Hiring Committee

Evan Woods

2023-11-11

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Setup and Installation

Install Packages

```
install.packages("tidyverse")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
if (!require(dplyr)) install.packages("dplyr")
if (!require(ggplot2)) install.packages("ggplot2")
if (!require(jtools)) install.packages("jtools")
```

```
## Loading required package: jtools
```

```
if (!require(caTools)) install.packages("caTools")
```

```
## Loading required package: caTools
```

```
if (!require(gmodels)) install.packages("gmodels")
```

```
## Loading required package: gmodels
```

```
if(!require(stargazer)) install.packages("stargazer")
```

```
## Loading required package: stargazer
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2022). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.3. https://CRAN.R-project.org/package=stargazer
```

Plot Save Location

```
plot_save_base_local = "/Users/evanwoods/Github/lpa/Round3/plots/"
plot_save_base_cloud = "/cloud/project/Hiring_Committee/plots/"
plot_save_base = plot_save_base_cloud
```

Read CSV location

```
read_csv_local = "/Users/evanwoods/Github/lpa/Round3/"
read_csv_cloud = "/cloud/project/Hiring_Committee/"
read_csv_base = read_csv_cloud
```

Assigning Weights

```
weights_360 <- 0.55
weights_group <- 0.45
weights_model <- 0.00
```

Manually Creating an Algorithm

360 Data

```
three_sixty_rankings <- c(8.19, 7.24, 9.44, 8.76, 7.11, 7.33, 7.9, 7.21, 8.12, 8.09, 9.29, 9.36)
```

Normalized 360

```
(normalized_three_sixty_rankings <- three_sixty_rankings / 10)
```

```
##  [1] 0.819 0.724 0.944 0.876 0.711 0.733 0.790 0.721 0.812 0.809 0.929 0.936
```

Normalized and Weighted 360 Rankings

```
(normalized_and_weighted_360_rankings <- normalized_three_sixty_rankings * weights_360)
```

```
##  [1] 0.45045 0.39820 0.51920 0.48180 0.39105 0.40315 0.43450 0.39655 0.44660
## [10] 0.44495 0.51095 0.51480
```

```
final_360_rankings <- normalized_and_weighted_360_rankings
```

Base Model

```r
# Manually Create Candidates
candidates_list <- c("Laura Andrews", "Teresa Baker", "Lewis Brennan", "Vivian Cheong", "Lucas Davies",

# Manually Assign Scores to Candidates
# Evan
e_woods_rankings <- c(12, 6, 8, 4, 1, 2, 5, 11, 7, 3, 9, 10)

# Deepa
deepa_rankings <- c(11,7,9,3,6,10,4,8,5,1,2,12)

# Jason
jason_rankings <- c(8,2,12,4,1,3,6,10,9,5,7,11)

# Eras
eras <- c(11,5,9,2,4,1,8,10,6,3,7,12)

# Celeste
celeste <- c(5,10,1,4,12,9,8,11,6,7,3,2)

# Calculating a total
total <- (e_woods_rankings + deepa_rankings + jason_rankings + eras + celeste) / 5

# Creating a dataframe
basic_candidate_rankings <- data.frame(candidates_list, e_woods_rankings, deepa_rankings,  jason_rankin

# Showing the dataframe by total
manual_candidate_rankings_by_total <- arrange(basic_candidate_rankings, total)

# variablity as a tie breaker
(manual_candidate_rankings_by_total)
```

```
##     candidates_list e_woods_rankings deepa_rankings jason_rankings eras celeste
## 1     Vivian Cheong                4              3              4    2       4
## 2         Ish Patel                3              1              5    3       7
## 3      Lucas Davies                1              6              1    4      12
## 4     Imani Kironde                2             10              3    1       9
## 5  Valerie Peterson                9              2              7    7       3
## 6      Teresa Baker                6              7              2    5      10
## 7   Samuel Melendez                5              4              6    8       8
## 8      Russell Myer                7              5              9    6       6
## 9     Lewis Brennan                8              9             12    9       1
## 10    Laura Andrews               12             11              8   11       5
## 11       David Rice               10             12             11   12       2
## 12       Amy Nguyen               11              8             10   10      11
##    total
## 1    3.4
## 2    3.8
## 3    4.8
## 4    5.0
## 5    5.6
## 6    6.0
## 7    6.2
## 8    6.6
```

```
## 9    7.8
## 10   9.4
## 11   9.4
## 12  10.0
```

Creating an Algorithmic Pipeline

Reading in Data

```
csv_filename <- "Weighted Buckets - Weighted_Totals_And_Rankings.csv"
read_csv_location <- str_c(read_csv_base, csv_filename)
candidate_df <- read_csv(read_csv_location)
```

```
## Rows: 12 Columns: 7-- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Candidate
## dbl (6): Weighted Totals, Jason, Deepa, Eras, Celeste, Evan
## 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.
```

Including three_sixty_rankings to the weighted buckets

```
(candidate_df <- mutate(candidate_df, normalized_and_weighted_360_rankings))
```

```
## # A tibble: 12 x 8
##    Candidate        `Weighted Totals` Jason Deepa  Eras Celeste  Evan
##    <chr>                        <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
##  1 Laura Andrews                0.429     8    11    11      12    12
##  2 Teresa Baker                 0.636     2     7     5       9     6
##  3 Lewis Brennan                0.575    12     9     9       7     8
##  4 Vivian Cheong                0.824     4     3     2       4     4
##  5 Lucas Davies                 0.720     1     6     4       8     1
##  6 Imani Kironde                0.670     3    10     1      10     2
##  7 Samuel Melendez              0.666     6     4     8       6     5
##  8 Amy Nguyen                   0.524    10     8    10       1    11
##  9 Russell Myer                 0.665     9     5     6       3     7
## 10 Ish Patel                    0.824     5     1     3       2     3
## 11 Valerie Peterson             0.658     7     2     7      11     9
## 12 David Rice                   0.521    11    12    12       5    10
## # i 1 more variable: normalized_and_weighted_360_rankings <dbl>
```

Creating Adjusted Values to Establish Values between 0 - 11 for Normalization

```
candidate_df <- mutate(candidate_df, Jason = candidate_df$Jason - 1)
candidate_df <- mutate(candidate_df, Deepa = candidate_df$Deepa - 1)
candidate_df <- mutate(candidate_df, Evan = candidate_df$Evan - 1)
candidate_df <- mutate(candidate_df, Celeste = candidate_df$Celeste - 1)
candidate_df <- (mutate(candidate_df, Eras = candidate_df$Eras - 1))
```

Creating a Group Sum

```
group_sum <- candidate_df$Jason + candidate_df$Deepa + candidate_df$Evan + candidate_df$Eras + candidat
```

Defining a difference from the total possible sum from the calculated sum: This will cause top candidates to have larger values

```
top_candidate_highest_value_sum <- 55 - group_sum
```

Creating an average of the sum

```
average_of_top_candidate_highest_value_sum <- top_candidate_highest_value_sum / 5
```

Normalizing the average: Values range from 0 to 11; Normalizing the values to range from 0 to 1

```
average_of_top_candidate_highest_value_sum <- average_of_top_candidate_highest_value_sum / 11
```

Scaling the Normalized average

```
weighted_normalized_group_average <- average_of_top_candidate_highest_value_sum * weights_group
```

Creating Weighted Totals

```
weighted_totals <- weighted_normalized_group_average + normalized_and_weighted_360_rankings
```

Creating a Data Frame Containing Candidates Ranked By Weighted Totals

```
candidates_weighted_totals <- data.frame(candidates_list, weighted_totals, row.names = NULL, check.rows
```

Creating a Sorted Data Frame Ranked By Weighted Totals

```
ranked_candidates_weighted_totals <- arrange(candidates_weighted_totals, desc(weighted_totals))
ranked_candidates_weighted_totals
```

```
##     candidates_list weighted_totals
## 1     Vivian Cheong       0.8336182
## 2         Ish Patel       0.8213136
## 3      Lucas Davies       0.7183227
## 4  Valerie Peterson       0.7073136
## 5      Russell Myer       0.6920545
## 6   Samuel Melendez       0.6881364
## 7     Imani Kironde       0.6813318
## 8      Teresa Baker       0.6518364
## 9     Lewis Brennan       0.6419273
## 10       David Rice       0.5966182
## 11       Amy Nguyen       0.5601864
## 12    Laura Andrews       0.4995409
```

Creating Weighted Rankings

```
# Adding the three_sixty_rankings to the candidate_df
mutate(candidate_df, three_sixty_rankings)
```

```
## # A tibble: 12 x 9
##    Candidate        `Weighted Totals` Jason Deepa  Eras Celeste  Evan
```

```
##    <chr>                        <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
##  1 Laura Andrews                0.429     7    10    10      11    11
##  2 Teresa Baker                 0.636     1     6     4       8     5
##  3 Lewis Brennan                0.575    11     8     8       6     7
##  4 Vivian Cheong                0.824     3     2     1       3     3
##  5 Lucas Davies                 0.720     0     5     3       7     0
##  6 Imani Kironde                0.670     2     9     0       9     1
##  7 Samuel Melendez              0.666     5     3     7       5     4
##  8 Amy Nguyen                   0.524     9     7     9       0    10
##  9 Russell Myer                 0.665     8     4     5       2     6
## 10 Ish Patel                    0.824     4     0     2       1     2
## 11 Valerie Peterson             0.658     6     1     6      10     8
## 12 David Rice                   0.521    10    11    11       4     9
## # i 2 more variables: normalized_and_weighted_360_rankings <dbl>,
## #   three_sixty_rankings <dbl>
```

```
arrange(candidate_df, desc(candidate_df$'Weighted Totals'))
```

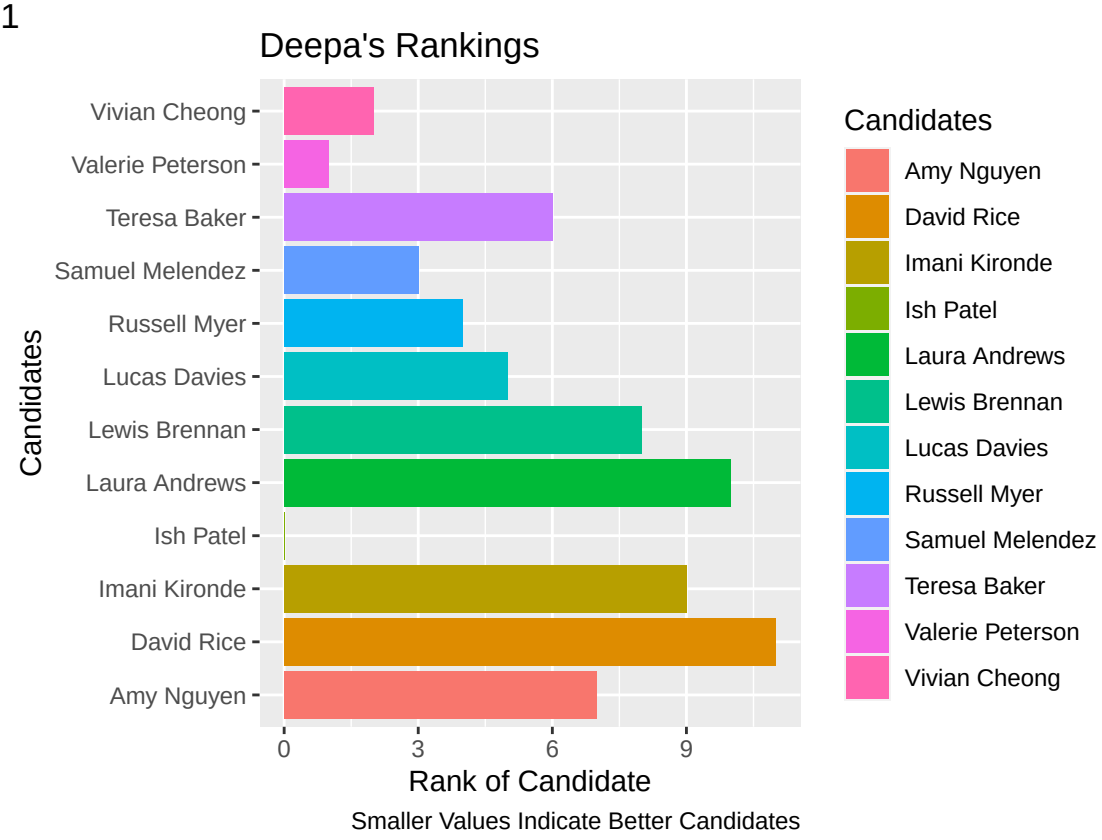
```
## # A tibble: 12 x 8
##    Candidate        `Weighted Totals` Jason Deepa  Eras Celeste  Evan
##    <chr>                        <dbl> <dbl> <dbl> <dbl>   <dbl> <dbl>
##  1 Vivian Cheong                0.824     3     2     1       3     3
##  2 Ish Patel                    0.824     4     0     2       1     2
##  3 Lucas Davies                 0.720     0     5     3       7     0
##  4 Imani Kironde                0.670     2     9     0       9     1
##  5 Samuel Melendez              0.666     5     3     7       5     4
##  6 Russell Myer                 0.665     8     4     5       2     6
##  7 Valerie Peterson             0.658     6     1     6      10     8
##  8 Teresa Baker                 0.636     1     6     4       8     5
##  9 Lewis Brennan                0.575    11     8     8       6     7
## 10 Amy Nguyen                   0.524     9     7     9       0    10
## 11 David Rice                   0.521    10    11    11       4     9
## 12 Laura Andrews                0.429     7    10    10      11    11
## # i 1 more variable: normalized_and_weighted_360_rankings <dbl>
```

Section: Plotting Individual Scores

Deepa's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Deepa, fill = candidates_list)) +
  labs(title = "Deepa's Rankings", y = "Rank of Candidate", x = "Candidates", fill = "Candidates", alpha
  coord_flip()
```

Figure 1



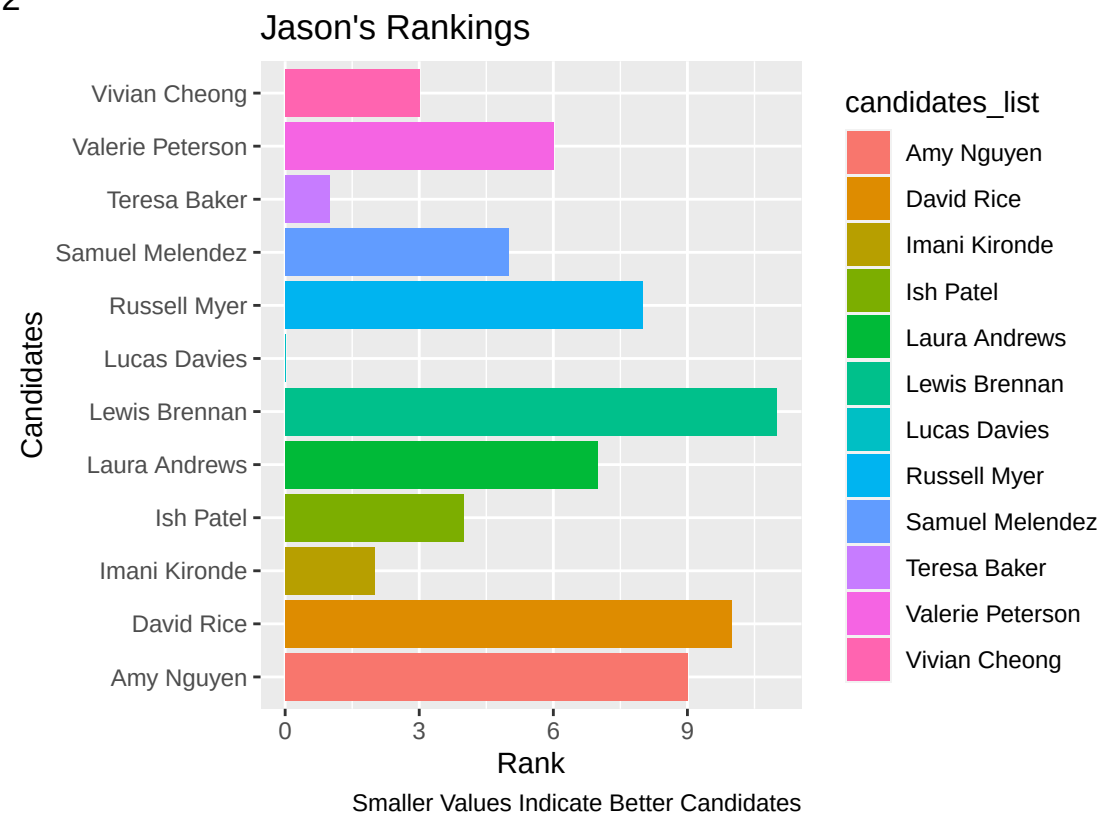

```
plot_save = str_c(plot_save_base, "Deepa_rankings.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Jason's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Jason, fill = candidates_list)) +
  labs(title = "Jason's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bet
  coord_flip()
```

Figure 2



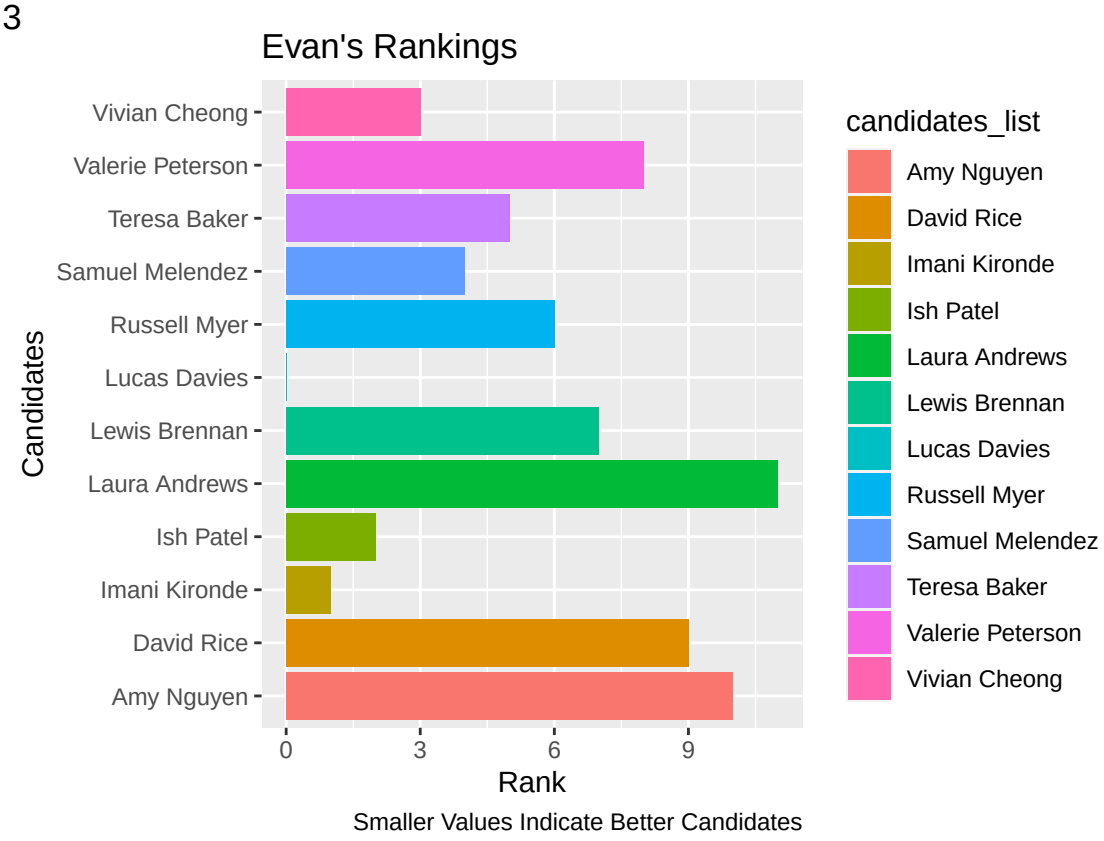

```
plot_save = str_c(plot_save_base, "Jason_rankings.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Evan's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Evan, fill = candidates_list)) +
  labs(title = "Evan's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bett
  coord_flip()
```

Figure 3



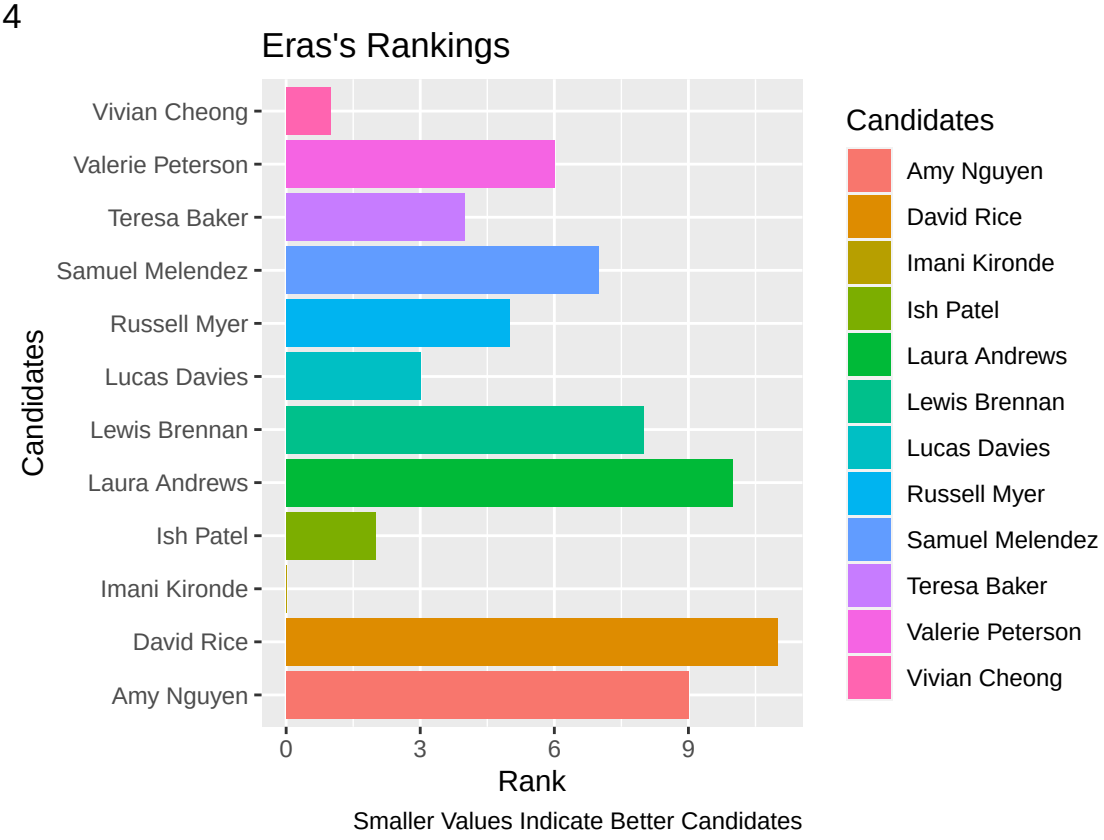

```
plot_save = str_c(plot_save_base, "Evan_rankings.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Eras' Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Eras, fill = candidates_list)) +
 labs(title = "Eras's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bett
  coord_flip()
```

Figure 4



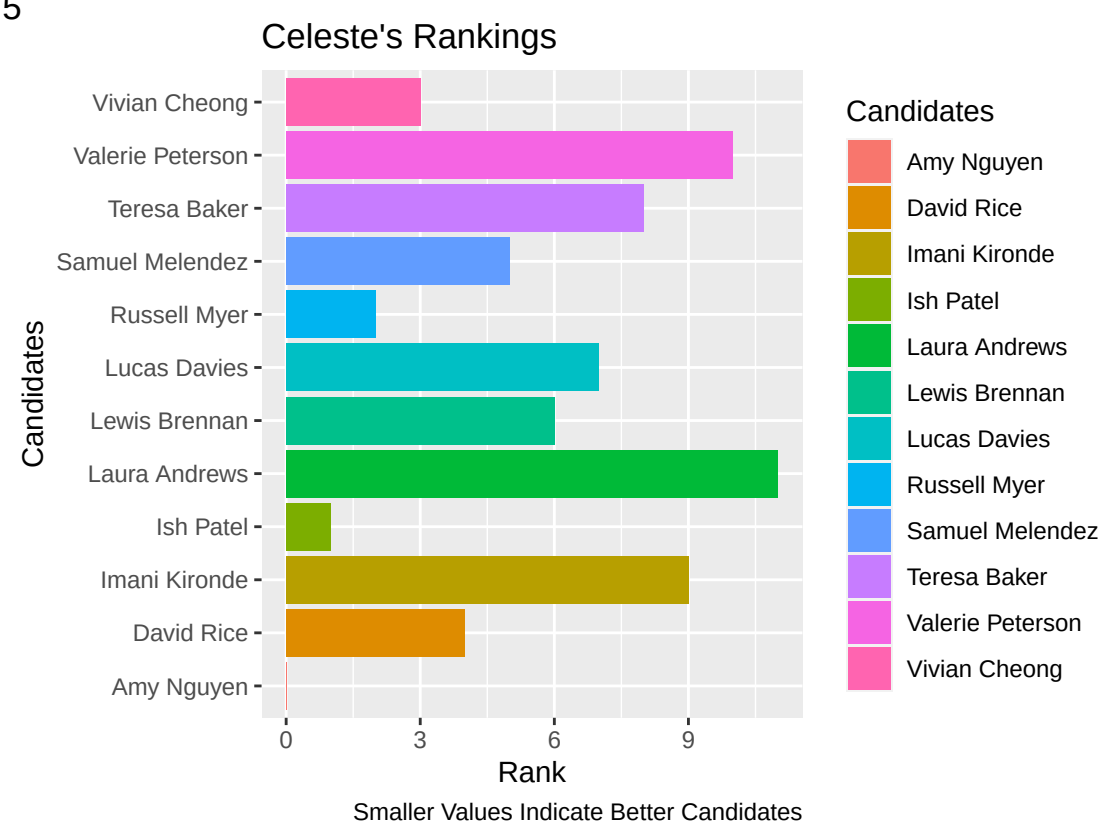

```
plot_save = str_c(plot_save_base, "Eras_ranking.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Celeste Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Celeste, fill = candidates_list)) +
  labs(title = "Celeste's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate B
  coord_flip()
```

Figure 5




```
plot_save = str_c(plot_save_base, "Celeste_ranking.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Section: Transforming Data

Candidates As Columns Original Rankings (Lower Rank Is Best Candidate for 360, Group, and Totals)

```
Metric <- row_names <- c("Evan", "Deepa", "Jason", "Celeste", "Eras", "360", "Group", "Totals")

for (x in seq_along(candidates_list)) {

  candidate <- c((candidate_df$Evan[x] + 1), (candidate_df$Deepa[x] + 1), (candidate_df$Jason[x] + 1),

  if (x == 1){
    candidates_as_columns <- data.frame(candidate)
  } else {
    candidates_as_columns <- data.frame(candidates_as_columns, candidate)
  }

  # print(candidates_list[x])
  # print(e_woods_rankings[x])
  # print(deepa_rankings[x])
  # print(jason_rankings[x])
  # print(normalized_and_weighted_360_rankings[x])
```

```
}

colnames(candidates_as_columns) <- candidates_list
rownames(candidates_as_columns) <- Metric

# candidates_as_columns <- data.frame(candidates_as_columns, Metric)

(candidates_as_columns)
```

```
##         Laura Andrews Teresa Baker Lewis Brennan Vivian Cheong Lucas Davies
## Evan       12.00000000    6.0000000     8.0000000     4.0000000    1.0000000
## Deepa      11.00000000    7.0000000     9.0000000     3.0000000    6.0000000
## Jason       8.00000000    2.0000000    12.0000000     4.0000000    1.0000000
## Celeste    12.00000000    9.0000000     7.0000000     4.0000000    8.0000000
## Eras       11.00000000    5.0000000     9.0000000     2.0000000    4.0000000
## 360         0.45045000    0.3982000     0.5192000     0.4818000    0.3910500
## Group       0.04909091    0.2536364     0.1227273     0.3518182    0.3272727
## Totals      0.49954091    0.6518364     0.6419273     0.8336182    0.7183227
##         Imani Kironde Samuel Melendez Amy Nguyen Russell Myer Ish Patel
## Evan        2.0000000       5.0000000 11.0000000    7.0000000 3.0000000
## Deepa      10.0000000       4.0000000  8.0000000    5.0000000 1.0000000
## Jason       3.0000000       6.0000000 10.0000000    9.0000000 5.0000000
## Celeste    10.0000000       6.0000000  1.0000000    3.0000000 2.0000000
## Eras        1.0000000       8.0000000 10.0000000    6.0000000 3.0000000
## 360         0.4031500       0.4345000  0.3965500    0.4466000 0.4449500
## Group       0.2781818       0.2536364  0.1636364    0.2454545 0.3763636
## Totals      0.6813318       0.6881364  0.5601864    0.6920545 0.8213136
##         Valerie Peterson  David Rice
## Evan           9.0000000 10.00000000
## Deepa          2.0000000 12.00000000
## Jason          7.0000000 11.00000000
## Celeste       11.0000000  5.00000000
## Eras           7.0000000 12.00000000
## 360            0.5109500  0.51480000
## Group          0.1963636  0.08181818
## Totals         0.7073136  0.59661818
```

Candidates As Columns Scaled Rankings (Higher Values Are Best Candidate; Aligned with 360 and Group and Totals)

```
Metric <- c("Evan", "Deepa", "Jason", "Celeste", "Eras", "360", "Group", "Totals")

for (x in seq_along(candidates_list)) {

  candidate <- c((1-(candidate_df$Evan[x]/11)), (1-(candidate_df$Deepa[x]/11)), (1-(candidate_df$Jason[

  if (x == 1){
    candidates_as_columns_scaled <- data.frame(candidate)
  } else {
    candidates_as_columns_scaled <- data.frame(candidates_as_columns_scaled, candidate)
  }

  # print(candidates_list[x])
```

```
  # print(e_woods_rankings[x])
  # print(deepa_rankings[x])
  # print(jason_rankings[x])
  # print(normalized_and_weighted_360_rankings[x])

}

colnames(candidates_as_columns_scaled) <- candidates_list
rownames(candidates_as_columns_scaled) <- Metric

# candidates_as_columns_scaled <- data.frame(candidates_as_columns_scaled, Metric)

(candidates_as_columns_scaled)
```

```
##         Laura Andrews Teresa Baker Lewis Brennan Vivian Cheong Lucas Davies
## Evan       0.00000000    0.5454545     0.3636364     0.7272727    1.0000000
## Deepa      0.09090909    0.4545455     0.2727273     0.8181818    0.5454545
## Jason      0.36363636    0.9090909     0.0000000     0.7272727    1.0000000
## Celeste    0.00000000    0.2727273     0.4545455     0.7272727    0.3636364
## Eras       0.09090909    0.6363636     0.2727273     0.9090909    0.7272727
## 360        0.45045000    0.3982000     0.5192000     0.4818000    0.3910500
## Group      0.04909091    0.2536364     0.1227273     0.3518182    0.3272727
## Totals     0.49954091    0.6518364     0.6419273     0.8336182    0.7183227
##         Imani Kironde Samuel Melendez Amy Nguyen Russell Myer Ish Patel
## Evan        0.9090909       0.6363636 0.09090909    0.4545455 0.8181818
## Deepa       0.1818182       0.7272727 0.36363636    0.6363636 1.0000000
## Jason       0.8181818       0.5454545 0.18181818    0.2727273 0.6363636
## Celeste     0.1818182       0.5454545 1.00000000    0.8181818 0.9090909
## Eras        1.0000000       0.3636364 0.18181818    0.5454545 0.8181818
## 360         0.4031500       0.4345000 0.39655000    0.4466000 0.4449500
## Group       0.2781818       0.2536364 0.16363636    0.2454545 0.3763636
## Totals      0.6813318       0.6881364 0.56018636    0.6920545 0.8213136
##         Valerie Peterson David Rice
## Evan          0.27272727 0.18181818
## Deepa         0.90909091 0.00000000
## Jason         0.45454545 0.09090909
## Celeste       0.09090909 0.63636364
## Eras          0.45454545 0.00000000
## 360           0.51095000 0.51480000
## Group         0.19636364 0.08181818
## Totals        0.70731364 0.59661818
```

Section: Viewing Candidates Data

Section: Candidates as Columns View

```
candidates_as_columns_scaled
```

```
##         Laura Andrews Teresa Baker Lewis Brennan Vivian Cheong Lucas Davies
## Evan       0.00000000    0.5454545     0.3636364     0.7272727    1.0000000
## Deepa      0.09090909    0.4545455     0.2727273     0.8181818    0.5454545
## Jason      0.36363636    0.9090909     0.0000000     0.7272727    1.0000000
## Celeste    0.00000000    0.2727273     0.4545455     0.7272727    0.3636364
```

```
## Eras       0.09090909    0.6363636     0.2727273     0.9090909    0.7272727
## 360        0.45045000    0.3982000     0.5192000     0.4818000    0.3910500
## Group      0.04909091    0.2536364     0.1227273     0.3518182    0.3272727
## Totals     0.49954091    0.6518364     0.6419273     0.8336182    0.7183227
##         Imani Kironde Samuel Melendez Amy Nguyen Russell Myer Ish Patel
## Evan        0.9090909       0.6363636 0.09090909    0.4545455 0.8181818
## Deepa       0.1818182       0.7272727 0.36363636    0.6363636 1.0000000
## Jason       0.8181818       0.5454545 0.18181818    0.2727273 0.6363636
## Celeste     0.1818182       0.5454545 1.00000000    0.8181818 0.9090909
## Eras        1.0000000       0.3636364 0.18181818    0.5454545 0.8181818
## 360         0.4031500       0.4345000 0.39655000    0.4466000 0.4449500
## Group       0.2781818       0.2536364 0.16363636    0.2454545 0.3763636
## Totals      0.6813318       0.6881364 0.56018636    0.6920545 0.8213136
##         Valerie Peterson David Rice
## Evan          0.27272727 0.18181818
## Deepa         0.90909091 0.00000000
## Jason         0.45454545 0.09090909
## Celeste       0.09090909 0.63636364
## Eras          0.45454545 0.00000000
## 360           0.51095000 0.51480000
## Group         0.19636364 0.08181818
## Totals        0.70731364 0.59661818
```

```
candidates_list
```

```
##  [1] "Laura Andrews"    "Teresa Baker"     "Lewis Brennan"    "Vivian Cheong"
##  [5] "Lucas Davies"     "Imani Kironde"    "Samuel Melendez"  "Amy Nguyen"
##  [9] "Russell Myer"     "Ish Patel"        "Valerie Peterson" "David Rice"
```

```
candidates_to_view <- c("Ish Patel", "Vivian Cheong", "")
select(candidates_as_columns, any_of(candidates_to_view))
```

Section: View Specific Candidates Original Rankings

```
##         Ish Patel Vivian Cheong
## Evan    3.0000000     4.0000000
## Deepa   1.0000000     3.0000000
## Jason   5.0000000     4.0000000
## Celeste 2.0000000     4.0000000
## Eras    3.0000000     2.0000000
## 360     0.4449500     0.4818000
## Group   0.3763636     0.3518182
## Totals  0.8213136     0.8336182
```

```
candidates_to_view <- c("David Rice", "Valerie Peterson", "")
select(candidates_as_columns_scaled, any_of(candidates_to_view))
```

Section: View Specific Candidates With Scaled Rankings

```
##         David Rice Valerie Peterson
## Evan    0.18181818       0.27272727
## Deepa   0.00000000       0.90909091
## Jason   0.09090909       0.45454545
## Celeste 0.63636364       0.09090909
```

```
## Eras    0.00000000       0.45454545
## 360     0.51480000       0.51095000
## Group   0.08181818       0.19636364
## Totals  0.59661818       0.70731364
```

Section: Candidates As Rows and Ranking

```
(master_candidates_as_rows_df <- mutate(candidates_weighted_totals, normalized_and_weighted_360_ranking
```

Section: Candidates By Row, Weighted Totals, Normalized and Weighted 360 Rankings, Weighted Normalized Group Average, Individual Group Member Rankings from 0 - 11 (0 being the most significant)

```
##     candidates_list weighted_totals normalized_and_weighted_360_rankings
## 1     Laura Andrews       0.4995409                                0.45045
## 2      Teresa Baker       0.6518364                                0.39820
## 3     Lewis Brennan       0.6419273                                0.51920
## 4     Vivian Cheong       0.8336182                                0.48180
## 5      Lucas Davies       0.7183227                                0.39105
## 6     Imani Kironde       0.6813318                                0.40315
## 7   Samuel Melendez       0.6881364                                0.43450
## 8        Amy Nguyen       0.5601864                                0.39655
## 9      Russell Myer       0.6920545                                0.44660
## 10        Ish Patel       0.8213136                                0.44495
## 11 Valerie Peterson       0.7073136                                0.51095
## 12       David Rice       0.5966182                                0.51480
##    weighted_normalized_group_average candidate_df$Evan candidate_df$Deepa
## 1                         0.04909091                11                 10
## 2                         0.25363636                 5                  6
## 3                         0.12272727                 7                  8
## 4                         0.35181818                 3                  2
## 5                         0.32727273                 0                  5
## 6                         0.27818182                 1                  9
## 7                         0.25363636                 4                  3
## 8                         0.16363636                10                  7
## 9                         0.24545455                 6                  4
## 10                        0.37636364                 2                  0
## 11                        0.19636364                 8                  1
## 12                        0.08181818                 9                 11
##    candidate_df$Jason candidate_df$Celeste candidate_df$Eras
## 1                   7                  11                10
## 2                   1                   8                 4
## 3                  11                   6                 8
## 4                   3                   3                 1
## 5                   0                   7                 3
## 6                   2                   9                 0
## 7                   5                   5                 7
## 8                   9                   0                 9
## 9                   8                   2                 5
## 10                  4                   1                 2
## 11                  6                  10                 6
## 12                 10                   4                11
```

```
(master_candidates_as_rows_df <- mutate(candidates_weighted_totals, normalized_and_weighted_360_rankings
```

```
##     candidates_list weighted_totals normalized_and_weighted_360_rankings
## 1     Laura Andrews       0.4995409                               0.45045
## 2      Teresa Baker       0.6518364                               0.39820
## 3     Lewis Brennan       0.6419273                               0.51920
## 4     Vivian Cheong       0.8336182                               0.48180
## 5      Lucas Davies       0.7183227                               0.39105
## 6     Imani Kironde       0.6813318                               0.40315
## 7   Samuel Melendez       0.6881364                               0.43450
## 8        Amy Nguyen       0.5601864                               0.39655
## 9      Russell Myer       0.6920545                               0.44660
## 10        Ish Patel       0.8213136                               0.44495
## 11 Valerie Peterson       0.7073136                               0.51095
## 12       David Rice       0.5966182                               0.51480
##    weighted_normalized_group_average candidate_df$Evan candidate_df$Deepa
## 1                         0.04909091                11                 10
## 2                         0.25363636                 5                  6
## 3                         0.12272727                 7                  8
## 4                         0.35181818                 3                  2
## 5                         0.32727273                 0                  5
## 6                         0.27818182                 1                  9
## 7                         0.25363636                 4                  3
## 8                         0.16363636                10                  7
## 9                         0.24545455                 6                  4
## 10                        0.37636364                 2                  0
## 11                        0.19636364                 8                  1
## 12                        0.08181818                 9                 11
##    candidate_df$Jason candidate_df$Celeste candidate_df$Eras
## 1                   7                  11               10
## 2                   1                   8                4
## 3                  11                   6                8
## 4                   3                   3                1
## 5                   0                   7                3
## 6                   2                   9                0
## 7                   5                   5                7
## 8                   9                   0                9
## 9                   8                   2                5
## 10                  4                   1                2
## 11                  6                  10                6
## 12                 10                   4               11
```

```
(scaled_master_candidates_as_rows_df <- mutate(candidates_weighted_totals, normalized_and_weighted_360_
```

Section: Scaled Candidates As Rows

```
##     candidates_list weighted_totals normalized_and_weighted_360_rankings
## 1     Laura Andrews       0.4995409                               0.45045
## 2      Teresa Baker       0.6518364                               0.39820
## 3     Lewis Brennan       0.6419273                               0.51920
## 4     Vivian Cheong       0.8336182                               0.48180
## 5      Lucas Davies       0.7183227                               0.39105
## 6     Imani Kironde       0.6813318                               0.40315
```

```
## 7   Samuel Melendez       0.6881364                              0.43450
## 8        Amy Nguyen       0.5601864                              0.39655
## 9      Russell Myer       0.6920545                              0.44660
## 10        Ish Patel       0.8213136                              0.44495
## 11 Valerie Peterson       0.7073136                              0.51095
## 12       David Rice       0.5966182                              0.51480
##    weighted_normalized_group_average 1 - candidate_df$Evan/11
## 1                         0.04909091               0.00000000
## 2                         0.25363636               0.54545455
## 3                         0.12272727               0.36363636
## 4                         0.35181818               0.72727273
## 5                         0.32727273               1.00000000
## 6                         0.27818182               0.90909091
## 7                         0.25363636               0.63636364
## 8                         0.16363636               0.09090909
## 9                         0.24545455               0.45454545
## 10                        0.37636364               0.81818182
## 11                        0.19636364               0.27272727
## 12                        0.08181818               0.18181818
##    1 - candidate_df$Deepa/11 1 - candidate_df$Jason/11
## 1                 0.09090909               0.36363636
## 2                 0.45454545               0.90909091
## 3                 0.27272727               0.00000000
## 4                 0.81818182               0.72727273
## 5                 0.54545455               1.00000000
## 6                 0.18181818               0.81818182
## 7                 0.72727273               0.54545455
## 8                 0.36363636               0.18181818
## 9                 0.63636364               0.27272727
## 10                1.00000000               0.63636364
## 11                0.90909091               0.45454545
## 12                0.00000000               0.09090909
##    1 - candidate_df$Celeste/11 1 - candidate_df$Eras/11
## 1                   0.00000000              0.09090909
## 2                   0.27272727              0.63636364
## 3                   0.45454545              0.27272727
## 4                   0.72727273              0.90909091
## 5                   0.36363636              0.72727273
## 6                   0.18181818              1.00000000
## 7                   0.54545455              0.36363636
## 8                   1.00000000              0.18181818
## 9                   0.81818182              0.54545455
## 10                  0.90909091              0.81818182
## 11                  0.09090909              0.45454545
## 12                  0.63636364              0.00000000
```

```
col_names_scaled_master_candidates_as_rows_df <- c("Candidates", "Totals", "Rankings_360", "Group_Averag
```

```
colnames(scaled_master_candidates_as_rows_df) <- col_names_scaled_master_candidates_as_rows_df
```

```
scaled_master_candidates_as_rows_df
```

```
##          Candidates    Totals Rankings_360 Group_Average Evan_rankings
## 1     Laura Andrews 0.4995409      0.45045    0.04909091    0.00000000
## 2      Teresa Baker 0.6518364      0.39820    0.25363636    0.54545455
```

```
## 3     Lewis Brennan 0.6419273      0.51920    0.12272727    0.36363636
## 4     Vivian Cheong 0.8336182      0.48180    0.35181818    0.72727273
## 5      Lucas Davies 0.7183227      0.39105    0.32727273    1.00000000
## 6     Imani Kironde 0.6813318      0.40315    0.27818182    0.90909091
## 7   Samuel Melendez 0.6881364      0.43450    0.25363636    0.63636364
## 8        Amy Nguyen 0.5601864      0.39655    0.16363636    0.09090909
## 9      Russell Myer 0.6920545      0.44660    0.24545455    0.45454545
## 10        Ish Patel 0.8213136      0.44495    0.37636364    0.81818182
## 11 Valerie Peterson 0.7073136      0.51095    0.19636364    0.27272727
## 12       David Rice 0.5966182      0.51480    0.08181818    0.18181818
##    Deepa_rankings Jason_rankings Celeste_rankings Eras_rankings
## 1      0.09090909     0.36363636       0.00000000    0.09090909
## 2      0.45454545     0.90909091       0.27272727    0.63636364
## 3      0.27272727     0.00000000       0.45454545    0.27272727
## 4      0.81818182     0.72727273       0.72727273    0.90909091
## 5      0.54545455     1.00000000       0.36363636    0.72727273
## 6      0.18181818     0.81818182       0.18181818    1.00000000
## 7      0.72727273     0.54545455       0.54545455    0.36363636
## 8      0.36363636     0.18181818       1.00000000    0.18181818
## 9      0.63636364     0.27272727       0.81818182    0.54545455
## 10     1.00000000     0.63636364       0.90909091    0.81818182
## 11     0.90909091     0.45454545       0.09090909    0.45454545
## 12     0.00000000     0.09090909       0.63636364    0.00000000
```

```
ranked_candidates_weighted_totals <- arrange(scaled_master_candidates_as_rows_df, desc(Totals))
```

```
ranked_candidates_weighted_totals
```

Section: Ranked Scaled Candidates By Rows Master List.

```
##          Candidates    Totals Rankings_360 Group_Average Evan_rankings
## 1     Vivian Cheong 0.8336182      0.48180    0.35181818    0.72727273
## 2         Ish Patel 0.8213136      0.44495    0.37636364    0.81818182
## 3      Lucas Davies 0.7183227      0.39105    0.32727273    1.00000000
## 4  Valerie Peterson 0.7073136      0.51095    0.19636364    0.27272727
## 5      Russell Myer 0.6920545      0.44660    0.24545455    0.45454545
## 6   Samuel Melendez 0.6881364      0.43450    0.25363636    0.63636364
## 7     Imani Kironde 0.6813318      0.40315    0.27818182    0.90909091
## 8      Teresa Baker 0.6518364      0.39820    0.25363636    0.54545455
## 9     Lewis Brennan 0.6419273      0.51920    0.12272727    0.36363636
## 10       David Rice 0.5966182      0.51480    0.08181818    0.18181818
## 11       Amy Nguyen 0.5601864      0.39655    0.16363636    0.09090909
## 12    Laura Andrews 0.4995409      0.45045    0.04909091    0.00000000
##    Deepa_rankings Jason_rankings Celeste_rankings Eras_rankings
## 1      0.81818182     0.72727273       0.72727273    0.90909091
## 2      1.00000000     0.63636364       0.90909091    0.81818182
## 3      0.54545455     1.00000000       0.36363636    0.72727273
## 4      0.90909091     0.45454545       0.09090909    0.45454545
## 5      0.63636364     0.27272727       0.81818182    0.54545455
## 6      0.72727273     0.54545455       0.54545455    0.36363636
## 7      0.18181818     0.81818182       0.18181818    1.00000000
## 8      0.45454545     0.90909091       0.27272727    0.63636364
## 9      0.27272727     0.00000000       0.45454545    0.27272727
```

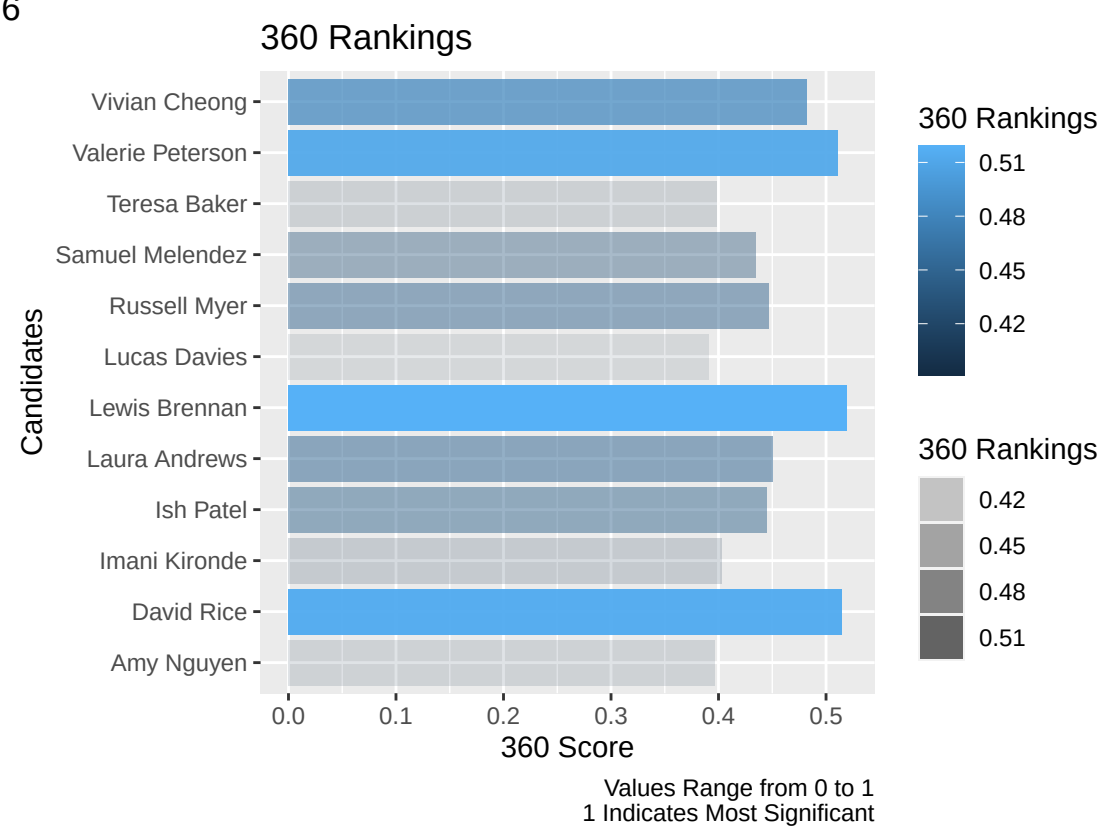
```
## 10      0.00000000      0.09090909      0.63636364      0.00000000
## 11      0.36363636      0.18181818      1.00000000      0.18181818
## 12      0.09090909      0.36363636      0.00000000      0.09090909
```

Plotting

360 Rankings

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = Candidates, y = Rankings_360, fill = Rankings_360, alpha = Rankings_360)) +
  ylab("360 Score") +
labs(title = "360 Rankings", fill = "360 Rankings", alpha = "360 Rankings", caption = "Values Range fro
  coord_flip()
```

Figure 6



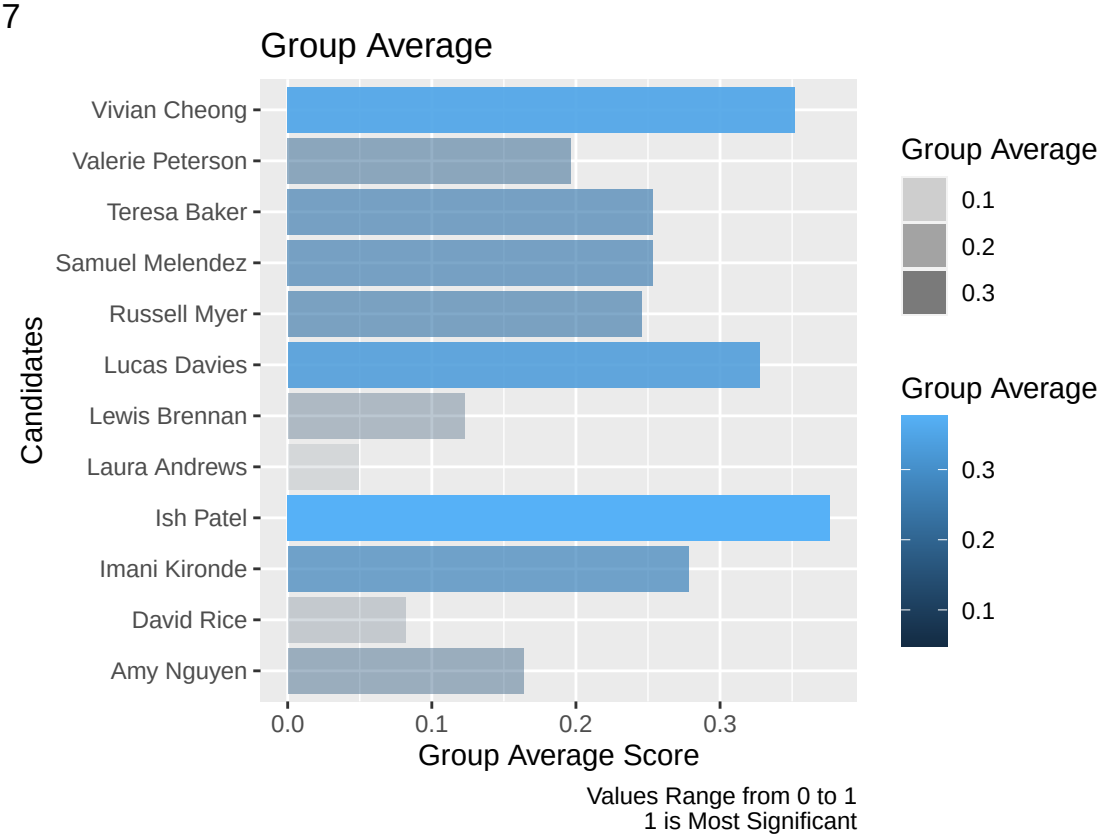

```
plot_save = str_c(plot_save_base, "360_Rankings.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Group Rankings

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = Candidates, y = Group_Average, fill = Group_Average, alpha = Group_Average)) +
  ylab("Group Average Score") +
    labs(title = "Group Average", fill = "Group Average", alpha = "Group Average", caption = "Values Ra
  coord_flip()
```

Figure 7




```
plot_save = str_c(plot_save_base, "Group_Average.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

```
ranked_candidates_weighted_totals
```

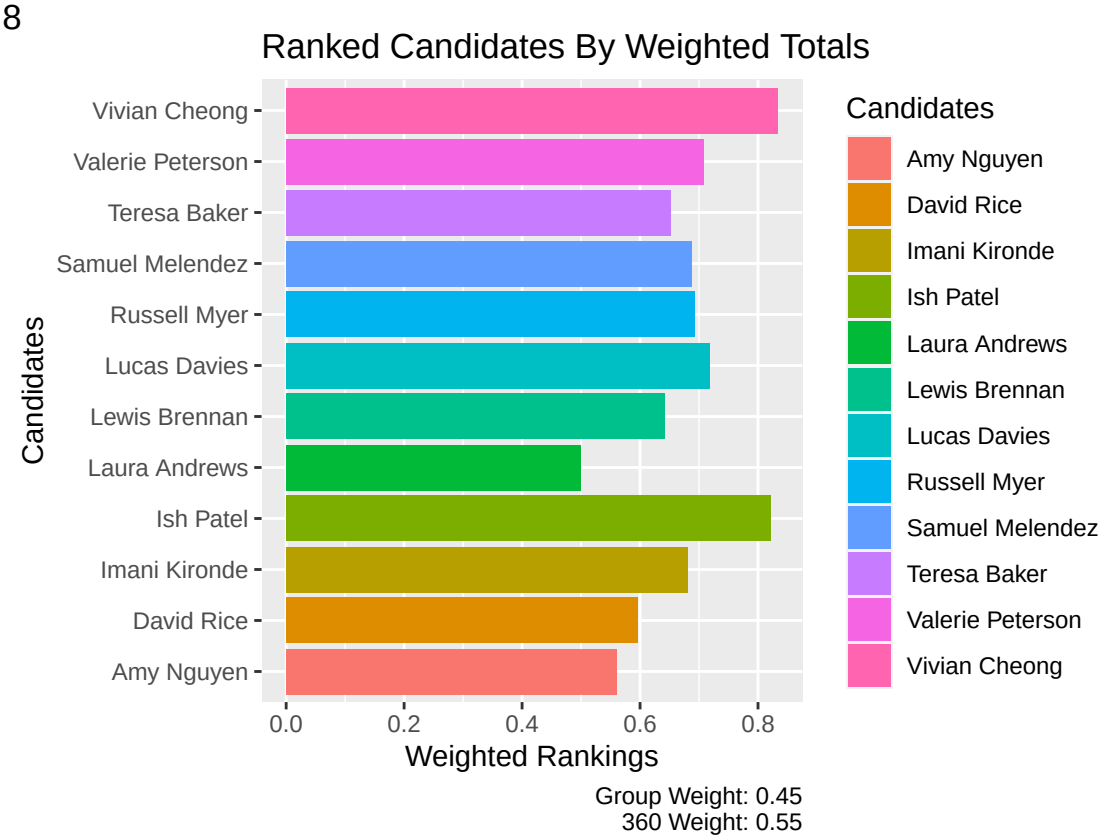
```
##          Candidates    Totals Rankings_360 Group_Average Evan_rankings
## 1     Vivian Cheong 0.8336182      0.48180    0.35181818    0.72727273
## 2         Ish Patel 0.8213136      0.44495    0.37636364    0.81818182
## 3      Lucas Davies 0.7183227      0.39105    0.32727273    1.00000000
## 4  Valerie Peterson 0.7073136      0.51095    0.19636364    0.27272727
## 5      Russell Myer 0.6920545      0.44660    0.24545455    0.45454545
## 6   Samuel Melendez 0.6881364      0.43450    0.25363636    0.63636364
## 7     Imani Kironde 0.6813318      0.40315    0.27818182    0.90909091
## 8      Teresa Baker 0.6518364      0.39820    0.25363636    0.54545455
## 9     Lewis Brennan 0.6419273      0.51920    0.12272727    0.36363636
## 10       David Rice 0.5966182      0.51480    0.08181818    0.18181818
## 11       Amy Nguyen 0.5601864      0.39655    0.16363636    0.09090909
## 12    Laura Andrews 0.4995409      0.45045    0.04909091    0.00000000
##    Deepa_rankings Jason_rankings Celeste_rankings Eras_rankings
## 1      0.81818182     0.72727273       0.72727273    0.90909091
## 2      1.00000000     0.63636364       0.90909091    0.81818182
## 3      0.54545455     1.00000000       0.36363636    0.72727273
## 4      0.90909091     0.45454545       0.09090909    0.45454545
## 5      0.63636364     0.27272727       0.81818182    0.54545455
## 6      0.72727273     0.54545455       0.54545455    0.36363636
## 7      0.18181818     0.81818182       0.18181818    1.00000000
```

```
## 8       0.45454545      0.90909091      0.27272727      0.63636364
## 9       0.27272727      0.00000000      0.45454545      0.27272727
## 10      0.00000000      0.09090909      0.63636364      0.00000000
## 11      0.36363636      0.18181818      1.00000000      0.18181818
## 12      0.09090909      0.36363636      0.00000000      0.09090909
```

Ranked Candidates By Weighted Totals

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = candidates_list, y = weighted_totals, fill = candidates_list)) +
    labs(y = "Weighted Rankings", x = "Candidates", fill = "Candidates", title = "Ranked Candidates By W
  coord_flip()
```

Figure 8




```
plot_save = str_c(plot_save_base, "ranked_candidates.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

```
weights_360 <- 0.55
weights_group <- 0.45
weights_model <- 0.00
```

Assigning Weights

Linear Model Creation

```
set.seed(42)
```

1) Key Point: Data Dictionary

```
# Data Dictionary

# employee_id: Unique ID for employee
# department: Department of employee
# region: Region of employment (unordered)
# education: Education Level
# gender: Gender of Employee
# recruitment_channel: Channel of recruitment for employee
# no_of_trainings: no of other trainings completed in previous year on soft skills, technical skills et
# age: Age of Employee
# previous_year_rating: Employee Rating for the previous year
# length_of_service: Length of service in years
# awards_won?: if awards won during previous year then 1 else 0
# avg_training_score: Average score in current training evaluations
# is_promoted: (Target) Recommended for promotion
```

Import Data

```
# https://www.kaggle.com/datasets/arashnic/hr-ana?select=train.csv

  csv_filename <- "employee_promotion.csv"
  read_csv_location <- str_c(read_csv_base, csv_filename)

  employee_promotion_dataset <- read_csv(read_csv_location)
```

```
## Rows: 54808 Columns: 13-- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): department, region, education, gender, recruitment_channel
## dbl (8): employee_id, no_of_trainings, age, previous_year_rating, length_of_...
## 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.
```

Clean Data

```
clean_employee_promotion_dataset <- drop_na(employee_promotion_dataset)
```

```
  (clean_employee_promotion_dataset)
```

```
## # A tibble: 46,380 x 13
##    employee_id department        region    education  gender recruitment_channel
##          <dbl> <chr>             <chr>     <chr>      <chr>  <chr>
##  1       65438 Sales & Marketing region_7  Master's ~ f      sourcing
##  2       65141 Operations        region_22 Bachelor's m      other
##  3        7513 Sales & Marketing region_19 Bachelor's m      sourcing
##  4        2542 Sales & Marketing region_23 Bachelor's m      other
##  5       48945 Technology        region_26 Bachelor's m      other
##  6       58896 Analytics         region_2  Bachelor's m      sourcing
```

```
##  7       20379 Operations        region_20 Bachelor's f      other
##  8       16290 Operations        region_34 Master's ~ m      sourcing
##  9       73202 Analytics         region_20 Bachelor's m      other
## 10       28911 Sales & Marketing region_1  Master's ~ m      sourcing
## # i 46,370 more rows
## # i 7 more variables: no_of_trainings <dbl>, age <dbl>,
## #   previous_year_rating <dbl>, length_of_service <dbl>, awards_won <dbl>,
## #   avg_training_score <dbl>, is_promoted <dbl>
```

Exploratory Data Analysis

```
summary(clean_employee_promotion_dataset)
```

```
##   employee_id     department          region           education
##  Min.   :    1   Length:46380       Length:46380       Length:46380
##  1st Qu.:19583   Class :character   Class :character   Class :character
##  Median :39170   Mode  :character   Mode  :character   Mode  :character
##  Mean   :39192
##  3rd Qu.:58858
##  Max.   :78298
##     gender          recruitment_channel no_of_trainings       age
##  Length:46380       Length:46380        Min.   : 1.000   Min.   :20.00
##  Class :character   Class :character    1st Qu.: 1.000   1st Qu.:30.00
##  Mode  :character   Mode  :character    Median : 1.000   Median :34.00
##                                         Mean   : 1.255   Mean   :35.57
##                                         3rd Qu.: 1.000   3rd Qu.:39.00
##                                         Max.   :10.000   Max.   :60.00
##  previous_year_rating length_of_service   awards_won     avg_training_score
##  Min.   :1.000        Min.   : 1.000    Min.   :0.0000   Min.   :39.00
##  1st Qu.:3.000        1st Qu.: 3.000    1st Qu.:0.0000   1st Qu.:51.00
##  Median :3.000        Median : 5.000    Median :0.0000   Median :60.00
##  Mean   :3.332        Mean   : 6.308    Mean   :0.0235   Mean   :63.93
##  3rd Qu.:4.000        3rd Qu.: 8.000    3rd Qu.:0.0000   3rd Qu.:77.00
##  Max.   :5.000        Max.   :37.000    Max.   :1.0000   Max.   :99.00
##   is_promoted
##  Min.   :0.00000
##  1st Qu.:0.00000
##  Median :0.00000
##  Mean   :0.08777
##  3rd Qu.:0.00000
##  Max.   :1.00000
```

Normalizing previous_year_rating

```
clean_employee_promotion_dataset$previous_year_rating <- clean_employee_promotion_dataset$previous_year
```

```
summary(clean_employee_promotion_dataset)
```

```
##   employee_id     department          region           education
##  Min.   :    1   Length:46380       Length:46380       Length:46380
##  1st Qu.:19583   Class :character   Class :character   Class :character
##  Median :39170   Mode  :character   Mode  :character   Mode  :character
##  Mean   :39192
##  3rd Qu.:58858
```

```
##  Max.   :78298
##     gender          recruitment_channel no_of_trainings       age
##  Length:46380       Length:46380        Min.   : 1.000   Min.   :20.00
##  Class :character   Class :character    1st Qu.: 1.000   1st Qu.:30.00
##  Mode  :character   Mode  :character    Median : 1.000   Median :34.00
##                                         Mean   : 1.255   Mean   :35.57
##                                         3rd Qu.: 1.000   3rd Qu.:39.00
##                                         Max.   :10.000   Max.   :60.00
##  previous_year_rating length_of_service   awards_won     avg_training_score
##  Min.   :0.2000       Min.   : 1.000    Min.   :0.0000   Min.   :39.00
##  1st Qu.:0.6000       1st Qu.: 3.000    1st Qu.:0.0000   1st Qu.:51.00
##  Median :0.6000       Median : 5.000    Median :0.0000   Median :60.00
##  Mean   :0.6665       Mean   : 6.308    Mean   :0.0235   Mean   :63.93
##  3rd Qu.:0.8000       3rd Qu.: 8.000    3rd Qu.:0.0000   3rd Qu.:77.00
##  Max.   :1.0000       Max.   :37.000    Max.   :1.0000   Max.   :99.00
##   is_promoted
##  Min.   :0.00000
##  1st Qu.:0.00000
##  Median :0.00000
##  Mean   :0.08777
##  3rd Qu.:0.00000
##  Max.   :1.00000
```

Separate Data Into Train and Test Sets

```
clean_employee_promotion_dataset$train <- sample.split(clean_employee_promotion_dataset$is_promoted, Spl
```

```
table(clean_employee_promotion_dataset$train, clean_employee_promotion_dataset$is_promoted)
```

```
##
##             0     1
##   FALSE 21155  2035
##   TRUE  21154  2036
```

```
train_set <- subset(clean_employee_promotion_dataset, clean_employee_promotion_dataset$train == TRUE)
```

```
test_set <- subset(clean_employee_promotion_dataset, clean_employee_promotion_dataset$train == FALSE)
```

Model 1: Education, Scoring From Peers, Length of Service, Awards Won

```
model1 <- glm(is_promoted ~ as.factor(education) + previous_year_rating + length_of_service + awards_wo
```

```
summ(model1, type = "text", digits = 2)
```

Model 1: Summary

```
## MODEL INFO:
## Observations: 23190
## Dependent Variable: is_promoted
## Type: Generalized linear model
##   Family: binomial
##   Link function: logit
##
## MODEL FIT:
```

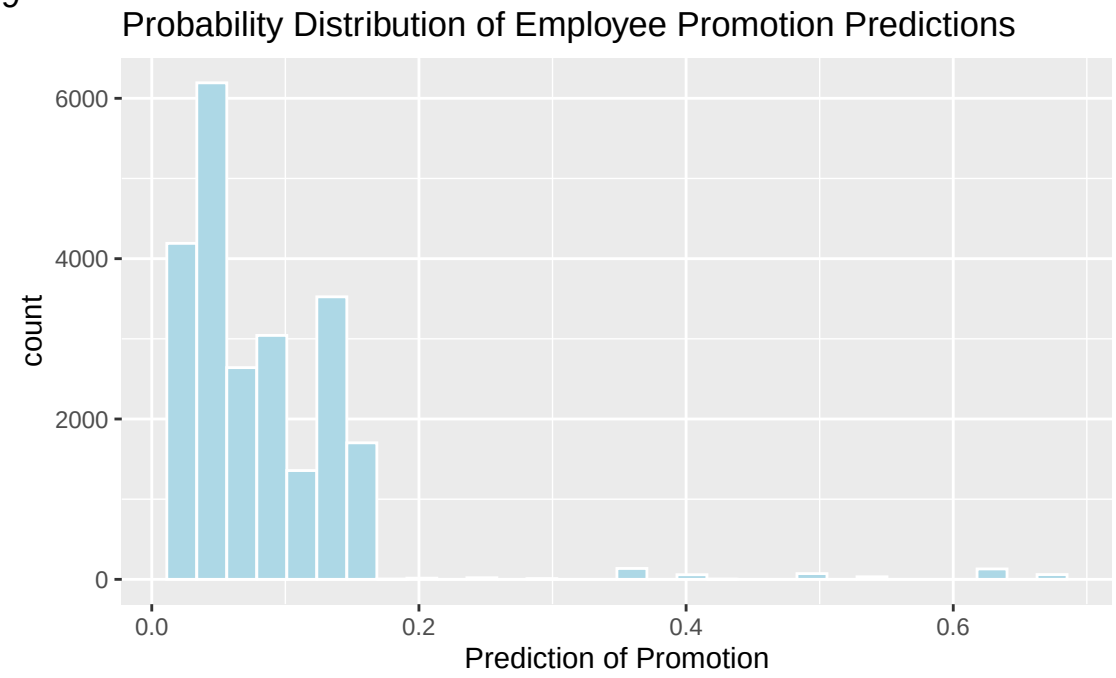
```
##  ²(5) = 1245.38, p = 0.00
## Pseudo-R² (Cragg-Uhler) = 0.12
## Pseudo-R² (McFadden) = 0.09
## AIC = 12560.49, BIC = 12608.80
##
## Standard errors: MLE
## ------------------------------------------------------------------
##                                    Est.   S.E.   z val.      p
## -------------------------------- ------- ------ -------- ------
## (Intercept)                       -4.45   0.10   -44.95   0.00
## as.factor(education)Below         -0.00   0.26    -0.01   1.00
## Secondary
## as.factor(education)Master's       0.21   0.05     4.03   0.00
## & above
## previous_year_rating              2.62   0.11    23.68   0.00
## length_of_service                 0.00   0.01     0.13   0.89
## awards_won                        2.32   0.09    25.35   0.00
## ------------------------------------------------------------------
```

```
test_set$is_promoted_prediction <- predict(model1, data=test_set, type="response")
```

```
ggplot(test_set, aes(is_promoted_prediction)) +
  geom_histogram(color = "white", fill = "lightblue", bins = 30) +
  labs(title = "Probability Distribution of Employee Promotion Predictions", x = "Prediction of Promoti
```

2) Key Point: Distribution of Test Set Predictions

Figure 9



Values Range From 0 to 1

Accuracy: 89.4
Precision: 9.58
Recall: 2.46

```
plot_save = str_c(plot_save_base, "Distribution_of_Employee_Promotion_Predictions.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

```
#install.packages("caret")
#install.packages("e1071")
#library(caret)
#library(e1071)

library(ModelMetrics)
```

```
##
## Attaching package: 'ModelMetrics'
```

```
## The following object is masked from 'package:base':
##
##     kappa
```

```
  CrossTable(as.numeric(test_set$is_promoted_prediction>0.2), as.numeric(test_set$is_promoted))
```

3) Key Point: Confusion Matrix

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  23190
##
##
##                                                | as.numeric(test_set$is_promoted)
## as.numeric(test_set$is_promoted_prediction > 0.2) |         0 |         1 | Row Total |
## -----------------------------------------------|-----------|-----------|-----------|
##                                              0 |     20683 |      1985 |     22668 |
##                                                |     0.001 |     0.009 |           |
##                                                |     0.912 |     0.088 |     0.977 |
##                                                |     0.978 |     0.975 |           |
##                                                |     0.892 |     0.086 |           |
## -----------------------------------------------|-----------|-----------|-----------|
##                                              1 |       472 |        50 |       522 |
##                                                |     0.037 |     0.384 |           |
##                                                |     0.904 |     0.096 |     0.023 |
##                                                |     0.022 |     0.025 |           |
##                                                |     0.020 |     0.002 |           |
## -----------------------------------------------|-----------|-----------|-----------|
##                                   Column Total |     21155 |      2035 |     23190 |
##                                                |     0.912 |     0.088 |           |
```

```
## -----------------------------------------------|----------|----------|----------|
##
##
```

```
tp <- 50
tn <- 20683
fp <- 472
fn <- 1985

accuracy <- (tp + tn) / (tp + tn + fp + fn)
precision <- tp / (tp + fp)
recall <- tp / (tp + fn)

(key_metrics <- data.frame(accuracy, precision, recall))
```

4) Key Metrics

```
##    accuracy  precision     recall
## 1 0.8940492 0.09578544 0.02457002
```

```
(key_metrics_percentiles <- key_metrics * 100)
```

```
##   accuracy precision   recall
## 1 89.40492  9.578544 2.457002
```

```
csv_filename <- "Weighted Buckets - Employee_data.csv"
read_csv_location <- str_c(read_csv_base, csv_filename)

employee_lm_model_data <- read_csv(read_csv_location)
```

Employee Predictions

```
## Rows: 12 Columns: 5-- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): education
## dbl (4): employee_id, previous_year_rating, length_of_service, awards_won
## 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.
```

```
employee_model_prediction <- predict(model1, employee_lm_model_data, type="response")
```

```
# With a cutoff of 0.01, Lucas Davies would be promoted.
lm_predicted_df <- data.frame(candidates_list, employee_model_prediction)
lm_predicted_df
```

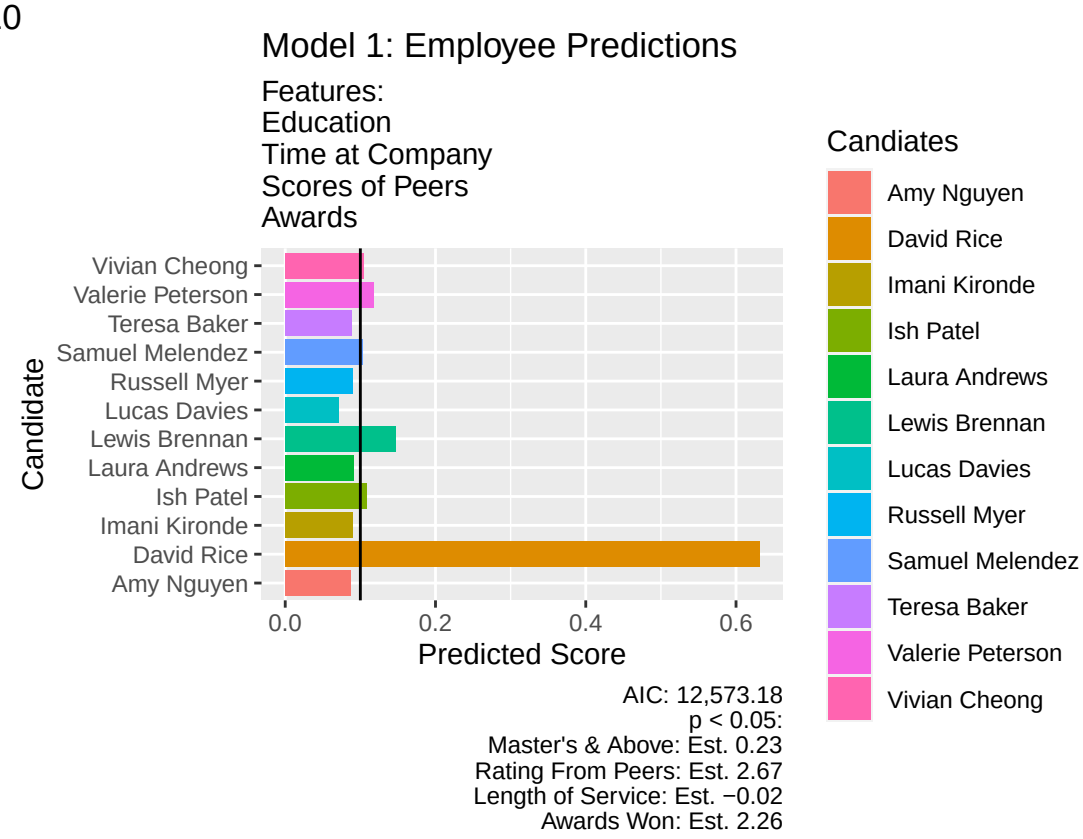
```
##     candidates_list employee_model_prediction
## 1     Laura Andrews                0.09151089
## 2      Teresa Baker                0.08813821
## 3     Lewis Brennan                0.14703753
## 4     Vivian Cheong                0.10486967
## 5      Lucas Davies                0.07063492
## 6     Imani Kironde                0.09024896
## 7   Samuel Melendez                0.10300383
## 8        Amy Nguyen                0.08750792
## 9      Russell Myer                0.08973649
```

```
## 10        Ish Patel               0.10792861
## 11 Valerie Peterson               0.11799022
## 12       David Rice               0.63106411
```

```
ggplot(lm_predicted_df, aes(candidates_list, employee_model_prediction)) +
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
    labs(y = "Predicted Score", x = "Candidate", title = "Model 1: Employee Predictions", subtitle = "F
  coord_flip()
```

Plot of Employee Predictions Model 1

Figure 10




```
plot_save = str_c(plot_save_base, "Employee_promotion_prediction_model_1.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

z### Model 2: Education, Scoring From Peers, Length of Service

```
model2 <- glm(is_promoted ~ as.factor(education) + previous_year_rating + length_of_service, data = tra
```

```
summ(model2, type = "text")
```

Model 2 Summary

```
## MODEL INFO:
## Observations: 23190
```

```
## Dependent Variable: is_promoted
## Type: Generalized linear model
##   Family: binomial
##   Link function: logit
##
## MODEL FIT:
##  ²(4) = 690.77, p = 0.00
## Pseudo-R² (Cragg-Uhler) = 0.07
## Pseudo-R² (McFadden) = 0.05
## AIC = 13113.10, BIC = 13153.35
##
## Standard errors: MLE
## ------------------------------------------------------------------
##                                    Est.   S.E.   z val.      p
## ------------------------------- ------- ------ -------- ------
## (Intercept)                       -4.27   0.10   -44.31   0.00
## as.factor(education)Below         -0.08   0.25    -0.33   0.74
## Secondary
## as.factor(education)Master's       0.21   0.05     4.19   0.00
## & above
## previous_year_rating               2.62   0.11    24.14   0.00
## length_of_service                 -0.01   0.01    -1.36   0.17
## ------------------------------------------------------------------
```

```
employee_promotion_prediction_model2_sans_awards <- predict(model2, employee_lm_model_data, type="respo
```

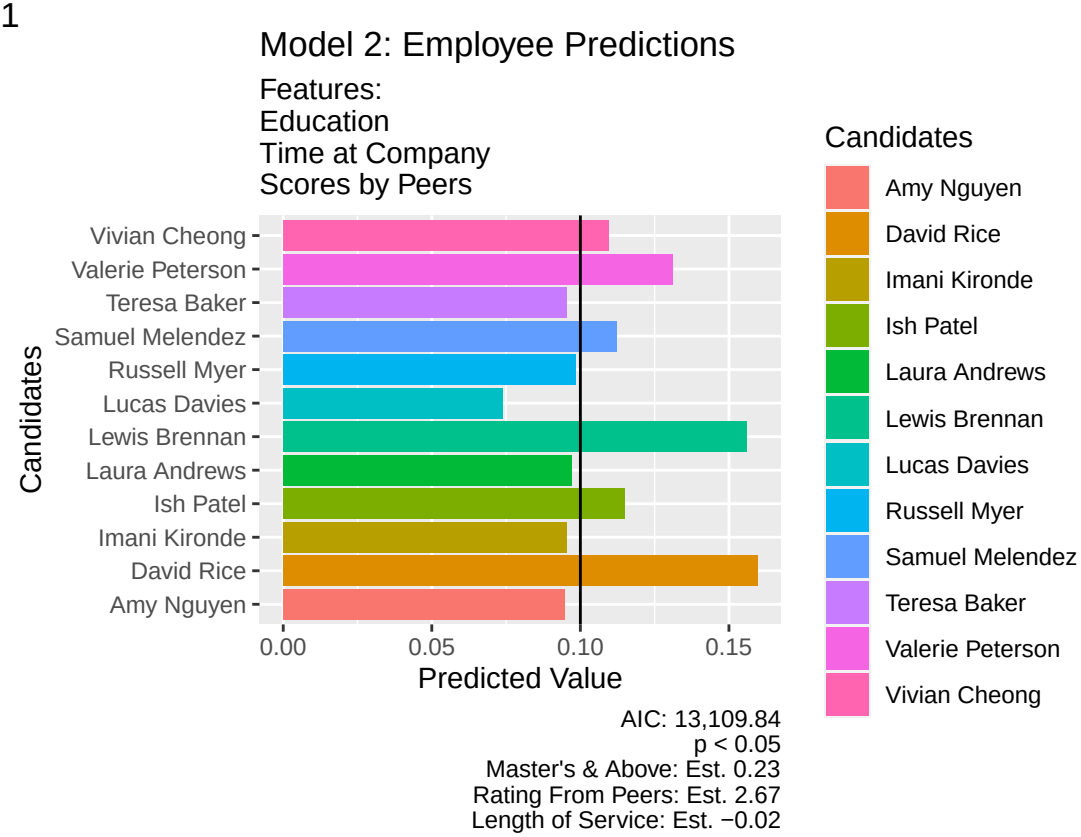
```
lm_employee_promotion_prediction_model2 <- data.frame(candidates_list, employee_promotion_prediction_mo
lm_employee_promotion_prediction_model2
```

```
##     candidates_list employee_promotion_prediction_model2_sans_awards
## 1     Laura Andrews                                         0.09716934
## 2      Teresa Baker                                         0.09537645
## 3     Lewis Brennan                                         0.15591570
## 4     Vivian Cheong                                         0.10950896
## 5      Lucas Davies                                         0.07388774
## 6     Imani Kironde                                         0.09532147
## 7   Samuel Melendez                                         0.11219287
## 8        Amy Nguyen                                         0.09469985
## 9      Russell Myer                                         0.09840105
## 10        Ish Patel                                         0.11476675
## 11 Valerie Peterson                                         0.13099966
## 12       David Rice                                         0.15956893
```

```
ggplot(lm_employee_promotion_prediction_model2, aes(candidates_list, employee_promotion_prediction_mode
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
    labs(title = "Model 2: Employee Predictions", subtitle = "Features: \nEducation \nTime at Company\n
  coord_flip()
```

Plot of Employee Predictions: Model 2 Plot

Figure 11




```
plot_save <- str_c(plot_save_base, "Employee_promotion_prediction_model_2.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Model 3

```
model3 <- glm(is_promoted ~ previous_year_rating + length_of_service, data = train_set, family = "binom
```

```
summ(model3, digits = 2)
```

Model 3 Summary

```
## MODEL INFO:
## Observations: 23190
## Dependent Variable: is_promoted
## Type: Generalized linear model
##   Family: binomial
##   Link function: logit
##
## MODEL FIT:
##  ²(2) = 673.22, p = 0.00
## Pseudo-R² (Cragg-Uhler) = 0.06
## Pseudo-R² (McFadden) = 0.05
## AIC = 13126.64, BIC = 13150.80
##
```

```
## Standard errors: MLE
## ------------------------------------------------------
##                              Est.   S.E.   z val.      p
## ------------------------- ------- ------ -------- ------
## (Intercept)                -4.24   0.10   -44.32   0.00
## previous_year_rating        2.63   0.11    24.20   0.00
## length_of_service          -0.00   0.01    -0.47   0.64
## ------------------------------------------------------
```

```
employee_promotion_prediction_model3_sans_awards_education <- predict(model3, employee_lm_model_data, ty
```

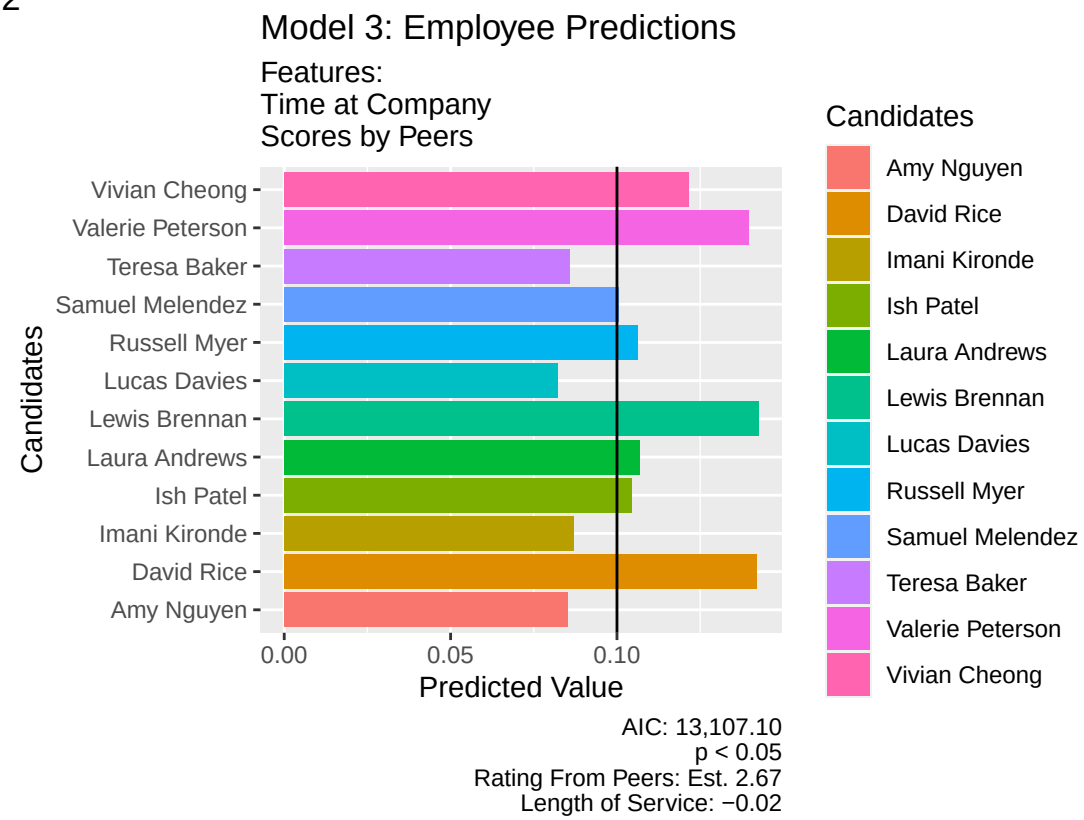
```
lm_employee_promotion_prediction_model3 <- data.frame(candidates_list, employee_promotion_prediction_mod
lm_employee_promotion_prediction_model3
```

```
##     candidates_list employee_promotion_prediction_model3_sans_awards_education
## 1     Laura Andrews                                                  0.10686960
## 2      Teresa Baker                                                  0.08569790
## 3     Lewis Brennan                                                  0.14249100
## 4     Vivian Cheong                                                  0.12145341
## 5      Lucas Davies                                                  0.08224807
## 6     Imani Kironde                                                  0.08692931
## 7   Samuel Melendez                                                  0.10053654
## 8        Amy Nguyen                                                  0.08508241
## 9      Russell Myer                                                  0.10613830
## 10        Ish Patel                                                  0.10438797
## 11 Valerie Peterson                                                  0.13966184
## 12       David Rice                                                  0.14188604
```

```
ggplot(lm_employee_promotion_prediction_model3, aes(candidates_list, employee_promotion_prediction_model
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
  labs(title = "Model 3: Employee Predictions", subtitle = "Features: \nTime at Company\nScores by Peers
  coord_flip()
```

Plot of Employee Predictions: Model 3

Figure 12




```
plot_save <- str_c(plot_save_base, "Employee_promotion_prediction_model_3.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Summary

Model 1 is the best fit. It is best to keep in mind the cutoff is a binary value of if the candidate would be promoted, not a goodness of fit based upon the qualifications for the particular role.

Including Model Weights

Re-Assigning Weights

```
pre_model_weights_360 <- 0.45
pre_model_weights_group <- 0.55

post_model_weights_360 <- 0.40
post_model_weights_group <- 0.55
post_model_weights_model <- 0.05
```

```
weighted_model_education_peerReview_lengthAtCompany_Awards <- employee_model_prediction * post_model_we

weighted_model_education_peerReview_lengthAtCompany <- employee_promotion_prediction_model2_sans_awards

weighted_model_peerReview_lengthAtCompany <- employee_promotion_prediction_model3_sans_awards_education
```

```
(post_model_normalized_and_weighted_360_rankings <- normalized_three_sixty_rankings * post_model_weights
```

```
##  [1] 0.3276 0.2896 0.3776 0.3504 0.2844 0.2932 0.3160 0.2884 0.3248 0.3236
## [11] 0.3716 0.3744
```

```
(post_model_weighted_normalized_group_average <- average_of_top_candidate_highest_value_sum * post_model
```

```
##  [1] 0.06 0.31 0.15 0.43 0.40 0.34 0.31 0.20 0.30 0.46 0.24 0.10
```

Creating Weighted Totals

```
model1_post_model_weighted_totals_education_peerReview_lengthAtCompany_Awards <- post_model_weighted_no
```

```
model2_post_model_weighted_totals_education_peerReview_lengthAtCompany <- post_model_weighted_normalized
```

```
model3_post_model_weighted_totals_peerReview_lengthAtCompany <- post_model_weighted_normalized_group_av
```

```
post_model_col_names_scaled_master_candidates_as_rows_df <- c("Candidates", "Totals", "Rankings_360", "
```

```
post_model_scaled_master_candidates_as_rows_df <- mutate(scaled_master_candidates_as_rows_df, model1_po
```

```
post_model_scaled_master_candidates_as_rows_df
```

```
##          Candidates    Totals Rankings_360 Group_Average Evan_rankings
## 1     Laura Andrews 0.4995409      0.45045    0.04909091    0.00000000
## 2      Teresa Baker 0.6518364      0.39820    0.25363636    0.54545455
## 3     Lewis Brennan 0.6419273      0.51920    0.12272727    0.36363636
## 4     Vivian Cheong 0.8336182      0.48180    0.35181818    0.72727273
## 5      Lucas Davies 0.7183227      0.39105    0.32727273    1.00000000
## 6     Imani Kironde 0.6813318      0.40315    0.27818182    0.90909091
## 7   Samuel Melendez 0.6881364      0.43450    0.25363636    0.63636364
## 8        Amy Nguyen 0.5601864      0.39655    0.16363636    0.09090909
## 9      Russell Myer 0.6920545      0.44660    0.24545455    0.45454545
## 10        Ish Patel 0.8213136      0.44495    0.37636364    0.81818182
## 11 Valerie Peterson 0.7073136      0.51095    0.19636364    0.27272727
## 12       David Rice 0.5966182      0.51480    0.08181818    0.18181818
##    Deepa_rankings Jason_rankings Celeste_rankings Eras_rankings
## 1      0.09090909     0.36363636       0.00000000    0.09090909
## 2      0.45454545     0.90909091       0.27272727    0.63636364
## 3      0.27272727     0.00000000       0.45454545    0.27272727
## 4      0.81818182     0.72727273       0.72727273    0.90909091
## 5      0.54545455     1.00000000       0.36363636    0.72727273
## 6      0.18181818     0.81818182       0.18181818    1.00000000
## 7      0.72727273     0.54545455       0.54545455    0.36363636
## 8      0.36363636     0.18181818       1.00000000    0.18181818
## 9      0.63636364     0.27272727       0.81818182    0.54545455
## 10     1.00000000     0.63636364       0.90909091    0.81818182
## 11     0.90909091     0.45454545       0.09090909    0.45454545
## 12     0.00000000     0.09090909       0.63636364    0.00000000
##    model1_post_model_weighted_totals_education_peerReview_lengthAtCompany_Awards
## 1                                                                       0.3921755
## 2                                                                       0.6040069
## 3                                                                       0.5349519
## 4                                                                       0.7856435
## 5                                                                       0.6879317
```

```
## 6                                                                    0.6377124
## 7                                                                    0.6311502
## 8                                                                    0.4927754
## 9                                                                    0.6292868
## 10                                                                   0.7889964
## 11                                                                   0.6174995
## 12                                                                   0.5059532
##    model2_post_model_weighted_totals_education_peerReview_lengthAtCompany
## 1                                                               0.3924585
## 2                                                               0.6043688
## 3                                                               0.5353958
## 4                                                               0.7858754
## 5                                                               0.6880944
## 6                                                               0.6379661
## 7                                                               0.6316096
## 8                                                               0.4931350
## 9                                                               0.6297201
## 10                                                              0.7893383
## 11                                                              0.6181500
## 12                                                              0.4823784
##    model3_post_model_weighted_totals_peerReview_lengthAtCompany
## 1                                                     0.3929435
## 2                                                     0.6038849
## 3                                                     0.5347246
## 4                                                     0.7864727
## 5                                                     0.6885124
## 6                                                     0.6375465
## 7                                                     0.6310268
## 8                                                     0.4926541
## 9                                                     0.6301069
## 10                                                    0.7888194
## 11                                                    0.6185831
## 12                                                    0.4814943
```

```
colnames(post_model_scaled_master_candidates_as_rows_df) <- post_model_col_names_scaled_master_candidat
```

```
post_model_scaled_master_candidates_as_rows_df <- select(post_model_scaled_master_candidates_as_rows_df
```

```
Ranked_model_1_post_model_scaled_master_candidates_as_rows_df <- arrange(post_model_scaled_master_candi
```

```
Ranked_model_2_post_model_scaled_master_candidates_as_rows_df <- arrange(post_model_scaled_master_candi
```

```
Ranked_model_3_post_model_scaled_master_candidates_as_rows_df <- arrange(post_model_scaled_master_candi
```

```
Ranked_model_1_post_model_scaled_master_candidates_as_rows_df
```

```
##          Candidates    Totals totals_model1 totals_model2 totals_model3
## 1         Ish Patel 0.8213136     0.7889964     0.7893383     0.7888194
## 2     Vivian Cheong 0.8336182     0.7856435     0.7858754     0.7864727
## 3      Lucas Davies 0.7183227     0.6879317     0.6880944     0.6885124
## 4     Imani Kironde 0.6813318     0.6377124     0.6379661     0.6375465
## 5   Samuel Melendez 0.6881364     0.6311502     0.6316096     0.6310268
## 6      Russell Myer 0.6920545     0.6292868     0.6297201     0.6301069
## 7  Valerie Peterson 0.7073136     0.6174995     0.6181500     0.6185831
## 8      Teresa Baker 0.6518364     0.6040069     0.6043688     0.6038849
```

```
## 9     Lewis Brennan 0.6419273     0.5349519     0.5353958     0.5347246
## 10       David Rice 0.5966182     0.5059532     0.4823784     0.4814943
## 11       Amy Nguyen 0.5601864     0.4927754     0.4931350     0.4926541
## 12    Laura Andrews 0.4995409     0.3921755     0.3924585     0.3929435
##    Rankings_360 Group_Average Evan_rankings Deepa_rankings Jason_rankings
## 1       0.44495    0.37636364    0.81818182     1.00000000     0.63636364
## 2       0.48180    0.35181818    0.72727273     0.81818182     0.72727273
## 3       0.39105    0.32727273    1.00000000     0.54545455     1.00000000
## 4       0.40315    0.27818182    0.90909091     0.18181818     0.81818182
## 5       0.43450    0.25363636    0.63636364     0.72727273     0.54545455
## 6       0.44660    0.24545455    0.45454545     0.63636364     0.27272727
## 7       0.51095    0.19636364    0.27272727     0.90909091     0.45454545
## 8       0.39820    0.25363636    0.54545455     0.45454545     0.90909091
## 9       0.51920    0.12272727    0.36363636     0.27272727     0.00000000
## 10      0.51480    0.08181818    0.18181818     0.00000000     0.09090909
## 11      0.39655    0.16363636    0.09090909     0.36363636     0.18181818
## 12      0.45045    0.04909091    0.00000000     0.09090909     0.36363636
##    Celeste_rankings
## 1        0.90909091
## 2        0.72727273
## 3        0.36363636
## 4        0.18181818
## 5        0.54545455
## 6        0.81818182
## 7        0.09090909
## 8        0.27272727
## 9        0.45454545
## 10       0.63636364
## 11       1.00000000
## 12       0.00000000
```

```
Ranked_model_2_post_model_scaled_master_candidates_as_rows_df
```

```
##          Candidates    Totals totals_model1 totals_model2 totals_model3
## 1         Ish Patel 0.8213136     0.7889964     0.7893383     0.7888194
## 2     Vivian Cheong 0.8336182     0.7856435     0.7858754     0.7864727
## 3      Lucas Davies 0.7183227     0.6879317     0.6880944     0.6885124
## 4     Imani Kironde 0.6813318     0.6377124     0.6379661     0.6375465
## 5   Samuel Melendez 0.6881364     0.6311502     0.6316096     0.6310268
## 6      Russell Myer 0.6920545     0.6292868     0.6297201     0.6301069
## 7  Valerie Peterson 0.7073136     0.6174995     0.6181500     0.6185831
## 8      Teresa Baker 0.6518364     0.6040069     0.6043688     0.6038849
## 9     Lewis Brennan 0.6419273     0.5349519     0.5353958     0.5347246
## 10       Amy Nguyen 0.5601864     0.4927754     0.4931350     0.4926541
## 11       David Rice 0.5966182     0.5059532     0.4823784     0.4814943
## 12    Laura Andrews 0.4995409     0.3921755     0.3924585     0.3929435
##    Rankings_360 Group_Average Evan_rankings Deepa_rankings Jason_rankings
## 1       0.44495    0.37636364    0.81818182     1.00000000     0.63636364
## 2       0.48180    0.35181818    0.72727273     0.81818182     0.72727273
## 3       0.39105    0.32727273    1.00000000     0.54545455     1.00000000
## 4       0.40315    0.27818182    0.90909091     0.18181818     0.81818182
## 5       0.43450    0.25363636    0.63636364     0.72727273     0.54545455
## 6       0.44660    0.24545455    0.45454545     0.63636364     0.27272727
## 7       0.51095    0.19636364    0.27272727     0.90909091     0.45454545
## 8       0.39820    0.25363636    0.54545455     0.45454545     0.90909091
```

```
## 9       0.51920    0.12272727    0.36363636     0.27272727     0.00000000
## 10      0.39655    0.16363636    0.09090909     0.36363636     0.18181818
## 11      0.51480    0.08181818    0.18181818     0.00000000     0.09090909
## 12      0.45045    0.04909091    0.00000000     0.09090909     0.36363636
##    Celeste_rankings
## 1        0.90909091
## 2        0.72727273
## 3        0.36363636
## 4        0.18181818
## 5        0.54545455
## 6        0.81818182
## 7        0.09090909
## 8        0.27272727
## 9        0.45454545
## 10       1.00000000
## 11       0.63636364
## 12       0.00000000
```

```
Ranked_model_3_post_model_scaled_master_candidates_as_rows_df
```

```
##          Candidates    Totals totals_model1 totals_model2 totals_model3
## 1         Ish Patel 0.8213136     0.7889964     0.7893383     0.7888194
## 2     Vivian Cheong 0.8336182     0.7856435     0.7858754     0.7864727
## 3      Lucas Davies 0.7183227     0.6879317     0.6880944     0.6885124
## 4     Imani Kironde 0.6813318     0.6377124     0.6379661     0.6375465
## 5   Samuel Melendez 0.6881364     0.6311502     0.6316096     0.6310268
## 6      Russell Myer 0.6920545     0.6292868     0.6297201     0.6301069
## 7  Valerie Peterson 0.7073136     0.6174995     0.6181500     0.6185831
## 8      Teresa Baker 0.6518364     0.6040069     0.6043688     0.6038849
## 9     Lewis Brennan 0.6419273     0.5349519     0.5353958     0.5347246
## 10       Amy Nguyen 0.5601864     0.4927754     0.4931350     0.4926541
## 11       David Rice 0.5966182     0.5059532     0.4823784     0.4814943
## 12    Laura Andrews 0.4995409     0.3921755     0.3924585     0.3929435
##    Rankings_360 Group_Average Evan_rankings Deepa_rankings Jason_rankings
## 1       0.44495    0.37636364    0.81818182     1.00000000     0.63636364
## 2       0.48180    0.35181818    0.72727273     0.81818182     0.72727273
## 3       0.39105    0.32727273    1.00000000     0.54545455     1.00000000
## 4       0.40315    0.27818182    0.90909091     0.18181818     0.81818182
## 5       0.43450    0.25363636    0.63636364     0.72727273     0.54545455
## 6       0.44660    0.24545455    0.45454545     0.63636364     0.27272727
## 7       0.51095    0.19636364    0.27272727     0.90909091     0.45454545
## 8       0.39820    0.25363636    0.54545455     0.45454545     0.90909091
## 9       0.51920    0.12272727    0.36363636     0.27272727     0.00000000
## 10      0.39655    0.16363636    0.09090909     0.36363636     0.18181818
## 11      0.51480    0.08181818    0.18181818     0.00000000     0.09090909
## 12      0.45045    0.04909091    0.00000000     0.09090909     0.36363636
##    Celeste_rankings
## 1        0.90909091
## 2        0.72727273
## 3        0.36363636
## 4        0.18181818
## 5        0.54545455
## 6        0.81818182
## 7        0.09090909
## 8        0.27272727
```

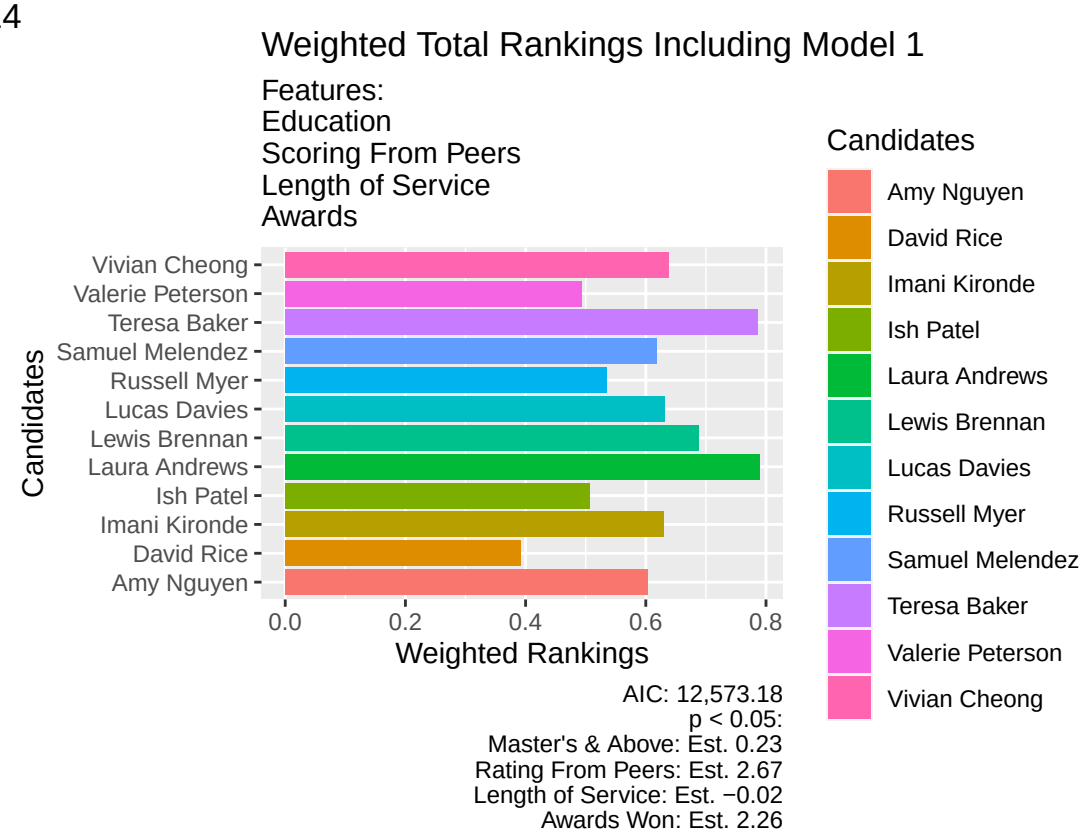
```
## 9        0.45454545
## 10       1.00000000
## 11       0.63636364
## 12       0.00000000
```

Plotting Models

```
ggplot(Ranked_model_1_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model1, fill = candidates_list)) +
  labs(title = "Weighted Total Rankings Including Model 1", y = "Weighted Rankings", x = "Candidates", f
  coord_flip()
```

Weighted Total Rankings With Model 1

Figure 14



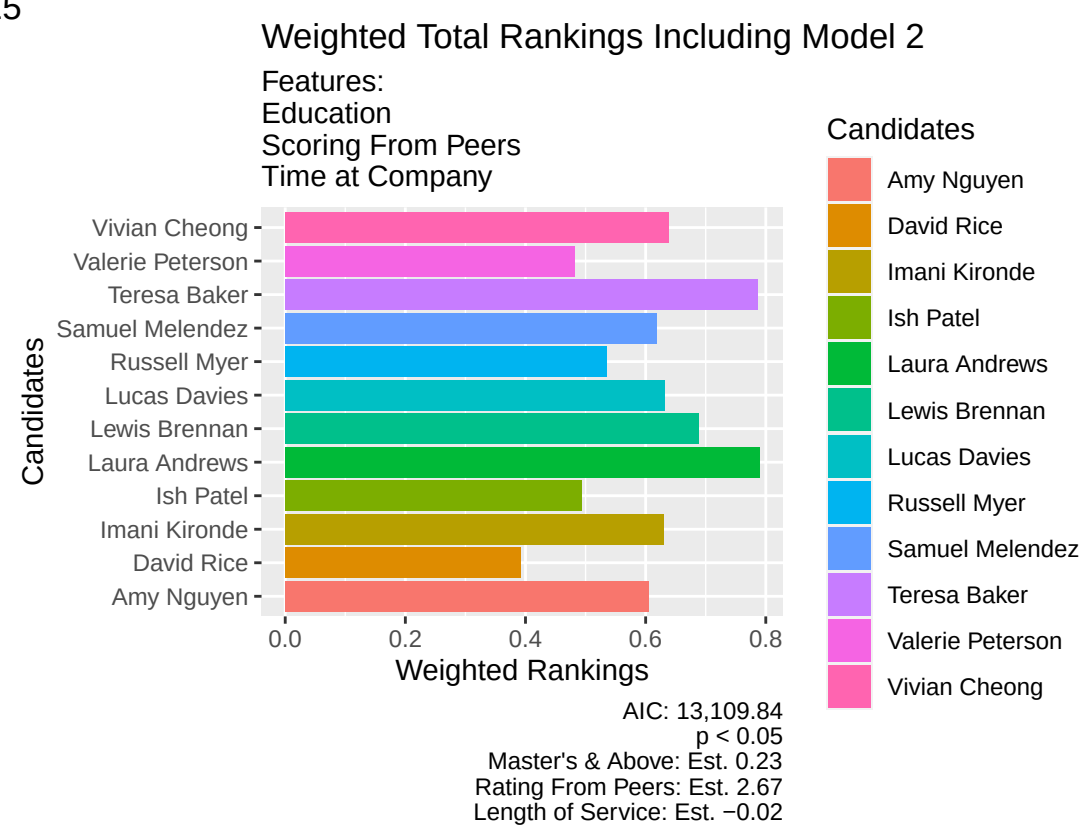

```
plot_save = str_c(plot_save_base, "ranked_candidates_model1.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

```
ggplot(Ranked_model_2_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model2, fill = candidates_list)) +
labs(title = "Weighted Total Rankings Including Model 2", y = "Weighted Rankings", x = "Candidates", fi
  coord_flip()
```

Weighted Total Rankings With Model 2

Figure 15



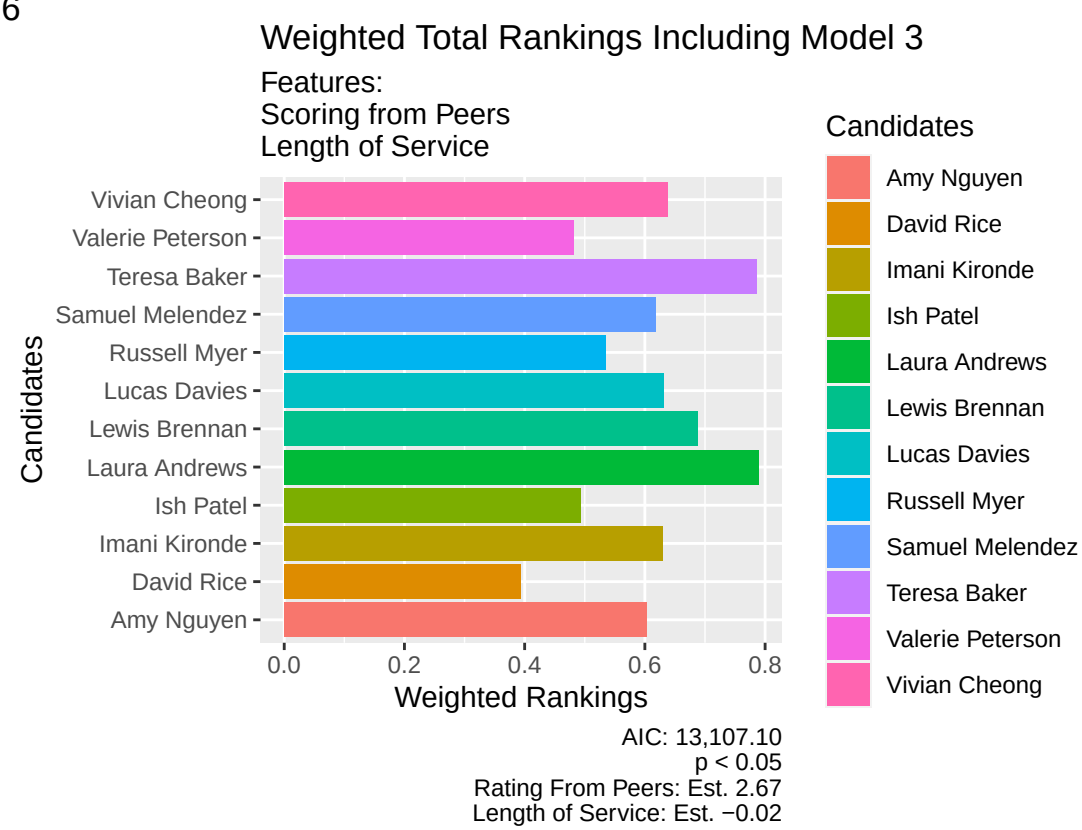

```
plot_save = str_c(plot_save_base, "ranked_candidates_model2.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

```
ggplot(Ranked_model_3_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model3, fill = candidates_list)) +
    labs(title = "Weighted Total Rankings Including Model 3", y = "Weighted Rankings", x = "Candidates"
  coord_flip()
```

Weighted Total Rankings With Model 3

Figure 16




```
plot_save = str_c(plot_save_base, "ranked_candidates_model3.png")
ggsave(plot_save)
```

```
## Saving 6.5 x 4.5 in image
```

Section: Final Plots.

Figure 1: Deepa's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Deepa, fill = candidates_list)) +
  labs(title = "Deepa's Rankings", y = "Rank of Candidate", x = "Candidates", fill = "Candidates", alpha
  coord_flip()
```

Figure 1

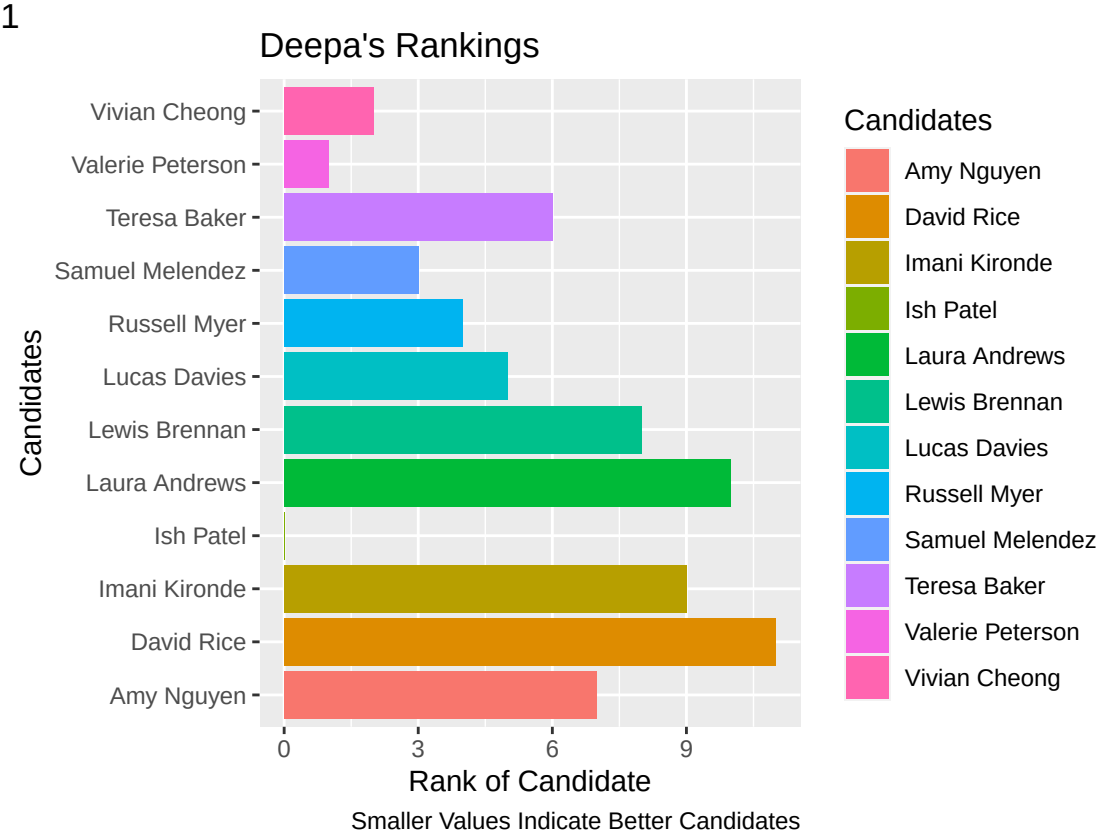


Figure 2: Jason's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Jason, fill = candidates_list)) +
  labs(title = "Jason's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bet
  coord_flip()
```

Figure 2

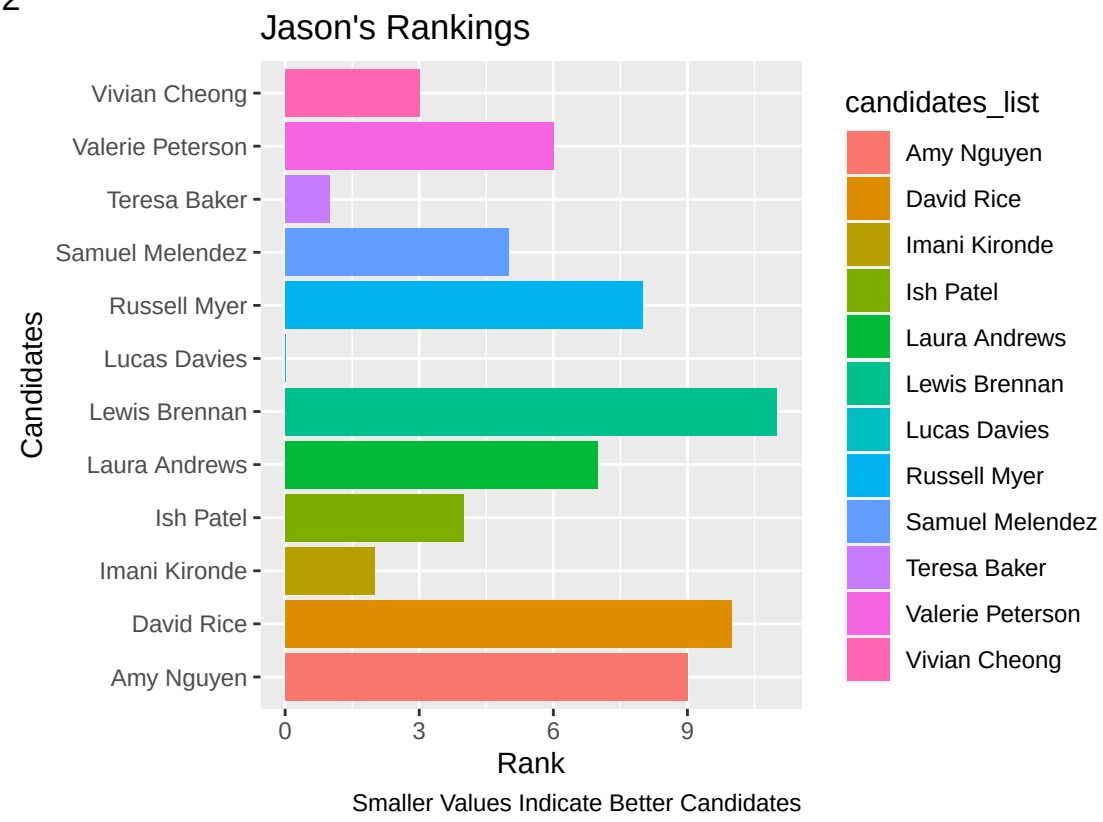


Figure 3: Evan's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Evan, fill = candidates_list)) +
  labs(title = "Evan's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bett
  coord_flip()
```

Figure 3

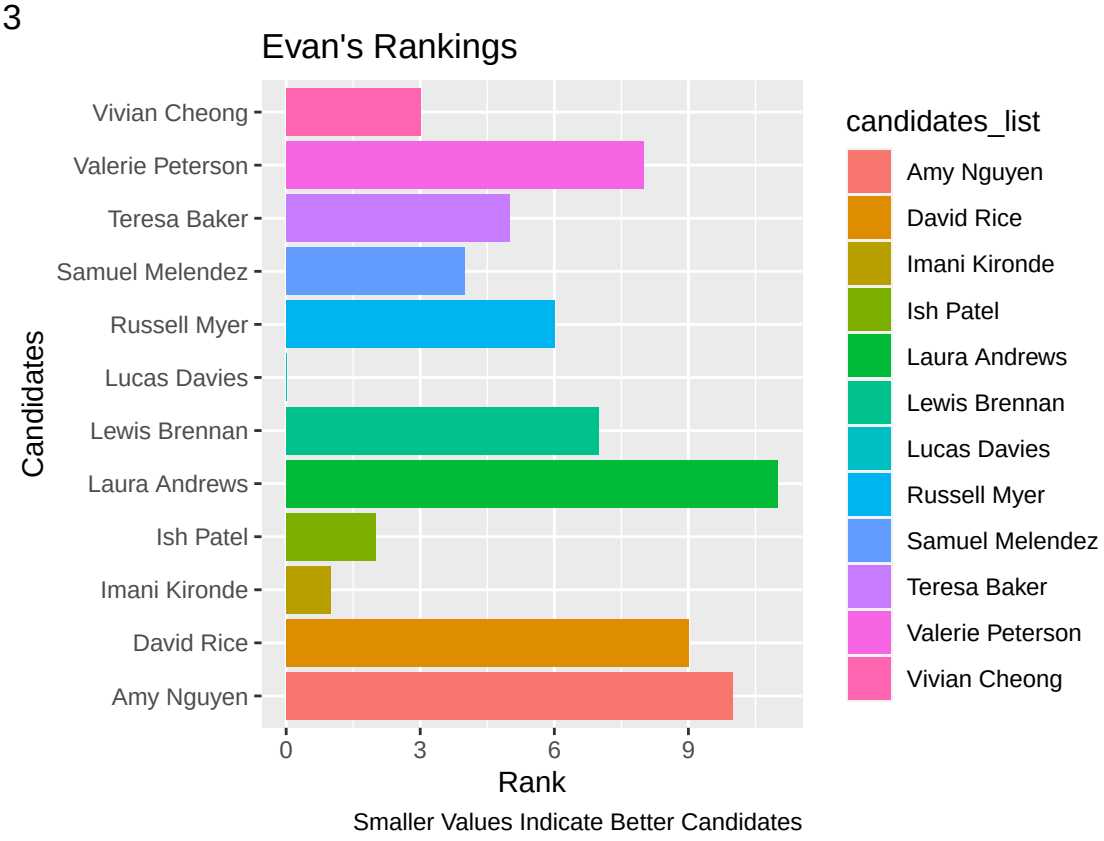


Figure 4: Eras' Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Eras, fill = candidates_list)) +
 labs(title = "Eras's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Bett
  coord_flip()
```

Figure 4

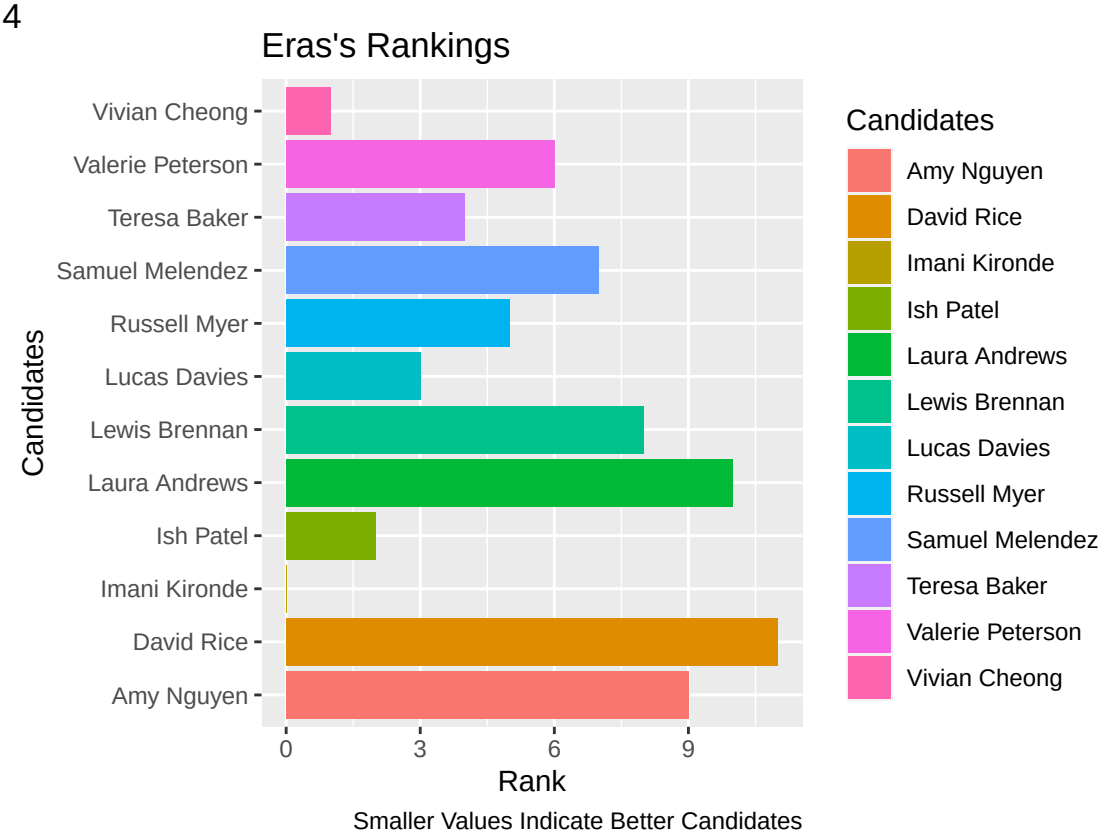


Figure 5: Celeste's Scores

```
ggplot(candidate_df) +
  geom_col(aes(candidates_list, Celeste, fill = candidates_list)) +
  labs(title = "Celeste's Rankings", x = "Candidates", y = "Rank", caption = "Smaller Values Indicate Be
  coord_flip()
```

Figure 5

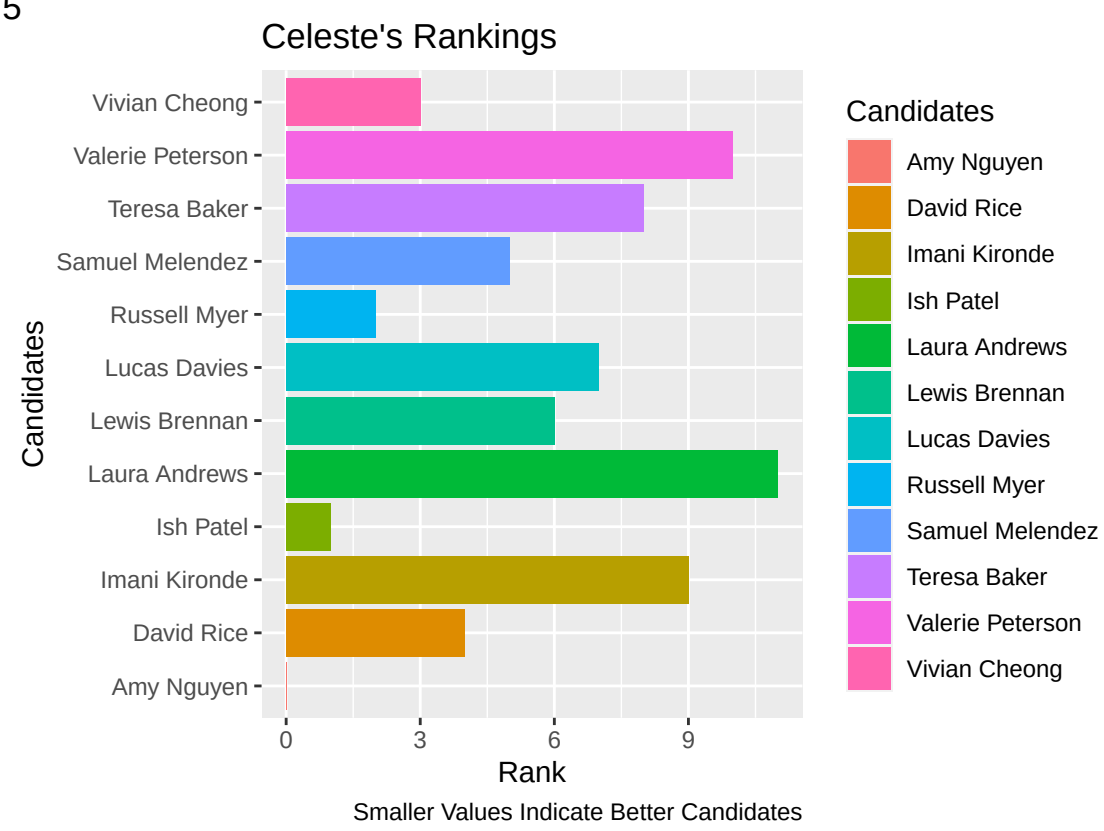


Figure 6: 360 Rankings

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = Candidates, y = Rankings_360, fill = Rankings_360, alpha = Rankings_360)) +
  ylab("360 Score") +
labs(title = "360 Rankings", fill = "360 Rankings", alpha = "360 Rankings", caption = "Values Range fro
  coord_flip()
```

Figure 6

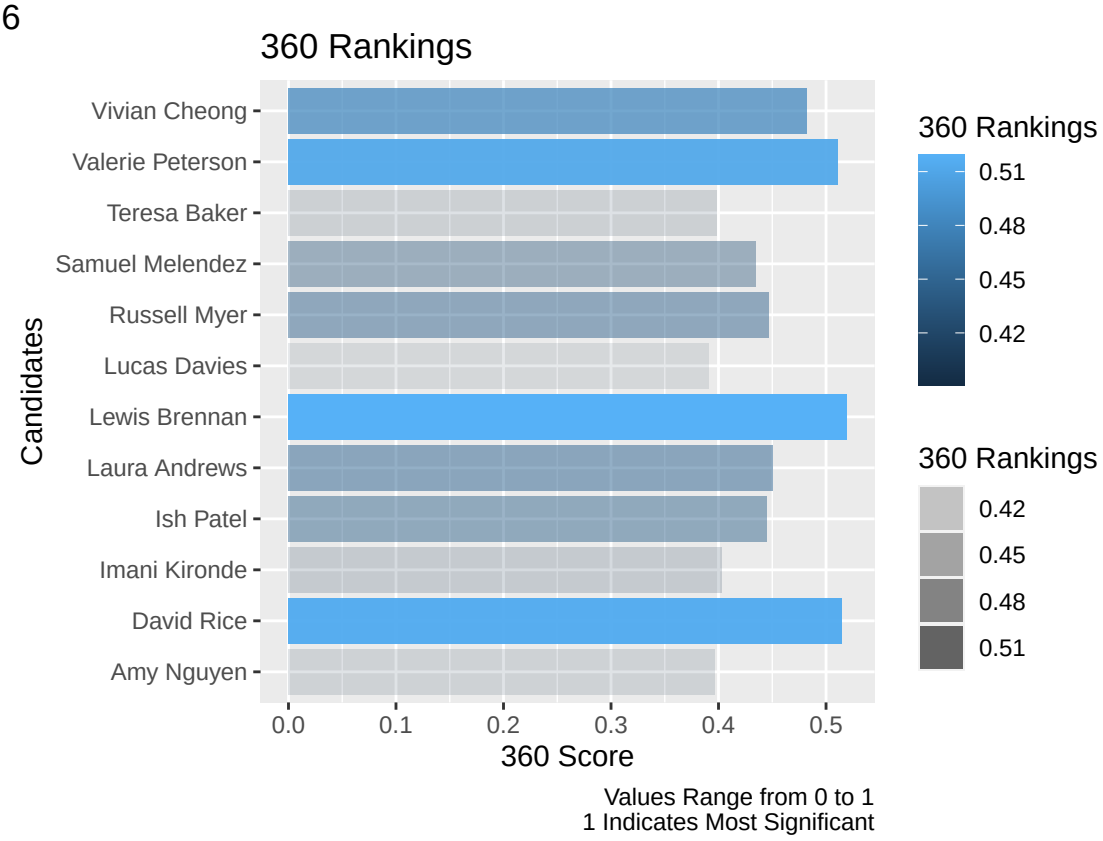


Figure 7: Group Rankings

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = Candidates, y = Group_Average, fill = Group_Average, alpha = Group_Average)) +
  ylab("Group Average Score") +
    labs(title = "Group Average", fill = "Group Average", alpha = "Group Average", caption = "Values Ra
  coord_flip()
```

Figure 7

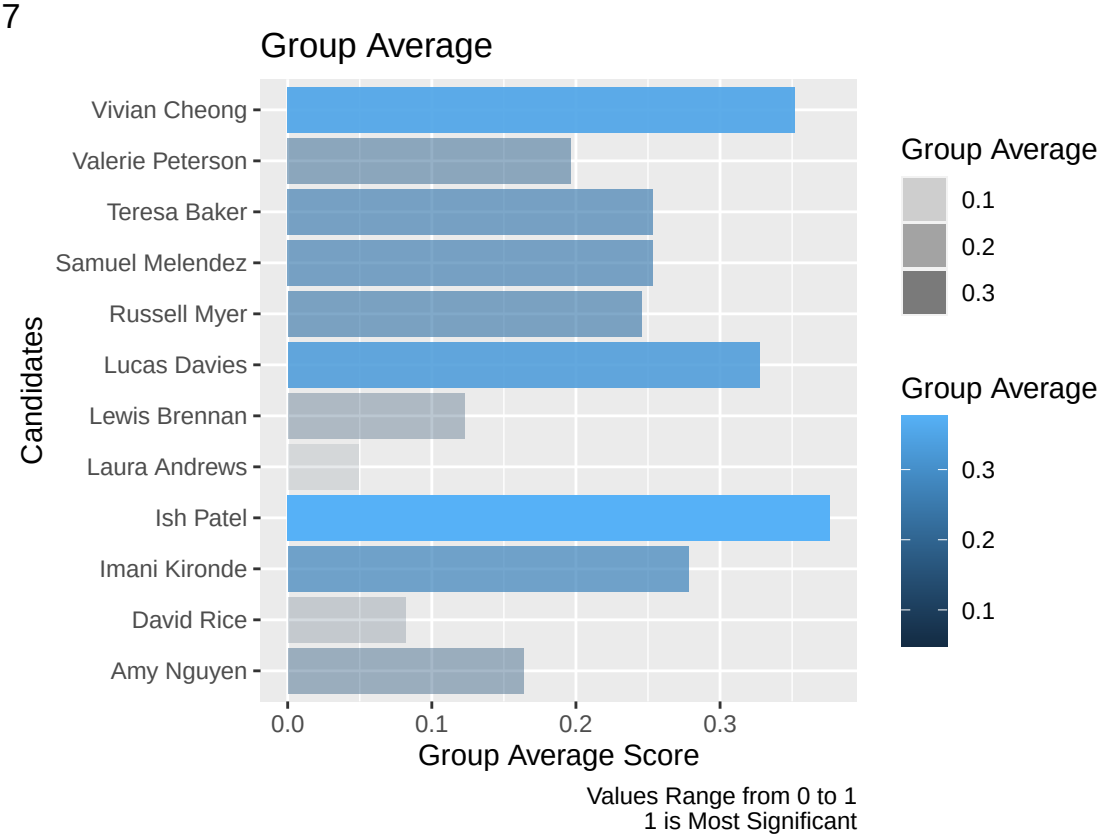


Figure 8: Ranked Candidates By Weighted Totals (original)

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = candidates_list, y = weighted_totals, fill = candidates_list)) +
    labs(y = "Weighted Rankings", x = "Candidates", fill = "Candidates", title = "Ranked Candidates By
  coord_flip()
```

Figure 8

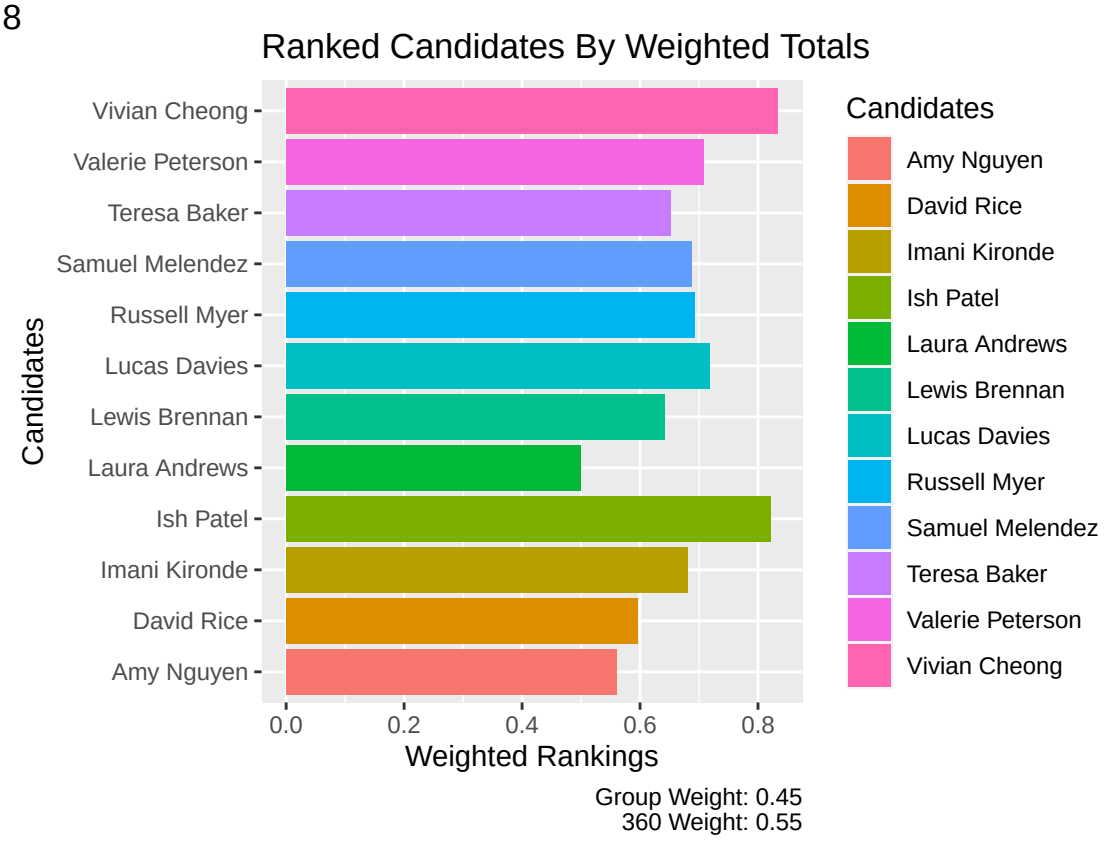


Figure 9: Distribution of Test Set Predictions

```
ggplot(test_set, aes(is_promoted_prediction)) +
  geom_histogram(color = "white", fill = "lightblue", bins = 30) +
  labs(title = "Probability Distribution of Employee Promotion Predictions", x = "Prediction of Promoti
```

Figure 9

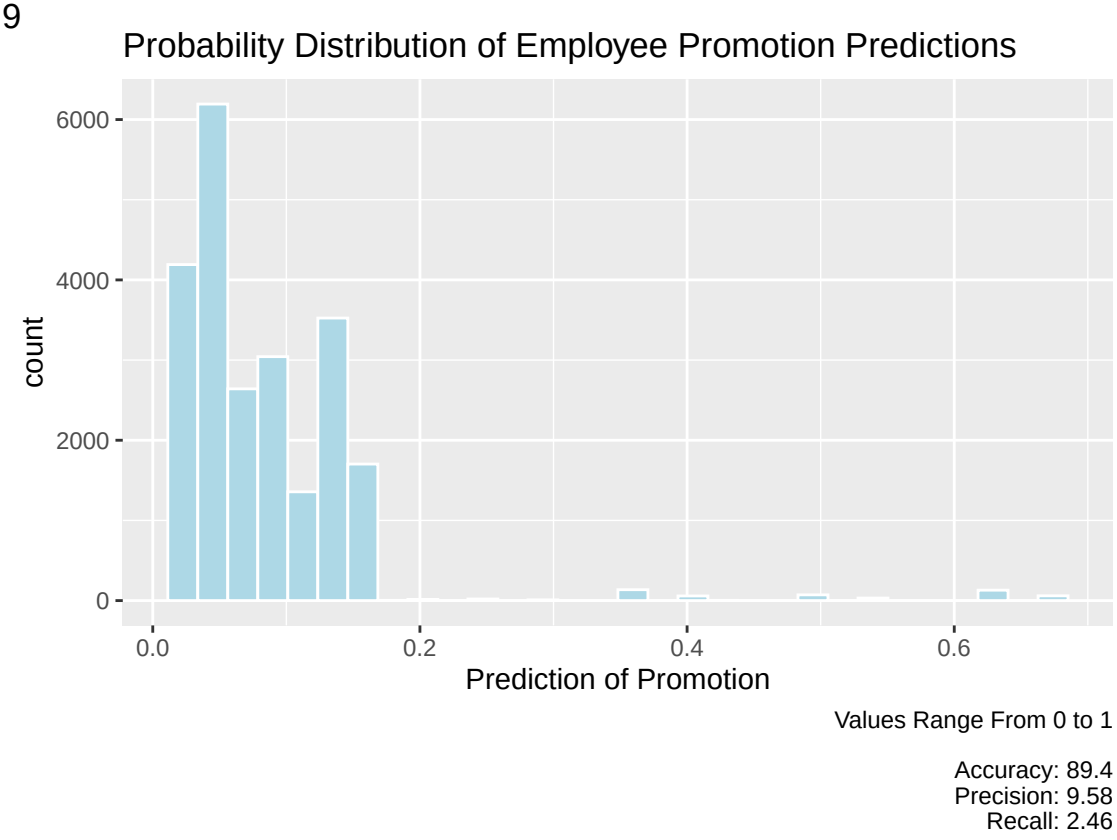


Figure 10: Plot of Employee Predictions Model 1

```
ggplot(lm_predicted_df, aes(candidates_list, employee_model_prediction)) +
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
    labs(y = "Predicted Score", x = "Candidate", title = "Model 1: Employee Predictions", subtitle = "F
  coord_flip()
```

Figure 10

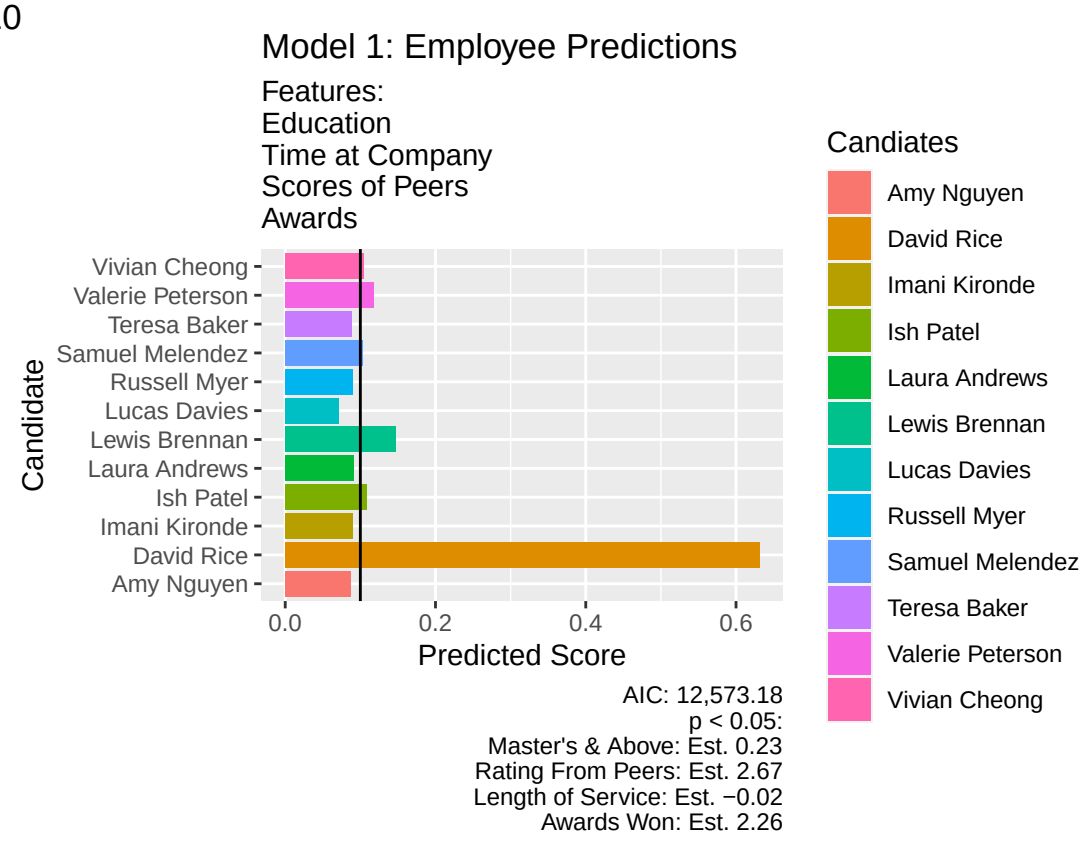


Figure 11: Plot of Employee Predictions: Model 2

```
ggplot(lm_employee_promotion_prediction_model2, aes(candidates_list, employee_promotion_prediction_model
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
    labs(title = "Model 2: Employee Predictions", subtitle = "Features: \nEducation \nTime at Company\n
  coord_flip()
```

Figure 11

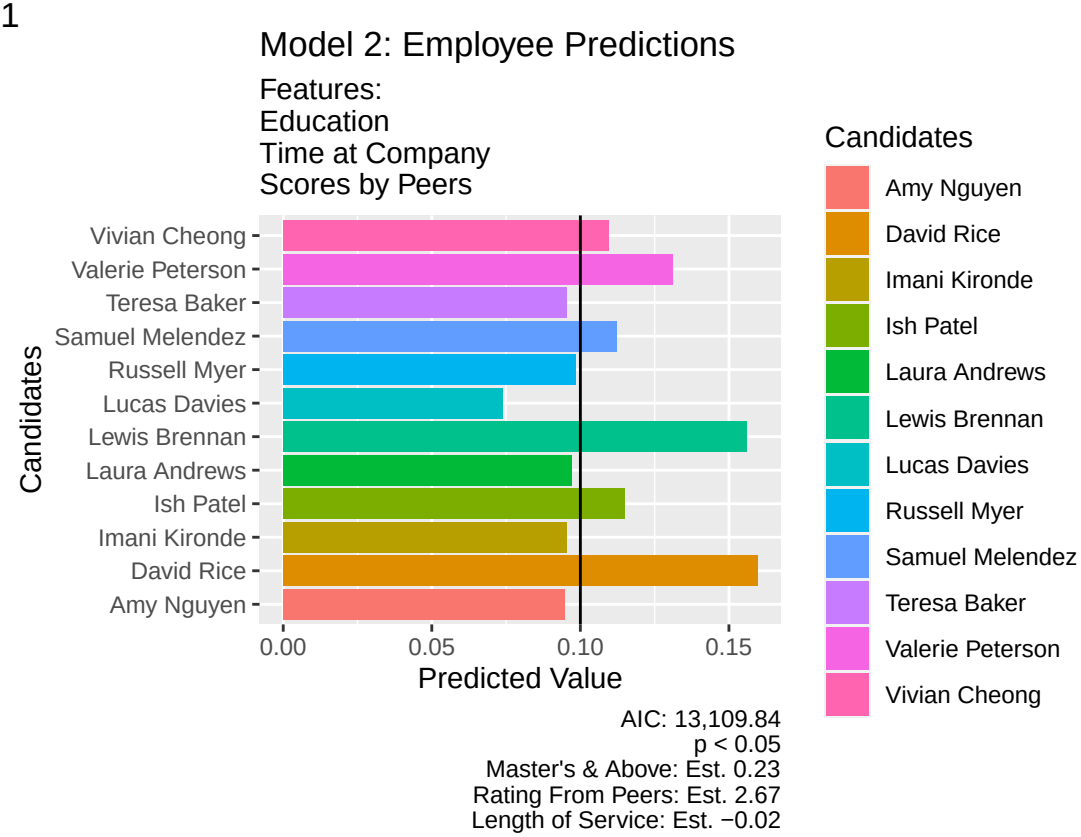


Figure 12: Plot of Employee Predictions: Model 3

```
ggplot(lm_employee_promotion_prediction_model3, aes(candidates_list, employee_promotion_prediction_model
  geom_col(aes(fill = candidates_list)) +
  geom_hline(yintercept = .1) +
  labs(title = "Model 3: Employee Predictions", subtitle = "Features: \nTime at Company\nScores by Peers
  coord_flip()
```

Figure 12

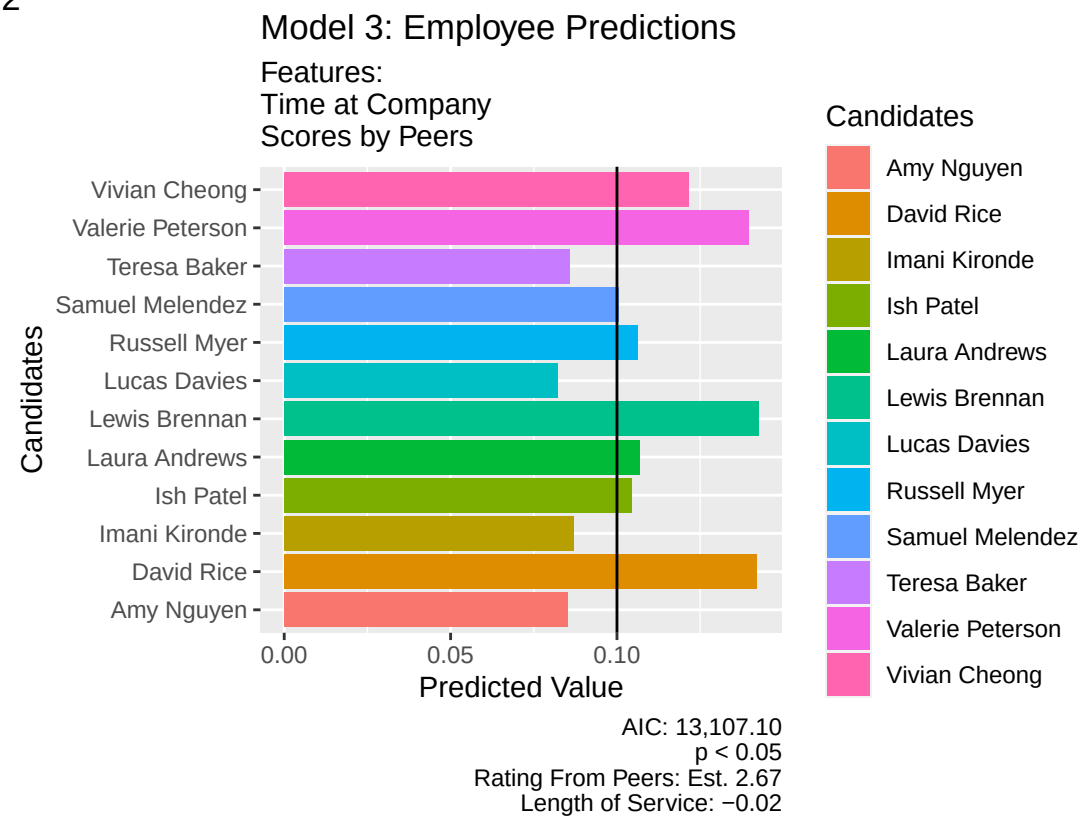


Figure 13: Weighted Total Rankings Without Model

```
ggplot(ranked_candidates_weighted_totals) +
  geom_col(aes(x = candidates_list, y = weighted_totals, fill = candidates_list, alpha = weighted_totals
    labs(title = "Weighted Total Rankings", y = "Weighted Rankings", x = "Candidates", fill = "Candidat
  coord_flip()
```

Figure 13

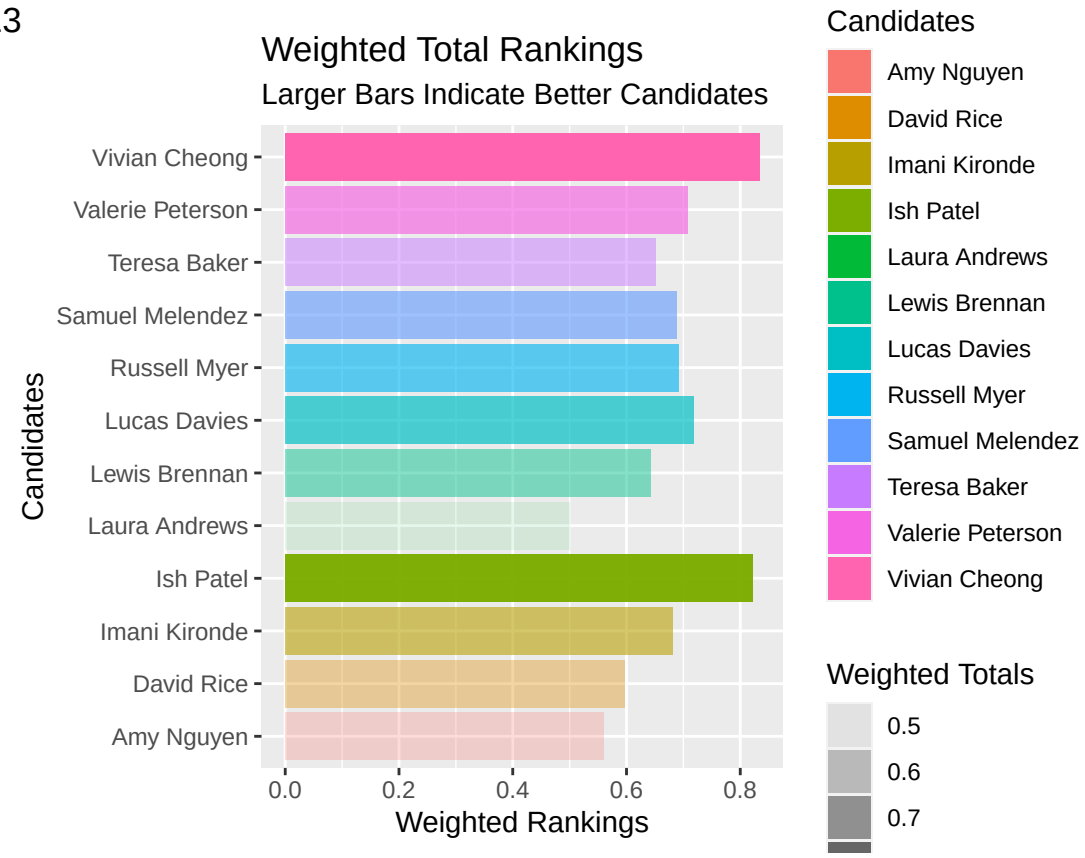


Figure 14: Weighted Total Rankings With Model 1

```
ggplot(Ranked_model_1_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model1, fill = candidates_list)) +
  labs(title = "Weighted Total Rankings Including Model 1", y = "Weighted Rankings", x = "Candidates",
  coord_flip()
```

Figure 14

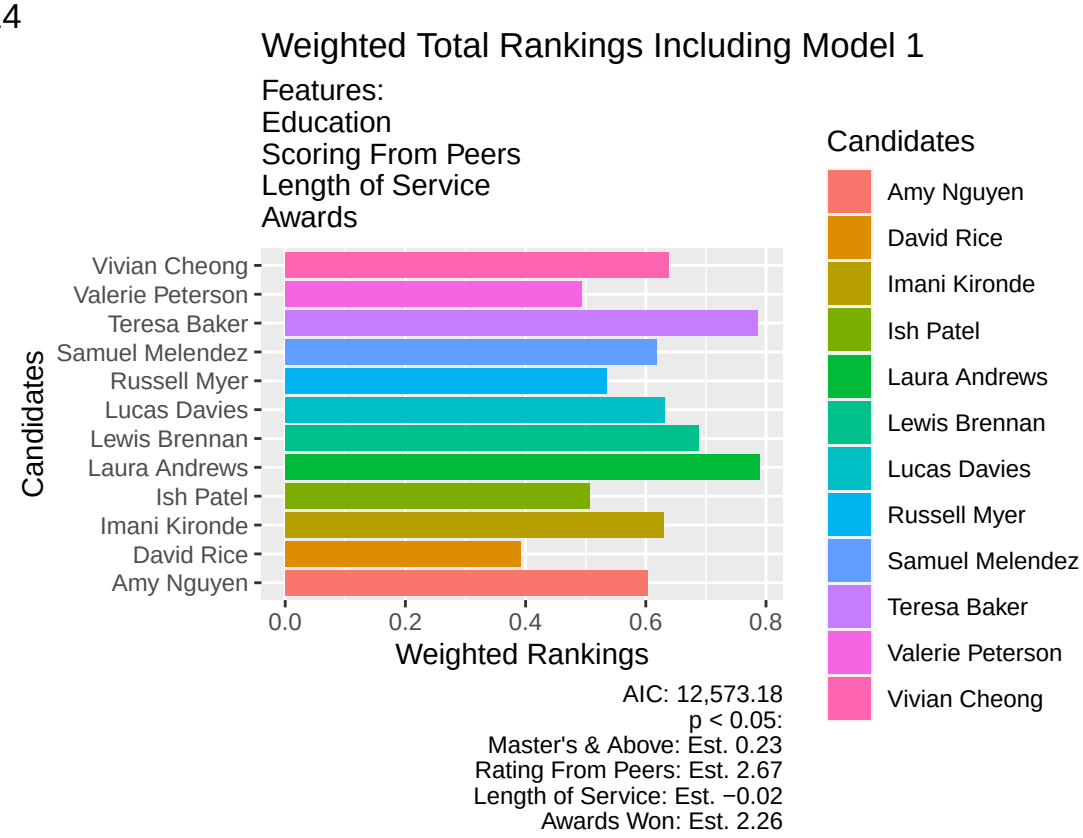


Figure 15: Weighted Total Rankings With Model 2

```
ggplot(Ranked_model_2_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model2, fill = candidates_list)) +
labs(title = "Weighted Total Rankings Including Model 2", y = "Weighted Rankings", x = "Candidates", fi
  coord_flip()
```

Figure 15

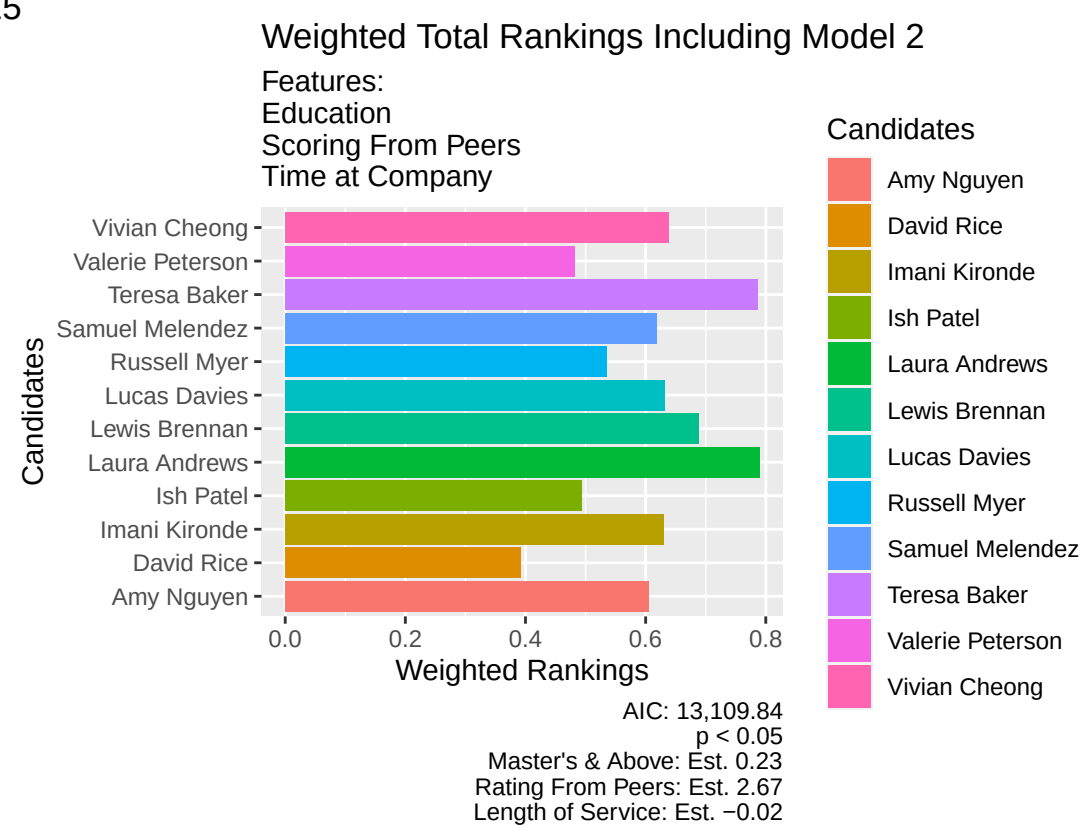
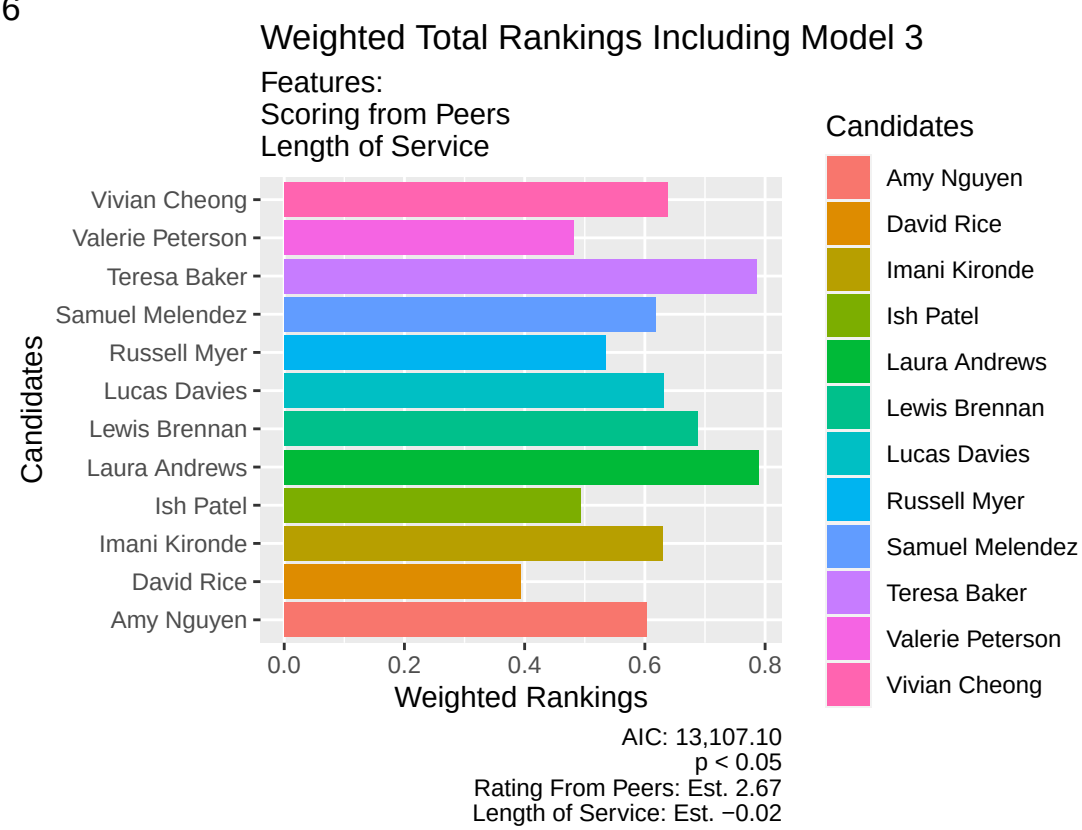


Figure 16: Weighted Total Rankings With Model 3

```
ggplot(Ranked_model_3_post_model_scaled_master_candidates_as_rows_df) +
  geom_col(aes(x = candidates_list, y = totals_model3, fill = candidates_list)) +
    labs(title = "Weighted Total Rankings Including Model 3", y = "Weighted Rankings", x = "Candidates"
  coord_flip()
```

Figure 16


Weighted Total Rankings Including Model 3
Features:
Scoring from Peers
Length of Service
Candidates
Vivian Cheong
Valerie Peterson
Teresa Baker
Samuel Melendez
Russell Myer
Lucas Davies
Lewis Brennan
Laura Andrews
Ish Patel
Imani Kironde
David Rice
Amy Nguyen
Candidates
0.0
0.2
0.4
0.6
0.8
Weighted Rankings
AIC: 13,107.10
p < 0.05
Rating From Peers: Est. 2.67
Length of Service: Est. −0.02