Library

```
import pandas as pd
import re
import numpy as np
import sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import FeatureUnion
from sklearn.pipeline import Pipeline, make_pipeline
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.metrics import classification_report
from sklearn.feature_extraction import DictVectorizer
from sklearn.feature_extraction import text
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn import model_selection
from sklearn.metrics import confusion_matrix, precision_score, precision_recall_curve, recall
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
     Mounted at /content/drive
```

Prepare Dataset

Dataset yang digunakan berupa file 2 file CSV. Dimana file tersebut memiliki atribut-atribut sebagai berikut :

- artist : memuat nama penyanyi
- song : memuat judul lagu
- lirik : memuat lirik lagu
- Label : memuat kategori/label lagu (true : mengandung badwords, false : tidak mengandung badwords)

```
# dataset 1
```

```
df1 = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/ML Semester 5/TUBES/asset/lirik/son
df1 = df1[['artist','song','lirik','Label']]
df1 = df1.loc[df1['Label'] != 'no match']
#remove'\n' from the lyrics
re_drop = re.compile(r'\n')
df1[['lirik']] = df1[['lirik']].applymap(lambda x:re_drop.sub(' ',x))

df1
```

	artist	song	lirik	Label
0	Yura Yunita	Cinta dan Rahasia	Terakhir kutatap mata indahmu Di bawah bintang...	False
1	Kaleb J	Now I know	Aku tak menyadari kau t'lah menaruh hati Kepad...	False
2	Azmi	Pernah	Ada apa kau bertemu dia Mungkinkah kau ingin b...	False
3	Tulus	Pamit	Tubuh saling bersandar Ke arah mata angin berb...	False
4	Sheila on 7	Anugerah Terindah	Melihat tawamu Mendengar senandungmu Terlihat ...	False
...	...	...	...	...
127	Young Lex	Plastik	Alah paling kontroversi lagi ni Pansos lagi sa...	True
128	Achmad Sawadi	Lelaki Kardus	Bapakku kawin lagi Aku ditinggalin Aku sakit h...	True
129	The Panas Dalam	Seperti Seekor Babi	Rambutnya tipis jadi gitaris Seperti seekor ba...	True
130	Anjar Ox's	Ngacca Dulu	Pembenci menghina, gua lawan tertawa Lu mau ka...	True
131	Jason Ranti	Variasi Pink	Terjadi lagi malaikatku, terlambat datang Keba...	True

```
# dataset 2
df2 = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/ML Semester 5/TUBES/asset/lirik/sub
df2 = df2[['artist','song','text','explicit_label']]
df2 = df2.loc[df2['explicit_label'] != 'no match']
#remove'\n' from the lyrics
re_drop = re.compile(r'\n')
df2[['text']] = df2[['text']].applymap(lambda x:re_drop.sub(' ',x))
df2.rename(columns = {"text": "lirik", "explicit_label": "Label"}, inplace=True)

df2
```

	artist	song	lirik	Label
1	ABBA	Andante, Andante	Take it easy with me, please Touch me gently...	False
2	ABBA	As Good As New	I'll never know why I had to go Why I had to...	False
4	ABBA	Bang-A-Boomerang	Making somebody happy is a question of give an...	False
7	ABBA	Chiquitita	Chiquitita, tell me what's wrong You're ench...	False

```
# menggabungkan 2 dataframe
song_df = pd.merge(df1,df2,how="outer")
```

...	...	...	...	...

```
song_df
```

	artist	song	lirik	Label
0	Yura Yunita	Cinta dan Rahasia	Terakhir kutatap mata indahmu Di bawah bintang...	False
1	Kaleb J	Now I know	Aku tak menyadari kau t'lah menaruh hati Kepad...	False
2	Azmi	Pernah	Ada apa kau bertemu dia Mungkinkah kau ingin b...	False
3	Tulus	Pamit	Tubuh saling bersandar Ke arah mata angin berb...	False
4	Sheila on 7	Anugerah Terindah	Melihat tawamu Mendengar senandungmu Terlihat ...	False
...	...	...	...	...
24803	Zao	To Think Of You Is To Treasure An Absent Memory	When you shut your eyes and fell asleep Dark...	False
24804	Zebra	As I Said Before	And I said before I don't want no more And...	False

Preprocessing

Import Stopword

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

```
from nltk.corpus import stopwords

# menggunakan 2 bahasa karena dataset yang digunakan meliputi 2 bahasa tersebut
idn_stopwords = set(stopwords.words('indonesian'))
eng_stopwords = set(stopwords.words('english'))
```

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

```
filtering = set(idn_stopwords)
filtering.update(eng_stopwords)
```

```
filtering
```

```
{'a',
 'about',
 'above',
 'ada',
 'adalah',
 'adanya',
 'adapun',
 'after',
 'again',
 'against',
 'agak',
 'agaknya',
 'agar',
 'ain',
 'akan',
 'akankah',
 'akhir',
 'akhiri',
 'akhirnya',
 'aku',
 'akulah',
 'all',
 'am',
 'amat',
 'amatlah',
 'an',
 'and',
 'anda',
 'andalah',
 'antar',
 'antara',
 'antaranya',
 'any',
 'apa',
 'apaan',
 'apabila',
 'apakah',
 'apalagi',
 'apatah',
 'are',
 'aren',
 "aren't",
 'artinya',
 'as',
 'asal',
 'asalkan',
 'at',
 'atas',
 'atau',
 'ataukah',
 'ataupun',
 'awal',
 'awalnya',
 'bagai',
 'bagaikan',
 'bagaimana',
 'bagaimanakah',
 'bagaimanapun',
```

```
len(filtering)
```

```
936
```

Cleaning

```
#fungsi untuk menghapus semua karakter non-alfabet pada atribut lirik
def clean(text):
  text = re.sub('[^A-Za-z]+', ' ', text)
  return text
#lowercase
def casefolding(tweet):
  tweet = tweet.lower()
  tweet = tweet.strip(" ")
  tweet = re.sub(r'[?|$|.|!²_:")(-+.]','',tweet)
  return tweet

song_df['lirik'] = song_df['lirik'].apply(clean)
song_df['lirik'] = song_df['lirik'].apply(casefolding)
song_df
```

	artist	song	lirik	Label
0	Yura Yunita	Cinta dan Rahasia	terakhir kutatap mata indahmu di bawah bintang...	False
1	Kaleb J	Now I know	aku tak menyadari kau t lah menaruh hati kepad...	False
2	Azmi	Pernah	ada apa kau bertemu dia mungkinkah kau ingin b...	False
3	Tulus	Pamit	tubuh saling bersandar ke arah mata angin berb...	False

Mengatasi Ketidak-konsitenan pada atribute Label

on / senandungmu terlihat ...

```
for i in range(song_df.shape[0]):
    l = song_df['Label'][i]
    if l==False:
      l = 'False'
    elif l==True :
      l = 'True'
    song_df['Label'][i] = l
```

```
song_df['Label'].values
```

```
     array(['False', 'False', 'False', ..., 'False', 'False', 'False'],
           dtype=object)
```

```
song_df[(song_df['Label']=='False')].shape
```

```
     (23418, 4)
```

```
song_df[(song_df['Label']=='True')].shape
```

```
     (1390, 4)
```

Training

Split Data

```
song_df_1 = song_df.loc[song_df['Label'] == 'True']
song_df_0 = song_df.loc[song_df['Label'] == 'False']
song_df_0 = song_df_0.sample(n=23418, replace=False, random_state=100)

x = song_df_0[['artist','song','lirik']].append(song_df_1[['artist','song','lirik']])
```

```
y = song_df_0[['Label']].append(song_df_1[['Label']])

#train : test = 8 : 2
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y, test_size=4
```

```
x_train
```

	artist	song	lirik
2990	George Strait	If You Ain't Lovin' (You Ain't Livin')	if you got a cadillac boy and a room shack boy...
17846	Little Mix	Secret Love	when you hold me in the street and you kiss me...
4076	John Martyn	Hole In The Rain	between the drizzle and the drop between the d...
13574	Eric Clapton	Knockin' On Heaven's Door	ma take this badge off of me i can t use it an...
15196	Hanson	Tearing It Down	i am taking a chance walking with my laces loo...
...	...	...	...
14149	Freddie Aguilar	Anak Pawis	anak pawis ang tawag sa akin ako raw ay basaha...
18919	Misfits	Spinal Remains	this isn t really death this isn t really life...
16749	Judas Priest	Living After Midnight	living after midnight rockin to the dawn lovin...
			everybody always asks me how i got to play so

```
y_train
```

	Label
2990	False

```
x_test
```

	artist	song	lirik
21708	Roy Orbison	Indian Wedding	there once was an indian brave by the name of ...
11008	Blur	Young And Lovely	friday s child is planning to out for the firs...
17529	Kris Kristofferson	Shipwrecked In The Eighties	well you fight like the devil to just keep you...
7995	Steve Miller Band	Lovin' Cup	my mama she done told me soon you be a man and...
24798	Zao	All Else Failed	a throne in heaven sat empty for years why for...
...	...	...	...
18627	Metallica	Am I Evil?	my mother was a witch she was burned alive tha...
14498	George Strait	Good News Bad News	i ve got some good news can t wait to tell you...
1865	Dolly Parton	Home For Pete's Sake	i became a woman of the world cause i was fed ...
13780	Faith Hill	When The Lights Go Down	when the lights go down he ll be fillin a pan ...
19829	O.A.R.	King Of The Thing	it s been a long long time since i lost myself...

```
y_test
```

Label

```
# mengubah type data train_label, test_label, train_data, test_data
train_label = []
for i in range(len(y_train)):
    l = y_train.iloc[i,0]
    if l=='False':
      l = 0
    else :
      l = 1
    train_label.append(l)

test_label = []
for i in range(len(y_test)):
    l = y_test.iloc[i,0]
    if l=='False':
      l = 0
    else:
      l = 1
    test_label.append(l)

train_data = []
for i in range(len(x_train)):
    text = x_train.iloc[i,2]
    train_data.append(text)

test_data = []
for i in range(len(x_test)):
    text = x_test.iloc[i,2]
    test_data.append(text)
```

```
type(test_data)
```

```
list
```

Custom Feature

```
file1 = open('/content/drive/MyDrive/Colab Notebooks/ML Semester 5/TUBES/asset/badwords/indon
file2 = open('/content/drive/MyDrive/Colab Notebooks/ML Semester 5/TUBES/asset/badwords/badwo
file1 = list(file1)
file2 = list(file2)
```

```
bad_words= []
for w in file1:
    bad_words.append(re.sub(r'\n','',w))
for w in file2:
    bad_words.append(re.sub(r'\n','',w))
```

```
bad_words
```

```
     ['adult',
      'akouka',
      'alkohol',
      'anak haram',
      'anak yatim',
      'analex',
      'anjing',
      'anjink',
      'anjir',
      'arsundal',
      'asu',
      'autis',
      'azizay',
      'babi',
      'babi lu',
      'bacot',
      'bajingan',
      'bajingan tengik',
      'bakka',
      'banci',
      'bandar',
      'bangke',
      'bangsat',
      'bawel',
      'bebon',
      'bedebah',
      'bedon',
      'beer',
      'bego',
      'begok',
      'bencong',
      'berak',
      'bercinta',
      'berengsek',
      'bersetubuh',
      'bestiality',
      'betting',
      'biadab',
      'bispak',
      'bitch',
      "blo'on",
      'blowjob',
      'bo’ol',
      'bodo',
      'bodoh',
      'bodooohhh',
      'bokep',
      'boker',
      'bokong',
      'borok',
      'bot',
      'breast',
      'brengsek',
      'brengsex',
      'brengsexxx',
      'buah dada',
      'buah zakar',
      'buaya',
```

```
len(bad_words)
```

```
814
```

```
def get_bad_words(review):
  target_word = bad_words
  count = 0
  threshold = 0
  for t in target_word:
        if review.find(t) != -1:
            count += 1
  return count > threshold

def get_num_words(review):
  threshold = 0
  words = review.split(' ')
  count = len(list(words))
  return count > threshold


def find_bad_words(review,finded):
  target_word = bad_words
  count = 0
  finded = []
  for t in target_word:
        if review.find(t) != -1:
            finded.append(t)
  return finded


class CustomFeats(BaseEstimator, TransformerMixin):
    def __init__(self):
      self.feat_names = set()

    def fit(self, x, y=None):
        return self

    @staticmethod
    def features(review):
      return {
          'num_word': get_num_words(review),
          'bad_word': get_bad_words(review)
      }

    def get_feature_names(self):
        return list(self.feat_names)
    def transform(self, reviews):
      feats = []
      for review in reviews:
        f = self.features(review)
        [self.feat_names.add(k) for k in f]
        feats.append(f)
      return feats

#feats = make_pipeline(CustomFeats(), DictVectorizer())
feats = FeatureUnion([
     ('custom', make_pipeline(CustomFeats(), DictVectorizer())),
     ('bag_of_words', TfidfVectorizer(stop_words=filtering))
 ])
```

Model Klasifikasi

Algoritma yang diuji:

- Random forest
- KNN
- SVM
- Decision Tree **bold text**

```
def classification(feats, model):
  train_vecs = feats.fit_transform(train_data)
  test_vecs = feats.transform(test_data)

  model.fit(train_vecs, train_label)

  train_preds = model.predict(train_vecs)
  test_preds = model.predict(test_vecs)

  cm = confusion_matrix(test_label, test_preds)
  print("Confusion Matrix : \n", cm, " \n")

  report = classification_report(test_label, test_preds)
  print(report)

  return test_preds
```

Algoritma Random Forest

```
model_rf = RandomForestClassifier()
y_preds_rf = classification(feats, model_rf)
```

```
y_preds_rf
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/feature_extraction/text.py:401: UserWarni
  % sorted(inconsistent)
Confusion Matrix :
 [[4678    9]
 [ 170  104]]

              precision    recall  f1-score   support

           0       0.96      1.00      0.98      4687
           1       0.92      0.38      0.54       274

    accuracy                           0.96      4961
   macro avg       0.94      0.69      0.76      4961
weighted avg       0.96      0.96      0.96      4961

array([0, 0, 0, ..., 0, 0, 0])
```

Algoritma Klasifikasi KNN

```
model_knn= KNeighborsClassifier(n_neighbors=10)
y_preds_knn = classification(feats, model_knn)
y_preds_knn
```

```
Confusion Matrix :
 [[4674   13]
 [ 196   78]]

              precision    recall  f1-score   support

           0       0.96      1.00      0.98      4687
           1       0.86      0.28      0.43       274

    accuracy                           0.96      4961
   macro avg       0.91      0.64      0.70      4961
weighted avg       0.95      0.96      0.95      4961

array([0, 0, 0, ..., 0, 0, 0])
```

Algoritma Klasifikasi Decision Tree

```
model_dt = DecisionTreeClassifier(min_samples_split=0.4, max_depth=77)
y_preds_dt = classification(feats, model_dt)
y_preds_dt
```

```
Confusion Matrix :
 [[4599   88]
 [  84  190]]
```

```
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      4687
           1       0.68      0.69      0.69       274

    accuracy                           0.97      4961
   macro avg       0.83      0.84      0.84      4961
weighted avg       0.97      0.97      0.97      4961

array([0, 0, 0, ..., 0, 0, 0])
```

Algoritma Klasifikasi SVM

```
model_svm = SVC(C = 10000, kernel = 'rbf')
y_preds_svm = classification(feats, model_svm)
y_preds_svm
```

```
Confusion Matrix :
 [[4661   26]
 [ 142  132]]

              precision    recall  f1-score   support

           0       0.97      0.99      0.98      4687
           1       0.84      0.48      0.61       274

    accuracy                           0.97      4961
   macro avg       0.90      0.74      0.80      4961
weighted avg       0.96      0.97      0.96      4961

array([0, 0, 0, ..., 0, 0, 0])
```

Fungsi Model

Berdasarkan pengujian ke-4 algoritma, didapatkan bahwa performa algoritma klasifikasi decision tree lebih unggul dibandingkan yang lainnya dalam melakukan klasifikasi lirik. Maka fungsi model yang dibuat menggunakan algorirma decision tree

```
# lirik = ['love you']
# test_vecs = feats.transform(lirik)


# train_vecs = feats.fit_transform(train_data)


# model = DecisionTreeClassifier(min_samples_split=0.4, max_depth=77)
# model.fit(train_vecs, train_label)
# test_preds = model.predict(test_vecs)


# test_preds


def classification_model(test_data):
  teks= [test_data]
  train_vecs = feats.fit_transform(train_data)
  test_vecs = feats.transform(teks)
  model = DecisionTreeClassifier(min_samples_split=0.4, max_depth=77)
  model.fit(train_vecs, train_label)
  test_preds = model.predict(test_vecs)

  if test_preds == 0 :
    return ("This song doesn't contain any badwords")
  else :
    return ("This song contains any badwords")
  return test_preds
```

Clasify Test

```
singer = str(input('Penyanyi : '))
title = str(input('Judul Lagu : '))
lirik = str(input('Lirik Lagu : '))

finded = []
lirik = clean(lirik)
lirik = casefolding(lirik)
find = find_bad_words(lirik,finded)
result = classification_model(lirik)

print(result)
print('Badwords yang ditemukan : ', find)
```

```
Penyanyi : Young Lex
Judul Lagu : Anjing
Lirik Lagu : Like Kung fu Rap ku keras tanpa master wu Ku sapu Bukan ikan tapi debu Kari
This song contains any badwords
Badwords yang ditemukan :  ['anjing', 'anjir', 'asu', 'bitch', 'eek', 'ewe', 'fuck', 'ga
```

```
singer = str(input('Penyanyi : '))
title = str(input('Judul Lagu : '))
lirik = str(input('Lirik Lagu : '))

finded = []
```

```
lirik = clean(lirik)
lirik = casefolding(lirik)
find = find_bad_words(lirik,finded)
result = classification_model(lirik)

print(result)
print('Badwords yang ditemukan : ', find)
```

```
Penyanyi : Andmesh
Judul Lagu : Ku Mau Dia
Lirik Lagu : Kuharap semua ini bukan sekedar harapan Dan juga harapan ini bukan sekedar
This song doesn't contain any badwords
Badwords yang ditemukan :  []
```