# Praktikum 11 Teks Classification: Naive Bayes, Rocchio, kNN, dan SVM


Nama : Raihan Rahmanda Junianto
NIM : 222112303
Kelas : 3SD2


---

## A. Penyiapan Library dan Dataset

In [ ]:
```
### 1. Tahap Persiapan
# Import package/library yang diperlukan
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# untuk pre-processing teks
import re, string

# bag of words
from sklearn.feature_extraction.text import TfidfVectorizer

#untuk pembangunan model
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, f1_score, accuracy_score, confus
```

In [ ]:
```
# Uncomment baris-baris berikut jika file data training disimpan di komputer
import os
# os.chdir('/Users/xxx/Documents/')
df_train=pd.read_csv('D:/RAIHAN STIS/Perkuliahan/SEMESTER 5/Praktikum INFORMATION R
print(df_train.shape)
df_train.head()
# Baris-baris berikut digunakan jika file data training disimpan di Google Drive
# from google.colab import drive
# drive.mount("/content/drive", force_remount=True)
# df_train=pd.read_csv('/content/drive/MyDrive/kuliah/Information Retrieval 22-23/B
# print(df_train.shape)
# df_train.head()
```

```
(3638, 2)
```

Out[ ]:

| | sentence | sentiment |
|---|---|---|
| 0 | Kangen NaBil @RealSyahnazS @bangbily RaGa @Raf... | 1 |
| 1 | Doa utk orang yg mberi makan: Ya Allah! Berila... | 1 |
| 2 | Setiap kali HP aku bunyi, aku selalu berharap ... | 1 |
| 3 | Belum pernah sedekat ini wawancara dgn Afgan S... | 1 |
| 4 | Dulu masa first pergi award show amatlah malas... | 1 |

In [ ]:
```
df_test=pd.read_csv('D:/RAIHAN STIS/Perkuliahan/SEMESTER 5/Praktikum INFORMATION RE
print(df_test.shape)
df_test.head()
```

```
(1011, 2)
```

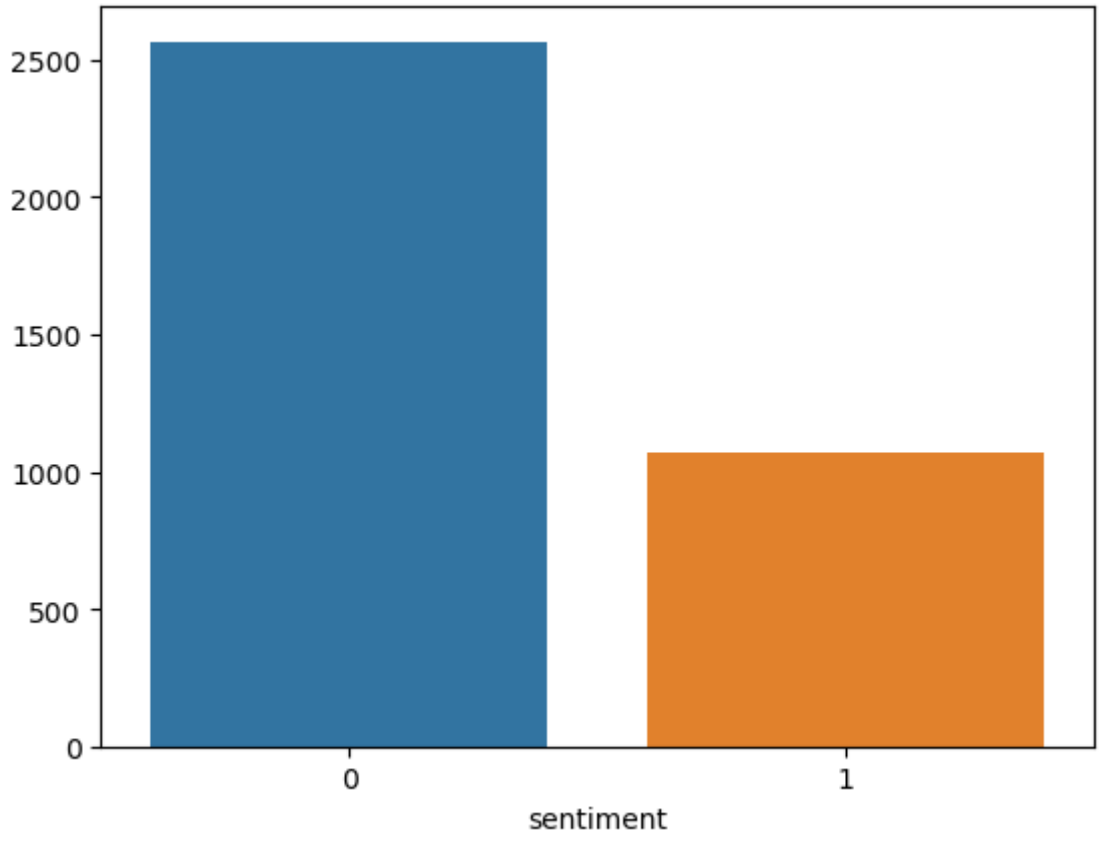
Out[ ]:

| | sentence | sentiment |
|---|---|---|
| **0** | #Sports Perempuan Golkar Makassar Dibekali Ilm... | 1 |
| **1** | Se-jauh"nya, Se-kenal"nya, Se-pisah"nya, Se-cu... | 1 |
| **2** | Sekedar Shared Ucapan Terimakasih Charles Hono... | 1 |
| **3** | Wah pak Jokowi sudah mendapat nilai positif di... | 1 |
| **4** | Penelpon : raffi ahmad oh raffi ahmad..... *bu... | 1 |

# B. Analisis Data Eksploratif

In [ ]:
```
# CLASS DISTRIBUTION
# mengecek apakah dataset yang digunakan balance atau tidak
x = df_train['sentiment'].value_counts()
print(x)
sns.barplot(x = x.index, y = x.values)  # Use x.values for the y-axis data
plt.show()
```

```
sentiment
0    2567
1    1071
Name: count, dtype: int64
```




In [ ]:
```
# Memeriksa missing values
df_train.isna().sum()
```

Out[ ]:
```
sentence     0
sentiment    0
dtype: int64
```

In [ ]:
```
#1. WORD-COUNT
print("Word Count")
```

```
df_train['word_count'] = df_train['sentence'].apply(lambda x:
len(str(x).split()))
print(df_train[df_train['sentiment']==1]['word_count'].mean()) #Positive
print(df_train[df_train['sentiment']==0]['word_count'].mean()) #Negative

#2. CHARACTER-COUNT
print("\nCharacter Count")
df_train['char_count'] = df_train['sentence'].apply(lambda x: len(str(x)))
print(df_train[df_train['sentiment']==1]['char_count'].mean()) #Positive
print(df_train[df_train['sentiment']==0]['char_count'].mean()) #Negative

#3. UNIQUE WORD-COUNT
print("\nUnique Word Count")
df_train['unique_word_count'] = df_train['sentence'].apply(lambda x:
len(set(str(x).split())))
#Positive
print(df_train[df_train['sentiment']==1]['unique_word_count'].mean())

#Negative
print(df_train[df_train['sentiment']==0]['unique_word_count'].mean())
```

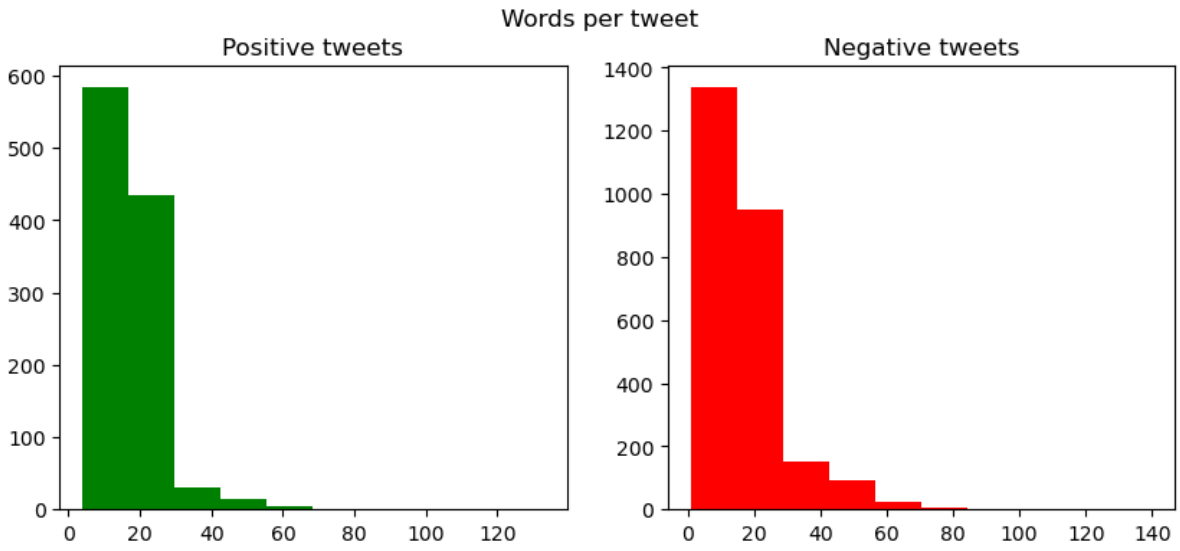
```
Word Count
16.985060690943044
16.684456564082588

Character Count
121.1484593837535
111.01051811453058

Unique Word Count
16.166199813258636
15.502532138683287
```

In [ ]:

```
# Plotting word-count per tweet
fig,(ax1,ax2)=plt.subplots(1,2,figsize=(10,4))
train_words=df_train[df_train['sentiment']==1]['word_count']
ax1.hist(train_words,color='green')
ax1.set_title('Positive tweets')
train_words=df_train[df_train['sentiment']==0]['word_count']
ax2.hist(train_words,color='red')
ax2.set_title('Negative tweets')
fig.suptitle('Words per tweet')
plt.show()
df_train=df_train.drop(columns=['word_count','char_count','unique_word_count'])
```



# C. Pre-processing Data

In [ ]:
```
# untuk pre-processing teks
#1. Common text preprocessing
text = "@user Teks ini mau dibersihkan. Ada beberapa karakter seperti: <br>, ?, :,
# mengubah ke huruf kecil (lowercase) dan menghapus tanda baca, karakter aneh dan s
def preprocess(text):
    text = text.lower() #Lowercase text
    text = text.strip() #Menghapus leading/trailing whitespace
    text = re.sub('@[^\s]+','atUser',text) #mengubah @user menjadi atUser
    text = re.sub(r'#([^\s]+)', r'\1', text) #menghapus hashtag di depan suatu kata
    text= re.compile('<.*?>').sub('', text) #Menghapus HTML tags/markups
    text = re.compile('[%s]' % re.escape(string.punctuation)).sub(' ', text)

    #Replace punctuation with space. Careful since punctuation can sometime be usef
    text = re.sub('\s+', ' ', text) #Menghapus extra space dan tabs
    text = re.sub(r'\[[0-9]*\]',' ',text) #[0-9] matches any digit (0 to 10000...)
    text = re.sub(r'[^\w\s]', '', str(text).strip())
    text = re.sub(r'\d',' ',text) #matches any digit from 0 to 100000..., \D matche
    text = re.sub(r'\s+',' ',text) #\s matches any whitespace, \s+ matches multiple
    return text

preprocess(text)
```

Out[ ]:
```
'atUser teks ini mau dibersihkan ada beberapa karakter seperti spasi berlebih dan
tab'
```

In [ ]:
```
def tokenisasi(text):
    tokens = text.split(" ")
    return tokens
#STOPWORD ELIMINATION DAN STEMMING
def stemming(text, stemmer):
    # stemming process
    output = stemmer.stem(text)
    return output
def stemming_stopword_elim(text, stopwords, stemmer):
    output = ""
    for token in tokenisasi(text):
        if not token in stopwords:
            output = output + stemming(token, stemmer) + " "
    return output[:-1]
```

In [ ]:
```
#FINAL PREPROCESSING
from spacy.lang.id import Indonesian
import spacy
nlp = Indonesian() # use directly
nlp = spacy.blank('id') # blank instance'
stopwords = nlp.Defaults.stop_words
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
# create stemmer
factory = StemmerFactory()
stemmer = factory.create_stemmer()
```

In [ ]:
```
# def finalpreprocess(string, stopwords, stemmer):
#     return stemming_stopword_elim(preprocess(string), stopwords, stemmer)

def finalpreprocess(string):
    return preprocess(string)
```

In [ ]:
```
df_train['clean_text'] = df_train['sentence'].apply(lambda x:finalpreprocess(x))
df_train.head()
```

Out[ ]:

| | sentence | sentiment | clean_text |
|---|---|---|---|
| **0** | Kangen NaBil @RealSyahnazS @bangbily RaGa @Raf... | 1 | kangen nabil atUser atUser raga atUser atUser ... |
| **1** | Doa utk orang yg mberi makan: Ya Allah! Berila... | 1 | doa utk orang yg mberi makan ya allah berilah ... |
| **2** | Setiap kali HP aku bunyi, aku selalu berharap ... | 1 | setiap kali hp aku bunyi aku selalu berharap i... |
| **3** | Belum pernah sedekat ini wawancara dgn Afgan S... | 1 | belum pernah sedekat ini wawancara dgn afgan s... |
| **4** | Dulu masa first pergi award show amatlah malas... | 1 | dulu masa first pergi award show amatlah malas... |

In [ ]:
```
df_test['clean_text'] = df_test['sentence'].apply(lambda x:finalpreprocess(x))
df_test.head()
```

Out[ ]:

| | sentence | sentiment | clean_text |
|---|---|---|---|
| **0** | #Sports Perempuan Golkar Makassar Dibekali Ilm... | 1 | sports perempuan golkar makassar dibekali ilmu... |
| **1** | Se-jauh"nya, Se-kenal"nya, Se-pisah"nya, Se-cu... | 1 | se jauh nya se kenal nya se pisah nya se cuek ... |
| **2** | Sekedar Shared Ucapan Terimakasih Charles Hono... | 1 | sekedar shared ucapan terimakasih charles hono... |
| **3** | Wah pak Jokowi sudah mendapat nilai positif di... | 1 | wah pak jokowi sudah mendapat nilai positif di... |
| **4** | Penelpon : raffi ahmad oh raffi ahmad..... *bu... | 1 | penelpon raffi ahmad oh raffi ahmad bukannya s... |

# D. Ekstraksi Feature dari Data Teks

In [ ]:
```
X_train = df_train['clean_text']
y_train = df_train['sentiment']
X_test = df_test['clean_text']
y_test = df_test['sentiment']
# TF-IDF
# Konversi x_train ke vector karena model hanya dapat memproses angka, bukan kata/k
tfidf_vectorizer = TfidfVectorizer(use_idf=True)
X_train_vectors_tfidf = tfidf_vectorizer.fit_transform(X_train)
# tfidf digunakan pada kalimat yang belum ditokenisasi, berbeda dengan word2vec
# Hanya men-transform x_test (bukan fit dan transform)
X_test_vectors_tfidf = tfidf_vectorizer.transform(X_test)
# Jangan melakukan fungsi fit() TfidfVectorizer ke data testing karena hal itu akan
# mengubah indeks kata & bobot sehingga sesuai dengan data testing. Sebaliknya, lak
# fungsi fit pada data training, lalu gunakan hasil model pada data training tadi p
# data testing untuk menunjukkan fakta bahwa Anda menganalisis data testing hanya
# berdasarkan apa yang dipelajari tanpa melihat data testing itu sendiri sebelumnya
```

# E. Pembangunan Model Klasifikasi Teks dengan Naive Bayes

In [ ]:
```
"""#### NB (tf-idf)"""
nb_tfidf = MultinomialNB()
nb_tfidf.fit(X_train_vectors_tfidf, y_train) #model

#Melakukan prediksi nilai y pada dataset testing
y_predict = nb_tfidf.predict(X_test_vectors_tfidf)
y_prob = nb_tfidf.predict_proba(X_test_vectors_tfidf)[:,1]
```

## F. Evaluasi Model Klasifikasi

In [ ]:
```
print(classification_report(y_test,y_predict))
print('Confusion Matrix: \n',confusion_matrix(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.79      0.96      0.87       713
           1       0.80      0.39      0.52       298

    accuracy                           0.79      1011
   macro avg       0.80      0.67      0.69      1011
weighted avg       0.79      0.79      0.76      1011

Confusion Matrix:
 [[685  28]
 [183 115]]
```

# Pengerjaan Praktikum 11

## A. Pembangunan Model Klasifikasi Teks dengan Rocchio Classification

In [ ]:
```
from sklearn.neighbors import NearestCentroid
rocchio_tfidf = NearestCentroid()
rocchio_tfidf.fit(X_train_vectors_tfidf, y_train) #model
#Melakukan prediksi nilai y pada dataset testing
y_predict = rocchio_tfidf.predict(X_test_vectors_tfidf)

print(classification_report(y_test,y_predict))
print('Confusion Matrix: \n',confusion_matrix(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.89      0.58      0.70       713
           1       0.45      0.84      0.59       298

    accuracy                           0.66      1011
   macro avg       0.67      0.71      0.65      1011
weighted avg       0.76      0.66      0.67      1011

Confusion Matrix:
 [[414 299]
 [ 49 249]]
```

## B. Pembangunan Model Klasifikasi Teks dengan kNN

In [ ]:
```
from sklearn.neighbors import KNeighborsClassifier
n_neighbors=5
knn_tfidf = KNeighborsClassifier(n_neighbors, weights='distance')
knn_tfidf.fit(X_train_vectors_tfidf, y_train) #model
#Melakukan prediksi nilai y pada dataset testing
y_predict = knn_tfidf.predict(X_test_vectors_tfidf)

print(classification_report(y_test,y_predict))
print('Confusion Matrix: \n',confusion_matrix(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.86      0.90      0.88       713
           1       0.73      0.66      0.69       298

    accuracy                           0.83      1011
   macro avg       0.79      0.78      0.78      1011
weighted avg       0.82      0.83      0.82      1011

Confusion Matrix:
 [[639  74]
 [102 196]]
```

# C. Pembangunan Model Klasifikasi Teks dengan SVM

In [ ]:
```
from sklearn import svm
svm_tfidf = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
svm_tfidf.fit(X_train_vectors_tfidf, y_train) #model
#Melakukan prediksi nilai y pada dataset testing
y_predict = svm_tfidf.predict(X_test_vectors_tfidf)

print(classification_report(y_test,y_predict))
print('Confusion Matrix: \n',confusion_matrix(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           0       0.88      0.89      0.89       713
           1       0.74      0.70      0.72       298

    accuracy                           0.84      1011
   macro avg       0.81      0.80      0.80      1011
weighted avg       0.84      0.84      0.84      1011

Confusion Matrix:
 [[638  75]
 [ 89 209]]
```

# D. Mencari Parameter Terbaik dengan Grid Search

C besar artinya semakin banyak support vector yang digunakan untuk membuat hyperplan.
Gamma besar artinya semakin tinggi bias, dan rendah variance

In [ ]:
```
from sklearn.model_selection import GridSearchCV
# defining parameter range
param_grid = {'C': [0.1, 1, 10, 100, 1000], 'gamma': [1, 0.1, 0.01, 0.001, 0.0001],
grid = GridSearchCV(svm.SVC(), param_grid, refit = True, verbose = 3)
# fitting the model for grid search
grid.fit(X_train_vectors_tfidf, y_train)
```

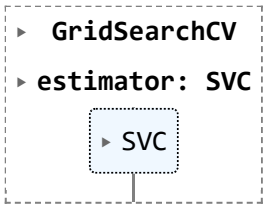
```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
[CV 1/5] END .....C=0.1, gamma=1, kernel=linear;, score=0.720 total time=   3.6s
[CV 2/5] END .....C=0.1, gamma=1, kernel=linear;, score=0.717 total time=   3.6s
[CV 3/5] END .....C=0.1, gamma=1, kernel=linear;, score=0.717 total time=   3.6s
[CV 4/5] END .....C=0.1, gamma=1, kernel=linear;, score=0.713 total time=   3.7s
[CV 5/5] END .....C=0.1, gamma=1, kernel=linear;, score=0.719 total time=   3.6s
[CV 1/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.705 total time=   4.3s
[CV 2/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.706 total time=   4.3s
[CV 3/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.706 total time=   4.3s
[CV 4/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.706 total time=   4.4s
[CV 5/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.706 total time=   4.4s
[CV 1/5] END ...C=0.1, gamma=0.1, kernel=linear;, score=0.720 total time=   3.5s
[CV 2/5] END ...C=0.1, gamma=0.1, kernel=linear;, score=0.717 total time=   3.5s
[CV 3/5] END ...C=0.1, gamma=0.1, kernel=linear;, score=0.717 total time=   3.5s
[CV 4/5] END ...C=0.1, gamma=0.1, kernel=linear;, score=0.713 total time=   3.5s
[CV 5/5] END ...C=0.1, gamma=0.1, kernel=linear;, score=0.719 total time=   3.6s
[CV 1/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.705 total time=   4.0s
[CV 2/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.706 total time=   3.7s
[CV 3/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.706 total time=   3.8s
[CV 4/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.706 total time=   3.7s
[CV 5/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.706 total time=   3.7s
[CV 1/5] END ..C=0.1, gamma=0.01, kernel=linear;, score=0.720 total time=   3.5s
[CV 2/5] END ..C=0.1, gamma=0.01, kernel=linear;, score=0.717 total time=   3.5s
[CV 3/5] END ..C=0.1, gamma=0.01, kernel=linear;, score=0.717 total time=   3.5s
[CV 4/5] END ..C=0.1, gamma=0.01, kernel=linear;, score=0.713 total time=   3.6s
[CV 5/5] END ..C=0.1, gamma=0.01, kernel=linear;, score=0.719 total time=   3.5s
[CV 1/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.705 total time=   3.7s
[CV 2/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.706 total time=   4.6s
[CV 3/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.706 total time=   4.0s
[CV 4/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.7s
[CV 5/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.7s
[CV 1/5] END .C=0.1, gamma=0.001, kernel=linear;, score=0.720 total time=   4.1s
[CV 2/5] END .C=0.1, gamma=0.001, kernel=linear;, score=0.717 total time=   4.2s
[CV 3/5] END .C=0.1, gamma=0.001, kernel=linear;, score=0.717 total time=   3.9s
[CV 4/5] END .C=0.1, gamma=0.001, kernel=linear;, score=0.713 total time=   4.3s
[CV 5/5] END .C=0.1, gamma=0.001, kernel=linear;, score=0.719 total time=   4.1s
[CV 1/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=0.705 total time=   3.3s
[CV 2/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.4s
[CV 3/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.4s
[CV 4/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 5/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.1s
[CV 1/5] END C=0.1, gamma=0.0001, kernel=linear;, score=0.720 total time=   3.6s
[CV 2/5] END C=0.1, gamma=0.0001, kernel=linear;, score=0.717 total time=   3.5s
[CV 3/5] END C=0.1, gamma=0.0001, kernel=linear;, score=0.717 total time=   3.5s
[CV 4/5] END C=0.1, gamma=0.0001, kernel=linear;, score=0.713 total time=   4.0s
[CV 5/5] END C=0.1, gamma=0.0001, kernel=linear;, score=0.719 total time=   3.6s
[CV 1/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=0.705 total time=   3.2s
[CV 2/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.1s
[CV 3/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.2s
[CV 4/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.1s
[CV 5/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.1s
[CV 1/5] END .......C=1, gamma=1, kernel=linear;, score=0.846 total time=   3.2s
[CV 2/5] END .......C=1, gamma=1, kernel=linear;, score=0.837 total time=   3.3s
[CV 3/5] END .......C=1, gamma=1, kernel=linear;, score=0.850 total time=   3.3s
[CV 4/5] END .......C=1, gamma=1, kernel=linear;, score=0.839 total time=   3.3s
[CV 5/5] END .......C=1, gamma=1, kernel=linear;, score=0.856 total time=   3.3s
[CV 1/5] END ..........C=1, gamma=1, kernel=rbf;, score=0.815 total time=   4.6s
[CV 2/5] END ..........C=1, gamma=1, kernel=rbf;, score=0.821 total time=   4.5s
[CV 3/5] END ..........C=1, gamma=1, kernel=rbf;, score=0.828 total time=   4.6s
[CV 4/5] END ..........C=1, gamma=1, kernel=rbf;, score=0.818 total time=   4.5s
[CV 5/5] END ..........C=1, gamma=1, kernel=rbf;, score=0.828 total time=   4.6s
[CV 1/5] END .....C=1, gamma=0.1, kernel=linear;, score=0.846 total time=   3.2s
[CV 2/5] END .....C=1, gamma=0.1, kernel=linear;, score=0.837 total time=   3.3s
[CV 3/5] END .....C=1, gamma=0.1, kernel=linear;, score=0.850 total time=   3.3s
```

```
[CV 4/5] END .....C=1, gamma=0.1, kernel=linear;, score=0.839 total time=   3.3s
[CV 5/5] END .....C=1, gamma=0.1, kernel=linear;, score=0.856 total time=   3.4s
[CV 1/5] END ........C=1, gamma=0.1, kernel=rbf;, score=0.751 total time=   3.6s
[CV 2/5] END ........C=1, gamma=0.1, kernel=rbf;, score=0.769 total time=   3.6s
[CV 3/5] END ........C=1, gamma=0.1, kernel=rbf;, score=0.772 total time=   3.6s
[CV 4/5] END ........C=1, gamma=0.1, kernel=rbf;, score=0.733 total time=   3.6s
[CV 5/5] END ........C=1, gamma=0.1, kernel=rbf;, score=0.761 total time=   3.6s
[CV 1/5] END ....C=1, gamma=0.01, kernel=linear;, score=0.846 total time=   3.2s
[CV 2/5] END ....C=1, gamma=0.01, kernel=linear;, score=0.837 total time=   3.3s
[CV 3/5] END ....C=1, gamma=0.01, kernel=linear;, score=0.850 total time=   3.3s
[CV 4/5] END ....C=1, gamma=0.01, kernel=linear;, score=0.839 total time=   3.2s
[CV 5/5] END ....C=1, gamma=0.01, kernel=linear;, score=0.856 total time=   3.3s
[CV 1/5] END .......C=1, gamma=0.01, kernel=rbf;, score=0.705 total time=   3.4s
[CV 2/5] END .......C=1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.6s
[CV 3/5] END .......C=1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.5s
[CV 4/5] END .......C=1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.5s
[CV 5/5] END .......C=1, gamma=0.01, kernel=rbf;, score=0.706 total time=   3.6s
[CV 1/5] END ...C=1, gamma=0.001, kernel=linear;, score=0.846 total time=   3.2s
[CV 2/5] END ...C=1, gamma=0.001, kernel=linear;, score=0.837 total time=   3.3s
[CV 3/5] END ...C=1, gamma=0.001, kernel=linear;, score=0.850 total time=   3.3s
[CV 4/5] END ...C=1, gamma=0.001, kernel=linear;, score=0.839 total time=   3.3s
[CV 5/5] END ...C=1, gamma=0.001, kernel=linear;, score=0.856 total time=   3.3s
[CV 1/5] END ......C=1, gamma=0.001, kernel=rbf;, score=0.705 total time=   3.3s
[CV 2/5] END ......C=1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 3/5] END ......C=1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 4/5] END ......C=1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 5/5] END ......C=1, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 1/5] END ..C=1, gamma=0.0001, kernel=linear;, score=0.846 total time=   3.2s
[CV 2/5] END ..C=1, gamma=0.0001, kernel=linear;, score=0.837 total time=   3.3s
[CV 3/5] END ..C=1, gamma=0.0001, kernel=linear;, score=0.850 total time=   3.3s
[CV 4/5] END ..C=1, gamma=0.0001, kernel=linear;, score=0.839 total time=   3.2s
[CV 5/5] END ..C=1, gamma=0.0001, kernel=linear;, score=0.856 total time=   3.3s
[CV 1/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=0.705 total time=   3.0s
[CV 2/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.0s
[CV 3/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.0s
[CV 4/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.1s
[CV 5/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.0s
[CV 1/5] END ......C=10, gamma=1, kernel=linear;, score=0.834 total time=   4.0s
[CV 2/5] END ......C=10, gamma=1, kernel=linear;, score=0.835 total time=   4.0s
[CV 3/5] END ......C=10, gamma=1, kernel=linear;, score=0.839 total time=   4.0s
[CV 4/5] END ......C=10, gamma=1, kernel=linear;, score=0.850 total time=   4.1s
[CV 5/5] END ......C=10, gamma=1, kernel=linear;, score=0.839 total time=   4.3s
[CV 1/5] END .........C=10, gamma=1, kernel=rbf;, score=0.839 total time=   5.5s
[CV 2/5] END .........C=10, gamma=1, kernel=rbf;, score=0.838 total time=   5.2s
[CV 3/5] END .........C=10, gamma=1, kernel=rbf;, score=0.841 total time=   5.1s
[CV 4/5] END .........C=10, gamma=1, kernel=rbf;, score=0.835 total time=   5.0s
[CV 5/5] END .........C=10, gamma=1, kernel=rbf;, score=0.853 total time=   5.1s
[CV 1/5] END ....C=10, gamma=0.1, kernel=linear;, score=0.834 total time=   4.1s
[CV 2/5] END ....C=10, gamma=0.1, kernel=linear;, score=0.835 total time=   4.2s
[CV 3/5] END ....C=10, gamma=0.1, kernel=linear;, score=0.839 total time=   4.1s
[CV 4/5] END ....C=10, gamma=0.1, kernel=linear;, score=0.850 total time=   4.1s
[CV 5/5] END ....C=10, gamma=0.1, kernel=linear;, score=0.839 total time=   4.1s
[CV 1/5] END .......C=10, gamma=0.1, kernel=rbf;, score=0.849 total time=   4.2s
[CV 2/5] END .......C=10, gamma=0.1, kernel=rbf;, score=0.842 total time=   4.3s
[CV 3/5] END .......C=10, gamma=0.1, kernel=rbf;, score=0.848 total time=   4.3s
[CV 4/5] END .......C=10, gamma=0.1, kernel=rbf;, score=0.856 total time=   4.3s
[CV 5/5] END .......C=10, gamma=0.1, kernel=rbf;, score=0.868 total time=   4.3s
[CV 1/5] END ...C=10, gamma=0.01, kernel=linear;, score=0.834 total time=   4.1s
[CV 2/5] END ...C=10, gamma=0.01, kernel=linear;, score=0.835 total time=   4.2s
[CV 3/5] END ...C=10, gamma=0.01, kernel=linear;, score=0.839 total time=   4.2s
[CV 4/5] END ...C=10, gamma=0.01, kernel=linear;, score=0.850 total time=   4.2s
[CV 5/5] END ...C=10, gamma=0.01, kernel=linear;, score=0.839 total time=   4.2s
[CV 1/5] END ......C=10, gamma=0.01, kernel=rbf;, score=0.760 total time=   3.6s
[CV 2/5] END ......C=10, gamma=0.01, kernel=rbf;, score=0.782 total time=   3.6s
```

```
[CV 3/5] END ......C=10, gamma=0.01, kernel=rbf;, score=0.777 total time=   3.7s
[CV 4/5] END ......C=10, gamma=0.01, kernel=rbf;, score=0.751 total time=   3.6s
[CV 5/5] END ......C=10, gamma=0.01, kernel=rbf;, score=0.769 total time=   3.7s
[CV 1/5] END ..C=10, gamma=0.001, kernel=linear;, score=0.834 total time=   4.1s
[CV 2/5] END ..C=10, gamma=0.001, kernel=linear;, score=0.835 total time=   4.2s
[CV 3/5] END ..C=10, gamma=0.001, kernel=linear;, score=0.839 total time=   4.2s
[CV 4/5] END ..C=10, gamma=0.001, kernel=linear;, score=0.850 total time=   4.1s
[CV 5/5] END ..C=10, gamma=0.001, kernel=linear;, score=0.839 total time=   4.1s
[CV 1/5] END .....C=10, gamma=0.001, kernel=rbf;, score=0.705 total time=   3.6s
[CV 2/5] END .....C=10, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.6s
[CV 3/5] END .....C=10, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.6s
[CV 4/5] END .....C=10, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.6s
[CV 5/5] END .....C=10, gamma=0.001, kernel=rbf;, score=0.706 total time=   3.7s
[CV 1/5] END .C=10, gamma=0.0001, kernel=linear;, score=0.834 total time=   4.1s
[CV 2/5] END .C=10, gamma=0.0001, kernel=linear;, score=0.835 total time=   4.2s
[CV 3/5] END .C=10, gamma=0.0001, kernel=linear;, score=0.839 total time=   4.2s
[CV 4/5] END .C=10, gamma=0.0001, kernel=linear;, score=0.850 total time=   4.2s
[CV 5/5] END .C=10, gamma=0.0001, kernel=linear;, score=0.839 total time=   4.2s
[CV 1/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=0.705 total time=   3.4s
[CV 2/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.4s
[CV 3/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.3s
[CV 4/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.4s
[CV 5/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.5s
[CV 1/5] END .....C=100, gamma=1, kernel=linear;, score=0.837 total time=   4.3s
[CV 2/5] END .....C=100, gamma=1, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END .....C=100, gamma=1, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END .....C=100, gamma=1, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END .....C=100, gamma=1, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ........C=100, gamma=1, kernel=rbf;, score=0.839 total time=   4.9s
[CV 2/5] END ........C=100, gamma=1, kernel=rbf;, score=0.838 total time=   4.9s
[CV 3/5] END ........C=100, gamma=1, kernel=rbf;, score=0.841 total time=   4.9s
[CV 4/5] END ........C=100, gamma=1, kernel=rbf;, score=0.835 total time=   4.9s
[CV 5/5] END ........C=100, gamma=1, kernel=rbf;, score=0.853 total time=   4.9s
[CV 1/5] END ...C=100, gamma=0.1, kernel=linear;, score=0.837 total time=   3.9s
[CV 2/5] END ...C=100, gamma=0.1, kernel=linear;, score=0.838 total time=   3.9s
[CV 3/5] END ...C=100, gamma=0.1, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END ...C=100, gamma=0.1, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END ...C=100, gamma=0.1, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ......C=100, gamma=0.1, kernel=rbf;, score=0.839 total time=   4.3s
[CV 2/5] END ......C=100, gamma=0.1, kernel=rbf;, score=0.839 total time=   4.3s
[CV 3/5] END ......C=100, gamma=0.1, kernel=rbf;, score=0.852 total time=   4.3s
[CV 4/5] END ......C=100, gamma=0.1, kernel=rbf;, score=0.856 total time=   4.3s
[CV 5/5] END ......C=100, gamma=0.1, kernel=rbf;, score=0.849 total time=   4.3s
[CV 1/5] END ..C=100, gamma=0.01, kernel=linear;, score=0.837 total time=   4.0s
[CV 2/5] END ..C=100, gamma=0.01, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END ..C=100, gamma=0.01, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END ..C=100, gamma=0.01, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END ..C=100, gamma=0.01, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END .....C=100, gamma=0.01, kernel=rbf;, score=0.845 total time=   4.1s
[CV 2/5] END .....C=100, gamma=0.01, kernel=rbf;, score=0.842 total time=   4.1s
[CV 3/5] END .....C=100, gamma=0.01, kernel=rbf;, score=0.848 total time=   4.1s
[CV 4/5] END .....C=100, gamma=0.01, kernel=rbf;, score=0.856 total time=   4.1s
[CV 5/5] END .....C=100, gamma=0.01, kernel=rbf;, score=0.867 total time=   4.3s
[CV 1/5] END .C=100, gamma=0.001, kernel=linear;, score=0.837 total time=   4.0s
[CV 2/5] END .C=100, gamma=0.001, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END .C=100, gamma=0.001, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END .C=100, gamma=0.001, kernel=linear;, score=0.847 total time=   4.2s
[CV 5/5] END .C=100, gamma=0.001, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ....C=100, gamma=0.001, kernel=rbf;, score=0.764 total time=   3.7s
[CV 2/5] END ....C=100, gamma=0.001, kernel=rbf;, score=0.782 total time=   3.7s
[CV 3/5] END ....C=100, gamma=0.001, kernel=rbf;, score=0.780 total time=   3.8s
[CV 4/5] END ....C=100, gamma=0.001, kernel=rbf;, score=0.751 total time=   3.7s
[CV 5/5] END ....C=100, gamma=0.001, kernel=rbf;, score=0.769 total time=   3.6s
[CV 1/5] END C=100, gamma=0.0001, kernel=linear;, score=0.837 total time=   4.0s
```

```
[CV 2/5] END C=100, gamma=0.0001, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END C=100, gamma=0.0001, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END C=100, gamma=0.0001, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END C=100, gamma=0.0001, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=0.705 total time=   3.5s
[CV 2/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.5s
[CV 3/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.6s
[CV 4/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.5s
[CV 5/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=0.706 total time=   3.5s
[CV 1/5] END ....C=1000, gamma=1, kernel=linear;, score=0.837 total time=   3.9s
[CV 2/5] END ....C=1000, gamma=1, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END ....C=1000, gamma=1, kernel=linear;, score=0.837 total time=   4.4s
[CV 4/5] END ....C=1000, gamma=1, kernel=linear;, score=0.847 total time=   4.3s
[CV 5/5] END ....C=1000, gamma=1, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END .......C=1000, gamma=1, kernel=rbf;, score=0.839 total time=   5.3s
[CV 2/5] END .......C=1000, gamma=1, kernel=rbf;, score=0.838 total time=   5.4s
[CV 3/5] END .......C=1000, gamma=1, kernel=rbf;, score=0.841 total time=   5.0s
[CV 4/5] END .......C=1000, gamma=1, kernel=rbf;, score=0.835 total time=   5.0s
[CV 5/5] END .......C=1000, gamma=1, kernel=rbf;, score=0.853 total time=   5.0s
[CV 1/5] END ..C=1000, gamma=0.1, kernel=linear;, score=0.837 total time=   4.0s
[CV 2/5] END ..C=1000, gamma=0.1, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END ..C=1000, gamma=0.1, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END ..C=1000, gamma=0.1, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END ..C=1000, gamma=0.1, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.839 total time=   4.3s
[CV 2/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.843 total time=   4.3s
[CV 3/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.849 total time=   4.3s
[CV 4/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.856 total time=   4.4s
[CV 5/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.847 total time=   4.4s
[CV 1/5] END .C=1000, gamma=0.01, kernel=linear;, score=0.837 total time=   4.0s
[CV 2/5] END .C=1000, gamma=0.01, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END .C=1000, gamma=0.01, kernel=linear;, score=0.837 total time=   4.1s
[CV 4/5] END .C=1000, gamma=0.01, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END .C=1000, gamma=0.01, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=0.835 total time=   4.1s
[CV 2/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=0.841 total time=   4.2s
[CV 3/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=0.837 total time=   4.1s
[CV 4/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=0.849 total time=   4.3s
[CV 5/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=0.828 total time=   4.1s
[CV 1/5] END C=1000, gamma=0.001, kernel=linear;, score=0.837 total time=   3.9s
[CV 2/5] END C=1000, gamma=0.001, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END C=1000, gamma=0.001, kernel=linear;, score=0.837 total time=   4.0s
[CV 4/5] END C=1000, gamma=0.001, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END C=1000, gamma=0.001, kernel=linear;, score=0.831 total time=   4.4s
[CV 1/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=0.845 total time=   3.7s
[CV 2/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=0.842 total time=   4.2s
[CV 3/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=0.848 total time=   4.1s
[CV 4/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=0.856 total time=   4.1s
[CV 5/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=0.867 total time=   4.2s
[CV 1/5] END C=1000, gamma=0.0001, kernel=linear;, score=0.837 total time=   4.0s
[CV 2/5] END C=1000, gamma=0.0001, kernel=linear;, score=0.838 total time=   4.0s
[CV 3/5] END C=1000, gamma=0.0001, kernel=linear;, score=0.837 total time=   4.1s
[CV 4/5] END C=1000, gamma=0.0001, kernel=linear;, score=0.847 total time=   4.1s
[CV 5/5] END C=1000, gamma=0.0001, kernel=linear;, score=0.831 total time=   4.0s
[CV 1/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=0.764 total time=   3.6s
[CV 2/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=0.782 total time=   3.6s
[CV 3/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=0.780 total time=   3.6s
[CV 4/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=0.751 total time=   3.6s
[CV 5/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=0.769 total time=   3.6s
```

Out[ ]:




Parameter terbaik hasil percobaan dengan Grid Search didapatkan dengan:

In [ ]:
```
# print best parameter after tuning
print(grid.best_params_)
# print how our model looks after hyper-parameter tuning
print(grid.best_estimator_)
```

```
{'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
SVC(C=10, gamma=0.1)
```

Selanjutnya, prediksi menggunakan parameter terbaik dapat dilakukan dengan kode berikut.

In [ ]:
```
grid_predictions = grid.predict(X_test_vectors_tfidf)
```

Metode klasifikasi SVM dengan C = 10, gamma = 0,1, dan kernel: rbf

In [ ]:
```
from sklearn import svm
svm_tfidf = svm.SVC(C=10, kernel='rbf', degree=3, gamma='auto')
svm_tfidf.fit(X_train_vectors_tfidf, y_train) #model
#Melakukan prediksi nilai y pada dataset testing
y_predict = svm_tfidf.predict(X_test_vectors_tfidf)

print(classification_report(y_test, grid_predictions))
print('Confusion Matrix: \n',confusion_matrix(y_test, grid_predictions))
```

```
              precision    recall  f1-score   support

           0       0.89      0.88      0.88       713
           1       0.71      0.74      0.73       298

    accuracy                           0.84      1011
   macro avg       0.80      0.81      0.80      1011
weighted avg       0.84      0.84      0.84      1011

Confusion Matrix:
 [[624  89]
 [ 77 221]]
```

# E. Menggunakan k-Fold Cross Validation

In [ ]:
```
X_join = pd.concat([X_train, X_test])
y_join = pd.concat([y_train, y_test])
X_join_vectors_tfidf = tfidf_vectorizer.transform(X_join)
```

Berikut kode yang digunakan untuk melakukan 5-fold cross validation, misalnya untuk model Naive Bayes.

In [ ]:
```
from sklearn.model_selection import cross_val_score, cross_val_predict, cross_valic
scores = cross_validate(nb_tfidf, X_join_vectors_tfidf, y_join, cv=5, scoring=('acc
predictions = cross_val_predict(nb_tfidf, X_join_vectors_tfidf, y_join, cv=5)
print(scores)
```

{'fit_time': array([0.00701976, 0.00599289, 0.00906825, 0.00904846, 0.01250291]),
'score_time': array([0.00649691, 0.0074904 , 0.01058197, 0.00855732, 0.00981069]),
'test_accuracy': array([0.81290323, 0.80860215, 0.82150538, 0.81397849, 0.7944025
8]), 'train_accuracy': array([0.87442861, 0.87335305, 0.87415972, 0.87684862, 0.88
602151]), 'test_f1': array([0.57971014, 0.57819905, 0.59708738, 0.57907543, 0.5260
5459]), 'train_f1': array([0.73749297, 0.73583847, 0.73589165, 0.74065685, 0.76779
847])}

# F. Interpretasi Hasil

Berdasarkan percobaan telah dilakukan diperoleh beberapa kesimpulan sebagai berikut.

1. Model klasifikasi Naive Bayes dan Support Vector Machine (SVM) memiliki akurasi yang lebih baik dibandingkan dengan Rocchio Classification dan kNN. Berdasarkan hasil percobaan, diperoleh nilai akurasi untuk kedua metode tersebut sebesar 0,84. Artinya, model klasifikasi tersebut mampu memprediksi kelas dari data teks dengan benar sebesar 84%. Nilai tersebut lebih baik dibandingkan dengan Rocchio Classification dan kNN yang hanya mampu memprediksi kelas dengan benar sebesar 0,66 dan 0,83.
2. Recall terbaik untuk kelas negatif (0) ditunjukkan oleh model klasifikasi kNN sebesar 0,9. Artinya, model klasifikasi tersebut mampu untuk mendeteksi data yang seharusnya masuk ke dalam kelas negatif sebesar 90% dari keseluruhan data kelas negatif. Sedangkan untuk kelas positif (1), recall terbaik ditunjukkan oleh model klasifikasi Rocchio sebesar 0,84. Artinya, model klasifikasi tersebut mampu untuk mendeteksi data yang seharusnya masuk ke dalam kelas positif sebesar 84% dari keseluruhan data kelas positif.
3. Precision terbaik untuk kelas negatif (0) ditunjukkan oleh model klasifikasi Rocchion sebesar 0,89. Artinya, model klasifikasi tersebut mampu untuk memprediksi data yang masuk ke dalam kelas negatif sebesar 89% dari keseluruhan data yang diprediksi masuk ke dalam kelas negatif. Sedangkan untuk kelas positif (1), precision terbaik ditunjukkan oleh model klasifikasi Naive Bayes dan SVM masing-masing sebesar 0,74. Artinya, model klasifikasi tersebut mampu untuk memprediksi data yang masuk ke dalam kelas positif sebesar 74% dari keseluruhan data yang diprediksi masuk ke dalam kelas positif.
4. Jika dilihat dari f-score, model klasifikasi Naive Bayes dan SVM juga memiliki nilai f-score yang lebih baik untuk masing-masing kelas. Pada kelas negatif (0), Naive Bayes dan SVM sama sama memiliki nilai f-score sebesar 0,89. Artinya, model klasifikasi tersebut mampu memprediksi kelas negatif dengan baik. Sedangkan pada kelas positif (1), NNaive Bayes dan SVM sama sama memiliki nilai f-score sebesar 0,72. Artinya, model klasifikasi tersebut juga mampu memprediksi kelas positif dengan baik. Hal tersebut berbeda dengan Rocchio Classification dan kNN yang memiliki nilai f-score masing-masing sebesar 0,59 dan 0,69.
5. Selanjutnya, diperoleh juga perbandingan SVM setelah memperoleh parameter terbaik, yaitu c = 10, gamma = 0.1, dan kernel: rbf. Mengggunakkan parameter tersebut akurasi dari SVM tetap sama, yaitu 0,84. Sedangkan, precission untuk kelas negatif (0) adalah sebesar 0,89 dan kelas positif (1) sebesar 0,71. Recall untuk kelas negatif (0) adalah sebesar 0,88 dan kelas posifif (1) sebesar 0,74. F-score untuk kelas negatif (0) adalah sebesar 0,88 dan kelas posifif (1) sebesar 0,73. Dari hasil tersebut, dapat disimpulkan bahwa SVM dengan parameter terbaik memiliki akurasi yang sama dengan SVM tanpa parameter terbaik. Namun, SVM dengan parameter terbaik memiliki nilai precision, recall, dan f-score yang berbeda dibandingkan dengan SVM tanpa parameter terbaik.

Hal tersebut menunjukkan bahwa parameter yang digunakan mempengaruhi nilai precision, recall, dan f-score dari model klasifikasi SVM.

Jadi, pemilihan metode klasifikasi yang tepat bergantung pada persyaratan spesifik dan trade-off dari kebutuhan pengguna.