BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Sentimen merupakan analisis terhadap opini, sikap, dan emosi terkait topik atau produk tertentu, yang diklasifikasikan berdasarkan polaritasnya, apakah positif, negatif, atau netral [15]. Analisis sentimen digunakan di berbagai domain, seperti perawatan kesehatan, pemantauan merek, hiburan, dan prediksi pasar saham, untuk membantu organisasi memahami sentimen pelanggan, meningkatkan layanan, dan membuat keputusan berbasis data [16].

2.2 Text Mining

Text mining adalah proses yang memerlukan banyak pengetahuan untuk mengubah data tekstual tidak terstruktur menjadi data terstruktur guna memperoleh informasi baru melalui dua fase: *Preprocessing*, yang mencakup normalisasi ejaan, *case folding*, tokenisasi dan *stemming*, serta pemilihan fitur, yang mengurangi dimensi kata dengan memilih istilah relevan yang mewakili isi dokumen, dengan kategorisasi sebagai salah satu alat yang digunakan selama analisis [17]. Beberapa *text mining* digunakan untuk klasifikasi dokumen (klasifikasi teks, standarisasi dokumen), pengambilan informasi (pencarian kata kunci, kueri, dan pengindeksan), pengelompokan dokumen (pengelompokan frasa), pemrosesan bahasa alami (koreksi ejaan, lemmatisasi, parsing gramatikal, dan variasi makna kata), dan analisis tautan web [18].

2.3 Text Preparation

Text Prepration merupakan proses menghilangkan data yang tidak diinginkan, seperti tag *HTML*, *hyperlink*, tanda baca, emotikon, dan angka yang tidak berkaitan dengan opini seseorang terhadap suatu hal [19]. Tujuannya adalah untuk mempersiapkan dan mengatur data teks agar lebih terstruktur sebelum dilakukan analisis lebih lanjut [20]. Pada tahap *case folding*, teks diubah menjadi huruf kecil [21], sehingga kalimat ”Waduh pertama kali gw denger Indodax kena exploit.” menjadi ”waduh pertama kali gw denger indodax kena exploit.” Setelah itu, dilakukan normalisasi untuk mengganti kata tidak baku menjadi kata baku

[15], misalnya kata ”gw” dikonversi menjadi ”saya”. Kemudian, tahap *tokenizing* memecah kalimat menjadi unit-unit kata [22], menghasilkan daftar token seperti [”waduh”, ”pertama”, ”kali”, ”saya”, ”denger”, ”indodax”, ”kena”, ”exploit”]. Selanjutnya dilakukan *stopword removal*, yaitu penghapusan kata-kata yang tidak memiliki makna signifikan dalam analisis [23], seperti ”waduh”, ”pertama”, dan ”kali”. Hasilnya menjadi [”saya”, ”denger”, ”indodax”, ”kena”, ”exploit”]. Terakhir, pada tahap *stemming*, kata-kata dikembalikan ke bentuk dasarnya [24], contohnya ”denger” menjadi ”dengar”, sehingga hasil akhir dari seluruh proses adalah [”saya”, ”dengar”, ”indodax”, ”kena”, ”exploit”].

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

Term Frequency-Inverse Document Frequency (TF-IDF) merupakan metode untuk memberikan bobot pada kata-kata yang telah melalui *preprocessing* [25]. Cara kerja TF-IDF terdiri dari dua komponen utama, yaitu *Term Frequency (TF)* dan *Inverse Document Frequency (IDF)*. *Term Frequency* menghitung seberapa sering sebuah fitur muncul dalam sebuah dokumen dibandingkan dengan total fitur yang muncul dalam dokumen tersebut, memberikan bobot lebih besar pada fitur tertentu yang sering muncul [26]. Sementara itu, *Inverse Document Frequency* memberikan bobot pada fitur yang jarang muncul di seluruh koleksi dokumen, membantu dalam menentukan fitur yang lebih penting untuk klasifikasi [26].

$$TF_t = (t, d) \tag{2.1}$$

$$IDF_t = \log \frac{n}{df(t)} + 1 \tag{2.2}$$

$$W_t = TF_t \times IDF_t \tag{2.3}$$

Persamaan (2.1) menunjukkan *Term Frequency (TF)*, yaitu perhitungan seberapa sering suatu kata t muncul dalam dokumen d, yang memberikan bobot lebih besar pada kata yang sering muncul dalam dokumen tertentu. Selanjutnya, *Inverse Document Frequency (IDF)* pada persamaan (2.2) digunakan untuk mengukur kepentingan kata t dengan memperhitungkan seberapa jarang kata

tersebut muncul dalam seluruh koleksi dokumen, yang dihitung menggunakan logaritma dari rasio jumlah total dokumen n terhadap jumlah dokumen yang mengandung kata tersebut $df(t)$, ditambah satu untuk menghindari pembagian dengan nol. Akhirnya, persamaan (2.3) menunjukkan perhitungan *TF-IDF*, di mana nilai *TF* dari persamaan (2.1) dikalikan dengan nilai *IDF* dari persamaan (2.2), menghasilkan bobot W_t yang lebih tinggi untuk kata yang sering muncul dalam dokumen tetapi jarang muncul dalam keseluruhan korpus, sehingga membantu dalam menyoroti kata-kata yang lebih signifikan dalam analisis teks.

2.5 Train Test Split

Train-Test Split merupakan metode machine learning untuk membagi dataset menjadi dua bagian yaitu data latih (Train) dan data uji (Test). Data pelatihan digunakan untuk membangun dan mengoptimalkan model, sementara data pengujian digunakan untuk mengukur kinerja model pada data yang belum pernah dilihat sebelumnya selama proses pelatihan [27]. Rasio yang umum digunakan untuk membagi dataset dalam metode *Train-Test Split* adalah 60:40, 70:30, dan 80:20. Dalam penelitian yang dilakukan oleh Daniel Febrian Sangkey dkk. berjudul *Effects of Kernels and the Proportion of Training Data on the Accuracy of SVM Sentiment Analysis in Lecturer Evaluation* [28], ditemukan bahwa semakin besar proporsi data yang digunakan untuk pelatihan, variasi akurasi yang diperoleh juga semakin besar. Hal ini menunjukkan bahwa model menjadi kurang stabil dengan peningkatan proporsi data pelatihan, yang dapat mengarah pada overfitting atau fluktuasi hasil yang tinggi pada data uji.

2.6 Support Vector Machine

Support Vector Machine (SVM) merupakan algoritma untuk mengklasifikasikan berbagai kelas dalam himpunan data pelatihan dengan menentukan sebuah *hyperplane* optimal yang memaksimalkan margin pemisahan antara kelas-kelas tersebut. Keunggulan utama SVM terletak pada kemampuannya dalam *generalization ability*, yaitu kemampuannya untuk mengklasifikasikan data baru dengan akurasi tinggi, bahkan dalam *dataset* berdimensi tinggi atau dengan jumlah sampel terbatas [29].

Hyperplane merupakan bidang pemisah yang digunakan dalam SVM untuk membagi data ke dalam dua kelas yang berbeda. Dalam ruang dua dimensi,

hyperplane berbentuk garis lurus, sedangkan dalam ruang tiga dimensi berupa bidang datar. Pada dimensi yang lebih tinggi, *hyperplane* dikenal sebagai hiperbidang yang berfungsi sebagai batas pemisah antar kelas. Secara matematis, *hyperplane* dapat dinyatakan dengan Persamaan 2.4:

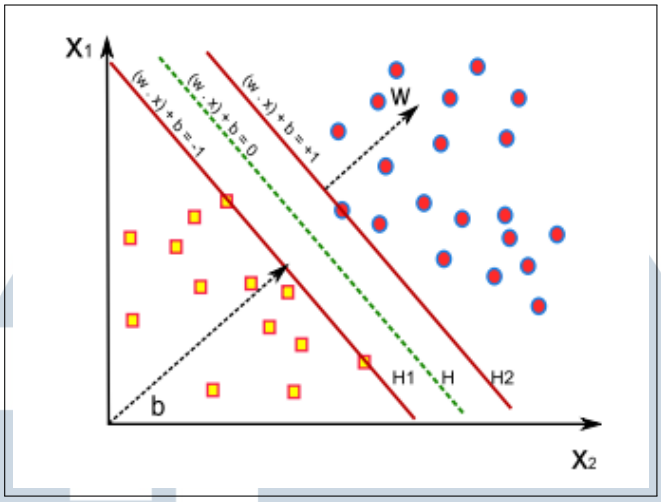
$$w \cdot x + b = 0 \tag{2.4}$$

di mana w adalah vektor bobot yang menentukan orientasi *hyperplane*, x merupakan vektor fitur dari data yang dianalisis, dan b adalah bias yang menggeser posisi *hyperplane* dalam ruang fitur.

Margin adalah jarak antara *hyperplane* dengan titik data terdekat dari setiap kelas, yang disebut sebagai *support vectors*. Semakin besar margin, semakin baik kemampuan model dalam menggeneralisasi data baru. Oleh karena itu, SVM bertujuan untuk memaksimalkan margin, yang secara matematis didefinisikan pada Persamaan 2.5:

$$\text{Margin} = \frac{2}{\|w\|} \tag{2.5}$$

Support vectors adalah titik-titik data yang paling dekat dengan *hyperplane* dan menentukan batas pemisahan antar kelas. Hanya *support vectors* yang memengaruhi posisi *hyperplane*, sedangkan titik-titik lain yang jauh dari *hyperplane* tidak berpengaruh terhadap proses klasifikasi. Ilustrasi mengenai konsep *hyperplane*, margin, serta *support vectors* ditampilkan pada Gambar 2.1.



Gambar 2.1. Ilustrasi *hyperplane*, margin, dan *support vectors* dalam SVM.

2.7 Confusion Matrix

Dalam membangun sebuah model, diperlukan sebuah pengukuran dari performa suatu model. *Confusion Matrix* merupakan indikator penilaian terhadap performa klasifikasi *machine learning*. Tabel yang mengklasifikasikan jumlah data uji yang benar dan salah juga dikenal sebagai *Confusion Matrix* [30]. Confusion matrix dalam proses klasifikasi terdiri dari empat istilah utama yang merepresentasikan hasil prediksi, yaitu *True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN)*. Contoh *confusion matrix* untuk klasifikasi ditunjukkan pada Tabel 2.1

Tabel 2.1. Confusion Matrix

Predicted	**Actual**	
	Positive	**Negative**
Positive	True Positive	False Positive
Negative	False Negative	True Negative

Keterangan :

- **True Positive (TP)**: Jumlah prediksi positif yang benar. Model memprediksi kelas positif dan kelas sebenarnya juga positif.
- **False Negative (FN)**: Jumlah prediksi negatif yang salah. Model memprediksi kelas negatif, tetapi kelas sebenarnya adalah positif.

- **False Positive (FP)**: Jumlah prediksi positif yang salah. Model memprediksi kelas positif, tetapi kelas sebenarnya adalah negatif.
- **True Negative (TN)**: Jumlah prediksi negatif yang benar. Model memprediksi kelas negatif dan kelas sebenarnya juga negatif.

Matriks ini merangkum hasil prediksi dalam masalah klasifikasi dan mengusulkan *"Attribution Confusion Matrix"* untuk metode atribusi fitur, yang mengevaluasi akurasi menggunakan metrik seperti presisi, akurasi, *recall*, dan skor F1[31]. Berikut penjelasan dari evaluasi dalam klasifikasi [32] :

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} \sum_{j=1}^{n} N_{ij}} \tag{2.6}$$

Persamaan 2.6 menunjukkan *accuracy*, yang mengukur proporsi prediksi yang benar dari keseluruhan prediksi yang dilakukan oleh model.

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i} \tag{2.7}$$

Selanjutnya, Persamaan 2.7 menggambarkan *recall*, yang mengukur sejauh mana model mampu menemukan seluruh contoh positif dalam dataset.

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \tag{2.8}$$

Persamaan 2.8 mendefinisikan *precision*, yang menunjukkan tingkat keakuratan dari prediksi positif yang dibuat oleh model. *Precision* sangat berguna dalam kasus di mana false positive perlu diminimalkan, seperti dalam diagnosis medis atau deteksi spam.

$$\text{F1-score}_i = 2 \cdot \frac{\text{Recall}_i \cdot \text{Precision}_i}{\text{Recall}_i + \text{Precision}_i} \tag{2.9}$$

Terakhir, Persamaan 2.9 merepresentasikan *F1-score*, yaitu rata-rata harmonis dari *recall* dan *precision*. Metrik ini digunakan ketika diperlukan keseimbangan antara *recall* dan *precision*, terutama dalam kasus di mana terdapat ketidakseimbangan kelas dalam dataset.