BAB II
KAJIAN LITERATUR

2.1 Analisis Sentimen

Analisis sentimen adalah bidang ilmu yang mempelajari cara untuk menganalisis opini, pendapat, dan penilaian suatu pihak tentang objek tertentu yang dapat berupa produk, layanan, atau suatu masalah. Analisis sentimen dapat diimplementasikan pada berbagai tingkat yaitu teks yang berupa dokumen atau kalimat [14]. Analisis sentimen adalah teknik *Natural Language Processing* (NLP) dan *Machine Learning* (ML) untuk mengevaluasi secara otomatis teks berdasarkan perasaan yang dirasakan penulis. Ada tiga jenis utama analisis sentimen, yaitu [15]:

1. *Multimodal sentiment analysis*

   Multimodal sentiment analysis adalah jenis analisis sentimen yang mempertimbangkan data seperti teks, video, dan audio untuk menganalisis emosi yang diekspresikan dalam data tersebut. Tanda visual dan pendengaran seperti ekspresi wajah dan nada suara dipertimbangkan yang memberikan spektrum sentimen yang luas.
2. *Aspect-based sentiment analysis*

   Dalam analisis berbasis aspek, metode NLP digunakan untuk menganalisis dan mengekstraksi emosi dan pendapat yang terkait dengan aspek atau fitur tertentu dari produk dan layanan. Misalnya, peneliti dapat mengekstraksi sentimen terkait makanan, layanan, suasana, dan lainnya dalam ulasan restoran.
3. *Multilingual sentiment analysis*

   Setiap bahasa memiliki tata bahasa, sintaksis, dan kosa kata yang berbeda, dan sentimen diekspresikan secara berbeda di setiap bahasa. Oleh karena itu, *Multilingual sentiment analysis* menggunakan kemampuan setiap bahasa secara khusus untuk mengekstrak sentimen dari teks yang dipelajari.

Tugas utama analisis sentimen adalah mengelompokkan teks yang terdapat dalam sebuah kalimat atau dokumen dan menentukan apakah pendapat yang diungkapkan dalam kalimat atau dokumen tersebut bersifat positif, negatif atau netral [16]. Informasi mengenai sentimen pengguna dapat dimanfaatkan untuk membuat strategi bisnis yang lebih baik, mengalokasikan sumber daya dengan lebih optimal, serta merencanakan langkah-langkah pengembangan produk berdasarkan preferensi dan harapan pengguna [17]. Analisis sentimen dapat dikategorikan menjadi tiga kelas sentimen, yaitu [18]:

1. Sentimen Positif

   Sentimen Positif didefinisikan dalam Kamus Besar Bahasa Indonesia (KBBI) sebagai tindakan atau sikap yang meningkatkan nilai seseorang atau sesuatu.

2. Sentimen Negatif

   Sentimen negatif menurut Kamus Besar Bahasa Indonesia (KBBI) adalah sikap atau respon yang menurunkan nilai seseorang atau sesuatu. Kalimat dengan sentimen negatif mengubah perspektif tentang sesuatu yang menyebabkan tren turun.

3. Sentimen Netral

   Menurut Kamus Besar Bahasa Indonesia (KBBI), kata "netral" berarti "tidak berpihak" dan kalimat bersentimen netral adalah kalimat yang tidak memiliki sifat positif atau negatif.

2.2 Machine Learning

Machine learning adalah kumpulan teknik yang bermanfaat untuk menangani dan memprediksi data yang berskala besar dengan cara menggunakan algoritma pembelajaran yang diterapkan pada data tersebut [19]. *Machine learning* juga dapat didefinisikan sebagai penggunaan komputer dan algoritma matematika untuk membuat prediksi di masa depan dengan belajar dari data yang sudah ada [20]. *Machine learning* terbagi menjadi empat jenis, yaitu:

1. *Supervised Learning*

   Teknik *supervised learning* digunakan untuk mengidentifikasikan hubungan di antara atribut *input* dan atribut target dimana klasifikasi adalah salah satu kategori untuk memprediksi keanggotaan grup di dalam contoh data, klasifikasi merupakan teknik data *mining*. *Logistic Regression, Support Vector Machine*, *Random Forest Classifier, Naive Bayes Bernoulli* adalah algoritma dari *supervised learning* [21].

2. *Unsupervised Learning*

   Unsupervised learning sangat baik untuk mengelola atau mengklasifikasi suatu pola dari banyaknya objek yang sejenis dimana objeknya tidak sepenuhnya sama. Tujuan dari *unsupervised learning* adalah untuk membuat penggunanya dapat mengelompokkan objek – objek yang memiliki nilai yang sama dalam ruang lingkup tertentu dimana *clustering* adalah proses membagi data ke dalam sejumlah kelompok tertentu [22]. Algoritma *unsupervised learning* adalah *K – Means, Hierarchical Clustering, DBSCAN* dan sebagainya.

3. *Semi-supervised Learning*

   Semi-supervised learning adalah jenis *machine learning* dimana menggabungkan antara data yang mempunyai label dan data yang tidak memiliki label dan melatih model tersebut untuk melakukan tugas regresi dan klasifikasi. Dimana ini relevan di dalam situasi dimana akan mendapatkan sejumlah besar data yang berlabel cukup sulit atau mahal, tetapi untuk mendapatkan sejumlah data yang berlabel lebih mudah [23].

4. *Reinforcement Learning*

   Reinforcement learning mengumpulkan pengetahuan untuk memilih tindakan yang akan menghasilkan hasil terbaik yang diinginkan dimana dia memiliki banyak keunggulan ketika dibandingkan dengan jenis lainnya dimana keunggulannya untuk menjelajahi sendiri lingkungan yang sangat dinamis dan stokastik dan membuat kebijakan kontrol yang ideal dengan menggunakan umpan balik evaluasi dari lingkungan tersebut [24].

Berdasarkan pada pendapat Peter Harington yang menjelaskan beberapa alur kinerja *machine learning*, yaitu [25]:

1. Mengumpulkan data, seperti *file Excel, Microsoft Access, file* teks dan lainnya.
2. Mempersiapkan data dengan cara menentukan kualitas data dan setelahnya mengambil langkah – langkah untuk memperbaiki masalahnya contohnya seperti kehilangan data.
3. Melatih sebuah model dengan membagi data yang sudah disiapkan menjadi dua bagian, yaitu data *training* yang digunakan dalam pengembangan model dan juga data *testing* digunakan sebagai referensi.
4. Mengevaluasi model dengan cara menentukan pilihan algoritma berdasarkan pada hasil dari pengujian.
5. Meningkatkan kinerja dengan memilih model yang lain atau dengan menambahkan lebih banyak variabel dalam hal untuk meningkatkan efisiensi.

Ada beberapa algoritma yang ada di dalam *machine learning* yaitu, *Support Vector Machine*, *K – Nearest Neighbors*, *Naive Bayes*, *Decision Tree*, *Random Forest* dan sebagainya. Dibandingkan dengan pendekatan berbasis aturan, *machine learning* dapat mengurangi jumlah pekerjaan yang dilakukan secara manual yang diperlukan untuk proses peninjauan sistematis. Ini membantu pengguna dalam membuat keputusan yang lebih akurat. Selain itu, ada kekurangan seperti kemungkinan sistem *machine learning* rentan terhadap penyusupan dan manipulasi melalui serangan yang berbahaya. Pihak yang melakukan serangan dapat dengan mudah memanfaatkan kelemahan dari sistem tersebut untuk mempengaruhi aktivitas yang awalnya tidak berbahaya menjadi berbahaya dikarenakan algoritma ini biasanya memprediksi kelas mayoritas dengan benar dan mereka sering

melakukan kesalahan saat mengidentifikasi kelas yang minoritas yang dapat menimbulkan ketidakseimbangan pada data [26].

2.3 Natural Language Processing (NLP)

NLP adalah cabang dari kecerdasan buatan (AI) dimana berhubungan dengan cara melatih komputer untuk memahami, memproses dan menghasilkan bahasa yang alami [27]. NLP juga melibatkan proses pemecahan beberapa tantangan di dalam pemprosesan bahasa manusia dimana tata bahasa, ambiguitas, penggunaan kata – kata yang berbeda dalam berbagai konteks dan ekspresi figuratif merupakan beberapa struktur bahasa manusia yang kompleks. Oleh karena itu, NLP menggunakan pendekatan komputasi untuk memahami dan memproses bahasa manusia [28]. Berikut adalah beberapa teori dan pendekatan yang digunakan di dalam NLP [29]:

1. Pemprosesan Bahasa Alami Berbasis Aturan (*Ruled-Based* NLP)
   Metode ini menggunakan aturan linguistik atau tata bahasa yang telah ditentukan sebelumnya dalam hal memahami dan memproses teks. Aturan ini mencakup sintaksis, semantik dan struktur bahasa dimana beberapa teknik yang digunakan termasuk pengurai sintaksis (*parser*) dan generator bahasa alami.
2. Pembelajaran Mesin dalam Pemprosesan Bahasa Alami (*Machine Learning in* NLP)
   Metode ini menggunakan algoritma *machine learning* untuk mengidentifikasi pola yang ada di dalam data bahasa alami dimana klasifikasi teks, ekstraksi informasi, pemodelan topik dan terjemahan mesin ialah beberapa contoh pekerjaan yang dapat dilakukan dengan algoritma ini.
3. Representasi Bahasa Alami (*Natural Language Representation*)
   Mengkodekan dan mengubah teks menjadi format yang mana dapat dipahami komputer dikenal sebagai representasi bahasa alami sementara representasi ini mencakup representasi kata, frase, dokumen serta representasi grafis untuk analisis sintaksis dan semantik. Vektor kata (*embedding* kata) merupakan model bahasa berbasis transformer dan memiliki grafik dependensi ini merupakan teknik representasi bahasa alami yang populer.

2.4 Pre-processing Data

Preprocessing adalah langkah pertama dalam proses analisis data [30]. *Preprocessing* sangat penting untuk analisis sentimen, karena *raw* data (data mentah) yang didapatkan bisa berupa data yang tidak terstruktur dan masih mengandung *noise*.

Preprocessing data adalah tahapan penting dalam analisis data dan pembelajaran mesin, yang melibatkan persiapan dan transformasi data mentah menjadi format yang lebih terstruktur dan siap untuk analisis [31]. Pada dasarnya, metode *preprocessing* data dibagi menjadi tiga cabang utama, yaitu [30]:

1. Data *Cleaning* (Pembersihan Data)

   Data yang dikumpulkan di dunia nyata seringkali mengandung nilai yang hilang dan *noise*. *Noise* adalah data yang tidak akurat yang disebabkan oleh kesalahan pengukuran atau kesalahan manusia dalam *dataset*. Mengidentifikasi dan membersihkan data *noise* ini merupakan salah satu tantangan dalam analisis data dan mengabaikan langkah ini dapat menyebabkan analisis yang tidak akurat dan keputusan yang tidak dapat diandalkan.
2. Data *Reduction* (Reduksi Data)

   Data yang dihasilkan oleh berbagai sensor dan aplikasi saat ini meningkat dengan cepat baik pada basis baris maupun kolom. Hal ini membuat analisis data menjadi lebih sulit dan menambah beban pada algoritma *machine learning* dan data mining. Hal ini dapat menyebabkan peningkatan kompleksitas dan waktu yang diperlukan untuk mencapai hasil. Selain itu, karena data yang tidak perlu dan tidak relevan yang terkandung dalam algoritma analisis data memungkinkan tidak dapat menghasilkan informasi yang akurat. Oleh karena itu, perlu untuk mengurangi ukuran data tanpa mengurangi kualitasnya.
3. Data *Transformation* (Transformasi Data)

   Metode data *transformation* adalah metode yang mengubah data ke dalam format yang sesuai untuk algoritma data mining. Data yang tidak terstruktur dengan baik dapat mengurangi efisiensi dan kinerja model data mining. Dua metode transformasi data penting adalah normalisasi dan agregasi data.

Adapun proses yang dilakukan dalam tahapan *preprocessing* sesuai kebutuhan penelitian adalah sebagai berikut:

1. Mengubah semua karakter huruf pada teks menjadi huruf kecil (*lowercase*) agar seragam.
2. Melakukan penghapusan atau menghilangkan simbol (@#&~), tanda baca, *link*, *emoticon*, angka, dan lain-lain yang tidak memiliki arti pada klasifikasi.
3. *Standardization* yaitu mengubah kata yang sebelumnya tidak baku menjadi baku sesuai Kamus Besar Bahasa Indonesia (KBBI) agar mudah dipahami dan kata-kata dalam bahasa inggris diterjemahkan ke dalam bahasa indonesia.
4. *Tokenization* yaitu memisahkan setiap kata pada data teks sehingga menjadi satuan kata.

5. *Stopwords removal* yaitu menghilangkan kata-kata yang tidak memiliki arti atau tidak relevan dengan teks, seperti ‘dan’, ‘atau’ ‘ke’, ‘yang’, ‘dari’.
6. *Stemming* yaitu mengubah kata yang berimbuhan menjadi kata dasar, contohnya kata ‘terbaik’ menjadi ‘baik’.

2.5 Term Frequency - Inverse Document Frequency (TF-IDF)

Term Frequency - Inverse Document Frequency atau biasa dikenal dengan sebutan TF - IDF ialah metode pembobotan kata atau *term* dimana memberikan bobot yang berbeda untuk setiap kata yang terdapat dalam sebuah dokumen dan didasarkan pada frekuensi kata di setiap dokumen serta total dari setiap frekuensi kata dalam dokumen tersebut [32]. *Term frequency* bisa dianggap memiliki proporsi kepentingan berdasarkan berapa kali jumlah kemunculannya dalam suatu teks atau dokumen. Untuk melakukan pengawasan terhadap kemunculan token di dalam sebuah himpunan teks, biasanya menggunakan metode pembobotan token yaitu *Inverse Document Frequency* (IDF) [33]. Berikut merupakan rumus dari *Term frequency - Inverse Document Frequency* (TF - IDF):

1. Menghitung nilai *Term Frequency* (TF)

   Menghitung frekuensi kata atau *term* yang muncul di dalam dokumen. Dituliskan dengan rumus:

   $$TF(t,d) = f(t,d) \quad (2.1)$$

   dimana:

   ft, *d* = jumlah kemunculan *term t* di dalam dokumen *d*

   Untuk mengurangi efek dari frekuensi yang sangat tinggi, TF biasanya dihitung dengan rumus:

   $$TF(t,d) = 1 + \log(ft,d) \quad (2.2)$$

   jika *term* tidak muncul di dalam dokumen maka nilai TF-nya adalah 0.

2. Menghitung nilai *Inverse Document Frequency* (IDF)

   Mengukur seberapa berpengaruhnya suatu kata dalam satu koleksi dokumen. Dituliskan dengan rumus:

   $$IDF(t) = \log\left(\frac{N}{df(t)}\right) \quad (2.3)$$

   dimana:

   N = total jumlah dokumen

   df(t) = jumlah dokumen yang mengandung kata atau *term t*.

Bertujuan untuk dapat memberikan bobot yang lebih besar pada kata -kata yang jarang muncul di dalam suatu koleksi dokumen.

3. Menghitung nilai TF – IDF

   Setelah dilakukannya proses menghitung nilai *tf* dan *idf* maka nilai *tf - idf* untuk *term* yang berada di dalam dokumen dapat dituliskan dengan rumus:

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \quad \text{.....(2.4)}$$

   memberikan bobot yang lebih besar pada kata yang sering muncul di dalam dokumen tertentu tetapi jarang muncul di dalam seluruh koleksi dokumen, sehingga dapat membantu dalam menentukan relevansi dari kata itu.

2.6 Chi – Square

Chi-square adalah seleksi fitur yang sering digunakan dalam pengklasifikasian teks karena berguna untuk mengurangi fitur yang tidak relevan [34]. Metode seleksi fitur *chi-square* memiliki kemampuan untuk menghilangkan banyak fitur tanpa mengurangi akurasi yang dihasilkannya [35] dan dapat mengurangi jumlah data yang sangat besar sehingga proses yang dibutuhkan menjadi cepat [36]. Dalam proses perhitungan yang dilakukan pada seleksi fitur *chi-square* ini, teori statistika digunakan untuk menguji hubungan antar kata. Berikut ini dituliskan rumus persamaan *chi-square*.

$$x^2(t,c) = \frac{N(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} \quad \text{.....(2.5)}$$

Keterangan:

t = *term* atau kata

c = kategori atau kelas sentimen

N = jumlah keseluruhan data dalam dokumen teks

A = banyaknya dokumen dalam kelas *c* yang mengandung kata *t*

B = banyaknya dokumen yang bukan kelas *c* tetapi mengandung kata *t*

C = banyaknya dokumen dalam kelas *c* tetapi tidak mengandung kata *t*

D = banyaknya dokumen yang bukan kelas *c* dan tidak mengandung kata *t*

Rumus pada uji *chi – square* jika tabel kontingensi yang digunakan adalah 2 x 2 maka rumus yang digunakan ialah *continuty correction.* Jika, tabel kontingensi 2 x 2 tetapi tidak memenuhi syarat dalam uji *chi – square* maka rumus yang digunakan adalah *fisher exact test* dan jika, tabel kontingensi lebih dari 2 x 2 maka rumus yang digunakan ialah *pearson chi – square*. Pengujian pada *chi – square* dapat dirumuskan sebagai berikut.

$$x^2 = \sum \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \quad \text{.....(2.6)}$$

Keterangan:

x^2 : distribusi *chi – square*

O_{ij} : frekuensi observasi (nilai yang sebenarnya terjadi) terhadap baris ke – *i* dan kolom ke - *j* di dalam tabel kontingensi

E_{ij} : frekuensi ekspektasi (nilai yang diharapkan) terhadap baris ke – *i* dan kolom ke – *j*

Rumus mencari nilai pada frekuensi harapan:

$$E_{\text{ij}} = \frac{(\text{total baris } i) \times (\text{total baris } j)}{N} \quad \text{.....(2.7)}$$

Keterangan:

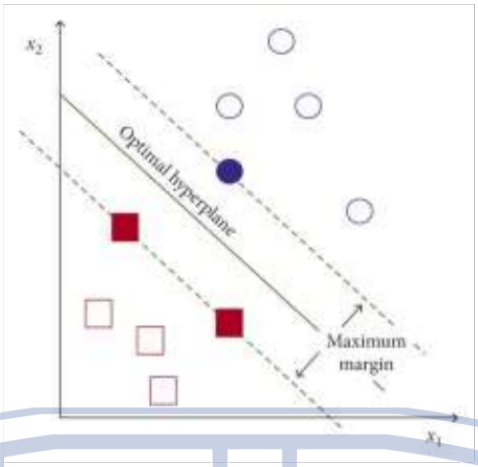
N : total keseluruhan

Setelah melakukan penilaian untuk setiap kata yang diketahui, selanjutnya urutkan kata-kata berdasarkan nilai *chi-square* dari yang tertinggi hingga terendah [34]. Nilai *chi-square* yang lebih tinggi menunjukkan bahwa fitur tersebut lebih penting untuk klasifikasi. Akibatnya, nilai kritis berkorelasi negatif dengan nilai taraf nyata (α). Semakin rendah nilai taraf nyata (α) yang diterapkan, maka semakin banyak fitur yang akan diseleksi atau dibuang.

2.7 Support Vector Machine

Pada tahun 1992, Vladimir N. Vapnik, seorang profesor di Columbia University di Amerika Serikat memperkenalkan sebuah algoritma pelatihan yang bertujuan untuk memaksimalkan *margin* di antara pola pelatihan dan batas keputusan (*decision boundary*). Algoritma ini kemudian dikenal luas sebagai *Support Vector Machine* (SVM) [37]. SVM adalah salah satu teknik prediksi yang baik untuk klasifikasi dan regresi [38]. SVM dapat membedakan dua kelas dengan cara menemukan *hyperplane* yang paling optimal dengan memaksimalkan nilai *margin* antara titik data yang terdekat dari kelas yang berlawanan atau berbeda [39].

Fungsi *hyperplane* digunakan untuk pemisah antar kelasnya [40]. Pada Gambar 2.1 digambarkan bahwa garis *solid* pada gambar adalah *hyperplane* yang optimal atau nilai *hyperplane* terbaik, dimana posisinya tepat berada diantara kedua kelas. *Margin* adalah jarak antar data pada masing - masing kelas dari *hyperplane* [41]. *Support vector* ialah data yang berada dalam bidang pembatas dimana, bidang pembatas ini digambarkan pada gambar dengan garis putus – putus [42].



Gambar 2.1 Klasifikasi *Linear*
Sumber: https://towardsai.net

Data ditunjukkan sebagai $\{x_i , \ldots, x_n\}$ dan label pada kelas ditunjukkan sebagai y_i $e\{+1, -1\}$ untuk $i = 1, 2, 3, \ldots, n$. Setiap bidang pembatas memiliki kelasnya tersendiri sehingga dapat diperoleh persamaan yaitu:

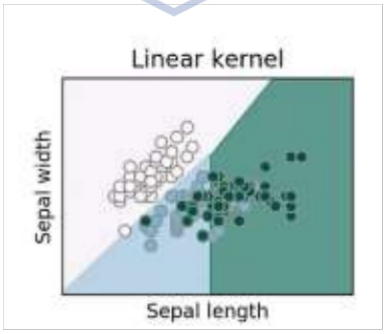
$$x_i w + b \geq +1,\ y_i = +1 \quad \text{.....(2.8)}$$

$$x_i w + b \geq -1, y_i = -1 \quad \text{.....(2.9)}$$

Pada persamaan (1) dan (2), w adalah sebuah bidang normal dan b adalah sebuah bidang relatif terhadap pusat koordinat. Data yang dapat dipisahkan secara *linear* oleh SVM disebut sebagai SVM *linear* tetapi, dalam kasus dimana data tidak dapat dipisahkan secara *linear* oleh SVM dapat menggunakan teknik transformasi *kernel* untuk mengubah data ke dimensi fitur yang lebih besar [43]. *Kernel* yang umum digunakan pada SVM yaitu [44]:

1. *Kernel Linear*

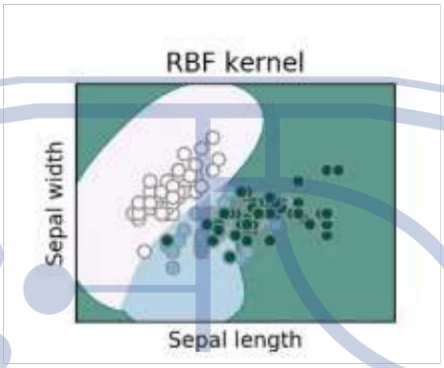
   Pada Gambar 2.2 menunjukkan bagaimana *kernel linear* dapat merepresentasikan data dan bagaimana klasifikasi data dapat dilakukan. *Kernel linear* adalah *kernel* yang paling sederhana dengan menyediakan metode klasifikasi yang konvensional di dalam SVM. Dapat digunakan ketika data sudah dipisahkan secara *linear*, artinya tidak ada lagi transformasi yang di terapkan di dalam data.



Gambar 2.2 *Linear Kernel*
Sumber: www.analyticsvidhya.com

2. *Kernel* Fungsi Basis *Radial* (RBF)

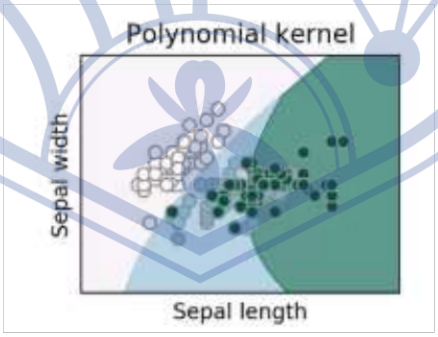
   Karena kemampuan *kernel* ini yang dapat memberikan hasil klasifikasi yang tinggi serta dikarenakan *kernel* ini dapat menangani hubungan yang sangat kompleks dan *non - linear* menjadikan *kernel* ini sebagai *kernel* yang paling sering atau paling umum digunakan pada SVM. *Kernel* ini juga biasa dikenal sebagai *kernel Gaussian*. Pada Gambar 2.3 menampilkan representasi dari *kernel* RBF.



Gambar 2.3 RBF *Kernel*
Sumber: www.analyticsvidhya.com

3. *Kernel Polynomial*

   Kernel ini sesuai dengan data yang sudah dilakukan normalisasi dengan menggunakan nilai derajat yang dapat digunakan pada *kernel* ini untuk menentukan fleksibilitas *margin* yang akan digunakan untuk mengklasifikasikan data. *Kernel* ini juga dapat menangkap interaksi antar fitur - fiturnya hingga tingkat tertentu. Gambar 2.4 menampilkan representasi dari *kernel polynomial*.

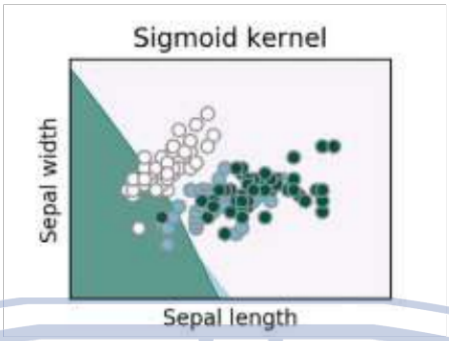


Gambar 2.4 *Polynomial Kernel*
Sumber: www.analyticsvidhya.com

4. *Kernel Sigmoid*

   Fungsi dari *kernel* ini mirip dengan jaringan syaraf, tetapi *kernel* ini melakukan pengklasifikasian paling sedikit diantara *kernel* lainnya. Pengklasifikasian *kernel* ini

dilakukan dengan membuat dua batas keputusan. Gambar 2.5 menampilkan representasi *kernel sigmoid*.



Gambar 2.5 *Sigmoid Kernel*
Sumber: www.analyticsvidhya.com

Pada dasarnya, SVM bekerja untuk melakukan klasifikasi biner dan klasifikasi *linear* namun, setelah dikembangkan SVM sekarang dapat melakukan pengklasifikasian *multiclass* dengan menggunakan teknik *one against one* (OAO), *one against all* (OAA), dan *directed acyclic graph* [45]. OAO menyelesaikan masalah *multiclass* dengan *decision boundary* $\frac{n(n-1)}{2}$ dari hasil pencarian *hyperplane* dari kelas ke – *i* dengan kelas lainnya. Sebaliknya pendekatan OAA menyelesaikan masalah *multiclass* dengan *decision boundary n* dari hasil pencarian *hyperplane* dari kelas ke – *i* dengan kelas lainnya [46]. Terdapat dua metode yang digunakan SVM untuk menentukan batas pemisah (*hyperplane*) antara kelas data yaitu:

1. *Soft Margin*

   Soft margin SVM digunakan untuk menangani kasus dimana data tidak dapat dipisahkan secara linier atau ketika variabel kurang memungkinkan berada di beberapa ketidakcocokan dalam titik data di area tertentu [47].

2. *Hard Margin*

   Masalah dengan *hard margin* SVM adalah jika di dalam *dataset* data tidak dapat dipisahkan secara *linear* ini berarti tidak mungkin untuk melakukan klasifikasi data karena tidak ditemukannya *hyperplane* pemisah dan *margin* sangat terpengaruh oleh *noise* atau *outlier*. Penjelasan ini didasarkan kepada asumsi dimana *hyperplane* dapat memisahkan kedua kelas secara sempurna. Namun, hal yang sering terjadi yaitu kedua kelas tersebut tidak dapat dipisahkan secara sempurna dan dapat menyebabkan masalah yang tidak dapat diselesaikan dalam proses optimisasi. Dapat dilakukan dengan menggunakan *soft margin* [48].

Algoritma SVM umum digunakan pada saat melakukan klasifikasi data dikarenakan SVM adalah algoritma yang sangat efektif dan stabil dalam ruang dimensi tinggi. SVM juga

memiliki akurasi yang tinggi dan mudah untuk dilatih dibandingkan dengan algoritma pembelajaran mesin yang lain serta pemetaan yang dilakukan pada *kernel* ke ruang fitur yang berdimensi tinggi dapat menghemat memori [49].

2.8 Confusion Matrix

Confusion matrix yang juga dikenal sebagai matriks kebingungan adalah alat untuk menggambarkan performa model klasifikasi pada data uji yang hasil sebenarnya sudah diketahui [50]. *Confusion matrix* digunakan untuk memvisualisasikan serta untuk menganalisis hasil prediksi yang diperoleh dari model, agar lebih mudah untuk dipahami kelebihan dan kekurangan model dalam proses klasifikasi. Pada kasus klasifikasi lebih dari dua kelas digunakan *multiclass confusion matrix*, yang tentu berbeda dengan *confusion matrix* 2 kelas. Perhitungan *confusion matrix* dilakukan dengan mencari nilai *True Positive* (TP), *False Positive* (FP), *False Negative* (FN) dan *True Negative* (TN) untuk masing-masing kelas [51]. Bentuk umum *confusion matrix* untuk tiga kelas dapat dilihat pada Tabel 2.1 [52].

Tabel 2.1 *Confusion Matrix* Untuk Klasifikasi Tiga Kelas

Kelas Aktual	**Kelas Prediksi**		
	Negatif	**Netral**	**Positif**
Negatif	c11	c12	c13
Netral	c21	c22	c23
Positif	c31	c32	c33

Menentukan nilai TP, FP, FN, dan TN dilakukan berdasarkan setiap kelas bukan untuk keseluruhan. Untuk kelas Negatif, didefinisikan:

1. $TP_{Negatif}$: data kelas Negatif diprediksi sebagai kelas Negatif (c11).
2. $FP_{Negatif}$: data selain kelas Negatif diprediksi sebagai kelas Negatif (c21 + c31).
3. $FN_{Negatif}$: data kelas Negatif diprediksi sebagai kelas lain (c12 + c13).
4. $TN_{Negatif}$: data selain kelas Negatif diprediksi sebagai selain kelas Negatif (c22 + c23 + c32 + c33). Prinsip yang sama diterapkan untuk kelas Netral dan Positif.

Tabel 2.2 Elemen *Confusion Matrix* Untuk Kelas Negatif

Kelas Aktual	**Kelas Prediksi**		
	Negatif	**Netral**	**Positif**
Negatif	$TP_{Negatif}$	$FN_{Negatif}$	$FN_{Negatif}$
Netral	$FP_{Negatif}$	$TN_{Negatif}$	$TN_{Negatif}$
Positif	$FP_{Negatif}$	$TN_{Negatif}$	$TN_{Negatif}$

Tabel 2.3 Elemen *Confusion Matrix* Untuk Kelas Netral

Kelas Aktual	**Kelas Prediksi**		
	Negatif	**Netral**	**Positif**
Negatif	TN_{Netral}	FP_{Netral}	TN_{Netral}
Netral	FN_{Netral}	TP_{Netral}	FN_{Netral}
Positif	TN_{Netral}	FP_{Netral}	TN_{Netral}

Tabel 2.4 Elemen *Confusion Matrix* Untuk Kelas Positif

Kelas Aktual	**Kelas Prediksi**		
	Negatif	**Netral**	**Positif**
Negatif	$TN_{Positif}$	$TN_{Positif}$	$FP_{Positif}$
Netral	$TN_{Positif}$	$TN_{Positif}$	$FP_{Positif}$
Positif	$FN_{Positif}$	$FN_{Positif}$	$TP_{Positif}$

$TP_{Negatif}$, TP_{Netral}, dan $TP_{Positif}$ adalah elemen *diagonal* dari *confusion matrix* pada Tabel 2.1 yaitu c11, c22, dan c33. *Confusion matrix* 3x3 pada Tabel 2.1, dapat diuraikan menjadi tiga *confusion matrix* 2x2 yang ditunjukkan pada tabel berikut.

Tabel 2.5 Kelas Negatif

Kelas Aktual	**Kelas Prediksi**	
	Negatif	Bukan Negatif
Negatif	$TP_{Negatif}$	$FN_{Negatif} = c12 + c13$
Bukan Negatif	$FP_{Negatif} = c21 + c31$	$TN_{Negatif} = c22 + c23 + c32 + c33$

Tabel 2.6 Kelas Netral

Kelas Aktual	**Kelas Prediksi**	
	Netral	Bukan Netral
Netral	TP_{Netral}	FN_{Netral} = c21 + c23
Bukan Netral	FP_{Netral} = c12 + c32	TN_{Netral} = c11 + c13 + c31 + c33

Tabel 2.7 Kelas Positif

Kelas Aktual	**Kelas Prediksi**	
	Positif	Bukan Positif
Positif	$TP_{Positif}$	$FN_{Positif}$ = c31 + c32
Bukan Positif	$FP_{Positif}$ = c13 + c23	$TN_{Positif}$ = c11 + c12 + c21 + c22

Dari elemen *confusion matrix* tersebut dapat dihitung berbagai metrik evaluasi, yaitu [53]:

1. *Accuracy*

   Accuracy adalah metrik yang mengukur jumlah prediksi benar dibagi dengan jumlah data aktual. Persamaannya dapat dituliskan sebagai berikut:

   $$Accuracy = \frac{TP\ per\ kelas}{Total} \quad \text{.....(2.10)}$$

2. *Precision*

   Precision adalah metrik yang dihitung dengan membagi total prediksi yang benar dengan jumlah keseluruhan yang diprediksi. Persamaannya dapat dituliskan sebagai berikut:

   $$Precision = \frac{TP}{TP+FP} \quad \text{.....(2.11)}$$

3. *Recall*

   Recall adalah jumlah prediksi yang benar dibagi dengan jumlah keseluruhan yang sebenarnya. Persamaannya dapat dituliskan sebagai berikut:

   $$Recall = \frac{TP}{TP+FN} \quad \text{.....(2.12)}$$

4. *F1 – score*

   F1 – score adalah metrik yang mengukur harmonisasi antara *precision* dan *recall*. Dapat dihitung dengan persamaan berikut:

   $$F1 - score = \frac{2 \times Precision \times Recall}{Precision+Recall} \quad \text{.....(2.13)}$$

5. *Weighted average*

   Weighted average adalah menghitung rata-rata berbagai metrik evaluasi, seperti akurasi, *precision*, *recall*, dan *f1-score* dengan memperhitungkan total sampel setiap kelas [54]. *Weighted average* untuk *precision*, *recall*, dan *f1-score* dapat dihitung dengan persamaan berikut [52]:

$$Precision_{weighted} = \frac{N_A \times (Precision_A) + N_B \times (Precision_B) + N_C \times (Precision_C)}{N} \quad .....(2.14)$$

$$Recall_{weighted} = \frac{N_A \times (Recall_A) + N_B \times (Recall_B) + N_C \times (Recall_C)}{N} \quad .....(2.15)$$

$$F1_{weighted} = \frac{2 \times Precision_{weighted} \times Recall_{weighted}}{Precision_{weighted} + Recall_{weighted}} \quad .....(2.16)$$

   Keterangan:

   A, B, C = kategori atau kelas

   N = jumlah sampel

   N_A, N_B, N_C = jumlah sampel per kelas

2.9 Visualisasi Data

Visualisasi data adalah teknik untuk menampilkan data dalam bentuk diagram, grafik atau peta [55]. Visualisasi data sangat bermanfaat untuk membersihkan data dan mempelajari struktur data, menemukan tren dan klaster, menemukan *outlier* dan kelompok yang tidak biasa, serta menunjukkan hasil [56]. Visualisasi data bukan hanya membuat data terlihat lebih menarik secara visual tetapi juga membuat data lebih mudah dipahami dan lebih efisien menyampaikan informasi. Berdasarkan tujuan dan jenis data yang ditampilkan, ada berbagai tipe visualisasi data yang dapat digunakan yaitu sebagai berikut [57]:

1. Grafik Garis (*Line Chart*)

   Grafik garis adalah jenis visualisasi data yang digunakan untuk menunjukkan perubahan data dari waktu ke waktu.

2. Diagram Batang (*Bar Chart*)

   Diagram batang digunakan untuk menunjukkan perbandingan antara berbagai kategori.

3. Diagram *Pie* (*Pie Chart*)

   Diagram *pie* berfungsi untuk menunjukkan proporsi nilai dalam satu kategori.

4. Peta (*Map*)

   Peta adalah visualisasi data yang menunjukkan distribusi geografis data.

5. *Heatmap*

   Heatmap digunakan untuk menampilkan perbandingan antara dua variabel dalam grafik warna.

2.10 Database

Database atau basis data adalah kumpulan data yang disimpan di dalam komputer dan dapat diakses, diubah, dan diorganisir dengan bantuan *Database Management System* (DBMS). *Database* memiliki beberapa model, dimana *relational* data model merupakan model yang paling populer dan memungkinkan untuk menyimpan data dalam sebuah tabel ataupun beberapa tabel sebagai *value* yang memiliki relasi antar satu sama lainnya [58]. Contoh *database* yang ada untuk saat ini yaitu *Microsoft SQL server, Oracle Database, MySQL, PostgreSQL, dan IBM Db2* [59]. *Database* sendiri memiliki beberapa fungsi dan juga beberapa jenis dari *database* itu sendiri yaitu [60]:

Fungsi dari *database* adalah sebagai berikut:

1. Pengelompokkan data untuk mempermudah proses identifikasi data, misalnya dengan membuat beberapa tabel atau *field* yang berbeda.
2. Mengurangi jika ada data yang ganda atau *double*.
3. Mempermudah dalam pengalaman pengguna di berbagai situasi, seperti ketika ingin menambahkan data baru.
4. Penyimpanannya dilakukan secara digital.
5. Menjadi alternatif lain untuk masalah dalam penyimpanan ruang di aplikasi.

Jenis-jenis dari *database*:

1. *Operational Database*, *database* ini menangani penyimpanan data yang sangat rumit, sehingga bisa digunakan dengan mudah. Biasa digunakan untuk *database* pelanggan.
2. *Relational Database*, *database* ini memungkinkan pengguna untuk bisa mengakses dan mencari informasi di dalam tabel atau *field* yang berbeda.
3. *Distributed Database*, *database* ini mendistribusikan data yang tersebar namun tetap saling berhubungan dan dapat diakses secara bersamaan.
4. *External Database*, *database* ini digunakan untuk kebutuhan komersial dikarenakan kemudahan dalam mengaksesnya dan dikhususkan untuk publik.

MySQL ialah *Relational Database Management System* (RDBMS) yang *open source* dan mendukung banyak pengguna, banyak *thread*, *database* yang populer dan gratis untuk digunakan dan menggunakan perintah SQL (*Structured Query Language*) [61, 62].

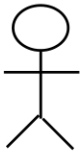
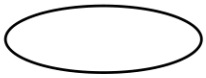
Dikarenakan keandalan, kecepatan serta kemudahan untuk penggunaanya, MySQL menjadi *database* pilihan untuk para pengembang perangkat lunak dan aplikasi pada platform *online* maupun *desktop* [63]. Basis data berisi beberapa struktur yaitu [64]:

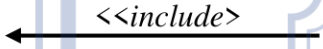
1. Bit adalah angka biner yang hanya terdiri dari dua angka yaitu 0 dan 1.
2. *Byte* adalah bagian paling kecil yang dapat berupa karakter numerik, huruf atau karakter khusus.
3. *Field* menunjukkan atribut suatu *record* dimana berisi suatu item yang berasal dari data, seperti nama atau alamat. Kumpulan dari *field* disebut *record*.
4. *Record* adalah sebuah unit data individu yang tertentu dimana kumpulan *record* membentuk suatu *file*.
5. *File* berisi *record – record* yang menunjukkan satu data yang sejenis.
6. Basis data terdiri dari sekumpulan berbagai jenis *record* yang memiliki hubungan terhadap suatu hal tertentu.
7. Sistem basis data terdiri dari basis data yang tersusun dari beberapa *file*.

2.11 Use Case Diagram

Use case menjelaskan fungsi dari sistem dan apa yang akan diprosesnya. *Use case* menggunakan skenario yang menunjukkan bagaimana urutan atau langkah – langkah yang menjelaskan apa yang dilakukan pengguna dengan sistem dan sebaliknya [65]. *Use case* bergantung pada jenis format yang dibutuhkan dan dapat ditulis secara informal, singkat atau lengkap. Selain itu, *use case scenario* menceritakan kondisi awal dan kondisi akhir sistem. Aktor, kondisi awal dan kondisi akhir serta skenario utama dan skenario alternatif adalah komponen penting di dalam skenario. *Use case diagram* menunjukkan hubungan antara aktor dan sistem. Aktor, *use case*, asosiasi, *include*, *extend*, dan hubungan generalisasi dimana [66]:

Tabel 2.8 Elemen *Usecase*

Simbol	Keterangan
	Aktor merupakan orang atau sistem yang memperoleh manfaat dari subjek dan bersifat eksternal. Disimbolkan dengan gambar orang – orangan seperti *stickman*
	*Use case* memiliki bentuk elips dengan nama kata kerja yang aktif dimana di

	dalamnya menceritakan aktivitas dari sudut pandang aktor. Setiap *use case* memiliki kemampuan untuk berinteraksi dengan sistem dan banyak aktor dapat menjalankan satu *use case*.
	Association merupakan abstraksi dan interaksi di antara *use case* dan aktor.
	Dalam generalisasi, arah panah mengarah pada kasus yang umum dan hubungan generalisasi adalah hubungan di antara umum dan khusus.
	*Include* adalah *use case* tambahan dimana *use case* ini dibutuhkan untuk menjalankan fungsi tersebut dengan arah panah menuju *use case* yang dibutuhkan
	Extend berfungsi sebagai hubungan tambahan ke sebuah *use case* dimana *use case* tersebut dapat berdiri sendiri tanpa *use case* tambahan dan arah panah mengarah ke *use case* yang ditambahkan.

2.12 BRImo

BRImo adalah aplikasi *mobile banking* yang diluncurkan oleh pihak Bank Rakyat Indonesia (BRI), aplikasi BRImo dirilis pada bulan Februari tahun 2019. Sebelumnya pihak bank BRI telah membuat sebuah aplikasi BRI *Mobile Banking* yang dimana aplikasi ini adalah layanan berbasis teknologi. Pada aplikasi ini hanya menyediakan layanan internet *banking*, info BRI dan Tbank BRI dimana saat ini aplikasinya telah berkembang menjadi BRImo [67]. Aplikasi BRImo ditujukan kepada para kalangan milenial dimana, yang terkait dengan layanan yang dilakukan secara digital seperti *fingerprint*, *face recognition*, cek mutasi, dan fitur lainnya dengan memiliki data internet dalam melakukan transaksi dengan menggunakan *user interface* dan *user experience* [68]. Keunggulan dari aplikasi BRImo yaitu [69]:

1. Dapat melakukan pembukaan untuk rekening tabungan yang baru

   Di aplikasi BRImo, pengguna bisa melakukan pembukaan rekening tabungan yang baru tanpa harus mendatangi bank terdekat dari lokasi rumah dengan melengkapi beberapa persyaratan seperti, memiliki KTP, mempunyai nomor telepon dan alamat *email* yang aktif, memiliki saldo pulsa minimal 2.000 supaya kode OTP yang dikirimkan pihak BRI bisa masuk ke nomor telepon aktif yang di daftar sebelumnya, serta uang setoran pertama sesuai dengan jenis rekening yang diambil untuk membuka rekening baru.

2. Dapat melakukan tarik dan setor tunai tanpa menggunakan kartu ATM

   Jika pengguna aplikasi BRImo belum memiliki kartu ATM, *chip* pada kartu ATM tidak dapat terbaca lagi, kartu ATM rusak atau terblokir dari sistem. Pengguna aplikasi BRImo tetap bisa melakukan transaksi tarik maupun setor tunai hanya dengan menggunakan aplikasi BRImo pada mesin ATM tarik dan setor tunai.

3. Dapat melakukan transfer

   Pengguna aplikasi BRImo bisa melakukan transfer uang tanpa datang ke bank ataupun mesin ATM, dikarenakan pada aplikasi BRImo pengguna bisa melakukan transfer uang dimana saja dan kapan saja baik antar sesama pengguna bank BRI ataupun dengan pengguna bank lainnya. Dalam aplikasi BRImo juga menyediakan QR pedagang dan layanan transfer untuk bank yang ada di luar negeri.

4. Dapat melakukan pembayaran untuk tagihan

   Pada aplikasi BRImo pengguna dapat melakukan pembayaran untuk tagihan seperti, pembayaran listrik, telkom, briva, LTMPT, pasca bayar, PDAM, pendidikan, TV kabel dan internet, asuransi, cicilan, dan kartu kredit.

5. Dapat melakukan transaksi untuk *Top Up*

   Pada aplikasi BRImo pengguna juga bisa melakukan *top up Brizzi* bisa dilakukan menggunakan aplikasi BRImo tanpa perlu mendatangi bank ataupun ATM terdekat. Selain bisa melakukan *top up* untuk Brizzi, di dalam aplikasi BRImo juga bisa melakukan *top up* untuk dompet digital seperti *DANA, ShopeePay, OVO, GoPay*, dan lain - lain.

6. Dapat melihat laporan keuangan

   Pada aplikasi BRImo, pengguna dapat melihat mutasi yang terjadi saat itu juga, kemarin, satu minggu, satu bulan serta dapat juga melihat tanggalnya tanpa perlu datang untuk mencetak di buku rekening. Terdapat juga menu catatan keuangan dimana dapat melihat pengeluaran, pemasukan dan laporan transaksi yang pernah dilakukan.

7. Dapat melakukan investasi

   Pada aplikasi BRImo juga dapat melakukan ivestasi e-SBN, RDN, emas, Deposito , dan lain - lain.

8. Dapat melakukan pembayaran iuran atau donasi

   Pada aplikasi BRImo, pengguna bisa melakukan pembayaran untuk iuran BPJS serta dapat memberikan donasi kepada yang lebih membutuhkan melalui menu donasi pada aplikasi BRImo.

9. Dapat membeli tiket untuk perjalanan

   Di aplikasi BRImo juga pengguna bisa membeli tiket perjalan yang terdapat di menu KAI dan *Travel*.

10. Dapat membayar pajak dan retribusi

    Di aplikasi BRImo pengguna bisa melakukan pembayaran untuk paspor, tilang polisi, dan lain - lain.