Experiments and Analysis Applying Various Methods in Email Spam Filter

Zhiqian Yu

Fall 2015

1. Dataset Description

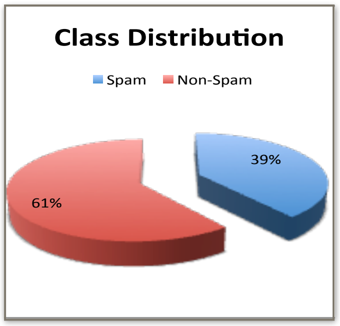
I use Spambase Data Set from UCI. This data set contains 4610 instances and 58 attributes, one of which is the target which separate the email to spam and non-spam with 1 and 0 respectively.

1.1 Attributes Information

All of the attributes are continuous real number except for the target attribute.
The first 48 attributes represent the percentage of the frequency of well-defined WORD in the email. In this case, WORD means any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string. It is defined by the specialists.
The 49th - 54th attributes represent the percentage of the frequency of well-defined CHAR in the email.
The 55th attributes represent the average length of uninterrupted sequences of capital letters.
The 56th attributes represent the length of longest uninterrupted sequence of capital letters.
The 57th attributes represent the total number of capital letters in the e-mail.
The last attribute is the target.

1.2 Missing Value

There are NO missing value in this dataset.



1.3 Class Distribution

Spam 1813 (39.4%)

Non-Spam 2788 (60.6%)

1.4 Attribute Statistics

The ranges of first 54 attributes are between 0 and 100. All of their mean are between 0 and 1.6, which means there are many values of first 54 attributes are small digits.

The values of 55th to 57th attributes are larger than 1. There is no upper bound of these attributes. Thus, the averages of these three attributes are 5.1915, 52.173 and 283.29 respectively.

Actually, in the data set, the values of 55th to 57th attributes are far more larger than the rest of attributes in this data set.

2. Objective

(1) Compare the performance of each classifier in predicting whether an email is spam or non-spam.

(2) Reducing the false positive result, which means that reduce the chance of marking a good mail as spam.

(3) Compare the result of each classifier with PCA and without PCA.

3. Classification Method

I have use 4 classification method to predict whether an email is spam or not.

(1) K-NearestNeighbors

I have implemented a general KNN classifier. It takes three parameters: Train, Test and K, which are the training data set and the test data set.

Classification is computed from a simple majority vote of the nearest neighbors of each point.

(2) Naive Bayes

I have implemented a general Naive Bayes Classifier. It takes two parameters: Train and Test, which are the training data set and the testing data set.
The NB classifier uses the posterior to separate two classes. It assumes every feature is independent given a class variable and the likelihood of feature is in Gaussian distribution.

(3) Support Vector Machine

I use build-in function in python to implement this classifier. The SVM classifier will use training data set to fit its model, and predict the class label for the testing data set. I use the class labels predicted by SVM to compute the accuracy. I have used the default kernel function for this classifier, which is

$$\exp(-\gamma|x - x'|^2)$$

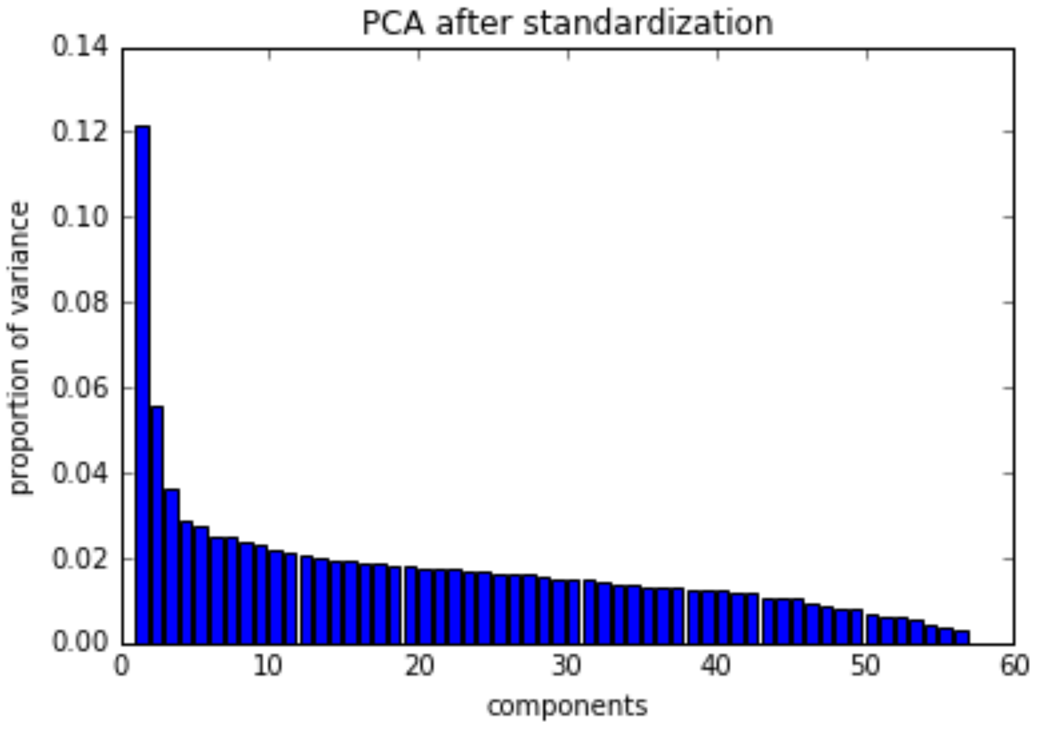
The main purpose of SVM is to find the hyperplane that maximize the margin between two classes.

(4) Random Forest

I use build-in function in python to implement this classifier. I have set the number of weak classifier in the forest to 25. I have set the maximum number of features to consider when looking for the best split to be sqrt(n), where n is the number of features in the training data set.
The main idea of Random Forest classifier is to use the results of weaker learners which randomly select features to fit the model and predict the class labels of testing data set. Then ensemble all the results of all classifier.

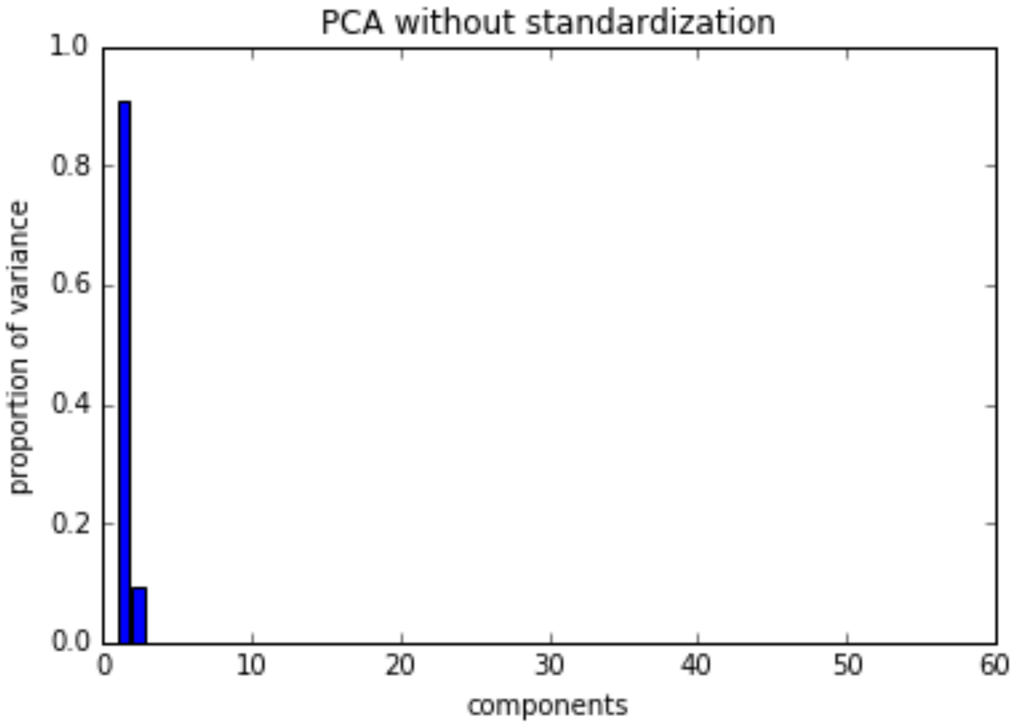
4. Data Pre-processing

There are 57 attributes in the data set, so it is reasonable to perform PCA before classification. Principal components analysis is used to reduce dimensions according to the variance of each feature.
The proportion of each feature is as follow:



There are 57 components in total, however from the above graph, we can easily find that the first component contains more than 90% information of the whole data set and from the third principal component to the last component, they contain too little information so that we can see nothing about them in the graph. Actually, the first three principal components contains more than 99% information of the data set, which means if we choose project to the new base constructed by the first three components, we will just lose less than 1% information. Conceptually, it is good enough to project the data set to the new base. However, if we use this new base to transform our data, we will get lower accuracy.

Before I explain the reason, let us look at the proportion of variance of each feature after standardization. Standardization means removing the mean and scaling to unit variance.
According to the graph, we can find that the first components just contains 12% information of the data set and even the last principal components contain 0.8% information of the data set.
After I add through the proportion of variance from the beginning to the end, I find that if we use first 43 principal components as the new base, we will get more than 91% information of the data set. The reason why I do not choose the new base that contain more than 99% information of the data set, is that after 43rd components, when you include one more components, the total proportion of variance will not change to much. In addition, even if you get to the 55th



components, the total proportion of variance is also less than 98%. Thus, I stop at the components that makes the total proportion of variance exceed 90%.

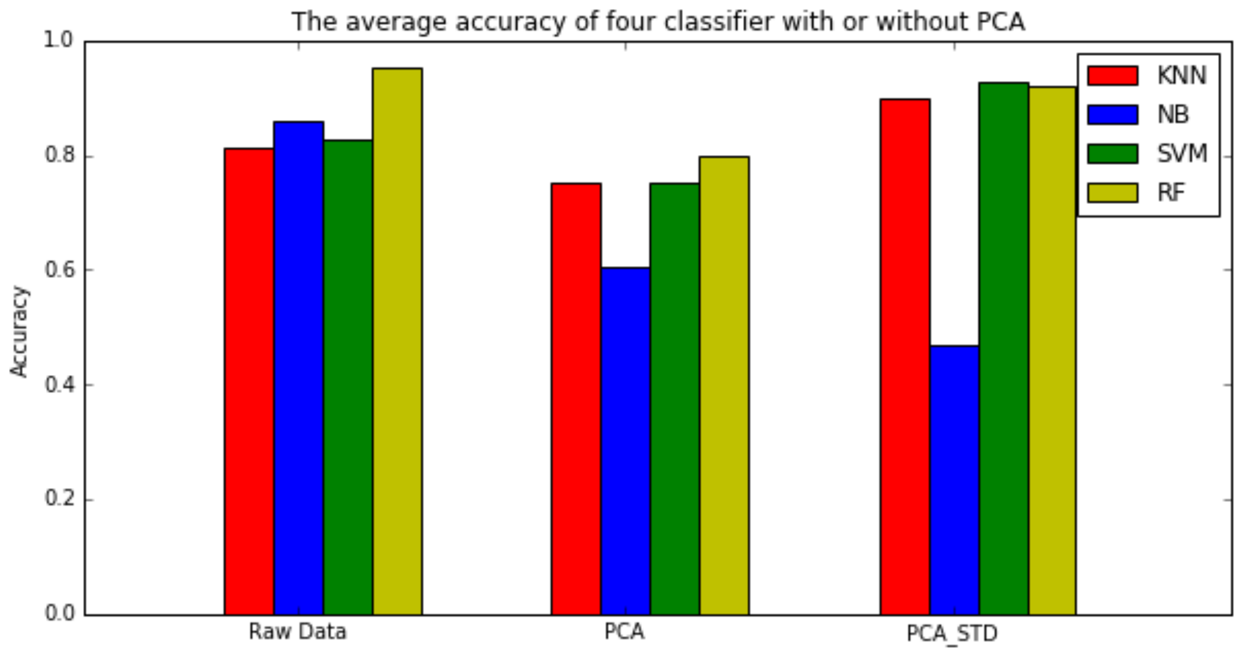
5. Results and Analysis

Randomly select 50%-70% data from original data set as my training data set. The rest of data set is the Testing data set.
Repeat the process for 10 times and calculate the average accuracy.

5.1 Accuracy

	Raw Data	PCA without STD	PCA with STD
KNN	0.811866542984	0.752931936387	0.89868496289
Naive Bayes	0.859479273703	0.606313465303	0.467521745027
SVM	0.827138404466	0.751673360785	0.927017141126
Random Forest	0.95190851144	0.798930476008	0.92186300239

The average accuracy of each classifier is as follow:
Using PCA without standardization give the lower accuracy, which is the second part of the graph. I think the reason why standardization will affect PCA is the



range of each attributes is different. As I have mentioned above, in this data set, I have 54 attributes which have mean between 0 and 1.6. The range of the those 54 attribute is 0~100. However there are three attributes whose mean are 5.1915, 52.173 and 283.29 respectively. If we do not standardize the raw data before we perform PCA, those three attributes will dominate the whole data set, especially the one with largest mean. That is why performing PCA without standardization will give us a new 3-D base.

For KNN and SVM, performing PCA really helps to improve the accuracy of the classifier. For KNN, PCA improve the accuracy by 8%. For SVM, PCA improve the accuracy by 10%.

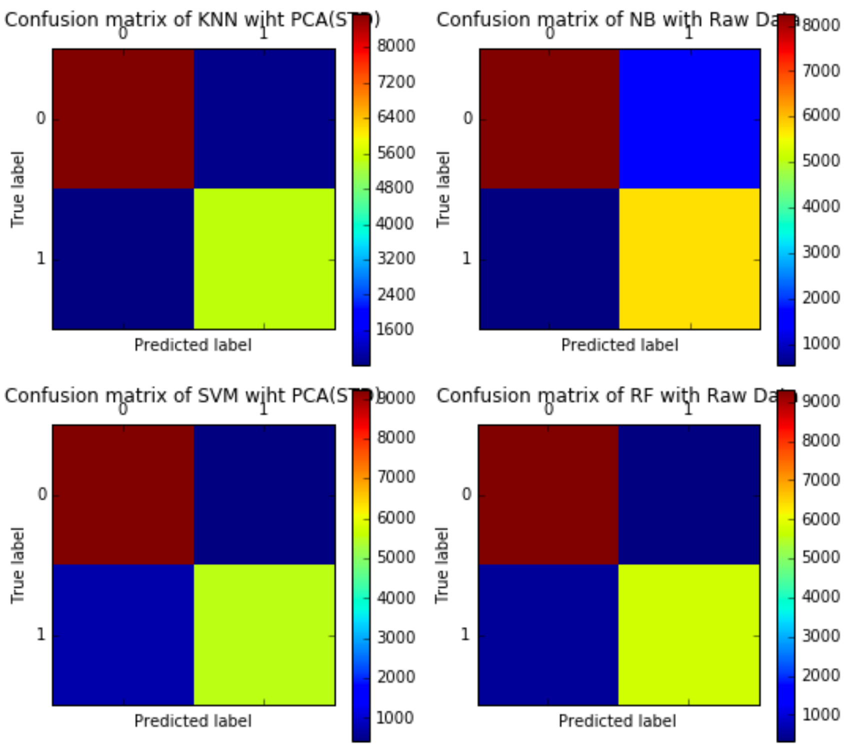
The Random Forest classifier give the highest accuracy among all classifiers. That is what I expected. However, after performing PCA, the accuracy decreased by almost 2%. Since random forest will build each decision tree with features that are randomly chosen from all 57 attributes, it is possible that it will cover all 57 attributes in this dataset when repeating for serval times. But after performing PCA with standardization, we still lose some features or information. That is why when performing PCA will reduce the accuracy of random forest classifier in this case.

The performance of Naive Bayes classifier is unstable when performing PCA. According the chart above, the Naive Bayes classifier get lower accuracy when performing PCA with standardization than the accuracy when performing PCA without standardization. Actually, the accuracy of Naive Bayes is fluctuating with the changing of the number of components chosen.

5.2 Precision and Recall

Using accuracy is not enough for determining which classifier is best. We should also check how useful the results are and how complete the results, which are precision and recall respectively.

There are four confusion matrixes of four classifiers. I have tested all three datasets,



which are raw data, data after PCA without standardization and data after PCA with standardization. I just choose the best one of each classifier.

Confusion Matrix give a direct view of true positives(left top), true negatives(right bottom), false positives(right top), and false negatives(left bottom). The true positives and true negatives are those been classified correctly. False positives mean that make good emails as spam ones in this case, and false negatives mean that make spam emails as ham ones.

According above figures, it is easy to compute the precision and recall of each classifier with formulas:

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

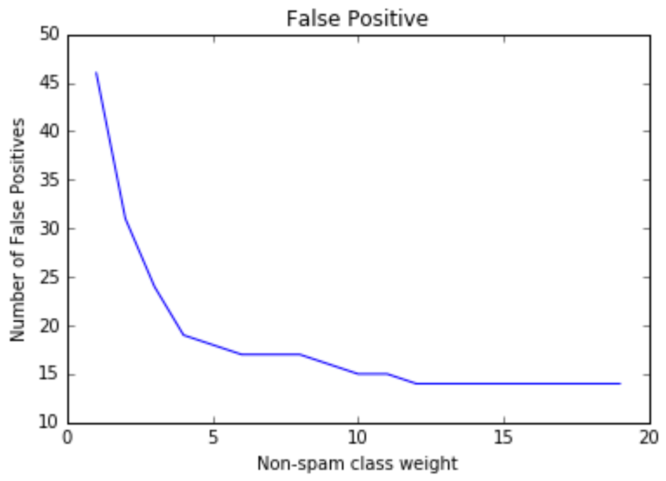
where TP stands for true positives, FP stands for false positives and FN stands for false negatives.
The precision and recall of each classifier are as follow.

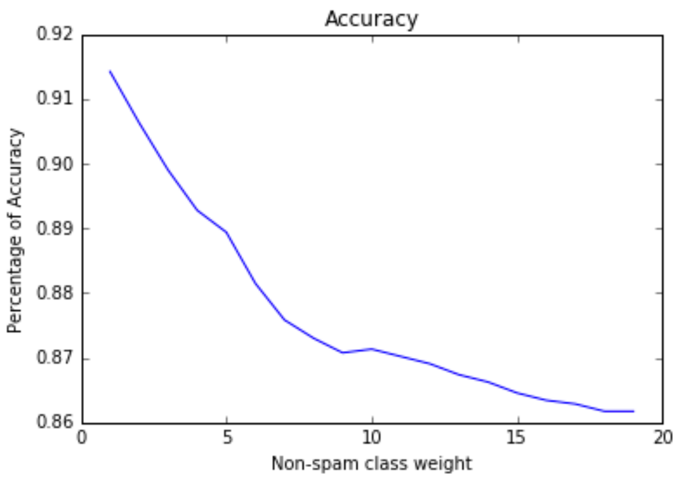
	Precision	Recall
KNN	0.906712172924	0.914181438999
Naive Bayes	0.855931326921	0.9386412612
SVM	0.955010859448	0.923954372624
Random Forest	0.965353190609	0.947614213198

The random forest classifier gives the best precision and recall among all four classifiers. Thus, the random forest classifier is the best for this data set, even it does not need data pre-processing.

5.3 Reduce False positives

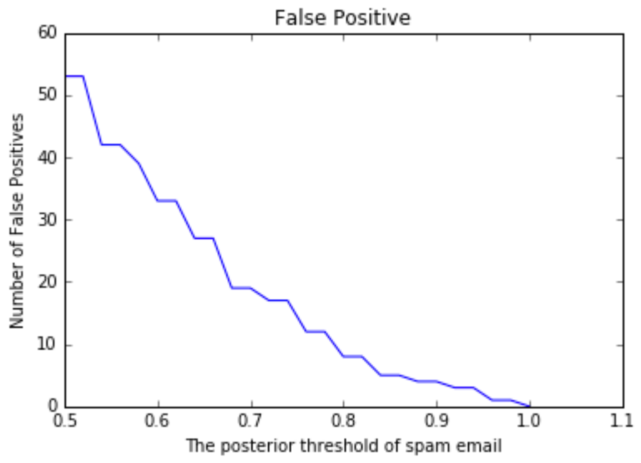
False positives means that make good emails as spam in this case. In real life, making such mistake is very undesirable. It is acceptable to receive some spam emails. However, it is horrible to miss an important email because there are some identical words which usually appears in the spam emails.
In this section, I use two ways to reduce the false positives in my predicting results.
(1) Change the class weight of SVM
By increasing the weight of non-spam email, we can reduce the false positives.

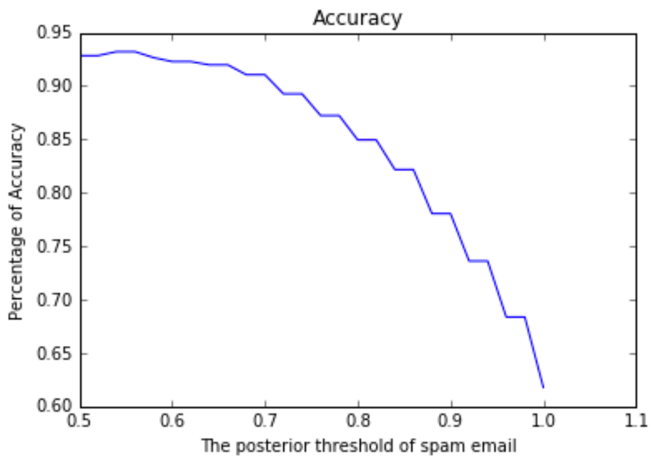




According to the figures above, the number of false positives is decreasing with the increasing of non-spam class weight. However, the accuracy is dropping when the non-spam class weight growing. When the weight of non-spam email is large than 12, the number of false positives keeps the same, but the accuracy is still going down. Thus, we can stop at the class weight of non-spam equaling to 12, which gives the smallest false positives and the highest accuracy.
Changing the class weight helps to reduce the false positives, but it can not guarantee that we can reach zero false positive. This is why I use the second method to reduce the number of false positives.

(2) Change the threshold of posterior of random forest
The random forest classifier use the posterior at leaves to determine which class one data point should belong to. It will set half to half by default to assign the class label to each data point. Thus, it is reasonable to change this threshold to let more data points to be in one certain class.





By changing the threshold, we can get zero false positive by classifying all emails as good emails. However, when the posterior threshold of spam email increasing to 1.0, the accuracy is only around 62%, which is very low.

We can get zero false positives by changing the decision rule of each classifier. For example, change the decision boundary of SVM and change the posterior threshold of Naive Bayes. It is better to change the decision rule of each classifier to reduce false positives than to change the class weight.

6. Conclusion

In the light of those experiments, the conclusion can be drawn.
(1) Random forest is a very good method in classification. The performance of random forest is hard to be exceed by other classifiers. In this case, random forest classifier is the best one among all four classifiers.
(2) Standardization before performing PCA to the dataset is very necessary. It is hard to find out whether the scope of each attribute or the mean of each attribute is the same. Thus, it is safe to standardize before performing PCA.
(3) PCA may not work for some case. We should not perform PCA blindly for every dataset. However, PCA has potential to improve the performance of classifier significantly.
(4) There is a tradeoff between getting fewer false positives and higher accuracy. Whether we should get fewer false positives or higher accuracy is depending on the requests of a certain task.
(5) It is better to change the decision rule of each classifier to reduce false positives than to change the class weight.

Reference

1. Should you apply PCA to your data? http://blog.explainmydata.com/2012/07/should-you-apply-pca-to-your-data.html
2. http://scikit-learn.org