Hao-ProjectFinal

Introduction

In the context of Daily Fantasy Football, are expert projections for individual players on a given week more accurate than simple historical season averages?

Fantasy sports is already immensely popular, and daily fantasy sports is a fast growing subset. An entire ecosystem of players, betters, prognosticators, bloggers, etc. has developed around real and fantasy sports. Furthermore, the substantial availability of statistical sports data has also led to fantasy sports quants entering the mainstream.

To answer this question, I will be using the www.draftkings.com scoring system for daily fantasy football. Basically, the scoring system converts player and team statistics from the actual football game into a normalized numeric score. For example, a player might earn 6 points for scoring a touchdown.

Data:

- Data collection: The site www.fantasyfootballanalytics.com compiles projections for each player for each week from various expert forecasters, e.g. ESPN, CBS Sports, etc. The site www.rotoguru.net then compiles actual points scored for each player for each week
- Cases: Each case is a player, the week in question, the points actually scored, the points projected to be scored and the historical average number of points
- Variables: The variables explored will be actual points scored vs. projected points and vs. historical average points - all of which are numeric variables; I may also take a look at how these variables compare over the course of time, measured in weeks in this case
- Type of study: This is clearly an observational study
- Scope of inference - generalizability: The sample is comprised of all football players from week 1 to week 13 in the 2016 season; while incorporating additional seasons would make the sample more representative of the entire population of football players across all seasons, this sample should be representative of football players within the current regime of football
- Scope of inference - causality: As an observational study, no causal conclusions can be made from this study

```
library(dplyr)
library(tidyr)
library(stringr)
library(ggplot2)

setwd("~/Google Drive/CUNY/git/DATA606/Project")
pathName = "~/Google Drive/CUNY/git/DATA606/Project"
```

Import and Cleanse Data

I downloaded data into text files from 1) www.fantasyfootballanalytics.com for projected points and 2) www.rotoguru.net for actual points. The data required a fair amount of cleansing in order to combine them mainly because they used different naming conventions. The final output of this phase is a single data frame

containing actual points and projected points for each player by week. In addition, I will add a column for the running average number of points for each player to serve as the baseline against which to compare the expert projections.

Note: Even after normalizing the vast majority of the names between the two data sets, there still remained about 30 names that show up on the actual points data that do not match names on the projected data. We could spend more time investigating the exact causes of the disconnect, but given the size of the data set, the omission should not be material.

```
# read in projected points data
fileNames = list.files(paste0(pathName, '/FFA_Data'))
weekNum = as.numeric(str_extract(fileNames[1], "\\d+"))
projectedPoints = read.csv(paste0(pathName, '/FFA_Data/', fileNames[1]),
                           header = T, stringsAsFactors = F, row.names = NULL)
projectedPoints$week = weekNum
for (i in 2:length(list.files(paste0(pathName, '/FFA_Data/')))) {
  weekNum = as.numeric(str_extract(fileNames[i], "\\d+"))
  tmp = read.csv(paste0(pathName, '/FFA_Data/', fileNames[i]),
                             header = T, stringsAsFactors = F, row.names = NULL)
  tmp$week = weekNum
  projectedPoints = bind_rows(projectedPoints, tmp)
}
# normalize naming conventions
projectedPoints$playername = str_replace_all(projectedPoints$playername, "\\.", "")
projectedPoints$playername = str_replace_all(projectedPoints$playername, "[:alpha:]+\\-", "")



# read in actual points data
actualPoints = read.delim(paste0(pathName, '/RotoGuru_Data/', list.files(paste0(pathName, '/RotoGuru_Da
                          sep = ';', stringsAsFactors = F)
for (i in 2:length(list.files(paste0(pathName, '/RotoGuru_Data')))) {
  tmp = read.delim(paste0(pathName, '/RotoGuru_Data/', list.files(paste0(pathName, '/RotoGuru_Data'))[i
                            sep = ';', stringsAsFactors = F)
  actualPoints = bind_rows(actualPoints, tmp)
}

# normalize actual data to match projected data
colnames(actualPoints) = str_to_lower(colnames(actualPoints))
# separate DST from other positions
actualPointsDst = actualPoints[actualPoints$pos=='Def',]
actualPointsOff = actualPoints[actualPoints$pos!='Def',]

# Offense
actualPointsOff$playername = str_replace_all(actualPointsOff$name, "\\'", '')
actualPointsOff$playername = str_replace_all(actualPointsOff$playername, "\\b(\\sJr\\.)|\\b(\\sSr\\.)|\
actualPointsOff$playername = str_replace_all(actualPointsOff$playername, "\\.", '')
actualPointsOff$playername = paste(str_extract(actualPointsOff$playername, '\\,\\s[:alpha:].+'),
                           str_extract(actualPointsOff$playername, '[:alpha:]+\\,'))
actualPointsOff$playername = str_trim(str_replace_all(actualPointsOff$playername, '\\,', ''))

# DST
actualPointsDst$team = str_to_upper(actualPointsDst$team)
actualPointsDst$team = str_replace(actualPointsDst$team, 'SDG', 'SD')
actualPointsDst$team = str_replace(actualPointsDst$team, 'GNB', 'GB')
```

```
actualPointsDst$team = str_replace(actualPointsDst$team, 'NWE', 'NE')
actualPointsDst$team = str_replace(actualPointsDst$team, 'LAR', 'LA')
lookup = unique(left_join(projectedPoints[projectedPoints$position=='DST',],
                  actualPointsDst[actualPointsDst$pos=='Def',], by = 'team')[, c('team', 'playername')])
actualPointsDst = left_join(actualPointsDst, lookup, by = 'team')

actualPoints = bind_rows(actualPointsOff, actualPointsDst)



# create combined data frame
df = actualPoints %>%
  filter(!is.na(playername)) %>%
  left_join(projectedPoints, by = c('playername', 'week')) %>%
  select(playername, pos, week, dk.points, points) %>%
  rename(actual_points = dk.points, projected_points = points)

# remove incomplete cases
df = df[complete.cases(df),]

# add running average number of points
for (i in 1:nrow(df)) {
  if (df[i, 'week'] == 1) {
    df[i, 'avg_points'] = NA
  } else {
  df[i, 'avg_points'] = df %>%
    filter(playername == df[i, 'playername'], week < df[i, 'week']) %>%
    summarise(mean(actual_points, na.rm = T))
  }
}
```

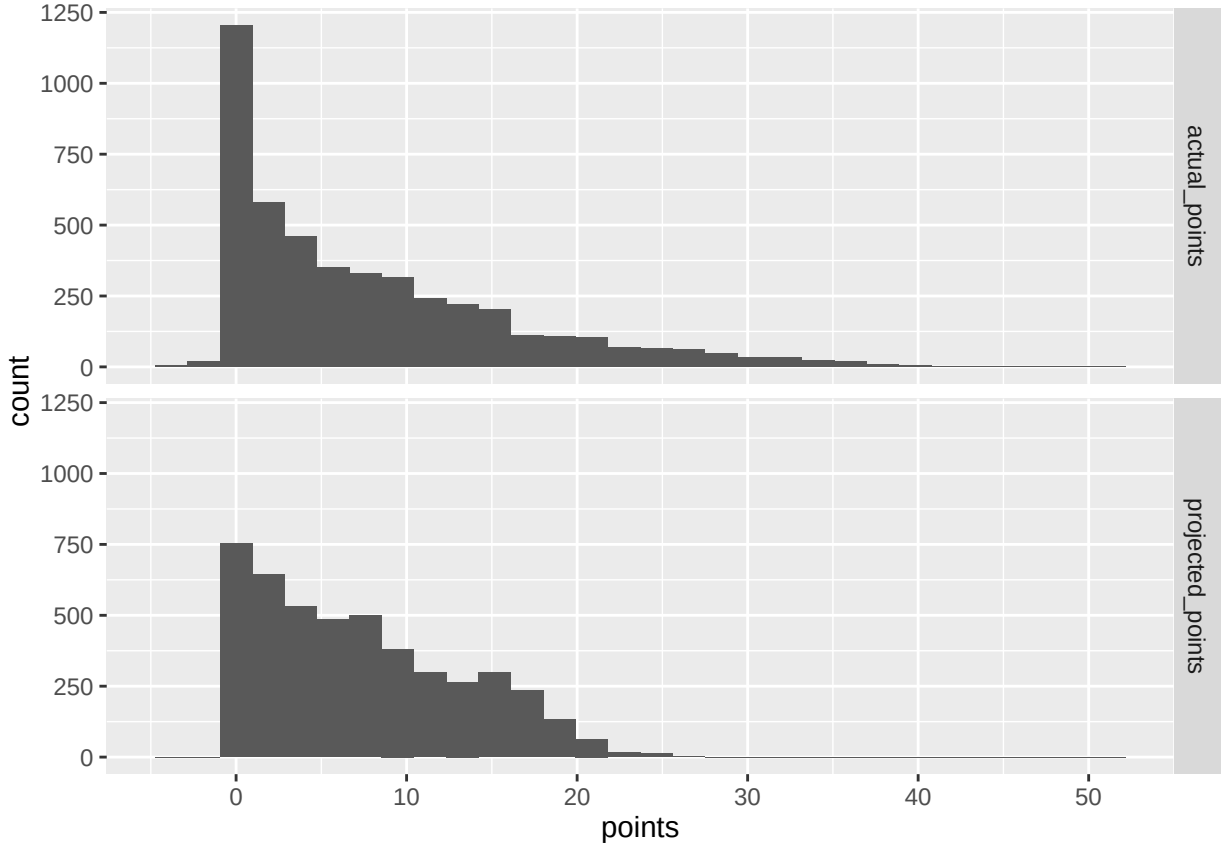
Exploratory Data Analysis

```
summary(df)
```

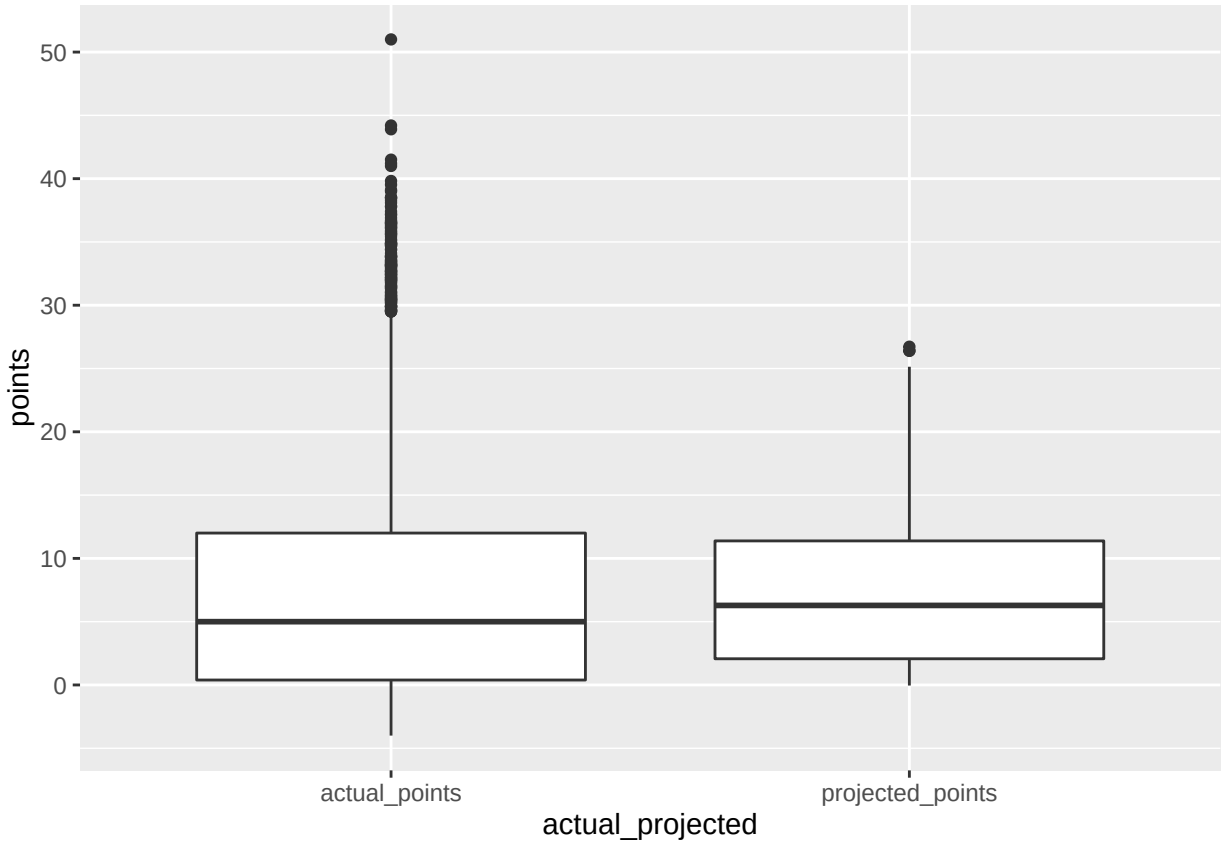
```
##   playername            pos                 week        actual_points
##  Length:4619        Length:4619        Min.   : 1.000   Min.   :-4.000
##  Class :character   Class :character   1st Qu.: 3.000   1st Qu.: 0.390
##  Mode  :character   Mode  :character   Median : 6.000   Median : 5.000
##                                        Mean   : 6.627   Mean   : 7.692
##                                        3rd Qu.:10.000   3rd Qu.:12.000
##                                        Max.   :13.000   Max.   :51.000
##
##  projected_points    avg_points
##  Min.   :-0.0455   Min.   :-3.000
##  1st Qu.: 2.0702   1st Qu.: 1.850
##  Median : 6.2820   Median : 6.050
##  Mean   : 7.2792   Mean   : 7.709
##  3rd Qu.:11.3819   3rd Qu.:12.333
##  Max.   :26.7247   Max.   :39.000
##                    NA's   :556
```

```
df %>% select(playername, actual_points, projected_points) %>%
  gather('actual_projected', 'points', -1) %>%
  ggplot(aes(x = points)) +
  geom_histogram() +
  facet_grid(actual_projected ~ .)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



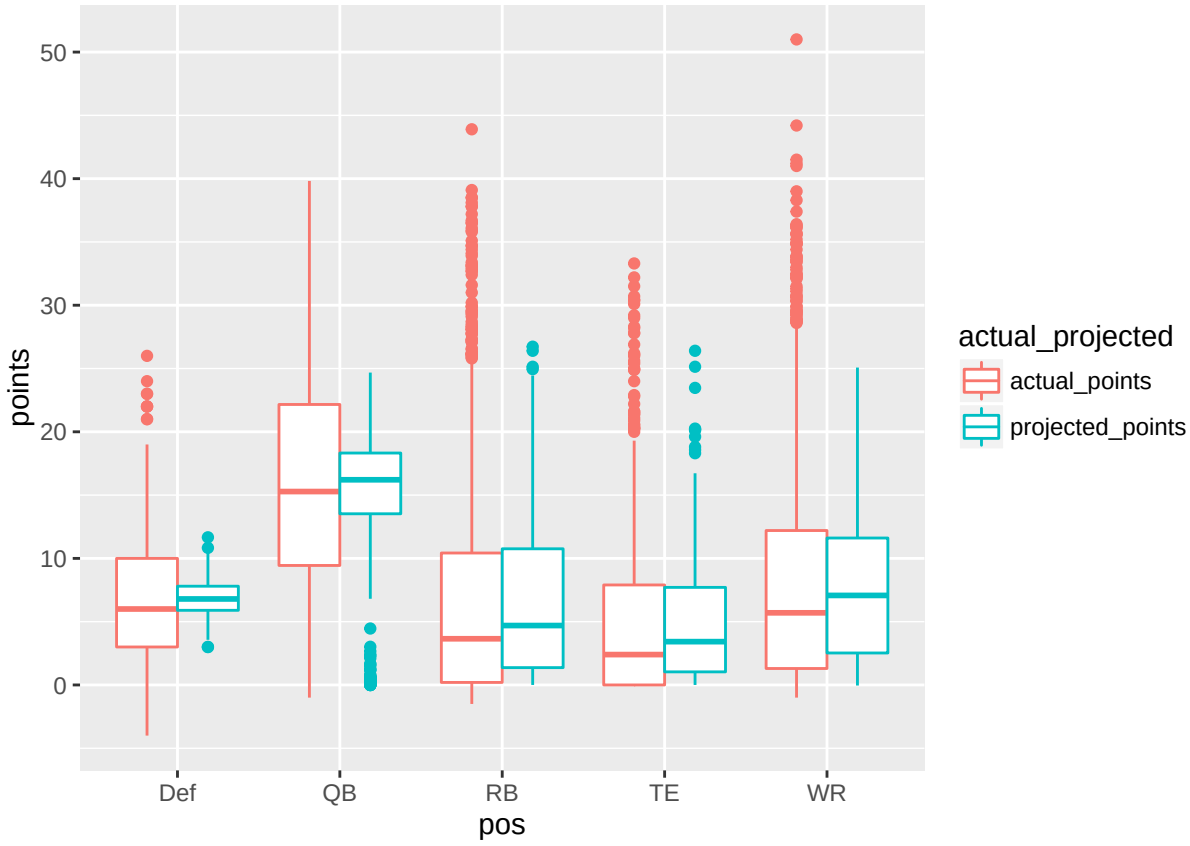

```
df %>% select(playername, actual_points, projected_points) %>%
  gather('actual_projected', 'points', -1) %>%
  ggplot(aes(x = actual_projected, y = points)) +
  geom_boxplot()
```



As illustrated in the summary statistics and histogram above, the distribution of actual points scored is highly right skewed. This is unsurprising as a small number of players significantly outperform, while most may hardly contribute any points at all. There are a small number of players with negative points, which might be surprising except that the scoring systems penalize players for certain results, e.g. fumbles or interceptions.

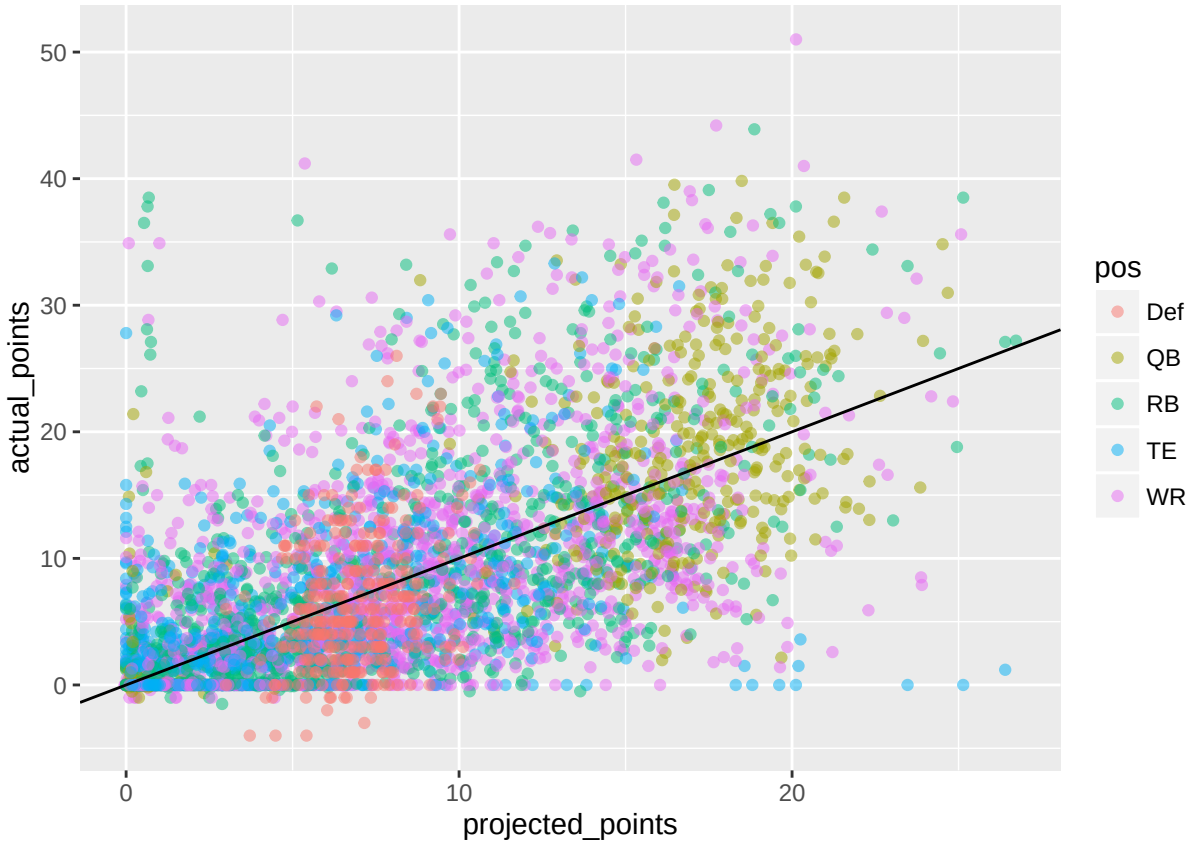
Interestingly, the actual data exhibits much more variation than the projected data. However, this makes sense as the projected data is based on averages of several expert projections, and the averaging should pull in the extremes. It's also likely that experts may be unlikely to take extreme views in fear of being extremely wrong, but we would need to look at individual expert projections in order to do so (and that is outside the scope of this study).

```
df %>% select(playername, pos, actual_points, projected_points) %>%
  gather('actual_projected', 'points', -(1:2)) %>%
  ggplot(aes(x = pos, y = points, col = actual_projected)) +
  geom_boxplot()
```



In the above chart, we again look at the distribution of data but this time by position. While the actual data is still clearly more variable than the projected data across all positions, the interquartile ranges for the running back, tight end and wide receiver positions are pretty close for actual and projected data. The projected IQRs are, however, much tighter than the actual IQRs for the defense and quarterback positions. This may be a result of having fewer observations as there is only one defense and one quarterback per team.

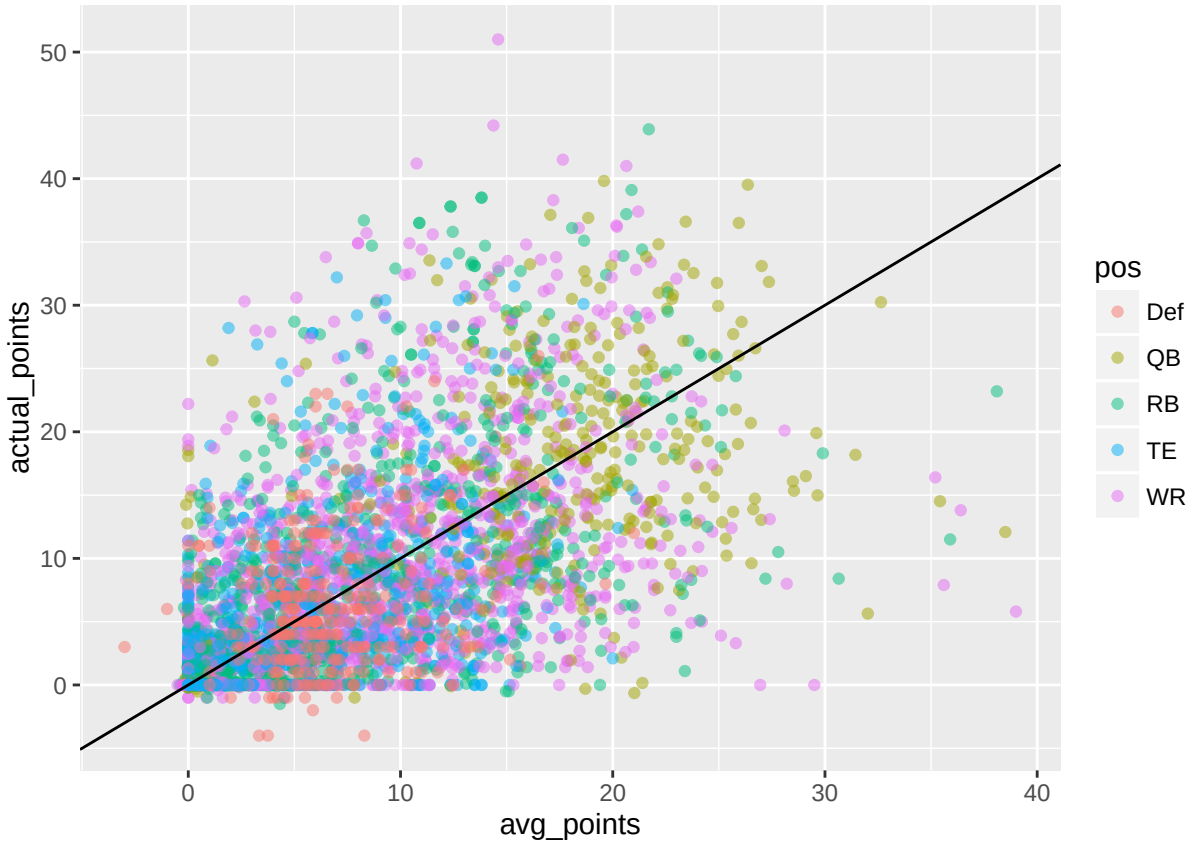
```
df %>% select(playername, pos, actual_points, projected_points) %>%
  ggplot(aes(x = projected_points, y = actual_points, col = pos)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = 0)
```



Looking at all the data together on a single scatter plot, as above, makes it a bit difficult to discern what is happening. Still, it's clear that there is some relationship between actual points and projected points; however, eyeballing the strength of that association is difficult.

```
df %>% select(playername, pos, actual_points, avg_points) %>%
  ggplot(aes(x = avg_points, y = actual_points, col = pos)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = 0)
```

```
## Warning: Removed 556 rows containing missing values (geom_point).
```



This time, plotting actual vs average points, we can again seem some associate but cannot make more specific determinations. In order to do so, we will next turn to measures of fit, specifically RMSE and MAE in this case.

```
# create functions for RMSE and MAE
# unction that returns Root Mean Squared Error
rmse <- function(error)
{
    sqrt(mean(error^2, na.rm = T))
}

# function that returns Mean Absolute Error
mae <- function(error)
{
    mean(abs(error), na.rm = T)
}

# measure RMSE and MAE for both the projected and average point data
df %>% mutate(projected_error = actual_points - projected_points, avg_error = actual_points - avg_points
  summarise(projected_rmse = rmse(projected_error), avg_rmse = rmse(avg_error),
            projected_mae = mae(projected_error), avg_mae = mae(avg_error))
```

```
##   projected_rmse avg_rmse projected_mae  avg_mae
## 1       6.313521 6.967551      4.341702 4.850052
```

Based on the RMSE and MAE values calculated above, we can now see that across the entire data set, the expert projections are more accurate than using the simple historical averages.

But let's see if how this breakdown both by position and over time...

```
df %>% mutate(projected_error = actual_points - projected_points, avg_error = actual_points - avg_points
  group_by(pos) %>%
  summarise(projected_rmse = rmse(projected_error), avg_rmse = rmse(avg_error),
            projected_mae = mae(projected_error), avg_mae = mae(avg_error))
```

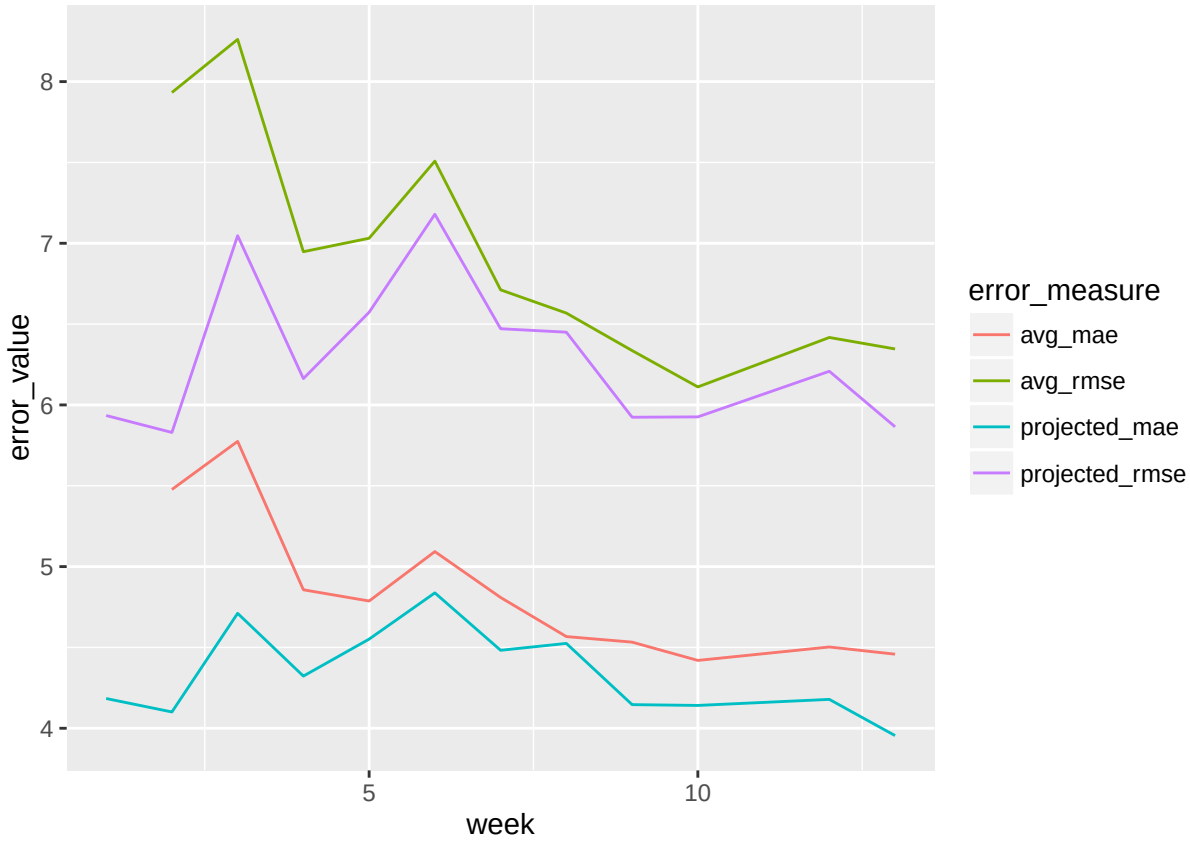
```
## # A tibble: 5 x 5
##     pos projected_rmse avg_rmse projected_mae  avg_mae
##   <chr>          <dbl>    <dbl>         <dbl>    <dbl>
## 1   Def       5.137230 5.889142      4.108430 4.722216
## 2    QB       6.956712 8.410533      5.144661 6.447365
## 3    RB       6.313119 6.819910      4.070112 4.531906
## 4    TE       5.473180 5.832000      3.677351 3.947875
## 5    WR       6.765238 7.439923      4.754790 5.218568
```

Again, we see that the expert projections are more accurate than the historical averages in predicting actual points scored. It's somewhat surprising that this has proven true for every position.

Let's now turn to a time-based analysis. One would expect that predictions might become more accurate as the season progresses and more data are available. This holds true for both the expert projection and the historical average, so the absolute error measurements may prove more interesting than the relative measurements in this case.

```
df %>% mutate(projected_error = actual_points - projected_points, avg_error = actual_points - avg_points
  group_by(week) %>%
  summarise(projected_rmse = rmse(projected_error), avg_rmse = rmse(avg_error),
            projected_mae = mae(projected_error), avg_mae = mae(avg_error)) %>%
  gather('error_measure', 'error_value', -1) %>%
  ggplot(aes(x = week, y = error_value, col = error_measure)) +
  geom_line()
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```



Apparently, our expectations may have been unfounded. Although the absolute error measurements at week 13 are below those of week 1, for the projected data all of the intervening weeks showed higher measures of error than in week 1. This is only a single season, so this particular observation may not be generalizable across other seasons. Separately, it is also interesting that the historical average-based method produced lower and lower error measures as the weeks progressed, while the expert projections-based method did not. However, the expert projections still proved more accurate across the board.

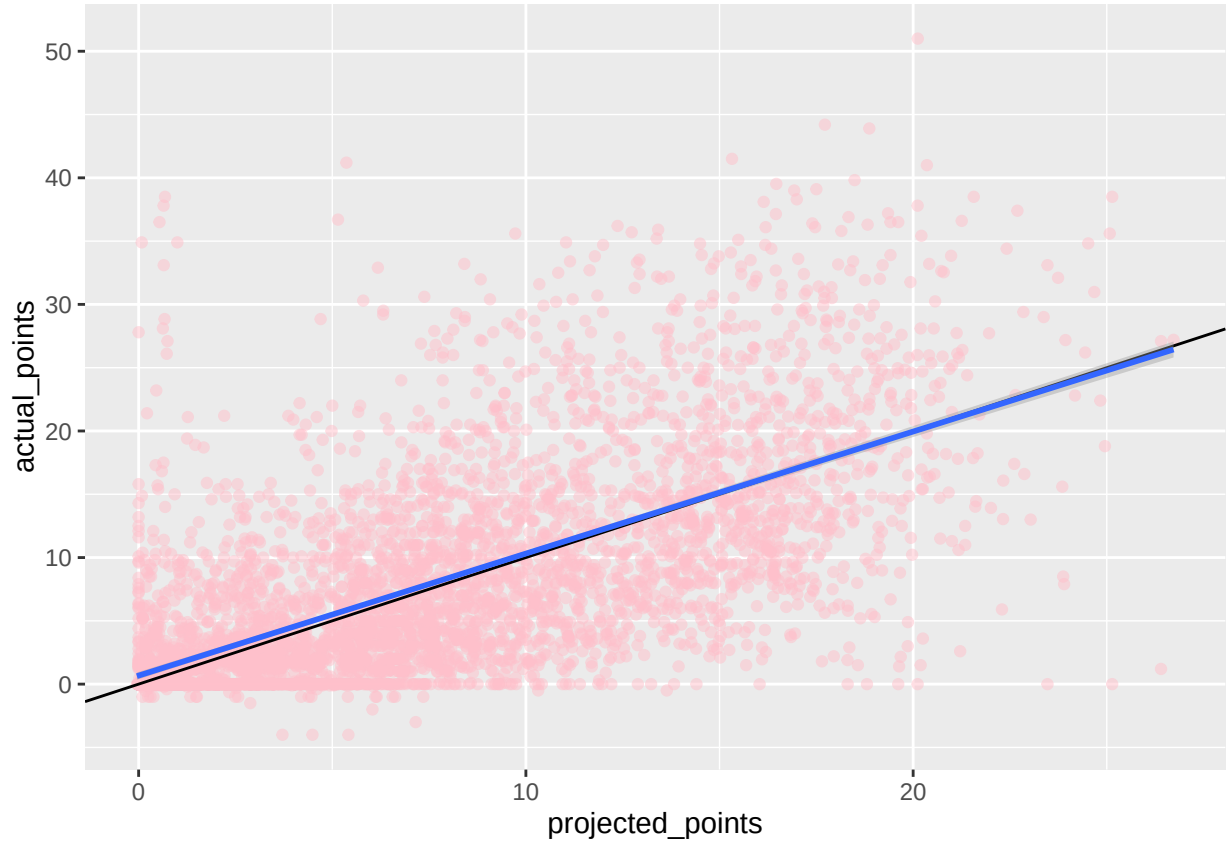
Inference

```
lm = lm(actual_points ~ projected_points, data = df)
summary(lm)
```

```
##
## Call:
## lm(formula = actual_points ~ projected_points, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.924  -3.473  -1.098   2.377  37.169
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         0.6750     0.1477   4.571 4.98e-06 ***
## projected_points    0.9639     0.0158  61.020  < 2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.298 on 4617 degrees of freedom
## Multiple R-squared:  0.4464, Adjusted R-squared:  0.4463
## F-statistic:  3723 on 1 and 4617 DF,  p-value: < 2.2e-16
```

```
df %>% select(playername, pos, actual_points, projected_points) %>%
  ggplot(aes(x = projected_points, y = actual_points)) +
  geom_point(col = 'pink', alpha = 0.5) +
  geom_abline(intercept = 0) +
  geom_smooth(method = 'lm')
```



Fitting a linear regression results in an estimated slope of near 1 (0.9639) with a strong statistical significance. As you can see from the scatter plot above, the fitted line is pretty close to the reference line.

However, this data set and research question do not lend themselves to making inferences since we are not using a model to make predictions. Instead, we are taking the predictions as given and measuring the accuracy of the predictions against the historical average as a base line.

In order to make a predictive model, we could fit a linear model to estimate fantasy points based on certain numeric variables (yards gained, touchdowns, etc.) and certain categorical variables (position, team, etc.). After fitting such a model, we would run certain model diagnostics (histograms, qq plots, etc.) to ensure that the residuals are about normally distributed without any remaining structural patterns. However, that is beyond the scope of this particular research question.

Conclusion

Based on the relative RMSE and MAE values, the expert projections were clearly more accurate than the baseline of using the historical averages. Unsurprisingly, the expert projections tended to be more concentrated around the mean with fewer outliers as compared to the actual results. But again this may be a function of averaging the expert projections rather, and we would need to look at the individual expert projections to confirm, which could be a topic for future study. Other areas for future exploration include using the expert projections for specific stats to fit a predictive model to estimate fantasy points. This would provide opportunities to make inferences and require additional diagnostics that were unnecessary here.

Additionally, in order to make profitable bets, our predictions would need to be more precise than the other bettors on average - a challenge much more difficult than beating a simple historical average base line. Also, without proprietary data from the betting platforms, it may be 'cost prohibitive' to obtain such information through trial and error. Nonetheless, with a sufficient bankroll, this could prove to be an interesting area for study.