Section **2: Data**

Data Description

In this section, I will describe the data used to solve the problem as described previously.

As noted below in the Further Development Section, it is possible to attempt quite complex and sophisticated scenarios when approaching this problem. However, given the size of the project and for simplicity only the following scenario will be addressed:

1. Query the FourSqaure website for the top sites in Casablanca
2. Use the FourSquare API to get supplemental geographical data about the top sites
3. Use the FourSquare API to get top restaurent recommendations closest to each of the top site
4. Use open source Casablanca Crime data to provide the user with additional crime data

Top Sites from FourSquare Website

Although FourSquare provides a comprehensive API, one of the things that API does not easily support is a mechanism to directly extract the top N sites / venues in a given city. This data, however, is easily available directly from the FourSquare Website. To do this simply go to www.foursquare.com, enter the city of your choise and select Top Picks from *I'm Looking For* selection field.

Using BeautifulSoup and Requests the results of the Top Pick for Casablanca was retrieved. A sample venue is shown below:

```
<div class="venueDetails">
    <div class="venueName">
        <h2>
    <a href="/v/Sidibad-park/42b75880f964a52090251fe3" target="_blank">Sindibad
Park
    </a>
</h2>
    </div>
    <div class="venueMeta">
        <div class="venueScore positive" style="background-color: #00B551;"
title="7.7/10 - People like this place">7.7</div>
        <div class="venueAddressData">
            <div class="venueAddress">Km 2 Avenue la corniche St, Ain Diab
Casablanca</div>
```

```
            <div class="venueData"><span class="venueDataItem"><span
class="categoryName">Park</span><span class="delim"> • </span></span>
            </div>
        </div>
    </div>
</div>
```

From this HTML the following data can be extracted:

- Venue Name
- Venue Score
- Venue Category
- Venue HREF
- Venue ID [Extracted from the HREF]

A sample of the extracted data is given below:

id	score	category	name	h
42b75880f964a52090251fe3	7.7	Park	Sindibad Park	/v/sindibad-park/42b75880f964a
4b9511c7f964a520f38d34e3	8.6	Trail	UAE Place	/v/UAE-Place/4b951
49e9ef74f964a52011661fe3	7.6	Art Museum	The Art Institute of Morocco	/v/the-art-institute-Morocco/49e9ef74f

We will have a closer look at this data gather later on when the supplemental geographical data has been added.

Supplemental Geographical Data

Using the `id` field extracted from the HTML it is then possible to get further supplemental geographical details about each of the top sites from FourSquare using the following sample API call:

```
# Get the properly formatted address and the latitude and longitude
url =
'https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.forma
t(
    venue_id,
    cfg['client_id'],
    cfg['client_secret'],
    cfg['version'])

result = requests.get(url).json()
result['response']['venue']['location']
```

The requests returns a JSON object which can then be queried for the details required. The last line in the sample code above returns the following sample JSON:


```
{
   "city":"Casablanca",
   "lng":-87.62323915831546,
   "crossStreet":"Avenue la courniche ",
   "neighborhood":"Morocco Mall",
   "postalCode":"20200",
   "cc":"MAR",
   "formattedAddress":[
      "Km 2 Avenue la corniche St, Ain Diab ",
      "Casablanca, IL 20200",
      "Morocco"
   ],
   "state":"IL",
   "address":" Km 2 Avenue la corniche St, Ain Diab Casablanca ",
   "country":"Morocco"
}
```


From this the following attributes are extracted:

- Venue Address
- Venue Postalcode
- Venue City
- Venue Latitude
- Venue Longitude

Data Analysis and Visualisation

An initial look at the data shows that there are 30 rows of data [as expected] each with 10 attributes. The variable types are all correct except the Venue Rating or Score which will be converted to a float. After converting the score column to a float it can clearly be seen that we have the top venues with a mean of `9.532`.

```
df_top_venues.shape
(30, 10)

df_top_venues.dtypes
id             object
score          object
category       object
name           object
address        object
postalcode     object
city           object
href           object
latitude      float64
longitude     float64
dtype: object
```

```
df_top_venues.score.describe()
count    30.000000
mean      9.523333
std       0.072793
min       9.400000
25%       9.500000
50%       9.500000
75%       9.600000
max       9.700000
Name: score, dtype: float64
```

We are now ready to get the top restaurents within 500 meters of each of the top sites.

FourSquare Restaurent Recommendation Data

Using the the list of all `id` values in the Top Sites DataFrame and the FourSquare `categoryID` that represents all food venues we now search for restaurants within a 500 meter radius.

```
# Configure additional Search parameters
categoryId = '4d4b7105d754a06374d81259'
radius = 500
limit = 15

url =
'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{
}&v={}&categoryId={}&radius={}&limit={}'.format(
    cfg['client_id'],
    cfg['client_secret'],
    ven_lat,
    ven_long,
    cfg['version'],
    categoryId,
    radius,
    limit)

results = requests.get(url).json()
```

The requests returns a JSON object which can then be queried for the restaurant details required.

From this JSON the following attributes are extraced and added to the Dataframe:

- Restaurant ID
- Restaurant Category Name
- Restaurant Category ID
- Restaurant Nest_name
- Restaurant Address
- Restaurant Postalcode
- Restaurant City

- Restaurant Latitude
- Restaurant Longitude
- Venue Name
- Venue Latitude
- Venue Longitude

The only piece of data that is missing is the Score or Rating of the Restaurant. To get this we need to make another FourSquare API query using the id of the Restaurant:

```
# Get the restaurant score and href
rest_url =
'https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.forma
t(
    rest_id,
    cfg['client_id'],
    cfg['client_secret'],
    cfg['version'])

result = requests.get(rest_url).json()
rest_score = result['response']['venue']['rating']
```

Using just the data in this DataFrame we will be able to generate maps displaying the chosen Top List Venue and the best scored surrounding restaurants.

Looking at the data we get an interesting insight into the range of restuarants that are included. From a list of 30 top venues only 28 actually had more than 10 to provide the user with a real choice. In total there were 387 restaurants found of which 240 were unique occuring only once in the data. There were 72 categories of restaurants. The mean score of all the restaurants wa `8.23` with a manimum value of `9.5` and a minimum value of `5.3`.

Coffee Shops (52) and Pizza Places (29) were the top two most frequently occurring categories but Pie Shops (9.4000) and French Restaurants (9.4000) were the restaurant categories with the highest average score.

```
# What is the shape of the Restaurants DataFrame
df_restaurant.shape
(387, 13)

# Get a count of the top venues that had more than 10 restaurant within 500 meters
#  The number of unique restaurants
#  The number of unique restaurant categories
df_restaurant.venue_name.nunique()
28
df_restaurant.name.nunique()
240
df_restaurant.category.nunique()
72

# Look at the data types
df_restaurant.dtypes
id                  object
```

```
score              float64
category            object
categoryID          object
name                object
address             object
postalcode          object
city                object
latitude           float64
longitude          float64
venue_name          object
venue_latitude     float64
venue_longitude    float64
dtype: object
```

```
# Describe the Score attribute
df_restaurant.score.describe()
```

```
count    387.000000
mean       8.286563
std        0.930138
min        5.300000
25%        7.800000
50%        8.500000
75%        9.000000
max        9.500000
Name: score, dtype: float64
```

```
df_restaurant.groupby('category')['name'].count().sort_values(ascending=False)[:10
]
```

```
category
Coffee Shops                        52
Pizza Places                        29
Cafés                               24
Bakeries                            15
Burger Joints                       15
Gastropubs                          15
New American Restaurants            15
Mexican Restaurants                 14
Breakfast Spots                     13
Fast Food Restaurants               13
```

```
df_restaurant.groupby('category')['score'].mean().sort_values(ascending=False)[:10
]
```

```
category
Pie Shops                           9.4000
French Restaurants                  9.4000
Molecular Gastronomy Restaurants    9.3000
Filipino Restaurants                9.2000
Cuban Restaurants                   9.1000
Ice Cream Shops                     9.0625
Mediterranean Restaurants           9.0600
Korean Restaurants                  9.0000
Latin American Restaurants          9.0000
Fish & Chips Shops                  9.0000
```