

```
import pandas as pd
import numpy as np

tweet_data = pd.read_excel("combined_data.xlsx")
tweet_data.head()
```



	Tweets	clean_tweets	Social	Historical	Dehumanization	Accusation	Attack	loyalty	Threat	HS	Non HS	Anticipation	Trust	Joy	Anger	Disgust	Fear	Sadness	Surprise
0	Sama banget Komentar ini dgn para pendukung se...	komentar dukung sella joko widodo presiden ri ...	1	0	0	0	0.0	0	0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0
1	Ha ha ha sigundul penguasa ancol karena selama...	sigundul kuasa ancol jilat gabenar buka jeroan...	1	0	0	1	0.0	0	0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0
2	Kashan itu ini jadi korban akibat dicuci otak...	kashan korban akibat cuci otak kadrun langgun...	0	0	1	1	0.0	0	0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0
3	Ha ha kakadrun pada stressssssss mengenal beber...	kakadrun stres-pimpin negara inggris turun an...	1	0	1	0	0.0	0	0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0
4	Kok sewot dgn Pidato Sambutan Bpk Joko Widodo	sewot pidato sambutan joko widodo presiden ri ha...	0	0	1	0	0.0	0	0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0

Activate Windows
Go to PC settings to activate Windows.

```
tweet_data = pd.read_excel('text_preprocessing_new.xlsx')

jumlah_hs = tweet_data['HS'].sum()
jumlah_nhs = tweet_data['Non HS'].sum()
jumlah_anticipation = tweet_data['Anticipation'].sum()
jumlah_trust = tweet_data['Trust'].sum()
jumlah_joy = tweet_data['Joy'].sum()
```

```
jumlah_anger = tweet_data['Anger'].sum()
jumlah_disgust = tweet_data['Disgust'].sum()
jumlah_fear = tweet_data['Fear'].sum()
jumlah_sadness = tweet_data['Sadness'].sum()
jumlah_surprise = tweet_data['Surprise'].sum()

print("Jumlah HS:", jumlah_hs)
print("Jumlah NHS:", jumlah_nhs)
print("Jumlah Anticipation:", jumlah_anticipation)
print("Jumlah Trust:", jumlah_trust)
print("Jumlah Joy:", jumlah_joy)
print("Jumlah Anger:", jumlah_anger)
print("Jumlah Disgust:", jumlah_disgust)
print("Jumlah Fear:", jumlah_fear)
print("Jumlah Sadness:", jumlah_sadness)
print("Jumlah Surprise:", jumlah_surprise)
```

```

Jumlah HS: 883
Jumlah NHS: 617
Jumlah Anticipation: 291
Jumlah Trust: 166
Jumlah Joy: 135
Jumlah Anger: 375
Jumlah Disgust: 400
Jumlah Fear: 21
Jumlah Sadness: 48
Jumlah Surprise: 59
```

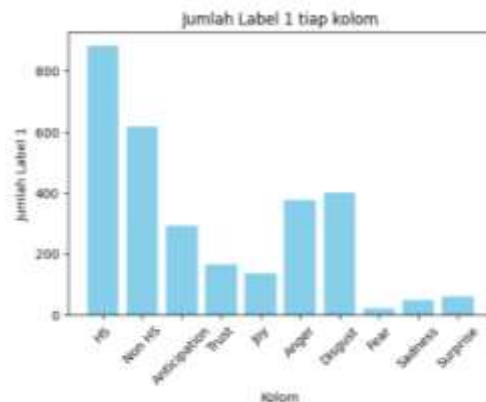
```
import matplotlib.pyplot as plt
```

```
kolom = ['HS', 'Non HS', 'Anticipation', 'Trust', 'Joy', 'Anger', 'Disgust', 'Fear', 'Sad']
jumlah_nilai = [jumlah_hs, jumlah_nhs, jumlah_anticipation, jumlah_trust, jumlah_joy, jumlah_anger, jumlah_disgust, jumlah_fear, jumlah_sadness, jumlah_surprise]
```

```
plt.figure(figsize=(6, 4))
plt.bar(kolom, jumlah_nilai, color='skyblue')
plt.xlabel('Kolom')
plt.ylabel('Jumlah Label 1')
plt.title('Jumlah Label 1 tiap kolom')
plt.xticks(rotation=45)
plt.show()
```

```


```



```
#-----case folding-----
```

```
# gunakan fungsi Series.str.lower() pada Pandas
```

```
tweet_data['Tweets']= tweet_data['Tweets'].str.lower()
```

```
print('Case Folding Result : \n')
print(tweet_data['Tweets'].head(5))
print('\n\n\n')
```

```
Case Folding Result :
```

```
0    sama banget komentar ini dgn para pendukung se...
1    ha ha ha sigundul penguasa ancol karena selama...
2    kasihan ibu ini jadi korban akibat dicuci otak...
3    ha ha hakadrun pada stresssssss mengenai beber...
4    kok sewot dgn pidato sambutan bpk joko widodo ...
Name: Tweets, dtype: object
```

```
import nltk
nltk.download('punkt')

[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
import nltk
import string
import re #regex library
```

```
# import word_tokenize & FreqDist from NLTK
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
```

```
#-----Tokenizing-----
```

```
def remove_tweet_special(text):
    # remove tab, new line, ans back slice
    text = text.replace('\t'," ").replace('\n'," ").replace('\u'," ").replace('\'',"")
    # remove non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    # remove mention, link, hashtag
    text = ' '.join(re.sub("([@#][A-Za-z0-9+])|(\w+:\/\/\/\S+)", " ",text).split())
    # remove incomplete URL
    return text.replace("http://", " ").replace("https://", " ")

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_tweet_special)

# remove number
def remove_number(text):
    return re.sub(r"\d+", "", text)

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_number)

# remove punctuation
def remove_punctuation(text):
    return text.translate(str.maketrans("", "", string.punctuation))

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_punctuation)

# remove whitespace leading & trailing
def remove_whitespace_LT(text):
    return text.strip()

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_whitespace_LT)

# remove multiple whitespace into single whitespace
def remove_whitespace_multiple(text):
    return re.sub('\s+', ' ',text)

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_whitespace_multiple)

# remove single char
def remove_singl_char(text):
    return re.sub(r"\b[a-zA-Z]\b", "", text)

tweet_data['Tweets'] = tweet_data['Tweets'].apply(remove_singl_char)

# NLTK word tokenize
def word_tokenize_wrapper(text):
    return word_tokenize(text)

tweet_data['tweet_tokens'] = tweet_data['Tweets'].apply(word_tokenize_wrapper)
```

```
print('Tokenizing Result : \n')
print(tweet_data['tweet_tokens'].head())
print('\n\n\n')
```

Tokenizing Result :

```
0    [sama, banget, komentar, ini, dgn, para, pendu...
1    [ha, ha, ha, sigundul, penguasa, ancol, karena...
2    [kasihan, ibu, ini, jadi, korban, akibat, dicu...
3    [ha, ha, hakadrun, pada, stresssssss, mengenai...
4    [kok, sewot, dgn, pidato, sambutan, bpk, joko,...
Name: tweet_tokens, dtype: object
```

```
# NLTK calc frequency distribution
```

```
def freqDist_wrapper(text):
    return FreqDist(text)
```

```
tweet_data['tweet_tokens_fdist'] = tweet_data['tweet_tokens'].apply(freqDist_wrapper)
```

```
print('Frequency Tokens : \n')
print(tweet_data['tweet_tokens_fdist'].head().apply(lambda x : x.most_common()))
```

Frequency Tokens :

```
0    [(sama, 1), (banget, 1), (komentar, 1), (ini, ...
1    [(ha, 3), (sigundul, 1), (penguasa, 1), (ancol...
2    [(waspada, 2), (kasihan, 1), (ibu, 1), (ini, 1...
3    [(ha, 2), (tdk, 2), (hakadrun, 1), (pada, 1), ...
4    [(kerja, 3), (dgn, 2), (&, 2), (amp, 2), (;, 2...
Name: tweet_tokens_fdist, dtype: object
```

```
import nltk
```

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
from nltk.corpus import stopwords
```

```
#-----get stopword from NLTK stopword-----
```

```
# get stopword indonesia
```

```
list_stopwords = stopwords.words('indonesian')
```

```
#-----manually add stopwords-----
```

```
# append additional stopword
```

```
list_stopwords.extend(["yg", "dg", "rt", "dgn", "ny", "d", 'klo', 'kalo',
                        'amp', 'biar', 'bikin', 'bilang', 'gak', 'ga', 'krn',
                        nya, nih, sih, 'si, tau, 'tdk, tuh, utk'
```

```

        'ya', 'jd', 'jgn', 'sdh', 'aja', 'n', 't', 'nyg', 'hehe',
        'pen', 'u', 'nan', 'loh', 'rt', '&', 'yah'])

```

```
#-----add stopwords from txt file-----
```

```
# read txt stopwords using pandas
```

```
txt_stopword = pd.read_csv("stopwords.txt", names=["stopwords"], header=None)
```

```
# convert stopwords string to list & append additional stopwords
```

```
list_stopwords.extend(txt_stopword["stopwords"][0].split(' '))
```

```
#-----
```

```
# convert list to dictionary
```

```
list_stopwords = set(list_stopwords)
```

```
# remove stopwords pada list token
```

```
def stopwords_removal(words):
```

```
    return [word for word in words if word not in list_stopwords]
```

```
tweet_data['tweet_tokens_WSW'] = tweet_data['tweet_tokens'].apply(stopwords_removal)
```

```
print(tweet_data['tweet_tokens_WSW'].head())
```

```

0    [banget, komentar, pendukung, setia, bpk, joko...
1    [sigundul, penguasa, ancol, taunya, jilat2, ga...
2    [kasihan, korban, akibat, dicuci, otaknya, kad...
3    [hakadrun, stresssssss, pemimpin, negara, ingg...
4    [sewot, pidato, sambutan, bpk, joko, widodo, p...
Name: tweet_tokens_WSW, dtype: object

```

```
normalizad_word = pd.read_csv("normalisasi.csv")
```

```
normalizad_word_dict = {}
```

```
for index, row in normalizad_word.iterrows():
```

```
    if row[0] not in normalizad_word_dict:
```

```
        normalizad_word_dict[row[0]] = row[1]
```

```
def normalized_term(document):
```

```
    return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in
            document]
```

```
tweet_data['tweet_normalized'] = tweet_data['tweet_tokens_WSW'].apply(normalized_term)
```

```
tweet_data['tweet_normalized'].head(10)
```

```

0    [banget, komentar, pendukung, setia, bpk, joko...
1    [sigundul, penguasa, ancol, taunya, jilat2, ga...
2    [kasihan, korban, akibat, dicuci, otaknya, kad...
3    [hakadrun, stresssssss, pemimpin, negara, ingg...
4    [sewot, pidato, sambutan, bpk, joko, widodo, p...
5    [@, bospurwa, pesong/gila, ye, kadrun, ,, joko...
6    [@, _melody_mellow, pesong/gila, berfantasi, k...
7    [sembrono, gabenar, calon, presiden, ,, koalis...
8    [sembrono, mencalonkan, presiden, wakil, presi...

```

```
9      [hutang, hutang, hutang, rosi, ko, bu, sri, uc...
```

```
Name: tweet_normalized, dtype: object
```

```
!pip install Sastrawi
```

```
Requirement already satisfied: Sastrawi in /usr/local/lib/python3.10/dist-packages (1
```

```
!pip install swifter
```

```
Collecting swifter
```

```
Downloading swifter-1.4.0.tar.gz (1.2 MB)
```

```
1.2/1.2 MB 8.0 MB/s eta 0:00:00
```

```
Preparing metadata (setup.py) ... done
```

```
Requirement already satisfied: pandas>=1.0.0 in /usr/local/lib/python3.10/dist-packag
```

```
Requirement already satisfied: psutil>=5.6.6 in /usr/local/lib/python3.10/dist-packag
```

```
Requirement already satisfied: dask[dataframe]>=2.10.0 in /usr/local/lib/python3.10/d
```

```
Requirement already satisfied: tqdm>=4.33.0 in /usr/local/lib/python3.10/dist-package
```

```
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-packages
```

```
Requirement already satisfied: cloudpickle>=1.5.0 in /usr/local/lib/python3.10/dist-p
```

```
Requirement already satisfied: fsspec>=2021.09.0 in /usr/local/lib/python3.10/dist-pa
```

```
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-pack
```

```
Requirement already satisfied: partd>=1.2.0 in /usr/local/lib/python3.10/dist-package
```

```
Requirement already satisfied: pyyaml>=5.3.1 in /usr/local/lib/python3.10/dist-packag
```

```
Requirement already satisfied: toolz>=0.10.0 in /usr/local/lib/python3.10/dist-packag
```

```
Requirement already satisfied: importlib-metadata>=4.13.0 in /usr/local/lib/python3.1
```

```
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/di
```

```
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-package
```

```
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packa
```

```
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packag
```

```
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.10/dist-packages (
```

```
Requirement already satisfied: locket in /usr/local/lib/python3.10/dist-packages (fro
```

```
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (f
```

```
Building wheels for collected packages: swifter
```

```
Building wheel for swifter (setup.py) ... done
```

```
Created wheel for swifter: filename=swifter-1.4.0-py3-none-any.whl size=16507 sha25
```

```
Stored in directory: /root/.cache/pip/wheels/e4/cf/51/0904952972ee2c7aa370943706527
```

```
Successfully built swifter
```

```
Installing collected packages: swifter
```

```
Successfully installed swifter-1.4.0
```

```
# import Sastrawi package
```

```
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
```

```
import swifter
```

```
# create stemmer
```

```
factory = StemmerFactory()
```

```
stemmer = factory.create_stemmer()
```

```
# stemmed
```

```
def stemmed_wrapper(term):
```

```
    return stemmer.stem(term)
```



```
term_dict = {}

for document in tweet_data['tweet_normalized']:
    for term in document:
        if term not in term_dict:
            term_dict[term] = ' '

print(len(term_dict))
print(" ----- ")

for term in term_dict:
    term_dict[term] = stemmed_wrapper(term)

print(term,":",term_dict[term])

print(term_dict)
print(" ----- ")

# apply stemmed term to dataframe
def get_stemmed_term(document):
    return [term_dict[term] for term in document]

tweet_data['tweet_tokens_stemmed'] = tweet_data['tweet_normalized'].swifter.apply(get_stemmed_term)
print(tweet_data['tweet_tokens_stemmed'])
```

```
tweet_data.to_excel("text_preprocessing.xlsx")
```

```
import pandas as pd
```

```
df = pd.read_excel("text_preprocessing.xlsx")

# menghapus baris yang mengandung nilai nan
df.dropna(axis=0, inplace=True)

# menyimpan dataframe yang telah diubah ke file excel terbaru
df.to_excel("text_preprocessing_new.xlsx", index=False)
```

```
import pandas as pd
import numpy as np

tweet_data = pd.read_excel("text_preprocessing_new.xlsx", usecols=["HS", "Non HS",
    "Anticipation", "Trust", "Joy", "Anger", "Disgust",
    "Fear", "Sadness", "Surprise", "tweet_tokens_stemmed"])
tweet_data.columns = ["HS", "Non HS", "Anticipation", "Trust", "Joy", "Anger",
    "Disgust", "Fear", "Sadness", "Surprise", "tweet"]

tweet_data.head()
```

	HS	Non HS	Anticipation	Trust	Joy	Anger	Disgust	Fear	Sadness	Surprise	tweet
0	0	1	0	1	0	0	0	0	0	0	['banget', 'komentar', 'dukung', 'setia', 'bpk...
1	1	0	0	0	0	0	-1	0	0	0	['sigunduf', 'kuasa', 'ancol', 'tau', 'jlat2'...
2	1	0	0	0	0	0	0	0	1	0	['kasihan', 'korban', 'akibat', 'cuci', 'otak'...
3	0	1	0	1	0	0	0	0	0	0	['hakadrun', 'stresssssss', 'pimpin', 'negara'...
4	0	1	1	0	0	0	0	0	0	0	['sewol', 'pidato', 'sambut', 'bpk', 'joko', '...

Langkah berikutnya:

Buat kode dengan `tweet_data`

Lihat plot yang direkomendasikan

```

# join list of token as single document string
import ast

def join_text_list(texts):
    texts = ast.literal_eval(texts)
    return ' '.join([text for text in texts])

tweet_data["tweet_join"] = tweet_data["tweet"].apply(join_text_list)
tweet_data["tweet_join"].head()

0    banget komentar dukung setia bpk joko widodo p...
1    sigundul kuasa ancol tau jilat2 gabenar buka j...
2    kasihan korban akibat cuci otak kadrun tanggun...
3    hakadrun stressssssss pimpin negara inggeris tu...
4    sewot pidato sambut bpk joko widodo presiden r...
Name: tweet_join, dtype: object

# proses tf idf
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer

X_train, X_test, y_train, y_test = train_test_split(tweet_data['tweet_join'], tweet_data[['Anticipation', 'Trust', 'Joy', 'Anger', 'Disgust', 'F
    'Surprise']], test_size=0.2, random_state=42)

tfidf_vectorizer = TfidfVectorizer()
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)

X_test_tfidf = tfidf_vectorizer.transform(X_test)

print(X_train_tfidf)

(0, 3162)    0.20533143639689172
(0, 704)     0.23669722953271066
(0, 1591)    0.5618963044053151
(0, 803)     0.5659990657268993
(0, 271)     0.21139830271886306
(0, 804)     0.4701508067186943
(1, 2154)    0.06617628938982557
(1, 3086)    0.26374631021720113
(1, 2303)    0.2685470856656998
(1, 459)     0.2685470856656998
(1, 3515)    0.27391389601489347
(1, 1376)    0.15285817366244198
(1, 2846)    0.13546158255328944
(1, 3454)    0.3370807084749853
(1, 2912)    0.13104440114996369
(1, 2733)    0.130797434564108
(1, 3688)    0.13255498888728456
(1, 1530)    0.13229972086086736

```

```

(1, 1243)    0.19323127856161212
(1, 2999)    0.14382370687744583
(1, 409)     0.1561734699051265
(1, 906)     0.41569889063228077
(1, 3095)    0.23696367943565386
(1, 778)     0.2005794977571093
(1, 2015)    0.2554387877252736
:
(1199, 2543) 0.44795557267879127
(1199, 1438) 0.4411090714321225
(1199, 3474) 0.22772753313933597
(1199, 700)  0.2046024989584194
(1199, 1309) 0.15173652937620252
(1199, 3231) 0.16943741242491114
(1199, 1833) 0.20685125748375255
(1199, 28)   0.12621518871002693
(1199, 716)  0.12621518871002693
(1199, 3529) 0.16286507649323295
(1199, 1170) 0.17318715922485142
(1199, 2076) 0.1391559426241584
(1199, 698)  0.08440904559245566
(1199, 1305) 0.0847192382386642
(1199, 849)  0.12066479500128333
(1199, 3083) 0.11423207794587434
(1199, 2762) 0.11489703851042657
(1199, 1875) 0.12251967012471951
(1199, 3452) 0.12279193808669257
(1199, 2154) 0.05713885863521338
(1199, 2912) 0.11314819221935989
(1199, 2733) 0.1129349528708418
(1199, 3688) 0.11445248504047012
(1199, 1530) 0.11423207794587434
(1199, 906)  0.11964285277834881

```

```

# import cosine_similarity metrics
from sklearn.metrics.pairwise import cosine_similarity

# compute similarity using cosine similarity
cos_sim = cosine_similarity(X_train_tfidf, X_train_tfidf)

print(cos_sim)

```

```

[[1.         0.         0.         ... 0.         0.         0.         ]
 [0.         1.         0.00488482 ... 0.0308877  0.0191763  0.1133998 ]
 [0.         0.00488482 1.         ... 0.         0.21269102 0.00421772]
 ...
 [0.         0.0308877  0.         ... 1.         0.         0.         ]
 [0.         0.0191763  0.21269102 ... 0.         1.         0.01655747]
 [0.         0.1133998  0.00421772 ... 0.         0.01655747 1.         ]]

```

```
import pandas as pd
```

```
tfidf_df = pd.DataFrame(X_train_tfidf.toarray(), columns=tfidf_vectorizer.get_feature_nam
```

```
tfidf_df.to_csv('tfidf_matrix_new1.csv', index=False)

from sklearn.svm import SVC

# membuat model SVM dengan pendekatan OvR
svm_models = {}
predictions = pd.DataFrame()

# melatih model svm untuk setiap label kelas
for label in y_train.columns:
    # inisialisasi model svm
    svm_model = SVC(kernel='linear', probability=True)

    # melatih model svm untuk label kelas tertentu
    svm_model.fit(X_train_tfidf, y_train[label])

    # menyimpan model dalam kamus
    svm_models[label] = svm_model

    predictions[label] = svm_model.predict(X_test_tfidf)

predictions.to_excel('svm_classification_predictions.xlsx', index=False)
print("Hasil klasifikasi telah disimpan ke file 'svm_classification_predictions.xlsx'")

    Hasil klasifikasi telah disimpan ke file 'svm_classification_predictions.xlsx'

import joblib

# simpan model dan vectorizer
joblib.dump(svm_models['HS'], 'svm_model_hs.pkl')
joblib.dump(svm_models['Non HS'], 'svm_model_nhs.pkl')
joblib.dump(svm_models['Anticipation'], 'svm_model_anticipation.pkl')
joblib.dump(svm_models['Trust'], 'svm_model_trust.pkl')
joblib.dump(svm_models['Joy'], 'svm_model_joy.pkl')
joblib.dump(svm_models['Anger'], 'svm_model_anger.pkl')
joblib.dump(svm_models['Disgust'], 'svm_model_disgust.pkl')
joblib.dump(svm_models['Fear'], 'svm_model_fear.pkl')
joblib.dump(svm_models['Sadness'], 'svm_model_sadness.pkl')
joblib.dump(svm_models['Surprise'], 'svm_model_surprise.pkl')
joblib.dump(tfidf_vectorizer, 'tfidf_vectorizer.pkl')

    ['tfidf_vectorizer.pkl']

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_sco
# inisialisasi list untuk menyimpan hasil evaluasi
accuracy_scores = []
precision_scores = []
recall_scores = []
```

```
f1_scores = []

# loop untuk setiap model ovr
for label, model in svm_models.items():

    # prediksi label untuk data uji menggunakan model
    y_pred_label = model.predict(X_test_tfidf)

    # hitung confusion matrix
    conf_matrix = confusion_matrix(y_test[label], y_pred_label)

    # hitung evaluasi matrix
    accuracy = accuracy_score(y_test[label], y_pred_label)
    precision = precision_score(y_test[label], y_pred_label, zero_division=1)
    recall = recall_score(y_test[label], y_pred_label)
    f1 = f1_score(y_test[label], y_pred_label)

    # simpan skor evaluasi
    accuracy_scores.append(accuracy)
    precision_scores.append(precision)
    recall_scores.append(recall)
    f1_scores.append(f1)

    # output hasil evaluasi untuk setiap model
    print(f"Evaluasi untuk kelas {label}:")
    print("Confusion Matrix:")
    print(conf_matrix)
    print("Accuracy:", accuracy)
    print("Precision:", precision)
    print("Recall:", recall)
    print("F1 Score:", f1)
    print("\n")

# output rata-rata skor evaluasi untuk semua model
print("Rata-rata skor evaluasi untuk semua model:")
print("Accuracy:", sum(accuracy_scores) / len(accuracy_scores))
print("Precision:", sum(precision_scores) / len(precision_scores))
print("Recall:", sum(recall_scores) / len(recall_scores))
print("F1 Score:", sum(f1_scores) / len(f1_scores))
```

```
Evaluasi untuk kelas HS:
Confusion Matrix:
[[ 96  22]
 [ 35 147]]
Accuracy: 0.81
Precision: 0.8698224852071006
Recall: 0.8076923076923077
F1 Score: 0.8376068376068375
```

```
Evaluasi untuk kelas Non HS:
```

Confusion Matrix:

```
[[147  35]
```

```
 [ 22  96]]
```

Accuracy: 0.81

Precision: 0.732824427480916

Recall: 0.8135593220338984

F1 Score: 0.7710843373493977

Evaluasi untuk kelas Anticipation:

Confusion Matrix:

```
[[230  11]
```

```
 [ 41  18]]
```

Accuracy: 0.8266666666666667

Precision: 0.6206896551724138

Recall: 0.3050847457627119

F1 Score: 0.40909090909090917

Evaluasi untuk kelas Trust:

Confusion Matrix:

```
[[269   0]
```

```
 [ 29   2]]
```

Accuracy: 0.9033333333333333

Precision: 1.0

Recall: 0.06451612903225806

F1 Score: 0.12121212121212122

Evaluasi untuk kelas Joy:

Confusion Matrix:

```
[[277   0]
```

```
 [ 21   2]]
```

Accuracy: 0.93

Precision: 1.0

Recall: 0.08695652173913043

F1 Score: 0.16

Evaluasi untuk kelas Anger:

Confusion Matrix:

```
[[207  15]
```

```
 [ 54  24]]
```

Accuracy: 0.77

Precision: 0.6153846153846154

Recall: 0.3076923076923077

F1 Score: 0.4102564102564103

```
print(svm_models)
```

```
{ 'HS': SVC(kernel='linear', probability=True), 'Non HS': SVC(kernel='linear', probabi
```

```
import matplotlib.pyplot as plt
```

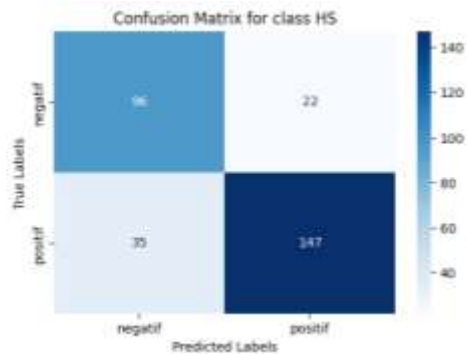
```
import
from sklearn.metrics import confusion_matrix

label = 'HS'

y_true_label = y_test[label]
y_pred_label = svm_models[label].predict(X_test_tfidf)

conf_matrix = confusion_matrix(y_true_label, y_pred_label)

plt.figure(figsize=(6, 4))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', cbar=True,
            xticklabels=['negatif', 'positif'], yticklabels=['negatif', 'positif'])
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title(f'Confusion Matrix for class {label}')
plt.show()
```



```
import pandas as pd

# Load the predictions file
predictions_path = 'svm_classification_predictions.xlsx'
```



```
predictions = pd.read_excel(predictions_path)

# Display the first few rows to understand its structure
print(predictions.head())

# Select relevant columns for correlation analysis
relevant_columns = ['HS', 'Anticipation', 'Trust', 'Joy', 'Anger', 'Disgust', 'Fear', 'Sad
```

```
# Create a subset with relevant columns
subset_predictions = predictions[relevant_columns]

# Calculate the correlation matrix
correlation_matrix = subset_predictions.corr()

correlation_matrix.fillna(0, inplace=True)

# Display the correlation matrix
print(correlation_matrix)

# Visualize the correlation matrix using a heatmap
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Matrix between Hate Speech and Emotions')
plt.show()
```

	HS	Mon	HS	Anticipation	Trust	Joy	Anger	Disgust	Fear	Sadness	%
0	0	1		0	0	0	0	0	0	0	
1	1	0		0	0	0	0	0	0	0	
2	0	1		1	0	0	0	0	0	0	
3	1	0		0	0	0	0	1	0	0	
4	1	0		0	0	0	0	0	0	0	

	Surprise
0	0
1	0
2	0
3	0
4	0

	HS	Anticipation	Trust	Joy	Anger	Disgust	%
HS	1.000000	-0.371554	-0.003050	-0.003050	0.348111	0.350295	
Anticipation	-0.371554	1.000000	-0.020799	-0.020799	-0.126412	-0.130114	
Trust	-0.003050	-0.020799	1.000000	-0.006711	-0.031600	-0.012095	
Joy	-0.003050	-0.020799	-0.006711	1.000000	-0.031600	-0.012095	
Anger	0.348111	-0.126412	-0.031600	-0.031600	1.000000	-0.120464	
Disgust	0.350295	-0.130114	-0.012095	-0.012095	-0.120464	1.000000	
Fear	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
Sadness	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
Surprise	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	

	Fear	Sadness	Surprise
HS	0.0	0.0	0.0
Anticipation	0.0	0.0	0.0
Trust	0.0	0.0	0.0
Joy	0.0	0.0	0.0
Anger	0.0	0.0	0.0
Disgust	0.0	0.0	0.0
Fear	0.0	0.0	0.0
Sadness	0.0	0.0	0.0
Surprise	0.0	0.0	0.0

