BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen adalah langkah-langkah pengumpulan dan evaluasi pendapat, pemikiran, serta kesan dari individu terkait beragam topik, produk, subjek, dan layanan. Opini masyarakat memiliki potensi untuk memberikan manfaat signifikan bagi perusahaan, pemerintah, dan individu dalam pengumpulan informasi dan pengambilan keputusan berdasarkan persepsi umum [16]. Dalam proses analisis opini, tanggapan tersebut dikelompokkan menjadi kategori positif, negatif, atau netral, memungkinkan pemahaman yang lebih mendalam mengenai sentimen yang terkandung dalam suatu konteks [17].

2.2 Twitter

Twitter adalah salah satu media sosial paling populer di kalangan pengguna internet. Hal ini disebabkan oleh antarmuka yang sederhana dan kemudahan dalam penggunaannya, yang memungkinkan pengguna dengan leluasa menyampaikan pendapat atau opini mereka [18]. Opini yang dibagikan oleh pengguna dapat dianalisis melalui analisis sentimen, yang bertujuan mengidentifikasi dan mengkategorikan sikap dalam sebuah teks untuk menentukan apakah konten tersebut bersifat positif, negatif, atau netral sesuai dengan kategorisasi yang ditetapkan. Sebagian besar pengguna Twitter memiliki kebebasan untuk mengekspresikan pandangan mereka mengenai berbagai topik [19].

2.3 *Naive Bayes*

Naive Bayes Classifier merupakan teknik klasifikasi yang berakar dari *teorema Bayes*, di mana tujuannya adalah memaksimalkan peluang hasil yang akan muncul dan juga dapat meningkatkan keyakinan terhadap klasifikasi berdasarkan variabel dan kondisi yang terlibat. Dengan kata lain, metode ini digunakan untuk meningkatkan kepercayaan terhadap cara data diklasifikasikan berdasarkan faktor-faktor yang terlibat [20]. Teorema Bayes dikemukakan oleh seorang ilmuwan Inggris yang bernama Thomas Bayes sebagai suatu prinsip untuk memproyeksikan probabilitas kejadian masa depan berdasarkan pengalaman sebelumnya. Berikut

rumus dari algoritma *bayes* [21].

$$P(H|U) = \frac{P(U|H)P(H)}{P(U)} \tag{2.1}$$

1. $P(H|U)$ menyatakan probabilitas akhir dari hipotesis H ketika sebuah peristiwa tertentu X terjadi.
2. $P(U|H)$ menyatakan probabilitas bukti bahwa peristiwa X akan memengaruhi H.
3. $P(H)$ menyatakan probabilitas awal ketika H terjadi, terlepas dari bukti apa pun.
4. $P(U)$ menyatakan probabilitas awal bukti X dari H atau bukti lainnya.

Algoritma *Naive Bayes* Memiliki beberapa varian dua diantaranya adalah *Bernoulli Naive Bayes* dan *Complement Naive Bayes* [22]. Bernoulli *Naive Bayes* menggunakan vektor biner untuk merepresentasikan sebuah dokumen, yang mengindikasikan apakah sebuah kata muncul atau tidak di dalam dokumen tersebut. Bernoulli *Naive Bayes* menggunakan angka biner dalam melakukan klasifikasi. Jenis *Naive Bayes* ini menggunakan data diskrit dan menerima fitur hanya sebagai nilai biner, seperti benar atau salah, ya atau tidak, berhasil atau gagal, 0 atau 1, dan sebagainya [23]. Berikut rumus dari Bernouli *Naive Bayes*.

$$P(y|x) = \frac{P(y) \times \prod_{i=1}^{n} P(x_i|y)}{P(x)} \tag{2.2}$$

Keterangan:

- $P(y|x)$ adalah probabilitas kelas y untuk instance x,
- $P(y)$ adalah probabilitas prior untuk kelas y,
- $P(x_i|y)$ adalah probabilitas kemunculan fitur biner x_i dalam kelas y, dan
- $P(x)$ adalah probabilitas dari instance x, tetapi sering kali diabaikan karena nilainya sama untuk setiap kelas dan hanya memengaruhi skala, tidak memengaruhi perbandingan probabilitas kelas.

Complement *Naive Bayes*. Model ini merupakan adaptasi dari algoritma *Multinomial Naive Bayes (MNB)* standar yang cocok digunakan untuk dataset yang tidak seimbang. *Complement NB* menggunakan statistik dari komplemen setiap

kelas untuk menghitung bobot model. Ini akan mengurangi bias dalam estimasi bobot dan akan meningkatkan akurasi klasifikasi. Berikut rumus dari *Complement Naive Bayes* [24].

$$P(x_i|y) = \frac{\sum_{j=1}^{m} \text{jumlah sampel dengan fitur } x_i \text{ dalam kelas } y_j \text{ (semua kelas kecuali } y)}{\sum_{j=1}^{m} \text{jumlah total sampel dalam kelas } y_j} \tag{2.3}$$

Keterangan:

- $P(x_i|y)$ adalah probabilitas kemunculan fitur x_i dalam kelas y,
- y_j adalah kelas yang berbeda-beda, dan
- m adalah jumlah kelas dalam dataset.

2.4 *Text Mining*

Text mining adalah proses di mana komputer digunakan untuk menemukan informasi baru yang sebelumnya belum diketahui dari berbagai sumber teks. Komputer melakukannya dengan mengekstrak informasi secara otomatis dari dokumen atau sumber teks lainnya. Pada dasarnya, *text mining* dapat membantu dalam menemukan pola, trend, atau informasi baru dari banyak teks yang mungkin sulit atau tidak mungkin ditemukan secara manual oleh manusia [25].

2.5 *Text-Preprocessing*

Bahasa yang digunakan di situs media sosial berbeda dengan bahasa yang biasa ditemukan di media arus utama, dan bentuk kata yang digunakan terkadang tidak sesuai dengan yang ada di kamus. Selain itu, banyak pengguna media sosial yang menggunakan slang (yaitu kata-kata informal atau tidak baku), yang membuat pentingnya dilakukan *text pre-processing* [26]. *Text Pre-processing* adalah langkah awal dalam analisis teks atau pemrosesan bahasa alami *(NLP)* yang bertujuan untuk membersihkan dan menyiapkan teks mentah sebelum analisis atau penggunaan dalam model *machine learning*. Tujuan utamanya adalah untuk meningkatkan kualitas data dan memastikan data dapat digunakan secara efektif dalam model pembelajaran mesin [27].

1. *Cleaning* Data merupakan proses dalam membersihkan data dengan cara menghapus *noise* seperti *username, hashtag*, emoji, *URL, RT(retweet)*, angka, dan tanda baca [28].

2. *Drop data duplicate* merupakan proses penghapusan entri atau baris yang duplikat dari suatu kumpulan data. Dalam konteks ini, drop mengacu pada tindakan menghapus atau menghilangkan, sementara data duplikat merujuk pada entri atau baris dalam dataset yang memiliki nilai yang sama atau identik dengan entri lain dalam dataset tersebut [29].

3. *Case Folding* merupakan proses mengonversi semua huruf dalam teks menjadi huruf kecil [30].

4. *Stemming* merupakan proses mengubah kata-kata menjadi kata dasarnya dengan menghapus afiks atau mengubah kata kerja menjadi kata benda. Tujuan dari proses ini adalah untuk menghilangkan berbagai afiks pada kata, memperoleh kata dasar sehingga hasil yang lebih akurat diperoleh [31].

5. *Tokenize* merupakan metode yang bertujuan dengan memisahkan urutan kata dalam sebuah kalimat, paragraf, atau halaman menjadi potongan-potongan kata. Dalam melakukan proses pemisahan kalimat menjadi kata-kata ini dilakukan dengan cara memotong kalimat berdasarkan spasi [32].

6. *Labeling* merupakan proses memberikan sebuah kategori atau label kepada data berdasarkan karakteristik atau atribut tertentu yang dimiliki oleh data tersebut. Dalam konteks analisis data atau pembelajaran mesin, labeling biasanya dilakukan untuk mempersiapkan data untuk tugas tertentu, seperti klasifikasi atau prediksi [33]

7. *Stop Word* merupakan proses menghapus kata-kata yang dianggap tidak penting untuk mengurangi kompleksitas dan beban ruang data. Kata-kata umum dalam dokumen teks, seperti kata ganti, preposisi, dan kata benda, seringkali tidak memiliki makna signifikan dalam konteks tertentu. Contoh kata-kata yang tidak dianggap penting adalah ”yang”, ”di”, ”ke”, ”dari”, ”pada”, dan lain-lain [32].

2.6 *TF-IDF*

TF-IDF adalah sebuah indikator numerik yang digunakan untuk menghitung seberapa sering sebuah kata muncul dalam satu dokumen dibandingkan dengan kumpulan dokumen secara keseluruhan [34]. *TF (Term Frequency)* dihitung dengan jumlah kemunculan kata target dalam dokumen, memberikan gambaran tentang frekuensi kata tersebut dalam konteks dokumen tersebut. *IDF (Inverse Document Frequency)*, di sisi lain, diukur berdasarkan seberapa banyak dokumen dalam kumpulan dokumen yang mengandung kata tersebut, menunjukkan pentingnya kata tersebut dalam konteks yang lebih luas. Oleh karena itu *TF-IDF* menggabungkan kedua konsep tersebut dengan memberikan sebuah skor yang menunjukkan seberapa pentingnya suatu kata-kata tersebut pada suatu dokumen. [35]. Berikut adalah rumus dari *TF-IDF*.

$$\text{TF-IDF}_{t,d} = \text{TF}_{t,d} \times \text{IDF}_t \tag{2.4}$$

$$\text{TF}_{t,d} = \frac{n_{t,d}}{\sum_k n_{k,d}} \tag{2.5}$$

$$\text{IDF}_t = \log\left(\frac{N}{DF_t}\right) \tag{2.6}$$

Keterangan:

t, d	: Kata-kata yang dihitung, Bobot kalimat
$\text{TF-IDF}_{t,d}$	: Kalimat bobot (d) terhadap kata (t)
$\text{TF}_{t,d}$	: Term frequency
IDF_t	: Inverse document frequency
N	: Jumlah kalimat
DF_t	: Jumlah kata yang terulang
$n_{i,j}$	: Total term yang muncul pada satu dokumen
$\sum_k n_{i,j}$	: Total seluruh kata dalam satu dokumen

2.7 *Confusion Matrix*

Confusion Matrix merupakan metode yang digunakan dalam mengevaluasi kinerja dari suatu klasifikasi. Metode ini menggunakan suatu tabel untuk menentukan antara prediksi yang benar dan prediksi yang salah yang dihasilkan oleh suatu algoritma klasifikasi pada suatu dataset [36]. Berikut tabel 2.1.

Tabel 2.1. Confusion Matrix

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

1. *True Positif (TP)* merujuk pada hasil klasifikasi yang tepat masuk ke dalam kelas yang benar.
2. *False Negative (FN)* terjadi ketika hasil klasifikasi salah diklasifikasikan ke dalam kelas yang benar.
3. *Positif Palsu (FP)* dihitung ketika hasil klasifikasi secara tidak tepat dikategorikan ke dalam kelas yang salah.
4. *True Negative (TN)* merujuk pada kasus di mana hasil klasifikasi salah diklasifikasikan ke dalam kelas yang salah.

Setelah mendapatkan hasil dari *Confusion Matrix*, dilakukan perhitungan metrik evaluasi kinerja model, termasuk akurasi, presisi, recall, dan F-1 score. Metode ini memungkinkan evaluasi mendalam tentang sejauh mana kinerja model klasifikasi tersebut.

1. *Recall* mengukur seberapa baik model dapat menemukan semua instance positif yang seharusnya ditemukan. Rumusnya adalah rasio antara *true positive* dengan jumlah true positive dan *false negative* [37]. Berikut persamaan dalam mencari nilai Recall.

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{2.7}$$

2. *Precision* menggambarkan fraksi dari *instance positif* yang diprediksi dengan benar di antara semua *instance positif* yang diprediksi oleh model. Rumusnya

adalah rasio antara *true positive* dengan jumlah *true positive* dan *false positive* [37]. Berikut adalah persamaan untuk mencari nilai precision.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100\% \tag{2.8}$$

3. *Accuracy* adalah persentase dari keseluruhan observasi yang diklasifikasikan dengan benar oleh model. Rumusnya adalah rasio antara jumlah *true positive* dan *true negative* dengan keseluruhan jumlah observasi [37]. Berikut persamaan dalam mencari nilai *accuracy*.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \times 100\% \tag{2.9}$$

4. *F-score* adalah ukuran yang menunjukkan keseimbangan antara *precision* dan *recall*. Ini memberikan gambaran tentang sejauh mana model mencapai keseimbangan antara mengidentifikasi instance positif dan menghindari kesalahan *false positive*. Rumusnya adalah 2 kali perkalian *precision* dan *recall*, dibagi dengan jumlah *precision* dan *recall* [37]. Berikut persamaan dalam mencari nilai F-score.

$$F\text{-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

2.8 *K-Fold Cross Validation*

Cross-validation (CV) adalah metode esensial dalam evaluasi dan seleksi model karena mampu memberikan informasi yang tidak bisa diperoleh hanya melalui kalibrasi model. *CV* menguji kemampuan model untuk menggeneralisasi pada data baru, yang penting untuk memastikan model bekerja baik tidak hanya pada data pelatihan tetapi juga pada situasi baru. Dengan *CV*, ketahanan atau robustitas model dapat diungkap melalui akurasi dan keberhasilannya dalam klasifikasi pada situasi baru. Dalam *CV*, data dibagi menjadi beberapa *subset (fold)* dan model dilatih pada beberapa *subset* tersebut lalu diuji pada *subset* yang tersisa. Proses ini diulang sehingga setiap subset digunakan sebagai data uji setidaknya sekali, memberikan gambaran jelas tentang performa model pada berbagai bagian data [38].

CV juga menjadi kunci dalam menentukan sejauh mana sebuah model mengalami *overfitting*. *Overfitting* terjadi ketika model terlalu kompleks dan mulai

menangkap *noise* atau pola acak dalam data pelatihan yang tidak berlaku pada data baru. *CV* membantu mengidentifikasi *overfitting* dengan membandingkan performa model pada data pelatihan dan data validasi. Jika performa model jauh lebih baik pada data pelatihan dibandingkan data validasi, ini adalah indikasi *overfitting*. Secara keseluruhan, *CV* memberikan penilaian komprehensif dan realistis tentang kemampuan model, membantu memastikan model *robust*, tidak *overfitting*, dan performa baik pada data baru. Hal ini menjadikan *CV* sebagai alat penting dalam pengembangan dan evaluasi model [38].

Untuk memastikan ketangguhan dan generalisasi model, teknik validasi silang 5 kali lipat *(5-fold cross-validation)* diimplementasikan. Pendekatan ini melibatkan partisi dataset menjadi lima *subset* yang berbeda, dengan setiap *subset* berfungsi sebagai set pengujian sementara data yang tersisa digunakan untuk pelatihan. Proses ini diulangi lima kali, setiap kali dengan *subset* yang berbeda sebagai set pengujian, sehingga memastikan bahwa setiap titik data digunakan untuk pelatihan dan pengujian. Validasi silang membantu mengurangi *overfitting* dan memberikan cerminan yang lebih akurat dari kinerja model pada data yang tidak terlihat. Pendekatan ini secara matematis direpresentasikan sebagai [39]:

$$E_{CV} = k_1 \sum_{i=1}^{k} E_i$$

di mana k adalah jumlah lipatan *(folds)* dan E_i adalah kesalahan pada lipatan ke-i.