Assignment2

November 28, 2021

Group Members:

_	Names	Student Number
1	Bongumusa Mavuso	1682836
2	Thabo Rachidi	1632496
3	Thobelani Makeleni	1199116
4	Siyabonga Hlomuka	1384685

0.1 Imports:

[2]:
```
import pandas as pd
import seaborn as sns
import numpy as np
sns.set_theme(style="white")

import scipy.cluster.hierarchy as shc
import matplotlib.pyplot as plt
%matplotlib inline
import math
from scipy import stats
import reverse_geocoder as rg
import pprint
import geopandas
import matplotlib.image as mpimg

from pandas.tseries.holiday import USFederalHolidayCalendar as calendar
import warnings
warnings.filterwarnings("ignore")
```

0.2 Reading Data:

[8]:
```
df = pd.read_csv('nyc_taxis.csv', delimiter=',', header=0, index_col=0)
df.head()
```

[8]:
```
           vendor_id      pickup_datetime     dropoff_datetime  \
id
```

```
id2875421          2  2016-03-14 17:24:55  2016-03-14 17:32:30
id2377394          1  2016-06-12 00:43:35  2016-06-12 00:54:38
id3858529          2  2016-01-19 11:35:24  2016-01-19 12:10:48
id3504673          2  2016-04-06 19:32:31  2016-04-06 19:39:40
id2181028          2  2016-03-26 13:30:55  2016-03-26 13:38:10

           passenger_count  pickup_longitude  pickup_latitude  \
id
id2875421                1        -73.982155        40.767937
id2377394                1        -73.980415        40.738564
id3858529                1        -73.979027        40.763939
id3504673                1        -74.010040        40.719971
id2181028                1        -73.973053        40.793209

           dropoff_longitude  dropoff_latitude store_and_fwd_flag  \
id
id2875421         -73.964630         40.765602                  N
id2377394         -73.999481         40.731152                  N
id3858529         -74.005333         40.710087                  N
id3504673         -74.012268         40.706718                  N
id2181028         -73.972923         40.782520                  N

           trip_duration
id
id2875421            455
id2377394            663
id3858529           2124
id3504673            429
id2181028            435
```

[9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1458644 entries, id2875421 to id1209952
Data columns (total 10 columns):
 #   Column              Non-Null Count    Dtype
---  ------              --------------    -----
 0   vendor_id           1458644 non-null  int64
 1   pickup_datetime     1458644 non-null  object
 2   dropoff_datetime    1458644 non-null  object
 3   passenger_count     1458644 non-null  int64
 4   pickup_longitude    1458644 non-null  float64
 5   pickup_latitude     1458644 non-null  float64
 6   dropoff_longitude   1458644 non-null  float64
 7   dropoff_latitude    1458644 non-null  float64
 8   store_and_fwd_flag  1458644 non-null  object
 9   trip_duration       1458644 non-null  int64
```

```
dtypes: float64(4), int64(3), object(3)
memory usage: 122.4+ MB
```

0.3 Question 1.2:

[4]:
```
def haversine_dist(lon1, lat1, lon2, lat2):
    """Calculate the great circle distance in kilometers between two points
    on the earth (specified in decimal degrees)

    :params lon1: First point longitude in degrees
    :params lat1: First point latitude in degrees
    :params lon2: Second point longitude in degress
    :params lat2: Second point latitude in degrees

    :returns distance in kilometers
    """
    # Convert decimal degrees to radians
    lon1 = np.deg2rad(lon1)
    lat1 = np.deg2rad(lat1)
    lon2 = np.deg2rad(lon2)
    lat2 = np.deg2rad(lat2)

    # haversine formula
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = np.sin(dlat/2)**2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon/2)**2
    c = 2 * np.arcsin(np.sqrt(a))
    r = 6371 # Radius of earth(km)

    distance = c*r

    return np.round(distance,decimals=3)
```

[5]:
```
df['trip_distance(km)'] = haversine_dist(df['pickup_longitude'],␣
↪df['pickup_latitude'],
             df['dropoff_longitude'], df['dropoff_latitude'])
df.head()
```

[5]:
```
           vendor_id      pickup_datetime     dropoff_datetime  \
id
id2875421          2  2016-03-14 17:24:55  2016-03-14 17:32:30
id2377394          1  2016-06-12 00:43:35  2016-06-12 00:54:38
id3858529          2  2016-01-19 11:35:24  2016-01-19 12:10:48
id3504673          2  2016-04-06 19:32:31  2016-04-06 19:39:40
id2181028          2  2016-03-26 13:30:55  2016-03-26 13:38:10

           passenger_count  pickup_longitude  pickup_latitude  \
```

```
id
id2875421               1        -73.982155        40.767937
id2377394               1        -73.980415        40.738564
id3858529               1        -73.979027        40.763939
id3504673               1        -74.010040        40.719971
id2181028               1        -73.973053        40.793209

           dropoff_longitude  dropoff_latitude store_and_fwd_flag  \
id
id2875421         -73.964630         40.765602                  N
id2377394         -73.999481         40.731152                  N
id3858529         -74.005333         40.710087                  N
id3504673         -74.012268         40.706718                  N
id2181028         -73.972923         40.782520                  N

           trip_duration  trip_distance(km)
id
id2875421            455              1.499
id2377394            663              1.806
id3858529           2124              6.385
id3504673            429              1.485
id2181028            435              1.189
```

0.3.1 Create Average speed column:

[6]:
```
df['avg_speed(km/hr)'] = df['trip_distance(km)']/(df['trip_duration'].
 ↪apply(lambda x: x/3600))
df.head()
```

[6]:
```
           vendor_id      pickup_datetime     dropoff_datetime  \
id
id2875421          2  2016-03-14 17:24:55  2016-03-14 17:32:30
id2377394          1  2016-06-12 00:43:35  2016-06-12 00:54:38
id3858529          2  2016-01-19 11:35:24  2016-01-19 12:10:48
id3504673          2  2016-04-06 19:32:31  2016-04-06 19:39:40
id2181028          2  2016-03-26 13:30:55  2016-03-26 13:38:10

           passenger_count  pickup_longitude  pickup_latitude  \
id
id2875421                1        -73.982155        40.767937
id2377394                1        -73.980415        40.738564
id3858529                1        -73.979027        40.763939
id3504673                1        -74.010040        40.719971
id2181028                1        -73.973053        40.793209

           dropoff_longitude  dropoff_latitude store_and_fwd_flag  \
id
```

```
id2875421         -73.964630         40.765602                  N
id2377394         -73.999481         40.731152                  N
id3858529         -74.005333         40.710087                  N
id3504673         -74.012268         40.706718                  N
id2181028         -73.972923         40.782520                  N

           trip_duration  trip_distance(km)  avg_speed(km/hr)
id
id2875421            455              1.499         11.860220
id2377394            663              1.806          9.806335
id3858529           2124              6.385         10.822034
id3504673            429              1.485         12.461538
id2181028            435              1.189          9.840000
```

0.3.2 Create date columns:

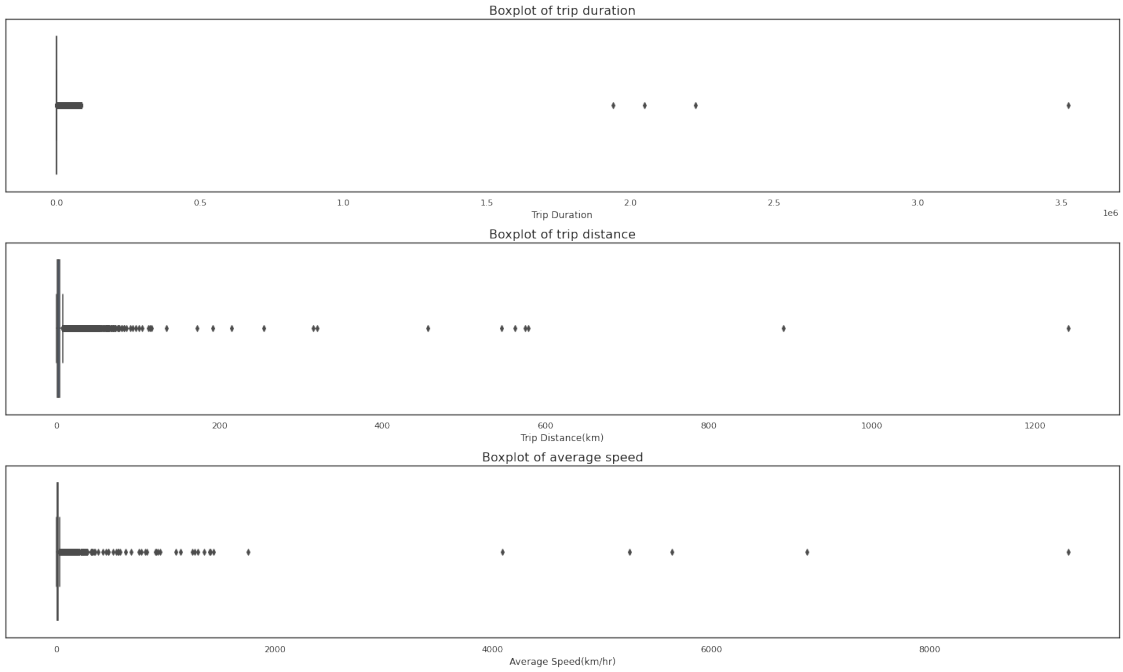
[7]:
```
# Year
df['trip_year'] = pd.to_datetime(df['pickup_datetime']).dt.year

# Month
df['trip_month'] = pd.to_datetime(df['pickup_datetime']).dt.month

# Day_of_week
df['trip_weekday'] = pd.to_datetime(df['pickup_datetime']).dt.weekday

df.head()
```

[7]:
```
           vendor_id      pickup_datetime     dropoff_datetime  \
id
id2875421          2  2016-03-14 17:24:55  2016-03-14 17:32:30
id2377394          1  2016-06-12 00:43:35  2016-06-12 00:54:38
id3858529          2  2016-01-19 11:35:24  2016-01-19 12:10:48
id3504673          2  2016-04-06 19:32:31  2016-04-06 19:39:40
id2181028          2  2016-03-26 13:30:55  2016-03-26 13:38:10

           passenger_count  pickup_longitude  pickup_latitude  \
id
id2875421                1        -73.982155        40.767937
id2377394                1        -73.980415        40.738564
id3858529                1        -73.979027        40.763939
id3504673                1        -74.010040        40.719971
id2181028                1        -73.973053        40.793209

           dropoff_longitude  dropoff_latitude store_and_fwd_flag  \
id
id2875421         -73.964630         40.765602                  N
id2377394         -73.999481         40.731152                  N
```

```
id3858529         -74.005333         40.710087                  N
id3504673         -74.012268         40.706718                  N
id2181028         -73.972923         40.782520                  N

           trip_duration  trip_distance(km)  avg_speed(km/hr)  trip_year  \
id
id2875421            455              1.499         11.860220       2016
id2377394            663              1.806          9.806335       2016
id3858529           2124              6.385         10.822034       2016
id3504673            429              1.485         12.461538       2016
id2181028            435              1.189          9.840000       2016

           trip_month  trip_weekday
id
id2875421           3             0
id2377394           6             6
id3858529           1             1
id3504673           4             2
id2181028           3             5
```

0.4 Question 1.1:

[8]:
```
# Plots
fig, axes = plt.subplots(figsize=(20,12),nrows=3, ncols = 1)

sns.boxplot(x=df['trip_duration'], ax=axes[0])
axes[0].set_title('Boxplot of trip duration',fontsize=16)
axes[0].set_xlabel('Trip Duration')

sns.boxplot(x=df['trip_distance(km)'], ax=axes[1])
axes[1].set_title('Boxplot of trip distance',fontsize=16)
axes[1].set_xlabel('Trip Distance(km)')

sns.boxplot(x=df['avg_speed(km/hr)'], ax=axes[2])
axes[2].set_title('Boxplot of average speed',fontsize=16)
axes[2].set_xlabel('Average Speed(km/hr)')

plt.tight_layout()
plt.show()
```



Inital observations:

- From the plots above we can see that there clearly some outliers in the dataset.
- The trip duration plot: It looks like the outliers occur from around 2.0×10^6seconds. This is roughly equal to 555 hours. This is not really possible.
- The trip distance plot: Some of the values from above 200km could be outliers but we can investigate further.
- The average speed plot could have values influenced by the outliers in the trip duration or trip distance.

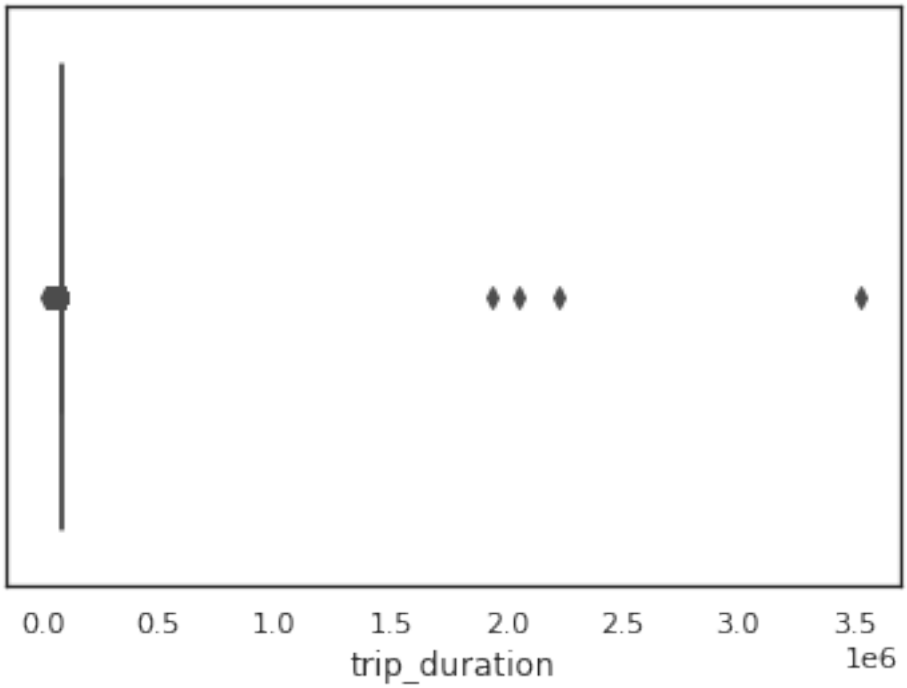
0.4.1 1. Investigate trip duration:

Z-score:

[9]:
```
z_score_1 = np.abs(stats.zscore(df['trip_duration']))
thresh = 3

# position of the outlier
filtered_entries = (z_score_1 > 3)
outlier_1 = df[filtered_entries]
print('Outliers found',len(outlier_1))
```

```
Outliers found 2073
```

[10]:
```
# View outliers
sns.boxplot(x=outlier_1['trip_duration'])
plt.show()
```



Here we can see that my inital assumption about some of the larger outliers was correct. What my inital observation missed was some the lower outliers that could occur.

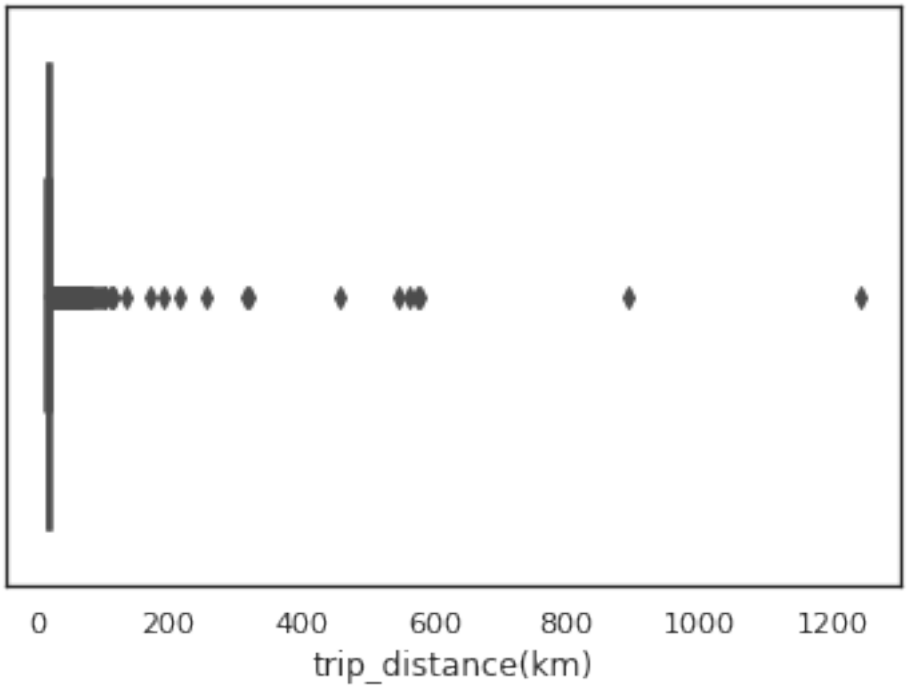
0.4.2 2. Investigate trip distance:

Z-score:

[11]:
```
z_score_2 = np.abs(stats.zscore(df['trip_distance(km)']))
thresh = 3

# position of the outlier
filtered_entries = (z_score_2 > thresh)
outlier_2 = df[filtered_entries]
print('Outliers found',len(outlier_2))
```

```
Outliers found 40117
```

[12]:
```
# View outliers
sns.boxplot(x=outlier_2['trip_distance(km)'])
plt.show()
```



We can see from the outliers above that my initial observation about the values from 200km was correct and that I may have also missed some values that are below 200km.

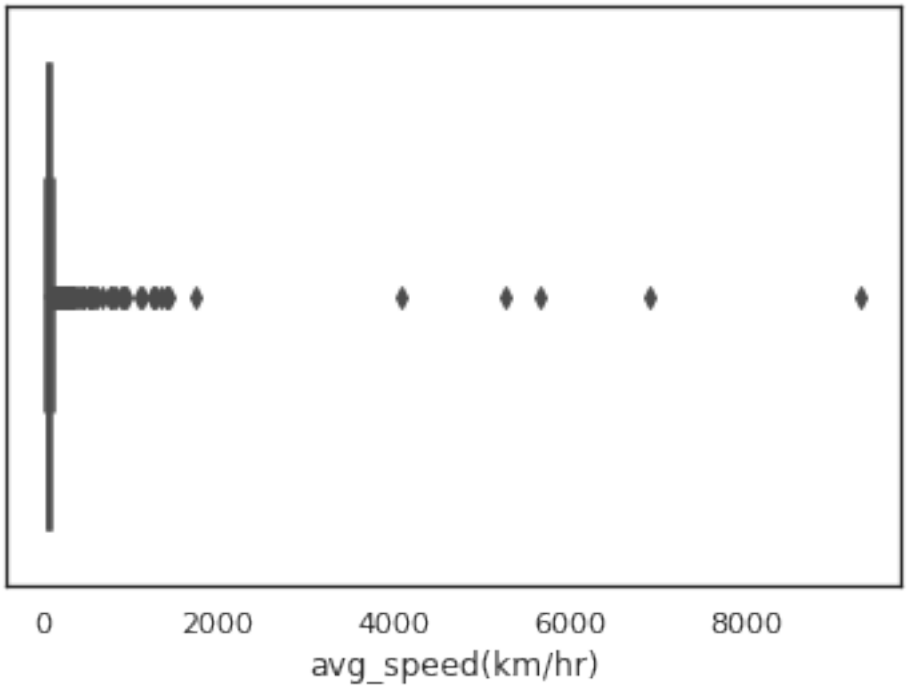
0.4.3 3. Investigate average speed:

Z-score

[13]:
```
z_score_3 = np.abs(stats.zscore(df['avg_speed(km/hr)']))
thresh = 3

# position of the outlier
filtered_entries = (z_score_3 > thresh)
outlier_3 = df[filtered_entries]
print('Outliers found',len(outlier_3))
```

```
Outliers found 734
```

[14]:
```
# View outliers
sns.boxplot(x=outlier_3['avg_speed(km/hr)'])
plt.show()
```



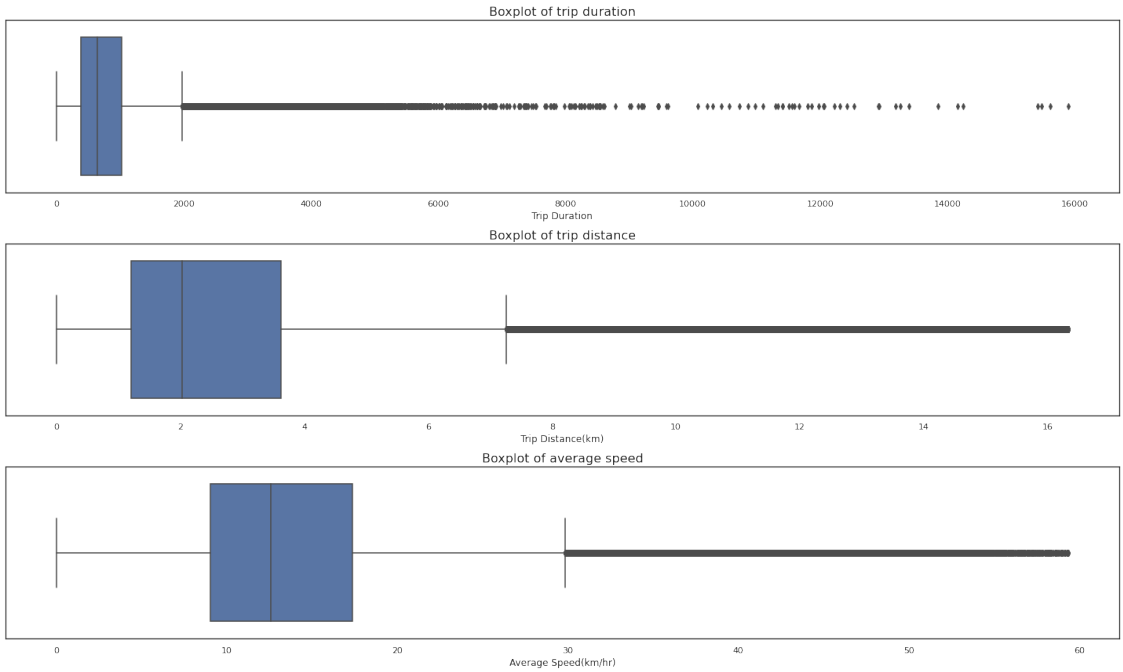
0.4.4 Dropping the outliers from the three feauture observations:

[15]:
```
# Drop the rows
outliers = outlier_1 + outlier_2 + outlier_3
df = df.drop(outliers.index)
```

0.4.5 Display Boxplots after outlier removal:

[16]:
```
# Plots
fig, axes = plt.subplots(figsize=(20,12),nrows=3, ncols = 1)

sns.boxplot(x=df['trip_duration'], ax=axes[0])
axes[0].set_title('Boxplot of trip duration',fontsize=16)
axes[0].set_xlabel('Trip Duration')

sns.boxplot(x=df['trip_distance(km)'], ax=axes[1])
axes[1].set_title('Boxplot of trip distance',fontsize=16)
axes[1].set_xlabel('Trip Distance(km)')

sns.boxplot(x=df['avg_speed(km/hr)'], ax=axes[2])
axes[2].set_title('Boxplot of average speed',fontsize=16)
axes[2].set_xlabel('Average Speed(km/hr)')

plt.tight_layout()
```

```
plt.show()
```

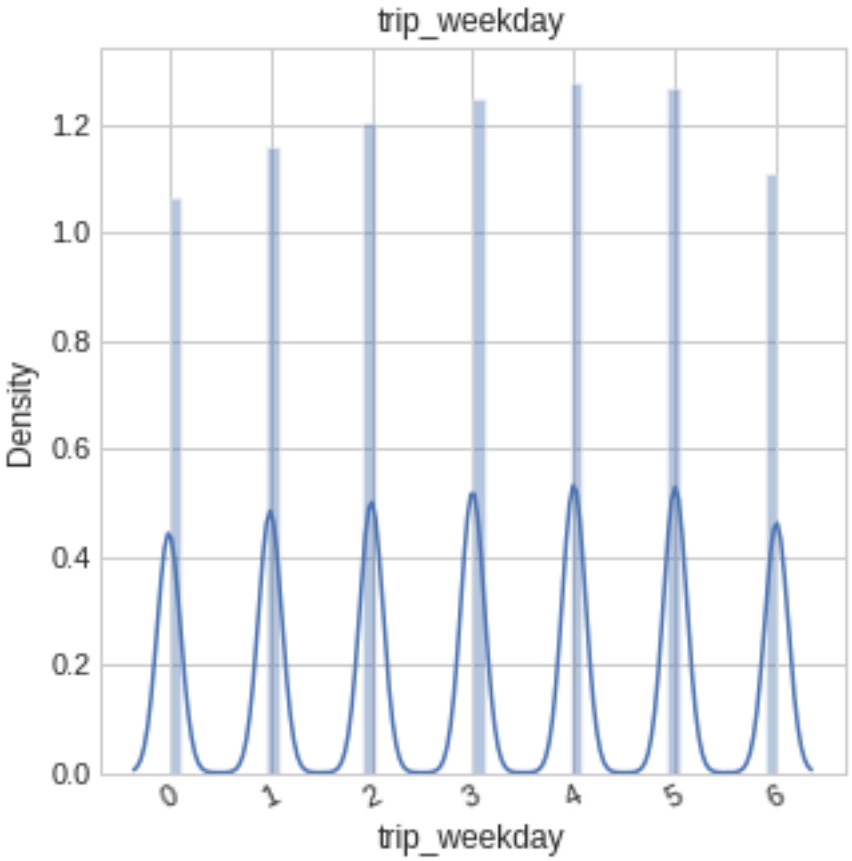


0.5 Qusetion 1.3:

Question 1.3.1:

[17]:
```
def plot_distribution(dataset, cols=5, width=20, height=15, hspace=0.2,␣
↪wspace=0.5):
    plt.style.use('seaborn-whitegrid')
    fig = plt.figure(figsize=(width,height))
    fig.subplots_adjust(left=None, bottom=None, right=None, top=None,␣
↪wspace=wspace, hspace=hspace)
    rows = math.ceil(float(dataset.shape[1]) / cols)
    for i, column in enumerate(dataset.columns):
        ax = fig.add_subplot(rows, cols, i + 1)
        ax.set_title(column)
        if dataset.dtypes[column] == object:
            g = sns.countplot(y=column, data=dataset)
            substrings = [s.get_text()[:18] for s in g.get_yticklabels()]
            g.set(yticklabels=substrings)
            plt.xticks(rotation=25)
        else:
            g = sns.distplot(dataset[column])
            plt.xticks(rotation=25)
```

[18]:
```
plot_distribution(df[['trip_weekday']], cols=3, width=20, height=5, hspace=0.5,␣
↪wspace=0.5)
```



[19]:
```
df['trip_weekday'].value_counts().idxmax()
```

[19]: 4

Question 1.3.2:

[20]:
```
df['pickup_datetime'] = pd.to_datetime(df['pickup_datetime'])
df['dropoff_datetime'] = pd.to_datetime(df['dropoff_datetime'])
df['trip_hour'] = df['pickup_datetime'].dt.hour
df.head()
```

[20]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	\
id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	
id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	
id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	
id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	

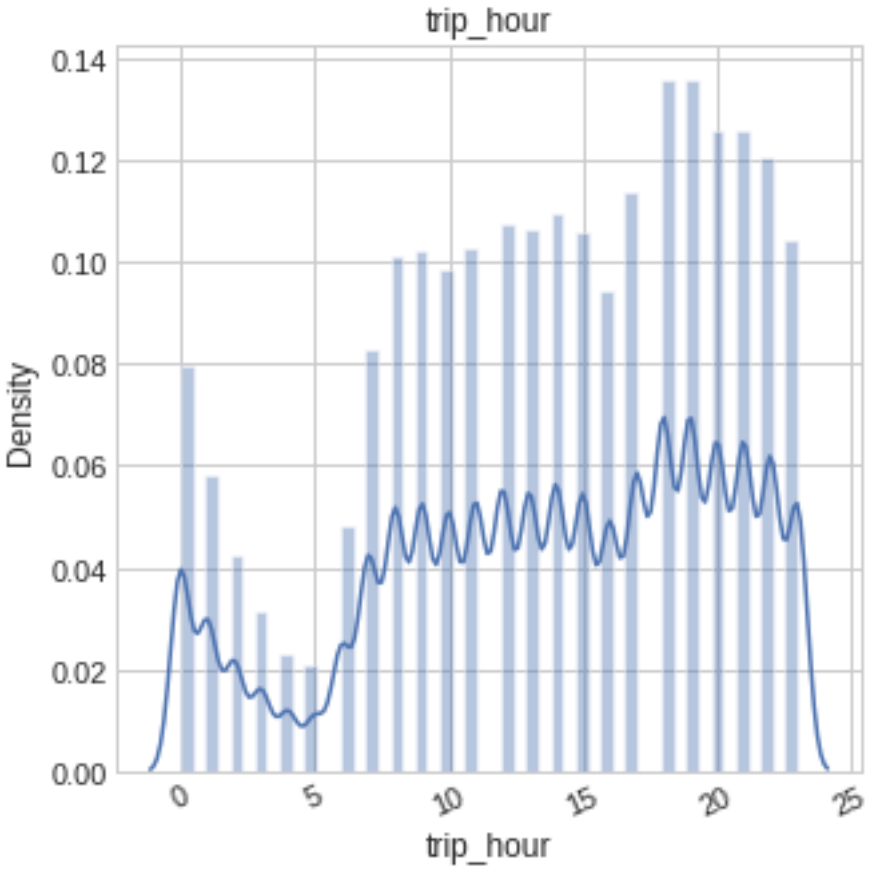
```
id2181028          2 2016-03-26 13:30:55 2016-03-26 13:38:10                1

           pickup_longitude  pickup_latitude  dropoff_longitude  \
id
id2875421        -73.982155        40.767937         -73.964630
id2377394        -73.980415        40.738564         -73.999481
id3858529        -73.979027        40.763939         -74.005333
id3504673        -74.010040        40.719971         -74.012268
id2181028        -73.973053        40.793209         -73.972923

           dropoff_latitude store_and_fwd_flag  trip_duration  \
id
id2875421         40.765602                  N            455
id2377394         40.731152                  N            663
id3858529         40.710087                  N           2124
id3504673         40.706718                  N            429
id2181028         40.782520                  N            435

           trip_distance(km)  avg_speed(km/hr)  trip_year  trip_month  \
id
id2875421              1.499         11.860220       2016           3
id2377394              1.806          9.806335       2016           6
id3858529              6.385         10.822034       2016           1
id3504673              1.485         12.461538       2016           4
id2181028              1.189          9.840000       2016           3

           trip_weekday  trip_hour
id
id2875421             0         17
id2377394             6          0
id3858529             1         11
id3504673             2         19
id2181028             5         13
```

[21]:
```
peak_hour_of_day = df.groupby('trip_weekday').agg({'trip_hour':pd.Series.mode})
peak_hour_of_day
```

[21]:
```
              trip_hour
trip_weekday
0                    18
1                    18
2                    19
3                    21
4                    19
5                    23
6                     0
```

On a Monday most people pickup at 6pm On a Tuesday most people pickup at 6pm On a Wednesday most people pickup at 7pm On a Thursday most people pickup at 9pm On a Friday most people pickup at 7pm On a Saturday most people pickup at 11pm On a Sunday most people pickup at 12am

[22]:
```
plot_distribution(df[['trip_hour']], cols=3, width=20, height=5, hspace=0.5,␣
↪wspace=0.5)
```



Most people pickup at 6pm and 7pm This can be due to most people are coming back from work at that time

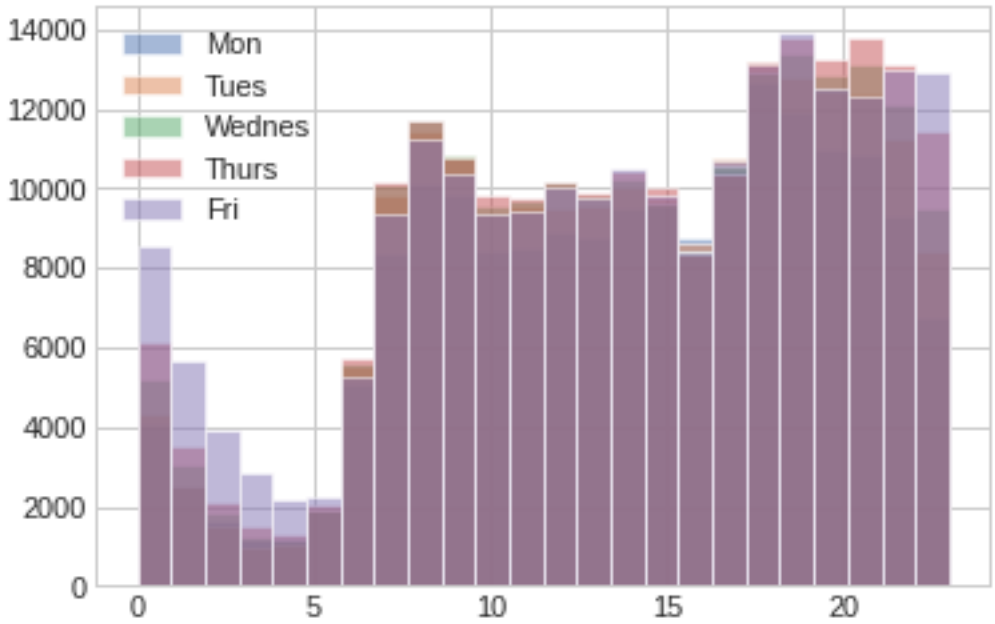
Question 1.3.3

[23]:
```
hours_in_day = df.groupby('trip_weekday')['trip_hour'].apply(list)
```

Weekdays:

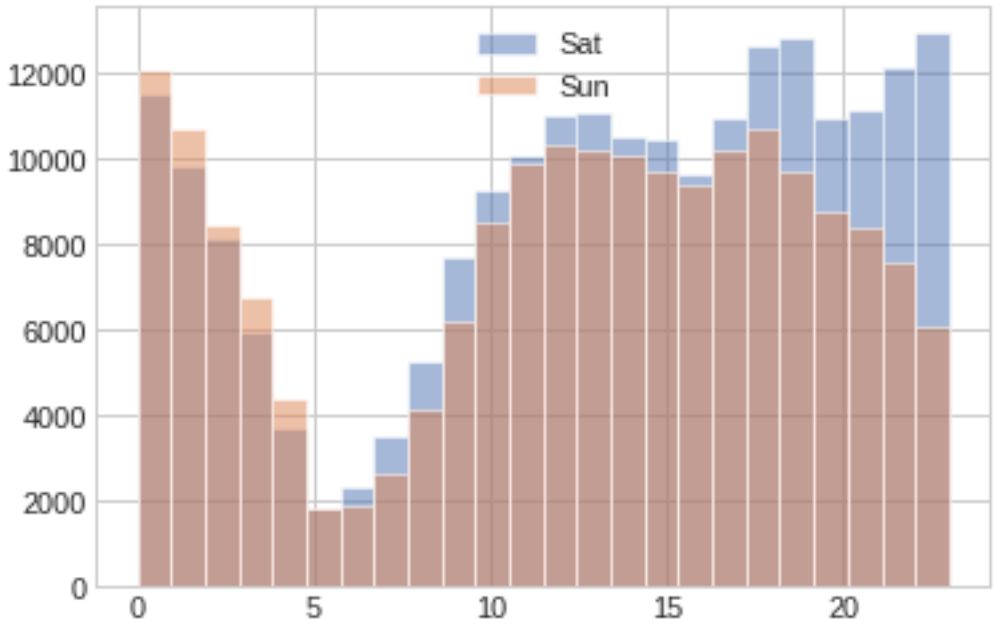
[24]:
```
plt.hist(hours_in_day[0], 24,label='Mon',alpha = 0.5)
plt.hist(hours_in_day[1], 24,label='Tues',alpha = 0.5)
plt.hist(hours_in_day[2], 24,label='Wednes',alpha = 0.5)
plt.hist(hours_in_day[3], 24,label='Thurs',alpha = 0.5)
```

```
plt.hist(hours_in_day[4], 24,label='Fri',alpha = 0.5)
plt.legend()
plt.show()
```



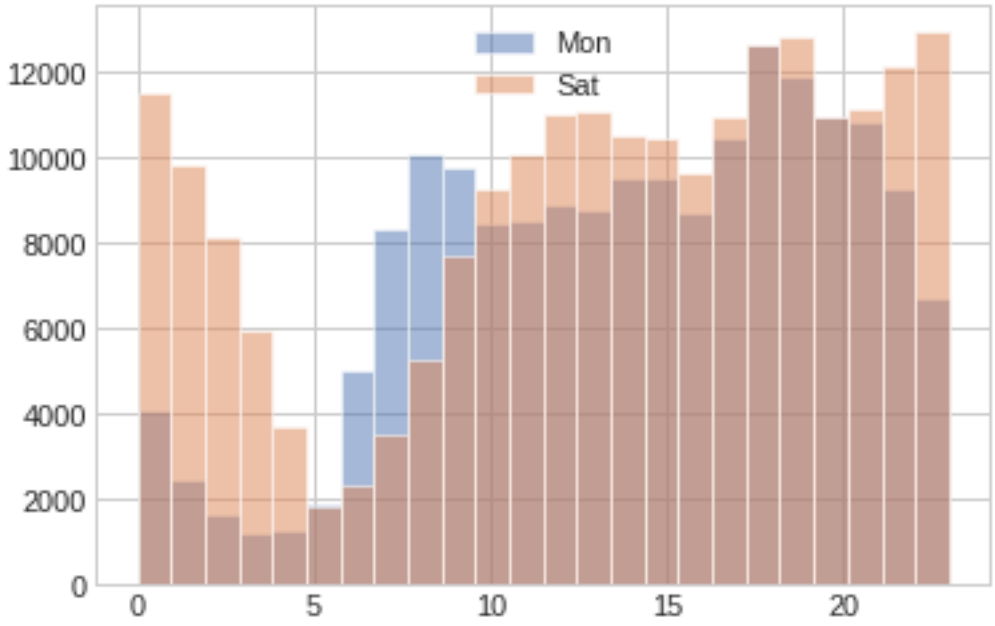
Weekends:

[25]:
```
plt.hist(hours_in_day[5], 24,label='Sat',alpha = 0.5)
plt.hist(hours_in_day[6], 24,label='Sun',alpha = 0.5)
plt.legend()
plt.show()
```



Weekday VS Weekend:

[26]:
```
plt.hist(hours_in_day[0], 24,label='Mon',alpha = 0.5)
plt.hist(hours_in_day[5], 24,label='Sat',alpha = 0.5)
plt.legend()
plt.show()
```



On weekends most people like to pickup at early Morning 12am to 5am. Saturday is similar to weekdays but there are more pickups in the early hours 12am to 5am On Sunday after 6pm most people don't pickup anymore. The early pickups on weekend might be due to people traveling to vist there families, it can also be due to people partying on the weekend. We can also see that not much pickups happen at 5am to 10am at the weekends, this can be due to people not going to work.

Question 1.3.4:

[27]:
```
cal = calendar()
Holidays = cal.holidays(start=df['pickup_datetime'].min(),␣
↪end=df['pickup_datetime'].max())
df['Holiday'] = df['pickup_datetime'].dt.date.astype('datetime64').
↪isin(Holidays)
Holidays
```

[27]:
```
DatetimeIndex(['2016-01-18', '2016-02-15', '2016-05-30'],
dtype='datetime64[ns]', freq=None)
```

[28]:
```
df.head()
```

[28]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count
id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1
id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1
id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1
id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1
id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1

id	pickup_longitude	pickup_latitude	dropoff_longitude
id2875421	-73.982155	40.767937	-73.964630
id2377394	-73.980415	40.738564	-73.999481
id3858529	-73.979027	40.763939	-74.005333
id3504673	-74.010040	40.719971	-74.012268
id2181028	-73.973053	40.793209	-73.972923

id	dropoff_latitude	store_and_fwd_flag	trip_duration
id2875421	40.765602	N	455
id2377394	40.731152	N	663
id3858529	40.710087	N	2124
id3504673	40.706718	N	429
id2181028	40.782520	N	435

id	trip_distance(km)	avg_speed(km/hr)	trip_year	trip_month

```
id
id2875421              1.499         11.860220       2016           3
id2377394              1.806          9.806335       2016           6
id3858529              6.385         10.822034       2016           1
id3504673              1.485         12.461538       2016           4
id2181028              1.189          9.840000       2016           3

           trip_weekday  trip_hour  Holiday
id
id2875421             0         17    False
id2377394             6          0    False
id3858529             1         11    False
id3504673             2         19    False
id2181028             5         13    False
```

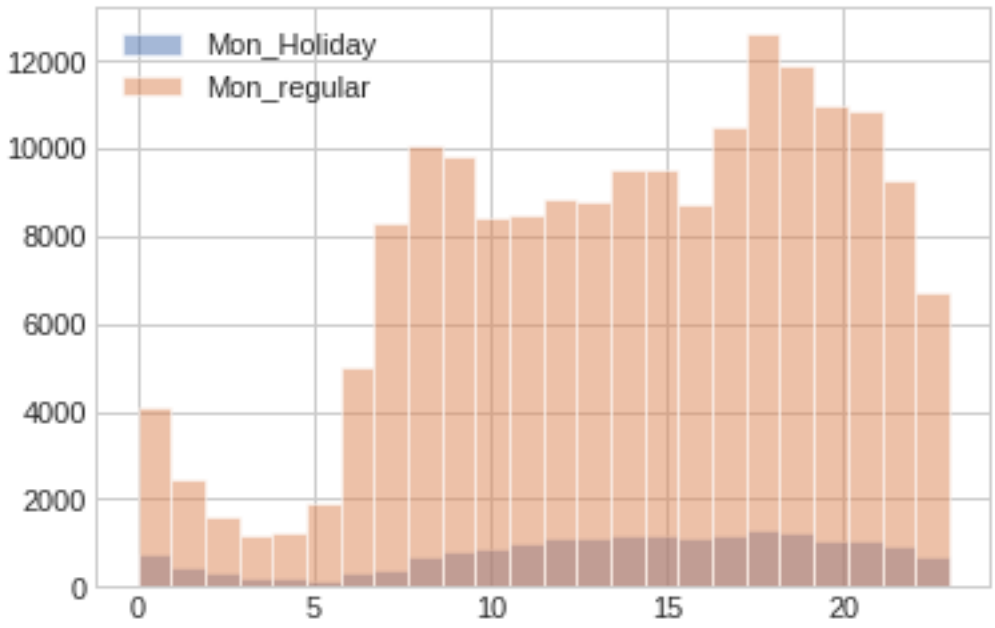
[29]:
```
Holidays_df = df[df['Holiday'] == True]
Holidays_df.head()
```

[29]:
```
           vendor_id     pickup_datetime    dropoff_datetime  passenger_count  \
id
id0675800          2 2016-02-15 09:25:15 2016-02-15 09:35:49                6
id2648478          1 2016-01-18 11:13:59 2016-01-18 11:18:56                1
id1674373          2 2016-02-15 17:52:27 2016-02-15 18:02:13                5
id2677357          2 2016-02-15 16:36:19 2016-02-15 16:41:50                2
id3013319          2 2016-02-15 22:28:54 2016-02-15 22:30:27                1

           pickup_longitude  pickup_latitude  dropoff_longitude  \
id
id0675800       -73.977753        40.754631         -74.001678
id2648478       -73.951576        40.766468         -73.960213
id1674373       -74.007500        40.740952         -74.016647
id2677357       -73.971634        40.781963         -73.981689
id3013319       -73.981400        40.778793         -73.976524

           dropoff_latitude store_and_fwd_flag  trip_duration  \
id
id0675800         40.756420                  N            634
id2648478         40.760540                  N            297
id1674373         40.704910                  N            586
id2677357         40.778996                  N            331
id3013319         40.782497                  N             93

           trip_distance(km)  avg_speed(km/hr)  trip_year  trip_month  \
id
id0675800              2.025         11.498423       2016           2
id2648478              0.982         11.903030       2016           1
id1674373              4.081         25.070990       2016           2
```

```
id2677357              0.909          9.886405       2016           2
id3013319              0.581         22.490323       2016           2

           trip_weekday  trip_hour  Holiday
id
id0675800             0          9     True
id2648478             0         11     True
id1674373             0         17     True
id2677357             0         16     True
id3013319             0         22     True
```

[30]:
```
hours_in_day_holidays = Holidays_df.groupby('trip_weekday')['trip_hour'].
↪apply(list)
```

[31]:
```
plt.hist(hours_in_day_holidays[0], 24,label='Mon_Holiday',alpha = 0.5)
plt.hist(hours_in_day[0], 24,label='Mon_regular',alpha = 0.5)
plt.legend()
plt.show()
```



Compared to a noraml Monday, the one on Holiday indicate that they are far less pickups then normal. This shows that most people don't work or travel on Holidays

Question 1.3.5:

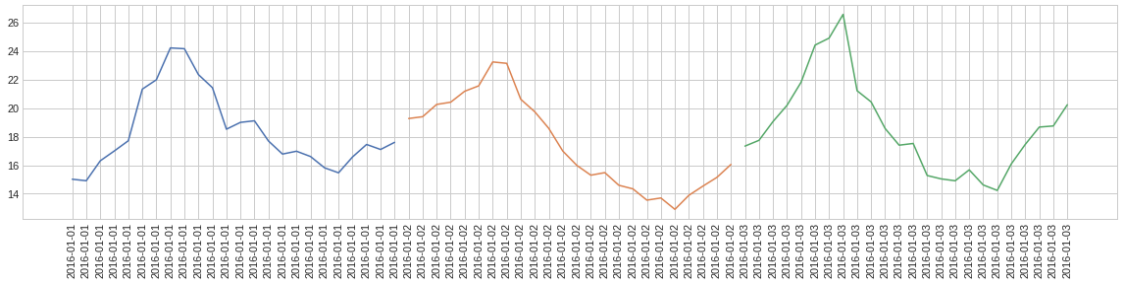
[32]:
```
day_speed = df[['pickup_datetime','avg_speed(km/hr)']]
day_speed.head()
```

[32]:
```
                pickup_datetime  avg_speed(km/hr)
id
id2875421 2016-03-14 17:24:55         11.860220
id2377394 2016-06-12 00:43:35          9.806335
id3858529 2016-01-19 11:35:24         10.822034
id3504673 2016-04-06 19:32:31         12.461538
id2181028 2016-03-26 13:30:55          9.840000
```

[33]:
```
day_speed_df = day_speed.groupby(pd.Grouper(key='pickup_datetime',␣
↪freq='60min')).mean().dropna()
day_speed_df.head()
```

[33]:
```
                     avg_speed(km/hr)
pickup_datetime
2016-01-01 00:00:00         15.019020
2016-01-01 01:00:00         14.912572
2016-01-01 02:00:00         16.311287
2016-01-01 03:00:00         16.999992
2016-01-01 04:00:00         17.716125
```

[34]:
```
plt.figure(figsize=(20, 4))
plt.plot(day_speed_df.index[:24],day_speed_df.values[:24])
plt.plot(day_speed_df.index[24:48],day_speed_df.values[24:48])
plt.plot(day_speed_df.index[48:72],day_speed_df.values[48:72])
plt.xticks(rotation=90)
plt.xticks(day_speed_df.index[:72])
plt.show()
```



From the graphs it is seen that at around 5am to 8am the highest average speeds are archieved around 22km/hr to 27km/hr. This can be caused by people rushing to work. The slowest speeds are around 5pm to 8pm. This is when the taxis travel the slowest below 14km/hr. This can be due to traffic when everyone is returning from work and because people are not necessary in a rush when returning from work.

0.6 Question 1.5:

[13]:
```
data = pd.read_csv('nyc_taxis.csv', delimiter=',', header=0, index_col=0)
data.head(5)
```

[13]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration
id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	-73.982155	40.767937	-73.964630	40.765602	N	455
id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	-73.980415	40.738564	-73.999481	40.731152	N	663
id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	-73.979027	40.763939	-74.005333	40.710087	N	2124
id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	-74.010040	40.719971	-74.012268	40.706718	N	429
id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1	-73.973053	40.793209	-73.972923	40.782520	N	435

[14]:
```
pickup_datetime_update = data['pickup_datetime'].astype('datetime64[ns]')
dropoff_datetime_update = data['dropoff_datetime'].astype('datetime64[ns]')
data['pickup_datetime'] = pickup_datetime_update
data['dropoff_datetime'] = dropoff_datetime_update
pickup_time = data['pickup_datetime'][0]
dropoff_time = data['dropoff_datetime'][0]

def to_seconds(duration):
    return duration.total_seconds()
```

```
travel_time =  data['dropoff_datetime'].sub(data['pickup_datetime'], axis = 0)

data['travel_time'] = travel_time

data['travel_time'] = data['travel_time'].apply(to_seconds)

data.head(5)
```

[14]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration	travel_time
id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	-73.982155	40.767937	-73.964630	40.765602	N	455	455.0
id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	-73.980415	40.738564	-73.999481	40.731152	N	663	663.0
id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	-73.979027	40.763939	-74.005333	40.710087	N	2124	2124.0
id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	-74.010040	40.719971	-74.012268	40.706718	N	429	429.0
id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1	-73.973053	40.793209	-73.972923	40.782520	N	435	435.0

[37]:
```
import reverse_geocoder as rg
import pprint

def reverseGeocode(coordinates):
    result = rg.search(coordinates)

    # result is a list containing ordered dictionary.
    pprint.pprint(result)
```

[38]:
```
# coordinates for Empire State Building
reverseGeocode((40.748541, -73.985758))
```

```
Loading formatted geocoded file…
[{'admin1': 'New York',
  'admin2': 'Queens County',
```

```
  'cc': 'US',
  'lat': '40.74482',
  'lon': '-73.94875',
  'name': 'Long Island City'}]
```

[39]:
```
# coordinates for JFK AIRPORT
reverseGeocode((40.647352,  -73.790534))
```

```
[{'admin1': 'New York',
  'admin2': 'Queens County',
  'cc': 'US',
  'lat': '40.69149',
  'lon': '-73.80569',
  'name': 'Jamaica'}]
```

[40]:
```
# coordinates for NEWARK LIBERTY AIRPORT
reverseGeocode((40.704197,  -74.190124))
```

```
[{'admin1': 'New Jersey',
  'admin2': 'Essex County',
  'cc': 'US',
  'lat': '40.73566',
  'lon': '-74.17237',
  'name': 'Newark'}]
```

[41]:
```
data_JFK = data[abs(data['pickup_latitude'] -  40.748541) <= 0.009999]
data_JFK = data_JFK[abs(data_JFK['pickup_longitude'] + 73.985758) <= 0.009999]
data_JFK = data_JFK[abs(data_JFK['dropoff_latitude'] - 40.647352) <= 0.009999]
data_JFK = data_JFK[abs(data_JFK['dropoff_longitude'] + 73.790534) <= 0.009999]

data_JFK.head()
```

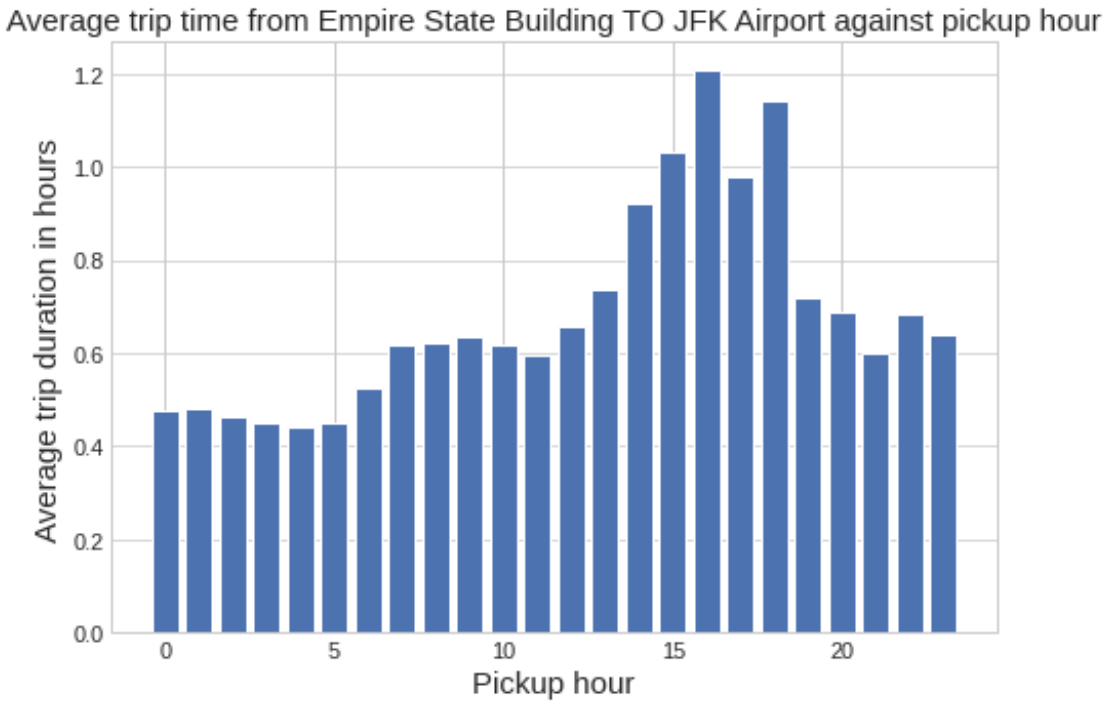
[41]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude	pickup_latitude	dropoff_longitude
id3930440	1	2016-06-01 14:29:00	2016-06-01 15:44:28	1	-73.987869	40.748219	-73.783760
id1365474	1	2016-04-26 16:24:10	2016-04-26 17:24:13	1	-73.992050	40.751514	-73.790039
id3401826	1	2016-04-27 10:20:02	2016-04-27 11:01:32	2	-73.992538	40.756557	-73.789757
id1909463	1	2016-03-27 10:13:49	2016-03-27 10:41:50	2	-73.990303	40.756023	-73.788696
id3861834	1	2016-05-09 16:24:51	2016-05-09 17:29:44	2	-73.978928	40.745937	-73.790367

id	dropoff_latitude	store_and_fwd_flag	trip_duration	travel_time
id3930440	40.643600	N	4528	4528.0
id1365474	40.646957	N	3603	3603.0
id3401826	40.643002	N	2490	2490.0
id1909463	40.647415	N	1681	1681.0
id3861834	40.646748	N	3893	3893.0

[42]:
```
times = pd.DatetimeIndex(data_JFK.pickup_datetime)
grouped_by_hour = data_JFK.groupby([times.hour]).mean()/(60*60)
average_trip_time = list(grouped_by_hour['trip_duration'])
pickup_hours = list(range(0, 24))
```

[43]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(pickup_hours, average_trip_time)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Pickup hour',fontsize=15)
plt.ylabel('Average trip duration in hours',fontsize=15)
plt.title('Average trip time from Empire State Building TO JFK Airport against␣
 ↪pickup hour',fontsize=15)
plt.show()
```



[44]:
```
data_Newark = data[abs(data['pickup_latitude'] -  40.748541) <= 0.009999]
data_Newark = data_Newark[abs(data_Newark['pickup_longitude'] + 73.985758) <= 0.
↪009999]
data_Newark = data_Newark[abs(data_Newark['dropoff_latitude'] - 40.704197) <= 0.
↪009999]
data_Newark = data_Newark[abs(data_Newark['dropoff_longitude'] + 74.190124) <=␣
↪0.009999]

data_Newark
```

[44]:

id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count
id1938148	1	2016-05-05 05:04:30	2016-05-05 05:30:56	4
id0922516	1	2016-04-14 17:42:41	2016-04-14 18:32:34	2
id1024232	2	2016-05-29 13:22:50	2016-05-29 14:01:47	1
id2930052	2	2016-03-26 13:28:12	2016-03-26 14:01:03	1
id3302698	1	2016-02-07 20:44:28	2016-02-07 21:09:56	1
id3774443	2	2016-03-24 02:49:53	2016-03-24 03:14:40	1
id0018553	1	2016-03-09 05:03:15	2016-03-09 05:23:31	2
id1831880	1	2016-03-20 13:57:33	2016-03-20 14:26:04	1
id0503169	2	2016-01-19 12:09:52	2016-01-19 12:46:29	1
id1425169	1	2016-01-01 23:59:40	2016-01-02 00:29:16	2
id0170205	1	2016-02-14 13:19:18	2016-02-14 13:39:04	1

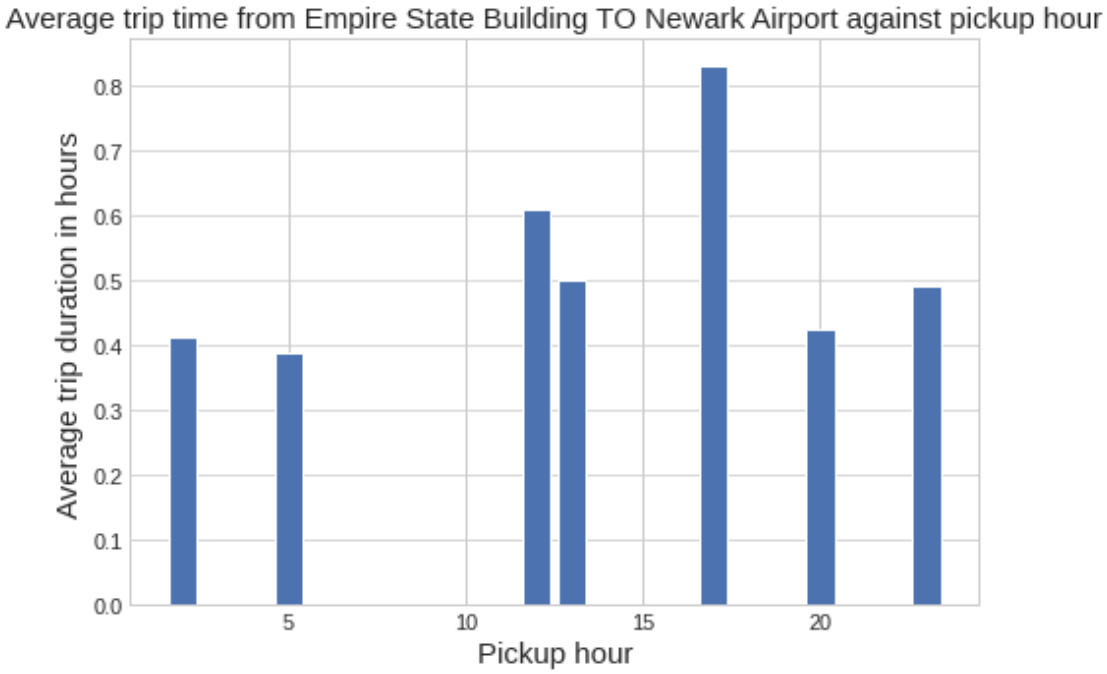
id	pickup_longitude	pickup_latitude	dropoff_longitude
id1938148	-73.987572	40.755486	-74.185478
id0922516	-73.985268	40.741959	-74.188423
id1024232	-73.984085	40.746181	-74.180962
id2930052	-73.976532	40.752068	-74.183891
id3302698	-73.995567	40.750332	-74.184097
id3774443	-73.992813	40.748081	-74.186218
id0018553	-73.992096	40.748821	-74.187927
id1831880	-73.982964	40.757057	-74.181496
id0503169	-73.977722	40.754810	-74.181709
id1425169	-73.992821	40.752155	-74.187431
id0170205	-73.992348	40.754353	-74.180374

id	dropoff_latitude	store_and_fwd_flag	trip_duration	travel_time
id1938148	40.694221	N	1586	1586.0
id0922516	40.695454	N	2993	2993.0
id1024232	40.695114	N	2337	2337.0
id2930052	40.700661	N	1971	1971.0
id3302698	40.696411	N	1528	1528.0
id3774443	40.697731	N	1487	1487.0
id0018553	40.696171	N	1216	1216.0

id1831880	40.705097	N	1711	1711.0
id0503169	40.694752	N	2197	2197.0
id1425169	40.705273	N	1776	1776.0
id0170205	40.705505	N	1186	1186.0

[45]:
```
times = pd.DatetimeIndex(data_Newark.pickup_datetime)
grouped_by_hour = data_Newark.groupby([times.hour]).mean()/(60*60)
average_trip_time_Newark = list(grouped_by_hour['trip_duration'])
pickup_hours_Newark = list(grouped_by_hour.index)
```

[46]:
```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(pickup_hours_Newark, average_trip_time_Newark)
plt.grid(axis='y', alpha=0.75)
plt.xlabel('Pickup hour',fontsize=15)
plt.ylabel('Average trip duration in hours',fontsize=15)
plt.title('Average trip time from Empire State Building TO Newark Airport␣
 ↪against pickup hour',fontsize=15)
plt.show()
```



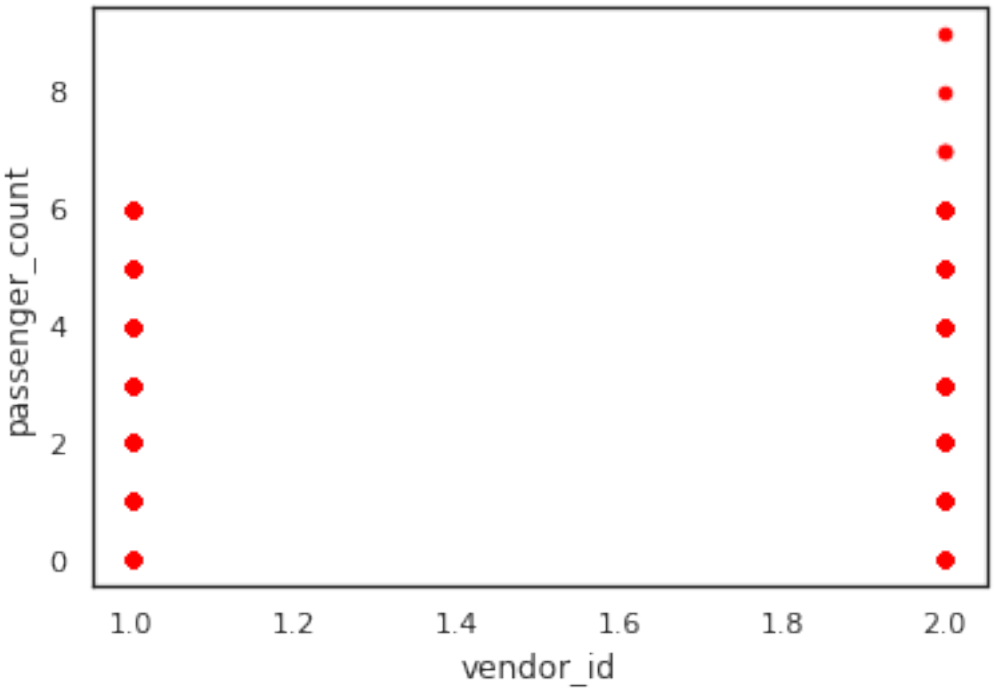
0.6.1 Comments:

The major difference is that there are always trips from Empire State Building to JFK Airport every hour. Whereas, this is not the case from Empire State Building to Newark Airport since some of the averages trip times in the second graph are zeros for several pickup hours.

0.7 Question 1.6:

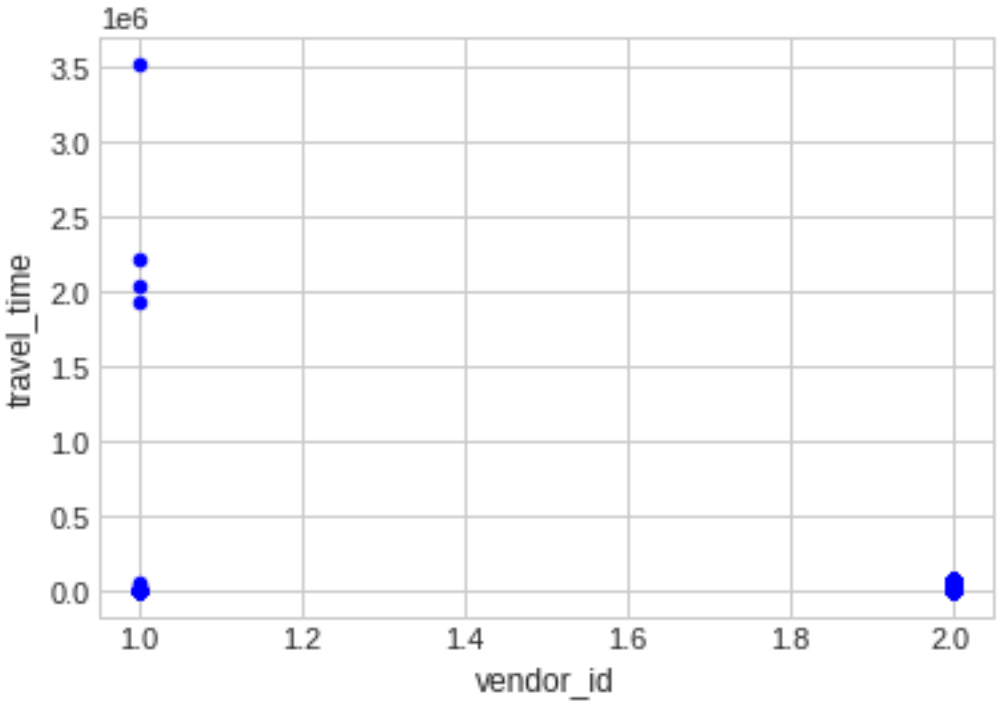
0.7.1 Number of passengers:

[15]:
```
data  = data.reset_index()
data.plot.scatter(x='vendor_id',y='passenger_count',color = 'red',␣
↪colormap='viridis')
plt.show()
```



0.7.2 Duration of trips:

[55]:
```
data.plot.scatter(x='vendor_id',y='travel_time',color = 'blue', ␣
↪colormap='viridis')
plt.show()
```



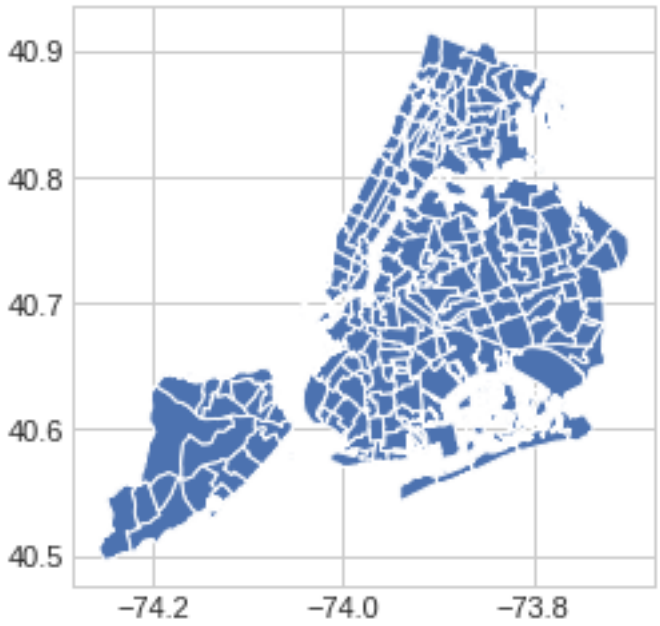
0.7.3 Pickup locations:

[49]:
```
pickup_data = data[['vendor_id', 'pickup_latitude', 'pickup_longitude']]
gdf_pickup = geopandas.GeoDataFrame(pickup_data, geometry=geopandas.
 ↪points_from_xy(data.pickup_latitude, data.pickup_longitude))

gdf_pickup.head()
```

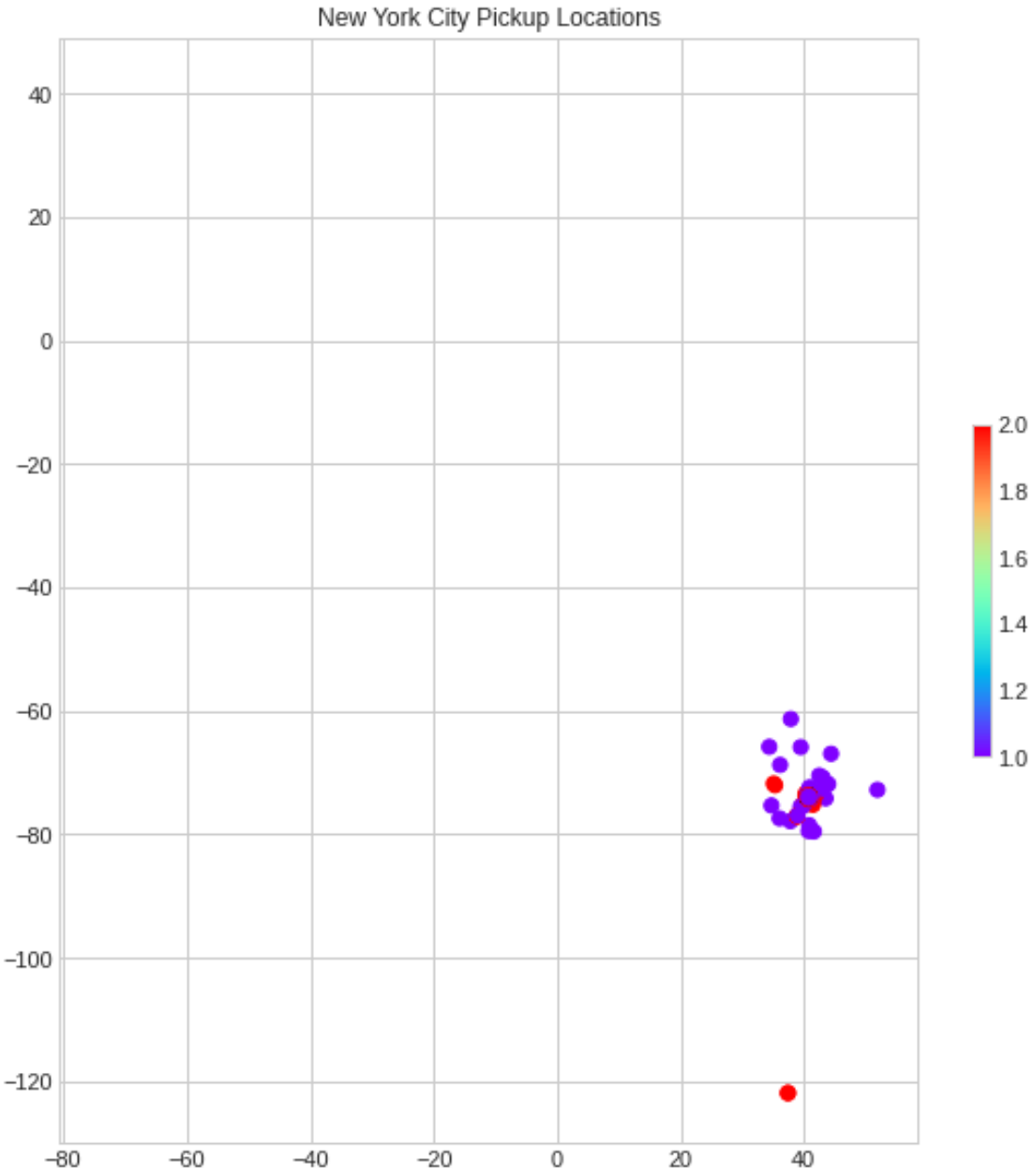
[49]:

	vendor_id	pickup_latitude	pickup_longitude	geometry
0	2	40.767937	-73.982155	POINT (40.76794 -73.98215)
1	1	40.738564	-73.980415	POINT (40.73856 -73.98042)
2	2	40.763939	-73.979027	POINT (40.76394 -73.97903)
3	2	40.719971	-74.010040	POINT (40.71997 -74.01004)
4	2	40.793209	-73.973053	POINT (40.79321 -73.97305)

[50]:
```
nyc_map = geopandas.read_file('geo_export_b0262261-5940-4b03-b89d-d4eb921ae481.
 ↪dbf')
nyc_map.to_crs(epsg=4326).plot()
plt.show()
```



[51]:
```
fig, ax = plt.subplots(figsize = (10,10))
nyc_map.to_crs(epsg=4326).plot(ax=ax)
gdf_pickup.plot(column = 'vendor_id', ax=ax, cmap = 'rainbow',legend = True,␣
↪legend_kwds={'shrink': 0.3}, markersize = 50)
ax.set_title('New York City Pickup Locations')
plt.show()
```



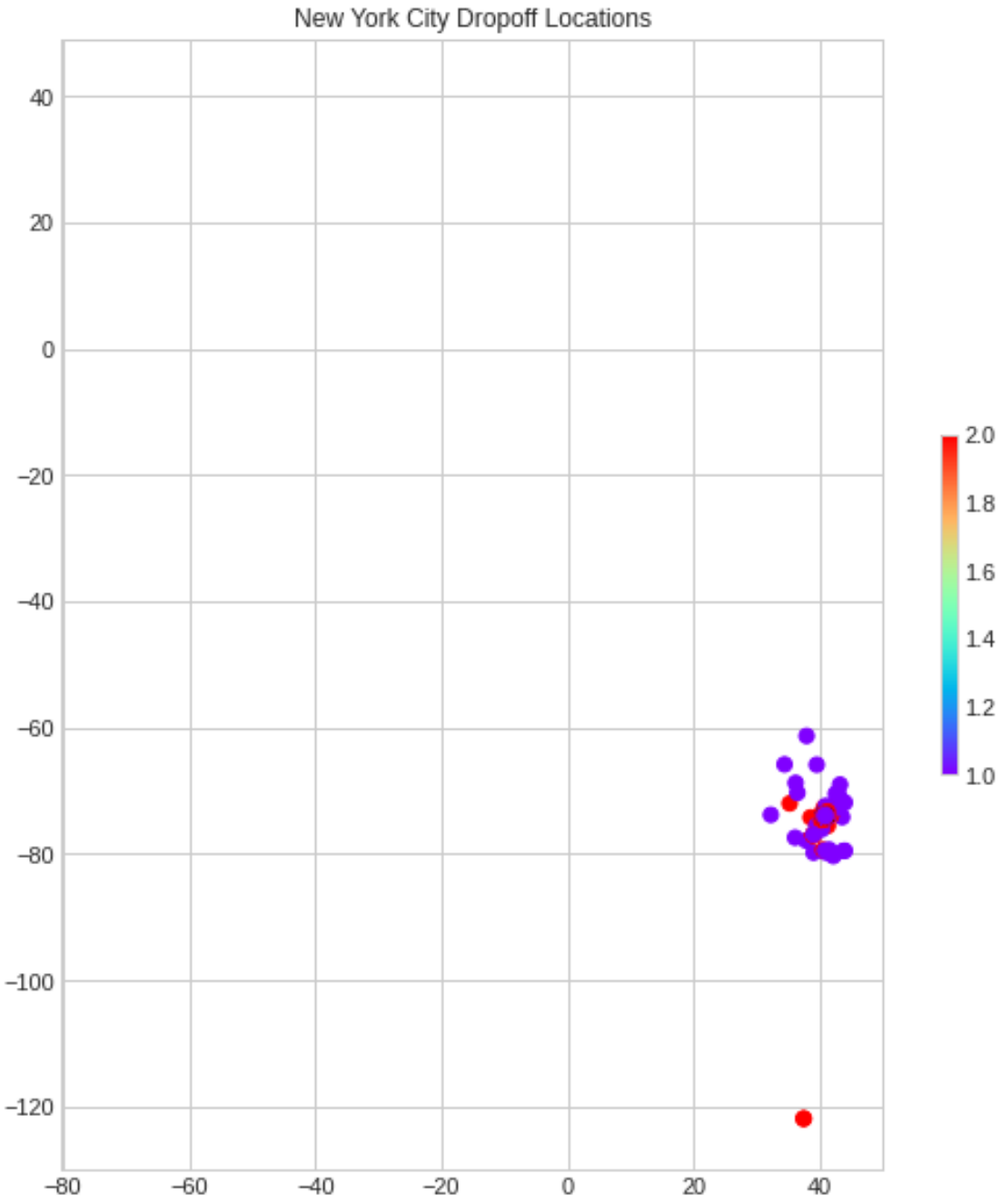
0.7.4 Dropoff locations:

[52]:
```
dropoff_data = data[['vendor_id', 'dropoff_latitude', 'dropoff_longitude']]
gdf_dropoff = geopandas.GeoDataFrame(dropoff_data, geometry=geopandas.
 ↪points_from_xy(data.dropoff_latitude, data.dropoff_longitude))

gdf_dropoff.head()
```

[52]:

	vendor_id	dropoff_latitude	dropoff_longitude	geometry
0	2	40.765602	-73.964630	POINT (40.76560 -73.96463)
1	1	40.731152	-73.999481	POINT (40.73115 -73.99948)
2	2	40.710087	-74.005333	POINT (40.71009 -74.00533)
3	2	40.706718	-74.012268	POINT (40.70672 -74.01227)
4	2	40.782520	-73.972923	POINT (40.78252 -73.97292)

[53]:
```
fig, ax = plt.subplots(figsize = (10,10))
nyc_map.to_crs(epsg=4326).plot(ax=ax)
gdf_dropoff.plot(column = 'vendor_id', ax=ax, cmap = 'rainbow',legend = True,␣
 ↪legend_kwds={'shrink': 0.3}, markersize = 50)
ax.set_title('New York City Dropoff Locations')
plt.show()
```



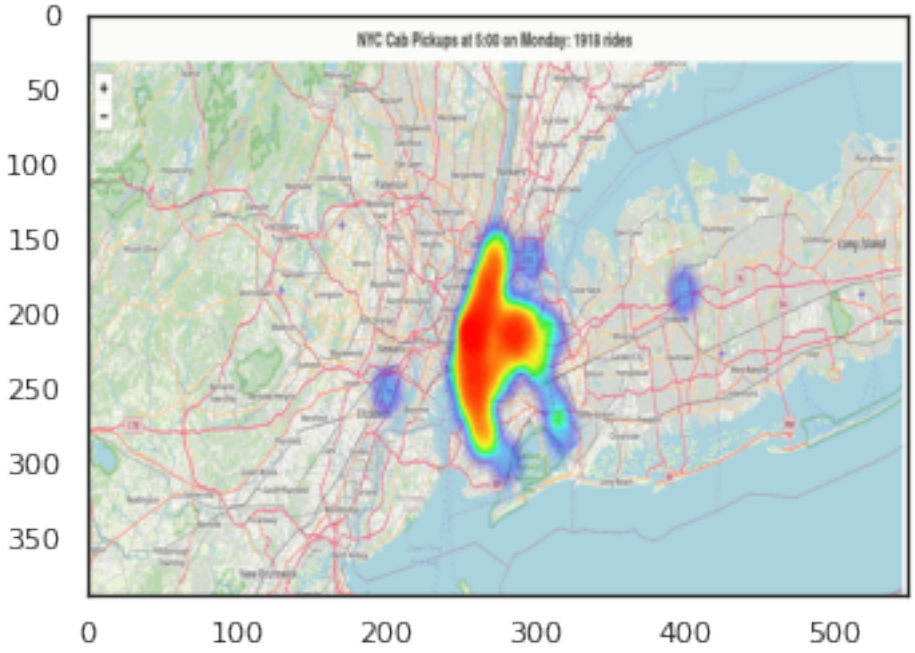
Comments: It is evident from the plots above that the vendor with a vendor_id 1 is Yellow Cabs and the vendor with a vendor_id 2 is boro taxi. We came to this conclusion since we know that Yellow Cabs operates all over New york city and Boro taxis operates within restricted areas. So, Yellow Cabs are expected to have more average travel duration than boro taxis and Yellow Cabs are expected to have more pickup locations and dropoff locations spread out New York City compared to Boros. Also, Yellow Cabs are expected to have at most 6 passengers whereas for Boros this number can be greater. These expectations were all visible from the plots.

0.8 Question 1.4:

0.8.1 Weekdays:

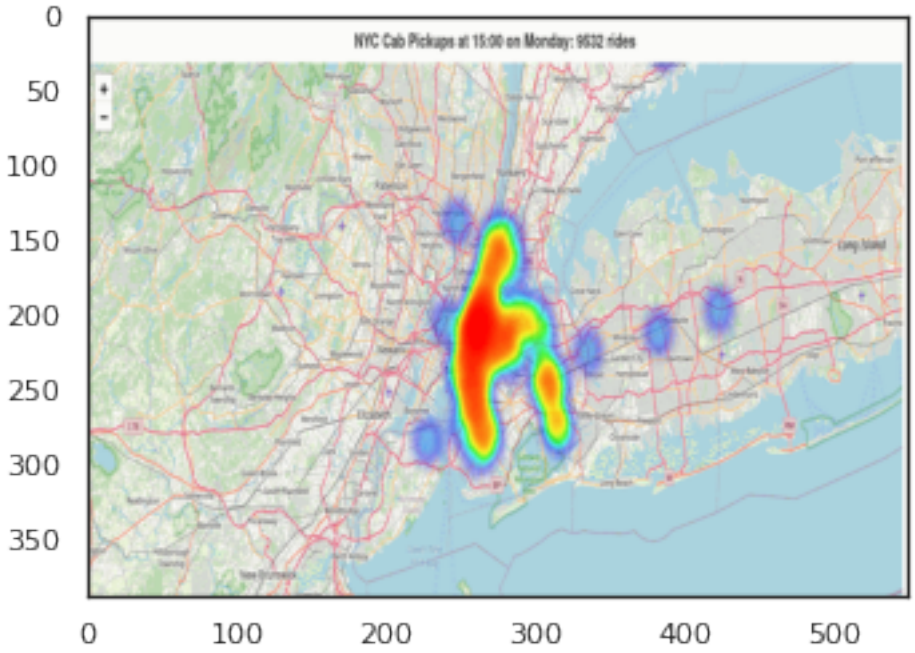
Mornings:

[5]:
```
image1 = mpimg.imread('pickup_weekday_morning.gif')
imgplot = plt.imshow(image1)
plt.show()
```



Afternoons:

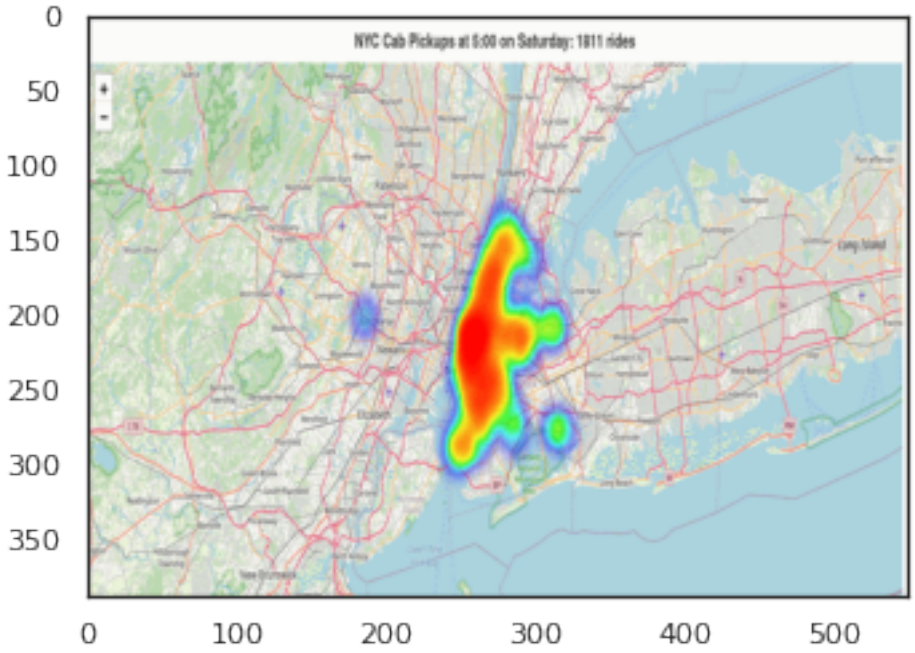
[6]:
```
image2 = mpimg.imread('pickup_weekday_afternoon.gif')
imgplot = plt.imshow(image2)
plt.show()
```



0.8.2 Weekends:

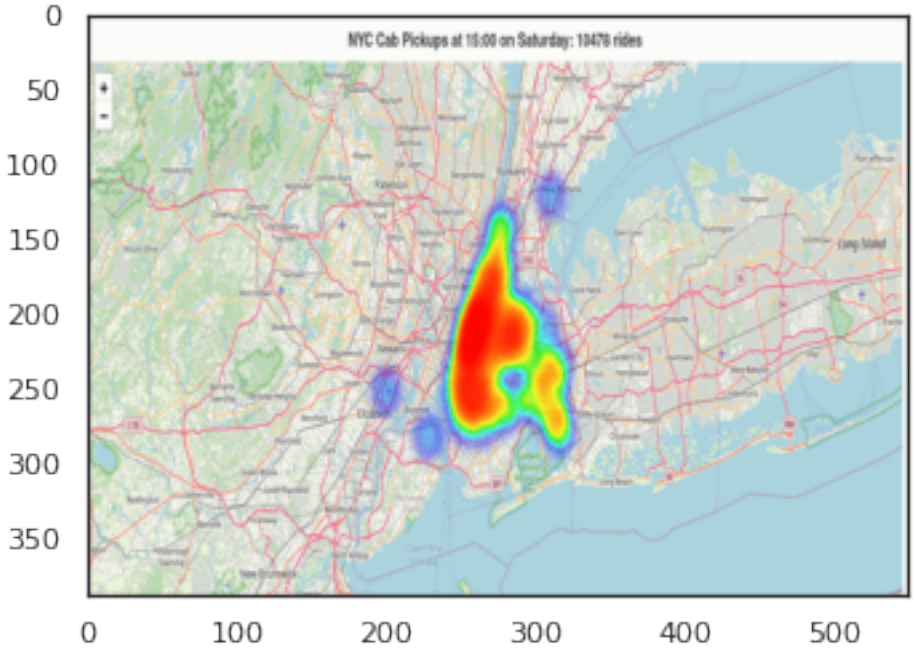
Mornings:

[11]:
```
image3 = mpimg.imread('pickup_weekend_morning.gif')
imgplot = plt.imshow(image3)
plt.show()
```



Afternoons:

[12]:
```
image4 = mpimg.imread('pickup_weekend_afternoon.gif')
imgplot = plt.imshow(image4)
plt.show()
```



0.8.3 Comments:

There are a lot of trips taken in the mornings of weekdays compared to the trips taken in the afternoons of weekdays.Also, there a lot of trips taken in the afternoons of weekends compared to trips taken in the mornings of weekends. In the morning of weekdays normally people are rushing to work or school so this explains the high volume of trips during these hours, whereas in the afternoon are people may opt to walk to their places. During weekends in the morning most people are usually in doors, so they make less trips. In the afternnons of weekends, people may come back late from events and parties and hence may need transport to go back home so this explains the huge volume of trips during this time.

[ ]: