Music-Related Venue Analysis: New York City

Greyson Ford, Nelson Leung, Faz Naimov

Columbia Data Analytics Bootcamp

Hypothesis

We expect music-related venues in New York City to have clustered together to create pockets of similar type venues in the same area. To that end, we expect to see the Midtown and Chelsea among the top neighborhoods for music-related venues based on their legacy as music hot spots and popular nightclub scenes. We expect larger venues to receive the greatest number of likes (our popularity measure) because of the sheer volume of customers - but we feel this is an important trend in music venues to observe.

Key Questions

1 - What are the most popular neighborhoods for music venues in NYC?

2 - What are the most popular venues by category?

3 - Which neighborhoods have the most highly rated/most liked venues? And is there a relationship between likes and ratings?

We wanted to understand which venues were popular, in general, and which were popular by neighborhood.

Our Findings

We were generally able to answer all of our questions.

- We found **1,226 music-related venues** in NYC (as defined by our selected FourSquare CategoryIDs).
- The most popular neighborhood for music-related venues was **University Heights** with 23 venues in the same neighborhood.
- While there were **few Manhattan neighborhoods represented in the top 20 neighborhoods**, they dominated in terms of popularity, or total number of likes.
- **Lounges, Music Venues, and Nightclubs** were the top 3 venue categories among the most liked venues.
- Positive correlation between likes and rating once the rating goes beyond 7 that you can see on the scatter plot below. Also **8 out of top 10 neighborhoods with average likes are located in Manhattan** as well as **7 out of the 10 locations for highest average rating**.

Data Exploration

Foursquare — 'Places API'

We used Foursquare's 'Places API' to acquire data related to music-related 'venues' (as defined by Foursquare).

Each Foursquare 'venue' is assigned a 'category' and each 'category' is associated with a particular 'categoryID.'

Foursquare Music-Related Venue CategoryIDs

```
'4bf58dd8d48988d1e5931735',# Music Venue
'4bf58dd8d48988d1e7931735',# Jazz Club
'4bf58dd8d48988d1e8931735',# Piano Bar
'4bf58dd8d48988d1e9931735',# Rock Club
'5267e4d9e4b0ec79466e48d1',# Music Festival
'56aa371be4b08b9a8d5734db',# Amphitheater
'5032792091d4c4b30a586d5c',# Concert Hall
'4d4b7105d754a06376d81259',# Nightlife Spot
'4bf58dd8d48988d120941735',# Kareoke Bar
```

Data Exploration

NYU's Spatial Data Repository

Our search was focused on the 5 boroughs of New York City. We used the *2014 New York City Neighborhood Names'* dataset hosted by NYU's Spatial Data Repository as the basis for defining neighborhoods' location (via location centroids) and name.

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude
0	Wakefield	40.894705	-73.847201
1	Wakefield	40.894705	-73.847201
2	Wakefield	40.894705	-73.847201
3	Wakefield	40.894705	-73.847201
4	Wakefield	40.894705	-73.847201

Data Exploration

Neighborhood Name & Location Centroid Data

We downloaded the '2014 New York City Neighborhood Names' dataset hosted by NYU's Spatial Data Repository as a JSON file and imported into a Jupyter Notebook as a Pandas dataframe. Then we used a loop to categorize the data into specific neighborhoods that returned neighborhood, latitude, and longitude. This serves as the foundation of the analysis.

```
column_names = ['Borough','Neighborhood', 'Latitude', 'Longitude']
neighborhoods = pd.DataFrame(columns=column_names)

for data in nycdata:
    borough = nycdata = data['properties']['borough']
    neighborhood_name = data['properties']['name']

    neighborhoodlatlon = data['geometry']['coordinates']
    lat = neighborhoodlatlon[1]
    lng = neighborhoodlatlon[0]

    neighborhoods = neighborhoods.append({'Borough':borough,
                                          'Neighborhood':neighborhood_name,
                                          'Latitude': lat,
                                          'Longitude': lng}, ignore_index=True)

neighborhoods.head()
```

Data Exploration

Music-related Venues: Category IDs and 'Get' Requests

Foursquare has numerous "Venue Categories" that are used to identify each type of venue.

A preliminary dataset of music-related venues associated with each NYC neighborhood was created by recursively sending "get" requests to the 'api.foursquare.com/v2/venues/search?' endpoint, making sure the results are specific to venues with music-related "Category IDs."

For each neighborhood, we included all of the selected category IDs in a single 'get' request by passing them as comma separated values.

Data Cleansing

Isolating New York City Venues

First, venues located in states other than “New York” or “NY” were removed. Entries with “Venue State” equal to “New York” were changed to “NY.”

```
ny_venue_data = prelim_venue_data[(prelim_venue_data['Venue State']=='New York') | (prelim_venu
ny_venue_data['Venue State'].replace(to_replace='New York', value = 'NY', inplace=True)
delta = prelim_venue_data.shape[0] - ny_venue_data.shape[0]
print (f'{delta} entries were removed from the preliminary dataset based on "Venue State"')
ny_venue_data.head()
```

```
19 entries were removed from the preliminary dataset based on "Venue State"
```

Data Cleansing

Duplicate Venues & N/A

Some venues are assigned to multiple neighborhoods because the venue is within 1,000 meters of the neighborhood's centroid location.

We chose an inclusive model which counts these venues in each neighborhood in order to properly size each neighborhood's music profile without excluding any businesses that contribute to the identity of the area. Entries returned by Foursquare with no 'Venue City' and given the 'N/A' treatment were also removed

```
ny_venue_data_with_city = ny_venue_data[(ny_venue_data['Venue City'] != "N/A"
delta = ny_venue_data.shape[0] - ny_venue_data_with_city.shape[0]
print (f'{delta} entries were removed based on "Venue City"')
ny_venue_data_with_city.head()
```

Data Cleansing

Removing Venues with that were not music venues

Although specific category IDs were used in our API calls, the data showed a significant amount of venues that did not fall within our categories. We utilized a filter to keep only venues that we were interested in.

```
music_related_categories = ['Music Venue', 'Lounge', 'Nightclub', 'Jazz Club', 'Recording Studi
ny_music_venues = ny_venue_data_with_city[ny_venue_data_with_city['Venue Category'].isin(music_
delta = ny_venue_data_with_city.shape[0] - ny_music_venues.shape[0]
print (f'{delta} entries were removed based on "Venue Category" not being related to music')
print (ny_music_venues['Venue Category'].unique())
ny_music_venues.head()
```

```
7001 entries were removed based on "Venue Category" not being related to music
['Music Venue' 'Nightclub' 'Lounge' 'Jazz Club' 'Piano Bar' 'Concert Hall'
 'Rock Club' 'Amphitheater' 'Music Festival' 'Record Shop' 'Opera House'
 'Recording Studio' 'Music School']
```

Data Enrichment

Music-related Venues: Adding Likes and Ratings

After cleaning our initial data we enriched it with additional user inputs with a second API call to a different Foursquare endpoint.

We wrote a function that accepted index values when called to work within the premium API call limit of 500 per day. By designing the function in this manner, each team member was able to utilize their respective API keys to retrieve a specific set of venues to complete our data set.

```
def venue_info(df, start_index, end_index, v=20180405):
    responses = []
    print("Beginning Data Retrieval")
    print("------------------------------")

    #start_index = 0 # index from which we begin polling venue information due to rate limited
    #end_index = 0 # index at which we stop polling, or earlier if failure returned from 4sq

    for index in range(start_index, end_index):
        id = df.loc[index,"Venue ID"]
        print ('Attempting to get information for venue with id = ', id)

        url = f"https://api.foursquare.com/v2/venues/{id}?&v={v}&client_id={CLIENT_ID}&client_s

        response = requests.get(url)

        if response.status_code != 200:
            print ('Failure response from Foursquare API, breaking...')
            break
        else:
            result = response.json()['response']
            responses.append(result)
            try:
                df.loc[index,"Likes"]=result['venue']['likes']['count']
            except:
                df.loc[index,"Likes"]=0
            try:
                df.loc[index,"Rating"]=result['venue']['rating']
            except:
                df.loc[index,"Rating"]="N/A"

    print("Data Retrieval Complete")
```

Final Data Frame

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue Name	Venue ID	Venue Category	Venue Latitude	Venue Longitude	Venue City	Venue State	Likes	Rating
5	Wakefield	40.894705	-73.847201	Wreckhouse	4d82dc4f8de9721e671f1151	Music Venue	40.898510	-73.855356	Bronx	NY	0.0	NaN
6	Wakefield	40.894705	-73.847201	Matic Records	4efb879502d5a2b50e03c616	Music Venue	40.899774	-73.857141	Bronx	NY	0.0	NaN
7	Wakefield	40.894705	-73.847201	Oasis HQ NightClub	5066b2d7e4b073ee75be2315	Nightclub	40.891920	-73.858484	Bronx	NY	0.0	NaN
8	Wakefield	40.894705	-73.847201	Karma Lounge	4f349355e4b0debe1e3c6118	Nightclub	40.901605	-73.850720	Bronx	NY	1.0	NaN
9	Wakefield	40.894705	-73.847201	Emma C. Brisbane Youth Leadership Center	519beb25498eb0c55bccdd53	Lounge	40.887923	-73.853688	Bronx	NY	0.0	NaN
10	Wakefield	40.894705	-73.847201	Wenbley Lounge	4f9339c7e4b0532d5da3d0ab	Lounge	40.897526	-73.859468	Bronx	NY	0.0	NaN
11	Co-op City	40.874294	-73.829939	Dreiser Auditorium	561175b5498ea48f07080f55	Jazz Club	40.876381	-73.829095	Bronx	NY	0.0	NaN
12	Co-op City	40.874294	-73.829939	Blok Work Studio	4b809653f964a520037f30e3	Music Venue	40.881773	-73.826736	Bronx	NY	0.0	NaN
13	Co-op City	40.874294	-73.829939	Damatrixstudios	4e97653c775be791ae1e9d01	Music Venue	40.871173	-73.836846	Bronx	NY	3.0	NaN
14	Co-op City	40.874294	-73.829939	The Garage	4eaccfbc775bad292f2d50a5	Nightclub	40.882165	-73.827593	Bronx	NY	10.0	6.7

Insights and Analysis

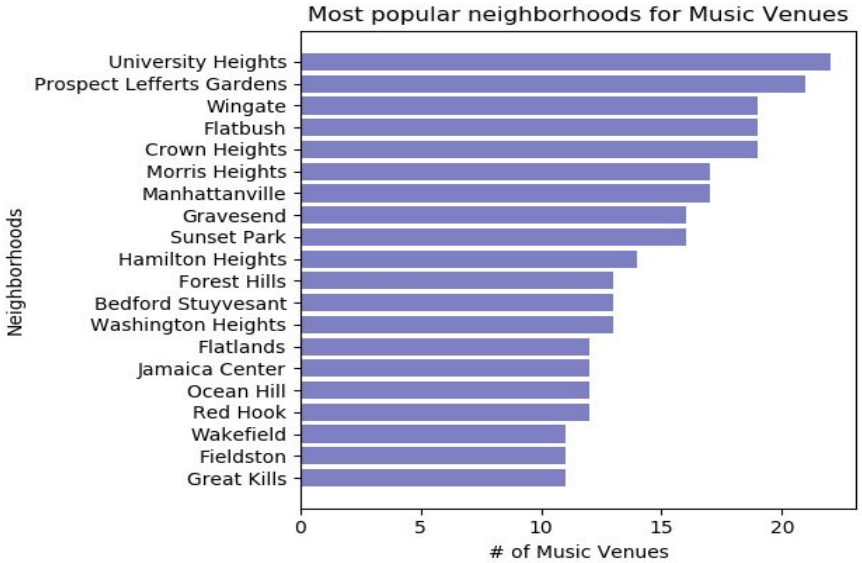
Analyzing Neighborhood Popularity

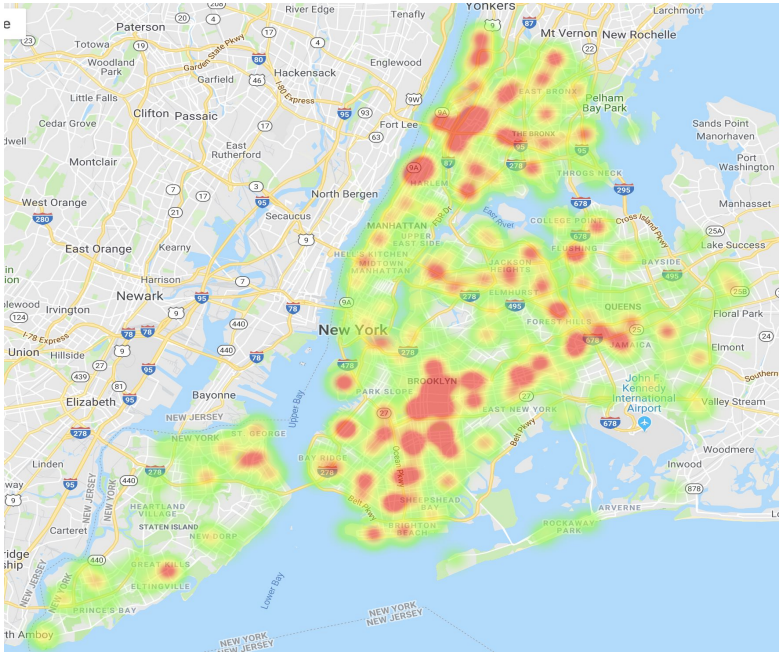
To initially determine popularity, we utilized the groupby function to aggregate venues by the neighborhoods they fall within. Creating a horizontal bar chart provides an easy way to determine top neighborhoods by concentration at a glance.

An interactive map with each venue plotted was also included for user interaction

```
# 1 - What are the most popular neighborhoods for music venues in New York City?
# rank neighborhoods in by count of venues bar chart

venue_count = df[['Neighborhood','Venue ID']].groupby('Neighborhood').count()
venue_count = venue_count.rename(columns={'Venue ID':'Venue Count'})
venue_count = venue_count.sort_values('Venue Count', ascending=True)
venue_count = venue_count.reset_index()
venue_count = venue_count[-21 : -1]

y_axis = np.arange(0, 20)
tick_locations = []
for y in y_axis:
    tick_locations.append(y)

plt.title("Most popular neighborhoods for Music Venues")
plt.xlabel("# of Music Venues")
plt.ylabel("Neighborhoods")

plt.barh(venue_count['Neighborhood'], venue_count['Venue Count'], facecolor="red"
plt.tight_layout()
plt.savefig("Output/Venue_Counts.png", bbox_inches="tight")
plt.show()
```

Insights and Analysis

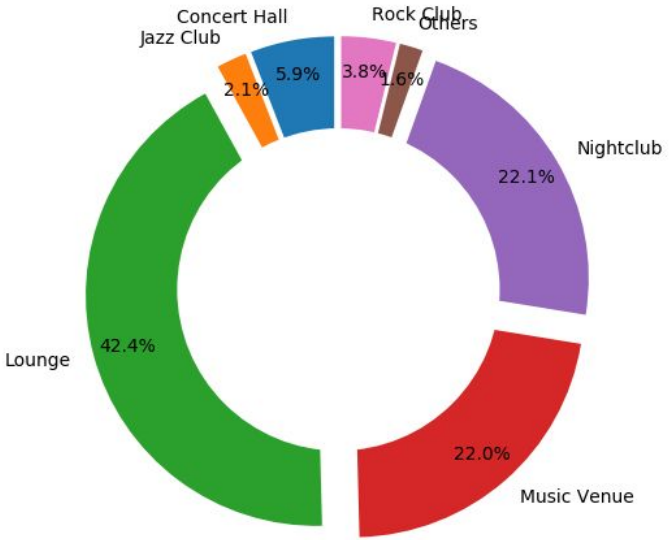




Insights and Analysis

Analyzing Venue Type Popularity

We then grouped venue categories together across neighborhoods to understand which were the most popular. **Lounges, Music Venues, and Nightclubs** made up the top 3.



Insights and Analysis

Analyzing Rating and Likes

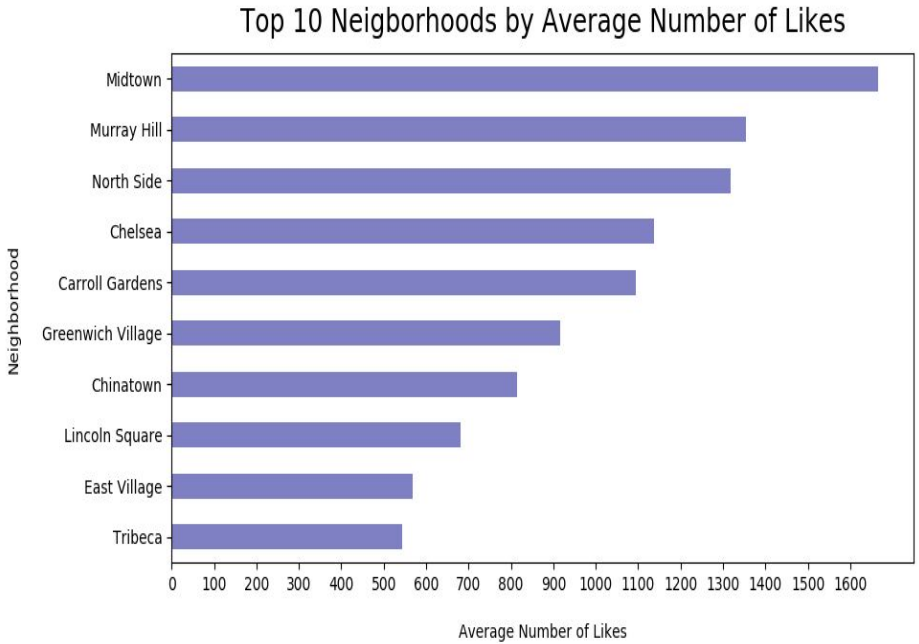
Only 215 venues out of 1226 had both likes and ratings. So in order to get the accurate data, we calculated average values of likes and rating by neighborhood.

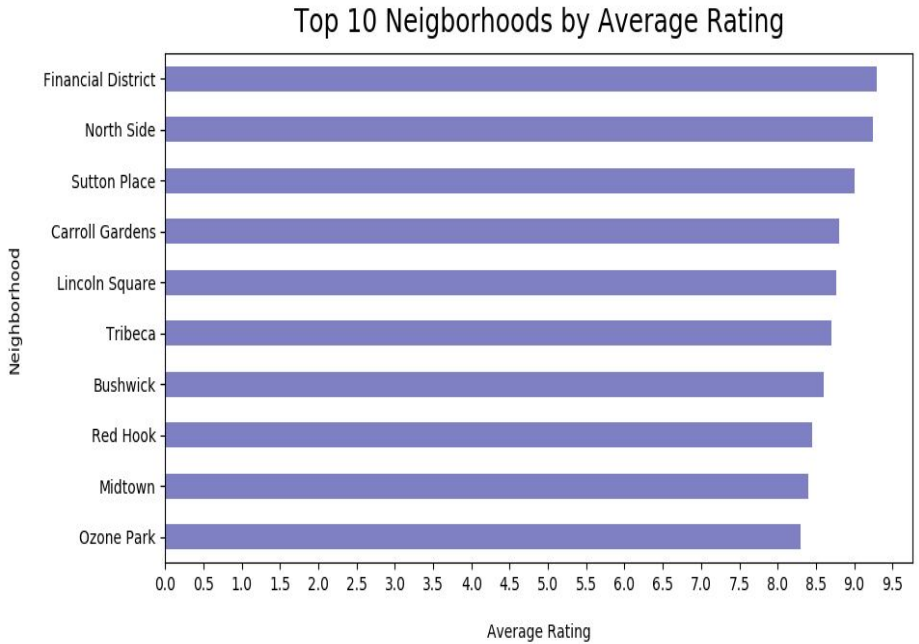
In [6]:
```
# 3 - Which neighborhoods have the most highly rated/most liked venues?

# Removing all the rows with none value in first dataframe
cleaned = df.dropna(how="any")

#Grouping by Neighborhood and getting needed columns for new dataframe
grouped_df = cleaned.groupby(["Neighborhood"])
likes = grouped_df["Likes"].mean()
rating = grouped_df["Rating"].mean()

#Creating dataframe
summary = pd.DataFrame({
    "Likes":likes,
    "Rating":rating,
})
summary.head()
```

Out[6]:

Neighborhood	Likes	Rating
Astoria Heights	10.0	6.15
Bay Ridge	9.0	7.20
Bayside	9.0	6.00
Bedford Stuyvesant	31.5	6.85
Bensonhurst	6.0	6.00

Insights and Analysis

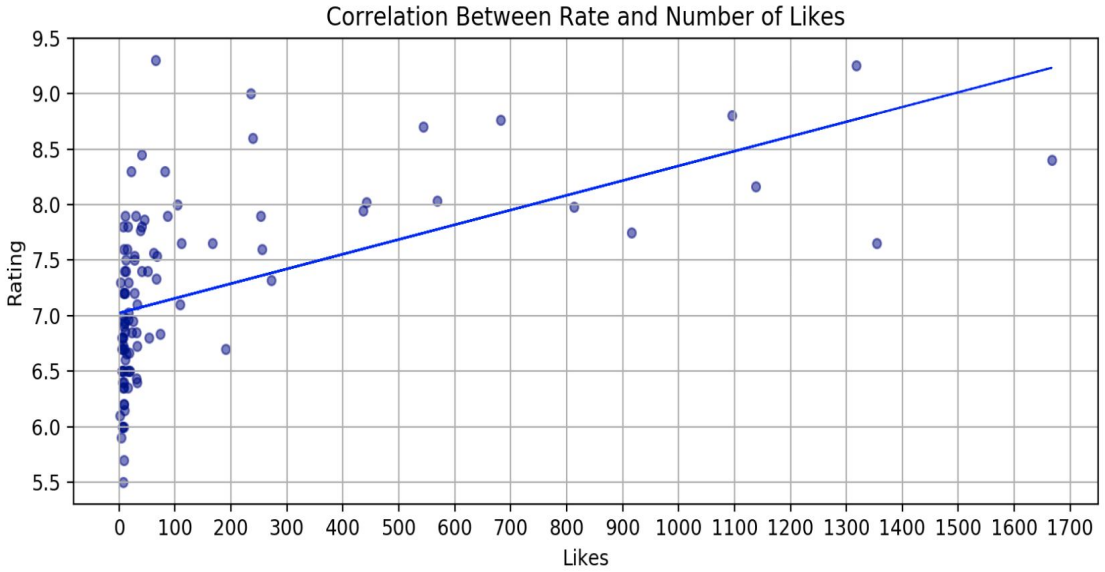




Insights and Analysis

Correlation Between Likes and Rating

We found a positive correlation between likes and rating once the rating goes beyond 7 that you can see on the scatter plot below.



Problems and Troubleshooting

API Premium Query Limitations

At first, we planned to analyze more Category IDs, but our initial search parameters would have returned over 10,000 venues, and we would only be able to append rating and like data for 500 per day, as this is considered to be a “premium query” by FourSquare. We narrowed our scope to less CategoryIDs, which reduced the total count to just over 1,200 venues - enough for each of us to query in one day.

Changes to FourSquare API Access

At first, we wanted to analyze check-in counts and visit counts to analyze loyalty and repeat customer business, but FourSquare deprecated these fields without updating their documentation. Instead we choose to analyse likes and ratings to determine popularity.

Areas of Future Study

Demographic Information

With more time, it would be interesting to know which venues are popular among which sex and whether there are any trends by neighborhoods in terms of sex.

Expanded Category IDs

With more time we would like to analyze more music-related Category IDs for all of New York City's five boroughs (over 10,000 venues) as there is a wide array of music-related venues that have evolved over time and interesting patterns might emerge in the data.

Any Questions?

Thank you