College-to-Pro QB Model Projection

By: Sam Burch

2023-03-17

Quarterback (QB) is the most important position in football. Because of this, an NFL team needs an elite QB to win the Superbowl, or they need a great team around the QB. Also, when money starts to become a factor, it is tough to pay an average QB $30-40 million a year and build a great team around them. However, if the QB is on a rookie contract, this allows the team to build such a roster. Thus, having only an average QB on a rookie deal is immensely valuable.

With this in mind, I decided to build a model projecting QBs from college to the NFL. Projecting QBs is still an unsolved problem (The Ringer). If I can develop a model that is good enough at projecting QBs, this should provide a better starting point for NFL teams. Then, after considering other factors like character, these teams will have enough information to make a decision and be right (on average) more than they were before.

Preliminary

```
# Necessary Libraries
library(cfbfastR)
library(nflreadr)
library(dplyr)
library(ggplot2)
library(ggrepel)
library(tidyverse)
library(lubridate)
library('Cairo')
```

NFL Data

The first aspect of building the model is determining what NFL success is. Ben Baldwin stated "[about] 95% of [how good a QB is can be determined from] looking at EPA (expected points added) and PFF (Pro Football Focus) grade." So, we will use this as our indicator of QB play in the NFL.

Play-By-Play Data (PBP)

```
pbp_nfl = load_pbp(2015:2022)
```

```
pbp_nfl = pbp_nfl |>
  mutate(split_game_id = str_split(game_id, '_', simplify = TRUE),
         year_id = as.double(split_game_id[,1]))

# Use this to figure out when the quarterback is running the ball.
qbs_nfl = pbp_nfl |>
  filter(pass == 1) |>
  group_by(passer) |>
  summarise(n = n()) |>
  filter(n >= 100) |>
  pull(passer) |>
  unique()

pbp_nfl_2 = pbp_nfl |>
  filter(!is.na(yards_gained), (pass == 1 & (passer %in% qbs_nfl)) | (rush == 1 & (rusher %in% qbs_nfl))
  mutate(passer_new = case_when(!is.na(passer) ~ passer, !is.na(rusher) ~ rusher)) |>
  mutate(passer_id_new = case_when(!is.na(passer) ~ passer_id, !is.na(rusher) ~ rusher_id)) |>
  group_by(passer_new, passer_id_new) |>
  summarise(mean_epa = mean(epa, na.rm = TRUE),
            ypa = mean(yards_gained),
            sack_rate = mean(sack),
            cpoe = mean(cpoe, na.rm = TRUE),
            plays = n(),
            .groups = 'drop') |>
  filter(plays >= 650) |>
  arrange(-mean_epa)
pbp_nfl_2
```

```
## # A tibble: 65 x 7
##    passer_new  passer_id_new mean_epa   ypa sack_rate  cpoe plays
##    <chr>       <chr>            <dbl> <dbl>     <dbl> <dbl> <int>
##  1 P.Mahomes   00-0033873       0.280  6.97    0.0358 3.14   4213
##  2 J.Garoppolo 00-0031345       0.184  6.70    0.0581 0.833  2167
##  3 D.Brees     00-0020531       0.182  6.84    0.0350 3.88   3709
##  4 T.Brady     00-0019596       0.179  6.38    0.0364 0.891  6425
##  5 A.Rodgers   00-0023459       0.168  6.04    0.0543 2.27   5469
##  6 J.Allen     00-0034857       0.150  6.15    0.0455 1.28   3777
##  7 P.Rivers    00-0022942       0.142  6.56    0.0448 1.88   4060
##  8 D.Watson    00-0033537       0.138  6.37    0.0764 3.20   2723
##  9 L.Jackson   00-0034796       0.138  6.22    0.0545 0.308  2773
## 10 J.Burrow    00-0036442       0.136  6.07    0.0703 3.94   2175
## # ... with 55 more rows
```

Note: The minimum play count (dropbacks and designed rushes) is 650 here to make sure at least a full year of QB play is present. Otherwise, a player with a small sample would affect the results greatly; for example, Brock Purdy would be 2nd. This number was calculated by the following: $((154 * .25) * 17 \sim 650$. There are 154 plays (on average) in a game (NFL Football Operations), and we'll assume half of that is an offensive play by the team and half of that is a pass play. Also, there are 17 games in a season. Of course there are playoff games and in previous seasons there were 16 games in a season, however, this should still give us a good baselin.

The best QBs from strictly EPA shows a decent list. However, there are some issues with EPA (like surrounding talent) that may lead Jimmy Garoppolo to look better and Justin Herbert to look worse. That is one reason why we must consider their PFF grading as well.

PFF

```
pff_nfl_22 = read_csv('pff_qb_nfl_22.csv')
pff_nfl_21 = read_csv('pff_qb_nfl_21.csv')
pff_nfl_20 = read_csv('pff_qb_nfl_20.csv')
pff_nfl_19 = read_csv('pff_qb_nfl_19.csv')
pff_nfl_18 = read_csv('pff_qb_nfl_18.csv')
pff_nfl_17 = read_csv('pff_qb_nfl_17.csv')
pff_nfl_16 = read_csv('pff_qb_nfl_16.csv')
pff_nfl_15 = read_csv('pff_qb_nfl_15.csv')

pff_nfl_22 = pff_nfl_22 |>
  mutate(year_id = 2022)
pff_nfl_21 = pff_nfl_21 |>
  mutate(year_id = 2021)
pff_nfl_20 = pff_nfl_20 |>
  mutate(year_id = 2020)
pff_nfl_19 = pff_nfl_19 |>
  mutate(year_id = 2019)
pff_nfl_18 = pff_nfl_18 |>
  mutate(year_id = 2018)
pff_nfl_17 = pff_nfl_17 |>
  mutate(year_id = 2017)
pff_nfl_16 = pff_nfl_16 |>
  mutate(year_id = 2016)
pff_nfl_15 = pff_nfl_15 |>
  mutate(year_id = 2015)

pff_nfl = pff_nfl_22 |>
  full_join(pff_nfl_22) |>
  full_join(pff_nfl_21) |>
  full_join(pff_nfl_20) |>
  full_join(pff_nfl_19) |>
  full_join(pff_nfl_18) |>
  full_join(pff_nfl_17) |>
  full_join(pff_nfl_16) |>
  full_join(pff_nfl_15)


# Key Stats by year
pff_nfl_2 = pff_nfl |>
  mutate(player_id = as.character(player_id)) |>
  select(player_id, player, year_id, grades_offense, pressure_to_sack_rate, avg_time_to_throw, dropbacks


# Totals of what we need -- average PFF grade
pff_nfl_3 = pff_nfl_2 |>
  filter(dropbacks >= 250) |>
  group_by(player_id, player) |>
  summarise(grade_avg = mean(grades_offense),
            .groups = 'drop') |>
  arrange(-grade_avg)
pff_nfl_3
```

```
## # A tibble: 75 x 3
##    player_id player          grade_avg
##    <chr>     <chr>               <dbl>
##  1 11765     Patrick Mahomes      89.4
##  2 698       Tom Brady            89.1
##  3 28022     Joe Burrow           86.3
##  4 802       Drew Brees           85.9
##  5 11767     Deshaun Watson       85.4
##  6 2241      Aaron Rodgers        85.1
##  7 40291     Jalen Hurts          82.6
##  8 28237     Justin Herbert       82.6
##  9 7077      Russell Wilson       82.0
## 10 46416     Lamar Jackson        81.2
## # ... with 65 more rows
```

Note: Because the grading is by year, I decided to define average PFF grade as the average of the grading over the seasons with 250+ dropbacks. This way, if a player is injured in a year - and plays a good chunck of the season - or the team passes less often, they still can be included. On top of this, joining this dataset with the play-by-play (PBP) dataset will eliminate small sample sizes (like Purdy).

This may look like a better list of ranking QBs, but there are still inconsistencies here. For instance, take Kirk Cousins' great passing ability but poor decision making (at times). Thus, he is ranked higher in PFF grading but lower in EPA. Or someone like Jimmy Garoppolo who isn't the best passer, but makes very optimal decisions.

Names

This data provides us with clean, consistent naming of the QBs.

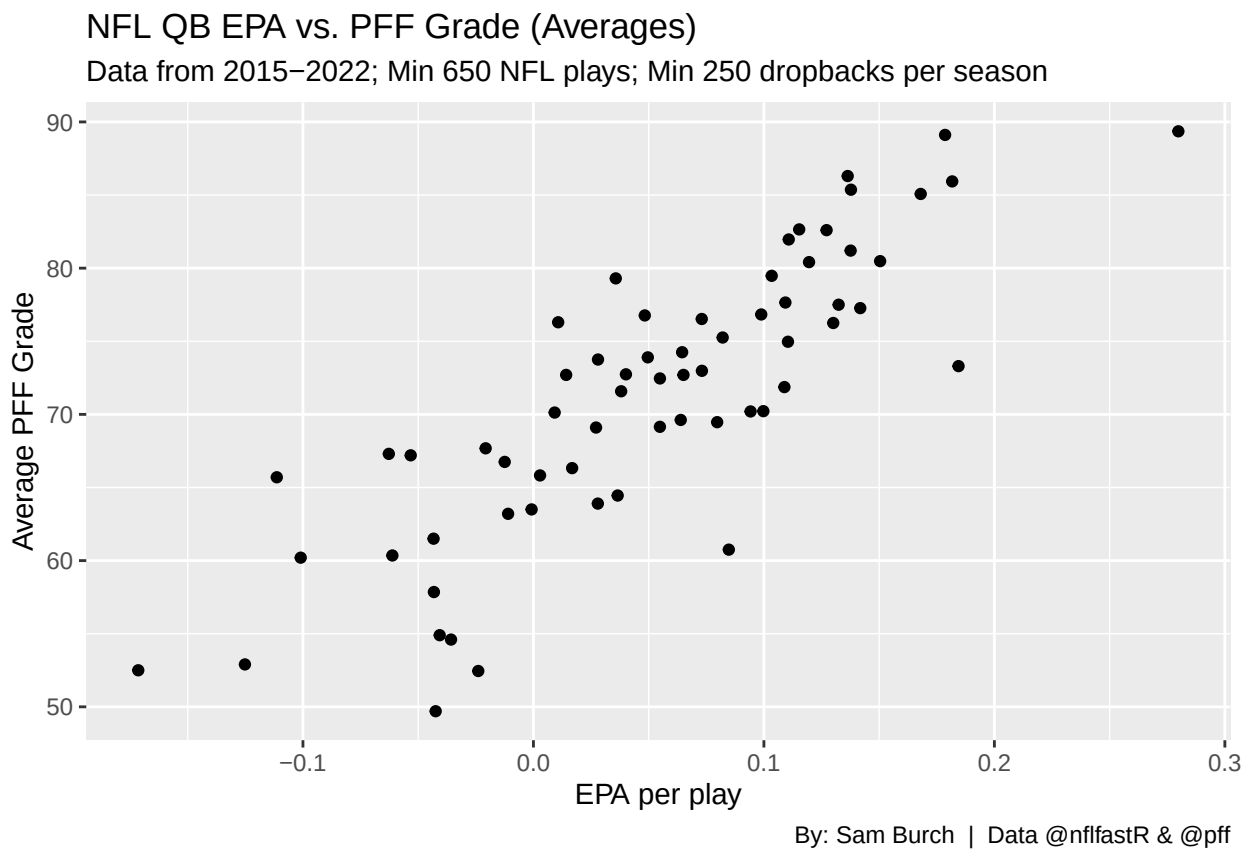
```
name_22 = load_player_stats(2022)
name_21 = load_player_stats(2021)
name_20 = load_player_stats(2020)
name_19 = load_player_stats(2019)
name_18 = load_player_stats(2018)
name_17 = load_player_stats(2017)
name_16 = load_player_stats(2016)
name_15 = load_player_stats(2015)

qb_names = name_22 |>
  full_join(name_21) |>
  full_join(name_20) |>
  full_join(name_19) |>
  full_join(name_18) |>
  full_join(name_17) |>
  full_join(name_16) |>
  full_join(name_15) |>
  filter(position == 'QB') |>
  select(player_id, player_display_name) |>
  unique()
```

Joining Data

```
# Adding grade and names to dataset
nfl_qbs = pbp_nfl_2 |>
  left_join(qb_names, by = c('passer_id_new' = 'player_id')) |>
  select(-passer_new, player_display_name, passer_id_new:plays) |>
  left_join(pff_nfl_3, by = c('player_display_name' = 'player')) |>
  select(player_display_name, everything(), -c(passer_id_new, player_id)) |>
  filter(!is.na(grade_avg))

# Comparison of EPA and Grade
ggplot(nfl_qbs, aes(mean_epa, grade_avg)) +
  labs(
    title = 'NFL QB EPA vs. PFF Grade (Averages)',
    subtitle = 'Data from 2015-2022; Min 650 NFL plays; Min 250 dropbacks per season',
    caption = 'By: Sam Burch  |  Data @nflfastR & @pff',
    x = 'EPA per play',
    y = 'Average PFF Grade'
  ) +
  geom_point()
```




```
z_score = function(values) {
  mu = mean(values)
  sig = sd(values)
```

```
  return((values - mu) / sig)
}

# Combined EPA and Grade to make new QBR metric
nfl_qbs = nfl_qbs |>
  mutate(qbr = .5*z_score(mean_epa) + .5*z_score(grade_avg)) |>
  mutate(qbr_pct = pnorm(qbr)) |>
  select(player_display_name, qbr, qbr_pct, mean_epa, grade_avg, everything()) |>
  arrange(-qbr)
nfl_qbs
```

```
## # A tibble: 64 x 9
##    player_display_name   qbr qbr_pct mean_epa grade_~1   ypa sack_~2  cpoe plays
##    <chr>               <dbl>   <dbl>    <dbl>    <dbl> <dbl>   <dbl> <dbl> <int>
##  1 Patrick Mahomes     2.30    0.989    0.280     89.4  6.97  0.0358 3.14   4213
##  2 Tom Brady           1.69    0.955    0.179     89.1  6.38  0.0364 0.891  6425
##  3 Drew Brees          1.54    0.938    0.182     85.9  6.84  0.0350 3.88   3709
##  4 Aaron Rodgers       1.42    0.921    0.168     85.1  6.04  0.0543 2.27   5469
##  5 Joe Burrow          1.29    0.902    0.136     86.3  6.07  0.0703 3.94   2175
##  6 Deshaun Watson      1.25    0.895    0.138     85.4  6.37  0.0764 3.20   2723
##  7 Josh Allen          1.07    0.857    0.150     80.5  6.15  0.0455 1.28   3777
##  8 Justin Herbert      1.04    0.852    0.127     82.6  6.13  0.0441 0.119  2359
##  9 Lamar Jackson       1.03    0.849    0.138     81.2  6.22  0.0545 0.308  2773
## 10 Jalen Hurts         0.976   0.836    0.115     82.6  6.07  0.0484 0.186  1716
## # ... with 54 more rows, and abbreviated variable names 1: grade_avg,
## #   2: sack_rate
```

```
# Average (40th - 60th Percentile)
nfl_qbs |>
  filter(qbr_pct <= .6 & qbr_pct >= .4)
```

```
## # A tibble: 15 x 9
##    player_display_~1      qbr qbr_pct mean_~2 grade~3   ypa sack_~4   cpoe plays
##    <chr>                <dbl>   <dbl>   <dbl>   <dbl> <dbl>   <dbl>  <dbl> <int>
##  1 Kyler Murray       0.233     0.592  0.0645    74.2  5.85  0.0509  0.772  2612
##  2 Jameis Winston     0.228     0.590  0.0998    70.2  6.34  0.0557  1.48   3466
##  3 Alex Smith         0.217     0.586  0.0731    73.0  5.99  0.0600  1.44   2683
##  4 Ryan Fitzpatrick   0.195     0.577  0.0942    70.2  6.29  0.0443 -0.450  2576
##  5 Jared Goff         0.155     0.562  0.0651    72.7  6.17  0.0472 -1.16   4258
##  6 Andy Dalton        0.127     0.551  0.0496    73.9  5.86  0.0579 -0.423  3885
##  7 Carson Wentz       0.0820    0.533  0.0549    72.5  5.45  0.0588 -0.714  4050
##  8 Teddy Bridgewater  0.0703    0.528  0.0797    69.5  5.94  0.0635  2.98   2046
##  9 Sam Bradford       0.0247    0.510  0.0107    76.3  5.77  0.0551  1.85   1379
## 10 Cam Newton         0.00997   0.504  0.0401    72.7  5.71  0.0525 -1.96   3656
## 11 Mac Jones         -0.00817   0.497  0.0280    73.8  5.76  0.0547  1.23   1189
## 12 Marcus Mariota    -0.0147    0.494  0.0639    69.6  6.02  0.0682  0.778  2828
## 13 Baker Mayfield    -0.0635    0.475  0.0381    71.6  5.81  0.0607 -1.42   2818
## 14 Gardner Minshew   -0.0927    0.463  0.0549    69.2  5.86  0.0613 -0.400  1158
## 15 Daniel Jones      -0.145     0.442  0.0142    72.7  5.43  0.0662 -0.821  2373
## # ... with abbreviated variable names 1: player_display_name, 2: mean_epa,
## #   3: grade_avg, 4: sack_rate
```

```
# Poor (<= 10th percentile)
nfl_qbs |>
  filter(qbr_pct <= .15)
```

```
## # A tibble: 11 x 9
##    player_display_name   qbr qbr_pct mean_epa grade~1   ypa sack_~2   cpoe plays
##    <chr>               <dbl>   <dbl>    <dbl>   <dbl> <dbl>   <dbl>  <dbl> <int>
##  1 Brian Hoyer         -1.08  0.141   -0.0433    61.5  5.69  0.0538 -2.56   1040
##  2 Sam Darnold         -1.24  0.107   -0.0612    60.4  5.41  0.0664 -2.04   2152
##  3 Blaine Gabbert      -1.25  0.105   -0.111     65.7  5.23  0.0695 -5.14    949
##  4 Brock Osweiler      -1.27  0.103   -0.0432    57.8  5.26  0.0569 -3.10   1407
##  5 Taylor Heinicke     -1.40  0.0815  -0.0358    54.6  5.65  0.0583  0.711  1081
##  6 Trevor Siemian      -1.41  0.0795  -0.0407    54.9  5.36  0.0610 -1.73   1263
##  7 Colin Kaepernick    -1.44  0.0750  -0.0239    52.4  5.46  0.0827 -4.95    774
##  8 Davis Mills         -1.48  0.0689  -0.101     60.2  5.31  0.0598 -2.21   1037
##  9 Kyle Allen          -1.69  0.0452  -0.0424    49.7  5.22  0.0708 -0.787   848
## 10 Zach Wilson         -2.01  0.0221  -0.125     52.9  4.89  0.0867 -8.10    773
## 11 DeShone Kizer       -2.31  0.0106  -0.171     52.5  4.83  0.0619 -6.89    678
## # ... with abbreviated variable names 1: grade_avg, 2: sack_rate
```

Because we are projecting a metric like above, there are many interpretations that can be had. For example, we could define an average QB as between 40th and 60th percentile. This would leave someone like Jared Goff to be a great example of an average QB – which makes sense. Another interpretation would be 90th percentile and above is elite. By this definition the elite consists of 5 players: 3 future HOFs and the 2 of the best QBs at the moment. Lastly, the poor category (as defined above) features Zach Wilson and Sam Darnold – 2 recent first round busts.

Now that we have our new QBR, we can move onto wrangling the college dataset.

College Data

For now, we will use the PFF dataset for college. The numbers may be slightly different than from the PBP dataset, but with the large sample size, these should be negligible.

PBP

```
pbp_cfb = load_cfb_pbp(2014:2022)
```

```
qbs_college = pbp_cfb |>
  group_by(passer_player_name) |>
  summarise(plays = n()) |>
  filter(plays >= 100,
         !is.na(passer_player_name)) |>
  pull(passer_player_name) |>
  unique()

# Just want EPA data here, as we will be using the PFF dataset mainly
cfb_epa = pbp_cfb |>
  filter(!is.na(yards_gained),
```

```
        (pass == 1 & (passer_player_name %in% qbs_college)) |
        (rush == 1 & (rusher_player_name %in% qbs_college))) |>
  mutate(passer_new = case_when(
        !is.na(passer_player_name) ~ passer_player_name,
        !is.na(rusher_player_name) ~ rusher_player_name),
        pass_epa = if_else(pass == 1, EPA, as.double(NA_integer_)),
        rush_epa = if_else(rush == 1, EPA, as.double(NA_integer_))
        ) |>
  group_by(passer_new) |>
  summarise(mean_epa = mean(EPA, na.rm = TRUE),
            rush_epa = mean(pass_epa, na.rm = TRUE),
            pass_epa = mean(rush_epa, na.rm = TRUE),
            ypa = mean(yards_gained, na.rm = TRUE),
            sacks = sum(sack_taken_stat, na.rm = TRUE),
            plays = n(),
            sack_rate = sacks/plays) |>
  arrange(desc(mean_epa))

cfb_epa = cfb_epa |>
  select(-c(sacks, sack_rate, plays))
cfb_epa
```

```
## # A tibble: 986 x 5
##    passer_new        mean_epa rush_epa pass_epa   ypa
##    <chr>                <dbl>    <dbl>    <dbl> <dbl>
##  1 Marcus Mariota       0.442    0.365    0.365  9.16
##  2 Mac Jones            0.436    0.446    0.446 10.0
##  3 Tua Tagovailoa       0.435    0.456    0.456  9.45
##  4 Lindsey Scott Jr.    0.419    0.431    0.431  9.18
##  5 Kyler Murray         0.413    0.403    0.403  9.27
##  6 C.J. Stroud          0.401    0.425    0.425  9.26
##  7 Grayson McCall       0.381    0.488    0.488  8.51
##  8 Baker Mayfield       0.371    0.401    0.401  8.88
##  9 Caleb Williams       0.371    0.324    0.324  8.21
## 10 Vernon Adams Jr.     0.365    0.444    0.444  8.71
## # ... with 976 more rows
```

Note: There are many more metrics we can get from this dataset, and this should be explored. For now though, we only want the EPA data.

The top of this table shows many NFL QBs, however they are all roughly average. This may be because certain weaknesses of EPA are more substantial in college – like surrounding talent and strength of schedule (SOS). Tua and Mac, for example, had a great supporting cast at Alabama with receivers like Devonta Smith and Jaylen Waddle. Hence, this may lead to EPA not being very predictive.

PFF

```
pff_cfb_22 = read_csv('pff_qb_cfb_22.csv')
pff_cfb_21 = read_csv('pff_qb_cfb_21.csv')
pff_cfb_20 = read_csv('pff_qb_cfb_20.csv')
pff_cfb_19 = read_csv('pff_qb_cfb_19.csv')
```

```
pff_cfb_18 = read_csv('pff_qb_cfb_18.csv')
pff_cfb_17 = read_csv('pff_qb_cfb_17.csv')
pff_cfb_16 = read_csv('pff_qb_cfb_16.csv')
pff_cfb_15 = read_csv('pff_qb_cfb_15.csv')
pff_cfb_14 = read_csv('pff_qb_cfb_14.csv')

pff_cfb_22 = pff_cfb_22 |>
  mutate(year_id = 2022)
pff_cfb_21 = pff_cfb_21 |>
  mutate(year_id = 2021)
pff_cfb_20 = pff_cfb_20 |>
  mutate(year_id = 2020)
pff_cfb_19 = pff_cfb_19 |>
  mutate(year_id = 2019)
pff_cfb_18 = pff_cfb_18 |>
  mutate(year_id = 2018)
pff_cfb_17 = pff_cfb_17 |>
  mutate(year_id = 2017)
pff_cfb_16 = pff_cfb_16 |>
  mutate(year_id = 2016)
pff_cfb_15 = pff_cfb_15 |>
  mutate(year_id = 2015)
pff_cfb_14 = pff_cfb_14 |>
  mutate(year_id = 2014)

pff_cfb = pff_cfb_22 |>
  full_join(pff_cfb_21) |>
  full_join(pff_cfb_20) |>
  full_join(pff_cfb_19) |>
  full_join(pff_cfb_18) |>
  full_join(pff_cfb_17) |>
  full_join(pff_cfb_16) |>
  full_join(pff_cfb_15) |>
  full_join(pff_cfb_14)
```

Joining Datasets

```
nfl_qbs2 = nfl_qbs |>
  select(player_display_name, qbr, qbr_pct)


df_final = pff_cfb |>
  filter(dropbacks >= 200) |>
  mutate(tot_ttt = avg_time_to_throw * dropbacks) |>
  group_by(player) |>
  summarise(off_grade_avg = mean(grades_offense),
            pass_grade_avg = mean(grades_pass),
            run_grade_avg = mean(grades_run),
            fum_grade_avg = mean(grades_hands_fumble),
            attempts = sum(attempts),
            dropbacks = sum(dropbacks),
```

```
            completions = sum(completions),
            cp = completions / attempts,
            yards = sum(yards),
            adj_cp = (completions + sum(drops)) / sum(aimed_passes),
            ypa = yards / attempts,
            tds = sum(touchdowns),
            ints = sum(interceptions),
            sacks = sum(sacks),
            sack_rate = sacks / dropbacks,
            pressures = sum(def_gen_pressures),
            pressure_to_sack_rate = sacks / pressures,
            ttt = sum(tot_ttt) / dropbacks,
            btts = sum(big_time_throws),
            btt_rate = btts / attempts,
            twps = sum(turnover_worthy_plays),
            twp_rate = twps / attempts,
            adot_avg = mean(avg_depth_of_target),
            bats = sum(bats),
            hats = sum(hit_as_threw),
            tas = sum(thrown_aways),
            scrambles = sum(scrambles),
            first_downs = sum(first_downs),
            nfl_pr = 100*(((completions/attempts - .3)*5 + (yards/attempts - 3)*.25 + (tds/attempts)*20
            penalties = sum(penalties),
            td_rate = tds / attempts,
            int_rate = ints / attempts,
            bat_rate = bats / attempts,
            ta_rate = tas / dropbacks,
            scramble_rate = scrambles / dropbacks,
            fd_rate = first_downs / dropbacks,
            pen_rate = penalties / dropbacks,
            .groups = 'drop') |>

  left_join(cfb_epa, by = c('player' = 'passer_new')) |>
  filter(!is.na(ypa.y)) |>

  left_join(nfl_qbs2, by = c('player' = 'player_display_name')) |>
  mutate(qbr_nfl = qbr,
         qbr_nfl_pct = qbr_pct,
         ypa_pass = ypa.x,
         ypa_tot = ypa.y) |>
  # Add back in pass_epa later
  select(-c(qbr, ypa.x, ypa.y))

df_final |>
  arrange(-mean_epa, -off_grade_avg) |>
  select(player, mean_epa, off_grade_avg)
```

```
## # A tibble: 570 x 3
##    player           mean_epa off_grade_avg
##    <chr>               <dbl>         <dbl>
##  1 Marcus Mariota      0.442          93
##  2 Mac Jones           0.436          95.8
```

```
##  3 Tua Tagovailoa      0.435          90.6
##  4 Kyler Murray        0.413          94.6
##  5 C.J. Stroud         0.401          90.6
##  6 Grayson McCall      0.381          90.1
##  7 Baker Mayfield      0.371          93.3
##  8 Caleb Williams      0.371          91.4
##  9 Vernon Adams Jr.    0.365          79.8
## 10 Justin Fields       0.321          92.5
## # ... with 560 more rows
```

Notes: There are some issues with pass_epa, so it has been removed for now. Also, the rate statistics were added to eliminate potential collinearity issues in the future. Two statistics that are based on volumes (e.g. tds and yards) will be correlated heavily. So, making td_rate or ypa will eliminated this volume element and focus of efficieny.

Stroud and Williams, whom we'll be looking at later, have performed very well in college.

```
# College players with little-to-no NFL experience
df_no_nfl = df_final |>
  filter(is.na(qbr_nfl))

# NFL players college performances
df_nfl = df_final |>
  filter(!is.na(qbr_nfl))

df_nfl |>
  arrange(-mean_epa, -off_grade_avg) |>
  select(player, mean_epa, off_grade_avg, qbr_pct)
```

```
## # A tibble: 29 x 4
##    player          mean_epa off_grade_avg qbr_pct
##    <chr>              <dbl>         <dbl>   <dbl>
##  1 Marcus Mariota     0.442          93     0.494
##  2 Mac Jones          0.436          95.8   0.497
##  3 Tua Tagovailoa     0.435          90.6   0.644
##  4 Kyler Murray       0.413          94.6   0.592
##  5 Baker Mayfield     0.371          93.3   0.475
##  6 Justin Fields      0.321          92.5   0.202
##  7 Joe Burrow         0.276          87.7   0.902
##  8 Trevor Lawrence    0.259          91.0   0.398
##  9 Jared Goff         0.238          89.2   0.562
## 10 Deshaun Watson     0.202          86.5   0.895
## # ... with 19 more rows
```

```
# Dataset for building our model
df_final_no_name = df_nfl |>
  select(-c(player, qbr_pct, qbr_nfl_pct, pass_epa))
```

Only NFL QBs above 80th percentile that were great in college are Joe Burrow and Deshaun Watson. However, Murray and Lawrence are among the best young QBs in the NFL also. One takeaway is, in general, it seems to be harder to be poor in the NFL if you were good in college. Similarly, it seems to be harder to be good in the NFL if you were poor in college. There are exceptions to both of these, but this seems to be a trend here.

Now we have a good starting point for college metrics. It is time to build the model!

Model Selection

The goal here is to fit a multiple linear regression model. We will hope to get a r^2 of at least .4, since this is a "strong" fit for football data (Pitcher List).

As we start the model selection process, there is one issue we run into if we try fitting a full model...

```
summary(lm(qbr_nfl ~ ., df_final_no_name))
```

```
##
## Call:
## lm(formula = qbr_nfl ~ ., data = df_final_no_name)
##
## Residuals:
## ALL 29 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (12 not defined because of singularities)
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.220e+01        NaN     NaN      NaN
## off_grade_avg            -7.094e-02        NaN     NaN      NaN
## pass_grade_avg            7.292e-02        NaN     NaN      NaN
## run_grade_avg            -1.173e-01        NaN     NaN      NaN
## fum_grade_avg             1.083e-01        NaN     NaN      NaN
## attempts                 -2.611e-01        NaN     NaN      NaN
## dropbacks                 2.692e-01        NaN     NaN      NaN
## completions              -2.798e-02        NaN     NaN      NaN
## cp                       -7.042e+01        NaN     NaN      NaN
## yards                    -1.758e-03        NaN     NaN      NaN
## adj_cp                    8.127e+01        NaN     NaN      NaN
## tds                      -9.446e-02        NaN     NaN      NaN
## ints                     -1.848e-01        NaN     NaN      NaN
## sacks                    -3.006e-01        NaN     NaN      NaN
## sack_rate                -1.937e+01        NaN     NaN      NaN
## pressures                 4.932e-02        NaN     NaN      NaN
## pressure_to_sack_rate    -2.350e+01        NaN     NaN      NaN
## ttt                       1.378e+00        NaN     NaN      NaN
## btts                      4.001e-01        NaN     NaN      NaN
## btt_rate                 -3.122e+02        NaN     NaN      NaN
## twps                     -1.229e-01        NaN     NaN      NaN
## twp_rate                  9.254e+01        NaN     NaN      NaN
## adot_avg                  5.932e-01        NaN     NaN      NaN
## bats                      1.182e-01        NaN     NaN      NaN
## hats                     -1.126e-01        NaN     NaN      NaN
## tas                      -9.068e-02        NaN     NaN      NaN
## scrambles                -2.676e-01        NaN     NaN      NaN
## first_downs              -2.258e-05        NaN     NaN      NaN
## nfl_pr                    2.076e-01        NaN     NaN      NaN
## penalties                        NA         NA      NA       NA
## td_rate                          NA         NA      NA       NA
## int_rate                         NA         NA      NA       NA
## bat_rate                         NA         NA      NA       NA
## ta_rate                          NA         NA      NA       NA
## scramble_rate                    NA         NA      NA       NA
## fd_rate                          NA         NA      NA       NA
```

```
## pen_rate                       NA         NA      NA       NA
## mean_epa                       NA         NA      NA       NA
## rush_epa                       NA         NA      NA       NA
## ypa_pass                       NA         NA      NA       NA
## ypa_tot                        NA         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 28 and 0 DF,  p-value: NA
```

Because of the sample size of NFL QBs that we want to predict, we cannot include all the other variables as predictors. Thus, we will try to eliminate those with a relatively high correlation, as these would've led to collinearity issues anyway.

```
# Need to fix dimension issues
dim(df_final_no_name)
```

```
## [1] 29 41
```

```
round(cor(df_final_no_name), 2)
```

```
##                        off_grade_avg pass_grade_avg run_grade_avg fum_grade_avg
## off_grade_avg                   1.00           0.97          0.53          0.44
## pass_grade_avg                  0.97           1.00          0.38          0.48
## run_grade_avg                   0.53           0.38          1.00          0.12
## fum_grade_avg                   0.44           0.48          0.12          1.00
## attempts                       -0.19          -0.19         -0.34         -0.19
## dropbacks                      -0.19          -0.20         -0.29         -0.22
## completions                    -0.07          -0.06         -0.31         -0.09
## cp                              0.78           0.81          0.41          0.58
## yards                           0.05           0.06         -0.20         -0.09
## adj_cp                          0.79           0.81          0.41          0.54
## tds                             0.23           0.25         -0.10         -0.06
## ints                           -0.39          -0.38         -0.47         -0.17
## sacks                          -0.21          -0.30          0.03         -0.45
## sack_rate                      -0.23          -0.31          0.22         -0.43
## pressures                      -0.25          -0.30         -0.19         -0.33
## pressure_to_sack_rate          -0.07          -0.12          0.31         -0.28
## ttt                             0.15           0.07          0.42         -0.28
## btts                            0.09           0.13         -0.26         -0.11
## btt_rate                        0.57           0.63          0.17          0.15
## twps                           -0.48          -0.48         -0.37         -0.39
## twp_rate                       -0.81          -0.80         -0.35         -0.54
## adot_avg                        0.21           0.18          0.27         -0.31
## bats                           -0.26          -0.30         -0.18         -0.18
## hats                           -0.22          -0.18         -0.35         -0.15
## tas                            -0.28          -0.32         -0.22         -0.39
## scrambles                      -0.02          -0.12          0.20         -0.29
## first_downs                     0.03           0.04         -0.13         -0.14
## nfl_pr                          0.77           0.78          0.52          0.29
## penalties                      -0.08          -0.04         -0.15         -0.26
## td_rate                         0.68           0.69          0.44          0.16
## int_rate                       -0.44          -0.42         -0.43         -0.01
```

```
## bat_rate                    -0.27          -0.32         -0.04         -0.22
## ta_rate                     -0.23          -0.26         -0.07         -0.41
## scramble_rate                0.14           0.04          0.43         -0.22
## fd_rate                      0.27           0.29          0.16         -0.01
## pen_rate                     0.06           0.10          0.19         -0.05
## mean_epa                     0.83           0.82          0.54          0.38
## rush_epa                     0.82           0.85          0.42          0.42
## qbr_nfl                      0.21           0.17          0.23          0.12
## ypa_pass                     0.75           0.75          0.55          0.30
## ypa_tot                      0.75           0.74          0.56          0.35
##                        attempts dropbacks completions    cp yards adj_cp   tds
## off_grade_avg             -0.19     -0.19       -0.07  0.78  0.05   0.79  0.23
## pass_grade_avg            -0.19     -0.20       -0.06  0.81  0.06   0.81  0.25
## run_grade_avg             -0.34     -0.29       -0.31  0.41 -0.20   0.41 -0.10
## fum_grade_avg             -0.19     -0.22       -0.09  0.58 -0.09   0.54 -0.06
## attempts                   1.00      0.99        0.98 -0.26  0.94  -0.12  0.81
## dropbacks                  0.99      1.00        0.98 -0.28  0.94  -0.13  0.82
## completions                0.98      0.98        1.00 -0.11  0.96   0.03  0.86
## cp                        -0.26     -0.28       -0.11  1.00 -0.03   0.96  0.13
## yards                      0.94      0.94        0.96 -0.03  1.00   0.09  0.94
## adj_cp                    -0.12     -0.13        0.03  0.96  0.09   1.00  0.23
## tds                        0.81      0.82        0.86  0.13  0.94   0.23  1.00
## ints                       0.83      0.82        0.76 -0.53  0.71  -0.43  0.53
## sacks                      0.66      0.73        0.62 -0.40  0.63  -0.26  0.53
## sack_rate                 -0.17     -0.08       -0.23 -0.42 -0.20  -0.39 -0.19
## pressures                  0.90      0.93        0.86 -0.41  0.83  -0.24  0.69
## pressure_to_sack_rate     -0.33     -0.26       -0.35 -0.15 -0.30  -0.17 -0.25
## ttt                       -0.28     -0.18       -0.28 -0.02 -0.16   0.02 -0.04
## btts                       0.84      0.82        0.85 -0.07  0.88   0.00  0.87
## btt_rate                  -0.21     -0.22       -0.16  0.39 -0.04   0.34  0.16
## twps                       0.89      0.90        0.82 -0.53  0.77  -0.43  0.60
## twp_rate                   0.09      0.11       -0.03 -0.76 -0.08  -0.83 -0.19
## adot_avg                  -0.01      0.04       -0.06 -0.19  0.10  -0.19  0.23
## bats                       0.70      0.70        0.65 -0.38  0.55  -0.21  0.37
## hats                       0.65      0.63        0.62 -0.31  0.52  -0.14  0.35
## tas                        0.73      0.76        0.69 -0.39  0.63  -0.20  0.51
## scrambles                  0.47      0.57        0.46 -0.22  0.50  -0.08  0.49
## first_downs                0.83      0.85        0.86  0.03  0.89   0.15  0.85
## nfl_pr                    -0.19     -0.18       -0.07  0.85  0.10   0.82  0.34
## penalties                  0.49      0.49        0.48 -0.12  0.51   0.00  0.46
## td_rate                   -0.15     -0.14       -0.06  0.66  0.12   0.63  0.40
## int_rate                  -0.06     -0.07       -0.15 -0.58 -0.20  -0.63 -0.30
## bat_rate                   0.14      0.15        0.08 -0.38 -0.02  -0.26 -0.15
## ta_rate                    0.10      0.14        0.05 -0.34  0.00  -0.20 -0.02
## scramble_rate             -0.15     -0.04       -0.16 -0.11 -0.09  -0.04  0.01
## fd_rate                    0.25      0.26        0.31  0.40  0.40   0.44  0.47
## pen_rate                  -0.37     -0.38       -0.37  0.16 -0.31   0.14 -0.24
## mean_epa                  -0.20     -0.19       -0.09  0.82  0.08   0.78  0.30
## rush_epa                  -0.18     -0.19       -0.05  0.87  0.11   0.82  0.33
## qbr_nfl                    0.38      0.41        0.41  0.13  0.44   0.22  0.47
## ypa_pass                  -0.23     -0.23       -0.13  0.78  0.07   0.73  0.28
## ypa_tot                   -0.20     -0.20       -0.09  0.81  0.09   0.75  0.29
##                         ints sacks sack_rate pressures pressure_to_sack_rate
## off_grade_avg          -0.39 -0.21     -0.23     -0.25                -0.07
```

```
## pass_grade_avg        -0.38 -0.30     -0.31     -0.30                -0.12
## run_grade_avg         -0.47  0.03      0.22     -0.19                 0.31
## fum_grade_avg         -0.17 -0.45     -0.43     -0.33                -0.28
## attempts               0.83  0.66     -0.17      0.90                -0.33
## dropbacks              0.82  0.73     -0.08      0.93                -0.26
## completions            0.76  0.62     -0.23      0.86                -0.35
## cp                    -0.53 -0.40     -0.42     -0.41                -0.15
## yards                  0.71  0.63     -0.20      0.83                -0.30
## adj_cp                -0.43 -0.26     -0.39     -0.24                -0.17
## tds                    0.53  0.53     -0.19      0.69                -0.25
## ints                   1.00  0.50     -0.17      0.78                -0.41
## sacks                  0.50  1.00      0.57      0.85                 0.35
## sack_rate             -0.17  0.57      1.00      0.16                 0.88
## pressures              0.78  0.85      0.16      1.00                -0.14
## pressure_to_sack_rate -0.41  0.35      0.88     -0.14                 1.00
## ttt                   -0.33  0.31      0.63      0.03                 0.51
## btts                   0.66  0.47     -0.23      0.68                -0.29
## btt_rate              -0.26 -0.26     -0.12     -0.28                -0.01
## twps                   0.87  0.67     -0.03      0.89                -0.27
## twp_rate               0.37  0.24      0.31      0.27                 0.08
## adot_avg               0.07  0.27      0.34      0.14                 0.26
## bats                   0.70  0.54     -0.03      0.66                -0.21
## hats                   0.62  0.43     -0.11      0.63                -0.31
## tas                    0.61  0.59      0.03      0.81                -0.26
## scrambles              0.35  0.77      0.41      0.70                 0.17
## first_downs            0.60  0.60     -0.17      0.73                -0.24
## nfl_pr                -0.48 -0.21     -0.25     -0.27                -0.01
## penalties              0.38  0.40      0.01      0.50                -0.03
## td_rate               -0.36 -0.15     -0.16     -0.20                 0.00
## int_rate               0.46 -0.09      0.02      0.02                -0.19
## bat_rate               0.26  0.20      0.14      0.19                -0.02
## ta_rate                0.12  0.21      0.20      0.28                -0.08
## scramble_rate         -0.16  0.38      0.59      0.15                 0.44
## fd_rate                0.05  0.13     -0.27      0.16                -0.17
## pen_rate              -0.37 -0.28      0.01     -0.36                 0.20
## mean_epa              -0.44 -0.23     -0.27     -0.26                -0.07
## rush_epa              -0.42 -0.30     -0.38     -0.30                -0.15
## qbr_nfl                0.23  0.29     -0.16      0.49                -0.39
## ypa_pass              -0.42 -0.23     -0.25     -0.29                -0.03
## ypa_tot               -0.40 -0.25     -0.32     -0.29                -0.09
##                         ttt  btts btt_rate  twps twp_rate adot_avg  bats  hats
## off_grade_avg          0.15  0.09     0.57 -0.48    -0.81     0.21 -0.26 -0.22
## pass_grade_avg         0.07  0.13     0.63 -0.48    -0.80     0.18 -0.30 -0.18
## run_grade_avg          0.42 -0.26     0.17 -0.37    -0.35     0.27 -0.18 -0.35
## fum_grade_avg         -0.28 -0.11     0.15 -0.39    -0.54    -0.31 -0.18 -0.15
## attempts              -0.28  0.84    -0.21  0.89     0.09    -0.01  0.70  0.65
## dropbacks             -0.18  0.82    -0.22  0.90     0.11     0.04  0.70  0.63
## completions           -0.28  0.85    -0.16  0.82    -0.03    -0.06  0.65  0.62
## cp                    -0.02 -0.07     0.39 -0.53    -0.76    -0.19 -0.38 -0.31
## yards                 -0.16  0.88    -0.04  0.77    -0.08     0.10  0.55  0.52
## adj_cp                 0.02  0.00     0.34 -0.43    -0.83    -0.19 -0.21 -0.14
## tds                   -0.04  0.87     0.16  0.60    -0.19     0.23  0.37  0.35
## ints                  -0.33  0.66    -0.26  0.87     0.37     0.07  0.70  0.62
## sacks                  0.31  0.47    -0.26  0.67     0.24     0.27  0.54  0.43
```

```
## sack_rate              0.63 -0.23    -0.12 -0.03     0.31     0.34 -0.03 -0.11
## pressures              0.03  0.68    -0.28  0.89     0.27     0.14  0.66  0.63
## pressure_to_sack_rate  0.51 -0.29    -0.01 -0.27     0.08     0.26 -0.21 -0.31
## ttt                    1.00 -0.21     0.15 -0.23    -0.02     0.58 -0.34 -0.35
## btts                  -0.21  1.00     0.32  0.68    -0.08     0.28  0.44  0.42
## btt_rate               0.15  0.32     1.00 -0.32    -0.36     0.57 -0.34 -0.23
## twps                  -0.23  0.68    -0.32  1.00     0.50     0.07  0.64  0.60
## twp_rate              -0.02 -0.08    -0.36  0.50     1.00     0.11  0.10  0.10
## adot_avg               0.58  0.28     0.57  0.07     0.11     1.00 -0.13 -0.12
## bats                  -0.34  0.44    -0.34  0.64     0.10    -0.13  1.00  0.81
## hats                  -0.35  0.42    -0.23  0.60     0.10    -0.12  0.81  1.00
## tas                    0.12  0.52    -0.25  0.72     0.18     0.07  0.56  0.55
## scrambles              0.54  0.35    -0.13  0.47     0.12     0.36  0.32  0.16
## first_downs           -0.07  0.81     0.08  0.69    -0.04     0.18  0.53  0.42
## nfl_pr                 0.23  0.08     0.54 -0.40    -0.62     0.23 -0.38 -0.30
## penalties             -0.10  0.42    -0.04  0.46     0.08     0.14  0.40  0.58
## td_rate                0.33  0.14     0.60 -0.31    -0.46     0.43 -0.35 -0.27
## int_rate              -0.08 -0.12    -0.18  0.15     0.53     0.09  0.10  0.05
## bat_rate              -0.18 -0.05    -0.28  0.15     0.10    -0.18  0.74  0.48
## ta_rate                0.42  0.01    -0.06  0.17     0.18     0.18  0.15  0.17
## scramble_rate          0.84 -0.12     0.12 -0.10     0.03     0.55 -0.11 -0.20
## fd_rate                0.03  0.40     0.43  0.15    -0.21     0.28  0.13  0.11
## pen_rate               0.02 -0.29     0.16 -0.32    -0.04     0.10 -0.23 -0.07
## mean_epa               0.22  0.04     0.51 -0.40    -0.60     0.27 -0.41 -0.31
## rush_epa               0.10  0.08     0.53 -0.40    -0.62     0.19 -0.41 -0.28
## qbr_nfl                0.14  0.34     0.03  0.32    -0.11     0.11  0.22  0.24
## ypa_pass               0.27  0.04     0.54 -0.40    -0.55     0.33 -0.41 -0.33
## ypa_tot                0.14  0.05     0.48 -0.38    -0.57     0.23 -0.35 -0.31
##                         tas scrambles first_downs nfl_pr penalties td_rate
## off_grade_avg         -0.28     -0.02        0.03   0.77     -0.08    0.68
## pass_grade_avg        -0.32     -0.12        0.04   0.78     -0.04    0.69
## run_grade_avg         -0.22      0.20       -0.13   0.52     -0.15    0.44
## fum_grade_avg         -0.39     -0.29       -0.14   0.29     -0.26    0.16
## attempts               0.73      0.47        0.83  -0.19      0.49   -0.15
## dropbacks              0.76      0.57        0.85  -0.18      0.49   -0.14
## completions            0.69      0.46        0.86  -0.07      0.48   -0.06
## cp                    -0.39     -0.22        0.03   0.85     -0.12    0.66
## yards                  0.63      0.50        0.89   0.10      0.51    0.12
## adj_cp                -0.20     -0.08        0.15   0.82      0.00    0.63
## tds                    0.51      0.49        0.85   0.34      0.46    0.40
## ints                   0.61      0.35        0.60  -0.48      0.38   -0.36
## sacks                  0.59      0.77        0.60  -0.21      0.40   -0.15
## sack_rate              0.03      0.41       -0.17  -0.25      0.01   -0.16
## pressures              0.81      0.70        0.73  -0.27      0.50   -0.20
## pressure_to_sack_rate -0.26      0.17       -0.24  -0.01     -0.03    0.00
## ttt                    0.12      0.54       -0.07   0.23     -0.10    0.33
## btts                   0.52      0.35        0.81   0.08      0.42    0.14
## btt_rate              -0.25     -0.13        0.08   0.54     -0.04    0.60
## twps                   0.72      0.47        0.69  -0.40      0.46   -0.31
## twp_rate               0.18      0.12       -0.04  -0.62      0.08   -0.46
## adot_avg               0.07      0.36        0.18   0.23      0.14    0.43
## bats                   0.56      0.32        0.53  -0.38      0.40   -0.35
## hats                   0.55      0.16        0.42  -0.30      0.58   -0.27
## tas                    1.00      0.61        0.57  -0.27      0.38   -0.20
```

```
## scrambles              0.61      1.00        0.58  -0.02      0.19    0.05
## first_downs            0.57      0.58        1.00   0.15      0.48    0.16
## nfl_pr                -0.27     -0.02        0.15   1.00      0.05    0.94
## penalties              0.38      0.19        0.48   0.05      1.00    0.07
## td_rate               -0.20      0.05        0.16   0.94      0.07    1.00
## int_rate              -0.02     -0.09       -0.23  -0.66     -0.16   -0.49
## bat_rate               0.20      0.09        0.07  -0.45      0.10   -0.43
## ta_rate                0.72      0.37        0.02  -0.23      0.00   -0.11
## scramble_rate          0.21      0.75        0.07   0.12     -0.09    0.22
## fd_rate                0.13      0.27        0.70   0.53      0.34    0.50
## pen_rate              -0.33     -0.30       -0.17   0.26      0.55    0.22
## mean_epa              -0.30     -0.01        0.09   0.96     -0.01    0.90
## rush_epa              -0.33     -0.10        0.14   0.96      0.03    0.88
## qbr_nfl                0.32      0.49        0.37   0.22      0.24    0.24
## ypa_pass              -0.33     -0.04        0.12   0.96      0.06    0.91
## ypa_tot               -0.33     -0.08        0.12   0.94      0.04    0.86
##                      int_rate bat_rate ta_rate scramble_rate fd_rate pen_rate
## off_grade_avg           -0.44    -0.27   -0.23          0.14    0.27     0.06
## pass_grade_avg          -0.42    -0.32   -0.26          0.04    0.29     0.10
## run_grade_avg           -0.43    -0.04   -0.07          0.43    0.16     0.19
## fum_grade_avg           -0.01    -0.22   -0.41         -0.22   -0.01    -0.05
## attempts                -0.06     0.14    0.10         -0.15    0.25    -0.37
## dropbacks               -0.07     0.15    0.14         -0.04    0.26    -0.38
## completions             -0.15     0.08    0.05         -0.16    0.31    -0.37
## cp                      -0.58    -0.38   -0.34         -0.11    0.40     0.16
## yards                   -0.20    -0.02    0.00         -0.09    0.40    -0.31
## adj_cp                  -0.63    -0.26   -0.20         -0.04    0.44     0.14
## tds                     -0.30    -0.15   -0.02          0.01    0.47    -0.24
## ints                     0.46     0.26    0.12         -0.16    0.05    -0.37
## sacks                   -0.09     0.20    0.21          0.38    0.13    -0.28
## sack_rate                0.02     0.14    0.20          0.59   -0.27     0.01
## pressures                0.02     0.19    0.28          0.15    0.16    -0.36
## pressure_to_sack_rate   -0.19    -0.02   -0.08          0.44   -0.17     0.20
## ttt                     -0.08    -0.18    0.42          0.84    0.03     0.02
## btts                    -0.12    -0.05    0.01         -0.12    0.40    -0.29
## btt_rate                -0.18    -0.28   -0.06          0.12    0.43     0.16
## twps                     0.15     0.15    0.17         -0.10    0.15    -0.32
## twp_rate                 0.53     0.10    0.18          0.03   -0.21    -0.04
## adot_avg                 0.09    -0.18    0.18          0.55    0.28     0.10
## bats                     0.10     0.74    0.15         -0.11    0.13    -0.23
## hats                     0.05     0.48    0.17         -0.20    0.11    -0.07
## tas                     -0.02     0.20    0.72          0.21    0.13    -0.33
## scrambles               -0.09     0.09    0.37          0.75    0.27    -0.30
## first_downs             -0.23     0.07    0.02          0.07    0.70    -0.17
## nfl_pr                  -0.66    -0.45   -0.23          0.12    0.53     0.26
## penalties               -0.16     0.10    0.00         -0.09    0.34     0.55
## td_rate                 -0.49    -0.43   -0.11          0.22    0.50     0.22
## int_rate                 1.00     0.21    0.12         -0.04   -0.41    -0.20
## bat_rate                 0.21     1.00    0.19         -0.03   -0.04    -0.05
## ta_rate                  0.12     0.19    1.00          0.39   -0.05    -0.22
## scramble_rate           -0.04    -0.03    0.39          1.00    0.12    -0.03
## fd_rate                 -0.41    -0.04   -0.05          0.12    1.00     0.27
## pen_rate                -0.20    -0.05   -0.22         -0.03    0.27     1.00
## mean_epa                -0.57    -0.54   -0.27          0.14    0.43     0.19
```

```
## rush_epa                -0.55    -0.53   -0.33          0.03    0.50     0.23
## qbr_nfl                 -0.18     0.01    0.04          0.25    0.18    -0.11
## ypa_pass                -0.49    -0.47   -0.28          0.13    0.52     0.31
## ypa_tot                 -0.53    -0.48   -0.33          0.03    0.49     0.27
##                       mean_epa rush_epa qbr_nfl ypa_pass ypa_tot
## off_grade_avg             0.83     0.82    0.21     0.75    0.75
## pass_grade_avg            0.82     0.85    0.17     0.75    0.74
## run_grade_avg             0.54     0.42    0.23     0.55    0.56
## fum_grade_avg             0.38     0.42    0.12     0.30    0.35
## attempts                 -0.20    -0.18    0.38    -0.23   -0.20
## dropbacks                -0.19    -0.19    0.41    -0.23   -0.20
## completions              -0.09    -0.05    0.41    -0.13   -0.09
## cp                        0.82     0.87    0.13     0.78    0.81
## yards                     0.08     0.11    0.44     0.07    0.09
## adj_cp                    0.78     0.82    0.22     0.73    0.75
## tds                       0.30     0.33    0.47     0.28    0.29
## ints                     -0.44    -0.42    0.23    -0.42   -0.40
## sacks                    -0.23    -0.30    0.29    -0.23   -0.25
## sack_rate                -0.27    -0.38   -0.16    -0.25   -0.32
## pressures                -0.26    -0.30    0.49    -0.29   -0.29
## pressure_to_sack_rate    -0.07    -0.15   -0.39    -0.03   -0.09
## ttt                       0.22     0.10    0.14     0.27    0.14
## btts                      0.04     0.08    0.34     0.04    0.05
## btt_rate                  0.51     0.53    0.03     0.54    0.48
## twps                     -0.40    -0.40    0.32    -0.40   -0.38
## twp_rate                 -0.60    -0.62   -0.11    -0.55   -0.57
## adot_avg                  0.27     0.19    0.11     0.33    0.23
## bats                     -0.41    -0.41    0.22    -0.41   -0.35
## hats                     -0.31    -0.28    0.24    -0.33   -0.31
## tas                      -0.30    -0.33    0.32    -0.33   -0.33
## scrambles                -0.01    -0.10    0.49    -0.04   -0.08
## first_downs               0.09     0.14    0.37     0.12    0.12
## nfl_pr                    0.96     0.96    0.22     0.96    0.94
## penalties                -0.01     0.03    0.24     0.06    0.04
## td_rate                   0.90     0.88    0.24     0.91    0.86
## int_rate                 -0.57    -0.55   -0.18    -0.49   -0.53
## bat_rate                 -0.54    -0.53    0.01    -0.47   -0.48
## ta_rate                  -0.27    -0.33    0.04    -0.28   -0.33
## scramble_rate             0.14     0.03    0.25     0.13    0.03
## fd_rate                   0.43     0.50    0.18     0.52    0.49
## pen_rate                  0.19     0.23   -0.11     0.31    0.27
## mean_epa                  1.00     0.98    0.27     0.94    0.95
## rush_epa                  0.98     1.00    0.21     0.94    0.95
## qbr_nfl                   0.27     0.21    1.00     0.20    0.20
## ypa_pass                  0.94     0.94    0.20     1.00    0.97
## ypa_tot                   0.95     0.95    0.20     0.97    1.00
```

High Correlations (> .5):

- off_grade_avg | pass_grade_avg | cp | adj_cp | btt_rate | twp_rate | fum_grade_avg | adot_avg | nfl_pr | td_rate | EPAs | YPAs
- attempts | dropbacks | completions | yards | tds | ints | sacks | pressures | btts | twps | bats | hats | tas | scrambles | first_downs | penalties

- sack_rate | pressure_to_sack_rate | ttt | scramble_rate

It is possible that I may have missed some, however, we will double check this later, so we can proceed.

Now that we have the different groups, let us fit a model to each group and eliminate as we go. To do this, each mini-model will have the number of predictors reduced through the Greedy Algorithm – using AIC as our optimization statistic. Then, the ones that are dropped through that process, will be dropped from our full dataset. This method is not perfect, but it will help us do a good job at dropping unnecessary predictors.

```
test_1 = lm(qbr_nfl ~ off_grade_avg + pass_grade_avg + cp + adj_cp + btt_rate + twp_rate + fum_grade_avg
summary(test_1)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ off_grade_avg + pass_grade_avg + cp +
##     adj_cp + btt_rate + twp_rate + fum_grade_avg + adot_avg +
##     nfl_pr + td_rate + mean_epa + rush_epa + ypa_tot + ypa_pass,
##     data = df_final)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9515 -0.4124 -0.2677  0.7161  1.6606
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -7.91525   23.07322  -0.343    0.737
## off_grade_avg    0.13213    0.19789   0.668    0.515
## pass_grade_avg  -0.14275    0.23283  -0.613    0.550
## cp             -46.87858   31.96947  -1.466    0.165
## adj_cp          51.43450   29.51003   1.743    0.103
## btt_rate        16.88356   41.90636   0.403    0.693
## twp_rate        51.88695   80.36981   0.646    0.529
## fum_grade_avg    0.02563    0.04866   0.527    0.607
## adot_avg        -0.27478    0.63694  -0.431    0.673
## nfl_pr          -0.02513    0.18693  -0.134    0.895
## td_rate         11.57622   59.32684   0.195    0.848
## mean_epa         6.95691   16.28401   0.427    0.676
## rush_epa         1.16064   13.95990   0.083    0.935
## ypa_tot         -0.20857    1.20238  -0.173    0.865
## ypa_pass         0.05461    1.18769   0.046    0.964
##
## Residual standard error: 1.275 on 14 degrees of freedom
##   (541 observations deleted due to missingness)
## Multiple R-squared:  0.3439, Adjusted R-squared:  -0.3122
## F-statistic: 0.5241 on 14 and 14 DF,  p-value: 0.8805
```

```
test_2 = lm(qbr_nfl ~ attempts + dropbacks + completions + yards + tds + ints + sacks + pressures + btts
summary(test_2)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ attempts + dropbacks + completions + yards +
##     tds + ints + sacks + pressures + btts + twps + bats + hats +
```

```
##     tas + scrambles + first_downs + penalties, data = df_final_no_name)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74339 -0.33853 -0.08541  0.27703  1.02920
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.643e-01  8.371e-01  -0.435   0.6712
## attempts     4.205e-02  9.074e-02   0.463   0.6514
## dropbacks   -3.499e-02  9.493e-02  -0.369   0.7189
## completions -1.061e-02  1.189e-02  -0.892   0.3898
## yards       -2.521e-04  4.754e-04  -0.530   0.6056
## tds          2.983e-02  2.854e-02   1.045   0.3165
## ints        -1.229e-01  4.891e-02  -2.514   0.0272 *
## sacks       -5.973e-02  1.097e-01  -0.545   0.5960
## pressures    3.183e-02  1.013e-02   3.142   0.0085 **
## btts        -3.413e-03  1.956e-02  -0.174   0.8644
## twps        -4.401e-02  5.555e-02  -0.792   0.4436
## bats         3.968e-02  4.553e-02   0.871   0.4006
## hats        -1.797e-02  5.824e-02  -0.309   0.7629
## tas         -6.393e-02  2.365e-02  -2.704   0.0192 *
## scrambles    5.141e-02  9.582e-02   0.537   0.6014
## first_downs  9.109e-05  3.195e-03   0.029   0.9777
## penalties    3.431e-03  8.763e-02   0.039   0.9694
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7597 on 12 degrees of freedom
## Multiple R-squared:  0.8004, Adjusted R-squared:  0.5342
## F-statistic: 3.007 on 16 and 12 DF,  p-value: 0.02977
```

```
test_3 = lm(qbr_nfl ~ sack_rate + pressure_to_sack_rate + ttt + scramble_rate, data = df_final_no_name)
summary(test_3)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ sack_rate + pressure_to_sack_rate + ttt +
##     scramble_rate, data = df_final_no_name)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.79931 -0.56811  0.01494  0.38730  1.88961
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.2131     3.1662   0.383  0.70499
## sack_rate                30.9823    26.6935   1.161  0.25719
## pressure_to_sack_rate   -25.8036     9.0299  -2.858  0.00868 **
## ttt                       0.2258     1.4191   0.159  0.87491
## scramble_rate            15.2106    12.0647   1.261  0.21952
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.9285 on 24 degrees of freedom
## Multiple R-squared:  0.4036, Adjusted R-squared:  0.3042
## F-statistic: 4.061 on 4 and 24 DF,  p-value: 0.01183
```

```
step(test_1, trace = 0)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ cp + adj_cp + mean_epa, data = df_final)
##
## Coefficients:
## (Intercept)           cp       adj_cp     mean_epa
##      -4.064      -36.177       35.475        4.372
```

```
step(test_2, trace = 0)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ attempts + completions + tds + ints +
##     sacks + pressures + twps + bats + tas + scrambles, data = df_final_no_name)
##
## Coefficients:
## (Intercept)     attempts  completions          tds         ints        sacks
##   -0.312994     0.007825    -0.013364     0.013980    -0.132033    -0.098949
##   pressures         twps         bats          tas    scrambles
##    0.031948    -0.044701     0.033761    -0.066176     0.018302
```

```
step(test_3, trace = 0)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ pressure_to_sack_rate + scramble_rate,
##     data = df_final_no_name)
##
## Coefficients:
##           (Intercept)  pressure_to_sack_rate          scramble_rate
##                 1.424                -16.551                 21.161
```

```
df_final_no_name_2 = df_final_no_name |>
  select(-c(off_grade_avg, pass_grade_avg, btt_rate, twp_rate, fum_grade_avg, adot_avg, nfl_pr, td_rate
```

Now, we will start by fitting the "full" model.

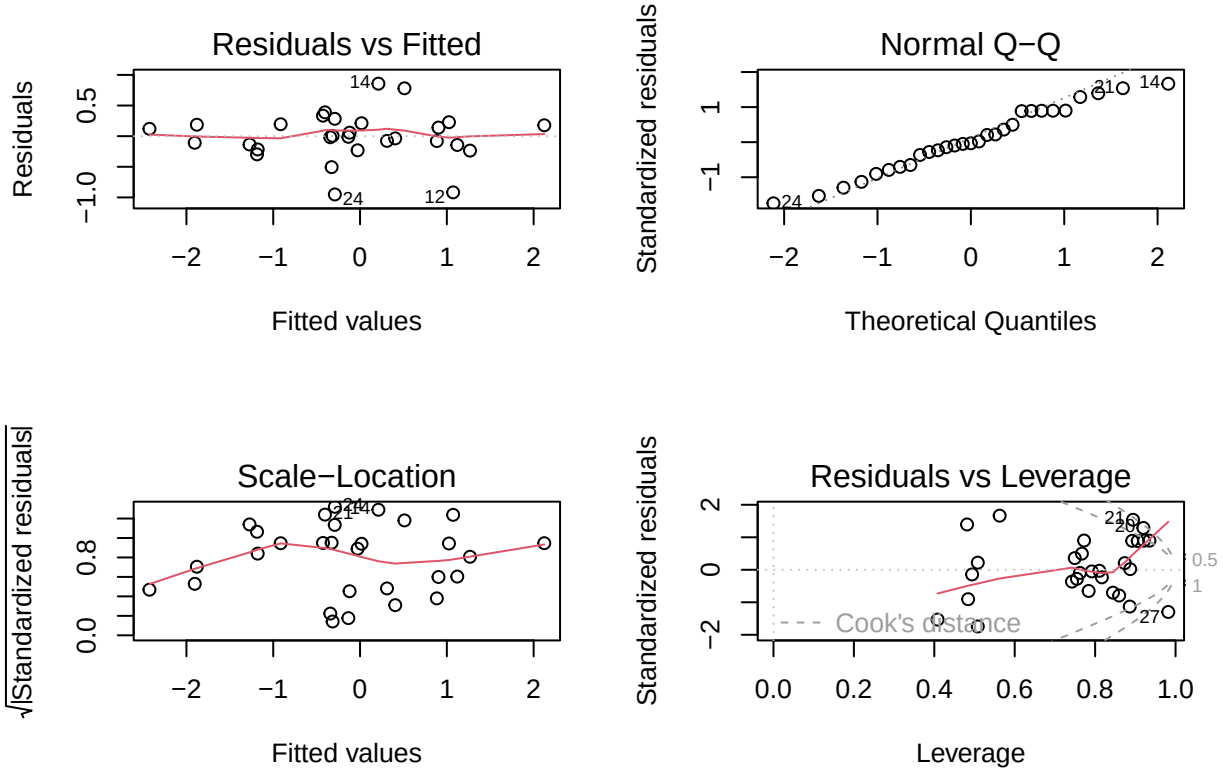
Model 1

```
m1 = lm(qbr_nfl ~ ., data = df_final_no_name_2)
summary(m1)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ ., data = df_final_no_name_2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.95152 -0.14380 -0.01057  0.19494  0.85687
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -7.213435  13.552297  -0.532    0.611
## run_grade_avg            0.005124   0.036313   0.141    0.892
## attempts                 0.014681   0.015079   0.974    0.363
## completions             -0.017054   0.019447  -0.877    0.410
## cp                      20.307119  38.317690   0.530    0.613
## adj_cp                  -9.971915  37.128579  -0.269    0.796
## tds                      0.006242   0.040390   0.155    0.882
## ints                    -0.180858   0.153278  -1.180    0.277
## sacks                   -0.044723   0.065403  -0.684    0.516
## pressures                0.021531   0.015053   1.430    0.196
## pressure_to_sack_rate -18.382690  18.444647  -0.997    0.352
## twps                    -0.017176   0.059319  -0.290    0.781
## bats                     0.017796   0.110102   0.162    0.876
## tas                     -0.128152   0.077304  -1.658    0.141
## scrambles                0.012918   0.026821   0.482    0.645
## int_rate                24.039520 106.379978   0.226    0.828
## bat_rate                10.544347 115.851590   0.091    0.930
## ta_rate                 67.743154  78.435656   0.864    0.416
## scramble_rate           17.142427  27.752606   0.618    0.556
## fd_rate                 -2.887436   2.388084  -1.209    0.266
## pen_rate               101.737244  97.515117   1.043    0.331
## mean_epa                -0.735111   6.371994  -0.115    0.911
##
## Residual standard error: 0.7781 on 7 degrees of freedom
## Multiple R-squared:  0.8779, Adjusted R-squared:  0.5115
## F-statistic: 2.396 on 21 and 7 DF,  p-value: 0.1194
```

```
# Good fit here

par(mfrow = c(2, 2))
plot(m1)
```




```
# Diagnostics look ok. We will move on.

library(caTools)
library(car)
sqrt(vif(m1))
```

```
##         run_grade_avg              attempts           completions
##              2.239273             35.643140              28.186599
##                    cp                adj_cp                   tds
##             12.409099             10.144659               7.155546
##                  ints                 sacks             pressures
##              9.149424             10.290134              12.991094
## pressure_to_sack_rate                  twps                  bats
##              5.207222              6.450952               6.453617
##                   tas             scrambles              int_rate
##              8.871952              6.431388               5.273719
##              bat_rate               ta_rate         scramble_rate
##              5.282256              5.965490               5.132471
##               fd_rate              pen_rate              mean_epa
##              2.342475              2.195384               5.995957
```

```
# Some clear collinearity issues still. We will move on for now.
```

Now, we will use the Greedy Algorithm - with AIC - to limit predictors again. This should improve the model and potentially help with diagnostics.

Model 2

```
step(m1, trace = 0)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ attempts + completions + cp + ints + sacks +
##     pressures + pressure_to_sack_rate + tas + scrambles + ta_rate +
##     fd_rate + pen_rate, data = df_final_no_name_2)
##
## Coefficients:
##           (Intercept)               attempts            completions
##               -5.94855                0.01161                -0.01276
##                     cp                   ints                  sacks
##                8.93743               -0.13795                -0.03388
##              pressures  pressure_to_sack_rate                    tas
##                0.01675              -19.54129                -0.14482
##              scrambles                ta_rate                 fd_rate
##                0.02933               89.75071                -3.34861
##              pen_rate
##             123.14225
```

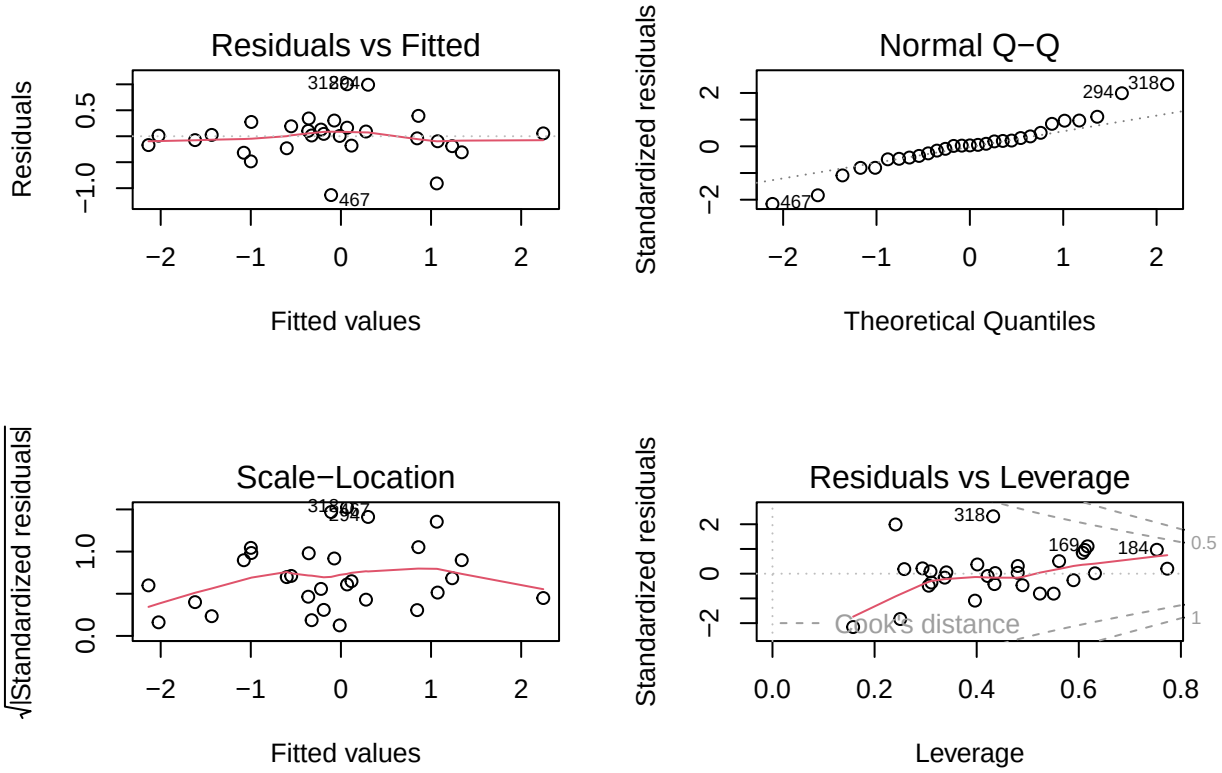
```
m2 = lm(qbr_nfl ~ attempts + completions + cp + ints +
    sacks + pressures + pressure_to_sack_rate + tas + scrambles +
    ta_rate + fd_rate + pen_rate, data = df_final)
summary(m2)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ attempts + completions + cp + ints + sacks +
##     pressures + pressure_to_sack_rate + tas + scrambles + ta_rate +
##     fd_rate + pen_rate, data = df_final)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.13279 -0.18105  0.01433  0.16477  1.00063
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -5.948547   5.337027  -1.115 0.281493
## attempts                 0.011610   0.005426   2.140 0.048121 *
## completions             -0.012756   0.007470  -1.708 0.107029
## cp                       8.937431   6.562160   1.362 0.192078
## ints                    -0.137948   0.037094  -3.719 0.001867 **
## sacks                   -0.033881   0.029135  -1.163 0.261927
## pressures                0.016749   0.006627   2.527 0.022400 *
## pressure_to_sack_rate  -19.541287   8.155954  -2.396 0.029153 *
## tas                     -0.144817   0.045884  -3.156 0.006116 **
## scrambles                0.029335   0.007077   4.145 0.000761 ***
## ta_rate                 89.750711  42.281885   2.123 0.049735 *
## fd_rate                 -3.348614   1.305911  -2.564 0.020797 *
## pen_rate               123.142248  53.736406   2.292 0.035832 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5721 on 16 degrees of freedom
##   (541 observations deleted due to missingness)
## Multiple R-squared:  0.8491, Adjusted R-squared:  0.7359
## F-statistic: 7.501 on 12 and 16 DF,  p-value: 0.0001691
```

```
# Great fit here!

par(mfrow = c(2, 2))
plot(m2)
```




```
sqrt(vif(m2))
```

```
##              attempts           completions                    cp
##             17.442590             14.724790              2.890312
##                  ints                 sacks             pressures
##              3.011446              6.234438              7.778522
## pressure_to_sack_rate                   tas             scrambles
##              3.131608              7.162034              2.308090
##               ta_rate               fd_rate              pen_rate
##              4.373648              1.742191              1.645372
```

```
#Better, but collinearity issues are too much.
```

Because this did not fix the diagnostic issues (specifically collinearity) we will take a step back. Going back to when we dropped the predictors from each group, we will consider dropping more.

The first group was narrowed down to cp, adj_cp, and mean_epa.

```
drop1(test_1) |>
  arrange(-AIC)
```

```
## Single term deletions
##
## Model:
## qbr_nfl ~ off_grade_avg + pass_grade_avg + cp + adj_cp + btt_rate +
##     twp_rate + fum_grade_avg + adot_avg + nfl_pr + td_rate +
##     mean_epa + rush_epa + ypa_tot + ypa_pass
##                Df Sum of Sq    RSS    AIC
## adj_cp          1    4.9397 27.704 26.674
## cp              1    3.4963 26.261 25.123
## <none>                      22.764 22.979
## off_grade_avg   1    0.7249 23.489 21.888
## twp_rate        1    0.6777 23.442 21.830
## pass_grade_avg  1    0.6112 23.376 21.748
## fum_grade_avg   1    0.4509 23.215 21.548
## adot_avg        1    0.3026 23.067 21.362
## mean_epa        1    0.2968 23.061 21.355
## btt_rate        1    0.2639 23.028 21.314
## td_rate         1    0.0619 22.826 21.058
## ypa_tot         1    0.0489 22.813 21.041
## nfl_pr          1    0.0294 22.794 21.017
## rush_epa        1    0.0112 22.776 20.994
## ypa_pass        1    0.0034 22.768 20.984
```

Since cp and adj_cp are very similar, and adj_cp is a little better here, we will drop cp. We will also keep mean_epa and come back if we still have issues later.

The second group consists of mainly volume-based statistics. Because many of these facets of the game are considered in the rate statistics, we should choose what represents volume the most. Out of these options, we will choose dropbacks. Attempts would be another option, but this eliminates situations like scrambles. Ideally, we would have a stat like games played; this should be considered in the future.

Finally, the for third group, we didn't seem to have collinearity issues with the predictors that we kept. So, we will keep both pressure_to_sack_rate and scramble_rate. Again, if issues show up later, we will come back to this step and consider this further.

```
df_final_no_name_3 = df_final_no_name |>
  select(-c(off_grade_avg, pass_grade_avg, btt_rate, twp_rate, fum_grade_avg, adot_avg, nfl_pr, td_rate
```

We will fit the "full" model again.

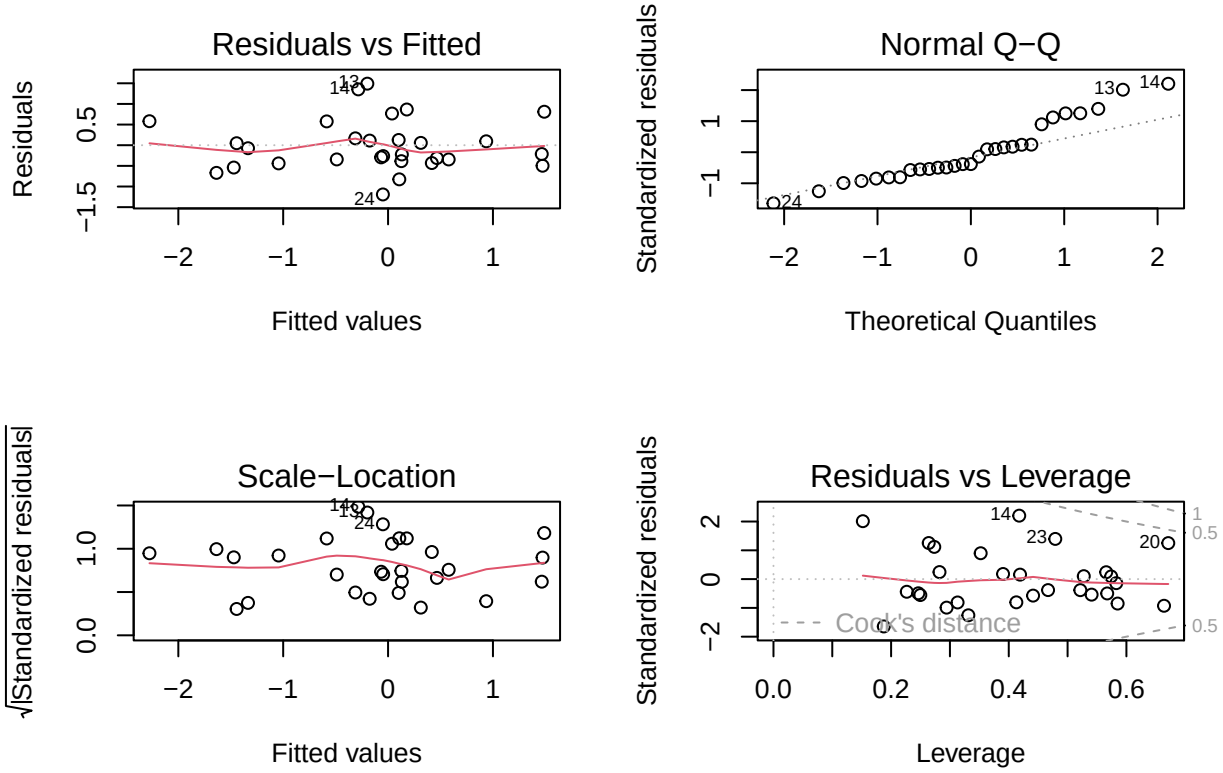
Model 3

```r
m3 = lm(qbr_nfl ~ ., data = df_final_no_name_3)
summary(m3)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ ., data = df_final_no_name_3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1922 -0.3858 -0.2135  0.1666  1.4907
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1.654e+00  6.434e+00   0.257  0.80017
## run_grade_avg           2.171e-02  2.633e-02   0.825  0.42104
## dropbacks               1.388e-03  5.611e-04   2.474  0.02420 *
## adj_cp                 -8.235e-01  7.443e+00  -0.111  0.91320
## pressure_to_sack_rate -2.165e+01  5.806e+00  -3.728  0.00167 **
## int_rate               -4.255e+01  3.547e+01  -1.200  0.24677
## bat_rate                1.612e+01  3.224e+01   0.500  0.62351
## ta_rate                -2.940e+01  1.807e+01  -1.627  0.12211
## scramble_rate           2.838e+01  9.112e+00   3.114  0.00631 **
## fd_rate                -3.104e+00  1.512e+00  -2.053  0.05579 .
## pen_rate                6.664e+01  5.804e+01   1.148  0.26681
## mean_epa                7.251e-01  2.783e+00   0.261  0.79758
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8044 on 17 degrees of freedom
## Multiple R-squared:  0.6829, Adjusted R-squared:  0.4778
## F-statistic: 3.329 on 11 and 17 DF,  p-value: 0.01297
```

```r
# Good model. Should get better when limiting predictors

par(mfrow = c(2, 2))
plot(m3)
```




```
sqrt(vif(m3))
```

```
##         run_grade_avg             dropbacks                adj_cp
##              1.570656              1.413177              1.966909
## pressure_to_sack_rate              int_rate              bat_rate
##              1.585380              1.700777              1.421575
##               ta_rate         scramble_rate               fd_rate
##              1.329336              1.629835              1.434324
##              pen_rate              mean_epa
##              1.263823              2.532817
```

```
# Looks better
# Still slight collinearity issues, but almost gone
```

Applying the greedy algorithm again.

Model 4

```
step(m3, trace = 0)
```

```
##
## Call:
```

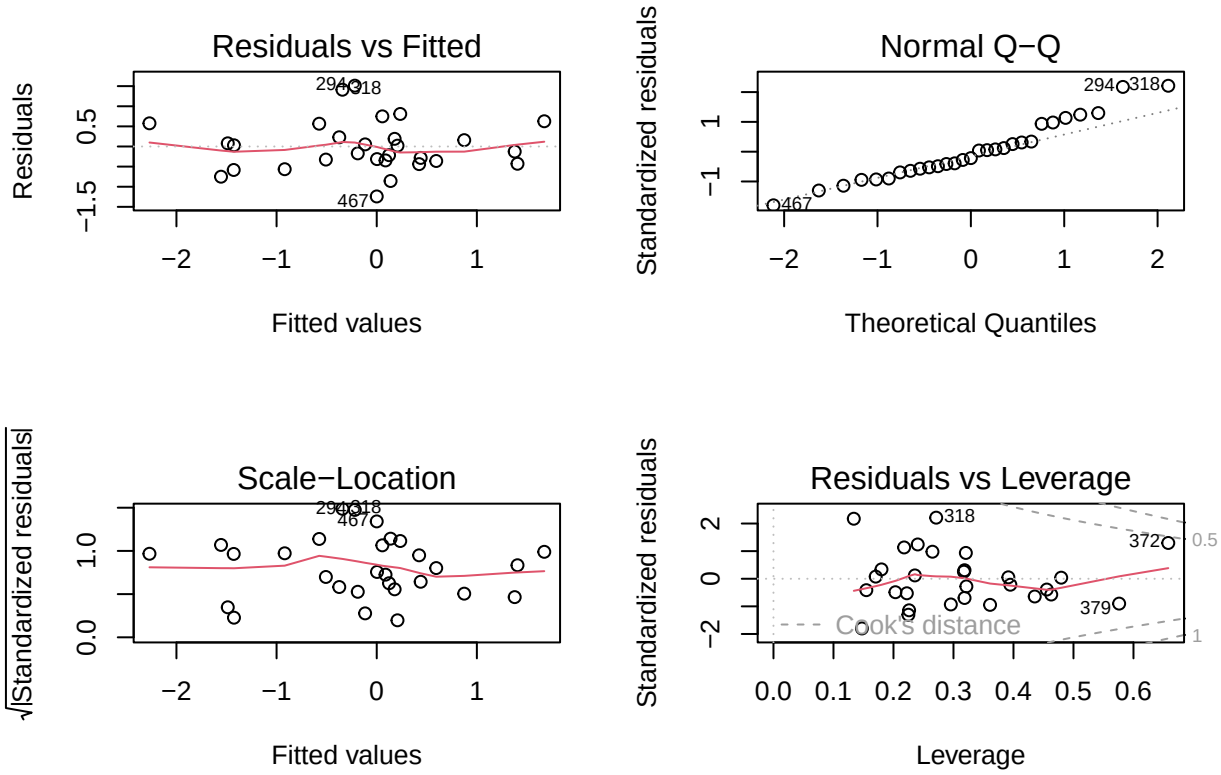
```
## lm(formula = qbr_nfl ~ run_grade_avg + dropbacks + pressure_to_sack_rate +
##     int_rate + ta_rate + scramble_rate + fd_rate + pen_rate,
##     data = df_final_no_name_3)
##
## Coefficients:
##           (Intercept)          run_grade_avg               dropbacks
##              1.021381               0.026847               0.001412
## pressure_to_sack_rate               int_rate                ta_rate
##            -21.823558             -40.732182             -29.085136
##         scramble_rate                fd_rate               pen_rate
##             28.126224              -2.977329              67.934521
```

```
m4 = lm(qbr_nfl ~ run_grade_avg + dropbacks + pressure_to_sack_rate +
    int_rate + ta_rate + scramble_rate + fd_rate + pen_rate, data = df_final)
summary(m4)
```

```
##
## Call:
## lm(formula = qbr_nfl ~ run_grade_avg + dropbacks + pressure_to_sack_rate +
##     int_rate + ta_rate + scramble_rate + fd_rate + pen_rate,
##     data = df_final)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2418 -0.3607 -0.1257  0.2291  1.5119
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.021e+00  2.201e+00   0.464 0.647567
## run_grade_avg          2.685e-02  2.143e-02   1.253 0.224638
## dropbacks              1.412e-03  4.698e-04   3.006 0.006982 **
## pressure_to_sack_rate -2.182e+01  4.606e+00  -4.738 0.000126 ***
## int_rate              -4.073e+01  2.556e+01  -1.593 0.126753
## ta_rate               -2.909e+01  1.523e+01  -1.910 0.070555 .
## scramble_rate          2.813e+01  7.971e+00   3.528 0.002111 **
## fd_rate               -2.977e+00  1.326e+00  -2.245 0.036237 *
## pen_rate               6.793e+01  5.298e+01   1.282 0.214451
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7472 on 20 degrees of freedom
##   (541 observations deleted due to missingness)
## Multiple R-squared:  0.6782, Adjusted R-squared:  0.5494
## F-statistic: 5.268 on 8 and 20 DF,  p-value: 0.001212
```

```
# R^2 of .55

par(mfrow = c(2, 2))
plot(m4)
```




```
sqrt(vif(m4))
```

```
##         run_grade_avg             dropbacks pressure_to_sack_rate
##              1.375740              1.273776              1.354170
##              int_rate               ta_rate         scramble_rate
##              1.319587              1.205855              1.535045
##               fd_rate              pen_rate
##              1.354596              1.242096
```

```
# Should be good, but we will check further to make sure...
```

Diagnostics in depth

```
sort(cooks.distance(m4), decreasing = TRUE)[1:10]
```

```
##         372         318         379         294         467         162         184
## 0.35894886 0.20232206 0.12249351 0.08117751 0.06224537 0.05614045 0.05476843
##         326         353         170
## 0.05425352 0.04586561 0.04238490
```

Since all CDs are less than 1, this means there are no highly influential points (HIPs).

```
#Some high-residual points:
df_final[c(467, 294, 318), ]
```

```
## # A tibble: 3 x 45
##   player     off_g~1 pass_~2 run_g~3 fum_g~4 attem~5 dropb~6 compl~7    cp yards
##   <chr>        <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <dbl> <dbl>
## 1 Sam Darno~    84.0    84.4    58.5    36.7     853     955     549 0.644  7221
## 2 Joe Burrow    87.7    87      71.3    60.7     904    1053     620 0.686  8558
## 3 Josh Allen    74.4    75.4    59.4    45.4     649     774     361 0.556  5014
## # ... with 35 more variables: adj_cp <dbl>, tds <dbl>, ints <dbl>, sacks <dbl>,
## #   sack_rate <dbl>, pressures <dbl>, pressure_to_sack_rate <dbl>, ttt <dbl>,
## #   btts <dbl>, btt_rate <dbl>, twps <dbl>, twp_rate <dbl>, adot_avg <dbl>,
## #   bats <dbl>, hats <dbl>, tas <dbl>, scrambles <dbl>, first_downs <dbl>,
## #   nfl_pr <dbl>, penalties <dbl>, td_rate <dbl>, int_rate <dbl>,
## #   bat_rate <dbl>, ta_rate <dbl>, scramble_rate <dbl>, fd_rate <dbl>,
## #   pen_rate <dbl>, mean_epa <dbl>, rush_epa <dbl>, pass_epa <dbl>, ...
```

These all make sense. Sam Darnold was a third overall pick, but has been a poor NFL QB. Hence, it makes sense he's been below his projections. Meanwhile, Joe Burrow and Josh Allen have developed to be some of the best QBs in the league. How would one predict this with Joe Burrow having a great surrounding cast in college and with Josh Allen being a poor passer in college?

```
library(lmtest)
bptest(m4)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  m4
## BP = 3.5513, df = 8, p-value = 0.8952
```

With the p-value $= .895 > .05 = \alpha$, we fail to reject the null and conclude the constant variance assumption is satisfied.

```
# Size of n = 29 -> shapiro test
shapiro.test(m4$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  m4$residuals
## W = 0.95994, p-value = 0.3277
```

With the p-value $= .3277 > .05 = \alpha$, we fail to reject the null and conclude the normality assumption is satisfied.

Projections

Now that we have our final model, we will apply this model to college QBs.

```
proj_m = predict(m4, newdata = df_no_nfl)


df_aug_no_nfl = df_no_nfl |>
  mutate(proj_nfl_qbr = proj_m) |>
  mutate(proj_pct = pnorm(proj_nfl_qbr)) |>
  arrange(-proj_nfl_qbr) |>
  select(player, proj_nfl_qbr, proj_pct, off_grade_avg, mean_epa, everything())
df_aug_no_nfl
```

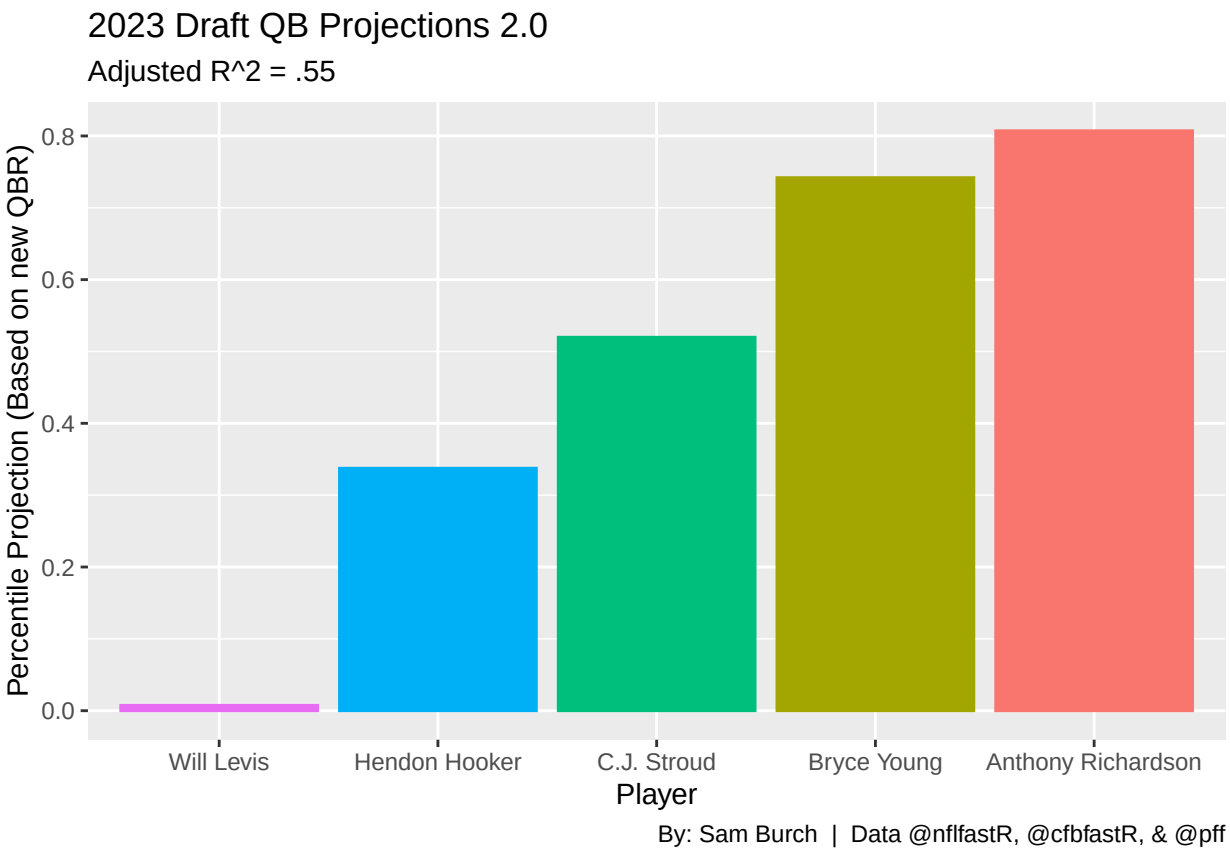
```
## # A tibble: 541 x 47
##    player        proj_~1 proj_~2 off_g~3 mean_~4 pass_~5 run_g~6 fum_g~7 attem~8
##    <chr>           <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 Cameron Risi~    2.74   0.997    81.9  0.225     77      75.8    58.3     707
##  2 Diego Pavia      2.59   0.995    82.9  0.189     67      89.7    74.4     190
##  3 Quinton Flow~    2.51   0.994    82.3  0.162     74.6    76.7    67.4     967
##  4 Riley Leonard    2.18   0.986    77.8  0.161     69.4    80      74.4     392
##  5 Spencer Sand~    1.99   0.977    73.2  0.0718    69.6    68.3    68.4    1259
##  6 Holton Ahlers    1.87   0.969    75.7  0.0901    72.4    67.4    62.0    1869
##  7 Sam Ehlinger     1.84   0.967    80.5  0.148     77.3    69.8    73.1    1484
##  8 Trevone Boyk~    1.82   0.966    84.8  0.178     83.3    66.6    70.1     890
##  9 Maty Mauk        1.76   0.961    68.8 -0.0158    65.5    71.2    79.1     419
## 10 Bo Nix           1.75   0.960    75.4  0.134     71.6    69.7    55.2    1478
## # ... with 531 more rows, 38 more variables: dropbacks <dbl>,
## #   completions <dbl>, cp <dbl>, yards <dbl>, adj_cp <dbl>, tds <dbl>,
## #   ints <dbl>, sacks <dbl>, sack_rate <dbl>, pressures <dbl>,
## #   pressure_to_sack_rate <dbl>, ttt <dbl>, btts <dbl>, btt_rate <dbl>,
## #   twps <dbl>, twp_rate <dbl>, adot_avg <dbl>, bats <dbl>, hats <dbl>,
## #   tas <dbl>, scrambles <dbl>, first_downs <dbl>, nfl_pr <dbl>,
## #   penalties <dbl>, td_rate <dbl>, int_rate <dbl>, bat_rate <dbl>, ...
```

```
proj_23_top_5 = df_aug_no_nfl |>
  filter(player == 'Bryce Young' |
         player == 'C.J. Stroud' |
         player == 'Will Levis'  |
         player == 'Anthony Richardson' |
         player == 'Hendon Hooker'
         ) |>
  select(player:mean_epa)
proj_23_top_5
```

```
## # A tibble: 5 x 5
##   player             proj_nfl_qbr proj_pct off_grade_avg mean_epa
##   <chr>                     <dbl>    <dbl>         <dbl>    <dbl>
## 1 Anthony Richardson       0.868   0.807           80.1    0.157
## 2 Bryce Young              0.649   0.742           91.8    0.249
## 3 C.J. Stroud              0.0496  0.520           90.6    0.401
## 4 Hendon Hooker           -0.419   0.337           87.6    0.258
## 5 Will Levis              -2.43    0.00745         79.6    0.104
```

```
ggplot(proj_23_top_5, aes(x = reorder(player, proj_nfl_qbr), y = proj_pct)) +
  geom_col(aes(color = player, fill = player)) +
```

```
  labs(
    x = 'Player',
    y = 'Percentile Projection (Based on new QBR)',
    title = '2023 Draft QB Projections 2.0',
    subtitle = 'Adjusted R^2 = .55',
    caption = 'By: Sam Burch  |  Data @nflfastR, @cfbfastR, & @pff'
  ) +
  theme(legend.position = "none")
```




```
# ggsave("qb_proj_2.png", width = 16, height = 9, units = "cm")
```

Note: These projections are version 2.0 because I have been working on building this model the right way.

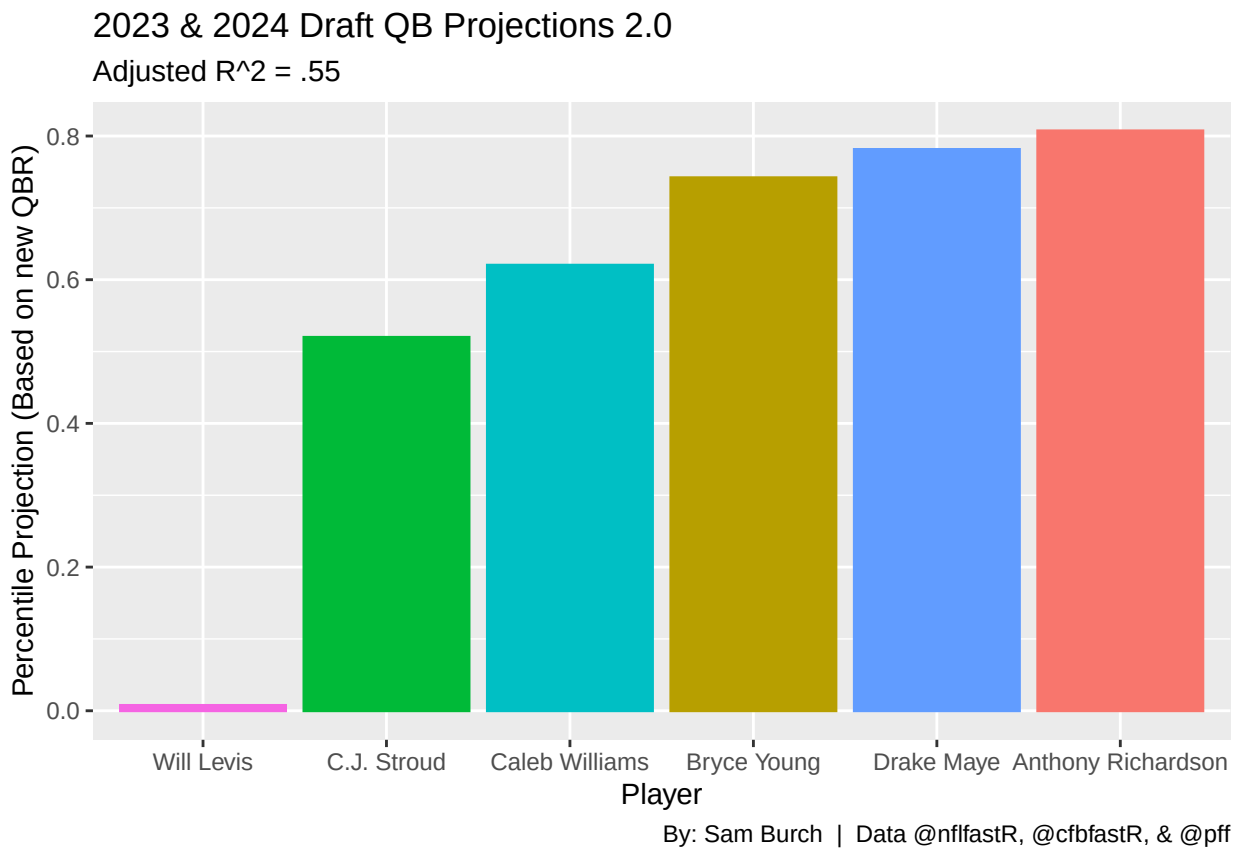
These projections love Richardson and hate Levis. With these two QBs both being "raw" prospects, why might this be the case? Richardson has elite rushing ability (84.8 PFF rush grade) and pocket awareness (.09 PTSR). Meanwhile, Levis has poor pocket awareness (.24 PTSR). The question with Richardson is if he can fix his accuracy (64% adjusted CP) which has been proven to be stable from college to pro (Josh Hermsmeyer). However, this model doesn't see accuracy as predictive of NFL performance as the other variables - since CP and adjusted CP were dropped from the model.

Bryce Young and C.J. Stroud, the more productive QBs in the class, are interesting too. Young projects well (74th percentile) but is being knocked for his height (5'10''). Maybe this is a worthy knock on him, however he still seems like a great QB prospect. On the other hand, Stroud had arguably better production. However, not being a purely pocket passer seemed to hurt him. The interesting part here is Stroud has shown his running ability before, but he just didn't use it much in college.

```r
# Adding in Maye and Williams
top_upcoming_qbs = df_aug_no_nfl |>
  filter(player == 'Bryce Young' |
         player == 'C.J. Stroud' |
         player == 'Will Levis'  |
         player == 'Anthony Richardson' |
         player == 'Caleb Williams' |
         player == 'Drake Maye'
         ) |>
  select(player:mean_epa)
top_upcoming_qbs
```

```
## # A tibble: 6 x 5
##   player             proj_nfl_qbr proj_pct off_grade_avg mean_epa
##   <chr>                     <dbl>    <dbl>         <dbl>    <dbl>
## 1 Anthony Richardson       0.868   0.807            80.1    0.157
## 2 Drake Maye               0.777   0.781            91.5    0.261
## 3 Bryce Young              0.649   0.742            91.8    0.249
## 4 Caleb Williams           0.306   0.620            91.4    0.371
## 5 C.J. Stroud              0.0496  0.520            90.6    0.401
## 6 Will Levis              -2.43    0.00745          79.6    0.104
```

```r
ggplot(top_upcoming_qbs, aes(x = reorder(player, proj_nfl_qbr), y = proj_pct)) +
  geom_col(aes(color = player, fill = player)) +
  labs(
    x = 'Player',
    y = 'Percentile Projection (Based on new QBR)',
    title = '2023 & 2024 Draft QB Projections 2.0',
    subtitle = 'Adjusted R^2 = .55',
    caption = 'By: Sam Burch  |  Data @nflfastR, @cfbfastR, & @pff'
  ) +
  theme(legend.position = "none")
```




```
# ggsave("qb_proj_2_more.png", width = 16, height = 9, units = "cm")
```

Drake Maye and Caleb Williams are being hyped up so much that people are looking down on this year's class. While they both project well, they are right in line with Young, Stroud, and Richardson. Hence, for people stressing teams in the top 10 to tank for them next year, that might not be a bad idea, but only if they miss out on this year's good group of QBs.

```
df_aug_no_nfl |>
  filter(player == 'Matt Corral' |
         player == 'Malik Willis' |
         player == 'Kenny Pickett' |
         player == 'Desmond Ridder' |
         player == 'Sam Howell' |
         player == 'Brock Purdy'
         ) |>
  select(player:mean_epa)
```

```
## # A tibble: 6 x 5
##   player         proj_nfl_qbr proj_pct off_grade_avg mean_epa
##   <chr>                 <dbl>    <dbl>         <dbl>    <dbl>
## 1 Matt Corral           1.68     0.954          82.2  0.243
## 2 Brock Purdy           1.67     0.953          81.8  0.147
## 3 Kenny Pickett         1.35     0.911          74.4  0.00776
## 4 Malik Willis          0.352    0.637          88.6  0.226
## 5 Desmond Ridder       -0.358    0.360          78.3  0.111
## 6 Sam Howell           -1.19     0.117          88.5  0.221
```

Looking at QBs who were drafted last year, these projections really liked Corral, Purdy, and Pickett. All three QBs were projected to be 90th percentile or better. However, all but one of these QBs were picked relatively low in the draft. Thus, unless NFL teams are clueless, there's something this model got wrong. This is possibly because the sample size of QBs had a certain amount of plays under center. This is a potential bias as QBs don't play if they are not good. Hence, this model shouldn't necessarily be trusted at predicting these late round guys. Although, Purdy doesn't look like a bad projection thus far and Pickett looked okay last year.

Conclusion

This is by no means a perfect model. However, for football data, .55 r^2 is very good. And assuming our metric for NFL success is 95% of QB play, that "translates" to this model being "correct" about 52.5% of the time. This may sound low, but we can use other facets to improve our prediction greatly.

This model is based on the following predictors: run_grade_avg, dropbacks, pressure_to_sack_rate, int_rate, ta_rate, scramble_rate, fd_rate, and pen_rate. So, avoiding sacks, running, and decision-making are key here. These all seem to be skills where you either are good at it or not; also, these may be hard skills to develop in the NFL. What is strange, though, is accuracy has been proven both to matter in the NFL and to translate from college to the NFL. Maybe due to new data of people like Josh Allen and Justin Herbert becoming more accurate and people like Baker Mayfield and Justin Fields being accurate in college is one reason for this. Hence, it is worth exploring if accuracy is still predictive of QB success, or how that has changed.

As a NFL fan, I really liked Stroud coming into this process and didn't like Levis. This was due to their respective production (EPA and PFF grading). However, as a statistician, I realized college production is relatively unstable when translating to the NFL. This helped me see just how good Young is, how Richardson is the "raw" prospect to take a chance on over Levis, and that Stroud may very well just be average in the NFL.

In the future, this model can be improved a variety of ways - outlined throughout this paper. I believe this is a good start though, and this can clearly tell us a lot about how good college QBs will be in the NFL!

References

- The-Ringer
- Ben-Baldwin
- NFL-Football-Operations
- Pitcher-List
- Josh-Hermsmeyer