Assignment 7
Advanced NLP
Akshay Reddy (reddyak@iu.edu)

INTRODUCTION:

The goal of this report is to highlight the process of building a sentiment analysis tool using the movies reviews given by the users. Naive Bayes techniques is used to build the model. The report also highlights some inaccuracies.

DATA COLLECTION AND MODIFICATIONS:

I have decided to used the Large Movie Review Dataset. This is a dataset for binary sentiment classification which has 25,000 highly polar movie reviews for training, and 25,000 for testing.

Source: **http://ai.stanford.edu/~amaas/data/sentiment/**
Modification: I have merged train and test data provide, which I will be splitting programmatically while calculating the accuracy.

With the help of python script I created a file **movie_reviews.txt** that has two columns: polarity (1 indicated positive sentiment and 0 indicates negative sentiment) and review

```
In [34]: df = pd.read_csv("movie_reviews.txt",sep = '\t', names = ['polarity','review'])
```

```
In [89]: df.sample(10)
```

Out[89]:

	polarity	review
3547	1	Corean cinema can be quite surprising for an o...
21010	0	The title should have been "The walker". That ...
10109	1	I had the pleasure of screening "The Big Bad S...
17453	0	This was one of the worst movies I've ever see...
10795	1	I have been a fan of Pushing Daisies since the...
18593	0	No ,I'm not kidding. If they ever propose a mo...
22429	0	I'm only rating this film as a 3 out of pity b...
20916	0	Is this the same Kim Ki Duk who directed the p...
5919	1	After not having much luck at selling his scre...
8034	1	I had never heard of this film prior to seeing...

SOFTWARES/ LANGUAGES / LIBRARIES USED:

Python 2.7, NLTK, Pandas, sklearn, numpy

VECTORIZATION:

I have converted the text in tokens and each of the token is associated with the weight based on the TF IDF technique.

Also, removing the english stop words using the **nltk.corpus** library.

```
stopset = set(stopwords.words('english'))
vectorizer = TfidfVectorizer(use_idf = True, lowercase = True, strip_accents= 'ascii', stop_words = stopset)
```

BUILDING THE CLASSIFIER:

- Setting the dependent (polarity) and the independent (review) variables.
- Splitting the data from testing and training purpose in a ratio of 8:2::train:test.

```
y = df.polarity  # dependent variable
X = vectorizer.fit_transform(df.review) # independent variable
X_train, X_test,y_train, y_test = train_test_split(X, y,test_size=0.80)
```

- Training the naive bayes classifier

```
NB = naive_bayes.MultinomialNB()
NB.fit(X_train,y_train)
```

ACCURACY:

```
In [95]: roc_auc_score(y_test,NB.predict_proba(X_test)[:,1])
Out[95]: 0.93024295388726241
```

The classifier got a 93% of accuracy.

CONCLUSIONS:

To check the classifier I manually passed a list of sentences.

```
In [99]: for word in [["I liked the movie"],["I hate this character"],["The plot of the movie was interesting"],["It has been a
             review_word = np.array(word)
             review_vector = vectorizer.transform(review_word)
             print(str(word[0]) + " " + str(NB.predict(review_vector)[0]))

         I liked the movie 1
         I hate this character 0
         The plot of the movie was interesting 0
         It has been and amazing depiction of charaters 1
         Movie was boring 0
```

List of words based on the sentiment

```
In [100]:
words_polarity = []

for word in vectorizer.vocabulary_:
    index =  vectorizer.vocabulary_[word]
    words_polarity.append((word, NB.coef_[0][index]))

words_polarity = sorted(words_polarity, key=lambda x: x[1])


print("10 Negative words :" + ", ".join([str(x[0]) for x in words_polarity[:10]]))

print("10 Positive words :" + ", ".join([str(x[0]) for x in words_polarity[-10:]]))
```

```
10 Negative words :fawn, nunnery, vani, spiders, trawling, localized, disobeying, yougoslavia, canet, acurately
10 Positive words :time, well, see, story, great, like, good, one, film, movie
```

The classifier looks to be working perfectly with a 93% accuracy but I found the out the classifier was slightly overfitting. The neutral statements/words (**completely a subjective view**) were more tending to be tagged as negative sentiment.

Since the relevance of a term is being calculated using TF IDF technique; the accuracy depends on the size and sparsity of the data.

FUTURE WORK:

- Using machine learning techniques like Neural networks.
- Sentiment analysis using svm.

.

TIME SPENT:

I have spent more time on data collection and modification that is about 1 hour.
Documentation, presenting and programing took about 3 hours.

GITHUB:

All the files related to this assignment is accessible at:
https://github.com/akshayreddy/Sentiment_Analysis_Naive_Bayes.git