2022-2023 NBA Basketball Player Statistics Analysis

Advanced Data Analysis
DSC 324/424

Team members

Alaa Al Zannan
Victor Oche
Anirudh Mehotra
Sachit Patel

Summary

The NBA is one of the most popular professional basketball leagues in the world. It consists of 30 teams divided into the Eastern conference and the Western conference. Players in the NBA are among the biggest superstars in the world, so a lot is required of them in terms of performance on the court and off the court. In this project, we analyze the 2022-2023 NBA Player Stats Dataset for the regular season, to gain insight on the dataset, recognize factors that affect the total point scored in an NBA/Basketball game and see how players can be classified based on their performance statistics. The overarching goal is to help team managers and stakeholders get perhaps 'a working model' that can be applied to get optimal results from their players and win games.

To perform this task, we applied different advanced data analysis techniques on our dataset, towing two major research paths: (1) to identify the hidden factors that affect a player's points per game, and (2) to determine the relationship between player's statistics and their position, helping us easily classify a player into a certain position based on their stats.

Our dataset consisted originally of 30 columns that represent basketball player statistics such as age, field goals, 2 points field goals, rebounds, assets, block, etc. And 679 observations (see data dictionary in the next section). We applied 3 major advanced data analysis techniques to achieve our research goals. These are: A regularized regression (Lasso and Relaxed Lasso), Factor Analysis (Principal and Common Factor Analyses), and linear discriminate analysis. All-subset regression analysis was used to verify the models and logistic regression was used alongside the linear discriminate analysis to aid further classification of position variables.

Hidden factors not immediately intuitive to the thinker or seasoned professionals such as the effect of the counteraction between the Shooting Guard and the Center position player, the synergies between other position pairs with the Shooting Guard, such as the Point-Guard and the Shooting guard in terms of assists to the Shooting guard, the Power-Forward and the Shooting-Guard and the Small-Forward and the shooting guard, are important factors that affect points per game. Also, defensive rebounds are an especially significant factor, affecting points per game. Models were built to show this effect.

Furthermore, while classifying player positions based on their statistics, we find that the most important statistics able to distinguish between the point guard and shooting guard positions are defensive rebounds, assists and points per game, while major distinguishers between the Small-Forward and Power-forward positions are age, 3-point field goals, defensive rebounds and assists. We determined that the most important player statistics required to classify the Center, Forward and Guard positions are points per game, steals, effective field goal percentage, field goal attempts, assists, 3 points field goal attempts, turnovers, field goal, defensive rebound, offensive-rebounds, and blocks.

Technical Summary

Exploratory analysis

Our dataset originally included 30 variables and 679 observations. Three variables being categorical and the remaining numeric. Exploring the dataset, we found that there were no missing values, however, there were a couple of duplicated rows. The duplicated rows were for the players who participated in different teams during the same season, and each one of such players had an extra row that calculated the total of their statistics. Therefore, we decided to drop those rows. In total, we dropped 70 duplicated rows.

The original dataset consists of the following variables and their definitions:

Rk: Rank	Player: Player's name	Pos: Player Position	Age: Player's age	Tm: Team
G: Games played	GS: Games started	MP: Minutes played per game	FG: Field goals-The number of field goals that a player has made. This includes both 2 pointers and 3 pointers	FGA: (The number of field goals that a player or team has attempted. This includes both 2 pointers and 3 pointers.)
(FG.): (Field goal percentage- the percentage of field goal attempts that a player makes, given by (FGM)/(FGA))	(X3P): 3-point field goals per game (The number of 3-point field goals that a player or team has made)	(X3PA): (The number of 3-point field goals that a player or team has attempted)	(X3P.): 3-point field goal percentage (The percentage of 3-point field goal attempts that a player makes)	(X2P): 2-point field goals per game
X2PA): 2-point field goal attempts per game	(X2p.): 2-point field goal percentage	(eFG.): (Field goal percentage adjusted for made 3-point field goals being 1.5 times more valuable than made 2-point field goals. ((FGM + (0.5 * 3PM)) / FGA)	FT: Free throws per game	FTA: Free throw attempts -the number of free throws that a player or team has attempted.
(FT.): Free throw percentage- the percentage of free throw attempts that a player or team has made (FTM)/(FTA)	ORB: Offensive rebounds- The number of rebounds a player or team has collected while they were on offense	DRB: (Defensive rebounds-the number of rebounds a player or team has collected while they were on defense.	TRB: Total rebounds- a rebound occurs when a player recovers the ball after a missed shot. This statistic is the number of total rebounds a player or team has collected on either offense or defense.	AST: (The number of assists passes that lead directly to a made basket -- by a player)
STL: (Steals- Number of times a defensive player or team takes the ball from a player on offense, causing a turnover.)	BLK: (Blocks-A block occurs when an offensive player attempts a shot, and the defense player tips the ball, blocking their chance to score)	TOV: (Turnovers- A turnover occurs when the player or team on offense loses the ball to the defense.) PF: Personal fouls per game	PF: Personal fouls per game	PTS: The number of points scored per game

While cleaning the dataset, examining correlations between variables, and making initial ordinary least squares regression, we discovered that we were dealing with a unique type of dataset- a perfectly multicollinear dataset. As seen from the correlation plots and indeed the data dictionary, there existed aliased variables and coefficients that were linear combinations of other variables present in the dataset. FG and eFG. are examples of such variables that were linear combinations of multiple variables (X2P, X2PA, X3P, X3PA). TRB is another example, being made up of ORB and DRB. These variables and others present alongside their linear combinations caused heavy aliasing and perfect multicollinearity. Fixing these required innovative solutions after several research attempts. We split the data into two data sets: Player stats data set included variables that show the individual player's contribution to the outcome of a game. This dataset included variables that show actions that a particular player will perform towards the outcome of a basketball game along with their positions. It included:

```
> head(nba_plyrstat)
  Pos Age  G GS   MP X3P X3PA X2P X2PA  FT FTA ORB DRB AST STL BLK TOV  PF  PTS
```

The team stats dataset included variables that represents an individual player's contribution to the total team performance. This usually included their percentage input to the total team performance as seen from the data dictionary. It included

```
> head(nba_tmstat)
  Pos Age  G GS   MP  FGA   FG. X3P X3PA  X3P. X2P X2PA  X2P.  FT FTA   FT. ORB DRB AST STL BLK TOV
```

Furthermore, correlation plots were examined to further reduce the team stats and player stats datasets to further fix aliasing by removing one of each pair of variables with the exact correlation profiles. We made sure to remove the more obvious variables of such pairs, leaving the less obvious since our aim was to discover latent factors. The cleaned player stats dataset was used for our first research goal and Team stats data set was used for the second line of research.

Dummy variables were created for the categorical position variable and the datasets were transformed to fix skewness to the best possible degree. (Refer to graphs in appendix).

Regularized Regression:

Introduction:

In the first approach of this project, we are modeling around the PTS predictor, and trying to discover what factors (or latent insights) contribute to higher value of points-per-game. An initial OLS model during the exploratory analysis was made with the dataset. However, it has a concerning issue. The OLS full model is highly susceptible to overfitting. Additionally, with the high amount of correlation shown between predictors within the correlation plot, multicollinearity is a present issue within the model. Refer to individual summaries for a breakdown on multicollinearity. The punishment of regularization was harsh on the overall fit when LASSO was first used, so a relaxed LASSO was opted for in order to ease the regularization punishment. Additionally, a base cross-validated LASSO was utilized before relaxed LASSO. Refer to figures 2.0 and onwards for the specific output.

Relaxed LASSO Attempts & Analysis:

After trying several runs with cross-validated GLMNET, different combinations of predictors and lambda 1se's were given. A specific output was settled on by using the lambda 1se value of 0.04653347 on the training split of the dataset. As for overall statistics, this gave an R^2 of around 93%, with an RMSE of roughly 0.18. Below is the sample plot and variables chosen.

Additionally, the gamma for 1se is 0.00, which indicates that overfitting is not a problem. The coefficients of this fit after LASSO's variable selection were games played (negligible effect, could be removed), 3-point attempts, 2-point attempts, free throws, and defensive rebounds. These discovered outputs help build a groundwork for later factor analyses and discriminant analyses, which other members of the group are doing. My role in this project as it played out was to address multicollinearity and to address the approach of building a model for points per game for individual players.

It took only a little addition of lambda to remove several predictors. Selection was quickly done with an extremely small lambda, drowning noise. 2 points, 3 points, free throws, and defensive rebounds are the mechanisms behind points per game. Keep in mind that we used the attempt variables for two points and 3 points, not the actual points scored. This is probably because the ball is in the hands of their most trusted players to score, or in the positions that score. The combination of predictors themselves are the types of scores that can be made, as well as the most important mechanism to upstart scoring points- defensive rebounds.

Residual Analysis:

A residual analysis was performed on the models to check how the model addresses the various normality assumptions, as well as identifying outliers and influential points. Refer to the appendix for normality plot output (figure 1.3, 1.4, 1.5). In the listed normality QQ plot, notice how the normality QQ plot has a bit of a curve in the beginning, but is mostly linear throughout. The spread of the plot seems to widen in the initial steps of the graph. We can see this same occurrence in the residuals vs fitted plot. The residuals vs fitted plot is mostly scattered around

the line of best fit evenly, but there is a fanning out here as well within the early fitted values of the graph. The spread in data for points is wider for the earlier values of the logarithm of points.

For further analysis, we could look at the influential points and outliers. Refer to the third image in the "Regularized Regression" appendix section for the output of Cook's Distance for various observations. We can see there are three key points with high influential points. They're labeled with 73, 91, and 299 in each of these figures. These are the most prominent outliers, and likely would be removed due to high studentized residuals (greater than 3 or less than -3). These points being removed could help the overall fit of the model, specifically in reducing the fanning in the early portions of the normal plots.

Revisiting the Relaxed Lasso Model:

A couple of improvements can be made to the relaxed lasso model. For one, a small training and testing split was used. Instead of 10% going to the test set, an 80 to 20 split is more balanced. Additionally, the relaxed lasso model found that the games played predictor was not contributing much. Is it possible we can create a strong model without it? After testing output, we are able to get a relaxed lasso model with 4 predictors (games played removed) for the fit of lambda 1se. Refer to the appendix for extended output. There is a slight drop in model fit, and a slight increase in RMSE. But for 1 less predictor, the model is overall more parsimonious. Unfortunately, a model with a test set RMSE higher than training set RMSE doesn't occur with this lambda.1se value.

Model Equation:

Adjusted Model: PTS = 0.3724088 + .3675337(X3PA) + .5287313(X2PA) + 0.2465942(FT) + 0.2774022(DRB)

Regularized Regression Conclusion:

In conclusion, regularized regression was utilized on a slimmed down version of the dataset that consisted solely of player statistics. The initial OLS model was extremely susceptible to overfitting. As such, regularized regression was an approach chosen, as adding some bias could help reduce the variance (bias-variance tradeoff).

The initial LASSO model was extremely punishing with its regularization. As such, a relaxed LASSO model was utilized to relax the penalty of regularization a bit. These created a more parsimonious model at the lambda.1se value with only 4 predictors (2-point attempts, 3-point attempts, free throws, and defensive rebounds). The plot output for relaxed LASSO shows that the lambda.1se model has a gamma value of 0, which means this model addresses the issue of overfitting. The model's RMSE is a bit finicky between the training and testing split, but overall has high predictive power (Dev %). The model ends up resembling a small portion of the larger model presented by principal and common factor analysis, usually accounted for as a factor that is interpreted as "actions in game" alongside defensive rebounds being its own component.

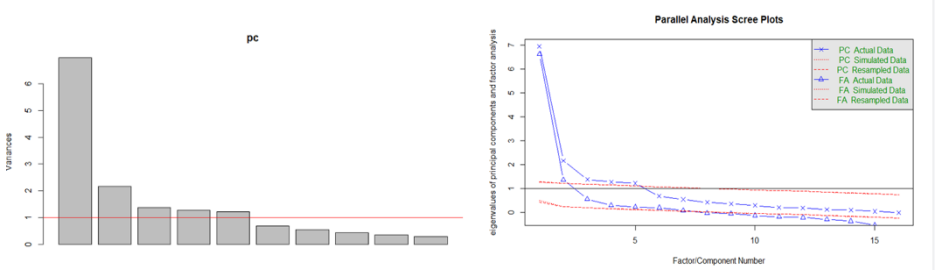
Principal and Common Factor Analysis

The goal of the Factor Analysis was to seek out latent factors that affect points per game in a basketball game. First observation of the correlation plots identified variables from the player stats data sets that had the same correlation profiles. Three points made and 3 points attempts for instance had the same correlation distribution. The same was the case for 2points and 2-point attempts, Offensive rebounds and Blocks had the same correlation profiles as well, amongst others.

Our first approach was to take out the obvious variables that would affect points per game when such variables had the same correlation as not so obvious ones. For example, most people would know that if you want to score more points, just make more three-point and two-point shots. However, we sought to see how attempts at field goal and other variables would latently affect the points per game. We therefore took 3points, 2points and free throws out, when their correlations were noticed to be exact with their attempts. This left us with room to seek latent factors. One PFA and CFA model were eventually chosen to be modeled for PTS after validation with each other and with all subsets' regression out of six factor Analysis conducted. Several factor Analysis were done on the log transformed and untransformed datasets.

Principal Factor Analysis 1(PFA1)

The Kaiser-Meyer-Olkin (KMO) test for factor adequacy gave an overall MSA value of 0.5. Considering that this value is not too far from the accepted range of > 0.5, we went ahead with our factor analysis. The scree plot for initial principal component analysis (PCA) for the first PFA identified 5 factors at the Var=1 criterion and at the knee. Parallel analysis suggested 5 components and 6 factors. Five factors were used. Defensive rebounds (DRB) and Age were found to be their own factors after doing a correlation test. DRB correlated with almost all variables and Age very weakly with few. These were taken out and later added to the scores for regression. Dummy variables were used for the position varibles. This helped us identify latent factors in the position variable.



The summary of the principal factor Analysis indicated a root mean square of residuals (RMSA) of 0.04.

PFA1 Loadings and Factor Model building

Fig 3 : Loadings

```
> print(pf1$loadings,cutoff=.4,sort=T)

Loadings:
       RC1    RC2    RC3    RC5    RC4
G       0.684
GS      0.868
MP      0.963
X2PA    0.892
FTA     0.852
AST     0.816
STL     0.754
TOV     0.875
PF      0.794
X3PA    0.597 -0.599
BLK     0.512  0.648
Pos_C          0.916
Pos_PG                0.929
Pos_PF                       0.974
Pos_SF                              0.951
Pos_SG        -0.437 -0.534 -0.446 -0.550

                 RC1   RC2   RC3   RC5   RC4
SS loadings    6.923 2.075 1.451 1.290 1.272
Proportion Var 0.433 0.130 0.091 0.081 0.079
Cumulative Var 0.433 0.562 0.653 0.734 0.813
```

The loadings of the first PFA show latent factors RC1—RC5. The first factor we named **"actions in game**". It represents 2-point attempts, free throw attempts, assists, steals, turnovers, personal fouls, 3-point attempts and blocks. Indeed, all actions that a player will engage in to score points and win a game. The scores from factor RC2 we named "3pattemps by Shooting guard and Blocks by Center". This factor indicates how the counteraction of 3-points scoring attempts by the SG and blocks by the center will affect the point per game. RC2 aptly named the 'synergy between the Point Guard and Shooting Guard' identified that latent contribution. RC5 named "synergy between the Shooting Guard and Power Forward, initiated by Power Forward" identified that interaction pair that affects points per game. RC4 named "synergy between the Shooting Guard and the Small Forward initiated by the Small Forward" signified that interaction parings between the positions to affect points per game. Varimax factor rotation was used to easily distinguish the factor loadings. The five factors covered a cumulative variance of 81.3%.

Defensive rebounds and Age, variables identified from the correlation tests as their own factors were added to the factor scores and a regression model was built from the combined factors. All-subsets regression was used to validate the factor selection from the OLS model built from the factors. (see section in appendix for regression images).

PTS = 1.738 + 0.08(actions_in_game) - 0.04(3pattemps_by_SG_and_BLK_by_C) - 0.03(synergy_btw_PG_and_SG) + 0.21 (DRB)

Is the model built from the first PFA. It had an R-squared value of 86.9% and and adjusted R-Squared of 86.7%. Because Age was its own factor but was not found significant in the model, a regression test of Age on PTS produced an R-squared of 1.1% indicating that it proved insignificant. Included in the model, it will do little to increase the predictive power of the model. Confirmatory factor analysis from the Lavan package to check goodness of fit unfortunately did not work on the model as the data contained position dummy variables that pointed out important factors in our PFA.

CFA1.

The Position dummy variables were removed to conduct factor analysis in factanal. After romoving DRB and Age which were again their own factors, the scree plots and parallel analysis suggested two factors. Two factors were used. The CFA identified two latent factors named on

the scores as “Actions in game played minus 3point attempts” for Factor 1 and “Actions in games played including three-point attempts minus blocks”. Considering the position variables were removed, these factors made sense when domain knowledge is applied. It showed the counter-interacting effects of the 3points attempts and blocks. Knowing that the shooting guard is tasked with taking shots including 3-point shots and the center being usually the biggest is tasked with making blocks, showed how these factors pointed towards the PFA. The cumulative variance captured on the CFA was about 68.4%.

An all-subsets regression done on the scores after the dummy variables, dependent variable and individual factors of DRB and Age were added back found “actions in games played minus #point attempts”, “Actions in games played including three-point attempts minus blocks,” “Defensive rebounds” and Position center as significant. It captured an R-sqaured of 84.3% and Adj R-squared of 84.2%. Regression tests on Age, and Position Shooting guard to see if they should be included in the model since the all-subsets did not capture them showed that they had very negligible r-squared and adj-squared values and would not really have any significant effect in affecting the predictive power of the model.(See section in appendix for all images of the CFA factor analysis).

The R-squared and Adj- squared value, explicability of the model and correspondence between the all-subsets and OLS regressions models helped us decide which Factor Analysis to pick as our final model. PFA 1 scored highest on all grounds.

Linear Discriminate analysis

In the third approach of this project, we decided to classify basketball players' positions based on the available statistics during the season. Therefore, we applied linear discriminate analysis (LDA) in two different ways along with logistic regression.

In our dataset there are five different positions which are (point guard (PG), shooting guard (SG), small forward (SF), power forward (PF), and center (C)), and each position has the following number of observations respectively 119, 154, 113, 108 and 115. In the initial discriminate analysis, we tried to classify the five different positions, and in the second discriminate analysis we combined the small forward (SF) with power forward (PF) and point guard (PG) with shooting guard (SG) to classify three positions only (center, forward and guard) to reduce classification confusion between positions. Finally, we run the logistic regression for forward positions and guard positions separately to find out the variables that are able to classify both of them clearly. In all discriminate analysis we decided to include all variables except game played (G), game started (GS) and minutes played (MP) and total rebound (TRB) since they are not essential factors that could determine player position during the game comparing to the other variables. Also, we split the dataset randomly to be trained on 70% of the observations and tested on the remaining 30% of the observations.

1st Discriminate analysis:

In this LDA the aim was to classify the five different positions. The separation percentage was 78% for LD1 and 15% for LD2 as represented in table 1. Based on that, we can use the first two components only since they have the most separation percentage comparing to the 3rd and 4th components. To visualize the separation of this analysis and see which variables contribute in this separation, I used histogram and scatterplot as it show on the right. Based on that, I can say that LD1 is able to separate the center position from shooting guard, small forward and power forward with some confusion. LD2 separates the point guard from small forward, power forward and shooting guard. Table 2 shows the most important variables that contribute to classifying those positions.

Finally, by looking at the confusion matrix it appears that training set classification is accurate by 64% while testing set is 45%.

2nd Discriminate analysis:

The second LDA aims to classify the three positions (center, forward and guard) and the separation percentage was 86% for LD2 and 15% for LD2 as represented in Table 1. The histogram and scatterplot show that LD1 is able to separate the center from forward clearly while LD2 does not show any separation.

Table 2 show the important variables that contribute either positively or negatively in this separation. By looking at the confusion matrix it appears that classification accuracy is 77% for training set and 75% for the testing set.

Logistic Regression:

To run the logistic regression, we created two different datasets, one for forward position,

which includes small forward (SF) and power forward (PF), and the other one for guard position which includes point guard (PG) and shooting guard (SG). After running the logistic regression on both sets it appears that the significant variables for forward set are age, 3-point field goal, defensive rebound and assets while the significant variables for guard set are defensive rebound, assets and points per game.

$$\text{Log}\frac{forward=1}{forward=0} = -0.27 + 0.09\,(\text{age}) - 9.43\,(\text{3-point field goal}) + 4.33\,(\text{defensive rebound}) - 2.5\,(\text{assets})$$

$$\text{Log}\frac{guard=1}{guard=0} = -2.41\,(\text{defensive rebound}) + 6.07\,(\text{assets}) - 8.21\,(\text{points per game})$$

To evaluate the fitting of the generated models I used the testing set to see how the model is doing when predicting the player position and the prediction accuracy was 66% for forward set and 76% for guard set as mentioned in table 3.

Conclusion

Using discriminate analysis to classify player positions based on their statistics shows that the most important player statistics required to classify the Center, Forward and Guard positions are points per game, steals, effective field goal percentage, field goal attempts, assists, 3 points field goal attempts, turnovers, field goal, defensive rebound, offensive-rebounds, and blocks. Also, logistic regression shows that the most important statistics to distinguish between the point guard and shooting guard positions are defensive rebounds, assists and points per game, while major distinguishers between the Small-Forward and Power-forward positions are age, 3-point field goals, defensive rebounds and assists.

TABLE 1: Separation Percentage of Each Component

	LDA For 5 Positions	LDA For 3 Positions
DL1	78%	86%
LD2	15%	15%
LD3	0.05%	-
LD4	0.02	

Table 2: The most important variables that contribute in position separation

Contribution	LDA For 5 Positions		LDA For 3 Positions	
	LD1	**LD2**	**LD1**	**LD2**
	eFG%	eFG%	PTS	FG%
	FGA	FG	STL	3P
	STL	3PA	eFG%	FGA
	AST	2PA	FGA	BLK
+	PTS	2P%	AST	PTS
	2P	STL	3PA	AST
		PTS	TOV	TOV
		DRB		FT%

-	FG DRB BLK ORB	FG% FGA 3P 2P AST BLK FT% FTA TOV	FG DRB ORB BLK FG% FTA	eFG% 3PA FG 2PA 2P% DRB ORB STL

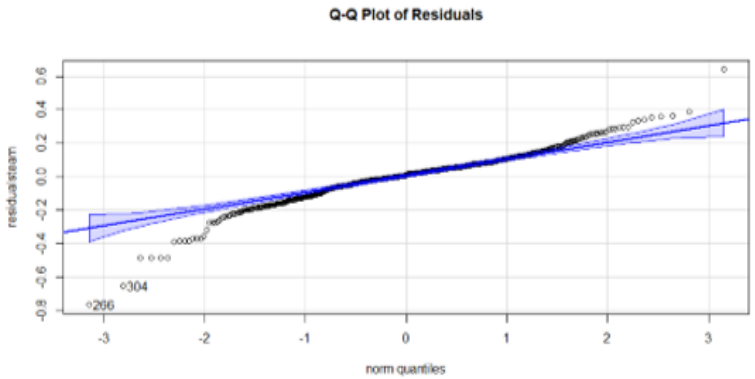
Table 3: Logistic regression models

		Forward Dataset	Guard Dataset
Significant Variables		AGE X3P DRB AST	DRB AST PTS
AIC	**Full Model**	207	220
	Final Model	198.97	194.25
Prediction Accuracy		66%	76%

All Subset Regression

We have used this approach to compare our results with that of Lasso and Factor Analysis. We performed this analysis on both team and player datasets. Our response variable is Points Per Game. So, we used BIC Scale after performing all subset regression on both the player and team statistics dataset and we found out the variables which were most significant were - like for the team dataset: Field Goal Percentage, 3 Points Per Game, 2 Points Assists Per Game, Free Throws Per Game. For the player dataset: Games Played, Game Started, 3 Points Per Game, 2 Points Per Game, Free Throw Attempts Per Game, Defensive Rebounds Per Game, Free Throw Attempts Per Game, Turnovers Per Game, Personal Fouls Per Game.

So, we created a linear model out of these variables for both the datasets and from the model so the equation for the team set is Points Per Game = 0.16 + 1.05(FG.) + 0.71(X3P) + 0.56(X2PA) + 0.168(FT). This indicates that for every unit change in 3 Points Per Game, Points per game will be increased by 0.71 units. The model for the player dataset gave us this equation: Points Per Game: 0.56+ 0.0017(G) - 0.02(GS) + 0.70(X3PA) + 0.74(X2P) + 0.20(FTA) + 0.13(DRB) - 0.12(TOV) + 0.04(PF). This indicates that for every unit increase in 3 Points Assists Per Game, Points Per Game will be increased by 0.70. Similarly, for every unit decrease in Turnovers Per Game, Points Per Game will be decreased by 0.12 units. We achieved parsimonious models with Adjusted R Square of 96.22% for the team set and 96.6% for the player set respectively. Both the models gave us the Variance Inflation factor less than 10 which indicates that there is not multicollinearity between the coefficients, and they are not inflated.



We did create residual plots (QQ Plots) for both the models, and we found that the residuals follow a normal distribution.

All Subset Regression Conclusion

Variable combinations to predict points per game showed combinations of all actions in a game which tallies with one of the prediction factors in the Principal Factor Analysis. It also shows a similar or more obvious version of the predictors of the relaxed lasso. This indicates that the lines of analysis used produced similar and valid results.

Appendix:

Individual reports:

Alaa

In our group project we decided to analyze 2022-2023 NBA Player Stats Dataset that collected from Basketball Reference to gain new insight into players performance for 2022-2023 season and recognize factors associated with it for team players who participated in The National Basketball Association for that season.

In this project we had to apply different analysis techniques to our dataset. Therefore, we came up with two different questions that could help us to satisfy this task. As part of the team, I was responsible for creating the second question that would allow us to run linear discriminate analysis (LDA), also, I gathered the important outputs to be able to discuss it with our professor.

By looking at the dataset we decided to figure out the relationship between a player's statistics and their position and whether we can classify players positions based on the available statistics. If we success in doing this classification, it could help the basketball coaches to place any player in the most suitable position depending on their previous statistics.

In the dataset we have five different positions which are (point guard (PG), shooting guard (SG), small forward (SF), power forward (PF), and center (C)). Each position has the following number of observations respectively 119, 154, 113, 108 and 115. After cleaning the dataset and removing some variables that highly correlated with each other and split the data into training and testing sets, I ran the linear discriminate analysis on the training set. The initial results of the LDA showed that the amount of separation in each component as the following LD1=80%, LD2=16%, LD3=2% and LD4 about 1%. After that I used the training set to predict the players position and visualize the results by using histogram that helps to recognize the separation, and scatterplot that helps to see the result of predicting the positions on training set. Then I did the same steps for testing set and visualized the result too. Finally, I compute the confusion matrix for both sets that help us to recognize the accuracy of LDA and interpret the results.

In this project, I learned the importance of understanding the dataset and figured out if we can reduce it by combining different variables or excluding unnecessary variables. This could help to avoid different problems such as multicollinearity. Also, understanding the dataset and the relation between variables can help to create different research questions. Honestly, I was afraid of using the basketball dataset, since I do not have any background on this game, however, I learned that having the tools of data analysis will help to perform analysis on any dataset and lots reading would help to gain the knowledge related the selected dataset.

To sum up, I could say that LDA did not separate the positions very well and this could happen due to different reasons such as our variables are not enough to detect the differences between positions or the strong correlation between variables affect the results. We might need to include different variables or combine two different datasets together to get better classification.

Victor

As the De Facto team lead, my role in this project consisted of suggesting research questions, guiding team members in their lines of analysis, and providing support throughout the project, including working on the first research goal. Throughout the project, I provided guidance and support to other team members, assisting them with coding issues and offering direction for their analyses. Additionally, I played a significant role in data cleaning and along with drafting the report.

The initial data set was perfectly multicollinear and posed issues at the initial stages of the project, as none of our chosen lines of analysis worked. To solve this issue, I split the data set in two and explained to my colleagues the reasons behind the split and how we should go about it. It worked.

I worked on the first research question where I used Factor Analysis to find latent factors that affect points per game. I employed Principal Factor Analysis (PFA) and Common Factor Analysis (CFA) to identify latent variables influencing this metric. I conducted three main PFAs: one using log-transformed variables from the player dataset after addressing normality, another using the original untransformed variables without removing aliased variables, and a third utilizing the player dataset with untransformed variables while removing highly correlated aliased variables. All PFAs were scaled to ensure comparability.

Furthermore, I conducted three CFAs: The first employed a log-transformed variables version of the data to achieve normality, while the second used the untransformed dataset without aliased variables removed. The third CFA used the untransformed variables but with the aliased variables removed. Dummy position variables were removed for all the CFAs.
I thought to try using both the log-transformed variables and the untransformed variables to see the effect on the results of the PFA and CFA. Interestingly two PFAs and two CFAs gave similar models (The untransformed and transformed with aliasing removed) gave similar models. The Untransformed PFA without Aliasing removed did not give any meaningful result. Based on the scores obtained from the PFAs and CFA, I built regression models to explore the relationship between the latent factors and points per game used all-subset regression models to validate the OLS models on the Factors. KMO on the dataset gave an MSA value of 0.5. The final Model Chosen for the report was the first PFA as all other models seemed to be a version of it. It also has the highest adj-squared value and was validated most by the all-subsets regression done on the scores when compared with initial OLS models. (See appendix section for all images).

Throughout the project, I gained valuable insights into data analysis and interpretation. One key lesson learned was the existence of perfect multicollinear datasets and the need to develop strategies to address such issues. Additionally, I discovered the significance of data dictionaries and the importance of domain knowledge, as they assisted in understanding the data and resolving analysis challenges. Also, learning to accommodate ideas from team members

regardless of how sure you are yours is better, is a good skill that helps actualize projects. Effective communication plays a huge role here. I also learned some of the individual strengths of my team members: Alaa (organization and the calm in a storm), Satchit the enthusiastic doer, and Anirudh the free soul enthusiast. We all worked together to make the project work. Lastly, confirmatory factor analysis from the Lavan package seems not to work on principal factor analysis when the data contains dummy variables. The errors I kept getting gave me a tough time.

In conclusion, Latent factors that were not immediately intuitive to the mind, affecting points per game were found out. The effect of the synergies or counteraction (if on opposing teams) between the Shooting guard and the Center, seemed to be especially important. Other synergies between other position-pairings with the shooting guard like PF-SG, SF-SG, PG-SG in terms of assists, setting screens to give the SG room for field goal attempts etc., along with defensive rebounds, seem to be latent factors that affect the points per game. The regression model expression from the principal factor analysis

PTS = 1.738 + 0.08(actions_in_game) - 0.04(3pattemps_by_SG_and_BLK_by_C) - 0.03(synergy_btw_PG_and_SG) + 0.21 (DRB)

Indicates that a unit increase in all the actions aimed at scoring points per game will result in a corresponding increase of 0.08 in points scored per game. Also, counteraction in terms of a block by a center on a unit 3point attempt by a shooting guard will decrease the points per game by 0.04 units. The synergy (or constructive collaboration) between the point guard and the Shooting guard in terms of assists will increase or decrease points per game by 0.03 per unit assist. Also, a unit increase in defensive rebounds will increase points per game by 0.21 units. Team managers can therefore use this model to optimize game points.

Sachit

My work in the project involved the first line of approach, which was modeling around points per game. As stated in the corresponding section's conclusion, regularized regression was used on a slimmed down version of the dataset consisting solely of player statistics. The initial OLS model was extremely susceptible to overfitting. As such, regularized regression was an approach chosen, as adding some bias could help reduce the variance (bias-variance tradeoff).

The end-result model of the relaxed LASSO is like a small subset of the various predictors captured by principal and common factor analyses. The relaxed lasso model can't track the specific synergies listed in factor analyses, since position was excluded from the model (although 3-point attempts are factored in, just not for specific positions). In this sense, the relaxed lasso model uses a subset of predictors that the principal and common factor analyses use. 2-point attempts, 3-point attempts, free throws and defensive rebounds appear near-universally within these analyses. The factor analyses are an expansion of the relaxed lasso model that considers the complexities created by positioning (scores and actions of specific positions and synergies between positions).

What influenced the direction of this analysis was the choice in which variables were removed for multicollinearity's sake. A lot of the multicollinearity lies within "partial" variables

within the dataset. There are some variables that are “pieces” of another. For example, there are 3 variables for 2 points. 2 points shots scored, 2 points shot attempted, and 2 point shot success rate percentage. This exists for 3 points, total field goal percent (as well as effective field goal%), free throws, and so on. Refer to figure 1 in the regularized regression output section for leftover variables and VIF scores.

Removing a lot of these overlapping variables was the key to removing the multicollinearity from the dataset, which allowed for a stronger relaxed lasso. Because of the “partial” variables with similar multicollinearity profiles, there is a choice in which partial variable to remove. I could have chosen to keep blocks over offensive rebounds, 2-point and 3-point over 2-point and 3-point attempts, etc.

There are a couple lessons I took away from this project. For one. I learned how to do relaxed lasso specifically for this project from the supplemental lecture. I learned some of the intuition and process in group-work, particularly through the group’s emphasis on direction and oversight of if something is truly an insight or not. Some smaller details I learned include specific coding techniques from the rest of the group, as this was an area I struggled in specifically.

Effective group communication I have found to not just be important, but also a skill in itself. A more efficient, respectful, and cooperative communicator can help organize, motivate, and coordinate a project while also not acting utilitarian. Lastly, I learned about the importance of multimedia in sharing findings (messaging simply is not enough). Being able to present screenshots, various plots, console outputs, and interpretations done ahead of time make a significant difference compared to plain text. Proper use of multimedia allows for more efficient and significant communication. One last thing I learned is that in the future, it would be easier for myself to create multiple R Scripts to stay organized. For my work, I combined it all into a single R script. It would’ve been more useful if I kept backup copies so that I would not have to remember all of the changes and risk losing old data.

Model equations for the initial and final relaxed LASSO models can be found below and at the end of the regularized regression section.

Initial CV GLMNET Model: PTS = 0.68913 + 0.00018(G) + 0.00663(MP) + 0.55103(x3p) + 0.67776(x2p) + 0.12714(FTA) + 0.07761(DRB)
Adjusted Model: PTS = 0.3724088 + .3675337(X3PA) + .5287313(X2PA) + 0.2465942(FT) + 0.2774022(DRB)

Anirudh

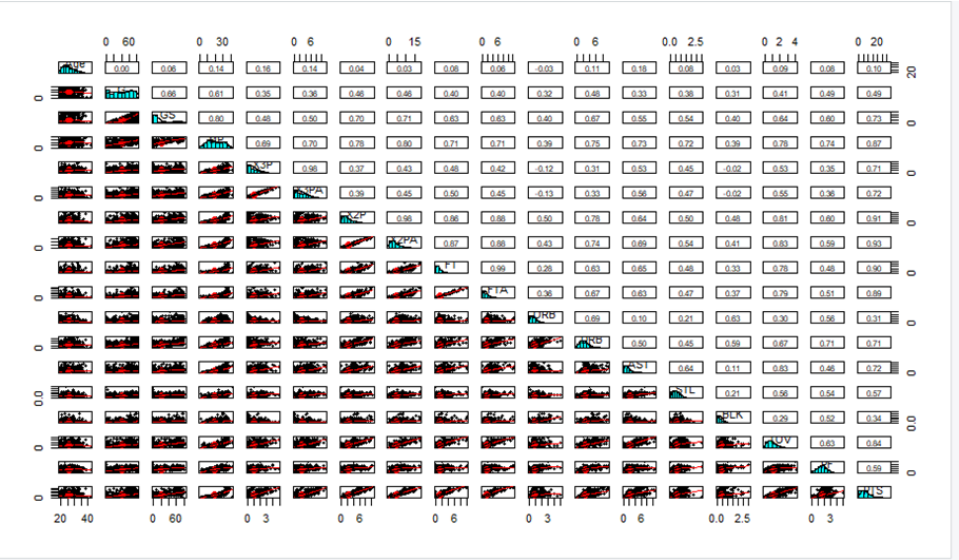
For this project, we divided the dataset into two datasets: team and player statistics sets. We are predicting the points per game, and I have used All subset regression analysis to get significant variables for both the datasets. So, I used the BIC Statistics to get the exact significant variables as it gives the most precise significant ones as compared to adj R square and AIC. I created a linear regression model using those variables for both the datasets and further created residual Quantile-Quantile Plots for the model's analysis. According to the plot, residuals follow a normal distribution for both the sets. All this analysis was used to compare the results with Lasso and Factor Analysis. We got comparable and valid results as per the analysis.

Apart from this I tried to go ahead with multidimensional scaling and Cluster analysis. I did K Means clustering as we had numerical data and I got three clusters formed. I was able to create the Shephard diagram with a stress of 3.4%. But somehow it was not getting fit for the datasets, and I could not draw much interpretation from the results as they were a bit complex, so I decided to go ahead with a different one.

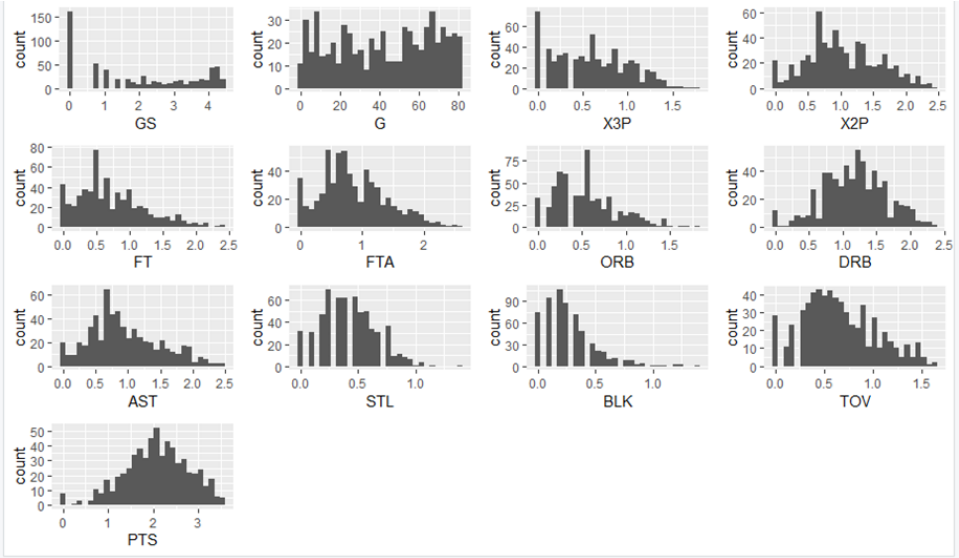
About learnings in this project, I would say that a sizable portion of the project involved data preprocessing techniques. This included handling missing values, dealing with outliers, scaling, and normalizing data, and performing feature selection and engineering. These steps were crucial for ensuring data quality and preparing it for analysis. Visualization played a crucial role as my learning curves in this project. Plots and charts, including scatter plots and histograms were used to present findings, patterns, and relationships in the data. I learned about translating complex analyses into actionable insights and presenting them in a clear and concise manner, which is crucial for driving decision-making processes. Overall, this project covered a wide range of topics and techniques, providing a comprehensive understanding of every topic.

Graphs and Outputs:

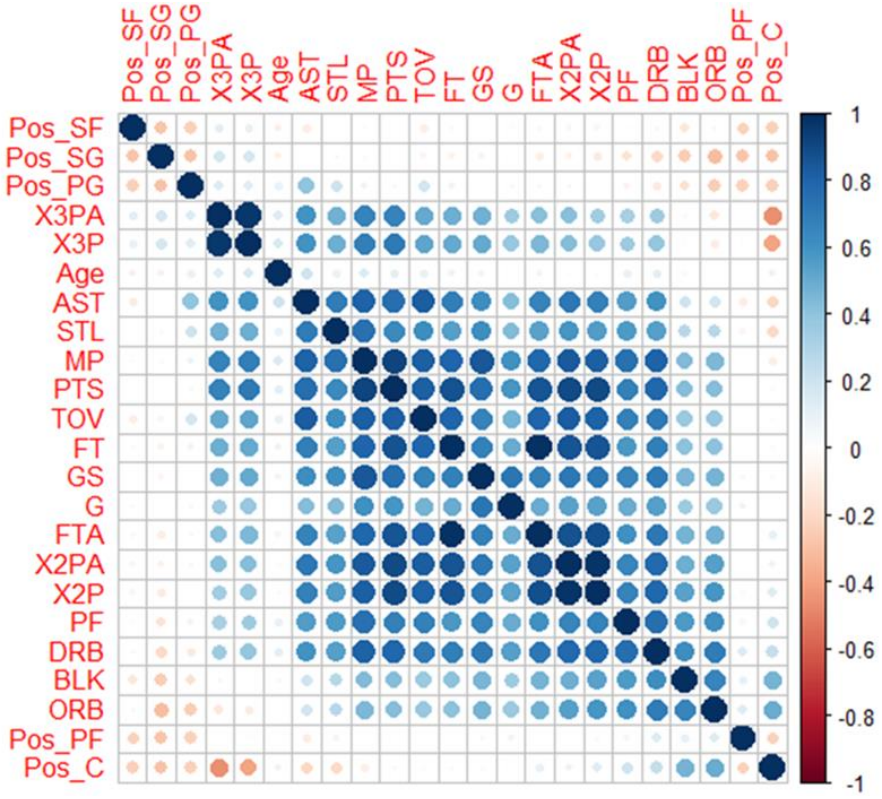
Initial Pairs panels plot showing Distribution of Variables:



Histograms of some variables after log transformation:



Initial Full Dataset Corrplot:



Addendum Regularized Regression Output:

```
> vif(fit1)
       Age          G         GS       X3PA       X2PA         FT        DRB
  1.148365   2.153664   3.852235   2.033714   5.669877   4.073425   4.928944
       AST        STL        BLK         PF Position_dummy
  4.646091   2.407818   2.225933   2.996943   2.796241
>
```

Figure 1: Remaining Player Stats variables and their VIF Values.

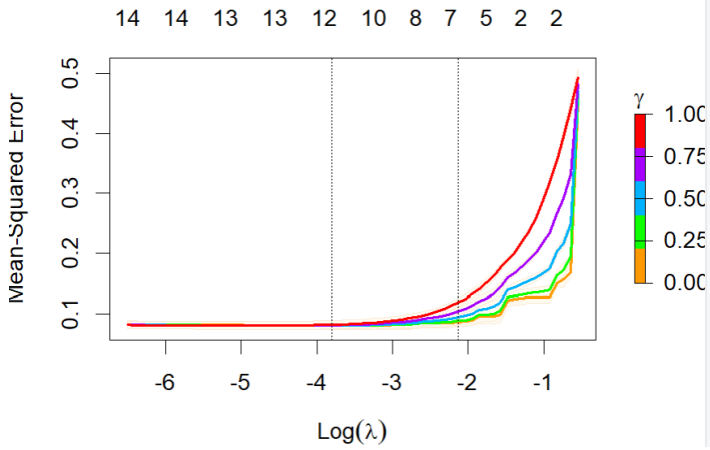


Figure 1.1: Relaxed LASSO error plot output

```
> coef(fitLasso2, s="lambda.1se")
13 x 1 sparse Matrix of class "dgCMatrix"
                             s1
(Intercept)     0.5363707794
Age             .
G               0.0009251931
GS              .
X3PA            0.3010356751
X2PA            0.4751484473
FT              0.2707712163
DRB             0.2206258663
AST             .
STL             .
BLK             .
PF              .
Position_dummy  .
```

Figure 1.2: Predictors chosen by initial Relaxed LASSO variable selection.

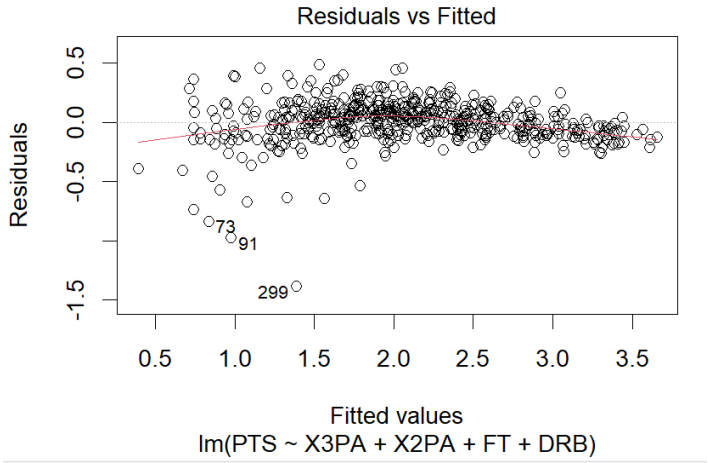


Figure 1.3: Residuals vs Fitted plot for OLS model with LASSO chosen predictors.

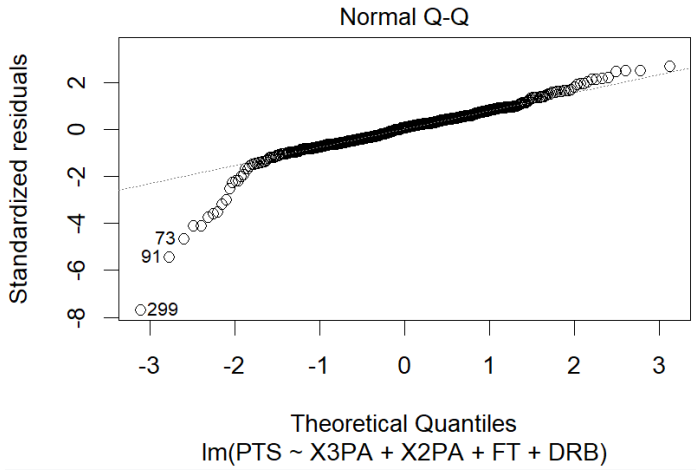


Figure 1.4: Normality QQ plot for OLS model with LASSO chosen predictors.

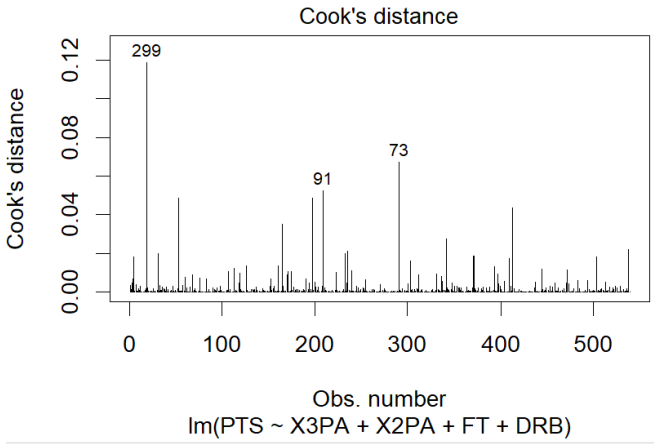


Figure 1.5: Cook's Distance plot for OLS model with LASSO chosen predictors.

```
> fit1se = glmnet(xTrain, yTrain, data=dsTrain, lambda = 0.04754607)
> glmnet(xTrain, yTrain, data=dsTrain, lambda = 0.04754607)

Call:  glmnet(x = xTrain, y = yTrain, lambda = 0.04754607, data = dsTrain)

  Df  %Dev  Lambda
1  5 92.94 0.04755
> #how do we get the dev%?
> pLassoTrain = predict(fit1se, xTrain, s="lambda.1se")
> rmse_lasso_train = sqrt(mean((pLassoTrain - yTrain)^2))
> rmse_lasso_train  # 0.18 with set seed
[1] 0.1903808
```

```
> fitLasso$lambda.1se
[1] 0.04754607
```

Figure 1.6: Various statistics for the improved relaxed LASSO model (RMSE, dev %, lambda.1se).

```
> coef(fitLasso, s="lambda.1se", gamma=0)
12 x 1 sparse Matrix of class "dgCMatrix"
                      s1
(Intercept)    0.3724088
Age            .
GS             .
X3PA           0.3675337
X2PA           0.5287313
FT             0.2465942
DRB            0.2774022
AST            .
STL            .
BLK            .
PF             .
Position_dummy .
>
```

Figure 1.7: Coefficients (variables selected) for the adjusted Relaxed LASSO model.

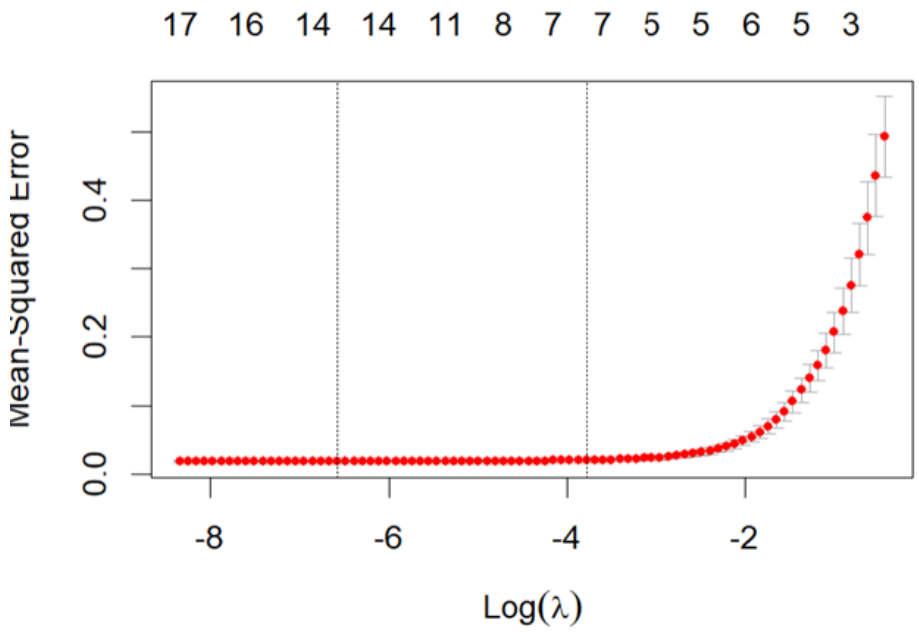


Figure 2.0: CV GLMNET Relaxed LASSO plot output

```
> coef(fitRange, s="lambda.1se")
19 x 1 sparse Matrix of class "dgCMatrix"
                      s1
(Intercept)   0.6891376961
Age           .
G             0.0001894733
GS            .
MP            0.0066331959
X3P           0.5510309268
X3PA          .
X2P           0.6777640524
X2PA          .
FT            .
FTA           0.1271495959
ORB           .
DRB           0.0776164331
AST           .
STL           .
BLK           .
TOV           .
PF            .
Position_dummy .
```

Figure 2.1: Predictors after variable selection in the initial CV GLMNET relaxed LASSO model.

```
> fitRange$lambda.1se
[1] 0.03956875
>
```

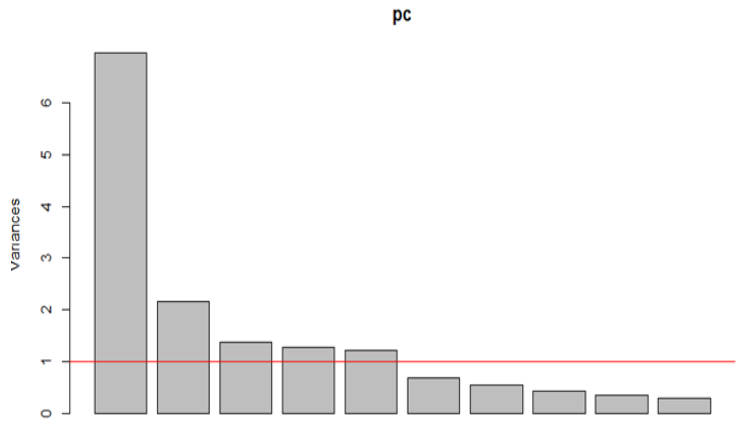
Figure 2.2: Lambda.1se for CV GLMNET model

Factor Analysis Output:

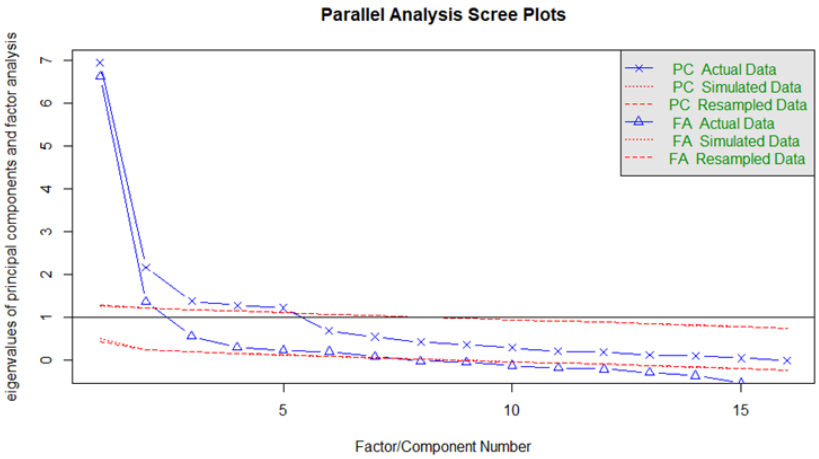
```
Kaiser-Meyer-Olkin factor adequacy
Call: KMO(r = nba_plyrstat4)
Overall MSA =  0.5
MSA for each item =
   Age      G     GS     MP    X3P   X3PA    X2P   X2PA     FT    FTA    ORB    DRB    AST    STL    BLK    TOV
   0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5    0.5
    PF    PTS  Pos_C Pos_PF Pos_PG Pos_SF Pos_SG
   0.5    0.5    0.5    0.5    0.5    0.5    0.5
>
```

KMO test on log transformed Player set.

PFA1



PFA1 Scree Plot.



PFA1 Parallel Analysis Plot.

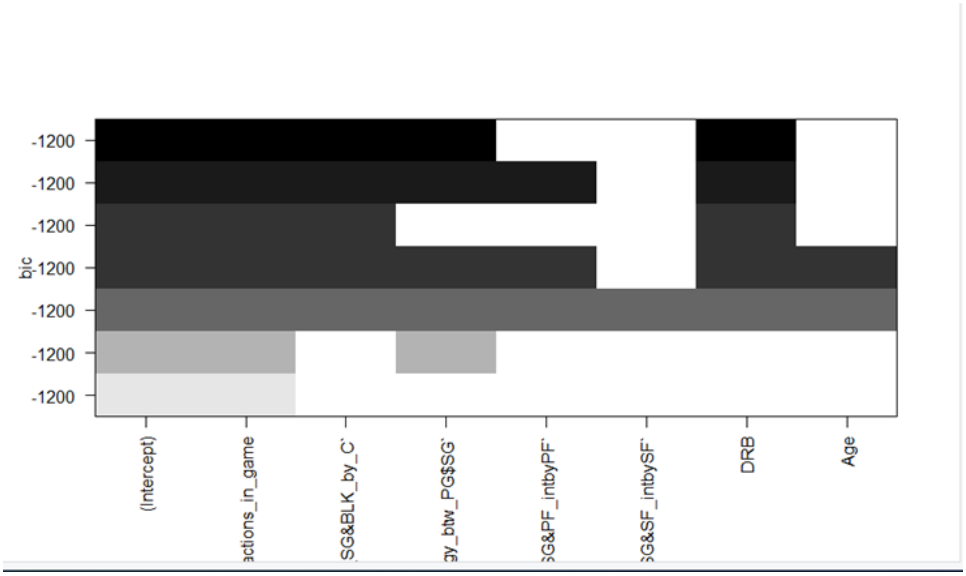
```
> print(pf1$loadings,cutoff=.4,sort=T)

Loadings:
        RC1    RC2    RC3    RC5    RC4
G        0.684
GS       0.868
MP       0.963
X2PA     0.892
FTA      0.852
AST      0.816
STL      0.754
TOV      0.875
PF       0.794
X3PA     0.597 -0.599
BLK      0.512  0.648
Pos_C           0.916
Pos_PG                0.929
Pos_PF                       0.974
Pos_SF                              0.951
Pos_SG          -0.437 -0.534 -0.446 -0.550

                 RC1   RC2   RC3   RC5   RC4
SS loadings    6.923 2.075 1.451 1.290 1.272
Proportion Var 0.433 0.130 0.091 0.081 0.079
Cumulative Var 0.433 0.562 0.653 0.734 0.813
```

Loadings for PFA1 (cutoff of .4 used).

```
> fitR1 = lm(PTS ~., data=scores)
> summary(fitR1)

Call:
lm(formula = PTS ~ ., data = scores)

Residuals:
     Min       1Q   Median       3Q      Max
-1.22639 -0.12801  0.02179  0.16023  0.82805

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)               1.738303   0.081828  21.243  < 2e-16 ***
actions_in_game           0.084580   0.003195  26.470  < 2e-16 ***
`3pattemps_by_SG&BLK_by_C` -0.035236   0.005822  -6.052 2.51e-09 ***
`synergy_btw_PG$SG`       -0.030903   0.007670  -4.029 6.31e-05 ***
`synergy_btw_SG&PF_intbyPF` -0.011765   0.008499  -1.384    0.167
`synergy_btw_SG&SF_intbySF` 0.006336   0.008252   0.768    0.443
DRB                       0.209720   0.048027   4.367 1.49e-05 ***
Age                       0.003143   0.002463   1.276    0.202
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2571 on 601 degrees of freedom
Multiple R-squared:  0.8689,    Adjusted R-squared:  0.8674
F-statistic: 569.3 on 7 and 601 DF,  p-value: < 2.2e-16
```

Initial OLS on PFA1.

All subsets plot using BIC scale for PFA 1:



All subsets plot using BIC scale for PFA 1.
(Notice same significant factors as the OLS was produced)

Checking for significance of Age in increasing predictive power for PFA1 model. R-squared 1% low.

```
> PFA1agechk <- lm(PTS~ Age, data= scores)
> summary(PFA1agechk)

Call:
lm(formula = PTS ~ Age, data = scores)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1258 -0.4473  0.0072  0.4842  1.4950

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.583280   0.173291   9.137  < 2e-16 ***
Age         0.018709   0.006602   2.834  0.00475 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.702 on 607 degrees of freedom
Multiple R-squared:  0.01306,   Adjusted R-squared:  0.01143
F-statistic: 8.032 on 1 and 607 DF,  p-value: 0.00475
```

Checking for significance for including Power forward and Shooting guard in Model increasing predictive Power. For PFA1. R-squared value too low. No significant increase in predictive power of model.

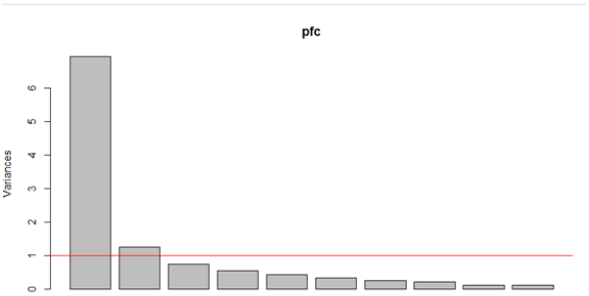
Checking for significance for including Power forward and Shooting guard in Model increasing predictive Power. For PFA1. R-squared value too low. significant increase in predictive power of model.

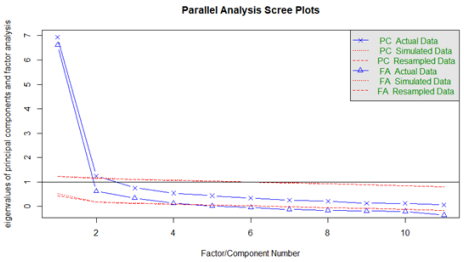
```
> PFandSGPF1 <- lm(PTS~`synergy_btw_SG&PF_intbyPF`, data=scores)
> summary(PFandSGPF1)

Call:
lm(formula = PTS ~ `synergy_btw_SG&PF_intbyPF`, data = scores)

Residuals:
     Min       1Q   Median       3Q      Max
-2.12417 -0.45477  0.00986  0.49401  1.46094

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                2.06773    0.02861  72.275   <2e-16 ***
`synergy_btw_SG&PF_intbyPF` 0.02290    0.02192   1.045    0.297
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.706 on 607 degrees of freedom
Multiple R-squared:  0.001794,  Adjusted R-squared:  0.0001499
F-statistic: 1.091 on 1 and 607 DF,  p-value: 0.2966
```

No

CFA1 images:

```
> summary(ptc)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10    PC11
Standard deviation     2.6337 1.1166 0.86793 0.73548 0.66005 0.58122 0.49622 0.45690 0.35706 0.33862 0.22810
Proportion of Variance 0.6306 0.1133 0.06848 0.04918 0.03961 0.03071 0.02238 0.01898 0.01159 0.01042 0.00473
Cumulative Proportion  0.6306 0.7439 0.81240 0.86158 0.90118 0.93189 0.95428 0.97326 0.98485 0.99527 1.00000
> |
```



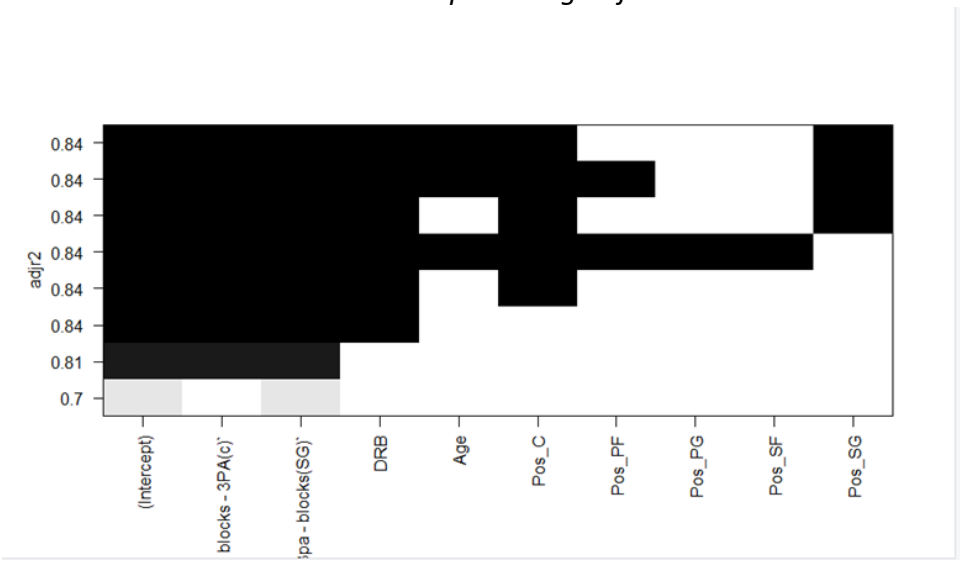


CFA 1 factor Loadings

```
> fitnba_ply1 = factanal(nba_factor, factors=2)
> print(fitnba_ply1)

Call:
factanal(x = nba_factor, factors = 2)

Uniquenesses:
    G    GS    MP  X3PA  X2PA   FTA   AST   STL   BLK   TOV    PF
0.608 0.286 0.040 0.264 0.176 0.258 0.285 0.437 0.489 0.256 0.380

Loadings:
     Factor1 Factor2
G     0.482   0.400
GS    0.624   0.570
MP    0.616   0.762
X3PA          0.858
X2PA  0.769   0.482
FTA   0.727   0.462
AST   0.434   0.725
STL   0.417   0.624
BLK   0.714
TOV   0.630   0.589
PF    0.675   0.406

                Factor1 Factor2
SS loadings       3.847   3.674
Proportion Var    0.350   0.334
Cumulative Var    0.350   0.684

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 866.54 on 34 degrees of freedom.
The p-value is 5.22e-160
```

CFA 1 Factor Loadings Output.

CFA 1 all subsets plot using Adjr2 scale.



```
> CFA1bestfit <- lm(PTS ~. -Pos_PF - Pos_PG - Pos_SF, data=scores3)
> summary(CFA1bestfit)

Call:
lm(formula = PTS ~ . - Pos_PF - Pos_PG - Pos_SF, data = scores3)

Residuals:
     Min       1Q   Median       3Q      Max
-1.86143 -0.14331  0.02667  0.17327  0.61071

Coefficients:
                                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                                      0.644835   0.075213   8.573  < 2e-16 ***
`Actions in games played including blocks - 3PA(c)` -0.136362   0.008103 -16.828  < 2e-16 ***
`Actions in games with 3pa - blocks(SG)`         0.168733   0.008919  18.919  < 2e-16 ***
DRB                                              0.327697   0.045285   7.236 1.41e-12 ***
Age                                             -0.003954   0.002721  -1.453  0.14667
Pos_C                                            0.115274   0.037408   3.082  0.00215 **
Pos_SG                                           0.053652   0.027794   1.930  0.05404 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.28 on 602 degrees of freedom
Multiple R-squared:  0.8443,    Adjusted R-squared:  0.8427
F-statistic: 544.1 on 6 and 602 DF,  p-value: < 2.2e-16
```

CFA 1 regression model output.

```
> agecheck <- lm(PTS~ Age, data= scores3)
> summary(agecheck)

Call:
lm(formula = PTS ~ Age, data = scores3)

Residuals:
    Min      1Q  Median      3Q     Max
-2.1258 -0.4473  0.0072  0.4842  1.4950

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.583280   0.173291   9.137  < 2e-16 ***
Age         0.018709   0.006602   2.834  0.00475 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.702 on 607 degrees of freedom
Multiple R-squared:  0.01306,   Adjusted R-squared:  0.01143
F-statistic: 8.032 on 1 and 607 DF,  p-value: 0.00475
```

Checking if Age should be in the CFA1 model. R-squared about 1%

```
> summary(PosSGcheck)

Call:
lm(formula = PTS ~ Pos_SG, data = scores3)

Residuals:
     Min       1Q   Median       3Q      Max
-2.07132 -0.46188  0.00812  0.49363  1.45798

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.07132    0.03313  62.526   <2e-16 ***
Pos_SG      -0.01421    0.06588  -0.216    0.829
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.7066 on 607 degrees of freedom
Multiple R-squared:  7.67e-05,  Adjusted R-squared:  -0.001571
F-statistic: 0.04656 on 1 and 607 DF,  p-value: 0.8292

>
```

Checking for Shooting guard too. R-squared is found to be negligible.

```
Factor analysis with Call: principal(r = nba_pstatpfa, nfactors = 7, rotate = "varimax")

Test of the hypothesis that 7 factors are sufficient.
The degrees of freedom for the model is 84  and the objective function was  26.93
The number of observations was  609  with Chi Square =  16037.15  with prob <  0

The root mean square of the residuals (RMSA) is  0.04
> print(ptpf2$loadings, cutoff=.4, sort=T)

Loadings:
       RC1    RC6    RC2    RC3    RC5    RC4    RC7
X2P     0.878
X2PA    0.892
FT      0.932
FTA     0.936
AST     0.648  0.430         0.410
TOV     0.796  0.410
G              0.715
GS      0.506  0.666
MP      0.596  0.753
X3P            0.639 -0.454
X3PA           0.638 -0.467
STL            0.668
PF             0.656  0.422
ORB                   0.794
BLK                   0.724
Pos_C                 0.833
Pos_PG                       0.919
Pos_PF                              0.983
Pos_SF                                     0.942
Pos_SG                -0.402 -0.523 -0.404 -0.568
Age                                                0.934

                 RC1   RC6   RC2   RC3   RC5   RC4   RC7
SS loadings    5.665 4.044 2.892 1.467 1.286 1.278 1.140
Proportion Var 0.270 0.193 0.138 0.070 0.061 0.061 0.054
Cumulative Var 0.270 0.462 0.600 0.670 0.731 0.792 0.846
```

PFA on the Untransformed data without Aliased vars removed.

```
> ftu =lm(PTS ~ `AST_TOV_3P_3PA_STL&PF in Games`+`Personal_fouls_ORB_BLK_on 3p & 3PA_by_C`+ `Synergy_btw_PG&SG_in_A
+ PF + `synergy_btw_SG&SF` + DRB  +  Age, data= scores4)
> summary(ftu)

Call:
lm(formula = PTS ~ `AST_TOV_3P_3PA_STL&PF in Games` + `Personal_fouls_ORB_BLK_on 3p & 3PA_by_C` +
    `Synergy_btw_PG&SG_in_Ast` + PF + `synergy_btw_SG&SF` + DRB +
    Age, data = scores4)

Residuals:
    Min      1Q  Median      3Q     Max
-12.201  -1.289   0.053   1.282   9.941

Coefficients:
                                      Estimate Std. Error t value Pr(>|t|)
(Intercept)                           7.841857   0.689676  11.370  < 2e-16 ***
`AST_TOV_3P_3PA_STL&PF in Games`      1.035392   0.030841  33.572  < 2e-16 ***
`Personal_fouls_ORB_BLK_on 3p & 3PA_by_C` -0.078694   0.049413  -1.593    0.112
`Synergy_btw_PG&SG_in_Ast`           -0.065183   0.072230  -0.902    0.367
PF                                   -0.124268   0.083901  -1.481    0.139
`synergy_btw_SG&SF`                   0.063032   0.080962   0.779    0.437
DRB                                   0.499542   0.123491   4.045 5.91e-05 ***
Age                                  -0.008168   0.024355  -0.335    0.737
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.543 on 601 degrees of freedom
Multiple R-squared:  0.8602,    Adjusted R-squared:  0.8586
F-statistic: 528.3 on 7 and 601 DF,  p-value: < 2.2e-16
```

OLS regression on Untransformed PFA2 without after multicollinear variables have been removed.

CFA 2 (on untransformed variable with Aliasing variable in the data) dummies removed:

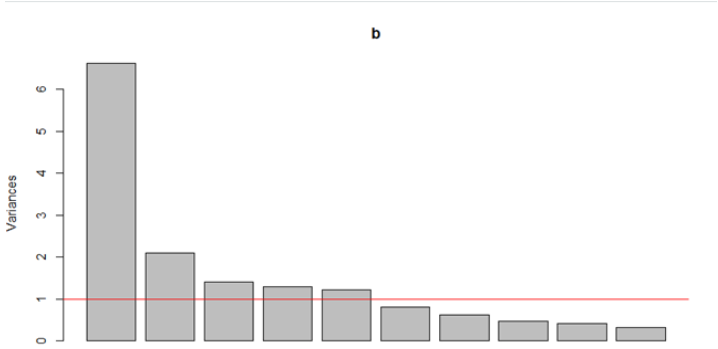
```
Call:
factanal(x = nba_pstatpfa2, factors = 2)

Uniquenesses:
    G    GS    MP   X3P  X3PA   X2P  X2PA    FT   FTA   ORB   AST   STL   BLK   TOV    PF
0.747 0.441 0.200 0.035 0.005 0.014 0.017 0.215 0.197 0.634 0.455 0.645 0.735 0.261 0.613

Loadings:
     Factor1 Factor2
G     0.348   0.364
GS    0.550   0.506
MP    0.547   0.707
X3P           0.982
X3PA          0.997
X2P   0.910   0.399
X2PA  0.878   0.460
FT    0.727   0.507
FTA   0.771   0.456
ORB   0.592  -0.124
AST   0.476   0.564
STL   0.364   0.472
BLK   0.515
TOV   0.653   0.559
PF    0.503   0.366

                Factor1 Factor2
SS loadings       5.10   4.687
Proportion Var    0.34   0.312
Cumulative Var    0.34   0.652

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 3408.55 on 76 degrees of freedom.
The p-value is 0
```

Found Similar factors like the one for the transformed.

```
> summary(utfit)

Call:
lm(formula = PTS ~ ., data = scores7)

Residuals:
     Min       1Q   Median       3Q      Max
-10.5327  -1.9003  -0.1313   1.5476  13.1985

Coefficients: (1 not defined because of singularities)
                                                      Estimate Std. Error t value Pr(>|t|)
(Intercept)                                            0.24902    0.37838   0.658   0.5107
`Actions_in_games_played_including blocks_minus_3PA,3p&STLs`  0.05973    0.04168   1.433   0.1523
`Actions_in_Games_Including_3PA&3PA -2P,ORB,BLK&PF`    0.07003    0.04126   1.698   0.0901 .
DRB                                                    1.22703    0.13458   9.118  < 2e-16 ***
Pos_C                                                 -2.92044    0.49857  -5.858 7.73e-09 ***
Pos_PF                                                -1.85659    0.44068  -4.213 2.91e-05 ***
Pos_PG                                                 0.33256    0.40553   0.820   0.4125
Pos_SF                                                -0.61104    0.41411  -1.476   0.1406
Pos_SG                                                     NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.317 on 601 degrees of freedom
Multiple R-squared:  0.7621,    Adjusted R-squared:  0.7593
F-statistic:   275 on 7 and 601 DF,  p-value: < 2.2e-16
```

OLS for CFA2 (untransformed) after adding back singular factors and Position dummies.

PFA 3 (untransformed data with Aliasing variables removed):



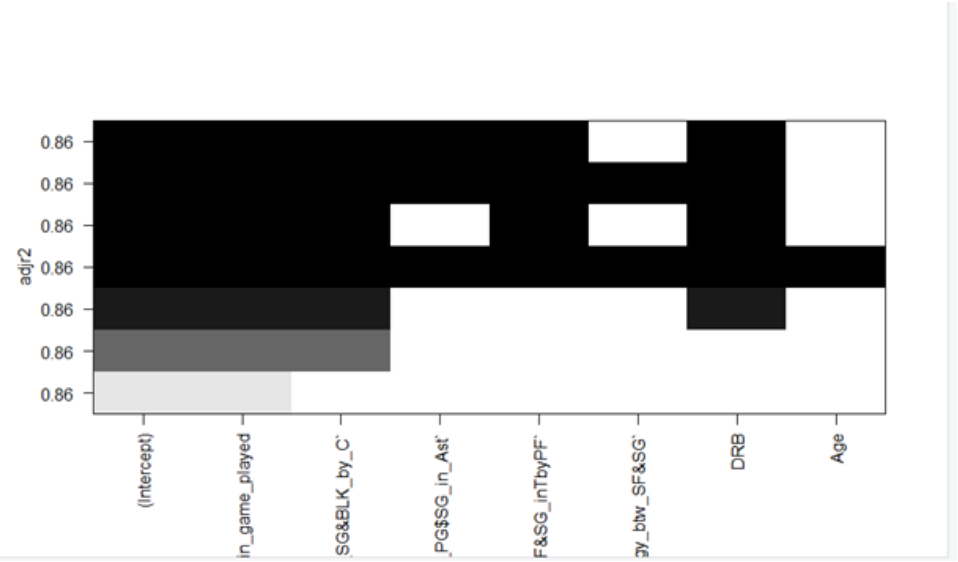
The scree plot output suggests 5 components.

```
> summary(pnt)

Factor analysis with Call: principal(r = nba_pstatpt4, nfactors = 5, rotate = "varimax")

Test of the hypothesis that 5 factors are sufficient.
The degrees of freedom for the model is 50  and the objective function was  21.94
The number of observations was  609  with Chi Square =  13131.42  with prob <  0

The root mean square of the residuals (RMSA) is  0.05
> print(pnt$loadings, cutoff=.4, sort=T)

Loadings:
       RC1    RC2    RC3    RC5    RC4
G       0.644
GS      0.841
MP      0.959
X3PA    0.679 -0.464
X2PA    0.875
FTA     0.812
AST     0.782         0.424
STL     0.727
TOV     0.870
PF      0.747
BLK     0.408  0.708
Pos_C          0.907
Pos_PG                0.904
Pos_PF                       0.973
Pos_SF                              -0.949
Pos_SG        -0.437 -0.529 -0.445  0.552
```

PFA 3 loadings output.

*Similar to the transformed PFA
Initial OLS for PFA 3 (Untransformed):

```
> summary(ftut)

Call:
lm(formula = PTS ~ ., data = scores5)

Residuals:
     Min       1Q   Median       3Q      Max
-13.1654  -1.2964   0.0113   1.3461  10.4788

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)                7.965211   0.685542  11.619  < 2e-16 ***
actions_in_game_played     0.911579   0.028617  31.854  < 2e-16 ***
`3pattemps_by_SG&BLK_by_C` -0.278494   0.062674  -4.444 1.05e-05 ***
`synergy_btw_PG$SG_in_Ast` -0.110514   0.072411  -1.526  0.12748
`synergy_btw_PF&SG_inTbyPF` -0.156140   0.083209  -1.876  0.06107 .
`synergy_btw_SF&SG`        -0.051899   0.080531  -0.644  0.51953
DRB                         0.314358   0.116883   2.690  0.00735 **
Age                         0.005931   0.024077   0.246  0.80549
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.522 on 601 degrees of freedom
Multiple R-squared:  0.8625,    Adjusted R-squared:  0.8609
F-statistic: 538.5 on 7 and 601 DF,  p-value: < 2.2e-16
```




```
> PFA3bestfit <- lm(PTS~.-`synergy_btw_SF&SG` -Age, data= scores5)
> summary(PFA3bestfit)

Call:
lm(formula = PTS ~ . - `synergy_btw_SF&SG` - Age, data = scores5)

Residuals:
     Min       1Q   Median       3Q      Max
-13.2043  -1.2860   0.0131   1.3372  10.6269

Coefficients:
                           Estimate Std. Error t value Pr(>|t|)
(Intercept)                 8.10597    0.32355  25.053  < 2e-16 ***
actions_in_game_played      0.91013    0.02850  31.934  < 2e-16 ***
`3pattemps_by_SG&BLK_by_C` -0.27891    0.06259  -4.456 9.96e-06 ***
`synergy_btw_PG$SG_in_Ast` -0.10977    0.07177  -1.529  0.12667
`synergy_btw_PF&SG_inTbyPF` -0.15421    0.08303  -1.857  0.06377 .
DRB                         0.31922    0.11640   2.742  0.00628 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.519 on 603 degrees of freedom
Multiple R-squared:  0.8624,    Adjusted R-squared:  0.8612
F-statistic: 755.7 on 5 and 603 DF,  p-value: < 2.2e-16

>
```

Returns the same model that the parsimonious ols for CFA3 gave. Check for age gave Residual standard error: 6.731 on 607 degrees of freedomMultiple R-squared: 0.01048, Adjusted R-squared: 0.008854

```
> summary(SGPGcheck)

Call:
lm(formula = PTS ~ `synergy_btw_PG$SG_in_Ast`, data = scores5)

Residuals:
    Min      1Q  Median      3Q     Max
-12.117  -4.511  -1.490   3.315  23.136

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                 8.9479     0.2614  34.235  < 2e-16 ***
`synergy_btw_PG$SG_in_Ast`  1.2946     0.1656   7.818 2.38e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.45 on 607 degrees of freedom
Multiple R-squared:  0.09148,   Adjusted R-squared:  0.08998
F-statistic: 61.12 on 1 and 607 DF,  p-value: 2.384e-14
```

Check for synergy between PF and SG.

```
> print(faut$loadings, cutoff=.4, sort=T)

Loadings:
     Factor1 Factor2
G    0.548
GS   0.623   0.512
MP   0.850   0.523
X3PA 0.672
STL  0.646
PF   0.638
X2PA         0.883
FTA          0.863
AST  0.513   0.568
TOV  0.463   0.745
BLK

               Factor1 Factor2
SS loadings      3.476   3.438
Proportion Var   0.316   0.313
Cumulative Var   0.316   0.629
> #
```

Synergy between PG&SG check gave an Adj R2 of 8%. It should probably be in the model. Therefore, The PFA1 model seems to be performing slightly better. The best model seems to be PFA1. CF1 is of worthy mention as it serves to validate the findings of PF1A.

```
> fitcof1<- cfa(PFA.model5, data = nba_plyrd)
Error in lav_samplestats_icov(COV = cov[[g]], ridge = 1e-05, x.idx = x.idx[[g]],  :
  lavaan ERROR: sample covariance matrix is not positive-definite
In addition: Warning message:
In lav_data_full(data = data, group = group, cluster = cluster,  :
  lavaan WARNING: some observed variances are (at least) a factor 1000 times larger than others; use varTable(fit) to
investigate
> #
```

Verifying that PFA1 model posed a challenge because of the dummy variables (error output).

Discriminate Analysis Output:

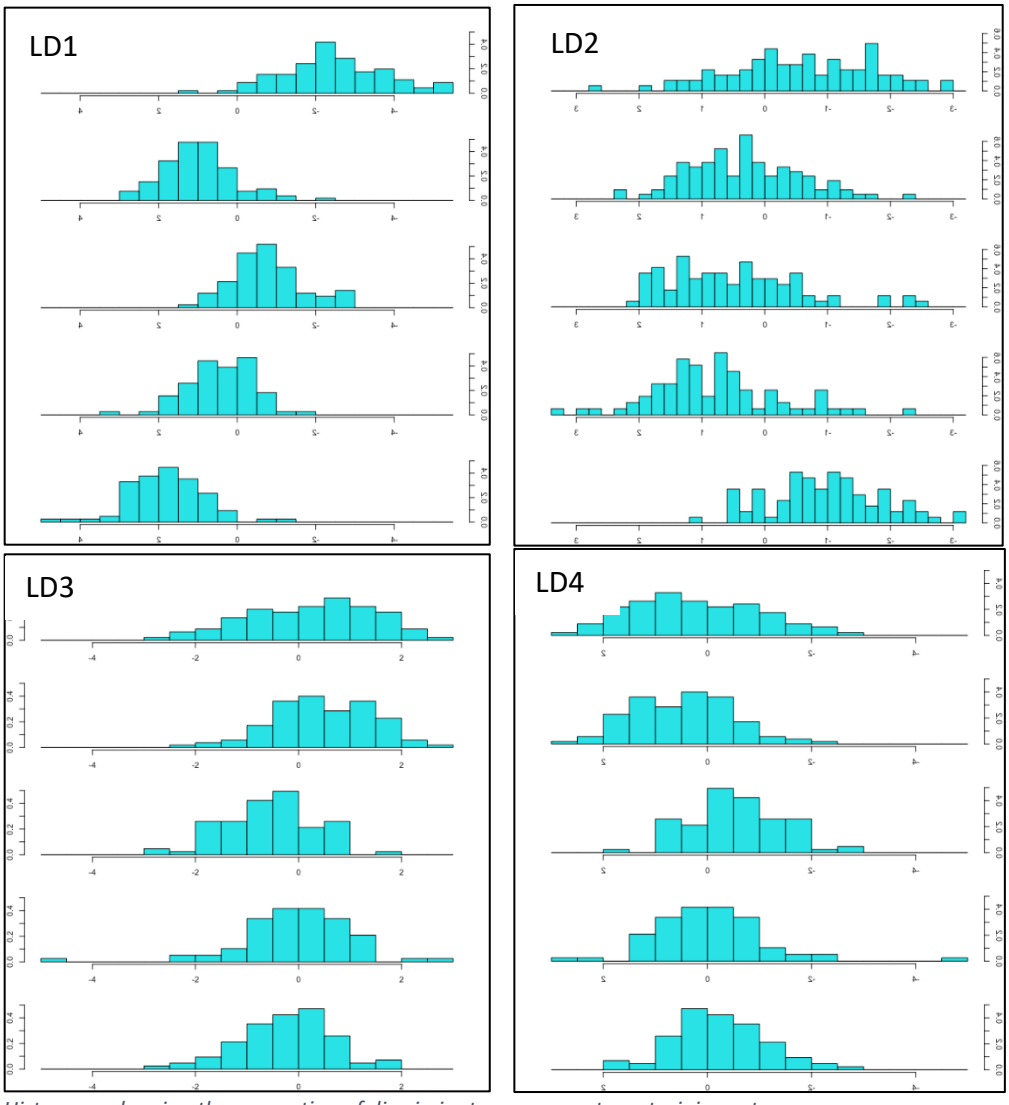
```
lda(position ~ ., data = train)

Prior probabilities of groups:
        1         2         3         4         5
0.2054176 0.2370203 0.1918736 0.1738149 0.1918736

Group means:
       Age       FG      FGA       FG.       X3P      X3PA      X3P.       X2P
1 26.59341 1.285311 1.713311 0.5537143 0.2380573 0.5150656 0.2507692 1.1982538
2 25.11429 1.309496 1.939555 0.4362857 0.7256188 1.3275224 0.3472095 0.9724429
3 25.70588 1.354985 1.913806 0.4822118 0.5663998 1.1364601 0.3243294 1.1292944
4 25.07792 1.252416 1.861026 0.4504545 0.6679998 1.2743996 0.3278571 0.9435097
5 26.48235 1.296334 1.965955 0.4085059 0.6927546 1.3215415 0.3193412 1.0010266
      X2PA      X2P.       eFG.        FT       FTA       FT.       ORB       DRB
1 1.548131 0.5943407 0.5829011 0.7630963 0.9728500 0.6799121 0.9110326 1.401858
2 1.409498 0.4881524 0.5214476 0.7246100 0.8408597 0.7307143 0.3790542 1.042335
3 1.506858 0.5774353 0.5455412 0.7562840 0.9042665 0.7235412 0.6713512 1.338861
4 1.345770 0.5170909 0.5343636 0.6561199 0.7856566 0.6724805 0.5049170 1.085816
5 1.466027 0.4563882 0.4825059 0.7725286 0.8875686 0.7240471 0.3582813 1.043785
        AST       STL       BLK       TOV       PF       PTS
1 0.7095006 0.3474485 0.5131784 0.6498294 1.947253 1.990502
2 0.9320702 0.4557841 0.2018925 0.6552036 1.503810 2.106613
3 0.8500674 0.4115781 0.3406205 0.6653396 1.780000 2.118637
4 0.8092808 0.4330453 0.2065212 0.5775447 1.593506 2.009043
5 1.3572042 0.5183068 0.1911225 0.7916296 1.469412 2.066546

Proportion of trace:
   LD1    LD2    LD3    LD4
0.7843 0.1511 0.0477 0.0169
```

Separation percentage of 1st discriminate analysis.

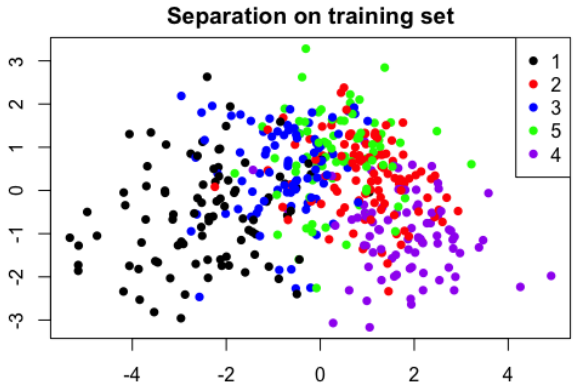
```
> print(fit.lda$scaling[order(fit.lda$scaling[,1]), ])
             LD1         LD2         LD3         LD4
FG   -4.02763606  4.38266888  0.89739751 -2.57128220
DRB  -1.98943012  0.93775814 -1.47889129  1.77586477
BLK  -1.39655382 -2.06168938  0.53301674  0.96152736
ORB  -1.27206638  0.11097264 -1.65760514 -2.19717425
X3P. -0.58133964 -0.57776780 -1.98016839  2.98340816
FT.  -0.57090178 -0.94368537 -0.05848004  0.50204063
FTA  -0.53572863 -0.75955042 -1.26510941 -1.90116657
X2P. -0.49820809  1.94559310 -2.56525567  1.94299323
FG.  -0.47644481 -17.07829257  6.63536432  8.82148857
PF   -0.38563483  0.23204581  0.33751372 -0.59572019
X2PA -0.24189945  4.27780861  0.42069909 -0.06307064
Age  -0.05981743 -0.02762026 -0.03037982 -0.02885819
FT   -0.03360846  0.53176378  0.98085089 -0.62359560
X3P   0.07388333 -4.48468100  4.74936258  0.08317034
X3PA  0.45567652  4.37741955 -5.52869531 -0.73048539
TOV   0.55239938 -0.66320513  0.38328592  1.44087962
X2P   0.91032774 -2.82073956 -4.22857099 -0.12732398
PTS   0.97685515  1.10801167  2.99081583  8.78923681
AST   1.44542641 -2.73600432 -1.49756812 -0.47194263
STL   2.11225533  1.47323728  0.33733375 -1.03141294
FGA   2.63724452 -6.16240966  2.50346490 -4.73946998
eFG.  2.65416671 14.35959385 -5.45993833 -19.74833968
```

```
> print(fit.lda$scaling[order(fit.lda$scaling[,2]), ]) #
             LD1         LD2         LD3         LD4
FG.  -0.47644481 -17.07829257  6.63536432  8.82148857
FGA   2.63724452 -6.16240966  2.50346490 -4.73946998
X3P   0.07388333 -4.48468100  4.74936258  0.08317034
X2P   0.91032774 -2.82073956 -4.22857099 -0.12732398
AST   1.44542641 -2.73600432 -1.49756812 -0.47194263
BLK  -1.39655382 -2.06168938  0.53301674  0.96152736
FT.  -0.57090178 -0.94368537 -0.05848004  0.50204063
FTA  -0.53572863 -0.75955042 -1.26510941 -1.90116657
TOV   0.55239938 -0.66320513  0.38328592  1.44087962
X3P. -0.58133964 -0.57776780 -1.98016839  2.98340816
Age  -0.05981743 -0.02762026 -0.03037982 -0.02885819
ORB  -1.27206638  0.11097264 -1.65760514 -2.19717425
PF   -0.38563483  0.23204581  0.33751372 -0.59572019
FT   -0.03360846  0.53176378  0.98085089 -0.62359560
DRB  -1.98943012  0.93775814 -1.47889129  1.77586477
PTS   0.97685515  1.10801167  2.99081583  8.78923681
STL   2.11225533  1.47323728  0.33733375 -1.03141294
X2P. -0.49820809  1.94559310 -2.56525567  1.94299323
X2PA -0.24189945  4.27780861  0.42069909 -0.06307064
X3PA  0.45567652  4.37741955 -5.52869531 -0.73048539
FG   -4.02763606  4.38266888  0.89739751 -2.57128220
eFG.  2.65416671 14.35959385 -5.45993833 -19.74833968
```

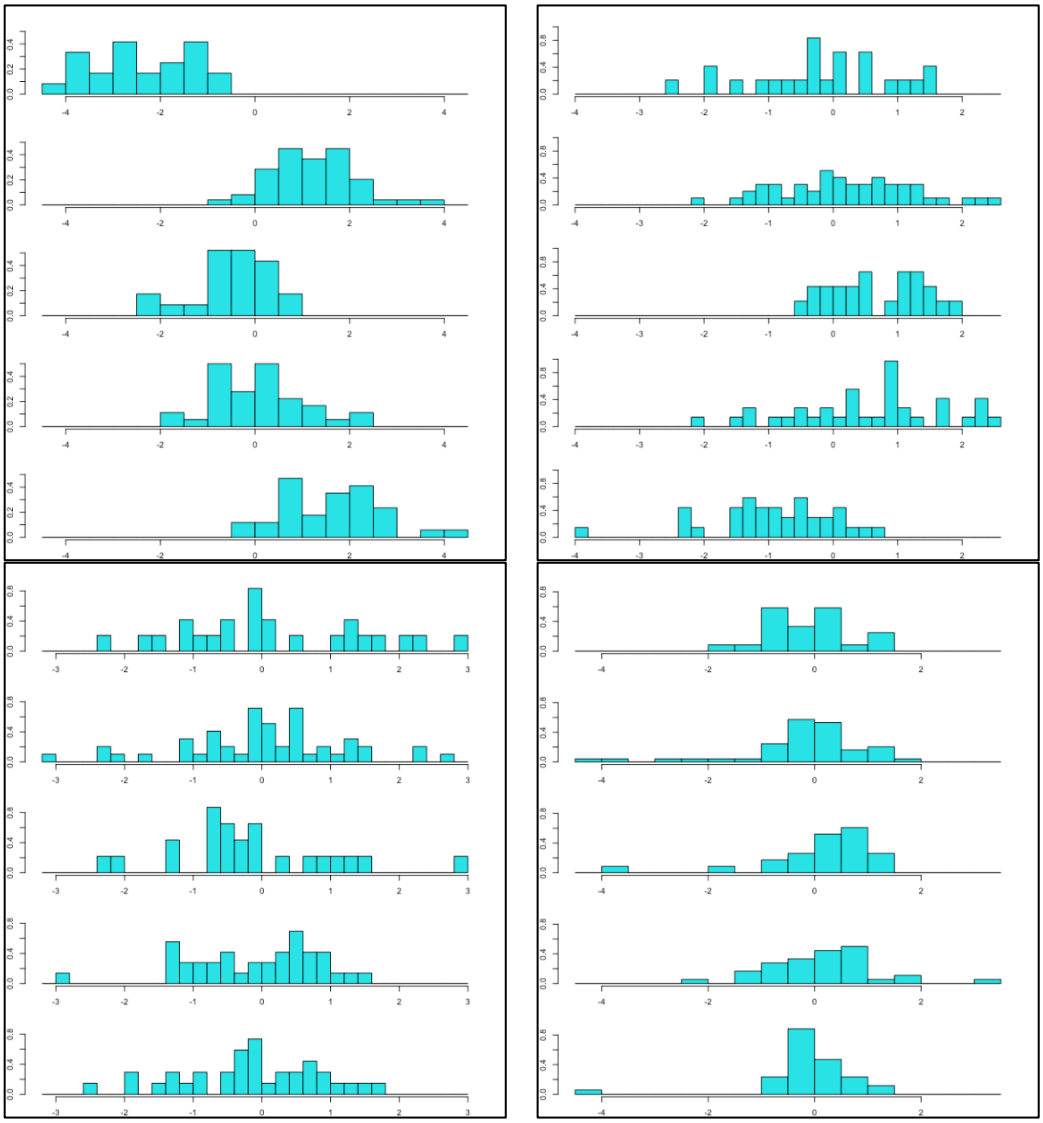
1st discriminate analysis coefficients.



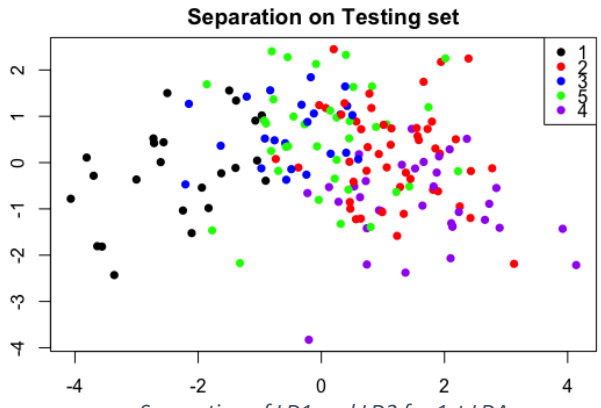
Histogram showing the separation of discriminate components on training set



Separation of LD1 and LD2 for 1st LDA



Histogram showing the separation of discriminate components on testing set of 1st LDA.



Separation of LD1 and LD2 for 1st LDA

```
> confusionMatrix(data = Train.pred$class, reference = as.factor(train$position))
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3  4  5
         1 68  1 11  1  0
         2  2 66 10 24 15
         3 20  6 56 16  2
         4  1 15  6 29  2
         5  0 17  2  7 66

Overall Statistics

               Accuracy : 0.6433
                 95% CI : (0.5968, 0.688)
    No Information Rate : 0.237
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.5519

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity            0.7473   0.6286   0.6588  0.37662   0.7765
Specificity            0.9631   0.8491   0.8771  0.93443   0.9274
Pos Pred Value         0.8395   0.5641   0.5600  0.54717   0.7174
Neg Pred Value         0.9365   0.8804   0.9155  0.87692   0.9459
Prevalence             0.2054   0.2370   0.1919  0.17381   0.1919
Detection Rate         0.1535   0.1490   0.1264  0.06546   0.1490
Detection Prevalence   0.1828   0.2641   0.2257  0.11964   0.2077
Balanced Accuracy      0.8552   0.7388   0.7680  0.65552   0.8519
```

```
> confusionMatrix(data = Test.pred$class, reference = as.factor(test$position))
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3  4  5
         1 15  0  1  2  0
         2  0 19  6 11 11
         3  9  2 12 12  1
         4  0 14  4  8  2
         5  0 14  0  3 20

Overall Statistics

               Accuracy : 0.4458
                 95% CI : (0.3687, 0.5248)
    No Information Rate : 0.2952
    P-Value [Acc > NIR] : 2.836e-05

                  Kappa : 0.2971

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity           0.62500   0.3878  0.52174  0.22222   0.5882
Specificity           0.97887   0.7607  0.83217  0.84615   0.8712
Pos Pred Value        0.83333   0.4043  0.33333  0.28571   0.5405
Neg Pred Value        0.93919   0.7479  0.91538  0.79710   0.8915
Prevalence            0.14458   0.2952  0.13855  0.21687   0.2048
Detection Rate        0.09036   0.1145  0.07229  0.04819   0.1205
Detection Prevalence  0.10843   0.2831  0.21687  0.16867   0.2229
Balanced Accuracy     0.80194   0.5742  0.67695  0.53419   0.7297
```

Confusion matrix of training and testing set of 1st LDA.

2nd Discriminate analysis

```
Call:
lda(Position ~ ., data = train)

Prior probabilities of groups:
        1         2         3
0.2054176 0.3656885 0.4288939

Group means:
       Age       FG      FGA       FG.       X3P      X3PA      X3P.       X2P     X2PA      X2P.      eFG.
1 26.59341 1.285311 1.713311 0.5537143 0.2380573 0.5150656 0.2507692 1.1982538 1.548131 0.5943407 0.5829011
2 25.40741 1.306233 1.888719 0.4671173 0.6146912 1.2020240 0.3260062 1.0409893 1.430292 0.5487531 0.5402284
3 25.72632 1.303608 1.951366 0.4238579 0.7109164 1.3248467 0.3347421 0.9852303 1.434788 0.4739421 0.5040263
        FT       FTA       FT.       ORB       DRB       AST       STL       BLK       TOV       PF       PTS
1 0.7630963 0.9728500 0.6799121 0.9110326 1.401858 0.7095006 0.3474485 0.5131784 0.6498294 1.947253 1.990502
2 0.7086751 0.8478902 0.6992716 0.5922436 1.218586 0.8306811 0.4217816 0.2768820 0.6236099 1.691358 2.066546
3 0.7460473 0.8617558 0.7277316 0.3697611 1.042984 1.1222618 0.4837548 0.1970743 0.7162363 1.488421 2.088688
```

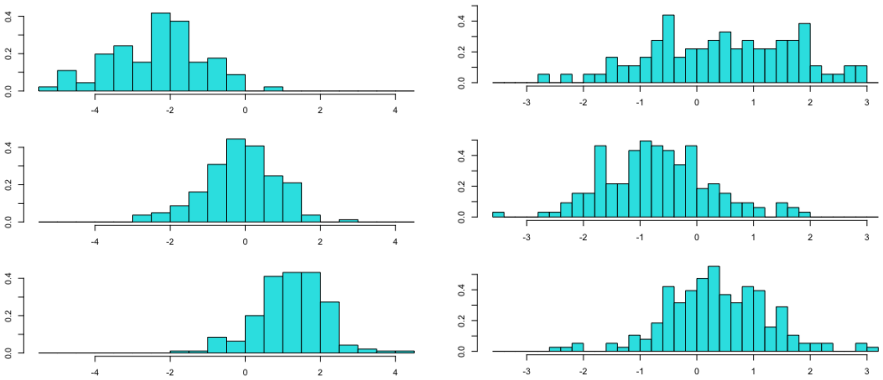
```
Proportion of trace:
   LD1    LD2
0.8644 0.1356
```

2nd LDA results.

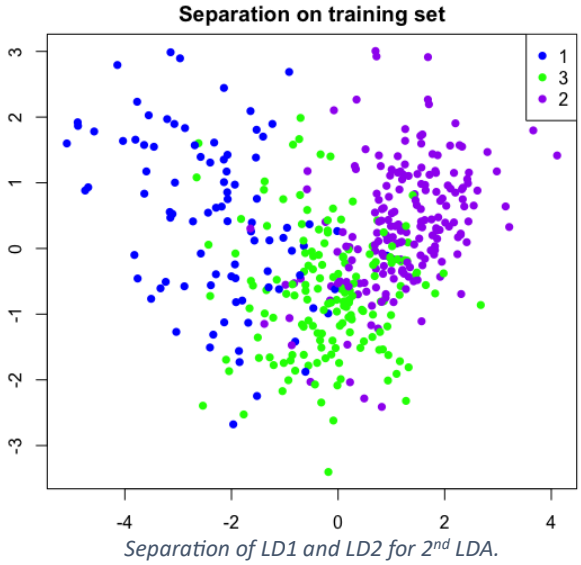
```
> print(fit.lda$scaling[order(fit.lda$scaling[,1]), ])
             LD1          LD2
FG   -3.53430532  -3.599089792
DRB  -1.50456570  -1.413722498
ORB  -1.40905339  -1.215929085
BLK  -1.40098625   2.016121510
FG.  -1.31454442  18.366372551
FTA  -0.79508523  -0.237716456
FT.  -0.56897445   0.779165816
X3P  -0.44903321   6.114663663
PF   -0.40252613  -0.099769654
X3P. -0.22899811  -0.231252856
Age  -0.06131375   0.001909421
FT   -0.05328941  -0.010169760
X2P.  0.00567827  -2.716397703
X2PA  0.23921460  -3.332770847
X2P   0.53938601   0.210948759
TOV   0.60439349   0.916941389
X3PA  0.85229636  -6.467886882
AST   0.98142082   1.507929324
FGA   1.16316191   5.953126376
eFG.  1.70528921 -16.603662157
STL   1.98019763  -1.068625430
PTS   2.05005848   1.574755947
```

```
> print(fit.lda$scaling[order(fit.lda$scaling[,2]), ])
             LD1          LD2
eFG.  1.70528921 -16.603662157
X3PA  0.85229636  -6.467886882
FG   -3.53430532  -3.599089792
X2PA  0.23921460  -3.332770847
X2P.  0.00567827  -2.716397703
DRB  -1.50456570  -1.413722498
ORB  -1.40905339  -1.215929085
STL   1.98019763  -1.068625430
FTA  -0.79508523  -0.237716456
X3P. -0.22899811  -0.231252856
PF   -0.40252613  -0.099769654
FT   -0.05328941  -0.010169760
Age  -0.06131375   0.001909421
X2P   0.53938601   0.210948759
FT.  -0.56897445   0.779165816
TOV   0.60439349   0.916941389
AST   0.98142082   1.507929324
PTS   2.05005848   1.574755947
BLK  -1.40098625   2.016121510
FGA   1.16316191   5.953126376
X3P  -0.44903321   6.114663663
FG.  -1.31454442  18.366372551
```

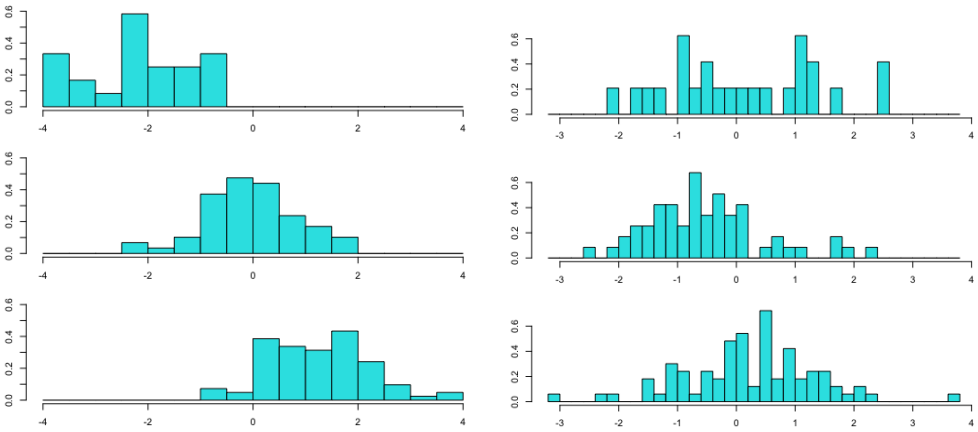
2nd discriminate analysis coefficients.



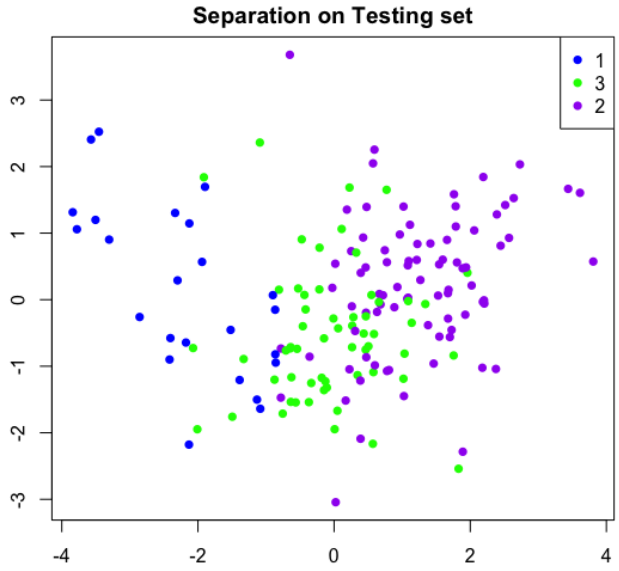
Histogram showing the separation of discriminate components on training set in 2nd LDA.



Separation of LD1 and LD2 for 2nd LDA.



Histogram showing the separation of discriminate components on testing set in 2nd LDA



Separation of LD1 and LD2 for 2nd LDA.

```
> confusionMatrix(data = Train.pred$class, reference = as.factor(train$Position))
Confusion Matrix and Statistics

          Reference
Prediction   1   2   3
         1  68  11   1
         2  22 119  34
         3   1  32 155

Overall Statistics

               Accuracy : 0.772
                 95% CI : (0.7301, 0.8103)
    No Information Rate : 0.4289
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.6418

 Mcnemar's Test P-Value : 0.2925

Statistics by Class:

                     Class: 1 Class: 2 Class: 3
Sensitivity            0.7473   0.7346   0.8158
Specificity            0.9659   0.8007   0.8696
Pos Pred Value         0.8500   0.6800   0.8245
Neg Pred Value         0.9366   0.8396   0.8627
Prevalence             0.2054   0.3657   0.4289
Detection Rate         0.1535   0.2686   0.3499
Detection Prevalence   0.1806   0.3950   0.4244
Balanced Accuracy      0.8566   0.7676   0.8427
```

```
> confusionMatrix(data = Test.pred$class, reference = as.factor(test$Position))
Confusion Matrix and Statistics

          Reference
Prediction  1  2  3
         1 15  3  0
         2  9 43 17
         3  0 13 66

Overall Statistics

               Accuracy : 0.747
                 95% CI : (0.6738, 0.8112)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : 6.798e-11

                  Kappa : 0.5774

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3
Sensitivity           0.62500   0.7288   0.7952
Specificity           0.97887   0.7570   0.8434
Pos Pred Value        0.83333   0.6232   0.8354
Neg Pred Value        0.93919   0.8351   0.8046
Prevalence            0.14458   0.3554   0.5000
Detection Rate        0.09036   0.2590   0.3976
Detection Prevalence  0.10843   0.4157   0.4759
Balanced Accuracy     0.80194   0.7429   0.8193
```

Confusion matrix of training and testing set of 2nd LDA.

```
Call:
glm(formula = position ~ ., family = "binomial", data = train.f)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.21361  -0.81393  -0.01189   0.89084   2.19622

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.27243    2.82833  -0.096 0.923264
Age           0.09445    0.05589   1.690 0.091021 .
FG           16.33606   12.85329   1.271 0.203742
FGA         -11.81723    7.10689  -1.663 0.096356 .
FG.          -1.06649   31.20003  -0.034 0.972732
X3P          -9.43145    5.64060  -1.672 0.094512 .
X3PA          4.48239    4.35531   1.029 0.303396
X3P.          9.45963    5.79690   1.632 0.102712
X2P          -6.78016    9.37467  -0.723 0.469531
X2PA          1.84158    5.85010   0.315 0.752918
X2P.         11.69439    8.34016   1.402 0.160862
eFG.        -22.20160   28.17600  -0.788 0.430719
FT           -0.63489    3.98715  -0.159 0.873485
FTA           0.27454    3.74143   0.073 0.941504
FT.           1.29130    1.45811   0.886 0.375833
ORB           0.37939    1.16216   0.326 0.744084
DRB           4.33141    1.30743   3.313 0.000923 ***
AST          -2.49757    1.18367  -2.110 0.034856 *
STL          -2.29778    1.50490  -1.527 0.126796
BLK           2.74329    1.49083   1.840 0.065753 .
TOV           1.05011    1.90760   0.550 0.581986
PF           -0.20001    0.47534  -0.421 0.673926
PTS           1.14108    6.78159   0.168 0.866378
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 228.73  on 164  degrees of freedom
Residual deviance: 161.94  on 142  degrees of freedom
AIC: 207.94

Number of Fisher Scoring iterations: 6
```

Logistic regression for forward dataset

```
Call:
glm(formula = position ~ ., family = "binomial", data = train.g)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6759  -0.7504  -0.2438   0.7257   2.5946

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.20877    2.84413  -0.073   0.9415
Age           0.05309    0.05139   1.033   0.3016
FG           12.30114    9.54559   1.289   0.1975
FGA          -0.49276    5.94088  -0.083   0.9339
FG.          47.68368   29.32200   1.626   0.1039
X3P          -3.29641    4.21062  -0.783   0.4337
X3PA          3.69539    3.33862   1.107   0.2684
X3P.          2.63807    4.33342   0.609   0.5427
X2P          -6.07374    6.15058  -0.988   0.3234
X2PA         -1.75461    4.22291  -0.415   0.6778
X2P.         -6.21846    4.52347  -1.375   0.1692
eFG.        -37.49573   23.41088  -1.602   0.1092
FT           -3.12461    3.70193  -0.844   0.3986
FTA           4.51287    3.35708   1.344   0.1789
FT.           1.98096    1.47741   1.341   0.1800
ORB          -0.47056    1.25347  -0.375   0.7074
DRB          -2.40557    1.17878  -2.041   0.0413 *
AST           6.07493    1.11939   5.427 5.73e-08 ***
STL           2.37523    1.46801   1.618   0.1057
BLK           0.21079    1.86825   0.113   0.9102
TOV          -0.20618    1.21610  -0.170   0.8654
PF           -0.77636    0.52988  -1.465   0.1429
PTS          -8.21002    3.82472  -2.147   0.0318 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 268.62  on 197  degrees of freedom
Residual deviance: 174.24  on 175  degrees of freedom
AIC: 220.24

Number of Fisher Scoring iterations: 5
```

Logistic regression for guard dataset

```
Call:
glm(formula = position ~ Age + X3P + DRB + AST, family = "binomial",
    data = train.f)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8701  -1.0026  -0.2550   0.9737   2.0151

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.53016    1.20082  -2.940 0.003284 **
Age          0.08203    0.04450   1.844 0.065253 .
X3P         -2.12373    0.63589  -3.340 0.000838 ***
DRB          3.94956    0.80831   4.886 1.03e-06 ***
AST         -2.43524    0.69779  -3.490 0.000483 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 228.73  on 164  degrees of freedom
Residual deviance: 188.97  on 160  degrees of freedom
AIC: 198.97

Number of Fisher Scoring iterations: 4
```

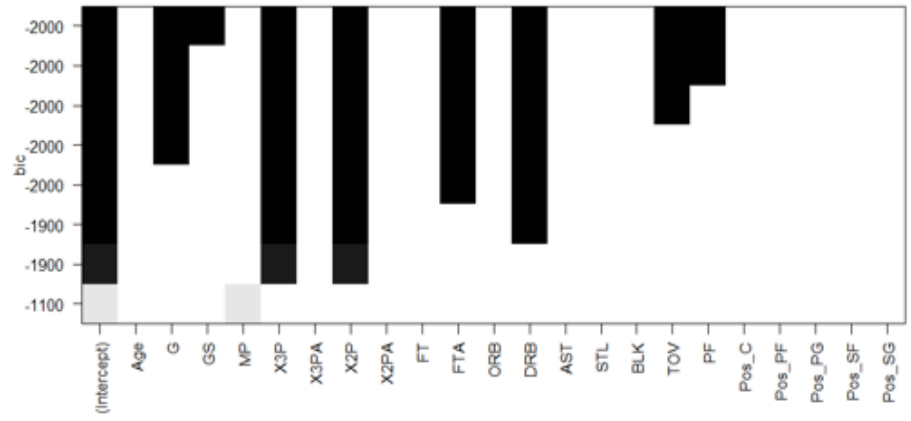
Final logistic regression model for forward dataset

```
Call:
glm(formula = position ~ DRB + AST + PTS, family = "binomial",
    data = train.g)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6050  -0.7170  -0.3199   0.7461   2.5219

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.4033     0.5312   0.759  0.44778
DRB          -2.4815     0.8739  -2.839  0.00452 **
AST           5.6648     0.8356   6.779 1.21e-11 ***
PTS          -2.2801     0.5469  -4.169 3.06e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 268.62  on 197  degrees of freedom
Residual deviance: 186.25  on 194  degrees of freedom
AIC: 194.25

Number of Fisher Scoring iterations: 5
```

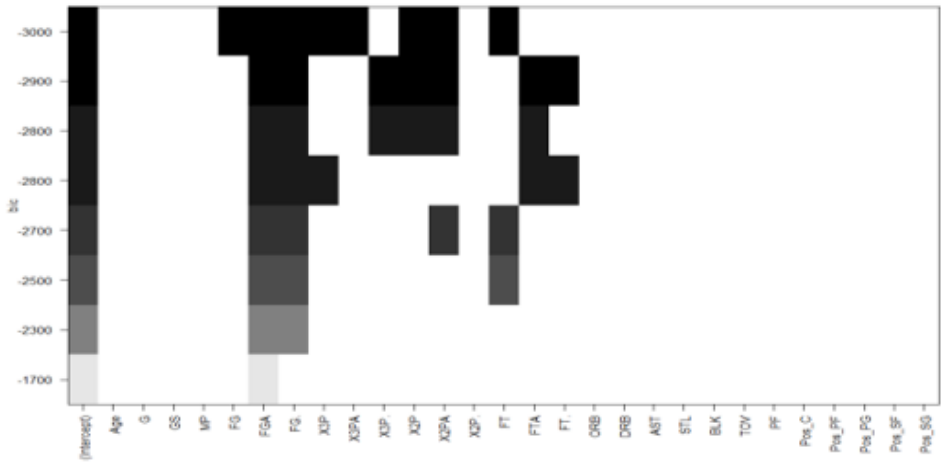
Final Logistic regression model for guard dataset

```
> #prediction for forward set
> fitted.results.f <- predict(f.model1,test.f, type='response')
> fitted.results.f <- ifelse(fitted.results.f > 0.5,1,0)
> misClasificError <- mean(fitted.results.f != test.f$position)
> print(paste('Accuracy',1-misClasificError))
[1] "Accuracy 0.660714285714286"
> #prediction for guard set
> fitted.results.g <- predict(g.model1, test.g, type='response')
> fitted.results.g <- ifelse(fitted.results.g > 0.5,1,0)
> misClasificError <- mean(fitted.results.g != test.g$position)
> print(paste('Accuracy',1-misClasificError))
[1] "Accuracy 0.76"
```

All Subset Analysis Output:



BIC Plot for player set.



BIC Plot for team set

```
> reg_team5 = lm(PTS ~ FG. + X3P + X2PA + FT, data = nba_teamstat3)
> summary(reg_team5)

Call:
lm(formula = PTS ~ FG. + X3P + X2PA + FT, data = nba_teamstat3)

Residuals:
     Min       1Q   Median       3Q      Max
-0.76737 -0.06075  0.01066  0.07374  0.64075

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.15917    0.02541   6.263 7.16e-10 ***
FG.          1.04085    0.05134  20.274  < 2e-16 ***
X3P          0.70901    0.01618  43.821  < 2e-16 ***
X2PA         0.55360    0.01741  31.791  < 2e-16 ***
FT           0.27188    0.02189  12.421  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1373 on 604 degrees of freedom
Multiple R-squared:  0.9625,    Adjusted R-squared:  0.9622
F-statistic:  3870 on 4 and 604 DF,  p-value: < 2.2e-16

> vif(reg_team5)
     FG.      X3P     X2PA       FT
1.231692 1.464042 3.930580 3.940987
```

Model For team set

```
> reg_player1 = lm(PTS ~ G + GS + X3P + X2P + FTA + DRB + TOV + PF, data = nba_playerstat3)
> summary(reg_player1)

Call:
lm(formula = PTS ~ G + GS + X3P + X2P + FTA + DRB + TOV + PF,
    data = nba_playerstat3)

Residuals:
     Min       1Q   Median       3Q      Max
-0.70240 -0.05931  0.00823  0.07745  0.33662

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.5503432  0.0168109  32.737  < 2e-16 ***
G            0.0017326  0.0003074   5.637 2.67e-08 ***
GS          -0.0229589  0.0062424  -3.678 0.000256 ***
X3P          0.6970232  0.0158277  44.038  < 2e-16 ***
X2P          0.7353393  0.0245622  29.938  < 2e-16 ***
FTA          0.1910758  0.0214622   8.903  < 2e-16 ***
DRB          0.1255805  0.0217221   5.781 1.19e-08 ***
TOV         -0.1155484  0.0287092  -4.025 6.43e-05 ***
PF           0.0405303  0.0110145   3.680 0.000254 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1301 on 600 degrees of freedom
Multiple R-squared:  0.9665,    Adjusted R-squared:  0.966
F-statistic:  2163 on 8 and 600 DF,  p-value: < 2.2e-16

> vif(reg_player1)
       G       GS      X3P      X2P      FTA      DRB      TOV       PF
2.134421 3.684532 1.559632 6.409905 4.865256 3.738321 3.891438 2.718180
```

Model For player set

R Scripts:

Regularized Regression (Relaxed Lasso) Script:

```
nba1 <- read.csv ("2022-2023 NBA Player Stats - Regular.csv", sep=";")
head(nba1)
library(tidyverse)
library(corrplot)
library(psych)
library(ggplot2)
library(glmnet)
library(leaps)
library(GGally)
library(car)

#let's view the structure of the dataset
str(nba1)
nba1

summary(nba1)
colSums(is.na(nba1))
num_teams <- nba1 %>% select (Tm) %>% n_distinct() #finding the number of teams
num_teams   #There are 31 teams

#Checking for duplicate rows
duplicates_df <- nba1[duplicate_values, ]
print(duplicates_df) #140 obs

#These players seem to have played in different teams in the season,
#so we will keep the stats. we will however remove the rows called total.

#drop duplicate rows TOT
nba3 <- nba1[nba1$Tm!= "TOT",]

#Creating dummy variables for Position to ease correlation plots
nba_new <- nba3 %>%
  mutate(Position_dummy = ifelse(Pos == "PG", 1,
                          ifelse(Pos == "SG", 2,
                               ifelse(Pos == "SF", 3,
                                    ifelse(Pos== "PF", 4, 5)))))

head(nba_new)

# Compute the correlation matrix
cor_matrix <- cor(nba_new[, -c(1,2,3,5)]) #excluding categorical with original position variable
cor_matrix

#creatring plots to examine the data
pairs.panels(nba_new[,c (-1, -2, -3, -5)])

corrplot(cor_matrix,)
#From the pairs.panels plot
#age is slightly skewed, FG is slightly left skewed and very highly correlated with PTS
#FGA is sligthl left skewed, X3P, X3PA, X2P, X2PA and X2P% are left skewed.
#eFG is fairly normal
#FT is left skewed, FTA is right skewed, CRB is left skewed, DRB is left skewed, TRB is right skewed
#AST is left skewed, STL is left skewed, BL is left skewed, TOR is left skewed, PT is normal PTS is left skewed.
 A+1)
nba_new$X3P. = log(nba_new$X3P.+1)
nba_new$X2P = log(nba_new$X2P+1)
```

```
nba_new$X2PA = log(nba_new$X2PA+1)
nba_new$X2P. = log(nba_new$X2P.+1)
nba_new$eFG. = log(nba_new$eFG. +1)
nba_new$FT = log(nba_new$FT+1)
nba_new$FTA = log(nba_new$FTA+1)
nba_new$FT. = log(nba_new$FT.+1)
nba_new$ORB = log(nba_new$ORB+1)
nba_new$DRB = log(nba_new$DRB+1)
nba_new$TRB = log(nba_new$TRB+1)
nba_new$AST = log(nba_new$AST+1)
nba_new$STL = log(nba_new$STL+1)
nba_new$BLK = log(nba_new$BLK+1)
nba_new$TOV = log(nba_new$TOV+1)
nba_new$PTS = log(nba_new$PTS+1)
pairs.panels(nba_new[,c(-1, -2, -3, -5)])
#let's check the new distributions
P1 <- ggplot(nba_new, aes(x= GS)) + geom_histogram()
P2 <- ggplot(nba_new, aes(x= FG)) + geom_histogram()
P3 <- ggplot(nba_new, aes(x= FGA)) + geom_histogram()
P4 <- ggplot(nba_new, aes(x= X3P)) + geom_histogram()
P5 <- ggplot(nba_new, aes(x= X3P.)) + geom_histogram()
P6 <- ggplot(nba_new, aes(x= X2P)) + geom_histogram()
P7 <- ggplot(nba_new, aes(x= eFG.)) + geom_histogram()
P8 <- ggplot(nba_new, aes(x= FT)) + geom_histogram()
P9 <- ggplot(nba_new, aes(x= FTA)) + geom_histogram()
P10 <- ggplot(nba_new, aes(x= ORB)) + geom_histogram()
P11 <- ggplot(nba_new, aes(x= FT.)) + geom_histogram()
P12 <- ggplot(nba_new, aes(x= DRB)) + geom_histogram()
P13 <- ggplot(nba_new, aes(x= TRB)) + geom_histogram()
P14 <- ggplot(nba_new, aes(x= AST)) + geom_histogram()
P15 <- ggplot(nba_new, aes(x= STL)) + geom_histogram(
#Lets perform log transforms on them.
nba_new$GS = log(nba_new$GS+1)
nba_new$FG = log(nba_new$FG+1)
nba_new$FGA = log(nba_new$FGA+1)
nba_new$X3P = log(nba_new$X3P+1)
nba_new$X3PA = log(nba_new$X3P)
P16 <- ggplot(nba_new, aes(x= BLK)) + geom_histogram()
P17 <- ggplot(nba_new, aes(x= TOV)) + geom_histogram()
P18 <- ggplot(nba_new, aes(x= PTS)) + geom_histogram()
library(gridExtra)
grid.arrange(P1,P2,P3,P4,P5,P6,P7,P8,P9,P10,P11,P12,P13,P14,P15,P16,P17,P18, nrow = 4
cor_matrix2 <- cor(nba_new[, -c(1,2,3,5)]) #doing new correlation matrix and plot
cor_matrix2
corrplot(cor_matrix2)
head(nba_new)
class(nba_new)
nba_new1 <- nba_new[, -c (1, 2, 3, 5)] #changed from 1, 2, 5, to 1, 2, 3 ,5
nba_new1
nrow(nba_new1)
#Aliasing Error
fitAlias = lm(PTS~., data=nba_new1)
vif(fitAlias)
#EXTRA STEP: divide set into individual and team stats.
#Pull out team stats
playerStats <- nba_new[c(4, 6:8, 12:13, 15:16, 19:20, 22:23, 25:31)]
teamStats = nba_new[-c(1,2,3,5,9.18,24)]
head(playerStats)
```

```
summary(playerStats)
head(teamStats)
summary(teamStats)

#Using playerStats for analysis
set.seed(708)
nrow(playerStats)
#90 10 split or 80 20 split
s = sample(609, 539)
s = sample(609, 487)
dsTrain = playerStats[s, ]
dsTest = playerStats[-s, ]

fit1 = lm(PTS ~., data=playerStats)
summary(fit1)
vif(fit1)
#Overfitting & Multicollinearity Present
#Multicollinearity Check
#theory behind multicollinearity elimination:
#remove "partial variables" and model around "complete variables"
#ex: remove 2p/3p per game + attempts per game and use percentage (removing 2 collinear variables)
dsTrain = dsTrain[, -c(4, 5, 7, 10, 11, 16)]
#EXPERIMENTAL: REMOVING GAMES PLAYED
dsTrain = dsTrain[, -c(2, 4, 5, 7, 10, 11, 16)]
fit1 = lm(PTS ~., data=dsTrain)
summary(fit1)
vif(fit1)
#all that's left is minutes played per game

#remove the same variables from test
dsTest = dsTest[, -c (4, 5, 7, 10, 11, 16)]
#EXPERIMENTAL: REMOVING GAMES PLAYED
dsTest = dsTest[, -c (2, 4, 5, 7, 10, 11, 16)]
#Checking datasets
dsTrain
dsTest
#they have the same variables.
#x & y splits
#playerStats FULL MODEL
yTrain = as.matrix(playerStats[,18])
xTrain = as.matrix(playerStats[,-18])
##
yTrain = as.matrix(dsTrain[,18])
xTrain = as.matrix(dsTrain[,-18])
yTest = as.matrix(dsTest[,18])
xTest = as.matrix(dsTest[,-18])
#Splits with limited predictors
#train
#11 if games played removed, 12 if not
yTrain = as.matrix(dsTrain[, 11])
yTrain #PTS variable
xTrain = as.matrix(dsTrain[, -11])
xTrain
#test
yTest = as.matrix(dsTest[, 11])
yTest
xTest = as.matrix(dsTest[, -11])
xTest
```

```
CompleteCases <- complete.cases(xTrain, yTrain)
CompleteCases
#Let me make an OLS model attempt
fit2Ols = lm(PTS~., data=dsTest)
summary(fit2Ols)
rmse_ols_train = sqrt(mean(fit2Ols$residuals^2))
rmse_ols_train
#There still seems to be some inflation.
#glmnet call, look at previous lectures before continuing
set.seed(918)
lRange = seq (0,5,.1)
fitRange = glmnet(xTrain, yTrain, data=dsTrain, lambda=lRange)
#Experimental
#Remove 2 point and 3 points
playerStats = playerStats[, -c(5,7)]
xTrain = xTrain[, -c(5,7)]
xTest = xTest[,-c (5,7)]

fitRange = cv.glmnet(xTrain, yTrain, data=playerStats, alpha=1, nfolds=3)
plot(fitRange)
summary(fitRange)
coef(fitRange, s="lambda.1se")
fitRange$lambda.1se

glmnet(xTrain, yTrain, data=playerStats, lambda=lRange)
fit1se = glmnet(xTrain, yTrain, data=playerStats, lambda ="lambda.1se")
####
summary(fitRange)
plot(fitRange)
fitLasso = cv.glmnet(xTrain, yTrain, relax=T, data=dsTrain)
print(fitLasso)
summary(fitLasso)
plot(fitLasso)
coef(fitLasso, s="lambda.min", gamma=0)
coef(fitLasso, s="lambda.1se", gamma=0)
coef(fitLasso, s="lambda.1se", gamma=1)
fitLasso$lambda.1se
fit1se = glmnet(xTrain, yTrain, data=dsTrain, lambda = 0.04754607)
glmnet(xTrain, yTrain, data=dsTrain, lambda = 0.04754607)
coef(fit1se)
#how do we get the dev%?
pLassoTrain = predict (fit1se, xTrain, s="lambda.1se")
rmse_lasso_train = sqrt (mean ((pLassoTrain - yTrain)^2))
rmse_lasso_train  # 0.18 with set seed

#glmnet with lambda
glmnet(xTrain, yTrain, data=dsTrain, relax=T, lambda=0.04754607)
fitLasso2 = glmnet(xTrain, yTrain, data=dsTrain, relax=T, lambda = 0.04653347)
summary(fitLasso2)
coef(fitLasso2, s="lambda.1se")
#RMSE
pLassoTrain2 = predict(fitLasso2, xTrain, s="lambda.1se")
rmse_lasso_train2 = sqrt(mean((pLassoTrain2 - yTrain)^2))
rmse_lasso_train2  #0.18

#Victor's RMSE code
#Checking
#predicting train set at lambda = 1se and RMSE of train set
```

```
#Used experimental
lassopredtr = predict(fit1se, xTrain, s="lambda.1se")
rmselassopredtr = sqrt(mean((lassopredtr - yTrain)^2))
rmselassopredtr

#predicting test set at lambda = 1se and RMSE of test set
lassopred = predict(fit1se, xTest, s="lambda.1se")
rmselassortest = sqrt(mean((lassopred - yTest)^2))
rmselassortest
#Ratio of RMSE of Train to test in Lasso cv model
rmselassortest / rmselassopredtr

#Alaa's Residual Analysis Code
#Residuals vs. Fitted
plot(fit3Ols, which=1)
#Normality
plot(fit3Ols, which=2)
#Residuals vs. Cook's Distance
plot(fit3Ols, which=4)
```

All Subset Code Script:

```
library("MASS")
library("caret")
library("glmnet")
library("leaps")
library("ggplot2")
library("GGally")
library("glmnet")
library("corrplot")
library("foreign")
library("QuantPsyc")
library("psych")
library("stats")
library("lavaan")
library("semPlot")
library("polycor")
library("ordPens")
library("ordinal")
library("GPArotation")
library("clusterGeneration")
library("tidyverse")
library("caret")
library("fastDummies")
library("dplyr")
library("ggplot2")
library("ca")

project_ds <- read.csv ("C:/Users/Anirudh/Desktop/All Quarters/Quarter 3/Advanced Data Analysis/Project/2022-2023 NBA Player Stats - Regular.csv", sep=";")
head(project_ds)
summary(project_ds)

colSums(is.na(project_ds))

###Cleaning
##Drop duplicates
```

```
project_dsnew <- project_ds[project_ds$Tm != "TOT", ]

library(dplyr)
project_dsnew1 <- project_dsnew %>%
  mutate_at(vars (GS), ~log (1 +.)) %>% mutate_at(vars (X2P), ~log (1 +.)) %>%
  mutate_at(vars (FGA), ~log (1 +.)) %>%
  mutate_at(vars(X3P), ~log (1 +.)) %>% mutate_at(vars(X3PA), ~log (1 +.)) %>%
  mutate_at(vars(X2PA), ~log (1 + .)) %>% mutate_at(vars (FT), ~log (1 +.)) %>%
  mutate_at(vars (FTA), ~log (1 +.)) %>% mutate_at(vars (ORB), ~log (1 +.)) %>%
  mutate_at(vars (DRB), ~log (1 +.)) %>% mutate_at(vars (AST), ~log (1 +.)) %>%
  mutate_at(vars (STL), ~log (1 +.)) %>% mutate_at(vars (BLK), ~log (1 +.)) %>%
  mutate_at(vars (TOV), ~log (1 +.)) %>% mutate_at(vars (PTS), ~log (1 +.))

pairs.panels(project_dsnew1[,c(-1)]) #scatterplot

#we'll divide the dataset into team and player stats

nba_playerstat <- project_dsnew1[c (3:4, 6:8, 12:13, 15:16, 19:20,22:23, 25:30)]
nba_teamstat <- project_dsnew1[ -c (1, 2, 5, 18, 24)]
head(nba_teamstat)
head(nba_playerstat)

#Create dummy
library(fastDummies)

nba_teamstat2 <- dummy_cols(nba_teamstat, select_columns = "Pos")
head(nba_teamstat2)

nba_playerstat2 <- dummy_cols(nba_playerstat, select_columns = "Pos")
head(nba_playerstat2)



nba_playerstat3 <- nba_playerstat2[-c(1)]
head(nba_playerstat3)

nba_teamstat3 <- nba_teamstat2[-c(1)]
head(nba_teamstat3)



# Now, try all-subsets

#PLAYERSTATS

fit1_all = regsubsets(PTS ~., data = nba_playerstat3)
summary(fit1_all)

plot (fit1_all, scale="bic")   # Interesting, chooses only 4 at top
#G, GS, X3P, X2P, FTA, DRB, TV, PF
library("car")

reg_player = lm(PTS ~ G + GS + X3P + X2P + X2PA + FTA + DRB + TOV + PF, data = nba_playerstat3)
summary(reg_player)
vif(reg_player)
```

```
reg_player1 = lm(PTS ~ G + GS + X3P + X2P + FTA + DRB + TOV + PF, data = nba_playerstat3)
summary(reg_player1)
vif(reg_player1)

# Obtain predicted values and residuals for playerstat
predictedplayer <- fitted(reg_player1)
residualsplayer <- residuals(reg_player1)

# Residual plots
# Scatterplot of residuals against predicted values
plot (predictedplayer, residualsplayer, xlab = "Predicted Values", ylab = "Residuals", main = "Residual Plot")

# Histogram of residuals
hist (residualsplayer, main = "Histogram of Residuals")

# Q-Q plot of residuals
qqPlot(residualsplayer, main = "Q-Q Plot of Residuals")

#teamstats
fit2_all = regsubsets(PTS ~., data = nba_teamstat3)
summary(fit2_all)
plot (fit2_all, scale="adjr2") # Many contributors & no good way to decide!  R^2 drops evenly
plot (fit2_all, scale="Cp")    # AIC --> No good way to decide!  R^2 drops evenly
plot (fit2_all, scale="bic")   # Interesting, chooses only 4 at top
#fg, fga, fg., x2p, x2pa, ft

reg_team = lm(PTS ~ FG + FGA + FG. + X3P + X3PA + X2P + X2PA + FT, data = nba_teamstat3)
summary(reg_team)
vif(reg_team)

reg_team2 = lm(PTS ~ FG + FGA + FG. + X3P + X3PA + X2PA + FT, data = nba_teamstat3)
summary(reg_team)
vif(reg_team2)

reg_team3 = lm(PTS ~ FG + FG. + X3P + X3PA + X2PA + FT, data = nba_teamstat3)
summary(reg_team3)
vif(reg_team3)

reg_team4 = lm(PTS ~ FG + FG. + X3P + X2PA + FT, data = nba_teamstat3)
summary(reg_team4)
vif(reg_team4)

reg_team5 = lm(PTS ~ FG. + X3P + X2PA + FT, data = nba_teamstat3)
summary(reg_team5)
vif(reg_team5)

# Obtain predicted values and residuals
predictedteam <- fitted(reg_team5)
residualsteam <- residuals(reg_team5)

# Residual plots
# Scatterplot of residuals against predicted values
plot (predictedteam, residualsteam, xlab = "Predicted Values", ylab = "Residuals", main = "Residual Plot")
```

```
# Histogram of residuals
hist (residualsteam, main = "Histogram of Residuals")

# Q-Q plot of residuals
qqPlot(residualsteam, main = "Q-Q Plot of Residuals")
```

LDA Script:

```
###Import data
ds <- read.csv("~/Desktop/Final Project DSC 424/2022-2023 NBA Player Stats - Regular (2).csv", sep=";")

###Cleaning
##Drop duplicates
dsnew <- ds[ds$Tm != "TOT", ]
dsnew <- dsnew[,-c(1,2,5)] # drop Rk, Player & Tm

##Transformation
library(dplyr)
new_df <- dsnew %>%
  mutate_at(vars(GS), ~log(1 + .)) %>% mutate_at(vars(X2P ), ~log(1 + .)) %>%
  mutate_at(vars(FG), ~log(1 + .)) %>% mutate_at(vars(FGA), ~log(1 + .)) %>%
  mutate_at(vars(X3P), ~log(1 + .)) %>% mutate_at(vars(X3PA), ~log(1 + .)) %>%
  mutate_at(vars(X2PA), ~log(1 + .)) %>% mutate_at(vars(FT), ~log(1 + .)) %>%
  mutate_at(vars(FTA), ~log(1 + .)) %>% mutate_at(vars(ORB), ~log(1 + .)) %>%
  mutate_at(vars(DRB), ~log(1 + .)) %>% mutate_at(vars(AST), ~log(1 + .)) %>%
  mutate_at(vars(STL), ~log(1 + .)) %>% mutate_at(vars(BLK), ~log(1 + .)) %>%
  mutate_at(vars(TOV), ~log(1 + .)) %>% mutate_at(vars(PTS), ~log(1 + .))

# to drop PTS, TRB, X3P., X2P., eFG.
#new_df <- new_df[,-c(11,14,15,21,27)]

# to drop FG PTS, TRB, X3P., X2P., eFG.
#new_df <- new_df[,-c(6,9,12,16,18,21,27)]

# to drop PTS, TRB, X3P., X2P., eFG.
#new_df <- new_df[,-c(6,9,10,12,13,16,18,21)]

names(new_df)
#new_df <- new_df[,-c(3,4,5,8, 11,14, 18,20, 21)]

new_df <- new_df[,-c(3,4,5,21)]

#Create dummy
new_df$position <- ifelse(new_df$Pos == "C", 1,
                          ifelse(new_df$Pos == "SG", 2,
                                 ifelse(new_df$Pos == "PF", 3,
                                        ifelse(new_df$Pos == "SF", 4, 5))))
#Splitting data
set.seed(1233)
sample <- sample(c(TRUE, FALSE), nrow(new_df), replace=TRUE, prob=c(0.7,0.3))
train1  <- new_df[sample, ]
test1   <- new_df[!sample, ]

train = train1[,-1]
test = test1[,-1]

#### 1st Linear Discriminate Analysis ###
```

```
library(MASS)
#lda on training set
fit.lda = lda(position ~ ., data=train)
print(fit.lda)
#to get the (scaling) results of original variables which is the (loading)
print(fit.lda$scaling[order(fit.lda$scaling[,1]), ])
print(fit.lda$scaling[order(fit.lda$scaling[,2]), ]) # to reorder LD2
#Predict on training set to see how much separation we can achieve on training set
Train.pred = predict(fit.lda)
#scores result
scores = Train.pred$x
scores
#Visualize the separation
#Separation on histogram
par(mar=c(2,2,2,2))
ldahist(data=Train.pred$x[,1], g=train$position)
ldahist(data=Train.pred$x[,2], g=train$position)
ldahist(data=Train.pred$x[,3], g=train$position)
ldahist(data=Train.pred$x[,4], g=train$position)
#plot
plot(Train.pred$x[,1], Train.pred$x[,2], col=c("black","red","blue","green", "purple")[train$position]
    , pch=16, main = "Separation on training set")

plot(Train.pred$x[,3], Train.pred$x[,4], col=c("black","red","blue","green", "purple")[train$position]
    , pch=16, main = "Separation on training set")

# Create a color vector
color_vec <- c("black","red","blue","green", "purple")
# Add a legend
legend("topright", legend = unique(train$position), col = unique(color_vec), pch = 16)

###### lda on testing set ######
#predict on test set
Test.pred = predict(fit.lda, test)
Test.pred$x  # score
ldahist (data = Test.pred$x[,1], g=test$position)
ldahist (data = Test.pred$x[,2], g=test$position)
ldahist (data = Test.pred$x[,3], g=test$position)
ldahist (data = Test.pred$x[,4], g=test$position)

plot(Test.pred$x[,1], Test.pred$x[,2], col=c("black","red","blue","green", "purple")[test$position], pch=16,
    main = "Separation on Testing set")
plot(Test.pred$x[,3], Test.pred$x[,4], col=c("black","red","blue","green", "purple")[test$position], pch=16,
    main = "Separation on Testing set")

# Add a legend
legend("topright", legend = unique(train$position), col = unique(color_vec), pch = 16)

#confusion matrix
library(caret)
confusionMatrix(data = Train.pred$class, reference = as.factor(train$position))
confusionMatrix(data = Test.pred$class, reference = as.factor(test$position))

#------------------------------------------------------------------------------

### 2nd Linear Discriminate Analysis ###

#combine F and G postions
```

```
new_df1 <- mutate(new_df, position = ifelse(Pos %in% c("PF", "SF"), "F",
                                  ifelse(Pos %in% c("SG", "PG"), "G", Pos)))

#Create dummy
new_df1$Position <- ifelse(new_df1$position == "C", 1,
                    ifelse(new_df1$position == "F", 2,3))

#Splitting data
set.seed(1233)
sample <- sample(c(TRUE, FALSE), nrow(new_df1), replace=TRUE, prob=c(0.7,0.3))
train1  <- new_df1[sample, ]
test1   <- new_df1[!sample, ]

train = train1[,-c(1,24)]
test = test1[,-c(1,24)]

#### Linear Discriminate Analysis ###
#lda on training set
fit.lda = lda(Position ~ ., data=train)
print(fit.lda)
#to get the (scaling) results of original variables which is the (loading)
print(fit.lda$scaling[order(fit.lda$scaling[,1]), ])
print(fit.lda$scaling[order(fit.lda$scaling[,2]), ]) # to reorder LD2
#Predict on training set to see how much separation we can achieve on training set
Train.pred = predict(fit.lda)

#Visualize the separation
#Separation on histogram
ldahist(data=Train.pred$x[,1], g=train$Position)
ldahist(data=Train.pred$x[,2], g=train$Position)
#plot
plot(Train.pred$x[,1], Train.pred$x[,2], col=c("blue","green", "purple")[train$Position]
    , pch=16, main = "Separation on training set")

# Create a color vector
color_vec <- c("blue","green", "purple")
# Add a legend
legend("topright", legend = unique(train$Position), col = unique(color_vec), pch = 16)

###### lda on testing set ######
#predict on test set
Test.pred = predict(fit.lda, test)
Test.pred$x  # score
ldahist (data = Test.pred$x[,1], g=test$Position)
ldahist (data = Test.pred$x[,2], g=test$Position)

plot(Test.pred$x[,1], Test.pred$x[,2], col=c("blue","green", "purple")[test$Position], pch=16,
    main = "Separation on Testing set")
# Add a legend
legend("topright", legend = unique(train$Position), col = unique(color_vec), pch = 16)

#confusion matrix
library(caret)
confusionMatrix(data = Train.pred$class, reference = as.factor(train$Position))
confusionMatrix(data = Test.pred$class, reference = as.factor(test$Position))

#------------------------------------------------------------------------------
```

```
### Logistic Regreesion ###

#create new ds for forward and guard to perform logistic regression
forward <- subset(new_df, Pos %in% c("PF", "SF"))
guard <- subset(new_df, Pos %in% c("SG", "PG"))

#dummy
forward$position <- ifelse(forward$Pos == "PF", 1,0)
guard$position <- ifelse(guard$Pos == "PG", 1,0)

#Splitting data
set.seed(1233)
sampleF <- sample(c(TRUE, FALSE), nrow(forward), replace=TRUE, prob=c(0.7,0.3))
sampleG <- sample(c(TRUE, FALSE), nrow(guard), replace=TRUE, prob=c(0.7,0.3))

train.f.1  <- forward[sampleF, ]
test.f.1   <- forward[!sampleF, ]
train.g.1  <- guard[sampleG, ]
test.g.1   <- guard[!sampleG, ]

train.f = train.f.1[,-1]
test.f = test.f.1[,-1]
train.g = train.g.1[,-1]
test.g = test.g.1[,-1]

#logistic regression
# Fit the logistic regression model
f.model <- glm(position ~ ., data = train.f, family = "binomial")
g.model <- glm(position ~ ., data = train.g, family = "binomial")
# Print the model summary
summary(f.model)
summary(g.model)

# Fit the final model then predict on testing set
f.model1 = glm(position ~ DRB+AST, data = train.f, family = "binomial")
summary(f.model1)
g.model1 = glm(position ~ DRB+AST+PTS, data = train.g, family = "binomial")
summary(g.model1)

#prediction for forward set
fitted.results.f <- predict(f.model1,test.f, type='response')
fitted.results.f <- ifelse(fitted.results.f > 0.5,1,0)
misClasificError <- mean(fitted.results.f != test.f$position)
print(paste('Accuracy',1-misClasificError))

#prediction for guard set
fitted.results.g <- predict(g.model1, test.g, type='response')
fitted.results.g <- ifelse(fitted.results.g > 0.5,1,0)

misClasificError <- mean(fitted.results.g != test.g$position)
print(paste('Accuracy',1-misClasificError))
```

Factor Analysis Code Script

```
library(tidyverse)
library(corrplot)
library(psych)

nba <- read.csv("NBA_Player_stats.csv", sep=";")
head(nba)

summary(nba)

colSums(is.na(nba))

#we'll divide the dataset into team and player stats
nbaF <- nba[nba1$Tm != "TOT", ]

nba_plyrstat <- nbaF[c(3:4, 6:8, 12:13, 15:16, 19:20, 22:23, 25:30)]

head(nba_plyrstat)

nba_tmstat <- nbaF[ -c(1, 2, 5, 9, 18, 24)]

head(nba_tmstat)

summary(nba_plyrstat)

summary(nba_tmstat)

duplicate_rows <- nba_plyrstat[duplicated(nba_plyrstat) | duplicated(nba_plyrstat, fromLast = TRUE), ] #checking for duplicate
rows
print(duplicate_rows)

summary(nba_plyrstat)

duplicate_rows2 <- nba_tmstat[duplicated(nba_tmstat) | duplicated(nba_tmstat, fromLast = TRUE), ]
print(duplicate_rows2)

#on the nba player stat
head(nba_plyrstat)

#Create dummy
library(fastDummies)
nba_plyrstat2 <- dummy_cols(nba_plyrstat, select_columns = "Pos")

head(nba_plyrstat2)

pairs.panels(nba_plyrstat[,c(-1)]) #scatterplot

##Transforming the variables
library(dplyr)
nba_plyrstat3 <- nba_plyrstat2 %>%
  mutate_at(vars(GS), ~log(1 + .)) %>% mutate_at(vars(X2P ), ~log(1 + .)) %>%
  mutate_at(vars(X3P), ~log(1 + .)) %>% mutate_at(vars(X3PA), ~log(1 + .)) %>%
  mutate_at(vars(X2PA), ~log(1 + .)) %>% mutate_at(vars(FT), ~log(1 + .)) %>%
  mutate_at(vars(FTA), ~log(1 + .)) %>% mutate_at(vars(ORB), ~log(1 + .)) %>%
  mutate_at(vars(DRB), ~log(1 + .)) %>% mutate_at(vars(AST), ~log(1 + .)) %>%
  mutate_at(vars(STL), ~log(1 + .)) %>% mutate_at(vars(BLK), ~log(1 + .)) %>%
  mutate_at(vars(TOV), ~log(1 + .)) %>% mutate_at(vars(PTS), ~log(1 + .))
```

```
head(nba_plyrstat3)
#guys do the same for the nba_tmstat dataset
library(ggplot2)

g1 <- ggplot(nba_plyrstat3, aes(x= GS)) + geom_histogram()
g2 <- ggplot(nba_plyrstat3, aes(x= G)) + geom_histogram()
g3 <- ggplot(nba_plyrstat3, aes(x= X3P)) + geom_histogram()
g4 <- ggplot(nba_plyrstat3, aes(x= X2P)) + geom_histogram()
g5 <- ggplot(nba_plyrstat3, aes(x= FT)) + geom_histogram()
g6 <- ggplot(nba_plyrstat3, aes(x= FTA)) + geom_histogram()
g7 <- ggplot(nba_plyrstat3, aes(x= ORB)) + geom_histogram()
g8 <- ggplot(nba_plyrstat3, aes(x= DRB)) + geom_histogram()
g9 <- ggplot(nba_plyrstat3, aes(x= AST)) + geom_histogram()
g10 <- ggplot(nba_plyrstat3, aes(x= STL)) + geom_histogram()
g11 <- ggplot(nba_plyrstat3, aes(x= BLK)) + geom_histogram()
g12 <- ggplot(nba_plyrstat3, aes(x= TOV)) + geom_histogram()
g13 <- ggplot(nba_plyrstat3, aes(x= PTS)) + geom_histogram()

library(gridExtra)

grid.arrange(g1, g2, g3, g4, g5, g6, g7, g8, g9, g10, g11, g12, g13, nrow = 4)

#we check correlations
head(nba_plyrstat3)

nba_plyrstat4 <- nba_plyrstat3[, -c(1)]
head(nba_plyrstat4)

cor1 = cor(nba_plyrstat4) #correlation
corrplot(cor1,order="AOE")



head(nba_plyrstat4)

KMO(nba_plyrstat4)

#all have 0.5. We investigate further.
#From our cor plot, we can see that X3PA, X3P, X2PA, X2P, BLK, FT and FTa, ORB have very similar correlation.
#we take them out.

#so we take one out leaving the other. we leave the not so obvious.
nba_plyrstatrd <- nba_plyrstat4[, -c(5, 7, 9, 11)] #so we take one out leaving the other. we leave the not so obvious
head(nba_plyrstatrd)

KMO(nba_plyrstatrd)
#Each of them still gives us 0.5
MCorrTest = corr.test(nba_plyrstatrd, adjust="none")

M = MCorrTest$p
round(M, 2)
MTest = ifelse(M < .01, T, F)
MTest

colSums(MTest) - 1
```

```
#we run an initial OLS to see if there is still heavy multi-collinearity
fit2 = lm(PTS ~., nba_plyrstatrd)
summary(fit2)

library(car)
vif(fit2)

#DRB will be its own factor, Also let Age be it's own factor for now*
#defensive rebounds may have to be its own factor
head(nba_plyrstatrd)

library(psych)
nba_plyrd2 <- nba_plyrstatrd[,-c(1)]
head(nba_plyrd2)
nba_plyrd <- nba_plyrstatrd[, -c(1, 8, 14)] # response var and highly correlated DRB and AGE vars since it is its own factor

head(nba_plyrd)
str(nba_plyrd)

pc = prcomp(nba_plyrd, scale=T)
summary(pc)
plot(pc)
abline(1, 0, col="red")

fa.parallel(nba_plyrd)
# components

#nba_plyrd$DRB <- nba_plyrstatrd$DRB
head(nba_plyrd)
#nba_plyrd <- nba_plyrd[,-c(18)]
pf1 = principal(nba_plyrd, rotate="varimax",  nfactors=5 )
summary(pf1)
print(pf1$loadings,cutoff=.4,sort=T)

scores = as.data.frame(pf1$scores)
head(scores)

names(scores) = c("actions_in_game",  "3pattemps_by_SG_and_BLK_by_C", "synergy_btw_PG_and_SG",
"synergy_btw_SG_and_PF_intbyPF", "synergy_btw_SG_and_SF_intbySF")
scores$PTS = nba_plyrstatrd$PTS
scores$DRB <- nba_plyrstatrd$DRB
scores$Age <- nba_plyrstatrd$Age
head(scores)

#running an OLS regression

fitR1 = lm(PTS ~., data=scores)
summary(fitR1)

#vif

vif(fitR1)

#Doing for factor Analysis
head(nba_plyrstatrd)
```

```
nba_factor <- nba_plyrstatrd[, -c(1, 8, 14, 15, 16, 17, 18, 19)]#removing Age, PTS, DRB and dummys.

head(nba_factor)
pfc = prcomp(nba_factor, scale=T)
summary(pfc)
plot(pfc)
abline(1, 0, col="red")

fa.parallel(nba_factor)

fitnba_ply1 = factanal(nba_factor, factors=2)
print(fitnba_ply1)
print(fitnba_ply1$loadings, cutoff=.4, sort=T)

fact_loadings <- fitnba_ply1$loadings

scores3 <- as.data.frame(as.matrix(nba_factor) %*% fact_loadings)

head(scores3)

names(scores3) = c("Actions in games played including blocks - 3PA(c)",  "Actions in games with 3pa - blocks(SG)")
scores3$PTS = nba_plyrstatrd$PTS
scores3$DRB <- nba_plyrstatrd$DRB
scores3$Age <- nba_plyrstatrd$Age
#adding back the position dummies vars
scores3$Pos_C <- nba_plyrstatrd$Pos_C
scores3$Pos_PF <- nba_plyrstatrd$Pos_PF
scores3$Pos_PG <- nba_plyrstatrd$Pos_PG
scores3$Pos_SF <- nba_plyrstatrd$Pos_SF
scores3$Pos_SG <- nba_plyrstatrd$Pos_SG

head(scores3)

#run an OLS reg

fft1 = lm(PTS ~., data=scores3)
summary(fft1)
#=============================================================================================================
#We run PCA and PFA and FA on the data without transforming it to see.

head(nba_plyrstat2)

nba_pstatpt2 <- nba_plyrstat2[, -c(1)]
head(nba_pstatpt2)

cor3 = cor(nba_pstatpt2)
corrplot(cor3, order ="AOE")

nba_pstatpt3 <- nba_pstatpt2[, -c(5, 7, 9, 11)]#removing equally correlated vars from the corplot and the response var

head(nba_pstatpt3)

MCorrTest3 = corr.test(nba_pstatpt2, adjust="none")

M3 = MCorrTest3$p
round(M3, 2)
MTest1 = ifelse(M3 < .01, T, F)
MTest1
```

```
colSums(MTest1) - 1

#DRB has to be it's own factor and age as well
head(nba_pstatpt2)

nba_pstatpfa <- nba_pstatpt2[, -c(1, 18, 12)]
head(nba_pstatpfa)

#doing PFA on the full untransformed data without removing highly corr to see

put = prcomp(nba_pstatpfa, scale=T) #p untransformed
summary(put)
plot(put)
abline(1, 0, col="red")

fa.parallel(nba_pstatpfa)


head(nba_pstatpfa)

ptpf2 = principal(nba_pstatpfa, rotate="varimax",  nfactors=6 )
summary(ptpf2)
print(ptpf2$loadings, cutoff=.4, sort=T)

scores4 = as.data.frame(ptpf2$scores)
head(scores4)
names(scores4) = c("2Pt_2Pa_FreeT&FreeTA_ASt&ToV_in_GS&MP", "AST_TOV_3P_3PA_STL&PF in Games",
"Personal_fouls_ORB_BLK_on 3p & 3PA_by_C", "Synergy_btw_PG&SG_in_Ast", "PF", "synergy_btw_SG&SF")

head(scores4)
scores4$PTS = nba_pstatpt2$PTS
scores4$DRB <- nba_pstatpt2$DRB
scores4$Age <- nba_pstatpt2$Age
head(scores4)

#run a regression
ftu1 = lm(PTS ~., data=scores4)
summary(ftu1)

vif(ftu1)

#removing those with high vif scores

ftu =lm(PTS ~ `AST_TOV_3P_3PA_STL&PF in Games`+`Personal_fouls_ORB_BLK_on 3p & 3PA_by_C`+
`Synergy_btw_PG&SG_in_Ast`+ PF + `synergy_btw_SG&SF` + DRB  +  Age, data= scores4)
summary(ftu)

vif(ftu)
summary(ftu)


head(nba_pstatpfa)


nba_pstatpfa2 <- nba_pstatpfa[, -c(16, 17, 18, 19, 20)] #takiing out dummies for CFA Untransformed

v = prcomp(nba_pstatpfa2, scale=T) #p untransformed
summary(v)
```

```
plot(v)
abline(1, 0, col="red")

fa.parallel(nba_pstatpfa2)

ftpfa = factanal(nba_pstatpfa2, factors=2)#Factanal did not work with the untransformed dataset
print(ftpfa)
print(ftpfa$loadings, cutoff=.4)
ftpfa_loadings <- ftpfa$loadings

scores7 <- as.data.frame(as.matrix(nba_pstatpfa2) %*% ftpfa_loadings)

head(scores7)

names(scores7) = c("Actions_in_games_played_including blocks_minus_3PA,3p&STLs", "Actions_in_Games_Including_3PA&3PA
-2P,ORB,BLK&PF")
scores7$PTS = nba_pstatpt2$PTS
scores7$DRB <- nba_pstatpt2$DRB
scores3$Age <- nba_pstatpt2$Age
#adding back the position dummies vars
scores7$Pos_C <- nba_pstatpt2$Pos_C
scores7$Pos_PF <- nba_pstatpt2$Pos_PF
scores7$Pos_PG <- nba_pstatpt2$Pos_PG
scores7$Pos_SF <- nba_pstatpt2$Pos_SF
scores7$Pos_SG <- nba_pstatpt2$Pos_SG
head(scores7)

#running OLS in the CFA 2(untransformed) scores
utfit <- lm(PTS ~., data=scores7)

summary(utfit)
#=============================================================================================================
#untransformed taking highly correlated out
head(nba_pstatpt3)
KMO(nba_pstatpt3)
MCorrTest4 = corr.test(nba_pstatpt3, adjust="none")

M4 = MCorrTest4$p
round(M4, 2)
MTest2 = ifelse(M4 < .01, T, F)
MTest2
colSums(MTest2) - 1

#again DRB and Age will be their own factor
head(nba_pstatpt3)
nba_pstatpt4 <-nba_pstatpt3[, -c(1, 8, 14)]

head(nba_pstatpt4)


b = prcomp(nba_pstatpt4, scale=T) #p untransformed 2
summary(b)
plot(b)
abline(1, 0, col="red")

fa.parallel(nba_pstatpt4)
#5 components
```

```
head(nba_pstatpt4)



pnt = principal(nba_pstatpt4, rotate="varimax",  nfactors=5)
summary(pnt)
print(pnt$loadings, cutoff=.4, sort=T)

scores5 = as.data.frame(pnt$scores)
head(scores5)
names(scores5) = c("actions_in_game_played",  "3pattemps_by_SG&BLK_by_C", "synergy_btw_PG$SG_in_Ast",
"synergy_btw_PF&SG_inTbyPF", "synergy_btw_SF&SG")
scores5$PTS = nba_pstatpt3$PTS # Adding back PTS, DRB and Age
scores5$DRB <- nba_pstatpt3$DRB
scores5$Age <- nba_pstatpt3$Age
head(scores5)

#fit the ols model

ftut <- lm(PTS ~., data= scores5)
summary(ftut)

vif(ftut)

#CFA with untransformed data and with Aliasing removed
head(nba_pstatpt4)

nba_statpt7 <- nba_pstatpt4[, -c(12,13,14,15,16)]

head(nba_statpt7)
#Factanal with the untransformed data
c1 = prcomp(nba_statpt7, scale=T) #p untransformed 3 with dummys removed
summary(c1)
plot(c1)
abline(1, 0, col="red")

fa.parallel(nba_statpt7)

head(nba_statpt7)

#Running CFA3

faut = factanal(nba_statpt7, factors =2)
print(faut)
print(faut$loadings, cutoff=.4, sort=T)

fact_loadingsut <- faut$loadings

scores6 <- as.data.frame(as.matrix(nba_statpt7) %*% fact_loadingsut)

head(scores6)

names(scores6) = c("Actions_in_ Game - 2PA_FTA&BLK", "Actions_in_games_started - 3PA_STL_PF&BLKS")

head(nba_pstatpt3)
#Adding back PTS, DRB, Age and Position dummy variables
scores6$PTS = nba_pstatpt3$PTS
scores6$DRB <- nba_pstatpt3$DRB
```

```
scores6$Age <- nba_pstatpt3$Age
#adding back the position dummies vars
scores6$Pos_C <- nba_pstatpt3$Pos_C
scores6$Pos_PF <- nba_pstatpt3$Pos_PF
scores6$Pos_PG <- nba_pstatpt3$Pos_PG
scores6$Pos_SF <- nba_pstatpt3$Pos_SF
scores6$Pos_SG <- nba_pstatpt3$Pos_SG

head(scores6)

#fit reg model

fitunt = lm(PTS ~., data = scores6)
summary(fitunt)

vif(fitunt)
#removing high var with high vif score

#ftut4 = lm(PTS ~.-`3pA_in _MP`-`2pa_FreeTA_Ast&Tov_in_MP&Gs`- `2Pa_FreeTA_BLKs_PF_STL_ingames`, data=scores6)
#summary(ftut4)

#vif(ftut4)

#head(scores)
#=============================================================================================================#
# All subsets for validating the models

#PFA 1 Scores
#CFA1 scores3
install.packages("leaps")
library(leaps)
PFA1Subsets = regsubsets(PTS ~ ., data=scores)

plot(PFA1Subsets, scale="bic")

#validating the model for CFA1 using allsubsets
CFA1Subsets = regsubsets(PTS ~ ., data=scores3)
plot(CFA1Subsets, scale="adjr2")

#fiiting it in an ols gives
#"Actions in games played including blocks - 3PA(c)",  "Actions in games with 3pa - blocks(SG)" DRB, Age, Pos_c, Pos_SG
CFA1bestfit <- lm(PTS ~. -Pos_PF - Pos_PG - Pos_SF, data=scores3)
summary(CFA1bestfit)

#we check how much the significance of Age

agecheck <- lm(PTS~ Age, data= scores3)
summary(agecheck)

PosSGcheck <- lm(PTS~Pos_SG,data=scores3)
summary(PosSGcheck)


#PFA3 scores5

PFA3Subsets = regsubsets(PTS ~ ., data=scores5)
```

```
plot(PFA3Subsets, scale="adjr2")

PFA3bestfit <- lm(PTS~.-`synergy_btw_SF&SG` -Age, data= scores5)
summary(PFA3bestfit)

#check for PF&SG

PFcheck <- lm(PTS ~ `synergy_btw_PF&SG_inTbyPF`, data=scores5)
summary(PFcheck)
#negligible R2 and Adj r2. doiing nothing to the predictive power of the model

Agecheck2 <- lm(PTS ~ Age, data=scores5)
summary(Agecheck2)
#negligible too

SGPGcheck <- lm(PTS~ `synergy_btw_PG$SG_in_Ast`, data=scores5)
summary(SGPGcheck)
#checking the model for PFA1 for synergy btw Pf and SG
summary(fitR1)

PFA1agechk <- lm(PTS~ Age, data= scores)
summary(PFA1agechk)

PFandSGPF1 <- lm(PTS~`synergy_btw_SG&PF_intbyPF`, data=scores)
summary(PFandSGPF1)
#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++
#Confirmatory Factor Analysis

#running confirmatory factor Analys on the chosen model PFA1

library(lavaan)
#checking the PF1

print(pf1$loadings,cutoff=.4,sort=T)
head(scores)

#names(scores) = c("actions_in_game",  "3pattemps_by_SGandBLK_by_C", "synergy_btw_PG$SG",
"synergy_btw_SG&PF_intbyPF", "synergy_btw_SG&SF_intbySF")

PFA.model6 = 'actions_in_game =~ G + GS + MP + X2PA + FTA + AST + STL + TOV + PF + X3PA + BLK
        Threepoint_attempts_by_SG_and_BLocK_by_C =~ BLK + (Pos_C) + (-Pos_SG)+ (-X3PA)
        synergy_btw_PG_and_SG =~  Pos_PG + Pos_SG
        synergy_btw_SG_and_PF_intbyPF =~ Pos_PF + (-Pos_SG)
        synergy_btw_SG_and_SF_intbySF =~ Pos_SF +(-Pos_SG)'

nba_plyrd5 <- nba_plyrd #remove dummies
nba_plyrd5$Pos <- nba_plyrstat3$Pos
head(nba_plyrd5)
nba_plyrd6 <- nba_plyrd5[,-(12:16)]

head(nba_plyrd6)
#nba_plyrd_standardized <- scale(nba_plyrd)
fitcof2<- cfa(PFA.model6, data = nba_plyrd6)
vt <- varTable(fitcof1)
summary(fitcof, fitcf.measures=TRUE)

print(fitnba_ply1$loadings, cutoff=.4, sort=T)
```

```
CFA.model = 'Actions in games played including blocks_minus_3PA_c =~ GS + X2PA + FTA + BLK + TOV + PF + MP + AST + STL + G
        Actions in games with 3pa_minus_blocks_SG =~ GS + X2PA + FTA + TOV + PF + MP + X3PA + AST + STL + G'

fit <- cfa(CFA.model, data=nba_factor)
summary(fit, fit.measures=TRUE)
```