The Battle of the Neighborhoods

Introduction: Business Opportunity

- An idea to bring foreign culture, happiness to neighborhoods and increase earnings.
- The target audiences are decision maker from bank, needs the bank to provide financial support.
- The idea is to leverage Foursquare data to explore nearby venues and find an optimal location to run an Italian restaurant in Germany.

Data acquisition and cleaning

- Based on the introduction, the following factors influence the decision making of candidate location.
  - the number of existing venues in the neighborhoods (any type of similar business).
  - the number of distance to the similar business in the neighborhoods.
  - the distance of neighborhood from city center.
- Following are data sources:
  - Address data come from Bing Map
  - Coordinates data come from Bing Map
  - Venues data come from Foursquare

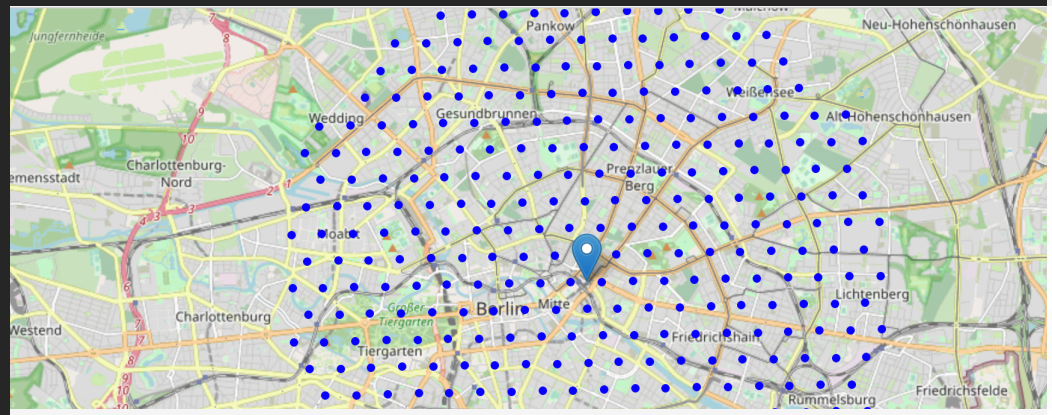
```
def get_address(lat, lon):
    g = geocoder.bing([lat, lon], method='reverse', key=BING_API_KEY)
    address = g.json['raw']['address']['formattedAddress']
    return address

addr = get_address(berlin_center[0], berlin_center[1])
print('Reverse geocoding check')
print('-'*23)
print('Address of [{}, {}] is: {}'.format(berlin_center[0], berlin_center[1], addr))
```

```
Reverse geocoding check
-----------------------
Address of [52.521671295166016, 13.413330078125] is: Alexanderplatz 1-5, 10178 Berlin
```

```
def get_venues_near_location(lat, lon, category, radius=500, limit=100):
    version = '20210101'
    url = 'https://api.foursquare.com/v2/venues/explore?ll={},{}&categoryId={}&radius={}&limit={}&v={}&client_id={}&client_secret={}'.format(\
    lat, lon, category, radius, limit, version, client_id, client_secret)
    #print(url)
    #results = requests.get(url).json()
    #print(requests.get(url).json())
    try:
        results = requests.get(url).json()['response']['groups'][0]['items']
        #print('\n',results[0],'\n')
        venues = [(item['venue']['id'],
                   item['venue']['name'],
                   get_categories(item['venue']['categories']),
                   (item['venue']['location']['lat'], item['venue']['location']['lng']),
                   format_address(item['venue']['location']),
                   item['venue']['location']['distance']) for item in results]
        #print(venues)
    except:
        venues = []
    return venues
```

Exploratory data analysis




```
er of restaurents: ', len(restaurants))
er of Italian restaurants: ', len(italian_restaura
 Italian restaurents: {:.2f}%'.format(len(italian_
nber of restaurants in neighborhood: ', np.array(
```

```
 restaurents:  2049
 Italian restaurants:  298
ian restaurents: 14.54%
of restaurants in neighborhood:  4.876373626373627
```

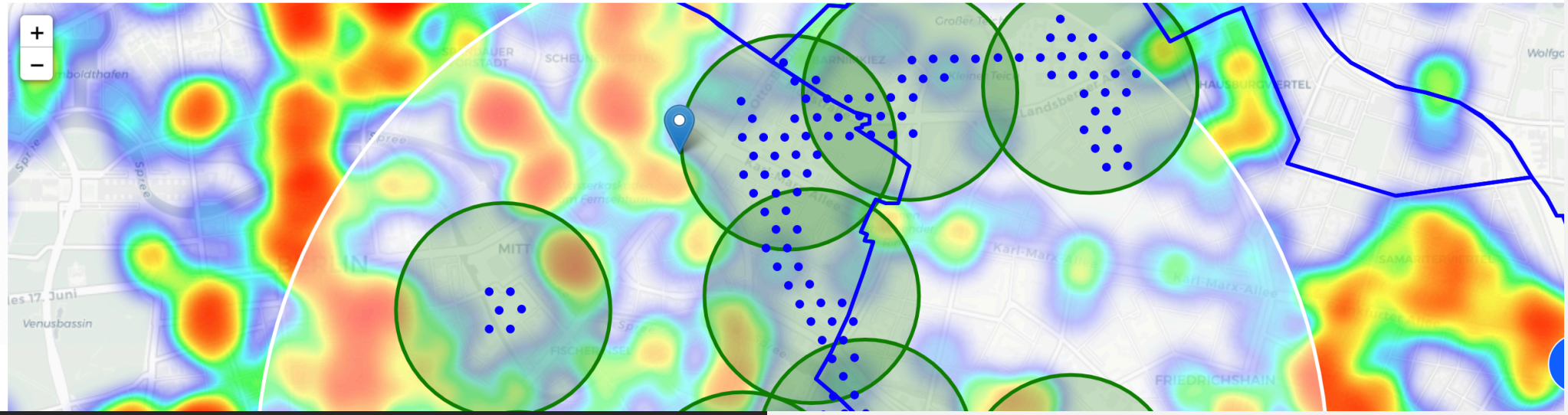
```
    try:
        results = requests.get(url).json()['response']['groups'][0]['items']
        #print('\n',results[0],'\n')
        venues = [(item['venue']['id'],
                   item['venue']['name'],
                   get_categories(item['venue']['categories']),
                   (item['venue']['location']['lat'], item['venue']['location']['lng']),
                   format_address(item['venue']['location']),
                   item['venue']['location']['distance']) for item in results]
        #print(venues)
    except:
        venues = []
    return venues

print('Venue coding check')
print('-'*18)
print(get_venues_near_location(berlin_center[0], berlin_center[1], food_category)[:1])
```

Address	Latitude	Longitude	X	Y	Distar
00, 12101 Berlin	52.469947	13.388690	390548.503582	5.814530e+06	
Berlin, Germany	52.470067	13.397520	391148.503582	5.814530e+06	
Berlin, Germany	52.470186	13.406349	391748.503582	5.814530e+06	
74, 12101 Berlin	52.470305	13.415178	392348.503582	5.814530e+06	
23, 12051 Berlin	52.470423	13.424008	392948.503582	5.814530e+06	

```
folium.Marker(berlin_center).add_to(map_berlin)
for lon, lat in cluster_centers:
    folium.Circle([lat, lon], radius=500, color='green', fill=True, fill_opacity=0.25).add_to(map_berlin)
for lat, lon in good_locations:
    folium.CircleMarker([lat, lon], radius=2, color='blue', fill=True, fill_color='blue', fill_opacity=1).add_to(map_berlin)
folium.GeoJson(berlin_boroughs, style_function=boroughs_style, name='geojson').add_to(map_berlin)
map_berlin
```

]:



Modeling

Conclusion

enters of areas recommended for further analysis
==========

ße 72, 10969 Berlin	=> 1.9km from Alexanderplatz
Vereinten Nationen 30, 10249 Berlin	=> 1.1km from Alexanderplatz
er Weg, 10179 Berlin	=> 1.6km from Alexanderplatz
traße 117, 12435 Berlin	=> 4.0km from Alexanderplatz
ner Straße 33, 10969 Berlin	=> 2.6km from Alexanderplatz
rktstraße, 10179 Berlin	=> 0.9km from Alexanderplatz
-Ernst Straße 28, 10243 Berlin	=> 2.4km from Alexanderplatz
Jakobstraße 114, 10969 Berlin	=> 1.9km from Alexanderplatz
enheide 81, 10965 Berlin	=> 3.9km from Alexanderplatz
Flutgraben 3, 10997 Berlin	=> 3.7km from Alexanderplatz
ichaelkirchplatz 23, 10179 Berlin	=> 1.6km from Alexanderplatz
Landsberger Allee 48-50, 10249 Berlin	=> 2.0km from Alexanderplatz
Schloßplatz 1, 10178 Berlin	=> 1.1km from Alexanderplatz
Berolinastraße 12, 10178 Berlin	=> 0.5km from Alexanderplatz
Neuenburger Straße 22D, 10969 Berlin	=> 2.5km from Alexanderplatz