Abdul Rehman Liaqat

Lab Course Machine Learning

Exercise Sheet 10

Report

Exercise 1: A spam filter using SVM

Part A: Build a spam filter using pre-processed dataset:

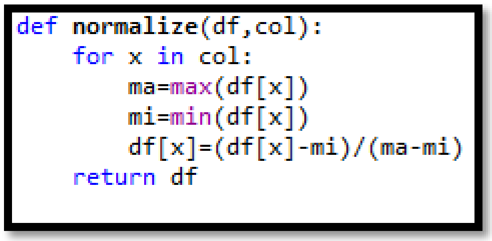
To implement a spam filter LibSVM library was used. There are two versions of LibSVM are available online. One through GitHub which is usually said to be the original version requires a change in pre-processed data before it can be fed to train the algorithm. On the other hand, second available version is provided by Scikit-learn. Scikit-learn.svm.libsvm provides hundred percent same functionality (low level C) as by original LibSVM the only difference is the data pre-processing. Later version doesn't require further processing of pre-processed dataset. Scikit-learn is providing a wrapper of LibSVM hence no further change in pre-processed data is needed.

After strong analysis and thinking Scikit-learn version of LibSVM was used because:-

1- It was providing more hands-on function which can be plus point when doing Grid Search.
2- Problem loading LibSVM installed through github.

Proceeding with the working of building a spam filter following steps were taken:

1- Normalized all features values to obtain values between [0,1]. Normalization had a humongous effect on execution time. Reducing execution time many folds. For this purpose, following function was used:



```
def normalize(df,col):
    for x in col:
        ma=max(df[x])
        mi=min(df[x])
        df[x]=(df[x]-mi)/(ma-mi)
    return df
```

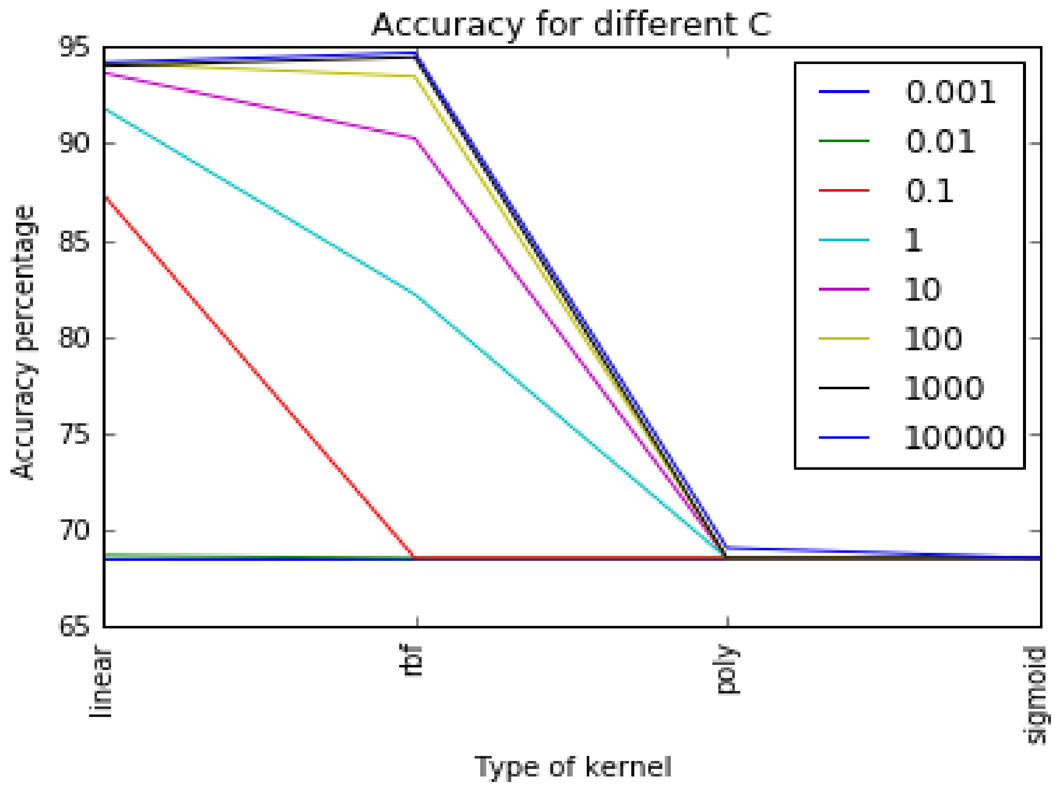
2- Separated 20% test data from randomly shuffled data.
3- Separated features and variables of both test and training data.
4- Defined a range of hypermeters. Here two hypermeters were used. One the value of constant C which is actually the value of error penalty and very important to optimize. A large range from 0.001 to 10000 was selected. Second hyperparameter, although not a hyperparameter actually but a method of comparison was kernel type. Four kernel types were tested for each value of C.

Four kernels used are

- Linear
- Polynomial
- Radial basis function
- Sigmoid

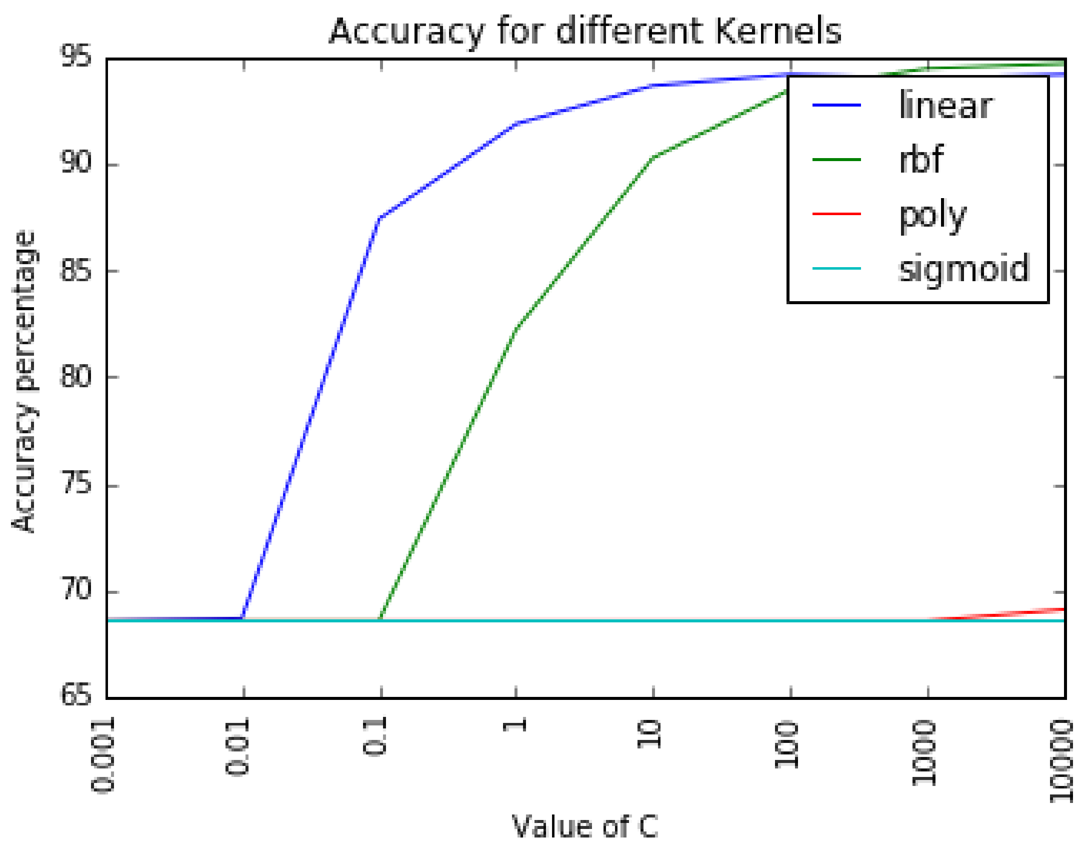
Although each kernel type has its own parameters which could be used as hyperparameters but to keep the comparison fair only default values of such parameters were used.

5- 5 fold cross validation was used.
6- Accuracy was used as a classification quality criterion.
7- After grid search and cross validation, we obtained following results:



As seen through the plot it is obvious that RBF is outperforming other kernel types for almost every value of C.

Abdul Rehman Liaqat



Plot for different values of C provide us a very interesting picture. When C is too small it provides us a very low accuracy percentage. With the increase in C it increases exponentially. For mid-range and most of the values of C linear kernel is comparable to RBF. It is also evident that at the upper end of C there is not much significant improve in accuracy percentage hence it can be deduced that further increase in C won't do much improving in prediction.

And finally training the data on hyperparameters obtained through Grid Search provide us following results:

```
final accuracy is  0.942391304348
best value of C is =  10000
```

And the best type of kernel is RBF.

a

Abdul Rehman Liaqat

Part B: Pre-Processed a dataset and learn SVM:

To convert tagged data of spams into useful input pre-processing is needed. Given data consists of two columns. One is the label about SMS if it is a spam or a ham and the other contains the SMS itself.

To pre-process the data a python file with the name Preprocess.py was created in which following steps were taken to pre-process the data and at the end a csv file was written into the system.

(All these are attached alongside)

1- Assembled all the messages of spam SMS and applied tfidf (term frequency-inverse document frequency) method to it using Scikit Learn feature extraction utility.
2- After tfidf a sorted dictionary of words and their respective inverse document frequency was obtained. This would tell us the frequency of words in spam SMS in descending order.
3- After filtering out numeric values (phone numbers, numbers) top 100 words were selected and a vocabulary was created.
4- Later on, for each SMS tfidf against vocabulary. Hence, we were able to obtain a data frame with one row representing one SMS and columns representing each word of vocabulary and its tfidf in the respective SMS.
5- Also, one last column before labels depicts the number of words in every message. This column serves two purposes. One to verify the hypothesis that all spam SMS has a length of two lines to five lines. Secondly to make sure that at least one column of an SMS is filled in every case.

This data frame was written into an external csv file to save the execution time and memory.

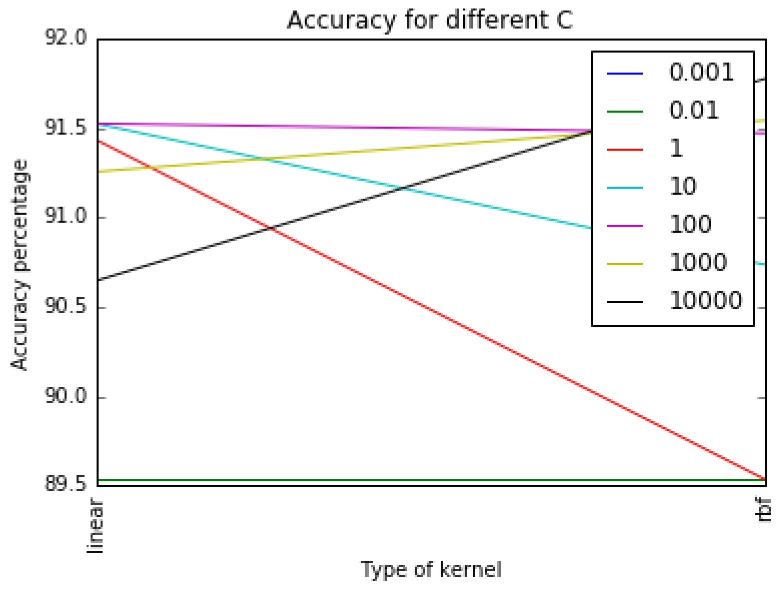
Later on, that csv file was used to train and test for spam filter.

Moving on towards the creation of spam filter. It was pretty much similar to the previous part.

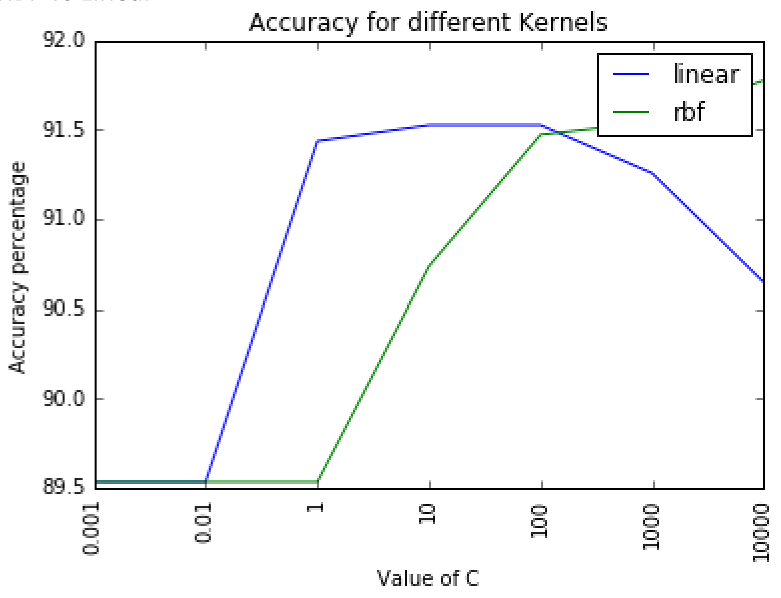
Following steps were taken:

1- Shuffled the whole dataset
2- Normalized the whole feature set.
3- 20% separation of test data.
4- Separation of labels and features.
5- Range setting of constant c from 0.001 to 10000.
6- This time two kernel types were used: Linear and Radial Basis Function (RBF)
7- Later on, fivefold cross validation with grid search was implemented to obtain following results:

   For different values of C:



RBF vs Linear



Accuracy of more than 90% is obtained through RBF.

Exercise 2: Compare SVM based spam filter with another model

To compare the performance SVM in the second part with another model simple logistic regression was chosen.

Logistic regression was implemented using Scikit learn. Regularization term lambda and starting point for Stochastic Gradient Descent were selected as hyperparameters. Their ranges were from {0.001,0.01 to 10000} and {0.001,0.01,0.1,1 and 10} respectively. Best performance was obtained on lambda 0.001 and alpha (starting point) 0.01.

After searching the grid and using parameters obtained to train the algorithm accuracy of 87.07% was obtained. Which is itself not so bad.

Overall It is obvious that SVM method is way superior than simple logistic regression.