

# NYC House Prediction for Harvard X Individual Project

Elizabeth Peiwen Li

June 15 2020

- Part 1 Executive Summary
  - 1.1 Project Introduction
  - 1.2 Project Goals
- Part 2 Method & Analysis
  - 2.1 Dataset & Analysis Methods Summary
  - 2.2 Data Analysis
    - 2.2.1 Data Loading & Wrangling
    - 2.2.2 Generate Training and Testing Datasets
    - 2.2.3 Exploratory Data Analysis (EDA) On Training Dataset
    - 2.2.4 Data Exploratory Analysis Summary
  - 2.3 Data Modeling
    - 2.3.1 Linear Regression Model
    - 2.3.2 Logical Regression Model
    - 2.3.3 Regression Tree Model
    - 2.3.4 Model Ensemble
- Part 3 Model Results & Performance on Testing Dataset
- Part 4 Conclusions
  - 4.1 Lesson Learned
  - 4.2 Limitation & Future Work
- Glossary Of Terms

## Part 1 Executive Summary

### 1.1 Project Introduction

This project is to conduct data analysis and modeling on a dataset of real estate transaction deals within a year in New York property market. The dataset is originally from New York City Department of Finance, <https://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page> (<https://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page>). On their website, the Department of Finance's Rolling Sales files lists properties that sold in a twelve-month period from September 2016 to September 2017 in New York City for all tax classes. This dataset is a concatenated and slightly cleaned-up version provided in kaggle. The dataset contains both continuous and categorical data, including the sales prices, building locations, building types over 12 months periods. In this project, all variables have been analyzed and studied, and 4 data models have been introduced, studied and compared.

# 1.2 Project Goals

The objectives of this project is to build and compare with the models predicting property sales price in New York City, and find a better model suitable for real estate transaction price prediction.

## Part 2 Method & Analysis

### 2.1 Dataset & Analysis Methods Summary

To smoothly carry forward this project, there will be 3 major steps:

\*Step1: Download the dataset, explore the dataset and clean the data for further analysis.

\*Step2: Split the cleaned dataset into training and testing datasets. Conduct exploratory data analysis (EDA) on training dataset, learn more about each variable and its distribution.

\*Step3: Choose variables for data modeling, build data models on training dataset, and test the model on testing dataset. Root Mean Square Error (RMSE) will be used to analyze and compare the result of the models.

### 2.2 Data Analysis

#### 2.2.1 Data Loading & Wrangling

##### NYC Housing Sales Dataset Loading

```
nyc_property_original <- read_csv("nyc-rolling-sales.csv")
```

```
## Parsed with column specification:
## cols(
##   BOROUGH = col_double(),
##   NEIGHBORHOOD = col_character(),
##   BUILDING_CLASS_CATEGORY = col_character(),
##   TAX_CLASS_AT_PRESENT = col_character(),
##   EASE_MENT = col_logical(),
##   BUILDING_CLASS_AT_PRESENT = col_character(),
##   ADDRESS = col_character(),
##   APARTMENT_NUMBER = col_character(),
##   ZIP_CODE = col_double(),
##   RESIDENTIAL_UNITS = col_double(),
##   COMMERCIAL_UNITS = col_double(),
##   TOTAL_UNITS = col_double(),
##   LAND_SQUARE_FEET = col_character(),
##   GROSS_SQUARE_FEET = col_character(),
##   YEAR_BUILT = col_double(),
##   TAX_CLASS_AT_TIME_OF_SALE = col_double(),
##   BUILDING_CLASS_AT_TIME_OF_SALE = col_character(),
##   SALE_PRICE = col_character(),
##   SALE_DATE = col_character()
## )
```

```
nyc_property <- nyc_property_original
```

## Dataset Glimpse

```
glimpse(nyc_property)
```

```
## Rows: 84,548
## Columns: 19
## $ BOROUGH          <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ NEIGHBORHOOD      <chr> "ALPHABET CITY", "ALPHABET CITY", "ALPHABET
CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET
CITY", "ALPHABET CITY", "ALPH...
## $ BUILDING_CLASS_CATEGORY <chr> "07 RENTALS - WALKUP APARTMENTS", "07 RENTA
LS - WALKUP APARTMENTS", "07 RENTALS - WALKUP APARTMENTS", "07 RENTALS - WALKUP APART
MENTS", "07 RENTALS - WALKUP A...
## $ TAX_CLASS_AT_PRESENT <chr> "2A", "2", "2", "2B", "2A", "2", "2B", "2",
"2", "2", "2", "2B", "2", "2", "2", "2", "2", "2", "2", "2C", "2", "2", "2", "2"
, "2", "2", "2C", "2C", "2", ...
## $ EASE_MENT         <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N
A, NA, NA, NA, NA, NA, NA...
## $ BUILDING_CLASS_AT_PRESENT <chr> "C2", "C7", "C7", "C4", "C2", "C4", "C4", "
C7", "D5", "D9", "D7", "D1", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C
```

```

6", "C6", "C6", "C6", "C6", "C...
## $ ADDRESS <chr> "153 AVENUE B", "234 EAST 4TH STREET", "1
97 EAST 3RD STREET", "154 EAST 7TH STREET", "301 EAST 10TH STREET", "516 EAST 12T
H STREET", "210 AVENUE B", "...
## $ APARTMENT_NUMBER <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,
NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N
A, NA, NA, NA, NA, NA, NA, NA...
## $ ZIP_CODE <dbl> 10009, 10009, 10009, 10009, 10009, 10009, 1
0009, 10009, 10009, 10009, 10009, 10009, 10009, 10009, 10009, 10009, 10
009, 10009, 10009, 10009, 1000...
## $ RESIDENTIAL_UNITS <dbl> 5, 28, 16, 10, 6, 20, 8, 44, 15, 24, 30, 10
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ COMMERCIAL_UNITS <dbl> 0, 3, 1, 0, 0, 0, 0, 2, 0, 0, 4, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ TOTAL_UNITS <dbl> 5, 31, 17, 10, 6, 20, 8, 46, 15, 24, 34, 10
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ LAND_SQUARE_FEET <chr> "1633", "4616", "2212", "2272", "2369", "25
81", "1750", "5163", "1534", "4489", "4295", "3717", "-", "-", "-", "-", "-", "-",
 "-", "-", "-", "-", "4131", "-",...
## $ GROSS_SQUARE_FEET <chr> "6440", "18690", "7803", "6794", "4615", "9
730", "4226", "21007", "9198", "18523", "21328", "12350", "-", "-", "-", "-", "-",
 "-", "-", "-", "-", "16776"...
## $ YEAR_BUILT <dbl> 1900, 1900, 1900, 1913, 1900, 1900, 1920, 1
900, 1920, 1920, 1910, 2009, 1920, 1920, 1920, 1920, 1925, 1920, 1920, 1900, 19
02, 1928, 1928, 1928, 1928, 19...
## $ TAX_CLASS_AT_TIME_OF_SALE <dbl> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2
, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2...
## $ BUILDING_CLASS_AT_TIME_OF_SALE <chr> "C2", "C7", "C7", "C4", "C2", "C4", "C4", "
C7", "D5", "D9", "D7", "D1", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C
6", "C6", "C6", "C6", "C6", "C...
## $ SALE_PRICE <chr> "6625000", "-", "-", "3936272", "8000000",
 "-", "3192840", "-", "-", "16232000", "-", "10350000", "1", "499000", "10", "529500",
 "423000", "501000", "450000",...
## $ SALE_DATE <chr> "7/19/17 0:00", "12/14/16 0:00", "12/9/16 0
:00", "9/23/16 0:00", "11/17/16 0:00", "7/20/17 0:00", "9/23/16 0:00", "7/20/17 0:00"
, "6/20/17 0:00", "11/7/16 0:0...

```

BOROUGH: A digit code for the borough area. The codes are Manhattan (1), Bronx (2), Brooklyn (3), Queens (4), and Staten Island (5).

## Data Type Transformation

```
# After have a glimpse of the data, the hour and minute part was removed from the sale date column because they are all zero.
```

```
nyc_property$SALE_DATE <- gsub( " .*$", "", nyc_property$SALE_DATE )
```

```
# Variable data type change
```

```
nyc_property$BOROUGH <- as.character(nyc_property$BOROUGH)
nyc_property$ZIP_CODE <- as.character(nyc_property$ZIP_CODE)
nyc_property$TAX_CLASS_AT_TIME_OF_SALE <- as.character(nyc_property$TAX_CLASS_AT_TIME_OF_SALE)
nyc_property$SALE_PRICE <- as.numeric(nyc_property$SALE_PRICE)
```

```
## Warning: NAs introduced by coercion
```

```
nyc_property$LAND_SQUARE_FEET <- as.numeric(nyc_property$LAND_SQUARE_FEET)
```

```
## Warning: NAs introduced by coercion
```

```
nyc_property$GROSS_SQUARE_FEET <- as.numeric(nyc_property$GROSS_SQUARE_FEET)
```

```
## Warning: NAs introduced by coercion
```

```
nyc_property <- nyc_property%>%mutate_if(is.integer,as.numeric)
```

```
# Remove irrelevant variable
```

```
nyc_property$APARTMENT NUMBER<- NULL
```

```
nyc_property$ADDRESS <- NULL # Location is represented by other more informative variables like BOROUGH, NEIGHBORHOOD and ZIPCODE
```

```
glimpse(nyc_property)
```

```
## Rows: 84,548
```

```
## Columns: 17
```

[illegible]

```
## $ NEIGHBORHOOD      <chr> "ALPHABET CITY", "ALPHABET CITY", "ALPHABET  
CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET CITY", "ALPHABET  
CITY", "ALPHABET CITY", "ALPH..."
```

```
## $ BUILDING_CLASS_CATEGORY      <chr> "07 RENTALS - WALKUP APARTMENTS", "07 RENTA
LS - WALKUP APARTMENTS", "07 RENTALS - WALKUP APARTMENTS", "07 RENTALS - WALKUP APART
MENTS", "07 RENTALS - WALKUP A...
```

```
## $ TAX CLASS AT PRESENT      <chr> "2A", "2", "2", "2B", "2A", "2", "2B", "2",
```

```
"2", "2", "2", "2B", "2", "2", "2", "2", "2", "2", "2C", "2", "2", "2", "2",  
,"2", "2", "2C", "2C", "2", ...  
## $ EASE_MENT          <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,  
NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N  
A, NA, NA, NA, NA, NA, NA, NA...  
## $ BUILDING_CLASS_AT_PRESENT    <chr> "C2", "C7", "C7", "C4", "C2", "C4", "C4", "  
C7", "D5", "D9", "D7", "D1", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C  
6", "C6", "C6", "C6", "C6", "C...  
## $ ZIP_CODE            <chr> "10009", "10009", "10009", "10009", "10009"  
,"10009", "10009", "10009", "10009", "10009", "10009", "10009", "10009", "1  
0009", "10009", "10009", "1000...  
## $ RESIDENTIAL_UNITS      <dbl> 5, 28, 16, 10, 6, 20, 8, 44, 15, 24, 30, 10  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...  
## $ COMMERCIAL_UNITS       <dbl> 0, 3, 1, 0, 0, 0, 0, 2, 0, 0, 4, 0, 0, 0, 0  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...  
## $ TOTAL_UNITS           <dbl> 5, 31, 17, 10, 6, 20, 8, 46, 15, 24, 34, 10  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 24, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  
,"0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...  
## $ LAND_SQUARE_FEET       <dbl> 1633, 4616, 2212, 2272, 2369, 2581, 1750, 5  
163, 1534, 4489, 4295, 3717, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 4131, NA, NA, NA  
,"NA, NA, NA, NA, NA, NA, NA, NA, ...  
## $ GROSS_SQUARE_FEET     <dbl> 6440, 18690, 7803, 6794, 4615, 9730, 4226,  
21007, 9198, 18523, 21328, 12350, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 16776, NA,  
NA, NA, NA, NA, NA, NA, NA, NA...  
## $ YEAR_BUILT            <dbl> 1900, 1900, 1900, 1913, 1900, 1900, 1920, 1  
900, 1920, 1920, 1910, 2009, 1920, 1920, 1920, 1920, 1920, 1925, 1920, 1920, 1900, 19  
02, 1928, 1928, 1928, 1928, 19...  
## $ TAX_CLASS_AT_TIME_OF_SALE   <chr> "2", "2", "2", "2", "2", "2", "2", "2", "2"  
,"2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2", "2"  
,"2", "2", "2", "2", "2", "2"...  
## $ BUILDING_CLASS_AT_TIME_OF_SALE <chr> "C2", "C7", "C7", "C4", "C2", "C4", "C4", "  
C7", "D5", "D9", "D7", "D1", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C6", "C  
6", "C6", "C6", "C6", "C6", "C...  
## $ SALE_PRICE            <dbl> 6625000, NA, NA, 3936272, 8000000, NA, 3192  
840, NA, NA, 16232000, NA, 10350000, 1, 499000, 10, 529500, 423000, 501000, 450000, 5  
10000, NA, 350000, 11900000, 1...  
## $ SALE_DATE             <chr> "7/19/17", "12/14/16", "12/9/16", "9/23/16"  
,"11/17/16", "7/20/17", "9/23/16", "7/20/17", "6/20/17", "11/7/16", "7/20/17", "10/1  
7/16", "9/6/16", "3/10/17", "4...
```

[illegible][illegible][illegible][illegible][illegible][illegible]

```
## $ LAND_SQUARE_FEET      <dbl> 1633, 4616, 2212, 2272, 2369, 2581, 1750, 5
163, 1534, 4489, 4295, 3717, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 4131, NA, NA, NA
, NA, NA, NA, NA, NA, NA, NA, NA, ...
```

```
## $ GROSS_SQUARE_FEET      <dbl> 6440, 18690, 7803, 6794, 4615, 9730, 4226,
21007, 9198, 18523, 21328, 12350, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, 16776, NA,
NA, NA, NA, NA, NA, NA, NA, NA, NA...
```

```
## $ YEAR_BUILT      <dbl> 1900, 1900, 1900, 1913, 1900, 1900, 1920, 1900, 1920, 1920, 1910, 2009, 1920, 1920, 1920, 1920, 1925, 1920, 1920, 1900, 1902, 1928, 1928, 1928, 1928, 1928
```

[illegible][illegible]

```
## $ SALE_PRICE      <dbl> 6625000, NA, NA, 3936272, 8000000, NA, 3192
840, NA, NA, 16232000, NA, 10350000, 1, 499000, 10, 529500, 423000, 501000, 450000, 5
10000, NA, 350000, 11900000, 1
```

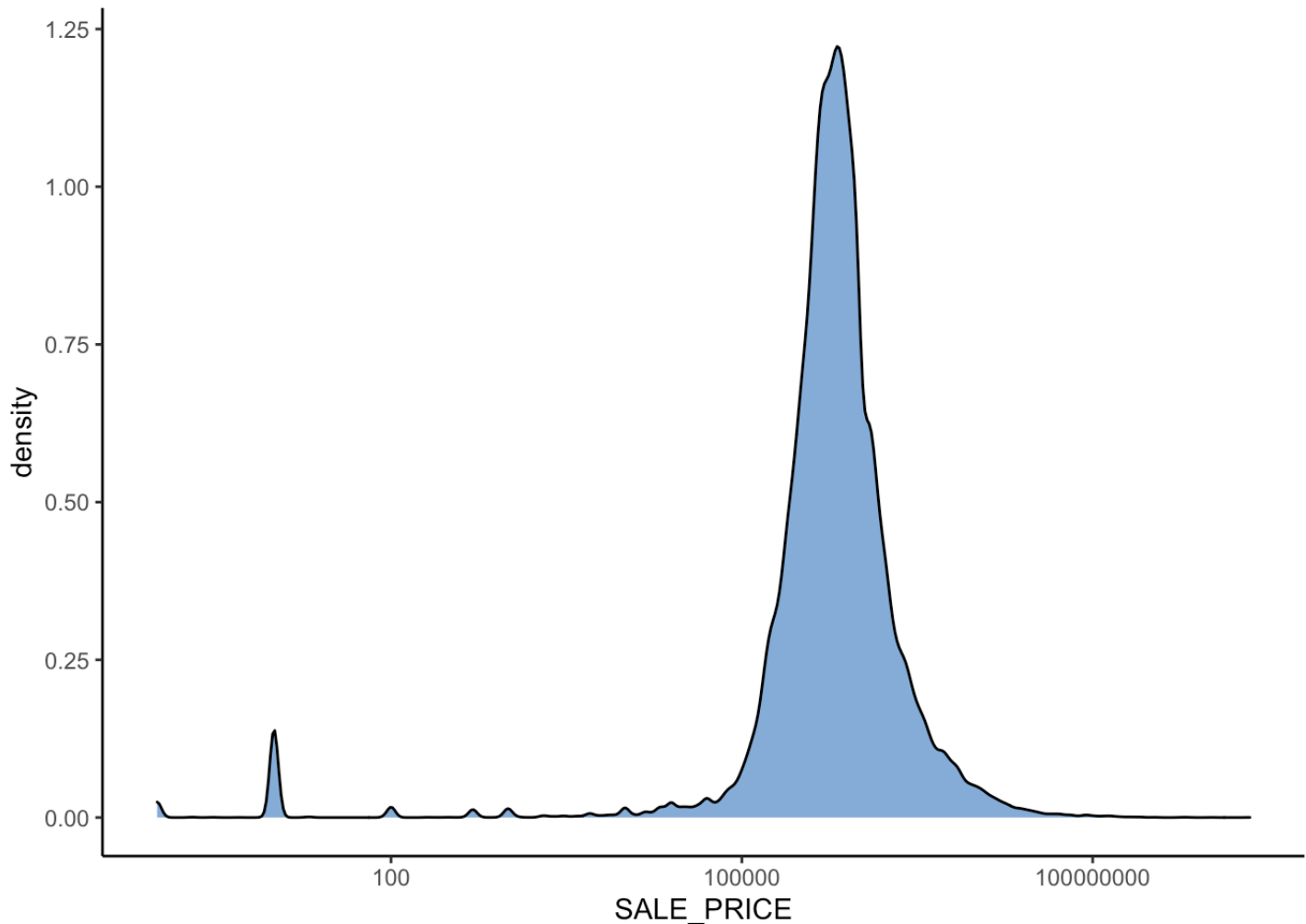
```
## $ SALE_DATE      <chr> "7/19/17", "12/14/16", "12/9/16", "9/23/16",
, "11/17/16", "7/20/17", "9/23/16", "7/20/17", "6/20/17", "11/7/16", "7/20/17", "10/1
7/16"  "9/6/16"  "3/10/17"  "4
```

## Outcome Variable Check - Sale Price

```
summary(nyc_property$SALE PRICE, na.rm = TRUE)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
##	0	225000	530000	1276456	950000	2210000000	14561

```
# Distribution plot on log scale
nyc_property%>%
  filter(SALE_PRICE> 0)%>%
  ggplot(aes(x =SALE_PRICE))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()
```



```
quantile(nyc_property$SALE_PRICE,na.rm = TRUE,0.995)
```

```
##      99.5%
## 20309325
```

Those outlier high price properties are skyscrapers, office building or a whole apartment building. While most of the extreme low property price (Many of those have a nonsensically small dollar amount: \$0 most commonly. These sales are actually transfers of deeds between parties: for example, parents transferring ownership to their home to a child after moving out for retirement.

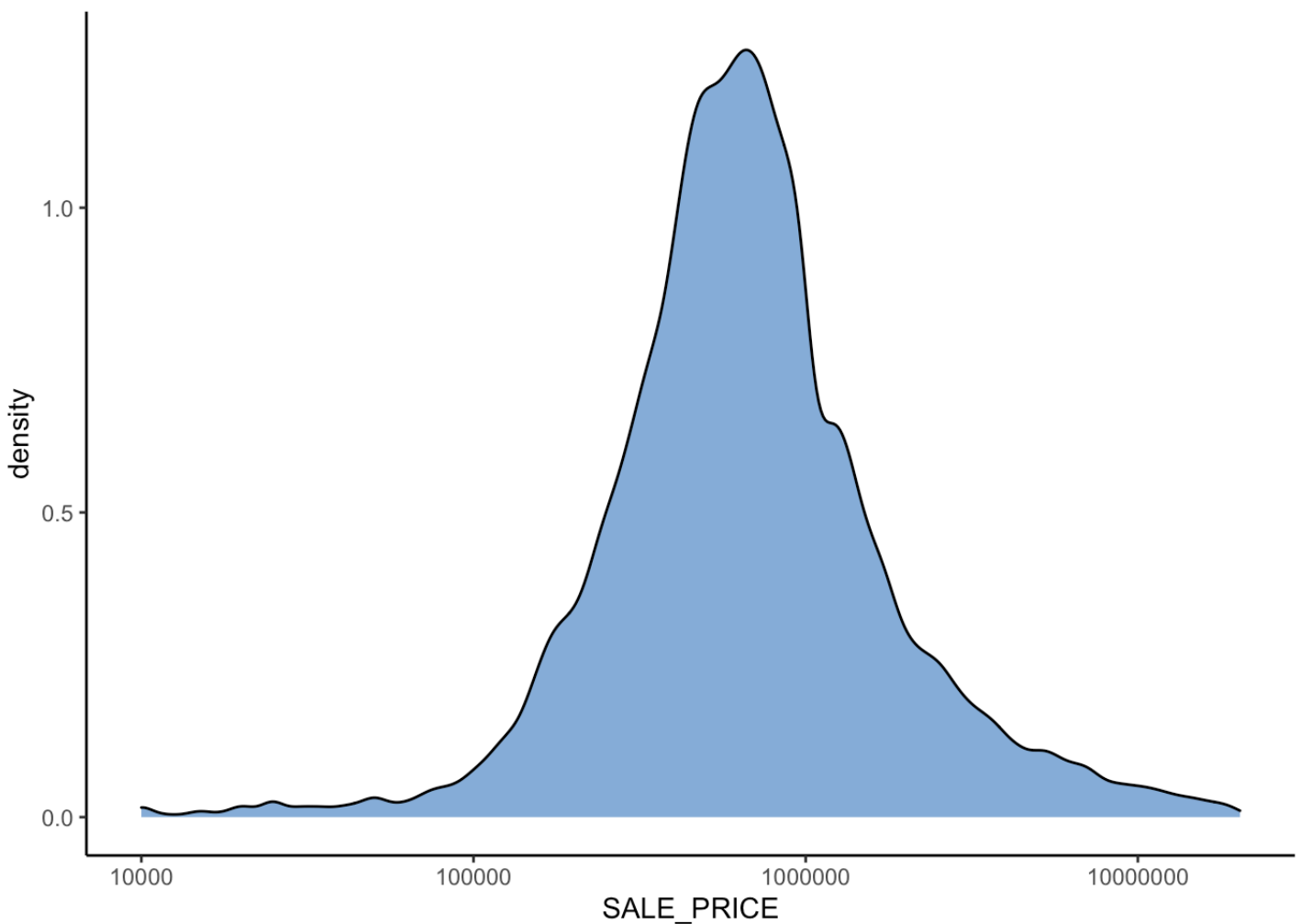
Records having sale price below than 10000 will be removed from the dataset because most of them are not fair market price. Only property with sale price within 99.5% quantile ( $\leq 20309325$ ) will be used in later analysis so that the independent variable is more balanced. Also, records which do not have sales price will

also be removed from the dataset because sale price is the outcome variable of this project.

## Data Cleaning - Remove Outliers Of Outcome Variables

```
# Remove outliers
nyc_property_subset <-
  nyc_property%>%
  filter(!is.na(SALE_PRICE) & SALE_PRICE>= 10000 & SALE_PRICE <= 20309325 )

# Distribution plot on log scale
nyc_property_subset%>%
  ggplot(aes(x=SALE_PRICE))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()
```



## Data Summary

### Categorical Variables



```

nonNumeric_variable <- nyc_property_subset %>% select_if(negate(is.numeric))
records_have_value = sapply(nonNumeric_variable, function(x) sum(!is.na(x)))
populated = round(records_have_value/nrow(nyc_property_subset),2)
unique_values = sapply(nonNumeric_variable,n_distinct)
calculate_mode <- function(x) {
  uniqx <- unique(na.omit(x))
  uniqx[which.max(tabulate(match(x, uniqx)))]
}

most_common_values = sapply(nonNumeric_variable,calculate_mode)
dataSummaryNonnum <- data.frame(records_have_value,populated,unique_values,most_commo
n_values)
dataSummaryNonnum

```

	records_have_value <int>	populated <dbl>	unique_values <int>	most_c <chr>
BOROUGH	58115	1.00	5	4
NEIGHBORHOOD	58115	1.00	253	FLUSH
BUILDING_CLASS_CATEGORY	58115	1.00	45	01 ONE
TAX_CLASS_AT_PRESENT	57523	0.99	10	2
EASE_MENT	0	0.00	1	NA
BUILDING_CLASS_AT_PRESENT	57523	0.99	144	D4
ZIP_CODE	58115	1.00	182	10314
TAX_CLASS_AT_TIME_OF_SALE	58115	1.00	3	2
BUILDING_CLASS_AT_TIME_OF_SALE	58115	1.00	144	D4
SALE_DATE	58115	1.00	335	6/29/17

1-10 of 10 rows

The table above shows that EASE\_MENT is a invalid column, and also suggests BUILDING\_CLASS\_AT\_PRESENT and BUILDING\_CLASS\_AT\_TIME\_OF\_SALE has the same number of unique value, therefore, we can check what is the relationship between the two variables.

## Remove Invalid Column - EASE\_MENT

```

nyc_property_subset$EASE_MENT <- NULL
nonNumeric_variable$EASE_MENT <- NULL

```

## Check BUILDING\_CLASS\_AT\_PRESENT and BUILDING\_CLASS\_AT\_TIME\_OF\_SALE

```
# Check if the two variables have the same unique value
unique(nyc_property_subset$BUILDING_CLASS_AT_PRESENT)[!(unique(nyc_property_subset$BUILDING_CLASS_AT_PRESENT) %in% unique(nyc_property_subset$BUILDING_CLASS_AT_TIME_OF_SALE))]
```

```
## [1] NA
```

```
# Check if the two variables are the same in each record
```

```
length(which(nyc_property_subset$BUILDING_CLASS_AT_PRESENT!= nyc_property_subset$BUILDING_CLASS_AT_TIME_OF_SALE))
```

```
## [1] 121
```

Out of 58115 records, only 121 properties BUILDING\_CLASS\_AT\_PRESENT is different from BUILDING\_CLASS\_AT\_TIME\_OF\_SALE. Thus, we can create a new variable called BUILDING\_CLASS\_STATUS\_CHANGE to demonstrate this status change instead of including both BUILDING\_CLASS\_AT\_PRESENT and BUILDING\_CLASS\_AT\_TIME\_OF\_SALE. I choose to keep BUILDING\_CLASS\_AT\_TIME\_OF\_SALE because it is fully populated and can better represent the property status at time of sale.

## Create BUILDING\_CLASS\_STATUS\_CHANGE

```
nyc_property_subset$BUILDING_CLASS_STATUS_CHANGE <- "0"
nyc_property_subset$BUILDING_CLASS_STATUS_CHANGE[which(nyc_property_subset$BUILDING_CLASS_AT_PRESENT!= nyc_property_subset$BUILDING_CLASS_AT_TIME_OF_SALE)] <- "1"
nyc_property_subset$BUILDING_CLASS_AT_PRESENT <- NULL
```

## Fill Missing Values

```
# Set NA Value to mode value for modeling purpose

nyc_property_subset$TAX_CLASS_AT_PRESENT[is.na(nyc_property_subset$TAX_CLASS_AT_PRESENT)] <- "2"

nonNumeric_variable <- nyc_property_subset %>% select_if(negate(is.numeric))
records_have_value = sapply(nonNumeric_variable, function(x) sum(!is.na(x)))
populated = round(records_have_value/nrow(nyc_property_subset),2)
unique_values = sapply(nonNumeric_variable,n_distinct)
calculate_mode <- function(x) {
  uniqx <- unique(na.omit(x))
  uniqx[which.max(tabulate(match(x, uniqx)))]
}

most_common_values = sapply(nonNumeric_variable,calculate_mode)
dataSummaryNonnum <- data.frame(records_have_value,populated,unique_values,most_common_values)
dataSummaryNonnum
```

	records_have_value	populated	unique_values	most_c
	<int>	<dbl>	<int>	<chr>
BOROUGH	58115	1	5	4
NEIGHBORHOOD	58115	1	253	FLUSH
BUILDING_CLASS_CATEGORY	58115	1	45	01 ONE
TAX_CLASS_AT_PRESENT	58115	1	9	2
ZIP_CODE	58115	1	182	10314
TAX_CLASS_AT_TIME_OF_SALE	58115	1	3	2
BUILDING_CLASS_AT_TIME_OF_SALE	58115	1	144	D4
SALE_DATE	58115	1	335	6/29/17
BUILDING_CLASS_STATUS_CHANGE	58115	1	2	0
9 rows				

## Numeric Variables

```

numeric_variable <- nyc_property_subset %>% select_if(is.numeric)
records_have_value = sapply(numeric_variable, function(x) sum(!is.na(x)))
populated = round(records_have_value/nrow(nyc_property_subset),2)
number_of_zero_value = sapply(numeric_variable,function(x) sum(x == 0,na.rm = TRUE))
minvalues <- sapply(numeric_variable,function(x) {min(x,na.rm = TRUE)})
meanvalues <- round(sapply(numeric_variable,function(x) {mean(x,na.rm = TRUE)}),2)
maxvalues <- sapply(numeric_variable,function(x) {max(x,na.rm = TRUE)})

dataSummaryNum <- data.frame(records_have_value,populated,number_of_zero_value,minvalues,meanvalues,maxvalues)
dataSummaryNum

```

	records_have_value <int>	populated <dbl>	number_of_zero_value <int>	minvalues <dbl>	meanvalues <dbl>
RESIDENTIAL_UNITS	58115	1.00	18286	0	111111
COMMERCIAL_UNITS	58115	1.00	55559	0	111111
TOTAL_UNITS	58115	1.00	16458	0	111111
LAND_SQUARE_FEET	37204	0.64	8049	0	111111
GROSS_SQUARE_FEET	36692	0.63	8489	0	111111
YEAR_BUILT	58115	1.00	4117	0	111111
SALE_PRICE	58115	1.00	0	10000	111111
7 rows					

The table above shows that there are missing values in LAND SQUARE FEET and GROSS SQUARE FEET, I will replace them with 0 as Unknown.

```

nyc_property_subset$LAND_SQUARE_FEET[is.na(nyc_property_subset$LAND_SQUARE_FEET)] <- 0

nyc_property_subset$GROSS_SQUARE_FEET[is.na(nyc_property_subset$GROSS_SQUARE_FEET)] <- 0

nyc_property_subset$YEAR_BUILT[is.na(nyc_property_subset$YEAR_BUILT)] <- 0

```

## 2.2.2 Generate Training and Testing Datasets

### Training/Testing Data Split

```
spec = c(train = .75, test = .25)

set.seed(1)

g = sample(cut(
  seq(nrow(nyc_property_subset)),
  nrow(nyc_property_subset)*cumsum(c(0,spec)),
  labels = names(spec)
))

res = split(nyc_property_subset, g)

Training <- res$train
Testing <- res$test
```

```
str(Training)
```

```
## tibble [43,586 × 16] (S3: tbl_df/tbl/data.frame)
## $ BOROUGH : chr [1:43586] "1" "1" "1" "1" ...
## $ NEIGHBORHOOD : chr [1:43586] "ALPHABET CITY" "ALPHABET CITY" "
ALPHABET CITY" "ALPHABET CITY" ...
## $ BUILDING_CLASS_CATEGORY : chr [1:43586] "07 RENTALS - WALKUP APARTMENTS"
"07 RENTALS - WALKUP APARTMENTS" "07 RENTALS - WALKUP APARTMENTS" "08 RENTALS - ELEVA
TOR APARTMENTS" ...
## $ TAX_CLASS_AT_PRESENT : chr [1:43586] "2A" "2B" "2A" "2" ...
## $ ZIP_CODE : chr [1:43586] "10009" "10009" "10009" "10009" .
..
## $ RESIDENTIAL_UNITS : num [1:43586] 5 10 6 24 0 0 0 0 0 0 ...
## $ COMMERCIAL_UNITS : num [1:43586] 0 0 0 0 0 0 0 0 0 0 ...
## $ TOTAL_UNITS : num [1:43586] 5 10 6 24 0 0 0 0 0 0 ...
## $ LAND_SQUARE_FEET : num [1:43586] 1633 2272 2369 4489 0 ...
## $ GROSS_SQUARE_FEET : num [1:43586] 6440 6794 4615 18523 0 ...
## $ YEAR_BUILT : num [1:43586] 1900 1913 1900 1920 1920 ...
## $ TAX_CLASS_AT_TIME_OF_SALE : chr [1:43586] "2" "2" "2" "2" ...
## $ BUILDING_CLASS_AT_TIME_OF_SALE: chr [1:43586] "C2" "C4" "C2" "D9" ...
## $ SALE_PRICE : num [1:43586] 6625000 3936272 8000000 16232000
529500 ...
## $ SALE_DATE : chr [1:43586] "7/19/17" "9/23/16" "11/17/16" "1
1/7/16" ...
## $ BUILDING_CLASS_STATUS_CHANGE : chr [1:43586] "0" "0" "0" "0" ...
```

```
nrow(Testing)
```

```
## [1] 14529
```

## Training/Testing Data Level Check

*# For variable levels that are in testing dataset and not in training dataset, I will set it's value to mode*

```
prepTest <- function(x)
{
  for (i in unique(Testing[[x]]))
  {
    if(!(i%in% unique(Training[[x]])))
    {
      return(x)
    }
  }
}

uniqueTest <- unlist(lapply(colnames(nonNumeric_variable),prepTest))

Testing$NEIGHBORHOOD[!(Testing$NEIGHBORHOOD %in% unique(Training$NEIGHBORHOOD))] <- "
FLUSHING-NORTH"

Testing$BUILDING_CLASS_CATEGORY[!(Testing$BUILDING_CLASS_CATEGORY %in% unique(Trainin
g$BUILDING_CLASS_CATEGORY))] <- "01 ONE FAMILY DWELLINGS"

Testing$ZIP_CODE[!(Testing$ZIP_CODE %in% unique(Training$ZIP_CODE))] <- "10314"

Testing$BUILDING_CLASS_AT_TIME_OF_SALE[!(Testing$BUILDING_CLASS_AT_TIME_OF_SALE %in%
unique(Training$BUILDING_CLASS_AT_TIME_OF_SALE))] <- "D4"
```

## 2.2.3 Exploratory Data Analysis (EDA) On Training Dataset

### Categorical Variable Distributions

#### Borough

Manhattan (1), Bronx (2), Brooklyn (3), Queens (4), Staten Island (5)

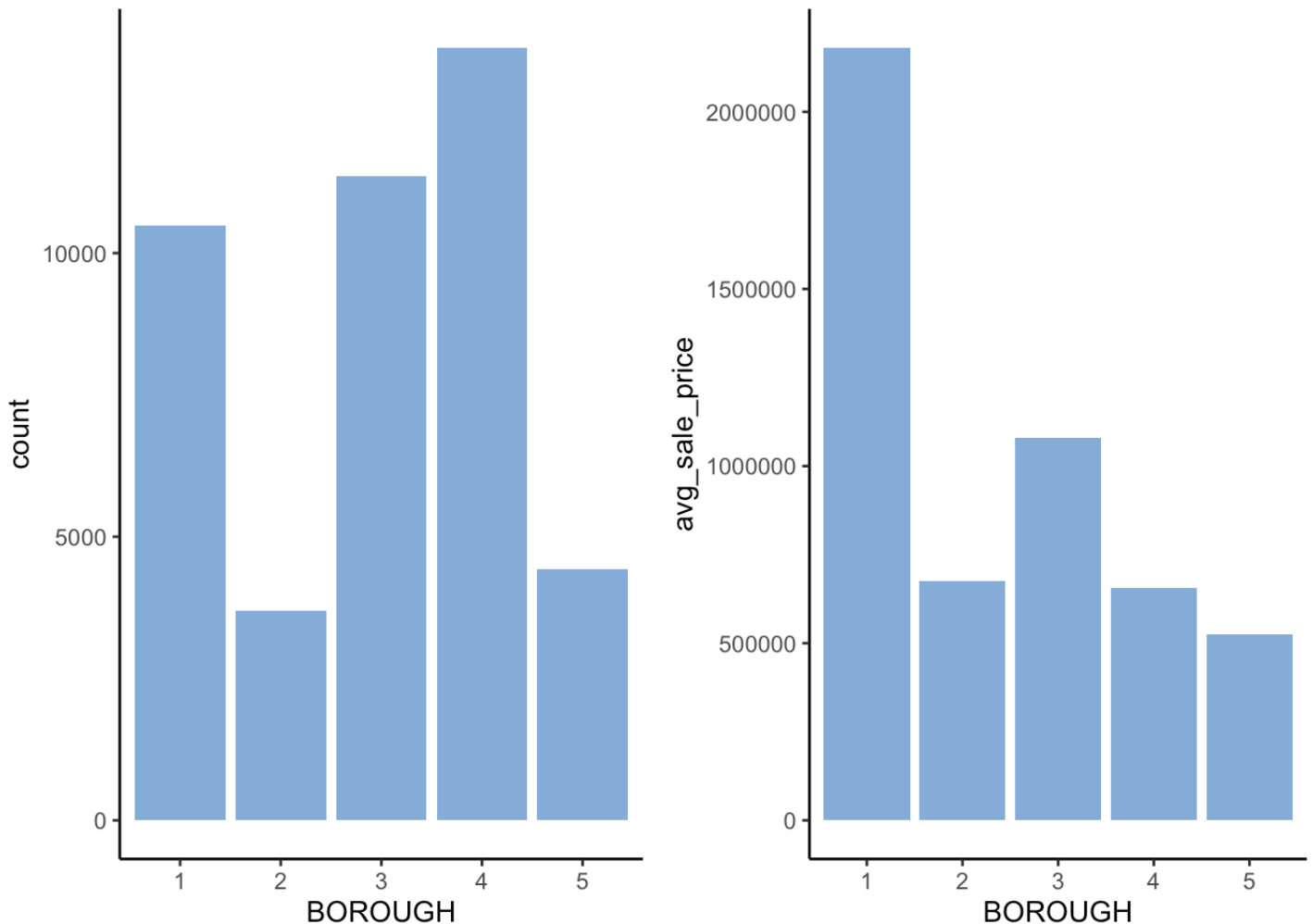
```

histPlot <- Training%>%
  filter(!is.na(BOROUGH))%>%
  group_by(BOROUGH)%>%
  ggplot(aes(x = BOROUGH))+
  geom_bar(stat = "count",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

corrPlot <- Training%>%
  filter(!is.na(BOROUGH))%>%
  group_by(BOROUGH)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x = BOROUGH,y = avg_sale_price))+
  geom_bar(stat = "identity",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

plot_grid(histPlot,corrPlot,nrow = 1)

```



The above plots shows that Borough 1,3 and 4 have the most of the sales, and location has an impact on the property sale price, borough 1 (Manhattan's average house price is way higher than other borough areas).

## NEIGHBORHOOD

```
Training%>%
  group_by(NEIGHBORHOOD)%>%
  summarise(quantity = length(NEIGHBORHOOD),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(quantity))
```

NEIGHBORHOOD	quantity	avg_sale_price
<chr>	<int>	<dbl>
FLUSHING-NORTH	1662	913703.9
UPPER EAST SIDE (59-79)	1073	2064445.1
UPPER EAST SIDE (79-96)	993	2250957.0
UPPER WEST SIDE (59-79)	794	2099253.6
MIDTOWN EAST	775	1537399.6
FOREST HILLS	660	514134.5
BAYSIDE	649	665426.9
BEDFORD STUYVESANT	646	1283527.2
JACKSON HEIGHTS	547	536293.1
EAST NEW YORK	502	588672.8

1-10 of 251 rows

Previous123456...26Next

From above, we can learn that, the best selling neighborhoods (Top5) are Flushing-North, Upper East Side (59-79,79-96),Upper West Side (59-79), and Midtown East.

```
Training%>%
  group_by(NEIGHBORHOOD)%>%
  summarise(quantity = length(NEIGHBORHOOD),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(avg_sale_price))
```

NEIGHBORHOOD	quantity	avg_sale_price
<chr>	<int>	<dbl>
EAST RIVER	1	11200000.0
CIVIC CENTER	210	5766352.6
BLOOMFIELD	2	5282500.0
LITTLE ITALY	20	5016468.8
SOHO	175	3734985.8
TRIBECA	404	3682138.1



FLATIRON	288	3593098.2
GREENWICH VILLAGE-WEST	330	3426342.2
FASHION	46	3229798.9
CHINATOWN	42	2831236.5
1-10 of 251 rows		
Previous 1 2 3 4 5 6 ... 26 Next		

From above, we can learn that, the neighborhood which have highest average sales prices (Top5) are East River, Civic Center, Bloomfield, Little Italy and Soho.

## BUILDING CLASS CATEGORY

```
Training%>%
  group_by(BUILDING_CLASS_CATEGORY)%>%
  summarise(quantity = length(BUILDING_CLASS_CATEGORY),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(quantity))
```

BUILDING_CLASS_CATEGORY	quantity	avg_sale_price
<chr>	<int>	<dbl>
01 ONE FAMILY DWELLINGS	9506	655937.49
10 COOPS - ELEVATOR APARTMENTS	8625	778963.47
13 CONDOS - ELEVATOR APARTMENTS	7659	1910343.66
02 TWO FAMILY DWELLINGS	7404	805529.69
09 COOPS - WALKUP APARTMENTS	1882	481579.57
03 THREE FAMILY DWELLINGS	1714	990753.72
07 RENTALS - WALKUP APARTMENTS	1285	2932078.50
04 TAX CLASS 1 CONDOS	948	552670.03
17 CONDO COOPS	808	919448.00
15 CONDOS - 2-10 UNIT RESIDENTIAL	777	1523221.15
1-10 of 43 rows		
Previous 1 2 3 4 5 Next		

From above, we can learn that more common building classes sold in Borough are one family dwellings, Coops- Elevator Apartments, Condo - Elevator Apartment, and two family dwellings.

## Tax Class At Present

Every property in the city is assigned to one of four tax classes (Classes 1, 2, 3, and 4),based on the use of the property.

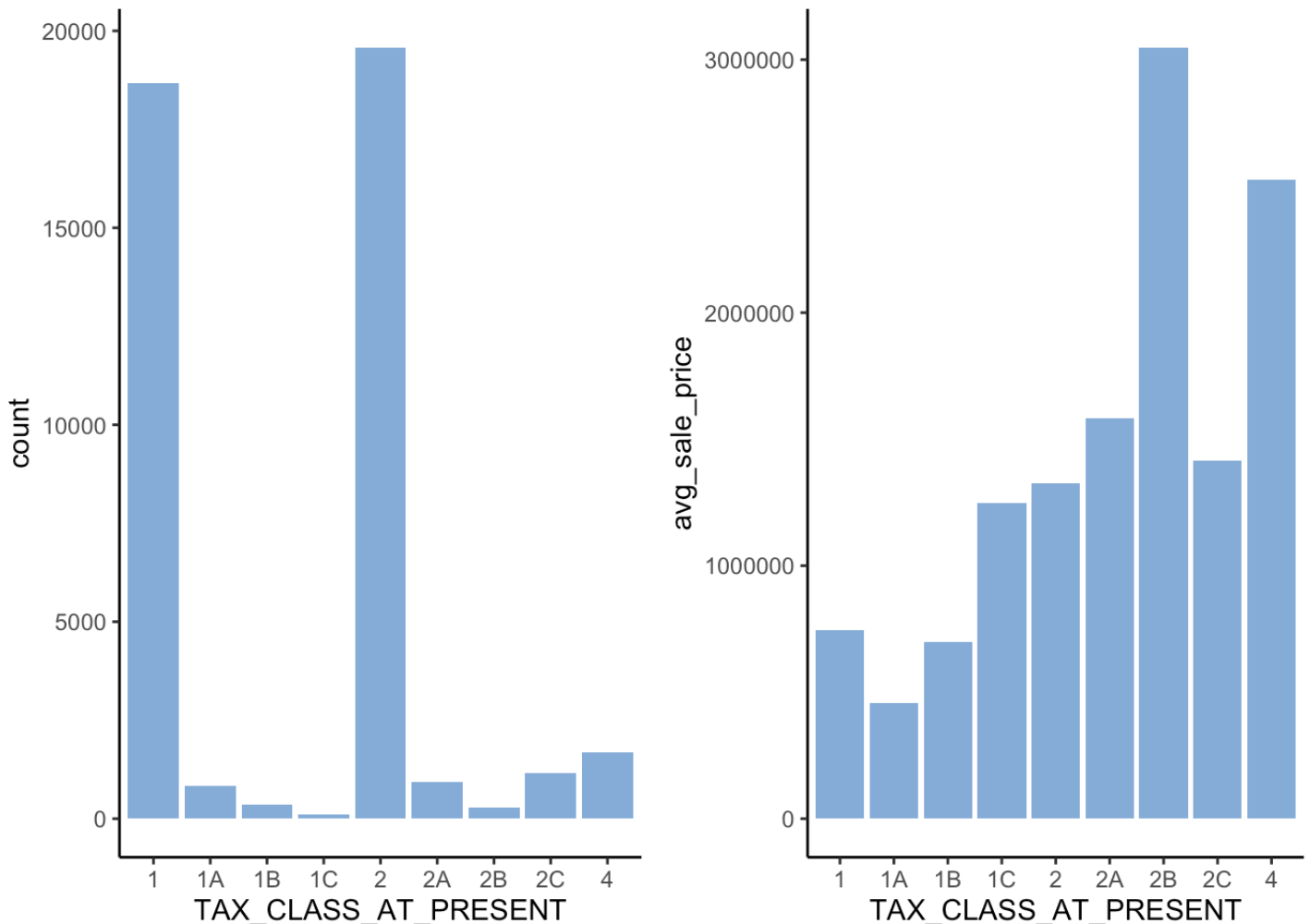
```

histPlot <- Training%>%
  filter(!is.na(TAX_CLASS_AT_PRESENT))%>%
  group_by(TAX_CLASS_AT_PRESENT)%>%
  ggplot(aes(x = TAX_CLASS_AT_PRESENT))+
  geom_bar(stat = "count",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

corrPlot <- Training%>%
  filter(!is.na(TAX_CLASS_AT_PRESENT))%>%
  group_by(TAX_CLASS_AT_PRESENT)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x = TAX_CLASS_AT_PRESENT,y = avg_sale_price))+
  geom_bar(stat = "identity",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

plot_grid(histPlot,corrPlot,nrow = 1)

```



The majority of sold properties are in Class 1 and class 2. And the average sales price in Class 2B is the highest. The average sales price in Class 2B is the second highest.

**Zip Code**

```

Training%>%
  group_by(ZIP_CODE)%>%
  summarise(quantity = length(ZIP_CODE),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(quantity))

```

ZIP_CODE <chr>	quantity <int>	avg_sale_price <dbl>
11354	857	929676.1
10314	854	563299.1
11201	767	1846833.7
11375	688	519717.5
10011	637	2808479.5
10016	609	1464051.9
10312	604	526947.6
11235	602	656385.3
10023	590	1774946.6
10022	576	1987169.9
1-10 of 181 rows	Previous 1 2 3 4 5 6 ... 19 Next	

Among 181 zip codes in Borough NewYork,the top 5 selling zip code are 11354,10314,11201,11375 and 10011.

```

Training%>%
  group_by(ZIP_CODE)%>%
  summarise(quantity = length(ZIP_CODE),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(avg_sale_price))%>%head()

```

ZIP_CODE <chr>	quantity <int>	avg_sale_price <dbl>
10013	379	5314647
10007	207	4605301
10018	31	3314121
10069	84	3096248
10037	45	3055746
10006	121	2985595

6 rows

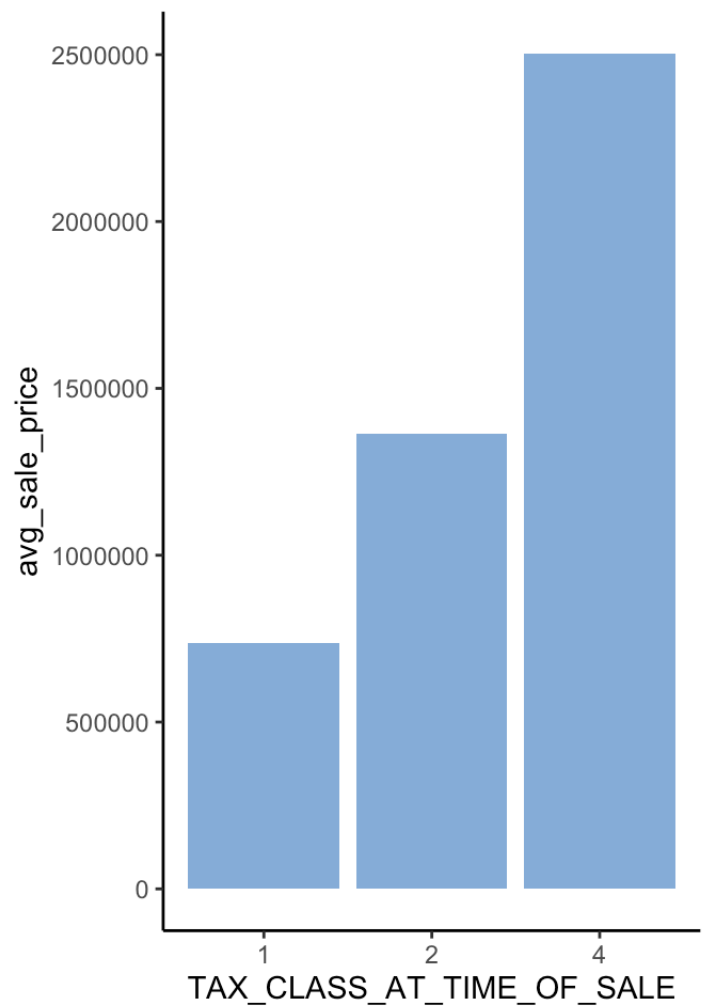
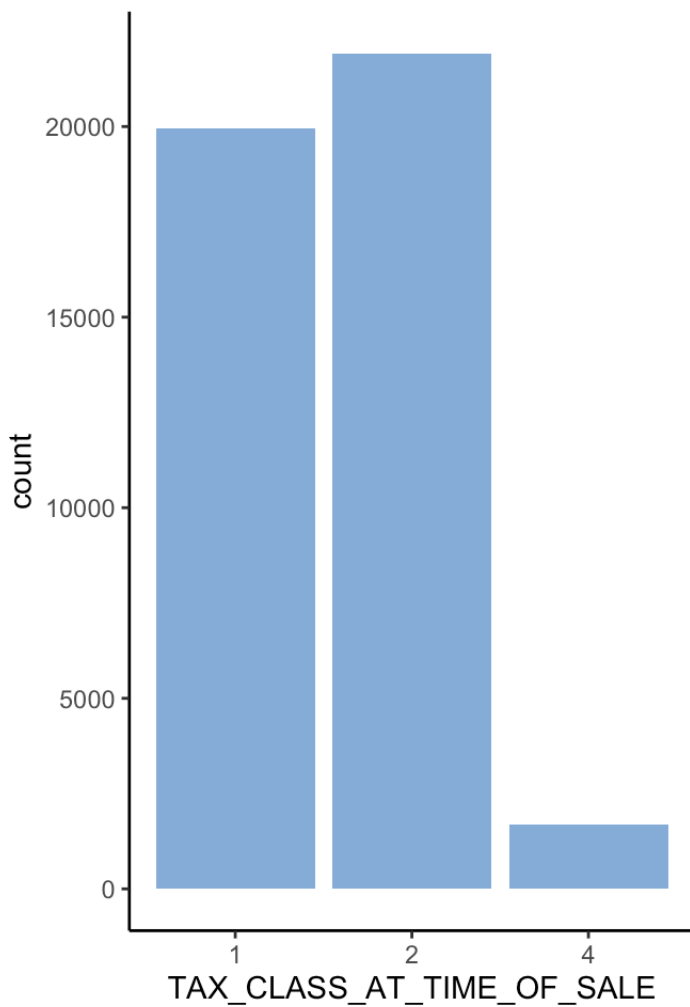
Among 181 zip codes in Borough NewYork,the top 5 average sales price zip code are 10013,10007,10018,10069,10037. And the highest average sales price is \$5314647.

## Tax Class At Time Of Sale

```
histPlot <- Training%>%
  filter(!is.na(TAX_CLASS_AT_TIME_OF_SALE))%>%
  group_by(TAX_CLASS_AT_TIME_OF_SALE)%>%
  ggplot(aes(x = TAX_CLASS_AT_TIME_OF_SALE))+
  geom_bar(stat = "count",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

corrPlot <- Training%>%
  filter(!is.na(TAX_CLASS_AT_TIME_OF_SALE))%>%
  group_by(TAX_CLASS_AT_TIME_OF_SALE)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x = TAX_CLASS_AT_TIME_OF_SALE,y = avg_sale_price))+
  geom_bar(stat = "identity",fill = "#6699cc",alpha = 0.8)+
  theme_classic()

plot_grid(histPlot,corrPlot,nrow = 1)
```



There are 4 different Tax Class At Time Of Sale value, but the Tax Class At Time Of Sale of three only has 4 record out of 84548 observations in the original dataset, and has been removed when deleting sale price outliers. Most tax class are class 1 and 2, however class 4 has the highest average sales price.

## BUILDING CLASS AT TIME OF SALE

```
Training%>%
  group_by(BUILDING_CLASS_AT_TIME_OF_SALE)%>%
  summarise(quantity = length(BUILDING_CLASS_AT_TIME_OF_SALE),
            avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(quantity))
```

BUILDING_CLASS_AT_TIME_OF_SALE <chr>	quantity <int>	avg_sale_price <dbl>
D4	8464	754882.44
R4	7659	1910343.66
A1	3545	631644.58
A5	3064	549379.92
B2	2402	694018.44

B1	2073	837929.12
B3	1902	778699.21
C6	1878	477884.85
C0	1714	990753.72
A2	1455	557772.27
1-10 of 132 rows		
Previous 1 2 3 4 5 6 ... 14 Next		

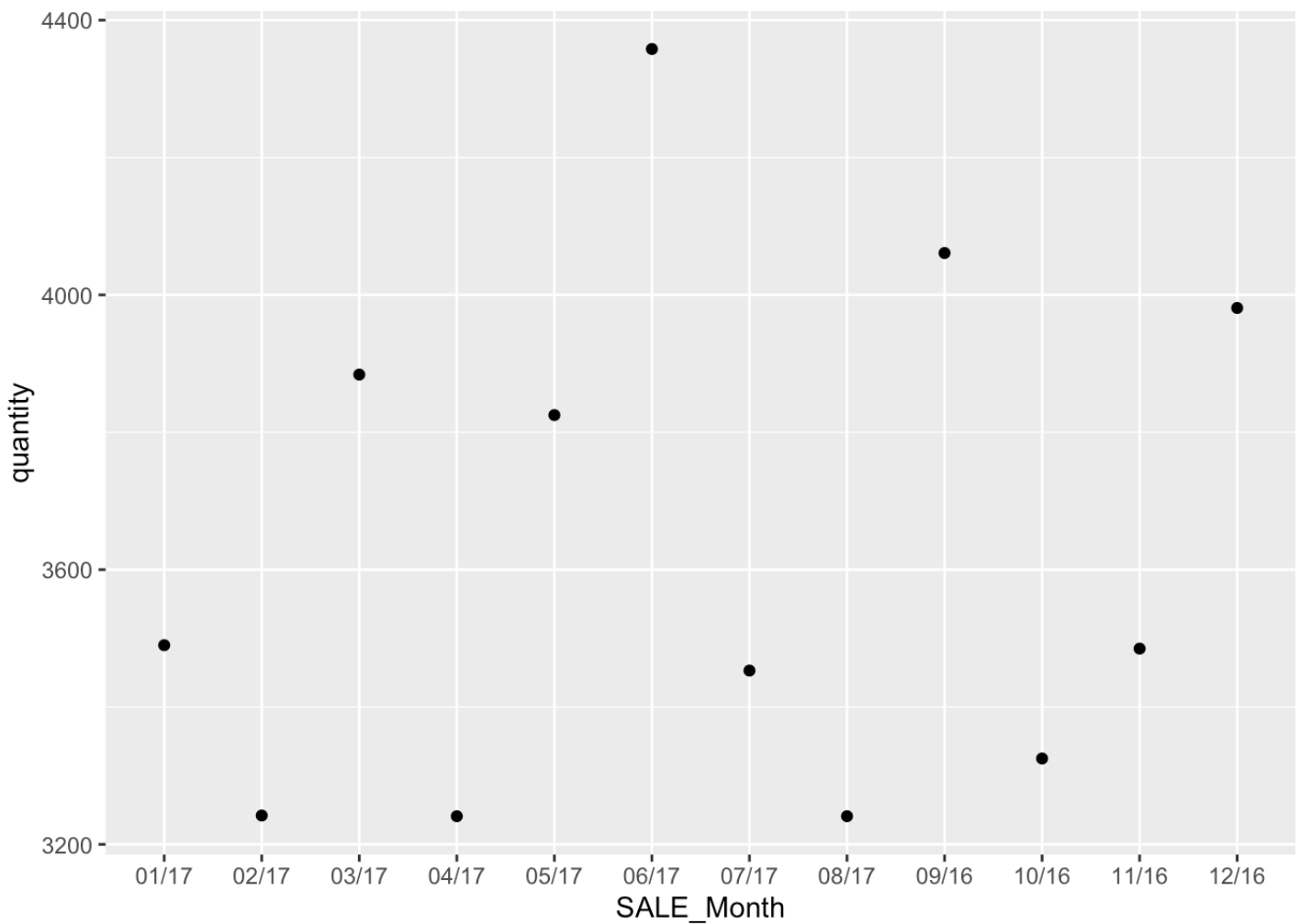
Building Class D4, R4, A1, A5 and B2 are the building classes with the top sales.

### SALE Amount and Price across the time

```
Training$SALE_DATE <- as.Date(Training$SALE_DATE, format = "%m/%d/%y")
Training$SALE_YEAR <- format(Training$SALE_DATE,"%y")
Training$SALE_Month <- format(Training$SALE_DATE,"%m/%y")
Training%>%
  group_by(SALE_YEAR)%>%
  summarise(quantity = length(SALE_YEAR),avg_sale_price = mean(SALE_PRICE))%>%
  arrange(desc(quantity))
```

SALE_YEAR	quantity	avg_sale_price
<chr>	<int>	<dbl>
17	28734	1142520
16	14852	1078850
2 rows		

```
Training%>%
  group_by(SALE_Month)%>%
  summarise(quantity = length(SALE_Month),avg_sale_price = mean(SALE_PRICE))%>%as.d
ata.frame()%>%ggplot(aes(x=SALE_Month,y = quantity))+geom_point()
```



From above, we can learn that the last month of every quarter had the highest sales amounts, maybe because of the pressure of the sales target.

There's no obvious seasonal trend. We can delete it from the training set in later analysis.

```
Training$SALE_Month <- NULL
Training$SALE_YEAR <- NULL
Training$SALE_DATE <- NULL
Testing$SALE_DATE <- NULL
```

## Numeric Variable Distributions

### RESIDENTIAL UNITS

```
summary(Training$RESIDENTIAL_UNITS)
```

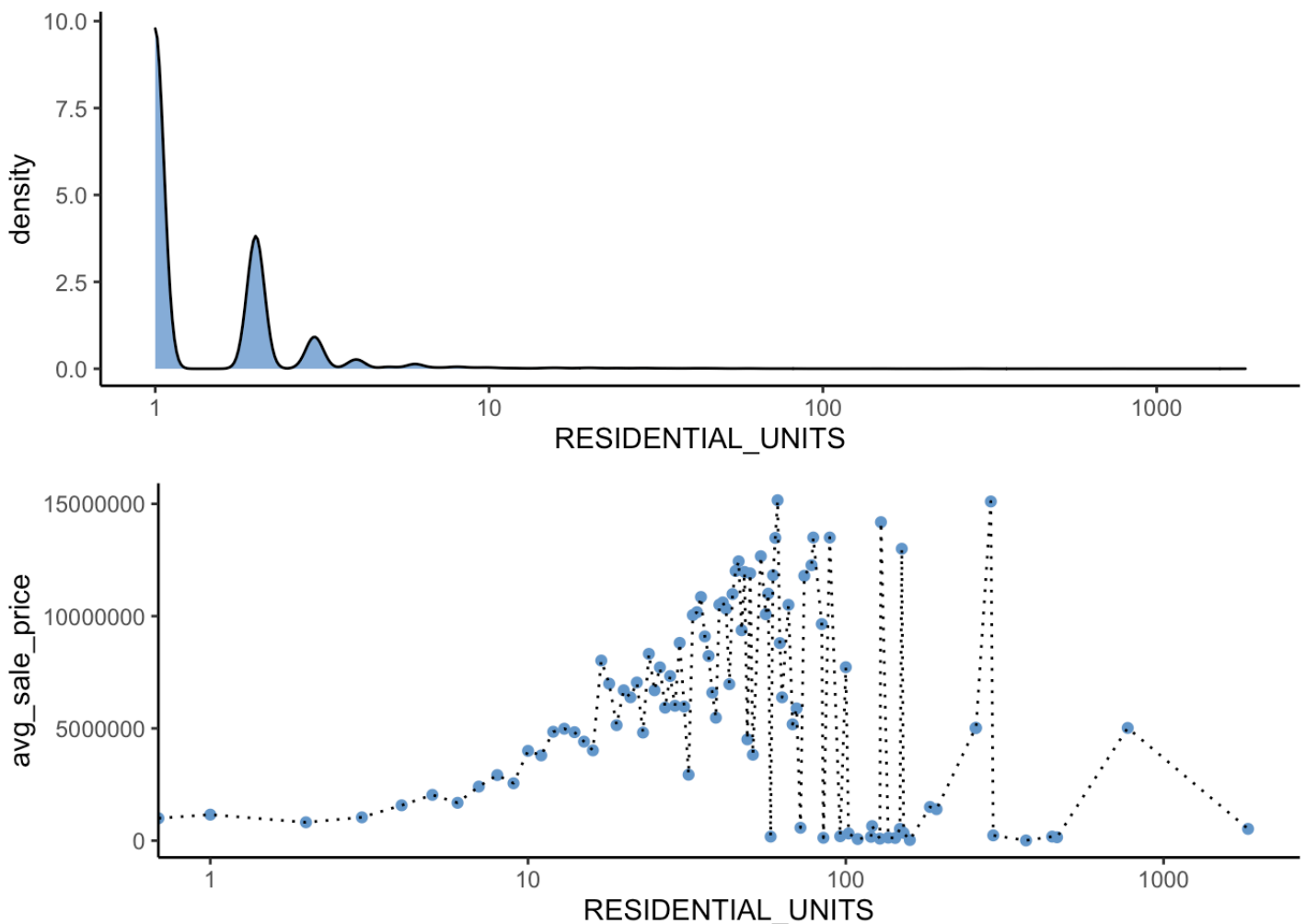
##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.000	0.000	1.000	1.527	1.000	1844.000

```

histPlot <- Training%>%
  filter(!is.na(RESIDENTIAL_UNITS))%>%
  ggplot(aes(x=RESIDENTIAL_UNITS))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()

corrPlot <- Training%>%
  group_by(RESIDENTIAL_UNITS)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x= RESIDENTIAL_UNITS, y = avg_sale_price ))+
  geom_point(color = "#6699cc")+
  geom_line(aes(group = 1),linetype = 'dotted')+
  theme_classic()+
  scale_x_log10()
plot_grid(histPlot,corrPlot,nrow = 2)

```



The plot above shows the residential units has a positive relationship with sale price when the number of units is lower than 100.

## COMMERCIAL UNITS



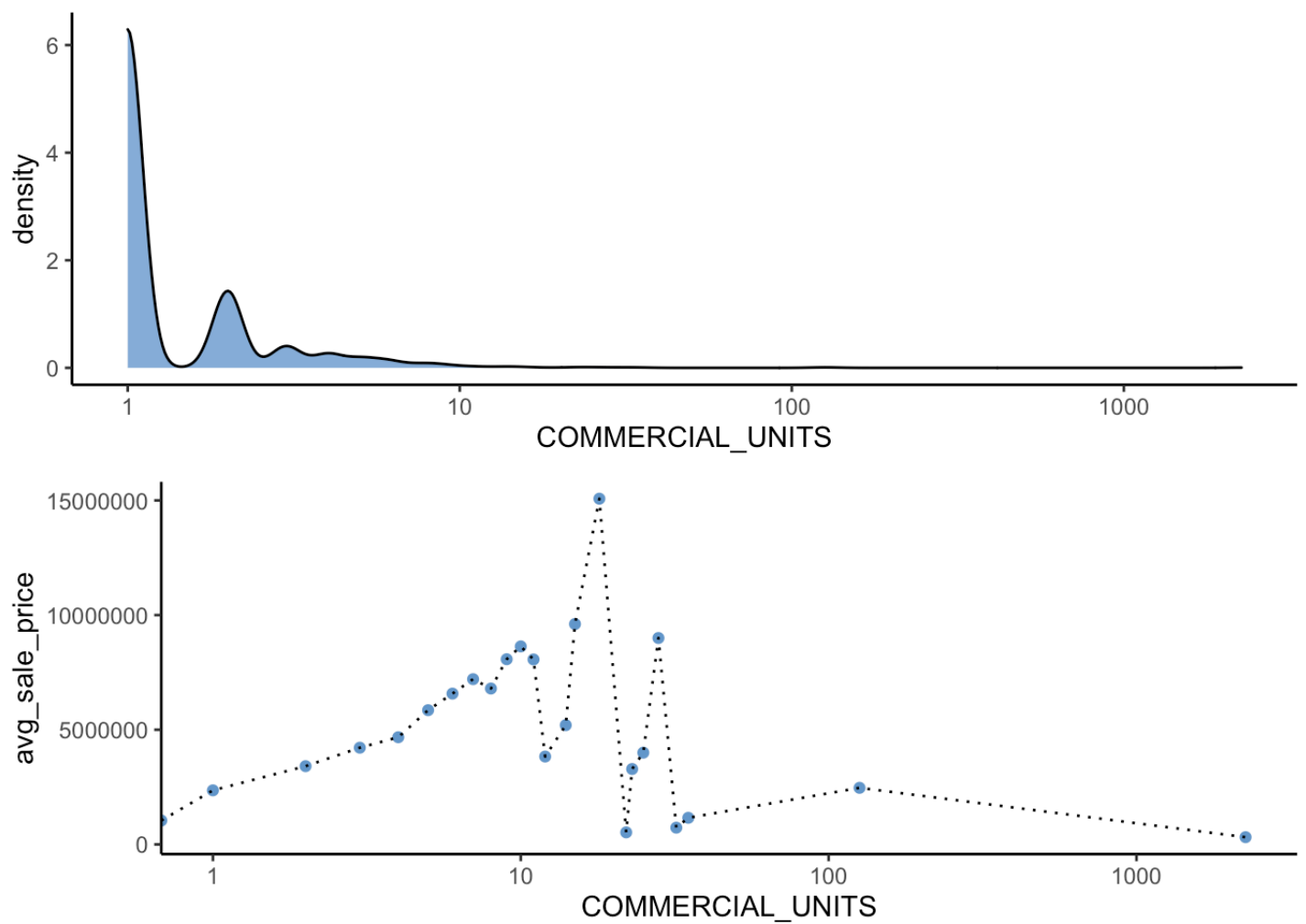
```
summary(Training$COMMERCIAL_UNITS)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.0000	0.0000	0.0000	0.1323	0.0000	2261.0000

```
histPlot <- Training%>%
  filter(!is.na(COMMERCIAL_UNITS))%>%
  ggplot(aes(x=COMMERCIAL_UNITS))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()

corrPlot <- Training%>%
  group_by(COMMERCIAL_UNITS)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x= COMMERCIAL_UNITS, y = avg_sale_price))+
  geom_point(color = "#6699cc")+
  geom_line(aes(group = 1),linetype = 'dotted')+
  theme_classic()+
  scale_x_log10()

plot_grid(histPlot,corrPlot,nrow = 2)
```



The plot above shows the majority of the commercial units is below 10, and it has a positive relationship when the number of units is lower than 10.

## TOTAL UNITS

```
summary(Training$TOTAL_UNITS)
```

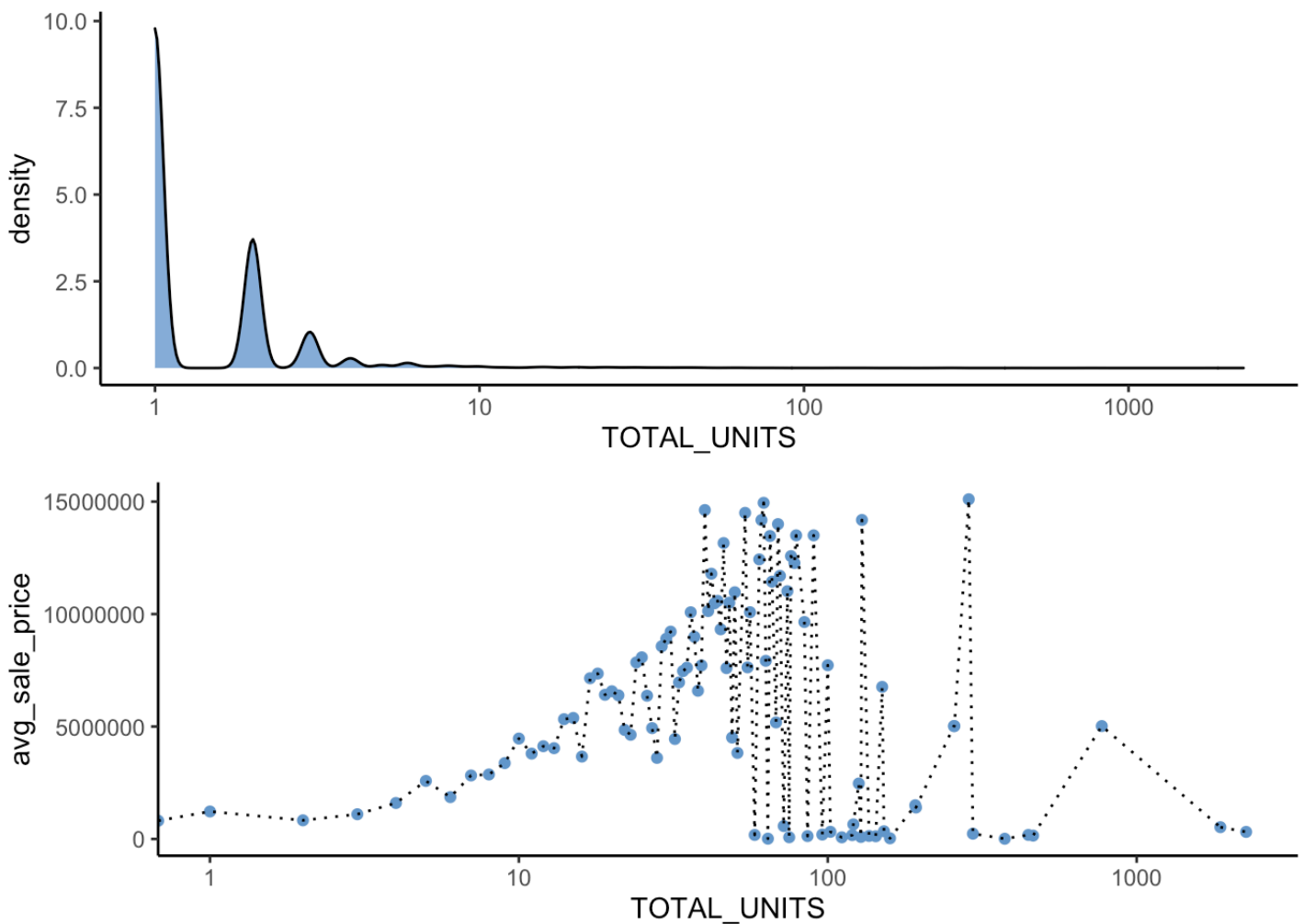
##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0.00	0.00	1.00	1.67	2.00	2261.00

```

histPlot <- Training%>%
  filter(!is.na(TOTAL_UNITS))%>%
  ggplot(aes(x=TOTAL_UNITS))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()

corrPlot <- Training%>%
  group_by(TOTAL_UNITS)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x= TOTAL_UNITS, y = avg_sale_price ))+
  geom_point(color = "#6699cc")+
  geom_line(aes(group = 1),linetype = 'dotted')+
  theme_classic()+
  scale_x_log10()
plot_grid(histPlot,corrPlot,nrow = 2)

```



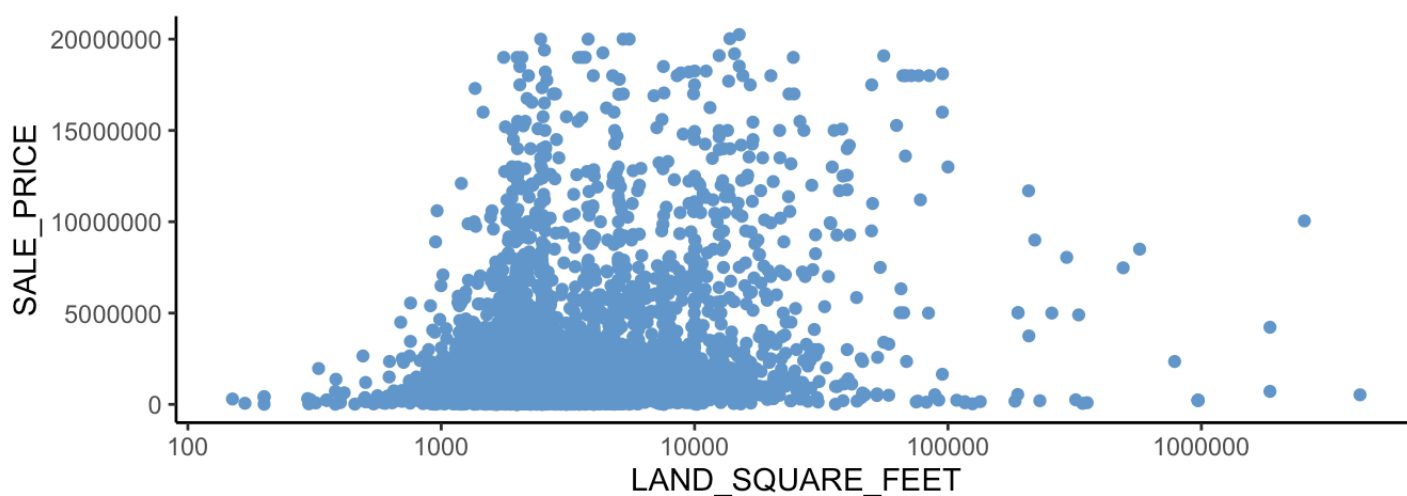
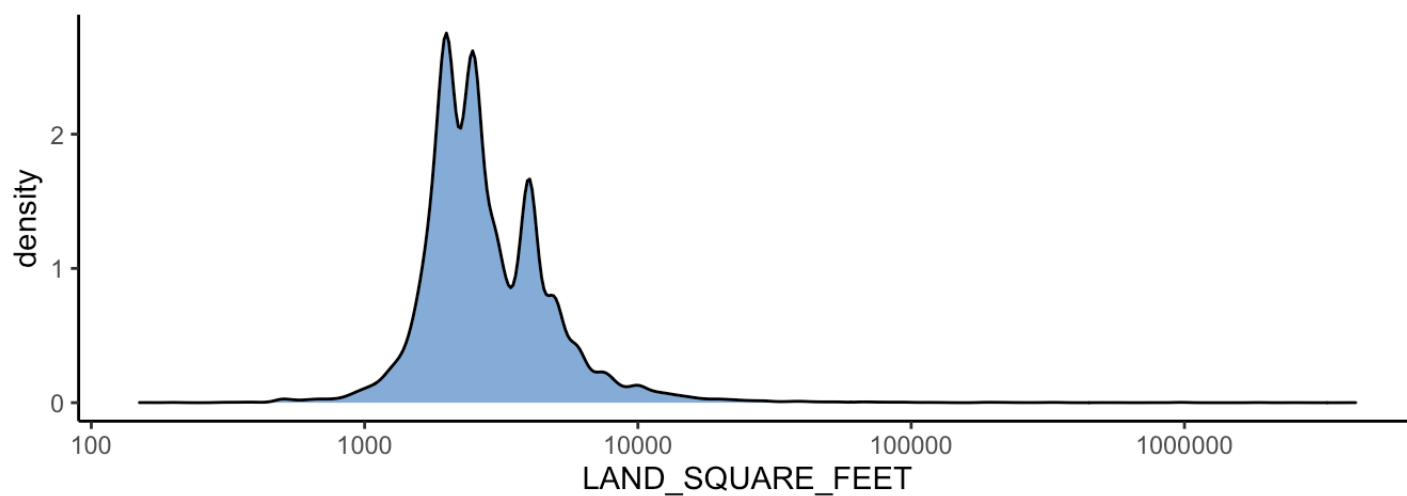
The plot above shows the total units has a positive relationship with sale price when the number of units is lower than 100. is lower than 100.

## LAND SQUARE FEET

```
summary(Training$LAND_SQUARE_FEET)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0	0	500	2157	2500	4228300

```
histPlot <- Training%>%  
  filter(LAND_SQUARE_FEET > 0)%>%  
  ggplot(aes(x=LAND_SQUARE_FEET))+  
  geom_density(fill = "#6699cc",alpha = 0.8)+  
  theme_classic()+  
  scale_x_log10()  
  
corrPlot <- Training%>%  
  filter(LAND_SQUARE_FEET > 0)%>%  
  ggplot(aes(x= LAND_SQUARE_FEET, y = SALE_PRICE))+  
  geom_point(color = "#6699cc")+  
  theme_classic()+  
  scale_x_log10()  
  
plot_grid(histPlot,corrPlot,nrow = 2)
```



Most of houses have a land square feet between 1000 and 10000, and there is no clear correlation between land square feet and sale price.

## GROSS SQUARE FEET

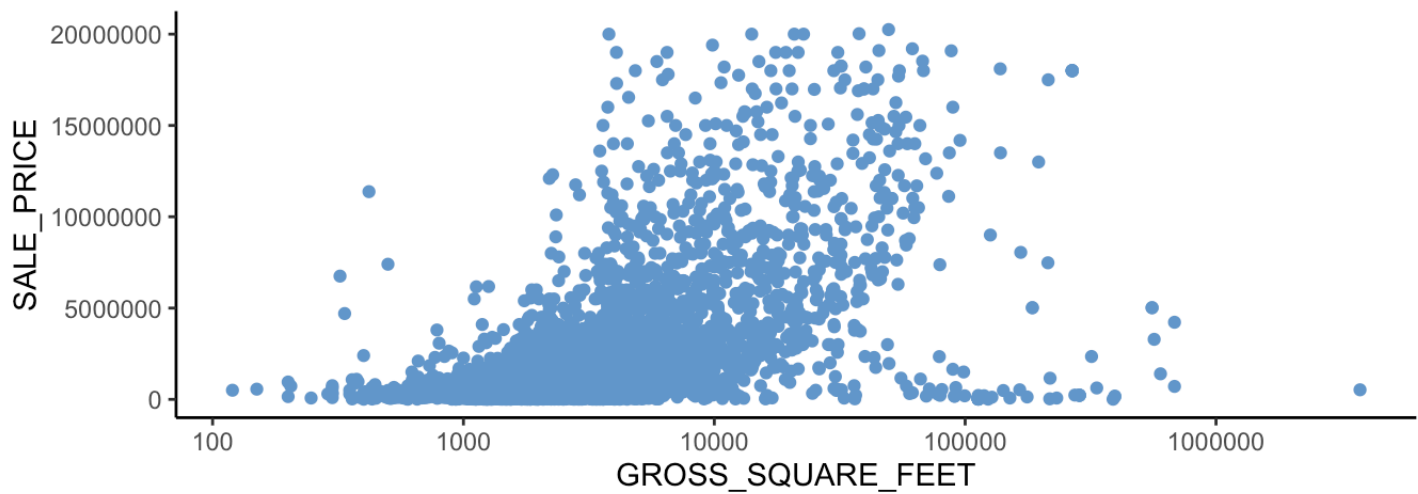
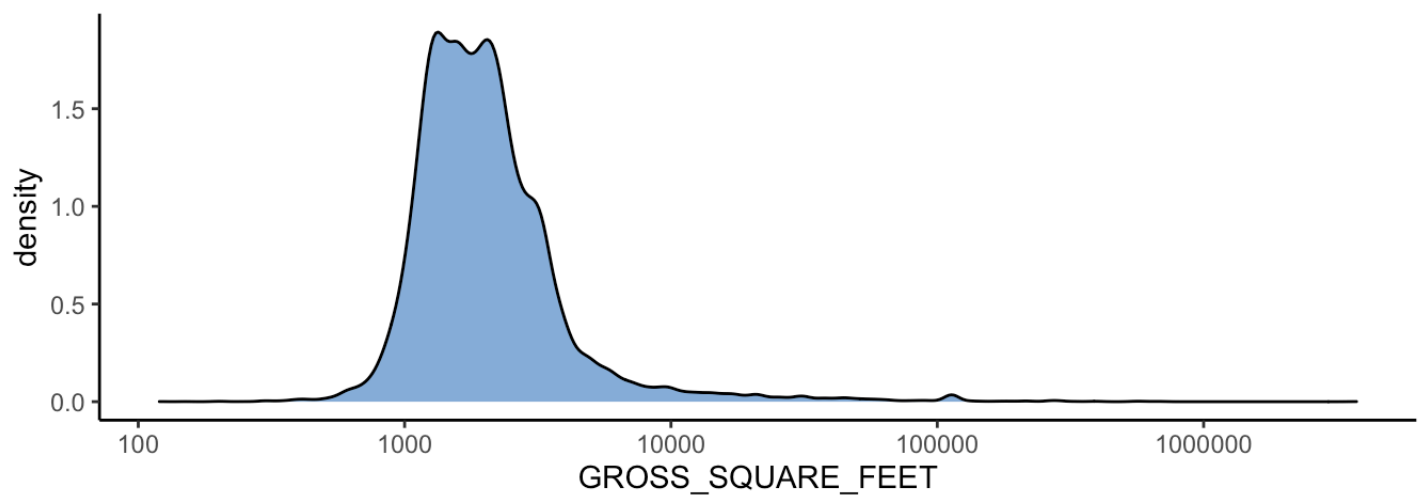
```
summary(Training$GROSS_SQUARE_FEET,na.rm = TRUE)
```

##	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
##	0	0	0	1806	1820	3750565

```
histPlot <- Training%>%
  filter(GROSS_SQUARE_FEET>0)%>%
  ggplot(aes(x=GROSS_SQUARE_FEET))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_log10()

corrPlot <- Training%>%
  filter(GROSS_SQUARE_FEET>0)%>%
  ggplot(aes(x= GROSS_SQUARE_FEET, y = SALE_PRICE))+
  geom_point(color = "#6699cc")+
  theme_classic()+
  scale_x_log10()

plot_grid(histPlot,corrPlot,nrow = 2)
```



Most of houses have a land square feet between 1000 and 100000, and there is a slightly positive correlation between gross square feet and sale price.

## YEAR BUILT

```
summary(Training$YEAR_BUILT)
```

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

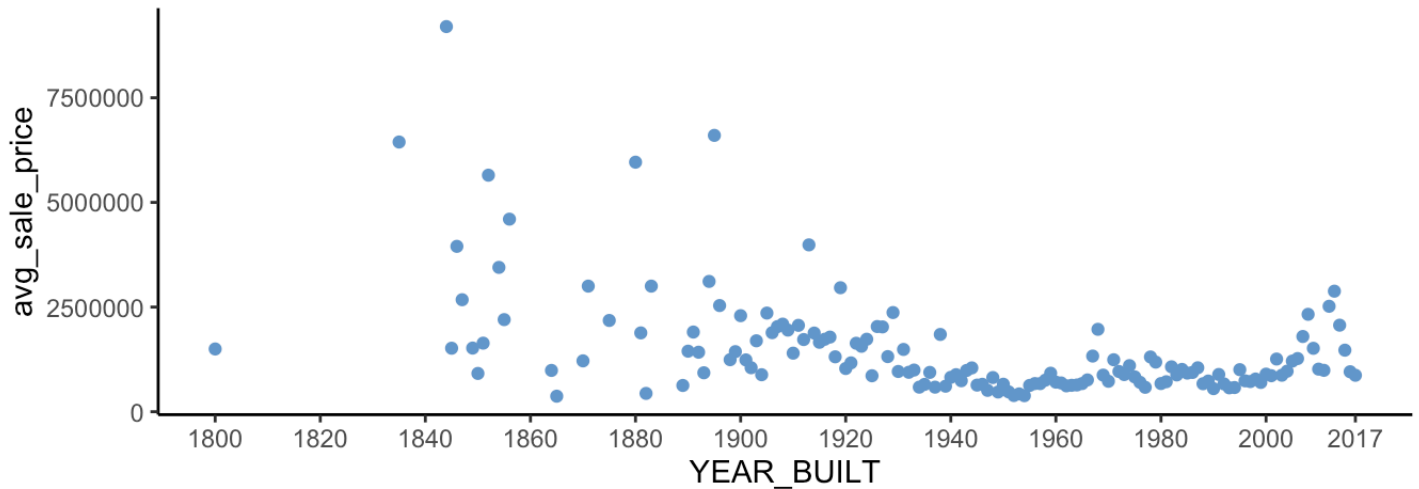
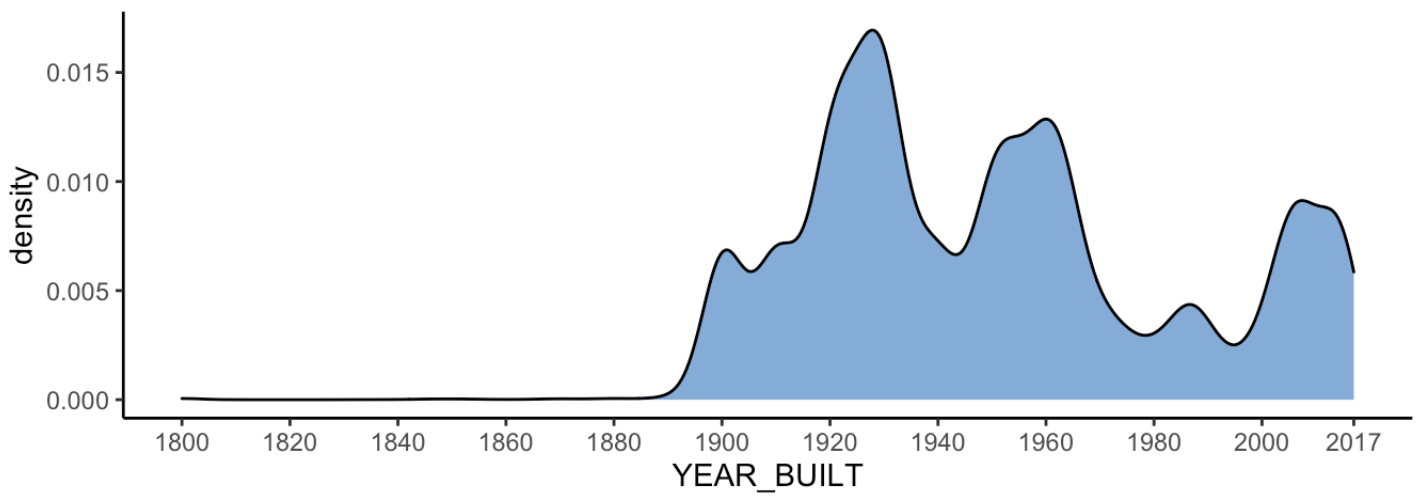
```

histPlot <- Training%>%
  filter(YEAR_BUILT >= 1800)%>%
  ggplot(aes(x=YEAR_BUILT))+
  geom_density(fill = "#6699cc",alpha = 0.8)+
  theme_classic()+
  scale_x_continuous(breaks=c(1800,1820,1840,1860,1880,1900,1920,1940,1960,
1980,2000,2017),
  labels=c("1800","1820","1840","1860","1880", "1900", "1920","1940","1960"
,"1980","2000","2017"))

corrPlot <- Training%>%
  filter(YEAR_BUILT >= 1800)%>%
  group_by(YEAR_BUILT)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x= YEAR_BUILT, y = avg_sale_price ))+
  geom_point(color = "#6699cc")+
  theme_classic()+
  scale_x_continuous(breaks=c(1800,1820,1840,1860,1880,1900,1920,1940,1960,
1980,2000,2017),
  labels=c("1800","1820","1840","1860","1880", "1900", "1920","1940","1960"
,"1980","2000","2017"))

plot_grid(histPlot,corrPlot,nrow = 2)

```

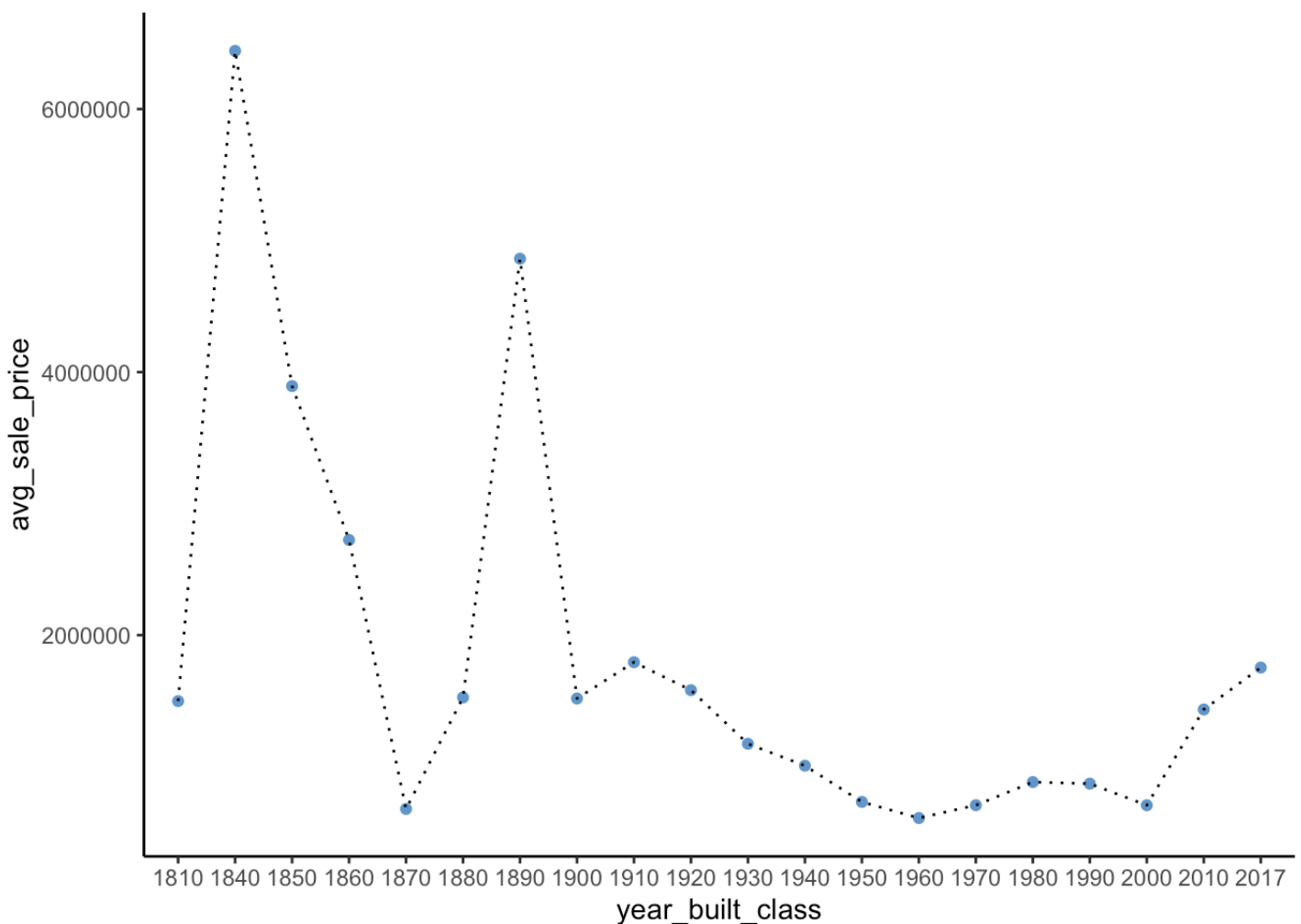


From the plot above, we can see that most of sold properties are built after 1900. It is hard to see the trend clearly from the scatter plot above, therefore, let's create a new group for year built called year built class, each class include 10 years.

```
Training%>%
  mutate(year_built_class = cut(YEAR_BUILT,breaks = seq(1799,2020,10),
                                labels=c("1810","1820","1830","1840","1850",
", "1860","1870","1880","1890",
                                "1900","1910","1920","1930","1940",
", "1950","1960","1970","1980","1990",
                                "2000","2010","2017")))%>%

  filter(!is.na(year_built_class))%>%
  group_by(year_built_class)%>%
  summarise(avg_sale_price = mean(SALE_PRICE))%>%
  ggplot(aes(x= year_built_class, y = avg_sale_price ))+
  geom_point(color = "#6699cc")+
  geom_line(aes(group = 1),linetype = 'dotted')+
  theme_classic()
```





From this plot above, we can see that the average house price has a down trend from 1810 to 1960, and start to increase after 1960.

## 2.2.4 Data Exploratory Analysis Summary

From above data exploratory analysis on the training set, we can get some insights as follows:

1. Among five areas of Borough: Manhattan (1), Bronx (2), Brooklyn (3), Queens (4), Staten Island (5), Manhattan, Brooklyn, and Queens are more popular, and sold more properties. The average sales price in Manhattan is much higher than other areas.
2. The top selling neighborhood in New York Borough are: Flushing-North, Upper East Side (59-79,79-96),Upper West Side (59-79), and Midtown East, and the top selling zip codes are 11354,10314,11201,11375 and 10011.
3. The most expensive neighborhoods are East River, Civic Center, Bloomfield, Little Italy and Soho. The most expensive zip codes are 10013,10007,10018,10069,10037. And the highest average sales price is around \$531,4647.
4. The top selling property types/classes in Borough are one family dwellings, Coops- Elevator Apartments, Condo - Elevator Apartment, and two family dwellings.
5. The majority of sold properties are in Tax Class 1 and Tax class 2. And the average sales price in Tax Class 2B is the highest. The average sales price in TaxClass 4 is the second highest.

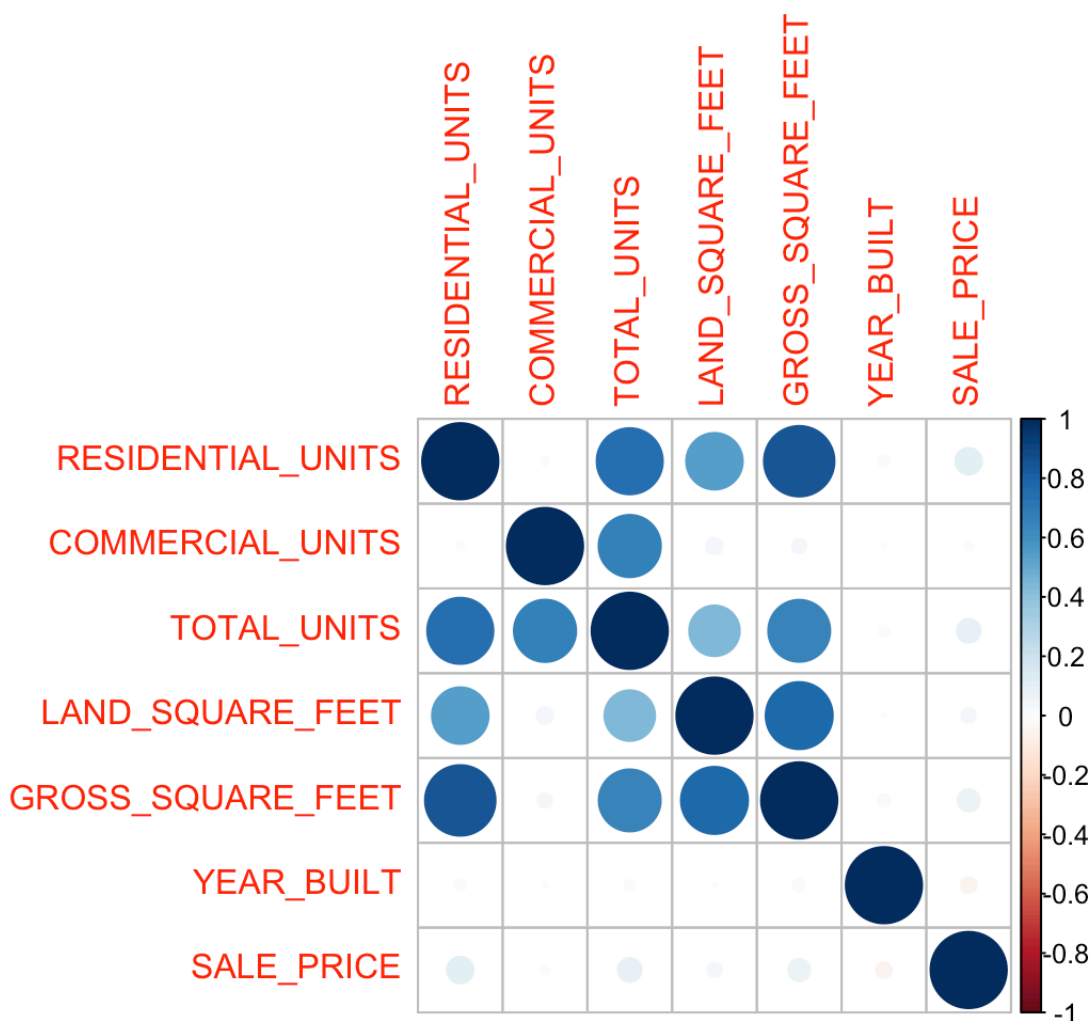
6. In the training dataset, the sales prices has a positive relationship with the following variables: the residential units when the number of units is lower than 100; the commercial units when the number of units is lower than 10; the total units when the number of units is lower than 100; and the sales prices has a slightly positive relationship correlation with gross square feet. There is no clear correlation between the sales price and the sales time and building ages.

## 2.3 Data Modeling

### Correlation Check

Correlation Check on numerical variables

```
Training.cor <- cor(Training%>%select_if(is.numeric))  
corrplot(Training.cor)
```



This plot shows the correlation among the variables. Since TOTAL\_UNITS is the sum of both RESIDENTIAL\_UNITS and COMMERCIAL\_UNITS, I will use TOTAL\_UNITS for later modeling. TOTAL\_UNITS has a strong correlation with GROSS\_SQUARE\_FEET, therefore, it will not be included in the following linear model.

Combined with previous EDA on training dataset and correlation on continuous variables, I will choose the four variables for later modeling: 1) continuous variable: TOTAL\_UNITS, LAND\_SQUARE\_FEET; 2) categorical variable: ZIP\_CODE, BUILDING\_CLASS\_AT\_TIME\_OF\_SALE.

## 2.3.1 Linear Regression Model

### Build Linear Model

```
myControl = trainControl(method = "cv", number = 5, verboseIter = TRUE)
model_lm = train(SALE_PRICE ~ TOTAL_UNITS + LAND_SQUARE_FEET,
                 data = Training,
                 method = "lm",
                 trControl = myControl)
```

```
## + Fold1: intercept=TRUE
## - Fold1: intercept=TRUE
## + Fold2: intercept=TRUE
## - Fold2: intercept=TRUE
## + Fold3: intercept=TRUE
## - Fold3: intercept=TRUE
## + Fold4: intercept=TRUE
## - Fold4: intercept=TRUE
## + Fold5: intercept=TRUE
## - Fold5: intercept=TRUE
## Aggregating results
## Fitting final model on full training set
```

```
model_lm
```

```
## Linear Regression
##
## 43586 samples
##      2 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 34868, 34868, 34869, 34869, 34870
## Resampling results:
##
##      RMSE      Rsquared    MAE
## 1804994  0.01895318  898234.6
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Define RMSE function

```
RMSE <- function(true_price, predicted_price) {  
  sqrt(mean((true_price-predicted_price)^2))  
}
```

## RMSE of Linear Model on Testing Dataset

```
pre_price_lm <- predict(model_lm,Testing)  
RMSE_lm <- rmse(Testing$SALE_PRICE,pre_price_lm)  
RMSE_lm
```

```
## [1] 1767925
```

## 2.3.2 Logical Regression Model

### Build Logistic Regression Model

```
myControl = trainControl(method = "cv", number = 5, verboseIter = TRUE)  
model_glm = train(SALE_PRICE ~ TOTAL_UNITS + LAND_SQUARE_FEET +factor(ZIP_CODE)+factor(BUILDING_CLASS_AT_TIME_OF_SALE),  
  data = Training,  
  method = "glm",  
  trControl = myControl)
```

```
## + Fold1: parameter=none
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :  
prediction from a rank-deficient fit may be misleading
```

```
## - Fold1: parameter=none  
## + Fold2: parameter=none
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :  
prediction from a rank-deficient fit may be misleading
```

```
## - Fold2: parameter=none  
## + Fold3: parameter=none
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :  
prediction from a rank-deficient fit may be misleading
```

```
## - Fold3: parameter=none  
## + Fold4: parameter=none
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :  
prediction from a rank-deficient fit may be misleading
```

```
## - Fold4: parameter=none  
## + Fold5: parameter=none
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :  
prediction from a rank-deficient fit may be misleading
```

```
## - Fold5: parameter=none  
## Aggregating results  
## Fitting final model on full training set
```

```
model_glm
```

```
## Generalized Linear Model  
##  
## 43586 samples  
##      4 predictor  
##  
## No pre-processing  
## Resampling: Cross-Validated (5 fold)  
## Summary of sample sizes: 34869, 34869, 34869, 34869, 34868  
## Resampling results:  
##  
##      RMSE      Rsquared    MAE  
##      1415245   0.3812707   640009.9
```

## RMSE of Logistic Model on Testing Dataset

```
pre_price_glm <- predict(model_glm,Testing)  
RMSE_glm <- rmse(Testing$SALE_PRICE,pre_price_glm)  
RMSE_glm
```

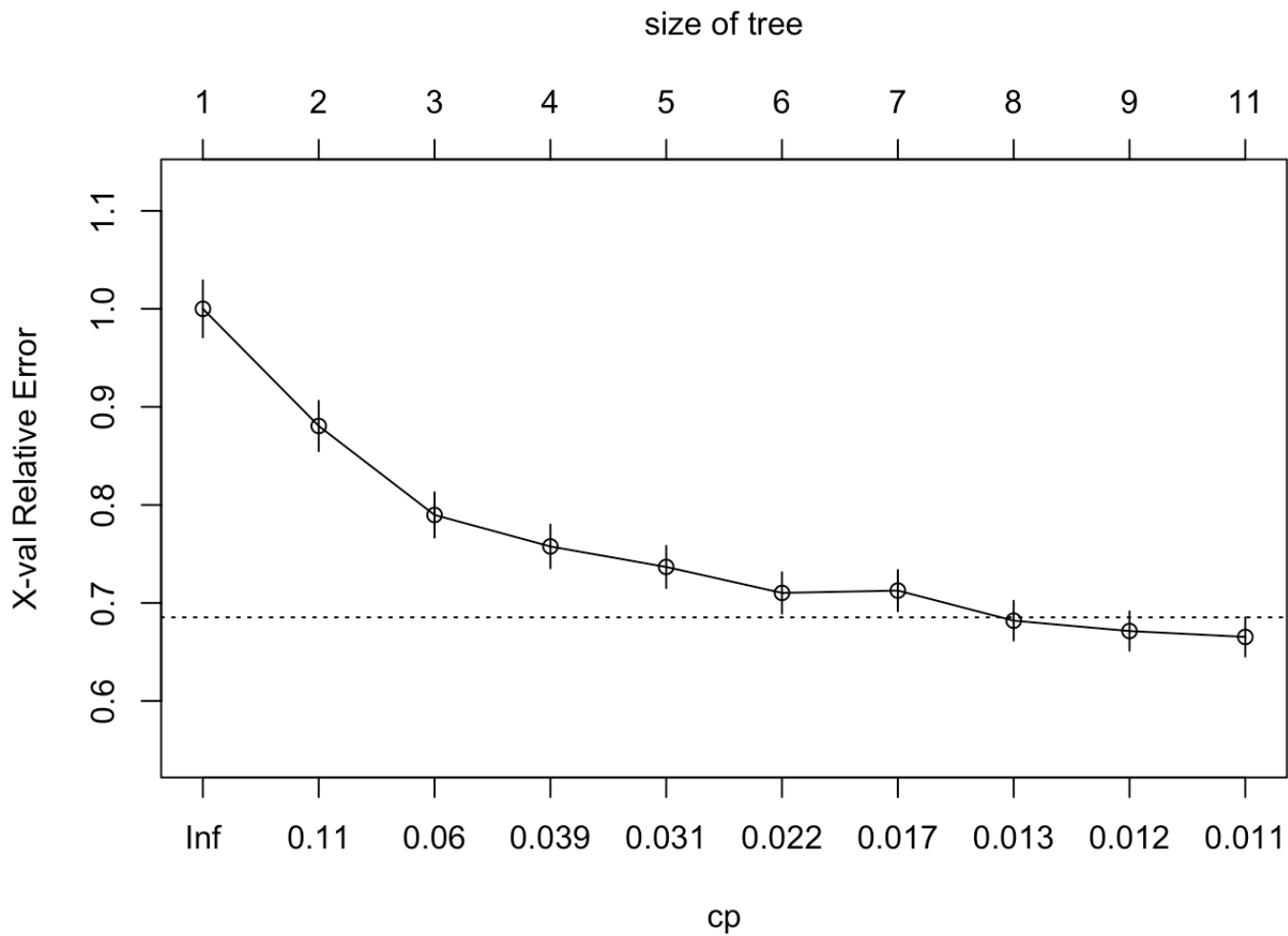
```
## [1] 1379589
```

## 2.3.3 Regression Tree Model

### Build Regression Tree Model

```
library(rpart)
model_retree <- rpart(
  formula = SALE_PRICE ~ TOTAL_UNITS + LAND_SQUARE_FEET +factor(ZIP_CODE)+factor(BUILDING_CLASS_AT_TIME_OF_SALE),
  data     = Training,
  method   = "anova"
)
```

```
plotcp(model_retree)
```



```
model_retree$cpstable
```

##	CP	nsplit	rel error	xerror	xstd
## 1	0.13870850	0	1.0000000	1.0000301	0.02904990
## 2	0.09104143	1	0.8612915	0.8805367	0.02569371
## 3	0.03951908	2	0.7702501	0.7897786	0.02316341
## 4	0.03856828	3	0.7307310	0.7576469	0.02228199
## 5	0.02418470	4	0.6921627	0.7367390	0.02156037
## 6	0.02067328	5	0.6679780	0.7102544	0.02108251
## 7	0.01472913	6	0.6473047	0.7126215	0.02107560
## 8	0.01214174	7	0.6325756	0.6819344	0.02032067
## 9	0.01140170	8	0.6204339	0.6713546	0.02009545
## 10	0.01000000	10	0.5976305	0.6653207	0.02002852

## Tuning

```
hyper_grid <- expand.grid(
  minsplit = seq(5, 20, 1),
  maxdepth = seq(8, 15, 1)
)

models <- list()

for (i in 1:nrow(hyper_grid)) {
  minsplit <- hyper_grid$minsplit[i]
  maxdepth <- hyper_grid$maxdepth[i]

  # train a model and store in the list
  models[[i]] <- rpart(
    formula = SALE_PRICE ~ TOTAL_UNITS + LAND_SQUARE_FEET +factor(ZIP_CODE)+factor(BUILDING_CLASS_AT_TIME_OF_SALE),
    data     = Training,
    method   = "anova",
    control  = list(minsplit = minsplit, maxdepth = maxdepth)
  )
}
```

```

get_cp <- function(x) {
  min <- which.min(x$cpstable[, "xerror"])
  cp <- x$cpstable[min, "CP"]
}

get_min_error <- function(x) {
  min <- which.min(x$cpstable[, "xerror"])
  xerror <- x$cpstable[min, "xerror"]
}

hyper_grid %>%
  mutate(
    cp = purrr::map_dbl(models, get_cp),
    error = purrr::map_dbl(models, get_min_error)
  ) %>%
  arrange(error) %>%
  top_n(-5, wt = error)

```

minsplit <dbl>	maxdepth <dbl>	cp <dbl>	error <dbl>
19	14	0.01	0.6420466
15	13	0.01	0.6446688
16	15	0.01	0.6471475
20	12	0.01	0.6480415
14	12	0.01	0.6493184

5 rows

```

op_model_retree <- rpart(
  formula = SALE_PRICE ~ TOTAL_UNITS + LAND_SQUARE_FEET +factor(ZIP_CODE)+factor(BUILDING_CLASS_AT_TIME_OF_SALE),
  data = Training,
  method = "anova",
  control = list(minsplit = 19, maxdepth = 12, cp = 0.01)
)

```

## RMSE of Regression Tree Model on Testing Dataset

```

pre_price_retree <- predict(op_model_retree,Testing)
RMSE_retree <- rmse(Testing$SALE_PRICE,pre_price_retree)
RMSE_retree

```

```
## [1] 1449782
```



## 2.3.4 Model Ensemble

Ensemble Linear and Regression Tree

```
pre_price_em <- (pre_price_retree + pre_price_glm)/2
RMSE_em <- rmse(Testing$SALE_PRICE,pre_price_em)
RMSE_em
```

```
## [1] 1367118
```

# Part 3 Model Results & Performance on Testing Dataset

```
modelname <-c("Linear","Logistic Regression","Regression Tree","Emsenble" )
rsme_result <- data.frame(model=modelname,rsme= c(RMSE_lm,RMSE_glm,RMSE_retree,RMSE_e
m))
print(rsme_result)
```

```
##           model      rsme
## 1          Linear 1767925
## 2 Logistic Regression 1379589
## 3      Regression Tree 1449782
## 4          Emsenble 1367118
```

For all the models, we can know that the emsenble model of both Logistic Regression and Regression Tree, has the best result. And the RSME is 1367718.

## Part 4 Conclusions

### 4.1 Lesson Learned

From this project, we can learn more about property market in New York City:

1. For real estate, the key factor is always location, location and location! As we can see from previous exploratory data analysis, the majorities of deals and highest average sales prices happened in Mahatton and Brooklyn, the hottest places in New York.
2. Most properties sold are residential properties. Only around 4% of transactions are commercial properties. Therefore, we can learn that, the residential market was very active during that period. There were strong needs for individual residential buyers instead of institutional commercial buyers.
3. The sales prices in this dataset are in a very big range, since the sales price is affected by its area, locations, ages, building type.

4. The RSME of the model is very large: \$136,7118, which indicate to get a better result, more data and variables need be introduced for further analysis.

## 4.2 Limitation & Future Work

As mentioned previously, though all real property transactions have a big number, for this dataset the max sales price is 22 million dollars, the RSME of \$136,7118 is still too large. One of the reasons is, though the dataset originally contains 19 variables, most of them are highly correlated. For example, in fact, zip\_code include all the information in area and neighborhood, building type contains the differences of tax classes. Gross Square feet is highly correlated with land square feet, because the zoning code is regulated by the government which set the rules of floor area ratio.

To build a better model to predict the sales prices, more variables and data for longer periods need to be introduced. For example, in the same zip code, with the same building type and similar scale of building, the prices can vary a lot, sometimes definitely over a million dollars. If the building is designed by a world known designer or represented by a super star, then its valuation is a lot higher than the average.

## Glossary Of Terms

**Borough:** The name of the borough in which the property is located.

**Zip Code:** The property's postal code.

**Residential Units:** The number of residential units at the listed property.

**Commercial Units:** The number of commercial units at the listed property.

**Total Units:** The total number of units at the listed property.

**\*\* Land Square Feet:\*\*** The land area of the property listed in square feet.

**\*\* Gross Square Feet:\*\*** The total area of all the floors of a building as measured from the exterior surfaces of the outside walls of the building, including the land area and space within any building or structure on the property.

### **Tax class:**

Class 1: Includes most residential property of up to three units (such as one-, two-, and three-family homes and small stores or offices with one or two attached apartments), vacant land that is zoned for residential use, and most condominiums that are not more than three stories.

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.

### **Building Class at Time of Sale:**

The Building Classification is used to describe a property's constructive use. The first position of the Building Class is a letter that is used to describe a general class of properties (for example "A" signifies one-family homes, "O" signifies office buildings. "R" signifies condominiums). The second position, a number, adds more specific information about the property's use or construction style (using our previous examples "A0" is a CapeCod style one family home, "O4" is a tower type office building and "R5" is a commercialcondominium unit). The term Building Class as used by the Department of Finance is interchangeable with the term Building Code as used by the Department of Buildings.