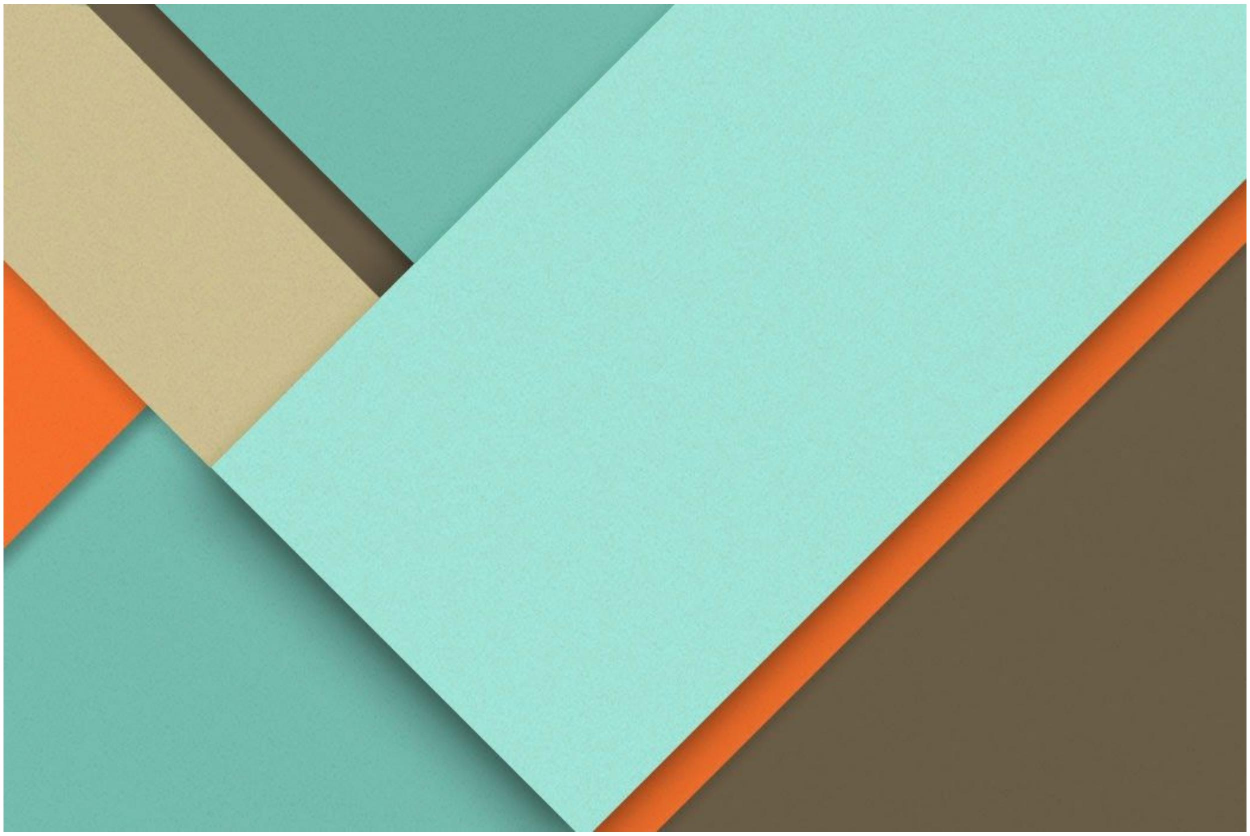


Battle of Neighbourhoods

Capstone Project

02/14/2020

Kalesha Mohammad

Overview

Tokyo is one of the largest cities in the world and it is well known for its culture and business. It's a multicultural city. Lots of indian/south aisan population lives there. And in recent times Indian cuisine has picked up its popularity. It will be a good business opportunity to setup an Indian restaurant in a good neighborhood.

Goal

1. Goal of the project is to find a good location in Tokyo city to setup an indian restaurant
2. To run a good restaurant there should be a good footfall around the locality/neighborhood

Data Acquisition and Cleaning

Data is acquired from Kaggle datasets onFoursquare NYC, Tokyo check ins. Here is the link for the dataset.

https://www.kaggle.com/chetanism/foursquare-nyc-and-tokyo-checkin-dataset

Dataset:

The dataset contains following columns

- userId
- venueId
- venueCategoryId
- venueCategory
- latitude
- longitude
- timezoneOffset
- utcTimestamp

Cleanup

First of all we need to find out the Venu category for Indian Restaurant(4bf58dd8d48988d10f941735). As this data is related to checkins we will have multiple entries related to a single venueId. Overall we have 2639 rows related to Indian restaurant.

	userId	venueId	venueCategoryId	venueCategory	latitude	longitude	timezoneOffset	utcTimestamp
1077	61	4e167b47483b3ee57cde7cb1	4bf58dd8d48988d10f941735	Indian Restaurant	35.585347	139.725556	540	Wed Apr 04 03:04:31 +0000 2012
1226	577	4b6cdfa0f964a520c45a2ce3	4bf58dd8d48988d10f941735	Indian Restaurant	35.668827	139.651013	540	Wed Apr 04 03:35:09 +0000 2012
1302	2027	4bd113bfcaff95212c3cd0f0	4bf58dd8d48988d10f941735	Indian Restaurant	35.673047	139.795044	540	Wed Apr 04 03:49:23 +0000 2012
1381	160	4c207c17920076b0e543c6e9	4bf58dd8d48988d10f941735	Indian Restaurant	35.632160	139.712407	540	Wed Apr 04 04:05:47 +0000 2012
1451	756	4c15a49082a3c9b6ed38fff8	4bf58dd8d48988d10f941735	Indian Restaurant	35.697391	139.759654	540	Wed Apr 04 04:19:45 +0000 2012
...	...	...	...	...	...	...	...	...
573098	255	4b738418f964a5202cb22de3	4bf58dd8d48988d10f941735	Indian Restaurant	35.700225	139.774401	540	Thu Feb 14 10:14:25 +0000 2013
573101	473	4b738418f964a5202cb22de3	4bf58dd8d48988d10f941735	Indian Restaurant	35.700225	139.774401	540	Thu Feb 14 10:14:35 +0000 2013
573196	1352	50fa90e1e4b0ba413b561131	4bf58dd8d48988d10f941735	Indian Restaurant	35.645962	139.669761	540	Thu Feb 14 10:36:54 +0000 2013
573208	779	4b57ca50f964a520524128e3	4bf58dd8d48988d10f941735	Indian Restaurant	35.713826	139.704385	540	Thu Feb 14 10:39:23 +0000 2013
573469	587	4b89f610f964a520e35832e3	4bf58dd8d48988d10f941735	Indian Restaurant	35.705770	139.577469	540	Thu Feb 14 11:50:53 +0000 2013

2639 rows × 8 columns

We need to get the unique VenuIDs to identify all the Indian restaurants located in Tokyo. There are 265 unique venueIds related to indian restaurants. It means we have 265 Indian restaurants located in Tokyo.

	userId	venueId	venueCategoryId	venueCategory	latitude	longitude	timezoneOffset	utcTimestamp
1226	577	4b6cdfa0f964a520c45a2ce3	4bf58dd8d48988d10f941735	Indian Restaurant	35.668827	139.651013	540	Wed Apr 04 03:35:09 +0000 2012
4643	2151	4f407b37e4b0085fed87706f	4bf58dd8d48988d10f941735	Indian Restaurant	35.715560	139.672422	540	Thu Apr 05 08:25:21 +0000 2012
5257	1963	4f3490a3e4b0993aec906d11	4bf58dd8d48988d10f941735	Indian Restaurant	35.777019	139.723455	540	Thu Apr 05 10:11:38 +0000 2012
5432	1474	4bee6b7fd355a5936cde0a60	4bf58dd8d48988d10f941735	Indian Restaurant	35.697160	139.785450	540	Thu Apr 05 10:53:03 +0000 2012
5576	245	4b9b0ec1f964a52069ef35e3	4bf58dd8d48988d10f941735	Indian Restaurant	35.653552	139.542270	540	Thu Apr 05 11:47:45 +0000 2012
...	...	...	...	...	...	...	...	...
564195	855	5023361ce4b0e522d483ab91	4bf58dd8d48988d10f941735	Indian Restaurant	35.674428	139.793740	540	Mon Feb 11 03:34:04 +0000 2013
564395	54	4bad9d28f964a5200f5f3be3	4bf58dd8d48988d10f941735	Indian Restaurant	35.707111	139.666830	540	Mon Feb 11 04:50:50 +0000 2013
564635	277	51188b7de4b0261e059a3fe5	4bf58dd8d48988d10f941735	Indian Restaurant	35.711858	139.810148	540	Mon Feb 11 06:11:28 +0000 2013
565655	921	4d4d08c49ee1a35df25621df	4bf58dd8d48988d10f941735	Indian Restaurant	35.653583	139.547428	540	Mon Feb 11 10:26:00 +0000 2013
571794	1830	4c22cb5e7e85c9285bb1bc21	4bf58dd8d48988d10f941735	Indian Restaurant	35.680311	139.762211	540	Thu Feb 14 02:42:00 +0000 2013

265 rows × 8 columns

```
df_ind=df_ind.reset_index()
```

We are mostly interested in venueId, venueCategoryId, venueCategory,latitude,longitude. Using this data we will be able to locate the Indian restaurants and their locations.

Methodology

I. Foursquare API

Using the foursquare API we got additional details about venues listed in the above section. This additional data will give us more info about the venue.

The following API is used to get the additional details about the venue:

https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.format(VenueId,CLIENT_ID, CLIENT_SECRET,VERSION)

Filtered the data for the following columns:

- 'Name',
- 'Categories',
- 'location.lat',
- 'Location.lng',
- 'verified',
- 'dislike',
- 'Rating',
- 'stats.tipCount',
- 'price.tier',
- 'price.message',
- 'Likes.count',
- 'likes.groups',
- 'beenHere.count'

After normalizing the data looks like this

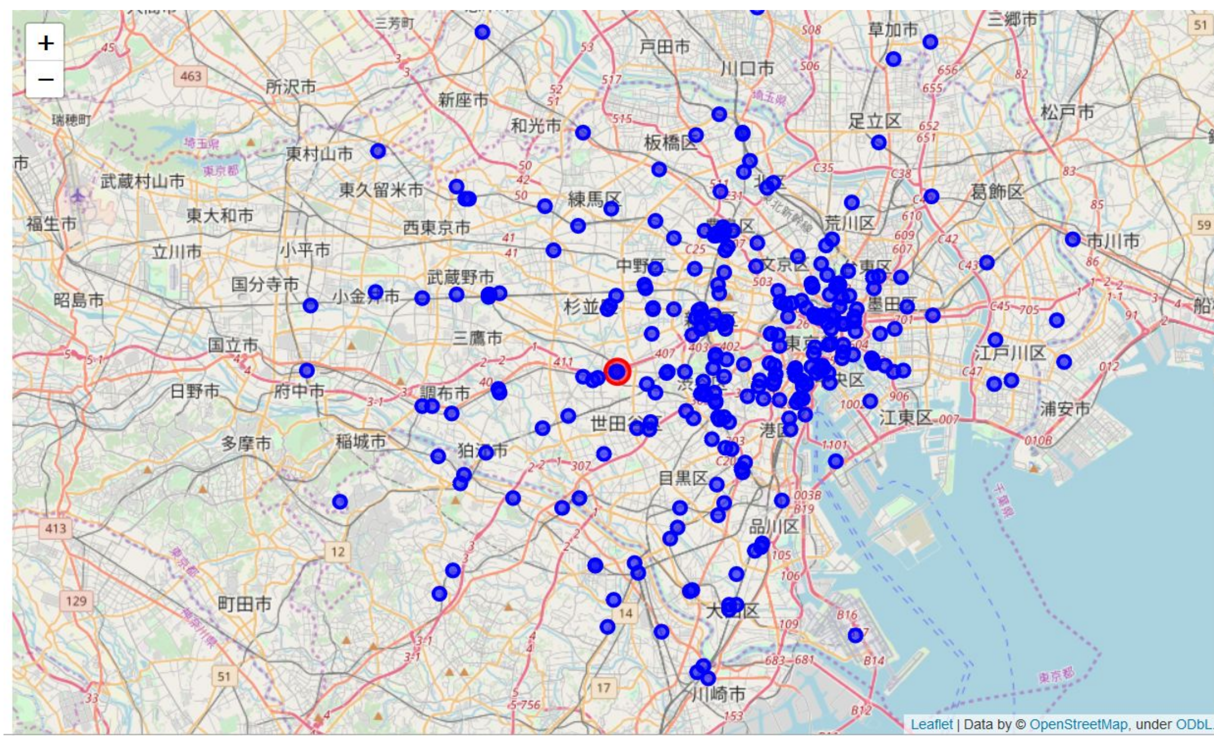
```
In [80]: final_data
```

Out[80]:

	name	categories	lat	lng	verified	dislike	rating	tipCount	tier	message	count	groups	count
0	コーヒヌール	Indian Restaurant	35.668740	139.651013	False	False	NaN	5	2.0	Moderate	7	[{'type': 'others', 'count': 7, 'items': [{'id...	0
0	CHAMI'S CURRY (チャミスカレー)	Japanese Curry Restaurant	35.715756	139.672343	False	False	NaN	2	NaN	NaN	4	[{'type': 'others', 'count': 4, 'items': [{'id...	0
0	Sitaara Diner (シターラ・ダイナー)	Indian Restaurant	35.777623	139.721100	False	False	6.8	2	2.0	Moderate	6	[{'type': 'others', 'count': 6, 'items': [{'id...	0
0	Stone (ストーン)	Yoshoku Restaurant	35.697209	139.785575	False	False	7.7	9	NaN	NaN	26	[{'type': 'others', 'count': 26, 'items': []}]	0
0	インド・ネパール料理 Raja 調布店	Indian Restaurant	35.653634	139.542344	False	False	6.0	8	2.0	Moderate	6	[{'type': 'others', 'count': 6, 'items': [{'id...	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
0	ダルハラ	Indian Restaurant	35.674670	139.793944	False	False	NaN	0	2.0	Moderate	4	[{'type': 'others', 'count': 4, 'items': [{'id...	0
0	Deep Jyoti (ディープジョティ)	Indian Restaurant	35.707202	139.666862	False	False	NaN	0	2.0	Moderate	1	[{'type': 'others', 'count': 1, 'items': [{'id...	0
0	華麗なるカレー 業平橋総本店	Indian Restaurant	35.711675	139.810153	False	False	NaN	2	2.0	Moderate	0	[]	0
0	牛たん処 い志井	Japanese Restaurant	35.653563	139.547401	False	False	8.2	2	2.0	Moderate	15	[{'type': 'others', 'count': 15, 'items': []}]	0
0	Restaurent & Pub SANGRIA (サングリア)	Indian Restaurant	35.680105	139.762026	False	False	6.8	9	2.0	Moderate	8	[{'type': 'others', 'count': 8, 'items': [{'id...	0

265 rows × 13 columns

Using folium we can visualize the 256 restaurants spread



II. Data Analysis

The final data looks like the below:

Out[186]:

	Unnamed: 0	index	name	categories	lat	lng	verified	dislike	rating	tipCount	tier	message	count	groups	count.1
0	0	0	コーヒヌール	Indian Restaurant	35.668740	139.651013	False	False	NaN	5	2.0	Moderate	7	[{'type': 'others', 'count': 7, 'items': [{'id...	0
1	1	0	CHAMI'S CURRY (チャミスカレー)	Japanese Curry Restaurant	35.715756	139.672343	False	False	NaN	2	NaN	NaN	4	[{'type': 'others', 'count': 4, 'items': [{'id...	0
2	2	0	Sitaara Diner (シターラ・ダイナー)	Indian Restaurant	35.777623	139.721100	False	False	6.8	2	2.0	Moderate	6	[{'type': 'others', 'count': 6, 'items': [{'id...	0
3	3	0	Stone (ストーン)	Yoshoku Restaurant	35.697209	139.785575	False	False	7.7	9	NaN	NaN	26	[{'type': 'others', 'count': 26, 'items': []}]	0
4	4	0	インド・ネパール料理 Raja 調布店	Indian Restaurant	35.653634	139.542344	False	False	6.0	8	2.0	Moderate	6	[{'type': 'others', 'count': 6, 'items': [{'id...	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
260	260	0	ダルハラ	Indian Restaurant	35.674670	139.793944	False	False	NaN	0	2.0	Moderate	4	[{'type': 'others', 'count': 4, 'items': [{'id...	0
261	261	0	Deep Jyoti (ディープジョティ)	Indian Restaurant	35.707202	139.666862	False	False	NaN	0	2.0	Moderate	1	[{'type': 'others', 'count': 1, 'items': [{'id...	0
262	262	0	華麗なるカレー業平橋総本店	Indian Restaurant	35.711675	139.810153	False	False	NaN	2	2.0	Moderate	0	[]	0
263	263	0	牛たん処 い志井	Japanese Restaurant	35.653563	139.547401	False	False	8.2	2	2.0	Moderate	15	[{'type': 'others', 'count': 15, 'items': []}]	0
264	264	0	Restaurent & Pub SANGRIA (サングリア)	Indian Restaurant	35.680105	139.762026	False	False	6.8	9	2.0	Moderate	8	[{'type': 'others', 'count': 8, 'items': [{'id...	0

We can observe from the above data there are rating, message, tier etc. Message and tier may not be an important factor for our analysis but rating is very important to determine the quality of the restaurant.

III. Missing data

Missing rating data will have a huge impact on final clustering, so we need to analyze how to incorporate the missing data.

First of all the missing data (NaN) has to be replaced with '0' so that we can do some data visualization. Ratings and likes count is considered for further data analysis.

```
In [298]: cdf.fillna(0,inplace=True)
          cdf

C:\Users\kmohammad\AppData\I
A value is trying to be set

See the caveats in the docum
-versus-a-copy
  **kwargs
```

Out[298]:

	rating	Likes_Count
0	0.0	7
1	0.0	4
2	6.8	6
3	7.7	26
4	6.0	6
...	...	...
260	0.0	4
261	0.0	1
262	0.0	0
263	8.2	15
264	6.8	8

265 rows × 2 columns

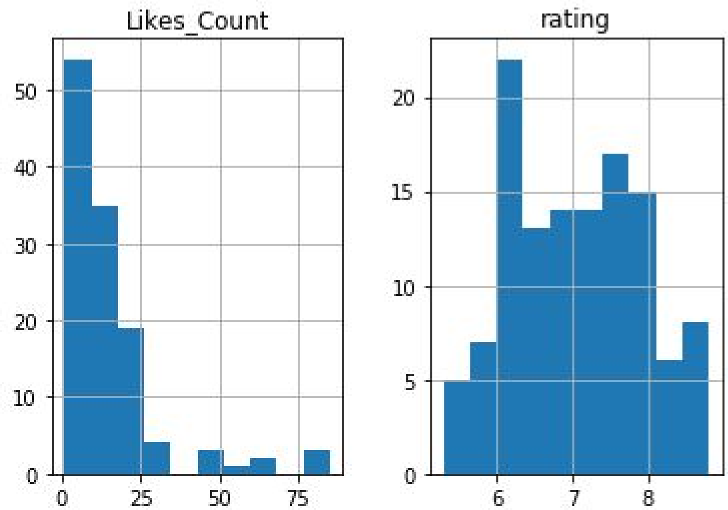
This data set with 265 rows is divided into two sets. One set with no rating and another set with rating.

Sets with no ratings have 144 rows and sets with ratings have 121 rows.

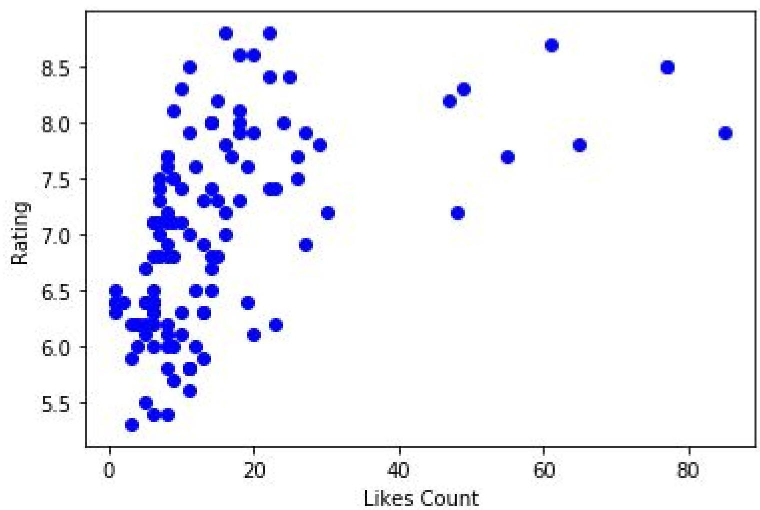
```
     rating  Likes_Count
0       0.0            7
1       0.0            4
6       0.0            3
7       0.0            0
9       0.0            2
..      ...          ...
257     0.0            1
259     0.0            2
260     0.0            4
261     0.0            1
262     0.0            0

[144 rows x 2 columns]
     rating  Likes_Count
2       6.8            6
3       7.7           26
4       6.0            6
5       7.9           20
8       8.3           10
..      ...          ...
250     5.6           11
251     8.0           14
258     6.4            6
263     8.2           15
264     6.8            8

[121 rows x 2 columns]
```

Data set with no rating is kept aside to predict the missing data.

Data set with ratings is considered for data visualization. Here is how the data is represented in plots



Next created a scatter plot between likes count and ratings. Here is how it looks

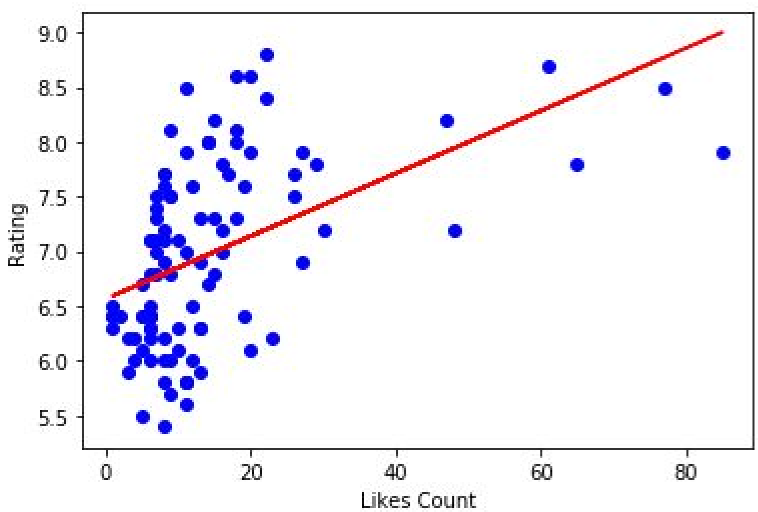


It is very clearly visible that there is a linear relationship between “likes count” and “rating”

IV. Linear Regression - Part 1

Linear regression can be applied to predict the missing ratings. Created train and test sets from the above data set and applied linear regression. Here are the results

Regression line looks like this:

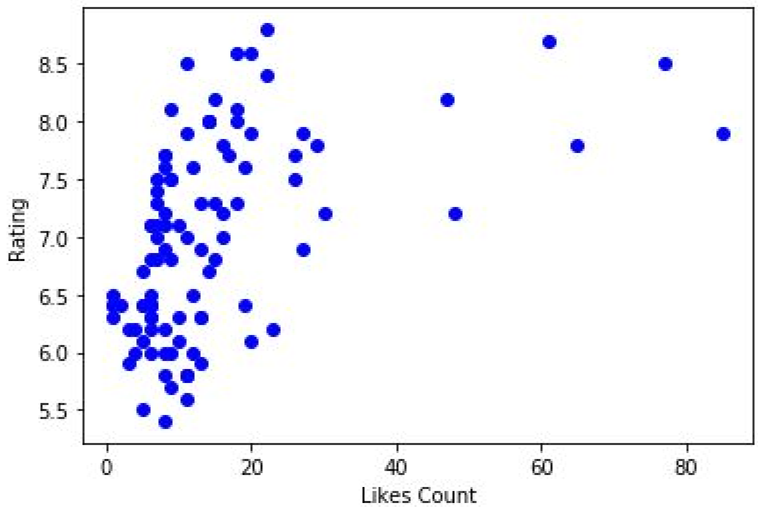


And here is the output of R2 values:

```
Mean absolute error: 0.65
Residual sum of squares (MSE): 0.65
R2-score: -1.34
```

Regression line as well as R2 values indicate that the predicted values are not good. Need some further data cleanup.

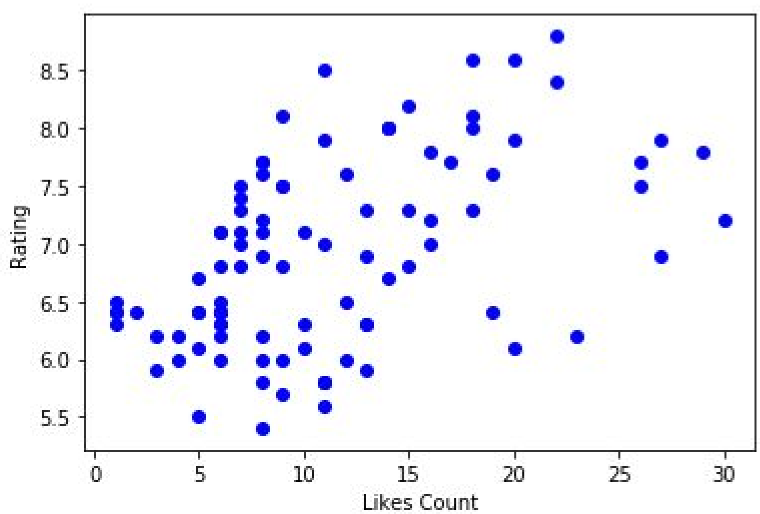
Lets draw the scatterplot again and see how it looks for deeper analysis:



One can observe that there are 6 values with above 40 likes count and seems to be away from the cluster. And these might be influencing the regression line drift away from the pack. So decided to drop the above 40 values and apply linear regression again.

V. Linear Regression - Part 2

Here is the scatter plot after dropping the outliers:

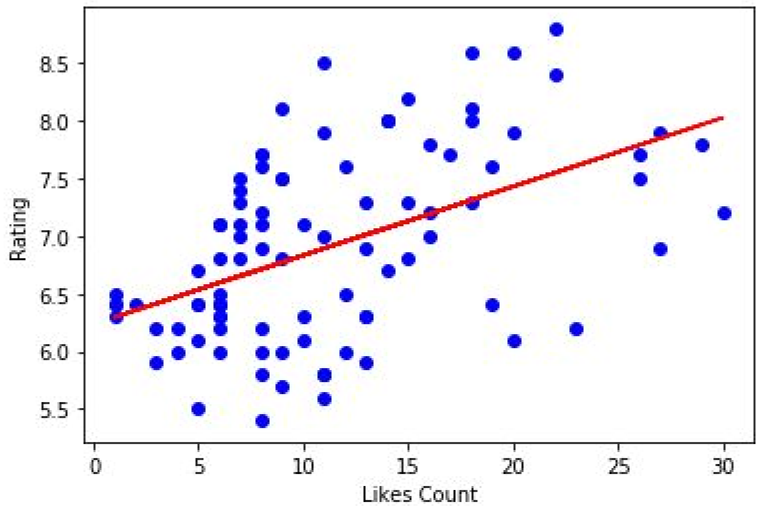



```
(94, 2)
```

Looks like a good group.

Applied linear regression on above data.

Here is the regression line:



Here are the R2 values:

```
print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_hat - test_y)))
print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_hat - test_y) ** 2))
print("R2-score: %.2f" % r2_score(test_y_hat , test_y) )
```

```
Mean absolute error: 0.75
Residual sum of squares (MSE): 0.92
R2-score: 0.23
```

Now the data looks good. Let's check actual and predicted values:

	Actual	Predicted
0	8.3	6.832519
1	5.4	6.593716
2	8.0	7.668331
3	7.1	6.772818
4	7.4	6.832519
5	6.8	7.071322
6	7.9	7.310126
7	8.3	9.160852
8	7.4	7.548929
9	8.4	7.728032
10	6.2	6.474314
11	7.4	7.071322
12	6.5	7.071322
13	7.7	9.519057
14	8.8	7.190724
15	5.3	6.414613
16	6.2	6.534015
17	8.5	10.832475
18	7.4	7.608630
19	6.1	6.713117
20	6.8	6.713117

Above is a satisfactory result.

We can go ahead and predict the missing values.

Here are the predicted missing values:

	rating	Likes_Count
0	6.653417	7
1	6.474314	4
6	6.414613	3
7	6.235511	0
9	6.354912	2
...	...	...
257	6.295212	1
259	6.354912	2
260	6.474314	4
261	6.295212	1
262	6.235511	0

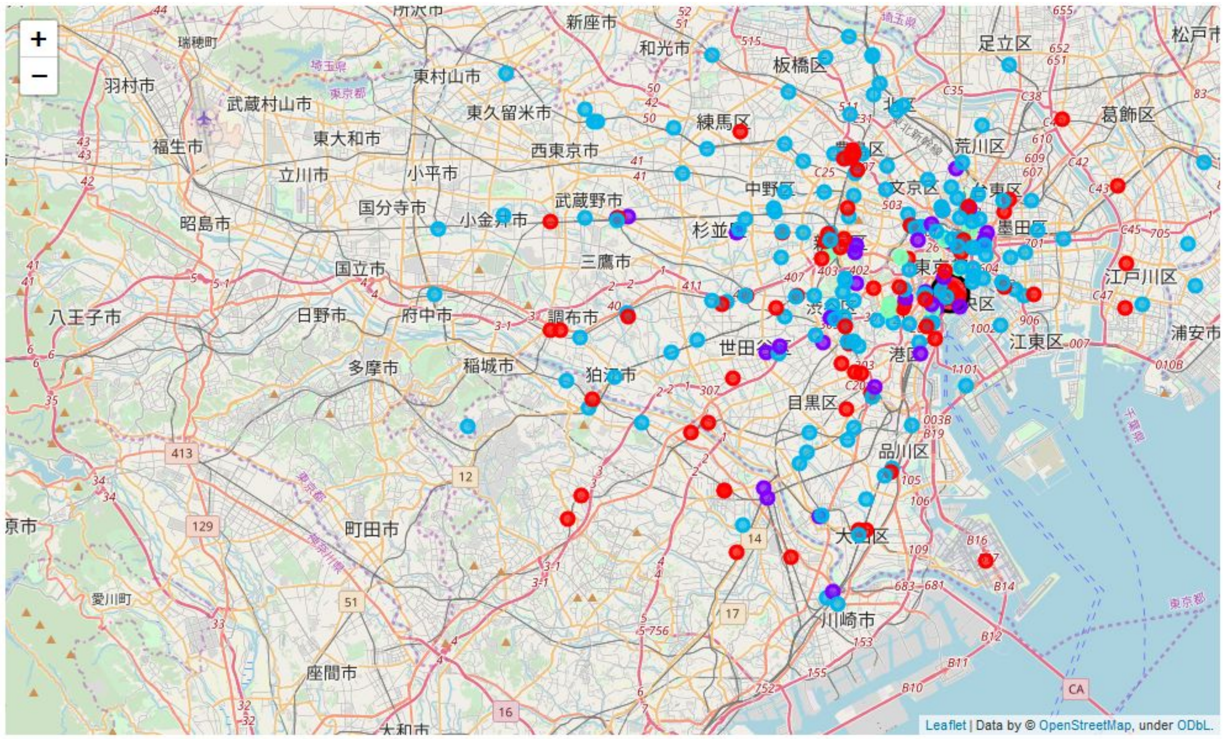
144 rows × 2 columns

Missing ratings are filled now. Thanks to linear regression.

Lets merge the above data and get ready for k_means clustering.

VI. K-Means clustering

After running k-means clustering we got the 5 clusters. Lets visualize on the map.



Great we have all the Indian restaurants clustered into 5 clusters.

Lest Analyze individual clusters.

Cluster1:

```
Cluster1.describe()
```

	lat	lng	rating	tipCount	tier	count
count	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000
mean	35.665373	139.706532	6.826973	5.971429	1.285714	9.514286
std	0.055543	0.076107	0.727551	2.812866	0.980134	3.183940
min	35.549179	139.542344	5.400000	1.000000	0.000000	3.000000
25%	35.635312	139.657696	6.300000	3.250000	0.000000	8.000000
50%	35.669329	139.710917	6.800000	6.000000	2.000000	9.000000
75%	35.700101	139.759346	7.300000	8.000000	2.000000	12.000000
max	35.860524	139.862495	8.500000	12.000000	3.000000	15.000000

Cluster2:

	lat	lng	rating	tipCount	tier	count
count	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000
mean	35.662558	139.716164	7.676777	8.620690	1.103448	21.551724
std	0.046285	0.055020	0.709904	3.052093	1.012240	4.420619
min	35.534758	139.579043	6.100000	3.000000	0.000000	16.000000
25%	35.646141	139.691734	7.300000	6.000000	0.000000	18.000000
50%	35.670416	139.722919	7.800000	8.000000	2.000000	20.000000
75%	35.694385	139.757106	8.026536	11.000000	2.000000	25.000000
max	35.726638	139.785575	8.800000	15.000000	2.000000	30.000000

Cluster3:

	lat	lng	rating	tipCount	tier	count
count	155.000000	155.000000	155.000000	155.000000	155.000000	155.000000
mean	35.687886	139.710403	6.325983	1.741935	1.600000	2.012903
std	0.053802	0.079564	0.211469	1.610996	0.802593	1.920454
min	35.529574	139.477709	5.300000	0.000000	0.000000	0.000000
25%	35.659573	139.676774	6.235511	0.000000	2.000000	0.000000
50%	35.691069	139.713790	6.295212	1.000000	2.000000	2.000000
75%	35.715718	139.765384	6.414613	3.000000	2.000000	3.000000
max	35.834330	139.906104	7.400000	7.000000	2.000000	7.000000

Cluster4:

	lat	lng	rating	tipCount	tier	count
count	10.000000	10.000000	10.000000	10.00000	10.000000	10.000000
mean	35.681480	139.739395	8.196085	21.40000	0.900000	61.300000
std	0.013342	0.026861	0.564022	9.31188	0.994429	14.111067
min	35.662304	139.698022	7.200000	9.00000	0.000000	47.000000
25%	35.669359	139.730707	7.825000	14.50000	0.000000	49.000000
50%	35.686161	139.736861	8.250000	19.50000	0.500000	58.000000
75%	35.689609	139.759972	8.500000	27.75000	2.000000	74.000000
max	35.698447	139.775764	9.160852	35.00000	2.000000	85.000000

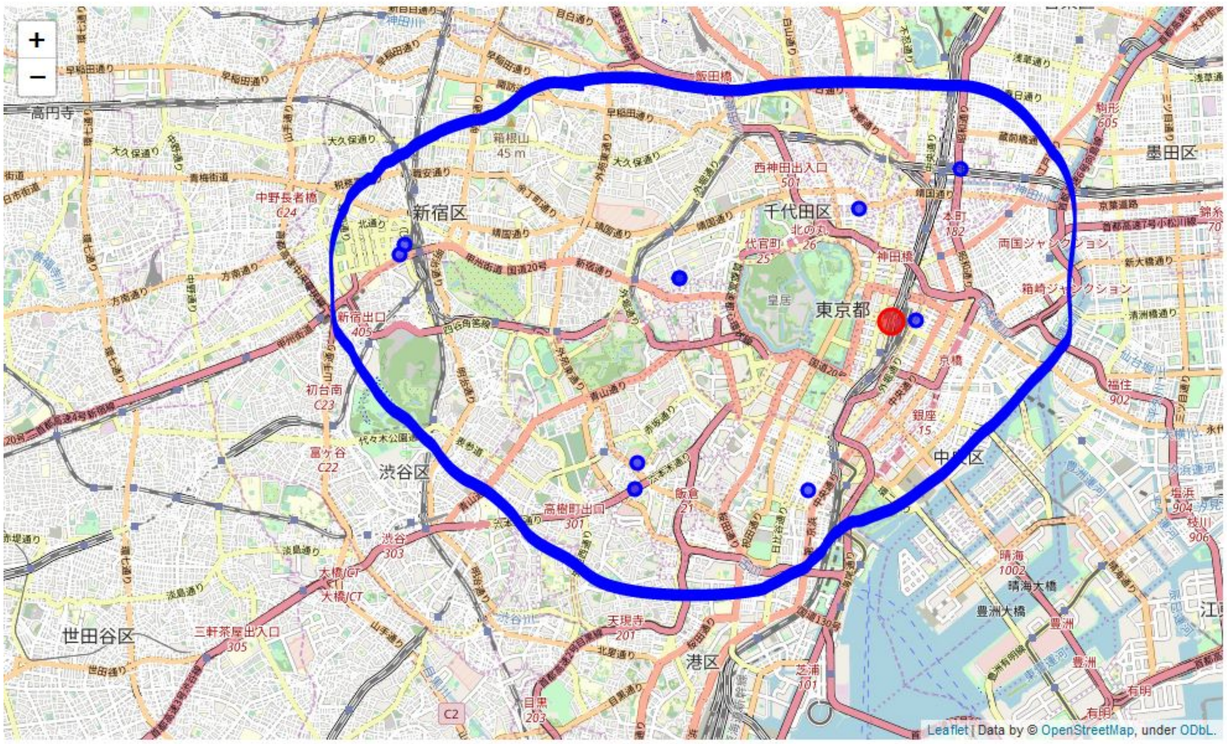
Cluster5:

	lat	lng	rating	tipCount	tier	count
count	1.00000	1.000000	1.0	1.0	1.0	1.0
mean	35.67009	139.764585	6.8	83.0	2.0	14.0
std	NaN	NaN	NaN	NaN	NaN	NaN
min	35.67009	139.764585	6.8	83.0	2.0	14.0
25%	35.67009	139.764585	6.8	83.0	2.0	14.0
50%	35.67009	139.764585	6.8	83.0	2.0	14.0
75%	35.67009	139.764585	6.8	83.0	2.0	14.0
max	35.67009	139.764585	6.8	83.0	2.0	14.0

Conclusion

From the above data we can infer that cluster 3 is least performing in terms of ratings, likes or tips. And Cluster 4 is top performing. Cluster 5 has only one restaurant with high number of likes count and average rating, thus we can consider it as an outlier.

Cluster4 is where we are going to set up our new Indian restaurant, which would give us good business. Lets visualize cluster 4 on the map.



Let's hunt for a lease space to setup our new Indian Restaurant. Good Luck and All the best.

By using various ML techniques like regression and clustering algorithms we were able to finalize a good place to setup an Indian Restaurant in Tokyo City.

Special acknowledgments to kaggle and Foursquare for providing the data for this project.