Relationships between crime rate and housing prices in New York City

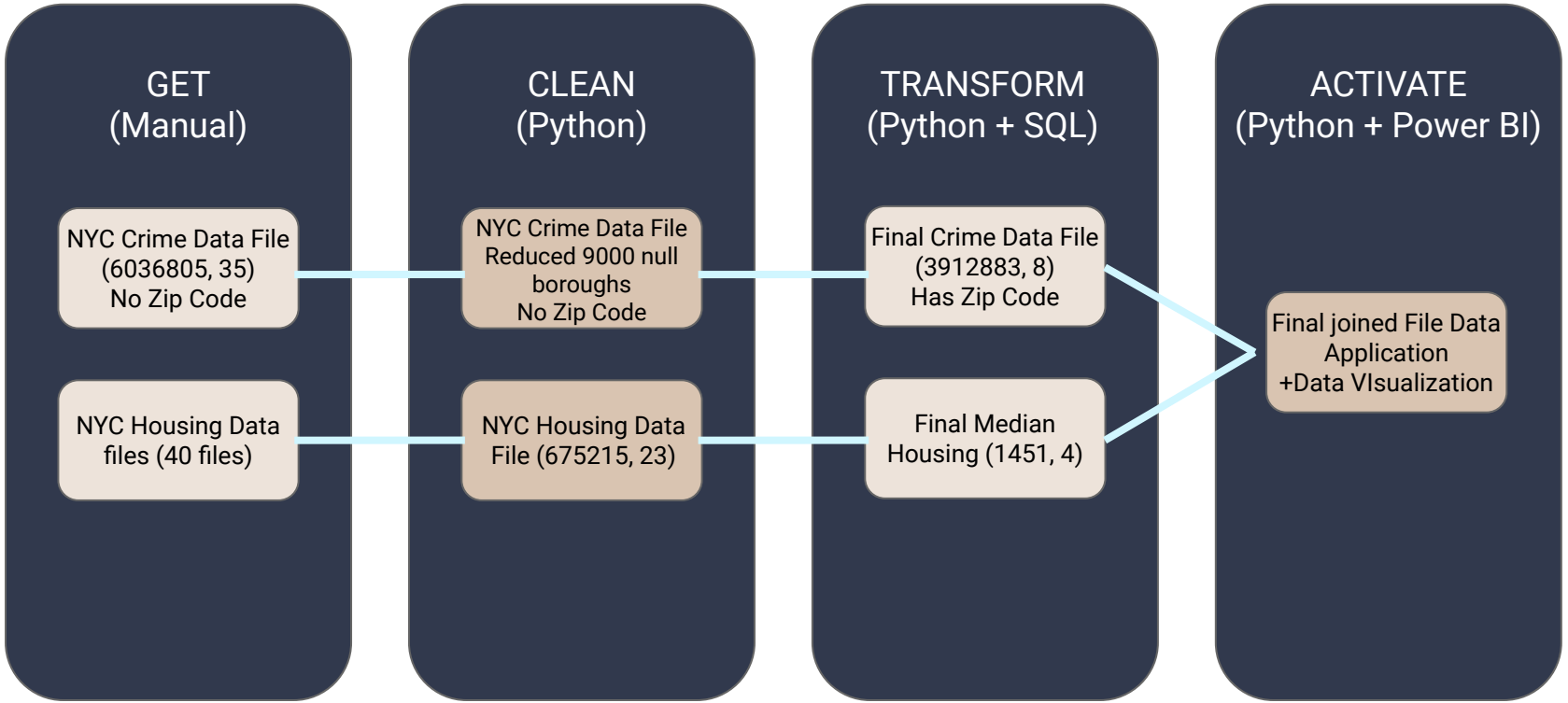
Team Anaconda 🐍:

Anirudh Dave, Samy Dolan, Navneet Poddar, Jiajun Yuan

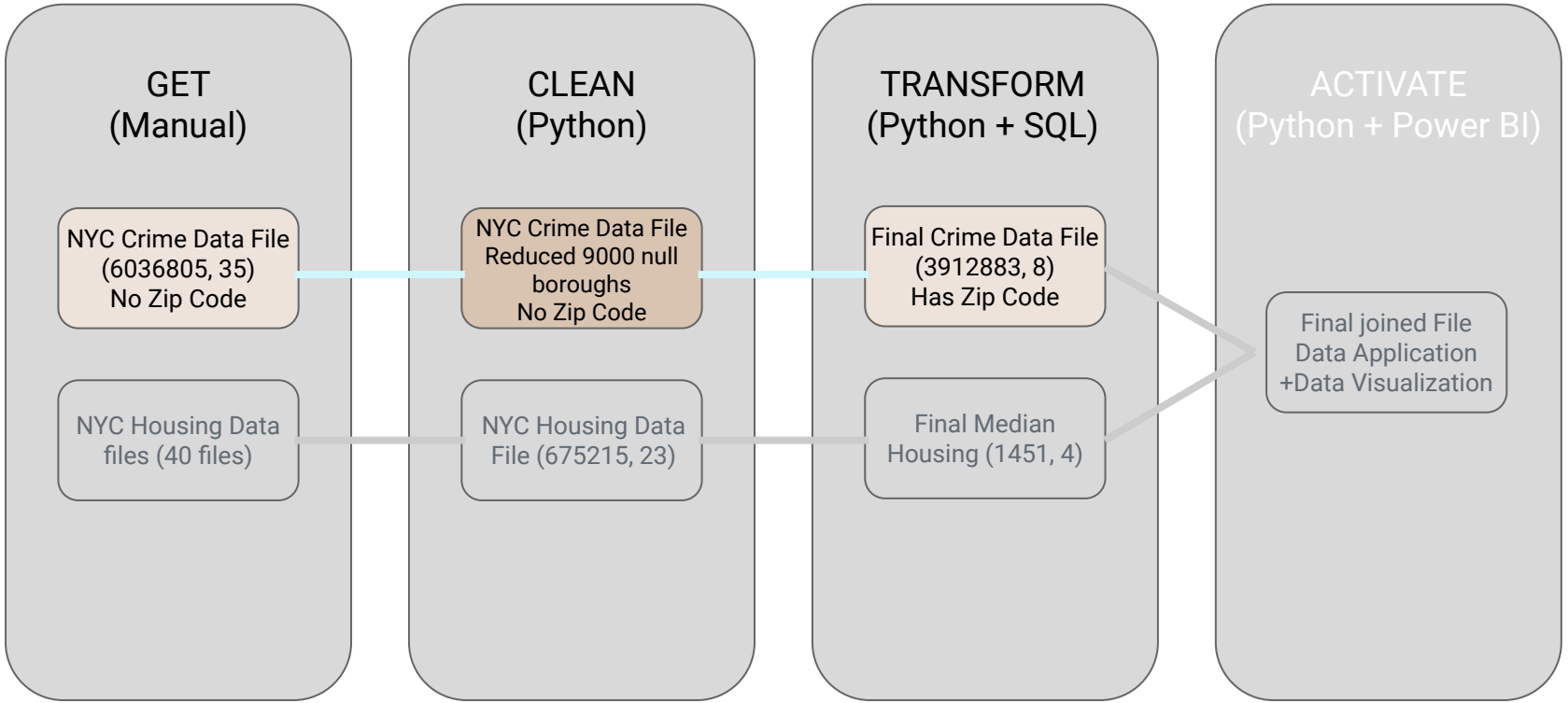
Problem Statement

The consultants of Team Anaconda are hired by a real estate company that is looking to invest in residential properties in New York City.

Project Pipeline



Project Pipeline



NYPD Complaints Data

- 35 columns and 6 million+ rows
- Trimmed down to 20 columns
- Dropped rows missing complaint origin date and latitudes data

- Processed the complaint origin date column

```
temp_raw['year'] = temp_raw.CMPLNT_FR_DT.str.split('/')
```

```
temp_raw[['month','day','year_for_complaint']] = pd.DataFrame(temp_raw.year.values.tolist(), index = temp_raw.index)
```

- Downsized the data by sub-setting the years between 2010 and 2017 inclusive

```
temp_raw = temp_raw[(temp_raw['year_for_complaint'] >= 2010) & (temp_raw['year_for_complaint'] <= 2017)]
```

Impute Where Possible...

- The column with borough names had nearly 10,219 null values
- 2 columns consisted of values that could fill borough names –
  - Borough Patrol Region
  - Precinct ID

```
temp_raw['BORO_NM'] = np.where(temp_raw['BORO_NM'].isnull() == True, temp_raw['PATROL_BORO'], temp_raw['BORO_NM'])
```

```
temp_raw['BORO_NM'] = np.where(temp_raw['BORO_NM'] == 'PATROL BORO BKLYN NORTH', 'BROOKLYN', temp_raw['BORO_NM'])
temp_raw['BORO_NM'] = np.where(temp_raw['BORO_NM'] == 'PATROL BORO BKLYN SOUTH', 'BROOKLYN', temp_raw['BORO_NM'])
```

```
temp_raw['BORO_NM'] = np.where((temp_raw['ADDR_PCT_CD'] >= 1.0) & (temp_raw['ADDR_PCT_CD'] <= 34.0), 'MANHATTAN' , temp_raw['BORO_NM'])
temp_raw['BORO_NM'] = np.where((temp_raw['ADDR_PCT_CD'] >= 40.0) & (temp_raw['ADDR_PCT_CD'] <= 52.0), 'BRONX' , temp_raw['BORO_NM'])
```

- These functions reduced the null values in the borough names column to 1327, imputing over 82% of the values
- The data is further reduced to 9 columns to improve processing speed for the next step

Postal Codes

- Unavailable in the complaints data
- Simple version of the ‘uszipcode’ package installed to write postal codes to the data frame.
- Latitude and longitude columns were converted to numpy array
- The search engine is activated for the uszipcode package and a UDF is created as follows –

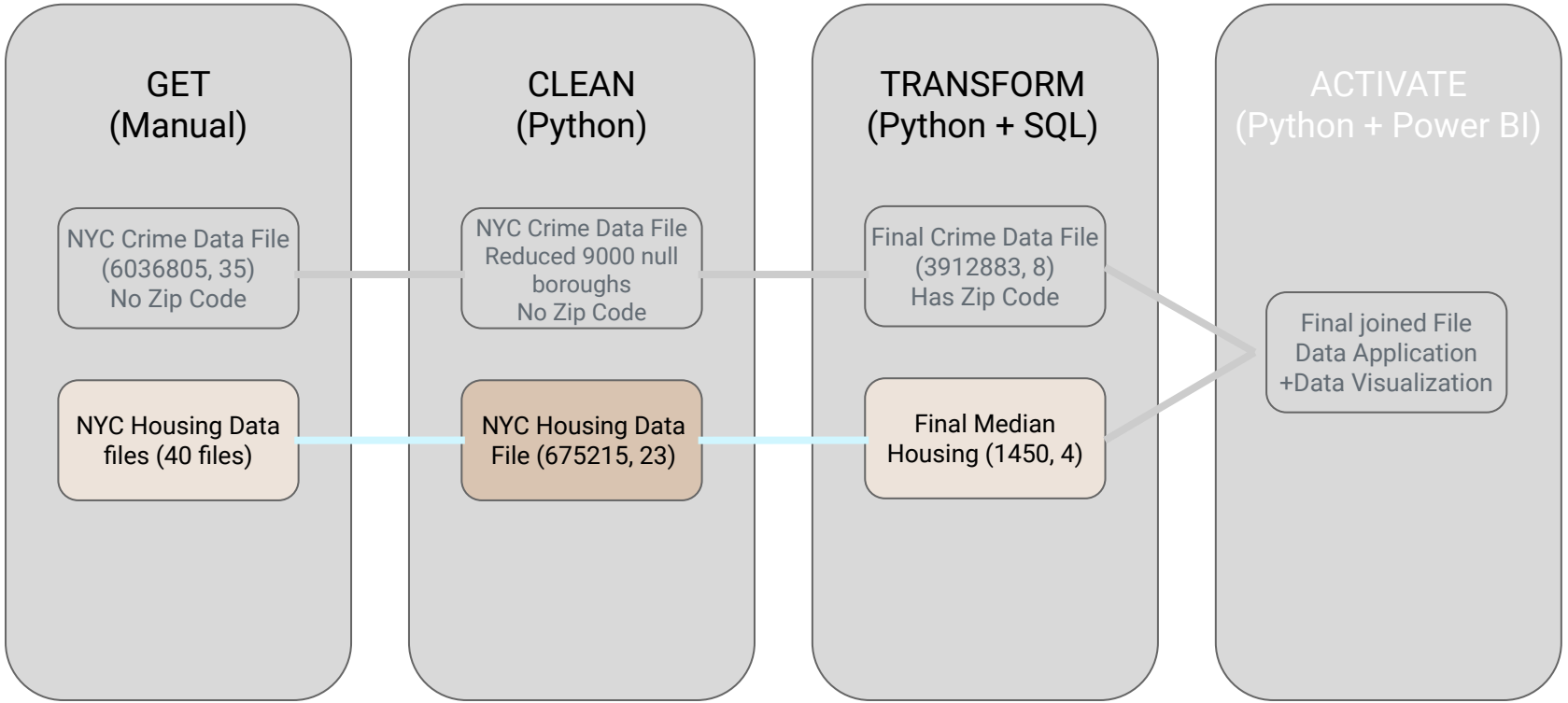
```
def enter_zipcode(a,b):
    #for each array
    for x in lat_long_matrix[a:b,:]:
        try:
            # 3rd position in each array is reassigned the zipcode value
            # since the data structure is not ideal, it is broken down by using indexing and values() function to return 1 zipcode value
            x[2] = search.by_coordinates(x[0], x[1], radius=3, returns=1)[0].values()[0]
        except:
            # if a zipcode cannot be found for a set of coordinates then write 'na' to the zipcode position
            x[2] = 'na'
```

Postal Codes (continue)

- The array is converted back to a dataframe and copied to the original dataframe
- The rows without postal codes or borough name are removed
- The final version of the data set has 8 columns and 3.7 million+ rows

	CMPLNT_NUM	ADDR_PCT_CD	KY_CD	OFNS_DESC	LAW_CAT_CD	BORO_NM	year_for_complaint	zipcode
0	712500291	45.0	233	SEX CRIMES	MISDEMEANOR	BRONX	2010	10465
1	919517331	44.0	235	DANGEROUS DRUGS	MISDEMEANOR	BRONX	2010	10456
2	439147997	24.0	578	HARRASSMENT 2	VIOLATION	MANHATTAN	2010	10026
3	697384902	66.0	360	NaN	MISDEMEANOR	BROOKLYN	2010	11218
4	500286412	71.0	352	CRIMINAL TRESPASS	MISDEMEANOR	BROOKLYN	2010	11213
5	421218921	52.0	578	HARRASSMENT 2	VIOLATION	BRONX	2010	10468
6	799980230	20.0	351	CRIMINAL MISCHIEF & RELATED OF	MISDEMEANOR	MANHATTAN	2010	10023

Project Pipeline



Raw Data #2

2010-2017 bronx, brooklyn, manhattan, queens, statenisland.xls (40 files)

2010 bronx.xls

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Bronx All Sales For 2010 (January 2010 - December 2010)																	
2	Neighborhood Name 05/06/12, Descriptive Data is as of 05/06/12																	
3	Building Class Category is based on Building Class at Time of Sale.																	
4	BOROUGH	NEIGHBORHOOD	BUILDING CLASS CATEGORY	TAX CLASS AT PRESENT	BLOCK	LOT	EASE-MENT	BUILDING CLASS AT PRESENT	ADDRESS	APARTMENT NUMBER	ZIP CODE	RESIDENTIAL UNITS	COMMERCIAL UNITS	TOTAL UNITS	LAND SQUARE FEET	GROSS SQUARE FEET	YEAR BUILT	TAX CLASS AT TIME OF SALE
5	2	BATHGATE	01 ONE FAMILY HOMES	1	3030	70		A1	4445 PARK AVENUE		10457	1	0	1	1,694	1,497	1899	1
6	2	BATHGATE	01 ONE FAMILY HOMES	1	3035	2		S1	441 EAST 178 STREET		10457	1	1	2	1,287	2,378	1899	1
7	2	BATHGATE	01 ONE FAMILY HOMES	1	3037	42		A1	4428 PARK AVENUE		10457	1	0	1	3,525	1,340	1899	1
8	2	BATHGATE	01 ONE FAMILY HOMES	1	3039	64		A1	467 EAST 185 STREET		10458	1	0	1	1,667	1,296	1910	1
9	2	BATHGATE	01 ONE FAMILY HOMES	1	3046	34		A1	2085 BATHGATE AVENUE		10457	1	0	1	2,060	1,629	1899	1
10	2	BATHGATE	01 ONE FAMILY HOMES	1	3046	42		A1	2069 BATHGATE AVENUE		10457	1	0	1	1,964	1,424	1899	1
11	2	BATHGATE	01 ONE FAMILY HOMES	1	3048	18		A9	2184 BATHGATE AVENUE		10457	1	0	1	1,768	1,236	1930	1
12	2	BATHGATE	01 ONE FAMILY HOMES	1	3048	19		A5	2186 BATHGATE AVENUE		10457	1	0	1	1,768	1,188	1901	1
13	2	BATHGATE	01 ONE FAMILY HOMES	1	3048	19		A5	2186 BATHGATE AVENUE		10457	1	0	1	1,768	1,188	1901	1
14	2	BATHGATE	01 ONE FAMILY HOMES	1	3048	28		A1	540 EAST 182 STREET		10457	1	0	1	1,209	1,048	1901	1
15	2	BATHGATE	02 TWO FAMILY HOMES	1	2912	137		B1	496 CLAREMONT PKWY		10457	2	0	2	2,000	2,400	1993	1
16	2	BATHGATE	02 TWO FAMILY HOMES	1	2927	123		B1	1489 CROTONA PLACE		10456	2	0	2	1,900	2,394	1994	1

2017 statenisland.xls

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	STATEN ISLAND ANNUALIZE SALE FOR 2017. (All Sales From January 1, 2017 - December 31, 2017)																		
2	Sales File as of 4/9/2018. Coop Sales Files as of 4/6/2018.																		
3	Neighborhood Name and Descriptive Data is as of Final Roll 17/18																		
4	Building Class Category is based on Building Class at Time of Sale.																		
5	BOROUGH	NEIGHBORHOOD	BUILDING CLASS CATEGORY	TAX CLASS AS OF FINAL ROLL 17/18	BLOCK	LOT	EASE-MENT	BUILDING CLASS AS OF FINAL ROLL 17/18	ADDRESS	APARTMENT NUMBER	ZIP CODE	RESIDENTIAL UNITS	COMMERCIAL UNITS	TOTAL UNITS	LAND SQUARE FEET	GROSS SQUARE FEET	YEAR BUILT	TAX CLASS AT TIME OF SALE	BUILDING CLASS AT TIME OF SALE
6	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	5395	19		A1	4 EDWIN STREET		10312	1	0	1	7,258	2,230	1980	1	A1
7	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	5407	10		A2	112 ELMBANK STREET		10312	1	0	1	6,242	1,768	1975	1	A2
8	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	5407	39		A2	193 BATHGATE STREET		10312	1	0	1	5,000	808	1920	1	A2
9	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	5426	32		A6	3 OCEAN DRIVEWAY		10312	1	0	1	2,500	540	1910	1	A6
10	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	6205	15		A5	95 EAGAN AVENUE		10312	1	0	1	1,546	1,579	1986	1	A5
11	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	6205	22		A5	83 EAGAN AVENUE		10312	1	0	1	1,546	1,579	1986	1	A5
12	5	ANNADALE	01 ONE FAMILY DWELLINGS	1	6205	58		A5	32 SEGUINE PLACE		10312	1	0	1	1,471	1,068	1986	1	A5

NYC Housing Data

- Merging 40 data files into one dataframe

```
path = os.getcwd()
files = os.listdir(path)

files_xls = [f for f in files if f[-3:] == 'xls'] # the housing data files are all .xls, so we are

nyc_housing_data = pd.DataFrame()

for f in files_xls:
    if f[3] == '0':         # need to create a certain rule around the 2010 files
        data = pd.read_excel(f, header=3, index_col=None) # the column names for the files 2011 -
        data.rename(columns=lambda x: x.strip(), inplace= True)
        nyc_housing_data = nyc_housing_data.append(data, sort = False)
    else:
        data = pd.read_excel(f, header=4, index_col=None)
        data.rename(columns=lambda x: x.strip(), inplace= True)
        nyc_housing_data = nyc_housing_data.append(data, sort = False)
```

- Merge result:

```
# getting the specs of the dataframe
nyc_housing_data.shape
```

```
(675215, 23)
```

Cleaning It Up

- Creating a year of sale column within the dataframe
  - Extracting year from “SALE DATE” column
- Getting the data trimmed down to needed rows
  - Eliminated commercial properties
  - Eliminated any sales less than 50k
  - Eliminated rows with no zip code

```
nyc_housing_data['SALE YEAR'] = nyc_housing_data['SALE DATE'].dt.year
```

```
nyc_housing_data_clean = nyc_housing_data_clean[nyc_housing_data_clean['COMMERCIAL UNITS'] < 1]
nyc_housing_data_clean = nyc_housing_data_clean[nyc_housing_data_clean['RESIDENTIAL UNITS'] < 5]
nyc_housing_data_clean = nyc_housing_data_clean[nyc_housing_data_clean['SALE PRICE'] > 50000]
nyc_housing_data_clean = nyc_housing_data_clean[nyc_housing_data_clean['ZIP CODE'] >0]
```

Removing Non-Residential Properties

- Examined other non-residential properties and eliminated the necessary building classes.
- Eliminated spaces within the rows

```
nyc_housing_data_clean['BUILDING CLASS CATEGORY'].unique()
```

```
array(['01  ONE FAMILY DWELLINGS                    ',
       '02  TWO FAMILY DWELLINGS                    ',
       '04  TAX CLASS 1 CONDOS                      ',
       '05  TAX CLASS 1 VACANT LAND                 ',
```

```
nyc_housing_data_clean['BUILDING CLASS CATEGORY'] = nyc_housing_data_clean['BUILDING CLASS CATEGORY'].map(lambda x: x.strip())
remove_list = ["05 TAX CLASS 1 VACANT LAND","05 TAX CLASS 1 VACANT LAND","06 TAX CLASS 1 - OTHER","06 TAX CLASS 1 - OTHE
nyc_housing_data_clean = nyc_housing_data_clean[~nyc_housing_data_clean['BUILDING CLASS CATEGORY'].isin(remove_list)]
```

Getting the Median Housing Price

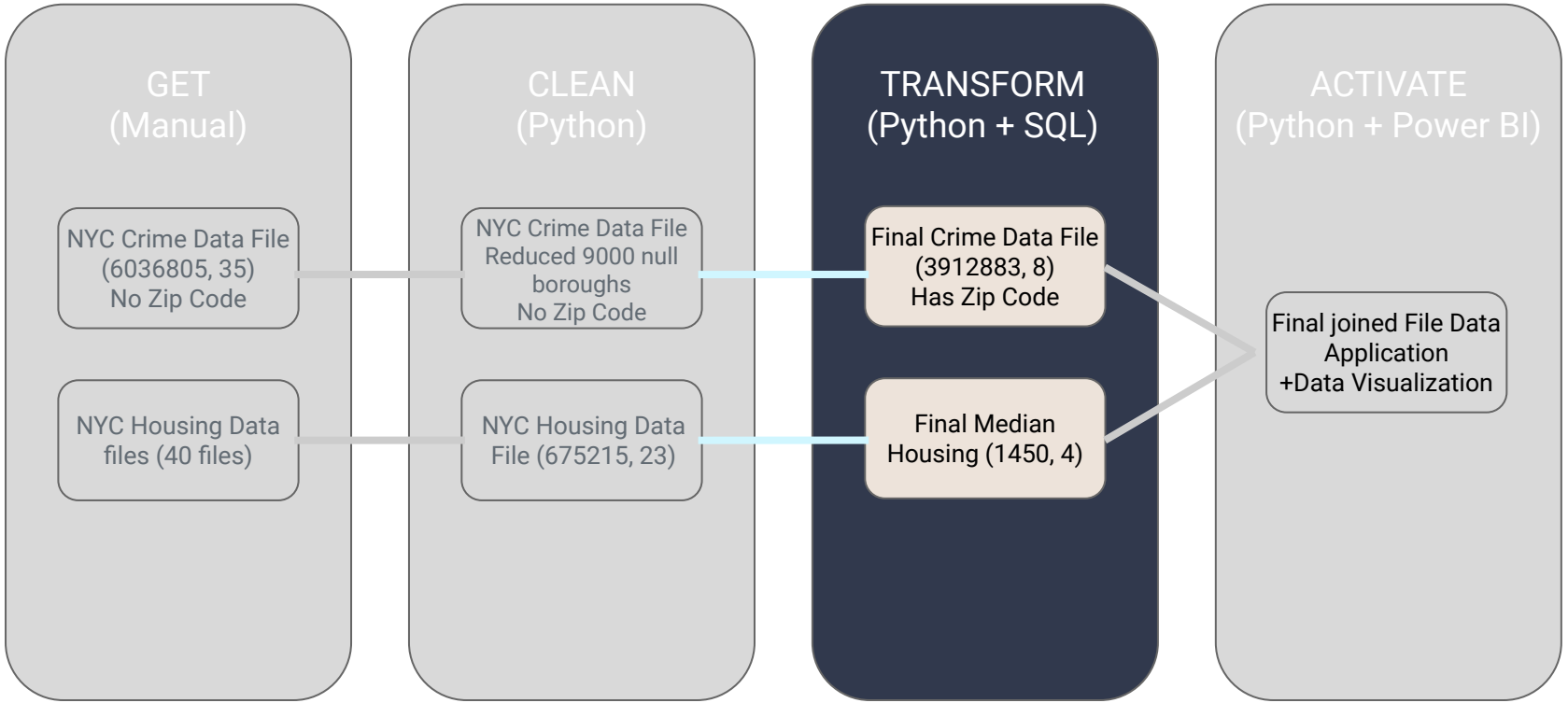
- To create the final dataset needed, we used the cleaned housing dataset to create a table with the median property value for each zip code for each year.
- We then exported the final data set - The final version of the data set has 4 columns and 1,450 rows

```
nyc_housing_data_summary=nyc_housing_data_clean.groupby(['ZIP CODE','SALE YEAR'], as_index=False)['SALE PRICE'].median()
nyc_housing_data_summary2=nyc_housing_data_clean.groupby(['ZIP CODE','SALE YEAR'], as_index=False)['SALE PRICE'].count()
nyc_housing_data_summary['Sales Count'] = nyc_housing_data_summary2['SALE PRICE'].values
nyc_housing_data_summary.rename(columns={'SALE PRICE': 'Median Sale Price'}, inplace=True)
nyc_housing_data_summary.to_csv("C:/Users/owner/Downloads/DWD/Final_Median_Housing_Data.csv",index=False)
```

Transform #2
Final_Median_Housing_Data.csv

	A	B	C	D
1	ZIP CODE	SALE YEAR	Median Sale Price	Sales Count
2	10001	2010	1237173	91
3	10001	2011	1120000	169
4	10001	2012	1090000	217
5	10001	2013	705000	209
6	10001	2014	910000	214
7	10001	2015	960000	205
8	10001	2016	1152500	196
9	10001	2017	810000	177
10	10002	2010	615475	134
11	10002	2011	530000	212
12	10002	2012	496000	279
13	10002	2013	540000	353
14	10002	2014	616000	285
15	10002	2015	690500	244
16	10002	2016	795000	221
17	10002	2017	808000	243
18	10003	2010	958650	353
19	10003	2011	806500	626
20	10003	2012	812500	642
21	10003	2013	839500	726
22	10003	2014	1125000	795
23	10003	2015	1115000	671
24	10003	2016	1150000	591
25	10003	2017	1225000	551
26	10004	2010	722957	61
27	10004	2011	818519.5	70
28	10004	2012	842500	68
29	10004	2013	876998	98
30	10004	2014	1032805.5	110
31	10004	2015	1163859	113
32	10004	2016	1160000	81
33	10004	2017	1095000	71

Project Pipeline



Joining Using SQL + Python

- SQLite3 package and pandas are used for this stage
- The connection and database cursor are defined and two tables created –

```
# Create table - crimes that corresponds to crime data csv file
conn_cur.execute('''CREATE TABLE crimes
            ([CMPLNT_NUM] VARCHAR,[ADDR_PCT_CD] VARCHAR, [KY_CD] VARCHAR, [OFNS_DESC] VARCHAR,
            [LAW_CAT_CD] VARCHAR, [BORO_NM] VARCHAR, [year_for_complaint] VARCHAR, [zipcode] VARCHAR)''')

# Create table - rental that corresponds to  housing data csv file
conn_cur.execute('''CREATE TABLE rental
            ([ZIP CODE] VARCHAR,[SALE YEAR] VARCHAR, [Median Sale Price] VARCHAR, [Sales Count] VARCHAR)''')
```

Writing to the Tables and Querying the Data

- The csv files are written to the database tables are follows –

```
# reading the final_crime_file created from file 2
crime_data = pd.read_csv('C:/Users/aniru/Desktop/DWD/final_crime_file.csv')

# writing the crime_data dataframe to the crimes table in the database
crime_data.to_sql('crimes', connection, if_exists = 'replace', index = False)
```

- The query used to join the two tables is –

```
conn_cur.execute('''
SELECT *
from crimes
inner join rental on (crimes.zipcode || crimes.year_for_complaint = rental."ZIP CODE" || rental."SALE YEAR");
 ''')

# storing the joined table to an object called results
results = conn_cur.fetchall()
```

Storing the Joined Data as a Pandas Dataframe

- The column headers are extracted using the description function
- A for loop to store header names in a list
- The final dataframe is created as follows –

```
hdr = []
for header in headers:
    hdr.append(header[0])
```

```
final_joined_df = pd.DataFrame(results, columns = hdr)
```

```
# dropping columns that are not needed for further analysis
final_joined_df = final_joined_df.drop(final_joined_df.columns[[1, 8, 9]], axis=1)
```

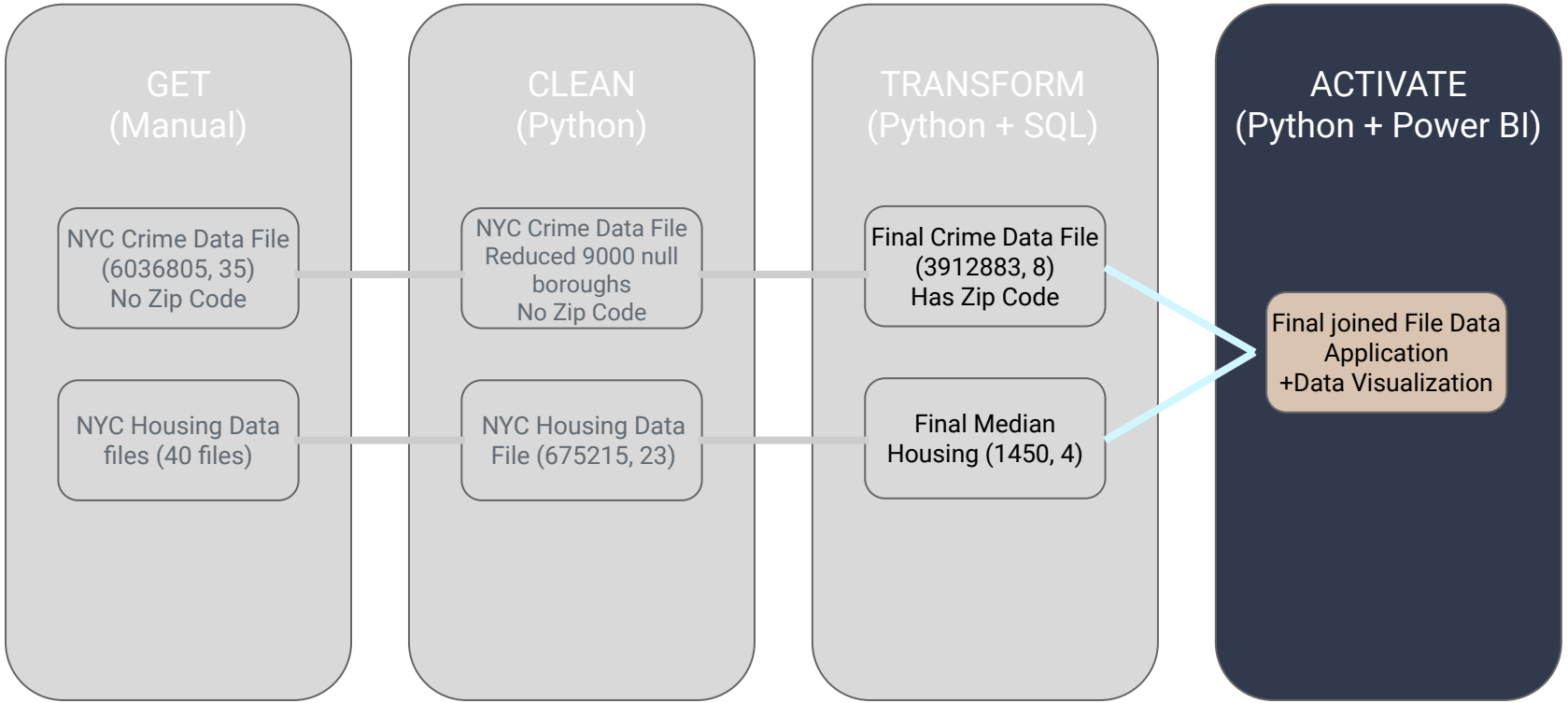
- The columns are renamed for readability –

```
# renaming the columns in final_joined_df
final_joined_df = final_joined_df.rename(columns = {'KY_CD':'OFNS_CODE','LAW_CAT_CD':'OFNS_LEVEL',
                                                    'BORO_NM':'BOROUGH','year_for_complaint':'YEAR','zipcode':'ZIPCODE',
                                                    'Median Sale Price':'MEDIAN_SALE','Sales Count':'SALES_COUNT'})
```

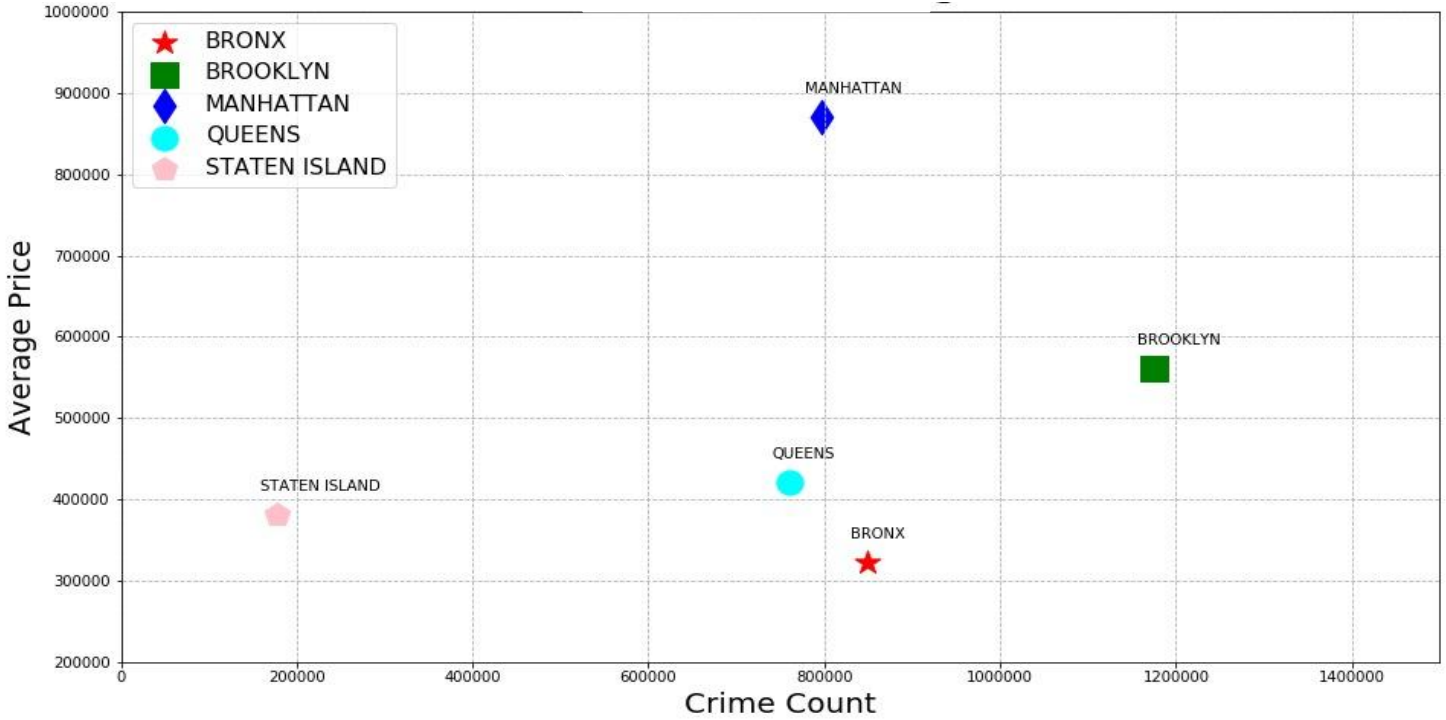
Activate
Final_Joined_File.csv

	A	B	C	D	E	F	G	H	I
1	CMPLNT_NUM	OFNS_CODE	OFNS_DESC	OFNS_LEVEL	BOROUGH	YEAR	ZIPCODE	MEDIAN_SALE	SALES_COUNT
2	712500291	233	SEX CRIMES	MISDEMEANOR	BRONX	2010	10465	398265.5	204
3	919517331	235	DANGEROUS DRUGS	MISDEMEANOR	BRONX	2010	10456	377975	64
4	439147997	578	HARRASSMENT 2	VIOLATION	MANHATTAN	2010	10026	650000	247
5	697384902	360		MISDEMEANOR	BROOKLYN	2010	11218	495000	231
6	500286412	352	CRIMINAL TRESPASS	MISDEMEANOR	BROOKLYN	2010	11213	475000	127
7	421218921	578	HARRASSMENT 2	VIOLATION	BRONX	2010	10468	222500	30
8	799980230	351	CRIMINAL MISCHIEF & RELATED OF	MISDEMEANOR	MANHATTAN	2010	10023	957960.5	720
9	879729896	344	ASSAULT 3 & RELATED OFFENSES	MISDEMEANOR	BRONX	2010	10472	410000	72
10	503691170	105	ROBBERY	FELONY	BROOKLYN	2010	11226	392500	133
11	443939472	105	ROBBERY	FELONY	BROOKLYN	2010	11205	427500	269
12	442740928	107	BURGLARY	FELONY	STATEN ISLAND	2010	10307	540000	112
13	864275387	578	HARRASSMENT 2	VIOLATION	BROOKLYN	2010	11211	570000	837
14	397050965	351	CRIMINAL MISCHIEF & RELATED OF	MISDEMEANOR	QUEENS	2010	11378	480000	213
15	411794554	344	ASSAULT 3 & RELATED OFFENSES	MISDEMEANOR	BRONX	2010	10469	382640	252
16	958870904	578	HARRASSMENT 2	VIOLATION	MANHATTAN	2010	10031	373152	75
17	370858177	578	HARRASSMENT 2	VIOLATION	MANHATTAN	2010	10031	373152	75
18	273323808	341	PETIT LARCENY	MISDEMEANOR	BROOKLYN	2010	11212	350000	97
19	672060879	578	HARRASSMENT 2	VIOLATION	QUEENS	2010	11368	470000	252
20	251644413	351	CRIMINAL MISCHIEF & RELATED OF	MISDEMEANOR	BROOKLYN	2010	11226	392500	133

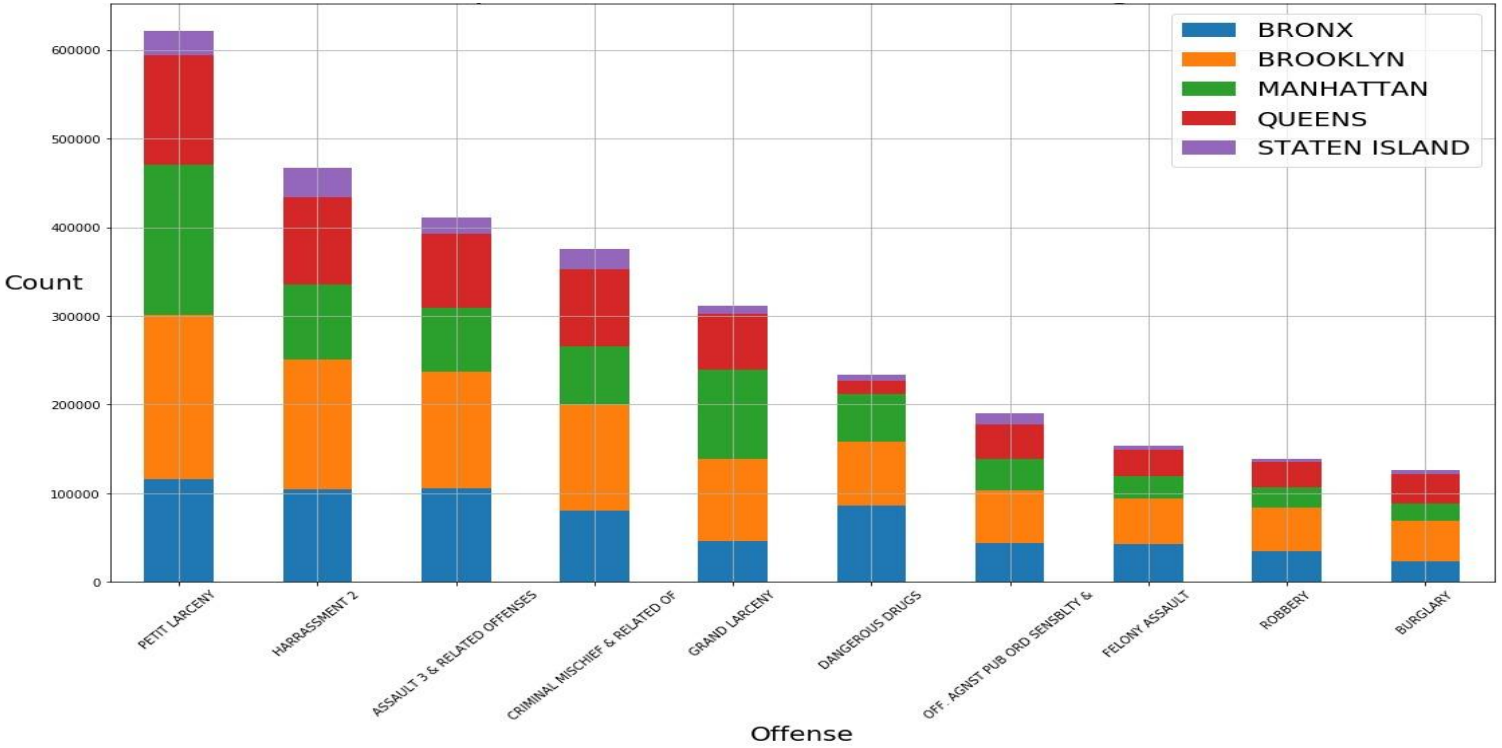
Project Pipeline



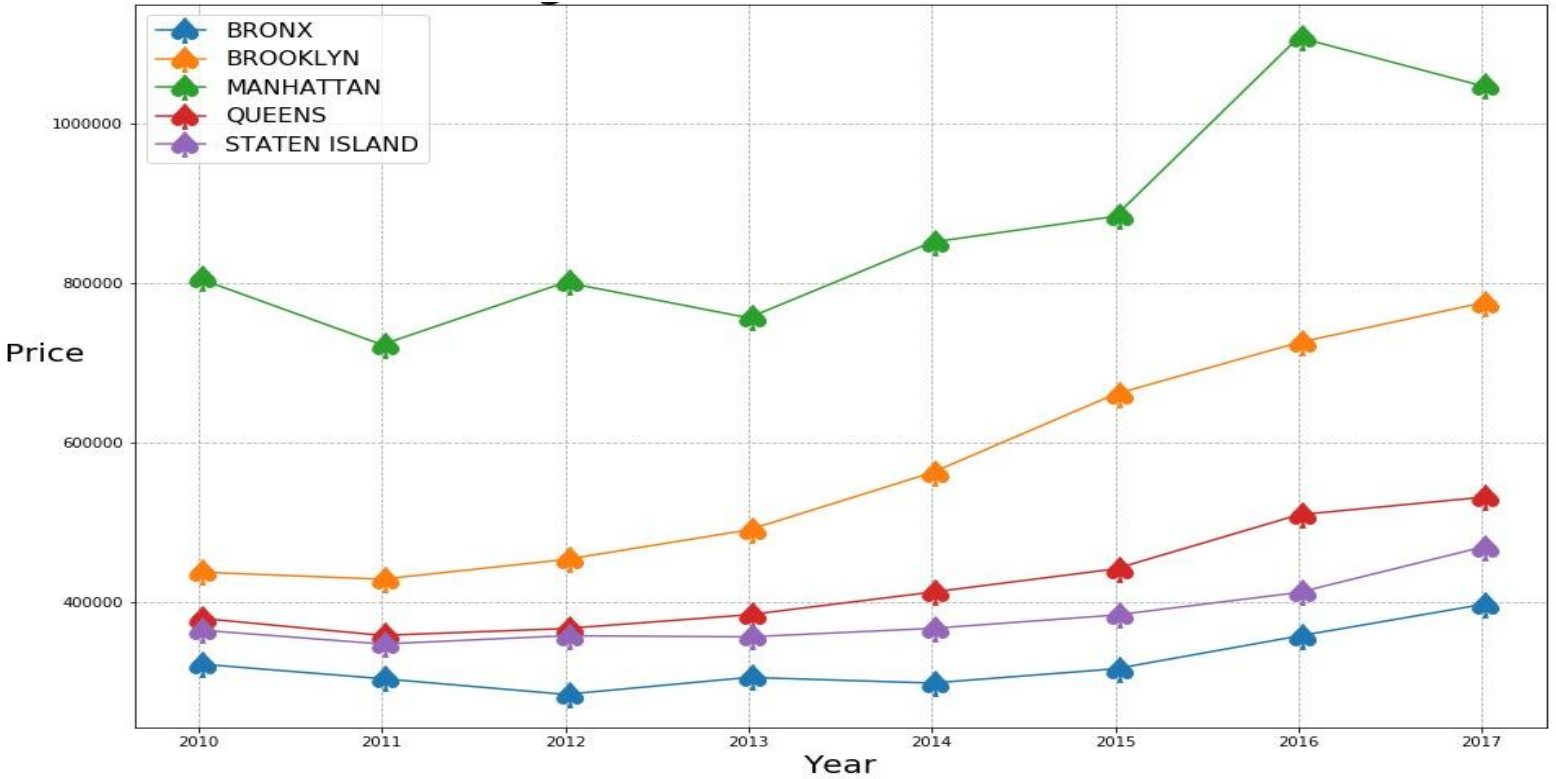
Housing Price vs. Crime across Boroughs



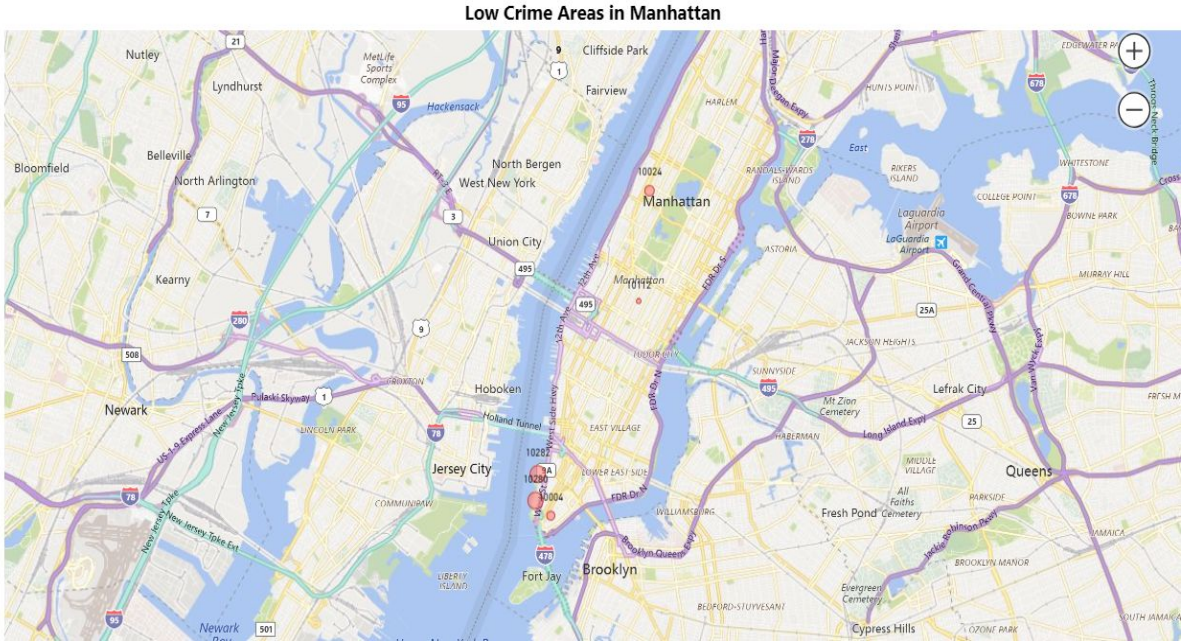
Top crimes distribution in 5 boroughs

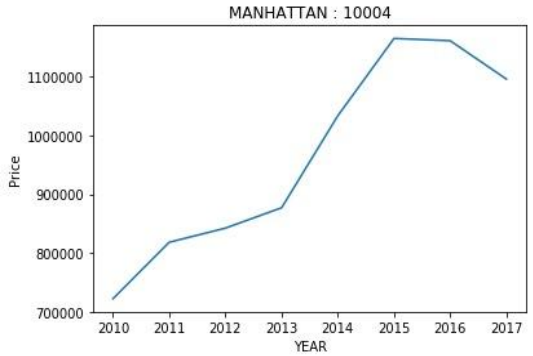


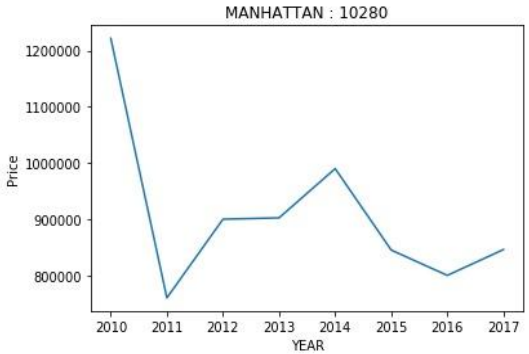
Housing Price Variance Year-over-Year



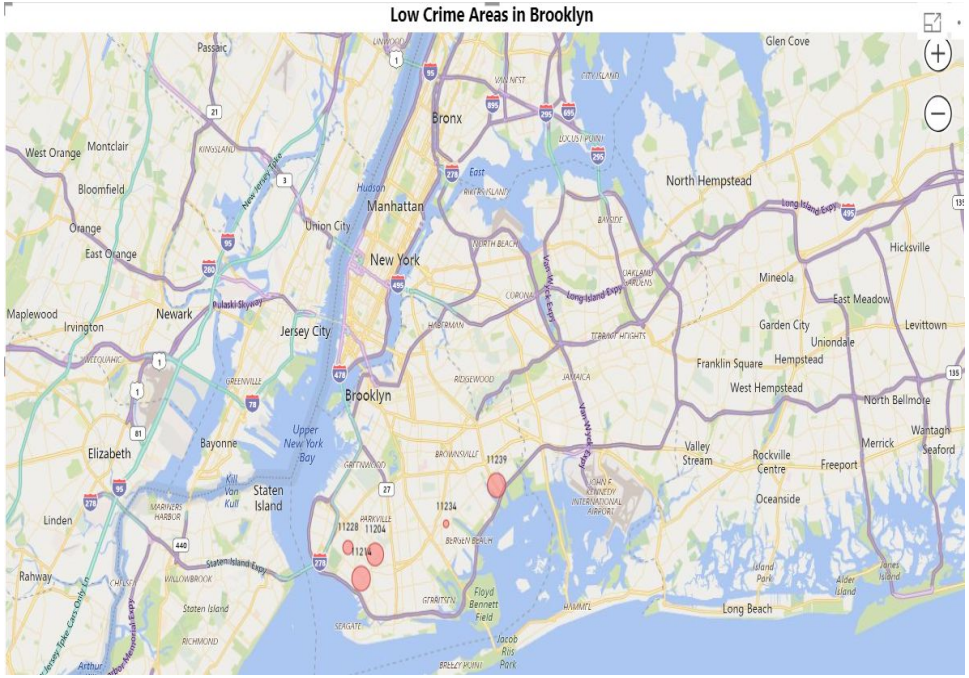
Housing Value Trend in Low Crime Areas of Manhattan

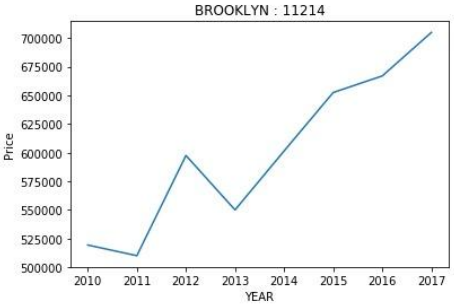


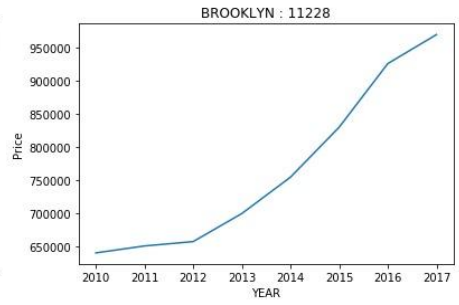


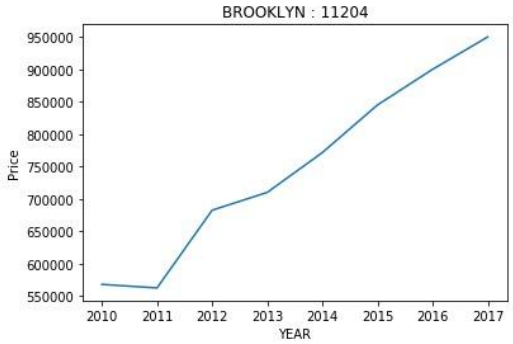


Housing Value Trend in Low Crime Areas of Brooklyn

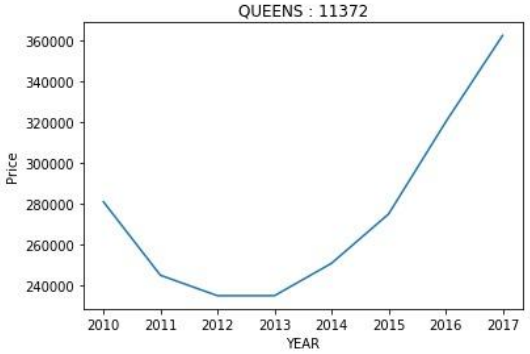


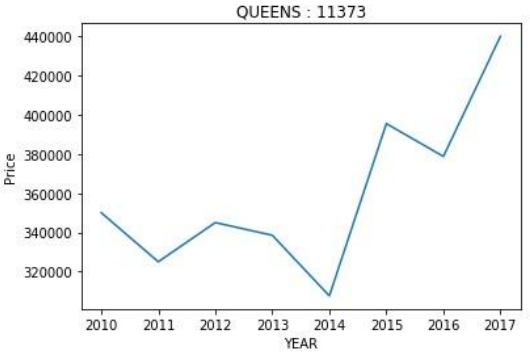






Housing Value Trend in Low Crime Areas of Queens





Recommendation

Risk Level	Borough	Zip Codes
High Risk Option	Manhattan	10004, 10280
Moderate Risk Option	Brooklyn	11204, 11214, 11228
Low Risk Option	Queens	11372, 11373


END OF PRESENTATION
THANK YOU FOR YOUR TIME