The Battle of the Neighborhoods

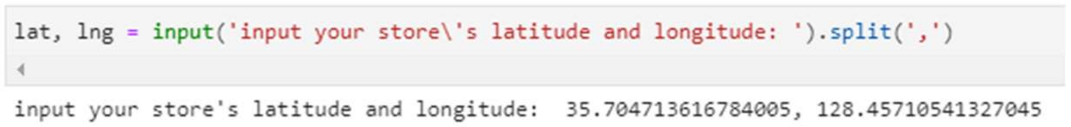
Contents

1. Introduction

- Taking a walk, we may notice that the shops in certain places change frequently. In this case, we can see that stores of the same type are already thriving. In order to prevent the catastrophe of missing some stores of same category, the theme of this project was selected. I am going to make a recommendation system for person who wants to be a shopkeeper. This machine learning model will help people to know what kind of category they should avoid.

2. Data

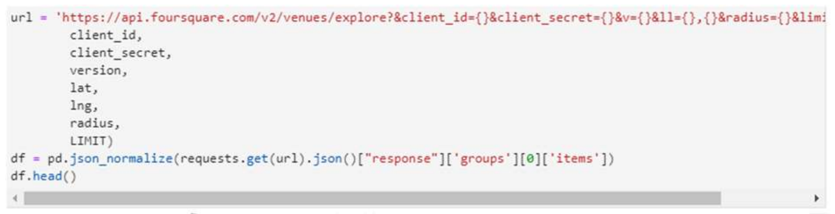
- We will use two kinds of data. First, the user's store's latitude and longitude are entered. Second, get store information near your store's location via foursquare api. I am going to set radius as 1000 meter, limit results to 100.
- **FIrst, get latitude and longitude of the store.**



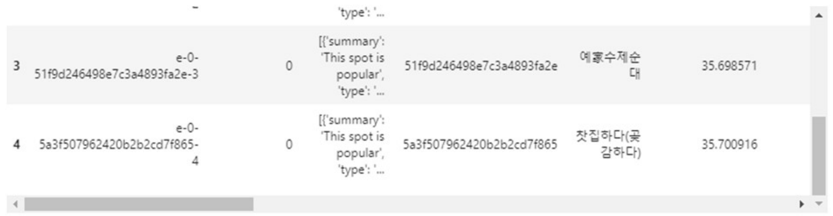
```
lat, lng = input('input your store\'s latitude and longitude: ').split(',')
```

```
input your store's latitude and longitude:  35.704713616784005, 128.45710541327045
```

- **Second, using foursquare api, get a dataframe of stores nearby**



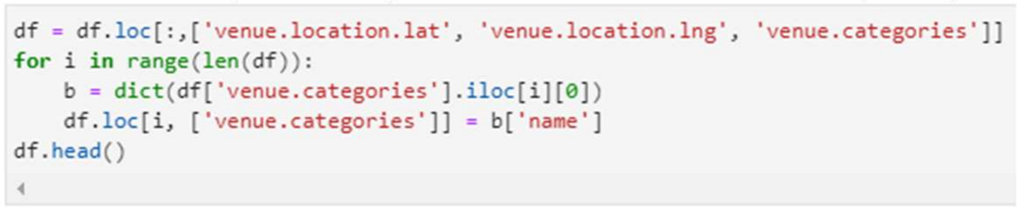
```
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limi
        client_id,
        client_secret,
        version,
        lat,
        lng,
        radius,
        LIMIT)
df = pd.json_normalize(requests.get(url).json()["response"]['groups'][0]['items'])
df.head()
```



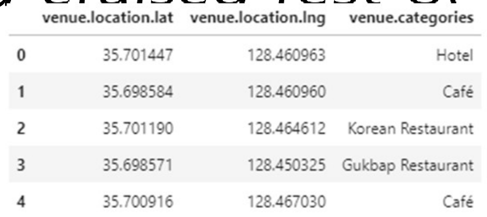
	-		'type': '...			
3	e-0-51f9d246498e7c3a4893fa2e-3	0	[{'summary': 'This spot is popular', 'type': '...	51f9d246498e7c3a4893fa2e	예家수제순대	35.698571
4	e-0-5a3f507962420b2b2cd7f865-4	0	[{'summary': 'This spot is popular', 'type': '...	5a3f507962420b2b2cd7f865	찻집하다(곶감하다)	35.700916

3. Methodology

- Sice algorithm only needs location and category, I left latitude, longitude and category data and eraised rest of

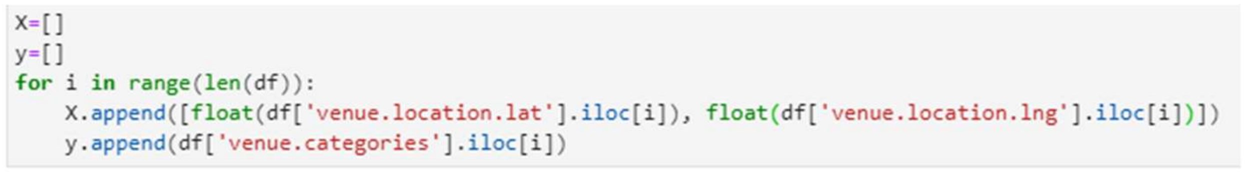


```
df = df.loc[:,['venue.location.lat', 'venue.location.lng', 'venue.categories']]
for i in range(len(df)):
    b = dict(df['venue.categories'].iloc[i][0])
    df.loc[i, ['venue.categories']] = b['name']
df.head()
```



	venue.location.lat	venue.location.lng	venue.categories
0	35.701447	128.460963	Hotel
1	35.698584	128.460960	Café
2	35.701190	128.464612	Korean Restaurant
3	35.698571	128.450325	Gukbap Restaurant
4	35.700916	128.467030	Café

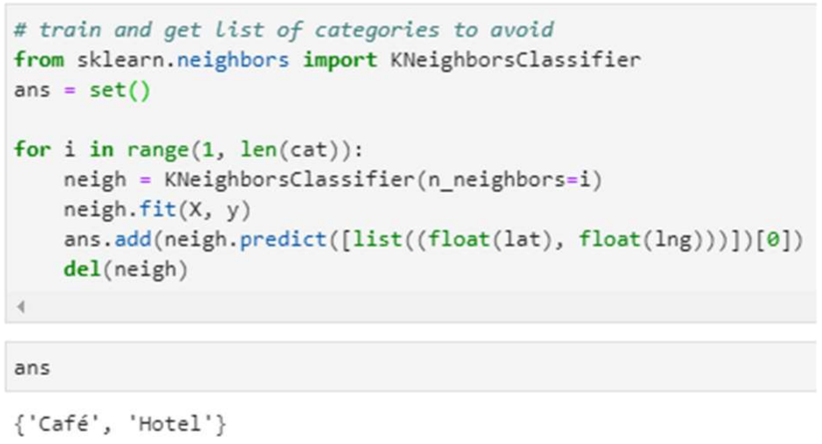
- First, I made a set of categories so I can get max number of K. K starting from 1 until length of set made right before, algorithm will determine what category will that place will fit in. Being classified in a specific category means that even if there is a store in that category, it does not have any advantage.



```
X=[]
y=[]
for i in range(len(df)):
    X.append([float(df['venue.location.lat'].iloc[i]), float(df['venue.location.lng'].iloc[i])])
    y.append(df['venue.categories'].iloc[i])
```

4. Results

- By machine learning user could get list of categories that he/she should avoid.



```
# train and get list of categories to avoid
from sklearn.neighbors import KNeighborsClassifier
ans = set()

for i in range(1, len(cat)):
    neigh = KNeighborsClassifier(n_neighbors=i)
    neigh.fit(X, y)
    ans.add(neigh.predict([list((float(lat), float(lng)))])[0])
    del(neigh)
```

```
ans
```

```
{'Café', 'Hotel'}
```

5. Discussion

- So user who was planning to open the shop at (35.704713616784005, 128.45710541327045), should avoid Café and Hotel.

6. Conclusion

- I only used KNN and didn't changed any hyperparameter except number of neighbors. If there were datasets that shows such as category of the shop, term of the shop survived, I think it would be able to make a model to predict the income of shop at specific place and shop of specific category.