

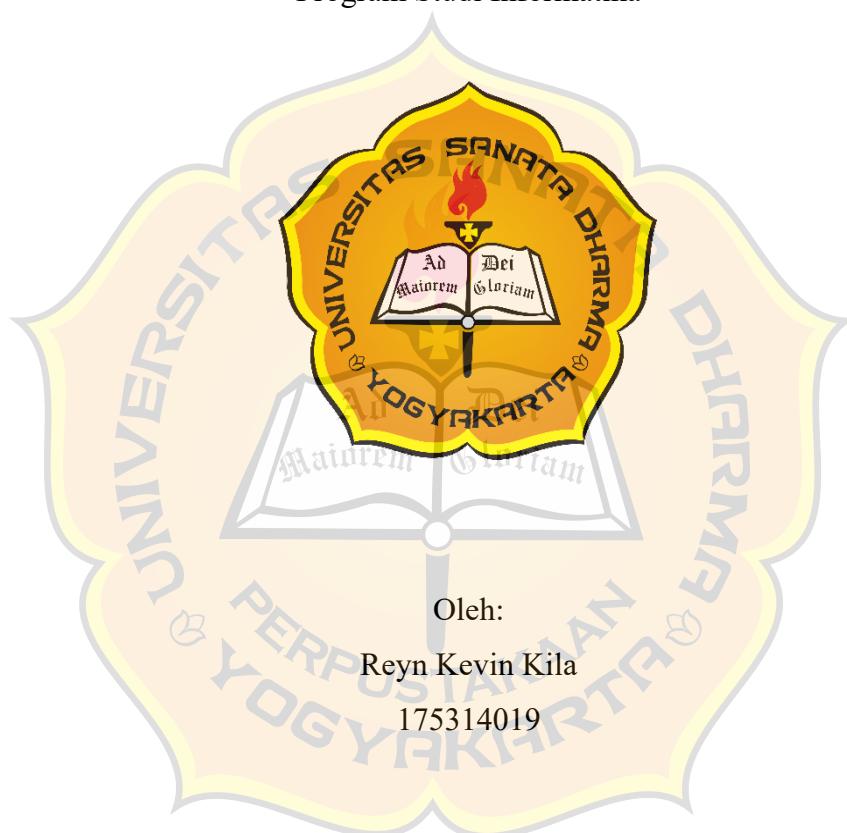
**ANALISIS SENTIMEN ULASAN GAME MOBILE
LEGENDS DI GOOGLE PLAY STORE
MENGGUNAKAN ALGORITMA SVM**

SKRIPSI

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

Program Studi Informatika



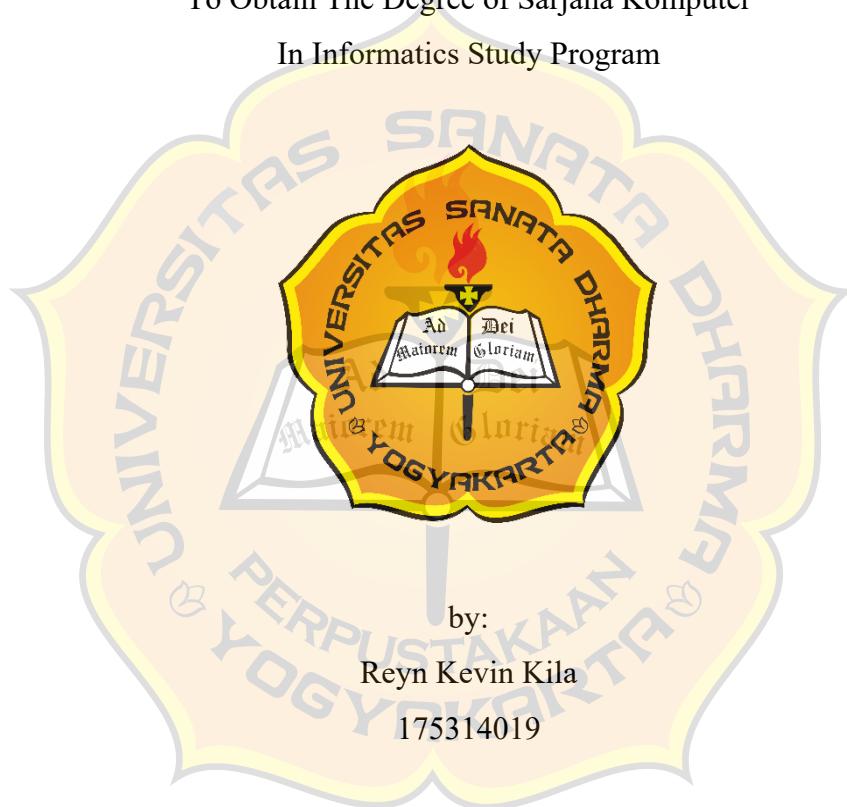
**PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS SANATA DHARMA
YOGYAKARTA**

2022

SENTIMENT ANALYSIS OF MOBILE LEGENDS GAME REVIEWS ON GOOGLE PLAY STORE USING SVM ALGORITHM

THESIS

Present as Partial Fullfillment of The Requirements
To Obtain The Degree of Sarjana Komputer
In Informatics Study Program



**INFORMATICS STUDY PROGRAM
FACULTY OF SCIENCE AND TECHNOLOGY
SANATA DHARMA UNIVERSITY
YOGYAKARTA**

2022

S k r i p s i

**ANALISIS SENTIMEN ULASAN GAME MOBILE LEGENDS
DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA**



Telah Disetujui Oleh :

Pembimbing

Ir. Kartono Pinaryanto, S. T., M. Cs.

Tanggal : 26 Januari 2023

Skripsi

ANALISIS SENTIMEN ULASAN GAME MOBILE LEGENDS DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA SVM

Dipersiapkan dan ditulis oleh:

Reyn Kevin Kila

NIM : 175314019

Telah dipertahankan di depan Dewan Pengaji
Pada Tanggal 16 januari 2023
dan dinyatakan memenuhi syarat

Susunan Dewan Pengaji

Nama Lengkap

Ketua
Agnes Maria Polina S.Kom., M.Sc.

Sekretaris
Ir. Robertus Adi Nugroho, M.Eng.

Anggota
Ir. Kartono Pinaryanto, S. T., M. Cs.

Tanda Tangan

Yogyakarta, 26 Januari 2023

Fakultas Sains dan Teknologi
Universitas Sanata Dharma

Dekan



Ir. Drs. Haris Sriwindono, M. Kom., Ph.D.

HALAMAN KUTIPAN

“Semuanya mungkin tidak akan pernah menjadi Ok. Tapi kita harus mencobanya.”

- Valdimir Vladimirovich Putin -





**UNIVERSITAS SANATA DHARMA
FAKULTAS SAINS DAN TEKNOLOGI
PROGRAM STUDI INFORMATIKA**

PERNYATAAN KEASLIAN KARYA TULIS SKRIPSI

Yang bertanda tangan di bawah ini, saya menyatakan bahwa Skripsi dengan judul:

Nama : Reyn Kevin Kila

NIM : 175314019

Program Studi : Informatika

Judul : Analisis Sentimen Ulasan Game Mobile Legends Di Google Play
Store Menggunakan Algoritma *SVM*

dan diajukan untuk diuji pada tanggal 16 Januari 2023 adalah hasil karya saya.

Dengan ini saya menyatakan dengan sesungguhnya bahwa dalam skripsi ini tidak terdapat keseluruhan atau sebagian tulisan orang lain yang saya ambil dengan cara menyalin, atau meniru dalam bentuk rangkaian kalimat atau simbol yang menunjukkan gagasan atau pendapat atau pemikiran dari penulis lain yang saya aku seolah-olah sebagai tulisan saya sendiri dan atau tidak terdapat bagian atau keseluruhan tulisan yang saya salin, tiru, atau yang saya ambil dari tulisan orang lain tanpa memberikan pengakuan pada penulis aslinya.

Apabila saya melakukan hal tersebut di atas, baik sengaja maupun tidak, dengan ini saya menyatakan menarik skripsi yang saya ajukan sebagai hasil tulisan saya sendiri ini. Bila kemudian terbukti bahwa saya ternyata melakukan tindakan menyalin atau meniru tulisan orang lain seolah-olah hasil pemikiran saya sendiri, berarti gelar dan ijazah yang telah diberikan oleh universitas batal saya terima.

Yogyakarta, 26 Januari 2023

Yang membuat pernyataan,



(Reyn Kevin Kila)

**LEMBAR PERNYATAAN PERSETUJUAN
PUBLIKASI KARYA ILMIAH UNTUK KEPENTINGAN AKADEMIS**

Yang bertanda tangan di bawah ini, saya mahasiswa Universitas Sanata Dharma :

Nama : Reyn Kevin Kila

Nomor Mahasiswa : 175314019

Demi pengembangan ilmu pengetahuan, saya memberikan kepada Perpustakaan Universitas Sanata Dharma karya ilmiah saya yang berjudul :

**ANALISIS SENTIMEN ULASAN GAME MOBILE LEGENDS
DI GOOGLE PLAY STORE MENGGUNAKAN ALGORITMA
SVM**

beserta perangkat yang diperlukan (bila ada). Dengan demikian saya memberikan kepada Perpustakaan Universitas Sanata Dharma hak untuk menyimpan, mengalihkan dalam bentuk media lain, mengelolanya dalam bentuk pangkalan data, mendistribusikan secara terbatas, dan mempublikasikannya di Internet atau media lain untuk kepentingan akademis tanpa perlu meminta ijin dari saya maupun memberikan royalti kepada saya selama tetap mencantumkan nama saya sebagai penulis.

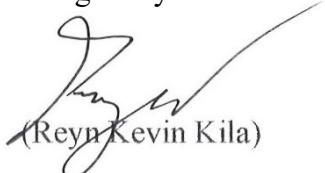
Atas kemajuan teknologi informasi, saya tidak berkeberatan jika nama, tanda tangan, gambar atau *image* yang ada di dalam karya ilmiah saya terindeks oleh mesin pencari (*search engine*), misalnya *google*.

Demikian pernyataan ini yang saya buat dengan sebenarnya.

Dibuat di Yogyakarta

Pada tanggal : 3 Februari 2023

Yang menyatakan


(Reyn Kevin Kila)

KATA PENGANTAR

Puji syukur dan terima kasih ke hadirat Tuhan Yesus Kristus, yang telah melimpahkan berkat dan karunia kepada penulis sehingga dapat menyelesaikan skripsi ini. Penulisan skripsi ini bertujuan untuk memenuhi salah satu syarat untuk memperoleh gelar sarjana pada Program Studi Informatika, Fakultas Sains dan Teknologi Universitas Sanata Dharma. Penulis mendapat bantuan dalam menyelesaikan skripsi ini, serta bimbingan dan arahan dari berbagai pihak. Penulis mengucapkan terima kasih yang tak terhingga kepada:

1. Ayah, Ibu, dan adik yang peduli pada pendidikan anak dan kakaknya, dan banyak mendorong serta mendoakan penulis hingga skripsi ini dapat selesai.
2. Bapak Ir. Drs. Haris Sriwindono, M.Kom., Ph.D. selaku Dekan Fakultas Sains dan Teknologi Universitas Sanata Dharma.
3. Bapak Kartono Pinaryanto, S. T., M. Cs. selaku Pembimbing yang telah membantu serta membimbing penulis dalam menyelesaikan skripsi ini.
4. Ibu Agnes Maria Polina S.Kom., M.Sc. dan Bapak Ir. Robertus Adi Nugroho, M.Eng. sebagai penguji yang memberikan masukan-masukan yang sangat bermanfaat dalam penyelesaian penulisan skripsi ini.
5. Seluruh dosen Informatika Universitas Sanata Dharma yang telah memberikan ilmu pengetahuan selama masa kuliah.
6. Semua pihak keluarga dan teman-teman yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa skripsi ini masih banyak kekurangannya, oleh karena itu penulis mengharapkan kritik dan saran. Semoga skripsi ini dapat bermanfaat bagi pembaca.

Yogyakarta, 26 Januari 2023



Penulis
(Reyn Kevin Kila)

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PERSETUJUAN PEMBIMBING	iii
HALAMAN PENGESAHAN.....	iv
HALAMAN KUTIPAN	v
PERNYATAAN KEASLIAN KARYA TULIS SKRIPSI	vi
LEMBAR PERNYATAAN PERSETUJUAN PUBLIKASI.....	vii
KATA PENGANTAR.....	viii
DAFTAR ISI.....	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR.....	xiv
ABSTRAK	xv
ABSTRACT	xvi
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	4
1.3 Tujuan	5
1.4 Batasan Masalah	5
1.5 Manfaat Penelitian	5
1.6 Sistematika Penulisan	6
BAB II DASAR TEORI.....	7
2.1 Tinjauan Pustaka	7
2.2 Analisis Sentimen	9
2.3 Web Scraping	9
2.4 Knowledge Discovery in Database (KDD).....	9
2.5 Text Mining.....	10
2.6 Preprocessing	10
2.6.1 Case Folding	10
2.6.2 Cleaning	11
2.6.3 Tokenize.....	11
2.6.4 Stemming	11
2.6.5 Stopword Removal.....	16

2.7	Labeling	16
2.8	Pembobotan TF-IDF	16
2.9	Klasifikasi	17
2.10	Parameter SVM	22
2.11	K-Fold Cross Validation	23
2.12	Confusion Matrix.....	25
BAB III METODOLOGI PENELITIAN	27	
3.1	Pengumpulan Data	27
3.2	Tahapan Penelitian.....	28
3.1.1	Studi Pustaka.....	28
3.1.2	Pengumpulan Data	29
3.1.3	Rancangan Sistem	29
3.1.4	Pembuatan alat Uji	29
3.1.5	Pengujian.....	29
3.3	Skema Desain.....	29
3.4	Preprocessing	31
3.4.1	Case-Folding	31
3.4.2	Cleaning	32
3.4.3	Tokenize.....	32
3.4.4	Stemming	33
3.4.5	Stopword Removal.....	35
3.5	Terjemahan Data dan Labeling Vader Lexicon	36
3.6	Pembobotan TF-IDF	37
3.7	Pengujian Metode	41
3.7.1	Klasifikasi Metode Support Vector Machine (SVM)	41
3.7.2	Confusion Matrix	44
3.7.3	Desain GUI Program.....	46
3.8	Skenario Pengujian	47
3.8.1	Pengujian Sistem Klasifikasi Support Vector Machine (SVM).....	47
3.8.2	Uji Data Tunggal.....	48
BAB IV IMPLEMENTASI SISTEM DAN ANALISIS HASIL	49	
4.1	Tahapan Analisis Sentimen.....	49

4.1.1	Scraping Data.....	49
4.1.2	<i>Preprocessing Data</i>	50
4.1.2.1	Clean Ulasan.....	50
4.1.2.2	Tokenize, Stemming, dan Stopword Removal	52
4.1.3	Menerjemahkan Data	54
4.1.4	Pelabelan <i>Vader Lexicon</i>	55
4.2	Klasifikasi Support Vector Machine (SVM).....	56
4.2.1	Uji Parameter Terbaik.....	58
4.2.1.1	Pengujian Kernel dengan Parameter C	58
4.2.1.2	Pengujian Kernel dengan parameter <i>gamma</i>	64
4.2.1.3	Pengujian Parameter <i>degree</i>	69
4.2.2	Perbandingan Hasil Uji Akurasi Kernel.....	72
4.3	Pengujian data Tunggal.....	73
4.4	Analisis Hasil	74
4.4.1	Klasifikasi Support Vector Machine (SVM).....	74
4.4.2	Confusion matrix.....	75
BAB 5 PENUTUP.	79
5.1	Kesimpulan	79
5.2	Saran.....	79
DAFTAR PUSTAKA	80
LAMPIRAN	84

DAFTAR TABEL

Tabel 2.1 Tabel awalan dan akhiran	13
Tabel 2.2 Aturan Awalan dan Peluruhan	14
Tabel 2.3 Ukuran evaluasi model klasifikasi	25
Tabel 2.4 Contoh <i>Confusion Matrix</i>	25
Tabel 3.1 Contoh Data <i>Ulasan</i>	27
Tabel 3.2 Hasil Tahap <i>Case-Folding</i>	31
Tabel 3.3 Hasil Tahap <i>Cleaning</i>	32
Tabel 3.4 sebelum <i>Tokenize</i>	33
Tabel 3.5 Setelah <i>Tokenize</i>	33
Tabel 3.6 Hasil Tahap <i>Stemming</i>	34
Tabel 3.7 Hasil Tahap <i>Stopword Removal</i>	35
Tabel 3.8 Terjemahan Data	36
Tabel 3.9 Hasil Pelabelan <i>Vader Lexicon</i>	36
Tabel 3.10 <i>Term Frequency</i> (TF)	37
Tabel 3.11 <i>DF(Document Frequent)</i>	38
Tabel 3.12 <i>Inverse Document Frequency</i> (IDF)	39
Tabel 3.13 Pembobotan kata (W)	40
Tabel 3.14 Cosine Normalization	41
Tabel 3.15 Contoh data sampel	42
Tabel 3.16 Plot <i>Hyperplane</i>	44
Tabel 3.17 Perbandingan label asli dan SVM	45
Tabel 3.18 <i>Confusion Matrix SVM</i>	45
Tabel 3.19 Hasil Pelabelan <i>Vader Lexicon</i>	48
Tabel 4. 1 Hasil <i>Cleaning</i>	52
Tabel 4.2 hasil <i>tokenize, stemming, dan stopword removal</i>	54
Tabel 4.3 Pengujian kernel <i>linear, k-fold = 3</i>	59
Tabel 4.4 Pengujian kernel <i>linear, k-fold = 5</i>	59
Tabel 4. 5 Pengujian kernel <i>linear, k-fold = 10</i>	60
Tabel 4.6 Pengujian Kernel <i>rbf, k-fold = 3</i>	60
Tabel 4.7 Pengujian kernel <i>rbf, k-fold = 5</i>	61

Tabel 4.8 Pengujian kernel <i>rbf</i> , <i>k-fold</i> = 10	61
Tabel 4.9 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 3	62
Tabel 4.10 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 5	62
Tabel 4.11 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 10	63
Tabel 4.12 Pengujian Kernel dengan Parameter C Terbaik	63
Tabel 4.13 Pengujian kernel <i>rbf</i> , <i>k-fold</i> = 3	65
Tabel 4.14 Pengujian kernel <i>rbf</i> , <i>k-fold</i> = 5	65
Tabel 4.15 Pengujian kernel <i>rbf</i> , <i>k-fold</i> = 10	66
Tabel 4.16 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 3	67
Tabel 4.17 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 5	67
Tabel 4.18 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 10	68
Tabel 4.19 Pengujian Kernel dengan parameter <i>gamma</i> terbaik	68
Tabel 4.20 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 3	70
Tabel 4.21 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 5	70
Tabel 4.22 Pengujian kernel <i>polynomial</i> , <i>k-fold</i> = 10	70
Tabel 4.23 Pengujian Kernel dengan Parameter <i>degree</i> Terbaik	71
Tabel 4.24 Kernel dengan Parameter Terbaik Berdasarkan Akurasi Tertinggi	72
Tabel 4.25 Model Evaluasi Terbaik	72
Tabel 4.26 Hasil Pengujian Data Tunggal	73
Tabel 4.27 <i>Confusion matrix</i> Pengujian Data Tunggal	74
Tabel 4.28 Hasil Klasifikasi Kernel dalam SVM	75
Tabel 4.29 Percobaan ke 1 <i>Confusion matrix</i>	75
Tabel 4.30 Percobaan ke 2 <i>Confusion matrix</i>	75
Tabel 4.31 Percobaan ke 3 <i>Confusion matrix</i>	76
Tabel 4.32 Percobaan ke 4 <i>Confusion matrix</i>	76
Tabel 4.33 Percobaan ke 5 <i>Confusion matrix</i>	76
Tabel 4.34 Percobaan ke 6 <i>Confusion matrix</i>	77
Tabel 4.35 Percobaan ke 7 <i>Confusion matrix</i>	77
Tabel 4.36 Percobaan ke 8 <i>Confusion matrix</i>	77
Tabel 4.37 Percobaan ke 9 <i>Confusion matrix</i>	78
Tabel 4.38 Percobaan ke 10 <i>Confusion matrix</i>	78

DAFTAR GAMBAR

Gambar 2.1 <i>Hyperlane</i> . (Nugroho, et al., 2003)	18
Gambar 2.2 Margin <i>Hyperplane</i>	18
Gambar 2. 3 Pemetaan data ke ruang vektor berdimensi lebih tinggi	20
Gambar 2. 4 Ilustrasi <i>5-fold cross validation</i>	23
Gambar 3. 1 Desain Skema.....	30
Gambar 3.2 koordinat Data Mentah.....	42
Gambar 3.3 Visualisasi <i>Hyperlane</i>	44
Gambar 3. 4 Desain GUI Program.....	46
Gambar 4.1 <i>Source Code</i> untuk <i>Scraping</i> Data pada <i>Google Play Store</i>	49
Gambar 4.2 Data Mentah Hasil <i>Scraping</i>	50
Gambar 4. 3 <i>Source Code Cleaning ulasan</i>	51
Gambar 4.4 source Code <i>tokenize, stemming, dan stopword removal</i>	53
Gambar 4.5 Data sebelum diterjemahkan	54
Gambar 4.6 Data setelah diterjemahkan <i>onlinedoctranslator.com</i>	54
Gambar 4.7 Source Code labeling menggunakan <i>Vader Lexicon</i>	55
Gambar 4.8 Hasil pelabelan mengguakan <i>vader lexicon</i>	55
Gambar 4. 9 Tampilan GUI program.....	56
Gambar 4.10 Source code <i>vectorizer</i>	56
Gambar 4.11 Hasil <i>Vectorize</i> dengan kernel <i>linear</i> dan data <i>training K = 5</i>	57
Gambar 4.12 Source code klasifikasi <i>Support Vector Machine (SVM)</i>	57
Gambar 4.13 Tampilan GUI kernel <i>RBF</i> , $C = 5$, $gamma = 9$	58
Gambar 4.14 Grafik Parameter C Terbaik	64
Gambar 4.15 Grafik Parameter <i>Gamma</i> Terbaik	69
Gambar 4.16 Grafik Parameter <i>degree</i> Terbaik	71
Gambar 4.17 Grafik Kernel dengan Parameter Terbaik Berdasarkan Akurasi Tertinggi.....	72

ABSTRAK

Analisis sentimen *Ulasan* game *Mobile Legends* di *Google Play Store* menggunakan algoritma *SVM* merupakan proses menganalisa, memahami dan mengklasifikasikan suatu penilaian yang dikeluarkan pengguna terhadap aplikasi game mobile legends. Data penelitian ini diambil dari website *Google Play Store* menggunakan teknik *scraping web*, pada *ulasan* game mobile legends. Data yang diambil yaitu data teks *ulasan* dengan jumlah 5000 *ulasan* berbahasa indonesia. Kemudian, data masuk ke tahap *preprocessing*, setelah itu masuk ke proses pelabelan *Vader Lexicon*, kemudian dilakukan pembobotan kata, lalu setelah pembobotan kata, data akan diklasifikasi menggunakan algoritma *SVM*. Pengujian data menggunakan kombinasi dari pembagian data *training* dan data *testing*, dan juga pengujian menggunakan sistem *k-fold cross-validation*, *confusion matrix*.

Pada pengujian untuk mencari akurasi terbaik, kernel *linear* menggunakan nilai *k-fold* = 10 dan nilai parameter $C = 10$ diperoleh hasil rata-rata akurasi sebesar 77.51%, *precision* sebesar 79.80%, dan *recall* sebesar 74.82%. pada kernel *polynomial* menggunakan menggunakan nilai *k-fold* = 10, parameter $C = 40$, dan $\gamma = \text{scale}$, dan *degree* = 2 diperoleh diperoleh rata-rata hasil akurasi sebesar 75.44%, *precision* sebesar 75.39%, dan *recall* sebesar 76.92%.

Hasil pengujian terbaik berdasarkan akurasi tertinggi terdapat pada kernel *rbf* menggunakan nilai *k-fold* = 10, parameter $C = 5$, dan $\gamma = 1$ diperoleh rata-rata hasil akurasi sebesar 77.54%, *precision* sebesar 79.81%, dan *recall* sebesar 74.99%.

Kata kunci : *Game, Ulasan, Klasifikasi, Mobile Legends, Support Vector Machine, analisis sentimen, Confusion Matrix*

ABSTRACT

Sentiment analysis Mobile Legends game reviews on the Google Play Store using the SVM algorithm are the process of analyzing, understanding, and classifying an assessment issued by users of the mobile legends game application. The research data was taken from the Google Play Store website using web scraping techniques, in reviews of mobile legends games. The data taken is review text data with a total of 5000 Indonesian language reviews. Then, the data enters the preprocessing stage, and after that, it enters the Vader Lexicon labeling process, then the words are weighted, then after the words are weighted, the data will be classified using the SVM algorithm. Data testing uses a combination of the distribution of training data and data testing, and also tests using the k-fold cross-validation system, and confusion matrix.

In testing to find the best accuracy, the linear kernel uses a k-fold value = 10 and a parameter value $C = 10$ to obtain an average accuracy of 77.51%, a precision of 79.80%, and a recall of 74.82%. on the polynomial kernel using k-fold = 10, parameter $C = 40$, and gamma = scale, and degree = 2 obtained an average accuracy of 75.44%, precision of 75.39%, and recall of 76.92%.

The best test results based on the highest accuracy were found in the RBF kernel using k-fold = 10, parameter $C = 5$, and gamma = 1, obtained an average accuracy of 77.54%, precision of 79.81%, and recall of 74.99%.

Keywords : Game, Review, Classification, Mobile Legends, Support Vector Machine, sentiment analysis, Confusion Matrix

BAB I

PENDAHULUAN

1.1 Latar Belakang

Game Mobile Legends: Bang Bang (MLBB) merupakan salah satu game online yang belakangan ini banyak dimainkan oleh hampir semua kalangan. *Game Mobile Legends* dapat dimainkan di *PC* menggunakan emulator dan juga *Smartphone*. *Game Mobile Legends: Bang Bang (MLBB)* adalah game *Multiplayer Online Battle Arena (MOBA)* dengan pertarungan *5 versus 5* melawan tim musuh. Game ini mempunyai 3 jalur (*line*) yang tiap jalurnya diperkuat dengan *tower*. Jika ingin memenangkan pertandingan, Setiap tim harus menghancurkan *tower* di setiap jajur yang dilalui, sehingga bisa mencapai markas (*base*) tim musuh dan menghancurkannya. Pemain bisa mendapat gelar *Most Valuable Player (MVP)* jika berhasil membunuh banyak musuh dalam satu pertandingan. *Game Mobile Legends: Bang Bang (MLBB)* kurang lebih sudah diunduh 100 juta kali di *Play Store* sehingga menjadikannya salah satu game *mobile* yang masuk dalam daftar game *e-sports* terpopuler tahun 2019.

E-sports merupakan singkatan dari *Electronic Sports*, memiliki arti Olahraga Elektronik dimana aspek dari olahraga ini semuanya di fasilitasi oleh sistem elektronik. *E-sports* adalah sebuah cabang olahraga yang bertanding dalam game dan dipertandingkan secara *online* melalui *PC* atau martphone. Setiap tahun Gamers di Indonesia terus meningkat, terlebih dalam game *online*. Game online merupakan sebuah permainan berbasis internet yang sedang di gemari saat ini (Januar & Turmudzi, 2006:52). Salah satu tempat untuk mencari aplikasi game adalah *Google Play Store*.

Google Play Store merupakan layanan penyedia konten digital milik Google yang menyediakan berbagai aplikasi seperti game, film atau musik, dan buku dengan beragam kategori. *Google Play Store* dapat diakses melalui website, aplikasi android, dan Google TV. Pada Google Play Store terdapat beberapa fitur salah satunya adalah fitur rating dan ulasan dari para pengguna aplikasi atau layanan yang tersedia. Ulasan merupakan suatu penilaian atau komentar dari seorang

terhadap sesuatu. Pentingnya ulasan tersebut sering digunakan sebagai tolak ukur suatu aplikasi apakah recommended atau tidak bagi para pengguna baru (Saputra et al, 2019). Dalam *Google Play Store*, disediakan tempat untuk memberikan ulasan pada setiap aplikasi yang di unduh. Kolom ulasan dapat menampung 500 karakter, sehingga pengguna cukup leluasa untuk memberikan ulasan terhadap aplikasi yang telah digunakannya. Pengguna mengulas sesuatu di *Google Play Store*, ulasan ditautkan ke Akun *Google* pengguna dan bersifat publik. Jika pengguna tidak ingin ulasannya dapat dilihat oleh semua orang, pengguna dapat menghapus ulasan tersebut. Ulasan diperlukan untuk membantu produsen dalam melihat tanggapan konsumen terhadap produk yang dihasilkannya serta membantu konsumen dalam menentukan atau memutuskan menggunakan produk tersebut. Konsumen dan produsen yang membaca sekumpulan ulasan suatu produk tentunya mencari apakah produk tersebut mempunyai opini yang positif atau negatif (Pang & Lee, 2008).

Pengguna *Google Play Store* dapat memberikan ulasan berupa opini atau masukkan ketika aplikasi yang di unduhnya, salah satunya aplikasi *Game Mobile Legends: Bang Bang (MLBB)* baik itu ulasan yang positif maupun ulasan yang negatif. Untuk mengklasifikasikan ulasan positif dan ulasan negatif terhadap *Game Mobile Legends: Bang Bang (MLBB)* di *Google Play Store*, dapat dilakukan dengan *Analisis Sentimen*. *Sentiment analysis* (analisis sentimen) atau sering disebut juga dengan opinion mining (penambangan opini) adalah studi komputasi untuk mengenali dan mengekspresikan opini, sentimen, evaluasi, sikap, emosi, subjektifitas, penilaian atau pandangan yang terdapat dalam suatu teks yang dikemukakan oleh masyarakat (Liu. B, 2012).

Pada penelitian ini, penulis akan melakukan analisis terhadap kualitas dan pelayanan *Game Mobile Legends: Bang Bang (MLBB)* berdasarkan ulasan pengguna di aplikasi *Google Play Store*, karena ulasan pada Aplikasi game *Mobile Legends: Bang Bang (MLBB)* berguna bagi developer untuk meningkatkan kualitas dan pelayanan kedepannya serta mengetahui sentimen pengguna terhadap game Mobile legends.

Dalam penelitian ini penulis melakukan analisis sentimen dengan menerapkan algoritma *Support Vector Machine (SVM)*. Beberapa penelitian terkait Sentimen Analisis yang berkaitan dengan penelitian ini antara lain Irfani, dkk (2020)

tentang Analisis Sentimen Review Aplikasi Ruangguru Menggunakan Algoritma *Support Vector Machine (SVM)*. penelitian ini menggunakan algoritma *Support Vector Machine (SVM)* untuk mengklasifikasikan data. Pengujian menggunakan kernel linear menghasilkan akurasi tertinggi dengan nilai 0.897. kombinasi data training 60% dan data testing 40% menghasilkan akurasi tertinggi sebesar 0.900. pengujian dengan menggunakan sistem k-fold, akurasi tertinggi berada di nilai k-fold 6, 9, dan 10 dengan nilai akurasi sebesar 0.902. Pada k-fold 10, mendapat nilai presisi sebesar 0.903.

Penelitian lainnya terkait sentimen analisis, yaitu dilakukan oleh Haryanto, Dimas Joko dkk. (2018) tentang Analisis Sentimen Review Barang Berbahasa Indonesia Dengan Metode *Support Vector Machine* Dan *Query Expansion*. Dalam penelitian ini menggunakan 400 data komentar yang terbagi menjadi dua yaitu positif dan negatif. Adapun metode yang digunakan adalah metode *Support Vector Machine kerne Polynomial* berderajat dua dengan *Query Expansion*. *Query Expansion* digunakan untuk memperluas kata pada data uji yang memiliki sinonim yang tidak terdapat pada data latih. Hasil pengujian akhir menghasilkan rata-rata akurasi sebesar 96,25% dengan parameter nilai *learning rate* = 0,001, nilai *lambda* = 0,1, nilai *complexity* = 0,01 dan iterasi maksimal adalah 50. Berdasarkan hasil pengujian, diperoleh akurasi metode *Support Vector Machine* dan *Query Expansion* sebesar 96,25% dan akurasi menggunakan metode *Support Vector Machine* tanpa *Query Expansion* sebesar 94,75%. Teknik SVM digunakan untuk mendapatkan fungsi pemisah (*hyperplane*) yang optimal untuk memisahkan observasi yang memiliki nilai variabel target yang berbeda (William, 2011). Metode *Support Vector Machine (SVM)* sendiri merupakan metode klasifikasi linier dengan menemukan *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah kelas pada inputspace (Saifinnuha, 2015). Banyak peneliti telah melaporkan bahwa *Support Vector Machine (SVM)* metode yang paling akurat untuk teks klasifikasi (Moraes, Valiati, & Gavião Neto, 2013).

Dalam penelitian yang dilakukan oleh N. Fitriyah, dkk. (2020). Berjudul “Analisis Sentimen Gojek Pada Media Sosial Twitter Dengan Klasifikasi Support Vector Machine (Svm),” penelitian ini menggunakan metode Support Vector Machine dengan evaluasi model 10-fold cross validation. Pelabelan

dilakukan secara manual dan menggunakan kernel linear dan RBF. Hasil pengujian menunjukkan bahwa kernel RBF mendapatkan akurasi sebesar 79,19% dan akurasi kappa adalah 16,52%. Sedangkan pelabelan data dengan skoring sentimen diperoleh akurasi keseluruhan sebesar 79,19% dan akurasi kappa sebesar 21%.

Penelitian lainnya yang dilakukan oleh Indrayuni (2018) dengan judul Komparasi Algoritma *Naïve Bayes* dan *Support Vector Machine* untuk Analisa Sentimen *Review Film*. Hasil dari penelitian ini mendapatkan nilai akurasi algoritma *Naive Bayes* sebesar 84.50%. Sedangkan nilai akurasi dari algoritma *Support Vector Machine* lebih besar yaitu 90.00%. Penelitian lainnya juga dilakukan oleh Pravina, Arsyia monica dkk. (2019) tentang Analisis Sentimen Tentang Opini Maskapai Penerbangan pada Dokumen Twitter Menggunakan Algoritme Support Vector Machine (SVM). Pada penelitian ini, digunakan algoritma *Support Vector Machine* untuk melakukan klasifikasi. Hasil dari penelitian ini menampilkan parameter optimal dan pengaruh penggunaan *Lexicon Based Features*. Nilai parameter C adalah 10 dan *learning rate* bernilai 0,03 serta digunakan *Lexicon Based Features* dengan iterasi sebanyak 50 kali memberikan hasil *accuracy* sebesar 40%, *precision* 40%, 100% *recall*, dan *f-measure* sebesar 57,14%.

Penelitian berjudul Analisis Sentimen Zoom Cloud Meetings di Play Store Menggunakan Naïve Bayes dan Support Vector Machine yang dilakukan oleh Nuraeni Herlinawati, dkk (2020). penelitian ini menggunakan dataset sebanyak 1.007 record setelah *preprocessing*. Data label positif sebanyak 546 ulasan dan label negatif 461 ulasan. Evaluasi model menggunakan 10 fold cross validation diperoleh nilai akurasi dan nilai AUC dari masing-masing algoritma yaitu untuk algoritma SVM nilai akurasi = 81,22% dan nilai AUC = 0,886. Sedangkan NB nilai akurasi = 74,37% dan nilai AUC = 0,659. Dari hasil tersebut tingkat akurasi algoritma SVM mengungguli Naïve Bayes.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang ada, rumusan masalahnya adalah sebagai berikut :

1. Bagaimana melakukan analisis sentimen terhadap ulasan *Game Mobile Legends: Bang Bang (MLBB)* di aplikasi *Google Play Store* menggunakan algoritma *Support Vector Machine*?
2. Bagaimana tingkat akurasi yang diperoleh algoritma *Support Vector Machine* terhadap ulasan *Game Mobile Legends: Bang Bang (MLBB)* di aplikasi *Google Play Store* dengan menggunakan kernel *Linear*, *Polynomial*, *Radial Basis Function (RBF)*?

1.3 Tujuan

1. Mengetahui cara melakukan analisis sentimen terhadap ulasan *Game Mobile Legends: Bang Bang (MLBB)* di aplikasi *Google Play Store* menggunakan algoritma *Support Vector Machine*
2. Mengetahui tingkat akurasi yang diperoleh algoritma *Support Vector Machine* terhadap ulasan *Game Mobile Legends: Bang Bang (MLBB)* di aplikasi *Google Play Store* dengan menggunakan kernel *Linear*, *Polynomial*, *Radial Basis Function (RBF)*.

1.4 Batasan Masalah

1. Menggunakan data ulasan *Game Mobile Legends: Bang Bang (MLBB)* yang berasal dari *Google Play Store*.
2. Ulasan yang diambil secara kumulatif bulan Desember 2021 sampai dengan bulan Juni 2022
3. Ulasan yang digunakan hanya dalam *Bahasa Indonesia*
4. Algoritma yang di gunakan yaitu *Support Vector Machine (SVM)* dalam klasifikasi data.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini, yaitu mengetahui sentimen pengguna terhadap game Mobile legends yang berguna bagi developer untuk meningkatkan kualitas dan pelayanan kedepannya, dan sebagai pertimbangan masyarakat untuk memainkan game *Mobile Legends: Bang Bang (MLBB)*.

1.6 Sistematika Penulisan

BAB I : Pendahuluan

Pada BAB I terdapat Pendahuluan serta latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat penelitian dan sistematika penulisan.

BAB II : Tinjauan Pustaka

Pada BAB II terdapat Tinjauan Pustaka yang berisi tentang teori dan sumber tertulis untuk menunjang penelitian yang berkaitan dengan analisis sentimen terhadap ulasan *Game Mobile Legends: Bang Bang (MLBB)*, *text mining*, KDD, *preprocessing*, dan algoritma *Support Vector Machine* (SVM).

BAB III : Metodologi Penelitian

Pada BAB III terdapat Metodologi Penelitian yang berisi tentang penerapan metode atau algoritma yang akan digunakan oleh penulis.

BAB IV : Implementasi Sistem dan Analisis Hasil

Pada BAB IV terdapat Implementasi Sistem dan Analisa Hasil yang berisi tentang implementasi dan analisis sistem berdasarkan hasil yang telah dibuat, serta akan menjelaskan kekurangan, kelebihan dan akurasi dari penelitian yang dibuat.

BAB V : Kesimpulan dan Saran

Pada BAB V terdapat kesimpulan dan saran berisi tentang kesimpulan hasil akhir penelitian serta saran.

BAB II

DASAR TEORI

2.1 Tinjauan Pustaka

Penulis mengkaji Penelitian terkait sebagai referensi dalam membuat penelitian. Berikut beberapa penelitian terkait yang menjadi referensi penulis.

Pada penelitian kali ini, penulis mendapat referensi dari penelitian sebelumnya yang sudah pernah dibuat oleh Irfani, dkk (2020) tentang Analisis Sentimen Review Aplikasi Ruangguru Menggunakan Algoritma *Support Vector Machine (SVM)*. penelitian ini menggunakan algoritma *Support Vector Machine (SVM)* untuk mengklasifikasikan data. Pengujian menggunakan kernel linear menghasilkan akurasi tertinggi dengan nilai 0.897. kombinasi data training 60% dan data testing 40% menghasilkan akurasi tertinggi sebesar 0.900. pengujian dengan menggunakan sistem k-fold, akurasi tertinggi berada di nilai k-fold 6, 9, dan 10 dengan nilai akurasi sebesar 0.902. Pada k-fold 10, mendapat nilai presisi sebesar 0.903.

Penelitian lainnya terkait sentimen analisis, yaitu dilakukan oleh Haryanto, Dimas Joko dkk. (2018) tentang Analisis Sentimen Review Barang Berbahasa Indonesia Dengan Metode *Support Vector Machine* Dan *Query Expansion*. Dalam penelitian ini menggunakan 400 data komentar yang terbagi menjadi dua yaitu positif dan negatif. Adapun metode yang digunakan adalah metode *Support Vector Machine kerne Polynomial* berderajat dua dengan *Query Expansion*. *Query Expansion* digunakan untuk memperluas kata pada data uji yang memiliki sinonim yang tidak terdapat pada data latih. Hasil pengujian akhir menghasilkan rata-rata akurasi sebesar 96,25% dengan parameter nilai *learning rate* = 0,001, nilai *lambda* = 0,1, nilai *complexity* = 0,01 dan iterasi maksimal adalah 50. Berdasarkan hasil pengujian, diperoleh akurasi metode *Support Vector Machine* dan *Query Expansion* sebesar 96,25% dan akurasi menggunakan metode *Support Vector Machine* tanpa *Query Expansion* sebesar 94,75%.

Dalam penelitian yang dilakukan oleh N. Fitriyah, dkk. (2020). Berjudul “Analisis Sentimen Gojek Pada Media Sosial Twitter Dengan Klasifikasi Support Vector Machine (Svm),” penelitian ini menggunakan metode Support Vector Machine dengan evaluasi model 10-fold cross validation. Pelabelan dilakukan secara manual dan menggunakan kernel linear dan RBF. Hasil pengujian menunjukkan bahwa kernel RBF mendapatkan akurasi sebesar 79,19% dan akurasi kappa adalah 16,52%. Sedangkan pelabelan data dengan skoring sentimen diperoleh akurasi keseluruhan sebesar 79,19% dan akurasi kappa sebesar 21%.

Penelitian lainnya yang dilakukan oleh Indrayuni (2018) dengan judul Komparasi Algoritma *Naïve Bayes* dan *Support Vector Machine* untuk Analisa Sentimen *Review Film*. Hasil dari penelitian ini mendapatkan nilai akurasi algoritma *Naive Bayes* sebesar 84.50%. Sedangkan nilai akurasi dari algoritma *Support Vector Machine* lebih besar yaitu 90.00%.

Penelitian yang dilakukan oleh Pravina, Arsyia monica dkk. (2019) tentang Analisis Sentimen Tentang Opini Maskapai Penerbangan pada Dokumen Twitter Menggunakan Algoritme Support Vector Machine (SVM). Pada penelitian ini, digunakan algoritma *Support Vector Machine* untuk melakukan klasifikasi. Hasil dari penelitian ini menampilkan parameter optimal dan pengaruh penggunaan *Lexicon Based Features*. Nilai parameter C adalah 10 dan *learning rate* bernilai 0,03 serta digunakan *Lexicon Based Features* dengan iterasi sebanyak 50 kali memberikan hasil *accuracy* sebesar 40%, *precision* 40%, 100% *recall*, dan *f-measure* sebesar 57,14%.

Penelitian berjudul Analisis Sentimen Zoom Cloud Meetings di Play Store Menggunakan Naïve Bayes dan Support Vector Machine yang dilakukan oleh Nuraeni Herlinawati, dkk (2020). penelitian ini menggunakan dataset sebanyak 1.007 record setelah *preprocessing*. Data label positif sebanyak 546 ulasan dan label negatif 461 ulasan. Evaluasi model menggunakan 10 fold cross validation diperoleh nilai akurasi dan nilai AUC dari masing-masing algoritma yaitu untuk algoritma SVM nilai akurasi = 81,22% dan nilai AUC = 0,886. Sedangkan NB nilai akurasi = 74,37% dan nilai AUC = 0,659. Dari hasil tersebut tingkat akurasi algoritma SVM mengungguli Naïve Bayes.

2.2 Analisis Sentimen

Sentiment Analysis (analisis sentimen) atau sering disebut juga dengan *Opinion Mining* (penambangan opini) adalah studi komputasi untuk mengenali dan mengekspresikan opini, sentimen, evaluasi, sikap, emosi, subjektifitas, penilaian atau pandangan yang terdapat dalam suatu teks. (Liu, 2008). Analisis sentimen atau opinion mining merupakan proses memahami, mengekstrak dan mengolah data tekstual secara otomatis untuk mendapatkan informasi sentimen yang terkandung dalam suatu kalimat opini.

2.3 Web Scraping

Web scraping adalah teknik untuk mendapatkan informasi dari website secara otomatis tanpa harus menyalinnya secara manual. Tujuan dari web scraper adalah untuk mencari informasi tententu dan kemudian mengumpulkannya dalam web yang baru. Web scraping berfokus dalam mendapatkan data dengan cara pengambilan dan ekstraksi. Manfaat dari web scraping ialah agar informasi yang dikeruk lebih terfokus sehingga memudahkan dalam melakukan pencarian sesuatu. Aplikasi Web Scraping hanya fokus pada cara memperoleh data melalui pengambilan dan ekstraksi data dengan ukuran data yang bervariasi. Web scraping adalah sebuah proses yang memanfaatkan dokumen berbentuk *semi-structured* yang diperoleh dari internet, yang dimana dokumen tersebut berbentuk sebuah halaman website yang dibangun oleh bahasa markup seperti HTML ataupun XHTML yang kemudian dianalisis untuk mendapatkan informasi yang berguna yang dapat dilakukan untuk konteks lain (Nikhil, 2015).

2.4 Knowledge Discovery in Database (KDD)

KDD adalah kegiatan yang meliputi pengumpulan, pemakaian data, historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar (Santoso, 2007). KDD berhubungan dengan teknik integrasi dan penemuan ilmiah, interpretasi dan visualisasi dari pola-pola sejumlah kumpulan data. KDD terdiri dari serangkaian langkah perubahan, termasuk data *pre-processing* dan juga *post processing*.

1. Pembersihan Data (*Data cleaning*), menghilangkan noise dan data yang inkonsisten.

2. Integrasi Data (*Data integration*, menggabungkan data dari berbagai sumber data yang berbeda)
3. Pemilihan Data (*Data selection*), mengambil data yang relevan dengan tugas analisis dari database
4. Transformasi Data (*Data transformation*), Mentransformasi atau menggabungkan data ke dalam bentuk yang sesuai untuk penggalian lewat operasi summary atau aggregation.
5. Penambangan Data (*Data mining*), proses esensial untuk mengekstrak pola dari data dengan metode cerdas.
6. Evaluasi Pola (*Pattern evaluation*), mengidentifikasi pola yang menarik.
7. Presentasi Pengetahuan (*Knowledge presentation*), penyajian pengetahuan kepada pengguna dengan menggunakan visualisasi dan teknik representasi pengetahuan.

2.5 Text Mining

Text mining adalah teknologi yang mampu menganalisa data teks semi-terstruktur maupun tidak terstruktur, sedangkan data mining mengolah data yang terstruktur (Han & Kamber, 2012).

Text mining pada umumnya memiliki beberapa karakteristik berdimensi tinggi, terdapat *noise* pada data, dan terdapat struktur teks yang tidak beraturan sehingga untuk mendapatkan bentuk data yang siap diproses, maka harus dilakukan *Pre-processing* terlebih dahulu.

2.6 Preprocessing

Preprocessing merupakan penting dalam melakukan *Text Mining*. *Preprocessing* dilakukan untuk mengubah data sesuai dengan format yang akan dibutuhkan untuk mendapat data yang akurat sebelum masuk ke proses selanjutnya. Tahap *Preprocessing* Meliputi:

2.6.1 Case Folding

Proses ini dilakukan untuk menyeragamkan seluruh teks yang akan dimasukkan kedalam model untuk di ubah menjadi huruf kecil seluruh teks (Lowercase). Contohnya :

Kalimat awal :

Game burikkk!! Berkali-kali keluar sendiri dari Game

Kalimat setelah proses *Case Folding* :

game burikkk!! berkali-kali keluar sendiri dari game

2.6.2 Cleaning

Proses ini bertujuan untuk menghilangkan bagian dari *ulasan* berupa noise, tanda baca dan karakter-karakter yang tidak penting. Contohnya :

Kalimat awal :

Game burikkk!! Berkali-kali keluar sendiri dari Game

Kalimat setelah proses *Cleaning* :

Game burikkk Berkali kali keluar sendiri dari Game

2.6.3 Tokenize

Proses ini dilakukan untuk memisahkan teks menjadi potongan-potongan perkata yang disebut token yang nantinya di analisis. Contohnya :

Kalimat awal :

game burikkk berkali kali keluar sendiri dari game

Kalimat setelah proses *Tokenizing* :

game	burikkk	berkali	kali	keluar	sendiri	Dari	game
------	---------	---------	------	--------	---------	------	------

2.6.4 Stemming

Proses ini bertujuan untuk mencari kata dasar(*root-word*) dari kata yang memiliki imbuhan. Pada penelitian ini, data *ulasan* menggunakan bahasa Indonesia sehingga proses *stemming* dilakukan dengan menggunakan library sastrawi. Contohnya :

Kalimat awal :

game	burik	berkali	kali	keluar	sendiri	Dari	game
------	-------	---------	------	--------	---------	------	------

Kalimat setelah proses *Stemming* :

game	burik	kali	kali	keluar	sendiri	Dari	game
------	-------	------	------	--------	---------	------	------

Algoritma Nazief & Adriani yang dibuat oleh Bobby Nazief dan Mirna Adriani ini memiliki tahap-tahap sebagai berikut:

- Langkah pertama yaitu pencarian kata dasar di kamus. Jika ditemukan maka diasumsikan kata adalah *root* atau kata dasar yang benar. Maka algoritma berhenti.
- Langkah kedua yaitu menghilangkan *Inflection Suffixes* seperti kata yang berimbuhan (“-lah”, “-kah”, “-pun”, “-mu”, atau “-nya”) dibuang. Kemudian dilanjutkan menghapus *Possesive Pronouns* (“-ku”, “-mu”, atau “-nya”), jika ada. Setelah melewati kedua tahap tersebut, selanjutnya dilakukan pengecekan kata di kamus untuk kata dasar. Jika tidak ditemukan kata dasar, maka dilanjutkan ke tahap berikutnya.
- Langkah ketiga yaitu menghapus *Derivation Suffixes* seperti kata berimbuhan (“-i”, “-an” atau “-kan”). Jika kata ditemukan di kamus, maka algoritma berhenti. Jika tidak maka ke langkah berikut :
 - a. Jika imbuhan “-an” telah dihapus dan huruf terakhir dari kata tersebut adalah “-k”, maka “-k” juga ikut dihapus. Jika kata tersebut ditemukan dalam kamus maka algoritma berhenti.
 - b. Jika tidak ditemukan, maka akhiran yang dihapus (“-i”, “-an” atau “-kan”) dikembalikan, lanjut ke langkah berikutnya.
- Langkah keempat yaitu menghapus *Derivation Prefix*. Jika pada langkah 3 ada *suffixes* yang dihapus maka dilanjutkan kelangkah keempat. Pada langkah ini terdapat 3 iterasi.
 - a. Iterasi akan berhenti jika:
 - i. Ditemukan awalan dan akhiran yang tidak diizinkan.

- ii. Terdapat kesamaan awalan dengan awalan yang telah dihilangkan sebelumnya ketika dideteksi.
- iii. Tiga awalan telah dihilangkan.

Tabel 2.1 Tabel awalan dan akhiran

Awalan	Akhiran yang tidak diizinkan
be-	-i
di-	-an
ke-	-i, -kan
me-	-an
se-	-i, -kan

- b. Tipe awalan ditentukan melalui langkah-langkah berikut:
 - i. Standar ("di-", "ke-", "se-") yang langsung dihilangkan dari kata.
 - ii. Kompleks ("me-", "be", "pe", "te") adalah tipe awalan yang bisa berubah sesuai dengan kata dasar yang mengikutinya. Untuk mendapatkan hasil pemenggalan yang tepat, dibutuhkan aturan sebagai berikut:

Tabel 2.2 Aturan Awalan dan Peluruhan

Aturan	Bentuk Awalan	Peluruhan
1	berV	Ber-V... be-rV...
2	Belajar	Bel-ajar
3	beC ₁ erC ₂	Be-C ₁ erC ₂ .. dimana C!={'r' 1}
5	teCer	Te-Cer... dimana C!=‘r’
6	teC ₁ erC ₂ ...	Te-C ₁ erC ₂ .. dimana C!= ‘r’
7	me{l r w y}V...	Me-{l r w y}V...
8	mem{b f v}...	Mem-{b f v}...
9	Mempe...	Mem-pe
10	Mem{rV V}...	Me-m{rV V}... Me-p{rV V}...
11	Men{c d z}...	Men-{c d z}...
12	menV...	Me-nV... me-tV...
13	Meng{g h q k}...	Meng-{g h q k}...
14	mengV...	Meng-V... meng-kV...
15	mengeC	Menge-C
16	menyV	Meny-V... meny-sV...
17	mempV	Mem-pV...
18	Pe{w y}V...	Pe-{w y}V...
19	perV	Per-V... per-rV
20	Pem{b f v}...	Pe-m{b f v}...
21	Pem{rV V}	Pe-m{rV V}... pe-p{rV V}
22	Pen{c d j z}...	Pen-{c d j z}...
23	penV	Pen-V... pe-tV
24	Peng{g h q}	Peng-{g h q}
25	pengV	Peng-V peng-kV
26	penyV	Pe-nya peny-sV
27	pelV	pelV...; kecuali untuk kata “pelajar” menjadi ajar
28	peCP	Pe-CP... dimana C!= {r q v l m n} dan P!=‘er’
29	perCerV	Per-CerV dimana C!= {r q v l m n}

Berdasarkan tabel 2.2 yang merupakan sebuah aturanbentuk awalan dan peluruhan yang memiliki awalan “be-”, “te-”, “me-”, dan “pe-”. Bentuk kata dasar yang memiliki awalan “be-”, “te-”, “me-”, dan “pe-” Perubahan karakter pada kata dasar setelah algoritma menghilangkan awalan yang melekat pada kata dasar tersebut terletak dalam kolom keriga pada tabel 2.2. Di dalam kolom kedua dan ketiga tersebut terdapat huruf “V” yang merupakan huruf vokal. Sedangkan huruf “C” merupakan huruf konsonan dan huruf “P” merupakan pecahan “er”.

- c. Mencari kata yang sudah tidak memiliki awalan, jika kata tidak ditemukan maka langkah diulangi kembali, algoritma akan berhenti jika kata ditemukan.
- Proses akan dilakukan dengan mengacu pada aturan tabel 2.2, jika langkah 4 kata dasar belum berhasil ditemukan.
- Kata awal akan diasumsikan sebagai *root word* jika semua langkah telah selesai tetapi tidak berhasil. Proses selesai.
- Dalam jurnal yang ditulis oleh Agusta (2009), Aturan akan ditambahkan untuk mengatasi keterbatasan dalam algoritma *stemming* sebagai berikut:
 - a. Aturan Reduplikasi
 - i. Jika kedua kata yang dihubungkan oleh kata penghubung merupakan kata yang sama maka *root word* adalah bentutunggalnya, contoh : “adik – adik” *root word*-nya adalah “adik”.
 - ii. Jika terdapat kata misalnya “berulang – ulangan”, maka untuk mendapatkan *root word*-nya, kedua kata tersebut akan diartikan secara terpisah. Kataakan diubah menjadi kalimat tunggal jika keduanya memiliki *root word* yang sama, contoh:kata “berulang – ulangan”, “berulang” dan “ulangan”, kedua kata tersebut memiliki *root word* yang sama yaitu “ulang”, maka *root word*-nya adalah “ulang”. Jika kedua kata memiliki *root word* yang berbeda contohnya : kata “mondar – mandir”, kata “mondar” dan “mandir” memiliki *root word* yang berbeda, maka *root word*-nya adalah “mondar-mandir”.
 - b. Tambahan Bentuk Awalan dan Akhiran Serta Aturannya.
 - i. Pada tipe kata yang berawalan “mem-“, kata yang diawali dengan “memp-“ memiliki tipe awalan “mem-“.
 - ii. Pada tipe awalan “meng-“, kata yang diawali dengan awalan “megk-“ memiliki tipe awalan“meng”.

2.6.5 Stopword Removal

Proses Ini dilakukan untuk menghilangkan kata yang termasuk kedalam kategori *Stopword*. *Stopword* adalah kata yang sering muncul namun dianggap tidak memiliki arti. Pada stopword, penulis menggunakan *library Sastrawi* pada *Python*.

2.7 Labeling

Dalam *text mining* terdapat proses labeling yang dilakukan untuk proses klasifikasi data. Proses *labeling* dilakukan untuk memberi label pada sebuah data. Dalam penelitian ini proses *labeling* dibantu dengan *tool* analisis sentimen, yaitu VADER (*Valence Aware Dictionary and Sentiment Reasoner*). Proses pembobotan didasarkan pada nilai label yang terdapat dalam Vader. Vader hanya dapat menganalisis data yang berbahasa Inggris. Dalam proses labeling, data yang masih berbahasa Indonesia yang telah melewati *preprocessing* diterjemahkan kedalam bahasa Inggris menggunakan onlinedoctranslator.com.

Pada proses pengelompokan, kriteria sentimen positif, netral dan negative, jika hasil compound > 0 maka dimasukkan kategori positif yang diwakilkan dengan angka 1 lalu jika hasil compound = 0 maka akan termasuk kategori netral yang diwakilkan dengan angka 0 dan yang terakhir jika hasil compound < 0 maka termasuk kategori negatif yang diwakilkan dengan angka -1 (Mustaqim, 2020).

Dalam penelitian ini, penulis menggunakan dua sentimen kelas, yaitu kelas positif dan kelas negatif.

2.8 Pembobotan TF-IDF

Term Frequency-Inverse Document Frequency (TF-IDF) adalah suatu metode yang digunakan untuk menghitung bobot dari setiap kata yang digunakan. Metode ini juga terkenal efisien, mudah dan memiliki hasil yang akurat. TF-IDF merupakan model integrasi dari model *term frequency* (TF) dan *inverse document frequency* (IDF). *Term Frequency* (TF) adalah frekuensi kemunculan sebuah term dalam sebuah dokumen. Semakin besar jumlah term yang muncul (TF tinggi) maka semakin besar bobot dokumen atau memberikan nilai kesesuaian yang semakin besar (Informatikalogi, 2016). *Inverse Document Frequency* (IDF) merupakan metode statistik numeric yang menghitung seberapa pentingnya kata dalam sebuah

dokumen. Metode ini digunakan sebagai bobot dalam pencarian informasi dalam text mining. (Fanani, 2017).

Berikut adalah tahapan pembobotan dengan TF-IDF:

$$W_{dt} = TF_{dt} * IDF_t \quad (1)$$

Keterangan:

d = dokumen

t = kata ke-t dari kata kunci

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

TF = Banyaknya kata yang muncul dalam dokumen

IDF = nilai idf dari hasil

$$IDF = \log \frac{D}{df} \quad (2)$$

D = Total dokumen

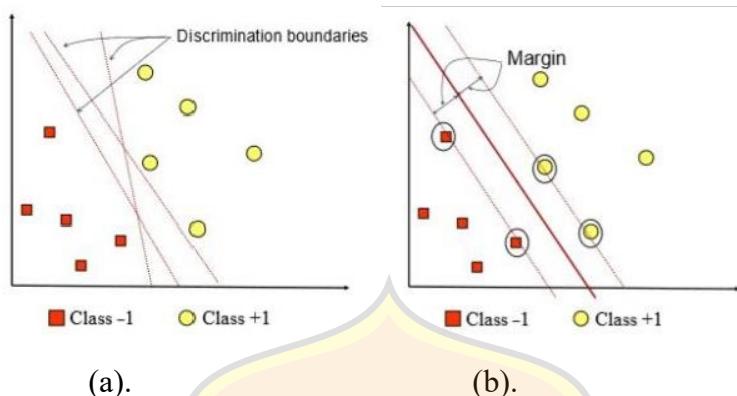
df = banyaknya dokumen yang muncul berdasarkan kata kunci

2.9 Klasifikasi

Setelah tahap *preprocessing* dan pembobotan kata, tahap berikutnya adalah klasifikasi. Metode yang akan digunakan oleh penulis, yaitu *Support Vector Machine (SVM)*. *Support Vector Machine (SVM)* diperkenalkan oleh Vapnik pada tahun 1992 sebagai konsep terbaik dalam pengenalan pola (*pattern recognition*). Feldman (Rahutomo et al, 2018). SVM merupakan metode pembelajaran linier untuk menemukan *Hyperline Optimal* untuk memisahkan dua kelas (positif dan negatif). SVM memiliki kemampuan dalam menerapkan pemisah input non-linier berdimensi tinggi yang membuat SVM menjadi istimewa dengan menggunakan fungsi kernelnya. Terdapat beberapa kernel SVM yang dapat digunakan, namun didalam suatu penelitian, kernel yang sering digunakan ialah *Linear, Polynomial, Radial Basis Function (RBF)*.

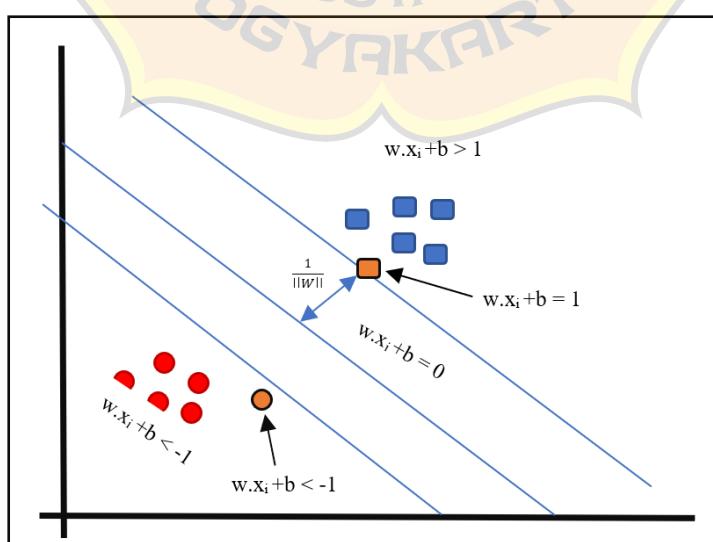
Support Vector Machine (SVM) adalah sebuah sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linear dalam ruang fitur (fitur space) berdimensi tinggi, dilatih dengan algoritma pembelajaran berdasarkan pada teori optimasi dengan mengimplementasikan learning bias yang berasal dari teori pembelajaran statistik. (Christianini & John S, 2000).

Menurut (Suyanto, 2018) SVM ditemukan oleh Vapnik pada tahun 1992 yang bertujuan untuk menghasilkan *hyperplane* paling optimum. Hyperplane merupakan garis yang memisahkan himpunan data ke dalam dua kelas secara linier.



Gambar 2.1 *Hyperlane*. (Nugroho, et al., 2003)

Konsep dasar dari SVM merupakan proses pelatihan mencari letak *hyperplane*. Pilihan untuk menemukan *hyperplane* suatu set data dapat dilihat pada gambar 2.1a, sedangkan untuk margin yang paling maksimal pada *hyperplane* dapat dilihat pada gambar 2.1b. Untuk mencari *Hyperplane* terbaik antara kedua kelas tersebut, dapat dilakukan dengan cara mengukur margin *hyperplane* dan kemudian mencari titik maksimalnya. Margin adalah jarak antara *hyperplane* tersebut dengan data terdekat dari masing-masing kelas. Data yang paling dekat ini disebut sebagai *support vector* (Prasetyo, 2014).



Gambar 2.2 Margin *Hyperplane*

Seperti pada Gambar 2.2, SVM bekerja untuk menemukan *hyperplane* dengan margin yang maksimal. *Hyperplane* klasifikasi linier memisahkan kedua kelas dengan persamaan :

$$w \cdot x_i + b = 0 \quad (3)$$

Keterangan :

w = vector bobot

x = nilai masukan atribut

b = bias

Sehingga diperoleh persamaan untuk kelas yang bernilai positif dan kelas yang bernilai negatif. Pada kelas positif (1), suatu data x_i dapat diklasifikasikan jika :

$$w \cdot x_i + b > 1 \quad (4)$$

dan dapat diklasifikasikan sebagai kelas -1 jika

$$w \cdot x_i + b \leq -1 \quad (5)$$

Untuk menemukan margin *hyperlane* terbaik, dilakukan dengan cara memaksimalkan nilai jarak antara *hyperplane* dengan titik terdekat menggunakan rumus $\frac{1}{||w||}$. Selanjutnya untuk mencari titik minimal dapat dirumuskan sebagai

Quadratic Programming (QP) Problem dengan persamaan :

$$\min_w \tau(w) = \frac{1}{2} ||w||^2 \quad (6)$$

$$y_i(x_i \cdot w + b) - 1 \geq 0 \quad (7)$$

Permasalahan ini dapat diselesaikan dengan menggunakan *Lagrange Multiplier* :

$$L(w, b, \alpha) = \frac{1}{2} ||w||^2 - \sum_{i=1}^l \alpha_i (y_i(x_i \cdot w + b) - 1), i = 1, 2, 3, \dots, l \quad (8)$$

Dengan *Lagrange multipliers* adalah $\alpha_i \geq 0$, nilai optimal dari persamaan tersebut dapat dihitung dengan meminimalkan L terhadap w dan b sekaligus memaksimalkan L terhadap α_i . Dengan diketahui titik optimal gradient L = 0, maka persamaan (2.8) dapat dimodifikasi dengan memaksimalkan :

$$\sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j x_i x_j \quad (9)$$

$$\alpha_i \geq 0 (i = 1, 2, \dots, l) \sum_{i=1}^l \alpha_i y_i = 0 \quad (10)$$

Maka dari Maksimalisasi ini, menghasilkan sejumlah α_i yang bernilai positif. *support vector* adalah data-data yang berhubungan dengan α_i positif, sehingga fungsi pemisah didefinisikan sebagai berikut :

$$g(x) := \operatorname{sgn}(f(x)) \quad (11)$$

$$\text{Dengan } f(x) = w^T x + b \quad (12)$$

(Santosa, n.d.)

Menyelesaikan masalah linier pada pembelajaran SVM mudah. Tetapi dalam kondisi nyata masalah yang dihadapi adalah masalah non-linier, sehingga SVM dimodifikasi dengan memasukan fungsi kernel. Dalam fungsi non-linier, pertama-tama SVM akan memetakan data x menggunakan fungsi $\Phi(\vec{x})$ ke ruang vektor yang berdimensi lebih tinggi. Selanjutnya fungsi Φ akan memetakan setiap data tersebut ke ruang vektor baru yang berdimensi lebih tinggi atau berdimensi tiga yanh dapat dirumuskan sebagai berikut:

$$\Phi: \mathbb{R}^d \rightarrow \mathbb{R}^q \quad d < q \quad (13)$$

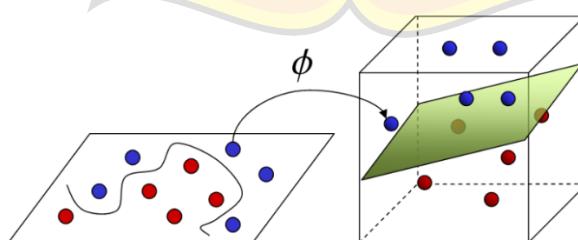
Keterangan :

\mathbb{R}^d : Ruang vector real 2 dimensi

\mathbb{R}^q : Ruang vector real 3 dimensi

D : Dimensi 2

q : Dimensi 3



Gambar 2. 3 Pemetaan data ke ruang vektor berdimensi lebih tinggi

Terdapat empat fungsi kernel yang dapat digunakan yaitu :

1. Kernel *Linear*

Kernel linear adalah fungsi kernel yang paling sederhana. Kernel linear digunakan ketika data yang dianalisis sudah terpisah secara linear. Kernel linear cocok ketika terdapat banyak fitur. Berikut persamaan dari kernel *linear* :

$$K(x, x_i) = x_K^T x \quad (14)$$

Keterangan :

K : Fungsi Kernel

x_i : Nilai atribut i

x_i^T : Input Data 1 transform

x_j : Nilai atribut j

2. Kernel *Polynomial*

Kernel polynomial adalah bentuk yang lebih umum dari kernel linear. Dalam machine learning, Fungsi kernel polynomial merupakan fungsi kernel yang digunakan ketika data tidak terpisah secara linear. Kernel polynomial juga digunakan untuk memecahkan masalah klasifikasi pada dataset pelatihan yang telah dinormalisasi. Berikut persamaan dari kernel *polynomial* :

$$K(x_i, x_j) = (yx_i^T x_j + r)^p, y > 0 \quad (15)$$

Keterangan :

y : Variabel *Gamma*

r : Variabel *Degree*

p : Varaibel Prediktor

x_i : Nilai atribut i

x_i^T : Input Data 1 transform

x_j : Nilai atribut j

3. Kernel Gaussian RBF(Radial Basis Function)

Kernel RBF atau juga disebut kernel Gaussian adalah konsep kernel yang paling banyak digunakan untuk memecahkan masalah klasifikasi data yang tidak dapat dipisahkan secara linear. Kernel ini dikenal memiliki performa yang baik dengan parameter tertentu, dan hasil dari pelatihan memiliki nilai error yang kecil dibandingkan dengan kernel lainnya. Fungsi kernel rbf

digunakan untuk klasifikasi data non-linear. Berikut persamaan dari kernel *Radial Basis Function (RBF)* :

$$K(x_i, x_j) = \exp\{-\|x_i - x_j\|_2^2 / \sigma^2\} \quad (16)$$

Keterangan :

σ : Nilai *spread*

x_i : Nilai atribut i

x_j : Nilai atribut j

2.10 Parameter SVM

1. Parameter Cost (C)

Parameter Cost (C) adalah parameter yang bekerja sebagai pengoptimalan SVM untuk menghindari misklasifikasi di setiap sampel dalam dataset training. Ketika nilai C terlalu besar, algoritma mencoba untuk mengurangi misklasifikasi atau kesalahan klasifikasi sebanyak mungkin. Hal ini akan menyebabkan hilangnya sifat generalisasi dari classifier (algoritma). Jika C terlalu besar akan menyebabkan batas keputusan (*decision boundary*) menjadi sangat kecil. Ketika nilai C terlalu kecil, kesalahan klasifikasi titik data akan terjadi karena batas keputusan (*decision boundary*) yang lebih luas. Batas keputusan (*decision boundary*) yang lebih luas dapat digeneralisasikan dengan baik pada data pelatihan dan pengujian tetapi dapat mengklasifikasikan beberapa data yang tidak benar.

2. Parameter *gamma*

Parameter gamma menentukan seberapa jauh pengaruh dari satu sampel dataset pelatihan. Dengan nilai gamma tinggi, rincian yang tepat dari batas keputusan (*decision boundary*) hanya akan tergantung pada titik-titik yang sangat dekat dengannya. Sedangkan, ketika nilai gamma yang rendah menunjukkan bahwa bahkan titik yang jauh pun dipertimbangkan ketika memutuskan di mana harus batas keputusan (*decision boundary*) seharusnya berada.

3. Parameter *degree*

Parameter *degree* adalah derajat dari fungsi kernel polynominal yang berfungsi untuk mencari nilai optimal pada setiap dataset. Semakin besar

nilai *degree* maka akurasi sistem yang dihasilkan akan fluktuatif dan kurang stabil. Hal ini terjadi karena semakin tinggi nilai parameter *degree* maka semakin melengkung garis hyperplane yang dihasilkan.

2.11 K-Fold Cross Validation

Cross validation adalah metode statistik yang digunakan untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan cara membagi data menjadi dua bagian: satu digunakan untuk belajar atau melatih model, satu untuk menguji model tersebut (Refaeilzadeh, et al., 2009). *5-fold cross validation* dan *10-fold cross validation* adalah *cross validation* yang paling sering digunakan. Nilai k ditentukan untuk proses dan pembagian model data *training* dan data *testing* dari *cross validation* berikutnya. Dengan menggunakan metode *k-fold cross-validation*, dapat digunakan untuk mengukur kualitas dari model klasifikasi yang dibangun. (Suyanto, 2019).

K-Fold Cross Validation adalah salah satu dari jenis pengujian cross validation yang berfungsi untuk menilai kinerja proses sebuah metode algoritme dengan membagi sampel data secara acak dan mengelompokkan data tersebut sebanyak nilai K k-fold. (Jiang, Ping., 2017).

DATA					
Iterasi 1	1	2	3	4	5
Iterasi 2	1	2	3	4	5
Iterasi 3	1	2	3	4	5
Iterasi 4	1	2	3	4	5
Iterasi 5	1	2	3	4	5
Data Training			Data Testing		

Gambar 2. 4 Ilustrasi *5-fold cross validation*

Pada gambar 2.4 menunjukkan Ilustrasi metode *k-fold cross validation* k = 5 dalam membagi dataset sebanyak 5 subset yang akan menjadi data *training* dan *testing*. Setelah dilakukan proses pembagian dataset sebanyak 5 subset, maka akan dilakukan lagi pengulangan(iterasi) sebanyak 5 kali untuk *training* dan *testing*.

Setiap proses pengulangan (iterasi) data *testing* akan berjumlah 1 subset, dan 4 subset lain akan menjadi data *training*.

Berikut merupakan proses *k-fold cross validation* dengan menggunakan $k = 5$:

1. iterasi 1 himpunan data D1 adalah data testing dan D2, D3, D4, D5, adalah data training.
2. iterasi 2 himpunan data D2 adalah data testing dan D1, D3, D4, D5, adalah data training.
3. iterasi 3 himpunan data D3 adalah data testing dan D1, D2, D4, D5, adalah data training.
4. iterasi 4 himpunan data D4 adalah data testing dan D1, D2, D3, D5, adalah data training.
5. iterasi 5 himpunan data D5 adalah data testing dan D1, D2, D3, D4, adalah data training.

Untuk performa klasifikasi dilakukan dengan cara membandingkan seluruh data uji yang diklasifikasikan benar dengan banyaknya data uji.

$$\text{akurasi} = \frac{\sum \text{klasifikasi benar}}{\sum \text{data uji}} \times 100\% \quad (19)$$

Dalam *k-fold cross validation*, simpangan baku (*standard deviation*) juga akan dihitung. Simpangan baku merupakan ukuran penyebaran data yang menunjukkan jarak rata-rata dari nilai tengah ke suatu titik nilai. Semakin besar simpangan baku yang dihasilkan, maka penyebaran dari nilai tengahnya juga besar, begitu juga sebaliknya. (Tempola et al, 2017). Tujuan dihitungnya simpangan baku yaitu untuk melihat jarak antara akurasi setiap percobaan dengan rata-rata akurasi. (Tempola et al, 2017).

$$\sigma = \sqrt{\frac{\sum (xi - \mu)^2}{N}} \quad (20)$$

Persamaan 2.21. Simpangan baku (*standard deviation*).

Keterangan :

N = banyaknya percobaan

μ = mean

- x = percobaan ke- i
 i = indeks setiap percobaan

Dalam jurnal yang ditulis oleh prangga (2017), Menurut pendapat Sanchez, Bolon dan Alonso, untuk pembagian data testing dan training, k -fold yang biasa digunakan yaitu $k = 3$, $k = 5$ dan $k = 10$. Penelitian ini menggunakan nilai k -fold $k = 3$, $k = 5$, $k = 10$.

2.12 Confusion Matrix

Confusion Matrix merupakan metode external evaluasi yang berisi informasi aktual dan dapat diprediksi (Kohavi dan Provost, 1998), sehingga evaluasi kinerja sistem dapat menggunakan data dalam matriks. Terdapat beberapa ukuran yang dapat digunakan dalam menilai atau mengevaluasi model klasifikasi seperti *accuracy* atau tingkat pengenalan, *error rate* atau tingkat kesalahan, *recall* atau *sensitivity* atau *true positive rate*, *specificity* atau *true negative rate*, *precision*, *F-measure* atau F_1 atau *F-score* atau rata-rata harmonik dari *precision* dan *recall*, serta F_β . Han, et al., (2012).

Tabel 2.3 Ukuran evaluasi model klasifikasi

No	Ukuran	Rumus
1	<i>Accuracy</i> atau tingkat pengenalan	$\frac{TP + TN}{P + N}$
2	<i>Error rate</i> atau tingkat kesalahan	$\frac{FP + FN}{P + N}$
3	<i>Recall</i> atau <i>true positive rate</i>	$\frac{TP}{P}$
4	<i>Specificity</i> atau <i>true negative</i>	$\frac{TN}{N}$
5	<i>Precision</i>	$\frac{TP}{TP + FP}$
6	F atau F_1	$\frac{2 \times precision \times recall}{precision + recall}$
7	F_β , di mana β adalah sebuah bilangan riil non-negatif	$\frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

Tabel 2.4 Contoh *Confusion Matrix*

		Kelas hasil prediksi	
		Positif	Negatif
Kelas aktual	Positif	TP	FN
	Negatif	FP	TN
Jumlah		P	N

Keterangan :

TP : Jumlah prediksi benar dan aktual benar

FN : Jumlah prediksi salah dan aktual benar

FP : Jumlah prediksi benar dan aktual salah

TN : Jumlah Prediksi salah dan aktual salah

Dalam *confusion matrix*, Perhitungan nilai akurasi, *precision* dan *recall* menggunakan rumus berikut :

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+FP+TN} \times 100\% \quad (21)$$

$$\text{Precision} = \frac{TP}{TP+FN} \quad (22)$$

$$\text{Recall} = \frac{TP}{TP+FP} \quad (23)$$

BAB III

METODOLOGI PENELITIAN

3.1 Pengumpulan Data

Pada penelitian ini, data yang diperoleh merupakan *Ulasan* dari website *Google Play Store*, yaitu data *ulasan* pengguna aplikasi *Game Mobile Legends: Bang-Bang* dengan bahasa Indonesia. Dari proses *Scraping* data menggunakan *tools Google Colab*, diperoleh 5000 *record ulasan* yang kemudian di ekspor dalam bentuk format file csv. Data diambil sebanyak 5000 data mentah dalam rentan waktu Desember 2021 sampai Juni 2022. kemudian data tersebut dibagi menjadi 2 untuk data *training* dan data *testing*. Berikut sampel dari data Ulasan :

Tabel 3.1 Contoh Data *Ulasan*.

userName	score	At	Review
Rezki Raplisal Siregar	1	12/27 /2021 18:17	Semenjak update patch baru tiba2 frame drop,padahal sebelumnya tidak ada masalah seperti itu. Spek hp pun mempunyai tapi kadang terjadi frame drop,dan pada mode refresh rate sangat berbeda dengan HFR sebelum update. Tolong segera di update
cika amelia putri	3	12/28 /2021 1:45	Tolong dong kasih tau masuk mode Rank, sya udh level 8 dan udh punya 5 hero, ada tutorial untuk masuk mode Rank, sya ikuti tapi ketika ingin bermain tidak bisa hanya ada bacaan "Tap di sini untuk Mode Rank..".. Ketika di Tap tidak bisa apa2,tolong yh kasih tau.. Kalau udh bisa nanti ditambah lagi Bintang nya
Alessandr o Kotalawala a	1	12/28 /2021 5:27	Match makingnya gak bener, random player yang dipilih selalu merugikan team. Sistem solo mohon diperbaiki, yang baru main ketemu yg baru main, saran sistem match making dilihat dri statistik dan jumlah match, terimakasih.
Dheno gaming center	1	12/28 /2021 12:13	Sistem match tidak seimbang . Dan hero terkadang ada yg powernya tidak masuk akal . Bertahun tahun main tidak ada perubahan dengan sistem match yang tidak seimbang . Se akan akan tim adalah bot sedangkan lawan pro player
Zaky Achsanul	1	12/28 /2021 15:18	Untuk update selanjutnya mungkin perlu fitur pengompresan data,hal ini sangat membantu player2 yang memiliki device pas2an,memang grafiknya sangat bagus daripada versi sebelumnya tapi yang membuat nyaman adalah pengalaman bermain yang optimal dengan grafik visual rendah tetapi masih bisa terlihat(warnanya dibuat lebih mencolok) stabilitas jaringan

			yang bagus (ini mungkin termasuk ke sumber2 data/WiFi masing-masing) dan refresh rate yang optimal/stabil, itu sangat membantu saat mencapai late game
Vania Stephani	4	12/30 /2021 3:04	Game-nya bagus.. Cuma tolong perbaiki sinyal-nya, padahal jaringan + sinyal saya bagus, buka apk yg lain lancar jaya, tapi kenapa pas main ML malah jadi 200+ms ,nge-frame, dan patah-patah.. Kadang juga sinyal bagus-nya pas saya mati doang.. pas lagi war sama musuh malah sinyal merah.. Tolong diperbaikiðŸ™ðŸ¶» Terimakasih.
Nanthul	1	12/30 /2021 17:33	Update terbaru game jd tambah berat, RAM 4Gb dan ROM masih sekitar lebih dari 20GB berat bgt buat main. Apalagi pas war, grafik jd patah2. Settingan sudah low semua. Versi sebelumnya gk kayak gini.
Near Queen Azkiah	5	12/30 /2021 22:08	Game nya seru,,,tapi maaf untuk kepada pihak moonton untuk memperbaiki keseimbangan game contoh setiap kali saya bermain saya bertemu musuh yang rank nya lebih jauh di bandingkan saya, contoh rank musuh gm/epic dan tentunya itu tdk seimbang. Mohon di perbaiki lagiðŸ™
Muhammad Nauramu htaromi	5	12/31 /2021 2:07	Baru baru ini ml update tambah berat dan menambah fps drop saat bermain itu tidak nyaman sekali, Padahal hp sudah ram 4 dan penyimpanan masih banyak yang kosong. Lah ini saat saya mencentang refresh rate yang tinggi dan saya tutup pengaturan lalu saya buka lagi centang nya tidak ada. Tolong dong moonton optimal kan lagi
Ahmad Khalilullah	5	1/6/2022 11:45	Gamenya bagus sekali, tapi ada beberapa hal yang membuat saya kesel, karena pengurangan kredit skor yang tidak masuk akal. Jadi setelah draft saat di tempat yang loading gambarnya hero, saya sudah 100% tapi lawan saya masih 99% lama banget, trs saya relog malah saya yang diitung afk dan kredit skor saya dari 97 jadi 89 dan jadi tidak bisa rank, tolong banget moonton perbaiki permasalahan ini.

3.2 Tahapan Penelitian

3.1.1 Studi Pustaka

Penulis mencantumkan beberapa teori terkait dengan penelitian yang dilakukan, yang meliputi *analisis sentimen*, *preprocessing*, pembobotan kata(*weight*), algoritma *SVM*, *K-Fold Cross Validation* dan *confusion matrix*. Penulis juga menggunakan literatur yang berasal dari jurnal ilmiah, karya ilmiah dan buku sebagai referensi dalam penelitian ini.

3.1.2 Pengumpulan Data

Data yang diambil merupakan data Ulasan yang di *scrapping* menggunakan *tools Google Colab* sebanyak 5000 ulasan. Data tersebut diambil dalam kurun waktu 7 bulan, yaitu desember 2021 sampai juni 2022.

3.1.3 Rancangan Sistem

Implementasi perangkat yang digunakan, sebagai berikut :

- Perangkat keras
 - a. Processor : AMD Ryzen 5 3550H
 - b. RAM : 16 GB
 - c. SSD : 1 TB
 - d. GPU : GTX 1050
 - e. VRAM : 3 GB

3.1.4 Pembuatan alat Uji

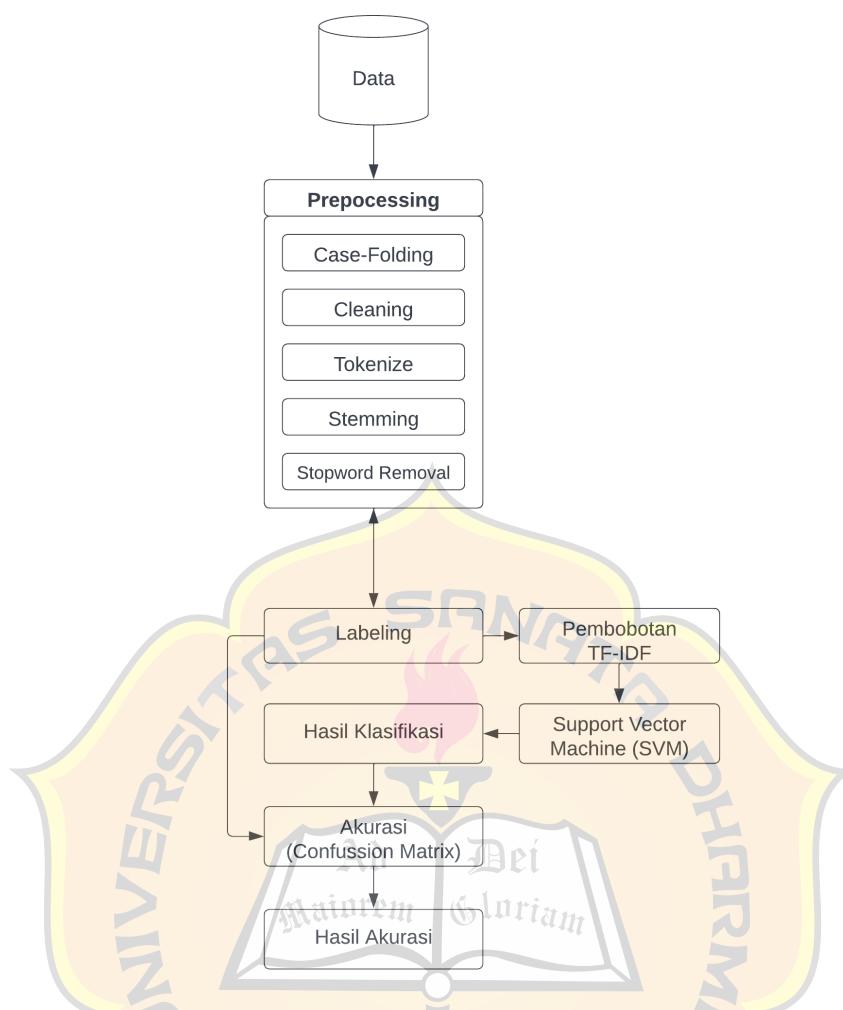
Tahap pembuatan alat uji adalah tahap untuk merancang alat uji dalam menguji klasifikasi dari algoritma *Support Vector Machine* (SVM) dalam mngklasifikasikan *Ulasan* yang sudah diperoleh dari *Google Play Store*.

3.1.5 Pengujian

Tahap pengujian merupakan tahap menguji alat yang telah dirancang pada tahap pembuatan alat uji. Sebelum dilakukan pengujian, Data terlebih dahulu melewati proses *preprocessing* dan labeling menggunakan *Vader Lexicon*.

3.3 Skema Desain

Berikut merupakan skema desain dalam penelitian dan perancangan sistem untuk melakukan analisis sentimen *ulasan Game Mobile Legends di Google Play Store* menggunakan algoritma *SVM* :



Gambar 3. 1 Desain Skema

Berdasarkan Gambar 3.1 desain skema, tahap pertama adalah pencarian data *ulasan Game Mobile Legends* di *Google Play Store*. Setelah mendapat data, kemudian data tersebut akan diolah melalui tahap *preprocessing*. Tahapan *preprocessing* data meliputi proses *case-folding*, *cleaning*, *tokenize*, *stemming* dan *stopword-removal*.

Setelah melewati tahap *preprocessing*, data akan masuk ke tahap pelabelan menggunakan *Vader Lexicon*. Kemudian data akan masuk ketahap pembobotan kata (TF-IDF). Setiap kata diberi bobot untuk masuk kedalam tahap klasifikasi. Tahap selanjutnya adalah tahap klasifikasi menggunakan algoritma *Support Vector Machine (SVM)*. Setelah data melewati tahap klasifikasi *SVM*, data kemudian dibandingkan dengan data hasil pelabelan *Vader Lexicon* menggunakan *confusion matrix* sehingga mendapatkan akurasi terbaik.

3.4 Preprocessing

Tahap preprocessing pada penelitian ini meliputi *case-folding*, *cleaning*, *tokenize*, *stemming* dan *stopword removal*.

3.4.1 Case-Folding

Tahap *case-folding* adalah tahap untuk merubah semua huruf menjadi huruf kecil (*lowercase*). Tahapan *case-folding* dapat dijabarkan sebagai berikut :

- Sistem membaca setiap baris dalam dokumen
- Jika terdapat huruf kapital, maka akan ganti ke huruf kecil. Jika tidak ditemukan, sistem lanjut membaca pada baris berikutnya dalam dokumen.

Tabel 3.2 Hasil Tahap *Case-Folding*

Sebelum Case-Folding	Setelah Case-Folding
Tolong lah masalah teman satu team nya di kondisi kan lagi!! Yg sesuai sama skill pemain.	tolong lah masalah teman satu team nya di kondisi kan lagi!! yg sesuai sama skill pemain.
Game MOBA yang bagus untuk dimainkan apalagi ada 2 hero yang mewakili Indonesia Gatotkaca dan Kadita.	game moba yang bagus untuk dimainkan apalagi ada 2 hero yang mewakili indonesia gatotkaca dan kadita.
RESOURCE Tidak bisa di download! Mohon di perbaiki karna saya juga tidak bisa mendownload resource padahal berada di jaringan yang bagus & memori ponsel masih banyak	resource tidak bisa di download! mohon di perbaiki karna saya juga tidak bisa mendownload resource padahal berada di jaringan yang bagus & memori ponsel masih banyak
GAME YG SANGAT TIDAK RECOMMENDED, saya sudah 2 tahun main ml dan cukup kecewa dgn ml yg sekarang, TIDAK RECOMMENDED	game yg sangat tidak recommended, saya sudah 2 tahun main ml dan cukup kecewa dgn ml yg sekarang, tidak recommended
Maaf MONTON saya punya keluhan Setelah UPDATE tadi siang kok malah jadi ngelag Ama patah patah ya	maaf monton saya punya keluhan setelah update tadi siang kok malah jadi ngelag ama patah patah ya

3.4.2 Cleaning

Pada tahap *cleaning*, data yang dibersihkan hanya pada atribut *Review*. Data akan dibersihkan dari bagian-bagian yang tidak diperlukan (*noise*) seperti tanda baca, angka, karakter dan emoticon. Tahap *cleaning* dapat dijabarkan sebagai berikut :

- Sistem membaca setiap baris dalam dokumen
- Jika terdapat *noise* pada dokumen, maka *noise* akan di bersihkan/dihapus. Jika tidak ditemukan, sistem lanjut membaca pada baris berikutnya dalam dokumen.

Tabel 3.3 Hasil Tahap *Cleaning*

Sebelum Cleaning	Setelah Cleaning
tolong lah masalah teman satu team nya di kondisi kan lagi!! yg sesuai sama skill pemain.	tolong lah masalah teman satu team nya di kondisi kan lagi yg sesuai sama skill pemain
game moba yang bagus untuk dimainkan apalagi ada 2 hero yang mewakili indonesia gatotkaca dan kadita.	game moba yang bagus untuk dimainkan apalagi ada hero yang mewakili indonesia gatotkaca dan kadita
resource tidak bisa di download! mohon di perbaiki karna saya juga tidak bisa mendownload resource padahal berada di jaringan yang bagus & memori ponsel masih banyak	resource tidak bisa di download mohon di perbaiki karna saya juga tidak bisa mendownload resource padahal berada di jaringan yang bagus memori ponsel masih banyak
game yg sangat tidak recommended, saya sudah 2 tahun main ml dan cukup kecewa dgn ml yg sekarang, tidak recommended	game yg sangat tidak recommended saya sudah tahun main ml dan cukup kecewa dgn ml yg sekarang tidak recommended
maaf mnton saya punya keluhan setelah update tadi siang kok malah jadi ngelag ama patah patah ya	maaf mnton saya punya keluhan setelah update tadi siang kok malah jadi ngelag ama patah patah ya

3.4.3 Tokenize

Tokenize adalah operasi memisahkan teks menjadi potongan-potongan berupa token. Tahap *tokenize* dapat dijabarkan sebagai berikut :

- Sistem membaca setiap baris dalam dokumen

- Sistem mengambil kata dalam kalimat kemudian dipisahkan kata-perkata menjadi token dengan menambahkan spasi.
- Sistem menyimpan hasil dalam bentuk token.

Tabel 3.4 sebelum *Tokenize*

Sebelum Tokenize
tolong lah masalah teman satu team nya di kondisi kan lagi yg sesuai sama skill pemain
game moba yang bagus untuk dimainkan apalagi ada hero yang mewakili indonesia gatotkaca dan kadita
resource tidak bisa di download mohon di perbaiki karna saya juga tidak bisa mendownload resource padahal berada di jaringan yang bagus memori ponsel masih banyak
game yg sangat tidak recommended saya sudah tahun main ml dan cukup kecewa dgn ml yg sekarang tidak recommended
maaf mnton saya punya keluhan setelah update tadi siang kok malah jadi ngelag ama patah patah ya

Tabel 3.5 Setelah *Tokenize*

Setelah Tokenize
[‘tolong’, ‘lah’, ‘masalah’, ‘teman’, ‘satu’, ‘team’, ‘nya’, ‘di’, ‘kondisi’, ‘kan’, ‘lagi’, ‘yg’, ‘sesuai’, ‘sama’, ‘skill’, ‘pemain’]
[‘game’, ‘moba’, ‘yang’, ‘bagus’, ‘untuk’, ‘dimainkan’, ‘apalagi’, ‘ada’, ‘hero’, ‘yang’, ‘mewakili’, ‘indonesia’, ‘gatotkaca’, ‘dan’, ‘kadita’]
[‘resource’, ‘tidak’, ‘bisa’, ‘di’, ‘download’, ‘mohon’, ‘di’, ‘perbaiki’, ‘karna’, ‘saya’, ‘juga’, ‘tidak’, ‘bisa’, ‘mendownload’, ‘resource’, ‘padahal’, ‘berada’, ‘di’, ‘jaringan’, ‘yang’, ‘bagus’, ‘memori’, ‘ponsel’, ‘masih’, ‘banyak’]
[‘game’, ‘yg’, ‘sangat’, ‘tidak’, ‘recommended’, ‘saya’, ‘sudah’, ‘tahun’, ‘main’, ‘ml’, ‘dan’, ‘cukup’, ‘kecewa’, ‘dgn’, ‘ml’, ‘yg’, ‘sekarang’, ‘tidak’, ‘recommended’]
[‘maaf’, ‘nton’, ‘saya’, ‘punya’, ‘keluhan’, ‘setelah’, ‘update’, ‘tadi’, ‘siang’, ‘kok’, ‘malah’, ‘jadi’, ‘ngelag’, ‘ama’, ‘patah’, ‘patah’, ‘ya’]

3.4.4 Stemming

Stemming adalah proses untuk mencari kata dasar(*root word*) dari kata yang memiliki imbuhan. Tahap *stemming* dapat dijabarkan sebagai berikut :

- Sistem membaca setiap baris dalam dokumen
- Sitem mengecek setiap token dalam dokumen menggunakan kamus *stemming*.
- Token dibandingkan dengan kamus *stemming*. Jika ditemukan kata berimbuhan pada token, token akan diganti dengan kata dasar (*root word*). Jika token tidak memiliki imbuhan, maka sistem akan lanjut ke pengecekan *stemming* pada token selanjutnya.

Tabel 3.6 Hasil Tahap *Stemming*

Sebelum Stemming	Setelah Stemming
[‘tolong’, ‘lah’, ‘masalah’, ‘teman’, ‘satu’, ‘team’, ‘nya’, ‘di’, ‘kondisi’, ‘kan’, ‘lagi’, ‘yg’, ‘sesuai’, ‘sama’, ‘skill’, ‘emain’]	[‘tolong’, ‘lah’, ‘salah’, ‘teman’, ‘satu’, ‘team’, ‘nya’, ‘di’, ‘kondisi’, ‘kan’, ‘lagi’, ‘yg’, ‘sesuai’, ‘sama’, ‘skill’, ‘main’]
[‘game’, ‘moba’, ‘yang’, ‘bagus’, ‘untuk’, ‘dimainkan’, ‘apalagi’, ‘ada’, ‘hero’, ‘yang’, ‘mewakili’, ‘indonesia’, ‘gatotkaca’, ‘dan’, ‘kadita’]	[‘game’, ‘moba’, ‘yang’, ‘bagus’, ‘untuk’, ‘main’, ‘apalagi’, ‘ada’, ‘hero’, ‘yang’, ‘wakil’, ‘indonesia’, ‘gatotkaca’, ‘dan’, ‘kadita’]
[‘resource’, ‘tidak’, ‘bisa’, ‘di’, ‘download’, ‘mohon’, ‘di’, ‘perbaiki’, ‘karna’, ‘saya’, ‘juga’, ‘tidak’, ‘bisa’, ‘mendownload’, ‘resource’, ‘padahal’, ‘berada’, ‘di’, ‘jaringan’, ‘yang’, ‘bagus’, ‘memori’, ‘ponsel’, ‘masih’, ‘banyak’]	[‘resource’, ‘tidak’, ‘bisa’, ‘di’, ‘download’, ‘mohon’, ‘di’, ‘baik’, ‘karna’, ‘saya’, ‘juga’, ‘tidak’, ‘bisa’, ‘download’, ‘resource’, ‘padahal’, ‘berada’, ‘di’, ‘jaring’, ‘yang’, ‘bagus’, ‘memori’, ‘ponsel’, ‘masih’, ‘banyak’]
[‘game’, ‘yg’, ‘sangat’, ‘tidak’, ‘recommended’, ‘saya’, ‘sudah’, ‘tahun’, ‘main’, ‘ml’, ‘dan’, ‘cukup’, ‘kecewa’, ‘dgn’, ‘ml’, ‘yg’, ‘sekarang’, ‘tidak’, ‘recommended’]	[‘game’, ‘yg’, ‘sangat’, ‘tidak’, ‘recommended’, ‘saya’, ‘sudah’, ‘tahun’, ‘main’, ‘ml’, ‘dan’, ‘cukup’, ‘kecewa’, ‘dgn’, ‘ml’, ‘yg’, ‘sekarang’, ‘tidak’, ‘recommended’]
[‘maaf’, ‘monton’, ‘saya’, ‘punya’, ‘keluhan’, ‘setelah’, ‘update’, ‘tadi’, ‘siang’, ‘kok’, ‘malah’, ‘jadi’, ‘ngelag’, ‘ama’, ‘patah’, ‘patah’, ‘ya’]	[‘maaf’, ‘monton’, ‘saya’, ‘punya’, ‘keluh’, ‘setelah’, ‘update’, ‘tadi’, ‘siang’, ‘kok’, ‘malah’, ‘jadi’, ‘ngelag’, ‘ama’, ‘patah’, ‘patah’, ‘ya’]

3.4.5 Stopword Removal

Tahap *stopword removal* adalah tahap untuk menghapus kata sambung atau kata *irrelevant*. Tahap *stopword* dapat dijabarkan sebagai berikut :

- Baca setiap baris dalam dokumen
- Mengecek setiap token dalam dokumen menggunakan kamus *stopword*.
- Jika token terdapat dalam kamus *stopword*, maka token dihapus, jika tidak ditemukan, maka akan lanjut ke pengecekan token berikutnya.

Tabel 3.7 Hasil Tahap *Stopword Removal*

Sebelum Stopword	Setelah Stopword
[‘tolong’, ‘lah’, ‘salah’, ‘teman’, ‘satu’, ‘team’, ‘nya’, ‘di’, ‘kondisi’, ‘kan’, ‘lagi’, ‘yg’, ‘sesuai’, ‘sama’, ‘skill’, ‘main’]	[‘lah’, ‘salah’, ‘teman’, ‘satu’, ‘team’, ‘nya’, ‘kondisi’, ‘kan’, ‘yg’, ‘sesuai’, ‘sama’, ‘skill’, ‘main’]
[‘game’, ‘moba’, ‘yang’, ‘bagus’, ‘untuk’, ‘main’, ‘apalagi’, ‘ada’, ‘hero’, ‘yang’, ‘wakil’, ‘indonesia’, ‘gatotkaca’, ‘dan’, ‘kadita’]	[‘game’, ‘moba’, ‘bagus’, ‘main’, ‘hero’, ‘wakil’, ‘indonesia’, ‘gatotkaca’, ‘kadita’]
[‘resource’, ‘tidak’, ‘bisa’, ‘di’, ‘download’, ‘mohon’, ‘di’, ‘baik’, ‘karna’, ‘saya’, ‘juga’, ‘tidak’, ‘bisa’, ‘download’, ‘resource’, ‘padahal’, ‘berada’, ‘di’, ‘jaring’, ‘yang’, ‘bagus’, ‘memori’, ‘ponsel’, ‘masih’, ‘banyak’]	[‘resource’, ‘tidak’, ‘download’, ‘mohon’, ‘baik’, ‘saya’, ‘tidak’, ‘bisa’, ‘download’, ‘resource’, ‘padahal’, ‘berada’, ‘jaring’, ‘bagus’, ‘memori’, ‘ponsel’, ‘banyak’]
[‘game’, ‘yg’, ‘sangat’, ‘tidak’, ‘recommended’, ‘saya’, ‘sudah’, ‘tahun’, ‘main’, ‘ml’, ‘dan’, ‘cukup’, ‘kecewa’, ‘dgn’, ‘ml’, ‘yg’, ‘sekarang’, ‘tidak’, ‘recommended’]	[‘game’, ‘yg’, ‘sangat’, ‘tidak’, ‘recommended’, ‘saya’, ‘tahun’, ‘main’, ‘ml’, ‘cukup’, ‘kecewa’, ‘dgn’, ‘ml’, ‘yg’, ‘sekarang’, ‘tidak’, ‘recommended’]
[‘maaf’, ‘monton’, ‘saya’, ‘punya’, ‘keluh’, ‘setelah’, ‘update’, ‘tadi’, ‘siang’, ‘kok’, ‘malah’, ‘jadi’, ‘ngelag’, ‘ama’, ‘patah’, ‘patah’, ‘ya’]	[‘maaf’, ‘monton’, ‘saya’, ‘punya’, ‘keluh’, ‘setelah’, ‘update’, ‘tadi’, ‘siang’, ‘kok’, ‘malah’, ‘jadi’, ‘ngelag’, ‘ama’, ‘patah’, ‘patah’]

3.5 Terjemahan Data dan Labeling Vader Lexicon

Setelah melewati proses *preprocessing*, selanjutnya data akan diterjemahkan terlebih dahulu menggunakan onlinedoctranslator.com ke dalam bahasa Inggris sebelum masuk ke tahap pelabelan menggunakan *Vader Lexicon*.

Tabel 3. 8 Terjemahan Data

Sebelum	Setelah
lah salah teman satu team nya kondisi kan yg sesuai sama skill main	it's one of his team mates, the conditions are in accordance with playing skills
game moba bagus main hero wakil indonesia gatotkaca kadita	good moba game playing the hero representative of Indonesia Gatotkaca Kadita
resource tidak download mohon baik saya tidak bisa download resource padahal berada jaring bagus memori ponsel banyak	the resource doesn't download please ok I can't download the resource even though there is a good net with a lot of cellphone memory
game yg sangat tidak recommended saya tahun main ml cukup kecewa dgn ml yg sekarang tidak recommended	a game that is not really recommended, I have been playing ml quite disappointed with ml which is now not recommended
maaf mownton saya punya keluh setelah update tadi siang kok malah jadi ngelag ama patah patah	sorry mownton I have a complaint after updating this afternoon how come it's lagging and broken

Tabel 3. 9 Hasil Pelabelan *Vader Lexicon*

Review	Pelabelan <i>Vader Lexicon</i>
it's one of his team mates, the conditions are in accordance with playing skills	Positif
good moba game playing the hero representative of Indonesia Gatotkaca Kadita	Positif
the resource doesn't download please ok I can't download the resource even though there is a good net with a lot of cellphone memory	Negatif
a game that is not really recommended, I have been playing ml quite disappointed with ml which is now not recommended	Negatif
sorry mownton I have a complaint after updating this afternoon how come it's lagging and broken	Negatif

3.6 Pembobotan TF-IDF

Sebelum masuk ke tahap klasifikasi, setiap kata dalam dokumen akan diberi bobot dengan metode TF-IDF.

1. Term Frequency (TF)

Term Frequency (TF) adalah tahap untuk menghitung jumlah kata yang terdapat dalam dokumen. Berikut contoh pembobotan TF-IDF sebanyak 10 dokumen (d1 – d10).

Tabel 3.10 *Term Frequency* (TF)

Kata	Banyak Kata Pada Tiap Dokumen (TF)									
	1	2	3	4	5	6	7	8	9	10
game	1	1	1	1	1	1	1	1	1	1
ini	1									
yg	1									
bisa		1					1			
main	1									
tidak	1						1		1	
tolong	1									
tapi	1									
bagus	1									
ada	1									
sangat			1							
tim			1							
untuk			1							
jaringan				1						
gak				1						
itu					1					
player					1					
padahal						1				
update						1				1
perbaiki						1				
juga							1			
aja								1		
jadi								1	1	
lagi							1			
dari								1		
banyak								1		
sinyal								1		
sama									1	
moonton									1	
sering									1	
mobile										1
ga										2
legends										1
udah										1
kasih										1
terus										1
dengan										1

2. Document Frequent (DF)

Document Frequent (DF) adalah tahap untuk menghitung jumlah dokumen yang terdapat kata yang dimaksud. *document frequent* diinisialisasikan dengan (df).

Tabel 3.11 DF(*Document Frequent*)

Kata	Banyak Kata Pada Tiap Dokumen (TF)										df
	1	2	3	4	5	6	7	8	9	10	
game	1	1	1	1	1	1	1	1	1	1	10
ini	1										1
yg	1										1
bisa		1					1				2
main		1									1
tidak		1					1		1		3
tolong		1									1
tapi		1									1
bagus		1									1
ada		1									1
sangat			1								1
tim			1								1
untuk			1								1
jaringan				1							1
gak				1							1
itu				1							1
player				1							1
padahal					1						1
update					1					1	2
perbaiki					1						1
juga						1					1
aja							1				1
jadi							1		1		2
lagi							1				1
dari								1			1
banyak							1				1
sinyal							1				1
sama								1			1
moonton								1			1
sering								1			1
mobile										1	1
ga										2	1
legends										1	1
udah										1	1
kasih										1	1
terus										1	1
dengan										1	1

3. Inverse Document Frequency (IDF)

Perhitungan *Inverse Document Frequency* (IDF) dapat dilakukan setelah mendapat nilai dari perhitungan DF(*Document Frequent*). Perhitungan *Inverse Document Frequency* (IDF) menggunakan persamaan :

$$\text{IDF} = \log_{10}/df \quad (24)$$

Tabel 3.12 *Inverse Document Frequency* (IDF)

df	n/df	idf
10	1	0
1	10	1
1	10	1
2	5	0.69897
1	10	1
3	3.333333	0.522879
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
2	5	0.69897
1	10	1
1	10	1
1	10	1
2	5	0.69897
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1
1	10	1

4. Pembobotan Kata (TF-IDF)

Pada tahap pembobotan, setiap kata akan di beri bobot. Perdamaan yang digunakan dalam pembobotan :

$$W = TF \times IDF \quad (24)$$

Tabel 3.13 Pembobotan kata (W)

Kata	W = tf-idf									
	1	2	3	4	5	6	7	8	9	10
game	0	0	0	0	0	0	0	0	0	0
ini	1	0	0	0	0	0	0	0	0	0
yg	1	0	0	0	0	0	0	0	0	0
bisa	0	0.7	0	0	0	0	0.7	0	0	0
main	0	1	0	0	0	0	0	0	0	0
tidak	0	0.5	0	0	0	0	0.5	0	0.5	0
tolong	0	1	0	0	0	0	0	0	0	0
tapi	0	1	0	0	0	0	0	0	0	0
bagus	0	1	0	0	0	0	0	0	0	0
ada	0	1	0	0	0	0	0	0	0	0
sangat	0	0	1	0	0	0	0	0	0	0
tim	0	0	1	0	0	0	0	0	0	0
untuk	0	0	1	0	0	0	0	0	0	0
jaringan	0	0	0	1	0	0	0	0	0	0
gak	0	0	0	1	0	0	0	0	0	0
itu	0	0	0	1	0	0	0	0	0	0
player	0	0	0	1	0	0	0	0	0	0
padahal	0	0	0	0	1	0	0	0	0	0
update	0	0	0	0	0.7	0	0	0	0	0.7
perbaiki	0	0	0	0	1	0	0	0	0	0
juga	0	0	0	0	0	1	0	0	0	0
aja	0	0	0	0	0	0	1	0	0	0
jadi	0	0	0	0	0	0	0.7	0	0.7	0
lagi	0	0	0	0	0	0	1	0	0	0
dari	0	0	0	0	0	0	0	1	0	0
banyak	0	0	0	0	0	0	0	1	0	0
sinyal	0	0	0	0	0	0	0	1	0	0
sama	0	0	0	0	0	0	0	0	1	0
moonton	0	0	0	0	0	0	0	0	1	0
sering	0	0	0	0	0	0	0	0	1	0
mobile	0	0	0	0	0	0	0	0	0	1
ga	0	0	0	0	0	0	0	0	0	2
legends	0	0	0	0	0	0	0	0	0	1
udah	0	0	0	0	0	0	0	0	0	1
kasih	0	0	0	0	0	0	0	0	0	1
terus	0	0	0	0	0	0	0	0	0	1
dengan	0	0	0	0	0	0	0	0	0	1

5. Normalisasi (*Cosine Normalization*)

Pada tahap normalisasi (*cosine Normalization*) vektor dokumen akan di normlaisasi dengan tujuan agar proses tidak dipengaruhi oleh perbedaan panjang data. Bobot vektor dari dokumen akan bernilai 0-1.

$$w_{t,d} = \frac{w_{t,d}}{\sqrt{\sum_{t=1}^n w_{t,d}^2}} \quad (25)$$

Tabel 3.14 Cosine Normalization

Kata	W = tf-idf (Cosine Normalization)									
	1	2	3	4	5	6	7	8	9	10
game	0	0	0	0	0	0	0	0	0	0
ini	0.7	0	0	0	0	0	0	0	0	0
yg	0.7	0	0	0	0	0	0	0	0	0
bisa	0	0.3	0	0	0	0	0.4	0	0	0
main	0	0.4	0	0	0	0	0	0	0	0
tidak	0	0.2	0	0	0	0	0.3	0	0.3	0
tolong	0	0.4	0	0	0	0	0	0	0	0
tapi	0	0.4	0	0	0	0	0	0	0	0
bagus	0	0.4	0	0	0	0	0	0	0	0
ada	0	0.4	0	0	0	0	0	0	0	0
sangat	0	0	0.6	0	0	0	0	0	0	0
tim	0	0	0.6	0	0	0	0	0	0	0
untuk	0	0	0.6	0	0	0	0	0	0	0
jaringan	0	0	0	0.5	0	0	0	0	0	0
gak	0	0	0	0.5	0	0	0	0	0	0
itu	0	0	0	0.5	0	0	0	0	0	0
player	0	0	0	0.5	0	0	0	0	0	0
padahal	0	0	0	0	0.6	0	0	0	0	0
update	0	0	0	0	0.4	0	0	0	0	0.2
perbaiki	0	0	0	0	0.6	0	0	0	0	0
juga	0	0	0	0	0	1	0	0	0	0
aja	0	0	0	0	0	0	0.6	0	0	0
jadi	0	0	0	0	0	0	0.4	0	0.4	0
lagi	0	0	0	0	0	0	0.6	0	0	0
dari	0	0	0	0	0	0	0	0.6	0	0
banyak	0	0	0	0	0	0	0	0.6	0	0
sinyal	0	0	0	0	0	0	0	0.6	0	0
sama	0	0	0	0	0	0	0	0	0.5	0
moonton	0	0	0	0	0	0	0	0	0.5	0
sering	0	0	0	0	0	0	0	0	0.5	0
mobile	0	0	0	0	0	0	0	0	0	0.3
ga	0	0	0	0	0	0	0	0	0	0.6
legends	0	0	0	0	0	0	0	0	0	0.3
udah	0	0	0	0	0	0	0	0	0	0.3
kasih	0	0	0	0	0	0	0	0	0	0.3
terus	0	0	0	0	0	0	0	0	0	0.3
dengan	0	0	0	0	0	0	0	0	0	0.3

3.7 Pengujian Metode

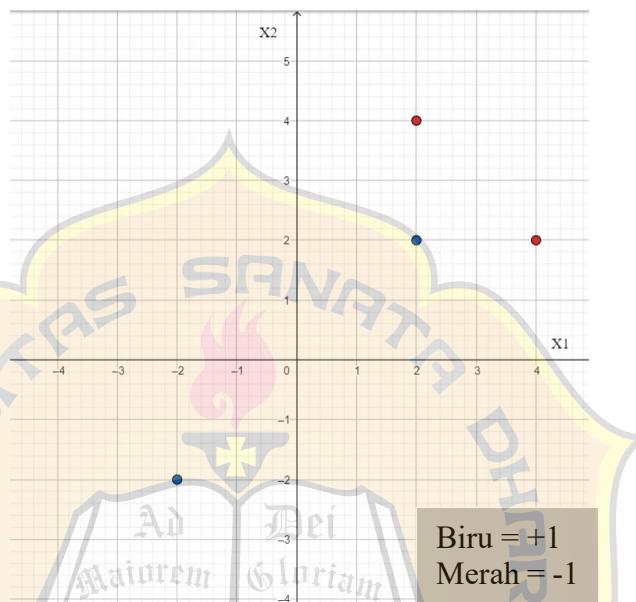
Pada proses ini, data yang digunakan akan diambil secara random sebagai sample untuk perhitungan manual sesuai dengan metode yang dipakai, perhitungan manual ini bertujuan untuk mengetahui metode yang digunakan.

3.7.1 Klasifikasi Metode Support Vector Machine (SVM)

Berikut ini merupakan illustrasi cara kerja *Support Vector Machine* dengan data yang diambil secara random.

Tabel 3.15 Contoh data sampel

x₁	x₂	Label (Y)
2	2	1
2	4	-1
4	2	-1
-2	-2	1



Gambar 3.2 koordinat Data Mentah

Pada tabel 3.15, Terdapat 2 atribut, yaitu x_1 dan x_2 yang akan menghasilkan 2 bobot, yaitu w_1 dan w_2 . Kemudian margin diminimalkan menggunakan rumus pada persamaan (6) dengan syarat sebagai berikut :

$$y_i(w \cdot x_i + b) \geq 1, i = 1, 2, 3, \dots, N \quad (25)$$

$$y_i(w_1 \cdot x_1 + w_2 \cdot x_2 + b) \geq 1 \quad (26)$$

Sehingga diperoleh persamaan sebagai berikut :

- $(2w_1 + 2w_2 + b) \geq 1$, untuk $y_1 = 1, x_1 = 2, x_2 = 2$
- $(-2w_1 - 4w_2 - b) \geq 1$, untuk $y_1 = -1, x_1 = 2, x_2 = 4$
- $(-4w_1 - 2w_2 - b) \geq 1$, untuk $y_1 = -1, x_1 = 4, x_2 = 2$
- $(-2w_1 - 2w_2 + b) \geq 1$, untuk $y_1 = 1, x_1 = -2, x_2 = -2$

Selanjutnya yaitu mencari nilai w dan b dari persamaan

(1) dan (2) sebagai berikut.

$$(2w_1 + 2w_2 + b) \geq 1$$

$$\begin{array}{r} (-2w_1 - 4w_2 - b) \geq 1 \\ \hline -2w_2 = 2 \\ w_2 = -1 \end{array}$$

Kemudian mencari nilai w dan b dari persamaan (1) dan (3) sebagai berikut:

$$\begin{array}{r} (2w_1 + 2w_2 + b) \geq 1 \\ (-4w_1 - 2w_2 - b) \geq 1 \\ \hline -2w_1 = 2 \\ w_1 = -1 \end{array}$$

Sehingga nilai b yang didapat dari persamaan (1) dan (4) yaitu :

$$\begin{array}{r} (2w_1 + 2w_2 + b) \geq 1 \\ (-1w_1 - 2w_2 + b) \geq 1 \\ \hline 2b = 2 \\ b = 1 \end{array}$$

Maka, persamaan *hyperplane* menjadi :

$$w_1x_1 + w_2x_2 + b = 0$$

$$-1x_1 + 1x_2 - 1 = 0$$

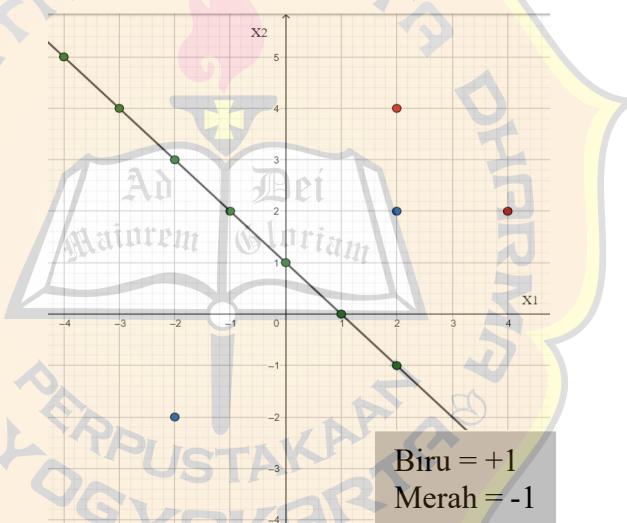
$$x_2 - 1 = x_1$$

Selanjutnya, dengan fungsi $f(x) = -x_1 + x_2 - 1$ dibuat plot *hyperplane* menggunakan data seperti pada Tabel 2.2.

Tabel 3.16 Plot Hyperplane

x_1	$x_2 = 1 - x_1$
-4	5
-3	4
-2	3
-1	2
0	1
1	0
2	-1
3	-2

Berdasarkan tabel 3.16, maka *hyperplane* dapat di visualisasikan seperti berikut :



Gambar 3.3 Visualisasi Hyperlane

3.7.2 Confusion Matrix

Confusion matrix digunakan untuk menghitung akurasi data acak yang telah diberikan label yang akan dibandingkan dengan hasil klasifikasi *SVM*. Perhitungan *accuracy*, *recall*, dan *precision* menggunakan persamaan (21), (22), dan (23).

Berikut merupakan tabel perbandingan label asli dan label SVM:

Tabel 3.17 Perbandingan label asli dan SVM

Data Mentah	Akurasi	
	Asli	SVM
1	-1	-1
2	1	1
3	1	1
4	1	-1
5	-1	-1

Tabel 3.18 *Confusion Matrix SVM*

Confusion Matrix		
		Actual Value
		P N
Predicted Value	P	2 1
	N	0 2

Berikut merupakan hasil perhitungan *accuracy*, *recall*, dan *precision* menggunakan persamaan (21), (22), dan (23) :

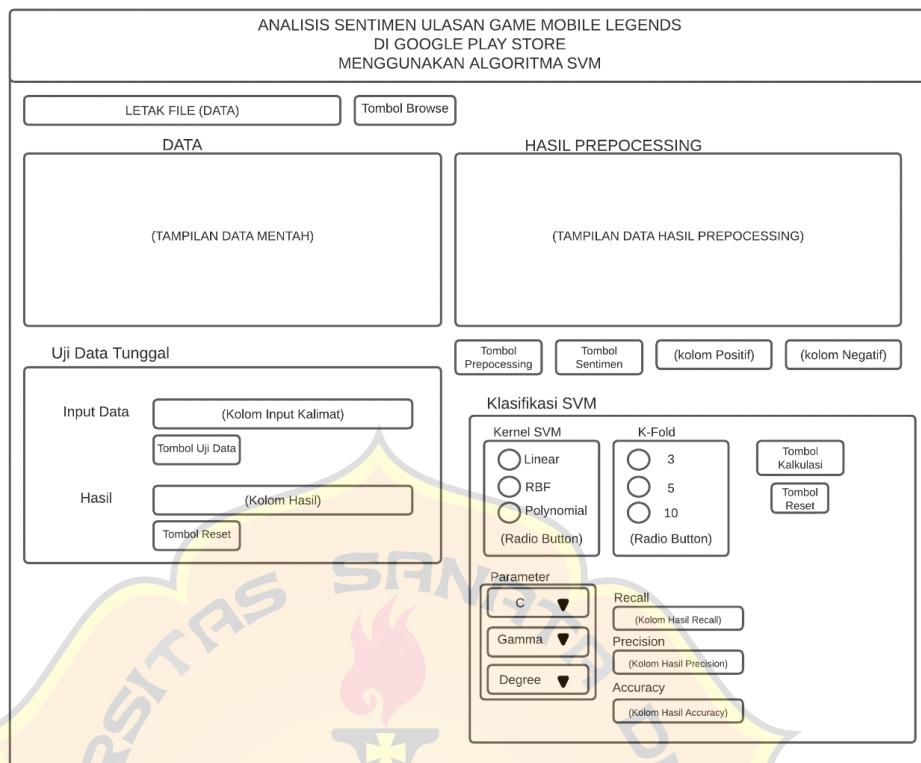
$$\text{Accuracy} = \frac{2+2}{2+2+0+1} \times 100\% = 80\%$$

$$\text{Recall} = \frac{2}{2+0} \times 100\% = 100\%$$

$$\text{Precision} = \frac{2}{2+1} = 66.6\%$$

Nilai dari *accuracy*, *recall*, dan *precision* yang diperoleh dari hasil *Confusion Matrix* menghasilkan *accuracy* 80%, *recall* 100%, dan *precision* 66.6%.

3.7.3 Desain GUI Program



Gambar 3. 4 Desain GUI Program

Gambar 3.3 merupakan desain GUI yang digunakan dalam penelitian ini untuk memproses data dan klasifikasi data menggunakan algoritma *Support Vector Machine (SVM)*. Dalam GUI terdapat tombol *Browse* untuk mencari data mentah yang akan di klasifikasi serta kolom untuk menampilkan letak/path file dalam format (.csv). Pada kolom *Tampilan Data Mentah* akan ditampilkan data mentah yang telah dimasukkan ke program. Tombol *Preprocessing* berfungsi untuk memproses data mentah melalui proses preprocessing sehingga menjadi data yang bersih dan sudah diberi label. Data yang sudah bersih dan telah diberi label akan ditampilkan pada kolom *Tampilan Data Hasil Preprocessing*. Tombol *Sentimen* berfungsi menghitung jumlah data sentimen positif dan sentimen negatif yang hasilnya akan ditampilkan di kolom *Positif* dan kolom *Negatif*.

Kolom *Klasifikasi SVM* terdapat kolom pilihan *K-fold* dan *kernel SVM* dengan bentuk *Radio Button*, untuk memilih kernel yang akan dipakai

beserta jumlah k-fold. Pada bagian kolom pilihan *Parameter* terdapat pilihan untuk menentukan *C*, *gamma* dan *degree*. Tombol *Kalkulasi* berfungsi menghitung *accuracy*, *Precision* dan *recall* dari klasifikasi SVM yang hasilnya akan di tampilkan pada kolom *recall*, *precision* dan *accuracy*. Tombol *Reset* berfungsi untuk membersihkan kolom Klasifikasi SVM agar dapat memasukkan data yang baru.

Kolom *Uji Data Tunggal* terdapat kolom *Input Data* untuk memasukkan data tunggal yang ingin di uji dan tombol *Uji Data* untuk mengklasifikasi data tunggal yang telah dimasukkan. Kolom *Hasil* akan menampilkan hasil klasifikasi data tunggal. Tombol *Reset* berfungsi untuk membersihkan kolom *Input Data* dan kolom *Hasil* agar dapat memasukkan data yang baru.

3.8 Skenario Pengujian

Skenario pengujian bertujuan untuk menguji performa sistem dan ketepatan sistem dalam melakukan klasifikasi. Dalam skenario pengujian data terbagi menjadi 2 tahap, yaitu tahap pengujian sistem untuk mengklasifikasikan data dan pengujian sistem untuk mengklasifikasi data tunggal.

3.8.1 Pengujian Sistem Klasifikasi Support Vector Machine (SVM).

Pengujian sistem dilakukan untuk mengklasifikasikan data dan mencari akurasi yang terbaik dari data menggunakan algoritma *SVM*. dalam Pengujian algoritma *SVM*, *k-fold* yang digunakan bernilai 3, 5, dan 10. Parameter *C (cost)* digunakan untuk ketiga kernel, yaitu *linear*, *rbf*, dan *polynomial*. Untuk kernel *linear*, *rbf* dan *polynomial* menggunakan nilai parameter $C = 0.1, 0.5, 1, 5, 10, 15, 20, 25, 30, 35, 40, 45$. Parameter *gamma* merupakan parameter untuk menentukan pengaruh dari satu sampel dataset pelatihan. Jika nilai *gamma* rendah berarti titik yang berada jauh dari garis pemisah dipertimbangkan dalam perhitungan, sedangkan jika nilai *gamma* tinggi berarti titik-titik yang berada di sekitar garis akan dipertimbangkan dalam perhitungan. Parameter *gamma* digunakan untuk kernel *rbf*, dan *polynomial* dengan nilai $\gamma = (0.1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, \text{ dan } scale)$. *Scale* merupakan nilai default dari *gamma* yang diperoleh dari ($scale = 1 /$

*n_features * X.var()*). Parameter *degree* merupakan parameter yang digunakan dalam kernel *polynomial*. Parameter *degree* berfungsi untuk fleksibilitas dari *decision boundary*. Nilai parameter *degree* yang digunakan, yaitu (0.1, 1, 2, 3, 4, 5, 6, 7).

3.8.2 Uji Data Tunggal

Pengujian data tunggal yang digunakan dalam klasifikasi *SVM* didasarkan pada hasil terbaik dari nilai *k-fold* dan nilai parameter. Pengujian data tunggal dilakukan dengan cara mengambil data secara acak sebanyak 10 *ulasan* yang telah melewati proses *preprocessing*, penerjemahan data, dan pelabelan menggunakan *Vader Lexicon*. Berikut merupakan tabel dari hasil pelabelan menggunakan *Vader Lexicon* sebanyak 10 *ulasan* yang diambil secara acak :

Tabel 3.19 Hasil Pelabelan Vader Lexicon

Ulasan	Sentimen
bagus bug freeze ganggu banget masuk match freeze gk gerak	Positif
depelover kecil mb hp kentang gini susah main masalah bug ya main bug gak enak segi grafik bagus kok	Negatif
perbaiki kota ku sinyal bagus frame frezy lag drop parah gk pengalaman main kurang baik	Negatif
bagus banget game maen bulan menghibur	Positif
game bagus alami peningkat grafis jauh tampilan unik fitur baru tambah	Positif
pihak developer perbaiki ulang habis update malah ngefreeze lag ping drop	Negatif
game bagus sedikit saran aj tim moonton klk kurang data simpan berat	Positif
tambah sulit menang team musuh yg pro team teman yg noob kesan gak adil	Negatif
game keren bagus buka pengaturan battle kok nyata afk ya perbaiki	Positif
game keren main lancar grafik bagus sinyal hilang	Positif

BAB IV

IMPLEMENTASI SISTEM DAN ANALISIS HASIL

4.1 Tahapan Analisis Sentimen

4.1.1 Scraping Data

Dalam penelitian ini, proses *scraping* data menggunakan fungsi *google-play-scrapers*. Data mentah yang diambil merupakan data yang berbahasa Indonesia sebanyak 5000 *ulasan* pada bulan Desember 2021 sampai dengan bulan Juni 2022. Data mentah disimpan dalam format file (.csv).

berikut merupakan *Source Code* dan hasil *scraping* menggunakan fungsi *google-play-scrapers* dengan tools *VS Code*:

```

1 from google_play_scraper import app
2 import pandas as pd
3 import numpy as np
4 from google_play_scraper import Sort, reviews
5 result, continuation_token = reviews(
6     'com.mobile.legend',
7     lang='id',
8     country='id',
9     sort=Sort.MOST_RELEVANT,
10    count=5000,
11    filter_score_with=None )
12 df_busu = pd.DataFrame(np.array(result),columns=['review'])
13 df_busu = df_busu.join(pd.DataFrame(df_busu.pop('review').tolist()))
14 df_busu[['user_name', 'score','at', 'content']].head()
15 new_df = df_busu[['user_name', 'score','at', 'content']]
16 sorted_df = new_df.sort_values(by='at', ascending=False)
17 my_df = sorted_df[['user_name', 'score','at', 'content']]
18 my_df.to_csv("ml_review.csv", index = False)
19

```

Gambar 4.1 *Source Code* untuk *Scraping* Data pada *Google Play Store*

User	Rating	Date	Review
Rezki Raplisal Siregar	1	12/27/2021 18:17	Semenjak update patch baru tiba2 frame drop,padahal sbelumnya tidak ada masalah seperti itu. Spek hp punya 4gb ram dan 64gb rom
cika amelia putri	3	12/28/2021 1:45	Tolong dong kasih tau masuk mode Rank, sya udh level 8 dan udh punya 5 hero, ada tutorial untuk masuk n
Alessandro Kotalawala	1	12/28/2021 5:27	Match makingnya gak bener, random player yang dipilih selalu merugikan team. Sistem solo mohon diperbaik
Dheno gaming center	1	12/28/2021 12:13	Sistem match tidak seimbang . Dan hero terkadang ada yg powernya tidak masuk akal . Bertahun tahun main game ini
Zaky Achsanul	1	12/28/2021 15:18	Untuk update selanjutnya mungkin perlu fitur pengompresan data,hal ini sangat membantu player2 yang r
Vania Stephani	4	12/30/2021 3:04	Game-nya bagus.. Cuma tolong perbaiki sinyal-nya, padahal jaringan + sinyal saya bagus, buka apk yg lain langsung
Nanthul	1	12/30/2021 17:33	Update terbaru game jd tambah berat, RAM 4Gb dan ROM masih sekitar lebih dari 20GB berat bgt buat main game
Near Queen Azkiah	5	12/30/2021 22:08	Game nya seru,,tapi maaf untuk kepada pihak moonton untuk memperbaiki keseimbangan game contoh saja
Muhammad Nauramuhtaror	5	12/31/2021 2:07	Baru baru ini ml update tambah berat dan menambah fps drop saat bermain itu tidak nyaman sekali, Padahal game ini
Ahmad Khalilullah	5	1/6/2022 11:45	Gamenya bagus sekali, tapi ada beberapa hal yang membuat saya kesel, karena pengurangan kredit skor yang
-Ray -	1	1/7/2022 7:22	Tim feeder terus muncul dan jelas disengajakan, terdapat pola yang sama. Terus terjadi, diisi 3 pemain yang
Yunia Chan	2	1/9/2022 1:08	Semenjak update pengalaman saya bermain ml menjadi buruk. Sebelumnya main aman aman saja. Sekarang
Grasyela Melodia putri	3	1/10/2022 4:57	Bagus sih.tqpi Kok tiba tiba aq gk bisa main di mode classic sih "skor anda masih terlalu rendah untuk main

Gambar 4.2 Data Mentah Hasil Scraping

4.1.2 *Preprocessing* Data

Dalam penelitian ini, *preprocessing* hanya menggunakan kolom *review(ulasan)*. Kolom-kolom yang tidak digunakan seperti user, rating, dan date akan dihapus. Tahap *preprocessing* meliputi *Cleaning*, *Case-Folding*, *Tokenize*, *Stemming*, dan *Stopword Removal*.

4.1.2.1 Clean Ulasan

Dalam tahap *cleaning ulasan*, terdapat tahap menghapus angka, simbol, remove punct(hapus tanda baca), simbol, *drop duplicates* (menghapus data yang memiliki indikasi duplikat), dan *emoticon*.

```

24     class PreprocessingLexicon:
25         def remove_pattern(input_txt, pattern):
26             r = re.findall(pattern, input_txt)
27             for i in r:
28                 input_txt = re.sub(i, '', input_txt)
29             return input_txt
30     df['remove_user'] = np.vectorize(pl.remove_pattern)(df['case-folding'], "@[\w]*")
31
32     def removePunct(text):
33         text = " ".join([char for char in text if char not in string.punctuation])
34         return text
35
36     def remove(review):
37
38         review = re.sub('[0-9]+', '', review)
39         review = re.sub(r'\$\w*', '', review)
40
41
42
43
44     def getEmoticons(self):
45         emoticons_happy = set([
46             ':)', ':)', ';)', ':o)', ':]', ':b)', ':c)', ':>', '=]', ':8)', '=:)', ';}',
47             ':-^)', ':-D', ':D', '8-D', '8D', 'x-D', 'xD', 'X-D', 'xD+', '=-D', '=D',
48             '=-3', '=3', ':-))', ";'-)", ";'-)", ':*+', ':**+', '>:P', '>-P', ':P', 'X-P',
49             'x-p', 'xp', 'xp', ':-p', ':p', '=p', ':b', ':b', '>:)', '>:-)', '<3'
50             ])
51
52
53         emoticons_sad = set([
54             ':(', ':/(', '>:/(', ':(s', '>:[(, ':@', ':-(', ':[', ':|-|', '=L', ':<',
55             ':-[', ':<', '=\\', '=/', '>(:', '>(:', '><', ">:-(", ">:(", '>:\\', '>-c',
56             ':c', ':{', '>:\\', ':('
57             ])
58
59         self.emoticons = emoticons_happy.union(emoticons_sad)
60     return self.emoticons

```

Gambar 4. 3 Source Code Cleaning ulasan

Source code pada gambar 4.3 digunakan untuk proses *cleaning ulasan*. Dalam proses *cleaning ulasan* terdapat 6 tahap, yaitu *case-folding*, *remove number*, *remove symbol*, *remove punct*, *remove emoticon*, dan *drop duplicates*.

Tahap *Case-folding* adalah tahap untuk mengganti huruf kapital menjadi huruf kecil(*lowercase*). Tahap *remove number*, *remove symbol*, dan *remove punct* adalah tahap untuk menghapus angka dari 0-9, menghapus simbol, dan menghapus tanda baca. Tahap *remove emoticon* untuk menghapus emoji. Tahap *drop duplicates* adalah tahap untuk menghapus duplikasi data yang telah diolah pada tahap sebelumnya.

Pada proses *cleaning*, Terdapat 4 modul yang digunakan, yaitu *re*, *lower*, *join*, dan *replace*. *re*(*regular expression*) adalah modul yang berfungsi untuk mencocokan *string* serta mensubtitusi pada sebuah *string*. modul *lower* berfungsi untuk merubah semua huruf kapital menjadi huruf kecil(*lowercase*). Modul *join* berfungsi

untuk menghapus ruang kosong di antara kata dalam sebuah kalimat. Modul *replace* memiliki fungsi yang sama seperti modul *re*. berikut merupakan output hasil *cleaning review* :

Tabel 4. 1 Hasil *Cleaning*

Data mentah	Hasil <i>cleaning</i>
Saya rasa game ini sudah benar-benar rusak	saya rasa game ini sudah benarbenar rusak
ini Game bagus si tapi ada 1 masalah yaitu bugnya 😞 😞	ini game bagus si tapi ada masalah yaitu bugnya
Sejauh ini permainan nya Bagus	sejauh ini permainan nya bagus
Makin lama size Ml makin berat bahkan device 3/32 juga udah gak sanggup lagi"	makin lama size ml makin berat bahkan device juga udah gak sanggup lagi
Ini kenapa nggak bisa masuk ML saya moonton?	Ini kenapa nggak bisa masuk ml saya moonton

4.1.2.2 Tokenize, Stemming, dan Stopword Removal

Setelah melewati proses cleaning, data akan masuk kedalam proses *tokenize*, *stemming*, dan *stopword removal*. Proses *tokenize* merupakan tahap untuk membagi setiap kata dalam review kedalam bentuk *token*. Proses *stemming* berfungsi untuk menghapus imbuhan pada setiap kata dalam *token*. proses *Stopword removal* berfungsi untuk menghapus kata-kata yang tidak memiliki makna.

```
76     tokenizer = Tokenizer(preserve_case=False, strip_handles=True, reduce_len=True)
77     reviews_tokens = tokenizer.tokenize(review)
78
79     review_clean = []
80     for word in reviews_tokens:
81         if (word not in self.stopwords_indonesia and
82             word not in self.emoticons and
83             word not in string.punctuation):
84             stem_word = self.stemmer.stem(word)
85             review_clean.append(stem_word)
86
87             print(review_clean)
88     return review_clean
89
90     def getStemming(self):
91         factory = StemmerFactory()
92         self.stemmer = factory.create_stemmer()
93         return self.stemmer
94
95     def getStopwords(self):
96         self.stopwords_indonesia = stopwords.words('indonesian')
97         dataRemove = ['tidak', 'siap', 'baik', 'kurang', 'jangan']
98         self.stopwords_indonesia = set(stopwords.words('indonesian')).difference(dataRemove)
99         return self.stopwords_indonesia
```

Gambar 4.4 source Code *tokenize*, *stemming*, dan *stopword removal*

Pada gambar 4.4, tahap *tokenize* menggunakan *library NLTK* yaitu *nltk.tokenize*. *Library NLTK.tokenize* adalah *library* yang digunakan untuk untuk membagi setiap kata dalam *ulasan* kedalam bentuk *token* (membagi kata dalam kalimat). Tahap *stopwords removal* menggunakan *library NLTK* yaitu *nltk.corpus*. *Library NLTK.corpus (stopwords)* digunakan untuk menganalisa teks dan menghapus teks yang tidak memiliki arti atau tudak relevan. Tahap berikutnya adalah tahap *Stemming* dengan menggunakan *StemmerFactory* dari *library Sastrawi.Stemmer*. *Stemmer Factory*. Berikut merupakan hasil dari proses *tokenize*, *stemming*, dan *stopword removal*:

Tabel 4.2 hasil *tokenize*, *stemming*, dan *stopword removal*

Cleaning	Tokenize, Stemming dan Stopword
saya rasa game ini sudah benarbenar rusak	[‘game’, ‘benarbenar’, ‘rusak’]
ini game bagus si tapi ada masalah yaitu bugnya	[‘game’, ‘bagus’, ‘si’, ‘masalah’, ‘bug’]
sejauh ini permainan nya bagus	[‘permainan’, ‘bagus’]
makin lama size ml makin berat bahkan device juga udah gak sanggup lagi	[‘size’, ‘ml’, ‘berat’, ‘device’, ‘udah’, ‘gak’, ‘sanggup’]
Ini kenapa nggak bisa masuk ml saya moonton	[‘nggak’, ‘masuk’, ‘ml’, ‘moonton’]

4.1.3 Menerjemahkan Data

Setelah melewati proses *preprocessing*, data selanjutnya diterjemahkan dalam bahasa Inggris. Data diterjemahkan menggunakan penerjemah dokumen berbasis web, yaitu onlinedoctranslator.com.

Berikut hasil sebelum dan sesudah data diterjemahkan:

	Review
0	semenjak update patch frame droppadah sbelumnya tidak spek hp mumpuni kadang frame drop dan mode refre
1	tolong kasih tau masuk mode rank sya udh level udh hero tutorial masuk mode rank sya ikut main tidak baca tap n
2	match making gak bener random player pilih rugi team sistem solo mohon baik main ketemu yg main saran sistem
3	sistem match tidak imbang hero terkadang yg powernya tidak masuk akal tahun main tidak ubah sistem match tidi
4	update fitur kompres datahal bantu player milik device pasanmemang grafik bagus versi nyaman alam main optim
5	game-nya bagus
6	update baru game jd berat ram gb rom gb berat bgt main pas war grafik jd patah settingan low versi gk kayak gin
7	game nya serutapi maaf moonton baik imbang game contoh kali main temu musuh rank nya banding contoh rank
8	ml update berat tambah fps drop main tidak nyaman hp ram simpan kosong centang refresh rate tutup atur buka :
9	gamennya bagus kesel kurang kredit skor tidak masuk akal draft loading gambar hero lawan banget trs relog diitung
10	tim feeder muncul sengaja pola isi main feeder maju tidak kontrol main sistem matchmaking game kualitas rendal

Gambar 4.5 Data sebelum diterjemahkan

	Review
0	since the update patch the frame drops even though previously the cellphone specs weren't qualified sometimes the frame dr
1	please tell me I'm in rank mode I'm already level tutorial hero I'm in rank mode I'm playing don't read tap on rank mode it's ok
2	match making is not correct random player chooses team loss solo system please play well meet the one playing suggest mat
3	the match system doesn't match the hero sometimes whose power doesn't make sense the year of playing doesn't change th
4	update the data compression feature the help of the player belonging to the paired device indeed good graphics comfortable
5	the game is good.
6	new update game so heavy ram gb rom gb too heavy to play when war graphics so broken low version setting it's not like gin
7	the game is similar but sorry moonton is good the game is balanced the example of playing meet the enemy the rank is compa
8	ml heavy update added fps drop play uncomfortable hp ram save empty check refresh rate close set uncheck no help moonto
9	the game is good annoyed lack of credit the score doesn't make sense the draft loading of the opponent's hero is very good tl
10	the feeder team appears intentionally the content pattern of the main feeder advances does not control the play low quality

Gambar 4.6 Data setelah diterjemahkan onlinedoctranslator.com

4.1.4 Pelabelan *Vader Lexicon*

Setelah data diterjemahkan ke dalam bahasa Inggris, selanjutnya masuk ke tahap pelabelan menggunakan *Vader Lexicon*. Pelabelan *Vader Lexicon* menggunakan Library NLTK.sentiment.vader dengan modul SentimentIntensityAnalyzer.

```
df = pd.read_excel('TranslateDataReview.xlsx').astype(str)
def Labeling(data):
    analyzer = SentimentIntensityAnalyzer()
    score = analyzer.polarity_scores(data)
    return score

df['scores'] = df['Review'].apply(lambda x: pl.Labeling(x))
df['compound'] = df['scores'].apply(lambda score_dict: score_dict['compound'])
df['label'] = df['compound'].apply(lambda c: 1 if c > 0 else 0 if c==0 else -1)

indexNames = df[(df['label'] == 0)].index
a = df.drop(indexNames, inplace=True)

hasilLexicon = df[['Review','label']]
hasilLexicon['Review'].replace('', np.nan, inplace=True)
hasilLexicon.dropna(subset=['Review'], inplace =True)

positif = hasilLexicon[hasilLexicon['label']==1]['label'].count()
negatif = hasilLexicon[hasilLexicon['label']==-1]['label'].count()

dataPos = hasilLexicon[hasilLexicon['label']==1]
dataNeg = hasilLexicon[hasilLexicon['label']==-1]
```

Gambar 4.7 Source Code labeling menggunakan *Vader Lexicon*

Dalam proses pelabelan, data diberi nilai negatif, *compound*, dan positif. Nilai *compound* digunakan untuk mendefinisikan atau memberi nilai pada *ulasan* bersentimen positif atau negatif. Jika *Ulasan* yang memiliki nilai compound > 0 , maka sentimen bernilai positif, dan *ulasan* dengan nilai compound < 0 , maka sentimen bernilai negatif. Penelitian ini hanya menggunakan sentimen positif dan sentimen negatif, sehingga apabila nilai compound = 0 (neutra), maka akan dihapus.

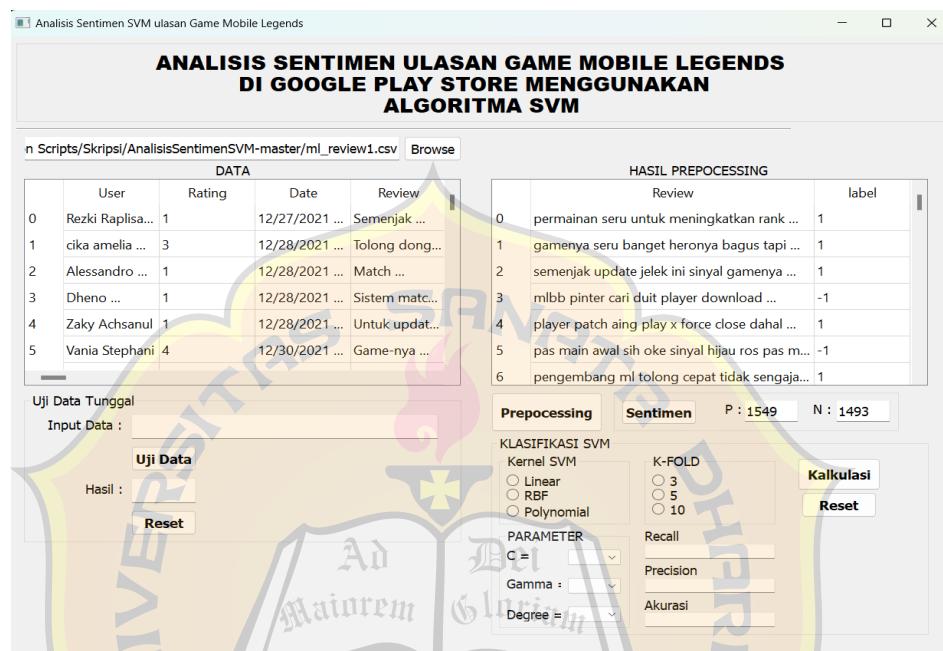
Berikut merupakan contoh hasil pelabelan menggunakan *Vader Lexicon* :

	Review	label
0	permainan seru untuk meningkatkan rank klasik musuh jangan bilang tim jangan main moonton mohon berbaik hati	1
1	gamenya seru banget heronya bagus tapi besar aku main wifi yang bagus banget tapi sinyalnya merah aku capek tem	1
2	semenjak update jelek ini sinyal gamenya merah padahal pake wifi pake kartu telkom kartu smartfren menurut ane g	1
3	mllb pinter cari duit player download gamenya lumayan gede buat naikin level hati performa gamenya rusak banget	-1
4	player patch aing play x force close dahal speed wifi data lancar hp gada mohon berbaik hati mengganggu format ma	1
5	pas main awal sih oke sinyal hijau ros pas mid game sinyal drop merah gak gerak trus pit pas udah pit sistem notifikasi	-1
6	pengembang ml tolong cepat tidak sengaja lalu masukkan kembali tolong cepat agar tidak masuk afk terima kasih ba	1
7	season kali fitur game bagus fitur kurang bagus matchmaking lag crash koneksi bagus akses aplikasi mathcmaking sila	1
8	percuma game nya beri tim pro musuh noob memang moonton kalo buat game gak pake otak tolong di imbangin mu	-1
9	Dari segi grafis bagus sinyalnya tidak sestabil main-main suka merah sinyalnya cepat mulus	1
10	njriitt bulan jangan masuk sekarang login savage yang bekerja keras untuk kehilangan legendaris yang mendapat ratu	-1

Gambar 4.8 Hasil pelabelan menggunakan *vader lexicon*

Dari hasil pelabelan *Vader Lexicon*, jumlah *ulasan* menjadi 3042. Jumlah sentimen bernilai positif sebanyak 1549 dan sentimen bernilai negatif sebanyak 1493.

Berikut ini merupakan tampilan GUI yang menampilkan hasil *preprocessing* yang telah diberi label menggunakan *Vader Lexicon*, serta jumlah sentimen positif dan negatif :



Gambar 4. 9 Tampilan GUI program

4.2 Klasifikasi Support Vector Machine (SVM)

Setelah melewati proses pelabelan menggunakan *Vader Lexicon*, data akan masuk ke tahap *vectorize TF-IDF* dan normalisasi (*Cosine normalization*) sebelum masuk ke tahap klasifikasi SVM. Tahap *vectorize TF-IDF* menggunakan library *sklearn.features_extraction.text* dengan modul *TfidfVectorizer*. Proses *TfidfVectorizer* menggunakan parameter *norm = 'l1'*, yaitu menggunakan *cosine normalization*.

```
vectorizer = TfidfVectorizer(norm = 'l1')
```

Gambar 4.10 Source code *vectorizer*

(0, 938)	0.08811562766367698
(0, 1772)	0.04431912469292012
(0, 1347)	0.12891797048595124
(0, 903)	0.09949956235865004
(0, 375)	0.15877845380613026
(0, 1666)	0.12246796932106552
(0, 1208)	0.10686527084474687
(0, 4245)	0.06684671512803067
(0, 1988)	0.11144161950046333
(0, 4660)	0.03374759290620391
(0, 3078)	0.03900009329216106
(1, 4152)	0.062119563789254414
(1, 3125)	0.06413509270966421
(1, 4111)	0.046454177889562034
(1, 2279)	0.08510738069504012
(1, 4623)	0.028910591582674335
(1, 4934)	0.04963127690583839
(1, 3124)	0.03362372612978369
(1, 2708)	0.05492744631354412
(1, 779)	0.04997462119954969
(1, 1888)	0.11294497414880461
(1, 2048)	0.045909496141784245
(1, 4453)	0.03901718831270247
(1, 265)	0.041799017599056396
(1, 1748)	0.03492142232379725

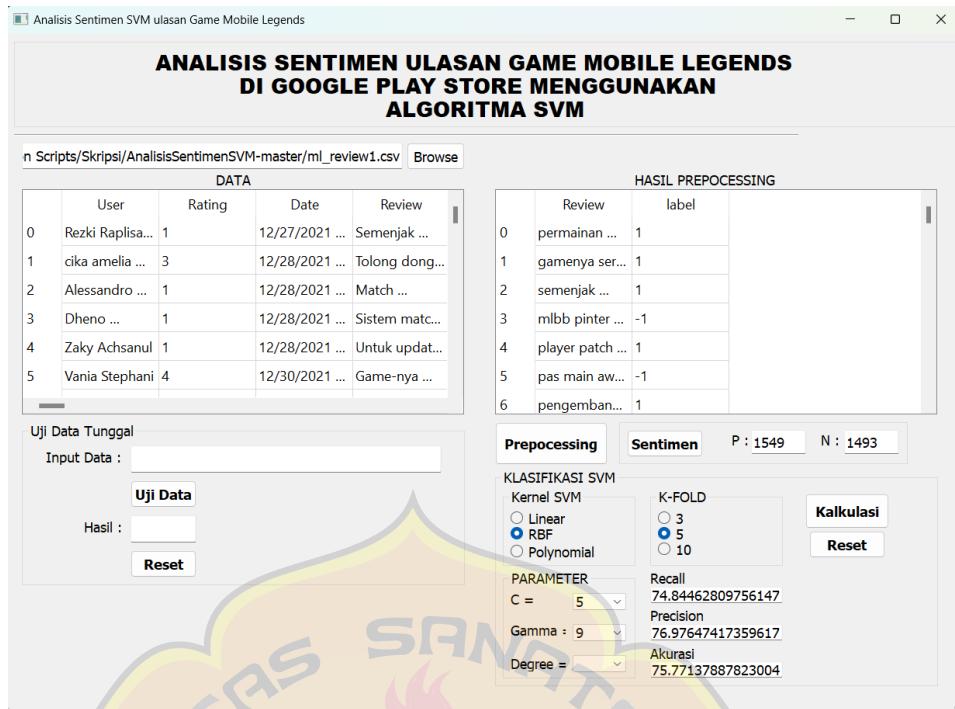
Gambar 4.11 Hasil *Vectorize* dengan kernel *linear* dan data *training K = 5*

Setelah data melewati proses *vectorize TF-IDF* dan normalisasi (*Cosine normalization*), data akan masuk ke proses klasifikasi *Support Vector Machine (SVM)*. klasifikasi *Support Vector Machine (SVM)* menggunakan fungsi *Support Vector Classification (SVC)* dari *library sklearn.svm* yang dimuat dalam *clf*.

```
clf=svm.SVC(kernel='rbf', C=c, gamma = g)
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
akurasi.append(metrics.accuracy_score(y_test, y_pred))
recall.append(metrics.recall_score(y_test, y_pred))
precision.append(metrics.precision_score(y_test, y_pred))
```

Gambar 4.12 Source code klasifikasi *Support Vector Machine (SVM)*

Gambar 4.12 merupakan source code klasifikasi dengan *clf* yang menggunakan parameter kernel *linear* dan nilai $C = c$ dan $\gamma = g$. *clf.fit* berfungsi untuk menyesuaikan dimensi matriks dari variabel *clf*. *clf.predict* berfungsi untuk mendapatkan hasil prediksi klasifikasi. Fungsi *metrics accuracy_score*, *recall_score* dan *precision_score* diperoleh dari *library sklearn*. Berikut adalah tampilan GUI saat melakukan klasifikasi *SVM* menggunakan kernel *rbf*, dan *K-Fold = 5*, $C = 5$ $\gamma = 9$.



Gambar 4.13 Tampilan GUI kernel *RBF*, $C = 5$, $\gamma = 9$

4.2.1 Uji Parameter Terbaik

Pengujian parameter *k-fold* terbaik menggunakan nilai *k-fold* 3, 5, 10. Parameter yang digunakan untuk masing-masing kernel, yaitu C , γ , dan $Degree$.

4.2.1.1 Pengujian Kernel dengan Parameter C

Pengujian dengan parameter C dilakukan pada kernel *linear*, *rbf*, dan *polynomial*. Berikut merupakan tabel pengujian dari ke 3 kernel:

- Kernel *Linear*
 - Berikut merupakan tabel hasil pengujian kernel *linear* dengan *k-fold* = 3 dan parameter C :

Tabel 4.3 Pengujian kernel *linear*, *k-fold* = 3

C	Akurasi Linear	Precision	Recall
0,1	49,50%	66.66%	67.47%
0,5	51,28%	66.33%	68.84%
1	64,20%	72.46%	68.98%
5	75,73%	78.48%	72.61%
10	75,77%	78.10%	73.05%
15	75,60%	77.47%	73.53%
20	75,27%	76.89%	73.65%
25	75,64%	77.04%	74.36%
30	75,54%	76.56%	74.94%
35	75,21%	75.96%	75.13%
40	74,95%	75.66%	74.95%
45	74,72%	75.52%	74.57%

- Berikut merupakan tabel hasil pengujian kernel *linear* dengan *k-fold* = 5 dan parameter *C* :

Tabel 4.4 Pengujian kernel *linear*, *k-fold* = 5

C	Akurasi Linear	Precision	Recall
0,1	50,92%	50.92%	90.43%
0,5	53,91%	58.06%	87.77%
1	71,13%	71.70%	78.31%
5	76,69%	79.30%	73.52%
10	76,98%	79.40%	74.07%
15	76,85%	78.46%	75.25%
20	76,62%	77.72%	75.89%
25	76,59%	77.85%	75.57%
30	76,42%	77.50%	75.76%
35	76,26%	77.37%	75.50%
40	75,73%	76.76%	75.09%
45	75,47%	76.37%	75.10%

- Berikut merupakan tabel hasil pengujian kernel *linear* dengan $k\text{-fold} = 10$ dan parameter C :

Tabel 4. 5 Pengujian kernel *linear*, $k\text{-fold} = 10$

C	Akurasi Linear	Precision	Recall
0,1	50.92%	50.92%	100.0%
0,5	55.65%	54.51%	97.35%
1	73,40%	74.11%	75.94%
5	77.48%	79.85%	74.72%
10	77.51%	79.80%	74.82%
15	77.38%	78.88%	75.96%
20	77.38%	78.22%	77.10%
25	77.31%	77.89%	77.48%
30	77.31%	77.72%	77.81%
35	76.69%	77.13%	77.22%
40	76,56%	77.01%	77.00%
45	76,13%	76.37%	77.01%

Berdasarkan pengujian kernel *linear* menggunakan nilai $k\text{-fold}$ 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai $k\text{-fold} = 10$ dan parameter $C = 10$ dengan akurasi = 77,51%, *precision* = 79.80%, dan *recall* = 74.82%.

- Kernel *RBF*
 - Berikut merupakan tabel hasil pengujian kernel *rbf* dengan $k\text{-fold} = 3$ dan parameter C :

Tabel 4.6 Pengujian Kernel *rbf*, $k\text{-fold} = 3$

C	Akurasi RBF	Precision	Recall
0,1	55.16%	58.96%	82.86%
0,5	73.24%	76.66%	69.70%
1	75.47%	77.91%	72.60%
5	74.72%	76.26%	73.29%
10	74.68%	76.24%	73.22%
15	74.68%	76.24%	73.22%
20	74.68%	76.24%	73.22%
25	74.68%	76.24%	73.22%
30	74.68%	76.24%	73.22%
35	74.68%	76.24%	73.22%
40	74.68%	76.24%	73.22%

- b. Berikut merupakan tabel hasil pengujian kernel *rbf* dengan *k-fold* = 5 dan parameter *C* :

Tabel 4.7 Pengujian kernel *rbf*, *k-fold* = 5

C	Akurasi RBF	Precision	Recall
0,1	53.55%	52.60%	99.08%
0,5	75.67%	79.27%	71.09%
1	76.42%	78.97%	73.31%
5	75.90%	77.50%	74.29%
10	75.93%	77.55%	74.29%
15	75.93%	77.55%	74.29%
20	75.93%	77.55%	74.29%
25	75.93%	77.55%	74.29%
30	75.93%	77.55%	74.29%
35	75.93%	77.55%	74.29%
40	75.93%	77.55%	74.29%

- c. Berikut merupakan tabel hasil pengujian kernel *rbf* dengan *k-fold* = 10 dan parameter *C* :

Tabel 4.8 Pengujian kernel *rbf*, *k-fold* = 10

C	Akurasi RBF	Precision	Recall
0,1	52.69%	51.98%	99.45%
0,5	76.72%	80.16%	72.39%
1	76.95%	78.17%	76.00%
5	77.02%	79.04%	74.84%
10	76.92%	78.15%	75.94%
15	76.92%	78.15%	75.94%
20	76.92%	78.15%	75.94%
25	76.92%	78.15%	75.94%
30	76.92%	78.15%	75.94%
35	76.92%	78.15%	75.94%
40	76.92%	78.15%	75.94%

Berdasarkan pengujian kernel *rbf* menggunakan nilai *k-fold* 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai *k-fold* = 10 dan parameter *C* = 5 dengan akurasi = 77.05%, *precision* = 79.04%, dan *recall* = 74.84%.

- Kernel *Polynomial*

- Berikut merupakan tabel hasil pengujian kernel *polynomial* dengan $k\text{-fold} = 3$ dan parameter C :

Tabel 4.9 Pengujian kernel *polynomial*, $k\text{-fold} = 3$

C	Akurasi Polynomial	Precision	Recall
0,1	51.74%	51.37%	99.68%
0,5	56.14%	54.23%	95.86%
1	63.41%	60.62%	87.82%
5	68.14%	65.59%	81.71%
10	69.03%	66.61%	80.86%
15	69.13%	67.30%	78.88%
20	69.32%	67.88%	77.62%
25	69.46%	68.31%	76.59%
30	69.23%	68.31%	75.90%
35	69.23%	68.31%	75.90%
40	69.23%	68.31%	75.90%

- Berikut merupakan tabel hasil pengujian kernel *polynomial* dengan $k\text{-fold} = 5$ dan parameter C :

Tabel 4.10 Pengujian kernel *polynomial*, $k\text{-fold} = 5$

C	Akurasi Polynomial	Precision	Recall
0,1	51.01%	50.97%	100.0%
0,5	55.49%	53.77%	96.40%
1	64.23%	61.26%	88.28%
5	69.26%	66.29%	82.01%
10	69.82%	67.24%	80.60%
15	70.21%	68.51%	78.18%
20	70.28%	68.97%	76.87%
25	70.38%	69.32%	76.17%
30	70.57%	69.56%	76.17%
35	70.57%	69.56%	76.17%
40	70.57%	69.56%	76.17%

- c. Berikut merupakan tabel hasil pengujian kernel *polynomial* dengan $k\text{-fold} = 10$ dan parameter C :

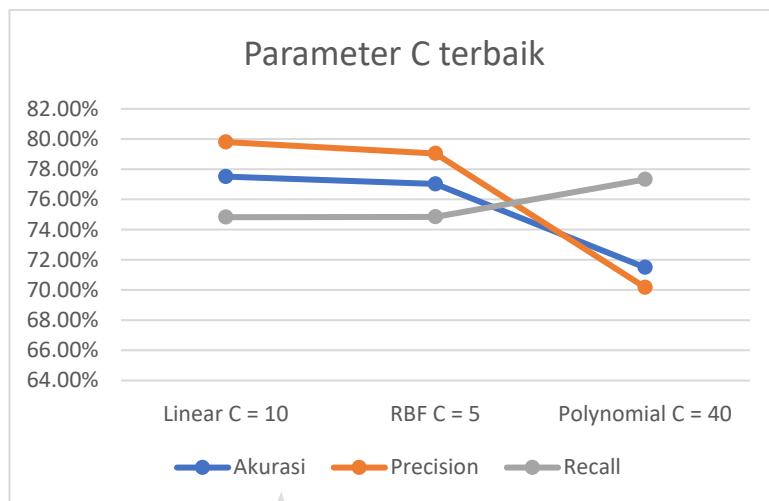
Tabel 4.11 Pengujian kernel *polynomial*, $k\text{-fold} = 10$

C	Akurasi Polynomial	Precision	Recall
0,1	51.02%	50.97%	100.0%
0,5	55.92%	53.97%	97.64%
1	63.18%	60.34%	88.42%
5	69.13%	66.26%	81.76%
10	69.82%	67.09%	81.04%
15	70.44%	68.46%	78.76%
20	71.00%	69.33%	77.97%
25	71.36%	69.93%	77.51%
30	71.49%	70.17%	77.32%
35	71.49%	70.17%	77.32%
40	71.49%	70.17%	77.32%

berdasarkan pengujian kernel *polynomial* menggunakan nilai $k\text{-fold}$ 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai $k\text{-fold} = 10$ dan parameter $C = 40$ dengan akurasi = 71.49%, *precision* = 70.17%, dan *recall* = 77.32%. Berikut merupakan tabel perbandingan hasil akurasi dari tiap kernel :

Tabel 4.12 Pengujian Kernel dengan Parameter C Terbaik

Kernel	C	Akurasi	Precision	Recall
Linear	10	77.51%	79.80%	74.82%
RBF	5	77.02%	79.04%	74.84%
Polynomial	40	71.49%	70.17%	77.32%



Gambar 4.14 Grafik Parameter C Terbaik

Pengujian parameter C terbaik terdapat pada kernel *linear* dengan nilai $k\text{-fold} = 10$ menggunakan parameter $C = 10$ mendapatkan akurasi sebesar 77,51%, *precision* sebesar 79,80% dan *recall* sebesar 74,82%.

4.2.1.2 Pengujian Kernel dengan parameter *gamma*

Pengujian parameter *gamma* dilakukan pada kernel *rbf* dan *polynomial*. Nilai parameter C yang digunakan pada kernel *rbf* adalah $C = 5$. Nilai tersebut merupakan hasil terbaik dari pengujian parameter C menggunakan kernel *rbf*.

Nilai parameter C yang digunakan pada kernel *polynomial* adalah $C = 40$. Nilai tersebut merupakan hasil terbaik dari pengujian parameter C menggunakan kernel *polynomial*. Berikut merupakan tabel pengujian dari ke 2 kernel:

- Kernel *RBF*
 - a. Berikut merupakan tabel hasil pengujian kernel *rbf* menggunakan nilai $k\text{-fold} = 3$ dengan parameter $C = 5$ dan parameter *gamma* :

Tabel 4.13 Pengujian kernel rbf , $k\text{-}fold} = 3$

C	Gamma	Akurasi RBF	Precision	Recall
5	0.1	64.26%	72.40%	69.49%
5	1	76.06%	78.48%	73.24%
5	2	75.77%	77.33%	74.24%
5	3	75.54%	76.91%	74.30%
5	4	75.44%	76.51%	74.74%
5	5	74.98%	75.91%	74.54%
5	6	74.91%	75.74%	74.66%
5	7	75.24%	76.06%	74.98%
5	8	75.14%	76.02%	74.80%
5	9	75.21%	76.09%	74.84%
5	10	75.21%	76.09%	74.84%
5	Scale	74.72%	76.26%	73.29%

- b. Berikut merupakan tabel hasil pengujian kernel rbf menggunakan nilai $k\text{-}fold} = 5$ dengan parameter $C = 5$ dan parameter $gamma$:

Tabel 4.14 Pengujian kernel rbf , $k\text{-}fold} = 5$

C	Gamma	Akurasi RBF	Precision	Recall
5	0.1	70.93%	71.41%	78.76%
5	1	77.08%	79.64%	73.93%
5	2	76.88%	78.32%	75.57%
5	3	76.79%	78.09%	75.69%
5	4	76.23%	77.52%	75.21%
5	5	75.90%	77.06%	75.08%
5	6	75.70%	76.88%	74.89%
5	7	75.80%	76.89%	75.13%
5	8	75.83%	77.02%	74.99%
5	9	75.77%	76.97%	74.84%
5	10	75.87%	77.07%	74.90%
5	Scale	75.90%	77.50%	74.29%

- c. Berikut merupakan tabel hasil pengujian kernel *rbf* menggunakan nilai *k-fold* = 10 dengan parameter C = 5 dan parameter *gamma* :

Tabel 4.15 Pengujian kernel *rbf*, *k-fold* = 10

C	Gamma	Akurasi RBF	Precision	Recall
5	0.1	73.10%	73.56 %	76.45%
5	1	77.54%	79.81 %	74.99%
5	2	77.51%	78.73 %	76.64%
5	3	77.05%	78.02 %	76.62%
5	4	76.72%	77.71 %	76.21%
5	5	76.56%	77.41%	76.26%
5	6	76.72%	77.49%	76.59%
5	7	76.46%	77.51%	75.80%
5	8	76.46%	77.65%	75.53%
5	9	76.59%	77.64%	75.93%
5	10	76.33%	77.30%	75.78%
5	Scale	76.95%	78.17%	76.00%

berdasarkan pengujian kernel *rbf* menggunakan nilai *k-fold* 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai *k-fold* = 10 dan parameter C = 5 dan parameter *gamma* = 1 dengan akurasi = 77.54%, *precision* = 79.83 %, *recall* = 75.06 %.

- Kernel *Polynomial*

- Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 3 dengan parameter C = 40 dan parameter *gamma*:

Tabel 4.16 Pengujian kernel *polynomial*, *k-fold* = 3

C	Gamma	Akurasi Polynomial	Precision	Recall
40	0.1	49.11%	66.66%	66.72%
40	1	49.11%	66.66%	66.72%
40	2	49.11%	66.66%	66.72%
40	3	49.11%	66.66%	66.72%
40	4	53.58%	52.65%	97.63%
40	5	52.03%	51.55%	99.37%
40	6	54.63%	53.38%	97.19%
40	7	55.62%	53.87%	96.68%
40	8	58.02%	55.72%	92.80%
40	9	62.13%	59.15%	89.58%
40	10	65.58%	63.20%	85.13%
40	Scale	68.14%	65.59%	81.71%

- Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 5 dengan parameter C = 40 dan parameter *gamma*:

Tabel 4.17 Pengujian kernel *polynomial*, *k-fold* = 5

C	Gamma	Akurasi Polynomial	Precision	Recall
40	0.1	50.92%	50.92%	100.0%
40	1	50.92%	50.92%	100.0%
40	2	50.92%	50.92%	100.0%
40	3	50.92%	50.92%	100.0%
40	4	51.01%	50.97%	100.0%
40	5	51.15%	51.04%	99.93%
40	6	52.30%	51.71%	99.51%
40	7	54.76%	53.26%	97.62%
40	8	58.77%	56.14%	94.14%
40	9	63.51%	60.25%	89.78%
40	10	66.63%	63.99%	85.27%
40	Scale	69.26%	66.29%	82.01%

- c. Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 10 dengan parameter C = 40 dan parameter *gamma*:

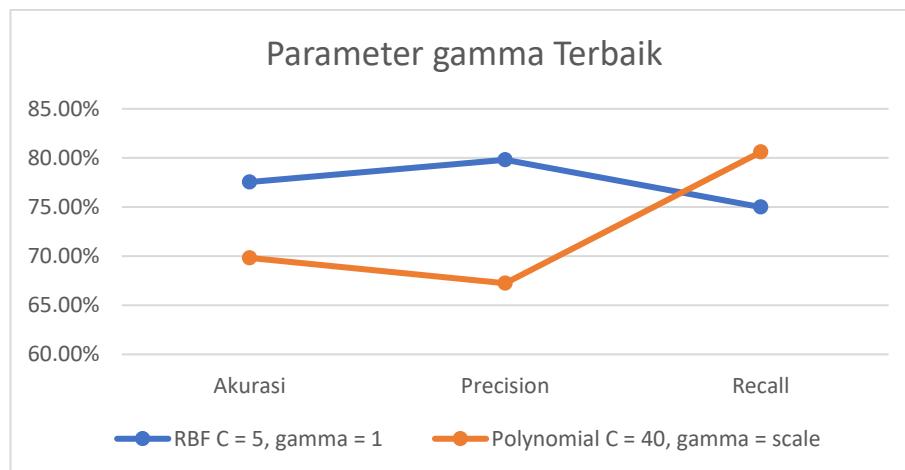
Tabel 4.18 Pengujian kernel *polynomial*, *k-fold* = 10

C	Gamma	Akurasi Polynomail	Precision	Recall
40	0.1	50.92%	50.92%	100.0%
40	1	50.92%	50.92%	100.0%
40	2	50.92%	50.92%	100.0%
40	3	50.95%	50.95%	100.0%
40	4	51.15%	51.04%	99.93%
40	5	53.09%	52.18%	99.14%
40	6	56.77%	54.64%	95.78%
40	7	62.72%	59.35%	91.22%
40	8	66.79%	64.13%	85.07%
40	9	67.78%	65.03%	83.80%
40	10	68.30%	65.28%	83.17%
40	Scale	69.82%	67.24%	80.60%

Berdasarkan pengujian kernel *polynomial* menggunakan nilai *k-fold* 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai *k-fold* = 10 dan parameter C = 40 dan *gamma* = ‘scale’ dengan akurasi = 69.82%, *precision* = 67.24%, *recall* = 80.60%. Berikut merupakan tabel perbandingan hasil akurasi dari tiap kernel :

Tabel 4.19 Pengujian Kernel dengan parameter *gamma* terbaik

Kernel	C	gamma	Akurasi	Precision	Recall
RBF	5	1	77.54%	79.81 %	74.99%
Polynomial	40	Scale	69.82%	67.24%	80.60%



Gambar 4.15 Grafik Parameter *Gamma* Terbaik

Pengujian parameter *gamma* terbaik terdapat pada kernel *rbf* dengan nilai *k-fold* = 10 menggunakan parameter *C* = 5 dan parameter *gamma* = 1 mendapatkan akurasi sebesar 77,54%, *precision* sebesar 79,81% dan *recall* sebesar 74,99%.

4.2.1.3 Pengujian Parameter *degree*

Pengujian parameter *degree* dilakukan pada kernel *polynomial*. Nilai parameter yang digunakan pada kernel *polynomial* adalah *C* = 40 dan nilai *gamma* = ‘*scale*’. Nilai tersebut merupakan hasil terbaik dari pengujian parameter *C* dan parameter *gamma* menggunakan kernel *polynomial*. Berikut merupakan tabel pengujian dari kernel *polynomial* :

- Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 3 dengan parameter *C* = 40, parameter *gamma* = ‘*scale*’ dan parameter *degree* :

Tabel 4.20 Pengujian kernel *polynomial*, *k-fold* = 3

C	gamma	degree	Akurasi	Precision	Recall
40	Scale	0.1	50.92%	50.92%	100.0%
40	Scale	1	72.15%	72.92%	72.07%
40	Scale	2	73.83%	73.82%	75.36%
40	Scale	3	69.00%	66.59%	80.80%
40	Scale	4	58.97%	56.33%	91.04%
40	Scale	5	52.43%	51.79%	97.01%
40	Scale	6	51.61%	51.30%	99.06%
40	Scale	7	51.15%	51.04%	99.49%

- Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 5 dengan parameter C = 40, parameter *gamma* = ‘scale’ dan parameter *degree* :

Tabel 4.21 Pengujian kernel *polynomial*, *k-fold* = 5

C	Gamma	degree	Akurasi	Precision	Recall
40	Scale	0.1	50.92%	50.92%	100.0%
40	Scale	1	73.30%	73.56%	74.29%
40	Scale	2	74.72%	74.84%	76.01%
40	Scale	3	69.82%	67.24%	80.60%
40	Scale	4	59.14%	56.36%	90.93%
40	Scale	5	52.59%	51.87%	96.94%
40	Scale	6	51.38%	51.16%	99.21%
40	Scale	7	51.21%	51.07%	99.87%

- Berikut merupakan tabel hasil pengujian kernel *polynomial* menggunakan nilai *k-fold* = 10 dengan parameter C = 40, parameter *gamma* = ‘scale’ dan parameter *degree* :

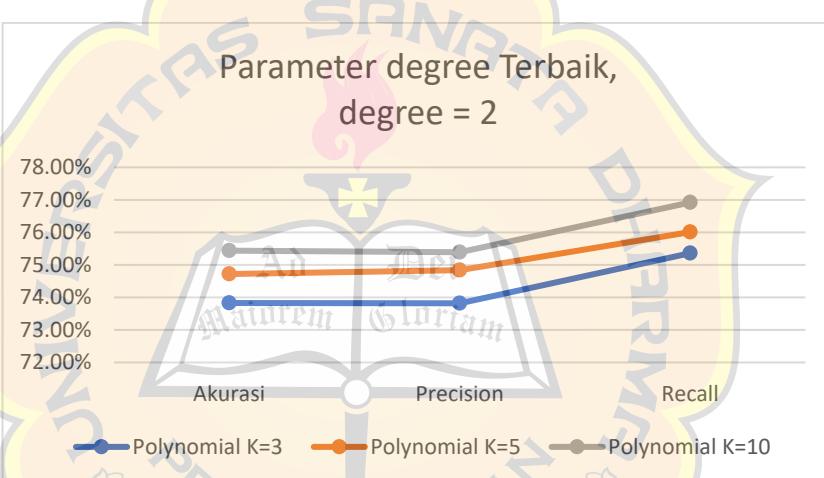
Tabel 4.22 Pengujian kernel *polynomial*, *k-fold* = 10

C	Gamma	degree	Akurasi	Precision	Recall
40	Scale	0.1	50.92%	50.92%	100.0%
40	Scale	1	72.64%	72.46%	74.44%
40	Scale	2	75.44%	75.39%	76.92%
40	Scale	3	71.43%	70.13%	77.19%
40	Scale	4	64.33%	62.19%	82.44%
40	Scale	5	53.48%	52.39%	95.75%
40	Scale	6	51.97%	51.52%	98.14%
40	Scale	7	51.15%	51.04%	99.42%

berdasarkan pengujian parameter *degree* dengan kernel *polynomial* menggunakan nilai *k-fold* 3, 5, dan 10, di dapatkan hasil akurasi tertinggi pada nilai *k-fold* = 10 dan *degree* = 2 dengan akurasi = 75.44%, *precision* = 75.39%, *recall* = 76.92%. Berikut merupakan tabel perbandingan hasil akurasi dari tiap kernel :

Tabel 4.23 Pengujian Kernel dengan Parameter *degree* Terbaik

Kernel	K-fold	C	gamma	degree	Akurasi	Precision	Recall
Polynomial	3	40	Scale	2	73.83%	73.82%	75.36%
Polynomial	5	40	Scale	2	74.72%	74.84%	76.01%
Polynomial	10	40	Scale	2	75.44%	75.39%	76.92%



Gambar 4.16 Grafik Parameter *degree* Terbaik

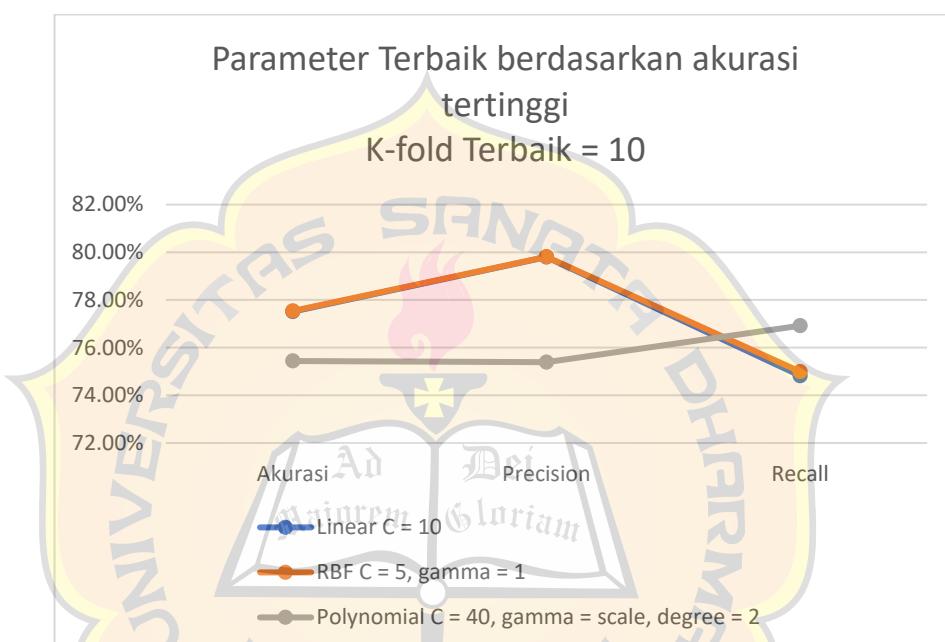
Pengujian parameter *degree* terbaik pada kernel *polynomial* dengan nilai *k-fold* = 10 menggunakan parameter *C* = 40, parameter *gamma* = *scale*, dan *degree* = 2 mendapatkan akurasi sebesar 75.44%, *precision* sebesar 75.39% dan *recall* sebesar 76.92%.

4.2.2 Perbandingan Hasil Uji Akurasi Kernel

Penentuan Kernel dengan parameter terbaik dapat dilihat dari akurasi tertinggi masing-masing kernel :

Tabel 4.24 Kernel dengan Parameter Terbaik berdasarkan akurasi tertinggi

Kernel	K-fold	C	gamma	degree	Akurasi	Precision	Recall
Linear	10	10	-	-	77.51%	79.80%	74.82%
RBF	10	5	1	-	77.54%	79.81 %	74.99%
Polynomial	10	40	scale	2	75.44%	75.39%	76.92%



Gambar 4.17 Grafik Parameter Terbaik berdasarkan akurasi tertinggi

Pengujian kernel dengan parameter terbaik berdasarkan akurasi tertinggi terdapat pada kernel *rbf* dengan nilai *k-fold* = 10 menggunakan parameter *C* = 5 dan parameter *gamma* = 1 mendapatkan akurasi sebesar 77.54%, *precision* sebesar 79.81% dan *recall* sebesar 74.99%.

Model evaluasi terbaik diperoleh dari kernel *rbf* dengan akurasi 77.54%, *precision* = 79.81 %, dan *recall* = 74.99%.

Tabel 4.25 Model Evaluasi Terbaik

Model Evaluasi Terbaik						
Kernel	RBF	C	gamma	Akurasi	Precision	Recall
	5	5	1	77.54%	79.81%	74.99%

4.3 Pengujian data Tunggal

Pengujian data tunggal menggunakan Kernel dengan model terbaik, yaitu kernel = *rbf* dengan *k-fold* = 10, *C* = 5, dan *gamma* = 1. Pengujian data tunggal dilakukan pada *ulasan* yang diambil secara acak sebanyak 10 *ulasan* dari *Google Play Store*. 10 *ulasan* tersebut akan melewati proses *preprocessing*, kemudian *ulasan* akan dilakukan pengujian satu per satu.

Tabel 4. 26 Hasil Pengujian Data Tunggal

No.	Ulasan	Sentimen Vader	Sentimen klasifikasi SVM
1	bagus bug freeze ganggu banget masuk match freeze gk gerak	Positif	Positif
2	depelover kecil mb hp kentang gini susah main masalah bug ya main bug gak enak segi grafik bagus kok	Negatif	Postif
3	perbaiki kota ku sinal bagus frame frezy lag drop parah gk pengalaman main kurang baik	Negatif	Negatif
4	bagus banget game maen bulan menghibur	Positif	Positif
5	game bagus alami peningkat grafis jauh tampilan unik fitur baru tambah	Positif	Positif
6	pihak developer perbaiki ulang habis update malah ngefreeze lag ping drop	Negatif	Negatif
7	game bagus sedikit saran aj tim moonton klk kurang data simpan berat	Positif	Positif
8	tambah sulit menang team musuh yg pro team teman yg noob kesan gak adil	Negatif	Negatif
9	game keren bagus buka pengaturan battle kok nyata afk ya perbaiki	Positif	Positif
10	game keren main lancar grafik bagus sinal hilang	Positif	Positif

Berdasarkan tabel 4.26 hasil klasifikasi pengujian data tunggal, dapat dihasilkan akurasi, *precision* dan *recall* dengan *confusion matrix* sebagai berikut :

Tabel 4.27 *Confusion matrix* Pengujian Data Tunggal

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	6	1
	Negatif	0	3

$$\text{Accuracy} = \frac{6+3}{6+1+0+3} \times 100\% = 90\%$$

$$\text{Precision} = \frac{6}{6+1} \times 100\% = 85.71\%$$

$$\text{Recall} = \frac{6}{6+0} \times 100\% = 100\%$$

Berdasarkan Hasil *confusion matrix* pengujian data tunggal pada tabel 4.27, didapat akurasi = 90%, *precision* = 85.71%, dan *recall* = 100%.

4.4 Analisis Hasil

4.4.1 Klasifikasi Support Vector Machine (SVM)

Dalam klasifikasi *Support Vector Machine*, sistem akan melalui tahap pelabelan menggunakan *Vader Lexicon*, setelah itu sistem akan masuk ke proses klasifikasi. Pada tahap klasifikasi digunakan *k-fold cross validation* dengan nilai *k-fold* = 3, 5 dan 10 untuk melakukan pembagian (*split*) data menjadi data *testing* dan data *training*.

Tahap selanjutnya masuk tahap pembobotan *tf-idf* untuk memberi bobot pada data/mengubah data menjadi data *vector*.

Setelah melewati tahap pembobotan *tf-idf*, tahap selanjutnya adalah klasifikasi *SVM* menggunakan nilai *k-fold* 3, 5 dan 10 serta menggunakan parameter *C*, *gamma*, dan *degree* pada kernel *linear*, *rbf*, *polynomial* berdasarkan inputan ke sistem. Tahap selanjutnya, yaitu *confusion matrix*. *Confusion matrix* digunakan untuk mengevaluasi hasil dari klasifikasi *SVM*. Berikut merupakan tabel hasil klasifikasi *SVM* berdasarkan kernel terbaik dari parameter .

Tabel 4.28 Hasil Klasifikasi Kernel dalam SVM

Kernel	K-fold	C	gamma	degree	Akurasi	Precision	Recall
Linear	10	10	-	-	77.51%	79.80%	74.82%
RBF	10	5	1	-	77.54%	79.81 %	74.99%
Polynomial	10	40	scale	2	75.44%	75.39%	76.92%

4.4.2 Confusion matrix

Berikut adalah hasil *confusion matrix* dari klasifikasi *Support Vector Machine (SVM)* menggunakan kernel = *rbf*, *k-fold* = 10, C = 5, dan gamma = 1 :

- Percobaan ke 1 *Confusion matrix*

Tabel 4.29 Percobaan ke 1 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	121	33
	Negatif	32	119

- Percobaan ke 2 *Confusion matrix*

Tabel 4.30 Percobaan ke 2 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	125	37
	Negatif	28	115

- Percobaan ke 3 *Confusion matrix*

Tabel 4.31 Percobaan ke 3 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	133	29
	Negatif	36	106

- Percobaan ke 4 *Confusion matrix*

Tabel 4.32 Percobaan ke 4 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	111	22
	Negatif	42	129

- Percobaan ke 5 *Confusion matrix*

Tabel 4.33 Percobaan ke 5 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	122	30
	Negatif	36	116

- Percobaan ke 6 *Confusion matrix*

Tabel 4.34 Percobaan ke 6 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	132	29
	Negatif	31	112

- Percobaan ke 7 *Confusion matrix*

Tabel 4.35 Percobaan ke 7 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	107	25
	Negatif	53	119

- Percobaan ke 8 *Confusion matrix*

Tabel 4.36 Percobaan ke 8 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	119	28
	Negatif	34	123

- Percobaan ke 9 *Confusion matrix*

Tabel 4.37 Percobaan ke 9 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	115	34
	Negatif	44	111

- Percobaan ke 10 *Confusion matrix*

Tabel 4.38 Percobaan ke 10 *Confusion matrix*

		<i>Actual Value</i>	
		Positif	Negatif
<i>Predicted Value</i>	Positif	115	26
	Negatif	54	109

Berdasarkan *confusion matrix*, klasifikasi *SVM* dengan kernel *rbf*, nilai *k-fold* = 10, *C* = 5, *gamma* = 1, mendapat hasil rata-rata akurasi = 77.54%, *precision* = 79.81% dan *recall* = 74.99%.

BAB 5

PENUTUP

5.1 Kesimpulan

Kesimpulan dari hasil analisis dan penelitian analisis sentimen ulasan game *Mobile Legends* di *Google Play Store* menggunakan algoritma *SVM*, dapat disimpulkan bahwa algoritma *Support Vector Machine (SVM)* dapat diimplementasikan dalam analisis sentimen ulasan game *Mobile Legends* di *Google Play Store*.

Berdasarkan hasil pengujian untuk mencari akurasi terbaik terdapat pada kernel *rbf* menggunakan nilai *k-fold* = 10, parameter $C = 5$, dan $\gamma = 1$ diperoleh rata-rata hasil akurasi sebesar 77.54%, *precision* sebesar 79.81%, dan *recall* sebesar 74.99%.

Pengujian data tunggal menggunakan 10 data uji yang di ambil secara acak pada ulasan di *Google Play Store* menggunakan hasil pengujian terbaik kernel *rbf* dengan nilai parameter $C = 5$ dan $\gamma = 1$ mendapatkan akurasi sebesar 90.00%, *precision* sebesar 85.71% dan *recall* sebesar 100%, sehingga dapat di simpulkan klasifikasi *Support Vector Machine (SVM)* dapat melakukan uji data tunggal dengan baik.

5.2 Saran

Berikut saran yang dapat diberikan berdasarkan penelitian ini :

1. Menambahkan Normalisasi kata pada proses *preprocessing* agar tidak terlalu banyak variasi kata yang tidak baku.
2. Menambahkan label sentimen netral untuk mengetahui *Ulasan* yang bersifat netral.
3. Menambahkan kernel *Sigmod* dalam klasifikasi *SVM*
4. Menggunakan algoritma/metode lain dalam analisis sentimen
5. Menggunakan dataset lain dalam analisis sentimen

DAFTAR PUSTAKA

- A. M. Pravina, I. Cholissodin, and P. P. Adikara. (2019). “Analisis Sentimen Tentang Opini Maskapai Penerbangan pada Dokumen Twitter Menggunakan Algoritme Support Vector Machine (SVM)”, J. Pengemb. Teknol. Inf. dan Ilmu Komput., vol. 3, no. 3, pp. 2789–2797, [Online]. Available: <http://j-ptiik.ub.ac.id>.
- A. Weichselbraun, S. Gindl and A. Scharl, “A Context-Dependent Supervised Learning Approach to Sentiment Detection in Large Textual Database,” Journal of Information and Data Management, vol. 1, no. 3, pp. 329-341, 2010.
- Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M. M., & Williams, H. E., 2007. Stemming Indonesian: A confix-stripping approach, ACM Transactions on AsianLanguage Information Processing (TALIP), vol. 6, no. 4, 1-33.
- Agusta, Ledy. (2009). Perbandingan Algoritma Stemming Porter dengan Algoritma Nazief & Adriani untuk Stemming Dokumen Teks Bahasa Indonesia, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana, 15(11-19). 1411-3201.
- B. Liu. (2012). “Sentiment Analysis and Opinion Mining,” Synth. Lect. Hum. Lang. Technol., vol. 5, no. 1, pp,1-167.
- Christianini, Nello & Shawe-Taylor, John. (2000). An Introduction to Support Vector Machines and Other Kernel Based Learning Methods.
- Fadholi Fat Haranto, B. W. (2019). “Implementasi Support Vector Machine Untuk Analisis Sentimen Pengguna Twitter Terhadap Pelayanan Telkom Dan Biznet.” 15, 171-175.
- Fanani, F. (2017). Klasifikasi Review Software pada Google Play Store Menggunakan Pendekatan Analisis Sentimen. Skripsi: Ilmu Komunikasi,Universitas Gadjah Mada.

- Han, J, Kamber, M, & Pei, J. (2012). *Data Mining: Concept and Techniques*, Third Edition. Waltham: Morgan Kaufmann Publishers.
- Herlinawati, Nuraeni & Yuliani, Yuri & Faizah, Siti & Gata, Windu & Samudi, Samudi. (2020). “Analisis Sentimen Zoom Cloud Meetings di Play Store Menggunakan Naïve Bayes dan Support Vector Machine.” CESS (Journal of Computer Engineering, System and Science). 5. 293. 10.24114/cess.v5i2.18186.
- Haryanto, D., Muflikhah, L., & Fauzi, M. “Analisis Sentimen Review Barang Berbahasa Indonesia Dengan Metode Support Vector Machine Dan Query Expansion.” Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, (2-9), p. 2909-2916.
- Indrayuni, E. (2018). “Komparasi Algoritma Naive Bayes Dan Support Vector Machine Untuk Analisa Sentimen Review Film.” PILAR Nusa Mandiri , 14(2): 175-180.
- Informatikalogi. (2016). Pembobotan Kata atau Term Weighting TF-IDF. Diambil dari Informatikalogi.com: <https://informatikalogi.com/term-weighting-tf-idf/>
- Irfani, Faizal. (2020). “Analisis Sentimen Review Aplikasi Ruangguru Menggunakan Algoritma Support Vector Machine.” Jbmi (Jurnal Bisnis, Manajemen, Dan Informatika). 16. 258. 10.26487
- Kohavi, R. and Provost, F. (1998). *Glossary of terms. Machine Learning—Special Issue on Applications of Machine Learning and the Knowledge Discovery Process*. Machine Learning, 30, 271-274.
- Kurniawan, T. (2017). “Implementasi Text Mining Pada Analisis Sentimen Pengguna Twitter Terhadap Media Mainstream Menggunakan Naive Bayes Classifier dan Support Vector Machine.” Tugas Akhir.
- M. Iwan Januar, E. F Turmudzi. (2006) Game Mania, Gema Insani, Jakarta.
- Moraes, Rodrigo & Valiati, Joao & Gavião Neto, Wilson. (2013). Document-Level Sentiment Classification: An Empirical Comparison Between SVM And

- ANN. Expert Systems With Applications. 40. (621–633). 10.1016/J.Eswa.2012.07.059.
- N. Fitriyah, B. Warsito, And D. A. I. Maruddani. (2020). “Analisis Sentimen Gojek Pada Media Sosial Twitter Dengan Klasifikasi Support Vector Machine (SVM),” Jurnal Gaussian, Vol. 9, No. 3, Pp. 376-390. <https://doi.Org/10.14710/J.Gauss.V9i3.28932>
- Pang, Bo. Lee, L dan Vaithyanathan, S. (2002). *Thumbs up? Sentiment classification using machine learning techniques*. Proceedings of the 7th Conference on Empirical Methods in Natural Language Processing (EMNLP-02).
- Rofiqoh, U., Perdana, R. S. & Fauzi, M. A., (2017). “Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dngan Metode Support Vector Machine dan Lexicon Based Features.” Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, I(12), pp. 1725- 1732.
- Rahutomo, S. F. (2018). Implementasi Twitter Sentiment Analysis Untuk Review Film Menggunakan Algoritma Support Vector Machine. *Informatika Polinema*, 93-100.
- Ramadhan Wp, A. N. (2017). “Analisis Sentimen Menggunakan Support Vector Machine Dan Maximum Entropy.” E-Proceeding Of Engineering, 4, 2389-2395.
- Saifinnuha, AmridioZulhilm (2015). “Penerapan Sentimen Analisis Pada Twitter Berbahasa Indonesia Untuk Mendapatkan Rating Program Televisi Menggunakan Metode Support Vector Machine.” Sarjana thesis, Universitas Brawijaya. Thesis
- Samsudiney. (2019, Juli 25). “Penjelasan Sederhana tentang Apa Itu SVM?” Diambil dari medium.com: <https://medium.com/@samsudiney/penjelasan-sederhana-tentang-apa-itu-svm-149fec72bd02>

- Santosa, Budi (2007). Data Mining Teknik Pemanfaatan Data untuk Keperluan Bisnis. Yogyakarta : Graha Ilmu.
- Saputra, S. A., Rosiyadi, D., Gata, W., & Husain, S. M. (2019). “Analisis Sentimen E-Wallet Pada Google Play Menggunakan Algoritma Naive Bayes Berbasis Particle Swarm Optimization.” *Resti*, 3(3), 377–382.
- Syahfitri Kartika Lidya, O. S. (2015). “Sentiment Analysis Pada Teks Bahasa Indonesia Menggunakan Support Vector Machine (Svm) Dan K-Nearest Neighbor (K-Nn).” 1-8.
- Tempola, F. M. (2018). “Perbandingan Klasifikasi Antara KNN Dan Naive Bayes Pada Penentuan Status Gunung Berapi Dengan K-Fold Cross Validation.” *Teknologi Informasi dan Ilmu Komputer (JTIK)*, 5(5): 577-584.

- Williams, G. (2011). Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery. Diunduh dari <http://users.umiacs.umd.edu/~oard/teaching/301/spring16/readings/Willia ms.pdf>

LAMPIRAN

1. Source Code Scraping Dataset

```

1  from google_play_scraper import app
2  import pandas as pd
3  import numpy as np
4  from google_play_scraper import Sort, reviews
5  result, continuation_token = reviews(
6      'com.mobile.legend',
7      lang='id',
8      country='id',
9      sort=Sort.MOST_RELEVANT,
10     count=10,
11     filter_score_with=None )
12 df_busu = pd.DataFrame(np.array(result),columns=['review'])
13 df_busu = df_busu.join(pd.DataFrame(df_busu.pop('review').tolist()))
14 df_busu[['user_name', 'score', 'at', 'content']].head()
15 new_df = df_busu[['user_name', 'score', 'at', 'content']]
16 sorted_df = new_df.sort_values(by='at', ascending=False)
17 my_df = sorted_df[['user_name', 'score', 'at', 'content']]
18 my_df.to_csv("Uji data tunggal.csv", index = False)

```

2. Source Code Preprocessing

```

4  from VaderPreprocessing import PreprocessingLexicon as pl
5  import pandas as pd
6  import numpy as np
7
8  df = pd.read_csv('ml_review1.csv')
9  df = df
10 df = df['Review']
11 df = df.to_frame()
12
13 df['remove_user'] = np.vectorize(pl.remove_pattern)(df, "@[\w]*")
14 # df['remove_user']
15
16 df.drop_duplicates(subset = "remove_user", keep = 'first', inplace = True)
17 # remove angka
18 df['remove_http'] = df['remove_user'].apply(lambda x: pl.remove(x))
19 df.sort_values("remove_http", inplace = True)
20 df.drop_duplicates(subset = "remove_http", keep = 'first', inplace = True)
21
22 emoticons = pl.getEmoticons(df)
23 stopwords_indonesia = pl.getStopwords(df)
24 stemmer = pl.getStemming(df)
25
26 #cleaning
27 df['review_clean'] = df['remove_http'].apply(lambda x: pl.cleanReview(x,df))
28
29     # punct
30     # remove_punct = pl.removePunct(self.df)
31 df['Review'] = df['review_clean'].apply(lambda x: pl.removePunct(x))
32 df.drop_duplicates(subset = "Review", keep = 'first', inplace = True)
33 df.sort_values("Review", inplace = True)
34 df = pd.DataFrame(df['Review'])
35 # mengurutkan index
36 df = df.sort_index()
37 #df.to_csv('CleanDataReview.csv',encoding='utf8', index=False)
38 df.to_excel('test10data.xlsx')

```

3. Source Code Vader Lexicon

```
8 import pandas as pd
9 import numpy as np
10 from nltk.sentiment.vader import SentimentIntensityAnalyzer
11
12
13 df = pd.read_excel('Cleandata_testTranslate.xlsx').astype(str)
14 def Labeling(data):
15     analyzer = SentimentIntensityAnalyzer()
16     score = analyzer.polarity_scores(data)
17     return score
18
19 df['scores'] = df['Review'].apply(lambda x: pl.Labeling(x))
20 df['compound'] = df['scores'].apply(lambda score_dict: score_dict['compound'])
21 df['label'] = df['compound'].apply(lambda c: 1 if c >0 else 0 if c==0 else -1)
22
23 indexNames = df[(df['label'] == 0)].index
24 a = df.drop(indexNames, inplace=True)
25
26 hasilLexicon = df[['Review','label']]
27 hasilLexicon['Review'].replace('', np.nan, inplace=True)
28 hasilLexicon.dropna(subset=['Review'], inplace =True)
29
30 positif = hasilLexicon[hasillexicon['label']==1]['label'].count()
31 negatif = hasilLexicon[hasillexicon['label']==-1]['label'].count()
32
33 dataPos = hasilLexicon[hasillexicon['label']==1]
34 dataNeg = hasilLexicon[hasillexicon['label']==-1]
35
36 dataPos = dataPos[:1550]
37 dataGabung = [dataPos, dataNeg]
38 dataset = pd.concat(dataGabung, ignore_index=True)
39 dataset = dataset.sample(frac=1).reset_index(drop=True)
40 dataset.to_excel('VaderDatatest.xlsx')
```

4. Source Code Klasifikasi SVM

```

8   from sklearn.model_selection import KFold
9   from sklearn.feature_extraction.text import TfidfVectorizer
10  from sklearn import svm
11  from sklearn import metrics
12  from sklearn.metrics import confusion_matrix, classification_report
13  import pandas as pd
14  import numpy as np
15
16  class KlasifikasiSVM:
17      def LinearSVM(k, X, y, c):
18          skf = KFold(n_splits=k)
19          akurasi = []
20          recall = []
21          precision=[]
22          for train_index, test_index in skf.split(X):
23              X_train, X_test, y_train, y_test = X[train_index], X[test_index], y[train_index], y[test_index]
24
25              vectorizer = TfidfVectorizer(norm = 'l1')
26
27              X_train=vectorizer.fit_transform(X_train)
28              X_test=vectorizer.transform(X_test)
29
30              clf=svm.SVC(kernel='linear', C=c)
31              clf.fit(X_train,y_train)
32              y_pred = clf.predict(X_test)
33              akurasi.append(metrics.accuracy_score(y_test, y_pred))
34              recall.append(metrics.recall_score(y_test, y_pred))
35              precision.append(metrics.precision_score(y_test, y_pred))
36
37              akurasiTotal = np.mean(akurasi)
38              recallTotal = np.mean(recall)
39              precisionTotal = np.mean(precision)
40              sv = clf.support_vectors_
41              return akurasiTotal, recallTotal, precisionTotal, sv
42
43  def RbfSVM(k, X, y, c, g):
44      skf = KFold(n_splits=k)
45      akurasi = []
46      recall = []
47      precision=[]
48      for train_index, test_index in skf.split(X):
49          X_train, X_test, y_train, y_test = X[train_index], X[test_index], y[train_index], y[test_index]
50
51          vectorizer = TfidfVectorizer(norm = 'l1')
52
53          X_train=vectorizer.fit_transform(X_train)
54          X_test=vectorizer.transform(X_test)
55
56          clf=svm.SVC(kernel='rbf', C=c, gamma = g)
57          clf.fit(X_train,y_train)
58          y_pred = clf.predict(X_test)
59          akurasi.append(metrics.accuracy_score(y_test, y_pred))
60          recall.append(metrics.recall_score(y_test, y_pred))
61          precision.append(metrics.precision_score(y_test, y_pred))
62
63          akurasiTotal = np.mean(akurasi)
64          recallTotal = np.mean(recall)
65          precisionTotal = np.mean(precision)
66          return akurasiTotal, recallTotal, precisionTotal
67
68  def PolySVM(k, X, y,c, g, d):
69      skf = KFold(n_splits=k)
70      akurasi = []
71      recall = []
72      precision=[]
73      for train_index, test_index in skf.split(X):
74          X_train, X_test, y_train, y_test = X[train_index], X[test_index], y[train_index], y[test_index]
75
76          vectorizer = TfidfVectorizer(norm = 'l1')
77
78          X_train=vectorizer.fit_transform(X_train)
79          X_test=vectorizer.transform(X_test)
80          clf=svm.SVC(kernel='poly', C= c, gamma = g, degree = d)
81          clf.fit(X_train,y_train)
82
83          y_pred = clf.predict(X_test)
84          akurasi.append(metrics.accuracy_score(y_test, y_pred))
85          recall.append(metrics.recall_score(y_test, y_pred))
86          precision.append(metrics.precision_score(y_test, y_pred))
87
88          akurasiTotal = np.mean(akurasi)
89          recallTotal = np.mean(recall)
90          precisionTotal = np.mean(precision)
91          return akurasiTotal, recallTotal, precisionTotal

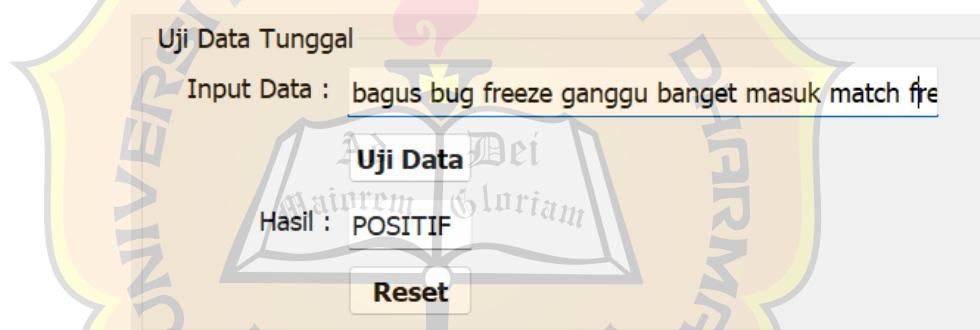
```

5. Klasifikasi SVM Uji Data Tunggal

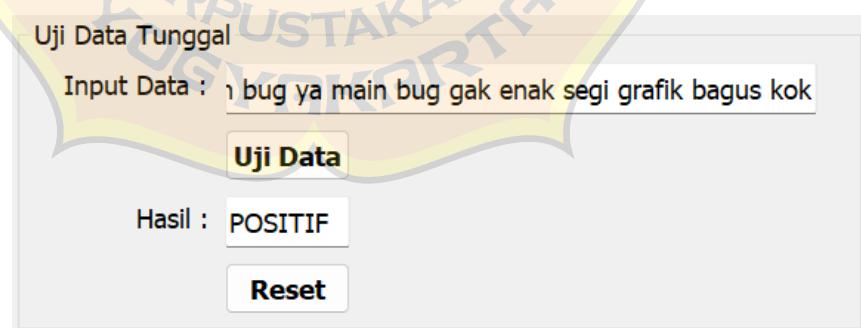
```
95     def c1SVMTunggal(X_test):
96         skf = KFold(n_splits=10)
97         X_train = X
98         y_train = y
99
100        vectorizer = TfidfVectorizer(norm = 'l1')
101
102        X_train=vectorizer.fit_transform(X_train)
103        X_test=vectorizer.transform(X_test)
104        clf=svm.SVC(kernel='rbf', C=5, gamma = 1)
105        clf.fit(X_train,y_train)
106
107
108        y_pred = clf.predict(X_test)
109        return y_pred
```

6. Hasil Uji Data Tunggal GUI

- Uji Data Tunggal 1



- Uji Data Tunggal 2



- Uji Data Tunggal 3

Uji Data Tunggal

Input Data : perbaiki kota ku sinyal bagus frame frezy lag drop

Uji Data

Hasil : NEGATIF

Reset

- Uji Data Tuggal 4

Uji Data Tunggal

Input Data : bagus banget game maen bulan menghibur

Uji Data

Hasil : POSITIF

Reset

- Uji Data Tuggal 5

Uji Data Tunggal

Input Data : ngkat grafis jauh tampilan unik fitur baru tambah

Uji Data

Hasil : POSITIF

Reset

- Uji Data Tuggal 6

Uji Data Tunggal

Input Data : pihak developer perbaiki ulang habis update malah

Uji Data

Hasil : NEGATIF

Reset

- Uji Data Tuggal 7

Uji Data Tunggal

Input Data : game bagus sedikit saran aj tim moonton klk kur

Uji Data

Hasil : POSITIF

Reset

• Uji Data Tuggal 8

Uji Data Tunggal

Input Data : tambah sulit menang team musuh yg pro team ter

Uji Data

Hasil : NEGATIF

Reset

• Uji Data Tuggal 9

Uji Data Tunggal

Input Data : game keren bagus buka pengaturan battle kok nya

Uji Data

Hasil : POSITIF

Reset

• Uji Data Tuggal 10

Uji Data Tunggal

Input Data : game keren main lancar grafik bagus sinyal hilang

Uji Data

Hasil : POSITIF

Reset

7. Confusion Matrix Model Evaluasi Terbaik

Kernel : *RBF*

K-Fold : 10

C : 5

Gamma : 1

```
Confusion Matrix
([array([[121,  33],
       [ 32, 119]], dtype=int64), array([[125,  37],
       [ 28, 115]], dtype=int64), array([[133,  29],
       [ 36, 106]], dtype=int64), array([[111,  22],
       [ 42, 129]], dtype=int64), array([[122,  30],
       [ 36, 116]], dtype=int64), array([[132,  29],
       [ 31, 112]], dtype=int64), array([[107,  25],
       [ 53, 119]], dtype=int64), array([[119,  28],
       [ 34, 123]], dtype=int64), array([[115,  34],
       [ 44, 111]], dtype=int64), array([[115,  26],
       [ 54, 109]]], [0.7868852459016393,
 0.7868852459016393, 0.7861842105263158, 0.78947368421052
63, 0.7828947368421053, 0.8026315789473685, 0.7434210526
31579, 0.7960526315789473, 0.743421052631579, 0.73684210
52631579], 0.7754691544434857)
```