011852253 Assignment4

Sharath Kumar Karnati

2024-10-02

#QUESTION 1: Problem 1 (50 pts). This problem will involve the Lahman dataset (including the tables Batting, Teams, Salaries, and Managers). It is available in R by loading the Lahman library using the following command: library(Lahman) Alternatively, you can download the csv files from the Modules page on Canvas. The files are Batting.csv, Teams.csv, Salaries.csv, and Managers.csv. You can use Lahman_Desc.txt (also from Modules) to check the column descriptions for each dataset. We will first use joins to search and manipulate the dataset, then we will produce a flight count visualization.

```r
# Clear all objects from the environment
rm(list = ls())
```

```r
# Set a CRAN mirror
options(repos = c(CRAN = "https://cloud.r-project.org"))

# Install the Lahman package if not already installed
if (!requireNamespace("Lahman", quietly = TRUE)) {
  install.packages("Lahman")
}


install.packages("Lahman")
```

```
##
## The downloaded binary packages are in
##  /var/folders/yj/p15tz89x7yx1zf0jjs6dj5t80000gn/T//RtmpxYS9NL/downloaded_packages
```

```r
library(Lahman)
library(dplyr)

# Read the CSV files

batting <- read.csv("/Users/sharathkarnati/Desktop/DS/SK assignment 4/Batting.csv")
teams <- read.csv("/Users/sharathkarnati/Desktop/DS/SK assignment 4/Teams.csv")
salaries <- read.csv("/Users/sharathkarnati/Desktop/DS/SK assignment 4/Salaries.csv")
managers <- read.csv("/Users/sharathkarnati/Desktop/DS/SK assignment 4/Managers.csv")
```

```r
# View the Batting dataset
head(Batting)
```

```
##    playerID yearID stint teamID lgID  G AB R H X2B X3B HR RBI SB CS BB SO IBB
## 1 aardsda01   2004     1    SFN   NL 11  0 0 0   0   0  0   0  0  0  0  0   0
## 2 aardsda01   2006     1    CHN   NL 45  2 0 0   0   0  0   0  0  0  0  0   0
## 3 aardsda01   2007     1    CHA   AL 25  0 0 0   0   0  0   0  0  0  0  0   0
## 4 aardsda01   2008     1    BOS   AL 47  1 0 0   0   0  0   0  0  0  0  1   0
## 5 aardsda01   2009     1    SEA   AL 73  0 0 0   0   0  0   0  0  0  0  0   0
## 6 aardsda01   2010     1    SEA   AL 53  0 0 0   0   0  0   0  0  0  0  0   0
##   HBP SH SF GIDP
## 1   0  0  0    0
## 2   0  1  0    0
## 3   0  0  0    0
## 4   0  0  0    0
## 5   0  0  0    0
## 6   0  0  0    0
```

```r
# View the Teams dataset
head(Teams)
```

```
##   yearID lgID teamID franchID divID Rank  G Ghome  W  L DivWin WCWin LgWin
## 1   1871   NA    BS1      BNA  <NA>    3 31    NA 20 10   <NA>  <NA>     N
## 2   1871   NA    CH1      CNA  <NA>    2 28    NA 19  9   <NA>  <NA>     N
## 3   1871   NA    CL1      CFC  <NA>    8 29    NA 10 19   <NA>  <NA>     N
## 4   1871   NA    FW1      KEK  <NA>    7 19    NA  7 12   <NA>  <NA>     N
## 5   1871   NA    NY2      NNA  <NA>    5 33    NA 16 17   <NA>  <NA>     N
## 6   1871   NA    PH1      PNA  <NA>    1 28    NA 21  7   <NA>  <NA>     Y
##   WSWin   R   AB   H X2B X3B HR BB SO SB CS HBP SF  RA  ER  ERA CG SHO SV
## 1  <NA> 401 1372 426  70  37  3 60 19 73 16  NA NA 303 109 3.55 22   1  3
## 2  <NA> 302 1196 323  52  21 10 60 22 69 21  NA NA 241  77 2.76 25   0  1
## 3  <NA> 249 1186 328  35  40  7 26 25 18  8  NA NA 341 116 4.11 23   0  0
## 4  <NA> 137  746 178  19   8  2 33  9 16  4  NA NA 243  97 5.17 19   1  0
## 5  <NA> 302 1404 403  43  21  1 33 15 46 15  NA NA 313 121 3.72 32   1  0
## 6  <NA> 376 1281 410  66  27  9 46 23 56 12  NA NA 266 137 4.95 27   0  0
##   IPouts  HA HRA BBA SOA   E DP    FP                    name
## 1    828 367   2  42  23 243 24 0.834    Boston Red Stockings
## 2    753 308   6  28  22 229 16 0.829 Chicago White Stockings
## 3    762 346  13  53  34 234 15 0.818  Cleveland Forest Citys
## 4    507 261   5  21  17 163  8 0.803    Fort Wayne Kekiongas
## 5    879 373   7  42  22 235 14 0.840        New York Mutuals
## 6    747 329   3  53  16 194 13 0.845  Philadelphia Athletics
##                           park attendance BPF PPF teamIDBR teamIDlahman45
## 1          South End Grounds I         NA 103  98      BOS            BS1
## 2      Union Base-Ball Grounds         NA 104 102      CHI            CH1
## 3 National Association Grounds         NA  96 100      CLE            CL1
## 4               Hamilton Field         NA 101 107      KEK            FW1
## 5     Union Grounds (Brooklyn)         NA  90  88      NYU            NY2
## 6     Jefferson Street Grounds         NA 102  98      ATH            PH1
##   teamIDretro
## 1         BS1
## 2         CH1
## 3         CL1
## 4         FW1
## 5         NY2
## 6         PH1
```

```r
# View the Salaries dataset
head(Salaries)
```

```
##   yearID teamID lgID  playerID salary
## 1   1985    ATL   NL barkele01 870000
## 2   1985    ATL   NL bedrost01 550000
## 3   1985    ATL   NL benedbr01 545000
## 4   1985    ATL   NL  campri01 633333
## 5   1985    ATL   NL ceronri01 625000
## 6   1985    ATL   NL chambch01 800000
```

```r
# View the Managers dataset
head(Managers)
```

```
##    playerID yearID teamID lgID inseason  G  W  L rank plyrMgr
## 1 wrighha01   1871    BS1   NA        1 31 20 10    3       Y
## 2  woodji01   1871    CH1   NA        1 28 19  9    2       Y
## 3 paborch01   1871    CL1   NA        1 29 10 19    8       Y
## 4 lennobi01   1871    FW1   NA        1 14  5  9    8       Y
## 5 deaneha01   1871    FW1   NA        2  5  2  3    8       Y
## 6 fergubo01   1871    NY2   NA        1 33 16 17    5       Y
```

#QUESTION 1.a: a) (10 pts) Filter the dataset (using a left join) to display the playerID , yearID , teamID , stint , G (games played), HR (home runs), and salary for all players who hit more than 30 home runs in a single season and played for a team in New York (teamID "NYA" or "NYN") between 2010 and 2020. How many players match these criteria?

```r
# Load necessary libraries


# Filter the Batting dataset for players who hit more than 30 home runs between 2010 and 2020
batting_filtered <- Batting %>%
  filter(HR > 30, yearID >= 2010, yearID <= 2020)

# Filter for New York teams
batting_ny <- batting_filtered %>%
  filter(teamID %in% c("NYA", "NYN"))

# Perform a left join with the Salaries dataset to include salary information
batting_with_salary <- batting_ny %>%
  left_join(Salaries, by = c("playerID", "yearID", "teamID"))

# Select the relevant columns
result <- batting_with_salary %>%
  select(playerID, yearID, teamID, stint, G, HR, salary)

# View the result
print(result)
```

```
##     playerID yearID teamID stint   G HR   salary
## 1  alonspe01   2019    NYN     1 161 53       NA
## 2   canoro01   2012    NYA     1 161 33 14000000
## 3  cespeyo01   2016    NYN     1 132 31 27328046
## 4  confomi01   2019    NYN     1 151 33       NA
## 5  davisik02   2012    NYN     1 156 32   506690
## 6  grandcu01   2011    NYA     1 156 41  8250000
## 7  grandcu01   2012    NYA     1 160 43 10000000
## 8  judgeaa01   2017    NYA     1 155 52       NA
## 9  rodrial01   2015    NYA     1 151 33 22000000
## 10 sanchga02   2017    NYA     1 122 33       NA
## 11 sanchga02   2019    NYA     1 106 34       NA
## 12 stantmi03   2018    NYA     1 158 38       NA
## 13 teixema01   2010    NYA     1 158 33 20625000
## 14 teixema01   2011    NYA     1 156 39 23125000
## 15 teixema01   2015    NYA     1 111 31 23125000
## 16 torregl01   2019    NYA     1 144 38       NA
```

```
# Count the number of players who match the criteria
num_players <- nrow(result)
cat("Number of players who hit more than 30 home runs and played ,\nfor a New York team between 2010 and
```

```
## Number of players who hit more than 30 home runs and played ,
## for a New York team between 2010 and 2020: 16
```

#QUESTION 1.b: b) (10 pts) What is the difference between the following two joins? Do not show the result of these anti_joins in your submission. anti_join(Salaries, Batting, by = c("playerID" = "playerID")) anti_join(Batting, Salaries, by = c("playerID" = "playerID")) What is the difference between semi_join and anti_join? Provide an example using the Salaries and Batting tables.

#ANSWER:

#Difference Between the Two anti_joins:

#anti_join(Salaries, Batting, by = c("playerID" = "playerID")):

All rows from the Salaries table for which the playerID does not have a matching entry in the Batting table will be returned as a result. Essentially, it lists all players in Salaries who are not in Bating.

#anti_join(Batting, Salaries, by = c("playerID" = "playerID")): All rows from the Batting table for which the playerID does not have a matching entry in the Salaries table will be returned as a result. Every player in batting who isn't on the salary list is displayed.

#Difference Between semi_join and anti_join:

#semi_join():

Returns every entry from the first table (salaries, for example) where the corresponding row exists in the second table (batting, for example).

The rows that match the secondd table are the only ones that are returned from the first table.

It does not return columns from the second table, in contrast to inner_join().

#anti_join(): Returns every entry from the first table (salaries, for example) in cases when there are no matching rows in the second table (batting, for example).

It displays the gaps in the second table and is the inverse of semi_join(). Using semi_join and anti_join as examples:

#Here is an example made with the Batting and Salaries tables:

```
semi_result <- semi_join(Salaries, Batting, by = "playerID")
head(semi_result)
```

```
##   yearID teamID lgID  playerID salary
## 1   1985    ATL   NL barkele01 870000
## 2   1985    ATL   NL bedrost01 550000
## 3   1985    ATL   NL benedbr01 545000
## 4   1985    ATL   NL  campri01 633333
## 5   1985    ATL   NL ceronri01 625000
## 6   1985    ATL   NL chambch01 800000
```

```
anti_result <- anti_join(Salaries, Batting, by = "playerID")
head(anti_result)
```

#Explanation: * When you wish to retain rows that have a match in a different table, you use semi_join. * When you wish to retain rows that don't match in another table, you use anti_join.

#QUESTION 1.c: c) (10 pts) Select the teamID , yearID , and the total number of runs batted in (RBI) for each team in the American League (AL) for the year 2015 (using one or more inner joins with the Teams and Batting tables). How many total home runs were hit by American League teams in 2015?

#ANSWER:

```
# Select total RBI for each team in American League for 2015
al_rbi_2015 <- Teams %>%
  filter(lgID == "AL", yearID == 2015) %>%
  inner_join(Batting, by = c("teamID", "yearID")) %>%
  group_by(teamID, yearID) %>%
  summarise(total_RBI = sum(RBI, na.rm = TRUE))

# Print the total RBI for each team in a nicely formatted table
cat("Total Runs Batted In (RBI) for American League Teams in 2015:\n")
```

```
## Total Runs Batted In (RBI) for American League Teams in 2015:
```

```
print(al_rbi_2015)
```

```
## # A tibble: 15 x 3
## # Groups:   teamID [15]
##    teamID yearID total_RBI
##    <fct>   <int>     <int>
##  1 BAL      2015       686
##  2 BOS      2015       706
##  3 CHA      2015       595
##  4 CLE      2015       640
##  5 DET      2015       660
##  6 HOU      2015       691
##  7 KCA      2015       689
##  8 LAA      2015       621
##  9 MIN      2015       661
## 10 NYA      2015       737
## 11 OAK      2015       661
## 12 SEA      2015       624
## 13 TBA      2015       612
## 14 TEX      2015       707
## 15 TOR      2015       852
```

```
# Calculate total home runs hit by American League teams in 2015
total_home_runs_al_2015 <- Batting %>%
  filter(yearID == 2015, teamID %in% al_rbi_2015$teamID) %>%
  summarise(total_HR = sum(HR, na.rm = TRUE))
# Print the total home runs with a clear statement
cat("\nTotal Home Runs hit by American League teams in 2015:", total_home_runs_al_2015$total_HR, "\n")
```

```
##
## Total Home Runs hit by American League teams in 2015: 2634
```

#QUESTION 1.d:

d) (10 pts) Using the Managers and Teams tables, determine the number of seasons each manager managed a team. Use group_by and count to get the number of unique managerID and teamID combinations. How many unique combinations of managerID and teamID are present? Are there any players with unusually high number of years as a manager? 2

#ANSWER:

```r
str(Managers)
```

```
## 'data.frame':    3749 obs. of  10 variables:
##  $ playerID: chr  "wrighha01" "woodji01" "paborch01" "lennobi01" ...
##  $ yearID  : int  1871 1871 1871 1871 1871 1871 1871 1871 1871 1871 ...
##  $ teamID  : Factor w/ 149 levels "ALT","ANA","ARI",..: 24 31 39 56 56 90 97 111 136 136 ...
##  $ lgID    : Factor w/ 7 levels "AA","AL","FL",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ inseason: int  1 1 1 1 2 1 1 1 1 2 ...
##  $ G       : int  31 28 29 14 5 33 28 25 4 25 ...
##  $ W       : int  20 19 10 5 2 16 21 4 1 12 ...
##  $ L       : int  10 9 19 9 3 17 7 21 3 12 ...
##  $ rank    : int  3 2 8 8 8 5 1 9 6 6 ...
##  $ plyrMgr : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
```

```r
# Count the number of seasons each manager managed a team
manager_seasons <- Managers %>%
  group_by(playerID, teamID) %>%
  summarise(seasons_managed = n_distinct(yearID)) %>%
  ungroup() %>%
  arrange(desc(seasons_managed))

# Display the sorted results
cat("Number of seasons each manager managed a team (sorted):\n")
```

```
## Number of seasons each manager managed a team (sorted):
```

```r
print(manager_seasons)
```

```
## # A tibble: 1,295 x 3
##    playerID  teamID seasons_managed
##    <chr>     <fct>            <int>
##  1 mackco01  PHA                 50
##  2 mcgrajo01 NY1                 31
##  3 coxbo01   ATL                 25
##  4 lasorto01 LAN                 21
##  5 alstowa01 LAN                 19
##  6 ansonca01 CHN                 19
##  7 harribu01 WS1                 18
##  8 robinwi01 BRO                 18
##  9 andersp01 DET                 17
## 10 weaveea99 BAL                 17
## # i 1,285 more rows
```

```r
# Count the number of unique combinations of managerID and teamID
unique_combinations <- nrow(manager_seasons)
cat("\nNumber of unique combinations of managerID and teamID:", unique_combinations, "\n")
```

```
##
## Number of unique combinations of managerID and teamID: 1295
```

```r
# Check for managers with unusually high number of years as a manager
unusually_high_managers <- manager_seasons %>%
  filter(seasons_managed > 10) %>%  # Example threshold for unusually high
  arrange(desc(seasons_managed))  # Sort in descending order

# Display the managers with unusually high number of seasons
```

```r
cat("\nManagers with unusually high number of seasons managed (sorted):\n")
```

```
##
## Managers with unusually high number of seasons managed (sorted):
```

```r
print(unusually_high_managers)
```

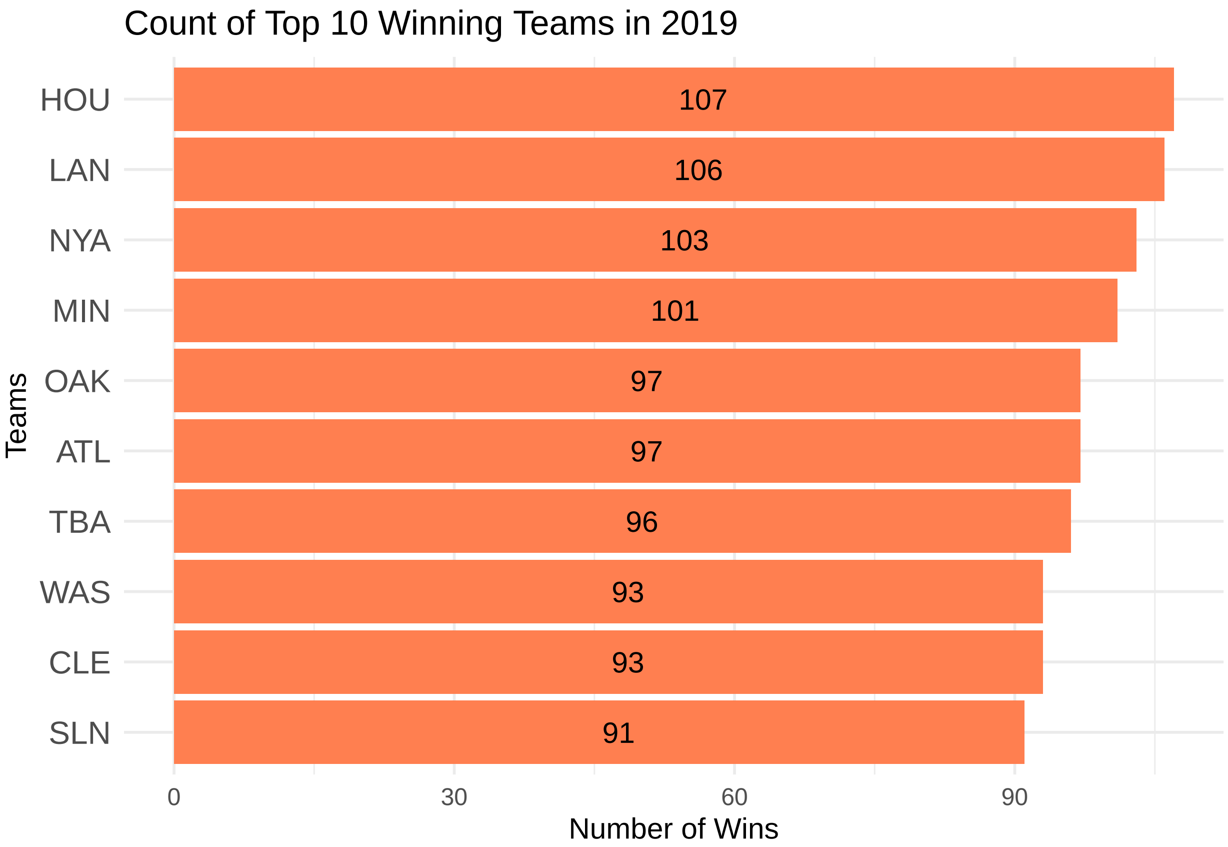
```
## # A tibble: 37 x 3
##    playerID  teamID seasons_managed
##    <chr>     <fct>            <int>
##  1 mackco01  PHA                 50
##  2 mcgrajo01 NY1                 31
##  3 coxbo01   ATL                 25
##  4 lasorto01 LAN                 21
##  5 alstowa01 LAN                 19
##  6 ansonca01 CHN                 19
##  7 harribu01 WS1                 18
##  8 robinwi01 BRO                 18
##  9 andersp01 DET                 17
## 10 weaveea99 BAL                 17
## # i 27 more rows
```

#QUESTION 1.e: e) (10 pts) Using the provided template as a start, produce a horizontal bar plot that shows the number of wins for the top 10 teams in 2019. Adjust the axis labels to clearly represent the teams and the number of wins. Add a meaningful title to the plot, and include the number of wins as text on each bar for clarity.

Teams %>% filter(yearID == 2019) %>% select(teamID, W) %>% ggplot(aes(x = reorder(teamID, W), y = W)) + geom_bar(stat = "identity", fill = "steelblue") + coord_flip()

#ANSWER:

```r
library(ggplot2)


# Create the horizontal bar plot for top 10 teams in 2019
top_teams_2019 <- Teams %>%
  filter(yearID == 2019) %>%
  select(teamID, W) %>%
  arrange(desc(W)) %>%           #
  head(10)

# Create the horizontal bar plot
ggplot(top_teams_2019, aes(x = reorder(teamID, W), y = W)) +
  geom_bar(stat = "identity", fill = "coral") +
  coord_flip() +
  geom_text(aes(label = W),
            position = position_stack(vjust = 0.5),
            hjust = -0.1) +
  labs(title = "Count of Top 10 Winning Teams in 2019",
       x = "Teams",
       y = "Number of Wins") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 12))
```



#QUESTION 2:

Problem 2 (30 pts). The goal of this problem to create a visualization of the US map showing the states/territories and the number of presidential votes received during an election year. For this task, you will work with the us-presidents.csv dataset. The dataset can be found on the Modules page on Canvas. There dataset consists of 612 observations of 4 variables: year, state, state_po, office, totalvotes. For this question, you will create two visualizations of the US map for two presidential years of your choice coloring the states or sizing the point/marker for the states according to the number of total votes received from that state for the presidential election. Compare both maps and comment on any observations. You are free to choose any mapping tool you wish to produce this visualization. Try to make your visualization as nice looking as possible. You can use the state column directly to visualize the observations or you could get the coordinates for each state (depending on the tool and your visualization). Research how this can be done and use what you find. The dataplusscience.com website has some blogs about mapping that you may find useful. After you have coordinates you can use different methods for mapping. You can use packages available in R or Python. Another simple method is probably through https://batchgeo.com/features/map-coordinates/ . However, you can also use d3 to map the locations, if you want to learn something that you could use for other projects later.

#ANSWER:

```
install.packages("maps")
```

```
##
## The downloaded binary packages are in
##  /var/folders/yj/p15tz89x7yx1zf0jjs6dj5t80000gn/T//RtmpxYS9NL/downloaded_packages
```

```
install.packages("usmap")
```

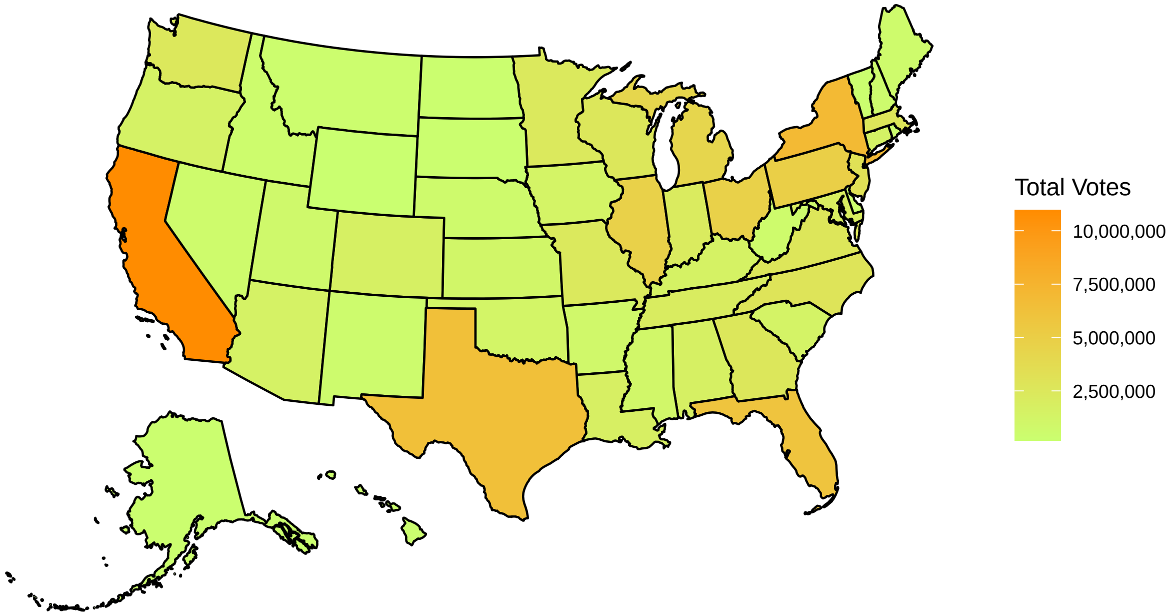
```
##
## The downloaded binary packages are in
##  /var/folders/yj/p15tz89x7yx1zf0jjs6dj5t80000gn/T//RtmpxYS9NL/downloaded_packages
```

```
# Load necessary libraries
library(usmap)
library(dplyr)
library(ggplot2)
library(scales)  # Load the scales package

# Load the dataset
us_presidents <- read.csv("/Users/sharathkarnati/Desktop/DS/SK assignment 4/us-presidents.csv")

# Get unique years and sort them
unique_years <- unique(us_presidents$year)
sorted_years <- sort(unique_years)

# Select the first and last years
first_year <- sorted_years[7]
last_year <- sorted_years[9]
print(paste("Selected years:", first_year, "and", last_year))
```

```
## [1] "Selected years: 2000 and 2008"
```

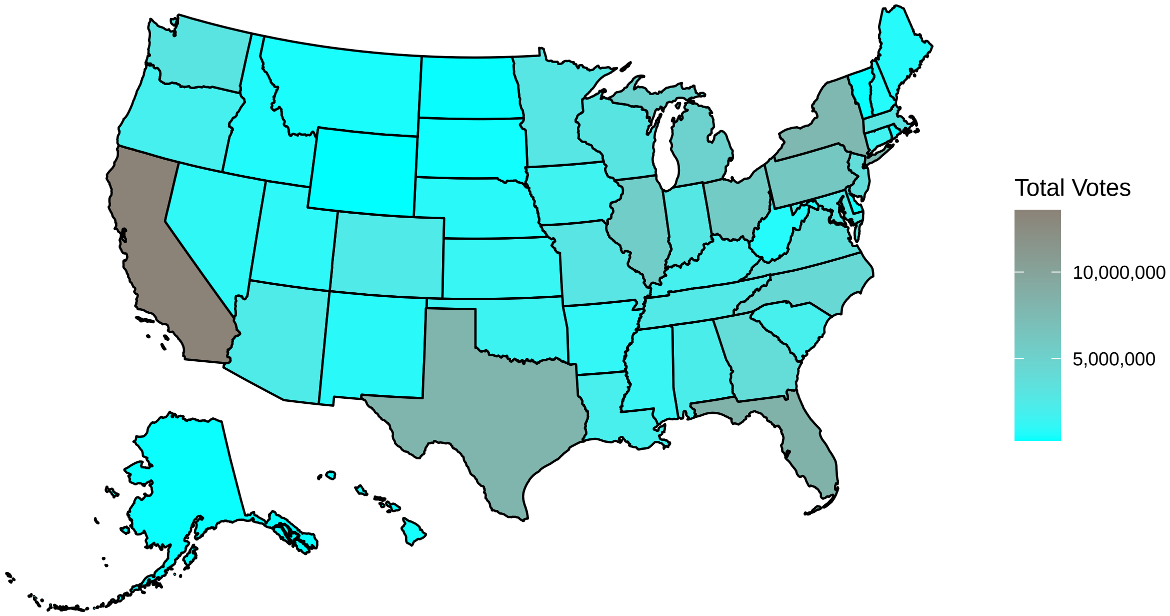
```
# Summarize total votes by state and year
votes_data <- us_presidents %>%
  filter(year %in% c(first_year, last_year)) %>%
  group_by(state, year) %>%
  summarize(total_votes = sum(totalvotes), .groups = 'drop')


# Create a map for the first year
map_first_year <- plot_usmap(data = votes_data %>% filter(year == first_year),
                             values = "total_votes",
                             color = "black") +
  scale_fill_continuous(name = "Total Votes",
                        low = "darkolivegreen1",
                        high = "darkorange",
                        labels = scales::comma) +  # Format labels with commas
  theme(legend.position = "right") +
  labs(title = paste("Total Presidential Votes in", first_year))

# Create a map for the last year
map_last_year <- plot_usmap(data = votes_data %>% filter(year == last_year),
                            values = "total_votes",
                            color = "black") +
  scale_fill_continuous(name = "Total Votes",
                        low = "cyan",
                        high = "antiquewhite4",
                        labels = scales::comma) +  # Format labels with commas
  theme(legend.position = "right") +
  labs(title = paste("Total Presidential Votes in", last_year))


# Print the maps
print(map_first_year)
```

Total Presidential Votes in 2000



```r
print(map_last_year)
```

Total Presidential Votes in 2008



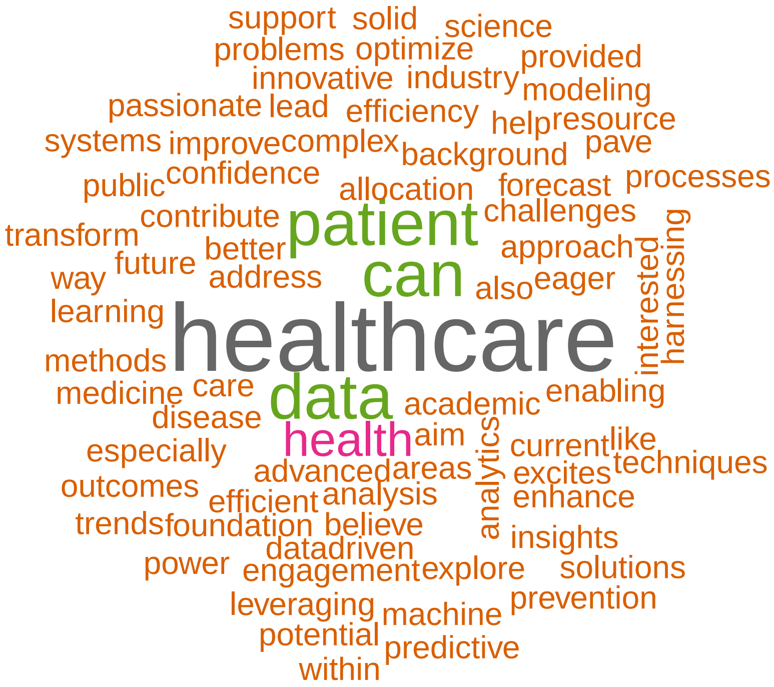
#Explanation:

Here, we have created maps for two years(i.e 2000 and 2008) of presidential votes. Here, the map in green colour resembels presidential votes in 2000 year and at the side we can see the colour map for voting ranges, and the map in blue colour resembles presidential votes in 2008 year and with a colour map of voting ranges at the side. I represented both maps using different colours.We can observe that the votes are evenly spread across the state and the dark coloured regions are where we got most of the votes from(which is around California region) in both maps. As the density of the colour decreases which refers to the voting low percent in that state. As the colour density increased the state voting percent also increased.

#QUESTION 3: Problem 3 (20 pts). Create a word cloud for an interesting (relatively short, say a couple of pages) document of your own choice. Examples of suitable documents include: summary of a recent project you are working or have worked on; your own recent Statement of Purpose or Research Statement or some other similar document. You can create the word clouds in R using the package called wordcloud or you can use another tool outside of R such as Wordle. If you do this in R, you will first need to install wordcloud (using install.packages("wordcloud")) and then load it (using library(wordcloud)). Then look up the documentation for the function called wordcloud in the package with the same name to create your cloud. Note that this function takes many arguments, but you would be mostly fine with the default settings. Only providing the text of your words may suffice for a minimalist purpose. 3 You are welcome (and encouraged) to take the generated word cloud and manipulate it using another software to enhance its aesthetic. If you have used Wordle instead of R, Wordle gives you functionalities to play with the look of the word cloud you get. Experiment till you get something you like most. Your submission for this would include the figure (cloud) and a brief caption that describes the text for the cloud. For example, it could be something like "Jenneth Joe's Essay on Life During Pandemic, written in June 2021."

#ANSWER:

```
# Load necessary libraries
library(dplyr)
library(tm)
library(wordcloud)
library(RColorBrewer)

# Sample text (your statement)
text <- "I am passionate about data science and its applications in healthcare. I believe that leveragin

# Create a text corpus
corpus <- Corpus(VectorSource(text))

# Preprocess the text
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeWords, stopwords("en"))

# Create a term-document matrix
tdm <- TermDocumentMatrix(corpus)
m <- as.matrix(tdm)
word_freqs <- sort(rowSums(m), decreasing = TRUE)
word_freqs_df <- data.frame(word = names(word_freqs), freq = word_freqs)

# Filter out long words (greater than a certain number of characters)
word_freqs_df <- word_freqs_df %>%
  filter(nchar(word) <= 10)  # Adjust the length limit as needed

# Create the word cloud
set.seed(1234)  # For reproducibility
wordcloud(words = word_freqs_df$word,
          freq = word_freqs_df$freq,
          min.freq = 1,
          max.words = 100,
          random.order = FALSE,
          rot.per = 0.1,
          colors = brewer.pal(8, "Dark2"),
          scale = c(3, 0.5))  # Adjust scale for better fitting
```



```r
# Caption for the word cloud
caption <- "Application of Data Science in Health Care."
caption <- paste0(caption)

cat(caption)
```

```
## Application of Data Science in Health Care.
```