Analisis Sentimen Pada Dataset *tweets*: Perbandingan Akurasi Model Machine Learning

Garret Junior Tanzil[1*], Brithney Elizabeth Tania[2]

[1*]Program Studi Teknik Informatika, Universitas Surabaya, Surabaya, Jawa Timur
[2] Program Studi Teknik Informatika, Universitas Surabaya, Surabaya, Jawa Timur
Email: [1*]s160421097@ubaya.ac.id, s160421148@ubaya.ac.id,

(Naskah masuk: dd mmm yyyy, direvisi: dd mmm yyyy, diterima: dd mmm yyyy)

Abstrak

Penelitian ini bertujuan untuk melakukan perbandingan akurasi berbagai model *machine learning* termasuk *Naïve Bayes*, *Random Forest*, *k-Nearest Neighbors (KNN*), dan *Logistic Regression*. Tujuan pemilihan model-model tersebut dikarenakan kemampuan untuk melakukan prediksi klasifikasi data yang lebih akurat. Dataset yang digunakan terdiri dari *tweet-tweet* yang telah diambil pada tahun 2015 yang berisi pendapat individu mengenai pendapat mereka terhadap berbagai macam perusahaan penerbangan asal Amerika Serikat dari platform media sosial *Twitter* yang sekarang diketahui sebagai *X*, dalam bahasa Inggris dengan label sentimen positif, netral, dan negatif. Pada tahap eksperimen, penulis melakukan *data pre-processing*, yang melibatkan *text sanitation* dan vektorisasi data menggunakan *Term Frequency – Inverse Document Frequency* (TF-IDF). Setelahnya, penulis melatih masing-masing model dan mengevaluasi hasilnya dengan menggunakan metrik seperti akurasi, presisi, *recall*, dan *F1-score*. Hasil eksperimen menunjukkan bahwa model *Logistic Regression* memiliki akurasi yang terbaik yakni 81%, dimana model yang lain memiliki akurasi di rentang 72% hingga 77%. Kesimpulan dan analisis dari penelitian ini dapat digunakan sebagai panduan dalam memilih model untuk melakukan analisis sentimen terhadap data *tweet* dalam konteks bahasa Inggris.

Kata Kunci: *machine learning,* analisis sentimen, prediksi, akurasi, klasifikasi

Sentiment Analysis on Tweets Dataset: Comparing Machine Learning Model Accuracy

Abstract

This research aims to compare the accuracy of different learning models-including Naïve Bayes, Random Forest, k-Nearest Neighbors (KNN) and Logistic Regression. The reasoning of the selection of these models being the better capability of predicting classification data in a more accurate manner. The dataset that is used in this research consists of tweets taken from 2015 containing individual's opinion on various airlines from the United States from the social media platform Twitter, now known as X, written in English with each of them having sentiment labels- positive, neutra and negative. In the experimentation phase, the writers uses data pre-processing, applying text sanitation and vectorization by using Term Frequency – Inverse Document Frequency (TF-IDF). Afterwards, the writers trained each model and evaluated the results using metrics such as accuracy, precision, recall and F1-score. The results of this experiment showed that the Logistic Regression model had a higher accuracy than the rest at 81%, while the other models are at 72% and up to 77% accuracy. The conclusion and analysis of this research can be used as a guide when selecting the model to use when doing sentiment analysis towards tweets that are written in English.

Keywords: *machine learning, sentiment analysis, prediction, accuracy, classification*

I. PENDAHULUAN

Dalam era globalisasi ini, perkembangan teknologi informasi dan komunikasi mempercepat arus penyampaian dan penerimaan informasi di seluruh dunia. Salah satu contoh globalisasi ini yakni media social, yang telah menjadi platform utama bagi individu untuk berbagi dan menggunggah berbagai macam hal-seperti pendapat mengenai suatu topik. *Twitter* atau *X* telah menjadi wadah bagi pengguna untuk menyampaikan pikiran mereka dalam format singkat melalui postingan yang dikenal sebagai *"tweet"*. Dengan menggunakan *sentiment analysis*, pola dan pandangan emosi yang terkandung dalam pikiran public dapat teridentifikasi.

Penelitian ini bertujuan untuk melakukan analisis sentiment terhadap dataset tweet dari tahun 2015 mengenai pendapat pengguna *Twitter* atau *X* terhadap berbagai maskapai penerbangan asal Amerika Serikat dengan focus pada pembandingan akurasi terhadap beberapa model *machine learning* yang umum digunakan untuk klasifikasi. Dengan memahami *sentiment* yang ada di sebuah *tweet* dan melakukan analisis terhadap model terbaik untuk melakukan prediksi, peneliti dapat mendapatkan wawasan yang berharga terkait repons pengguna terhadap suatu topik maupun peristiwa.

Melalui perbandingan akurasi model *machine learning*, penelitian ini diharapkan dapat memberikan panduan kepada penelitian kedepannya dalam memilih pendekatan model yang paling akurat dan efektif untuk menganalisis sentimen pada data *tweet* berbahasa Inggris, yang bisa digenerelasiskan terhadap topik yang lebih luas.

II. TINJAUAN PUSTAKA

A. *Sentiment Analysis*

Sentiment Analysis merupakan metode untuk mengidentifikasi apakah suatu teks yang biasanya berupa unggapan orang mengandung sentiment yang *positif, negative*, maupun *neutral*. Hal ini dapat membantu dalam memahami respon massa mengenai suatu topik tertentu, yang telah dimudahkan dengan adanya media sosial yang memudahkan akses kepada opini maupun tanggapan publik [2].

Pemahaman ini kemudian dapat digunakan oleh perusahaan untuk pengambilan keputusan dan untuk fungsi pemasaran yang sesuai dengan tanggapan asli dari *target market*. Tren yang sedang terjadi juga tampak dengan adanya opini pasar yang telah diproses dan disajikan dengan hasil klasifikasi [3].

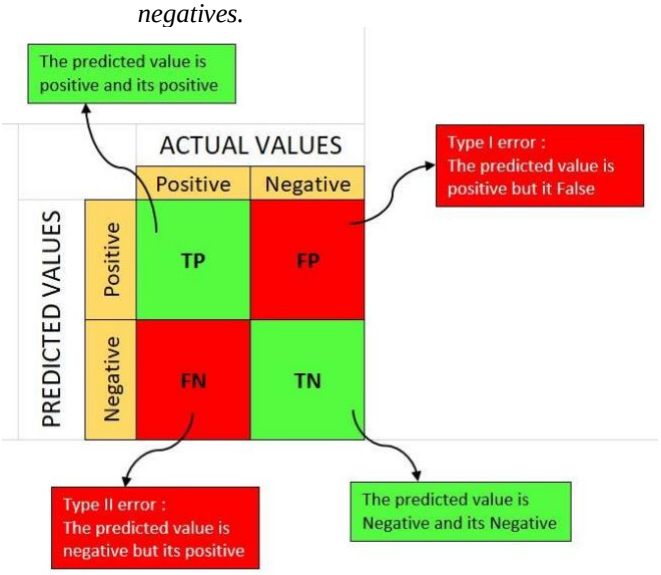
B. *Model Evaluation*

Metode untuk mengevaluasi performa dari suatu model adalah menggunakan *Model Evaluation* seperti fungsi *Accuracy Score, Classification Report, Confusion Matrix* yang telah disediakan oleh *library sklearn*.

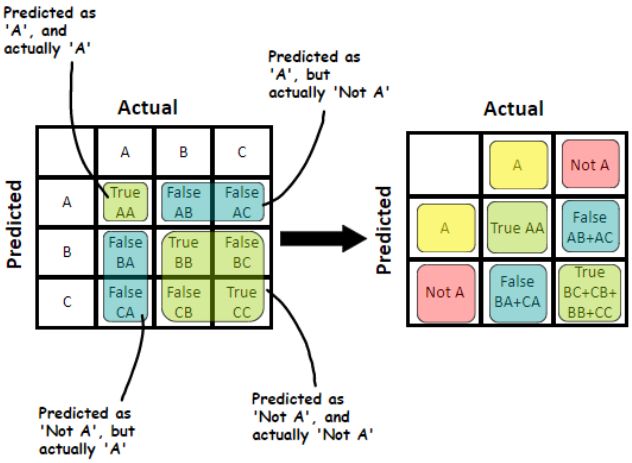
i) *Accuracy Score*
Menggunakan *functon* langsung dari *sklearn* yang menampilkan akurasi prediksi model secara keseluruhan dalam persen.

ii) *Classification Report*
Precision – kemampuan sebuah model klasifikasi untuk mengidentifikasi hanya data poin yang relevan [4]
Recall – metrik untuk mengukur performa model untuk mendapatkan semua *case* relevan dari sebuah data set [4].
F1-score – mengukur akurasi suatu model dengan menggabungkan nilai *precision* dan *recall* suatu model [5].
Support – merupakan jumlah kemunculan suatu class di dataset tertentu [6].

iii) *Confusion Matrix*
Merupakan alat evaluasi performa suati *model machine learning*, yang menunjukkan *true positives, true negatives, false positives*, dan *false negatives.*



Gambar 1. Penjelasan tampilan output *Confusion Matrix binary classification* [17]



Gambar 2. Penjelasan tampilan output *Confusion Matrix multiclass classification* [18]

C. *Naïve Bayes Classifier*

Salah satu model *machine learning* yang seringkali digunakan adalah *Naïve Bayes Classifier*. Seiring dengan namanya, model tersebut cocok digunakan untuk masalah klasifikasi.

Alasan mengapa performa *Naïve Bayes Classifier* relatif akurat meskipun dengan asumsi independensi bersyarat, yang menjadi dasar Naive Bayes, yang jarang ada dalam aplikasi dunia nyata, kinerjanya yang baik dalam klasifikasi masih relatif efektif [7].

D. *K-nearest Neighbor (KNN)*

K-Nearest Neighbor adalah metode klasifikasi yang sangat umum dan relatif simple. Model *machine learning* ini digunakan saat terdapat sedikit atau bahkan tidak ada pengetahuan sebelumnya mengenai distribusi data yang digunakan [8].

E. *Logistic Regression*

Logistic Regression merupakan model yang digunakan untuk klasifikasi data ke dalam kelompok atau grup yang berbeda [9].

Walaupun *Logistic Regression* umumnya digunakan untuk dataset yang bersifat *bivariate*, model tersebut dapat di adaptasikan terhadap dataset yang bersifat *multivariate* [9].

F. *Random Forest*

Algoritma *Random Forest* adalah suatu model yang dapat diterapkan di berbagai macam permasalahan, seperti *Network Intrusion Detection*, *Email Spam Detection*, dan *Text Classification [11]*.

Salah satu kekuatan *Random Forest Classifier* adalah kemampuannya untuk memproses jumlah input variabel yang sangat tinggi tanpa *overfitting [12]*.

III. BAHAN PENELITIAN

A. Dataset

Dataset yang digunakan untuk melatih model yang berupa 14640 observasi berupa *tweet* pengguna *Twitter* atau *X* yang mengandung segala hal mengenai 6 maskapai penerbangan yang berbeda asal Amerika Serikat yang tersedia di *Kaggle* [1]. Fitur yang di observasi di dataset tersebut mencakup *tweet_id*, *airline_sentiment*, *airline_sentiment_confidence*, *negativereason*, *negativereason_confidence*, *airline*, *airline_sentiment_gold*, *name*, *negativereason_gold*, *retweet_count*, *text*-tetapi fitur yang digunakan untuk penelitian ini adalah airline_sentiment yang berisi *sentiment* dari masing-masing *tweet* yakni positif, netral, maupun negative beserta *text* yang mencakup isi dari masing-masing *tweet* tersebut.

B. Perangkat Keras dan Perangkat Lunak

Perangkat keras yang digunakan untuk percobaan ini adalah laptop *Macbook Air M1 Apple Silicon Chip*, dengan penggunaan perangkat lunak *Google Collab* untuk menulis dan menjalankan kode berbasis *cloud* dan *Microsoft Word* untuk penulisan laporan ini. Bahasa pemrograman yang digunakan adalah *Python* dengan menggunakan *import google.colab*, *pandas*, dan *sklearn.*

IV. METODOLOGI

A. Pengambilan dataset

Karena menggunakan perangkat lunak coding berbasis *cloud*, dataset dapat terbaca oleh *Google Collab* dengan kode berikut:

```
1 from google.colab import drive
2 drive.mount('/content/drive')
3 import pandas as pd
4
5 tweet_df =
pd.read_csv('/content/drive/MyDrive/Twee
ts.csv', encoding="ISO-8859-1")
6
7 print(tweet_df)
```

Cuplikan kode tersebut difungsikan untuk mengambil dataset yang penulis sudah letakkan di *Google Drive* dengan penggunaan "*import drive*" pada baris 1 dan 2. Dataset tersebut kemudian dimasukkan kepada *dataframe* dengan penggunaan "*import pandas*" kepada *dataframe* "*tweet_df*" yang kemudian di tampilkan untuk menkonfirmasi bahwa dataset tersebut sudah terpanggil dan tersimpan dengan benar.

B. *Naïve Bayes Classifier*

Klasifikasi *Naïve Bayes* adalah klasifikasi probabilitas sederhana yang didasarkan pada Teorema Bayes. Kata "naïve" merujuk pada asumsi kemandirian kondisional (kuat) yang naif antara fitur-fitur.

Salah satu alasan mengapa model *Naïve Bayes* efisien adalah bahwa mereka mempelajari parameter dengan memeriksa setiap fitur secara individual dan mengumpulkan statistik sederhana per kelas dari setiap fitur [13].

Berikut merupakan cuplikan kode yang digunakan untuk melakukan *training* dan evaluasi model *Naïve Bayes*:

```
1 import pandas as pd
2 from sklearn.model_selection import
train_test_split
3 from sklearn.feature_extraction.text
import TfidfVectorizer
4 from sklearn.naive_bayes import
MultinomialNB
5 from sklearn.metrics import
accuracy_score, classification_report,
confusion_matrix
6
7 df = pd.read_csv
```

```
('/content/drive/MyDrive/Tweets.csv',
encoding="ISO-8859-1")
8
9 X = df['text']
10 y = df['airline_sentiment']
11
12 X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
13
14 vectorizer =
TfidfVectorizer(max_features=5000)
15 X_train_tfidf =
vectorizer.fit_transform(X_train)
16 X_test_tfidf =
vectorizer.transform(X_test)
17
18 model = MultinomialNB()
19 model.fit(X_train_tfidf, y_train)
20
21 y_pred = model.predict(X_test_tfidf)
22
23 accuracy = accuracy_score(y_test,
y_pred)
24 print(f"Accuracy: {accuracy:.2f}")
25
26 print("Classification Report:")
27 print(classification_report(y_test,
y_pred))
28
29 print("Confusion Matrix:")
30 print(confusion_matrix(y_test,
y_pred))
```

Baris 1 hingga 5 berisi *library* yang akan digunakan saat menggunakan model *Naïve Bayes*.

Baris 7 berfungsi untuk menyimpan dataset *tweets* yang digunakan kepada variable *dataframe "df"*.

Baris 9 dan 10 berfungsi untuk mengambil seluruh observasi dari kolom *"text"* dan *"airline_sentiment"* dan menyimpannya ke dalam variabel *X* dan *Y* secara berurutan

Baris 12 bertujuan untuk membagi dataset untuk *training* dan *testing*, dalam percobaan ini, penulis menggunakan *test size* sebesar 0.2 (20%) dan *training size* sebesar 0.8 (80%). *Random state* yang digunakan adalah 42, yang merupakan sebuah angka acak yang dapat digunakan untuk mereplikasi ulang percobaan ini dengan menggunakan "*random seed"* yang sama setiap kalinya, yang akan menghasilkan hasil yang sama.

Baris 14 hingga 16 merupakan tahap *pre-processing* yang menggunakan *TF-IDF (Term Frequency – Inverse Document Frequency)* untuk mengubah data yang berupa teks di kolom "*text*" dan mentransformasikannya ke bentuk numerik hingga dapat di proses oleh model *machine learning* tersebut [2]. Parameter *max_features* yang bernilai 5000 berarti hanya 5000 kata atau *term* yang paling relevan yang akan digunakan untuk melatih model.

Baris 18 dan 19 bertujuan untuk melatih model *Naïve Bayes* dengan menggunakan data *train*.

Baris 21 melakukan prediksi data dengan menggunakan model yang sudah terlatih tersebut, dengan parameter *test* data.

Baris 23 bertujuan untuk menghitung akurasi dari prediksi yang telah dilakukan model yang sudah dilatih tersebut.

Baris 24, 26 hingga 27, dan 29 hingga 30 bertujuan untuk menampilkan akurasi, *Classification Report*, dan *Confusion Matrix* dari hasil percobaan model tersebut secara berurutan.

C. *K-Nearest Neighbor (KNN) Classifier*

K-Nearest Neighbors (KNN) adalah model klasifikasi yang termasuk sederhana. *KNN* menggunakan kemiripan dengan tetangga terdekat. *KNN* memprediksi kelas dari observasi yang tidak diketahui berdasarkan kelas mayoritas dari k tetangga terdekatnya. *KNN* menggunakan jarak untuk mengukur kemiripan antara observasi, di mana semakin kecil jaraknya, semakin mirip observasinya [14].

Berikut merupakan cuplikan kode yang digunakan untuk melakukan *training* dan evaluasi model *K-Nearest Neighbor*:

```
1 import pandas as pd
2 from sklearn.model_selection import
train_test_split
3 from sklearn.feature_extraction.text
import TfidfVectorizer
4 from sklearn.neighbors import
KNeighborsClassifier
5 from sklearn.metrics import
accuracy_score, classification_report,
confusion_matrix
6
7 df =
pd.read_csv('/content/drive/MyDrive/Twee
ts.csv', encoding="ISO-8859-1")
8
9 X = df['text']
10 y = df['airline_sentiment']
11
12 X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
13
14 vectorizer =
TfidfVectorizer(max_features=5000)
```

```
15 X_train_tfidf =
vectorizer.fit_transform(X_train)
16 X_test_tfidf =
vectorizer.transform(X_test)
17
18 knn_model =
KNeighborsClassifier(n_neighbors=5)
19 knn_model.fit(X_train_tfidf, y_train)
20
21 y_pred =
knn_model.predict(X_test_tfidf)
22
23 accuracy = accuracy_score(y_test,
y_pred)
24 print(f"Accuracy: {accuracy:.2f}")
25
26 print("Classification Report:")
27 print(classification_report(y_test,
y_pred))
28
29 print("Confusion Matrix:")
30 print(confusion_matrix(y_test,
y_pred))
```

Baris 1 hingga 5 berisi *library* yang akan digunakan saat menggunakan model *K-Nearest Neighbor (KNN).*

Baris 7 berfungsi untuk menyimpan dataset *tweets* yang digunakan kepada variable *dataframe "df"*.

Baris 9 dan 10 berfungsi untuk mengambil seluruh observasi dari kolom *"text"* dan *"airline_sentiment"* dan menyimpannya ke dalam variabel *X* dan *Y* secara berurutan

Baris 12 bertujuan untuk membagi dataset untuk *training* dan *testing*, dalam percobaan ini, penulis menggunakan *test size* sebesar 0.2 (20%) dan *training size* sebesar 0.8 (80%). *Random state* yang digunakan adalah 42, yang merupakan sebuah angka acak yang dapat digunakan untuk mereplikasi ulang percobaan ini dengan menggunakan "*random seed"* yang sama setiap kalinya, yang akan menghasilkan hasil yang sama.

Baris 14 hingga 16 merupakan tahap *pre-processing* yang menggunakan *TF-IDF (Term Frequency – Inverse Document Frequency)* untuk mengubah data yang berupa teks di kolom "*text*" dan mentransformasikannya ke bentuk numerik hingga dapat di proses oleh model *machine learning* tersebut [2]. Parameter *max_features* yang bernilai 5000 berarti hanya 5000 kata atau *term* yang paling relevan yang akan digunakan untuk melatih model.

Baris 18 dan 19 bertujuan untuk melatih model *K-Nearest Neighbor (KNN)* dengan menggunakan data *train*.

Baris 21 melakukan prediksi data dengan menggunakan model yang sudah terlatih tersebut, dengan parameter *test* data.

Baris 23 bertujuan untuk menghitung akurasi dari prediksi yang telah dilakukan model yang sudah dilatih tersebut.

Baris 24, 26 hingga 27, dan 29 hingga 30 bertujuan untuk menampilkan akurasi, *Classification Report*, dan *Confusion Matrix* dari hasil percobaan model tersebut secara berurutan.

D. *Logistic Regression*

Banyak model klasifikasi *linear* hanya digunakan untuk klasifikasi biner, dan tidak secara alami diperluas ke kasus *multiclass*, dengan pengecualian regresi logistik. Teknik umum untuk memperluas algoritma klasifikasi biner ke algoritma klasifikasi *multiclass* adalah pendekatan *one vs all*. Dalam pendekatan *one vs all*, model biner dipelajari untuk setiap kelas yang mencoba memisahkan kelas tersebut dari semua kelas lainnya, menghasilkan sebanyak model biner seperti jumlah kelas. Untuk membuat prediksi, semua pengklasifikasi biner dijalankan pada titik uji. Pengklasifikasi yang memiliki skor tertinggi pada kelas tunggalnya "menang," kemudian label kelas ini diambil sebagai prediksi [15].

Berikut merupakan cuplikan kode yang digunakan untuk melakukan *training* dan evaluasi model *Logistic Regression*:

```
1 import pandas as pd
2 from sklearn.model_selection import
train_test_split
3 from sklearn.feature_extraction.text
import TfidfVectorizer
4 from sklearn.neighbors import
KNeighborsClassifier
5 from sklearn.metrics import
accuracy_score, classification_report,
confusion_matrix
6
7 df =
pd.read_csv('/content/drive/MyDrive/Twee
ts.csv', encoding="ISO-8859-1")
8
9 X = df['text']
10 y = df['airline_sentiment']
11
12 X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
13
14 vectorizer =
TfidfVectorizer(max_features=5000)
15 X_train_tfidf =
vectorizer.fit_transform(X_train)
16 X_test_tfidf =
vectorizer.transform(X_test)
```

```
17
18 logreg_model = LogisticRegression()
19 logreg_model.fit(X_train_tfidf,
y_train)
20
21 y_pred =
logreg_model.predict(X_test_tfidf)
22
23 accuracy = accuracy_score(y_test,
y_pred)
24 print(f"Accuracy: {accuracy:.2f}")
25
26 print("Classification Report:")
27 print(classification_report(y_test,
y_pred))
28
29 print("Confusion Matrix:")
30 print(confusion_matrix(y_test,
y_pred))
```

Baris 1 hingga 5 berisi *library* yang akan digunakan saat menggunakan model *Logistic Regression*

Baris 7 berfungsi untuk menyimpan dataset *tweets* yang digunakan kepada variable *dataframe "df"*.

Baris 9 dan 10 berfungsi untuk mengambil seluruh observasi dari kolom *"text"* dan *"airline_sentiment"* dan menyimpannya ke dalam variabel *X* dan *Y* secara berurutan

Baris 12 bertujuan untuk membagi dataset untuk *training* dan *testing*, dalam percobaan ini, penulis menggunakan *test size* sebesar 0.2 (20%) dan *training size* sebesar 0.8 (80%). *Random state* yang digunakan adalah 42, yang merupakan sebuah angka acak yang dapat digunakan untuk mereplikasi ulang percobaan ini dengan menggunakan "*random seed"* yang sama setiap kalinya, yang akan menghasilkan hasil yang sama.

Baris 14 hingga 16 merupakan tahap *pre-processing* yang menggunakan *TF-IDF (Term Frequency – Inverse Document Frequency)* untuk mengubah data yang berupa teks di kolom "*text*" dan mentransformasikannya ke bentuk numerik hingga dapat di proses oleh model *machine learning* tersebut [2]. Parameter *max_features* yang bernilai 5000 berarti hanya 5000 kata atau *term* yang paling relevan yang akan digunakan untuk melatih model.

Baris 18 dan 19 bertujuan untuk melatih model *Logistic Regression* dengan menggunakan data *train*.

Baris 21 melakukan prediksi data dengan menggunakan model yang sudah terlatih tersebut, dengan parameter *test* data.

Baris 23 bertujuan untuk menghitung akurasi dari prediksi yang telah dilakukan model yang sudah dilatih tersebut.

Baris 24, 26 hingga 27, dan 29 hingga 30 bertujuan untuk menampilkan akurasi, *Classification Report*, dan *Confusion Matrix* dari hasil percobaan model tersebut secara berurutan.

E. *Random Forest Classifier*

Random Forest adalah solusi untuk mengatasi kelemahan *decision tree* yang cenderung *overfitting* pada data *training*. Ini adalah kumpulan pohon keputusan yang masing-masing sedikit berbeda satu sama lain. Meskipun setiap pohon mungkin cenderung overfitting pada sebagian data, dengan membangun banyak pohon yang beragam, kita dapat mengurangi *overfitting* dengan merata-ratakan hasil mereka. Pengurangan *overfitting* ini, sambil mempertahankan kekuatan prediksi, dapat ditunjukkan secara matematis. Untuk menerapkan strategi ini, kita perlu membangun banyak *decision tree*, setiap pohon seharusnya dapat memprediksi dengan baik dan berbeda dari pohon lainnya. Nama "*decision tree*" berasal dari pengenalan unsur keacakan dalam membangun *tree* atau pohon, baik dalam pemilihan data maupun fitur pada setiap uji pemisahan [16].

Berikut merupakan cuplikan kode yang digunakan untuk melakukan *training* dan evaluasi model *Random Forest Classifier*:

```
1 import pandas as pd
2 from sklearn.model_selection import
train_test_split
3 from sklearn.feature_extraction.text
import TfidfVectorizer
4 from sklearn.neighbors import
KNeighborsClassifier
5 from sklearn.metrics import
accuracy_score, classification_report,
confusion_matrix
6
7 df =
pd.read_csv('/content/drive/MyDrive/Twee
ts.csv', encoding="ISO-8859-1")
8
9 X = df['text']
10 y = df['airline_sentiment']
11
12 X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)
13
14 vectorizer =
TfidfVectorizer(max_features=5000)
15 X_train_tfidf =
vectorizer.fit_transform(X_train)
16 X_test_tfidf =
vectorizer.transform(X_test)
17
```

```
18 model =
RandomForestClassifier(n_estimators=100,
random_state=42)
19 model.fit(X_train_tfidf, y_train)
20
21 y_pred = model.predict(X_test_tfidf)
22
23 accuracy = accuracy_score(y_test,
y_pred)
24 print(f"Accuracy: {accuracy:.2f}")
25
26 print("Classification Report:")
27 print(classification_report(y_test,
y_pred))
28
29 print("Confusion Matrix:")
30 print(confusion_matrix(y_test,
y_pred))
```

Baris 1 hingga 5 berisi *library* yang akan digunakan saat menggunakan model *Random Forest Classifier*.

Baris 7 berfungsi untuk menyimpan dataset *tweets* yang digunakan kepada variable *dataframe "df"*.

Baris 9 dan 10 berfungsi untuk mengambil seluruh observasi dari kolom *"text"* dan *"airline_sentiment"* dan menyimpannya ke dalam variabel *X* dan *Y* secara berurutan

Baris 12 bertujuan untuk membagi dataset untuk *training* dan *testing*, dalam percobaan ini, penulis menggunakan *test size* sebesar 0.2 (20%) dan *training size* sebesar 0.8 (80%). *Random state* yang digunakan adalah 42, yang merupakan sebuah angka acak yang dapat digunakan untuk mereplikasi ulang percobaan ini dengan menggunakan "*random seed"* yang sama setiap kalinya, yang akan menghasilkan hasil yang sama.

Baris 14 hingga 16 merupakan tahap *pre-processing* yang menggunakan *TF-IDF (Term Frequency – Inverse Document Frequency)* untuk mengubah data yang berupa teks di kolom "*text*" dan mentransformasikannya ke bentuk numerik hingga dapat di proses oleh model *machine learning* tersebut [2]. Parameter *max_features* yang bernilai 5000 berarti hanya 5000 kata atau *term* yang paling relevan yang akan digunakan untuk melatih model.

Baris 18 dan 19 bertujuan untuk melatih model *Random Forest Classifier* dengan menggunakan data *train*.

Baris 21 melakukan prediksi data dengan menggunakan model yang sudah terlatih tersebut, dengan parameter *test* data.

Baris 23 bertujuan untuk menghitung akurasi dari prediksi yang telah dilakukan model yang sudah dilatih tersebut.

Baris 24, 26 hingga 27, dan 29 hingga 30 bertujuan untuk menampilkan akurasi, *Classification Report*, dan *Confusion Matrix* dari hasil percobaan model tersebut secara berurutan.

IV. HASIL DAN DISKUSI

A. Hasil Percobaan

Hasil akurasi dari percobaan *training* model-model klasifikasi *machine learning* adalah sebagai berikut:

Tabel 1. Hasil akurasi dari model-model *klasifikasi machine learning*

Model yang digunakan	**Akurasi (%)**
Naïve Bayes	74
K-Nearest Neighbor (KNN)	72
Random Forest	*77*
Logistic Regression	81

Berikut adalah hasil *model evaluation* berupa *classification report* dan *confusion matrix* masing-masing model:

Tabel 2.1. Tabel *classification report Naïve Bayes Classifier*

	Precision (%)	**Recall (%)**	**F1-score (%)**	**Support**
Negative	72	99	84	1889
Neutral	76	24	37	580
Positive	91	32	48	459
Accuracy			74	2928
Macro Avg	80	52	56	2928
Weighted Avg	76	74	69	2928

Tabel 2.2. Tabel *confusion matrix Naïve Bayes Classifier*

	Predicted Negative	Predicted Neutral	Predicted Positive
Actual Negative	1869	17	3
Actual Neutral	427	142	11
Actual Positive	283	27	149

Tabel 3.1. Tabel *classification report K-Nearest Neighbors*

	Precision (%)	**Recall (%)**	**F1-score (%)**	**Support**
Negative	78	96	86	1889
Neutral	67	40	50	580
Positive	81	46	59	459
Accuracy			77	2928
Macro Avg	76	61	65	2928

Weighted Avg	76	77	75	2928

Tabel 3.2. Tabel *confusion matrix K-Nearest Neighbors*

	Predicted Negative	Predicted Neutral	Predicted Positive
Actual Negative	1559	251	79
Actual Neutral	243	281	56
Actual Positive	112	83	264

Tabel 4.1. Tabel *classification report Random Forest Classifier*

	Precision (%)	**Recall (%)**	**F1-score (%)**	**Support**
Negative	81	83	82	1889
Neutral	46	48	47	580
Positive	66	58	62	459
Accuracy			72	2928
Macro Avg	64	63	64	2928
Weighted Avg	72	72	72	2928

Tabel 4.2. Tabel *confusion matrix Random Forest Classifier*

	Predicted Negative	Predicted Neutral	Predicted Positive
Actual Negative	1815	56	18
Actual Neutral	317	231	32
Actual Positive	190	56	213

Tabel 5.1. Tabel *classification report Logistic Regression*

	Precision (%)	**Recall (%)**	**F1-score (%)**	**Support**
Negative	84	94	88	1889
Neutral	67	53	59	580
Positive	80	62	70	459
Accuracy			81	2928
Macro Avg	77	70	72	2928
Weighted Avg	80	81	80	2928

Tabel 5.2. Tabel *confusion matrix Logistic Regression*

	Predicted Negative	Predicted Neutral	Predicted Positive
Actual Negative	1767	92	30
Actual Neutral	231	306	43
Actual Positive	116	57	286

B. Diskusi

Berikut adalah hasil analisis dari hasil akurasi, dan tabel *classification report* dan *confusion matrix*:

Naïve Bayes memiliki akurasi ke 3 paling tinggi, dan memiliki nilai *precision-recall-f1* yang cukup bervariasi, tetapi memiliki *precision "negative"* yang cukup bagus, tetapi rendah untuk "*neutral"* dan "*positive"*. Hasil dari *confusion matrix* cukup seimbang untuk "*negative"*, tetapi buruk untuk *neutral* dan *positive*.

Random Forest memiliki nilai akurasui ke 2 paling tinggi, dan memiliki precision dan recall yang bagus untuk "*negative*", tetapi relatif biasa untuk "*neutral*" dan "*positive"*. Hasil dari *confusion matrix* menunjukkan hasil yang cukup bagus untuk seluruh *class*.

k-Nearest Neighbors (KNN) memiliki akurasi terburuk dari 4 model, tetapi masih memiliki *precision* yang cukup seimbang untuk "*negative"*, yang lebih rendah untuk "*neutral*" dan *"positive"*. *Confusion matrix* menunjukkan bahwa model *KNN* cukup buruk dalam melakukan prediksi untuk semua *class*.

Logistic regression memiliki akurasi tertinggi dari 4 model dan memiliki *precision* yang bagus untuk "*negative"* dan cukup untuk *"neutral" dan* "*positive*". *Confusion Matrix* menunjukkan bahwa *Logistic Regression* dapat memprediksi dengan cukup bagus untuk semua *class*.

Dapat dilihat bahwa secara general, performa *class neutral* cukup buruk jika disbanding dengan kedua *class* lainnya, terutama *class positive* yang memiliki jumlah dataset yang lebih sedikit, tetapi memiliki performa yang lebih baik dari *class neutral*. Hal ini dapat disebabkan oleh misklasifikasi dataset awal. Tweet yang terklasifikasi sebagai netral oleh dataset mungkin dapat memiliki sifat ambigu, dimana model tidak dapat mengklasifikasikan *tweet* netral dengan baik karena berada di antara *positif* dan *negative.*

Limitasi dataset yang digunakan juga mungkin berpengaruh terhadap akurasi model prediksi yang dicoba, karena akurasi pralabel *sentiment* yang berfluktuasi dari dataset *tweets*.

IV. KONKLUSI

Secara garis besar, *Logistic regression* memiliki performa yang lebih bagus dari pada model yang lain. *Random forest* juga memiliki performa yang cukup baik, walaupun lebih rendah jika disbanding dengan *Logistic*

Regression. *Naïve Bayes* tampak mengalami kesulitan dalam *recall* untuk *sentiment "neutral" dan "positive*", dan *k-Nearest Neighbors* memiliki performa yang seimbang, tetapi akurasi yang lebih rendah.

Konklusi yang dapat diperoleh dari percobaan ini adalah model yang paling cocok untuk melakukan *sentiment analysis* terhadap dataset *tweets* yang mengandung pendapat publik mengenai perusahaan maskapai asal Amerika Serikat adalah *Logistic Regression,* yang dapat digeneralisasikan kepada keperluan *sentiment analysis* dengan dataset lainnya.

REFERENSI

[1] Kaggle. (https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment?resource=download)

[2] Taboada, M. (2016). Sentiment analysis: An overview from linguistics. *Annual Review of Linguistics*, *2*(1), 325–328. https://doi.org/10.1146/annurev-linguistics-011415-040518

[3] Mehta, P., & Pandya, S. (2020). A Review On Sentiment Analysis Methodologies, Practices And Applications. International Journal of Scientific & Technology Research, 9(2), 601-603. https://www.researchgate.net/profile/Pooja-Mehta-26/publication/344487215_A_Review_On_Sentiment_Analysis_Methodologies_Practices_And_Applications/links/5f7bfb2992851c14bcb16528/A-Review-On-Sentiment-Analysis-Methodologies-Practices-And-Applications.pdf

[4] *Precision and recall: How to evaluate your classification model*. Built In. (n.d.). https://builtin.com/data-science/precision-and-recall#

[5] *F1 score in Machine Learning: Intro & Calculation*. V7. (n.d.). https://www.v7labs.com/blog/f1-score-guide

[6] *Classification report*. Classification Report - Yellowbrick v1.5 documentation. (n.d.). https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html#:~:text=Support%20is%20the%20number%20of,for%20stratified%20sampling%20or%20rebalancing.

[7] Zhang, H. (2004). The Optimality of Naive Bayes. Faculty of Computer Science, University of New Brunswick. Retrieved from http://www.aaai.org/

[8] Peterson, L. (2009). K-Nearest Neighbor. *Scholarpedia*, *4*(2), 1883. https://doi.org/10.4249/scholarpedia.1883

[9] Çokluk, Ö. (2010). Logistic Regression: Concept and Application. Educational Sciences: Theory & Practice, 10(3), 1397-1407. Ankara University Faculty of Educational Sciences. Retrieved from https://files.eric.ed.gov/fulltext/EJ919857.pdf

[10] Fredric J. Janzen, Hal S. Stern, LOGISTIC REGRESSION FOR EMPIRICAL STUDIES OF MULTIVARIATE SELECTION, *Evolution*, Volume 52, Issue 6, 1 December 1998, Pages 1564–1571, https://doi.org/10.1111/j.1558-5646.1998.tb02237.x

[11] Zakariah, M. (2014). Classification of large datasets using Random Forest Algorithm in various applications: Survey. International Journal of Engineering and Innovative Technology (IJEIT), 4(3), ISSN: 2277-3754. King Saud University, College of Computer and Information Sciences, Riyadh, Kingdom of Saudi Arabia. ISO 9001:2008 Certified.

[12] Biau, G. (2012). Analysis of a random forests model. *The Journal of Machine Learning Research*, *13*, 1063-1095.

[13] Müller, Andreas C., and Sarah Guido. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc., 2016, pp. 68-69.

[14] Müller, Andreas C., and Sarah Guido. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc., 2016, pp. 20-21, 35-36.

[15] Müller, Andreas C., and Sarah Guido. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc., 2016, pp. 63-64.

[16] Müller, Andreas C., and Sarah Guido. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc., 2016, pp. 83.

[17] Suresh, A. (2021, June 22). *What is a confusion matrix?*. Medium. https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5

[18] Gamal, B. (2022, January 14). *Performance metrics for classification models in Machine Learning: Part II*. Medium. https://bassantgz30.medium.com/performance-metrics-for-classification-models-in-machine-learning-part-ii-9303a1c7cadd