BAB 2
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen merupakan proses dalam menentukan sentimen serta mengelompokkan teks kedalam dokumen atau kalimat sehingga dapat dikategorikan sebagai sentimen bersifat positif atau negatif [8]. Analisis sentimen kerap disamakan dengan *opinion mining* [9], dikarenakan berfokus pada pendapat yang menyatakan positif maupun negatif. Data tekstual seperti ulasan produk, layanan, fenomena, individu dapat dijadikan sebagai topik dalam melakukan penelitian mengenai analisis sentimen [10].

2.2 Analog Switch Off

Analog Switch Off (ASO) merupakan sebuah kebijakan dalam melakukan migrasi dari televisi analog ke televisi digital [3]. Kebijakan tersebut dilaksanakan sejalan dengan perkembangan teknologi dan informasi di Indonesia. Peraturan Pemerintah Nomor 46 Tahun 2021 tentang Pos, Telekomunikasi dan Penyiaran, pada Pasal 72 angka 8 menyatakan bahwa migrasi penyiaran terestial teknologi analog ke digital harus diselesaikan paling lambat dua tahun sejak diundangkan. Dengan begitu, kebijakan ASO paling lambat terjadi pada tanggal 2 November 2022 [2].

2.3 Twitter

Twitter merupakan layanan media sosial yang menyediakan wadah bagi masyarakat untuk menulis teks singkat mengenai opini secara singkat, jelas, dan padat yang dapat dipublikasikan [11]. Pada penelitian ini library *snscrape* akan digunakan. *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*.

2.4 Text Preprocessing

Text Processing merupakan tahapan yang bertujuan untuk mencegah penurunan performa analisis secara signifikan [12]. Secara umum, tahapan *text processing* terbagi menjadi beberapa bagian seperti *data cleaning*, *data labelling*, *casefolding*, *stemming* , dan *tokenizing* [13]. Penjelasan tahapan *text processing* adalah sebagai berikut:

- Data Cleaning

  Pada tahap ini, data yang ada akan dibersihkan di mana karakter seperti simbol akan dihapus. Selain itu, emoji, URL, dan *username* juga akan dihapus pada tahap ini. Penghapusan ini bertujuan untuk mengurangi dan menghindari terjadinya gangguan pada hasil klasifikasi [14].

- Data Labelling

  Pada tahap ini, akan dilakukan proses untuk memberi label pada data *tweets*. Proses ini dilakukan menggunakan *library* yang telah ada untuk melakukan *labelling* menjadi sentimen positif atau negatif.

- Casefolding

  Pada tahapan ini dilakukan konversi teks atau kalimat menjadi satu format yang sama, yaitu menggunakan format huruf kecil [14].

- 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 [14].

- Tokenizing

  Pada tahapan ini dilakukan pemecahan kalimat menjadi kata demi kata yang disebut sebagai token [14].

2.5 Decision Tree

Decision Tree merupakan salah satu metode klasifikasi yang cukup populer, hal ini dikarenakan algoritma tersebut mudah untuk diinterpretasikan oleh manusia.

Decision Tree menerapkan model prediksi menggunakan struktur pohon atau disebut dengan struktur hierarki [15]. Konsep dari algoritma ini sendiri ialah mengubah data menjadi sebuah pohon keputusan dan aturan keputusan. Algoritma ini memiliki kemampuan dalam melakukan *break down* proses pengambilan keputusan kompleks menjadi jauh lebih sederhana.

Decision Tree dinamakan pohon keputusan, karena aturan yang terbentuk membentuk sebuah pohon. Data dalam pohon keputusan itu sendiri diinterpretasikan dalam bentuk tabel dengan atribut serta *record*. Pohon Keputusan memiliki *node* yang merepresentasikan atribut dimana setiap cabangnya merupakan hasil uji serta *node* daun (leaf) menggambarkan kelompok dari suatu kelas tertentu [16]. Konstruksi pada *decision tree* terdapat tiga jenis *node*, yakni:

1. Root

   Node teratas dan tidak ada *input*, serta memiliki *output* lebih dari satu.

2. Internal node

   Node percabangan dan hanya terdapat satu *input*, serta memiliki *output* minimal dua.

3. Leaf

   Node terakhir dan hanya terdapat satu *input*, serta tidak memiliki *output*.

Pohon keputusan dimulai dengan cara menghitung nilai *entropy* sebagai penentu tingkat ketidakmurnian atribut dan *information gain* [17].

1. *Entropy* digunakan dalam mengukur keacakan dataset dengan nilai antara 0 dan 1 [18]. Rumus untuk menghitung nilai *entropy* adalah [18]:

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

   Dimana P_i merupakan rasio dari sampel dari subset dan nilai atribut ke-i.

2. *Information Gain* merupakan salah satu metrik yang digunakan dalam proses segmentasi [18]. Rumus untung menghitung nilai *information gain* adalah [18]:

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

   Dimana *range* atribut A adalah $V(A)$ dan S_v merupakan himpunan bagian dari himpunan S yang nilainya sama dengan nilai dri atribut V.

2.6 Random Forest Classifier

Random Forest Classifier merupakan salah satu algoritma yang digunakan dalam melakukan proses klasifikasi data dalam jumlah yang besar. Algoritma ini mengkombinasikan masing-masing pohon dari model *decision tree* yang baik menjadi satu model [5]. Penggunaan pohon yang semakin banyak juga mempengaruhi akurasi menjadi lebih baik. Rangkaian yang tersusun dari pohon dengan masing-masing berisi kumpulan variabel acak.

Random Forest akan menggabungkan masing-masing *tree* dari model *decision tree* menjadi satu model. Banyaknya jumlah *tree* yang dipakai mampu mempengaruhi nilai akurasi, presisi, dan *recall* dari model *Random Forest*. Pemilihan *tree* dari model *decision tree* dimulai dari perhitungan nilai *entropy* dan dicari terbaik untuk dipakai di model *Random Forest* [19].

Langkah-langkah cara kerja algoritma *Random Forest* dapat dijabarkan sebagai berikut:

- Algoritma akan memilih sampel acak dari dataset yang telah disediakan.
- *Decision Tree* akan dibuat untuk setiap sampel yang terpilih. Hasil prediksi akan didapatkan dari setiap *decision tree* yang telah dibuat.
- Proses voting akan dilakukan untuk setiap hasil prediksi. nilai yang paling sering muncul (modus) akan digunakan untuk permasalahan klasifikasi, dan nilai rata-rata *(mean)* akan digunakan untuk permasalahan regresi.
- Algoritma akan memilih hasil prediksi yang paling banyak dipilih (vote terbanyak) sebagai prediksi akhir.

2.7 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF merupakan teknik atau metode yang digunakan untuk memberikan bobot atau nilai terhadap suatu kata [20]. TF-IDF dapat dipakai untuk berbagai tugas seperti ekstrasi token kata dari artikel, menentukan peringkat, dan menghitung tingkat kemiripan antar dokumen. *Text Frequency (TF)* menandakan seberapa banyak sebuah kata muncul dalam satu dokumen [20]. *Inverse Document Frequency (IDF)* menandakan seberapa penting sebuah kata [20].

Rumus untuk melakukan perhitungan TF-IDF dapat dijabarkan sebagai berikut [21]:

$$TF - \mathrm{IDF}_{t,d} = \mathrm{TF}_{t,d} \times iDF_t \tag{2.3}$$

$$IDF_t = log\frac{N}{DF_t} \tag{2.4}$$

$$TF_{t,d} = \frac{n_{i,j}}{\sum_k n_{i,j}} \tag{2.5}$$

Keterangan:

$TF - IDF_{t,d}$ = Kalimat bobot (d) terhadap kata (t)
$TF_{t,d}$ = *Term Frequency*
IDF_t = *Inverse Document Frequency*
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
N = Jumlah kalimat
t = Kata-kata yang dihitung
d = Bobot dari kalimat

2.8 Hyperparameter Tuning

Hyperparameter Tuning merupakan sebuah proses dalam melakukan pencarian kombinasi parameter yang optimal untuk model *machine learning* [22]. Tujuan dari proses ini adalah mencari kombinasi hiperparameter yang paling optimal baik dalam hal kinerja, serta mengurangi resiko terjadinya *overfitting* dan *underfitting* [22]. Pada penelitian kali ini, berikut parameter yang akan dicari kombinasi terbaiknya menggunakan metode *random search*:

1. *n_estimators*: Parameter ini mengacu pada jumlah pohon keputusan yang akan dibentuk pada algoritma. Semakin tinggi nilai n_estimators, semakin banyak pohon yang akan dibentuk. [22]
2. *min_samples_split*: Parameter ini mengacu pada jumlah sampel minimum yang diperlukan dalam melakukan proses pembagian *node internal* baru dalam sebuah pohon. [22]
3. *min_samples_leaf*: Parameter ini mengacu pada jumlah sampel minimum yang harus ada pada *node* daun. [22]

4. *max_depth*: Parameter ini mengacu pada kedalaman maksimum setiap pohon keputusan dalam metode ensembel. [22]

5. *max_features*: Parameter ini mengacu pada cara pengontrolan jumlah fitur yang akan diambil secara acak dalam membangun setiap pohon keputusan. [22]

6. *bootstrap*: Parameter ini mengacu pada pengaturan yang memungkinkan atau mengontrol penggunaan *bootstrap* saat membangun setiap pohon keputusan. Apabila nilai *bootstrap* bernilai *True*, maka setiap pohon dibangun menggunakan sampel *bootstrap*. Apabila bernilai *False*, maka setiap pohon dibangun menggunakan *dataset* pelatihan asli. [22]

2.9 Confusion Matrix

Confusion Matrix merupakan metode yang digunakan dalam melakukan perhitungan nilai akurasi. Metode ini dapat digambarkan dengan sebuah tabel yang menyatakan jumlah data uji yang pengklasifikasiannya benar dan jumlah data uji yang pengklasifikasiannya salah [10].

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Tabel 2.1. Tabel *Confusion Matrix*

Keterangan:

1. TP (True Positive): Jumlah data positif yang diprediksi benar.
2. TN (True Negative): Jumlah data negatif yang diprediksi benar.
3. FP (False Positive): Jumlah data negatif yang diprediksi salah.
4. FN (False Negative): Jumlah data positif yang diprediksi salah.

Terdapat berbagai ukuran yang bisa digunakan dalam mengevaluasi model klasifikasi, yakni sebagai berikut [4]:

- *Accuracy* menyatakan persentase jumlah tuple dalam data uji yang diklasifikasikan dengan benar dengan rumus:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

- *Recall* menyatakan tingkat pengenalan terhadap tupel positif, atau menyatakan seberapa besar tupel positif yang diklasifikasikan dengan benar dengan rumus:

$$recall = \frac{TP}{TP + FN} \tag{2.7}$$

- *Precision* menyatakan persentase tupel yang dilabeli positif adalah benar dengan rumus:

$$precision = \frac{TP}{TP + FP} \tag{2.8}$$

- *f1-score* menyatakan rata-rata harmonik dari *precision* dan *recall* dengan rumus:

$$f1 - score = \frac{2\ x\ Precision\ x\ Recall}{Recall + Precision} \tag{2.9}$$