Movie Review Classification Report

Name	Adarsh Narasimha Murthy
SJSU ID	014952275

Data Preprocessing

1. **Remove HTML Tags**
   The movie review dataset contains lot of *
* tags. Since these tags will not be useful to the classifier, they have been removed using regular expressions.

2. **Remove Punctuation marks, special characters, and numbers**
   Remove all punctuation marks, special characters, and numbers from text since do not contribute to movie sentiment. The punctuation list from string library is used here

3. **Stop words**
   Stop words such as 'and', 'an', 'would'… are very common in any text, and do not help the classification task. Hence, we will remove stopwords to save computational effort. I have used nltk.corpus and sklearn.feature_extraction.text.ENGLISH_STOP_WORDS dictionary to remove the stopwords from the text corpus.

4. **Remove words that do not contribute to sentiment**
   Some words such as *actor, actress, age, movie, film, Hollywood, character, etc.,* are common across both positive and negative reviews and do not help in the classification task. Hence these words have been added to stop words for removal.

5. **Limit repetition of characters**
   Words such as *'Greeeaaaaat', 'goooooooooood', 'awesomeeeeee'* are common in reviews. We will limit the repetition of such repeating characters to two using regular expressions, since a single letter cannot repeat more than twice in English.
   Eg: Greeeeaaaatttt will be converted to Greeaatt

6. **Lemmatization**
   Lemmatization is the process of converting a word to its base form. *(Eg: liked to like).* Stemming is another way obtaining the base word; however, stemming does not provide meaningful base forms always *(eg: caring is converted to car)*. Hence, lemmatization is performed on the text corpus using Wordnet library along with positional tags.

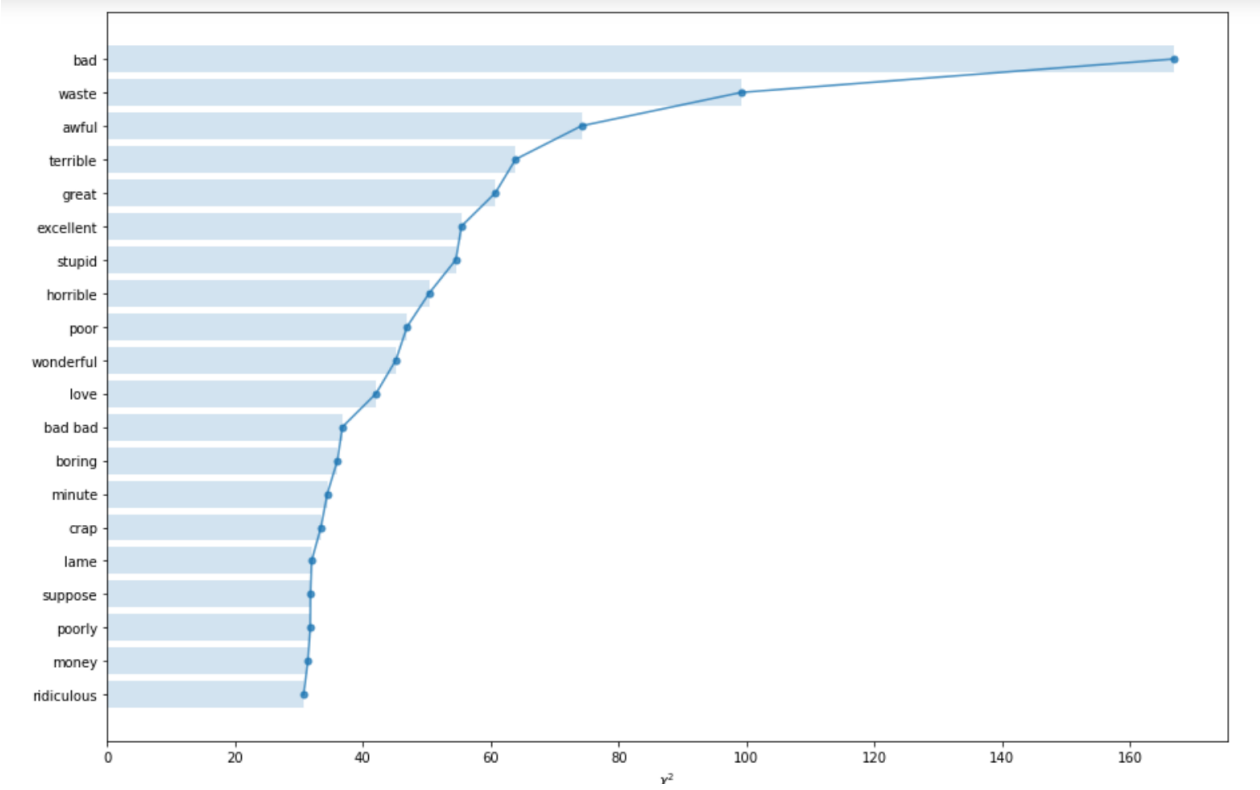
Feature extraction

I have experimented with the below feature extraction methods; we will compare the results of them at the end.

1. **Bag of Words**
   This will transform the text in our data frame into a bag of words model, which will contain a sparse matrix of integers. The number of occurrences of each word will be counted and printed. we will use a count vectorizer from the Scikit-learn library for this.

2. **TF- IDF**
   TF-IDF (term frequency-inverse document frequency) is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. A score is computed for each word to signify its importance in the document.

3. **Chi Square**
   The chi-squared statistic measures the lack of independence between a feature (in this case, one term within a review) and class (whether the reviews are positive or negative). If a feature has high chi-squared score compared to other features, it means that the feature is useful to predict the class. For example, below are the top useful features in the given movie dataset:



Dimensionality Reduction (SVD)

We used SVD on TF-IDF vector to reduce the dimensions to 3000 from 10000. However, this did not yield the best accuracy.

Classification

Knn Algorithm

The train.dat file was split into 80% training data and 20% testing for gauging the metrics of the result. *KNeighborsClassifier* algorithm from *sklearn* was used. After trail and error, the value chosen for **k** was **1900.**

Distance Metrics

We tested the results based on two distance metrics:

- Euclidean
- Cosine

Cosine yielded the best accuracy among the two. We also tested with Manhattan; however, its accuracy was very low compared to other two.

Comparison of accuracy(left) and F1-score(right):

	euclidean	cosine
Count Vectorizor	63.460000	76.870000
TF-IDF	84.940000	84.940000
TF-IDF + CHI2	71.130000	85.570000
TF-IDF+SVD	72.730000	83.970000

	euclidean	cosine
Count Vectorizor	75.300000	77.680000
TF-IDF	84.990000	84.990000
TF-IDF + CHI2	80.720000	85.590000
TF-IDF+SVD	79.860000	84.040000

Below are the observations from the above results:

1. Bag of words gives the worst accuracy
2. **Chi square along with TF-IDF gives the best accuracy, precision, and recall of 86%**
3. The difference in accuracy between chi-square + TF-IDF and only TF-IDF is marginal. There is only an improvement of 1- 1.5% accuracy.
4. The accuracy of SVD is lower than that of TF-IDF and chi-square. However, the difference is only between 1-2%.
5. Cosine distance generally performs better compared to Euclidean distance. But in case of TF-IDF there is no difference between the two.

Since, Chi square along with TF-IDF gave the best accuracy, we will be using the same for other classifiers.

Logistic Regression:

Logistic Regression uses a logistic function to map the input variables to categorical response/dependent variables. In contrast to Linear Regression, Logistic Regression outputs a probability between 0 and 1. In essence, Logistic Regression estimates the probability of a binary outcome, rather than predicting the outcome itself.

Below is the classification report with Logistic Regression:

	precision	recall	f1-score	support
+1	0.91	0.86	0.88	2815
-1	0.85	0.91	0.88	2517
accuracy			0.88	5332
macro avg	0.88	0.88	0.88	5332
weighted avg	0.88	0.88	0.88	5332

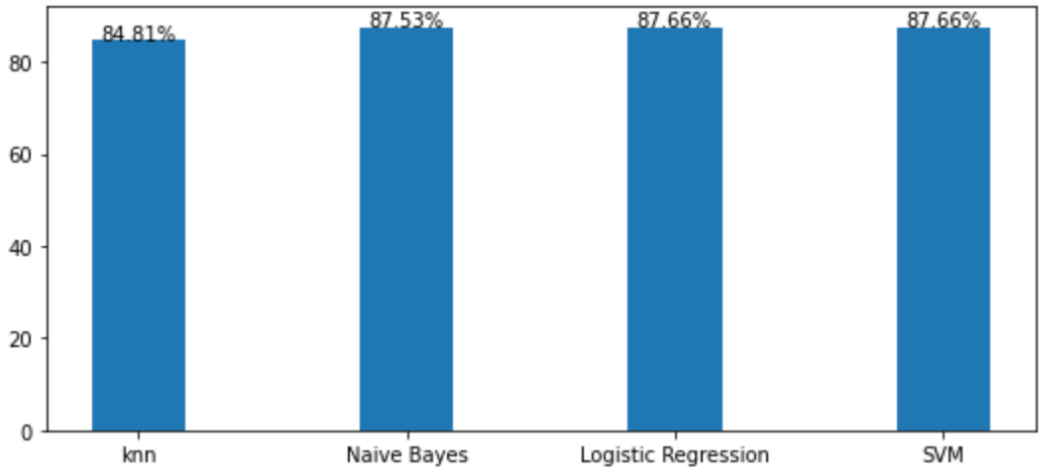
Support Vector Machine (SVM)

SVM works well for text classification due to its advantages such as its potential to handle large features. Another advantage is SVM is robust when there is a sparse set of examples and because most of the problem are linearly separable.

Below is the classification report with SVM:

	precision	recall	f1-score	support
+1	0.90	0.86	0.88	2784
-1	0.85	0.90	0.87	2548
accuracy			0.88	5332
macro avg	0.88	0.88	0.88	5332
weighted avg	0.88	0.88	0.88	5332

Conclusion:



- Knn had the worst accuracy compared to other models
- Support Vector Machine and Logistic Regression provided the best accuracy, however SVM is painfully slow and takes almost an hour to run on the complete dataset with 5000 features selected for training.
- Logistic Regression can be concluded as the best model among four. The model also runs very fast computing within 1-2 mins.

- It was observed that the accuracy of Logistic regression increased marginally between 0.2-0.6% with increase in number of features selected for training.
- Since it is not practical to use SVM, let's use logistic regression to predict the test data set.

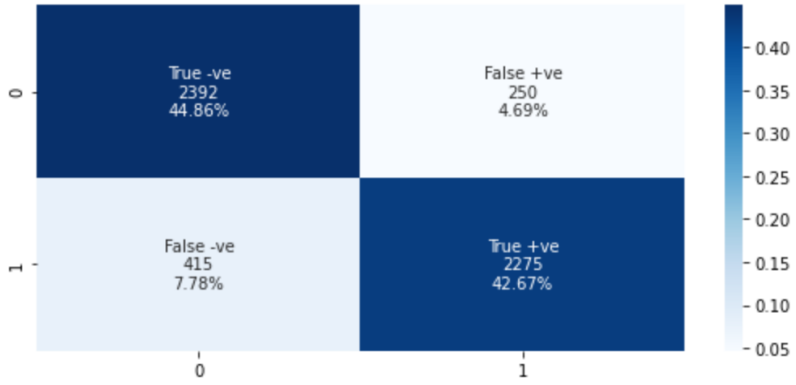


Figure 1:Confusion matrix for training data with logistic regression

The results of CLP leaderboard with **Logistic regression and TF-ID with Chi-square for feature selection** are as below:

Features selected with TF-ID: 40000

Features selected with chi-square: 30000

CLP leaderboard

Accuracy	**88.46%**
Ranking	**10**

Note: Because it will take time to compute for all models, only the logistic regression code has been uploaded on CLP dashboard, the complete code with comparison of different models is available at this github location. (the repo will be made public after due date lapse)