Pattern Recognition and Machine Learning Assignment 3

Barenya Kumar Nandy, CS22M028

November 17, 2022

Abstract

Spam/Ham is an exemplary classification problem and a good way to explore popular classification methods. We go through several classification algorithms and a use a data set which is a '.csv' version of Enron1 for our purposes. We implement the basic Naive Bayes algorithm which assumes that features are conditionally independent, thus the name Naive Bayes. The Naive Bayes is simple and robust which is the reason why we implement it

1 Data set used

Here, we use three different data sets, the lingspam dataset availiable at the link given in the last section of the report.

We also use the Enron and the CompleteSpamAssassin data set which were all downloaded in csv format from Kaggle.

1.1 Data set description and importing

A function imp-data() is used in order to import the data from the csv files. Here for training we use the file named "enronSpamSubset.csv" This csv has 10000 emails examples which are all available in the Enron data set, along with their labels. The 0 in the labels denote a non spam email and a 1 in the labels denotes a spam email.

1.2 Data processing

For cleaning up, we have to remove punctuation. The existence of punctuation would cause an anomaly such as "money,", "money", "money.", "money!" will all be treated differently.

They are to be removed, and here the string library is being used.

We also have to consider only the top few thousand most frequently occuring words. On these words, we do the further processing the lower frequency words will be of almost no consequence and will add no value to whether the email is spam or not.

Now, stop words will also have to be removed. We use the stop word database from nltk for English, we must remove these to avoid the accuracy from faltering too much.

1.3 making the dictionary

We make a dictionary where each word is mapped to a unique index. Then we use these indexes to refer to the words when using the one hot encoding where the presence of a word is indicated by a 1 and its absence by a 0.

1.4 representing the emails as one hot encoded

Every mail is represented by the words contained in them, irrelevant of the frequency or the position of the word in the email. These one hot representations of the emails are fed into the Naive Bayes and are used for updating the parameters during the training phase

1.5 dealing with the testing

For testing purposes, any text file inside the test folder will be treated as a viable email for testing. The test-on-folder function will load up the parameters from the files, and browse through the test folder.

We will run the classifier and check the value of the decision boundary for each text file and output 0 if it is ham and 1 if not spam.

2 Using Naive Bayes Classifier

2.1 Data Processing:

Firstly, we do not want there to be any tokens in our dictionary(which is necessary for the construction of our classifier) that can cause our accuracy to decrease. This includes words such as 'up', 'down', etc. We use the NLTK stop-words data set to remove these from our training data. Unfortunately, for these we have to compare each word in our data to all the stop-words in the data set to check their presence or absence which turns out to be computationally expensive.

We do not want punctuation marks, abbreviations and special characters which are not a part of what the e-mail tries to convey. Having these will only decrease our accuracy.

We also just want the most frequently used characters only. Otherwise, we will end up with a huge dictionary which increases the time complexity enormously.

2.2 Implementation

The Naive Bayes' is a generative model and therefore can give good accuracy even with a small data set. The processed data set needs to be scanned through mail by mail and corresponding parameters(depending on whether we it is spam or ham) will have to be updated.

There are (2d+1) parameters in total, where d is the length of the dictionary.

We have d parameters for the case of ham emails We have d parameters for the case of spam emails

We have 1 parameter that decides whether the mail is a spam or a ham. This is our generative model and we will use these parameters to classify further mails as spam or not spam.

For the Naive Bayes', we use the multidimensional Bernoulli distribution.

$$p_{0_0}, p_{0_1}, \ldots, p_{0_n}$$

$$p_{1_0}, p_{1_1}, \ldots, p_{1_n}$$

where p_{ij} is the parameter for jth feature having label i

We update the parameters using the following:

$$p_i^y = \frac{\sum_{i=1}^{n} 1(f_j = 1, y_i = y)}{\sum_{i=1}^{n} 1(y_i = y)}$$

and, for the parameter p, we use the following:

$$p = \frac{1}{n} \sum_{i=1}^{n} y_i$$

Here y indicates the label of the particular mail in consideration.

2.3 Prediction:

We simply use the multivariate Bernoulli to predict whether the e-mail is spam or non spam.

Which ever gives a greater value for the probability density function gets out of p_0 and p_1, we take that as the label.

We use the following logic for our purposes:

prediction = 1 if $P(y_test = 1/x_test) > P(y_test = 0/x_test)$

prediction = 0 otherwise

Therefore we compare as follows:

$$P(y_test = 1/x_test) \propto P(x/y_test = 1)P(y_test = 1) \ldots equation_1$$

we replace $P(x/y_test = 1)$ with $\prod_{k=1}^{n}(p_k^1)^{f_k}(1-p_k^1)^{1-f_k}$
and for the cases with label = 0

$$P(y_test = 0/x_test) \propto P(x/y_test = 0)P(y_test = 0) \ldots equation_2$$

we replace $P(x/y_test = 0)$ with $\prod_{k=1}^{n}(p_k^0)^{f_k}(1-p_k^0)^{1-f_k}$
Now, we compare equation 1 and equation 2 such that
if equation 1 ¿ equation 2 we classify as spam else, we classify as ham

2.4 Problems encountered

When a word is not encountered during training phase, we may get a parameter value that is 0. This will cause the entire value to become 0, this bringing in anomalous behaviour for certain messages. To avoid this, we use Laplacian smoothing. We simply add a message which has all the words and is a spam and a message which has all the words and is a ham.

When the message is sparse, there may be too many ¡1 values for (1-p_j^y) values. These when multiplied too many times results in too little of values for equation 1 and equation 2 to compare and this leads to anomalous output

This is why we use the decision boundary to make the classification decision:

$$\sum \log \frac{p_i^1(1-p_i^0}{p_i^0(1-p_i^1} + \sum \log \frac{1-p_i^1}{1-p_i^0} + \log \frac{p}{1-p}$$

If the above returns a value greater than 0, we have a spam and if it returns a value lesser than 0, it is a ham.

2.5 Testing accuracy over a different data set

Here we use the ling spam data set provided at the following website for the purpose of obtaining a testing accuracy. "http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearningdoc=exercis

It is an open classroom data set and is therefore available to all. This data set has a lot of hard-ham emails which are hard for the classifier to classify as spam.

We also test on the completeSpamAssassin data set which has very hard hams and is a bench mark for testing spam classifiers.

When we train using the CompleteSpamAssassin data set, we have the best accuracy for all the possible testing data sets.

The code can be used to test mails in txt format from the test folder, when kept along with the same python file.

2.6 important for convenient testing

The Naive Bayes classifier here is being trained over a data set of over 10000 emails from the Enron data set. This would take quite some time. For the sake of convenience, the numpy arrays for the values of the parameters have been stored in the same folder as that where the python file is.

The function with loading these values ensures the possibility for commenting out the training phase entirely.