أكاديمية سدايا
SDAIA Academy

MTA Exploratory Data Analysis: Subway Gathering Limitation Recommendations

Mohammed Osailan

Outlines

DATABASE

DATA CLEANING

EXPLORATORY DATA ANALYSIS

CONCLUSIONS

Database

- First download the data:

```
def download_MTA_data(start_ = date(2014, 10, 25), end_= datetime.date(datetime.now())):
    df_url = pd.DataFrame()
    for i in pd.date_range(start=start_, end=end_, freq='7D'):
        ii = str(i)
        formated_date = ii[2:4] + ii[5:7] + ii[8:10]
        df_url = df_url.append(pd.read_csv(
            "http://web.mta.info/developers/data/nyct/turnstile/turnstile_"+formated_date+".txt"))
    return df_url
```

```
start_date = date(2020, 1, 4)
end_date = date(2020, 4, 4)
df = download_MTA_data(start_date, end_date)
```

```
file_name = 'C:/Users/Windows10/SDAIA_Bootcamp/NBM_EDA_Gamma/EDA_MVP_Moh_Os.csv'
df.to_csv(file_name, index=False)
```

Database

- Two PostgreSQL databases were created.
- Cloud:

```
C:\Users\Windows10>psql --host=database-eda.czjmtbvofhbk.us-east-2.rds.amazonaws.com --port=5432 --username=postgres_ --
password --dbname=eda_database_os
Password:
psql (13.4, server 12.5)
eda_database_os=> \COPY mta FROM 'C:\Users\Windows10\Desktop\EDA_MVP_Moh_Os.csv' with (FORMAT csv);
```

Database

- Local:

```
C:\Users\Windows10>cd "C:\Users\Windows10\SDAIA_Bootcamp\NBM_EDA_Gamma\sqlite-tools-win32-x86-3360000"

C:\Users\Windows10\SDAIA_Bootcamp\NBM_EDA_Gamma\sqlite-tools-win32-x86-3360000>sqlite3 eda_1.db
SQLite version 3.36.0 2021-06-18 18:36:39
Enter ".help" for usage hints.
sqlite> CREATE TABLE "table_df" (
   ...> "C/A"TEXT,
   ...> "UNIT"TEXT,
   ...> "SCP"TEXT,
   ...> "STATION"TEXT,
   ...> "LINENAME"TEXT,
   ...> "DIVISION"TEXT,
   ...> "DATE"TEXT,
   ...> "TIME"TEXT,
   ...> "DESC"TEXT,
   ...> "ENTRIES"INTEGER,
   ...> "EXITS"INTEGER
   ...> );
```

```
sqlite> .mode csv
sqlite> .import EDA_MVP_Moh_Os.csv
```

Database

- Connecting to the database in python:
- Cloud:

```
engine = create_engine('postgresql://postgres_:#mohammed@database-eda.czjmtbvofhbk.us-east-2.rds.amazonaws.com:5432/eda_database_os', echo=True)
```

- Read only access user:

```
engine = create_engine('postgresql://guest:1234@database-eda.czjmtbvofhbk.us-east-2.rds.amazonaws.com:5432/eda_database_os', echo=True)
```

```
cloud_database_data = pd.read_sql('SELECT * FROM mta;', engine)
```

Database

- Connecting to the database in python:
- Local:

```
engine = create_engine("sqlite:///eda_1.db", echo=True)

local_database_data = pd.read_sql('SELECT * FROM mta;', engine)
```

Data Cleaning

- Columns names.

```
# Let us first clear the columns names so that we can use it.
print(df.columns)
# Here we clean columns names and remove all unnecessary spaces.
df.columns = [column.strip() for column in df.columns]
print(df.columns)
```

```
Index(['C/A', 'UNIT', 'SCP', 'STATION', 'LINENAME', 'DIVISION', 'DATE', 'TIME',
       'DESC', 'ENTRIES',
       'EXITS                                                               '],
      dtype='object')
Index(['C/A', 'UNIT', 'SCP', 'STATION', 'LINENAME', 'DIVISION', 'DATE', 'TIME',
       'DESC', 'ENTRIES', 'EXITS'],
      dtype='object')
```

Data Cleaning

- Dates type conversion.

```
3  df['DATE_TIME'] = pd.to_datetime(df['DATE'] + ' ' + df['TIME'])
4  df['DATE'] = pd.to_datetime(df['DATE'])

1  print(df[['DATE_TIME']].dtypes)
2  print(df[['DATE']].dtypes)
```

```
DATE_TIME    datetime64[ns]
dtype: object
DATE    datetime64[ns]
dtype: object
```

Data Cleaning

- Separating turnstiles.

```
turnstile_df['TURNSTILE_ID'] = turnstile_df["C/A"] + turnstile_df["UNIT"] + turnstile_df["SCP"] + turnstile_df["STATION"]
turnstile_df['ENTRIES_PFH’] = np.where(turnstile_df['TURNSTILE_ID'] == turnstile_df["TURNSTILE_ID"].shift(-1),
                                turnstile_df['ENTRIES'].diff().shift(-1) ,
                                0)
turnstile_df['EXITS_PFH’] = np.where(turnstile_df['TURNSTILE_ID'] == turnstile_df["TURNSTILE_ID"].shift(-1),
                               turnstile_df['EXITS'].diff().shift(-1),
                               0)
```

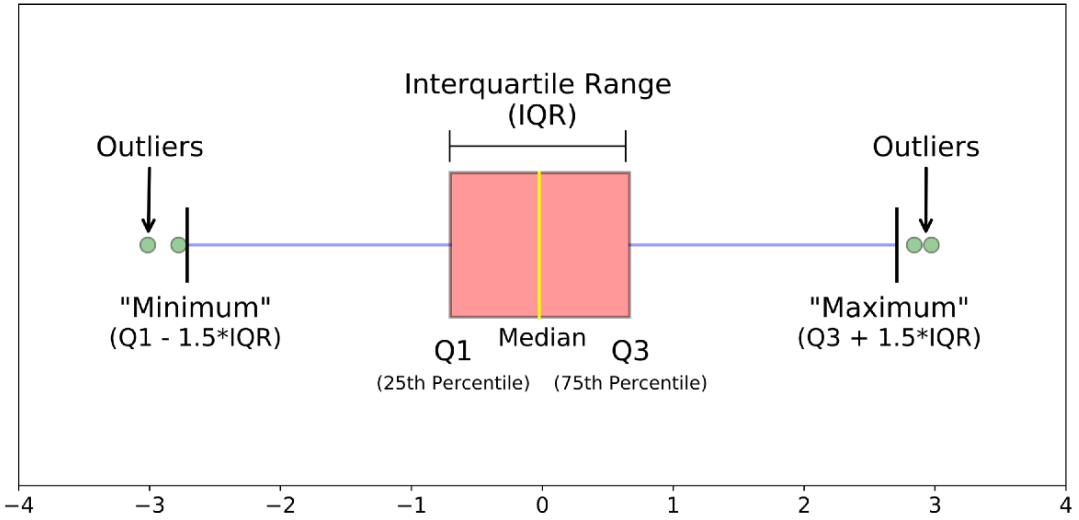
Data Cleaning

- Cleaning negative numbers by taking the absolute value.

```
turnstile_df.loc[turnstile_df.ENTRIES_PFH < 0,'ENTRIES_PFH'] = abs(turnstile_df[turnstile_df.ENTRIES_PFH<0].ENTRIES_PFH)
turnstile_df.loc[turnstile_df.EXITS_PFH < 0,'EXITS_PFH'] = abs (turnstile_df[turnstile_df.EXITS_PFH < 0].EXITS_PFH)
```

Data Cleaning

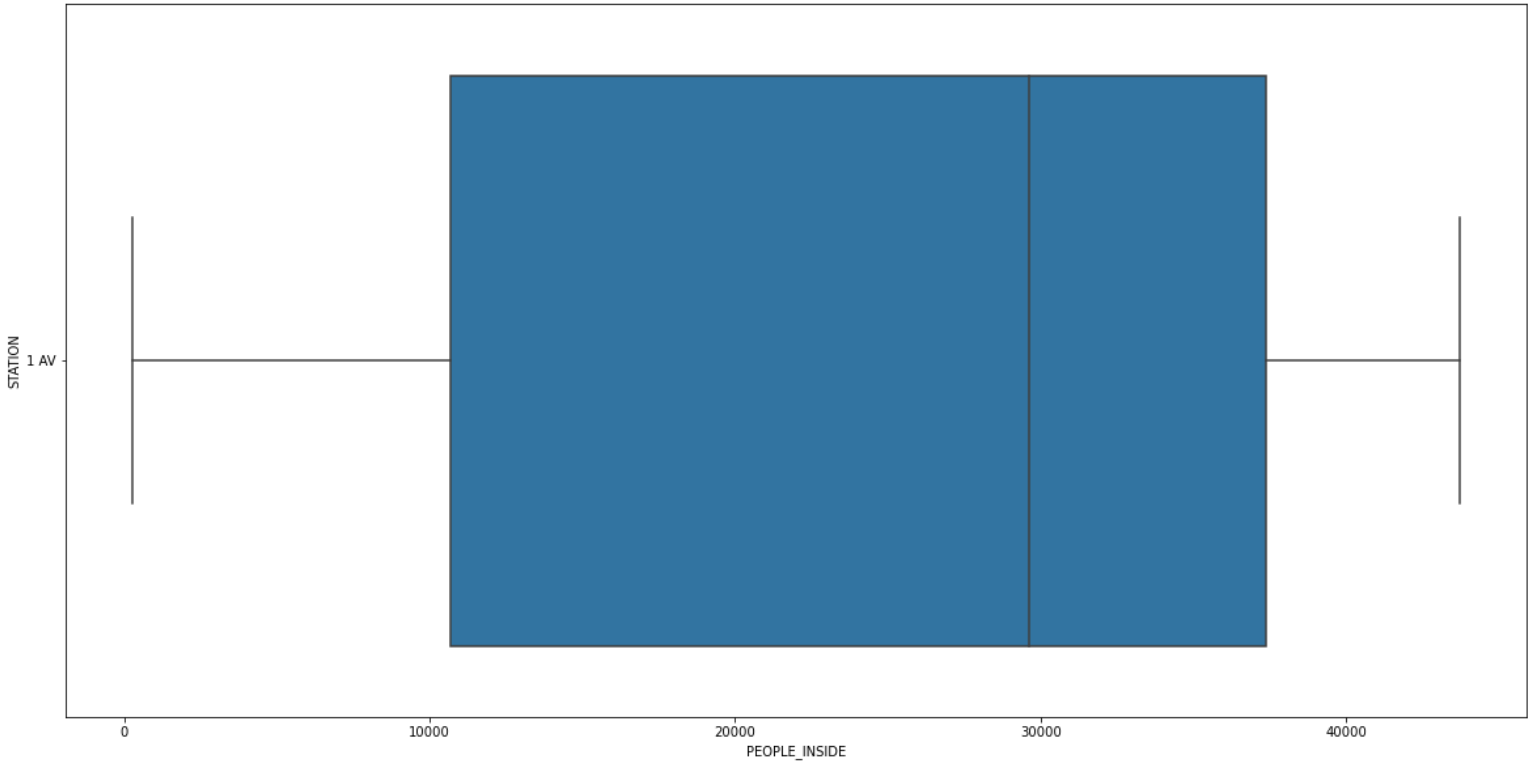
- Dropping outliers.




```
for i in list(turnstile_df['STATION'].unique()):
    x = turnstile_df[turnstile_df['STATION'] == i]['ENTRIES_PFH'].describe()
    max_value = x[6] + 1.5*(x[6] - x[4])
    xx = turnstile_df[turnstile_df['STATION'] == i][turnstile_df[
                                            turnstile_df['STATION'] == i]['ENTRIES_PFH'] > max_value]
    turnstile_df.drop(list(xx.index), inplace=True)
```

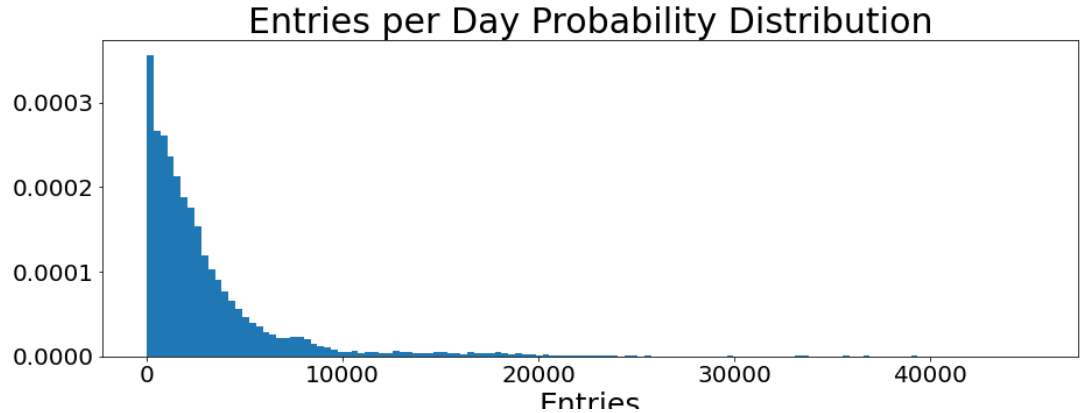
Data Cleaning

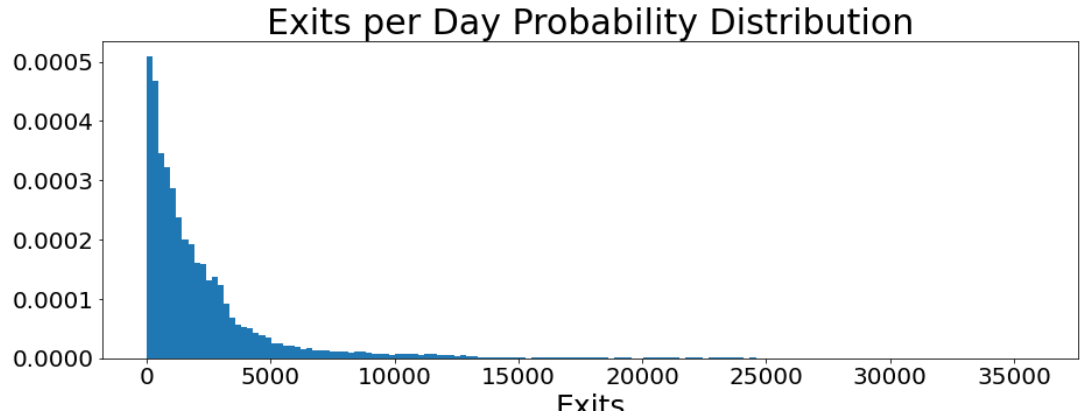
- Station '1 AV'

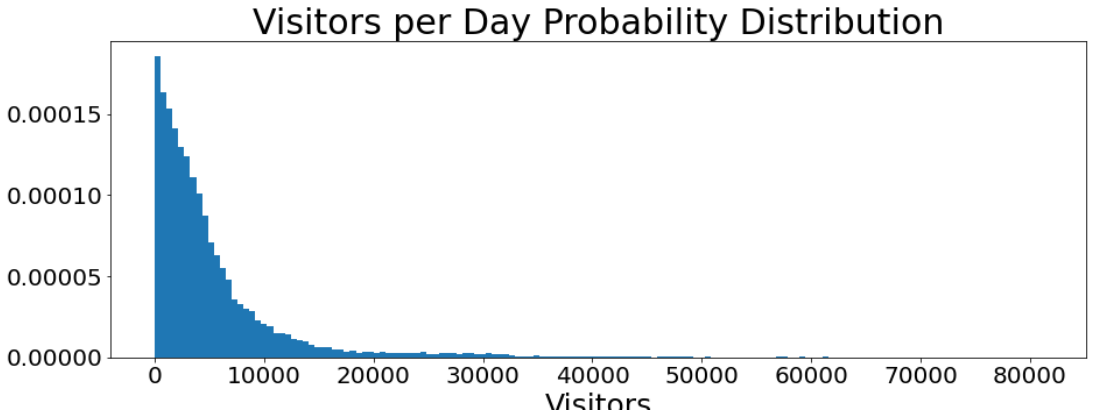


Exploratory Data Analysis

- Probability distribution of daily Exits and Entries data.

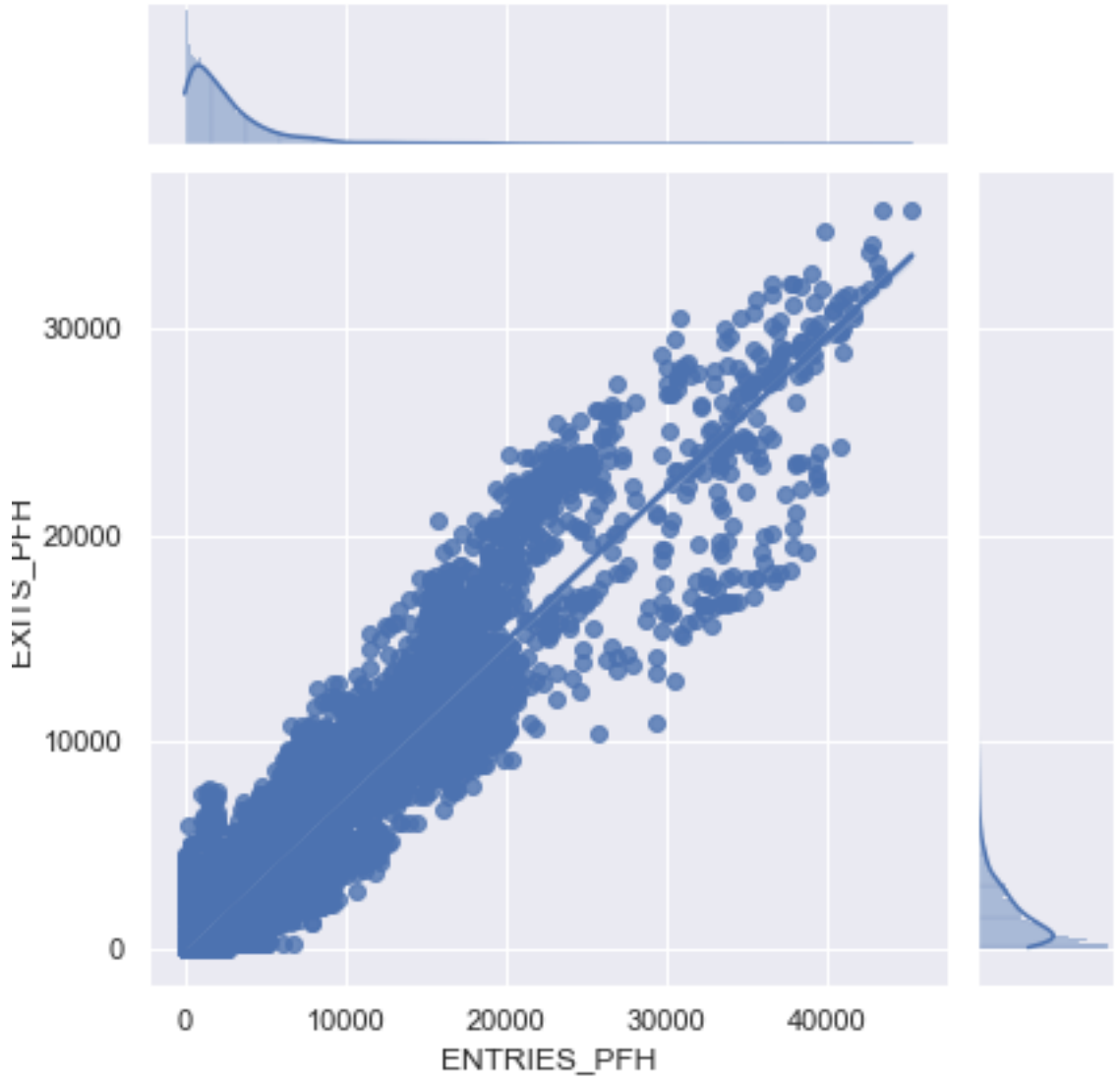






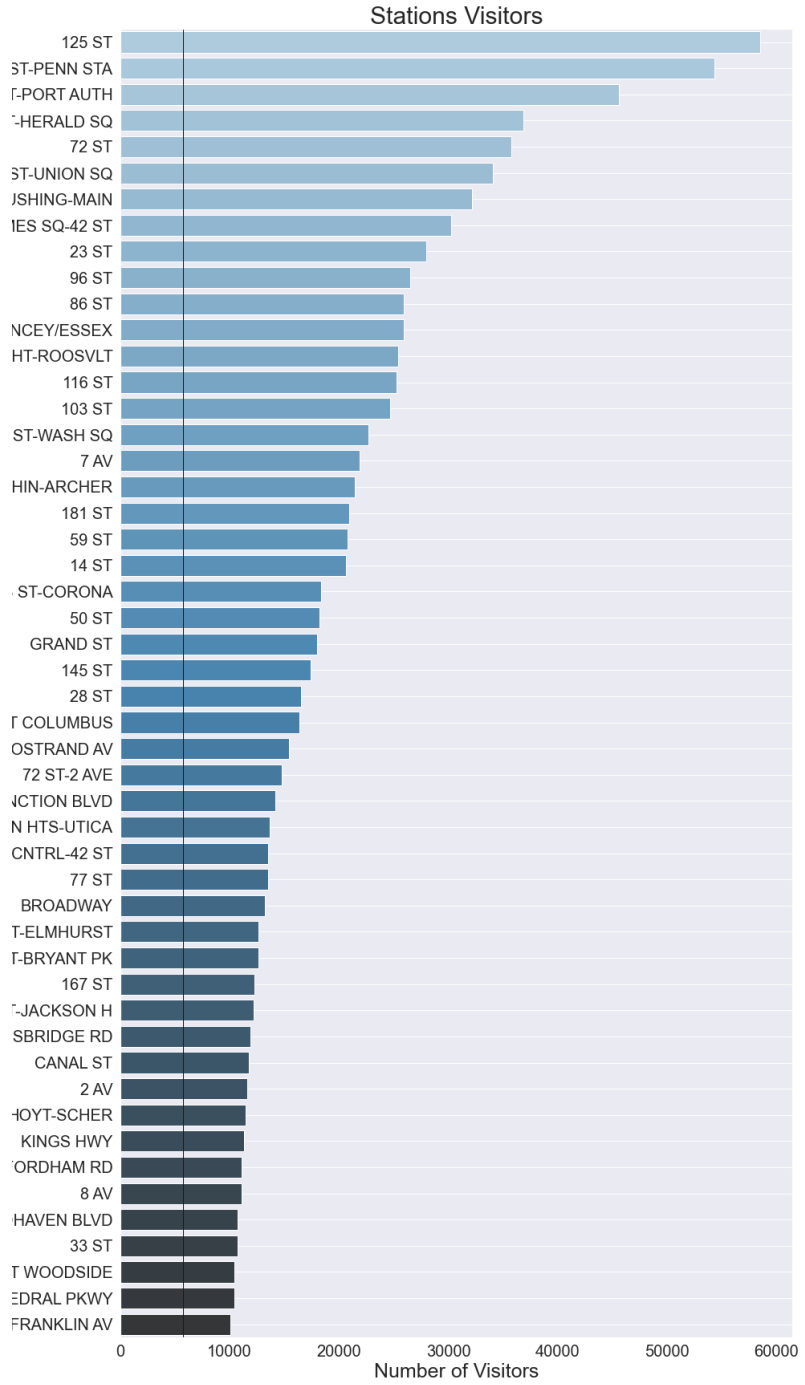
Exploratory Data Analysis

- Joint plot of daily Exits and Entries data.



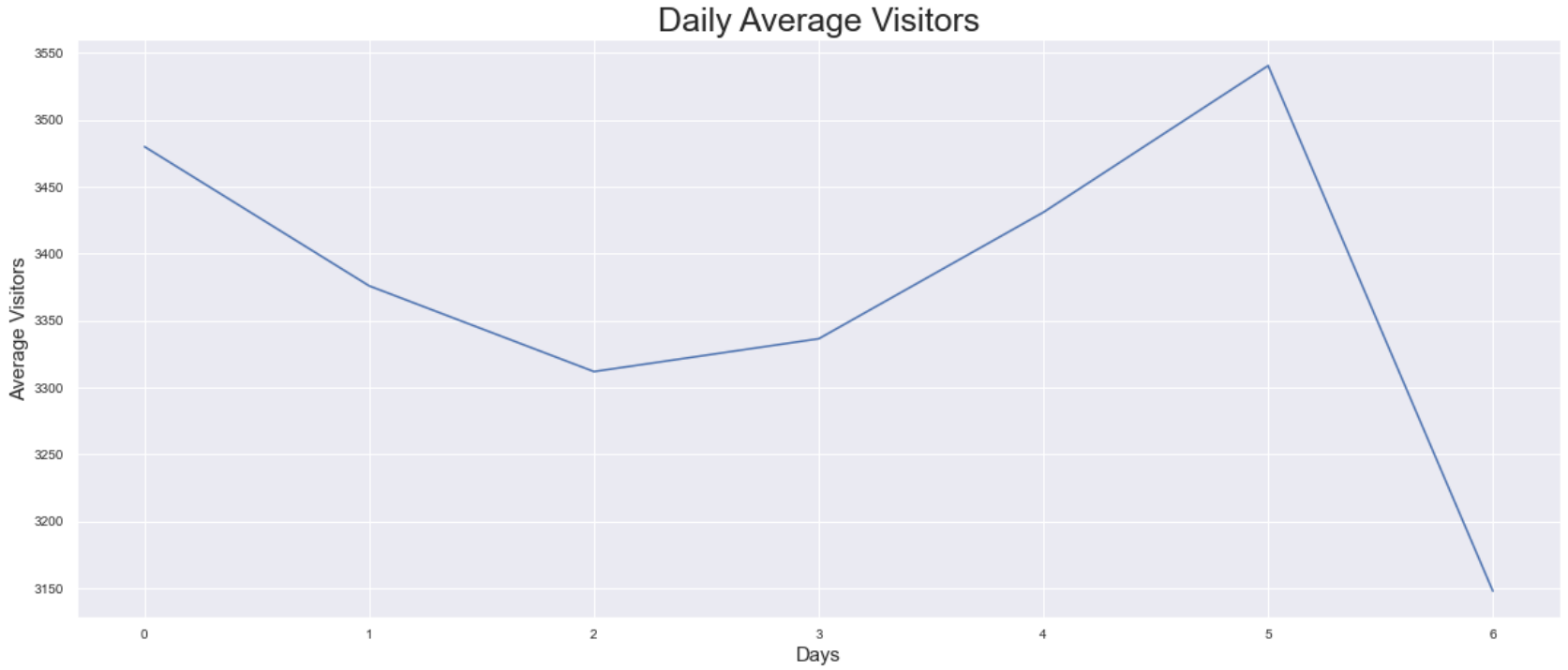
Exploratory Data Analysis

- Most 50 crowded stations.



Exploratory Data Analysis

- Average visitors per day.



Conclusion

- Focus on the 10 most crowded stations.
- Apply a strict gathering limitations regulations.
- Ease the entrance, exits processes.

Thank you