DSC 520 - Final Project

Dan Wiltse

February 26th 2020

Introduction

College football is a game enjoyed by millions of fans across the country in the fall. Part of the uniqueness of college football, unlike the pros, is that each player decides where they want to go to school, so the recruiting process plays a big part in how the teams perform on the field each week. The better players you can recruit to your school, the more likely you are to have a winning team during the season. Players can redshirt in college, so they have a chance to develop over five years, so you have to look at the recruiting across that time period to see if the players you brought in over that time worked together on the field to acheive success.

Problem Statement

I want to understand how much recruiting impacts on-field performance in college football, based on recruiting ranking and points, to quantify the impact it has on differences between winning and losing.

How I addressed problem statement

Data

The data used on this project came from a collge football API for R that utilized the resources below. The analysis used the following metrics, defined below:

Season Data 1) Team - College football teams from the five major football conferences (ACC, Big Ten, Big 12, Pac 12, and SEC) 2) Wins - Total Wins by the team during the 2019 season. 3) Five Year Average Recruiting Points - Average recruiting points (as determined by year by collegefootlldata.com) across last 5 recruiting classes (2015-2019)

Game Data 1) Home Team - Team hosting the game (or determined home team on neutral site) 2) Home Team Score - Final score of game by home team 3) Away Team - Visit team of game 4) Away Team Score - Final score of game by visiting team 5) Home Team Five Year Average Recruiting Points - Average recruiting points for the home team (as determined by year by collegefootlldata.com) across last recruiting classes (2015-2019) 6) Away Team Five Year Average Recruiting Points - Average recruiting points for the away team (as determined by year by collegefootlldata.com) across last 5 recruiting classes (2015-2019) 7) Recruiting Point Difference - Home Team Five Year Average Points - Away Team Five Year Average Points 8) Home Team Win - flag on whether or not the home team won the game

Resources: 1) https://github.com/meysubb/cfbscrapR 2) https://collegefootballdata.com/

Methodolgy

I addressed the problem statement in two ways:

1) First, I looked at the season win-loss totals for the 2019 season for the major college football conferences, and analyzed how how much impact recruiting has by using a five year average recruiting points, using linear regression.
2) In addition, I wanted to see if I could predict the game by game performance based on recruiting ranking comparisons as well. To accomplish this, I used logistic regression to see how accurate the model would be in picking the winner of games during the 2019 season based on which teams had higher recruiting rankings.

Analysis

Summarize the interesting insights that your analysis provided

There were several interesting findings in my analysis. Using linear regression, there was a significant overall relationship between recruiting points and on-field performance ($p < 0.0001$).

It also showed that overall, average recruiting points accounted for 28% of the variance in wins. Breaking out the regression results by conferences, there were also big differences between conferences as well. In the SEC, 56% of the variance could be explained by recruiting, where as in the Pac 12, only 3% of the variance was explained.

In my analysis using logistic regression, it was able to accurately predict the outcome of nearly two thirds of the games in the validation data set. This was suprising to me as it was only using one variable to predict the outcome of the game.

Implications

Summarize the implications to the consumer (target audience) of your analysis

The analysis shows just how important recruiting is to the performance of teams on the field in college football throughout the season. Across the major conferences in college football, recruiting accounts to almost 1/3 of the variance in winning, not accounting for coaching or development of the talent they recruited. It also seems to be a bigger impact in certain conferences, especially the Southeastern Conference.

The other implication is that you can predict the outcome of games based on discrepancies in five year recruiting performance almost 2/3rd of the time, showing just how big an impact recruiting has on week to week performance as well.

Limitations

The limitations of my analysis is that I just looked at the average recruiting performance overall, I didnt break it out by position to see if there were certain positions that were more important than others, like quarterbacks versus kickers. I also did not account for attrition in the analysis, so if highly recruited players left the school, they would still factor into the recruiting rankings but not the on-field performance, skewing the results. I also looked at only the current season of data, it would be helpful to build a more meaningful trend by using several seasons worth of data to train my model to see if the results are consistent over time.

Concluding Remarks

Overall, my analysis discovered a strong relationship between off field performance (recruiting) and on-field performance (wins). It would be worthwhile to expand the analysis to see the deeper impact recruiting has on how the teams perform on the field.

Appendix

R script

While correlation does not imply causation, the two metrics were significantly correlated with each other, with a 54% correlation.

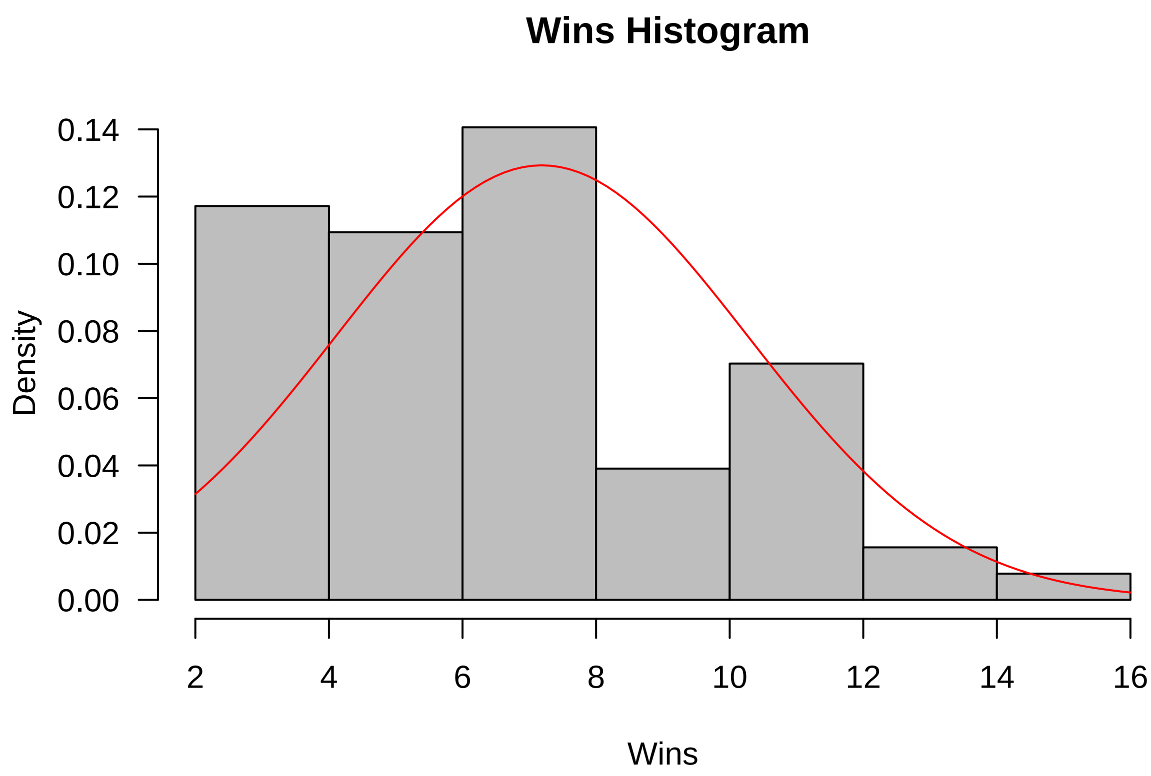
```r
#Filter data down to only the 5 major football conferences
majorconf2019 <-filter(recruiting2019, conference %in% c('SEC', 'Big 12', 'Big Ten', 'ACC', 'Pac 12'))

# Filter to each conference separately for regression
SEC <- majorconf2019[majorconf2019$conference == "SEC",]

# Filter to each conference separately for regression
Big12 <- majorconf2019[majorconf2019$conference == "Big 12",]

# Filter to each conference separately for regression
BigTen <- majorconf2019[majorconf2019$conference == "Big Ten",]

# Filter to each conference separately for regression
ACC <- majorconf2019[majorconf2019$conference == "ACC",]


# Filter to each conference separately for regression
Pac12 <- majorconf2019[majorconf2019$conference == "Pac 12",]

## Correlation Analysis of wins and recruiting

cor.test(majorconf2019$wins, majorconf2019$five_year_avg_points)
```

```
##
##  Pearson's product-moment correlation
##
## data:  majorconf2019$wins and majorconf2019$five_year_avg_points
## t = 5.0742, df = 62, p-value = 0.000003793
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.3413323 0.6949649
## sample estimates:
##       cor
## 0.5416922
```

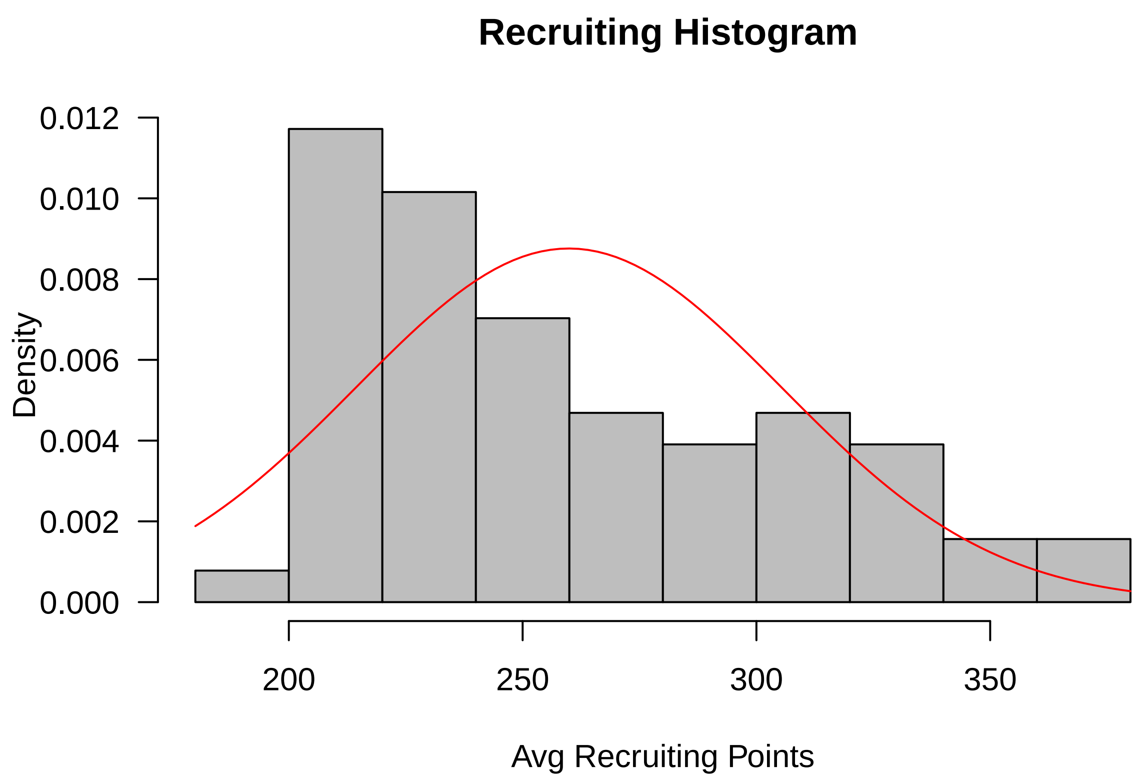
Data Validation

Data was already clean before analysis with no missing values, the histograms show that the data is normally distributed and does not violate assumptions necessary for linear regression.

```
# adding a normal distribution line in histogram for wins
hist(majorconf2019$wins, freq=FALSE, col="gray", xlab="Wins", main=" Wins Histogram", las=1)
curve(dnorm(x, mean=mean(majorconf2019$wins), sd=sd(majorconf2019$wins)), add=TRUE, col="red") #line
```




```
# adding a normal distribution line in histogram for recruiting points
hist(majorconf2019$five_year_avg_points, freq=FALSE, col="gray", xlab="Avg Recruiting Points", main=" R
curve(dnorm(x, mean=mean(majorconf2019$five_year_avg_points), sd=sd(majorconf2019$five_year_avg_points)
```

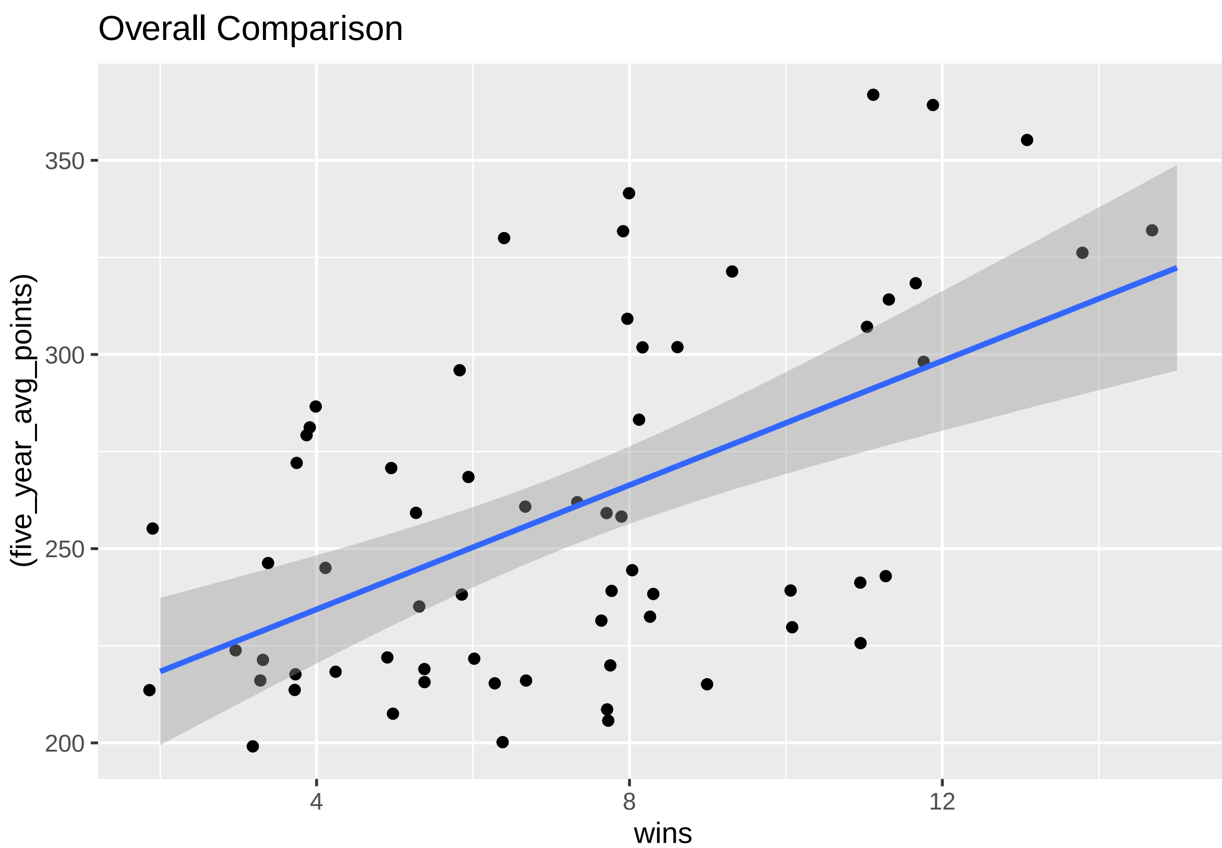


Analysis

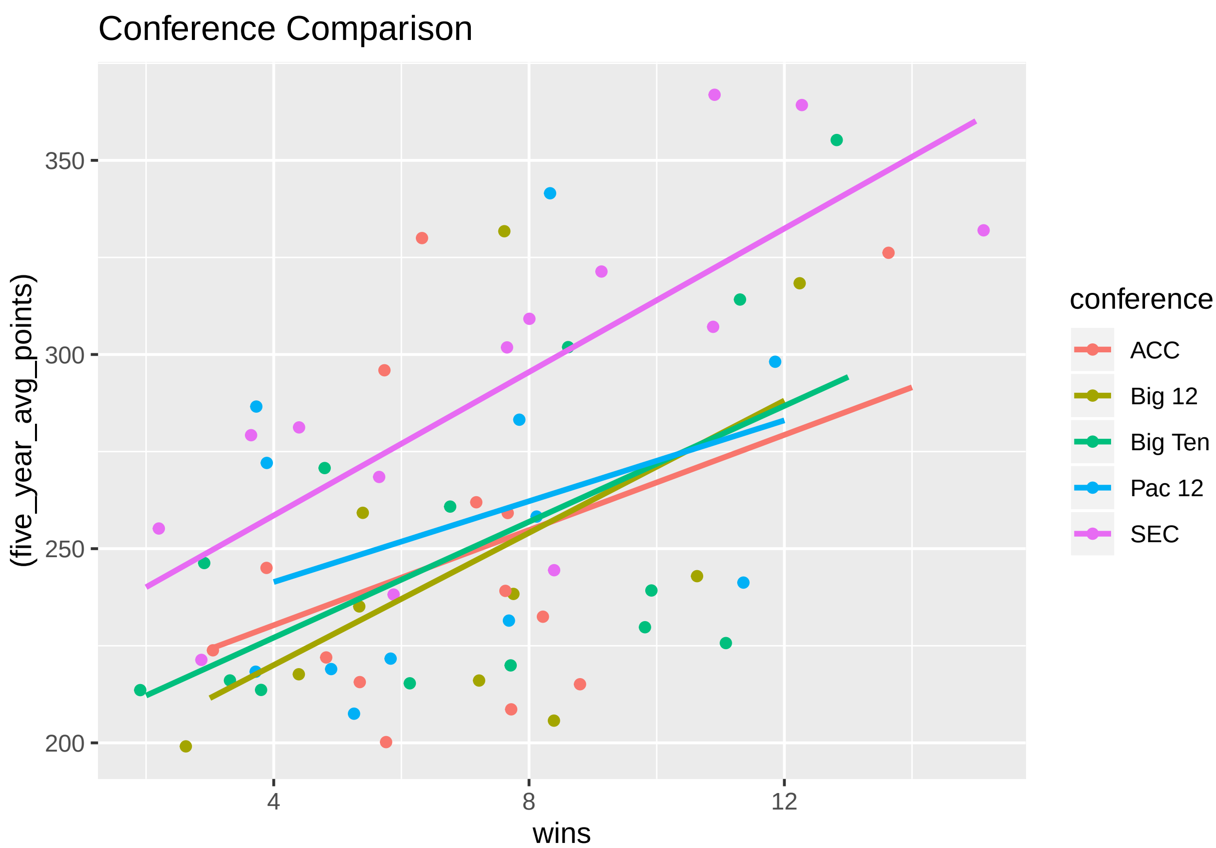
Scatterplot with linear regression line. Shows a linear relationship between wins and points, especially in the SEC

Output a scatterplot overall and by conference

```
#Overall
ggplot(data = majorconf2019, aes(x =wins , y = (five_year_avg_points))) +
  ggtitle("Overall Comparison") +
  geom_point(position = "jitter") +
  stat_smooth(method = "lm", se = TRUE)
```




```r
#By Conference
ggplot(data = majorconf2019, aes(x =wins , y = (five_year_avg_points), col = conference)) +
  ggtitle("Conference Comparison") +
  geom_point(position = "jitter") +
  stat_smooth(method = "lm", se = FALSE)
```



Linear Regression

Output for overall linear regression model, as well as breakout by conference

```
simplemodel <-lm(wins ~ five_year_avg_points, data = majorconf2019)
summ(simplemodel)
```

```
## MODEL INFO:
## Observations: 64
## Dependent Variable: wins
## Type: OLS linear regression
##
## MODEL FIT:
## F(1,62) = 25.75, p = 0.00
## R² = 0.29
## Adj. R² = 0.28
##
## Standard errors: OLS
## --------------------------------------------------------
##                               Est.   S.E.   t val.      p
## -------------------------- -------- ------ -------- ------
## (Intercept)                  -2.35   1.91    -1.23   0.22
## five_year_avg_points          0.04   0.01     5.07   0.00
## --------------------------------------------------------
```

```r
#Breakout by conference

simplemodelSEC <-lm(wins ~ five_year_avg_points, data = SEC)
summ(simplemodelSEC)
```

```
## MODEL INFO:
## Observations: 14
## Dependent Variable: wins
## Type: OLS linear regression
##
## MODEL FIT:
## F(1,12) = 17.67, p = 0.00
## R² = 0.60
## Adj. R² = 0.56
##
## Standard errors: OLS
## ----------------------------------------------------------
##                                 Est.   S.E.   t val.      p
## -------------------------- --------- ------ --------- ------
## (Intercept)                   -11.20   4.53    -2.47   0.03
## five_year_avg_points            0.06   0.02     4.20   0.00
## ----------------------------------------------------------
```

```r
simplemodelACC <-lm(wins ~ five_year_avg_points, data = ACC)
summ(simplemodelACC)
```

```
## MODEL INFO:
## Observations: 14
## Dependent Variable: wins
## Type: OLS linear regression
##
## MODEL FIT:
## F(1,12) = 2.14, p = 0.17
## R² = 0.15
## Adj. R² = 0.08
##
## Standard errors: OLS
## --------------------------------------------------------
##                               Est.   S.E.   t val.      p
## -------------------------- ------ ------ --------- ------
## (Intercept)                   0.79   4.25     0.18   0.86
## five_year_avg_points          0.02   0.02     1.46   0.17
## --------------------------------------------------------
```

```r
simplemodelBigTen <-lm(wins ~ five_year_avg_points, data = BigTen)
summ(simplemodelBigTen)
```

```
## MODEL INFO:
## Observations: 14
## Dependent Variable: wins
## Type: OLS linear regression
##
```

```
## MODEL FIT:
## F(1,12) = 6.52, p = 0.03
## R² = 0.35
## Adj. R² = 0.30
##
## Standard errors: OLS
## ----------------------------------------------------------
##                                Est.   S.E.   t val.      p
## -------------------------- ------- ------ -------- ------
## (Intercept)                   -4.59   4.72    -0.97   0.35
## five_year_avg_points           0.05   0.02     2.55   0.03
## ----------------------------------------------------------
```

```r
simplemodelBig12 <-lm(wins ~ five_year_avg_points, data = Big12)
summ(simplemodelBig12)
```

```
## MODEL INFO:
## Observations: 10
## Dependent Variable: wins
## Type: OLS linear regression
##
## MODEL FIT:
## F(1,8) = 3.46, p = 0.10
## R² = 0.30
## Adj. R² = 0.21
##
## Standard errors: OLS
## ----------------------------------------------------------
##                                Est.   S.E.   t val.      p
## -------------------------- ------- ------ -------- ------
## (Intercept)                   -1.64   4.77    -0.34   0.74
## five_year_avg_points           0.04   0.02     1.86   0.10
## ----------------------------------------------------------
```

```r
simplemodelPac12 <-lm(wins ~ five_year_avg_points, data = Pac12)
summ(simplemodelPac12)
```

```
## MODEL INFO:
## Observations: 12
## Dependent Variable: wins
## Type: OLS linear regression
##
## MODEL FIT:
## F(1,10) = 1.37, p = 0.27
## R² = 0.12
## Adj. R² = 0.03
##
## Standard errors: OLS
## ---------------------------------------------------------
##                               Est.   S.E.   t val.      p
## -------------------------- ------ ------ -------- ------
## (Intercept)                   0.97   5.14     0.19   0.85
## five_year_avg_points          0.02   0.02     1.17   0.27
## ---------------------------------------------------------
```

Logistic Regression

The Logistic regression analysis details below. It was found to be nearly 65% accurate in predicting if the home team would win the game, based on recruiting point difference.

```r
lgm1 <- glm(home_team_win ~ home_recruiting_dff , data = game2019, family = binomial()  )
summary(lgm1)
```

```
##
## Call:
## glm(formula = home_team_win ~ home_recruiting_dff, family = binomial(),
##     data = game2019)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9184  -1.0671   0.5156   0.9660   1.9292
##
## Coefficients:
##                     Estimate Std. Error z value        Pr(>|z|)
## (Intercept)         0.302870   0.127071   2.383          0.0172 *
## home_recruiting_dff 0.016437   0.002486   6.612 0.0000000000379 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 420.62  on 306  degrees of freedom
## Residual deviance: 364.38  on 305  degrees of freedom
## AIC: 368.38
##
## Number of Fisher Scoring iterations: 3
```

```r
# Split the data into training and validation data sets
split <- sample.split(game2019, SplitRatio = 0.8)
split
```

```
##  [1]  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE FALSE FALSE  TRUE
```

```r
train <- subset(game2019, split == "TRUE")
```

```
## Warning: Length of logical index must be 1 or 307, not 16
```

```r
validate <- subset(game2019, split == "FALSE")
```

```
## Warning: Length of logical index must be 1 or 307, not 16
```

```r
# Train model using training data set
lgm2 <- glm(home_team_win ~ home_recruiting_dff , data = game2019, family = binomial()  )
summary(lgm2)
```

```
##
## Call:
## glm(formula = home_team_win ~ home_recruiting_dff, family = binomial(),
##     data = game2019)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9184  -1.0671   0.5156   0.9660   1.9292
##
## Coefficients:
##                     Estimate Std. Error z value        Pr(>|z|)
## (Intercept)         0.302870   0.127071   2.383          0.0172 *
## home_recruiting_dff 0.016437   0.002486   6.612 0.0000000000379 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 420.62  on 306  degrees of freedom
## Residual deviance: 364.38  on 305  degrees of freedom
## AIC: 368.38
##
## Number of Fisher Scoring iterations: 3
```

```
# Run validation data through the model built on training data
res <- predict(lgm2, validate, type = "response")
res
```

```
##         1         2         3         4         5         6         7         8
## 0.3905130 0.5778810 0.3701937 0.6677053 0.5171696 0.3132759 0.9148178 0.6357483
##         9        10        11        12        13        14        15        16
## 0.6138976 0.7251771 0.5858790 0.8044058 0.7230756 0.5207559 0.4817868 0.1145025
##        17        18        19        20        21        22        23        24
## 0.4433867 0.8792544 0.6761188 0.7573331 0.5193692 0.6126267 0.9229538 0.7207128
##        25        26        27        28        29        30        31        32
## 0.9148562 0.6222861 0.5963906 0.5742120 0.4415296 0.4417566 0.3258052 0.2459922
##        33        34        35        36        37        38        39        40
## 0.8646525 0.4869430 0.7255962 0.5211744 0.5348719 0.8453979 0.3603542 0.4197219
##        41        42        43        44        45        46        47        48
## 0.6645688 0.7688783 0.4454972 0.1601373 0.4491626 0.8627173 0.4687109 0.5144191
##        49        50        51        52        53        54        55        56
## 0.6178872 0.6308547 0.5065164 0.5400450 0.6734062 0.6385452 0.6818939 0.8091054
##        57        58        59        60        61        62        63        64
## 0.6870059 0.8986433 0.5962481 0.7977109 0.1681418 0.2392210 0.4994323 0.6433342
##        65        66        67        68        69        70        71        72
## 0.3911783 0.7695087 0.7751502 0.5438316 0.6430927 0.8364372 0.7565528 0.9033750
##        73        74        75        76
## 0.3202410 0.6512969 0.4549931 0.6679825
```

```
res2 <-predict(lgm2, train, type = "response")
res2
```

```
##         1         2         3         4         5         6         7         8
```

```
## 0.2488324 0.3440651 0.2116926 0.9111648 0.7661330 0.8340759 0.7230031 0.5854483
##         9        10        11        12        13        14        15        16
## 0.4657159 0.5058754 0.6832472 0.5315580 0.8946002 0.8270586 0.5256764 0.6170874
##        17        18        19        20        21        22        23        24
## 0.4233934 0.5566364 0.6389321 0.3970894 0.8238235 0.6278332 0.7196992 0.6971496
##        25        26        27        28        29        30        31        32
## 0.7601134 0.7479391 0.8019425 0.7701145 0.1803314 0.3201981 0.2445135 0.2629260
##        33        34        35        36        37        38        39        40
## 0.3926904 0.7207525 0.8325186 0.6612410 0.9360958 0.7138330 0.5472059 0.3258196
##        41        42        43        44        45        46        47        48
## 0.6707036 0.4693741 0.3035995 0.1699984 0.5598306 0.1555189 0.1381684 0.3267519
##        49        50        51        52        53        54        55        56
## 0.3461678 0.4008891 0.1186701 0.5677697 0.8158317 0.5397347 0.6161315 0.5683182
##        57        58        59        60        61        62        63        64
## 0.7409408 0.6318723 0.7261064 0.8642865 0.5496077 0.5193118 0.8158466 0.7264397
##        65        66        67        68        69        70        71        72
## 0.6780380 0.6494056 0.5816214 0.8791461 0.5332112 0.5795161 0.7843519 0.6414008
##        73        74        75        76        77        78        79        80
## 0.7915524 0.6983075 0.5483786 0.6676251 0.7124611 0.9065053 0.4808265 0.8576892
##        81        82        83        84        85        86        87        88
## 0.4505132 0.7750070 0.5987228 0.1801614 0.3227725 0.7061096 0.3625698 0.4910338
##        89        90        91        92        93        94        95        96
## 0.8902646 0.6322393 0.6192911 0.4363886 0.3229450 0.7031401 0.2864414 0.6234676
##        97        98        99       100       101       102       103       104
## 0.3518143 0.2603413 0.3933884 0.7266814 0.3836885 0.5261518 0.5259223 0.5914511
##       105       106       107       108       109       110       111       112
## 0.5360903 0.7635907 0.7008431 0.5065821 0.6001121 0.9148818 0.6416200 0.3930511
##       113       114       115       116       117       118       119       120
## 0.4265667 0.4885118 0.3663467 0.3524518 0.3922201 0.3461157 0.4752077 0.5639417
##       121       122       123       124       125       126       127       128
## 0.4547322 0.5062288 0.7727173 0.5906724 0.5722014 0.1277693 0.2524506 0.1208184
##       129       130       131       132       133       134       135       136
## 0.1165481 0.3603694 0.1674440 0.6278716 0.1753091 0.2815616 0.2665735 0.4907216
##       137       138       139       140       141       142       143       144
## 0.4840853 0.5621462 0.8511516 0.4081506 0.7301995 0.5314680 0.6648106 0.7341961
##       145       146       147       148       149       150       151       152
## 0.4127725 0.8324361 0.6675011 0.8572351 0.3073589 0.2527237 0.3604224 0.2402636
##       153       154       155       156       157       158       159       160
## 0.7267271 0.8382461 0.5325566 0.6773486 0.8760064 0.6383327 0.5848657 0.8525373
##       161       162       163       164       165       166       167       168
## 0.6005065 0.5821653 0.8761064 0.4965065 0.4001866 0.8412016 0.6550726 0.3189543
##       169       170       171       172       173       174       175       176
## 0.3613171 0.3715513 0.6914000 0.6004197 0.6657184 0.3901766 0.3996580 0.3596877
##       177       178       179       180       181       182       183       184
## 0.4899165 0.4060479 0.7815306 0.3452086 0.3222984 0.2166499 0.2912969 0.6415595
##       185       186       187       188       189       190       191       192
## 0.3578348 0.4526870 0.7698700 0.6631310 0.5777767 0.8762954 0.6722121 0.8982594
##       193       194       195       196       197       198       199       200
## 0.3177272 0.3058632 0.5160449 0.3417614 0.1530354 0.2562288 0.1815494 0.8755307
##       201       202       203       204       205       206       207       208
## 0.4373268 0.7295708 0.4814338 0.7779738 0.7836337 0.8471598 0.8355537 0.6677491
##       209       210       211       212       213       214       215       216
## 0.7452148 0.8936885 0.3392220 0.7124409 0.4308904 0.3960825 0.5409676 0.6030826
##       217       218       219       220       221       222       223       224
```

```
## 0.5299287 0.3178056 0.2805784 0.7117939 0.8262814 0.6139365 0.7882353 0.5874096
##       225       226       227       228       229       230       231
## 0.4281838 0.8416840 0.6061632 0.4775120 0.9011076 0.6944777 0.5200585
```

```
#Validate model using confusion matrix
confmatrix <- table(Actual_Value=train$home_team_win, Predicted_Value = res2 >0.5)
confmatrix
```

```
##             Predicted_Value
## Actual_Value FALSE TRUE
##            0    61   47
##            1    30   93
```

```
#Accuracy
(confmatrix[[1,1]] + confmatrix[[2,2]])/sum(confmatrix)
```

```
## [1] 0.6666667
```

Logistic Regression Validation

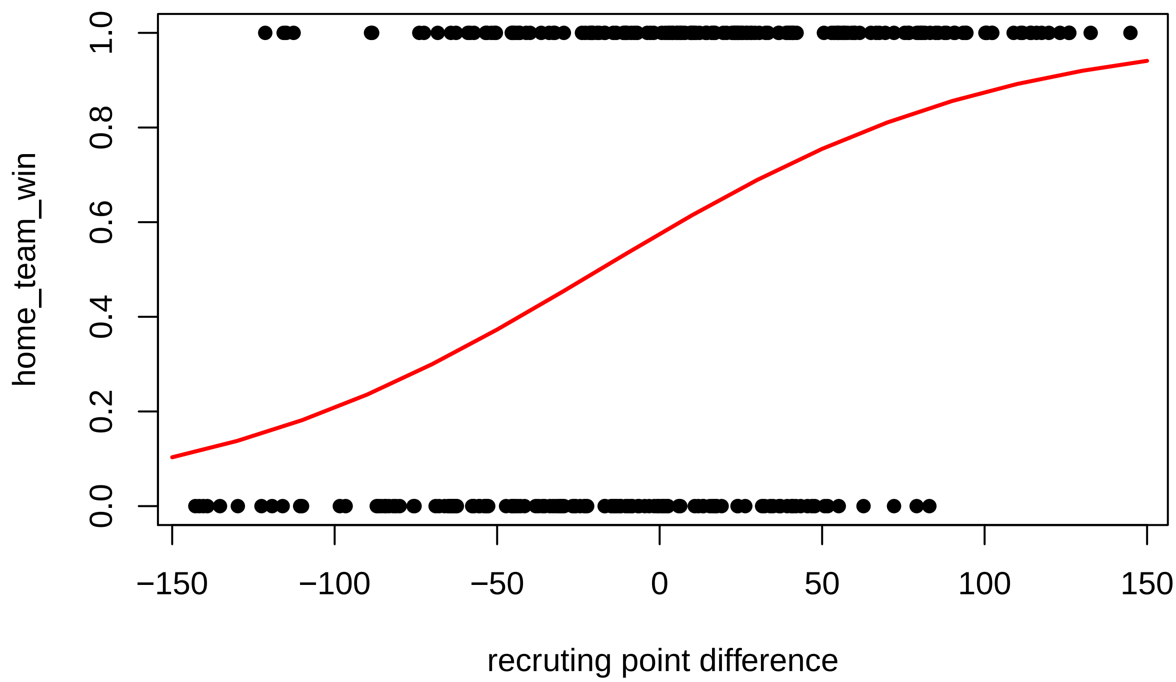
The first graph shows that as the recruiting point difference increases positively for the home team, the more likely they are going to win the game2019

The second graph (Receiver Operating Characteristic (ROC) curve) shows the model is better than chance at predicting who would win the game.

```
range(game2019$home_recruiting_dff)
```

```
## [1] -142.872  144.882
```

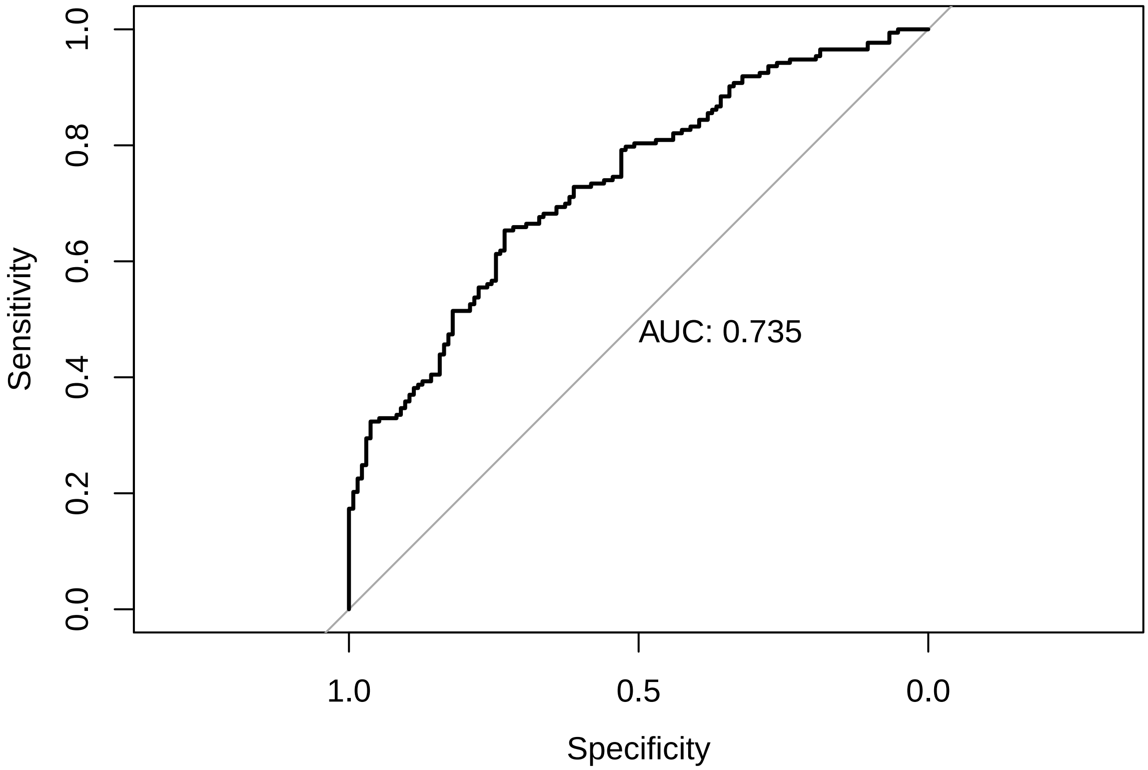
```
xnumeracy <-seq(-150, 150, 20)

ynumeracy <- predict(lgm2, list(home_recruiting_dff=xnumeracy),type="response")


plot(game2019$home_recruiting_dff, game2019$home_team_win, pch = 16, xlab = "recruting point difference

lines(xnumeracy, ynumeracy, col = "red", lwd = 2)
```




```r
test_prob = predict(lgm2, newdata = game2019, type = "response")
test_roc = roc(game2019$home_team_win ~ test_prob, plot = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```


AUC: 0.735
Sensitivity
Specificity
1.0
0.8
0.6
0.4
0.2
0.0
1.0
0.5
0.0