NYC Property Sales Project

Kuntal Bhar

INTRODUCTIONS

As we know New York City real state is among the most expensive cities in the world.We uses NYC Property Sales Dataset provided by the New York City Department of Finance to do our analysis.

This dataset have record of every building or building unit (apartment, etc.) sold in the New York City property market over a 12-month period from September 1, 2016 to August 31, 2017. We try discover about New York City real estate by looking at a year's worth of raw transaction records and spot trends in the market, or build a model that predicts sale value in the future.

Objective

We do analysis on 1.Data analysis that includes data import ,analysing the data and its variable like Borough, Neighborhood, Age of the building/property, Size of property and type of building), cleaning of data if needed, data preview etc. 2.Descriptive, Visualization and Predictive Analysis various categories results like Most In-Demand Borough, Most In-Demand Neighborhood, Preferred Buildings, Property sizes, Age of the buildings and spot trends in the market.

```
## package 'plotly' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\kunta\AppData\Local\Temp\Rtmpov5ER9\downloaded_packages

## package 'formattable' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\kunta\AppData\Local\Temp\Rtmpov5ER9\downloaded_packages
```

DATASET ANALYSIS

We explore the data provided by New York City Department of Finance. We do various steps here for preparing the data. Steps include data import, data cleaning, data preview, data details.

The NYC Property Sales Dataset is a record of every building or apartment unit that was sold in the NYC Property market over a 12 month period of 5 boroughs in NYC - Manhattan, the Bronx, Queens, Brookyln and Staten Island. For furthe detail please sheck the refrence section at the bottom

Here we prepare the data. Each sation section contains the logic and the steps performed in bringing the data to a form suitable for statistical analysis. Each section explains the related steps for importing and cleaning the data.

The dataset was downloaded nyc-rolling-sales.csv from Kaggle www.kaggle.com/new-york-city/nyc-property-sales

NOTE:We can downloaded the file nyc-rolling-sales.csv dataset was manualy from Kagglewww.kaggle.com/new-york-city/nyc-property-sales and save it in save the file nyc-rolling-sales.csv in same same folder were we are R and RMD File You can also download from git hub site https://github.com/kuntalbhar/edx-nyc-project where i have uploaded the csv file nyc-rolling-sales.csv

Data Import

We import downloaded NYC Property Sales downloaded csv file from site (file-> nyc-rolling-sales.csv)

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
#dimension of the read file
dim(nyc_orig_data)
```

```
## [1] 84548    22
```

NYC Property Sales Data has 84548 observations and 22 variables.

Column names in the file

```
#show column name present in the file
names(nyc_orig_data)
```

```
##  [1] "V1"                              "BOROUGH"
##  [3] "NEIGHBORHOOD"                    "BUILDING CLASS CATEGORY"
##  [5] "TAX CLASS AT PRESENT"            "BLOCK"
##  [7] "LOT"                             "EASE-MENT"
##  [9] "BUILDING CLASS AT PRESENT"       "ADDRESS"
## [11] "APARTMENT NUMBER"                "ZIP CODE"
## [13] "RESIDENTIAL UNITS"               "COMMERCIAL UNITS"
## [15] "TOTAL UNITS"                     "LAND SQUARE FEET"
## [17] "GROSS SQUARE FEET"               "YEAR BUILT"
## [19] "TAX CLASS AT TIME OF SALE"       "BUILDING CLASS AT TIME OF SALE"
## [21] "SALE PRICE"                      "SALE DATE"
```

Data Cleaning

First column “V1” has observation number which we can ignore. Column “BUILDING CLASS CATEGORY” has category number and title together which will split into 2 column “BUILDING CLASS CATEGORY NUMBER” and “BUILDING CLASS CATEGORY” Removed “EASE-MENT” column that has only blank. ‘SALE DATE’ also has date and time together we remove the time part. Add a another column “BUILDING AGE” which provde the age of the building which we will use later for calculation. This column will have age caluclated based on latest year.

```
#Removed the first column 'V1'
nyc_curr_data<- as_data_frame(nyc_orig_data[,-1])

#Separate the columns in 'BUILDING CLASS CATEGORY'
nyc_curr_data <- nyc_curr_data %>%
  separate(col = "BUILDING CLASS CATEGORY", into = c("BUILDING CLASS CATEGORY NUMBER",
                                                     "BUILDING CLASS CATEGORY"), sep = 3) %>%
```

```
  separate(col = "SALE DATE", into = c("SALE DATE", "TIME"), sep = " ")

#Removed columns - 'EASE-MENT', 'TIME' (column #8,23)
nyc_curr_data <- nyc_curr_data[,c(-8,-23)]

#adding column BUILDING AGE
nyc_curr_data <- nyc_curr_data %>%
  mutate(`BUILDING AGE` = 2017 - `YEAR BUILT`)

#display column
names(nyc_curr_data)
```

```
##  [1] "BOROUGH"                         "NEIGHBORHOOD"
##  [3] "BUILDING CLASS CATEGORY NUMBER" "BUILDING CLASS CATEGORY"
##  [5] "TAX CLASS AT PRESENT"            "BLOCK"
##  [7] "LOT"                             "BUILDING CLASS AT PRESENT"
##  [9] "ADDRESS"                         "APARTMENT NUMBER"
## [11] "ZIP CODE"                        "RESIDENTIAL UNITS"
## [13] "COMMERCIAL UNITS"                "TOTAL UNITS"
## [15] "LAND SQUARE FEET"                "GROSS SQUARE FEET"
## [17] "YEAR BUILT"                      "TAX CLASS AT TIME OF SALE"
## [19] "BUILDING CLASS AT TIME OF SALE" "SALE PRICE"
## [21] "SALE DATE"                       "BUILDING AGE"
```

For further dataset refrence nyc_curr_data as current dataset and nyc_orig_data as original dataset..

```
#Removing the Duplicates in the data frame, nyc_property
nyc_curr_data %>% filter(duplicated(nyc_curr_data) == TRUE) %>% nrow()
```

```
## [1] 765
```

Checked and remove the 765 duplicates entries in the data if any

```
#Removing the Duplicates in the data frame, nyc_property
nyc_curr_data <- unique(nyc_curr_data)
dim(nyc_curr_data)
```

```
## [1] 83783    22
```

Dataset contains 83783 observations and 22 variables.

Column Type Conversion

Column types imported are “int” and “chr” whic is not helpful for our analysis. So we change the column data types to it appropriate types based on data.

Current structure is shown below

```
  str(nyc_curr_data)
```

```
## tibble [83,783 x 22] (S3: tbl_df/tbl/data.frame)
##  $ BOROUGH                       : int [1:83783] 1 1 1 1 1 1 1 1 1 1 ...
##  $ NEIGHBORHOOD                  : chr [1:83783] "ALPHABET CITY" "ALPHABET CITY" "ALPHABET CITY" "ALI
##  $ BUILDING CLASS CATEGORY NUMBER: chr [1:83783] "07 " "07 " "07 " "07 " ...
##  $ BUILDING CLASS CATEGORY       : chr [1:83783] "RENTALS - WALKUP APARTMENTS" "RENTALS - WALKUP APAI
##  $ TAX CLASS AT PRESENT          : chr [1:83783] "2A" "2" "2" "2B" ...
##  $ BLOCK                         : int [1:83783] 392 399 399 402 404 405 406 407 379 387 ...
##  $ LOT                           : int [1:83783] 6 26 39 21 55 16 32 18 34 153 ...
##  $ BUILDING CLASS AT PRESENT     : chr [1:83783] "C2" "C7" "C7" "C4" ...
##  $ ADDRESS                       : chr [1:83783] "153 AVENUE B" "234 EAST 4TH   STREET" "197 EAST 3RI
##  $ APARTMENT NUMBER              : chr [1:83783] "" "" "" "" ...
##  $ ZIP CODE                      : int [1:83783] 10009 10009 10009 10009 10009 10009 10009 10009 100
##  $ RESIDENTIAL UNITS             : int [1:83783] 5 28 16 10 6 20 8 44 15 24 ...
##  $ COMMERCIAL UNITS              : int [1:83783] 0 3 1 0 0 0 0 2 0 0 ...
##  $ TOTAL UNITS                   : int [1:83783] 5 31 17 10 6 20 8 46 15 24 ...
##  $ LAND SQUARE FEET              : chr [1:83783] "1633" "4616" "2212" "2272" ...
##  $ GROSS SQUARE FEET             : chr [1:83783] "6440" "18690" "7803" "6794" ...
##  $ YEAR BUILT                    : int [1:83783] 1900 1900 1900 1913 1900 1900 1920 1900 1920 1920 .
##  $ TAX CLASS AT TIME OF SALE     : int [1:83783] 2 2 2 2 2 2 2 2 2 2 ...
##  $ BUILDING CLASS AT TIME OF SALE: chr [1:83783] "C2" "C7" "C7" "C4" ...
##  $ SALE PRICE                    : chr [1:83783] "6625000" "-" "-" "3936272" ...
##  $ SALE DATE                     : chr [1:83783] "2017-07-19" "2016-12-14" "2016-12-09" "2016-09-23"
##  $ BUILDING AGE                  : num [1:83783] 117 117 117 104 117 117 97 117 97 97 ...
```

The "BOROUGH" are converted into Factor and the Id's (1-5) are change to name as mentiond in the project detail content. BUILDING CLASS CATEGORY,TAX CLASS AT PRESENT, TAX CLASS AT TIME OF SALE, were converted into Factor's. BLOCK, LOT, ADDRESS, APARTMENT NUMBER, ZIP CODE were made Char data types. LAND SQUARE FEET, GROSS SQUARE FEET, YEAR BUILT, SALE PRICE were coverted into Numeric data types. SALE DATE was converted into a Date and format (mdy).

```
#Data Type conversions to the Data set

#factor conversion
colPos <- c(1,3,4,5,8,11,18,19)
nyc_curr_data %<>% mutate_at(colPos, funs(factor(.)))

#change id's to name of BOROUGH column
levels(nyc_curr_data$BOROUGH) <- c("Manhattan", "Bronx", "Brooklyn", "Queens", "Staten Island")

#numeric conversion
num <- c(15,16,17,20)
nyc_curr_data %<>% mutate_at(num, funs(as.numeric(.)))

#char conversion
chr <- c(6,7)
nyc_curr_data %<>% mutate_at(chr, funs(as.character(.)))

#date conversion
nyc_curr_data$`SALE DATE` <- ymd(nyc_curr_data$`SALE DATE`)

#display modified structure
str(nyc_curr_data)
```

```
## tibble [83,783 x 22] (S3: tbl_df/tbl/data.frame)
##  $ BOROUGH                       : Factor w/ 5 levels "Manhattan","Bronx",..: 1 1 1 1 1 1 1 1 1 1 ..
##  $ NEIGHBORHOOD                  : chr [1:83783] "ALPHABET CITY" "ALPHABET CITY" "ALPHABET CITY" "ALP
##  $ BUILDING CLASS CATEGORY NUMBER: Factor w/ 47 levels "01 ","02 ","03 ",..: 7 7 7 7 7 7 7 7 8 8 ...
##  $ BUILDING CLASS CATEGORY       : Factor w/ 47 levels " CONDO-RENTALS",..: 34 34 34 34 34 34 34 34 3
##  $ TAX CLASS AT PRESENT          : Factor w/ 11 levels "","1","1A","1B",..: 7 6 6 8 7 6 8 6 6 6 ...
##  $ BLOCK                         : chr [1:83783] "392" "399" "399" "402" ...
##  $ LOT                           : chr [1:83783] "6" "26" "39" "21" ...
##  $ BUILDING CLASS AT PRESENT     : Factor w/ 167 levels "","A0","A1","A2",..: 17 22 22 19 17 19 19 21
##  $ ADDRESS                       : chr [1:83783] "153 AVENUE B" "234 EAST 4TH   STREET" "197 EAST 3RD
##  $ APARTMENT NUMBER              : chr [1:83783] "" "" "" "" ...
##  $ ZIP CODE                      : Factor w/ 186 levels "0","10001","10002",..: 9 9 9 9 9 9 9 9 9 9
##  $ RESIDENTIAL UNITS             : int [1:83783] 5 28 16 10 6 20 8 44 15 24 ...
##  $ COMMERCIAL UNITS              : int [1:83783] 0 3 1 0 0 0 0 2 0 0 ...
##  $ TOTAL UNITS                   : int [1:83783] 5 31 17 10 6 20 8 46 15 24 ...
##  $ LAND SQUARE FEET              : num [1:83783] 1633 4616 2212 2272 2369 ...
##  $ GROSS SQUARE FEET             : num [1:83783] 6440 18690 7803 6794 4615 ...
##  $ YEAR BUILT                    : num [1:83783] 1900 1900 1900 1913 1900 ...
##  $ TAX CLASS AT TIME OF SALE     : Factor w/ 4 levels "1","2","3","4": 2 2 2 2 2 2 2 2 2 2 ...
##  $ BUILDING CLASS AT TIME OF SALE: Factor w/ 166 levels "A0","A1","A2",..: 16 21 21 18 16 18 18 21 3
##  $ SALE PRICE                    : num [1:83783] 6625000 NA NA 3936272 8000000 ...
##  $ SALE DATE                     : Date[1:83783], format: "2017-07-19" "2016-12-14" ...
##  $ BUILDING AGE                  : num [1:83783] 117 117 117 104 117 117 97 117 97 97 ...
```

Zeros And Missing Values

Because of column data types conversation it may have introduce NA on values "-"

```
#number observations column  introduce NA's on data types conversation
sum(is.na(nyc_curr_data))
```

```
## [1] 67615
```

```
#List of  introduce NA's on data types conversation
colSums(is.na(nyc_curr_data))
```

```
##                        BOROUGH                   NEIGHBORHOOD
##                              0                              0
## BUILDING CLASS CATEGORY NUMBER        BUILDING CLASS CATEGORY
##                              0                              0
##           TAX CLASS AT PRESENT                          BLOCK
##                              0                              0
##                            LOT      BUILDING CLASS AT PRESENT
##                              0                              0
##                        ADDRESS               APARTMENT NUMBER
##                              0                              0
##                       ZIP CODE             RESIDENTIAL UNITS
##                              0                              0
##               COMMERCIAL UNITS                   TOTAL UNITS
##                              0                              0
##               LAND SQUARE FEET             GROSS SQUARE FEET
##                          26054                          27385
```

```
##                       YEAR BUILT      TAX CLASS AT TIME OF SALE
##                                0                             0
## BUILDING CLASS AT TIME OF SALE                    SALE PRICE
##                                0                         14176
##                        SALE DATE                  BUILDING AGE
##                                0                             0
```

From above we found that 3 column has NA's. After further analysing, we found that these column lost zero values due to data type convesion to numeric.

For example we compare "SALE PRICE" values in original data to modified data

```
#compare it with SALE PRICE original data on - and 0's
sum(nyc_orig_data$`SALE PRICE` == "-")
```

```
## [1] 14561
```

```
sum(is.na(nyc_curr_data$`SALE PRICE`))
```

```
## [1] 14176
```

```
sum(nyc_orig_data$`SALE PRICE` == 0)
```

```
## [1] 10228
```

```
nyc_curr_data %>% filter(`SALE PRICE` == 0) %>% nrow()
```

```
## [1] 10012
```

Looking at the results of column SALE PRICE, count in the original dataset had 14561 observations for '-' values, After converting the data type, the current dataset has 14176 NAs. The current dataset has less counts than the original. We may suspect it may be because of blanks.

Similarly results of column SALE PRICE shows in 0's case, count in the original dataset had 10228 observations for '-' values, After converting the data type, the current dataset has 10012 NAs. This conclude we have not lost any zeros or missing values during type conversion. Note that this data set has both NA values and 0 values. Simmilary the same thing happen for all the numuric column which LAND SQUARE FEET and GROSS SQUARE FEET.

Lets look at columns SALE PRICE,LAND SQUARE FEET and GROSS SQUARE FEET
SALE PRICE

Since this column contains observations 10012 that are 0 values and 14176 that are missing values we will analyze it seperately. We create 2 dataset defect dataset nyc_defect_data that will have missing and 0's and and nyc_clean_data without them.

Dataset with zero's and missing values contains

```
# Missing & Zero values in 'Sale Price' - 28.86% of the original dataset
nyc_defect_data <- nyc_curr_data %>% filter(nyc_curr_data$`SALE PRICE` == 0 | is.na(nyc_curr_data$`SALE
nyc_defect_data  %>% nrow()
```

```
## [1] 24188
```

Dataset without zero's and missing values contains

```
# Base Dataset - (Sale Price = 0 or NULL) - 71.13% of original dataset. nyc_clean_data dataset will be
nyc_clean_data <- nyc_curr_data %>% filter((!nyc_curr_data$`SALE PRICE` == 0) & !is.na(nyc_curr_data$`S
nyc_clean_data %>% nrow()
```

```
## [1] 59595
```

This clean data will be use for doing the analysis requiring Sale price values.

LAND SQUARE FEET and GROSS SQUARE FEET Will evalute 0's and missing values on these column. one assumprtion is that 0 Sq feet could mean the property sold as land only.

```
#get the count of zerow's observation on column LAND SQUARE FEET and  GROSS SQUARE FEET
nyc_clean_data %>% filter(nyc_clean_data$`LAND SQUARE FEET` == 0 | nyc_clean_data$`GROSS SQUARE FEET` ==
```

```
## [1] 8674
```

There are 8674 rows in clean dataset where Land Sq Footage or Gross Sq Footage is 0.

Lets check the summary of BUILDING CLASS CATEGORY where these column are 0's.

```
#get the summary of BUILDING CLASS CATEGORY with zero's observation on LAND SQUARE FEET and GROSS SQUAR
sqf_data <- nyc_clean_data %>% filter(nyc_clean_data$`LAND SQUARE FEET` == 0 | nyc_clean_data$`GROSS SQ
summary(sqf_data$`BUILDING CLASS CATEGORY`)
```

```
##                          CONDO-RENTALS                     ASYLUMS AND HOMES
##                                      9                                     0
##                     COMMERCIAL CONDOS                   COMMERCIAL GARAGES
##                                      3                                    80
##                 COMMERCIAL VACANT LAND                          CONDO COOPS
##                                     89                                    32
##  CONDO CULTURAL/MEDICAL/EDUCATIONAL/ETC                          CONDO HOTELS
##                                      2                                     2
##              CONDO NON-BUSINESS STORAGE                 CONDO OFFICE BUILDINGS
##                                     29                                    33
##                          CONDO PARKING                  CONDO STORE BUILDINGS
##                                    185                                    23
##          CONDO TERRACES/GARDENS/CABANAS          CONDO WAREHOUSES/FACTORY/INDUS
##                                     11                                     9
##          CONDOS - 2-10 UNIT RESIDENTIAL CONDOS - 2-10 UNIT WITH COMMERCIAL UNIT
##                                    785                                    15
##            CONDOS - ELEVATOR APARTMENTS              CONDOS - WALKUP APARTMENTS
##                                   2732                                   379
##             COOPS - ELEVATOR APARTMENTS               COOPS - WALKUP APARTMENTS
##                                   2832                                   545
##                 EDUCATIONAL FACILITIES                             FACTORIES
##                                      0                                     1
##          HOSPITAL AND HEALTH FACILITIES   INDOOR PUBLIC AND CULTURAL FACILITIES
##                                      0                                     0
```

```
##                          LOFT BUILDINGS                          LUXURY HOTELS
##                                       0                                      0
##                       OFFICE BUILDINGS                  ONE FAMILY DWELLINGS
##                                       0                                      9
##                           OTHER HOTELS          OUTDOOR RECREATIONAL FACILITIES
##                                       0                                      1
##                     RELIGIOUS FACILITIES                     RENTALS - 4-10 UNIT
##                                       0                                      1
##            RENTALS - ELEVATOR APARTMENTS             RENTALS - WALKUP APARTMENTS
##                                       1                                      1
##         SELECTED GOVERNMENTAL FACILITIES              SPECIAL CONDO BILLING LOTS
##                                       0                                      0
##                          STORE BUILDINGS                    TAX CLASS 1 - OTHER
##                                       0                                     22
##                       TAX CLASS 1 CONDOS                TAX CLASS 1 VACANT LAND
##                                     557                                    241
##        TAX CLASS 3 - UNTILITY PROPERTIES                    TAX CLASS 4 - OTHER
##                                       0                                     28
##                                THEATRES                 THREE FAMILY DWELLINGS
##                                       0                                      8
##                TRANSPORTATION FACILITIES                   TWO FAMILY DWELLINGS
##                                       1                                      8
##                              WAREHOUSES
##                                       0
```

From above we gather observations where LAND SQUARE FEET or GROSS SQUARE FEET are 0's, we can see that these properties are of different categories. Hence, the assumption of 0 sq footage being land only is wrong. Hence, 0 and missing values are equivalent in this dataset.

Number of observations of zero and NA in LAND SQUARE FEET

```
#get the count of zero and NA observations on column LAND SQUARE FEET
nyc_clean_data %>% filter(nyc_clean_data$`LAND SQUARE FEET` == 0 | is.na(nyc_clean_data$`LAND SQUARE FE
```

```
## [1] 29343
```

Number of observations of zero and NA in GROSS SQUARE FEET

```
#get the count of zero and NA observations on column GROSS SQUARE FEET
nyc_clean_data %>% filter(nyc_clean_data$`GROSS SQUARE FEET` == 0 | is.na(nyc_clean_data$`GROSS SQUARE
```

```
## [1] 30380
```

As these are equivalent, we convert the zero to NA for column LAND SQUARE FEET and GROSS SQUARE FEET and applying this on current and clean dataset.

```
nyc_data <- nyc_clean_data

# converting zero to NA for column LAND SQUARE FEET
nyc_data$`LAND SQUARE FEET`[nyc_data$`LAND SQUARE FEET` == 0] <- NA

# converting zero to NA for column LAND SQUARE FEET
nyc_data$`GROSS SQUARE FEET`[nyc_data$`GROSS SQUARE FEET` == 0] <- NA
```

Lets look at columns RESIDENTIAL UNITS, COMMERCIAL UNITS and TOTAL UNITS
Though we dont have missing values in these columns, there are 18634, 56819, 16589 zero values individually.

```
#get the count of zero observations on column GRESIDENTIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`RESIDENTIAL UNITS` == 0) %>% nrow()
```

```
## [1] 18634
```

```
#get the count of zero observations on column COMMERCIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`COMMERCIAL UNITS` == 0) %>% nrow()
```

```
## [1] 56819
```

```
#get the count of zero observations on column GRESIDENTIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`TOTAL UNITS` == 0) %>% nrow()
```

```
## [1] 16589
```

We will analyse this seperately as we assume that the TOTAL UNITS = RESIDENTIAL UNITS + COMMERCIAL UNITS.As expected, for 58801 rows that is 98% of the clean dataset it holds.

```
nyc_clean_data %>% filter(nyc_clean_data$`TOTAL UNITS` != nyc_clean_data$`RESIDENTIAL UNITS` +
                            nyc_clean_data$`COMMERCIAL UNITS`) %>% nrow()
```

```
## [1] 794
```

For 794 rows where it doesn't seem to match, TOTAL UNITS = 1 in a majority of the observations and it might be a subtitute for these columns.

Lets now look at Categorical Variables Lets look at CATEGORAL columns

```
#get the count of zero observations on column GRESIDENTIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`YEAR BUILT` == 0) %>% nrow()
```

```
## [1] 4296
```

```
#get the count of zero observations on column COMMERCIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`COMMERCIAL UNITS` == 0) %>% nrow()
```

```
## [1] 56819
```

```
#get the count of zero observations on column GRESIDENTIAL UNITS
nyc_clean_data %>% filter(nyc_clean_data$`TOTAL UNITS` == 0) %>% nrow()
```

```
## [1] 16589
```

Below table shows the category columns with number of levels (include blanks or 0) and number of missing or zero observations. We also view categories of the each columns

Column	Level	MissingOrZero
BOROUGH	5	0
TAX CLASS AT PRESENT	11	593
BUILDING CLASS CATEGORY NUMBER	47	0
BUILDING CLASS AT PRESENT	167	593
BUILDING CLASS AT TIME OF SALE	166	0
ZIP CODE	186	655
YEAR BUILT	153	4296

```
## [1] "TAX CLASS AT PRESENT"

##  [1] ""   "1"  "1A" "1B" "1C" "2"  "2A" "2B" "2C" "3"  "4"

## [1] "BUILDING CLASS CATEGORY NUMBER"

##  [1] "01 " "02 " "03 " "04 " "05 " "06 " "07 " "08 " "09 " "10 " "11 " "11A"
## [13] "12 " "13 " "14 " "15 " "16 " "17 " "18 " "21 " "22 " "23 " "25 " "26 "
## [25] "27 " "28 " "29 " "30 " "31 " "32 " "33 " "34 " "35 " "36 " "37 " "38 "
## [37] "39 " "40 " "41 " "42 " "43 " "44 " "45 " "46 " "47 " "48 " "49 "

## [1] "BUILDING CLASS AT PRESENT"

##   [1] ""   "A0" "A1" "A2" "A3" "A4" "A5" "A6" "A7" "A9" "B1" "B2" "B3" "B9" "C0"
##  [16] "C1" "C2" "C3" "C4" "C5" "C6" "C7" "C8" "C9" "CM" "D0" "D1" "D2" "D3" "D4"
##  [31] "D5" "D6" "D7" "D8" "D9" "E1" "E2" "E7" "E9" "F1" "F2" "F4" "F5" "F9" "G0"
##  [46] "G1" "G2" "G3" "G4" "G5" "G6" "G7" "G8" "G9" "GU" "GW" "H1" "H2" "H3" "H4"
##  [61] "H6" "H8" "H9" "HB" "HH" "HR" "HS" "I1" "I3" "I4" "I5" "I6" "I7" "I9" "J1"
##  [76] "J4" "J5" "J8" "J9" "K1" "K2" "K3" "K4" "K5" "K6" "K7" "K8" "K9" "L1" "L3"
##  [91] "L8" "L9" "M1" "M2" "M3" "M4" "M9" "N2" "N9" "O1" "O2" "O3" "O4" "O5" "O6"
## [106] "O7" "O8" "O9" "P2" "P5" "P6" "P7" "P8" "P9" "Q1" "Q8" "Q9" "R0" "R1" "R2"
## [121] "R3" "R4" "R5" "R6" "R7" "R8" "R9" "RA" "RB" "RG" "RH" "RK" "RP" "RR" "RS"
## [136] "RT" "RW" "S0" "S1" "S2" "S3" "S4" "S5" "S9" "T2" "U1" "U6" "V0" "V1" "V2"
## [151] "V3" "V6" "V9" "W1" "W2" "W3" "W4" "W6" "W8" "W9" "Y1" "Y3" "Z0" "Z2" "Z3"
## [166] "Z7" "Z9"

## [1] "BUILDING CLASS AT TIME OF SALE"

##   [1] "A0" "A1" "A2" "A3" "A4" "A5" "A6" "A7" "A9" "B1" "B2" "B3" "B9" "C0" "C1"
##  [16] "C2" "C3" "C4" "C5" "C6" "C7" "C8" "C9" "CM" "D0" "D1" "D2" "D3" "D4" "D5"
##  [31] "D6" "D7" "D8" "D9" "E1" "E2" "E7" "E9" "F1" "F2" "F4" "F5" "F9" "G0" "G1"
##  [46] "G2" "G3" "G4" "G5" "G6" "G7" "G8" "G9" "GU" "GW" "H1" "H2" "H3" "H4" "H6"
##  [61] "H8" "H9" "HB" "HH" "HR" "HS" "I1" "I3" "I4" "I5" "I6" "I7" "I9" "J1" "J4"
##  [76] "J5" "J8" "J9" "K1" "K2" "K3" "K4" "K5" "K6" "K7" "K8" "K9" "L1" "L3" "L8"
##  [91] "L9" "M1" "M2" "M3" "M4" "M9" "N2" "N9" "O1" "O2" "O3" "O4" "O5" "O6" "O7"
## [106] "O8" "O9" "P2" "P5" "P6" "P7" "P8" "P9" "Q1" "Q8" "Q9" "R0" "R1" "R2" "R3"
## [121] "R4" "R5" "R6" "R7" "R8" "R9" "RA" "RB" "RG" "RH" "RK" "RP" "RR" "RS" "RT"
## [136] "RW" "S0" "S1" "S2" "S3" "S4" "S5" "S9" "T2" "U1" "U6" "V0" "V1" "V2" "V3"
## [151] "V6" "V9" "W1" "W2" "W3" "W4" "W6" "W8" "W9" "Y1" "Y3" "Z0" "Z2" "Z3" "Z7"
## [166] "Z9"
```

```
## [1] "ZIP CODE"
```

```
##   [1] "0"     "10001" "10002" "10003" "10004" "10005" "10006" "10007" "10009"
##  [10] "10010" "10011" "10012" "10013" "10014" "10016" "10017" "10018" "10019"
##  [19] "10021" "10022" "10023" "10024" "10025" "10026" "10027" "10028" "10029"
##  [28] "10030" "10031" "10032" "10033" "10034" "10035" "10036" "10037" "10038"
##  [37] "10039" "10040" "10044" "10065" "10069" "10075" "10105" "10128" "10167"
##  [46] "10280" "10281" "10282" "10301" "10302" "10303" "10304" "10305" "10306"
##  [55] "10307" "10308" "10309" "10310" "10312" "10314" "10451" "10452" "10453"
##  [64] "10454" "10455" "10456" "10457" "10458" "10459" "10460" "10461" "10462"
##  [73] "10463" "10464" "10465" "10466" "10467" "10468" "10469" "10470" "10471"
##  [82] "10472" "10473" "10474" "10475" "10803" "11001" "11004" "11005" "11040"
##  [91] "11101" "11102" "11103" "11104" "11105" "11106" "11109" "11201" "11203"
## [100] "11204" "11205" "11206" "11207" "11208" "11209" "11210" "11211" "11212"
## [109] "11213" "11214" "11215" "11216" "11217" "11218" "11219" "11220" "11221"
## [118] "11222" "11223" "11224" "11225" "11226" "11228" "11229" "11230" "11231"
## [127] "11232" "11233" "11234" "11235" "11236" "11237" "11238" "11239" "11249"
## [136] "11354" "11355" "11356" "11357" "11358" "11360" "11361" "11362" "11363"
## [145] "11364" "11365" "11366" "11367" "11368" "11369" "11370" "11372" "11373"
## [154] "11374" "11375" "11377" "11378" "11379" "11385" "11411" "11412" "11413"
## [163] "11414" "11415" "11416" "11417" "11418" "11419" "11420" "11421" "11422"
## [172] "11423" "11426" "11427" "11428" "11429" "11430" "11432" "11433" "11434"
## [181] "11435" "11436" "11691" "11692" "11693" "11694"
```

```
## [1] "YEAR BUILT"
```

```
##   [1] 1900 1913 1920 2009 1925 1902 1928 1910 1930 1935 1937 1915 1950 1929 1901
##  [16] 1940 2005    0 1989 2014 2008 1965 2013 2003 2006 2007 1951 1899 1850 1905
##  [31] 1864 1917 1911 1983 1926 1963 1960 1889 1898 1939 1938 1927 1909 1958 1904
##  [46] 1907 1987 1931 1984 1948 2004 1875 2012 1973 2011 1922 2001 1932 1980 1908
##  [61] 1953 1906 1921 2010 1111 1918 1990 1890 1991 1895 1924 2016 1957 1986 1988
##  [76] 1998 1870 2015 1969 1956 1982 1914 1903 1967 1912 1964 1955 1961 1851 2000
##  [91] 1959 1962 1972 1976 1880 1970 1846 1941 1952 1923 1896 1985 1966 1981 1954
## [106] 1946 1947 1975 1974 1968 2002 1994 1892 1996 1945 1916 1949 1999 1800 1977
## [121] 1942 1979 1978 1971 1919 1894 1881 1936 1997 1995 1933 1934 1992 1993 1944
## [136] 1883 1943 1847 1844 1835 1852 1856 1854 1832 1845 1849 1855 1893 1865 1882
## [151] 1871 1891 2017
```

Data Details

Below table shows final cleaned and structured dataset's columns and their datatype and description. Descriptions are taken from provided GLOSSARY OF TERMS link https://www.nyc.gov/assets/finance/downloads/pdf/07pdf/glossary_rsf071607.pdf

Column Name	Data Type	Column Description
BOROUGH	factor	The name of the borough in which the property is located
NEIGHBORHOOD	character	Neighbourhood name
BUILDING CLASS CATEGORY NUMBER	factor	Building class category code to identify similar properties
BUILDING CLASS CATEGORY	factor	Building class category title to identify similar properties

Column Name	Data Type	Column Description
TAX CLASS AT PRESENT	factor	Assigned tax class of the property in the city - Classes 1, 2, 3 or 4
BLOCK	character	Sub-division of the borough on which real properties are located
LOT	character	Sub-division of a Tax Block, used to uniquely represent the property location
BUILDING CLASS AT PRESENT	factor	Used to describe a property's constructive use
ADDRESS	character	Property's street address
APARTMENT NUMBER	character	Property's apartment number
ZIP CODE	factor	Property's postal code
RESIDENTIAL UNITS	integer	Number of residential units at the listed property
COMMERCIAL UNITS	integer	Number of commercial units at the listed property
TOTAL UNITS	integer	Total number of units at the listed property
LAND SQUARE FEET	numeric	Land area of the property listed in square feet
GROSS SQUARE FEET	numeric	Total area of all the floors of a building
YEAR BUILT	numeric	Property's construction year
TAX CLASS AT TIME OF SALE	factor	Assigned tax class of the property in the city at the time of sale
BUILDING CLASS AT TIME OF SALE	factor	Used to describe a property's constructive use at the time of sale
SALE PRICE	numeric	Price paid for the property
SALE DATE	Date	Date of property sale
BUILDING AGE	numeric	Age of the Building

ANALYSIS/METHODS

We uses various section under this section to analyze NYC Property Sales data. We uses 3 main analysis section 1. Numerical Columns -> explores each Numerical Column. 2. Visualization -> where we view the data and there relations of various columns. 3. Prediction -> This does predective analysis where its uses varios methods like corellations, single or multi regression.

Descriptive Analysis

Here we explore the columns SALE PRICE, LAND SQUARE FEET, GROSS SQUARE FEET, RESIDENTIAL UNITS , COMMERCIAL UNITS, TOTAL UNITS and BUILDING AGE. We will understand further on data distrbution in these columns.

Lets check the clean current dataset that we have created.

```
dim(nyc_data)
```

```
## [1] 59595    22
```

The data set has 59595 rows and 22 column.

SALE PRICE Lets apply the quintal function to understand the data distribution

```
#apply the quantile function to get the distribution
quantile(nyc_data$`SALE PRICE`, probs = seq(from = 0, to = 1, by = .1))
```

```
##          0%         10%         20%         30%         40%         50%         60%
##           1      199000      319410      420810      515000      628000      753403
##         70%         80%         90%        100%
##      935000     1290000     2250000 2210000000
```

```
nyc_data %>% filter(`SALE PRICE` <= 1000) %>% nrow()
```

```
## [1] 1125
```

As we have remove defective value earlier, we will now check on sale price less than equal to 1000 dollars which looks like low data points. There are 1125 observations with Sale Price <= $1000. We will treat it as faulty. Will add these values into defect dataset and updated dataset.

```
#Adding these rows to deffect datse
nyc_data <-  nyc_curr_data %>% filter(`SALE PRICE` <= 1000) %>%
  bind_rows(nyc_data)

#Update current dataset
nyc_data <- nyc_data %>% filter(!`SALE PRICE` <= 1000)

#apply quantile to check this data distribution
quantile(nyc_data$`SALE PRICE`, probs = seq(from = 0, to = 1, by = .1))
```

```
##          0%         10%         20%         30%         40%         50%         60%
##        1110      220000      333240      435000      529000      640000      765000
##         70%         80%         90%        100%
##      950000     1300000     2300000 2210000000
```

Though its created little bit better distribution but you can still find data haveing values like 2210000000 and will also have low outliers

LAND SQUARE FEET We will apply both summary and quintal to check the data distribution

```
# get the summary
summary(nyc_data$`LAND SQUARE FEET`)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      33    2000    2500    4618    4000 4252327   29082
```

```
#apply quantile to check this data distribution
quantile(nyc_data$`LAND SQUARE FEET`, probs = seq(from = 0, to = 1, by = .1), na.rm = TRUE)
```

```
##         0%        10%        20%        30%        40%        50%        60%        70%
##       33.0     1680.0     1950.0     2003.0     2351.0     2500.0     2847.2     3500.0
##         80%        90%       100%
##     4000.0     5300.0 4252327.0
```

From above distrbution value we can guess above 500,000sq looks like commercial building, commercial land, factories etc.

GROSS SQUARE FEET We will apply both summary and quintal to check the data distribution

```
# get the summary
summary(nyc_data$`GROSS SQUARE FEET`)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     120    1360    1872    4396    2658 3750565   30070
```

```
#apply quantile to check this data distribution
quantile(nyc_data$`GROSS SQUARE FEET`, probs = seq(from = 0, to = 1, by = .1), na.rm = TRUE)
```

```
##        0%       10%       20%       30%       40%       50%       60%       70%
##     120.0    1120.0    1280.0    1448.0    1640.0    1872.0    2120.0    2420.0
##       80%       90%      100%
##    2975.2    3984.0 3750565.0
```

As expected, over 500,000sq properties are for comercial purpose.

RESIDENTIAL UNITS , COMMERCIAL UNITS and TOTAL UNITS We will apply both summary and check for 5 RESIDENTIAL UNITS. Will look into our asumption that TOTAL UNITS = RESIDENTIAL UNITS +COMMERCIAL UNITS.

```
# get the summary RESIDENTIAL UNITS
summary(nyc_data$`RESIDENTIAL UNITS`)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    0.000    0.000    1.000    1.702    1.000 1844.000
```

```
#lets check for 5 units
nyc_data %>% filter(`RESIDENTIAL UNITS` > 5) %>% nrow()
```

```
## [1] 1376
```

We will apply both summary and check for 5 COMMERCIAL UNITS

```
# get the summary RESIDENTIAL UNITS
summary(nyc_data$`COMMERCIAL UNITS`)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##    0.0000    0.0000    0.0000    0.1546    0.0000 2261.0000
```

```
#lets check for 5 units
nyc_data %>% filter(`COMMERCIAL UNITS` > 5) %>% nrow()
```

```
## [1] 140
```

```
#lets check for 5 units
nyc_data %>% filter(`TOTAL UNITS` > 5) %>% nrow()
```

```
## [1] 1573
```

This proof Total Units (1573) is not equal to Residential and Commericial Units (1376 + 140= 1516). As Total Units has NAs, we will be using this field for further analysis.

BUILDING AGE There are 4195 properties with Building age. When we remove properties that don't have a Year Built entry or Year Built = 0, we get 28 property details.

```
# get the summary BUILDING AGE
summary(nyc_data$`BUILDING AGE`)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    51.0    77.0   205.2    97.0  2017.0
```

```
# get the summary YEAR BUILT
summary(nyc_data$`YEAR BUILT`)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0    1920    1940    1812    1966    2017
```

```
#number of BUILDING AGE properties over the mean age
nyc_data %>% filter(`BUILDING AGE` > 205) %>% nrow()
```

```
## [1] 4195
```

```
#number of BUILDING AGEproperties over the mean age and YEAR BUILT is not NA and Zero
nyc_data %>% filter(`BUILDING AGE` > 205 & `YEAR BUILT` != 0) %>%
  arrange(desc(`BUILDING AGE`)) %>% nrow()
```

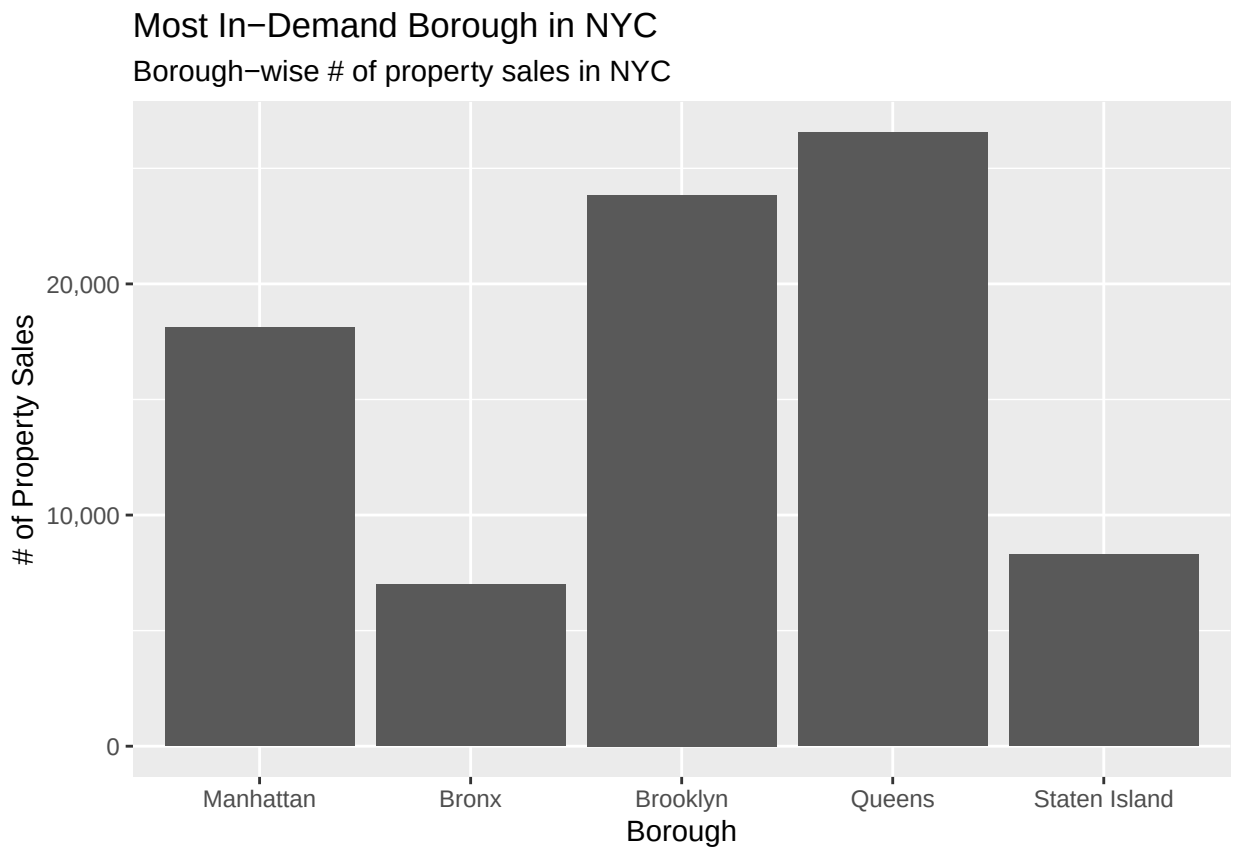
```
## [1] 28
```

Visualization Analysis

This explore the relationship between the Numerical and Category columns by visualizing the fields.

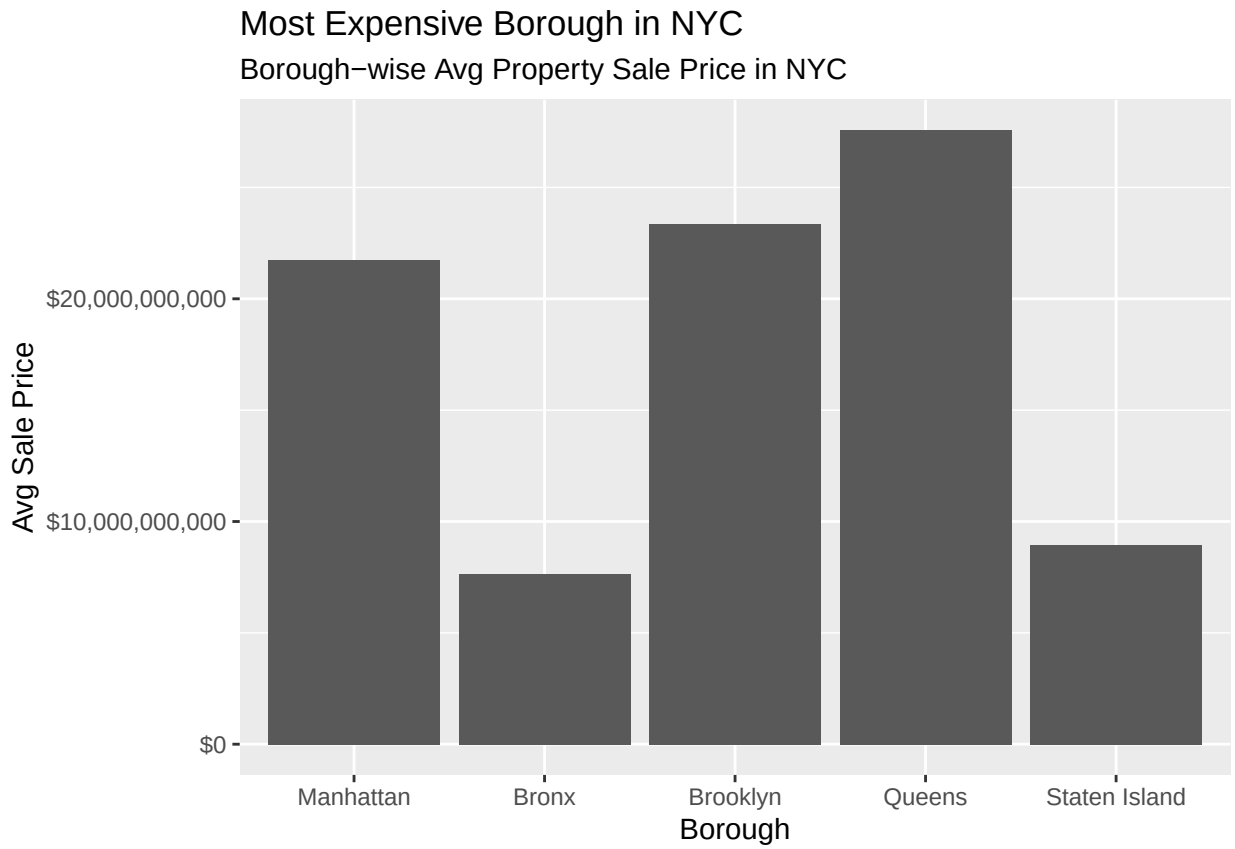
BOROUG We plot most property sales in each brough

```
ggplot(data = nyc_curr_data, aes(x = `BOROUGH`)) +
  geom_bar() +
  ggtitle("Most In-Demand Borough in NYC", subtitle = "Borough-wise # of property sales in NYC") +
  scale_y_continuous("# of Property Sales", labels = scales::comma) +
  scale_x_discrete("Borough")
```



We plot average property sales in each brough

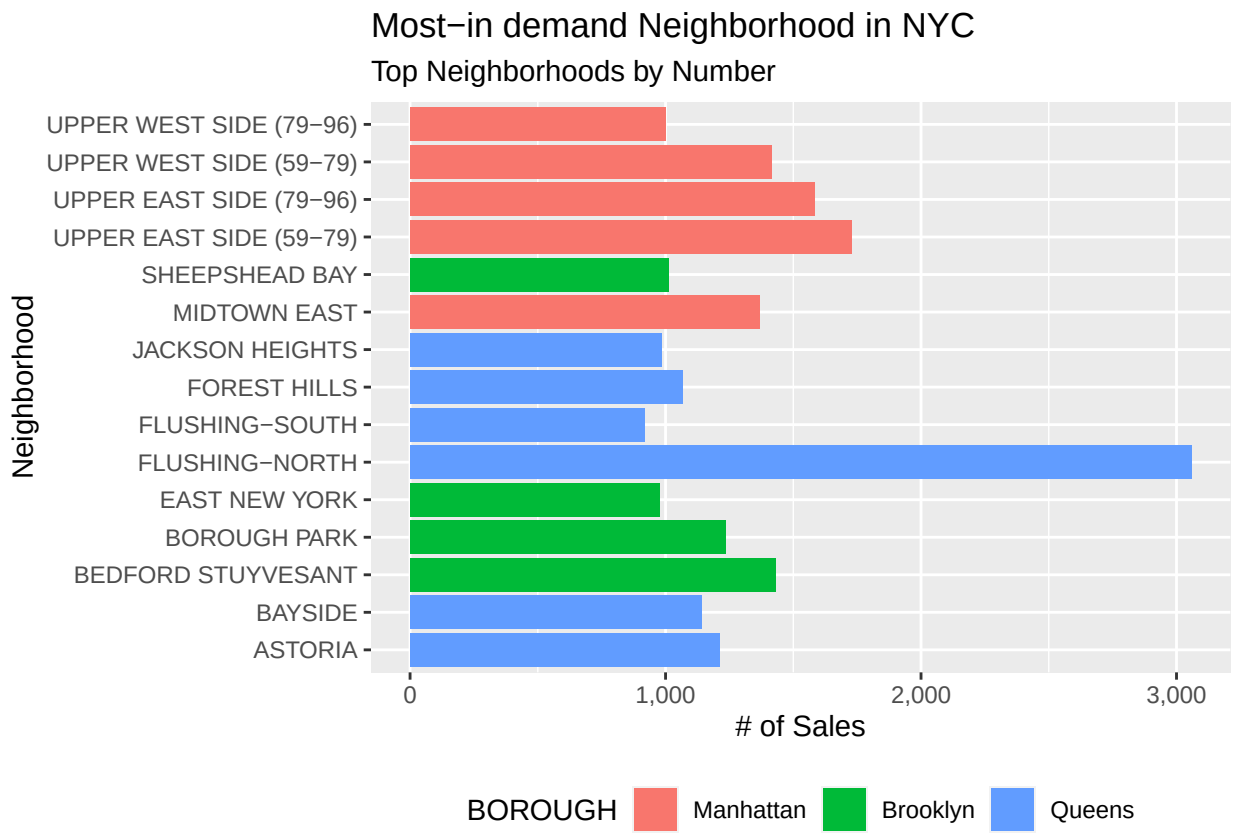
```
ggplot(data = nyc_data, aes(x = `BOROUGH`, y = mean(`SALE PRICE`) )) +
 geom_bar(stat = "identity") +
 ggtitle("Most Expensive Borough in NYC", subtitle = "Borough-wise Avg Property Sale Price in NYC") +
 scale_y_continuous("Avg Sale Price", labels = scales::dollar) +
 scale_x_discrete("Borough")
```



The Average Sale Price of a property in Queens is higher than Manhattan though Manhattan cost price is more.

NEIGHBORHOOD We do furthe analysis to understand the price of neighborhood inside Borough. We look into Number of Sales the most in-demand neighborhoods. We look for top 15 neighborhoods

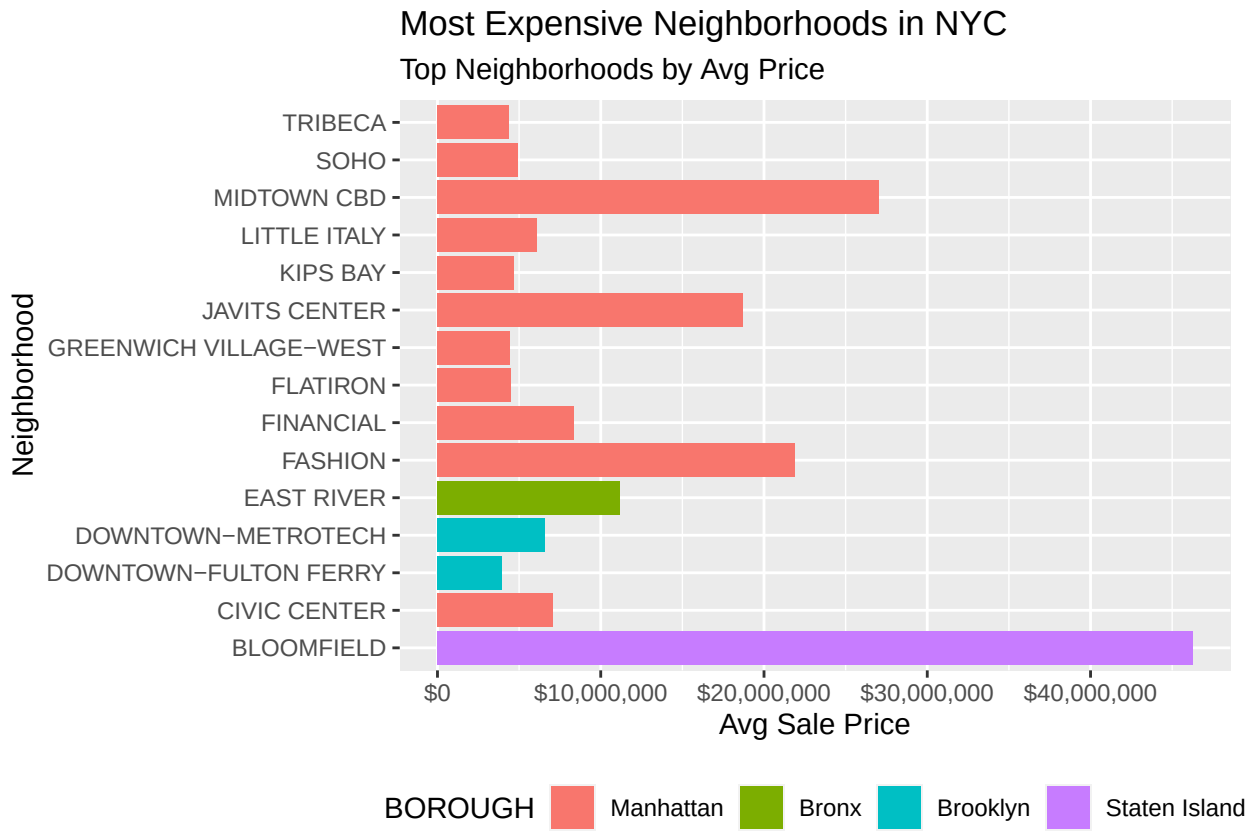
```
#create data frame based on descending order
df1 <- as.data.frame(table(nyc_curr_data$BOROUGH, nyc_curr_data$NEIGHBORHOOD))
names(df1) <- c('BOROUGH','NEIGHBORHOOD', 'Freq')
df1 <- df1 %>% arrange(desc(Freq)) %>% head(15)

#plot Top Neighborhood by sales price
ggplot(df1, aes(x = `NEIGHBORHOOD`, y = `Freq`, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  ggtitle("Most-in demand Neighborhood in NYC", subtitle = "Top Neighborhoods by Number") +
  theme(legend.position = "bottom") +
  scale_y_continuous("# of Sales", labels = scales::comma) +
  scale_x_discrete("Neighborhood")
```



Top Neighborhoods by # of Property Sales plot shows that neighborhoods in Queens and Manhattan had the most number of properties sold and this accounted to 11 out of the 15 top neighborhoods.

We plot Average property Sales Prices across the most in-demand neighborhoods.

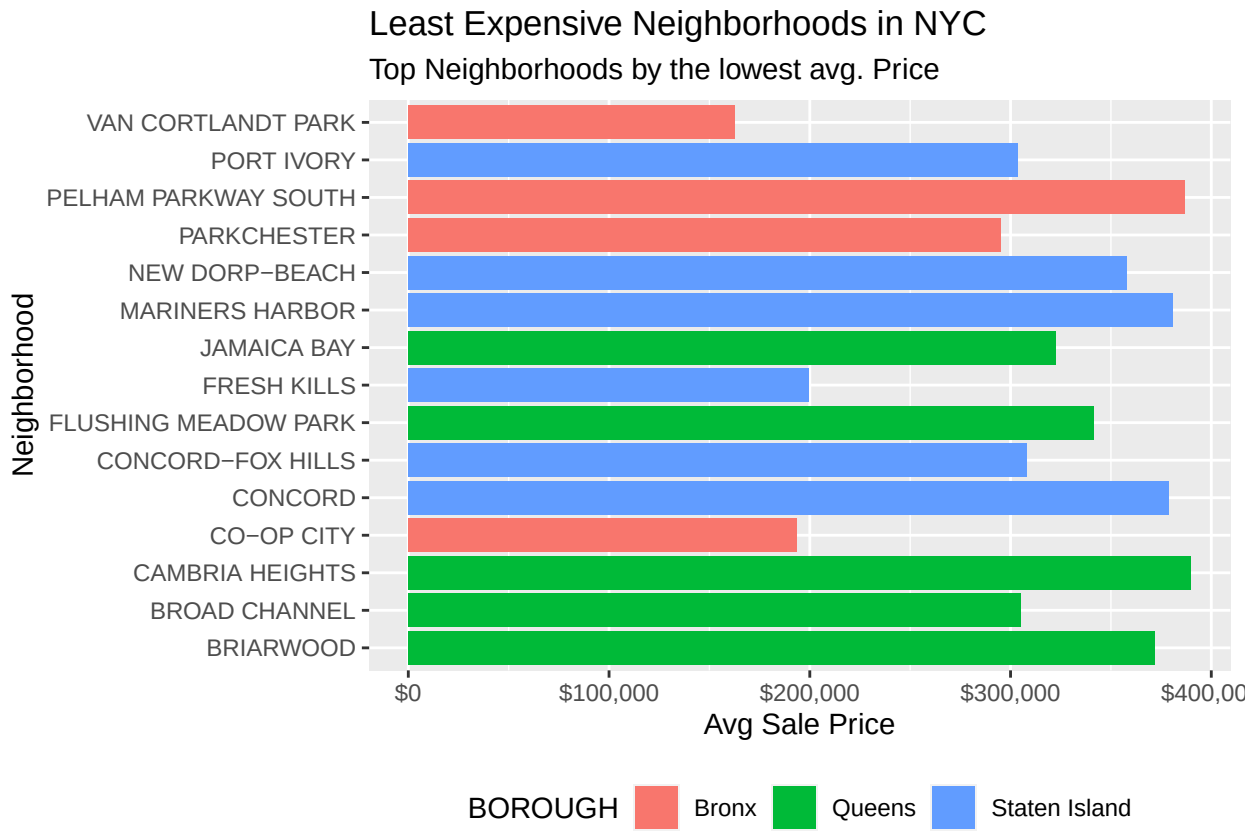
```
##filter data based on descending order
df2 <-
  nyc_data %>% group_by(BOROUGH, NEIGHBORHOOD) %>%
  summarise(MeanSP = mean(`SALE PRICE`)) %>%
  arrange(desc(MeanSP)) %>% head(15)

#plot Most Expensive Neighborhoods by Avg Price
ggplot(data = df2, aes(x = `NEIGHBORHOOD`, y = MeanSP, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("Most Expensive Neighborhoods in NYC",
          subtitle = "Top Neighborhoods by Avg Price") +
  scale_y_continuous("Avg Sale Price", labels = scales::dollar) +
  scale_x_discrete("Neighborhood")
```



The avg. price of the Most expensive Neighborhood and the second most expensive by $12.5 billion. Needless to say, the standard deviation of the property prices in NYC is large. Also interesting to note is that 11 out the 15 top property value neighborhoods are from Manhattan. Also even though no neighborhood in Queens fetched top bucks, it sold much more properties than the other boroughs. Staten Island show the biggest sale of 46 millon.

We plot least Sales Prices across the most in-demand neighborhoods.

```
##filter data
df2 <-
  nyc_data %>% group_by(BOROUGH, NEIGHBORHOOD) %>%
  summarise(MeanSP = mean(`SALE PRICE`)) %>%
  arrange(MeanSP) %>% head(15)

#plot Top Neighborhoods by the lowest avg. Price
ggplot(data = df2, aes(x = `NEIGHBORHOOD`, y = MeanSP, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("Least Expensive Neighborhoods in NYC", subtitle = "Top Neighborhoods by the lowest avg. Price
  scale_y_continuous("Avg Sale Price", labels = scales::dollar) +
  scale_x_discrete("Neighborhood")
```

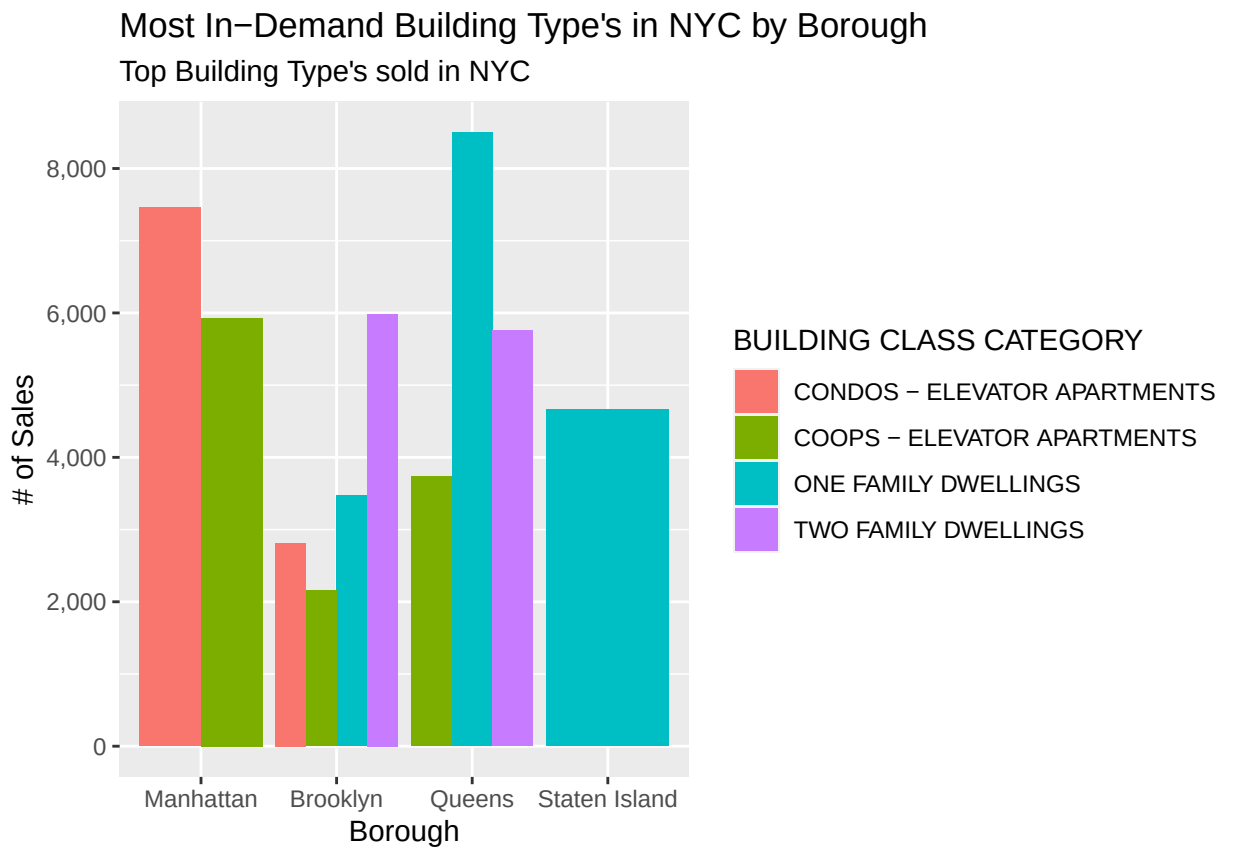


Interesting to note is the differences in the Avg Sale Price scale of the Most and the Least expensive properties in NYC. The avg property price in Van Cortlandt Park in the Bronx sold for dollar 160,000, while the most expensive property in Staten Island, Bloomsfield sold for $46 billion.

BUILDING CLASS CATEGORY We will try to find which type of building is sold across all the Borough in New York

We plot Most In-Demand Buildings and Expensive in NYC by Borough

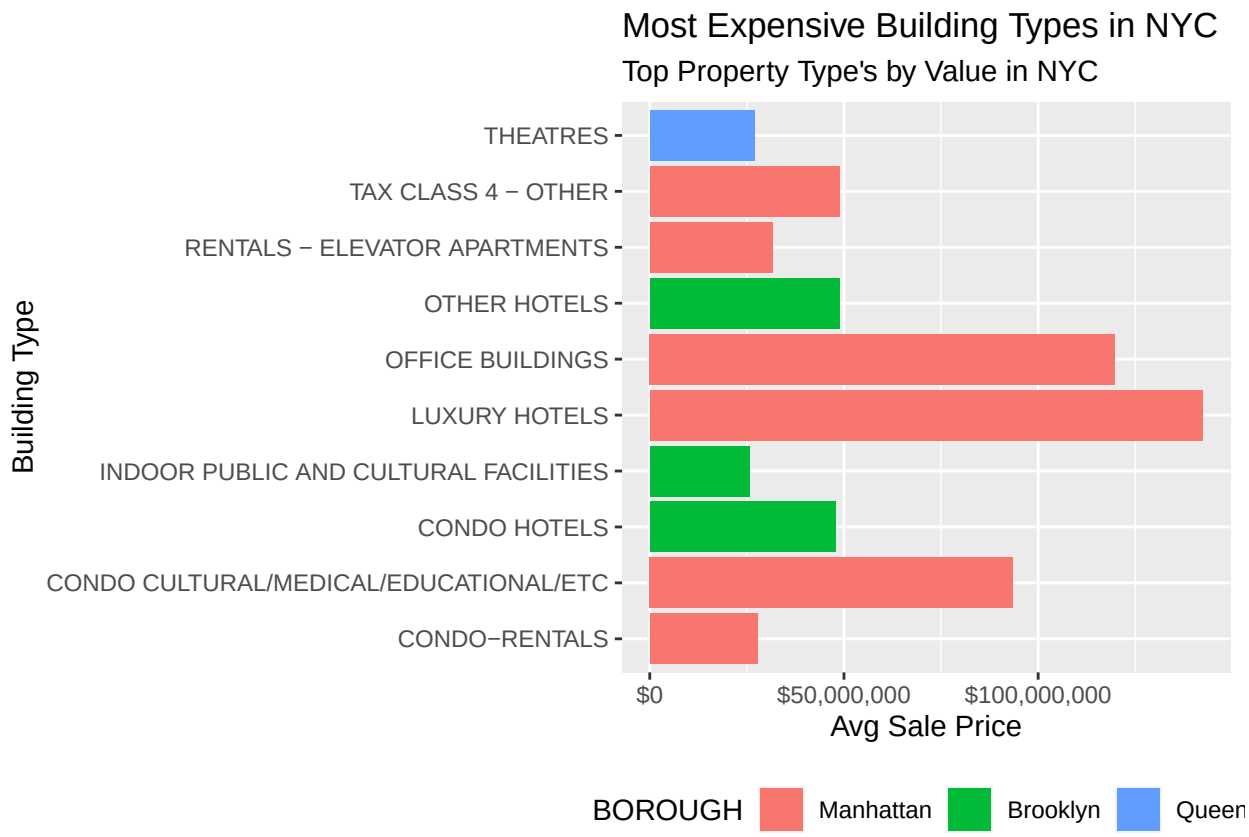
```
##filter data
df1 <- as.data.frame(table(nyc_curr_data$BOROUGH, nyc_curr_data$`BUILDING CLASS CATEGORY`))
names(df1) <- c('BOROUGH','BUILDING CLASS CATEGORY', 'Freq')
df1 <- df1 %>% group_by(BOROUGH) %>% arrange(desc(Freq)) %>% head(10)

##plot Most In-Demand Building Type class by Borough
ggplot(df1, aes(x = `BOROUGH`, y = `Freq`, fill = `BUILDING CLASS CATEGORY`)) +
  geom_bar(stat = "identity", position = "dodge") +
  ggtitle("Most In-Demand Building Type's in NYC by Borough", subtitle = "Top Building Type's sold in N
  scale_y_continuous("# of Sales", labels = scales::comma) +
  scale_x_discrete("Borough")
```



From above we see that most one family dwellings sold in queens, Staten Island and Brooklyn. Similarly Coops in Elevator Apartments are also wanted in Manhatttan, Queens and Brooklyn.

Now lets plot Most Expensive Buildings by value in NYC

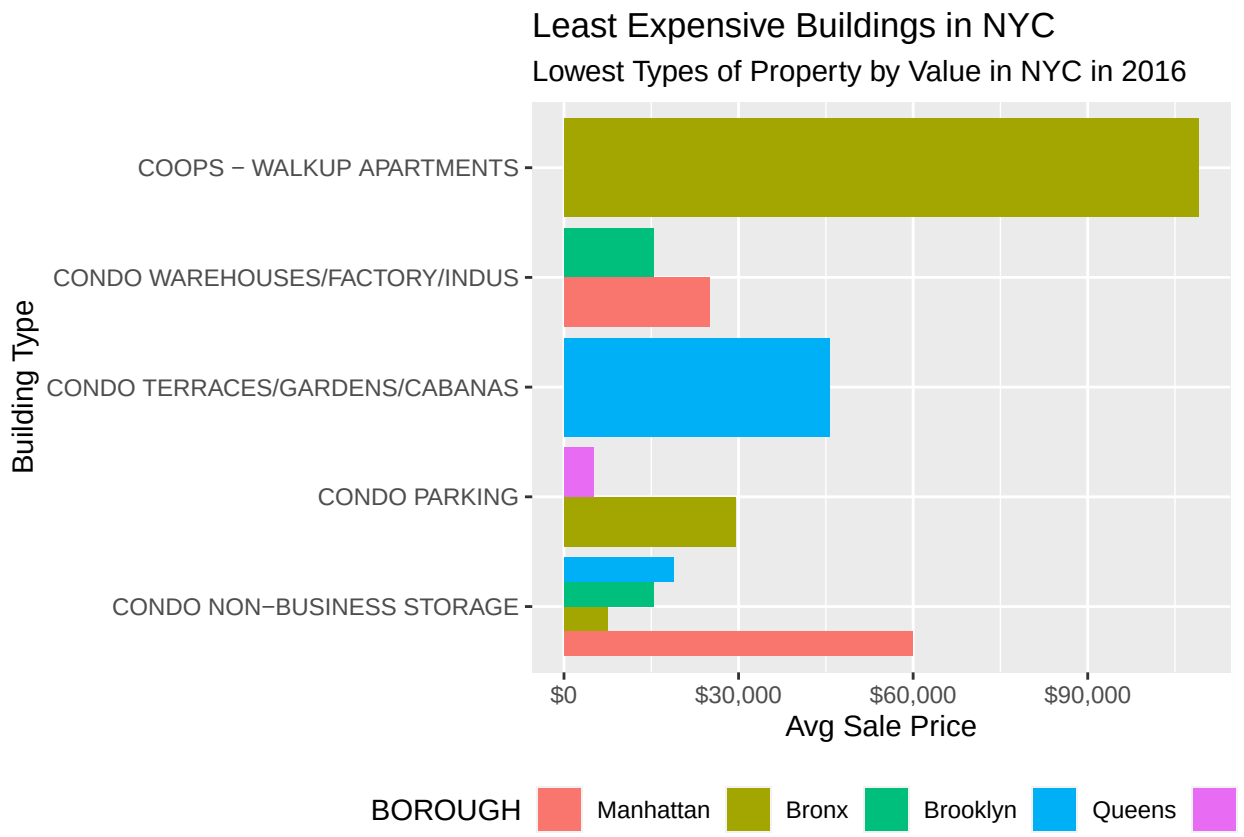
```
##filter data
df2 <-
  nyc_data %>% group_by(BOROUGH, `BUILDING CLASS CATEGORY`) %>%
  summarise(MeanSP = mean(`SALE PRICE`)) %>%
  arrange(desc(MeanSP)) %>% head(10)

#plot Most Expensive Building Types by Borough
ggplot(data = df2, aes(x = `BUILDING CLASS CATEGORY`, y = MeanSP, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("Most Expensive Building Types in NYC", subtitle = "Top Property Type's by Value in NYC") +
  scale_y_continuous("Avg Sale Price", labels = scales::dollar) +
  scale_x_discrete("Building Type")
```



We see Manhattan mostly has commercial buildings and are expensive. The apartments and condos are expensive but cheaper side.

Lets look at Least expensive Buildings in NYC

```
##filter data
df2 <-
  nyc_data %>% group_by(BOROUGH, `BUILDING CLASS CATEGORY`) %>%
  summarise(MeanSP = mean(`SALE PRICE`)) %>%
  arrange(MeanSP) %>% head(10)

#plot Least Expensive Building Types by Borough
ggplot(data = df2, aes(x = `BUILDING CLASS CATEGORY`, y = MeanSP, fill = `BOROUGH`)) +
  geom_bar(stat = "identity", position = "dodge") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("Least Expensive Buildings in NYC", subtitle = "Lowest Types of Property by Value in NYC in 2
  scale_y_continuous("Avg Sale Price", labels = scales::dollar) +
  scale_x_discrete("Building Type")
```

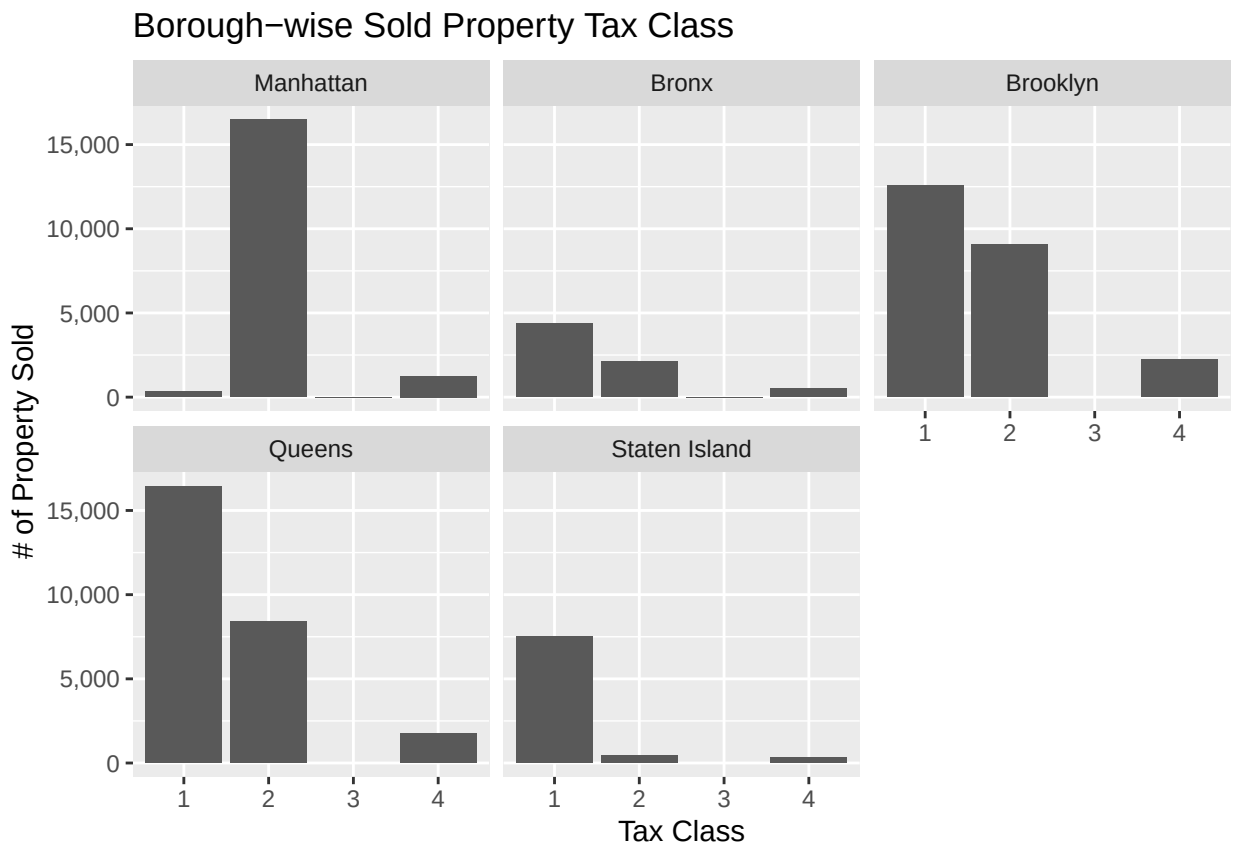


Most expensive and the least expensive buildings in NYC are commercial buildings. Intrestingly the least expensive property in NYC is a Condo Parking space in Staten island and $45,000 you could also buy a Condo Terrace in the Queens.

BOROUGH BY TAX CLASS There are 4 type of class defined in Glossary of Terms Class 1: Includes most residential property of up to three units Class 2: Includes all other property that is primarily residential, such as cooperatives and condominiums. Class 3: Includes property with equipment owned by a gas, telephone or electric company Class 4: Includes all other properties not included in class 1,2, and 3, such as offices, factories, warehouses, garage buildings, etc.

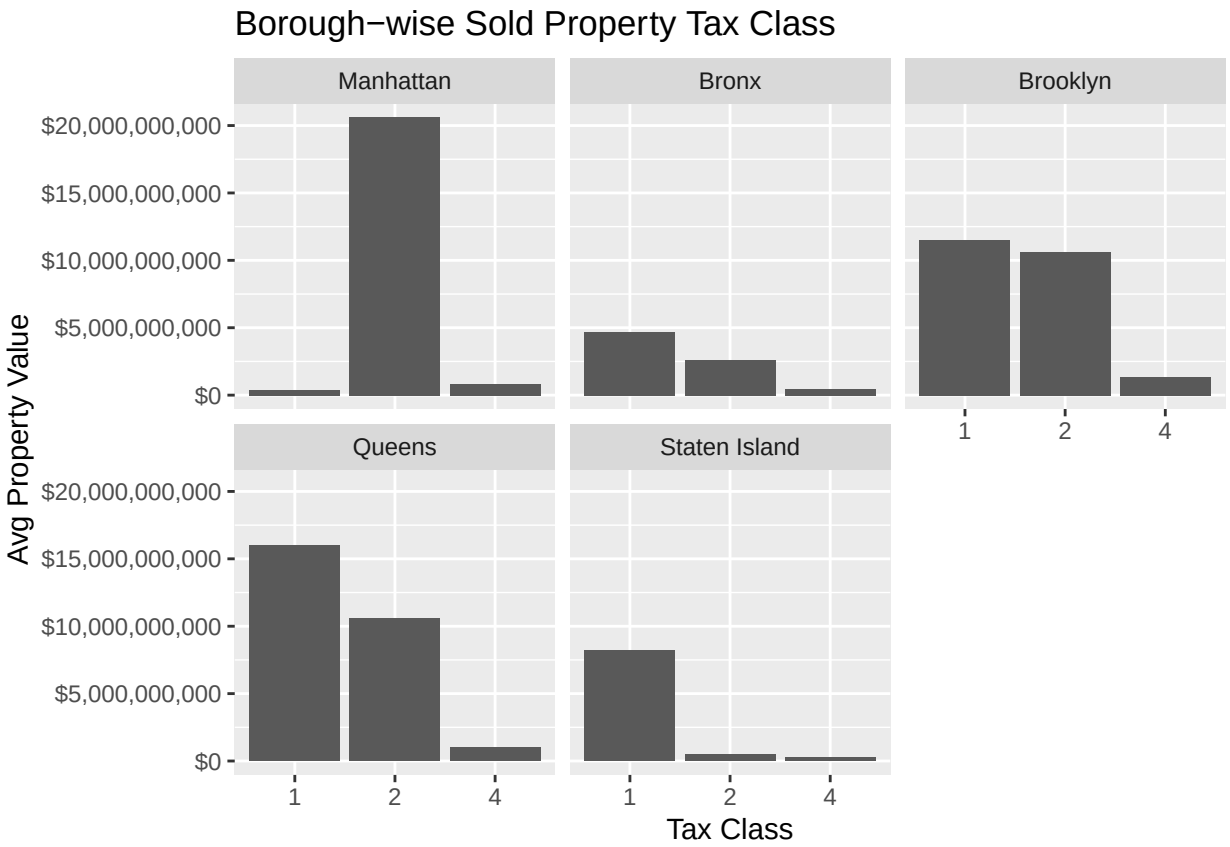
Now let's look at the Tax Class wise of the Properties sold in NYC.

```
#plot nuber of Property Tax Class wise property by Borough
ggplot(data = nyc_curr_data, aes(x = `TAX CLASS AT TIME OF SALE`)) +
  geom_bar() +
  facet_wrap(~ BOROUGH) +
  ggtitle("Borough-wise Sold Property Tax Class") +
  scale_y_continuous("# of Property Sold", labels = scales::comma) +
  scale_x_discrete("Tax Class")
```



We now plot Avg Property Price by Tax Class sold in NYC

```
#plot value of Property Tax Class wise property by Borough
ggplot(data = nyc_data, aes(x = `TAX CLASS AT TIME OF SALE`, y = mean(`SALE PRICE`))) +
  geom_bar(stat = "identity") +
  facet_wrap(~ BOROUGH) +
  ggtitle("Borough-wise Sold Property Tax Class") +
  scale_y_continuous("Avg Property Value", labels = scales::dollar) +
  scale_x_discrete("Tax Class")
```

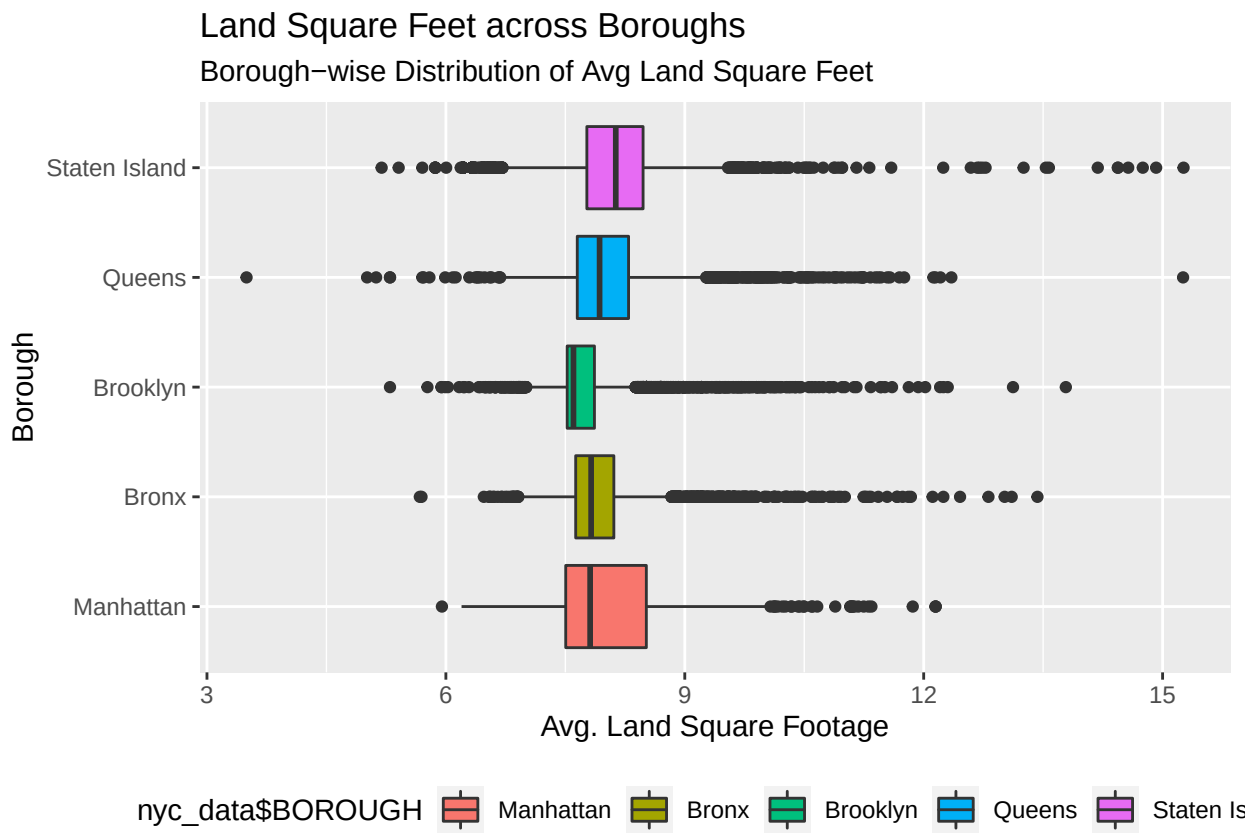


Queens has a maximum number of lower-valued small sized residential properties while Manhattan has a large number of high-value residential condos and coops.

PROPERTY BY SQUARE FEET (LAND and GROSS) To analyze the size of the property

Lets look at Land Square Feet across Boroughs

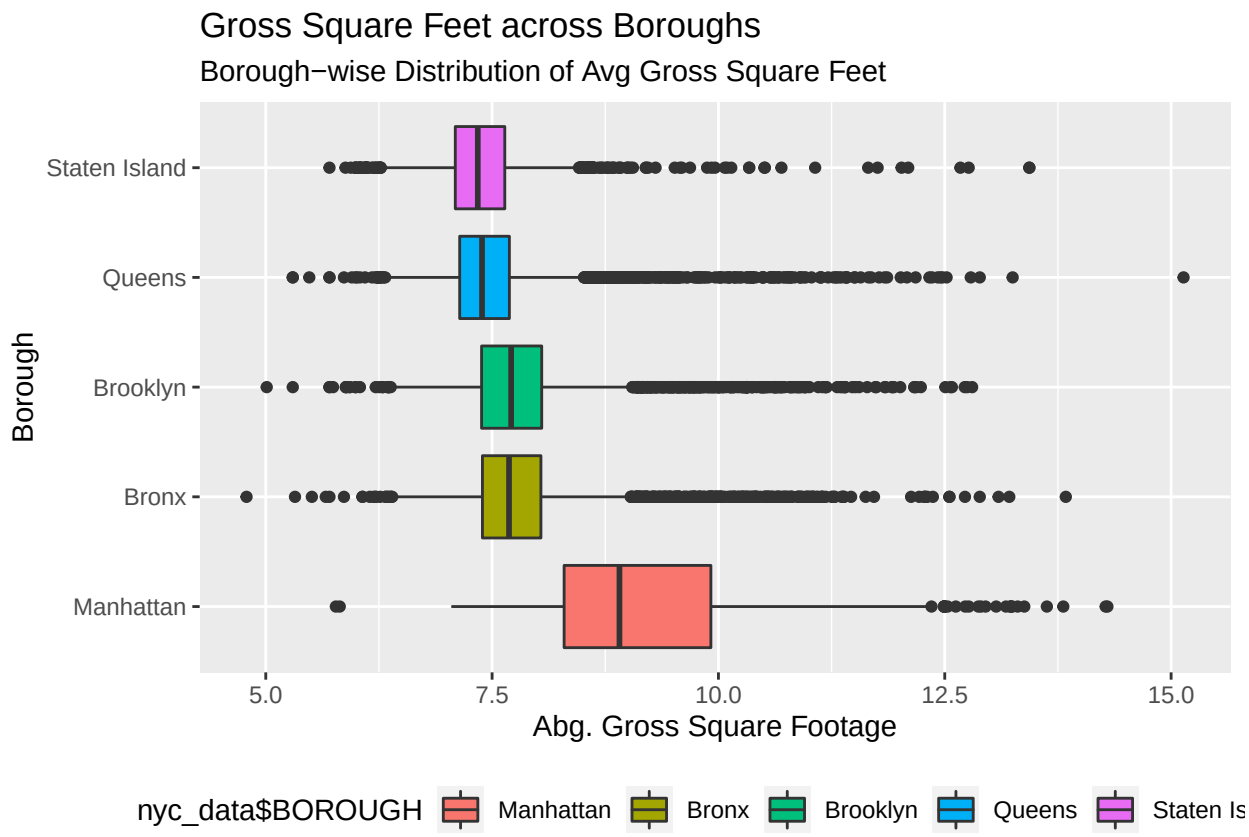
```
#plot Land Square Feet across Boroughs
ggplot(data = nyc_data, aes(x = `BOROUGH`, y = log(`LAND SQUARE FEET`), fill = nyc_data$BOROUGH)) +
  geom_boxplot() +
  theme(legend.position = "bottom") +
  ggtitle("Land Square Feet across Boroughs", subtitle = "Borough-wise Distribution of Avg Land Square
  scale_y_continuous("Avg. Land Square Footage", labels = scales::comma) +
  scale_x_discrete("Borough") +
  coord_flip()
```



Here we can see the outliers. These are the outliers we identified in the previous section's. Staten Island Borough has many outliers.

We now plot Gross Square Feet across Boroughs

```
#plot Gross Square Feet across Boroughs
ggplot(data = nyc_data, aes(x = `BOROUGH`, y = log(`GROSS SQUARE FEET`), fill = nyc_data$BOROUGH)) +
  geom_boxplot() +
  theme(legend.position = "bottom") +
  ggtitle("Gross Square Feet across Boroughs", subtitle = "Borough-wise Distribution of Avg Gross Square
  scale_y_continuous("Abg. Gross Square Footage", labels = scales::comma) +
  scale_x_discrete("Borough") +
  coord_flip()
```

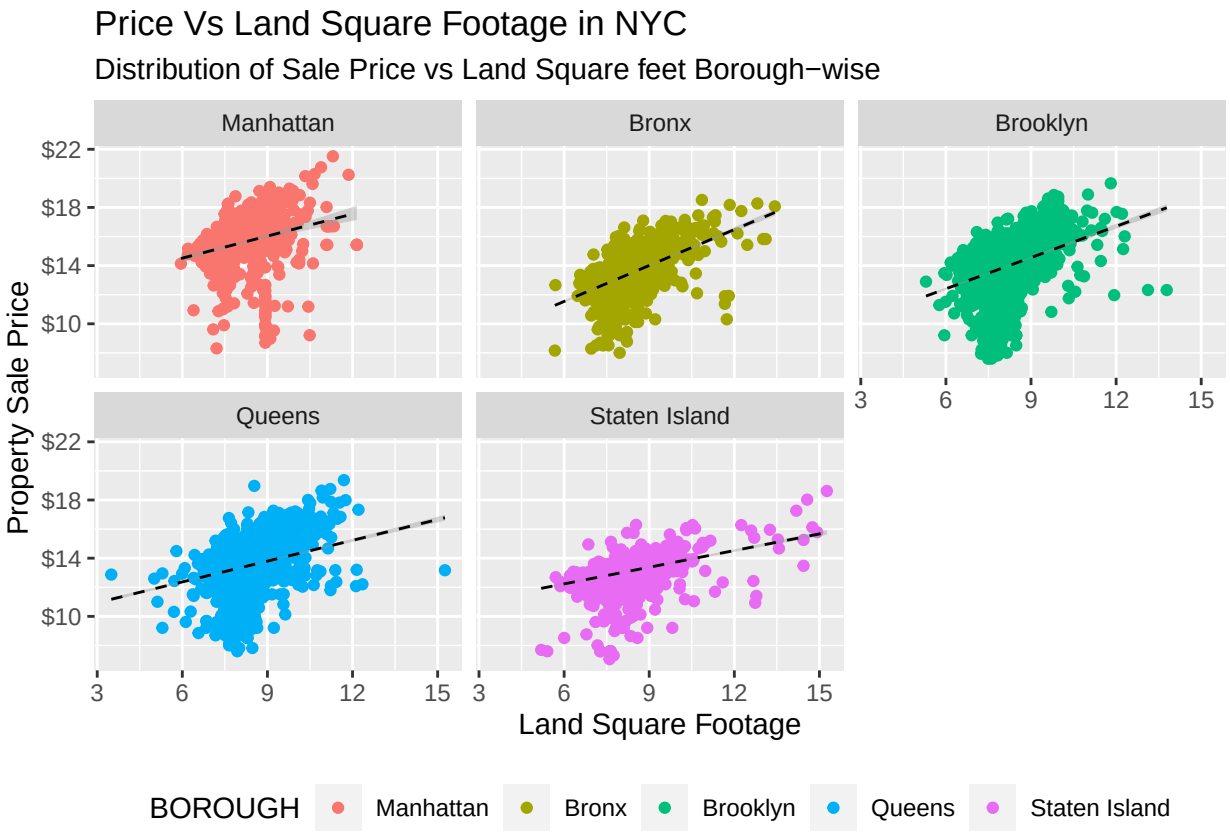


Analizing we find many property sales greater than 500,000 sq feet. Here also we can see the outliers. However, in contrast to the previous plot, a majority of these seem to be in Manhattan.

Now lets see how property size vs sale price perform.

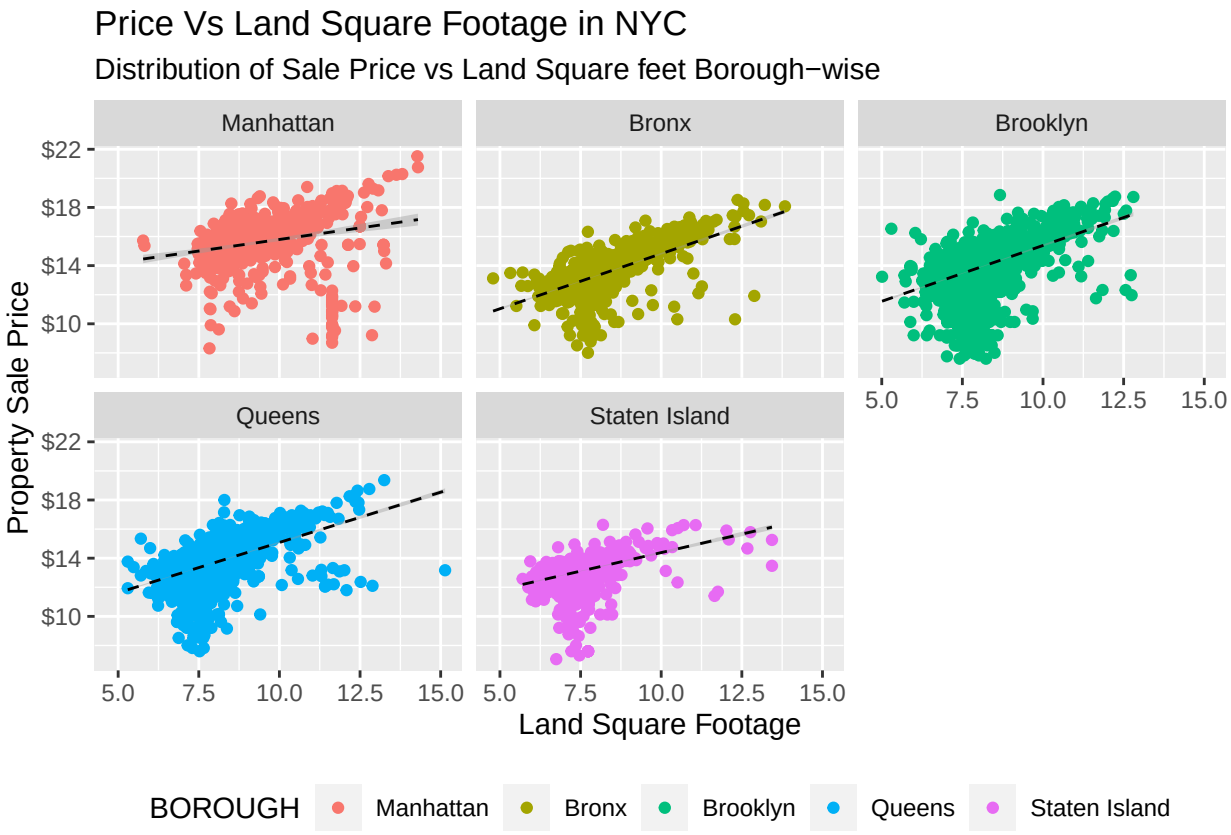
Let see the pattern SALES PRICE VS LAND SQUARE FEET

```
#plot pattern Land Square Feet vs sales price by Boroughs
ggplot(data = nyc_data, aes(x = log(`LAND SQUARE FEET`), y = log(`SALE PRICE`), color = `BOROUGH`)) +
  geom_jitter() +
  geom_smooth(method = "lm", colour="black", size=0.5, linetype = "dashed") +
  theme(legend.position = "bottom") +
  facet_wrap(~ BOROUGH) +
  ggtitle("Price Vs Land Square Footage in NYC",
          subtitle = "Distribution of Sale Price vs Land Square feet Borough-wise") +
  scale_y_continuous("Property Sale Price", labels = scales::dollar) +
  scale_x_continuous("Land Square Footage", labels = scales::comma)
```



Let see the pattern SALES PRICE VS GROSS SQUARE FEET

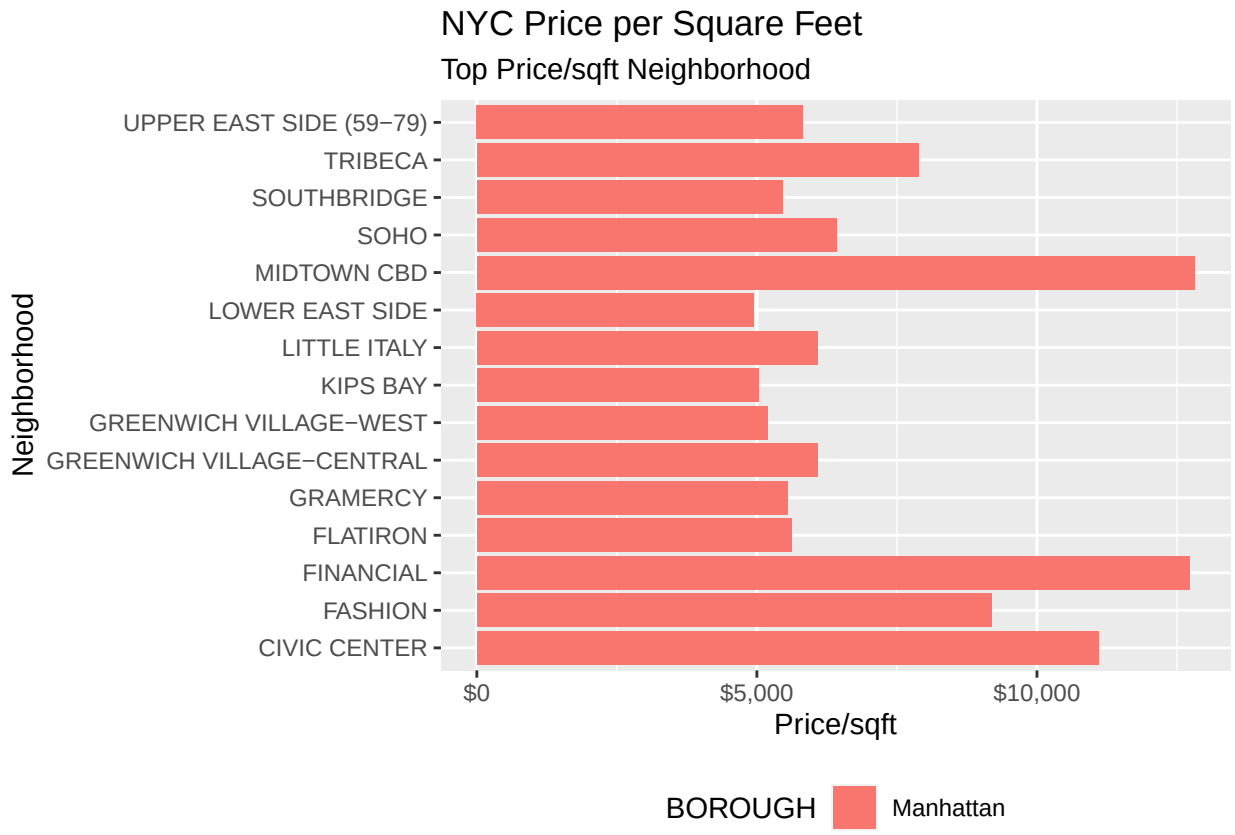
```
#plot pattern Gross Square Feet vs sales price by Boroughs
ggplot(data = nyc_data, aes(x = log(`GROSS SQUARE FEET`), y = log(`SALE PRICE`), color = `BOROUGH`)) +
  geom_jitter() +
  geom_smooth(method = "lm", colour="black", size=0.5, linetype = "dashed") +
  theme(legend.position = "bottom") +
  facet_wrap(~ BOROUGH) +
  ggtitle("Price Vs Land Square Footage in NYC",
          subtitle = "Distribution of Sale Price vs Land Square feet Borough-wise") +
  scale_y_continuous("Property Sale Price", labels = scales::dollar) +
  scale_x_continuous("Land Square Footage", labels = scales::comma)
```



The trends except Manhattan seems seem for Gross Square Footage and Land Square Footage.

We also look int the maximum price per sq/feet price for neighborhood

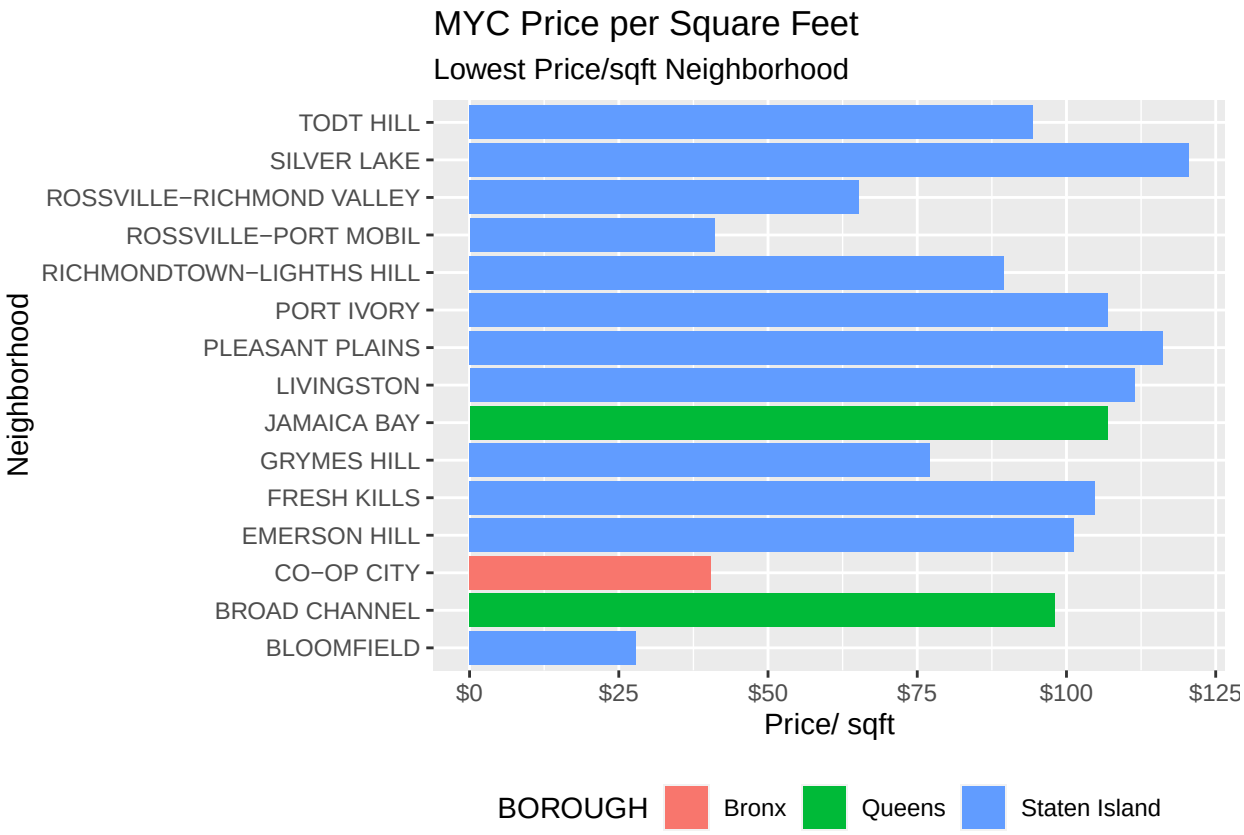
```
#filter data

df1 <- nyc_data %>% filter(`LAND SQUARE FEET` != 0) %>%
      mutate(PriceLSF = `SALE PRICE`/`LAND SQUARE FEET`) %>%
      group_by(`BOROUGH`, `NEIGHBORHOOD`) %>%
      summarise(MeanPriceLSF = mean(PriceLSF, na.rm = TRUE)) %>%
      arrange(desc(MeanPriceLSF)) %>% head(15)

ggplot(data = df1, aes(x = `NEIGHBORHOOD`, y = MeanPriceLSF, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("NYC Price per Square Feet", subtitle = "Top Price/sqft Neighborhood") +
  scale_y_continuous("Price/sqft", labels = scales::dollar) +
  scale_x_discrete("Neighborhood")
```



As expected neighbourhoods in borough Manhattan has highest price per square feet in top 15

We also look int the minimum price per sq/feet price for neighborhood

```
#filter data

df2 <- nyc_data %>% filter(`LAND SQUARE FEET` != 0) %>%
  mutate(PriceLSF = `SALE PRICE`/`LAND SQUARE FEET`) %>%
  group_by(`BOROUGH`, `NEIGHBORHOOD`) %>%
  summarise(MeanPriceLSF = mean(PriceLSF, na.rm = TRUE)) %>%
  arrange(MeanPriceLSF) %>% head(15)

ggplot(data = df2, aes(x = `NEIGHBORHOOD`, y = MeanPriceLSF, fill = `BOROUGH`)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme(legend.position = "bottom") +
  ggtitle("MYC Price per Square Feet", subtitle = "Lowest Price/sqft Neighborhood") +
  scale_y_continuous("Price/ sqft", labels = scales::dollar) +
  scale_x_discrete("Neighborhood")
```

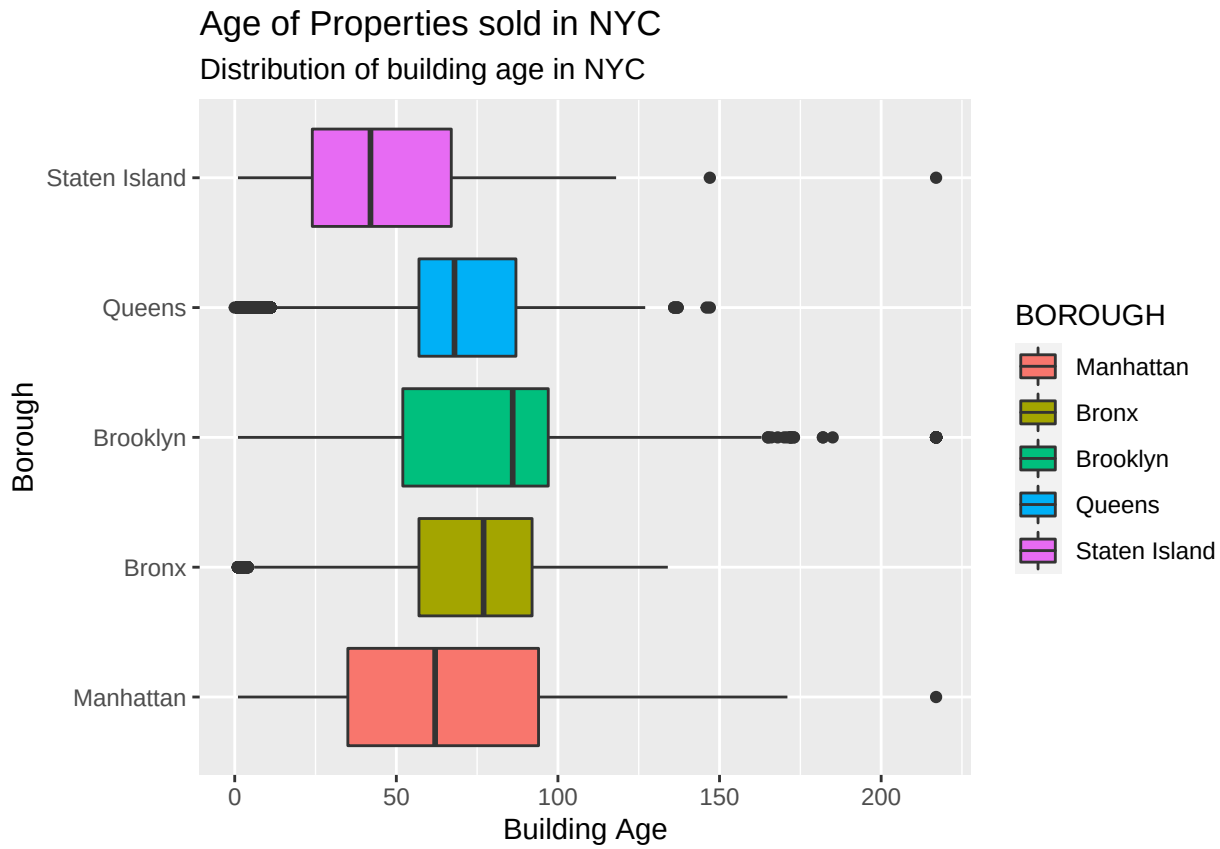


Intersting that one neighborhood of Manhattan and majority from Staten Island falls in price per square feet in lowest 15.

BUILDING AGE Here we explore the building age relationship with borough and sales price

Now lets look at how building age distributed across borough. We have use 300 as age limit.

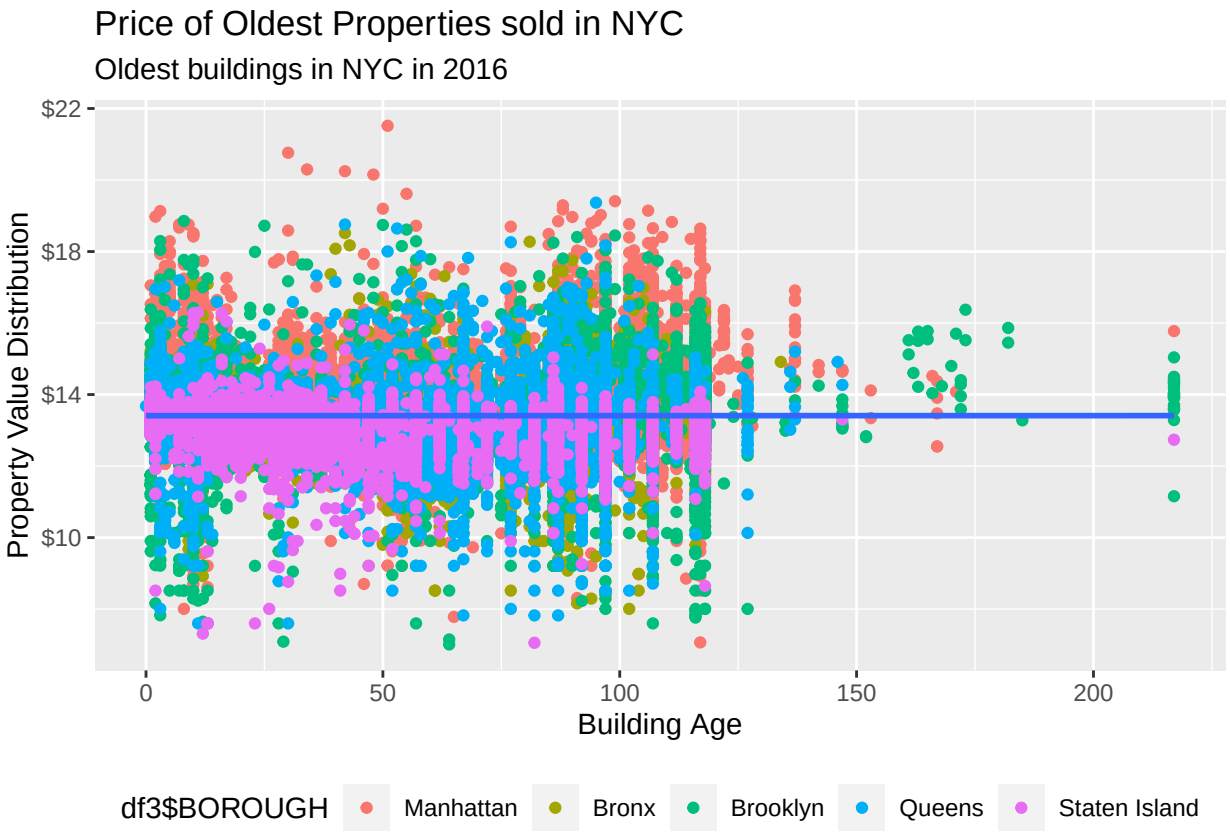
```
#filter data
df3 <-
  nyc_data %>%filter(`BUILDING AGE` <=300)

#plot building age across borough
ggplot(data = df3, aes(x = `BOROUGH`, y = `BUILDING AGE`, fill = `BOROUGH`)) +
  geom_boxplot() +
  coord_flip() +
  ggtitle("Age of Properties sold in NYC", subtitle = "Distribution of building age in NYC") +
  scale_y_continuous("Building Age", labels = scales::comma) +
  scale_x_discrete("Borough")
```



The plot shows that the new buildings are in Queens and Bronx, and the oldest building is in Manhattan. There are some outliers where building constructed in 1820

Now lets see how Sale Price distributed across Building age

```
#plot sale price of borough across building age
ggplot(data = df3, aes(x = `BUILDING AGE`, y = log(`SALE PRICE`))) +
  geom_point(aes(col = df3$BOROUGH)) +
  geom_smooth(method = "lm") +
  theme(legend.position = "bottom") +
  ggtitle("Price of Oldest Properties sold in NYC", subtitle = "Oldest buildings in NYC in 2016") +
  scale_y_continuous("Property Value Distribution", labels = scales::dollar) +
  scale_x_continuous("Building Age")
```



Prediction Analysis

We will implement various methods get the to improve our prediction

We will create a new that will needed in our prediction dataset and remove all unwanted character type columns Also change Sale date into the month. Now split the data into training and test set in 80-20% ratio

```
library(caret)
# creating the new dataset

#transforming sale date into months
nyc_data$`SALE MONTH` <- as.factor(months(nyc_data$`SALE DATE`))

#removing all unwanted columns that are not needed for prediction
nyc_predtion <- nyc_data[, -c(3, 5, 6, 7, 8, 9, 10, 17, 19, 21)]
nyc_predtion$NEIGHBORHOOD <- as.factor(nyc_predtion$NEIGHBORHOOD)

nyc_predtion <- nyc_predtion[c(1:10, 12,13,11)]

str(nyc_predtion)
```

```
## tibble [58,470 x 13] (S3: tbl_df/tbl/data.frame)
##  $ BOROUGH                  : Factor w/ 5 levels "Manhattan","Bronx",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ NEIGHBORHOOD             : Factor w/ 253 levels "AIRPORT LA GUARDIA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ BUILDING CLASS CATEGORY  : Factor w/ 47 levels " CONDO-RENTALS",..: 34 34 34 34 33 33 20 20 20 20
```

```
##  $ ZIP CODE                    : Factor w/ 186 levels "0","10001","10002",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ RESIDENTIAL UNITS           : int [1:58470] 5 10 6 8 24 10 0 0 0 0 ...
##  $ COMMERCIAL UNITS            : int [1:58470] 0 0 0 0 0 0 0 0 0 0 ...
##  $ TOTAL UNITS                 : int [1:58470] 5 10 6 8 24 10 0 0 0 0 ...
##  $ LAND SQUARE FEET            : num [1:58470] 1633 2272 2369 1750 4489 ...
##  $ GROSS SQUARE FEET           : num [1:58470] 6440 6794 4615 4226 18523 ...
##  $ TAX CLASS AT TIME OF SALE: Factor w/ 4 levels "1","2","3","4": 2 2 2 2 2 2 2 2 2 2 ...
##  $ BUILDING AGE                : num [1:58470] 117 104 117 97 97 8 97 97 97 92 ...
##  $ SALE MONTH                  : Factor w/ 12 levels "April","August",..: 6 12 10 12 10 11 8 7 6 8 ...
##  $ SALE PRICE                  : num [1:58470] 6625000 3936272 8000000 3192840 16232000 ...
```

```r
# split the data into training nyc_pred_train and test set nyc_pred_test in 80-20% ratio
set.seed(101)
#index <- sample(nrow(nyc_predtion),nrow(nyc_predtion)*0.80)

index = createDataPartition(nyc_predtion$`SALE PRICE`, p = 0.80, list = FALSE)
nyc_pred_train <- nyc_predtion[index,]
nyc_pred_test <- nyc_predtion[-index,]
```

We will use all the columns/variabls that used in visualization analysis. We will use all the predictor columns and the sale price

Method 1: Correlation

Corealtion Table 1

We create corealtion and plot table below

```r
library(corrplot)
#compute correlation
distance_corr <- vapply(nyc_pred_train[c(1:12)],
                        function(x) { cor(nyc_pred_train$`SALE PRICE`, as.numeric(x), use = "pairwise.c
                        FUN.VALUE = numeric(1))
effect_corr <- vapply(distance_corr, function(x) { ifelse(x >= 0, "Positive", "Negative")},
                      FUN.VALUE = character(1))
#get the name
var_corr <- names(nyc_pred_train[c(1:12)])

# create correlation table
table1 <- data.frame(var_corr, abs(distance_corr), effect_corr)
table1 <- table1[order(-abs(distance_corr)),]
names(table1) <- c("Column/Variable", "Correlation Size", "Correlation Effect")
kable(table1)
```

	Column/Variable	Correlation Size	Correlation Effect
GROSS SQUARE FEET	GROSS SQUARE FEET	0.4770545	Positive
TOTAL UNITS	TOTAL UNITS	0.1857953	Positive
COMMERCIAL UNITS	COMMERCIAL UNITS	0.1673004	Positive
RESIDENTIAL UNITS	RESIDENTIAL UNITS	0.1557683	Positive
TAX CLASS AT TIME OF SALE	TAX CLASS AT TIME OF SALE	0.1081029	Positive
BOROUGH	BOROUGH	0.0758999	Negative
ZIP CODE	ZIP CODE	0.0691712	Negative

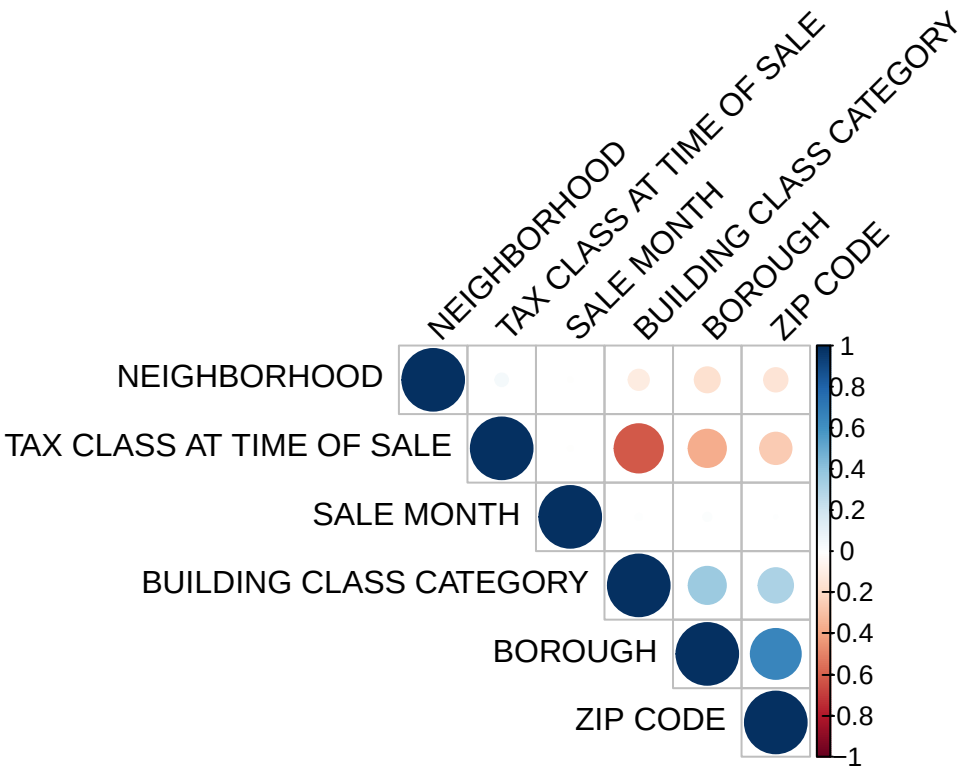
	Column/Variable	Correlation Size	Correlation Effect
LAND SQUARE FEET	LAND SQUARE FEET	0.0596938	Positive
BUILDING CLASS CATEGORY	BUILDING CLASS CATEGORY	0.0155359	Negative
BUILDING AGE	BUILDING AGE	0.0056948	Positive
NEIGHBORHOOD	NEIGHBORHOOD	0.0021094	Positive
SALE MONTH	SALE MONTH	0.0013021	Positive

From above correlation matrix table only variable Gross Square feet that we need to check the statistical significance

Corealtion Table 2

Now look at the correlation between the categorical predictors in the data chart and table below

```
#applying the TOTAL UNITS instead of RESIDENTIAL UNITS and COMMERCIAL UNITS. And LAND SQUARE FEET inste
temp <- nyc_predtion[,c(1:4,10,12)]
num <- c(1:6)
temp %<>% mutate_at(num, funs(as.numeric(.)))
#plot table of correlations
corrplot(cor(temp), type = "upper",order = "hclust",
         tl.col = "black", tl.srt = 45)
```



Now lets look at the table

```r
#plot table of correlations
round(cor(temp), 3)
```

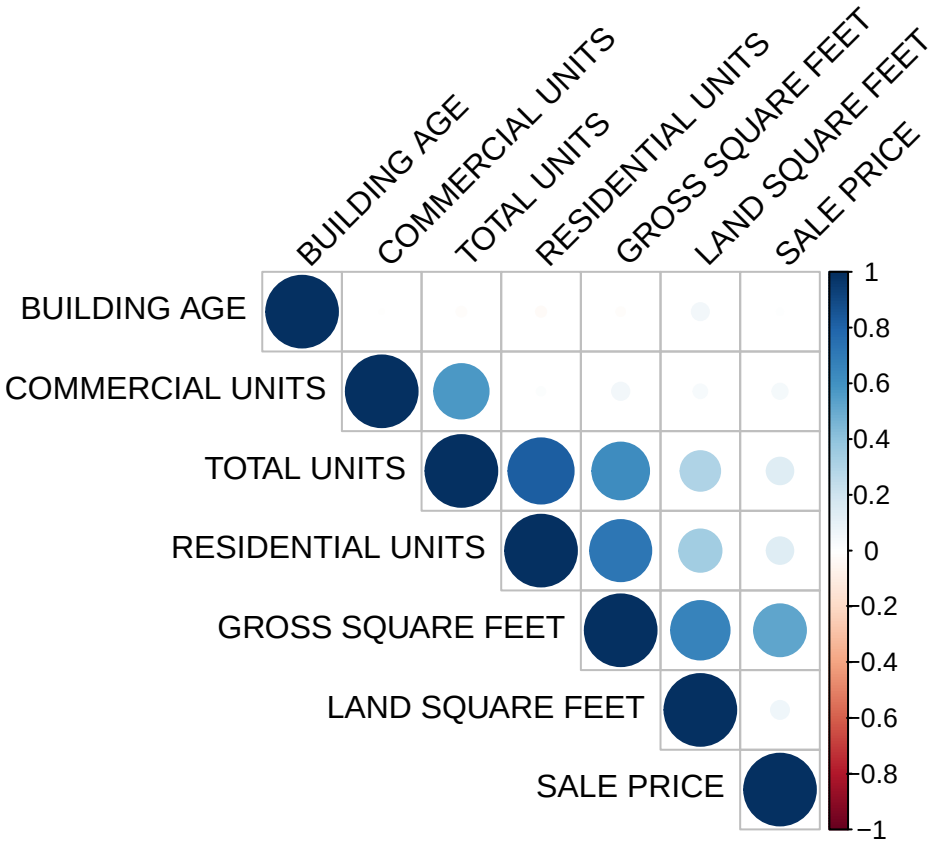
```
##                              BOROUGH NEIGHBORHOOD BUILDING CLASS CATEGORY ZIP CODE
## BOROUGH                        1.000       -0.165                   0.362    0.655
## NEIGHBORHOOD                  -0.165        1.000                  -0.108   -0.144
## BUILDING CLASS CATEGORY        0.362       -0.108                   1.000    0.312
## ZIP CODE                       0.655       -0.144                   0.312    1.000
## TAX CLASS AT TIME OF SALE     -0.365        0.042                  -0.613   -0.259
## SALE MONTH                     0.017       -0.005                   0.009    0.002
##                              TAX CLASS AT TIME OF SALE SALE MONTH
## BOROUGH                                         -0.365      0.017
## NEIGHBORHOOD                                     0.042     -0.005
## BUILDING CLASS CATEGORY                         -0.613      0.009
## ZIP CODE                                        -0.259      0.002
## TAX CLASS AT TIME OF SALE                        1.000     -0.006
## SALE MONTH                                      -0.006      1.000
```

From above here are the observation BOROUGH and ZIP CODE are highly correlated 0.65 BUILDING CLASS CATEGORY, TAX CLASS AT TIME OF SALE are highly correlated 0.613

Corealtion Table 3

Now look at the correlation between the numeric predictors in the data chart and table below

```r
#Correlate numeric predictors
correl <- cor(nyc_predtion[sapply(nyc_predtion, is.numeric)], use = "pairwise.complete.obs")
#plot the corelations chart
corrplot(correl, type = "upper",order = "AOE",
         tl.col = "black", tl.srt = 45)
```



Now lets look at the table

```
#plot table of correlations
round(correl, 3)
```

```
##                   RESIDENTIAL UNITS COMMERCIAL UNITS TOTAL UNITS
## RESIDENTIAL UNITS             1.000            0.014       0.825
## COMMERCIAL UNITS              0.014            1.000       0.576
## TOTAL UNITS                   0.825            0.576       1.000
## LAND SQUARE FEET              0.347            0.036       0.305
## GROSS SQUARE FEET             0.726            0.058       0.626
## BUILDING AGE                 -0.020           -0.004      -0.018
## SALE PRICE                    0.137            0.045       0.137
##                   LAND SQUARE FEET GROSS SQUARE FEET BUILDING AGE SALE PRICE
## RESIDENTIAL UNITS            0.347             0.726       -0.020      0.137
## COMMERCIAL UNITS             0.036             0.058       -0.004      0.045
## TOTAL UNITS                  0.305             0.626       -0.018      0.137
## LAND SQUARE FEET             1.000             0.664        0.056      0.063
## GROSS SQUARE FEET            0.664             1.000       -0.014      0.524
## BUILDING AGE                 0.056            -0.014        1.000      0.007
## SALE PRICE                   0.063             0.524        0.007      1.000
```

From above here are the observation COMMERCIAL UNITS and TOTAL UNITS are nearly correlated 0.577
RESIDENTIAL UNITS, TOTAL UNITS, GROSS SQUARE FEET are highly correlated approx >.7 LAND SQUARE FEET and GROSS SQUARE FEET are highly correlated 0.664 We can consider using TOTAL

UNITS instead of RESIDENTIAL UNITS and COMMERCIAL UNITS almost close to 90% of the data Same for LAND SQUARE FEET instead of LAND SQUARE FEET and GROSS SQUARE FEET.

Method 2: Single Linear Regressions

We will check here single factor linear regrection and find out how our predection working and if they are nessary. As we know there are many categorical variables running full regression doesnt seem correct as it will create too many dummy variables. We will maintain 4 dummy variable

We apply regresion on BOROUGH and SALE PRICE and check the summary and p-value

```
#apply single linear regressions SALE PRICE price with BOROUGH
#library(semEff)
slr_model <- lm(nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$BOROUGH)

reg <- summary(slr_model)

#F-statistic p-Value
print("p-value")
```

```
## [1] "p-value"
```

```
pf(reg$fstatistic[1],reg$fstatistic[2],reg$fstatistic[3],lower.tail = FALSE)
```

```
##        value
## 9.875196e-74
```

```
reg
```

```
##
## Call:
## lm(formula = nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$BOROUGH)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
##   -3397054    -804432    -384432     -28464 2206601771
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        3398230     118723   28.62   <2e-16 ***
## nyc_pred_train$BOROUGHBronx       -2602683     232638  -11.19   <2e-16 ***
## nyc_pred_train$BOROUGHBrooklyn    -2133797     164937  -12.94   <2e-16 ***
## nyc_pred_train$BOROUGHQueens      -2637478     158859  -16.60   <2e-16 ***
## nyc_pred_train$BOROUGHStaten Island -2854767     220312  -12.96   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12680000 on 46772 degrees of freedom
## Multiple R-squared:  0.007381,   Adjusted R-squared:  0.007296
## F-statistic: 86.95 on 4 and 46772 DF,  p-value: < 2.2e-16
```

```
#Adjusted R Squared
print("Adjusted r squared")
```

```
## [1] "Adjusted r squared"
```

```
reg$r.squared
```

```
## [1] 0.00738125
```

Borough is significant.

We apply regresion on NEIGHBORHOOD and SALE PRICE and check the summary and p-value

```
#apply single linear regressions SALE PRICE price with NEIGHBORHOOD

slr_model <- lm(nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$NEIGHBORHOOD)
#print summary
reg <- summary(slr_model)

#F-statistic p-Value
print("p-value")
```

```
## [1] "p-value"
```

```
pf(reg$fstatistic[1],reg$fstatistic[2],reg$fstatistic[3],lower.tail = FALSE)
```

```
##         value
## 6.174519e-156
```

```
#Adjusted R Squared
print("Adjusted r squared")
```

```
## [1] "Adjusted r squared"
```

```
reg$r.squared
```

```
## [1] 0.02920401
```

Only 4 neighborhoods are significant. We will create manually these 4 significant neighborhoods as dummy variable and put remaining under "all others" variable

```
nyc_pred_train$N_BLOOMFIELD = ifelse(nyc_pred_train$NEIGHBORHOOD == "BLOOMFIELD", 1,0)
nyc_pred_train$N_FASHION = ifelse(nyc_pred_train$NEIGHBORHOOD == "FASHION", 1,0)
nyc_pred_train$`N_JAVITS CENTER` = ifelse(nyc_pred_train$NEIGHBORHOOD == "JAVITS CENTER", 1,0)
nyc_pred_train$`N_MIDTOWN CBD` = ifelse(nyc_pred_train$NEIGHBORHOOD == "MIDTOWN CBD", 1,0)
nyc_pred_train$`N_OTHERS` = ifelse((nyc_pred_train$NEIGHBORHOOD != "MIDTOWN CBD") &
                                                    (nyc_pred_train$NEIGHBORHOOD != "JAVITS CENTER")
                                                    (nyc_pred_train$NEIGHBORHOOD != "FASHION") &
                                                    (nyc_pred_train$NEIGHBORHOOD != "BLOOMFIELD"), 1


# Removing the original Neighborhood predictor
nyc_pred_train <- nyc_pred_train[,-2]
```

We apply regresion on BUILDING CLASS CATEGORY and SALE PRICE and check the summary and p-value

```
#apply single linear regressions SALE PRICE price with BUILDING CLASS CATEGORY and p-value
slr_model <- lm(nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$`BUILDING CLASS CATEGORY`)
#print summary
reg <- summary(slr_model)

#F-statistic p-Value
print("p-value")
```

```
## [1] "p-value"
```

```
pf(reg$fstatistic[1],reg$fstatistic[2],reg$fstatistic[3],lower.tail = FALSE)
```

```
## value
##     0
```

```
#Adjusted R Squared
print("Adjusted r squared")
```

```
## [1] "Adjusted r squared"
```

```
reg$r.squared
```

```
## [1] 0.07759175
```

Only 8 of the 46 variables created are NOT significant. So, at this point we wii leave all the 46 dummy levels created for this variable.

We apply regresion on TAX CLASS AT TIME OF SALE and SALE PRICE and check the summary and p-value

```
#apply single linear regressions SALE PRICE price withTAX CLASS AT TIME OF SALE and p-value
slr_model <- lm(nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$`TAX CLASS AT TIME OF SALE`)
#print summary
reg <- summary(slr_model)
reg
```

```
##
## Call:
## lm(formula = nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$`TAX CLASS AT TIME OF SALE`)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
##   -8744227   -1011692    -349383      28117 2201254573
##
## Coefficients:
##                                          Estimate Std. Error t value
## (Intercept)                                744383      86613   8.594
## nyc_pred_train$`TAX CLASS AT TIME OF SALE`2   867309     119461   7.260
```

```
## nyc_pred_train$`TAX CLASS AT TIME OF SALE`4  8001044     297814  26.866
##                                                Pr(>|t|)
## (Intercept)                                     < 2e-16 ***
## nyc_pred_train$`TAX CLASS AT TIME OF SALE`2 3.93e-13 ***
## nyc_pred_train$`TAX CLASS AT TIME OF SALE`4  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12630000 on 46774 degrees of freedom
## Multiple R-squared:  0.01525,    Adjusted R-squared:  0.01521
## F-statistic: 362.2 on 2 and 46774 DF,  p-value: < 2.2e-16
```

```
#F-statistic p-Value
print("p-value")
```

```
## [1] "p-value"
```

```
pf(reg$fstatistic[1],reg$fstatistic[2],reg$fstatistic[3],lower.tail = FALSE)
```

```
##         value
## 7.869358e-157
```

```
#Adjusted R Squared
print("Adjusted r squared")
```

```
## [1] "Adjusted r squared"
```

```
reg$r.squared
```

```
## [1] 0.01525184
```

All the levels of Tax Class are significant hence we main maintaining them.

We apply regresion on SALE MONTH and SALE PRICE and check the summary and p-value

```
#apply single linear regressions SALE PRICE price with SALE MONTH and p-value
slr_model <- lm(nyc_pred_train$`SALE PRICE` ~ nyc_pred_train$`SALE MONTH`)
#print summary
reg <- summary(slr_model)

#F-statistic p-Value
print("p-value")
```

```
## [1] "p-value"
```

```
pf(reg$fstatistic[1],reg$fstatistic[2],reg$fstatistic[3],lower.tail = FALSE)
```

```
##      value
## 0.08842748
```

```
#Adjusted R Squared
print("Adjusted r squared")
```

```
## [1] "Adjusted r squared"
```

```
reg$r.squared
```

```
## [1] 0.000378706
```

None of the levels of the sales month variable are significant we ignore this variable

Method 3: Multi Linear Regressions

We will removd the SALE MONTH as this variable is not significant

```
attach(nyc_pred_train)
full_mlr_model <- lm(`SALE PRICE` ~ . -`SALE MONTH` , data = nyc_pred_train)
reg <- summary(full_mlr_model)
gl<- glance(full_mlr_model)
gl
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.s~1  sigma stati~2 p.value    df  logLik    AIC    BIC devia~3
##       <dbl>     <dbl>  <dbl>   <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>   <dbl>
## 1     0.893     0.892 5.84e6    856.       0   220 -3.85e5 7.71e5 7.73e5 7.65e17
## # ... with 2 more variables: df.residual <int>, nobs <int>, and abbreviated
## #   variable names 1: adj.r.squared, 2: statistic, 3: deviance
```

```
print("MSE")
```

```
## [1] "MSE"
```

```
mean(reg$residuals^2)
```

```
## [1] 3.37343e+13
```

We will ignore the varaible ZIP CODE as we a variable reduction process We have already have process the NEIGHBORHOOD so ZIP CODE.

From above table though Adjusted R-squared value is quite high at 0.8924342, the Model MSE is very large at 3.37343e+13 and AIC is very high

```
attach(nyc_pred_train)
full_mlr_model <- lm(`SALE PRICE` ~ . -`SALE MONTH` -`ZIP CODE`  , data = nyc_pred_train)
reg <- summary(full_mlr_model)


gl<- glance(full_mlr_model)
gl
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.s~1  sigma stati~2 p.value    df  logLik    AIC    BIC devia~3
##       <dbl>     <dbl>  <dbl>   <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>   <dbl>
## 1     0.696     0.695 9.83e6   1233.       0    42 -3.97e5 7.95e5 7.95e5 2.19e18
## # ... with 2 more variables: df.residual <int>, nobs <int>, and abbreviated
## #   variable names 1: adj.r.squared, 2: statistic, 3: deviance
```

```
print("MSE")
```

```
## [1] "MSE"
```

```
mean(reg$residuals^2)
```

```
## [1] 9.635337e+13
```

From above table though Adjusted R-squared value is reduce to 0.695746, though MSE is very large at 9.635337 and AIC is very high.

We need furter analysis and predictions using different methods and alogrithms. Which includes varable reductions, cross validations, random forest etc.

RESULT

We have clean the data, treated the missing values and identified the outliers. We have visualize the data and figure out the trend.

Here are what we have found from our analysis:

Property prices in NYC range from Dollar 220,000 (10% percentile of Property prices) all the way to 2.2 billion (95% percentile of Property prices).

Price per square footage in Manhattan is as high as Dollar 16,000/sqft, while in Bloomfield, Staten Island is $26/sqft.

Manhattan and Bronx sold the most residential condo apartments in large buildings/ residential societies, while Queens sold the most residential homes.

NYC has a place for everyone and you move to Staten Island with cheaper per sqft.

We need Further analyses.

CONCLUSION

We have applied correlation of the variables and applied regressions modals. We can conclude Single Linear Regression method is not appropriate as there are too many categorical predictors. Multi Linear Regression method the R-squared value is quite high and MSE is very large. We can continue to apply variable selection methods to get the optimal number of parameters to the modal. Apply cross-validation methods to test the out-of-sample error and try other algorithm modal like Random Forest, KNN to create optimal modal for predection

Moreover, this exploration show the relation between Sale Prices and the variables explored such as Borough, Neighborhood, Building Type, Tax Class, Building Age, Land/Gross Square Feet.

We need further analysis by applying Random Forest.A futher extension of this study will be using KNN imputation to fill in missing values and then evaluate the regression strength. But due to hardware constrain we have done analysis till MLR methods.

REFERENCES

Project taken from Kaggle site metioned in EDX link https://www.kaggle.com/code/annavictoria/ml-friendly-public-datasets/notebook?utm_medium=email&utm_source=intercom&utm_campaign=data+projects+onboarding

Project NYC Property Sales https://www.kaggle.com/new-york-city/nyc-property-sales

The dataset was downloaded from Kaggle www.kaggle.com/new-york-city/nyc-property-sales

For further reference on individual fields see the Glossary of Terms https://www.nyc.gov/assets/finance/downloads/pdf/07pdf/glossary_rsf071607.pdf.

For the building classification codes see the Building Classifications Glossary https://www.nyc.gov/assets/finance/jump/hlpbldgcode.html.