Summary of Analysis

Anita Dash

2022-05-23

Libraries used in the analysis

```
## ── Attaching packages ─────────────────────────────────────── tidyverse
1.3.1 ──

## ✓ ggplot2 3.3.6     ✓ purrr   0.3.4
## ✓ tibble  3.1.7     ✓ dplyr   1.0.9
## ✓ tidyr   1.2.0     ✓ stringr 1.4.0
## ✓ readr   2.1.2     ✓ forcats 0.5.1

## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✕ dplyr::filter() masks stats::filter()
## ✕ dplyr::lag()    masks stats::lag()

##
## Attaching package: 'janitor'

## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

##
## Attaching package: 'hms'

## The following object is masked from 'package:lubridate':
##
##     hms

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

Analysis

The clean data- One_year_final.csv was read as One_year_data, and the Structure of the dataset was seen.

```
One_year_data <- fread("/Users/Anita Dash/Desktop/Google-
Capstone/One_year_data_final.csv")
#finding the structure of the data
str(One_year_data)

## Classes 'data.table' and 'data.frame':   5718079 obs. of  12 variables:
##  $ RideId      : chr  "C809ED75D6160B2A" "DD59FDCE0ACACAF3"
"0AB83CB88C43EFC2" "7881AC6D39110C60" ...
##  $ RideableType: chr  "electric_bike" "electric_bike" "electric_bike"
"electric_bike" ...
##  $ MemberCasual: chr  "casual" "casual" "casual" "casual" ...
##  $ StartDate   : IDate, format: "2021-05-30" "2021-05-30" ...
##  $ EndDate     : IDate, format: "2021-05-30" "2021-05-30" ...
##  $ StartTime   : chr  "11:58:15" "11:29:14" "14:24:01" "14:25:51" ...
##  $ EndTime     : chr  "12:10:39" "12:14:09" "14:25:13" "14:41:04" ...
##  $ RideLength  : int  744 2695 72 913 413 1416 883 1075 157 1581 ...
##  $ StartDay    : chr  "Sunday" "Sunday" "Sunday" "Sunday" ...
##  $ EndDay      : chr  "Sunday" "Sunday" "Sunday" "Sunday" ...
##  $ MonthRide   : chr  "May" "May" "May" "May" ...
##  $ RideSeason  : chr  "Summer" "Summer" "Summer" "Summer" ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Descriptive Analysis on RideLength

The mean, median, maximum and minimum Ride Length were calculated

```
#calculating mean, median, maximum and minimum ride length
mean_ride_length = as_hms(mean(One_year_data$RideLength))
print(mean_ride_length)

## 00:19:24.195633

median_ride_length = as_hms(median(One_year_data$RideLength))
print(median_ride_length)

## 00:11:27

max_ride_length = as_hms(max(One_year_data$RideLength))
print(max_ride_length)
```

```
## 932:24:09

min_ride_length = as_hms(min(One_year_data$RideLength))
print(min_ride_length)

## 00:00:00
```

Descriptive Analysis of RideLength Based on Rider Types (Member-Casual)

The mean, median, maximum, minimum Ride Length based on Types of Riders were calculated

```
#calculating mean,median, maximum, and minimum Ride length based on rider
types
aggregate(One_year_data$RideLength ~ One_year_data$MemberCasual, FUN = mean)

##   One_year_data$MemberCasual One_year_data$RideLength
## 1                     casual                1665.8109
## 2                     member                 772.5073

aggregate(One_year_data$RideLength ~ One_year_data$MemberCasual, FUN =
median)

##   One_year_data$MemberCasual One_year_data$RideLength
## 1                     casual                      927
## 2                     member                      550

aggregate(One_year_data$RideLength ~ One_year_data$MemberCasual, FUN = max)

##   One_year_data$MemberCasual One_year_data$RideLength
## 1                     casual                  3356649
## 2                     member                    93594

aggregate(One_year_data$RideLength ~ One_year_data$MemberCasual, FUN = min)

##   One_year_data$MemberCasual One_year_data$RideLength
## 1                     casual                        0
## 2                     member                        0
```

Descriptive Analysis on RideLength on Types of Rider (Member-Casual) and Day of ride

The mean Ride Length based on Types of Riders and the day of ride were calculated

```
#calculating mean ride length each day in the week based on rider types
One_year_data$StartDay <- ordered(One_year_data$StartDay, levels=c("Sunday",
"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))
aggregate(One_year_data$RideLength ~ One_year_data$MemberCasual +
One_year_data$StartDay, FUN = mean)

##    One_year_data$MemberCasual One_year_data$StartDay
One_year_data$RideLength
```

```
## 1                       casual                 Sunday
1983.8568
## 2                       member                 Sunday
888.5449
## 3                       casual                 Monday
1692.8693
## 4                       member                 Monday
747.2132
## 5                       casual                Tuesday
1427.0702
## 6                       member                Tuesday
721.1845
## 7                       casual              Wednesday
1456.1414
## 8                       member              Wednesday
731.0946
## 9                       casual               Thursday
1458.7314
## 10                      member               Thursday
730.5031
## 11                      casual                 Friday
1525.2225
## 12                      member                 Friday
752.4607
## 13                      casual               Saturday
1803.2321
## 14                      member               Saturday
868.3810
```

Further Analysis

```
#number of rides, average ride length for each day in the week based on rider
types
One_year_data %>%
  group_by(MemberCasual, StartDay) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay)
```

```
## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.
```

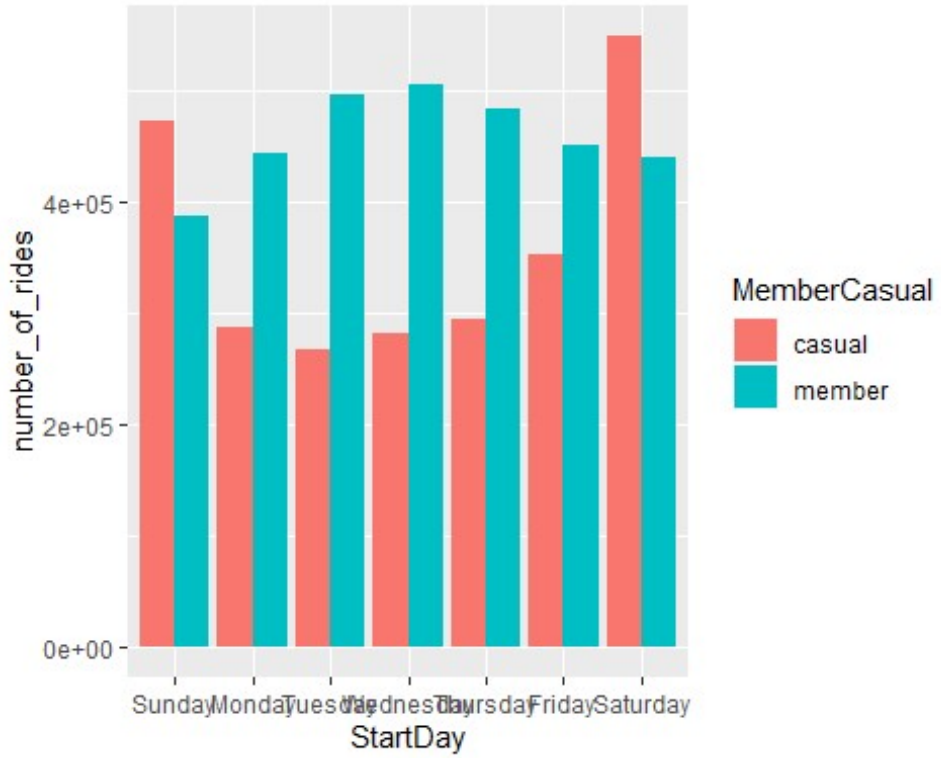
```
## # A tibble: 14 × 4
## # Groups:   MemberCasual [2]
##    MemberCasual StartDay  number_of_rides average_duration
##    <chr>        <ord>              <int>            <dbl>
##  1 casual       Sunday            473118            1984.
##  2 casual       Monday            286688            1693.
##  3 casual       Tuesday           268253            1427.
##  4 casual       Wednesday         282443            1456.
```

```
##  5 casual       Thursday           294652            1459.
##  6 casual       Friday             352197            1525.
##  7 casual       Saturday           549865            1803.
##  8 member       Sunday             387090             889.
##  9 member       Monday             444736             747.
## 10 member       Tuesday            497616             721.
## 11 member       Wednesday          505777             731.
## 12 member       Thursday           484400             731.
## 13 member       Friday             451028             752.
## 14 member       Saturday           440216             868.
```

Visualization of above analysis

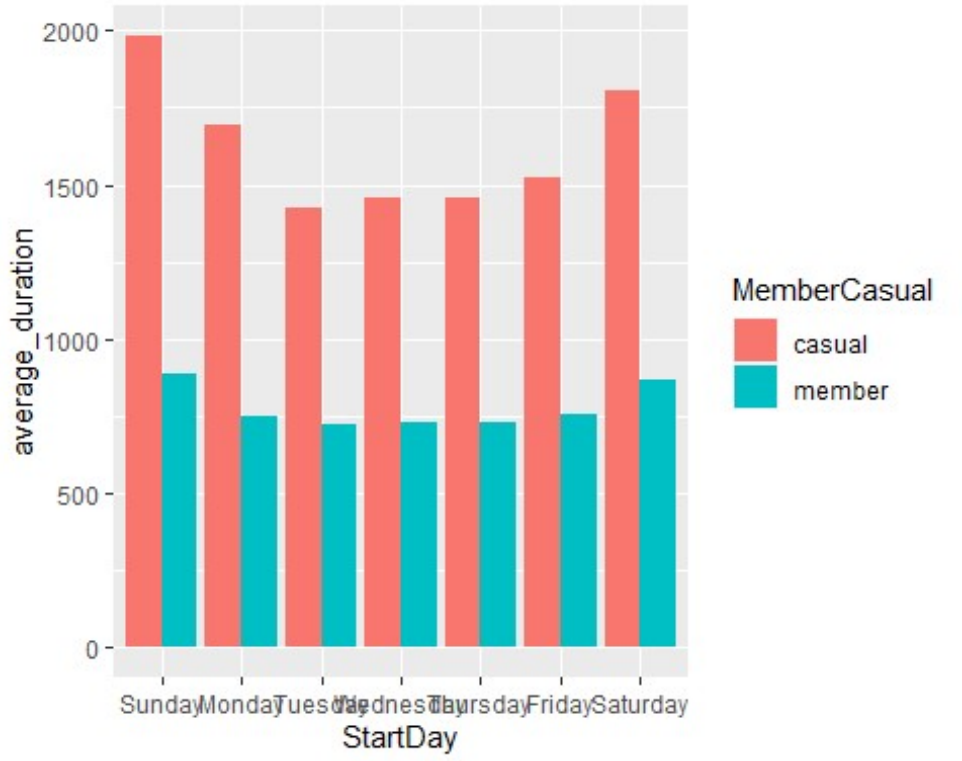
```
#visualizing the above analysis, 1. based on number of rides, 2. based on
average duration
One_year_data %>%
  group_by(MemberCasual, StartDay) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay)  %>%
  ggplot(aes(x = StartDay, y = number_of_rides, fill = MemberCasual)) +
  geom_col(position = "dodge")
```

```
## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.
```



```
One_year_data %>%
  group_by(MemberCasual, StartDay) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay)  %>%
  ggplot(aes(x = StartDay, y = average_duration, fill = MemberCasual)) +
  geom_col(position = "dodge")
```

```
## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.
```



Descriptive Analysis for each month on Types of Riders

Ordering the months in the order from may to april

```
One_year_data$MonthRide <- ordered(One_year_data$MonthRide, levels=c("May",
"June", "July", "August", "September", "October", "November", "December",
"January", "February", "March", "April"))

#number of rides, average ride length for each month based on rider types
One_year_data %>%
  group_by(MemberCasual, MonthRide) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, MonthRide)
```

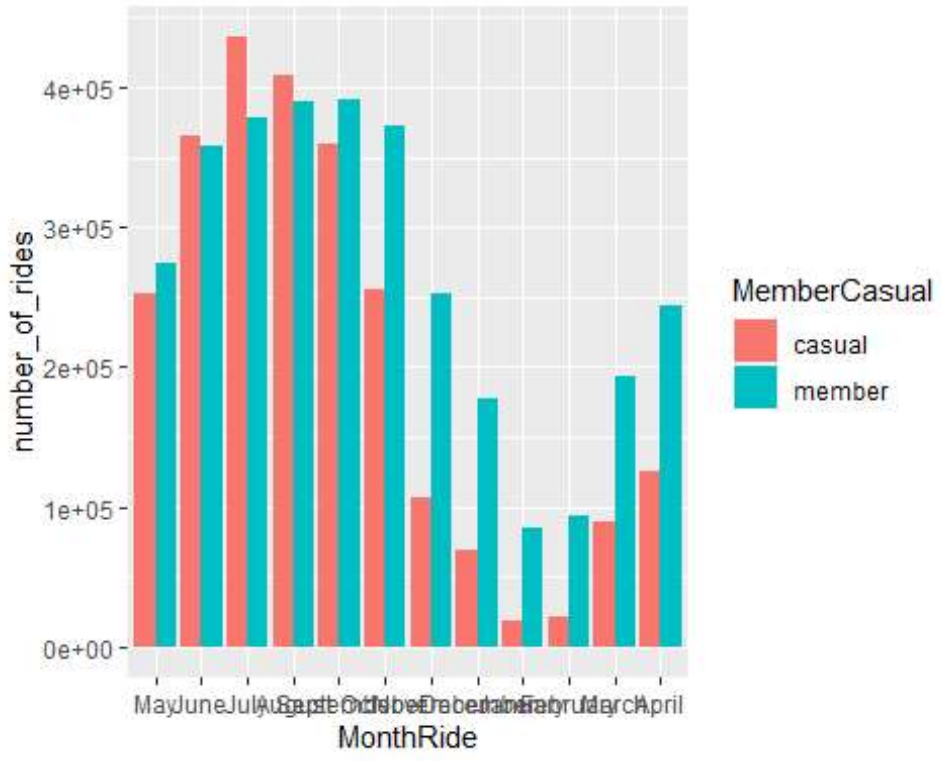
```
## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.

## # A tibble: 24 × 4
## # Groups:   MemberCasual [2]
##    MemberCasual MonthRide number_of_rides average_duration
##    <chr>        <ord>               <int>            <dbl>
##  1 casual       May                253157            1996.
##  2 casual       June               364684            1910.
##  3 casual       July               436318            1755.
##  4 casual       August             408229            1611.
##  5 casual       September          360200            1545.
##  6 casual       October            254951            1514.
##  7 casual       November           106213            1199.
##  8 casual       December            69216            1248.
##  9 casual       January             18389            1496.
## 10 casual       February            21278            1399.
## # … with 14 more rows
```

Visualization of above analysis

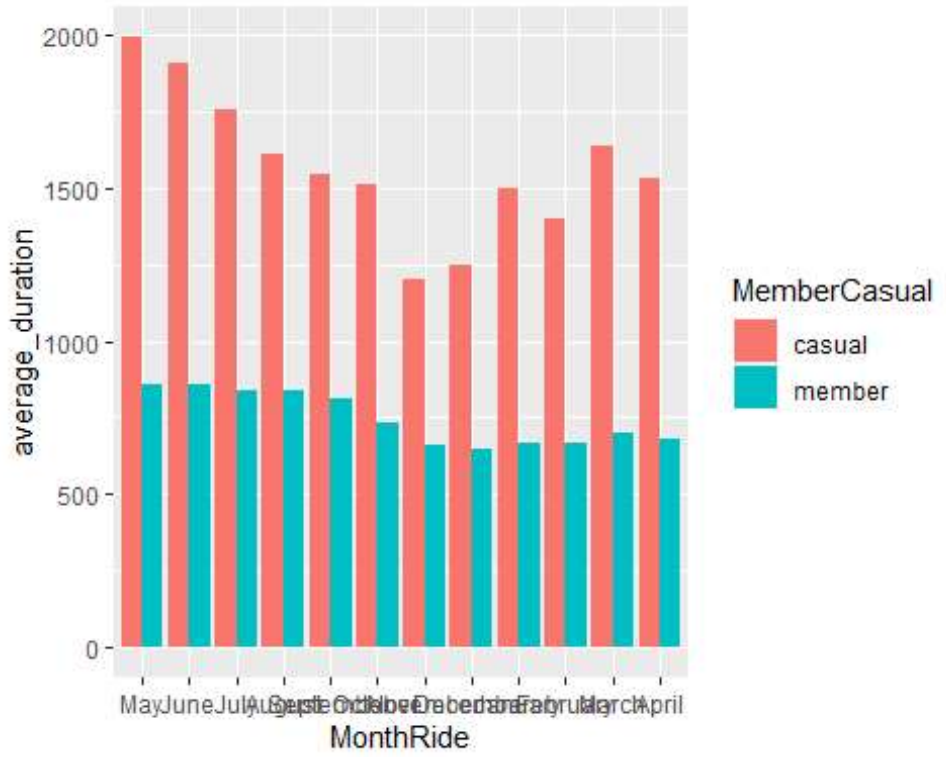
```
#visualizing the above analysis, 1. based on number of rides, 2. based on
average duration
One_year_data %>%
  group_by(MemberCasual, MonthRide) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, MonthRide)  %>%
  ggplot(aes(x = MonthRide, y = number_of_rides, fill = MemberCasual)) +
  geom_col(position = "dodge")

## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.
```




```
One_year_data %>%
  group_by(MemberCasual, MonthRide) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, MonthRide)  %>%
  ggplot(aes(x = MonthRide, y = average_duration, fill = MemberCasual)) +
  geom_col(position = "dodge")
```

```
## `summarise()` has grouped output by 'MemberCasual'. You can override using
the
## `.groups` argument.
```



Descriptive Analysis based on Type of Riders and Season

Adding a column- RideSeason to One_year_data that specifies the season the ride took place

```
#adding a column that gives the season the ride took place (DO NOT RUN this
portion of code, as column has already been added in the previous run)
One_year_data <- One_year_data %>%
  mutate(RideSeason = case_when(MonthRide == "January" ~ "Spring",
                                MonthRide == "February" ~ "Spring",
                                MonthRide == "March" ~ "Spring",
                                MonthRide == "April" ~ "Summer",
                                MonthRide == "May" ~ "Summer",
                                MonthRide == "June" ~ "Summer",
                                MonthRide == "July" ~ "Fall",
                                MonthRide == "August" ~ "Fall",
                                MonthRide == "September" ~ "Fall",
                                MonthRide == "October" ~ "Winter",
                                MonthRide == "November" ~ "Winter",
                                MonthRide == "December" ~ "Winter"))

#Calculating number of rides, mean ride length based on seasons.
One_year_data %>%
  group_by(MemberCasual, StartDay, RideSeason) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay, RideSeason)
```

```
## `summarise()` has grouped output by 'MemberCasual', 'StartDay'. You can
## override using the `.groups` argument.

## # A tibble: 56 × 5
## # Groups:   MemberCasual, StartDay [14]
##    MemberCasual StartDay RideSeason number_of_rides average_duration
##    <chr>        <ord>    <chr>                <int>            <dbl>
##  1 casual       Sunday   Fall                222811            1887.
##  2 casual       Sunday   Spring               23161            1741.
##  3 casual       Sunday   Summer              149072            2263.
##  4 casual       Sunday   Winter               78074            1801.
##  5 casual       Monday   Fall                140479            1747.
##  6 casual       Monday   Spring               21181            1718.
##  7 casual       Monday   Summer               80702            1805.
##  8 casual       Monday   Winter               44326            1306.
##  9 casual       Tuesday  Fall                123465            1426.
## 10 casual       Tuesday  Spring               15250            1267.
## # … with 46 more rows
```

Visualization of the above analysis

```
#visualizing the above analysis, 1. number of rides, 2. average duration
One_year_data %>%
  group_by(MemberCasual, StartDay, RideSeason) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay,RideSeason)  %>%
  ggplot(aes(x = RideSeason, y = number_of_rides, fill = MemberCasual)) +
  geom_col(position = "dodge")

## `summarise()` has grouped output by 'MemberCasual', 'StartDay'. You can
## override using the `.groups` argument.
```


```
One_year_data %>%
  group_by(MemberCasual, StartDay, RideSeason) %>%
  summarise(number_of_rides = n()
            ,average_duration = mean(RideLength)) %>%
  arrange(MemberCasual, StartDay, RideSeason)  %>%
  ggplot(aes(x = RideSeason, y = average_duration, fill = MemberCasual)) +
  geom_col(position = "dodge")
```

```
## `summarise()` has grouped output by 'MemberCasual', 'StartDay'. You can
## override using the `.groups` argument.
```

Saving the final data

After adding a new column, saving the new changes in one_year_final.csv

```
#DO NOT RUN this portion of the code as the updates have already been made in
the previous run
fwrite(One_year_data,
       file = "/Users/Anita Dash/Desktop/Google-
Capstone/One_year_data_final.csv",
       sep = ",")
```