BAB II

LANDASAN TEORI

2.1 User Feedback

User Feedback atau ulasan pengguna merupakan informasi yang diberikan pengguna tentang apakah pengguna merasa puas atau tidak dengan produk atau jasa yang diberikan (Ceban, 2018).

2.2 Tokenizing

Tokenizing adalah proses pemotongan string masukan berdasarkan tiap kata yang menyusunnya. Pada prinsipnya proses ini adalah memisahkan setiap kata yang menyusun suatu dokumen (Echa, 2012).

2.3 Filtering

Filtering adalah proses pembuangan kata-kata penghubung ataupun kata yang dianggap tidak berpengaruh pada penelitian, seperti aku, kamu, dan. *Stoplist* atau *Stopwords* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan kamus kata. Terdapat banyak kumpulan *stop words* yang disediakan di internet dalam berbagai bahasa.

2.4 Stemming

Stemming adalah proses merubah kata menjadi kata dasar. Pada proses ini, setiap kata akan diubah menjadi kata dasarnya dengan menerapkan aturan-aturan tertentu (Heidenrich, 2018). Contohnya kata menari menjadi tari.

2.5 TF / IDF

TF / IDF adalah suatu algoritma yang berdasarkan nilai statistik menunjukkan kemunculan suatu kata di dalam dokumen (Wijaya & Santoso, 2016). TF (*Term Frequency)* menyatakan banyaknya suatu kata muncul dalam sebuah dokumen. IDF (*Inverse Document Frequency)* menyatakan banyaknya dokumen yang mengandung suatu kata dalam satu segmen publikasi (Kalokasari, et al., 2017).

$$W_{ij} = tf_{ij}\ x\ idf_j \quad \text{...(2.1)}$$

$$W_{ij} = tf_{ij}\ x \log\left(\frac{D}{df_j}\right) \quad \text{...(2.2)}$$

Dimana Wij merupakan bobot kata *i* pada dokumen *j,* sementara D adalah jumlah dokumen, dan term frequency yaitu tf_{ij} adalah jumlah dari kemunculan kata *i* pada dokumen *j*, idf_j adalah jumlah dokumen *j* yang berisi kata i (Wijaya & Santoso, 2016).

2.6 Naive Bayes Classifier

Naive Bayes Classifier adalah sebuah algoritma klasifikasi yang mengacu pada Teorema Bayes. Pada persamaan 3, dapat dijabarkan P(A|B) adalah peluang kejadian A bila B terjadi. Sedangkan P(B|A) adalah peluang kejadian B apabila A terjadi. *Naive Bayes Classifier* menganggap bahwa setiap kata pada kalimat adalah suatu individu yang independen dan tidak terikat dengan yang lain (Gandhi, 2018).

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \text{...(2.3)}$$

2.7 Multinomial Naive Bayes

Sedangkan *multinomial naive bayes* sama seperti *naive bayes,* namun *multinomial naive bayes* memperhitungkan distribusi tiap fitur dalam dokumen sehingga mendapatkan akurasi yang lebih baik (Huang, 2017).

$$Cmap = \arg \max_{C \in \{Cl,Cs\}} P(c) \prod_{k=1}^{m} P(t_k \mid c) \quad \text{...(2.4)}$$

Parameter $P(t_k \mid c)$ atau probability likelihood diestimasikan dengan menghitung kejadian t_k pada semua dokumen training di c, menggunakan Laplacean Prior (Kalokasari, et al., 2017) :

$$P(t_k \mid c) = \frac{1+N_k}{|V|+N} \quad \text{...(2.5)}$$

Dimana N_k adalah jumlah kemunculan t_k dalam dokumen pelatihan di c dan N adalah jumlah total kemunculan kata dalam c (Kalokasari, et al., 2017). Contoh kasusnya, apabila didapat data latih seperti berikut :

Tabel 2.1 Data latih (Stecanella, 2017)

Teks	Kategori
Sebuah permainan yang baik	Sports
Pemilihan presiden sudah selesai	Not Sports
Sebuah permainan yang bersih	Sports
Permainan yang bersih namun tidak terlalu bagus	Sports
Itu merupakan pemilihan presiden yang seimbang	Not Sports

Lalu, akan dilakukan klasifikasi terhadap kalimat "Permainan yang seimbang". Dimana target kelasnya adalah Sports atau Not Sports. Maka selanjutnya akan dilakukan perhitungan posterior probabilitasnya (Stecanella, 2017).

$$P(Permainan\ yang\ seimbang \mid sports) = P(permainan|sports)x\ P(yang|sports)x\ P(seimbang|sports) \quad \text{...(2.6)}$$

$$P(Permainan\ yang\ seimbang \mid not\ sports) = P(Permainan|not\ sports)\ x\ P(yang|not\ sports)\ x\ P(seimbang|not\ sports) \quad \text{...(2.7)}$$

Namun, karena seluruh hasil akan dilakukan perkalian, apabila ada salah satu kata yang memiliki nilai 0 maka seluruh kalimat akan bernilai 0. Oleh karena itu dapat dilakukan *Laplace Smoothing* dengan cara menambahkan nilai 1 pada setiap kata (Stecanella, 2017).

Tabel 2.2 Perhitungan posterior probabilitas (Stecanella, 2017)

Kata	**P(kata| sports)**	**P(kata | not sports)**
Permainan	$\frac{3+1}{14+16}$	$\frac{0+1}{10+16}$
yang	$\frac{3+1}{14+16}$	$\frac{0+1}{10+16}$
seimbang	$\frac{0+1}{14+16}$	$\frac{1+1}{10+16}$

Langkah terakhir yang perlu dilakukan adalah menghitung posterior probabilitasnya (Stecanella, 2017).

$$P(Permainan|sports)\ x\ P(yang|sports) x\ P(seimbang|sports) =$$

$$\frac{3+1}{14+16}\ x\ \frac{3+1}{14+16}\ x\ \frac{0+1}{14+16} = 0.000593 \quad \text{...(2.8)}$$

$$P(Permainan|not\ sports)\ x\ P(yang|not\ sports)$$

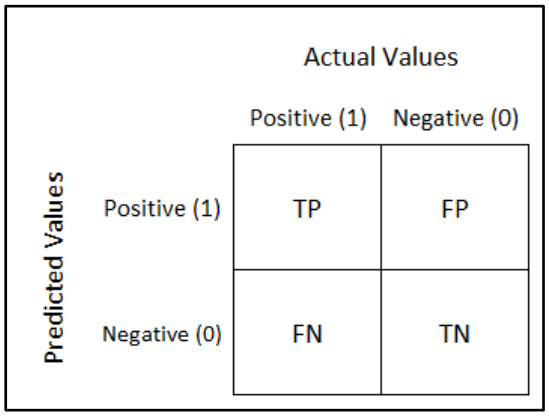
$$x\ P(seimbang|not\ sports) = \frac{0+1}{10+16}\ x\ \frac{0+1}{10+16}$$

$$x\ \frac{1+1}{10+16} = 0.000113 \quad \text{...(2.9)}$$

Dari perhitungan diatas, didapatkan bahwa posterior probabilitas pada kategori *Sports* lebih tinggi, sehingga kalimat "Permainan yang seimbang" akan masuk kedalam kelas *Sports*.

2.8 Confusion Matrix

Confusion Matrix adalah sebuah metode untuk mengukur performa dari klasifikasi pembelajaran mesin dimana klasifikasi tersebut memiliki keluaran berupa 2 atau lebih kelas. *Confusion Matrix* berbentuk sebuah tabel dengan 4 kombinasi berbeda dari *predicted* dan *actual value* (Narkhede, 2018).



Gambar 2.1 Confusion Matrix (Narkhede, 2018)

Hasil dari implementasi algoritma *Multinomial Naive Bayes* akan dievaluasi menggunakan *Confusion Matrix* tersebut. TP berarti *True Positive*, dimana sistem memprediksi positif dan hasilnya benar. TN berarti *True Negative*, dimana sistem memprediksi negatif dan hasilnya benar. Sedangkan FP berarti *False Positive*, yaitu saat sistem memprediksi positif namun hasilnya tidak terjadi positif. Lalu yang terakhir yaitu FN berarti *False Negative*, dimana sistem memprediksi negatif namun hasilnya tidak terjadi sesuai yang diprediksi (Narkhede, 2018). Dari *confusion matrix* yang telah dibuat, dapat dihitung nilai *accuracy, precision, recall,* dan *F1-Score.* Nilai *accuracy* digunakan untuk mengukur ketepatan prediksi yang dilakukan oleh *classifier*. Nilai *precision* menghitung perbandingan prediksi positif yang benar (*True Positive*) terhadap seluruh prediksi positif. Sedangkan nilai *recall* merupakan nilai yang mengukur perbandingan prediksi *true positive* terhadap seluruh prediksi di kelas aktual. *F1-Score* merupakan suatu nilai yang mengukur nilai harmoni diantara kedua nilai *precision* dan *recall*. Perhitungan nilai *accuracy, precision, recall,* dan *F1-Score* dapat dilihat pada Persamaan 2.10, 2.11, 2.12, dan 2.13.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \ldots(2.10)$$

$$Precision = \frac{TP}{(TP+FP)} \quad \ldots(2.11)$$

$$Recall = \frac{TP}{(TP+FN)} \quad \ldots(2.12)$$

$$F1 - Score = \frac{2\ x\ Precision\ x\ Recall}{Precision+Recall} \quad \ldots(2.13)$$