BAB II
LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen ialah suatu proses yang dilakukan secara otomatis untuk memahami, mengekstrak atau mengolah data teks dengan tujuan untuk mengidentifikasi informasi mengenai sentimen yang terdapat dalam kalimat [6]. Penelitian mengenai analisis sentimen merupakan bagian dari *text mining* yaitu penelitian komputasi berdasarkan sentimen, emoji, pendapat, ulasan, komentar dan segala ekspresi yang diungkapkan melalui teks. Analisis sentimen memiliki dua bagian yaitu dokumen klasifikasi terhadap pendapat atau fakta yang dikenal dengan klasifikasi subjektivitas dan dokumen klasifikasi ke dalam positif, negatif atau netral yang dikenal sebagai analisis sentimen [15]. Secara umum, analisis sentimen digunakan untuk menganalisa tentang suatu ulasan produk, media sosial dan ulasan aplikasi. Dengan memahami sentimen yang terkandung dalam ulasan yang diberikan pengguna, pengembang aplikasi dapat menanggapi ulasan tersebut dan memperbaiki serta meningkatkan kualitas dari aplikasi [7]. Analisis sentimen dibagi menjadi tiga kategori pengelompokan yaitu kategori positif, negatif dan netral berdasarkan teks atau kalimat tersebut [8].

2.2 M-PASPOR

M-PASPOR adalah aplikasi layanan pengurusan paspor yang diterbitkan oleh Direktorat Imigrasi Kemenkumhum Republik Indonesia (Kementerian Hukum dan Hak Asasi Manusia). Aplikasi M-PASPOR bertujuan untuk memberikan kemudahan masyarakat dalam mengajukan dan penggantian paspor secara online [3]. Aplikasi M-PASPOR adalah bentuk baru dari Aplikasi Pendaftaran Antrean Paspor Online (APAPO) yang diterapkan dengan tujuan pelayanan paspor dapat lebih transparan, akuntabel dan cepat. Aplikasi M-Paspor berguna dalam mempermudah pengurusan paspor. Pemohon melengkapi dan upload dokumen persyaratan aplikasi M-PASPOR, Melakukan Pembayaran dan kemudian mendatangi kantor imigrasi pilihan sesuai jadwal untuk wawancara dan perekaman data biometrik [16].

2.3 TF-IDF

Term Frequency - Inverse Document Frequency (TF-IDF) adalah sebuah metode yang berfungsi untuk memberikan nilai bobot untuk teks atau kata yang terdapat didalam kalimat. TF-IDF merupakan metode yang menggabungkan dua konsep dalam perhitungannya, yaitu *term frequency* dan *inversed document frequeny*. *Term Frequency* adalah frekuensi kata dalam dokumen atau kalimat sedangkan, *inversed document frequency* adalah frekuensi dari dokumen yang mengandung kata tersebut. Frekuensi dalam kemunculan suatu kata diberikan untuk menunjukan seberapa penting kata [17]. Rumus dalam melakukan tahap TF-IDF dapat dilihat pada rumus 2.1, 2.2 dan 2.3.

2.3.1 Rumus perhitungan TF

Term Frequency (TF) merupakan pembobotan kata dalam dokumen dengan rumus sebgai berikut:

$$tf(t,d) = \frac{tf}{max(tf)} \tag{2.1}$$

Keterangan:
tf(t,d) = Frekuensi yang terdapat pada term (TF).
max(tf) = Total dari keseluruhan kata yang terdapat dalam dokumen.
tf = Jumlah dari kemunculan term terbanyak. didalam dokumen yang sama.

2.3.2 Rumus perhitungan IDF

inversed document frequency (IDF) adalah frekuensi dari dokumen yang mengandung kata dengan rumus sebagai berikut:

$$idf_t = \log(\frac{D}{df_t}) \tag{2.2}$$

Keterangan:
D = Total kesuluruhan dari dokumen.
idf(t) = bobot pada kemunculan term t pada setiap dokumen.
df_t = jumlah dari dokumen yang mengandung term t

2.3.3 Rumus perhitungan TF-IDF

$$W_{t,d} = tf(t,d) \times idf_t \tag{2.3}$$

Keterangan:
$W_{t,d}$ = bobot term dalam suatu dokumen
idf(t) = bobot dari kemunculan term t pada setiap dokumen.
tf(t,d) = Frekuensi term (TF).

2.3.4 *Naive Bayes*

Naive bayes merupakan algoritma yang populer dalam klasifikasi dengan teknik *machine learning*. Algoritma *Naive bayes* sangat berguna untuk digunakan dengan data yang berdimensi tinggi seperti data teks. Secara umum, pemodelan bayes terdiri dari model struktural dan kondisional antara variabel seacara acak [9]. Rumus 2.4 merupakan rumus untuk menghitung *Naive Bayes Classifier* [18].

$$P(c_j|w_i) = \frac{P(w_i|c_j) \times P(c_j)}{P(w_i)} \tag{2.4}$$

Keterangan:
$P(c_j|w_i)$ = Peluang dari kateogri j yang terdapat pada kemunculan kata i.
$P(w_i|c_j)$ = Peluang dari kata i yang terdapat pada kategori j.
$P(c_j)$ = Peluang dari kategori j.
$P(w_i)$ = peluang dari kemunculan kata.

Pada Klasifikasi kemunculan kata dapat dihilangkan dalam proses perhitungan. Oleh karena itu, dapat disederhanakan dengan persamaan 2.5 [18].

$$P(c_j|w_i) = (P(c_j)(P(w_i|c_j) \tag{2.5}$$

Dalam proses klasifikasi, terdapat prior yang berguna untuk menghitung peluang kemunculan kategori di setiap dokumen. Perhitungan prior dapat dilihat pada persamaan 2.6 [18].

$$P(c_j) = \frac{N_c}{N} \tag{2.6}$$

Keterangan:
N_c = Banyak dari dokumen berkategori c_j yang terdapat pada dokumen latih

N = Jumlah dari semua dokumen data latih yang digunakan.

A *Multinomial Naive Bayes*

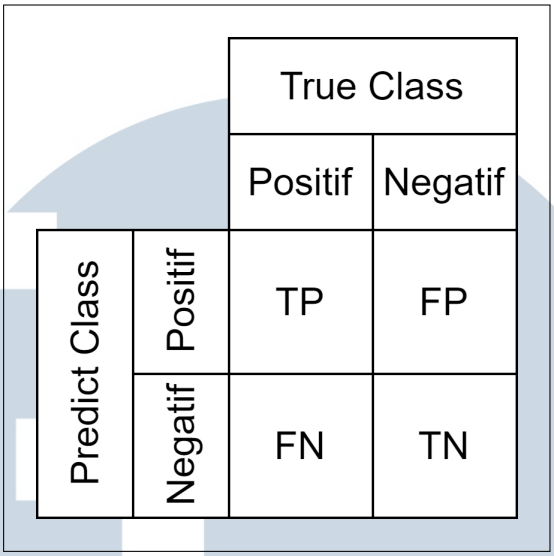
Multinomial Naive Bayes merupakan varian dari algoritma *Naive Bayes*. *Multinomial Naive Bayes* adalah model klasifikasi teks yang berbasis frekuensi yang diwakili oleh kumpulan kata yang muncul dalam data atau dokumen [9]. Algoritma *multinomial naive bayes* memiliki beberapa keunggulan yaitu, algoritma ini mudah digunakan pada data yang bersifat kontinu dan diskrit, Dapat menangani kumpulan data yang besar, dapat digunakan untuk mengklasifikasikan atau mengelompokan data dengan banyak label dan paling baik digunakan untuk melatih model *natural language processing* [10]. Dalam melakukan perhitungan *Multinomial Naive Bayes* dapat melakukan perhitungan peluang dengan persamaan yang terdapat pada rumus 2.7 [18].

$$P(w_i|c_j) = \frac{count(w_i|c_j) + 1}{\sum_{weV} count(w_i|c_j) + |V|} \tag{2.7}$$

Keterangan:
$count(w_i|c_j)$ = Jumlah kemunculan kata kunci dalam kelas atau kategori.
$\sum_{weV} count(w_i|c_j)$ = Jumlah dari semua kata yang terdapat pada kelas atau kategori. c_j.
$|V|$ = Jumlah dari semua kata unik yang terdapat pada setiap kategori.

2.4 *Confusion Matrix*

Confusion Matrix adalah alat pengukuran kinerja untuk klasifikasi model pembelajaran mesin, yang memungkinkan klasifikasi dapat berupa beberapa kelas. Dalam *confusion matrix* memiliki empat kombinasi dari nilai aktual atau nilai sebenarnya dan nilai prediksi yaitu hasil dari pengklasifikasian menggunakan algoritma dalam *machine learning*. *Confusion Matrix* menggunakan empat istilah yang digunakan dalam merepresentasikan hasil proses klasifikasi yaitu *true positif(tp), true negatif(tn), false positif(fp) dan false negatif(fn)*. Gambar 2.1 merupakan bentuk dari *confusion matrix* [19].



(Gambar 2.1. Confusion Matrix)

(Sumber: binus.ac.id, 2020)

Confusion Matrix memiliki beberapa rumus dalam menghitung performa dalam klasifikasi model *machine learning* yaitu;

2.4.1 *Accuracy*

Accuracy dari *confusion matrix* memiliki fungsi untuk menggambarkan seberapa besar akurasi model klasifikasi *machine learning* yang telah dirancang dan dibangun. Untuk menghitung *accuracy* dalam *confusion matrix* dapat diliat pada rumus 2.8.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.8}$$

2.4.2 *Precision*

Precision dapat dihitung menggunakan congusion matrix yang berfungsi untuk menggambarkan akurasi antar data dengan hasil klasifikasi pemodelan machine learning. Untuk menghitung *Precision* dalam *confusion matrix* dapat diliat pada rumus 2.9.

$$Precision = \frac{TP}{TP + FP} \tag{2.9}$$

2.4.3 *Recall*

Recall merupakan rumus yang berfungsi sebagai penggambaran dari keberhasilan model yang telah diterapkan dalam menemukan kembali sebuah informasi. Untuk menghitung *Recall* dalam *confusion matrix* dapat diliat pada rumus 2.10.

$$Recall = \frac{TP}{TP + FN} \tag{2.10}$$

2.4.4 *F1-score*

F1-score merupakan rumus yang berguna untuk perbandingan rata-rata dari hasil *precision* dan *recal* yang telah dibobotkan. ntuk menghitung *F1-score* dalam *confusion matrix* dapat diliat pada rumus 2.11.

$$F1 - Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.11}$$

2.5 *Textblob*

Textblob merupakan *library* pada bahasa pemrograman *python* yang digunakan sebagai pemroses data tekstual. *Textblob* meyediakan API yang dapat digunakan untuk pemrosesan bahasa alami atau *Natural Language Processing (NLP)* seperti analisis sentimen, klasifikasi data, penerjemahan dan lain-lain [20]. Dalam pelabelan menggunakan *library textblob* jika skor polaritas > 0 adalah positif, skor polaritas < 0 adalah negatif dan skor polaritas $= 0$ adalah netral [21].

2.6 *Text-preprocessing*

Text-preprocessing merupakan tahapan yang memiliki fungsi untuk menyeleksi teks menjadi terstruktur melalui serangkaian tahapan [22]. Dalam penelitian ini *ext-prepprocessing* memiliki beberapa tahapan yaitu;

1. *Case Folding*
   Case Folding merupakan tahapan dalam *text-preprocessing* yang memiliki fungsi untuk mengubah teks menjadi huruf kecil atau *lowercase*.

2. *Cleaning Data*

   Menyeleksi dan membersihkan data teks dari *symbol, emoticon, link, single char, number, whites pace dan punctuation* yang tidak memiliki sentimen dalam suatu teks yang terdapat dalam kalimat atau dokumen.

3. *Tokenizing*

   Tahapan *Tokenizing* merupakan tahapan dalam *text-preprocessing* yang memiliki fungsi sebagai pemisah kalimat pada data menjadi daftar kata. Tahapan ini dilakukan karena untuk mengimplementasikan tahapan selanjutnya dalam *text-preprocessing* yaitu; *Normalisasi, Stopwords* dan *Stemming*.

4. *Normalisasi*

   Normalisasi berfungsi untuk memperbaiki salah penulisan kata dalam teks atau typo. Contoh dari penggunaan *normalisasi* yaitu kata ”mrpkn” menjadi kata ”merupakan”.

5. *Stopwords*

   stopwords adalah tahapan dalam *text-preprocessing* yang memiliki fungsi untuk memilih kata-kata yang dianggap penting dalam sebuah teks dan menghilangkan kata-kata dan simbol yang tidak bermakna dalam teks seperti konjungsi.

6. *Stemming*

   Stemming merupakan tahapan akhir dari melakukan tahapan *text-preprocessing* yang memiliki fungsi untuk mengubah kata yang memiliki imbuhan menjadi bentuk kata dasar. Contoh penggunaan *stemming* dalam teks adalah kata ”melakukan” dirubah menjadi kata dasarnya yaitu ”lakukan”.

Dari hasil penelitian yang telah dilakukan oleh Sheila Shevira bersama mitranya menyatakan bahwa melakukan tahapan normalisasi sebelum tahapan stopword dan stemming merupakan tahapan terbaik dalam *text-preprocessing* [23].