Final Project - Predicting March Madness

Anmol Sapru and Rohit Gunda

The following packages were used:

```
library(tidyverse)
library(tidymodels)
library(Stat2Data)
library(caret)
library(leaps)
library(MASS)
```

Data was found from the Sports Reference college basketball team stats website and Bart Torvik analytics website, with these general and deeper stats being taken going back to 2008 (excluding the cancelled March Madness of 2020). From here, the data was cleaned in Excel to separate the year and result of each team then joined for all of this data together.

For further cleaning, the non-March Madness teams added in the join were removed and simple variable names and values were cleaned up to be easier to work with.

```
Rows: 5234 Columns: 23
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): TEAM
dbl (22): RK, G, WINS, LOSSES, ADJOE, ADJDE, EFG%, EFGD%, TOR, TORD, ORB, DR...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 4878 Columns: 14
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): School
dbl (13): G, Overall W-L%, Overall SRS, Overall SOS, Conf. W-L%, Home W-L%, ...

i Use `spec()` to retrieve the full column specification for this data.
```

```
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 896 Columns: 4
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (3): TEAM, march_madness, Conference
dbl (1): YEAR

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
Joining, by = "TEAM"
```

We determined it would be best to try different regressions, with logistic regressions by round and an ordinal regression—each with their different assumptions to look into. The issue with the first of these is the low levels of data that limit the creation of a model beyond around the Elite Eight. In preparing for the round-by-round logistic regressions we split up the data from the original data sets into multiple data sets for each round, with a TRUE or FALSE value of whether they won their game in that round.

The following is an example of the regression that was ran on all of the Round of 64 teams to create a regression that predicts winners (round_64 = TRUE) against losers (round_64 = FALSE) for the round. Using stepAIC works to attempt to limit the overfitting with the data. This was also done for the Round of 32 and Sweet Sixteen.

```
round_64 <- na.omit(round_64)

round_64_max <- glm(round_64 ~ G.x + WINS + LOSSES + ADJOE + ADJDE +
                      `EFG%` + `EFGD%` + TOR + TORD + ORB + DRB + FTR +
                      FTRD + `2P%` + `2P%D` + `3P%` + `3P%D` + `3PR` +
                      `3PRD` + `ADJ T.` + `Overall SRS` + `Overall SOS` +
                      `Conf. W-L%` + `Home W-L%` + `Away W-L%` +
                      `AVG PPG` + `AVG DPPG` + `AVG PD` + `AST/TOV` +
                      `PF/G` + WINS*G.x + LOSSES*G.x + ADJOE*ADJDE +
                      `EFG%`*`EFGD%` + TOR*TORD + ORB*DRB + FTR*FTRD +
                      `2P%`*`2P%D` + `3P%`*`3P%D` + `3PR`*`3PRD` +
                      `2P%`*`3P%` + `2P%D`*`3P%D` + `AVG PPG`*`AVG DPPG` +
                      WINS*`Overall SRS` + WINS*`Overall SOS` +
                      LOSSES*`Overall SRS` + LOSSES*`Overall SOS`,
                      data = round_64,
                      family = "binomial")

round_64_min <- glm(round_64 ~ 1,
                    data = round_64,
```

```
                    family = "binomial")

round_64_model <- stepAIC(round_64_max,
        scope = list(lower = round_64_min, upper = round_64_max),
        data = round_64, direction = "both")
```

The following are the regression models that were found from the three rounds of data and regression models run through stepAIC

```
tidy(round_64_model)
```

```
# A tibble: 32 x 5
   term        estimate std.error statistic  p.value
   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
 1 (Intercept) -73.4     11.0         -6.66 2.78e-11
 2 G.x          -0.0225   0.0113      -1.98 4.74e- 2
 3 WINS          0.0282   0.0121       2.34 1.92e- 2
 4 ADJOE         0.108    0.0910       1.19 2.34e- 1
 5 ADJDE         0.209    0.101        2.07 3.88e- 2
 6 `EFG%`        1.17     0.371        3.16 1.60e- 3
 7 `EFGD%`       1.12     0.381        2.93 3.35e- 3
 8 TORD          0.0362   0.0137       2.64 8.40e- 3
 9 ORB           0.0139   0.00747      1.86 6.27e- 2
10 FTR          -0.0531   0.0241      -2.20 2.76e- 2
# ... with 22 more rows
```

```
tidy(round_32_model)
```

```
# A tibble: 25 x 5
   term         estimate std.error statistic p.value
   <chr>           <dbl>     <dbl>     <dbl>   <dbl>
 1 (Intercept) -35.8      13.7         -2.61 0.00908
 2 ADJOE         0.306     0.110        2.79 0.00519
 3 ADJDE         0.351     0.125        2.80 0.00518
 4 `EFG%`       -0.827     0.259       -3.20 0.00139
 5 `EFGD%`      -0.641     0.243       -2.64 0.00826
 6 TOR          -0.211     0.118       -1.79 0.0729
 7 TORD         -0.167     0.110       -1.52 0.128
 8 FTR          -0.00909   0.00629     -1.45 0.148
 9 `2P%`         0.337     0.135        2.51 0.0122
```

```
10 `3P%`         0.672     0.246        2.74 0.00622
# ... with 15 more rows
```

```
  tidy(sweet_sixteen_model)
```

```
# A tibble: 21 x 5
   term         estimate std.error statistic  p.value
   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
 1 (Intercept) -126.      17.7         -7.12 1.08e-12
 2 ADJOE          0.747    0.156        4.79 1.69e- 6
 3 ADJDE          0.912    0.182        5.01 5.41e- 7
 4 `EFGD%`        0.347    0.195        1.78 7.53e- 2
 5 TOR           -0.366    0.170       -2.16 3.11e- 2
 6 TORD          -0.267    0.156       -1.71 8.67e- 2
 7 FTR           -0.0139   0.00865     -1.60 1.09e- 1
 8 `2P%`         -0.539    0.204       -2.64 8.41e- 3
 9 `2P%D`        -0.872    0.269       -3.24 1.19e- 3
10 `3P%`         -0.381    0.227       -1.68 9.39e- 2
# ... with 11 more rows
```

```
  `2023sr` <- read_csv("data/2023sportsreference.csv")
```

```
Rows: 363 Columns: 13
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): School
dbl (12): G, Overall W-L%, Overall SRS, Overall SOS, Conf. W-L%, Home W-L%, ...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
  `2023analytics` <- read_csv("data/2023torvik.csv")
```

```
Rows: 363 Columns: 22
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): TEAM
dbl (21): RK, G, WINS, LOSSES, ADJOE, ADJDE, EFG%, EFGD%, TOR, TORD, ORB, DR...
```

```
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
`2023stats` <- left_join(`2023analytics`, `2023sr`, by = c("TEAM" = "School"))

`2023teams` <- read_csv("data/2023teams.csv")
```

```
Rows: 64 Columns: 1
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): TEAM

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
`2023stats` <- `2023stats` |>
  filter(`2023stats`$TEAM %in% `2023teams`$TEAM)

tibble(predict(round_64_model, `2023stats`)) |>
  mutate(rank = seq(1:64)) |>
  left_join(mutate(`2023stats`, rank = seq(1:64))) |>
  arrange(desc(predict(round_64_model, `2023stats`)))
```

```
Joining, by = "rank"
```

```
# A tibble: 64 x 36
   predi~1  rank    RK TEAM    G.x  WINS LOSSES ADJOE ADJDE `EFG%` `EFGD%`   TOR
     <dbl> <int> <dbl> <chr> <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>   <dbl> <dbl>
 1    3.90     3     3 UCLA     34    29      5  113.  87.4   50.9    46.8  15.3
 2    3.20    13    13 Kans~    34    27      7  113.  91.5   52.4    47.1  17.5
 3    2.76     1     1 Hous~    34    31      3  117.  88     52.7    42.5  15.3
 4    2.68     2     2 Alab~    34    29      5  115.  88.3   52.7    41.5  19
 5    2.50     9     9 Texas    34    26      8  115.  91.6   52.7    47.8  16.5
 6    2.21     6     6 Purd~    34    29      5  118.  92.6   52.2    47.2  17
 7    2.11    12    12 San ~    32    26      6  111.  90.1   50.1    47.5  17.6
 8    1.94    10    10 Gonz~    32    27      5  123.  98.6   58.5    51.7  14.6
 9    1.93     8     8 Sain~    32    25      7  112.  89.1   52.5    46.7  16.4
10    1.86    11    11 Marq~    34    28      6  119.  96.1   56      51.1  15.2
```

```
# ... with 54 more rows, 24 more variables: TORD <dbl>, ORB <dbl>, DRB <dbl>,
#   FTR <dbl>, FTRD <dbl>, `2P%` <dbl>, `2P%D` <dbl>, `3P%` <dbl>,
#   `3P%D` <dbl>, `3PR` <dbl>, `3PRD` <dbl>, `ADJ T.` <dbl>, G.y <dbl>,
#   `Overall W-L%` <dbl>, `Overall SRS` <dbl>, `Overall SOS` <dbl>,
#   `Conf. W-L%` <dbl>, `Home W-L%` <dbl>, `Away W-L%` <dbl>, `AVG PPG` <dbl>,
#   `AVG DPPG` <dbl>, `AVG PD` <dbl>, `AST/TOV` <dbl>, `PF/G` <dbl>, and
#   abbreviated variable name 1: `predict(round_64_model, \`2023stats\`)`
```

```
tibble(predict(round_32_model, `2023stats`)) |>
  mutate(rank = seq(1:64)) |>
  left_join(mutate(`2023stats`, rank = seq(1:64))) |>
  arrange(desc(predict(round_32_model, `2023stats`)))
```

```
Joining, by = "rank"
```

```
# A tibble: 64 x 36
   predi~1  rank    RK TEAM    G.x  WINS LOSSES ADJOE ADJDE `EFG%` `EFGD%`   TOR
     <dbl> <int> <dbl> <chr> <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>   <dbl> <dbl>
 1   1.77      1     1 Hous~    34    31      3  117.  88     52.7    42.5  15.3
 2   1.71      3     3 UCLA     34    29      5  113.  87.4   50.9    46.8  15.3
 3   1.68      2     2 Alab~    34    29      5  115.  88.3   52.7    41.5  19
 4   1.22     10    10 Gonz~    32    27      5  123.  98.6   58.5    51.7  14.6
 5   1.08      6     6 Purd~    34    29      5  118.  92.6   52.2    47.2  17
 6   0.945    13    13 Kans~    34    27      7  113.  91.5   52.4    47.1  17.5
 7   0.913     9     9 Texas    34    26      8  115.  91.6   52.7    47.8  16.5
 8   0.821     4     4 Tenn~    33    23     10  111.  86.2   50.3    42.4  18.1
 9   0.774     5     5 Conn~    33    25      8  119.  92.5   53.5    45.5  18.9
10   0.754     8     8 Sain~    32    25      7  112.  89.1   52.5    46.7  16.4
# ... with 54 more rows, 24 more variables: TORD <dbl>, ORB <dbl>, DRB <dbl>,
#   FTR <dbl>, FTRD <dbl>, `2P%` <dbl>, `2P%D` <dbl>, `3P%` <dbl>,
#   `3P%D` <dbl>, `3PR` <dbl>, `3PRD` <dbl>, `ADJ T.` <dbl>, G.y <dbl>,
#   `Overall W-L%` <dbl>, `Overall SRS` <dbl>, `Overall SOS` <dbl>,
#   `Conf. W-L%` <dbl>, `Home W-L%` <dbl>, `Away W-L%` <dbl>, `AVG PPG` <dbl>,
#   `AVG DPPG` <dbl>, `AVG PD` <dbl>, `AST/TOV` <dbl>, `PF/G` <dbl>, and
#   abbreviated variable name 1: `predict(round_32_model, \`2023stats\`)`
```

```
tibble(predict(sweet_sixteen_model, `2023stats`)) |>
  mutate(rank = seq(1:64)) |>
  left_join(mutate(`2023stats`, rank = seq(1:64))) |>
  arrange(desc(predict(sweet_sixteen_model, `2023stats`)))
```

```
Joining, by = "rank"
```

```
# A tibble: 64 x 36
   predict(swe~1  rank    RK TEAM    G.x  WINS LOSSES ADJOE ADJDE `EFG%` `EFGD%`
           <dbl> <int> <dbl> <chr> <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>   <dbl>
 1       1.49       13    13 Kans~    34    27      7  113.  91.5   52.4    47.1
 2       1.39        3     3 UCLA     34    29      5  113.  87.4   50.9    46.8
 3       1.39        1     1 Hous~    34    31      3  117.  88     52.7    42.5
 4       1.19        2     2 Alab~    34    29      5  115.  88.3   52.7    41.5
 5       0.530       6     6 Purd~    34    29      5  118.  92.6   52.2    47.2
 6       0.425       9     9 Texas    34    26      8  115.  91.6   52.7    47.8
 7       0.240      22    22 Texa~    34    25      9  113.  94.8   49      47.9
 8       0.129      12    12 San ~    32    26      6  111.  90.1   50.1    47.5
 9       0.00623    11    11 Marq~    34    28      6  119.  96.1   56      51.1
10      -0.00623     5     5 Conn~    33    25      8  119.  92.5   53.5    45.5
# ... with 54 more rows, 25 more variables: TOR <dbl>, TORD <dbl>, ORB <dbl>,
#   DRB <dbl>, FTR <dbl>, FTRD <dbl>, `2P%` <dbl>, `2P%D` <dbl>, `3P%` <dbl>,
#   `3P%D` <dbl>, `3PR` <dbl>, `3PRD` <dbl>, `ADJ T.` <dbl>, G.y <dbl>,
#   `Overall W-L%` <dbl>, `Overall SRS` <dbl>, `Overall SOS` <dbl>,
#   `Conf. W-L%` <dbl>, `Home W-L%` <dbl>, `Away W-L%` <dbl>, `AVG PPG` <dbl>,
#   `AVG DPPG` <dbl>, `AVG PD` <dbl>, `AST/TOV` <dbl>, `PF/G` <dbl>, and
#   abbreviated variable name ...
```