```
# Text Classification

# importing the libraries

import numpy as np
import re
import pickle
import nltk
from nltk.corpus import stopwords
from sklearn.datasets import load_files
nltk.download('stopwords')

# importing datasets
reviews = load_files('txt_sentoken/')
X,y = reviews.data,reviews.target

# storing as pickle files
with open('X.pickle','wb') as f:
    pickle.dump(X,f)

with open('y.pickle','wb') as f:
    pickle.dump(y,f)

# unpickling the dataset
with open('X.pickle','rb') as f:
    X = pickle.load(f)

with open('y.pickle','rb') as f:
    y = pickle.load(f)

# preprocessing the data

# creating the corpus
corpus = []
for i in range(0,len(X)):
    review = re.sub(r'\W',' ',str(X[i]))
    review = review.lower()
    review = re.sub(r'\s+[a-z]\s+',' ',review)
    review = re.sub(r'^[a-z]\s+',' ',review)
    review = re.sub(r'\s+',' ',review)
    corpus.append(review)

from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(max_features=2000,min_df = 5, max_df =0.6, stop_words = stopwords.word
X = vectorizer.fit_transform(corpus).toarray()


from sklearn.feature_extraction.text import TfidfTransformer
transformer = TfidfTransformer()
X = transformer.fit_transform(X).toarray()

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(max_features=2000,min_df = 5, max_df =0.6, stop_words = stopwords.word
X = vectorizer.fit_transform(corpus).toarray()


from sklearn.model_selection import train_test_split
```

```
text_train,text_test,sent_train,sent_test = train_test_split(X,y,test_size=0.2,random_state = 0)


from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
classifier.fit(text_train,sent_train)

sent_pred = classifier.predict(text_test)

from sklearn.metrics import confusion_matrix
cm = confusion_matrix(sent_test,sent_pred)

# pickling the classifier
with open('classifier.pickle','wb') as f:
    pickle.dump(classifier,f)

# pickling the vectorizer
with open('tfidfmodel.pickle','wb') as f:
    pickle.dump(vectorizer,f)

# unpickling the classifier and vectorizer
with open('classifier.pickle','rb') as f:
    clf = pickle.load(f)

with open('tfidfmodel.pickle','rb') as f:
    tfidf = pickle.load(f)

sample = ["you are a nice person, have a good life"]
sample = tfidf.transform(sample).toarray()
print(clf.predict(sample))


sample = ["he is a bad person"]
sample = tfidf.transform(sample).toarray()
print(clf.predict(sample))
```