BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis Sentimen adalah studi komputer tentang pendapat, sikap, dan emosi orang terhadap suatu entitas. Secara umum, analisis sentimen membantu mengumpulkan informasi tentang aspek positif dan negatif dari topik tertentu [4]. Berikut merupakan beberapa jenis analisis sentimen yang digunakan untuk mengidentifikasi respon pengguna [12].

1. *Fine-Grained Sentiment Analysis*
   Analisis sentimen ini merupakan salah satu jenis yang paling umum. Fokusnya adalah pada tingkat polaritas pendapat. Jenis analisis sentimen ini akan mengelompokkan tanggapan atau pendapat ke dalam beberapa kategori, seperti sangat positif, sedikit positif, netral, sedikit negatif, dan negatif.

2. *Intent Sentiment Analysis*
   Analisis sentimen ini bertujuan untuk mengidentifikasi motivasi di balik pesan pengguna dan menelusuri untuk menentukan apakah berisi keluhan, saran, pendapat, pertanyaan, atau pujian terhadap suatu produk.

3. *Aspect - Based Sentiment Analysis*
   Jenis analisis sentimen ini berfokus pada elemen produk yang lebih spesifik. Analisis sentimen berbasis aspek ini juga memungkinkan untuk mengasosiasikan emosi tertentu dengan aspek produk yang berbeda.

2.2 Multinomial Naïve Bayes

Multinomial Naïve Bayes merupakan model berbasis frekuensi untuk klasifikasi teks yang direpresentasikan oleh kumpulan kata yang muncul dari suatu dokumen [5]. Adapun kelebihan dari *multinomial naïve bayes*, yaitu mudah digunakan pada data kontinu dan diskrit, dapat menangani kumpulan data besar, dapat mengklasifikasi data dengan banyak label, dan paling baik ketika melatih model *natural language processing* [6]. Metode ini mengikuti prinsip distribusi multinomial dalam probabilitas bersyarat.

Algoritma *Naïve Bayes* memiliki 2 fase, yaitu *Learning Phase* dan *Test Phase*. Pada fase *Learning*, sebagian data yang diketahui akan dijadikan sebuah

objek untuk membangun estimasi model. Pada fase *Test*, model yang terbentuk akan diuji dengan menggunakan sebagian besar data lain untuk mengetahui keakuratan model tersebut. Apabila akurasinya terbatas, maka model tersebut dapat digunakan untuk memprediksi kelas data yang tidak diketahui [13].

Menurut Shimodaira [14], tahapan dari klasifikasi *multinomial naïve bayes* dapat dijabarkan sebagai berikut.

1. Menentukan *vocabulary* (V), jumlah kata yang menentukan dimensi dari vektor fitur.
2. Menghitung hal-hal berikut pada set *training*:
   (a) N merupakan jumlah keseluruhan dokumen
   (b) N_k merupakan dokumen yang diklasifikasi ke kelas k
   (c) x_{it} dimana frekuensi kemunculan w_t pada suatu dokumen D_i untuk setiap kata pada V.
3. Menghitung *likelihoods* $P(w_t|C = k)$, frekuensi kemunculan dari suatu kata w_t di semua dokumen yang termasuk ke dalam suatu kategori C_k dengan menggunakan *Laplace's law of succession* untuk menghindari munculnya *zero probability problem*. Perhitungan *likelihoods* menggunakan *product* ($\prod$). Untuk mendapatkan *likelihoods* dengan menerapkan *Laplace's law of succession* dapat diihat pada persamaan berikut.

$$P(W_t|C_k) = \frac{1 + n_k(w_t)}{|V| + \sum_{S=1}^{|V|} n_k(w_s)} \tag{2.1}$$

4. Menghitung *priors*, probabilitas terklasifikasinya sebuah dokumen ke dalam sebuah kategori $P(C_k)$ dengan menggunakan persamaan berikut.

$$P(C_k) = \frac{N_k}{N} \tag{2.2}$$

Tahap selanjutnya adalah menerima dokumen yang belum diklasifikasi (D) dan kemudian akan diklasifikasi ke kelas yang telah ditentukan sebelumnya. Berikut merupakan persamaan untuk menghitung probabilitas suatu dokumen, dimana x merupakan kemunculan suatu kata dari dokumen (D).

$$P(C_k|D) = a\, P(C_k) \prod_{J=1}^{|V|} P(W_t \mid C_k)^{x_t} \tag{2.3}$$

2.3 Confusion Matrix

Confusion matrix merupakan matriks dua dimensi. *Confusion matrix* biasanya digunakan untuk meringkas kinerja klasifikasi dari pengklasifikasi terhadap data yang diuji [15].

Tabel 2.1. *Confusion Matrix*

Aktual \ Prediksi	Positif	Negatif
Positif	TP	FP
Negatif	FN	TN

TP merupakan *true positive*, jumlah dokumen dari kelas positif yang benar diklasifikasikan sebagai kelas positif. TN merupakan *true negative*, jumlah dokumen dari kelas negatif yang benar diklasifikasikan sebagai kelas negatif. FP merupakan *false positive*, jumlah dokumen dari kelas negatif yang salah diklasifikasikan sebagai kelas positif. FN merupakan *false negative*, jumlah dokumen dari kelas positif yang salah diklasifikasikan sebagai kelas negatif. Keempat nilai ini akan digunakan untuk perhitungan performa dari model. Ada beberapa rumus perhitungan yang umum digunakan, antara lain *accuracy*, *precision*, dan *recall*. Berikut merupakan formula *confusion matrix* [16].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.4}$$

$$Precision = \frac{TP}{TP+FP} \tag{2.5}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.6}$$

$$F1\text{-}score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2.7}$$

2.4 Text Preprocessing

Text Preprocessing adalah suatu cara untuk mengurangi *noise* dari data yang akan digunakan pada tahapan berikutnya. *Text preprocessing* berpotensi besar pada performa akhir [17]. *Text preprocessing* mempersiapkan teks yang tidak terstruktur

menjadi data yang siap untuk diolah. Tidak ada urutan yang baku terkait tahapan dalam penggunaan *text preprocessing*. Berikut merupakan beberapa prosedur yang umum digunakan dalam *text preprocessing* [18].

1. *Case Folding*
   Dalam suatu dokumen atau data biasanya ditemukan sebuah kesalahan penulisan dalam menggunakan huruf kapital dan huruf kecil. *Case Folding* merupakan proses untuk mengkonversi teks ke dalam format huruf kecil. Tujuan dari *case folding* utama adalah untuk memberikan bentuk standar pada teks [19].

2. *Tokenizing*
   Tokenizing adalah proses pemotongan teks menjadi bagian yang lebih kecil, yang disebut sebagai token. Tujuan dari proses ini adalah untuk menghilangkan angka, tanda baca dan karakter lain yang dianggap tidak memiliki pengaruh terhadap pemrosesan teks.

3. *Stopword*
   Stopword Removal adalah tahap pemilihan kata-kata yang dianggap penting atau menghapus kata-kata yang tidak penting seperti kata depan dan kata benda. Terdapat dua metode yang dapat digunakan dalam tahap ini, antara lain *Stoplist* dan *Wordlist*.

4. *Stemming*
   Stemming merupakan proses pengubahan bentuk kata menjadi kata dasar atau tahap mencari root dari tiap kata. Berikut merupakan contoh algoritma *stemming* yang dapat digunakan.

   (a) *Porter Stemmer*

   (b) *Stemming Nazief-Adriani*

   (c) *Stemming Arifin-Setiono*

   (d) *Khoja*

2.5 Term Frequency - Inverse Document Frequency

Term Frequency - Inverse Document Frequency atau TF-IDF merupakan metode untuk memberikan bobot pada setiap kata. TF-IDF menghitung jumlah *term* yang muncul dalam suatu dokumen (*term frequency*) dan kemudian menghitung

kemunculan *term* di semua dokumen [20]. TF merupakan sebuah pengukuran dari banyaknya suatu kata dalam kalimat. Perhitungan TF dapat menggunakan banyaknya kata dalam suatu kalimat tersebut [21]. IDF atau *Inverse Document Frequency* berguna untuk melihat kemunculan setiap kata pada kumpulan kalimat. Jika suatu kata terdapat dalam satu kalimat, maka dihitung satu, dan jika terdapat dalam dua kalimat, maka akan dihitung dua. Berikut merupakan rumus perhitungan TF-IDF.

1. *Term Frequency* (TF)

$$tf(t,d) = \frac{n_i, j}{\sigma_k n_i, j} \tag{2.8}$$

Keterangan :

$tf(t,d)$: Frekuensi *term*

n_i, j : Total suatu *term* yang muncul pada suatu dokumen

$\sigma_k n_i, j$: Jumlah seluruh kata dalam suatu dokumen

2. *Inverse Document Frequency* (IDF)

$$idf(t) = \log \frac{N}{df_t} \tag{2.9}$$

Keterangan :

N : Total dokumen

df_t : Jumlah dokumen yang mengandung *term* t

3. *Term Frequency - Inverse Document Frequency* (TF-IDF)

$$w_{t,d} = tf(t,d) \times idf(t) \tag{2.10}$$

Keterangan :

$w_{t,d}$: Bobot *term* pada suatu dokumen

$tf(t,d)$: Frekuensi *term*

$idf(t)$: Bobot kemunculan *term* t pada seluruh dokumen