R Notebook

Code ▾

Hide

```
library(corrplot)
```

```
corrplot 0.94 loaded
```

Hide

```
library(tidyverse)
```

```
── Attaching core tidyverse packages
─────────────────────────────────────────────────────────────────────────────────────
────────────────────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.3     ✔ tidyr     1.3.1
✔ purrr     1.0.2     
── Conflicts
─────────────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors
```

Hide

```
library(ggplot2)
library(maps)
```

```


Attaching package: ‘maps’


The following object is masked from ‘package:purrr’:


    map
```

Hide

```
library(ggmap)
```

```
ℹ Google's Terms of Service: ]8;;https://mapsplatform.google.com<https://mapsplatform.google.com>]8;;
  Stadia Maps' Terms of Service: ]8;;https://stadiamaps.com/terms-of-service/<https://stadiamaps.com/terms-of-service/>]8;;
  OpenStreetMap's Tile Usage Policy: ]8;;https://operations.osmfoundation.org/policies/tiles/<https://operations.osmfoundation.org/policies/tiles/>]8;;
ℹ Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

Hide

```
library(ggplot2)
library(tmap)
```

```
Registered S3 method overwritten by 'htmlwidgets':
  method           from         
  print.htmlwidget tools:rstudio
Breaking News: tmap 3.x is retiring. Please test v4, e.g. with
remotes::install_github('r-tmap/tmap')
```

Hide

```
library(geosphere)
library(sf)
```

```
Linking to GEOS 3.12.1, GDAL 3.8.4, PROJ 9.3.1; sf_use_s2() is TRUE
```

Hide

```
trips_df <- read_csv("../data/trips_data.csv")
```

```
Rows: 6453999 Columns: 13
── Column specification ──────────────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────
Delimiter: ","
chr  (7): ride_id, rideable_type, start_station_name, start_station_id, end_station_name, end_station_id, m
ember_casual
dbl  (4): start_lat, start_lng, end_lat, end_lng
dttm (2): started_at, ended_at

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

#dealing with the docked_bike issue

Docked_bike types are supposed to be classic_bike types

Hide

```
trips_df <- trips_df %>%
  mutate(rideable_type = ifelse(rideable_type == "docked_bike", 
                                "classic_bike", rideable_type))
```

Hide

```
unique(trips_df$rideable_type)
```

```
[1] "electric_bike" "classic_bike" 
```

#Fixing end_lng and end_lat problems

there are 2 problems found. One being 0 values on some of the rows and another being the missing values on some of the rows. Lastly there seem to be Outliers on the histograms(This could be the 0’s but I don’t know yet.)

##dealing with the 0’s

Hide

```
trips_df %>%
  select(everything()) %>%
  filter(end_lng == 0 | end_lat == 0)
```

ride_id <chr>	rideable_type <chr>	started_at <S3: POSIXct>	ended_at <S3: POSIXct>	start_station_name <chr>
873D50153BBC0686	electric_bike	2023-06-15 12:38:05	2023-06-15 12:38:41	OH Charging Stx - Test
ADFF57D27B5BF9D2	classic_bike	2023-06-15 09:38:07	2023-06-15 09:42:57	State St & 54th St
43107577DF9B498D	classic_bike	2023-08-21 18:43:22	2023-08-21 22:05:55	Dearborn St & Erie St

3 rows | 1-5 of 13 columns

There are only 3 values that have 0 and in both cases this occurs on end_lng and end_lat at the same time. 2 of them are tests and one isn't. I will convert the 0's into an NA value and deal with it together with the other missing values.

Hide

```
trips_df <- trips_df %>%
  mutate(end_lng = na_if(end_lng, 0),
         end_lat = na_if(end_lat, 0))
  trips_df %>%
    select(everything()) %>%
    filter(end_lat == 0 | end_lng == 0 )
```

0 rows | 1-6 of 13 columns

##Cleaning Test data

I want to make sure that there are no more tests outside of the 2 I just found.

Hide

```
trips_df %>%
  select(everything()) %>%
  filter(grepl("(Test)$", end_station_name, ignore.case=TRUE) | grepl("(Test)$", start_station_name, ignor
e.case=TRUE) |  grepl("(Test)$", start_station_id, ignore.case=TRUE) |  grepl("(Test)$", start_station_id,
ignore.case=TRUE))
```

ride_id <chr>	rideable_type <chr>	started_at <S3: POSIXct>	ended_at <S3: POSIXct>	start_station_name <chr>
1EC494994DFD4553	electric_bike	2023-06-28 15:32:50	2023-06-28 15:33:07	OH Charging Stx - Test
A34B8C56E7692CB5	electric_bike	2023-06-29 14:29:06	2023-06-29 14:29:13	OH Charging Stx - Test
465EA70E1D719562	electric_bike	2023-06-29 14:41:13	2023-06-29 14:41:19	OH Charging Stx - Test
89D9BB1625C66CE3	classic_bike	2023-06-29 14:36:06	2023-06-29 14:36:13	OH Charging Stx - Test
E3AC9546FB4F0BEB	classic_bike	2023-06-28 15:44:00	2023-06-28 15:44:06	OH Charging Stx - Test
3AA20CC3FE43F678	classic_bike	2023-06-28 10:56:35	2023-06-28 10:56:40	OH Charging Stx - Test
103567010777D572	classic_bike	2023-06-28 15:43:40	2023-06-28 15:43:44	OH Charging Stx - Test

ride_id <chr>	rideable_type <chr>	started_at <S3: POSIXct>	ended_at <S3: POSIXct>	start_station_name <chr>
12A36ED2AAE587FD	electric_bike	2023-06-28 15:32:11	2023-06-28 15:32:27	OH Charging Stx - Test
0D77713ADEE7A4ED	electric_bike	2023-06-28 15:34:27	2023-06-28 15:34:33	OH Charging Stx - Test
B6A90F07CCAEEB50	electric_bike	2023-06-28 15:38:05	2023-06-28 15:38:13	OH Charging Stx - Test

1-10 of 19 rows | 1-5 of 13 columns Previous **1** 2 Next

There are more. This data doesn’t serve us since they are tests.

Hide

```
trips_df <- trips_df %>%
  select(everything()) %>%
  filter(!(grepl("(Test)$", end_station_name, ignore.case=TRUE) | grepl("(Test)$", start_station_name, ignore.case=TRUE) |  grepl("(Test)$", start_station_id, ignore.case=TRUE) |  grepl("(Test)$", start_station_id, ignore.case=TRUE)))
```

Hide

```
trips_df %>%
  select(everything()) %>%
  filter(grepl("(Test)$", end_station_name, ignore.case=TRUE) | grepl("(Test)$", start_station_name, ignore.case=TRUE) |  grepl("(Test)$", start_station_id, ignore.case=TRUE) |  grepl("(Test)$", start_station_id, ignore.case=TRUE)) %>%
  count()
```

n <int>
0

1 row

The data is now cleaned of tests.

#Fixing the outliers

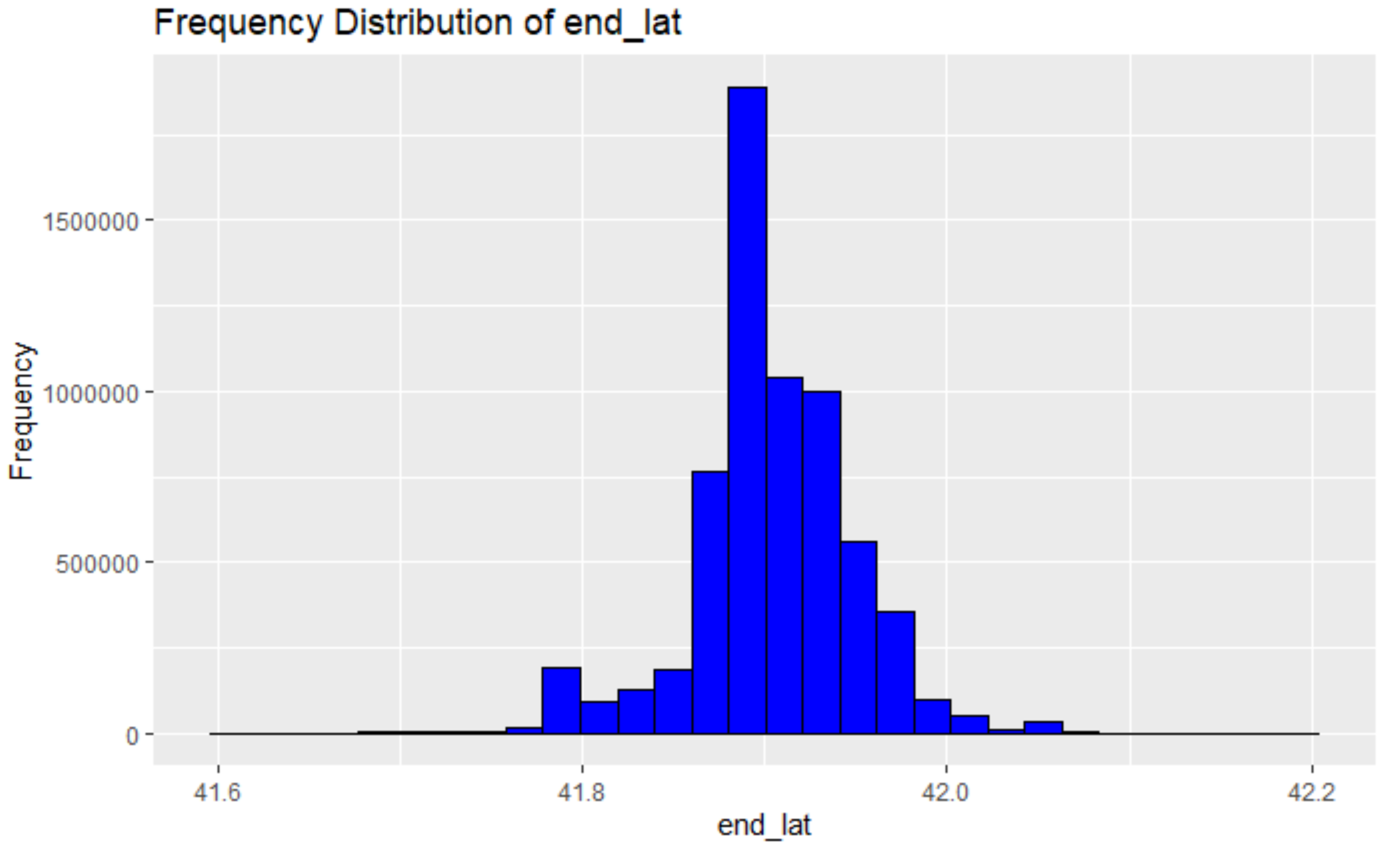
I want to now see if removing the 0’s from the data removed the outliers.

Hide

```
ggplot(trips_df, aes(x = end_lat)) +
    geom_histogram(fill = "blue", color = "black") +
    labs(title = "Frequency Distribution of end_lat", x = "end_lat", y = "Frequency")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
Warning: Removed 8809 rows containing non-finite outside the scale range (`stat_bin()`).
```

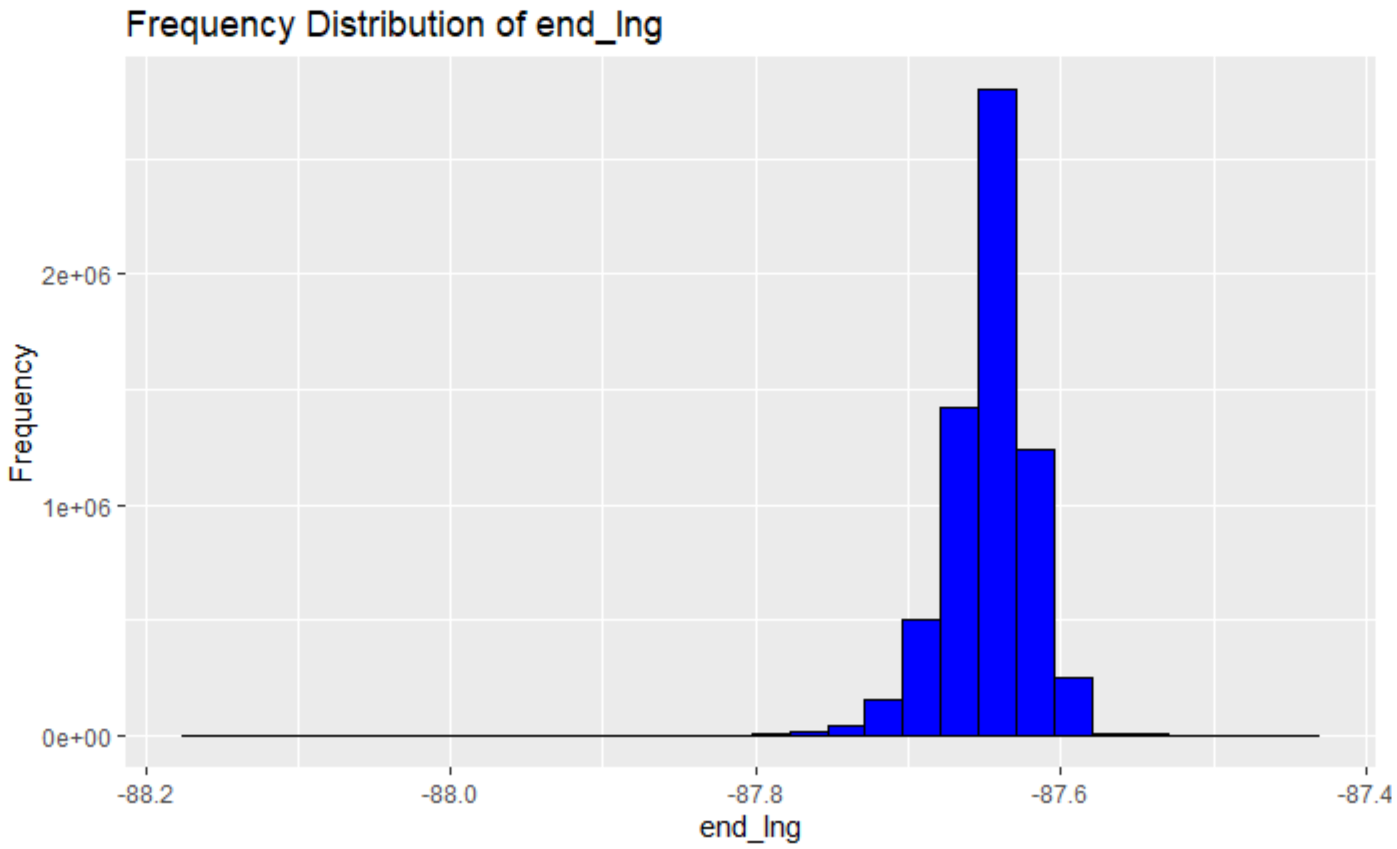


Hide

```
ggplot(trips_df, aes(x = end_lng)) +
    geom_histogram(fill = "blue", color = "black") +
    labs(title = "Frequency Distribution of end_lng", x = "end_lng", y = "Frequency")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
Warning: Removed 8809 rows containing non-finite outside the scale range (`stat_bin()`).
```



The only outliers where the 0's so I can move on to the next step now.

#Dealing with the missing end_lng and end_lat

I know that they are both missing at the same time. Lets see if we can find a pattern. to it

Hide

```
missing_lng_lat <- trips_df %>%
  select(everything()) %>%
  filter(!complete.cases(end_lng))

missing_lng_lat %>% count()
```

n
<int>
8809

1 row

Hide

```
missing_lng_lat %>%
  select(everything())
```

ride_id	rideable_type	started_at	ended_at ▸
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>
685DB4D7A6AF6CAE	classic_bike	2023-06-29 17:35:41	2023-06-29 17:45:38
E4AB9F672ECD0966	classic_bike	2023-06-29 14:50:04	2023-06-29 15:06:49

ride_id	rideable_type	started_at	ended_at ▸
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>
B0A0B0C83B363BC3	classic_bike	2023-06-23 16:06:21	2023-06-23 16:16:55
D49A2A420AE05181	classic_bike	2023-06-30 17:08:02	2023-07-01 18:07:52
2C35DCA44370EAD8	classic_bike	2023-06-30 18:45:11	2023-06-30 19:06:34
D19CA7CA83B28F88	classic_bike	2023-06-29 17:36:45	2023-06-29 18:02:39
1005EAF8E1C29D5C	classic_bike	2023-06-24 13:56:09	2023-06-24 14:18:23
B34C24AD17CCB667	classic_bike	2023-06-22 22:32:59	2023-06-22 23:09:50
64D120B77FA6F330	classic_bike	2023-06-22 22:32:41	2023-06-22 23:09:48
92008CFA88E93F44	classic_bike	2023-06-22 22:33:03	2023-06-22 23:09:44

1-10 of 8,809 rows | 1-4 of 13 columns Previous **1** 2 3 4 5 6 ... 100 Next

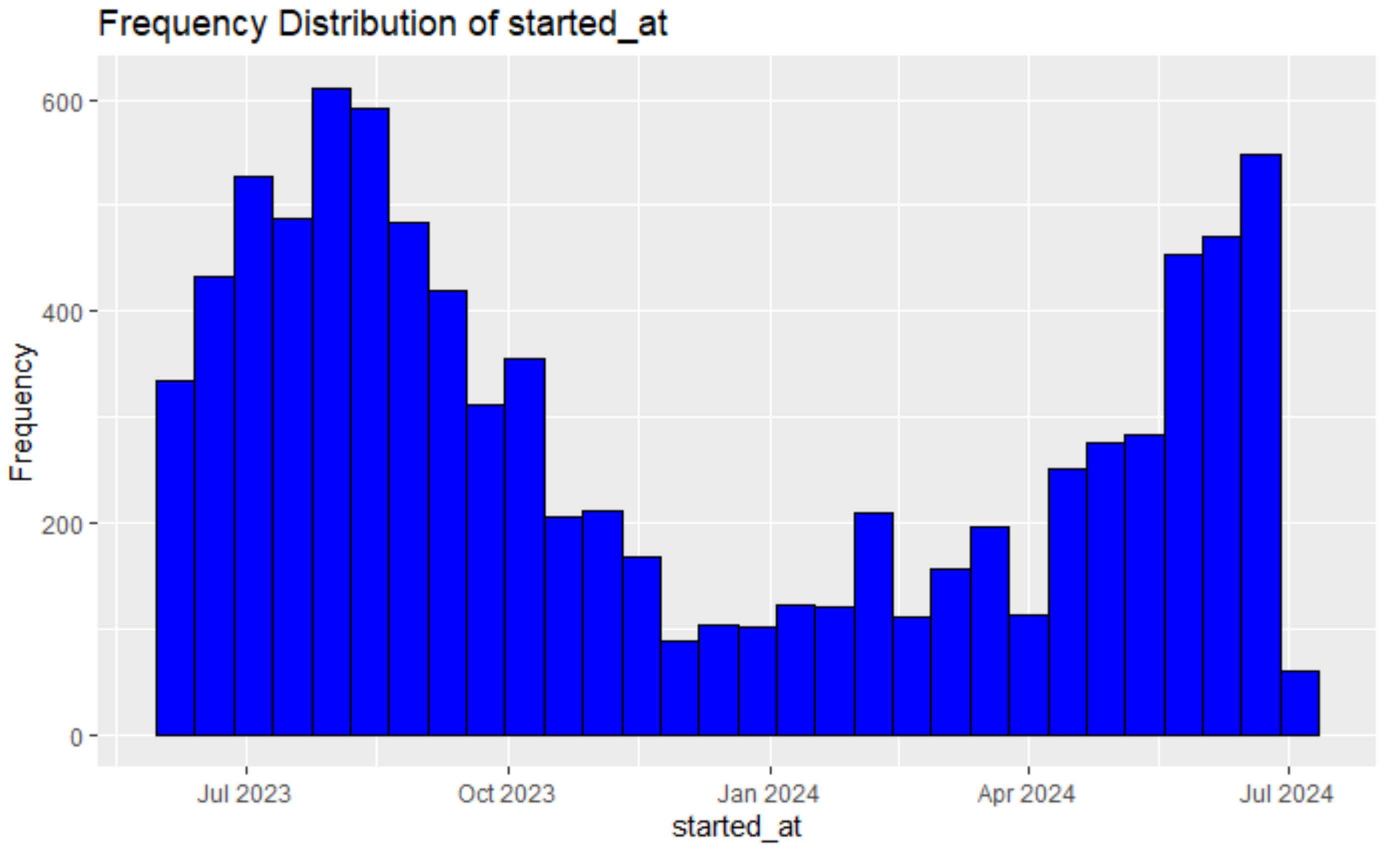
A lot of the data seems to also have missing end station names. I also don't see any electric bikes on the type of bike.

Are dates related?

Hide

```
ggplot(missing_lng_lat, aes(x = started_at)) +
    geom_histogram(fill = "blue", color = "black") +
    labs(title = "Frequency Distribution of started_at", x = "started_at", y = "Frequency")
```

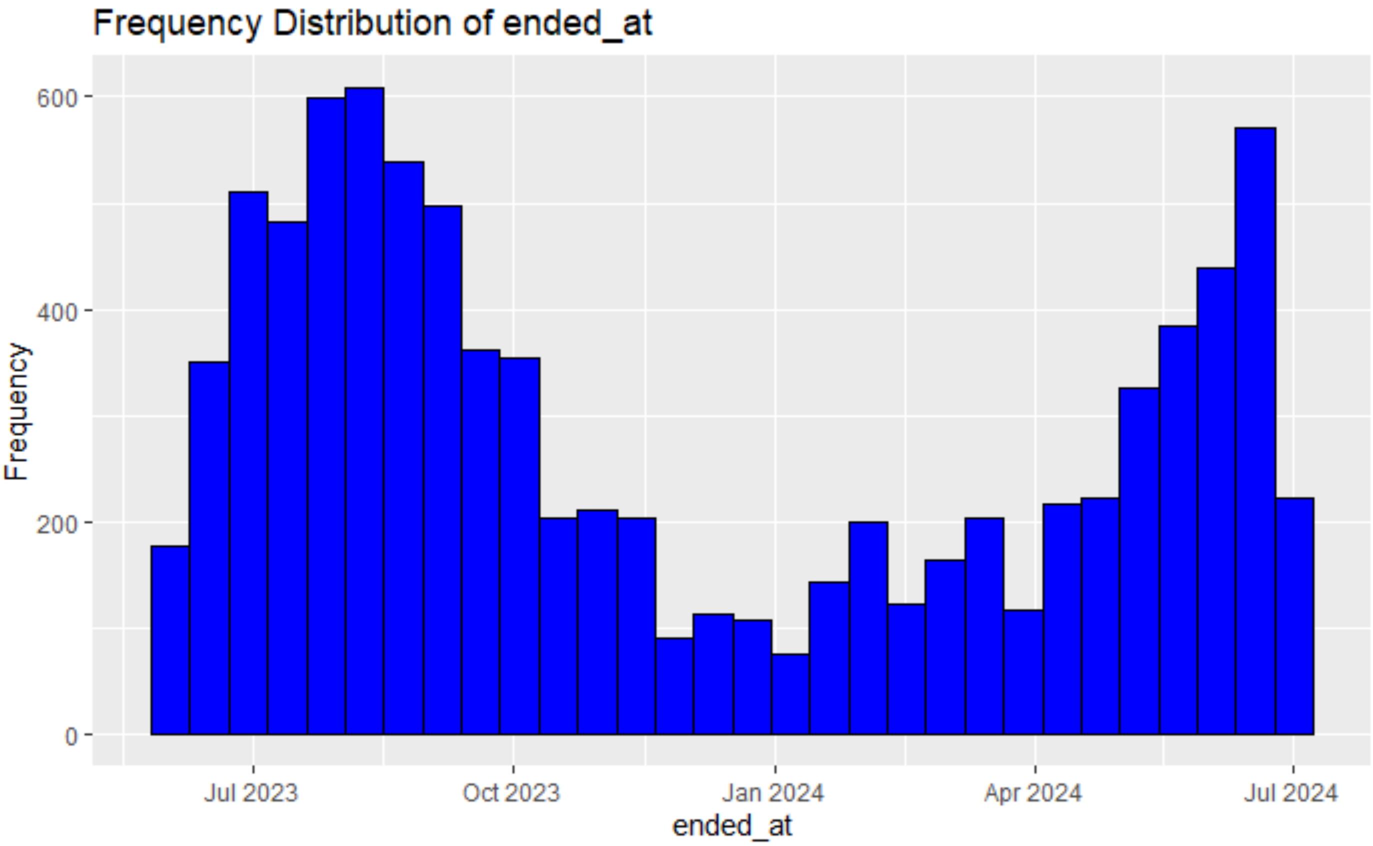
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Hide

```
ggplot(missing_lng_lat, aes(x = ended_at)) +
    geom_histogram(fill = "blue", color = "black") +
    labs(title = "Frequency Distribution of ended_at", x = "ended_at", y = "Frequency")
```

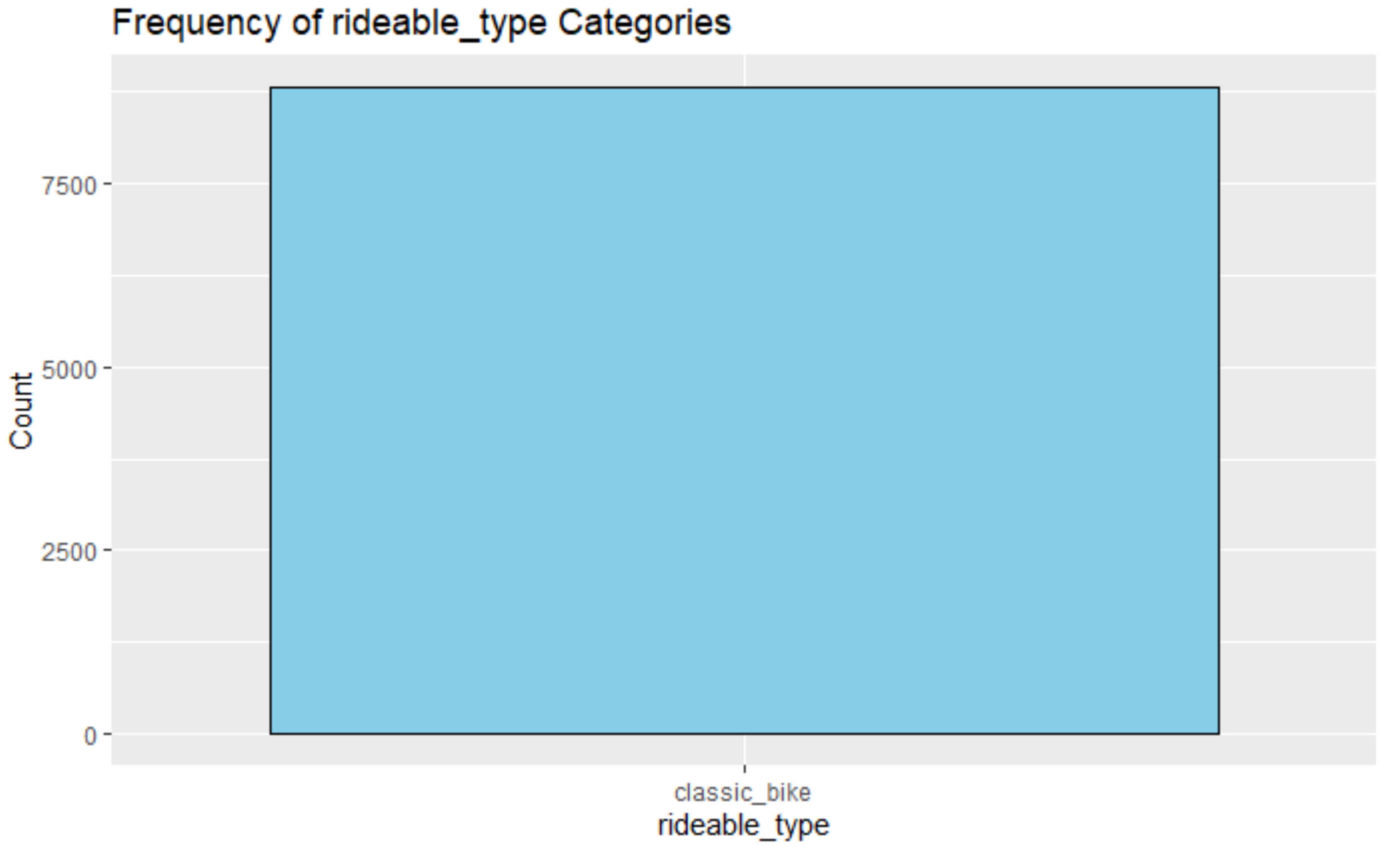
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The dates have the same spread as the full dataset.

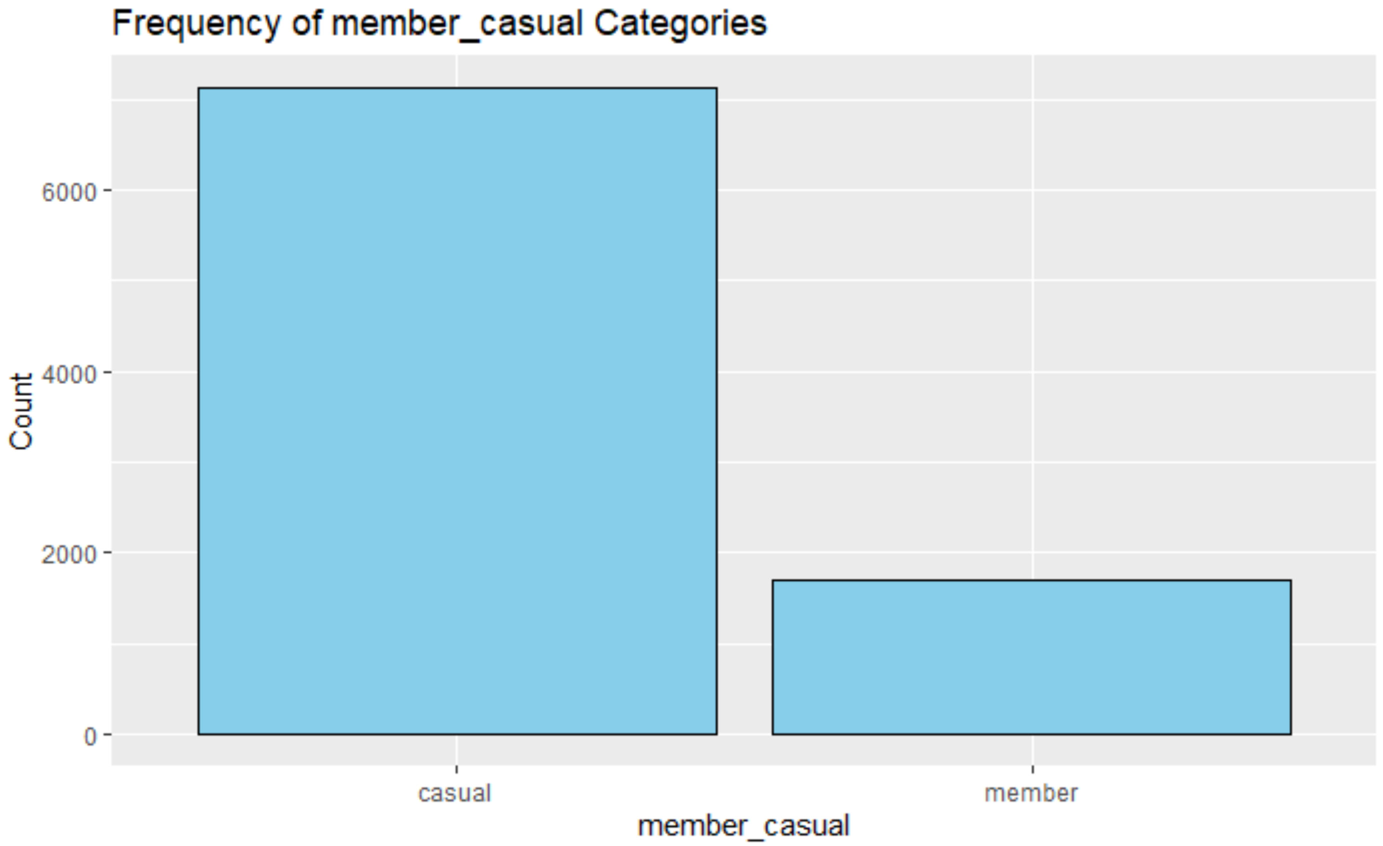
Hide

```
ggplot(missing_lng_lat, aes(x = rideable_type)) +
    geom_bar(fill = "skyblue", color = "black") +
    labs(title = "Frequency of rideable_type Categories", x = "rideable_type", y = "Count")
```



Hide

```
ggplot(missing_lng_lat, aes(x = member_casual)) +
    geom_bar(fill = "skyblue", color = "black") +
    labs(title = "Frequency of member_casual Categories", x = "member_casual", y = "Count")
```



There are no electric bikes on the subset at all. There also are more casuals than members. This is weird because there are almost double the members on the data.

My current hypothesis is that this bikes were lost/stolen. And since the classic bikes have no tracker there would be no longitude, lattitude. couple that with the fact that there are probably more casual drivers without the gps tracker on on their phones making it the reason why there are not that many members with missnig end_lng and end_lat. since that is how those values are obtained when the driver uses a classic bike.

Hide

```
missing_lng_lat_with_durations <- missing_lng_lat %>%
  select(everything()) %>%
  mutate(trip_duration = as.numeric(difftime(ended_at, started_at), units = "hours"))
```

Hide

```
missing_lng_lat_with_durations %>%
  select(everything())
```

ride_id	rideable_type	started_at	ended_at
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>
685DB4D7A6AF6CAE	classic_bike	2023-06-29 17:35:41	2023-06-29 17:45:38
E4AB9F672ECD0966	classic_bike	2023-06-29 14:50:04	2023-06-29 15:06:49
B0A0B0C83B363BC3	classic_bike	2023-06-23 16:06:21	2023-06-23 16:16:55
D49A2A420AE05181	classic_bike	2023-06-30 17:08:02	2023-07-01 18:07:52
2C35DCA44370EAD8	classic_bike	2023-06-30 18:45:11	2023-06-30 19:06:34
D19CA7CA83B28F88	classic_bike	2023-06-29 17:36:45	2023-06-29 18:02:39
1005EAF8E1C29D5C	classic_bike	2023-06-24 13:56:09	2023-06-24 14:18:23
B34C24AD17CCB667	classic_bike	2023-06-22 22:32:59	2023-06-22 23:09:50
64D120B77FA6F330	classic_bike	2023-06-22 22:32:41	2023-06-22 23:09:48
92008CFA88E93F44	classic_bike	2023-06-22 22:33:03	2023-06-22 23:09:44

1-10 of 8,809 rows | 1-4 of 14 columns Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
summary(missing_lng_lat_with_durations$trip_duration)
```

```
    Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
  0.0342   24.9942   24.9978   34.8923   24.9986 1641.4844 
```

For some reason there are huge outliers with some trips lasting thousands of hours. However if those outliers are removed most of the data ends up being 4 hours. As per divvy rules (https://help.divvybikes.com/hc/en-us/articles/360033123412-My-bike-was-lost-or-stolen) a trip cannot last longer than 24 hours so I will limit the search to that time.

Hide

```
missing_lng_lat_with_durations %>%
  select(everything()) %>%
  filter(trip_duration < 23)
```

ride_id	rideable_type	started_at	ended_at ▸
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>
685DB4D7A6AF6CAE	classic_bike	2023-06-29 17:35:41	2023-06-29 17:45:38
E4AB9F672ECD0966	classic_bike	2023-06-29 14:50:04	2023-06-29 15:06:49
B0A0B0C83B363BC3	classic_bike	2023-06-23 16:06:21	2023-06-23 16:16:55
2C35DCA44370EAD8	classic_bike	2023-06-30 18:45:11	2023-06-30 19:06:34
D19CA7CA83B28F88	classic_bike	2023-06-29 17:36:45	2023-06-29 18:02:39
1005EAF8E1C29D5C	classic_bike	2023-06-24 13:56:09	2023-06-24 14:18:23
B34C24AD17CCB667	classic_bike	2023-06-22 22:32:59	2023-06-22 23:09:50
64D120B77FA6F330	classic_bike	2023-06-22 22:32:41	2023-06-22 23:09:48
92008CFA88E93F44	classic_bike	2023-06-22 22:33:03	2023-06-22 23:09:44
84106BA64096E4EC	classic_bike	2023-06-30 14:06:15	2023-06-30 14:09:53

1-10 of 477 rows | 1-4 of 14 columns   Previous **1** 2 3 4 5 6 ... 48 Next

Hide

```
missing_lng_lat_with_durations %>%
  select(everything()) %>%
  filter(trip_duration >= 23)
```

ride_id	rideable_type	started_at	ended_at ▸
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>
D49A2A420AE05181	classic_bike	2023-06-30 17:08:02	2023-07-01 18:07:52
02D5A44D96224756	classic_bike	2023-06-08 22:28:04	2023-06-09 23:27:57
83A29914F1691318	classic_bike	2023-06-30 18:19:11	2023-07-01 19:18:55
B8BCACEAFE5A25B2	classic_bike	2023-06-23 14:01:53	2023-06-24 15:01:45
2C220E5809EEAFF4	classic_bike	2023-06-17 15:22:41	2023-06-18 16:22:34
E79E266999056B7C	classic_bike	2023-06-23 15:23:32	2023-06-25 04:53:39
3A508060E0A3498B	classic_bike	2023-06-04 06:24:51	2023-06-12 05:20:17
434701FA9456E46A	classic_bike	2023-06-23 02:50:25	2023-06-24 03:50:19
DC8F171134015094	classic_bike	2023-06-14 22:19:26	2023-06-15 23:19:21
E4A2932EBE3AFB20	classic_bike	2023-06-17 13:18:16	2023-06-20 11:58:24

1-10 of 8,332 rows | 1-4 of 14 columns   Previous **1** 2 3 4 5 6 ... 100 Next

Hide

```
NA
```

Hide

```
missing_lng_lat_with_durations %>%
  filter(trip_duration < 23 & complete.cases(end_station_name)) %>%
  count()
```

```
    n
<int>
  117
1 row
```

Hide

```

missing_lng_lat_with_durations %>%
  filter(trip_duration >= 23 & complete.cases(end_station_name)) %>%
  count()
```

```
    n
<int>
    0
1 row
```

Most of this data is greater than or equal to 24 which shows me that the great majority are lost bikes. So to clean this data I will do the following:

If the trips is longer than or equal 23 hours it will be marked as Lost/Stolen.

If the trips is shorter than 23 hours. It will be deleted. Unless the trip has an end_station_name. In which case I will use the median lng and lat of the the trips with the same station to fill out the end_lng and end_lat variables.

Hide

```
short_trips_to_remove_list <- missing_lng_lat_with_durations %>%
  filter(trip_duration < 23 & !complete.cases(end_station_name)) %>%
  pull(ride_id)


short_trips_to_fix_list <- missing_lng_lat_with_durations %>%
  filter(trip_duration < 23 & complete.cases(end_station_name)) %>%
  pull(ride_id)

lost_bikes_list <- missing_lng_lat_with_durations %>%
  filter(trip_duration > 23) %>%
    pull(ride_id)

trips_df <- trips_df %>%
  mutate(
    end_station_name = ifelse(ride_id %in% lost_bikes_list, "Lost",end_station_name),
    end_station_id = ifelse(ride_id %in% lost_bikes_list, "Lost",end_station_id),
  )
trips_df <- trips_df %>%
  filter(!(ride_id %in% short_trips_to_remove_list))



median_df <- trips_df %>%
  group_by(end_station_name) %>%
  summarize(median_end_lng = median(end_lng, na.rm = TRUE),
            median_end_lat = median(end_lat, na.rm = TRUE)
            )



trips_df <- trips_df %>%
    left_join(median_df, by = "end_station_name") %>%
    mutate(end_lng = ifelse(!complete.cases(end_lng) | (ride_id %in% short_trips_to_fix_list), median_end_l
ng, end_lng),
           end_lat = ifelse(!complete.cases(end_lat) | (ride_id %in% short_trips_to_fix_list), median_end_l
at, end_lat)) %>%
    select(-median_end_lng, -median_end_lat)
```

Hide

```
trips_df %>%
  filter(!complete.cases(end_lat, end_lng) & end_station_name != "Lost") %>%
  count()
```

```
    n
<int>
    0
1 row
```

The end_lng and end_lat will both still have Na’s But only on the trips that are marked as lost. I don’t want to mark end_lng/lat as lost since that will cause the whole column to become character type instead of a numeric type.

#mising data on names and ids

“You can either dock or lock your ebike (not both at once). Dock at any Divvy station, or use the cable to lock at any e-station or at the 500+ Divvy approved public bike racks for no additional cost. For an extra $2.40 ($1.20 for Divvy members), you can also lock to any other public bike rack, light pole, signpost, or retired parking meter within the service area." source (https://divvybikes.com/how-it-works/parking)

With this in mind this could be a good reason why there are so many start and end ids and names. Because some Ebikes can start and end a trip outside a station. I want to confirm this being looking at which types of bikes are missing its stations.

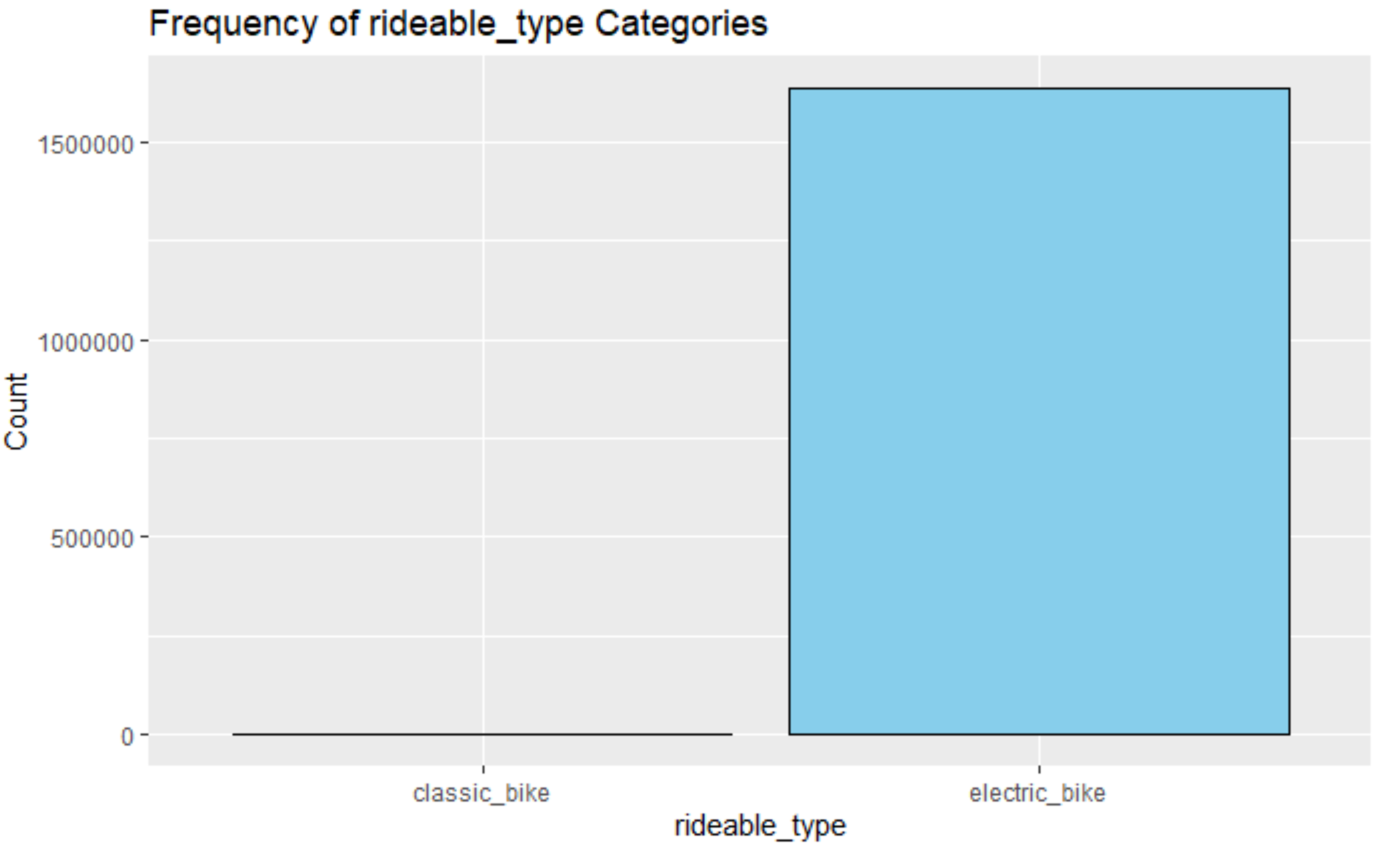
Hide

```
missing_stations <- trips_df %>%
  select(everything()) %>%
  filter(!complete.cases(start_station_id,end_station_id))

missing_stations_with_duration <- missing_stations %>%
  select(everything()) %>%
  mutate(trip_duration = as.numeric(difftime(ended_at, started_at), units = "hours"))
```

Hide

```
ggplot(missing_stations, aes(x = rideable_type)) +
    geom_bar(fill = "skyblue", color = "black") +
    labs(title = "Frequency of rideable_type Categories", x = "rideable_type", y = "Count")
```



The great majority are electric and rarely they are 24.Now how many electric bike trips are 24 hours?

Hide

```
missing_stations_with_duration %>%
  filter(rideable_type == "electric_bike", trip_duration > 23) %>%
  count()
```

```
    n
<int>
```

n
<int>
0

1 row

Hide

```
missing_stations_with_duration %>%
  filter(rideable_type == "classic_bike", trip_duration > 23) %>%
  count()
```

n
<int>
243

1 row

Which confirms that all bikes that all classic bikes which have missing stations are lost while electric ones aren’t. I will now mark the bikes which started out of the staion as out of station.

Hide

```
trips_df <- trips_df %>%
  mutate(start_station_name = ifelse(rideable_type == "electric_bike" & is.na(start_station_id),
                                   "Out of station", start_station_name))

trips_df <- trips_df %>%
  mutate(start_station_id = ifelse(rideable_type == "electric_bike" & is.na(start_station_id),
                                   "Out of station", start_station_id))

trips_df <- trips_df %>%
  mutate(end_station_name = ifelse(rideable_type == "electric_bike" & is.na(end_station_id),
                                   "Out of station", end_station_name))

trips_df <- trips_df %>%
  mutate(end_station_id = ifelse(rideable_type == "electric_bike" & is.na(end_station_id),
                                   "Out of station", end_station_id))
```

now lets deal with the classic bikes. first the ones lasting longer than 24 hours are lost/stolen so lets fix that.

Hide

```
trips_df <- trips_df %>%
  mutate(end_station_name = ifelse(rideable_type == "classic_bike" & is.na(end_station_id),
                                   "Lost", end_station_name))

trips_df <- trips_df %>%
  mutate(end_station_id = ifelse(rideable_type == "classic_bike" & is.na(end_station_id),
                                   "Lost", end_station_id))
```

Hide

```
trips_df %>%
  filter(!complete.cases(.) & end_station_id != "Lost")
```

ride_id	rideable_type	started_at	ended_at	start_station_name
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>	<chr>
9D6A7C67EFB688FD	classic_bike	2023-10-28 15:29:02	2023-10-28 15:40:51	NA
62F2649E24950EED	classic_bike	2023-10-28 22:02:07	2023-10-28 22:16:04	NA
5685FB0E8547AD9E	classic_bike	2023-10-27 16:14:14	2023-10-27 16:27:02	NA
A4F8B87B8F8DB9FE	classic_bike	2023-10-27 16:23:25	2023-10-27 16:30:47	NA
938CB5BE37B0378E	classic_bike	2023-11-05 09:00:18	2023-11-05 09:10:07	NA
66661D737DFC2B72	classic_bike	2023-11-05 12:06:15	2023-11-05 12:15:12	NA

6 rows | 1-5 of 13 columns

Because its only 6 rows of data and I don't know the reason why the stations are missing I will remove this rows.

Hide

```
trips_df <- trips_df %>%
  filter(complete.cases(.) | end_station_id == "Lost")
```

Hide

```
trips_df %>%
  select(everything()) %>%
  filter(!complete.cases(.) & end_station_id != "Lost") %>%
  count()
```

n
<int>
0

1 row

this section is done.

#lattitude and longitude low accuracy.

There are inconsistencies with decimal points on the latitude and longitude the decimal values go form 2 decimal places to up to 13. I would need at least 4 to have a somewhat accurate analysis since bellow 4 there is an error of 1.11 km. Which is more than enough to have more than one station at the same location. This canno't be changed but I will abstain myself from checking the distance between beginning and end.

#Creating a new variable.

Hide

```
trips_df %>%
  select(everything())
```

ride_id	rideable_type	started_at	ended_at	start_station_name
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>	<chr>
6F1682AC40EB6F71	electric_bike	2023-06-05 13:34:12	2023-06-05 14:31:56	Out of station
622A1686D64948EB	electric_bike	2023-06-05 01:30:22	2023-06-05 01:33:06	Out of station

ride_id	rideable_type	started_at	ended_at	start_station_name
<chr>	<chr>	<S3: POSIXct>	<S3: POSIXct>	<chr>
3C88859D926253B4	electric_bike	2023-06-20 18:15:49	2023-06-20 18:32:05	Out of station
EAD8A5E0259DEC88	electric_bike	2023-06-19 14:56:00	2023-06-19 15:00:35	Out of station
5A36F21930D6A55C	electric_bike	2023-06-19 15:03:34	2023-06-19 15:07:16	Out of station
CF682EA7D0F961DB	electric_bike	2023-06-09 21:30:25	2023-06-09 21:49:52	Out of station
4910FBB710157754	electric_bike	2023-06-03 13:34:09	2023-06-03 13:34:28	Out of station
EA19D850A42F56D8	electric_bike	2023-06-03 13:34:46	2023-06-03 13:35:00	Out of station
E68F43784662A2D0	electric_bike	2023-06-02 22:27:35	2023-06-02 22:35:26	Out of station
5A013E29CC001611	electric_bike	2023-06-02 21:18:31	2023-06-03 01:27:19	Out of station

1-10 of 6,453,614 rows | 1-5 of 13 columns Previous **1** 2 3 4 5 6 ... 100 Next

I will need to new columns for the analysis trip_duration.

Hide

```
trips_df <- trips_df %>%
  select(everything()) %>%
  mutate(trip_duration = as.numeric(difftime(ended_at, started_at), units = "mins"))
```

Hide

```
summary(trips_df$trip_duration)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
-16656.52      5.60      9.83     18.47     17.45  98489.07 
```

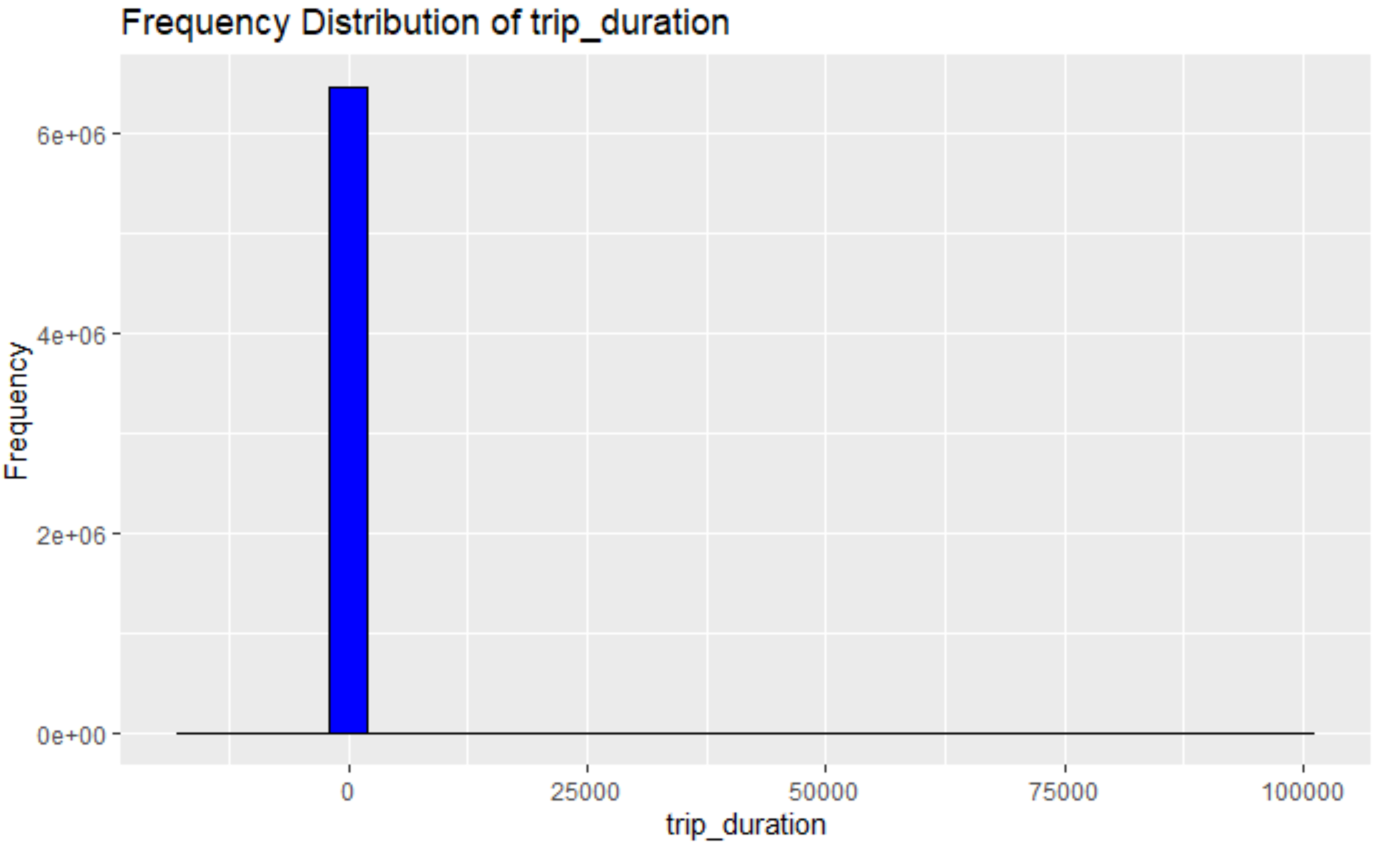
There are negative values on the trip duration. as well as a maximum of 1641.4844 hours on a trip. As we discussed before any trip above 24 hours is already considered a lost so this should not be possible. For the distance there 0.0 distances traveled which maybe trips that were started and ended immediately but I need to check further. there also seems to be very big maximun distance. I will also need to check this further.

Lets see how all of this problems look.

Hide

```
ggplot(trips_df, aes(x = trip_duration)) +
    geom_histogram(fill = "blue", color = "black") +
    labs(title = "Frequency Distribution of trip_duration", x = "trip_duration", y = "Frequency")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



I want to first explore the negative times. Which should not be possible.

Hide

```
trips_df %>% count()
```

```
                                                                        n
                                                                    <int>
                                                                  6453614
1 row
```

Hide

```
negative_trips <- trips_df %>%
  select(everything()) %>%
  filter(trip_duration <= 0)
```

This data encompasses 0.0006% of the total

Next lets check the trip duration that are longer than 24 hours since this should not be possible.

Hide

```
Long_trips <- trips_df %>%
  select(everything()) %>%
  filter(trip_duration > 1440 & end_station_name != "Lost")
```

This encompasses 0.0008% of the data.

This trips should not be possible, and due to them together en composing 0.001% of the data. I will delete them to avoid it from affecting the analysis.

Hide

```
trips_df <- trips_df %>%
  select(everything()) %>%
  filter(!(trip_duration > 1440 & end_station_name != "Lost"))

trips_df <- trips_df %>%
  select(everything()) %>%
  filter(trip_duration > 0)
```

#creating categorical columns

rideable_type and member_casual, start_station_name, start_station_id, end_station_name, end_station_id are all categorical variables on the data set. I transform them into factors on the code bellow.

Hide

```
trips_df$rideable_type <- as.factor(trips_df$rideable_type)
class(trips_df$rideable_type)
```

```
[1] "factor"
```

Hide

```
trips_df$member_casual <- as.factor(trips_df$member_casual)
class(trips_df$rideable_type)
```

```
[1] "factor"
```

Hide

```
trips_df$start_station_id <- as.factor(trips_df$start_station_id)
class(trips_df$start_station_id)
```

```
[1] "factor"
```

Hide

```
trips_df$start_station_name <- as.factor(trips_df$start_station_name)
class(trips_df$start_station_name)
```

```
[1] "factor"
```

Hide

```
trips_df$end_station_name <- as.factor(trips_df$end_station_name)
class(trips_df$end_station_name)
```

```
[1] "factor"
```

Hide

```
trips_df$end_station_id <- as.factor(trips_df$end_station_id)
class(trips_df$end_station_id)
```

```
[1] "factor"
```

With this I conclude the cleaning.

Hide

```
write.csv(trips_df, "../data/trips_data_cleaned.csv", row.names = FALSE)
```