BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen, yang juga dikenal sebagai *opinion mining*, merupakan bidang dalam *Natural Language Processing* (NLP) yang berfokus pada identifikasi dan ekstraksi informasi subjektif dari sumber teks [10]. Tujuan utamanya adalah menentukan apakah suatu teks mengandung opini positif, negatif, atau netral terhadap suatu entitas, produk, layanan, atau kejadian [26].

Secara metodologis, analisis sentimen dapat dilakukan melalui tiga pendekatan utama. Pertama, pendekatan berbasis leksikon (*lexicon-based*) yang mengandalkan kamus sentimen berisi kata-kata dengan bobot polaritas yang telah ditentukan sebelumnya. Kedua, pendekatan berbasis *machine learning* yang melatih model klasifikasi pada data berlabel untuk mempelajari pola sentimen secara otomatis. Ketiga, pendekatan berbasis *deep learning* yang memanfaatkan arsitektur jaringan saraf tiruan seperti LSTM, GRU, dan BERT untuk menangkap representasi semantik kontekstual [10]. Penelitian ini menggunakan pendekatan *machine learning* dengan algoritma *Support Vector Machine* (SVM)

2.2 Natural Language Processing

Natural Language Processing (NLP) adalah subdisiplin ilmu komputer yang berada di persimpangan antara linguistik dan kecerdasan buatan. NLP memungkinkan mesin untuk memahami, menginterpretasi, dan menghasilkan bahasa manusia dalam bentuk teks maupun ucapan [27].

Pemrosesan teks berbahasa Indonesia menghadirkan tantangan tersendiri dibandingkan Bahasa Inggris, antara lain karena penggunaan afiksasi yang kaya (prefiks, sufiks, konfiks, dan infiks), banyaknya kata tidak baku dan singkatan informal pada teks media sosial serta ulasan aplikasi, serta minimnya sumber daya NLP standar dibandingkan Bahasa Inggris [26]. Oleh karena itu, tahapan *preprocessing* yang tepat menjadi sangat krusial dalam menghasilkan representasi fitur teks yang berkualitas untuk model *machine learning*.

2.3 Weak Supervision Labeling

Teknik *weak supervision labeling*. *Weak supervision* adalah pendekatan di mana label data (yang mungkin tidak sepenuhnya sempurna) dihasilkan secara otomatis menggunakan aturan heuristik, metadata, atau model eksternal, alih-alih menggunakan tenaga manusia (*human annotator*).

Pada penelitian ini, metadata berupa nilai *rating* atau skor (bintang 1 hingga 5) yang diberikan oleh pengguna bersamaan dengan ulasan teks dimanfaatkan sebagai aturan heuristik untuk menentukan polaritas sentimen. Berdasarkan kode yang diimplementasikan, proses pelabelan mengategorikan ulasan ke dalam tiga klasifikasi sentimen: positif, negatif, dan netral [28]. Aturan pelabelan didefinisikan sebagai berikut:

1. Sentimen Negatif: Ulasan yang memiliki nilai skor ≤ 2 (bintang 1 dan 2).
2. Sentimen Netral: Ulasan yang memiliki nilai skor tepat bernilai 3.
3. Sentimen Positif: Ulasan yang memiliki nilai skor > 3 (bintang 4 dan 5).

Selanjutnya, pelabelan kelas netral (skor 3) dipertahankan pada tahap awal untuk melihat distribusi data secara keseluruhan kemudian dibuang sebelum disimpan dalam bentuk dataset yang sudah berlabel karena, dalam banyak kasus analisis sentimen yang berfokus pada preferensi atau keluhan utama pengguna, ulasan bersentimen netral seringkali tidak memiliki muatan evaluasi yang eksplisit dan tidak secara jelas mencerminkan preferensi pengguna [28].

2.4 Preprocessing Teks Bahasa Indonesia

Tahapan *preprocessing* teks merupakan proses transformasi teks mentah menjadi representasi yang bersih dan terstandarisasi sebelum dilakukan ekstraksi fitur. Dalam penelitian ini digunakan enam tahapan *preprocessing* yang mengacu pada metode yang diusulkan oleh Putri dkk. [28], yaitu *cleaning*, *case folding*, *normalization*, *tokenization*, *stopword removal*, dan *stemming*. Selain itu, ditambahkan satu tahapan *negation marking* untuk mempertahankan informasi negasi dalam teks sehingga konteks polaritas sentimen dapat direpresentasikan dengan lebih baik [29]. .

2.4.1 Cleaning

Cleaning adalah tahapan pertama yang bertujuan menghilangkan elemen non-informatif dari teks mentah. Elemen yang dihapus meliputi URL dan tautan web, *mention* pengguna (@username) dan *hashtag* (#topik), karakter emoji dan simbol Unicode, angka, serta tanda baca dan karakter khusus [26]. Hasil dari tahapan ini adalah teks yang hanya mengandung karakter alfabetis dan spasi.

2.4.2 Casefolding

Casefolding adalah proses konversi seluruh karakter teks menjadi huruf kecil (*lowercase*). Tahapan ini memastikan bahwa kata-kata yang secara semantik identik namun berbeda kapitalisasi (misalnya "Bagus", "BAGUS", dan "bagus") diperlakukan sebagai token yang sama oleh model [27].

2.4.3 Normalisasi

Normalisasi bertujuan menstandarkan variasi penulisan yang umum ditemukan pada teks informal berbahasa Indonesia. Proses ini mencakup penggantian singkatan tidak baku (misalnya "yg" menjadi "yang", "gak" menjadi "tidak"), koreksi kesalahan ejaan umum, dan penyeragaman kata gaul menjadi padanan baku yang bermakna [26]. Kamus normalisasi yang komprehensif sangat diperlukan mengingat tingginya frekuensi penggunaan bahasa informal pada ulasan aplikasi.

2.4.4 Negation Marking

Negation marking merupakan teknik praproses yang digunakan untuk mempertahankan informasi negasi dalam teks. Pada analisis sentimen, keberadaan kata negasi seperti 'tidak", 'bukan", 'jangan", dan 'belum" dapat mengubah makna serta polaritas suatu kata atau frasa secara signifikan. Sebagai contoh, kata 'bagus" memiliki konotasi positif, sedangkan frasa 'tidak bagus" menunjukkan sentimen negatif. Oleh karena itu, informasi negasi perlu dipertahankan agar makna asli suatu ulasan tidak hilang selama proses ekstraksi fitur [30, 29].

Apabila negasi tidak ditangani secara khusus, metode ekstraksi fitur berbasis TF-IDF cenderung merepresentasikan kata 'bagus" dan 'tidak bagus" sebagai fitur yang serupa karena keduanya mengandung token utama yang sama. Kondisi ini

dapat menyebabkan model kesulitan membedakan polaritas sebenarnya dari suatu kalimat [29]. Untuk mengatasi permasalahan tersebut, penelitian ini menerapkan pendekatan *negation marking* dengan menambahkan prefiks `Neg_` pada token yang berada dalam cakupan negasi, misalnya 'tidak bagus" menjadi `Neg_bagus` dan 'tidak akurat" menjadi `Neg_akurat`.

Pendekatan tersebut termasuk dalam kategori *feature space augmentation*, yaitu memperkaya representasi fitur dengan menambahkan informasi negasi ke dalam token sehingga model dapat membedakan kata dalam konteks normal dan konteks negasi [31]. Selanjutnya, token yang telah diberi prefiks `Neg_` dipertahankan selama proses stemming agar informasi negasi tidak hilang pada tahap praproses berikutnya. Penelitian terdahulu menunjukkan bahwa penanganan negasi mampu meningkatkan kualitas representasi teks dan membantu model menghasilkan klasifikasi sentimen yang lebih akurat dibandingkan tanpa penanganan negasi [32].

2.4.5 Tokenisasi

Tokenisasi adalah proses pemisahan teks menjadi unit-unit kata individual yang disebut *token*. Pada teks berbahasa Indonesia, proses ini umumnya cukup dilakukan dengan pemisahan berdasarkan spasi (*whitespace tokenization*) setelah tahapan *cleaning* dan normalisasi selesai [28].

2.4.6 Penghapusan Stopword

Stopword removal adalah tahapan penghapusan kata-kata yang frekuensinya sangat tinggi namun kontribusi semantiknya terhadap makna kalimat sangat rendah. Contoh *stopword* Bahasa Indonesia antara lain "yang", "di", "dan", "ke", "dari", "untuk", dan "dengan". Penghapusan *stopword* dilakukan menggunakan daftar gabungan dari pustaka PySastrawi dan NLTK, diperkaya dengan kamus *stopword* kustom yang disesuaikan dengan konteks domain ulasan aplikasi [27, 26]. Tahapan ini secara signifikan mengurangi dimensionalitas ruang fitur dan meningkatkan kualitas representasi TF-IDF.

2.4.7 Stemming

Stemming adalah proses pereduksian kata berimbuhan menjadi bentuk dasarnya (*stem*). Pada Bahasa Indonesia, proses ini sangat penting mengingat

kekayaan sistem afiksasi yang dapat menghasilkan puluhan variasi morfologis dari satu kata dasar (misalnya “membaca”, “dibaca”, “bacaan”, “pembaca”, dan “keterbacaan” semuanya bermuara pada kata dasar “baca”). *Stemming* dilakukan menggunakan pustaka PySastrawi yang menerapkan algoritma *Enhanced Confix Stripping* (ECS) yang telah terbukti efektif untuk teks berbahasa Indonesia [27].

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

TF-IDF adalah metode ekstraksi fitur teks yang mengubah dokumen teks menjadi vektor numerik berdasarkan bobot kepentingan setiap kata dalam konteks korpus [13]. Metode ini terdiri atas dua komponen utama.

2.5.1 Term Frequency (TF)

Komponen pertama adalah *Term Frequency* (TF), yang mengukur seberapa sering suatu kata t muncul dalam dokumen d. Semakin tinggi frekuensi kemunculan sebuah kata dalam suatu dokumen, semakin besar nilai TF-nya, yang mengindikasikan bahwa kata tersebut berperan penting dalam dokumen tersebut. Formulasinya ditunjukkan pada Rumus 2.1.

$$\mathrm{TF}(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{2.1}$$

di mana:

- $f_{t,d}$ adalah frekuensi kemunculan kata t dalam dokumen d,
- $\sum_{t' \in d} f_{t',d}$ adalah jumlah total seluruh kata dalam dokumen d (digunakan sebagai normalisasi agar nilai TF tidak bergantung pada panjang dokumen).

Nilai TF yang dinormalisasi berada pada rentang $[0,1]$. Nilai 0 menunjukkan bahwa kata t tidak muncul sama sekali dalam dokumen d, sementara nilai mendekati 1 menunjukkan bahwa hampir seluruh kata dalam dokumen adalah kata t tersebut.

2.5.2 Inverse Document Frequency (IDF)

Komponen kedua adalah *Inverse Document Frequency* (IDF), yang mengukur seberapa jarang atau unik suatu kata t muncul di seluruh korpus D. Kata-

kata yang muncul di hampir semua dokumen (seperti kata sambung atau preposisi) memiliki nilai IDF rendah karena dianggap kurang informatif. Sebaliknya, kata-kata yang hanya muncul di sebagian kecil dokumen memiliki nilai IDF tinggi karena lebih bersifat diskriminatif dan informatif. Formulasinya ditunjukkan pada Rumus 2.2.

$$\text{IDF}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{2.2}$$

di mana:

- $|D|$ adalah jumlah total dokumen dalam korpus,
- $|\{d \in D : t \in d\}|$ adalah jumlah dokumen yang mengandung kata t paling tidak satu kali (disebut juga *document frequency*).

Penggunaan fungsi logaritma bertujuan meredam perbedaan skala yang terlalu besar antara kata yang sangat jarang dan kata yang sangat sering muncul di seluruh korpus.

2.5.3 Bobot TF-IDF

Bobot TF-IDF diperoleh dari perkalian kedua komponen tersebut, sebagaimana ditunjukkan pada Rumus 2.3.

$$\text{TF-IDF}(t,d,D) = \text{TF}(t,d) \times \text{IDF}(t,D) \tag{2.3}$$

Dengan demikian, suatu kata mendapatkan bobot tinggi apabila sering muncul dalam satu dokumen tertentu (TF tinggi) namun jarang muncul di dokumen lain dalam korpus (IDF tinggi). Sebaliknya, kata yang umum di seluruh korpus akan mendapatkan bobot rendah meskipun sering muncul dalam sebuah dokumen.

Dalam penelitian ini digunakan konfigurasi *sublinear TF scaling*, yakni menerapkan transformasi $\text{TF}_{\text{sub}}(t,d) = 1 + \log(f_{t,d})$ sebagai pengganti nilai TF mentah. Transformasi ini meredam pengaruh kata yang sangat sering muncul sehingga perbedaan antara frekuensi kemunculan 100 dan 1000 kali tidak dianggap sebesar perbedaan antara 1 dan 10 kali. Selain itu, digunakan konfigurasi *n-gram* $(1,2)$ untuk menangkap konteks frasa dua kata (*bigram*) yang sering bermakna berbeda dari kata-kata penyusunnya secara individual [13].

2.6 Support Vector Machine (SVM)

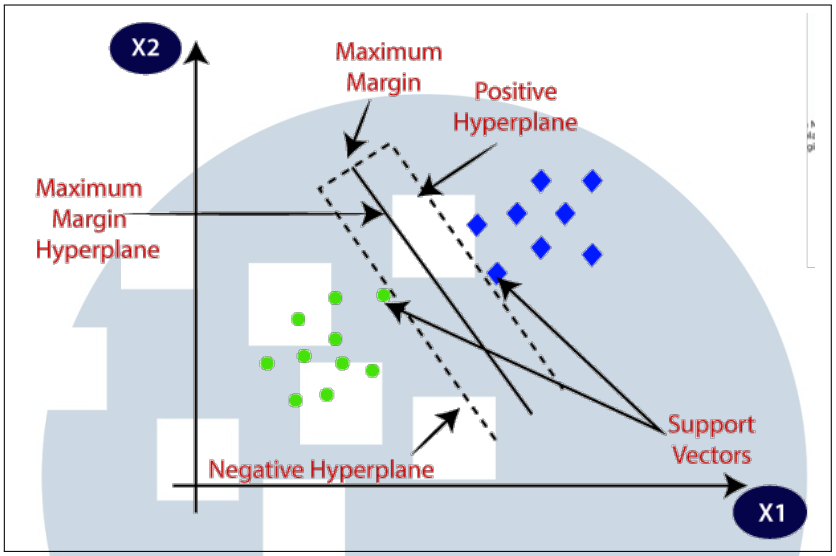
Support Vector Machine (SVM) merupakan algoritma klasifikasi yang menggunakan konsep *hyperplane*, yaitu garis atau bidang yang memisahkan data dengan label yang berbeda. Algoritma ini berfokus untuk membuat *hyperplane* yang paling optimal untuk mengkategorikan data *training* yang sudah diberi label (*supervised learning*) [14]. Secara umum, vektor direpresentasikan sebagai urutan elemen-elemen yang mewakili titik dalam ruang berdimensi-*n* yang memiliki arah. Misalnya, dalam ruang dua dimensi, sebuah vektor dapat direpresentasikan sebagai $[x,y]$ di mana x dan y membentuk sebuah titik koordinat. Dalam ruang tiga dimensi, vektor direpresentasikan sebagai $[x,y,z]$ dan seterusnya [11]. Sementara itu, *support vectors* merupakan titik koordinat yang terletak paling dekat dengan *hyperplane*, dan sangat penting karena menentukan posisi dan orientasi *hyperplane* [14]. Persamaan *hyperplane* dapat dituliskan ke dalam rumus (2.4), sedangkan persamaan *margin* dapat dilihat pada rumus (2.5).

$$|\mathbf{W} \cdot \mathbf{X} + b| = 1 \tag{2.4}$$

$$M = \frac{2}{\|\mathbf{W}\|} \tag{2.5}$$

Dalam konteks ini, $\mathbf{W}$ merupakan vektor bobot yang tegak lurus terhadap *hyperplane*, sedangkan $\frac{2}{\|\mathbf{W}\|}$ mengindikasikan *margin* yang merupakan jarak *hyperplane* dari titik asal sepanjang vektor $\mathbf{W}$. Dataset pelatihan $(x_1,y_1),\ldots,(x_n,y_n)$ mewakili n titik yang digunakan dalam pelatihan model. Di dalam penelitian ini, x_n mencerminkan kalimat atau aspek yang telah divektorisasi, sedangkan y_n mencerminkan label sentimen dari kalimat tersebut.

SVM telah banyak digunakan dan terbukti efektif untuk analisis sentimen dan *text mining* [14, 11]. SVM memiliki fungsi untuk memecahkan masalah klasifikasi. Metode ini menggunakan *kernel* untuk mengubah data ke dalam ruang dimensi yang lebih tinggi agar dapat mengklasifikasikan data yang tidak dapat dipisahkan secara linear. *Kernel* membutuhkan pengaturan parameter yang dapat disesuaikan, yang biasa disebut sebagai *hyperparameter*. Ada beberapa jenis fungsi *kernel* yang umum digunakan, seperti *kernel linear*, *kernel sigmoid*, *kernel* polinomial, dan *kernel Radial Basis Function* (RBF).



Gambar 2.1. Cara kerja *Support Vector Machine*.

Berikut adalah persamaan matematis untuk *kernel linear* yang dapat dilihat pada persamaan (2.6).

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j \tag{2.6}$$

Berikut adalah persamaan matematis untuk *kernel* RBF yang dapat dilihat pada persamaan (2.7).

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right) \tag{2.7}$$

Berikut adalah persamaan matematis untuk *kernel* polinomial yang dapat dilihat pada persamaan (2.8).

$$K(\mathbf{x}_i, \mathbf{x}_j) = \left(\gamma \mathbf{x}_i^\top \mathbf{x}_j + r\right)^d \tag{2.8}$$

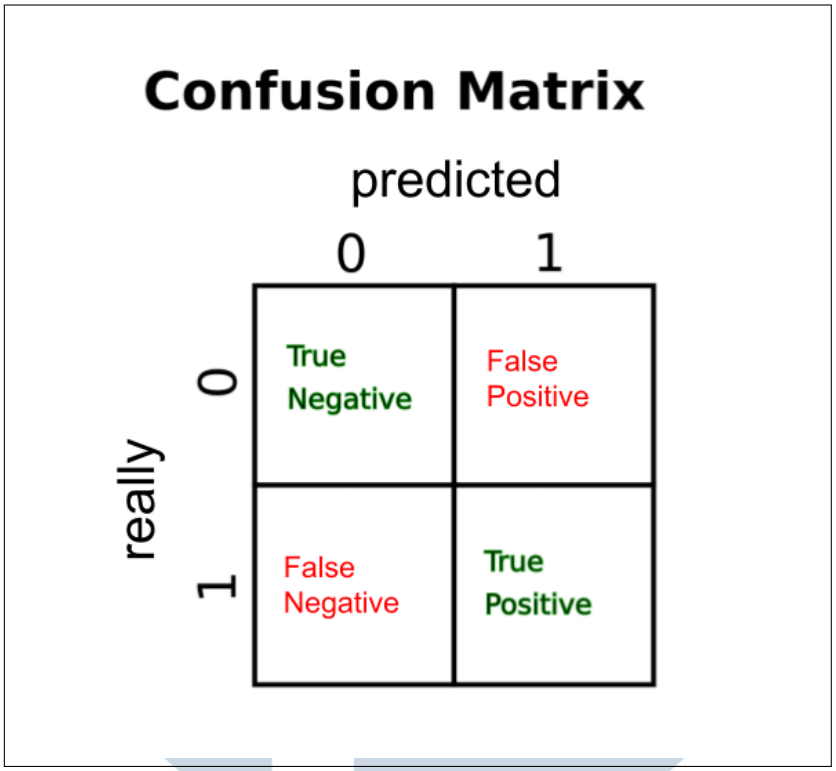
Dalam penelitian ini digunakan *linear kernel*, *RBF kernel*, *polynomial kernel* yang kemudian akan dibandingkan performa diantara semua kernel tersebut.

2.7 Metrik Evaluasi Model Klasifikasi

Evaluasi performa model klasifikasi sentimen dilakukan menggunakan beberapa metrik standar yang mengukur aspek berbeda dari kualitas prediksi.

2.7.1 Confusion Matrix

Confusion matrix adalah tabel yang merangkum hasil klasifikasi berdasarkan perbandingan antara label prediksi dan label aktual [33]. Untuk klasifikasi biner, *confusion matrix* memuat empat nilai: (1) *True Positive* (TP): sampel positif yang diprediksi benar sebagai positif; (2) *True Negative* (TN): sampel negatif yang diprediksi benar sebagai negatif; (3) *False Positive* (FP): sampel negatif yang salah diprediksi sebagai positif (*Type I Error*); dan (4) *False Negative* (FN): sampel positif yang salah diprediksi sebagai negatif (*Type II Error*). Keempat nilai dasar pada *confusion matrix* tersebut selanjutnya tidak hanya sekadar merepresentasikan matriks kesalahan prediksi, melainkan menjadi parameter utama untuk menghitung metrik evaluasi kinerja model diantaranya adalah *accuracy*, *recall*, *precision*, dan *f1-score*[34]. Gambaran dari confusion matrix bisa dilihat pada Gambar 2.2



Gambar 2.2. Confusion Matrix

2.7.2 Akurasi

Akurasi adalah proporsi prediksi benar dari seluruh prediksi yang dibuat, sebagaimana ditunjukkan pada Rumus 2.9.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.9}$$

Akurasi memberikan gambaran umum performa model, namun menjadi tidak informatif pada kondisi dataset yang tidak seimbang karena model yang hanya memprediksi kelas mayoritas pun dapat memiliki akurasi tinggi.

2.7.3 Presisi

Presisi mengukur proporsi prediksi positif yang benar-benar positif. Metrik ini penting ketika biaya *False Positive* tinggi, sebagaimana ditunjukkan pada Rumus 2.10.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.10}$$

2.7.4 Recall

Recall (juga disebut *sensitivity* atau *true positive rate*) mengukur proporsi instans positif aktual yang berhasil diidentifikasi oleh model. Metrik ini penting ketika biaya *False Negative* tinggi, sebagaimana ditunjukkan pada Rumus 2.11.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.11}$$

2.7.5 F1-Score

F1-*score* adalah rata-rata harmonik antara presisi dan *recall*. Rata-rata harmonik dipilih karena memberikan bobot lebih besar pada nilai yang lebih kecil, sehingga F1-*score* yang tinggi hanya dapat dicapai apabila *both* presisi *and recall* sama-sama tinggi, sebagaimana ditunjukkan pada Rumus 2.12.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.12}$$

Sebagai contoh, model dengan presisi 0,9 dan *recall* 0,1 menghasilkan F1-*score* $\approx 0{,}18$, jauh lebih rendah dari rata-rata aritmetika 0,5 sehingga mencerminkan ketidakseimbangan performa model secara lebih akurat.

2.8 SHapley Additive Explanations (SHAP)

SHAP adalah kerangka kerja interpretabilitas model berbasis teori permainan kooperatif Shapley yang memberikan penjelasan konsisten dan lokal atas prediksi model *machine learning* [10]. Nilai SHAP untuk setiap fitur mengukur kontribusi marginalnya terhadap prediksi, dengan mempertimbangkan seluruh kemungkinan urutan kemunculan fitur dalam koalisi.

2.8.1 Nilai Shapley

Ide dasar nilai Shapley berasal dari teori permainan kooperatif: setiap pemain (fitur) mendapatkan "pembayaran" (kontribusi) yang adil berdasarkan rata-rata kontribusi marginalnya di seluruh kemungkinan koalisi (*subset*) pemain lainnya. Nilai Shapley untuk fitur *i* didefinisikan pada Rumus 2.13.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{i\}}(\mathbf{x}) - f_S(\mathbf{x}) \right] \tag{2.13}$$

di mana:

- F adalah himpunan semua fitur dengan $|F|$ sebagai jumlah totalnya,
- S adalah subset fitur yang tidak menyertakan fitur i ($S \subseteq F \setminus \{i\}$),
- $\frac{|S|!\,(|F|-|S|-1)!}{|F|!}$ adalah faktor pembobotan yang merepresentasikan probabilitas koalisi S muncul dalam suatu urutan acak dari semua $|F|$ fitur—faktor ini memastikan setiap urutan kemunculan fitur mendapat bobot yang sama,
- $f_{S \cup \{i\}}(\mathbf{x})$ adalah nilai prediksi model ketika fitur dalam S *dan* fitur i tersedia,
- $f_S(\mathbf{x})$ adalah nilai prediksi model ketika hanya fitur dalam S yang tersedia (fitur i tidak digunakan),
- $f_{S \cup \{i\}}(\mathbf{x}) - f_S(\mathbf{x})$ adalah *kontribusi marginal* fitur i ketika ditambahkan ke koalisi S.

Secara intuitif, ϕ_i adalah rata-rata tertimbang dari semua kontribusi marginal fitur i ketika ia "bergabung" dengan berbagai koalisi yang mungkin terbentuk dari fitur-fitur lainnya.

Nilai SHAP positif ($\phi_i > 0$) menunjukkan bahwa fitur i mendorong prediksi ke arah kelas positif (meningkatkan probabilitas prediksi di atas nilai dasar *base value*), sedangkan nilai SHAP negatif ($\phi_i < 0$) menunjukkan dorongan ke arah kelas negatif.

2.8.2 KernelExplainer

`KernelExplainer` digunakan untuk menghitung nilai SHAP pada model SVM dengan berbagai jenis *kernel*, seperti *linear*, *polynomial*, dan *Radial*

Basis Function (RBF). `KernelExplainer` bersifat *model-agnostic* sehingga dapat digunakan secara konsisten pada semua varian kernel tanpa bergantung pada asumsi linearitas model [24].

Meskipun tersedia beberapa varian SHAP yang dirancang untuk jenis model tertentu, penelitian ini menggunakan `KernelExplainer` karena bersifat *model-agnostic* dan dapat diterapkan pada berbagai jenis model pembelajaran mesin. Pemilihan `KernelExplainer` didasarkan pada fakta bahwa penelitian ini menguji beberapa model SVM dengan *kernel* yang berbeda, yaitu *linear*, *radial basis function* (RBF), dan *polynomial*. Dengan demikian, penggunaan `KernelExplainer` memungkinkan penerapan metode interpretasi yang konsisten pada seluruh model yang diuji tanpa bergantung pada karakteristik khusus dari masing-masing *kernel* [24].

2.8.3 Visualisasi SHAP

Empat jenis visualisasi SHAP digunakan dalam penelitian ini untuk menganalisis interpretabilitas model dari perspektif yang berbeda.

Pertama, *beeswarm plot* menampilkan distribusi nilai SHAP untuk setiap fitur di seluruh sampel dataset, dengan setiap titik mewakili satu sampel. Warna titik menunjukkan nilai fitur (merah untuk nilai tinggi, biru untuk nilai rendah), sehingga memungkinkan analisis arah dan besar kontribusi setiap fitur secara global.

Kedua, *bar plot* menampilkan rata-rata nilai absolut SHAP ($\overline{|\phi_i|}$) sebagai metrik kepentingan fitur global. Metrik ini dirumuskan sebagai:

$$\bar{\phi}_i = \frac{1}{N} \sum_{n=1}^{N} |\phi_i^{(n)}| \tag{2.14}$$

di mana $\phi_i^{(n)}$ adalah nilai SHAP fitur i untuk sampel ke-n dan N adalah jumlah total sampel. Metrik ini dapat dibandingkan langsung antar model.

Ketiga, *waterfall plot* menampilkan kontribusi kumulatif setiap fitur terhadap prediksi satu instans spesifik, dari nilai dasar (*base value* $= \mathbb{E}[f(\mathbf{x})]$) hingga nilai prediksi akhir $f(\mathbf{x})$.

2.9 Google Play Store sebagai Sumber Data Ulasan

Google Play Store adalah platform distribusi aplikasi Android milik Google yang menyediakan fitur ulasan dan penilaian pengguna secara publik.

Ulasan di Google Play Store memiliki beberapa karakteristik yang menjadikannya sumber data berharga untuk analisis sentimen: bersifat terbuka dan dapat diakses secara publik, berskala besar dengan pembaruan berkelanjutan, representatif terhadap pengalaman nyata pengguna, serta mengandung *rating* bintang yang dapat digunakan sebagai proksi label awal [8].

Rating dan ulasan di Google Play Store secara signifikan memengaruhi keputusan unduh pengguna lain, sehingga memantau dan memahami sentimen ulasan menjadi aspek penting dalam manajemen aplikasi mobile [8]. Pengumpulan data dalam penelitian ini dilakukan menggunakan pustaka `google-play-scraper` berbasis Python yang memungkinkan pengambilan ulasan secara terprogram dengan filter berdasarkan *rating* bintang, bahasa, dan negara.

2.10 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai dasar untuk mengetahui penerapan *Support Vector Machine* (SVM) dalam analisis sentimen serta penggunaan *SHapley Additive exPlanations* (SHAP) dalam menginterpretasikan hasil prediksi model. Ringkasan penelitian terdahulu yang relevan dengan penelitian ini disajikan pada Tabel 2.1.

Tabel 2.1. Ringkasan penelitian terdahulu

No.	Judul Penelitian	Peneliti (Tahun)	Metode	Hasil Terbaik atau Temuan	Link/DOI
1	*Sentiment Analysis on Customer Satisfaction of Digital Banking in Indonesia*	Andrian, Simanungkalit, Budi, dan Wicaksono (2022)	Klasifikasi sentimen menggunakan beberapa algoritma *machine learning*, termasuk SVM	SVM menghasilkan *F1-score* terbaik sebesar 73,34%	10.14569/IJACSA.2022.0130356
2	Analisis Sentimen Ulasan Google Play Store: Studi Komparatif Algoritma SVM, Naïve Bayes, dan Logistic Regression	Jatmiko dan Dometian (2025)	Studi komparatif SVM, *Naïve Bayes*, dan *Logistic Regression* dengan berbagai konfigurasi TF-IDF	SVM memperoleh akurasi 93%, mengungguli *Naïve Bayes* dan *Logistic Regression* (masing-masing 92%)	10.37859/jf.v15i3.10016
3	Analisis Sentimen terhadap Aplikasi Pluang Menggunakan Algoritma Naïve Bayes dan SVM	Maulana, Fahmi, Imran, dan Hidayati (2024)	Perbandingan performa *Naïve Bayes* dan SVM pada ulasan aplikasi Pluang	SVM: akurasi 99,5%; *Naïve Bayes*: akurasi 99,25%	journal.irpi.or.id
4	Sentiment Analysis Meets Explainable Artificial Intelligence	Diwali, Dashtipour, Gogate, dan Hussain (2024)	Tinjauan penerapan metode *Explainable AI* (SHAP dan LIME) pada analisis sentimen	SHAP dan LIME meningkatkan interpretabilitas model yang bersifat *black-box*	10.1109/TAFFC.2023.3296157
5	Optimizing Sentiment Analysis for Usability Testing: Enhancing SVM Accuracy Through Kernel Selection and Tuning Methods	Basri, Kusyadi, dan Putri (2024)	Komparasi kernel SVM (Linear, RBF, Sigmoid, Polynomial) dioptimasi dengan *Grid Search*	Kernel Linear terbaik: akurasi 70,50%, *F1-score* 0,8618	ResearchGate
6	Shapley Additive Explanations for Text Classification and Sentiment Analysis of Internet Movie Database	Zhao, Joshi, Singh, Sudjiano, dan Nair (2022)	Penerapan SHAP untuk menginterpretasikan model klasifikasi teks pada ulasan film IMDB	SHAP menjelaskan kontribusi tiap kata terhadap prediksi sentimen ulasan film	ResearchGate

Berdasarkan penelitian terdahulu tersebut, SVM menunjukkan performa yang kompetitif dalam klasifikasi sentimen dan dalam beberapa penelitian mampu mengungguli algoritma klasifikasi lainnya. Selain itu, pemilihan kernel dan proses optimasi *hyperparameter* memiliki pengaruh terhadap performa model SVM. Penelitian mengenai SHAP juga menunjukkan bahwa metode tersebut dapat digunakan untuk mengidentifikasi kontribusi setiap fitur atau kata terhadap hasil prediksi model.

Perbedaan penelitian ini dengan penelitian terdahulu terletak pada objek, metode pengujian, dan pendekatan interpretabilitas yang digunakan. Penelitian ini menerapkan SVM dengan kernel Linear, RBF, dan Polynomial untuk menganalisis sentimen ulasan aplikasi Kompas.com pada Google Play Store. Optimasi *hyperparameter* dilakukan menggunakan *GridSearchCV* dengan *Stratified K-Fold Cross-Validation*, sedangkan SHAP digunakan untuk menjelaskan kontribusi kata terhadap prediksi sentimen yang dihasilkan oleh model.