BAB II
LANDASAN TEORI

2.1 Tinjauan Pustaka

Dalam melakukan penelitian ini, beberapa penelitian terdahulu dijadikan penulis sebagai referensi dan acuan. Sebuah penelitian yang dilakukan oleh (Darwis, Pratiwi, & Pasaribu, 2020), mengkaji tentang analisis sentimen terhadap komentar di Twitter milik Komisi Pemberantasan Korupsi (KPK) dengan menggunakan algoritma *Support Vector Machine* (SVM). Data yang digunakan dalam penelitian tersebut berupa kumpulan *tweet* berbahasa Indonesia yang berasal dari akun Twitter resmi KPK RI. Proses analisis dimulai dengan mengolah opini atau komentar dari pengguna melalui metode-metode yang sesuai dengan teknik pengambilan teks atau text mining. Selanjutnya, algoritma SVM diimplementasikan untuk mengklasifikasikan sentimen dari data tersebut, dan akurasi hasil klasifikasi dievaluasi. Tujuan dari penelitian ini adalah untuk menghasilkan informasi terkait sentimen masyarakat terhadap KPK yang dapat digunakan sebagai bahan pertimbangan bagi lembaga tersebut dalam meningkatkan kinerja dan upaya pemberantasan tindak pidana korupsi di Indonesia.

Suatu penelitian yang dilakukan oleh (Nugraha & Astuti, 2023) mengkaji analisis sentimen terhadap data kuisioner evaluasi dosen oleh mahasiswa dengan memanfaatkan algoritma *Support Vector Machine* (SVM). Penelitian ini menggunakan data Evaluasi Dosen Oleh Mahasiswa (EDOM) yang terkumpul dari 2.465 dosen pengajar aktif di Program Studi Sistem Informasi, Universitas Telkom. Dengan menganalisis komentar-komentar pada data kuisioner tersebut menggunakan algoritma SVM, penelitian ini berhasil mencapai tingkat akurasi kecocokan sebesar 75% dalam mengklasifikasikan sentimen mahasiswa terhadap dosen pengampunya.

Penelitian mengenai analisis sentimen ulasan aplikasi Gojek pada Google Playstore yang dilakukan oleh (Muttaqin & Kharisudin, 2021) menggunakan algoritma SVM dan KNN. Hasil dari penelitian tersebut menunjukkan

perbandingan analisis sentimen menggunakan dua metode algoritma yang berbeda. Hasil akurasi untuk metode KNN yaitu 82,14%, sedangkan untuk metode algoritma SVM mendapatkan nilai akurasi 87,98% yang menunjukkan bahwa metode SVM bekerja lebih baik dalam melakukan analisis sentimen dibandingkan metode algoritma KNN.

2.2 Sistem Analisis Sentimen

Sistem analisis sentimen adalah sistem yang digunakan untuk menganalisis tulisan online dan mendapatkan nada emosional dari penulisnya. Dalam konteks bisnis, sistem analisis sentimen berguna untuk mengetahui emosi atau sentimen konsumen terhadap brand milik perusahaan. Ini membantu perusahaan dalam mengambil perancangan bisnis dan strategi pemasaran yang tepat demi kemajuan merek ke depannya.

Sentimen atau opini yang terdapat dalam data teks dapat dikategorikan sebagai big data karena volume data teks tersebut terus bertambah besar dan maknanya semakin beragam. Analisis sentimen memiliki kaitan erat dengan bidang *Natural Language Processing* (NLP) yang merupakan cabang ilmu dalam teks mining. NLP berfokus untuk melakukan estimasi atau penafsiran makna yang terkandung dalam sebuah teks (Amrustian, Widayat, & Wirawan, 2022).

2.3 Text Mining

Pengolahan *text mining* merupakan bidang ilmu yang berkembang dari data mining. Data yang diolah tidak hanya berupa data numerik, tetapi juga dapat berupa data teks. Pengetahuan baru kini dapat ditemukan dari kumpulan teks seperti ulasan atau bentuk teks lainnya (Yusril, Larasati, & Aini, 2020). Dalam text mining, ekstraksi teks dapat dilakukan pada dokumen tunggal maupun multiple dokumen. Proses ekstraksi teks tersebut dapat memanfaatkan berbagai tool dan algoritma yang telah ada. Hal ini membuktikan bahwa teori dan metode yang digunakan dalam text mining dapat diimplementasikan dan menghasilkan pengetahuan baru. (Pamungkas & Februariyanti, 22).

2.4 Preprocessing Data

Praproses teks merupakan tahapan awal yang penting dalam pengolahan teks, di mana dokumen teks diubah menjadi data terstruktur sesuai dengan kebutuhan agar dapat diolah lebih lanjut dalam proses text mining. Tujuan dari tahapan praproses teks dalam klasifikasi adalah untuk meningkatkan akurasi hasil klasifikasi data teks tersebut. Dengan melakukan praproses yang tepat, data teks menjadi lebih siap dan berkualitas untuk diproses pada tahap-tahap selanjutnya. Tahapan dalam praproses teks adalah sebagai berikut (Kasim & Sudarsono, 2019):

a. *Cleaning*, yaitu proses menghapus simbol sebutan (*mention*) nama pengguna.
b. *Case Folding*, merupakan proses mengubah karakter teks hurus besar menjadi huruf kecil serta menghilangkan seluruh tanda baca dan angka termasuk menghilangkan karakter spasi yang berlebih.
c. *Tokenizing*, merupakan proses memecah yang semula kalimat utuh menjadi kumpulan per kata.
d. Istilah stopwords merujuk pada kata-kata yang sangat lazim digunakan dalam suatu teks, tetapi tidak memberikan kontribusi signifikan terhadap makna atau informasi utama dari teks tersebut. Dalam proses menganalisis isi teks, stopwords cenderung diabaikan atau dihilangkan karena keberadaannya tidak membantu dalam mengungkap pola atau gagasan pokok yang terkandung di dalam teks yang dianalisis.
e. *Stemming*, yakni proses untuk mendapatkan kata dasar dengan cara menghilangkan awalan, akhiran, sisipan, dan confixes atau kombinasi dari awalan dan akhiran.

2.5 Ekstraksi Fitur

Ekstraksi fitur merupakan tahap dimana data yang telah dilakukan preprocessing diubah menjadi data numerik. Hal ini dikarenakan prinsip pada komputer yang tidak mampu mengolah data selain data numerik. Selain itu

ekstraksi fitur ini digunakan untuk menggali informasi dalam mempresentasikan kata-kata sebagai vektor. Salah satu teknik yang digunakan dalam ekstraksi fitur teks adalah pembobotan TF-IDF atau *Term Frequency-Inverse Document Frequency*. Metode ini melibatkan perhitungan frekuensi kemunculan kata pada setiap dokumen yang disebut *Term Frequency* (TF), serta nilai invers dari *Document Frequency* (DF) yang merupakan nilai kebalikan dari frekuensi dokumen yang mengandung kata tersebut. Hasil akhir dari metode TF-IDF adalah sebuah matriks yang berisi kata-kata unik beserta nilai-nilai bobot yang dihasilkan dari perhitungan TF-IDF untuk setiap kata dalam keseluruhan data. Matriks ini merepresentasikan fitur-fitur teks yang akan digunakan untuk proses selanjutnya (Amalia & Yustanti, 2021). Berikut merupakan nilai persamaan untuk menghitung TF (Baeza-Yates & Rebeiro-Neto, 2011):

$$TF(Term\ Frequency) = \frac{jumlah\ kemunculan\ kata\ dalam\ dokumen}{jumlah\ total\ kata\ dalam\ dokumen}$$

Sedangkan DF atau *Document Frequency* merupakan banyaknya dokumen yang dimiliki oleh term *t*. Berikut merupakan nilai persamaan untuk menghitung IDF :

$$IDF_{(t)} = log\left(\frac{N}{df}\right) + 1 \qquad (2.1)$$

Penambahan nilai "1" pada persamaan IDF di atas, bertujuan ntuk menghindari nilai IDF yang sangat rendah atau nol, dimana term yang muncul di hampir semua dokumen dalam koleksi set pelatihan tidak akan diabaikan seluruhnya (Manning, Raghavan, & Schütze, 2009).

Keterangan :

N = jumlah dokumen

df = jumlah dokumen yang mengandung kata t.

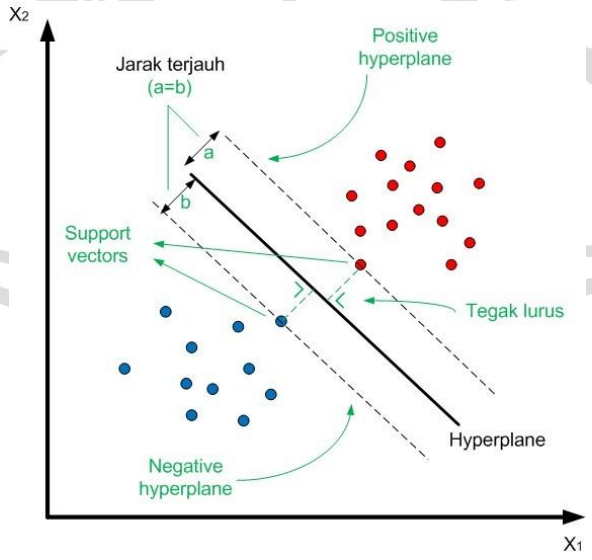
Sehingga untuk menghitung TF-IDF yaitu:

$$TF\text{-}IDF = TF \times IDF \qquad (2.2)$$

2.6 Algoritma Support Vector Machine (SVM)

Support Vector Machine (SVM) adalah sebuah metode yang digunakan untuk melakukan klasifikasi dengan memanfaatkan teknik pembelajaran mesin (*machine learning*). Metode ini bekerja dengan membangun model atau pola berdasarkan data yang telah melalui proses pelatihan (*training*) (Novantirani, Manuaba, Dantes, & Indrawan, 2022). Model atau pola yang dihasilkan kemudian digunakan untuk memprediksi kelas atau kategori dari data baru yang belum diketahui kelasnya.

Sistem kerja SVM yaitu dengan mencari hyperplane atau garis pemisah (*decision boundary*) optimal yang dapat memisahkan suatu kelas data dari kelas lainnya. yang berfungsi memisahkan tweet bersentimen positif dengan komentar bersentimen negatif (Pradana, Slamet, & Zukhronah, 2022). SVM berusaha menemukan *hyperplane* terbaik dengan memanfaatkan konsep *support vector* dan *margin*. *Support vector* merujuk pada data dari masing-masing kelas yang terletak terdekat dengan hyperplane. Sementara margin adalah jarak antara *support vector* dengan *hyperplane* itu sendiri. SVM mencari hyperplane yang dapat memaksimalkan jarak atau margin dari *support vector*, sehingga diperoleh garis pemisah terbaik untuk mengklasifikasikan data ke dalam kelasnya masing-masing. (Novantirani, Manuaba, Dantes, & Indrawan, 2022).



Gambar 2.1. Model Support Vector Machine

Konsep dasar *Support Vector Machine* (SVM) dapat dipahami sebagai upaya untuk mencari hyperplane atau bidang pembatas terbaik yang berfungsi sebagai pemisah antara dua kelas data. Gambar 1 memperlihatkan pola-pola yang merupakan anggota dari dua kelas, yaitu kelas positif digambarkan dengan lingkaran merah dan kelas negatif digambarkan dengan lingkaran biru. Permasalahan klasifikasi dapat dijelaskan dengan usaha menemukan *hyperplane* yang dapat memisahkan kedua kelompok tersebut. *Hyperplane* pemisah terbaik antara kedua kelas ditemukan dengan cara mengukur *margin hyperplane* dan mencari nilai maksimumnya. Margin adalah jarak antara *hyperplane* dengan pola terdekat dari setiap kelas, di mana pola terdekat ini disebut sebagai *support vector*. Garis solid pada Gambar 1 menunjukkan hyperplane terbaik, yaitu yang terletak tepat di tengah-tengah kedua kelas, sedangkan 2 garis titik-titik yang berdekatan dengan garis solid tersebut merupakan *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM (Kasim & Sudarsono, 2019).

Pada ruang cartesius 2 dimensi, persamaan garis $ax + by + c = 0$ sudah sangat familiar. Untuk membuat persamaan yang lebih umum dari persamaan tersebut yang dapat mencakup ruang berdimensi banyak, pertama-tama harus mengubah notasi variabel dan konstanta dari persamaan garis tersebut. x menjadi x_1, y menjadi x_2, a menjadi w, b menjadi w_2, dan c menjadi b atau biasa disebut bias. Sehingga persamaannya menjadi:

$$f(x) = w_1x_1 + w_2x_2 + b \tag{2.3}$$

jika data yang diperoleh berdimensi $k > 1$, maka persamaan 2.4 & 2.5 menjadi:

$$f(x) = w_1x_1 + \cdots + w_kx_k + b \tag{2.4}$$

Atau

$$\sum_{k=1}^{k} w_k x_k + b = 0 \tag{2.5}$$

Decision rule dari algoritma SVM didefinisikan sebagai berikut:

$$f(x) = \begin{cases} +1, & jika \geq 1 \\ -1, & jika \leq -1 \end{cases}$$

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara

hyperplane dan titik terdekatnya. Fungsi untuk mencari nilai *Lagrange Multiplier* α sebagai berikut (Nugroho, Witarto, & Handoko, 2003):

$$max_{\alpha} L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (2.6)$$

Dengan syarat: $\alpha_i \geq 0$ (i = 1,2, … , n) dan $\sum_{i=1}^{n} \alpha_i y_i = 0$

Data yang tersedia direpresentasikan dengan simbol $x_i \in \Re_r$ di mana setiap data memiliki label yang dinotasikan dengan $y_i \in \{ -1,+1 \}$ untuk $i = 1,2,\ldots,l$. l merupakan jumlah total data yang tersedia. Setelah mendapatkan nilai dari fungsi yang disebutkan sebelumnya, langkah selanjutnya adalah mencari nilai *w*, nilai bias, dan mencari fungsi keputusan klasifikasi *sign (f(x))* dengan persamaan atau formula sebagai berikut :

a. Persamaan untuk mencari nilai w sebagai berikut :

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i \quad (2.7)$$

b. Persamaan untuk mencari nilai bias sebagai berikut :

$$b = -\frac{1}{2} (xi^{+} . w + xi^{-} . w) \quad (2.8)$$

c. Persamaan untuk mencari fungsi keputusan klasifikasi sign(f(x)) sebagai berikut :

$$f(x) = sign[\, \vec{w} \cdot \vec{x} + b] \quad (2.9)$$

Fungsi `sign()` adalah sebuah fungsi normalisasi yang menghasilkan nilai 1 (kelas positif) ketika nilai x lebih besar dari 0, dan nilai -1 (kelas negatif) ketika nilai x kurang dari atau sama dengan 0 (Nugroho, Witarto, & Handoko, 2003).

d. Sedangkan fungsi untuk Linear Kernel:

$$K(x_i, x_j) = x_i^T x_j \quad (2.10)$$

Dengan x_i merupakan data latih dan x_j adalah data uji.

Keterangan :

$w =$ parameter *hyperplane* yang dicari (garis yang tegak lurus antara garis *hyperplane* dan titik *support vector*)

$y =$ label masing-masing dinotasikan $y_i \in \{ -1,+1 \}$

$x =$ titik data masukan masukan *Support Vector Machine*

$b =$ parameter nilai bias

$\alpha =$ Lagrange multipliers