STAA57_Project

Muhammad Enrizky Brillian_1009713712

2023-04-04

1. Merging Datasets from multiple CSVs

```
# loading libraries
library(tidyverse)
library(dplyr)
#Setting the working Directory
setwd("~/STAA57_Project")
# Create empty data frame
bikeshare_data <- data.frame()
# Loop through all CSV files in folder
for (file in list.files(pattern=".csv")) {
  temp_data <- read.csv(file, header=TRUE)  # Read in CSV file
  bikeshare_data <- rbind(bikeshare_data, temp_data)  # Bind data to existing data frame
  rm(temp_data)  # Remove original data frame to free up memory
}
```

2. Description of the Dataset

```
names(bikeshare_data)
```

```
## [1] "trip_id"               "trip_duration_seconds" "from_station_id"
## [4] "trip_start_time"       "from_station_name"     "trip_stop_time"
## [7] "to_station_id"         "to_station_name"       "user_type"
```

1. Trip ID: A unique identifier for each trip
2. Trip Duration (Seconds): The length of the trip in seconds
3. From Station ID: The unique identifier for the starting station of the trip
4. Trip Start Time: The date and time the trip began
5. From Station Name: The name of the starting station for the trip
6. Trip Stop Time: The date and time the trip ended
7. To Station ID: The unique identifier for the ending station of the trip
8. Bike ID: The name of the ending station for the trip
9. User Type: A categorical variable indicating whether the user is a “Annual Member” (annual pass holder of the bikeshare program) or a “Casual Member”(24 or 72 hour pass holders of the bikeshare program)

3. The Background of the Data

The bikeshare data for 2018 was collected by the City of Toronto, specifically by the Transportation Services division. The data was collected in the context of a bikeshare program called Bike Share Toronto, which is a public bicycle sharing system in Toronto, Canada. Bike Share Toronto provides access to bicycles at various stations throughout the city, allowing users to rent a bike for short trips and return it to any station within the system. The data collected includes information about bike trips, such as trip duration, start and end station, and user type, and other variables.This data is publicly available through the City of Toronto's Open Data portal for research and analysis purposes.

The dataset can be used for analysis and insights into the usage patterns and trends of the bikeshare program in Toronto during the year 2018. It is typically used by researchers, policymakers, and urban planners to analyze bike usage patterns, evaluate the effectiveness of the bikeshare program, and inform transportation planning and policy decisions. Overall, the bikeshare data for 2018 is a valuable resource for investigating bike usage patterns, user behavior, and system performance in Toronto, and can contribute to evidence-based decision making in transportation planning and policy.

4. What is the over all research question that you are trying to answer in your report

As researchers, our overall research question is to investigate the usage patterns and trends of bikeshare data in Toronto for the year 2018. We aim to analyze various variables, such as trip duration, trip start time, user type, and station information, to gain insights into how the bikeshare system was utilized by different user groups, and to identify any patterns or trends that may emerge from the data. Our research question may include exploring factors affecting bikeshare usage, identifying popular trip routes or stations, understanding user behavior, and assessing the impact of weather or other external factors on bikeshare ridership. By conducting a comprehensive analysis of the bikeshare data, we aim to provide valuable insights and findings that can contribute to a deeper understanding of bikeshare usage in Toronto in 2018.

5. Tables

5.1.1 Trip Duration (in Seconds) Summary

```
# Load required libraries
library(dplyr)
library(knitr)
summary_stats <- bikeshare_data %>%
  summarise(
    Mean_Trip_Duration_Seconds = mean(trip_duration_seconds),
    Median_Trip_Duration_Seconds = median(trip_duration_seconds),
    Mode_Trip_Duration_Seconds = as.numeric(names(table(trip_duration_seconds)
                                                  [table(trip_duration_seconds) ==
                                                     max(table(trip_duration_seconds))])),
    SD_Trip_Duration_Seconds = sd(trip_duration_seconds),
    Q1_Trip_Duration_Seconds = quantile(trip_duration_seconds, 0.25),
    Q3_Trip_Duration_Seconds = quantile(trip_duration_seconds, 0.75),
  )

kable(summary_stats,
      caption = "Trip Duration (in Seconds) Summary",
      col.names = c( "Mean", "Median", "Mode", "SD", "Q1", "Q3"),
```

```
      align = c("l", "c", "c", "c", "c", "c")
)
```

Table 1: Trip Duration (in Seconds) Summary

Mean	Median	Mode	SD	Q1	Q3
962.976	670	450	1595.53	422	1051

5.1.2 Trip Duration (in Seconds) Summary By User Type

```
library(dplyr)

# Create summary statistics of trip duration by user type
trip_duration_summary <- bikeshare_data %>%
  group_by(user_type) %>%
  summarise(
    Number_Trips = n(),
    Average_duration = mean(trip_duration_seconds),
    Median_duration = median(trip_duration_seconds),
    Min_duration = min(trip_duration_seconds),
    Max_duration = max(trip_duration_seconds)
  )

# Display the summary table
knitr::kable(
  trip_duration_summary,
  caption = "Summary of trip duration (in seconds) by user type",
  align=c("l","c","c","c","c")
)
```

Table 2: Summary of trip duration (in seconds) by user type

user_type	Number_Trips	Average_duration	Median_duration	Min_duration	Max_duration
Annual Member	1572980	725.0167	600	60	55077
Casual Member	349975	2032.4958	1220	60	54971

5.2 Table of Total Trips by Month for each user type

```
# Load required libraries
library(dplyr)
library(lubridate)
# Convert trip_start_time to a datetime object
bikeshare_data$trip_start_time <- strptime(bikeshare_data$trip_start_time,
                                           format = "%m/%d/%Y %H:%M")

# Extract month from trip_start_time
bikeshare_data$trip_month <- month(bikeshare_data$trip_start_time, label = TRUE)
```

```r
# Create a table of trip count by month
trip_count_by_month <- bikeshare_data %>%
  na.omit() %>%
  group_by(trip_month,user_type) %>%
  summarize(trip_count = n()) %>%
  pivot_wider(names_from = user_type, values_from = trip_count)

# Print the table
library(knitr)
kable(caption = "Table of Total Trips by Month for each user type",
      trip_count_by_month)
```

Table 3: Table of Total Trips by Month for each user type

trip_month	Annual Member	Casual Member
Jan	42469	1390
Feb	47276	2455
Mar	78564	6405
Apr	82194	12589
May	160989	51761
Jun	186463	64374
Jul	215835	70481
Aug	209770	71449
Sep	207789	47212
Oct	160326	15553
Nov	100675	3612
Dec	80630	2694

5.3 Table of Trip Count by Day

```r
# Load required libraries
library(dplyr)
library(lubridate)

# Extract month from trip_start_time
bikeshare_data$trip_day <- wday(bikeshare_data$trip_start_time, label = TRUE,abbr = F)

# Create a table of trip count by month
trip_count_by_day <- bikeshare_data %>%
  group_by(trip_day) %>%
  reframe(trip_count = n())

# Print the table
kable(caption = "Table of Trip Count by Day",
      trip_count_by_day)
```

Table 4: Table of Trip Count by Day

trip_day	trip_count
Sunday	222205
Monday	267284
Tuesday	290658

trip_day	trip_count
Wednesday	314261
Thursday	303873
Friday	292299
Saturday	232375

5.4 The Top 10 Most Popular Stations

```
# Group the data by starting station and count the number of trips
starting_stations <- bikeshare_data %>%
  group_by(from_station_name) %>%
  summarize(num_trips = n()) %>%
  arrange(desc(num_trips))

# Select the top 10 most popular starting stations
top_starting_stations <- head(starting_stations, 10)

# Create the table
knitr::kable(top_starting_stations, caption = "Top 10 most popular starting stations")
```

Table 5: Top 10 most popular starting stations

from_station_name	num_trips
York St / Queens Quay W	24017
Bay St / Queens Quay W (Ferry Terminal)	22743
Union Station	19869
Bay St / Wellesley St W	19184
Sherbourne St / Wellesley St E	19131
Front St W / Blue Jays Way	17282
Princess St / Adelaide St E	17089
Dundas St W / Yonge St	17054
Bay St / College St (East Side)	16965
Bathurst St/Queens Quay(Billy Bishop Airport)	16794

5.5 Average trip duration by hour of day and user type

```
library(dplyr)
library(lubridate)

bikeshare_data %>%
  mutate(trip_start_hour = hour((trip_start_time))) %>%
  group_by(user_type, trip_start_hour) %>%
  summarize(avg_duration_seconds = mean(trip_duration_seconds)) %>%
  pivot_wider(names_from = user_type, values_from = avg_duration_seconds) %>%
  mutate_all(~ round(., digits = 2)) %>%
  knitr::kable(caption = "Average trip duration by hour of day and user type")
```

Table 6: Average trip duration by hour of day and user type

trip_start_hour	Annual Member	Casual Member
0	682.33	1773.73
1	699.77	1819.08
2	689.93	1884.11
3	691.84	1863.71
4	765.28	1742.38
5	662.48	1787.04
6	656.30	1490.34
7	695.73	1494.91
8	724.24	1446.85
9	662.16	1975.01
10	692.09	2334.44
11	725.18	2377.54
12	696.13	2325.43
13	719.42	2362.41
14	751.10	2311.90
15	741.37	2260.44
16	738.49	2067.92
17	759.00	1900.17
18	751.92	1856.47
19	743.64	1826.21
20	734.82	1720.60
21	729.67	1607.26
22	713.16	1576.33
23	693.11	1650.28

5.6 Trip duration More than 30 minutes and More than 45 minutes

```
# Create a new column to indicate whether the trip duration was greater than 1800 seconds (30 minutes)
bikeshare_data <- bikeshare_data %>%
  mutate(greater_than_1800 = ifelse(trip_duration_seconds > 1800, 1, 0))

# Create a new column to indicate whether the trip duration was greater than 2700 seconds (45 minutes)
bikeshare_data <- bikeshare_data %>%
  mutate(greater_than_2700 = ifelse(trip_duration_seconds > 2700, 1, 0))

# Create a table with the number of users for each duration threshold
duration_table <- bikeshare_data %>%
  group_by(greater_than_1800, greater_than_2700) %>%
  summarize(num_users = n())
```

```
## `summarise()` has grouped output by 'greater_than_1800'. You can override using
## the `.groups` argument.
```

```
# Print the table
print(duration_table)
```

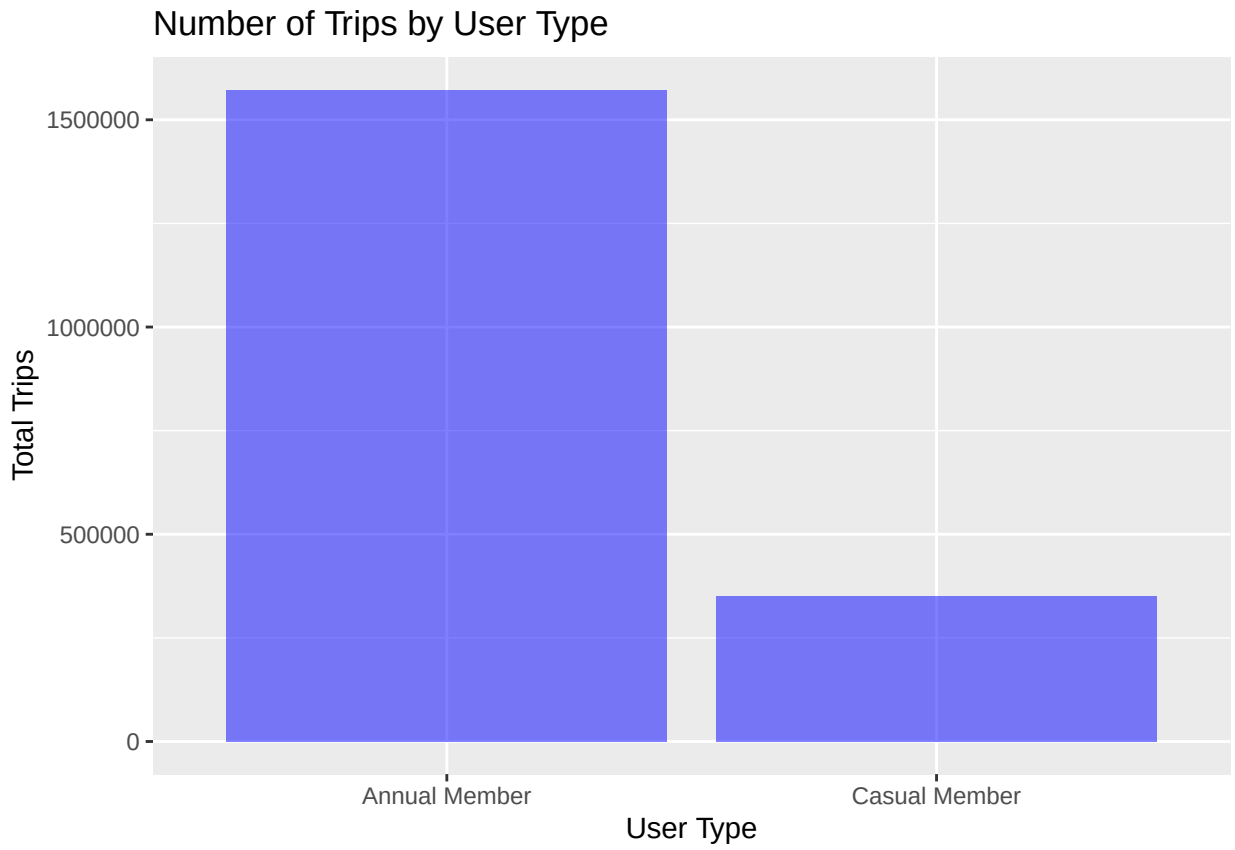
```
## # A tibble: 3 x 3
## # Groups:   greater_than_1800 [2]
##   greater_than_1800 greater_than_2700 num_users
##               <dbl>             <dbl>     <int>
```

```
## 1                0                0   1802516
## 2                1                0     58284
## 3                1                1     62155
```

6. Graphs

6.1 Bar chart of the number of trips by user type:

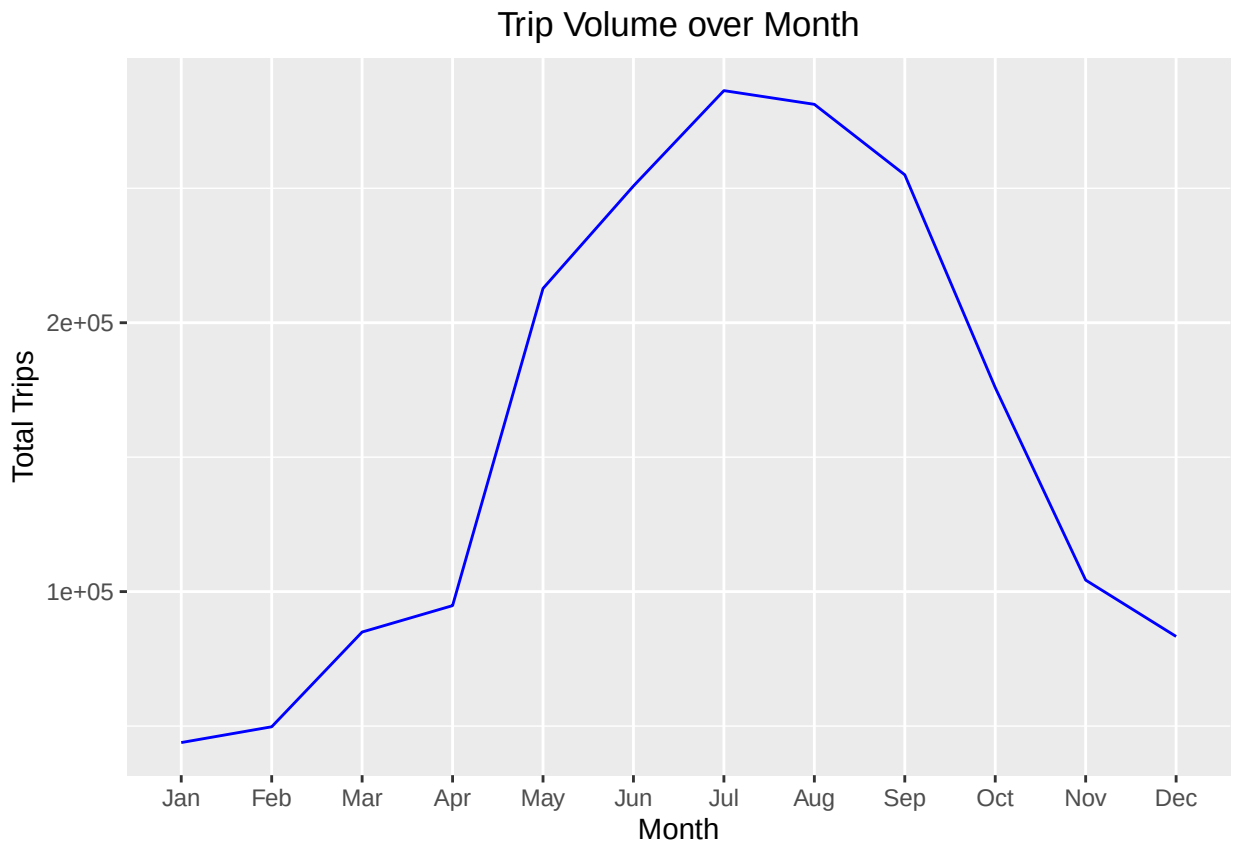
```r
ggplot(bikeshare_data, aes(x = user_type)) +
  geom_bar(fill = "blue", alpha = 0.5) +
  labs(title = "Number of Trips by User Type",
       x = "User Type",
       y = "Total Trips")
```



6.2 Line plot of trip volume over time

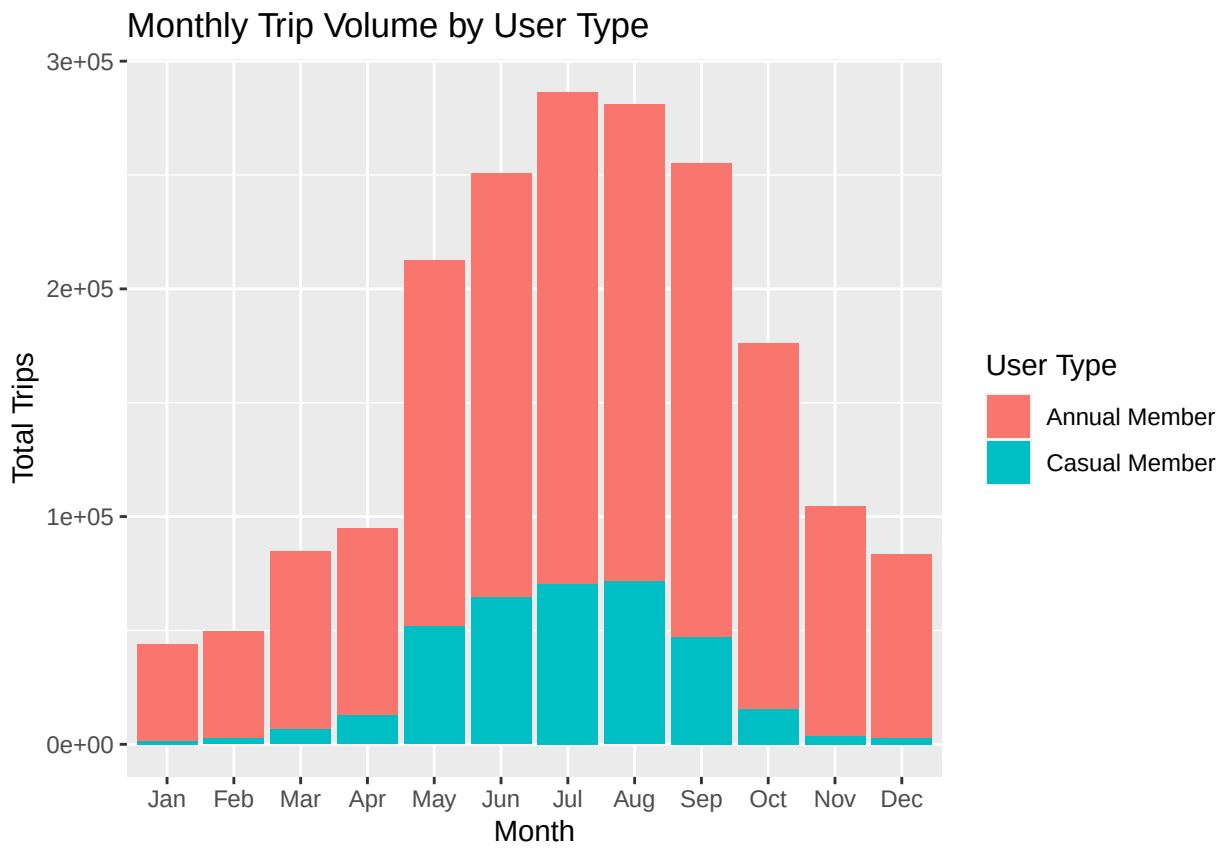
```r
library(ggplot2)

# count the number of trips for each month
trip_count_by_month <- bikeshare_data %>%
  group_by(trip_month) %>%
  summarize(total_trips = n())

# plot the line graph
ggplot(trip_count_by_month, aes(x = trip_month, y = total_trips, group = 1)) +
```

```
  geom_line(color = "blue") +
  labs(x = "Month", y = "Total Trips", title = "Trip Volume over Month") +
  theme(plot.title = element_text(hjust = 0.5))
```



6.3 Stacked bar plot of trip volume by user type and month

```
library(ggplot2)

# group the data by month and user type and calculate the total number of trips
monthly_user_trips <- bikeshare_data %>%
  group_by(trip_month, user_type) %>%
  summarize(total_trips = n())

# create the stacked bar plot
ggplot(monthly_user_trips, aes(x = trip_month, y = total_trips, fill = user_type)) +
  geom_bar(stat = "identity") +
  labs(title = "Monthly Trip Volume by User Type",
       x = "Month",
       y = "Total Trips",
       fill = "User Type")
```



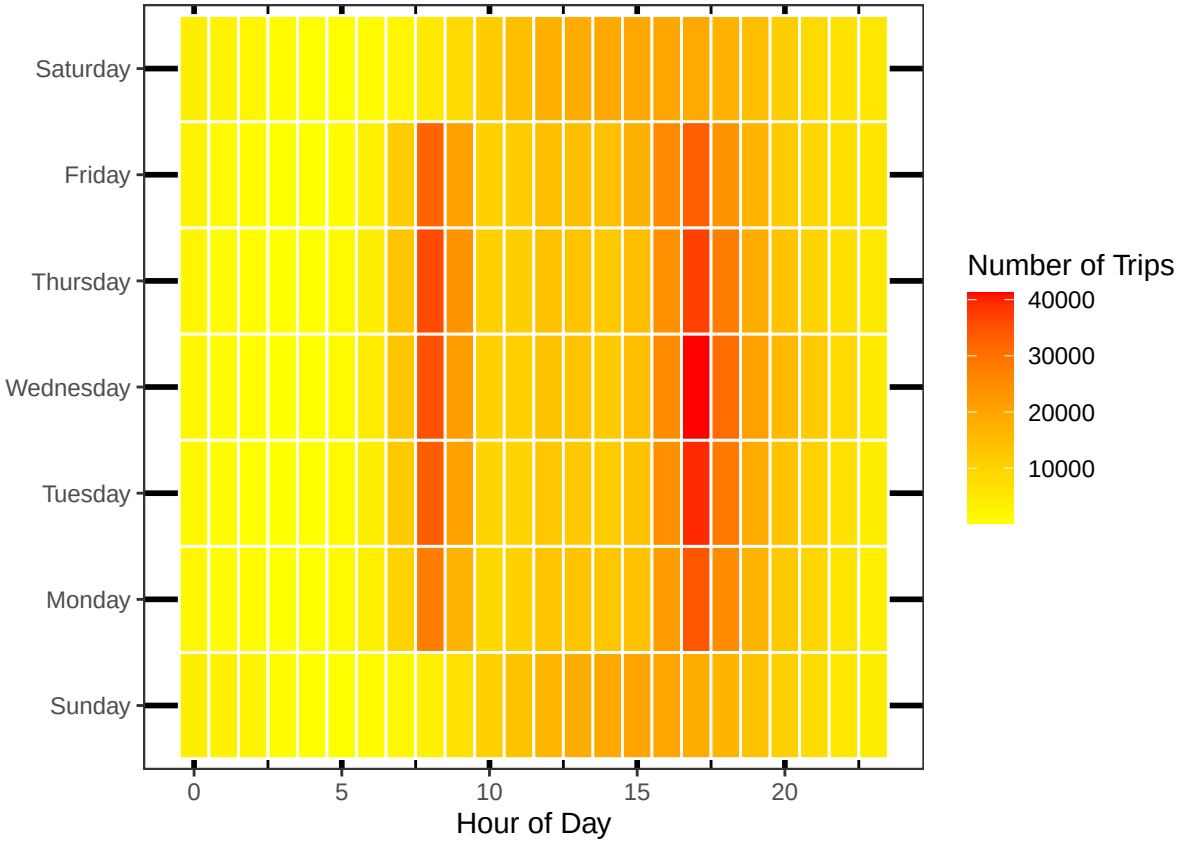
6.4 Heatmap of trips by day of the week

```
library(ggplot2)
library(dplyr)

bikeshare_data <- bikeshare_data %>%
  mutate(trip_start_hour = hour(trip_start_time))

# Aggregate the data by day of the week and hour
heatmap_data <- bikeshare_data %>%
  group_by(trip_start_hour, trip_day) %>%
  summarize(trip_count = n())

# Create the heatmap
ggplot(heatmap_data, aes(x=trip_start_hour, y=trip_day, fill = trip_count)) +
  geom_tile(color = "white", size = 0.5) +
  scale_fill_gradient(low = "yellow", high = "red") +
  labs(x = "Hour of Day", y = "", fill = "Number of Trips") +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5),
        panel.grid.major = element_line(color = "black", size = 1),
        panel.grid.minor = element_line(color = "black", size = 0.5))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
```

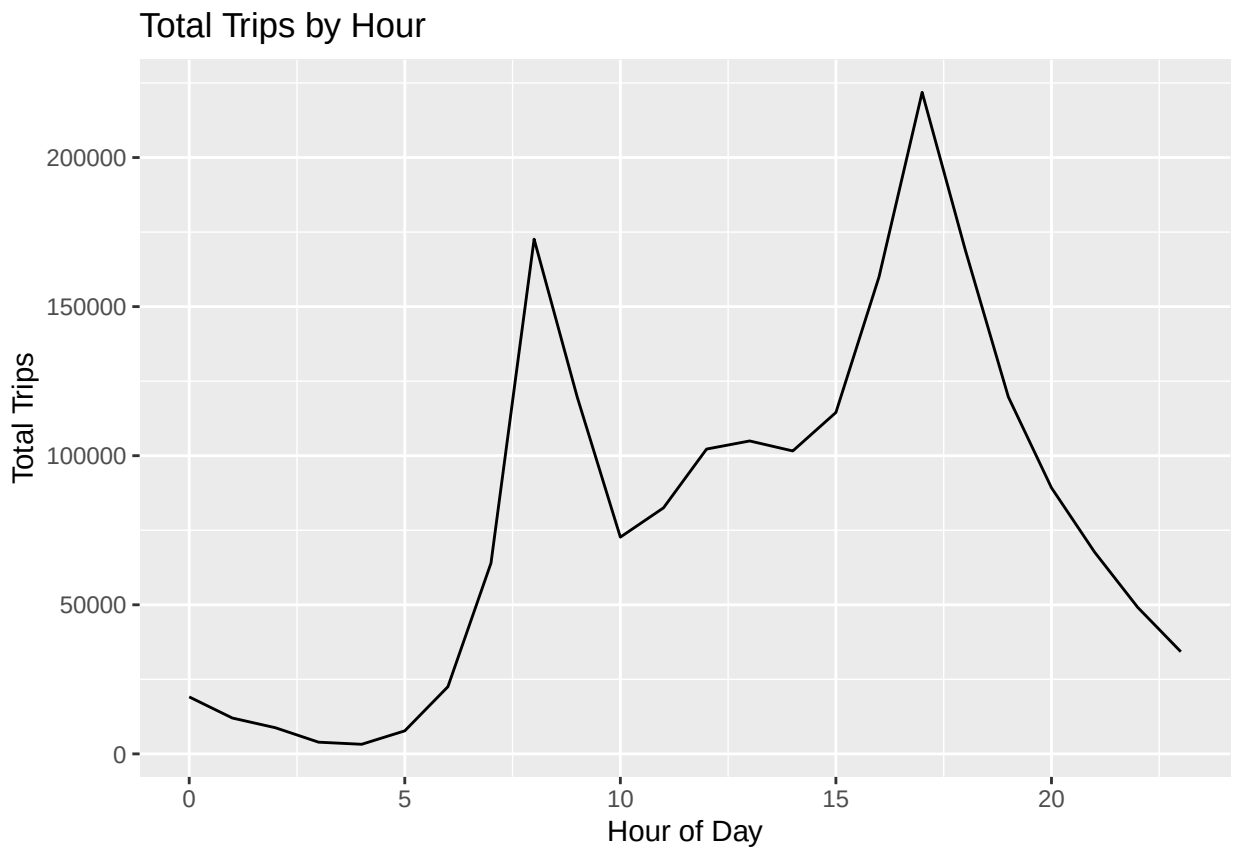
```
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



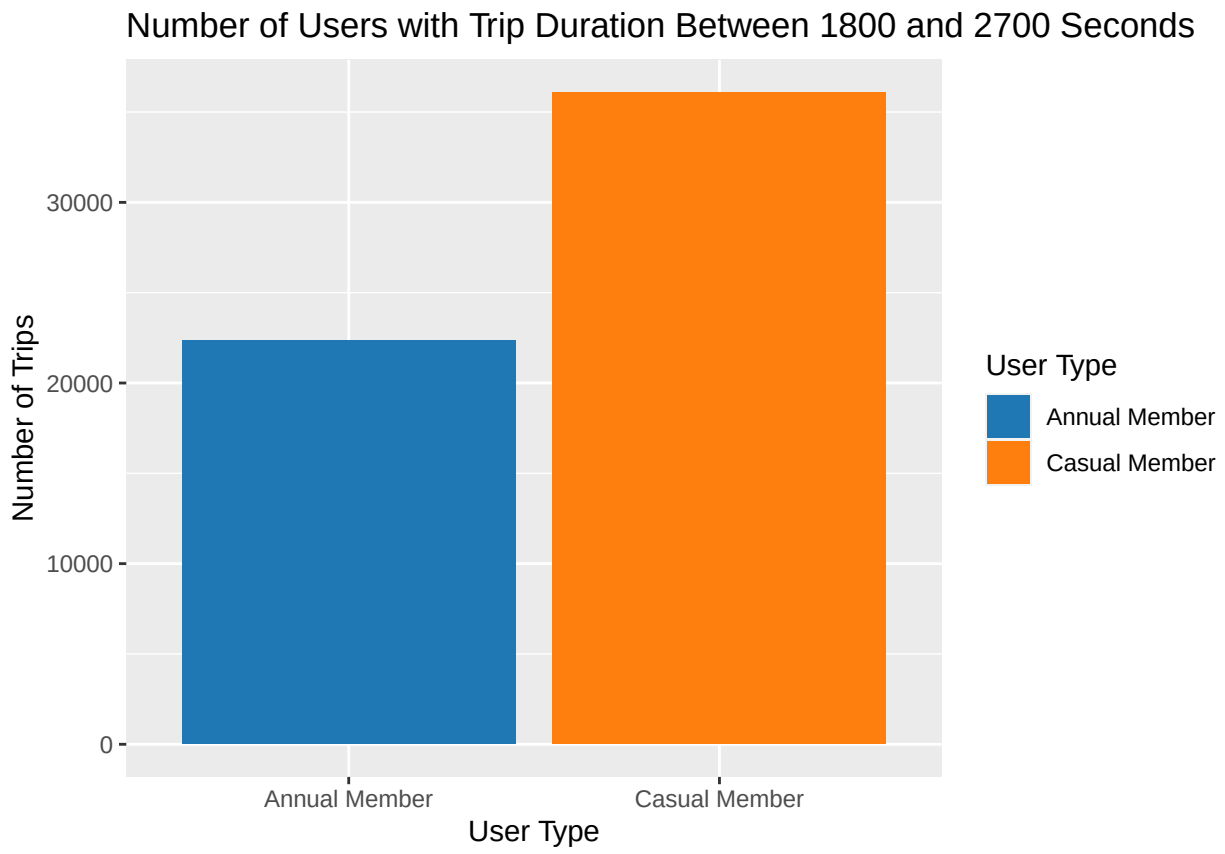
6.5 Line Chart of Total Trips by Hour

```
# summarize data by hour of day
hourly_trips <- bikeshare_data %>%
  group_by(trip_start_hour) %>%
  summarize(total_trips = n())

# create line chart of total trips by hour
ggplot(hourly_trips, aes(x = trip_start_hour, y = total_trips)) +
  geom_line() +
  ggtitle("Total Trips by Hour") +
  xlab("Hour of Day") + ylab("Total Trips")
```



6.6 Trip duration More than 30 minutes and More than 45 minutes

```
library(ggplot2)

# Filter the data for trips longer than 1800 and 2700 seconds
long_trips <- bikeshare_data[bikeshare_data$trip_duration_seconds >= 1800 & bikeshare_data$trip_duratio

# Group the data by user_type
grouped_data <- aggregate(long_trips$trip_duration_seconds, by = list(long_trips$user_type), FUN = leng

# Rename the columns
colnames(grouped_data) <- c("User Type", "Number of Trips")

# Create the chart
ggplot(grouped_data, aes(x = `User Type`, y = `Number of Trips`, fill = `User Type`)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("#1F77B4", "#FF7F0E")) +
  labs(x = "User Type", y = "Number of Trips", title = "Number of Users with Trip Duration Between 1800
```



7. Hypothesis Testing

7.1 What is the average trip duration for casual members vs. annual members?

We would like to determine if there is a statistically significant difference in the mean trip duration between two groups: casual members and annual members.

To formally test this hypothesis, we would start by defining the null hypothesis (H_0) and the alternative hypothesis (H_1):

- H_0: **The average trip duration for casual members (μ_c) is equal to the average trip duration for annual members** (μ_a), i.e., $\mu_c = \mu_a$
- H_1: **The average trip duration for casual members (μ_c) is not equal to the average trip duration for annual members** (μ_a), i.e., $\mu_c \neq \mu_a$

We can use a two-sample t-test to determine if there is a significant difference in the mean trip duration between these two groups. In this case, the two samples are the trip durations for casual members and annual members.

The t-test will provide us with a test statistic and a p-value. If the p-value is less than our chosen significance level (usually 0.05), we reject the null hypothesis and conclude that there is a significant difference in the mean trip duration between casual members and annual members. If the p-value is greater than our chosen significance level, we fail to reject the null hypothesis and conclude that there is not enough evidence to support the claim that there is a difference in the mean trip duration between these two groups.

We can also calculate confidence intervals to estimate the true difference in the mean trip duration between

the two groups. If the confidence interval does not contain zero, this would provide additional evidence that there is a significant difference in the mean trip duration between casual members and annual members.

```
# Perform the t-test
t.test(trip_duration_seconds ~ user_type, data = bikeshare_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  trip_duration_seconds by user_type
## t = -250.48, df = 361923, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1317.710 -1297.248
## sample estimates:
## mean in group Annual Member mean in group Casual Member
##                    725.0167                   2032.4958
```

Based on the Welch Two Sample t-test result, we can see that the p-value is less than the significance level of 0.05. Therefore, we reject the null hypothesis that there is no difference in the average trip duration between annual members and casual members. This means that there is strong evidence to suggest that there is a significant difference in the average trip duration between the two groups. Additionally, we can see from the sample estimates that the mean trip duration for casual members (2032.4958 seconds) is significantly higher than the mean trip duration for annual members (725.0167 seconds). The 95% confidence interval (-1317.710, -1297.248) does not include zero, which also supports the alternative hypothesis that there is a significant difference between the average trip duration for casual members and annual members..

7.2 trip duration between trips taken on weekdays versus weekends.

We would like to determine if there is a significant difference in trip duration between trips taken on weekdays versus weekends.

Hypothesis:

- H_0: **The average trip duration on weekdays (μ_d) is equal to the average trip duration on weekend** (μ_e), i.e., $\mu_d = \mu_e$
- H_1: **The average trip duration on weekdays (μ_d) is not equal to the average trip duration on weekend** (μ_e), i.e., $\mu_d \neq \mu_e$

To test this hypothesis, we can use a two-sample t-test to compare the mean trip duration of weekday trips with the mean trip duration of weekend trips. We can split the data into two groups: weekday trips (Monday to Friday) and weekend trips (Saturday and Sunday).

We can then calculate the p-value associated with the t-statistic, which is the probability of observing a t-value as extreme or more extreme than the one calculated, assuming the null hypothesis is true. If the p-value is less than our significance level (typically 0.05), we can reject the null hypothesis and conclude that there is a significant difference in trip duration between weekday and weekend trips.

We can also calculate a confidence interval for the difference between the means to estimate the range of values in which the true difference between the means is likely to fall.

Overall, this hypothesis testing approach allows us to determine whether there is a statistically significant difference in trip duration between trips taken on weekdays versus weekends, and provides us with an estimate of the magnitude of this difference.

```
# Subset data to separate trips taken on weekdays and weekends
weekdays <- subset(bikeshare_data, trip_day %in% c("Monday", "Tuesday", "Wednesday",
```

```
                                                    "Thursday", "Friday"))
weekends <- subset(bikeshare_data, trip_day %in% c("Saturday", "Sunday"))
# Conduct two-sample t-test
t.test(x=weekdays$trip_duration_seconds, y=weekends$trip_duration_seconds, var.equal = FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  weekdays$trip_duration_seconds and weekends$trip_duration_seconds
## t = -112.86, df = 579131, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -383.8696 -370.7647
## sample estimates:
## mean of x mean of y
##  873.7795 1251.0967
```

The Welch Two Sample t-test shows a significant difference between the mean trip durations of weekdays and weekends. The p-value (2.2e-16) is less than the significance level of 0.05, indicating strong evidence against the null hypothesis of no difference. The confidence interval (-383.8696, -370.7647) also does not include 0, further supporting the conclusion that there is a significant difference between the two groups. Therefore, we can conclude that there is a significant difference in trip duration between trips taken on weekdays versus weekends. Specifically, the mean trip duration on weekends (1251.0967 seconds) is longer than that on weekdays (873.7795 seconds).

8. Bootstrapping

8.1 Estimating the average trip duration for casual members

We could use bootstrapping to estimate the sampling distribution of the mean and compute a confidence interval.

```
library(boot)


# Subset data for casual members
casual_data <- bikeshare_data %>% filter(user_type == "Casual Member")

# Define a function to calculate the average trip duration
mean_duration <- function(data) {
  mean(data$trip_duration_seconds)
}

# Use bootstrapping to estimate the mean trip duration
library(dplyr)
set.seed(123)
#There are more than one million observations, so
# if n_boot is large, then R would crash again
n_boot <- 100
bootstrap_means <- replicate(n_boot,
                             sample_n(casual_data, nrow(casual_data), replace=TRUE) %>%
                               mean_duration())
```

```r
# Calculate the 95% confidence interval
ci <- quantile(bootstrap_means, c(0.025, 0.975))

# Print the results
cat("Mean trip duration for casual members:", mean(casual_data$trip_duration_seconds), "\n")
```

```
## Mean trip duration for casual members: 2032.496
```

```r
cat("95% confidence interval:","( ", ci[1], "-", ci[2]," )", "\n")
```

```
## 95% confidence interval: (  2022.838 - 2042.171  )
```

Based on the result, the mean trip duration for casual members is 2032.496 seconds. The 95% confidence interval is (2022.838 - 2042.171) seconds. This means that we are 95% confident that the true mean trip duration for casual members falls within this range.

8.2 Estimating the average trip duration for Annual members

We could use bootstrapping to estimate the sampling distribution of the mean and compute a confidence interval.

```r
library(boot)


# Subset data for casual members
annual_data <- bikeshare_data %>% filter(user_type == "Annual Member")

# Define a function to calculate the average trip duration
mean_duration <- function(data) {
  mean(data$trip_duration_seconds)
}

# Use bootstrapping to estimate the mean trip duration
library(dplyr)
set.seed(123)
#There are almost two millions observations, so
# if n_boot is large, then R would crash again
n_boot <- 100
bootstrap_means <- replicate(n_boot,
                             sample_n(annual_data, nrow(annual_data), replace=TRUE) %>%
                               mean_duration())

# Calculate the 95% confidence interval
ci <- quantile(bootstrap_means, c(0.025, 0.975))

# Print the results
cat("Mean trip duration for annual members:", mean(annual_data$trip_duration_seconds), "\n")
```

```
## Mean trip duration for annual members: 725.0167
```

```r
cat("95% confidence interval:","( ", ci[1], "-", ci[2]," )", "\n")
```

```
## 95% confidence interval: (  723.9105 - 726.033  )
```

Based on the result, the mean trip duration for annual members is 2032.496 seconds. The 95% confidence interval is (2022.838 - 2042.171) seconds. This means that we are 95% confident

that the true mean trip duration for annual members falls within this range.

9. Non-Linear Regression Analysis

Analyzing the relationship between the number of trips and hour of the day.

In this case, we would treat the hour of the day as a continuous variable and fit a non-linear regression model with the number of trips as the dependent variable and the hour of the day as the independent variable. The resulting model would give us information on the direction and strength of the relationship between the hour of the day and the number of trips.

```
# Convert the "trip_start_hour" column from time (hour) to numeric
bikeshare_data$start_hour_numeric <- as.numeric(bikeshare_data$trip_start_hour)

# Group the data by hour and count the number of trips for each hour
trips_by_hour_numeric <- bikeshare_data %>%
  group_by(start_hour_numeric) %>%
  summarise(num_trips = n())

# Use linear regression to analyze the relationship between the number of trips and the hour of the day
model <- lm(num_trips ~ start_hour_numeric+I(start_hour_numeric^2)+I(start_hour_numeric^3), data = trips

# Print the summary of the model
summary(model)
```

```
##
## Call:
## lm(formula = num_trips ~ start_hour_numeric + I(start_hour_numeric^2) +
##     I(start_hour_numeric^3), data = trips_by_hour_numeric)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -38269 -21133  -9337   9561 103476
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              11345.41   25688.88   0.442  0.66348
## start_hour_numeric       -5051.71    9882.68  -0.511  0.61483
## I(start_hour_numeric^2)   2225.45    1011.04   2.201  0.03964 *
## I(start_hour_numeric^3)    -86.35      28.87  -2.991  0.00721 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36500 on 20 degrees of freedom
## Multiple R-squared:  0.6909, Adjusted R-squared:  0.6445
## F-statistic:  14.9 on 3 and 20 DF,  p-value: 2.501e-05
```

Explanation

The non-linear regression model suggests that there is a significant relationship between the number of bike trips and the hour of the day, and that this relationship is best explained by a third-order polynomial function. **The coefficient of determination (R-squared) of 0.69 indicates that the model explains 69% of the variation in the data.** The p-values for the coefficients of the quadratic and cubic terms are both significant, suggesting that the non-linear terms are needed to better explain the relationship between the number of bike trips and the hour of the day. However, the p-value for the coefficient of the linear term is not

significant, suggesting that there may not be a significant linear relationship between the number of bike trips and the hour of the day.

Intrepreting Regression Parameter

The regression parameters in this case refer to the coefficients of the independent variables in the multiple regression equation.

For this non-linear regression model, we have the following coefficients:

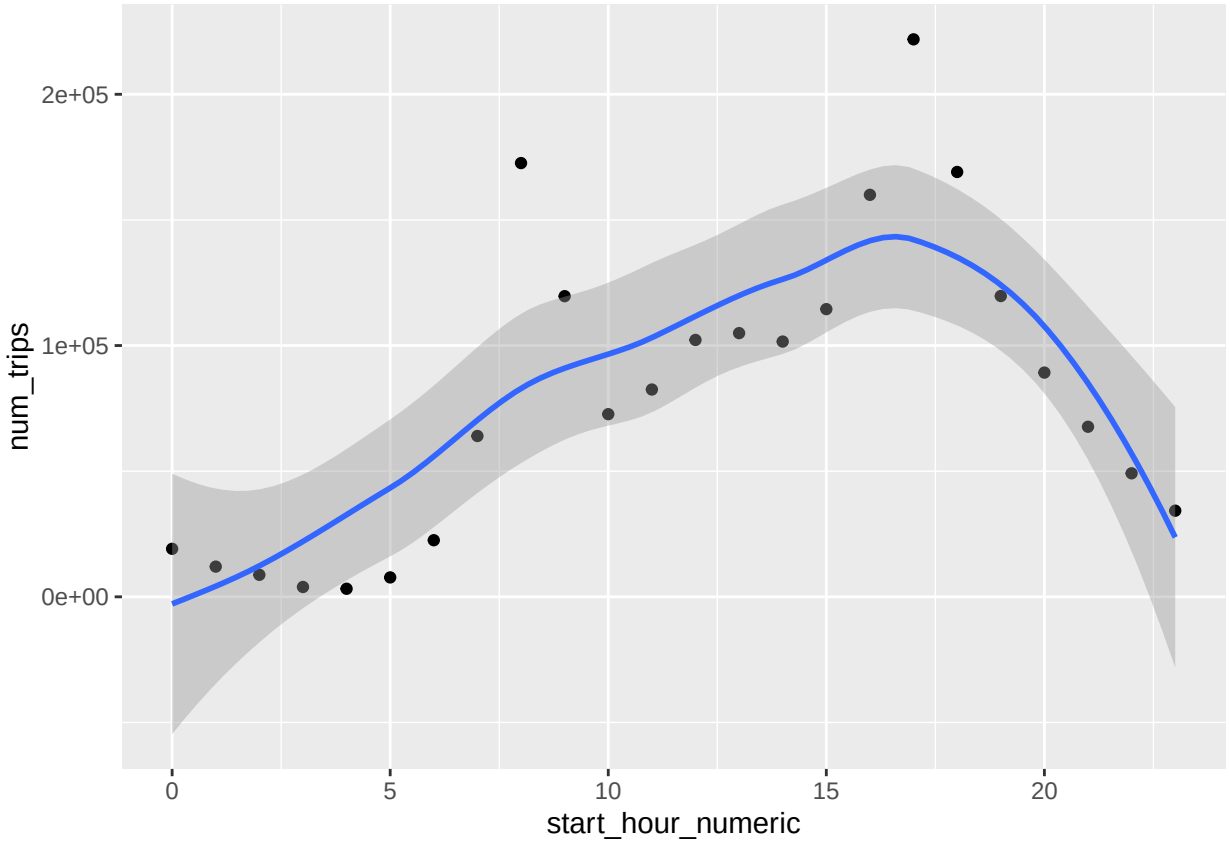
- The intercept coefficient represents the value of the dependent variable (num_trips) when all independent variables are equal to zero. In this case, the intercept coefficient is 11345.41, which means that when the start_hour_numeric, start_hour_numeric squared, and start_hour_numeric cubed are all zero, the expected value of num_trips is 11345.41.
- The coefficient of start_hour_numeric represents the linear effect of the start hour on the number of trips. In this case, the coefficient is negative (-5051.71), which suggests that as the start hour increases by one unit, the expected value of num_trips decreases by 5051.71.
- The coefficient of I(start_hour_numericˆ2) represents the quadratic effect of start hour on the number of trips. In this case, the coefficient is positive (2225.45), which suggests that the relationship between start hour and num_trips is curvilinear, with a maximum value of num_trips occurring at a specific start hour.
- The coefficient of I(start_hour_numericˆ3) represents the cubic effect of start hour on the number of trips. In this case, the coefficient is negative (-86.35), which suggests that the curvilinear relationship between start hour and num_trips is concave down, with the rate of decrease in num_trips slowing down as start hour increases.

Overall, this non-linear regression suggests that the relationship between start hour and the number of trips is not simply linear, but rather a curvilinear relationship with a peak at a specific start hour. Additionally, the cubic term indicates that this curvilinear relationship is concave down, meaning that the rate of decrease in the number of trips as start hour increases slows down as start hour increases.

Overall, this suggests that there is a non-linear relationship between the hour of the day and the number of trips taken, and this relationship is well captured by the polynomial regression model.

Plotting the Result

```
library(ggplot2)

ggplot(trips_by_hour_numeric, aes(x=start_hour_numeric, y=num_trips) ) +
geom_point() +
stat_smooth()
```



10. Cross Validation

Analyzing the Relationship between Trip Duration(in hour) and hour of the day

In this case, we can use cross-validation to analyze the relationship between trip duration and hour of the day in the bikeshare data.

The idea behind this approach is to split the dataset into two parts, with one part used to train the model and the other part used to validate it. We can then measure the accuracy of the model on the validation set and adjust the model as needed to improve its performance.

To use cross-validation to analyze the relationship between trip duration (in hour) and hour of the day, we would start by splitting the dataset into a training set and a validation set. We could then use **Non-linear regression** to build a model that predicts trip duration based on the hour of the day, using the training set. Once we have built the model, we would use the validation set to evaluate its performance.

Our approach is k-fold cross-validation, where the dataset is divided into k equally-sized subsets. We then train the model on k-1 of the subsets and use the remaining subset for validation. We repeat this process k times, so that each subset is used for validation once.

```
# Calculate average trip duration (in hour) by hour of the day
avg_duration <- bikeshare_data %>%
  group_by(start_hour_numeric) %>%
  summarize(avg_trip_duration = mean(trip_duration_seconds/3600))

# Define the model formula
formula <- avg_trip_duration ~ start_hour_numeric +
```

```
  I(start_hour_numeric^2) + I(start_hour_numeric^3)

# Define the number of folds for cross-validation
k <- 10

# Initialize a vector to store the mean squared errors for each fold
cv_mse <- rep(0, k)

# Define the indices for each fold
set.seed(123)
folds <- cut(seq(1, nrow(avg_duration)), breaks = k, labels = FALSE)

# Perform k-fold cross-validation
for (i in 1:k) {
  # Split the data into training and test sets for this fold
  test_indices <- which(folds == i, arr.ind = TRUE)
  test_set <- avg_duration[test_indices, ]
  train_set <- avg_duration[-test_indices, ]

  # Fit the model to the training set
  model <- lm(formula, data = train_set)

  # Use the model to predict the test set
  predictions <- predict(model, newdata = test_set)

  # Calculate the mean squared error for this fold
  mse <- mean((predictions - test_set$avg_trip_duration)^2)

  # Store the mean squared error in the cv_mse vector
  cv_mse[i] <- mse
}

# Calculate the average mean squared error over all folds
avg_mse <- mean(cv_mse)

# Print the average mean squared error over all folds
cat("The Average MSE is:",avg_mse,"\n")
```

```
## The Average MSE is: 0.002516312
```

Conclusion

The average MSE of 0.002516312 is quite small, which suggests that the non-linear regression model is fitting the data quite well. This means that there is likely a relationship between the hour of the day and the average trip duration (in hour), and that this relationship can be captured by the non-linear regression model.

Plotting the Result

```
# Plot the results
ggplot(avg_duration, aes(x = start_hour_numeric, y = avg_trip_duration)) +
  geom_point() +
  stat_smooth() +
  labs(title = "Average Trip Duration by Hour of the Day",
```

```
      x = "Hour of the Day",
      y = "Average Trip Duration (Hour)",
      caption = paste0("Average MSE: ", round(avg_mse, 2)))
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```