Homework #2

CS 7930, Spring 2016

Task 1:

Think of **at least 10 features** which are important features to distinguish between spammers and legitimate users, and extract the features from the training and test sets. Write definition of the features and justification why you think these features will be useful to detect spammers. These features should be used in the following task 2.

- Following are the list of important features:

1) **Large number URLs in tweets**: These are the count of URLs contained in the tweets of a user. If the number of tweets of a user containing URLs are more than a predefined threshold, then user is considered a spammer.
2) **Length of screen name**: This is the word count of the full name. Spammer tend to have very common and short names without middle name or prefix, e.g. John Adams, Jenna Williams.
3) **Profile/Bio Description Length**: This is the word count of the bio description of the twitter user. Amount of text in spammers twitter profile/bio is quite less as compared to real users. Spammers tend to be too lazy to fill up the bio.
4) **Ratio of following- followers**: This is the ratio of number of following people a user is following to the number of users following the original user. Spam user follow large of users but not many other users follow them. So the following-follower ratio would be high for spammers.
5) **Mention of many other users in tweets**: This is the number of @ symbol present in user's tweets. Spammers tend to mention multiple users in their tweets using @.
6) **Multiple duplicate tweets (Similarity among user's tweets)**: Spammers are paid to promote some product or some page. So they re-tweet the same text containing the above, again and again. So we can count the number of re-tweets to detect spammers.
7) **Count of Hashtags**: Spammer make the use of tending hashtags (#) to post unrelated content along with it to gain high visibility. We use the count of the number of hashtag mentions as a feature.
8) **Number of Re-tweets**: Spammers generally tend to re-tweet other users tweet more than the normal users.
9) **Tweets distribution in a day**: Spammers tend to be more active during 'normal working hours (10 AM to 6 PM), whereas genuine users use twitter after the 'normal working hours.
10) **Abnormal following (Spikes)**: Spammers are paid to follow certain accounts, and they do this in short period of time, like in few minutes, so there will be a spikes in time series graph. Whereas genuine user's time series will be flat.

Task 2:

Build/train spammer classifiers with the extracted feature values from the training set using some classification algorithms (e.g., decision tree algorithms, SVM, boosting algorithms). Then use your classifiers to classify users in the test set. Choose **at least three** classification algorithms and then compare and **report their classification accuracy, F-1 measure, false positive rate and false negative rates**. You are not supposed to use directly use GUIs of machine learning tools. Instead import a machine learning/classification library, and build and test your classifiers.

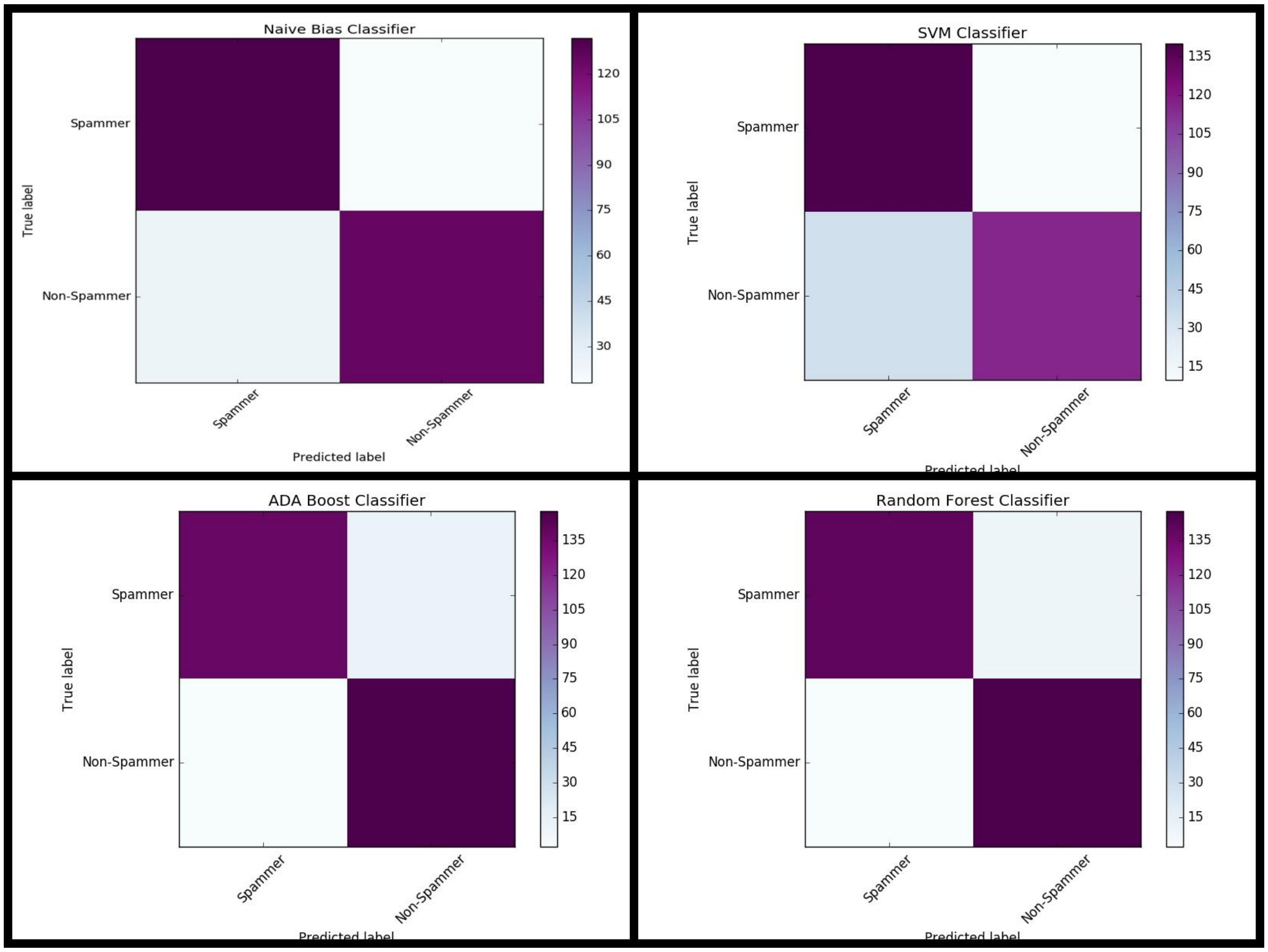
- For this task I have used first six features mentioned in the Task 1, namely:
    1) Number of URLs
    2) Length of name
    3) Bio description length
    4) Ratio of followers to following
    5) Number of mentioning of other users in tweets (@)
    6) TF-IDF similarity of a user's tweets

- I have used Python and it's libraries to code requirements mentioned in the Task 2, specifically I have carried out following stuffs:
    - Importing all the txt files, extracting and storing the user's details and tweets in a custom class called TwitterUser.
    - Extracting features from the user details.
    - Building, training and evaluating various machine learning classifiers.

- For classifying the users into spammer and legit user, I have used following four machine learning classification algorithms:
    - **Naïve Bias**
    - **Support Vector Machine**
    - **Adaptive Boosting**
    - **Random Forest**

➢ Performance comparison of Classifiers:

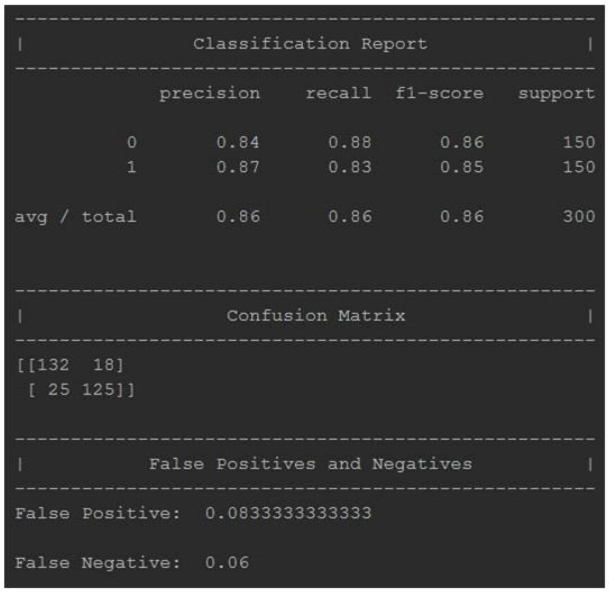
❖ Confusion Matrix:



❖ Detailed classifier comparison based on:

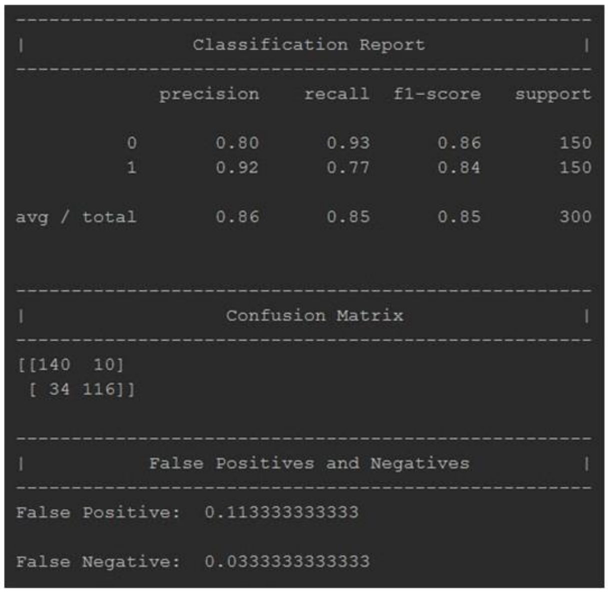
- Accuracy (avg / total)
- F-1 measure
- Recall
- False positive rate & False Negative rate
- Confusion matrix

Naive Bias

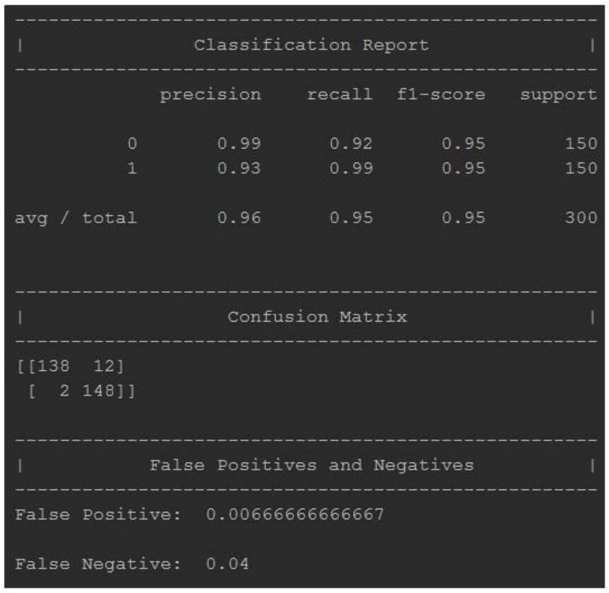


```
------------------------------------------------------------
|                 Classification Report                    |
------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.84      0.88      0.86       150
          1       0.87      0.83      0.85       150

avg / total       0.86      0.86      0.86       300


------------------------------------------------------------
|                    Confusion Matrix                      |
------------------------------------------------------------
[[132  18]
 [ 25 125]]

------------------------------------------------------------
|             False Positives and Negatives                |
------------------------------------------------------------
False Positive:  0.0833333333333

False Negative:  0.06
```

SVM

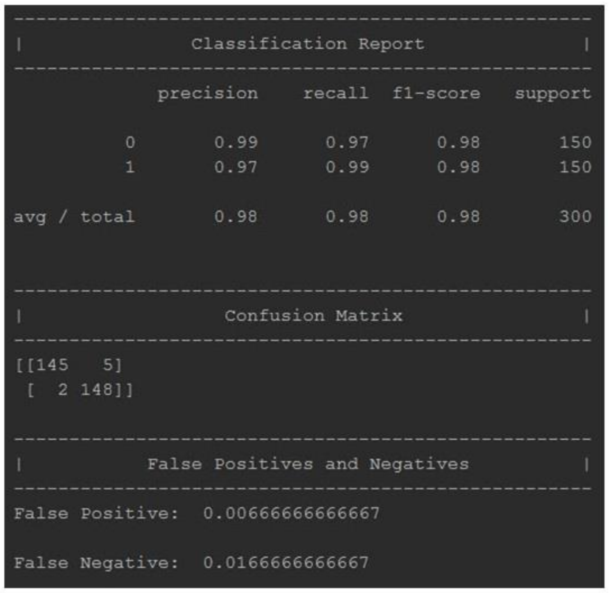


```
------------------------------------------------------------
|                 Classification Report                    |
------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.80      0.93      0.86       150
          1       0.92      0.77      0.84       150

avg / total       0.86      0.85      0.85       300


------------------------------------------------------------
|                    Confusion Matrix                      |
------------------------------------------------------------
[[140  10]
 [ 34 116]]

------------------------------------------------------------
|             False Positives and Negatives                |
------------------------------------------------------------
False Positive:  0.113333333333

False Negative:  0.0333333333333
```



```
------------------------------------------------------------
|                 Classification Report                    |
------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.99      0.92      0.95       150
          1       0.93      0.99      0.95       150

avg / total       0.96      0.95      0.95       300


------------------------------------------------------------
|                    Confusion Matrix                      |
------------------------------------------------------------
[[138  12]
 [  2 148]]

------------------------------------------------------------
|             False Positives and Negatives                |
------------------------------------------------------------
False Positive:  0.00666666666667

False Negative:  0.04
```

ADA Boost



```
------------------------------------------------------------
|                 Classification Report                    |
------------------------------------------------------------
             precision    recall  f1-score   support

          0       0.99      0.97      0.98       150
          1       0.97      0.99      0.98       150

avg / total       0.98      0.98      0.98       300


------------------------------------------------------------
|                    Confusion Matrix                      |
------------------------------------------------------------
[[145   5]
 [  2 148]]

------------------------------------------------------------
|             False Positives and Negatives                |
------------------------------------------------------------
False Positive:  0.00666666666667

False Negative:  0.0166666666667
```

Random Forest

❖ Observations:

- **Accuracy**: Random Forest gave the best accuracy of 98%, closely followed by Ada Boost at 96%. Naïve Bias and SVM performed the worst with accuracy at 86%.

- **F1 Measure**: Again Random Forest gave the best f1 score at 98%, followed by Ada Boost at 95%, while SVM performed worst at 86%.

- **False Positive**: Both Random Forest and Ada Boost produced the least amount of false positive, at 0.00667%, indicating that they didn't misclassify many non-spammers as spammers. Whereas SVM produced most number of false positives, at 0.1333%.

- **False Negatives**: Random Forest misclassified least number of spammers as non-spammers, meaning that its False Negative rate was low, at 0.01667%. Naïve Bias produced most false negatives at 0.06%.

Task 3:

Based on your experience of tasks 1 and 2, discuss how you could improve performance (i.e., accuracy) of the classifiers. Give some concrete examples or ideas about how you could improve the quality of your classifiers.

- Accuracy of the classifiers can be improved in following ways:

    1) **Adding more sophisticated features**: Prediction accuracy of the classifiers can be greatly improved by extracting more features for the dataset and adding those extracted features to train the classifier (in addition to the previously used features. For this particular twitter problem, following features can also be used to improve the classifier performance:
        - Distribution of tweets in a day.
        - Extreme deviation of 'followings' or 'likes' in a given period of time.
        - Large number of people blocking a user (Insider data).
        - Number of re-tweets, etc.

    2) **Removing noise from the data**: Training classifier with noisy data could result in high number of misclassified testing data. There could be high amount of false positive and false negatives. So the data should be cleaned of noise before applying to the classifier.

    3) **Removal of irrelevant features**: Feature selection is one of the most important steps in building a classifier. Features should be carefully selected, such that they help in improving the accuracy of the classifier, not vice versa. There could be some features which do not truly reflect the property of an item. And these features when added to classifier reduce its accuracy. For example (hypothetical): It is believed that spammers' tweets contain less amount of words than a legit user, and this could be used as a feature. But in reality the word count for both is almost the same. So this feature wouldn't be able to improve the accuracy of the classifier, in fact it could lower the accuracy.

    4) **Tweaking the classifier algorithm parameters**: Tweaking the parameters of classification algorithm can significantly improve the accuracy. For example:
        - In ADA Boost, I tried different values of max_depth and n_estimators to get the best accuracy. max_depth = 1 and n_estimators = 90 gave me the best accuracy.
        - Similarly for Random Forest, n_estimators = 150 gave the best accuracy.