Capstone Project - Residential Area For Education

Applied Data Science Capstone by IBM/Coursera

Table of contents

Introduction: Business Problem

In this project we will try to find an optimal location for a better education options available in Miami. Specifically, this report will be targeted to stakeholders interested in selecting a **residence location** in **Miami**, Florida, USA which has maximum **Education institution**.

There are lots educational institutes available in Miami and we will try to detect **locations that has more options for Education**.

We will use our data science powers to generate a few most promising neighborhoods based on this criteria. Advantages of each area will then be clearly expressed so that best possible final location can be chosen by stakeholders.

Data

Based on definition of our problem, factors that will influence our decision are:

- Number of existing education institution available in the neighborhood

Following data sources will be needed to extract/generate the required information:

- Neighborhoods in Miami and its Co-ordinates. This data will be extracted from the following web page. https://en.wikipedia.org/wiki/List_of_neighborhoods_in_Miami (https://en.wikipedia.org/wiki/List_of_neighborhoods_in_Miami)

- Number of educational institutes and their type and location in every neighborhood will be obtained using **Foursquare API**

Before we get the data and start exploring it, let's download all the dependencies that we will need.

In [1]:
```
import numpy as np # library to handle data in a vectorized manner

import pandas as pd # library for data analsysis
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

!pip install geopy

import json # library to handle JSON files

#!conda install -c conda-forge geopy --yes # uncomment this line if you haven't c
from geopy.geocoders import Nominatim # convert an address into latitude and long

import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas data

# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors


#!conda install -c conda-forge folium=0.5.0 --yes # uncomment this line if you ha
import folium # map rendering library

print('Libraries imported.')
```

```
Requirement already satisfied: geopy in c:\users\jomyjohn\anaconda3\lib\site-pa
ckages (2.0.0)
Requirement already satisfied: geographiclib<2,>=1.49 in c:\users\jomyjohn\anac
onda3\lib\site-packages (from geopy) (1.50)
Libraries imported.
```

1. Download and Explore Dataset

Miami has a total of 24 neighborhoods. In order to segment the neighborhoods and explore them, we will essentially need a dataset that contains the neighborhoods well as the latitude and longitude coordinates of each neighborhood.

Load and explore the data

Next, let's load the data.

In [2]:
```
from bs4 import BeautifulSoup
import requests

url = 'https://en.wikipedia.org/wiki/List_of_neighborhoods_in_Miami'

r = requests.get(url)

soup = BeautifulSoup(r.content)
```

In [3]:
```

table = soup.find('table')
df = pd.read_html(str(table))[0]
```

In [4]:
```
df.head()
```

Out[4]:

	Neighborhood	Demonym	Population2010	Population/Km²	Sub-neighborhoods	Coordinates
0	Allapattah	NaN	54289	4401	NaN	25.815-80.224
1	Arts & Entertainment District	NaN	11033	7948	NaN	25.799-80.190
2	Brickell	Brickellite	31759	14541	West Brickell	25.758-80.193
3	Buena Vista	NaN	9058	3540	Buena Vista East Historic District and Design ...	25.813-80.192
4	Coconut Grove	Grovite	20076	3091	Center Grove, Northeast Coconut Grove, Southwe...	25.712-80.257

Remove columns which is not required

In [5]:
```
df.drop(["Demonym","Population2010","Population/Km²","Sub-neighborhoods"], axis =
```

Remove rows which has no coordinate value

In [6]:
```
df.drop(df[df['Coordinates'].isnull()].index, inplace=True)
```

Split the coordinate in to Latitude and Longitude columns

In [7]:
```
df[['Latitude','Longitude']] = df['Coordinates'].str.split('-',expand=True)
```

remove the Coordinates columns

In [8]:
```
df.drop(["Coordinates"], axis = 1, inplace=True)
```

While splitiing columns we lost -, just add that to longitude.

In [9]:
```
df['Longitude'] = '-' + df['Longitude']
```

In [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 24 entries, 0 to 24
Data columns (total 3 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Neighborhood  24 non-null     object
 1   Latitude      24 non-null     object
 2   Longitude     24 non-null     object
dtypes: object(3)
memory usage: 768.0+ bytes
```

Convert Latitude and Longitude columns to flot

In [11]:
```
df[['Latitude', 'Longitude']] = df[['Latitude', 'Longitude']].apply(pd.to_numeric
```

In [12]:
```
neighborhoods = df
```

Let's take a quick look at the data.

In [13]:
```
neighborhoods.head()
```

Out[13]:

	Neighborhood	Latitude	Longitude
0	Allapattah	25.815	-80.224
1	Arts & Entertainment District	25.799	-80.190
2	Brickell	25.758	-80.193
3	Buena Vista	25.813	-80.192
4	Coconut Grove	25.712	-80.257

Take a look at the empty dataframe to confirm that the columns are as intended.

In [14]:
```
print('The dataframe has {} neighborhoods.'.format(neighborhoods.shape[0]))
```

```
The dataframe has 24 neighborhoods.
```

Use geopy library to get the latitude and longitude values of Miami City.

In order to define an instance of the geocoder, we need to define a user_agent. We will name our agent *ny_explorer*, as shown below.

In [15]:
```
address = 'Miami, FL'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Miami City are {}, {}.'.format(latitude, lor
```

```
The geograpical coordinate of Miami City are 25.7742658, -80.1936589.
```

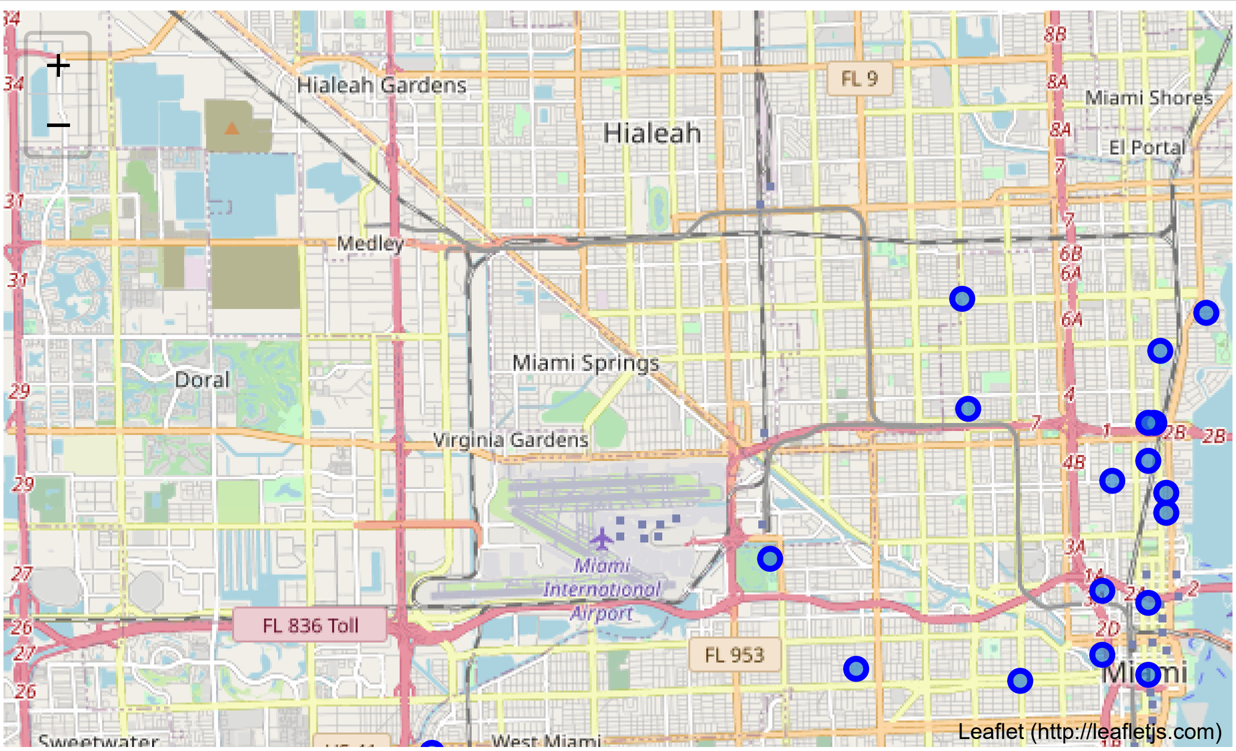
Create a map of Miami with neighborhoods superimposed on top.

In [16]:
```
# create map of Miami using latitude and longitude values
map_miami = folium.Map(location=[latitude, longitude], zoom_start=12)

# add markers to map
for lat, lng, neighborhood in zip(neighborhoods['Latitude'], neighborhoods['Longi
    label = '{}'.format(neighborhood)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_miami)

map_miami
```

Out[16]:



In [17]:
```
#neighborhoods.info()
neighborhoods.head(40)
```

Out[17]:

	Neighborhood	Latitude	Longitude
0	Allapattah	25.815	-80.224
1	Arts & Entertainment District	25.799	-80.190
2	Brickell	25.758	-80.193
3	Buena Vista	25.813	-80.192
4	Coconut Grove	25.712	-80.257
5	Coral Way	25.750	-80.283
6	Design District	25.813	-80.193
7	Downtown	25.774	-80.193
8	Edgewater	25.802	-80.190
9	Flagami	25.762	-80.316
10	Grapeland Heights	25.792	-80.258
12	Liberty City	25.832	-80.225
13	Little Haiti	25.824	-80.191
14	Little Havana	25.773	-80.215
15	Lummus Park	25.777	-80.201
16	Midtown	25.807	-80.193
17	Overtown	25.787	-80.201
18	Park West	25.785	-80.193
19	The Roads	25.756	-80.207
20	Upper Eastside	25.830	-80.183
21	Venetian Islands	25.791	-80.161
22	Virginia Key	25.736	-80.155
23	West Flagler	25.775	-80.243
24	Wynwood	25.804	-80.199

Next, we are going to start utilizing the Foursquare API to explore the neighborhoods and segment them.

Define Foursquare Credentials and Version

In [45]:
```
CLIENT_ID = '' # your Foursquare ID
CLIENT_SECRET = '' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version
LIMIT = 100 # A default Foursquare API limit value

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID:
CLIENT_SECRET:
```

Let's explore the first neighborhood in our dataframe.

Get the first neighborhood's name.

In [19]:
```
neighborhoods.loc[0, 'Neighborhood']
```

Out[19]:
```
'Allapattah'
```

Get the neighborhood's latitude and longitude values.

In [20]:
```
neighborhood_latitude = round(neighborhoods.loc[0, 'Latitude'],3) # neighborhood
neighborhood_longitude = round(neighborhoods.loc[0, 'Longitude'],3) # neighborhoo

neighborhood_name = neighborhoods.loc[0, 'Neighborhood'] # neighborhood name

print('Latitude and longitude values of {} are {}, {}.'.format(neighborhood_name,
                                                               neighborhood_latit
                                                               neighborhood_longi
```

```
Latitude and longitude values of Allapattah are 25.815, -80.224.
```

Now, let's get the top 100 venues that are in Allapattah within a radius of 1500 meters.

First, let's create the GET request URL. Name your URL **url**.

In [21]:
```
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 1500 # define radius
categoryId='4bf58dd8d48988d13b941735' # category for school
# create URL
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    categoryId,
    radius,
    LIMIT)


url # display URL
```

Out[21]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=QBDVQDIAMM0I0Q3QVMOL31
R3XQRX0BPV5OOSEIILFNFXDGEJ&client_secret=JPGFSDCCFB1POQDRPVMJOCGOBWJQOLEGD24GUT
MDAI3SPM1N&v=20180605&ll=25.815,-80.224&categoryId=4bf58dd8d48988d13b941735&rad
ius=1500&limit=100'
```

Send the GET request and examine the resutls

In [22]:
```
results = requests.get(url).json()
results
```

Out[22]:

```
{'meta': {'code': 200, 'requestId': '5f93e6eea106903c946369f2'},
 'response': {'headerLocation': 'Model City',
  'headerFullLocation': 'Model City, Miami',
  'headerLocationGranularity': 'neighborhood',
  'query': 'school',
  'totalResults': 7,
  'suggestedBounds': {'ne': {'lat': 25.828500013500015,
    'lng': -80.2090313978883},
   'sw': {'lat': 25.801499986499987, 'lng': -80.23896860211171}},
  'groups': [{'type': 'Recommended Places',
    'name': 'recommended',
    'items': [{'reasons': {'count': 0,
       'items': [{'summary': 'This spot is popular',
         'type': 'general',
         'reasonName': 'globalInteractionReason'}]},
      'venue': {'id': '4c3a53500a71c9b6d01844c9',
       'name': 'Miami Jackson Senior High School',
       'location': {'address': '1751 NW 36th St',
        'lat': 25.809993097862588,
```


From the Foursquare lab in the previous module, we know that all the information is in the *items* key. Before we proceed, let's borrow the **get_category_type** function from the Foursquare lab.

In [23]:
```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

Now we are ready to clean the json and structure it into a *pandas* dataframe.

In [24]:
```
venues = results['response']['groups'][0]['items']

nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venu
nearby_venues =nearby_venues.loc[:, filtered_columns]

# filter the category for each row
nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1

# clean columns
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

nearby_venues.head()
```

```
<ipython-input-24-561c05f0fdd1>:3: FutureWarning: pandas.io.json.json_normalize
is deprecated, use pandas.json_normalize instead
  nearby_venues = json_normalize(venues) # flatten JSON
```

Out[24]:

	name	categories	lat	lng
0	Miami Jackson Senior High School	High School	25.809993	-80.225484
1	Allapattah Middle School	College Academic Building	25.817823	-80.218966
2	Lenora Braynon Smith Elementary School	Elementary School	25.818559	-80.217238
3	Brownsville Middle School	Middle School	25.819448	-80.235542
4	Earlington Heights Elementary	Elementary School	25.818344	-80.232717

And how many venues were returned by Foursquare?

In [25]:
```
print('{} venues were returned by Foursquare.'.format(nearby_venues.shape[0]))
```

```
7 venues were returned by Foursquare.
```

Methodology

In this project we will direct our efforts on detecting areas of Miami that have high educational institute density,

In first step we have collected the required **data: location and type (category) of every educational institutes in Miami ** (Allapattah). We have also **identified educational institute types** (according to Foursquare categorization).

Second step in our analysis will be calculation and exploration of '**educational institute density**' across different areas of Miami - we will use **heatmaps** to identify a few promising areas with high number of educational institutes in general.

We will present map of all neighborhoods with its total educational institute count of those locations to identify / neighborhoods which should be a optimal potential location for stakeholders.

Analysis

Let's perform some basic explanatory data analysis and derive some additional info from our raw data. Lets Explore each Neighborhoods in Miami

2. Explore Neighborhoods in Miami

Let's create a function to repeat the same process to all the neighborhoods in Miami

In [26]:
```
def getNearbyVenues(names, latitudes, longitudes, radius=1500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT,
            categoryId
        )

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

Now write the code to run the above function on each neighborhood and create a new dataframe called miami_venues.

In [27]:
```
neighborhoods
```

Out[27]:

	Neighborhood	Latitude	Longitude
0	Allapattah	25.815	-80.224
1	Arts & Entertainment District	25.799	-80.190
2	Brickell	25.758	-80.193
3	Buena Vista	25.813	-80.192
4	Coconut Grove	25.712	-80.257
5	Coral Way	25.750	-80.283
6	Design District	25.813	-80.193
7	Downtown	25.774	-80.193
8	Edgewater	25.802	-80.190
9	Flagami	25.762	-80.316
10	Grapeland Heights	25.792	-80.258
12	Liberty City	25.832	-80.225
13	Little Haiti	25.824	-80.191
14	Little Havana	25.773	-80.215
15	Lummus Park	25.777	-80.201
16	Midtown	25.807	-80.193
17	Overtown	25.787	-80.201
18	Park West	25.785	-80.193
19	The Roads	25.756	-80.207
20	Upper Eastside	25.830	-80.183
21	Venetian Islands	25.791	-80.161
22	Virginia Key	25.736	-80.155
23	West Flagler	25.775	-80.243
24	Wynwood	25.804	-80.199

In [28]:
```
miami_venues = getNearbyVenues(names=neighborhoods['Neighborhood'],
                                   latitudes=neighborhoods['Latitude'],
                                   longitudes=neighborhoods['Longitude'])
```

```
Allapattah
Arts & Entertainment District
Brickell
Buena Vista
Coconut Grove
Coral Way
Design District
Downtown
Edgewater
Flagami
Grapeland Heights
Liberty City
Little Haiti
Little Havana
Lummus Park
Midtown
Overtown
Park West
The Roads
```

Let's check the size of the resulting dataframe

In [29]:
```
print(miami_venues.shape)
miami_venues.head()
```

```
(342, 7)
```

Out[29]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Allapattah	25.815	-80.224	Miami Jackson Senior High School	25.809993	-80.225484	High School
1	Allapattah	25.815	-80.224	Allapattah Middle School	25.817823	-80.218966	College Academic Building
2	Allapattah	25.815	-80.224	Lenora Braynon Smith Elementary School	25.818559	-80.217238	Elementary School
3	Allapattah	25.815	-80.224	Brownsville Middle School	25.819448	-80.235542	Middle School
4	Allapattah	25.815	-80.224	Earlington Heights Elementary	25.818344	-80.232717	Elementary School

Let's check how many venues were returned for each neighborhood

In [30]:
```
dfGroup = miami_venues.groupby('Neighborhood').count()
```

In [31]:
```
dfGroup.drop(["Neighborhood Latitude","Neighborhood Longitude","Venue Latitude",'
dfGroup.rename(columns={'Venue': 'NumberOfInstitutes'}, inplace=True)
dfGroup
```

Out[31]:

Neighborhood	NumberOfInstitutes
Allapattah	7
Arts & Entertainment District	20
Brickell	25
Buena Vista	17
Coconut Grove	5
Coral Way	6
Design District	18
Downtown	37
Edgewater	24
Flagami	9
Grapeland Heights	2
Liberty City	6
Little Haiti	10
Little Havana	13
Lummus Park	28
Midtown	21
Overtown	17
Park West	19
The Roads	13
Upper Eastside	4
Venetian Islands	1
Virginia Key	2
West Flagler	12
Wynwood	26

Let's find out how many unique categories can be curated from all the returned venues

In [32]:
```
print('There are {} uniques categories.'.format(len(miami_venues['Venue Category'
```

```
There are 28 uniques categories.
```

3. Analyze Each Neighborhood

In [33]:
```
# one hot encoding
miami_onehot = pd.get_dummies(miami_venues[['Venue Category']], prefix="", prefix

# add neighborhood column back to dataframe
miami_onehot['Neighborhood'] = miami_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [miami_onehot.columns[-1]] + list(miami_onehot.columns[:-1])
miami_onehot = miami_onehot[fixed_columns]

miami_onehot.head()
```

Out[33]:

	Neighborhood	Adult Education Center	Art Gallery	Business Service	Church	College Academic Building	College Arts Building	College Lab	Daycare	Dr Sc
0	Allapattah	0	0	0	0	0	0	0	0	
1	Allapattah	0	0	0	0	1	0	0	0	
2	Allapattah	0	0	0	0	0	0	0	0	
3	Allapattah	0	0	0	0	0	0	0	0	
4	Allapattah	0	0	0	0	0	0	0	0	

And let's examine the new dataframe size.

In [34]:
```
miami_onehot.shape
```

Out[34]: (342, 29)

Next, let's group rows by neighborhood and by taking the mean of the frequency of occurrence of each category

In [35]:
```
miami_grouped = miami_onehot.groupby('Neighborhood').mean().reset_index()
miami_grouped
```

Out[35]:

	Neighborhood	Adult Education Center	Art Gallery	Business Service	Church	College Academic Building	College Arts Building	College Lab	Daycar
0	Allapattah	0.000000	0.000000	0.0	0.000000	0.142857	0.000000	0.0	0.00000
1	Arts & Entertainment District	0.050000	0.050000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
2	Brickell	0.040000	0.000000	0.0	0.040000	0.000000	0.000000	0.0	0.00000
3	Buena Vista	0.000000	0.058824	0.0	0.000000	0.000000	0.058824	0.0	0.00000
4	Coconut Grove	0.000000	0.000000	0.2	0.200000	0.400000	0.000000	0.0	0.00000
5	Coral Way	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
6	Design District	0.000000	0.055556	0.0	0.000000	0.000000	0.055556	0.0	0.00000
7	Downtown	0.027027	0.000000	0.0	0.027027	0.000000	0.000000	0.0	0.02702
8	Edgewater	0.041667	0.041667	0.0	0.000000	0.000000	0.041667	0.0	0.00000
9	Flagami	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.11111
10	Grapeland Heights	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
11	Liberty City	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
12	Little Haiti	0.000000	0.000000	0.0	0.000000	0.000000	0.100000	0.0	0.00000
13	Little Havana	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
14	Lummus Park	0.000000	0.000000	0.0	0.035714	0.000000	0.000000	0.0	0.03571
15	Midtown	0.047619	0.047619	0.0	0.000000	0.000000	0.047619	0.0	0.00000
16	Overtown	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
17	Park West	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.05263
18	The Roads	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
19	Upper Eastside	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
20	Venetian Islands	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
21	Virginia Key	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.5	0.00000
22	West Flagler	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.0	0.00000
23	Wynwood	0.038462	0.038462	0.0	0.000000	0.000000	0.038462	0.0	0.00000

Let's confirm the new size

In [36]:
```
miami_grouped.shape
```

Out[36]:
```
(24, 29)
```

Let's print each neighborhood along with the top 5 most common venues

In [37]:
```
num_top_venues = 5

for hood in miami_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = miami_grouped[miami_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(r
    print('\n')
```

```
2                 Church   0.2
3      Elementary School   0.2
4    Adult Education Center   0.0


----Coral Way----
                    venue  freq
0                  School  0.50
1       Elementary School  0.33
2          Nursery School  0.17
3  Adult Education Center  0.00
4      Miscellaneous Shop  0.00


----Design District----
                venue  freq
0              School  0.33
1   Elementary School  0.22
2         High School  0.17
3  Miscellaneous Shop  0.06
```

Let's put that into a *pandas* dataframe

First, let's write a function to sort the venues in descending order.

In [38]:
```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

Now let's create the new dataframe and display the top 10 venues for each neighborhood.

In [39]:
```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = miami_grouped['Neighborhood']

for ind in np.arange(miami_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(miami_g
```

In [40]:
```
dfGroup.reset_index()
neighborhoods_venues_sorted = neighborhoods_venues_sorted.join(dfGroup, on='Neigh
neighborhoods_venues_sorted.sort_values(['NumberOfInstitutes'], ascending=[False]
neighborhoods_venues_sorted.head(28)
```

Out[40]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7t Cc
7	Downtown	School	Elementary School	Language School	Preschool	Music School	General College & University	
14	Lummus Park	School	Elementary School	High School	Music School	General College & University	Language School	
23	Wynwood	School	Elementary School	High School	Language School	Preschool	Art Gallery	(E
2	Brickell	School	Preschool	Language School	Private School	Nursery School	Church	Elen
8	Edgewater	School	Elementary School	High School	Language School	Miscellaneous Shop	Art Gallery	(E
15	Midtown	School	Elementary School	High School	Language School	Preschool	Art Gallery	(E
1	Arts & Entertainment District	School	Elementary School	Language School	High School	Miscellaneous Shop	Art Gallery	Un
17	Park West	School	High School	Elementary School	Language School	Daycare	General College & University	Un
6	Design District	School	Elementary School	High School	Language School	Art Gallery	College Arts Building	Pre
3	Buena Vista	School	Elementary School	High School	Language School	Art Gallery	College Arts Building	Pre
16	Overtown	School	Elementary School	High School	Language School	General College & University	Playground	
18	The Roads	School	Preschool	Elementary School	Middle School	Zoo	Golf Course	Art
13	Little Havana	Elementary School	School	Language School	Private School	Music School	Middle School	
22	West Flagler	School	Elementary School	High School	Religious School	Middle School	Zoo	
12	Little Haiti	School	Elementary School	High School	College Arts Building	Preschool	Middle School	

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7t Cc
9	Flagami	School	Music School	Trade School	Daycare	Middle School	Elementary School	
0	Allapattah	Elementary School	High School	College Academic Building	Middle School	Zoo	Art Gallery	Bu
11	Liberty City	Elementary School	School	High School	Zoo	Art Gallery	Business Service	
5	Coral Way	School	Elementary School	Nursery School	Zoo	High School	Art Gallery	Bu
4	Coconut Grove	College Academic Building	Business Service	Church	Elementary School	Zoo	University	Art
19	Upper Eastside	School	High School	Elementary School	Zoo	Art Gallery	Business Service	
10	Grapeland Heights	Driving School	Golf Course	Zoo	University	Art Gallery	Business Service	
21	Virginia Key	College Lab	High School	Zoo	University	Art Gallery	Business Service	
20	Venetian Islands	Zoo	University	Art Gallery	Business Service	Church	College Academic Building	(E

In [41]:
```
neighborhoods_venues_sorted.sort_values(['NumberOfInstitutes'], ascending=[False]
neighborhoods_venues_sorted.reset_index(inplace=True)
```

Top 10 Neighborhood based on Educational Density

In [42]:
```
neighborhoods_venues_sorted.drop(["index"], axis = 1, inplace=True)
neighborhoods_venues_sorted.head(10)
```

Out[42]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue
0	Downtown	School	Elementary School	Language School	Preschool	Music School	General College & University	Private School
1	Lummus Park	School	Elementary School	High School	Music School	General College & University	Language School	Private School
2	Wynwood	School	Elementary School	High School	Language School	Preschool	Art Gallery	College Arts Building
3	Brickell	School	Preschool	Language School	Private School	Nursery School	Church	Elementary School
4	Edgewater	School	Elementary School	High School	Language School	Miscellaneous Shop	Art Gallery	College Arts Building
5	Midtown	School	Elementary School	High School	Language School	Preschool	Art Gallery	College Arts Building
6	Arts & Entertainment District	School	Elementary School	Language School	High School	Miscellaneous Shop	Art Gallery	University
7	Park West	School	High School	Elementary School	Language School	Daycare	General College & University	University
8	Design District	School	Elementary School	High School	Language School	Art Gallery	College Arts Building	Preschool
9	Buena Vista	School	Elementary School	High School	Language School	Art Gallery	College Arts Building	Preschool

In [43]:
```
miami_merged = neighborhoods

# merge miami_merged with neighborhoods to add latitude/longitude for each neighb
miami_merged = miami_merged.join(dfGroup, on='Neighborhood')


miami_merged
```

Out[43]:

	Neighborhood	Latitude	Longitude	NumberOfInstitutes
0	Allapattah	25.815	-80.224	7
1	Arts & Entertainment District	25.799	-80.190	20
2	Brickell	25.758	-80.193	25
3	Buena Vista	25.813	-80.192	17
4	Coconut Grove	25.712	-80.257	5
5	Coral Way	25.750	-80.283	6
6	Design District	25.813	-80.193	18
7	Downtown	25.774	-80.193	37
8	Edgewater	25.802	-80.190	24
9	Flagami	25.762	-80.316	9
10	Grapeland Heights	25.792	-80.258	2
12	Liberty City	25.832	-80.225	6
13	Little Haiti	25.824	-80.191	10
14	Little Havana	25.773	-80.215	13
15	Lummus Park	25.777	-80.201	28
16	Midtown	25.807	-80.193	21
17	Overtown	25.787	-80.201	17
18	Park West	25.785	-80.193	19
19	The Roads	25.756	-80.207	13
20	Upper Eastside	25.830	-80.183	4
21	Venetian Islands	25.791	-80.161	1
22	Virginia Key	25.736	-80.155	2
23	West Flagler	25.775	-80.243	12
24	Wynwood	25.804	-80.199	26

Finally, let's visualize the resulting clusters

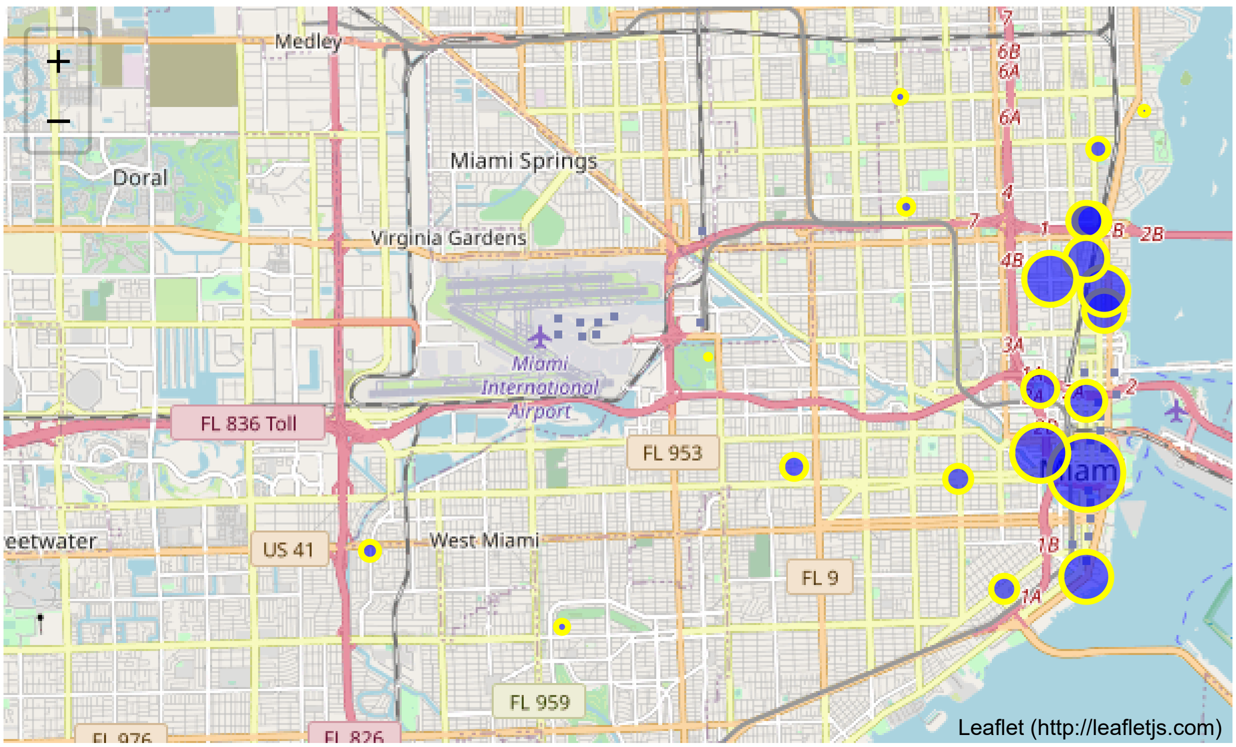
In [44]:
```python
from folium import plugins

# let's start again with a clean copy of the map of San Francisco
miami_map = folium.Map(location = [latitude, longitude], zoom_start = 12)

for lat, lng, label, size in zip(miami_merged.Latitude, miami_merged.Longitude, 
    folium.features.CircleMarker(
        [lat, lng],
        radius=size/2, # define how big you want the circle markers to be
        color='yellow',
        fill=True,
        popup=label,
        fill_color='blue',
        fill_opacity=0.6
    ).add_to(miami_map)

# display map
miami_map
```

Out[44]:



Results and Discussion

Our analysis shows that although there is a great number of educational institutes in Miami, when moving away from city center its density is reducing. Highest concentration of educational institutes was detected near to the coastal area especially souther area of the city. So we focused our attention to areas northern & costal area .

By considering data and exploring the map we can see that area near to Midtown neighborhood is most suitable for requirement.

We are also providing 1st Most Common institute type 2nd Most Common institute type, so other than educational institutes density user can also select residence based on institute type also.

Conclusion

```
Purpose of this project was to identify Miami areas with high number of
educational institutions in order to aid stakeholders in narrowing down the
search for optimal location for their residence. By calculating educational
institutions density distribution from Foursquare data we have identified.

By considering data and map we can see that area near to Midtown neighborhood
is most suitable for requirement.
```

In [ ]: