlah data keseluruhan.

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

b. Presisi

Presisi bertujuan untuk mengukur sejauh mana model dapat mengidentifikasi data positif dengan benar dari semua data yang terdeteksi sebagai data positif.

$$Precision = \frac{TP}{TP + FP}$$

c. Recall

Recall berfungsi untuk menghitung rasio keberhasilan suatu sistem dalam menemukan sebuah informasi. Mengukur seberapa baik model dalam mengidentifikasi data positif yang berhasil ditemukan oleh model secara benar

$$Recall = \frac{TP}{TP + FN}$$

2.10 State of the art

Penelitian mengenai *hate speech* pernah dilakukan oleh Liang & Kusnadi (2021) dengan membandingkan antara metode *Support Vector Machine* (SVM), XGBoost, dan *Neural Network* dalam mengklasifikasikan ujaran kebencian di Twitter. Data diambil dari sentimen publik Twitter dalam Bahasa Indonesia sebanyak 18793 data dengan distribusi kelas 11282 data *non-hatespeech* dan 7511 data *hatespeech*. *Support Vector Machine* menggunakan kernel *linear*, *gamma scale*, nilai parameter regulasi sebesar 1.0, dan nilai *degree* sebesar 3 memiliki akurasi tertinggi yaitu 83.2% dibandingkan dengan metode XGBoost yang memiliki akurasi 79.6% dan *Neural Network* sebesar 82.9%. Tetapi *Support Vector Machine* memiliki *Error Rate* sekitar 16.8% karena dalam proses prediksi, algoritma mengalami kesulitan untuk membedakan antara atribut berpengaruh dan tidak berpengaruh. Untuk mendapatkan akurasi yang lebih baik, penelitian ini menyarankan agar algoritma *Support Vector Machine* menggunakan data yang dinormalisasi.

Penelitian Ivan et al. (2019) juga melakukan klasifikasi *hatespeech* berbahasa Indonesia di Twitter. Penelitian ini menggunakan metode *Naive Bayes* dan seleksi fitur *Information Gain* dengan menerapkan normalisasi kata dalam proses *preprocessing* yang diambil dari Pujangga Indonesian *Natural Language Processing* REST API. Dataset berjumlah 250 data dengan perbandingan data latih sebesar 80% dan data uji 20%. Total *training* data yang digunakan sebanyak 100 data *hate speech* dan 100 data *non hate speech*. Sedangkan untuk *testing* data berjumlah 25 data. *Threshold* yang digunakan sebesar 20%, 40%, 60%, 80%, dan 90%. Hasil akurasi menggunakan normalisasi kata dan seleksi fitur *Information Gain* dengan *threshold* 80% mengalami peningkatan sebesar 4% yaitu dari 94% menjadi 98% dengan nilai *precision* 100%, nilai *recall* 96,15%, dan nilai *F-measure* 98,03%.

Penelitian Rahman et al. (2021) melakukan klasifikasi ujaran kebencian menjadi lima kelas berdasarkan suku, agama, ras, antar golongan, dan netral dengan menggunakan metode *Support Vector Machine*. Dataset yang digunakan sebanyak 1000 data yang diambil dari Twitter dengan cara *crawling* data. Dalam penelitian ini juga dilakukan pengujian dengan membandingkan tiga buah kernel yang umum dipakai yaitu *linear*, *sigmoid*, dan RBF dengan komposisi data latih sebanyak 700 data, data uji 300 data dan menggunakan nilai $C=1$. Kernel RBF menghasilkan nilai akurasi paling tinggi yaitu 93%, nilai *precision* 84%, nilai *recall* 86%, dan *F-measure* 83%. Sedangkan kernel *linear* menghasilkan nilai akurasi sebesar 92%, nilai *precision* 85%, nilai *recall* 88%, dan *F-measure* 85%. Kernel *sigmoid* menghasilkan nilai akurasi 92%, nilai *precision* 90%, nilai *recall* 89%, dan *F-measure* 87%.

Penelitian Lyrawati (2019) melakukan deteksi ujaran kebencian pada data twitter menjelang pemilihan presiden 2019 menggunakan *Support Vector Machine* (SVM) dengan kernel RBF. Data diambil dari *Tweepy* API yang mengandung kata "#pilpres2019", "#2019gantipresiden", "#debatcapres", "debatpilpres" dan "#jokowi2periode" sebanyak 597 dengan distribusi kelas 320 HS dan 277 Non_HS. Parameter kernel yang digunakan yaitu C dengan nilai 1, 5, dan 10. Serta *epsilon* (ϵ) dengan nilai 10^{-12} , 10^{-9} dan 10^{-6} . Akurasi tertinggi dicapai sebesar 61.667% dengan waktu 1.21 menggunakan nilai *epsilon* dan $C = 1$

Penelitian Fu et al. (2010) membandingkan metode *Mixing Linear SVM*, *Non-linear SVM*, dan *Linear SVM* untuk klasifikasikan kulit. Kernel yang digunakan pada *non-linear SVM* yaitu kernel RBF. Pada penelitian ini menggunakan data yang diambil dari data *U.S. Geological Survey* oleh *the Colorado State University* dengan jumlah 581.012 data. Setiap *instance* memiliki 54 atribut fitur. *Support Vector Machine* menggunakan kernel RBF menghasilkan akurasi tertinggi yaitu 95.45% dengan waktu 76.25 detik, *Mixing Linear SVM* menghasilkan akurasi 95.20% dengan waktu 6.55 detik, dan *Linear SVM* menghasilkan akurasi 91.34% dengan waktu tercepat yaitu 1.65 detik. Dari hasil pengujian dapat dilihat bahwa *Mixing Linear SVM* jauh lebih efisien, menghasilkan kinerja yang sebanding dengan *Non-linear SVM*. Disisi lain *Linear SVM* cukup efisien untuk pengujian tetapi memiliki tingkat akurasi yang lebih rendah dibandingkan dengan *non-linear SVM* dan pendekatan *Mixing Linear SVM*.

Penelitian Priyambodo et al. (2022) melakukan klasifikasikan kematangan tanaman hidroponik pakcoy menggunakan metode *Support Vector Machine* dengan kernel RBF, *hyperparameter C* bernilai 10, dan *gamma* 0.0001. Data yang digunakan merupakan dataset *non-linear* sejumlah 200 gambar *Red*, *Green*, dan *Blue* (RGB) dengan kategori “besar” untuk Pakcoy yang sudah matang sebanyak 100 gambar dan kategori “kecil” untuk yang belum matang sebanyak 100 gambar. Citra diambil secara langsung menggunakan kamera. Hasil pengujian menggunakan metode *Support Vector Machine* memperoleh akurasi sebesar 100% , presisi 100%, *recall* 100% serta *f1-score* 100% pada proses *training* dalam mengklasifikasi kematangan tanaman pakcoy dan 79% pada proses evaluasi model terhadap citra acak diluar dataset yang digunakan.

Olive et al. (2020) mengimplementasikan *text mining* untuk menganalisis opini masyarakat terhadap layanan Grab dan Gojek menggunakan algoritma *Naïve Bayes Classifier* kedalam dua kelas yaitu sentimen positif dan negatif. Pada penelitian ini juga menggunakan *Principal Component Analysis* (PCA) untuk menentukan faktor dari setiap sentiment yang telah terklasifikasi. Data yang digunakan diambil dari Twitter API dengan kata kunci “grab” dan “gojek”. Grab sebanyak 597 *tweet* dengan distribusi kelas 260 *tweet* positif dan 337 *tweet* negatif. Sedangkan Gojek sebanyak 2249 *tweet* dengan distribusi kelas 959 *tweet* positif dan 1290 *tweet* negatif. Hasil pengujian menggunakan *threshold* sebesar 1.1 didapatkan akurasi Grab 74.34% dan Gojek 68,84%. Analisis faktor menggunakan *Principal Component Analysis* pada Grab menghasilkan 6 faktor positif dan 5 faktor negatif, sedangkan pada Gojek diperoleh 8 faktor positif dan 6 faktor negatif.

Mantik et al. (2022) menerapkan cabang dari data mining yaitu *Music Information Retrieval* (MIR) dengan melakukan klasifikasi label gamelan rindik berdasarkan suasana hati menggunakan metode *K-Nearest Neighbor*. Data yang digunakan adalah data *non-linear* yaitu potongan file musik berformat mono .wav dengan durasi 20 detik sebanyak 140 file yang diperoleh dari gamelan rindik yang ada di empat wilayah Denpasar yaitu Sanur, Tente, Pedungan dan Paguyangan kangin serta dilabeli langsung oleh para ahli menjadi 2 label yaitu 70 file *contentment* (tenang) dan 70 file *exuberance* (senang). Hasil dari pengukuran evaluasi pada penelitian ini menggunakan nilai k dari 1 sampai 15 dan diperoleh nilai k terbaik pada k=7 sehingga menghasilkan akurasi sebesar 81%, presisi 76%, *recall* 90.5%, dan *f-score* 82.6% dengan waktu tercepat 0.0258 detik.

Pusean et al. (2023) menggunakan metode *Naïve Bayes* dan *Support Vector Machine* untuk melakukan analisis sentimen mengenai acara televisi X Factor Indonesia. selain menggunakan dua metode tersebut, penelitian ini juga menggunakan dua skenario *preprocessing*. Skenario *preprocessing* pertama meliputi *case folding*, *removing emojis*, *stopwords removal*, *stemming*, dan *tokenization*. Skenario *preprocessing* kedua meliputi *case folding*, *removing emoji*, *cleansing*, *removing repetition characters*, *normalizing words*, *negation handling*, *stopwords removal*, *stemming*, dan *tokenization*. Data yang digunakan merupakan data *non-linear* yaitu sebanyak 900 data tweet yang diambil dengan keyword “XFactorID” pada bulan juli 2021 sd 2022. Data diklasifikasikan kedalam 3 kelas yaitu positif, negatif, dan netral. *Support Vector Machine* dengan skenario kedua menggunakan kernel *linear* menghasilkan akurasi terbaik dibandingkan *Naïve Bayes* yaitu sebesar 79.44%, nilai *presisi* 76.91%, nilai *recall* 76.94%, dan nilai *F1-score* 76.83%. Hasil dari skenario pertama dengan menggunakan metode *Super Vector Machine* memperoleh akurasi sebesar 74.44%.

Penelitian Tanjum (2018) membandingkan metode klasifikasi *non-linear* untuk mengetahui metode terbaik dalam mengklasifikasikan data *Balance Weight* dan *Distance*. Metode yang digunakan yaitu *K-Nearest Neighbor*, *Super Vector Machine*, *Naïve Bayes*, *Random Forest* dan *Classification and Regression Tree* (CART). Data yang digunakan sebanyak 625 data, dengan persebaran klasifikasi data 49 data *balance* (B), 288 data *right* (R), dan 288 data *left* (L). Metode terbaik diperoleh oleh metode *Support Vector Machine* yang menghasilkan akurasi sebesar 92% dan presisi 95.58% dimana *Support Vector Machine* dapat menangkap semua kategori pada klasifikasi atribut *class*. Metode KNN tidak berhasil menangkap model dengan kategori *balance*, tetapi cukup baik dalam menangkap kategori *right* dan *left* sehingga menghasilkan akurasi sebesar 83.2% dan presisi 82.2%. Metode *Naïve Bayes* mendapatkan nilai kebaikan model sebesar 89.6%, akurasi 88.8%, dan presisi 83%. *Naïve Bayes* tidak mampu menangkap kategori *balance* pada atribut kelas. Metode *Random Forest* mendapatkan nilai akurasi sebesar 80.8% dengan presisi 83.6%. Pada penelitian ini, *Random Forest* tidak mampu menangkap kategori *balance* dengan baik. Selanjutnya *Classification and Regression Tree* (CART) mendapatkan akurasi sebesar 78.4% dengan presisi 77.56% karena tidak cukup baik dalam menangkap kategori *balance*.

Tabel 2.2 State of The Art

No	Penulis	Metode	Dataset	Hasil
1.	Liang & Kusnadi (2021)	<i>Support Vector Machine</i> (SVM), <i>XGBoost</i> , dan <i>Neural Network</i>	Data diambil dari sentimen publik Twitter dalam Bahasa Indonesia sebanyak 18793 data. 11282 data <i>non-hatespeech</i> dan 7511 data <i>hatespeech</i>	Akurasi tertinggi diperoleh dari metode <i>Support Vector Machine</i> (SVM) dengan kernel <i>linear</i> yaitu 83.2%
2.	Ivan et al. (2019)	<i>Naive Bayes</i> + <i>Information Gain</i>	Data berasal dari Twitter sebanyak 250 data. 100 data <i>training hatespeech</i> , 100 data <i>training non hatespeech</i> , dan 25 data <i>testing</i>	Hasil akurasi menggunakan normalisasi kata dan seleksi fitur <i>Information Gain</i> dengan <i>threshold</i> 80% yaitu 98%.

Tabel 2.3 State of The Art (Lanjutan)

No	Penulis	Metode	Dataset	Hasil
3.	Rahman et al. (2021)	<i>Support Vector Machine</i> (SVM)	Data yang digunakan sebanyak 1000 <i>tweet</i> . Data latih sebanyak 700 data dan data uji sebanyak 300 data	Kernel RBF memiliki akurasi tertinggi dibandingkan kernel <i>linear</i> dan kernel <i>sigmoid</i> yaitu sebesar 93%
4.	Lyrawati (2019)	<i>Support Vector Machine</i> (SVM)	597 <i>tweet</i> yang diambil dari <i>Tweepy</i> API pada tanggal 10-24 Januari 2019	Akurasi tertinggi menggunakan kernel RBF sebesar 61.667% dengan waktu 1.21 detik menggunakan nilai <i>epsilon</i> dan $C = 1$.
5.	Fu et al. (2010)	<i>Support Vector Machine</i> (SVM)	Dataset diambil dari data <i>U.S. Geological Survey</i> oleh <i>the Colorado State University</i> yang berjumlah 581.012 data. Setiap <i>instance</i> memiliki 54 atribut fitur.	<i>Support Vector Machine</i> menggunakan kernel RBF menghasilkan akurasi tertinggi yaitu 95.45% dengan waktu 76.25 detik, <i>MLSVM</i> menghasilkan akurasi 95.20% dengan waktu 6.55 detik, dan <i>LSVM</i> menghasilkan akurasi 91.34 dengan waktu tercepat 1.65 detik
6.	Priambodo et al. (2022)	<i>Support Vector Machine</i> (SVM)	Dataset <i>non-linear</i> berjumlah 200 data RGB yang terdiri dari 2 kategori yaitu 100 gambar “besar” untuk Pakcoy yang sudah matang dan “kecil” untuk yang belum matang.	<i>Support Vector Machine</i> dengan menggunakan kernel RBF menghasilkan akurasi sebesar 100% pada proses training dan 79% pada proses evaluasi model terhadap citra acak diluar dataset.
7.	Olive et al. (2020)	<i>Naive Bayes</i> + <i>Principal Component Analysis</i> (PCA)	Data diperoleh dari Twitter API berupa 597 <i>tweet</i> mengenai Grab dan 2249 <i>tweet</i> Gojek	Akurasi pada Grab adalah 74.34% dan Gojek 68.84% dengan nilai <i>threshold</i> sebesar 1.1
8.	Mantik et al. (2022)	<i>K-Nearest Neighbor</i>	Dataset <i>non-linear</i> berjumlah 140 file gamelan rindik berformat .wav dengan durasi 20 detik yang terdiri dari 2 label yaitu <i>contentment</i> dan <i>exuberance</i> .	<i>K-Nearest Neighbor algorithm</i> menghasilkan akurasi sebesar 81%, presisi 76%, <i>recall</i> 90.5%, dan <i>f-score</i> 82.6%. Waktu tercepat menghasilkan 0.0258 detik.
9.	Pusean et al. (2023)	<i>Support Vector Machine</i> (SVM) dan <i>Naïve Bayes</i>	Data yang digunakan merupakan data <i>non-linear</i> , 900 data <i>tweet</i> yang diambil dengan <i>keyword</i> “XFactorID” pada bulan Juli 2021 sd 2022. Data terdiri dari 3 kelas yaitu positif, negatif, dan netral	<i>Support Vector Machine</i> dengan skenario kedua menggunakan kernel <i>linear</i> menghasilkan akurasi terbaik dibandingkan <i>Naïve Bayes</i> yaitu sebesar 79.44% dan untuk skenario pertama diperoleh akurasi sebesar 74.44%.

Tabel 2.4 State of The Art (Lanjutan)

No	Penulis	Metode	Dataset	Hasil
10.	Tanjum (2018)	<i>K-Nearest Neighbor</i> , <i>Super Vector Machine</i> , <i>Naïve Bayes</i> , <i>Random Forest</i> dan <i>Classification and Regression Tree</i> (CART)	Data yang digunakan sebanyak 625 data, dengan persebaran klasifikasi data 49 data <i>balance</i> (B), 288 data <i>right</i> (R), dan 288 data <i>left</i> (L).	Metode terbaik yaitu diperoleh oleh metode <i>Support Vector Machine</i> yang menghasilkan akurasi sebesar 92% dan presisi 95.58%. dari hasil <i>confussion matrix</i> , <i>Support Vector Machine</i> dapat menangkap semua kategori pada klasifikasi atribut <i>class</i> .

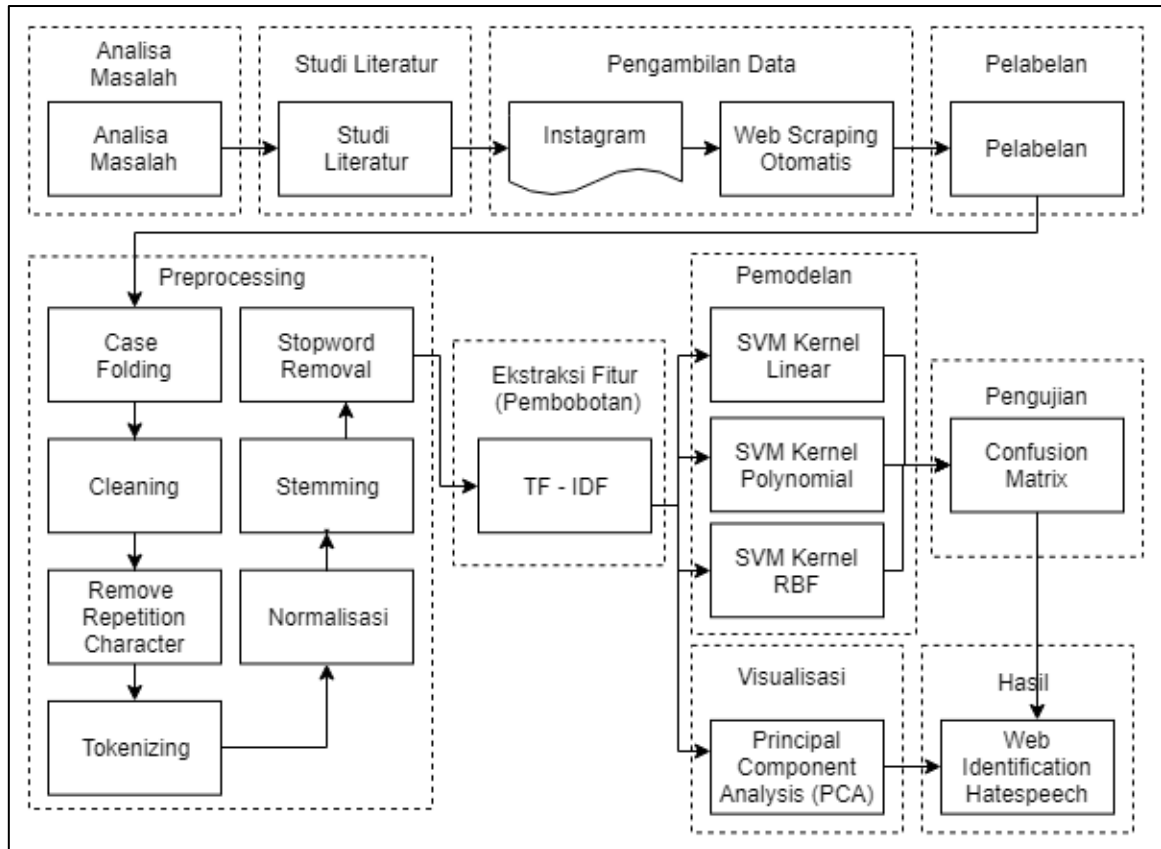
Berdasarkan penelitian terdahulu, maka penelitian yang akan dilakukan memiliki karakteristik yang relatif sama dalam hal klasifikasi. Namun penelitian ini akan memiliki perbedaan dari sisi data, dan metode yang digunakan. Penelitian Ivan et al. (2019), Mantik et al. (2022), Tanjum (2018), dan Olive et al. (2020) menggunakan metode seperti *K-Nearest Neighbor*, *Naïve Bayes*, *Random Forest* serta *Classification and Regression Tree* (CART) sedangkan pada penelitian Liang & Kusnadi (2021), Pusean et al. (2023), Fu et al. (2010), Priyambodo et al. (2022), Rahman et al. (2021), dan Lyrawati (2019) menggunakan metode *Support Vector Machine* tetapi dengan kernel *linear*, *sigmoid*, dan RBF. Rahman et al. (2021), Lyrawati (2019), dan Olive et al. (2020) tidak menggunakan normalisasi pada tahap *preprocessing*. Pada penelitian ini menggunakan metode *Support Vector Machine* (SVM) dengan kernel *polynomial*, kernel RBF, dan kernel *linear*. Dengan menggunakan data *non-linear* mengenai komentar-komentar *hatespeech* di media sosial Instagram. Selain itu, penelitian ini juga menggunakan metode *Principal Component Analysis* (PCA) untuk memvisualisasi data sehingga dapat diketahui data yang digunakan merupakan data *non-linear*, serta menggunakan proses normalisasi pada tahap *preprocessing*.

BAB III

METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM

3.1 Metodologi Penelitian

Tahapan-tahapan metodologi penelitian pada penelitian ini meliputi analisa masalah, studi literatur, pengambilan data, pelabelan data, *preprocessing*, pembobotan (ekstraksi fitur), pembuatan model (kernel *linear*, kernel *polynomial*, dan kernel RBF), dan pengujian. Tahapan metodologi penelitian dapat dilihat pada Gambar 3.1 berikut.



Gambar 3.1 Tahapan Metodologi Penelitian

3.1.1 Analisa Masalah

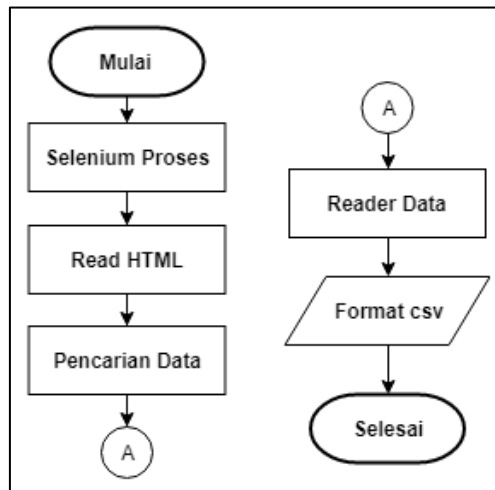
Analisa masalah merupakan gambaran permasalahan yang diangkat pada penulisan skripsi mengenai identifikasi hatespeech berbahasa Indonesia pada komentar Instagram yang memiliki bentuk data *non-linear*. Salah satu permasalahan yang sering dihadapi *Support Vector Machine* (SVM) adalah ketika data yang dimiliki berbentuk *non-linear*. Hal ini terjadi karena data tidak dapat dibagi secara *linear* menggunakan garis lurus sehingga mengakibatkan *Support Vector Machine* kesulitan dalam menemukan pemisah yang optimal antar kelas-kelas data. Untuk mengatasi hal tersebut, maka solusi yang diberikan yaitu dengan menggunakan fungsi kernel seperti kernel *polynomial* sehingga memungkinkan *Support Vector Machine* untuk melakukan tranformasi data *non-linear* kedalam ruang fitur yang lebih tinggi, dimana data tersebut dapat dipisahkan secara *linear*.

3.1.2 Studi Literatur

Studi literatur digunakan untuk mencari serta mengumpulkan informasi yang berkaitan dengan topik pada penelitian ini. Informasi yang dicari adalah informasi mengenai masalah penelitian, data yang digunakan, metode yang digunakan, serta kelebihan dan kekurangan pada penelitian yang pernah dilakukan. Hal tersebut digunakan untuk memperkuat argumentasi dan mempermudah penulis dalam melakukan penelitian ini. Informasi yang didapat bersumber dari artikel, buku, jurnal ilmiah dan sumber-sumber lain yang terpercaya.

3.1.3 Pengambilan Data

Data diambil dari komentar Instagram dengan cara *scraping* menggunakan Selenium. Selenium merupakan library *scraping* yang telah disediakan oleh bahasa pemrograman *python*. Data yang diambil yaitu *username* dan komentar text berbahasa Indonesia pada akun Instagram @putridelinaa, @fuji_an, @lestykejora, dan @kikysaputrii. Total jumlah komentar yang diambil sebanyak 918 komentar dengan 463 komentar *non-hatespeech* dan 455 komentar *hatespeech*.



Gambar 3.2 Flowchart Scraping Data

3.1.4 Pelabelan Data

Pada tahap ini, pelabelan data dilakukan secara manual oleh seorang guru bahasa Indonesia di SMKN 4 Bandar Lampung. Proses pelabelan data ini bertujuan untuk mengelompokkan semua data hasil *scraping* menjadi dua kelas yaitu komentar yang mengandung *hatespeech* dan komentar yang mengandung *non-hatespeech*.

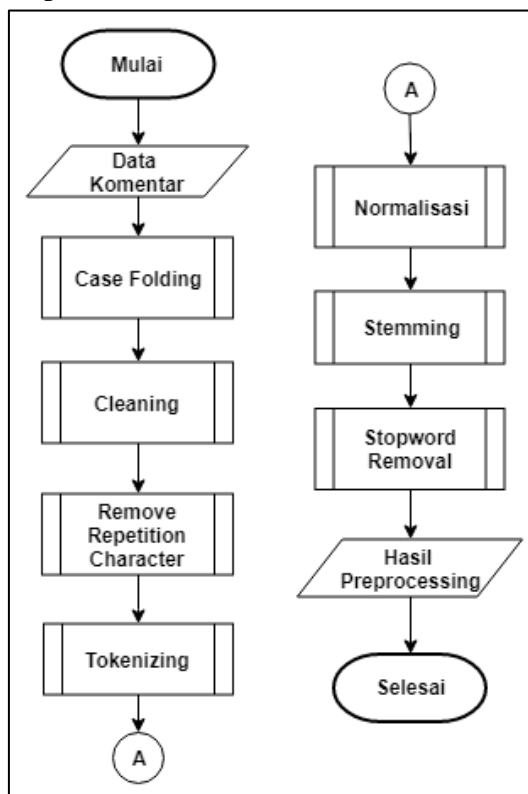
Tabel 3.1 Hasil Label Manual

No	Comment	Label
1	Masyaaallah.. ibunya cantik terlihat sopan dan anggun*.*	<i>Non-Hatespeech</i>
2	Makin jelek aja anaknya, padahal ibu ayahnya cakep2	<i>Hatespeech</i>

3.1.5 Preprocessing

Tahap *preprocessing* dilakukan untuk membersihkan serta mengubah data input menjadi format yang lebih sesuai dan siap untuk digunakan. Tahap *preprocessing* pada penelitian ini menggunakan library NLTK (*Natural Language Tool Kit*) yang telah tersedia di pemrograman *python*, yaitu terdiri dari proses *case folding*, *cleaning*, *remove*

repetition character, tokenizing, normalisasi, stemming, dan stopwords removal. Flowchart preprocessing dapat dilihat pada Gambar 3.3.



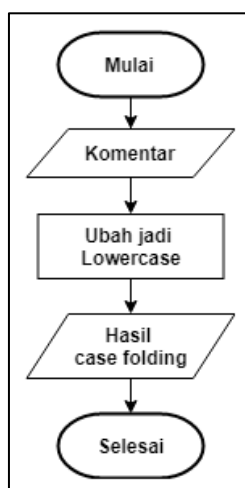
Gambar 3.3 Flowchart Tahap Preprocessing

a. Case Folding

Case folding merupakan tahap awal pada proses *preprocessing* data. Pada tahap ini, semua huruf yang terdapat di dalam data diubah menjadi huruf kecil menggunakan fungsi *.lower()*. Berikut adalah hasil beserta *flowchart* dari proses *case folding*.

Tabel 3.2 Hasil *Case Folding*

Komentar	Anaknya yg JELEK dan TOLOL mulai aktif yaaa bund...
Hasil case folding	anaknya yg jelek dan tolol ini mulai aktif yaaa bund...



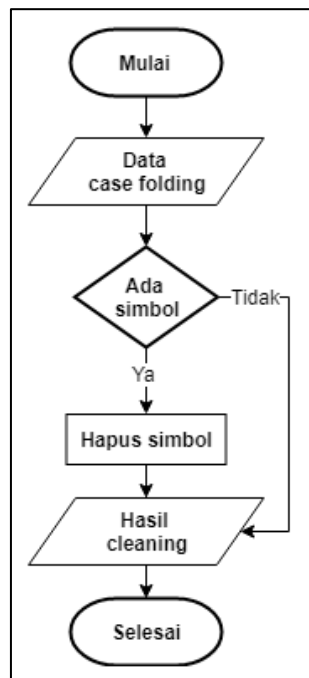
Gambar 3.4 Flowchart Case Folding

b. Cleaning

Tahap selanjutnya pada proses *preprocessing* yaitu tahap *case folding* yang bertujuan untuk membantu membersihkan data hasil *case folding* dari karakter-karakter yang tidak berpengaruh seperti tanda baca, *link url*, angka, emoji, simbol dan *hastag*. Berikut contoh hasil dan *flowchart* dari proses *preprocessing* tahap *cleaning*.

Tabel 3.3 Hasil *Cleaning*

Komentar	anaknya yg jelek dan tolol ini mulai aktif yaaa bund...
Hasil <i>cleaning</i>	anaknya yg jelek dan tolol ini mulai aktif yaaa bund



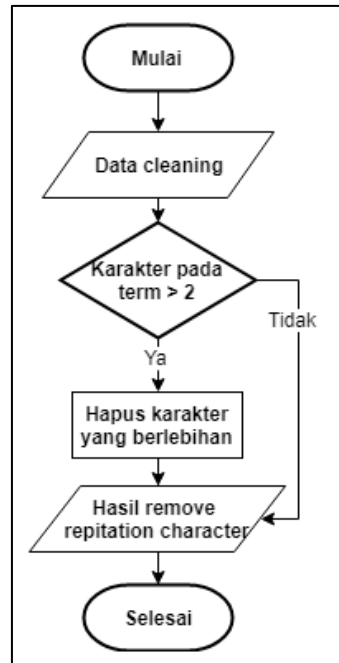
Gambar 3.5 Flowchart *Cleaning*

c. Remove Repetition Character

Pada tahap *remove repetition character*, karakter berulang yang tidak memberikan informasi tambahan pada teks akan dihapus sehingga dapat mengurangi varian yang tidak perlu dalam representasi teks. Hal ini dilakukan bertujuan untuk mengurangi beban saat melakukan pemrosesan. Penelitian ini mengatur pengulangan karakter lebih dari dua. Proses ini dimulai dengan memasukkan data hasil *cleaning* yang kemudian data tersebut akan diperiksa. Jika memiliki karakter berulang lebih dari dua karakter maka karakter lebih dari dua akan dihapus dari data. Jika tidak terdapat pengulangan karakter lebih dari dua maka data tersebut akan menjadi *output*/hasil dari proses ini dan proses *remove repetition character* selesai. Berikut contoh hasil dan *flowchart* dari proses *preprocessing* *remove repetition character*.

Tabel 3.4 Hasil *Remove Repetition Character*

Komentar	anaknya yg jelek dan tolol ini mulai aktif yaaa bund
Hasil <i>remove repetition character</i>	anaknya yg jelek dan tolol ini mulai aktif yaa bund



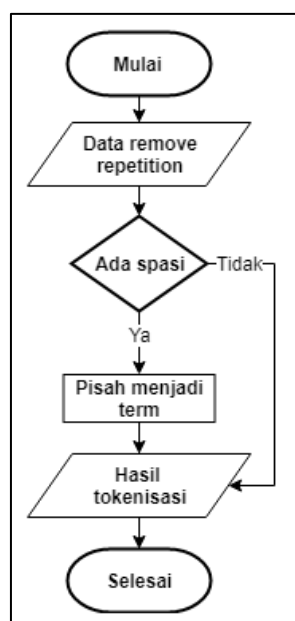
Gambar 3.6 Flowchart Remove Repetition Character

d. *Tokenizing*

Tahapan selanjutnya dari proses *remove repetition character* adalah proses *tokenizing*. Proses *tokenizing* digunakan untuk memisahkan kalimat pada data menjadi kata perkata/*term*. Spasi yang terdapat dalam data menjadi kunci dalam pemisahan kata, sehingga setiap ditemukan spasi pada data maka akan dipisahkan menjadi *term*. Berikut hasil dan *flowchart* dari proses *preprocessing tokenizing*.

Tabel 3.5 Hasil *Tokenizing*

Komentar	anaknya yg jelek dan tolol ini mulai aktif yaa bund
Hasil tokenezing	[anaknya,yg,jelek,dan,tolol,ini,mulai,aktif,yaa,bund]



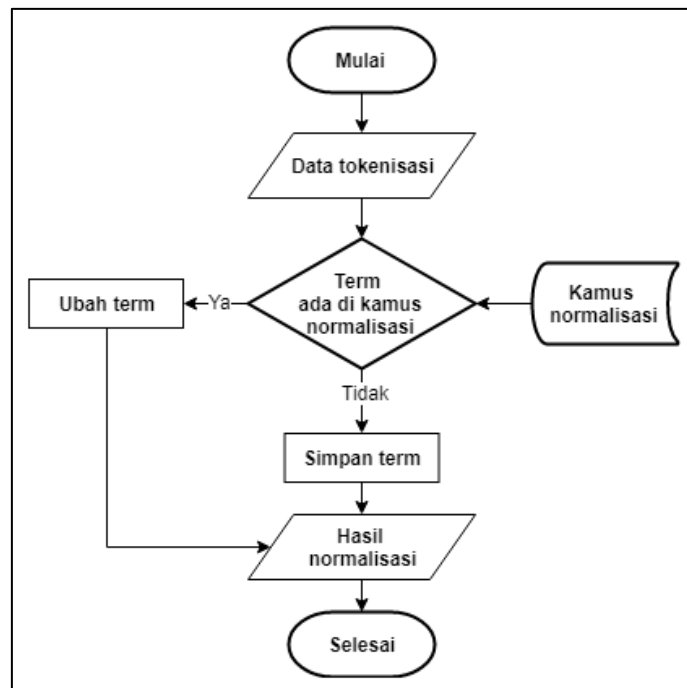
Gambar 3.7 Flowchart Tokenizing

e. Normalisasi

Tahap normalisasi dilakukan untuk mengubah data teks kedalam format yang seragam, seperti mengubah kata tidak baku menjadi kata baku serta mengganti singkatan ke dalam kosa kata sebenarnya. Tahapan ini dimulai dengan memasukkan data hasil *tokenizing* yang kemudian data tersebut akan dilakukan pemeriksaan kata apakah *term* terdapat di kamus normalisasi. Jika *term* tersebut terdapat dalam kamus normalisasi maka *term* akan diubah sesuai dengan *term* yang ada di dalam kamus normalisasi. Sedangkan jika sebaliknya, *term* tidak terdapat dalam kamus normalisasi maka *term* tersebut akan disimpan yang kemudian menjadi hasil dari proses normalisasi serta proses normalisasi selesai. Berikut merupakan contoh hasil beserta *flowchart* dari proses *preprocessing* normalisasi.

Tabel 3.6 Hasil Normalisasi

Komentar	[anaknya,yg,jelek,dan,tolol,ini,mulai,aktif,yaa,bund]
Hasil normalisasi	[anaknya,yang,jelek,dan,tolol,ini,mulai,aktif,ya,bunda]



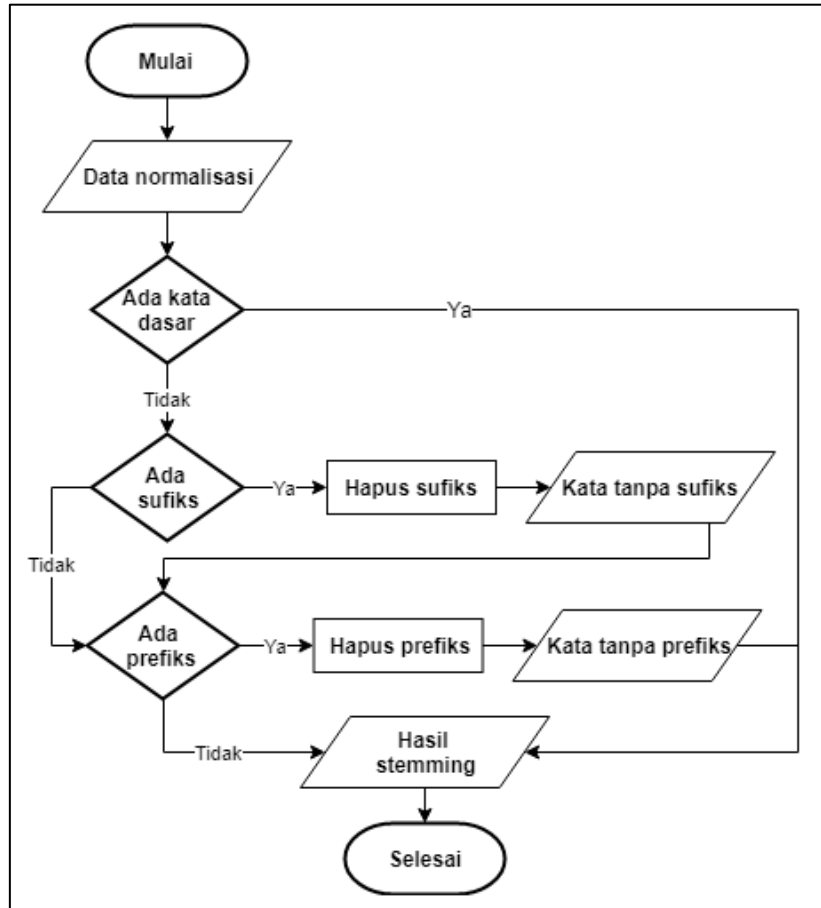
Gambar 3.8 Flowchart Normalisasi

f. Stemming

Setelah dilakukan proses normalisasi, selanjutnya adalah proses *stemming*. Pada tahap *stemming*, kata-kata berimbuhan diubah menjadi sebuah kata dasar. Proses ini dilakukan dengan menggunakan fungsi *stemmer* dari *library* sastrawi. Stemming akan mencari kata dasar dari setiap term dengan cara menghapus imbuhan pada *term*. Pertama data akan diperiksa apakah termasuk kata dasar atau tidak, jika *term* merupakan kata dasar maka kata tersebut akan disimpan. Selanjutnya data akan diperiksa *sufiks* yang merupakan imbuhan kata depan dan *prefix* yang merupakan imbuhan kata belakang. Berikut hasil dan *flowchart* dari proses *preprocessing stemming*.

Tabel 3.7 Hasil *Stemming*

Komentar	[anaknya,yang,jelek,dan,tolol,ini,mulai,aktif,ya,bunda]
Hasil stemming	[anak,yang,jelek,dan,tolol,ini,mulai,aktif,ya,bunda]



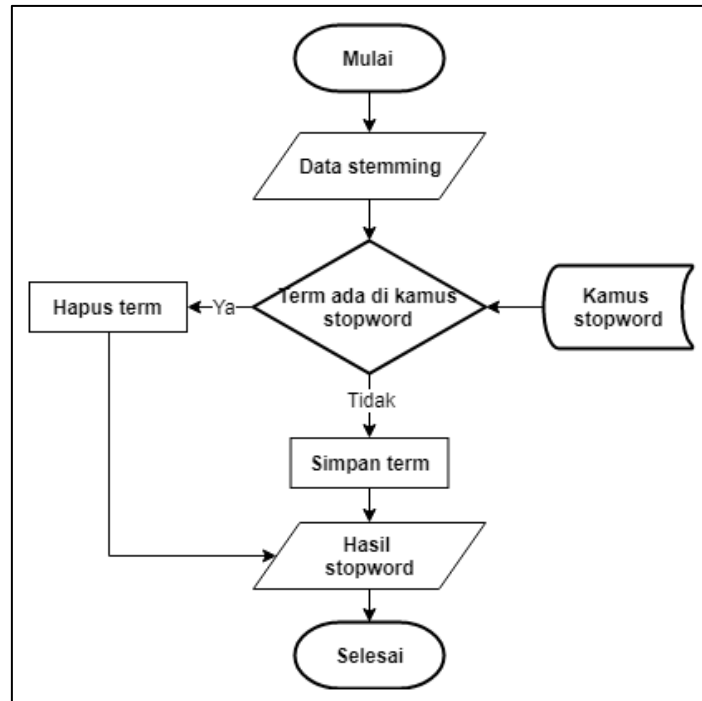
Gambar 3.9 Flowchart Stemming

g. Stopword Removal

Penghapusan *stopword* dilakukan untuk menghilangkan kata-kata yang sering muncul tetapi memiliki sedikit informasi penting, seperti kata penghubung dan lain-lain. Pada proses ini menggunakan kamus *stopword* bahasa Indonesia yang telah tersedia di *library* sastrawi serta menambahkan kata *stopword* lainnya secara manual. Tahapan ini dilakukan setelah proses stemming. Setelah memasukkan data hasil stemming, data tersebut akan dilakukan pemeriksaan kata apakah *term* terdapat di kamus *stopword*. Jika *term* tersebut terdapat dalam kamus *stopword* maka *term* akan dihapus. Sedangkan jika sebaliknya, *term* tidak terdapat dalam kamus *stopword* maka *term* tersebut akan disimpan yang kemudian menjadi hasil dari proses *stopword removal* dan proses *stopword removal* selesai. Berikut merupakan contoh hasil beserta *flowchart* dari proses *preprocessing stopwords removal*.

Tabel 3.8 Hasil *Stopword Removal*

Komentar	[anak,yang,jelek,dan,tolol,ini,mulai,aktif,ya,bunda]
Hasil stopwords removal	[anak,jelek,tolol,mulai,aktif,bunda]



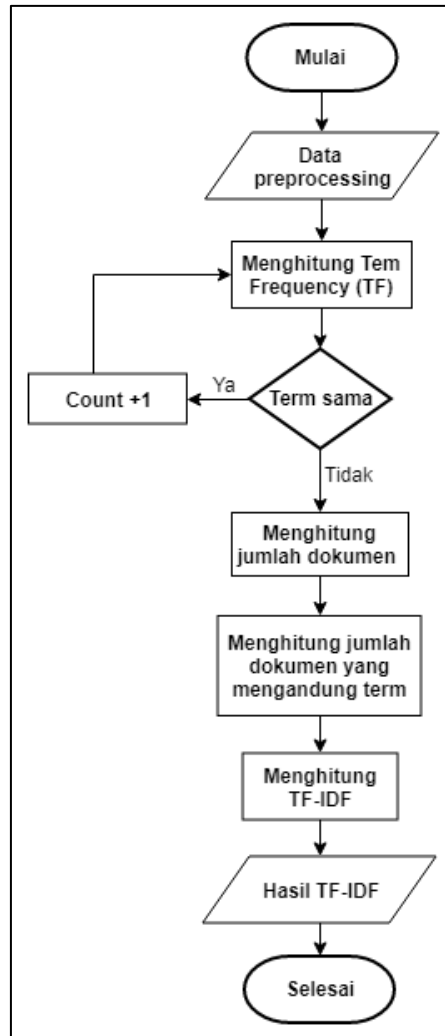
Gambar 3.10 Flowchart Stopword Removal

3.1.6 Term Frequency Inverse Document Frequency (TF-IDF)

Pembobotan *Term Frequency Inverse Document Frequency* (TF-IDF) dilakukan untuk melihat korelasi antar kata. *Term Frequency Inverse Document Frequency* merupakan perkalian dari *Term Frequency* dan *Inverse Document Frequency*. *Term Frequency* merupakan pembobotan term berdasarkan jumlah kemunculan kata pada satu dokumen, sedangkan *Inverse Document Frequency* kemunculan pada dokumen lain. Pada penelitian ini proses *Term Frequency Inverse Document Frequency* menggunakan library *TFidfVectorizer*. Berikut adalah flowchart dan contoh perhitungan manual dari proses *Term Frequency Inverse Document Frequency* (TF-IDF).

Tabel 3.9 Data Hasil *Preprocessing*

ID	Komentar	Komentar hasil preprocessing	Label
D1	Masyaaallah.. ibunya cantik terlihat sopan dan anggun*.*	masyaallah ibu cantik lihat sopan anggun	<i>Non-Hatespeech</i>
D2	Makin jelek aja anaknya, padahal ibu ayahnya cakep2	makin jelek anak padahal ibu ayah cakep	<i>Hatespeech</i>
U	Anaknya yg JELEK dan TOLOL mulai aktif yaaa bund...	anak jelek tolol mulai aktif bunda	?



Gambar 3.11 Flowchart TF-IDF

Untuk mendapatkan hasil pembobotan TF-IDF, langkah pertama yaitu dilakukan perhitungan *Term Frequency* dengan menghitung jumlah kemunculan *term* atau kata tiap dokumen.

Tabel 3.10 Contoh Perhitungan TF

Term	TF		
	D1	D2	U1
masyaallah	1	0	0
ibu	1	1	0
cantik	1	0	0
lihat	1	0	0
sopan	1	0	0
anggun	1	0	0
makin	0	1	0
jelek	0	1	1
anak	0	1	1
padahal	0	1	0
ayah	0	1	0
cakep	0	1	0
tolol	0	0	1

Tabel 3.11 Contoh Perhitungan TF (Lanjutan)

Term	TF		
	D1	D2	U1
mulai	0	0	1
aktif	0	0	1
bunda	0	0	1

Langkah selanjutnya adalah melakukan perhitungan nilai *Inverse Document Frequency*. $df(t)$ merupakan dokumen yang mengandung suatu term, dalam hal ini term yang akan digunakan sebagai contoh yaitu “aktif”. Sedangkan nilai n adalah jumlah seluruh dokumen yang ada di dalam sistem.

$$\begin{aligned}
 IDF_t &= \log \frac{n}{df(t)} \\
 &= \log \frac{3}{1} \\
 &= 0.4771
 \end{aligned}$$

Setelah nilai IDF didapat, maka dilanjutkan dengan melakukan perkalian nilai TF dan IDF menggunakan persamaan sebagai berikut:

$$\begin{aligned}
 TF - IDF &= TF_t \times IDF_t \\
 &= 1 \times 0.4771 \\
 &= 0.4771
 \end{aligned}$$

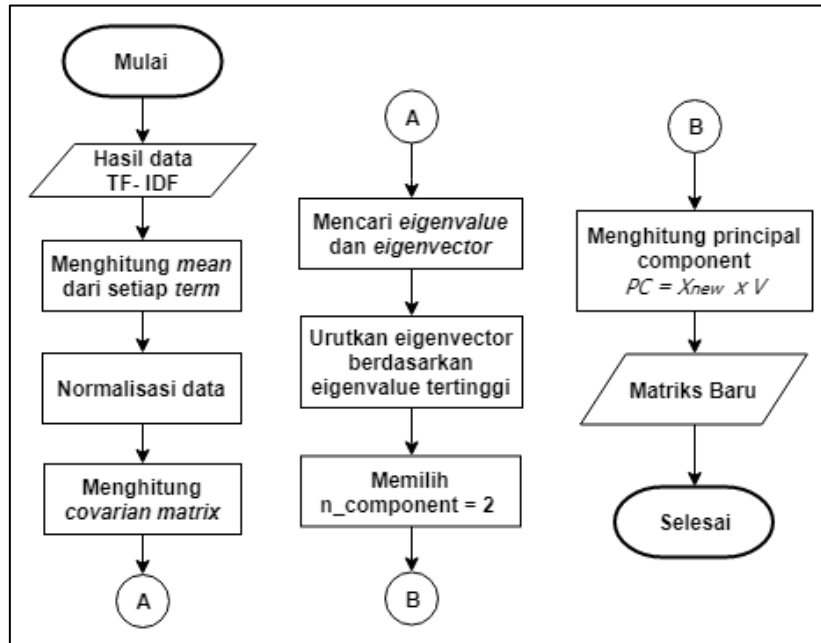
Tabel 3.12 Contoh Hasil TF-IDF

Term	TF	IDF	TF-IDF		
			D1	D2	U1
Masyaallah	1	0.4771	0.4771	0	0
Ibu	2	0.1761	0.1761	0.1761	0
Cantik	1	0.4771	0.4771	0	0
Lihat	1	0.4771	0.4771	0	0
Sopan	1	0.4771	0.4771	0	0
Anggun	1	0.4771	0.4771	0	0
Makin	1	0.4771	0	0.4771	0
Jelek	2	0.1761	0	0.1761	0.1761
Anak	2	0.1761	0	0.1761	0.1761
Padahal	1	0.4771	0	0.4771	0
Ayah	1	0.4771	0	0.4771	0
Cakep	1	0.4771	0	0.4771	0
Tolol	1	0.4771	0	0	0.4771
Mulai	1	0.4771	0	0	0.4771
Aktif	1	0.4771	0	0	0.4771
Bunda	1	0.4771	0	0	0.4771

3.1.7 Principal Component Analysis (PCA)

Pada tahap ini, data yang digunakan merupakan data yang telah melalui proses *preprocessing* dan pembobotan *Term Frequency Inverse Document Frequency* (TF-IDF). Hasil dari penerapan *Principal Component Analysis* (PCA) pada penelitian ini tidak akan dilanjutkan ke tahap klasifikasi. *Principal Component Analysis* (PCA) hanya digunakan untuk membantu memvisualisasikan data yang dipakai, sehingga dapat diketahui data

tersebut merupakan data *linear* atau *non-linear*. Berikut *flowchart* dari proses *Principal Component Analysis* (PCA).



Gambar 3.12 *Flowchart Principal Component Analysis*

Langkah-langkah penerapan metode *Principal Component Analysis* sebagai berikut.

- a. Mencari mean dari setiap term

$$\bar{x} = \frac{\sum x_i}{n}$$

$$\bar{x}_{term1} = \frac{0.4771 + 0}{2} = 0.23855$$

Tabel 3.13 Contoh Hasil Perhitungan *Mean* (\bar{x})

No	Term	TF-IDF		Mean
		D1	D2	
1	masyaallah	0.4771	0	0.23855
2	ibu	0.1761	0.1761	0.1761
3	cantik	0.4771	0	0.23855
4	lihat	0.4771	0	0.23855
5	sopan	0.4771	0	0.23855
6	anggun	0.4771	0	0.23855
7	makin	0	0.4771	0.23855
8	jelek	0	0.1761	0.08805
9	anak	0	0.1761	0.08805
10	padahal	0	0.4771	0.23855
11	ayah	0	0.4771	0.23855
12	cakep	0	0.4771	0.23855

- b. Normalisasi data

$$x_{new} = (x_i - \bar{x})$$

$$x_{1new} = (x_1 - \bar{x})$$

$$= (0.4771 - 0.23855)$$

$$= 0.23855$$

Tabel 3.14 Contoh Hasil Normalisasi Data

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
D1	0.239	0	0.239	0.239	0.239	0.239	-0.239	-0.088	-0.088	-0.239	-0.239	-0.239
D2	-0.239	0	-0.239	-0.239	-0.239	-0.239	0.239	0.088	0.088	0.239	0.239	0.239

c. Menghitung *covarian matrix*

$$C_x = \frac{\sum (x_1 - \bar{x})(y_1 - \bar{y})}{n - 1}$$

$$C_{T1,T2} = \frac{(0.4771 - 0.23855)(0.4771 - 0.23855) + (0 - 0.23855)(0 - 0.23855)}{2 - 1}$$

$$= 0.114$$

Tabel 3.15 Contoh Hasil Perhitungan *Covarian Matrix*

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
T1	0.114	0	0.114	0.114	0.114	0.114	-0.114	-0.042	-0.042	-0.114	-0.114	-0.114
T2	0	0	0	0	0	0	0	0	0	0	0	0
T3	0.114	0	0.114	0.114	0.114	0.114	-0.114	-0.042	-0.042	-0.114	-0.114	-0.114
T4	0.114	0	0.114	0.114	0.114	0.114	-0.114	-0.042	-0.042	-0.114	-0.114	-0.114
T5	0.114	0	0.114	0.114	0.114	0.114	-0.114	-0.042	-0.042	-0.114	-0.114	-0.114
T6	0.114	0	0.114	0.114	0.114	0.114	-0.114	-0.042	-0.042	-0.114	-0.114	-0.114
T7	-0.114	0	-0.114	-0.114	-0.114	-0.114	0.114	0.042	0.042	0.114	0.114	0.114
T8	-0.042	0	-0.042	-0.042	-0.042	-0.042	-0.042	0.042	0.016	0.042	0.042	0.042
T9	-0.042	0	-0.042	-0.042	-0.042	-0.042	-0.042	0.042	0.016	0.042	0.042	0.042
T10	-0.114	0	-0.114	-0.114	-0.114	-0.114	0.114	0.042	0.042	0.114	0.114	0.114
T11	-0.114	0	-0.114	-0.114	-0.114	-0.114	0.114	0.042	0.042	0.114	0.114	0.114
T12	-0.114	0	-0.114	-0.114	-0.114	-0.114	0.114	0.042	0.042	0.114	0.114	0.114

d. Mencari *eigenvalue* dan *eigenvector*

$$C_x V_m = \lambda_m V_m$$

$$(C - \lambda I)V = 0$$

$$\det(C - \lambda I) = 0$$

$$\det \begin{pmatrix} 0.114 - \lambda & 0 & 0.114 & 0.114 & 0.114 & 0.114 & -0.114 & -0.042 & -0.042 & -0.114 & -0.114 & -0.114 \\ 0 & 0 - \lambda & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.114 & 0 & 0.114 - \lambda & 0.114 & 0.114 & 0.114 & -0.114 & -0.042 & -0.042 & -0.114 & -0.114 & -0.114 \\ 0.114 & 0 & 0.114 & 0.114 - \lambda & 0.114 & 0.114 & -0.114 & -0.042 & -0.042 & -0.114 & -0.114 & -0.114 \\ 0.114 & 0 & 0.114 & 0.114 & 0.114 - \lambda & 0.114 & -0.114 & -0.042 & -0.042 & -0.114 & -0.114 & -0.114 \\ 0.114 & 0 & 0.114 & 0.114 & 0.114 & 0.114 - \lambda & -0.114 & -0.042 & -0.042 & -0.114 & -0.114 & -0.114 \\ -0.114 & 0 & -0.114 & -0.114 & -0.114 & -0.114 & 0.114 - \lambda & 0.042 & 0.042 & 0.114 & 0.114 & 0.114 \\ -0.042 & 0 & -0.042 & -0.042 & -0.042 & -0.042 & -0.042 & 0.042 - \lambda & 0.042 & 0.042 & 0.042 & 0.042 \\ -0.042 & 0 & -0.042 & -0.042 & -0.042 & -0.042 & -0.042 & 0.042 & 0.016 - \lambda & 0.042 & 0.042 & 0.042 \\ -0.114 & 0 & -0.114 & -0.114 & -0.114 & -0.114 & 0.114 & 0.042 & 0.016 & 0.114 - \lambda & 0.114 & 0.114 \\ -0.114 & 0 & -0.114 & -0.114 & -0.114 & -0.114 & 0.114 & 0.042 & 0.042 & 0.114 & 0.114 - \lambda & 0.114 \\ -0.114 & 0 & -0.114 & -0.114 & -0.114 & -0.114 & 0.114 & 0.042 & 0.042 & 0.114 & 0.114 & 0.114 - \lambda \end{pmatrix} V = 0$$

Menghasilkan nilai eigen dan vektor eigen sebagai berikut

Tabel 3.16 Contoh Hasil Perhitungan *Eigenvalue*

λ_1	0
λ_2	3.05532106
λ_3	2.61926472
λ_4	-6.10482037

Tabel 3.17 Contoh Hasil Perhitungan *Eigenvalue* (Lanjutan)

λ_5	-6.06081502
λ_6	-2.96300430
λ_7	-2.96300430
λ_8	-2.34165445
λ_9	1.57613247
λ_{10}	1.01797331
λ_{11}	1.00259989
λ_{12}	0

Tabel 3.18 Contoh Hasil Perhitungan *Eigenvector*

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9	λ_{10}	λ_{11}	λ_{12}
v_1	-0.945	0.328	-0.239	0.185	-0.122	-0.923	-0.923	0.944	0.945	0.945	-0.945	0
v_2	0	0	0	0	0	0	0	0	0	0	0	1
v_3	0.114	0.328	-0.202	0.185	-0.122	0.109	0.109	-0.108	-0.114	-0.114	0.114	0
v_4	0.114	0.328	-0.202	0.185	-0.122	0.109	0.109	-0.108	-0.114	-0.114	0.114	0
v_5	0.114	0.328	-0.202	0.185	-0.122	0.112	0.112	-0.118	-0.114	-0.114	0.114	0
v_6	0.114	0.328	-0.202	0.185	-0.122	0.112	0.112	-0.118	-0.114	-0.114	0.114	0
v_7	-0.114	-0.328	0.202	-0.185	0.122	-0.112	-0.112	0.118	0.114	0.114	-0.114	0
v_8	-0.042	-0.121	-0.011	0.114	0.367	-0.183	-0.183	0.042	0.042	0.042	-0.042	0
v_9	-0.042	-0.121	-0.011	0.114	0.367	0.101	0.101	0.042	0.042	0.042	-0.042	0
v_{10}	-0.114	-0.328	-0.607	0.606	-0.561	-0.112	-0.112	0.115	0.114	0.114	-0.114	0
v_{11}	-0.114	-0.328	-0.607	0.606	-0.561	-0.112	-0.112	0.115	0.114	0.114	-0.114	0
v_{12}	-0.114	-0.328	-0.029	-0.185	0.122	-0.112	-0.112	0.114	0.114	0.114	-0.114	0

e. Mengurutkan *eigenvector* berdasarkan *eigenvalue* tertinggi

Tabel 3.19 Contoh Hasil *Eigenvector* yang telah diurutkan

	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	E_{11}	E_{12}
	λ_2	λ_3	λ_9	λ_{10}	λ_{11}	λ_{12}	λ_1	λ_8	λ_6	λ_7	λ_5	λ_4
v_1	0.328	-0.239	0.945	0.945	-0.945	0	-0.945	0.944	-0.923	-0.923	-0.122	0.185
v_2	0	0	0	0	0	1	0	0	0	0	0	0
v_3	0.328	-0.202	-0.114	-0.114	0.114	0	0.114	-0.108	0.109	0.109	-0.122	0.185
v_4	0.328	-0.202	-0.114	-0.114	0.114	0	0.114	-0.108	0.109	0.109	-0.122	0.185
v_5	0.328	-0.202	-0.114	-0.114	0.114	0	0.114	-0.118	0.112	0.112	-0.122	0.185
v_6	0.328	-0.202	-0.114	-0.114	0.114	0	0.114	-0.118	0.112	0.112	-0.122	0.185
v_7	-0.328	0.202	0.114	0.114	-0.114	0	-0.114	0.118	-0.112	-0.112	0.122	-0.185
v_8	-0.121	-0.011	0.042	0.042	-0.042	0	-0.042	0.042	-0.183	-0.183	0.367	0.114
v_9	-0.121	-0.011	0.042	0.042	-0.042	0	-0.042	0.042	0.101	0.101	0.367	0.114
v_{10}	-0.328	-0.607	0.114	0.114	-0.114	0	-0.114	0.115	-0.112	-0.112	-0.561	0.606
v_{11}	-0.328	-0.607	0.114	0.114	-0.114	0	-0.114	0.115	-0.112	-0.112	-0.561	0.606
v_{12}	-0.328	-0.029	0.114	0.114	-0.114	0	-0.114	0.114	-0.112	-0.112	0.122	-0.185

Setelah *eigenvector* diurutkan, kemudian memilih komponen utama sesuai jumlah komponen yang diinginkan. Pada penelitian ini menggunakan $n_component = 2$ sehingga akan dipilih 2 vektor eigen teratas.

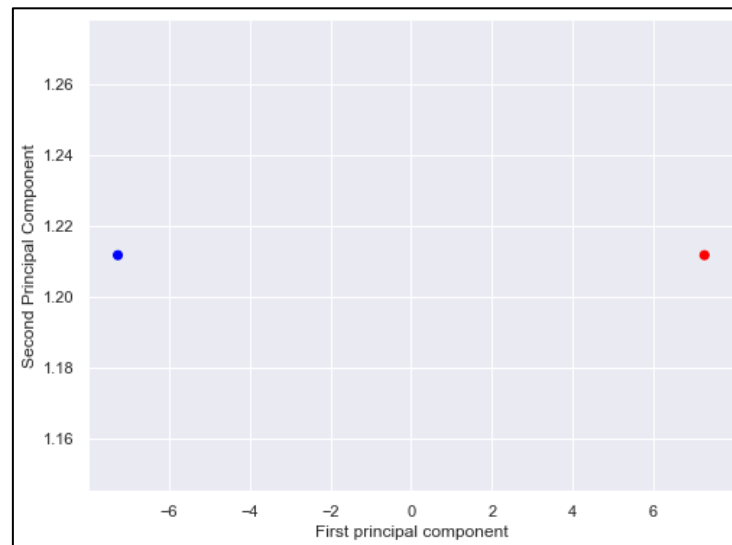
Tabel 3.20 Contoh Hasil *Eigenvector* $n_component = 2$

E_1	E_2
0.328	-0.239
0	0
0.328	-0.202
0.328	-0.202
0.328	-0.202
0.328	-0.202
-0.328	0.202
-0.121	-0.011
-0.121	-0.011
-0.328	-0.607
-0.328	-0.607
-0.328	-0.029

f. *Principal Components*

$$\begin{aligned}
 PC &= X_{1new} \times V \\
 &= \begin{pmatrix} 0.239 & 0 & 0.239 & 0.239 & 0.239 & 0.239 & -0.239 & -0.088 & -0.088 & -0.239 & -0.239 & -0.239 \\ -0.239 & 0 & -0.239 & -0.239 & -0.239 & -0.239 & 0.239 & 0.088 & 0.088 & 0.239 & 0.239 & 0.239 \end{pmatrix} \\
 &\quad \times \begin{pmatrix} 0.328 & -0.239 \\ 0 & 0 \\ 0.328 & -0.202 \\ 0.328 & -0.202 \\ 0.328 & -0.202 \\ 0.328 & -0.202 \\ -0.328 & 0.202 \\ -0.121 & -0.011 \\ -0.121 & -0.011 \\ -0.328 & -0.607 \\ -0.328 & -0.607 \\ -0.328 & -0.029 \end{pmatrix} \\
 &= \begin{pmatrix} 7.2640245 & 1.2116971 \\ -7.2670245 & 1.2118971 \end{pmatrix}
 \end{aligned}$$

Berikut grafik dari contoh hasil perhitungan *Principal Component Analysis* pada 2 data



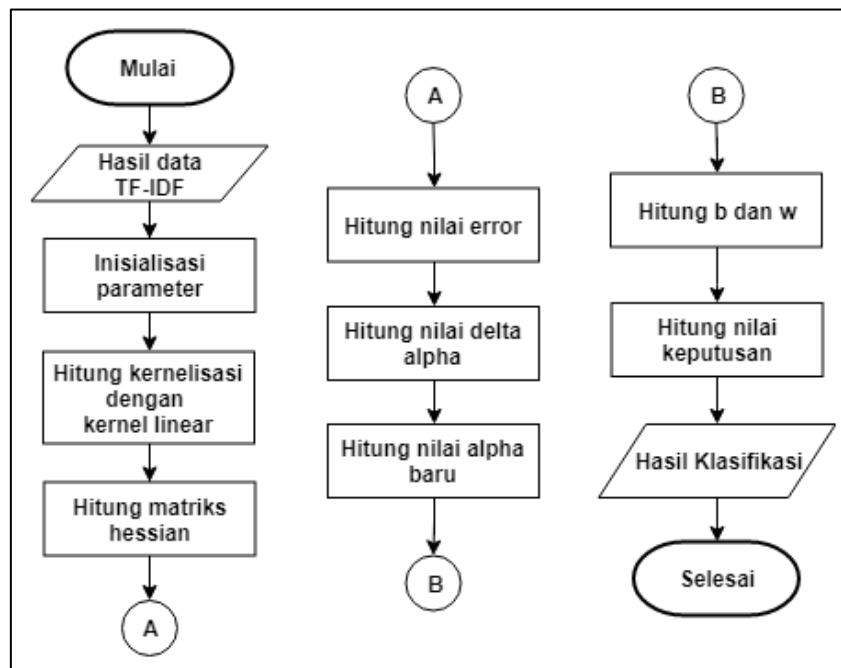
Gambar 3.13 Grafik Contoh Hasil *Principal Component Analysis* (PCA)

3.1.8 Pembuatan Model

Pembuatan model dilakukan setelah melewati proses preprocessing dan pembobotan *Term Frequency Inverse Document Frequency*. Pada penelitian ini akan menggunakan metode *Support Vector Machine* (SVM) dengan menerapkan dua kernel, yaitu kernel *linear*, *polynomial*, dan RBF.

A. *Support Vector Machine Kernel Linear*

Penerapan metode *Support Vector Machine* menggunakan kernel *linear* terdiri dari dua tahap, yaitu penerapan metode pada data latih (*training*) dan penerapan metode pada data uji (*test*) yang sudah melewati proses *preprocessing* dan pembobotan. Proses pada data latih (*training*) digunakan untuk menghasilkan model yang dijadikan acuan dalam mengklasifikasikan kelas pada data uji. Berikut *flowchart* dari model SVM kernel *Linear*.



Gambar 3.14 *Flowchart Model SVM Kernel Linear*

Berikut langkah penerapan *Support Vector Machine* (SVM) menggunakan kernel *linear* pada klasifikasi data:

a. Melakukan inisialisasi pada parameter

Maksimum iterasi = 2

$$\gamma = 0.001$$

$$a = 0$$

$$C = 0.01$$

$$\lambda = 0.5$$

b. Mencari nilai kernelisasi

$$X_1 = [0.4771, 0.1761, 0.4771, 0.4771, 0.4771, 0.4771, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

$$X_2 = [0, 0.1761, 0, 0, 0, 0, 0.4771, 0.1761, 0.1761, 0.4771, 0.4771, 0.4771, 0, 0, 0, 0]$$

$$K(x_i, x_j) = (x_i^T \cdot x_j)$$

$$K(x_1, x_2) = (x_1^T \cdot x_2)$$

$$\begin{aligned}
&= ((0.4771 \times 0) + (0.1761 \times 0.1761) + (0.4771 \times 0) + (0.4771 \times 0) \\
&\quad + (0.4771 \times 0) + (0.4771 \times 0) + (0 \times 0.4771) + (0 \times 0.1761) + (0 \times 0.1761) \\
&\quad + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0) + (0 \times 0) + (0 \times 0) \\
&\quad + (0 \times 0)) \\
&= 0.03101121
\end{aligned}$$

Tabel 3.21 Matriks K Kernel *Linear*

Dokumen	D1	D2
D1	1.16913326	0.03101121
D2	0.03101121	1.00353127

c. Menghitung matriks Hessian

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

$$\begin{aligned}
D_{12} &= (0)(1)(0.03101121) + 0.5^2 \\
&= 0.25
\end{aligned}$$

Tabel 3.22 Matriks Hessian Kernel *Linear*

Dokumen	D1	D2
D1	0.25	0.25
D2	0.25	1.25353127

d. Menghitung nilai *error*

$$\begin{aligned}
E_i &= \sum_{j=1}^n a_i D_{ij} \\
&= (0 \times 0.25) + (0 \times 0.25) \\
&= 0
\end{aligned}$$

Tabel 3.23 Hasil Perhitungan Nilai *Error* Kernel *Linear*

Dokumen	E_i
D1	0
D2	0

e. Menghitung *delta* a_i

$$\begin{aligned}
\delta a_i &= \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\
&= \min\{\max[0.001(1 - 0), 0], 0.01 - 0\} \\
&= 0.001
\end{aligned}$$

Tabel 3.24 Hasil Perhitungan δa_i Kernel *Linear*

Dokumen	δa_i
D1	0.001
D2	0.001

f. Menghitung a_i baru

$$\begin{aligned}
a_i \text{ baru} &= a_i + \delta a_i \\
&= 0 + 0.001 \\
&= 0.001
\end{aligned}$$

Tabel 3.25 Hasil Perhitungan a_i Kernel *Linear*

Dokumen	a_i
D1	0.001
D2	0.001

Kemudian melakukan perulangan langkah mencari nilai *error*, mencari nilai δa_i dan mencari nilai a_i sampai batas iterasi yang telah ditentukan.

$$\begin{aligned}
 E_i &= \sum_{j=1}^n a_i D_{ij} \\
 &= (0.001 \times 0.25) + (0.001 \times 0.25) \\
 &= 0.0005
 \end{aligned}$$

Tabel 3.26 Hasil Iterasi Perhitungan Nilai *Error* Kernel *Linear*

Dokumen	E_i
D1	0.0005
D2	0.00150353127

$$\begin{aligned}
 \delta a_i &= \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\
 &= \min\{\max[0.001(1 - 0.0005), 0], 1 - 0.001\} \\
 &= \min\{\max[0.0009995, 0], 0.999\} \\
 &= 0.0009995
 \end{aligned}$$

Tabel 3.27 Hasil Iterasi Perhitungan δa_i Kernel *Linear*

Dokumen	δa_i
D1	0.0009995
D2	0.000998496469

$$\begin{aligned}
 a_i \text{ baru} &= a_i + \delta a_i \\
 &= 0.001 + 0.0009995 \\
 &= 0.0019995
 \end{aligned}$$

Tabel 3.28 Hasil Iterasi Perhitungan a_i Kernel *Linear*

Dokumen	a_i
D1	0.0019995
D2	0.001998496469

g. Menghitung nilai $w.x^+$ dan $w.x^-$ untuk mendapatkan nilai bias

$$\begin{aligned}
 w.x^+ &= a_i Y_i K(w.x^+) \\
 &= (0.0019995 \times 0 \times 0.25) + (0.001998496469 \times 1 \times 0.25) \\
 &= 0.00049962411725 \\
 w.x^- &= a_i Y_i K(w.x^-) \\
 &= (0.0019995 \times 0 \times 0.25) + (0.001998496469 \times 1 \times 1.25353127) \\
 &= -0.0025051778168761
 \end{aligned}$$

$$\begin{aligned}
b &= -\frac{1}{2}(w \cdot x^+ + w \cdot x^-) \\
&= -\frac{1}{2}(0.00049962411725 + (-0.0025051778168761)) \\
&= 0.0010027768498131
\end{aligned}$$

h. Menghitung nilai keputusan

$$U = [0, 0, 0, 0, 0, 0, 0.1761, 0.1761, 0, 0, 0, 0, 0.4771, 0.4771, 0.4771, 0.4771]$$

$$K(x_i, x_j) = (x_i^T \cdot x_j)$$

$$\begin{aligned}
K(x_u, x_1) &= (x_u^T \cdot x_1) \\
&= ((0 \times 0.4771) + (0 \times 0.1761) + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0.4771) \\
&\quad + (0 \times 0.4771) + (0.1761 \times 0) + (0.1761 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) \\
&\quad + (0 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0)) \\
&= 0
\end{aligned}$$

$$\begin{aligned}
K(x_u, x_2) &= (x_u^T \cdot x_2) \\
&= ((0 \times 0) + (0 \times 0.1761) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0.1761 \times 0) \\
&\quad + (0.1761 \times 0.1761) + (0 \times 0.1761) + (0 \times 0.4771) + (0 \times 0.4771) \\
&\quad + (0 \times 0.4771) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0)) \\
&= 0.03101121
\end{aligned}$$

Tabel 3.29 Hasil Kernelisasi Seluruh Data Uji Kernel *Linear*

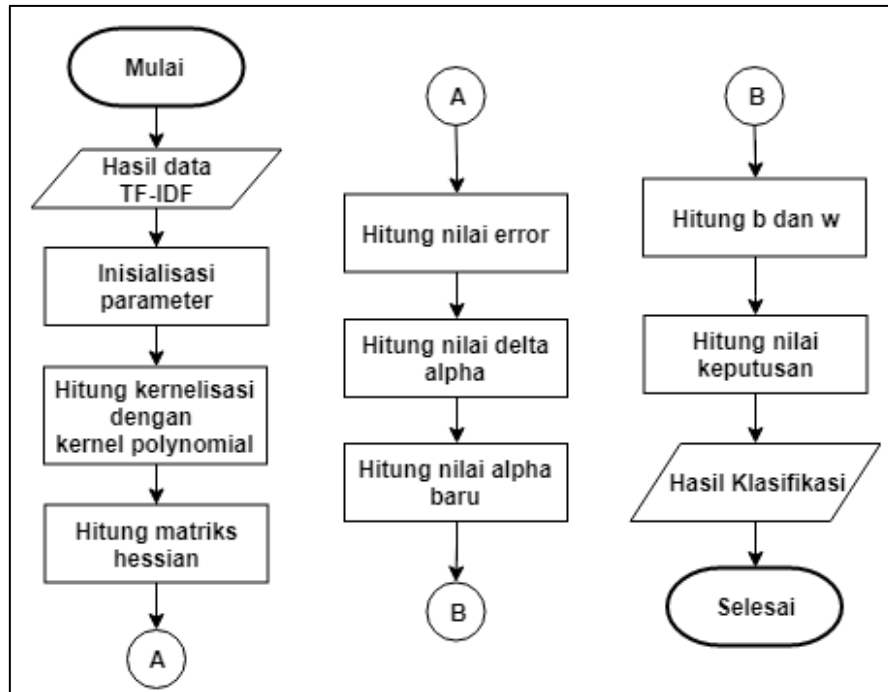
$K(x_u, x_1)$	$K(x_u, x_2)$
0	0.03101121

$$\begin{aligned}
f(x) &= \sum_{i=0}^m \text{sign}(a_i Y_i K(x, x_i) + b) \\
&= \text{sign}((0.0019995 \times 0 \times 0) + (0.001998496469 \times 1 \times 0.03101121) \\
&\quad + 0.0010027768498131) \\
&= \text{sign}(0.0010647526434975) \\
&= 1
\end{aligned}$$

Dari hasil $f(x)$ maka data U dikategorikan masuk ke dalam komentar yang mengandung *hatespeech*. Jika nilai $f(x) > 0$ maka data tersebut akan masuk ke dalam kategori *hatespeech*. Sebaliknya jika nilai $f(x) \leq 0$ maka data tersebut termasuk ke dalam kategori *non-hatespeech*.

B. Support Vector Machine Kernel Polynomial

Penerapan metode *Support Vector Machine* menggunakan kernel *polynomial* mempunyai langkah yang sama dengan *Support Vector Machine* kernel *linear*, perbedaannya terdapat pada penggunaan rumus kernel. Berikut *flowchart* dari model SVM kernel *polynomial*.



Gambar 3.15 Flowchart Model SVM Kernel Polynomial

Berikut langkah penerapan *Support Vector Machine* (SVM) menggunakan kernel *polynomial* pada klasifikasi data:

a. Melakukan inisialisasi pada parameter

Maksimal iterasi = 2

$$\gamma = 0.001$$

$$a = 0$$

$$C = 1$$

$$\lambda = 0.5$$

$$d = 1$$

b. Mencari nilai kernelisasi

$$X_1 = [0.4771, 0.1761, 0.4771, 0.4771, 0.4771, 0.4771, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

$$X_2 = [0, 0.1761, 0, 0, 0, 0, 0.4771, 0.1761, 0.1761, 0.4771, 0.4771, 0.4771, 0, 0, 0, 0]$$

$$K(x_i, x_j) = (x_i^T x + 1)^d$$

$$K(x_1, x_2) = (x_i^T x + 1)^d$$

$$\begin{aligned}
 &= (((0.4771 \times 0) + (0.1761 \times 0.1761) + (0.4771 \times 0) + (0.4771 \times 0) \\
 &\quad + (0.4771 \times 0) + (0.4771 \times 0) + (0 \times 0.4771) + (0 \times 0.1761) + (0 \times 0.1761) \\
 &\quad + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0) + (0 \times 0) + (0 \times 0) \\
 &\quad + (0 \times 0)) + 1)^1
 \end{aligned}$$

$$= 1.03101121$$

Tabel 3.30 Matriks K Kernel Polynomial

Dokumen	D1	D2
D1	2.16913326	1.03101121
D2	1.03101121	2.00353127

c. Menghitung matriks Hessian

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

$$D_{12} = (0)(1)(1.03101121) + 0.5^2 \\ = 0.25$$

Tabel 3.31 Matriks Hessian Kernel *Polynomial*

Dokumen	D1	D2
D1	0.25	0.25
D2	0.25	1.28101121

d. Menghitung nilai *error*

$$E_i = \sum_{j=1}^n a_i D_{ij} \\ = (0 \times 0.25) + (0 \times 0.25) \\ = 0$$

Tabel 3.32 Hasil Perhitungan Nilai *Error* Kernel *Polynomial*

Dokumen	E_i
D1	0
D2	0

e. Menghitung *delta* a_i

$$\delta a_i = \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\ = \min\{\max[0.001(1 - 0), 0], 1 - 0\} \\ = 0.001$$

Tabel 3.33 Hasil Perhitungan δa_i Kernel *Polynomial*

Dokumen	δa_i
D1	0.001
D2	0.001

f. Menghitung a_i baru

$$a_i \text{ baru} = a_i + \delta a_i \\ = 0 + 0.001 \\ = 0.001$$

Tabel 3.34 Hasil Perhitungan a_i Kernel *Polynomial*

Dokumen	a_i
D1	0.001
D2	0.001

Kemudian melakukan perulangan langkah mencari nilai *error*, mencari nilai δa_i dan mencari nilai a_i sampai batas iterasi yang telah ditentukan.

$$E_i = \sum_{j=1}^n a_i D_{ij} \\ = (0.001 \times 0.25) + (0.001 \times 0.25) \\ = 0.0005$$

Tabel 3.35 Hasil Iterasi Perhitungan Nilai *Error Kernel Polynomial*

Dokumen	E_i
D1	0.0005
D2	0.00153101121

$$\begin{aligned}
\delta a_i &= \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\
&= \min\{\max[0.001(1 - 0.0005), 0], 1 - 0.001\} \\
&= \min\{\max[0.0009995, 0], 0.999\} \\
&= 0.0009995
\end{aligned}$$

Tabel 3.36 Hasil Iterasi Perhitungan δa_i Kernel *Polynomial*

Dokumen	δa_i
D1	0.0009995
D2	0.00099846898879

$$\begin{aligned}
a_i \text{ baru} &= a_i + \delta a_i \\
&= 0.001 + 0.0009995 \\
&= 0.0019995
\end{aligned}$$

Tabel 3.37 Hasil Iterasi Perhitungan a_i Kernel *Polynomial*

Dokumen	a_i
D1	0.0019995
D2	0.00199846898879

g. Menghitung nilai $w.x^+$ dan $w.x^-$ untuk mendapatkan nilai bias

$$\begin{aligned}
w.x^+ &= a_i Y_i K(w.x^+) \\
&= (0.0019995 \times 0 \times 0.25) + (0.00199846898879 \times 1 \times 0.25) \\
&= 0.0004996172471975 \\
w.x^- &= a_i Y_i K(w.x^-) \\
&= (0.0019995 \times 0 \times 0.25) + (0.00199846898879 \times 1 \times 1.28101121) \\
&= -0.0025600611774774
\end{aligned}$$

$$\begin{aligned}
b &= -\frac{1}{2}(w.x^+ + w.x^-) \\
&= -\frac{1}{2}(0.0004996172471975 + (-0.0025600611774774)) \\
&= 0.0020604439302799
\end{aligned}$$

h. Menghitung nilai keputusan

$$U = [0, 0, 0, 0, 0, 0, 0.1761, 0.1761, 0, 0, 0, 0, 0.4771, 0.4771, 0.4771, 0.4771]$$

$$K(x_i, x_j) = (x_i^T x + 1)^d$$

$$K(x_w, x_1) = (x_i^T x + 1)^d$$

$$\begin{aligned}
&= (((0 \times 0.4771) + (0 \times 0.1761) + (0 \times 0.4771) + (0 \times 0.4771) + (0 \times 0.4771) \\
&\quad + (0 \times 0.4771) + (0.1761 \times 0) + (0.1761 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) \\
&\quad + (0 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0)) + 1)^1 \\
&= 1
\end{aligned}$$

$$\begin{aligned}
K(x_u, x_2) &= (x_i^T x + 1)^d \\
&= (((0 \times 0) + (0 \times 0.1761) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0 \times 0) + (0.1761 \times 0) \\
&\quad + (0.1761 \times 0.1761) + (0 \times 0.1761) + (0 \times 0.4771) + (0 \times 0.4771) \\
&\quad + (0 \times 0.4771) + (0.4771 \times 0) + (0.4771 \times 0) + (0.4771 \times 0) \\
&\quad + (0.4771 \times 0)) + 1)^1 \\
&= 1.03101121
\end{aligned}$$

Tabel 3.38 Hasil Kernelisasi Seluruh Data Uji Kernel *Polynomial*

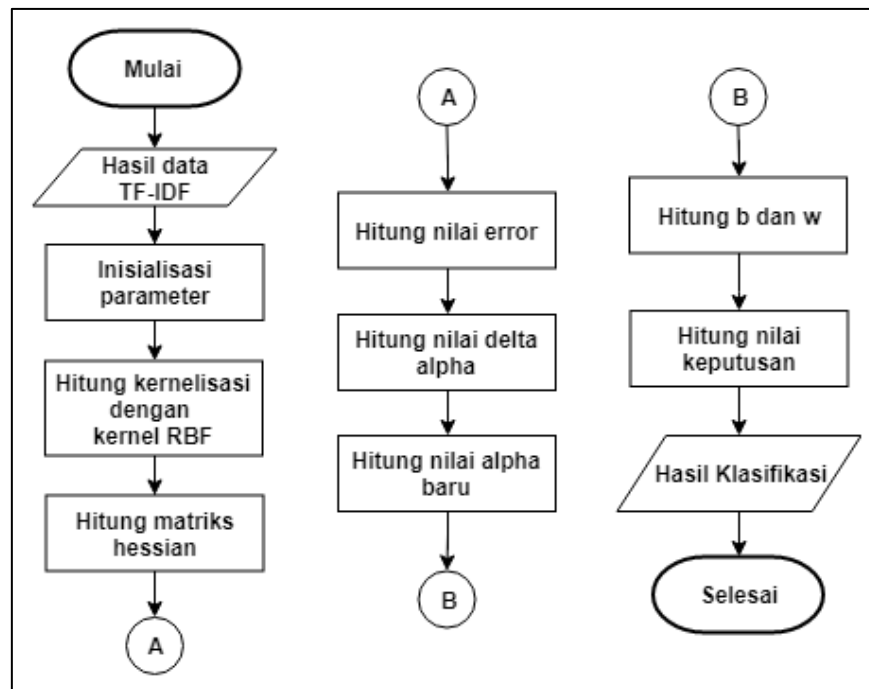
$K(x_u, x_1)$	$K(x_u, x_2)$
1	1.03101121

$$\begin{aligned}
f(x) &= \sum_{i=0}^m \text{sign}(a_i Y_i K(x, x_i) + b) \\
&= \text{sign}((0.0019995 \times 0 \times 1) + (0.001998496469 \times 1 \times 1.03101121) \\
&\quad + 0.0020604439302799) \\
&= \text{sign}(0.0041209161929643) \\
&= 1
\end{aligned}$$

Dari hasil $f(x)$ maka data U termasuk ke dalam komentar yang mengandung *hatespeech* karena nilai $f(x) > 0$.

C. *Support Vector Machine Kernel RBF*

Penerapan metode *Support Vector Machine* menggunakan kernel RBF mempunyai langkah yang sama dengan *Support Vector Machine* kernel *linear* dan *polynomial*, perbedaannya terdapat pada penggunaan rumus kernel. Berikut *flowchart* dari model SVM kernel RBF.



Gambar 3.16 Flowchart Model SVM Kernel RBF

Berikut langkah penerapan *Support Vector Machine* (SVM) menggunakan kernel *polynomial* pada klasifikasi data:

a. Melakukan inisialisasi pada parameter

Maksimal iterasi = 2

$$\gamma = 0.001$$

$$a = 0$$

$$C = 2$$

$$\lambda = 0.5$$

b. Mencari nilai kernelisasi

$$X_1 = [0.4771, 0.1761, 0.4771, 0.4771, 0.4771, 0.4771, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

$$X_2 = [0, 0.1761, 0, 0, 0, 0, 0.4771, 0.1761, 0.1761, 0.4771, 0.4771, 0.4771, 0, 0, 0]$$

$$K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2} \|x_i - x_j\|^2\right)$$

$$K(x_1, x_2) = \exp(-\gamma \|x_i - x_j\|^2)$$

$$\begin{aligned} &= \exp(-0.001 \| (0.4771 - 0) + (0.1761 - 0.1761) + (0.4771 - 0) + (0.4771 - 0) \\ &\quad + (0.4771 - 0) + (0.4771 - 0) + (0 - 0.4771) + (0 - 0.1761) + (0 - 0.1761) \\ &\quad + (0 - 0.4771) + (0 - 0.4771) + (0 - 0.4771) + (0 - 0) + (0 - 0) + (0 - 0) \\ &\quad + (0 - 0) \|^2) \\ &= 0.99998440 \end{aligned}$$

Tabel 3.39 Matriks K Kernel RBF

Dokumen	D1	D2
D1	1	0.99998440
D2	0.99998440	1

c. Menghitung matriks Hessian

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

$$\begin{aligned} D_{12} &= (0)(1)(0.99998440) + 0.5^2 \\ &= 0.25 \end{aligned}$$

Tabel 3.40 Matriks Hessian Kernel RBF

Dokumen	D1	D2
D1	0.25	0.25
D2	0.25	1.25

d. Menghitung nilai *error*

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

$$= (0 \times 0.25) + (0 \times 0.25)$$

$$= 0$$

Tabel 3.41 Hasil Perhitungan Nilai *Error* Kernel RBF

Dokumen	E_i
D1	0
D2	0

e. Menghitung δa_i

$$\begin{aligned}\delta a_i &= \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\ &= \min\{\max[0.001(1 - 0), 0], 2 - 0\} \\ &= 0.001\end{aligned}$$

Tabel 3.42 Hasil Perhitungan δa_i Kernel RBF

Dokumen	δa_i
D1	0.001
D2	0.001

f. Menghitung a_i baru

$$\begin{aligned}a_i \text{ baru} &= a_i + \delta a_i \\ &= 0 + 0.001 \\ &= 0.001\end{aligned}$$

Tabel 3.43 Hasil Perhitungan a_i Kernel RBF

Dokumen	a_i
D1	0.001
D2	0.001

Kemudian melakukan perulangan langkah mencari nilai $error$, mencari nilai δa_i dan mencari nilai a_i sampai batas iterasi yang telah ditentukan.

$$\begin{aligned}E_i &= \sum_{j=1}^n a_i D_{ij} \\ &= (0.001 \times 0.25) + (0.001 \times 0.25) \\ &= 0.0005\end{aligned}$$

Tabel 3.44 Hasil Iterasi Perhitungan Nilai $Error$ Kernel RBF

Dokumen	E_i
D1	0.0005
D2	0.0025

$$\begin{aligned}\delta a_i &= \min\{\max[\gamma(1 - E_i), a_i], C - a_i\} \\ &= \min\{\max[0.001(1 - 0.0005), 0], 2 - 0.001\} \\ &= \min\{\max[0.0009995, 0], 1.999\} \\ &= 0.0009995\end{aligned}$$

Tabel 3.45 Hasil Iterasi Perhitungan δa_i Kernel RBF

Dokumen	δa_i
D1	0.0009995
D2	0.0009975

$$\begin{aligned}a_i \text{ baru} &= a_i + \delta a_i \\ &= 0.001 + 0.0009995 \\ &= 0.0019995\end{aligned}$$

Tabel 3.46 Hasil Iterasi Perhitungan a_i Kernel RBF

Dokumen	a_i
D1	0.0019995
D2	0.0019975

g. Menghitung nilai $w \cdot x^+$ dan $w \cdot x^-$ untuk mendapatkan nilai bias

$$\begin{aligned} w \cdot x^+ &= a_i Y_i K(w \cdot x^+) \\ &= (0.0019995 \times 0 \times 0.25) + (0.0019975 \times 1 \times 0.25) \\ &= 0.000499375 \\ w \cdot x^- &= a_i Y_i K(w \cdot x^-) \\ &= (0.0019995 \times 0 \times 0.25) + (0.0019975 \times 1 \times 1.25) \\ &= -0.002496875 \end{aligned}$$

$$\begin{aligned} b &= -\frac{1}{2}(w \cdot x^+ + w \cdot x^-) \\ &= -\frac{1}{2}(0.000499375 + (-0.002496875)) \\ &= 0.00099875 \end{aligned}$$

h. Menghitung nilai keputusan

$$U = [0, 0, 0, 0, 0, 0, 0.1761, 0.1761, 0, 0, 0, 0, 0.4771, 0.4771, 0.4771, 0.4771]$$

$$K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2} \|x_i - x\|^2\right)$$

$$\begin{aligned} K(x_w, x_1) &= \exp(-\gamma \|x_i - x\|^2) \\ &= \exp(-0.001 \| (0-0.4771) + (0-0.1761) + (0-0.4771) + (0-0.4771) \\ &\quad + (0-0.4771) + (0-0.4771) + (0.1761-0) + (0.1761-0) + (0-0) + (0-0) \\ &\quad + (0-0) + (0-0) + (0.4771-0) + (0.4771-0) + (0.4771-0) + (0.4771-0) \|^2) \\ &= 0.99990940 \end{aligned}$$

$$\begin{aligned} K(x_w, x_2) &= \exp(-\gamma \|x_i - x\|^2) \\ &= \exp(-0.001 \| (0-0) + (0-0.1761) + (0-0) + (0-0) + (0-0) + (0-0) \\ &\quad + (0.1761-0) + (0.1761-0.1761) + (0-0.1761) + (0-0.4771) + (0-0.4771) \\ &\quad + (0-0.4771) + (0.4771-0) + (0.4771-0) + (0.4771-0) + (0.4771-0) \|^2) \\ &= 0.99990940 \end{aligned}$$

Tabel 3.47 Hasil Kernelisasi Seluruh Data Uji Kernel RBF

$K(x_w, x_1)$	$K(x_w, x_2)$
0.99990940	0.99990940

$$\begin{aligned} f(x) &= \sum_{i=0}^m \text{sign}(a_i Y_i K(x, x_i) + b) \\ &= \text{sign}((0.0019995 \times 0 \times 0.99990940) + (0.0019975 \times 1 \times 0.99990940) \\ &\quad + 0.00099875) \\ &= \text{sign}(0.0029960690265) \\ &= 1 \end{aligned}$$

Dari hasil $f(x)$ maka data U termasuk ke dalam komentar yang mengandung *hatespeech* karena nilai $f(x) > 0$.

3.1.9 Pengujian Model

Pada penelitian ini, model diuji menggunakan *confusion matrix* untuk menghitung nilai akurasi, presisi, serta *recall* dari kernel *linear*, *polynomial* dan RBF pada *Support Vector Machine* (SVM). Berikut table dari pengujian dua model kernel.

Tabel 3.48 Pengujian Model

Pengujian	SVM Kernel Linear	SVM Kernel Polynomial	SVM Kernel RBF
Akurasi			
Presisi			
Recall			

Tabel 3.49 Pengujian Actual Model SVM Kenel *Linear*

Komentar	SVM Kernel Linear		Hasil Actual
	Hatespeech	Non-Hatespeech	

Tabel 3.50 Pengujian Actual Model SVM Kenel *Polynomial*

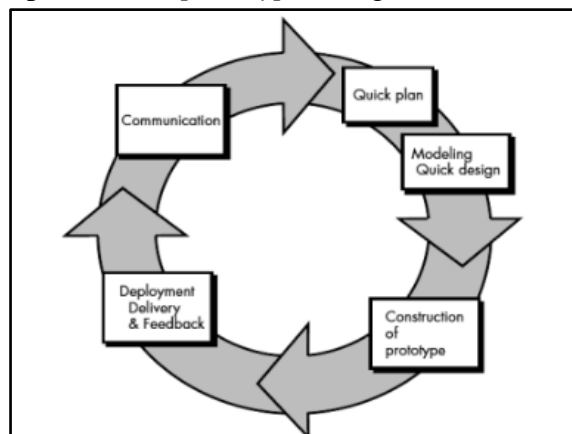
Komentar	SVM Kernel Polynomial		Hasil Actual
	Hatespeech	Non-Hatespeech	

Tabel 3.51 Pengujian Actual Model SVM Kenel RBF

Komentar	SVM Kernel RBF		Hasil Actual
	Hatespeech	Non-Hatespeech	

3.2 Metode Pengembangan Sistem

Pengembangan sistem pada penelitian ini menggunakan metode *prototype*. Langkah-langkah penerapan metode *prototype* sebagai berikut:



Gambar 3.17 Metode Prototyping

Sumber : (Pressman, 2005)

A. *Communication*

Communication merupakan tahap awal untuk mengetahui kebutuhan perangkat lunak yang akan dikembangkan. Kebutuhan tersebut dibagi menjadi dua, yaitu kebutuhan fungsional dan kebutuhan non fungsional.

a. **Kebutuhan Fungsional**

Kebutuhan fungsional merupakan gambaran dari keseluruhan proses yang dapat dilakukan oleh sistem. Kebutuhan fungsional pada penelitian ini meliputi:

1. Sistem dapat menampilkan dataset yang digunakan
2. Sistem dapat menampilkan hasil *preprocessing* dari dataset yang digunakan
3. Sistem dapat melakukan input data teks komentar
4. Sistem dapat melakukan *preprocessing* pada data input teks
5. Sistem dapat melakukan proses klasifikasi pada data teks
6. Sistem dapat menampilkan hasil pengujian yaitu akurasi, presisi, dan *recall*
7. Sistem dapat menampilkan visualisasi data dan grafik visualisasi dari hasil klasifikasi

b. **Kebutuhan Non Fungsional**

Kebutuhan non fungsional adalah kebutuhan perangkat yang digunakan dalam melakukan penelitian ini. Kebutuhan non fungsional meliputi:

1. Kebutuhan perangkat keras (*Hardware*)

Tabel 3.52 Kebutuhan Non Fungsional

No	Perangkat Keras	Keterangan
1	Processor	Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz
2	RAM	4.00 GB
3	Storage	1 TB
4	System Type	64 bit

2. Kebutuhan perangkat lunak (*Software*)

Tabel 3.53 Kebutuhan Fungsional

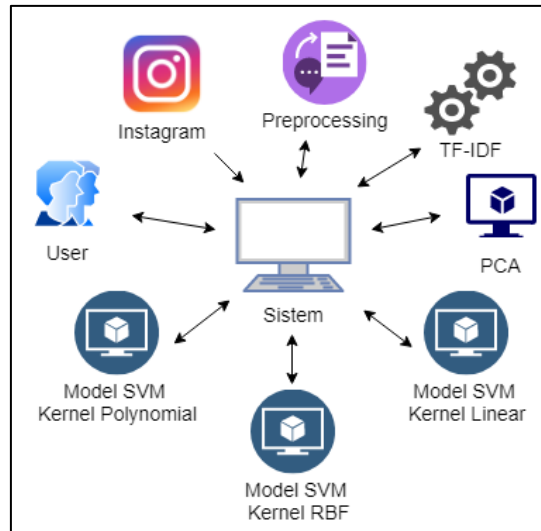
No	Perangkat Lunak	Keterangan
1	Sistem operasi	Windows 10 64 bit
2	Bahasa pemrograman	Python 3.9
3	IDE	Visual Studio Code
4	Web browser	Google Chrome
5	Software pembuat diagram dan UI	Draw.io

B. *Quick Plan and Modelling Quick Design*

Tahap ini memberikan gambaran umum mengenai bagaimana sistem dilakukan. Tahap *quick plan and modeling quick design* sebagai berikut.

1. **Perancangan Arsitektur**

Perancangan arsitektur sistem digunakan untuk memberi gambaran sistem yang akan dikembangkan. Perancangan arsitektir sistem dapat dilihat pada Gambar 3.16.



Gambar 3.18 Arsitektur Sistem

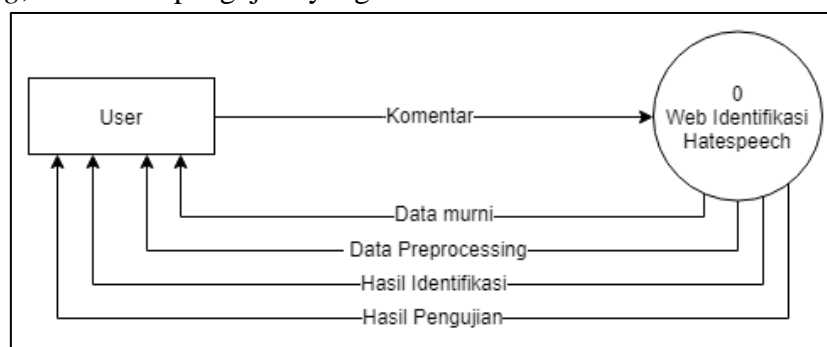
Dalam pengoprasian sistem ini terdiri dari 1 aktor yaitu *user*. Data diperoleh dari hasil *scraping* pada komentar di media sosial Instagram. Kemudian data yang sudah dilabeli, dilakukan *preprocessing* dan pembobotan kata menggunakan TF-IDF yang kemudian akan dilakukan identifikasi *hatespeech* menggunakan model *Support Vector Machine* kernel *linear* dan *Support Vector Machine* kernel *polynomial* yang telah dibangun. Hasil identifikasi kemudian akan ditampilkan kepada *user*.

2. Perancangan Proses

Perancangan proses merupakan gambaran proses yang berjalan di sistem yang dikembangkan. Perancangan proses dilakukan dengan membuat *Data Flow Diagram* (DFD).

a. *Data Flow Diagram Level 0*

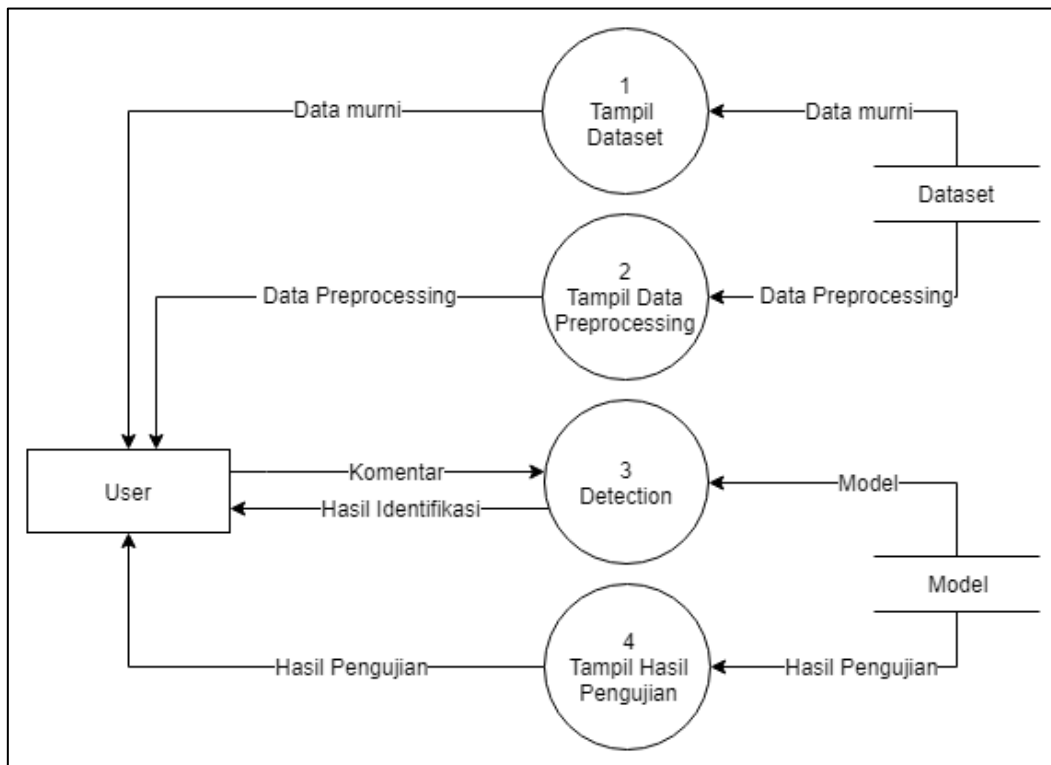
Data Flow Diagram level 0 merupakan diagram yang mempresentasikan seluruh elemen yang terdapat di dalam sistem berupa aliran data tunggal yang menggambarkan adanya *input* dan *output* secara umum. Dalam sistem ini terdapat satu entitas yaitu *user* yang dapat memasukkan komentar teks pada sistem untuk dilakukan identifikasi *hatespeech* dan *non-hatespeech*. Sistem akan memproses dan melakukan klasifikasi pada komentar tersebut yang kemudian hasilnya akan ditampilkan kepada *user*. *User* juga dapat melihat dataset yang digunakan, hasil dari dataset yang telah melewati proses *preprocessing*, serta hasil pengujian yang telah dilakukan oleh sistem.



Gambar 3.19 DFD Level 0

b. Data Flow Diagram Level 1

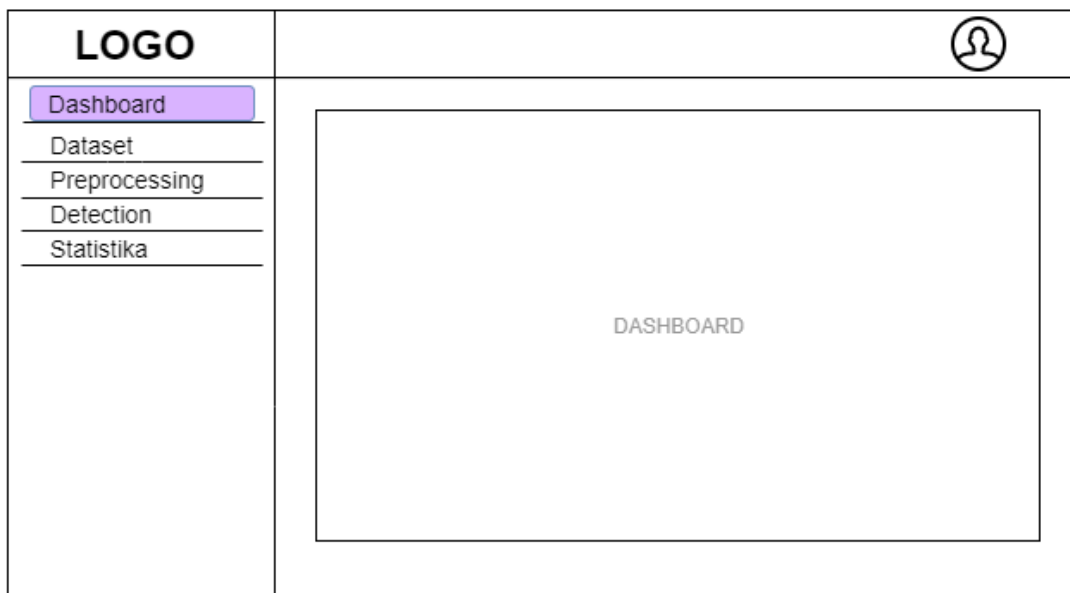
Data Flow Diagram level 1 merupakan turunan dari level 0 yang mencakup hal lebih mendetail dari sebuah sistem yang dikembangkan. Terdapat 4 proses pada *Data Flow Diagram* level 1 ini. Proses pertama adalah tampil dataset yang bekerja menampilkan dataset murni yang telah dimasukkan ke sistem dengan format file csv untuk dapat dilihat oleh *user*. Proses kedua adalah tampil data *preprocessing*, dataset yang telah melewati proses *preprocessing* data dan tersimpan di dataset.csv ditampilkan pada *user*. Proses ketiga yaitu *detection*, *user* menginputkan kalimat yang ingin dideteksi kemudian sistem akan memproses dan menghasilkan *output* berupa hasil deteksi yang dapat diberikan kepada *user*. Proses terakhir yaitu tampil hasil pengujian yang menampilkan visualisasi data dan perbandingan nilai akurasi, presisi, serta *recall* antara metode *Support Vector Machine* dengan kernel *linear*, metode *Support Vector Machine* dengan kernel *polynomial*, dan metode *Support Vector Machine* dengan kernel RBF. Data *storage* dataset dan model pada *Data Flow Diagram* level 1 ini bukan merupakan *database*, melainkan berupa *file*.



Gambar 3.20 DFD Level 1

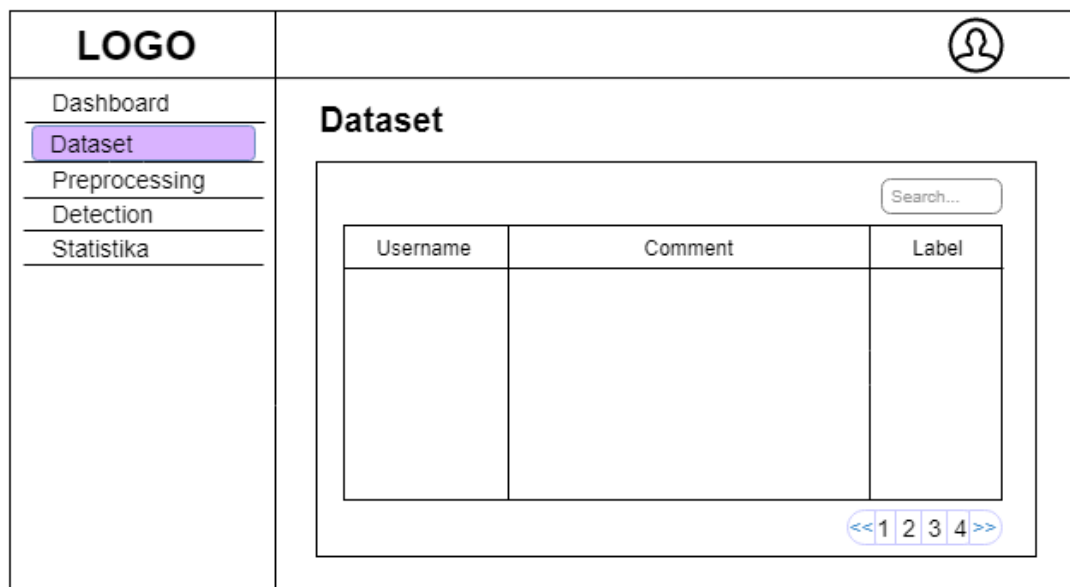
3. Perancangan Antarmuka

Perancangan antarmuka berfungsi untuk menggambarkan tampilan dari sistem yang dikembangkan. Pada sistem ini terdiri dari 5 halaman yaitu *Dashboard*, *Dataset*, *Preprocessing*, *Detection*, dan *Statistika*.



Gambar 3.21 Rancangan Halaman *Dashboard*

Gambar 3.19 adalah perancangan antarmuka dari halaman *dashboard* yang merupakan tampilan awal yang akan dilihat oleh *user* ketika membuka sistem ini.



Gambar 3.22 Rancangan Halaman Dataset

Gambar 3.20 merupakan perancangan antarmuka dari halaman dataset. Halaman ini akan menampilkan dataset murni hasil *scraping* yang telah melalui proses pelabelan. Informasi yang diberikan berupa informasi *username*, *comment*, dan *label*.

LOGO Dashboard Dataset Preprocessing Detection Statistika	<div> </div> <h2>Preprocessing</h2> <div> <input type="text" value="Search..."/> </div> <table border="1"> <thead> <tr> <th>Username</th> <th>Comment</th> <th>Case Folding</th> <th>Cleaning</th> <th>Repetition</th> <th>Tokenizing</th> <th>Normalisasi</th> <th>Stemming</th> <th>Stopword</th> </tr> </thead> <tbody> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> <tr><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td><td> </td></tr> </tbody> </table> <div> << 1 2 3 4 >> </div>	Username	Comment	Case Folding	Cleaning	Repetition	Tokenizing	Normalisasi	Stemming	Stopword																																													
	Username	Comment	Case Folding	Cleaning	Repetition	Tokenizing	Normalisasi	Stemming	Stopword																																														

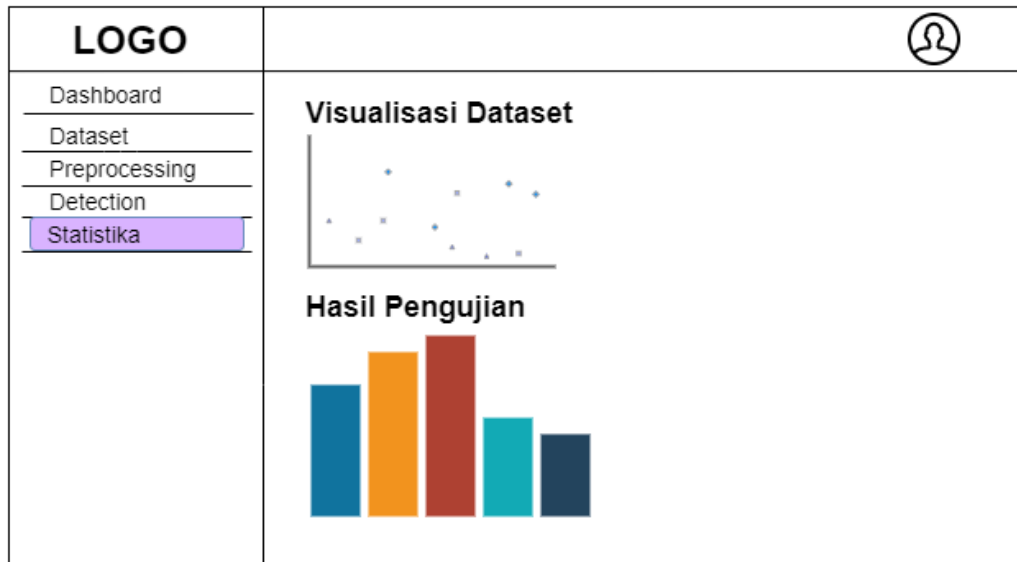
Gambar 3.23 Rancangan Halaman *Preprocessing*

Gambar 3.21 merupakan perancangan antarmuka dari halaman *preprocessing*. Halaman ini menampilkan dataset yang telah melewati proses *preprocessing* yang terdiri dari *case folding*, *cleaning*, *repetition*, *tokenizing*, *normalisasi*, *stemming*, dan *stopword*.

LOGO Dashboard Dataset Preprocessing Detection Statistika	<div> </div> <h2>Hatespeech Detection</h2> <div> <input type="text" value="Input Komentar..."/> <input type="button" value="Submit"/> </div> <div> <h3>Hasil Detection</h3> <p>Komentar :</p> <table border="1"> <thead> <tr> <th>Username</th> <th>Comment</th> </tr> </thead> <tbody> <tr><td>Case Folding</td><td> </td></tr> <tr><td>Cleaning</td><td> </td></tr> <tr><td>Remove Repetition</td><td> </td></tr> <tr><td>Tokonezing</td><td> </td></tr> <tr><td>Normalisasi</td><td> </td></tr> <tr><td>Stemming</td><td> </td></tr> <tr><td>Stopword Removal</td><td> </td></tr> </tbody> </table> <p> Kernel Linear : Kernel Polynomial : Kernel RBF: Probabilitas Linear : Probabilitas Polynomial : Probabilitas RBF: </p> </div>	Username	Comment	Case Folding		Cleaning		Remove Repetition		Tokonezing		Normalisasi		Stemming		Stopword Removal	
	Username	Comment															
	Case Folding																
	Cleaning																
	Remove Repetition																
	Tokonezing																
Normalisasi																	
Stemming																	
Stopword Removal																	

Gambar 3.24 Rancangan Halaman *Detection*

Gambar 3.22 adalah perancangan antarmuka dari halaman *detection* yang merupakan halaman untuk mendeteksi kalimat *hatespeech* dan *non-hatespeech*. Pada halaman ini, setelah *user* menginputkan *text* yang ingin dideteksi maka sistem akan menampilkan hasil deteksi berupa hasil proses *preprocessing* data, hasil klasifikasi serta probabilitas dari 3 model klasifikasi yaitu *Support Vector Machine* dengan kernel *linear*, kernel *polynomial*, dan kernel RBF.



Gambar 3.25 Rancangan Halaman Statistika

Gambar 3.23 Merupakan halaman statistika yang akan menampilkan visualisasi data dan perbandingan nilai akurasi, presisi, serta *recall* antara metode *Support Vector Machine* dengan kernel *linear*, metode *Support Vector Machine* dengan kernel *polynomial*, dan metode *Support Vector Machine* dengan kernel RBF.

C. Construction of Prototype

Pada tahap ini, hasil perancangan yang sudah dilakukan sebelumnya kemudian dilanjutkan dengan membangun sebuah *prototype* yang diimplementasi menggunakan bahasa pemrograman *python*.

D. Deployment Delivery & Feedback

Tahap *deployment delivery* dan *feedback* merupakan tahap terakhir pada metode pengembangan *prototype*. Pada tahap ini dilakukan pengujian dan evaluasi perangkat lunak untuk mengetahui kesiapan dari perangkat lunak yang dikembangkan. Pengujian sistem dengan menggunakan *blackbox* testing dapat dilihat pada Tabel 3.40

Tabel 3.54 Pengujian *Blackbox*

No	Nama Halaman	Pengujian	Hasil
1	<i>Dashboard</i>	User dapat mengakses sistem dan menampilkan halaman utama yaitu <i>dashboard</i>	
2	<i>Dataset</i>	User dapat mengakses halaman dataset dan melihat informasi dari seluruh dataset yang berisikan <i>username</i> , <i>comment</i> , dan <i>label</i>	

Tabel 3.55 Pengujian *Blackbox* (Lanjutan)

No	Nama Halaman	Pengujian	Hasil
3	<i>Preprocessing</i>	User dapat mengakses halaman <i>preprocessing</i> dan melihat seluruh dataset hasil <i>preprocessing</i>	
4	<i>Detection</i>	User dapat memasukkan komentar yang ingin dideteksi dan sistem dapat menampilkan hasil klasifikasi berupa hasil <i>preprocessing</i> data, hasil deteksi beserta probabilitas dari masing-masing model	
5	Statistika	User dapat mengakses halaman statistika dan melihat visualisasi dari data serta perbandingan performa dari masing-masing model	

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil Penelitian

4.1.1 Pengambilan Data

Pengumpulan data dilakukan dengan cara *scraping* data pada media sosial instagram menggunakan *library selenium* yang disediakan oleh bahasa pemrograman python. Data yang diambil berupa *username* dan komentar teks berbahasa Indonesia pada tahun 2019 sampai tahun 2022. Total keseluruhan data sebanyak 918 dengan 463 komentar *non-hatespeech* dan 455 komentar *hatespeech*. Proses *scraping* yang dilakukan dalam penelitian ini adalah sebagai berikut.

Kode Program 1: Pengambilan Data

```
# link postingan
driver.get(
    'https://www.instagram.com/p/Cm3kF51Jbam/?utm_source=ig_web_copy_link')

# Membuka seluruh data di kolom komentar
i = 0
while i < 20:
    load_more_comment=
WebDriverWait(driver,20).until(EC.element_to_be_clickable((By.XPATH, '/html/body/div[2]/div/div/div/div[1]/div/div/div/div[1]/div[1]/div[2]/section/main/div[1]/div[1]/article/div/div[2]/div/div[2]/div[1]/ul/li/div/button')))
    try:
        load_more_comment.click()
        time.sleep(3)
    except exceptions.StaleElementReferenceException as e:
        print(str(e))
        pass
    finally:
        i += 1

time.sleep(10)

# Menentukan indeks data
user_names = []
user_comments = []

# Meminta webdriver untuk menemukan elemen kolom komentar
comment = driver.find_elements(By.CLASS_NAME, '_a9zj ')

# Mengambil data username dan komentarnya
for c in comment:
    container = c.find_element(By.CLASS_NAME, '_a9zr')
    name = container.find_element(By.CLASS_NAME, '_a9zc').text
    content = container.find_element(By.CLASS_NAME, '_a9zs').text
    content = content.replace('\n', ' ').strip().rstrip()
    user_names.append(name)
    user_comments.append(content)
```

Modul Program 4.1 Pengambilan Data

4.1.2 Preprocessing

Data yang telah dilabeli kemudian dilakukan tahap *preprocessing* yang terdiri dari 7 proses yaitu *case folding*, *cleaning*, *remove repetition character*, *tokenizing*, *normalisasi*, *stemming*, dan *stopword removal* guna membersihkan serta menyeragamkan data menjadi format yang lebih sesuai sehingga siap untuk digunakan ditahap selanjutnya.

a. Case Folding

Pada proses ini menggunakan fungsi *.lower()* untuk mengubah semua huruf yang terdapat pada data komentar menjadi huruf kecil. Berikut adalah kode program dari proses *case folding* yang dilakukan.

Kode Program 2: Case Folding

```
def case_folding(text):  
    text = text.lower()  
    return text
```

Modul Program 4.2 Case Folding

b. Cleaning

Pada proses *cleaning* semua simbol, emoji, *mention*, tanda baca, *link url*, angka, dan *hashtag* pada data komentar akan dihapus dengan menggunakan fungsi *re.sub()*. Berikut adalah kode program dari proses *cleaning* yang dilakukan.

Kode Program 3: Cleaning

```
def cleaning_text(text):  
    text = re.sub(r'^\x00-\x7f', r'', text) # Remove non ASCII chars (emoji)  
    text = re.sub(r'(\u[0-9A-Fa-f]+)', r'', text)  
    text = re.sub(r'@[\w]*', ' ', text) # Remove mention handle user (@)  
    text = re.sub(r"^[A-Za-z0-9^,!.\/'+-=]", " ", text)  
    text = re.sub(r'\u[w\w\w\w\w]', '', text) # Remove link web  
    text = re.sub(r'http\S+', '', text)  
    text = re.sub(r'#([^\s]+)', '', text) # Remove #tagger  
    # Remove simbol, angka dan karakter aneh  
    text = re.sub(r"[.,;+!\\_<^/?\"'\\(\\)\\d\\*]", " ", text)  
    return text
```

Modul Program 4.3 Cleaning

c. Remove Repetition Character

Pada proses ini akan dilakukan penghapusan karakter berulah lebih dari 2 karakter dalam suatu kata pada data komentar. Berikut adalah kode program dari proses *remove repetition character* yang dilakukan.

Kode Program 4: Remove Repetition Character

```
def repetition(text):  
    pattern = re.compile(r"(\.)\1{1,}", re.DOTALL)  
    return pattern.sub(r"\1\1", text)
```

Modul Program 4.4 Remove Repetition Character

d. Tokenizing

Pada proses *tokenizing* menggunakan fungsi *word_tokenize* dari library NLTK yang dimiliki oleh Python guna memisahkan kata perkata/frasa pada data komentar. Berikut adalah kode program dari proses *tokenizing* yang dilakukan.

Kode Program 5: Tokenizing

```
def tokenize(text):  
    return word_tokenize(text)
```

Modul Program 4.5 Tokenizing**e. Normalisasi**

Pada proses ini kata tidak baku diubah menjadi kata baku serta mengganti singkatan ke dalam kosa kata sebenarnya dengan cara mencocokkan kata sesuai dengan file *dictionary* yang telah tersedia dalam bentuk txt. Berikut adalah kode program dari proses normalisasi yang dilakukan.

Kode Program 6: Normalisasi

```
def normalisasi(text):  
    kamus_normalisasi = eval(open("dictionary.txt").read())  
    pattern = re.compile(r'\b( ' + '|'.join(kamus_normalisasi.keys()) + r')\b')  
    content = []  
    for kata in text:  
        filtered = pattern.sub(lambda x: kamus_normalisasi[x.group()], kata)  
        content.append(filtered.lower())  
    text = content  
    return text
```

Modul Program 4.6 Normalisasi**f. Stemming**

Pada proses *stemming* dilakukan pengubahan kata-kata berimbuhan menjadi sebuah kata dasar yang dibantu dengan menggunakan fungsi *stemmer* dari *library* Sastrawi. Berikut adalah kode program dari proses *stemming* yang dilakukan.

Kode Program 7: Stemming

```
def stemming(text):  
    factory = StemmerFactory()  
    stemmer = factory.create_stemmer()  
    return stemmer.stem(text)
```

Modul Program 4.7 Stemming**g. Stopword Removal**

Proses *stopword removal* merupakan proses terakhir dari tahapan *preprocessing*. Pada proses ini dilakukan penghapusan kata-kata yang tidak penting/berpengaruh terhadap kalimat dengan menggunakan kamus *stopword* dari *library* Sastrawi. Berikut adalah kode program dari proses *stopword removal* yang dilakukan.

Kode Program 8: Stopword Removal

```
def remove_stopword(text, stop_words=stop_words):  
    word_tokens = word_tokenize(text)  
    filtered_sentence = [w for w in word_tokens if not w in stop_words]  
    return ' '.join(filtered_sentence)
```

Modul Program 4.8 Stopword Removal**4.1.3 Split Data**

Setelah dilakukan proses *preprocessing*, maka tahap selanjutnya yaitu dilakukan *split data* guna membagi data menjadi data *training* dan data *testing*. Pada penelitian ini, membagi data dengan perbandingan 80:20 dimana 80% adalah data *training* dengan

jumlah komentar sebanyak 734 komentar dan 20% adalah data *testing* yaitu sebanyak 184 komentar. Berikut adalah kode program dari proses *split data* yang dilakukan.

Kode Program 9: Split Data

```
x_train,x_test,y_train, y_test = train_test_split(data['Hasil_prepro'],
                                                data['label'],
                                                test_size = 0.2,
                                                random_state = 42 )
```

Modul Program 4.9 Split Data**4.1.4 Pembobotan Kata TF-IDF**

Pada penelitian ini dalam melakukan pembobotan kata *Term Frequency Inverse Document Frequency* (TF-IDF) menggunakan bantuan *library TFidfVectorizer* yang telah tersedia di bahasa pemrograman python. Berikut adalah kode program dari proses *split data* yang dilakukan.

Kode Program 10: TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf_vect = TfidfVectorizer(max_df=1.0, min_df=1, norm=None, smooth_idf=True)
train_X_tfidf = tfidf_vect.fit_transform(x_train)
tes_X_tfidf = tfidf_vect.transform(x_test)
tfidf_pca = tfidf_vect.fit_transform(data['Hasil_prepro'])
```

Modul Program 4.10 TF-IDF**4.1.5 Pembuatan Visualisasi Principal Component Analysis**

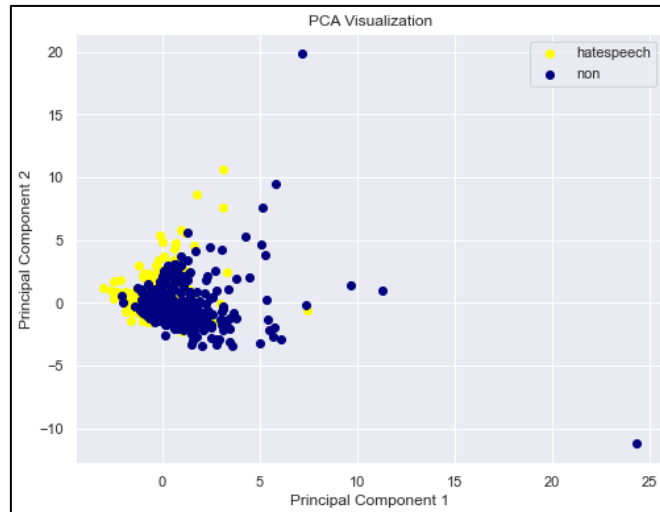
Visualisasi data pada penelitian ini menggunakan teknik reduksi *Principal Component Analysis* untuk mengetahui bahwa data yang dipakai merupakan data *non-linear*. Data yang digunakan adalah data hasil dari pembobotan kata *Term Frequency Inverse Document Frequency* (TF-IDF). *Principal Component Analysis* pada penelitian ini hanya digunakan untuk visualisasi data, yang mana hasil dari tahapan ini tidak akan dilanjutkan ke dalam tahap klasifikasi. Berikut adalah kode program dari proses pembuatan visualisasi data menggunakan *Principal Component Analysis*, sehingga menghasilkan visualisasi data seperti pada Gambar 4.1

Kode Program 11: Principal Component Analysis

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
pca_result = pca.fit_transform(tfidf_pca.toarray())
plt.figure(figsize=(8, 6))
unique_labels = np.unique(labels)
for label in unique_labels:
    mask = (labels == label)
    plt.scatter(pca_result[mask, 0], pca_result[mask, 1],
                label=label,c=color_palette[label]
    )
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('PCA Visualization Data')
plt.legend()
```

Modul Program 4.11 Principal Component Analysis



Gambar 4.1 Visualisasi Data

4.1.6 Pembuatan Model

Pembuatan model pada penelitian ini dilakukan setelah melalui tahap *preprocessing* dan pembobotan kata *Term Frequency Inverse Document Frequency* (TF-IDF) yaitu dengan menggunakan bantuan model SVC dari *library sklearn* yang telah tersedia di bahasa pemrograman python. Kode program pada metode *Support Vector Machine* kernel *linear* dan *polynomial* hanya memiliki perbedaan pada jenis kernel dan inisialisasi parameter. Pada kernel *polynomial* penelitian ini menggunakan parameter *degree=1* dan *gamma=0.001*. Berikut merupakan kode program dari proses pembuat model *Support Vector Machine* pada kernel *linear* dan *polynomial*.

Kode Program 12: Pembuatan Model

```
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix

# Support Vector Machine Kernel Linear
svm = SVC(kernel='linear', probability=True)
svm.fit(train_X_tfidf, y_train)
y_pred_svc = svm.predict(test_X_tfidf)

# Support Vector Machine Kernel Polynomial
svm_poly = SVC(kernel='poly', degree=1, gamma=0.001, probability=True)
svm_poly.fit(train_X_tfidf, y_train)
y_pred_svcp = svm_poly.predict(test_X_tfidf)

# Support Vector Machine Kernel RBF
svm_rbf = SVC(kernel='rbf', C=2, gamma=0.001, probability=True)
svm_rbf.fit(train_X_tfidf, y_train)
y_pred_svcr = svm_rbf.predict(test_X_tfidf)
```

Modul Program 4.12 Pembuatan Model

4.1.7 Pengujian Model

Setelah model dibuat, maka selanjutnya dilakukan pengujian model untuk mengetahui performa dari model yang telah dibuat, yang mana pada penelitian ini menggunakan pengujian *confussion matrix* dan pengujian *actual*.

a. Pengujian Model SVM Kernel *Linear*

Tabel 4.1 Hasil Pengujian *Confussion Matrix* SVM Kernel *Linear*

		Predicted		Jumlah
		Non-Hatespeech	Hatespeech	
Actual	Non-Hatespeech	63	21	84
	Hatespeech	21	79	100
Jumlah		85	99	184

Berikut adalah perhitungan untuk mendapatkan nilai akurasi, presisi, dan *recall* dari pengujian model menggunakan metode *Support Vector Machine* dengan kernel *linear*.

$$Accuracy = \frac{63 + 79}{63 + 79 + 21 + 21} \times 100\% = 77.17\%$$

$$Precision (+) = \frac{63}{63 + 21} \times 100\% = 0.75$$

$$Precision (-) = \frac{79}{21 + 79} \times 100\% = 0.79$$

$$Precision = \frac{0.75 + 0.79}{2} \times 100\% = 77.17\%$$

$$Recall (+) = \frac{63}{63 + 21} \times 100\% = 0.75$$

$$Recall (-) = \frac{79}{79 + 21} \times 100\% = 0.79$$

$$Recall = \frac{0.75 + 0.79}{2} \times 100\% = 77.17\%$$

Untuk pengujian model secara *actual* menggunakan metode *Support Vector Machine* dengan kernel *linear* dapat dilihat sebagai berikut.

Tabel 4.2 Hasil Pengujian *Actual* Model SVM Kernel *Linear*

Komentar	Probabilitas		Hasil Actual
	Hatespeech	Non-Hatespeech	
Cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala Lo di wawancara SMA orang dewasa belagu nya selagit Lo di ajarin sopan santun dan punya etika g sih	0.874112	0.125888	Hatespeech

b. Pengujian Model SVM Kernel *Polynomial*

Tabel 4.3 Hasil Pengujian *Confussion Matrix* SVM Kernel *Polynomial*

		Predicted		Jumlah
		Non-Hatespeech	Hatespeech	
Actual	Non-Hatespeech	72	12	84
	Hatespeech	19	81	100
Jumlah		89	95	184

Berikut adalah perhitungan untuk mendapatkan nilai akurasi, presisi, dan *recall* dari pengujian model menggunakan metode *Support Vector Machine* dengan kernel *polynomial*.

$$Accuracy = \frac{72 + 81}{72 + 81 + 19 + 12} \times 100\% = 83.15\%$$

$$Precision (+) = \frac{72}{72 + 19} \times 100\% = 0.79$$

$$Precision (-) = \frac{81}{22 + 77} \times 100\% = 0.87$$

$$Precision = \frac{0.79 + 0.87}{2} \times 100\% = 83.45\%$$

$$Recall (+) = \frac{72}{72 + 12} \times 100\% = 0.86$$

$$Recall (-) = \frac{81}{81 + 19} \times 100\% = 0.81$$

$$Recall = \frac{0.86 + 0.81}{2} \times 100\% = 83.15\%$$

Untuk pengujian model secara *actual* menggunakan metode *Support Vector Machine* dengan kernel *polynomial* dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil Pengujian *Actual* Model SVM Kernel *Polynomial*

Komentar	Probabilitas		Hasil Actual
	Hatespeech	Non-Hatespeech	
Cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala Lo di wawancara SMA orang dewasa belagu nya selagit Lo di ajarin sopan santun dan punya etika g sih	0.913759	0.086241	Hatespeech

a. Pengujian Model SVM Kernel RBF

Tabel 4.5 Hasil Pengujian *Confussion Matrix* SVM Kernel RBF

		Predicted		Jumlah
		Non-Hatespeech	Hatespeech	
Actual	Non-Hatespeech	62	22	84
	Hatespeech	14	86	100
Jumlah		76	108	184

Berikut adalah perhitungan untuk mendapatkan nilai akurasi, presisi, dan *recall* dari pengujian model menggunakan metode *Support Vector Machine* dengan kernel RBF.

$$Accuracy = \frac{62 + 86}{62 + 86 + 14 + 22} \times 100\% = 80.43\%$$

$$Precision (+) = \frac{62}{62 + 19} \times 100\% = 0.82$$

$$Precision (-) = \frac{86}{22 + 77} \times 100\% = 0.80$$

$$Precision = \frac{0.82 + 0.80}{2} \times 100\% = 80.51\%$$

$$Recall (+) = \frac{62}{62 + 22} \times 100\% = 0.74$$

$$Recall (-) = \frac{86}{86 + 14} \times 100\% = 0.86$$

$$Recall = \frac{0.74 + 0.86}{2} \times 100\% = 80.43\%$$

Untuk pengujian model secara *actual* menggunakan metode *Support Vector Machine* dengan kernel RBF dapat dilihat pada Tabel 4.4.

Tabel 4.6 Hasil Pengujian *Actual* Model SVM Kernel RBF

Komentar	Probabilitas		Hasil Actual
	Hatespeech	Non-Hatespeech	
Cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala Lo di wawancara SMA orang dewasa belagu nya selagit Lo di ajarin sopan santun dan punya etika g sih	0.874112	0.125888	Hatespeech

Tabel 4.7 Hasil Pengujian Model

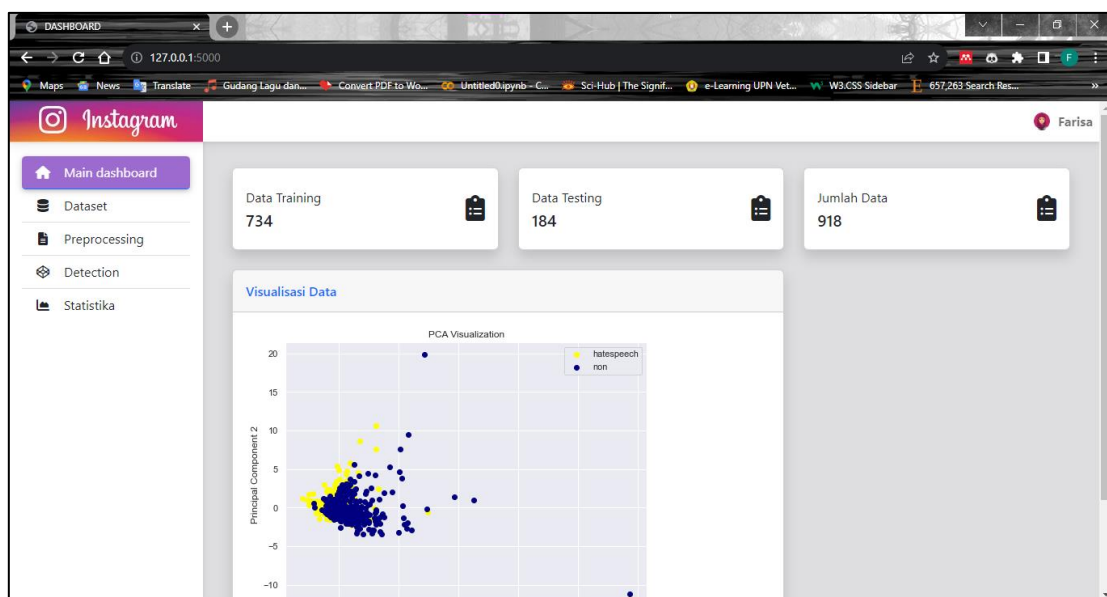
Pengujian	SVM Kernel Linear	SVM Kernel Polynomial	SVM Kernel RBF
Akurasi	77.17%	83.15%	80.43%
Presisi	77.17%	83.45%	80.51%
Recall	77.17%	83.15%	80.43%

4.1.8 Implementasi Sistem

Bagian ini menampilkan *interface* dari hasil implementasi dari rancangan sistem yang telah dibuat di bab sebelumnya yang terdiri dari halaman *Dashboard*, *Dataset*, *Preprocessing*, *Detection*, dan *Statistika*.

1. Halaman *Dashboard*

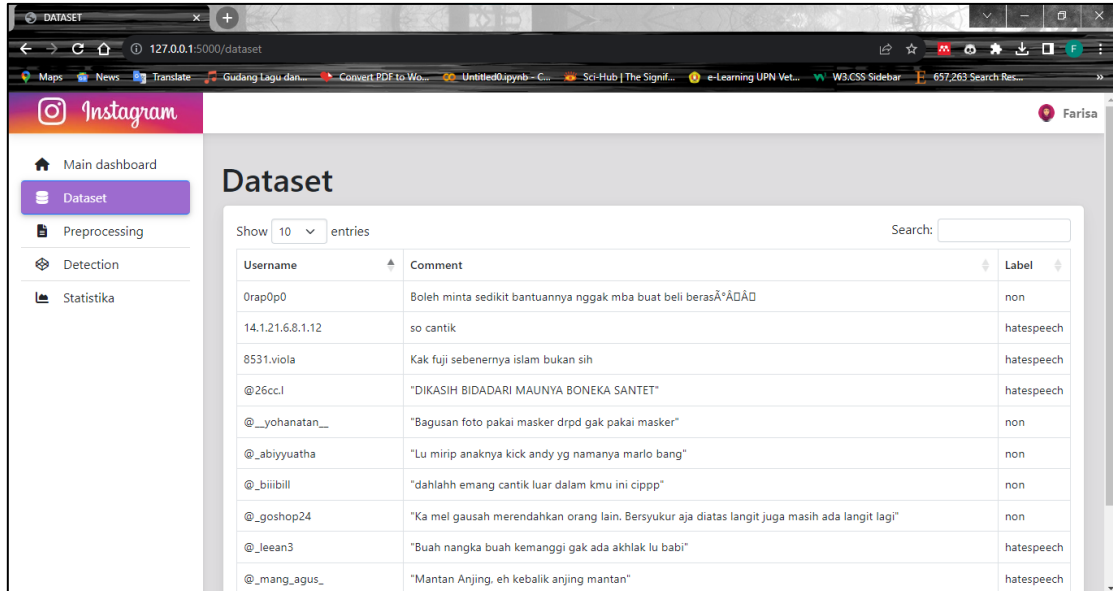
Halaman ini merupakan tampilan awal yang akan dilihat oleh *user* ketika membuka sistem ini yaitu yang berisikan jumlah data *training*, data *testing*, dan jumlah keseluruhan data yang dipakai pada penelitian ini. Selain itu, sistem juga akan menampilkan visualisasi data data yang digunakan. Berikut merupakan *interface* dari implementasi sistem pada halaman *dashboard*.



Gambar 4.2 Halaman *Dashboard*

2. Halaman Dataset

Pada halaman ini, sistem akan menampilkan tabel yang berisi dataset yang dipakai yaitu berupa *username*, *comment*, dan label. *User* dapat mengurutkan data secara *descending* dan *ascending* berdasarkan *username*, *comment*, dan label, serta dapat mencari berdasarkan kata pada data menggunakan fitur *search*. Berikut merupakan *interface* dari implementasi sistem pada halaman dataset.



Username	Comment	Label
0rap0p0	Boleh minta sedikit bantuannya nggak mba buat beli beras	non
14.1.21.6.8.1.12	so cantik	hatespeech
8531.viola	Kak fuji sebenarnya islam bukan sih	hatespeech
@26cc.I	"DIKASIH BIDADARI MAUNYA BONEKA SANTET"	hatespeech
@_yohanatan_	"Bagusan foto pakai masker drpd gak pakai masker"	non
@_abiyyuatha	"Lu mirip anaknya kick andy yg namanya marlo bang"	non
@_biibilill	"dahlahh emang cantik luar dalam kmu ini cippp"	non
@_goshop24	"Ka mel gausah merendahkan orang lain. Bersyukur aja diatas langit juga masih ada langit lagi"	non
@_leean3	"Buah nangka buah kemanggi gak ada ahlak lu babi"	hatespeech
@_mang_agus_	"Mantan Anjing, eh kebalik anjing mantan"	hatespeech

Gambar 4.3 Halaman Dataset

3. Halaman Preprocessing

Pada halaman ini, sistem akan menampilkan tabel yang berisikan hasil dari dataset yang telah melewati proses *preprocessing*. *User* dapat mengurutkan data secara *descending* dan *ascending* berdasarkan kategori di setiap kolom, serta terdapat fitur *search* yang dapat digunakan untuk mencari data berdasarkan kata pada data. Berikut merupakan *interface* dari implementasi sistem pada halaman *preprocessing*.

PREPROCESSING

127.0.0.1:5000/preprocessing

Maps News Translate Gudang Lagu dan... Convert PDF to Wo... Untitled0.ipynb - C... Sci-Hub | The Signif... e-Learning UPN Vet... W3.CSS Sidebar 657,263 Search Res...

Instagram

Farisa

Main dashboard

Dataset

Preprocessing

Detection

Statistika

Preprocessing

Show 10 entries

Search:

Username	Comment	Case Folding	Cleaning	Repitation	Tokenizing	Normalization	Stemming	Stopword
0rap0p0	Boleh minta sedikit bantuannya nggak mba buat beli beras	boleh minta sedikit bantuannya nggak mba buat beli beras	boleh minta sedikit bantuannya nggak mba buat beli beras	boleh minta sedikit bantuannya nggak mba buat beli beras	['boleh', 'minta', 'sedikit', 'bantuannya', 'nggak', 'mba', 'buat', 'beli', 'beras']	['boleh', 'minta', 'sedikit', 'bantuannya', 'tidak', 'kakak', 'bantu', 'tidak', 'kakak', 'buat', 'beli', 'beras']	['boleh', 'minta', 'sedikit', 'bantu', 'tidak', 'kakak', 'buat', 'beli', 'beras']	['boleh', 'minta', 'sedikit', 'bantu', 'tidak', 'kakak', 'buat', 'beli', 'beras']
14.1.21.6.8.1.12	so cantik	so cantik	so cantik	so cantik	['so', 'cantik']	['sok', 'cantik']	['sok', 'cantik']	['sok', 'cantik']
8531.viola	Kak fuji sebenarnya islam bukan sih	kak fuji sebenarnya islam bukan sih	kak fuji sebenarnya islam bukan sih	kak fuji sebenarnya islam bukan sih	['kak', 'fuji', 'sebenarnya', 'islam', 'bukan', 'sih']	['kakak', 'fuji', 'sebenarnya', 'islam', 'bukan', 'sih']	['kakak', 'fuji', 'benar', 'islam', 'bukan', 'sih']	['kakak', 'fuji', 'benar', 'islam', 'bukan', 'sih']

Gambar 4.4 Halaman Preprocessing

4. Halaman *Detection*

Pada halaman ini, *user* dapat menginputkan *text* yang ingin dideteksi dimana nantinya sistem akan menampilkan hasil deteksi berupa hasil proses *preprocessing* data, hasil deteksi serta nilai probabilitas dari 3 model klasifikasi yaitu *Support Vector Machine* dengan kernel *linear*, *Support Vector Machine* dengan kernel *polynomial*, dan *Support Vector Machine* dengan kernel RBF. Berikut merupakan *interface* dari implementasi sistem pada halaman *detection*.

Hatespeech Detection

input comment

Hasil

Special title treatment

Komentar : ['Cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala lo di wawancara sma orang dewasa belagu nya selagit lo di ajarin sopan santun dan punya etika g sih']

Preprocessing	Comment
Case Folding	cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala lo di wawancara sma orang dewasa belagu nya selagit lo di ajarin sopan santun dan punya etika g sih
Cleaning	cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala lo di wawancara sma orang dewasa belagu nya selagit lo di ajarin sopan santun dan punya etika g sih
Remove Repetition Character	cantik di luar tpi otak tidak di pakai dan mulut kaya tidak makan bangku sekolah gd kepala lo di wawancara sma orang dewasa belagu nya selagit lo di ajarin sopan santun dan punya etika g sih
Tokenezing	['cantik', 'di', 'luar', 'tpi', 'otak', 'tidak', 'di', 'pakai', 'dan', 'mulut', 'kaya', 'tidak', 'makan', 'bangku', 'sekolah', 'gd', 'kepala', 'lo', 'di', 'wawancara', 'sma', 'orang', 'dewasa', 'belagu', 'nya', 'selagit', 'lo', 'di', 'ajarin', 'sopan', 'santun', 'dan', 'punya', 'etika', 'g', 'sih']
Normalization	['cantik', 'di', 'luar', 'tetapi', 'otak', 'tidak', 'di', 'pakai', 'dan', 'mulut', 'kaya', 'tidak', 'makan', 'bangku', 'sekolah', 'gd', 'kepala', 'kamu', 'di', 'wawancara', 'sama', 'orang', 'dewasa', 'belagu', 'nya', 'selagit', 'kamu', 'di', 'ajarin', 'sopan', 'santun', 'dan', 'punya', 'etika', 'tidak', 'sih']
Stemming	['cantik', 'di', 'luar', 'tetapi', 'otak', 'tidak', 'di', 'pakai', 'dan', 'mulut', 'kaya', 'tidak', 'makan', 'bangku', 'sekolah', 'gd', 'kepala', 'kamu', 'di', 'wawancara', 'sama', 'orang', 'dewasa', 'belagu', 'nya', 'selagit', 'kamu', 'di', 'ajarin', 'sopan', 'santun', 'dan', 'punya', 'etika', 'tidak', 'sih']
Stopword Removal	['cantik', 'luar', 'otak', 'tidak', 'pakai', 'mulut', 'kaya', 'tidak', 'makan', 'bangku', 'sekolah', 'gd', 'kepala', 'kamu', 'wawancara', 'sama', 'orang', 'dewasa', 'belagu', 'selagit', 'kamu', 'ajarin', 'sopan', 'santun', 'punya', 'etika', 'tidak', 'sih']

Kernel Linear : Hatespeech

Kernel Polynomial : Hatespeech

Kernel RBF : Hatespeech

Probabilitas Linear: [non-hatespeech : 0.125888] | [hatespeech : 0.874112]

Probabilitas Polynomial: [non-hatespeech : 0.086241] | [hatespeech : 0.913759]

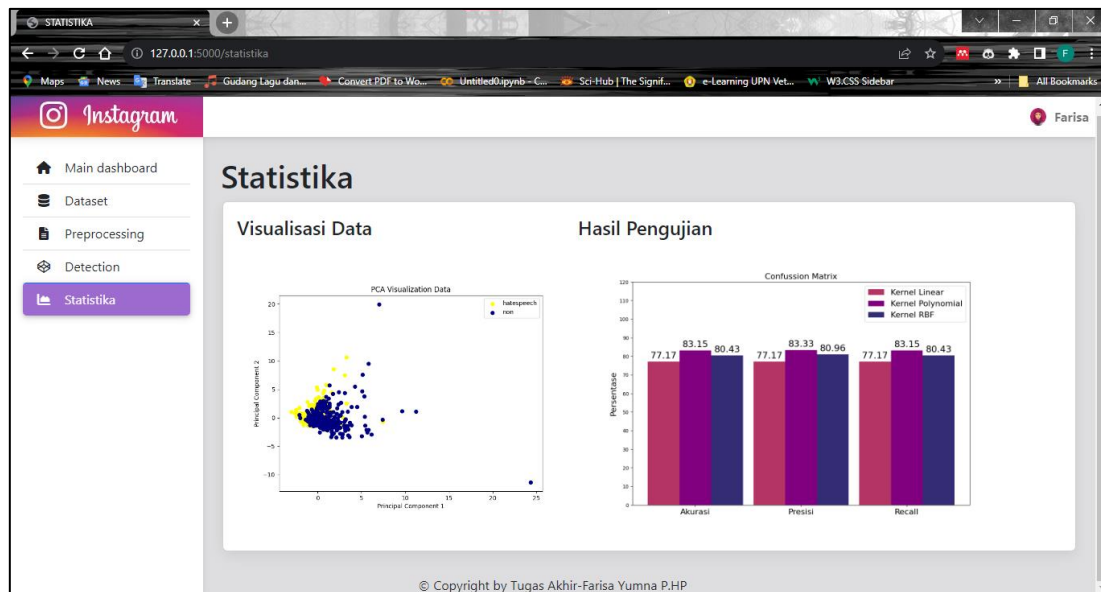
Probabilitas RBF: [non-hatespeech : 0.125888] | [hatespeech : 0.874112]

© Copyright by Tugas Akhir-Farisa Yumna P,HP

Gambar 4.5 Halaman *Detection*

5. Halaman *Statistika*

Pada halaman ini, sistem akan menampilkan hasil visualisasi data dan perbandingan dari nilai akurasi, presisi, serta *recall* antara metode *Support Vector Machine* dengan kernel *linear* dan metode *Support Vector Machine* dengan kernel *polynomial*. Berikut merupakan *interface* dari implementasi sistem pada halaman statistika.



Gambar 4.6 Halaman Statistika

4.1.9 Pengujian Sistem

Pengujian sistem menggunakan *blackbox* testing dapat dilihat pada Tabel 4.8 adalah sebagai berikut.

Tabel 4.8 Hasil Pengujian *Blackbox*

No	Nama Halaman	Pengujian	Hasil
1	<i>Dashboard</i>	User dapat mengakses sistem dan menampilkan halaman utama yaitu <i>dashboard</i>	Berhasil
2	<i>Dataset</i>	User dapat mengakses halaman dataset dan melihat informasi dari seluruh dataset yang berisikan <i>username</i> , <i>comment</i> , dan <i>label</i>	Berhasil
3	<i>Preprocessing</i>	User dapat mengakses halaman <i>preprocessing</i> dan melihat seluruh dataset hasil <i>preprocessing</i>	Berhasil
4	<i>Detection</i>	User dapat memasukkan komentar yang ingin dideteksi dan sistem dapat menampilkan hasil klasifikasi berupa hasil <i>preprocessing</i> data, hasil deteksi beserta probabilitas dari masing-masing model	Berhasil
5	<i>Statistika</i>	User dapat mengakses halaman statistika dan melihat visualisasi dari data serta perbandingan performa dari masing-masing model	Berhasil

4.2 Pembahasan

Berdasarkan implementasi dan pengujian yang telah dilakukan, diperoleh hasil bahwa metode *Support Vector Machine* menggunakan kernel *polynomial* memiliki performa yang lebih baik dibandingkan kernel *linear* dan kernel RBF dalam melakukan identifikasi data *non-linear* berjumlah 918 komentar yang terdiri dari 463 *non-hatespeech* dan 455 *hatespeech* dengan proses *split* data *training* dan *testing* 80:20, serta melalui tujuh proses *preprocessing* yaitu *case folding*, *cleaning*, *remove repetation character*, *tokenizing*, *normalisasi*, *stemming*, dan *stopword removal*. Selain itu, penentuan nilai parameter *C*, *degree*, dan *gamma* juga memiliki pengaruh dalam meningkatkan performa pada model. Nilai *C* dan *gamma* harus lebih besar dari nol sehingga rentang dari parameter *C* dan *gamma* berada dalam *interval* $(0, \infty)$ (Prangga, 2017). Semakin besar nilai parameter *C*

maka *margin* akan menjadi lebih kecil, sehingga memberikan penalti yang lebih besar terhadap *error* klasifikasi pada model dan semakin besar nilai *gamma* maka semakin tajam pula distribusi kernel dan semakin fokus pada titik data yang terdekat. Sedangkan untuk nilai *degree* berdasarkan *scikit-learn 1.3.1 documentation* mengenai *sklearn.svm.SVC* dinyatakan bahwa nilai *degree* harus *non-negative*, yang mana semakin besar nilai *degree* maka semakin melengkung garis *hyperplane* yang dihasilkan.

Penentuan nilai parameter *C*, *gamma*, dan *degree* dalam penelitian ini didasarkan pada penelitian yang telah dilakukan oleh Zaiem & Charibaldi (2021) dan hasil observasi empiris Huang et al (2007) yang menyatakan bahwa nilai *C* paling tepat untuk *Support Vector Machine* adalah antara 10^{-2} hingga 10^4 . Penelitian ini melakukan percobaan nilai parameter *C* sebesar 0.5, 1, dan 2 dengan nilai *gamma* yaitu 0.1, 0.01, dan 0.001 serta nilai *degree* yaitu 1, 2, dan 3 untuk model kernel *polynomial*. Hasil percobaan nilai parameter pada kernel *polynomial* dapat dilihat pada tabel berikut.

Tabel 4.9 Hasil Percobaan Nilai Parameter Pada Kernel *Polynomial*

C	degree	gamma	Persentase (%)		
			Akurasi	Presisi	Recall
0,1	1	0.1	77.17	77.17	77.17
0,1	1	0.01	83.15	83.45	83.15
0,1	1	0.001	45.65	20.84	45.65
0,1	2	0.1	72.82	73.27	72.82
0,1	2	0.01	77.17	78.38	77.17
0,1	2	0.001	45.65	20.84	45.65
0,1	3	0.1	62.5	67.01	62.5
0,1	3	0.01	59.78	70.46	59.78
0,1	3	0.001	45.65	20.84	45.65
1	1	0.1	73.91	73.91	73.91
1	1	0.01	77.17	77.17	77.17
1	1	0.001	83.15	83.45	83.15
1	2	0.1	69.02	69.33	69.02
1	2	0.01	74.45	74.91	74.45
1	2	0.001	45.65	20.84	45.65
1	3	0.1	60.32	66.27	60.32
1	3	0.01	62.5	68.04	62.5
1	3	0.001	45.65	20.84	45.65
2	1	0.1	73.36	73.47	73.36
2	1	0.01	76.08	76.14	76.08
2	1	0.001	82.60	82.65	82.60
2	2	0.1	69.02	69.33	69.02
2	2	0.01	75	75.78	75
2	2	0.001	45.65	20.84	45.65
2	3	0.1	60.32	66.27	60.32
2	3	0.01	63.58	69.16	63.58
2	3	0.001	45.65	20.84	45.65

Dari tabel 4.9 dapat dilihat bahwa *Support Vector Machine* dengan kernel *polynomial* pada parameter *C* = 1, parameter *degree* = 1, parameter *gamma* = 0.001, dan parameter *C* = 0.1, parameter *degree* = 1, parameter *gamma* = 0.01 memiliki hasil akurasi, presisi, dan *recall* yang sama yaitu memiliki akurasi 83.15%, presisi 83.45% dan *recall* 83.15%. Namun untuk nilai parameter *C* = 0.1, *degree* = 1, *gamma* = 0.01 memiliki hasil

presisi negatif dan *recall* positif lebih besar dibandingkan menggunakan parameter $C = 1$, $degree = 1$, $gamma = 0.001$. nilai parameter $C = 0.1$, $degree = 1$, $gamma = 0.01$ mendapatkan hasil presisi negatif sebesar 0.87 dan *recall* positif 0.86, sedangkan untuk parameter $C = 1$, $degree = 1$, $gamma = 0.001$ memperoleh hasil lebih rendah yaitu presisi negatif sebesar 0.85 dan *recall* positif 0.83. Sehingga kernel *polynomial* mencapai performa tertinggi dengan nilai parameter $C = 0.1$, $degree = 1$, dan $gamma = 0.01$. Kernel RBF melakukan percobaan nilai parameter C dan $gamma$ yang sama dengan kernel *polynomial* memiliki hasil performa yang baik juga namun lebih rendah dari kernel *polynomial*. *Support Vector Machine* dengan kernel RBF memiliki performa tertinggi pada parameter $C = 2$, dan parameter $gamma = 0.001$ dengan akurasi sebesar 80.43%, presisi 80.51% dan *recall* 80.43%. Berikut adalah hasil percobaan dari nilai parameter pada kernel RBF.

Tabel 4.10 Hasil Percobaan Nilai Parameter Pada Kernel RBF

C	gamma	Persentase (%)		
		Akurasi	Presisi	Recall
0,1	0.1	45.65	20.84	45.65
0,1	0.01	48.91	75.89	48.91
0,1	0.001	45.65	20.84	45.65
1	0.1	46.19	75.30	46.19
1	0.01	70.65	71.17	70.65
1	0.001	79.89	80.26	79.89
2	0.1	46.19	75.30	46.19
2	0.01	72.28	73.50	72.28
2	0.001	80.43	80.51	80.43

Sementara itu, kernel *linear* melakukan percobaan nilai parameter C sebesar 0.01, 0.1, 1 dan 2 menghasilkan performa yang lebih rendah lagi dibandingkan dengan kernel *polynomial* dan RBF. Kernel *linear* memiliki performa tertinggi pada nilai parameter $C=0.01$ yaitu akurasi 77.17%, presisi 77.17%, dan *recall* 77.17% yang dapat dilihat pada tabel berikut.

Tabel 4.11 Hasil Percobaan Nilai Parameter Pada Kernel *Linear*

C	Presentase (%)		
	Akurasi	Presisi	Recall
0.01	77.17	77.17	77.17
0.1	73.91	73.91	73.91
1	72.82	72.97	72.82
2	72.82	72.97	72.82

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat ditarik kesimpulan sebagai berikut.

1. Dari hasil pengujian, penggunaan kernel *polynomial* pada penelitian ini dalam mengidentifikasi data *non-linear* dapat mempengaruhi nilai performa dari metode *Support Vector Machine*. Metode *Support Vector Machine* dengan menggunakan kernel *polynomial* pada 463 komentar *non-hatespeech* dan 455 komentar *hatespeech* memiliki hasil performa yang lebih tinggi dibandingkan menggunakan kernel *linear* yaitu terjadi peningkatan sekitar 5% dan kernel RBF sekitar 2%.
2. Kernel *polynomial* dalam mengidentifikasi *hatespeech* berbahasa Indonesia pada komentar Instagram menghasilkan performa terbaik dengan nilai parameter $C=0.1$, $degree=1$, dan $gamma=0.01$ yaitu akurasi 83.15%, presisi 83.45% dan *recall* sebesar 83.15%. Kernel RBF diperoleh performa terbaik pada nilai parameter $C=2$ dan $gamma=0.001$ yaitu akurasi 80.43%, presisi 80.51%, dan *recall* 80.43%. Sementara kernel *linear* performa terbaik didapatkan dengan nilai parameter $C=0.01$ yaitu akurasi, presisi, dan *recall* sebesar 77.17%. Ini menunjukkan bahwa model *polynomial* mampu mengidentifikasi *hatespeech* dengan lebih baik dan memiliki kemampuan dalam menghindari kesalahan *false positive* dan *false negative* dibandingkan dengan model kernel *linear* dan kernel RBF. Oleh karena itu, dalam penelitian ini kernel *polynomial* adalah pilihan yang lebih baik untuk identifikasi *hatespeech*. Namun selain kernel, penentuan nilai pada parameter juga dapat mempengaruhi performa dari model.

5.2 Saran

Penelitian ini masih terdapat beberapa kekurangan dan keterbatasan, adapun saran yang dapat digunakan untuk penelitian selanjutnya diantara lain sebagai berikut.

1. Melakukan identifikasi lebih dari dua kelas dengan menggunakan pendekatan *multiclass*, seperti *one vs one* atau *one vs all*
2. Melakukan identifikasi menggunakan kernel *non-linear* lainnya, seperti *sigmoid*
3. Melakukan visualisasi data menggunakan teknik selain *Principal Component Analysis*

DAFTAR PUSTAKA

- Abro, S., Shaikh, S., Ali, Z., Khan, S., Mujtaba, G., & Khand, Z. H. (2020). Automatic hate speech detection using machine learning: A comparative study. *International Journal of Advanced Computer Science and Applications*, 11(8), 484–491. <https://doi.org/10.14569/IJACSA.2020.0110861>
- Arifin, N., Enri, U., & Sulistiyowati, N. (2021). Penerapan Algoritma Support Vector Machine (SVM) dengan TF-IDF N-Gram untuk Text Classification. *STRING (Satuan Tulisan Riset Dan Inovasi Teknologi)*, 6(2).
- Bhavsar, H., & Panchal, M. H. (2012). A Review on Support Vector Machine for Data Classification. *International Journal of Advanced Research in Computer Engineering & Technology*, 1(10), 2278–1323.
- Chu, Y. X., Liu, X. G., & Gao, C. H. (2011). Multiscale models on time series of silicon content in blast furnace hot metal based on Hilbert-Huang transform. *Proceedings of the 2011 Chinese Control and Decision Conference, CCDC 2011*, 842–847. <https://doi.org/10.1109/CCDC.2011.5968300>
- Cindo, M., Rini, D. P., & Ermatita, E. (2019). Literatur Review: Metode Klasifikasi Pada Sentimen Analisis. *Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)*, 1(1), 66–70. <https://prosiding.seminar-id.com/index.php/sainteks/article/view/124>
- Diani, R. (2017). Analisis Pengaruh Kernel Support Vector Machine (SVM) pada Klasifikasi Data Microarray untuk Deteksi Kanker. *Indonesian Journal on Computing (Indo-JC)*, 2(1), 109. <https://doi.org/10.21108/INDOJC.2017.2.1.169>
- Drajana, I. C. R. (2017). Metode Support Vector Machine Dan Forward Selection Prediksi Pembayaran Pembelian Bahan Baku Kopra. *ILKOM Jurnal Ilmiah*, 9(2), 116–123. <https://doi.org/10.33096/ilkom.v9i2.134.116-123>
- Fu, Z., Robles-Kelly, A., & Zhou, J. (2010). Mixing linear SVMs for nonlinear classification. *IEEE Transactions on Neural Networks*, 21(12), 1963–1975. <https://doi.org/10.1109/TNN.2010.2080319>
- Han, J., Kamber, M., & Pei, J. (2011). *Data Mining. Concepts and Techniques, 3rd Edition (The Morgan Kaufmann Series in Data Management Systems)*.
- Hediyati, D., & Suartana, I. M. (2021). Penerapan Principal Component Analysis (PCA) Untuk Reduksi Dimensi Pada Proses Clustering Data Produksi. *Journal Information Engineering and Educational Technology*, 05, 49–54.
- Hendriyanto, M. D., Ridha, A. A., & Enri, U. (2022). Analisis Sentimen Ulasan Aplikasi Mola Pada Google Play Store Menggunakan Algoritma Support Vector Machine. *INTECOMS: Journal of Information Technology and Computer Science*, 5(1), 1–7. <https://doi.org/10.31539/INTECOMS.V5I1.3708>
- Hilmiah, F. (2017). Prediksi Kinerja Mahasiswa Menggunakan Support Vector Machine untuk Pengelola Program Studi di Perguruan Tinggi (Studi Kasus: Program Studi

- Magister Statistika ITS). *Departemen Manajemen Teknologi Bidang Keahlian Manajemen Teknologi Informasi Fakultas Bisnis Dan Manajemen Teknologi Institut Teknologi Sepuluh Nopember Surabaya*, 1–99. <http://repository.its.ac.id/46712/>
- Huang, C.-M., Lee, Y.-J., Lin, D. K. J., & Huang, S.-Y. (2007). Model Selection for Support Vector Machines via Uniform Design. *Computational Statistics & Data Analysis*, 52(1), 335–346. <https://doi.org/10.1016/J.CSDA.2007.02.013>
- Husni, N. (2022). *Sentimen Analisis Pada Komentar Instagram Selebgram dan Influencer Terkait Pariwisata di Yogyakarta Menggunakan Metode Lexicon Based dan SVM*. Universitas Pembagunan Nasional “Veteran” Yogyakarta.
- Ikanovrianti. (2021). *Analisis Sentimen Pada Komentar Pendek Tentang Idol Group BTS dengan Metode Support Vector Machine (SVM) dan Query Expansion (QE)*. Universitas Pembagunan Nasional “Veteran” Yogyakarta.
- Ivan, Y., ... P. A.-T. I. dan I. K. e, & 2019, undefined. (2019). Klasifikasi Hate Speech Berbahasa Indonesia di Twitter Menggunakan Naive Bayes dan Seleksi Fitur Information Gain dengan Normalisasi Kata. *Researchgate.Net*. https://www.researchgate.net/profile/Yuita-Arum-Sari/publication/334194823_Klasifikasi_Hate_Speech_Berbahasa_Indonesia_di_Twitter_Menggunakan_Naive_Bayes_dan_Seleksi_Fitur_Information_Gain_dengan_Normalisasi_Kata/links/5d1c8008a6fdcc2462bb3fbe/Klasifikasi
- Kaur, H., Mangat, V., & Nidhi. (2017). A survey of sentiment analysis techniques. *Proceedings of the International Conference on IoT in Social, Mobile, Analytics and Cloud, I-SMAC 2017*, 921–925. <https://doi.org/10.1109/I-SMAC.2017.8058315>
- Kurniawan, B., Fauzi, M. A., & Widodo, A. W. (2017). Klasifikasi Berita Twitter Menggunakan Metode Improved Naïve Bayes. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, 1(10), 1193–1200.
- Liang, S., & Kusnadi, R. (2021). Comparative Analysis of SVM, XGBoost and Neural Network on Hate Speech Classification. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 5(5), 896–903. <https://doi.org/10.29207/RESTI.V5I5.3506>
- Lyrawati, D. P. N. (2019). Deteksi Ujaran Kebencian pada Twitter Menjelang Pilpres 2019 dengan Machine Learning. *Jurnal Ilmiah Matematika*, 7(3).
- Mahmoodi, D., Soleimani, A., Khosravi, H., & Taghizadeh, M. (2011). FPGA Simulation of Linear and Nonlinear Support Vector Machine. *Journal of Software Engineering and Applications*, 04(05), 320–328. <https://doi.org/10.4236/JSEA.2011.45036>
- Manalu, B. U. (2014). Analisis Sentimen pada Twitter Menggunakan Text Mining. In *Skripsi*.
- Mandala, R. (2006). Evaluasi Kinerja Sistem Penyaringan Informasi Model Ruang Vektor. *Seminar Nasional Aplikasi Teknologi Informasi (SNATI)*. <https://journal.uui.ac.id/Snati/article/view/1547>
- Mantik, J., Mirah, I. A., Dewi, C., Gede, I., Gunadi, A., & Indrawan, G. (2022). Gamelan

- Rindik Classification Based On Mood Using K-Nearest Neighbor Method. *Jurnal Mantik*, 6(2), 1693–1702. <https://doi.org/10.35335/MANTIK.V6I2.2592>
- Mustikasari, D., Widaningrum, I., Arifin, R., & Putri, W. H. E. (2021). Comparison of Effectiveness of Stemming Algorithms in Indonesian Documents. *Proceedings of the 2nd Borobudur International Symposium on Science and Technology (BIS-STE 2020)*, 203, 154–158. <https://doi.org/10.2991/aer.k.210810.025>
- Ningrum, H. C. S. (2018). *Perbandingan Metode Support Vector Machine (SVM) Linear, Radial Basis Function (RBF), dan Polinomial Kernel dalam Klasifikasi Bidang Studi Lanjut Pilihan Alumni*.
- Nugroho, A. S., Witarto, A. B., & Handoko, D. (2003). *Support Vector Machine-Teori dan Aplikasinya dalam Bioinformatika I*. <http://asnugroho.net>
- Olive, I., Putra, D., Rega Prilianti, K., Lucky, P., & Irawan, T. (2020). Implementasi Text Mining untuk Analisis Masyarakat Terhadap Kinerja Layanan Transportasi Online Dengan Analisis Faktor. *Jurnal Simantec*, 8(2).
- Pardede, J., Miftahuddin, Y., & Kahar, W. (2020). Deteksi Komentar Cyberbullying Pada Media Sosial Berbahasa Inggris Menggunakan Naïve Bayes Classification. *Jurnal Informatika*, 7(1), 46–54. <https://doi.org/10.31294/JI.V7I1.6920>
- Prangga, S. (2017). *Optimasi Parameter Pada Support Vector Machine Menggunakan Pendekatan Metode Taguchi Untuk Data High-Dimensional*. Institut Teknologi Sepuluh Nopember.
- Pratama, A., Wihandika, R. C., & Ratnawati, D. E. (2018). Implementasi Algoritme Support Vector Machine (SVM) untuk Prediksi Ketepatan Waktu Kelulusan Mahasiswa. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 2(4), 1704–1708. <https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/1351>
- Priyambodo, L., Fuadi, H. L., Nazhifah, N., Huzaimi, I., Prawira, A. B., Saputri, T. E., Afandi, M. A., Nugraha, E. S., Wicaksono, A., & Goran, P. K. (2022). Klasifikasi Kematangan Tanaman Hidroponik Pakcoy Menggunakan Metode SVM. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 6(1), 153–160. <https://doi.org/10.29207/RESTI.V6I1.3828>
- Pusean, N. V., Charibaldi, N., & Santosa, B. (2023). *Comparison of Scenario Pre-processing Performance on Support Vector Machine and Naïve Bayes Algorithms for Sentiment Analysis*. 8(1), 57–63.
- Rachmat, A., #1, C., Sudiarto, W., #2, R., & Lukito, Y. (2019). JEPIN (Jurnal Edukasi dan Penelitian Informatika) Firefox Extension untuk Klasifikasi Komentar Spam pada Instagram Berbasis REST Services. *JEPIN (Jurnal Edukasi Dan Penelitian Informatika)*, 5(2), 146–156. <https://jurnal.untan.ac.id/index.php/jepin/article/view/33010>
- Rahat, A. M., Kahir, A., & Masum, A. K. M. (2020). Comparison of Naive Bayes and SVM Algorithm based on Sentiment Analysis Using Review Dataset. *Proceedings of the 2019 8th International Conference on System Modeling and Advancement in*

- Rahman, O. H., Abdillah, G., & Komarudin, A. (2021). Klasifikasi Ujaran Kebencian pada Media Sosial Twitter Menggunakan Support Vector Machine. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 5(1), 17–23.
<https://doi.org/10.29207/RESTI.V5I1.2700>
- Setiawan, A., Sembiring, I., Informasi, U. S., & Komputer, T. (2021). *Deteksi Rumusan Teks Kritik dan Ujaran Kebencian di Jejaring Sosial Online Twitter for Kerangka Sistem Rekomendasi Semantik*. 203, 294–301.
- Setiyono, A., & Pardede, H. F. (2019). Klasifikasi Sms Spam Menggunakan Support Vector Machine. *Jurnal Pilar Nusa Mandiri*, 15(2), 275–280.
<https://doi.org/10.33480/pilar.v15i2.693>
- Sianturi, M., Adiwijaya, A., & Faraby, S. (2017). Klasifikasi Dokumen Menggunakan Kombinasi Algoritma Principal Component Analysis Dan Svm. *EProceedings of Engineering*, 4(3).
<https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/5267>
- Susilowati, E., Sabariah, M. K., & Gozali, A. A. (2015). Impelentasi Metode Support Vector Machine Untuk Melakukan Klasifikasi Kemacetan Lalu Lintas Pada Twitter. *EProceedings of Engineering*, 2(1).
<https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/2578>
- Tanjung, G. (2018). Analisis Perbandingan Metode Klasifikasi Non-Linear pada Data Balance Acale Weight and Distance. *Final Project Data Mining*.
https://www.academia.edu/en/38012158/ANALISIS_PERBANDINGAN_METODE_KLASIFIKASI_NON_LINIER_PADA_DATA_BALANCE_SCALE_WEIGHT_AND_DISTANCE
- Vogel, I., & Meghana, M. (2021). Profiling Hate Speech Spreaders on Twitter: SVM vs. Bi-LSTM. *Ceur-Ws.Org*. <http://ceur-ws.org/Vol-2936/paper-196.pdf>
- Zaiem, A., & Charibaldi, N. (2021). *Komparasi Fungsi Kernel Metode Support Vector Machine untuk Analisis Sentimen Instagram dan Twitter (Studi Kasus : Komisi Pemberantasan Korupsi)*. 9(2), 33–42.

LAMPIRAN

Lampiran 1 Hasil Pelabelan Manual oleh Guru Bahasa Indonesia

HASIL PELABELAN MANUAL

No	Comment	Hasil Label Ahli
1	Putriiiii sadar ga si Lo itu nggak cantik	Hatespeech
2	Si paling bener si paling merasa tersakiti si anak durhaka , kebanyKN drama Mulu,artis tidak dikenal ya ini	Hatespeech
3	Jijik liatnya tukang prank gk gunaðŸ™, ,	Hatespeech
4	Gosah nongol nongol lagi di tv udah paling bagus si buat lo, biar berkurang orang kaya lo sliweran di tv. muak ngeliat tingkah lo yang kaya bocah nangis nangis ngejjik in	Hatespeech
5	Segitunya sih @kikysaputrii niruin @lida_selfi07 ,dasar kucing garong ðŸ™, ,	Hatespeech
6	Gak pantes bgt jd model ðŸ™”	Hatespeech
7	Jika tidak bisa berkata baik, maka akan lebih baik diam! ðŸ™™, setiap manusia punya kurang dan lebih, ðŸ™	Non-Hatespeech
8	Alhamdulillah begini lah hubungan jika masih bisa di perbaiki pasti akan membuat hati senang ðŸ™•	Non-Hatespeech
9	Ternyata bisa nyanyi juga Kereeenn parah niih. ðŸ™,ðŸ™‘ðŸ™”¥	Non-Hatespeech
10	Kecanduan adsens lebih berbahaya dari adiksi narkoba	Non-Hatespeech
11	Alangkah lebih baik konten2nya yg bermanfaat buat oranglain ðŸ™	Non-Hatespeech
12	Moana masih kecil udah banyak aktivitasnya ya,,, seru lihat nya anaknya ceriaâ••. Tumbuh Kembangnya pesat ðŸ™‘ðŸ™‘ðŸ™‘•. Sehat selalu Moana ðŸ™	Non-Hatespeech

Bandarlampung, 26 Februari 2023
Mengetahui,



Handayani, M. Pd.
NIP. 18720821 199802 2 004