FINAL PROJECT

CIS668: Natural Language Processing

Instructor: Nancy McCracken

Topic:

Option 2: Processing and Classification of Sentiment or other Data

Dataset: Kaggle competition movie review phrase data, labeled for sentiment

Author: Manjunath Panduranga

SUiD: 637425699

Manjunath Panduranga
SUiD: 637425699

Project Overview:

I have chosen the topic of processing and classification of sentiment with the dataset of Kaggle movie review phrase data. For this project on Kaggle movie review, I have focused more on the creation of my own functions to carry out some experiments on the data set. At each level of analysis, I have used filtering of the data set using NLTK-Stopword filter, punctuation filters and NTLK-porter stemmer and "bag of words" features to obtain more accurate results.

Further I have applied some function features to the dataset and then determine the polarity by labelling it as strong positive, positive, neutral, negative, strong negative. Then I used the training data train.tsv and test data test.tsv to be analyzed by Navies Bayes classifier which has 156,060 phrases in the training data file.

As provided in the document, I have performed various experiments to analyze the features function with various classifiers. Experiment-1 uses NLTK Naïve Bayes classifier with 3 folds to find accuracy and cross validation of different sizes of words. Experiment-2 compares the NLTK Naïve Bayes classifier with and without stop words. Experiment-3 uses weka-explorer. Experiment-4 and 5 uses sci-it learn for different classifiers. Experiment-6 compares the accuracy of Naïve Bayes classifier with decision tree and entropy classifier. Experiment-7 compare the accuracy of Naïve Bayes classifier with Bi-gram tree classifiers.

Detailed illustration of Project:

Step 1: Processing of data

In this step I have made all the words to lower case, then removed all the punctuations and also all the stop words from the texts. And then I have used the WordNet Lemmatize.

The steps are explained as follows:

1) converting all the texts to Lower cases
   Reason: Lower casing of all words in the texts will not create two different types of identity for same word (as different word). Hence lower casing will lead to more meaningful ratios

2) Removal of the non-alphabetical characters from the text

Reason: since there are more non-alpha characters present in the text which includes these non-alpha numeric will affect the top frequency lists and it also results in accurate frequency ratio of these words in the bigrams list. Hence, we need to obtain more accurate frequencies of meaningful words

3) Removal of the stopwords from the text
Reason: The stop words, if they are not eliminated from the text, they will result in inaccurate ratio of frequencies, which in turn will make impact on bigrams as well. And the actual words that are being used in the text, will not be included in the top frequency lists due to the more usage of stop words. Hence, I am eliminating them.

4) Usage of WordNet Lemmatizer
Reason: To reduce the derivationally related forms of a word to a common base form, I have used the wordnet lemmatizer. By using this, it would result in the calibration of more accurate frequency ratio of actual words. For Ex.: words like smallest, smaller, small in the text would be converted one simple word like 'small' and will eventually result in the calibration of the frequency of actual word in the text.

Step 2: features and feature functions

Here I have created a features function that assigns a particular score to a token based on its polarity defined in the dictionary '.tff' file. I have created four labels here namely weakpos, weakneg, strongpos, strongneg for weak positive, weak negative, strong positive and strong negative tokens respectively and assigned a score value of 2, -2, 4, -4 respectively.

Secondly, I used the .tff file to classify a token as weakpos or weakneg or strongpos or strongneg and assigned scores for tokens in a given sentence or document. Then by summing the score values of all tokens of a sentence I arrived at the final score of a sentence which was then used to classify the sentence label as Strong Negative or Strong Positive or Weak Negative or Weak Positive or Neutral based on the below equivalently segregated ranges.

Further I used the **NLTK Naïve Bayes classifier** to train and test a classifier on your feature sets. I used cross-validation to obtain precision, recall and F-measure scores. I also produced the features as a csv file and then used **Weka** and **Sci-Kit Learn** to train and test a classifier, using cross-validation scores.

Step 3: Experiments

These are the experiments I have performed:

Experiment 1:

I have performed the accuracy and cross validation analysis with three folds (num_folds = 3) for vocabulary of three different sizes. By taking all the pre-processing filters into consideration (including Stop-Words). Here I have used NLTK Naïve Bayes classifier.

These are the different sizes I have used:

Most_Common_50_words

Most_Common_2500_words

Most_Common_5000_words

Most_Common_50_words :

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 50000 E:\myFinalProject/new50words.csv
Read 156060 phrases, using 50000 random phrases
--------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5397333333333333
--------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
--------------------------------------------------------------------------------
----------------------------<< CROSS VALIDATION >>----------------------------------
--------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.26898326898326896       0.008831809462119136       0.01710208814763926

 Positive       :  0.33164332850474815       0.03791713395650587       0.06805362472461145

 Neutral        :  0.5906210230130146       0.8718215064432676       0.7041864547138267

 Negative       :  0.40657505391553656       0.3792646176239402       0.39244527336378593

Strong Negative :  0.3713492496363844       0.08586341844246632       0.13947695781835087

--------------------------------------------------------------------------------
> Feature sets transferred to CSV file
```

Most_Common_2500_words :

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 50000 E:\myFinalProject/new2500words.csv
Read 156060 phrases, using 50000 random phrases
---------------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5458
---------------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
---------------------------------------------------------------------------------------
---------------------------<< CROSS VALIDATION >>--------------------------------------
---------------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.37076906712051905       0.017557765421072728       0.03352782120815616

 Positive       :  0.4811965811965811       0.01201351331406063       0.023441781095828132

 Neutral        :  0.5879392317973688       0.8739189556487124       0.7029563385144896

 Negative       :  0.4105551054378371       0.38037872660503674       0.3948912586133395

Strong Negative :  0.3932384413594338       0.10086222410108281       0.16054584245738146

---------------------------------------------------------------------------------------
> Feature sets transferred to CSV file
```

Most_Common_5000_words:

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 50000 E:\myFinalProject/new5000words.csv
Read 156060 phrases, using 50000 random phrases
---------------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5424
---------------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
---------------------------------------------------------------------------------------
---------------------------<< CROSS VALIDATION >>--------------------------------------
---------------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.28703703703703703       0.002409603642236357       0.00477908804384824

 Positive       :  0.5972222222222222       0.0030116233240984006       0.005993025509840305

 Neutral        :  0.5882930775189904       0.8736569561639195       0.7031243580093829

 Negative       :  0.4135930914717887       0.39018597961495005       0.40154871248316093

Strong Negative :  0.38210804228580336       0.09066178134615283       0.14655163696442153

---------------------------------------------------------------------------------------
> Feature sets transferred to CSV file
```

Observations from Expt-1:

MostCommon_50_words have the accuracy of 0.539733

MostCommon_2500_words have the accuracy of 0.5458

MostCommon_5000_words have the accuracy of 0.5424

Experiment 2:

I have performed the accuracy and cross validation analysis with three folds (num_folds = 3) for vocabulary of three different sizes. But in this experiment, I have **excluded the *Stop-Word filter***. And considered all the other filters. Here I have used NLTK Naïve Bayes classifier.

Here in this experiment I am comparing the accuracy and cross validation of using with Stop-word and accuracy and cross validation of without using Stop-word filter.

With_StopWord_Filter:

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 50000 E:\myFinalProject/new5000words.csv
Read 156060 phrases, using 50000 random phrases
---------------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5424
---------------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
---------------------------------------------------------------------------------------
---------------------------<< CROSS VALIDATION >>--------------------------------------
---------------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.28703703703703703       0.002409603642236357       0.00477908804384824

 Positive       :  0.5972222222222222       0.0030116233240984006       0.005993025509840305

 Neutral        :  0.5882930775189904       0.8736569561639195       0.7031243580093829

 Negative       :  0.4135930914717887       0.39018597961495005       0.40154871248316093

Strong Negative :  0.38210804228580336       0.09066178134615283       0.14655163696442153

---------------------------------------------------------------------------------------
> Feature sets transferred to CSV file
```

Without_StopWord_Filter:

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 50000 E:\myFinalProject/new5000_NoStopWord.csv
Read 156060 phrases, using 50000 random phrases
---------------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5416
---------------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
---------------------------------------------------------------------------------------
----------------------------<< CROSS VALIDATION >>-------------------------------------
---------------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.25368407179512525       0.01580540295822873       0.029756850299818838

 Positive       :  0.3724279835390946       0.013127933771527859       0.02536187195190117

 Neutral        :  0.5881171624030511       0.8724172903148162       0.7025970257073031

 Negative       :  0.4142847455328383       0.382362677528556       0.39768414476868214

Strong Negative :  0.3757947566510696       0.09353695407403566       0.14979041940220741

---------------------------------------------------------------------------------------
> Feature sets transferred to CSV file
```

Observations from Expt-2:

Accuracy with Stop-Word filter is 0.542

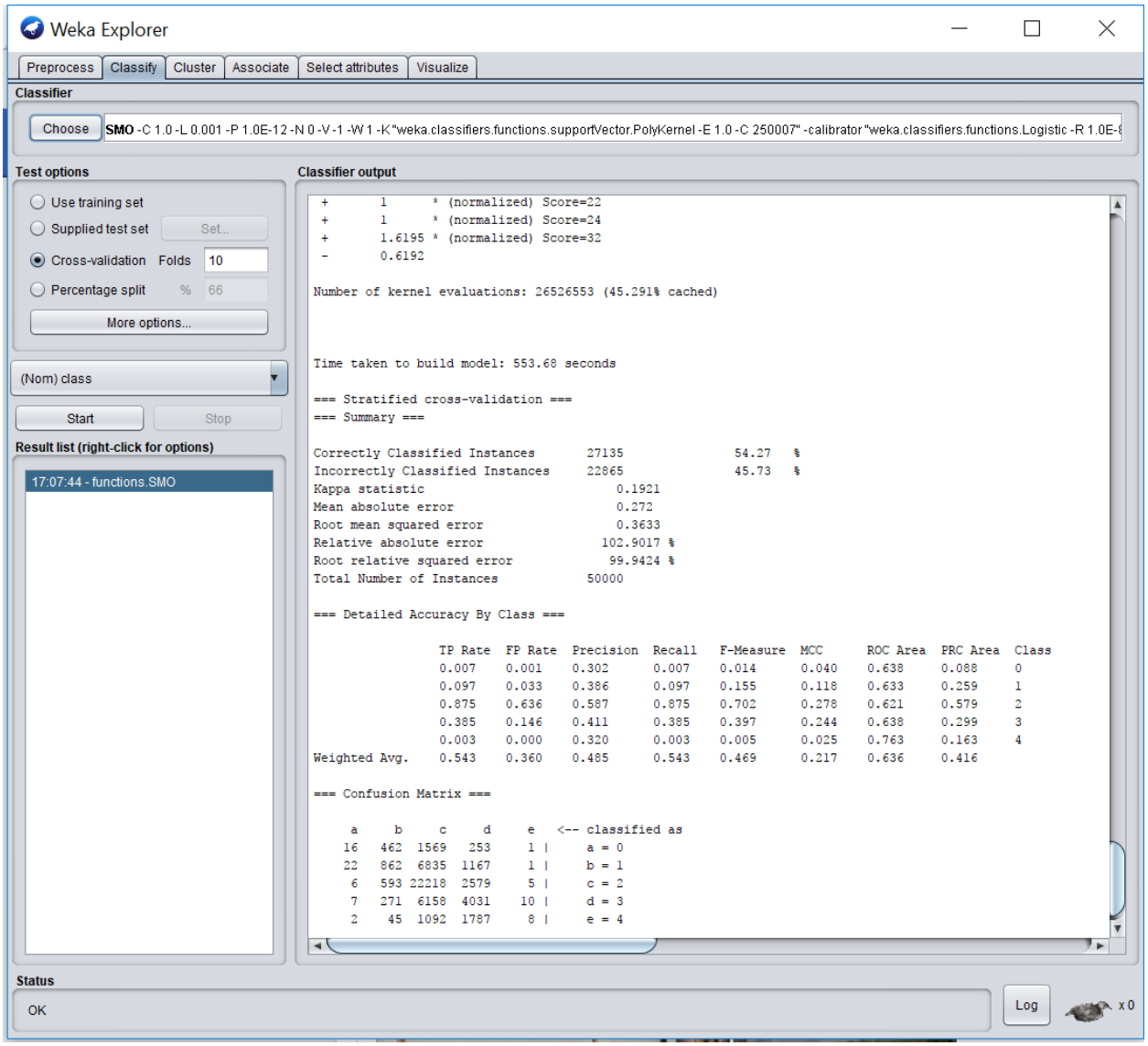
Accuracy without Stop-Word filter is 0.5416

Experiment 3: (using weka-Explorer)

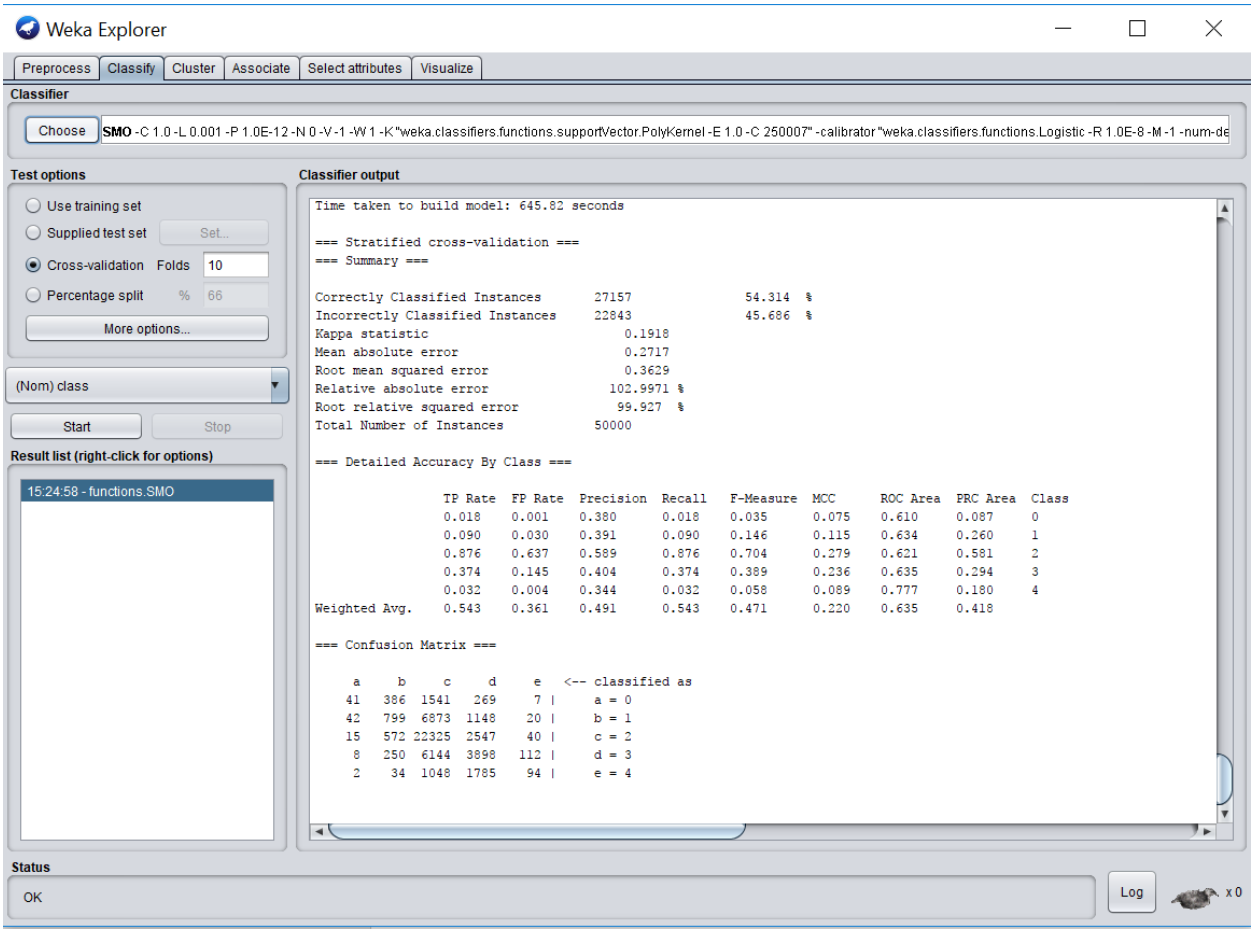
In this Experiment I have installed the **Weka Explorer** and then I have used 10 folds (num_folds = 10) to perform cross validation on these classifiers and generated the Confusion Matrix and results of the predicted and actual data.

The results of weka-explorer with stop-word and without stop-word filter is shown as below:

Weka explorer: With Stop-words Filter



Weka explorer: Without_StopWords Filter



Experiment 4: (using Sci-kit Learn)

I have the features with all the pre-processing filters including the 'Stop-Words' from NLTK as a .csv file using the function supplied i.e. 'writeFeatureSets' and then here I have used the **Sci-kit Learn** to train and test two classifiers i.e. 'LinearSVC' and 'LogisticRegression'.

Then I have used 10 folds (num_folds = 10) to perform cross validation on these classifiers and generated the Confusion Matrix and results of the predicted and actual data.

Logistic Regression:

```
  'precision', 'predicted', average, warn_for)
             precision    recall  f1-score   support

          0       0.12      0.58      0.20      2301
          1       0.00      0.00      0.00      8887
          2       0.69      0.65      0.67     25401
          3       0.35      0.16      0.22     10477
          4       0.18      0.61      0.28      2934

avg / total       0.44      0.43      0.41     50000



Predicted      0      2     3     4    All
Actual
0           1324    554   168   255   2301
1           3803   3200   718  1166   8887
2           4276  16601  1941  2583  25401
3           1410   3314  1712  4041  10477
4            255    494   389  1796   2934
All        11068  24163  4928  9841  50000

(E:\Anaconda3) E:\myFinalProject\ExternalClassifier>
```

Linear SVC:

```
  'precision', 'predicted', average, warn_for)
             precision    recall  f1-score   support

          0       0.21      0.19      0.20      2301
          1       0.00      0.00      0.00      8887
          2       0.58      0.88      0.70     25401
          3       0.00      0.00      0.00     10477
          4       0.18      0.61      0.28      2934

avg / total       0.32      0.49      0.38     50000



Predicted     0      2     4    All
Actual
0           428   1618   255   2301
1           778   6943  1166   8887
2           530  22288  2583  25401
3           239   6197  4041  10477
4            39   1099  1796   2934
All        2014  38145  9841  50000

(E:\Anaconda3) E:\myFinalProject\ExternalClassifier>
```

Experiment 5:

Here I have repeated the experiment 3, but in this case without using the **stop-word filter**. I have considered all the remaining pre-processing filters except for stop-words.

Logistic Regression: Without_StopWords Filter

```
E:\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1135: UndefinedMetricWa
 ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
             precision    recall  f1-score   support

          0       0.11      0.56      0.19      2244
          1       0.00      0.00      0.00      8882
          2       0.69      0.65      0.67     25499
          3       0.35      0.16      0.22     10412
          4       0.19      0.63      0.29      2963

avg / total       0.44      0.43      0.41     50000



Predicted      0      2     3     4    All
Actual
0           1267    542   159   276   2244
1           3800   3206   708  1168   8882
2           4367  16665  1880  2587  25499
3           1367   3367  1667  4011  10412
4            219    481   382  1881   2963
All        11020  24261  4796  9923  50000

(E:\Anaconda3) E:\myFinalProject\ExternalClassifier>
```

Linear SVC: Without_StopWords Filter

```
  'precision', 'predicted', average, warn_for)
             precision    recall  f1-score   support

          0       0.23      0.13      0.17      2244
          1       0.00      0.00      0.00      8882
          2       0.58      0.89      0.70     25499
          3       0.00      0.00      0.00     10412
          4       0.19      0.63      0.29      2963

avg / total       0.32      0.50      0.38     50000



Predicted     0      2     4    All
Actual
0           299   1669   276   2244
1           492   7222  1168   8882
2           330  22582  2587  25499
3           143   6258  4011  10412
4            21   1061  1881   2963
All        1285  38792  9923  50000

(E:\Anaconda3) E:\myFinalProject\ExternalClassifier>
```

Experiment 6:

I have performed the accuracy analysis for bi-gram feature function on NaivesBayes Classifier and compared it with feature function of Naïve Bayes Classifier in the observation column.

In the observation column we can view the accuracy of both the Naïve Bayes Classifier and Bi-gram features.

```
(E:\Anaconda3) E:\myFinalProject\kagglemoviereviews>python classifykaggle.py E:\myFinalProject\kagglemoviereviews\corpus
 500 E:\myFinalProject/new5000.csv
Read 156060 phrases, using 500 random phrases
--------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.56
--------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
--------------------------------------------------------------------------------
----------------------------<< CROSS VALIDATION >>------------------------------
--------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.3333333333333333       0.037037037037037035       0.06666666666666667

 Positive       :  0.0       0.0       Fmeasure is 0

 Neutral        :  0.5733845969577196       0.8915854676724241       0.6979273998527462

 Negative       :  0.5166167166167166       0.3990740740740741       0.4503012150629418

Strong Negative :  0.1619047619047619       0.05411255411255411       0.0811146101727264

--------------------------------------------------------------------------------
> Feature sets transferred to CSV file
--------------------------------------------------------------------------------
Bigram - Featuresets Naive Bayes Classifier Accuracy : 0.52
--------------------------------------------------------------------------------
```

Observations from Expt-6:

Naïve Bayes Classifier has the accuracy of 0.56

Bi-gram feature function have the accuracy of 0.52

Experiment 7:

I have performed the accuracy analysis for 2 more classifiers along with Naives Bayes Classifier. i.e I have included Decision Tree Classifier and Maximum Entropy Classifier. Later I compared it with feature function of Naïve Bayes Classifier in the observation column.

```
Read 156060 phrases, using 50000 random phrases
--------------------------------------------------------------------------------
Naive Bayes Classifier Accuracy for Training/Test Set given : 0.5362666666666667
--------------------------------------------------------------------------------
Start Fold 0
Start Fold 1
Start Fold 2
Done with cross-validation
--------------------------------------------------------------------------------
----------------------------<< CROSS VALIDATION >>--------------------------------
--------------------------------------------------------------------------------

                 << PRECISION MEAN >>        << RECALL MEAN >>         << F-MEASURE >>

Strong Positive :  0.460752688172043       0.02358450913390819       0.0448721512331086

 Positive       :  0.18357487922705315       0.002667243992206105       0.005258090761367409

 Neutral        :  0.5880718052638522       0.875975270945335       0.7037155666949483

 Negative       :  0.40966340381383975       0.3795894347591931       0.39405344471455606

Strong Negative :  0.3824587123191869       0.08911130231745168       0.1445443640587677

--------------------------------------------------------------------------------
> Feature sets transferred to CSV file
--------------------------------------------------------------------------------
Decision Tree Classifier Accuracy : 0.5364666666666666
--------------------------------------------------------------------------------

  ==> Training (1 iterations)

      Iteration    Log Likelihood    Accuracy
      ---------------------------------------
            1          -1.60944        0.058
        Final          -0.90494        0.546
--------------------------------------------------------------------------------
Max Entropy Classifier Accuracy : 0.5367333333333333
--------------------------------------------------------------------------------
```

Observations from Expt-7:

Naïve Bayes Classifier has the accuracy of 0.536266667

Decision Tree Classifier has the accuracy of 0.5364666

Max Entropy Classifier has the accuracy of 0.53673333

Conclusion:

I have performed 7 experiments on the dataset of Kaggle movie review by making various changes to the filters, introducing different classifiers, bag of words and used Naive Bayes Classifier, weka-explorer and ski-kit learn to compare the accuracy level of data. From my observation I can conclude that our feature functions are generating the accuracy levels of in and around 0.53 for various changes.

The End