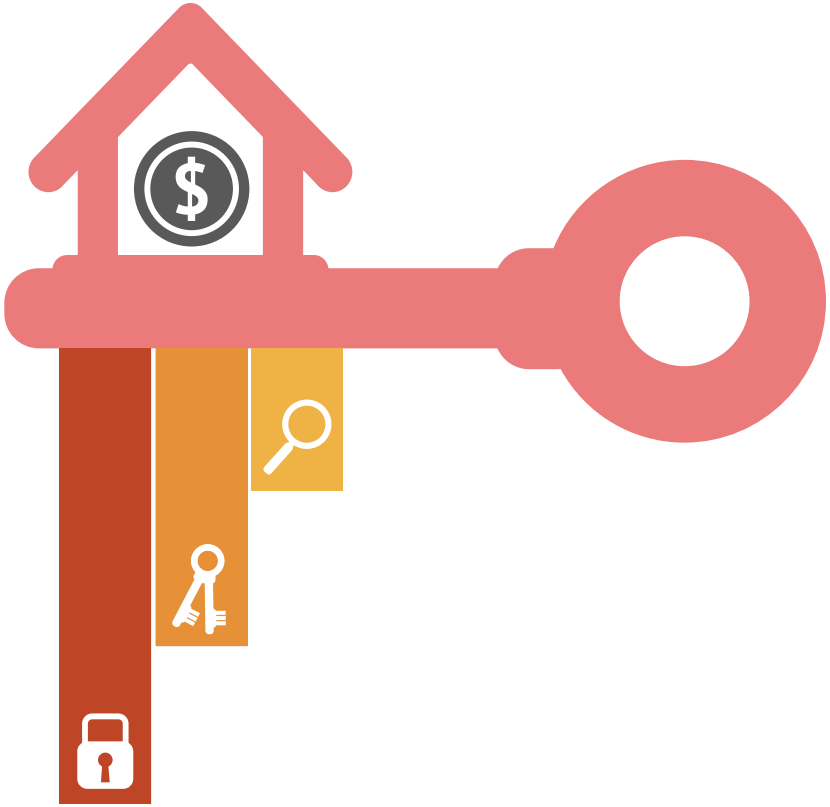
NYC Real Estate Market Analysis

By Yash Bhatia
& Khadija Bouzekri

Point of Interest and Purpose

As college students who have either lived in or strongly considered staying in New York City for summer internships or jobs

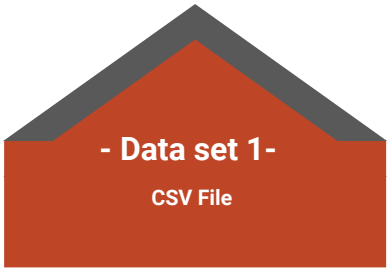


Findings Expectations

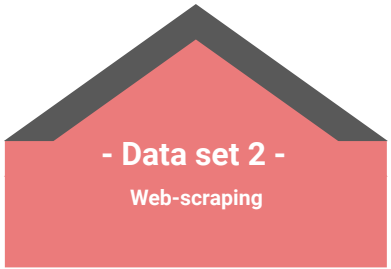
The final goal of the study is to predict future real estate prices in NYC by examining trends.

1. Analyzing the real estate volatility and price evolution of the 5 boroughs in NYC
2. Investigating the correlation between air pollution and the average real estate prices.
3. Exploring the interplay between racial demographics, median income and their impact on the average real estate prices in 2019 across the 5 boroughs in NYC.
4. Evaluating the influence of a neighborhood age on its real estate price in 2023.
5. Forecasting Housing Prices of the most volatile housing market by boroughs over 2024

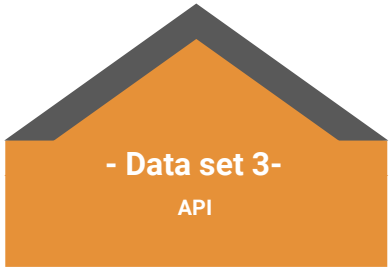
Data Collection Process



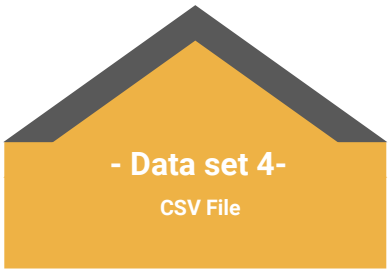
To collect data on the historical progression of average real estate prices in each neighborhood, we opted for the well-known platform "Zillow." This user-friendly website provides extensive, reliable real estate information.



We utilized the reputable "Wikipedia" page and employed web scraping to extract essential information about the average year of establishment for each neighborhood across NYC.



While searching for an API that could provide us with pertinent information about the demographics and median income in New York City neighborhoods, we faced challenges with the initial API. We promptly reached out to the owner, who then shared the updated API with us.



NYC Open Data was one of the recommended data sources by the Professor. It provided us valuable information about the air pollution level in different neighborhoods across NYC.

Data Cleansing : Data set 1

```python
import pandas as pd

def data_parser(zillow_data):
    file_path = '/Users/yashbhatia/Downloads/Neighborhood_zhvi_uc_sfrcondo_tier_0.33_0.67_sm_sa_month (2)
    zillow_data = pd.read_csv(file_path)
    # missing values for 'City' and 'Metro'
    zillow_data['City'].fillna('Unknown', inplace=True)
    zillow_data['Metro'].fillna('Unknown', inplace=True)

    # missing values in the monthly data replaced with mean of each column
    for column in zillow_data.columns[9:]: #9th column is the one after which we have monthly data
        zillow_data[column].fillna(zillow_data[column].mean(), inplace=True)

    # Converting to date time objects
    for column in zillow_data.columns[9:]:
        # Renaming the columns to a 'YYYY-MM-DD' format
        new_column_name = pd.to_datetime(column).strftime('%Y-%m-%d')
        zillow_data.rename(columns={column: new_column_name}, inplace=True)

    # Filtering the dataset New York
    nyc_data = zillow_data[zillow_data['City'] == 'New York']

    return nyc_data
    output_file_path = '/Users/yashbhatia/Desktop/CS2316/cleaned_nyc_datazillow.csv'
    cleaned_nyc_data.to_csv(output_file_path, index=False)

original_zillow_data = pd.read_csv(file_path)


# Call the data_parser function and get the cleaned NYC data
cleaned_nyc_data = data_parser(original_zillow_data)
cleaned_nyc_data
```

Filled every null value with “Unknown” in the columns ‘City’ and ‘Metro’. **Step 1**

Filled every null value in every other column with the mean of the respective column. **Step 2**

Convert column names to a standardized date format ‘YYYY-MM-DD’. **Step 3**

Since the study concerns NYC, we only kept rows that had ‘New York’ as a city. **Step 4**

Created a csv file containing the clean version of the original CSV. **Step 5**

Data Cleansing: Data set 2

NYCHA Property	Neighborhood/Subsection	No.# of Buildings	No.# of Stories	No.# of Apartments	Date of Completion
1010 East 178th Street	West Farms	1	21	218	March 31, 1971
1162-1176 Washington Avenue	Morrisania	1	6	64	December 31, 1975
1471 Watson Avenue	Soundview	1	6	96	December 31, 1970

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd
import re

def web_parser1():
    url = "https://en.wikipedia.org/wiki/List_of_New_York_City_Housing_Authority_properties"
    resp = requests.get(url)
    soup = BeautifulSoup(resp.text, "html.parser")
    final_list = []
    table = soup.find_all(class_="wikitable sortable")

    for each in table:
        rows = each.find("tbody").find_all("tr")[1:]
        for row in rows:
            cells = row.find_all("td")
            dates = re.findall(r'\b\d{4}\b', cells[5].text)
            neighborhood = cells[1].text.strip()
            if dates:
                dates = [int(year) for year in dates]
                mean_year = sum(dates) / len(dates)
                final_list.append((neighborhood, mean_year))

    df = pd.DataFrame(final_list, columns=['Neighborhood Name', 'Mean Year Established'])
    # Group by neighborhood and get mean of year established
    df2 = df.groupby(['Neighborhood Name'])['Mean Year Established'].mean().reset_index()
    df2['Mean Year Established'] = df['Mean Year Established'].astype(int)

    df2.to_csv('webscrapeddatawiki.csv', index=False)
    return df2
```

Used BeautifulSoup to parse through all 5 tables in the wikipedia page — **Step 1**

Looped through each row of each table and proceeded to loop through each cell of each row — **Step 2**

Used regex to extract only the establishment year of each neighborhood then added the dates to the list “dates” — **Step 3**

Looped through every neighborhood and added their names to the list “neighborhood” — **Step 4**

Combined both list and created a dictionary — **Step 5**

Created a new dataframe containing the dictionary data with two columns ‘Neighborhood Name’ and ‘Mean Year Established’ — **Step 6**

Data Cleansing Data set 3

```python
import requests
import pandas as pd
import json
def get_demographics_data(latitude, longitude):
    base_url = "https://maptile2.org/sp/api6.php"
    params = {
        'fLat': latitude,
        'fLon': longitude,
        'sGeo': 'county',
        'iPolygon': '1',}
    response = requests.get(base_url, params=params)
    if response.status_code == 200:
        return response.json()
    else:
        print(f"Failed to retrieve data: {response.status_code}")
        return None
# Define coordinates for the 5 boroughs of NYC
boroughs_coordinates = {
    'Manhattan': (40.776677, -73.97132),
    'Bronx': (40.83, -73.87),
    'Brooklyn': (40.65, -73.95),
    'Queens': (40.7282, -73.79),
    'Staten Island': (40.579021, -74.151535)}
# Initialize an empty list to store the data for each borough
boroughs_data = []
# Loop through each borough and retrieve demographics data
for borough, (latitude, longitude) in boroughs_coordinates.items():
    data = get_demographics_data(latitude, longitude)
    if data:
        demographics_data = {
            'Borough': borough,
            'Asian': data['asian'],
            'Black': data['black'],
            'Hispanic': data['hispanic'],
            'Indian': data['indian'],
            'Island': data['island'],
            'Multi': data['multi'],
            'White': data['white'],
            'Population': data['pop'],
            'Median Income': data['income'],       }
        boroughs_data.append(demographics_data)
# Create a DataFrame from the list of boroughs' demographics data
boroughs_df = pd.DataFrame(boroughs_data)
# Specify the output CSV file path
output_file_path = '/Users/yashbhatia/Desktop/CS2316/demographics_data.csv'
# Export the DataFrame to a CSV file
boroughs_df.to_csv(output_file_path, index=False)
# Print the DataFrame
boroughs_df
```

Setup and Library Imports:
Requests to Enable HTTP requests, **Pandas** for data handling and **JSON** for processing JSON data structures

Step 1

Define Function to Fetch Data: get_demographics_data that takes geographical coordinates as input and makes an APi call to mapit2le.org to retrieve demographic data

Step 2

Coordinate Specification for Multiple Locations:
Establishing a dictionary 'boroughs_coordinates' containing the latitude and longitude pairs of 5 boroughs in NYC

Step 3

Data Retrieval in a Loop: Using the function to fetch demographic data for each brough and put it into a dictionary

Step 4

Data Aggregation into a Data Frame: Collecting individual dictionaries of demographic data into a list and convert to pandas dataframe

Step 5

	Borough	Asian	Black	Hispanic	Indian	Island	Multi	White	Population	Median Income
0	Manhattan	13.1028	13.5376	23.7651	0.588402	0.105947	10.3551	46.8227	1694251	89812
1	Bronx	4.72338	33.0776	54.7626	1.4626	0.111364	13.0494	8.88165	1472654	41895
2	Brooklyn	13.6575	28.2174	18.8747	0.752831	0.0543479	8.67297	35.3948	2736074	63973
3	Queens	27.4638	16.7567	27.7643	1.26849	0.0714207	10.1171	22.8379	2405464	72028
4	Staten Island	11.9575	10.4971	19.5584	0.725975	0.0472015	7.83061	56.0732	495747	85381

Data Cleansing Data Set 4

	Unique ID	Indicator ID	Name	Measure	Measure Info	Geo Type Name	Geo Join ID	Geo Place Name	Time Period	Start_Date	Data Value	Message
0	172653	375	Nitrogen dioxide (NO2)	Mean	ppb	UHF34	203	Bedford Stuyvesant - Crown Heights	Annual Average 2011	12/01/2010	25.30	NaN

```
def extra_source1():
    air_quality_missing_values = air_quality_data.isnull().sum()
    air_quality_data_types = air_quality_data.dtypes

    ## converting to datetime format to handle inconsistencies
    air_quality_data['Start_Date'] = pd.to_datetime(air_quality_data['Start_Date'])

    geo_place_names = air_quality_data['Geo Place Name'].unique()

    # Getting year from 'Start_Date' column
    air_quality_data['Year'] = air_quality_data['Start_Date'].dt.year

    # Grouping by Geo Place Name and Year and then calculating the mean 'Data Value' (this is the pollution level) for each group
    air_quality_annual_mean = air_quality_data.groupby(['Geo Place Name', 'Year'])['Data Value'].mean().reset_index()

# Zillow data has monthly values, we are creating an annual average for each neighborhood in NYC
# because we want to look at the effect of air quality on home prices from the zillow dataset
# Extracting year from the datetime columns
    zillow_years = [column for column in cleaned_nyc_data.columns if column.endswith('-31')]
    zillow_annual_mean = cleaned_nyc_data.melt(id_vars=['RegionName'], value_vars=zillow_years)
    zillow_annual_mean['Year'] = zillow_annual_mean['variable'].str[:4].astype(int)
    zillow_annual_mean.drop('variable', axis=1, inplace=True)
    zillow_annual_mean = zillow_annual_mean.groupby(['RegionName', 'Year'])['value'].mean().reset_index()

# Joining on Year and neibhorhoood age
    air_quality_zillow_joined = pd.merge(
        zillow_annual_mean,
        air_quality_annual_mean,
        left_on=['RegionName', 'Year'],
        right_on=['Geo Place Name', 'Year'],
        how='inner')

    df2 = air_quality_zillow_joined
    output_file_path = '/Users/yashbhatia/Desktop/CS2316/airqualitymerged.csv'
    df2.to_csv(output_file_path, index=False)
    return df2

############ Function Call ############
extra_source1()
```

	RegionName	Year	value	Geo Place Name	Data Value
0	Borough Park	2005	538496.879143	Borough Park	15.073333

Downloaded the Data set from NYC Open Data **Step 1**

Used .to_datetime to handle inconsistencies in the date time column level for each group **Step 2**

Grouped the rows by Geo Place and the Year and finding the average pollution **Step 3**

Calculated the mean housing price for a year for a ‘RegionName’ **Step 4**

Combined both data frames using the “merge” function using the columns “RegionName” and “Geo Place Name” **Step 5**

We obtained our combined dataset **Step 6**

Insight 1: Which Borough's housing market has been the most volatile?

1. Which borough in New York City has had the highest percentage change in housing price?
2. Which borough has had the highest volatility in its housing prices?

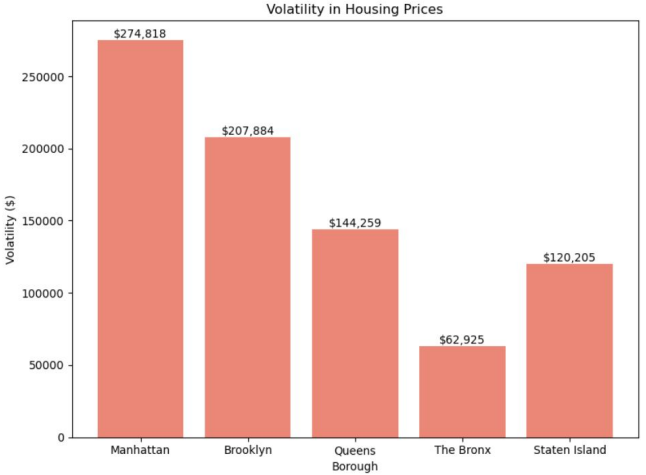
	PercentChange	Volatility
Manhattan	174.889486	274818.226690
Brooklyn	361.490200	207884.248283
Queens	196.948845	144259.452360
The Bronx	100.879811	62925.155454
Staten Island	238.064073	120205.075874

Manhattan is the most volatile, with 274,818$ in volatility.

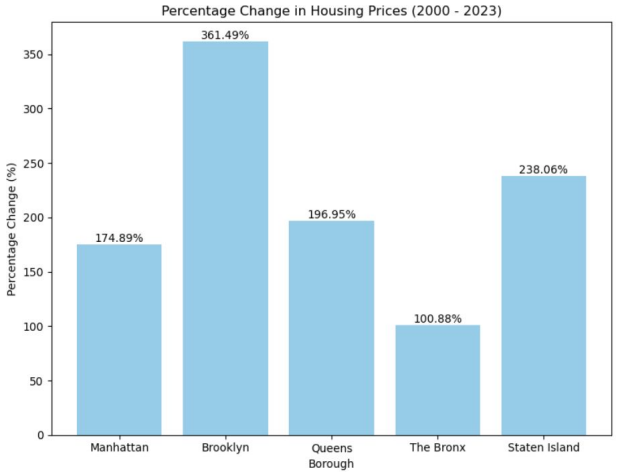
The Bronx is the least volatile, with 62,925$ in volatility.

Brooklyn shows the biggest percentage change in its housing price (361%) in the last 23 years

The Bronx has shown the least percent change in its housing prices (100.88%)

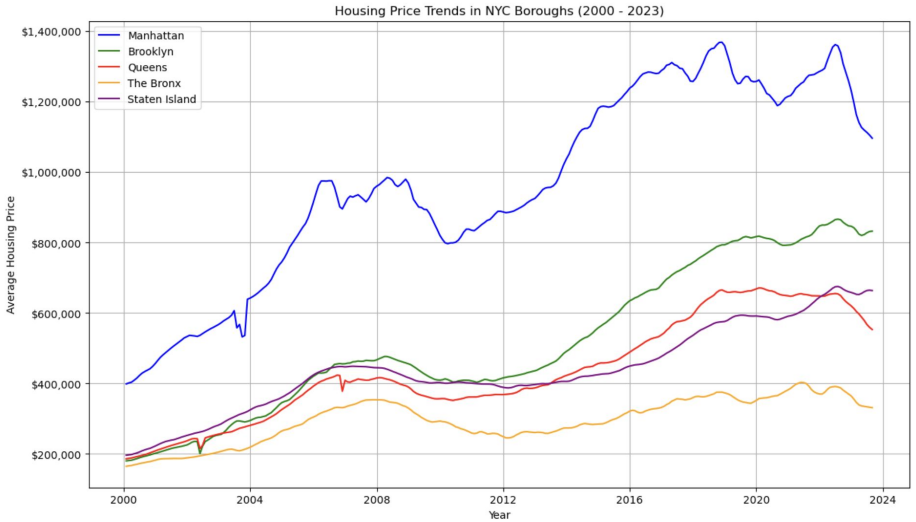



```python
import pandas as pd
def calculate_borough_stats(data, borough_mapping, housing_price_columns):
    def aggregate_by_borough(data, borough_mapping):
        borough_data = {}
        for borough, neighborhoods in borough_mapping.items():
            borough_neighborhoods_data = data[data['RegionName'].isin(neighborhoods)]
            borough_data[borough] = borough_neighborhoods_data[housing_price_columns].mean()
        return borough_data
    borough_housing_data = aggregate_by_borough(data, borough_mapping)
    borough_housing_df = pd.DataFrame(borough_housing_data)
    borough_housing_df.index = pd.to_datetime(borough_housing_df.index)
    borough_stats = {}
    for borough in borough_housing_df.columns:
        borough_data = borough_housing_df[borough]
        percent_change = ((borough_data.iloc[-1] - borough_data.iloc[0]) / borough_data.iloc[0]) * 100
        std_deviation = borough_data.std()
        borough_stats[borough] = {'PercentChange': percent_change, 'Volatility': std_deviation}
    borough_stats_df = pd.DataFrame(borough_stats).T
    return borough_stats_df
borough_stats_df = calculate_borough_stats(nyc_housing_data, borough_mapping, housing_price_columns)
borough_stats_df
```



Visualization 2: Price Evolution of the 5 borough of NYC

```python
def aggregate_by_borough(data, borough_mapping):
    borough_data = {}
    for borough, neighborhoods in borough_mapping.items():
        borough_neighborhoods_data = data[data['RegionName'].isin(neighborhoods)]
        borough_data[borough] = borough_neighborhoods_data[housing_price_columns].mean()
    return borough_data

borough_housing_data = aggregate_by_borough(nyc_housing_data, borough_mapping)

borough_housing_df = pd.DataFrame(borough_housing_data)
borough_housing_df.index = pd.to_datetime(borough_housing_df.index)  # Convert index to datetime
def plot_housing_trends(borough_housing_df):
    formatter = FuncFormatter(dollar_formatter)
    plt.figure(figsize=(14, 8))
    borough_colors = ['blue', 'green', 'red', 'orange', 'purple']

    for (borough, color) in zip(borough_housing_df.columns, borough_colors):
        plt.plot(borough_housing_df.index, borough_housing_df[borough], label=borough, color=color)

    plt.gca().yaxis.set_major_formatter(formatter)
    plt.title('Housing Price Trends in NYC Boroughs (2000 - 2023)')
    plt.xlabel('Year')
    plt.ylabel('Average Housing Price')
    plt.legend()
    plt.grid(True)
    plt.show()
plot_housing_trends(borough_housing_df)
```



- We can notice several periods when the housing market for certain boroughs stayed resilient and how the housing prices for Staten Island have surpassed queens in recent years.
- Manhattan real estate price have been severely decreasing during the last past years.
- The Bronx has seen almost no increase in its real estate prince since the last 20 years.

Insight 2: Demographics Data Vs Housing Prices

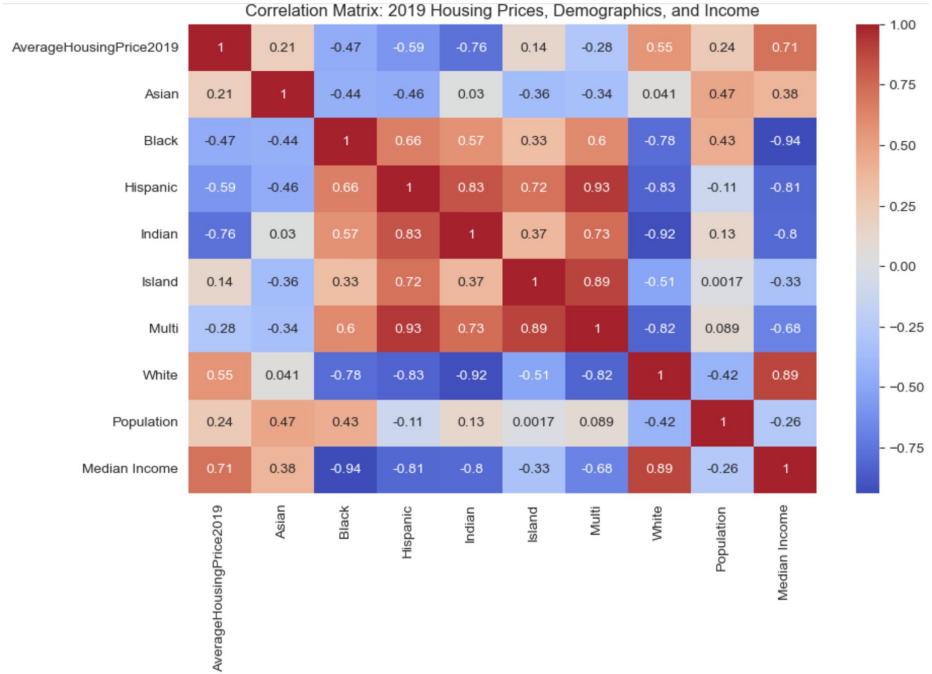
How does demographics impact housing prices of the 5 boroughs of New York City?

- There is a strong positive correlation between housing prices and median income, indicating that higher median incomes are typically associated with higher housing prices.
- The correlations between housing prices and the percentages of different racial groups show varied strengths and directions.
  - Positive correlation with the percentage of the White population and Asian population
  - Negative correlations with other groups such as Hispanic, Indian, Island, Multi and Black with Indian population having the strongest negative correlation with Average Housing Price in 2019.
- These correlations provide insights into how demographic makeup might relate to housing prices, though it's crucial to remember that correlation does not imply causation.

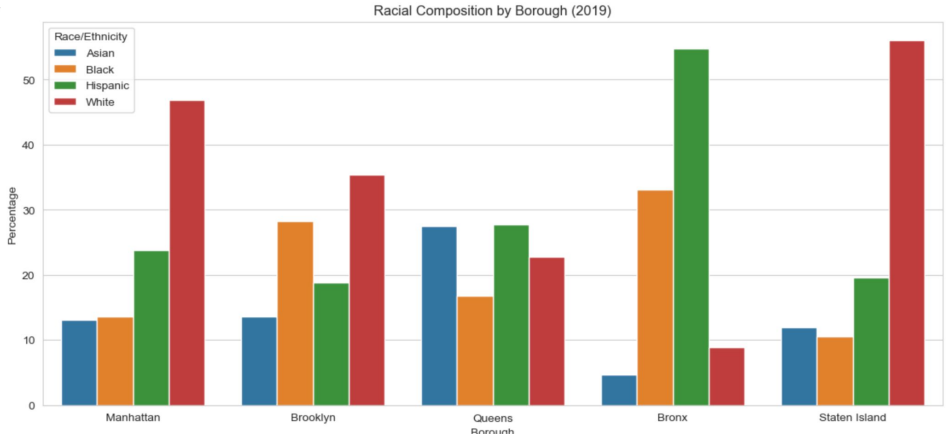
```
borough_mapping = {
    'Manhattan': ['Upper West Side', 'Upper East Side', 'Harlem', 'Washington Heights', 'Chelsea', 'Greenwich Village', 'East Harlem'],
    'Brooklyn': ['East New York', 'Bedford-Stuyvesant', 'Williamsburg', 'Crown Heights', 'Borough Park', 'Bushwick', 'Sheepshead Bay', 'Flatbush'],
    'Queens': ['Flushing', 'Astoria', 'Elmhurst', 'Jackson Heights', 'Forest Hills'],
    'Bronx': ['Riverdale', 'Fordham', 'Concourse', 'Kingsbridge'],
    'Staten Island': ['St. George', 'Tottenville', 'Great Kills']}
housing_price_columns_2019 = [col for col in cleaned_nyc_data.columns if "2019" in col]
borough_housing_prices_2019 = {}
for borough, neighborhoods in borough_mapping.items():
    borough_data = cleaned_nyc_data[cleaned_nyc_data['RegionName'].isin(neighborhoods)]
    borough_housing_prices_2019[borough] = borough_data[housing_price_columns_2019].mean().mean()

borough_housing_prices_df = pd.DataFrame.from_dict(borough_housing_prices_2019, orient='index', columns=['AverageHousingPrice2019'])
borough_housing_prices_df.reset_index(inplace=True)
borough_housing_prices_df.rename(columns={'index': 'Borough'}, inplace=True)
merged_data = pd.merge(borough_housing_prices_df, boroughs_df, on="Borough")
merged_data
melted_data = merged_data.melt(id_vars=['Borough', 'AverageHousingPrice2019'], value_vars=['Asian', 'Black', 'Hispanic', 'White'])
```

Real Estate Infographics

```
import seaborn as sns
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix_2019, annot=True, cmap="coolwarm")
plt.title("Correlation Matrix: 2019 Housing Prices, Demographics, and Income")
plt.show()
```




```
melted_data['value'] = pd.to_numeric(melted_data['value'], errors='coerce')
melted_data = melted_data.dropna()
plt.figure(figsize=(14, 6))
sns.barplot(x='Borough', y='value', hue='variable', data=melted_data)
plt.title("Racial Composition by Borough (2019)")
plt.xlabel("Borough")
plt.ylabel("Percentage")
plt.legend(title="Race/Ethnicity")
plt.show()
```

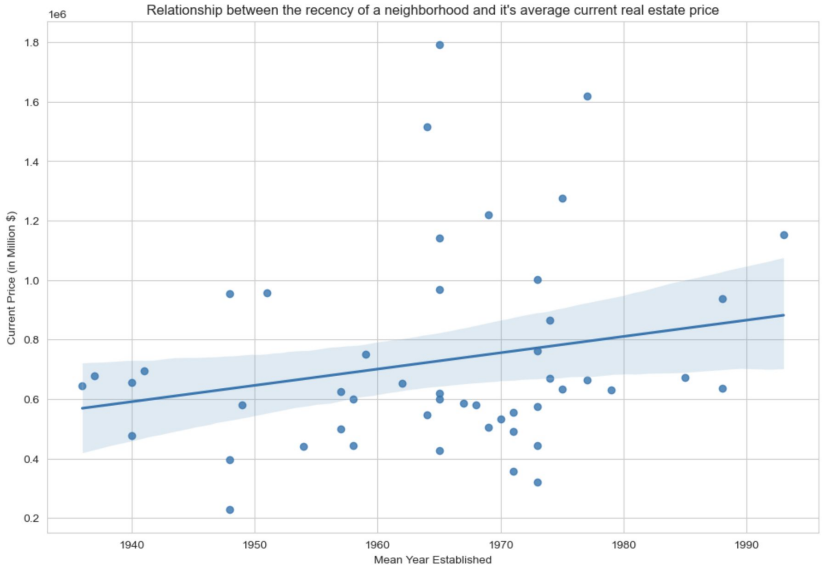


Insight 3: Age of Neighborhood vs Current Price of its Real Estate

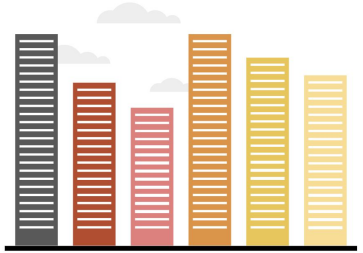
How does the establishment year of a neighborhood in New York City impact its current real estate market prices?

```python
import pandas as pd

year_info_df = df2
pricing_info_df = cleaned_nyc_data

common_neighborhoods = set(year_info_df['Neighborhood Name']).intersection(pricing_info_df['RegionName'])

new_data = {'Neighborhood Name': [],
            'Mean Year Established': [],
            'Current Price': []}

for neighborhood in common_neighborhoods:
    mean_year_established = year_info_df.loc[year_info_df['Neighborhood Name'] == neighborhood, 'Mean Year Established'].values[0]
    pricing = pricing_info_df.loc[pricing_info_df['RegionName'] == neighborhood].iloc[:, -1].values[0]

    new_data['Neighborhood Name'].append(neighborhood)
    new_data['Mean Year Established'].append(mean_year_established)
    new_data['Current Price'].append(pricing)
new_df = pd.DataFrame(new_data)
new_df = new_df.sort_values(by='Mean Year Established').reset_index()[['Neighborhood Name','Mean Year Established','Current Price']]

new_df.to_csv('insight1.csv', index=False)

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
def visualization1():
    df = pd.read_csv('insight1.csv')
    correlation_coefficient1 = df['Mean Year Established'].corr(df['Current Price'])
    print(f'Correlation Coefficient: {correlation_coefficient1}')
    plt.figure(figsize=(12, 8))
    sns.regplot(x='Mean Year Established', y='Current Price', data=df)
    plt.title("Relationship between the recency of a neighborhood and it's average current real estate price")
    plt.xlabel('Mean Year Established')
    plt.ylabel('Current Price (in Million $)')
    plt.grid(True)
    plt.show()
visualization1()
```

```
Correlation Coefficient: 0.2237541218739322
```



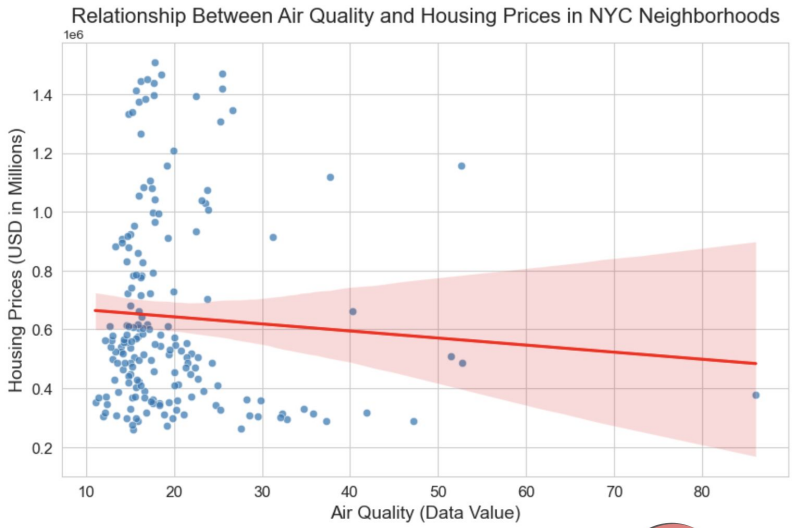
Positive correlation of +0.20 between the mean year of establishment of neighborhoods and their current real estate prices. This indicates that, on average, newer neighborhoods tend to have higher real estate values compared to older ones.



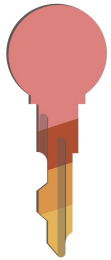
Insight 4 Explanation: Exploring the Relationship between Air Quality and Housing Prices in New York City Neighborhoods: A Correlational Analysis

```python
def analyze_air_quality_housing_relationship(file_path):

    import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns


    merged_data = pd.read_csv(file_path)
    correlation = merged_data['Data Value'].corr(merged_data['value']) # Calculate the Pearson correlation coefficient
    print(f"Pearson Correlation Coefficient: {correlation}")

    sns.set_style("whitegrid")
    plt.figure(figsize=(10, 6))
    sns.scatterplot(data=merged_data, x='Data Value', y='value', alpha=0.7)
    sns.regplot(data=merged_data, x='Data Value', y='value', scatter=False, color='red')
    plt.title('Relationship Between Air Quality and Housing Prices in NYC Neighborhoods', fontsize=16)
    plt.xlabel('Air Quality (Data Value)', fontsize=14)
    plt.ylabel('Housing Prices (USD in Millions)', fontsize=14)
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)

    plt.show()
analyze_air_quality_housing_relationship('/Users/yashbhatia/Desktop/CS2316/airqualitymerged.csv')
```

```
Pearson Correlation Coefficient: -0.06410203054229097
```



- Weak Negative Correlation Identified (correlation coefficient of approximately -0.064) between air quality and housing prices
- This suggests that while there is a slight tendency for housing prices to decrease as air quality worsens, the effect is minimal.
- The findings highlight the complexity of the housing market, where numerous factors interplay to determine prices. Air quality, while an important environmental concern, appears to have a limited direct linear impact on housing values in the context of this analysis.



Insight 5: Forecasting 2024 Housing Prices in NYC's Most Volatile Broughs: Manhattan, Queens & Brooklyn

- ARIMA: AutoRegressive (AR) Integrated (I) Moving Average (MA) Model, a time series forecasting model, to provide investors with actionable insights. Very well suited for Seasonal time series data
- **Objective:** Help people who are planning on investing in New York City to decide which borough to invest in

Breakdown of the Model: Step 1) Stationarity Check: ARIMA model required that time series data is stationary.

```
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
manhattan_data = borough_housing_df['Manhattan']
brooklyn_data = borough_housing_df['Brooklyn']
Queens_data = borough_housing_df['Queens']

# Function to check stationarity
def check_stationarity(timeseries):
    result = adfuller(timeseries, autolag='AIC')
    p_value = result[1]
    return p_value

# Checking if the Manhattan data is stationary
p_value1 = check_stationarity(manhattan_data)
p_value2 = check_stationarity(brooklyn_data)
p_value3 = check_stationarity(Queens_data)
p_value4 = check_stationarity(Bronx_data)
p_value5 = check_stationarity(Staten_data)

# Displaying the p-value
print(p_value1)
print(p_value2)
print(p_value3)
```

P- values using Dicker- Fuller test

```
0.2724477451726762
0.7846051604227302
0.4199690225161504
```

Not stationary so we had to perform differencing on them

ARIMA Forecasting Model Results:

Predictions for Manhattan

```
# Differencing the data to achieve stationarity
manhattan_diff = manhattan_data.diff().dropna()
p_value_diff = check_stationarity(manhattan_diff) # Rechecking stationarity after differenci
p_value_diff

model = ARIMA(manhattan_diff, order=(1, 1, 1))
model_fit = model.fit()

# Forecasting the next 12 months
forecast = model_fit.forecast(steps=12)
forecast_dates = pd.date_range(start=manhattan_diff.index[-1], periods=13, freq='M')[1:]
forecast_values = forecast

# Since the model was fitted to differenced data, we added back the last known value to get
last_known_value = manhattan_data.iloc[-1]
forecast_values = forecast_values.cumsum() + last_known_value
formatted_forecast = forecast_values.apply(lambda x: "${:,.0f}".format(x))
forecast_df = pd.DataFrame({'Date': forecast_dates, 'Forecasted Price': formatted_forecast})
forecast_df
```

Manhattan

	Date	Forecasted Price
0	2023-09-30	$1,089,471
1	2023-10-31	$1,083,307
2	2023-11-30	$1,077,630
3	2023-12-31	$1,072,205
4	2024-01-31	$1,066,911
5	2024-02-29	$1,061,685
6	2024-03-31	$1,056,493
7	2024-04-30	$1,051,320
8	2024-05-31	$1,046,156
9	2024-06-30	$1,040,998
10	2024-07-31	$1,035,841
11	2024-08-31	$1,030,687

Queens

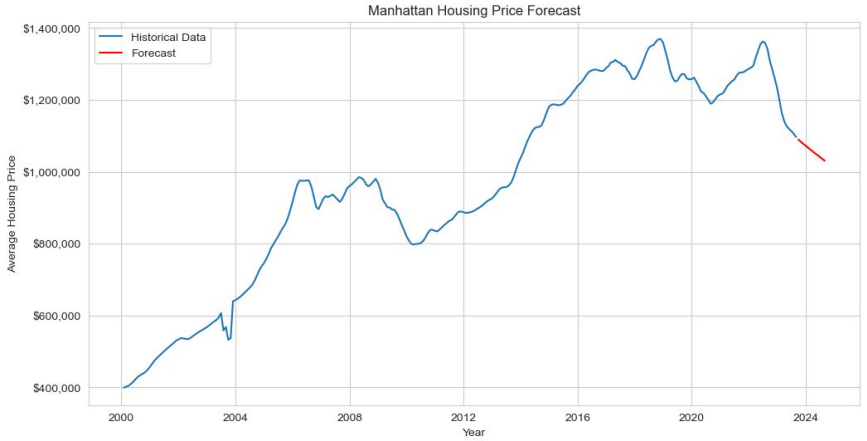
	Date	Forecasted Price
0	2023-09-30	$545,766
1	2023-10-31	$538,395
2	2023-11-30	$531,000
3	2023-12-31	$523,609
4	2024-01-31	$516,217
5	2024-02-29	$508,824
6	2024-03-31	$501,432
7	2024-04-30	$494,040
8	2024-05-31	$486,648
9	2024-06-30	$479,256
10	2024-07-31	$471,864
11	2024-08-31	$464,472

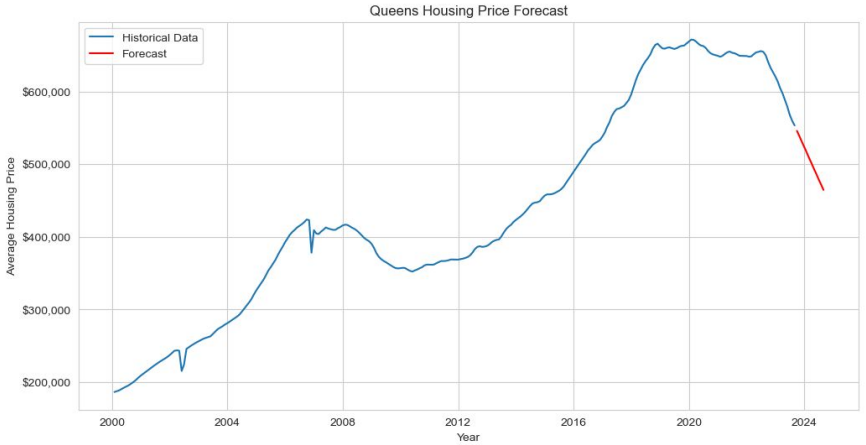
Brooklyn

	Date	Forecasted Price
0	2023-09-30	$832,553
1	2023-10-31	$832,256
2	2023-11-30	$831,909
3	2023-12-31	$831,552
4	2024-01-31	$831,193
5	2024-02-29	$830,833
6	2024-03-31	$830,473
7	2024-04-30	$830,113
8	2024-05-31	$829,753
9	2024-06-30	$829,393
10	2024-07-31	$829,033
11	2024-08-31	$828,673

- The forecasts suggest a gradual decrease in the average housing prices in Manhattan, Queens and Brooklyn over 2024.
- Brooklyn will be the least volatile housing market with the rate of decrease being the slowest
- By mid-2024, the prices could stabilize, with very slight decreases.

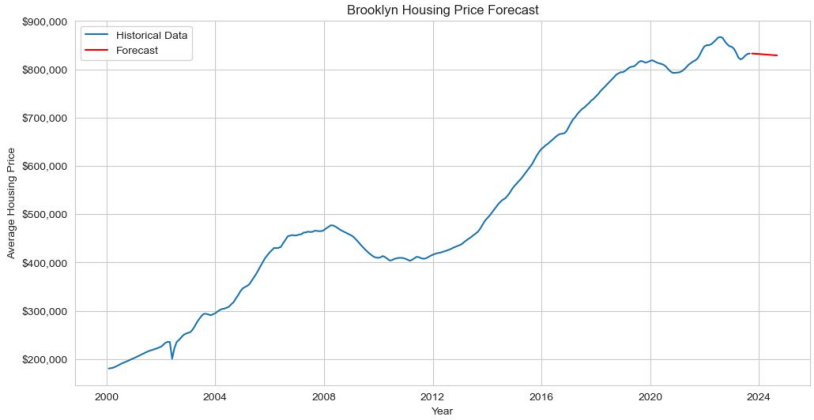
Insight 5: Visualizations



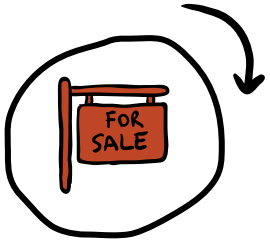


Brooklyn ⟹ **Least volatile**

⟹ **Upward Trajectory**

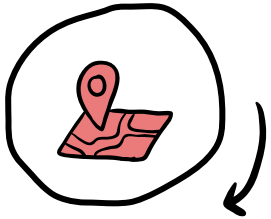


Conclusion



Volatility & price evolution

Manhattan is the most volatile, while the Bronx has been the most stable.

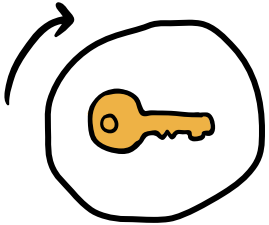


Recency of a neighborhood

Newer neighborhoods tend to have higher real estate values compared to older ones.

Air pollution index

Slight tendency for housing prices to decrease as air quality worsens, the effect is minimal.



Demographics & median income

Higher median incomes & higher white population are associated with higher housing prices.

Arima Forecast Model Result

The 3 most volatile boroughs: Manhattan, Queens and Brooklyn show a downward trajectory in prices while Brooklyn falls the least.