BAB 2
LANDASAN TEORI

Dalam melakukan penelitian analisis sentimen ini, terdapat beberapa literatur yang digunakan sebagai dasar teori untuk mendukung penelitian. Berikut adalah definisi dan model matematika yang berkaitan dengan masalah yang diteliti.

2.1 Analisis Sentimen

Analisis sentimen merupakan studi komputasi tentang pendapat, sikap, dan emosi orang terhadap suatu topik dan akan diklasifikasi sebagai pernyataan opini sentimen positif atau negatif [10]. Analisis sentimen merupakan proses dalam mendeteksi polaritas kontekstual teks yang dapat menentukan apakah teks yang diberikan positif, negatif, atau netral [11]. Sehingga dapat disimpulkan bahwa analisis sentimen adalah proses yang digunakan untuk mengidentifikasi dan mengekstrak informasi dari teks. Jenis analisis ini sering digunakan di bidang media sosial dan pemasaran, karena dapat membantu bisnis memahami sentimen pelanggan terhadap produk atau layanan yang diberikan.

Proses analisis sentimen biasanya melibatkan pelatihan model pembelajaran mesin pada kumpulan data teks berlabel yang besar. Kumpulan data ini berisi teks yang telah diberi label sebagai positif atau negatif. Setelah model dilatih, model tersebut dapat digunakan untuk mengklasifikasikan teks baru secara otomatis dengan sentimen positif atau negatif.

Dalam melakukan analisis sentimen, terdapat banyak tantangan yang harus dilewati. Dalam survey yang dilakukan oleh S.Rajalakshmi, S.Asha, dan N.Pazhaniraja ditemukan tantangan untuk melakukan analisis sentimen [12]. Berikut adalah contoh dari tantangan dalam melakukan analisis sentimen.

1. Tinjauan palsu dan deteksi spam
   Dalam melakukan analisis sentimen, sangat penting untuk menghilangkan konten spam sebelum dilakukan pra-pemrosesan. Klasifikasi sentimen yang efektif dapat dilakukan dengan mendeteksi outlier dengan mempertimbangkan reputasi tinjauan dan dengan mengidentifikasi duplikat.

2. Batasan penyaringan klasifikasi
   Beberapa pendapat tidak relevan akan disaring untuk menentukan konsep

yang paling populer yang merupakan batasan hasil dalam penyaringan klasifikasi. Keterbatasan ini yang harus dikurangi untuk klasifikasi sentimen yang lebih baik tetapi hasilnya adalah ringkasan sentimen yang salah.

2.2 Twitter

Twitter merupakan layanan komunikasi *online* yang menyediakan wadah bagi masyarakat untuk menulis teks singkat mengenai opininya secara singkat, jelas, dan padat yang dapat dipublikasikan [13]. Pada penelitian ini akan menggunakan *library* snscrape. Snscrape adalah alat *scraping* untuk layanan jejaring sosial media (SNS). *Library* ini dapat melakukan *scrapes* seperti pengguna, profil pengguna, tagar, pencarian, dan posting tanpa menggunakan API Twitter.

2.3 Text Pre-Processing

Text pre-processing adalah proses menyiapkan data teks untuk digunakan dalam tugas pemrosesan bahasa alami (NLP). *Pre-processing* dilakukan untuk menghilangkan data yang tidak lengkap, *noisy*, dan tidak konsisten, sehingga harus melewati proses ini terlebih dahulu sebelum dilakukan proses selanjutnya [14]. Pada tahap ini, data biasanya akan dibersihkan dan akan dilakukan normalisasi teks, serta ekstraksi fitur yang relevan dari teks seperti kata, frasa, dan simbol. *Pre-processing* merupakan salah satu langkah penting dalam pemrosesan bahasa alami karena membantu untuk meningkatkan akurasi, presisi, dan recall serta kinerja dari model NLP. Berikut adalah tahapan yang dilakukan pada saat melakukan proses *text pre-processing*.

1. *Data Cleansing*

   Data Cleansing merupakan tahap pembersihan dan normalisasi teks untuk menghilangkan informasi yang salah, tidak konsisten, dan kurang relevan. Proses ini mencakup penghapusan tanda baca, karakter khusus, angka, dan standarisasi teks dengan mengubahnya menjadi huruf kecil (*case folding*), mengubah kata slang, serta menghapus *stopword*. Proses ini dilakukan untuk membantu mengurangi *noisy* dan membuat teks lebih mudah diolah, sehingga dapat meningkatkan kinerja model NLP. Selain itu, dengan dilakukan pembersihan dan hanya memfokuskan kepada fitur penting akan meningkatkan kemampuan dalam melakukan prediksi.

2. *Back-Translation*

   Back-Translation merupakan teknik pengolahan NLP dengan cara menerjemahkan teks ke bahasa lain, kemudian menerjemahkan kembali teks tersebut ke bahasa asli. Hasil akhir dari proses ini adalah teks baru yang serupa dengan teks asli, namun mungkin memiliki beberapa perbedaan. Tahap ini digunakan untuk meningkatkan performa model, karena akan menambah variasi pada dataset terjemahan, sehingga model dapat mempelajari lebih banyak pola dan konteks yang lebih bervariasi.

3. *Tokenization*

   Tokenization merupakan proses pemecahan teks menjadi unit yang lebih kecil yang disebut dengan token, yang biasanya sesuai dengan kata. Dengan melakukan proses ini, model NLP dimungkinkan untuk bekerja dengan token individu dibandingkan dengan bekerja dengan seluruh teks. Dimana hal ini dapat membantu untuk meningkatkan analisis teks pada tingkat yang lebih terperinci dan membuat akurasi, presisi, dan recall prediksi meningkat.

4. *Stemming*

   Stemming adalah proses untuk mereduksi kata menjadi bentuk dasarnya yang dapat disebut dengan stem. Proses ini membantu dalam mengurangi kosa kata yang akan dikerjakan oleh model NLP, sehingga dapat membuat model menjadi lebih efisien. *Stemming* dapat meningkatkan performa model NLP dengan mengurangi variasi kata seperti bentuk tunggal dan jamak, contohnya adalah kata ”bersama” dan ”kebersamaan”, keduanya akan direduksi menjadi kata dasar ”sama” yang akan mempermudah model dalam mengidentifikasinya.

5. *Data Labelling*

   Data labelling adalah proses untuk menglabel data *tweets*. Proses ini akan dilakukan menggunakan metode kamus Inset. Didalam kamus Inset terdapat kata berbahasa Indonesia dan bobot pada setiap katanya untuk menunjukan kata yang bersifat positif maupun negatif.

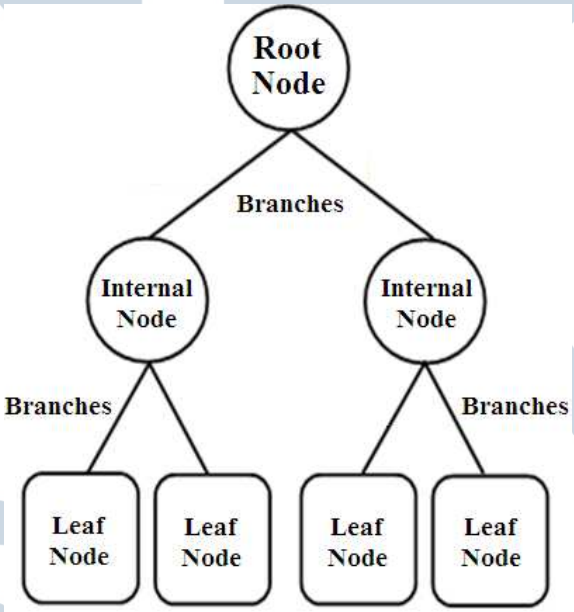
2.4 Decision Tree

Decision tree merupakan salah satu metode klasifikasi dengan graf yang merepresentasikan pilihan dan hasilnya dalam bentuk pohon [15]. Setiap *node* dalam grafik akan mewakili peristiwa atau pilihan dan tepi grafik akan mewakili

aturan atau kondisi keputusan. Setiap pohon terdiri dari simpul (*node*) dan cabang (*branch*). Setiap simpul akan mewakili atribut dalam grup yang akan diklasifikasikan dan setiap cabang akan mewakili nilai yang dapat diambil oleh *node* tersebut.

Dalam penggunaannya, *decision tree* memiliki beberapa kelebihan [16] sebagai berikut.

1. Dapat menangani masalah multi-output
2. Dapat divisualisasikan dan mudah untuk dipahami
3. Dapat bekerja dengan data kecil, sedangkan model lain membutuhkan normalisasi data, pembuatan variabel dummy, dan menghapus nilai *null*.
4. Dapat menangani data kategorik dan numerik, sedangkan model lain hanya dikhususkan untuk satu jenis variabel.



Gambar 2.1. Struktur Dasar *Decision Tree*
Sumber: [17]

Dalam penggunaan algoritma *decision tree* akan dilakukan perhitungan *entropy* dan *information gain*.

1. *Entropy* digunakan untuk mengukur keacakan dataset dengan nilai antara 0 dan 1 [18]. Semakin mendekati 0, nilai *entropy* semakin dikatakan baik.

Namun, dikatakan buruk jika *entropy* bernilai 0. Berikut merupakan rumus dari perhitungan *entropy*.

$$Entropy(S) = \sum_{i=1}^{c} - p_i \log_2 p_i \tag{2.1}$$

Dimana c adalah jumlah kelas yang mungkin dalam dataset dan p_i adalah probabilitas dari kelas ke-i dalam dataset S.

2. *Information Gain* adalah salah satu metrik yang digunakan untuk segmentasi [18]. Ini secara intuitif menginformasikan banyaknya pengetahuan tentang variabel acak, dimana nilai ini berkebalikan dengan *entropy*, semakin tinggi nilainya semakin baik. Berikut merupakan rumus dari perhitungan *information gain*.

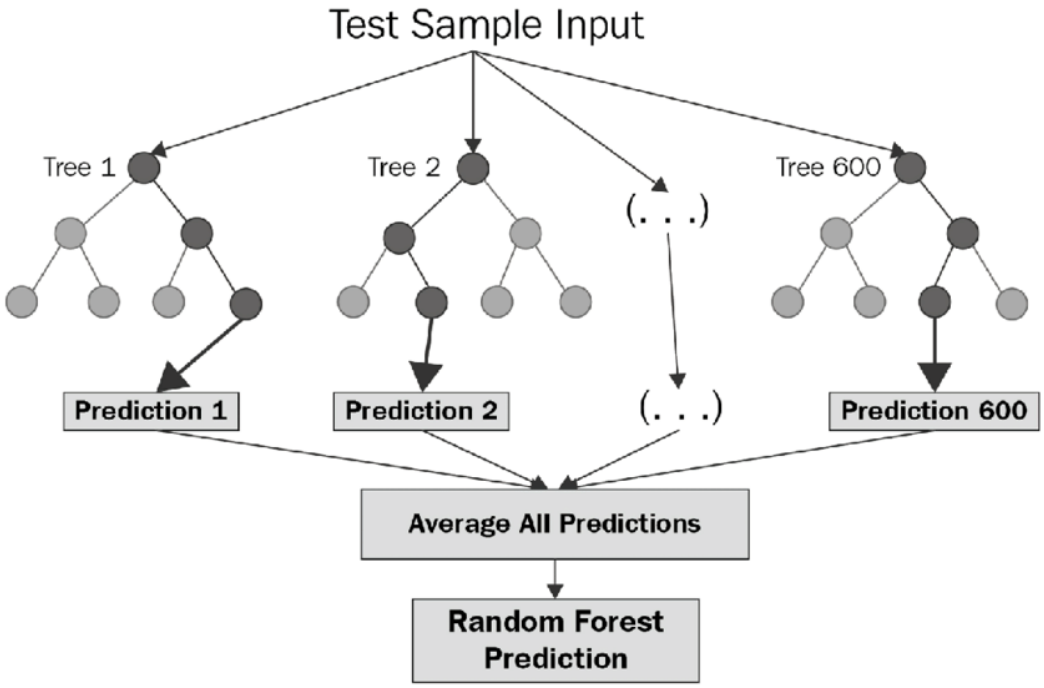
$$Gain(S,A) = Entropy(S) - \sum_{v \varepsilon V(A)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{2.2}$$

Dimana $V(A)$ adalah himpunan dari semua nilai yang mungkin untuk atribut A. S_v adalah jumlah sampel dalam subset S_v yang memiliki atribut A bernilai v. S adalah jumlah sampel total dalam dataset S.

2.5 Random Forest

Random Forest adalah teknik ensembel yang menggabungkan prediksi dari beberapa prediktor dasar yang dibangun dengan algoritma *decision tree* untuk meningkatkan ketahanan terhadap estimator individual [19]. Prediktor pohon akan disusun dalam suatu format sehingga setiap pohon akan bergantung pada nilai-nilai vektor acak yang memiliki pola secara independen dan semua pohon didistribusikan seragam di seluruh hutan [20]. Dari pengertian tersebut dapat disimpulkan bahwa *Random Forest* adalah metode pembelajaran ensembel yang terdiri dari kumpulan *decision tree*. Metode ini disebut sebagai ”hutan” karena melibatkan banyak pohon dan disebut ”acak” karena pohon akan dilatih dengan data acak. *Random Forest* akan mengkombinasikan masing-masing *tree* dari model *decision tree* menjadi 1 model. Banyaknya jumlah *tree* yang dipakai dapat mempengaruhi nilai akurasi, presisi, dan recall dari model *Random Forest*. Pemilihan *tree* dari model *decision tree* dimulai dari perhitungan nilai *entropy* dan dicari *entropy* terbaik untuk dipakai di model *Random Forest*.

Tujuan dari *Random Forest* ini adalah untuk membuat prediksi dengan menggabungkan banyak pohon di hutan. Hal ini dilakukan dengan melakukan perhitungan voting atau rata-rata probabilitas prediksi untuk tugas klasifikasi. *Random Forest* dianggap sebagai model pembelajaran mesin yang sangat akurat dan kuat [21]. Model ini dapat digunakan secara luas untuk menyelesaikan berbagai tugas, termasuk regresi dan klasifikasi.



Gambar 2.2. Struktur *Random Forest*
Sumber: [22]

2.6 Term Frequency Inverse Document Frequency (TF-IDF)

TF-IDF merupakan metode pembobotan statistik untuk mengukur pentingnya suatu kata dalam dokumen terhadap keseluruhan bahasa alami [23]. TF-IDF dapat dipakai untuk berbagai tugas seperti ekstraksi token kata dari artikel, menentukan peringkat, dan menghitung tingkat kemiripan antar dokumen [24]. Dalam metode ini, terdapat tiga ukuran statistik, yaitu TF, IDF, dan TF-IDF dihitung untuk setiap token kata.

1. *Term Frequency* (TF)

   Perhitungan statistika TF adalah matrik yang akan mencerminkan seberapa

penting suatu kata sering muncul dalam sebuah dokumen. Semakin tinggi nilai TF, maka akan semakin penting suatu kata dalam dokumen tersebut. Berikut merupakan rumus dari perhitungan TF [19].

$$TF(t,d) = 1 + log(f_{t,d}) \tag{2.3}$$

Dimana, $f_{t,d}$ menunjukan kemunculan berulang suatu kata atau istilah t dalam dokumen d.

2. *Inverse Document Frequency* (IDF)
   Perhitungan statistika IDF merupakan keterbalikan dari perhitungan TF, dimana IDF akan menghitung seberapa jarang suatu kata atau istilah muncul pada semua dokumen. Semakin tinggi nilai IDF, maka semakin jarang kata tersebut muncul pada suatu dokumen. Berikut merupakan rumus dari perhitungan IDF [19].

$$IDF(t) = 1 + log(\frac{N_d}{df_t}) \tag{2.4}$$

   Dimana Nd adalah jumlah dokumen dalam dataset dan df_t adalah jumlah dokumen yang mengandung kata atau istilah t.

3. TF-IDF
   Ketika frekuensi suatu kata tertentu dalam suatu dokumen tinggi dan jumlah dokumen yang mengandung kata tersebut rendah, maka nilai bobot TF-IDF akan naik. Berikut adalah rumus dari perhitungan TF-IDF [19].

$$TF - IDF = TFt,d \cdot IDFt,d \tag{2.5}$$

2.7 K-Fold Cross-Validation

Cross-Validation adalah metode dalam *machine learning* dan statistik untuk menilai kinerja dan kemampuan generalisasi model prediksi [25]. Bentuk *cross-validation* paling umum disebut dengan *k-fold cross validation*, dimana metode ini akan memisahkan kumpulan sampel secara acak menjadi serangkaian lipatan berukuran sama, dimana k menunjukan jumlah lipatan atau partisi [26]. Misalnya, jika nilai k adalah lima, maka *dataset* akan dibagi menjadi lima partisi.

2.8 Confusion Matrix

Dalam melakukan evaluasi klasifikasi, salah satu metode yang dapat digunakan untuk evaluasi model adalah dengan *confusion matrix*. *Confusion matrix* berisi perbandingan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya, dimana terdapat empat kondisi untuk mengukur kinerja [27].

1. *True Positive* (TP), akan menandakan data positif yang masuk ke sistem dan terdeteksi positif oleh sistem.
2. *False Positive* (FP), akan menandakan data negatif yang masuk ke sistem, namun terdeteksi positif oleh sistem.
3. *True Negative* (TN), akan menandakan data negatif yang masuk ke sistem dan terdeteksi negatif oleh sistem.
4. *False Negative* (FN), akan menandakan data positif yang masuk ke sistem, namun terdeteksi negatif oleh sistem.

Tabel 2.1. *Confusion Matrix*

Class		Actual	
		Positive	Negative
Predict	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

sumber: [27]

Fokus pada pemakaian *confusion matrix* ini adalah pada kemampuan prediksi model daripada seberapa cepat model yang diperlukan untuk melakukan klasifikasi [28]. Setiap baris akan mewakili instance dalam kelas yang diprediksi, sedangkan setiap kolom akan mewakili dalam sebuah kelas yang sebenarnya. Berdasarkan empat kondisi diatas, akan diperoleh nilai akurasi, presisi, dan recall. Nilai akurasi akan menggambarkan seberapa akurat sistem dapat melakukan klasifikasi data dengan benar. Nilai presisi akan menggambarkan tingkat akurasi antara informasi yang diminta dengan jawaban yang diberikan oleh sistem. Sedangkan recall akan menggambarkan tingkat keberhasilan sistem dalam menemukan kembali informasi [27].

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} * 100\% \tag{2.6}$$

$$Presisi = \frac{TP}{TP + FP} * 100\% \tag{2.7}$$

$$Recall = \frac{TP}{TP + FN} * 100\% \tag{2.8}$$