Problem Set 3

Political Data Science - Spring 2020

Gencer, Alper Sukru

Due February 27, 10:00 AM (Before Class)

Instructions

1. The following questions should each be answered within an R script. Be sure to provide many comments in the script to facilitate grading. Undocumented code will not be graded.
2. Work on git. Fork the repository found at https://github.com/domlockett/PDS-PS3 and add your code, committing and pushing frequently. Use meaningful commit messages – these may affect your grade.
3. You may work in teams, but each student should develop their own R script. To be clear, there should be no copy and paste. Each keystroke in the assignment should be your own.
4. If you have any questions regarding the Problem Set, contact the TAs or use their office hours.
5. For students new to programming, this may take a while. Get started.
6. You will need to install ggplot2 and dplyr to complete this dataset.

Q1 - ggplot2

1. Finish the exercise we started in class on 2/11/2020:
    - Alabama, Arkansas, California, Colorado, Maine, Massachusetts, Minnesota, North Carolina, Oklahoma, Tennessee, Texas, Utah, Vermont, and Virginia will all hold their primraries on March 3
    - You have been assigned to create a visulaization of the state of the race for this date.
    - You will make a plot to show this.
    - In addition to the kinds of issues discussed above 1- Change to the minimial theme 2– Figure out how to change the axis labels and legends beyond the defaults
    - Visit https://ggplot2.tidyverse.org/reference/
2. Finish the exercise we started in class on 2/13/2020:
    - Re-organize the dataset so that there is only one row for each candidate-state dyad
    - Feel free to limit this down to only the relevant candidates
    - Compare the size of this dataset to our original dataset using the object size command.

A1 - ggplot2

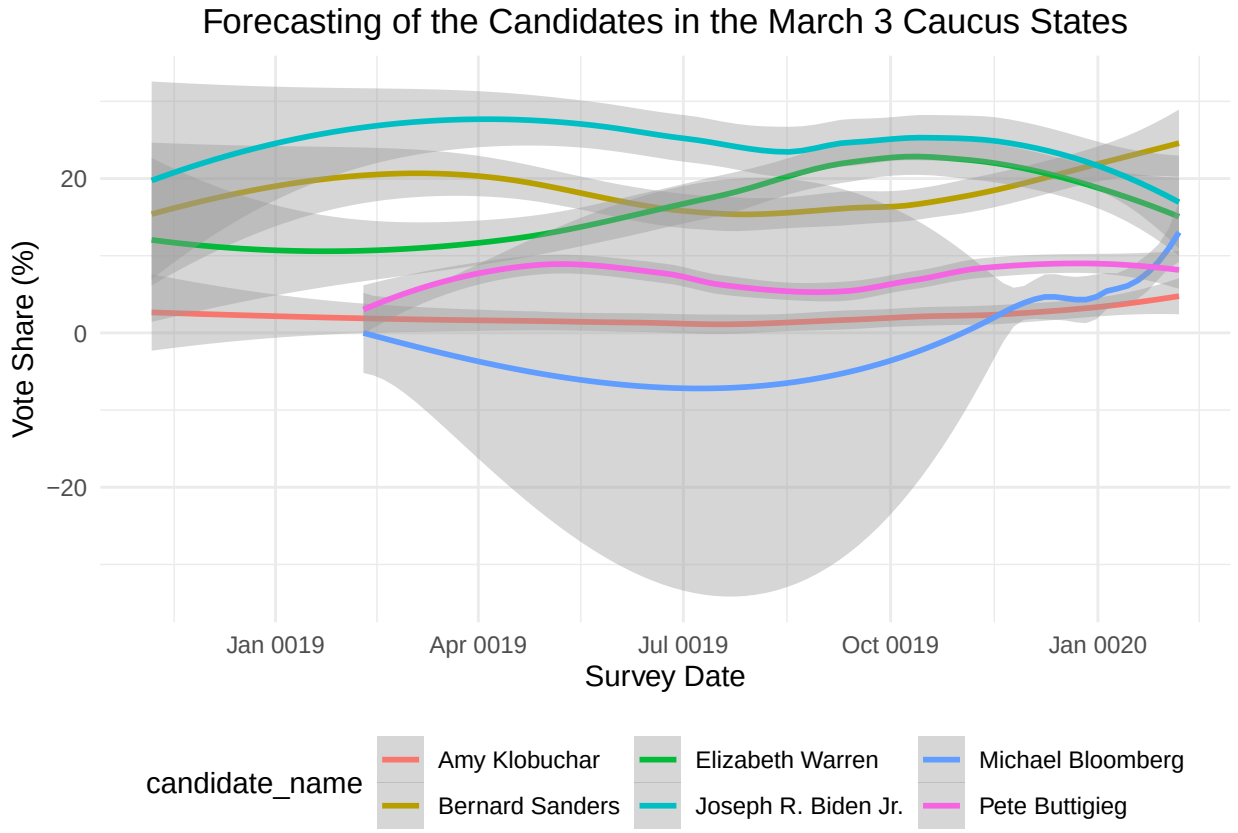
```
rm(list = ls())
```

Finish the exercise we started in class on 2/11/2020:

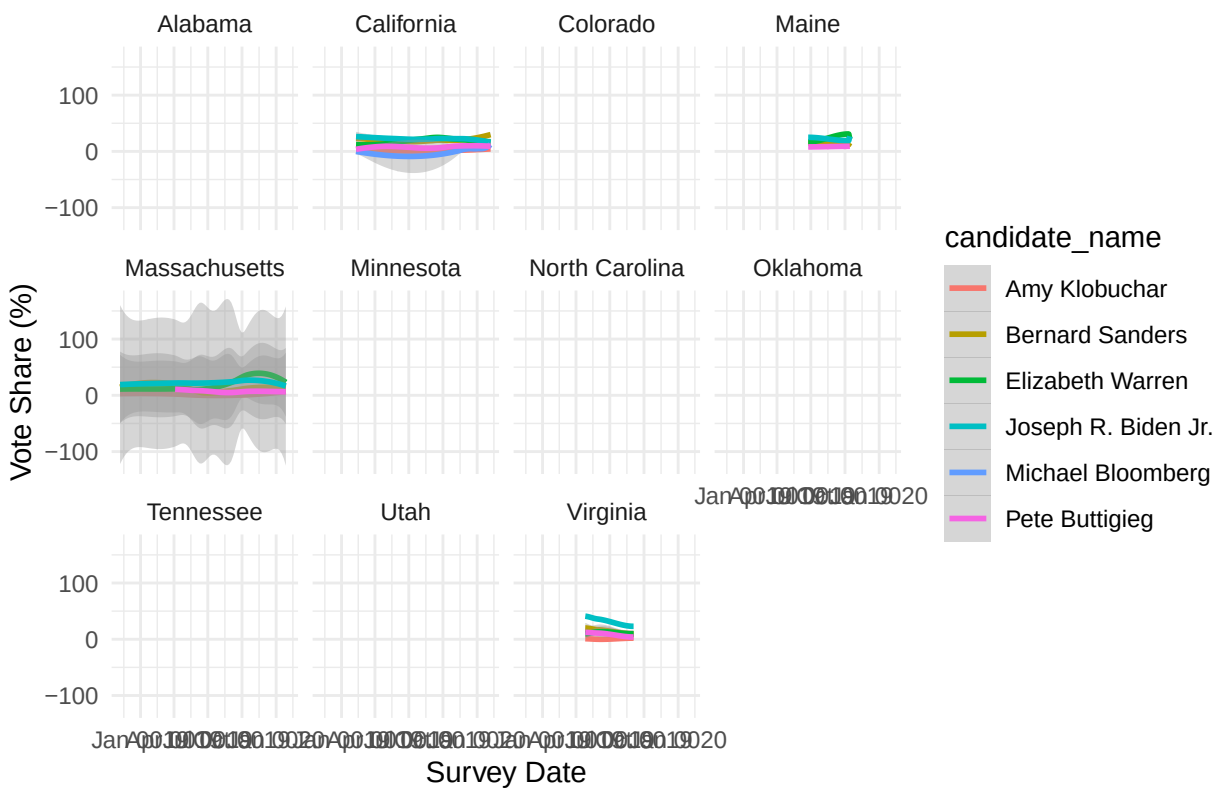
```
####  Let's load the dataset and clean the data:
pP <- primaryPolls <- read.csv('https://jmontgomery.github.io/PDS/Datasets/president_primary_polls_feb2
pP <- primaryPolls<-primaryPolls[primaryPolls$candidate_name%in%c("Amy Klobuchar", "Bernard Sanders", "
pP$start_date <- primaryPolls$start_date <-as.Date(primaryPolls$start_date, "%m/%d/%Y")


####  Now let's filter the states we would be doing forecast:
pP.f <- primaryPolls.forcasting<-primaryPolls[primaryPolls$state=="Alabama" | primaryPolls$state=="Arka
unique(primaryPolls.forcasting$state)
```

```
##  [1] "California"     "North Carolina" "Massachusetts"  "Utah"
##  [5] "Maine"          "Minnesota"      "Virginia"       "Colorado"
##  [9] "Oklahoma"       "Alabama"        "Tennessee"
```

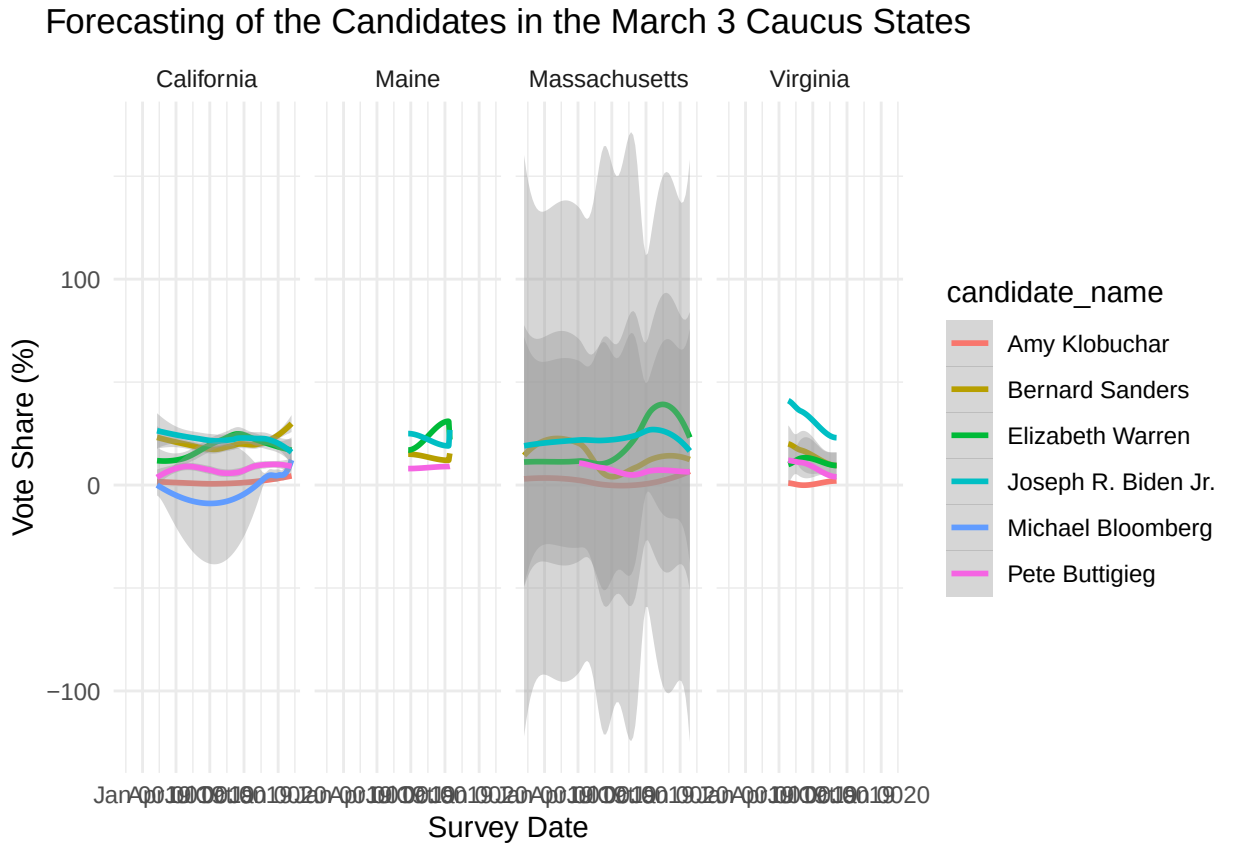
```
####  Now, let's forecast and change the theme to the minimial theme
library("tidyverse")

ggplot(data=pP.f)+
  geom_smooth(mapping = aes(x=start_date, y=pct, color=candidate_name)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  xlab("Survey Date") +
  ylab("Vote Share (%)") +
  theme(legend.position="bottom") +
  ggtitle("Forecasting of the Candidates in the March 3 Caucus States")
```




```
####  Also check the situation in different states

ggplot(data=pP.f)+
  geom_smooth(mapping = aes(x=start_date, y=pct, color=candidate_name)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  facet_wrap(~ state, nrow=3) +
  xlab("Survey Date") +
  ylab("Vote Share (%)") +
  ggtitle("Forecasting of the Candidates in the March 3 Caucus States")
```




```
####  Note that many of these states do not have variation in time. Now,
# let's look at the states with time variation:

pP.f2 <- pP.f %>%
  filter(state == "California" | state == "Maine" | state == "Massachusetts" | state == "Virginia" )

ggplot(data=pP.f2)+
  geom_smooth(mapping = aes(x=start_date, y=pct, color=candidate_name)) +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  facet_wrap(~ state, nrow=1) +
  xlab("Survey Date") +
  ylab("Vote Share (%)") +
  ggtitle("Forecasting of the Candidates in the March 3 Caucus States")
```



It can be seen thatin many of these states the popularity of Warren and Biden is in decline whereas in California, Massachusetts, and Virginia, we see an increasing trend for Bernie Sanders.

Finish the exercise we started in class on 2/13/2020:

```
####  Let's load the dataset and then clean it.
rm(list = ls())
pP <- primaryPolls <- read.csv('https://jmontgomery.github.io/PDS/Datasets/president_primary_polls_feb2
pP <- primaryPolls<-primaryPolls[primaryPolls$candidate_name%in%c("Amy Klobuchar", "Bernard Sanders", "
pP$start_date <- primaryPolls$start_date <-as.Date(primaryPolls$start_date, "%m/%d/%Y")


####  Let's re-organize the dataset so that there is only one row for each candidate-state dyad:
library(dplyr)
library(tidyr)
library(readr)

pP.v1 <- pP %>%
  filter(!((state) == "")) %>%
  select(candidate_name, state, pct, start_date)
object.size(pP.v1)
```

```
## 71056 bytes
```

```
pP.v2 <- pP.v1 %>%
  pivot_wider(names_from = start_date, values_from = pct)
```

```r
object.size(pP.v2)
```

```
## 528280 bytes
```

```r
print(paste("The size of new dataframe is", round(object.size(pP.v2)/object.size(pP.v1),2),
            "times greater than the previous one."))
```

```
## [1] "The size of new dataframe is 7.43 times greater than the previous one."
```

Q2 - tidyverse

Now you are going to combine two datasets in order to observe how many endorsements each candidate recieved using only dplyr functions. First, create two new objects polls and Endorsements:

```
rm(list = ls())
library(fivethirtyeight)
library(tidyverse)
polls <- read_csv('https://jmontgomery.github.io/PDS/Datasets/president_primary_polls_feb2020.csv')
Endorsements <- endorsements_2020
```

- Change the Endorsements variable name endorsee to candidate name
- Change the Endorsements dataframe into a tibble object.
- Filter the poll variable to only include the following 6 candidates: Amy Klobuchar, Bernard Sanders,Elizabeth Warren, Joseph R. Biden Jr., Michael Bloomberg, Pete Buttigieg and subset the dataset to the following five variables: *candidatename*, *samplesize*, *startdate*, *party*, *pct*
- Compare the candidate names in the two datasets and find instances where the a candidates name is spelled differently i.e. Bernard vs. Bernie. Using only dplyr functions, make these the same across datasets.
- Now combine the two datasets by candidate name using dplyr (there will only be five candidates after joining).
- Create a variable which indicates the number of endorsements for each of the five candidates using dplyr.
- Plot the number of endorsement each of the 5 candidates have. Save your plot as an object p.
- Run the following code: p + theme dark(). Notice how you can still customize your plot without rerunning the plot with new options. Save this plot in your forked repository
- Now, using the knowledge from the last step change the label of the X and Y axes to be more informative, add a title, and use your favorite theme. Save the plot in your forked repository.

A2 - ggplot2

Change the Endorsements variable name endorsee to candidate name

```
Endorsements <- Endorsements %>%
  rename(candidate_name = endorsee)
```

Change the Endorsements dataframe into a tibble object.

```
Endorsements <- as_tibble(Endorsements)
```

\subsection{Filter the poll variable to only include the following 6 candidates: Amy Klobuchar, Bernard Sanders,Elizabeth Warren, Joseph R. Biden Jr., Michael Bloomberg, Pete Buttigieg and subset the dataset to the following five variables: candidate_name, sample_size, start_date, party, pct}

```
polls.cand <- polls %>%
  filter(candidate_name == "Amy Klobuchar" | candidate_name == "Amy Klobuchar" | candidate_name ==  "Be
  select(candidate_name, sample_size, start_date, party, pct)
unique(polls.cand$candidate_name)
```

```
## [1] "Bernard Sanders"     "Pete Buttigieg"      "Joseph R. Biden Jr."
## [4] "Amy Klobuchar"       "Elizabeth Warren"    "Michael Bloomberg"
```

Compare the candidate names in the two datasets and find instances where the a candidates name is spelled differently i.e. Bernard vs. Bernie. Using only dplyr functions, make these the same across datasets.

```
unique(sort(Endorsements$candidate_name))
```

```
##  [1] "Amy Klobuchar"      "Bernie Sanders"     "Beto O'Rourke"
##  [4] "Cory Booker"        "Elizabeth Warren"   "Eric Swalwell"
##  [7] "Jay Inslee"         "Joe Biden"          "John Delaney"
## [10] "John Hickenlooper"  "Julian Castro"      "Kamala Harris"
## [13] "Kirsten Gillibrand" "Pete Buttigieg"     "Steve Bullock"
```

```
unique(sort(polls.cand$candidate_name))
```

```
## [1] "Amy Klobuchar"       "Bernard Sanders"     "Elizabeth Warren"
## [4] "Joseph R. Biden Jr." "Michael Bloomberg"   "Pete Buttigieg"
```

```
####  There are two differences,
# 1-  "Amy Klobuchar" vs "Amy Klobuchar"
# 2-  "Bernie Sanders" vs "Bernard Sanders"       <-----
# 3-  "Joe Biden" vs "Joseph R. Biden Jr."        <-----
# 4-  "Elizabeth Warren" vs "Elizabeth Warren"
# 5-  "Michael Bloomberg" vs nope
# 6-  "Pete Buttigieg" vs "Pete Buttigieg"

polls.cand <- polls.cand %>%
  mutate(candidate_name = str_replace(candidate_name, "Bernard Sanders", "Bernie Sanders"))

polls.cand <- polls.cand %>%
  mutate(candidate_name = str_replace(candidate_name, "Joseph R. Biden Jr.", "Joe Biden"))

unique(sort(Endorsements$candidate_name))
```

```
##  [1] "Amy Klobuchar"      "Bernie Sanders"     "Beto O'Rourke"
##  [4] "Cory Booker"        "Elizabeth Warren"   "Eric Swalwell"
##  [7] "Jay Inslee"         "Joe Biden"          "John Delaney"
## [10] "John Hickenlooper"  "Julian Castro"      "Kamala Harris"
## [13] "Kirsten Gillibrand" "Pete Buttigieg"     "Steve Bullock"
```

```
unique(sort(polls.cand$candidate_name))
```

```
## [1] "Amy Klobuchar"     "Bernie Sanders"    "Elizabeth Warren"
## [4] "Joe Biden"         "Michael Bloomberg" "Pete Buttigieg"
```

```
####  COOL!!
```

Now combine the two datasets by candidate name using dplyr (there will only be five candidates after joining).

```
polls.cand <- polls.cand %>%
  group_by(candidate_name) %>%
  mutate(pct.mean = mean(pct))

####  The question says there will be only 5 candidates after merging. Therefore, I assume
# that we are expected to use 'inner.join'.

inner.merged <- Endorsements %>%
  inner_join(polls.cand, by = 'candidate_name')

unique(inner.merged$candidate_name)
```

```
## [1] "Joe Biden"        "Bernie Sanders"   "Amy Klobuchar"    "Elizabeth Warren"
## [5] "Pete Buttigieg"
```

```
####  Yes, there are five candidates!!

####  However, note that this merging is not a good one as we merged 1000 observations
# into 5000 observations by only candidate names without no other criteria!!
```

Create a variable which indicates the number of endorsements for each of the five candidates using dplyr.

```
inner.merged <- inner.merged %>%
  group_by(candidate_name) %>%
  mutate(endorse.count = n()) %>%
  arrange(endorse.count)


unique(data.frame(inner.merged$candidate_name, inner.merged$endorse.count))
```

```
##       inner.merged.candidate_name inner.merged.endorse.count
## 1                   Pete Buttigieg                       4450
## 4451              Elizabeth Warren                       8667
## 13118                Amy Klobuchar                       8740
## 21858               Bernie Sanders                      15360
## 37218                    Joe Biden                      28159
```

Plot the number of endorsement each of the 5 candidates have. Save your plot as an object p.

```
p <- ggplot(data=inner.merged)+
  geom_point(mapping = aes(x = reorder(candidate_name, endorse.count), y = endorse.count, size = pct.mea
  xlab("Candidate Names") +
  scale_y_continuous(name = "Number of Endorsements", limits=c(0, 30000)) +
  labs( title = "Candidates by their Endorsements", size = "Mean Percentage Vote") +
  theme(legend.position="bottom")

p
```

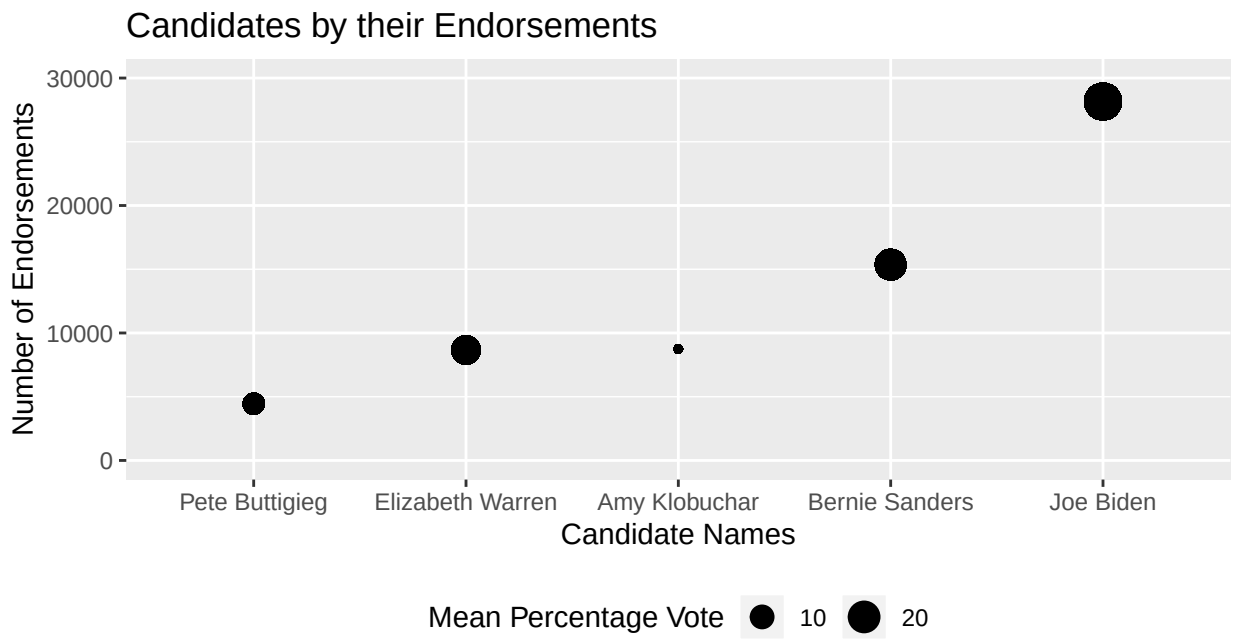


Figure 1: Candidates by Number of Endorsements - Default Theme

\subsection{ Run the following code: p + theme_dark(). Notice how you can still customize your plot without rerunning the plot with new options. Save this plot in your forked repository}

```
p.new <- p + theme_dark() + theme(legend.position="bottom")
p.new
```

```
ggsave("candidate_endorsements.pdf")
```

Now, using the knowledge from the last step change the label of the X and Y axes to be more informative, add a title, and use your favorite theme. Save the plot in your forked repository.

```
####  I have already done with changing labels.
####  Now, I am changing to another theme.

#install.packages("ggthemes")
```

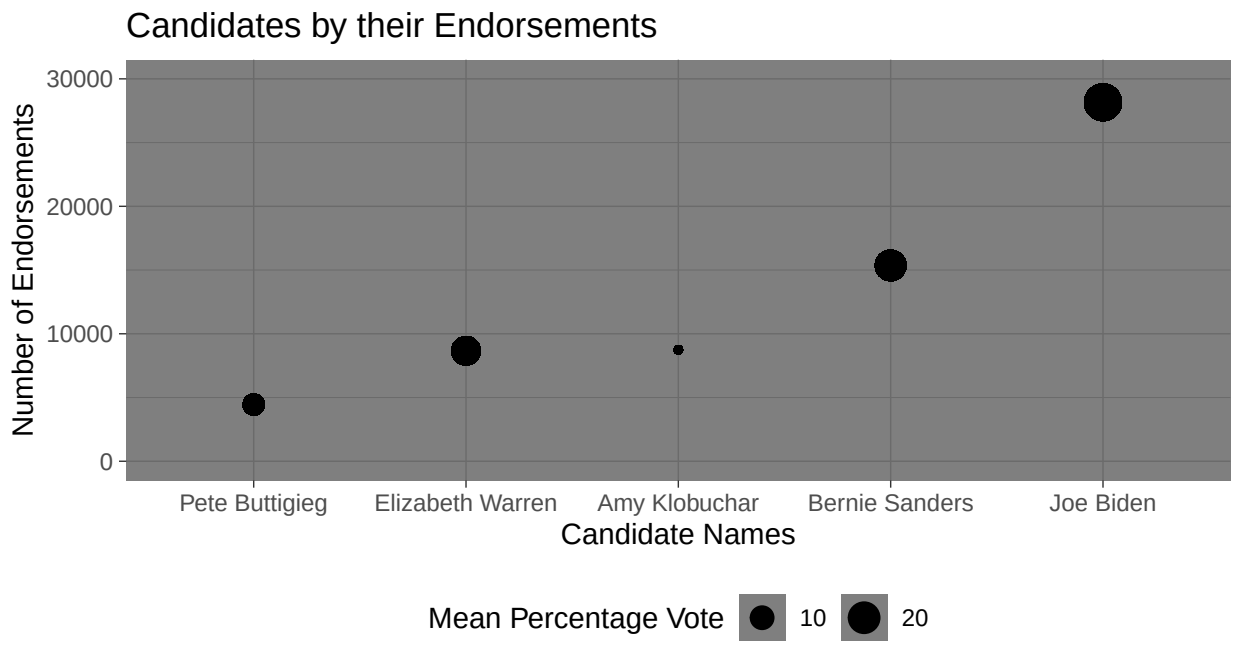
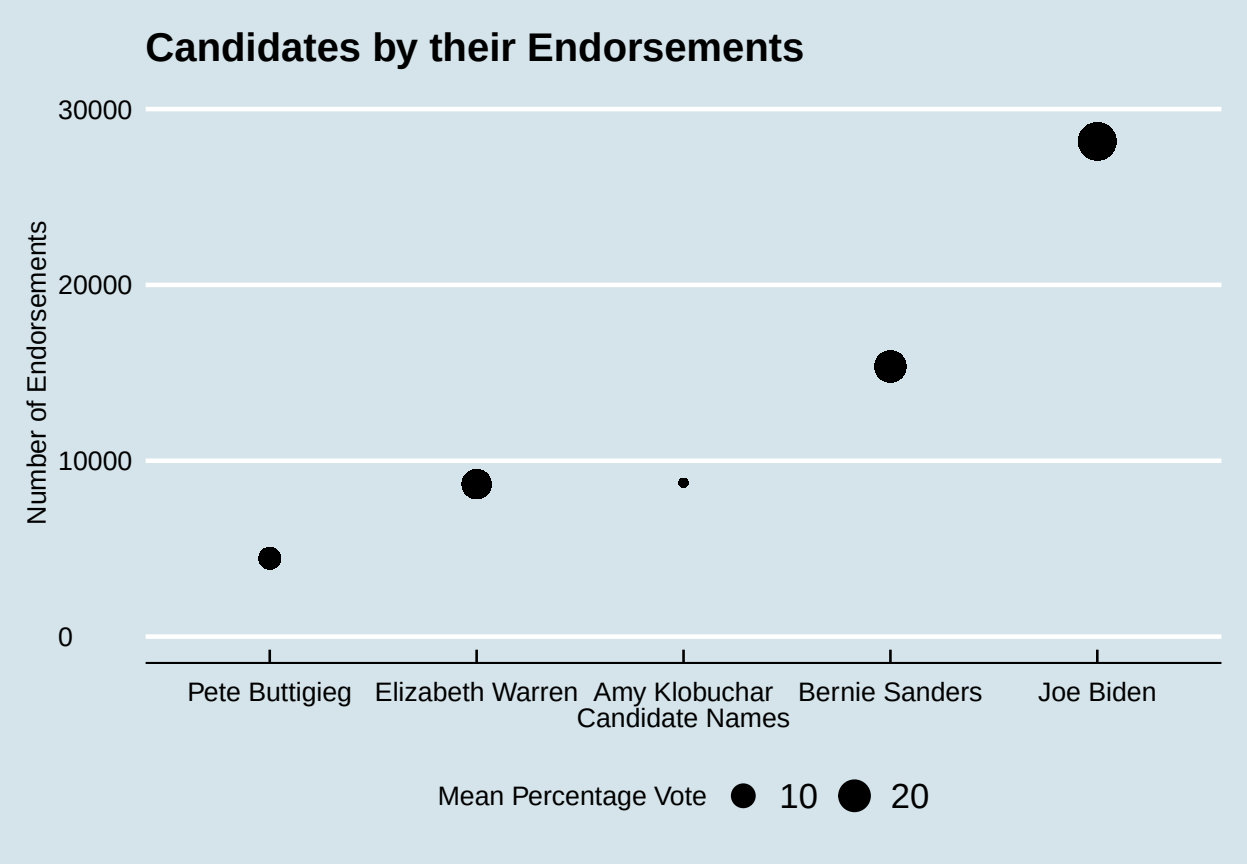


Figure 2: Candidates by Number of Endorsements - Dark Theme

```
library( ggthemes)
?theme_economist

p.new.1 <- p + theme_economist() + theme(legend.position="bottom")
p.new.1
```




```
ggsave("candidate_endorsements_my_fav.pdf")
```

Q3 - Text-as-Data

For this assignment you will be analyzing Tweets from President Trump for various characteristics.

```
rm(list = ls())
library(tidyverse)
#install.packages('tm')
library(tm)
#install.packages('lubridate')
library(lubridate)
#install.packages('wordcloud')
library(wordcloud)
tweets <- read_csv('https://politicaldatascience.com/PDS/Datasets/trump_tweets.csv')
```

A3 - Text-as-Data

\subsection{First separate the created_at variable into two new variables where the date and the time are in separate columns. Then report the range of dates that is in this dataset.}

Using dplyr subset the data to only include original tweets (remove retweets) and show the text of the President's top 5 most popular and most retweeted tweets. (Hint: The match function can help you find the index once you identify the largest values.)

Remove extraneous whitespace, remove numbers and punctuation, convert everything to lower case and remove the standard english stop words and include the following as stop words: c("see", "people",'new','want','one','even','must','need','done','ba 'know', 'can','said','like','many','like','realdonaldtrump').

Now create a wordcloud to visualize the top 50 words the President uses in his tweets. Use only words that occur at least three times. Save the plot into your forked repository.

Create a document term matrix called DTM that includes the argument control = list(weighting = weightTfIdf)

Finally, report the 50 words with the the highest tf.idf scores using a lower frequency bound of .8.