def extract\_features(tweet):

tweet\_words = set(tweet)

features = {}

for word in word\_features:

features['contains(%s)' % word] = (word in tweet\_words)

return features

word\_features = buildVocabulary(preprocessedTrainingData)

trainingFeatures = nltk.classify.apply\_features(extract\_features, preprocessedTrainingData)

NBResultLabels = [NBayesClassifier.classify(extract\_features(tweet[0])) for tweet in preprocessedTestDataSet]

# get the majority vote

if NBResultLabels.count('positive') > NBResultLabels.count('negative'):

print("Overall Positive Sentiment")

print("Positive Sentiment Percentage = " + str(100\*NBResultLabels.count('positive')/len(NBResultLabels)) + "%")

else:

print("Overall Negative Sentiment")

print("Negative Sentiment Percentage = " + str(100\*NBResultLabels.count('negative')/len(NBResultLabels)) + "%")

NBayesClassifier = nltk.NaiveBayesClassifier.train(trainingFeatures)