PS1 report

Colin McDonald, Mrinal Chaudhari

October 1, 2021

1 Summary

In this problem, we implemented and trained the parameter of a bag-of-words uni-gram NB classifier using pseudo-code described in “Jurafsky and Martin fig 4.2”. This pseudo-code has been described in a simple way that can be implemented in python easily. The algorithm is a multi naive Bayes classifier in which the document out of all classes returns the maximum posterior probability. This can be written as follows,
c= argmax P$(c|d)$ where c ϵ C
We have followed the steps including calculating log prior and log-likelihood probabilities.Procedure to train naive Bayes is simple. We have estimated the maximum likelihood probability and added the alpha-smoothing in the denominator of the equation. The problem with log-likelihood probability is if we count the number of words for “Good” given class “Positive”, but suppose there are no training documents that both contain the word “Good” and are classified as positive, then the probability becomes zero. To avoid this problem we added the alpha smoothing parameter in the denominator.

2 Procedure

We did not make use of any sklearn library for the Naive Bayes algorithm. We have implemented this problem in python using the pseudo-code mentioned in the given book. We added the list of Stopwords using a NLTK library. The reason behind using stopwords is that it can remove the unnecessary words from the text that do not add much meaning to sentences such as: a, the, she, it, they, etc. we have done the necessary steps in our implementation including reading the file, cleaning, preprocessing of the data-set. Later, we have calculated the log prior and likelihood probabilities using the algorithm. We had to handle the features with zero occurrences and then we made the decision to add alpha smoothing parameter in log-likelihood probability. In the implementation, we observed that the addition of a stop words list does not improve the performance significantly. In most pre-processing techniques the entire vocabulary is being used instead of stopwords.

3 Results

We have done the pre-processing on training and testing dataset. In the pre-processing we have remove stopwords, punctuations, and conersion of lower case. After pre-processing we, we have followed the pseudo code and alogorithm to implement naive bayes classifier. The accuracy that we are getting is 49.9%

Accuracy: 0.49142550911039656
recall: [0. 1.]
precision score: [0. 0.49142551]

We have done implementing Naive Bayes using Sklearn library. We have imported Multinomial Naive bayes from sklearn.naive_bayes and we fit the model on traning dataset. The result we have obtained using scikit-learn implementations are improved to 77%.

Accuracy 77.93%

4 Limitations

The limitation of this implementation is that the dataset is quite ambiguous in nature and some of the lines from the dataset are skipping. The dataset is a large text file and it contains a section with a column that's causing the program to crash. Hence, we used error_bad_lines to ignore those rows. In the implementation of naive bayes algorithm, We noticed that the author has made use of for loop to iterate over the classes and rows. When we run the program, it takes 30 seconds to run the whole algorithm. The time complexity of the problem needs to be improved. The another problem we have observed that the prediction are only made on POSITIVE class and accuracy is low when we run experiments on training dataset. After running model using scikit learn library, the results are significantly imporved and accuracy is 77%.