Name: Ekiyor T.S. Odoko Student ID: 67742019

Course: Natural Language Processing

Topic: Lab Project 4 Report

Due: 12-November-2018

1. **Aims & Objectives**

In this document, I am to report the behavior of two classifiers, Logistic Regression and Naïve Bayes for binary text classification. Each classifier will be tested with two separate forms of the data, normalized and unnormalized form. In documenting their behavior, I will compare performance of both classifiers and attempt to explain the resulting values.

1. **Approach**

The tests will be done in a supervised setting where training datasets will include text documents and their labels.

* 1. **Dataset**

I’ll be using a dataset from ‘*From Group to Individual Labels using Deep Features'* a paper by Kotzias et. al,. KDD 2015. This dataset is extracted from reviews of products, movies, and restaurant. Further information can be found in dataset/readme.txt. Find a sample of dataset below:

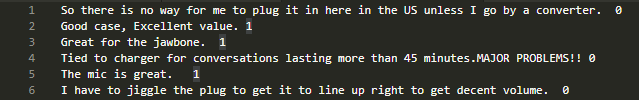


Figure 1: Example of extracted dataset

The review is accompanied with a binary label. 1 meaning the review is positive and 0 meaning the review is negative. The text and corresponding label are in the format [text] [tab(‘\t’)] [label].

Number of observations is 3000

* 1. **External Libraries**

Pandas – An open source easy to use python library for data structuring and management. This library would be used to store the data in a desired format

Scikit-learn – A widely used open source python library for machine learning, data mining and data analysis. This library provides me with an implementation of both classifiers that would be used in the project.

Numpy – An open source python package that performs sophisticated numerical operations on arrays.

* 1. **Feature engineering**

I use the pandas library to transform data from ordinary text to pandas dataframe format as shown in *fig. 2.*

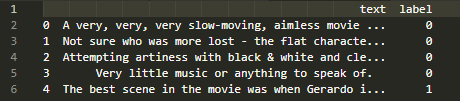


Figure 2: Dataset restructured to a pandas dataframe

* + 1. **Tokenization**

Now, I convert the ordinary text (review) into a matrix of count tokens that will be fed into the classifiers using scikit-learn’s CountVectorizer(). This converts each review to a list of features where a feature is characeterized by the count of the occurrence of a word. The result is a count matrix of observations (text reviews).

* + 1. **Split dataset**

I use sklearn’s train\_test\_split() function to split the dataset into training and testing with the 80:20 ratio.

* 1. **Train and Predict**
     1. **Classifier Selection**

I choose the classifier I want to use either Logistic Regression, or Naïve Bayes. For the naïve bayes’ classifier, I’ve decided to use the Gaussian Naïve Bayes’ as opposed to the Multinomial or Bernoulli version because it’s more basic than the rest.

* + 1. **Train**

After classifier has been selected, I train the classifier using the scikit-learn’s classifier.fit() function. This fits the classifier according to the independent variables of the training data and the labels.

* + 1. **Predict**

I use sklearn to predict the appropriate labels for the test dataset that was separated earlier. This returns an array of the labels for each test observation.

* 1. **Normalization process**

As mentioned earlier, this document will report behavior of classifiers on both unnormalized and normalized datasets. To run test on normalized data, I’ll carry out a few processes before the data is tokenized.

1. Case folding – Convert all characters in the dataset to lower case
2. Punctuation removal – Remove all punctuations as they might affect count for word occurrence.
3. **Results**

The results of the test is carried out using the accuracy metric. This is a good metric to use in this scenario because the there is an “almost even” representation of both positive and negative reviews (no underrepresentation or rarity of data in a category).

Accuracy was calculated by using numpy to find the mean after summing up the count of all correct prediction.

The results of the project are show in the table below:

|  |  |  |
| --- | --- | --- |
|  | Naïve Bayes’ (Gaussian) | Logistic Regression |
| Unnormalized | 66.55% | 81.09% |
| Normalized | 68.55% | 81.46% |

1. **Insights**

The results of the test are surprising. My expectations were that the normalization process will make a huge difference in the prediction accuracy, but as per the test it is not so. Normalizing the dataset only improved the naïve bayes’ classifier by 2% and less than half a percentage for the logistic regression classifier. This shows that normalization does not have significant effect on the classifiers. Or, this can be due to the size of the dataset. Due to its relatively small size, it is possible that the true potential of normalization was not realized.

Also, from this test we can see that the logistic regression classifier performs better than that of naïve Bayes’ on every front.