The Battle of the Neighborhoods - Week 2

Introduction & Business Problem :

A group of investors is exploring cities in the southern US State of Texas for their potential to successfully support a Latin American restaurant franchise.

Problem Background: The City of Houston, is one of the most populous city in the state of Texas. It is diverse and has a large population of Latin America origin inhabitants. Texas State is famous for their business friendly environment and It provides lot of opportunities. It has attracted many different players into the market. It is a global hub of business and commerce. The city is a major center for Oil & Gas, banking and finance, retailing, world trade, real estate, advertising, legal services, accountancy and engineering. This also translates into a highly competitive business environment.

Problem Description: We need to help a pool of potential inventors, about the best location to open a Latin American Food Restaurant in the City of Houston. The city has a large pool of potential patrons, and it's relatively large Latino population could make opening a Latin Restaurant a promising business opportunity; nevertheless due to its geographical extension and disparity of incomes among its population. Makes the restaurant location choice, a key factor for the restaurant commercial viability.

So it is evident that to survive in such market, it is very important to strategically plan the best possible location for the venue. Various factors need to be studied in order to decide on the Location such as :

1. Houston Population
2. Houston Demographics
3. Are there any venues like Gyms, Entertainment zones, Parks, malls, Offices nearby where floating population is high etc.
4. Who are the competitors in that location?
5. Cuisine served / Menu of the competitors
6. Segmentation of the neighbourhood

Target Audience: The target audience for this new restaurant was identified to be Latino Houston residents. To recommend the correct location, is key to identify the neighbourhoods with the highest number of Latino population.

Success Criteria: The success criteria of the project will be a good recommendation of neighbourhood choice to Group of investors, based on the relative scarcity of similar restaurants in that location, and the potential size of the Latin Population (target market) in the targeted neighbourhoods.

Data : One city will be analyzed in this project : Houston.

We will be using the below datasets for analyzing Houston

Data 1 : Neighborhood has a total of 88 neighborhoods. In order to segment the neighborhoods and explore them, we will essentially need a dataset that contains the 88 Houston's neighborhoods as well as the coordinates (latitude and longitude) for each of them. This dataset exists for free on the web. Link to the dataset is : https://en.wikipedia.org/wiki/List_of_Houston_neighborhoods (https://en.wikipedia.org/wiki/List_of_Houston_neighborhoods)

Data 2 : For the required analysis we will get data from following sources as given below :

1. Houston Neighbourhoods: : https://opendata.arcgis.com/datasets/f50cc53c15bf4fb3940ab6e8c2534f3d_2.csv (https://opendata.arcgis.com/datasets/f50cc53c15bf4fb3940ab6e8c2534f3d_2.csv)
2. Houston Demographics https://en.wikipedia.org/wiki/Demographics_of_Houston (https://en.wikipedia.org/wiki/Demographics_of_Houston)
3. Houston Hispanic Population by neighbourhood: https://opendata.arcgis.com/datasets/f50cc53c15bf4fb3940ab6e8c2534f3d_2.csv (https://opendata.arcgis.com/datasets/f50cc53c15bf4fb3940ab6e8c2534f3d_2.csv)
4. Houston Income (including Hispanics) data:http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN (http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN.pdf) (replaced the originally proposed source due to be more recent https://opendata.arcgis.com/datasets/35ef9379a9fd491aab08cb63aa33893e_1.csv (https://opendata.arcgis.com/datasets/35ef9379a9fd491aab08cb63aa33893e_1.csv))

Data 3 : Houston geographical coordinates data will be sourced from Geocoder package and will be utilized as input for the Foursquare API, that will be leveraged to provide venue information for each neighborhood. We will use the Foursquare API to explore neighborhoods in Houston. The below is image of the Foursquare API data. Additionally we will be using Geocoder Folium & Bautiful Soup to facilitate the attribution of coordinates on data sourced from the 2010 Census database (GIS). For data in Acrobat format (*.pdf) we will extract data using Tabula and wrangling the data using pandas.

In [63]:

```
# Install Geocoder Folium & Beautiful Soup
import sys
!{sys.executable} -m pip install geocoder
!{sys.executable} -m pip install folium

#!conda install -c conda-forge geopy --yes
print('Packages installed.')
```

```
Requirement already satisfied: geocoder in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.6/site-packages (1.38.1)
Requirement already satisfied: ratelim in /home/jupyterlab/conda/envs/pyth
on/lib/python3.6/site-packages (from geocoder) (0.1.6)
Requirement already satisfied: requests in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.6/site-packages (from geocoder) (2.22.0)
Requirement already satisfied: future in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (from geocoder) (0.18.2)
Requirement already satisfied: six in /home/jupyterlab/conda/envs/python/l
ib/python3.6/site-packages (from geocoder) (1.14.0)
Requirement already satisfied: click in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (from geocoder) (7.1.1)
Requirement already satisfied: decorator in /home/jupyterlab/conda/envs/py
thon/lib/python3.6/site-packages (from ratelim->geocoder) (4.4.1)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from reque
sts->geocoder) (1.25.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /home/jupyterlab/c
onda/envs/python/lib/python3.6/site-packages (from requests->geocoder) (3.
0.4)
Requirement already satisfied: idna<2.9,>=2.5 in /home/jupyterlab/conda/en
vs/python/lib/python3.6/site-packages (from requests->geocoder) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in /home/jupyterlab/cond
a/envs/python/lib/python3.6/site-packages (from requests->geocoder) (2019.
11.28)
Requirement already satisfied: folium in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (0.5.0)
Requirement already satisfied: requests in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.6/site-packages (from folium) (2.22.0)
Requirement already satisfied: six in /home/jupyterlab/conda/envs/python/l
ib/python3.6/site-packages (from folium) (1.14.0)
Requirement already satisfied: branca in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (from folium) (0.3.1)
Requirement already satisfied: jinja2 in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (from folium) (2.11.1)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from reque
sts->folium) (1.25.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /home/jupyterlab/c
onda/envs/python/lib/python3.6/site-packages (from requests->folium) (3.0.
4)
Requirement already satisfied: idna<2.9,>=2.5 in /home/jupyterlab/conda/en
vs/python/lib/python3.6/site-packages (from requests->folium) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in /home/jupyterlab/cond
a/envs/python/lib/python3.6/site-packages (from requests->folium) (2019.1
1.28)
Requirement already satisfied: MarkupSafe>=0.23 in /home/jupyterlab/conda/
envs/python/lib/python3.6/site-packages (from jinja2->folium) (1.1.1)
Packages installed.
```

In [64]:

```
pip install BeautifulSoup4
```

```
Requirement already satisfied: BeautifulSoup4 in /home/jupyterlab/conda/en
vs/python/lib/python3.6/site-packages (4.8.2)
Requirement already satisfied: soupsieve>=1.2 in /home/jupyterlab/conda/en
vs/python/lib/python3.6/site-packages (from BeautifulSoup4) (2.0)
Note: you may need to restart the kernel to use updated packages.
```

In [65]:

```
pip install geopy
```

```
Requirement already satisfied: geopy in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (1.21.0)
Requirement already satisfied: geographiclib<2,>=1.49 in /home/jupyterlab/
conda/envs/python/lib/python3.6/site-packages (from geopy) (1.50)
Note: you may need to restart the kernel to use updated packages.
```

In [66]:

```
import numpy as np
import pandas as pd
import requests
from sklearn.cluster import KMeans
import folium
from bs4 import BeautifulSoup # library to parse HTML and XML documents
import os
import matplotlib.cm as cm
import matplotlib.colors as colors
import json # library to handle JSON files
from geopy.geocoders import Nominatim # convert an address into latitude and longitude
 values
import geocoder # to get coordinates
#import Nominatim # convert an address into latitude and longitude values
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe

print('Packages installed.')
```

```
Packages installed.
```

In [67]:

```
# Getting USA 2010 census data for obtaining the population etnicity per Houston Neighb
orhood

Demog_df = pd.read_csv('https://opendata.arcgis.com/datasets/f50cc53c15bf4fb3940ab6e8c2
534f3d_2.csv')

Demog_df.head()
```

Out[67]:

	OBJECTID	SUM_POP100	SUM_HU100	SUM_TotPop	SUM_HispPop	SUM_NonHispPop	SU
0	1	3881	2104	3881	711	3170	
1	2	13471	5120	13471	10603	2868	
2	3	16716	3664	16716	3266	13450	
3	4	2497	1133	2497	603	1894	
4	5	49277	31563	49277	7311	41966	

5 rows × 32 columns

In [68]:

```
#Replacing the filed 'Name' for 'Neighborhood' in the dataset.
Demog_df.rename(columns={'Name':'Neighborhood'},inplace=True)

Demog_df.head(88)
```

Out[68]:

	OBJECTID	SUM_POP100	SUM_HU100	SUM_TotPop	SUM_HispPop	SUM_NonHispPop	S
0	1	3881	2104	3881	711	3170	
1	2	13471	5120	13471	10603	2868	
2	3	16716	3664	16716	3266	13450	
3	4	2497	1133	2497	603	1894	
4	5	49277	31563	49277	7311	41966	
...	...	...	...	...	...	...	
83	84	39031	19959	39031	8992	30039	
84	85	28957	19231	28957	5738	23219	
85	86	45294	20069	45294	5951	39343	
86	87	31352	12350	31352	18051	13301	
87	88	7323	4015	7323	2121	5202	

88 rows × 32 columns

In [69]:

```
# define a function to get coordinates
def get_latlng(Neighborhood):
    # initialize your variable to None
    lat_lng_coords = None
    # loop until you get the coordinates
    while(lat_lng_coords is None):
        g = geocoder.arcgis('{}, Houston, USA'.format(Neighborhood))
        lat_lng_coords = g.latlng
    return lat_lng_coords
```

In [70]:

```
# call the function to get the coordinates, store in a new list using list comprehensio
n
coords = [ get_latlng(neighborhood) for neighborhood in Demog_df["Neighborhood"].tolist
() ]
```

In [71]:

```
# create temporary dataframe to populate the coordinates into Latitude and Longitude
coords_df = pd.DataFrame(coords, columns=['Latitude', 'Longitude'])
```

In [72]:

```
#Include the coordinates in the dataframe
# merge the coordinates into the original dataframe
Demog_df['Latitude'] = coords_df['Latitude']
Demog_df['Longitude'] = coords_df['Longitude']

Demog_df.head(88)
```

Out[72]:

	OBJECTID	SUM_POP100	SUM_HU100	SUM_TotPop	SUM_HispPop	SUM_NonHispPop	S
0	1	3881	2104	3881	711	3170	
1	2	13471	5120	13471	10603	2868	
2	3	16716	3664	16716	3266	13450	
3	4	2497	1133	2497	603	1894	
4	5	49277	31563	49277	7311	41966	
...	...	...	...	...	...	...	
83	84	39031	19959	39031	8992	30039	
84	85	28957	19231	28957	5738	23219	
85	86	45294	20069	45294	5951	39343	
86	87	31352	12350	31352	18051	13301	
87	88	7323	4015	7323	2121	5202	

88 rows × 34 columns

In [73]:

```
# Dropping all unncessary fields
Demog_df.drop(['SUM_POP100', 'SUM_HU100', 'SUM_NHOneRace','SUM_NH_White','SUM_NH_Black'
,'SUM_NH_AmInd','SUM_NH_Asian','SUM_NH_HawPacI','SUM_NH_Other','SUM_NH_2orMore','SUM_VA
P_TotPop','SUM_VAP_HispPo','SUM_VAP_NonHis','SUM_VAP_NHOneR','SUM_VAP_NH_Whi','SUM_VAP_
NH_Bla','SUM_VAP_NH_AmI','SUM_VAP_NH_Asi','SUM_VAP_HawPac','SUM_VAP_NH_Oth','SUM_VAP_NH
_2or','Shapearea','Shapelen'], axis=1)
```

Out[73]:

	OBJECTID	SUM_TotPop	SUM_HispPop	SUM_NonHispPop	SUM_TotHousing	SUM_OccH
0	1	3881	711	3170	2104	197
1	2	13471	10603	2868	5120	440
2	3	16716	3266	13450	3664	292
3	4	2497	603	1894	1133	94
4	5	49277	7311	41966	31563	2743
...	...	...	...	...	...	
83	84	39031	8992	30039	19959	1750
84	85	28957	5738	23219	19231	1706
85	86	45294	5951	39343	20069	1826
86	87	31352	18051	13301	12350	1076
87	88	7323	2121	5202	4015	352

88 rows × 11 columns

In [74]:

```
#Initilizing Foursquare session:

CLIENT_ID = 'BNHS4Y4BKWGLP4DGK5NN3NDXK3MK310IXOBBFNDRUEOCCL4D' # your Foursquare ID
CLIENT_SECRET = 'FXNPOQJUJPKPZETGLNB5E0QOP3LF3YGHPI2B33KG4YQ3VCAG' # your Foursquare Se
cret
VERSION = '20180605' # Foursquare API version

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

```
Your credentails:
CLIENT_ID: BNHS4Y4BKWGLP4DGK5NN3NDXK3MK310IXOBBFNDRUEOCCL4D
CLIENT_SECRET:FXNPOQJUJPKPZETGLNB5E0QOP3LF3YGHPI2B33KG4YQ3VCAG
```

In [75]:

```
neighborhood_name = Demog_df.loc[0, 'Neighborhood'] # neighborhood name
neighborhood_latitude = Demog_df.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = Demog_df.loc[0, 'Longitude'] # neighborhood longitude value

print('Latitude and longitude values of {} are {}, {}.'.format(neighborhood_name,
                                                               neighborhood_latitude,
                                                               neighborhood_longitude))

LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}
&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
url
```

```
Latitude and longitude values of FOURTH WARD are 29.75762000000003, -95.38
448999999997.
```

Out[75]:

```
'https://api.foursquare.com/v2/venues/explore?&client_id=BNHS4Y4BKWGLP4DGK
5NN3NDXK3MK310IXOBBFNDRUEOCCL4D&client_secret=FXNPOQJUJPKPZETGLNB5E0QOP3LF
3YGHPI2B33KG4YQ3VCAG&v=20180605&ll=29.75762000000003,-95.38448999999997&ra
dius=500&limit=100'
```

In [76]:

```
def getNearbyVenues(names, latitudes, longitudes):
    radius=500
    LIMIT=100
    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_secret
={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # 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 venue_
list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
Houston_venues = getNearbyVenues(names=Demog_df['Neighborhood'],
                                 latitudes=Demog_df['Latitude'],
                                 longitudes=Demog_df['Longitude']
                                )
```

Your credentails:
CLIENT_ID: BNHS4Y4BKWGLP4DGK5NN3NDXK3MK310IXOBBFNDRUEOCCL4D
CLIENT_SECRET:FXNPOQJUJPKPZETGLNB5E0QOP3LF3YGHPI2B33KG4YQ3VCAG
FOURTH WARD
SECOND WARD
DOWNTOWN
CLINTON PARK TRI-COMMUNITY
GREATER UPTOWN
GREATER INWOOD
GREATER HOBBY AREA
GOLFCREST / BELLFORT / REVEILLE
ELDRIDGE / WEST OAKS
WASHINGTON AVENUE COALITION / MEMORIAL PARK
GREATER FIFTH WARD
DENVER HARBOR / PORT HOUSTON
PLEASANTVILLE AREA
NORTHSHORE
LAZY BROOK / TIMBERGROVE
GREATER HEIGHTS
KASHMERE GARDENS
MINNETEX
NORTHSIDE VILLAGE
SPRING BRANCH EAST
SPRING BRANCH NORTH
EL DORADO / OATES PRAIRIE
SPRING BRANCH CENTRAL
HUNTERWOOD
SETTEGAST
LANGWOOD
INDEPENDENCE HEIGHTS
CENTRAL NORTHWEST
TRINITY / HOUSTON GARDENS
CARVERDALE
EASTEX - JENSEN AREA
EAST HOUSTON
ACRES HOME
NORTHSIDE/NORTHLINE
HIDDEN VALLEY
EAST LITTLE YORK / HOMESTEAD
WILLOWBROOK
GREATER GREENSPOINT
IAH / AIRPORT AREA
KINGWOOD AREA
LAKE HOUSTON
FAIRBANKS / NORTHWEST CROSSING
WESTBRANCH
SHARPSTOWN
WESTWOOD
FORT BEND / HOUSTON
FONDREN GARDENS
SOUTH BELT / ELLINGTON
SOUTH ACRES / CRESTMONT PARK
BRAYS OAKS
CENTRAL SOUTHWEST
SUNNYSIDE
ALIEF
PECAN PARK
CLEAR LAKE
WESTBURY
WILLOW MEADOWS / WILLOWBEND AREA
BRAEBURN

```
SOUTH MAIN
SOUTH PARK
ASTRODOME AREA
OST / SOUTH UNION
PARK PLACE
MEADOWBROOK / ALLENDALE
MEDICAL CENTER AREA
GULFTON
MACGREGOR
GULFGATE RIVERVIEW / PINE VALLEY
HARRISBURG / MANCHESTER
UNIVERSITY PLACE
WESTCHASE
MUSEUM PARK
LAWNDALE / WAYSIDE
GREENWAY / UPPER KIRBY AREA
GREATER THIRD WARD
MID WEST
GREATER EASTWOOD
MIDTOWN
BRAESWOOD PLACE
MEYERLAND AREA
EDGEBROOK AREA
MAGNOLIA PARK
AFTON OAKS / RIVER OAKS AREA
BRIARFOREST AREA
NEARTOWN - MONTROSE
MEMORIAL
SPRING BRANCH WEST
ADDICKS PARK TEN
```

In [77]:

```
Houston_venues.head()
```

Out[77]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	FOURTH WARD	29.75762	-95.38449	Paper Street Crossfit	29.757435	-95.385846	Gym
1	FOURTH WARD	29.75762	-95.38449	Lucio's BYOB	29.758326	-95.385591	Bar
2	FOURTH WARD	29.75762	-95.38449	Extreme Custom & Classic Cars	29.757708	-95.385589	Auto Garage
3	FOURTH WARD	29.75762	-95.38449	Pat Greer's Kitchen	29.755708	-95.387013	Vegetarian / Vegan Restaurant
4	FOURTH WARD	29.75762	-95.38449	Reflection Pool	29.759820	-95.383963	Park

In [78]:

```
Houston_venues.groupby('Neighborhood').count()
```

Out[78]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
ACRES HOME	1	1	1	1	1	1
AFTON OAKS / RIVER OAKS AREA	2	2	2	2	2	2
ALIEF	3	3	3	3	3	3
ASTRODOME AREA	7	7	7	7	7	7
BRAEBURN	4	4	4	4	4	4
...	...	...	...	...	...	...
WESTBURY	9	9	9	9	9	9
WESTCHASE	1	1	1	1	1	1
WESTWOOD	10	10	10	10	10	10
WILLOW MEADOWS / WILLOWBEND AREA	2	2	2	2	2	2
WILLOWBROOK	18	18	18	18	18	18

80 rows × 6 columns

In [79]:

```
Houston_venues['Venue Category'].unique()[:50]
```

Out[79]:

```
array(['Gym', 'Bar', 'Auto Garage', 'Vegetarian / Vegan Restaurant',
       'Park', 'Mexican Restaurant', 'Fried Chicken Joint',
       'Clothing Store', 'Record Shop', 'Track', 'Bakery',
       'Baseball Stadium', 'Hotel', 'Steakhouse', 'Sports Bar',
       'Italian Restaurant', 'Taco Place', 'Cajun / Creole Restaurant',
       'BBQ Joint', 'Comic Shop', 'Vietnamese Restaurant', 'Beer Garden',
       'Coffee Shop', 'Food Truck', 'Playground', 'French Restaurant',
       'Music Venue', 'Gastropub', 'Peruvian Restaurant',
       'Salon / Barbershop', 'Yoga Studio', 'Breakfast Spot',
       'Mediterranean Restaurant', 'Dog Run', 'Bank',
       'Fast Food Restaurant', 'Pharmacy', 'Gift Shop', 'Cosmetics Shop',
       'Liquor Store', 'Gas Station', 'Pizza Place', 'Ice Cream Shop',
       'Wings Joint', 'Mobile Phone Shop', 'Chinese Restaurant',
       'Kids Store', 'Hardware Store', 'Arcade', 'Shipping Store'],
      dtype=object)
```

In [80]:

```
Houston_onehot = pd.get_dummies(Houston_venues[['Venue Category']], prefix="", prefix_sep="")
Houston_onehot.insert(loc=0, column='Neighborhood', value=Houston_venues['Neighborhood'] )
Houston_onehot.shape
```

Out[80]:

```
(704, 197)
```

In [81]:

```
Houston_grouped = Houston_onehot.groupby('Neighborhood').mean().reset_index()
Houston_grouped.head(88)
```

Out[81]:

	Neighborhood	Adult Boutique	African Restaurant	Airport Terminal	American Restaurant	Arcade	Art Gallery	Art Museum
0	ACRES HOME	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	AFTON OAKS / RIVER OAKS AREA	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	ALIEF	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	ASTRODOME AREA	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	BRAEBURN	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...
75	WESTBURY	0.0	0.0	0.0	0.0	0.0	0.0	0.0
76	WESTCHASE	0.0	0.0	0.0	0.0	0.0	0.0	0.0
77	WESTWOOD	0.0	0.1	0.0	0.0	0.0	0.0	0.0
78	WILLOW MEADOWS / WILLOWBEND AREA	0.0	0.0	0.0	0.0	0.0	0.0	0.0
79	WILLOWBROOK	0.0	0.0	0.0	0.0	0.0	0.0	0.0

80 rows × 197 columns

In [82]:

```
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]
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'] = Houston_grouped['Neighborhood']

for ind in np.arange(Houston_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(Houston_group
ed.iloc[ind, :], num_top_venues)

neighborhoods_venues_sorted.head(88)
```

Out[82]:

	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
0	ACRES HOME	Discount Store	Zoo Exhibit	Electronics Store	Food	Flower Shop	Flea Market
1	AFTON OAKS / RIVER OAKS AREA	Shop & Service	Public Art	Donut Shop	Flower Shop	Flea Market	Financial or Legal Service
2	ALIEF	Football Stadium	Pool	Other Repair Shop	Zoo Exhibit	Electronics Store	Flower Shop
3	ASTRODOME AREA	Food Truck	Moving Target	Business Service	Pizza Place	Auto Garage	Chinese Restaurant
4	BRAEBURN	Pizza Place	Sandwich Place	Supplement Shop	Thrift / Vintage Store	Zoo Exhibit	Dry Cleaner
...	...	...	...	...	...	...	...
75	WESTBURY	Liquor Store	Pizza Place	Tennis Court	Dive Bar	Discount Store	Salon / Barbershop
76	WESTCHASE	Pizza Place	Zoo Exhibit	Dry Cleaner	Flower Shop	Flea Market	Financial or Legal Service
77	WESTWOOD	Fast Food Restaurant	Deli / Bodega	African Restaurant	Flea Market	Filipino Restaurant	Discount Store
78	WILLOW MEADOWS / WILLOWBEND AREA	Food Truck	Soccer Stadium	Zoo Exhibit	Electronics Store	Flower Shop	Flea Market
79	WILLOWBROOK	Furniture / Home Store	Steakhouse	Discount Store	Mobile Phone Shop	Movie Theater	Fast Food Restaurant

80 rows × 11 columns

In [83]:

```
#Create a Dataframe with the Restaurant information for peer clustering
Houston_Rest = Houston_grouped[["Neighborhood","Restaurant"]]
Houston_Rest.head()
```

Out[83]:

	Neighborhood	Restaurant
0	ACRES HOME	0.0
1	AFTON OAKS / RIVER OAKS AREA	0.0
2	ALIEF	0.0
3	ASTRODOME AREA	0.0
4	BRAEBURN	0.0

In [84]:

```
##Cluster Neighborhoods
## Run k-means to cluster the neighborhoods in Houston into 3 clusters.

# set number of clusters
kclusters = 3

Houston_clustering = Houston_Rest.drop(["Neighborhood"], 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(Houston_clustering)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

Out[84]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```

In [85]:

```
# create a new dataframe that includes the cluster as well as the top 10 venues for eac
h neighborhood.
Hou_rest_merged = Houston_Rest.copy()

# add clustering labels
Hou_rest_merged["Cluster Labels"] = kmeans.labels_
Hou_rest_merged.head()
```

Out[85]:

	Neighborhood	Restaurant	Cluster Labels
0	ACRES HOME	0.0	0
1	AFTON OAKS / RIVER OAKS AREA	0.0	0
2	ALIEF	0.0	0
3	ASTRODOME AREA	0.0	0
4	BRAEBURN	0.0	0

In [86]:

```
# merge Houston_rest_merged with Demog_df data to add latitude/longitude for each neighborhood for our clustered data
Hou_rest_merged = Hou_rest_merged.join(Demog_df.set_index("Neighborhood"), on="Neighborhood")

print(Hou_rest_merged.shape)
Hou_rest_merged.head() # check the last columns!
```

```
(80, 36)
```

Out[86]:

	Neighborhood	Restaurant	Cluster Labels	OBJECTID	SUM_POP100	SUM_HU100	SUM_TotPop
0	ACRES HOME	0.0	0	33	24465	9288	24465
1	AFTON OAKS / RIVER OAKS AREA	0.0	0	83	14007	8069	14007
2	ALIEF	0.0	0	53	102235	35498	102235
3	ASTRODOME AREA	0.0	0	61	17697	11311	17697
4	BRAEBURN	0.0	0	58	19341	8216	19341

5 rows × 36 columns

Now, let's visualize the resulting clusters

In [87]:

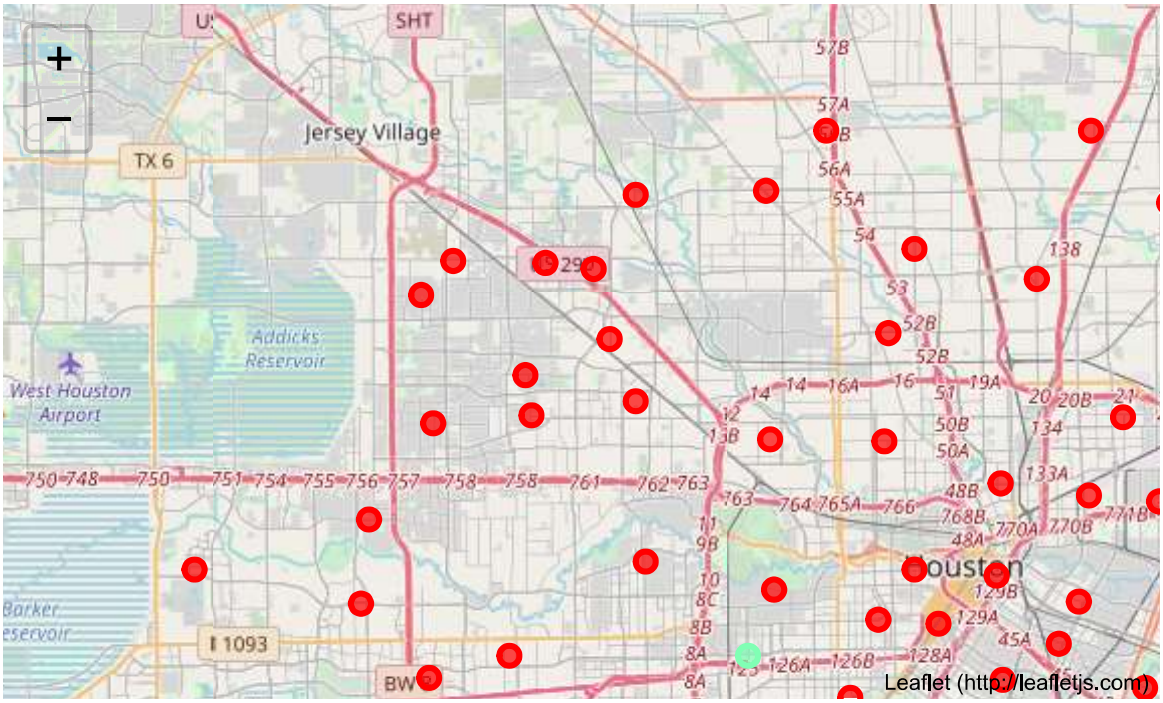
```
# create map
map_clusters = folium.Map(location=[29.749907, -95.358421], zoom_start=11)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(Hou_rest_merged['Latitude'], Hou_rest_merged['Longitude'], Hou_rest_merged['Neighborhood'], Hou_rest_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[87]:



In [88]:

```
# Lets save the map as HTML file
map_clusters.save('HOURest_map_clusters.html')
```

In [89]:

```
# Now let's analize the Latino population by neighborhoods in Houston

HouPop = Hou_rest_merged.sort_values('SUM_HispPop', ascending=False)
HouPop.head(10)
```

Out[89]:

	Neighborhood	Restaurant	Cluster Labels	OBJECTID	SUM_POP100	SUM_HU100	SUI
55	NORTHSIDE/NORTHLINE	0.000000	0	34	58830	18327	
2	ALIEF	0.000000	0	53	102235	35498	
62	SHARPSTOWN	0.052632	1	44	75724	30285	
23	GOLFCREST / BELLFORT / REVEILLE	0.000000	0	8	49757	17530	
10	CENTRAL SOUTHWEST	0.000000	0	51	60857	19004	
64	SOUTH BELT / ELLINGTON	0.000000	0	48	54434	19643	
26	GREATER GREENSPOINT	0.000000	0	38	42793	18633	
54	NORTHSIDE VILLAGE	0.000000	0	19	26831	9664	
67	SPRING BRANCH CENTRAL	0.000000	0	23	28081	9499	
49	MID WEST	0.000000	0	76	47958	25981	

10 rows × 36 columns

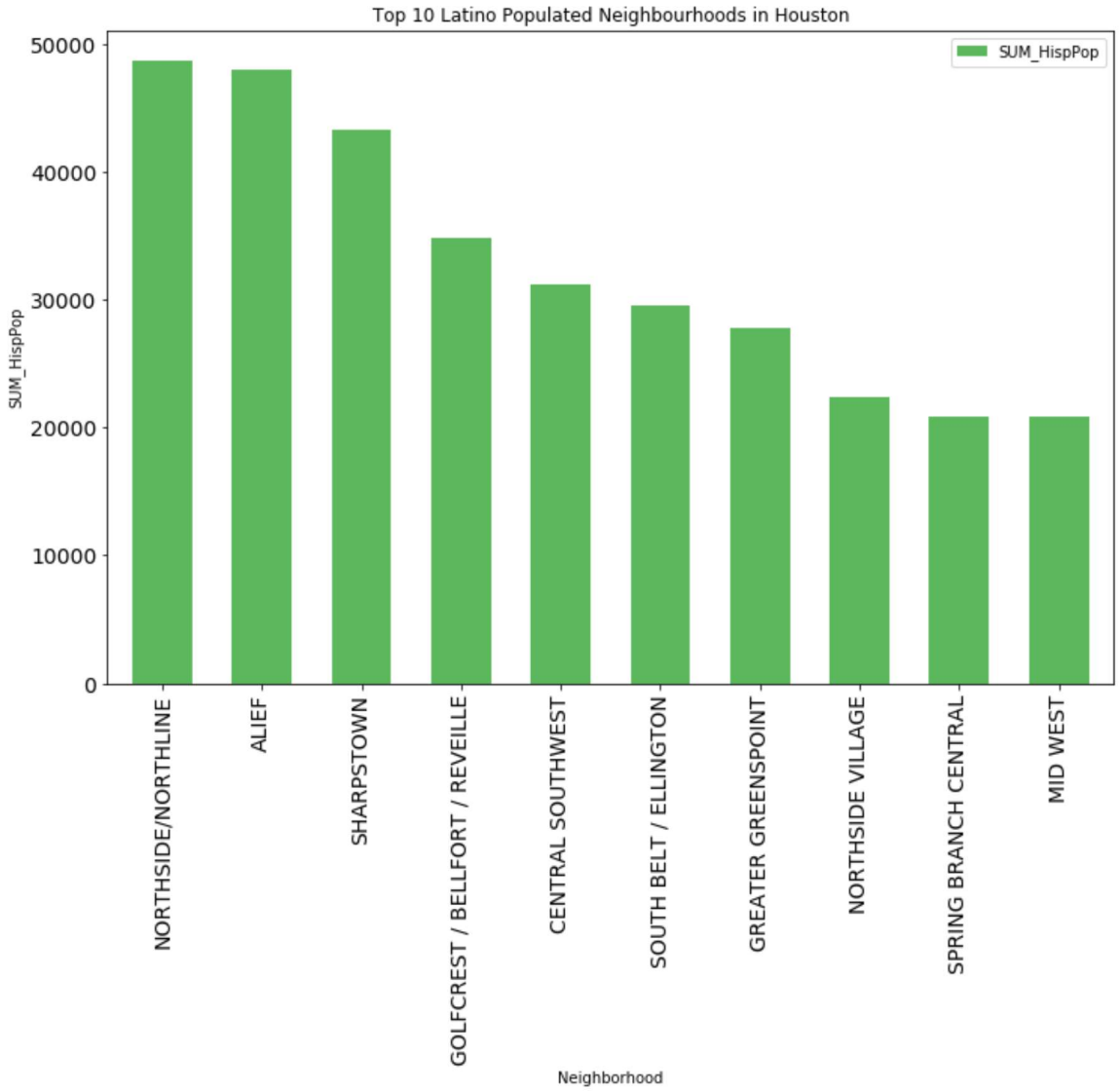
In [90]:

```
#Lets plot the 10 most highly Latino populated neighborhoods in Houston
import matplotlib as mpl
import matplotlib.pyplot as plt
population_chart = HouPop[['Neighborhood', 'SUM_HispPop']].copy()
top_chart = population_chart.head(10)
top_chart.set_index('Neighborhood', inplace=True)

# plot data
colors = ['#5cb85c', '#5bc0de', '#d9534f']
top_chart.plot(kind='bar', figsize=(12, 8), width=0.6, fontsize=14, color=colors)

plt.xlabel('Neighborhood') # add to x-label to the plot
plt.ylabel('SUM_HispPop') # add y-label to the plot
plt.title('Top 10 Latino Populated Neighbourhoods in Houston') # add title to the plot

plt.show()
```



In [31]:

```
# Getting the Houston Income Data per Super Neighborhoods from http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN.pdf
```

In [2]:

```
pip install tabula-py
```

```
Collecting tabula-py
  Downloading https://files.pythonhosted.org/packages/53/a4/66add528eca003
98af98f181772006750019eb9f2d68c7c6fdd53ba661c5/tabula_py-2.1.0-py3-none-an
y.whl (10.4MB)
     |████████████████████████████████| 10.4MB 562kB/s  eta 0:00:01
Collecting distro (from tabula-py)
  Downloading https://files.pythonhosted.org/packages/ea/35/82f79b92fa4d93
7146c660a6482cee4f3dfa1f97ff3d2a6f3ecba33e712e/distro-1.4.0-py2.py3-none-a
ny.whl
Requirement already satisfied: numpy in /home/jupyterlab/conda/envs/pytho
n/lib/python3.6/site-packages (from tabula-py) (1.14.2)
Requirement already satisfied: pandas>=0.25.3 in /home/jupyterlab/conda/en
vs/python/lib/python3.6/site-packages (from tabula-py) (1.0.0)
Requirement already satisfied: pytz>=2017.2 in /home/jupyterlab/conda/env
s/python/lib/python3.6/site-packages (from pandas>=0.25.3->tabula-py) (201
9.3)
Requirement already satisfied: python-dateutil>=2.6.1 in /home/jupyterlab/
conda/envs/python/lib/python3.6/site-packages (from pandas>=0.25.3->tabula
-py) (2.8.1)
Requirement already satisfied: six>=1.5 in /home/jupyterlab/conda/envs/pyt
hon/lib/python3.6/site-packages (from python-dateutil>=2.6.1->pandas>=0.2
5.3->tabula-py) (1.14.0)
Installing collected packages: distro, tabula-py
Successfully installed distro-1.4.0 tabula-py-2.1.0
Note: you may need to restart the kernel to use updated packages.
```

In [3]:

```
import tabula
from tabula import read_pdf
import ssl
```

In [4]:

```
# Converting the pdf Income data into a CSV
tabula.convert_into( "http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN.pdf", "Median_Household_Income_by_SN1.csv", pages =1, output_format="csv")
```

```
Got stderr: Mar 18, 2020 9:08:21 PM org.apache.pdfbox.pdmodel.font.FileSystemFontProvider loadDiskCache
WARNING: New fonts found, font cache will be re-built
Mar 18, 2020 9:08:21 PM org.apache.pdfbox.pdmodel.font.FileSystemFontProvider <init>
WARNING: Building on-disk font cache, this may take a while
Mar 18, 2020 9:08:23 PM org.apache.pdfbox.pdmodel.font.FileSystemFontProvider <init>
WARNING: Finished building on-disk font cache, found 18 fonts
Mar 18, 2020 9:08:23 PM org.apache.pdfbox.pdmodel.font.PDTrueTypeFont <init>
WARNING: Using fallback font 'LiberationSans' for 'Times New Roman,Italic'
Mar 18, 2020 9:08:24 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFO: OpenType Layout tables used in font ABCDEE+Georgia,Italic are not implemented in PDFBox and will be ignored
Mar 18, 2020 9:08:35 PM org.apache.pdfbox.pdmodel.font.PDTrueTypeFont <init>
WARNING: Using fallback font 'LiberationSans' for 'Times New Roman,Italic'
Mar 18, 2020 9:08:35 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFO: OpenType Layout tables used in font ABCDEE+Georgia,Italic are not implemented in PDFBox and will be ignored
Mar 18, 2020 9:08:45 PM org.apache.pdfbox.pdmodel.font.PDTrueTypeFont <init>
WARNING: Using fallback font 'LiberationSans' for 'Times New Roman,Italic'
Mar 18, 2020 9:08:45 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFO: OpenType Layout tables used in font ABCDEE+Georgia,Italic are not implemented in PDFBox and will be ignored
```

In [5]:

```
tabula.convert_into( "http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN.pdf", "Median_Household_Income_by_SN2.csv", pages =2, output_format="csv")
```

In [6]:

```
tabula.convert_into( "http://www.houstontx.gov/planning/Demographics/docs_pdfs/SN/Median_Household_Income_by_SN.pdf", "Median_Household_Income_by_SN3.csv", pages =3, output_format="csv")
```

In [35]:

```
import pandas as pd

MHI_SN1DF = pd.read_csv("Median_Household_Income_by_SN1.csv", thousands=',')
# Preview the first 5 lines of the loaded data
MHI_SN1DF.head()
```

Out[35]:

	SN #	Unnamed: 1	Super Neighborhood Name	Median Income
0	1	NaN	Willowbrook	$36,498
1	2	NaN	Greater Greenspoint	$27,334
2	3	NaN	Carverdale	$56,139
3	4	NaN	Fairbanks / Northwest Crossing	$37,278
4	5	NaN	Greater Inwood	$39,086

In [36]:

```
# Clean the nill fields in the dataframe

MHI_SN1DF.drop(["Unnamed: 1"], axis=1, inplace=True)
MHI_SN1DF.head()
```

Out[36]:

	SN #	Super Neighborhood Name	Median Income
0	1	Willowbrook	$36,498
1	2	Greater Greenspoint	$27,334
2	3	Carverdale	$56,139
3	4	Fairbanks / Northwest Crossing	$37,278
4	5	Greater Inwood	$39,086

In [37]:

```
MHI_SN2DF = pd.read_csv("Median_Household_Income_by_SN2.csv", thousands=',')
# Preview the first 5 lines of the loaded data
MHI_SN2DF.head()
```

Out[37]:

	SN #	Super Neighborhood Name	Median Income
0	31	Meyerland Area	$71,479
1	32	Braeswood	$82,535
2	33	Medical Center Area	$59,497
3	34	Astrodome Area	$43,607
4	35	South Main	$33,488

In [38]:

```
MHI_SN3DF = pd.read_csv("Median_Household_Income_by_SN3.csv", thousands=',')
# Preview the first 5 lines of the loaded data
MHI_SN3DF.head()
```

Out[38]:

	SN #	Super Neighborhood Name	Median Income
0	71.0	Sunnyside	$24,462
1	72.0	South Park	$31,589
2	73.0	Golfcrest / Bellfort / Reveille	$37,696
3	74.0	Park Place	$32,447
4	75.0	Meadowbrook / Allendale	$40,775

In [39]:

```
# Let's join all 3 Income dataframes into one

MHI_SN1DF = MHI_SN1DF.append(MHI_SN2DF, ignore_index=True)
```

In [40]:

```
MHI_SN1DF = MHI_SN1DF.append(MHI_SN3DF, ignore_index=True)
MHI_SN1DF.head(88)
```

Out[40]:

	SN #	Super Neighborhood Name	Median Income
0	1.0	Willowbrook	$36,498
1	2.0	Greater Greenspoint	$27,334
2	3.0	Carverdale	$56,139
3	4.0	Fairbanks / Northwest Crossing	$37,278
4	5.0	Greater Inwood	$39,086
...	...	...	...
83	84.0	Spring Branch North	$43,795
84	85.0	Spring Branch Central	$36,252
85	86.0	Spring Branch East	$43,397
86	87.0	Greenway / Upper Kirby Area	$77,323
87	88.0	Lawndale / Wayside	$33,168

88 rows × 3 columns

In [41]:

```
# call the function to get the coordinates, store in a new list using list comprehensio
n
coordsInc = [ get_latlng(neighborhood) for neighborhood in MHI_SN1DF ["Super Neighborho
od Name"].tolist() ]

# create temporary dataframe to populate the coordinates into Latitude and Longitude
coordsInc_df = pd.DataFrame(coords, columns=['Latitude', 'Longitude'])

#Include the coordinates in the dataframe
# merge the coordinates into the original dataframe
MHI_SN1DF['Latitude'] = coordsInc_df['Latitude']
MHI_SN1DF['Longitude'] = coordsInc_df['Longitude']

MHI_SN1DF.head(88)
```

Out[41]:

	SN #	Super Neighborhood Name	Median Income	Latitude	Longitude
0	1.0	Willowbrook	$36,498	29.75762	-95.38449
1	2.0	Greater Greenspoint	$27,334	29.74848	-95.32843
2	3.0	Carverdale	$56,139	29.75595	-95.35679
3	4.0	Fairbanks / Northwest Crossing	$37,278	29.74322	-95.25751
4	5.0	Greater Inwood	$39,086	29.76015	-95.47671
...	...	...	...	...	...
83	84.0	Spring Branch North	$43,795	29.74768	-95.57424
84	85.0	Spring Branch Central	$36,252	29.74280	-95.39723
85	86.0	Spring Branch East	$43,397	29.77263	-95.57129
86	87.0	Greenway / Upper Kirby Area	$77,323	29.80110	-95.54933
87	88.0	Lawndale / Wayside	$33,168	29.81402	-95.61619

88 rows × 5 columns

In [42]:

```
MHI_SN1DF = MHI_SN1DF.append(MHI_SN2DF, ignore_index=True)
MHI_SN1DF = MHI_SN1DF.append(MHI_SN3DF, ignore_index=True)
MHI_SN1DF.head(88)
```

Out[42]:

	SN #	Super Neighborhood Name	Median Income	Latitude	Longitude
0	1.0	Willowbrook	$36,498	29.75762	-95.38449
1	2.0	Greater Greenspoint	$27,334	29.74848	-95.32843
2	3.0	Carverdale	$56,139	29.75595	-95.35679
3	4.0	Fairbanks / Northwest Crossing	$37,278	29.74322	-95.25751
4	5.0	Greater Inwood	$39,086	29.76015	-95.47671
...	...	...	...	...	...
83	84.0	Spring Branch North	$43,795	29.74768	-95.57424
84	85.0	Spring Branch Central	$36,252	29.74280	-95.39723
85	86.0	Spring Branch East	$43,397	29.77263	-95.57129
86	87.0	Greenway / Upper Kirby Area	$77,323	29.80110	-95.54933
87	88.0	Lawndale / Wayside	$33,168	29.81402	-95.61619

88 rows × 5 columns

In [43]:

```
#Standarizing the neighborhoods names into uppercase for facilitationg joining datafram
es
MHI_SN1DF['Super Neighborhood Name'] = MHI_SN1DF['Super Neighborhood Name'].str.upper()

MHI_SN1DF.head()
```

Out[43]:

	SN #	Super Neighborhood Name	Median Income	Latitude	Longitude
0	1.0	WILLOWBROOK	$36,498	29.75762	-95.38449
1	2.0	GREATER GREENSPOINT	$27,334	29.74848	-95.32843
2	3.0	CARVERDALE	$56,139	29.75595	-95.35679
3	4.0	FAIRBANKS / NORTHWEST CROSSING	$37,278	29.74322	-95.25751
4	5.0	GREATER INWOOD	$39,086	29.76015	-95.47671

In [44]:

```
#Standarizing the neighborhoods field name in MHI_SNDF into "Neigborhood" for facilitat
iong joining dataframes
MHI_SN1DFM = MHI_SN1DF.rename(columns={"Super Neighborhood Name":"Neighborhood"})
MHI_SN1DFM.head(5)
```

Out[44]:

	SN #	Neighborhood	Median Income	Latitude	Longitude
0	1.0	WILLOWBROOK	$36,498	29.75762	-95.38449
1	2.0	GREATER GREENSPOINT	$27,334	29.74848	-95.32843
2	3.0	CARVERDALE	$56,139	29.75595	-95.35679
3	4.0	FAIRBANKS / NORTHWEST CROSSING	$37,278	29.74322	-95.25751
4	5.0	GREATER INWOOD	$39,086	29.76015	-95.47671

In [45]:

```
#Merge the Population and Incomes data in a single dataframe and clean any duplicate re
cords from the merge.
MHI_SN1DFM1 = pd.merge(MHI_SN1DFM, HouPop,  on='Neighborhood', how='outer')

MHI_SN1DFM1 = MHI_SN1DFM1[pd.notnull(MHI_SN1DFM1['Latitude_x'])]
MHI_SN1DFM1.head(170)
```

Out[45]:

	SN #	Neighborhood	Median Income	Latitude_x	Longitude_x	Restaurant	Cluster Labels	OBJECTID
0	1.0	WILLOWBROOK	$36,498	29.75762	-95.38449	0.000000	0.0	37.0
1	2.0	GREATER GREENSPOINT	$27,334	29.74848	-95.32843	0.000000	0.0	38.0
2	3.0	CARVERDALE	$56,139	29.75595	-95.35679	0.000000	0.0	30.0
3	4.0	FAIRBANKS / NORTHWEST CROSSING	$37,278	29.74322	-95.25751	0.000000	0.0	42.0
4	5.0	GREATER INWOOD	$39,086	29.76015	-95.47671	0.000000	0.0	6.0
...	...	...	...	...	...	...	...	...
136	84.0	SPRING BRANCH NORTH	$43,795	29.74768	-95.57424	0.000000	0.0	21.0
138	85.0	SPRING BRANCH CENTRAL	$36,252	29.74280	-95.39723	0.000000	0.0	23.0
140	86.0	SPRING BRANCH EAST	$43,397	29.77263	-95.57129	0.000000	0.0	20.0
142	87.0	GREENWAY / UPPER KIRBY AREA	$77,323	29.80110	-95.54933	0.034483	2.0	74.0
144	88.0	LAWNDALE / WAYSIDE	$33,168	29.81402	-95.61619	0.000000	0.0	73.0

88 rows × 40 columns

In [46]:

```
#Sort the neighborhoods per Median Income
MHI_SN1DFM1.sort_values(['Median Income'], ascending=False, axis=0, inplace=True)

# get the top 10 highest Median Income neigborhoods in the dataframe
MHI_SN1DFM1_top10 = MHI_SN1DFM1.head(10)

MHI_SN1DFM1_top10.head(170)
```

Out[46]:

	SN #	Neighborhood	Median Income	Latitude_x	Longitude_x	Restaurant	Cluster Labels	OBJECTID
56	44.0	LAKE HOUSTON	$96,869	29.701260	-95.517980	0.000000	0.0	41.0
54	43.0	KINGWOOD AREA	$95,916	29.839370	-95.553610	0.000000	0.0	40.0
21	22.0	WASHINGTON AVENUE COALITION / MEMORIAL PARK	$89,474	29.807480	-95.240520	0.000000	0.0	10.0
20	21.0	GREATER UPTOWN	$84,539	29.803517	-95.515896	0.000000	0.0	5.0
92	62.0	MIDTOWN	$82,877	29.693880	-95.353010	0.000000	0.0	78.0
32	32.0	BRAESWOOD	$82,535	29.844770	-95.255830	NaN	NaN	NaN
130	81.0	CLEAR LAKE	$81,315	29.642620	-95.226860	0.000000	0.0	55.0
142	87.0	GREENWAY / UPPER KIRBY AREA	$77,323	29.801100	-95.549330	0.034483	2.0	74.0
30	31.0	MEYERLAND AREA	$71,479	29.843980	-95.342880	0.000000	0.0	80.0
23	24.0	NEARTOWN - MONTROSE	$68,523	29.819610	-95.210380	0.000000	0.0	85.0

10 rows × 40 columns

In [47]:

```
#Extract the required fields from the top 10 listing

MHI_SN1DFM1_T10 = MHI_SN1DFM1_top10[['Neighborhood','Median Income','SUM_HispPop','Lati
tude_x','Longitude_x','Restaurant','Cluster Labels']].copy()

#MHI_SN1DFM1_T10['Median Income'] = MHI_SN1DFM1_T10['Median Income'].astype(float)
MHI_SN1DFM1_T10.head()
```

Out[47]:

	Neighborhood	Median Income	SUM_HispPop	Latitude_x	Longitude_x	Restaurant	Cluster Labels
56	LAKE HOUSTON	$96,869	2586.0	29.701260	-95.517980	0.0	0.0
54	KINGWOOD AREA	$95,916	7093.0	29.839370	-95.553610	0.0	0.0
21	WASHINGTON AVENUE COALITION / MEMORIAL PARK	$89,474	7890.0	29.807480	-95.240520	0.0	0.0
20	GREATER UPTOWN	$84,539	7311.0	29.803517	-95.515896	0.0	0.0
92	MIDTOWN	$82,877	1309.0	29.693880	-95.353010	0.0	0.0

In [48]:

```
print(MHI_SN1DFM1_T10.dtypes)
```

```
Neighborhood       object
Median Income      object
SUM_HispPop       float64
Latitude_x        float64
Longitude_x       float64
Restaurant        float64
Cluster Labels    float64
dtype: object
```

In [49]:

```
MHI_SN1DFM1_T10['Median Income_Obj'] = MHI_SN1DFM1_T10['Median Income'].str.replace('$'
, '').astype(object)
MHI_SN1DFM1_T10['Median Income_Float'] = MHI_SN1DFM1_T10['Median Income_Obj'].str.repla
ce(',', '').astype(float)

MHI_SN1DFM1_T10.dropna(subset=['Median Income_Obj'],inplace=True)
```

In [50]:

```
print(MHI_SN1DFM1_T10.dtypes)
```

```
Neighborhood            object
Median Income           object
SUM_HispPop            float64
Latitude_x             float64
Longitude_x            float64
Restaurant             float64
Cluster Labels         float64
Median Income_Obj       object
Median Income_Float    float64
dtype: object
```

In [51]:

```
# Calculate the potential market size for these top 10 Listing Formula== Median Income
 x Hispanic population= Potential Market size
MHI_SN1DFM1_T10['PotMktSize'] = MHI_SN1DFM1_T10['Median Income_Float']* MHI_SN1DFM1_T10
['SUM_HispPop']
MHI_SN1DFM1_T10 = MHI_SN1DFM1_T10.sort_values(by=['PotMktSize'], ascending=False)
MHI_SN1DFM1_T10.head(10)
```

Out[51]:

	Neighborhood	Median Income	SUM_HispPop	Latitude_x	Longitude_x	Restaurant	Cluster Labels	In
130	CLEAR LAKE	$81,315	9623.0	29.642620	-95.226860	0.000000	0.0	
21	WASHINGTON AVENUE COALITION / MEMORIAL PARK	$89,474	7890.0	29.807480	-95.240520	0.000000	0.0	
54	KINGWOOD AREA	$95,916	7093.0	29.839370	-95.553610	0.000000	0.0	
20	GREATER UPTOWN	$84,539	7311.0	29.803517	-95.515896	0.000000	0.0	
23	NEARTOWN - MONTROSE	$68,523	5738.0	29.819610	-95.210380	0.000000	0.0	
56	LAKE HOUSTON	$96,869	2586.0	29.701260	-95.517980	0.000000	0.0	
142	GREENWAY / UPPER KIRBY AREA	$77,323	2620.0	29.801100	-95.549330	0.034483	2.0	
30	MEYERLAND AREA	$71,479	2756.0	29.843980	-95.342880	0.000000	0.0	
92	MIDTOWN	$82,877	1309.0	29.693880	-95.353010	0.000000	0.0	
32	BRAESWOOD	$82,535	NaN	29.844770	-95.255830	NaN	NaN	

In [52]:

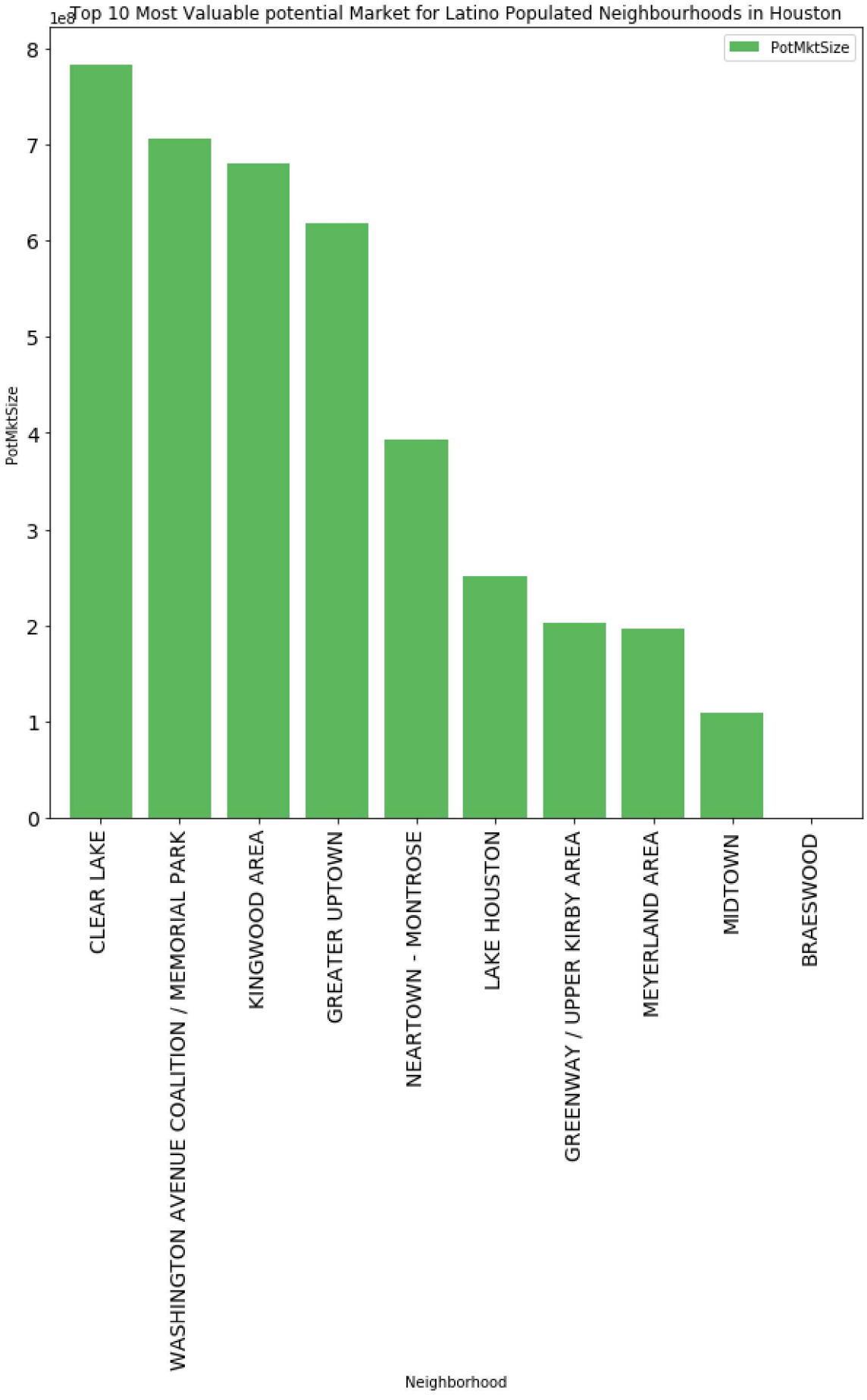
```
#Lets plot the 10 most highly valuable Market-neighborhoods in Houston
import matplotlib as mpl
import matplotlib.pyplot as plt
population_chart = MHI_SN1DFM1_T10[['Neighborhood', 'PotMktSize']].copy()
top_chart = population_chart.head(10)
top_chart.set_index('Neighborhood', inplace=True)

# plot data
colors = ['#5cb85c', '#5bc0de', '#d9534f']
top_chart.plot(kind='bar', figsize=(10, 10), width=0.8, fontsize=14, color=colors)

plt.xlabel('Neighborhood') # add to x-label to the plot
plt.ylabel('PotMktSize') # add y-label to the plot
plt.title('Top 10 Most Valuable potential Market for Latino Populated Neighbourhoods in
Houston') # add title to the plot



plt.show()
```


Top 10 Most Valuable potential Market for Latino Populated Neighbourhoods in Houston
1e8
PotMktSize
8
7
6
5
4
3
2
1
0
PotMktSize
CLEAR LAKE
WASHINGTON AVENUE COALITION / MEMORIAL PARK
KINGWOOD AREA
GREATER UPTOWN
NEARTOWN - MONTROSE
LAKE HOUSTON
GREENWAY / UPPER KIRBY AREA
MEYERLAND AREA
MIDTOWN
BRAESWOOD
Neighborhood

In [53]:

```
# Let's Examine the Clusters
#Cluster 0
MHI_SN1DFM1_T10.loc[MHI_SN1DFM1_T10['Cluster Labels'] == 0].sort_values(by=['PotMktSiz
e'], ascending=False)
```

Out[53]:

	Neighborhood	Median Income	SUM_HispPop	Latitude_x	Longitude_x	Restaurant	Cluster Labels	In
130	CLEAR LAKE	$81,315	9623.0	29.642620	-95.226860	0.0	0.0	
21	WASHINGTON AVENUE COALITION / MEMORIAL PARK	$89,474	7890.0	29.807480	-95.240520	0.0	0.0	
54	KINGWOOD AREA	$95,916	7093.0	29.839370	-95.553610	0.0	0.0	
20	GREATER UPTOWN	$84,539	7311.0	29.803517	-95.515896	0.0	0.0	
23	NEARTOWN - MONTROSE	$68,523	5738.0	29.819610	-95.210380	0.0	0.0	
56	LAKE HOUSTON	$96,869	2586.0	29.701260	-95.517980	0.0	0.0	
30	MEYERLAND AREA	$71,479	2756.0	29.843980	-95.342880	0.0	0.0	
92	MIDTOWN	$82,877	1309.0	29.693880	-95.353010	0.0	0.0	

In [54]:

```
#Cluster 1
MHI_SN1DFM1_T10.loc[MHI_SN1DFM1_T10['Cluster Labels'] == 1]
```

Out[54]:

	Neighborhood	Median Income	SUM_HispPop	Latitude_x	Longitude_x	Restaurant	Cluster Labels	M Incom

In [55]:

```
#Cluster 2
MHI_SN1DFM1_T10.loc[MHI_SN1DFM1_T10['Cluster Labels'] == 2]
```

Out[55]:

	Neighborhood	Median Income	SUM_HispPop	Latitude_x	Longitude_x	Restaurant	Cluster Labels	In
142	GREENWAY / UPPER KIRBY AREA	$77,323	2620.0	29.8011	-95.54933	0.034483	2.0	

In [56]:

```
# Lets analyze the top Three largest Markets on regards to venues

neighborhoods_venues_sorted.loc[(neighborhoods_venues_sorted['Neighborhood'] == "CLEAR
 LAKE") ]
```

Out[56]:

	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 Mos Commo Venu
11	CLEAR LAKE	Boutique	Spa	Playground	Salon / Barbershop	Outdoors & Recreation	Dry Cleaner	Fle Marke

In [57]:

```
neighborhoods_venues_sorted.loc[(neighborhoods_venues_sorted['Neighborhood'] == "WASHIN
GTON AVENUE COALITION / MEMORIAL PARK")]
```

Out[57]:

	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 Mo Comm Ven
73	WASHINGTON AVENUE COALITION / MEMORIAL PARK	Construction & Landscaping	Video Store	Mexican Restaurant	Zoo Exhibit	Electronics Store	Food	Flow Sh

In [58]:

```
neighborhoods_venues_sorted.loc[(neighborhoods_venues_sorted['Neighborhood'] == "KINGWOOD AREA")]
```

Out[58]:

	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
38	KINGWOOD AREA	Pizza Place	Sandwich Place	Fast Food Restaurant	Burger Joint	Fried Chicken Joint	Coffee Shop	Chinese Restaurant

In [59]:

```
# create map for the proposed location = CLEAR LAKE
map_clusters2 = folium.Map(location=[29.642620,-95.226860], zoom_start=15)


# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(MHI_SN1DFM1_T10['Latitude_x'], MHI_SN1DFM1_T10['Longitude_x'], MHI_SN1DFM1_T10['Neighborhood'], MHI_SN1DFM1_T10['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        fill=True,
        fill_opacity=0.7).add_to(map_clusters)

map_clusters2
#'Neighborhood','Median Income','SUM_HispPop','Latitude_x','Longitude_x','Restaurant','Cluster Labels'
```

Out[59]:

In [60]:

```
# Lets save the map as HTML file
map_clusters2.save('ProposedLocation.html')
```

As per the information analyzed and the Market sizes it seems clear that the Latin American restaurant could be make good business sense in any of the top 3 Super Neighborhoods we identified above (CLEAR LAKE, WASHINGTON AVENUE COALITION / MEMORIAL PARK & KINGWOOD AREA).

Nevertheless, CLEAR LAKE seems to offer a larger potential market with less competition from other Latin American restaurants. The area near Wilson Memorial Park (Gilpin Street) looks very suitable (Subject to planning approvals) for a restaurant location.

This will be our preferred recommended location for the potential investors in the restaurant.

In [ ]:

```

```