NBA Shot Prediction

By: Christian St. Louis

SN: 10089832

For: Dr. Takahara in STAT 499

Introduction

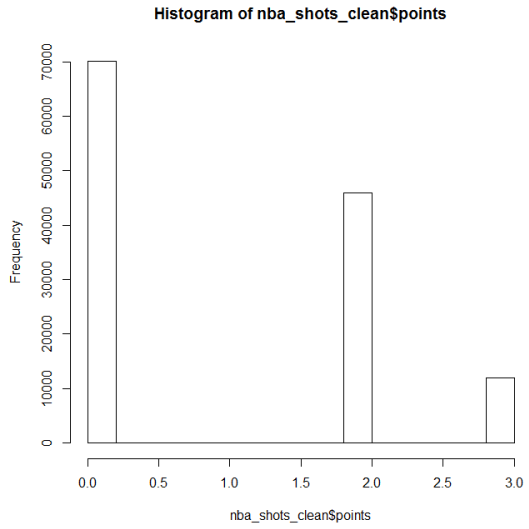
The goal of this study was to predict whether a field goal will be made or missed in the NBA. The data used was all of the shots taken (and other shot related data) during the 2015-2016 season.

Data Cleaning/Analysis

I at first decided to include the dependent variable as FGM which is field goals made and has two possible values, a 0 which counts as a shot missed and a 1 which counts as a shot made. I then included the following predictor variables in my first model: location, shot number, closest defender, defender distance, points, player name, dribbles, quarter, touch time. I quickly realized that there were issues with a lot of these variables and decided to remove a lot of them and change the types of the ones that remained so that they were compatible with the models I used. The details of this data cleaning and variable reduction is in the appendix of this paper.

I then went on to analyze all the important variables and get a handle on the data. I was working with 128069 shots recorded from the 2016-2017 season and so this was the number of rows in my data frame.

The points variable consisted of three different entries, either a 0, 2 or a 3. This was for a shot missed, a 2 point field goal made or a 3 point field goal made. Here is a histogram of the shots taken:



Here we can see that 55% of shots taken were missed, 36% of shots taken were made 2 point field goals and 9% of shots taken were made 3 point field goals. I then went on to create a field goal percentage for the NBA that season which was 45.3%. This means that 45.3% of shots taken during the 2015-2016 NBA season went in.

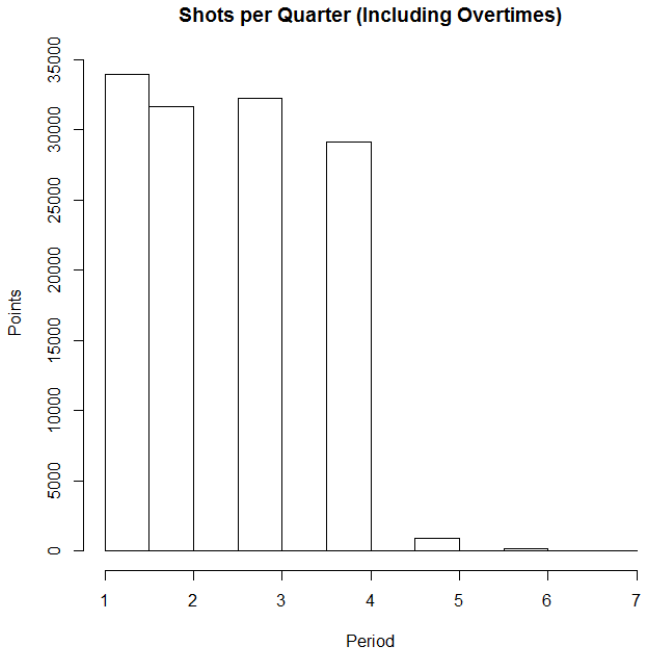
Here is a table that describes a lot of the data I was looking at and is interesting for this project:

Number of NBA Games Played	904
Number of Players that took at least 1 shot	281
Average distance of defenders from shooter	4.123 (feet)
Average amount of dribbles taken before shot	2.02

When looking at the shot clock data I found that were 5567 NAs in this column (they also happened to be the only NAs in the entire data frame). I assumed that these NAs meant that the shot clock had been turned off because there was under 24 seconds (length of the shot clock) left in the quarter when the possession that was being recorded began. I set these NAs equal to zero so that I could work with the data a bit more easily. In hindsight, it would have been better to replace these NAs with the time on the game clock, it would have given a much more accurate representation of how many seconds were left in the quarter when the shot was taken.

I then looked at the amount of shots taken per quarter (where 5, 6, 7 were overtimes). The results were as follows. I descending order from triple overtime to 1st quarter.

7	6	5	4	3	2	1
43	168	912	29123	31651	32211	33961



This is interesting because it tells us that as the game goes on, less shots are taken.

After this I considered players with most misses, twos made and threes made. James Harden ended up missing the most shots with 580 shots missed. Nikola Vucevic made 478 two-point field goals and Steph Curry set the NBA record with 190 three-point field goals made. This makes complete sense, as James Harden takes the most shots in the NBA. Nikola Vucevic is a 7 foot, 260-pound center and so most of his shots are within 5 feet of the rim and Steph Curry is self-explanatory.

I then looked at the most "clutch" player in the NBA for the 2015-2016 season. I based this off the most field goals made in the fourth quarter (2 and 3 pointers). Jamal Crawford won this title as he made 114 shots in the fourth quarter for the Los Angeles Clippers. Now this could be a bit incorrect because this includes games where the Clippers were winning by a lot. In the future, I would only look at games were the point difference at the end of the game was very small, this would indicate a competitive game where the shots made would be considered a lot more "clutch".

Statistical Modelling of the Data

After fitting models with several of the predictor variables I listed above I decided upon 6 predictor variables that gave me the best results. These 6 predictor variables were shot number, defender distance, dribbles, shot clock, quarter and touch time.

I looked at Logistic Regression first. The results of the prediction were summarized in the following confusion matrix:

FALSE	TRUE
17752	3676
13766	4211

This means that the logistic regression model predicted 53.39% of the shots made correctly.

Linear Regression Model:

FALSE	TRUE

17793	3635
13807	4170

This gave a shot prediction accuracy of 53.43%. This was slightly better than the logistic regression model.

KNN prediction model (with k=10) :

FALSE	TRUE
13534	7894
10558	7419

This gave a shot prediction accuracy of 48.45%. I found that as I increased k (until I got to 10) that the prediction accuracy increased as well.

Ridge Regression Model: (using the train data for prediction comparison) – lambda chosen by cv

FALSE	TRUE
41126	7610
30686	9242

This gave a shot prediction accuracy of 54.84%.

Lasso Model: (with lambda chose by cv)

FALSE	TRUE
41021	7715
30608	9320

The Lasso Model gave a shot prediction accuracy of 54.71%.

Conclusion

Originally I included location in my model which consisted of a column of “A” and “H” for home and away. I coded them as 0s and 1s so they would work with the knn function I was using (which did not work with strings). However, after including location and excluding it I found that my prediction percentage went up by approximately 2% across all of my models (linear, logistic, knn, ridge, ridge, lasso). So I decided to proceed with a model that included 6 predictor variables instead of 7 because this optimized my predictions of field goals made (shot being made or missed). For example for my linear regression model when I included location, the prediction of FGM (field goals made) went from 0.514061 to 0.5342729. It turns out that with my 4 main models with the 7 variables I chose as predictors, that the ridge regression model gave the best prediction with 54.7% of shots being predicted correctly. The model had the following variables:

```
FGM ~ shot_number + defender_distance + dribbles + shot_clock + quarter + touch_time.
```

Improvements

If I were to continue with this project I would definitely look to adjust the variables, perhaps include a few more predictor variables if possible. I would also look into more types of advanced regression models, such as Random Forests, XGBoost, Neural Networks, etc. as the consensus of shot prediction models that are online is that these types of models give the best prediction results.

References

DataCamp Website (https://www.datacamp.com/home)

Elements of Statistical Learning by

Github (https://github.com/hwchase17/sportvu/blob/master/SportVu_1.ipynb)

Kaggle (https://www.kaggle.com/dansbecker/nba-shot-logs)

StackOverFlow (http://stackoverflow.com/questions)

Appendix

```
# STAT 499 Final Project
# loading in the data
nba_shots <- read.csv("shot_logs.csv")
head(nba_shots)
tail(nba_shots)
class(nba_shots)
str(nba_shots)

# loading the required packages
install.packages("class")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("data.table")
install.packages("caTools")
install.packages("glmnet")
install.packages("caret")
install.packages("randomForest")

library(class)
library(dplyr)
library(ggplot2)
library(data.table)
library("caTools")
library(glmnet)
library(caret)
library(randomForest)

# exploring the raw data
glimpse(nba_shots)
summary(nba_shots)

# cleaning up the data
# also analyzing all of the data we have and cleaning them up before predicting
colnames(nba_shots)

# creating a new data frame with the columns I chose as predictor variables
nba_shots_clean <- data.frame("game_id" = nba_shots$GAME_ID, "location" = nba_shots$LOCATION,
                          "shot_number" = nba_shots$SHOT_NUMBER,
                          "closest_defender" = nba_shots$CLOSEST_DEFENDER,
                          "defender_distance" = nba_shots$CLOSE_DEF_DIST, "points" = nba_shots$PTS,
                          "player_name" = nba_shots$player_name, "dribbles" = nba_shots$DRIBBLES,
                          "shot_clock" = nba_shots$SHOT_CLOCK, "quarter" = nba_shots$PERIOD,
                          "touch_time" = nba_shots$TOUCH_TIME, "game_result" = nba_shots$W
                          , "FGM" = nba_shots$FGM)

head(nba_shots_clean)
str(nba_shots_clean)
colnames(nba_shots_clean)

# checking to see if there are any duplicate entries in our new data frame
cat("The number of duplicated rows are", nrow(nba_shots_clean) - nrow(unique(nba_shots_clean))) #
no duplicates

# the number of shots I am working with is as follows
nrow(nba_shots_clean) # 128069 shots taken during the season

# checking to see what type of points were recorded
unique(nba_shots_clean$points) # this tells us that the points recorded were either a 2, 3 or 0
hist(nba_shots_clean$points) # histogram of the types of points
# percentage of each type of shot
sum(nba_shots_clean$points==0)/sum(nba_shots_clean$points) # 55 % of shots taken were missed
sum(nba_shots_clean$points==2)/sum(nba_shots_clean$points) # 36 % of shots taken were made 2s
sum(nba_shots_clean$points==3)/sum(nba_shots_clean$points) # 9 % of shots taken were made 3s
# creating a field goal percentage
field_goal_percentage <- (sum(nba_shots_clean$points==2) +
sum(nba_shots_clean$points==3))/sum(nba_shots_clean$points)
# 45.3 % shots taken were made
```

```r
###
# I may need to switch the location column to type int
# I will code "A" as 0 and "H" as 1
# looking at the field goal percentage at home vs away
# this is field_goal_percentage grouped by A and H
###

# 1 is a make and 0 is a miss
unique(nba_shots_clean$FGM)

# Checking the amount of games I am dealing with
length(unique(nba_shots_clean$game_id)) # 904 NBA games played

# Checking the amount of "closest defenders"
# not sure if this makes sense...
# it doesn't really matter, the distance matters more
length(unique(nba_shots_clean$closest_defender)) # 473 unique closest defenders


# Checking the amount of players that took shots
length(unique(nba_shots_clean$player_id)) # 281 players took shots


# Now looking at the average distance of the defenders from the shot taker
mean(nba_shots_clean$defender_distance) # 4.123

# the average amount of dribbles taken before a shot was put up by that player
mean(nba_shots_clean$dribbles) # 2.02

# average shot clock time when the shot was taken
mean(nba_shots_clean$shot_clock) # NA
# this gave NA return which means that there are NAs in this column that we need to clean up
# if the shot clock was NA I assume that this means it was the end of a quarter and the shot
clock was off.
# For now I'm going to just set all of these NAs equal to zero, so all zeros mean it is the end
of a quarter
# checking the amount of NAs
last_shots <- nba_shots_clean[is.na(nba_shots_clean$shot_clock),]
nrow(last_shots) # this tells me there is 5567 shots taken when the shot clock was turned off at
the end of a quarter
# setting these NAs equal to zero
nba_shots_clean[is.na(nba_shots_clean)] <- 0
# checking to see if it worked
nrow(nba_shots_clean[is.na(nba_shots_clean$shot_clock),]) # it worked as there are 0 NAs in the
dataframe now


# Check to see the period column is clean
unique(nba_shots_clean$quarter)
# there are 7 quarters here, this means that 5, 6, 7 are single OT, double OT and triple OT
# let's see how many shots in each quarter was recorded
sort(table(nba_shots_clean$quarter))
# 43 shots intriple overtime, 168 shots in double overtime and 912 shots in overtime
# showing a histogram of these numbers (including first 4 quarters)
hist(nba_shots_clean$quarter, main = "Shots per Quarter (Including Overtimes)", ylab = "Points",
xlab = "Period")
# this tells us that most shots were taken in the following order
# 1st quarter, 3rd quarter, 2nd quarter and least shots taken in the 4th quarter
# overtimes intuitively go single, double, triple


# need to convert a few of my predictor variables to factors instead of type numeric
str(nba_shots_clean)
nba_shots_clean$quarter <- as.factor(nba_shots_clean$quarter)
nba_shots_clean$points <- as.factor(nba_shots_clean$points)

# number of wins and losses
unique(nba_shots_clean$game_result)

# Looking at the difference in wins vs losses in home vs away teams
```

```r
# Looking at the difference for field goal percentage in home and away teams

# The amount of points per player (0,2,3)
length(unique(nba_shots_clean$player_name)) # this says there are 281 players that took a shot
(consistent)
points_per_player <- table(nba_shots_clean$player_name, nba_shots_clean$points)
# I'm going to export this table to excel and then load it back in as a dataframe with 3 columns
- easier to deal with
write.csv(points_per_player, "points_per_player.csv")
points_per_player_new <- read.csv("points_per_player.csv")
# renaming columns
colnames(points_per_player_new)[1] <- "Player"
colnames(points_per_player_new)[2] <- "Missed"
colnames(points_per_player_new)[3] <- "Twos"
colnames(points_per_player_new)[4] <- "Threes"
# checking type
str(points_per_player_new) #data.frame

# Player with most misses, twos, threes, and overall makes
max(points_per_player_new$Missed, na.rm=TRUE) # 580 misses
points_per_player_new[which.max(points_per_player_new$Missed),] # James Harden Missed 580 shots
points_per_player_new[which.max(points_per_player_new$Twos),] # Nikola Vucevic made 478 twos
points_per_player_new[which.max(points_per_player_new$Threes),]# Steph Curry made 190 threes


# Looking at the best players per quarter
# this is just based off of most FGM per quarter
fgm_per_quarter <- table(nba_shots_clean$player_name ,nba_shots_clean$quarter,
nba_shots_clean$FGM)
write.csv(fgm_per_quarter, "fgm_per_quarter.csv")
fgm_per_quarter_new <- read.csv("fgm_per_quarter.csv")
# renaming the columns
colnames(fgm_per_quarter_new)[1] <- "Nothing"
colnames(fgm_per_quarter_new)[2] <- "Player"
colnames(fgm_per_quarter_new)[3] <- "Period"
colnames(fgm_per_quarter_new)[4] <- "FGM"
colnames(fgm_per_quarter_new)[5] <- "Frequency"

# now looking at the player with most fgm in fourth quarter (clutch player)
# when Period = 4, FGM = 1, what is the row with the max Frequency?
Fourth <- fgm_per_quarter_new[fgm_per_quarter_new$Period==4 & fgm_per_quarter_new$FGM==1,]
Fourth[which.max(Fourth$Frequency),] # Jamal Crawford hit 114 shots in the fourth quarter (most
in NBA)

# Best defender?


##### MODEL TRAINING
# create a test and train set
#set.seed(0)
# need to split A into 0 and H into 1 for location
nba_shots_clean$location <- as.character(nba_shots_clean$location)
nba_shots_clean$location[nba_shots_clean$location == "A"] <- 0
nba_shots_clean$location[nba_shots_clean$location == "H"] <- 1
nba_shots_clean$location <- as.factor(nba_shots_clean$location)
# didn't end up using this in my final model

split = sample.split(nba_shots_clean, SplitRatio=0.75)
nbaTrain = subset(nba_shots_clean, split==TRUE)
nbaTest = subset(nba_shots_clean, split==FALSE)



# logistic regression
# should probably code location as 0 for A and 1 for H
nbaLogitModel <- glm(FGM ~ shot_number + defender_distance + dribbles
                + shot_clock + quarter + touch_time, data=nbaTrain, family="binomial", na.action =
na.omit)

nbaPredict = predict(nbaLogitModel, newdata=nbaTest, type="response")
```

```
cm = table(nbaTest$FGM, nbaPredict > 0.5)
print(cm)
# FALSE  TRUE
# 0 17752  3676
# 1 13766  4211
cm[2,2] / (cm[1,2]+cm[2,2]) # [1] 0.5339166


# linear regression
# should probably code location as 0 for A and 1 for H
nbaLinearModel <- lm(FGM ~ shot_number + defender_distance + dribbles
                     + shot_clock + quarter + touch_time, data=nbaTrain, na.action = na.omit)

nbaPredict1 <- predict(nbaLinearModel, newdata=nbaTest)
cm1 = table(nbaTest$FGM, nbaPredict1 > 0.5)
print(cm1)
# FALSE  TRUE
# 0 17793  3635
# 1 13807  4170
cm1[2,2] / (cm1[1,2]+cm1[2,2]) # [1] 0.514061 to [1] [1] 0.5342729

# best predictor so far is linear just barely

# K Nearest Neighbours
# need to change location to numeric value
# decided to remove location from the model
# may have to change quarter to type numeric ?
vars = c("shot_number", "defender_distance", "dribbles", "shot_clock", "quarter", "touch_time",
"FGM")
nbaSub = nba_shots_clean[,vars]
split1 = sample.split(nbaSub$FGM, SplitRatio=0.75)
nbaTrain1 = subset(nbaSub, split==TRUE)
nbaTest1 = subset(nbaSub, split==FALSE)
nbaKnnPred <- knn(train = nbaTrain1[,-7], test = nbaTest1[-7], cl = nbaTrain1[,7], k=10)
# need to change from factor to numeric in order to compare with the original F
str(nbaKnnPred)
nbaKnnPred <- as.numeric(as.character(nbaKnnPred))
# making sure the lengths are the same
length(nbaKnnPred)
length(nbaTest1$FGM)
# checking out this knn function
nbaKnnPred <- knn3Train(train = nbaTrain1[,-7], test = nbaTest1[-7], cl = nbaTrain1[,7], k=10)
# comparing knn pred with the original fgm
cm2 = table(nbaTest1$FGM, nbaKnnPred > 0.5)
print(cm2)
# FALSE  TRUE
# 0 13534  7894
# 1 10558  7419
cm2[2,2] / (cm2[1,2]+cm2[2,2]) # 0.4844903

# as we can see this model did not predict very well compared to logistic and linear regression


# ridge, lasso
str(nbaTrain1)
# the issue is the factor for quarter
# turning factor into numeric while maintain integrity
nbaTrain1$quarter <- as.numeric(as.character(nbaTrain1$quarter))
glm.cv.ridge = cv.glmnet(as.matrix(nbaTrain1[,-7]), as.matrix(nbaTrain1[,7]), alpha = 0)
glm.cv.lasso = cv.glmnet(as.matrix(nbaTrain1[,-7]), as.matrix(nbaTrain1[,7]), alpha = 1)

# use the lamdba that minimizes the error
penalty.ridge = glm.cv.ridge$lambda.min
penalty.lasso = glm.cv.lasso$lambda.min


# train our model using chosen lambdas
glm.ridge <- glmnet(x = as.matrix(nbaTrain1[,-7]), y = nbaTrain1[,7], alpha = 0, lambda =
penalty.ridge)
glm.lasso <- glmnet(x = as.matrix(nbaTrain1[,-7]), y = nbaTrain1[,7], alpha = 1, lambda =
penalty.lasso)
```

```r
y_pred.ridge <- as.numeric(predict(glm.ridge, as.matrix(nbaTrain1[,-7])))
y_pred.lasso <- as.numeric(predict(glm.lasso, as.matrix(nbaTrain1[,-7])))

# checking lengths
length(nbaTrain1[,7])
length(y_pred.ridge)
length(y_pred.lasso)
# lengths all match up

# comparing the predictions to the original y values (FGM)
# ridge
cm3 = table(nbaTrain1$FGM, y_pred.ridge > 0.5)
print(cm3)
# FALSE  TRUE
# 0 41126  7610
# 1 30686  9242
cm3[2,2] / (cm3[1,2]+cm3[2,2]) # [1] 0.5484216

# lasso
cm4 = table(nbaTrain1$FGM, y_pred.lasso > 0.5)
print(cm4)
# FALSE  TRUE
# 0 41021  7715
# 1 30608  9320
cm4[2,2] / (cm4[1,2]+cm4[2,2]) # [1] 0.5471089

# here we can see from the results of the confusion matrices that the ridge regression preforms
slightly better
# then lasso
# this means that ridge is our best model thus far


# because the knn model did not preform well I won't look at the lda and qda either

# in conclusion the model that preformed the best was the ridge regression model which predicted
# 54.84 % of the shots correctly


# Now going a bit further and looking at random forest
nbaRandomForest <- randomForest(FGM ~ ., data=nbaTrain1, nodesize=25, ntree=200) # very long run
time
nbaPredictRF = predict(nbaRandomForest, newdata=nbaTest1)
```