databricks 02 eda

(https://databricks.com)
```
%run ./includes/includes
```

```
Out[3]: DataFrame[]
```

VERY IMPORTANT TO UNDERSTAND THE USE OF THESE VARIABLES! Please ask if you are confused about their use.

Variable Name	Value	Description
NYC_WEATHER_FILE_PATH	dbfs:/FileStore/tables/raw/weather/	Historic NYC Weather for Model Building
BIKE_TRIP_DATA_PATH	dbfs:/FileStore/tables/raw/bike_trips/	Historic Bike Trip Data for Model Building (Stream this data source)
BRONZE_STATION_INFO_PATH	dbfs:/FileStore/tables/bronze_station_info.delta	Station Information (30 min refresh)
BRONZE_STATION_STATUS_PATH	dbfs:/FileStore/tables/bronze_station_status.delta	Station Status (30 min refresh)
BRONZE_NYC_WEATHER_PATH	dbfs:/FileStore/tables/bronze_nyc_weather.delta	NYC Weather (30 min refresh)
USER_NAME	jtschopp@u.rochester.edu	Email of the user executing this code/notebook
GROUP_NAME	G11	Group Assigment for this user
GROUP_STATION_ASSIGNMENT	Cleveland Pl & Spring St	Station Name to be modeled by this group
GROUP_DATA_PATH	dbfs:/FileStore/tables/G11/	Path to store all of your group data files (delta ect)
GROUP_MODEL_NAME	G11_model	Mlflow Model Name to be used to register your model
GROUP_DB_NAME	G11_db	Group Database to store any managed tables (pre-defined for you)

Organization of the Notebook

The first half of this notebook is mostly pandas-profiling results showing us information about the individual variables. The second half are graphs and relationships between hours, days, weeks, months, years, and weather and their relationship with the number of trips that occur.

```
pip install -U pandas-profiling
```

```
Python interpreter will be restarted.
Requirement already satisfied: pandas-profiling in /databricks/python3/lib/python3.9/site-packages (3.1.0)
Collecting pandas-profiling
  Downloading pandas_profiling-3.6.6-py2.py3-none-any.whl (324 kB)
Collecting ydata-profiling
  Downloading ydata_profiling-4.1.2-py2.py3-none-any.whl (345 kB)
Requirement already satisfied: tqdm<4.65,>=4.48.2 in /databricks/python3/lib/python3.9/site-packages (from ydata-profilin
g->pandas-profiling) (4.62.3)
Collecting imagehash==4.3.1
  Downloading ImageHash-4.3.1-py2.py3-none-any.whl (296 kB)
Requirement already satisfied: requests<2.29,>=2.24.0 in /databricks/python3/lib/python3.9/site-packages (from ydata-prof
iling->pandas-profiling) (2.26.0)
Requirement already satisfied: scipy<1.10,>=1.4.1 in /databricks/python3/lib/python3.9/site-packages (from ydata-profilin
g->pandas-profiling) (1.7.1)
Collecting typeguard<2.14,>=2.13.2
  Downloading typeguard-2.13.3-py3-none-any.whl (17 kB)
Requirement already satisfied: pydantic<1.11,>=1.8.1 in /databricks/python3/lib/python3.9/site-packages (from ydata-profi
ling->pandas-profiling) (1.9.2)
Requirement already satisfied: seaborn<0.13,>=0.10.1 in /databricks/python3/lib/python3.9/site-packages (from ydata-profi
ling->pandas-profiling) (0.11.2)
Requirement already satisfied: htmlmin==0.1.12 in /databricks/python3/lib/python3.9/site-packages (from ydata-profiling->
```

```
pip install holidays
```

```
Python interpreter will be restarted.
Requirement already satisfied: holidays in /databricks/python3/lib/python3.9/site-packages (0.15)
Requirement already satisfied: python-dateutil in /databricks/python3/lib/python3.9/site-packages (from holidays) (2.8.2)
Requirement already satisfied: convertdate>=2.3.0 in /databricks/python3/lib/python3.9/site-packages (from holidays) (2.
4.0)
Requirement already satisfied: hijri-converter in /databricks/python3/lib/python3.9/site-packages (from holidays) (2.2.4)
Requirement already satisfied: korean-lunar-calendar in /databricks/python3/lib/python3.9/site-packages (from holidays)
(0.3.1)
```

```
Requirement already satisfied: pymeeus<=1,>=0.3.13 in /databricks/python3/lib/python3.9/site-packages (from convertdate>=
2.3.0->holidays) (0.5.11)
Requirement already satisfied: six>=1.5 in /databricks/python3/lib/python3.9/site-packages (from python-dateutil->holiday
s) (1.16.0)
Python interpreter will be restarted.
```

Imports

```
from pathlib import Path
from pyspark.sql.functions import *

import matplotlib.pyplot as plt
import seaborn as sns

import numpy as np
import requests

import pandas_profiling
import pandas as pd
from pandas_profiling.utils.cache import cache_file
```

```
historic_trip_data_df = (spark.read
     .format("delta")
     .load("dbfs:/FileStore/tables/G11/bronze/historic_trip_data/"))
historic_trip_data_df.display()
historic_trip_data_df.printSchema()
historic_trip_data_df.count()
```

Table

	ride_id	rideable_type	started_at	ended_at	start_station_name
1	CC063972EDD9AE33	classic_bike	2021-11-02T21:08:54.000+0000	2021-11-02T21:18:44.000+0000	Cleveland Pl & Spring St
2	8D9AC22469D10D86	classic_bike	2021-11-29T18:19:20.000+0000	2021-11-29T18:25:20.000+0000	Cleveland Pl & Spring St
3	850908D3431D0402	classic_bike	2021-11-24T23:36:41.000+0000	2021-11-24T23:43:18.000+0000	Cleveland Pl & Spring St
4	35DD206C234BFA8C	classic_bike	2021-11-28T13:18:16.000+0000	2021-11-28T13:26:10.000+0000	Cleveland Pl & Spring St
5	30E796AD561C7355	electric_bike	2021-11-27T11:47:13.000+0000	2021-11-27T11:57:29.000+0000	Cleveland Pl & Spring St
6	BD57CB4E01F71249	electric_bike	2021-11-28T13:18:14.000+0000	2021-11-28T13:24:44.000+0000	Cleveland Pl & Spring St
7	B62422180DDC6F22	classic_bike	2021-11-16T10:51:05.000+0000	2021-11-16T11:08:01.000+0000	Cleveland Pl & Spring St

8,634 rows | Truncated data

```
root
 |-- ride_id: string (nullable = true)
 |-- rideable_type: string (nullable = true)
 |-- started_at: timestamp (nullable = true)
 |-- ended_at: timestamp (nullable = true)
 |-- start_station_name: string (nullable = true)
 |-- start_station_id: double (nullable = true)
 |-- end_station_name: string (nullable = true)
 |-- end_station_id: double (nullable = true)
 |-- start_lat: double (nullable = true)
 |-- start_lng: double (nullable = true)
 |-- end_lat: double (nullable = true)
 |-- end_lng: double (nullable = true)
 |-- member_casual: string (nullable = true)

Out[2]: 235560
```

```
bronze_station_status_df = (spark.read
                            .format("delta")
                            .load("dbfs:/FileStore/tables/G11/bronze/station_status"))
bronze_station_status_df.display()
```

Table

	num_ebikes_available	is_installed	num_docks_available	num_scooters_unavailable	num_scooters_available	station_id
1	2	1	16	0	0	66db2fd0-
2	1	1	12	0	0	66db2fd0-
3	6	1	8	0	0	66db2fd0-
4	3	1	9	0	0	66db2fd0-
5	6	1	8	0	0	66db2fd0-
6	9	1	3	0	0	66db2fd0-
7	6	1	9	0	0	66db2fd0-

2,197 rows

```
from pyspark.sql.functions import *
df2 = (historic_trip_data_df.withColumn("day", dayofyear("started_at")))
df3 = (df2.select("day", "rideable_type"))
df3.display()
```

Table

	day	rideable_type
1	306	classic_bike
2	333	classic_bike
3	328	classic_bike
4	332	classic_bike
5	331	electric_bike
6	332	electric_bike
7	320	classic_bike

10,000 rows | Truncated data

```
from pyspark.sql.functions import *
import pandas as pd
import matplotlib.pyplot as plt

df2 = (historic_trip_data_df.withColumn("day", dayofyear("started_at")))
df3 = (df2.select("day", "rideable_type"))
df4 = df2.groupBy(df3.day).count().orderBy(df2.day)
df4.show(31)
#df4.select('count').hist(by=df4.select('day'))
```

```
+---+-----+
|day|count|
+---+-----+
|  1|  382|
|  2|  619|
|  3|  458|
|  4|  692|
|  5|  602|
|  6|  632|
|  7|  445|
|  8|  442|
|  9|  486|
| 10|  578|
| 11|  521|
| 12|  556|
| 13|  759|
| 14|  505|
| 15|  377|
| 16|  425|
| 17|  542|
| 18|  711|
```

```
import pandas_profiling
import pandas as pd
from pandas_profiling.utils.cache import cache_file
import numpy as np


historic_trip_data_df = historic_trip_data_df.select("*").toPandas()
historic_trip_data_df['started_at']= pd.to_datetime(historic_trip_data_df['started_at'])
historic_trip_data_df['ended_at']= pd.to_datetime(historic_trip_data_df['ended_at'])


historic_trip_data_profile = pandas_profiling.ProfileReport(historic_trip_data_df)
historic_trip_data_profile
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	13
Number of observations	235560
Missing cells	897
Missing cells (%)	< 0.1%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	23.4 MiB
Average record size in memory	104.0 B

Variable types

Categorical	5
DateTime	2
Numeric	6

Alerts

`ride_id` has a high cardinality: 235560 distinct values	High cardinality
`start_station_name` has a high cardinality: 1192 distinct values	High cardinality
`end_station_name` has a high cardinality: 1191 distinct values	High cardinality
`start_station_id` is highly overall correlated with `start_lat`	High correlation
`end_station_id` is highly overall correlated with `end_lat`	High correlation
`start_lat` is highly overall correlated with `start_station_id`	High correlation
`end_lat` is highly overall correlated with `end_station_id`	High correlation

Out[6]:

```
bronze_station_info_df = (spark.read
                          .format("delta")
                          .load("dbfs:/FileStore/tables/G11/bronze/station_info"))
bronze_station_info_df.display()
```

Table

	has_kiosk	station_type	region_id	short_name	lat	electric_bike_surcharge_waiver	capacity
1	true	classic	71	5492.05	40.722103786686034	false	33

1 row

bronze_station_status

```
import pandas_profiling
import pandas as pd
from pandas_profiling.utils.cache import cache_file

bronze_station_info_df = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/bronze/station_info"))

df = bronze_station_info_df.select("*").toPandas()


bronze_station_status = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/bronze_station_status"))
bronze_station_status.display()
```

Table

	num_ebikes_available	is_installed	num_docks_available	num_scooters_unavailable	num_scooters_available	station_id
1	1	1	17	0	0	66db2fd0-
2	4	1	1	0	0	66db2fd0-
3	1	1	14	0	0	66db2fd0-
4	0	1	18	0	0	66db2fd0-
5	0	1	7	0	0	66db2fd0-
6	0	1	13	0	0	66db2fd0-
7	6	1	1	0	0	66db2fd0-

1,769 rows

bronze_station_info

```
bronze_station_info = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/bronze_station_info"))
bronze_station_info.display()
```

Table

	has_kiosk	station_type	region_id	short_name	lat	electric_bike_surcharge_waiver	capacity
1	true	classic	71	5492.05	40.722103786686034	false	33

1 row

```
historic_weather_df = (spark.readStream.format("delta").load("dbfs:/FileStore/tables/G11/bronze/historic_weather_data"))
historic_weather_df.display()

historic_weather_df = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/bronze/historic_weather_data"))
historic_weather_df.display()
historic_weather_df.printSchema()
```

display_query_1 (id: 08b0cd09-8d48-4024-8c0e-71080a7110eb) *Last updated: 15 seconds ago*

Table

	dt	temp	feels_like	pressure	humidity	dew_point	uvi	clouds	visibility
1	1637355600	280.6	276.92	1026	45	269.87	0.16	75	10000
2	1637359200	280.78	277.25	1026	44	269.75	0	61	10000
3	1637373600	279.23	275.71	1031	53	270.48	0	6	10000

3	1637373600	279.23	275.71	1031	53	270.48	0	6	10000	
4	1637384400	278.18	274.91	1032	57	270.42	0	6	10000	4.2
5	1637406000	277.25	276.04	1034	54	268.65	0	64	10000	1.5
6	1637420400	279.1	279.1	1035	43	267.5	1.2	11	10000	1.0
7	1637427600	280.65	279.11	1034	48	270.68	1.76	1	10000	2.3
8	1637434800	280.84	278.54	1033	45	270.07	0.91	41	10000	3.5
9	1637474400	281.33	279.06	1030	65	275.14	0	100	10000	3.7
10	1637481600	281.05	278.78	1029	69	275.58	0	100	10000	3.6
11	1637492400	280.89	278.99	1028	65	274.76	0	100	10000	2.9
12	1637499600	280.83	279.29	1028	64	274.31	0.14	100	10000	2.4
13	1637506800	282.01	280.3	1028	57	273.85	0.95	97	10000	2.9
14	1637528400	285.82	284.66	1022	58	277.78	0.14	40	10000	4.0
15	1637539200	284.8	283.87	1020	71	279.74	0	75	10000	4.5
16	1637542800	283.96	283.16	1020	79	280.44	0	100	10000	3.7
17	1637546400	283.79	283.08	1018	83	281.04	0	100	10000	4.5

1,000 rows | Truncated data

```
import pandas_profiling
import pandas as pd
from pandas_profiling.utils.cache import cache_file

historic_weather_data_df = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/bronze/historic_weather_data"))
df = historic_weather_data_df.select("*").toPandas()

#Reading stream for historic trip data bronze
historic_trip_data_df = (spark.read
     .format("delta")
     .load("dbfs:/FileStore/tables/G11/bronze/historic_trip_data"))
historic_trip_data_df.display()
historic_trip_data_df.printSchema()
#here we have a bar chart displaying the end spots of the bikes and which spots are most common
#It is also grouped by if its a member or a casual user
```

Table

	ride_id	rideable_type	started_at	ended_at	start_station_name
1	CC063972EDD9AE33	classic_bike	2021-11-02T21:08:54.000+0000	2021-11-02T21:18:44.000+0000	Cleveland Pl & Spring St
2	8D9AC22469D10D86	classic_bike	2021-11-29T18:19:20.000+0000	2021-11-29T18:25:20.000+0000	Cleveland Pl & Spring St
3	850908D3431D0402	classic_bike	2021-11-24T23:36:41.000+0000	2021-11-24T23:43:18.000+0000	Cleveland Pl & Spring St
4	35DD206C234BFA8C	classic_bike	2021-11-28T13:18:16.000+0000	2021-11-28T13:26:10.000+0000	Cleveland Pl & Spring St
5	30E796AD561C7355	electric_bike	2021-11-27T11:47:13.000+0000	2021-11-27T11:57:29.000+0000	Cleveland Pl & Spring St
6	BD57CB4E01F71249	electric_bike	2021-11-28T13:18:14.000+0000	2021-11-28T13:24:44.000+0000	Cleveland Pl & Spring St
7	B62422180DDC6F22	classic_bike	2021-11-16T10:51:05.000+0000	2021-11-16T11:08:01.000+0000	Cleveland Pl & Spring St

8,634 rows | Truncated data

```
root
 |-- ride_id: string (nullable = true)
 |-- rideable_type: string (nullable = true)
 |-- started_at: timestamp (nullable = true)
 |-- ended_at: timestamp (nullable = true)
 |-- start_station_name: string (nullable = true)
 |-- start_station_id: double (nullable = true)
 |-- end_station_name: string (nullable = true)
 |-- end_station_id: double (nullable = true)
 |-- start_lat: double (nullable = true)
 |-- start_lng: double (nullable = true)
 |-- end_lat: double (nullable = true)
 |-- end_lng: double (nullable = true)
 |-- member_casual: string (nullable = true)
```

historic_weather

```
#historic_weather_df = (spark.read
#                       .option("header", True)
#                       .csv("dbfs:/FileStore/tables/G11/historic_weather_df"))
#historic_weather_df.display()

dbfs:"/FileStore/tables/G11/"

#historic_weather = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/historic_weather_df"))
#historic_weather.display()
```

```
#Counts how many disctinct descriptions of the weather
#print("Distinct Count: " + str(historic_weather.select("description").distinct().count()))
#print("Distinct Count: " + str(historic_weather.select("description").distinct().count()))
```

```
#historic_weather_df = historic_weather.select("*").toPandas()
#historic_weather_profile = pandas_profiling.ProfileReport(historic_weather_df)
#historic_weather_profile
```

```
day2 = (historic_trip_data_df.withColumn("day", dayofyear("started_at")))
day3 = (day2.select("day", "rideable_type"))

day4 = day2.groupBy(day3.day).count().orderBy(day2.day)
day4.show(365)
```

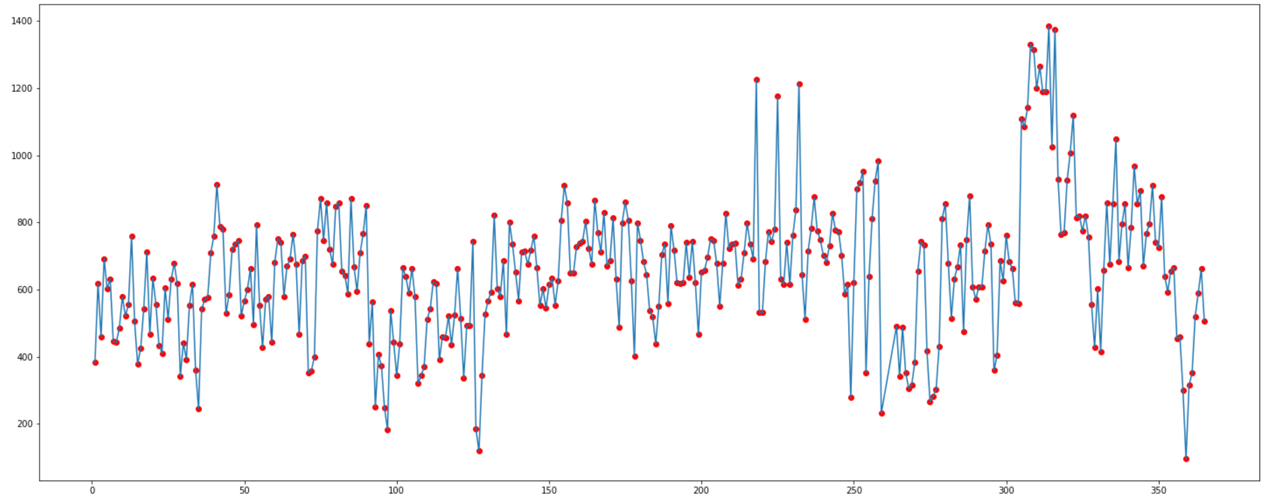
```
+---+-----+
|day|count|
+---+-----+
|  1|  382|
|  2|  619|
|  3|  458|
|  4|  692|
|  5|  602|
|  6|  632|
|  7|  445|
|  8|  442|
|  9|  486|
| 10|  578|
| 11|  521|
| 12|  556|
| 13|  759|
| 14|  505|
| 15|  377|
| 16|  425|
| 17|  542|
| 18|  711|
```

Daily Yearly Trends

```
import matplotlib.pyplot as plt

counts=day4.select('count').toPandas()
days=day4.select('day').toPandas()

plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(days,counts,color='red')
plot = plt.plot(days,counts)
plt.show()
```



Daily Yearly Trends and Holidays Explanations

Based on the graph above a we see that there is a lot of variations in the amount of trips taken throughout a year. It seems that there are four big spikes.

There are three spikes on August 6th, 13th and, 20th. These three spikes may be due to either very nice weather on those three days, or there could have been a lot of events on those three days. There is also a larger/longer spike from Novermber 1st to November 13. This may be due to people trying to buy gifts before the holiday season.

There are also a few dips in trips in April, May, and December. This makes sense since in the spring there tends to be a lot of rain and, in Decmeber it gets very cold, windy, and snowy neither of which is ideal for bike trips.

In terms of how holidays impact the number of trips there seems to have some impact. Specifically we see a big drop in trips on Christmas. There also a dip in rides on Memorial Day. Both these holidays causing a dip in rides makes sense because most people don't work on those days and aren't going to be traveling on bikes on holidays.

```
day2 = (historic_trip_data_df.withColumn("day", hour("started_at")))
day3 = (day2.select("day", "rideable_type"))

day4 = day2.groupBy(day3.day).count().orderBy(day2.day)
day4.show(24)
```

```
+---+-----+
|day|count|
+---+-----+
|  0| 3172|
|  1| 2060|
|  2| 1331|
|  3|  712|
|  4|  459|
|  5|  537|
|  6| 2339|
|  7| 4773|
|  8|10353|
|  9|11239|
| 10|10883|
| 11|12555|
| 12|14342|
| 13|16440|
| 14|17426|
| 15|19016|
| 16|19143|
| 17|22161|
```

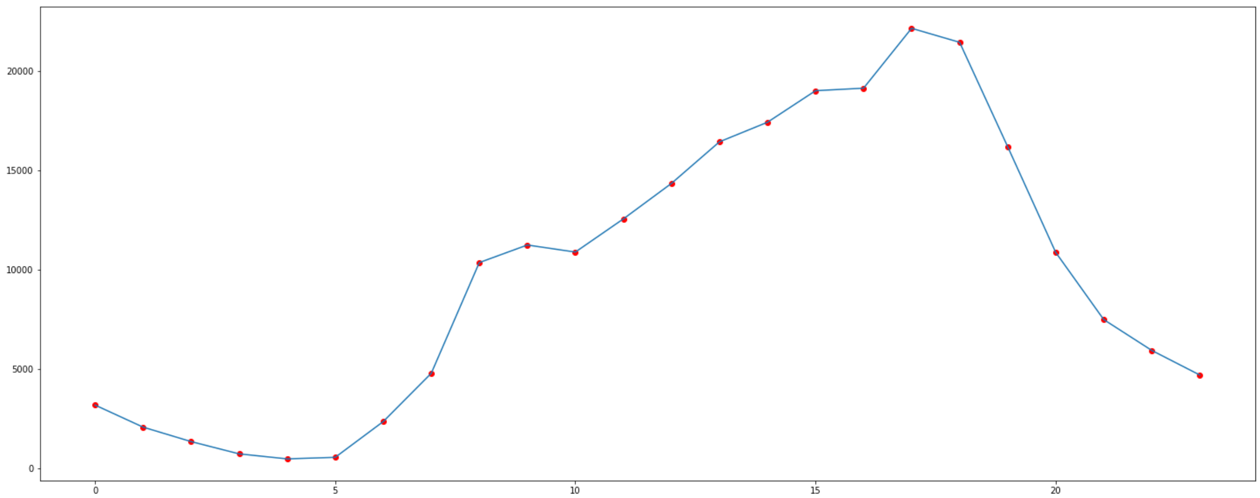
Hourly Trends

```
import matplotlib.pyplot as plt

counts=day4.select('count').toPandas()
days=day4.select('day').toPandas()

plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(days,counts,color='red')
plot = plt.plot(days,counts)

plt.show()
```



Hourly Trends

The hourly trends tell us that most rides occur during the afternoon. There is a gradual increase in riders throughout the day with it peaking at 5pm. This makes sense since most people get out of work in the afternoon.

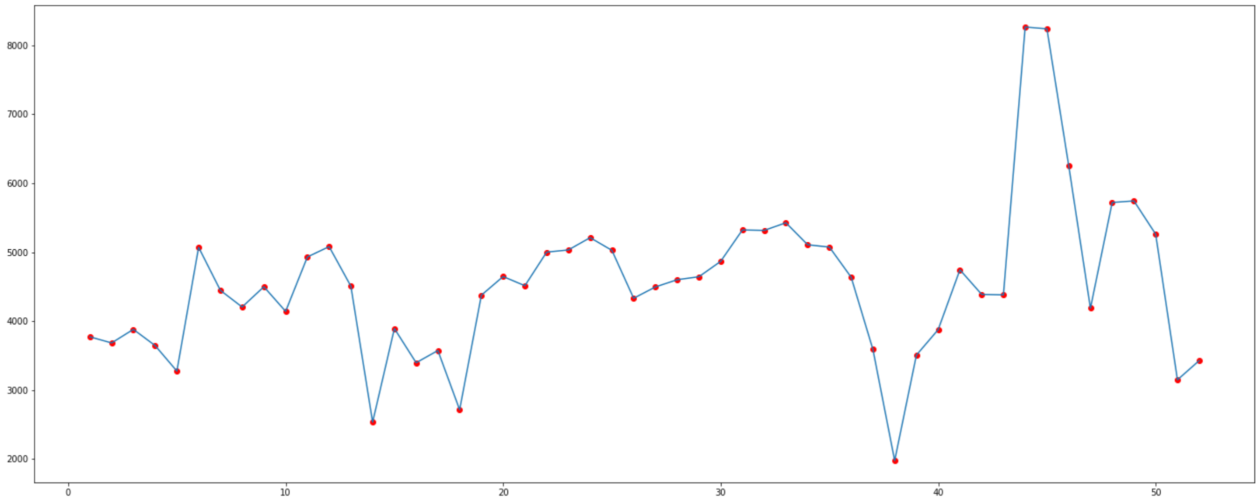
```
week2 = (historic_trip_data_df.withColumn("week", weekofyear("started_at")))
week3 = (week2.select("week", "rideable_type"))
week4 = week2.groupBy(week3.week).count().orderBy(week2.week)
week4.show(52)
```

```
+----+-----+
|week|count|
+----+-----+
|   1| 3772|
|   2| 3683|
|   3| 3880|
|   4| 3645|
|   5| 3275|
|   6| 5070|
|   7| 4446|
|   8| 4204|
|   9| 4498|
|  10| 4141|
|  11| 4930|
|  12| 5080|
|  13| 4506|
|  14| 2534|
|  15| 3892|
|  16| 3396|
|  17| 3573|
|  18| 2714|
```

Yearly Weekly Trends

```
import matplotlib.pyplot as plt
counts=week4.select('count').toPandas()
weeks=week4.select('week').toPandas()
plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(weeks,counts,color='red')
plot = plt.plot(weeks,counts)

plt.show()
```



Yearly Weekly Trends Explanation

The yearly weekly trends follows the same pattern as the daily yearly trends, but the graph is smoother. There is a two week peak in early Novmber and the lows in April, May, and December. We also see a strong deep in late September.

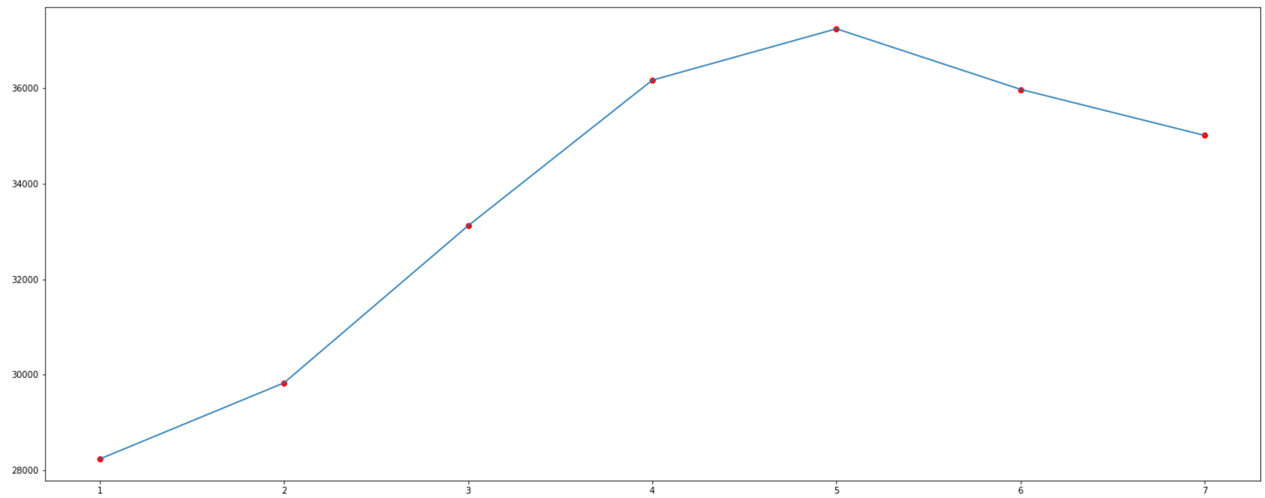
```
week5 = (historic_trip_data_df.withColumn("week", dayofweek("started_at")))
week6 = (week5.select("week", "rideable_type"))
week7 = week5.groupBy(week6.week).count().orderBy(week5.week)
week7.show()

+----+-----+
|week|count|
+----+-----+
|   1|28240|
|   2|29831|
|   3|33118|
|   4|36161|
|   5|37236|
|   6|35969|
|   7|35005|
+----+-----+
```

Weekly Trends

```
import matplotlib.pyplot as plt
counts=week7.select('count').toPandas()
weeks=week7.select('week').toPandas()
plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(weeks,counts,color='red')
plot = plt.plot(weeks,counts)

plt.show()
```



Weekly Trends Explanation

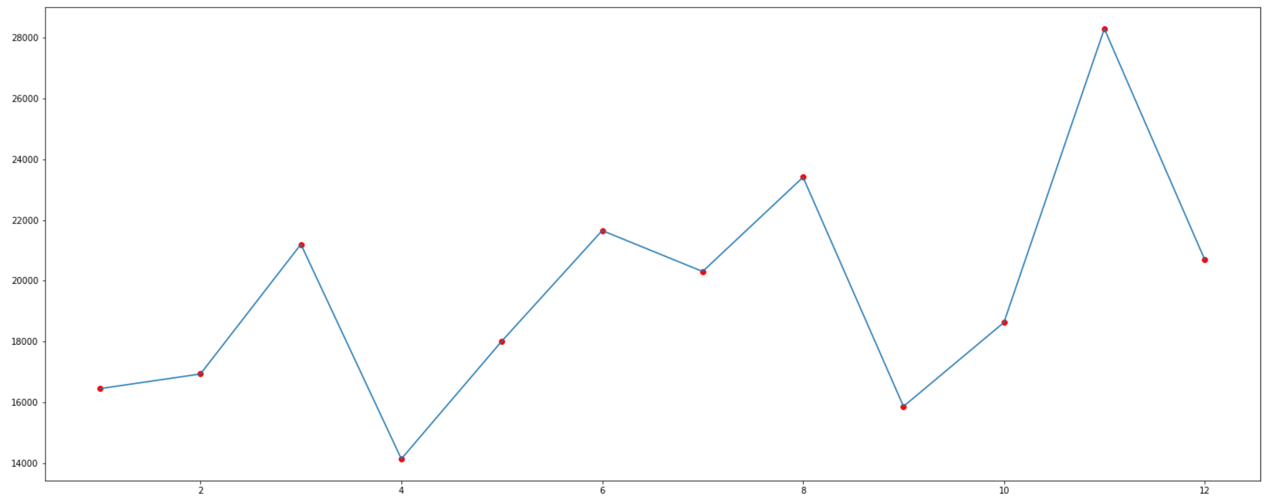
Based on the weekly trends that the most common days to ride bikes are Fridays, Saturdays, and Sundays. This makes sense because people tend to have the most time off on those days. Mondays likely have the lowest number of trips because people tend to be pretty low energy on Mondays.

```
month2 = (historic_trip_data_df.withColumn("month", month("started_at")))
month3 = (month2.select("month", "rideable_type"))
month4 = month2.groupBy(month3.month).count().orderBy(month2.month)
month4.show()
```

```
+-----+-----+
|month|count|
+-----+-----+
|    1|16448|
|    2|16930|
|    3|21200|
|    4|14133|
|    5|18010|
|    6|21652|
|    7|20304|
|    8|23408|
|    9|15863|
|   10|18626|
|   11|28296|
|   12|20690|
+-----+-----+
```

```
import matplotlib.pyplot as plt
counts=month4.select('count').toPandas()
months=month4.select('month').toPandas()
plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(months,counts,color='red')
plot = plt.plot(months,counts)

plt.show()
```



Monthly Yearly Trends Explanations

The monthly trends to follow the same pattern as the daily yearly trends and weekly yearly trends but is smoother. There seems to be a lot of bike trips in November and significant dip in trips in April.

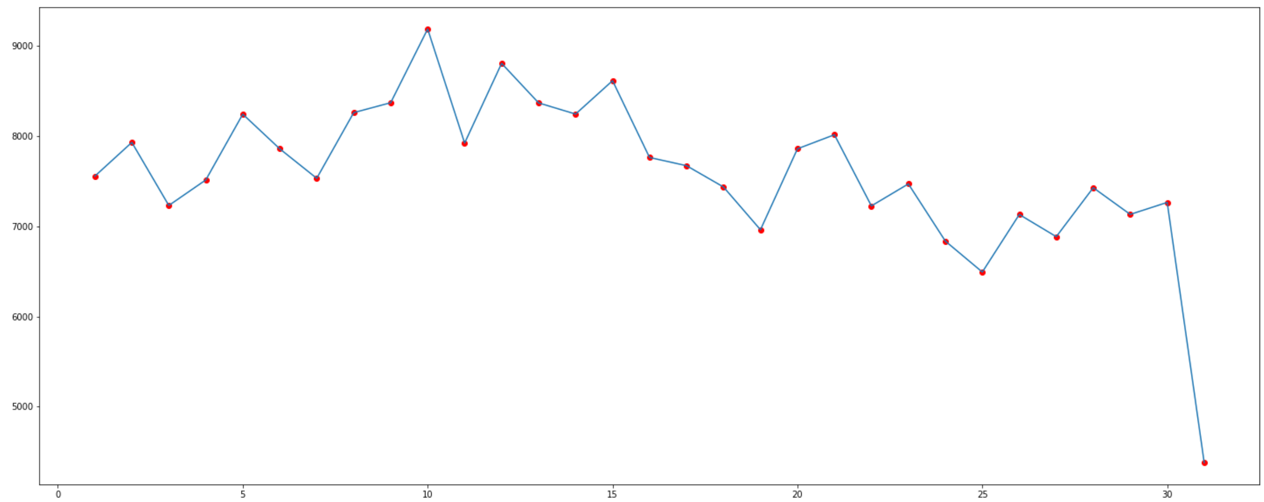
Monthly Trends

```
month2 = (historic_trip_data_df.withColumn("month", dayofmonth("started_at")))
month3 = (month2.select("month", "rideable_type"))
month4 = month2.groupBy(month3.month).count().orderBy(month2.month)
month4.show(31)
```

```
+-----+-----+
|month|count|
+-----+-----+
|    1| 7555|
|    2| 7929|
|    3| 7230|
|    4| 7513|
|    5| 8242|
|    6| 7858|
|    7| 7531|
|    8| 8259|
|    9| 8369|
|   10| 9186|
|   11| 7922|
|   12| 8805|
|   13| 8367|
|   14| 8243|
|   15| 8613|
|   16| 7762|
|   17| 7671|
|   18| 7435|
```

```
import matplotlib.pyplot as plt
counts=month4.select('count').toPandas()
months=month4.select('month').toPandas()
plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(months,counts,color='red')
plot = plt.plot(months,counts)

plt.show()
```



Monthly Trends

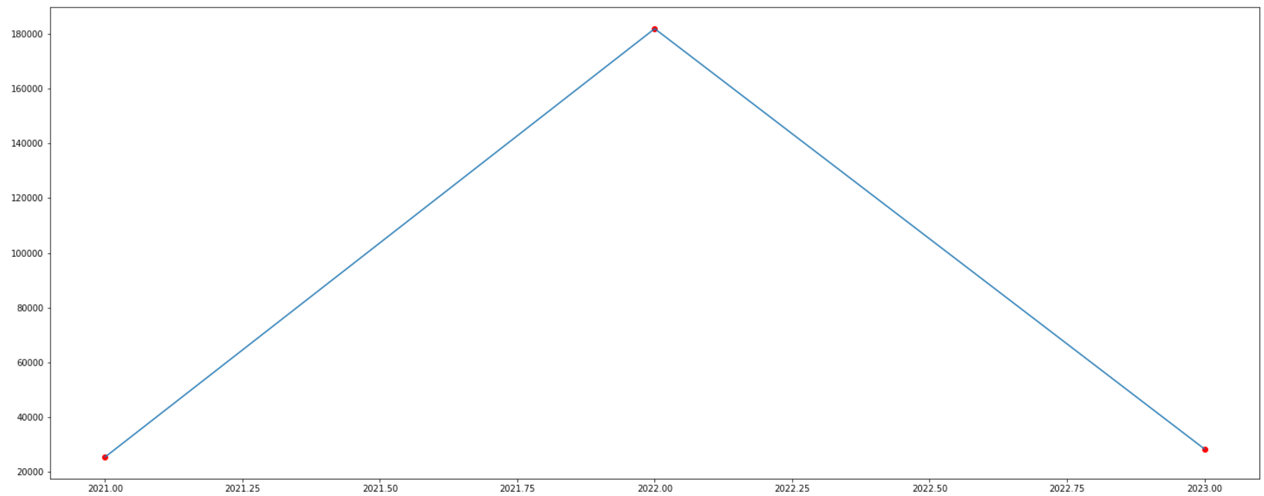
It seems that the most ridden days tend to be at the start-middle of the month, and then drops off at the end of the month. This could be attributed to motivation. Sometimes people want to get very fit at the start of a month and then lose motivation towards the end of the month. Also not every month has 31 days, which explains the very sudden drop at the end.

```
year2 = (historic_trip_data_df.withColumn("year", year("started_at")))
year3 = (year2.select("year", "rideable_type"))
year4 = year2.groupBy(year3.year).count().orderBy(year2.year)
year4.show()
```

```
+----+------+
|year| count|
+----+------+
|2021| 25277|
|2022|182020|
|2023| 28263|
+----+------+
```

```
import matplotlib.pyplot as plt
counts=year4.select('count').toPandas()
years=year4.select('year').toPandas()
plt.figure(figsize=(25, 10))
scatterplt = plt.scatter(years,counts,color='red')
plot = plt.plot(years,counts)

plt.show()
```



Yearly Trends Explanations

The year trends tell us that there were a lot more trips in 2022 than in 2021 or 2023.

2021 was still pretty heavily impacted by covid regualtions so people were probably less willing to go on bikes that are shared by a lot of people.

2022 probably has the most trips in it, because in 2022 pretty much all covid restrictions were lifted and people were more motivated to go outside more and ride more bikes.

2023 has less trips in it, because the year is still not done.

Weather Trends

```
weather_trips = (spark.read.format("delta").load("dbfs:/FileStore/tables/G11/silver/inventory/"))
weather_trips.display()
```

Table

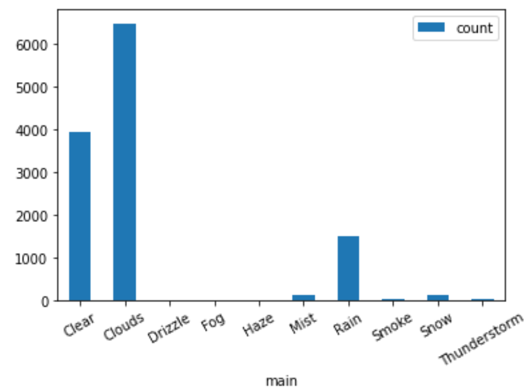
	dt	temp	feels_like	snow_1h	main	rain_1h	net_hour_change
1	2021-11-19 21:00:00	7.45	3.77	0	Clouds	0	-3
2	2021-11-19 22:00:00	7.63	4.1	0	Clouds	0	0
3	2021-11-19 23:00:00	7.61	4.13	0	Clouds	0	-1
4	2021-11-20 00:00:00	7.35	3.92	0	Clouds	0	-4
5	2021-11-20 01:00:00	6.82	3.37	0	Clouds	0	2
6	2021-11-20 02:00:00	6.08	2.56	0	Clear	0	-3
7	2021-11-20 03:00:00	5.68	2.32	0	Clear	0	-2

10,000 rows | Truncated data

```
from pyspark.sql.functions import *

df4 = (weather_trips.withColumn("hour_change", abs("net_hour_change")))
df5 = (df4.select("main", "hour_change"))


import matplotlib.pyplot as plt
df6 = df4.groupBy(df5.main).count().orderBy(df5.main)
df = df6.select("*").toPandas()
ax = df.plot.bar(x='main', y='count', rot=30)
```



Weather Trends Explanations

Our findings here are pretty interesting. There are pretty much no rides being done under more extreme conditions such as thunderstorms, snow, and smoke. It seems as though most of the activity is being done during times where it is either cloudy or clear skies. Although there is a decrease in activity while it is raining, there is still a fair amount of activity. As you can see on the histogram there are basically no rides during drizzle, fog, or haze. This is most likely due to lack of reporting those weather conditions, it likely happens more often than displayed here.

```
Notebook exited: {"exit_code": "OK"}
```