Multivariate Analysis

Subhrajyoty Roy (MB-1911)
November 11, 2019

1 Introduction

The National Basketball Association (NBA) is a men's professional basketball league in North America, composed of 30 teams (29 in the United States and 1 in Canada). It is one of the four major professional sports leagues in the United States and Canada, and is widely considered to be the premier men's professional basketball league in the world. It is of utmost importance for the team managers to scout promising players from college and also figure out whether a player will be in a basketball career long enough (at least 5 years) to create specific contracts with the player for the team.

This project aims to find a simple statistical basis of classifying whether a rookie NBA player has `5 year carrer longeveity` i.e. whether a rookie NBA player will remain a popular and professional basketball player based on the predictors / features of his playing statistics and skills in college level.

2 The Dataset

The dataset was obtain from the website **data.world**.

The dataset comprises of the information of 1340 NBA rookie players, with information on 20 variables as predictors and one binary response, which takes value 1 if that player has a career length of at least 5 years and 0 otherwise. The 20 features / predictors are as follows:

1. Name.
2. Games played.
3. Minutes played.
4. Points per game.
5. Fields goal made per game.
6. Fields goal attempted per game.
7. Fields goal success rate = 100 × Fields goal made / Fields goal attempted.
8. 3 Pointer made per game.
9. 3 pointer attempted per game.
10. 3 pointer success rate = 100 × 3 pointer made / 3 pointer attempted.
11. Free throw made per game.
12. Free throw attempted per game.
13. Free throw success rate = 100 × Free throw made / Free throw attempted.
14. Offensive rebound per game.
15. Defensive rebound per game.
16. Total rebound per game = (Offensive rebound + Defensive rebound) / Total games played.
17. Assist per game.
18. Steals per game.
19. Blocks per game.
20. Turnovers per game.

It is clear that the `Name` column should not be an useful predictor of the response variable `TARGET_5Yrs`. Therefore, we remove the `Name` variable from the dataset.

It was also found that the `3P%` contains some `NA` values. The reason for this is possibly because of the indeterminate 0/0 form, when the player has not made an attempt of 3 pointer ever, since professional players usually never try to attempt a shooting from 3 Pointer range because of its difficulty, unless they are absolutely confident. Hence, for those values, we set `3P%` to 0 by default.

3 Descriptive Statistics

3.1 Correlation Analysis

Before proceeding with a formal analysis of the dataset, it is good practice to perform some exploratory analysis of the data. Firstly, we consider the correlation structure of the variables present in the data to see whether multicollinearity is present. It is found that there are clusters of variables which are very much correlated to each other (Figure 1). Therefore, this indicates that the data might lie in a very low dimensional vector space within $\mathbb{R}^{19}$, therefore, use of Principal Component Analysis (PCA) might help to visualize the clusters and help us predict the target response.

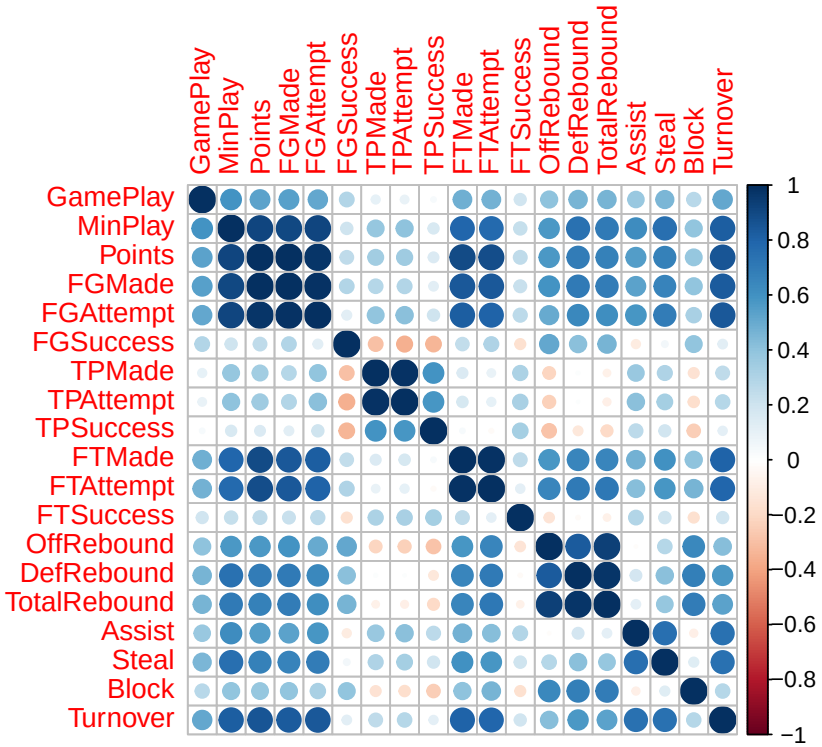

Figure 1: Correlation Plot of the Variables in the dataset

3.2 Principal Component Analysis

To perform the principal component analysis, we use all the 19 variable together, since all of them are numerical in nature. The screeplot helps us to understand how much effective PCA is for this particular dataset. We see that almost 90% of the variability is accounted by only 3 principal components, and first 4 principal components together accounts for 95% of the variability (Figure 2). However, only 2 principal components at a time fail to capture the true clusters. Bur, the plot for PC1 vs PC3 seems to be the best among above, in terms of splitting the two clusters (Figure 3).

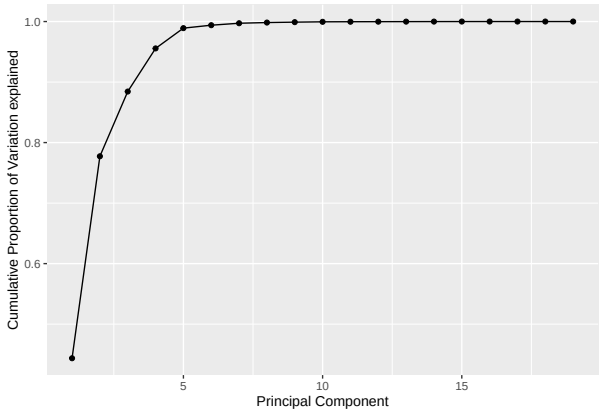

Figure 2: Screeplot of Principal Components

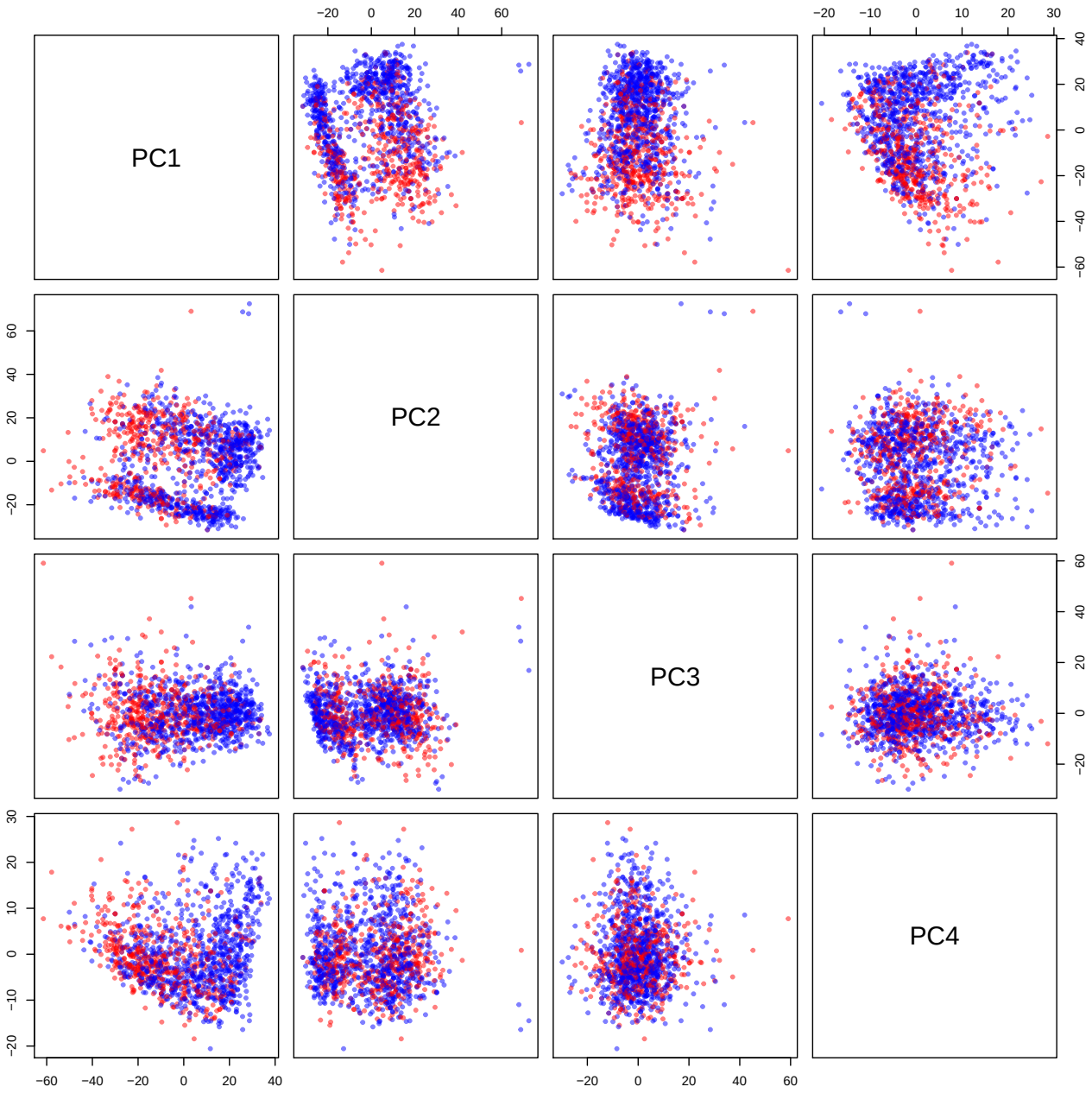

Figure 3: Pairs plot of Principal Components indicated by True labels

3.3 Creation of Training and Testing Set

Now, we shall use some simple classification tehcniques (described in class) which would help us in achieving the goal to predict the `5 year carrer longeveity` of NBA players. For this reason, to compare performances of various classification techniques, we randomly divide the whole dataset into two parts, one containing 75% of all the observation, which is used to train the classification algorithms called `Training Set` and the rest 25% will be used to measure the performance of those trained classification learners, called `Testing Set`. Note that both the training set and testing set contains similar proportion of target response variable.

Set	Number of Target = 0	Number of Target = 1	Total
Training Set	370	635	1005
Testing Set	139	196	335

4 Predictive Analysis

4.1 Discriminant Analysis

4.1.1 Linear Discriminant Analysis (LDA)

In this section, we use a linear discriminant analysis to predict the target variable. The inherent assumption of LDA model is that the datapoints under both class are distributed according to a multivariate normal distribution, with same covariance matrix, but with different means. Based on this, the naive Bayes classifier (MAP estimation) would give the form of a linear separating hyperplane. It was found that the contribution of most of the variables to form the separating hyperplane was very less. While Three Pointers made, Free Throw made, Offensive Rebounds, Block etc. has a positive impact, the variables like Three Pointers attempted, Turnovers etc. has a negative coefficient to form the decision boundary.

Next, the prediction for the response variable of each player is made, and the accuracy is obtained as the total number of correct predictions divided by the total number of predictions made. Clearly, as there is no class imbalance problem, the accuracy is not heavily affected, and is a good measure for comparison of different techniques. On the training set, the accuracy is found to be 73.9%, while on the testing set the accuracy is found to be 67.4%.

4.1.2 Quadratic Discriminant Analysis

Similar to LDA, we also consider using QDA as a potential classifier. It theoretically improves upon the situation of LDA by allowing different groups / levels of response to have different covariance matrices. Therefore, rather than obtaining a linear classifier, QDA produces a quadratic decision boundary. However, the training set accuracy drops to 71% and the testing set accuracy drops to 65%. Therefore, it suggests that that the assumption that both the classes may have different covariance matrix is not quite valid. It also provides an indication why principal components fail to detect the classifiers, since the variance structure is same for both the levels of response variable.

4.2 Classification Tree

Now, we use a classification tree to predict `5 year career longeiveity`, the response variable. We use Gini index as a measure of disparity. The resulting tree turns out to be quite small, with 7 leaf nodes, and according to it, the most important variables turn out to be Number of Games played, Offensive Rebounds, Fields Gold Made and Three Pointer Success percentage (Figure 4). Evaluating its performance, we see about 72% accuracy in training set, and about 66% accuracy in testing set.

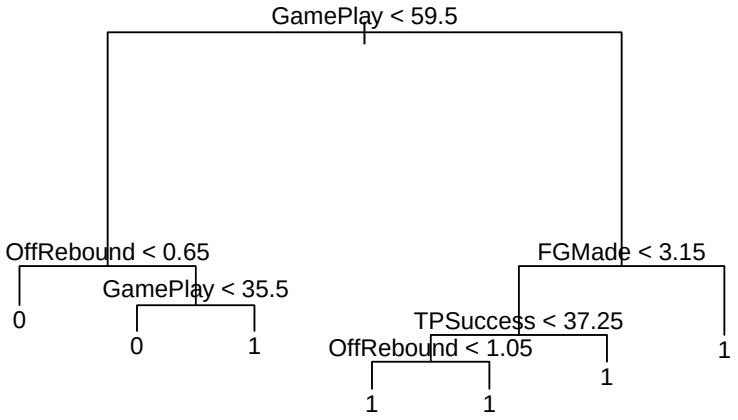

Figure 4: Trained Classification Tree to predict the response

4.3 Support Vector Machine (SVM)

Finally, we use a Support Vector Machine (SVM) to perform the binary classification of the response variable. Support Vector Machine basically maps the feature space (i.e. the column space of covariates) to a higher dimensional space by computing various nonlinear functions through computation of kernel functions (or generalized dot products), and then obtaining a linear separating hyperplane through that higher dimensional feature space.

The best SVM Classifier was obtained with a radial (or gaussian) kernel function, which obtained 75% accuracy in training set and about 72.8% accuracy in testing set. SVM with linear, polynomial or sigmoid kernel gives a result of lesser accuracy.

5 Conclusion

From the correlation analysis, it seems evident that some of the variables are very much correlated. However, a principal component analysis fails to separate the two classes because their variance structure is pretty much similar. It is also suggested by the result of Linear Discriminant Analysis.

However, Quadratic Discriminant Analysis, Classification Tree could not improve upon the prediction power of LDA. However, Support Vector Machine (SVM) does slightly better on both training and testing set. Also, SVM with sigmoid kernel is essentially similar to that of simple logistic regression, hence its accuracy being low would suggest that logistic regression would also not give a very good classifier in this context.

Therefore, in terms of interpretability and simplification of classification models, LDA would be a reasonable classifier for this data. However, in terms of performance metric of accuracy, SVM seems much better. But, a mere 72% accuracy for predicting the career longeiveity is not so good. It may be possible that there are some other factors (like number of fans, popularity measures etc.) which actually controls the target response, the variables in the dataset may not be adequate enough to explain whether a basketball player will pursue long career in NBA. However, the above study provides insights to which variables are important for such predictions.

6 Appendix

6.1 Code

The necessary codes and corresponding outputs to reproduce the analysis is shown here.

```
# read the data into R
library(readr)
nbadata <- read_csv('./nba_logreg.csv')
head(nbadata)
nbadata <- nbadata[, -1]  # remove name column
nbadata$TARGET_5Yrs <- factor(nbadata$TARGET_5Yrs)  # convert to factor variable
dim(nbadata)  # dataset contains 1340 observations
summary(nbadata)
```

```
# A tibble: 6 x 21
Name     GP   MIN   PTS   FGM   FGA `FG%` `3P Made` `3PA` `3P%`   FTM
<chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>     <dbl> <dbl> <dbl> <dbl>
1 Bran~    36  27.4   7.4   2.6   7.6  34.7       0.5   2.1  25     1.6
2 Andr~    35  26.9   7.2   2     6.7  29.6       0.7   2.8  23.5   2.6
3 JaKa~    74  15.3   5.2   2     4.7  42.2       0.4   1.7  24.4   0.9
4 Mali~    58  11.6   5.7   2.3   5.5  42.6       0.1   0.5  22.6   0.9
5 Matt~    48  11.5   4.5   1.6   3    52.4       0     0.1   0     1.3
6 Tony~    75  11.4   3.7   1.5   3.5  42.3       0.3   1.1  32.5   0.4
# ... with 10 more variables: FTA <dbl>, `FT%` <dbl>, OREB <dbl>,
#   DREB <dbl>, REB <dbl>, AST <dbl>, STL <dbl>, BLK <dbl>, TOV <dbl>,
#   TARGET_5Yrs <dbl>
```

```
GP              MIN              PTS              FGM
Min.   :11.00   Min.   : 3.10   Min.   : 0.700   Min.   : 0.300
1st Qu.:47.00   1st Qu.:10.88   1st Qu.: 3.700   1st Qu.: 1.400
```

```
Median :63.00   Median :16.10   Median : 5.550   Median : 2.100
Mean   :60.41   Mean   :17.62   Mean   : 6.801   Mean   : 2.629
3rd Qu.:77.00   3rd Qu.:22.90   3rd Qu.: 8.800   3rd Qu.: 3.400
Max.   :82.00   Max.   :40.90   Max.   :28.200   Max.   :10.200

FGA              FG%            3P Made            3PA
Min.   : 0.800   Min.   :23.80   Min.   :0.0000   Min.   :0.0000
1st Qu.: 3.300   1st Qu.:40.20   1st Qu.:0.0000   1st Qu.:0.0000
Median : 4.800   Median :44.10   Median :0.1000   Median :0.3000
Mean   : 5.885   Mean   :44.17   Mean   :0.2476   Mean   :0.7792
3rd Qu.: 7.500   3rd Qu.:47.90   3rd Qu.:0.4000   3rd Qu.:1.2000
Max.   :19.800   Max.   :73.70   Max.   :2.3000   Max.   :6.5000

3P%              FTM             FTA              FT%
Min.   :  0.00   Min.   :0.000   Min.   : 0.000   Min.   :  0.00
1st Qu.:  0.00   1st Qu.:0.600   1st Qu.: 0.900   1st Qu.: 64.70
Median : 22.40   Median :1.000   Median : 1.500   Median : 71.25
Mean   : 19.31   Mean   :1.298   Mean   : 1.822   Mean   : 70.30
3rd Qu.: 32.50   3rd Qu.:1.600   3rd Qu.: 2.300   3rd Qu.: 77.60
Max.   :100.00   Max.   :7.700   Max.   :10.200   Max.   :100.00
NA's   :11
OREB            DREB            REB              AST
Min.   :0.000   Min.   :0.200   Min.   : 0.300   Min.   : 0.000
1st Qu.:0.400   1st Qu.:1.000   1st Qu.: 1.500   1st Qu.: 0.600
Median :0.800   Median :1.700   Median : 2.500   Median : 1.100
Mean   :1.009   Mean   :2.026   Mean   : 3.034   Mean   : 1.551
3rd Qu.:1.400   3rd Qu.:2.600   3rd Qu.: 4.000   3rd Qu.: 2.000
Max.   :5.300   Max.   :9.600   Max.   :13.900   Max.   :10.600

STL              BLK              TOV         TARGET_5Yrs
Min.   :0.0000   Min.   :0.0000   Min.   :0.100   0:509
1st Qu.:0.3000   1st Qu.:0.1000   1st Qu.:0.700   1:831
Median :0.5000   Median :0.2000   Median :1.000
Mean   :0.6185   Mean   :0.3686   Mean   :1.194
3rd Qu.:0.8000   3rd Qu.:0.5000   3rd Qu.:1.500
Max.   :2.5000   Max.   :3.9000   Max.   :4.400
```

```r
nbadata$`3P%`[is.na(nbadata$`3P%`)] <- 0
colnames(nbadata) <- c("GamePlay","MinPlay","Points","FGMade","FGAttempt", "FGSuccess",
"TPMade","TPAttempt","TPSuccess","FTMade","FTAttempt","FTSuccess",
"OffRebound","DefRebound","TotalRebound","Assist","Steal","Block","Turnover",
"TARGET_5Yrs")        # changing column names for easier interpretation

# computing the correlation plot
corr <- cor(nbadata[, -20])
corrplot::corrplot(corr)

# Performing PCA
pca <- prcomp(nbadata[, -20])
library(ggplot2)

pca.var <- pca$sdev^2

df <- data.frame(index = 1:19, Cumprop = cumsum(pca.var)/sum(pca.var))
ggplot(df, aes(x = index, y = Cumprop)) + geom_point() + geom_line() +
    xlab("Principal Component") + ylab("Cumulative Proportion of Variation explained")
```

```r
cols <- ifelse(nbadata$TARGET_5Yrs == "0", rgb(1, 0, 0, 0.5), rgb(0, 0, 1, 0.5))
pairs(pca$x[, 1:4], col = cols, pch = 20)

# Creation of training and testing set
set.seed(1911) # my roll number as a seed for reproducibility
trainIndex <- sample(nrow(nbadata), 0.75 * nrow(nbadata))
traindata <- nbadata[trainIndex, ]
testdata <- nbadata[-trainIndex, ]
table(traindata$TARGET_5Yrs)
table(testdata$TARGET_5Yrs)

# An utility function to get accuracy and confusion matrix
getResults <- function(preds, true_labels) {
         tab <- table(preds, true_labels)
         acc <- sum(diag(tab))/sum(tab)
        return(list("Confusion Matrix" = tab, "Accuracy" = acc))
}
```

```r
# performing LDA
library(MASS)
fit.lda <- lda(TARGET_5Yrs ~ ., data = traindata)
fit.lda
preds <- predict(fit.lda)
getResults(preds$class, traindata$TARGET_5Yrs)
preds <- predict(fit.lda, newdata = testdata)
getResults(preds$class, testdata$TARGET_5Yrs)
```

```
Call:
lda(TARGET_5Yrs ~ ., data = traindata)

Prior probabilities of groups:
0         1
0.3681592 0.6318408

Group means:
GamePlay  MinPlay   Points   FGMade FGAttempt FGSuccess    TPMade
0 51.52162 14.09703 4.986486 1.924595  4.499189  42.37054 0.2297297
1 66.01417 20.04850 8.117165 3.147087  6.905039  45.46409 0.2645669
TPAttempt TPSuccess    FTMade FTAttempt FTSuccess OffRebound DefRebound
0 0.7548649  19.17595 0.9110811  1.310541  68.35027  0.7113514   1.542973
1 0.8124409  19.11559 1.5612598  2.161732  71.74315  1.2075591   2.370866
TotalRebound   Assist     Steal     Block  Turnover
0     2.253514 1.201351 0.4862162 0.2486486 0.9262162
1     3.578110 1.792441 0.7031496 0.4329134 1.3788976

Coefficients of linear discriminants:
LD1
GamePlay      0.0342093155
MinPlay      -0.0213688200
Points       -0.4089458026
FGMade        0.0643446173
FGAttempt     0.4133374765
FGSuccess     0.0664504699
TPMade        3.2198175925
TPAttempt    -1.0358430463
TPSuccess     0.0005456628
```

```
FTMade        0.4682341358
FTAttempt    -0.0301922875
FTSuccess     0.0238139973
OffRebound    1.2406807679
DefRebound    0.2921629002
TotalRebound -0.5239048104
Assist        0.1448387323
Steal         0.1023330528
Block         0.4119611377
Turnover     -0.0389136136

# Training Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0 195  87
1 175 548

$Accuracy
[1] 0.7393035

# Testing Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0  68  38
1  71 158

$Accuracy
[1] 0.6746269
```

```r
# Performing QDA
fit.qda <- qda(TARGET_5Yrs ~ ., data = traindata)
fit.qda
preds <- predict(fit.qda)
getResults(preds$class, traindata$TARGET_5Yrs)
preds <- predict(fit.qda, newdata = testdata)
getResults(preds$class, testdata$TARGET_5Yrs)
```

```
Call:
qda(TARGET_5Yrs ~ ., data = traindata)

Prior probabilities of groups:
0         1
0.3681592 0.6318408

Group means:
GamePlay  MinPlay   Points   FGMade FGAttempt FGSuccess    TPMade
0 51.52162 14.09703 4.986486 1.924595  4.499189  42.37054 0.2297297
1 66.01417 20.04850 8.117165 3.147087  6.905039  45.46409 0.2645669
TPAttempt TPSuccess    FTMade FTAttempt FTSuccess OffRebound DefRebound
0 0.7548649  19.17595 0.9110811  1.310541  68.35027  0.7113514   1.542973
1 0.8124409  19.11559 1.5612598  2.161732  71.74315  1.2075591   2.370866
TotalRebound   Assist     Steal     Block  Turnover
0     2.253514 1.201351 0.4862162 0.2486486 0.9262162
1     3.578110 1.792441 0.7031496 0.4329134 1.3788976
```

```
# Training Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0 321 234
1  49 401

$Accuracy
[1] 0.718408

# Testing Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0 105  83
1  34 113

$Accuracy
[1] 0.6507463
```

```r
# Performing Classification tree
library(tree)
fit.tree <- tree::tree(TARGET_5Yrs ~ ., data = traindata)
summary(fit.tree)

plot(fit.tree) # plot decision tree
text(fit.tree, pretty = 0)


preds <- predict(fit.tree, type = "class")
getResults(preds, traindata$TARGET_5Yrs)
preds <- predict(fit.tree, newdata = testdata, type = "class")
getResults(preds, testdata$TARGET_5Yrs)
```

```
Classification tree:
tree::tree(formula = TARGET_5Yrs ~ ., data = traindata)
Variables actually used in tree construction:
[1] "GamePlay"   "OffRebound" "FGMade"     "TPSuccess"
Number of terminal nodes:  7
Residual mean deviance:  1.085 = 1083 / 998
Misclassification error rate: 0.2726 = 274 / 1005

# Training Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0 188  92
1 182 543

$Accuracy
[1] 0.7273632

# Testing Set Metrics
$`Confusion Matrix`
```

```
true_labels
preds   0   1
0  62  35
1  77 161

$Accuracy
[1] 0.6656716
```

```r
# Using SVM
library(e1071)
fit.svm <- svm(TARGET_5Yrs ~ ., data = traindata, kernel = "radial")
preds <- predict(fit.svm)
getResults(preds, traindata$TARGET_5Yrs)
preds <- predict(fit.svm, newdata = testdata)
getResults(preds, testdata$TARGET_5Yrs)
```

```
# Training Data Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0 207  88
1 163 547

$Accuracy
[1] 0.7502488

# Testing Set Metrics
$`Confusion Matrix`
true_labels
preds   0   1
0  88  40
1  51 156

$Accuracy
[1] 0.728358
```