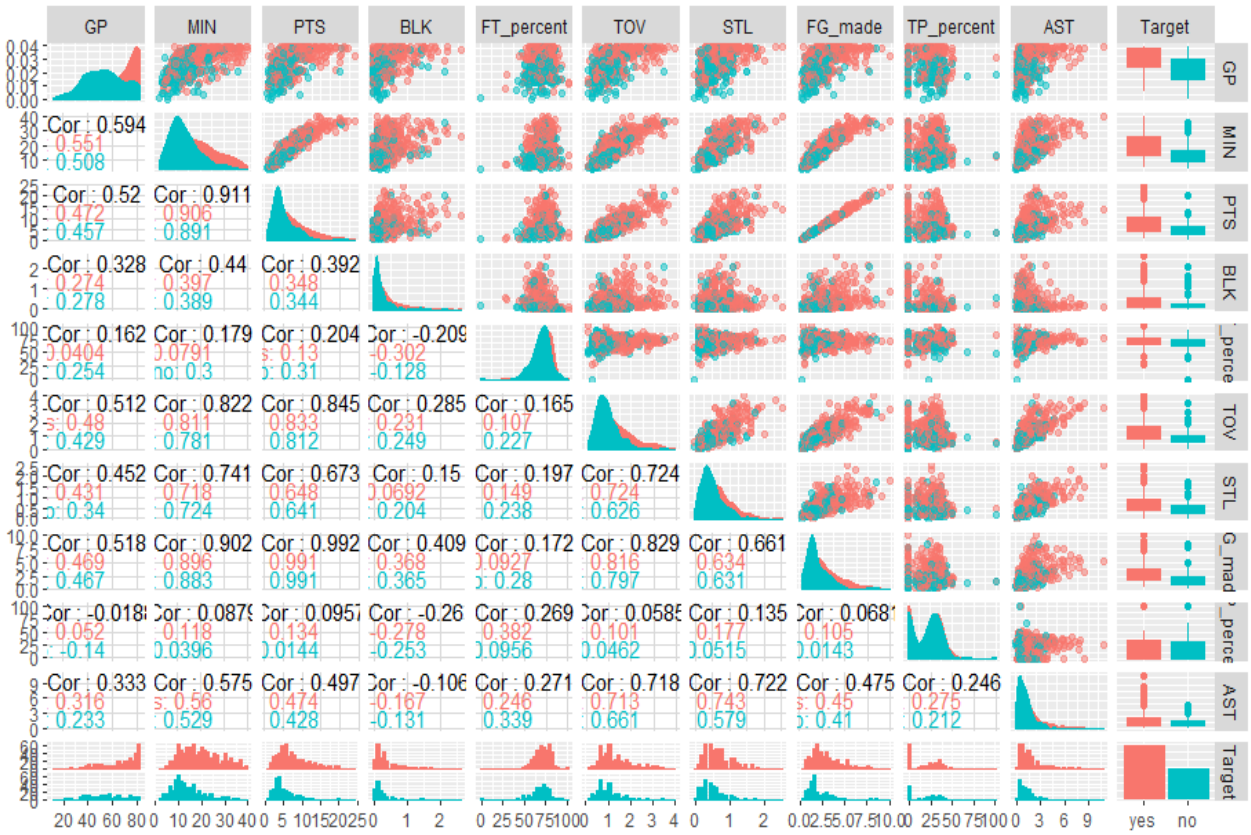
Machine Learning

Introduction

In this project , different classifications methods will be used to predict whether a basketball player will play more than 5 seasons in the NBA by just looking at their statistics from their rookie seasons

Exploratory Data Analysis

Our data comprises of 12 variables and 1069 observations. The variable of interest is Target which indicates whether a basketball player had an NBA career more than 5 years (it takes the value 1 in that case and 0 otherwise).From the data,the number of 1's for Target variable outnumber the 0's (Imbalanced classification) Looking at the data variance,there are big differences between the variables. Also,most of the variables are fairly high correlated. Number of outliers are detected. Moreover,all variables are highly skewed.



Methodology

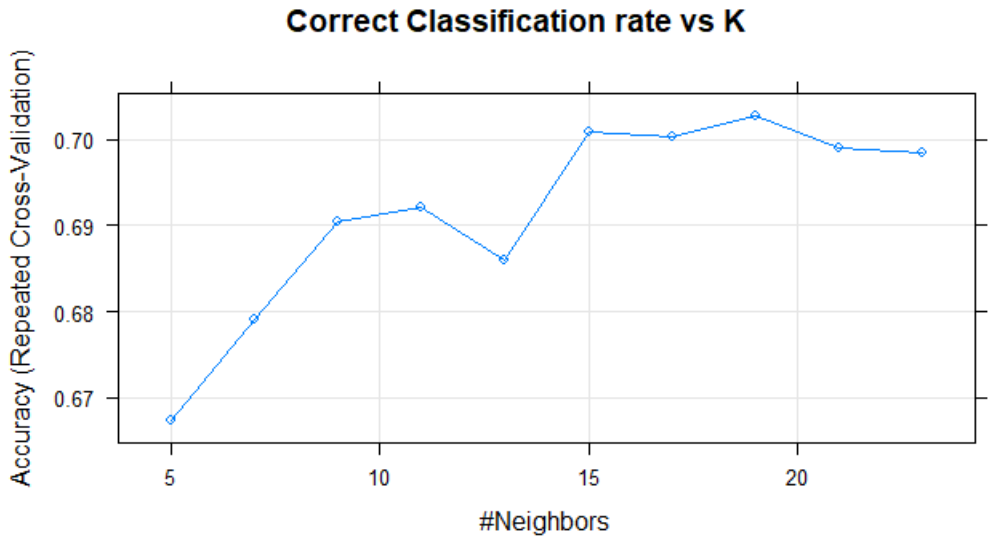
We split the data to training set,validation set and test set. The model is built on the training set, and evaluated on the test set. The validation set is used for parameter selection and to avoid over-fitting. When 10-fold repeated cross validation is used ,a number of temporary validation sets from the training set are created so there is no need for distinct validation set. As the result of using cross validation,most of our results are the same for valid set and test set because the valid set here is no longer part of the model building process. Three kinds of algorithms are use:

- k-nearest neighbor
- Tree based modeling
- Support vector machine (SVM)

Most of the algorithms are applied using CARET package.

k-nearest Neighbor

We fit the classification model using train function and 10-fold repeated cross validation to choose the optimal value of k. Also, as the data is highly variable as stated before,the data is centered and scaled using pre-processing option in train function. The figure below show the correct classification rate for the validation data versus k to choose the optimal one. The best k close to the theoretical value,square root of the number of observations ,which is 19.

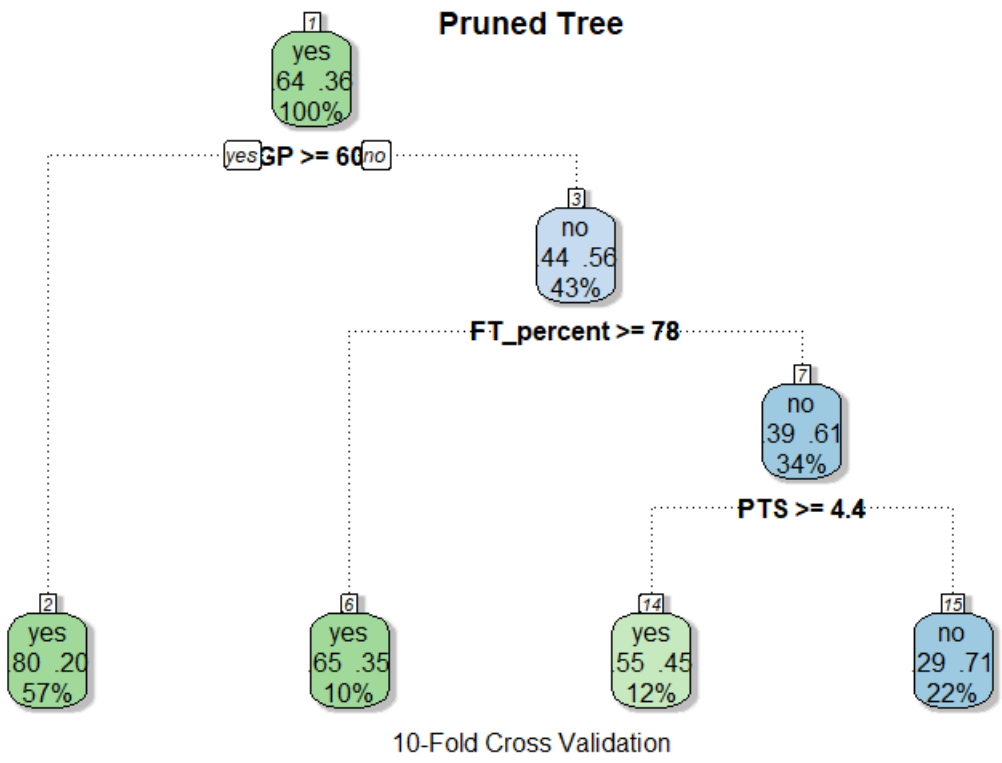


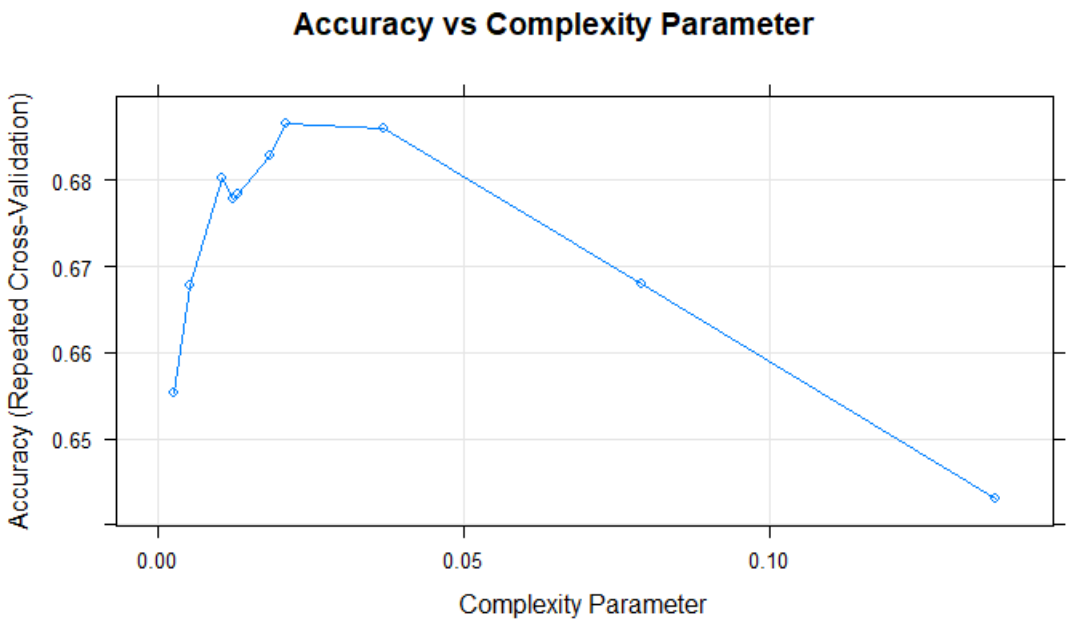
Tree Based Modelling

Fully grown Tree and pruning

Fully grown trees tend to be over-fitted ,may fail to predict future observations and suffer from high variance. We prune the tree with optimal complexity parameter(cp=0.02) to reduce the size in order to improve predictions and create generalized model. To choose the optimal cp,10-fold repeated cross validation is used. We used rpart function to model the tree and train function with 10-fold repeated cross validation to prune it (Figure) . It can be noticed that accuracy of the pruned tree is decreased to give us approximation of the real model accuracy without over-fitting.

```
## [1] "Accuracy of full Tree 0.6742"

## [1] "AUC of  full tree 0.6933"

## [1] "Accuracy of pruned tree 0.6667"

## [1] "AUC of pruned Tree 0.6626"
```

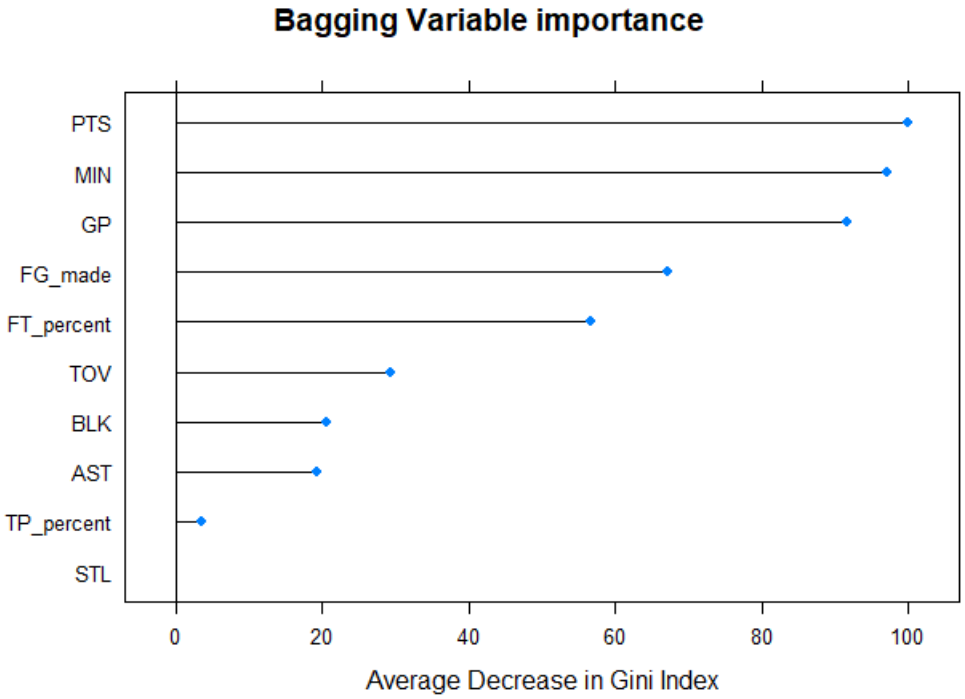




Bagging

In order to reduce the trees high variance,bootstrap aggregation (bagging) is used. First we fitted the model with randomforest function and then we used 10-fold repeated cross validation. From the model output,the number of trees should be at least 100 to reduce the error. The out of bag error (oob) is 32.4% . More realistic values of the accuracy and AUC are given by the cross validated model. The variable importance figure shows that points per game(PTS) and minutes per game (MIN) are the most important variables, and steals per game (STL) is the least.

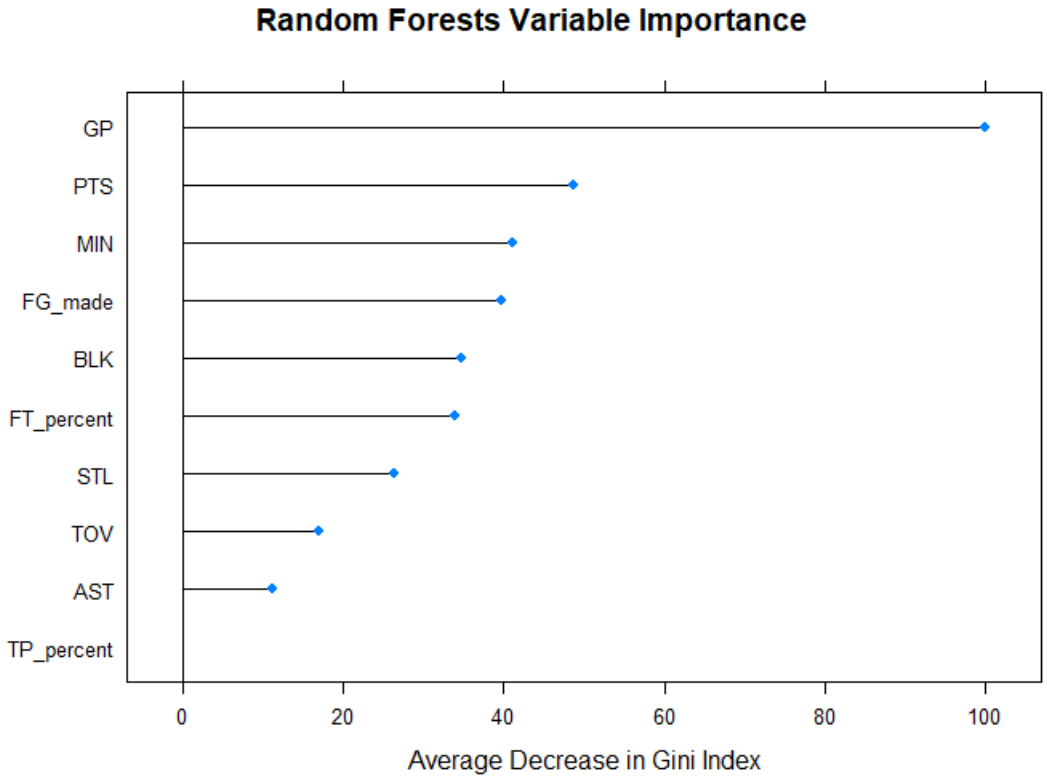
```
## [1] "Accuracy 0.6704"

## [1] "Bagging AUC 0.7123"

## [1] "Accuracy of Bagging with Cross Validation 0.6629"

## [1] "Bagging AUC with cross validation  0.7076"
```



Random Forests

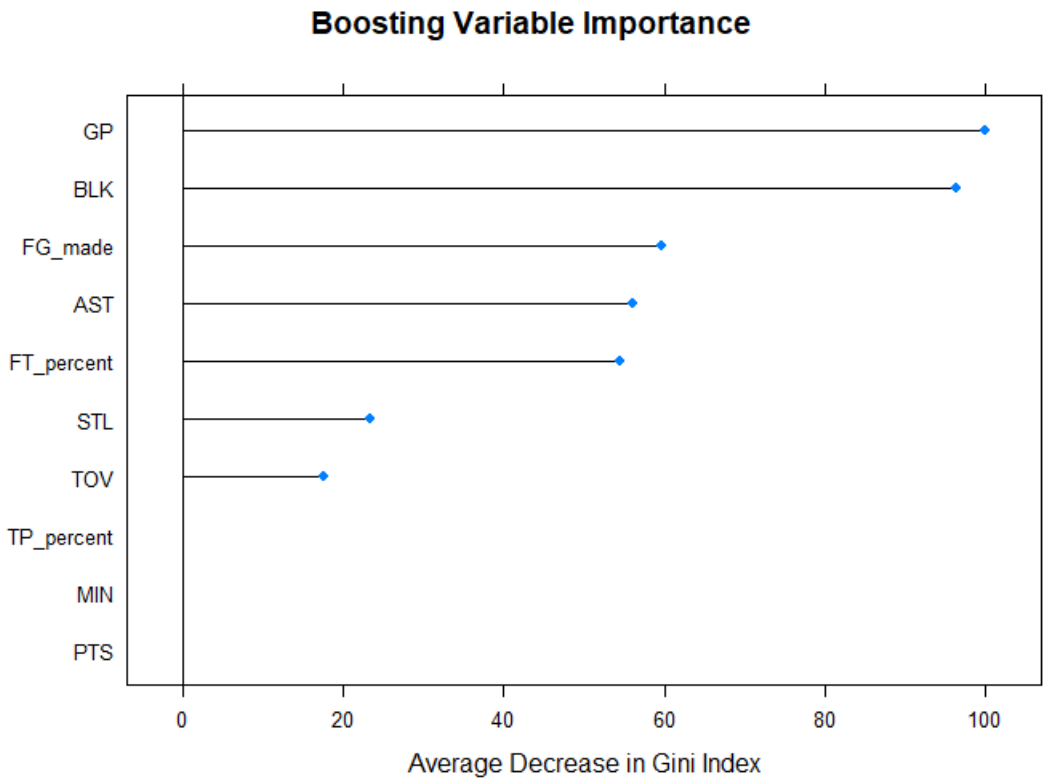
Rather than using all the features to grow the tree,random forests takes random selection of features. This will result in uncorrelated trees and averaging across them will highly reduce the variance. First we fitted random forest with randomforest function .The number of trees used is 500 and number of variables at each split matches the theoretical one,square root of number of predictors,which is 3. The OOB is 31.84% which is less than bagging by 0.5%. Then we used 10-fold repeated cross validation for parameter tuning. The optimal number of features used is 6. The accuracy and AUC shows that random forests provides an improvement over bagging. Same like bagging the number of trees should be at least 100 to reduce the error. Moreover,the PTS is not exclusively used to generate the first split for each tree like bagging. The most important predictors are GP and PTS and TP_percent is the least one.

```
## [1] "Random forest Accuracy 0.6629"

## [1] "random forest AUC 0.737"

## [1] "Random forest with cross validation Accuracy 0.6667"

## [1] "Random forest with cross validation AUC 0.7342"
```



Boosting

Trees are grown sequentially,each tree is grown using past fits of the training data from previously grown trees. We used too kind of boosting packages gbm and C50 and compare between them. Our results show that C50 gives higher accuracy and AUC compared to gbm.(More details and figures in the code).The accuracy of C50 is 0.74 and the AUC is 0.74. GP and BLK are the most important variables and MIN is the least one.

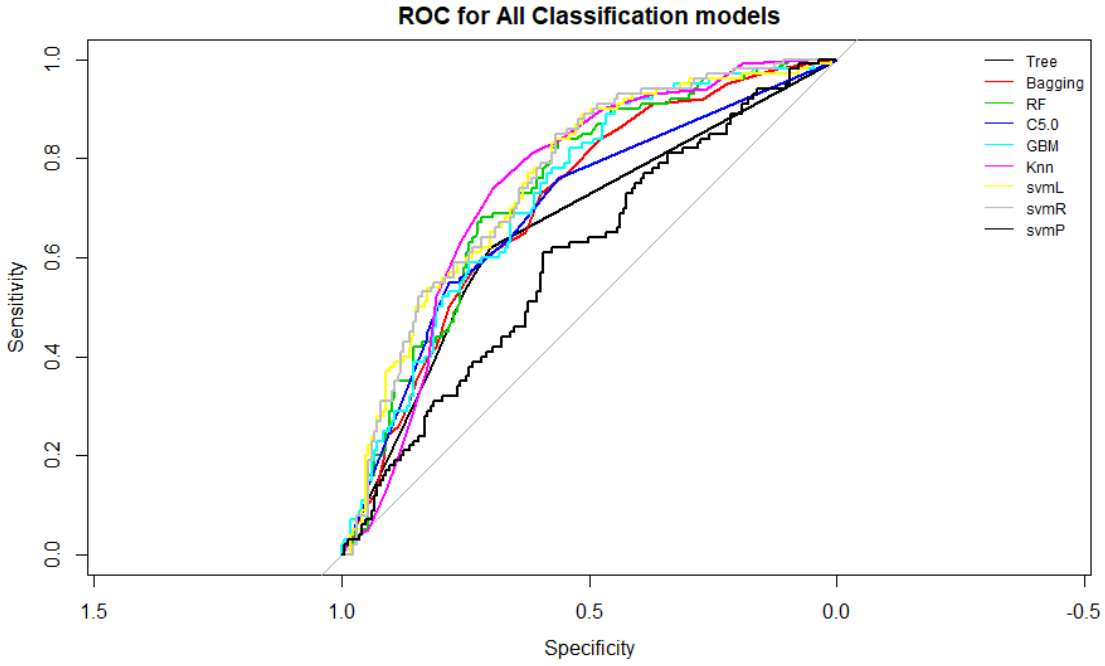


Support Vector Machine (SVM)

Svm uses kernel trick which converts not separable problem to separable one. We used three kind of kernels linear polynomial and radial. The kernel trick depends on inner product between all observations. So we need to scale the data in order to avoid variables in greater ranges dominating the variables with smaller ranges. Comparing the three kind of kernel performance,The radial is the best one.

```
## [1] "Accuracy of Linear kernel 0.7079"

## [1] "Accuracy of tuned linear kernel 0.7079"

## [1] "AUC linear 0.7528"

## [1] "AUC tuned linear  0.7527"

## [1] "Accuracy of radial kernel 0.7154"

## [1] "Accuracy of tuned radial kernel 0.6891"

## [1] "AUC radial  0.7309"

## [1] "AUC tuned radial  0.756"
```

```
## [1] "Accuracy of polynomial kernel 0.6404"

## [1] "Accuracy of tuned polynomial kernel 0.618"

## [1] "AUC polynomial 0.7065"

## [1] "AUC tuned polynomial  0.5991"
```



Results and Discussion

As it's mentioned before,in the response variable the number of 1's (positive) is more than the number of 0's (negative).That will result in a biased classifier towards the majority class. So when the classifier is trained on unbalanced data and applied to also unbalanced data,the test data set,the estimated accuracy is optimistic. So the using the accuracy as a performance measure and comparison between classifiers is misleading. Different performance measure is used like balanced accuracy which is defined as the average accuracy produced from each class. Also, ROC curve which consists of the Sensitivity and specificity, is considered and AUC is Calculated.

Classification	Accuracy	Balanced Accuracy	kappa
Random Forests	0.7649	0.7126	0.4479
Boosting-C5.0	0.7425	0.6903	0.3987
SVM Radial	0.7414	0.6523	0.3373
Boosting-GBM	0.7388	0.6848	0.3882
Knn	0.7239	0.6815	0.3709
Bagging	0.7052	0.6487	0.3115
Tree	0.6866	0.6025	0.2269
SVM Linear	0.6866	0.6399	0.2859
SVM Polynomial	0.6754	0.5327	0.0828

Classification	System Time(s)	Object Size (Bytes)
Random Forests	75.83	3,705,680
SVM Radial	18.12	170,832
Boosting-GBM	11.09	1,561,160
SVM Linear	10.25	165,784
SVM Polynomial	8.36	173,216
Bagging	5.76	6,230,408
Boosting-C5.0	5.23	969,136
Knn	2.44	411,496
Tree	1.88	1,722,320

Classification	AUC	Sensitivity (TPR or recall)	Specificity (TNR)	Precision (PPV)	F-Score
Random Forests	0.7903	0.8757	0.5495	0.7908	0.8311
Knn	0.7774	0.8136	0.5495	0.7784	0.7956
Boosting-GBM	0.7739	0.8531	0.5165	0.7744	0.8118
Bagging	0.7554	0.8249	0.4725	0.7526	0.7871
SVM Radial	0.7494	0.8983	0.4066	0.7465	0.8154
Boosting-C5.0	0.7400	0.8531	0.5275	0.7784	0.8140
SVM Polynomial	0.6600	0.9774	0.0879	0.6758	0.7991
Tree	0.6451	0.8644	0.3407	0.7183	0.7846
SVM Linear	0.5000	0.7853	0.4945	0.7514	0.7680

Looking at the table results,we find that: -Random forests has the higher accuracy balanced accuracy, and AUC.It has a fairly high sensitivity (fraction of positive classes correctly identified) coming after svm with radial and polynomial kernel. Also,it has the highest specificity among all models,that means the highest ability to correctly classify the negative classs. The kappa value is moderately good (between 0.4-0.6).Also it takes longer time in training than other classifiers. Moreover,it consumes larger amount of memory than most of the classifiers except for bagging. Random forests uses the variable importance as a kind of interpretation.

- Boosting with C5.0 has the second higher accuracy and balanced accuracy. The AUC is better than single tree and svm (linear and polynomial) but lower than rest of classifiers. Its ability to correctly classify the negative classes is good compared to other classifiers. Also the kappa value is fair(0.2-0.4).It spends small time in training compared to other classifiers and consumes less memory than all tree based models.C50 is similar to adaboost and perform better than

gradient boosting machines (gbm).Tuning parameters is very important and computationally expensive.

- SVM : Radial kernel has higher accuracy than linear or polynomial kernel. It has higher sensitivity than random forests but lower specificity . It can be used if our main focus is positive class detection. It has the second higher training time after random forests ,but it consumes low memory in comparison to other classifiers. The final model and variables weights are difficult to understand and to interpret. SVM can be over-fitted easily

- Boosting with GBM takes more training time and consumes memory more than most of the predictors. Its performance is comparable to the other classifiers but it consumes higher memory and has longer training time. Generally GBM has better performance than random forests but its harder to tune.

- Knn has a lower accuracy than random forests but it has the same specificity,the ability to detect negative class . The training time is less than most of the algorithms but the prediction time is high. It needs pre-processing to the data. The kappa value is within the fair range. Knn is simple to understand and easy to implement , but high in computational cost. It can suffer from unbalanced class and tend towards the majority one.

- Bagging performance is less than random forests due to the existence of correlated features. That's result in still high variance and over-fitting. Also, bagging consumes higher memory than all other classifiers. Bagging is not as easy to visually interpret but variable importance is also used a random forests.

- Single Tree has the worst performance between all of the classifiers just after the svm linear. It can be over-fitted easily. Its simple and easy to interpret for anyone with any background. It takes shorter time to train than all other classifiers.

Conclusion

We have shown that random forest ,svm radial and boosting (C5.0 and GBM) are classification algorithms that most likeley will result in the highest accuracy. In terms of training time svm radial and boosting are faster than random forests . SVM radial and boosting parameters selection is very important and computationally expensive. Boosting and random forests are easier to understand and interpret than svm radial.

For the NBA data, we will assume that both classes detection is equally important. As the result of this assumption ,we choose random forests as the best classifier to our data.

For future work, We can try different kind of boosting like XGBoost. Also,Parallel random forests may improve the accuracy significantly. We can also try stacking

which is building primary models and supervisor model combines their predictions efficiently. Moreover, we can use inference to make sure there is a significant difference in performance between classifiers

Optional Assignment

Predicting the Target variable using random Forests.

	yes	no	prediction
1	0.75	0.25	yes
2	0.708	0.292	yes
3	0.822	0.178	yes
4	0.826	0.174	yes
5	0.616	0.384	yes
6	0.51	0.49	yes
7	0.498	0.502	no
8	0.972	0.028	yes
9	0.944	0.056	yes

The positive predictive value =0.7908 (how likley for someone predicted as positive to be actually poistive)

The negative predictive value = 0.6944 (how likely for someone predicted as negative to be actually negative.

For the poistive predictions ,the probability these predictions are right is 0.7908

For the negative predictions ,the probability these predictions are right is 0.6944

Also The positive likelihood ratio = 1.94 increases the probability of having 1

The negative likelihood ratio = 0.226 decrease the probability of having 0.