Final Project

Austin Burns

4/23/2023

```
# required libraries
library(dplyr)
library(corrplot)
library(ROCR)
library(ResourceSelection)
library(LaplacesDemon)
library(corrplot)
library(ggplot2)
library(GGally)
```

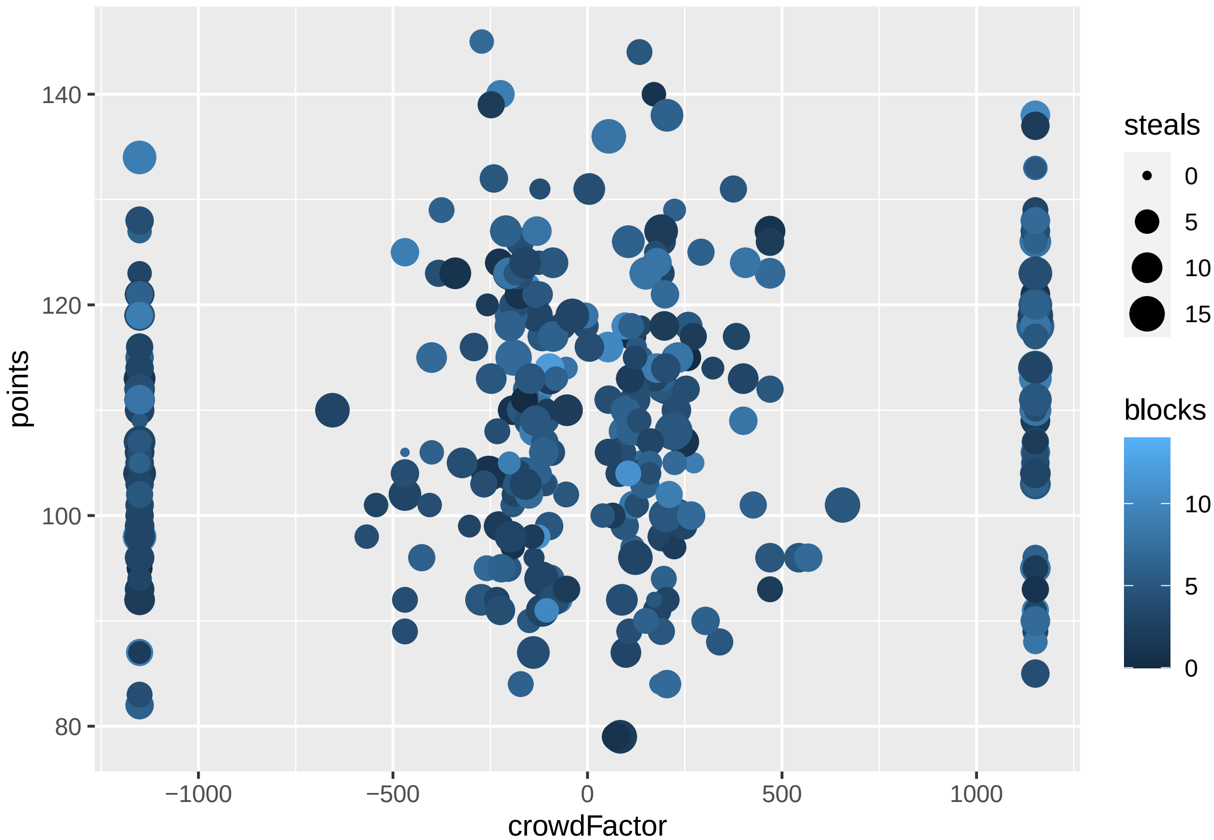
Project Functions

```
# function for crowd factor (unused in final)
cfGen1 <- function(cf) {
  y <- exp(logit(cf)) / (1 + exp(logit(cf)))
  return(y)
}
# function for crowd factor
cfGen2 <- function(cf, crowd) {
  if (crowd == "home") {
    y <- 100 * log((cf)/(1-cf))
    return(y)
  }
  else {
    y <- 100 * log((cf)/(1-cf))
    return(-y)
  }

  return(y)
}
# read csv for games statistics
readCSV500 <- function() {
  games <- read.csv("tidy500_conf_true.csv")
  attach(games)
  games <- games %>%
    filter(attendance != "no_info")
  games <- games %>%
    mutate(capacity = strtoi(capacity))
  detach(games)
  return(games)
}
```

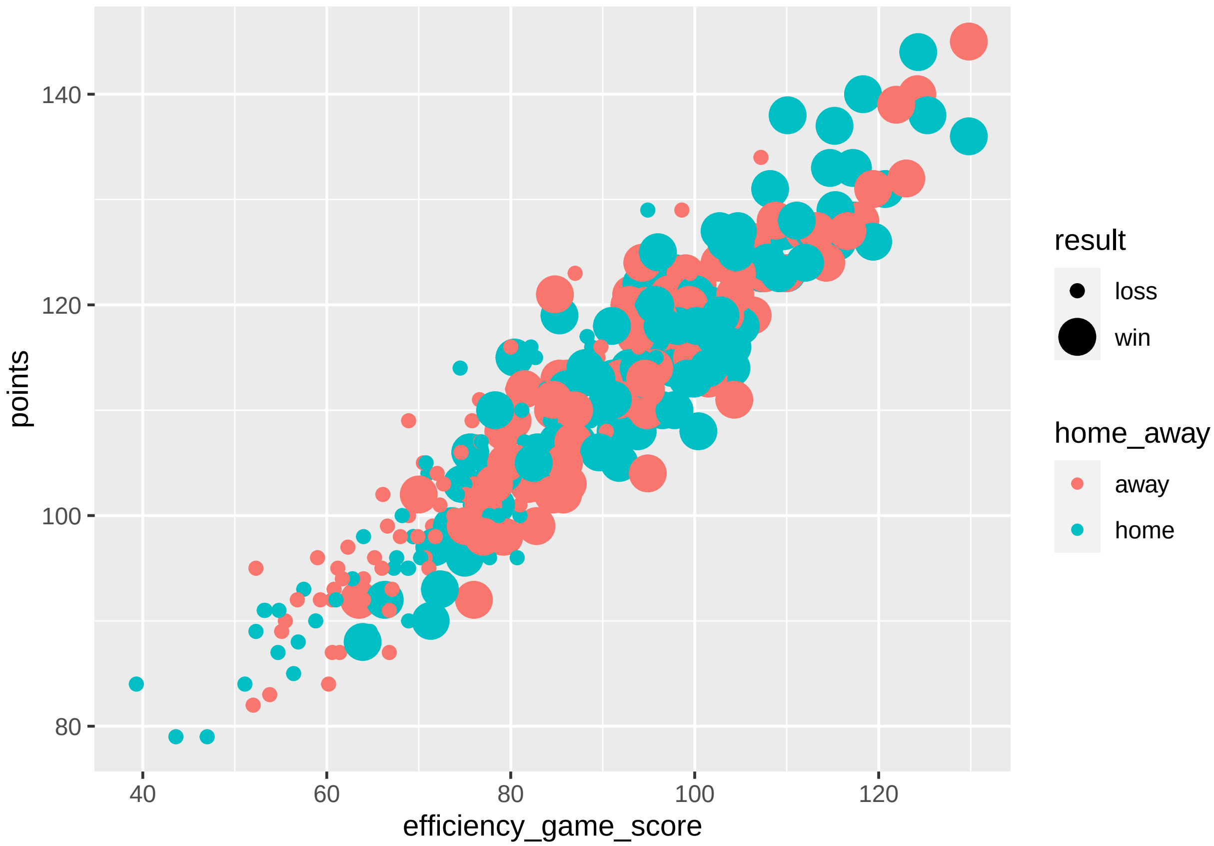
```r
# function to read tidy500_conf_true.csv, applies crowFactor transformation
readCSVConf <- function() {
  dat <- read.csv("tidy500_conf_true.csv")
  attach(dat)
  dat <- dat %>%
    filter(attendance != "no_info")
  dat <- dat %>%
    mutate(capacity = strtoi(capacity))

  crowdFactor <- as.numeric(attendance) / capacity
  subSet5 <- data.frame(result, crowdFactor, home_away, fast_break_made, blocks)
  rm(crowdFactor)
  # keep crowd factor on open interval (0, 1)
  subSet5 <- subSet5 %>%
    mutate(crowdFactor = if_else(crowdFactor > 1 |
                                 crowdFactor == 1, .99999, crowdFactor))
  subSet5 <- subSet5 %>%
    mutate(crowdFactor = if_else(crowdFactor == 0, .00001, crowdFactor))

  detach(dat)
  attach(subSet5)
  # cleaning up subSet3 and adding croud factor transformation (improved p-val)
  subSet5 <- na.omit(subSet5)
  subSet5 <- subSet5 %>%
    mutate(home_away = if_else(home_away == "home", 1, 0))
  # apply transformation to crowdFactor
  subSet5 <- subSet5 %>%
    mutate(crowdFactor = if_else(home_away == 1, cfGen2(crowdFactor, "home"),
                                cfGen2(crowdFactor, "away")))
  subSet5 <- subSet5 %>%
    mutate(result = if_else(result == "win", 1, 0))
  detach(subSet5)
  return(subSet5)
}
```

A) Initial Testing

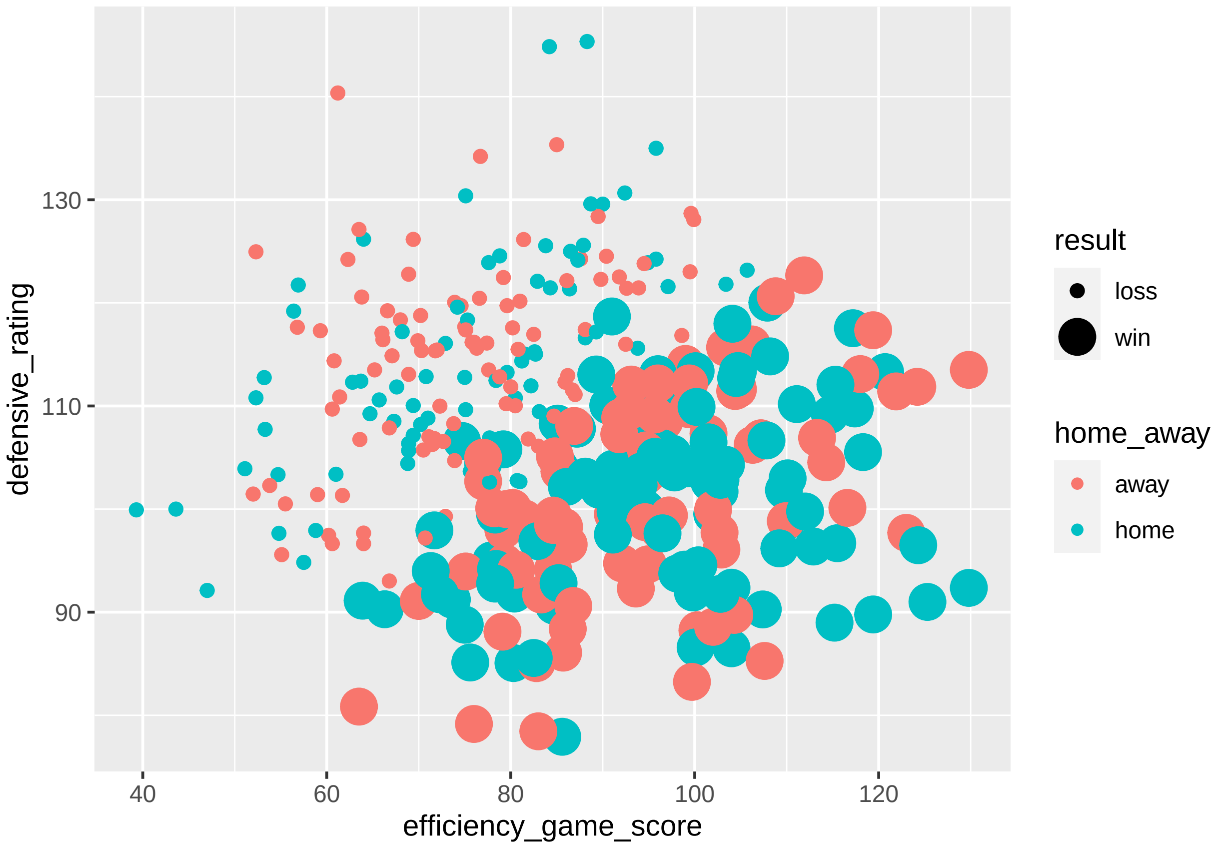
```r
games <- readCSV500()
subSet5 <- readCSVConf()
# first look at some of the data
attach(games)
ggplot(subSet5, aes(x = crowdFactor, y = points, color = blocks,
                    size = steals)) +
  geom_point()
```




```r
ggplot(games, aes(x = efficiency_game_score, y = points,
                  color = home_away, size = result)) +
  geom_point()
```




```r
ggplot(games, aes(x = efficiency_game_score, y = defensive_rating,
                  color = home_away, size = result)) +
  geom_point()
```




```
detach(games)
```

a) Comparing Factors

```
# make a paired t-test comparison between...
# home and away points
home <- games %>%
  filter(home_away == "home") %>%
  pull(points)
away <- games %>%
  filter(home_away == "away") %>%
  pull(points)

t.test(home, away, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  home and away
## t = 1.4377, df = 177, p-value = 0.1523
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.6154394  3.9188102
## sample estimates:
## mean of the differences
##                1.651685
```

```r
rm(home)
rm(away)
```

We see an insignificant p-value of 0.15, and we fail to reject the null hypothesis that the difference in means is zero. We don't have enough evidence to say that teams have different mean score between home and away games. This will be of interest to us later on.

```r
# make a paired t-test comparison between...
# home and away efficiency game score
home <- games %>%
  filter(home_away == "home") %>%
  pull(efficiency_game_score)
away <- games %>%
  filter(home_away == "away") %>%
  pull(efficiency_game_score)

t.test(home, away, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  home and away
## t = 1.4043, df = 177, p-value = 0.162
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.074944  6.379439
## sample estimates:
## mean of the differences
##                2.652247
```

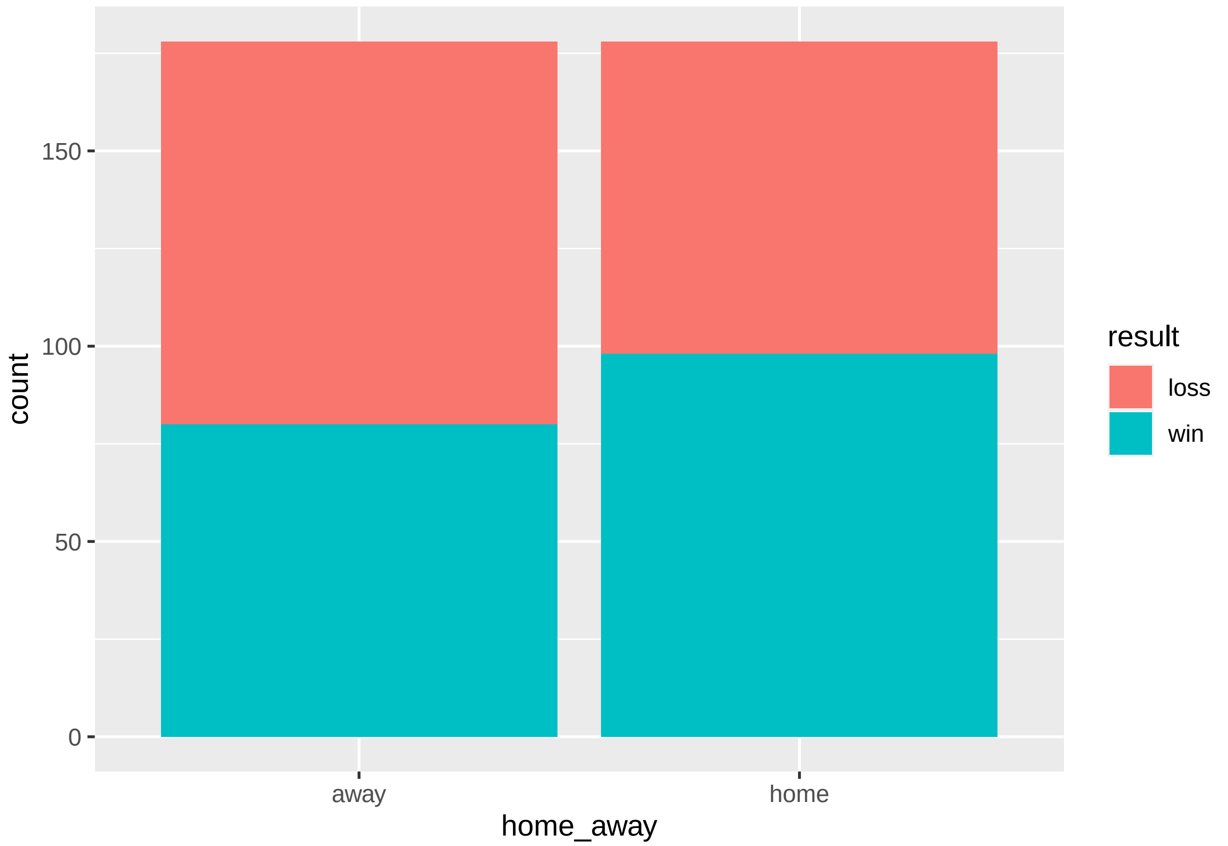
```r
rm(home)
rm(away)
```

We see $p = 0.162$ *, we cannot reject the hypothesis that the difference between mean efficiency_game_score is 0 between home and away teams.*

```r
attach(games)
table(result, home_away)
```

```
##       home_away
## result away home
##   loss   98   80
##   win    80   98
```

```r
ggplot(games) +
  aes(x = home_away, fill = result) +
  geom_bar()
```




```
chiTest <- chisq.test(table(result, home_away))
chiTest
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(result, home_away)
## X-squared = 3.2472, df = 1, p-value = 0.07155
```

```
detach(games)
```

Our Chi-squared test for independence gives us a p-value $p = 0.0739$ *: Our null hypothesis is that there is not a significant relationship between home_away and the result of the game. We later improve on this by creating the crowdFactor variable.*

B) Obvious Models

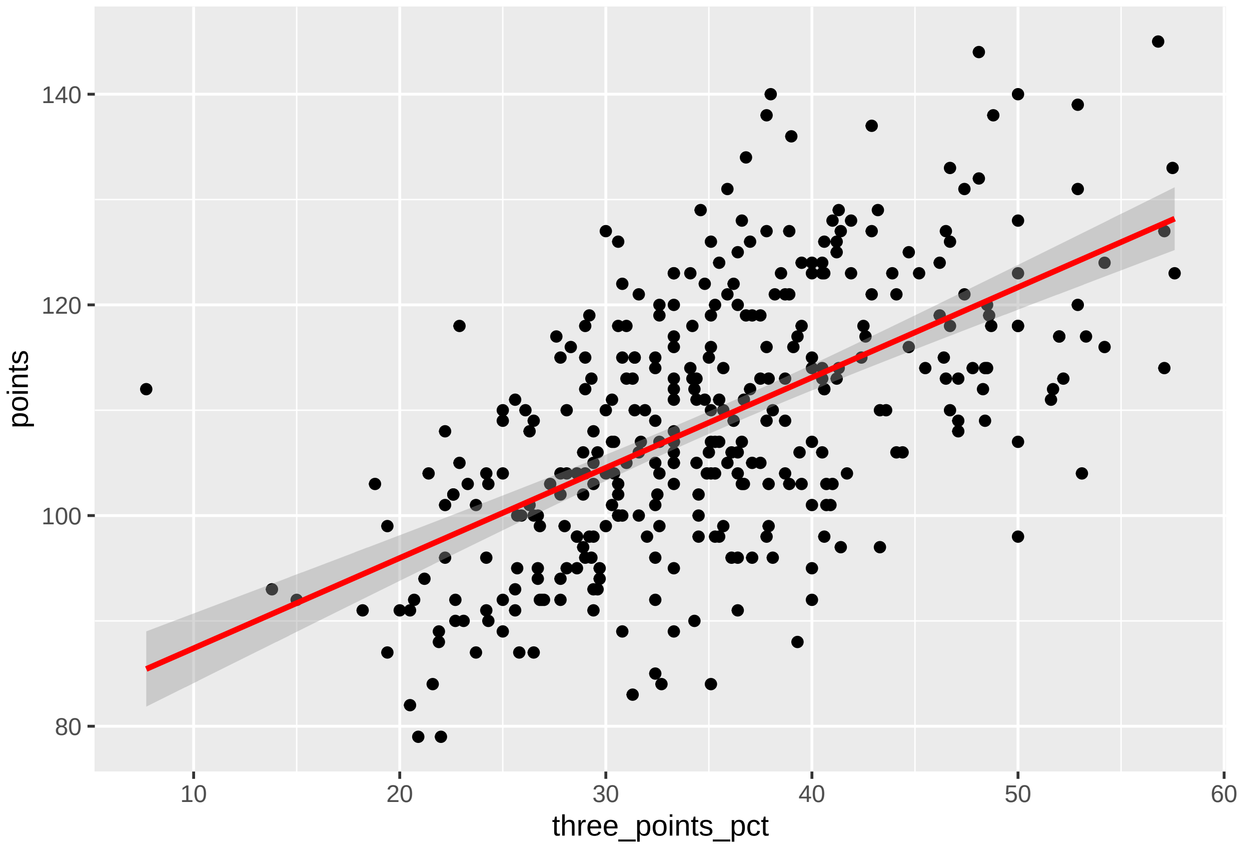
```
attach(games)
subSet1 <- data.frame(result, attendance, points, three_points_pct,
                      two_points_pct, free_throws_pct, true_shooting_att)
detach(games)
attach(subSet1)
subSet1 <- subSet1 %>%
  mutate(attendance = as.numeric(attendance))
subSet1 <- subSet1 %>%
  mutate(result = if_else(result == "win", 1, 0))
```

```r
# three point percentage model
threePtPcModel <- lm(points ~ three_points_pct)
summary(threePtPcModel)
```

```
##
## Call:
## lm(formula = points ~ three_points_pct)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.9078  -7.5090  -0.7049   7.4963  28.6076
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      78.83611    2.28547   34.49   <2e-16 ***
## three_points_pct  0.85674    0.06311   13.58   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.2 on 354 degrees of freedom
## Multiple R-squared:  0.3424, Adjusted R-squared:  0.3405
## F-statistic: 184.3 on 1 and 354 DF,  p-value: < 2.2e-16
```

```r
ggplot(subSet1, aes(x = three_points_pct, y = points)) +
  geom_point() +
  stat_smooth(method = "lm", col = "red")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

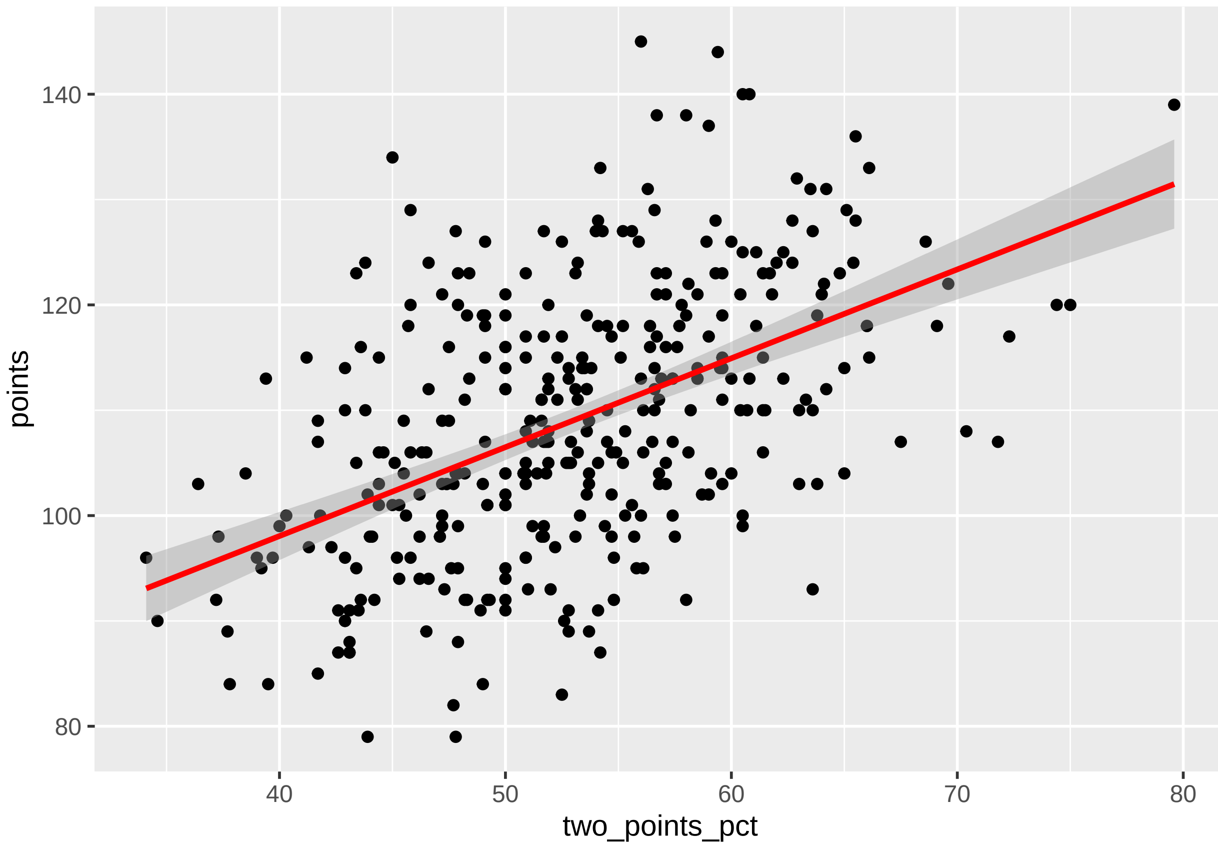



```r
# two point percentage model
twoPtPcModel <- lm(points ~ two_points_pct)
summary(twoPtPcModel)
```

```
##
## Call:
## lm(formula = points ~ two_points_pct)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -25.639  -7.498  -0.850   7.345  33.442
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    64.30958    4.15921   15.46   <2e-16 ***
## two_points_pct  0.84372    0.07779   10.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.9 on 354 degrees of freedom
## Multiple R-squared:  0.2494, Adjusted R-squared:  0.2473
## F-statistic: 117.6 on 1 and 354 DF,  p-value: < 2.2e-16
```

```r
ggplot(subSet1, aes(x = two_points_pct, y = points)) +
  geom_point() +
  stat_smooth(method = "lm", col = "red")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



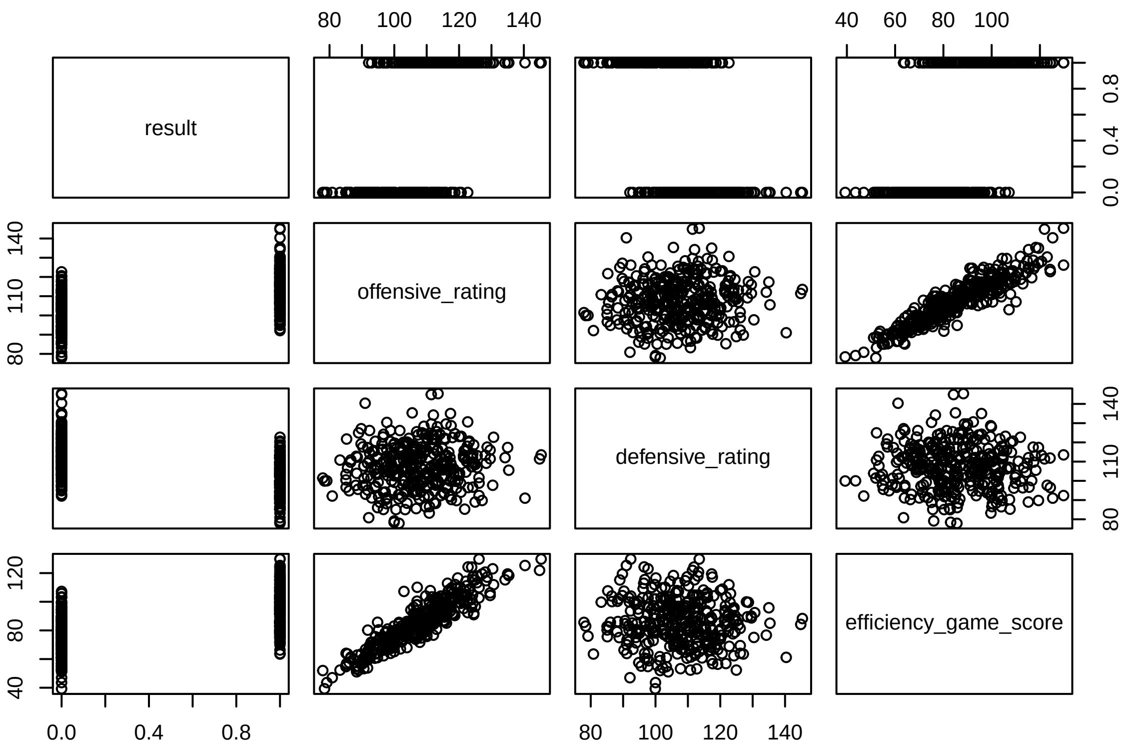

```r
naiveFit <- lm(points ~ three_points_pct + two_points_pct + free_throws_pct)
summary(naiveFit)
```

```
##
## Call:
## lm(formula = points ~ three_points_pct + two_points_pct + free_throws_pct)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20.0179  -5.4596  -0.6162   4.6531  30.4798
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      23.20105    4.84641   4.787 2.49e-06 ***
## three_points_pct  0.80517    0.05062  15.905  < 2e-16 ***
## two_points_pct    0.78128    0.05839  13.379  < 2e-16 ***
## free_throws_pct   0.21017    0.04435   4.738 3.13e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.164 on 352 degrees of freedom
## Multiple R-squared:  0.5814, Adjusted R-squared:  0.5778
## F-statistic:   163 on 3 and 352 DF,  p-value: < 2.2e-16
```

```r
detach(subSet1)
```

```r
attach(games)
# can't really use offensive & defensive together
```

```r
subSet2 <- data.frame(result, offensive_rating, defensive_rating,
                      efficiency_game_score)
subSet2 <- subSet2 %>%
  mutate(result = if_else(result == "win", 1, 0))

detach(games)
attach(subSet2)
pairs(subSet2)
```

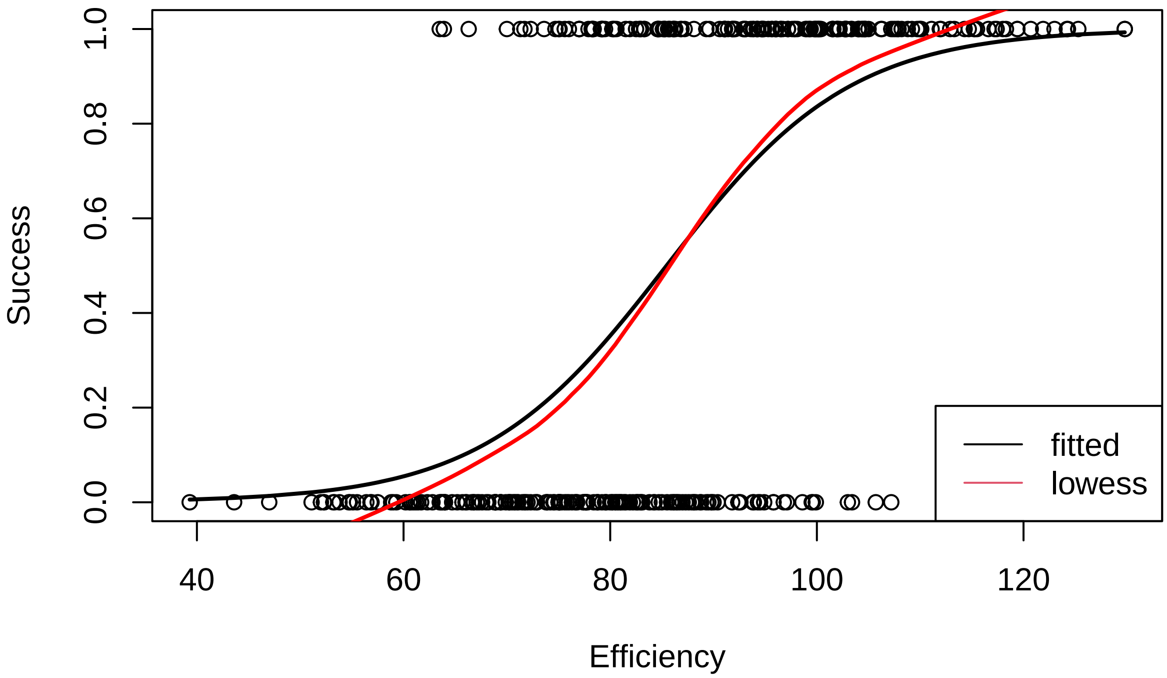



```r
# Efficiency game score logistic regression model
GLM1 <- glm(result ~ efficiency_game_score, family=binomial("logit"))
summary(GLM1)
```

```
##
## Call:
## glm(formula = result ~ efficiency_game_score, family = binomial("logit"))
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.24390  -0.75909   0.00566   0.73470   2.25332
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -9.56195    1.03383  -9.249   <2e-16 ***
## efficiency_game_score  0.11190    0.01202   9.306   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 493.52  on 355  degrees of freedom
## Residual deviance: 335.79  on 354  degrees of freedom
## AIC: 339.79
##
## Number of Fisher Scoring iterations: 5
```

```
index=order(efficiency_game_score)

plot(efficiency_game_score, result, xlab = "Efficiency", ylab = "Success")
lines(efficiency_game_score[index],fitted(GLM1)[index],lwd=2)
lines(lowess(efficiency_game_score, result),col="red",lwd=2)
legend("bottomright", c("fitted", "lowess"), lty =1, col = 1:2)
```




```
# make a multiple logistic regression model
multiGLM1 <- glm(result ~ offensive_rating + efficiency_game_score,
                 family=binomial("logit"))
summary(multiGLM1)
```

```
##
## Call:
## glm(formula = result ~ offensive_rating + efficiency_game_score,
##     family = binomial("logit"))
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.17541  -0.76865   0.00261   0.73619   2.26187
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -10.17196    1.93939  -5.245 1.56e-07 ***
## offensive_rating        0.01127    0.03008   0.375    0.708
## efficiency_game_score   0.10482    0.02225   4.712 2.46e-06 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 493.52  on 355  degrees of freedom
## Residual deviance: 335.65  on 353  degrees of freedom
## AIC: 341.65
##
## Number of Fisher Scoring iterations: 5
```

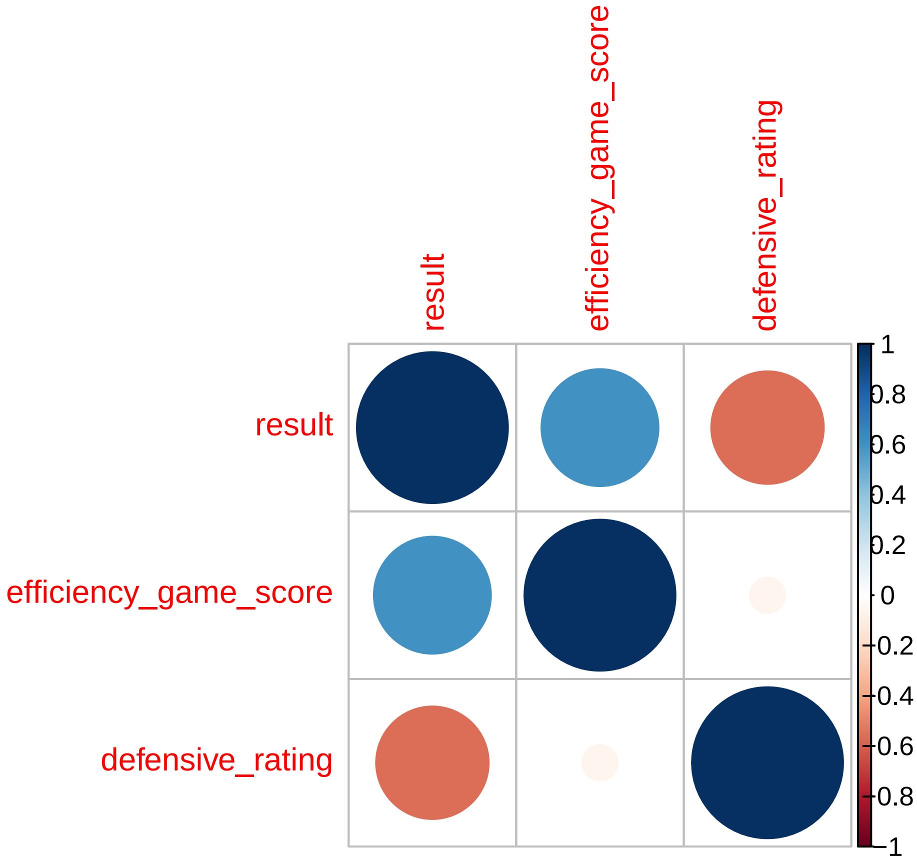
```
detach(subSet2)
```

The definition for efficiency game score is given by data source as:

$$(\text{points})+(0.4\cdot\text{fieldGoals})-(0.7\cdot\text{fieldGoalsAtt})-(0.4\text{fieldGoalsMiss})+(0.7\cdot\text{offensiveRbd})+(0.3\cdot\text{defensiveRbd})$$

$$+(\text{steals})+(0.7\cdot\text{assists})+(0.7\cdot\text{blocks})-(0.4\cdot\text{personalFouls})-(\text{turnovers})$$

```
attach(games)
# select efficiency_game_score and defensive rating
subSet2 <- data.frame(result, efficiency_game_score, defensive_rating)
subSet2 <- subSet2 %>%
  mutate(result = if_else(result == "win", 1, 0))
subSet2 <- subSet2 %>%
  mutate(result = strtoi(result))

detach(games)
attach(subSet2)
# checking correlations between predictor variables
predictorCor <- cor(subSet2)
corrplot(predictorCor)
```




```r
# make a multiple logistic regression model
multiGLM1.1 <- glm(result ~ efficiency_game_score + defensive_rating,
                   family=binomial("logit"))
summary(multiGLM1.1)
```

```
##
## Call:
## glm(formula = result ~ efficiency_game_score + defensive_rating,
##     family = binomial("logit"))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4904  -0.1228   0.0000   0.1646   2.3775
##
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)            17.78647    3.54043   5.024 5.07e-07 ***
## efficiency_game_score   0.27272    0.03579   7.620 2.53e-14 ***
## defensive_rating       -0.38228    0.05191  -7.364 1.79e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 493.52  on 355  degrees of freedom
## Residual deviance: 118.01  on 353  degrees of freedom
## AIC: 124.01
```

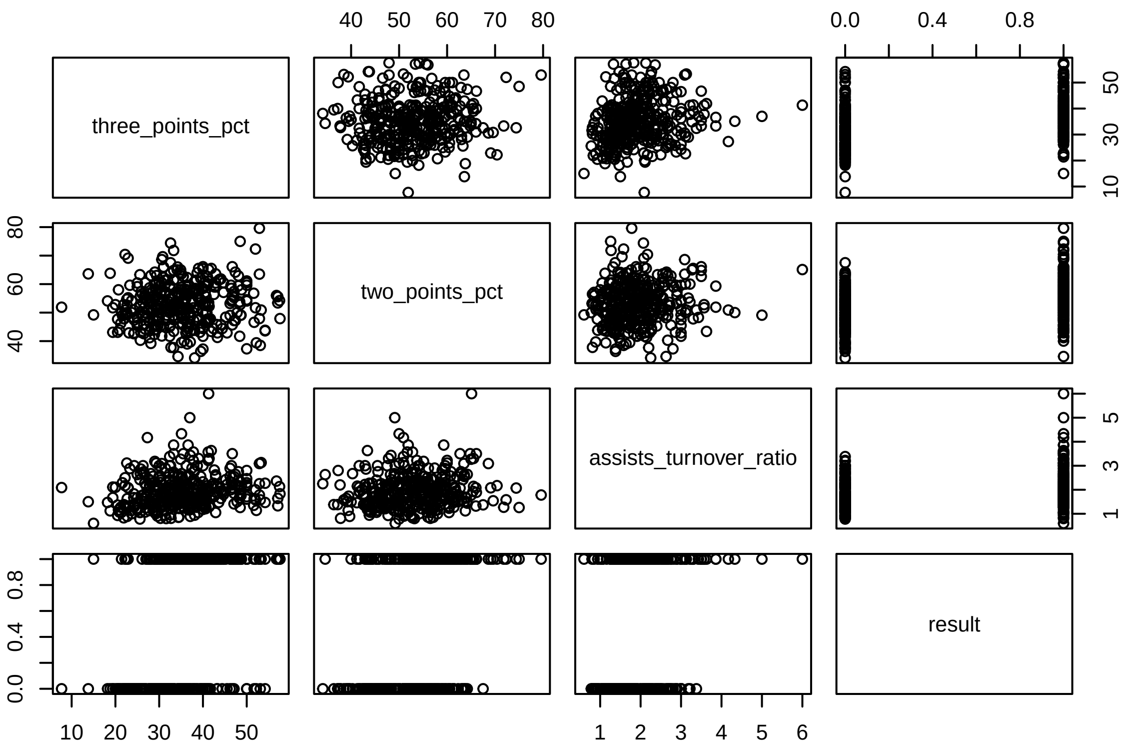
```
##
## Number of Fisher Scoring iterations: 8
```

```r
detach(subSet2)
```

We see, perhaps unsurprisingly, that both efficiency_game_score and defensive_rating are significant predictors in a team's result.

a) Ratio in Model

```r
attach(games)

subSet3 <- data.frame(three_points_pct, two_points_pct, assists_turnover_ratio)
subSet3 <- subSet3  %>%
  mutate(result = if_else(result == "win", 1, 0))
subSet3 <- subSet3 %>%
  mutate(result = strtoi(result))
pairs(subSet3)
```




```r
detach(games)
attach(subSet3)

multiGLM2 <- glm(result ~ three_points_pct + two_points_pct +
                   assists_turnover_ratio, family=binomial("logit"))
summary(multiGLM2)
```

```
##
## Call:
## glm(formula = result ~ three_points_pct + two_points_pct + assists_turnover_ratio,
##     family = binomial("logit"))
##
```

```
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.16038  -0.82126  -0.07239   0.80281   2.84714
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -12.57394    1.45981  -8.613  < 2e-16 ***
## three_points_pct         0.13029    0.01879   6.933 4.12e-12 ***
## two_points_pct           0.12510    0.02025   6.177 6.55e-10 ***
## assists_turnover_ratio   0.71521    0.20315   3.521 0.000431 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 493.52  on 355  degrees of freedom
## Residual deviance: 356.79  on 352  degrees of freedom
## AIC: 364.79
##
## Number of Fisher Scoring iterations: 5
```

```
detach(subSet3)
```

Transforming assist/turnover ratio logarithmically did not improve the p-value significantly.

C) Developing Confidence Statistic

a) Initial Testing

```
# getting data with home/away stat
games200 <- read.csv("tidy200_conf.csv")
attach(games200)
subSet4 <- data.frame(result, attendance, home_away, fast_break_made, blocks)
detach(games200)
attach(subSet4)
subSet4 <- subSet4 %>%
  mutate(attendance = strtoi(attendance))
subSet4 <- subSet4 %>%
  mutate(home_away = if_else(home_away == "home", 1, 0))
subSet4 <- subSet4 %>%
  mutate(attendance = if_else(home_away == 1, attendance, attendance * -1))
subSet4 <- subSet4 %>%
  mutate(result = if_else(result == "win", 1, 0))
detach(subSet4)
```

```
attach(subSet4)
multiGLM3 <- glm(result ~ attendance + fast_break_made + blocks)
summary(multiGLM3)
```

```
##
## Call:
## glm(formula = result ~ attendance + fast_break_made + blocks)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
```

```
## -0.66928  -0.48559  -0.00102   0.49114   0.63961
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.755e-01  8.574e-02   4.379 1.73e-05 ***
## attendance      2.943e-06  1.862e-06   1.581    0.115
## fast_break_made 1.233e-02  1.199e-02   1.029    0.305
## blocks          1.285e-02  1.259e-02   1.021    0.308
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.248132)
##
##     Null deviance: 65.500  on 261  degrees of freedom
## Residual deviance: 64.018  on 258  degrees of freedom
##   (2 observations deleted due to missingness)
## AIC: 384.32
##
## Number of Fisher Scoring iterations: 2
```

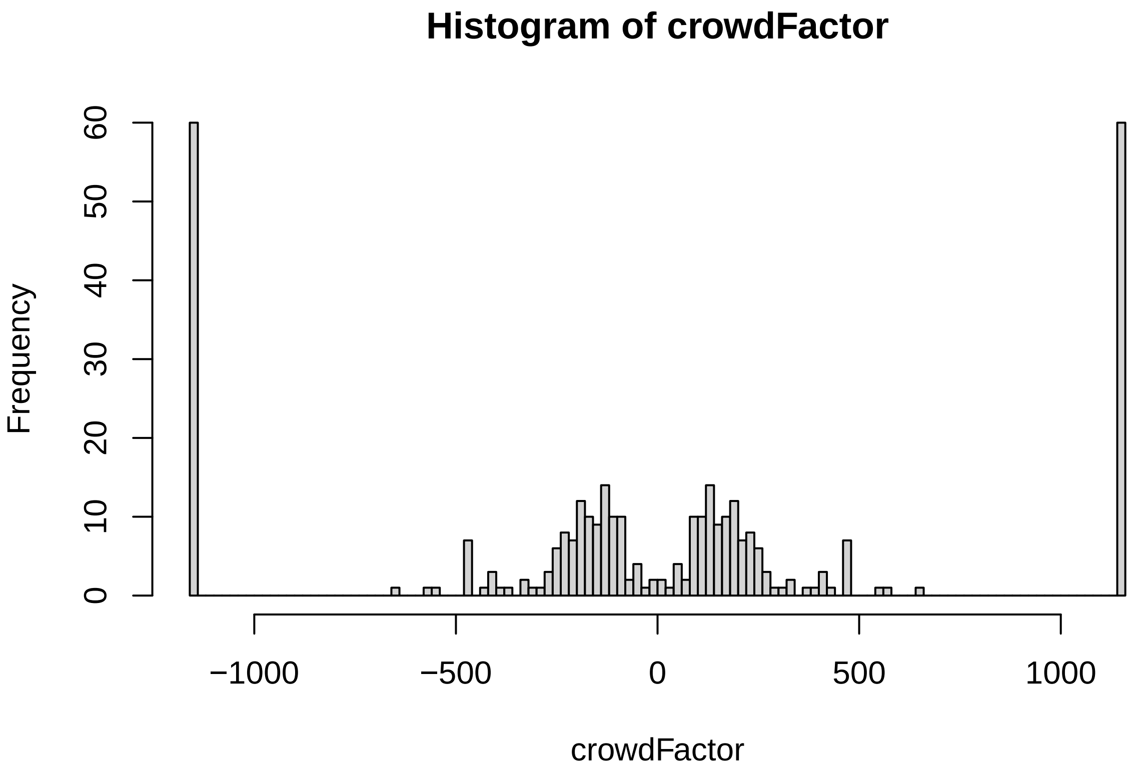
```
detach(subSet4)
```

b) Developing crowdFactor

```
# using a larger number of observations
attach(subSet5)
multiGLM2 <- glm(result ~ crowdFactor)
summary(multiGLM2)
```

```
##
## Call:
## glm(formula = result ~ crowdFactor)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6270  -0.4887   0.0000   0.4887   0.6270
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.000e-01  2.626e-02  19.040   <2e-16 ***
## crowdFactor 1.103e-04  3.774e-05   2.922   0.0037 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.245491)
##
##     Null deviance: 89.000  on 355  degrees of freedom
## Residual deviance: 86.904  on 354  degrees of freedom
## AIC: 514.28
##
## Number of Fisher Scoring iterations: 2
```

```r
hist(crowdFactor, breaks=100)
```




```r
detach(subSet5)
```

We see a somewhat significant p-value for our crowdFactor predictor at n = 132. But at n=356 we see a very significant value with $p \approx 0.004$.

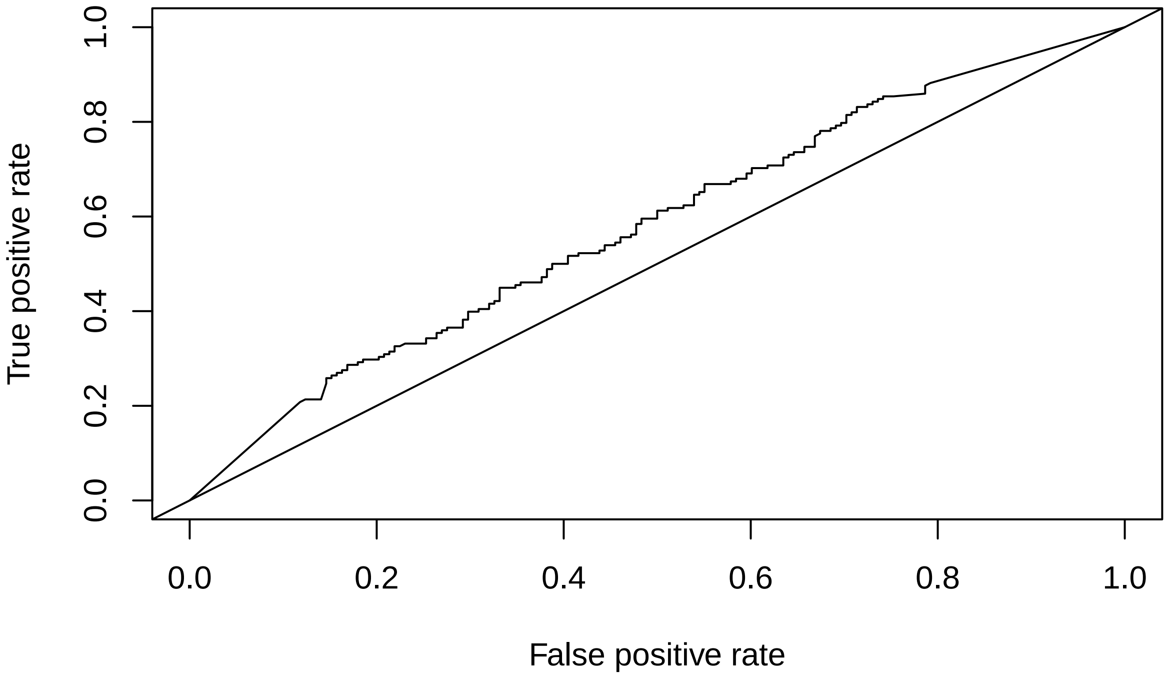
```r
# doing some further testing on our crowdFactor statistic
attach(subSet5)
hoslem.test(result, fitted(multiGLM2))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  result, fitted(multiGLM2)
## X-squared = 3.5826, df = 8, p-value = 0.8927
```

```r
pred = predict(multiGLM2,type="response")
pred1 = prediction(pred, labels = result)
roc = performance(pred1,"tpr", "fpr")
roc
```

```
## A performance instance
##   'False positive rate' vs. 'True positive rate' (alpha: 'Cutoff')
##   with 227 data points
```

```r
# plot ROC curve
plot(roc) # ROC curve
abline(0,1)
```




```r
detach(subSet5)
```

We see an p-value of $p = 0.000325$ *in our goodness of fit test, indicating we can reject the null hypothesis that the model is fit to the data. Our graph looks much better now (n=132->356), with the area under the T/F curve increasing. Also we see a much larger p-value at* $p = 0.8361$ *indicating we fail to reject the null hypothesis, which suggests the model is a good fit.*

c) Confidence Model

```r
attach(subSet5)
multiGLM4 <- glm(result ~ crowdFactor + blocks + fast_break_made)
summary(multiGLM4)
```

```
##
## Call:
## glm(formula = result ~ crowdFactor + blocks + fast_break_made)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.73057  -0.47868   0.00747   0.48328   0.71550
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.248e-01  7.640e-02   4.251 2.73e-05 ***
## crowdFactor     1.001e-04  3.776e-05   2.652  0.00837 **
## blocks          2.155e-02  1.118e-02   1.928  0.05464 .
## fast_break_made 1.501e-02  1.071e-02   1.402  0.16182
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2427073)
##
##     Null deviance: 89.000  on 355  degrees of freedom
```

```
## Residual deviance: 85.433  on 352  degrees of freedom
## AIC: 512.2
##
## Number of Fisher Scoring iterations: 2
```

```
detach(subSet5)
```

Not a bad start to adding variables to our sought after team confidence statistic. We have a somewhat significant p-value at $p \approx 0.056$.

```
attach(games)
subSet5.1 <- subSet5 %>%
  mutate(steals = steals)

detach(games)
attach(subSet5.1)
multiGLM5 <- glm(result ~ crowdFactor + blocks + steals)
summary(multiGLM5)
```

```
##
## Call:
## glm(formula = result ~ crowdFactor + blocks + steals)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.72833  -0.48049  -0.02694   0.47797   0.80975
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.447e-01  9.368e-02   2.612   0.0094 **
## crowdFactor 9.589e-05  3.776e-05   2.539   0.0115 *
## blocks      2.333e-02  1.112e-02   2.097   0.0367 *
## steals      1.866e-02  9.187e-03   2.031   0.0430 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2412345)
##
##     Null deviance: 89.000  on 355  degrees of freedom
## Residual deviance: 84.915  on 352  degrees of freedom
## AIC: 510.03
##
## Number of Fisher Scoring iterations: 2
```

```
AIC(multiGLM4)
```

```
## [1] 512.2016
```

```
AIC(multiGLM5)
```

```
## [1] 510.0346
```

```
hoslem.test(result, fitted(multiGLM5))
```

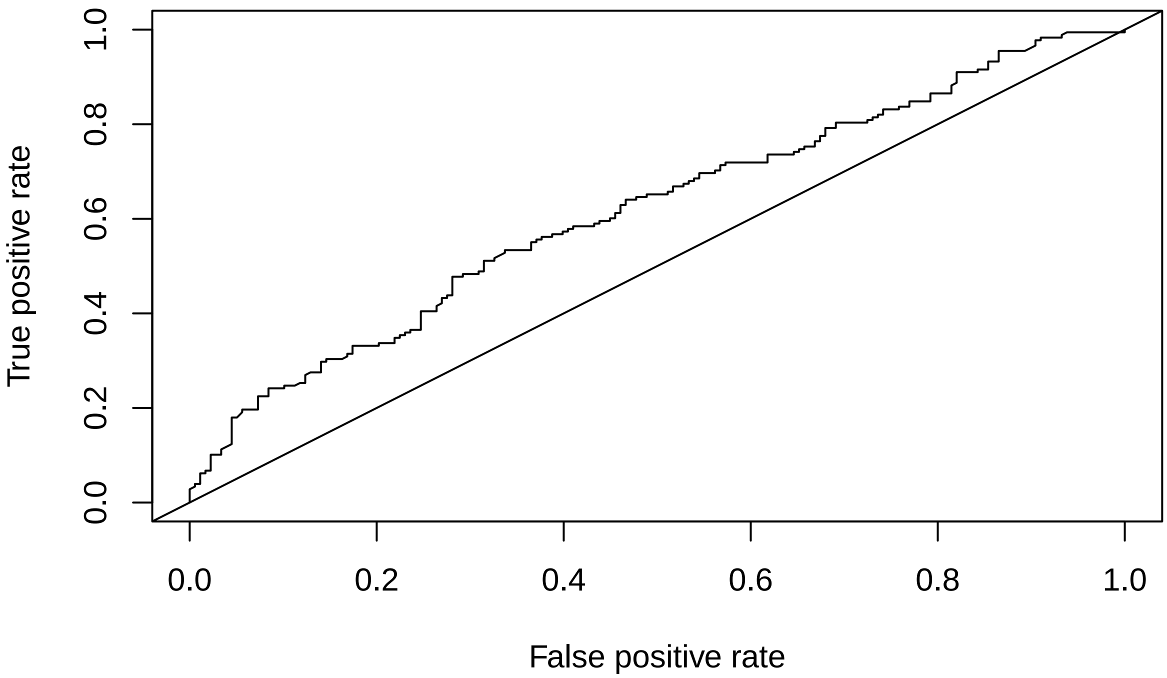
```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  result, fitted(multiGLM5)
```

```
## X-squared = 7.0873, df = 8, p-value = 0.5272
```

```
pred = predict(multiGLM5,type="response")
pred1 = prediction(pred, labels = result)
roc = performance(pred1,"tpr", "fpr")
roc
```

```
## A performance instance
##   'False positive rate' vs. 'True positive rate' (alpha: 'Cutoff')
##   with 324 data points
```

```
# plot ROC curve
plot(roc) # ROC curve
abline(0,1)
```



Adding blocks and steals to the team confidence model gives significant p-values for each predictor. We get a better AIC score for our confidence model with blocks and steals, as opposed to our confidence model with blocks and fast breaks made.