# R Notebook

Set seed to ensure reproducibility

```
set.seed(12)
```

## 1. Data Exploration and Preparation

### a) Load data, choose relevant column, and drop rows with NA values

Load the data

```
nba <- read.csv("nba_final.csv")
```

Take a peek at the data

```
head(nba)
```

```
##    Rk        Player.x Player_ID Pos1 Pos2 Age  Tm  G GS   MP  FG  FGA   FG. X3P
## 1 170    A.J. Hammons hammoaj01    C <NA>  24 DAL 22  0  7.4 0.8  1.9 0.405 0.2
## 2  58    Aaron Brooks brookaa01   PG <NA>  32 IND 65  0 13.8 1.9  4.6 0.403 0.7
## 3 157    Aaron Gordon gordoaa01   SF <NA>  21 ORL 80 72 28.7 4.9 10.8 0.454 1.0
## 4 352   Adreian Payne paynead01   PF <NA>  25 MIN 18  0  7.5 1.3  3.0 0.426 0.2
## 5  10 Al-Farouq Aminu aminual01   PF <NA>  26 POR 61 25 29.1 3.0  7.6 0.393 1.1
## 6 203      Al Horford horfoal01    C <NA>  30 BOS 68 68 32.3 5.6 11.8 0.473 1.3
##   X3PA  X3P. X2P X2PA  X2P.  eFG.  FT FTA   FT. ORB DRB TRB AST STL BLK TOV  PF
## 1  0.5 0.500 0.5  1.5 0.375 0.464 0.4 0.9 0.450 0.4 1.3 1.6 0.2 0.0 0.6 0.5 1.0
## 2  2.0 0.375 1.1  2.6 0.424 0.483 0.5 0.6 0.800 0.3 0.8 1.1 1.9 0.4 0.1 1.0 1.4
## 3  3.3 0.288 4.0  7.5 0.528 0.499 2.0 2.7 0.719 1.5 3.6 5.1 1.9 0.8 0.5 1.1 2.2
## 4  0.8 0.200 1.1  2.2 0.513 0.454 0.8 1.1 0.737 0.5 1.3 1.8 0.4 0.4 0.4 0.4 1.8
## 5  3.5 0.330 1.9  4.2 0.445 0.468 1.6 2.2 0.706 1.3 6.1 7.4 1.6 1.0 0.7 1.5 1.7
## 6  3.6 0.355 4.3  8.2 0.524 0.527 1.6 2.0 0.800 1.4 5.4 6.8 5.0 0.8 1.3 1.7 2.0
##    PTS   Salary mean_views  Season Conference  Role  Fvot FRank Pvot PRank Mvot
## 1  2.2       NA    3.32000 2016-17       West Front   786   123   NA    NA   NA
## 2  5.0  2700000   11.15574 2016-17        Est  Back  2474    64   NA    NA   NA
## 3 12.7  4351320 1713.98634 2016-17        Est Front 22774    29   NA    NA   NA
## 4  3.5  2022240  205.85519 2016-17       West Front   861   120    1    52   NA
## 5  8.7  7680965  604.34153 2016-17       West Front  4971    69    7    23   NA
## 6 14.0 26540100 1556.38251 2016-17        Est Front 12219    14    8    16    2
##   MRank Score Play
## 1    NA  83.5   No
## 2    NA  48.2   No
## 3    NA  40.0   No
## 4    NA  75.5   No
## 5    NA  42.8   No
## 6     6  12.5   No
```

Select only the columns we're interested in

```
nba <- nba[, c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL", "Pos1", "Play", "Salary")]
```

Check initial number of rows

```
nrow(nba)
```

```
## [1] 1408
```

See which column has missing data

```
missing_summary <- colSums(is.na(nba))
print(missing_summary)
```

```
##    PTS    AST    BLK    TRB   eFG.     MP    STL   Pos1   Play Salary
##      0      0      0      0      4      0      0      0      0     62
```

Clean missing data and check the number of rows again

```
nba <- na.omit(nba)
print(nrow(nba))
```

```
## [1] 1342
```

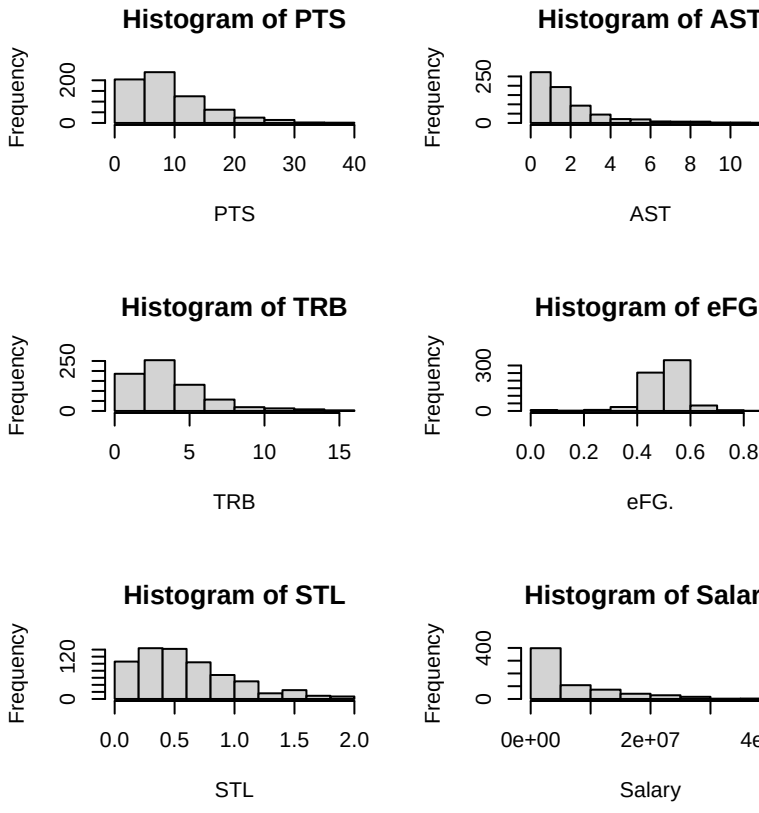
66 observations (from 1408) with missing values were dropped, leaving 1342 data

**b) Split the dataset into training and test dataset (each 50%)**

```
s <- sample(1:nrow(nba), nrow(nba)/2, replace=F)
train <- nba[s,]
test <- nba[-s,]
```

**c) Data visualization: see distribution of response variable and all predictor variables, as well as outliers**

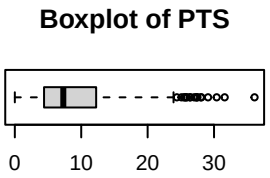
```
numeric_vars <- c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL", "Salary")
par(mfrow = c(3, 3))
for (var in numeric_vars) {
  hist(train[[var]], main = paste("Histogram of", var), xlab = var)
}
```

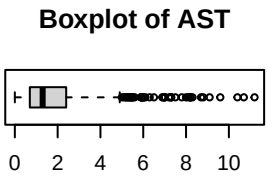


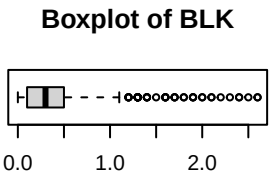

**i. Histogram for response variable and numeric predictors**

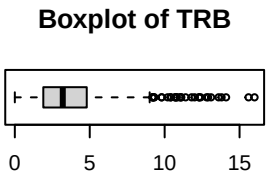
#### ii. Boxplot for response variable and numeric predictors

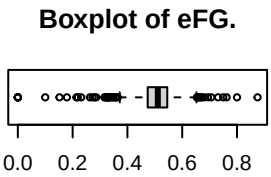
```
par(mfrow = c(3, 3))
for (var in numeric_vars) {
  boxplot(train[[var]], main = paste("Boxplot of", var), horizontal = TRUE)
}
```

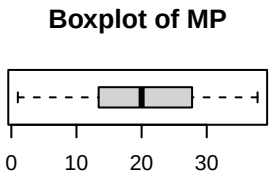


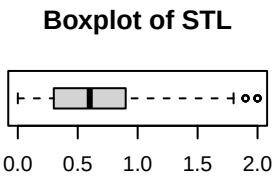



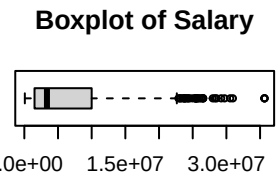



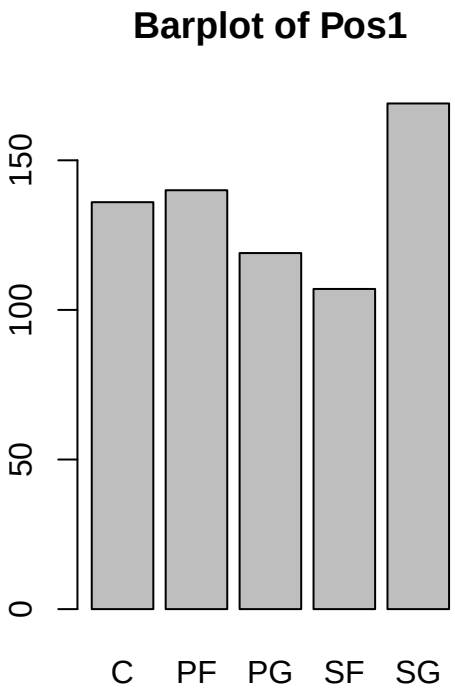



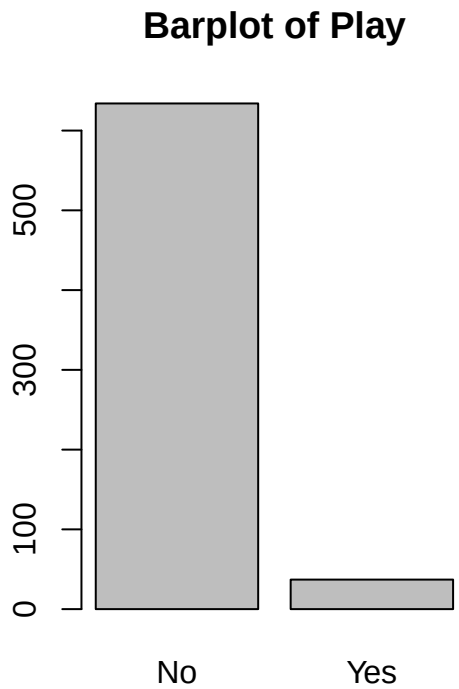




#### iii. Barplot for categorical predictors

```
par(mfrow = c(1, 2))
for (var in c("Pos1", "Play")) {
  barplot(table(train[[var]]), main = paste("Barplot of", var))
}
```




### d) Check outliers for all variables (more extreme than 1.5 x IQR)

```
find_outliers <- function(x) {
  q1 <- quantile(x, 0.25)
  q3 <- quantile(x, 0.75)
  iqr <- q3 - q1
  lower_bound <- q1 - 1.5 * iqr
  upper_bound <- q3 + 1.5 * iqr
  outliers <- x[x < lower_bound | x > upper_bound]
```

```
  return(paste(length(outliers), "outliers"))
}

sapply(numeric_vars, function(var) find_outliers(train[[var]]))
```

```
##           PTS           AST           BLK           TRB          eFG.
## "18 outliers" "47 outliers" "40 outliers" "31 outliers" "43 outliers"
##            MP           STL        Salary
##  "0 outliers"  "7 outliers" "34 outliers"
```

## 2. Preliminary Modeling

**a) Fit SLR for each predictor variable, assess statistical significance (T-test) and R^2**

```
# Create a function to extract key statistics from model summaries
get_model_stats <- function(model, predictor) {
  summary_stats <- summary(model)

  # Extract relevant statistics
  r_squared <- round(summary_stats$r.squared, 3)
  adj_r_squared <- round(summary_stats$adj.r.squared, 3)
  f_stat <- round(summary_stats$fstatistic[1], 2)
  p_value <- format.pval(pf(summary_stats$fstatistic[1],
                            summary_stats$fstatistic[2],
                            summary_stats$fstatistic[3],
                            lower.tail = FALSE), digits = 3)

  # Return as named vector
  return(c(Predictor = predictor,
           R_squared = r_squared,
           Adj_R_squared = adj_r_squared,
           F_statistic = f_stat,
           P_value = p_value))
}

# Fit all models and collect statistics
modelPts <- lm(Salary ~ PTS, data=train)
modelAst <- lm(Salary ~ AST, data=train)
modelBlk <- lm(Salary ~ BLK, data=train)
modelTrb <- lm(Salary ~ TRB, data=train)
modelefg <- lm(Salary ~ eFG., data=train)
modelStl <- lm(Salary ~ STL, data=train)
modelPos1 <- lm(Salary ~ Pos1, data=train)
modelPlay <- lm(Salary ~ Play, data=train)

models_list <- list(modelPts, modelAst, modelBlk, modelTrb, modelefg, modelStl, modelPos1, modelPlay)

predictors <- c("PTS", "AST", "BLK", "TRB", "eFG.", "STL", "Pos1", "Play")

# Create table
results_matrix <- t(sapply(seq_along(models_list), function(i) {
  get_model_stats(models_list[[i]], predictors[i])
}))

# Convert to data frame and display
results_df <- as.data.frame(results_matrix)
results_df
```

```
##   Predictor R_squared Adj_R_squared F_statistic.value  P_value
## 1       PTS      0.41         0.409            464.55   <2e-16
```

| Predictor | R_squared | Adj_R_squared | F_statistic.value | P_value |
|---|---|---|---|---|
| PTS | 0.41 | 0.409 | 464.55 | <2e-16 |
| AST | 0.255 | 0.254 | 229.39 | <2e-16 |
| BLK | 0.097 | 0.096 | 71.87 | <2e-16 |
| TRB | 0.264 | 0.263 | 240.53 | <2e-16 |
| eFG. | 0.02 | 0.019 | 13.85 | 0.000214 |
| STL | 0.224 | 0.223 | 193.54 | <2e-16 |
| Pos1 | 0.023 | 0.017 | 3.93 | 0.00368 |
| Play | 0.203 | 0.202 | 170.49 | <2e-16 |

```
## 2       AST     0.255         0.254           229.39   <2e-16
## 3       BLK     0.097         0.096            71.87   <2e-16
## 4       TRB     0.264         0.263           240.53   <2e-16
## 5      eFG.      0.02         0.019            13.85 0.000214
## 6       STL     0.224         0.223           193.54   <2e-16
## 7      Pos1     0.023         0.017             3.93  0.00368
## 8      Play     0.203         0.202           170.49   <2e-16
```

```
if (requireNamespace("kableExtra", quietly = TRUE)) {
  library(kableExtra)
  kable(results_df, format = "latex") %>%
    kable_styling(bootstrap_options = c("striped", "hover", "condensed"),
                  full_width = FALSE)
}
```

**b) Fit MLR for all predictor variables to assess ANOVA and T-test**

```
full_model <- lm(Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL + Pos1 +  Play, data=train)
summary(full_model)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL +
##     Pos1 + Play, data = train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -16696084  -3007065   -822167   2531663  28591808
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2968364    1474249   2.013 0.044471 *
## PTS           392254      89111   4.402 1.25e-05 ***
## AST           651228     225170   2.892 0.003952 **
## BLK         -1594474     730498  -2.183 0.029409 *
## TRB           337974     166203   2.033 0.042403 *
## eFG.        -2952908    2610275  -1.131 0.258356
## MP             63466      63899   0.993 0.320967
## STL           890026     847687   1.050 0.294128
## Pos1PF      -2526025     706851  -3.574 0.000378 ***
## Pos1PG      -3674876     994378  -3.696 0.000238 ***
## Pos1SF      -1795629     860733  -2.086 0.037348 *
## Pos1SG      -3210440     868696  -3.696 0.000238 ***
## PlayYes      3988951    1207343   3.304 0.001005 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5297000 on 658 degrees of freedom
## Multiple R-squared:  0.4687, Adjusted R-squared:  0.459
```

```
## F-statistic: 48.36 on 12 and 658 DF,  p-value: < 2.2e-16
```

Examine VIF for multicollinearity

```
library(car)
```

```
## Loading required package: carData
```

```
vif(full_model)
```

```
##           GVIF Df GVIF^(1/(2*Df))
## PTS   7.126053  1        2.669467
## AST   3.862389  1        1.965296
## BLK   2.331042  1        1.526775
## TRB   4.342256  1        2.083808
## eFG. 1.181838  1        1.087124
## MP    8.030341  1        2.833786
## STL   2.921076  1        1.709116
## Pos1 3.861793  4        1.183992
## Play 1.816036  1        1.347604
```

## 3. Model Refinement

### a) Check assumptions for using residual plots in MLR

```
y_hat <- fitted(full_model)

plot(y_hat, full_model$Salary,
    xlab = "Fitted",
    ylab = "Salary",
    main = "Salary vs Fitted")
```


#### i. Scatterplot of Response vs Fitted values

#### ii. Pairwise scatterplots of predictors

```
install.packages("GGally")
```

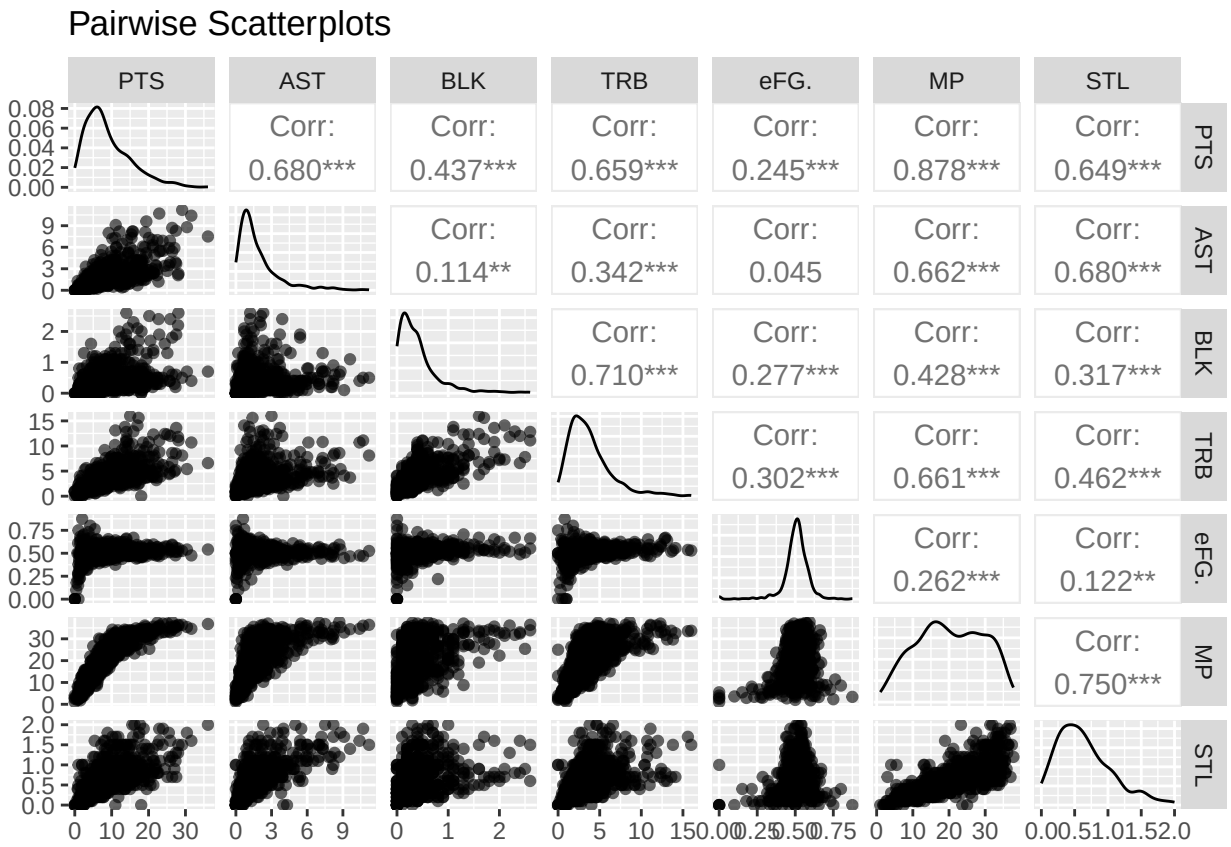
```
## Installing package into '/usr/local/lib/R/site-library'
## (as 'lib' is unspecified)
```

```
library(GGally)
```

```
## Loading required package: ggplot2
```

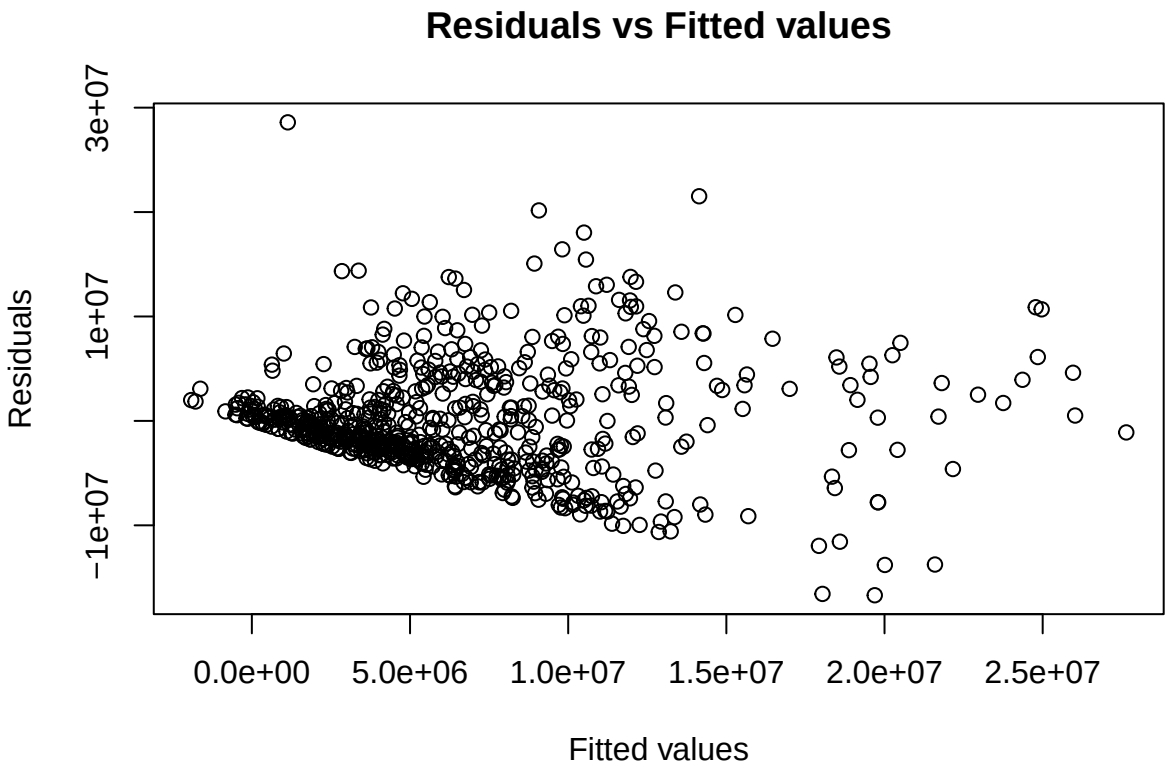
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
ggpairs(train[, c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL")],
        title = "Pairwise Scatterplots",
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6, size = 1.5)),
        progress = FALSE)
```



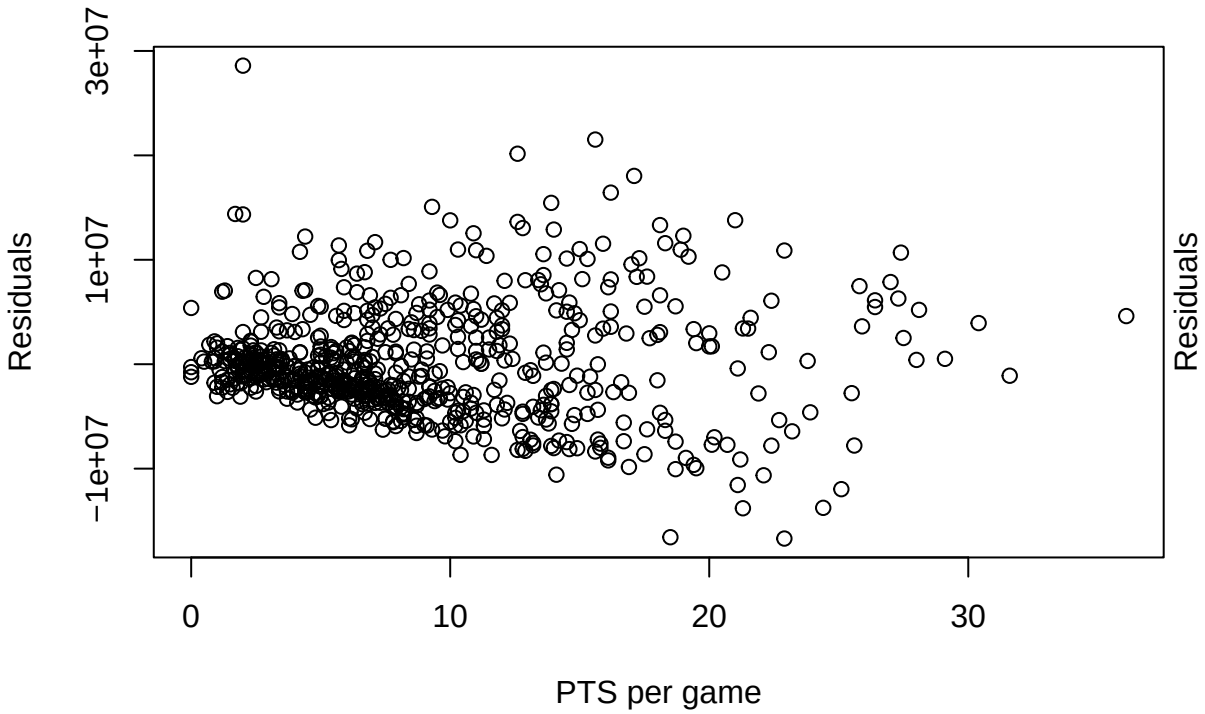

### b) Check all assumptions (linearity, uncorrelated errors, constant variance, normality) #### i. Residuals vs fitted values

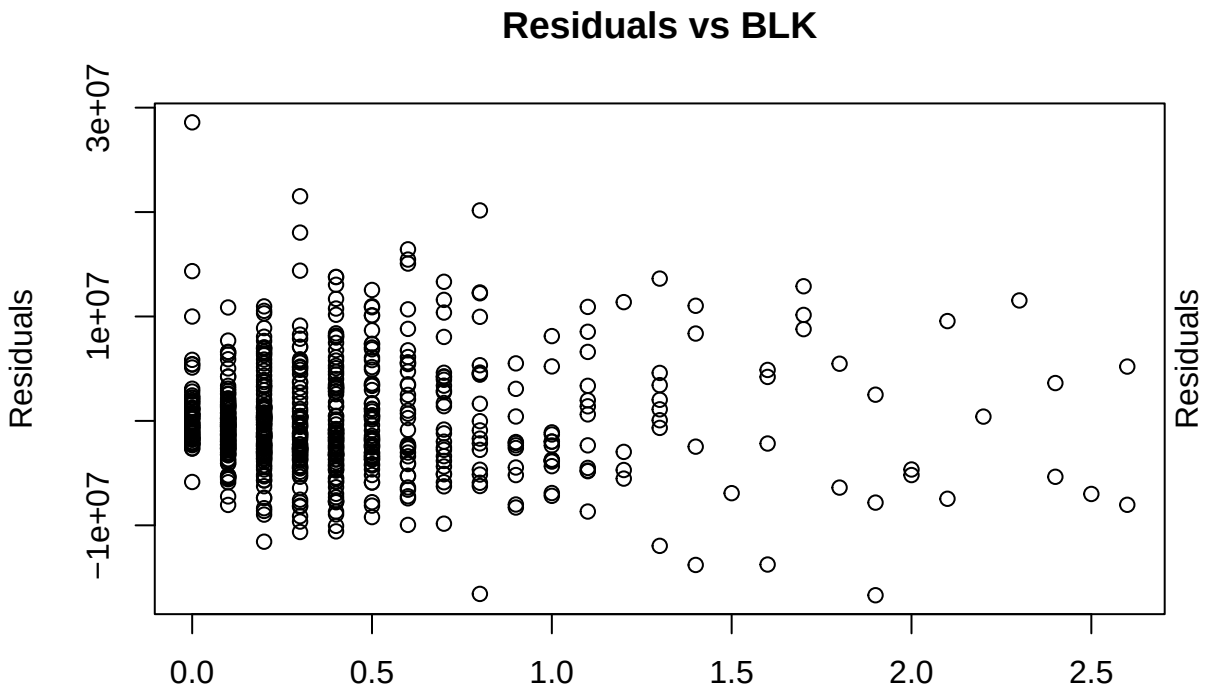
```
y_value <- resid(full_model)
x_value <- fitted(full_model)
plot(x_value, y_value, xlab = "Fitted values", ylab = "Residuals", main = "Residuals vs Fitted values")
```

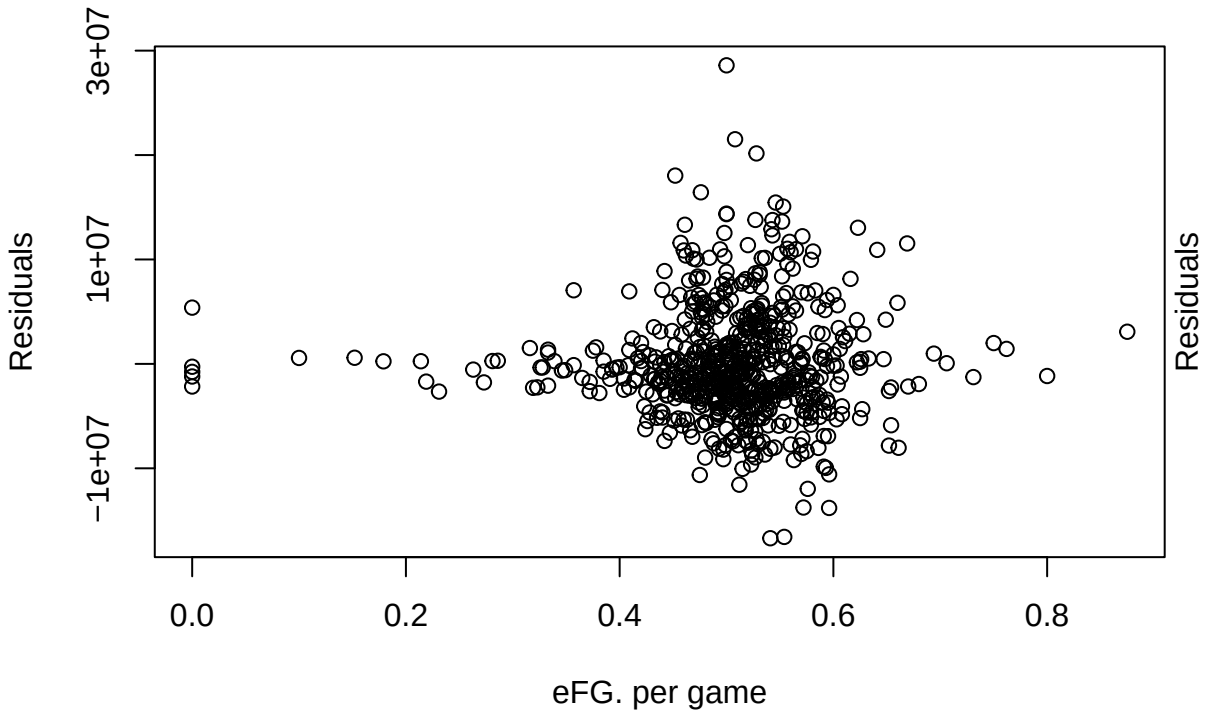


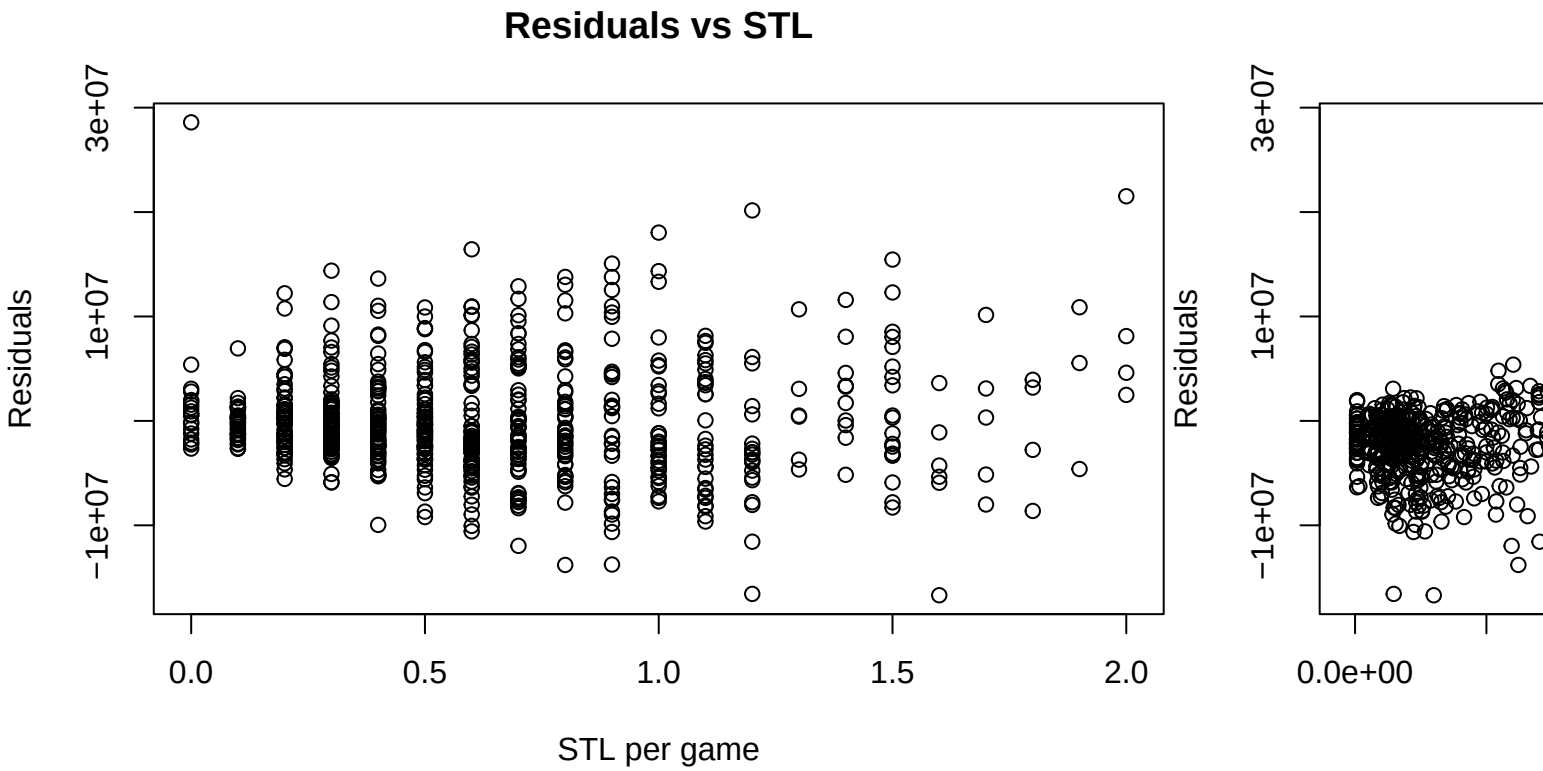

#### ii. Residuals vs each numerical predictors

```
for (var in numeric_vars) {
  plot(train[[var]], y_value,
       xlab = paste(var, "per game"),
       ylab = "Residuals",
       main = paste("Residuals vs", var)
  )
}
```



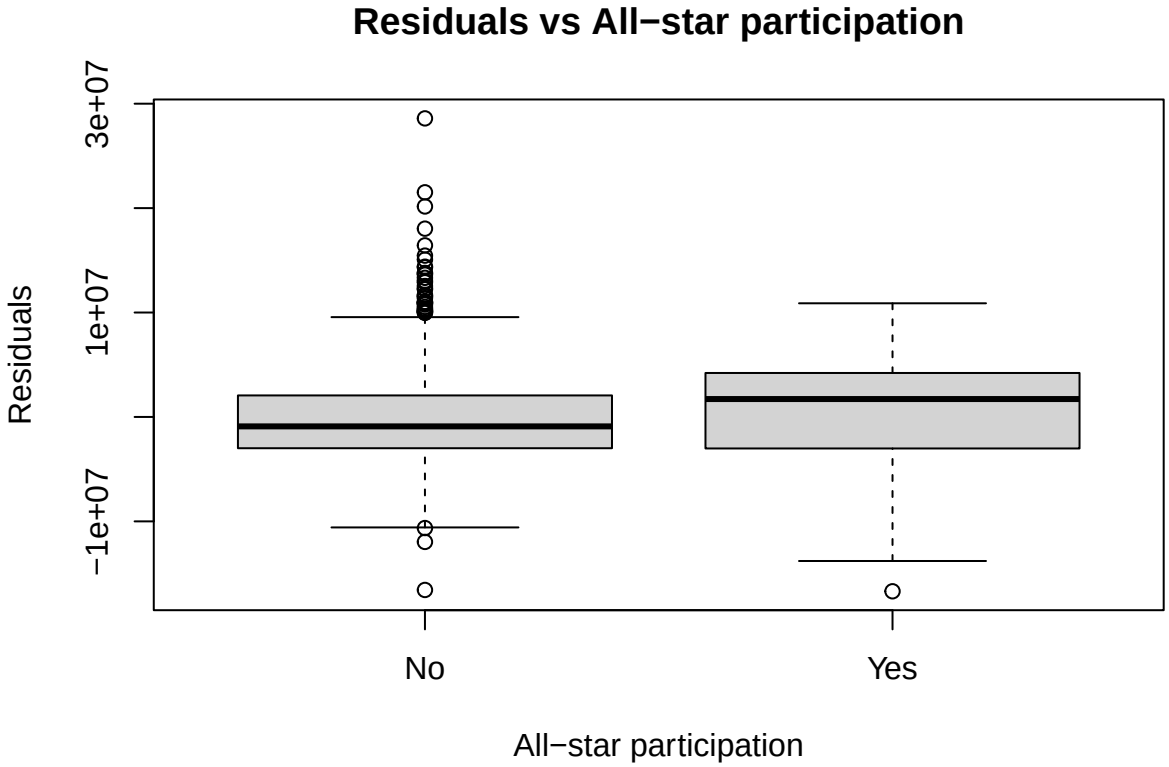

Residuals vs BLK
Residuals
3e+07
1e+07
−1e+07
0.0
0.5
1.0
1.5
2.0
2.5
BLK per game


Residuals vs eFG.
Residuals
3e+07
1e+07
−1e+07
0.0
0.2
0.4
0.6
0.8
eFG. per game

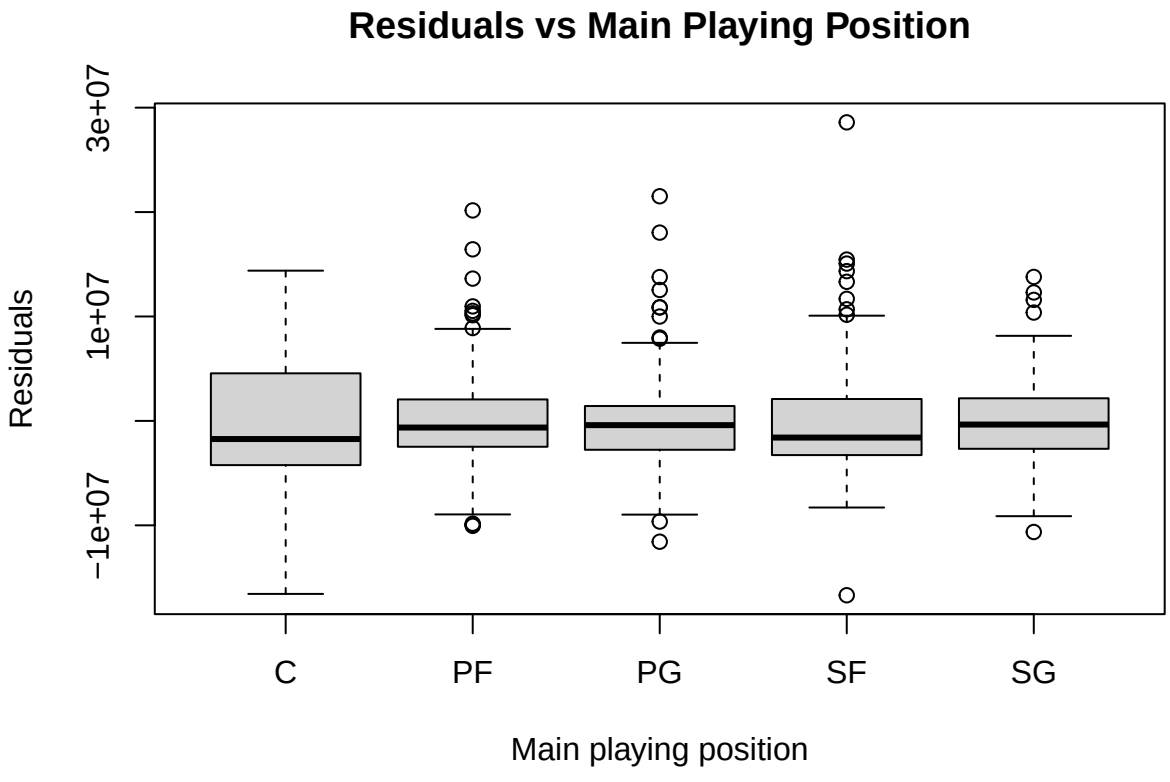



#### iii. Residuals vs each categorical predictors

```
boxplot(y_value ~ train$Play, xlab = "All-star participation", ylab = "Residuals", main = "Residuals vs All-st
```

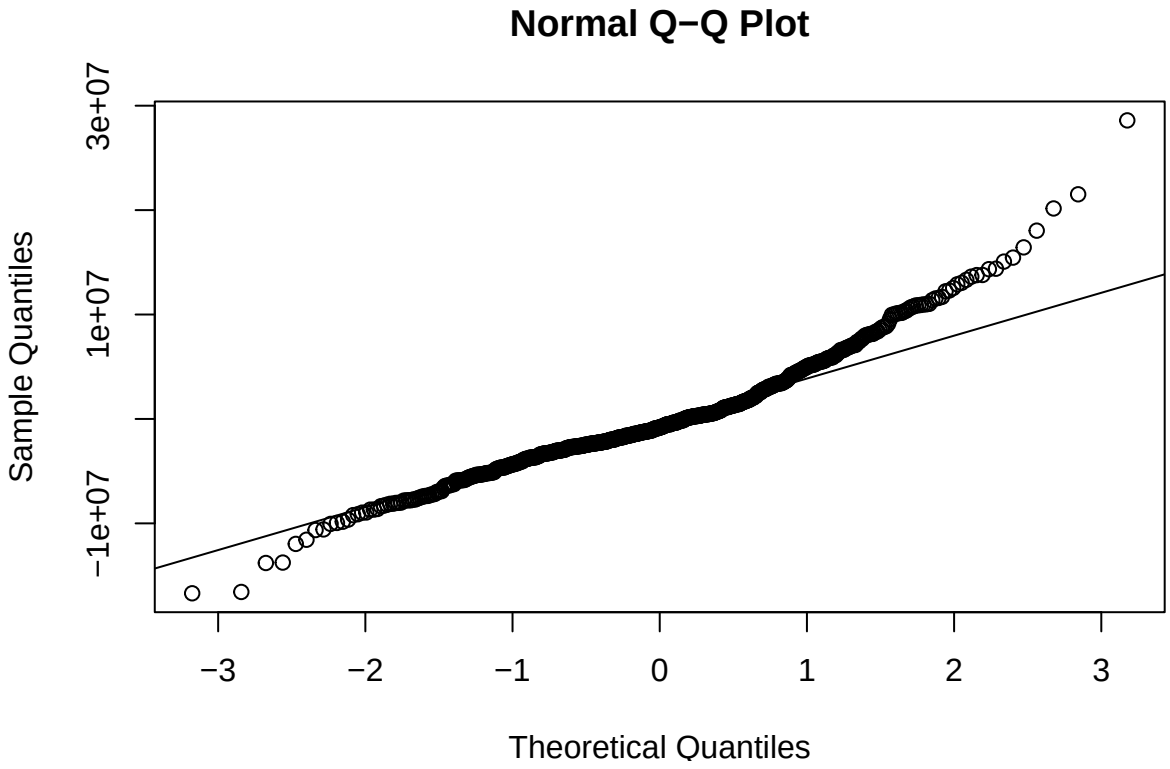



```
boxplot(y_value ~ train$Pos1, xlab = "Main playing position", ylab = "Residuals", main = "Residuals vs Main Pl
```




#### iv. QQ plot

```
qqnorm(y_value)
qqline(y_value)
```




**c) Apply box cox to any predictor or response variables if necessary**

```
# Shift everything by 0.5 since we have some zero values (box-cox doesn't allow)
train_shifted <- train
for (var in numeric_vars) {
  min_value <- min(train[[var]], na.rm = TRUE)
  if (min_value <= 0) {
    train_shifted[[var]] <- train[[var]] - min_value + 1
```

```
  }
}
```

Use powerTransform to find lambda values

```
p <- powerTransform(train_shifted[, numeric_vars])
summary(p)
```

```
## bcPower Transformations to Multinormality
##        Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## PTS       0.0398        0.00      -0.0217       0.1014
## AST      -0.6044       -0.50      -0.7109      -0.4980
## BLK      -1.8948       -2.00      -2.1577      -1.6319
## TRB      -0.0511        0.00      -0.1377       0.0355
## eFG.      3.6037        3.60       3.0068       4.2007
## MP        0.7091        0.71       0.6280       0.7902
## STL      -0.8094       -1.00      -1.0400      -0.5787
## Salary    0.1999        0.20       0.1559       0.2439
##
## Likelihood ratio test that transformation parameters are equal to 0
##  (all log transformations)
##                                           LRT df       pval
## LR test, lambda = (0 0 0 0 0 0 0 0) 1165.094  8 < 2.22e-16
##
## Likelihood ratio test that no transformations are needed
##                                           LRT df       pval
## LR test, lambda = (1 1 1 1 1 1 1 1) 3278.404  8 < 2.22e-16
```

Apply box-cox transformation using the lambda values

```
train_transformed <- train_shifted

train_transformed$PTS <- log(train_shifted$PTS)
train_transformed$AST <- train_shifted$AST^(-0.5)
train_transformed$BLK <- train_shifted$BLK^(-2)
train_transformed$TRB <- log(train_shifted$TRB)
train_transformed$eFG. <- train_shifted$eFG^(4)
train_transformed$MP <- train_shifted$MP^(0.75)
train_transformed$STL <- train_shifted$STL^(-1)
train_transformed$Salary <- train_shifted$Salary^(0.25)

full_model2 <- lm(Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL + Pos1 +  Play, train_transformed)
summary(full_model2)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL +
##     Pos1 + Play, data = train_transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -32.830  -6.983   0.072   7.367  45.028
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  34.9018     7.8008   4.474 9.04e-06 ***
## PTS           3.5836     1.8584   1.928 0.054246 .
## AST          -9.0887     6.0196  -1.510 0.131561
## BLK           2.3408     2.7726   0.844 0.398829
## TRB          -2.2579     1.9046  -1.185 0.236250
## eFG.         -0.6593     0.4259  -1.548 0.122080
## MP            2.0592     0.4327   4.759 2.39e-06 ***
```

```
## STL           1.0822     4.7887   0.226 0.821275
## Pos1PF       -6.6130     1.3543  -4.883 1.31e-06 ***
## Pos1PG       -9.6214     2.1607  -4.453 9.95e-06 ***
## Pos1SF       -6.3439     1.6973  -3.738 0.000202 ***
## Pos1SG      -10.2760     1.8221  -5.640 2.53e-08 ***
## PlayYes       7.6231     1.9323   3.945 8.84e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.3 on 658 degrees of freedom
## Multiple R-squared:  0.4708, Adjusted R-squared:  0.4612
## F-statistic: 48.78 on 12 and 658 DF,  p-value: < 2.2e-16
```

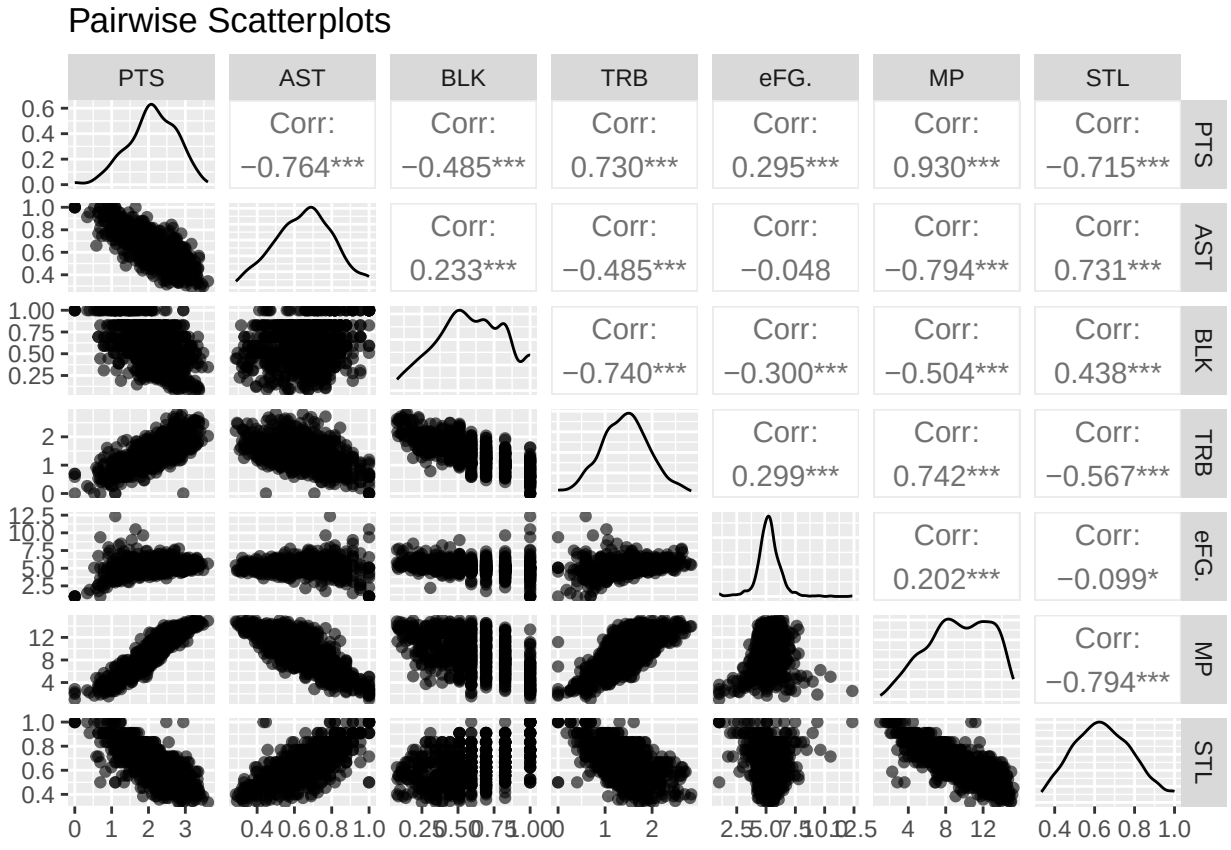
**d) Check assumptions for using residual plots in MLR again**

```
y_hat <- fitted(full_model2)

plot(y_hat, full_model2$Salary,
    xlab = "Fitted",
    ylab = "Salary",
    main = "Salary vs Fitted")
```


**i. Scatterplot of Response vs Fitted values**

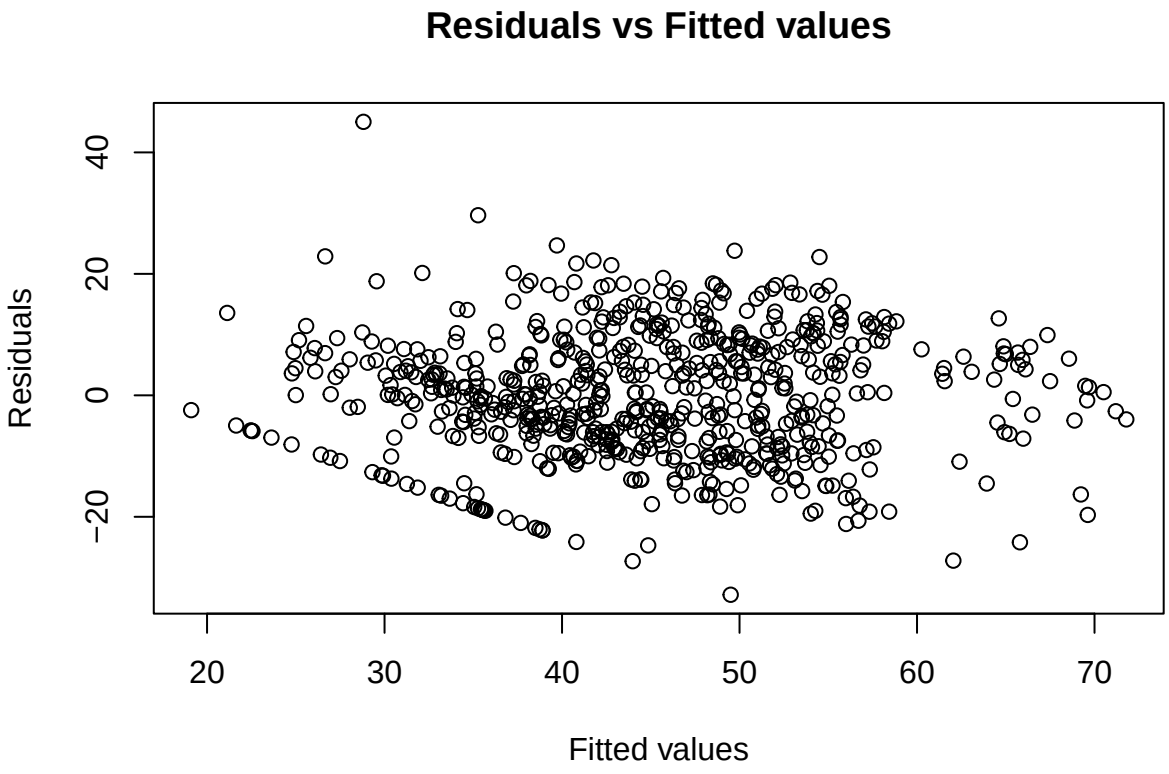
#### ii. Pairwise scatterplots of predictors

```
ggpairs(train_transformed[, c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL")],
        title = "Pairwise Scatterplots",
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6, size = 1.5)),
        progress = FALSE)
```




### e) Check all assumptions (linearity, uncorrelated errors, constant variance, normality) again #### i. Residuals vs fitted values

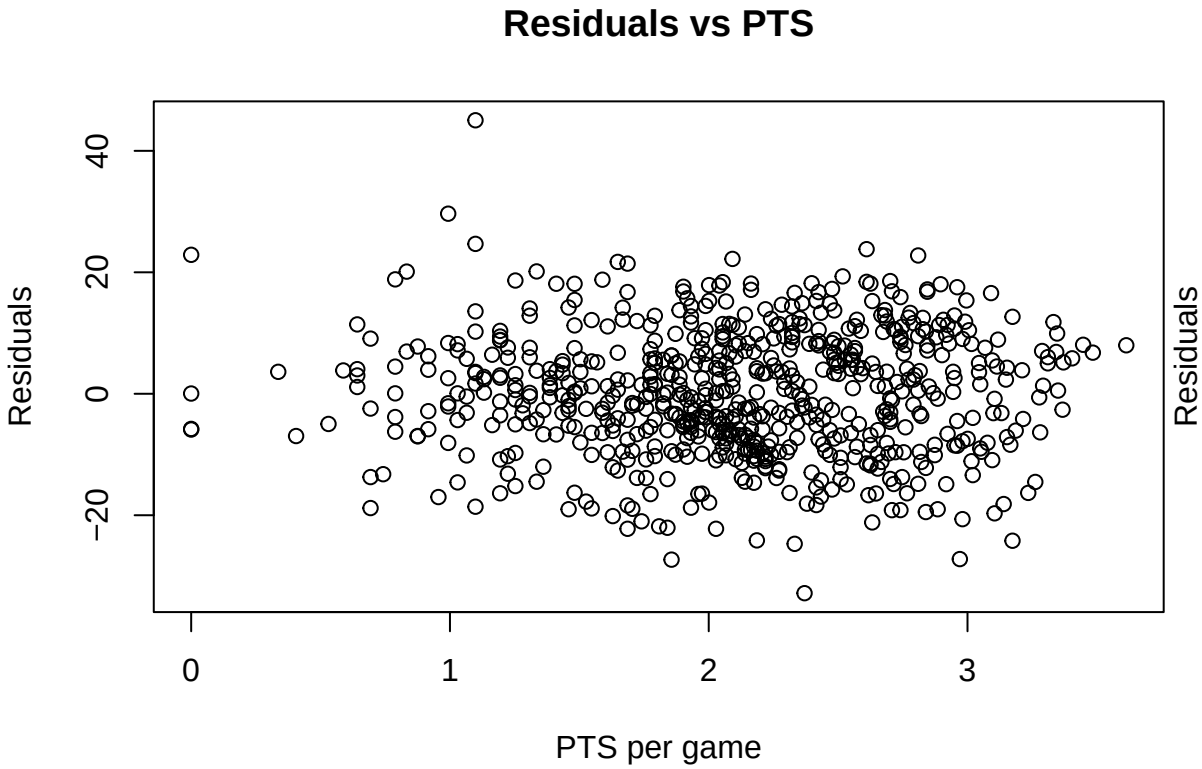
```
y_value <- resid(full_model2)
x_value <- fitted(full_model2)
plot(x_value, y_value, xlab = "Fitted values", ylab = "Residuals", main = "Residuals vs Fitted values")
```

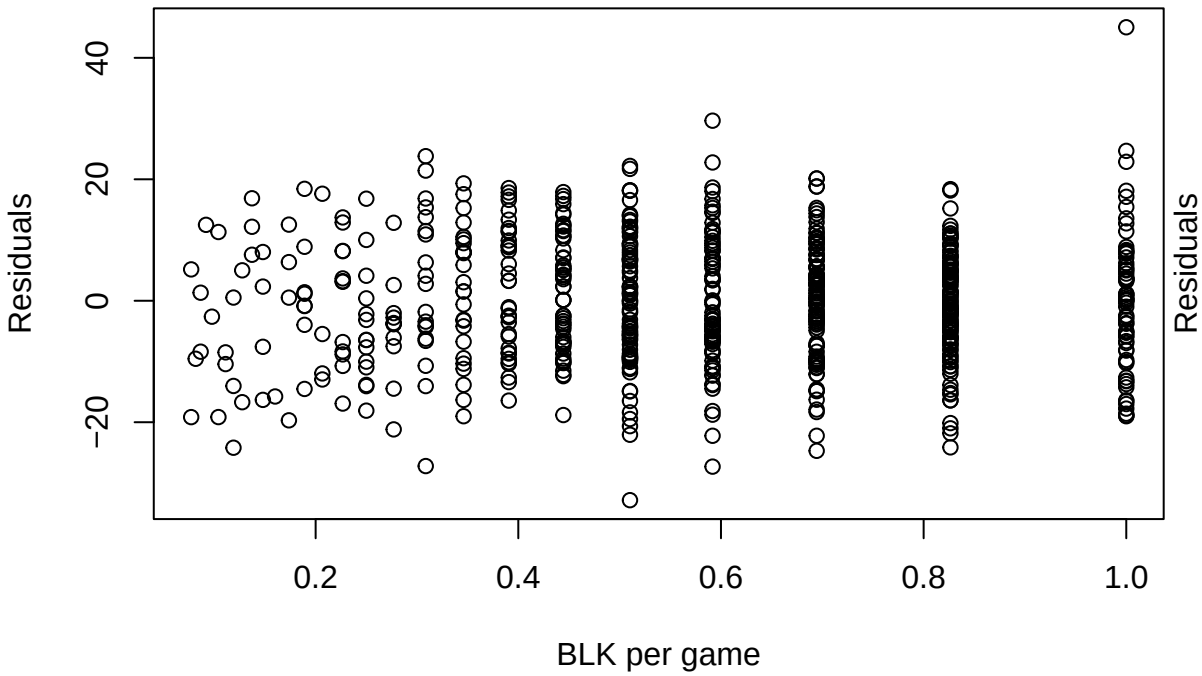


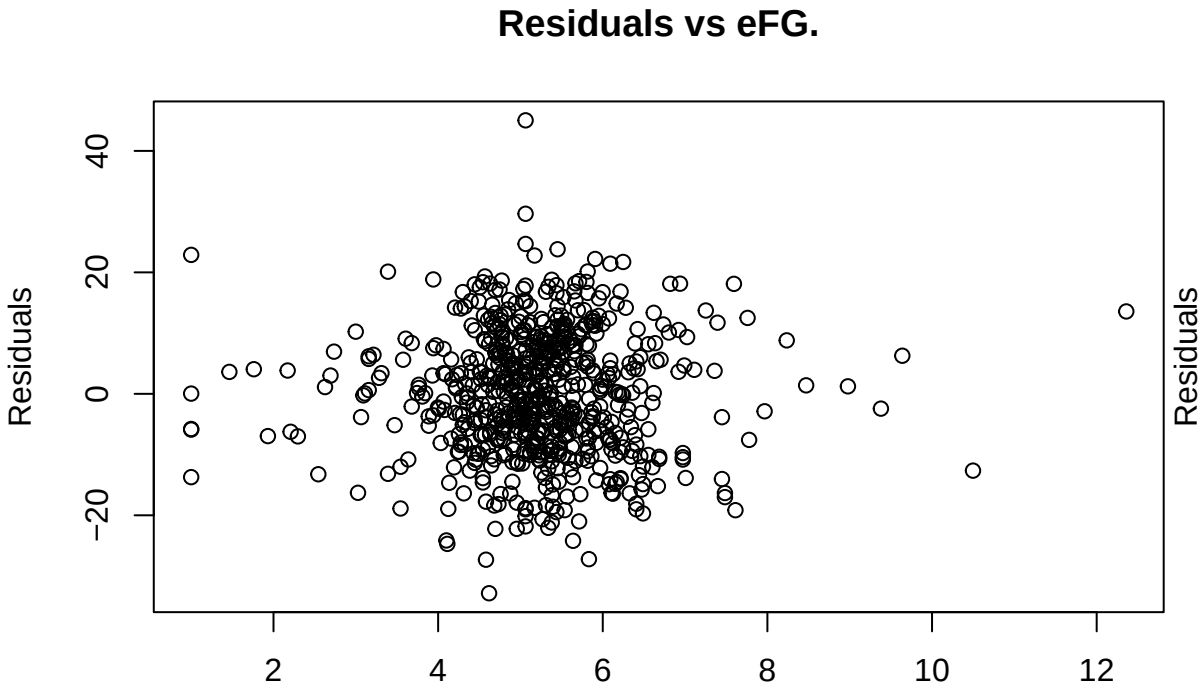

#### ii. Residuals vs each numerical predictors

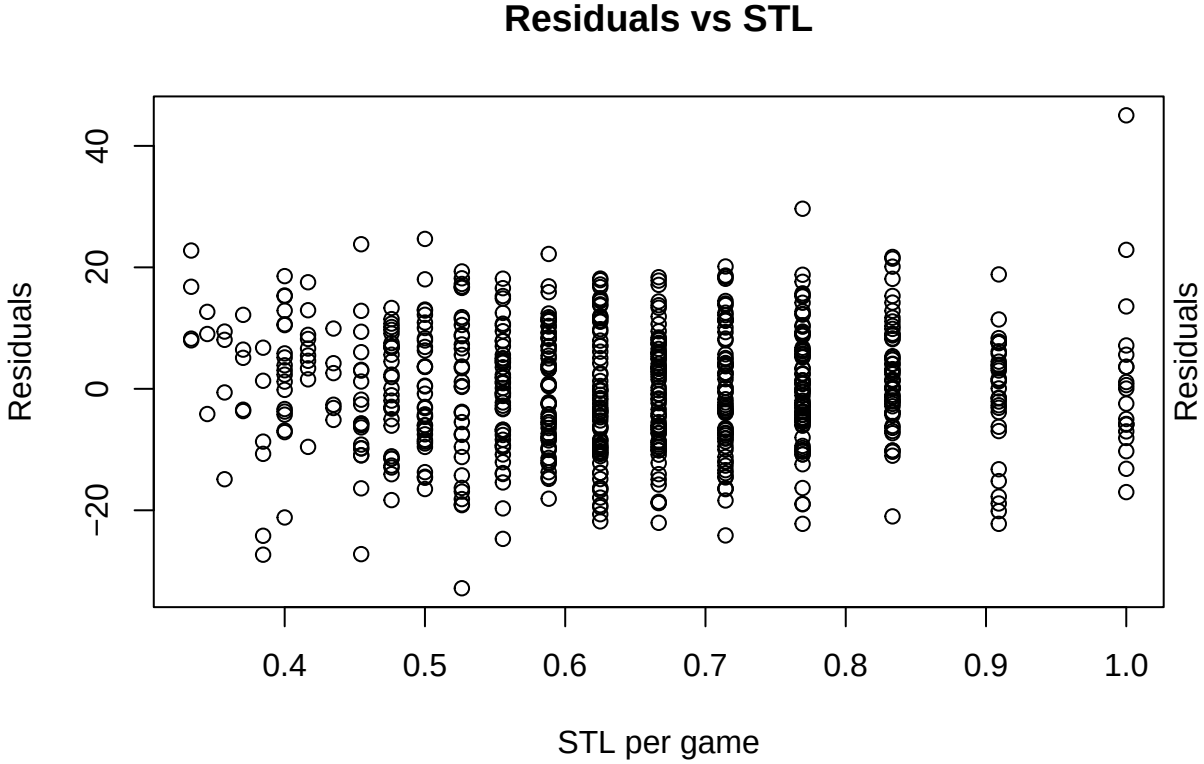
```
for (var in numeric_vars) {
  plot(train_transformed[[var]], y_value,
       xlab = paste(var, "per game"),
```

```
        ylab = "Residuals",
        main = paste("Residuals vs", var)
    )
}
```



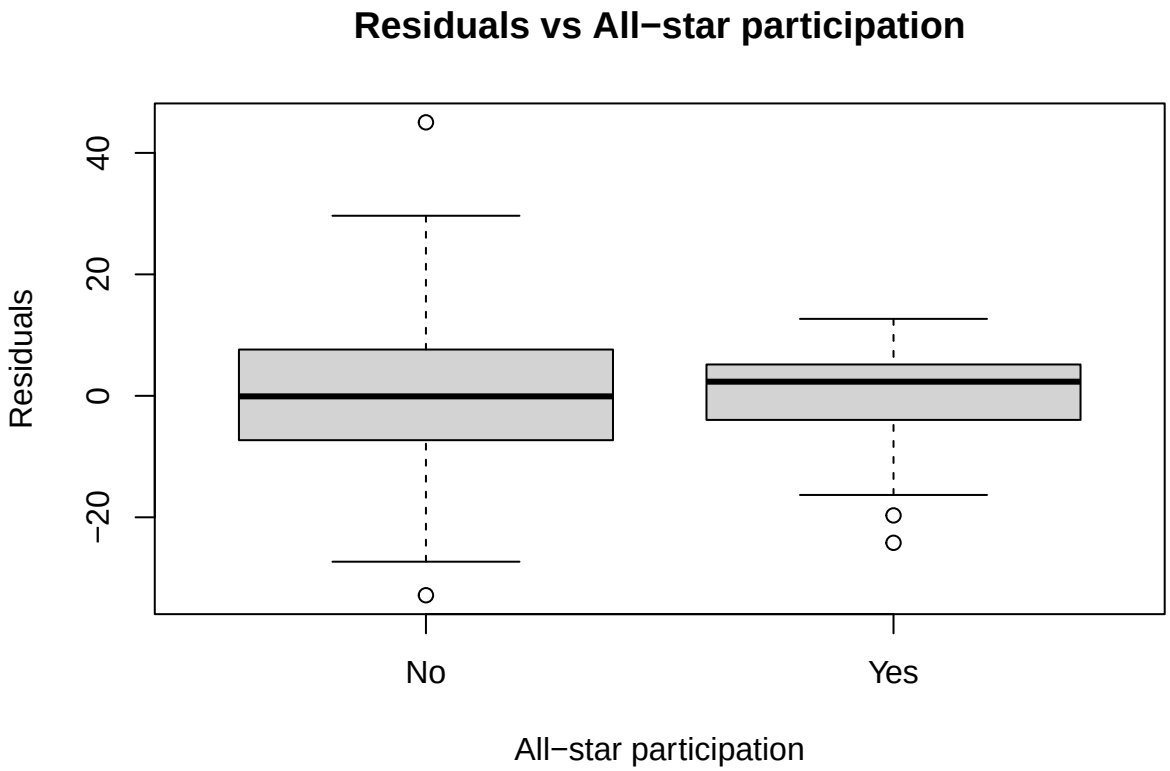





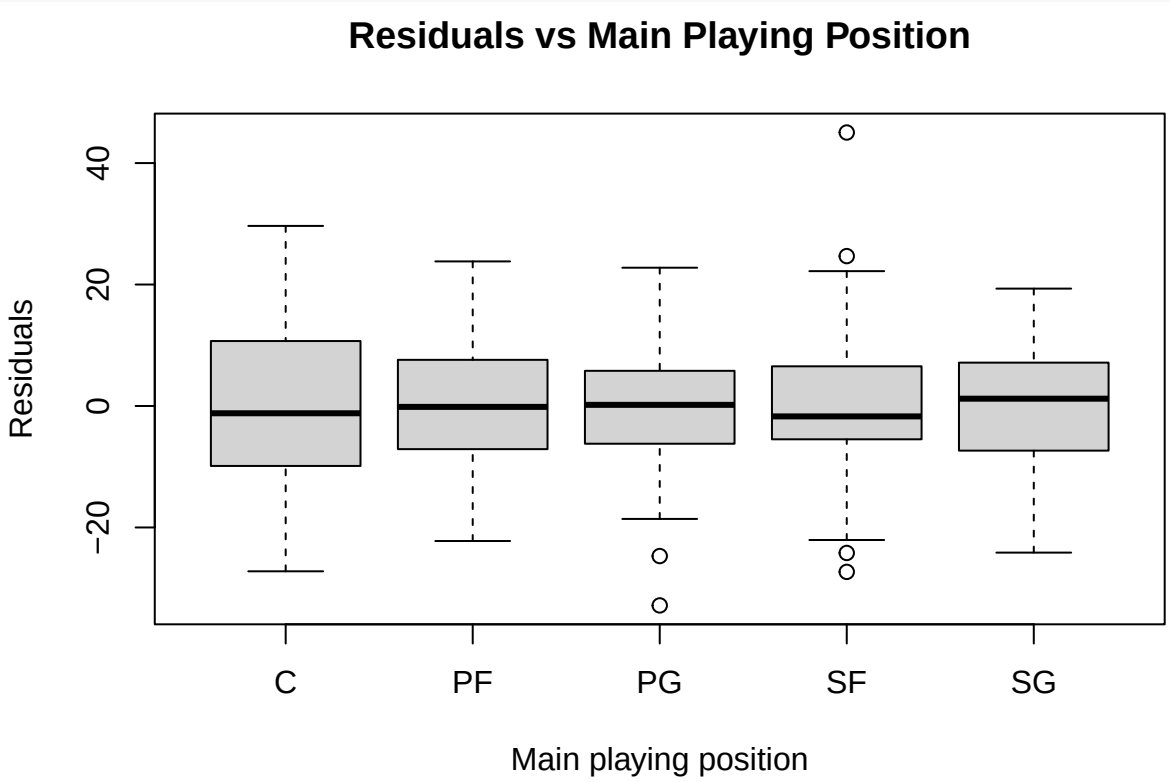




#### iii. Residuals vs each categorical predictors

```
boxplot(y_value ~ train_transformed$Play, xlab = "All-star participation", ylab = "Residuals", main = "Residua
```

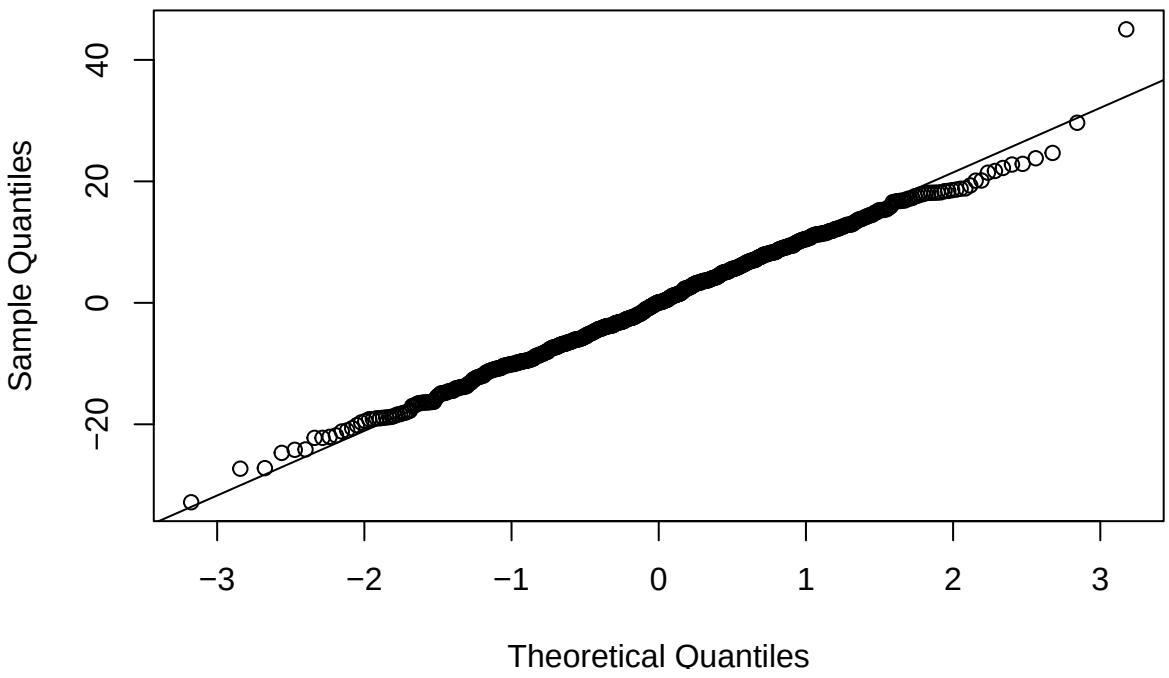



```
boxplot(y_value ~ train_transformed$Pos1, xlab = "Main playing position", ylab = "Residuals", main = "Residual
```




#### iv. QQ plot

```
qqnorm(y_value)
qqline(y_value)
```




## 4. Model Selection

### a) Forward selection

```
library(MASS)

stepAIC(
  lm(Salary ~ 1, data = train_transformed),
  scope = list(upper = lm(Salary ~ ., data=train_transformed)),
  direction = "forward",
  k = 2
)
```

```
## Start:  AIC=3545.43
## Salary ~ 1
##
##        Df Sum of Sq    RSS    AIC
## + MP    1     53631  78241 3197.1
## + PTS   1     52022  79850 3210.8
## + TRB   1     38864  93009 3313.2
## + AST   1     35688  96184 3335.7
## + STL   1     31664 100209 3363.2
## + BLK   1     19233 112639 3441.7
## + Play  1     16490 115383 3457.8
## + eFG.  1      2652 129220 3533.8
## + Pos1  4      3191 128681 3537.0
## <none>              131872 3545.4
##
## Step:  AIC=3197.14
## Salary ~ MP
##
##        Df Sum of Sq   RSS    AIC
## + Pos1  4    4970.5 73270 3161.1
## + Play  1    2824.8 75416 3174.5
## + TRB   1    1433.9 76807 3186.7
## + PTS   1    1200.0 77041 3188.8
```

```
## + BLK   1     646.4 77594 3193.6
## <none>              78241 3197.1
## + STL   1      97.6 78143 3198.3
## + AST   1      71.1 78170 3198.5
## + eFG.  1      24.2 78217 3198.9
##
## Step:  AIC=3161.1
## Salary ~ MP + Pos1
##
##        Df Sum of Sq   RSS    AIC
## + Play  1   2258.74 71012 3142.1
## + PTS   1    751.67 72519 3156.2
## + AST   1    597.56 72673 3157.6
## <none>              73270 3161.1
## + eFG.  1    189.94 73080 3161.4
## + BLK   1    189.33 73081 3161.4
## + TRB   1     62.25 73208 3162.5
## + STL   1     17.19 73253 3162.9
##
## Step:  AIC=3142.09
## Salary ~ MP + Pos1 + Play
##
##        Df Sum of Sq   RSS    AIC
## + AST   1    356.39 70655 3140.7
## + PTS   1    295.84 70716 3141.3
## + BLK   1    288.86 70723 3141.3
## <none>              71012 3142.1
## + eFG.  1    166.10 70846 3142.5
## + TRB   1    146.92 70865 3142.7
## + STL   1     23.45 70988 3143.9
##
## Step:  AIC=3140.71
## Salary ~ MP + Pos1 + Play + AST
##
##        Df Sum of Sq   RSS    AIC
## + BLK   1    249.33 70406 3140.3
## + PTS   1    226.93 70428 3140.6
## + TRB   1    213.56 70442 3140.7
## <none>              70655 3140.7
## + eFG.  1    127.14 70528 3141.5
## + STL   1     74.79 70580 3142.0
##
## Step:  AIC=3140.34
## Salary ~ MP + Pos1 + Play + AST + BLK
##
##        Df Sum of Sq   RSS    AIC
## + PTS   1    217.85 70188 3140.3
## <none>              70406 3140.3
## + eFG.  1    104.44 70301 3141.3
## + TRB   1    103.97 70302 3141.3
## + STL   1     19.54 70386 3142.2
##
## Step:  AIC=3140.26
## Salary ~ MP + Pos1 + Play + AST + BLK + PTS
##
##        Df Sum of Sq   RSS    AIC
## + eFG.  1   244.722 69943 3139.9
## <none>              70188 3140.3
## + TRB   1   144.658 70043 3140.9
## + STL   1     6.984 70181 3142.2
```

```
##
## Step:  AIC=3139.92
## Salary ~ MP + Pos1 + Play + AST + BLK + PTS + eFG.
##
##        Df Sum of Sq   RSS    AIC
## <none>              69943 3139.9
## + TRB   1   152.628 69791 3140.4
## + STL   1     8.993 69934 3141.8
```

```
##
## Call:
## lm(formula = Salary ~ MP + Pos1 + Play + AST + BLK + PTS + eFG.,
##     data = train_transformed)
##
## Coefficients:
## (Intercept)           MP       Pos1PF       Pos1PG       Pos1SF       Pos1SG
##     32.2748       1.8742      -6.2457      -8.3855      -5.5500      -9.1413
##     PlayYes          AST          BLK          PTS          eFG.
##      7.5698      -7.8335       3.5169       3.3686      -0.6467
```

**b) Backward selection**

```
stepAIC(
  lm(Salary ~ ., data = train_transformed),
  scope = list(upper = lm(Salary ~ 1, data=train_transformed)),
  direction = "backward",
  k = 2
)
```

```
## Start:  AIC=3142.4
## Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL + Pos1 + Play
##
##        Df Sum of Sq   RSS    AIC
## - STL   1       5.4 69791 3140.5
## - BLK   1      75.6 69861 3141.1
## - TRB   1     149.1 69934 3141.8
## <none>              69785 3142.4
## - AST   1     241.8 70027 3142.7
## - eFG.  1     254.2 70039 3142.8
## - PTS   1     394.4 70180 3144.2
## - Play  1    1650.6 71436 3156.1
## - MP    1    2401.9 72187 3163.1
## - Pos1  4    3862.5 73648 3170.5
##
## Step:  AIC=3140.45
## Salary ~ PTS + AST + BLK + TRB + eFG. + MP + Pos1 + Play
##
##        Df Sum of Sq   RSS    AIC
## - BLK   1      91.9 69883 3139.3
## - TRB   1     152.6 69943 3139.9
## <none>              69791 3140.5
## - AST   1     237.6 70028 3140.7
## - eFG.  1     252.7 70043 3140.9
## - PTS   1     410.4 70201 3142.4
## - Play  1    1652.1 71443 3154.2
## - MP    1    2539.3 72330 3162.4
## - Pos1  4    3994.7 73785 3169.8
##
## Step:  AIC=3139.33
## Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play
##
```

```
##        Df Sum of Sq   RSS    AIC
## <none>              69883 3139.3
## - TRB   1     263.1 70146 3139.9
## - AST   1     267.9 70150 3139.9
## - eFG.  1     279.8 70162 3140.0
## - PTS   1     444.0 70327 3141.6
## - Play  1    1605.3 71488 3152.6
## - MP    1    2447.8 72330 3160.4
## - Pos1  4    3926.4 73809 3168.0
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
##     data = train_transformed)
##
## Coefficients:
## (Intercept)          PTS          AST          TRB         eFG.           MP
##     38.4309       3.7660      -9.2958      -2.8416      -0.6893       1.9632
##      Pos1PF       Pos1PG       Pos1SF       Pos1SG      PlayYes
##     -6.4473      -9.3097      -6.1311      -9.9576       7.4984
```

**c) Stepwise selection**

```
stepAIC(
  lm(Salary ~ ., data = train_transformed),
  direction = "both",
  k = 2
)
```

```
## Start:  AIC=3142.4
## Salary ~ PTS + AST + BLK + TRB + eFG. + MP + STL + Pos1 + Play
##
##        Df Sum of Sq   RSS    AIC
## - STL   1       5.4 69791 3140.5
## - BLK   1      75.6 69861 3141.1
## - TRB   1     149.1 69934 3141.8
## <none>              69785 3142.4
## - AST   1     241.8 70027 3142.7
## - eFG.  1     254.2 70039 3142.8
## - PTS   1     394.4 70180 3144.2
## - Play  1    1650.6 71436 3156.1
## - MP    1    2401.9 72187 3163.1
## - Pos1  4    3862.5 73648 3170.5
##
## Step:  AIC=3140.45
## Salary ~ PTS + AST + BLK + TRB + eFG. + MP + Pos1 + Play
##
##        Df Sum of Sq   RSS    AIC
## - BLK   1      91.9 69883 3139.3
## - TRB   1     152.6 69943 3139.9
## <none>              69791 3140.5
## - AST   1     237.6 70028 3140.7
## - eFG.  1     252.7 70043 3140.9
## - PTS   1     410.4 70201 3142.4
## + STL   1       5.4 69785 3142.4
## - Play  1    1652.1 71443 3154.2
## - MP    1    2539.3 72330 3162.4
## - Pos1  4    3994.7 73785 3169.8
##
## Step:  AIC=3139.33
## Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play
```

```
##
##        Df Sum of Sq   RSS    AIC
## <none>              69883 3139.3
## - TRB   1     263.1 70146 3139.9
## - AST   1     267.9 70150 3139.9
## - eFG.  1     279.8 70162 3140.0
## + BLK   1      91.9 69791 3140.5
## + STL   1      21.7 69861 3141.1
## - PTS   1     444.0 70327 3141.6
## - Play  1    1605.3 71488 3152.6
## - MP    1    2447.8 72330 3160.4
## - Pos1  4    3926.4 73809 3168.0
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
##     data = train_transformed)
##
## Coefficients:
## (Intercept)          PTS          AST          TRB         eFG.           MP
##     38.4309       3.7660      -9.2958      -2.8416      -0.6893       1.9632
##      Pos1PF       Pos1PG       Pos1SF       Pos1SG      PlayYes
##     -6.4473      -9.3097      -6.1311      -9.9576       7.4984
```

**d) Compare the R^2, AIC, BIC, and AICc between these three models to pick the best model**

Fit the result from the forward selection

```
forward_model <- lm(formula = Salary ~ MP + Pos1 + Play + AST + BLK + PTS + eFG.,
    data = train_transformed)

p = length(coef(forward_model)) - 1
n = nrow(train_transformed)

cbind(summary(forward_model)$adj.r.squared,
      extractAIC(forward_model, k=2)[2],
      extractAIC(forward_model, k=log(n))[2],
      extractAIC(forward_model, k=2)[2] + (2 * (p + 2) * (p + 3) / (n - p - 1)))
```

```
##          [,1]     [,2]     [,3]     [,4]
## [1,] 0.461577 3139.917 3189.513 3140.389
```

Fit the result from the backward/stepwise selection (produce the same result)

```
backward_model <- lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
    data = train_transformed)
summary(backward_model)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
##     data = train_transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.029  -7.097  -0.058   7.388  45.593
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  38.4309     6.5643   5.855 7.54e-09 ***
## PTS           3.7660     1.8390   2.048 0.040975 *
## AST          -9.2958     5.8441  -1.591 0.112171
## TRB          -2.8416     1.8025  -1.576 0.115410
```

```
## eFG.         -0.6893     0.4241  -1.626 0.104527
## MP            1.9632     0.4083   4.808 1.89e-06 ***
## Pos1PF       -6.4473     1.3322  -4.840 1.62e-06 ***
## Pos1PG       -9.3097     2.0898  -4.455 9.86e-06 ***
## Pos1SF       -6.1311     1.6251  -3.773 0.000176 ***
## Pos1SG       -9.9576     1.7369  -5.733 1.50e-08 ***
## PlayYes       7.4984     1.9258   3.894 0.000109 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.29 on 660 degrees of freedom
## Multiple R-squared:  0.4701, Adjusted R-squared:  0.462
## F-statistic: 58.55 on 10 and 660 DF,  p-value: < 2.2e-16
```

```
cbind(summary(backward_model)$adj.r.squared,
      extractAIC(backward_model, k=2)[2],
      extractAIC(backward_model, k=log(n))[2],
      extractAIC(backward_model, k=2)[2] + (2 * (p + 2) * (p + 3) / (n - p - 1)))
```

```
##           [,1]     [,2]    [,3]     [,4]
## [1,] 0.4620447 3139.334 3188.93 3139.806
```

Final model: backward selection (slightly larger adjusted R-squared but lower AIC, BIC and AICc)

```
final_model <- lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
    data = train_transformed)
summary(final_model)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + TRB + eFG. + MP + Pos1 + Play,
##     data = train_transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.029  -7.097  -0.058   7.388  45.593
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  38.4309     6.5643   5.855 7.54e-09 ***
## PTS           3.7660     1.8390   2.048 0.040975 *
## AST          -9.2958     5.8441  -1.591 0.112171
## TRB          -2.8416     1.8025  -1.576 0.115410
## eFG.         -0.6893     0.4241  -1.626 0.104527
## MP            1.9632     0.4083   4.808 1.89e-06 ***
## Pos1PF       -6.4473     1.3322  -4.840 1.62e-06 ***
## Pos1PG       -9.3097     2.0898  -4.455 9.86e-06 ***
## Pos1SF       -6.1311     1.6251  -3.773 0.000176 ***
## Pos1SG       -9.9576     1.7369  -5.733 1.50e-08 ***
## PlayYes       7.4984     1.9258   3.894 0.000109 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.29 on 660 degrees of freedom
## Multiple R-squared:  0.4701, Adjusted R-squared:  0.462
## F-statistic: 58.55 on 10 and 660 DF,  p-value: < 2.2e-16
```

### e) Examine VIF for the final model

```
vif(final_model)
```

```
##           GVIF Df GVIF^(1/(2*Df))
```

```
## PTS   9.088298  1        3.014680
## AST   5.214550  1        2.283539
## TRB   5.404959  1        2.324857
## eFG.  1.276132  1        1.129660
## MP   11.593663  1        3.404947
## Pos1  4.342336  4        1.201477
## Play  1.224495  1        1.106569
```

Remove MP based on VIF (because it has high collinearity)

```
final_model2 <- lm(formula = Salary ~ PTS + AST + TRB + eFG. + Pos1 + Play,
    data = train_transformed)
summary(final_model2)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + TRB + eFG. + Pos1 + Play, data = train_transformed)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.857  -7.207  -0.537   7.093  44.272
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  45.7947     6.4890   7.057 4.30e-12 ***
## PTS           9.5557     1.4131   6.762 2.99e-11 ***
## AST         -18.8443     5.5875  -3.373 0.000788 ***
## TRB           0.6138     1.6805   0.365 0.715069
## eFG.         -0.9082     0.4286  -2.119 0.034463 *
## Pos1PF       -5.2021     1.3284  -3.916 9.94e-05 ***
## Pos1PG       -8.4242     2.1162  -3.981 7.63e-05 ***
## Pos1SF       -3.7084     1.5706  -2.361 0.018507 *
## Pos1SG       -7.6375     1.6962  -4.503 7.93e-06 ***
## PlayYes       6.3008     1.9413   3.246 0.001231 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.46 on 661 degrees of freedom
## Multiple R-squared:  0.4515, Adjusted R-squared:  0.444
## F-statistic: 60.46 on 9 and 661 DF,  p-value: < 2.2e-16
```

Recheck the VIF, since some variable still has high VIF, remove another predictor

```
vif(final_model2)
```

```
##          GVIF Df GVIF^(1/(2*Df))
## PTS  5.191886  1        2.278571
## AST  4.612420  1        2.147655
## TRB  4.545879  1        2.132107
## eFG. 1.261425  1        1.123132
## Pos1 3.701198  4        1.177722
## Play 1.204012  1        1.097275
```

**f) Conduct partial F-test to remove least significant predictor (TRB since it's the only non-statistically significant predictor)**

```
final_model3 <- lm(formula = Salary ~ PTS + AST + eFG. + Pos1 + Play,
    data = train_transformed)
summary(final_model3)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + eFG. + Pos1 + Play, data = train_transformed)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.718  -7.254  -0.522   7.127  44.306
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  46.7853     5.8911   7.942 8.58e-15 ***
## PTS           9.8242     1.2061   8.146 1.88e-15 ***
## AST         -19.4321     5.3472  -3.634 0.000301 ***
## eFG.         -0.9147     0.4280  -2.137 0.032940 *
## Pos1PF       -5.3281     1.2820  -4.156 3.66e-05 ***
## Pos1PG       -8.9180     1.6270  -5.481 6.01e-08 ***
## Pos1SF       -3.9817     1.3801  -2.885 0.004040 **
## Pos1SG       -8.0479     1.2697  -6.339 4.29e-10 ***
## PlayYes       6.2993     1.9400   3.247 0.001225 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.45 on 662 degrees of freedom
## Multiple R-squared:  0.4514, Adjusted R-squared:  0.4448
## F-statistic: 68.09 on 8 and 662 DF,  p-value: < 2.2e-16
```

Check VIF after removing TRB

```
vif(final_model3)
```

```
##          GVIF Df GVIF^(1/(2*Df))
## PTS  3.787271  1        1.946091
## AST  4.229698  1        2.056623
## eFG. 1.259275  1        1.122174
## Pos1 1.805485  4        1.076649
## Play 1.204006  1        1.097272
```

Check Anova after removing TRB

```
qf(0.95, 1, 661)
```

```
## [1] 3.855565
```

```
anova(final_model3, final_model2)
```

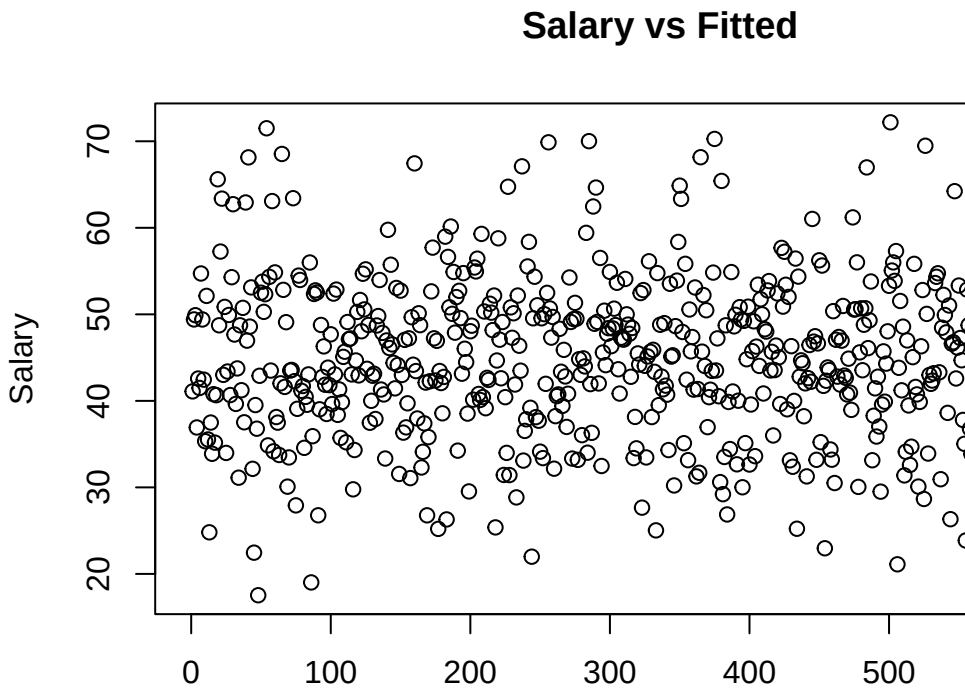
```
## Analysis of Variance Table
##
## Model 1: Salary ~ PTS + AST + eFG. + Pos1 + Play
## Model 2: Salary ~ PTS + AST + TRB + eFG. + Pos1 + Play
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1    662 72345
## 2    661 72330  1    14.595 0.1334 0.7151
```

Since the result is not statistically significant, remove TRB from the model

```
final_model <- final_model3
```

### g) Verify assumptions for using residual plots in MLR again

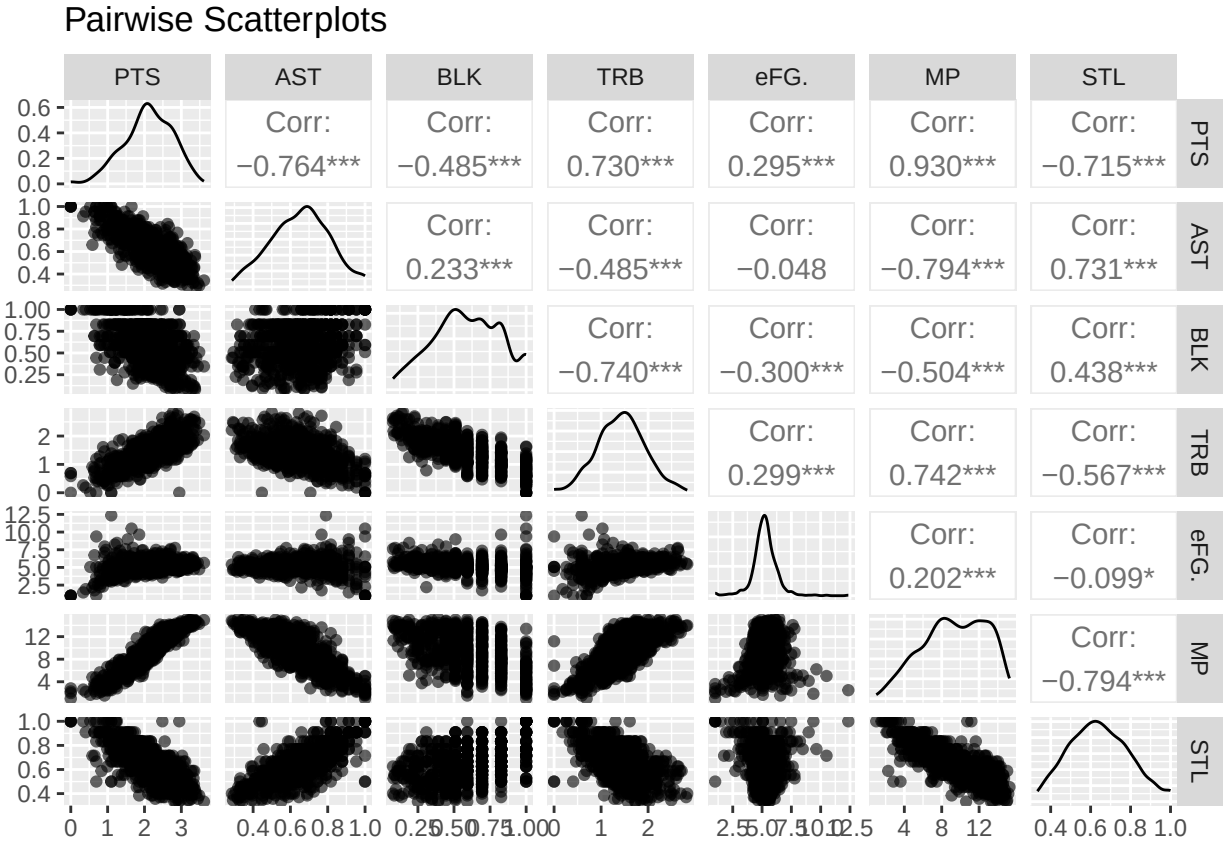
```
y_hat <- fitted(final_model)

plot(y_hat, final_model$Salary,
    xlab = "Fitted",
    ylab = "Salary",
    main = "Salary vs Fitted")
```




**i. Scatterplot of Response vs Fitted values**

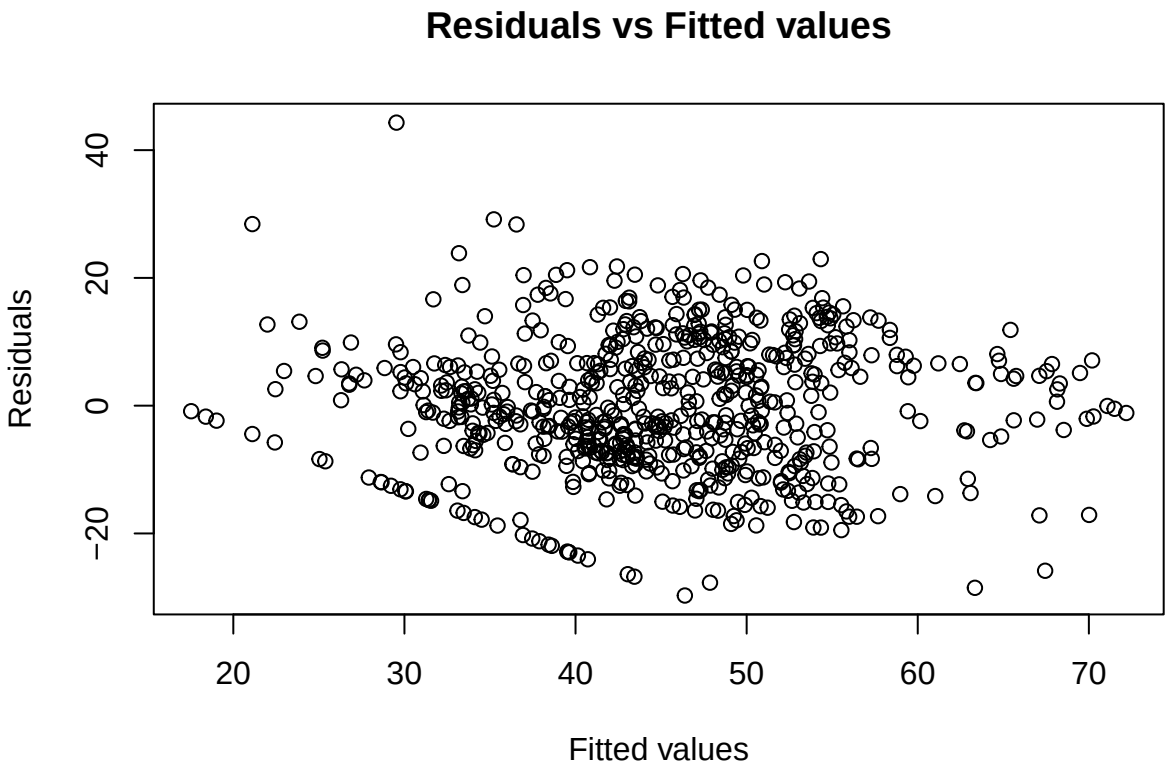
#### ii. Pairwise scatterplots of predictors

```
ggpairs(train_transformed[, c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL")],
        title = "Pairwise Scatterplots",
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6, size = 1.5)),
        progress = FALSE)
```



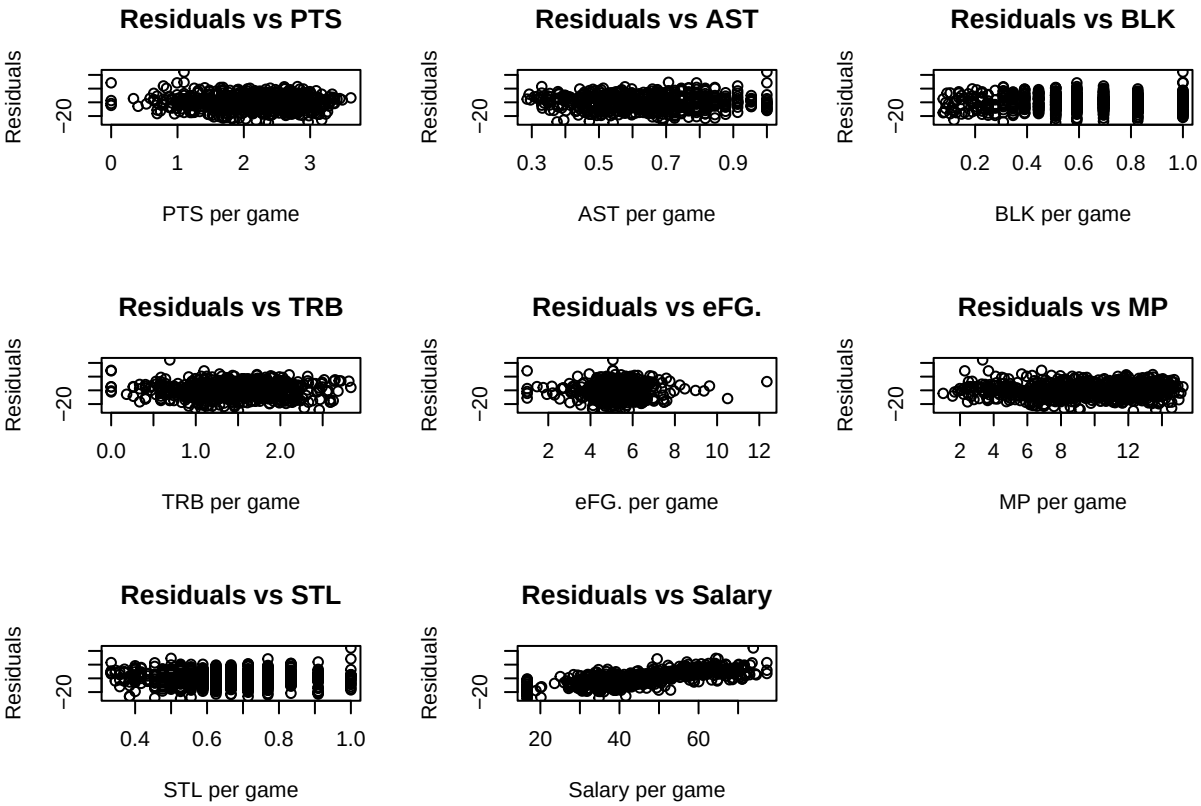

### h) Check all assumptions (linearity, uncorrelated errors, constant variance, normality) again #### i. Residuals vs fitted values

```r
y_value <- resid(final_model)
x_value <- fitted(final_model)
plot(x_value, y_value, xlab = "Fitted values", ylab = "Residuals", main = "Residuals vs Fitted values")
```



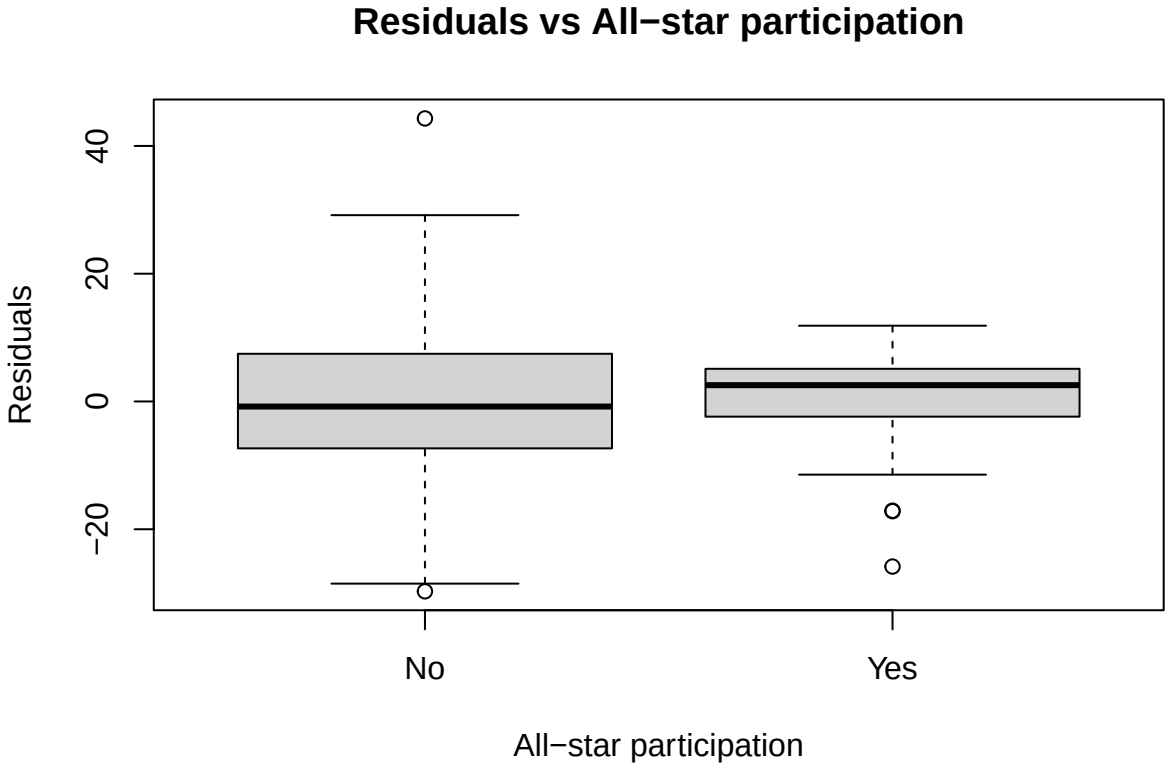

#### ii. Residuals vs each numerical predictors

```r
par(mfrow = c(3, 3))
for (var in numeric_vars) {
  plot(train_transformed[[var]], y_value,
       xlab = paste(var, "per game"),
       ylab = "Residuals",
       main = paste("Residuals vs", var)
  )
}
```

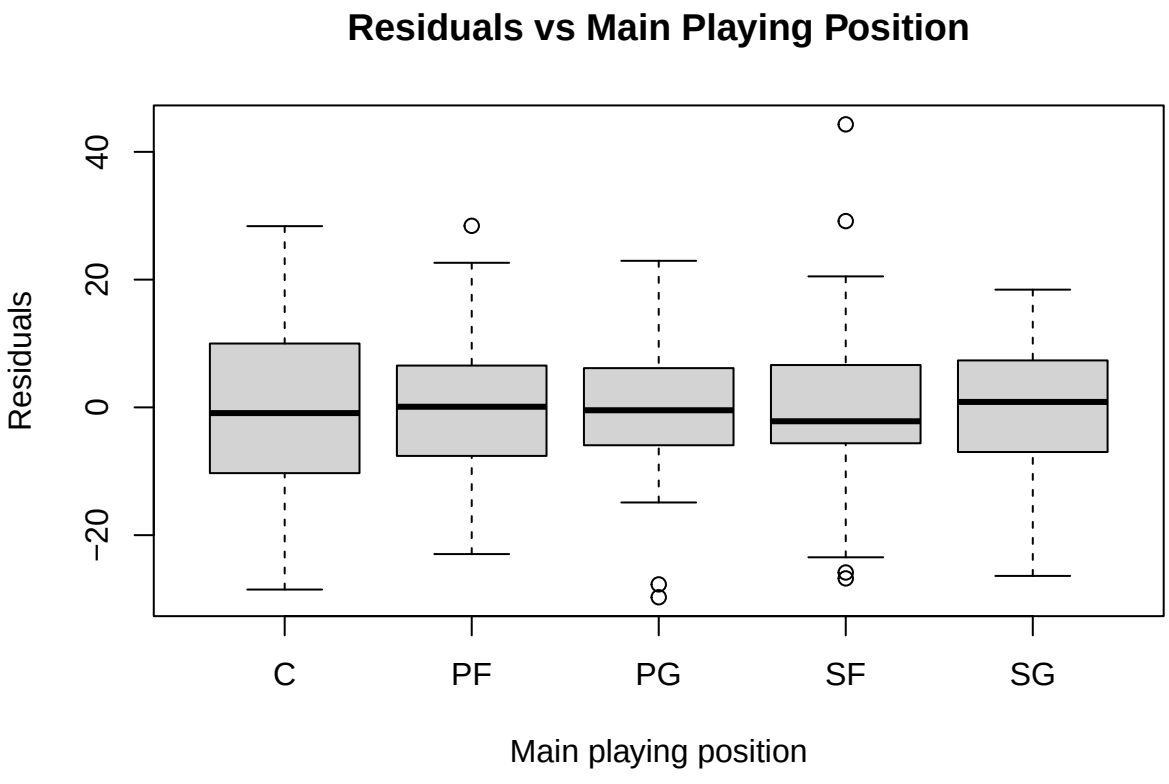



#### iii. Residuals vs each categorical predictors

```
boxplot(y_value ~ train_transformed$Play, xlab = "All-star participation", ylab = "Residuals", main = "Residua
```

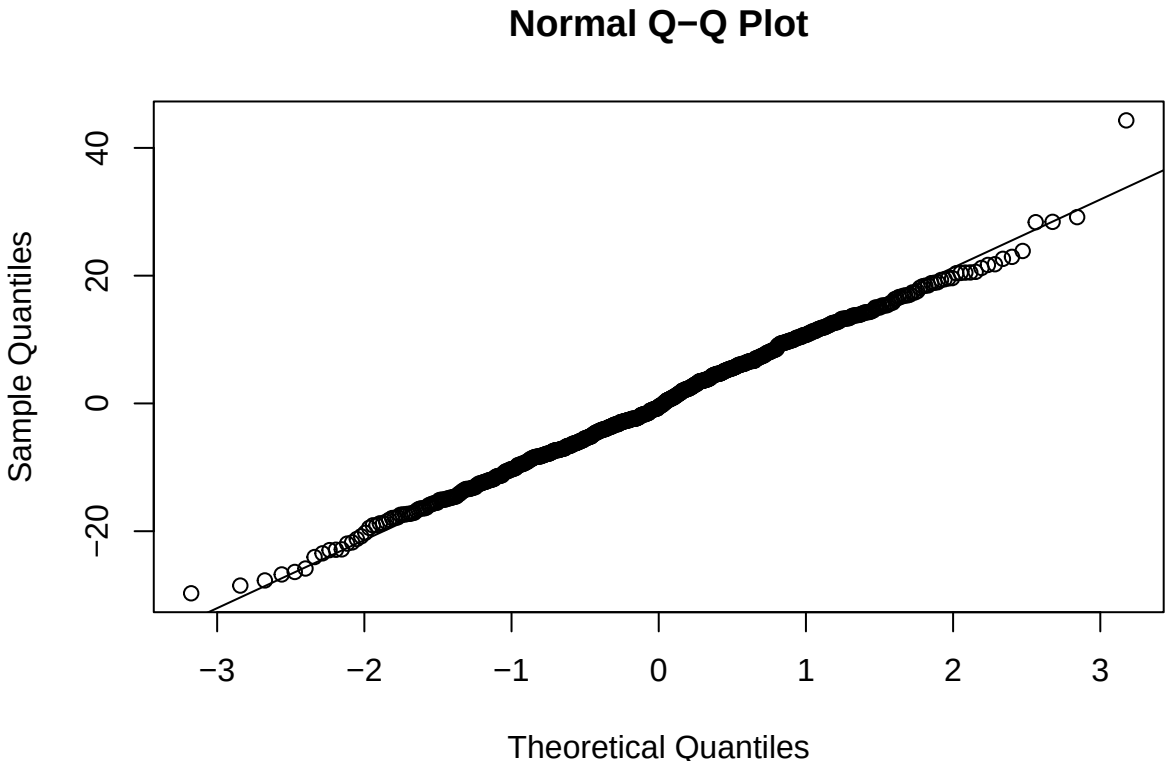



```
boxplot(y_value ~ train_transformed$Pos1, xlab = "Main playing position", ylab = "Residuals", main = "Residual
```




#### iv. QQ plot

```
qqnorm(y_value)
qqline(y_value)
```




**i) Check Problematic Observations**

```
p = length(coef(final_model)) - 1
n = nrow(train_transformed)

hii <- hatvalues(final_model)
cutoff_hii <- 2 * (p + 1) / n
```

```
which(hii > cutoff_hii)
```

### i. Leverage Points

```
##  454  332  934  570  251 1006 1221  801  835  958  196 1220 1213  627 1211 1051
##   19   22   25   29   30   39   41   45   48   54   65   73   75   86  141  158
##  146  719  100  175  103  273  461 1194 1010 1209  714   25 1322  917  295  526
##  160  220  227  237  244  256  280  285  290  350  365  375  380  431  454  480
##  232  457 1164  693  261 1198  115   60  706  359  744  504 1227  440  640 1132
##  484  488  501  506  526  547  559  567  568  571  594  603  608  609  613  621
##  149  235  463 1280  244
##  631  652  656  657  662
```

```
ri <- rstandard(final_model)
which(ri > 2 | ri < - 2)
```

### ii. Outlier points

```
## 1128  997  614  465  782  146  792  602 1377  967  884 1384  529  803  909 1298
##   46   56  101  103  137  160  161  199  202  213  307  335  341  351  419  424
##  810  748 1179  693 1355  665  901 1304
##  448  451  495  506  542  616  628  629
```

```
di <- cooks.distance(final_model)
cutoff_di <- qf(0.5, p + 1, n - p - 1)
which(di > cutoff_di)
```

### iii. Influential on all fitted values

```
## named integer(0)
```

```
dffits <- dffits(final_model)
cutoff_dffits <- 2 * sqrt((p + 1) / (n))
which(abs(dffits) > cutoff_dffits)
```

### iv. Influential on own fitted values

```
## 1128  997 1357  465 1063  782  146  792  602 1377  103 1194 1384  803   17  909
##   46   56   70  103  134  137  160  161  199  202  244  285  335  351  370  419
##  952  810  748  693 1278  706  440  665
##  430  448  451  506  543  568  609  616
```

```
dfbetas <- dfbetas(final_model)
cutoff_dfbetas <- 2 / sqrt(n)

for(i in 1:5){
  print(paste0("Beta ", i-1))
  print(which(abs(dfbetas[,i]) > cutoff_dfbetas))    # this checks all betas in a loop
}
```

### v. Influential on coefficients

```
## [1] "Beta 0"
##  570 1128  598  269 1357 1199 1063  602 1377  772  103  312  856  153 1011 1384
##   29   46   57   68   70   93  134  199  202  238  244  246  258  300  319  335
##  803 1136  965  685   17 1236  534  354  909  508  952  286  810  748  693  497
##  351  360  364  366  370  373  381  385  419  420  430  433  448  451  506  539
## 1278  988  440  665  360 1304  453  302
##  543  551  609  616  622  629  653  663
```

```
## [1] "Beta 1"
##  570 1128  598  269 1357  607  111 1199  465   18 1063  218  146  951  777  602
##   29   46   57   68   70   77   85   93  103  123  134  154  160  173  198  199
##  723  851  772   80  103  532  856  153  781  152 1384  803 1136  685   17  534
##  204  224  238  242  244  252  258  300  305  311  335  351  360  366  370  381
##  354   19  909  952  286  204  810  748  693    9  307 1077  497 1278  988 1381
##  385  389  419  430  433  445  448  451  506  513  518  537  539  543  551  555
##  665  360 1304  453
##  616  622  629  653
## [1] "Beta 2"
##  570 1128  997  598  269 1357  607 1199  465 1063  146  602 1377 1192  762  772
##   29   46   56   57   68   70   77   93  103  134  160  199  202  203  228  238
##  312  856  153 1011 1384  803 1136  965  685   17  354   19  224  909 1052  952
##  246  258  300  319  335  351  360  364  366  370  385  389  414  419  423  430
##  286  748  695 1031    9 1290  497 1278  988  181  440  665  360  453 1246
##  433  451  452  504  513  534  539  543  551  578  609  616  622  653  659
## [1] "Beta 3"
##  787 1357 1213  465  782 1016  777  103  532 1259  781 1384  685   17  306 1070
##   47   70   75  103  137  159  198  244  252  267  305  335  366  370  396  397
##  952  917  752  810 1321  748 1375 1031  693 1036  497  358  456 1381  115  949
##  430  431  441  448  450  451  478  504  506  515  539  550  553  555  559  574
##  440 1304
##  609  629
## [1] "Beta 4"
##  128 1349  231  215  607  614  524  785   18  446 1063  603  475  530 1061 1142
##   14   40   58   63   77  101  109  120  123  128  134  140  142  147  166  172
##  951  697 1262  793  967  772 1217  103  532  856  162  320  604  283 1194  153
##  173  182  194  195  213  238  241  244  252  258  264  266  271  276  285  300
##  884  770 1384  803 1329  347 1097 1136   17  306  898  727 1052 1298  286  204
##  307  331  335  351  354  357  359  360  370  396  398  413  423  424  433  445
##  810  300  693    9  307  304 1290  497  358 1381  706  949  181  479 1165  440
##  448  499  506  513  518  519  534  539  550  555  568  574  578  579  601  609
##  665  360  901  565  302 1121
##  616  622  628  649  663  668
```

## 5. Final Model Validation

### a) Apply model to test dataset

```r
# Box Cox transformation requires all input data to be positive
test_shifted <- test
for (var in numeric_vars) {
  min_value <- min(test[[var]], na.rm = TRUE)
  if (min_value <= 0) {
    test_shifted[[var]] <- test[[var]] - min_value + 1
  }
}
```

Transform test dataset exactly the same way as train dataset (applying the box-cox transformation)

```r
test_transformed <- test_shifted

test_transformed$PTS <- log(test_shifted$PTS)
test_transformed$AST <- test_shifted$AST^(-0.5)
test_transformed$BLK <- test_shifted$BLK^(-2)
test_transformed$TRB <- log(test_shifted$TRB)
test_transformed$eFG. <- test_shifted$eFG^(4)
test_transformed$MP <- test_shifted$MP^(0.75)
test_transformed$STL <- test_shifted$STL^(-1)
test_transformed$Salary <- test_shifted$Salary^(0.25)
```

```
final_model_test <- lm(formula = Salary ~ PTS + AST + eFG. + Pos1 + Play,
    data = test_transformed)
summary(final_model_test)
```

```
##
## Call:
## lm(formula = Salary ~ PTS + AST + eFG. + Pos1 + Play, data = test_transformed)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -30.8860  -7.6996  -0.0067   8.0591  27.9178
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  59.8541     5.6526  10.589  < 2e-16 ***
## PTS           7.9015     1.1337   6.969 7.71e-12 ***
## AST         -34.7525     5.1694  -6.723 3.85e-11 ***
## eFG.         -0.6062     0.2537  -2.389 0.017154 *
## Pos1PF       -2.4083     1.3104  -1.838 0.066524 .
## Pos1PG      -14.6350     1.5818  -9.252  < 2e-16 ***
## Pos1SF       -6.3406     1.4185  -4.470 9.21e-06 ***
## Pos1SG       -7.5369     1.3622  -5.533 4.55e-08 ***
## PlayYes       7.7410     2.0375   3.799 0.000158 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.94 on 662 degrees of freedom
## Multiple R-squared:  0.4825, Adjusted R-squared:  0.4763
## F-statistic: 77.16 on 8 and 662 DF,  p-value: < 2.2e-16
```

**b) Examine VIF of the test model**

```
vif(final_model_test)
```

```
##          GVIF Df GVIF^(1/(2*Df))
## PTS  3.179179  1        1.783025
## AST  3.839898  1        1.959566
## eFG. 1.092979  1        1.045456
## Pos1 1.677936  4        1.066834
## Play 1.182264  1        1.087320
```

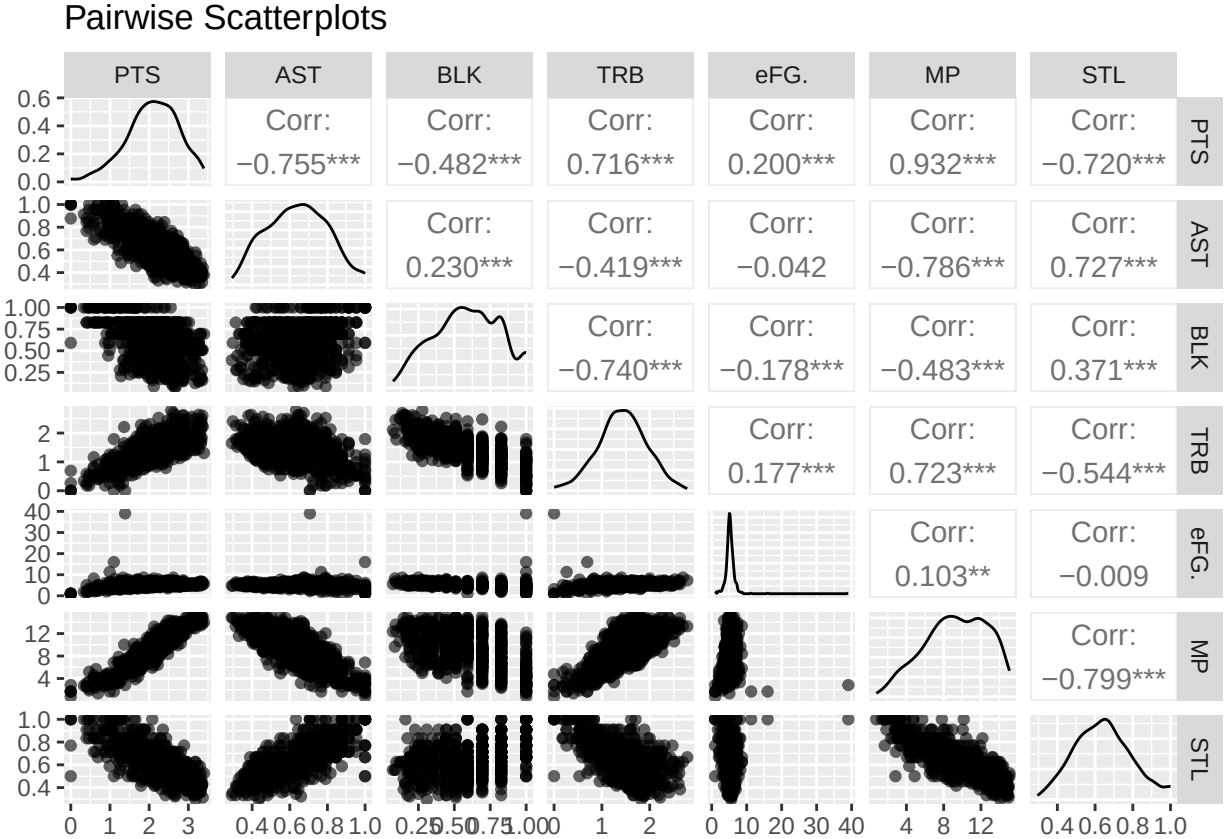
**c) Compare number of significant predictors, different coefficient estimates, adj R^2, and VIF to the test model**

- Number of significant predictors: similar
- different estimates but within 2 SE: satisfied.
- similar R^2 adj
- similar VIFs ### d) Verify assumptions for using residual plots in MLR again for test model #### i. Scatterplot of Response vs Fitted values

```
y_hat <- fitted(final_model_test)

plot(y_hat, final_model_test$Salary,
    xlab = "Fitted",
    ylab = "Salary",
    main = "Salary vs Fitted")
```

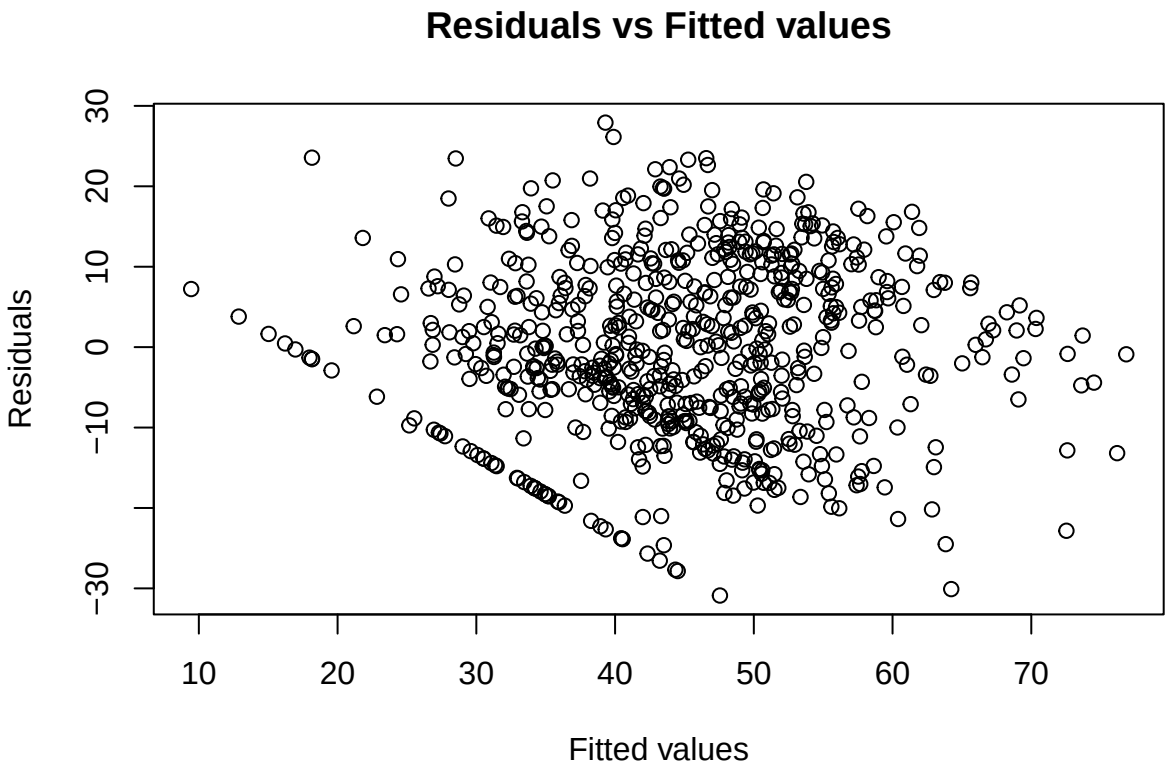

#### ii. Pairwise scatterplots of predictors

```
ggpairs(test_transformed[, c("PTS", "AST", "BLK", "TRB", "eFG.", "MP", "STL")],
        title = "Pairwise Scatterplots",
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6, size = 1.5)),
        progress = FALSE)
```



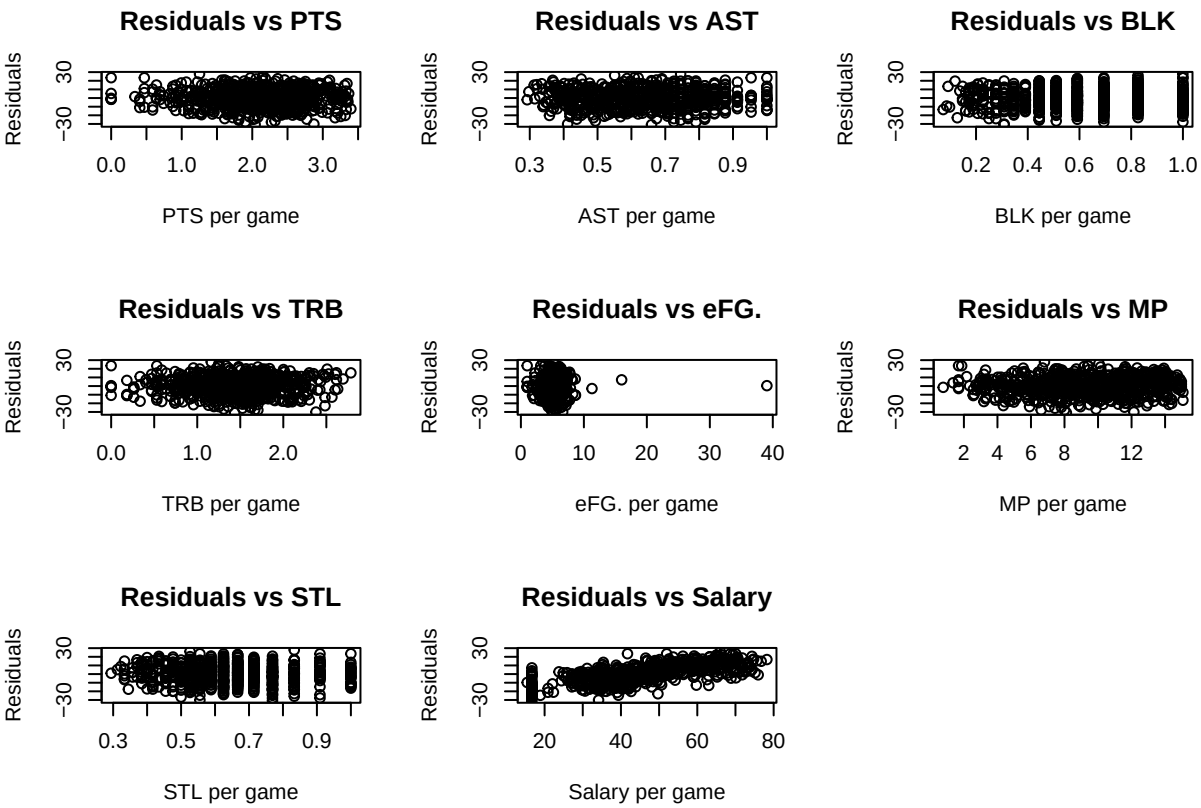

### e) Check all assumptions (linearity, uncorrelated errors, constant variance, normality) again for test model #### i. Residuals vs fitted values

```r
y_value <- resid(final_model_test)
x_value <- fitted(final_model_test)
plot(x_value, y_value, xlab = "Fitted values", ylab = "Residuals", main = "Residuals vs Fitted values")
```



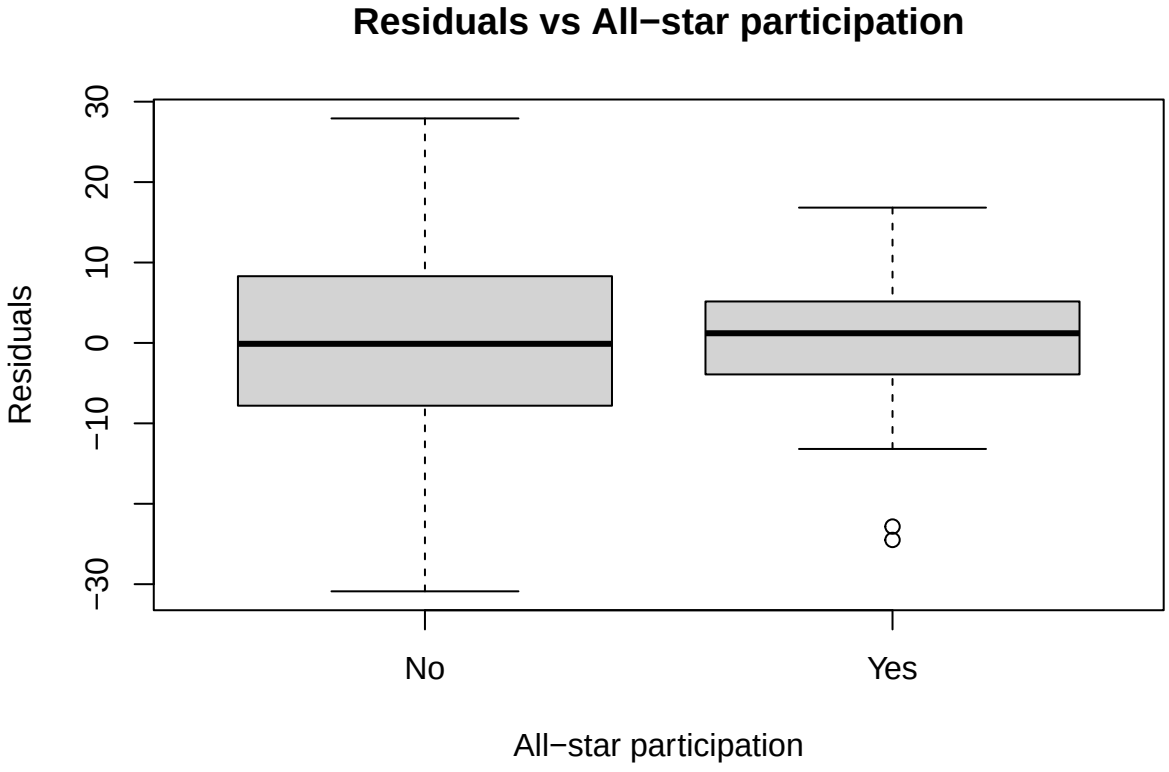

#### ii. Residuals vs each numerical predictors

```r
par(mfrow = c(3, 3))
for (var in numeric_vars) {
  plot(test_transformed[[var]], y_value,
       xlab = paste(var, "per game"),
       ylab = "Residuals",
       main = paste("Residuals vs", var)
  )
}
```

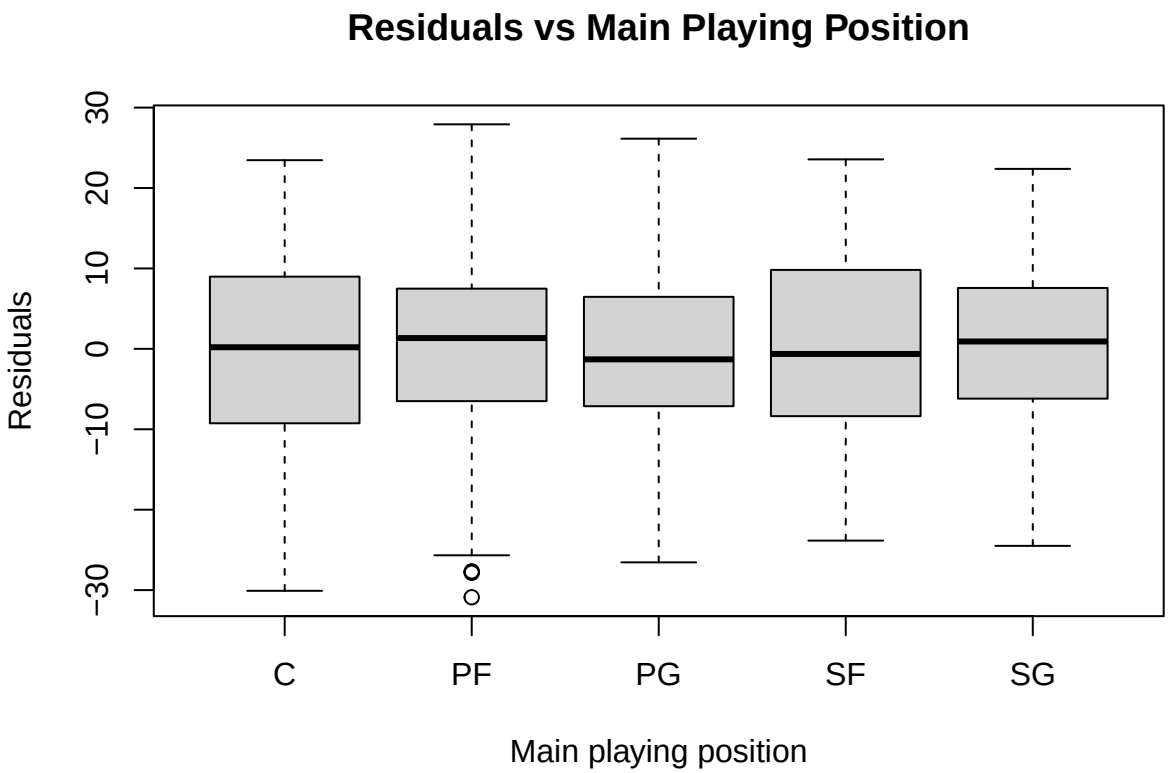



#### iii. Residuals vs each categorical predictors

```
boxplot(y_value ~ test_transformed$Play, xlab = "All-star participation", ylab = "Residuals", main = "Residual
```

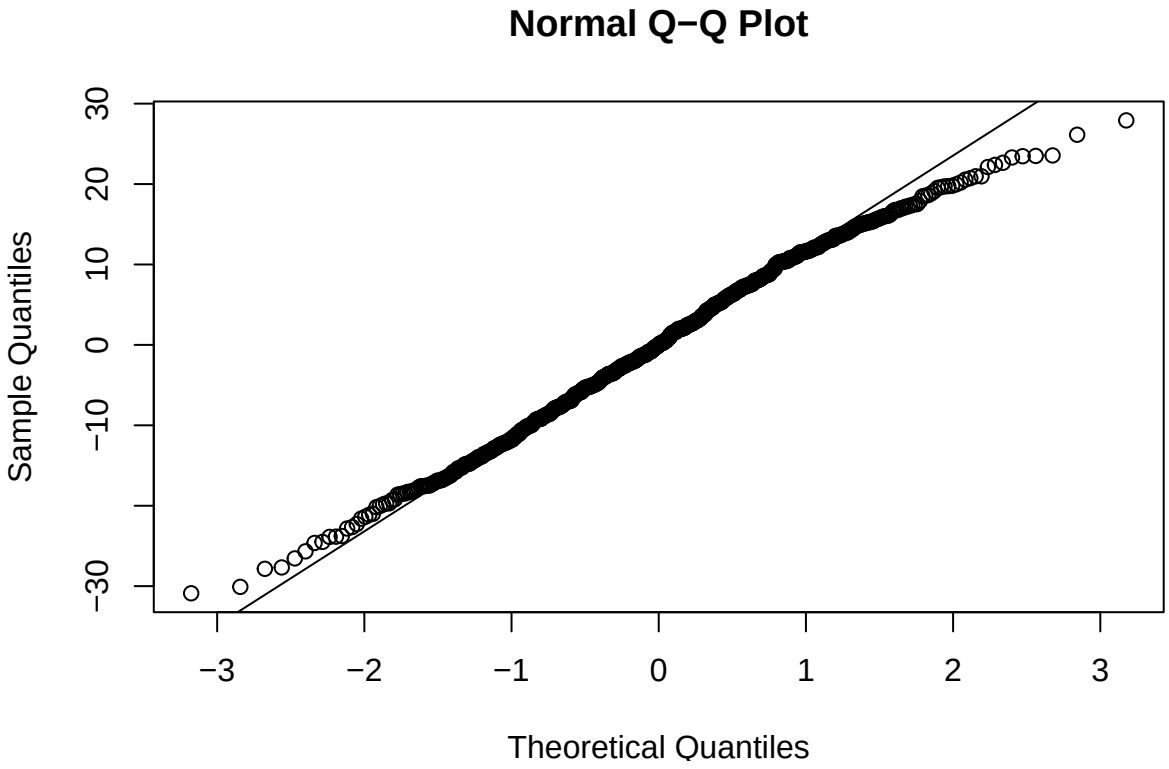



```
boxplot(y_value ~ test_transformed$Pos1, xlab = "Main playing position", ylab = "Residuals", main = "Residuals
```




#### iv. QQ plot

```
qqnorm(y_value)
qqline(y_value)
```




**f) Check Problematic Observations**

```
p = length(coef(final_model_test)) - 1
n = nrow(test_transformed)

hii <- hatvalues(final_model_test)
cutoff_hii <- 2 * (p + 1) / n
```

```
which(hii > cutoff_hii)
```

### i. Leverage Points

```
##   96  101  123  159  209  239  255  331  380  442  462  482  520  545  578  586
##   45   48   61   76   91  106  115  143  166  203  214  221  238  246  266  271
##  599  601  672  698  710  728  731  732  737  788  821  824  849  867  914  966
##  278  279  311  324  330  338  341  342  346  364  385  388  401  411  430  456
##  968  974 1055 1059 1065 1093 1108 1195 1203 1218 1282 1297 1342 1370 1406
##  457  461  501  503  506  519  530  570  574  581  608  617  637  649  669
```

```
ri <- rstandard(final_model_test)
which(ri > 2 | ri < - 2)
```

### ii. Outlier points

```
##   61  315  499  510  626  638  644  672  678  747  788  942  944  947  991 1046
##   29  137  229  235  287  296  300  311  314  350  364  445  447  449  469  497
## 1065 1080 1089 1113 1168 1169 1323
##  506  512  517  531  556  557  630
```

```
di <- cooks.distance(final_model_test)
cutoff_di <- qf(0.5, p + 1, n - p - 1)
which(di > cutoff_di)
```

### iii. Influential on all fitted values

```
## named integer(0)
```

```
dffits <- dffits(final_model_test)
cutoff_dffits <- 2 * sqrt((p + 1) / (n))
which(abs(dffits) > cutoff_dffits)
```

### iv. Influential on own fitted values

```
##  101  315  496  510  644  672  678  728  747  788  864  867 1065 1218 1323 1342
##   48  137  227  235  300  311  314  338  350  364  409  411  506  581  630  637
```

```
dfbetas <- dfbetas(final_model_test)
cutoff_dfbetas <- 2 / sqrt(n)

for(i in 1:5){
  print(paste0("Beta ", i-1))
  print(which(abs(dfbetas[,i]) > cutoff_dfbetas))    # this checks all betas in a loop
}
```

### v. Influential on coefficients

```
## [1] "Beta 0"
##   10   14   21   93  197  228  299  315  337  447  455  496  510  523  578  628
##    5    8   11   44   87  101  131  137  148  205  210  227  235  240  266  288
##  631  634  642  644  646  678  692  725  774  788  799  800  837  864  913  928
##  291  293  298  300  302  314  322  337  359  364  371  372  392  409  429  439
##  947  987 1019 1059 1096 1104 1137 1160 1169 1244 1296 1317 1323 1342
##  449  466  481  503  521  527  541  552  557  591  616  626  630  637
## [1] "Beta 1"
##   14   15   21   70   72   93  109  117  131  197  212  299  349  452  455  510
##    8    9   11   35   36   44   54   57   64   87   93  131  150  209  210  235
##  523  557  573  628  646  678  692  698  725  774  788  790  837  864  913  928
```

```
##  240  254  263  288  302  314  322  324  337  359  364  366  392  409  429  439
##  947 1065 1102 1104 1137 1148 1169 1264 1269 1296 1317 1323
##  449  506  525  527  541  547  557  598  601  616  626  630
## [1] "Beta 2"
##   10   14   21   93  197  212  228  299  315  337  447  455  459  496  523  578
##    5    8   11   44   87   93  101  131  137  148  205  210  212  227  240  266
##  631  634  642  644  646  678  692  725  755  774  799  800  814  837  864  947
##  291  293  298  300  302  314  322  337  352  359  371  372  379  392  409  449
##  956  987 1059 1096 1104 1169 1170 1237 1244 1296 1317 1323 1342
##  452  466  503  521  527  557  558  587  591  616  626  630  637
## [1] "Beta 3"
## 462 510 578 624 644 788 790 956
## 214 235 266 285 300 364 366 452
## [1] "Beta 4"
##   35   70   72   93  101  117  124  137  156  180  219  229  315  388  399  432
##   19   35   36   44   48   57   62   67   74   81   97  102  137  170  178  198
##  510  541  573  610  626  672  678  704  711  747  778  791  814  822  838  850
##  235  244  263  281  287  311  314  326  331  350  360  367  379  386  393  402
##  863  869  924  944  956 1034 1060 1096 1102 1104 1107 1113 1117 1160 1168 1169
##  408  412  435  447  452  488  504  521  525  527  529  531  532  552  556  557
## 1170 1244 1268 1283 1296 1323 1344 1356 1399
##  558  591  600  609  616  630  639  643  663
```