An Ensemble Classifier for Rectifying Classification Error

Project Report

Aniket Gaikwad
anikgaik@indiana.edu

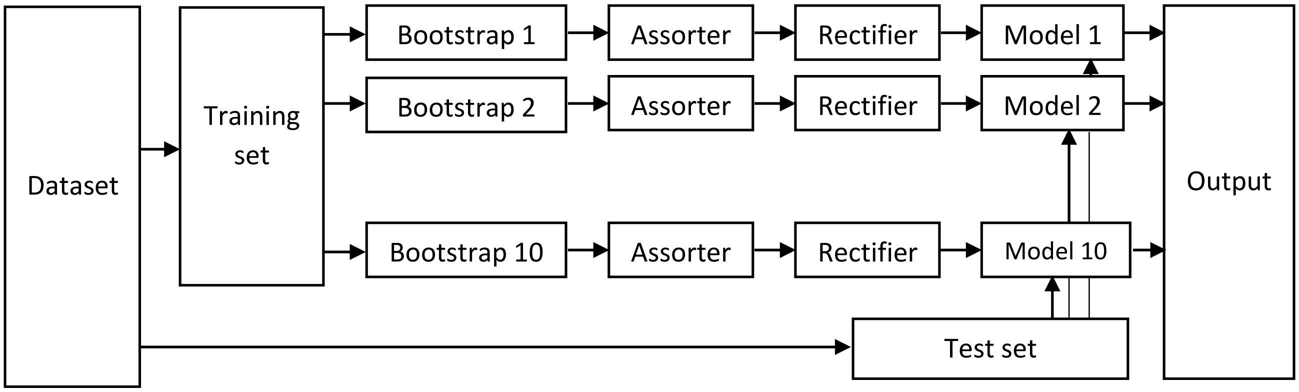
Dileep Vishwanathan
diviswan@indiana.edu

Introduction

This project is an implementation of the proposal by Cheuk Ting LI from Stanford, in his project - An Ensemble Classifier for Rectifying Classification Error. The main idea is to use ensemble methods to combine multiple classifiers in which each constituent classifier would focus on correcting errors made by the previous one. In our case we consider two constituent classifiers - an assorter and a rectifier. The first classifier, the assorter, would perform classification on the training set by generating a probability distribution of the class labels for each example and ranking them accordingly i.e. the label with the highest probability would be ranked 1 and so on. The subsequent classifier, or the rectifier, would use the rank as the class label to perform classification. The rectifier would then learn from the training set and decide if the prediction of the assorter could be used or of it would have to pick from the other classes determined as less likely by the assorter. To classify a test instance, run the assorter to obtain a ranking and pick the class label predicted by the rectifier. Intuitively, using ensemble methods such as a bagging and boosting on a high bias classifier as the assorter followed by a high variance classifier as the rectifier would enhance performance.

Model

Each dataset is divided into a training and test set with 66% of the instances picked randomly to constitute the training set while the remaining 34% form the test set. The training set is split into 30 bootstraps using random sampling with replacement such that 63% of the instances in each subset is unique. Each bootstrap is used to train an assorter such as Naïve Bayes or Logistic regression classifier which outputs the rank as the class label based on the probability distribution it generates. The output of assorter becomes the input to the rectifier which is a Decision tree or k-Nearest neighbor classifier which generates a model that predicts the output as rank which is then translated to the corresponding class label. The output of all the models are aggregated and the final predicted class label is determine based on majority vote.



Assorter-Rectifier model

Algorithm

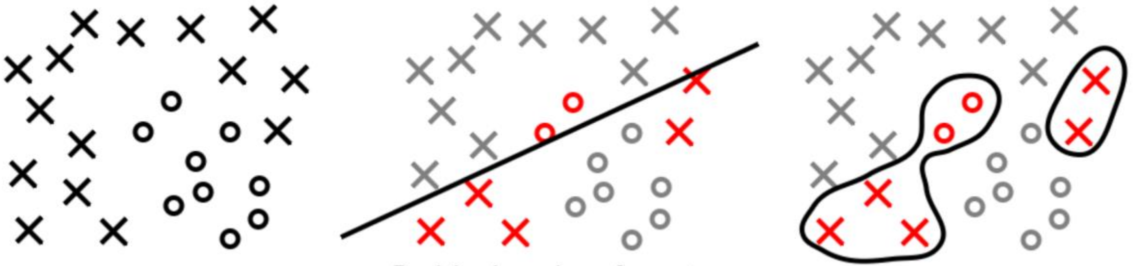
Training:

1. Assorter : Generate probability distribution of class labels
2. Define the rank for examples as per order of probability distribution
3. Replace the class labels for the training sets to be passed to rectifier by the rank of the correct class

   r^{-1} (y(i), ha(x(i))) = Rank
4. Train the rectifier on modified data - {x(i), r-1(y(i), ha(x(i)))}

Test:

1. For each test instance obtain probability distribution - $h_a(x)$
2. Output the class corresponding to the rank guessed by the rectifier



Training data

Decision boundary of assorter and incorrectly classified instances

Decision boundary of rectifier

Datasets

Four datasets from the University of California, Irvine (UCI) Machine Learning repository have been picked for analysis of the model. UCI repository became our choice for dataset selection because of the diversity and balance of the datasets available in it. We picked the following datasets:

1. Credit Approval Info
2. Breast Cancer
3. Bank Note Authentication
4. Parkinson's Disease

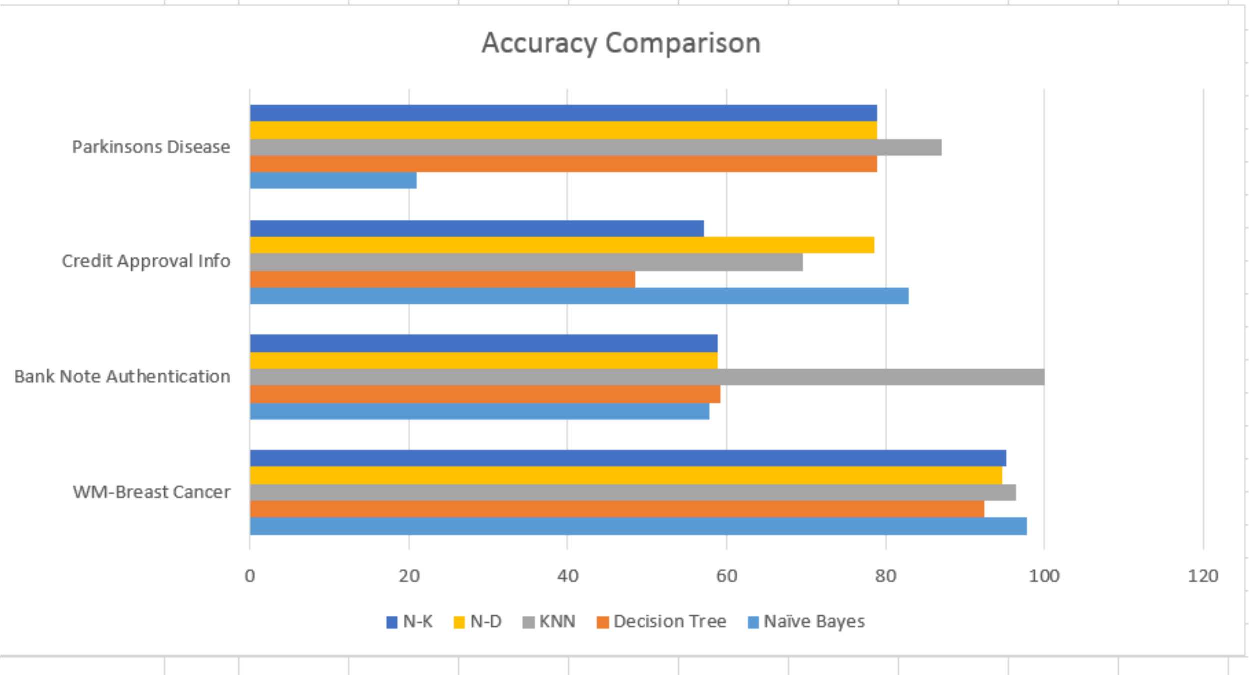
Preprocessing

Preprocessing is a very important step in analyzing the model. We had to preprocess the datasets to obtain certain information that would make life easier for prediction without losing out on important information. Our preprocessing involved 3 steps:

1. Collection of metadata

2. Handling unknown values
3. Normalization (Linear and Z)

Results:



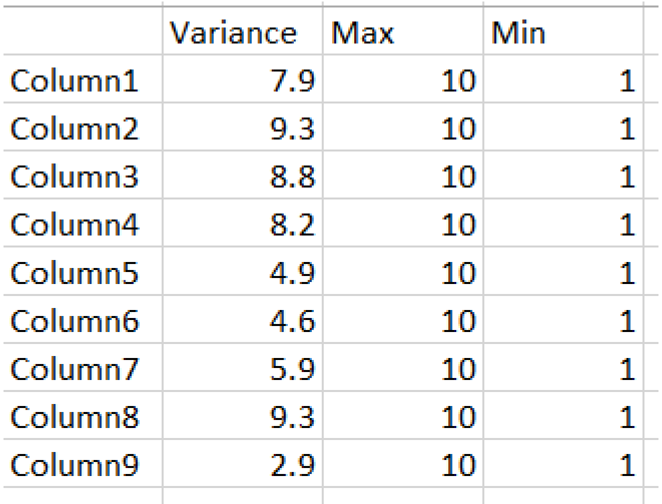
Analysis :

Our intuition of building this model is to counter wrongly classified examples of one classifier by another classifier who is able tackle those example. Our goal is to use different characteristic possed by different classifier in group to produce better results.Basically we are trying to introduce bias by using first classifier and then counter that bias by introducing variance in second classifier so that we can get optimal result. So if data is properly balanced then after passing it through first classifier we get distribution of data which is biased towards rightly classified examples. So the next classifier will focus on wrong examples which will be small quantity.

As per results we got, it's not totally proving to be the result we are expecting. But one of the interesting thing is observed from results that AR model always give results which is intermediate one. So for "Parkinson's Disease" dataset Naïve Bayes performs poorly however, its AR model performs considerable good. Same is case of "Credit Approval Info" datasets where decision tree don't show good results but AR model does.

For analysis we chose "Wisconsin Madison Breast cancer" dataset.

Dataset Statistics :



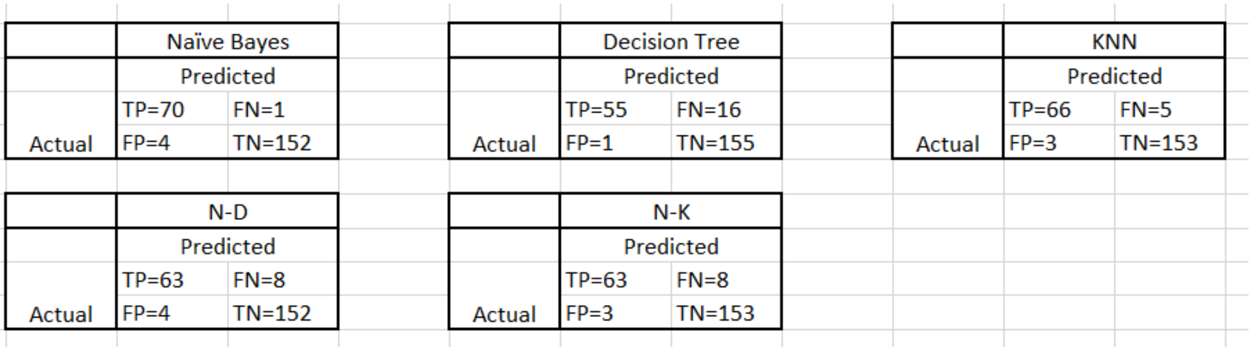
	Variance	Max	Min
Column1	7.9	10	1
Column2	9.3	10	1
Column3	8.8	10	1
Column4	8.2	10	1
Column5	4.9	10	1
Column6	4.6	10	1
Column7	5.9	10	1
Column8	9.3	10	1
Column9	2.9	10	1

As we can see, there isn't much difference in variance of dataset. So data is evenly balanced. There are 16 unknown values. So we decided to ignore those values.

When tested individual Naïve Bayes we got accuracy of 98% while individual decision tree got accuracy of 92%. When we used Assorter-Rectifier model (Naïve Bayes –Decision trees) we get accuracy of 94%. So performance increased by 2% as compared to Decision tree but as compared to Naïve Bayes by 4%.Possible reason we think is that we used 30 Bootstrap (Bagging) which will introduce bias , also naïve Bayes being stable classifier will introduce bias of its own. As per formula of Error = Bias^{2} +Variance+Noise ,bias will tend to impact the accuracy. We used decision tree of moderate height of 29 so it won't be much over fitting (introducing moderate variance). So possible error in output is due to introduction of more bias.

With respect to Naïve Bayes and KNN combination model, our result of AR model is that our result has actually less that both of classifiers. We used K=1 , so we are over fitting the data to get high variance (As we try with moderate variance in decision tree , we think of using 1-NN to see effect of high variance as immediate rectifier). For this we expecting at least same accuracy as 1-NN.They are almost similar.

Confusion Matrix:



	Naïve Bayes	
	Predicted	
	TP=70	FN=1
Actual	FP=4	TN=152

	Decision Tree	
	Predicted	
	TP=55	FN=16
Actual	FP=1	TN=155

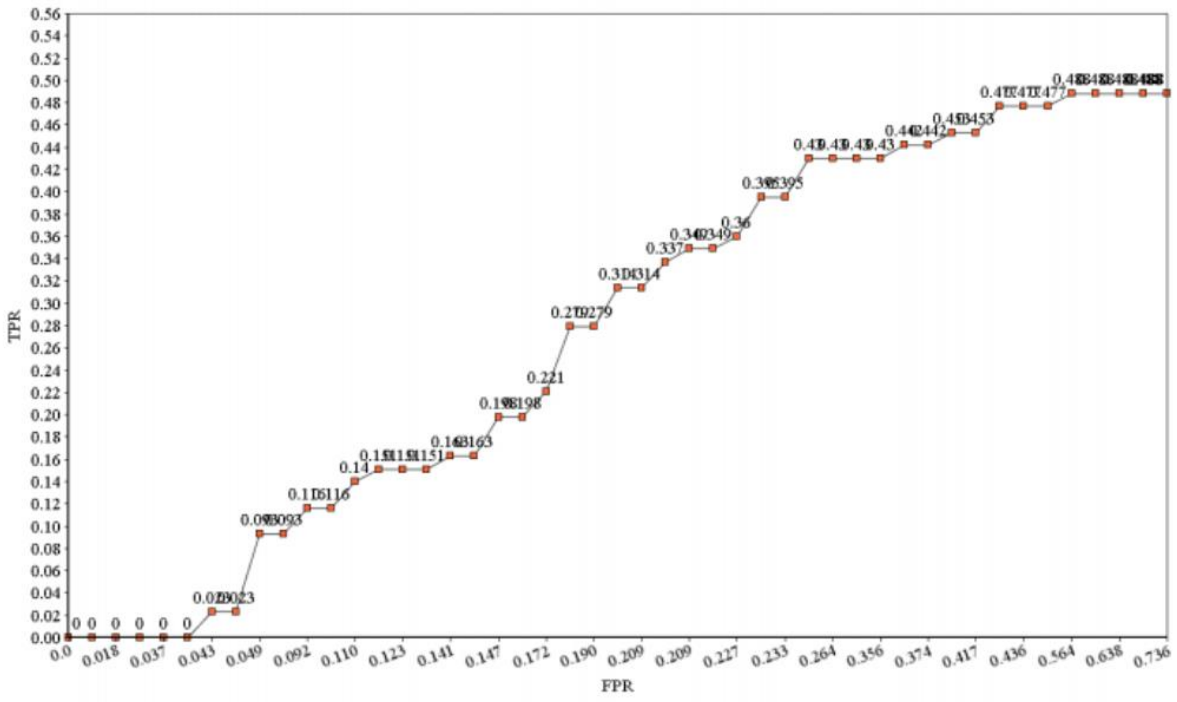
	KNN	
	Predicted	
	TP=66	FN=5
Actual	FP=3	TN=153

	N-D	
	Predicted	
	TP=63	FN=8
Actual	FP=4	TN=152

	N-K	
	Predicted	
	TP=63	FN=8
Actual	FP=3	TN=153

As per confusion matrix, if we compare N-K Model (Naïve Bayes – KNN model) with individual KNN, some of the example which are expected to be "malignant"(classlabel=4) has been predicted as "Non-malignant"(classlabel=2). (Difference between True positive and False Negative in KNN and N-k).The reason we think is that if two example are close enough and has different class label caused this.

ROC Curve of N-D Model :



Conclusion and Future work:

As per results we got we can conclude that, even though we didn't get better result than both of classifier which has been used in combination, but we get results which are better than at least one of the classifier used in model-combination. Further modification to the existing model could be addition of actual class labels as attribute along with rank function. Actual label as attribute will be assigned higher weight.

References

[1]http://cs229.stanford.edu/proj2013/Li-AnEnsembleClassifierForRectifyingClassificationError.pdf