Project 1 (Exploratory Data Analysis)

Sagar Patel

2020-03-03

Introduction:

The two datasets I have chosen are NFL player statistics from the 2019 regular season, as retrieved from Pro-Football-Reference. The first dataset contains rushing stats for all applicable players (variables like rushing attempts, rushing yards, rushing touchdowns, etc.), and the second dataset contains receiving stats for all applicable players (variables like catches, receiving yards, receiving touchdowns, etc.). It is important to note that this list does *not* exclude quarterbacks, but it does exclude passing stats. That is, quarterbacks' rushing and receiving stats are included in the datasets but their passing stats are not. There is also a large number of players such as punters and offensive tackles involved in "trick plays" where they accumulate a very small number of touches/yards but are still included in the datasets.

I have chosen these datasets because I am an avid NFL fan as well as a devoted fantasy football player. I am interested in visualizing this data and seeing which players truly set themselves apart from others in 2019. I believe most, if not all, variables will be positively correlated. This makes sense because the more often a football player possesses the football, the more yards and touchdowns he will likely have.

```r
library(tidyverse)
```

```
## -- Attaching packages -------------------------------------------------------------- tidyverse 1.3.0
```

```
## v ggplot2 3.3.0     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.0.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ----------------------------------------------------------------- tidyverse_conflicts()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(readxl)
library(ggrepel)
library(cluster)

rushing <- read_excel('Rushing.xlsx',
    col_types = c('numeric', 'text', 'text',
        'numeric', 'text', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric',
        'numeric', 'numeric'))
```

```r
receiving <- read_excel('Receiving.xlsx',
    col_types = c('numeric', 'text', 'text',
        'numeric', 'text', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric',
        'numeric', 'numeric', 'numeric'))
```

#Tidying:

We will tidy the datasets by first making all positions uppercase, as the original datasets from Pro-Football-Reference had a mix of uppercase and lowercase positions. Then we will be removing variables that are deemed unnecessary. These are either variables that we can create ourselves (e.g. a yards per game variable could be created using the games and yards variables) or variables that will not serve a purpose in our analysis/visualization (such as the "longest play" variable). We will demonstrate the use of the pivot_longer() and pivot_wider() functions on the `rushing` dataset.

```r
rushing$Pos <- toupper(rushing$Pos)
rushing <- rushing %>% select(-'Rk', -'Y/A', -'Y/G', -'Lng') %>%
  rename(Rush_Yds = 'Yds', Rush_TD = 'TD', Rush_1D = '1D')

receiving$Pos <- toupper(receiving$Pos)
receiving <- receiving %>%
  select(-'Rk', -'Ctch%', -'Y/R', -'Y/Tgt', -'R/G', -'Y/G', -'Lng') %>%
  rename(Rec_Yds = 'Yds', Rec_TD = 'TD', Rec_1D = '1D')

rushing <- rushing %>% pivot_longer(c(G:Fmb), names_to = 'Stat', values_to = 'Value')
head(rushing)
```

```
## # A tibble: 6 x 6
##   Player        Tm      Age Pos   Stat     Value
##   <chr>         <chr> <dbl> <chr> <chr>    <dbl>
## 1 Derrick Henry TEN      25 RB    G           15
## 2 Derrick Henry TEN      25 RB    GS          15
## 3 Derrick Henry TEN      25 RB    Att        303
## 4 Derrick Henry TEN      25 RB    Rush_Yds  1540
## 5 Derrick Henry TEN      25 RB    Rush_TD     16
## 6 Derrick Henry TEN      25 RB    Rush_1D     73
```

```r
rushing <- rushing %>% pivot_wider(names_from = 'Stat', values_from = 'Value')
head(rushing)
```

```
## # A tibble: 6 x 11
##   Player       Tm      Age Pos       G    GS   Att Rush_Yds Rush_TD Rush_1D   Fmb
##   <chr>        <chr> <dbl> <chr> <dbl> <dbl> <dbl>    <dbl>   <dbl>   <dbl> <dbl>
## 1 Derrick He~ TEN      25 RB       15    15   303     1540      16      73     5
## 2 Nick Chubb   CLE      24 RB       16    16   298     1494       8      62     3
## 3 Christian ~ CAR      23 RB       16    16   287     1387      15      57     1
## 4 Ezekiel El~ DAL      24 RB       16    16   301     1357      12      78     3
## 5 Chris Cars~ SEA      25 RB       15    15   278     1230       7      75     7
## 6 Lamar Jack~ BAL      22 QB       15    15   176     1206       7      71     9
```

#Joining/Merging:

For these two datasets, we will be performing a full join. This is because we want a dataset of *all* players who carried or caught the football, but not necessarily did both. Because some players only did one (e.g. caught the ball but did not carry it), we will replace all NAs with 0s. An NA after joining is essentially a 0 because it means the player did not record anything for that statistic. We do not want to remove these players from the full dataset, but rather fill their stats with 0s so descriptive statistics can be determined. Then, we will group all positions that are not quarterback, running back, tight end, and wide receiver into one position called "Other". This will help us exclude the aforementioned trick players with very small stats if need be. Lastly, we will arrange alphabetically by first names.

```
full_dat <- rushing %>% full_join(receiving)
```

```
## Joining, by = c("Player", "Tm", "Age", "Pos", "G", "GS", "Fmb")
```

```
full_dat[is.na(full_dat)] <- 0
full_dat$Pos[full_dat$Pos != 'QB' & full_dat$Pos != 'RB'&
               full_dat$Pos != 'TE' & full_dat$Pos != 'WR'] <- 'Other'
full_dat <- full_dat %>% arrange(Player)
```

#Wrangling:

The first way we will wrangle the data is by using the mutate() function to create two new variables. The first new variable is "Total Yards", which is the sum of "Receiving Yards" and "Rushing Yards". The second new variable is "Total TDs", which is the sum of "Receiving TDs" and "Rushing TDs". Then we will create a new dataset using the select() function called `nums`. `nums` is just the full data without categorical variables. This will be used to generate summary statistics. The `nums` dataframe will be tidied to create the `tidycor` dataframe which will later be used to generate a correlation heatmap. Lastly, we will create three new dataframes using combinations of group_by(), select(), summarize_all(), arrange(), and filter(). The `tm` dataframe contains total stats for each team in the league. The `tm_pos` dataframe contains total stats for each skill position group for each team in the league. The `top` dataframe contains the full stats for the top 30 players in total yards. These new dataframes will be used for visualization.

```
full_dat <- full_dat %>% mutate(Tot_Yds = Rec_Yds + Rush_Yds, Tot_TD = Rec_TD + Rush_TD)

nums <- full_dat %>% select(-Player, -Tm, -Pos)
nums %>% summary
```

```
##       Age              G               GS              Att
##  Min.   :21.00   Min.   : 1.00   Min.   : 0.000   Min.   :  0.00
##  1st Qu.:24.00   1st Qu.: 8.00   1st Qu.: 1.000   1st Qu.:  0.00
##  Median :25.00   Median :13.00   Median : 3.000   Median :  1.00
##  Mean   :26.08   Mean   :11.43   Mean   : 5.491   Mean   : 23.08
##  3rd Qu.:27.00   3rd Qu.:16.00   3rd Qu.:10.000   3rd Qu.: 12.00
##  Max.   :42.00   Max.   :17.00   Max.   :16.000   Max.   :303.00
##     Rush_Yds          Rush_TD           Rush_1D            Fmb
##  Min.   : -12.00   Min.   : 0.0000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.:   0.00   1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.: 0.000
##  Median :   2.00   Median : 0.0000   Median : 0.000   Median : 0.000
##  Mean   :  99.66   Mean   : 0.7707   Mean   : 5.367   Mean   : 1.034
##  3rd Qu.:  50.25   3rd Qu.: 0.0000   3rd Qu.: 3.000   3rd Qu.: 1.000
##  Max.   :1540.00   Max.   :16.0000   Max.   :78.000   Max.   :18.000
##       Tgt              Rec             Rec_Yds            Rec_TD
##  Min.   :  0.00   Min.   :  0.00   Min.   :  -5.00   Min.   : 0.000
##  1st Qu.:  2.00   1st Qu.:  1.00   1st Qu.:  11.75   1st Qu.: 0.000
##  Median : 14.00   Median :  9.00   Median :  89.50   Median : 0.000
##  Mean   : 29.52   Mean   : 19.54   Mean   : 222.26   Mean   : 1.374
##  3rd Qu.: 45.00   3rd Qu.: 31.00   3rd Qu.: 306.00   3rd Qu.: 2.000
##  Max.   :185.00   Max.   :149.00   Max.   :1725.00   Max.   :11.000
##      Rec_1D          Tot_Yds           Tot_TD
##  Min.   : 0.00   Min.   :  -9.0   Min.   : 0.000
##  1st Qu.: 1.00   1st Qu.:  34.0   1st Qu.: 0.000
##  Median : 4.00   Median : 166.0   Median : 1.000
##  Mean   :10.69   Mean   : 321.9   Mean   : 2.145
##  3rd Qu.:16.00   3rd Qu.: 454.2   3rd Qu.: 3.000
##  Max.   :91.00   Max.   :2392.0   Max.   :19.000
```

```
nums %>% cor %>% head()
```

```
##                  Age           G        GS        Att   Rush_Yds     Rush_TD
## Age       1.00000000 -0.03880076 0.1063264 -0.1087697 -0.1276163 -0.08253355
## G        -0.03880076  1.00000000 0.5173811  0.2227367  0.2252936  0.20518546
## GS        0.10632635  0.51738110 1.0000000  0.3143615  0.3152542  0.32112120
## Att      -0.10876971  0.22273671 0.3143615  1.0000000  0.9854292  0.86334260
## Rush_Yds -0.12761629  0.22529359 0.3152542  0.9854292  1.0000000  0.87339451
## Rush_TD  -0.08253355  0.20518546 0.3211212  0.8633426  0.8733945  1.00000000
##             Rush_1D        Fmb         Tgt         Rec     Rec_Yds      Rec_TD
## Age      -0.1120369 0.04793299 -0.02198411 -0.01950684 -0.02303655 -0.01668191
## G         0.2272676 0.16164930  0.42643861  0.42722527  0.39831139  0.34882832
## GS        0.3363171 0.38298511  0.57529612  0.57282541  0.56405702  0.49805969
## Att       0.9728218 0.29029812  0.09402431  0.16042454  0.01607470 -0.04344289
## Rush_Yds  0.9800890 0.29369609  0.09936723  0.16251020  0.02651026 -0.03106386
## Rush_TD   0.8857986 0.33356405  0.06584095  0.11838549  0.01339995 -0.03334066
##                 Rec_1D     Tot_Yds     Tot_TD
## Age      -0.0033008447 -0.09661942 -0.0708696
## G         0.3927161256  0.44590374  0.3994747
## GS        0.5606814297  0.62911155  0.5904154
## Att      -0.0043639803  0.62150802  0.5830005
## Rush_Yds  0.0055912986  0.63854863  0.5991489
## Rush_TD  -0.0007345649  0.55019576  0.6876175
```

```
tidycor <- cor(nums) %>% as.data.frame %>% rownames_to_column %>%
  pivot_longer(-1, names_to='name', values_to='correlation')
head(tidycor)
```

```
## # A tibble: 6 x 3
##   rowname name     correlation
##   <chr>   <chr>          <dbl>
## 1 Age     Age           1
## 2 Age     G            -0.0388
## 3 Age     GS            0.106
## 4 Age     Att          -0.109
## 5 Age     Rush_Yds     -0.128
## 6 Age     Rush_TD      -0.0825
```

```r
tm <- full_dat %>% group_by(Tm) %>% select(-Player, -G, -GS, -Pos)%>%
  summarize_all(sum) %>% arrange(desc(Tot_Yds)) %>% filter(Tm != '3TM' & Tm != '2TM')
head(tm)
```

```
## # A tibble: 6 x 14
##   Tm      Age   Att Rush_Yds Rush_TD Rush_1D   Fmb   Tgt   Rec Rec_Yds Rec_TD
##   <chr> <dbl> <dbl>    <dbl>   <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>
## 1 DAL     337   449     2153      18     120    15   574   388    4902     30
## 2 TAM     527   409     1521      15      81    22   607   382    5127     33
## 3 BAL     433   595     3295      21     188    15   424   289    3350     37
## 4 KAN     428   374     1565      16      92    16   556   377    4684     30
## 5 TEN     586   445     2223      21     104    20   426   297    3956     29
## 6 LAR     376   401     1499      20      92    16   611   397    4669     22
## # ... with 3 more variables: Rec_1D <dbl>, Tot_Yds <dbl>, Tot_TD <dbl>
```

```r
tm_pos <- full_dat %>% group_by(Tm, Pos) %>% select(-Player, -G, -GS)%>%
  summarize_all(sum) %>% arrange(desc(Tot_Yds)) %>% filter(Tm != '3TM' & Tm != '2TM')%>%
  filter(Pos == 'RB' | Pos == 'WR' | Pos == 'TE')
head(tm_pos)
```

```
## # A tibble: 6 x 15
## # Groups:   Tm [6]
##   Tm    Pos     Age   Att Rush_Yds Rush_TD Rush_1D   Fmb   Tgt   Rec Rec_Yds
##   <chr> <chr> <dbl> <dbl>    <dbl>   <dbl>   <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 TAM   WR      220     6       51       0       3     3   376   223    3566
## 2 DAL   WR      180    10       64       1       2     3   357   224    3475
## 3 LAR   WR      128    30      194       1      10     3   393   249    3218
## 4 ATL   WR      151     8       43       0       2     2   379   244    3107
## 5 DET   WR      138     3        7       0       0     1   325   199    2968
## 6 JAX   WR      179     9       54       0       1     3   355   217    2868
## # ... with 4 more variables: Rec_TD <dbl>, Rec_1D <dbl>, Tot_Yds <dbl>,
## #   Tot_TD <dbl>
```

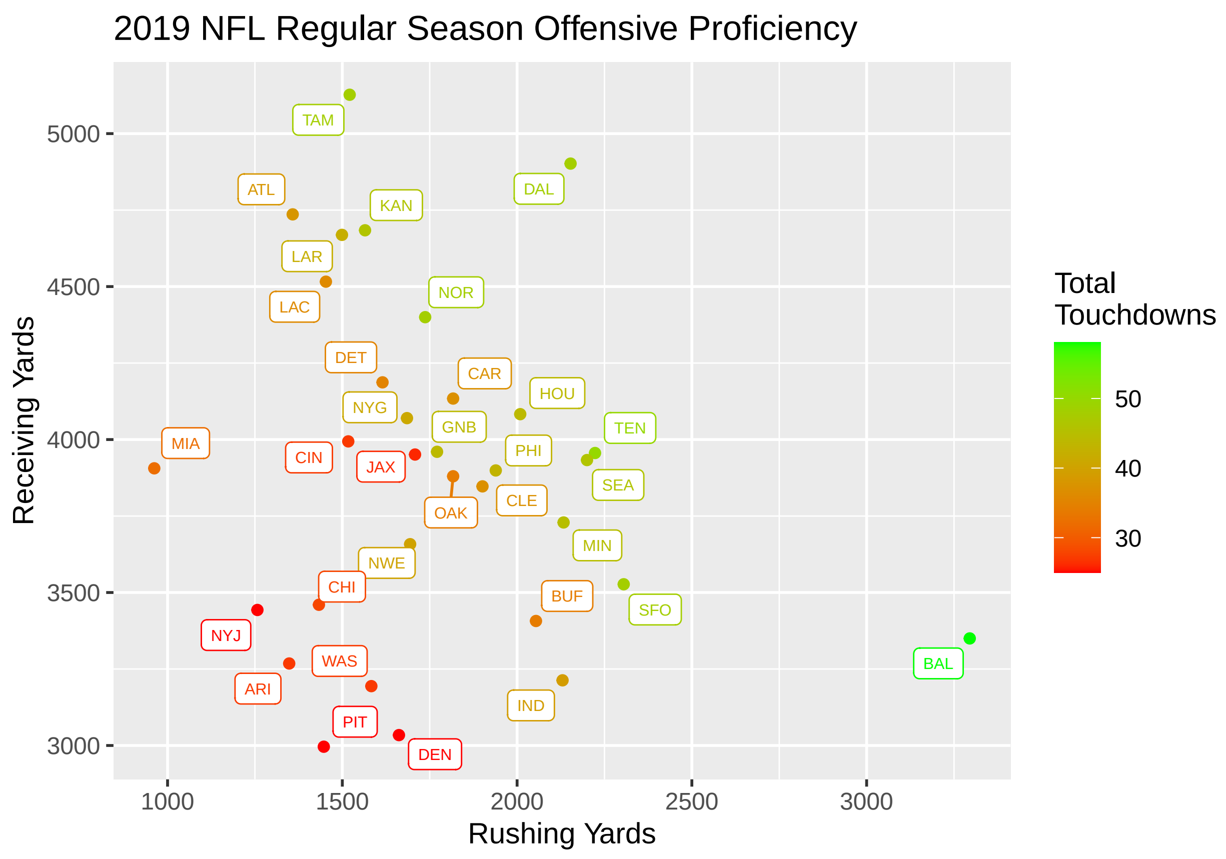
```r
top <- full_dat %>% arrange(desc(Tot_Yds)) %>% top_n(30, Tot_Yds)
head(top)
```

```
## # A tibble: 6 x 18
##   Player Tm      Age Pos       G    GS   Att Rush_Yds Rush_TD Rush_1D   Fmb
##   <chr>  <chr> <dbl> <chr> <dbl> <dbl> <dbl>    <dbl>   <dbl>   <dbl> <dbl>
## 1 Chris~ CAR      23 RB       16    16   287     1387      15      57     1
## 2 Ezeki~ DAL      24 RB       16    16   301     1357      12      78     3
## 3 Nick ~ CLE      24 RB       16    16   298     1494       8      62     3
## 4 Derri~ TEN      25 RB       15    15   303     1540      16      73     5
## 5 Micha~ NOR      26 WR       16    15     1       -9       0       0     1
## 6 Leona~ JAX      24 RB       15    15   265     1152       3      55     1
## # ... with 7 more variables: Tgt <dbl>, Rec <dbl>, Rec_Yds <dbl>, Rec_TD <dbl>,
## #   Rec_1D <dbl>, Tot_Yds <dbl>, Tot_TD <dbl>
```

#Visualizing:

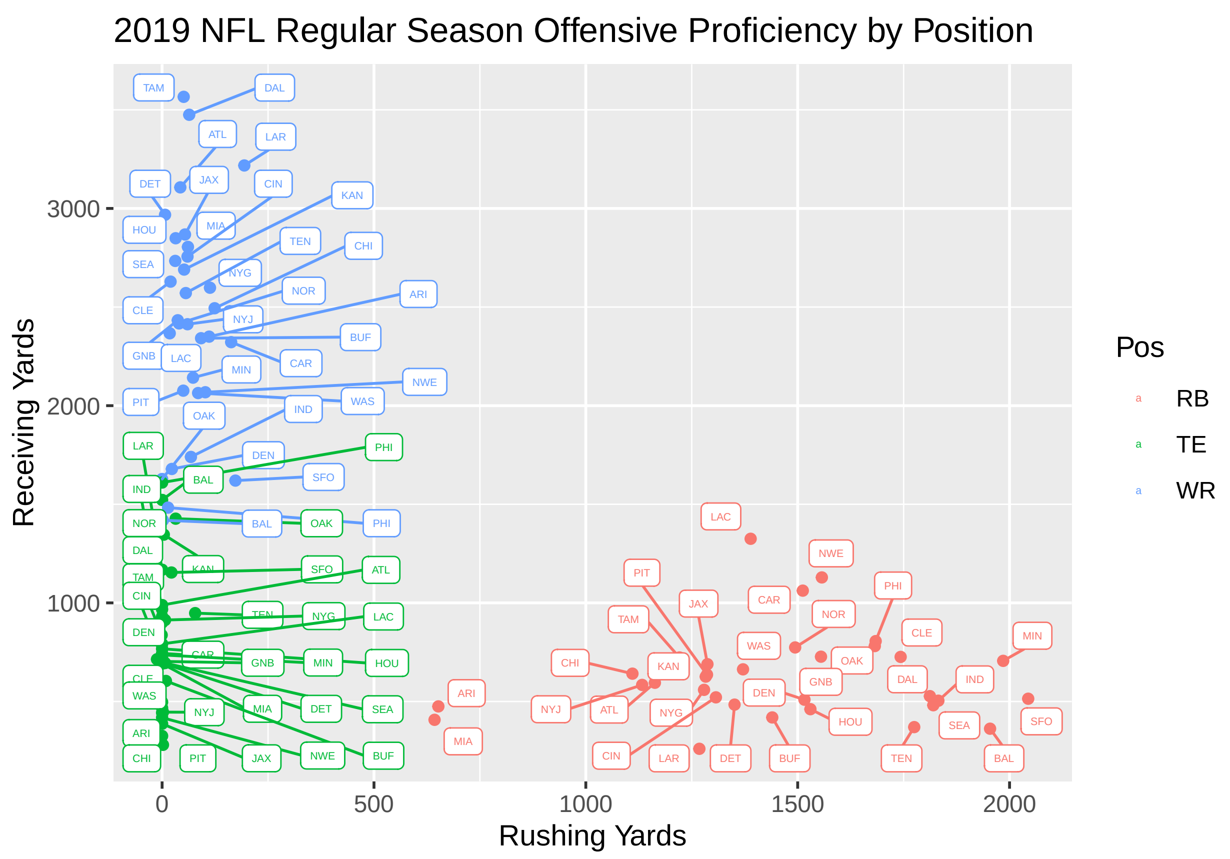
We will be making four different visualizations. The first three will be scatterplots that utilize the new dataframes we made above. The fourth will be a correlation heatmap of our numeric variables.

```r
ggplot(tm, aes(Rush_Yds, Rec_Yds, color=Tot_TD)) + geom_point() + xlab('Rushing Yards') +
  ylab('Receiving Yards') + geom_label_repel(aes(label=Tm), size=2) +
  scale_color_gradient(low='red', high='green', name='Total\nTouchdowns') +
  ggtitle('2019 NFL Regular Season Offensive Proficiency')
```



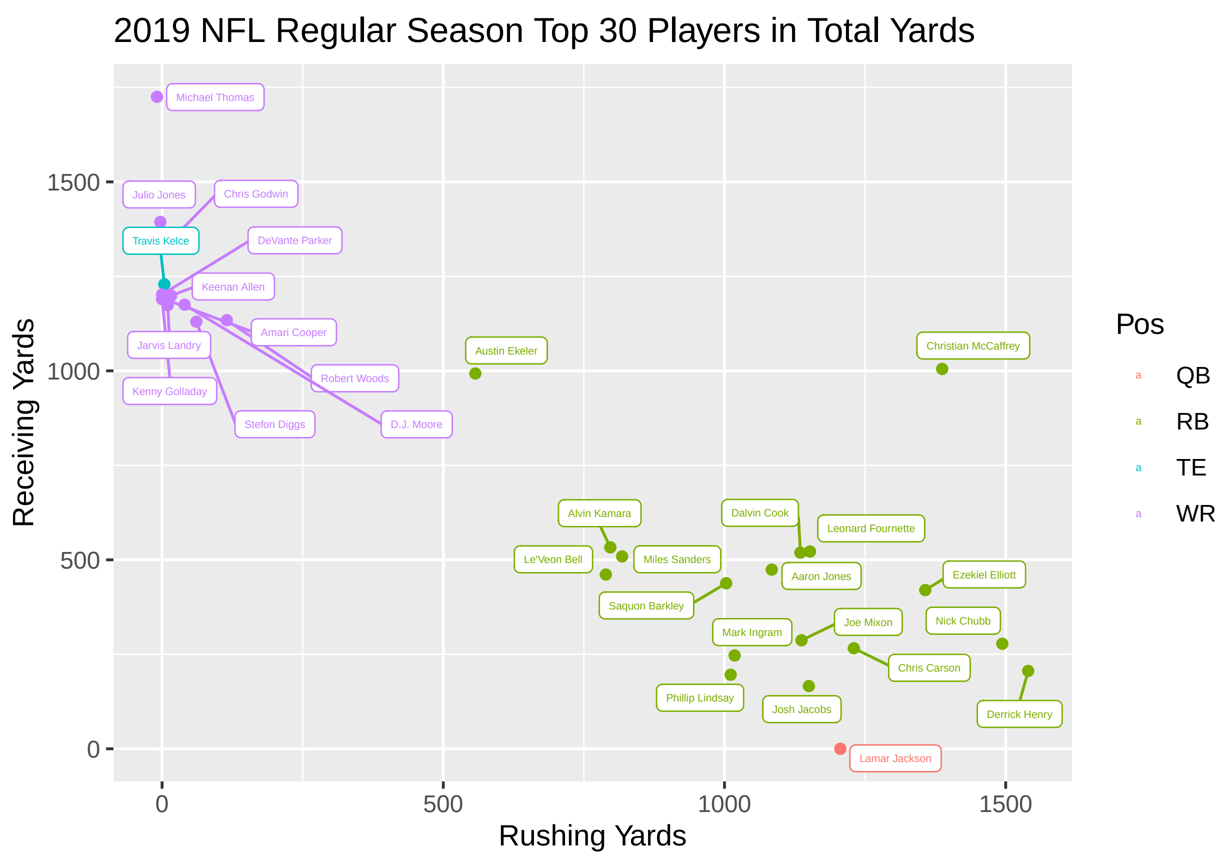
This is a scatterplot that shows each team's total rushing yards on the x-axis and total receiving yards on the y-axis. The points are colored by the number of total touchdowns the team's offense scored, with red being a low number of total TDs and green being a high number of total TDs. From this plot we can see the Baltimore Ravens (BAL) were leaps ahead of other teams in terms of total TDs and rushing yardage. Teams like the Cincinnati Bengals (CIN) and Jacksonville Jaguars (JAX) scored fewer total TDs than you would expect them to given their yardage.

```r
ggplot(tm_pos, aes(Rush_Yds, Rec_Yds, color=Pos)) + geom_point() + xlab('Rushing Yards') +
  ylab('Receiving Yards') + geom_label_repel(aes(label=Tm), size=1.5) +
  ggtitle('2019 NFL Regular Season Offensive Proficiency by Position')
```



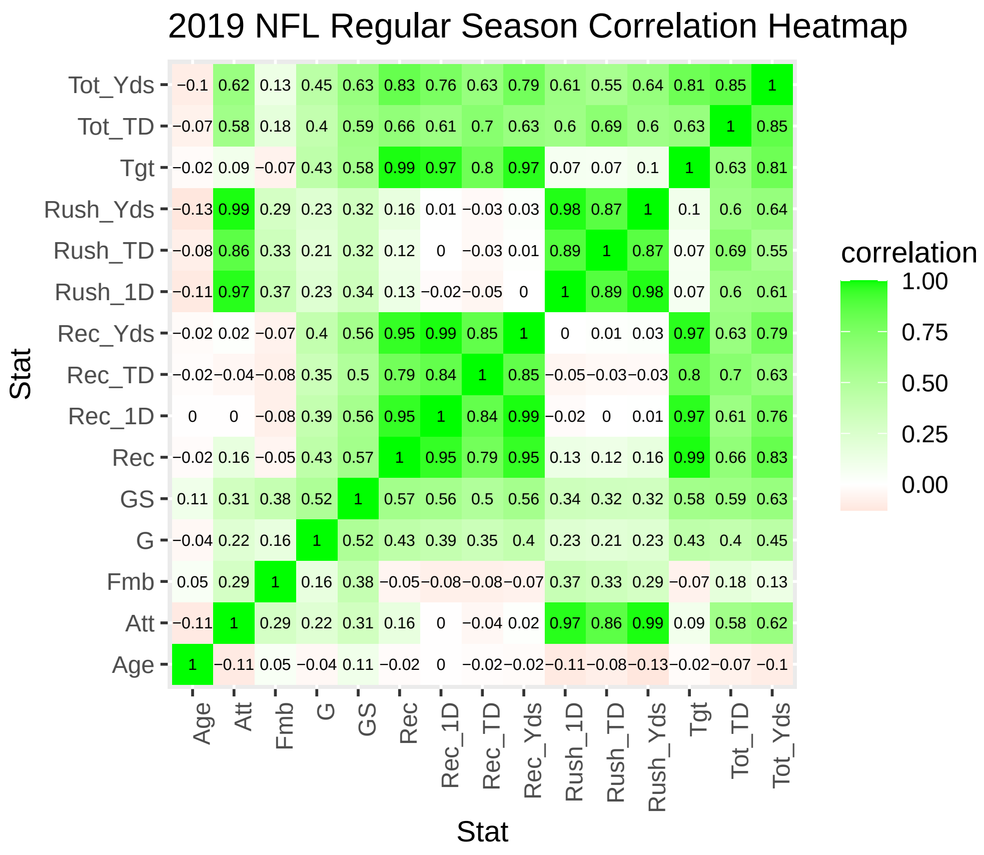
This is a scatterplot that shows the total rushing yards on the x-axis and total receiving yards on the y-axis for each position group for each team (e.g. the red point labeled "LAC" is the combined yardage for *all* Los Angeles Chargers running backs). From this plot we can see that the San Francisco 49ers (SFO) had the most rushing yards from running backs, while the Tampa Bay Buccaneers (TAM) had the most receiving yards from wide receivers. The three position groups are quite clearly defined against one another, except where poor wide receiver groups and great tight end groups meet.

```r
ggplot(top, aes(Rush_Yds, Rec_Yds, color=Pos)) + geom_point() + xlab('Rushing Yards') +
  ylab('Receiving Yards') + geom_label_repel(aes(label=Player), size=1.5, force=1) +
  ggtitle('2019 NFL Regular Season Top 30 Players in Total Yards')
```



This is a scatterplot that shows the total rushing yards on the x-axis and total receiving yards on the y-axis for the top 30 players in total yards. We can see that 17 of the top 30 are running backs, 11 are wide receivers, and only 1 each of quarterbacks and tight ends. There was a very special case in the 2019 NFL Regular Season and his name is Lamar Jackson. Lamar Jackson is a quarterback for the Baltimore Ravens. If you remember from the introduction, QBs are not excluded but their passing stats are. So it is quite remarkable that Lamar Jackson made it into the top 30 players in total yards *without* his passing stats. Even more impressive is that he had the 6th most rushing yards in the league, more than most running backs.

```r
ggplot(tidycor, aes(rowname, name, fill=correlation)) + geom_tile() +
  scale_fill_gradient2(low='red', high='green') +
  geom_text(aes(label=round(correlation, 2)), color='black', size=2) +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  coord_fixed() + xlab('Stat') + ylab('Stat') +
  ggtitle('2019 NFL Regular Season Correlation Heatmap')
```



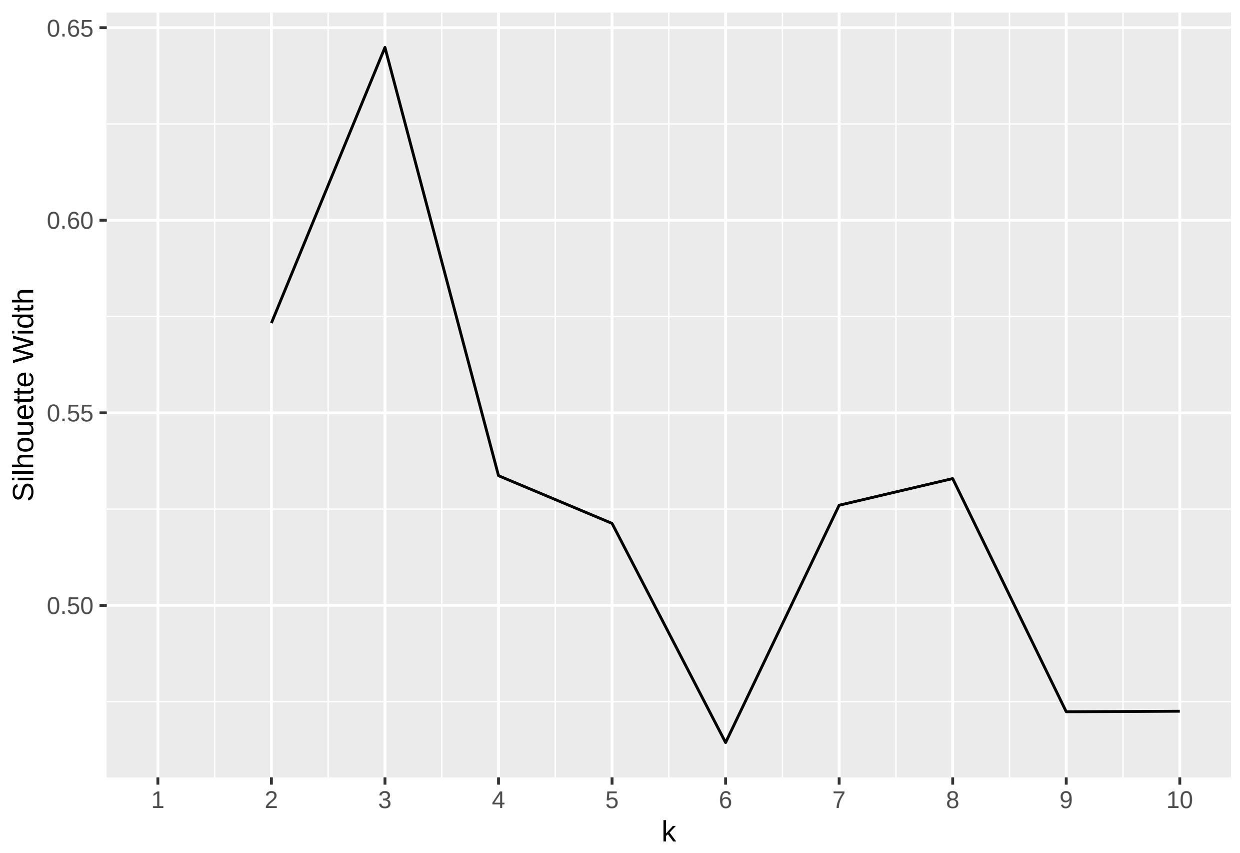
This is a correlation heatmap of all of the numeric variables in the full dataset. As predicted in the introduction, most statistics are positively correlated. Receiving stats such as targets, receptions, yards, first downs, and touchdowns are all strongly positively correlated with each other, while rushing stats such as attempts, yards, first downs, and touchdowns are also strongly positively correlated with each other. It is important to note that rushing stats and receiving stats barely, if at all, correlate with each other. This makes sense because outside of a few outliers, wide receivers purely receive and running backs purely rush. An interesting correlation here is the one that age has with other variables. Most of the correlations that age has with other variables are negative. This makes sense because as players age, their performance generally tends to decline.

#Dimensionality Reduction:

We will be performing PAM clustering on all quarterbacks, running backs, wide receivers, and tight ends. This is to exclude trick players who had very small stats. This data will still include players of the aforementioned positions that did not contribute much.

```r
pam_dat <- full_dat %>% filter(Pos != 'Other') %>% select(Rush_Yds, Rec_Yds)
sil_width <- vector()
for(i in 2:10){
pam_fit <- pam(pam_dat, k = i)
sil_width[i] <- pam_fit$silinfo$avg.width
}
ggplot() + geom_line(aes(x=1:10, y=sil_width)) +
  scale_x_continuous(name='k', breaks=1:10) + ylab('Silhouette Width')
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

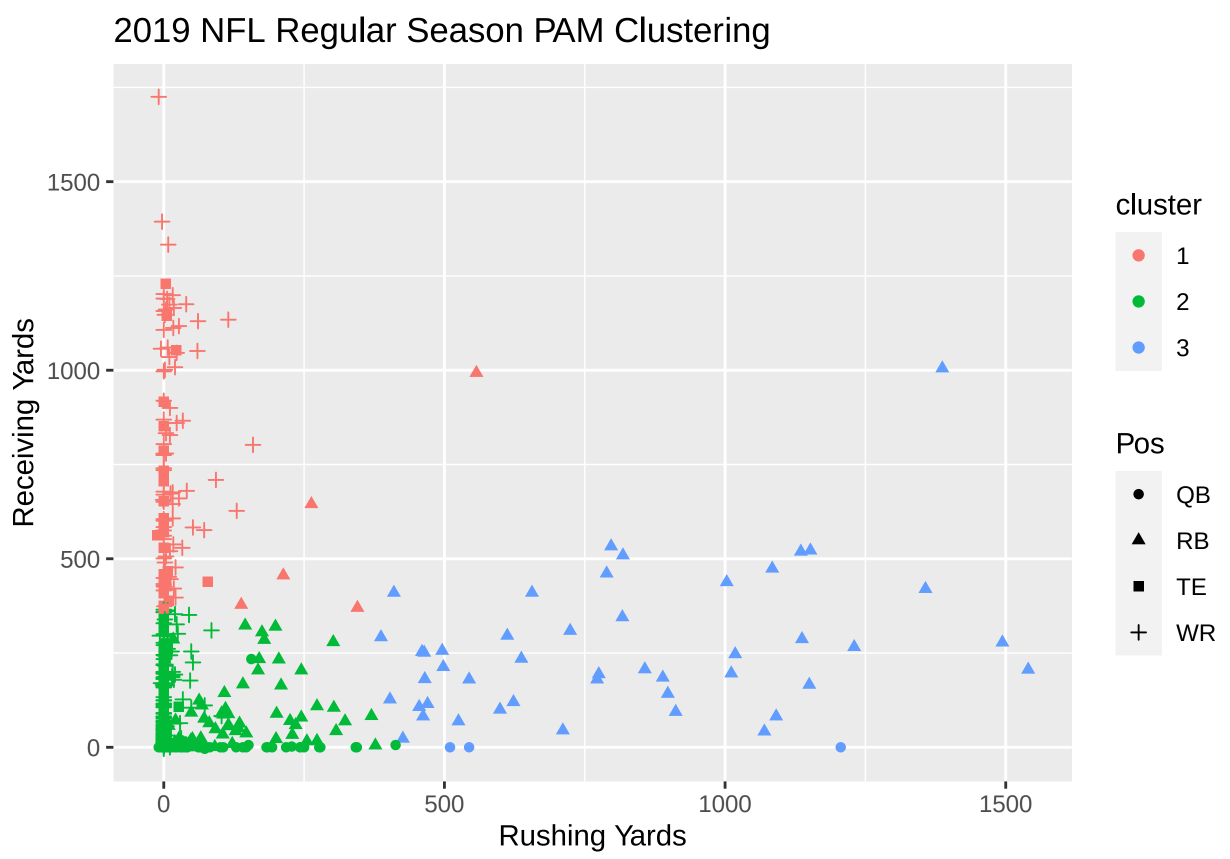


As we can see here, the silhouette width is greatest at k=3. This means we will have 3 clusters in our PAM clustering.

```
pam_fit$silinfo$avg.width
```

```
## [1] 0.4725123
```

```
pam1 <- pam_dat %>% pam(k=3)
pamclust <- full_dat %>% filter(Pos != 'Other') %>%
  mutate(cluster=as.factor(pam1$clustering))
pamclust %>% ggplot(aes(Rush_Yds, Rec_Yds, color=cluster, shape=Pos)) +
  geom_point() + xlab('Rushing Yards') + ylab('Receiving Yards') +
  ggtitle('2019 NFL Regular Season PAM Clustering')
```



As you can see above, our PAM clusters generally seem to coincide with position groups. We can interpret the clusters to tell us which players had meaningful impacts for their teams. That is, players in cluster 1 (red) had a meaningful impact in the pass game, players in cluster 3 (blue) had a meaningful impact in the rush game, and players in cluster 2 (green) did not have a meaningful impact for their team.