

# DATA 621 Final Project

## Fantasy Football Scoring Projections

Shane Hylton

2022-12-14

### Contents

<b>Introduction</b>	<b>1</b>
<b>Research Questions:</b>	<b>2</b>
<b>Data Preparation</b>	<b>2</b>
<b>Question 1: Can I predict Expected Fantasy Points Given Targets?</b>	<b>4</b>
Model 1: Simple Linear Regression . . . . .	4
Validation . . . . .	5
Model 2: XGBoost Linear Regression . . . . .	6
Comments . . . . .	6
<b>Question 2: Can I Predict Fantasy Points Given Targets and Yards Per Catch as Inputs?</b>	<b>7</b>
Model 3: Multiple Regression . . . . .	7
Model 4: Negative Binomial . . . . .	7
Model 5: XGBoost Regression Model with TGT and YPR as Inputs . . . . .	8
<b>Question 3: Can I Predict Touchdowns Given All Inputs?</b>	<b>9</b>
Model 6: Multiple Regression . . . . .	9
Model 7: XGBoost Touchdowns From All Inputs . . . . .	10
Comments . . . . .	11
<b>Model Selection and Week 14 Predictions</b>	<b>12</b>
Week 14 Projections . . . . .	12
<b>Week 14 Performance</b>	<b>12</b>
Comments . . . . .	12
<b>Sources:</b>	<b>13</b>

## Introduction

Over the last few years, fantasy football has been cemented as my favorite hobby. One of the big mysteries I have been plagued by in previous years is the art of projecting scores. Different fantasy football websites will all have different expected point totals for players, and I have never been able to find out exactly how they project point totals and expected statistics. I even reached out to ESPN's head fantasy football projection modeler, and he couldn't really tell me anything specific. He mainly told me to find something unique and master it.

I am relatively confident that a lot of week to week projections are built on at least one input using an expert's best guess. For example, an expert may believe that a player will be targeted 7 times in a game, and using a model, they will predict the total score for the week, along with all of the other important stat categories.

Below is a sample of ESPN's Wide Receiver Scoring Projections for Week 13 of the 2022 NFL Season.




WIDE RECEIVERS				STATUS		PASSING			RUSHING			RECEIVING				RESEARCH		TOTAL
RANK	PLAYER	TYPE	ACTION	C / A	YDS	TD	INT	CAR	YDS	TD	REC	YDS	TD	TAR	%ROST	+/-	FPTS	
1	<div><div>Davante Adams</div><div>LV WR</div></div>			0/0	0	0	0	1	3	0	7	94	1	10	99.9	0	20.7	
2	<div><div>Tyreek Hill</div><div>Mia WR</div></div>			0/0	0	0	0	1	3	0	8	97	0	11	99.9	0	20.4	
3	<div><div>Justin Jefferson</div><div>Min WR</div></div>			0/0	0	0	0	0	1	0	7	92	0	11	99.9	0	19.5	

Figure 1: ESPN Top 3 Week 13 Wide Receiver Projections

For Wide Receivers, passing statistics don't matter, so I won't go into detail with those. The remaining data column descriptions are below.

### Rushing:

CAR: Carries YDS: Rushing Yards TD: Rushing Touchdowns

### Receiving:

REC: Catches YDS: Receiving Yards TD: Receiving Touchdowns TAR: Targets (The amount of times the football is thrown to the receiver)

The "RESEARCH" columns do not mean much from a projections standpoint, but they do indicate how many fantasy football teams across the ESPN platform own a certain player, and the  $\pm$  indicates the percent change in player ownership.

### TOTAL

Total points represent the total fantasy points a player is expected to score.

## Research Questions:

The response variable in fantasy football is total points scored, but there is already an equation that is used to score a player's performance.

The scoring format I will explore is as follows:

$$FPTS = 1 \times REC + 0.1 \times YDS + 6 \times TD$$

TD represents both receiving and rushing touchdowns, and YDS represents both rushing and receiving yards.

Can I predict the expected fantasy point total in a given week with only targets as an input?

Can I predict fantasy points given targets and yards per catch as inputs?

Can I predict touchdowns scored given all inputs?

In the first two questions, fantasy points will be the target variable. In the final question, the target variable will be touchdowns.

## Data Preparation

I will be using data acquired from FantasyPros as the input. We'll be exploring predictions for Week 14 of the NFL season. I will use data from weeks 1 to 13, and I will construct several different regression models.

All weeks have been combined into one dataframe titled "season." I will remove the following columns that are not necessary to the model.

Player, ROST, Rank, FPTS.G (FPTS covers this)

The remaining columns all may be of value to the projections, and they will be adjusted or removed from models as needed.

Table 1: Summary of Season Data

REC	TGT	YDS	YPR	LG
Min. : 0.000	Min. : 0.00	Min. : -5.00	Min. : -5.000	Min. : 0.000
1st Qu.: 0.000	1st Qu.: 0.00	1st Qu.: 0.00	1st Qu.: 0.000	1st Qu.: 0.000
Median : 0.000	Median : 0.00	Median : 0.00	Median : 0.000	Median : 0.000
Mean : 1.078	Mean : 1.69	Mean : 13.59	Mean : 3.866	Mean : 6.407
3rd Qu.: 1.000	3rd Qu.: 2.00	3rd Qu.: 12.00	3rd Qu.: 7.000	3rd Qu.: 9.000
Max. :14.000	Max. :19.00	Max. :193.00	Max. :75.000	Max. :98.000

Table 2: Summary of Season Data Continued

X20.	TD	ATT	RuYD	RuTD
Min. :0.0000	Min. :0.00000	Min. :0.00000	Min. : -26.0000	Min. :0.000000
1st Qu.:0.0000	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.: 0.0000	1st Qu.:0.000000
Median :0.0000	Median :0.00000	Median :0.00000	Median : 0.0000	Median :0.000000
Mean :0.3196	Mean :0.07543	Mean :0.07837	Mean : 0.4502	Mean :0.004409
3rd Qu.:0.0000	3rd Qu.:0.00000	3rd Qu.:0.00000	3rd Qu.: 0.0000	3rd Qu.:0.000000
Max. :9.0000	Max. :3.00000	Max. :8.00000	Max. : 68.0000	Max. :2.000000

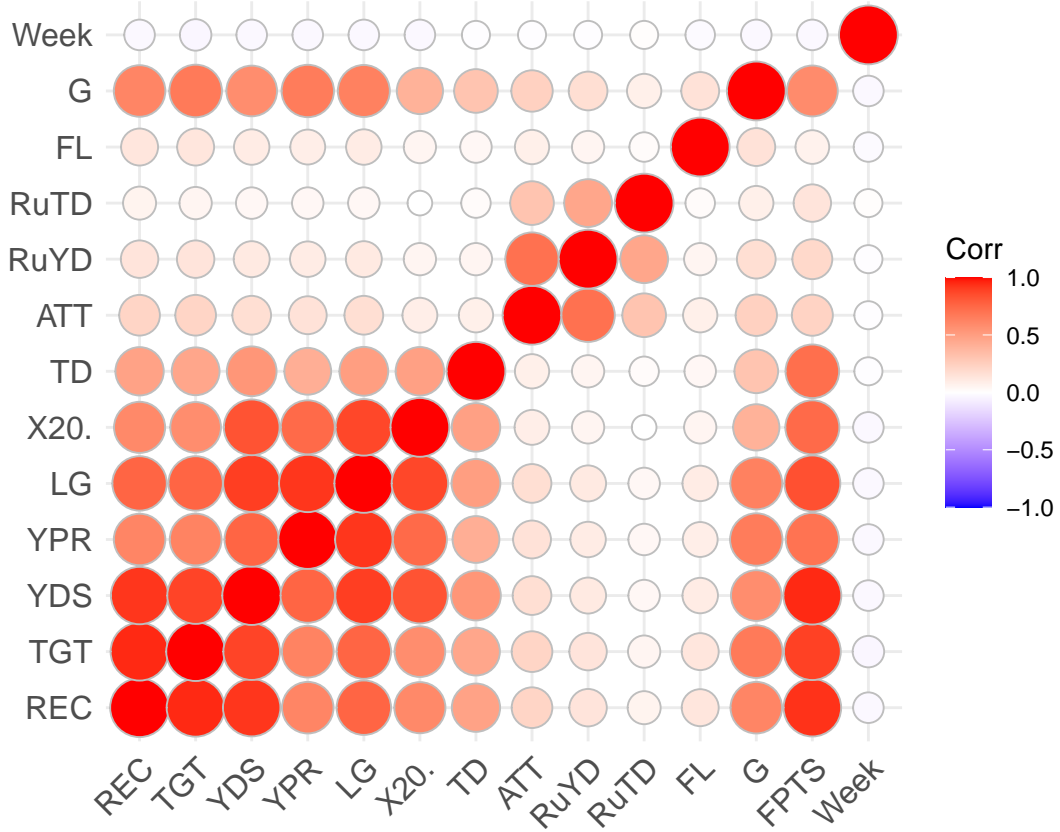
Table 3: Summary of Season Data Continued

FL	G	FPTS	Week
Min. :0.00000	Min. :0.000	Min. : -4.000	Min. : 1.000
1st Qu.:0.00000	1st Qu.:0.000	1st Qu.: 0.000	1st Qu.: 3.000
Median :0.00000	Median :0.000	Median : 0.000	Median : 6.000
Mean :0.01445	Mean :0.399	Mean : 2.939	Mean : 6.494
3rd Qu.:0.00000	3rd Qu.:1.000	3rd Qu.: 2.600	3rd Qu.: 9.000

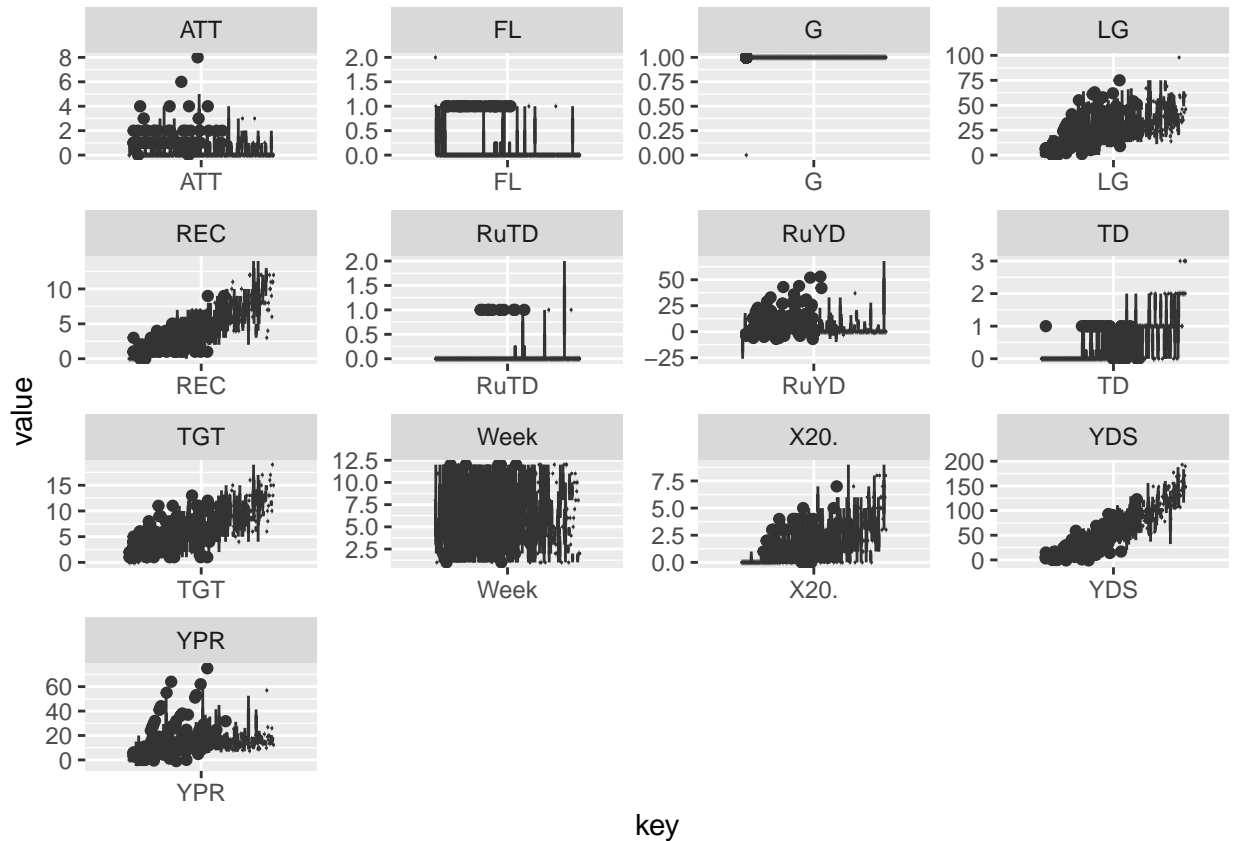
Table 3: Summary of Season Data Continued (*continued*)

FL	G	FPTS	Week
Max. :2.00000	Max. :1.000	Max. :44.800	Max. :12.000

FPTS is expected to be highly correlated with the other variables, so I will show the correlation plot that excludes FPTS.



On the whole, week has no impact on any other variables. G indicates whether a player played or not. If they don't play, then they'll have 0 points for that week, which explains the heavy correlation. I expect X20 (catches of 20 yards or greater) to be heavily correlated with LG (longest catch) and TD (touchdowns). What I am most interested in is the correlation between TGT and the other variables. I would like to be able to confidently predict everything given TGT as the only input.



All of these plots show expected results. I expect that every successive TD will lead to a far higher score, because TDs are the most valuable statistic. TGT is the variable that I am most interested in as a predictor of all others.

## Question 1: Can I predict Expected Fantasy Points Given Targets?

The first question is relatively simple. I will use simple regression to project FPTS given TGT.

FPTS ~ TGT Linear Model

I also considered the following:

TGT ~ . Multiple Regression

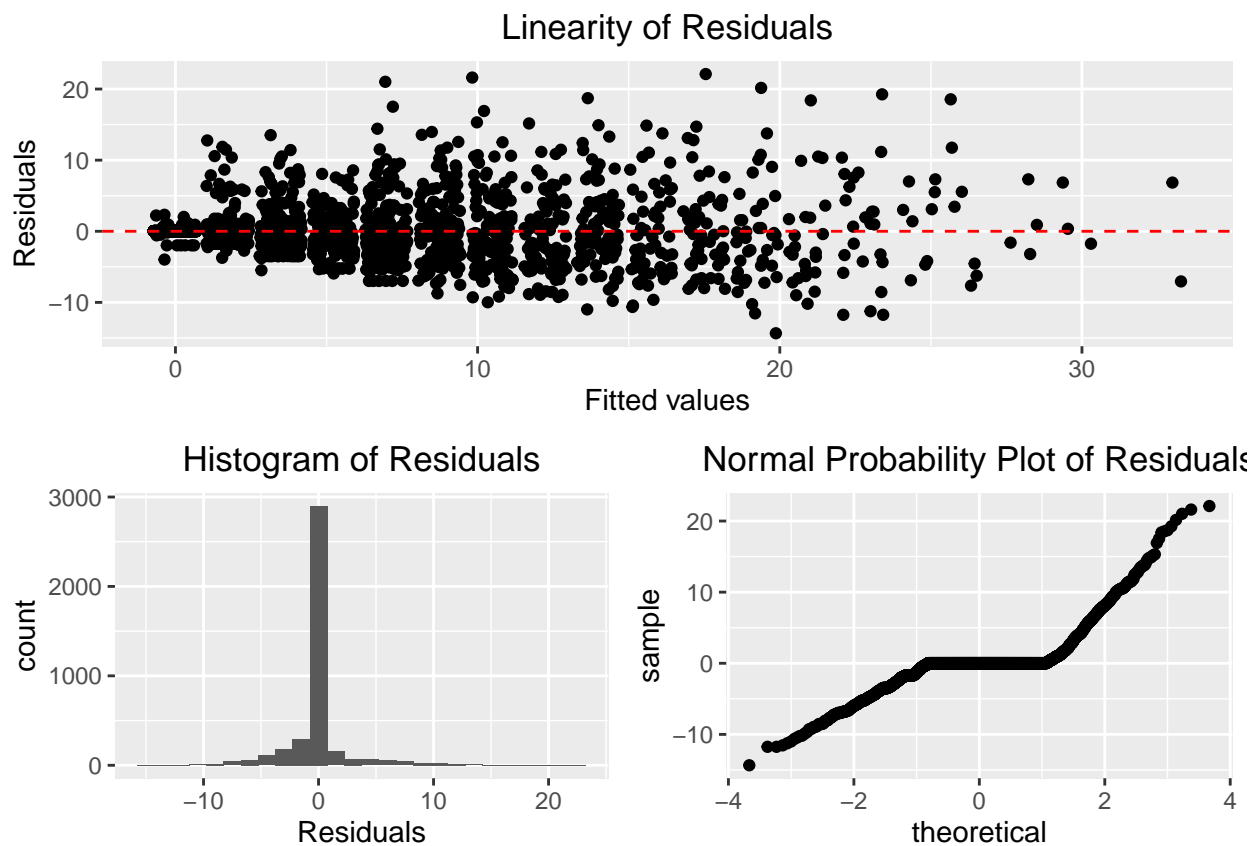
The initial thought was that if I could project TGT from all of the other inputs, I could then reverse the process to compute expected FPTS and other stats. The problem here is that we can't just reverse the regression process to calculate the input variables.

### Model 1: Simple Linear Regression

```
##
## Call:
## lm(formula = FPTS ~ TGT, data = season)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -14.3428    0.0212    0.0212    0.0212    22.1085
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.02122    0.04958  -0.428   0.669
## TGT          1.75128    0.01407 124.512 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.78 on 4081 degrees of freedom
## Multiple R-squared:  0.7916, Adjusted R-squared:  0.7916
## F-statistic: 1.55e+04 on 1 and 4081 DF,  p-value: < 2.2e-16
```

## Validation



The residual values seem to be relatively equally distributed above and below zero.

The histogram of the residuals shows a near normal distribution.

The NPP is moderately linear just below zero, but above zero, the slope is much more substantial.

The model certainly falls off when players outperform expectations. Underperforming players still produce results that are linearly distributed.

## Model 2: XGBoost Linear Regression

Next, I will use xgboost to provide another linear model.

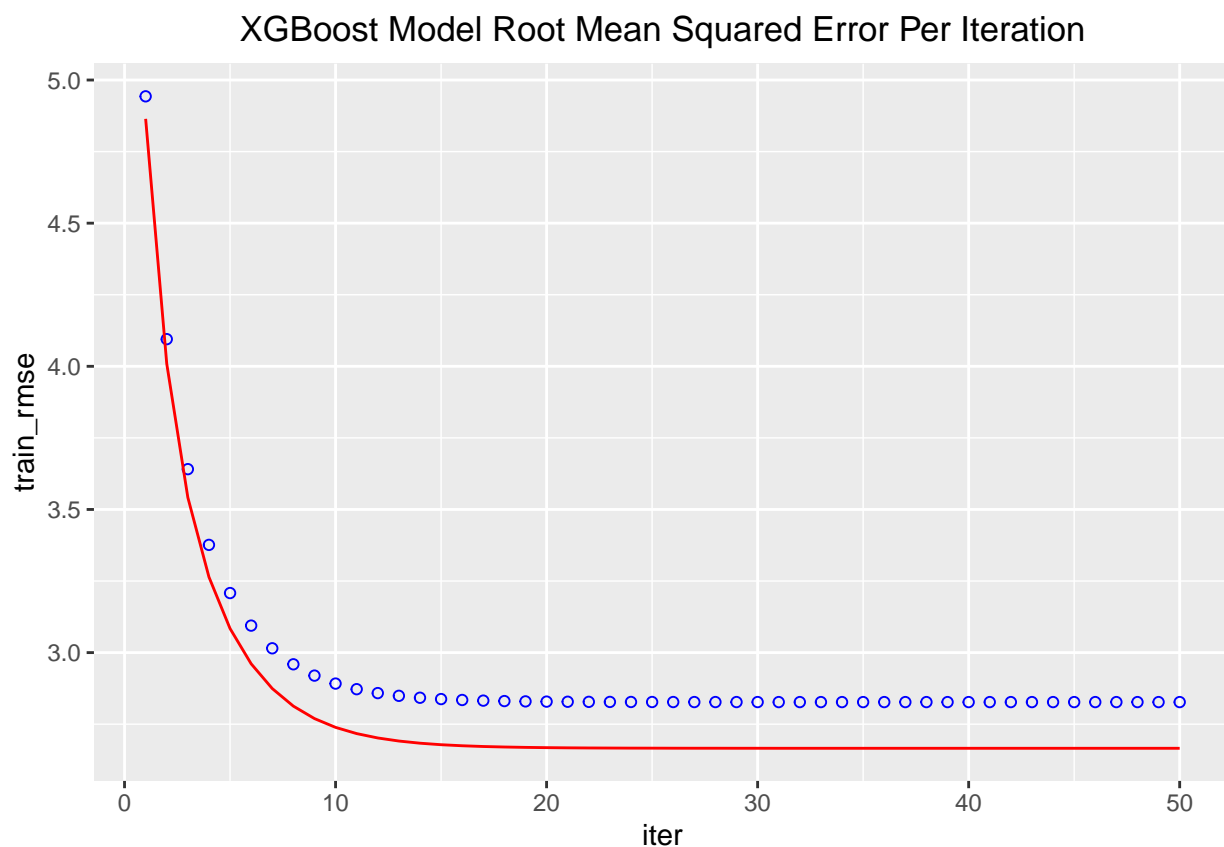


Table 4: XGBoost Model Metrics

XGBoost Model Score	
Mean_Squared_Error	7.106
Root_Mean_Squared_Error	2.666
Test_Mean	2.822
R_Squared	0.754

```
##                                XGBoost Model Score
## Mean_Squared_Error            7.106
## Root_Mean_Squared_Error       2.666
## Test_Mean                     2.822
## R_Squared                     0.754
```

#### Comments

The XGBoost Model is prone to overfitting. The parameter, eta, is the main valve to open or close to limit overfitting. A lower eta value will help prevent overfitting.

## Question 2: Can I Predict Fantasy Points Given Targets and Yards Per Catch as Inputs?

Perhaps we can use expected targets and yards per catch as inputs to predict fantasy points. If we use a player's average yards per catch, combined with the player's expected targets, we may be able to further tune the model.

### Model 3: Multiple Regression

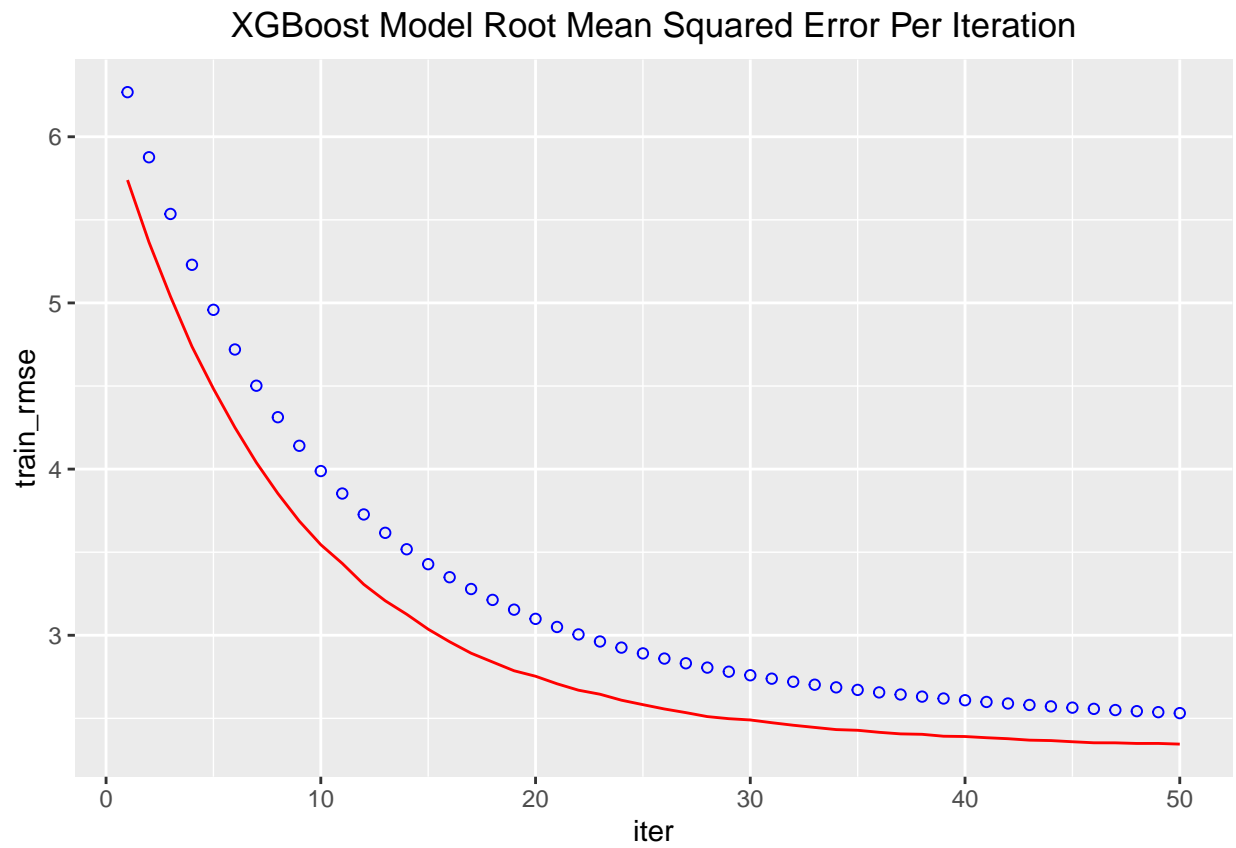
```
##
## Call:
## lm(formula = FPTS ~ TGT + YPR, data = train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.7955  -0.0101   0.3050   0.3050  20.8721
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.30500     0.05640  -5.408 6.92e-08 ***
## TGT          1.43935     0.02016  71.402 < 2e-16 ***
## YPR          0.20944     0.00877  23.882 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.578 on 2855 degrees of freedom
## Multiple R-squared:  0.8217, Adjusted R-squared:  0.8215
## F-statistic: 6577 on 2 and 2855 DF, p-value: < 2.2e-16
```

### Model 4: Negative Binomial

The Negative Binomial Model provides a very poor fit. With an AIC of 6795.4 and predictions that are immensely far from expectation, this model will not be included.



Model 5: XGBoost Regression Model with TGT and YPR as Inputs



TGT

YPR

0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7

Table 5: XGBoost Model Metrics

XGBoost Model Score	
Mean_Squared_Error	5.499
Root_Mean_Squared_Error	2.345
Test_Mean	2.863
R_Squared	0.776

```
##                               XGBoost Model Score
## Mean_Squared_Error           5.499
## Root_Mean_Squared_Error      2.345
## Test_Mean                     2.863
## R_Squared                     0.776
```

### Question 3: Can I Predict Touchdowns Given All Inputs?

Fantasy point totals will not be included in the touchdown prediction. Most experts I listen to often discuss how difficult touchdowns are to predict.

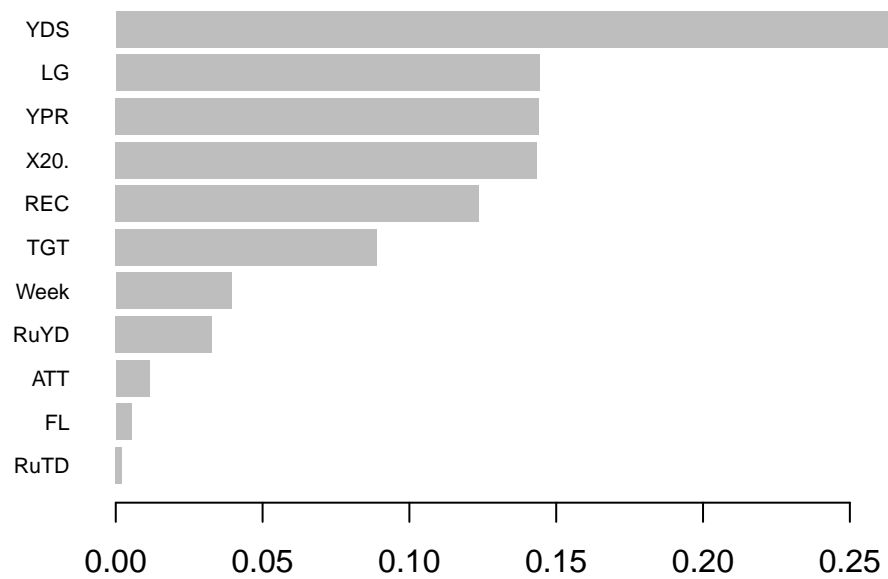
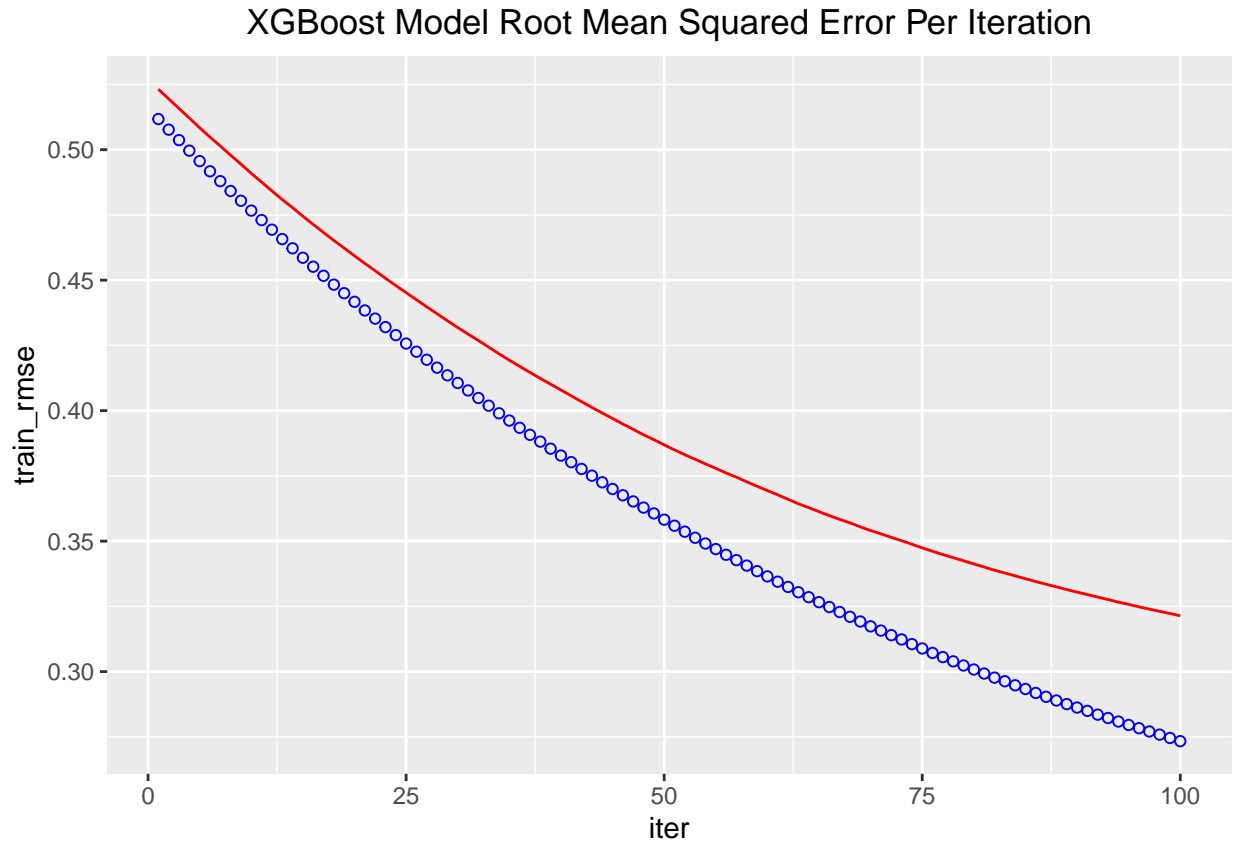
### Model 6: Multiple Regression

For the touchdown prediction model, I will remove any variables with p-values greater than 0.05.

```
##
## Call:
## lm(formula = TD ~ . - Week - YPR - RuTD - G - TGT, data = train_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13907 -0.00096 -0.00096 -0.00096  2.28652
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0009567  0.0056181   0.170  0.864793
## REC          0.0266603  0.0091346   2.919  0.003544 **
## YDS          0.0028303  0.0009973   2.838  0.004574 **
## LG          -0.0007367  0.0010148  -0.726  0.467878
## X20.         0.0576297  0.0148461   3.882  0.000106 ***
## ATT         -0.0114093  0.0178035  -0.641  0.521674
## RuYD        -0.0024746  0.0020155  -1.228  0.219619
## FL           0.0255906  0.0405618   0.631  0.528154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2573 on 2850 degrees of freedom
## Multiple R-squared:  0.3142, Adjusted R-squared:  0.3125
## F-statistic: 186.5 on 7 and 2850 DF,  p-value: < 2.2e-16
```

As expected, the model falls well short of correctly projecting touchdowns. I will now look to predict touchdowns scored with an xgboost model.

## Model 7: XGBoost Touchdowns From All Inputs



The same table is shown twice below due to a bug that I cannot resolve; it is unexpected behavior coming from XGBoost.

Table 6: XGBoost Model Metrics

	XGBoost Model Score
Mean_Squared_Error	0.087
Root_Mean_Squared_Error	0.295
Test_Mean	0.078
R_Squared	-1.159

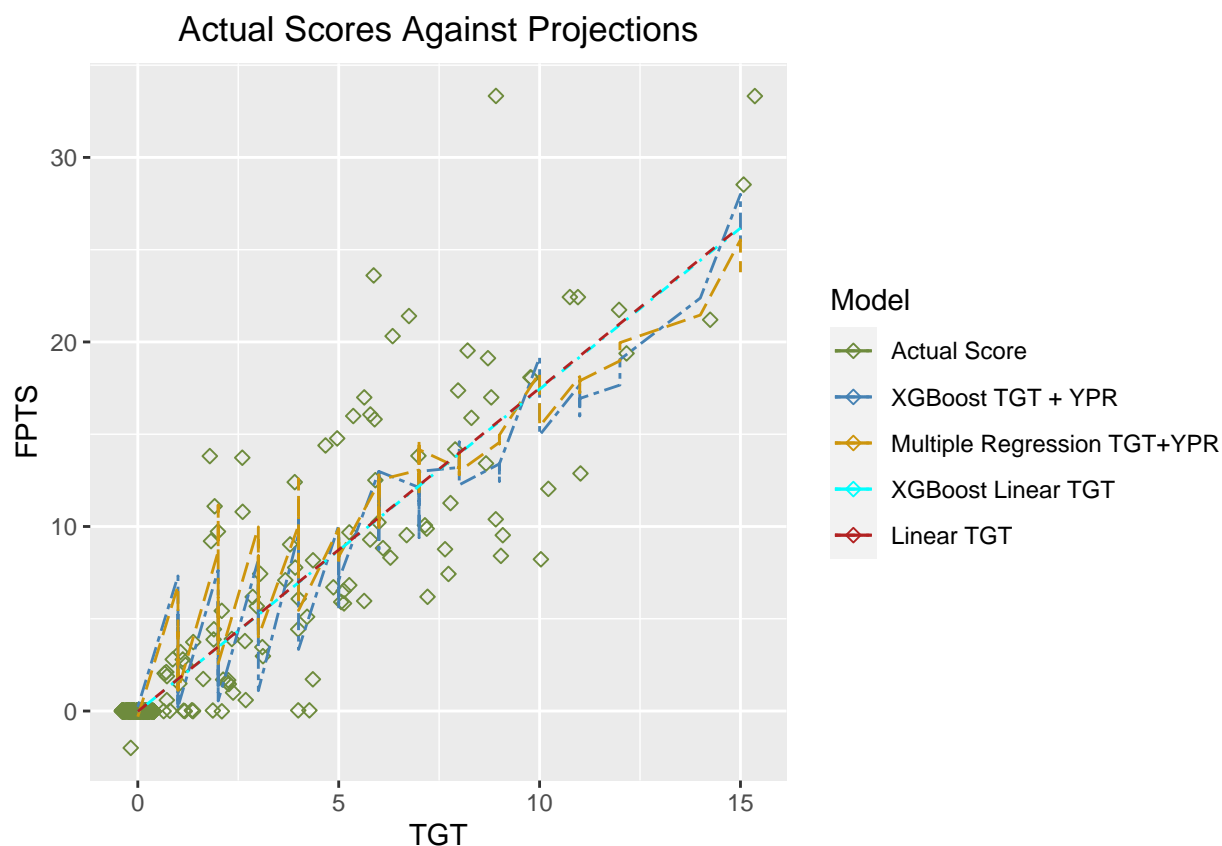
```
##                               XGBoost Model Score
## Mean_Squared_Error           0.087
## Root_Mean_Squared_Error       0.295
## Test_Mean                     0.078
## R_Squared                     -1.159
```

### Comments

Because touchdowns are either 0, 1, 2, 3, (or on very rare occasions 4) I can see why it is difficult to properly fit touchdowns to this model. Touchdowns are effectively random, but it is interesting to note that the XGBoost model was very effective with predicting touchdowns scored given all inputs available. In the knitted document, the seed is different from what it was when the model originally ran, and the  $R^2$  decreased from 0.95 to a negative value.

## Model Selection and Week 14 Predictions

### Week 14 Projections



### Week 14 Performance

Table 7: Week 14 Fantasy Point Prediction Results

	XGBoost TGT + YPR	Mult: TGT + YPR	Linear: TGT	XGBoost TGT
RMSE	2.313	2.372	2.524	2.525

### Comments

Overall, in Week 14, the models that performed the best were the XGBoost multiple regression and the standard multiple regression model that took into account expected yards per reception and targets. All four of the final models resulted in similar predictions, and they all were remarkably accurate. I have exported the Week 14 projections and results in a .csv file.

## Sources:

“Fantasy Football Scoring Leaders.” ESPN, ESPN Internet Ventures, <https://fantasy.espn.com/football/leaders?lineupSlot=4&scoringPeriodId=13&statSplit=singleScoringPeriod>.

“Week 14 NFL WR Statistics.” FantasyPros, <https://www.fantasypros.com/nfl/stats/wr.php?range=week&week=14>.

## Appendix: R Code

```
# Seed was set as 34.
knitr::opts_chunk$set(cache = TRUE)
library(tidyverse)
library(xgboost)
library(Matrix)
library(kableExtra)
library(ggcorrplot)
library(yardstick)
library(caret)
library(MASS)
library(pROC)
library(ggpubr)
library(AICcmodavg)
library(ggpubr)

# Introduction

wk1 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR.csv')
wk1 <- rename(wk1, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk1$Week <- 1

wk2 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-2.csv')
wk2 <- rename(wk2, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk2$Week <- 2

wk3 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-3.csv')
wk3 <- rename(wk3, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk3$Week <- 3

wk4 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-4.csv')
wk4 <- rename(wk4, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk4$Week <- 4

wk5 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-5.csv')
wk5 <- rename(wk5, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk5$Week <- 5
```

```

wk6 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-6.csv')
wk6 <- rename(wk6, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk6$Week <- 6

wk7 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-7.csv')
wk7 <- rename(wk7, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk7$Week <- 7

wk8 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-8.csv')
wk8 <- rename(wk8, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk8$Week <- 8

wk9 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-9.csv')
wk9 <- rename(wk9, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk9$Week <- 9

wk10 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-10.csv')
wk10 <- rename(wk10, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk10$Week <- 10

wk11 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-11.csv')
wk11 <- rename(wk11, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk11$Week <- 11

wk12 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-12.csv')
wk12 <- rename(wk12, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk12$Week <- 12

wk13 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-13.csv')
wk13 <- rename(wk13, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")
wk12$Week <- 12

season <- rbind(wk1,wk2,wk3,wk4,wk5,wk6,wk7,wk8,wk9,wk10,wk11,wk12)

wk14 <- read.csv('FantasyPros_Fantasy_Football_Statistics_WR-14.csv')
wk14 <- rename(wk14, "YPR" = "Y.R", "RuYD" = "YDS.1", "RuTD" = "TD.1")

# Data Preparation

season <- season[,-c(1,2,16,17)]

#summary(season)

sum_season <- summary(season)

sum_season_1 <- sum_season[, 1:5]

sum_season_2 <- sum_season[,6:10]

```

```

sum_season_3 <- sum_season[,11:14]

kbl(sum_season_1, longtable = T, booktabs = T, caption = "Summary of Season Data") %>%
  kable_styling(latex_options = c("repeat_header"))

kbl(sum_season_2, longtable = T, booktabs = T, caption = "Summary of Season Data Continued") %>%
  kable_styling(latex_options = c("repeat_header"))

kbl(sum_season_3, longtable = T, booktabs = T, caption = "Summary of Season Data Continued") %>%
  kable_styling(latex_options = c("repeat_header"))

sample_rows <- sample(nrow(season), nrow(season) *.7)

dt <- sort(sample_rows)
test <- season[-dt,]
train_set <- season[dt,]

corr <- data.frame(round(cor(season), 3))

cp1 <- ggcorrplot(corr, method = "circle")

corr2 <- data.frame(round(cor(season), 3))

corr2 <- corr2[, -13]

cp2 <- ggcorrplot(corr2, method = "circle")

cp2
#ggarrange(cp1, cp2, nrow=2, ncol=1)

season %>%
  gather(key, value, -c(FPTS)) %>%
  mutate(key = factor(key), target = factor(FPTS)) %>%
  ggplot(aes(x = key, y = value)) +
  geom_boxplot(aes(fill = target)) +
  facet_wrap(~key, scales = "free", ncol = 4) +
  theme(legend.position = "none")

# Question 1: Can I predict Expected Fantasy Points Given Targets?

## Model 1: Simple Linear Regression

sample_rows <- sample(nrow(season), nrow(season) *.7)

dt <- sort(sample_rows)
test_set <- season[-dt,]
train_set <- season[dt,]

mod_1 <- lm(FPTS ~ TGT, data=season)

```



```

summary(mod_1)

test_set$mod_1 <- predict(mod_1,test_set)

predictions <- data.frame(predict(mod_1,test_set))

test_mean_mod_1 <- mean(predict(mod_1,test_set))

sct <- ggplot(data = mod_1, aes(x = .fitted, y = .resid)) +
  geom_jitter() +
  geom_hline(yintercept = 0, linetype = "dashed", color = 'red') +
  xlab("Fitted values") +
  ylab("Residuals") +
  ggtitle("Linearity of Residuals")+
  theme(plot.title = element_text(hjust = 0.5))

hst <- ggplot(data = mod_1, aes(x = .resid)) +
  geom_histogram(binwidth = 1.5) +
  xlab("Residuals") +
  ggtitle("Histogram of Residuals")+
  theme(plot.title = element_text(hjust = 0.5))

npp <- ggplot(data = mod_1, aes(sample = .resid)) +
  stat_qq()+
  ggtitle("Normal Probability Plot of Residuals")+
  theme(plot.title = element_text(hjust = 0.5))

plt <- ggarrange(sct,ggarrange(hst,npp,nrow=1,ncol=2),
  nrow = 2, ncol = 1)
plt

## Model 2: XGBoost Linear Regression

sample_rows <- sample(nrow(season),nrow(season) *.7)

dt <- sort(sample_rows)
test_set_xgb <- season[-dt,c(2,13)]
train_set <- season[dt,c(2,13)]

train_x_mat <- data.matrix(train_set[,1]) #xgboost requires a matrix as the input
train_y <- train_set[,2] #Response Variable

test_x_mat <- data.matrix(test_set_xgb[,1])
test_y <- test_set_xgb[,2] #Response Variable

xgb_train <- xgb.DMatrix(data = train_x_mat,label=train_y)
xgb_test <- xgb.DMatrix(data = test_x_mat,label=test_y)

watchlist <- list(train = xgb_train,test = xgb_test)

xgb_model <- xgb.train("booster" = "gblinear",data=xgb_train,watchlist=watchlist,nrounds=50,verbose=0,e

```

```

xgb_lin_mod <- xgb_model
#summary(xgb_model)

e <- data.frame(xgb_model$evaluation_log)

#which(e$test_rmse == min(e$test_rmse))

ggplot(e, aes(iter,train_rmse))+
  geom_point(color = 'blue',shape = 1)+
  geom_line(data = e, aes(x=iter,y=test_rmse),color = 'red')+
  ggtitle("XGBoost Model Root Mean Squared Error Per Iteration")+
  theme(plot.title = element_text(hjust = 0.5))

#imp <- xgb.importance(colnames(xgb_train),model = xgb_model)
#xgb.plot.importance(imp)

xgb_pred <- predict(xgb_model,xgb_test)
mse_xgb <- mean((test_y - xgb_pred)^2)
rmse_xgb <- RMSE(test_y,xgb_pred)

xgb_residuals <- data.frame(xgb_pred - test_y)
test_mean <- mean(xgb_pred)
tss <- sum((xgb_pred - test_mean)^2)
rss <- sum(xgb_residuals^2)
R2_xgb <- 1 - (rss/tss)

test_set_xgb$xgb_Pred <- xgb_pred
xgb_stats_tuned <- data.frame(rbind(mse_xgb,rmse_xgb,test_mean,R2_xgb))
colnames(xgb_stats_tuned) <- ("XGBoost Model Score")
xgb_stats_tuned$`XGBoost Model Score` <- round(xgb_stats_tuned$`XGBoost Model Score`,3)
rownames(xgb_stats_tuned) <- c("Mean_Squared_Error","Root_Mean_Squared_Error",
                             "Test_Mean","R_Squared")

kbl(xgb_stats_tuned, longtable = T, booktabs = T, caption = "XGBoost Model Metrics") %>%
  kable_styling(latex_options = c("repeat_header"))

xgb_stats_tuned

predictions <- cbind(predictions,xgb_pred)

# Question 2: Can I Predict Fantasy Points Given Targets and Yards Per Catch as Inputs?

## Model 3: Multiple Regression

season$FPTS[which(season$FPTS < 0)] <- 0

dt <- sort(sample_rows)
test_set <- season[-dt,]
train_set <- season[dt,]

mult_1 <- lm(FPTS ~ TGT + YPR, data = train_set)

```

```

summary(mult_1)

test_set$mult_1 <- predict(mult_1,test_set)

predictions <- cbind(predictions,predict(mult_1,test_set))

test_mean_mult_1 <- mean(predict(mult_1,test_set))

## Model 4: Negative Binomial

season$FPTS[which(season$FPTS < 0)] <- 0

dt <- sort(sample_rows)
test_set <- season[-dt,]
train_set <- season[dt,]

nbinom_1 <- glm.nb(FPTS ~ TGT + YPR, data = train_set)
#summary(nbinom_1)

test_set$nbinom_1 <- predict(nbinom_1,test_set)

predictions <- cbind(predictions,predict(nbinom_1,test_set))

### Model 5: XGBoost Regression Model with TGT and YPR as Inputs

sample_rows <- sample(nrow(season),nrow(season) *.7)

dt <- sort(sample_rows)
test_set_xgb <- season[-dt,c(2,4,13)]
train_set <- season[dt,c(2,4,13)]

train_x_mat <- data.matrix(train_set[, -3]) #xgboost requires a matrix as the input
train_y <- train_set[,3] #Response Variable

test_x_mat <- data.matrix(test_set_xgb[, -3])
test_y <- test_set_xgb[,3] #Response Variable

xgb_train <- xgb.DMatrix(data = train_x_mat,label=train_y)
xgb_test <- xgb.DMatrix(data = test_x_mat,label=test_y)

watchlist <- list(train = xgb_train,test = xgb_test)

xgb_model <- xgb.train(data=xgb_train,max_depth = 1,watchlist=watchlist,nrounds=50,verbose=0,eta = 0.1)
xgb_model_2 <- xgb_model
#summary(xgb_model)

e <- data.frame(xgb_model$evaluation_log)

#which(e$test_rmse == min(e$test_rmse))

ggplot(e, aes(iter,train_rmse))+

```

```

geom_point(color = 'blue',shape = 1)+
geom_line(data = e, aes(x=iter,y=test_rmse),color = 'red')+
ggtitle("XGBoost Model Root Mean Squared Error Per Iteration")+
theme(plot.title = element_text(hjust = 0.5))

imp <- xgb.importance(colnames(xgb_train),model = xgb_model)
xgb.plot.importance(imp)

xgb_pred <- predict(xgb_model,xgb_test)
mse_xgb <- mean((test_y - xgb_pred)^2)
rmse_xgb <- RMSE(test_y,xgb_pred)

xgb_residuals <- data.frame(xgb_pred - test_y)
test_mean <- mean(xgb_pred)
tss <- sum((xgb_pred - test_mean)^2)
rss <- sum(xgb_residuals^2)
R2_xgb <- 1 - (rss/tss)

test_set_xgb$xgb_Pred <- xgb_pred
xgb_stats_tuned <- data.frame(rbind(mse_xgb,rmse_xgb,test_mean,R2_xgb))
colnames(xgb_stats_tuned) <- ("XGBoost Model Score")
xgb_stats_tuned$`XGBoost Model Score` <- round(xgb_stats_tuned$`XGBoost Model Score`,3)
rownames(xgb_stats_tuned) <- c("Mean_Squared_Error","Root_Mean_Squared_Error",
                             "Test_Mean","R_Squared")

kbl(xgb_stats_tuned, longtable = T, booktabs = T, caption = "XGBoost Model Metrics") %>%
  kable_styling(latex_options = c("repeat_header"))

xgb_stats_tuned

predictions <- cbind(predictions,xgb_pred)

# Question 3: Can I Predict Touchdowns Given All Inputs?

## Model 6: Multiple Regression

season$FPTS[which(season$FPTS < 0)] <- 0

season_td <- season[,-13]

dt <- sort(sample_rows)
test_set <- season_td[-dt,]
train_set <- season_td[dt,]

mult_td <- lm(TD ~ . - Week - YPR - RuTD - G - TGT, data = train_set)
summary(mult_td)

test_set$mult_td <- predict(mult_td,test_set)

pred_td <- data.frame(predict(mult_td,test_set))

```

```

test_mean_mult_td <- mean(predict(mult_td, test_set))

## Model 7: XGBoost Touchdowns From All Inputs

# Grid Search Algorithm for Optimizing XGBoost Models

searchGridSubCol <- expand.grid(subsample = c(0.5, 0.6),
                                colsample_bytree = c(0.5, 0.6),
                                max_depth = c(3, 4),
                                min_child = seq(1),
                                eta = c(0.1)
)

ntrees <- 100

system.time(
rmseErrorsHyperparameters <- apply(searchGridSubCol, 1, function(parameterList){

  #Extract Parameters to test
  currentSubsampleRate <- parameterList[["subsample"]]
  currentColsampleRate <- parameterList[["colsample_bytree"]]
  currentDepth <- parameterList[["max_depth"]]
  currentEta <- parameterList[["eta"]]
  currentMinChild <- parameterList[["min_child"]]
  xgboostModelCV <- xgb.cv(data = xgb_train, nrounds = ntrees, nfold = 5, showsd = TRUE,
                           metrics = "rmse", verbose = TRUE, "eval_metric" = "rmse",
                           "objective" = "reg:squarederror", "max.depth" = currentDepth, "eta" = currentEta,
                           "subsample" = currentSubsampleRate, "colsample_bytree" = currentColsampleRate
                           , print_every_n = 10, "min_child_weight" = currentMinChild, booster = "gbtree",
                           early_stopping_rounds = 10)

  xvalidationScores <- as.data.frame(xgboostModelCV$evaluation_log)
  rmse <- tail(xvalidationScores$test_rmse_mean, 1)
  trmse <- tail(xvalidationScores$train_rmse_mean, 1)
  output <- return(c(rmse, trmse, currentSubsampleRate, currentColsampleRate, currentDepth, currentEta,

output <- as.data.frame(t(rmseErrorsHyperparameters))
varnames <- c("TestRMSE", "TrainRMSE", "SubSampRate", "ColSampRate", "Depth", "eta", "currentMinChild")
names(output) <- varnames
#head(output)

sample_rows <- sample(nrow(season), nrow(season) *.7)

dt <- sort(sample_rows)
test_set_xgb <- season[-dt, -13]
train_set <- season[dt, -13]

train_x_mat <- data.matrix(train_set[, -7]) #col 7 is the target variable
train_y <- train_set[, 7] #Response Variable

test_x_mat <- data.matrix(test_set_xgb[, -7])
test_y <- test_set_xgb[, 7] #Response Variable

```

```

xgb_train <- xgb.DMatrix(data = train_x_mat,label=train_y)
xgb_test  <- xgb.DMatrix(data = test_x_mat,label=test_y)

watchlist <- list(train = xgb_train,test = xgb_test)

xgb_model <- xgb.train(data=xgb_train,max_depth = 6,subsample = 0.5,colsample_bytree = 0.5,watchlist=wa

#summary(xgb_model)

e <- data.frame(xgb_model$evaluation_log)

#which(e$test_rmse == min(e$test_rmse))

ggplot(e, aes(iter,train_rmse))+
  geom_point(color = 'blue',shape = 1)+
  geom_line(data = e, aes(x=iter,y=test_rmse),color = 'red')+
  ggtitle("XGBoost Model Root Mean Squared Error Per Iteration")+
  theme(plot.title = element_text(hjust = 0.5))

imp <- xgb.importance(colnames(xgb_train),model = xgb_model)
xgb.plot.importance(imp)

xgb_pred <- predict(xgb_model,xgb_test)
mse_xgb <- mean((test_y - xgb_pred)^2)
rmse_xgb <- RMSE(test_y,xgb_pred)

xgb_residuals <- data.frame(xgb_pred - test_y)
test_mean <- mean(xgb_pred)
tss <- sum((xgb_pred - test_mean)^2)
rss <- sum(xgb_residuals^2)
R2_xgb <- 1 - (rss/tss)

test_set_xgb$xgb_Pred <- xgb_pred
xgb_stats_tuned <- data.frame(rbind(mse_xgb,rmse_xgb,test_mean,R2_xgb))
colnames(xgb_stats_tuned) <- ("XGBoost Model Score")
xgb_stats_tuned$`XGBoost Model Score` <- round(xgb_stats_tuned$`XGBoost Model Score`,3)
rownames(xgb_stats_tuned) <- c("Mean_Squared_Error","Root_Mean_Squared_Error",
                              "Test_Mean","R_Squared")

kbl(xgb_stats_tuned, longtable = T, booktabs = T, caption = "XGBoost Model Metrics") %>%
  kable_styling(latex_options = c("repeat_header"))

xgb_stats_tuned

pred_td <- cbind(pred_td,xgb_pred)

# Model Selection and Week 14 Predictions

#Fit_Summary <- data.frame(cbind(confusion1$byClass[7],confusion2$byClass[7],confusion3$byClass[7],conf

R2 <- data.frame(cbind(summary(mod_1)$adj.r.squared,summary(mult_1)$adj.r.squared))

```

```

colnames(R2) <- c("Lin_Model","Multiple_Regression_1")

mse <- function(model){
  mean(model$residuals^2)
}

rmse <- function(model){
  sqrt(mean(model$residuals^2))
}

MSE <- data.frame(cbind(mse(summary(mod_1)),mse(summary(mult_1))))
RMSE <- data.frame(cbind(rmse(summary(mod_1)),rmse(summary(mult_1))))

#rownames(Fit_Summary) <- c("F1-Score", "AIC","AUC","Sensitivity","Specificity","Precision","Balanced A

#kbl(Fit_Summary, longtable = T, booktabs = T, caption = "Summary of Models") %>%
# kable_styling(latex_options = c("repeat_header"))

### Week 14 Projections

wk14 <- wk14[,-c(1,16,17)]
wk14_mat <- data.matrix(wk14[,c(3,5)])
test_y <- wk14[,13] #Response Variable

xgb_test_wk14 <- xgb.DMatrix(data = wk14_mat,label=test_y)

wk14$xgb_model_pred <- predict(xgb_model_2,xgb_test_wk14)

wk14$multiple_reg_model_pred <- predict(mult_1,wk14)

wk14$lin_tgt_only <- predict(mod_1,wk14)

wk14_mat_tgt <- data.matrix(wk14[,c(3)])
xgb_tgt_wk14 <- xgb.DMatrix(data = wk14_mat_tgt,label=test_y)
wk14$xgb_lin_tgt_only <- predict(xgb_lin_mod,xgb_tgt_wk14)

ggplot(wk14, aes(TGT))+
  geom_jitter(aes(y=FPTS,color='Actual Score'),shape = 5)+
  geom_line(aes(y=xgb_model_pred,color="XGBoost TGT + YPR"),linetype="twodash")+
  geom_line(aes(y=multiple_reg_model_pred,color="Multiple Regression TGT+YPR"),linetype="longdash")+
  geom_line(aes(y=xgb_lin_tgt_only,color="XGBoost Linear TGT"),linetype="dotdash")+
  geom_line(aes(y=lin_tgt_only,color="Linear TGT"),linetype="dashed")+
  ggtitle("Actual Scores Against Projections")+
  scale_color_manual(name = 'Model', breaks = c("Actual Score", "XGBoost TGT + YPR", "Multiple Regression
                    values =c('Actual Score'='darkolivegreen4', 'XGBoost TGT + YPR'='steelblue', 'Mult
  theme(plot.title = element_text(hjust = 0.5))

# Week 14 Performance

rmse_eval <- data.frame(round(RMSE(wk14$FPTS,wk14$xgb_model_pred),3))

```

```

rmse_eval <- cbind(rmse_eval,data.frame(round(RMSE(wk14$FPTS,wk14$multiple_reg_model_pred),3)))
rmse_eval <- cbind(rmse_eval,data.frame(round(RMSE(wk14$FPTS,wk14$lin_tgt_only),3)))
rmse_eval <- cbind(rmse_eval,data.frame(round(RMSE(wk14$FPTS,wk14$xgb_lin_tgt_only),3)))

colnames(rmse_eval) <- c("XGBoost TGT + YPR","Mult: TGT + YPR","Linear: TGT","XGBoost TGT")
rownames(rmse_eval) <- c("RMSE")

kbl(rmse_eval, longtable = T, booktabs = T, caption = "Week 14 Fantasy Point Prediction Results") %>%
  kable_styling(latex_options = c("repeat_header"))

write.csv(wk14, "DATA_621_Final_Wk14_Predictions.csv",row.names = FALSE)
write.csv(season,"DATA_621_Final_Season_Statistics.csv",row.names=FALSE)

```