Email Spam Detection Report

Abstract

We build a spam detector using 4 Machine Learning models and evaluate them with test data using different performance metrics used. The dataset we used was from a shuffled sample of email subjects and bodies containing both spam and non spam emails in different proportions, which we converted into stem word using stemmer. As per our analysis, Naive Bayes model and Random Forest models worked well for spam detection, whereas SVM performed the poorest among the 4 models.

Methodology¶

To prepare the data, we followed the steps below:

1. Read the spam email dataset given to use and read the datas et
2. Add length column in the dataset in order the visualize the rel ation of spam and non spam mails with respect to length.

3. Create a dataframe with message and label column which is t o be used as our actual dataset

4. Now clean the data by removing all the unnecessary spaces,s pecial character, digits and other punctuation using re library

5. Also use stopword to remove basic English words which does not add much details to the dataset.

5. Split the dataframe into train and test dataframes. The test d ata was 30% of the original dataset.

6. Split the message of mail into stem(that is root word) and ap ply Bag of words transformation using CountVectorizer.

7. Trained four models using the training data:
   a. Naive Bayes
   b. Decision Tree
   c. SVM
   d. Random Forest

8. Using the trained models, predicted the email label for test d ataset. Calculated four metrics to gauge performance of the mo dels:
   a. Accuracy
   b. Precision
   c. Recall

ML Models Employed

Email spam detection is a classification problem. Some algorithms like Naive Bayes Classifier, Decision Trees work well for spam detection. Algorithms like KNN, Linear Regression don't really work well due to inherent disadvantages such as curse of dimensionality.

A. Naive Bayes with bag of words

Naive Bayes is the easiest classification algorithm (fast to build, regularly used for spam detection). It is a popular (baseline) method for

text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features.

Why use Naive Bayes?

1. NB is very simple, easy to implement and fast because essentially you're just doing a bunch of counts.
2. If the NB conditional independence assumption holds, then it will converge quicker than discriminative models like logistic regression.
3. NB needs works well even with less sample data.
4. NB is highly scalable. It scales linearly with the number of predictors and data points.
5. NB can be used for both binary and multi-class classification problems and handles continuous and discrete data [2].
6. NB is not sensitive to irrelevant features.

B. Decision Trees

Decision trees are used for classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. The decision of making strategic splits heavily affects a tree's accuracy. The decision criteria is different for classification and regression trees. Information theory is a measure to define this degree of disorganization in a system known as Entropy. If the sample is completely homogeneous, then the entropy is zero and if the sample is an equally divided (50% – 50%), it has entropy of one. It chooses the split which has lowest entropy compared to parent node and other splits. The lesser the entropy, the better it is.

Why use Decision Trees?

1. Decision trees implicitly perform variable screening or feature selection. When we fit a decision tree to a training dataset, the

top few nodes on which the tree is split are essentially the most important variables within the dataset and feature selection is completed automatically
2. Decision trees are easy to understand, easy to represent visually and easy to communicate.
3. Nonlinear relationships between parameters do not affect tree performance. Also trees can explain the non-linearity in an intuitive manner.

C. Support Vector Machine (SVM)

SVM is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyperplane that differentiate the two classes very well [4]. Support Vectors are simply the coordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line). If the data requires non-linear classification, SVM can employe Kernels, which are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. they convert non separable problem to separable problem.

D. Random Forest

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process (say) 10 times and then

make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction[4]. Random forest gives much more accurate predictions when compared to simple CART/CHAID or regression models in many scenarios. These cases generally have high number of predictive variables and huge sample size. This is because it captures the variance of several input variables at the same time and enables high number of observations to participate in the prediction. Metrics Used

Following are the metrics we used to evaluate the performance of ML techniques:

1. Precision

Precision refers to the closeness of two or more measurements to each other. In Machine Learning, precision is the fraction of relevant instances among the retrieved instances. Precision = TP / (TP + FP) (Where TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative).

2. Accuracy

Accuracy refers to the closeness of a measured value to a standard or known value. Accuracy = (TP+TN) / ALL

3. Recall

Recall is how many of the true positives were recalled (found), i.e. how many of the correct hits were also found. Recall = TP / (TP + FN)

4. F-Score

F-scores are a statistical method for determining accuracy accounting for both precision and recall. It is essentially the harmonic mean of precision and recall.

.

Comparison of Performance of the Models:

1. Naive Bayes

For this problem, Naive Bayes algorithm worked well as expected. Following were the confusion matrix and scores:

confusion matrix
[[468 6]
[5 100]]

Accuracy
0.9810017271157168
Classification report

	precision	recall	f1-score	support
0	0.99	0.99	0.99	474
1	0.94	0.95	0.95	105
accuracy			0.98	579
macro avg	0.97	0.97	0.97	579
weighted avg	0.98	0.98	0.98	579

High accuracy suggests that the model is very good at correctly classifying the mails as ham or spam. Precision value is also good at

0.99, means the model has a low false positive rate. This can be corroborated by looking at false positives found - only 6.. This indicates that results are complete to a large extent. In other words, probability of detection is lower compared to Decision Tree and Random Forest. Together, having a high precision and low recall means that the most of spam predictions are correct, expected.

2. Decision Tree

Accuracy and Precision of the decision tree is low compared to Naive Bayes model. However, it has a higher Recall and F-Score. Good recall of 0.96 means predictions of decision tree are more complete compared to Naive Bayes. Good precision of 0.91 indicates that model has low false positive rate. Since the dataset has an uneven distribution of non-spam and spam, F-Score becomes an important metric. F-Score of 0.97 and 0.89 indicates a fairly good model in terms of precision and recall both.

```
[[455  19]
 [  6  99]]
accuracy
0.9568221070811744
Classification report
              precision    recall  f1-score   support

           0       0.99      0.96      0.97       474
           1       0.84      0.94      0.89       105

    accuracy                           0.96       579
   macro avg       0.91      0.95      0.93       579
weighted avg       0.96      0.96      0.96       579
```

3. SVM

confusion matrix
[[474 0]
[33 72]]
accuracy
0.9430051813471503
Classification report

	precision	recall	f1-score	support
0	0.93	1.00	0.97	474
1	1.00	0.69	0.81	105
accuracy			0.94	579
macro avg	0.97	0.84	0.89	579
weighted avg	0.95	0.94	0.94	579

4. Random Forest

confusion matrix
[[474 0]
[8 97]]
accuracy
0.9861830742659758

Classification report

	precision	recall	f1-score	support
0	0.98	1.00	0.99	474
1	1.00	0.92	0.96	105
accuracy			0.99	579
macro avg	0.99	0.96	0.98	579
weighted avg	0.99	0.99	0.99	579

Random Forest algorithm had a high precision and accuracy. Considering lower accuracy as compared to Naive Bayes, good precision value indicates low false positive rate. Recall value is also good, especially when compared to Naive Bayes. Having high precision and recall suggests that the model is correctly predicting positive class (spam) and also capturing most spam in the test data. It follows that the model also has a good F-score, since it is directly proportional to Precision and Accuracy.

Results:

It is clear from the comparison that SVM model did not work out very well to solve our problem of spam detection. Naive Bayes and Random Forest both work pretty well. While Naive Bayes algorithm is has a high accuracy and a good precision, the recall value is poorer compared to Decision Tree and Random Forest. Since SVM model could not predict any positive values at all, its accuracy, recall and F-score were 0. As far as the F-score is concerned, Decision Tree and Random Forest have a good score as a result of good precision and recall . Overall, we think that both Naive Bayes and Random Forest will be very good for spam detection. Of course, we can employ boosting or stacking ensemble

methods to combine two or more of these models into a really good spam detector.