What Makes the Best Tennis Player?

Steven Herrera and Ethan Shen

11/09/2018

```
library(tidyverse)
library(olsrr)
library(cowplot)
library(car)
library(broom)
library(knitr)
library(arm)
library(tidyr)
library(pROC)
library(arm)
library(rlm)
```

Data

Using data manipulation skills in R, we shaped the dataset to show each observation as the outcome of the match for the winner and the loser of each match from 2010-2017. Below, is a glimpse of our dataset.

Because we have 11,037 observations, we will randomly select observations to be included in a smaller dataset so that we can effectively examine exploratory data analysis. Below, is our code on how we randomly selected the observations in our new dataset.

```
set.seed(1234)
ten <- tennis %>% sample_n(1000)
```

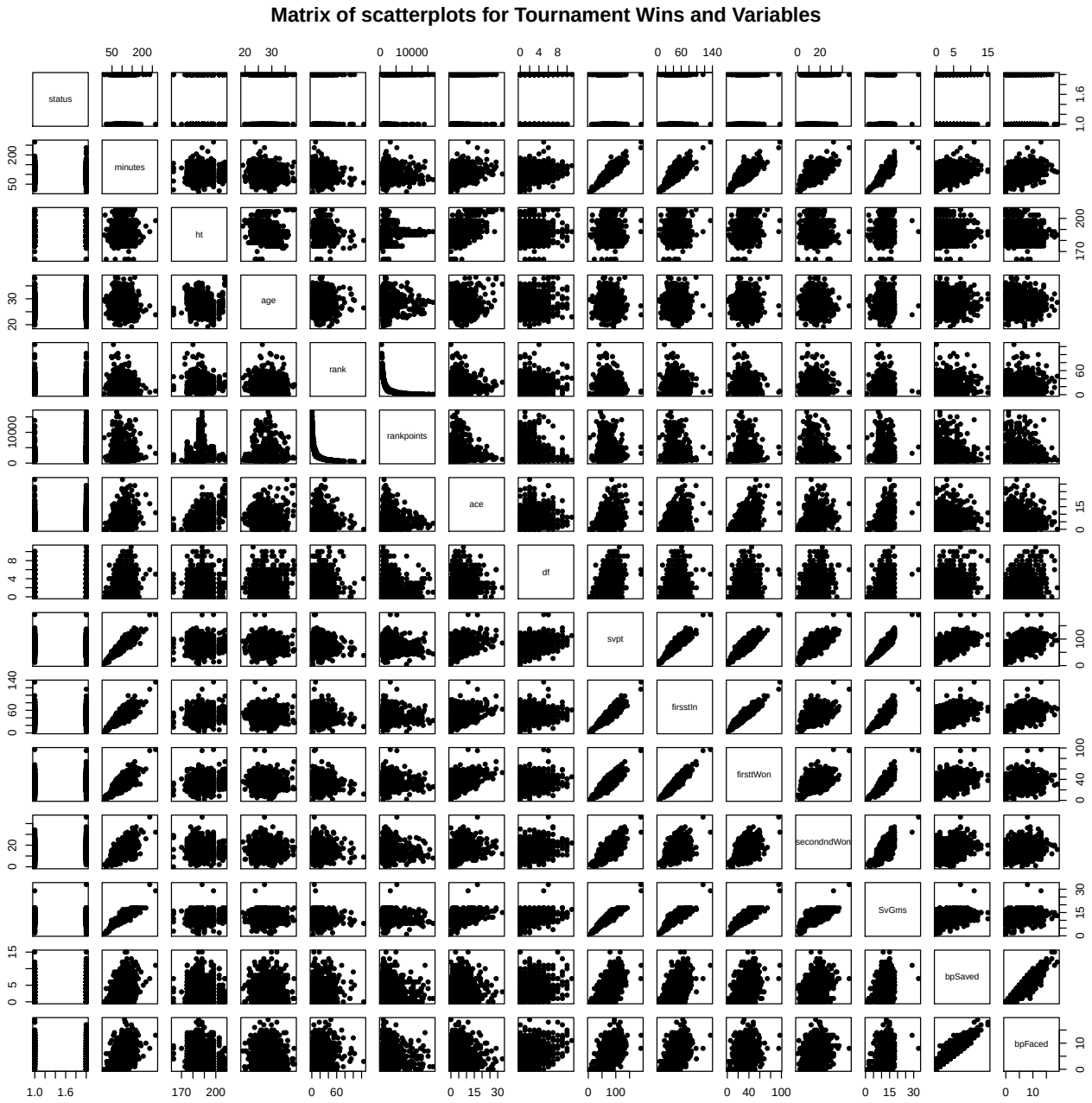
Here is what our new dataset looks like:

Exploratory Data Analysis

To begin our exploratory data analysis, we will examine a matrix plot of the variables in our dataset to consider multicollinearity and large leverage of certain observations.

Matrix Plot

```
ten <- ten %>%
  mutate(status = as.factor(status))

pairs(status ~ minutes + ht + age + rank + rankpoints + ace +
        df + svpt + firsstIn + firsttWon + secondndWon +
        SvGms + bpSaved + bpFaced, data=ten, pch = 16,
      main = "Matrix of scatterplots for Tournament Wins and Variables")
```

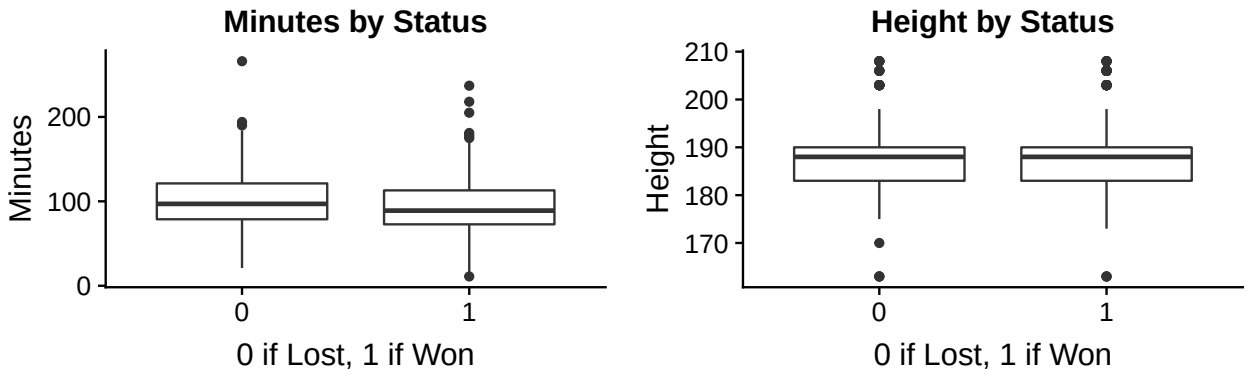


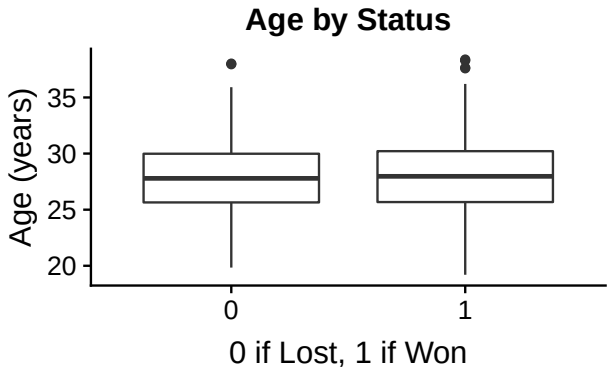
Looking at the matrix plot, we will already consider removing the following favirables because of multi-collinearity: `svpt`, `firsstIn`, `firsttWon`, `secondndWon`, `SvGms`, and `bpFaced`.

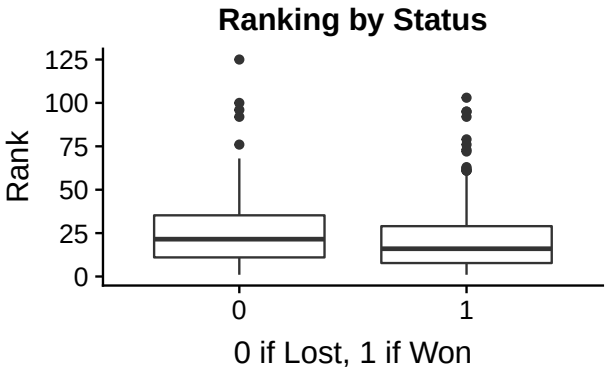
We will now look at box plots of the numeric variables we will include in our full model:

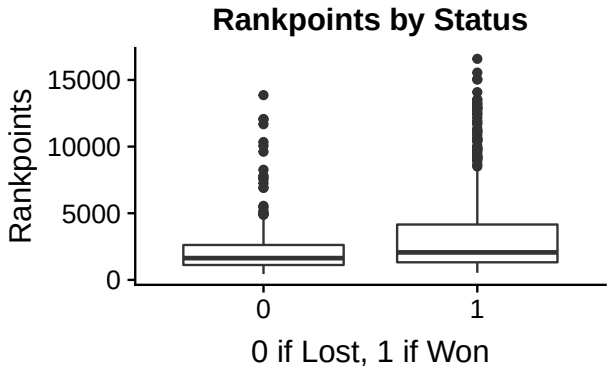
```
p1 <- ggplot(data=ten,aes(x=status,y=minutes, group=status)) +
  geom_boxplot() +
  labs(title="Minutes by Status",
       x = "0 if Lost, 1 if Won",
       y = "Minutes")
p2 <- ggplot(data=ten,aes(x=status,y=ht, group=status)) +
  geom_boxplot() +
  labs(title="Height by Status",
       x = "0 if Lost, 1 if Won",
       y = "Height")
```

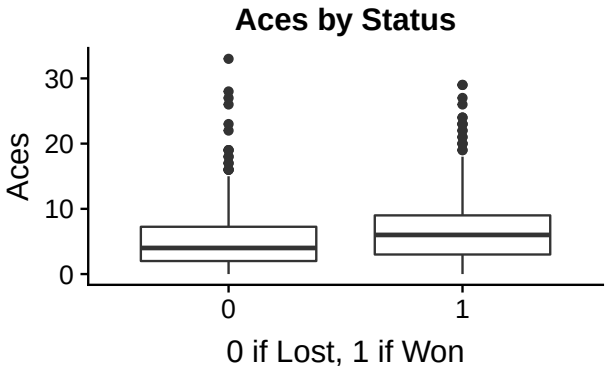
```
p3 <- ggplot(data=ten,aes(x=status,y=age, group=status)) +
  geom_boxplot() +
  labs(title="Age by Status",
       x = "0 if Lost, 1 if Won",
       y = "Age (years)")
p4 <- ggplot(data=ten,aes(x=status,y=rank, group=status)) +
  geom_boxplot() +
  labs(title="Ranking by Status",
       x = "0 if Lost, 1 if Won",
       y = "Rank")
p5 <- ggplot(data=ten,aes(x=status,y=rankpoints, group=status)) +
  geom_boxplot() +
  labs(title="Rankpoints by Status",
       x = "0 if Lost, 1 if Won",
       y = "Rankpoints")
p6 <- ggplot(data=ten,aes(x=status,y=ace, group=status)) +
  geom_boxplot() +
  labs(title="Aces by Status",
       x = "0 if Lost, 1 if Won",
       y = "Aces")
p7 <- ggplot(data=ten,aes(x=status,y=df, group=status)) +
  geom_boxplot() +
  labs(title="Double Faults by Status",
       x = "0 if Lost, 1 if Won",
       y = "Double Faults")
p8 <- ggplot(data=ten,aes(x=status,y=bpSaved, group=status)) +
  geom_boxplot() +
  labs(title="Saved Breakpoints by Status",
       x = "0 if Lost, 1 if Won",
       y = "Saved Breakpoints")

plot_grid(p1,p2,p3,p4,p5,
          p6,p7,p8,ncol=2)
```

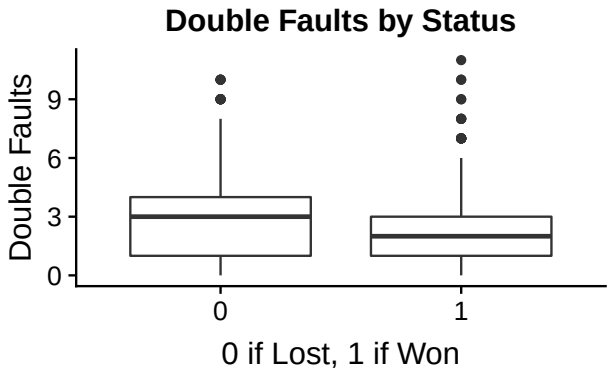


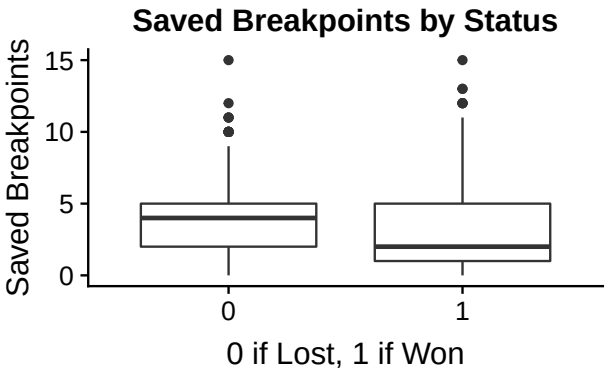






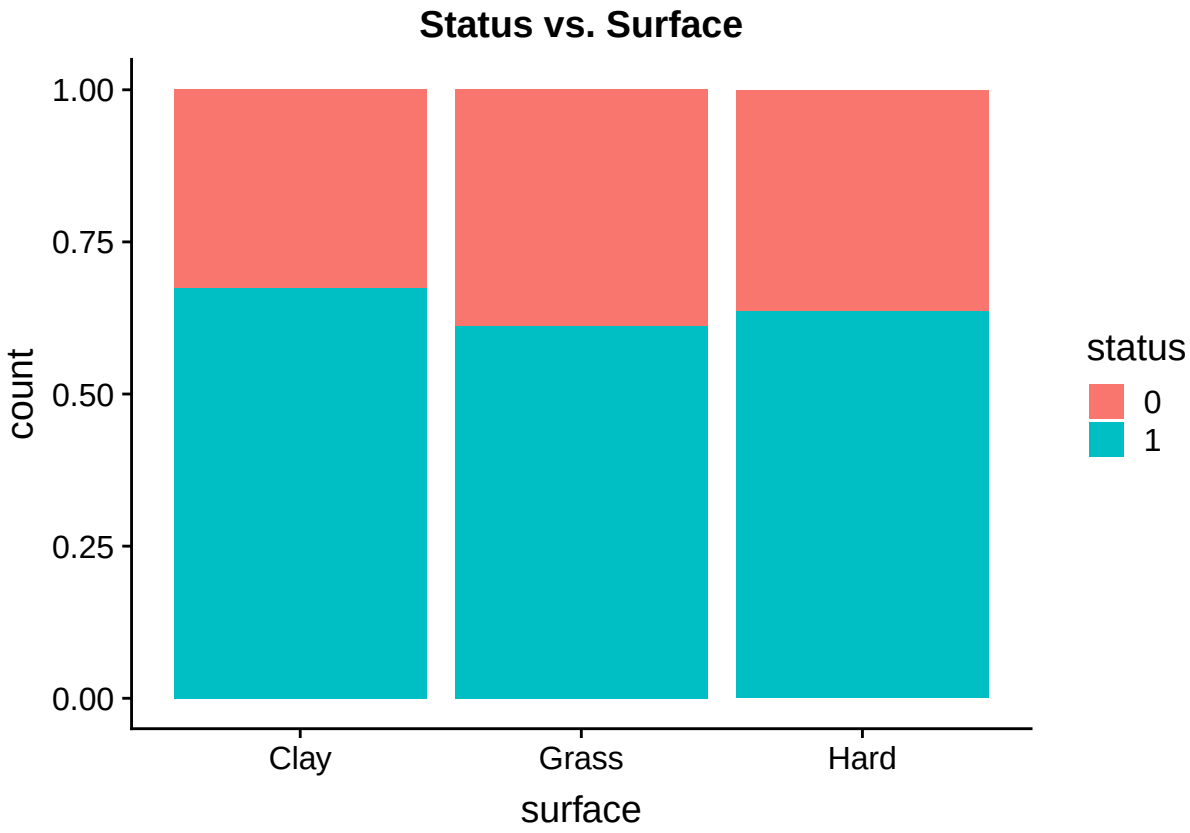






And we will include a stacked bar graph for the variable `surface`.

```
ggplot(data=ten,aes(x=surface, fill = status)) + geom_bar(position = "fill") +
  labs(title="Status vs. Surface")
```



In looking at all of these observations, it seems like the medians of the numeric distributions do not seem to differ that much by status of winning or losing. The same can be said about the proportions of winning and losing matches against all three surfaces. In creating our model, it could be difficult to see which variables could be helpful in differentiating between whether a player will win a match or not. But, we hope to see that a combination of these variables will be helpful in determining a model that best predicts the percentage of winning a match.

Logistic Regression Model

To begin our regression models, we will use all of the variables we deemed important from our exploratory data analysis.

```
full_model <- glm(status ~ minutes + ht + age + rank +
                  rankpoints + ace + df + bpSaved + surface,
                  family=binomial,data=ten)
kable(tidy(full_model), format="markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	5.060	2.219	2.280	0.023
minutes	-0.007	0.003	-2.607	0.009
ht	-0.024	0.011	-2.278	0.023
age	0.024	0.022	1.085	0.278
rank	-0.002	0.006	-0.279	0.780
rankpoints	0.000	0.000	3.963	0.000
ace	0.108	0.019	5.674	0.000
df	-0.140	0.037	-3.830	0.000

term	estimate	std.error	statistic	p.value
bpSaved	-0.077	0.029	-2.650	0.008
surfaceGrass	-0.580	0.280	-2.076	0.038
surfaceHard	-0.494	0.165	-3.000	0.003

```
full_w_interactions <- glm(status ~ minutes + ht + age + rank +
                rankpoints + ace + df + bpSaved + surface + surface * minutes +
                surface * ht + surface * age + surface * rank + surface * rankpoints +
                surface * ace + surface * df + surface * bpSaved,
                family=binomial,data=ten)
```

```
tidy(anova(full_model, full_w_interactions, test = "Chisq"))
```

```
## Warning in tidy.anova(anova(full_model, full_w_interactions, test =
## "Chisq")): The following column names in ANOVA output were not recognized
## or transformed: Resid..Df, Resid..Dev, Deviance
```

```
## Warning: Unknown or uninitialised column: 'term'.
```

```
## # A tibble: 2 x 5
##   Resid..Df Resid..Dev    df Deviance p.value
## *     <dbl>      <dbl> <dbl>    <dbl>   <dbl>
## 1       989      1176.    NA     NA    NA
## 2       973      1149.    16     26.6   0.0457
```

```
model.selected.interactions <- step(full_w_interactions,direction="backward")
```

```
## Start:  AIC=1202.97
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + surface * minutes + surface * ht + surface *
##     age + surface * rank + surface * rankpoints + surface * ace +
##     surface * df + surface * bpSaved
##
##                       Df Deviance    AIC
## - rank:surface         2   1149.4 1199.4
## - age:surface          2   1150.3 1200.3
## - bpSaved:surface      2   1150.5 1200.5
## - rankpoints:surface   2   1150.5 1200.5
## - minutes:surface      2   1150.6 1200.6
## <none>                     1149.0 1203.0
## - df:surface           2   1154.7 1204.7
## - ht:surface           2   1154.7 1204.7
## - ace:surface          2   1156.9 1206.9
##
## Step:  AIC=1199.39
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + minutes:surface + ht:surface + age:surface +
##     rankpoints:surface + ace:surface + df:surface + bpSaved:surface
##
##                       Df Deviance    AIC
## - bpSaved:surface      2   1150.7 1196.7
## - age:surface          2   1150.8 1196.8
## - minutes:surface      2   1150.9 1196.9
## - rankpoints:surface   2   1151.1 1197.1
## - rank                 1   1149.6 1197.6
```

```
## <none>                    1149.4 1199.4
## - ht:surface          2   1154.8 1200.8
## - df:surface          2   1154.8 1200.8
## - ace:surface         2   1157.6 1203.6
##
## Step:  AIC=1196.71
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + minutes:surface + ht:surface + age:surface +
##     rankpoints:surface + ace:surface + df:surface
##
##                       Df Deviance    AIC
## - minutes:surface     2   1151.6 1193.6
## - age:surface         2   1151.9 1193.9
## - rankpoints:surface  2   1152.4 1194.4
## - rank                1   1150.9 1194.9
## <none>                    1150.7 1196.7
## - ht:surface          2   1156.4 1198.4
## - ace:surface         2   1158.1 1200.1
## - bpSaved             1   1156.8 1200.8
## - df:surface          2   1159.6 1201.6
##
## Step:  AIC=1193.59
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + ht:surface + age:surface + rankpoints:surface +
##     ace:surface + df:surface
##
##                       Df Deviance    AIC
## - age:surface         2   1153.0 1191.0
## - rankpoints:surface  2   1153.7 1191.7
## - rank                1   1151.8 1191.8
## <none>                    1151.6 1193.6
## - ht:surface          2   1156.8 1194.8
## - ace:surface         2   1158.4 1196.4
## - bpSaved             1   1157.6 1197.6
## - df:surface          2   1160.4 1198.4
## - minutes             1   1159.3 1199.3
##
## Step:  AIC=1190.98
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + ht:surface + rankpoints:surface + ace:surface +
##     df:surface
##
##                       Df Deviance    AIC
## - rankpoints:surface  2   1154.9 1188.9
## - rank                1   1153.2 1189.2
## - age                 1   1153.5 1189.5
## <none>                    1153.0 1191.0
## - ht:surface          2   1158.7 1192.7
## - ace:surface         2   1160.6 1194.6
## - bpSaved             1   1159.2 1195.2
## - df:surface          2   1162.2 1196.2
## - minutes             1   1160.8 1196.8
##
## Step:  AIC=1188.92
```

```
## status ~ minutes + ht + age + rank + rankpoints + ace + df +
##     bpSaved + surface + ht:surface + ace:surface + df:surface
##
##               Df Deviance    AIC
## - rank         1   1155.1 1187.1
## - age          1   1155.5 1187.5
## <none>             1154.9 1188.9
## - ht:surface   2   1160.6 1190.6
## - ace:surface  2   1162.5 1192.5
## - bpSaved      1   1161.3 1193.3
## - minutes      1   1162.9 1194.9
## - df:surface   2   1164.9 1194.9
## - rankpoints   1   1173.8 1205.8
##
## Step:  AIC=1187.07
## status ~ minutes + ht + age + rankpoints + ace + df + bpSaved +
##     surface + ht:surface + ace:surface + df:surface
##
##               Df Deviance    AIC
## - age          1   1155.6 1185.6
## <none>             1155.1 1187.1
## - ht:surface   2   1160.7 1188.7
## - ace:surface  2   1162.7 1190.7
## - bpSaved      1   1161.7 1191.7
## - minutes      1   1162.9 1192.9
## - df:surface   2   1165.1 1193.1
## - rankpoints   1   1193.0 1223.0
##
## Step:  AIC=1185.62
## status ~ minutes + ht + rankpoints + ace + df + bpSaved + surface +
##     ht:surface + ace:surface + df:surface
##
##               Df Deviance    AIC
## <none>             1155.6 1185.6
## - ht:surface   2   1161.4 1187.4
## - ace:surface  2   1163.5 1189.5
## - bpSaved      1   1162.1 1190.1
## - minutes      1   1163.8 1191.8
## - df:surface   2   1165.8 1191.8
## - rankpoints   1   1193.6 1221.6
```

Linear Regression Assumptions

```
final.base.model <- model.selected.interactions
kable(tidy(final.base.model), format = "markdown", digits = 3)
```

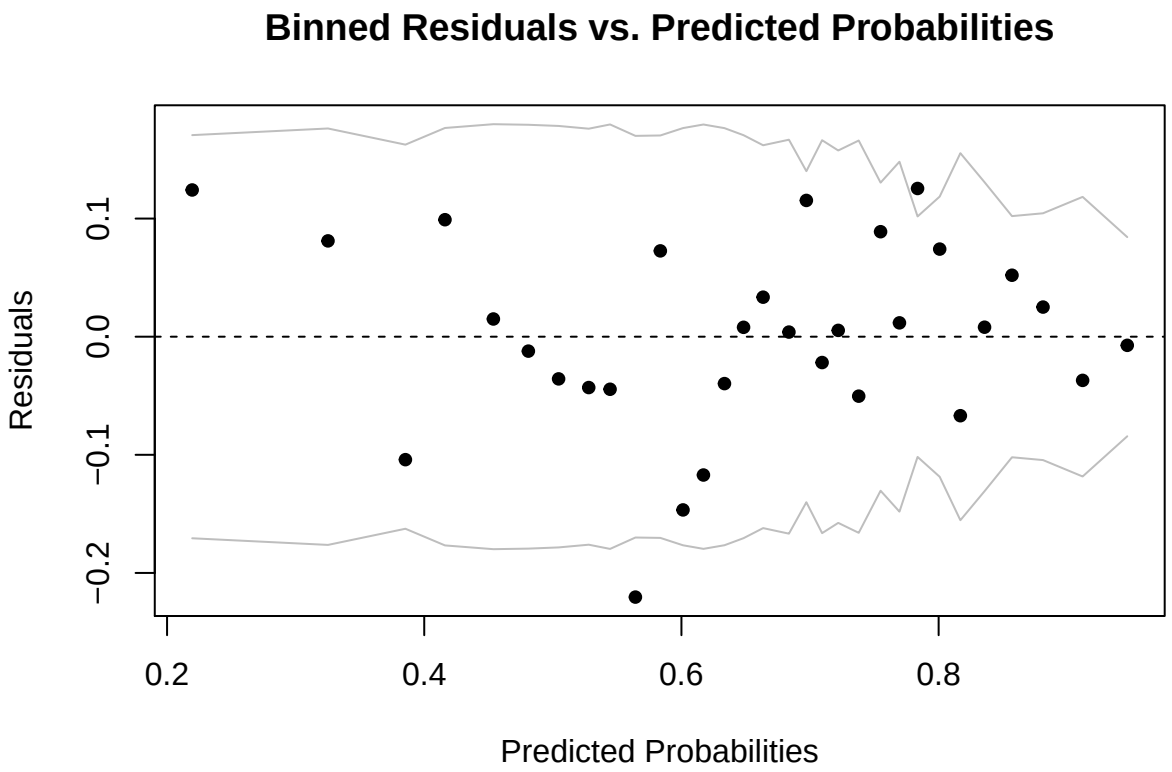
term	estimate	std.error	statistic	p.value
(Intercept)	10.392	3.518	2.954	0.003
minutes	-0.008	0.003	-2.844	0.004
ht	-0.048	0.019	-2.547	0.011
rankpoints	0.000	0.000	5.441	0.000

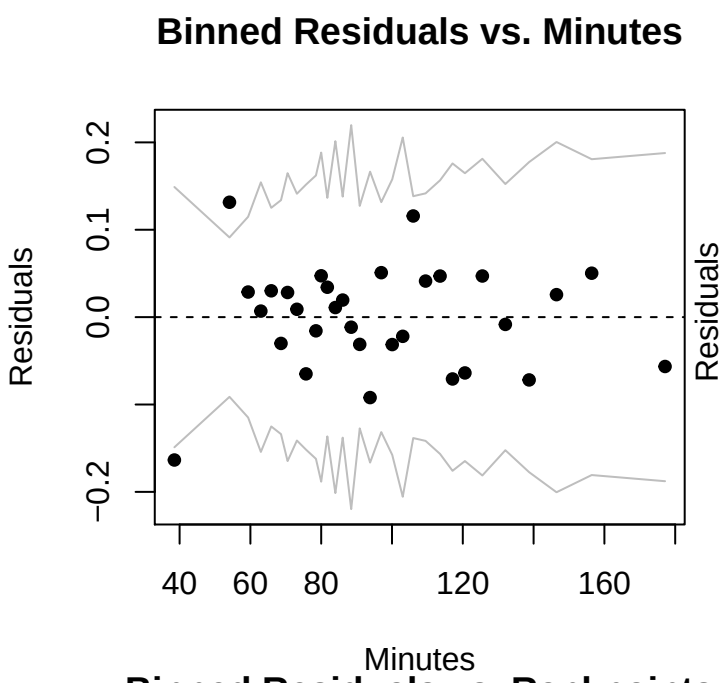
term	estimate	std.error	statistic	p.value
ace	0.110	0.040	2.764	0.006
df	-0.243	0.070	-3.474	0.001
bpSaved	-0.075	0.029	-2.548	0.011
surfaceGrass	5.285	8.299	0.637	0.524
surfaceHard	-8.048	4.294	-1.874	0.061
ht:surfaceGrass	-0.043	0.045	-0.940	0.347
ht:surfaceHard	0.040	0.023	1.710	0.087
ace:surfaceGrass	0.164	0.080	2.048	0.041
ace:surfaceHard	-0.021	0.045	-0.469	0.639
df:surfaceGrass	0.425	0.132	3.205	0.001
df:surfaceHard	0.107	0.082	1.297	0.195

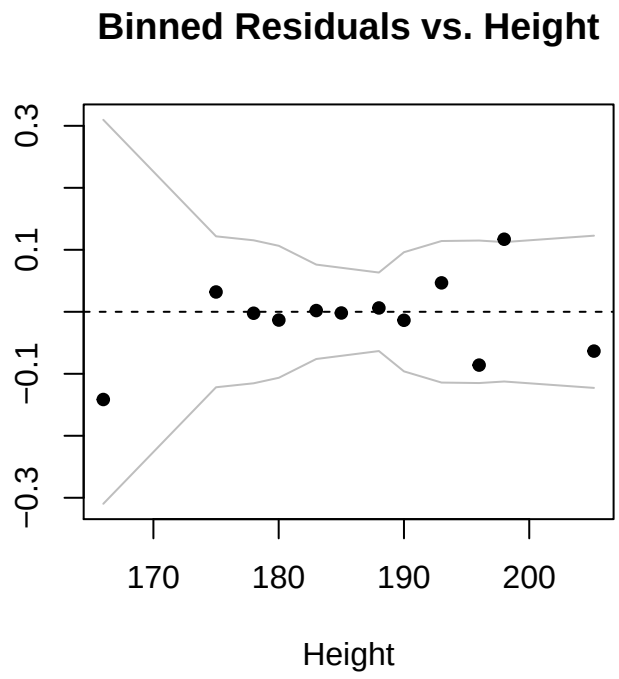
Model Assessment

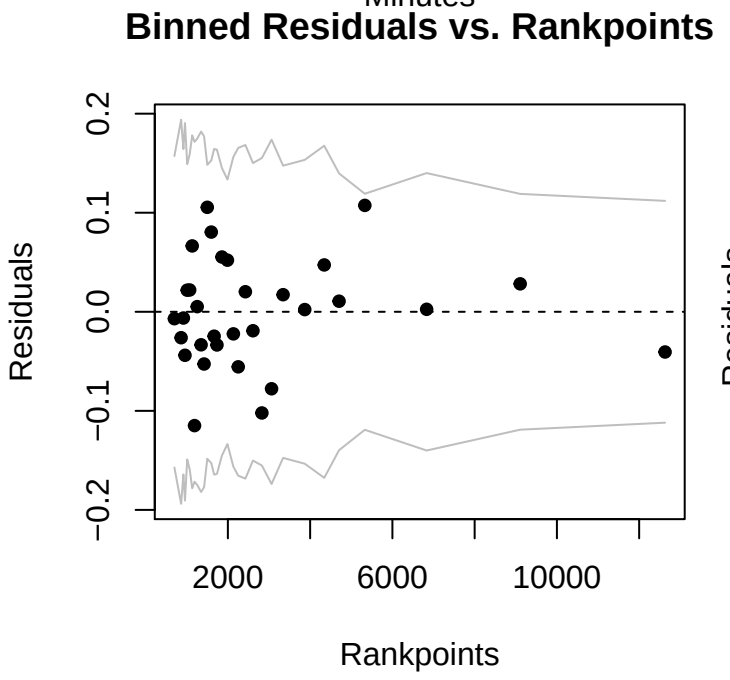
Binned Plots with Residuals vs Predicted

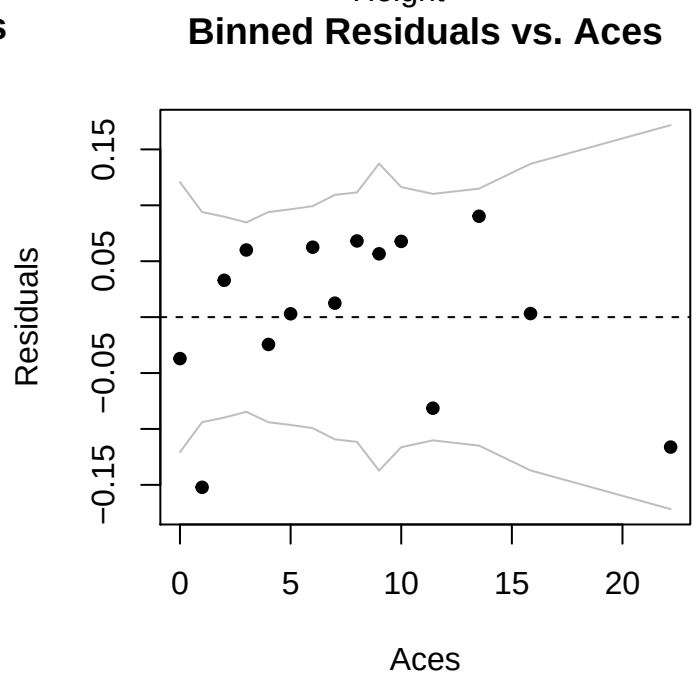
```
ten <- ten %>% mutate(Residuals = residuals.glm(final.base.model,type="response"),
                      Predicted = predict.glm(final.base.model,type="response"))

binnedplot(ten$Predicted, ten$Residuals,xlab="Predicted Probabilities",
           ylab="Residuals",main="Binned Residuals vs. Predicted Probabilities")
```

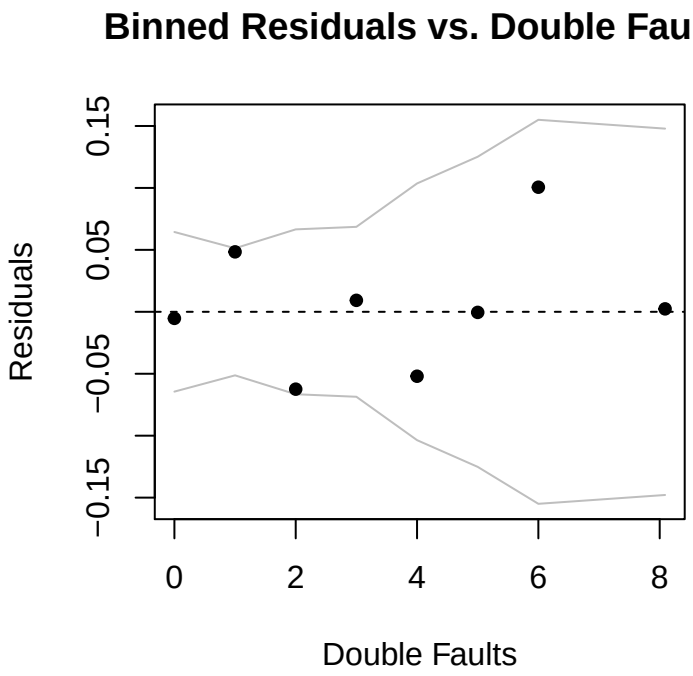


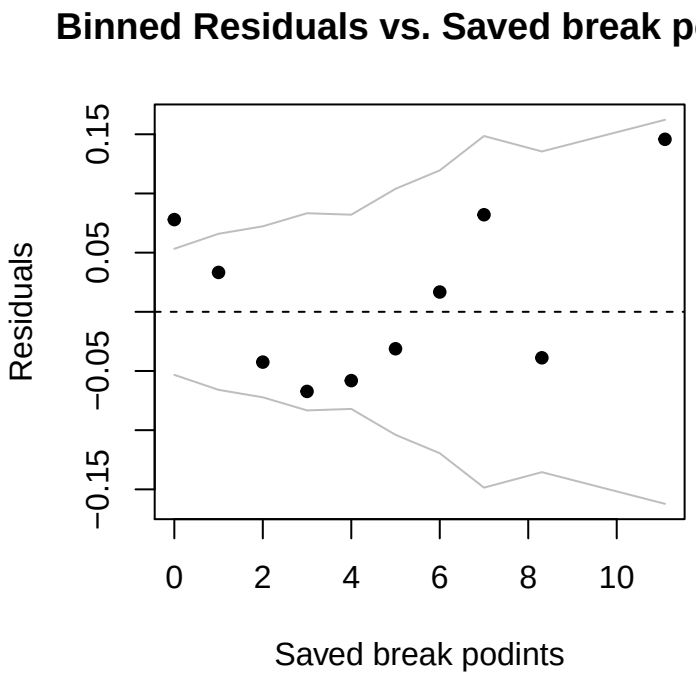

Binned Residuals vs. Minutes
Residuals
0.2
0.1
0.0
−0.2
40
60
80
120
160
Minutes


Binned Residuals vs. Height
Residuals
0.3
0.1
−0.1
−0.3
170
180
190
200
Height

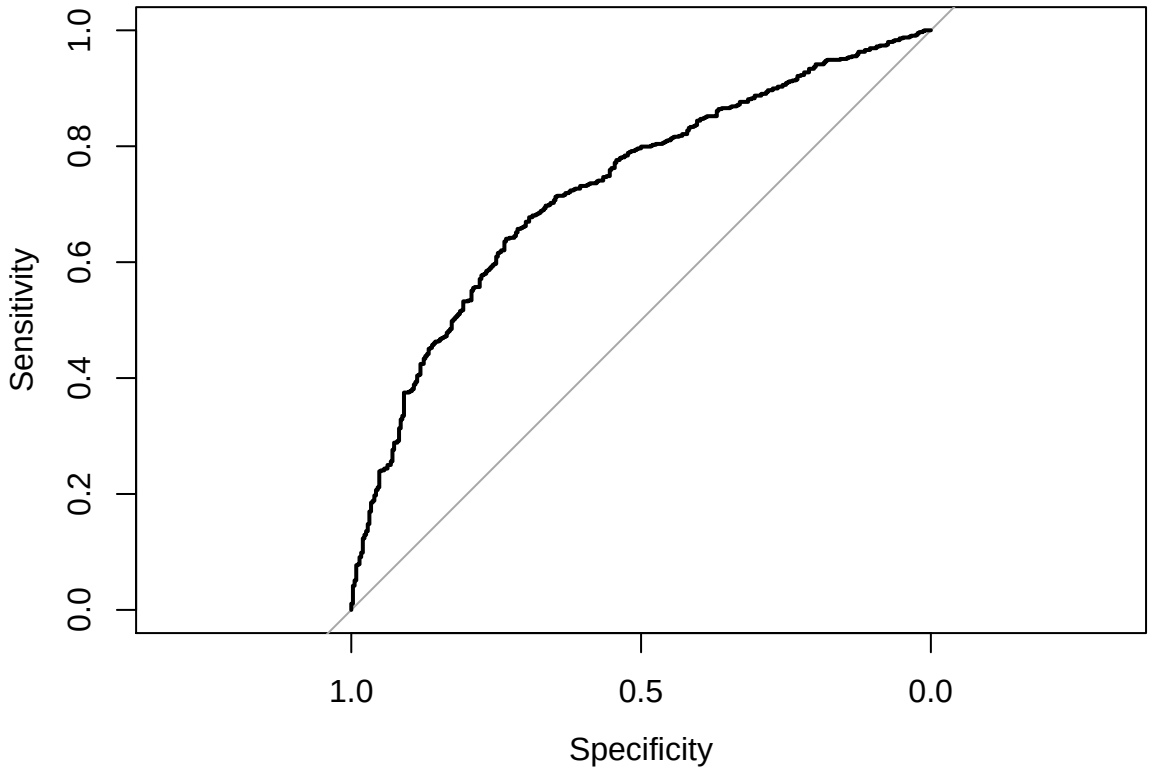

Binned Residuals vs. Rankpoints
Residuals
0.2
0.1
0.0
−0.1
−0.2
2000
6000
10000
Rankpoints


Binned Residuals vs. Aces
Residuals
0.15
0.05
−0.05
−0.15
0
5
10
15
20
Aces






```
ROC.ten <- roc(ten$status,ten$Predicted,plot=T)
```




```
ROC.ten$auc
```

```
## Area under the curve: 0.7268
```

```
threshold = 0.30
table(ten$status, ten$Predicted > threshold)
```

```
##
##     FALSE TRUE
##   0    26  326
##   1    13  635
```

```
(326 + 13)/(14+13+326+635)
```

```
## [1] 0.3431174
```

Linear Regression Assumptions: Revised

Unfortunately, one of our residuals plots has a non-linear relationship. Thus, we will remove `bpSaved` from the model.

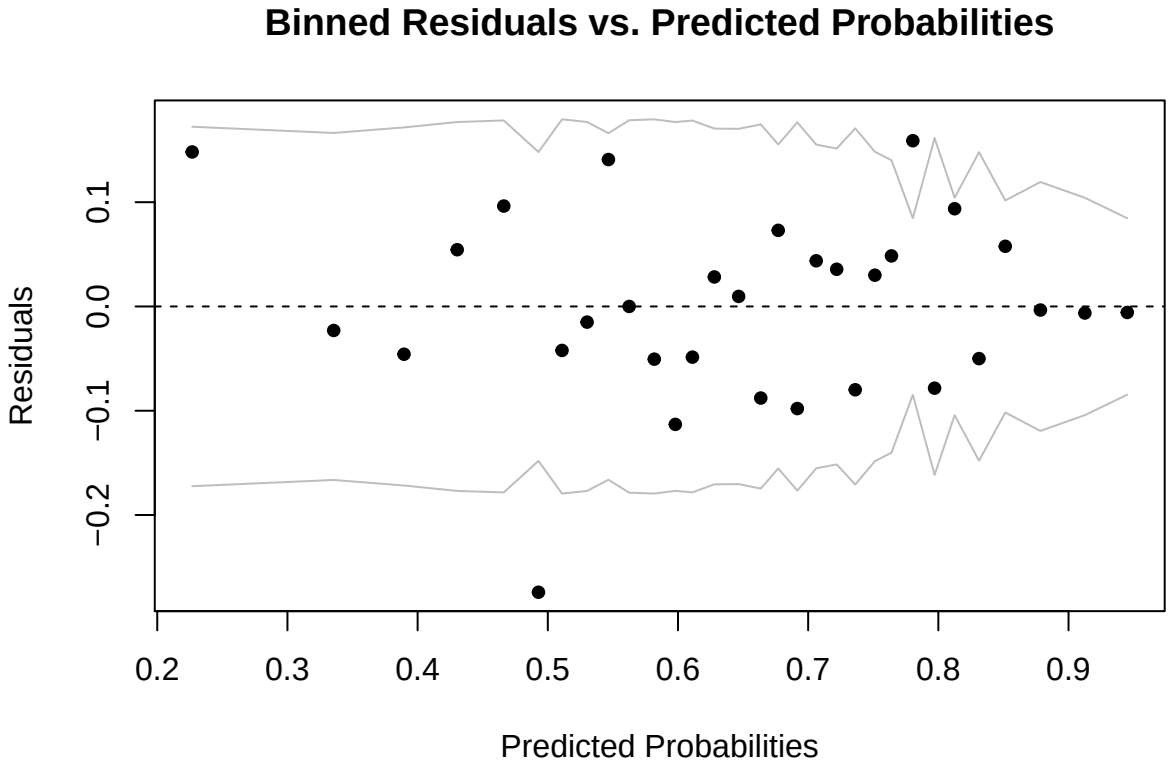
```
newten <- ten
final <- glm(status ~ minutes + ht + rankpoints + ace + df + surface + surface * ht + surface * ace + su
kable(tidy(final), format = "markdown", digits = 3)
```

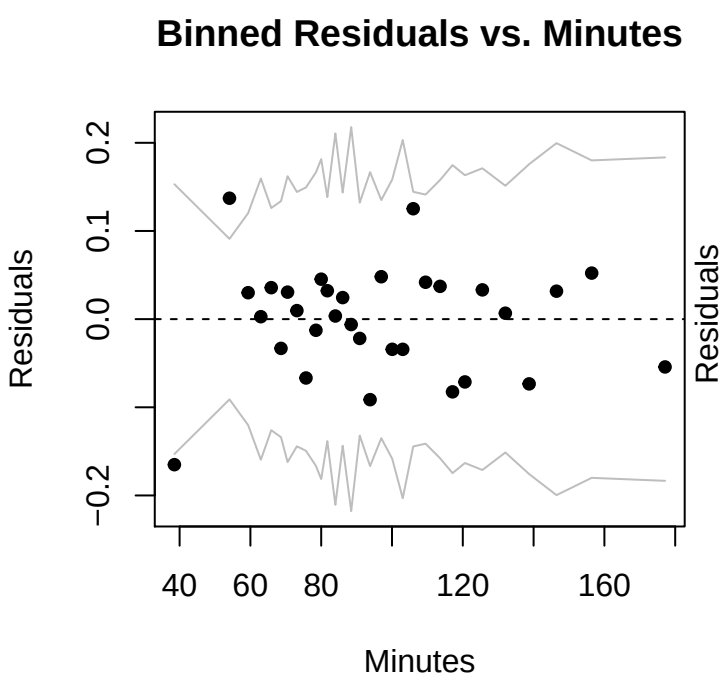
term	estimate	std.error	statistic	p.value
(Intercept)	10.684	3.495	3.057	0.002
minutes	-0.011	0.002	-4.258	0.000
ht	-0.050	0.019	-2.639	0.008
rankpoints	0.000	0.000	5.527	0.000
ace	0.112	0.040	2.830	0.005
df	-0.261	0.070	-3.746	0.000
surfaceGrass	4.682	8.233	0.569	0.570
surfaceHard	-8.871	4.261	-2.082	0.037
ht:surfaceGrass	-0.039	0.045	-0.869	0.385
ht:surfaceHard	0.044	0.023	1.914	0.056
ace:surfaceGrass	0.169	0.080	2.119	0.034
ace:surfaceHard	-0.022	0.045	-0.488	0.626
df:surfaceGrass	0.405	0.131	3.098	0.002
df:surfaceHard	0.114	0.082	1.385	0.166

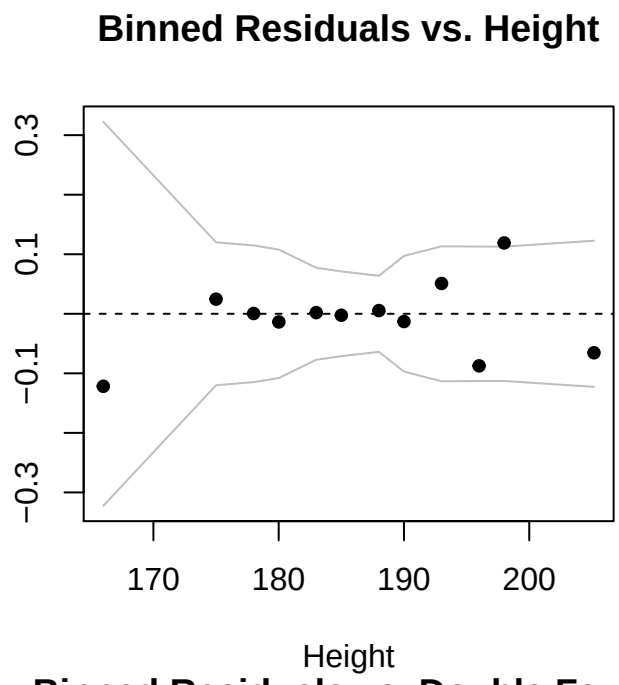
Model Assessment

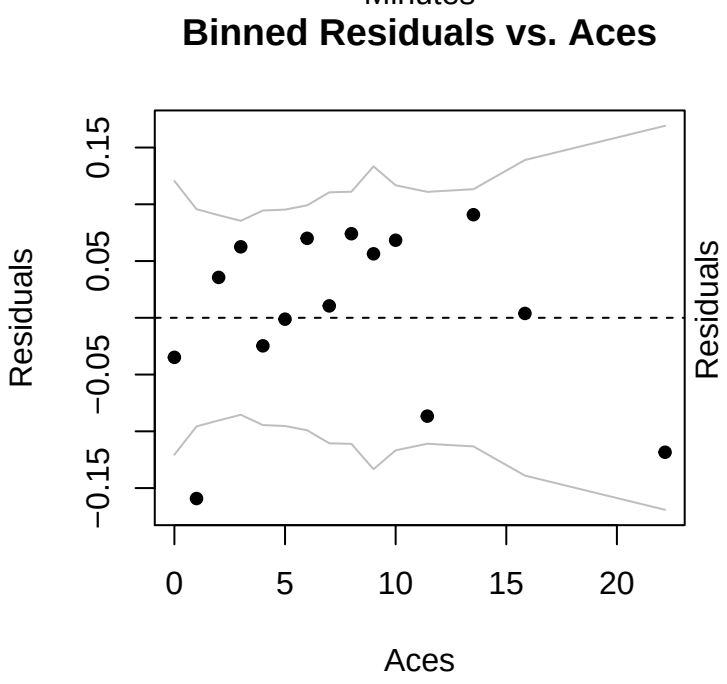
Binned Plots with Residuals vs Predicted

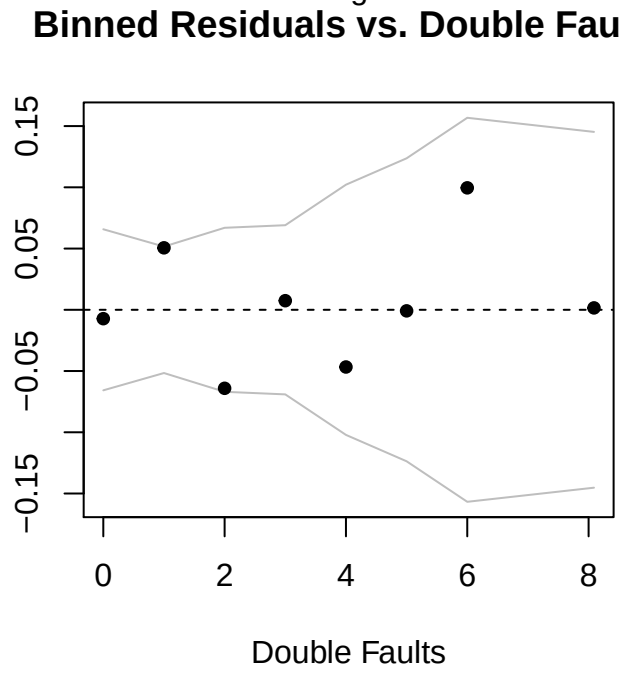
```
newten <- newten %>% mutate(Residuals = residuals.glm(final,type="response"),
                            Predicted = predict.glm(final,type="response"))

binnedplot(newten$Predicted, newten$Residuals,xlab="Predicted Probabilities",
           ylab="Residuals",main="Binned Residuals vs. Predicted Probabilities")
```

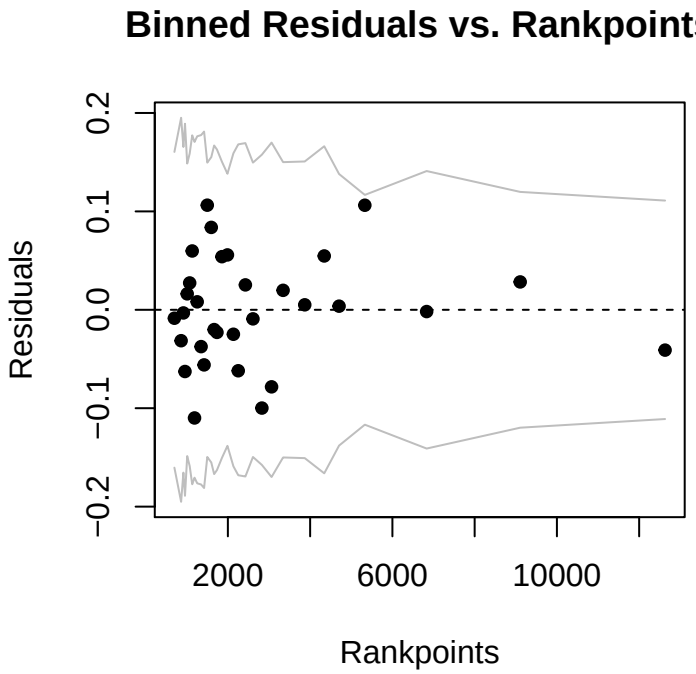

Binned Residuals vs. Predicted Probabilities
Residuals
0.1
0.0
−0.1
−0.2
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
Predicted Probabilities


Binned Residuals vs. Minutes
Residuals
0.2
0.1
0.0
−0.2
40
60
80
120
160
Minutes

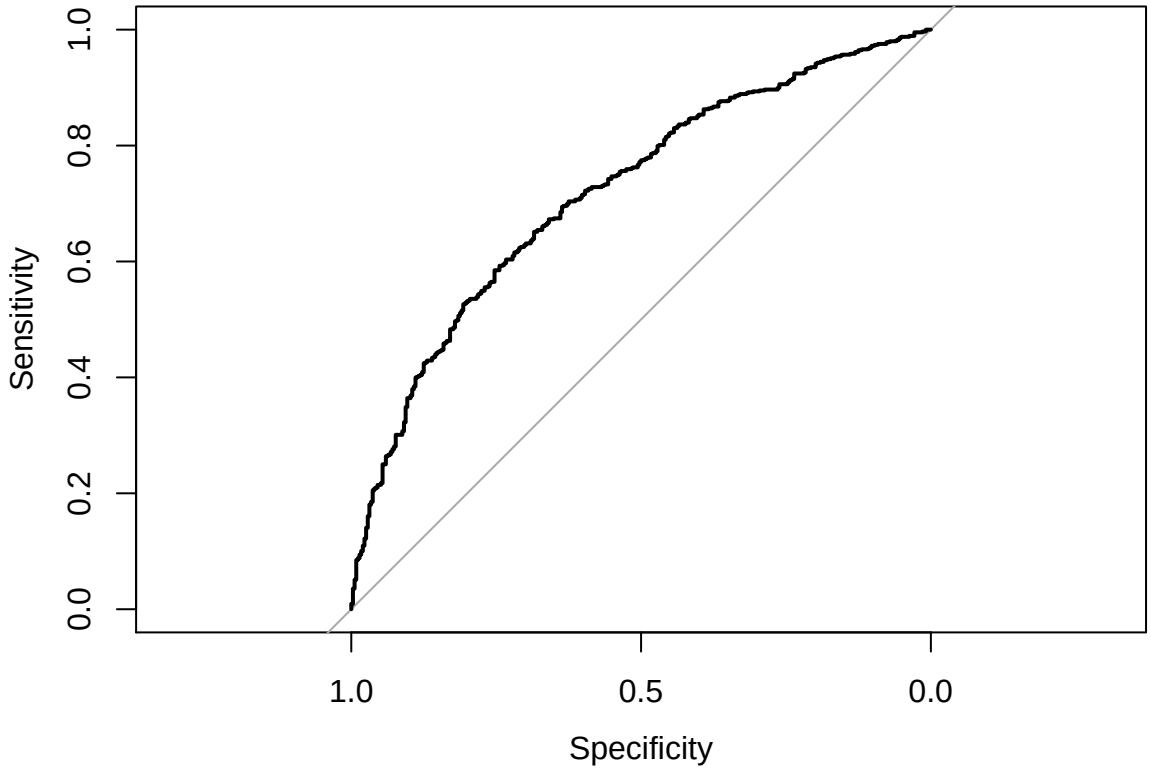

Binned Residuals vs. Height
Residuals
0.3
0.1
−0.1
−0.3
170
180
190
200
Height


Binned Residuals vs. Aces
Residuals
0.15
0.05
−0.05
−0.15
0
5
10
15
20
Aces


Binned Residuals vs. Double Faults
Residuals
0.15
0.05
−0.05
−0.15
0
2
4
6
8
Double Faults




```
ROC.newten <- roc(newten$status,newten$Predicted,plot=T)
```




```
ROC.newten$auc
```

```
## Area under the curve: 0.7198
```

```
threshold = 0.30
table(newten$status, newten$Predicted > threshold)
```

```
##
##     FALSE TRUE
##   0    20  332
##   1    12  636
```

```
(326 + 13)/(14+13+326+635)
```

```
## [1] 0.3431174
```

Prediction

Test cases:

We will look at a match that was played the Monte Carlo Masters in 2014 between Roger Federer and Novak Djokovic. We want to see the percentage that a player will win the match.

```
ten %>%
  filter(tourney_name == "Monte Carlo Masters",
         name == "Roger Federer" |  name == "Novak Djokovic",
         tourney_id == "2014-410",
         loser_name == "Novak Djokovic")
```

```
## # A tibble: 1 x 68
##   tourney_id tourney_name surface draw_size tourney_level tourney_date
##   <chr>      <chr>        <chr>       <int> <chr>               <int>
## 1 2014-410   Monte Carlo~ Clay           56 M                20140413
## # ... with 62 more variables: match_num <int>, winner_id <int>,
## #   winner_seed <int>, winner_entry <chr>, winner_name <chr>,
## #   winner_hand <chr>, winner_ht <int>, winner_ioc <chr>,
## #   winner_age <dbl>, winner_rank <int>, winner_rank_points <int>,
## #   loser_id <int>, loser_seed <int>, loser_entry <chr>, loser_name <chr>,
## #   loser_hand <chr>, loser_ht <int>, loser_ioc <chr>, loser_age <dbl>,
## #   loser_rank <int>, loser_rank_points <int>, score <chr>, best_of <int>,
## #   round <chr>, minutes <int>, w_ace <int>, w_df <int>, w_svpt <int>,
## #   w_1stIn <int>, w_1stWon <int>, w_2ndWon <int>, w_SvGms <int>,
## #   w_bpSaved <int>, w_bpFaced <int>, l_ace <int>, l_df <int>,
## #   l_svpt <int>, l_1stIn <int>, l_1stWon <int>, l_2ndWon <int>,
## #   l_SvGms <int>, l_bpSaved <int>, l_bpFaced <int>, seed <int>,
## #   name <chr>, hand <chr>, ht <int>, age <dbl>, rank <int>,
## #   rankpoints <int>, ace <int>, df <int>, svpt <int>, firsstIn <int>,
## #   firsttWon <int>, secondndWon <int>, SvGms <int>, bpSaved <int>,
## #   bpFaced <int>, status <fct>, Residuals <dbl>, Predicted <dbl>
```

```
fed <- data.frame(minutes = 75, ht = 185, rankpoints = 5355 , ace = 3 , df = 1, surface = "Hard")
djo1 <- data.frame(minutes = 75, ht = 188, rankpoints = 11680, ace = 2, df = 0, surface = "Hard")
predict.glm(final, newdata=fed, type="response")
```

```
##         1
## 0.7709045
```

```
predict.glm(final, newdata=djo1, type="response")
```

```
##         1
## 0.9252598
```

We see that the model predicts Roger Federer has a 77.09% chance of winning the match, while Novak Djokovic has a 92.52% chance of winning the match. This is mainly Djokovic's rank points were almost double that of Federer's. However, Federer won the match.

The the next match we will look at was played at the China Open in 2013 and was between Novak Djokovic and Rafael Nadal.

```
tennis %>%
  filter(name == "Novak Djokovic" | name == "Rafael Nadal",
         tourney_name == "Beijing",
         loser_name == "Rafael Nadal",
         tourney_id == "2013-747")
```

```
## # A tibble: 2 x 66
##   tourney_id tourney_name surface draw_size tourney_level tourney_date
##   <chr>      <chr>        <chr>       <int> <chr>                <int>
## 1 2013-747   Beijing      Hard           32 A                 20130930
## 2 2013-747   Beijing      Hard           32 A                 20130930
## # ... with 60 more variables: match_num <int>, winner_id <int>,
## #   winner_seed <int>, winner_entry <chr>, winner_name <chr>,
## #   winner_hand <chr>, winner_ht <int>, winner_ioc <chr>,
## #   winner_age <dbl>, winner_rank <int>, winner_rank_points <int>,
## #   loser_id <int>, loser_seed <int>, loser_entry <chr>, loser_name <chr>,
## #   loser_hand <chr>, loser_ht <int>, loser_ioc <chr>, loser_age <dbl>,
## #   loser_rank <int>, loser_rank_points <int>, score <chr>, best_of <int>,
## #   round <chr>, minutes <int>, w_ace <int>, w_df <int>, w_svpt <int>,
## #   w_1stIn <int>, w_1stWon <int>, w_2ndWon <int>, w_SvGms <int>,
## #   w_bpSaved <int>, w_bpFaced <int>, l_ace <int>, l_df <int>,
## #   l_svpt <int>, l_1stIn <int>, l_1stWon <int>, l_2ndWon <int>,
## #   l_SvGms <int>, l_bpSaved <int>, l_bpFaced <int>, seed <int>,
## #   name <chr>, hand <chr>, ht <int>, age <dbl>, rank <int>,
## #   rankpoints <int>, ace <int>, df <int>, svpt <int>, firsstIn <int>,
## #   firsttWon <int>, secondndWon <int>, SvGms <int>, bpSaved <int>,
## #   bpFaced <int>, status <dbl>
```

```
djo2 <- data.frame(minutes = 87, ht = 188, rankpoints = 11120, ace = 5, df = 1, surface = "Hard")
nadal <- data.frame(minutes = 87, ht = 185, rankpoints = 10860, ace = 2, df = 2, surface = "Hard")
predict.glm(final, newdata=djo2, type="response")
```

```
##         1
## 0.9169758
```

```
predict.glm(final, newdata=nadal, type="response")
```

```
##         1
## 0.8751727
```

We see that the model predicts Novak Djokovic has a 91.7% chance of winning the match, while Rafael Nadal has a 87.51% chance of winning the match. Djokovic won the match.

Test Case with Similar Rank Points

Now, we'll look at a match where both players had similar rank points. The match we will look at is between Jo Wilfried Tsonga and Michael Llodra, and was played at the Queen's Club Championships in 2011.

```
ten %>%
  filter(winner_rank_points > 1000 & winner_rank_points < 2000 & loser_rank_points > 1000 & loser_rank_
```

```
        tourney_id == "2011-311",
        tourney_name == "Queen's Club",
        winner_name == "Jo Wilfried Tsonga"
        )
```

```
## # A tibble: 1 x 68
##   tourney_id tourney_name surface draw_size tourney_level tourney_date
##   <chr>      <chr>        <chr>       <int> <chr>                <int>
## 1 2011-311   Queen's Club Grass          56 A                 20110606
## # ... with 62 more variables: match_num <int>, winner_id <int>,
## #   winner_seed <int>, winner_entry <chr>, winner_name <chr>,
## #   winner_hand <chr>, winner_ht <int>, winner_ioc <chr>,
## #   winner_age <dbl>, winner_rank <int>, winner_rank_points <int>,
## #   loser_id <int>, loser_seed <int>, loser_entry <chr>, loser_name <chr>,
## #   loser_hand <chr>, loser_ht <int>, loser_ioc <chr>, loser_age <dbl>,
## #   loser_rank <int>, loser_rank_points <int>, score <chr>, best_of <int>,
## #   round <chr>, minutes <int>, w_ace <int>, w_df <int>, w_svpt <int>,
## #   w_1stIn <int>, w_1stWon <int>, w_2ndWon <int>, w_SvGms <int>,
## #   w_bpSaved <int>, w_bpFaced <int>, l_ace <int>, l_df <int>,
## #   l_svpt <int>, l_1stIn <int>, l_1stWon <int>, l_2ndWon <int>,
## #   l_SvGms <int>, l_bpSaved <int>, l_bpFaced <int>, seed <int>,
## #   name <chr>, hand <chr>, ht <int>, age <dbl>, rank <int>,
## #   rankpoints <int>, ace <int>, df <int>, svpt <int>, firsstIn <int>,
## #   firsttWon <int>, secondndWon <int>, SvGms <int>, bpSaved <int>,
## #   bpFaced <int>, status <fct>, Residuals <dbl>, Predicted <dbl>
```

```
jo <- data.frame(minutes = 23, ht = 188, rankpoints = 1480, ace = 2, df = 1, surface = "Grass")
llodra <- data.frame(minutes = 23, ht = 190, rankpoints = 1400, ace = 0, df = 0, surface = "Grass")
predict.glm(final, newdata=jo, type="response")
```

```
##         1
## 0.3646036
```

```
predict.glm(final, newdata=llodra, type="response")
```

```
##         1
## 0.1891476
```

We see that the model predicts Jo Wilfried Tsonga has a 36.46% of chance of winning the match, while Michael Llodra has a 18.91% chance of winning the match. These percentages are lower because the players' rankpoints are significantly lower than those of Nadal's and Djokovic's.

Conclusion