APPLIED DATA SCIENCE WITH R CAPSTONE PROJECT

NUSHRATE AHMED

10.5.2024

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

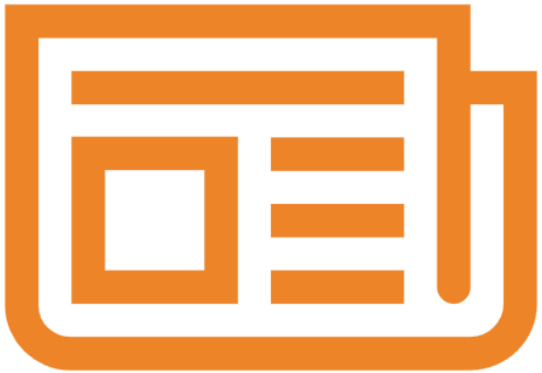
EXECUTIVE SUMMARY

- High Demands:
  - Seasons: Summer months
  - Countries: South Korea and China
  - Peak Hours: Afternoon and evening
  - Influencing Factors: Weather significantly affected bike rentals

INTRODUCTION

- The project aims to investigate how weather conditions influence bike-sharing demand in urban environments.
- Data: **Seoul Bike Sharing Demand Data Set**
- Bike rentals are available in cities across the globe, making it essential to maintain a consistent and reliable supply for public accessibility and convenience. Additionally, keeping costs low is crucial, which can be accomplished by aligning the number of bikes available with actual demand.
- To achieve this optimization, it is vital to forecast hourly bike demand based on current factors, such as weather conditions. The Seoul Bike Sharing Demand Data Set has been developed for this purpose, offering detailed weather information including:
- - Temperature
- - Humidity
- - Wind speed
- - Visibility
- - Dew point
- - Solar radiation
- - Snowfall and rainfall
- - Hourly bike rental figures

METHODOLOGY



Perform data collection

Perform data wrangling

Perform exploratory data analysis (EDA) using SQL and visualization

Perform predictive analysis using regression models

How to build the baseline model

How to improve the baseline model

Build a R Shiny dashboard app

DATA COLLECTION

```r
# Load necessary libraries
library(rvest)
library(tidyverse)
library(dplyr)
library(httr)

# Data Collection

# Define the URL and read the webpage
url <- "https://en.wikipedia.org/wiki/List_of_bicycle-sharing_systems"
root_node <- read_html(url)

# Extract and convert the second table into a data frame
bike_share_df <- html_table(html_nodes(root_node, "table")[[2]], fill = TRUE)

# Display the head and summary of the dataframe
head(bike_share_df)
summary(bike_share_df)

# Export the dataframe to a CSV file
write.csv(bike_share_df, "raw_bike_sharing_systems.csv", row.names = FALSE)

# Define a function to get the weather forecast
get_weather_forecast_by_cities <- function(city_names) {
  results_list <- list()  # Store results in a list

  for (city_name in city_names) {
    forecast_url <- 'https://api.openweathermap.org/data/2.5/forecast'
    forecast_query <- list(q = city_name, appid = "your_api_key_here", units = "metric")

    response <- GET(forecast_url, query = forecast_query)
    json_list <- content(response, as = "parsed")

    for (result in json_list$list) {
      # Append results to the list
      results_list[[length(results_list) + 1]] <- data.frame(
        CITY = city_name,
        WEATHER = result$weather[[1]]$main,
        VISIBILITY = result$visibility,
        TEMPERATURE = result$main$temp,
        MIN_TEMP = result$main$temp_min,
        MAX_TEMP = result$main$temp_max,
        PRESSURE = result$main$pressure,
        HUMIDITY = result$main$humidity,
        WIND_SPEED = result$wind$speed,
        WIND_DEG = result$wind$deg,
        FORECAST_DATE = result$dt_txt,
        SEASON = case_when(
          as.numeric(strftime(result$dt_txt, format="%m")) %in% 3:5 ~ "Spring",
          as.numeric(strftime(result$dt_txt, format="%m")) %in% 6:8 ~ "Summer",
          as.numeric(strftime(result$dt_txt, format="%m")) %in% 9:11 ~ "Autumn",
          TRUE ~ "Winter"
        )
      )
    }
  }

  # Combine all data frames in the list into a single data frame
  do.call(rbind, results_list)
}

# Retrieve weather data for a list of cities
cities <- c("Seoul", "Washington, D.C.", "Paris", "Suzhou")
cities_weather_df <- get_weather_forecast_by_cities(cities)

```

- Data was collected from **Global Bike-Sharing Systems Wiki Page.**
- API calls were made to "GET" wiki page
  - Once received, read_html was called to read the page
- Web scraping was used:
  - Extracting bike sharing system data
  - Converting the extracted data to a data frame
  - Exporting the data frame and storing the raw data in csv format

DATA WRANGLING

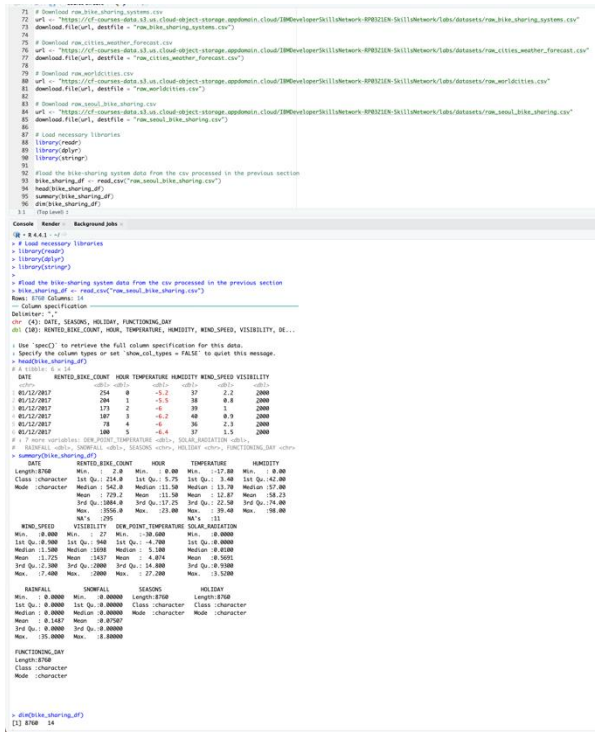
Data Wrangling w Dplyr was preferred method.

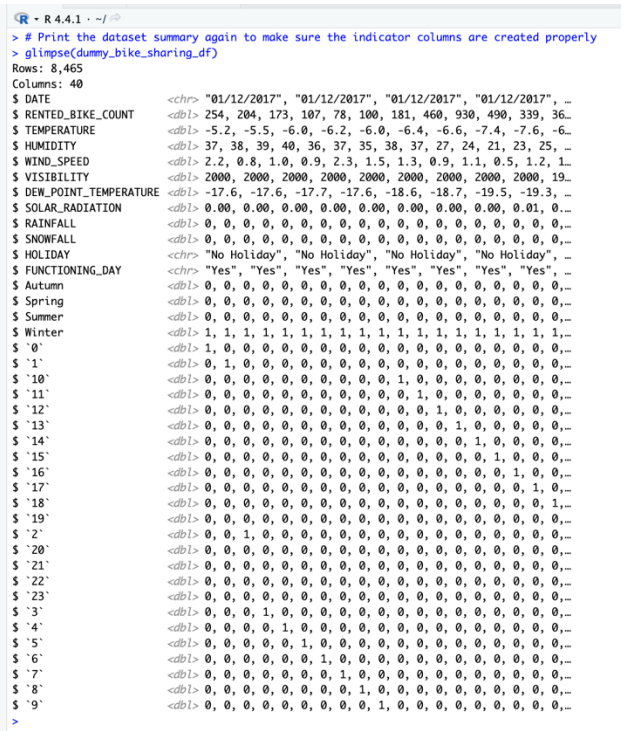
Steps:

- Detected and handled missing values
- Created indicator (dummy variables) for categorical variables
- Normalized data

Check next slide for code snippets

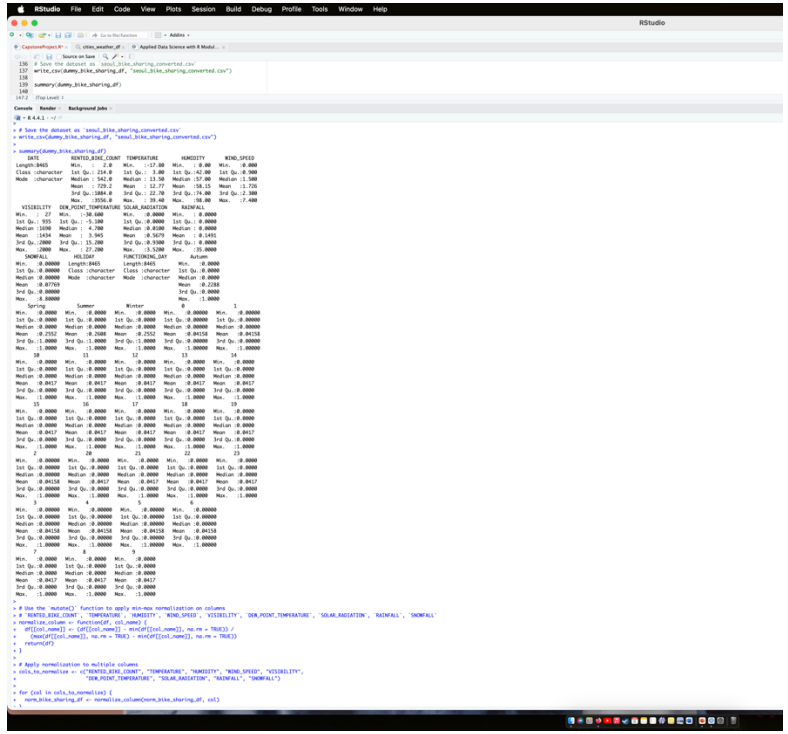
DATA WRANGLING

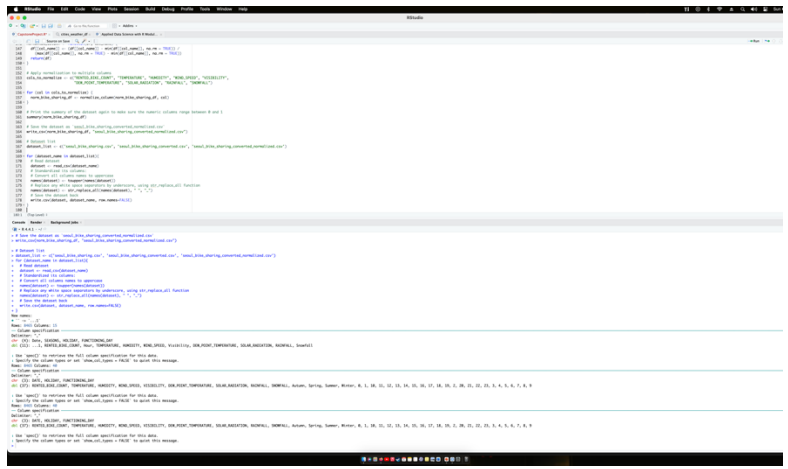




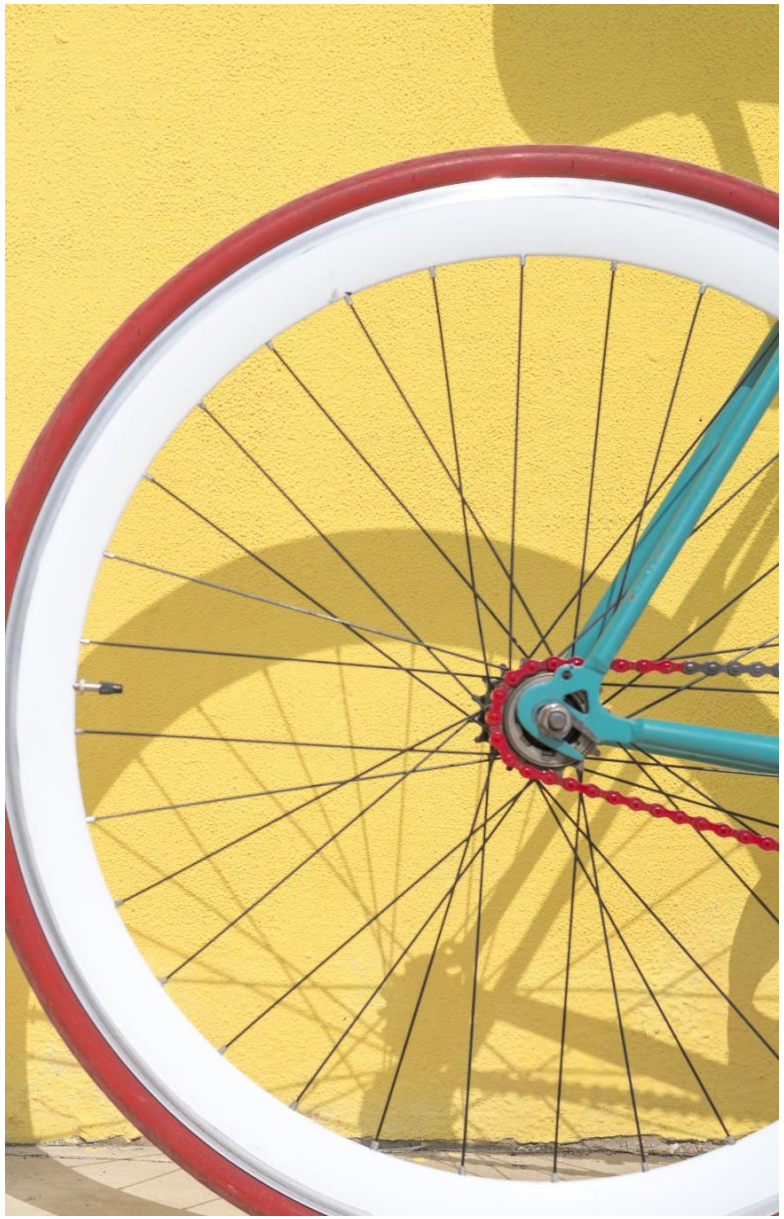
```
# Print the dataset summary again to make sure the indicator columns are created properly
glimpse(dummy_bike_sharing_df)

# Save the dataset as `seoul_bike_sharing_converted.csv`
write_csv(dummy_bike_sharing_df, "seoul_bike_sharing_converted.csv")

summary(dummy_bike_sharing_df)

# Initial DataFrame is named dummy_bike_sharing_df
norm_bike_sharing_df <- dummy_bike_sharing_df  # Initialize norm_bike_sharing_df

# Use the `mutate()` function to apply min-max normalization on columns
# `RENTED_BIKE_COUNT`, `TEMPERATURE`, `HUMIDITY`, `WIND_SPEED`, `VISIBILITY`, `DEW_POINT_TEMPERATURE`, `SOLAR_RADIATION`, `RAINFALL`, `SNOWFALL`
normalize_column <- function(df, col_name) {
  df[[col_name]] <- (df[[col_name]] - min(df[[col_name]], na.rm = TRUE)) /
    (max(df[[col_name]], na.rm = TRUE) - min(df[[col_name]], na.rm = TRUE))
  return(df)
}

# Apply normalization to multiple columns
cols_to_normalize <- c("RENTED_BIKE_COUNT", "TEMPERATURE", "HUMIDITY", "WIND_SPEED", "VISIBILITY",
                       "DEW_POINT_TEMPERATURE", "SOLAR_RADIATION", "RAINFALL", "SNOWFALL")

for (col in cols_to_normalize) {
  norm_bike_sharing_df <- normalize_column(norm_bike_sharing_df, col)
}

# Print the summary of the dataset again to make sure the numeric columns range between 0 and 1
summary(norm_bike_sharing_df)
```

```
> # Initial DataFrame is named dummy_bike_sharing_df
> norm_bike_sharing_df <- dummy_bike_sharing_df  # Initialize norm_bike_sharing_df
> # Use the `mutate()` function to apply min-max normalization on columns
> # `RENTED_BIKE_COUNT`, `TEMPERATURE`, `HUMIDITY`, `WIND_SPEED`, `VISIBILITY`, `DEW_POINT_TEMPERATURE`, `SOLAR_RADIATION`, `RAINFALL`, `SNOWFALL`
> normalize_column <- function(df, col_name) {
+   df[[col_name]] <- (df[[col_name]] - min(df[[col_name]], na.rm = TRUE)) /
+     (max(df[[col_name]], na.rm = TRUE) - min(df[[col_name]], na.rm = TRUE))
+   return(df)
+ }
> # Apply normalization to multiple columns
> cols_to_normalize <- c("RENTED_BIKE_COUNT", "TEMPERATURE", "HUMIDITY", "WIND_SPEED", "VISIBILITY",
+                        "DEW_POINT_TEMPERATURE", "SOLAR_RADIATION", "RAINFALL", "SNOWFALL")
> for (col in cols_to_normalize) {
+   norm_bike_sharing_df <- normalize_column(norm_bike_sharing_df, col)
+ }
> # Print the summary of the dataset again to make sure the numeric columns range between 0 and 1
> summary(norm_bike_sharing_df)
      DATE           RENTED_BIKE_COUNT  TEMPERATURE        HUMIDITY       WIND_SPEED      VISIBILITY     DEW_POINT_TEMPERATURE SOLAR_RADIATION      RAINFALL        SNOWFALL         HOLIDAY          FUNCTIONING_DAY       Autumn          Spring          Summer          Winter             0                1                10               11               12               13
14
 Length:8465        Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000000   Min.   :0.000000   Min.   :0.000000   Length:8465        Length:8465        Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.
0000  Min.   :0.0000
 Class :character   1st Qu.:0.05965   1st Qu.:0.3636   1st Qu.:0.4286   1st Qu.:0.1216   1st Qu.:0.4602   1st Qu.:0.4412   1st Qu.:0.000000   1st Qu.:0.000000   1st Qu.:0.000000   Class :character   Class :character   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.
0000  1st Qu.:0.0000
 Mode  :character   Median :0.15194   Median :0.5472   Median :0.5816   Median :0.2027   Median :0.8429   Median :0.6107   Median :0.002841   Median :0.000000   Median :0.000000   Mode  :character   Mode  :character   Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.
0000  Median :0.0000
                    Mean   :0.20460   Mean   :0.5345   Mean   :0.5933   Mean   :0.2332   Mean   :0.7131   Mean   :0.5977   Mean   :0.161326   Mean   :0.004261   Mean   :0.008828                                        Mean   :0.2288   Mean   :0.2552   Mean   :0.2608   Mean   :0.2552   Mean   :0.04158   Mean   :0.04158   Mean   :0.0417   Mean   :0.0417   Mean   :0.0417   Mean   :0.
0417   Mean   :0.0417
                    3rd Qu.:0.30445   3rd Qu.:0.7080   3rd Qu.:0.7551   3rd Qu.:0.3108   3rd Qu.:1.0000   3rd Qu.:0.7924   3rd Qu.:0.264205   3rd Qu.:0.000000   3rd Qu.:0.000000                                        3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.
0000  3rd Qu.:0.0000
                    Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000000   Max.   :1.000000   Max.   :1.000000                                        Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.
0000  Max.   :1.0000
       14                16                17                18                19                2                 20                21                22                23                3                 4                 5                 6                 7                 8                 9
 Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000   Median :0.00000
 Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417   Mean   :0.00158   Mean   :0.00417   Mean   :0.00158   Mean   :0.00158   Mean   :0.00417   Mean   :0.00417   Mean   :0.00417
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
>
```



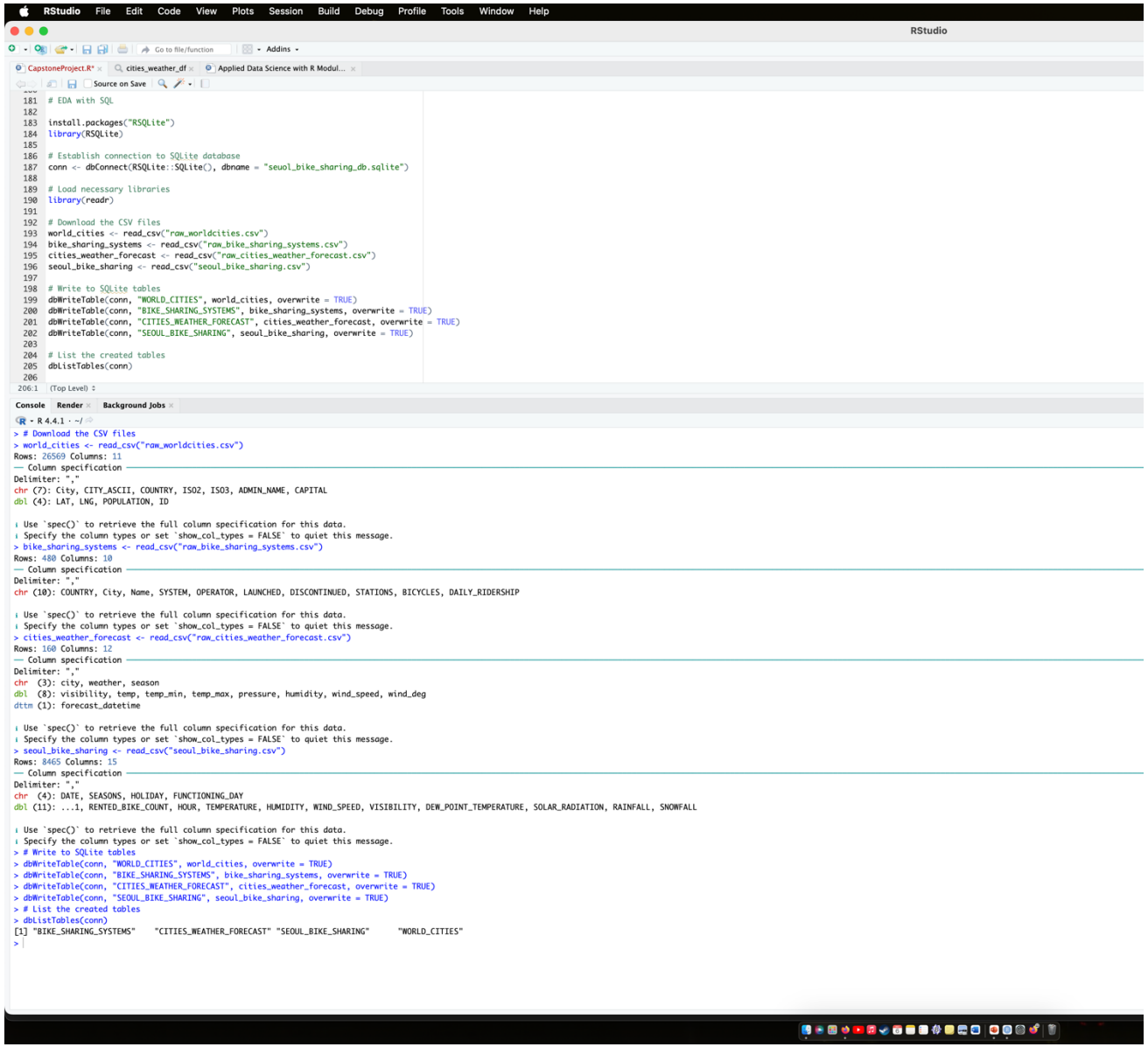


DATA WRANGLING



EDA WITH SQL

- 11 Tasks performed:
  - Record count – Found the number of records
  - Operation Hours - Found Operational Hours w/ non-zero rented bike count
  - Weather Outlook – Queried the weather forecast for Seoul over the next 3 hours
  - Seasons – Found what seasons are included in the Seoul bike sharing dataset
  - Data Range – Found the first and last dates in the Seoul Bike Sharing data set
  - Subquery – "all-time high"
  - Hourly popularity and temperature by season
  - Rental Seasonality
  - Weather Seasonality
  - Total Bike Count and City info for Seoul
  - Found all city names and coordinates with comparable bike sale to Seoul's bike sharing system
- Code Snippets in the couple next slides



INITIALIZED DB CONNECTION AND CREATED TABLES

TASK 1: RECORD COUNT – FOUND THE NUMBER OF RECORDS

```
207  # Task 1
208  record_count <- dbGetQuery(conn, "SELECT COUNT(*) AS Total_Number_of_Records FROM SEOUL_BIKE_SHARING")
209  print(record_count)
210
```

210:1 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 1
> record_count <- dbGetQuery(conn, "SELECT COUNT(*) AS Total_Number_of_Records FROM SEOUL_BIKE_SHARING")
> print(record_count)
  Total_Number_of_Records
1                    8465
>
```

TASK 2: OPERATION HOURS - FOUND OPERATIONAL HOURS W/ NON-ZERO RENTED BIKE COUNT

```
211  # Task 2
212  non_zero_operational__hours <- dbGetQuery(conn, "SELECT COUNT(DISTINCT HOUR) AS Non_Zero__Operartional_Hours
213                                     FROM SEOUL_BIKE_SHARING
214                                     WHERE RENTED_BIKE_COUNT > 0")
215  print(non_zero_operational__hours)
216:1  (Top Level)
```

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 2
> non_zero_operational__hours <- dbGetQuery(conn, "SELECT COUNT(DISTINCT HOUR) AS Non_Zero__Operartional_Hours
+                                     FROM SEOUL_BIKE_SHARING
+                                     WHERE RENTED_BIKE_COUNT > 0")
> print(non_zero_operational__hours)
  Non_Zero__Operartional_Hours
1                           24
>
```

TASK 3: WEATHER OUTLOOK – QUERIED THE WEATHER FORECAST FOR SEOUL OVER THE NEXT 3 HOURS

```
217  # Task 3
218  weather_forecast_seoul <- dbGetQuery(conn, "SELECT * FROM CITIES_WEATHER_FORECAST
219                                              WHERE CITY = 'Seoul'
220                                              ORDER BY FORECAST_DATETIME
221                                              LIMIT 1")
222  print(weather_forecast_seoul)
```

222:30 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 3
> weather_forecast_seoul <- dbGetQuery(conn, "SELECT * FROM CITIES_WEATHER_FORECAST
+                                             WHERE CITY = 'Seoul'
+                                             ORDER BY FORECAST_DATETIME
+                                             LIMIT 1")
> print(weather_forecast_seoul)
   city weather visibility  temp temp_min temp_max pressure humidity wind_speed wind_deg season forecast_datetime
1 Seoul   Clear      10000 12.32    10.91    12.32     1015       50       2.18      248 Spring        1618574400
>
```

```
224  # Task 4
225  seasons <- dbGetQuery(conn, "SELECT DISTINCT(SEASONS)
226                               FROM SEOUL_BIKE_SHARING
227                               ORDER BY SEASONS ASC")
228  print(seasons)
229
```

229:1 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 4
> seasons <- dbGetQuery(conn, "SELECT DISTINCT(SEASONS)
+                               FROM SEOUL_BIKE_SHARING
+                               ORDER BY SEASONS ASC")
> print(seasons)
  SEASONS
1  Autumn
2  Spring
3  Summer
4  Winter
>
```

TASK 4: SEASONS – FOUND WHAT SEASONS ARE INCLUDED IN THE SEOUL BIKE SHARING DATASET

TASK 5: DATA RANGE – FOUND THE FIRST AND LAST DATES IN THE SEOUL BIKE SHARING DATA SET

```
230 # Task 5
231 date_range <- dbGetQuery(conn, "SELECT MIN(DATE) AS first_date_of_Sbiking, MAX(DATE) AS last_date_of_Sbiking
232                                  FROM SEOUL_BIKE_SHARING")
233 print(date_range)
234
```

234:1 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 5
> date_range <- dbGetQuery(conn, "SELECT MIN(DATE) AS first_date_of_Sbiking, MAX(DATE) AS last_date_of_Sbiking
+                                  FROM SEOUL_BIKE_SHARING")
> print(date_range)
  first_date_of_Sbiking last_date_of_Sbiking
1            01/01/2018           31/12/2017
>
```

TASK 6: SUBQUERY – "ALL-TIME HIGH"

```
235  # Task 6
236  all_time_high <- dbGetQuery(conn, "SELECT DATE, HOUR, MAX(RENTED_BIKE_COUNT) AS max_rentals
237                                     FROM SEOUL_BIKE_SHARING
238                                     GROUP BY DATE, HOUR
239                                     ORDER BY max_rentals DESC
240                                     LIMIT 1")
241  print(all_time_high)
242
242:1  (Top Level)
```

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 6
> all_time_high <- dbGetQuery(conn, "SELECT DATE, HOUR, MAX(RENTED_BIKE_COUNT) AS max_rentals
+                                    FROM SEOUL_BIKE_SHARING
+                                    GROUP BY DATE, HOUR
+                                    ORDER BY max_rentals DESC
+                                    LIMIT 1")
> print(all_time_high)
        DATE HOUR max_rentals
1 19/06/2018   18        3556
>
```

```
243  # Task 7
244  avg_hourly_season <- dbGetQuery(conn, "SELECT SEASONS,HOUR,
245                                         AVG(TEMPERATURE) AS avg_temperature,
246                                         AVG(RENTED_BIKE_COUNT) AS avg_bike_cour
247                                       FROM SEOUL_BIKE_SHARING
248                                       GROUP BY SEASONS, HOUR
249                                       ORDER BY AVG(RENTED_BIKE_COUNT) DESC
250                                       LIMIT 10")
251  print(avg_hourly_season)
252
```

252:1 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 7
> avg_hourly_season <- dbGetQuery(conn, "SELECT SEASONS,HOUR,
+                                         AVG(TEMPERATURE) AS avg_temperature,
+                                         AVG(RENTED_BIKE_COUNT) AS avg_bike_count
+                                       FROM SEOUL_BIKE_SHARING
+                                       GROUP BY SEASONS, HOUR
+                                       ORDER BY AVG(RENTED_BIKE_COUNT) DESC
+                                       LIMIT 10")
> print(avg_hourly_season)
   SEASONS HOUR avg_temperature avg_bike_count
1   Summer   18        29.38791       2135.141
2   Autumn   18        16.03185       1983.333
3   Summer   19        28.27378       1889.250
4   Summer   20        27.06630       1801.924
5   Summer   21        26.27826       1754.065
6   Spring   18        15.97222       1689.311
7   Summer   22        25.69891       1567.870
8   Autumn   17        17.27778       1562.877
9   Summer   17        30.07691       1526.293
10  Autumn   19        15.06346       1515.568
>
```

TASK 7: HOURLY POPULARITY AND TEMPERATURE BY SEASON

```
253  # Task 8
254  rental_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
255                                          AVG(RENTED_BIKE_COUNT) AS avg_bike_count,
256                                          MIN(RENTED_BIKE_COUNT) AS min_bike_count,
257                                          MAX(RENTED_BIKE_COUNT) AS max_bike_count
258                                   FROM SEOUL_BIKE_SHARING
259                                   GROUP BY SEASONS")
260  print(rental_seasonality)
261
```

261:1 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 8
> rental_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
+                                         AVG(RENTED_BIKE_COUNT) AS avg_bike_count,
+                                         MIN(RENTED_BIKE_COUNT) AS min_bike_count,
+                                         MAX(RENTED_BIKE_COUNT) AS max_bike_count
+                                  FROM SEOUL_BIKE_SHARING
+                                  GROUP BY SEASONS")
> print(rental_seasonality)
  SEASONS avg_bike_count min_bike_count max_bike_count
1  Autumn       924.1105              2           3298
2  Spring       746.2542              2           3251
3  Summer      1034.0734              9           3556
4  Winter       225.5412              3            937
>
```

TASK 8: RENTAL SEASONALITY

TASK 9: WEATHER SEASONALITY

```
261
262 # Task 9
263 weather_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
264                                           AVG(TEMPERATURE) AS avg_temperature,
265                                           AVG(HUMIDITY) AS avg_humidity,
266                                           AVG(WIND_SPEED) AS avg_wind_speed,
267                                           AVG(VISIBILITY) AS avg_visibility,
268                                           AVG(DEW_POINT_TEMPERATURE) AS avg_dew_point_temp,
269                                           AVG(SOLAR_RADIATION) AS avg_solar_radiation,
270                                           AVG(RAINFALL) AS avg_rainfall,
271                                           AVG(SNOWFALL) AS avg_snowfall
272                                    FROM SEOUL_BIKE_SHARING
273                                    GROUP BY SEASONS
274                                    ORDER BY RENTED_BIKE_COUNT DESC")
275 print(weather_seasonality)
276
276:2  (Top Level)
```

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 9
> weather_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
+                                           AVG(TEMPERATURE) AS avg_temperature,
+                                           AVG(HUMIDITY) AS avg_humidity,
+                                           AVG(WIND_SPEED) AS avg_wind_speed,
+                                           AVG(VISIBILITY) AS avg_visibility,
+                                           AVG(DEW_POINT_TEMPERATURE) AS avg_dew_point_temp,
+                                           AVG(SOLAR_RADIATION) AS avg_solar_radiation,
+                                           AVG(RAINFALL) AS avg_rainfall,
+                                           AVG(SNOWFALL) AS avg_snowfall
+                                    FROM SEOUL_BIKE_SHARING
+                                    GROUP BY SEASONS
+                                    ORDER BY RENTED_BIKE_COUNT DESC")
> print(weather_seasonality)
  SEASONS avg_temperature avg_humidity avg_wind_speed avg_visibility avg_dew_point_temp avg_solar_radiation avg_rainfall avg_snowfall
1  Autumn       13.821580     59.04491       1.492101       1558.174           5.150594           0.5227827   0.11765617   0.06350026
2  Summer       26.587711     64.98143       1.609420       1501.745          18.750136           0.7612545   0.25348732   0.00000000
3  Winter       -2.540463     49.74491       1.922685       1445.987         -12.416667           0.2981806   0.03282407   0.24750000
4  Spring       13.021685     58.75833       1.857778       1240.912           4.091389           0.6803009   0.18694444   0.00000000
>
```

TASK 10: TOTAL BIKE COUNT AND CITY INFO FOR SEOUL

```
277  # Task 10
278  total_bike_count_seoul <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
279                                                     b.BICYCLES
280                                              FROM WORLD_CITIES wc
281                                              LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
282                                              WHERE wc.CITY = 'Seoul'")
283  print(total_bike_count_seoul)
284
```

283:30 (Top Level)

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 10
> total_bike_count_seoul <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
+                                                    b.BICYCLES
+                                             FROM WORLD_CITIES wc
+                                             LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
+                                             WHERE wc.CITY = 'Seoul'")
> print(total_bike_count_seoul)
   City      COUNTRY     LAT LNG POPULATION BICYCLES
1 Seoul Korea, South 37.5833 127   21794000     <NA>
>
```

```
285  # Task 11
286  similar_cities <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
287                                             b.BICYCLES
288                                      FROM WORLD_CITIES wc
289                                      LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
290                                      WHERE b.BICYCLES BETWEEN 15000 AND 20000")
291  print(similar_cities)
291:22  (Top Level)
```

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 11
> similar_cities <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
+                                             b.BICYCLES
+                                      FROM WORLD_CITIES wc
+                                      LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
+                                      WHERE b.BICYCLES BETWEEN 15000 AND 20000")
> print(similar_cities)
```

	City	COUNTRY	LAT	LNG	POPULATION	BICYCLES
1	Tirana	Albania	41.3300	19.8200	418495	200
2	Sydney	Australia	-33.8650	151.2094	5312163	2000
3	Beijing	China	39.9050	116.3914	19433000	16000
4	Heze	China	35.2333	115.4333	8750000	2000
5	Lanzhou	China	36.0617	103.8318	3616163	2000
6	Ningbo	China	29.8750	121.5492	7639000	15000
7	Weifang	China	36.7167	119.1000	9373000	20000
8	Zhuzhou	China	27.8407	113.1469	3855609	20000
9	Ostrava	Czechia	49.8356	18.2925	287968	180
10	Prostějov	Czechia	49.4722	17.1106	43651	180[76]
11	Copenhagen	Denmark	55.6786	12.5635	1085000	1860
12	Belfort	France	47.6400	6.8500	114445	200
13	Bordeaux	France	44.8400	-0.5800	257804	1545
14	Calais	France	50.9481	1.8564	73911	160
15	Dunkirk	United States	42.4803	-79.3323	23874	200
16	Nice	France	43.7034	7.2663	1006402	1750
17	Strasbourg	France	48.5833	7.7458	465069	1852
18	Vannes	France	47.6559	-2.7603	53352	174
19	Budapest	Hungary	47.4983	19.0408	1752286	1526[159]
20	Győr	Hungary	47.6842	17.6344	132034	180[161]
21	Rome	United States	34.2661	-85.1862	61537	200
22	Rome	Italy	41.8931	12.4828	2872800	200
23	Rome	United States	43.2260	-75.4909	31863	200
24	Almaty	Kazakhstan	43.2500	76.9000	1806833	1700
25	Shymkent	Kazakhstan	42.3000	69.6000	1018974	200
26	Trondheim	Norway	63.4400	10.4000	183378	200
27	Opole	Poland	50.6722	17.9253	127792	164
28	Changhua	Taiwan	24.0667	120.5333	750000	1695
29	Austin	United States	30.3004	-97.7522	1687311	200
30	Austin	United States	43.6718	-92.9783	25626	200

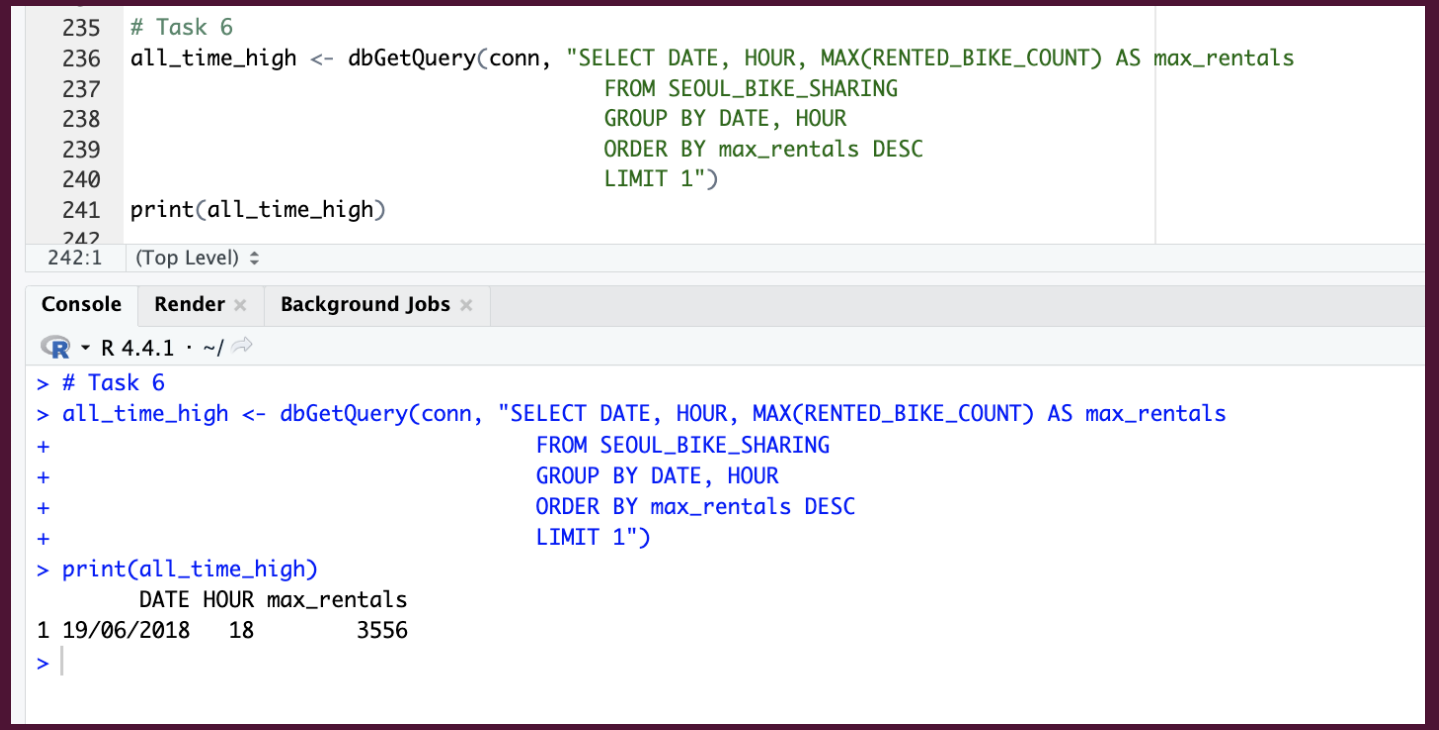
```
>
```

TASK 11:FOUND ALL CITY NAMES AND COORDINATES WITH COMPARABLE BIKE SALE TO SEOUL'S BIKE SHARING SYSTEM

EXPLORATORY DATA ANALYSIS RESULTS

BUSIEST BIKE RENTAL TIMES

- The peak bike rental times were as follows:
  - Date: June 19, 2018
  - Hour: 18:00
  - Maximum Rentals: 3,556



HOURLY POPULARITY AND TEMPERATURE BY SEASONS

- Based on the top 10 findings:
  - The Summer season experienced the highest number of bike rentals at 18:00.
  - The Autumn season recorded the lowest average temperature at 19:00.
- In summary, Summer had the highest average bike rentals compared to all other seasons.

```
243  # Task 7
244  avg_hourly_season <- dbGetQuery(conn, "SELECT SEASONS,HOUR,
245                                                AVG(TEMPERATURE) AS avg_temperature,
246                                                AVG(RENTED_BIKE_COUNT) AS avg_bike_cour
247                                        FROM SEOUL_BIKE_SHARING
248                                        GROUP BY SEASONS, HOUR
249                                        ORDER BY AVG(RENTED_BIKE_COUNT) DESC
250                                        LIMIT 10")
251  print(avg_hourly_season)
252
252:1  (Top Level)
```

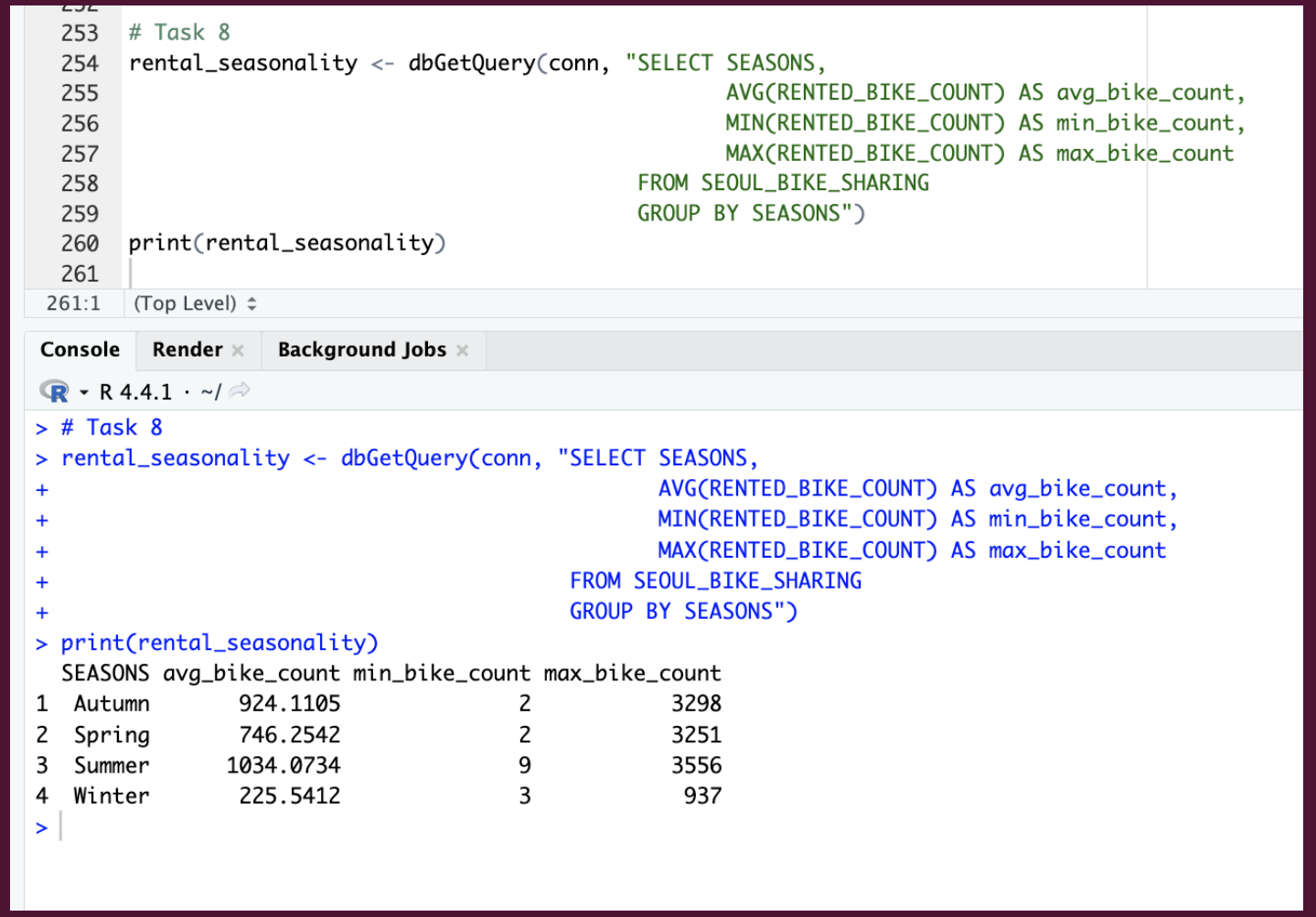
Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 7
> avg_hourly_season <- dbGetQuery(conn, "SELECT SEASONS,HOUR,
+                                                AVG(TEMPERATURE) AS avg_temperature,
+                                                AVG(RENTED_BIKE_COUNT) AS avg_bike_count
+                                        FROM SEOUL_BIKE_SHARING
+                                        GROUP BY SEASONS, HOUR
+                                        ORDER BY AVG(RENTED_BIKE_COUNT) DESC
+                                        LIMIT 10")
> print(avg_hourly_season)
   SEASONS HOUR avg_temperature avg_bike_count
1   Summer   18        29.38791       2135.141
2   Autumn   18        16.03185       1983.333
3   Summer   19        28.27378       1889.250
4   Summer   20        27.06630       1801.924
5   Summer   21        26.27826       1754.065
6   Spring   18        15.97222       1689.311
7   Summer   22        25.69891       1567.870
8   Autumn   17        17.27778       1562.877
9   Summer   17        30.07691       1526.293
10  Autumn   19        15.06346       1515.568
>
```

RENTAL SEASONALITY

The highest number of bike rentals occurred during the Summer, with a total of 1,034 rentals. This season also saw both the maximum and minimum number of bikes rented overall. In contrast, the Winter season recorded the fewest bike rentals, totaling just 226.



WEATHER SEASONALITY

Both Summer and Autumn experienced the highest recorded temperatures and the most bike rentals. In Summer, there were 1,034 rentals at an average temperature of 26.6°F, while Autumn saw 924 rentals with an average temperature of 13.8°F. Winter, on the other hand, recorded the lowest temperatures, which likely contributed to a decrease in bike rentals due to adverse weather conditions such as snowfall and high wind speeds.

```
253 # Task 8
254 rental_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
255                                          AVG(RENTED_BIKE_COUNT) AS avg_bike_count,
256                                          MIN(RENTED_BIKE_COUNT) AS min_bike_count,
257                                          MAX(RENTED_BIKE_COUNT) AS max_bike_count
258                                   FROM SEOUL_BIKE_SHARING
259                                   GROUP BY SEASONS")
260 print(rental_seasonality)
261
```

Console | Render | Background Jobs

```
> # Task 8
> rental_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
+                                          AVG(RENTED_BIKE_COUNT) AS avg_bike_count,
+                                          MIN(RENTED_BIKE_COUNT) AS min_bike_count,
+                                          MAX(RENTED_BIKE_COUNT) AS max_bike_count
+                                   FROM SEOUL_BIKE_SHARING
+                                   GROUP BY SEASONS")
> print(rental_seasonality)
  SEASONS avg_bike_count min_bike_count max_bike_count
1  Autumn       924.1105              2           3298
2  Spring       746.2542              2           3251
3  Summer      1034.0734              9           3556
4  Winter       225.5412              3            937
>
```

```
262 # Task 9
263 weather_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
264                                          AVG(TEMPERATURE) AS avg_temperature,
265                                          AVG(HUMIDITY) AS avg_humidity,
266                                          AVG(WIND_SPEED) AS avg_wind_speed,
267                                          AVG(VISIBILITY) AS avg_visibility,
268                                          AVG(DEW_POINT_TEMPERATURE) AS avg_dew_point_temp,
269                                          AVG(SOLAR_RADIATION) AS avg_solar_radiation,
270                                          AVG(RAINFALL) AS avg_rainfall,
271                                          AVG(SNOWFALL) AS avg_snowfall
272                                   FROM SEOUL_BIKE_SHARING
273                                   GROUP BY SEASONS
274                                   ORDER BY RENTED_BIKE_COUNT DESC")
275 print(weather_seasonality)
276 3
```

Console | Render | Background Jobs

```
> # Task 9
> weather_seasonality <- dbGetQuery(conn, "SELECT SEASONS,
+                                          AVG(TEMPERATURE) AS avg_temperature,
+                                          AVG(HUMIDITY) AS avg_humidity,
+                                          AVG(WIND_SPEED) AS avg_wind_speed,
+                                          AVG(VISIBILITY) AS avg_visibility,
+                                          AVG(DEW_POINT_TEMPERATURE) AS avg_dew_point_temp,
+                                          AVG(SOLAR_RADIATION) AS avg_solar_radiation,
+                                          AVG(RAINFALL) AS avg_rainfall,
+                                          AVG(SNOWFALL) AS avg_snowfall
+                                   FROM SEOUL_BIKE_SHARING
+                                   GROUP BY SEASONS
+                                   ORDER BY RENTED_BIKE_COUNT DESC")
> print(weather_seasonality)
  SEASONS avg_temperature avg_humidity avg_wind_speed avg_visibility avg_dew_point_temp avg_solar_radiation avg_rainfall avg_snowfall
1  Autumn       13.821580     59.04491       1.492101       1558.174          5.150594          0.5227827   0.11765617   0.06350026
2  Summer       26.587711     64.98143       1.609420       1501.745         18.750136          0.7612545   0.25348732   0.00000000
3  Winter       -2.540463     49.74491       1.922685       1445.987        -12.416667          0.2981806   0.03282407   0.24750000
4  Spring       13.021685     58.75833       1.857778       1240.912          4.091389          0.6803009   0.18694444   0.00000000
>
```

BIKE-SHARING INFO IN SEOUL

- Country: South Korea
- City: Seoul
- Population: 21,794,000

```
277  # Task 10
278  total_bike_count_seoul <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
279                                                     b.BICYCLES
280                                              FROM WORLD_CITIES wc
281                                              LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
282                                              WHERE wc.CITY = 'Seoul'")
283  print(total_bike_count_seoul)
284
283:30  (Top Level)

Console  Render  Background Jobs

R 4.4.1 · ~/

> # Task 10
> total_bike_count_seoul <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
+                                                     b.BICYCLES
+                                              FROM WORLD_CITIES wc
+                                              LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
+                                              WHERE wc.CITY = 'Seoul'")
> print(total_bike_count_seoul)
   City      COUNTRY     LAT LNG POPULATION BICYCLES
1 Seoul Korea, South 37.5833 127   21794000     <NA>
>
```

CITIES SIMILAR TO SEOUL

As population size grows, the number of bike rentals tends to rise as well. Cities like Shanghai and Beijing record the highest levels of bike rentals.

```
285  # Task 11
286  similar_cities <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
287                                             b.BICYCLES
288                                      FROM WORLD_CITIES wc
289                                      LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
290                                      WHERE b.BICYCLES BETWEEN 15000 AND 20000")
291  print(similar_cities)
291:22  (Top Level)
```

Console | Render | Background Jobs

R 4.4.1 · ~/

```
> # Task 11
> similar_cities <- dbGetQuery(conn, "SELECT wc.CITY, wc.COUNTRY, wc.LAT, wc.LNG, wc.POPULATION,
+                                             b.BICYCLES
+                                      FROM WORLD_CITIES wc
+                                      LEFT JOIN BIKE_SHARING_SYSTEMS b ON wc.CITY = b.CITY
+                                      WHERE b.BICYCLES BETWEEN 15000 AND 20000")
> print(similar_cities)
```

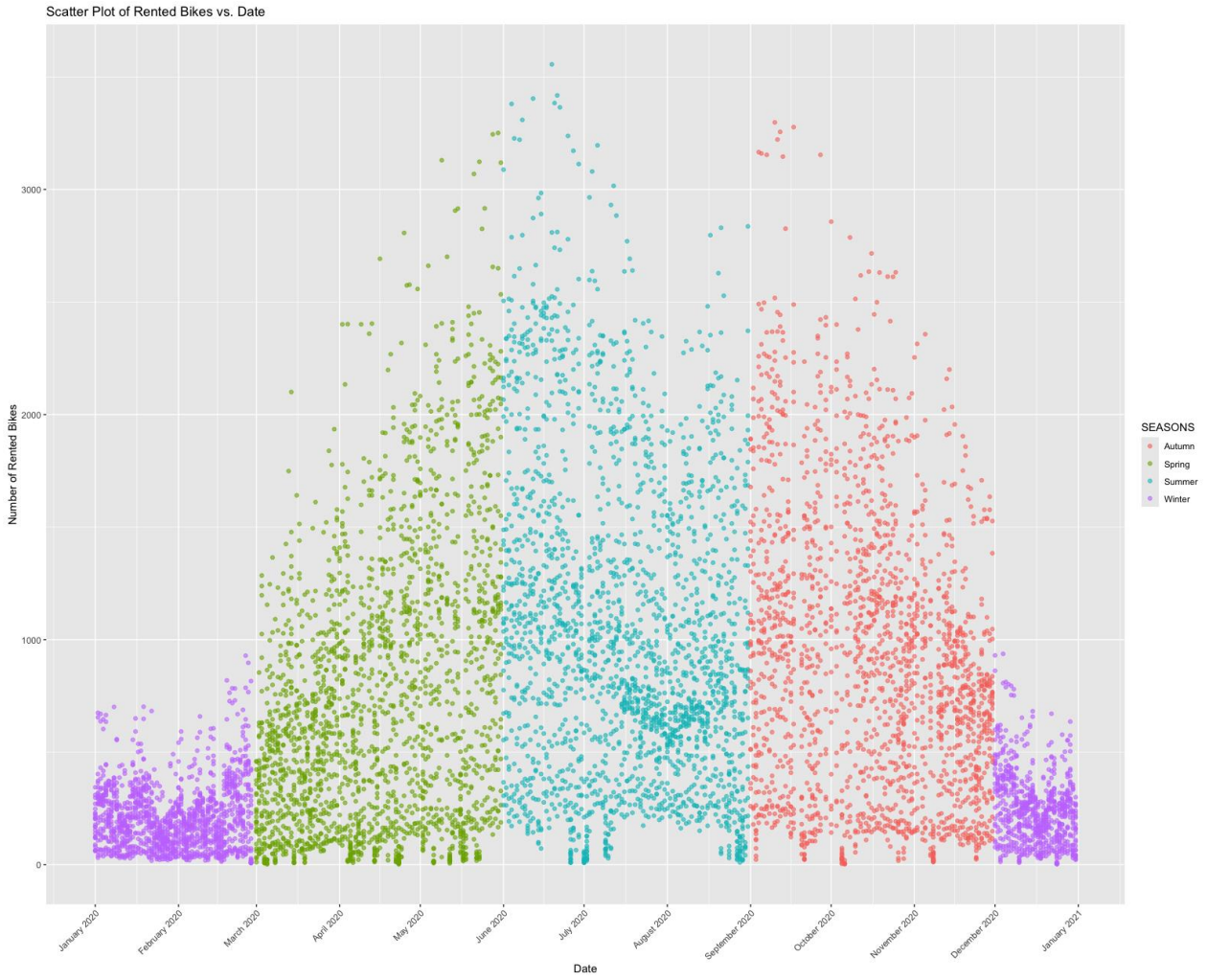
	City	COUNTRY	LAT	LNG	POPULATION	BICYCLES
1	Tirana	Albania	41.3300	19.8200	418495	200
2	Sydney	Australia	-33.8650	151.2094	5312163	2000
3	Beijing	China	39.9050	116.3914	19433000	16000
4	Heze	China	35.2333	115.4333	8750000	2000
5	Lanzhou	China	36.0617	103.8318	3616163	2000
6	Ningbo	China	29.8750	121.5492	7639000	15000
7	Weifang	China	36.7167	119.1000	9373000	20000
8	Zhuzhou	China	27.8407	113.1469	3855609	20000
9	Ostrava	Czechia	49.8356	18.2925	287968	180
10	Prostějov	Czechia	49.4722	17.1106	43651	180[76]
11	Copenhagen	Denmark	55.6786	12.5635	1085000	1860
12	Belfort	France	47.6400	6.8500	114445	200
13	Bordeaux	France	44.8400	-0.5800	257804	1545
14	Calais	France	50.9481	1.8564	73911	160
15	Dunkirk	United States	42.4803	-79.3323	23874	200
16	Nice	France	43.7034	7.2663	1006402	1750
17	Strasbourg	France	48.5833	7.7458	465069	1852
18	Vannes	France	47.6559	-2.7603	53352	174
19	Budapest	Hungary	47.4983	19.0408	1752286	1526[159]
20	Győr	Hungary	47.6842	17.6344	132034	180[161]
21	Rome	United States	34.2661	-85.1862	61537	200
22	Rome	Italy	41.8931	12.4828	2872800	200
23	Rome	United States	43.2260	-75.4909	31863	200
24	Almaty	Kazakhstan	43.2500	76.9000	1806833	1700
25	Shymkent	Kazakhstan	42.3000	69.6000	1018974	200
26	Trondheim	Norway	63.4400	10.4000	183378	200
27	Opole	Poland	50.6722	17.9253	127792	164
28	Changhua	Taiwan	24.0667	120.5333	750000	1695
29	Austin	United States	30.3004	-97.7522	1687311	200
30	Austin	United States	43.6718	-92.9783	25626	200

```
>
```

EDA WITH DATA VISUALIZATION

BIKE RENTAL VS. DATETIME

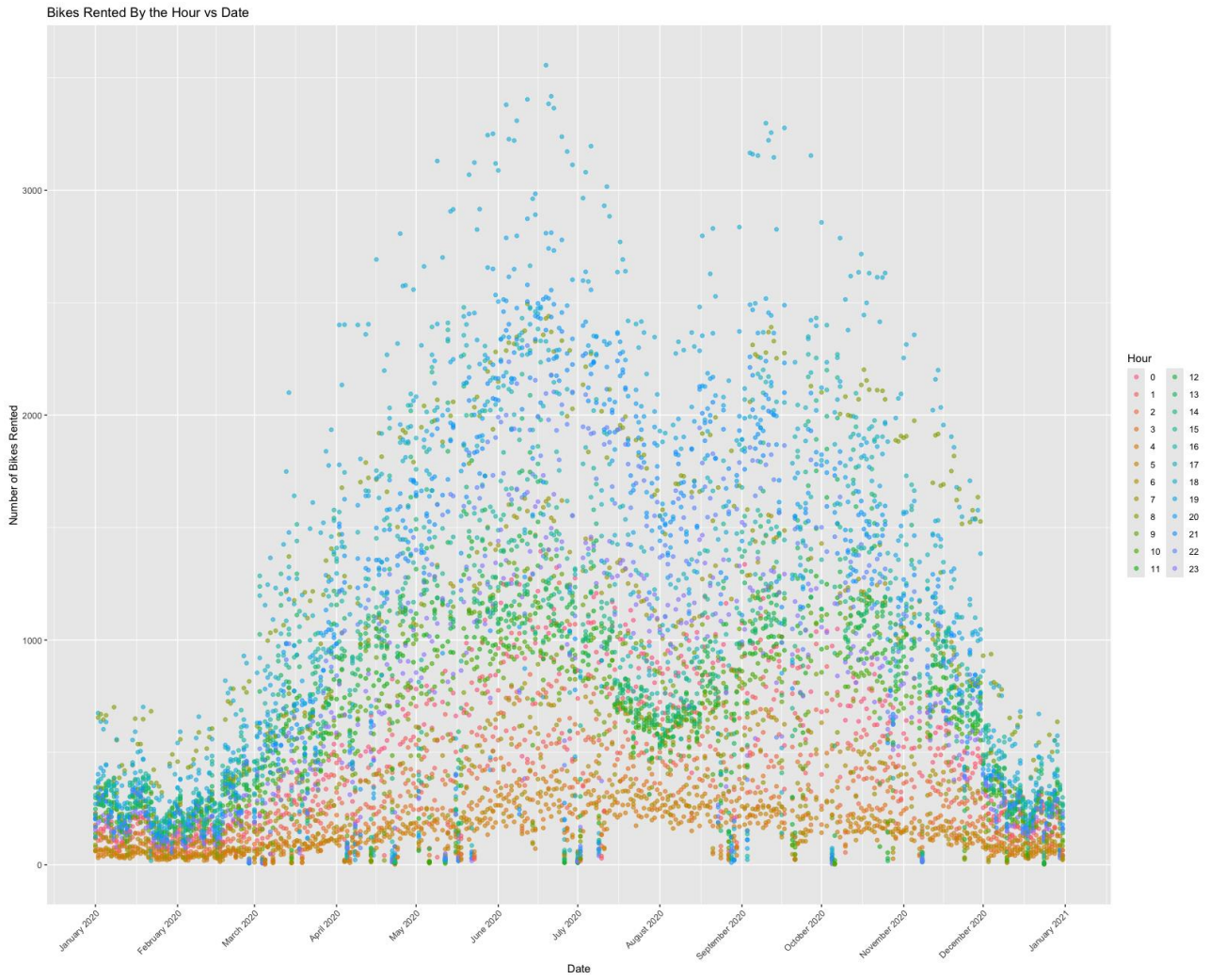
The seasonal trends in the data reveal that bike rentals reach their highest levels during the summer and their lowest levels in the winter.

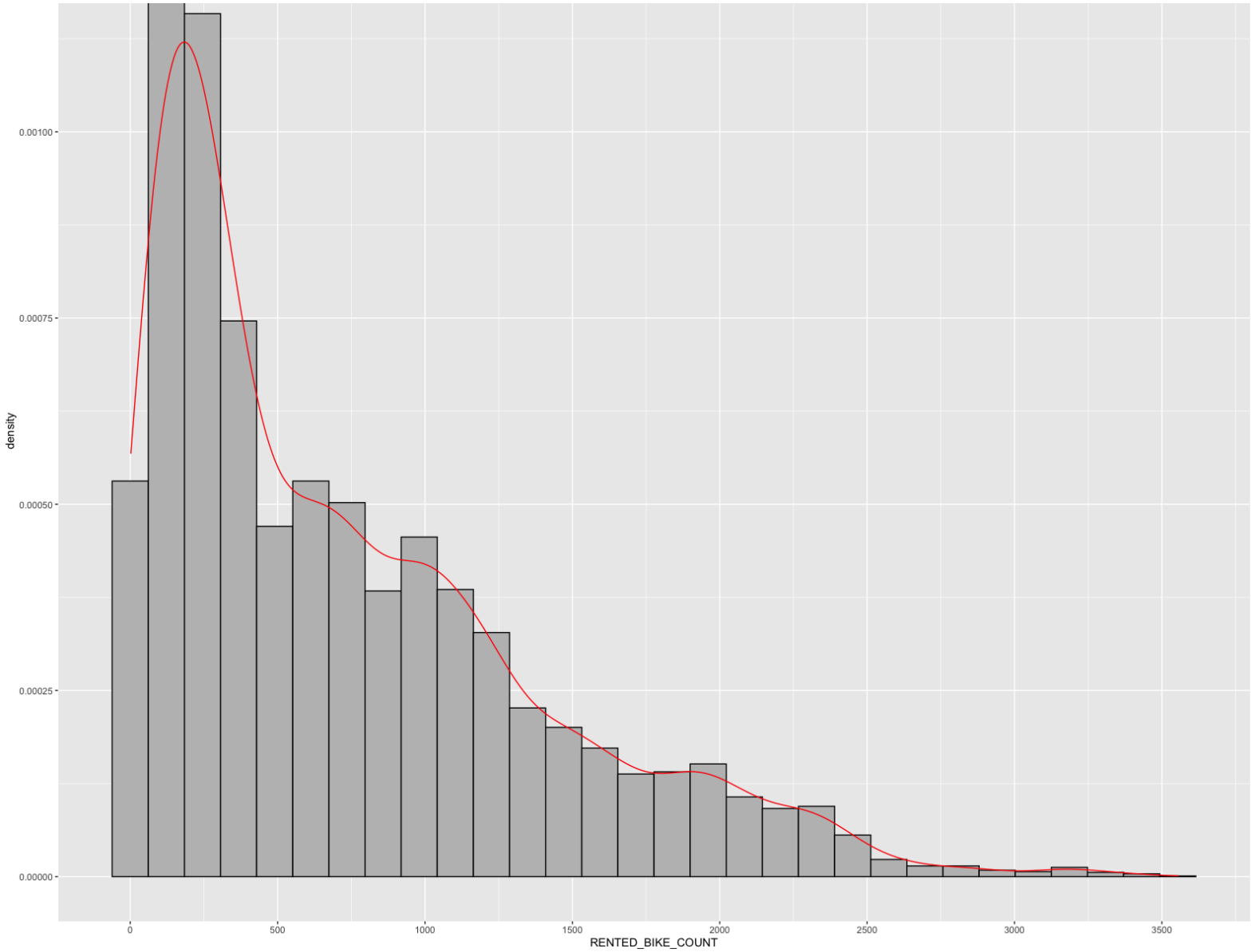


BIKE RENTAL VS. DATETIME

Bike rentals tend to rise throughout the day, particularly between the hours of 8 AM and 6-7 PM.

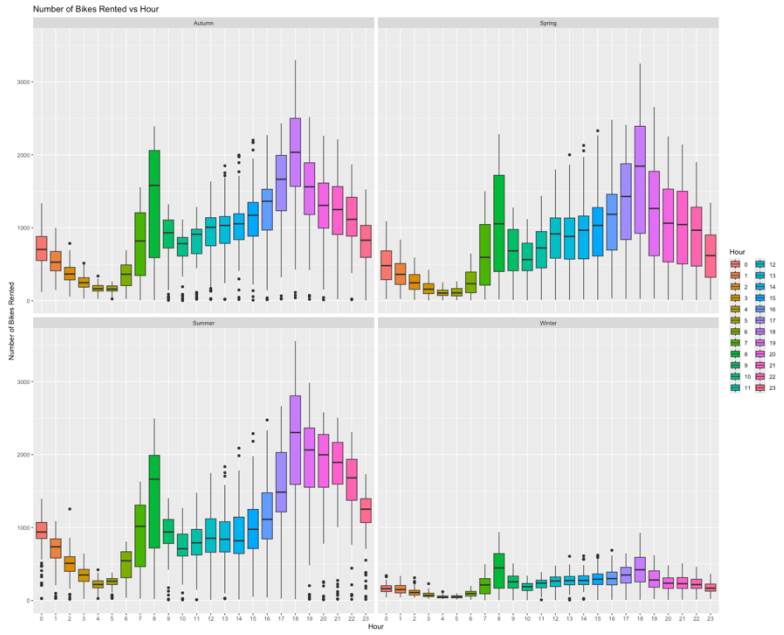
The seasonal trends in the data reveal that bike rentals reach their highest levels during the summer and their lowest levels in the winter.

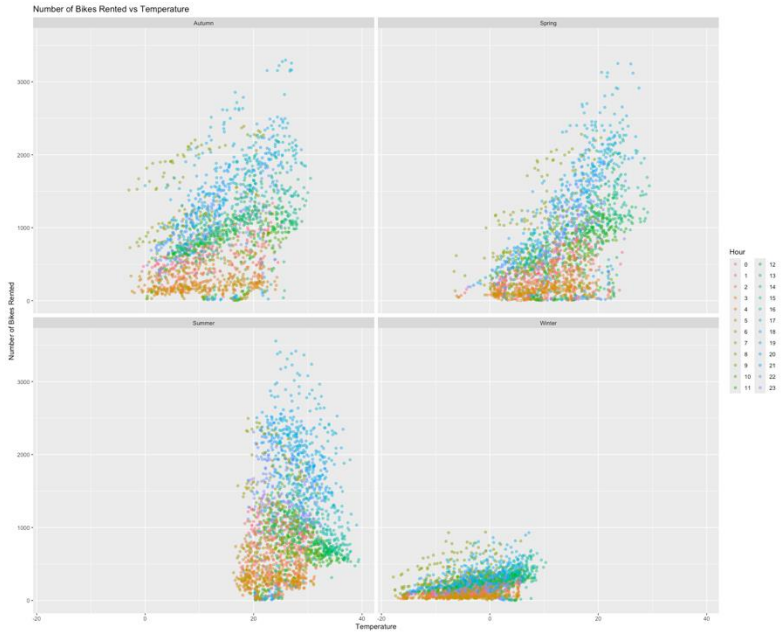


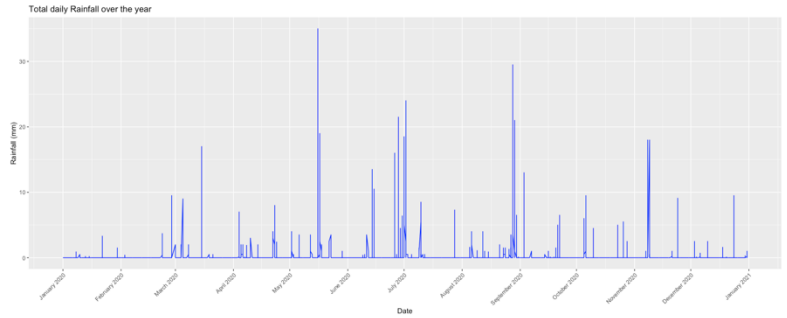


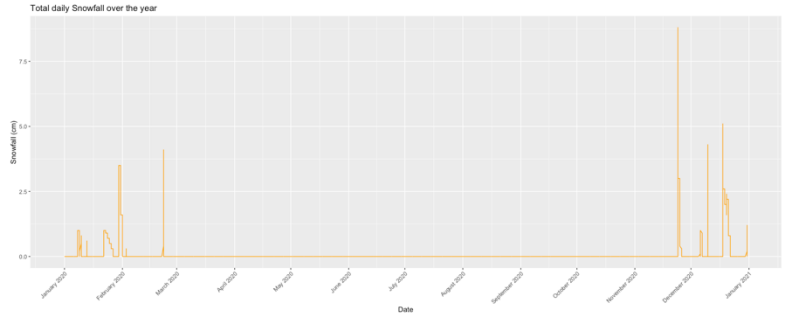
BIKE RENTAL - HISTOGRAM

DAILY TOTAL SNOW AND RAINFALL





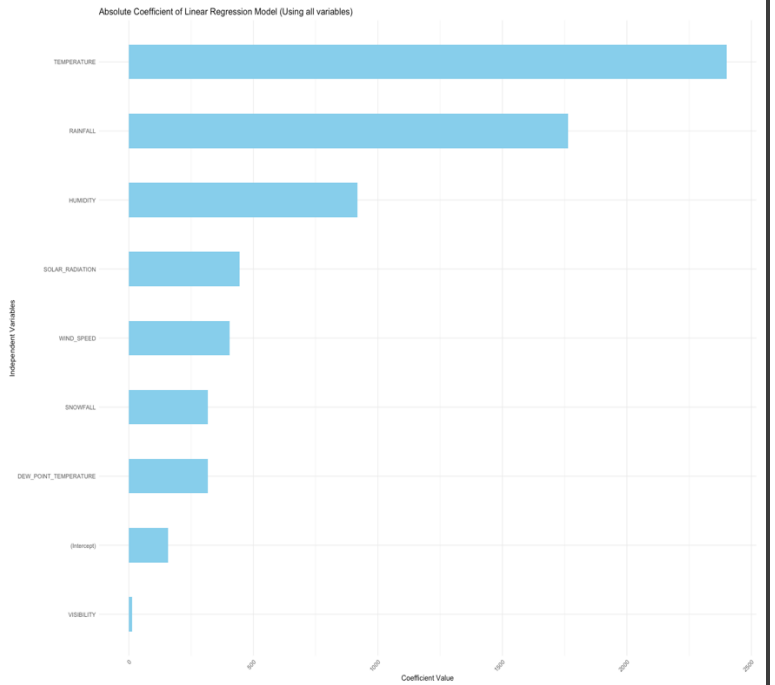




PREDICTIVE ANALYSIS

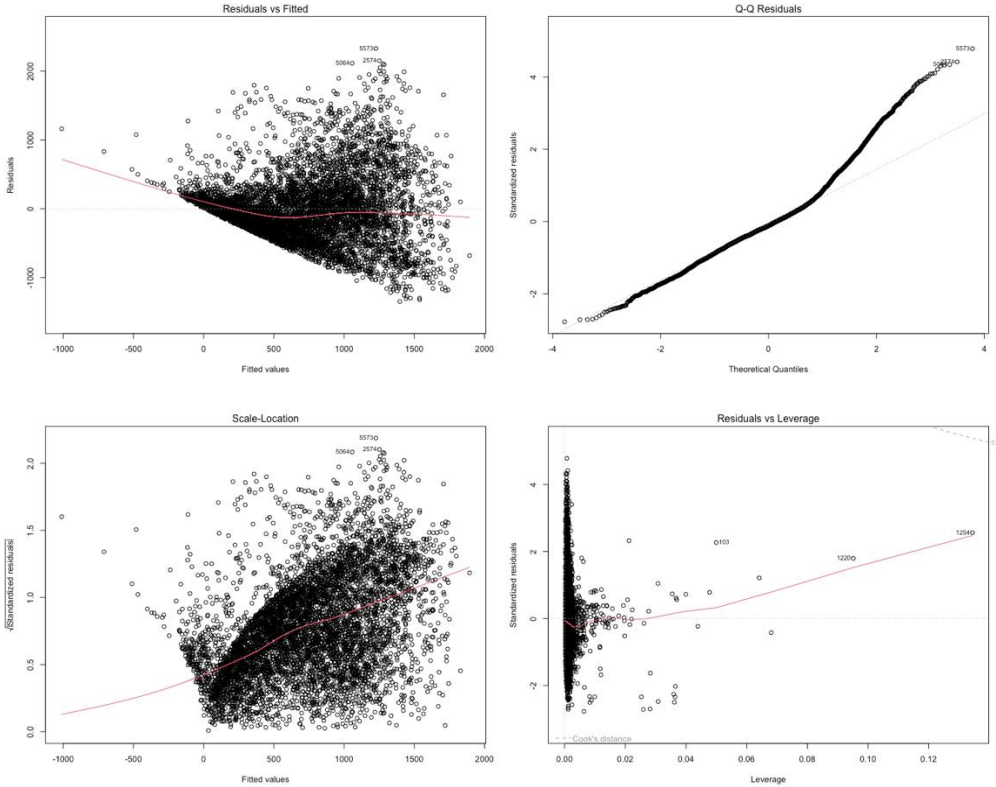
A tibble: 9 × 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
1 TEMPERATURE	2400.	262.	9.17	6.20e-20
2 RAINFALL	-1764.	183.	-9.66	6.43e-22
3 HUMIDITY	-918.	127.	-7.24	4.90e-13
4 SOLAR_RADIATION	-445.	34.7	-12.8	3.47e-37
5 WIND_SPEED	404.	48.2	8.40	5.52e-17
6 SNOWFALL	318.	132.	2.42	1.58e- 2
7 DEW_POINT_TEMPERATURE	-317.	279.	-1.14	2.56e- 1
8 (Intercept)	157.	58.1	2.70	6.98e- 3
9 VISIBILITY	12.6	24.9	0.505	6.14e- 1

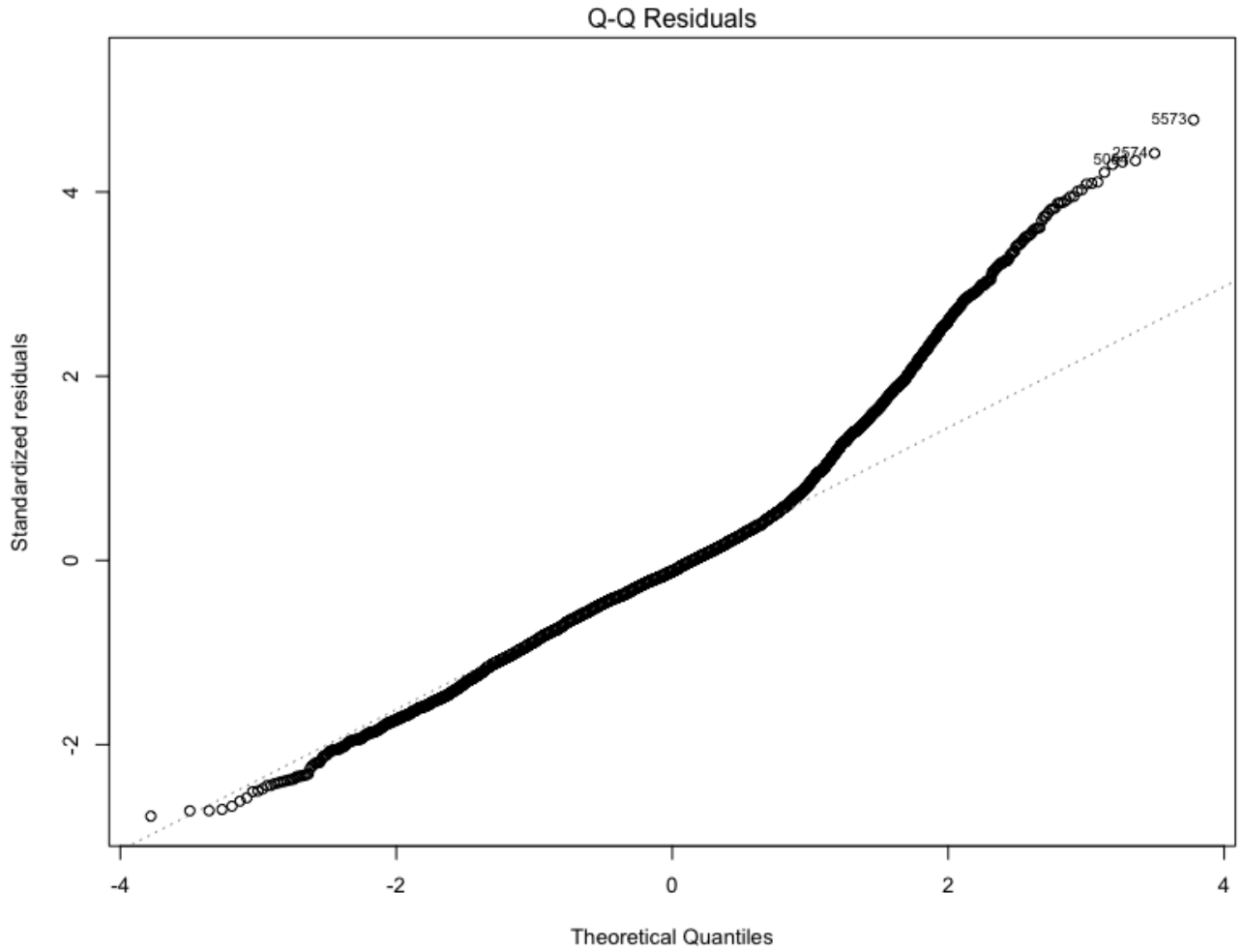


RANKED COEFFICIENTS

```
Mean Absolute Error (MAE): 356.5258
> cat("Mean Squared Error (MSE):", MSE, "\n")
Mean Squared Error (MSE): 225268.6
> cat("Root Mean Squared Error (RMSE):", RMSE, "\n")
Root Mean Squared Error (RMSE): 474.6247
> cat("R-squared:", R_squared, "\n")
R-squared: 0.4302915
> # Residual analysis
> par(mfrow=c(2,2))
> plot(lm_model_weather)  # Diagnostic plots
> # Cross-validation
> library(caret)
> train_control <- trainControl(method = "cv", number = 10) # 10-fold cross-validation
> cv_model <- train(RENTED_BIKE_COUNT ~ ., data = train_data, method = "lm", trControl = train_control)
> print(cv_model)
Linear Regression

6348 samples
  38 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 5713, 5714, 5713, 5714, 5713, 5714, ...
Resampling results:

  RMSE      Rsquared  MAE
  378.8143  0.655412  282.6379

Tuning parameter 'intercept' was held constant at a value of TRUE
```



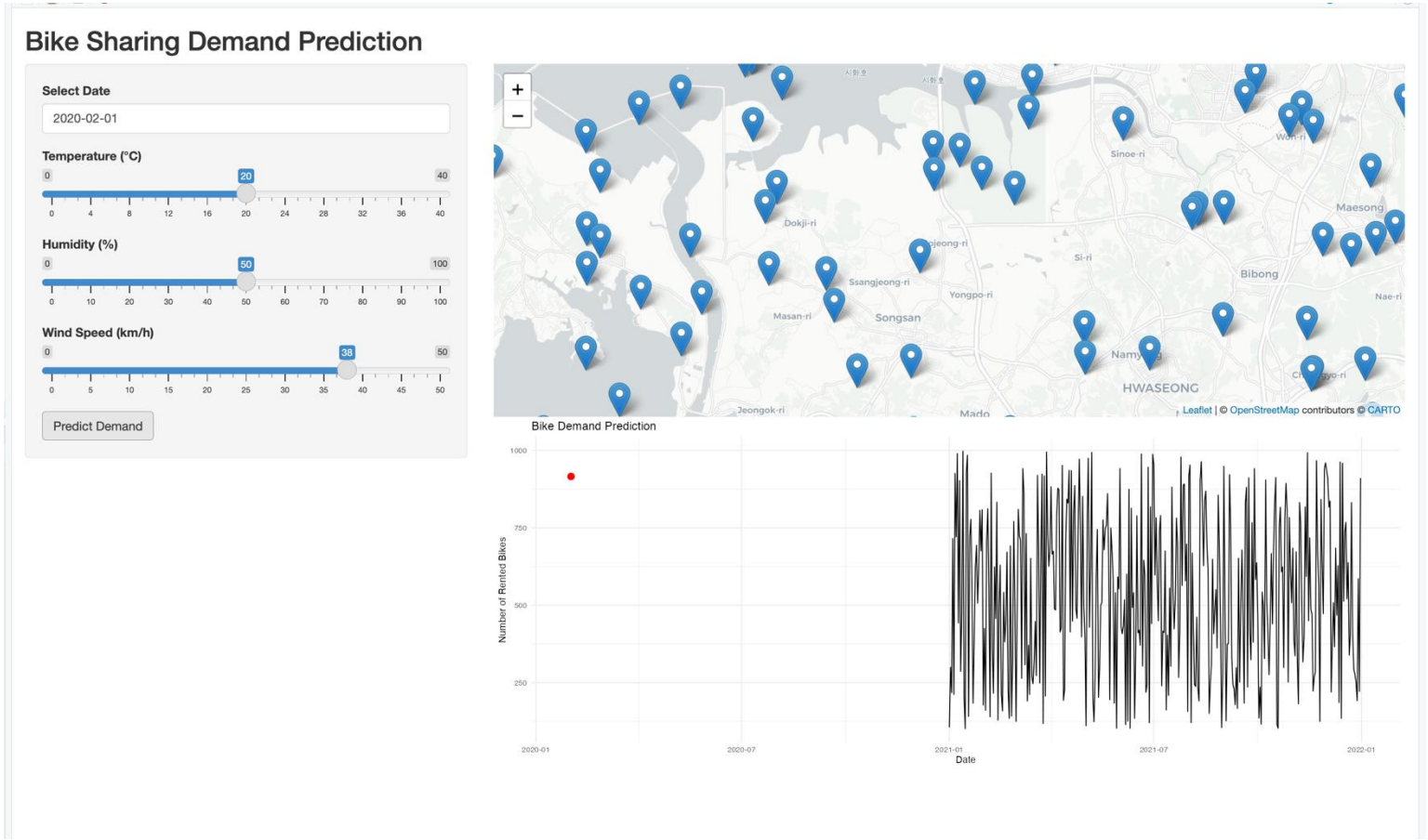
MODEL EVALUATION



Q-Q PLOT OF THE BEST MODEL

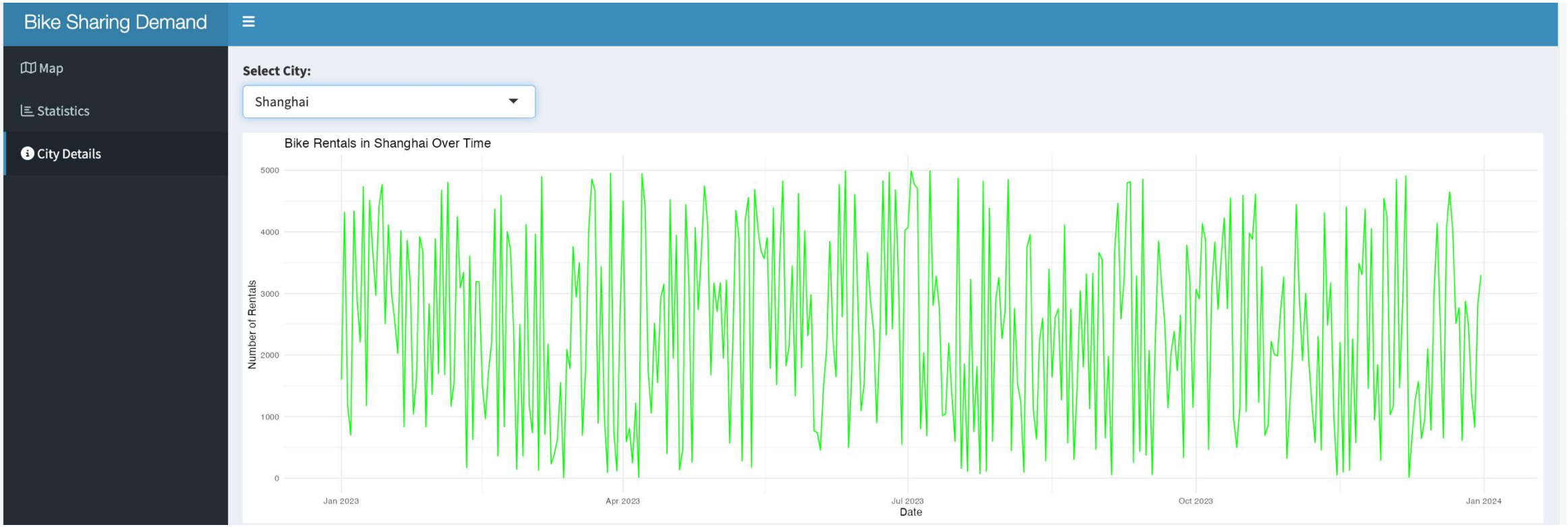
DASHBOARD

BIKE SHARING DEMAND PREDICTION DASHBOARD



SEOUL BIKE PREDICTION

SHANGHAI BIKE PREDICTION



CONCLUSION

- **Key Findings for "Seoul":**
  - **Temperature Influence:** Higher average temperatures in the Summer and Autumn months are associated with increased bike rentals.
  - **Weather Conditions:** Clear skies, lower dew points, and elevated solar radiation in Summer likely contribute to higher bike rental numbers.
  - **Winter Challenge:** Winter sees the lowest bike rental rates, likely due to colder temperatures, stronger winds, and snowfall.
  - **Patterns:** The peak in average bike rentals occurs at 18:00 in Summer, while the lowest average temperature is recorded at 19:00 in Autumn.
  - **Seasonal Trends:** Summer shows a higher average bike rental count than Autumn and Spring.