ITCS-6100 Big Data for Computational Advantage

Group -18

Project Deliverable - 2

Team Members

Manasa Avula - 801307493
Nikhita Sai Boyidapu - 801327682
Srikar Chamarthy - 801317299
Rachana Gullipalli - 801311637
Aravind Pabbisetty - 801274519

Dataset:

The chosen dataset is of a bike sharing company based in chicago called Cyclistic. The dataset consists of information of all the rides taken using different types of bikes by various citizens recently during the period of 12/2021 to 11/2022. The data is stored in CSV files where there is an individual CSV file present for trips taken each month. The dataset consists of attributes such as ride_id, rideable_type, started_at_date, started_at_time, ended_at_date, ended_at_time,time_of_ride, start_station_name, end_station_name ,start_lat, start_lng, end_lat, end_lng, member_casual.

Dataset Source:

https://www.kaggle.com/datasets/jasfre/gcc-cyclistic-case-study-present-report-prompt

AWS Services used:

- **S3:** We have used S3 to create a bucket and store csv files of each month and also stored the combined csv file of all months.

- **Sagemaker:** We have used the jupyter notebook in sagemaker for the data preparation.
- **QuickSight:** We have used quicksight to create a dashboard of the data visualizations.

Data Understanding:

Understanding the Nature of the Data:

The chosen dataset consists of 1200000 records and 19 columns. Some of the Important columns include:

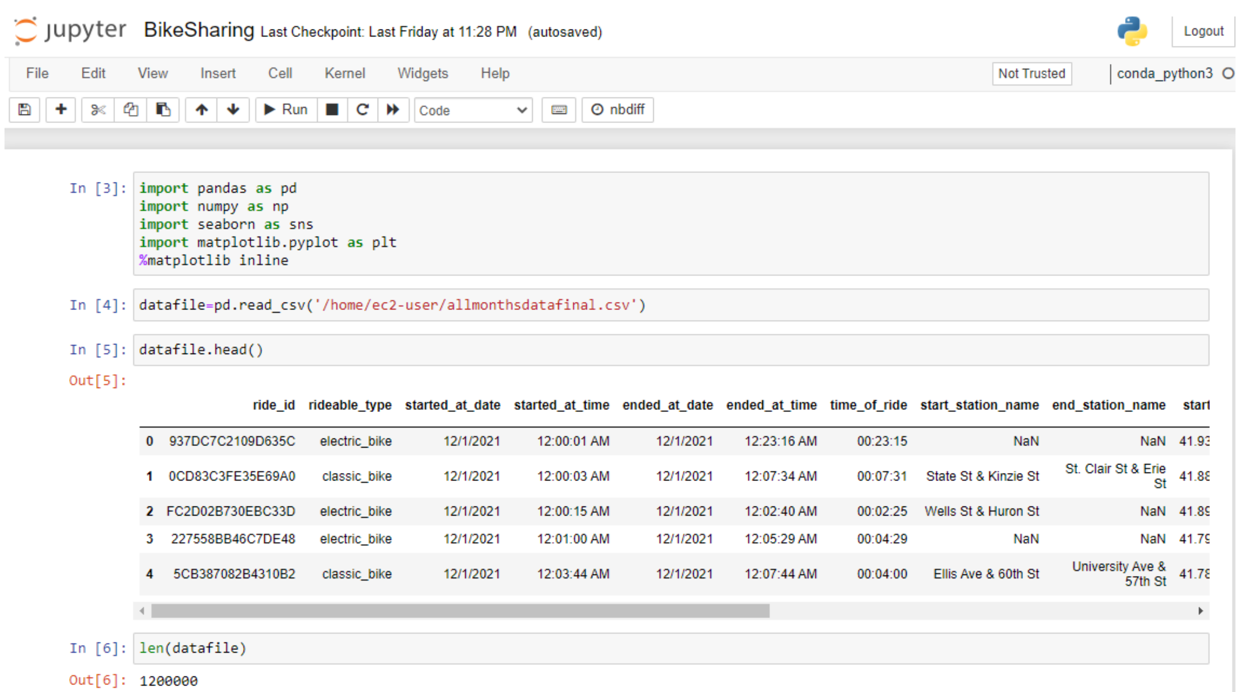
ride_id: It is the unique identifier assigned to each ride taken.

rideable_type: It consists of the type of bike used in the ride.

start_station_name: It consists of the station name where the ride starts.

member_casual: It describes the membership of the customer such as casual_member or member.

There are other columns such as started_at_date, started_at_time, ended_at_date, ended_at_time, time_of_ride, end_station_name, start_lat, start_lng, end_lat, end_lng.


jupyter BikeSharing Last Checkpoint: Last Friday at 11:28 PM (autosaved)

```
In [3]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        %matplotlib inline

In [4]: datafile=pd.read_csv('/home/ec2-user/allmonthsdatafinal.csv')

In [5]: datafile.head()
```

Out[5]:

	ride_id	rideable_type	started_at_date	started_at_time	ended_at_date	ended_at_time	time_of_ride	start_station_name	end_station_name	start
0	937DC7C2109D635C	electric_bike	12/1/2021	12:00:01 AM	12/1/2021	12:23:16 AM	00:23:15	NaN	NaN	41.93
1	0CD83C3FE35E69A0	classic_bike	12/1/2021	12:00:03 AM	12/1/2021	12:07:34 AM	00:07:31	State St & Kinzie St	St. Clair St & Erie St	41.88
2	FC2D02B730EBC33D	electric_bike	12/1/2021	12:00:15 AM	12/1/2021	12:02:40 AM	00:02:25	Wells St & Huron St	NaN	41.89
3	227558BB46C7DE48	electric_bike	12/1/2021	12:01:00 AM	12/1/2021	12:05:29 AM	00:04:29	NaN	NaN	41.79
4	5CB387082B4310B2	classic_bike	12/1/2021	12:03:44 AM	12/1/2021	12:07:44 AM	00:04:00	Ellis Ave & 60th St	University Ave & 57th St	41.78

```
In [6]: len(datafile)
Out[6]: 1200000
```

Fig 1: Reading data from csv file and viewing the first 5 records.

In [7]:
```
datafile.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1200000 entries, 0 to 1199999
Data columns (total 19 columns):
 #   Column              Non-Null Count    Dtype
---  ------              --------------    -----
 0   ride_id             1200000 non-null  object
 1   rideable_type       1200000 non-null  object
 2   started_at_date     1200000 non-null  object
 3   started_at_time     1200000 non-null  object
 4   ended_at_date       1200000 non-null  object
 5   ended_at_time       1200000 non-null  object
 6   time_of_ride        1200000 non-null  object
 7   start_station_name  1035381 non-null  object
 8   end_station_name    1033906 non-null  object
 9   start_lat           1200000 non-null  float64
 10  start_lng           1200000 non-null  float64
 11  end_lat             1199332 non-null  float64
 12  end_lng             1199332 non-null  float64
 13  member_casual       1200000 non-null  object
 14  Unnamed: 14         0 non-null        float64
 15  Unnamed: 15         0 non-null        float64
 16  Unnamed: 16         0 non-null        float64
 17  Unnamed: 17         0 non-null        float64
 18  Unnamed: 18         0 non-null        float64
dtypes: float64(9), object(10)
memory usage: 174.0+ MB
```

In [6]:
```
datafile.describe()
```

Out[6]:

	start_lat	start_lng	end_lat	end_lng	Unnamed: 14	Unnamed: 15	Unnamed: 16	Unnamed: 17	Unnamed: 18
count	1.200000e+06	1.200000e+06	1.199332e+06	1.199332e+06	0.0	0.0	0.0	0.0	0.0
mean	4.189974e+01	-8.764810e+01	4.189989e+01	-8.764824e+01	NaN	NaN	NaN	NaN	NaN
std	4.725555e-02	3.221943e-02	4.722291e-02	2.971153e-02	NaN	NaN	NaN	NaN	NaN
min	4.164000e+01	-8.784000e+01	4.155000e+01	-8.789000e+01	NaN	NaN	NaN	NaN	NaN
25%	4.188000e+01	-8.766357e+01	4.188000e+01	-8.766260e+01	NaN	NaN	NaN	NaN	NaN
50%	4.189691e+01	-8.764421e+01	4.189745e+01	-8.764435e+01	NaN	NaN	NaN	NaN	NaN
75%	4.192889e+01	-8.762963e+01	4.192914e+01	-8.762963e+01	NaN	NaN	NaN	NaN	NaN
max	4.563503e+01	-7.379648e+01	4.212000e+01	-8.730000e+01	NaN	NaN	NaN	NaN	NaN

Fig 2: Information and summary statistics of the data frame

In [7]:
```
datafile.shape
```

Out[7]:
```
(1200000, 19)
```

Fig 3: Number of rows and columns in the dataframe

Data Preparation:

```
In [27]: datafile=datafile.drop('Unnamed: 14',axis=1)
         datafile=datafile.drop('Unnamed: 15',axis=1)
         datafile=datafile.drop('Unnamed: 16',axis=1)
         datafile=datafile.drop('Unnamed: 17',axis=1)
         datafile=datafile.drop('Unnamed: 18',axis=1)

In [30]: print(datafile.isnull().sum())

         ride_id                    0
         rideable_type              0
         started_at_date            0
         started_at_time            0
         ended_at_date              0
         ended_at_time              0
         time_of_ride               0
         start_station_name    164619
         end_station_name      166094
         start_lat                  0
         start_lng                  0
         end_lat                  668
         end_lng                  668
         member_casual              0
         dtype: int64
```

Fig 4: Dropping unnamed columns and checking columns having null values

```
In [31]: # Replacing NA values with mode value of its respective column
         datafile['start_station_name'] = datafile['start_station_name'].fillna(datafile['start_station_name'].mode()[0])
         datafile['end_station_name'] = datafile['end_station_name'].fillna(datafile['end_station_name'].mode()[0])
         datafile['end_lat'] = datafile['end_lat'].fillna(datafile['end_lat'].mode()[0])
         datafile['end_lng'] = datafile['end_lng'].fillna(datafile['end_lng'].mode()[0])

In [32]: #Checking if all NA values are filled:
         print(datafile.isnull().sum())

         ride_id               0
         rideable_type         0
         started_at_date       0
         started_at_time       0
         ended_at_date         0
         ended_at_time         0
         time_of_ride          0
         start_station_name    0
         end_station_name      0
         start_lat             0
         start_lng             0
         end_lat               0
         end_lng               0
         member_casual         0
         dtype: int64
```

Fig 5: Replacing null values with mode value of respective columns and checking is still null values exists

```
In [12]: datafile = datafile.drop_duplicates()

In [33]: len(datafile)

Out[33]: 1200000
```

Fig 6: Dropping the Duplicate rows

In [46]:
```
# Creating new column 'rideable_type_value' which stores the categorical variable(rideable_type) as a numeric value
datafile['rideable_type_value'] = datafile['rideable_type'].replace({'classic_bike': 0,'electric_bike': 1, 'docked_bike': 2})
datafile[['rideable_type','rideable_type_value']].head()
```

Out[46]:

	rideable_type	rideable_type_value
0	electric_bike	1
1	classic_bike	0
2	electric_bike	1
3	electric_bike	1
4	classic_bike	0

Fig 7: Adding a new column rideable_type_value that stores the categorical variable (rideable_type) as a numeric value.

In [70]:
```
# creating a new column 'weekday' which can used later for exploratory data analysis
datafile['weekday'] = pd.to_datetime(datafile['started_at_date']).dt.day_name()
datafile[['started_at_date','weekday']].head()
```

Out[70]:

	started_at_date	weekday
0	12/1/2021	Wednesday
1	12/1/2021	Wednesday
2	12/1/2021	Wednesday
3	12/1/2021	Wednesday
4	12/1/2021	Wednesday

Fig 8: Adding a new column weekday for the started date

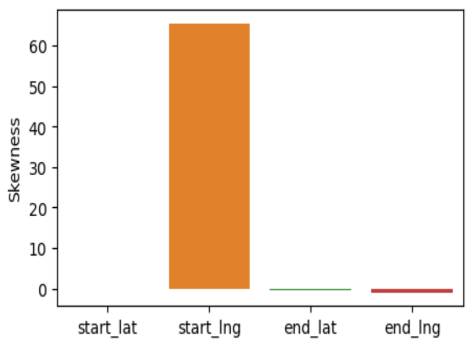
Exploratory Data Analysis:

1. Skew Numeric columns

In [36]:
```
skew=datafile.skew(numeric_only=True)
skew
```

Out[36]:
```
start_lat    -0.026866
start_lng    65.478280
end_lat      -0.449547
end_lng      -0.955464
dtype: float64
```

In [40]:
```
skew_df=pd.DataFrame(skew,index=None,columns=['Skewness'])
plt.figure(figsize=(5,3))
sns.barplot(x=skew_df.index,y='Skewness',data=skew_df)
plt.show()
```



2. Word Cloud to get the Destination Stations with most rides.

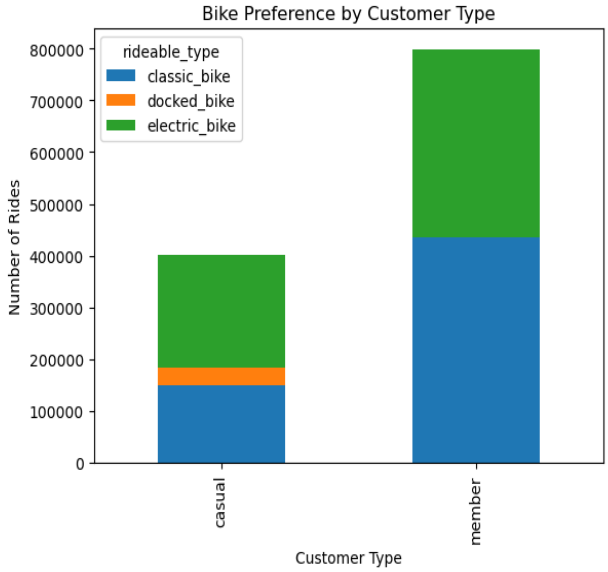
In [64]:
```
from wordcloud import WordCloud as wd
end_station_data = datafile["end_station_name"].value_counts()
wordcloud = wd(width=300,height=100,background_color="white").generate_from_frequencies(end_station_data)
plt.figure(figsize=(8,8))
plt.imshow(wordcloud)
plt.axis("off")
```

Out[64]: (-0.5, 299.5, 99.5, -0.5)

3. Bike type is preferred by different customers

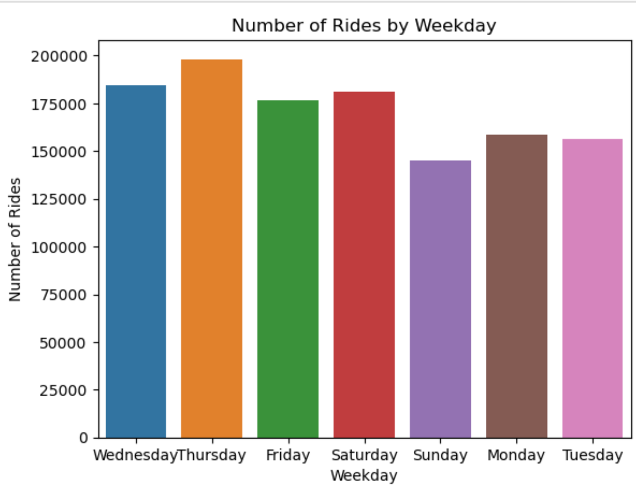
In [65]:
```
# Group the data by member/casual and bike type, then count the number of occurrences
bike_preference = datafile.groupby(['member_casual', 'rideable_type']).size().unstack()

bike_preference.plot(kind='bar', stacked=True)
plt.xlabel('Customer Type')
plt.ylabel('Number of Rides')
plt.title('Bike Preference by Customer Type')
plt.show()
```



4. Rides per each day of the week

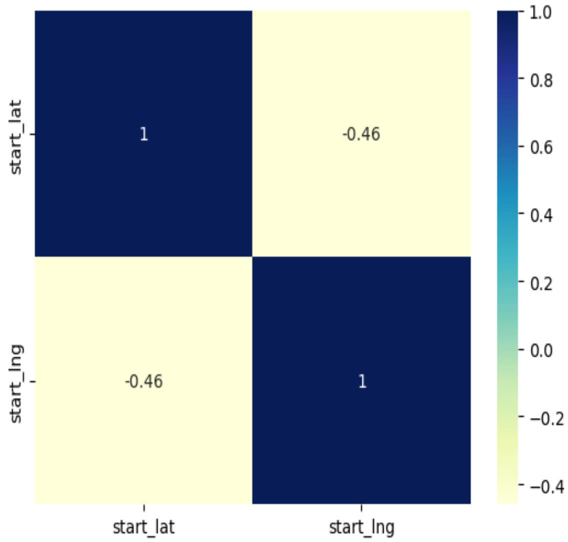
In [67]:
```
sns.countplot(x='weekday', data=datafile)
plt.title('Number of Rides by Weekday')
plt.xlabel('Weekday')
plt.ylabel('Number of Rides')
plt.show()
```



5. Correlation Between ride start latitude and longitude

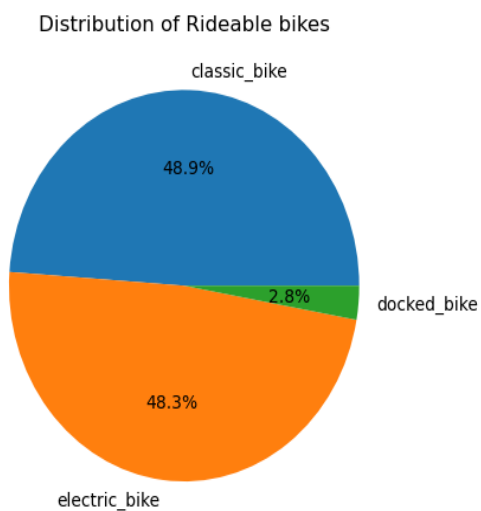
```
In [75]: corr_matrix = datafile[['start_lat', 'start_lng']].corr()
         sns.heatmap(corr_matrix, annot=True, cmap='YlGnBu')

Out[75]: <AxesSubplot: >
```



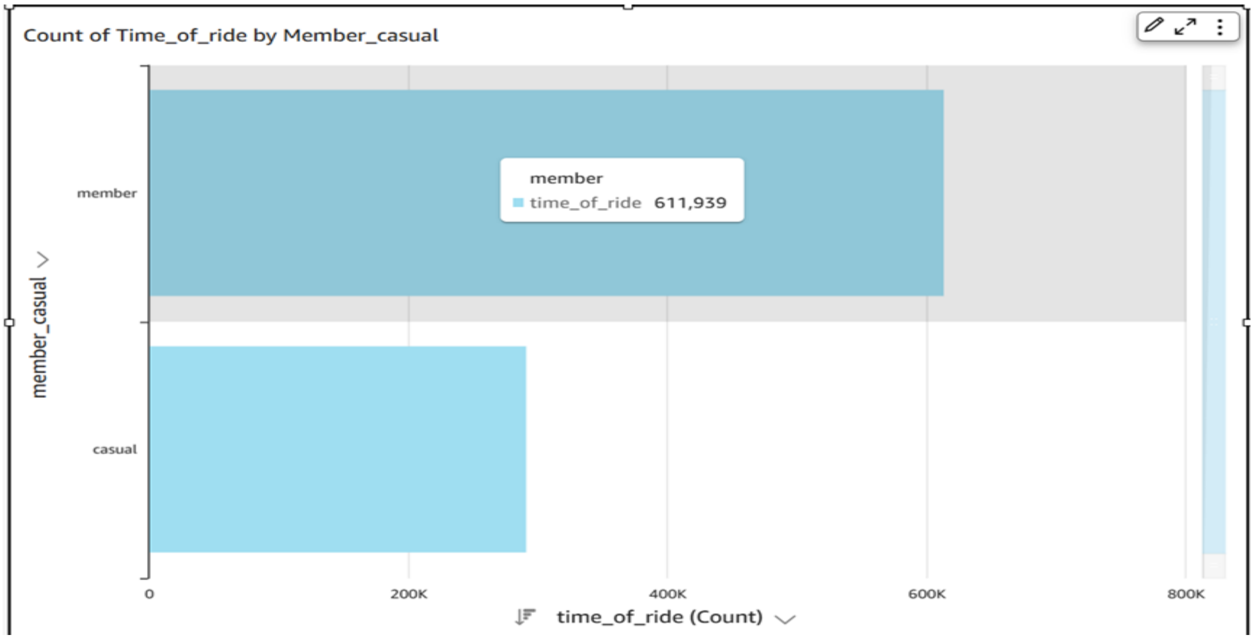
6. Pie Chart representing Distribution of Rideable Bikes

```
In [79]: ride_counts = datafile['rideable_type'].value_counts()
         plt.pie(ride_counts, labels=ride_counts.index, autopct='%1.1f%%')
         plt.title("Distribution of Rideable bikes")
         plt.show()
```

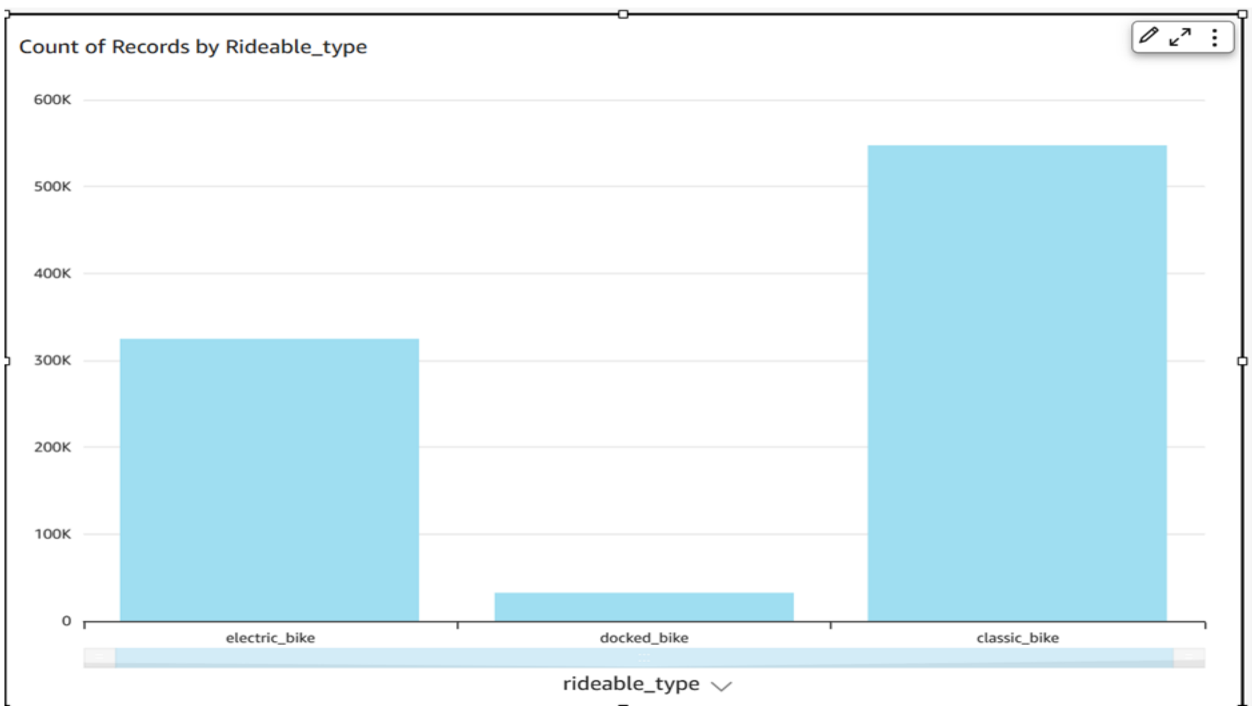


DashBoard:

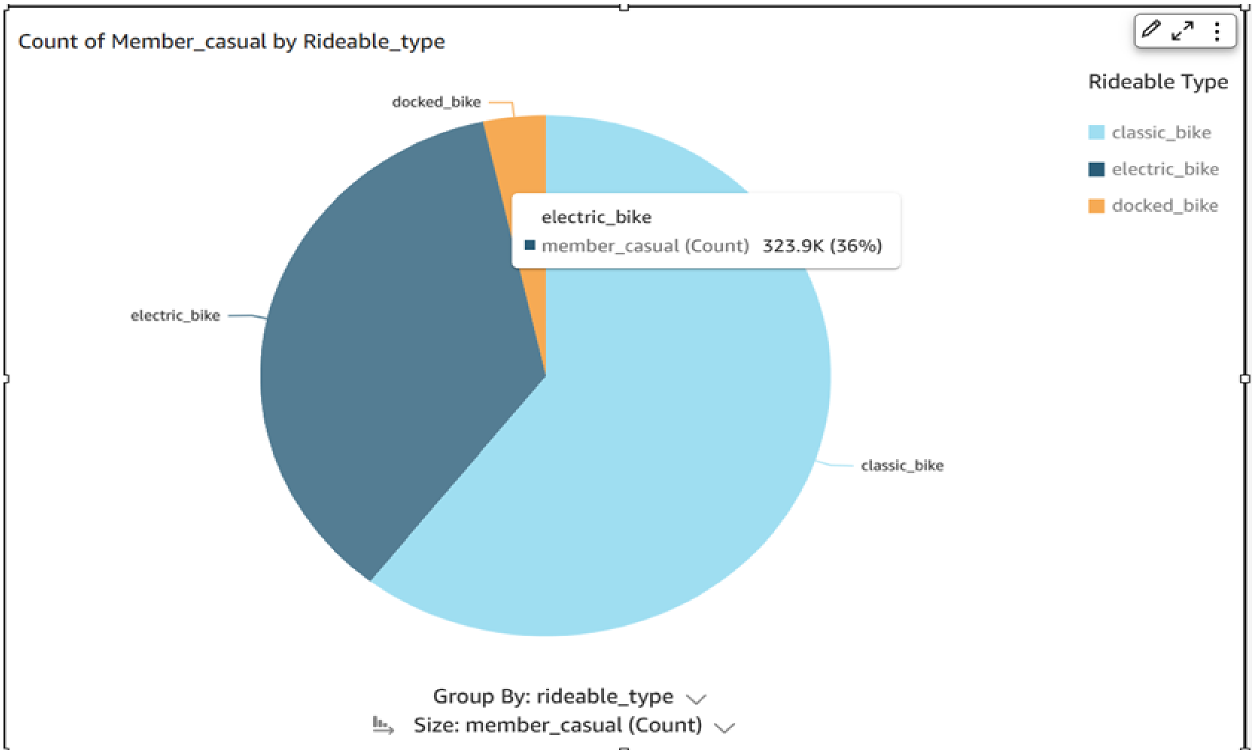
1. Which customer type is using the bike for a longer period of time?



2. Which type of bike is rented most by customers?



3. Count of different types of members using different types of rides



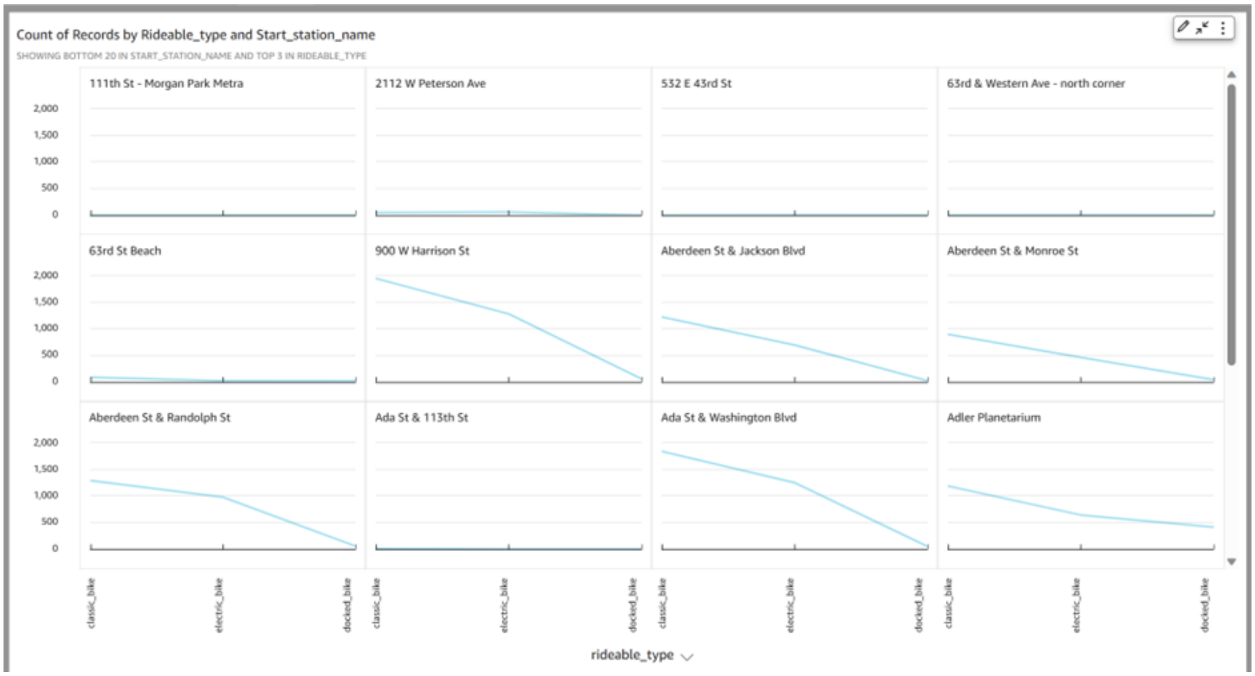
4. Which type of bike is used for a longer period of time by the customers per each rent?


Count of Records by Time_of_ride, Rideable_type, and Member_casual

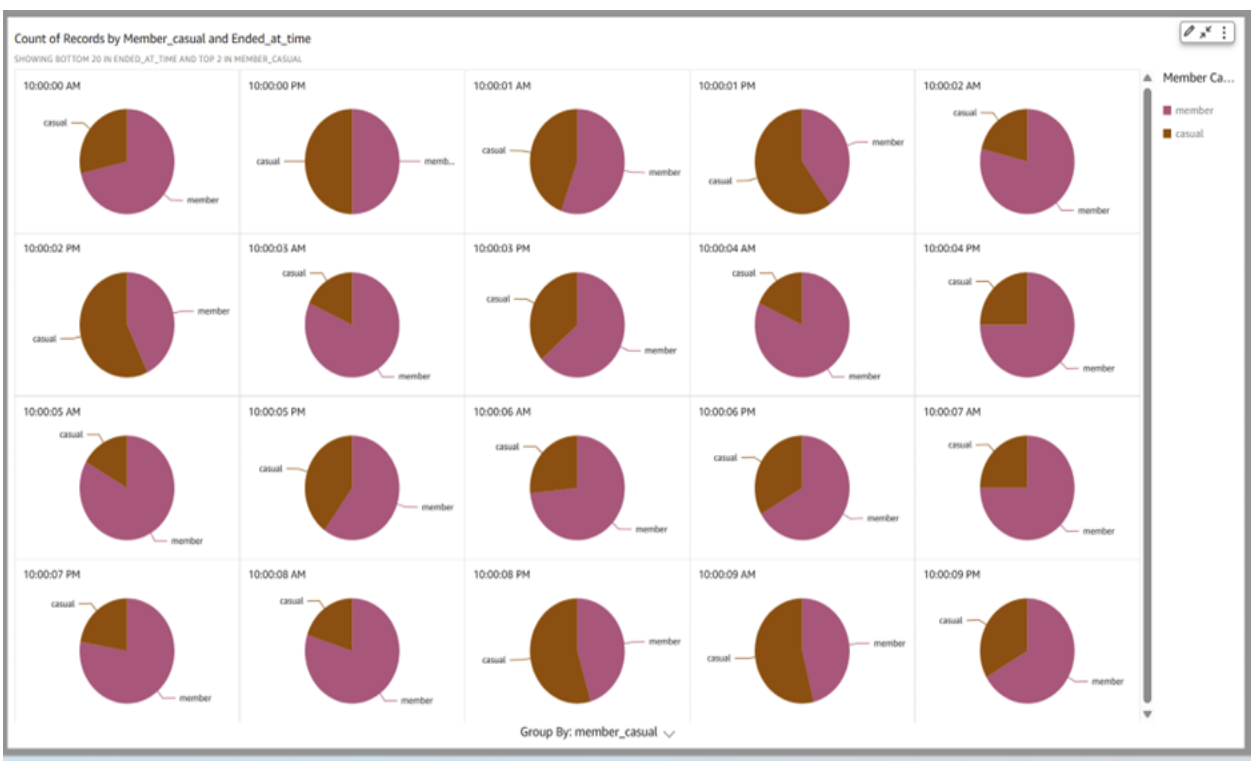
rideable_type > member_casual

	classic_bike		docked...
	member	casual	casual
time_of_ride	Count	Count	Count
23:55:36			1
23:55:28	1		
23:55:22			1
23:45:52			1
23:44:07			1
23:41:12	1		
23:40:09			1
23:38:25			1
23:37:03	1		
23:32:46		1	
23:30:56			1
23:28:29		1	
23:25:38	1		
23:18:09			1

5. Which type of bike is most preferable in each station?



6. What is the ratio of customers to return the bike classified by each hour?



7. Start time of ride and end time of ride and total time of ride

Started_at_time, Ended_at_time, and Time_of_ride

started_at_time	ended_at_time	time_of_ride
11:10:56 AM	11:18:47 AM	00:07:51
11:57:09 AM	12:02:18 PM	00:05:09
1:23:51 PM	1:28:41 PM	00:04:50
1:38:00 PM	1:41:39 PM	00:03:39
3:16:37 PM	3:31:05 PM	00:14:28
3:53:57 PM	4:02:03 PM	00:08:06
4:15:49 PM	4:18:12 PM	00:02:23
4:23:31 PM	4:32:12 PM	00:08:41
4:27:11 PM	4:35:09 PM	00:07:58
4:41:28 PM	5:00:10 PM	00:18:42
4:51:23 PM	4:58:33 PM	00:07:10
5:04:59 PM	5:10:39 PM	00:05:40
5:17:17 PM	5:22:15 PM	00:04:58
5:32:29 PM	5:39:12 PM	00:06:43
5:33:56 PM	5:38:43 PM	00:04:47
6:26:56 PM	6:34:42 PM	00:07:46

8. Different rides starting at different stations

Start_station_name and Rideable_type

start_station_name	rideable_type
Yates Blvd & Exchange Ave	electric_bike
Yates Blvd & 93rd St	classic_bike
Yates Blvd & 93rd St	docked_bike
Yates Blvd & 93rd St	electric_bike
Yates Blvd & 75th St	classic_bike
Yates Blvd & 75th St	docked_bike
Yates Blvd & 75th St	electric_bike
Woodlawn Ave & Lake Park Ave	classic_bike
Woodlawn Ave & Lake Park Ave	docked_bike
Woodlawn Ave & Lake Park Ave	electric_bike
Woodlawn Ave & 75th St	classic_bike
Woodlawn Ave & 75th St	electric_bike
Woodlawn Ave & 55th St	classic_bike
Woodlawn Ave & 55th St	docked_bike
Woodlawn Ave & 55th St	electric_bike
Woodlawn & 103rd - Olive Harvey Vaccination Site	electric_bike
Wood St & Webster Ave	classic_bike
Wood St & Webster Ave	docked_bike
Wood St & Webster Ave	electric_bike
Wood St & Taylor St (Temp)	classic_bike
Wood St & Taylor St (Temp)	docked_bike
Wood St & Taylor St (Temp)	electric_bike
Wood St & Milwaukee Ave	classic_bike
Wood St & Milwaukee Ave	docked_bike
Wood St & Milwaukee Ave	electric_bike
Wood St & Hubbard St	classic_bike

View: 500 items 1 of 6

9. Which customer type is most seen classified based on the start and end stations?

Sheet 1 | Sheet 2 | Sheet 3 | Sheet 4 | Sheet 5 | Sheet 6 | Sheet 7 | Sheet 8 | Sheet 9 | Sheet 10 | Sheet 11 | Sheet 12 | Sheet 13 | Sheet 14 | **Sheet 15**

Start_station_name, End_station_name, and Member_casual

start_station_name	end_station_name	member_casual
Yates Blvd & Exchange Ave	Harper Ave & 59th St	casual
Yates Blvd & Exchange Ave	Yates Blvd & Exchange Ave	casual
Yates Blvd & 93rd St	Clyde Ave & 87th St	member
Yates Blvd & 93rd St	Constance Ave & 95th St	casual
Yates Blvd & 93rd St	Major Taylor Trail & 115th St	casual
Yates Blvd & 93rd St	Marquette Ave & 89th St	member
Yates Blvd & 93rd St	Public Rack - Saginaw Ave & 93rd St	member
Yates Blvd & 93rd St	Stony Island Ave & South Chicago Ave	member
Yates Blvd & 93rd St	Yates Blvd & 93rd St	casual
Yates Blvd & 93rd St	Yates Blvd & 93rd St	member
Yates Blvd & 75th St	63rd St Beach	member
Yates Blvd & 75th St	Bennett Ave & 79th St	casual
Yates Blvd & 75th St	Clyde Ave & 87th St	casual
Yates Blvd & 75th St	Cornell Ave & Hyde Park Blvd	casual
Yates Blvd & 75th St	East End Ave & 87th St	casual
Yates Blvd & 75th St	Exchange Ave & 79th St	member
Yates Blvd & 75th St	Jeffery Blvd & 67th St	member
Yates Blvd & 75th St	Jeffery Blvd & 71st St	member
Yates Blvd & 75th St	Jeffery Blvd & 76th St	member
Yates Blvd & 75th St	Kimbark Ave & 53rd St	casual
Yates Blvd & 75th St	Museum of Science and Industry	casual
Yates Blvd & 75th St	Phillips Ave & 79th St	casual
Yates Blvd & 75th St	Phillips Ave & 79th St	member
Yates Blvd & 75th St	Phillips Ave & 83rd St	member

10. What is the most popular bike among casual and members respectively?