UCDPA – John Lenehan

# GitHub URL

https://github.com/jlenehan/UCDPA\_JohnLenehan

# Abstract

(Short overview of the entire project and features)

# Introduction

For this project, the use case is defined as a non-profit in New York city providing support for eviction cases across the municipality. Such a non-profit would naturally need to gather data to determine where best to concentrate their efforts in the city; i.e. where the most evictions are per capita and what drives these trends. The project here is therefore set up as an analysis that such a non-profit might perform.

# Dataset

The dataset used for this analysis is a merging of data from three sources; a dataset of evictions in New York city from 2017 to 2022, a dataset of household income in New York city from 2013 to 2020, and census data of county population across New York state.

# Implementation Process

Step 0: Import Libraries

First thing, the necessary libraries for analysis must be imported; these are laid out below:

* Import os
* Import pandas as pd
* From datetime import date
* Import matplotlib.pyplot as plt
* Import seaborn as sns

Step 1: Import Data

Step 1 is to import the data; the eviction and income data are imported from the web using the pd.read\_json() method. The data for each can be found at the below addresses:

|  |  |  |
| --- | --- | --- |
| Data name | Data Source (URL) | JSON link |
| Eviction Data | https://data.cityofnewyork.us/City-Government/Evictions/6z8x-wfk4 | https://data.cityofnewyork.us/resource/6z8x-wfk4.json?$limit=100000 |
| Income Data | https://datausa.io/profile/geo/new-york-ny#economy | Data downloaded and imported as a csv file |
| Population Data | https://data.ny.gov/Government-Finance/Annual-Population-Estimates-for-New-York-State-and/krt9-ym2k | https://data.ny.gov/resource/krt9-ym2k.json?$limit=100000 |

Note that the query “?$limit=100000” is added to the json strings of both datasets, to increase the json row limit from 1,000 to 100,000.

Text

Description automatically generated with medium confidence

For the income data, the data is downloaded and stored as a csv file (included in the zip file for this assignment) and the data is read into the jupyter notebook using the .read\_csv() method. This method is used as a review of the data showed it hadn’t been updated since 2020, therefore the data is already static and wouldn’t benefit from pulling the data live. The directory of the notebook is set to the assignment folder using the os.chdir() function:



From there the file can be read into the read\_csv() function locally:

Text

Description automatically generated

Step 2: Describe Data

For each dataset, the methods .columns, .info, .describe, and .shape are used to give an idea of what each dataset looks like.

Text

Description automatically generated

Additionally the unique values of each dataset column is printed using a for loop, along with a count of unique values for each column:

Text

Description automatically generated

1. Eviction Data

The eviction data has a number of columns which could be useful for analysis; the marshal first and last names are given, along with latitude and longitude data of each eviction, and the council district in which the eviction was carried out. The data shown is for the past 6 years (2017 to 2022), across all 5 boroughs of New York city. For this analysis the scope is limited to the number of evictions by borough and year – for these columns the .info() function shows no null values, so it won’t be necessary to drop null values from this dataset. The year data is captured in the eviction\_date column but not in a form useful for analysis, so this needs to be extracted.

1. Income Data

Income data is given here for 8 years, from 2013 to 2020, and is also for all 5 boroughs of New York city. Similar to the eviction data, the income data has no null values at all across any of its columns. There is a column for race, but this only has 1 unique value – therefore the column should be filtered out as it has no use. Geography data is also given, but in longform instead of by county – this needs to be separated.

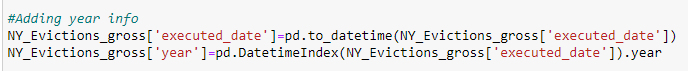
1. Population Data

The census population data contains no null values, and contains population data for all New York counties from 1970 to 2021. Some filtering is necessary to get only the most recent census data for the 5 counties (boroughs) under the New York municipality, and an aggregating function is necessary to get the income data in a useful format, but otherwise there isn’t as much cleaning required as the other datasets.

Step 3: Clean and Manipulate Data

1. Eviction Data

From looking at the eviction data, the executed\_date column is stored as an object datatype which makes datetime functions difficult – as such this needs to be converted to a datetime datatype. This is done using the pandas to\_datetime() method. From there the .year method is used on the DatetimeIndex() function to extract the year of each eviction.



Following this a count of evictions via the eviction\_zip column is executed, producing a pivot table grouped by year and borough using groupby(). For further analysis this is then ungrouped using reset\_index(), and the column is renamed from eviction\_zip to sum\_evictions. This gives the final evictions dataset of evictions by year and borough.

Text

Description automatically generated with medium confidence

1. Income Data

The original income data has a geography column giving the census tract, county and state the data was collected from; this is separated out using the str.split() on the ‘,’ delimiter into 3 columns.

From there the new data is merged with the original data using .merge() and an inner join, with the indices used as the merge key (left\_index = True, right\_index = True).

A screenshot of a computer

Description automatically generated

Columns are renamed using .rename, and the values in the Borough column are replaced to correspond with the terms for those boroughs given in the evictions dataset (i.e. Bronx, Brooklyn, Manhatten, Queens, Staten Island).

Text

Description automatically generated with medium confidence

Finally, similar to the process used with the eviction data, the mean household income and household income margin of error (MOE) by year and borough is displayed using groupby(). The values of mean household income and household income MOE are rounded to 2 decimal places using .round(), and the index on the table is reset using .reset\_index():

A picture containing text

Description automatically generated

1. Population Data

With the population data, the dataset gives 3 different population figures under the program\_type column, 2 of which are population estimates:

Text

Description automatically generated with low confidence

For this analysis and for convenience only the “Census Base Population” entries from 2020 are used; to filter for this the following line of code is run:



The original dataset showed census figures for all counties in New York state – however as this analysis is only interested in the counties for New York city, this is filtered down to those 5 counties (or boroughs) using .loc() and .isin(). Subsequently the names of the boroughs are replaced to match the values in the eviction dataset using .replace(), and the name of the column is changed from ‘geography’ to ‘borough’ using .rename():

Text

Description automatically generated with medium confidence

Finally the dataset is filtered down to borough, year and population data, and the borough is set as the dataframe index:

A picture containing text

Description automatically generated

For subsequent analysis, the 2020 census data for these boroughs will be assumed as constant across all years under review.

Step 4: Merge Data

With all individual dataframes now cleaned and aggregated to show the desired data, these data must now be merged to produce the final amalgamated dataframe for analysis. Both joins used for this process are left joins, as the primary goal for this analysis is evictions; any data from the income or population datasets that doesn’t correspond with the can be dropped using the dropna() function.

Firstly the income dataframe is merged to the population data using a left join, on the borough column; this has the effect of attaching the 2020 population data to all rows in the income dataframe, irrespective of year:

A screenshot of a computer

Description automatically generated with low confidence

Next the evictions dataframe is merged to the previously merged income-population dataframe, also via a left join. However for this join it’s necessary to join on both the year and borough columns for each dataframe, to ensure income data matches the year in the evictions table as well as the borough:

Text

Description automatically generated with medium confidence

This produces a dataframe of evictions across the 5 boroughs from 2017 to 2020, with income data accurate to each year and population data by borough from the 2020 census, assumed to be constant for this analysis. Some further cleaning of the new “NY\_Merged” dataframe is required; rows with null values are dropped using dropna(), redundant columns from the joins are filtered out, columns are renamed, and the year column is changed from a number entry to a string (a string entry is preferrable for the plotting step):

Chart, scatter chart

Description automatically generated with medium confidence

Subsequently the evictions data can be normalized to the population data by dividing the ‘Sum Evictions’ column by the ‘Population’ column. Furthermore the upper and lower margins of the household income can be calculated by either adding or subtracting the ‘Household Income MOE’ column from the ‘Household Income’ column.

Text

Description automatically generated

Lastly the correlation function .corr() is run to show how each column correlates against the others:

Table

Description automatically generated

Step 5: Plot Data

Seaborn lineplots are first deployed showing the evictions and evictions per 1,000, with hue set to borough to show the distinction between boroughs. This is done using the .lineplot() function:

Text

Description automatically generated

Next a scatter plot is produced showing the evictions per 1,000 against household income, grouped by borough:

Text

Description automatically generated

A custom function is defined to produce a scatter plot with a trendline using lnplot(), along with a correlation of the variables using .corr(); this function is named plot\_correlation():

Text

Description automatically generated

Lastly this new function is used to show the correlation of evictions per 1,000 against 2 income variables in the dataframe; household income, and household income MOE:

Text

Description automatically generated

# Results

Chart, line chart

Description automatically generated

Figure 1: Graph showing total eviction trends by borough from 2017 to 2020

Chart, line chart

Description automatically generated

Figure 2: Graph showing eviction trends per 1,000 population by borough from 2017 to 2020.

Table

Description automatically generated

Figure 3: Table of Evictions per 1,000 by borough and year, subsetted from the NY\_Merged dataframe.

Chart, scatter chart

Description automatically generated

Figure 4: Scatter plot of household income against evictions per 1,000, grouped by borough.

Chart, scatter chart

Description automatically generated

Figure 5: Correlation analysis of evictions per 1,000 against mean household income.

Chart, scatter chart

Description automatically generated

Figure 6: Correlation analysis of evictions per 1,000 against mean household income margin of error (standard deviation).

# Insights

* The scatter plot in figure 1 shows the Bronx has the highest eviction trends in the city, followed by Brooklyn and Queens. Adjusting for population in figure 2 the discrepancy is more stark – in 2017 the Bronx had 5.2 evictions per 1,000 people, compared to 2.3 and 2.04 for Brooklyn and Queens respectively. This indicates that the Bronx is most heavily affected by evictions for the city as a whole.
* The Bronx has more evictions per capita than the next 2 boroughs combined, as indicated by the table in figure 3, for 2017 through 2019. Note that the dropoff in evictions for 2020 can be attributed to reduced evictions in the city due to the pandemic.
* The scatter plot of household income against evictions per 1,000 in figure 4 shows the Bronx has the lowest mean household income along with the highest evictions per capita across the city.

# References

(Include any references if required)