Data Science - CS-481

Project Title: Sentiment Analysis Using Naive Bayes Classifier

Ali Hussam (K16-3903)

May 4, 2019

Section: Gr2

National University of Computer and Emerging Sciences, Karachi Campus

1 Project Introduction

Micro-blogging is a popular communication tool these days with millions of popular websites like Twitter, Facebook and Tumblr. People write millions of messages on these communication platforms that express their views about services and products or political and religious views. These facts make the micro-blogging services the best source to gather people's opinion and sentiments which can be used by companies in marketing to conduct social studies.

2 Research Goal

The goal of this project is to develop a classifier for sentiment analysis using reviews from different movies, restaurants and twitter posts. This project will use Nave Bayes classifier to train model that can be used by companies to conduct social studies.

3 Data Collection

Dataset used in this project was created for the Paper 'From Group to Individual Labels using Deep Features'. This data set is extracted from reviews of products, movies and restaurants[1]. The size of data largely relies on the size that can be refined manually. The dataset contains data from three different websites (imdb.com, amazon.com, yelp.com). The goal of the researchers [1] while selecting data was to select sentences with clear positive or negative connotation.

3.1 Data Format and Details

- Dataset contains sentences labelled with positive and negative sentiment.
- 1 represents positive sentiment and 0 represents negative sentiment.
- The format for data before processing was "sentence (tab) score (newline)"

4 Data Preparation

From the original dataset we got 3 different files each with its own unique data. In the beginning the data from all three files were merged into a single file and a new file was created in "Dataset/Raw_Data/final_data.txt"

The columns for the data were "Sentence" and "Sentiment". The Sentiment column only contained positive and negative labels i.e 1 and 0 respectively and had no invalid or missing values. The column "Sentence" contained text reviews. The text reviews are scanned for all possible punctuation marks and special characters so they don't help in creating false predictions. During this stage this was made sure that short forms like "don't", "I'll", "he'd" etc remains as single words. After this stage the dataset was stored as csv file at path "Dataset/Processed_Data/dataset.csv".

5 Model Training

5.1 Feature Extraction

In order to train the model, Naive Bayes algorithm based on Bayes theorem is used with the assumption of independence among predictors. For the purpose of training the classifier unigrams are used. First the dataset is read and is divided into training and testing data. The ratio of division was that 80% of data is used as training data and 20% is used as testing data.

```
negative_text, no_negative_sentences = getText(train_data, '0')
positive_text, no_positive_sentences = getText(train_data, '1')
```

After data division the positive and negative data is extracted from the training data and is stored as separate list along with the count of sentences of each class.

```
negative_words = Counter(re.split('\s+', negative_text))
positive_words = Counter(re.split('\s+', positive_text))
```

The "getText" function returns count of sentences along with a single string of all the sentences separated by spaces so this string can be used to create bag of words for each class i.e negative and positive. The segmentation is done by splitting text by space to form vocabulary with consideration of not splitting the short forms of words as mentioned in the Data preparation section.

5.2 Classifier

The prior probability (P(c)) of negative and positive class is calculated so that this can be used in classification.

Now we can use all of this information to train our model. Adopting our assumption of independence among predictors, the likelihood becomes the product of independent likelihoods for each feature. So the formula becomes:

$$P(c/X) = P(x_1/c) * P(x_2/c) * P(x_3/c) * ... * P(c) \tag{1}$$

where,

- P(/x) is the posterior probability of class (c) given predictor (x).
- P(c) is the prior probability of class.
- P(x/c) is the likelihood which is the probability of predictor given class.

To handle zero-factor in the calculation Laplacian smoothing is used.

$$P(w) = C(w) + 1/N + V \tag{2}$$

From above,

- P(w) is the probability that the next word is w
- C(w) is the count of the word w
- N is total number of words
- V is the vocabulary size (distinct words)

The negative and positive probabilities for all the sentences is calculated and the sentence is classified as the one which it has the highest probability of that is negative or positive.

5.3 Accuracy Test

After all the classifications are done completely,these predictions are then matched with the original labels in test data and the accuracy of the model is checked using formula:

$$accuracy = N(correct - classifications)/N(all - classifications) \tag{3}$$

5.4 Result

The accuracy achieved by model for random division of dataset into testing and training data is given below

Data Division Ratio	With Laplace Smoothing	Without Laplace Smoothing
80:20	0.841666 84.16%	0.616666 61.66%
80:20	0.828333 82.83%	0.611666 61.16%
80:20	0.788333 78.83%	0.623333 62.33%
80:20	0.808333 80.83%	0.593333 59.33%

Table 1: Accuracy achieved for random data division into 80:20

The model achieved the accuracy of approximately 81.66%. When the model was trained and run without laplacian smoothing the accuracy of the model dropped by approximately 20.5% as it can be seen in the table 1.

6 Conclusion

Micro blogging has become one of the major types of communication. The large amount of data from micro blogging websites is a great source of opinion mining and sentiment analysis. The project focused on developing a classifier for sentiment analysis using Naive Bayes that uses unigrams.

Naive Bayes gives better results for problems with small amount of training data as is the case with our project. It is quick for giving predictions for new data as we only need to calculate posterior probability for new data point for each class and the label would be the class with maximum posterior. We also found the effect of Laplacian smoothing on Naive Bayes algorithm and how much it effects the accuracy of Naive Bayes Classifier.

References

[1] Kotzias et. al. *'From Group to Individual Labels using Deep Features'*. KDD 2015.