Uber Pickup Analysis In New York City

CMT212 Visual Communication And Information Design
MSc Data Science And Analytics
Cardiff University
C1753774
May 2018

Private real-time cab services such as Uber follow a rising trend during the past years, so commuters have a lot of options for their daily trips but professional drivers face a lot of competition to get passengers. As a result, its vital for drivers working in such cab services to be in the right place at the right time. Choosing a metropolitan place like New York City, the purpose of this study is to analyse Uber pickups across 6 months from April to September in 2014 and report interesting findings. For further exploration of the data set a cluster analysis was used in order to find particular hot spots through the city of New York over a particular day and where an Uber driver is more likely to find a passenger at a given hour in the city.

Also, a separate data set containing again Uber pickups in New York City from 01/01/2015 to 30/06/2015 but with several weather information was used. The purpose of this additional analysis was to examine if there is a certain connection between extreme weather temperatures and people's preference to Uber.

1 Data Pre-Processing

Before starting the initial analysis of the data, some pre-processing and checking must be done in order to ensure that the data set is clean and formatted according to our needs. The first step taken is to concatenate 6 different files containing data from Uber trips such as Date/Time, Latitude, Longitude and Base, each one of these files representing a month. After, the whole data set containing 4.534.327 Uber trips is checked for missing and duplicate values which could deteriorate our analysis. No missing or duplicate values were found so we can move to the formatting.

The main purpose of the formatting process is to extract valuable information from already existing values on the data set which can provide information needed later in our analysis. Converting the Date/Time column in a timestamp panda object will help a lot distinguishing features like date, month and year easily later. We could use the Date/Time column for sorting and grouping directly, but it will be much easier now that we are still at an early stage of the analysis to just create new columns and extract some features from the timestamp object that will be used in the later stages. Given the size of the dataset the whole reformatting process took 7.7 minutes approximately to run and the results can be displayed on Table 1.

	timestamp	lat	lon	base	weekday	month	day	hour	minute	dayofweek
0	2014-04-01 00:11:00	40.7690	-73.9549	B02512	Tuesday	4	1	0	11	1
1	2014-04-01 00:17:00	40.7267	-74.0345	B02512	Tuesday	4	1	0	17	1
2	2014-04-01 00:21:00	40.7316	-73.9873	B02512	Tuesday	4	1	0	21	1
3	2014-04-01 00:28:00	40.7588	-73.9776	B02512	Tuesday	4	1	0	28	1
4	2014-04-01 00:33:00	40.7594	-73.9722	B02512	Tuesday	4	1	0	33	1

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4534327 entries, 0 to 4534326
Data columns (total 10 columns):
timestamp    object
lat          float64
lon          float64
base         object
weekday      object
month        int64
day          int64
hour         int64
minute       int64
dayofweek    int64
dtypes: float64(2), int64(5), object(3)
memory usage: 345.9+ MB
```

Table 1: First Uber Trips Dataset Containing Trip Information

The second data set included is in a format that suits our analysis, so no formatting is needed. It was only checked for missing and duplicate values and none of the tests performed found any.

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
0	2015-01-01 01:00:00	Bronx	152	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
1	2015-01-01 01:00:00	Brooklyn	1519	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
2	2015-01-01 01:00:00	EWR	0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
3	2015-01-01 01:00:00	Manhattan	5258	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
4	2015-01-01 01:00:00	Queens	405	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29101 entries, 0 to 29100
Data columns (total 13 columns):
pickup_dt    29101 non-null object
borough      26058 non-null object
pickups      29101 non-null int64
spd          29101 non-null float64
vsb          29101 non-null float64
temp         29101 non-null float64
dewp         29101 non-null float64
slp          29101 non-null float64
pcp01        29101 non-null float64
pcp06        29101 non-null float64
pcp24        29101 non-null float64
sd           29101 non-null float64
hday         29101 non-null object
dtypes: float64(9), int64(1), object(3)
memory usage: 2.9+ MB
```

Table 2: Second Uber Trips Dataset Containing Pickup Timestamp and Weather Condition Measurements

2 Explanatory Analysis

Since our data are checked and formatted we can now move to the stage of the analysis. At first the number of trips is plotted against different times of a day, separated in 1-hour intervals and the graph is visualized in *Figure 1*. The plot was initially created using Python, but then I thought it would more efficient to add an interactive map for better understanding of the data, so a Tableau dashboard was created containing both graphs. The interactive map will visualize Uber trips changing automatically every hour by pressing the ⇒ button in the tooltip. It can be observed from the histogram that the number of Uber trips follow an increasing trend from 13:00 and reaches a total maximum of 336.19k trips at 17:00. Another rapid increase in the number of trips can be observed from 4:00 to 7:59 reaching a maximum of 193.000 Uber rides. A valuable insight that can be obtained from this graph is that people use Uber a lot during the afternoon and the early morning hours, probably going in and out of work. Also, by clicking on any figure of the histogram, the map will filter the data points according to the time selected.

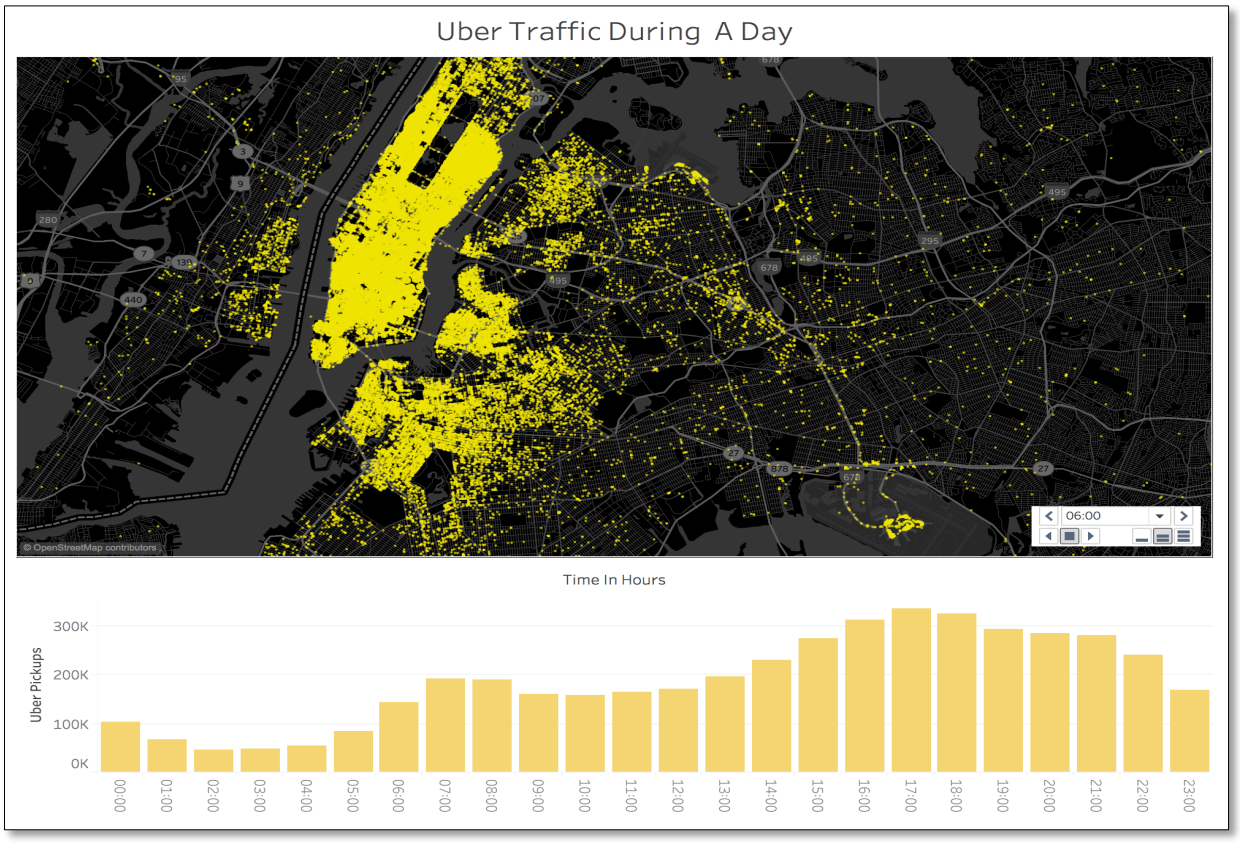

Figure 1: Uber Traffic In 1 Hour Intervals

Since we now got an idea of who Uber pickups distribute over different times of a day, another interesting insight can be observed by investigating what happens during different days. *Figure 2 presents* Uber trips during different days of the week and it can be observed that Uber is used more frequently during the middle of the week and more specifically reaching a peak point at Thursday with 755.145k trips. The histogram was initially created using Python, but then I thought it would more efficient to add an interactive map for better understanding of the data so a Tableau visualization was created containing both graphs. By pressing the ⇒ button on the tooltip included in the interactive map, the trips will be visualized live on the map for each day separately. Also, by clicking on any figure of the histogram, the map will filter the data points according to the day selected.

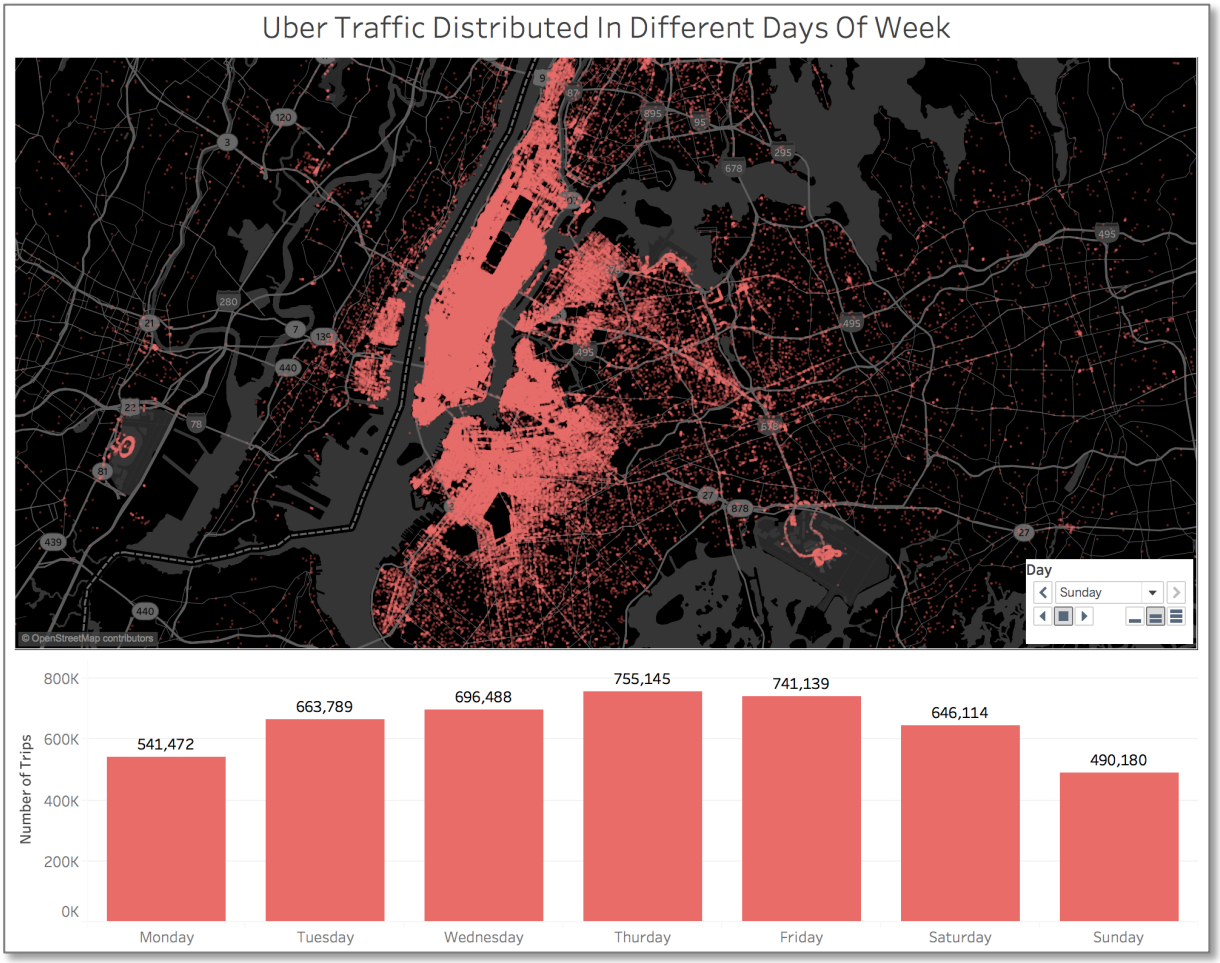

Figure 2: Demonstrating Distribution Of Uber Trips Over Days

Moving further with our analysis and wanting to understand better the information contained in the data set it would be useful to combine both Uber rides during different times and different days. A heatmap containing both elements would give us a broader knowledge of how Uber trips are distributed and for that purpose a heatmap was created and visualized in *Figure 3*. Also, the end user can further explore the data set choosing the 3D Surface option contained in the graph. For the creation of the graph, different days of week where plotted against different times using the number of Uber pickups as a metric obtained by a count function. The final data frame used for the creation of this visualization is displayed in *Table 3.* Some of the information gained from the heatmap validate or even improve the

accuracy of the ones obtained through the *Figures 1* and 2. The heatmap indicate that most people leave from home on weekdays between 6:00 and 8:59 and return between 17:00 and 18:59. Another valuable insight that can be derived from *Figure 3* is that people stay out late on Friday and Saturday nights leading to brighter than usual spots between 21:00 to 23:59 on Friday, 00:00 to 2:59 on Sunday. Also, it can be observed that people start their weekends later than usual.

hour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
dayofweek																					
0	6436	3737	2938	6232	9640	15032	23746	31159	29265	22197	...	28157	32744	38770	42023	37000	34159	32849	28925	20158	11811
1	6237	3509	2571	4494	7548	14241	26872	36599	33934	25023	...	34846	41338	48667	55500	50186	44789	44661	39913	27712	14869
2	7644	4324	3141	4855	7511	13794	26943	36495	33826	25635	...	35148	43388	50684	55637	52732	47017	47772	44553	32868	18146
3	9293	5290	3719	5637	8505	14169	27065	37038	35431	27812	...	36699	44442	50560	56704	55825	51907	51990	51953	44194	27764
4	13716	8163	5350	6930	8806	13450	23412	32061	31509	25230	...	36206	43673	48169	51961	54762	49595	43542	48323	49409	41260

Table 3: Data Frame Used For The Creation Of The Heatmap

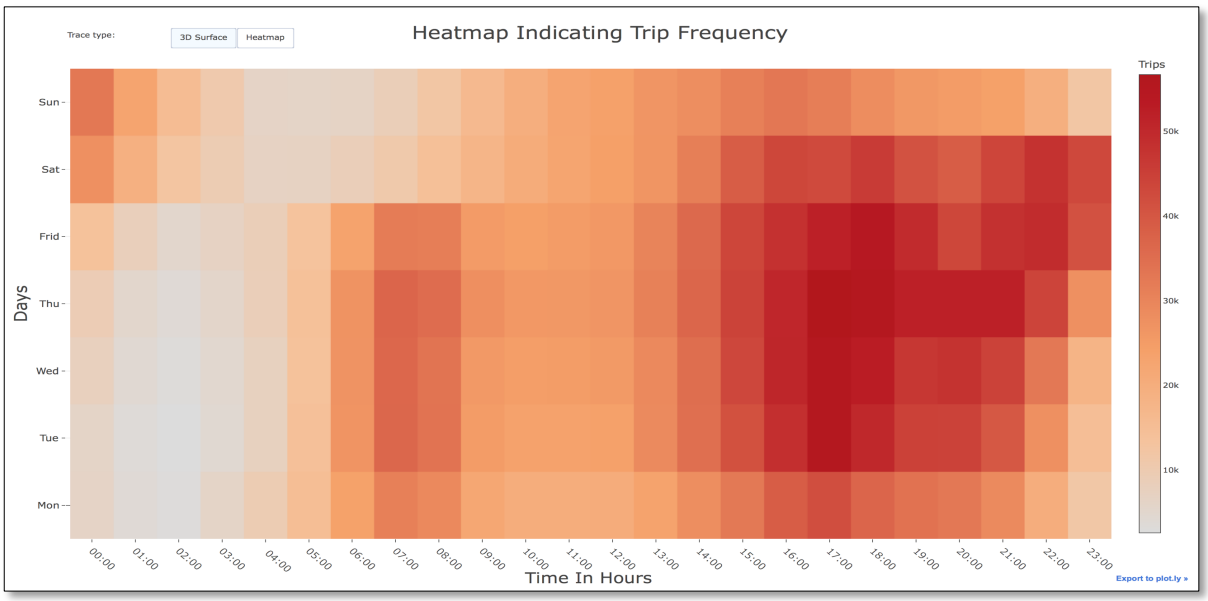

Figure 3: Heatmap Displaying Number of Trips During Different Time/Days

Another interesting way to visualize how the number of Uber trips are distributed over different days and time is using a point plot. For the creation of this graph the data frame previously created and displayed in *Table 3* was used with some modifications. Microsoft Excel was used to transpose the data set and Pandas to rename the columns and the index row the output of this modifications is displayed in *Table 4*. Now that the necessary changes have been made each day (row) was used to plot different days, the different colours used in the visualization are colour blind friendly. In the created graph there is a filter option where the user can choose which day of the weeks he wants information about. The final visualization is demonstrated in *Figure 4*. Analysing the graph, most of the trends presented previously in the heatmap are validated, most of the trends representing the weekdays follow similar paths indicating the people's daily program but there is a wide variation in the afternoon - night usage of Uber. Saturday and Friday have far more trips during the afternoon and night times, which was expected. Also, lines representing Saturday and Sunday are closely related, except from the afternoon times, where people tend to use Uber a lot more during Saturday which is normal because Sunday is a day-off for the majority of the population.

	Monday	Tuesday	Wedenday	Thursday	Friday	Saturday	Sunday	Time
0	6436	6237	7644	9293	13716	27633	32877	0
1	3737	3509	4324	5290	8163	19189	23015	1
2	2938	2571	3141	3719	5350	12710	15436	2
3	6232	4494	4855	5637	6930	9542	10597	3
4	9640	7548	7511	8505	8806	6846	6374	4

Table 4: Data Frame Used For The Point Plot

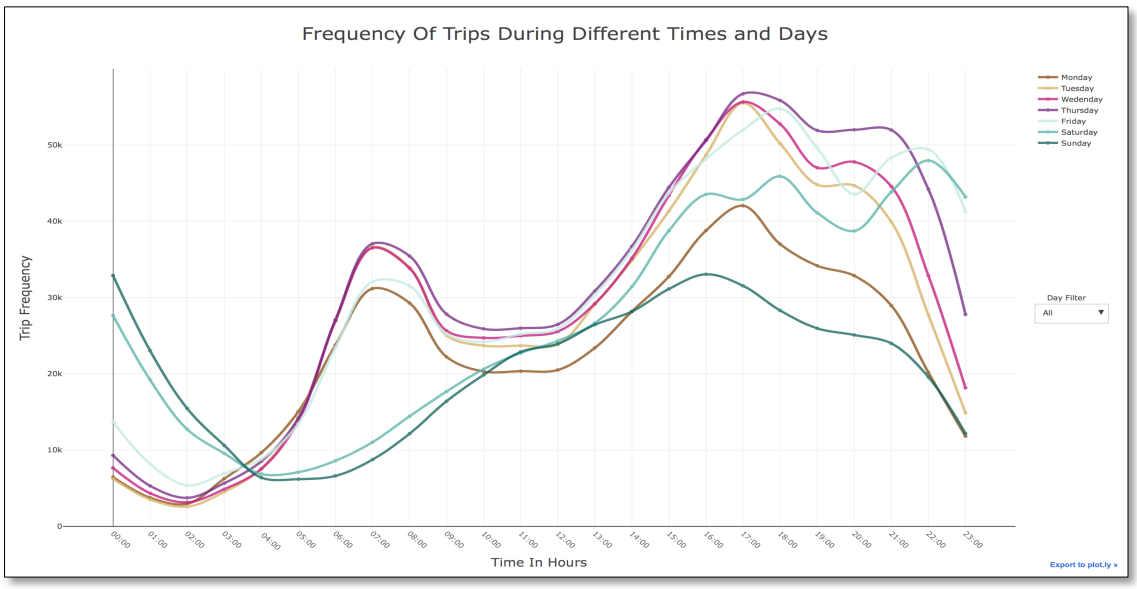

Figure 4 : Point Plot Displaying Number of Trip During Different Times/Days

Until now we have investigated how Uber tips are distributed by various days and times, now it can be useful to understand what happens with the location of the pickups. By plotting Latitude combined with Longitude and the number of trips, it can be observed that the vast majority of the trips are located in the centre and the suburbs of the Manhattan area.

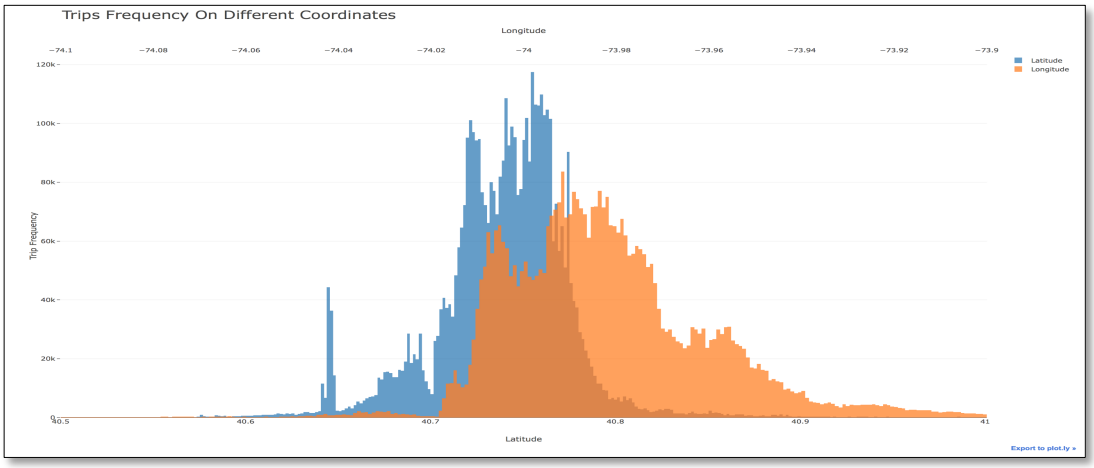

Figure 5: Number Of Trips On Different Coordinates

3 Temperature Analysis

Having a good understanding how Uber trips are distributed over various Times, Days and Location, it's time to move a step further in our analysis and investigate for any association between Uber's usage and temperature. Using the data set that contains combined data of Uber rides and weather conditions the number of trips are plotted against different temperatures using Tableau 10.5.0 and the result can be displayed at *Figure* 6. It can be observed that high number of trips occur during mid-range temperatures reaching a maximum of 288,500 trips at 37 degrees. Also, it must be noted that that during extreme temperatures the numbers of trips is actually low.

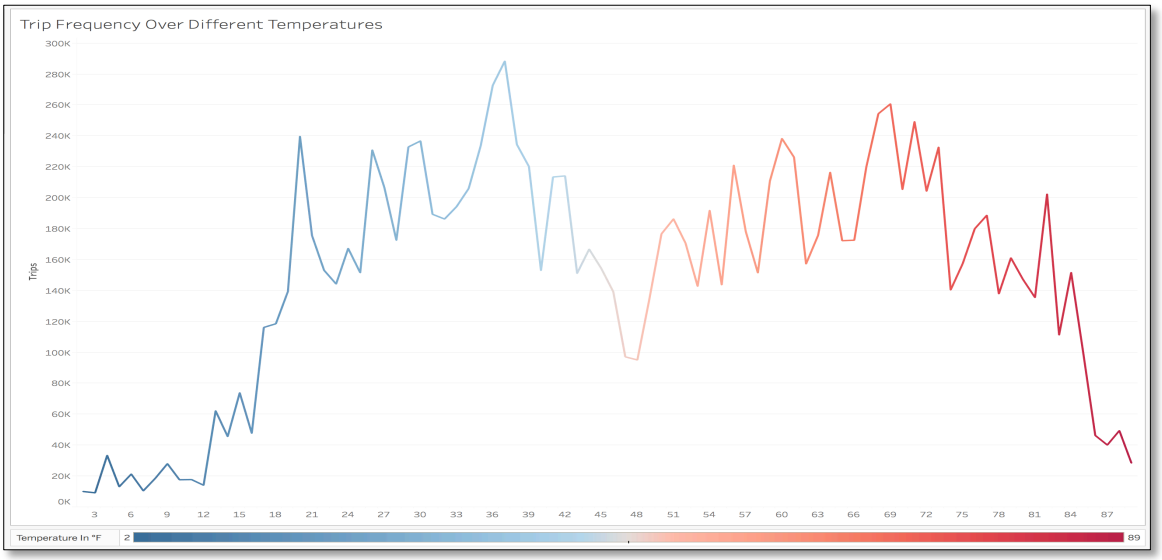

Figure 6: Number Of Uber Trips Over Different Temperatures

3.1 Spearman Rank Correlation

The Spearman rank-order correlation will be used in order to test if there is any association between the number of trips and the temperature measurements. The Spearman rank correlation coefficient is a nonparametric measure of rank correlation that assesses how well the relationship between two variables can be described using a monotonic function. A perfect Spearman correlation will output a +1 or -1 depending on the type on association, as closer it gets to zero the less the relationship between our two variables. Therefore, before moving to the actual test itself the assumption of homoscedasticity should be checked. In order to check our data for homoscedasticity the Lavene's test for equal variances was performed and the result was:

```
LeveneResult(statistic=6662.946618158333, pvalue=0.0)
```

meaning that we violate the assumption of homoscedasticity since $p < 0.05$ and the Spearman rank correlation test cannot be used on our data.

3 Cluster Analysis

Until this point we have analysed thoroughly information contained in our data set but in order to reach into actionable conclusions a quantitative comparison between different regions of New York city must be made. Since we have geographical information (latitude, longitude) in our data set a clustering method will be used in order to observe Uber's trips around the city, and search for particular hot spots.

3.1 K-Means and DBSCAN

The K-Means algorithm is likely the most common clustering algorithm. But for our application which involves spatial data, the DBSCAN algorithm is far more superior. The k-means algorithm groups N observations into k clusters. The reason why k-means is not an ideal algorithm for latitude-longitude spatial data is because it minimizes the variance and not the geodetic distance. Also, with k-means, locations that are near to each other will be over-represented because the initial random selection to seed the k-means algorithm would select them multiple times. Thus, more rows near a given location in the data set means a higher probability of having more rows selected randomly for that location. Given our data, most of our locations are fairly close to each other so an accurate clustering procedure is of vital importance. The k-means algorithm would still work but it would produce poor results. (Boeing, 2018)

Instead the DBSCAN algorithm will be used which clusters spatial data points based on: the physical distance from each point, and a minimum cluster size. This algorithm is far more robust for spatial latitude-longitude data according to Geoff Boeing.

3.2 Spatial Data Clustering With DBSCAN

An implementation of DBSCAN clustering method will be used through the Scikit-Learn. DBSCAN is a density-based clustering algorithm that takes a set of points in some space and groups together those points that are closely packet together (points with many neighbours) it also marks as outliers' points that lie alone in low-density regions.

Specifically, for our example DBSCAN will cluster together Latitude and Longitude points and create a cluster. The minimum number of neighbours for a group to qualify as a cluster as well as the maximum distance to make two individual points count as neighbours should be predefined. Then the algorithm will sort the points into groups which meet the criteria and discard all the outliers. After the clusters are created the MultiPoint class from the Shapely package is used in order to calculate the cluster centroids. For the purposes of our analysis the clustering process is optimized in order to find the hot spots that have minimum 200 pickups with maximum distance 0.07km from one another at Mondays after 18:00. After DBSCAN algorithm creates the clusters given the parameters defined, it returns the identified clusters:

```
Number of clusters: 47
```

The identified clusters are saved in a CSV file and after they have been transposed using Microsoft Excel a map was created using Tableau 10.5.0. In the interactive map the end user can identify the clusters created with a colour differentiation depending on the number of trips identified near that

location. That makes it really easy to interpret what cluster has the most pickups, and the actual location of it.

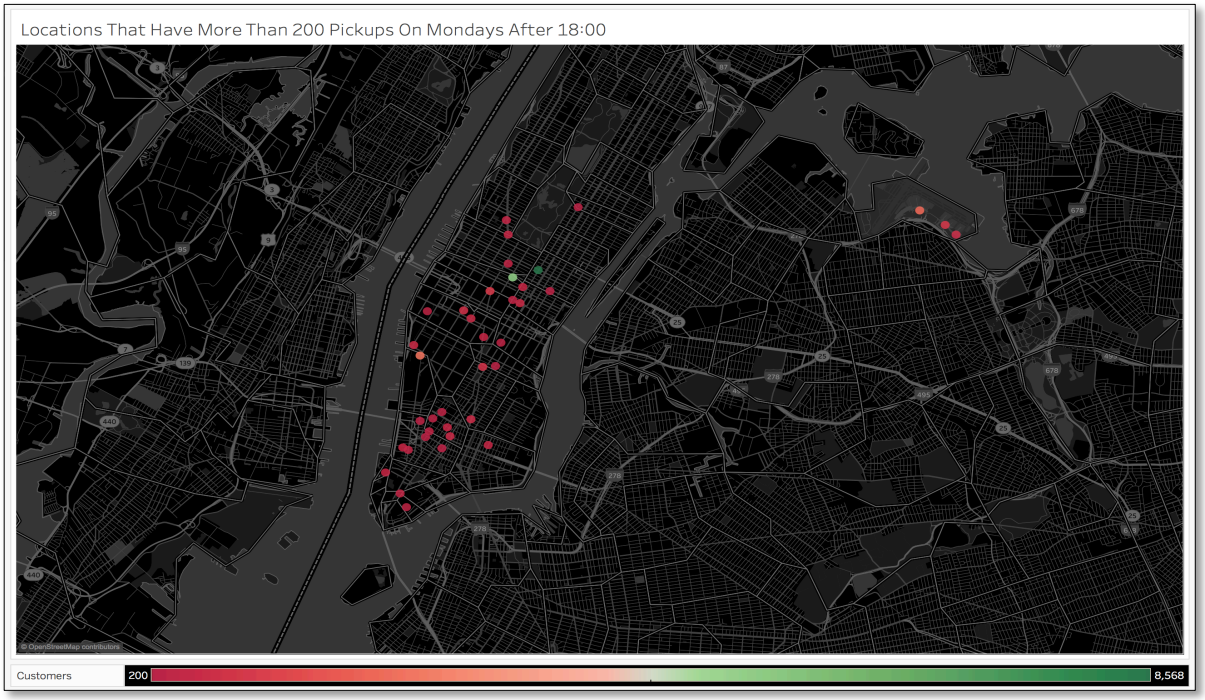

Figure 7: DBSCAN Output Clusters Plotted

4 Conclusion

For the majority of the data-pre-processing and the analysis, the programming language Python was used. More specifically the python library Pandas was used for the data handling and formatting, and the Plotly API library for the creation of the visualizations. Initially most of the graphs where created in Matplotlib and Seaborn libraries, but during the programming and testing process I found out that these libraries lack of interactive elements. After thorough research I concluded that the Plolty API library can output the desired results. Wanting to further evolve my visualizations some of them where combined with maps plotted using the Tableau software which I found particularly user friendly.

Moving to the actual outcomes of this study. During the explanatory analysis a lot of patterns concerning Uber's usage where distinguished. A few noted patterns distinguished are as below:

- Manhattan is found to be the busiest are for Uber traffic.
- Of all the days of the week, Thursday is found to be busiest.
- Weekday and Weekend trip habits of people living in NYC are clearly distinguishable.

Moving to the weather analysis the dis:

- Temperature doesn't seem to affect Uber's usage according to our study, with maximum number of trips happening during normal temperatures.
- During extreme temperatures the number of trips is low.

As far as the cluster analysis is concerned:

- The DBSCAN algorithm successfully identified hot-spots with the given parameters and the interactive map plotted can be really useful for professional Uber drivers.

Most of the insights derived from this study can be used in the design of incentive programs to Uber drivers. The scope of the study can be extended by analysing at least a year worth of data which will bring further insights about seasonality. Also, a predictive model could be introduced and in combination with the clustering algorithm, a more accurate result of Uber's demand could be obtained.

References

[1] Boeing, G. (2018) Clustering to Reduce Spatial Data Set Size, SSRN Electronic Journal.

[2] Omohundro, S. M. (1989) Five Balltree Construction Algorithms.

[3] Celebi, M. E. (2015) Partitional Clustering Algorithms.

[4] McKinney, W. (2013) Python For Data Analysis. Beijing; Farnham: O'Reilly.

[5] Data Society. (2016). Uber Pickups In NYC. Retrieved From https://data.world/data-society/uber-pickups-in-nyc.

[6] FiveThirtyEight. (2017). *Uber Pickups in New York City*. Available At: https://www.kaggle.com/fivethirtyeight/uber-pickups-in-new-york-city.

[7] Yannis Pappas. (2017). EDA on Uber's Ridership. Available At: https://www.kaggle.com/yannisp/eda-on-uber-s-ridership.

[8] Python Core Team (2015). Python: A dynamic, open source programming language. Python Software Foundation. Available at: https://www.python.org/.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. D. (2011) 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research*, 12, pp. 2825–2830.

[10] Plotly Technologies Inc. (2015) Collaborative Data Science. Available At: https://plot.ly.

[11] Wes McKinney. (2010). Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference. p.51-56