BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen merupakan pembelajaran komputasi tentang perilaku komentar dan emosi orang terhadap entitas [7]. Analisis sentimen dilakukan untuk mengetahui opini publik pada komentar di media sosial tentang perilaku yang bersifat positif dan negatif.

2.2 Multinomial Naive Bayes

Multinomial Naïve Bayes ialah variasi model dari algoritma *Naïve Bayes* yang teruji baik dalam permasalahan klasifikasi teks. *Multinomial Naïve Bayes* digunakan karena algoritma ini telah teruji dapat melakukan komputasi dengan performa yang baik dan cepat dalam mengklasifikasi dokumen teks dan algoritma ini mampu menyelesaikan permasalahan yang *multiple class*[9]. Berikut merupakan rumus dari *bayes*.

$$P(w_i|x) = \frac{P(x|w_i).P(w_i)}{P(x)} \tag{2.1}$$

1. $P(x|w_i)$ = sebuah peluang kata x muncul di kelas w.
2. $P(w_i)$ = Peluang Kata pada kelas c.
3. $P(x)$ = peluang kemunculan pada kata x.

Tahapan menggunakan algoritma *Multinomial Naïve Bayes* yaitu menjumlahkan total kemunculan kata pada setiap dokumen, berikut merupakan persamaan dari *Multinomial Naïve Bayes*[10].

$$C_{map} = argmax P(c|d) \prod 1 \leq k \leq n_d P(t_k|C) \tag{2.2}$$

1. *argmax* = Mencari nilai *posterior probability* terbesar pada suatu kelas.
2. $P(t_k|C)$ = *Conditional probability*, yaitu peluang munculnya kata k dalam kelas tertentu.
3. $P(c)$ = *Prior Probability* pada kelas c.

Rumus 2.3 merupakan persamaan untuk menghitung $P(c)$.

$$P(c) = \frac{N_c}{N} \tag{2.3}$$

1. N_c = Jumlah kelas c pada seluruh dokumen.
2. N = Jumlah seluruh dokumen.

Untuk mencari conditional probability dapat menggunakan pada rumus 2.4.

$$P(t|c) = \frac{T_{ct}}{\sum t' \in V T_{ct'}'} \tag{2.4}$$

1. T_{ct} = Frekuensi suatu kata pada kelas c dalam dokumen yang berulang.
2. $T_{ct'}$ = Jumlah seluruh kata dalam kelas c.

Selama proses klasifikasi, terkadang terjadi kasus di mana ada kata yang tidak pernah muncul dalam kelas tertentu. Hal ini mengakibatkan kelas memiliki nilai probabilitas kondisional nol, yang pada akhirnya dapat menyebabkan kesalahan sistem dalam melakukan klasifikasi terhadap kata-kata dalam sebuah dokumen[10]. Untuk mengatasi masalah ini, digunakan metode add one smoothing (Laplace Smoothing). Laplace Smoothing adalah metode yang menambahkan angka satu pada setiap perhitungan probabilitas. Dengan demikian, kata-kata yang belum pernah muncul dalam kelas akan memiliki probabilitas yang lebih besar dari nol, sehingga mengurangi kemungkinan terjadinya kesalahan dalam klasifikasi. Rumus 2.5 meruoakan rumus *laplace smoothing*.

$$P(t|c) = \frac{T_{ct} + 1}{(\sum t' \in V T_{ct'}) + B'} \tag{2.5}$$

1. B' = Jumlah keseluruhan kosakata unik pada seluruh kelas dalam dokumen.

2.3 Logistic Regression

Regresi logistik merupakan algoritma pembelajaran mesin yang populer setelah regresi linier. Secara umum, terdapat banyak kesamaan antara regresi linier dan regresi logistik. Namun, perbedaan utama terletak pada penggunaan keduanya, algoritma regresi linier digunakan untuk memprediksi atau mengestimasi nilai, sedangkan regresi logistik digunakan untuk melakukan klasifikasi[11]. Algoritma ini digunakan untuk mendeskripsikan hubungan antara variabel terikat yang

memiliki dua kategori atau lebih dengan satu atau lebih variabel bebas, berskala kategori atau kontinu[11]. Berikut adalah persamaan Logistic Regression [12].

Simple Linear:

$$y = \alpha + \beta x \tag{2.6}$$

Multiple Linear:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 +, ..., +\beta_n x_n \tag{2.7}$$

Logistic Regression:

$$g(X) = sigmoid(\alpha + \beta X) \tag{2.8}$$

$$sigmoid(x) = \left(\frac{1}{1 + exp - x}\right) \tag{2.9}$$

Keterangan rumus:

1. y = Variabel respon atau variabel akibat.
2. x = Variabel independent atau variabel faktor penyebab
3. α = Konstanta.
4. β = Koefisien regresi (kemiringan).

2.4 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE *(Synthetic Minority Oversampling Technique)* adalah sebuah metode *oversampling* yang digunakan untuk mengatasi masalah ketidakseimbangan kelas dalam dataset. Teknik ini melakukan sintesis sampel baru dari kelas minoritas dengan cara menciptakan *instance* baru berdasarkan kombinasi cembung dari *instance* tetangga. Hal ini dilakukan untuk mencapai keseimbangan dataset dengan menambahkan lebih banyak sampel pada kelas minoritas yang kurang representatif[13].

2.5 Labelling

Labelling merupakan sebuah pemrosesan yang dilakukan untuk mengasumsikan setiap data tweet termasuk kedalam sub kelas yang terdiri dari positif dan negatif. Labeling dilakukan otomatis dengan bantuan *libary* daftar kata opini berbahasa Indonesia [14].

2.6 Text Processing

Text Processing merupakan proses penting yang dilakukan sebelum melakukan klasifikasi sentimen. Proses ini bertujuan dalam memilah data teks yang tidak terstruktur menjadi terstruktur sehingga mendapatkan masukan yang baik pada pemodelan data dan analisis. Tahap ini sangat berpengaruh terutama dalam melakukan sentimen analisis untuk mendapatkan data bersih [15], berikut tahapan *text processing* yang dilakukan dalam analisis sentimen.

2.6.1 Case Folding

Case Folding adalah langkah pertama dalam pemrosesan teks yang melibatkan mengubah semua huruf dalam dokumen menjadi huruf kecil.

2.6.2 Data Cleaning

Data cleaning adalah proses membersihkan data teks dengan tujuan menghapus simbol, angka, emoji, dan tanda baca yang terdapat pada setiap kata.

2.6.3 Tokenization

Tokenization adalah proses membagi kalimat menjadi daftar kata tunggal *(token)* dengan tujuan mempermudah proses stemming dalam analisis untuk mencari kata dasar. Berikut adalah contoh pemisahan kalimat menjadi token:

Kalimat : ChatGPT kecerdasan buatan tercanggih

Output : | *ChatGPT* | | *kecerdasan* | | *buatan* | | *tercanggih* |

2.6.4 Stopword Removal

Stopword Removal adalah tahap dalam pemrosesan teks yang bertujuan untuk menghapus kata-kata yang tidak memiliki arti penting dalam sebuah kalimat teks. Contoh stopword dalam bahasa Indonesia mencakup kata-kata seperti ”yang”, ”dan”, atau ”di”.

2.6.5 Stemming

Stemming adalah proses yang menggunakan bahasa Indonesia dengan bantuan *library* Sastrawi untuk mencari kata dasar dalam sebuah kalimat teks dengan cara menghilangkan imbuhan awal atau akhir. Contohnya, kata ”membelikan” akan melalui proses *stemming* dan menjadi ”beli”, karena ”membelikan” memiliki kata dasar ”ambil” dan imbuhan ”mem” dan ”kan” dihilangkan.

2.7 Term Frequency - Inverse Document Frequency

TF-IDF (Term Frequency-Inverse Document Frequency) adalah metode yang digunakan untuk menghitung bobot kata dalam sebuah dokumen. Bobot kata ini digunakan untuk menentukan seberapa penting kata tersebut dalam dokumen tersebut. Bobot kata ini dihitung dengan mengalikan frekuensi kemunculan kata dalam dokumen (term frequency) dengan kebalikan frekuensi kemunculan kata tersebut dalam seluruh dokumen (inverse document frequency).

TF-IDF (Term Frequency-Inverse Document Frequency) adalah sebuah teknik yang digunakan untuk mengukur bobot kata dalam suatu dokumen. Bobot kata tersebut digunakan untuk menentukan tingkat kepentingan kata tersebut di dalam dokumen. Bobot kata dihitung dengan mengalikan frekuensi kemunculan kata dalam dokumen (term frequency) dengan kebalikan frekuensi kemunculan kata tersebut dalam seluruh dokumen (inverse document frequency). Dengan demikian, kata-kata yang sering muncul dalam dokumen tertentu tetapi jarang muncul dalam dokumen-dokumen lain akan memiliki bobot yang tinggi, menunjukkan tingkat keunikan dan relevansi kata tersebut terhadap dokumen tersebut[16].

1. *Term Frequency* (TF)

   Pembobotan kata dengan *Term Frequency* dilakukan dengan formula berikut.

$$tf(t,d) = \frac{n_i j}{\sum_k n_i, j} \tag{2.10}$$

Keterangan:

- $tf(t,d)$ = Frekuensi kata
- $n_i j$ = Total suatu kata yang muncul pada suatu dokumen
- $\sum_k n_i, j$ = Total seluruh kata dalam suatu dokumen

2. *Inverse Document Frequency (IDF)*
   Rumus 2.11 merupakan formula untuk menghitung IDF.

$$idf = log\frac{N}{df_j} \tag{2.11}$$

Keterangan:

- N = Total kelas

3. Menghitung TF-IDF *(*Term Frequency Inverse Document Frequency)
   Rumus 2.12 berguna untuk menjumlahkan kedua hasil TF (*term frequency*) dan IDF(*inverse document frequency*):

$$w_{ij} = tf_{ij} \times idf \tag{2.12}$$

Keterangan:

(a) w_{ij} = Bobot kata i pada kelas j
(b) tf_{ij} = Total kemunculan kata i pada kelas j
(c) df_j = Total kelas j yang berisi kata i

2.8 Matrix Confussion

Confusion Matrix adalah sebuah metode yang digunakan untuk mengevaluasi performa suatu sistem klasifikasi. Metode ini menyediakan berbagai informasi yang membandingkan hasil klasifikasi yang diberikan oleh sistem dengan hasil klasifikasi yang seharusnya.. *Confusion Matrix* merupakan metode yang penting dalam melakukan visualisasi pada *machine learning* yang

biasanya memuat dua kategori maupun lebih [17]. Pada tabel 2.1 terdapat contoh penggunaan *confussion matrix* yang membandingkan dua kelas.

Tabel 2.1. Confusion Matrix

		Kelas Sebenarnya	
		Positive	Negative
Kelas Prediksi	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

- *True Positive*, Jumlah data dengan nilai positif yang diprediksi bernilai positif.
- *True Negative*, Jumlah data dengan nilai negatif yang diprediksi bernilai negatif.
- *False Positive*, Jumlah data dengan nilai positif yang diprediksi bernilai negatif.
- *False Negative*, Jumlah data dengan nilai negatif yang diprediksi bernilai positif.

Berikutnya akan dilakukan perhitungan metrik evaluasi yang akan digunakan untuk mengukur performa algoritma. Metrik evaluasi yang dihitung adalah *accuracy, precision, recall* dan *f-score* yang akan dijabarkan sebagai berikut:

1. *Accuracy*
   Akurasi merupakan perbandingan antara jumlah prediksi yang benar (postif dan negatif) dengan jumlah total data yang ada.

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + FalseNegative + FalsePpositive + TrueNegative} \quad (2.13)$$

2. *Precision*
   Presisi adalah ukuran yang mengukur keakuratan prediksi positif dari model berdasarkan jumlah total data yang diidentifikasi sebagai kelas positif. Presisi dihitung dengan membagi jumlah prediksi positif yang benar oleh jumlah total prediksi positif (baik yang benar maupun yang salah).

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (2.14)$$

3. *Recall*

   Recall adalah ukuran yang menggambarkan sejauh mana model dapat mengenali atau mengidentifikasi data yang sebenarnya berlabel positif. Recall dihitung dengan membagi jumlah data positif yang teridentifikasi dengan benar oleh model dengan jumlah keseluruhan sampel yang sebenarnya memiliki label positif.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{2.15}$$

4. *F-Score*

   F-Score merupakan perbandingan antara *precision* dan *recall* yang telah dihitung.

$$F1 = 2X\frac{Recall * Precision}{Recall + Precision} \tag{2.16}$$