511_project_EDA

2022-11-15

EDA

```r
# Import data

data <- read.csv('/Users/anthonymoubarak/Desktop/511 Project/Data/Combined_df.csv')
```

```r
# Data cleaning

# Convert format to yyyy
library(dplyr)
```

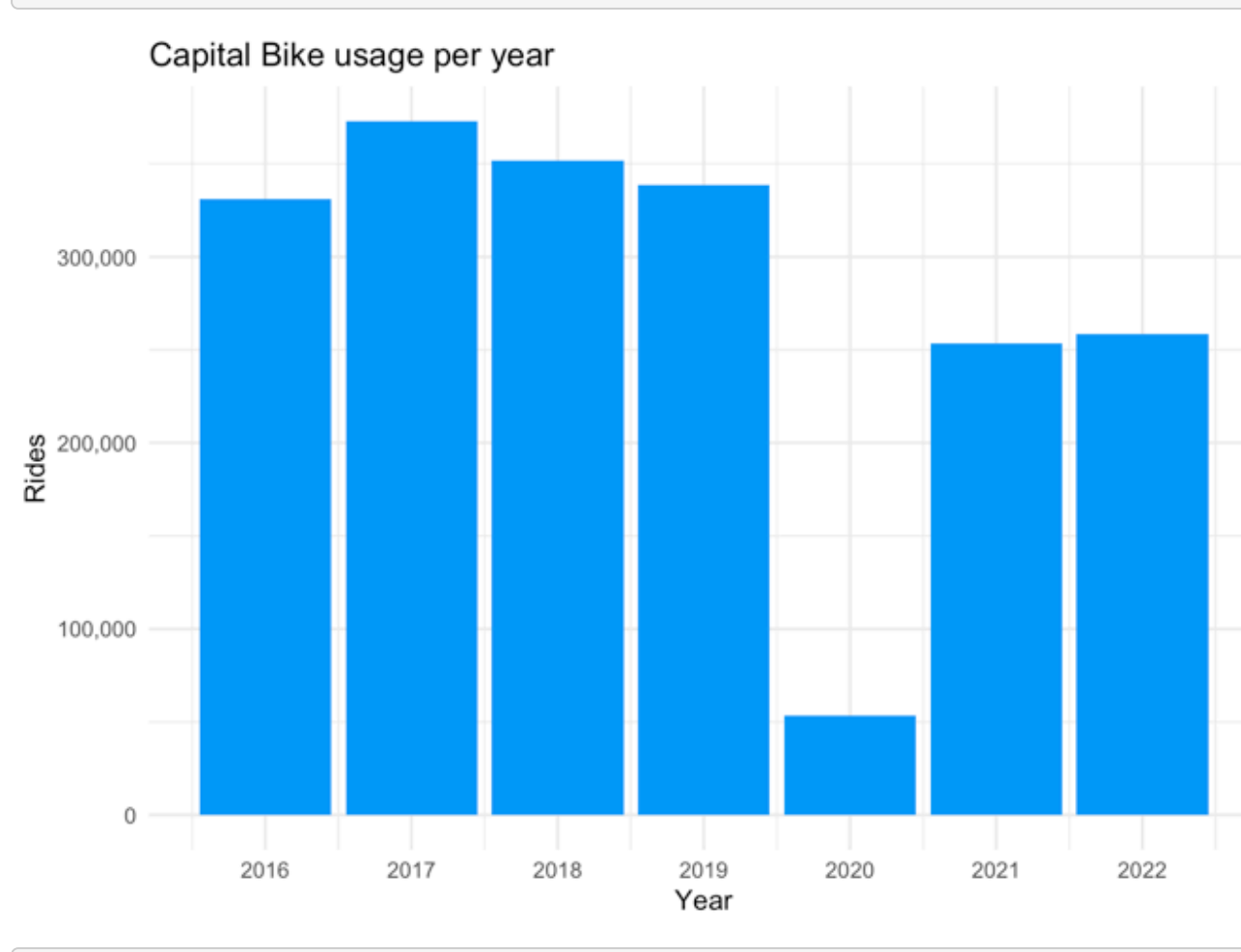
```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

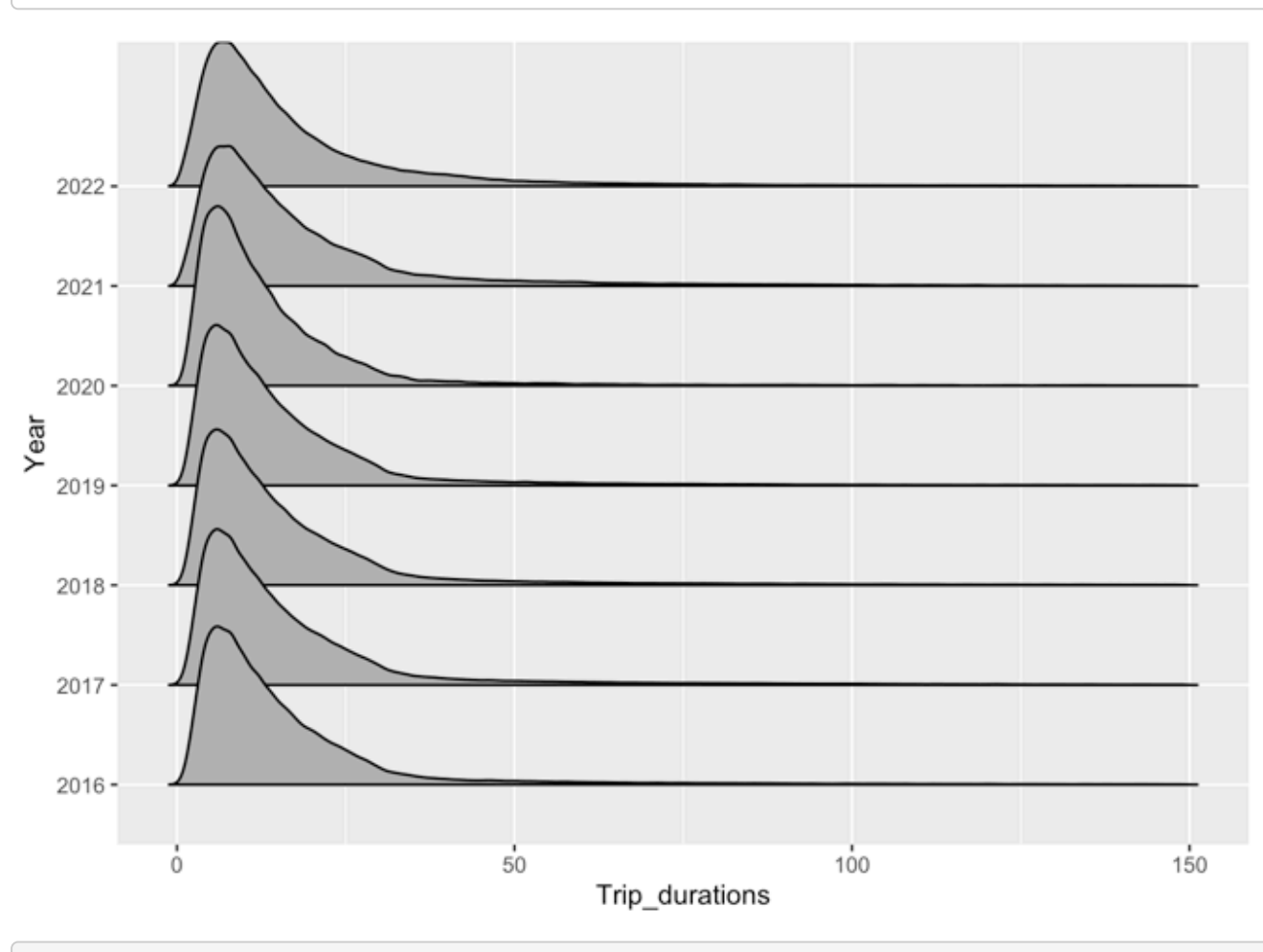
```r
y <- data$Start.date
y1 <- as.Date(y)
year <- as.numeric(format(y1,'%Y'))


data$start_year <- year

# Drop useless columns

useless_columns <- c("Start.date", "End.date", "Start.station", "End.station.number", "End.station", "Start.stati
on")

df_cleaned <- select(data,-c("Start.date", "End.date", "Start.station", "End.station.number", "End.station", "Sta
rt.station"))
```

```r
# EDA

# Start by analyzing the number of trips per year (barplot)

usage <- data.frame(unclass(table(df_cleaned$start_year)))

usage <- data.frame(c(2016,2017,2018,2019,2020, 2021, 2022) , usage$unclass.table.df_cleaned.start_year..)

names(usage) <- c("Year" , "Rides")


library(ggplot2)
library(scales)
ggplot(usage, aes(x = Year, y = Rides)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per year"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma) +
  scale_x_continuous(breaks = usage$Year) + theme_minimal()
```

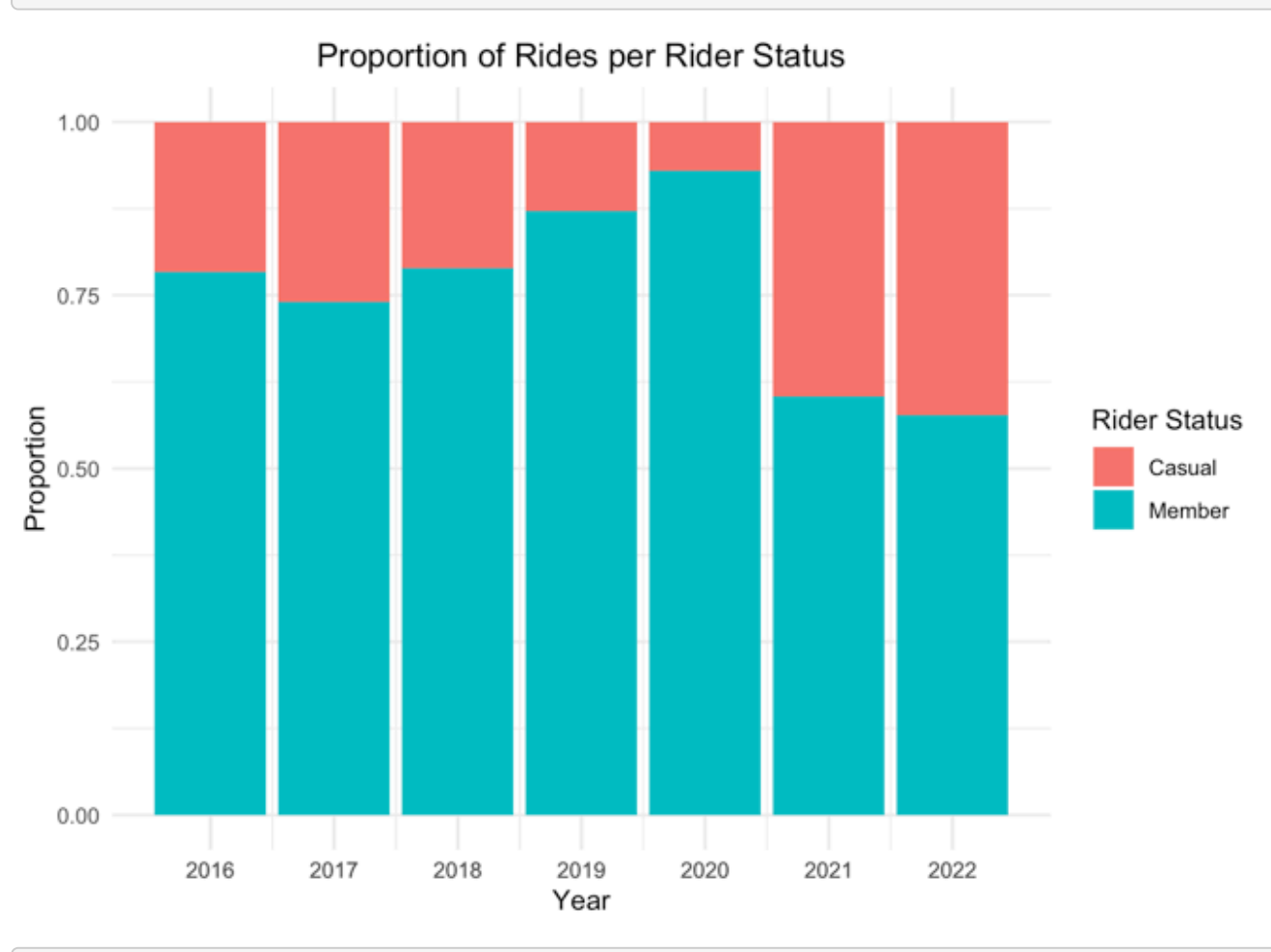


```r
# EDA

# Start by analyzing the average trip time per year (boxplot)


durations_2016 <- df_cleaned[df_cleaned$start_year == 2016,]$Duration
durations_2017 <- df_cleaned[df_cleaned$start_year == 2017,]$Duration
durations_2018 <- df_cleaned[df_cleaned$start_year == 2018,]$Duration
durations_2019 <- df_cleaned[df_cleaned$start_year == 2019,]$Duration
durations_2020 <- df_cleaned[df_cleaned$start_year == 2020,]$Duration
durations_2021 <- df_cleaned[df_cleaned$start_year == 2021,]$Duration
durations_2022 <- df_cleaned[df_cleaned$start_year == 2022,]$Duration


df_durations<-data.frame(Trip_durations=c(durations_2016,durations_2017, durations_2018, durations_2019,
                    durations_2020, durations_2021, durations_2022),Year=rep(c("2016","2017", "2018", "2019", "202
0", "2021", "2022"),times=c(330906,372946,351797,338450, 53555,253582,258237)))

library(ggridges)

#ggplot(df_durations, aes(x=X , y=as.factor(Grp), fill=after_stat(x))) + geom_density_ridges_gradient(scale=0.4,
rel_min_height=0.05) + scale_fill_viridis_c(name = 'Duration', option='C')

x <- df_durations[(df_durations$Trip_durations < 150) & (df_durations$Trip_durations > 0),]
ggplot(x, aes(x = Trip_durations, y = Year)) +
  geom_density_ridges2()
```

```
## Picking joint bandwidth of 0.725
```



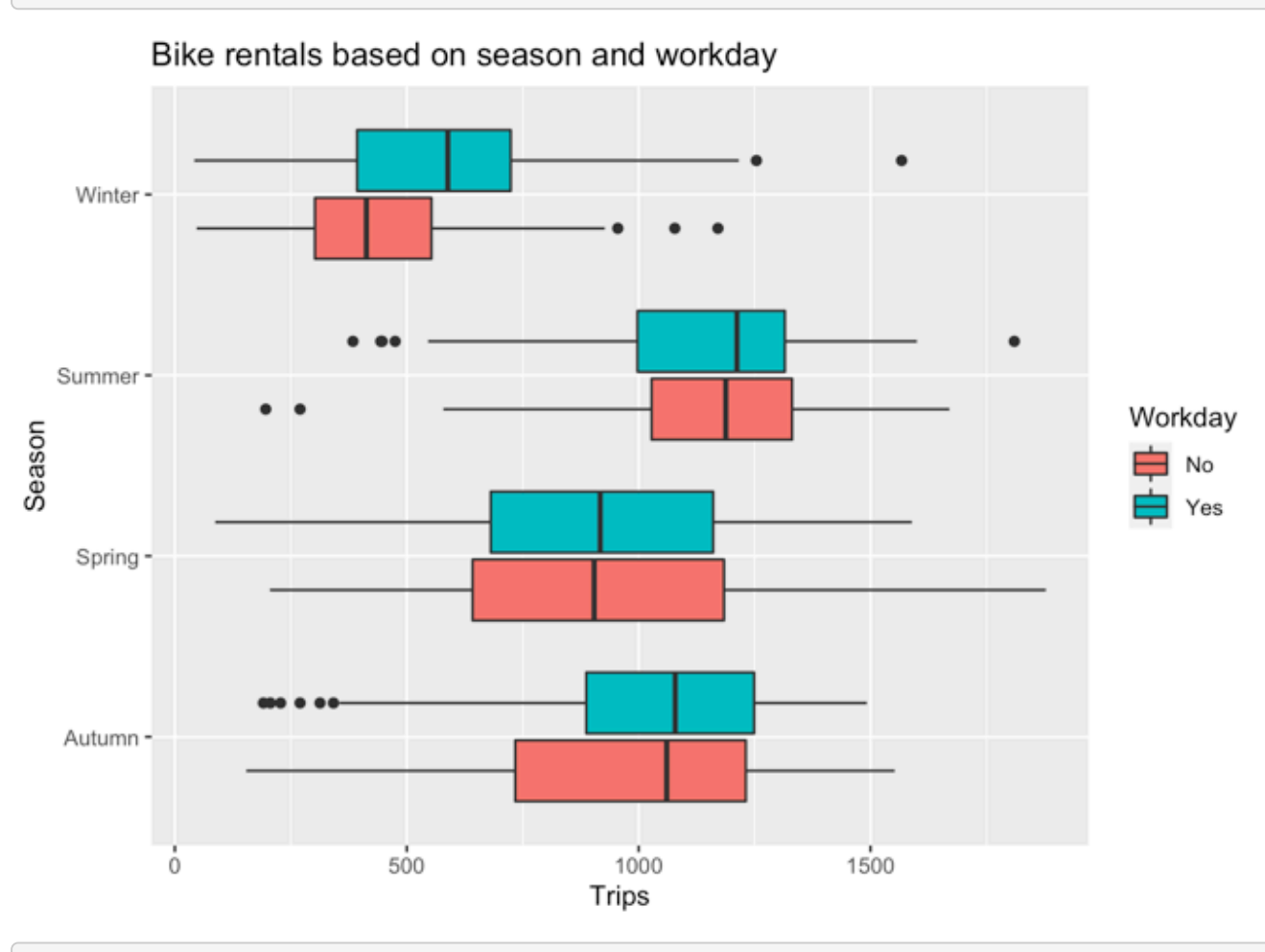
```r
# Membership growth per year


df_cleaned$Member.type[df_cleaned$Member.type == 'casual'] <- 'Casual'
df_cleaned$Member.type[df_cleaned$Member.type == 'member'] <- 'Member'


# 2016 membership info
membership_16 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2016 , ]$Member.type)))

membership_16 <- data.frame(c('Casual','Member') , membership_16$unclass.table.df_cleaned.df_cleaned.start_year..
..2016....Member.type..)

names(membership_16) <- c('Membrship_status' , 'Count')


# 2017 membership split

membership_17 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2017 , ]$Member.type)))

membership_17 <- data.frame(c('Casual','Member') , membership_17$unclass.table.df_cleaned.df_cleaned.start_year..
..2017....Member.type..)

names(membership_17) <- c('Membrship_status' , 'Count')

# 2018 membership split

membership_18 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2018 , ]$Member.type)))

membership_18 <- data.frame(c('Casual','Member') , membership_18$unclass.table.df_cleaned.df_cleaned.start_year..
..2018....Member.type..)

names(membership_18) <- c('Membrship_status' , 'Count')


# 2019 memberhsip split

membership_19 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2019 , ]$Member.type)))

membership_19 <- data.frame(c('Casual','Member') , membership_19$unclass.table.df_cleaned.df_cleaned.start_year..
..2019....Member.type..)

names(membership_19) <- c('Membrship_status' , 'Count')


# 2020 memberhsip split

membership_20 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2020 , ]$Member.type)))

membership_20 <- data.frame(c('Casual','Member') , membership_20$unclass.table.df_cleaned.df_cleaned.start_year..
..2020....Member.type..)

names(membership_20) <- c('Membrship_status' , 'Count')

# 2021 membership split

membership_21 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2021 , ]$Member.type)))

membership_21 <- data.frame(c('Casual','Member') , membership_21$unclass.table.df_cleaned.df_cleaned.start_year..
..2021....Member.type..)

names(membership_21) <- c('Membrship_status' , 'Count')

# 2022 membership split

membership_22 <- data.frame(unclass(table(df_cleaned[df_cleaned$start_year == 2022 , ]$Member.type)))

membership_22 <- data.frame(c('Casual','Member') , membership_22$unclass.table.df_cleaned.df_cleaned.start_year..
..2022....Member.type..)

names(membership_22) <- c('Membrship_status' , 'Count')



df_membership <- data.frame(c(rep(2016 , 2) , rep(2017,2), rep(2018 , 2), rep(2019, 2), rep(2020, 2), rep(2021,2)
, rep(2022 , 2) ) ,   rep(c("Member" , "Casual") , 7))
names(df_membership) <- c("start_year" , "Member.type")
df_membership$value <- ''




df_membership <- df_membership %>%
  mutate(value = case_when(
    start_year == 2016 & Member.type == 'Casual'  ~ as.numeric(membership_16[membership_16['Membrship_status'] ==
'Casual'][2]),
    start_year == 2016 & Member.type == 'Member'  ~ as.numeric(membership_16[membership_16['Membrship_status'] ==
'Member'][2]),

    start_year == 2017 & Member.type == 'Casual'  ~ as.numeric(membership_17[membership_17['Membrship_status'] ==
'Casual'][2]),
    start_year == 2017 & Member.type == 'Member'  ~ as.numeric(membership_17[membership_17['Membrship_status'] ==
'Member'][2]),

    start_year == 2018 & Member.type == 'Casual'  ~ as.numeric(membership_18[membership_18['Membrship_status'] ==
'Casual'][2]),
    start_year == 2018 & Member.type == 'Member'  ~ as.numeric(membership_18[membership_18['Membrship_status'] ==
'Member'][2]),


    start_year == 2019 & Member.type == 'Casual'  ~ as.numeric(membership_19[membership_19['Membrship_status'] ==
'Casual'][2]),
    start_year == 2019 & Member.type == 'Member'  ~ as.numeric(membership_19[membership_19['Membrship_status'] ==
'Member'][2]),

    start_year == 2020 & Member.type == 'Casual'  ~ as.numeric(membership_20[membership_20['Membrship_status'] ==
'Casual'][2]),
    start_year == 2020 & Member.type == 'Member'  ~ as.numeric(membership_20[membership_20['Membrship_status'] ==
'Member'][2]),

    start_year == 2021 & Member.type == 'Casual'  ~ as.numeric(membership_21[membership_21['Membrship_status'] ==
'Casual'][2]),
    start_year == 2021 & Member.type == 'Member'  ~ as.numeric(membership_21[membership_21['Membrship_status'] ==
'Member'][2]),

    start_year == 2022 & Member.type == 'Casual'  ~ as.numeric(membership_22[membership_22['Membrship_status'] ==
'Casual'][2]),
    start_year == 2022 & Member.type == 'Member'  ~ as.numeric(membership_22[membership_22['Membrship_status'] ==
'Member'][2])))


# Stacked + percent
ggplot(df_membership, aes(fill=Member.type, y=value, x=start_year) ) +
    geom_bar(position="fill", stat="identity") + ggtitle('Proportion of Rides per Rider Status') +scale_x_continu
ous(breaks = df_membership$start_year ) +xlab('Year') + ylab("Proportion")  + labs(fill = 'Rider Status') + theme
_minimal() + theme(plot.title = element_text(hjust = 0.5))
```



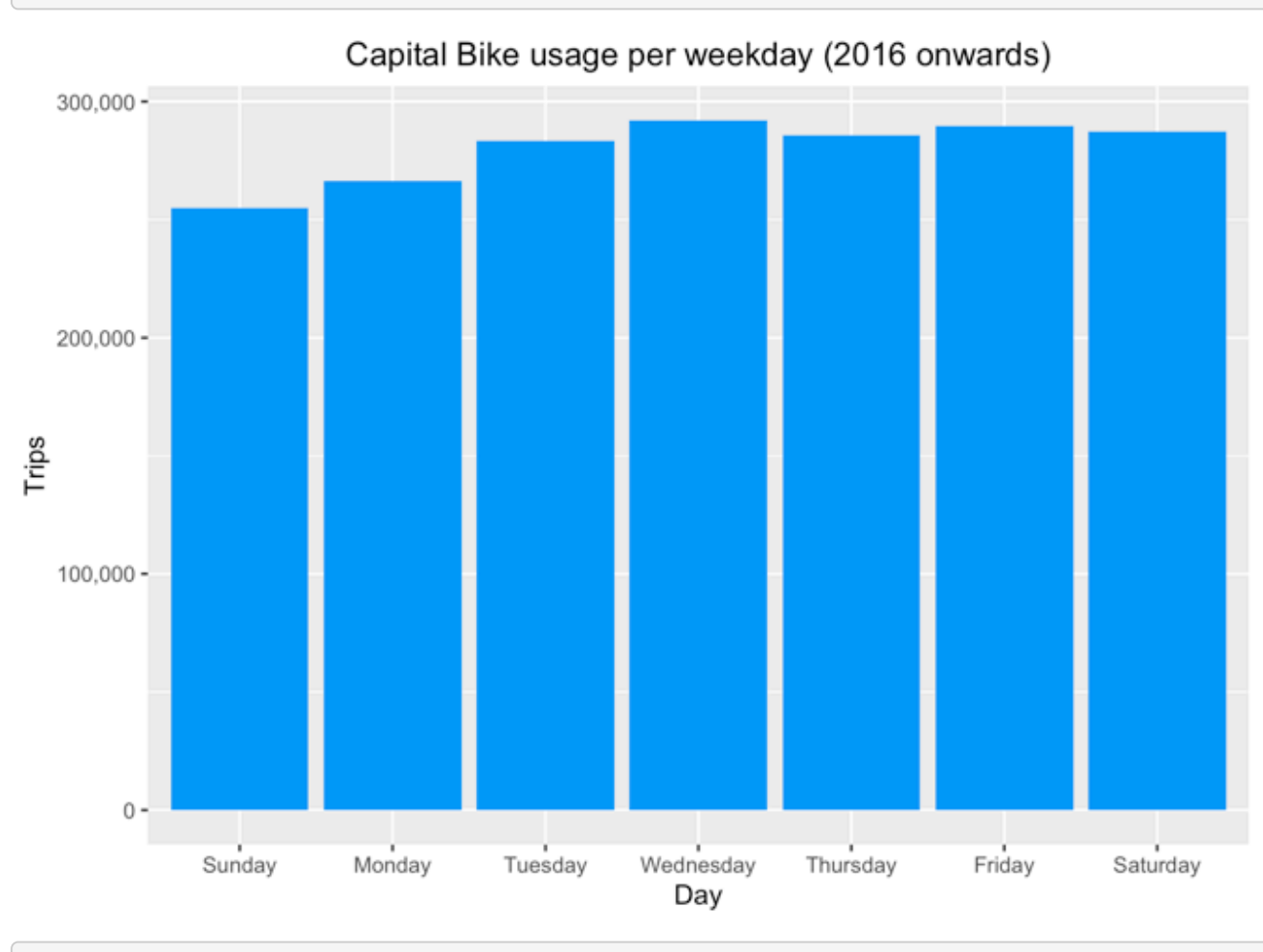
```r
# Season and work day related analysis

# Get the day and month of each entry

df_seasonal <- data['Start.date']

df_seasonal <- df_seasonal %>% mutate(Start.date = format(as.Date(Start.date), "%y-%m-%d")) %>% mutate(Month = fo
rmat(as.Date(Start.date), "%m")) %>%
    mutate(Day = weekdays(as.Date(Start.date))) %>%
    mutate(Season = case_when(Month %in% c('09' , '10', '11') ~ 'Autumn',
                              Month %in% c('12' , '01', '02') ~ 'Winter',
                              Month %in% c('03' , '04', '05') ~ 'Spring',
                              Month %in% c('06' , '07', '08') ~ 'Summer'
                              ) ) %>%
    mutate(Workday = case_when(Day %in% c('Saturday' , 'Sunday') ~ 'No',
                               Day %in% c('Monday' , 'Tuesday', 'Wednesday', 'Thursday', 'Friday') ~ 'Yes'))

df_seasonal$N_trips <- c(1)
# Group by the results

df_seasonal_grouped <- df_seasonal %>% group_by(Start.date , Season , Workday) %>% summarise(Trips = n())
```

```
## `summarise()` has grouped output by 'Start.date', 'Season'. You can override
## using the `.groups` argument.
```

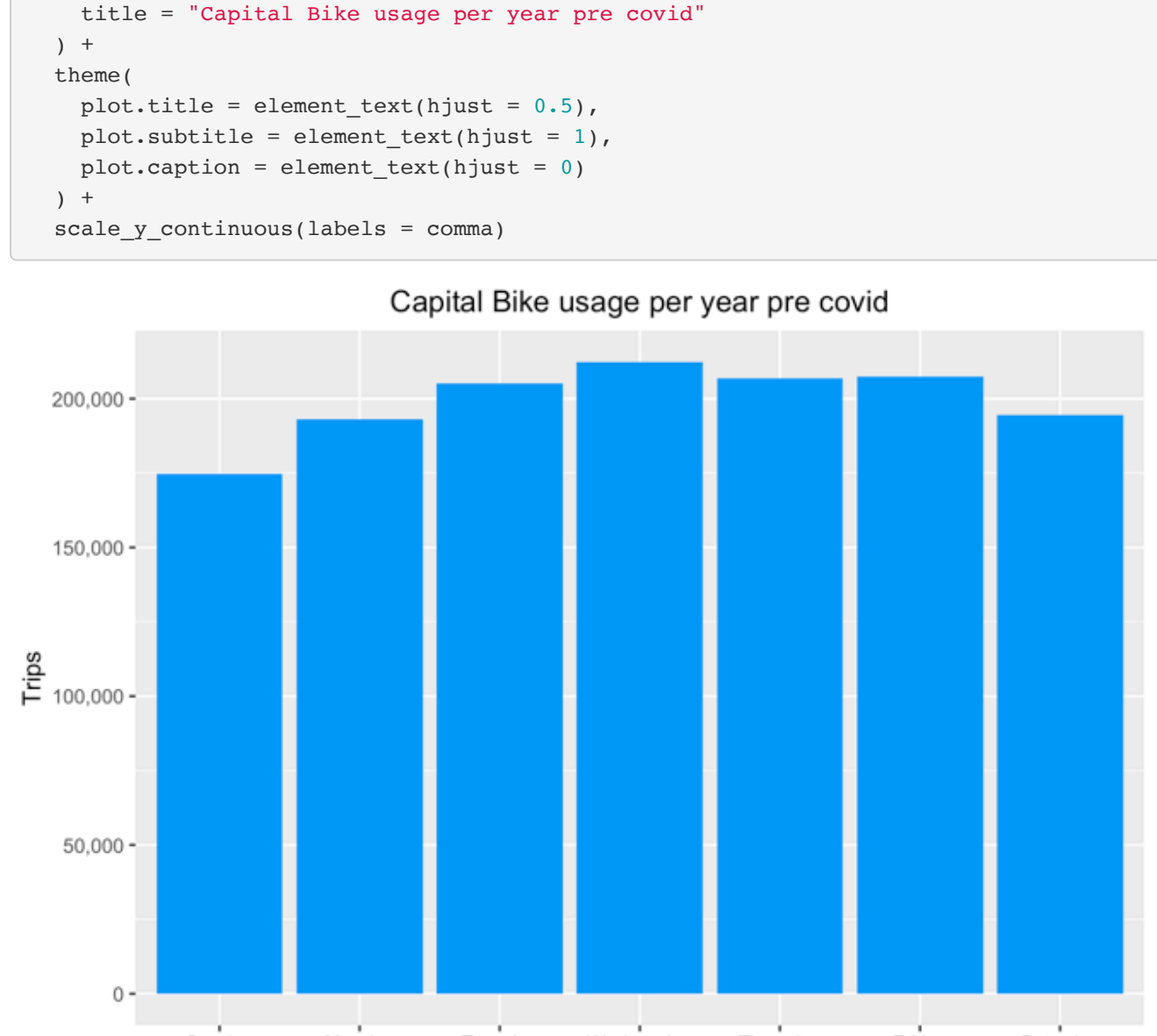
```r
df_seasonal_grouped %>%
  ggplot( aes(x = Season, y = Trips, fill=Workday)) +
  geom_boxplot()  +ggtitle("Bike rentals based on season and workday")+
  coord_flip()
```



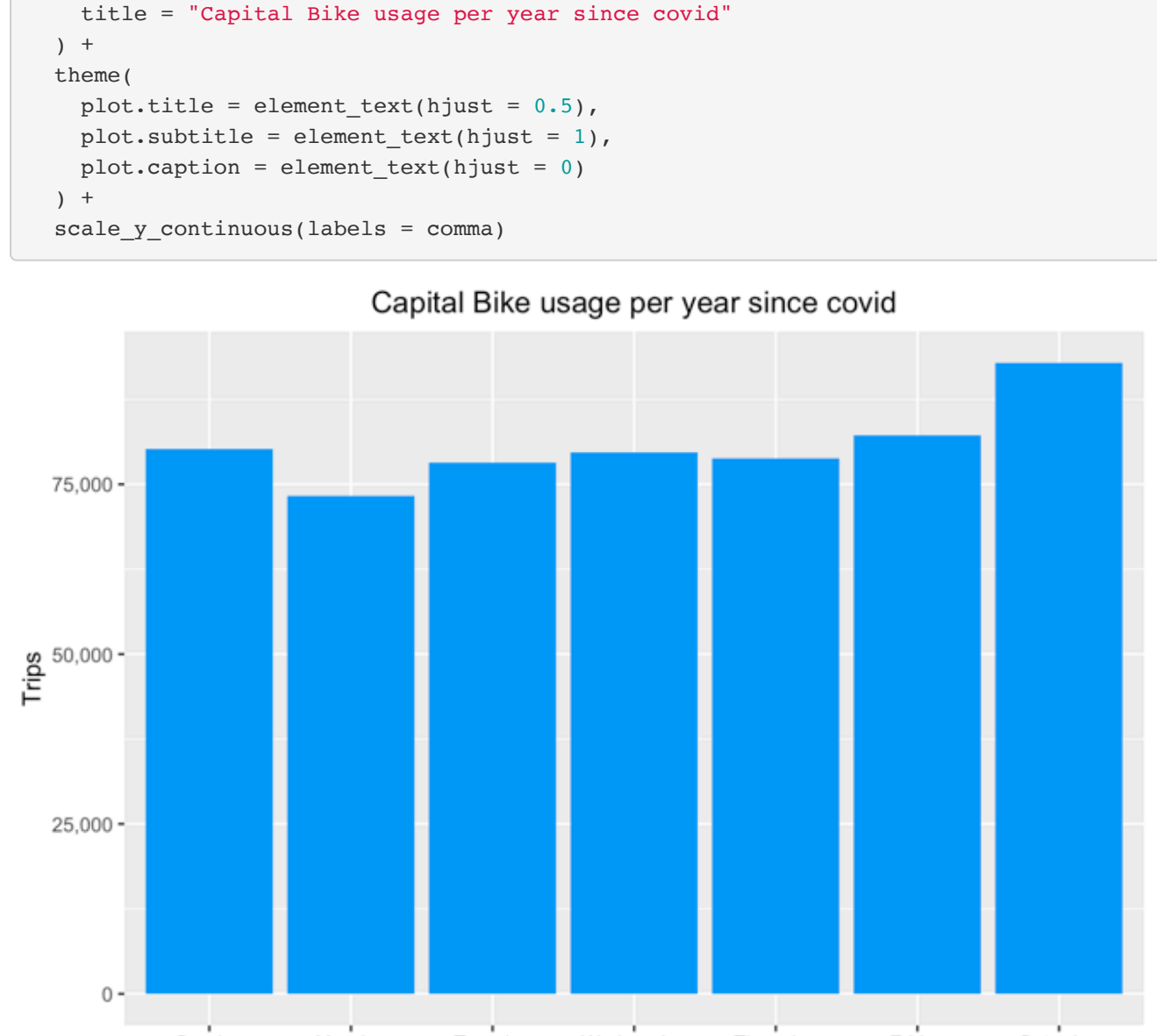
```r
# Rides per day (total)

df_day <- df_seasonal %>% group_by(Day) %>% summarise(Trips = n())



df_day <- df_seasonal %>% group_by(Day) %>% summarise(Trips = n())


df_day$Day <- factor(df_day$Day, levels= c("Sunday", "Monday",
    "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))



df_day <- df_day[order(df_day$Day) , ]


ggplot(df_day, aes(x = Day, y = Trips)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per weekday (2016 onwards)"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma)
```



```r
# Rides per day (pre covid)

df_seasonal$Start.date <- as.Date(df_seasonal$Start.date, format = "%y")


df_seasonal_pre2020 <- df_seasonal[as.Date(df_seasonal$Start.date) < as.Date('2020' , format = "%y") , ]


df_day_pre2020 <- df_seasonal_pre2020 %>% group_by(Day) %>% summarise(Trips = n())



df_day_pre2020 <- df_seasonal_pre2020 %>% group_by(Day) %>% summarise(Trips = n())


df_day_pre2020$Day <- factor(df_day_pre2020$Day, levels= c("Sunday", "Monday",
    "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))



df_day_pre2020 <- df_day_pre2020[order(df_day_pre2020$Day) , ]


ggplot(df_day_pre2020, aes(x = Day, y = Trips)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per year pre covid"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma)
```



```r
# Rides per day (post covid)


df_seasonal_post2020 <- df_seasonal[as.Date(df_seasonal$Start.date) >= as.Date('2020' , format = "%y") , ]



df_seasonal_post2020 <- df_seasonal_post2020 %>% group_by(Day) %>% summarise(Trips = n())




df_seasonal_post2020$Day <- factor(df_seasonal_post2020$Day, levels= c("Sunday", "Monday",
    "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))



df_seasonal_post2020 <- df_seasonal_post2020[order(df_seasonal_post2020$Day) , ]


ggplot(df_seasonal_post2020, aes(x = Day, y = Trips)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per year since covid"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma)
```

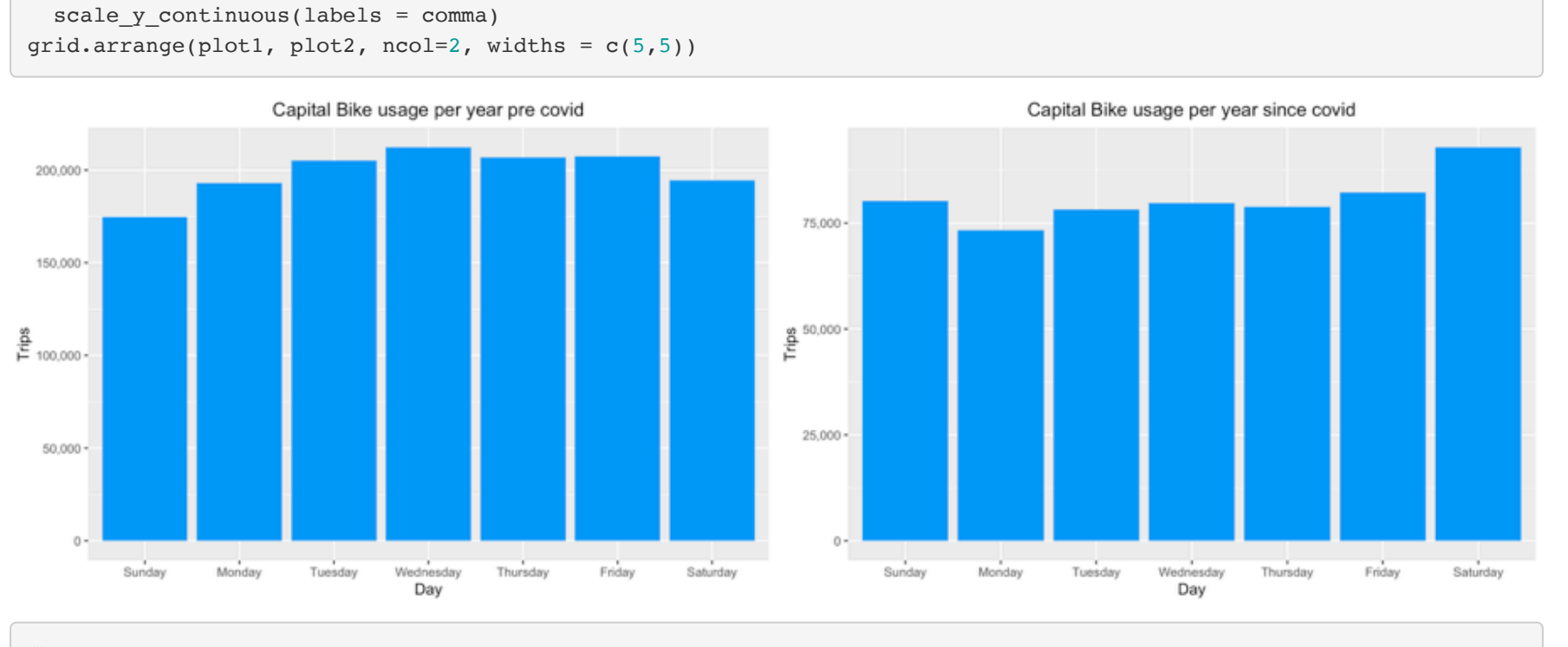


```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
plot1 <- ggplot(df_day_pre2020, aes(x = Day, y = Trips)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per year pre covid"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma)

plot2 <- ggplot(df_seasonal_post2020, aes(x = Day, y = Trips)) +
  geom_col(fill = "#0099f9") +
  labs(
    title = "Capital Bike usage per year since covid"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 1),
    plot.caption = element_text(hjust = 0)
  ) +
  scale_y_continuous(labels = comma)
grid.arrange(plot1, plot2, ncol=2, widths = c(5,5))
```



```r
# Distribution of rides per hour of day
```