```
pip install emoji
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting emoji
  Downloading emoji-2.0.0.tar.gz (197 kB)
     |████████████████████████████████| 197 kB 5.1 MB/s
Building wheels for collected packages: emoji
  Building wheel for emoji (setup.py) ... done
  Created wheel for emoji: filename=emoji-2.0.0-py3-none-any.whl size=193022 sha256=c0230beb5236ec196f6dbd89d226e2241a1
  Stored in directory: /root/.cache/pip/wheels/ec/29/4d/3cfe7452ac7d8d83b1930f8a6205c3c9649b24e80f9029fc38
Successfully built emoji
Installing collected packages: emoji
Successfully installed emoji-2.0.0
```

```
pip install contractions
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting contractions
  Downloading contractions-0.1.72-py2.py3-none-any.whl (8.3 kB)
Collecting textsearch>=0.0.21
  Downloading textsearch-0.0.21-py2.py3-none-any.whl (7.5 kB)
Collecting pyahocorasick
  Downloading pyahocorasick-1.4.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (106 kB)
     |████████████████████████████████| 106 kB 5.2 MB/s
Collecting anyascii
  Downloading anyascii-0.3.1-py3-none-any.whl (287 kB)
     |████████████████████████████████| 287 kB 42.3 MB/s
Installing collected packages: pyahocorasick, anyascii, textsearch, contractions
Successfully installed anyascii-0.3.1 contractions-0.1.72 pyahocorasick-1.4.4 textsearch-0.0.21
```

```
import pandas as pd
import numpy as np
import emoji
import contractions
import re
from bs4 import BeautifulSoup
import matplotlib.pyplot as plt
```

```
%matplotlib inline
import nltk

import nltk
nltk.download("punkt")
from nltk.tokenize import word_tokenize
nltk.download('stopwords')
from nltk.corpus import stopwords
nltk.download('wordnet')

from nltk.stem import LancasterStemmer, WordNetLemmatizer
from nltk import pos_tag
from nltk.corpus import wordnet
nltk.download('sentiwordnet')
from sklearn import svm
from sklearn.svm import SVC
from nltk.corpus import sentiwordnet as swn
import pickle

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import KFold, StratifiedKFold, cross_val_score
from sklearn.metrics import classification_report,confusion_matrix, accuracy_score
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package sentiwordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/sentiwordnet.zip.
```

Import Library

```
df = pd.read_csv('ebay_reviews.csv')
df
```

	category	review title	review content	rating
0	Headsets	Wireless gaming headset	This gaming headset ticks all the boxes # look...	5
1	Headsets	Good for those with a big head, low budget	Easy setup, rated for 6 hours battery but mine...	3
2	Headsets	MezumiWireless Gaming Headset	I originally bought this wireless headset for ...	5
3	Headsets	HW- S2 great headset.	This is my 2nd Mezumi headset, It kills the fi...	5
4	Headsets	BEST HEADPHONES I'VE PURCHASED IN MY ENTIRE LIFE	This is probably the best headset I've purchas...	5
...	...	...	...	...
44751	Racks & Holders	Utensil holder	Reasonably priced but a little flimsy	3
44752	Racks & Holders	Recommended	As described	5
44753	Racks &	cheap looking	cheap looking	1

Preprocessing

1. Cleaning Data

```
def html_remover(data):
    beauti = BeautifulSoup(data,'html.parser')
    return beauti.get_text()
```

```
def convert_emoji(data):
    return emoji.demojize(data)

def url_remover(data):
    return re.sub('(http|https):\/\/\S+', '',data)

def remove_round_brackets(data):
    return re.sub('\(.*?\)','',data)

def remove_punc(data):
    document = re.sub(r'[^\w\s]','', data)
    return document

def white_space(data):
    return ' '.join(data.split())

def text_lower(data):
    return data.lower()

def contraction_replace(data):
    return contractions.fix(data)

def remove_number(data):
    return re.sub(r"\d+", "", data)

def remove_singl_char(data):
    return re.sub(r"\b[a-zA-Z]\b", "", data)

def web_associated(data):
    new_data = html_remover(data)
    new_data = convert_emoji(new_data)
    new_data = url_remover(new_data)
    new_data = remove_round_brackets(new_data)
    new_data = remove_punc(new_data)
    new_data = white_space(new_data)
    new_data = text_lower(new_data)
    new_data = contraction_replace(new_data)
    new_data = remove_number(new_data)
    new_data = remove_singl_char(new_data)
    return new_data

df['cleaning data'] = df['review content'].apply(web_associated)
```

```
/usr/local/lib/python3.7/dist-packages/bs4/__init__.py:273: UserWarning: "b'.'" looks like a filename, not markup. You
  ' Beautiful Soup.' % markup)
/usr/local/lib/python3.7/dist-packages/bs4/__init__.py:273: UserWarning: "b'..'" looks like a filename, not markup. You
  ' Beautiful Soup.' % markup)
```

```
df.head()
```

	category	review title	review content	rating	cleaning data
0	Headsets	Wireless gaming headset	This gaming headset ticks all the boxes # look...	5	this gaming headset ticks all the boxes looks ...
1	Headsets	Good for those with a big head, low budget	Easy setup, rated for 6 hours battery but mine...	3	easy setup rated for hours battery but mine h...
2	Headsets	MezumiWireless Gaming Headset	I originally bought this wireless headset for ...	5	originally bought this wireless headset for m...
[illegible]	[illegible]	[illegible]	This is my 2nd Mezumi headset, It	[illegible]	this is my nd mezumi headset it

2. Tokenizing

```
def tokenize(data):
    return nltk.word_tokenize(data)
```

```
df['tokenizing'] = df['cleaning data'].apply(tokenize)
```

3. Negation Handling

```
import nltk
nltk.download('omw-1.4')

def Negation(sentence):
    temp = int(0)
    for i in range(len(sentence)):
        if sentence[i-1] in ['not',"n't"]:
            antonyms = []
            for syn in wordnet.synsets(sentence[i]):
                syns = wordnet.synsets(sentence[i])
                w1 = syns[0].name()
                temp = 0
                for l in syn.lemmas():
                    if l.antonyms():
                        antonyms.append(l.antonyms()[0].name())
                max_dissimilarity = 0
                for ant in antonyms:
                    syns = wordnet.synsets(ant)
                    w2 = syns[0].name()
                    syns = wordnet.synsets(sentence[i])
                    w1 = syns[0].name()
                    word1 = wordnet.synset(w1)
                    word2 = wordnet.synset(w2)
                    if isinstance(word1.wup_similarity(word2), float) or isinstance(word1.wup_similarity(word2), int):
                        temp = 1 - word1.wup_similarity(word2)
                    if temp>max_dissimilarity:
                        max_dissimilarity = temp
                        antonym_max = ant
                        sentence[i] = antonym_max
                        sentence[i-1] = ''
    while '' in sentence:
        sentence.remove('')
    return sentence
```

```
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
```

```
df['negation'] = df['tokenizing'].apply(Negation)
```

Stopword

```
def stopword(data):
    nltk.download('stopwords')
    clean = []
    for i in data:
        if i not in stopwords.words('english'):
            clean.append(i)
    return clean
```

```
df['stopword'] = df['negation'].apply(stopword)
```

```
Streaming output truncated to the last 5000 lines.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
```

```
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

4. Lemmatization

[ ] ↳ *3 cells hidden*

5. Pos Tagging

```
#POS tagger dictionary
post_dict = {'J':wordnet.ADJ, 'V':wordnet.VERB, 'N':wordnet.NOUN, 'R':wordnet.ADV}

def pos_tagging(tokens):
    return pos_tag(tokens)
```

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

df['pos_tag'] = df['lemma'].apply(pos_tagging)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

```
df
```

	category	review title	review content	rating	cleaning data	tokenizing	negation	stopword	lemma	pos_t
0	Headsets	Wireless gaming headset	This gaming headset ticks all the boxes # look...	5	this gaming headset ticks all the boxes looks ...	[this, gaming, headset, ticks, all, the, boxes...	[this, gaming, headset, ticks, all, the, boxes...	[gaming, headset, ticks, boxes, looks, grate, ...	[game, headset, tick, box, look, grate, build,...	[(gam NN (heads NN), (tic NN), (box,
1	Headsets	Good for those with a big head, low budget	Easy setup, rated for 6 hours battery but mine...	3	easy setup rated for hours battery but mine h...	[easy, setup, rated, for, hours, battery, but,...	[easy, setup, rated, for, hours, battery, but,...	[easy, setup, rated, hours, battery, mine, las...	[easy, setup, rat, hours, battery, mine, last,...	[(easy, J (setup, NN (rat, NN (hours, N
2	Headsets	MezumiWireless Gaming Headset	I originally bought this wireless headset for ...	5	originally bought this wireless headset for m...	[originally, bought, this, wireless, headset, ...	[originally, bought, this, wireless, headset, ...	[originally, bought, wireless, headset, xbox, ...	[originally, buy, wireless, headset, xbox, plu...	[(original RB), (bu VE (wireles JJ),
3	Headsets	HW- S2 great	Mezumi	5	headset it	nd, mezumi,	mezumi	headset	headset,	NN

```
df.to_csv('preprocessing1.csv', index=False)
```

```
df = pd.read_csv('preprocessing1.csv')
df
```

	category	review title	review content	rating	cleaning data	tokenizing	negation	stopword	lemma	po
0	Headsets	Wireless gaming headset	This gaming headset ticks all the boxes # look...	5	this gaming headset ticks all the boxes looks ...	['this', 'gaming', 'headset', 'ticks', 'all', ...	['this', 'gaming', 'headset', 'ticks', 'all', ...	['gaming', 'headset', 'ticks', 'boxes', 'looks...	['game', 'headset', 'tick', 'box', 'look', 'gr...	[(('he 'NN')
1	Headsets	Good for those with a big head, low budget	Easy setup, rated for 6 hours battery but mine...	3	easy setup rated for hours battery but mine h...	['easy', 'setup', 'rated', 'for', 'hours', 'ba...	['easy', 'setup', 'rated', 'for', 'hours', 'ba...	['easy', 'setup', 'rated', 'hours', 'battery',...	['easy', 'setup', 'rat', 'hours', 'battery', '...	[('easy ('setup ('rat'
2	Headsets	MezumiWireless Gaming Headset	I originally bought this wireless headset for	5	originally bought this wireless headset	['originally', 'bought', 'this',	['originally', 'bought', 'this',	['originally', 'bought', 'wireless',	['originally', 'buy', 'wireless',	[('orig 'RB')

Pelabelan

TextBlob

[ ] ↳ *5 cells hidden*

Vader Sentiment

[ ] ↳ *8 cells hidden*

Split Data

```
X_train, X_test, y_train, y_test = train_test_split(data_label.values, label, test_size = 0.20, random_state = 42)
y_actual_train, y_actual_test = train_test_split(label, test_size = 0.20, random_state = 42)
```

```
X_test.shape
```

```
(8952,)
```

EKSTRAKSI FITUR: Term presence

```
count_vect = CountVectorizer(binary=True)
X_train_counts = count_vect.fit_transform(data_label)
print(X_train_counts.shape)
count_vect.vocabulary_
```

```
(44756, 32506)
{'this': 28335,
 'gaming': 12677,
 'headset': 13937,
 'ticks': 28445,
 'all': 2633,
 'the': 28164,
 'boxes': 4907,
 'looks': 17150,
 'grate': 13269,
 'built': 5221,
 'to': 28587,
 'last': 16517,
 'excellent': 10872,
 'sound': 26303,
 'mic': 18103,
 'comfortable': 6712,
 'wear': 30685,
 'easy': 9927,
 'set': 25191,
 'up': 29734,
 'what': 30830,
 'more': 18586,
 'could': 7552,
 'you': 31594,
 'ask': 3424,
 'for': 12051,
 'setup': 25213,
 'rated': 22793,
 'hours': 14392,
 'battery': 4097,
 'but': 5327,
 'mine': 18233,
 'has': 13798,
 'lasted': 16519,
 'sessions': 25190,
 'over': 20155,
 'good': 13065,
 'loudness': 17213,
 'from': 12338,
 'earcups': 9875,
 'seal': 24921,
 'and': 2896,
 'thick': 28296,
 'padding': 20344,
 'around': 3323,
 'ears': 9903,
 'person': 20864,
 'with': 31098,
 'med': 17872,
 'big': 4457,
 'head': 13903,
 'would': 31276,
 'look': 17139,
 'funny': 12469,
 'on': 19788,
 'small': 25935,
```

```
X_train_TP = count_vect.transform(X_train)
X_test_TP = count_vect.transform(X_test)
```

```
print(X_train_TP)
```

```
  (0, 2633)	1
  (0, 2896)	1
```

```
(0, 4097)     1
(0, 4152)     1
(0, 4457)     1
(0, 5031)     1
(0, 6024)     1
(0, 7554)     1
(0, 8221)     1
(0, 12051)    1
(0, 12410)    1
(0, 13065)    1
(0, 13375)    1
(0, 13696)    1
(0, 13737)    1
(0, 14053)    1
(0, 15551)    1
(0, 18333)    1
(0, 18586)    1
(0, 18871)    1
(0, 19216)    1
(0, 19224)    1
(0, 21051)    1
(0, 23893)    1
(0, 24817)    1
:     :
(35803, 13841)        1
(35803, 14872)        1
(35803, 15323)        1
(35803, 15613)        1
(35803, 16876)        1
(35803, 16990)        1
(35803, 17224)        1
(35803, 18954)        1
(35803, 19865)        1
(35803, 20958)        1
(35803, 21929)        1
(35803, 24508)        1
(35803, 24683)        1
(35803, 24974)        1
(35803, 25223)        1
(35803, 26112)        1
(35803, 28164)        1
(35803, 28489)        1
```

```
  (35803, 28587)	1
  (35803, 30154)	1
  (35803, 30850)	1
  (35803, 31098)	1
  (35803, 31224)	1
  (35803, 31259)	1
  (35803, 31594)	1
```

KLASIFIKASI dengan term presence

```
SVM_Clasifier = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')

# fitting data latih pada classifier
SVM_Clasifier.fit(X_train_TP,y_train)
# memprediksi label pada set data uji
predictions_SVM_TP = SVM_Clasifier.predict(X_test_TP)

# Menggunakan fungsi accuracy_score untuk mendapat nilai akurasi
print('Confusion Matrix: \n',confusion_matrix(y_test, predictions_SVM_TP))
print()
print('Accuracy: ', accuracy_score(y_test, predictions_SVM_TP))
```

EKSTRAKSI FITUR: TF-IDF

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(data_label)
features_train_transformed = vectorizer.transform(X_train)
features_test_transformed = vectorizer.transform(X_test)
print(features_train_transformed)
```

KLASIFIKASI dengan TF-IDF

```
SVM_Clasifier = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
mymodel= SVM_Clasifier.fit(features_train_transformed,y_train)
```

```
#Evaluate the model on the training data set

predictions_SVM_Tfidf2 = SVM_Clasifier.predict(features_train_transformed)
print(classification_report(y_train, predictions_SVM_Tfidf2))
print('Confusion Matrix: \n',confusion_matrix(y_train, predictions_SVM_Tfidf2))
print()
print('Accuracy: ', accuracy_score(y_train, predictions_SVM_Tfidf2))
```

```
#Evaluate the model on the testing data set

predictions_SVM_Tfidf = SVM_Clasifier.predict(features_test_transformed)
print(classification_report(y_test, predictions_SVM_Tfidf))
print('Confusion Matrix: \n',confusion_matrix(y_test, predictions_SVM_Tfidf))
print()
print('Accuracy: ', accuracy_score(y_test, predictions_SVM_Tfidf))
```

Evaluation with K-Fold and Classification Report:

```
from sklearn.model_selection import KFold

# vectorizer = TfidfVectorizer()
# X = vectorizer.fit_transform(data_label)
kf = KFold(n_splits=10, shuffle=True, random_state=42)
scores = []
for fold, (train_index, test_index) in enumerate(kf.split(data_label,label), 1):
    X_train, X_test = data_label[train_index], data_label[test_index]
    y_train, y_test = label[train_index], label[test_index]
    vectorizer = TfidfVectorizer()
    vectorizer.fit_transform(data_label)
    features_train_transformed = vectorizer.transform(X_train)
    features_test_transformed = vectorizer.transform(X_test)

    SVM_Clasifier = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
    SVM_Clasifier.fit(features_train_transformed,y_train)
    predictions_SVM_Tfidf = SVM_Clasifier.predict(features_test_transformed)

    print(f'# Fold {fold}, Train set: {len(train_index)}, Test set:{len(test_index)}')
    print(classification_report(y_test, predictions_SVM_Tfidf), "\n")
```

```
# Fold 1, Train set: 40280, Test set:4476
              precision    recall  f1-score   support

    Negative       0.89      0.36      0.51       184
     Neutral       0.92      0.90      0.91       509
    Positive       0.96      0.99      0.98      3783

    accuracy                           0.96      4476
   macro avg       0.92      0.75      0.80      4476
weighted avg       0.95      0.96      0.95      4476


# Fold 2, Train set: 40280, Test set:4476
              precision    recall  f1-score   support

    Negative       0.84      0.35      0.50       217
     Neutral       0.92      0.88      0.90       504
    Positive       0.96      0.99      0.97      3755

    accuracy                           0.95      4476
   macro avg       0.90      0.74      0.79      4476
weighted avg       0.95      0.95      0.94      4476


# Fold 3, Train set: 40280, Test set:4476
              precision    recall  f1-score   support

    Negative       0.84      0.35      0.49       217
     Neutral       0.90      0.89      0.89       494
    Positive       0.96      0.99      0.97      3765

    accuracy                           0.95      4476
   macro avg       0.90      0.74      0.78      4476
weighted avg       0.94      0.95      0.94      4476


# Fold 4, Train set: 40280, Test set:4476
              precision    recall  f1-score   support

    Negative       0.90      0.40      0.56       204
     Neutral       0.92      0.90      0.91       494
    Positive       0.96      0.99      0.98      3778

    accuracy                           0.96      4476
   macro avg       0.93      0.76      0.81      4476
weighted avg       0.95      0.96      0.95      4476


# Fold 5, Train set: 40280, Test set:4476
              precision    recall  f1-score   support

    Negative       0.89      0.45      0.60       214
     Neutral       0.93      0.91      0.92       498
    Positive       0.96      0.99      0.98      3764

    accuracy                           0.96      4476
   macro avg       0.93      0.78      0.83      4476
weighted avg       0.96      0.96      0.95      4476
```

```
print('\n\nCross-Validation accuracy: %.3f +/- %.3f' %(np.mean(scores), np.std(scores)))
```

```
Cross-Validation accuracy: nan +/- nan
/usr/local/lib/python3.7/dist-packages/numpy/core/fromnumeric.py:3441: RuntimeWarning: Mean of empty slice.
  out=out, **kwargs)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:189: RuntimeWarning: invalid value encountered in double_
  ret = ret.dtype.type(ret / rcount)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:263: RuntimeWarning: Degrees of freedom <= 0 for slice
  keepdims=keepdims, where=where)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:223: RuntimeWarning: invalid value encountered in true_di
```

```
  subok=False)
/usr/local/lib/python3.7/dist-packages/numpy/core/_methods.py:254: RuntimeWarning: invalid value encountered in double_
  ret = ret.dtype.type(ret / rcount)
```

```
review_input_transformed = vectorizer.transform(data_label)
SVM_prediction = SVM_Clasifier.predict(review_input_transformed)
```

```
import collections, numpy
print("Jumlah review: {}".format(len(data_label)))
SVM = collections.Counter(predictions_SVM_Tfidf)
print("Hasil Klasifikasi SVM : ", SVM)

results = pd.DataFrame({
    "Labeled_Data" : data_label,
    "Label" : SVM_prediction
 })
results.to_csv("Hasil_SVM2.csv", index = False)
```

```
Jumlah review: 44756
Hasil Klasifikasi SVM :  Counter({'Positive': 3899, 'Neutral': 492, 'Negative': 84})
```

```
X_test.shape
```