Performance vs. salary - a relationship analysis for NBA players

Vladimira Gabor, Hartmann Urs, Eitam Shafran

07/01/2022

Applied Machine Learning and Predictive Modeling 1

Lecturers: Dr. Matteo Tanadini, Daniel Meister

Contents

1 Introduction

In this work, a relationship analysis of NBA players' performance during the 2019-20 season was conducted. While the relationship between player's performance to their salaries during the following season was explored in chapter 3 and 4, a predictive model for the number of successful field goals per season is presented in chapter 5. Chapter 6 of this paper will describe a fictitious marketing optimization problem.

1.1 Dataset

Statistic data of NBA players' performance during the 2019-20 season was downloaded from "basketball-reference.com" . The data consists of general information about the players, such as their age, team, position and number of games played. In addition, information about their performance during the season, such as the overall shooting tries and success, number of steals, blocks and more is available. Supplementary data on the player salaries during the season of 2020-21, was scraped from "hoopshype.com" and merged with the statistic data, using python. The merged data contain 33 attributes for 651 NBA players.

1.2 Dataset glossary

"Rk" - ranking
"Player"- name
"Pos"- position
"Age"- age
"Tm"- team
"G"- number of games
"GS"- games started
"MP"- minute played
"FG"- Field goal
"FGA"- Field goal attempts
"FG."- Field goal %
"X3P"- 3 point field goals
"X3PA"- 3 point field goals attempts
"X3P."- 3 point field goals %
"X2P"- 2 point field goals
"X2PA"- 2 point field goals attempts
"X2P." - 2 point field goals %
"eFG."- Effective field goals %
"FT"- Free throws
"FTA"- Free throws attempts
"FT."- Free throws %
"ORB"- Offensive rebounds
"DRB"- Defensive rebounds
"TRB"- Total rebounds
"AST"- Assists
"STL"- Steals
"BLK"- Blocks
"TOV"- Turnovers
"PF"-Personal fouls
"PTS"- Points
"X2019.20"- 2019/20 salary in $
"X2020.21"- 2020/21 salary in $

2 Data preprocessing and exploratory data analysis

Getting an overview of data frame

```
df %>% dplyr::select(1:14) %>% head() %>% kable()
```

Rk	Player	Pos	Age	Tm	G	GS	MP	FG	FGA	FG.	X3P	X3PA	X3P.
1	Steven Adams	C	26	OKC	63	63	1680	283	478	0.592	1	3	0.333
2	Bam Adebayo	PF	22	MIA	72	72	2417	440	790	0.557	2	14	0.143
3	LaMarcus Aldridge	C	34	SAS	53	53	1754	391	793	0.493	61	157	0.389
4	Kyle Alexander	C	23	MIA	2	0	13	1	2	0.500	0	0	NA
5	Nickeil Alexander-Walker	SG	21	NOP	47	1	591	98	266	0.368	46	133	0.346
6	Grayson Allen	SG	24	MEM	38	0	718	117	251	0.466	57	141	0.404

```
df %>% dplyr::select(1,2,15:24) %>% head() %>% kable()
```

Rk	Player	X2P	X2PA	X2P.	eFG.	FT	FTA	FT.	ORB	DRB	TRB
1	Steven Adams	282	475	0.594	0.593	117	201	0.582	207	376	583
2	Bam Adebayo	438	776	0.564	0.558	264	382	0.691	176	559	735
3	LaMarcus Aldridge	330	636	0.519	0.532	158	191	0.827	103	289	392
4	Kyle Alexander	1	2	0.500	0.500	0	0	NA	2	1	3
5	Nickeil Alexander-Walker	52	133	0.391	0.455	25	37	0.676	9	75	84
6	Grayson Allen	60	110	0.545	0.580	39	45	0.867	8	77	85

```
df %>% dplyr::select(1,2,25:33) %>% head() %>% kable()
```

Rk	Player	AST	STL	BLK	TOV	PF	PTS	X2019.20	X2020.21	Rank
1	Steven Adams	146	51	67	94	122	684	25842697	26009571	41
2	Bam Adebayo	368	82	93	204	182	1146	3454080	3476384	235
3	LaMarcus Aldridge	129	36	87	74	128	1001	26000000	26167890	40
4	Kyle Alexander	0	0	0	1	1	2	79568	80081	463
5	Nickeil Alexander-Walker	89	17	8	54	57	267	2964840	2983984	247
6	Grayson Allen	52	10	2	33	53	330	2429400	2445087	281

2.1 Removing bias and redundant data

During the pre-processing stage, the following features were removed:

- all columns representing proportionate % values, those were considered as redundant data entries
- players' names
- column "Rk" Ranking

The data frame is left with 24 columns.

```
df <- df[,c('Age', 'AST', 'BLK', 'DRB', 'FG', 'FGA', 'FT', 'FTA', 'MP', 'ORB',
            'PF', 'PTS', 'STL', 'TOV', 'TRB', 'X2P', 'X2PA', 'X3P', 'X3PA',
            'Tm', 'Pos', 'G', 'GS','X2020.21')]
```

2.2 Consolidating players' position column

Positions in Basketball vary. Mix positions with insufficient amount of data (based on previews analysis) are aggregated to Center C, Small Forward SF, Shooting Guard SG and Point Guard PG.

```
df <- df %>%
   dplyr::mutate(Pos= car::recode(Pos,"c('C', 'C-PF') = 'C'; c('SF', 'SF-C',
   'SF-PF', 'SF-SG')= 'SF';
   c('SG', 'SG-PG') = 'SG'; c('PG', 'PG-SG') = 'PG'" ))
```

2.3 Missing values

To detect and expose missing values, the following function was created.

```
## Function, returning a table with the name of columns as rows,
##  number of missing values and their percentage as columns
missing.values <- function(df) {
  missing.values <- df %>%
    gather(key = "key", value = "val") %>%
    mutate(is.missing = is.na(val)) %>%
    group_by(key, is.missing) %>%
    summarise(num.missing = n()) %>%
    filter(is.missing==T) %>%
    dplyr::select(-is.missing) %>%
    arrange(desc(num.missing)) %>%
    mutate(percentage = round(num.missing/nrow(df)*100, 3))
  return(missing.values)
}
```

key	num.missing	percentage
X2020.21	99	15.207

Key insight: There are 99 rows with missing salary entries. We will split them from the original data:

```
Missing_salary <- df[is.na(df$X2020.21),] ## data we can later make a prediction on
df <- df[!is.na(df$X2020.21),] ## df without missing data
```

2.4 Splitting the data for training and testing

```
training_size <- 0.8
training_rows <- sample(seq_len(nrow(df)),
                        size = floor(training_size * nrow(df)))
train <- df[training_rows, ]
test <- df[-training_rows, ]
```

3 Players' performance vs. Salary

In this chapter, we will focus on features representing the player's performance during the season, in addition to his team, position and age data. To make sure our model represents the actual players' performance, we will normalize the performance data by the time each player played during the season.

Dependent variable DV = 'X2020.21'

Feature columns

"Age"- age
"Tm"- team
"FG"- Field goal
"FGA"- Field goal attempts
"X3P"- 3 point field goals
"X3PA"- 3 point field goals attempts
"X2P"- 2 point field goals
"X2PA"- 2 point field goals attempts
"FT"- Free throws
"FTA"- Free throws attempts
"ORB"- Offensive rebounds
"DRB"- Defensive rebounds
"TRB"- Total rebounds
"AST"- Assists
"STL"- Steals
"BLK"- Blocks
"TOV"- Turnovers
"PF"- Personal fouls
"PTS"- Points
"Pos"- position

Normalizing players' performance by time played

Normalized players' performance by time played and reunite with countable data (for train and test data).

```
train_norm <- train[,c( 'AST', 'BLK', 'DRB', 'FG', 'FGA', 'FT', 'FTA', 'ORB',
                        'PF', 'PTS', 'STL', 'TOV', 'TRB', 'X2P', 'X2PA', 'X3P',
                        'X3PA')]/train[,'MP']
train_norm <- cbind(train_norm, train[,c('Age','Tm', 'Pos','X2020.21' )])
```

3.1 Exploratory data analysis

Columns statistical characteristics

```
##       AST              BLK                DRB               FG
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.05000   1st Qu.:0.008008   1st Qu.:0.1005   1st Qu.:0.1134
##  Median :0.07236   Median :0.015617   Median :0.1337   Median :0.1434
##  Mean   :0.08875   Mean   :0.021995   Mean   :0.1426   Mean   :0.1494
##  3rd Qu.:0.11458   3rd Qu.:0.028986   3rd Qu.:0.1778   3rd Qu.:0.1786
##  Max.   :0.50000   Max.   :0.133621   Max.   :0.4242   Max.   :0.3573
##
##       FGA              FT               FTA               ORB
##  Min.   :0.08219   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.26178   1st Qu.:0.03416   1st Qu.:0.04839   1st Qu.:0.01939
##  Median :0.32916   Median :0.05386   Median :0.07238   Median :0.03175
##  Mean   :0.33739   Mean   :0.06172   Mean   :0.08291   Mean   :0.04753
##  3rd Qu.:0.39277   3rd Qu.:0.08333   3rd Qu.:0.11005   3rd Qu.:0.06447
##  Max.   :1.00000   Max.   :0.40000   Max.   :0.40000   Max.   :0.75000
##
##        PF              PTS              STL               TOV
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.07302   1st Qu.:0.3125   1st Qu.:0.02174   1st Qu.:0.03784
##  Median :0.09123   Median :0.3885   Median :0.02929   Median :0.05082
##  Mean   :0.09758   Mean   :0.4040   Mean   :0.03202   Mean   :0.05391
##  3rd Qu.:0.11475   3rd Qu.:0.4754   3rd Qu.:0.04060   3rd Qu.:0.06681
##  Max.   :0.24566   Max.   :0.9687   Max.   :0.12121   Max.   :0.14286
##
##       TRB              X2P               X2PA              X3P
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.1239   1st Qu.:0.06725   1st Qu.:0.1463   1st Qu.:0.02000
```

```
##  Median :0.1707   Median :0.09831   Median :0.1991   Median :0.04355
##  Mean   :0.1901   Mean   :0.10581   Mean   :0.2044   Mean   :0.04356
##  3rd Qu.:0.2346   3rd Qu.:0.13547   3rd Qu.:0.2532   3rd Qu.:0.06296
##  Max.   :1.0000   Max.   :0.33333   Max.   :1.0000   Max.   :0.19048
##
##       X3PA                Age              Tm            Pos
##  Min.   :0.00000   Min.   :19.00   TOT    : 36   SG     :108
##  1st Qu.:0.08511   1st Qu.:22.00   MIN    : 19   PF     : 94
##  Median :0.13174   Median :25.00   DET    : 18   C      : 90
##  Mean   :0.13298   Mean   :25.56   PHO    : 17   SF     : 74
##  3rd Qu.:0.18182   3rd Qu.:28.00   SAC    : 17   PG     : 69
##  Max.   :0.40476   Max.   :39.00   GSW    : 16   PF-C   :  4
##                                    (Other):318   (Other):  2
##     X2020.21
##  Min.   :   80081
##  1st Qu.:  904110
##  Median : 2494846
##  Mean   : 6498611
##  3rd Qu.: 8786614
##  Max.   :40491547
##
```

3.1.1 Exploring Numerical features

Extracting names of numerical columns

```
names_numerical  <- train_norm %>% dplyr::select(where(is.integer)|where(is.numeric)) %>%
  colnames()
```

Density plots

Numerical variables are plotted as density plots to explore data distribution.

```
train_norm[names_numerical] %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~key, scales ='free') +
  geom_density(color = 'black', fill = 'lightblue')+
  ggtitle("Numerical variables")
```

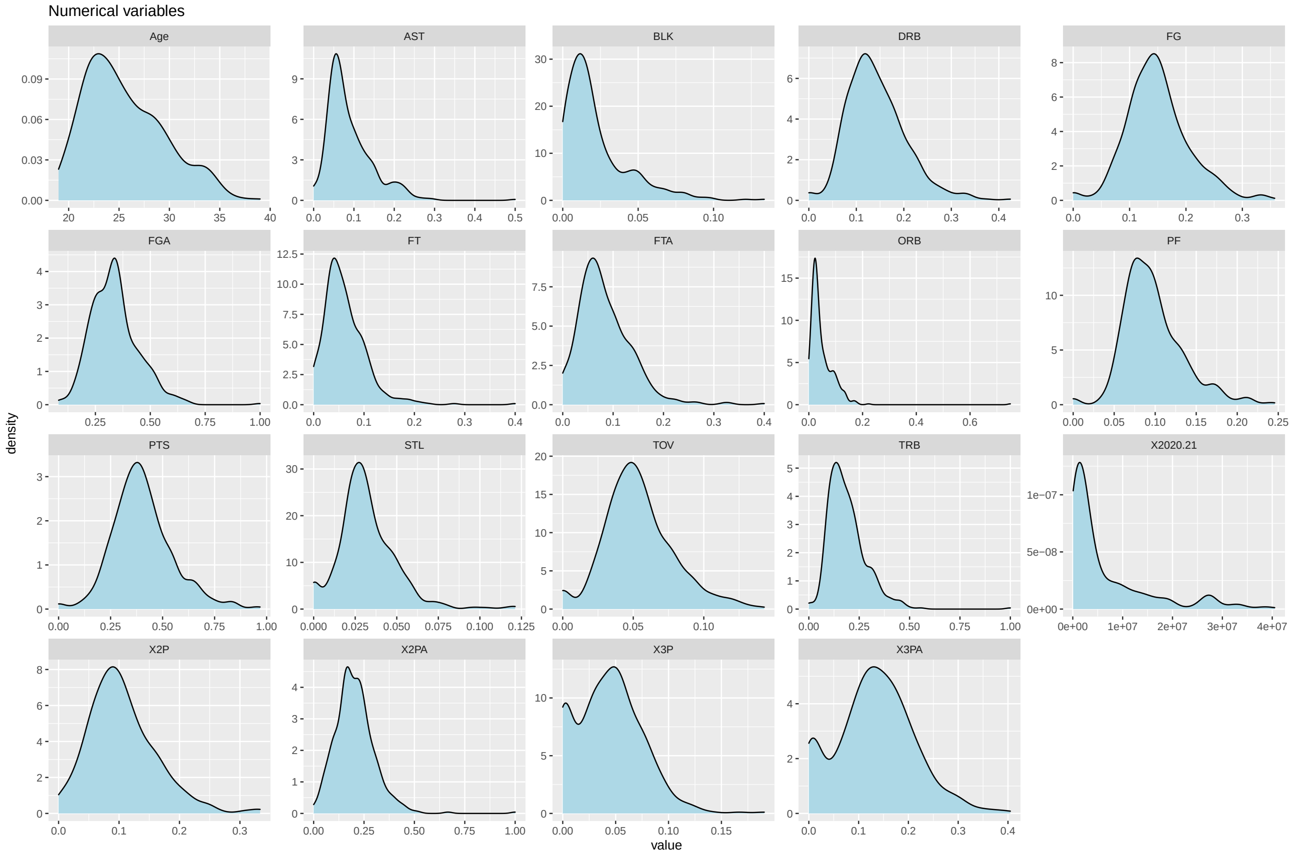


Fig. 1: Density plots of numerical values

Data skewness

```r
sk <-sapply(train_norm[names_numerical], function(x) skewness(x, na.rm = TRUE)) %>% sort()
sk[1:10] %>% t() %>% kable(row.names = FALSE)
```

X3PA	FG	TOV	PTS	X3P	Age	DRB	PF	X2P	FGA
0.2669629	0.5086178	0.5441557	0.6005894	0.6096736	0.642035	0.8231024	0.8924864	0.9254029	0.9427027

```r
sk[11:19] %>% t() %>% kable(row.names = FALSE)
```

STL	FTA	X2PA	BLK	AST	X2020.21	TRB	FT	ORB
1.292887	1.427631	1.600169	1.722317	1.736165	1.821725	2.0106	2.068556	6.616318

Key insights: 1) Salary column is an amount, based on figure 1 we can also see that it's right-skewed. Therefore, we should log-transform it before building the model.
2) All features are right skewed except for the X3PA feature.

Looking for outliers

To analyze numerical values further and spot potential outliers, we plot our data in box plots.

```r
train_norm[names_numerical] %>% gather() %>% ggplot( aes(x = key, y=value)) +
  facet_wrap(facets = 'key', scale = 'free') + geom_boxplot(na.rm = TRUE)
```

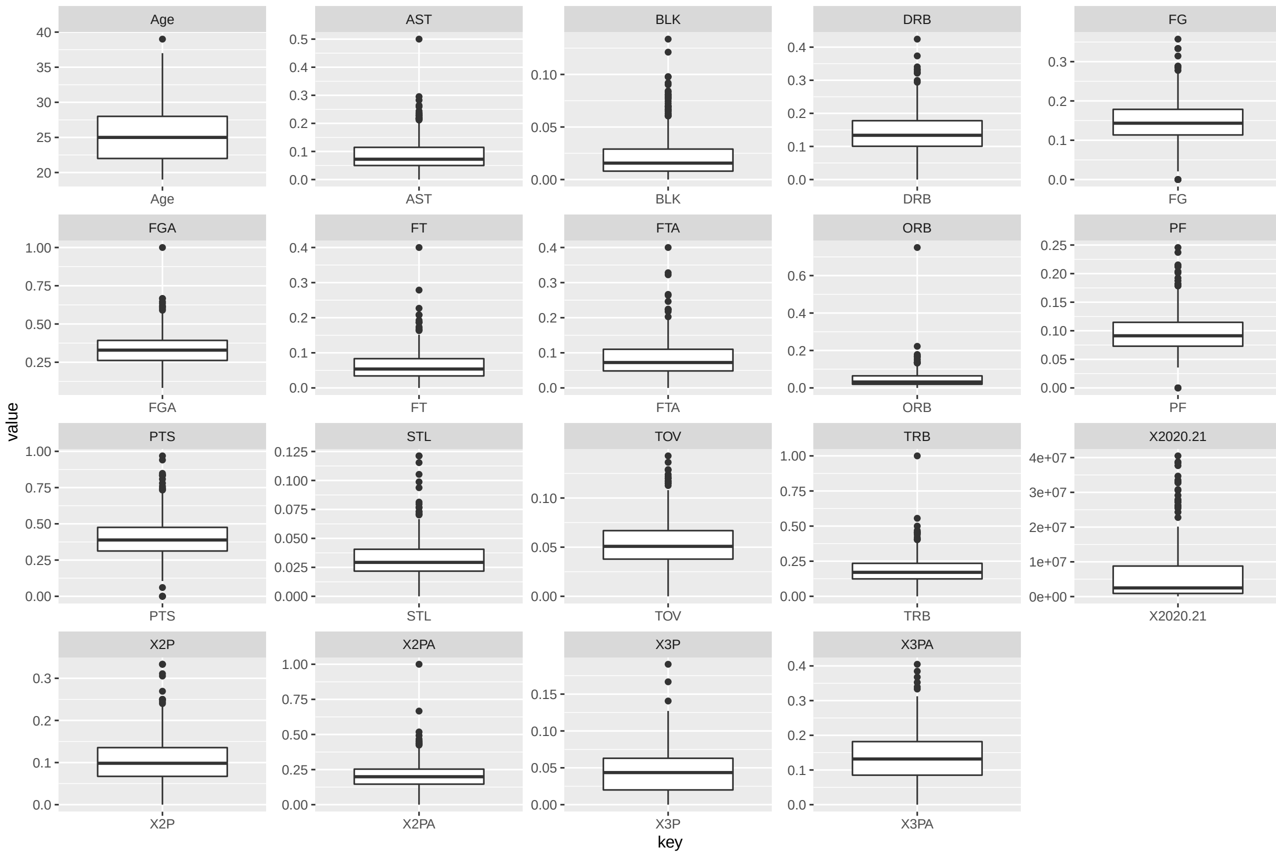


Fig. 2: Box plots of numerical data (before log-transformation)

Heatmap correlation matrix

To make possible correlations visible, numerical values are plotted in a heatmap. Dark blue color represents strong positive correlation, while dark red describes strong negative correlation among features.

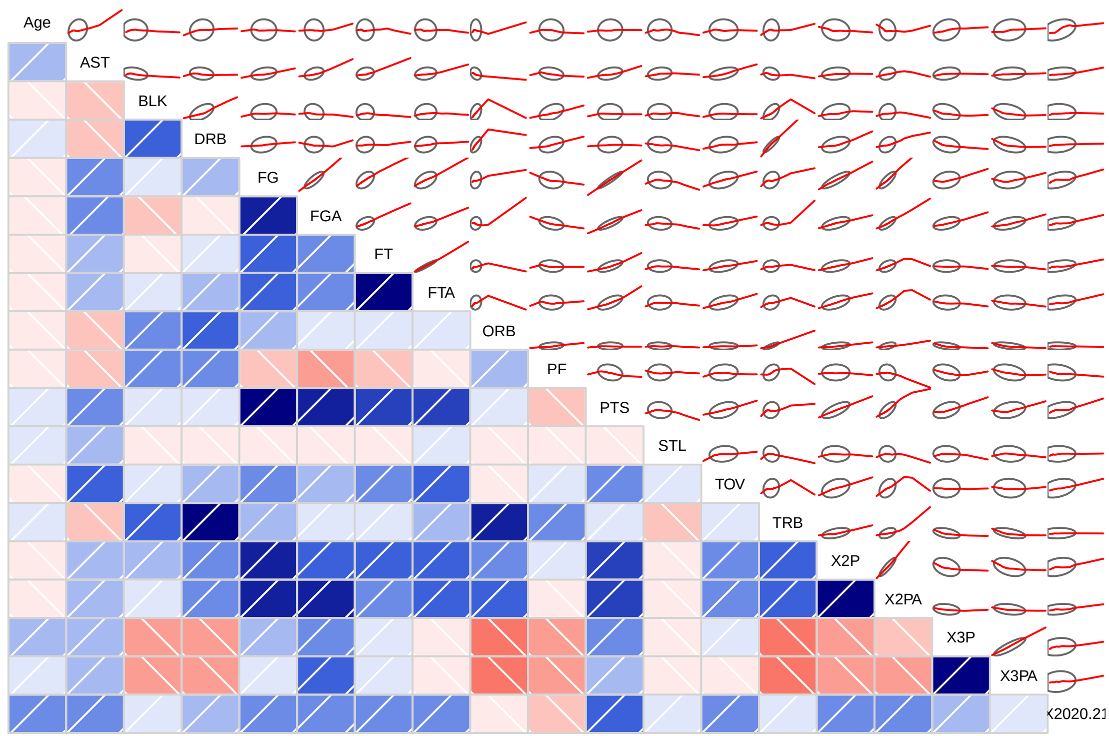


Fig. 3: Heat map of numerical data

Key insights: 1) Some of the features are highly correlated to each other (e.g. FG-PTS, DRB-TRB). These correlations need to test for multi-co-linearity.
2) To prevent misleading results, highly correlated variables are better not used together in interpretive models.

Linear correlation with dependent variable

```
cgram["X2020.21",11:19] %>% sort() %>% t() %>% round(3) %>% kable()
```

STL	TRB	X3PA	X3P	X2P	X2PA	TOV	PTS	X2020.21
0.031	0.077	0.143	0.185	0.297	0.301	0.355	0.473	1

```
cgram["X2020.21",1:10] %>% sort() %>% t() %>% round(3) %>% kable()
```

PF	ORB	BLK	DRB	AST	Age	FGA	FTA	FG	FT
-0.241	-0.055	0.04	0.165	0.317	0.365	0.382	0.398	0.408	0.411

Key insight: With 47.25% variable PTS shows highest correlation to dependent variable X2020.21.

3.1.2 Categorical data: Position

How differ salaries of players in different positions from each other? This can be displayed by box-plotting position vs. salary.

```
train_norm %>% ggplot( aes(x = Pos, y=log(X2020.21))) + geom_boxplot(na.rm = TRUE)
```

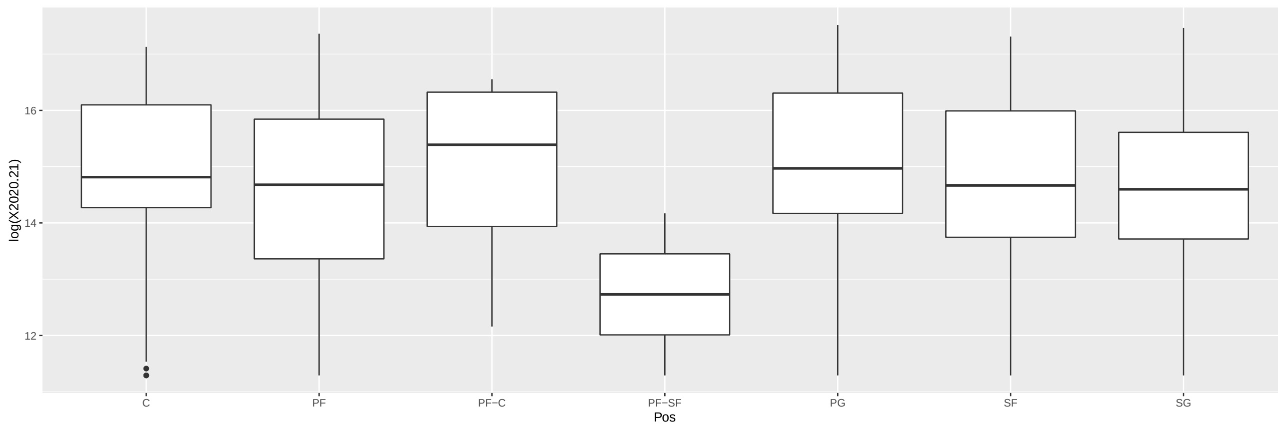


Fig. 4: Boxplots salary (log-transformed) vs. position

Key insight: It seems like players in PF-SF position - on average - earn less than players in other positions. Let's see if adding player position will improve the linear model on the overall.

Creating a base model without effect on other features

```
lm_0 <- lm(log(X2020.21) ~ 1, data = train_norm)  ## Linear model with 1 feature position
lm_pos <- lm(log(X2020.21) ~ Pos, data = train_norm) ## Comparing model with position to base model
anova(lm_0, lm_pos)
```

```
## Analysis of Variance Table
##
## Model 1: log(X2020.21) ~ 1
## Model 2: log(X2020.21) ~ Pos
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    440 1287.0
## 2    434 1272.8  6    14.108 0.8017 0.5689
```

Key insights: 1) Adding players' position to the model does not have a statistically significant impact on the model performance.

2) Players in position PF-SF - on average - earn less than players in other positions.

3.1.3 Categorical data: Team

How differ salaries of different teams from each other? This can be displayed by box-plotting team vs. salary.

```r
## Boxplot team vs salary
train_norm %>% ggplot( aes(x = Tm, y=log(X2020.21))) + geom_boxplot(na.rm = TRUE)
```

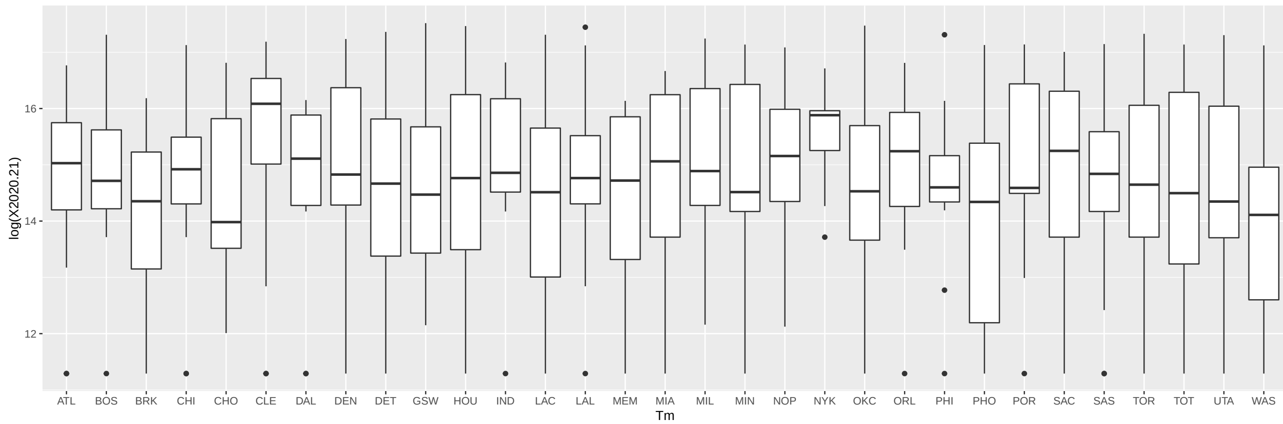


Fig.5: Boxplots of salary (log-transformed) vs. team

Linear model with 1 feature team

```r
lm_team <- lm(log(X2020.21) ~ Tm, data = train_norm)
anova(lm_0, lm_team)  ## Comparing model with team to base model
```

```
## Analysis of Variance Table
##
## Model 1: log(X2020.21) ~ 1
## Model 2: log(X2020.21) ~ Tm
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1    440 1287.0
## 2    410 1239.5 30     47.47 0.5234 0.9833
```

Key insights: 1) Adding players' team to the model doesn't have a statistically significant impact on the model performance.
2) From Fig 5 we can see a significant change of variance in the players' salaries between the teams. (E.g. the variance in salaries among players in NYK is smaller then the variance of salaries among players in PHO)

3.1.4 Scatter plots

Highest linear correlated features on dependent variable X2020.21 by order (before log-transformation):

"PTS"- Points
"FT"- Free throws
"FG"- Field goal
"FTA"- Free throws attempts
"FGA"- Field goal attempts
"Age"- age
"TOV"- Turnovers
"AST"- Assists
"X2PA"- 2 point field goals attempts
"X2P"- 2 point field goals
"X3P"- 3 point field goals
"DRB"- Defensive rebounds
"X3PA"- 3 point field goals attempts
"TRB"- Total rebounds
"BLK"- Blocks
"STL"- Steals
"ORB"- Offensive rebounds
"PF"- Personal fouls

Scatter plots will be created of all the numerical variables with the log(salaries) as dependent variable.

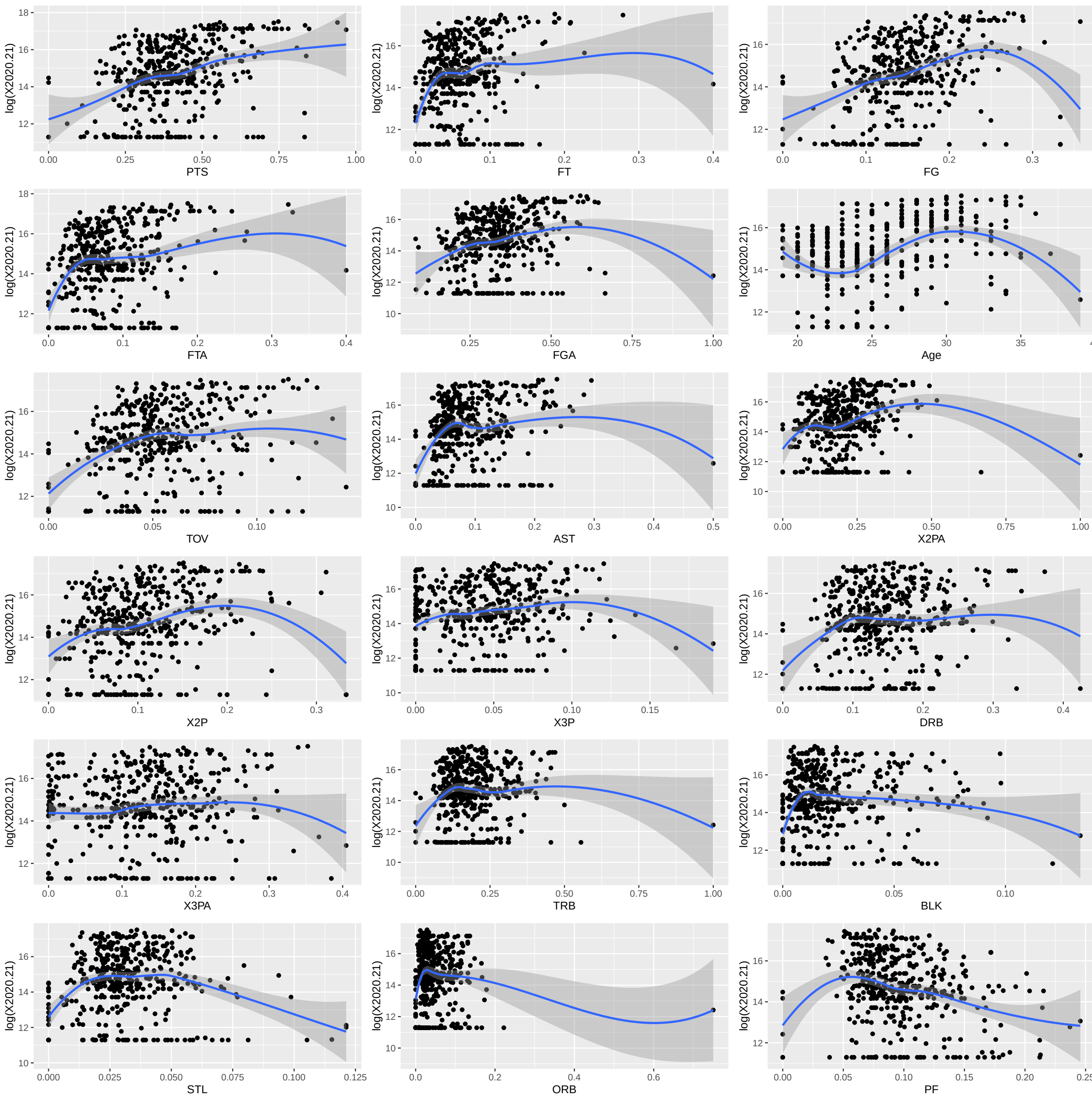


Fig.6: Scatter plot of numerical features with log-transformed salaries

Key insights: 1) It is visible that the outliers affect the correlation between the variables.
2) From the density plots, it is also known that most of the feature variables are right-skewed.
3) As a next step, let's first log-transform the skewed variables and see if it reduces the outliers.

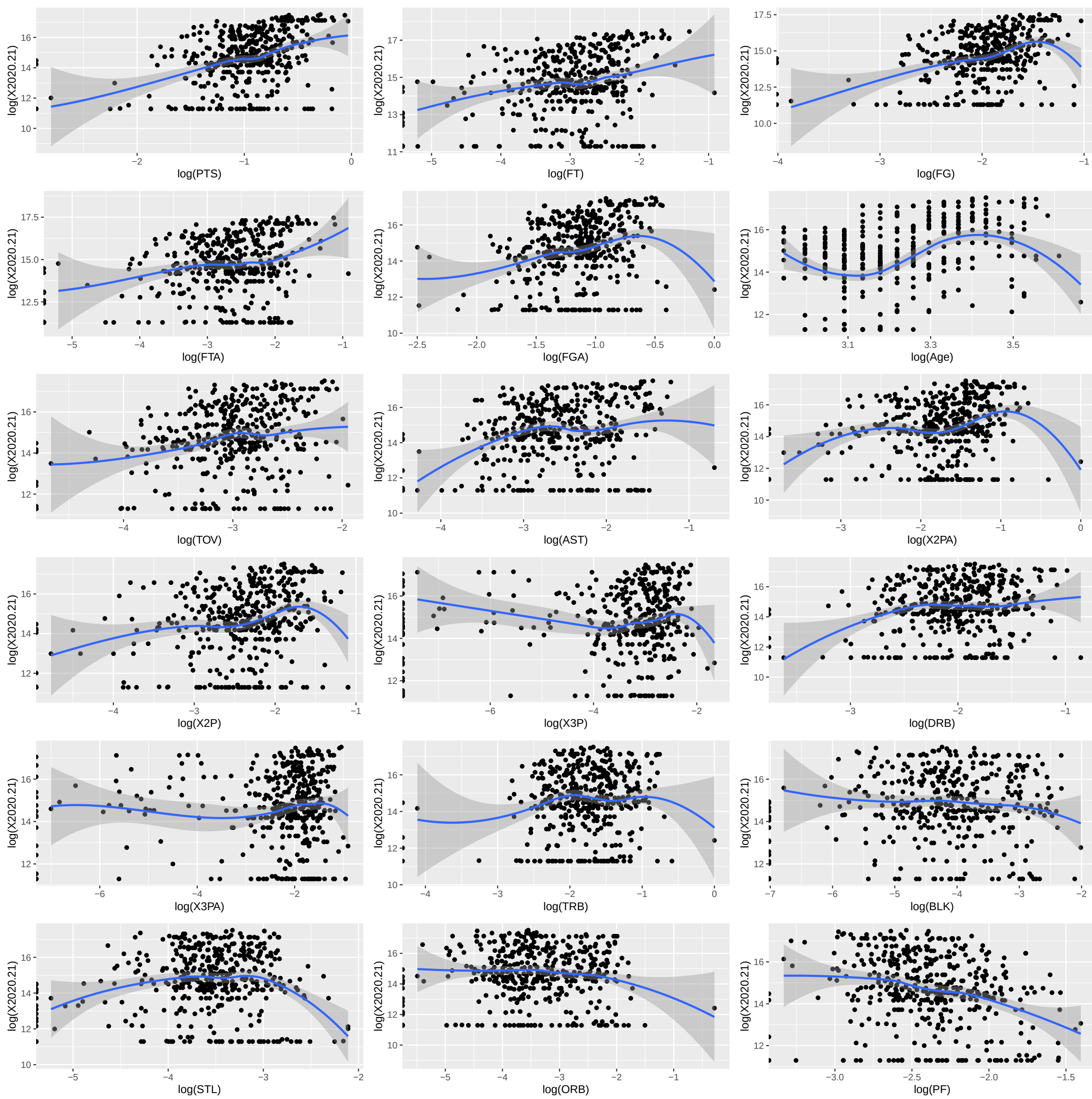


Fig.7: Scatter plots of numerical features after log-transformation vs. log-transformed salaries

Key insights: 1) Most variables show a better correlation to the dependent variable after log-transformation.
2) From Fig 3, extremely high correlation between several of the features is evident.
FG-PTS-FG-FGA, DRB-TRB should be examined for multi-co-linearity if used in the same model.

Moving on, following variables will be checked for our initial model, taking into consideration their linear correlation with the dependent variable, possible multi-co-linearity, prior knowledge and assumptions.

Variables selection for base model so far:

log(PTS), log(FT),FTA, FGA, AGE, log(TOV), log(AST), log(X2P), log(PF), Tm, Pos, Tm:FT + Pos:PF + Pos:FT

3.1.5 Removing outliers from feature FGA

At first some of the outliers which seem to have higher effect on the correlation will be removed.

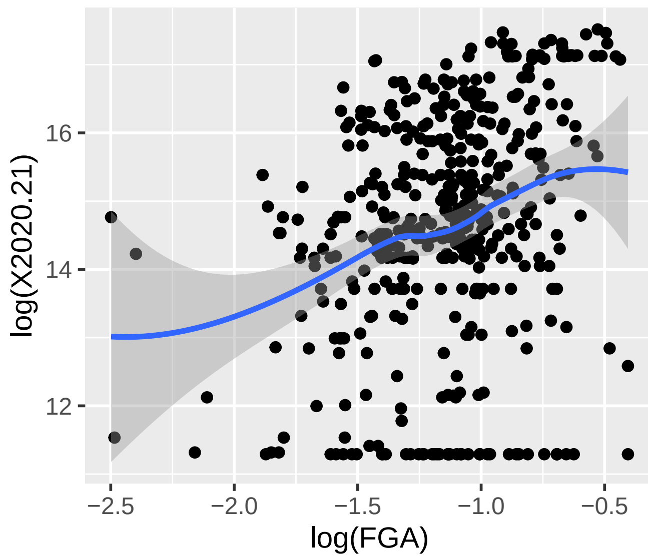


Fig.8: Scatter plot of features after removing of outliers

3.2 Models

As mentioned before, for the initial model, the following features will be used:
log(PTS), log(FT),FTA, FGA, AGE, log(TOV), log(AST), log(X2P), log(PF), Tm, Pos, Tm:FT, Pos:PF, Pos:FT

In case the minimum value of a feature which needs to be transformed is 0, we added half of the smallest value that is not 0 in the feature to all values. This allows for the transformation. (e.g log(PTS+ min(PTS[PTS>0])/2))

3.2.1 Base Model - linear regression

Single term deletion with drop1 function will be used until we reach the most parsimonious model.

```
linear_0 <- lm(log(X2020.21) ~ 1, train_norm)

linear_1 <- lm(log(X2020.21) ~ log(PTS+ min(PTS[PTS>0])/2) + log(FT+ min(FT[FT>0])/2) +
                 log(FTA+ min(FTA[FTA>0])/2) + log(FGA) + Age +
                 log(TOV + min(TOV[TOV>0])/2) + log(AST+ min(AST[AST>0])/2) +
                 log(X2P + min(X2P[X2P>0])/2)
               + log(PF+ min(PF[PF>0])/2) + Tm + Pos + Tm:FT  + Pos:PF + Pos:FT
                 , train_norm)

drop1(linear_1, test = "F")
```

```
## Single term deletions
##
## Model:
## log(X2020.21) ~ log(PTS + min(PTS[PTS > 0])/2) + log(FT + min(FT[FT >
##     0])/2) + log(FTA + min(FTA[FTA > 0])/2) + log(FGA) + Age +
##     log(TOV + min(TOV[TOV > 0])/2) + log(AST + min(AST[AST >
##     0])/2) + log(X2P + min(X2P[X2P > 0])/2) + log(PF + min(PF[PF >
##     0])/2) + Tm + Pos + Tm:FT + Pos:PF + Pos:FT
##                                   Df Sum of Sq     RSS     AIC F value   Pr(>F)
```

```
## <none>                                          583.85 302.46
## log(PTS + min(PTS[PTS > 0])/2)   1      0.004 583.86 300.47  0.0024 0.960615
## log(FT + min(FT[FT > 0])/2)      1     16.150 600.00 312.47  9.7090 0.001985 **
## log(FTA + min(FTA[FTA > 0])/2)   1      1.000 584.85 301.22  0.6012 0.438647
## log(FGA)                         1      3.921 587.77 303.41  2.3572 0.125608
## Age                              1     79.252 663.10 356.47 47.6450 2.40e-11 ***
## log(TOV + min(TOV[TOV > 0])/2)   1     34.099 617.95 325.44 20.4998 8.18e-06 ***
## log(AST + min(AST[AST > 0])/2)   1      3.159 587.01 302.84  1.8989 0.169079
## log(X2P + min(X2P[X2P > 0])/2)   1      0.284 584.14 300.68  0.1705 0.679933
## log(PF + min(PF[PF > 0])/2)      1      0.189 584.04 300.61  0.1138 0.736066
## Tm:FT                           31     60.381 644.23 283.77  1.1710 0.247787
## Pos:PF                           6     19.715 603.57 305.08  1.9754 0.068448 .
## Pos:FT                           5     20.802 604.65 307.87  2.5012 0.030410 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
linear_1_1 <- update(linear_1, .~. -log(PF + min(PF[PF > 0])/2))
drop1(linear_1_1, test = "F")
```

```
linear_1_2 <- update(linear_1_1, .~. -log(X2P + min(X2P[X2P > 0])/2))
drop1(linear_1_2, test = "F")
```

```
linear_1_3 <- update(linear_1_2, .~. -log(FTA + min(FTA[FTA > 0])/2))
drop1(linear_1_3, test = "F")
```

```
linear_1_4 <- update(linear_1_3, .~. -log(PTS + min(PTS[PTS > 0])/2))
drop1(linear_1_4, test = "F")
```

```
linear_1_5 <- update(linear_1_4, .~. -Tm:FT)
drop1(linear_1_5, test = "F")
```

```
linear_1_6 <- update(linear_1_5, .~. -Tm)
drop1(linear_1_6, test = "F")
```

```
linear_1_7 <- update(linear_1_6, .~. -Pos:FT)
drop1(linear_1_7, test = "F")
```

```
linear_1_8 <- update(linear_1_7, .~. -log(AST + min(AST[AST > 0])/2))
drop1(linear_1_8, test = "F")
```

```
## Single term deletions
##
## Model:
## log(X2020.21) ~ log(FT + min(FT[FT > 0])/2) + log(FGA) + Age +
##     log(TOV + min(TOV[TOV > 0])/2) + Pos + Pos:PF
##                                 Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                                       724.33 255.33
## log(FT + min(FT[FT > 0])/2)      1    56.861 781.19 286.58 33.1278 1.662e-08 ***
## log(FGA)                         1    17.480 741.81 263.82 10.1840  0.001523 **
## Age                              1   125.455 849.78 323.61 73.0910 2.297e-16 ***
## log(TOV + min(TOV[TOV > 0])/2)   1    66.054 790.38 291.73 38.4837 1.319e-09 ***
## Pos:PF                           7   107.333 831.66 302.13  8.9333 2.802e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Key insight: After executing drop1 function several times, all features left are statistically significant for the model.

Examining base model for multi-co-linearity

```
##                                          GVIF Df GVIF^(1/(2*Df))
## log(FT + min(FT[FT > 0])/2)    1.294433e+00  1        1.137731
## log(FGA)                       1.369989e+00  1        1.170465
## Age                            1.031329e+00  1        1.015544
## log(TOV + min(TOV[TOV > 0])/2) 1.381728e+00  1        1.175469
## Pos                            4.396619e+06  6        3.577612
## Pos:PF                         5.115667e+06  7        3.014445
```

Key insight: Potential multi-co-linearity issue with Pos:PF and PF can be seen (highest GVIF values). Feature "Pos" will be dropped from the model as well.

```
## Single term deletions
##
## Model:
## log(X2020.21) ~ log(FT + min(FT[FT > 0])/2) + log(FGA) + Age +
##     log(TOV + min(TOV[TOV > 0])/2) + Pos:PF
##                                Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                                      738.09 251.61
## log(FT + min(FT[FT > 0])/2)     1    55.631 793.72 281.58 32.2592 2.495e-08 ***
## log(FGA)                        1    14.763 752.85 258.32  8.5608  0.003618 **
## Age                             1   121.871 859.96 316.85 70.6705 6.327e-16 ***
## log(TOV + min(TOV[TOV > 0])/2)  1    67.704 805.79 288.22 39.2600 9.053e-10 ***
## Pos:PF                          7   105.876 843.96 296.59  8.7707 4.333e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Multi-co-linearity is re-examined:

```
##                                    GVIF Df GVIF^(1/(2*Df))
## log(FT + min(FT[FT > 0])/2)    1.287271  1        1.134580
## log(FGA)                       1.337401  1        1.156461
## Age                            1.019315  1        1.009611
## log(TOV + min(TOV[TOV > 0])/2) 1.332636  1        1.154398
## Pos:PF                         1.388577  7        1.023726
```

Key insight: The multi-co-linearity is no longer evident.

Base Model interpretation

```
summary(linear_1_9)$coefficient
```

```
##                                   Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)                     17.1905997  0.60544242 28.393451 1.783452e-100
## log(FT + min(FT[FT > 0])/2)      0.4691262  0.08259678  5.679715  2.495066e-08
## log(FGA)                         0.6528790  0.22313868  2.925889  3.617558e-03
## Age                              0.1346971  0.01602282  8.406576  6.326627e-16
## log(TOV + min(TOV[TOV > 0])/2)   0.7842147  0.12515842  6.265777  9.053412e-10
## PosC:PF                        -11.0692985  1.92385040 -5.753721  1.665160e-08
## PosPF:PF                       -15.4119019  2.44057846 -6.314856  6.778473e-10
## PosPF-C:PF                     -10.3133890  4.55207415 -2.265646  2.397192e-02
```

```
## PosPF-SF:PF                    -43.8828997 12.44373582 -3.526505  4.665795e-04
## PosPG:PF                       -21.5763530  3.33145609 -6.476553  2.580124e-10
## PosSF:PF                       -13.6431571  2.55453814 -5.340753  1.506253e-07
## PosSG:PF                       -18.7343301  2.63227453 -7.117164  4.675579e-12
```

Key insights: 1) If a player increases his successful free throws per minute by 1%, his salary will increase by 0.469%
2) If a player increases his free goal attempts per minute by 1%, his salary will increase by 0.65%
3) For each increase of 1 year of age, a player's salary will increase by 14% (exp(0.1346971)=1.14419)
4) If a player will increase his Turn overs per minute by 1%, his salary will increase by 0.78%
5) There is strong evidence that personal fouls have an effect on salary and this effect differs among the players' position.

Base Model performance on training data

Setting control parameters for cross validation

```r
ctrl <- trainControl(method = "cv", number = 10, verboseIter = TRUE)
```

```r
linear_1_9_cv <- train(log(X2020.21) ~ log(FT + min(FT[FT > 0])/2) + log(FGA) + Age +
    log(TOV + min(TOV[TOV > 0])/2) + Pos:PF,
    train_norm, method = "lm", trControl = ctrl)
```

10 folds cross-validation results

```
##   intercept     RMSE  Rsquared    MAE    RMSESD RsquaredSD    MAESD
## 1      TRUE 1.336649 0.3881333 1.0541 0.1281804  0.1189035 0.124647
```

Base Model performance on testing data

Using the model to predict salary of players in test data .

```
## [1] "Model evaluation"
```

```
## [1] "RMSE: 1.59379756028198"
```

```
## [1] "R2: 0.214038202324226"
```

Key insights: RMSE of 1.59 and R2 of 0.21 do not say much without comparison to another model. Another model is needed for comparison.

3.2.2 General additive model (GAM)

Setting GAM with the same initial features as the linear model, this time without the interactions. Features transformation would be determined by the model:
PTS, FT,FTA, FGA, AGE, TOV, AST, X2P, PF, Tm, Pos

```r
library(mgcv)
gam.0 <- gam(log(X2020.21) ~ s(PTS) + s(FT) + s(FGA) + s(Age) + s(TOV) + s(AST) +
               s(X2P) + s(PF) ,
              data = train_norm,
             select = TRUE) ## select=T similar to lasso selection pulling less important parameter coe
summary(gam.0)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(X2020.21) ~ s(PTS) + s(FT) + s(FGA) + s(Age) + s(TOV) + s(AST) +
##     s(X2P) + s(PF)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.6467     0.0593     247   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df      F  p-value
## s(PTS) 2.944e+00      9  1.029 0.010347 *
## s(FT)  6.961e+00      9  3.562 1.07e-05 ***
## s(FGA) 1.318e-06      9  0.000 0.543580
## s(Age) 5.964e+00      9 13.008  < 2e-16 ***
## s(TOV) 3.127e+00      9  2.184 7.30e-05 ***
## s(AST) 3.814e+00      9  0.970 0.036415 *
## s(X2P) 3.878e+00      9  1.862 0.000655 ***
## s(PF)  9.795e-01      9  3.787  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.47   Deviance explained = 50.4%
## GCV = 1.6549  Scale est. = 1.5471    n = 440
```

Key insight: Feature FGA is not statistically significant.
We will remove it for further evaluation.

```
gam.1 <- update(gam.0, .~. -s(FGA))
summary(gam.1)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(X2020.21) ~ s(PTS) + s(FT) + s(Age) + s(TOV) + s(AST) + s(X2P) +
##     s(PF)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 14.64670    0.05931   246.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##           edf Ref.df      F  p-value
## s(PTS) 2.5695      9  0.952 0.010219 *
## s(FT)  7.0023      9  3.574 1.06e-05 ***
## s(Age) 6.0538      9 13.013  < 2e-16 ***
## s(TOV) 3.0794      9  2.179 7.04e-05 ***
```

```
## s(AST) 3.8246      9  0.955 0.039526 *
## s(X2P) 3.8295      9  1.817 0.000747 ***
## s(PF)  0.9865      9  3.826  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.47   Deviance explained = 50.3%
## GCV = 1.6546  Scale est. = 1.548     n = 440
```

Looking for co-linearity issues (using concurvity)

```
##                  para    s(PTS)     s(FT)    s(Age)    s(TOV)    s(AST)
## worst    3.996889e-20 0.8806880 0.8353133 0.9730735 0.5962755 0.9741914
## observed 3.996889e-20 0.8175069 0.4051482 0.1715596 0.4904704 0.5437596
## estimate 3.996889e-20 0.7379383 0.5584378 0.1958769 0.5126933 0.5666816
##             s(X2P)     s(PF)
## worst    0.6905576 0.5556744
## observed 0.4843881 0.4181534
## estimate 0.5864721 0.3809157
```

Key insight: PTS feature shows high co-linearity with other features in the model. We will drop it from our model.

```
gam.2 <- update(gam.1, .~. -s(PTS))
summary(gam.2)
```

After removing PTS from our model, we observed high co-linearity of AST feature, therefore we repeated the process again.

```
gam.3 <- update(gam.2, .~. -s(AST))
summary(gam.3)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## log(X2020.21) ~ s(FT) + s(Age) + s(TOV) + s(X2P) + s(PF)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 14.64670    0.06025   243.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df      F  p-value
## s(FT)  7.162      9  4.749 4.53e-07 ***
## s(Age) 4.771      9 12.680  < 2e-16 ***
## s(TOV) 3.650      9  3.241 1.83e-06 ***
## s(X2P) 3.774      9  2.770 1.41e-05 ***
## s(PF)  1.000      9  5.916  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.453   Deviance explained = 47.8%
## GCV = 1.6785  Scale est. = 1.5971    n = 440
```

```
##                    para     s(FT)    s(Age)    s(TOV)    s(X2P)     s(PF)
## worst    2.172459e-20 0.5189045 0.2162250 0.4702628 0.4600148 0.5020866
## observed 2.172459e-20 0.2408868 0.1072955 0.3766206 0.3510334 0.1682462
## estimate 2.172459e-20 0.3431662 0.1052759 0.3595019 0.3553587 0.1770138
```

Key Insight: No more multi-co-linearity issues.

GAM Model performance 10 folds cross-validation

```r
gam.3_cv <- train(log(X2020.21) ~ FT + Age + TOV + X2P + PF, family = "gaussian",
   data = train_norm, method = "gam", trControl = ctrl)
```

```
##   select method     RMSE  Rsquared      MAE    RMSESD RsquaredSD     MAESD
## 1  FALSE GCV.Cp 1.295497 0.4342578 1.020755 0.1605039  0.1279610 0.1540144
## 2   TRUE GCV.Cp 1.314392 0.4279436 1.029704 0.1940030  0.1402875 0.1634121
```

Model prediction and evaluation on test data

```r
#Use model to predict probability of default
predicted <- predict(gam.3_cv, test_norm)
print("Model evaluation")
```

```
## [1] "Model evaluation"
```

```r
print(paste0("RMSE: ", RMSE(predicted, log(test_norm$X2020.21))))
```

```
## [1] "RMSE: 1.46876451096835"
```

```r
print(paste0("R2: ", R2(predicted, log(test_norm$X2020.21))))
```

```
## [1] "R2: 0.309250594296559"
```

Key insight: We can see a small improvement in RMSE and R2 values relative to the base linear model.

4 Classification problems

In this section, we will use a predictive modelling approach. The model will be trained with all available features, and the results will be compared using a confusion matrix and corresponding evaluation matrices.

Our predictive model will try to evaluate whether or not a player's salary would be higher than the median players' salaries or not in the following year.

4.1 Preparing the data

A new binary column based on median salary of players is created.

```
train_norm$binarSalary <- ifelse(train_norm$X2020.21 > median(train_norm$X2020.21), 1,0)
test_norm$binarSalary <- ifelse(test_norm$X2020.21 > median(test_norm$X2020.21), 1,0)
```

```
train_norm %>% dplyr :: select(X2020.21, binarSalary) %>% head()
```

4.2 Generalised linear model - Binomial

```
## 10 folds cross-validation
binom.1_cv <- train(binarSalary ~ . -X2020.21, family = "binomial",
   data = train_norm, method = "glm", trControl = ctrl) ## 10 folds cross validation
```

10 folds cross-validation results

```
##   parameter      RMSE  Rsquared       MAE     RMSESD RsquaredSD      MAESD
## 1      none 0.4764108 0.1549575 0.3958872 0.03939175 0.09568152 0.03697523
```

Model evaluation on test data- confusion Matrix

```
#Use model to predict probability of default
predicted <- predict(binom.1_cv, test_norm)

#Find optimal cutoff probability to use to maximize accuracy
optimal <- InformationValue:: optimalCutoff(test_norm$binarSalary, predicted)[1]

predicted_value <- ifelse(predicted > optimal, 1,0)

#Create confusion matrix
binom.1_cm <- caret :: confusionMatrix(factor(test_norm$binarSalary), factor(predicted_value))
binom.1_cm$table
```

```
##           Reference
## Prediction  0  1
##          0 41 15
##          1 15 40
```

4.3 GAM with familiy set to Binomial

```r
binom_2 <- gam(binarSalary ~ Tm +Pos + s(Age) + s(X3PA) + s(X3P) + s(X2PA) + s(X2P) + s(TRB) +
                 s(TOV) + s(STL) + s(PTS) + s(PF) + s(ORB) + s(FTA) + s(FT) + s(FGA) +
                 s(FG) + s(DRB) + s(BLK) + s(AST),
family = "binomial",
data = train_norm)
```

10 folds cross-validation results

```
##   select method      RMSE  Rsquared       MAE     RMSESD RsquaredSD      MAESD
## 1  FALSE GCV.Cp 0.4612262 0.2163291 0.3534971 0.04878676  0.1192904 0.04217100
## 2   TRUE GCV.Cp 0.4596814 0.2226307 0.3535870 0.05002932  0.1242140 0.03878292
```

Model evaluation on test data- confusion Matrix

```
##           Reference
## Prediction  0  1
##          0 40 16
##          1 13 42
```

4.4 Supervised vector machine model

Cross validation on SVM model was not performed due to lack of sufficient computer power.

```r
svm.1 <- svm(binarSalary ~., train_norm, kernel = "linear", scale = TRUE, cost = 10)
```

Model evaluation on test data- confusion Matrix

```
##           Reference
## Prediction  0  1
##          0 45 11
##          1  6 49
```

4.5 Neural network model

One hot encoding for categorical data

```r
train_norm_dummy <- data.frame(train_norm[, !colnames(train_norm) %in% c("Tm", "Pos")],
                               model.matrix(~Tm +Pos -1, train_norm))
test_norm_dummy <- data.frame(test_norm[, !colnames(test_norm) %in% c("Tm", "Pos")],
                               model.matrix(~Tm +Pos -1, test_norm))
```

Model training and optimizing with 5 folds cross validation

```r
tuneGrid <- expand.grid(.layer1=c(2:4), .layer2=c(0:4), .layer3=c(0))
control <- trainControl(method="cv", number=5)

NN.models <- train(train_norm_dummy %>% dplyr:: select(-c(X2020.21, binarSalary)),
                   train_norm_dummy %>% dplyr:: pull(binarSalary),
                   method="neuralnet",
                   metric = 'F1',
                   ### Parameters for optimization
```

```
                    preProcess = c('center', 'scale'),
                    tuneGrid = tuneGrid,
                    trControl = control,
                    tuneLength=5
                    )
```

Model evaluation on test data- confusion Matrix

```
##           Reference
## Prediction  0  1
##          0 41 15
##          1 23 32
```

4.6 Summary classification models

```
##     Model Precision    Recall        F1
## 1 binom.1 0.7321429 0.7321429 0.7321429
## 2 binom.2 0.7142857 0.7547170 0.7339450
## 3   svm.1 0.8035714 0.8823529 0.8411215
## 4    NN.1 0.7321429 0.6406250 0.6833333
```

Key insights: Better performance of Support Vector Machine model in terms of precision, recall and F1 evaluation matrices.
The model was able to predict correctly 49 out of 60 players with salaries above the median based on their performance in the previews season (see SVM model confusion matrix). In addition the model was able to predict correctly 45 out of 51 players with salary below the median.

5 Players' performance - Predicting FG

Predicting the number of FG (field goal) based on Pos, Age, Tm, X3PA, X2PA, FTA

5.1 Preparing data

```
train2 <- train[,c( 'Pos', 'Age', 'Tm', 'X3PA', 'X2PA', 'FTA', 'FG')]
test2 <- test[,c( 'Pos', 'Age', 'Tm', 'X3PA', 'X2PA', 'FTA', 'FG')]
```

5.2 Exploring data

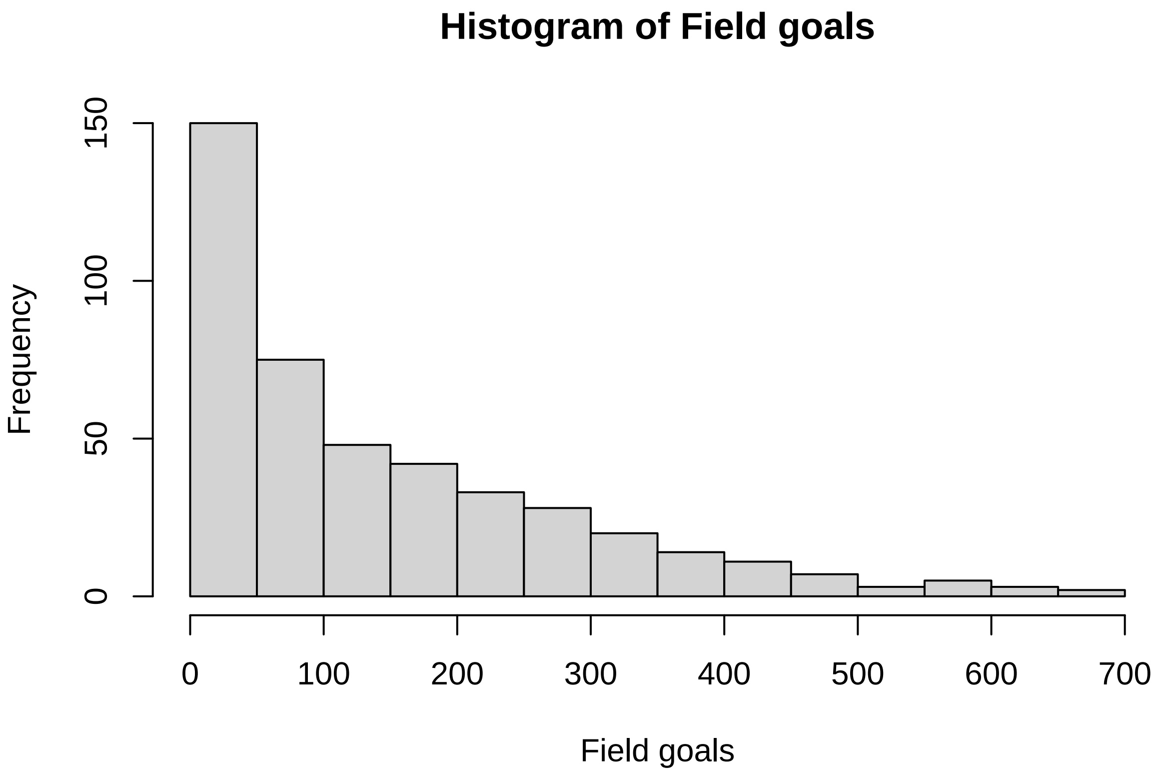


Fig.1: Histogram of FG

Key insights: Over dispersion of data is observed. (Long right tail distribution)

5.3 Categorical data

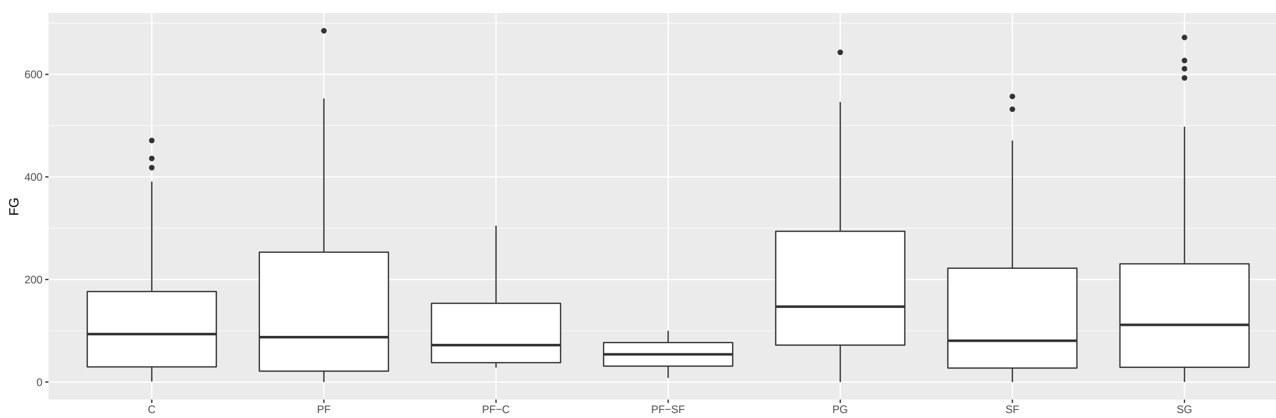


Fig.2: Boxplot of log FG by Position

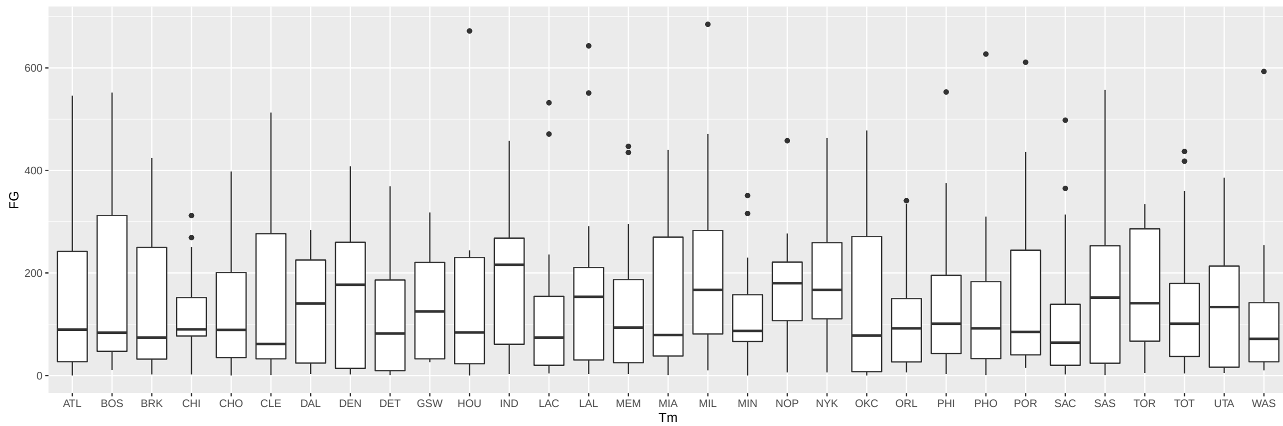


Fig.3: Boxplot of log FG by team

5.4 GLM model

Family set to poisson

```
pois_1 <- glm(FG ~ Tm +Pos + Age  + X3PA + X2PA + FTA,
family = "poisson", ## we specify the distribution!
data = train2)
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 59545  on 440  degrees of freedom
## Residual deviance: 10921  on 400  degrees of freedom
## AIC: 13676
##
## Number of Fisher Scoring iterations: 5
```

Key insight: The residual deviance is bigger than the degrees of freedom. This is indicative of over-dispersion of the dependent variable. We can also see this by the difference in mean and variance of the dependent variable.

We will redo the model, this time using the quasi-poisson model.

Family set to quasi-poisson

```
qpois_1 <- glm(FG ~ Tm +Pos + Age  + X3PA + X2PA + FTA,
family =  "quasipoisson", ## we specify the distribution!
data = train2)
```

```
drop1(qpois_1, test = "F")
```

```
## Single term deletions
##
## Model:
## FG ~ Tm + Pos + Age + X3PA + X2PA + FTA
##        Df Deviance  F value    Pr(>F)
## <none>       10920
## Tm     30    11682   0.9302  0.575330
## Pos     6    11437   3.1535  0.004936 **
## Age     1    10962   1.5357  0.215991
## X3PA    1    15434 165.3129 < 2.2e-16 ***
## X2PA    1    20046 334.2610 < 2.2e-16 ***
```

```
## FTA      1    11364  16.2231 6.738e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
qpois_2 <- update(qpois_1, .~. -Tm)
```

```
drop1(qpois_2, test = "F")
```

```
qpois_3 <- update(qpois_2, .~. -Age)
```

```
drop1(qpois_3, test = "F")
```

```
## Single term deletions
##
## Model:
## FG ~ Pos + X3PA + X2PA + FTA
##        Df Deviance  F value    Pr(>F)
## <none>       11691
## Pos     6    12245   3.4022  0.002728 **
## X3PA    1    16990 195.3506 < 2.2e-16 ***
## X2PA    1    25052 492.5483 < 2.2e-16 ***
## FTA     1    12188  18.3176 2.306e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
qpois_3$coefficients
```

```
##  (Intercept)        PosPF      PosPF-C     PosPF-SF        PosPG        PosSF
##  4.126903780 -0.216042961 -0.180192152 -0.446159125 -0.228226641 -0.228342023
##        PosSG         X3PA         X2PA          FTA
## -0.333887033  0.002251756  0.002831483 -0.000996727
```

Key insights: 1) All predictors are statistically significant
2) If a player increases his 3 points attempts by 1%, his overall field goals will increase by 0.0022%
3) If a player increases his 2 points attempts by 1%, his overall field goals will increase by 0.0028%
4) If a player increases his free throws attempts by 1%, his overall field goals will decrease by -0.0009%
(Interesting results, one explanation could be that players who go more to the line usually have less field goals)
5) There is relatively strong evidence that positions have an effect on player's field goals.

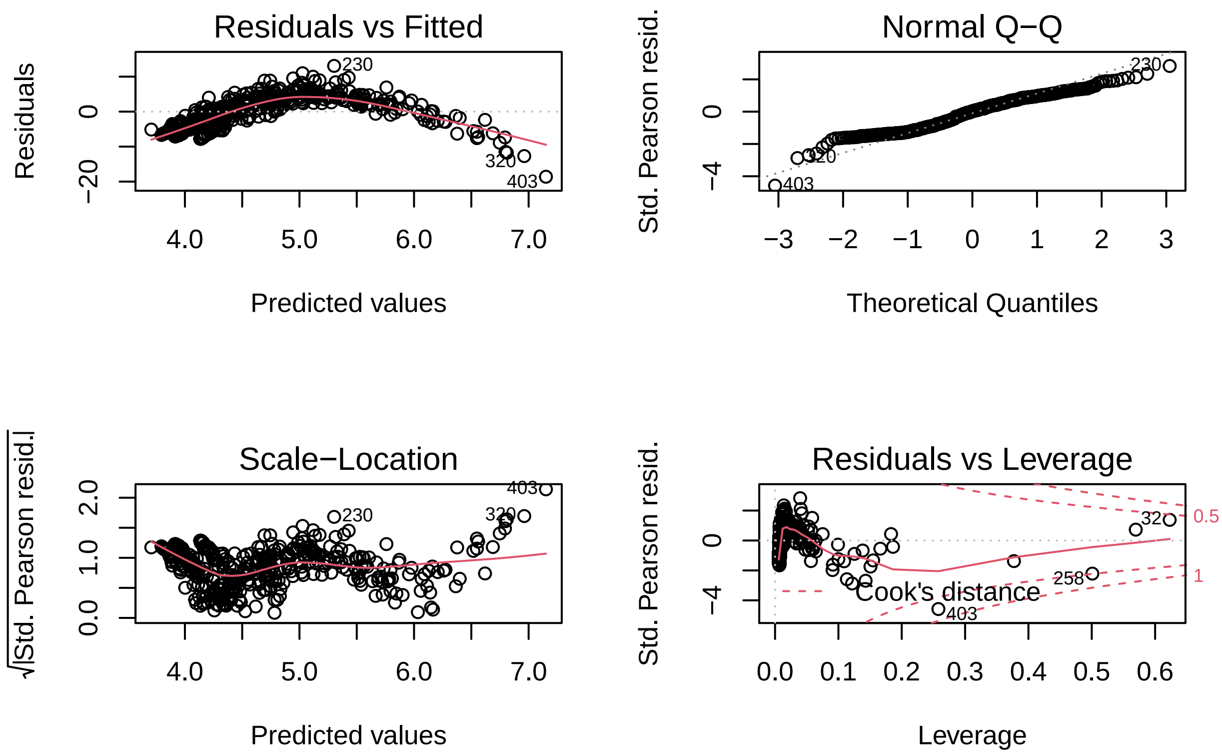
Model evaluation on test data

```
q_predict <- predict(qpois_3, test2, type="response")
RMSE(test2$FG, q_predict)
```

```
## [1] 88.74863
```

```
R2(test2$FG, q_predict)
```

```
## [1] 0.8021642
```



Key insights: 1) From the residuals plot we can clearly see a non-linear relationship in our model 2) The data looks fairly normally distributed

6 Optimization

With regard to the NBA dataset, the group did not come up with an idea on optimization. Instead a fictious case from the area of marketing is presented.
For a marketing campaign the aim is to reach maximum of listeners at given budget. The options are through radio (A) or television (B) adverts. 'A' can reach 7,000 people at CHF 600/min, 'B' can reach 50,000 people at CHF 9,000/min. The budget for the campaign is capped at CHF 100,000.

```
objective.in <- c(7000,50000)
const.mat <- matrix(c(600, 9000, 5,1), nrow=2,  byrow=TRUE)
const.rhs <- c(100000, 60)
const.dir <- c("<=", "<=")
optimum <- lp(direction="max", objective.in, const.mat, const.dir, const.rhs)
print(optimum$solution)
```

```
## [1]  9.90991 10.45045
```

Key insight: By choosing 9.91 units of A and 10.45 units of B we reach most listeners while not exceeding a budget of CHF 100k.