Case-study

Jayanth

2023-05-18

Process phase:

install packages:

install the packages:

```
# install.packages("tidyverse")
# install.packages("lubridate")
# install.packages("geosphere")
# install.packages("ggplot2")
# install.packages("dplyr")
```

import libraries:

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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
```

```
library(lubridate)
library(geosphere)
library(ggplot2)
library(dplyr)
```

loading the dataset:

```r
jan <- read.csv("dataset/202201-divvy-tripdata.csv")
feb <- read.csv("dataset/202202-divvy-tripdata.csv")
mar <- read.csv("dataset/202203-divvy-tripdata.csv")
apr <- read.csv("dataset/202204-divvy-tripdata.csv")
may <- read.csv("dataset/202205-divvy-tripdata.csv")
jun <- read.csv("dataset/202206-divvy-tripdata.csv")
jul <- read.csv("dataset/202207-divvy-tripdata.csv")
aug <- read.csv("dataset/202208-divvy-tripdata.csv")
sep <- read.csv("dataset/202209-divvy-tripdata.csv")
oct <- read.csv("dataset/202210-divvy-tripdata.csv")
nov <- read.csv("dataset/202211-divvy-tripdata.csv")
dec <- read.csv("dataset/202212-divvy-tripdata.csv")
```

merging all the data to create year data:

```r
year_data <- bind_rows(jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec)
```

creating new columns day and month:

```r
year_data$started_at <- as.POSIXct(year_data$started_at, format = "%Y-%m-%d %H:%M:%S")
year_data$ended_at <- strptime(year_data$ended_at, format = "%Y-%m-%d %H:%M:%S")

year_data <- year_data %>%
  mutate(ride_weekday = wday(started_at, label = TRUE, abbr = FALSE),
         ride_month = month(started_at, label = TRUE, abbr = FALSE))
```

add time in hours form start to end

```r
year_data <- year_data %>%
  mutate(ride_time_mins = as.numeric(difftime(ended_at, started_at, units="mins")))
```

ride_distance:

```r
year_data$ride_distance <- distHaversine(matrix(c(year_data$start_lng, year_data$start_lat)
                                                , ncol = 2),
                matrix(c(year_data$end_lng, year_data$end_lat), ncol = 2))

year_data$ride_distance <- year_data$ride_distance/ 1000
```

basic cleaing:

1. removing nulls

```
year_data <- drop_na(year_data)
```

2. removing negative distances

```
year_data <- year_data %>%
  filter(ride_distance>0)
```

Analyze phase:

finding no.of casual and member rides took place in the last year.

```
per_membership <- year_data %>% summarize(
  casual_percentage = sum(member_casual=="casual")*100/n(),
  member_percentage = sum(member_casual=="member")*100/n()
)

per_membership
```

```
##   casual_percentage member_percentage
## 1          40.02238          59.97762
```

finding different rideable_type took place in the last year.

```
rideable_types <- unique(year_data$rideable_type)
print(rideable_types)
```

```
## [1] "electric_bike" "classic_bike"  "docked_bike"
```

```
per_ridetypes <- year_data %>% summarize(
  electric_bike_percentage = sum(rideable_type=="electric_bike")*100/n(),
  classic_bike_percentage = sum(rideable_type=="classic_bike")*100/n(),
  docked_bike_percentage = sum(rideable_type=="docked_bike")*100/n()
)

per_ridetypes
```

```
##   electric_bike_percentage classic_bike_percentage docked_bike_percentage
## 1                  51.7019                45.61628               2.681818
```

summarize rideable types and group by their membership:

```
per_ridetypes_group_menbership <- year_data %>% group_by(member_casual) %>% summarize(
  electric_bike_percentage = sum(rideable_type=="electric_bike")*100/n(),
  classic_bike_percentage = sum(rideable_type=="classic_bike")*100/n(),
  docked_bike_percentage = sum(rideable_type=="docked_bike")*100/n()
)

per_ridetypes_group_menbership
```

```
## # A tibble: 2 x 4
##   member_casual electric_bike_percentage classic_bike_percentage
##   <chr>                            <dbl>                   <dbl>
## 1 casual                            55.7                    37.6
## 2 member                            49.0                    51.0
## # i 1 more variable: docked_bike_percentage <dbl>
```

monthly and weekly rides:

monthly:

```
monthly_summarize_data <- year_data %>% summarise(
  Jan = sum(ride_month=="January")*100/n(),
  Feb = sum(ride_month=="February")*100/n(),
  Mar = sum(ride_month=="March")*100/n(),
  Apr = sum(ride_month=="April")*100/n(),
  May = sum(ride_month=="May")*100/n(),
  Jun = sum(ride_month=="June")*100/n(),
  Jul = sum(ride_month=="July")*100/n(),
  Aug = sum(ride_month=="August")*100/n(),
  Sep = sum(ride_month=="September")*100/n(),
  Oct = sum(ride_month=="October")*100/n(),
  Nov = sum(ride_month=="November")*100/n(),
  Dec = sum(ride_month=="December")*100/n(),
)

monthly_summarize_data
```

```
##        Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 1 1.838579 2.031862 4.967572 6.453197 11.08053 13.53395 14.50834 13.90212
##        Sep      Oct     Nov      Dec
## 1 12.45529 9.950568 6.02881 3.249181
```

monthly_summarize data group by membership:

```
monthly_summarize_groupby_member_data <- year_data %>% group_by(member_casual) %>% summarise(
  Jan = sum(ride_month=="January")*100/n(),
  Feb = sum(ride_month=="February")*100/n(),
```

```
  Mar = sum(ride_month=="March")*100/n(),
  Apr = sum(ride_month=="April")*100/n(),
  May = sum(ride_month=="May")*100/n(),
  Jun = sum(ride_month=="June")*100/n(),
  Jul = sum(ride_month=="July")*100/n(),
  Aug = sum(ride_month=="August")*100/n(),
  Sep = sum(ride_month=="September")*100/n(),
  Oct = sum(ride_month=="October")*100/n(),
  Nov = sum(ride_month=="November")*100/n(),
  Dec = sum(ride_month=="December")*100/n(),
)

monthly_summarize_groupby_member_data
```

```
## # A tibble: 2 x 13
##   member_casual   Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct
##   <chr>         <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 casual        0.795 0.915  3.80  5.37  11.9  15.8  17.5  15.5  12.9  9.11
## 2 member        2.53  2.78   5.74  7.17  10.5  12.0  12.5  12.8  12.2 10.5
## # i 2 more variables: Nov <dbl>, Dec <dbl>
```

Week Day:

```
weekly_summarize_data <- year_data %>% summarise(
  Mon = sum(ride_weekday=="Monday")*100/n(),
  Tue = sum(ride_weekday=="Tuesday")*100/n(),
  Wed = sum(ride_weekday=="Wednesday")*100/n(),
  Thu = sum(ride_weekday=="Thursday")*100/n(),
  Fri = sum(ride_weekday=="Friday")*100/n(),
  Sat = sum(ride_weekday=="Saturday")*100/n(),
  Sun = sum(ride_weekday=="Sunday")*100/n(),
)

weekly_summarize_data
```

```
##        Mon      Tue      Wed      Thu      Fri      Sat      Sun
## 1 13.22716 13.89498 14.19133 14.93743 14.17828 16.05743 13.51339
```

week_summarize data group by membership:

```
weekly_summarize_groupby_member_data <- year_data %>% group_by(member_casual) %>% summarise(
  Mon = sum(ride_weekday=="Monday")*100/n(),
  Tue = sum(ride_weekday=="Tuesday")*100/n(),
  Wed = sum(ride_weekday=="Wednesday")*100/n(),
  Thu = sum(ride_weekday=="Thursday")*100/n(),
  Fri = sum(ride_weekday=="Friday")*100/n(),
  Sat = sum(ride_weekday=="Saturday")*100/n(),
  Sun = sum(ride_weekday=="Sunday")*100/n(),
)
```

```
weekly_summarize_groupby_member_data
```

```
## # A tibble: 2 x 8
##   member_casual   Mon   Tue   Wed   Thu   Fri   Sat   Sun
##   <chr>         <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 casual         11.9  11.4  11.9  13.4  14.5  20.3  16.6
## 2 member         14.1  15.6  15.7  16.0  14.0  13.2  11.5
```

average ride times

```
avg_ridetime <- year_data %>% summarise(
  avg_ride_time_in_mins = mean(ride_time_mins)
)

avg_ridetime
```

```
##   avg_ride_time_in_mins
## 1              15.93793
```

group by membership and ride types:

```
avg_ridetime_group <- year_data %>% group_by(member_casual, rideable_type) %>%
  summarise(avg_ride_time_in_mins = mean(ride_time_mins)
)
```

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

```
avg_ridetime_group
```

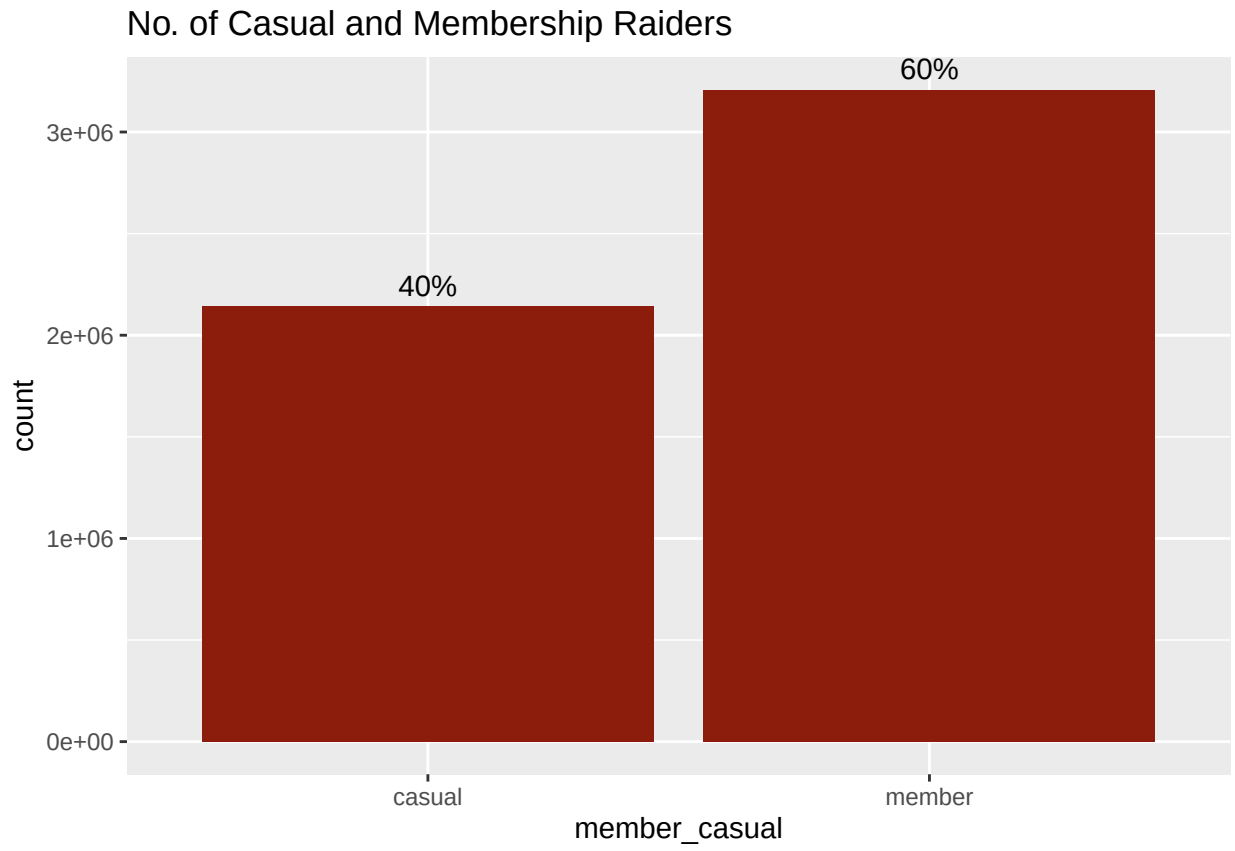
```
## # A tibble: 5 x 3
## # Groups:   member_casual [2]
##   member_casual rideable_type avg_ride_time_in_mins
##   <chr>         <chr>                         <dbl>
## 1 casual        classic_bike                   23.4
## 2 casual        docked_bike                    47.7
## 3 casual        electric_bike                  16.5
## 4 member        classic_bike                   13.2
## 5 member        electric_bike                  11.6
```

Share phase:

no.of casual and membership raiders:

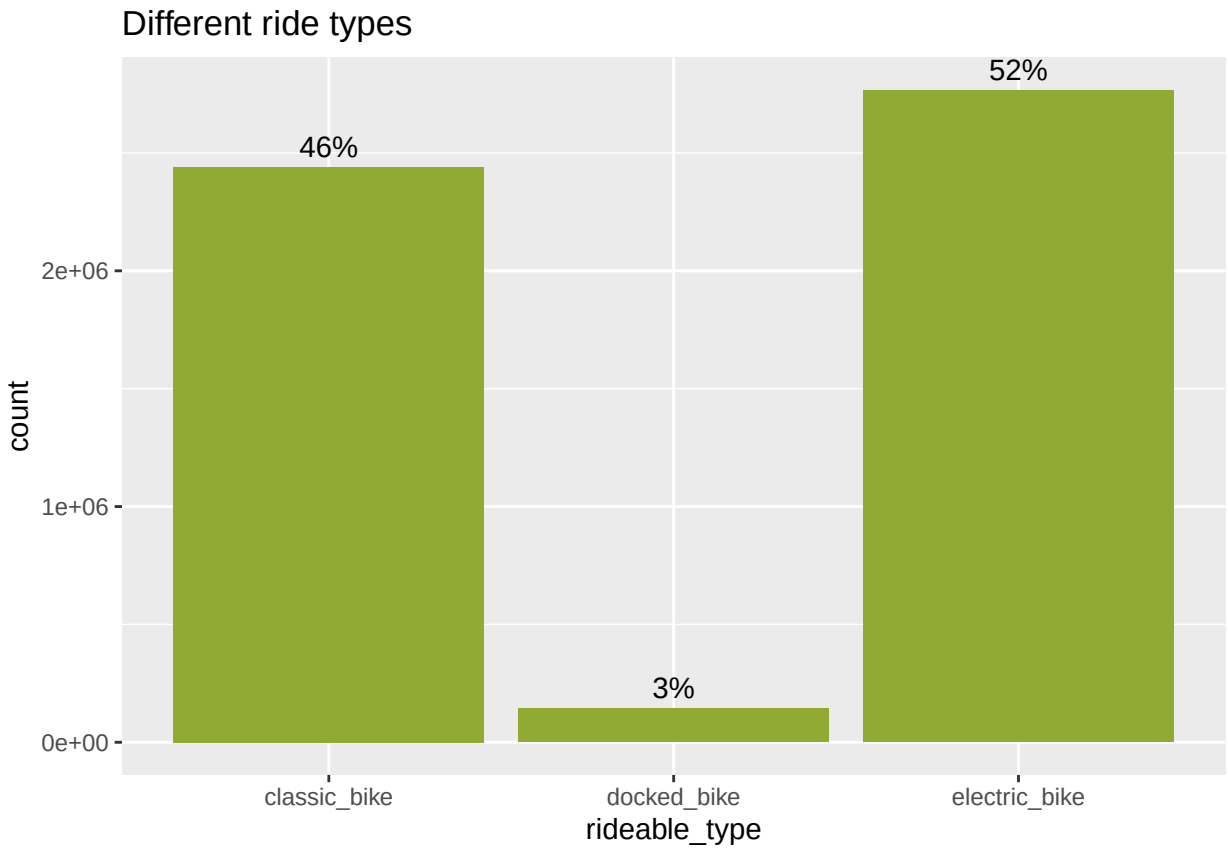
```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x = member_casual, y = ..count..), stat = "count", fill="#8c1c0b") +
  geom_text(mapping = aes(x = member_casual, y = ..count.., label = paste0(round((..count.. / sum(..cou
    stat = "count",vjust = -0.5,size = 4) +
  labs(title = "No. of Casual and Membership Raiders")
```

```
## Warning: The dot-dot notation (‘..count..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(count)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



different types of rides used:

```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x=rideable_type, y = ..count..), stat = "count" ,fill = "#91aa31") +
  geom_text(mapping = aes(x=rideable_type, y = ..count.., label = paste0(round((..count.. / sum(..count
            stat = "count",vjust = -0.5,size = 4)+
  labs(title = "Different ride types")
```

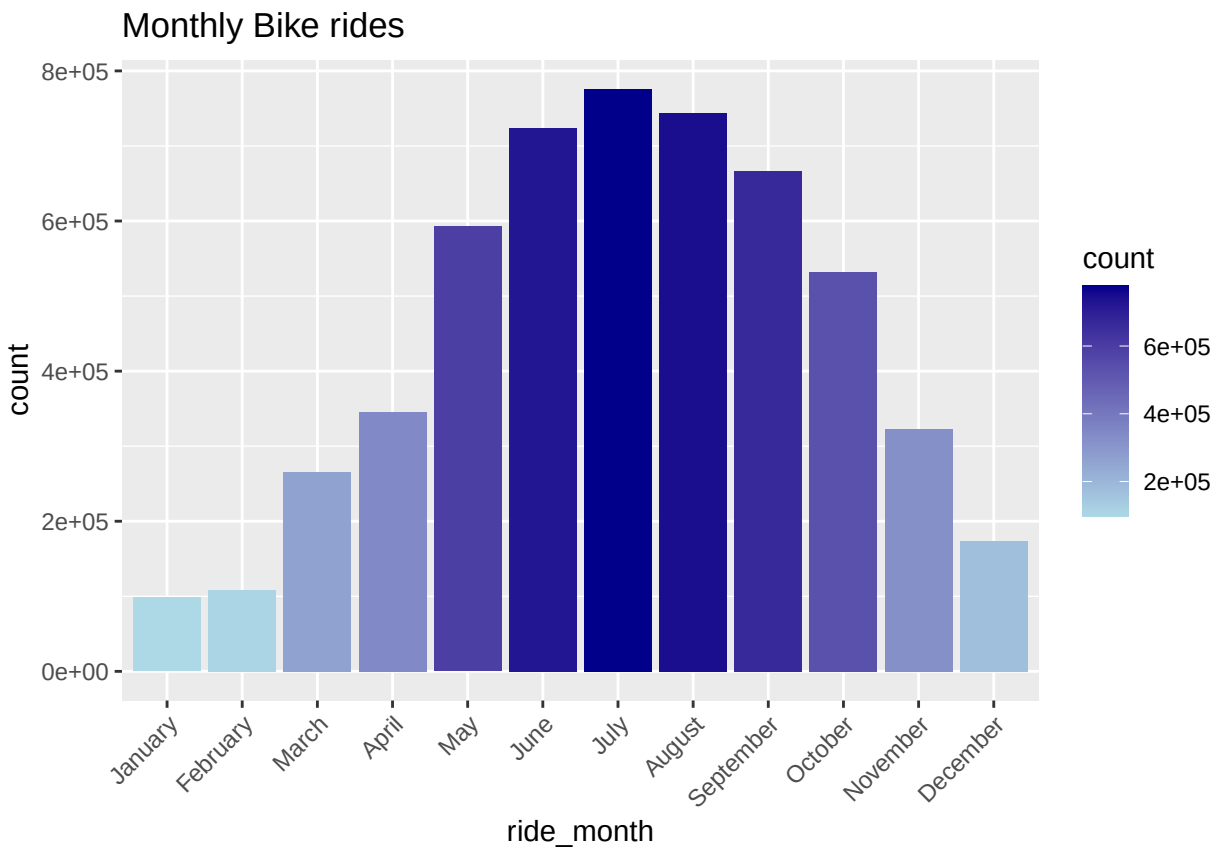


different rides and membership:

```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x = member_casual, fill = rideable_type))+
  labs(title = "different ride types and membership")
```

Monthly bike rides:

```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x=ride_month, y = ..count.., fill = after_stat(count))
           , stat = "count") +
  scale_fill_gradient(low = "lightblue", high = "darkblue") +
  labs(title = "Monthly Bike rides")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

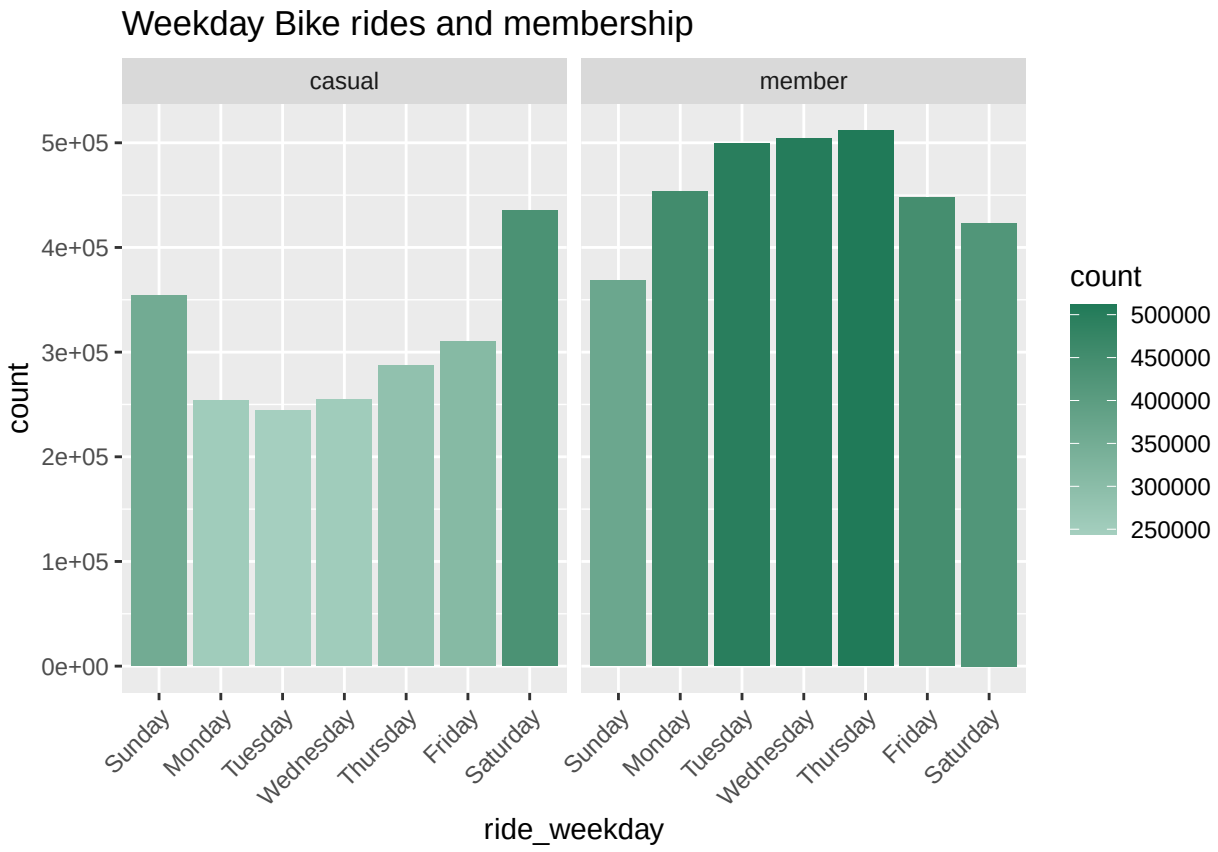


Monthly bike rides and their membership

```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x=ride_month, fill=member_casual),
           stat = "count") +
  labs(title = "Monthly Bike rides and membership")+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

weekly bike rides and membership

```
ggplot(data = year_data) +
  geom_bar(mapping = aes(x=ride_weekday, y = ..count.., fill = after_stat(count)),
           stat = "count") +
  labs(title = "Weekday Bike rides and membership")+
  scale_fill_gradient(low = "#a5cfbf", high = "#207a58") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  facet_wrap(~member_casual)
```


Weekday Bike rides and membership
casual
member
count
5e+05
4e+05
3e+05
2e+05
1e+05
0e+00
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
ride_weekday
count
500000
450000
400000
350000
300000
250000