LAPORAN TUGAS ML TEXT CLASSIFICATION

IF5153 NATURAL LANGUAGE PROCESSING

DISUSUN OLEH

MUHAMMAD FADHIL AMRI 13521066

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2024

DAFTAR ISI

DAFTAR ISI **2**
BAB I **4**
KODE PROGRAM **4**
1.1 Setup 4
1.2 Load Dataset 4
1.3 Exploratory Data Analysis 4
1.3.1 Persebaran label data 4
1.3.2 Persebaran panjang text 5
1.3.3 Persebaran panjang kata setiap text 5
1.3.4 Kata stopwords yang paling sering muncul 5
1.3.5 Kata non-stopwords yang paling sering muncul 5
1.3.6 Word cloud dari text. 6
1.4 Data Preprocessing 6
1.4.1 Convert text to lowercase 6
1.4.2 Remove number 6
1.4.3 Remove punctuation 6
1.4.4 Remove whitespace 6
1.4.5 Remove stopwords 6
1.5 Feature Extraction 7
1.6 Modeling 7
1.6.1 Split Data 7
1.6.2 Multinomial Naive Bayes 7
1.6.3 SVM 7
1.6.4 Random Forest 7
1.6.5 KNN 7
1.7 Error Analysis 7
BAB II **9**
SKENARIO EKSPERIMEN **9**
2.1 Data Understanding 9
2.2 Experiment Design 9
2.2.1 Recognition of and Statement of the Problem 9
2.2.2 Selection of Response Variable 9
2.2.3 Choices of Factors, Levels, and Ranges 9
2.2.4 Choice of Experimental Design 9
2.2.5 Performing the Experiment 9
BAB III **10**
HASIL EKSPERIMEN **10**
3.1 Data Understanding 10
3.1.1 Persebaran label data 10
3.1.2 Persebaran panjang text 10

3.1.3 Persebaran panjang kata setiap text 11
3.1.4 Kata stopwords yang paling sering muncul 11
3.1.5 Kata non-stopwords yang paling sering muncul 12
3.2 Experiment 12
3.2.1 Multinomial Naive Bayes 12
3.2.2 Support Vector Machine 12
3.2.3 Random Forest 13
3.2.4 K-Nearest Neighbour 13
BAB IV **14**
ERROR ANALYSIS **14**
4.1 Best Result Error 14
4.2 Another Model Error 15

BAB I

KODE PROGRAM

Link repository: https://github.com/Mehmed13/traditional_ml_nlp

1.1 Setup

Program diawali dengan melakukan setup berupa instalasi dan import dependency yang dibutuhkan

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from collections import  Counter
from wordcloud import WordCloud, STOPWORDS
import re
import string
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score, f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier


import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
stop=set(stopwords.words('indonesian'))
```

1.2 Load Dataset

Program selanjutnya melakukan load dari dataset menggunakan tautan dataset yang telah diberikan. Dataset yang diload adalah dataset train dan dataset validation

```
train_data = pd.read_csv("https://raw.githubusercontent.com/IndoNLP/indonlu/refs/heads/maste
```

```
val_data = pd.read_csv("https://raw.githubusercontent.com/IndoNLP/indonlu/refs/heads/master/d
```

1.3 Exploratory Data Analysis

Program selanjutnya melakukan proses Exploratory Data Analysis (EDA) untuk memahami data dengan lebih baik dan sebagai bahan pertimbangan dalam menentukan preprocessing yang perlu dilakukan. EDA yang dilakukan meliputi:

1.3.1 Persebaran label data

```
train_data_count = train_data.groupby('label').count()
plt.bar(train_data_count.index.values, train_data_count["text"])
plt.xlabel('Label')
plt.ylabel('Number of Texts')
```

1.3.2 Persebaran panjang text

```
train_data["text"].str.len().hist()
plt.show()
```

1.3.3 Persebaran panjang kata setiap text

```
train_data['text'].str.split() \
    .apply(lambda x : [len(i) for i in x]). \
   map(lambda x: np.mean(x)).hist()

plt.show()
```

1.3.4 Kata stopwords yang paling sering muncul

```
corpus=[]
data = train_data['text'].str.split()
data = data.values.tolist()
corpus=[word for i in data for word in i]

from collections import defaultdict
dic=defaultdict(int)
for word in corpus:
    if word in stop:
        dic[word]+=1

top=sorted(dic.items(), key=lambda x:x[1],reverse=True)[:10]
x,y=zip(*top)
plt.bar(x,y)
plt.show()
```

1.3.5 Kata non-stopwords yang paling sering muncul

```
data= train_data["text"].str.split()
data=data.values.tolist()
corpus=[word for i in data for word in i]

counter=Counter(corpus)
most=counter.most_common()
x, y=[], []
for word,count in most[:40]:
    if (word not in stop):
        x.append(word)
        y.append(count)

sns.barplot(x=y,y=x)
plt.show()
```

1.3.6 Word cloud dari text.

```
stopwords = set(STOPWORDS)

def show_wordcloud(data):
    wordcloud = WordCloud(
        background_color='white',
        stopwords=stopwords,
        max_words=100,
        max_font_size=30,
        scale=3,
        random_state=1)

    wordcloud=wordcloud.generate(str(data))

    fig = plt.figure(1, figsize=(12, 12))
    plt.axis('off')

    plt.imshow(wordcloud)
    plt.show()

show_wordcloud(corpus)
```

1.4 Data Preprocessing

Program selanjutnya memasuki tahap preprocessing yang terbagi atas case folding dan stopword removal. Berikut preprocessing yang telah dilakukan.

1.4.1 Convert text to lowercase

```
def convert_to_lowercase(df: pd.DataFrame):
    df["text"] = df["text"].apply(lambda text: text.lower())
    return df
```

1.4.2 Remove number

```
def remove_number(df: pd.DataFrame):
    df["text"] = df["text"].apply(lambda text: re.sub(r"\d+", "", text))
    return df
```

1.4.3 Remove punctuation

```
def remove_punctuation(df: pd.DataFrame):
    df["text"] = df["text"].apply(lambda text: text.translate(str.maketrans("","", string.punctuation)))
    return df
```

1.4.4 Remove whitespace

```
def remove_whitespace(df: pd.DataFrame):
    df["text"] = df["text"].apply(lambda text: text.strip())
    return df
```

1.4.5 Remove stopwords

```
def remove_stopwords(df: pd.DataFrame):
    # According to the bar plot, we will remove word "nya" as it was a reference
    df["text"] = df["text"].apply(lambda text: text.replace(" nya", ""))
    return df
```

1.5 Feature Extraction

Program selanjutnya memasuki tahap feature extaction menggunakan Bag of Words

```
# Creating the Bag of Words model
vectorizer = CountVectorizer()
train_text_counts = vectorizer.fit_transform(train_data["text"])
val_text_counts = vectorizer.transform(val_data["text"])
```

1.6 Modeling

Program selanjutnya memasuki tahap modeling yang mencakup split data dan pembangunan model

1.6.1 Split Data

```
X_train, y_train = train_text_counts, train_data["label"]
X_val, y_val = val_text_counts, val_data["label"]
```

1.6.2 Multinomial Naive Bayes

```
mnb_clf = MultinomialNB().fit(X_train, y_train)
prediction_mnb = mnb_clf.predict(X_val)
print(classification_report(y_val, prediction_mnb, target_names=y_train.unique()))
```

1.6.3 SVM

```
svm_clf = SVC().fit(X_train, y_train)
prediction_svm = svm_clf.predict(X_val)
print(classification_report(y_val, prediction_svm, target_names=y_train.unique()))
```

1.6.4 Random Forest

```
rf_clf = RandomForestClassifier().fit(X_train, y_train)
prediction_rf = rf_clf.predict(X_val)
print(classification_report(y_val, prediction_rf, target_names=y_train.unique()))
```

1.6.5 KNN

```
knn_clf = KNeighborsClassifier().fit(X_train, y_train)
prediction_knn = knn_clf.predict(X_val)
print(classification_report(y_val, prediction_knn, target_names=y_train.unique()))
```

1.7 Error Analysis

Terakhir, program memasuki tahapan error analysis untuk mengecek error yang terjadi dan menganalisis kemungkinan penyebabnya

```
# Error analysis using svm prediction, as it gives the best result
val_data["pred"] = prediction_svm
error_predictions = val_data[val_data["pred"]!=val_data["label"]]
error_predictions
```

```
sample_error = error_predictions.sample(5)
```

```
for index, row in sample_error.iterrows():
    print(f"Index: {index}")
    print(f"Text: {row['text']}, Label: {row['label']}, Prediction: {row['pred']}")
```

BAB II

SKENARIO EKSPERIMEN

2.1 Data Understanding

Data understanding dilakukan untuk memahami karakteristik dari data. Ini mencakup tahap mendeskripsikan data, seperti variabel-variabel yang ada dan statistika deskriptif lainnya. Selain itu, pada tahap ini dilakukan juga tahapan Exploratory Data Analysis (EDA) yang bertujuan untuk menggali informasi tentang data. EDA yang dilakukan adalah pengecekan persebaran data yang mencakup label, panjang text, panjang kata, dan kata-kata yang paling sering muncul. Hasil dari tahapan Data Understanding ini nantinya akan menjadi pertimbangan untuk tahap preprocessing dan modeling yang dilakukan.

2.2 Experiment Design

2.2.1 Recognition of and Statement of the Problem

Permasalahan yang akan dibahas pada eksperimen ini adalah sentiment analysis dari kumpulan text berbahasa Indonesia. Keluaran dari eksperimen ini adalah sebuah model traditional machine learning yang mampu melakukan analisis sentimen memanfaatkan data training yang ada.

2.2.2 Selection of Response Variable

Response variable pada eksperimen ini adalah variabel label yang memiliki nilai categorical berupa positive, neutral, dan negative. Performansi dari model akan diukur menggunakan metrik Akurasi, Precision, Recall dan F1-score.

2.2.3 Choices of Factors, Levels, and Ranges

Faktor-faktor yang akan mempengaruhi eksperimen ini adalah model yang digunakan, sedangkan variabel yang berperan sebagai fitur hanya variabel text.

2.2.4 Choice of Experimental Design

Strategi yang akan digunakan saat eksperimen adalah best guess agar proses eksperimen dapat berlangsung lebih cepat dan berfokus pada pemilihan model. Sementara itu, skema split data yang dilakukan adalah hold-out validation karena data yang tersedia sudah dipisahkan antara training data dengan validation data.

2.2.5 Performing the Experiment

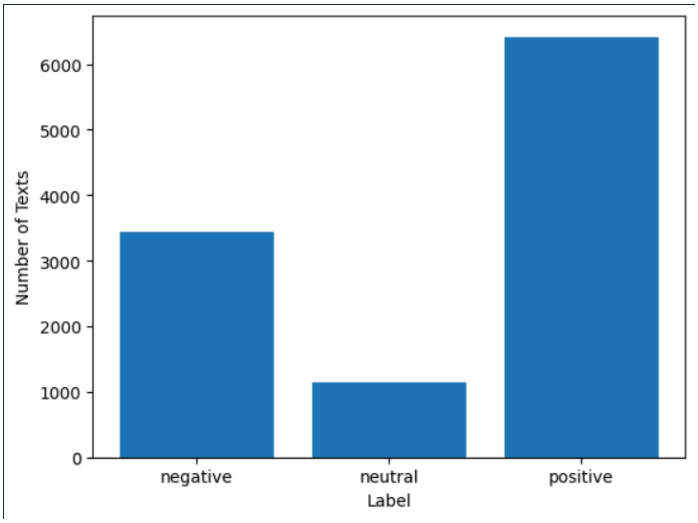
Eksperimen diawali dengan preprocessing dari text yang terdiri atas lowercasing, remove number, remove punctuation, remove whitespace, dan remove stopwords. Setelah preprcessing dilakukan, data dibagi menjadi fitur dan target dengan kolom “text” menjadi fitur dan kolom “label” menjadi target. Data yang sudah di-split kemudian digunakan untuk training model. Model-model yang digunakan dalam eksperimen ini adalah Multinomial Naive Bayes, Support Vector Machine, Random Forest, dan K Nearest Neighbours.

BAB III

HASIL EKSPERIMEN

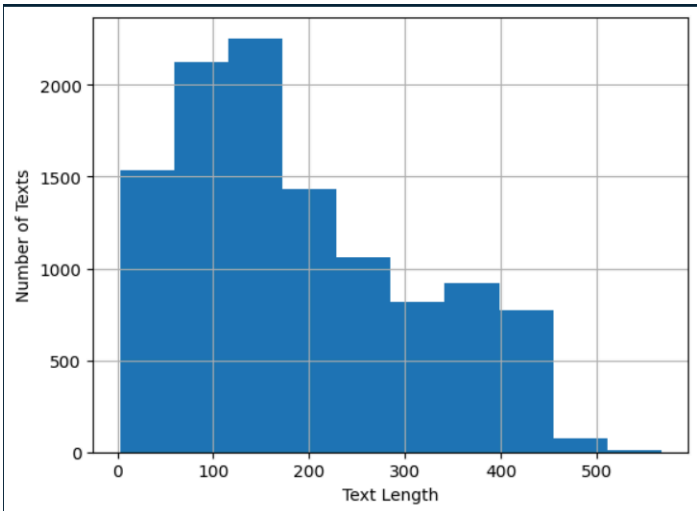
3.1 Data Understanding

3.1.1 Persebaran label data



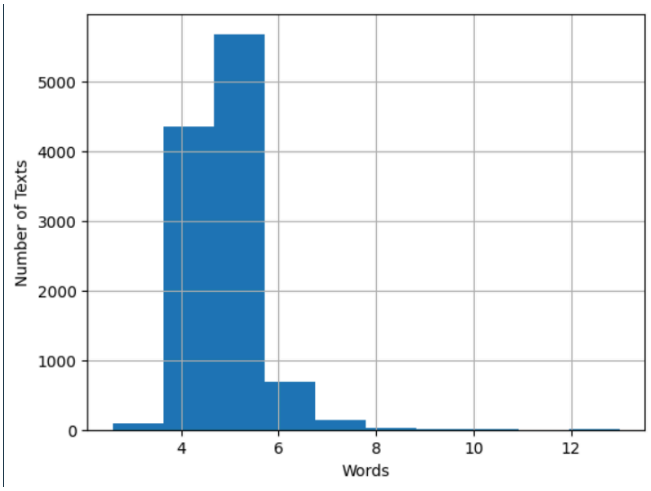
Pada bar chart terlihat bahwa dataset yang digunakan imbalance dengan mayoritas label adalah positive. Hal ini dapat membuat model menjadi bias sehingga berpotensi menurunkan f1-score dari model.

3.1.2 Persebaran panjang text



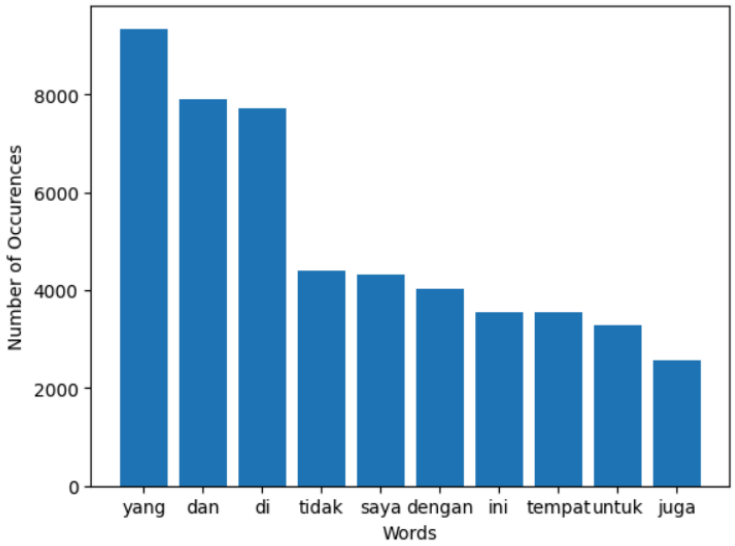
Pada histogram chart terlihat bahwa panjang text mayoritas berada pada rentang 50-150 huruf.

3.1.3 Persebaran panjang kata setiap text



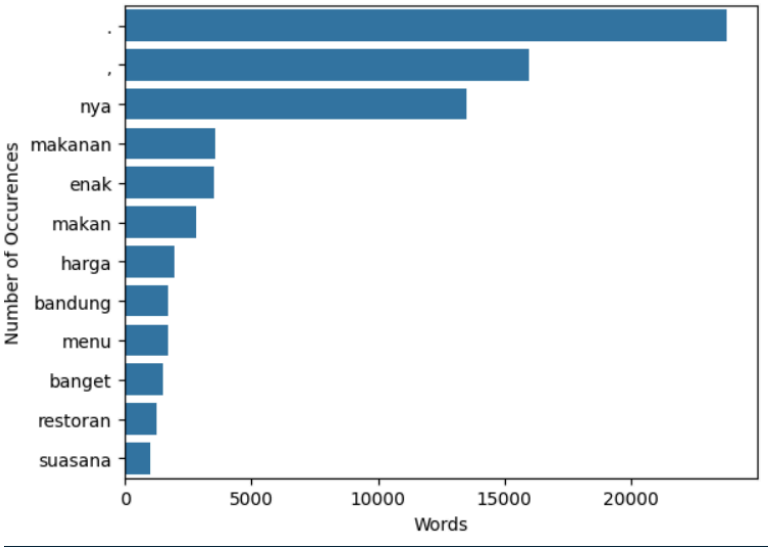
Pada histogram chart terlihat bahwa banyak kata dalam text mayoritas berada pada rentang 4-5 kata.

3.1.4 Kata stopwords yang paling sering muncul



Pada bar chart terlihat bahwa kata stopwords yang paling sering muncul adalah kata “yang”, “dan”, “di”. Kata-kata tersebut bisa dihapus karena tidak akan mengubah konteks.

3.1.5 Kata non-stopwords yang paling sering muncul



Pada bar chart terlihat bahwa kata non-stopwords yang paling sering muncul adalah kata “.”, “,”, “nya”. Kata-kata tersebut bisa dihapus karena tidak akan mengubah konteks, terutama tanda baca.

3.2 Experiment

3.2.1 Multinomial Naive Bayes

	precision	recall	f1-score	support
positive	0.77	0.86	0.81	394
neutral	0.90	0.58	0.71	131
negative	0.90	0.90	0.90	735
accuracy			0.85	1260
macro avg	0.86	0.78	0.81	1260
weighted avg	0.86	0.85	0.85	1260

3.2.2 Support Vector Machine

	precision	recall	f1-score	support
positive	0.80	0.86	0.83	394
neutral	0.77	0.69	0.73	131
negative	0.92	0.90	0.91	735
accuracy			0.86	1260
macro avg	0.83	0.81	0.82	1260
weighted avg	0.87	0.86	0.86	1260

3.2.3 Random Forest

	precision	recall	f1-score	support
positive	0.78	0.81	0.79	394
neutral	0.89	0.55	0.68	131
negative	0.89	0.93	0.91	735
accuracy			0.85	1260
macro avg	0.85	0.76	0.79	1260
weighted avg	0.86	0.85	0.85	1260

3.2.4 K-Nearest Neighbour

	precision	recall	f1-score	support
positive	0.45	0.61	0.52	394
neutral	0.27	0.09	0.14	131
negative	0.70	0.64	0.67	735
accuracy			0.58	1260
macro avg	0.47	0.45	0.44	1260
weighted avg	0.58	0.58	0.57	1260

Berdasarkan classification report, dapat dilihat bahwa model yang memberikan performa terbaik adalah model Support Vector Machine dengan accuracy 0.86, f1-score macro average 0.82, precision average 0.83, dan recall average 0.82.

BAB IV

ERROR ANALYSIS

4.1 Best Result Error

	text	label	pred
0	meski masa kampanye sudah selesai bukan berat...	neutral	negative
22	senang pakai kartu	positive	neutral
24	kalau subscriber youtube netmediatama mencapai...	neutral	negative
26	kalau melihat desain pixel sih biasa saja ta...	positive	negative
29	salut	positive	neutral
...	...	...	...
1197	tidak aib	positive	negative
1198	sebenarnya makanan steak di sini sangat enak d...	negative	positive
1200	ada membandingkan antara jakarta dan kota sby ...	negative	positive
1231	aku banyak diblokir fpi buat aplikasi baru pe...	neutral	negative
1241	semoga virus kebaikan dimiliki masyarakat dhar...	positive	neutral

171 rows × 3 columns

Dapat dilihat bahwa model SVM menghasilkan error pada 171 instances. Misal jika ditelusuri lebih lanjut dari 3 sample berikut.

1. Index 100

   ```
   Text: tidak iri, Label: positive, Prediction: negative
   ```

   Error tersebut kemungkinan besar disebabkan oleh kata ‘iri’. Di sini model akan memiliki konotasi negatif untuk kata tersebut, meskipun terdapat kata “tidak” sebelumnya. Ketidakmampuan ini disebabkan karena model tradisional yang digunakan belum mampu menangkap konteks sehingga kata “tidak” dan “iri” akan dianggap sebagai dua kata yang terpisah.

2. Index 1033

   ```
   Text: saya sarankan untuk ada penjelasan lebih jelas untuk setiap
   layanan paket supaya kita tidak bingung dan salah,
   Label: neutral, Prediction: negative
   ```

   Error tersebut kemungkinan disebabkan oleh kata “bingung’ dan “salah”. Di sini model akan memiliki konotasi negatif untuk kata tersebut, meskipun terdapat kata “tidak” sebelumnya. Ketidakmampuan ini disebabkan karena model tradisional yang digunakan belum mampu menangkap konteks sehingga kata “tidak” dan “bingung”, “salah” akan dianggap sebagai kata-kata yang terpisah.

3. Index 388

   ```
   Text: mari terus upayakan perubahan  perubahan ke arah lebih baik
   tentunya  moeldoko,
   ```

```
Label: positive, Prediction: negative
```

Error tersebut kemungkinan disebabkan oleh bias model terhadap entitas. Pada text kata-kata yang ada memiliki nuansa positif, misalnya kata “baik”. Namun, di akhir terdapat kata “Moeldoko” yang merupakan entitas pejabat di Indonesia. Untuk mengatasi hal tersebut, dapat dilakukan preprocessing entity masking.

4.2 Another Model Error

Tingginya tingkat error pada model KNN disebabkan oleh struktur data yang digunakan sebagai input. Bag of words yang bernilai diskrit tentu akan membuat model kebingungan karena banyak kombinasi yang memungkinkan untuk memberi nilai “distance” yang sama. Hasilnya model terlihat seperti menebak-menebak yang dibuktikan dengan akurasi 0.58 dan macro average di angka 0.44. Sementara itu, random forest meskipun memberikan hasil yang cukup baik, struktur data input yang memiliki fitur sangat banyak juga dapat membuat model menjadi overfit dan ukuran tree yang digunakan juga bisa menjadi overgeneralisasi apabila pemangkasan dilakukan terlalu cepat.