Final Project: Citi Bike NYC Bikeshare Data Analysis

Will Breckwoldt

Northeastern University

ALY6040: Data Mining

Dr. Marcus Ellis

June 26, 2022

Introduction

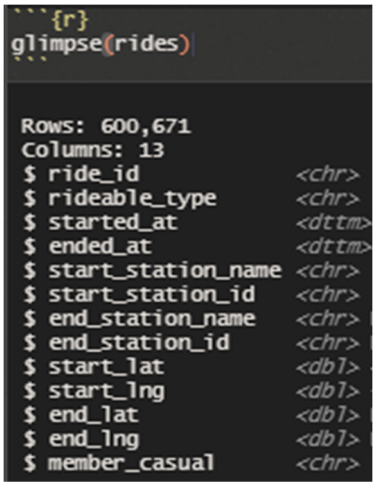
I analyzed Citi Bike NYC bikeshare data, which is publicly available at CityBikeNYC.com. I combined several of the files to create one dataset that spans from June 2021 to March 2022. This report first walks through the exploratory data analysis. Next, the building of a clustering model with the DBSCAN algorithm is outlined. The report concludes with the interpretations of my results and recommendations for Citi Bike NYC.

1. Exploratory Data Analysis

Dimensions

There were 600,671 bikeshare records across 13 variables in the initial data pulled from the Citi Bike site, with each record representing one ride. All the Citi Bike bikeshare data includes unique IDs for each ride, the type of bike, timestamps of the ride start and ride completion, the names and coordinates of starting and ending stations, and whether the rider is a member or casual user. The structure of the dataset is shown with the glimpse() function in Figure 1.1.

Figure 1.1: Glimpse(rides) Output



Summary Statistics

Figure 1.2 shows the five summary values for all numeric variables in our sample. The first recorded ride started on June 1st, 2021, and the last ride ended on March 1st, 2022. While the summary statistics of the coordinates are not interpretable on the surface, I figured that if all stations were used as both starting and ending points, the range of coordinates should be the same for both starting and ending points. But end_lat had a wider range than start_lat, and end_lng had a wider range than start_lng. This is explored further in the EDA Findings section below.

Figure 1.2: summary(rides) Output

```
   ride_id          rideable_type        started_at                      ended_at                    start_station_name
 Length:600671      Length:600671      Min.   :2021-06-01 00:01:55   Min.   :2021-06-01 00:08:10   Length:600671
 Class :character   Class :character   1st Qu.:2021-07-30 22:40:45   1st Qu.:2021-07-30 23:01:47   Class :character
 Mode  :character   Mode  :character   Median :2021-09-17 17:20:47   Median :2021-09-17 17:36:26   Mode  :character
                                       Mean   :2021-09-23 08:09:48   Mean   :2021-09-23 08:31:19
                                       3rd Qu.:2021-11-08 08:14:43   3rd Qu.:2021-11-08 08:29:16
                                       Max.   :2022-02-28 23:50:58   Max.   :2022-03-01 17:12:42

 start_station_id   end_station_name   end_station_id       start_lat       start_lng        end_lat          end_lng
 Length:600671      Length:600671      Length:600671      Min.   :40.71   Min.   :-74.09   Min.   :40.65   Min.   :-74.09
 Class :character   Class :character   Class :character   1st Qu.:40.72   1st Qu.:-74.05   1st Qu.:40.72   1st Qu.:-74.05
 Mode  :character   Mode  :character   Mode  :character   Median :40.73   Median :-74.04   Median :40.73   Median :-74.04
                                                          Mean   :40.73   Mean   :-74.04   Mean   :40.73   Mean   :-74.04
                                                          3rd Qu.:40.74   3rd Qu.:-74.03   3rd Qu.:40.74   3rd Qu.:-74.03
                                                          Max.   :40.75   Max.   :-74.02   Max.   :40.87   Max.   :-73.90
                                                                                           NA's   :2160    NA's   :2160

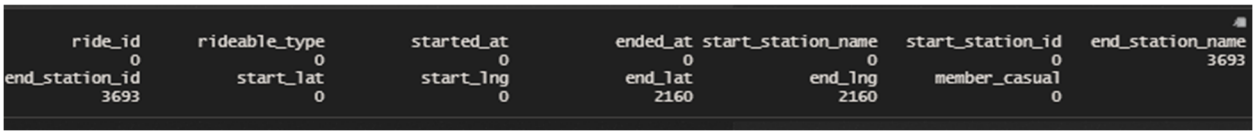
 member_casual
 Length:600671
 Class :character
 Mode  :character
```

Missing Values and Data Removal

There were 2,160 records in the dataset that were missing the name, ID, and coordinates of the end station. I believe these missing values occur when a bike does not reach the end station due to an accident or a breakdown, or if a bike is lost or stolen. There were also 1,533 records that were only missing the name and ID for the end station. However, the end coordinates for these records are imprecise—since they were rounded to two decimal places, they can only identify an area within a radius of roughly a third of a mile (GBFS, 2022). This leads to several

records which we cannot attribute to only one end station. We believe this may be due to a data processing error and should be investigated. For the purposes of our analysis, we removed the 3,693 records that had any missing values. After all, they only made up 0.06% of the observations that we had collected.

Figure 1.3: Initial colSums(is.na(rides)) Output



When running length(unique(rides$ride_id)) our output was 596,978, which is the same number of rows in the rides dataframe after removing rows with missing values. This means that all ride_id values were unique; no rides were recorded multiple times. Furthermore, when looking to see if the same ride was recorded multiple times under different ride_id values, we decided to see if any observations had the same values for both started_at and ended_at values using length(unique(rides$started_at, rides$ended_at)), which confirmed that there were no duplicate ride IDs.

EDA Findings

For the most part, the data was clean at this point. We did find one station ID that was duplicated. One station name had five entries with one of the spaces in the character string replaced with "\\t". We believe that this may have been due to a data processing error and the root cause should be investigated further.

In the summary() call, it was revealed that not all stations are used as both starting points and ending points, by way of difference in the range of values for latitude and longitude. This led

to an analysis of the different formats of station IDs used in the dataset. Additionally, we take a closer look at categorical and temporal variables in the data.

There were two different formats of station IDs. One format was numeric with four digits and two decimal places (e.g., “5297.02”) and one format as alphanumeric with five characters (e.g., “JC107”). The numeric IDs appeared to be for stations that were only used as end stations, except for 1 station out of the 174 stations with these IDs that was also a start station. These can be seen in Figure 1.5 below. Further, these 174 stations only appeared in the dataset 441 times out of the 596,978 observations. This point is addressed in the code walk-through of the Clustering section below.

The distribution of station traffic in this dataset is shown in Figures 1.4 and 1.5. There were a small handful of stations that accounted for a considerable portion of ride traffic. Three stations were the most frequently used as both start and stop stations: “Hoboken Terminal - Hudson St & Hudson Pl”, “Grove St PATH", “South Waterfront Walkway - Sinatra Dr & 1 St". These account for about 12.9% of ride starting points and about 13.1% of ride ending points. Further, it became clear that the end station IDs that refer to stations that were not used as starting points had low traffic, but there were a relatively high number of them compared to the number of stations with the alphanumeric ID type.

Figure 1.4: Distribution of Start Station Frequency

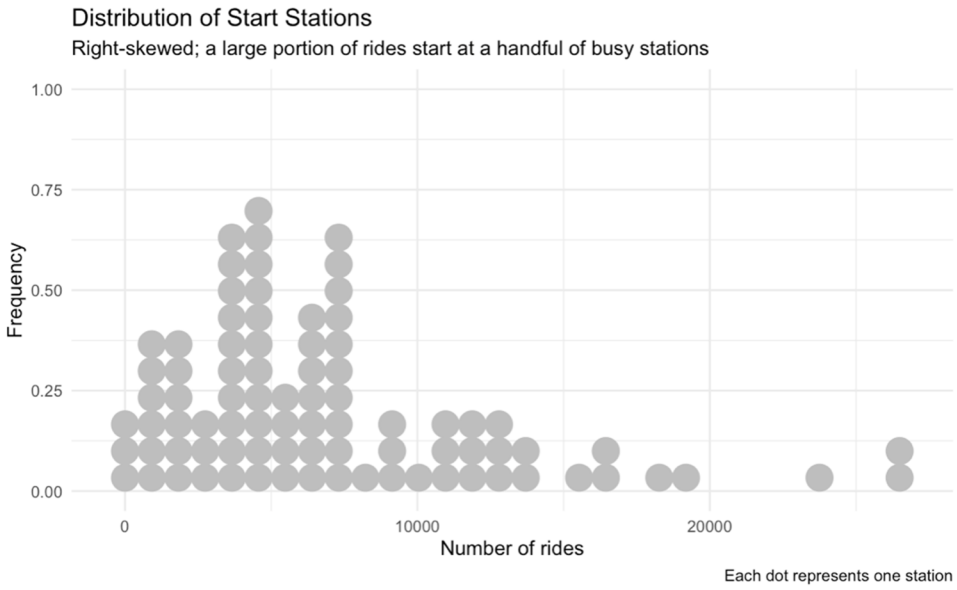
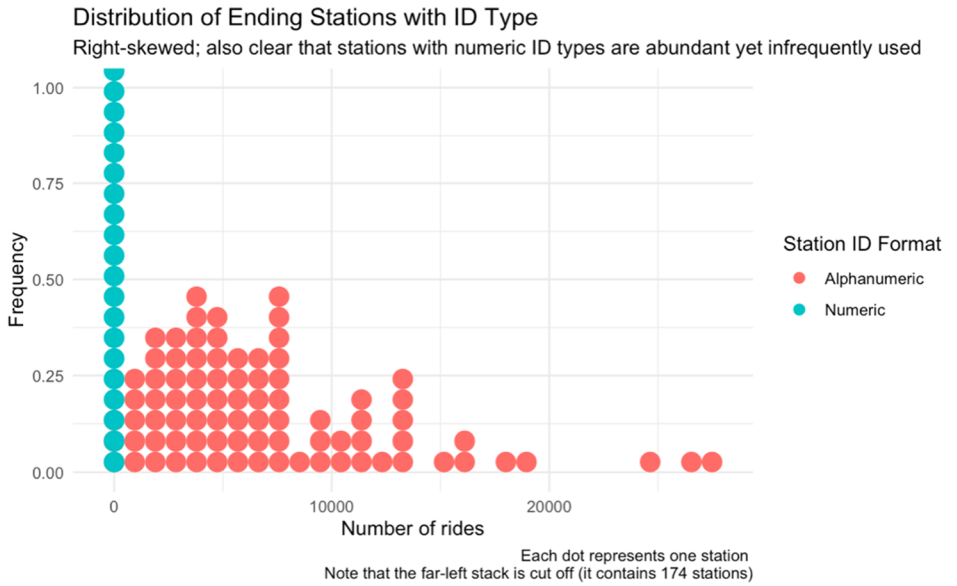


Figure 1.5: Distribution of End Stations (with ID Type)



We inspected the distributions of the categorical variables in the dataset; namely, the membership status of riders and the type of vehicle used. These are shown in Figures 1.6 and 1.7. About 60% of the rides were by members of the Citi Bike program. Members pay $15.42 a month for a subscription to unlimited rides on classic bikes and reduced fares on e-bikes, while non-members pay $3.99 per ride or $15 for a day pass. (CityBikeNYC.com, 2022). Additionally, we found that about two-thirds of rides were on classic bikes, while about 26% were on e-bikes. The remaining 7% were tagged as “docked bikes”. This is unclear—according to the site, all bikes are to be returned to docking stations (CityBikeNYC.com, 2022).

Figure 1.6: Distribution of memberships across rides

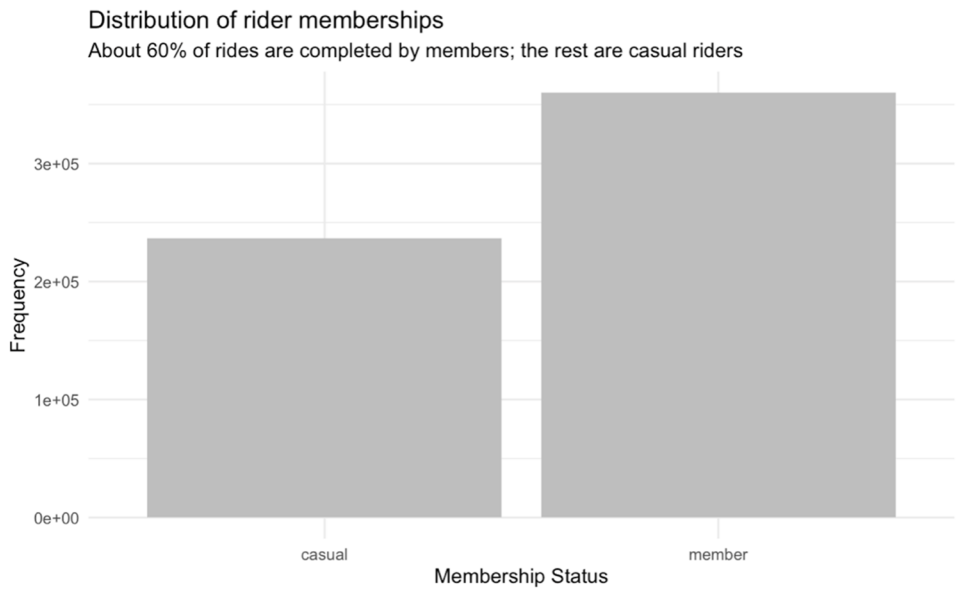
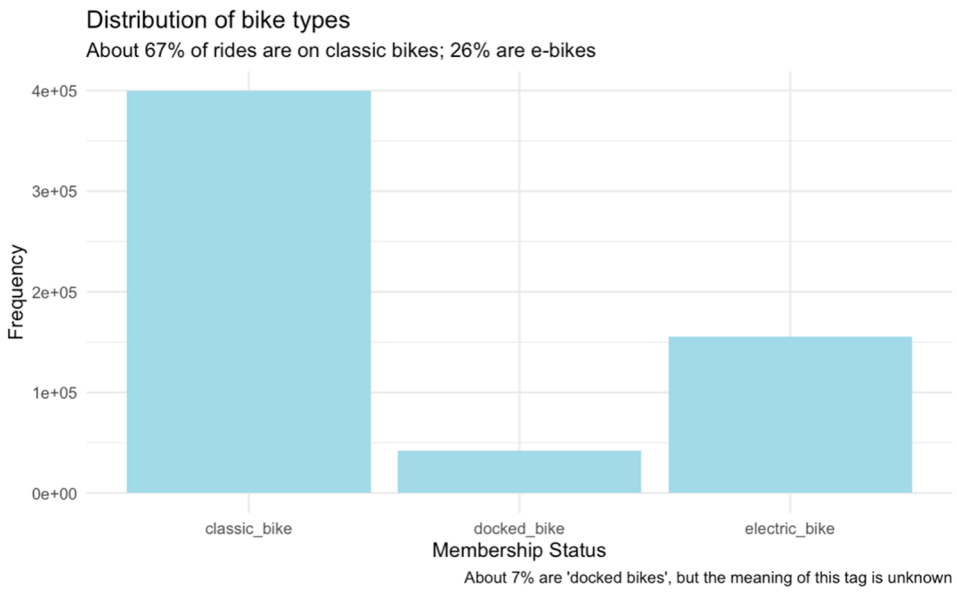
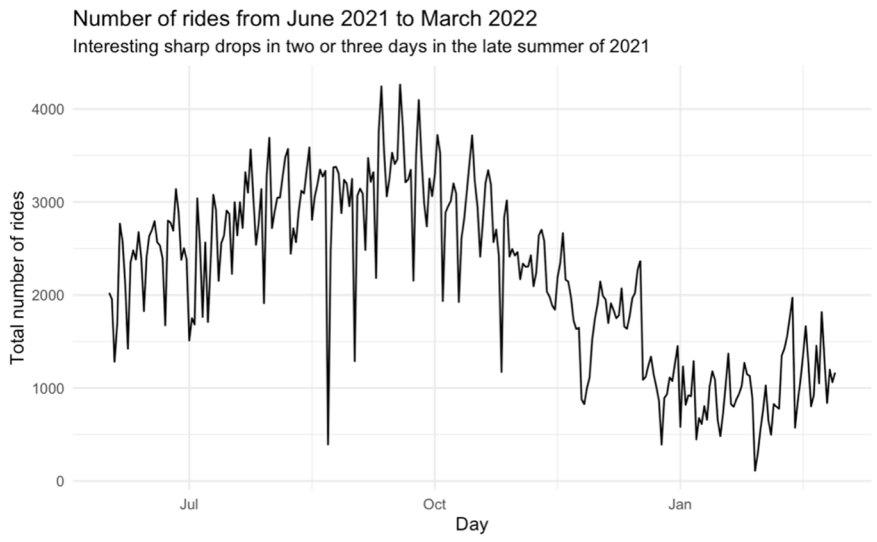


Figure 1.7: Distribution of bike types across rides



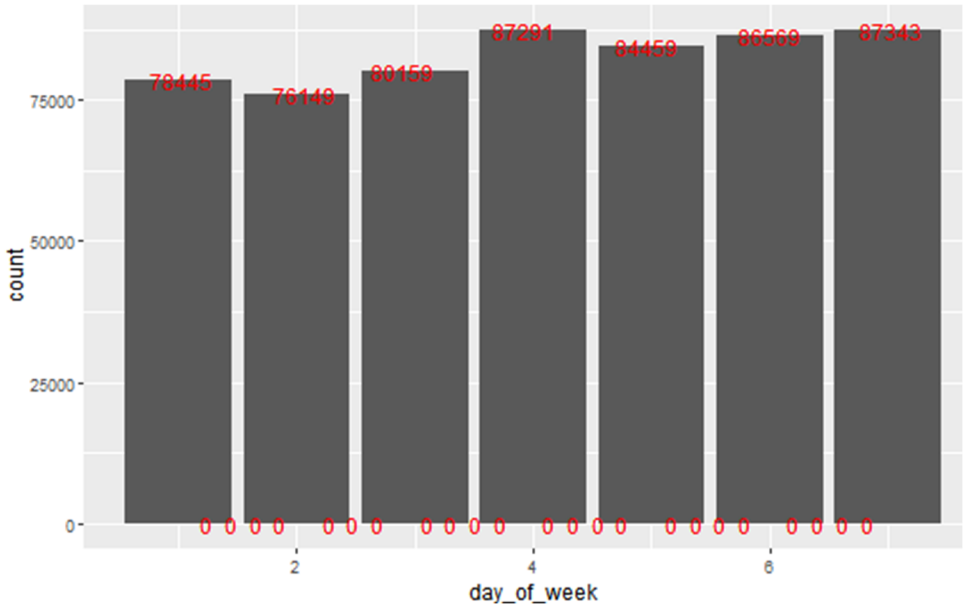
Finally, we inspected the time-series component of the data. As mentioned, the sample of rides collected spans from June 2021 to March 2022. Since each ride in the data has timestamps for its own beginning and end, there was an opportunity to analyze trends. The number of rides throughout the time period contained in the sample are shown in Figure 1.8.

Figure 1.8: Bike rides over time



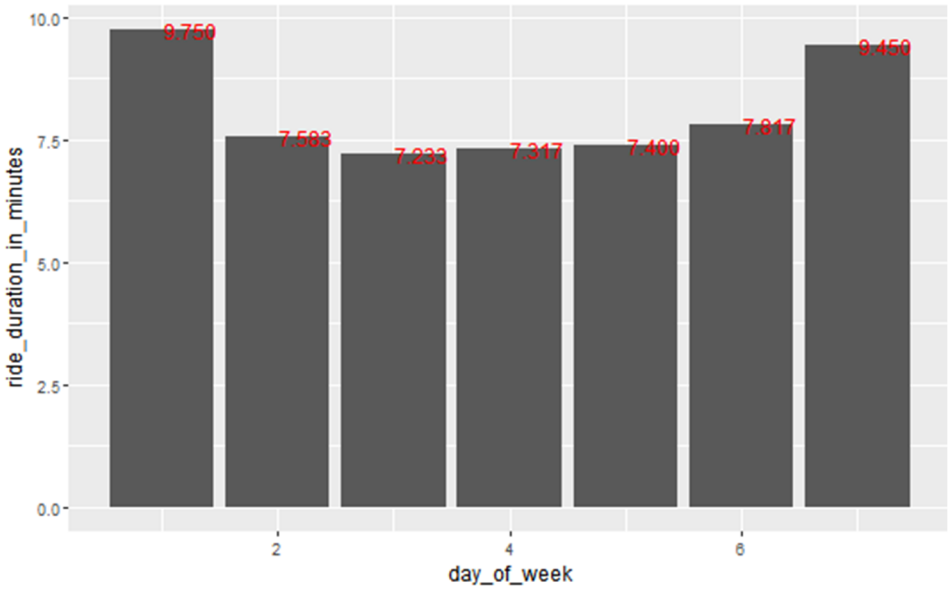
Following this observation, new variables were created for days of the week and months. This may provide insight into the days of the week and time of the year bikes are used the most. In Figure 1.8, a vertical bar plot was created that displays the number of rides by day of the week. In must be noted that 1 = Sunday, 2= Monday, etc..

Figure 1.8 Number of rides by day of the week



Saturday (87,343) is the most popular day of the week followed by Wednesday (87,291) and Friday (86,569). Next, we created a vertical bar plot to demonstrate the median duration by day shown in Figure 1.9. The rides are longest on the weekends. Sunday has the highest median ride duration (9.75 minutes) followed by Saturday (9.45 minutes).

Figure 1.9 Ride duration by day of the week



2. Cluster Analysis

Code Walk-through

The decision was made to convert the time stamps into one variable representing the amount of time taken for each ride, in seconds. This revealed some interesting aspects of the data that were not found in the initial round of EDA.

Figure 2.1: Summary of trip times in seconds

```
trip_time_sec
Min.    :   -3200
1st Qu.:     297
Median :     482
Mean   :     963
3rd Qu.:     840
Max.   :11221289
```

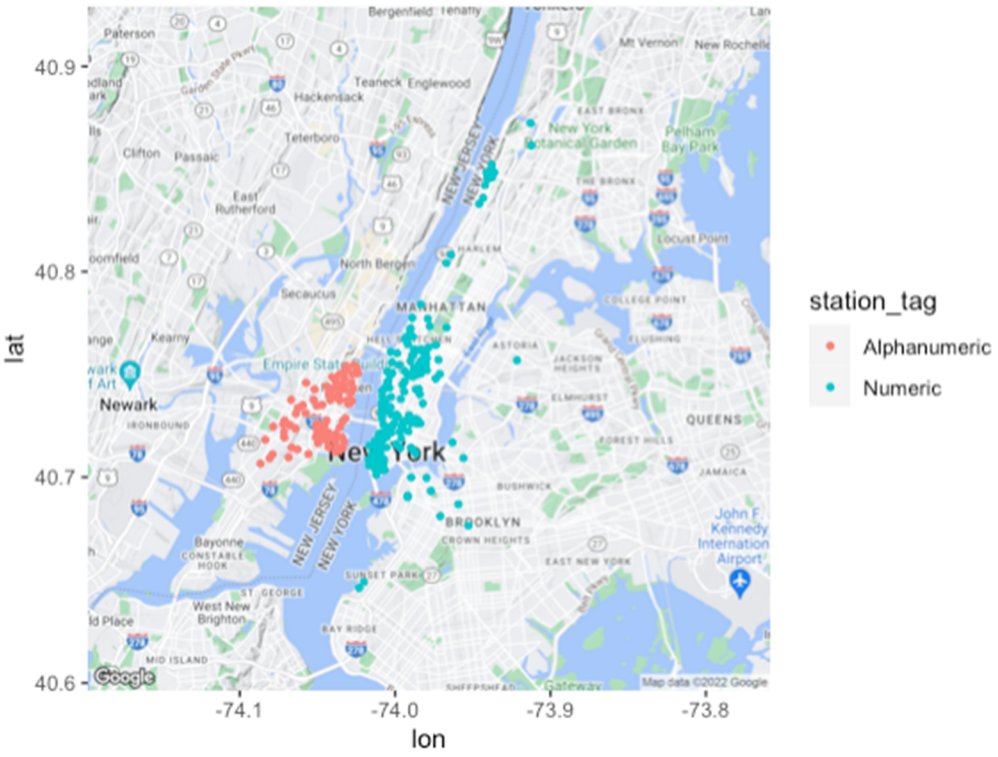
As shown in Figure 2.1, the distribution of trip times clearly contained some extreme outliers on the high end. This may represent bikes that were stolen or not returned for a long time. It is also clear that there must be some erroneous entries, given that the return time should

not have been before the start time. For the sake of this project, the trips were limited to those longer than one minute and less than two hours. This reduced the number of observations by about 3%.

With the potential creation of a clustering model in mind, a map of all the unique starting locations and ending locations were generated. This revealed an important fact about the data that was not found in the initial data exploration. It turns out that all rides in this dataset actually start in Jersey City, NJ, and almost all rides end in Jersey City. Only 441 rides end in NYC. This explains the different station ID types that were pointed out in the initial EDA section.

Figure 2.2: Map of all unique trip starting points (84 locations)

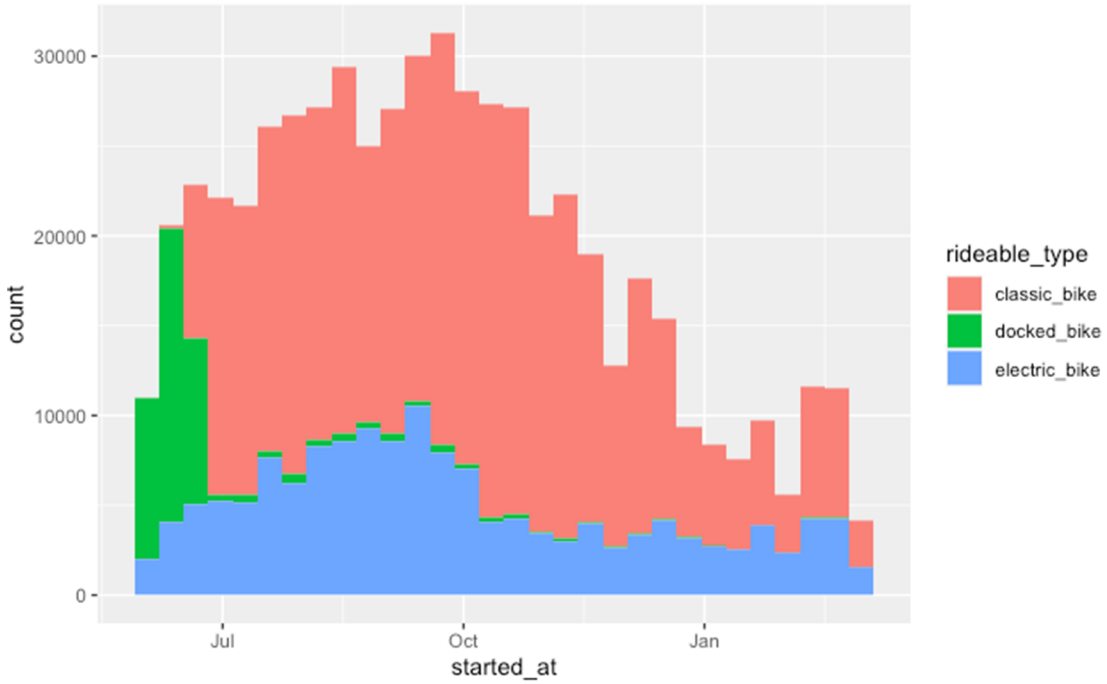
Figure 2.3: Map of all unique trip ending points (258 locations) with station



It was decided that the focus would be on Jersey City, NJ rather than NYC given the large disparity between the two cities. It should be noted that the cause for this is likely due to the non-random process of data selection from Citi Bike's site—we chose several files that were listed in a row and uploaded on the same day.

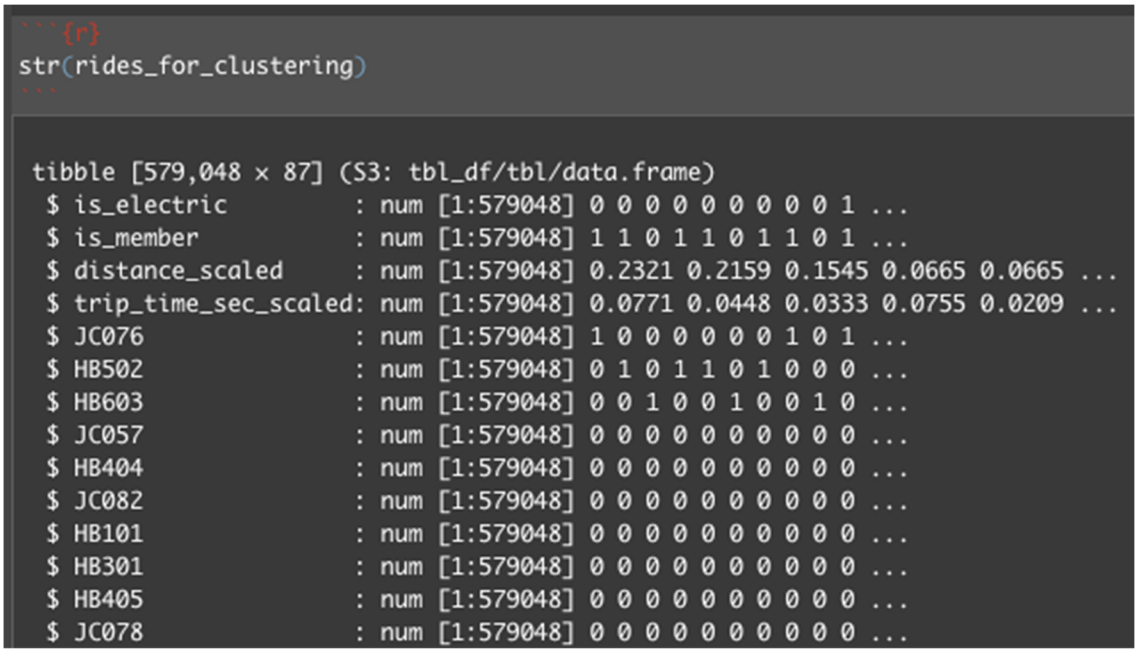
As noted in the initial analysis, there is a bike type called 'docked bike' that only made up about 7% of observations. Though unable to confirm what the meaning of this tag is, it was pointed out by users of a public forum for the data that it could be due to a change in the system's reporting (CitiBikeNYC Hackers, 2021). Figure 2.4 shows justification for combining the docked bike category into the classic bike category.

Figure 2.4: Distribution of bike rides over time with bike type overlay



After these findings, it was decided that the focus of the clustering model would be to cluster rides based on their starting station, distance traveled, amount of time traveled, bike type, and whether or not the user is a member. The dbscan() function in the fpc package was used. In order to reformat the data for this function, the membership and bike type variables were converted to binary variables, the trip time and distance variables were min-max scaled, and one-hot encoding was used for the start station IDs. The structure of the final dataset is shown in Figure 2.5.

Figure 2.5: str() for the dataset after preparing for dbscan()



```
```{r}
str(rides_for_clustering)
```

tibble [579,048 × 87] (S3: tbl_df/tbl/data.frame)
 $ is_electric   : num [1:579048] 0 0 0 0 0 0 0 0 0 1 ...
 $ is_member   : num [1:579048] 1 1 0 1 1 0 1 1 0 1 ...
 $ distance_scaled   : num [1:579048] 0.2321 0.2159 0.1545 0.0665 0.0665 ...
 $ trip_time_sec_scaled: num [1:579048] 0.0771 0.0448 0.0333 0.0755 0.0209 ...
 $ JC076   : num [1:579048] 1 0 0 0 0 0 0 1 0 1 ...
 $ HB502   : num [1:579048] 0 1 0 1 1 0 1 0 0 0 ...
 $ HB603   : num [1:579048] 0 0 1 0 0 1 0 0 1 0 ...
 $ JC057   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ HB404   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ JC082   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ HB101   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ HB301   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ HB405   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
 $ JC078   : num [1:579048] 0 0 0 0 0 0 0 0 0 0 ...
```

Results

A random sample of 10,000 observations from the updated dataset was used for the clustering model. The two DBSCAN parameters are epsilon (a distance specifying the size of the 'neighborhood' around core points) and the minimum number of points required to make a cluster. In this attempt, epsilon is set to 0.15 and MinPts is set to 90, following guidance that MinPts should be greater than or equal to the number of dimensions in the data plus one (Nabriya, 2020).

The resulting clusters are shown in Figure 2.6. Glaringly, nearly 80% of observations were marked as outliers, a clear sign that this clustering approach may not be suitable for the data given. Additionally, there are 15 different clusters created by the model, which drastically reduces the interpretability of the results. A scatterplot of the sample of observations showing distance versus time with an overlay of the cluster tag is shown in Figure 2.7. This figure does not

contain the roughly 80% of observations that are marked as outliers. It tells us that the clusters are likely heavily reliant on other variables.

Figure 2.6: Results of dbscan() with epsilon = 0.15 and MinPts = 90.

```
dbscan Pts=10000 MinPts=90 eps=0.15
          0   1   2  3   4   5   6   7   8   9  10  11  12  13  14 15
border 7958  18  60 88  17  27  23  12  29  33  13  13  16  47  53 90
seed      0 150  45  2 284  95 120 120 115 107 137  92 133  55  47  1
total  7958 168 105 90 301 122 143 132 144 140 150 105 149 102 100 91
```

Figure 2.7: Distance vs Time in sample, cluster outliers excluded

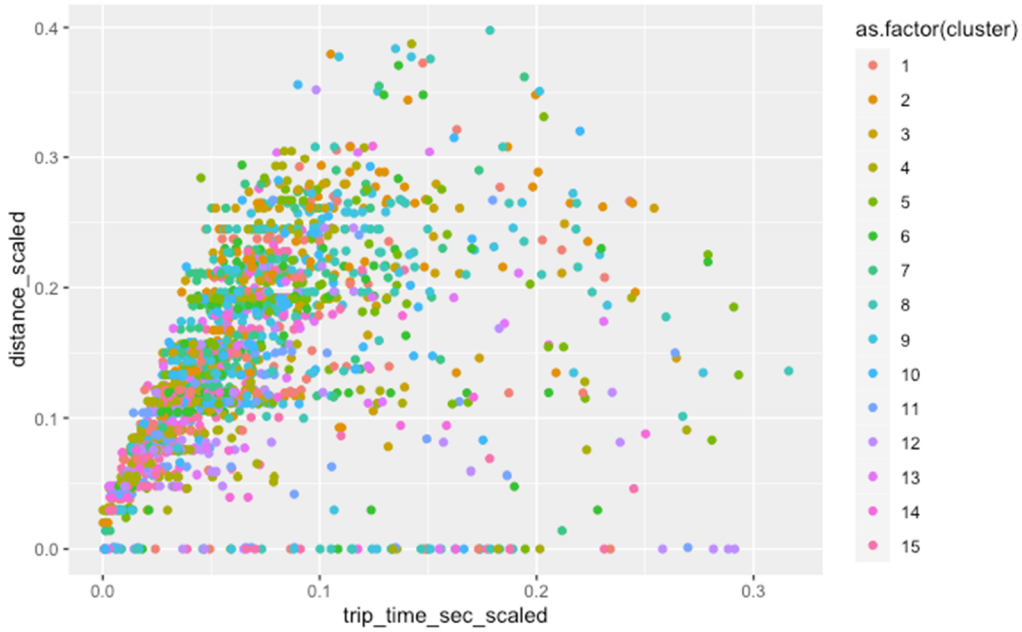


Figure 2.8 shows a faceted plot including outliers, and Figure 2.9 shows a faceted plot without outliers. In both cases, the faceted rows represent membership (0 = casual user, 1 = member) and the faceted columns represent whether the bike is electric (0 = classic, 1 = electric). Interestingly, the non-outlier points only contain classic bikes, as is emphasized in Figure 2.9.

There also appears to be some clusters that rely heavily on whether the user is a member or not, but it is hard to tell for sure and should be investigated more closely.

Figure 2.8: All points, faceted by is_electric (columns) and is_member (rows)

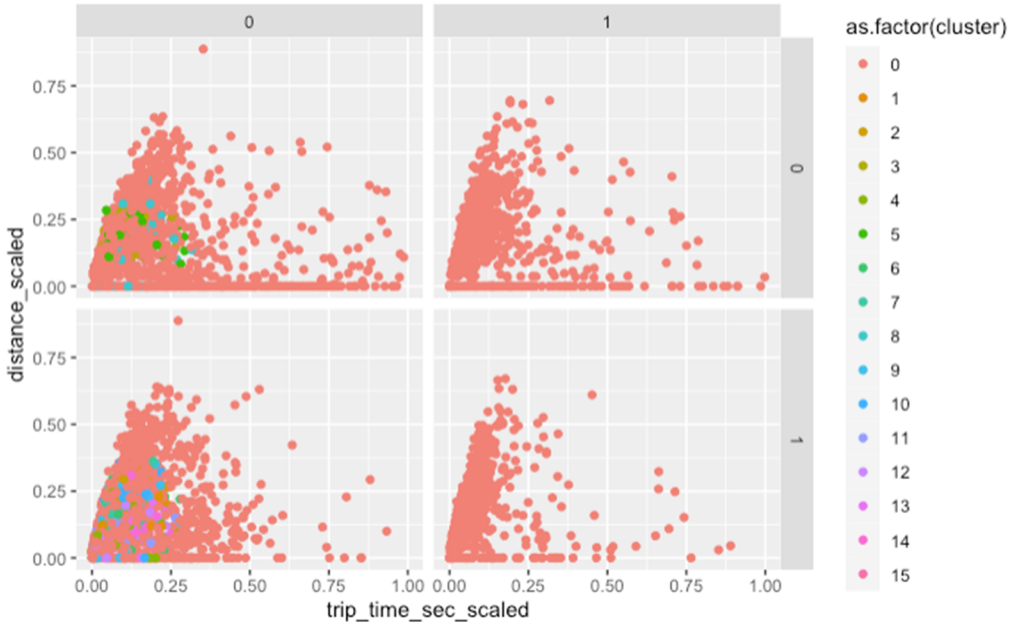
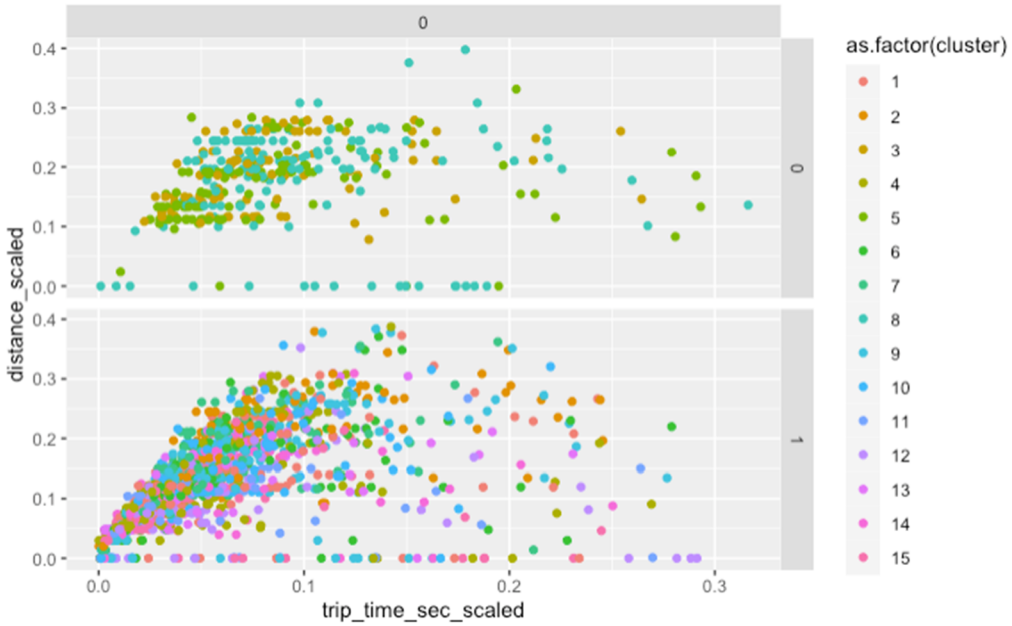


Figure 2.9: Excluded outliers, faceted by is_electric (columns) and is_member (rows)



Interpretation and recommendations

In preparation for the clustering model, key findings helped in the building of the model and will be used to guide further analyses of this dataset. One of the key takeaways came from mapping the unique starting and ending locations—finding that almost the entire dataset is comprised of Jersey City bike rides rather than New York City. Additionally, a closer inspection of the timestamp variables allowed for the removal of outliers and erroneous data.

The clustering model itself was ultimately not successful. However, further steps could be taken in attempt to improve results before ruling out clustering completely. First, the epsilon and minimum points parameters should be optimized. Additionally, principal component analysis or linear discriminant analysis could be used to reduce the number of dimensions in the dataset. Even taking a closer look at the stations and setting a minimum number of rides for a station to be included could help with dimensionality reduction.

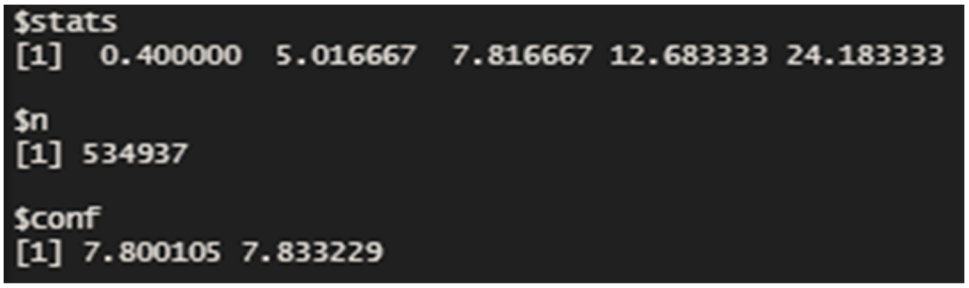
There are limitations that were uncovered that should be addressed. First, in considering clustering based on distances, it was realized that the bikes themselves do not contain GPS systems, so actual paths of rides cannot be traced. One can only find the amount of time elapsed during a ride and the minimum possible distance between the start and end distance—in the case of this analysis, a straight line using coordinates and Haversine distance (Kranish, 2021). To illustrate the negative effect of this, rides that start and end at the same stations could be misleading because they will have a distance of zero. Second, using distances rather than coordinates leaves out the opportunity to analyze patterns based on physical locations. Finally, the clustering model itself was difficult to interpret. The high number of clusters paired with the high number of dimensions means that it would take a lot of manual exploratory analysis to find meaning in each of the clusters.

3. Probability Table

Code Walk-through

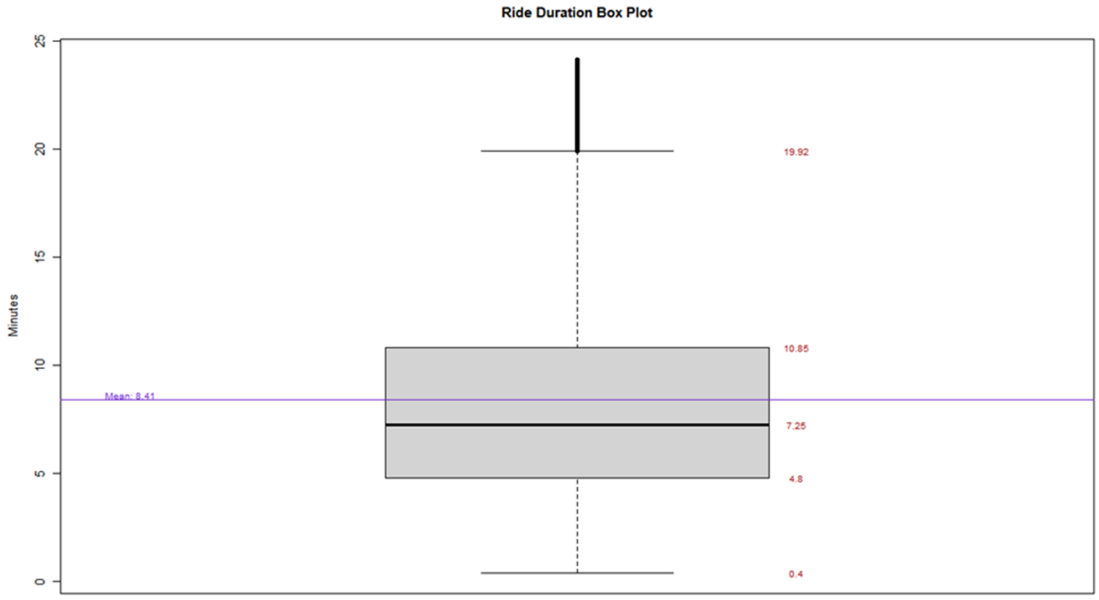
With the new variable we created (ride_duration_in_minutes), we will evaluate for the distribution of ride duration. We remove rides with no end station and remove rides where the start station is the same as the end station. We also remove rides with a duration less than 0 minutes.

Figure 3.1: Ride duration in minutes box plot stats



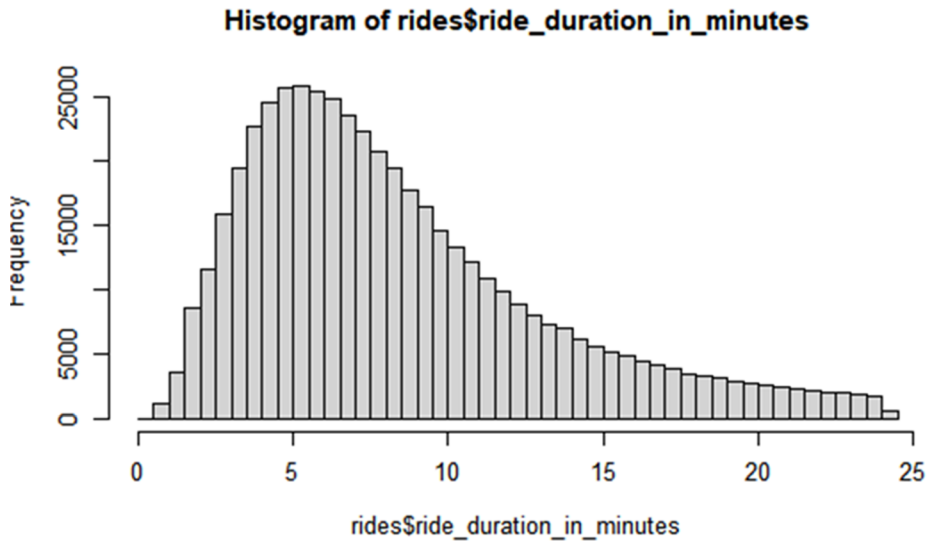
The output gives us our minimum, quartiles, and maximum, along with the number of entries and a 95% confidence interval for our medians. For the purpose of the probability table, we remove the outliers above the maximum (rides more than 24.183 minutes). Figure 3.2 shows the boxplot of our subset.

Figure 3.2: Ride duration box plot



Looking at the boxplot we have a mean larger than the median which suggests that the histogram will be positively skewed. It must be noted values are round to two decimal places. We will create a histogram in Figure 3.3 to demonstrate further demonstrate the ride duration distribution.

Figure 3.3: Ride duration histogram



Our prior assumption that since the mean is greater than the medium, the histogram will be skewed to the right or positively skewed is true. It seems like most rides are between 2 and a half and 12 minutes in duration. Below is our probability table for ride duration.

Figure 3.4: Ride duration probability table

	duration_intervals	Freq	cumFreq	relative
1	(0,2.5]	24833	24833	0.05098540
2	(2.5,5]	108107	132940	0.22195782
3	(5,7.5]	121841	254781	0.25015552
4	(7.5,10]	88775	343556	0.18226670
5	(10,12.5]	54673	398229	0.11225083
6	(12.5,15]	33753	431982	0.06929933
7	(15,17.5]	22229	454211	0.04563905
8	(17.5,20]	15396	469607	0.03161000
9	(20,22.5]	11421	481028	0.02344881
10	(22.5,25]	6033	487061	0.01238654

Results

We have created intervals starting at 0 and ending at 25 that have a length of 2.5 to further determine the frequency of ride duration. The output gives us the number of rides per interval, which is shown in the Freq column. The cumFreq column shows the cumulative

frequency of the interval (the number of rides in the interval plus the lesser intervals). The relative column gives us the proportion of the interval relative to the subset.

The most frequent interval is between 5 minutes and 7 and a half minutes and makes up for about 25% of the rides. Furthermore, about 76.67% of rides are between 2 and a half and 12 and a half minutes.

Interpretations and Recommendations

3. Regression Diagnostics

Code Walk-through

The purpose here was to see if we could create a regression model using ride_duration as a target variable. We created dummy variables for month, day_of_week, member_casual, rideable_type, and end_station_id. Our subset included only the most frequently used start station Grove St Path, which was the start station for 25,698 rides. We calculated the distance between stations and recorded it as dis. Our subset for our model included only numeric variables. Coordinate variables, day_of_week, month, Mar, Apr, and May were removed.

4.1 Scatterplot dis vs. ride_duration_in_minutes

We first fit our model using all variables from the subset as predictor variables. The summary told us that we had many high Pr(>|t|) values, which indicated we may need to remove some variables. Furthermore, our residual standard error was 9.272 and our adjusted R-squared value was 0.2529. This indicated that the model had high variance and low

precision. Furthermore, performed AIC and BIC, which had significantly high results indicating low accuracy.

We ran a stepwise algorithm to remove our unwanted predictor variables and choose a model by lowest possible AIC. There were 28 steps, and 28 variables were removed.

Figure 4.1: Stepwise, final step

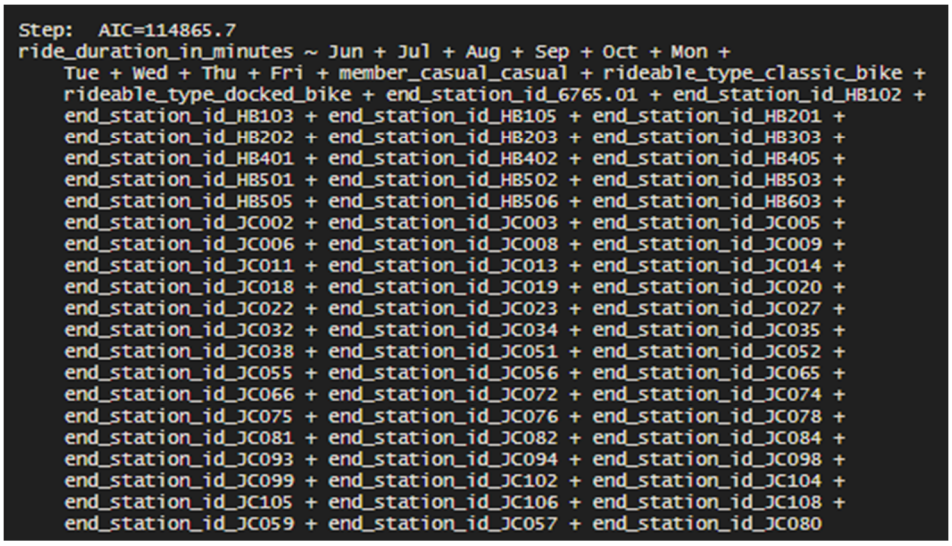
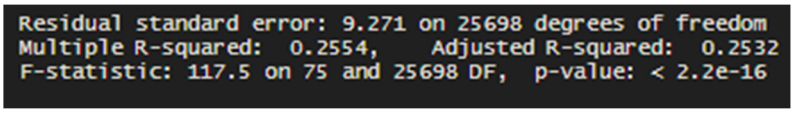
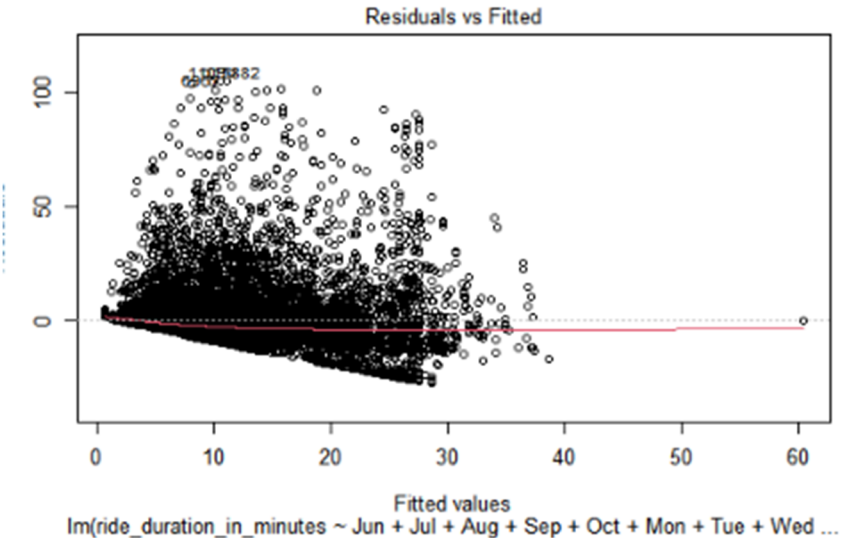


Figure 4.2: Summary of the fitted regression model



The final step gives us 75 predictor variables. Our results indicate that our model again has high variance and low precision. The Residual standard error was 9.271 and our adjusted R-squared value was 0.2532. Below we plotted the fitted regression model.

Figure 4.3: Residuals vs. Fitted plot



There is random dispersion in the Residual vs Fitted plot as we hoped. The assumption of linearity appears to be true. The lowess line remains close to the regression line. It appears that as residuals increase, our model becomes less accurate.

Figure 4.4: Q-Q plot

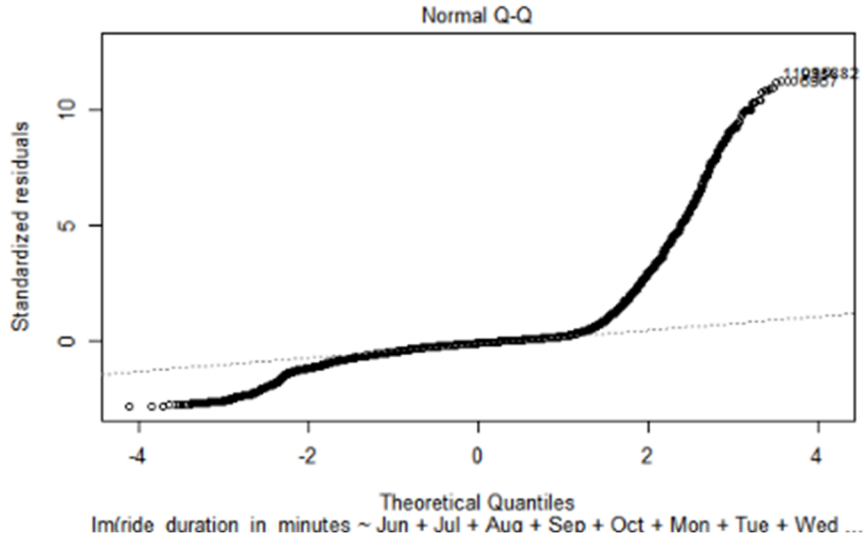
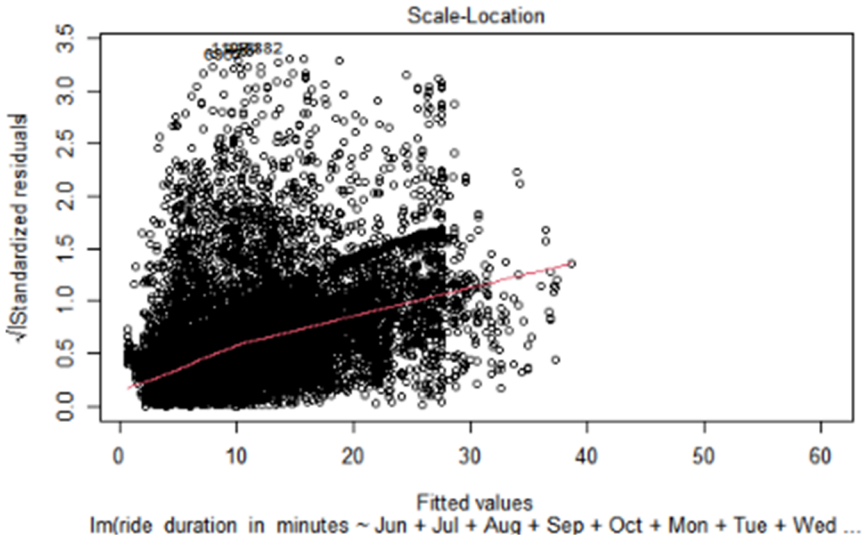
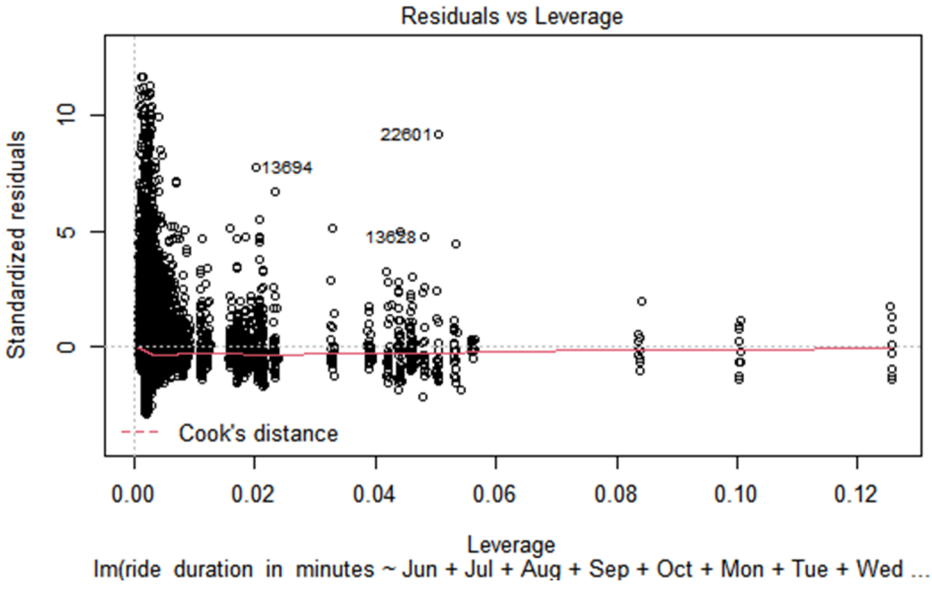


Figure 4.5 Scale-Location plot



The plot shows constant variance or homoscedasticity. There seems to be a random, not distinct pattern, which is what we were hoping for.

Figure 4.6 Residuals vs Leverage plot



The plot helps us identify unusual observations. As we can see there are a few unusual observations labeled: 13,694, 22,601, and 13,628.

Results

Our scatter plot demonstrated that there must be a correlation with ride duration and distance. Unfortunately, our regression model was not very precise and demonstrated high variance as shown in our summary. However, our plots were what we were hoping for, minus some unusual observations in our residuals vs. leverage plot. It seems that by adding new predictor variables to the dataset and removing outliers we may create a more precise model.

Interpretations and Recommendations

It appears that estimating ride duration based on our current variables may be difficult. With our current variables, he has a high residual standard error and a high adjusted r squared error. However, our plots look hopeful and we demonstrated some correlation between ride duration and our predictor variables.

The more variables in the model, the more accurate the model. Citi Bike may consider recording and publishing more variables to provide further insights into ride duration. Bikers may stop at other locations before reaching the end station. Other impacts on ride duration may be the time of day, weather, rider information (weight, age, gender, etc.), traffic, holidays, events, and bike status. Using all these variables, Citi Bike may be able to provide their users with trip estimates specific to the user, the bike they are using, and the day. These ride details may be shared on an app or website and would surely improve customer satisfaction and retention.

References

CitiBikeNYC Hackers. (2021, June 15). *Feb+March 2021 data quality issues.* ctbk.dev. [Google Groups]. Retrieved June 15, 2022 from https://groups.google.com/g/citibike-hackers/c/dQl18PMPu9I/m/H5gEvxTkGgAJ

Citi Bike: NYC's Official Bike Sharing System: Citi Bike NYC. Citi Bike: NYC's Official Bike Sharing System. (n.d.). Retrieved June 12, 2022, from https://citibikenyc.com/

General Bikeshare Feed Specification (GBFS) - NABSA. GitHub. (2022, April 5). Retrieved June 4, 2022, from https://github.com/NABSA/gbfs/blob/master/gbfs.md#coordinate-precision

Nabriya, P. (2020, July 23). *A routine to choose eps and minPts for DBSCAN*. StatsExchange. Retrieved June 15, 2022 from https://stats.stackexchange.com/questions/88872/a-routine-to-choose-eps-and-minpts-for-dbscan