msmdesign.nyc

data analytics & insights information design

exploratorydataanalysis

> ## What is Exploratory Data Analysis?
>
> **Exploratory data analysis (EDA)** is a technique used by data scientists to inspect, characterize and briefly summarize the contents of a dataset. EDA is often the first step when encountering a new or unfamiliar dataset. EDA helps the data scientist become acquainted with a dataset and test some basic assumptions about the data. By the end of the EDA process, some initial insights can be drawn from the dataset and a framework for further analysis or modeling is established.

Storefronts Reported Vacant or Not

Dataset Analyzed: *Storefronts Reported Vacant or Not*

About This Dataset: Storefronts Reported Vacant or Not. The dataset was created to comply with Local Law 157 of 2019. Data is collected using an online portal, which allows owners to provide information about ground and second floor storefronts on their properties.

Each row shows a ground floor or second floor storefront that was registered with the department as of December 31 of the reporting year and legally required updates provided as of June 30 (or date sold if earlier) of the calendar year immediately following the reporting year. Each row contains the property's borough, block and lot number and the storefront's street address (and zip code), either field can be used to search for individual storefronts. Data provided by the Department of Department of Finance (DOF), the City of New York: https://data.cityofnewyork.us/Business/NYC-Business-Acceleration-Businesses-Served-and-Jo/9b9u-8989

Acknowledgements: NYC Open Data https://opendata.cityofnewyork.us/

EDA Catalogue Number: INS-011

EDA Publication Date: Thursday, January 12, 2023

Language: Python

Libraries Used: NumPy, pandas, matplotlib, seaborn

EDA Author: David White

Contact: david@msmdesign.nyc | msmdesign.nyc

---

0. Prepare the workspace

0.1 Import Python libraries, packages and functions

In [2]:
```
# import libraries for data wrangling, aggregate functions and basic descriptive stati
import numpy as np
import pandas as pd

# import data visualization packages
import matplotlib.pyplot as plt
import seaborn as sns
```

0.2 Adjust display options to make plots easier to read and understand

In [3]:
```
# specify seaborn styling options
sns.set_theme(
    context='talk',
    style='whitegrid',
    palette='viridis',
    font='Courier New',
    font_scale=1.15)

# allow plots to display inline within the notebook
%matplotlib inline
```

0.3 Set Markdown tables to align-left within notebook cells

In [4]:
```
%%html
<style>
table {float:left}
</style>
```

0.4 Display all rows of output by default

In [5]:
```
pd.set_option('display.max_rows', None)

# to reset:
# pd.reset_option('display.max_rows')
```

0.5 Format large numbers and display floating point values to two decimal places

In [6]:
```
pd.set_option('display.float_format',  '{:,.2f}'.format)

# to reset:
# pd.reset_option('display.float_format')
```

0.6 Load the raw data file into the notebook and visually confirm that it has been read in as expected

In [7]:
```
# load the data from a csv file (stored locally) into a new DataFrame object

csv = r"F:\Creative Cloud Files\MSM Client 001 - Mister Shepherd Media LLC\MSM Design\

storefronts = pd.read_csv(csv, encoding='utf-8', low_memory=False)
```

In [8]:
```
# glimpse the first three rows

storefronts.head(3)
```

Out[8]:

	Reporting Year	BOROUGH-BLOCK-LOT	PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS	BOROUGH	ZIP CODE	SOLD DATE	VACANT ON 12/31	CONSTRUCTION REPORTED	VAC 6/3(D SOL EAR
0	2019 and 2020	1000010010	1 GOVERNORS ISLAND	MANHATTAN	10004	NaN	YES	NaN	
1	2019 and 2020	1000050010	115 BROAD STREET	MANHATTAN	10004	NaN	NO	NaN	
2	2019 and 2020	1000050010	115 BROAD STREET	MANHATTAN	10004	NaN	NO	NaN	

3 rows × 24 columns

In [9]:
```
# glimpse the last three rows

storefronts.tail(3)
```

Out[9]:

	Reporting Year	BOROUGH-BLOCK-LOT	PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS	BOROUGH	ZIP CODE	SOLD DATE	VACANT ON 12/31	CONSTRUCTION REPORTED
214243	2021 and 2022	5080460001	7423-25 Amboy Road	STATEN ISLAND	10307	NaN	NO	NaN
214244	2021 and 2022	5080460013	7447 AMBOY RD	STATEN ISLAND	10307	NaN	NO	NaN
214245	2021 and 2022	5080470023	241 Main street	STATEN ISLAND	10307	NaN	NO	NaN

3 rows × 24 columns

In [10]:
```
# glimpse ten randomly selected rows

storefronts.sample(10, random_state=89)
```

Out[10]:

	Reporting Year	BOROUGH-BLOCK-LOT	PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS	BOROUGH	ZIP CODE	SOLD DATE	VACANT ON 12/31	CONSTRUCTION REPORTED
53822	2019 and 2020	3076180138	1777 FLATBUSH AVENUE	BROOKLYN	11210	NaN	NO	NaN
96104	2020 and 2021	1015340012	226 East 89th St.	MANHATTAN	10128	NaN	NO	NaN
68834	2019 and 2020	4095650044	128-11 LIBERTY AVENUE	QUEENS	11419	NaN	YES	NaN
53433	2019 and 2020	3074220002	2952 AVENUE X	BROOKLYN	11235	NaN	NO	NaN
37907	2019 and 2020	3002020001	28 OLD FULTO N ST	BROOKLYN	11201	NaN	NO	NaN
100648	2020 and 2021	1021420237	700 West 175 Street Store #5	MANHATTAN	10033	NaN	NO	NaN
184697	2021 and 2022	3005990002	201 RICHARDS STREET	BROOKLYN	11231	NaN	NO	NaN
2270	2019 and 2020	1002390038	90 BOWERY	MANHATTAN	10013	NaN	YES	NaN
44191	2019 and 2020	3026500035	139 NASSAU AVENUE	BROOKLYN	11222	NaN	NO	NaN
207318	2021 and 2022	4060770048	34-51 FRANCIS LEWIS BLVD	QUEENS	11358	NaN	NO	NaN

10 rows × 24 columns

The data has been loaded and has been read in as expected.

0.7. Check the data type of each column

In [11]:
```
# display a listing of each of the DataFrame's columns and its data type
```

```
storefronts.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214246 entries, 0 to 214245
Data columns (total 24 columns):
 #   Column                                          Non-Null Count   Dtype  
---  ------                                          --------------   -----  
 0   Reporting Year                                  214246 non-null  object 
 1   BOROUGH-BLOCK-LOT                               214246 non-null  int64  
 2   PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS   214245 non-null  object 
 3   BOROUGH                                         214246 non-null  object 
 4   ZIP CODE                                        213175 non-null  object 
 5   SOLD DATE                                       348 non-null     object 
 6   VACANT ON 12/31                                 214246 non-null  object 
 7   CONSTRUCTION REPORTED                           1166 non-null    object 
 8   VACANT 6/30 OR DATE SOLD IF EARLIER             8838 non-null    object 
 9   PRIMARY BUSINESS ACTIVITY                       214246 non-null  object 
 10  PROPERTY NUMBER                                 213655 non-null  object 
 11  PROPERTY STREET                                 214062 non-null  object 
 12  UNIT                                            21107 non-null   object 
 13  BOROUGH 1                                       214246 non-null  object 
 14  POSTCODE                                        214246 non-null  object 
 15  LATITUDE                                        208281 non-null  float64
 16  LONGITUDE                                       208281 non-null  float64
 17  COMMUNITY BOARD                                 214074 non-null  float64
 18  COUNCIL DISTRICT                                214199 non-null  float64
 19  CENSUS TRACT                                    214245 non-null  object 
 20  BIN                                             213263 non-null  float64
 21  BBL                                             214246 non-null  int64  
 22  NTA                                             214163 non-null  object 
 23  NBHD                                            214166 non-null  object 
dtypes: float64(5), int64(2), object(17)
memory usage: 39.2+ MB
```

0.8 Refer to the data dictionary and make sure that our DataFrame's data types match the source data. Reassign data types where needed.

In [12]:
```
# cast column(s) containing categorical varibles to categorical data type

storefronts['Reporting Year']                       = storefronts['Reporting Year'].ast
storefronts[ 'PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS'] = storefronts[ 'PROPERTY
storefronts['BOROUGH']                              = storefronts['BOROUGH'].astype('ca
storefronts['VACANT ON 12/31']                      = storefronts['VACANT ON 12/31'].as
storefronts['CONSTRUCTION REPORTED']                = storefronts['CONSTRUCTION REPORTE
storefronts['VACANT 6/30 OR DATE SOLD IF EARLIER'] = storefronts['VACANT 6/30 OR DATE
storefronts['PRIMARY BUSINESS ACTIVITY']            = storefronts['PRIMARY BUSINESS ACT
storefronts['COMMUNITY BOARD']                      = storefronts['COMMUNITY BOARD'].as
storefronts['COUNCIL DISTRICT']                     = storefronts['COUNCIL DISTRICT'].a
storefronts['NBHD']                                 = storefronts['NBHD'].astype('categ
```

In [13]:
```
# display the DataFrame info once again to confirm that the data type changes have bee

storefronts.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214246 entries, 0 to 214245
Data columns (total 24 columns):
 #   Column                                          Non-Null Count   Dtype
---  ------                                          --------------   -----
 0   Reporting Year                                  214246 non-null  category
 1   BOROUGH-BLOCK-LOT                               214246 non-null  int64
 2   PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS   214245 non-null  category
 3   BOROUGH                                         214246 non-null  category
 4   ZIP CODE                                        213175 non-null  object
 5   SOLD DATE                                       348 non-null     object
 6   VACANT ON 12/31                                 214246 non-null  category
 7   CONSTRUCTION REPORTED                           1166 non-null    category
 8   VACANT 6/30 OR DATE SOLD IF EARLIER             8838 non-null    category
 9   PRIMARY BUSINESS ACTIVITY                       214246 non-null  category
 10  PROPERTY NUMBER                                 213655 non-null  object
 11  PROPERTY STREET                                 214062 non-null  object
 12  UNIT                                            21107 non-null   object
 13  BOROUGH 1                                       214246 non-null  object
 14  POSTCODE                                        214246 non-null  object
 15  LATITUDE                                        208281 non-null  float64
 16  LONGITUDE                                       208281 non-null  float64
 17  COMMUNITY BOARD                                 214074 non-null  category
 18  COUNCIL DISTRICT                                214199 non-null  category
 19  CENSUS TRACT                                    214245 non-null  object
 20  BIN                                             213263 non-null  float64
 21  BBL                                             214246 non-null  int64
 22  NTA                                             214163 non-null  object
 23  NBHD                                            214166 non-null  category
dtypes: category(10), float64(3), int64(2), object(9)
memory usage: 28.4+ MB
```

---

1. Describe the characteristics of the dataset

1.1 How many rows and how many columns are in our dataset?

In [14]:
```
# display the number of rows and columns in the DataFrame

rows = storefronts.shape[0]
columns = storefronts.shape[1]

print(f'There are {rows} rows and {columns} columns in the dataset.')
```

```
There are 214246 rows and 24 columns in the dataset.
```

1.2 Identify the index of our DataFrame

In [15]:
```
# display the index of the DataFrame

storefronts.index
```

Out[15]:
```
RangeIndex(start=0, stop=214246, step=1)
```

Our DataFrame has an interger index. We know from the data dictionary that each row is an individual constituent case.

1.3 What are the column headings in our dataset?

In [16]:
```
# display a list of the DataFrame's columns

list(storefronts.columns)
```

Out[16]:
```
['Reporting Year',
 'BOROUGH-BLOCK-LOT',
 'PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS',
 'BOROUGH',
 'ZIP CODE',
 'SOLD DATE',
 'VACANT ON 12/31',
 'CONSTRUCTION REPORTED',
 'VACANT 6/30 OR DATE SOLD IF EARLIER',
 'PRIMARY BUSINESS ACTIVITY',
 'PROPERTY NUMBER',
 'PROPERTY STREET',
 'UNIT',
 'BOROUGH 1',
 'POSTCODE',
 'LATITUDE',
 'LONGITUDE',
 'COMMUNITY BOARD',
 'COUNCIL DISTRICT',
 'CENSUS TRACT',
 'BIN',
 'BBL',
 'NTA',
 'NBHD']
```

1.4 What are the data types of each column?

In [17]:
```
# display the data type of each column in the DataFrame

storefronts.dtypes
```

Out[17]:
```
Reporting Year                                    category
BOROUGH-BLOCK-LOT                                    int64
PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS    category
BOROUGH                                          category
ZIP CODE                                           object
SOLD DATE                                          object
VACANT ON 12/31                                  category
CONSTRUCTION REPORTED                            category
VACANT 6/30 OR DATE SOLD IF EARLIER              category
PRIMARY BUSINESS ACTIVITY                        category
PROPERTY NUMBER                                    object
PROPERTY STREET                                    object
UNIT                                               object
BOROUGH 1                                          object
POSTCODE                                           object
LATITUDE                                          float64
LONGITUDE                                         float64
COMMUNITY BOARD                                  category
COUNCIL DISTRICT                                 category
CENSUS TRACT                                       object
BIN                                               float64
BBL                                                 int64
NTA                                                object
NBHD                                             category
dtype: object
```

1.5 How many null values are in each column?

In [18]:
```
# display the number of missing values in each column of the DataFrame

storefronts.isna().sum()
```

Out[18]:
```
Reporting Year                                       0
BOROUGH-BLOCK-LOT                                    0
PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS        1
BOROUGH                                              0
ZIP CODE                                          1071
SOLD DATE                                       213898
VACANT ON 12/31                                      0
CONSTRUCTION REPORTED                           213080
VACANT 6/30 OR DATE SOLD IF EARLIER             205408
PRIMARY BUSINESS ACTIVITY                            0
PROPERTY NUMBER                                    591
PROPERTY STREET                                    184
UNIT                                            193139
BOROUGH 1                                            0
POSTCODE                                             0
LATITUDE                                          5965
LONGITUDE                                         5965
COMMUNITY BOARD                                    172
COUNCIL DISTRICT                                    47
CENSUS TRACT                                         1
BIN                                                983
BBL                                                  0
NTA                                                 83
NBHD                                                80
dtype: int64
```

1.6 How many unique values are there in each column?

In [19]:
```
# display the count of unique elements in each column

storefronts.nunique(axis=0, dropna=True)
```

Out[19]:
```
Reporting Year                                      3
BOROUGH-BLOCK-LOT                               45013
PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS    86036
BOROUGH                                             5
ZIP CODE                                          221
SOLD DATE                                         130
VACANT ON 12/31                                     2
CONSTRUCTION REPORTED                               1
VACANT 6/30 OR DATE SOLD IF EARLIER                 1
PRIMARY BUSINESS ACTIVITY                          18
PROPERTY NUMBER                                 11127
PROPERTY STREET                                  2708
UNIT                                              959
BOROUGH 1                                           5
POSTCODE                                          221
LATITUDE                                        42597
LONGITUDE                                       42543
COMMUNITY BOARD                                    19
COUNCIL DISTRICT                                   51
CENSUS TRACT                                     1078
BIN                                             44486
BBL                                             45013
NTA                                               193
NBHD                                              193
dtype: int64
```

---

2. Briefly summarize the contents of the dataset

2.1 Summarize the columns containing categorical variables

In [20]:
```
# summarize the data contained in columns with the 'category' data type only

storefronts.describe(include=['category'])
```

Out[20]:

	Reporting Year	PROPERTY STREET ADDRESS OR STOREFRONT ADDRESS	BOROUGH	VACANT ON 12/31	CONSTRUCTION REPORTED	VACANT 6/30 OR DATE SOLD IF EARLIER	PRIMARY BUSINESS ACTIVITY	COM
count	214246	214245	214246	214246	1166	8838	214246	2
unique	3	86036	5	2	1	1	18	
top	2020 and 2021	75 9 AVENUE	MANHATTAN	NO	YES	YES	RETAIL	
freq	75540	190	77566	192151	1166	8838	61170	

---

3. Select a subset of data for closer examination

3.1 Select a subset of columns

In [21]:
```
# display all reporting years

storefronts['Reporting Year'].unique()
```

Out[21]:
```
['2019 and 2020', '2021 and 2022', '2020 and 2021']
Categories (3, object): ['2019 and 2020', '2020 and 2021', '2021 and 2022']
```

In [22]:
```
# select a subset of columns to examine
selected_cols = ['Reporting Year',
                 'BOROUGH',
                 'VACANT ON 12/31',
                 'CONSTRUCTION REPORTED',
                 'VACANT 6/30 OR DATE SOLD IF EARLIER',
                 'PRIMARY BUSINESS ACTIVITY',
                 'COMMUNITY BOARD',
                 'COUNCIL DISTRICT',
                 'NBHD']

storefronts_focuscols = storefronts[selected_cols]
```

3.2 Display the shape of the data subset

In [23]:
```
rows = storefronts_focuscols.shape[0]
columns = storefronts_focuscols.shape[1]

print(f'There are {rows} rows and {columns} columns in the subset.')
```

```
There are 214246 rows and 9 columns in the subset.
```

---

4. Examine the individual variables in the dataset

4.1 What is the distribution of reporting years represented in the dataset?

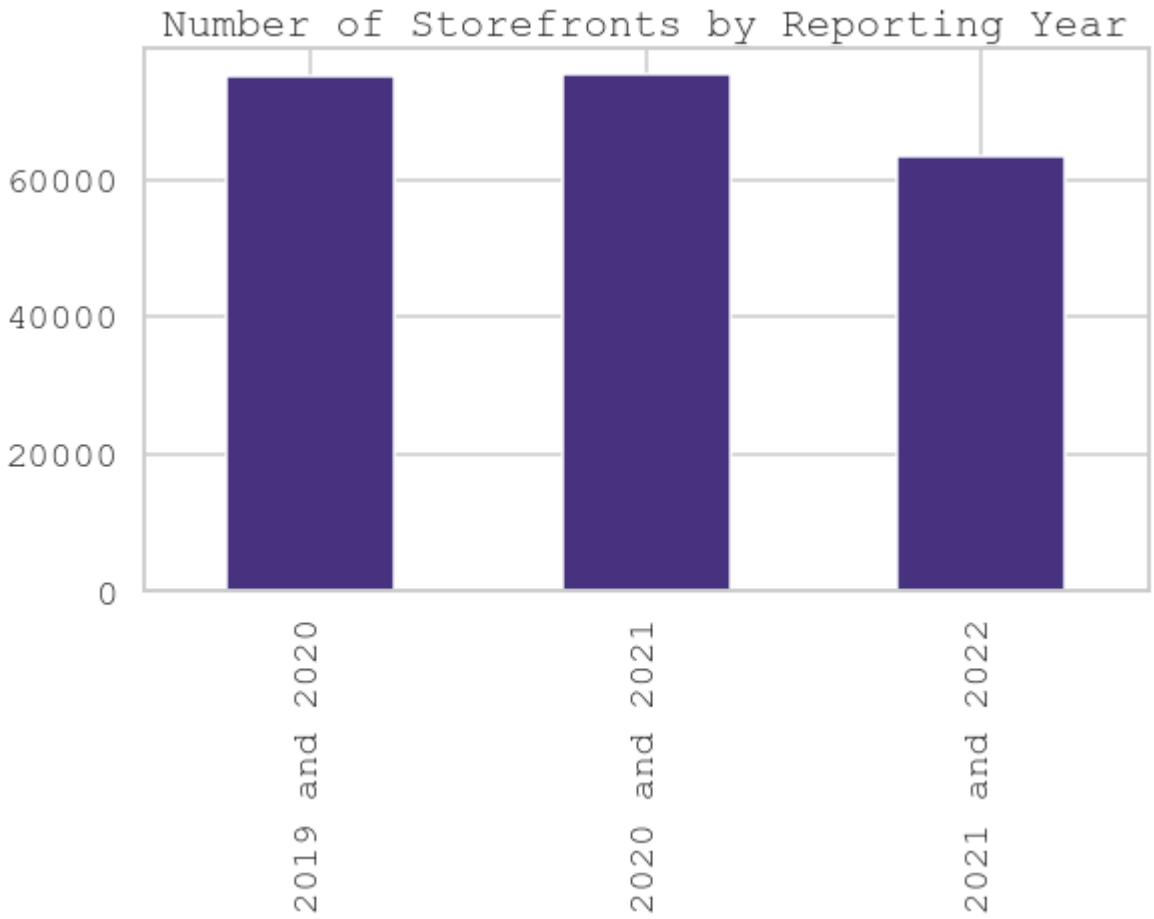
In [24]:
```
storefronts_focuscols['Reporting Year'].value_counts(sort=False)
```

Out[24]:
```
2019 and 2020    75250
2020 and 2021    75540
2021 and 2022    63456
Name: Reporting Year, dtype: int64
```

In [25]:
```
storefronts_focuscols['Reporting Year'].value_counts(sort=False, normalize=True)
```

Out[25]:
```
2019 and 2020   0.35
2020 and 2021   0.35
2021 and 2022   0.30
Name: Reporting Year, dtype: float64
```

In [26]:
```
storefronts_focuscols['Reporting Year'].value_counts(sort=False).plot(kind='bar',
                                                                      figsize=(9,5),
                                                                      title='Number of
```



4.2 What is the distribution of boroughs represented in the dataset?

In [27]:
```
storefronts_focuscols['BOROUGH'].value_counts()
```

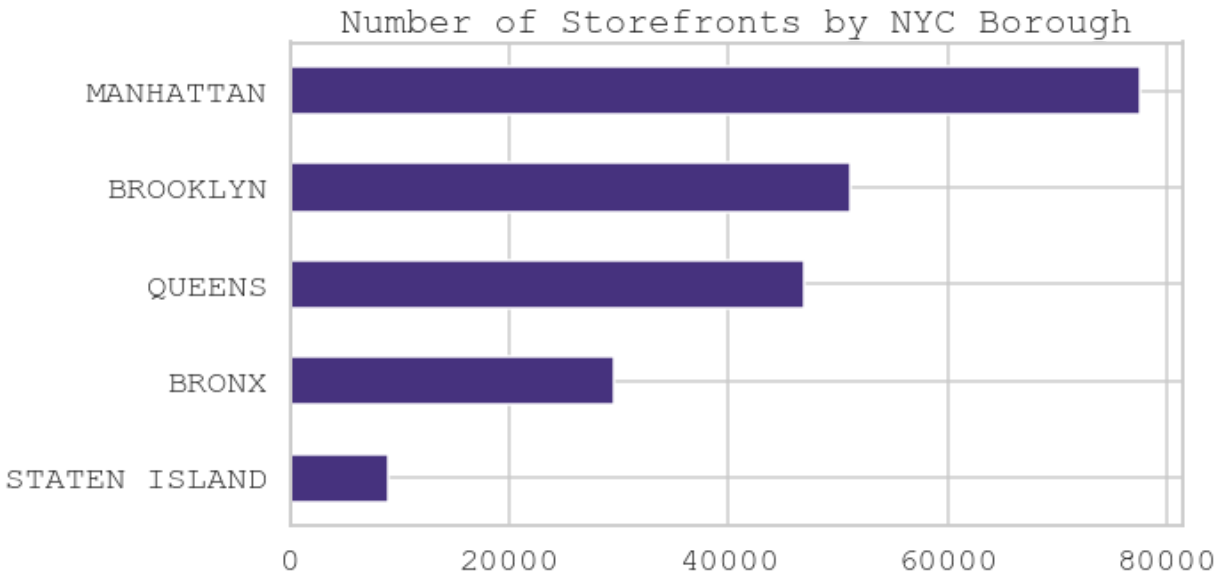
Out[27]:
```
MANHATTAN        77566
BROOKLYN         51211
QUEENS           46920
BRONX            29510
STATEN ISLAND     9039
Name: BOROUGH, dtype: int64
```

In [28]:
```
storefronts_focuscols['BOROUGH'].value_counts(normalize=True)
```

Out[28]:
```
MANHATTAN        0.36
BROOKLYN         0.24
QUEENS           0.22
BRONX            0.14
STATEN ISLAND    0.04
Name: BOROUGH, dtype: float64
```

In [29]:
```
storefronts_focuscols['BOROUGH'].value_counts().sort_values().plot(kind='barh',
                                                    figsize=(9,5),
                                                    title='Number of Storefronts by N
```

Out[29]:
```
<AxesSubplot:title={'center':'Number of Storefronts by NYC Borough'}>
```



4.3 What was the proportion of storefronts vacant as of 12/31 for each reporting year?

In [30]:
```
# for '2019 and 2020' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2019 and 2020'][
```

Out[30]:
```
NO     0.91
YES    0.09
Name: VACANT ON 12/31, dtype: float64
```

In [31]:
```
# for '2020 and 2021' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2020 and 2021'][
```

Out[31]:
```
NO     0.89
YES    0.11
Name: VACANT ON 12/31, dtype: float64
```

In [32]:
```
# for '2021 and 2022' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2021 and 2022'][
```

Out[32]:
```
NO    0.90
YES   0.10
Name: VACANT ON 12/31, dtype: float64
```

4.4 What was the proportion of storefronts undergoing construction as for each reporting year?

In [33]:
```
# for '2019 and 2020' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2019 and 2020'][
```

Out[33]:
```
NaN   0.99
YES   0.01
Name: CONSTRUCTION REPORTED, dtype: float64
```

In [34]:
```
# for '2020 and 2021' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2020 and 2021'][
```

Out[34]:
```
NaN   0.99
YES   0.01
Name: CONSTRUCTION REPORTED, dtype: float64
```

In [35]:
```
# for '2021 and 2022' reporting year

storefronts_focuscols.loc[storefronts_focuscols['Reporting Year'] == '2021 and 2022'][
```

Out[35]:
```
NaN   1.00
YES   0.00
Name: CONSTRUCTION REPORTED, dtype: float64
```

4.5 What is the distribution of primary business activities represented in the dataset?

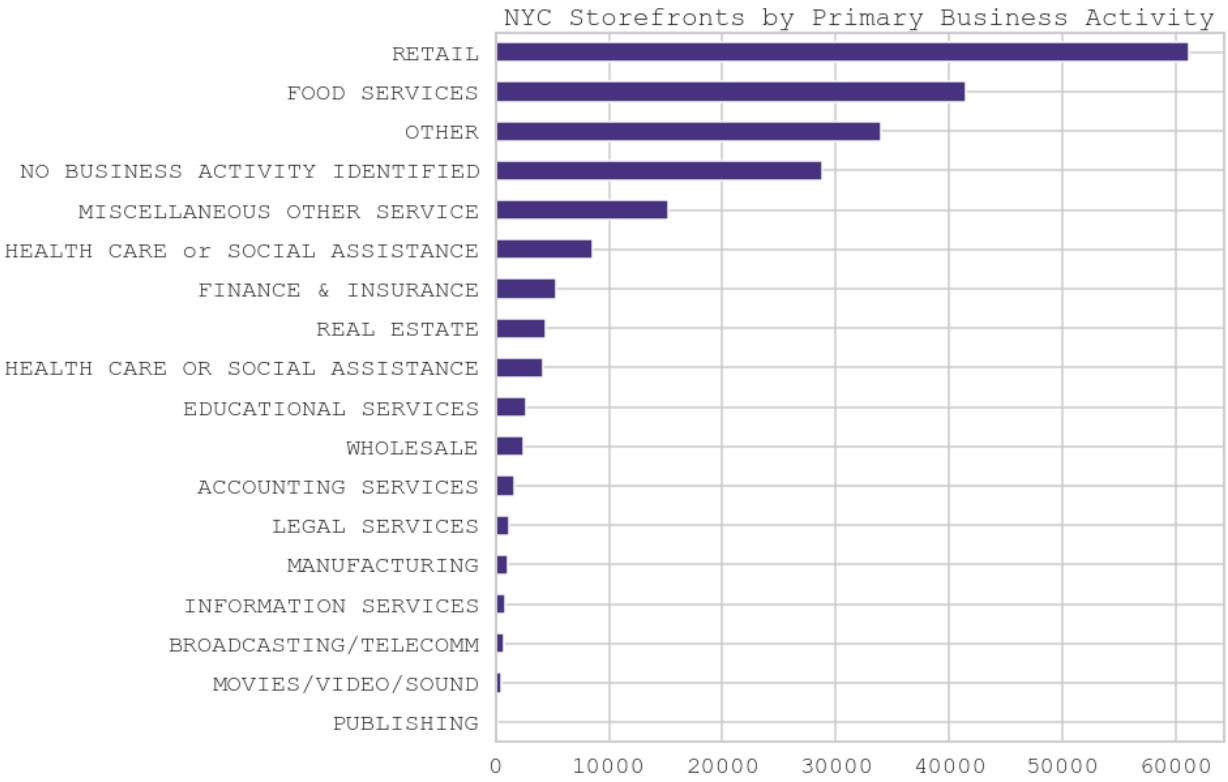
In [36]:
```
storefronts_focuscols[ 'PRIMARY BUSINESS ACTIVITY'].value_counts()
```

Out[36]:
```
RETAIL                                61170
FOOD SERVICES                         41421
OTHER                                 33950
NO BUSINESS ACTIVITY IDENTIFIED       28846
MISCELLANEOUS OTHER SERVICE           15256
HEALTH CARE or SOCIAL ASSISTANCE       8499
FINANCE & INSURANCE                    5295
REAL ESTATE                            4434
HEALTH CARE OR SOCIAL ASSISTANCE       4213
EDUCATIONAL SERVICES                   2649
WHOLESALE                              2441
ACCOUNTING SERVICES                    1646
LEGAL SERVICES                         1237
MANUFACTURING                          1114
INFORMATION SERVICES                    789
BROADCASTING/TELECOMM                   695
MOVIES/VIDEO/SOUND                      474
PUBLISHING                              117
Name: PRIMARY BUSINESS ACTIVITY, dtype: int64
```

In [38]:
```
storefronts_focuscols[ 'PRIMARY BUSINESS ACTIVITY'].value_counts()
```

Out[38]:
```
RETAIL                                61170
FOOD SERVICES                         41421
OTHER                                 33950
NO BUSINESS ACTIVITY IDENTIFIED       28846
MISCELLANEOUS OTHER SERVICE           15256
HEALTH CARE or SOCIAL ASSISTANCE       8499
FINANCE & INSURANCE                    5295
REAL ESTATE                            4434
HEALTH CARE OR SOCIAL ASSISTANCE       4213
EDUCATIONAL SERVICES                   2649
WHOLESALE                              2441
ACCOUNTING SERVICES                    1646
LEGAL SERVICES                         1237
MANUFACTURING                          1114
INFORMATION SERVICES                    789
BROADCASTING/TELECOMM                   695
MOVIES/VIDEO/SOUND                      474
PUBLISHING                              117
Name: PRIMARY BUSINESS ACTIVITY, dtype: int64
```

In [39]:
```
storefronts_focuscols[ 'PRIMARY BUSINESS ACTIVITY'].value_counts().sort_values().plot(
                                                                    figsize=(10,10
                                                                    title="NYC Sto
```



4.6 What is the distribution of neighborhoods represented in the dataset?

In [45]:
```
storefronts_focuscols['NBHD'].value_counts()
```

Out[45]:

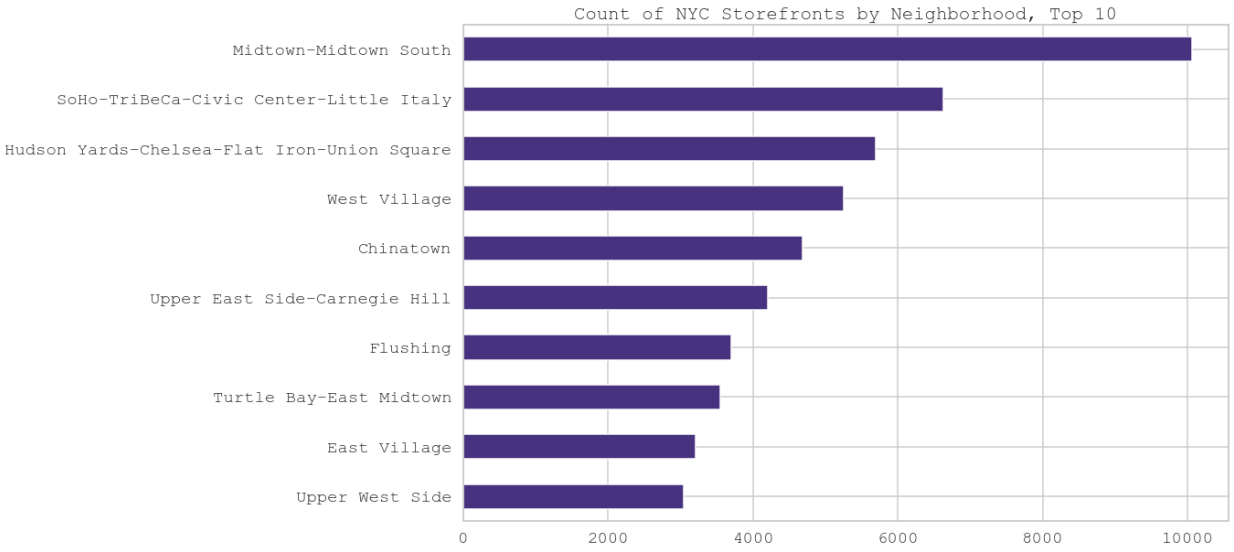
Midtown-Midtown South	10057
SoHo-TriBeCa-Civic Center-Little Italy	6625
Hudson Yards-Chelsea-Flat Iron-Union Square	5687
West Village	5249
Chinatown	4687
Upper East Side-Carnegie Hill	4204
Flushing	3698
Turtle Bay-East Midtown	3546
East Village	3204
Upper West Side	3043
Hunters Point-Sunnyside-West Maspeth	2836
Astoria	2801
Jackson Heights	2737
Clinton	2609
Lenox Hill-Roosevelt Island	2609
Washington Heights South	2553
North Side-South Side	2452
Park Slope-Gowanus	2396
Murray Hill-Kips Bay	2358
Battery Park City-Lower Manhattan	2311
DUMBO-Vinegar Hill-Downtown Brooklyn-Boerum Hill	2172
Washington Heights North	2086
Bay Ridge	2080
Sunset Park West	2072
Forest Hills	2026
Yorkville	2020
Sunset Park East	2001
Jamaica	1939
Bensonhurst West	1901
Borough Park	1892
Central Harlem North-Polo Grounds	1809
Elmhurst	1773
East Harlem North	1761
Ridgewood	1639
Crown Heights North	1637
Lincoln Square	1621
Bushwick North	1499
Mott Haven-Port Morris	1491
Sheepshead Bay-Gerritsen Beach-Manhattan Beach	1490
Marble Hill-Inwood	1486
Greenpoint	1476
Flatbush	1417
Williamsbridge-Olinville	1406
Gramercy	1405
Bayside-Bayside Hills	1396
Prospect Lefferts Gardens-Wingate	1381
Central Harlem South	1367
Murray Hill	1333
Lower East Side	1326
Woodside	1285
Melrose South-Mott Haven North	1279
East Harlem South	1250
Bushwick South	1250
Homecrest	1244
Mount Hope	1242
East Williamsburg	1239
North Corona	1229
Hamilton Heights	1225
West Concourse	1213
Spuyten Duyvil-Kingsbridge	1186

Steinway	1173
East Concourse-Concourse Village	1171
Bedford Park-Fordham North	1160
South Ozone Park	1146
Brownsville	1131
Bensonhurst East	1117
Flatlands	1109
Hunts Point	1085
Van Nest-Morris Park-Westchester Square	1063
Bedford	1006
Morningside Heights	998
East Tremont	997
Fordham South	994
East New York	980
Canarsie	971
Charleston-Richmond Valley-Tottenville	968
Carroll Gardens-Columbia Street-Red Hook	943
Co-op City	933
Belmont	931
New Dorp-Midland Beach	923
Richmond Hill	916
Norwood	908
Stuyvesant Heights	905
Madison	902
Rugby-Remsen Village	876
Queensbridge-Ravenswood-Long Island City	873
Brighton Beach	870
Great Kills	861
Pelham Parkway	844
University Heights-Morris Heights	817
Brooklyn Heights-Cobble Hill	804
Rego Park	798
Morrisania-Melrose	795
Woodhaven	792
East Flatbush-Farragut	788
Clinton Hill	767
Midwood	753
Bronxdale	750
Erasmus	731
Queens Village	716
West Farms-Bronx River	715
New Springville-Bloomfield-Travis	702
Whitestone	685
Westerleigh	679
Douglas Manor-Douglaston-Little Neck	677
Woodlawn-Wakefield	676
Maspeth	669
West New Brighton-New Brighton-St. George	664
Longwood	659
Westchester-Unionport	645
Jamaica Estates-Holliswood	639
Old Town-Dongan Hills-South Beach	629
Georgetown-Marine Park-Bergen Beach-Mill Basin	625
Fort Greene	611
Claremont-Bathgate	598
Crown Heights South	596
Cypress Hills-City Line	593
Corona	593
St. Albans	592
Old Astoria	591

Dyker Heights	588
Stapleton-Rosebank	583
Soundview-Bruckner	580
Schuylerville-Throgs Neck-Edgewater Park	577
College Point	565
Van Cortlandt Village	561
Far Rockaway-Bayswater	554
Middle Village	553
Highbridge	550
Fresh Meadows-Utopia	543
Parkchester	539
Briarwood-Jamaica Hills	525
Mariner's Harbor-Arlington-Port Ivory-Graniteville	524
Lindenwood-Howard Beach	522
Pelham Bay-Country Club-City Island	521
Eastchester-Edenwald-Baychester	512
Elmhurst-Maspeth	511
Glendale	502
Kensington-Ocean Parkway	489
Kew Gardens Hills	480
Crotona Park East	477
Soundview-Castle Hill-Clason Point-Harding Park	475
Pomonok-Flushing Heights-Hillcrest	473
Allerton-Pelham Gardens	456
Springfield Gardens South-Brookville	452
Bath Beach	439
Kew Gardens	424
Port Richmond	424
Ft. Totten-Bay Terrace-Clearview	418
Todt Hill-Emerson Hill-Heartland Village-Lighthouse Hill	417
Seagate-Coney Island	413
East Flushing	405
Kingsbridge Heights	402
Prospect Heights	395
Ozone Park	391
Manhattanville	390
Laurelton	374
Gravesend	372
Glen Oaks-Floral Park-New Hyde Park	364
Ocean Hill	363
Breezy Point-Belle Harbor-Rockaway Park-Broad Channel	363
Auburndale	358
New Brighton-Silver Lake	351
Annadale-Huguenot-Prince's Bay-Eltingville	347
Oakland Gardens	347
Williamsburg	328
Ocean Parkway South	303
North Riverdale-Fieldston-Riverdale	296
Bellerose	270
Baisley Park	269
Grasmere-Arrochar-Ft. Wadsworth	261
Hammels-Arverne-Edgemere	257
East New York (Pennsylvania Ave)	235
Windsor Terrace	224
Cambria Heights	212
East Elmhurst	206
Rosedale	204
West Brighton	201
Todt Hill-Emerson Hill-Heartland Village-Lighthouse H	195
South Jamaica	181

```
Rossville-Woodrow                                179
Hollis                                           177
Queensboro Hill                                  175
Oakwood-Oakwood Beach                            168
Springfield Gardens North                        149
Grymes Hill-Clifton-Fox Hills                    119
park-cemetery-etc-Queens                         109
park-cemetery-etc-Brooklyn                       104
Stuyvesant Town-Cooper Village                    72
Starrett City                                     50
Arden Heights                                     41
park-cemetery-etc-Bronx                            2
park-cemetery-etc-Manhattan                        1
Name: NBHD, dtype: int64
```

In [40]:
```
storefronts_focuscols['NBHD'].value_counts().nlargest(10).sort_values().plot(kind='bar
                                                                              figsize=(15,10
                                                                              title="Count o
```



Next steps

Export data for data graphic creation

In [49]:
```
# subset storefronts by neighborhood data for most recent reporting year and non-vacan

open_storefronts_21_22 = storefronts_focuscols.loc[(storefronts_focuscols['Reporting Y
                                                    (storefronts_focuscols['VACANT ON 1
```

In [50]:
```
# confirm that the data has been subset as expected

open_storefronts_21_22.sample(20, random_state=1)
```

Out[50]:

	Reporting Year	BOROUGH	VACANT ON 12/31	CONSTRUCTION REPORTED	VACANT 6/30 OR DATE SOLD IF EARLIER	PRIMARY BUSINESS ACTIVITY	COMMUNITY BOARD
178031	2021 and 2022	BRONX	NO	NaN	NaN	RETAIL	5.00
213054	2021 and 2022	STATEN ISLAND	NO	NaN	NaN	RETAIL	2.00
177900	2021 and 2022	BRONX	NO	NaN	NaN	RETAIL	3.00
198129	2021 and 2022	BROOKLYN	NO	NaN	NaN	RETAIL	13.00
153796	2021 and 2022	MANHATTAN	NO	NaN	NaN	FOOD SERVICES	3.00
204645	2021 and 2022	QUEENS	NO	NaN	NaN	FOOD SERVICES	5.00
191277	2021 and 2022	BROOKLYN	NO	NaN	NaN	FOOD SERVICES	5.00
168144	2021 and 2022	MANHATTAN	NO	NaN	NaN	INFORMATION SERVICES	8.00
154412	2021 and 2022	MANHATTAN	NO	NaN	NaN	FOOD SERVICES	3.00
185393	2021 and 2022	BROOKLYN	NO	NaN	NaN	RETAIL	7.00
207868	2021 and 2022	QUEENS	NO	NaN	NaN	RETAIL	8.00
166526	2021 and 2022	MANHATTAN	NO	NaN	NaN	OTHER	5.00
178100	2021 and 2022	BRONX	NO	NaN	NaN	FOOD SERVICES	6.00
197582	2021 and 2022	BROOKLYN	NO	NaN	NaN	HEALTH CARE or SOCIAL ASSISTANCE	18.00
213024	2021 and 2022	STATEN ISLAND	NO	NaN	NaN	RETAIL	2.00
207995	2021 and 2022	QUEENS	NO	NaN	NaN	RETAIL	8.00
172919	2021 and	MANHATTAN	NO	NaN	NaN	RETAIL	9.00

	Reporting Year	BOROUGH	VACANT ON 12/31	CONSTRUCTION REPORTED	VACANT 6/30 OR DATE SOLD IF EARLIER	PRIMARY BUSINESS ACTIVITY	COMMUNITY BOARD
	2022						
163659	2021 and 2022	MANHATTAN	NO	NaN	NaN	FOOD SERVICES	4.00
176318	2021 and 2022	BRONX	NO	NaN	NaN	FOOD SERVICES	1.00
160134	2021 and 2022	MANHATTAN	NO	NaN	NaN	FOOD SERVICES	5.00

In [56]:
```
# confirm reporting year filtering

open_storefronts_21_22['Reporting Year'].nunique()
```

Out[56]:
```
1
```

In [54]:
```
open_storefronts_21_22['Reporting Year'].unique()
```

Out[54]:
```
['2021 and 2022']
Categories (3, object): ['2019 and 2020', '2020 and 2021', '2021 and 2022']
```

In [59]:
```
# confirm filtering on 'vacant or not'

open_storefronts_21_22['VACANT ON 12/31'].nunique()
```

Out[59]:
```
1
```

In [58]:
```
open_storefronts_21_22['VACANT ON 12/31'].unique()
```

Out[58]:
```
['NO']
Categories (2, object): ['NO', 'YES']
```

In [63]:
```
# find the ccount of open storefronts by neighborhood for '2021 and 2022' reporting ye

top10_open_storefronts_21_22 = open_storefronts_21_22['NBHD'].value_counts().nlargest(
```

In [64]:
```
top10_open_storefronts_21_22
```

Out[64]:
```
Midtown-Midtown South                          2505
SoHo-TriBeCa-Civic Center-Little Italy         1779
Hudson Yards-Chelsea-Flat Iron-Union Square    1471
West Village                                   1422
Chinatown                                      1205
Upper East Side-Carnegie Hill                  1177
Flushing                                       1016
Turtle Bay-East Midtown                         883
Upper West Side                                 871
East Village                                    861
Name: NBHD, dtype: int64
```

In [65]:
```
# export filtered data to csv

top10_open_storefronts_21_22.to_csv('top10_open_storefronts_21_22.csv')
```