First Half

April 6, 2022

[1]:
```
install.packages("qqplotr")
```

```
Installing package into ‘/home/jupyter/R/x86_64-pc-linux-gnu-library/4.1’
(as ‘lib’ is unspecified)
```

[2]:
```
install.packages("leaps")
```

```
Installing package into ‘/home/jupyter/R/x86_64-pc-linux-gnu-library/4.1’
(as ‘lib’ is unspecified)
```

[3]:
```
library(ggplot2)
library(qqplotr)
library(tidyverse)
library(repr)
library(tidymodels)
library(stringr)
library(leaps)
```

```
Attaching package: ‘qqplotr’


The following objects are masked from ‘package:ggplot2’:

    stat_qq_line, StatQqLine


Warning message in system("timedatectl", intern = TRUE):
"running command 'timedatectl' had status 1"
  Attaching packages                                   tidyverse
1.3.1

  tibble  3.1.6      dplyr   1.0.8
  tidyr   1.2.0      stringr 1.4.0
  readr   2.1.2      forcats 0.5.1
  purrr   0.3.4
```

```
── Conflicts ──────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter()         masks stats::filter()
✖ dplyr::lag()            masks stats::lag()
✖ qqplotr::stat_qq_line() masks ggplot2::stat_qq_line()
```

```
── Attaching packages ──────────────────────── tidymodels 0.2.0 ──

✔ broom        0.7.12      ✔ rsample      0.1.1
✔ dials        0.1.0       ✔ tune         0.2.0
✔ infer        1.0.0       ✔ workflows    0.2.6
✔ modeldata    0.1.1       ✔ workflowsets 0.2.1
✔ parsnip      0.2.1       ✔ yardstick    0.0.9
✔ recipes      0.2.0

── Conflicts ─────────────────────────────────────── tidymodels_conflicts() ──
✖ scales::discard()       masks purrr::discard()
✖ dplyr::filter()         masks stats::filter()
✖ recipes::fixed()        masks stringr::fixed()
✖ dplyr::lag()            masks stats::lag()
✖ yardstick::spec()       masks readr::spec()
✖ qqplotr::stat_qq_line() masks ggplot2::stat_qq_line()
✖ recipes::step()         masks stats::step()
• Use tidymodels_prefer() to resolve common conflicts.
```

[5]:
```
data <- read.csv(url("https://drive.google.com/uc?
↪export=download&id=1_MECmUXZuuILYeEOfonSGqodW6qVdhsS"))
```

[6]:
```
data <- mutate(data, str_replace(data$Age, " \\s*\\([^\\)]+\\)", ""))
data <- mutate(data, Age = str_replace(data$Age, " \\s*\\([^\\)]+\\)", ""))
data <- mutate(data, Current.Rank = str_replace(data$Current.Rank, "␣
 ↪\\s*\\([^\\)]+\\)", ""))
data <- mutate(data, Best.Rank = str_replace(data$Best.Rank, "␣
 ↪\\s*\\([^\\)]+\\)", ""))

money <- c(data$Prize.Money)
money <- money %>%
        lapply(gsub, pattern="$", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="US", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="all-time leader in earnings", fixed=TRUE,␣
 ↪replacement="") %>%
        lapply(gsub, pattern="All-time leader in earnings", fixed=TRUE,␣
 ↪replacement="") %>%
        lapply(gsub, pattern="all-time in earnings", fixed=TRUE,␣
 ↪replacement="") %>%
        lapply(gsub, pattern="11th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="24th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="10th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="14th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="2nd", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="27th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="15th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="30th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="4th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="28th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="6th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="33rd", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="26th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="24th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="48th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="41st", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="24th", fixed=TRUE, replacement="") %>%
        lapply(gsub, pattern="15th", fixed=TRUE, replacement="")

data_selected <- data %>%
            mutate(data, Prize.Money = money) %>%
            select(Age, Name,Country, Current.Rank, Best.Rank, Prize.
 ↪Money,Seasons) %>%
            mutate(Prize.Money = gsub(",","", Prize.Money))

tidy_data <- data_selected %>%
        filter(Prize.Money != "") %>%
        mutate(Prize.Money = as.numeric(Prize.Money)) %>%
        mutate(Age = as.numeric(Age)) %>%
        mutate(Current.Rank = as.numeric(Current.Rank)) %>%
```

```
        mutate(Best.Rank = as.numeric(Best.Rank)) %>%
        mutate(Seasons = as.numeric(Seasons))

tidy_data <- drop_na(tidy_data)
```

```
Warning message in mask$eval_all_mutate(quo):
"NAs introduced by coercion"
```

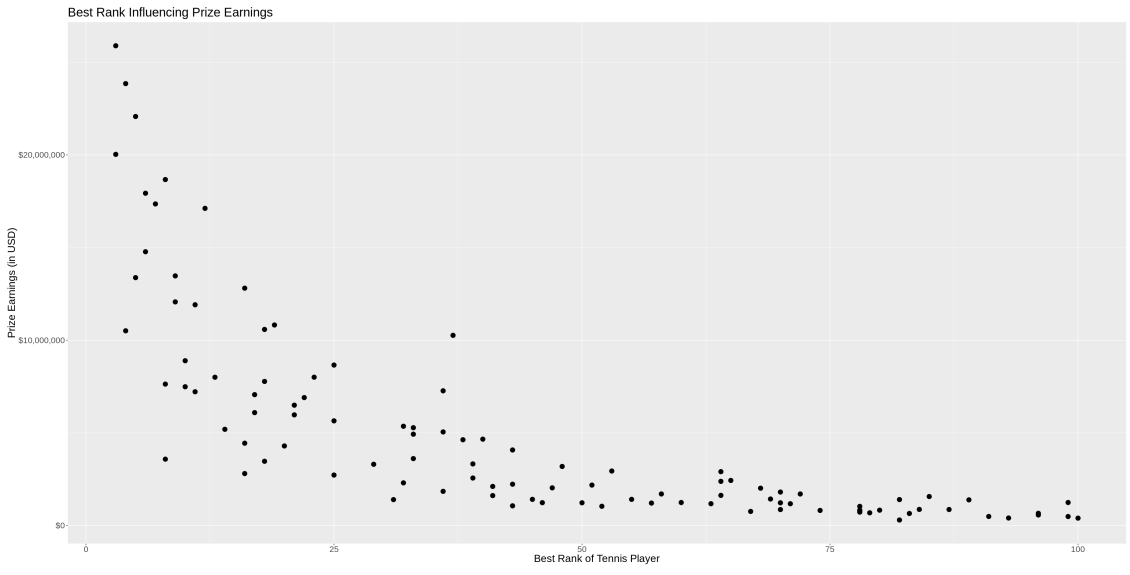
[7]:
```
# Find average prize money for each country's players
table1 <- tidy_data %>%
    group_by(Country) %>%
    summarize(avg_award_in_USD = mean(Prize.Money))

avg_award_in_USD <- table1$avg_award_in_USD

# count each country's number of players and then bind the data with the␣
 ↪average prize money column from above
final_table <- tidy_data %>%
    group_by(Country) %>%
    summarize(n = n()) %>%
    bind_cols(avg_award_in_USD) %>%
    mutate(avg_award_in_USD = ...3) %>%
    select(-...3)

# Find top 10 country with the most players
top_10 <- final_table %>%
        arrange(n) %>%
        tail(10)

# Plot the number of players for each top 10 country
top_10_graph <- ggplot(top_10,aes(x = Country, y = n)) +
        geom_bar(stat = "identity") +
        labs(x = "Country", y = "Number of People in Top 500 Tennis Players") +
        ggtitle("Top 10 Countries with most People in Top 500 Tennis Players") +
        coord_flip()

top_10_names <- pull(top_10, Country)
```

```
New names:
* `` -> ...3
```

[14]:
```
# Start working on Top 10
top_10_data <- tidy_data %>%
                filter(Country == "United States"| Country == "United Kingdom"␣
 ↪| Country == "Spain" |
```

```
                    Country == "Russian Federation" | Country == "Japan" |␣
 ↪Country == "Italy" |
                    Country == "Germany" | Country == "France" | Country ==␣
 ↪"Australia"| Country == "Argentina") %>%
                select(-Name)

# averaged Best Rank
top_10_mean_money_over_rank <- top_10_data %>%
    group_by(Best.Rank) %>%
    summarize(mean_Prize_Money = mean(Prize.Money))

top_10_data <- filter(top_10_data, Prize.Money != 61544007 & Prize.Money !=␣
 ↪119601561)
```

[15]:
```
# Start working on top 100 ranked players
top_10_data_top_100 <- filter(top_10_data, Best.Rank <= 100)

options(repr.plot.width = 30, repr.plot.height = 15)

bestRank_over_money_plot <- ggplot(top_10_data_top_100, aes(x = Best.Rank, y =␣
 ↪Prize.Money)) +
        geom_point(size = 4) +
        labs(x = "Best Rank of Tennis Player", y = "Prize Earnings (in USD)") +
        scale_y_continuous(labels = dollar_format()) +
        ggtitle("Best Rank Influencing Prize Earnings") +
        theme(text = element_text(size = 20))

bestRank_over_money_plot
```

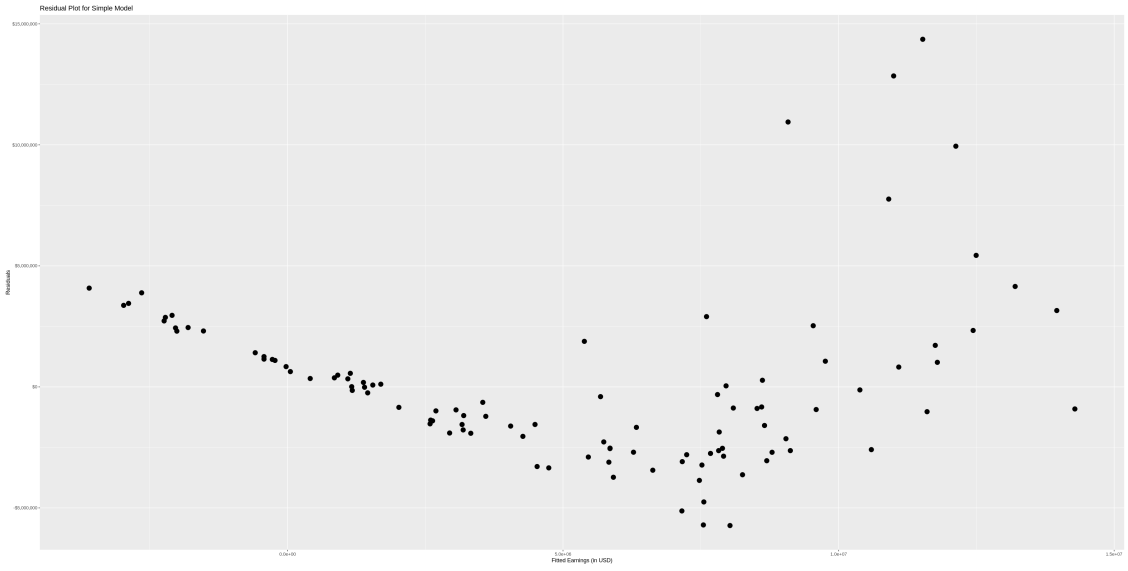


Now start model fitting

[23]:
```
#After removing those two outliers
# Prize Money = Age + Seasons + Best Rank + Best Rank^2 + Age * Best Rank
simple_model <- lm(Prize.Money~Age+Best.Rank+Seasons, data =top_10_data_top_100)
simple_model_summary <- summary(simple_model)
simple_residual_plot <- ggplot(simple_model,  aes(x = fitted.values(simple_model), y = residuals(simple_model))) +
       geom_point(size = 4) +
       labs(x = "Fitted Earnings (in USD)", y = "Residuals") +
       scale_y_continuous(labels = dollar_format()) +
       ggtitle("Residual Plot for Simple Model")
       theme(text = element_text(size = 20))

simple_normal_plot <- ggplot(simple_model, mapping = aes(sample =residuals(simple_model))) +
       stat_qq_point(size = 2) +
       ggtitle("Normal Plot for Simple Model") +
       theme(text = element_text(size = 20))

AIC_simple <- AIC(simple_model)
BIC_simple <- BIC(simple_model)



simple_model_summary
simple_residual_plot
simple_normal_plot
AIC_simple
BIC_simple
```

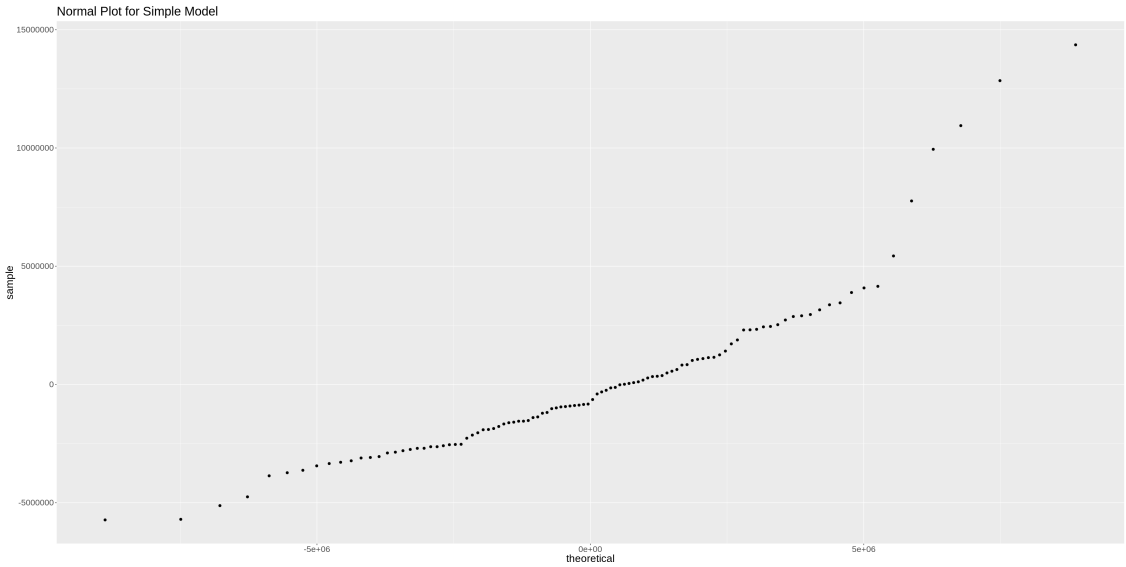
```
List of 1
 $ text:List of 11
  ..$ family       : NULL
  ..$ face         : NULL
  ..$ colour       : NULL
  ..$ size         : num 20
  ..$ hjust        : NULL
  ..$ vjust        : NULL
  ..$ angle        : NULL
  ..$ lineheight   : NULL
  ..$ margin       : NULL
  ..$ debug        : NULL
  ..$ inherit.blank: logi FALSE
  ..- attr(*, "class")= chr [1:2] "element_text" "element"
 - attr(*, "class")= chr [1:2] "theme" "gg"
```

```
 - attr(*, "complete")= logi FALSE
 - attr(*, "validate")= logi TRUE


Call:
lm(formula = Prize.Money ~ Age + Best.Rank + Seasons, data = top_10_data_top_100)

Residuals:
     Min       1Q   Median       3Q      Max
-5728201 -2337373  -733594  1174226 14362439

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  6420403    2821847   2.275   0.0250 *
Age           -27402     147947  -0.185   0.8534
Best.Rank    -107022      15895  -6.733 1.06e-09 ***
Seasons       448376     172570   2.598   0.0108 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3495000 on 100 degrees of freedom
Multiple R-squared:  0.6329,        Adjusted R-squared:  0.6219
F-statistic: 57.46 on 3 and 100 DF,  p-value: < 2.2e-16
```



3434.97594475095

3448.19789924666



[18]:
```
# regsusbets model selection
data_without_country <- select(top_10_data_top_100, -Country)
s <- regsubsets(Prize.Money~., data = data_without_country, method =␣
 ↪"exhaustive")
model_selection_stats <- summary(s)

model_selection_stats
model_selection_stats$adjr2
model_selection_stats$cp
model_selection_stats$rsq
```

```
Subset selection object
Call: regsubsets.formula(Prize.Money ~ ., data = data_without_country,
    method = "exhaustive")
4 Variables  (and intercept)
             Forced in Forced out
Age              FALSE      FALSE
Current.Rank     FALSE      FALSE
Best.Rank        FALSE      FALSE
Seasons          FALSE      FALSE
1 subsets of each size up to 4
Selection Algorithm: exhaustive
         Age Current.Rank Best.Rank Seasons
1  ( 1 ) " " " "          "*"       " "
2  ( 1 ) " " " "          "*"       "*"
3  ( 1 ) " " "*"          "*"       "*"
4  ( 1 ) "*" "*"          "*"       "*"
```

1. 0.541710918200971 2. 0.625474721696029 3. 0.642534700289479 4. 0.639050164968224

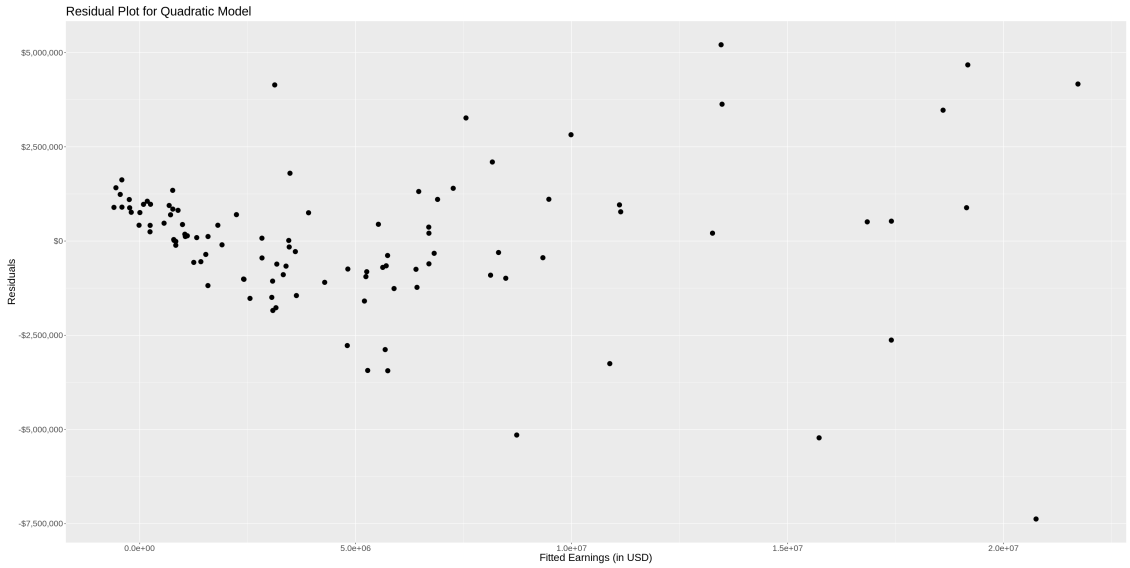
1. 29.5068782602042 2. 6.79864357153947 3. 3.03462060844338 4. 5

1. 0.546160326762127 2. 0.632747057197077 3. 0.652946310960659 4. 0.653067634289847

[46]:
```
# Prize Money = Age + Seasons + Best Rank + Best Rank^2
quadratic_model <- lm(Prize.Money~Age+I(Best.Rank^2)+I(Best.Rank^3)+I(Best.
 ↪Rank^4)+I(Best.Rank^5)+I(Best.Rank^6)+Best.Rank+Seasons, data =␣
 ↪top_10_data_top_100)
quadratic_model_summary <- summary(quadratic_model)
quadratic_model_plot <- ggplot(quadratic_model, aes(y =␣
 ↪residuals(quadratic_model), x = fitted.values(quadratic_model))) +
        geom_point(size = 4) +
        labs(x = "Fitted Earnings (in USD)", y = "Residuals") +
        scale_y_continuous(labels = dollar_format()) +
        ggtitle("Residual Plot for Quadratic Model") +
        theme(text = element_text(size = 20))

quadratic_normal_plot <- ggplot(quadratic_model, mapping = aes(sample =␣
 ↪residuals(quadratic_model))) +
        stat_qq_point(size = 2) +
        ggtitle("Normal Plot for Quadratic Model") +
        theme(text = element_text(size = 20))

AIC_quadratic <- AIC(quadratic_model)
BIC_quadratic <- BIC(quadratic_model)


quadratic_model_summary
quadratic_model_plot
quadratic_normal_plot
AIC_quadratic
BIC_quadratic
```

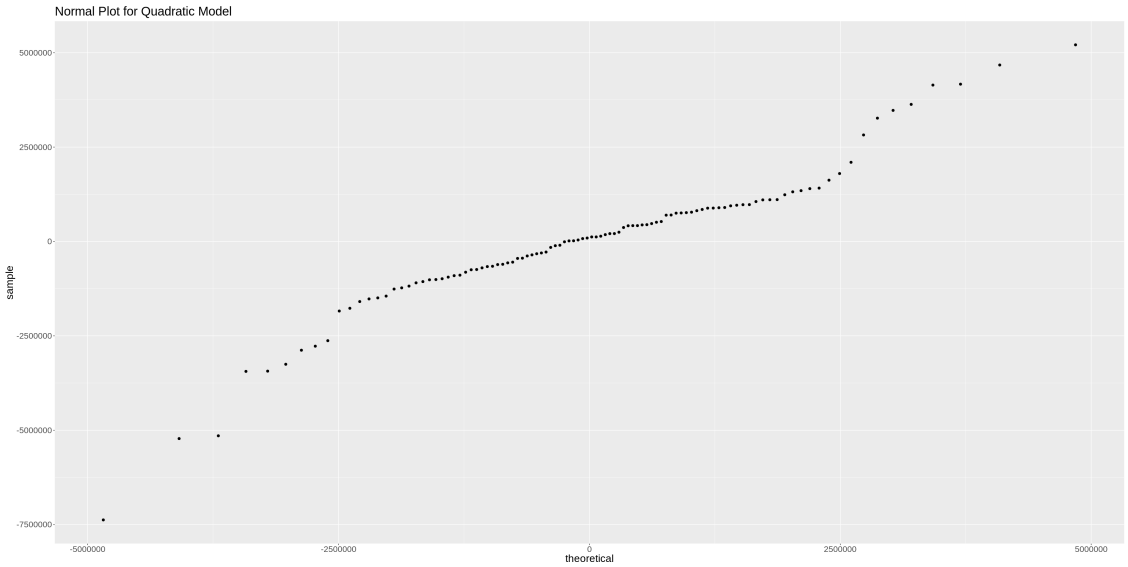
```
Call:
lm(formula = Prize.Money ~ Age + I(Best.Rank^2) + I(Best.Rank^3) +
    I(Best.Rank^4) + I(Best.Rank^5) + I(Best.Rank^6) + Best.Rank +
    Seasons, data = top_10_data_top_100)

Residuals:
     Min       1Q   Median       3Q      Max
-7376272  -833768   105608   886957  5209099

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     2.391e+07  2.538e+06   9.419 2.89e-15 ***
Age            -7.881e+02  8.639e+04  -0.009 0.992740
I(Best.Rank^2)  1.890e+05  4.147e+04   4.558 1.54e-05 ***
```

```
I(Best.Rank^3) -5.655e+03  1.512e+03  -3.741 0.000314 ***
I(Best.Rank^4)  8.852e+01  2.694e+01   3.286 0.001423 **
I(Best.Rank^5) -6.876e-01  2.302e-01  -2.987 0.003580 **
I(Best.Rank^6)  2.089e-03  7.543e-04   2.770 0.006750 **
Best.Rank      -3.249e+06  5.021e+05  -6.471 4.22e-09 ***
Seasons         4.310e+05  9.981e+04   4.318 3.87e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1957000 on 95 degrees of freedom
Multiple R-squared:  0.8906,        Adjusted R-squared:  0.8814
F-statistic: 96.69 on 8 and 95 DF,  p-value: < 2.2e-16
```



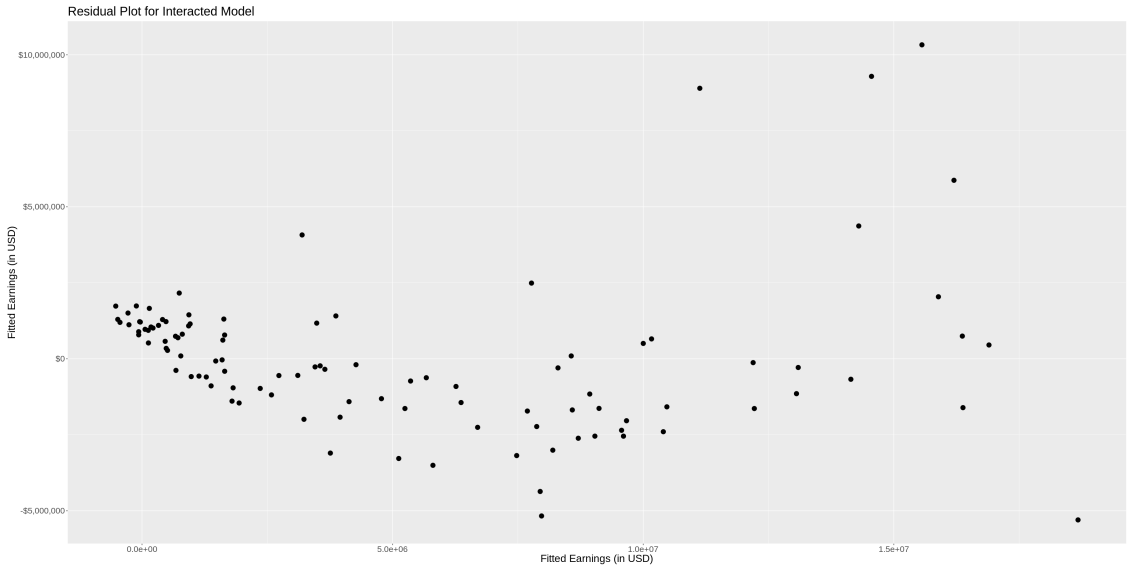
3319.0428144187

3345.48672341011



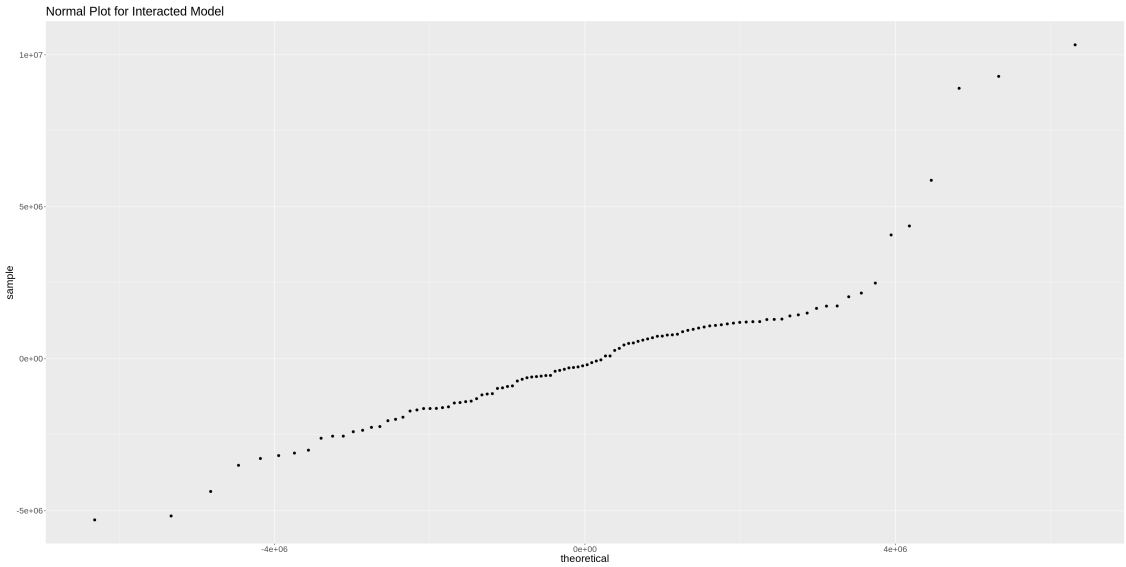
[22]:
```
#Prize Money = Age + Seasons + Best Rank
interacted_model <- lm(Prize.Money~Age*Best.Rank+I(Best.Rank^2)+Seasons, data =␣
 ↪top_10_data_top_100)
interacted_model_summary <- summary(interacted_model)
interacted_model_plot <- interacted_model %>%
        ggplot(aes(y = residuals(interacted_model), x = fitted.
 ↪values(interacted_model))) +
        geom_point(size = 4) +
        labs(x = "Fitted Earnings (in USD)", y = "Fitted Earnings (in USD)") +
        scale_y_continuous(labels = dollar_format()) +
        ggtitle("Residual Plot for Interacted Model") +
        theme(text = element_text(size = 20))

interacted_normal_plot <- ggplot(mapping = aes(sample =␣
 ↪residuals(interacted_model))) +
        stat_qq_point(size = 2) +
        ggtitle("Normal Plot for Interacted Model") +
        theme(text = element_text(size = 20))

AIC_interacted<- AIC(interacted_model)
BIC_interacted<- BIC(interacted_model)


interacted_model_summary
interacted_model_plot
interacted_normal_plot
AIC_interacted
BIC_interacted
```

```
Call:
lm(formula = Prize.Money ~ Age * Best.Rank + I(Best.Rank^2) +
    Seasons, data = top_10_data_top_100)

Residuals:
     Min       1Q   Median       3Q      Max
-5298825 -1441774  -212167  1087462 10329109

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    3295699.5  3659182.9   0.901 0.369974
Age             280385.7   164526.7   1.704 0.091513 .
Best.Rank      -243513.0    68643.6  -3.547 0.000599 ***
I(Best.Rank^2)    3079.3      347.1   8.871  3.4e-14 ***
Seasons         344501.1   136853.7   2.517 0.013448 *
Age:Best.Rank    -5885.7     2066.5  -2.848 0.005359 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2513000 on 98 degrees of freedom
Multiple R-squared:  0.8141,        Adjusted R-squared:  0.8046
F-statistic: 85.81 on 5 and 98 DF,  p-value: < 2.2e-16
```



3368.22230491242

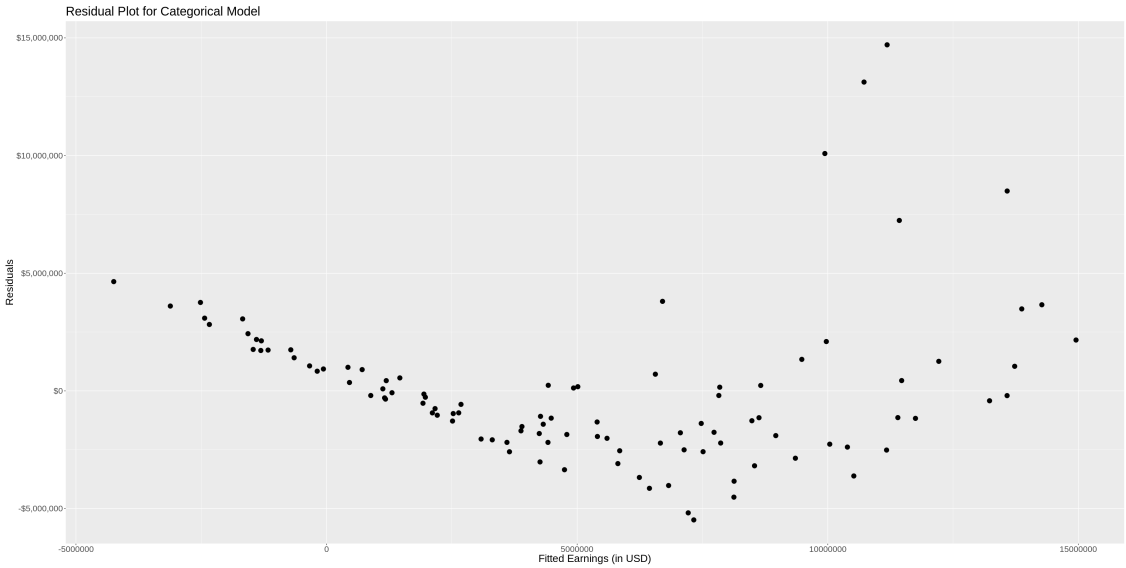
3386.73304120641



[21]:
```
# Categorical model
categorical_model <- lm(Prize.Money~Age+Current.Rank+Best.Rank+Seasons+Country,␣
 ↪data = top_10_data_top_100)
categorical_model_summary <- summary(categorical_model)
categorical_model_plot <- ggplot(categorical_model, aes(y =␣
 ↪residuals(categorical_model), x = fitted.values(categorical_model))) +
        geom_point(size = 4) +
        labs(x = "Fitted Earnings (in USD)", y = "Residuals") +
        scale_y_continuous(labels = dollar_format()) +
        ggtitle("Residual Plot for Categorical Model") +
        theme(text = element_text(size = 20))

categorical_normal_plot <- ggplot(categorical_model, mapping = aes(sample =␣
 ↪residuals(categorical_model))) +
        stat_qq_point(size = 2) +
        ggtitle("Normal Plot for Categorical Model") +
        theme(text = element_text(size = 20))

AIC_categorical <- AIC(categorical_model)
BIC_categorical <- BIC(categorical_model)

categorical_model_summary
categorical_model_plot
categorical_normal_plot
AIC_categorical
BIC_categorical
```

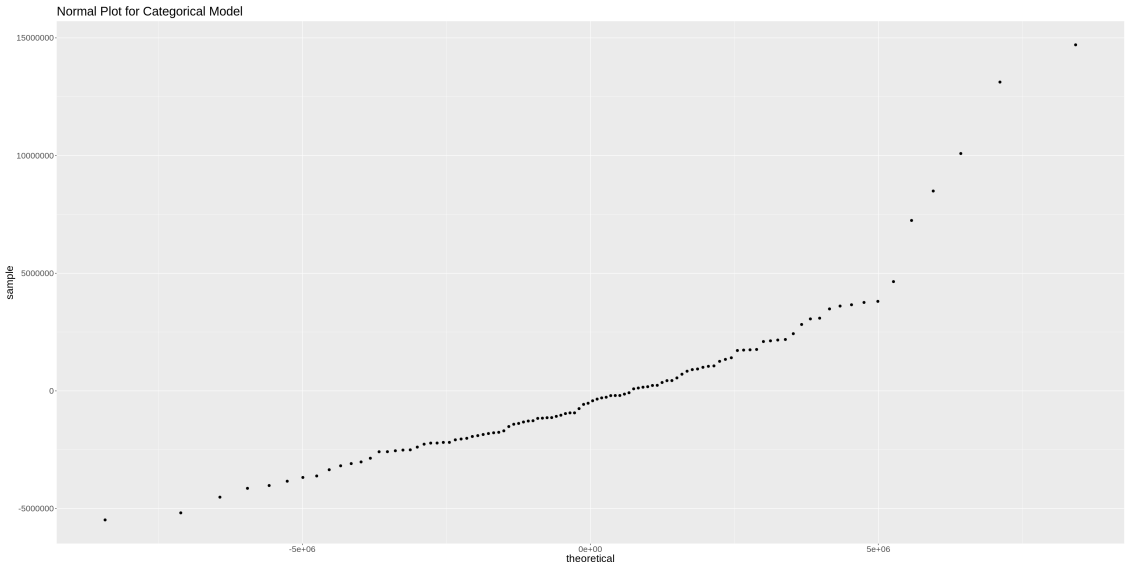
```
Call:
```

```
lm(formula = Prize.Money ~ Age + Current.Rank + Best.Rank + Seasons +
    Country, data = top_10_data_top_100)

Residuals:
     Min       1Q   Median       3Q      Max
-5483248 -2022586  -472673  1110080 14705406

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                5691324    3494011   1.629   0.1068
Age                         -14136     164473  -0.086   0.9317
Current.Rank                -10073       4652  -2.165   0.0330 *
Best.Rank                   -82722      19147  -4.320    4e-05 ***
Seasons                     528429     186551   2.833   0.0057 **
CountryAustralia           -397441    1706817  -0.233   0.8164
CountryFrance               660071    1582046   0.417   0.6775
CountryGermany              653078    1522261   0.429   0.6689
CountryItaly               -613308    1529071  -0.401   0.6893
CountryJapan               -901127    1806119  -0.499   0.6190
CountryRussian Federation  -933860    1861437  -0.502   0.6171
CountrySpain               -259503    1555768  -0.167   0.8679
CountryUnited Kingdom     -2739247    2125156  -1.289   0.2007
CountryUnited States       -327982    1485311  -0.221   0.8257
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3498000 on 90 degrees of freedom
Multiple R-squared:  0.6691,        Adjusted R-squared:  0.6213
F-statistic:    14 on 13 and 90 DF,  p-value: < 2.2e-16
```



3444.16941985904

3483.83528334616



[ ]: