Assignment 4

Athul Jose P
11867566

School of Electrical Engineering and Computer Science

Washington State University

CptS 575 Data Science

1.a.

```r
# loading packages
library(dplyr)
library(Lahman)
library(ggplot2)

# batting data: HR > 30
batting_filtered <- Batting %>%
  filter(HR > 30)

# teams data: criteria
ny_teams <- Teams %>%
  filter(teamID %in% c("NYA", "NYN") & yearID >= 2010 & yearID <= 2020)

# join operations
final_filtered_data <- batting_filtered %>%
  left_join(Salaries, by = c("playerID", "yearID", "teamID")) %>%
  inner_join(ny_teams, by = c("teamID", "yearID")) %>%
  select(playerID, yearID, teamID, stint, G.x, HR.x, salary)

# print result
final_filtered_data
```

```
    playerID yearID teamID stint G.x HR.x   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
```

```r
# number of distinct players
n_distict_players <- final_filtered_data %>%
  distinct(playerID) %>%
  nrow()

print(paste("Players matches the criteria:", n_distict_players))
```

```
[1] "Players matches the criteria: 12"
```

1.b.

Difference between the two anti_joins:

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

   This operation will return all rows from the Salaries table where the playerID does not exist in the Batting table. It is essentially asking: "Which players in the Salaries table do not have a corresponding entry in the Batting table?"

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

   This operation will return all rows from the Batting table where the playerID does not exist in the Salaries table.It is asking: "Which players in the Batting table do not have a corresponding entry in the Salaries table?"

Difference between semi_join and anti_join:

semi_join:

Returns all rows from the left table where there is a match in the right table. It only keeps the columns from the left table. In other words, it selects rows from the left table that have a corresponding entry in the right table.

anti_join:

Retrieves all rows from the left table that don't have a matching entry in the right table, while retaining only the columns from the left table. Essentially, it selects rows from the left table that lack a corresponding match in the right table.

semi_join Example:

To find all the players in the Salaries table who have a corresponding record in the Batting table:

```r
semi_result <- semi_join(Salaries, Batting, by = c("playerID" = "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_join Example:

To find all the players in the Salaries table who do not have a corresponding record in the Batting table:

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

```
[1] yearID   teamID   lgID     playerID salary
<0 rows> (or 0-length row.names)
```

1.c.

```r
# filter teams
teams_2015 <- Teams %>%
  filter(lgID == "AL", yearID == 2015) %>%
  select(teamID, yearID, HR)

# filter batting
batting_2015 <- Batting %>%
  filter(yearID == 2015) %>%
  select(teamID, yearID, RBI)

# join operation
joined_data_2015 <- inner_join(
  teams_2015,
  batting_2015,
  by = c("teamID", "yearID")
)
```

```
# summary
hr_summary <- joined_data_2015 %>%
  group_by(teamID, yearID) %>%
  summarise(
    total_HR = sum(HR, na.rm = TRUE),
    .groups = 'drop'
  )

# print result
n_HR <- sum(hr_summary$total_HR, na.rm = TRUE)
print(paste("Total Home Runs:", n_HR))
```

```
[1] "Total Home Runs: 130695"
```

1.d.

```
# join managers and teams
combined_df <- inner_join(Managers, Teams, by = c("teamID", "yearID"))

combined_count <- combined_df %>%
  group_by(playerID, teamID) %>%
  summarise(num_seasons = n(),
  .groups = 'drop') %>%
  arrange(desc(num_seasons))

n_combination <- nrow(combined_count)

print(paste("Number of unique combinations:", n_combination))
```

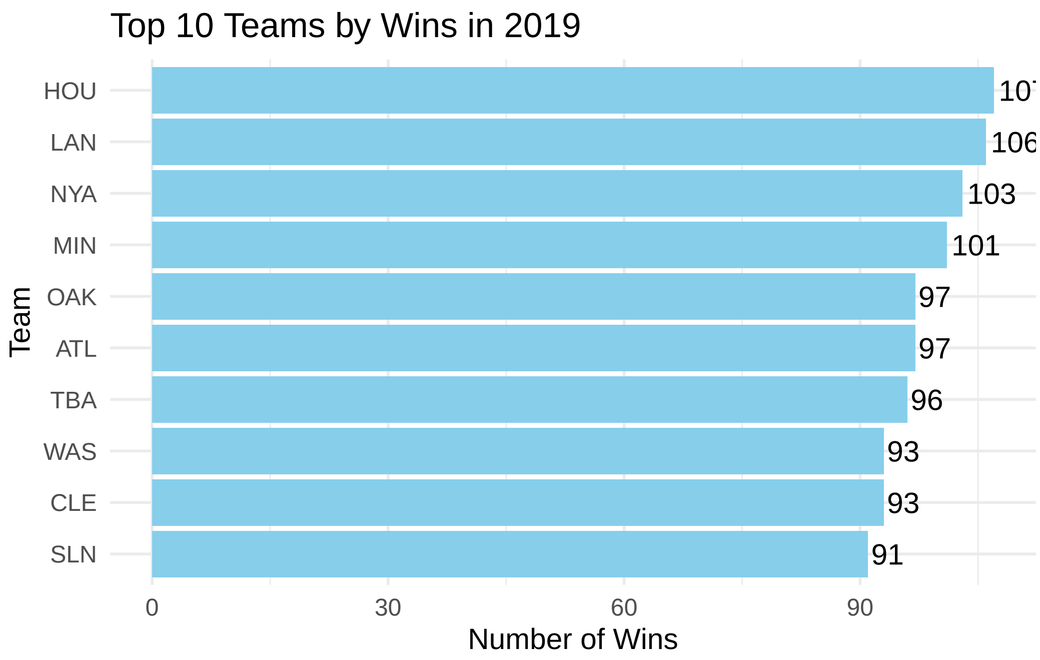
```
[1] "Number of unique combinations: 1295"
```

```
long_tenure_managers <- combined_count %>%
  filter(num_seasons > 20)

head(long_tenure_managers)
```

```
# A tibble: 4 x 3
  playerID  teamID num_seasons
  <chr>     <fct>        <int>
1 mackco01  PHA             50
2 mcgrajo01 NY1             33
3 coxbo01   ATL             25
4 lasorto01 LAN             21
```

1.e.

```
Teams %>%
  filter(yearID == 2019) %>%
  select(teamID, W) %>%
  arrange(desc(W)) %>%
  top_n(10, W) %>%
  ggplot(aes(x = reorder(teamID, W), y = W)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +
  geom_text(aes(label = W), hjust = -0.1) +
  labs(title = "Top 10 Teams by Wins in 2019",
       x = "Team",
       y = "Number of Wins") +
  theme_minimal()
```



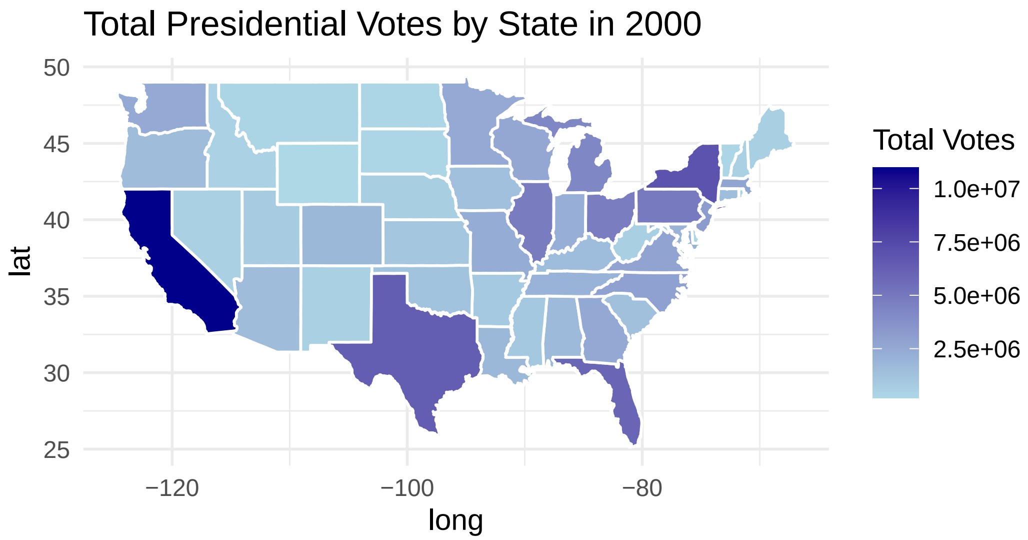
2.a.

```r
# loading libraries
library(ggplot2)
library(dplyr)
library(maps)
```

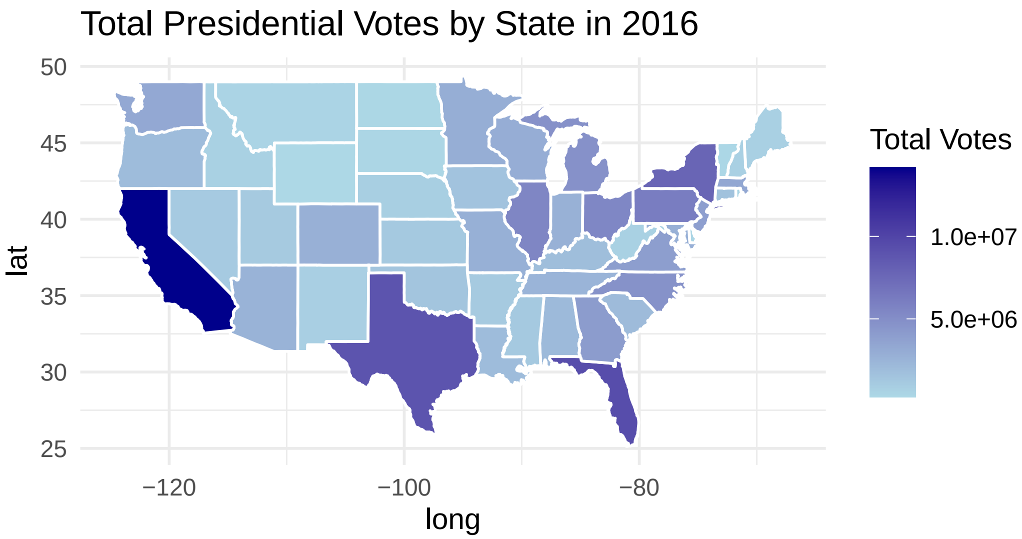
```
Attaching package: 'maps'

The following object is masked from 'package:purrr':

    map
```

```r
# loading data
us_presidents <- read.csv("us-presidents.csv")

# sample years
year1 <- 2000
year2 <- 2016

data_year1 <- us_presidents %>% filter(year == year1)
data_year2 <- us_presidents %>% filter(year == year2)

# Get map data for the US
states_map <- map_data("state")

# Function to merge state data with total votes
prepare_map_data <- function(election_data) {
  election_data$region <- tolower(election_data$state)
  merged_data <- merge(
    states_map,
    election_data,
    by = "region",
    all.x = TRUE
  )
  return(merged_data)
}

# Plotting data
map_data_year1 <- prepare_map_data(data_year1)
```

```r
# Plot the map1
map1 <- ggplot(
  map_data_year1,
  aes(x = long, y = lat, group = group, fill = totalvotes)
) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  theme_minimal() +
  scale_fill_gradient(
    low = "lightblue",
    high = "darkblue"
  ) +
  ggtitle(
    paste("Total Presidential Votes by State in", year1)
  ) +
  labs(fill = "Total Votes")
print(map1)
```



```r
# Plotting data
map_data_year2 <- prepare_map_data(data_year2)

# Plot the map2
map2 <- ggplot(
  map_data_year2,
  aes(x = long, y = lat, group = group, fill = totalvotes)
) +
  geom_polygon(color = "white") +
  coord_fixed(1.3) +
  theme_minimal() +
  scale_fill_gradient(
    low = "lightblue",
    high = "darkblue"
  ) +
  ggtitle(
    paste("Total Presidential Votes by State in", year2)
  ) +
  labs(fill = "Total Votes")
print(map2)
```



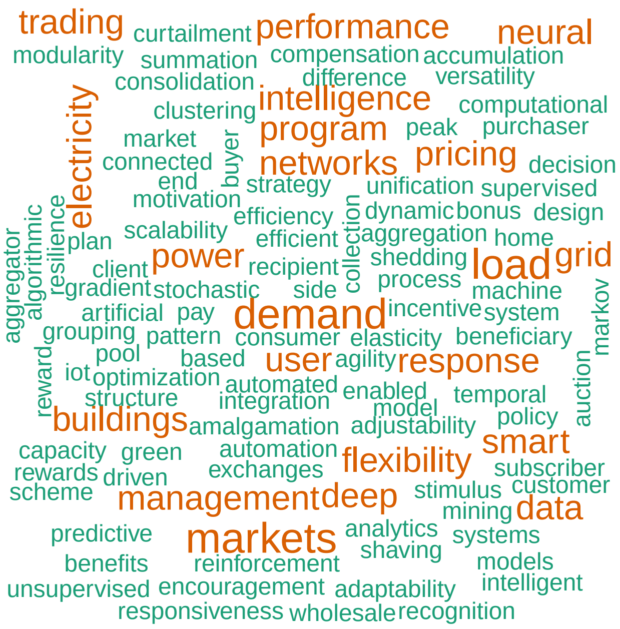
3.a.

```r
# loading libraries
library(wordcloud)
```

```
Loading required package: RColorBrewer
```

```r
# plotting wordcloud
my_text <- tolower(readLines("Research.txt", warn = FALSE))
my_text <- gsub("[[:punct:][:digit:]]", " ", my_text)
wordcloud(words = unlist(strsplit(my_text, " ")),
          min.freq = 1,
          scale = c(3, 0.5),
          colors = brewer.pal(6, "Dark2"))
```

```
Loading required namespace: tm
```


trading
curtailment
performance
neural
modularity
summation
compensation
accumulation
consolidation
difference
versatility
intelligence
computational
clustering
electricity
market
program
peak
purchaser
buyer
connected
networks
pricing
decision
end
strategy
unification
supervised
motivation
collection
efficiency
dynamic
bonus
design
scalability
efficient
aggregation
home
aggregator
algorithmic
resilience
plan
client
power
recipient
shedding
load
grid
gradient
stochastic
side
process
machine
markov
artificial
pay
demand
incentive
system
grouping
pattern
consumer
elasticity
beneficiary
pool
based
user
agility
response
reward
iot
optimization
automated
enabled
temporal
auction
structure
integration
model
buildings
amalgamation
adjustability
policy
capacity
green
automation
flexibility
smart
rewards
driven
exchanges
subscriber
scheme
management
deep
stimulus
customer
mining
data
predictive
markets
analytics
systems
shaving
benefits
reinforcement
models
unsupervised
encouragement
adaptability
intelligent
responsiveness
wholesale
recognition