Battle of the Neighborhoods: Cluster analysis of Top 100 US cities by population

Coursera Data Science Capstone

Rich Timm

June 2, 2020

Problem

- Help someone decide on where to live, if moving cities
- Scope = Top 100 US cities by population
- Can use cities similar to a known city they either like or dislike to help inform the decision

Approach

- Web scrape list of Top 100 cities
- Pull Foursquare data on venues of these cities
- Prepare & join data for cluster analysis
- Explore cluster analysis parameters and impact on cluster error (elbow graph)
- Choose optimized cluster parameter and visualize results

Wikipedia scraping for data

	City	State	Population	% Change	Area sqmi	Pop density p/sqmi	Latitude	Longitude
0	New York[d]	New York	8336817	+1.98%	301.5 sq mi	28,317/sq mi	40.6635	-73.9387
1	Los Angeles	California	3979576	+4.93%	468.7 sq mi	8,484/sq mi	34.0194	-118.4108
2	Chicago	Illinois	2693976	−0.06%	227.3 sq mi	11,900/sq mi	41.8376	-87.6818
3	Houston[3]	Texas	2320268	+10.48%	637.5 sq mi	3,613/sq mi	29.7866	-95.3909
4	Phoenix	Arizona	1680992	+16.28%	517.6 sq mi	3,120/sq mi	33.5722	-112.0901
5	Philadelphia[e]	Pennsylvania	1584064	+3.80%	134.2 sq mi	11,683/sq mi	40.0094	-75.1333

Plotting Top 100 Cities by Population with Folium

Extracting Foursquare venues

- Radius = 32km (~20 miles) from city latitude/ longitude
- Limit = 2000 results, otherwise the API call would choke/fail
- Here is the size of the output:

Cool, the Foursquare call worked. It was a big one. Let's see how many results there were.

```
us_venues.shape
```

```
: (9907, 7)
```

Process the Venue data into dummies to quantify

	City	Accessories Store	Adult Boutique	Afghan Restaurant	African Restaurant	Airport	Airport Lounge	American Restaurant	Amphitheater	Antique Shop	...	Waterfront	Whisky Bar	Wine Bar	Wine Shop	Winery	Wings Joint	Women's Store	S
0	New York[d]	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
1	New York[d]	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
2	New York[d]	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
3	New York[d]	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	
4	New York[d]	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	

5 rows × 371 columns

Group the venue dummies by city

Now group the dummy sums by City to get the number of rows back to the 100 cities

```
In [241]: us_grouped = us_onehot.groupby('City').sum()
          us_grouped
```

Out[241]:

City	Accessories Store	Adult Boutique	Afghan Restaurant	African Restaurant	Airport	Airport Lounge	American Restaurant	Amphitheater	Antique Shop	Aquarium	...	Waterfront	Whisky Bar	Wine Bar	Wine Shop	Winery	Wi J
Albuquerque	0	0	0	0	0	0	3	0	0	0	...	0	0	0	2	0	
Anaheim	0	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	
Anchorage[p]	0	0	0	0	0	0	4	0	0	0	...	0	0	0	0	0	
Arlington	0	0	0	0	0	1	2	0	0	0	...	0	0	0	1	0	
Atlanta	0	0	0	0	0	0	1	0	0	1	...	0	0	0	1	0	
Aurora	0	0	0	0	0	0	2	0	0	0	...	0	0	0	2	0	
Austin	0	0	0	0	0	0	2	0	0	0	...	0	0	0	0	0	
Bakersfield	0	0	0	0	0	0	3	0	0	0	...	0	0	0	0	0	
Baltimore[m]	0	0	1	0	0	0	1	0	0	1	...	0	0	0	3	0	

```
In [242]: us_grouped.shape
```

```
Out[242]: (100, 370)
```

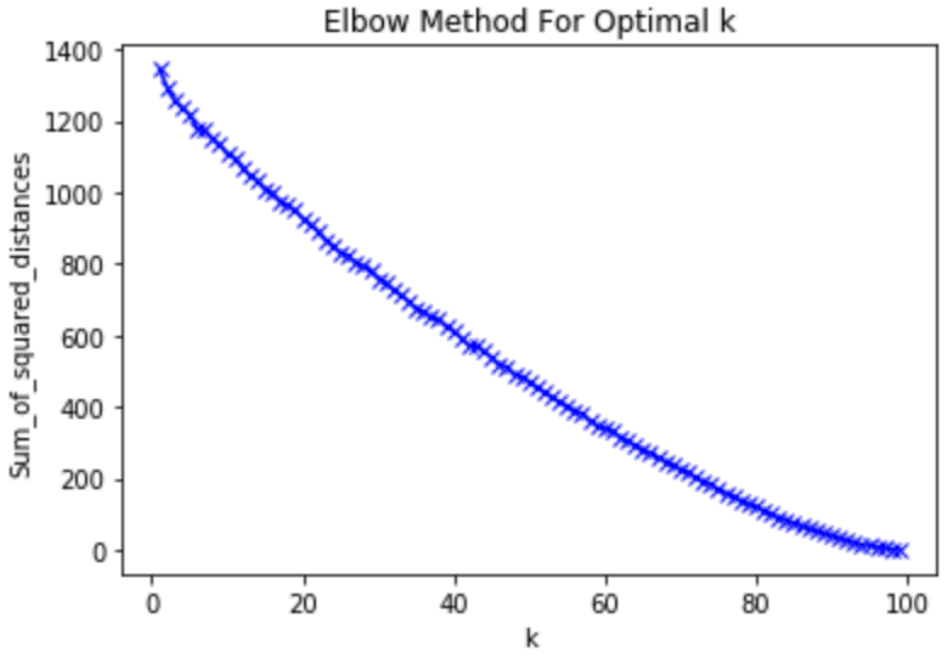
Join the two dataframes

City	State	Population	% Change	Area sqmi	Pop density p/sqmi	Latitude	Longitude	Accessories Store	Adult Boutique	Afghan Restaurant	...	Waterfront	Whisky Bar	Wine Bar	Wine Shop	Winery	Wings Joint
Albuquerque	New Mexico	560513	2.69	188.2	2972.0	35.1056	-106.6474	0	0	0	...	0	0	0	2	0	0
Anaheim	California	350365	4.19	50.0	7021.0	33.8555	-117.7601	0	0	0	...	0	0	0	0	0	0
Anchorage[p]	Alaska	288000	-1.31	1706.6	175.0	61.1743	-149.2843	0	0	0	...	0	0	0	0	0	0
Arlington	Texas	398854	9.14	95.8	4100.0	32.7007	-97.1247	0	0	0	...	0	0	0	1	0	1
Atlanta	Georgia	506811	20.67	133.5	3539.0	33.7629	-84.4227	0	0	0	...	0	0	0	1	0	0

5 rows × 377 columns

Scale data (not shown) and run Kmeans with different K values to compare sum of squared error

```
plt.plot(K, Sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distances')
plt.title('Elbow Method For Optimal k')
plt.show()
```



This should be an elbow plot, but it's not!!

Ended up choosing K=10 after playing around with the cluster map

Cluster results for KMeans, K=10