Code ▾

Urban Pedals: Unveiling Toronto’s Bike Share Story

Overview

This report contains the code and data for a data analysis project focused on exploring the ridership patterns of the bike-share scheme in Toronto. The analysis is conducted using the R programming language to gain insights into user behaviors, popular routes, and temporal trends.

Data Source

- Ref:
  - https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/ (https://open.toronto.ca/dataset/bike-share-toronto-ridership-data/)

Download and Import data.

- Import libraries.

Hide

```
# load library
library(opendatatoronto)
library(dplyr)
library(tidyr)
library(ggplot2)
```

- Manually download data files from data source.
  - We have 52 files of 5-year data, from 2019 to 2023.
  - List of all data files

Hide

```
# Define path of dataset
DATA_PATH <- "./source/"

# get the paths for each data file.
FILE_PATH_LIST <- list.files(DATA_PATH, pattern = "\\.csv$", full.names = TRUE)
FILE_PATH_LIST
```

```
 [1] "./source/2019-Q1.csv"                      "./source/2019-Q2.csv"                     
 [3] "./source/2019-Q3.csv"                      "./source/2019-Q4.csv"                     
 [5] "./source/2020-01.csv"                      "./source/2020-02.csv"                     
 [7] "./source/2020-03.csv"                      "./source/2020-04.csv"                     
 [9] "./source/2020-05.csv"                      "./source/2020-06.csv"                     
[11] "./source/2020-07.csv"                      "./source/2020-08.csv"                     
[13] "./source/2020-09.csv"                      "./source/2020-10.csv"                     
[15] "./source/2020-11.csv"                      "./source/2020-12.csv"                     
[17] "./source/Bike share ridership 2021-01.csv" "./source/Bike share ridership 2021-02.csv"
[19] "./source/Bike share ridership 2021-03.csv" "./source/Bike share ridership 2021-04.csv"
[21] "./source/Bike share ridership 2021-05.csv" "./source/Bike share ridership 2021-06.csv"
[23] "./source/Bike share ridership 2021-07.csv" "./source/Bike share ridership 2021-08.csv"
[25] "./source/Bike share ridership 2021-09.csv" "./source/Bike share ridership 2021-10.csv"
[27] "./source/Bike share ridership 2021-11.csv" "./source/Bike share ridership 2021-12.csv"
[29] "./source/Bike share ridership 2022-01.csv" "./source/Bike share ridership 2022-02.csv"
[31] "./source/Bike share ridership 2022-03.csv" "./source/Bike share ridership 2022-04.csv"
[33] "./source/Bike share ridership 2022-05.csv" "./source/Bike share ridership 2022-06.csv"
[35] "./source/Bike share ridership 2022-07.csv" "./source/Bike share ridership 2022-08.csv"
[37] "./source/Bike share ridership 2022-09.csv" "./source/Bike share ridership 2022-10.csv"
[39] "./source/Bike share ridership 2022-11.csv" "./source/Bike share ridership 2022-12.csv"
[41] "./source/Bike share ridership 2023-01.csv" "./source/Bike share ridership 2023-02.csv"
[43] "./source/Bike share ridership 2023-03.csv" "./source/Bike share ridership 2023-04.csv"
[45] "./source/Bike share ridership 2023-05.csv" "./source/Bike share ridership 2023-06.csv"
[47] "./source/Bike share ridership 2023-07.csv" "./source/Bike share ridership 2023-08.csv"
[49] "./source/Bike share ridership 2023-09.csv" "./source/Bike share ridership 2023-10.csv"
[51] "./source/Bike share ridership 2023-11.csv" "./source/Bike share ridership 2023-12.csv"
```

- Import data from data files.
    - Check the columns of each csv file for data consisitence.
    - Combine all data into a uniformed dataframe for further anaysis.

Hide

```
# # Apply read.csv, a function to import data from csv file, for each file.
# # Get a list of df
# df_list <- lapply(FILE_PATH_LIST, read.csv)
#
#
# # Union all df by row
# raw_df <- do.call(rbind, df_list)
# raw_df
```

- Test using selective data

Hide

```
# paths <- c(
#   "./source/2019-Q1.csv",
#   "./source/2019-Q2.csv",
#   "./source/2019-Q3.csv",
#   "./source/2019-Q4.csv",
#   "./source/2020-01.csv",
#   "./source/2020-02.csv",
#   "./source/2020-03.csv",
#   "./source/2020-04.csv",
#   "./source/2020-05.csv",
#   "./source/2020-06.csv",
#   "./source/2020-07.csv",
#   "./source/2020-08.csv",
#   "./source/2020-09.csv",
#   "./source/2020-10.csv",
#   "./source/2020-11.csv",
#   "./source/2020-12.csv"
# )
# # paths <- c("./source/2020-01.csv")
# df_list <- lapply(paths, read.csv)
# raw_df <- do.call(rbind, df_list)
# raw_df
```

Data Processing

Handling NA and NULL value

- Handle NA value that exist in the raw data

Hide

```
# # Remove rows with any NA values
# proc_NA_df <- raw_df[complete.cases(raw_df), ]
#
# # Remove rows with "NULL" values
# proc_null_df <- proc_NA_df %>%
#   filter(!(End.Station.Name == "NULL" | End.Station.Id == "NULL" | Start.Station.Id =="NULL" | Start.Station.Name == "NULL"))
# proc_df <- proc_null_df
```

Converting Time-Dimension Data

- Divide time into year, month, date, hour, and minute.

Hide

```
# # Divide "Start.Time" into columns
# proc_df$Start.Time <- as.POSIXct(proc_df$Start.Time, format = "%m/%d/%Y %H:%M")
# proc_df$Start.Year <- as.factor(format(proc_df$Start.Time, "%Y"))
# proc_df$Start.Month <- as.factor(format(proc_df$Start.Time, "%m"))
# proc_df$Start.Date <- as.factor(format(proc_df$Start.Time, "%d"))
# proc_df$Start.Hours <- as.factor(format(proc_df$Start.Time, "%H"))
# proc_df$Start.Minutes <- as.factor(format(proc_df$Start.Time, "%M"))
#
# # Divide "End" into columns
# proc_df$End.Time <- as.POSIXct(proc_df$End.Time, format = "%m/%d/%Y %H:%M")
# proc_df$End.Year <- as.factor(format(proc_df$End.Time, "%Y"))
# proc_df$End.Month <- as.factor(format(proc_df$End.Time, "%m"))
# proc_df$End.Date <- as.factor(format(proc_df$End.Time, "%d"))
# proc_df$End.Hours <- as.factor(format(proc_df$End.Time, "%H"))
# proc_df$End.Minutes <- as.factor(format(proc_df$End.Time, "%M"))
#
# # factor user.type
# proc_df$User.Type <- as.factor(proc_df$User.Type)
#
#
# # Drop Start.Time and End.Time
# proc_df <- proc_df %>% select(-Start.Time, -End.Time)
```

- Check NA value again, in case of any possible values generated during the data processing.

Hide

```
# is_miss <- any(is.na(proc_df))
#
# # if the processed_df contains missing value, drop the rows with missing values and assign to df
# if (is_miss) {
#   df <- proc_df[complete.cases(proc_df), ]
# # otherwise, df = proc_df
# }else{
#   df <- proc_df
# }
#
# is_miss <- any(is.na(df))
# cat("df has missing value? ", is_miss)  # output result
```

- Data Overview after data processing.

Hide

```
# # Data overview after data processing
# num_row <- nrow(df)             # total rows
# column_names <- colnames(df)    # column names
# cat("\n\nNumber of rows: ", "\n", num_row)
# cat("\n\nColumn names: ", "\n", column_names)
#
# cat("\n\nDisplay the Structure:\n")
# str(df)
#
# cat("\n\nDisplay Summaries:\n")
# summary(df)
```

- Preview data

Hide

```
# df
# head(df, 10)
```

Exporting Processed Data(Optional)

- Export the Processed data to permanently store the processed data.

Hide

```
# # Path to export
# output_file <- "./data/dataset.csv"
# # export
# write.csv(df, file = output_file, row.names = FALSE)
```

ReLoading Processed Data

Hide

```
data_file <- "./data/dataset.csv"
df <- read.csv(data_file)
df
```

Trip.Id <dbl>	Trip..Duration <int>	Start.Station.Id <int>	Start.Station.Name <chr>
4581278	1547	7021	Bay St / Albert St
4581279	1112	7160	King St W / Tecumseth St
4581280	589	7055	Jarvis St / Carlton St
4581281	259	7012	Elizabeth St / Edward St (Bus Terminal)
4581282	281	7041	Edward St / Yonge St
4581283	624	7041	Edward St / Yonge St
4581284	604	7041	Edward St / Yonge St
4581285	416	7275	Queen St W / James St
4581286	192	7071	161 Bleecker St (South of Wellesley)
4581287	518	7199	College St / Markham St

1-10 of 17,789,237 rows | 1-4 of 18 columns Previous **1** 2 3 4 5 6 ... 100 Next

- Data Overview afater loading data.

Hide

```
# Data overview after data processing
num_row <- nrow(df)             # total rows
column_names <- colnames(df)    # column names
cat("\n\nNumber of rows: ", "\n", num_row)
```

```
Number of rows:
 17789237
```

Hide

```
cat("\n\nColumn names: ", "\n", column_names)
```

```
Column names:
 Trip.Id Trip..Duration Start.Station.Id Start.Station.Name End.Station.Id End.Station.Name Bike.Id Use
r.Type Start.Year Start.Month Start.Date Start.Hours Start.Minutes End.Year End.Month End.Date End.Hour
s End.Minutes
```

Hide

```
cat("\n\nDisplay the Structure:\n")
```

```
Display the Structure:
```

Hide

```
str(df)
```

```
'data.frame':	17789237 obs. of  18 variables:
 $ Trip.Id           : num  4581278 4581279 4581280 4581281 4581282 ...
 $ Trip..Duration    : int  1547 1112 589 259 281 624 604 416 192 518 ...
 $ Start.Station.Id  : int  7021 7160 7055 7012 7041 7041 7041 7275 7071 7199 ...
 $ Start.Station.Name: chr  "Bay St / Albert St" "King St W / Tecumseth St" "Jarvis St / Carlton St" "E
lizabeth St / Edward St (Bus Terminal)" ...
 $ End.Station.Id    : int  7233 7051 7013 7235 7257 7031 7031 7041 7311 7252 ...
 $ End.Station.Name  : chr  "King / Cowan Ave - SMART" "Wellesley St E / Yonge St (Green P)" "Scott St
/ The Esplanade" "Bay St / College St (West Side) - SMART" ...
 $ Bike.Id           : chr  "1296" "2947" "2293" "283" ...
 $ User.Type         : chr  "Annual Member" "Annual Member" "Annual Member" "Annual Member" ...
 $ Start.Year        : int  2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...
 $ Start.Month       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Start.Date        : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Start.Hours       : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Start.Minutes     : int  8 10 15 16 19 26 26 26 34 38 ...
 $ End.Year          : int  2019 2019 2019 2019 2019 2019 2019 2019 2019 2019 ...
 $ End.Month         : int  1 1 1 1 1 1 1 1 1 1 ...
 $ End.Date          : int  1 1 1 1 1 1 1 1 1 1 ...
 $ End.Hours         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ End.Minutes       : int  33 29 25 20 24 36 36 33 37 46 ...
```

Hide

```
cat("\n\nDisplay Summaries:\n")
```

```
Display Summaries:
```

Hide

```
summary(df)
```

```
    Trip.Id         Trip..Duration     Start.Station.Id Start.Station.Name End.Station.Id
 Min.   : 4581278   Min.   :       0   Min.   :7000     Length:17789237    Min.   :7000  
 1st Qu.: 9607762   1st Qu.:     439   1st Qu.:7078     Class :character   1st Qu.:7077  
 Median :14783657   Median :     729   Median :7227     Mode  :character   Median :7224  
 Mean   :15075490   Mean   :    1047   Mean   :7246                        Mean   :7243  
 3rd Qu.:20359050   3rd Qu.:    1164   3rd Qu.:7383                        3rd Qu.:7381  
 Max.   :26682738   Max.   :12403785   Max.   :7681                        Max.   :7681  
 End.Station.Name     Bike.Id           User.Type           Start.Year    Start.Month    
 Length:17789237    Length:17789237    Length:17789237    Min.   :2019   Min.   : 1.000  
 Class :character   Class :character   Class :character   1st Qu.:2020   1st Qu.: 6.000  
 Mode  :character   Mode  :character   Mode  :character   Median :2021   Median : 7.000  
                                                          Mean   :2021   Mean   : 7.246  
                                                          3rd Qu.:2023   3rd Qu.: 9.000  
                                                          Max.   :2023   Max.   :12.000  
   Start.Date     Start.Hours     Start.Minutes      End.Year      End.Month         End.Date    
 Min.   : 1.00   Min.   : 0.00   Min.   : 0.00   Min.   :2019   Min.   : 1.000   Min.   : 1.00  
 1st Qu.: 8.00   1st Qu.:11.00   1st Qu.:14.00   1st Qu.:2020   1st Qu.: 6.000   1st Qu.: 8.00  
 Median :16.00   Median :15.00   Median :30.00   Median :2021   Median : 7.000   Median :16.00  
 Mean   :15.73   Mean   :14.57   Mean   :29.55   Mean   :2021   Mean   : 7.246   Mean   :15.73  
 3rd Qu.:23.00   3rd Qu.:18.00   3rd Qu.:45.00   3rd Qu.:2023   3rd Qu.: 9.000   3rd Qu.:23.00  
 Max.   :31.00   Max.   :23.00   Max.   :59.00   Max.   :2024   Max.   :12.000   Max.   :31.00  
   End.Hours      End.Minutes   
 Min.   : 0.00   Min.   : 0.00  
 1st Qu.:11.00   1st Qu.:15.00  
 Median :16.00   Median :30.00  
 Mean   :14.71   Mean   :29.79  
 3rd Qu.:18.00   3rd Qu.:45.00  
 Max.   :23.00   Max.   :59.00  
```

- Preview data

Hide

```
# df
head(df, 10)
```

	Trip.Id	Trip..Duration	Start.Station.Id	Start.Station.Name	End
	<dbl>	<int>	<int>	<chr>	
1	4581278	1547	7021	Bay St / Albert St	
2	4581279	1112	7160	King St W / Tecumseth St	
3	4581280	589	7055	Jarvis St / Carlton St	
4	4581281	259	7012	Elizabeth St / Edward St (Bus Terminal)	
5	4581282	281	7041	Edward St / Yonge St	
6	4581283	624	7041	Edward St / Yonge St	
7	4581284	604	7041	Edward St / Yonge St	

	Trip.Id	Trip..Duration	Start.Station.Id	Start.Station.Name	End
	<dbl>	<int>	<int>	<chr>	
8	4581285	416	7275	Queen St W / James St	
9	4581286	192	7071	161 Bleecker St (South of Wellesley)	
10	4581287	518	7199	College St / Markham St	

Trip analysis

Yearly Trip Trends

Hide

```
trip_yearly <- df %>%
  group_by(Year = Start.Year) %>%
  summarize(Trip = n())
trip_yearly
```

Year	Trip
<int>	<int>
2019	2439047
2020	2908239
2021	3565746
2022	4266232
2023	4609973

5 rows

Hide

```
ggplot(
  data = trip_yearly,
  mapping = aes(
      x = Year,
      y = Trip,
      group = 1
  )
) +
  geom_line() +
  geom_point() +
  geom_text(aes(label = as.character(Trip)), vjust = 1, hjust = 0.5) +
  labs(
    title = "Yearly Trip Trend",
    x = "Year",
    y = "Trip"
  ) +
  theme_minimal()
```

Monthly Trip Distribution

- Comparing monthly trip to unveil patterns over the months

Hide

```
trip_monthly_across_year <- df %>%
  group_by(
    Year = Start.Year,
    Month = Start.Month
    ) %>%
  summarize(Trip = n())
```

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

Hide

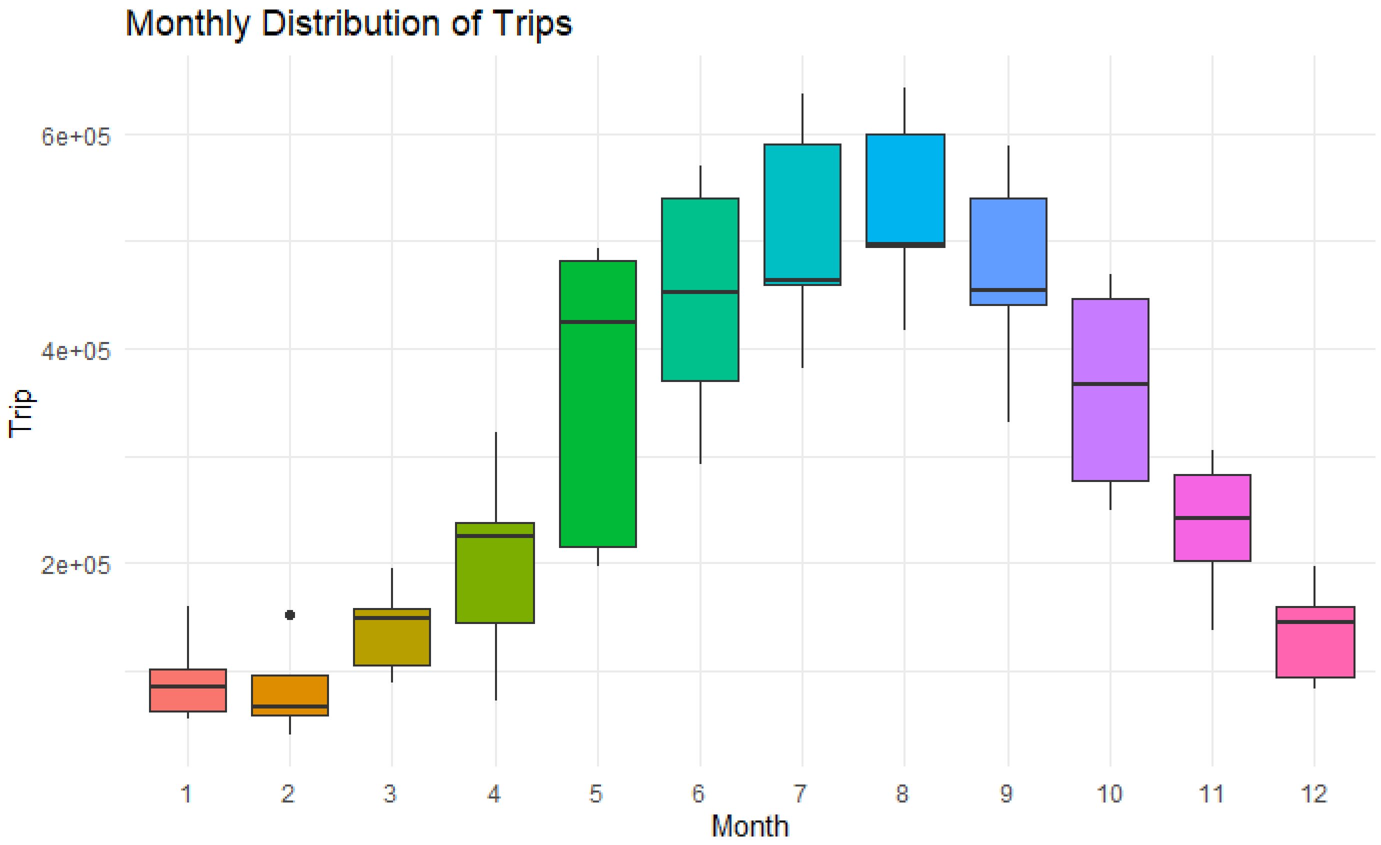
```
trip_monthly_across_year
```

Year <int>	Month <int>	Trip <int>
2019	1	61461
2019	2	40055
2019	3	87540
2019	4	145150
2019	5	214613
2019	6	291917
2019	7	382235
2019	8	417393
2019	9	330703
2019	10	248467

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

Hide

```
ggplot(
  data = trip_monthly_across_year,
  mapping = aes(
    x = factor(Month),
    y = Trip,
    fill = factor(Month)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Monthly Distribution of Trips",
    x = "Month",
    y = "Trip",
    fill = "Month"
  ) +
  theme_minimal()
```



Hourly Trip Pattern

- Exploring Patterns Throughout the Day

Hide

```
trip_hourly_across_month <- df %>%
  group_by(
    Month = Start.Month,
    Hour = Start.Hours
    ) %>%
  summarize(Trip = n())
```

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

Hide

```
trip_hourly_across_month
```

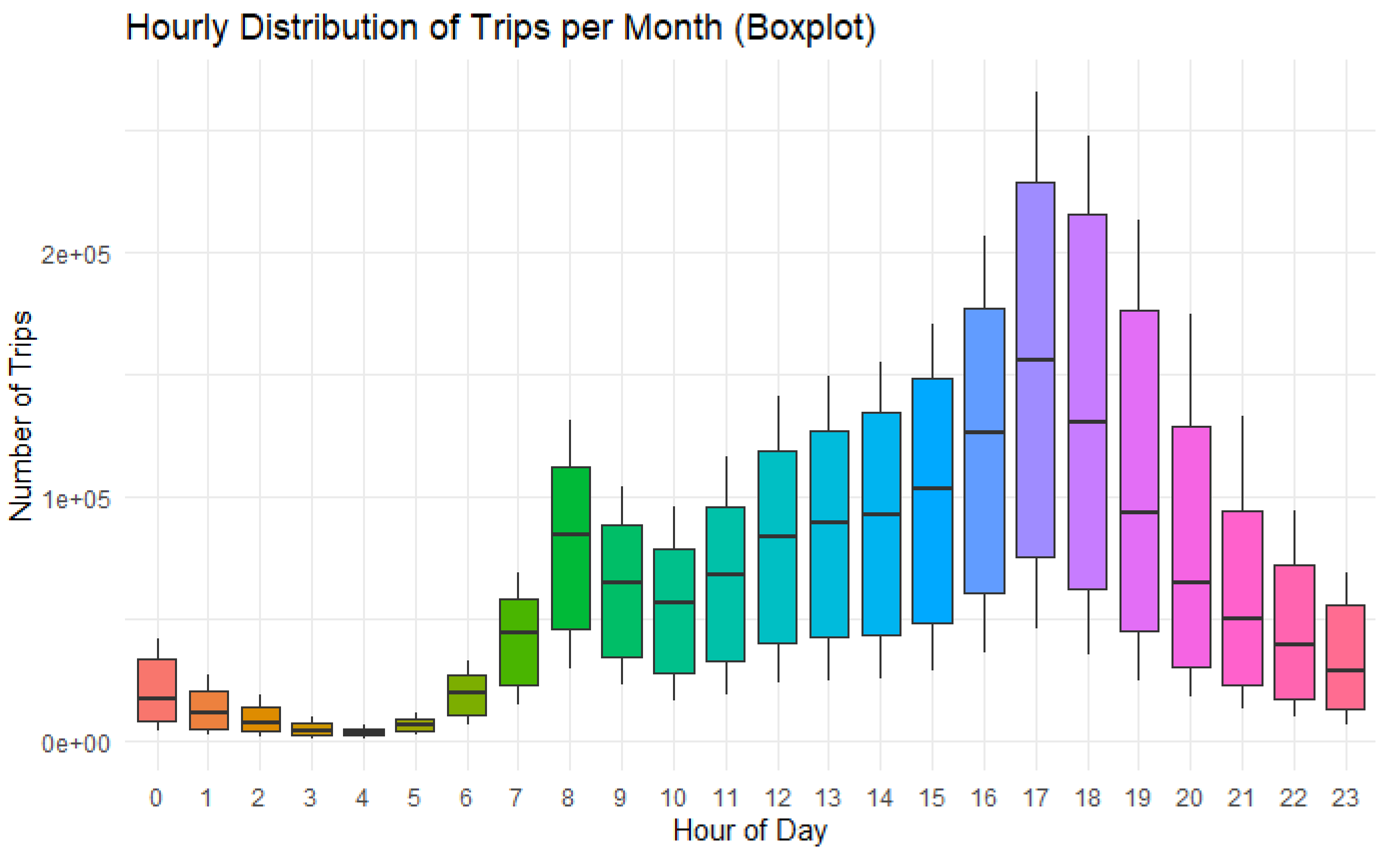
Month <int>	Hour <int>	Trip <int>
1	0	4571
1	1	3076
1	2	2324
1	3	1436
1	4	1403
1	5	2635
1	6	7521

Month	Hour	Trip
<int>	<int>	<int>
1	7	17508
1	8	37136
1	9	27818

1-10 of 288 rows Previous **1** 2 3 4 5 6 ... 29 Next

Hide

```
ggplot(
  trip_hourly_across_month,
  aes(
    x = factor(Hour),
    y = Trip,
    fill = factor(Hour)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Hourly Distribution of Trips per Month (Boxplot)",
    x = "Hour of Day",
    y = "Number of Trips",
    fill = "Hour of Day") +
  theme_minimal()
```



Duration Analysis

Duration Yearly Trend

Hide

```
duration_yearly <- df %>%
  group_by(Year = Start.Year) %>%
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2))

duration_yearly
```

Year <int>	Mean_Duration <dbl>
2019	17.39
2020	19.38
2021	17.27
2022	16.00
2023	17.74

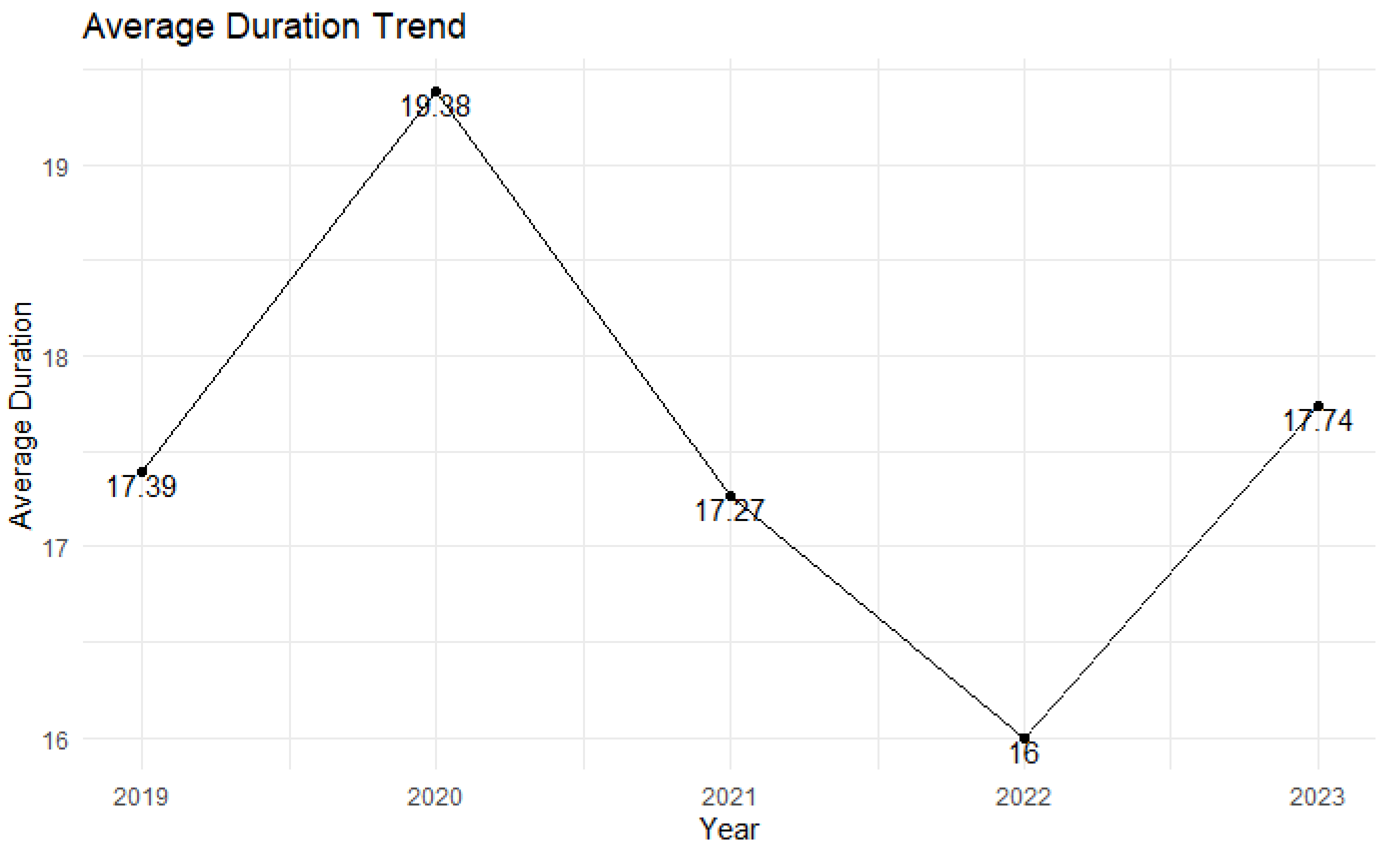
5 rows

Hide

```
NA
```

Hide

```
ggplot(
  data = duration_yearly,
  mapping = aes(
      x = Year,
      y = Mean_Duration,
      group = 1
    )
  ) +
  geom_line() +
  geom_point() +
  geom_text(
    aes(
      label = as.character(Mean_Duration)
      ),
    vjust = 1,
    hjust = 0.5
  ) +
  labs(
    title = "Average Duration Trend",
    x = "Year",
    y = "Average Duration"
  ) +
  theme_minimal()
```



Duration Pattern over Months

Hide

```
duration_monthly_across_year <- df %>%
  group_by(
    Year = Start.Year,
    Month = Start.Month) %>%
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2))  # Convert mean to minutes
```

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

Hide

```
duration_monthly_across_year
```

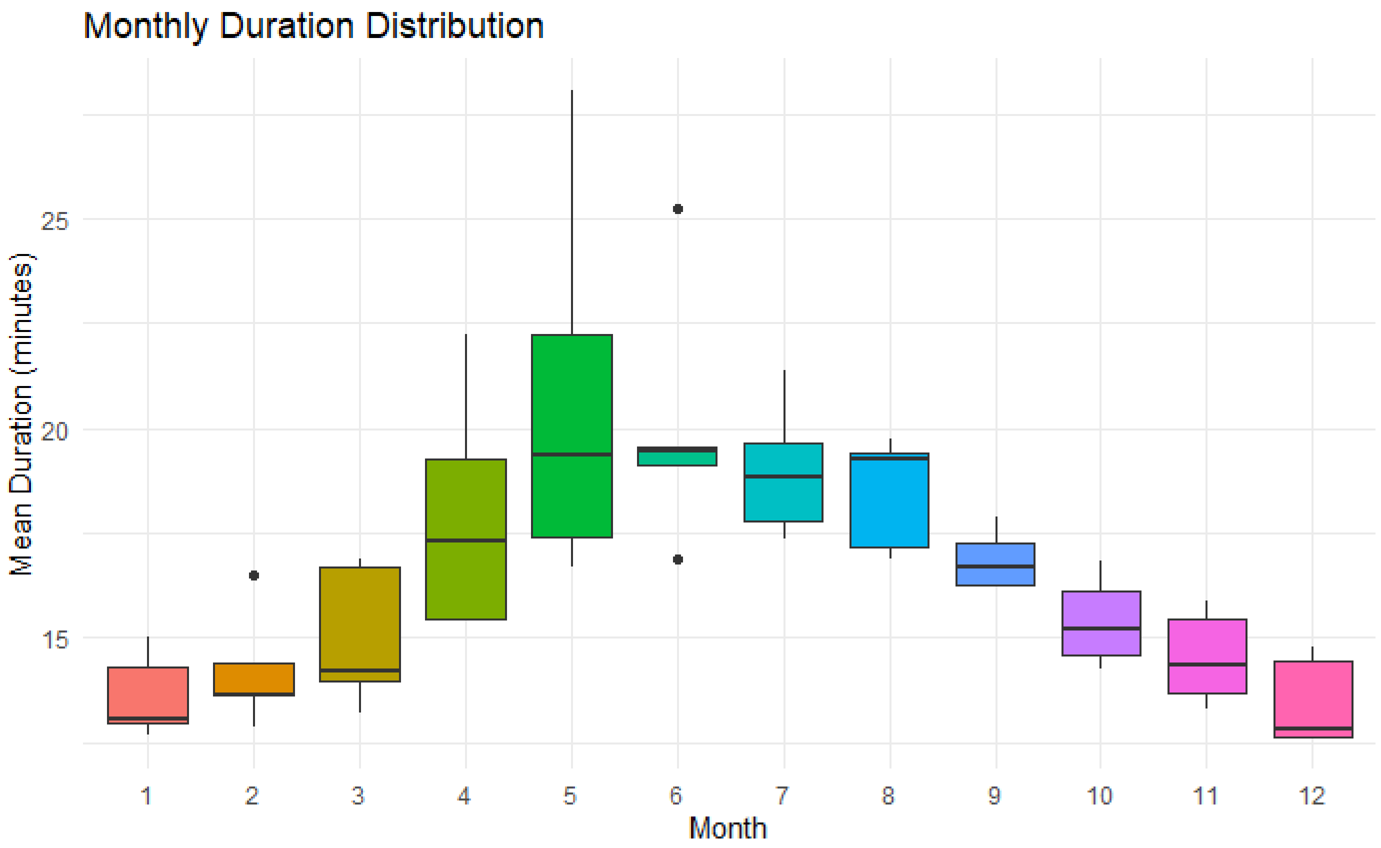
Year <int>	Month <int>	Mean_Duration <dbl>
2019	1	12.96
2019	2	14.41
2019	3	13.20
2019	4	15.45
2019	5	16.69
2019	6	19.56
2019	7	18.82
2019	8	19.75
2019	9	16.67

Year	Month	Mean_Duration
<int>	<int>	<dbl>
2019	10	16.11

1-10 of 60 rows Previous **1** 2 3 4 5 6 Next

Hide

```
ggplot(
  data = duration_monthly_across_year,
  mapping = aes(
    x = factor(Month),
    y = Mean_Duration,
    fill = factor(Month)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Monthly Duration Distribution",
    x = "Month",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()
```



Duration Pattern throughout A Day

Hide

```
duration_hourly_across_month <- df %>%
  group_by(
    Month = Start.Month,
    Hour = Start.Hours
    ) %>%
  summarize(Mean_Duration = round(mean(Trip..Duration) / 60, 2))  # Convert mean to minutes
```

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

Hide

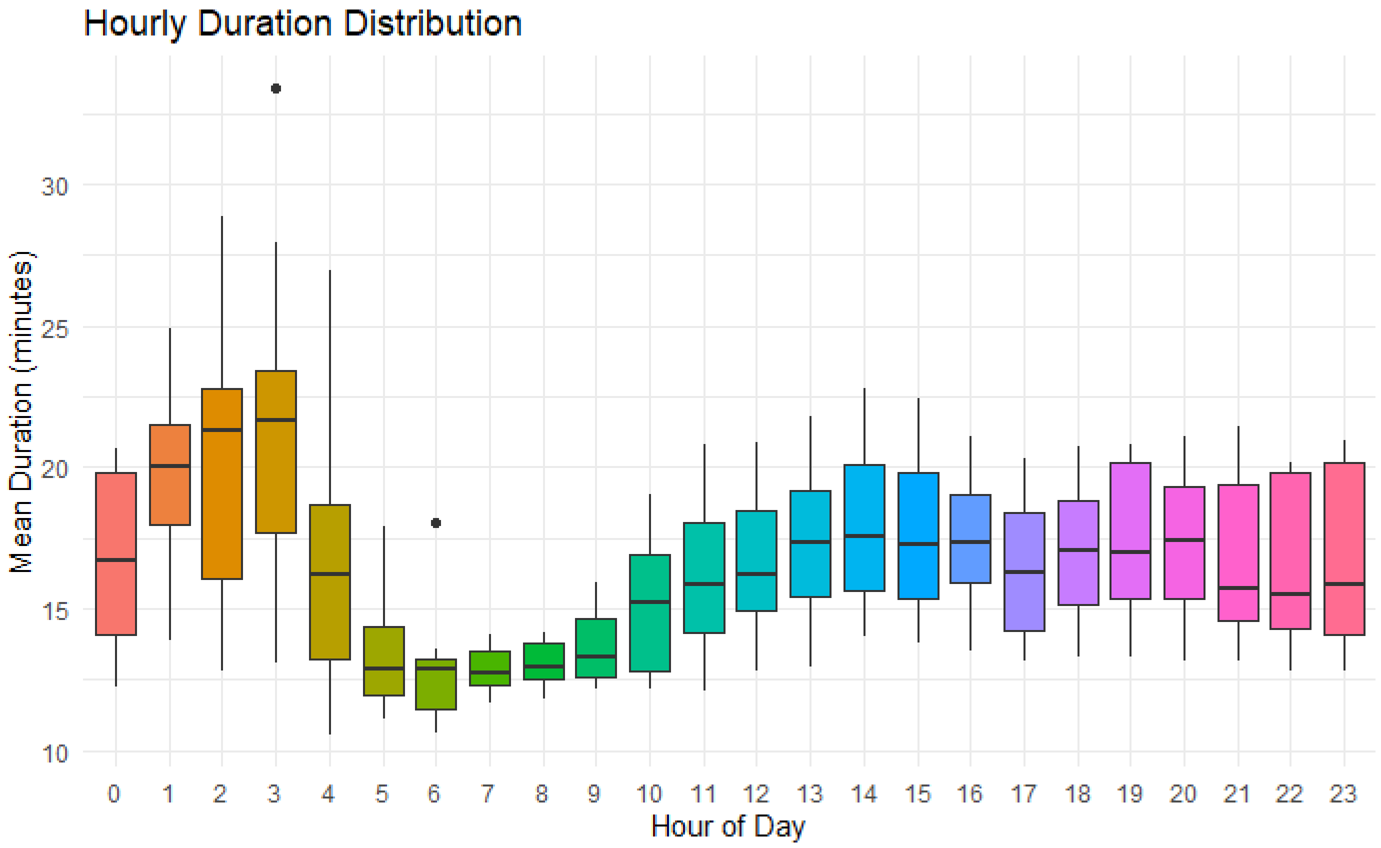
```
duration_hourly_across_month
```

Month <int>	Hour <int>	Mean_Duration <dbl>
1	0	13.33
1	1	16.31
1	2	24.04
1	3	33.38
1	4	13.05
1	5	12.86
1	6	12.74
1	7	12.37
1	8	12.19
1	9	12.71

1-10 of 288 rows Previous **1** 2 3 4 5 6 ... 29 Next

Hide

```
ggplot(
  data = duration_hourly_across_month,
  mapping = aes(
    x = factor(Hour),
    y = Mean_Duration,
    fill = factor(Hour)
    )
  ) +
  geom_boxplot(show.legend = FALSE) +
  labs(
    title = "Hourly Duration Distribution",
    x = "Hour of Day",
    y = "Mean Duration (minutes)"
  ) +
  theme_minimal()
```



Geolocation Analysis

Hot Start Locations

Hot start spot (Top 10)

Hide

```
hot_start_spot <- df %>%
  group_by(Start.Station.Name) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_start_spot
```

Start.Station.Name <chr>	Total_Trips <int>
York St / Queens Quay W	167820
Bay St / Queens Quay W (Ferry Terminal)	129723
Bay St / College St (East Side)	124516
Queens Quay E / Lower Sherbourne St	114306
HTO Park (Queens Quay W)	111224
Lake Shore Blvd W / Ontario Dr	111058
Bay St / Wellesley St W	109131
Ontario Place Blvd / Lake Shore Blvd W (East)	109044
Bathurst St/Queens Quay(Billy Bishop Airport)	108858

Start.Station.Name <chr>	Total_Trips <int>
Dundas St E / Regent Park Blvd	105913

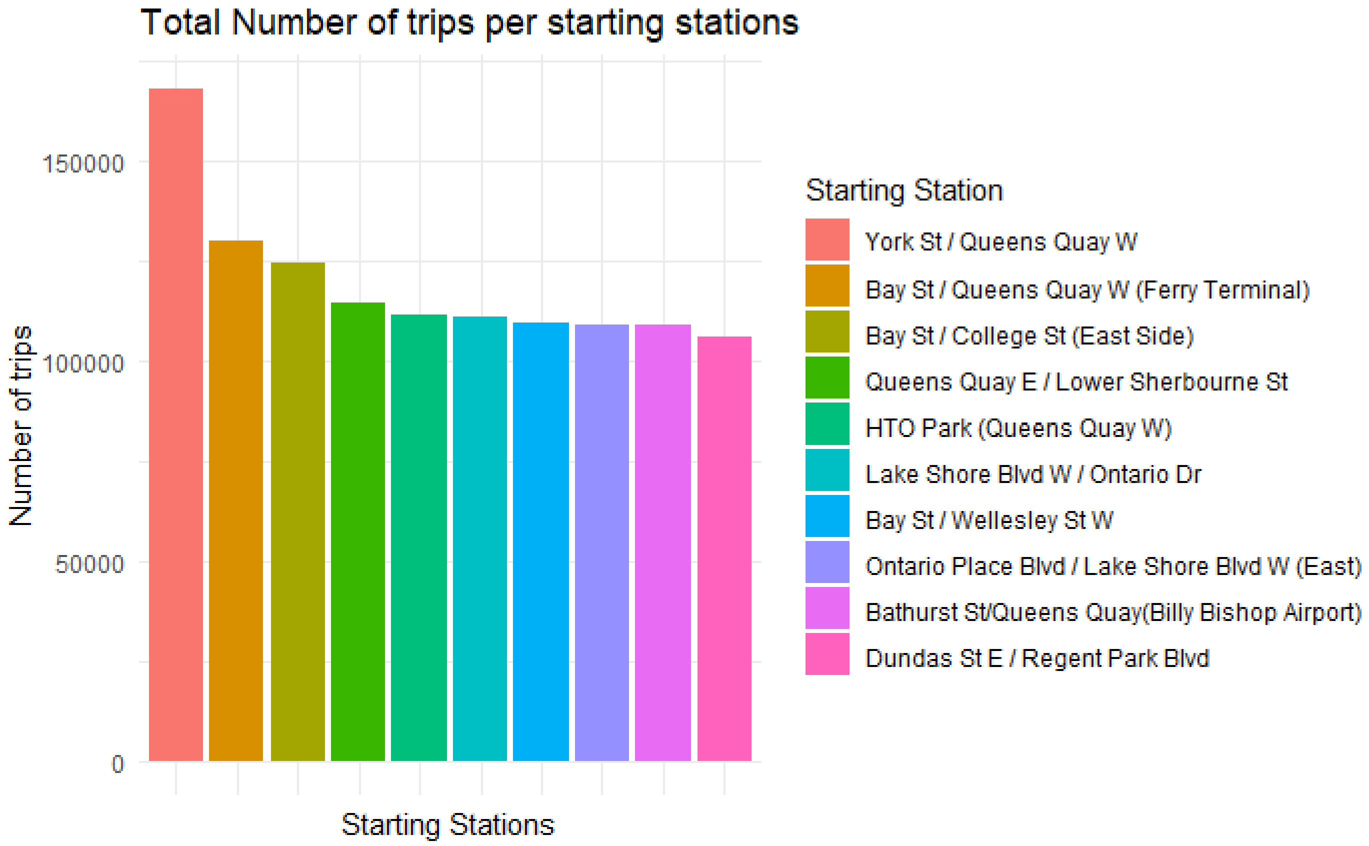
1-10 of 10 rows

Hide

```
NA
```

Hide

```
ggplot(
  data = hot_start_spot,
  mapping = aes(
      x = reorder(Start.Station.Name, -Total_Trips),
      y = Total_Trips,
      fill = reorder(Start.Station.Name, -Total_Trips)
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per starting stations",
    x = "Starting Stations",
    y = "Number of trips",
    fill = "Starting Station"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank())
```



Hide

```
NA
NA
```

Hot start street (Top 10)

Hide

```
library(stringr)

hot_start_streets <- df %>%
  separate_rows(Start.Station.Name, sep = " / ") %>% # Separate the station names
  mutate(Start.Station.Street = Start.Station.Name) %>%
  group_by(Start.Station.Street) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
```

```
Warning: 输入的字符串5347586不是UTF-8Warning: 输入的字符串5347588不是UTF-8Warning: 输入的字符串5350238不是UT
F-8Warning: 输入的字符串5350495不是UTF-8Warning: 输入的字符串5350926不是UTF-8
```

Hide

```
hot_start_streets
```

Start.Station.Street	**Total_Trips**
<chr>	<int>
Yonge St	905277
Queen St W	777197
Bay St	754436
Bloor St W	648682
King St W	630577
Dundas St W	596545
College St	507797
Spadina Ave	430988
Dundas St E	393854
Adelaide St W	391302

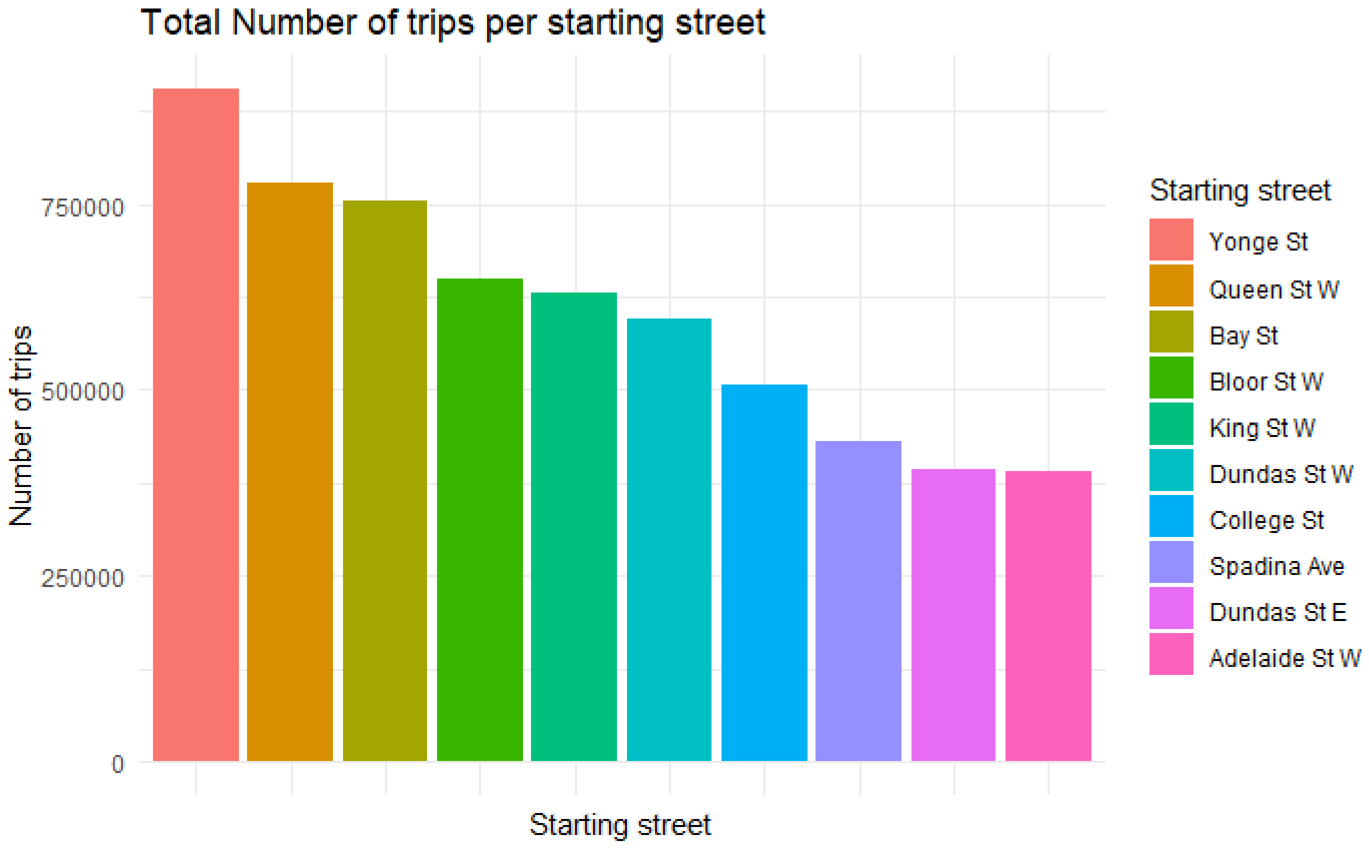
1-10 of 10 rows

Hide

```
NA
```

Hide

```
ggplot(
  data = hot_start_streets,
  mapping = aes(
      x = reorder(Start.Station.Street, -Total_Trips),
      y = Total_Trips,
      fill = reorder(Start.Station.Street, -Total_Trips)
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per starting street",
    x = "Starting street",
    y = "Number of trips",
    fill = "Starting street"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank())
```



Hot End Locations

Hot End spot (Top 10)

Hide

```
hot_end_spot <- df %>%
  group_by(End.Station.Name) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_end_spot
```

End.Station.Name <chr>	Total_Trips <int>
York St / Queens Quay W	191618
Bay St / Queens Quay W (Ferry Terminal)	137039
Union Station	134622
Bay St / College St (East Side)	125581
HTO Park (Queens Quay W)	125254
Bathurst St/Queens Quay(Billy Bishop Airport)	121870
Queens Quay E / Lower Sherbourne St	119303
Queens Quay / Yonge St	114252
Dundas St W / Yonge St	113685
Lake Shore Blvd W / Ontario Dr	110234

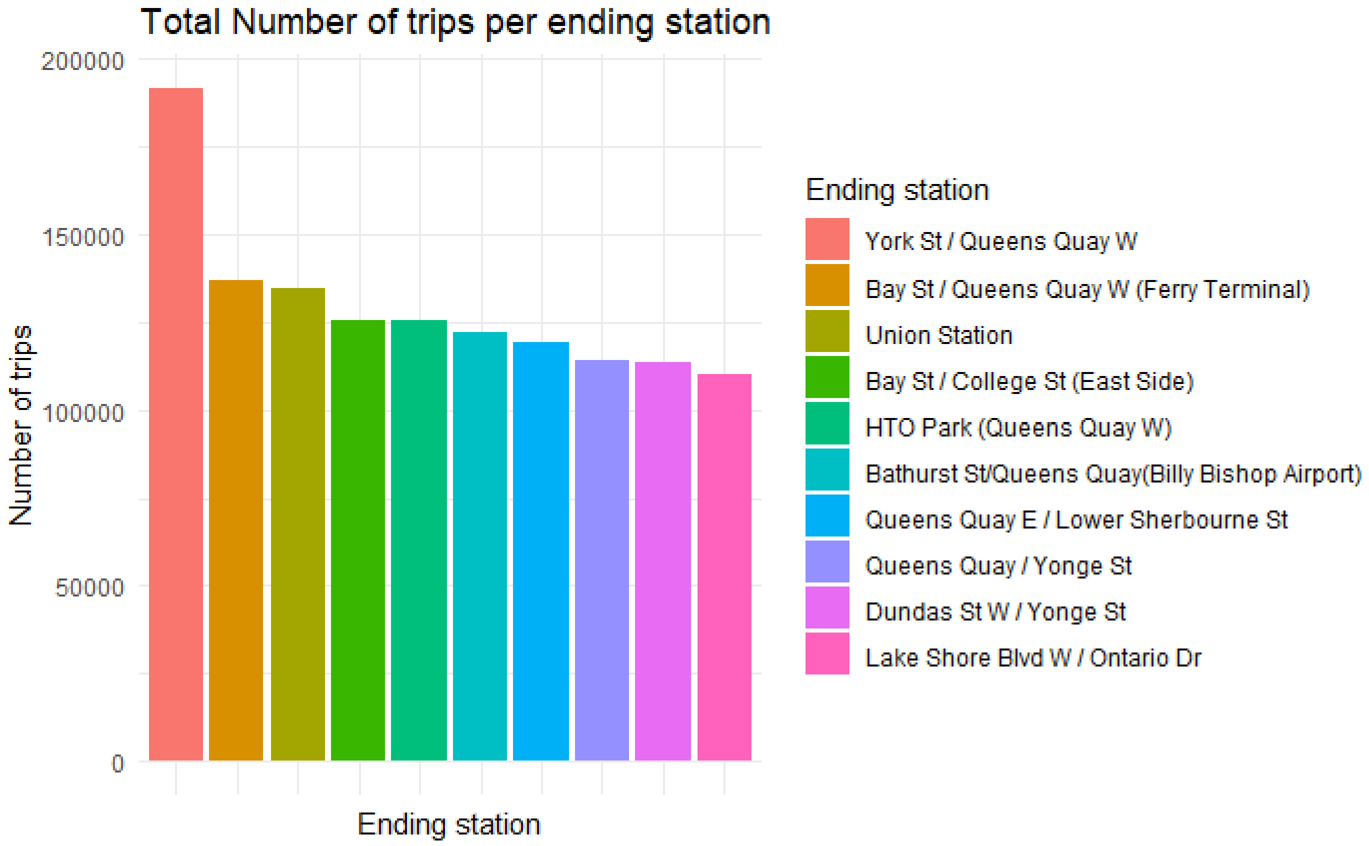
1-10 of 10 rows

Hide

```
NA
```

Hide

```
ggplot(
  data = hot_end_spot,
  mapping = aes(
      x = reorder(End.Station.Name, -Total_Trips),
      y = Total_Trips,
      fill = reorder(End.Station.Name, -Total_Trips)
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per ending station",
    x = "Ending station",
    y = "Number of trips",
    fill = "Ending station"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank())
```



Hot end street (Top 10)

Hide

```
hot_end_streets <- df %>%
  separate_rows(End.Station.Name, sep = " / ") %>% # Separate the station names
  mutate(End.Station.Street = End.Station.Name) %>%
  group_by(End.Station.Street) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
```

```
Warning: 输入的字符串5348071不是UTF-8Warning: 输入的字符串5348390不是UTF-8Warning: 输入的字符串5348452不是UT
F-8Warning: 输入的字符串5349390不是UTF-8Warning: 输入的字符串5349530不是UTF-8
```

Hide

```
hot_end_streets
```

End.Station.Street <chr>	Total_Trips <int>
Yonge St	878065
Queen St W	827472
Bay St	720596
King St W	692586
Bloor St W	605196
Dundas St W	601900
College St	501619
Spadina Ave	424328

End.Station.Street	Total_Trips
<chr>	<int>
Adelaide St W	409449
Dundas St E	406959

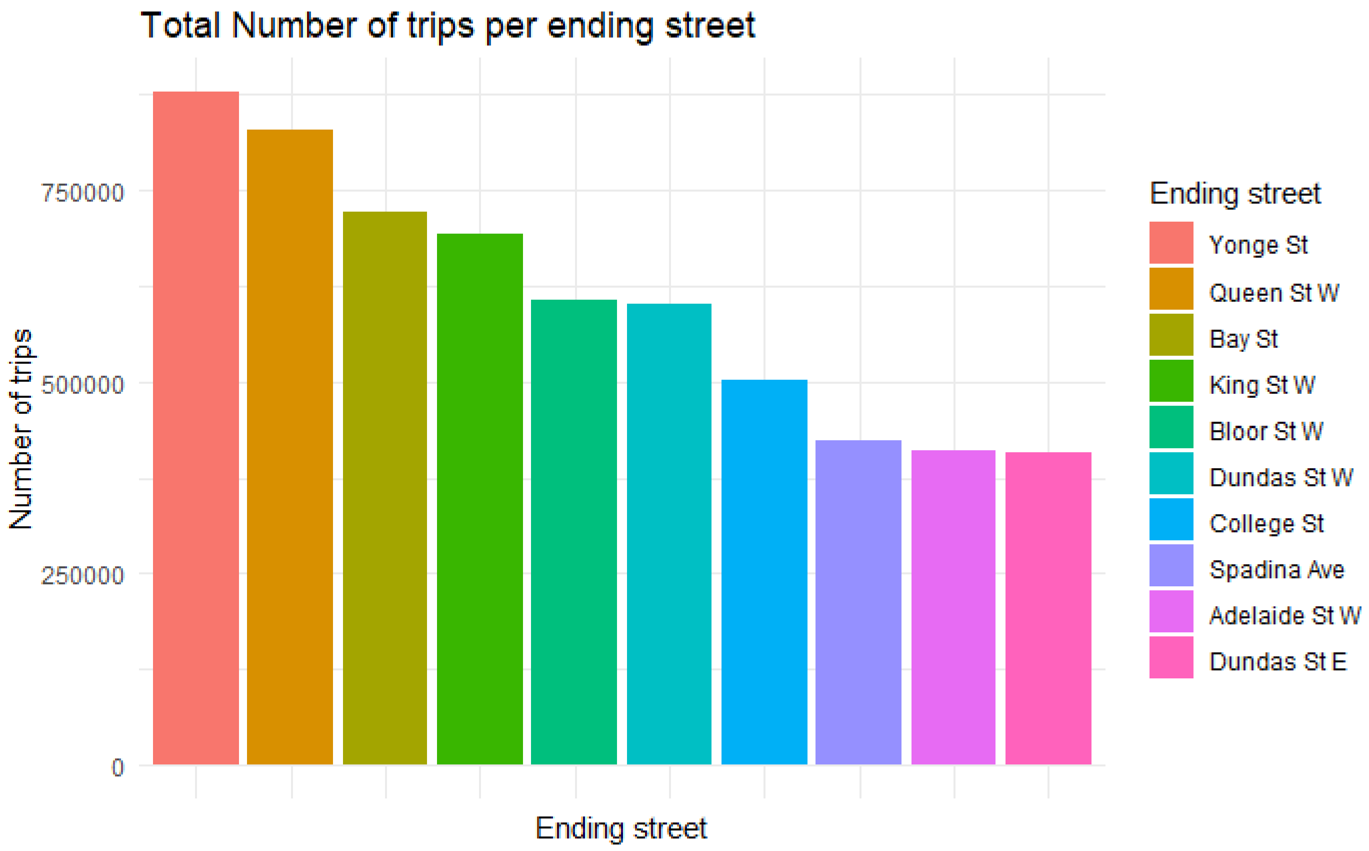
1-10 of 10 rows

Hide

```
NA
NA
```

Hide

```
ggplot(
  data = hot_end_streets,
  mapping = aes(
      x = reorder(End.Station.Street, -Total_Trips),
      y = Total_Trips,
      fill = reorder(End.Station.Street, -Total_Trips)
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per ending street",
    x = "Ending street",
    y = "Number of trips",
    fill = "Ending street"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank())
```



Hot Trips (Top 10)

Hide

```
hot_trip <- df %>%
  mutate(Trip.Spots = paste(Start.Station.Name, End.Station.Name, sep =" -> ")) %>%
  group_by(Trip.Spots) %>%
  summarize(Total_Trips = n()) %>%
  arrange(desc(Total_Trips)) %>%
  slice_head(n = 10)
hot_trip
```

Trip.Spots
<chr>
Tommy Thompson Park (Leslie Street Spit) -> Tommy Thompson Park (Leslie Street Spit)
Ontario Place Blvd / Lake Shore Blvd W (East) -> Ontario Place Blvd / Lake Shore Blvd W (East)
Humber Bay Shores Park West -> Humber Bay Shores Park West
Bay St / Queens Quay W (Ferry Terminal) -> Bay St / Queens Quay W (Ferry Terminal)
HTO Park (Queens Quay W) -> HTO Park (Queens Quay W)
York St / Queens Quay W -> York St / Queens Quay W
Cherry Beach -> Tommy Thompson Park (Leslie Street Spit)
Humber Bay Shores Park / Marine Parade Dr -> Humber Bay Shores Park / Marine Parade Dr
Tommy Thompson Park (Leslie Street Spit) -> Cherry Beach
Humber Bay Shores Park East -> Humber Bay Shores Park East

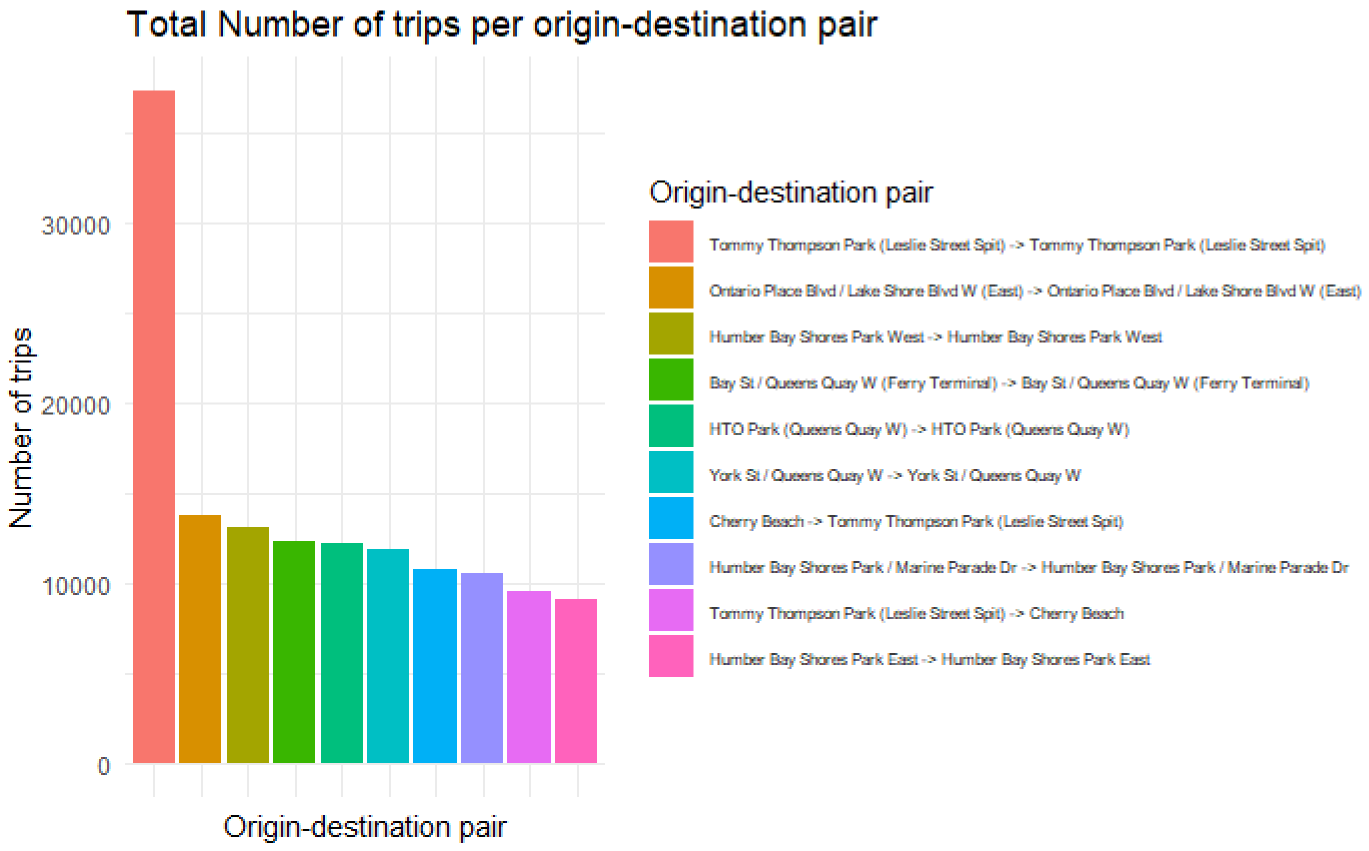
1-10 of 10 rows | 1-1 of 2 columns

Hide

```
NA
NA
```

Hide

```
ggplot(
  data = hot_trip,
  mapping = aes(
      x = reorder(Trip.Spots, -Total_Trips),
      y = Total_Trips,
      fill = reorder(Trip.Spots, -Total_Trips)
      )
  ) +
  geom_bar(
    stat = "identity"
  ) +
  labs(
    title = "Total Number of trips per origin-destination pair",
    x = "Origin-destination pair",
    y = "Number of trips",
    fill = "Origin-destination pair"
  ) +
  theme_minimal()+
  theme(axis.text.x = element_blank()  ,
        legend.text = element_text(size = 6))
```



Skiny Rendering