Table of Contents

Abstract

This paper aims to establish the significance of the data pre-processing and feature engineering process in the data analysis operation. This is an extension of the work previously done where the dataset used here is the same as that in the previous work. The dataset relates to the sale price of the houses in King County, Washington, together with the attributes of the respective houses. Since initial data exploration was done in the previous paper, this paper continues the work by performing firstly, data pre-processing, followed by an exploratory data analysis, and finally, feature engineering, so as to prepare the data for the modelling stage. Data pre-processing performed in this paper includes the conversion of attributes into their rightful data type, treatment of inconsistencies and outliers previously detected, and the handling of missing values. Exploratory data analysis includes the visualization of the properties of individual attributes, the evaluation of their descriptive statistics, as well the identifying of potential relationships between the variables. The feature engineering performed includes feature creation, transformation of the variables, and finally the scaling of the variables. Five hypothesis statements were then formulated to check for the interrelationships between the attributes of the dataset.

Keywords: Data Pre-Processing, Feature Engineering, Hypothesis Testing, House Price Prediction

1.0 Introduction

This paper aims to demonstrate how data pre-processing and feature engineering techniques are carried out as part of the data analysis workflow. It is an extension of the work previously done and this previous work mentioned will be referred to as "*Assignment Part 1*" throughout this entire report for simplicity's sake. Reference to it will be made whenever necessary to provide for further clarity and coherence. *Assignment Part 1* focused primarily on the process of performing initial data exploration on the chosen dataset, as well as the evaluation of data warehousing concepts through the case study of an actual United Kingdom based company known as Landbay. Recall that the dataset is about the different records of houses sold in King County, Washington, between May 2014 to May 2015. Apart from the house prices, the dataset also contains different attributes such the square footage of the living area, the number of bedrooms and bathrooms, and the view from the houses to name a few. The initial data exploration previously done includes the identification of the type of data of each attribute, their summarizing properties (spread, distribution, median, mean, variance, and percentiles), as well as the detection of outliers, missing values and any inconsistencies present within the dataset.

Proceeding to these discoveries would then be the work documented in this report, where the order of the next few sections follows, firstly, the discussion on some published papers which are related to the same topic of interest as this paper (price of houses sold); secondly, the documentation of the necessary data pre-processing to be carried out, so that the dataset could be cleaned; thirdly, an Exploratory Data Analysis (EDA) on the cleaned data; fourthly, the performing of the respective necessary feature engineering work based on the findings from the EDA carried out; fifthly, the testing of the hypotheses of interest; and lastly, a discussion and conclusion to adequately conclude this paper. Note that the purpose of including related work of the same interest, is not only just for data understanding purposes, but these related work at the same time, would serve as a benchmark for which results from the hypothesis testing will be compared and contrasted with. The expected outcome is for both results (hypothesis testing and published work of others) to be consistent. This is because consistency would then demonstrate once again how having an adequately pre-processed and feature engineered dataset is crucial to ensure that underlying patterns and relationships are correctly identified.

2.0 Related Work

The real estate valuation is inherently multifaceted. That is, the value of a property is composed of a multitude of factors that are working in tandem (Ge & Du, 2007). According to Bello and Bello (2007), the authors proposed that factors influencing the prices of properties could be categorized into two separate groups namely, external and internal factors. These factors could either positively or negatively impact the value of properties. Internal factors are defined as those solely associated with the structural features of the property. External factors, on the other hand, are those externalities, be it be positive or negative ones, which have the ability to influence the value of the property (Babawale & Adewunmi, 2011). External factors could be further distinguished from one another by grouping them into three distinct groups namely, economical attributes, locational attributes, and lastly, social attributes. Economical attributes, according to Khoiry, Tawil, Hamzah, Ani and Sood (2012) include factors such as mortgage loans and interest rates, inflation and unemployment rates, the economic health of the country, as indicated by the Gross Domestic Product (GDP), and also the tax on real property gains, to name a few. Locational attributes on the contrary, refer to the availability and proximity from the property to several amenities such as public transportation, schools, hospitals and retail outlets, just to mention a few (Chiang, Peng & Chang, 2015). Finally, social attributes encompass factors such as population growth as identified by Ong (2013).

While there is no denying that all three of the attribute types of the external factors are equally important among themselves, and when compared with those of the internal factors, note that only the locational attribute ,together with the internal factors of the property will be given focus in this paper. Constraint in terms of the attributes available within the chosen dataset is the reason behind this. In terms of these two factors focused upon, findings from the various literature work done points to a few observations and they are as follows:

1. *Ceteris paribus*, the age of a property is inversely related to its price (Kain & Quigley, 1970; Godman & Thibodeau, 1995; Pashardes & Savva, 2009).
2. Neighborhood qualities precisely in terms of living space and adequacies of living units, are found to significantly affect the satisfaction level experienced by housing residents, subsequently affecting the sale price of the property (Vrbka & Combs, 1993; Owusu-Ansah, 2012). Housing areas that are centralized, densely populated, and legally established with appropriate allocations are found to always provide greater satisfaction for housing residents when compared to those which are unauthorized and without proper allocations (Türkoğlu, 1997).

3. Holding all else equal, as the number of bathrooms and bedrooms increases, so will the price of the property (Rodriguez & Sirmans, 1994; Adair, McGreal, Smyth, Cooper & Ryley, 2000; Kauko, Hooimeijer & Hakfoort., 2002; Wilhelmsson, 2002; Babawale & Adewunmi, 2011; Oloke, Simon & Adesulu, 2013).
4. As built-up area increases, or in other words as the total usable space increases, the value of the property would appreciate too, while keeping all else constant (Chin & Chau, 2003; Reibel, Chernobai, & Carney, 2008; Iman, Hamidi & Liew, 2009; Chiang et al., 2015).
5. Houses with nearby aesthetic views tend to command higher prices than those that have zero or less appealing views (Jayasekare, Herath, Wickramasuriya & Perez, 2019).

Given the above findings from the authors of previous literature work, these findings will serve as a benchmark, for when validating the results of the hypothesis testing of this paper. The said validation and the necessary discussion could be found in *Section 5.0* of this paper.

3.0 Method

This section consists of three sub parts namely *Data Pre-processing*, *Exploratory Data Analysis,* and lastly, *Feature Engineering*. The detailed explanation of each sub parts will be provided in their respective sections. Note that the sequencing of the work and techniques carried out would follow the order in which they are listed.

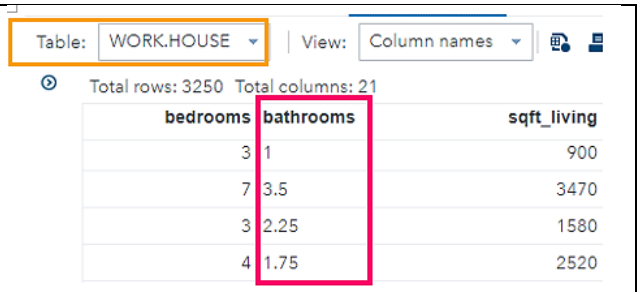
3.1 Data Pre-processing

The data pre-processing work includes the conversion of data type of an attribute which was wrongfully categorized, the dealing of the inconsistencies found within the problematic attributes, the extraction of year information from one of the columns, the handling of missing values through the imputation technique, and lastly, the treatment of outliers which were initially detected in *Assignment Part 1*. Note that for *Section 3.1.1* to *Section 3.1.4*, there are no particular sequencing as to which must be performed first, meaning sequencing of these sections does not matter and their orders are interchangeable. This however is not the case for the two sub sections thereafter. That is, missing values must be first dealt with before any treatments of outliers are to be carried out. This is because if one were to address outliers first (by removing them, for instance), and only then handle the missing values (by either imputing or removing them), the distribution of the data may have already been altered. That is to say, after removing the outliers, the dataset will now be left with a modified distribution. Consequently, statistical measures such as the mean, median, standard deviation, and skewness,

to name a few would now reflect the characteristics of this newly modified dataset. This modified dataset would then have a distribution that is different from the original distribution that contains the outliers. Now, as one proceeds to handle the missing values within the dataset by using the imputation technique for instance, imputation done would thereby be based on the altered distribution. Recall that in the context of continuous variables, the choice between the use of mean or median for imputation purposes is strictly dependent on the skewness of the attribute. That said, if the outliers had a significant influence on the original distribution, the choice of imputation and subsequently the imputed values would then be influenced by this altered distribution, essentially causing data integrity to not be upheld. In other words, to preserve the integrity of data, is it important to ensure that data is complete and accurate by addressing missing data first before carrying out any treatments for outliers.

3.1.1 Converting bathrooms Attribute

No.	Source Code:	Output:
1	14 *Printing the Metadata; 15 proc contents data = house; 16 run; 9 *Make a copy of the dataset; 10 data house; 11 set m_data.house; 12 run;	Alphabetic List of Variables and Attributes \# | Variable | Type | Len | Format | Informat 5 | bathrooms | Char | 4 | $4. | $4. 4 | bedrooms | Num | 8 | BEST12. | BEST32. 11 | condition | Num | 8 | BEST12. | BEST32. 2 | date | Char | 17 | $17. | $17. 8 | floors | Num | 8 | BEST12. | BEST32. 12 | grade | Num | 8 | BEST12. | BEST32. 1 | id | Num | 8 | BEST12. | BEST32. 18 | lat | Num | 8 | BEST12. | BEST32. 19 | long | Num | 8 | BEST12. | BEST32. 3 | price | Num | 8 | BEST12. | BEST32. 13 | sqft_above | Num | 8 | BEST12. | BEST32. 14 | sqft_basement | Num | 8 | BEST12. | BEST32. 6 | sqft_living | Num | 8 | BEST12. | BEST32. 20 | sqft_living15 | Num | 8 | BEST12. | BEST32. 7 | sqft_lot | Num | 8 | BEST12. | BEST32. 21 | sqft_lot15 | Num | 8 | BEST12. | BEST32. 10 | view | Num | 8 | BEST12. | BEST32. 9 | waterfront | Num | 8 | BEST12. | BEST32. 15 | yr_built | Num | 8 | BEST12. | BEST32. 16 | yr_renovated | Num | 8 | BEST12. | BEST32. 17 | zipcode | Num | 8 | BEST12. | BEST32.



As identified in *Assignment Part 1*, notice that the *bathrooms* attribute was categorized as a character data type. Such categorization is incorrect because upon further investigation into the attribute itself, data values were entered in numeric formats ("1" for instance) rather than in strings formats ("One"). That said, the *bathrooms* attribute should then be rightfully categorized as a numeric data type rather than one that is of the character type.

No.	Source Code:	Output:
2	See code block below	See table below

```
/* ------------------------ */
/* bathrooms VARIBALE */
/* ------------------------ */

*Convert the bathrooms attribute from character to numeric;
data house;
    set house;

    /* Convert the 'bathrooms' character variable to a numeric variable
       using the INPUT function. The informat 'BEST32.' specifies that
       the character value should be converted to its numeric representation. */
    bathrooms_num = input(bathrooms, BEST32.);

    /* Assign a format to the newly converted numeric variable.
       The format 'BEST12.' specifies that the numeric value should
       be displayed with up to 12 characters, using the best format available. */
    format bathrooms_num BEST12.;

    /* Drop the original 'bathrooms' character variable
       as we now have its numeric counterpart. */
    drop bathrooms;

    /* Rename the newly created numeric variable to the original
       variable name for consistency and ease of analysis. */
    rename bathrooms_num=bathrooms;
run;

*Printing the Metadata to check if the above changes have been made;
proc contents data = house;
run;
```

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat
21	bathrooms	Num	8	BEST12.	
4	bedrooms	Num	8	BEST12.	BEST32.
10	condition	Num	8	BEST12.	BEST32.
2	date	Char	17	$17.	$17.
7	floors	Num	8	BEST12.	BEST32.
11	grade	Num	8	BEST12.	BEST32.
1	id	Num	8	BEST12.	BEST32.
17	lat	Num	8	BEST12.	BEST32.
18	long	Num	8	BEST12.	BEST32.
3	price	Num	8	BEST12.	BEST32.
12	sqft_above	Num	8	BEST12.	BEST32.
13	sqft_basement	Num	8	BEST12.	BEST32.
5	sqft_living	Num	8	BEST12.	BEST32.
19	sqft_living15	Num	8	BEST12.	BEST32.
6	sqft_lot	Num	8	BEST12.	BEST32.
20	sqft_lot15	Num	8	BEST12.	BEST32.
9	view	Num	8	BEST12.	BEST32.
8	waterfront	Num	8	BEST12.	BEST32.
14	yr_built	Num	8	BEST12.	BEST32.
15	yr_renovated	Num	8	BEST12.	BEST32.
16	zipcode	Num	8	BEST12.	BEST32.

The code above executes the conversion data type from character to numeric for the *bathrooms* attribute. The output column on the right indicates that the conversion has been successful.

3

```
46 /* Compute quartiles for the 'bathrooms' variable */
47 PROC UNIVARIATE DATA=house;
48     VAR bathrooms;
49     OUTPUT OUT=OutliersBathrooms (RENAME=(bedrooms=OriginalBathr
50     Q1=Q1_bathrooms Q3=Q3_bathrooms;
51 RUN;

53 /* Detect and store outliers for 'bathrooms' in a new dataset */
54 DATA OutliersListBathrooms (keep=ObsNum OutlierValue);
55     SET house;
56     IF _N_ = 1 THEN SET OutliersBathrooms;
57     IQR = Q3_bathrooms - Q1_bathrooms;
58     LowerBound = Q1_bathrooms - 1.5 * IQR;
59     UpperBound = Q3_bathrooms + 1.5 * IQR;
60     IF bathrooms < LowerBound OR bathrooms > UpperBound THEN DO;
61         ObsNum = _N_;
62         OutlierValue = bathrooms;
63         OUTPUT;
64     END;
65     DROP IQR LowerBound UpperBound Q1_bathrooms Q3_bathrooms;
66 RUN;

68 /* Print detected outliers */
69 PROC PRINT DATA=OutliersListBathrooms;
70 RUN;
```

The UNIVARIATE Procedure
Variable: bathrooms

Moments			
N	3246	Sum Weights	3246
Mean	2.09881392	Sum Observations	6812.75
Std Deviation	0.7646601	Variance	0.58470506
Skewness	0.48847861	Kurtosis	0.88673875
Uncorrected SS	16196.0625	Corrected SS	1897.36793
Coeff Variation	36.4329628	Std Error Mean	0.01342129

Basic Statistical Measures			
Location		Variability	
Mean	2.098814	Std Deviation	0.76466
Median	2.250000	Variance	0.58471
Mode	2.500000	Range	6.00000
		Interquartile Range	1.00000

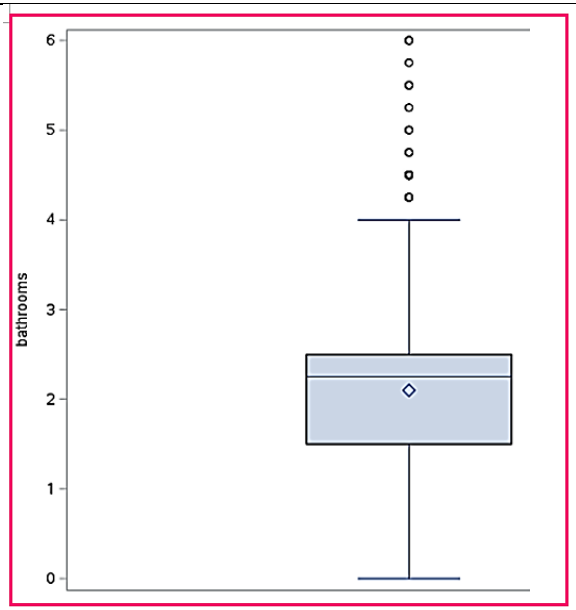
Quantiles (Definition 5)	
Level	Quantile
100% Max	6.00
99%	4.25
95%	3.50
90%	3.00
75% Q3	2.50
50% Median	2.25
25% Q1	1.50
10%	1.00
5%	1.00
1%	1.00
0% Min	0.00

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0.00	2679	5.50	2274
0.75	3200	5.75	1581
0.75	3172	6.00	432
0.75	2804	6.00	2088
0.75	2366	6.00	3226

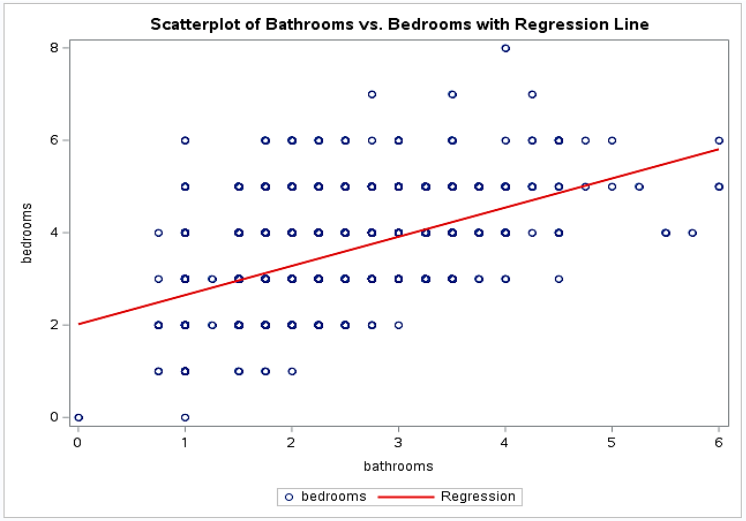
Missing Values			
Missing Value	Count	Percent Of All Obs	Percent Of Missing Obs
.	4	0.12	100.00

4

```
/* Visualize 'bathrooms' distribution with a boxplot */
PROC SGPLOT DATA=house;
    VBOX bathrooms;
RUN;
```




```
/* Scatter plot with regression line for Bathrooms vs. Bedrooms*/
proc sgplot data=house;
    scatter x=bathrooms y=bedrooms;
    reg x=bathrooms y=bedrooms / degree=1 lineattrs=(color=red);
    title "Scatterplot of Bathrooms vs. Bedrooms with Regression Line";
run;
```



Now that the *bathrooms* attribute is in its rightful data type, initial data exploration on the attribute will then be performed as was done with other attributes in *Assignment Part 1*.

Summarizing Properties:

- There are 4 missing values.
- The mean value suggests that most houses within the chosen dataset have 2 bathrooms on average.
- The standard deviation of 0.7647 indicates that there is moderate variability within the *bathrooms* attribute.
- The positive skewness of 0.4885 suggests that the distribution is positively skewed.
- The positive kurtosis value of 0.8867 suggests that while most property houses have numbers of bathrooms that are close to the average, there are nevertheless a few houses with a relatively high number of bathrooms, as indicated by the peakness of the distribution.
- As highlighted in Code Block Number 5 below, there appeared to be inconsistencies within the attribute such that its minimum value is 0. This is clearly an inconsistency because it does not make sense for a house to have zero number of bathrooms. To treat this inconsistency, imputation using the mode (which takes on a value of "2.5") of the *bathrooms* variable was performed, to replace the values of observations where the number of bathrooms is 0. The mode was used because the *bathrooms* attribute is treated as a categorical variable instead of a continuous one. This is so because, while there is a mixture of both integer and decimal values within the attribute, such as 0.75 (meaning the bathroom contains a toilet, a sink and a shower) for decimal values, and 1 (meaning there is a shower, a bathtub, a sink and a toilet) these values were taken as categorical because there is a fixed number of possible sets that the attribute could take. The imputation was successful as there are no longer any observations where the number of bathrooms is "0".

Outliers:

- The boxplot shown in Code Block Number 4 indicates that there are some outliers present within the *bathrooms* attribute. Nevertheless, upon further inspection by plotting a scatterplot between the number of bathrooms and the number of bedrooms, the greater number of

bathrooms do appear logical, unlike what was suggested by the boxplot. That is, the greater number of bathrooms was met with a greater number of bedrooms, essentially suggesting that these outliers within the *bathrooms* attribute are in fact genuine. No further action is thereby necessary.

5

```
/* Notice that the min value is 0 which is illogical for a house to have 0 bathrooms */
/* Calculate the mode (most frequent value) for the 'bathrooms' variable */
proc freq data=house;
    tables bathrooms;
    where bathrooms ne 0; /* Exclude observations where bathrooms is 0 */
run;


/* Impute the observations where the bathroom number is 0 with the mode */
data house;
    set house;
    /* If bathrooms is 0, replace it with the mode */
    if bathrooms = 0 then bathrooms = 2.5; /* Replace X with the identified mode value */
run;

/* Check for observations where 'bathrooms' is 0 */
proc sql;
    select count(*) as Zero_Bathrooms_Count
    from house
    where bathrooms = 0;
quit;
```

The FREQ Procedure

bathrooms	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0.75	9	0.28	9	0.28
1	594	18.31	603	18.58
1.25	2	0.06	605	18.64
1.5	225	6.93	830	25.58
1.75	465	14.33	1295	39.91
2	285	8.78	1580	48.69
2.25	331	10.20	1911	58.89
2.5	792	24.41	2703	83.30
2.75	165	5.08	2868	88.38
3	96	2.96	2964	91.34
3.25	92	2.84	3056	94.18
3.5	106	3.27	3162	97.44
3.75	21	0.65	3183	98.09
4	28	0.86	3211	98.95
4.25	8	0.25	3219	99.20
4.5	15	0.46	3234	99.66
4.75	2	0.06	3236	99.72
5	2	0.06	3238	99.78
5.25	1	0.03	3239	99.82
5.5	2	0.06	3241	99.88
5.75	1	0.03	3242	99.91
6	3	0.09	3245	100.00
Frequency Missing = 4				

Zero_Bathrooms_Count
0

3.1.2 Dealing with Inconsistencies in *bedrooms* Attribute

No. 1

Source Code:

```
/* ------------------------- */
/* bedrooms VARIABLE */
/* ------------------------- */

/* Investigate the 'bedrooms' variable */
PROC UNIVARIATE DATA=house;
    VAR bedrooms;
    OUTPUT OUT=OutliersBedrooms (RENAME=(bedrooms=OriginalBedrooms))
    Q1=Q1_bedrooms Q3=Q3_bedrooms;
RUN;

/* Notice that the min value is 0 which is illogical for a house to have 0 bedrooms */
/* Calculate the mode (most frequent value) for the 'bedrooms' variable */
proc freq data=house noprint;
    tables bedrooms / out=bedroom_freq (drop=percent);
    where bedrooms ne 0; /* Exclude observations where bedrooms is 0 when calculating the mode */
run;

/* Impute the observations where the bedroom number is 0 with the mode */
data house;
    set house;
    /* If bedrooms is 0, replace it with the mode */
    if bedrooms = 0 then bedrooms = 3; /* Replace X with the identified mode value */
run;

/* Check for observations where 'bedrooms' is 0 */
proc sql;
    select count(*) as Zero_Bedrooms_Count
    from house
    where bedrooms = 0;
quit;
```

Output:

Extreme Observations

Lowest		Highest	
Value	Obs	Value	Obs
0	2679	6	3174
0	1239	7	7
1	3172	7	1637
1	3159	7	2178
1	3145	8	268

Total rows: 9 Total columns: 2

	bedrooms	COUNT
1	.	1
2	1	30
3	2	441
4	3	1487
5	4	1008
6	5	238
7	6	39
8	7	3
9	8	1

Zero_Bedrooms_Count
0

As pointed out in *Assignment Part 1*, there were also similar inconsistencies within the *bedrooms* attribute where its minimum value is "0" as well. Treatment of such inconsistency is the same as that for the *bathrooms* attribute. The mode which takes on a value of "3" was used to impute for the observations where the bedroom number is "0". Similar reasonings as to why the mode was used for the above-discussed attribute (*bathrooms*) applies to the *bedrooms* attribute too . Imputation was successful because there are no longer any observations with zero number of bedrooms.

3.1.3 Extract Year from *date* Attribute

No. 1

Source Code:

```
/* ------------------------ */
/* date VARIBALE */
/* ------------------------ */

*Extract the year of the sale from the string and convert it to numeric;
data house;
    set house;

    /* Extract the year from the date string and convert to numeric */
    year_of_sale = input(substr(date, 1, 4), 4.);

    drop date; /* Drop the original date variable */
run;
```

Output:

Total rows: 3250 Total columns: 21 Rows 1-100

long	sqft_living15	sqft_lot15	bathrooms	year_of_sale
-121.958	2530	15389	2.5	2015
-121.968	2000	46173	1.5	2015
-122.161	2550	8800	2.25	2014
-122.102	3360	9755	2.5	2015
-122.307	1720	7503	2.25	2014
-122.272	1460	10643	1	2015
-122.149	3040	13500	3.5	2015
-122.357	1730	8051	2.25	2014
-122.264	1680	10000	1.75	2014
-121.756	1600	16817	1.75	2014
-122.134	3040	5787	1.75	2015
-122.37	2950	29152	2	2015
-122.28	1440	5378	1.75	2014
-122.086	2890	42421	2.5	2014

The source code was executed to extract information about the respective year of the sale of each house within the dataset from the *date* variable. This is done because prior to the extraction and in its original format, records within the *date* variable were confusing. For instance, the date records for one of the houses was recorded as “20150307T000000”. Given that, extraction would then make interpretation and analysis clearer. On a side note, it is important to note that while this process is considered part of feature engineering, it was done at the beginning of the pre-processing step solely for the purpose of data clarity. After extraction, and as indicated in the code shown above, the attribute was then renamed to “year_of_sale” for better clarity, followed by the conversion of the attribute to a numeric data type. The changes performed could be observed in the output column on the right.

3.1.4 Dealing with Inconsistencies in *yr_renovated* Attribute

No. 1

Source Code:

```
/* ------------------------- */
/* yr_renovated  VARIABLE (PRE Name Change) */
/* renovation  VARIABLE (POST Name Change) */
/* ------------------------- */

/*Deal with the inconsistencies within the yr_renovated column, where '1' indicates ... */
/* ... that renovation has been done and '0' indicates otherwise. */
data house;
    set house;

    /* Check if the 'yr_renovated' value is greater than the 'yr_built' value */
    if yr_renovated > yr_built then
        /* If true (i.e., the house was renovated after it was built), set 'yr_renovated' to 1 */
        yr_renovated = 1;
    else
        /* If false (i.e., the house was either not renovated or renovated the ... */
        /* ... same year it was built), set 'yr_renovated' to 0 */
        yr_renovated = 0;

    /* Rename the 'yr_renovated' column to 'renovation' */
    rename yr_renovated=renovation;

/* End of the data step */
run;
```

Output:

Total rows: 3250 Total columns: 21

renovation	zipcode
0	98019
0	98053
0	98006
0	98006
1	98125
0	98001
0	98007
0	98023

No. 2

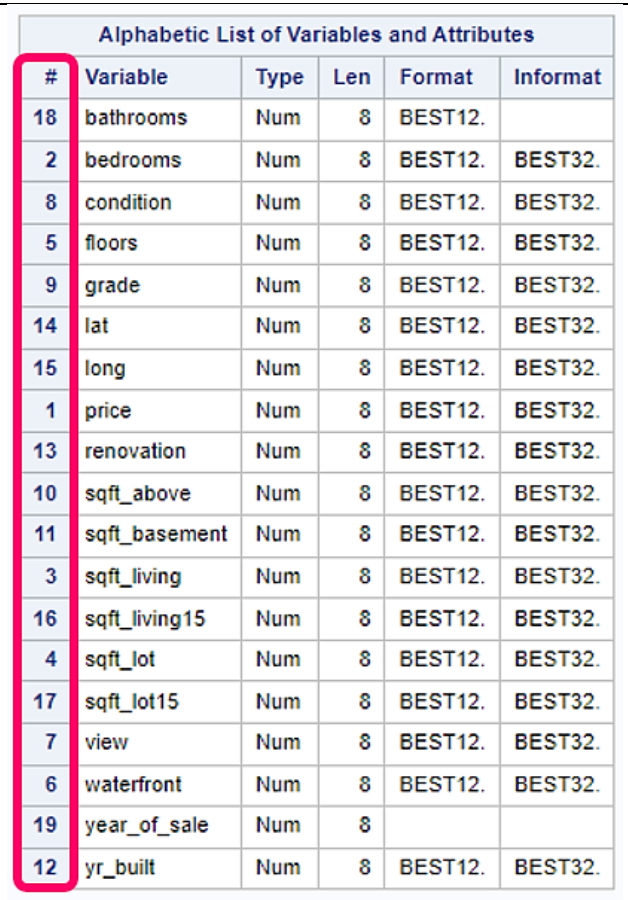
Source Code:

```
/* Create a new dataset 'house_updated' without the specified columns */
data house_updated;
  set house;
  drop id zipcode;
run;

*Display the metadata of the modified dataset to see the changes in variable types and attributes;
proc contents data=house_updated;
run;
```

Output:

The CONTENTS Procedure

Data Set Name	WORK.HOUSE_UPDATED	Observations	3250
Member Type	DATA	Variables	19
Engine	V9	Indexes	0
Created	10/10/2023 12:14:53	Observation Length	152
Last Modified	10/10/2023 12:14:53	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64		
Encoding	utf-8 Unicode (UTF-8)		

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat
18	bathrooms	Num	8	BEST12.	
2	bedrooms	Num	8	BEST12.	BEST32.
8	condition	Num	8	BEST12.	BEST32.
5	floors	Num	8	BEST12.	BEST32.
9	grade	Num	8	BEST12.	BEST32.
14	lat	Num	8	BEST12.	BEST32.
15	long	Num	8	BEST12.	BEST32.
1	price	Num	8	BEST12.	BEST32.
13	renovation	Num	8	BEST12.	BEST32.
10	sqft_above	Num	8	BEST12.	BEST32.
11	sqft_basement	Num	8	BEST12.	BEST32.
3	sqft_living	Num	8	BEST12.	BEST32.
16	sqft_living15	Num	8	BEST12.	BEST32.
4	sqft_lot	Num	8	BEST12.	BEST32.
17	sqft_lot15	Num	8	BEST12.	BEST32.
7	view	Num	8	BEST12.	BEST32.
6	waterfront	Num	8	BEST12.	BEST32.
19	year_of_sale	Num	8		
12	yr_built	Num	8	BEST12.	BEST32.

As indicated in *Assignment Part 1*, there exists inconsistencies within the *yr_renovated* attribute such that while the attribute should contain values that represent the year in which renovation has been done, there were however observations where their values were 0. While these 0 values serve to indicate that no renovations have been done, they are however, conflicting with those observations where their values were indicative of the years when renovation has been done instead (1980, for instance). Hence, to deal with this inconsistency, the *yr_renovated* attribute was converted

into a dummy variable, by flagging houses that has gone through renovations as “1” and “0” for houses that were otherwise. The attribute is then renamed from “yr_renovated” to “renovation” for better clarity. Changes made are evident in the metadata shown in the output column on the right. A new dataset called “house_updated” where the *id* and *zipcode* attributes were dropped is then created. The *id* attribute was dropped because it does not contain any meaningful information. The *zipcode* attribute was dropped because it is lacking of explanatory and predictive power when comparing to its other geographical counterparts namely, *lat* (expressing latitude) and *long* (expressing longitude).

3.1.5 Handling Missing Values

In this subsection, the imputation process will be separated into two different parts – one part being the imputation for categorical variables using their respective mode values, and the second part being the imputation for continuous variables using either their median or mean value, depending on the respective skewness of their distributions. That is, median values will be used when the attribute’s distribution is skewed, while mean values would be used instead when the distribution of the attribute follows a normal distribution. Median is said to be a more robust and reliable choice for when distributions are skewed because, it is resistant to the undue influence of outliers as compared to the mean values. In essence, the median value enables for the preservation of the essential characteristics of the original distribution. There are a total 10 categorical attributes namely *waterfront, bedrooms, bathrooms, floors, view, grade, condition, yr_built, renovation,* and *year_of_sale*. The continuous attributes on the other hand consisted of *price, sqft_living, sqft_lot, sqft_above, sqft_basement, lat, long, sqft_living15,* and *sqft_lot15*, bringing it to a total of 9 attributes that are continuous.

Before that however, a summary of all of the missing values present within the dataset is shown below through the execution of Code Block Number 1. The dataset called “missing_data” is used to store all of the observations with missing values, and from the output shown below, there are a total of 52 observations within the dataset which contained missing values.

Source Code:

```
/* Create a new dataset 'missing_data' from the 'house' dataset */
data missing_data;
    set house_updated;

    /* Create arrays to hold numeric and character variables */
    array num_vars[*] _NUMERIC_;      /* Array for numeric variables */
    array char_vars[*] _CHARACTER_;   /* Array for character variables */

    /* Initialize a flag to indicate if an observation has missing values */
    missing_flag = 0;

    /* Iterate over numeric variables */
    do i = 1 to dim(num_vars);
        /* Check if the current numeric variable has a missing value */
        if num_vars{i} = . then missing_flag = 1;
    end;

    /* Iterate over character variables */
    do i = 1 to dim(char_vars);
        /* Check if the current character variable is an empty string (indicating missing) */
        if char_vars{i} = ' ' then missing_flag = 1;
    end;

    /* Output rows with missing values to the 'missing_data' dataset */
    if missing_flag then output;

    /* Drop unnecessary variables from the output dataset */
    drop i missing_flag;
run;
```

```
/* Print the observations with missing values to the SAS output */
proc print data=missing_data;
run;
```

Output:

Table: WORK.MISSING_DATA | View: Column names

Total rows: 52 Total columns: 19

	price	bedrooms
1	520000	3
2	245000	2
3	82000	3
4	491000	5

3.1.5.1 Imputation for Categorical Attribute

This section will address the imputation done for all categorical attributes previously identified. The methods are repetitive, hence only the first instance for the *waterfront* attribute will be explained in depth. For the rest of the attributes to come, the only difference would be the individual mode values to be imputed with for each attribute. Given that, only the individual mode value (without any further explanation) will be explicitly mentioned for the rest of the variables coming after the first instance, which is the *waterfront* attribute.

No.	Source Code:	Output:
1	See code block below	See output tables below

```
/* Create an initial copy of the dataset for imputation */
data house_imputed;
    set house_updated;
run;

/* ------------------------ */
/* waterfront VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the waterfront variable */
proc freq data=house_imputed;
    tables waterfront / out=stats_waterfront;
    where waterfront is not missing;
run;

/* Impute missing values for the 'waterfront' variable with "0" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'waterfront' is missing, impute with the mode 0 */
    if waterfront = . then waterfront = 0;

run;

/* Check the number of missing values for 'waterfront' after imputation */
proc means data=house_imputed n nmiss;
    var waterfront;
run;
```

The FREQ Procedure

waterfront	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	3216	99.05	3216	99.05
1	31	0.95	3247	100.00

The MEANS Procedure

Analysis Variable : waterfront	
N	N Miss
3250	0

Making reference to Code Block Number 1 shown above, the imputation process was initiated by firstly creating a copy of the dataset for imputation purposes. The new dataset is called “house_imputed”. Following that, since the waterfront attribute is categorical, the mode value is then identified. Once identified, the missing values will then be imputed with the mode value of “0” in this case. Since there are now no longer any missing values within the *waterfront* attribute, as indicated by the “N Miss” column in the MEANS Procedure output, imputation is thereby said to be a success.

2

```
/* ------------------------- */
/* bedrooms VARIABLE */
/* ------------------------- */

/* Calculate and display mode for the bedrooms variable */
proc freq data=house_imputed;
    tables bedrooms / out=stats_bedrooms;
    where bedrooms is not missing;
run;

/* Impute missing values for the 'bedrooms' variable with "3" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'bedrooms' is missing, impute with the mode 3 */
    if bedrooms = . then bedrooms = 3;

run;

/* Check the number of missing values for 'bedrooms' after imputation */
proc means data=house_imputed n nmiss;
    var bedrooms;
run;
```

The FREQ Procedure

bedrooms	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	30	0.92	30	0.92
2	441	13.57	471	14.50
3	1489	45.83	1960	60.33
4	1008	31.02	2968	91.35
5	238	7.33	3206	98.68
6	39	1.20	3245	99.88
7	3	0.09	3248	99.97
8	1	0.03	3249	100.00

The MEANS Procedure

Analysis Variable : bedrooms	
N	N Miss
3250	0

The mode value is 3.

3

```
/* ------------------------ */
/* bathrooms VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the bathrooms variable */
proc freq data=house_imputed;
    tables bathrooms / out=stats_bathrooms;
    where bathrooms is not missing;
run;

/* Impute missing values for the 'bathrooms' variable with "2.5" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'bathrooms' is missing, impute with the mode 2.5 */
    if bathrooms = . then bathrooms = 2.5;

run;

/* Check the number of missing values for 'bathrooms' after imputation */
proc means data=house_imputed n nmiss;
    var bathrooms;
run;
```

The FREQ Procedure

bathrooms	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0.75	9	0.28	9	0.28
1	594	18.30	603	18.58
1.25	2	0.06	605	18.64
1.5	225	6.93	830	25.57
1.75	465	14.33	1295	39.90
2	285	8.78	1580	48.68
2.25	331	10.20	1911	58.87
2.5	793	24.43	2704	83.30
2.75	165	5.08	2869	88.39
3	96	2.96	2965	91.34
3.25	92	2.83	3057	94.18
3.5	106	3.27	3163	97.44
3.75	21	0.65	3184	98.09
4	28	0.86	3212	98.95
4.25	8	0.25	3220	99.20
4.5	15	0.46	3235	99.66
4.75	2	0.06	3237	99.72
5	2	0.06	3239	99.78
5.25	1	0.03	3240	99.82
5.5	2	0.06	3242	99.88
5.75	1	0.03	3243	99.91
6	3	0.09	3246	100.00

The MEANS Procedure

Analysis Variable : bathrooms	
N	N Miss
3250	0

The mode value is 2.5.

4

```
309 /* ------------------------ */
310 /* floors VARIABLE */
311 /* ------------------------ */
312
313 /* Calculate and display mode for the floors variable */
314 proc freq data=house_imputed;
315     tables floors / out=stats_floors;
316     where floors is not missing;
317 run;
318
319 /* Impute missing values for the 'floors' variable with "1" since it is the mode */
320 data house_imputed;
321     set house_imputed;
322
323     /* If 'floors' is missing, impute with the mode 1 */
324     if floors = . then floors = 1;
325
326 run;
327
328 /* Check the number of missing values for 'floors' after imputation */
329 proc means data=house_imputed n nmiss;
330     var floors;
331 run;
```

The FREQ Procedure

floors	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	1610	49.60	1610	49.60
1.5	306	9.43	1916	59.03
2	1205	37.12	3121	96.15
2.5	23	0.71	3144	96.86
3	101	3.11	3245	99.97
3.5	1	0.03	3246	100.00

The MEANS Procedure

Analysis Variable : floors	
N	N Miss
3250	0

The mode value is 1.

5

```
/* ------------------------ */
/* view VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the view variable */
proc freq data=house_imputed;
    tables view / out=stats_view;
    where view is not missing;
run;

/* Impute missing values for the 'view' variable with "0" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'view' is missing, impute with the mode 0 */
    if view = . then view = 0;

run;

/* Check the number of missing values for 'view' after imputation */
proc means data=house_imputed n nmiss;
    var view;
run;
```

The FREQ Procedure

view	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	2930	90.21	2930	90.21
1	53	1.63	2983	91.84
2	144	4.43	3127	96.27
3	67	2.06	3194	98.34
4	54	1.66	3248	100.00

The MEANS Procedure

Analysis Variable : view	
N	N Miss
3250	0

The mode value is 0.

6

```
/* ------------------------ */
/* condition VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the condition variable */
proc freq data=house_imputed;
    tables condition / out=stats_condition;
    where condition is not missing;
run;

/* Impute missing values for the 'condition' variable with "3" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'condition' is missing, impute with the mode 3 */
    if condition = . then condition = 3;

/* Check the number of missing values for 'condition' after imputation */
proc means data=house_imputed n nmiss;
     var condition;
run;
```

The FREQ Procedure

condition	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	5	0.15	5	0.15
2	22	0.68	27	0.83
3	2093	64.42	2120	65.25
4	877	26.99	2997	92.24
5	252	7.76	3249	100.00

The MEANS Procedure

Analysis Variable : condition	
N	N Miss
3250	0

The mode value is 3.

7

```
/* ------------------------ */
/* grade VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the grade variable */
proc freq data=house_imputed;
    tables grade / out=stats_grade;
    where grade is not missing;
run;

/* Impute missing values for the 'grade' variable with "7" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'grade' is missing, impute with the mode 7 */
    if grade = . then grade = 7;

run;

/* Check the number of missing values for 'grade' after imputation */
proc means data=house_imputed n nmiss;
    var grade;
run;
```

The FREQ Procedure

grade	Frequency	Percent	Cumulative Frequency	Cumulative Percent
4	4	0.12	4	0.12
5	33	1.02	37	1.14
6	336	10.34	373	11.48
7	1302	40.07	1675	51.55
8	938	28.87	2613	80.42
9	389	11.97	3002	92.40
10	169	5.20	3171	97.60
11	61	1.88	3232	99.48
12	16	0.49	3248	99.97
13	1	0.03	3249	100.00

The MEANS Procedure

Analysis Variable : grade	
N	N Miss
3250	0

The mode value is 7.

8

```
406 /* ------------------------ */
407 /* yr_built VARIABLE */
408 /* ------------------------ */
409
410 /* Calculate and display mode for the yr_built variable */
411 /* Order the output by frequency in descending order */
412 proc freq data=house_imputed order=freq;
413     tables yr_built / out=stats_yr_built;
414     where yr_built is not missing;
415 run;

417 /* Impute missing values for the 'yr_built' variable with "2014" since it is the mode */
418 data house_imputed;
419     set house_imputed;
420
421     /* If 'yr_built' is missing, impute with the mode 2014 */
422     if yr_built = . then yr_built = 2014;
423
424 run;
425
426 /* Check the number of missing values for 'yr_built' after imputation */
427 proc means data=house_imputed n nmiss;
428     var yr_built;
429 run;
```

The FREQ Procedure

yr_built	Frequency	Percent	Cumulative Frequency	Cumulative Percent
2014	80	2.46	80	2.46
2005	73	2.25	153	4.71
1978	67	2.06	220	6.78
2006	66	2.03	286	8.81
1968	65	2.00	351	10.81
2004	65	2.00	416	12.82
1979	62	1.91	478	14.73
1977	60	1.85	538	16.57
2007	58	1.79	596	18.36
1967	56	1.73	652	20.09
2003	56	1.73	708	21.81
1959	55	1.69	763	23.51
2008	55	1.69	818	25.20
1990	52	1.60	870	26.80
2001	51	1.57	921	28.37
1947	48	1.48	969	29.85
1962	46	1.42	1015	31.27
1980	46	1.42	1061	32.69
1969	45	1.39	1106	34.07

The MEANS Procedure

Analysis Variable : yr_built	
N	N Miss
3250	0

The mode value is 2014.

9

```
/* ------------------------ */
/* renovation VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the renovation variable */
proc freq data=house_imputed;
    tables renovation / out=stats_renovation;
    where renovation is not missing;
run;
```

```
/* Impute missing values for the 'renovation' variable with "0" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'renovation' is missing, impute with the mode 0 */
    if renovation = . then renovation = 0;

run;

/* Check the number of missing values for 'renovation' after imputation */
proc means data=house_imputed n nmiss;
    var renovation;
run;
```

The FREQ Procedure

renovation	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	3098	95.32	3098	95.32
1	152	4.68	3250	100.00

The MEANS Procedure

Analysis Variable : renovation	
N	N Miss
3250	0

The mode value is 0.

10

```
/* ------------------------ */
/* year_of_sale VARIABLE */
/* ------------------------ */

/* Calculate and display mode for the year_of_sale variable */
proc freq data=house_imputed;
    tables year_of_sale / out=stats_year_of_sale;
    where year_of_sale is not missing;
run;

/* Impute missing values for the 'year_of_sale' variable with "2014" since it is the mode */
data house_imputed;
    set house_imputed;

    /* If 'year_of_sale' is missing, impute with the mode 2014 */
    if year_of_sale = . then year_of_sale = 2014;

run;

/* Check the number of missing values for 'year_of_sale' after imputation */
proc means data=house_imputed n nmiss;
    var year_of_sale;
run;
```

The FREQ Procedure

year_of_sale	Frequency	Percent	Cumulative Frequency	Cumulative Percent
2014	2202	67.82	2202	67.82
2015	1045	32.18	3247	100.00

The MEANS Procedure

Analysis Variable : year_of_sale	
N	N Miss
3250	0

The mode value is 2014.

3.1.5.2 Imputation for Continuous Attributes

This section will address the imputation done for all continuous attributes which were previously identified.

No.	Imputation for Continuous Attributes	
1	486 /* Check skewness for each numerical variable */ 487 proc means data=house_imputed skewness; 488 var price sqft_living sqft_lot sqft_above sqft_basement 489 lat long sqft_living15 sqft_lot15; 490 output out=SkewnessOutput skewness=; 491 run; 493 /* Calculate the mean and median for each continuous variable in the house_imputed dataset. */ 494 proc means data=house_imputed mean median; 495 /* List of continuous variables for which we want to calculate mean and median */ 496 var price sqft_living sqft_lot sqft_above sqft_basement lat long sqft_living15 sqft_lot15; 497 498 /* Output the calculated mean and median values to the stats_continuous dataset. */ 499 /* The mean and median values will be named with suffixes _mean and _median, respectively. */ 500 output out=stats_continuous mean= median= / autoname; 501 run;	The MEANS Procedure Variable / Skewness price 3.9249660 sqft_living 1.3345748 sqft_lot 9.8838803 sqft_above 1.3657281 sqft_basement 1.5988860 lat -0.5199184 long 0.9185949 sqft_living15 1.0918865 sqft_lot15 8.0607245 The MEANS Procedure Variable / Mean / Median price 543405.95 459975.00 sqft_living 2061.44 1880.00 sqft_lot 15002.50 7560.00 sqft_above 1771.19 1540.00 sqft_basement 289.1629698 0 lat 47.5636364 47.5776000 long -122.2146305 -122.2290000 sqft_living15 1984.13 1840.00 sqft_lot15 12710.77 7563.50

2

```
509 /* Begin a new DATA step to handle imputation of missing values in the house_imputed dataset. */
510 data house_imputed (keep = price bedrooms sqft_living sqft_lot floors waterfront view condition
511                            grade sqft_above sqft_basement yr_built renovation lat long
512                            sqft_living15 sqft_lot15 bathrooms year_of_sale);
513     /* Read the house_imputed dataset */
514     set house_imputed;
515
516     /* For the first observation, read in the mean and median values from stats_continuous */
517     /* These values will be available for all subsequent observations in the DATA step */
518     if _N_ = 1 then set stats_continuous;
519
520     /* Impute missing values for each continuous variable based on skewness */
521     /* For each variable, if its value is missing (represented by .), */
522     /* replace it with its corresponding median value. */
523     if price = . then price = price_median;
524     if sqft_living = . then sqft_living = sqft_living_median;
525     if sqft_lot = . then sqft_lot = sqft_lot_median;
526     if sqft_above = . then sqft_above = sqft_above_median;
527     if sqft_basement = . then sqft_basement = sqft_basement_median;
528     if lat = . then lat = lat_median;
529     if long = . then long = long_median;
530     if sqft_living15 = . then sqft_living15 = sqft_living15_median;
531     if sqft_lot15 = . then sqft_lot15 = sqft_lot15_median;
532
533     /* End the DATA step. The modified data will be written back to the house_imputed dataset. */
534 run;

536 /* Check for missing values in the continuous variables */
537 proc means data=house_imputed n nmiss;
538     var price sqft_living sqft_lot sqft_above sqft_basement lat long sqft_living15 sqft_lot15;
539 run;
```

The MEANS Procedure

Variable	N	N Miss
price	3250	0
sqft_living	3250	0
sqft_lot	3250	0
sqft_above	3250	0
sqft_basement	3250	0
lat	3250	0
long	3250	0
sqft_living15	3250	0
sqft_lot15	3250	0

Before performing imputation for the missing values within the continuous variables of the dataset, Code Block Number 1 began by checking for the skewness of each of the continuous variable. This is to determine whether the mean value or the median value should be used for when imputing for the missing values. The mean and median for each continuous variables were then computed, as could be seen in the output column on the right. As shown in Code Block Number 2, and given that all of the continuous variables have skewed distributions, the missing values of each variable were then replaced with its corresponding median value. Imputation was successful because there are now no longer any missing values present within the continuous variables, as could be seen in the output column above.

3.1.6 Treatment of outlier initially detected in Assignment Part 1

This section will address the outlier within the *sqft_lot* attribute which was initially identified in *Assignment Part 1*.

No. 1

Source Code:

```
545 /* Identify and display the observation with sqft_lot = 533610 in house_imputed dataset */
546 proc print data=house_imputed;
547     where sqft_lot = 533610;
548 run;
```

Output:

Obs	price	bedrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	yr_built	lat	long	sqft_living15	sqft_lot15	bathrooms	year_of_sale
2627	375000	1	800	533610	1.5	0	0	5	5	800	0	1950	47.4134	-121.986	1790	216057	1	2014

No. 2

Source Code:

```
550 /* Compute the mean (average) values for the variables sqft_lot and sqft_lot15 in the house_imputed dataset */
551 proc means data=house_imputed mean;
552     /* Specify the two variables of interest */
553     var sqft_lot;
554     var sqft_lot15;
555
556     /* Output the computed mean values to a new dataset called average_ratios */
557     /* The mean values for sqft_lot and sqft_lot15 are saved as mean_sqft_lot and mean_sqft_lot15, respectively */
558     output out=average_ratios mean=mean_sqft_lot mean_sqft_lot15;
559 run;
```

Output:

The MEANS Procedure

Variable	Mean
sqft_lot	14997.92
sqft_lot15	12707.60

3

```
561 /* Calculate the average ratio of sqft_lot to sqft_lot15 using the mean values computed in the previous step */
562 data ratios;
563     /* Read the average_ratios dataset containing the mean values */
564     set average_ratios;
565 
566     /* Compute the average ratio by dividing the mean of sqft_lot by the mean of sqft_lot15 */
567     avg_ratio = mean_sqft_lot / mean_sqft_lot15;
568 run;
```

Obs	_TYPE_	_FREQ_	mean_sqft_lot	mean_sqft_lot15	avg_ratio
1	0	3250	14997.923385	12707.601846	1.18023

4

```
574 /* Impute the sqft_lot for the specific observation based on its sqft_lot15 value and the computed avg_ratio */
575 data house_imputed;
576     set house_imputed;
577 
578     /* If the sqft_lot is the identified outlier value (533610), then compute the imputed value */
579     if sqft_lot = 533610 then sqft_lot = sqft_lot15 * 1.18023;
580 run;
```

No.	Source Code:

5

```
582 /* Display the observation where the original sqft_lot was the identified outlier value (533610) */
583 proc print data=house_imputed;
584     where sqft_lot15 = 216057; /* Using the sqft_lot15 value to identify the specific observation */
585 run;
```

Output:

Obs	price	bedrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	grade	sqft_above	sqft_basement	yr_built	lat	long	sqft_living15	sqft_lot15	bathrooms	year_of_sale
2627	375000	1	800	254996.95311	1.5	0	0	5	5	800	0	1950	47.4134	-121.986	1790	216057	1	2014

No.	Source Code:	Output:
6	587 /* Given the revised observation and considering the features of the house, It's still indeed unusual for a ... */ 588 /*... property with such a large lot size to have such a small living space. */ 589 /* Remove the observation with sqft_lot15 = 216057 from the house_imputed dataset */ 590 data house_imputed; 591 set house_imputed; 592 593 /* Exclude the observation with sqft_lot15 = 216057 */ 594 if sqft_lot15 ne 216057; 595 run; 596 597 /* Display the observation with sqft_lot15 = 216057 to verify its removal */ 598 proc print data=house_imputed; 599 where sqft_lot15 = 216057; 600 run;	NOTE: No observations were selected from data set WORK.HOUSE_IMPUTED. NOTE: There were 0 observations read from the data set WORK.HOUSE_IMPUTED. WHERE sqft_lot15=216057;

Code Block Number 1 was first executed to display the entire observation for which the *sqft_lot* outlier with a value of 533610 was present. Following that, Code Block Number 2 then proceeded to compute the mean values for both *sqft_lot* and *sqft_lot15* variables. The average ratio of both variables were then subsequently computed in Code Block Number 3. The average ratio between both variables was computed in such a way because considering that the *sqft_lot15* variable is an indication of the total average square footage of the land lots of the nearest 15 neighbours, the ratio computed then serves to ensure that the soon-to-be imputed value will not deviate too much from that of the 15 nearest neighbours. Imputation as shown in Code Block Number 4, was performed on the outlier identified by multiplying the outlier value with the average ratio previously calculated. The following Code Block then went on to display the resulting imputation where the *sqft_lot* for that particular observation is now approximately 254996 instead of 533610. Such imputed value however still did not make sense, because given that the square footage of the interior living space is only 800 square feet, the *sqft_lot* of 254196 of land lots (after taking into account the interior living space) is still too big of a plot of land for such a small living space. That said, Code Block Number 6 was executed to remove the revised observation from the

dataset. The removal was successful because, when asked to display the observation with the particular value of *sqft_lot 15*, no observations were found.

3.2 Exploratory Data Analysis

Having performed all of the necessary pre-processing steps, Exploratory Data Analysis (EDA) will then be performed on the already cleaned dataset. The purpose of performing EDA is not just to provide for a clearer picture of what the data looks like, but at the same time to also help in identifying patterns, correlations, and potential relationships between variables. That said, by better understanding the dataset, and the relationships between variables, features creation could then be more appropriately guided, essentially benefiting the predictive modeling process. The EDA performed include the investigation of the metadata, the descriptive statistics, the checking for missing values, the univariate visualization for each attribute, the visualization for potential relationships between variables, and lastly, the correlation analysis on the dataset.

No.	Source Code:	Output:
	Metadata	
1	603 /* ------------------------ */ 604 /* EXPLORATORY DATA ANALYSIS */ 605 /* ------------------------ */ 606 607 *Printing the Metadata for the house_imputed to once again identify the data types of each attributes; 608 proc contents data = house_imputed; 609 run;	

Alphabetic List of Variables and Attributes

#	Variable	Type	Len	Format	Informat
18	bathrooms	Num	8	BEST12.	
2	bedrooms	Num	8	BEST12.	BEST32.
8	condition	Num	8	BEST12.	BEST32.
5	floors	Num	8	BEST12.	BEST32.
9	grade	Num	8	BEST12.	BEST32.
14	lat	Num	8	BEST12.	BEST32.
15	long	Num	8	BEST12.	BEST32.
1	price	Num	8	BEST12.	BEST32.
13	renovation	Num	8	BEST12.	BEST32.
10	sqft_above	Num	8	BEST12.	BEST32.
11	sqft_basement	Num	8	BEST12.	BEST32.
3	sqft_living	Num	8	BEST12.	BEST32.
16	sqft_living15	Num	8	BEST12.	BEST32.
4	sqft_lot	Num	8	BEST12.	BEST32.
17	sqft_lot15	Num	8	BEST12.	BEST32.
7	view	Num	8	BEST12.	BEST32.
6	waterfront	Num	8	BEST12.	BEST32.
19	year_of_sale	Num	8		
12	yr_built	Num	8	BEST12.	BEST32.

The metadata above shows that there are now only 19 variables instead of the initial 21 variables prior to pre-processing. All of the variables are now in their rightful data type.

Descriptive Statistics

Source Code:

```
/* Descriptive Statistics */
proc means data=house_imputed mean median mode std var min max;
run;
```

Output:

The MEANS Procedure

Variable	Mean	Median	Mode	Std Dev	Variance	Minimum	Maximum
price	543406.42	459975.00	400000.00	368893.45	136082375421	82000.00	5300000.00
bedrooms	3.3444137	3.0000000	3.0000000	0.8870304	0.7868230	1.0000000	8.0000000
sqft_living	2061.66	1880.00	1640.00	915.1676845	837531.89	390.0000000	8010.00
sqft_lot	14838.30	7560.00	5000.00	38119.19	1453072736	690.0000000	920423.00
floors	1.4913820	1.5000000	1.0000000	0.5433087	0.2951843	1.0000000	3.5000000
waterfront	0.0095414	0	0	0.0972279	0.0094533	0	1.0000000
view	0.2333026	0	0	0.7686829	0.5908734	0	4.0000000
condition	3.4145891	3.0000000	3.0000000	0.6481663	0.4201196	1.0000000	5.0000000
grade	7.6589720	7.0000000	7.0000000	1.1791681	1.3904375	4.0000000	13.0000000
sqft_above	1771.28	1540.00	1300.00	817.8109933	668814.82	390.0000000	6220.00
sqft_basement	288.8959680	0	0	442.9941005	196243.77	0	3500.00
yr_built	1970.75	1975.00	2014.00	29.6070895	876.5797464	1900.00	2015.00
renovation	0.0467836	0	0	0.2112076	0.0446086	0	1.0000000
lat	47.5636956	47.5776000	47.6845000	0.1383071	0.0191289	47.1776000	47.7775000
long	-122.2147230	-122.2290000	-122.3650000	0.1401715	0.0196480	-122.5060000	-121.3160000
sqft_living15	1983.97	1840.00	1440.00	687.5229380	472687.79	399.0000000	5790.00
sqft_lot15	12645.01	7563.50	5000.00	26463.88	700337171	659.0000000	434728.00
bathrooms	2.1004155	2.2500000	2.5000000	0.7633363	0.5826822	0.7500000	6.0000000
year_of_sale	2014.32	2014.00	2014.00	0.4671766	0.2182540	2014.00	2015.00

From the descriptive statistics shown above, the average sale price of the houses within the dataset is approximately $543406 on average, but with substantial variability among the prices. In terms of the interior of the houses, most of them have a total of three bedrooms and two bathrooms, with an approximate living space of 2061 square feet. The majority of the houses within the dataset are not overlooking a waterfront, and the average rating of the condition of the houses are deemed as moderately average. Notably, there is however a wide range in terms of the lot sizes, essentially suggesting that there is a great mix of housing types within the dataset. In terms of geographical location, most houses predominantly lie in the coordinates of approximately (47.56, -122.21). Houses within the dataset are also of different age, where the age goes far back to the 1900s up until as recent as 2015. Most houses have not undergone any renovations. Other features like views and grades also serve to provide additional perspective on the housing characteristics of the properties within the dataset.

Missing Values

Source Code:

```
615 /* Checking for missing values */
616 proc means data=house_imputed nmiss;
617 run;
```

Output:

The MEANS Procedure

Variable	N Miss
price	0
bedrooms	0
sqft_living	0
sqft_lot	0
floors	0
waterfront	0
view	0
condition	0
grade	0
sqft_above	0
sqft_basement	0
yr_built	0
renovation	0
lat	0
long	0
sqft_living15	0
sqft_lot15	0
bathrooms	0
year_of_sale	0

Making reference to the output shown above, there are no longer any missing values within the attributes of the dataset.

Visualization for Ordinal and Dummy Variables

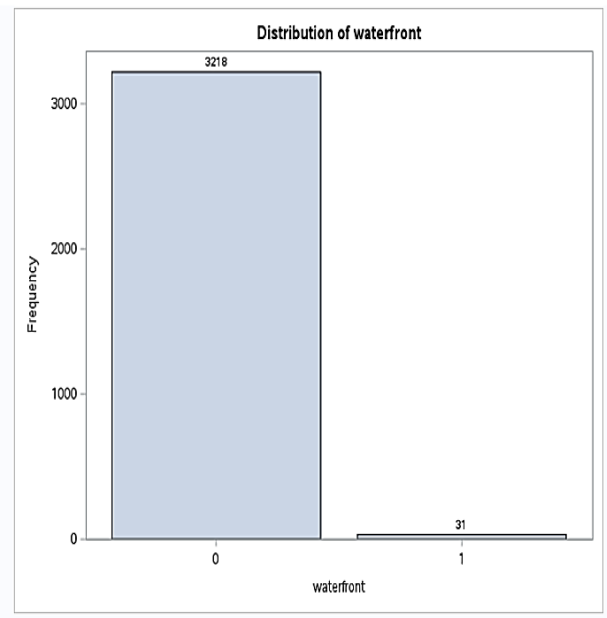
Source Code:

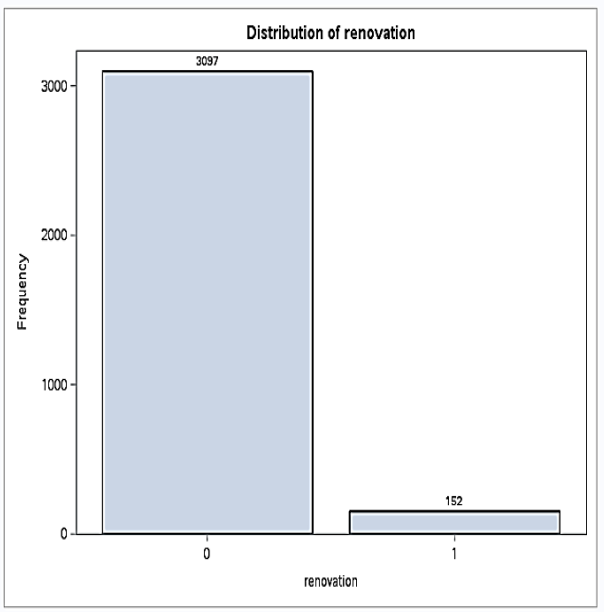
```
/* ORDINAL and DUMMY VARIABLES */
/* Create a subset of the original dataset with only the specified categorical and ordinal variables */
data subset_cats;
    set house_imputed(keep = waterfront renovation view condition grade bedrooms bathrooms floors);
    /* Convert the numeric variables to character to treat them as categorical */
    waterfront_c = put(waterfront, 1.);
    renovation_c = put(renovation, 1.);
    view_c = put(view, 1.);
    condition_c = put(condition, 1.);
    grade_c = put(grade, 1.);
run;
```

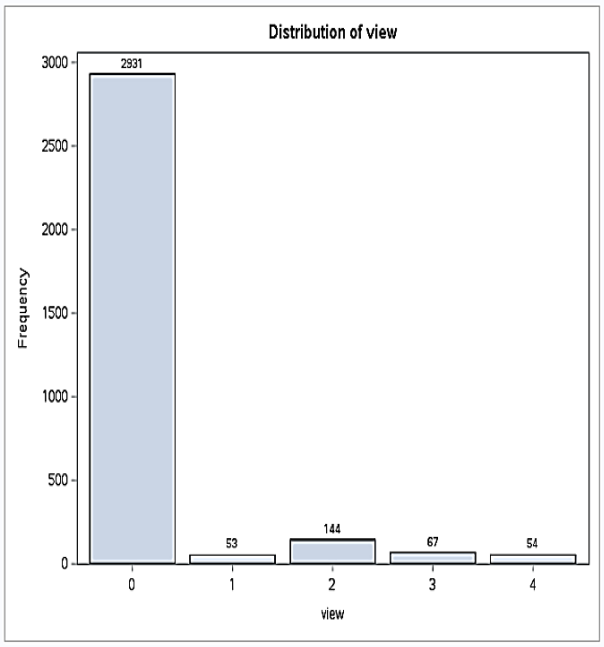
```
/* Define a macro to generate bar plots for a list of categorical variables */
%macro plotBars(data, varlist);

    /* Count the number of variables in the provided list */
    %let numVars = %sysfunc(countw(&varlist));

    /* Loop through each variable in the list */
    %do i = 1 %to &numVars;
        /* Extract the current variable name */
        %let currentVar = %scan(&varlist, &i);

        /* Generate a bar plot for the current variable */
        proc sgplot data=&data;
            vbar &currentVar / datalabel;
            title "Distribution of &currentVar";
        run;

    %end; /* End of loop */

%mend; /* End of macro definition */

/* Call the macro with the dataset name 'subset_cats' and the list of ordinal and dummy variables */
%plotBars(subset_cats, waterfront renovation view condition grade bedrooms bathrooms floors);
```

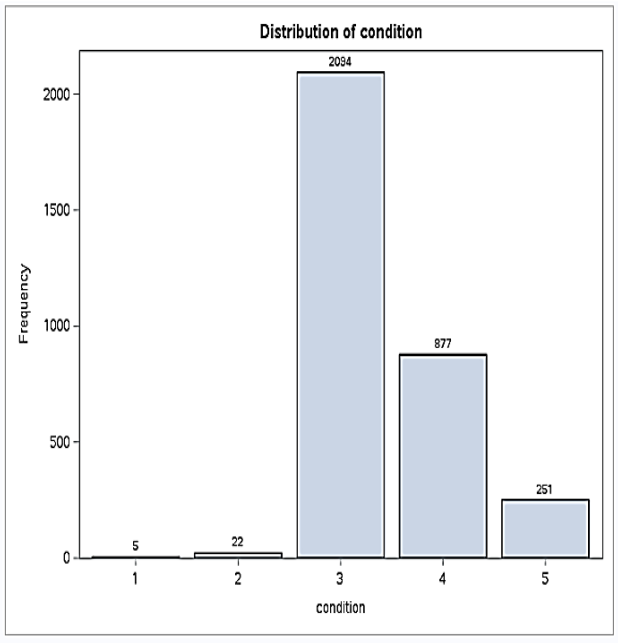
The source code above consisted firstly of creating a subset of the original dataset with only the variables that are categorical and ordinal in nature, while the following step involved the generating of the individual bar plots for the list of variables within the subset which was previously created. The specified categorical and ordinal variables are namely, *waterfront*, *renovation*, *view*, *condition*, *grade*, *bedrooms*, *bathrooms* and *floors*. Note that, similar reasoning as to why the *bedrooms*, *bathrooms*, and *floors* attributes were treating as categorical variables when performing imputation for missing values applies here too.

Output:

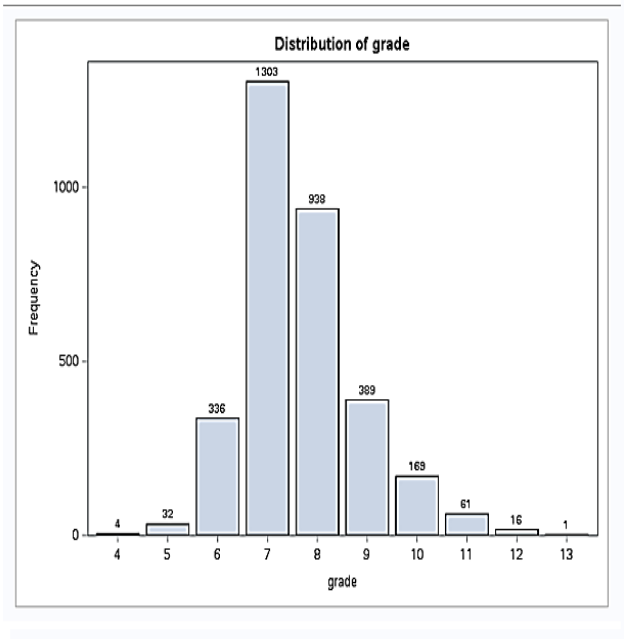


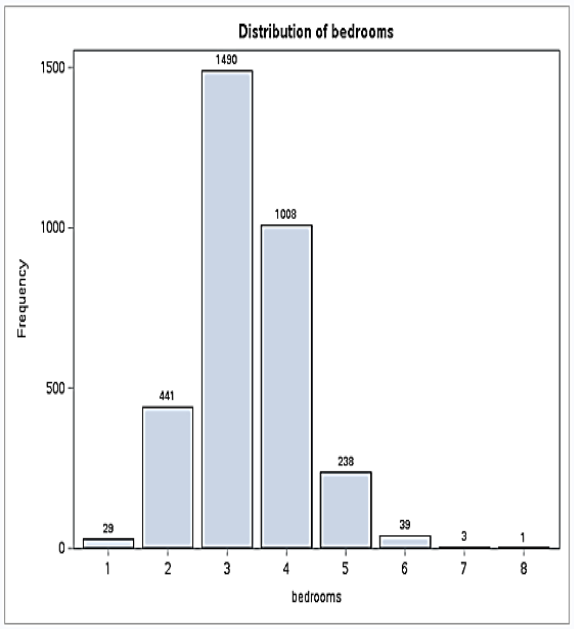


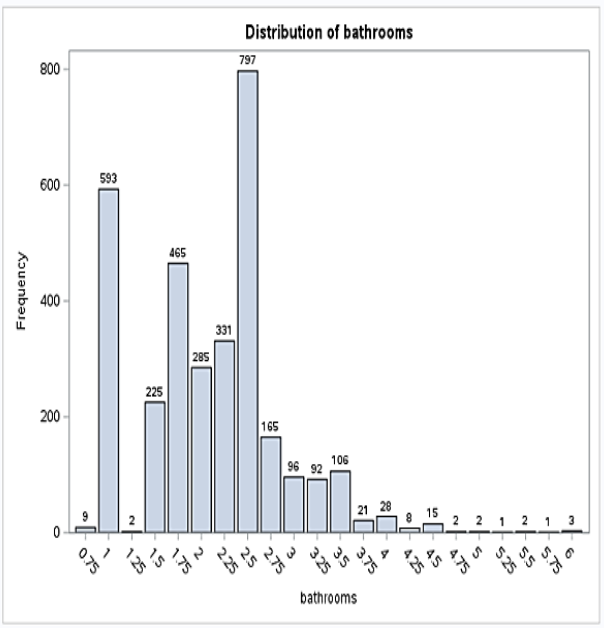


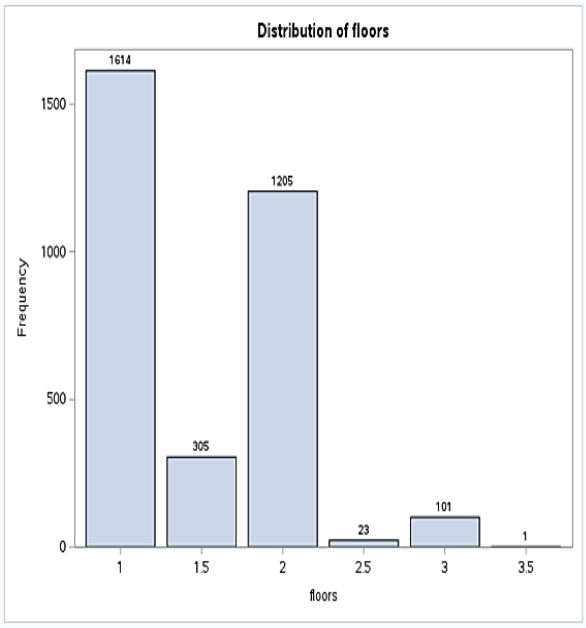


The bar plots above suggests that the majority of the houses within the dataset do not overlook a waterfront and that most of them has never undergo any renovation work before. Most houses also had no views at all, and that the current state of these houses, as indicated by the *condition* attribute, are mostly considered as moderately average.









From the four bar plots shown above, it is observed that most houses have a grade rating of 7, suggesting that these houses are of the average level in terms of their structures, furnishings, and construction. Majority of the houses also tended to have 3 bedrooms, and 2.5 bathrooms (meaning, there are two full bathrooms with a sink, toilet, tub, and shower, plus 0.5 of the bathrooms which includes only the sink and the toilet), and are mostly single-storied.

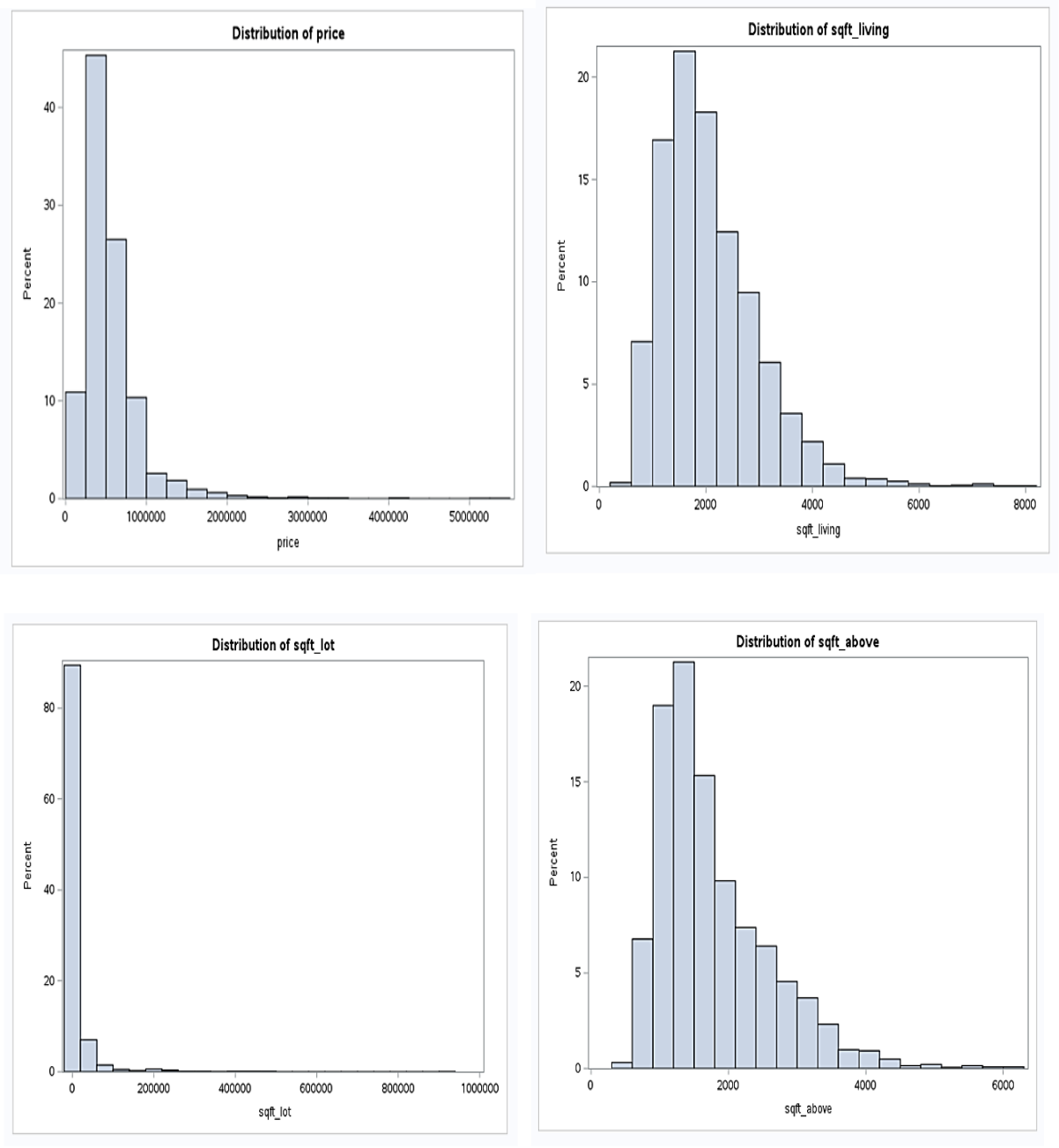
Visualization for Continuous Variables

Source Code:

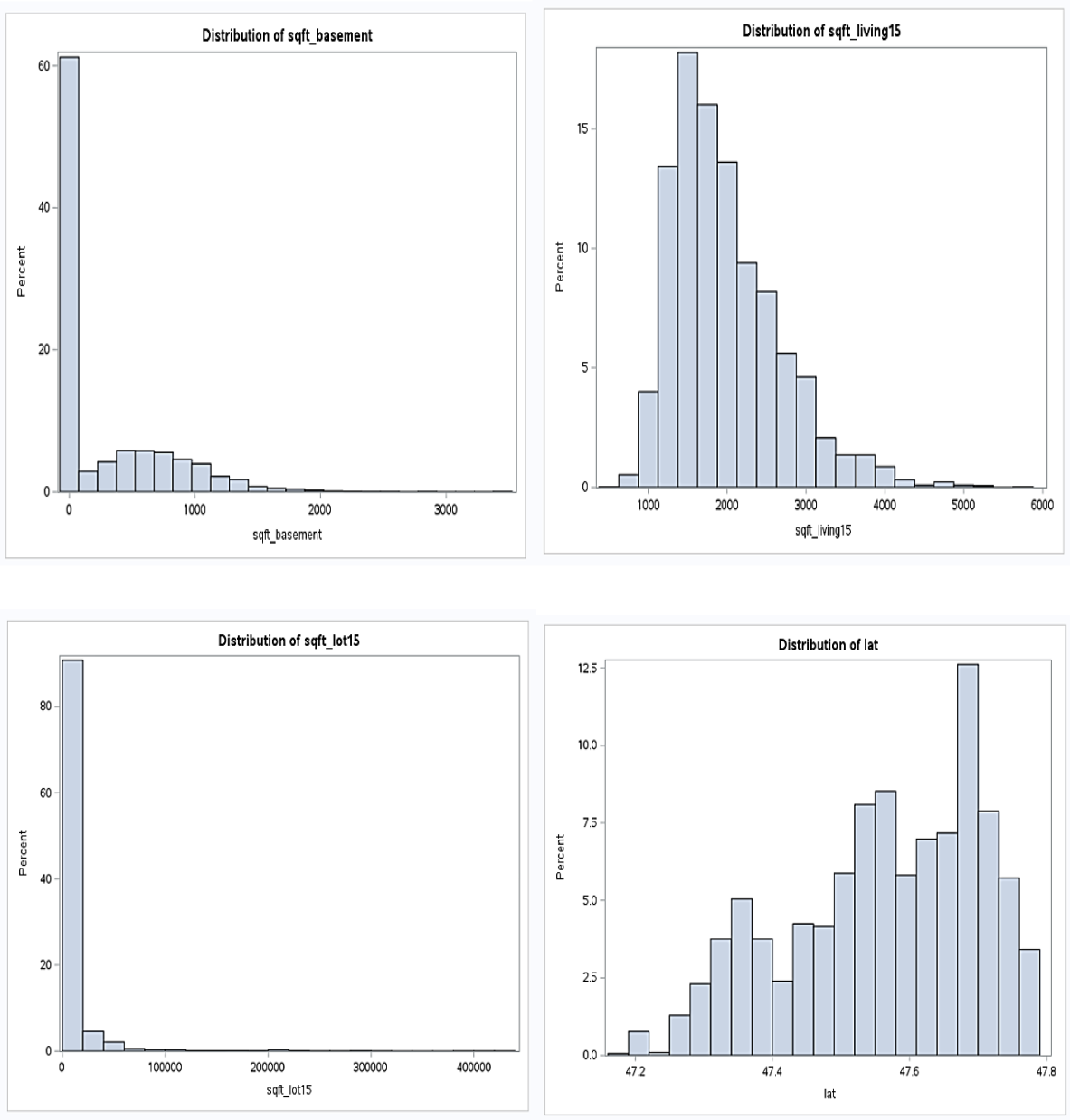
```
/*CONTINUOUS VARIABLES */
/* Create a subset of the original dataset with only the specified continuous variables */
data subset;
    set house_imputed(keep = price sqft_living sqft_lot sqft_above sqft_basement
                             sqft_living15 sqft_lot15 lat long yr_built year_of_sale);
run;

/* Define a macro to generate histograms for a list of variables */
%macro plotHistograms(data, varlist);

    /* Count the number of variables in the provided list */
    %let numVars = %sysfunc(countw(&varlist));

    /* Loop through each variable in the list */
    %do i = 1 %to &numVars;
        /* Extract the current variable name */
        %let currentVar = %scan(&varlist, &i);

        /* Generate a histogram for the current variable */
        proc sgplot data=&data;
            histogram &currentVar;
            title "Distribution of &currentVar";
        run;

    %end; /* End of loop */

%mend; /* End of macro definition */

/* Call the macro with the dataset name 'subset' and the list of numeric variables */
%plotHistograms(subset, price sqft_living sqft_lot sqft_above sqft_basement
                             sqft_living15 sqft_lot15 lat long yr_built year_of_sale);

```

The source code above was executed to display the respective histograms of all continuous variables within the dataset. Similar to that of the categorical and ordinal variables, the code began firstly by creating a subset of the original dataset with only the specified continuous variables (*price*, *sqft_living*, *sqft_lot*, *sqft_above*, *sqft_basement*, *sqft_living15*, *sqft_lot15*, *lat*, *long*, *yr_built*, and *year_of_sale*). Following that, the individual histograms for each continuous attribute is then plotted as shown below.

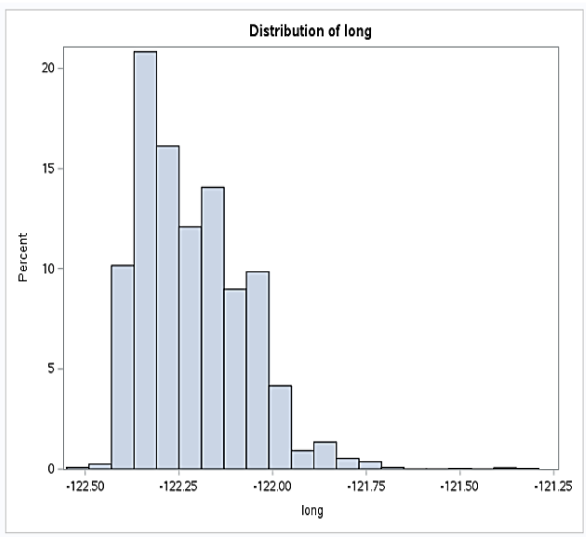
Output:

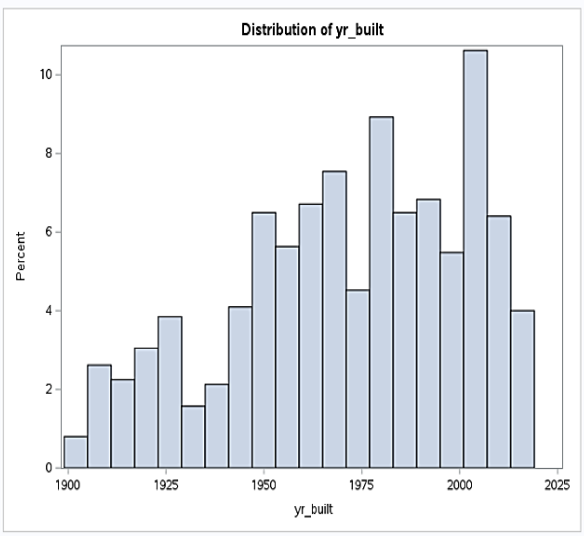


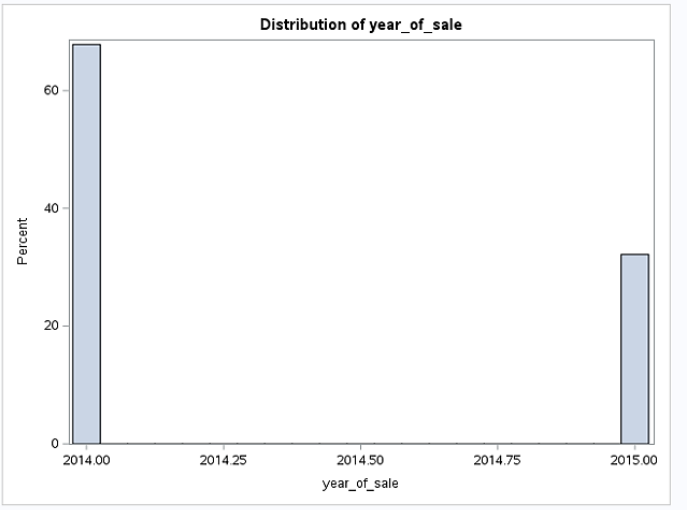
All four histograms shown above suggests that the distributions of these attributes are positively skewed. These positively skewed distributions suggest that while most houses have followed their respective average values, there are however a few houses with very high values for the four attributes listed above. Nevertheless, it is important to note that such positively skewed distributions in housing dataset are not uncommon, as luxury houses or high-end properties are naturally always fewer in number, when compared to more affordable or standard houses.



Both distribution of the *sqft_basement* and *sqft_lot15* are skewed to the right. This suggests that even though houses with larger basements and large neighbouring lot sizes are uncommon, there are however not impossible to have houses with such attributes. The distribution *sqft_living15* attribute on the other hand, has an almost normal distribution, essentially suggesting that there is a balance in the distribution of smaller and larger houses in the case of the neighbouring communities. The multimodal nature of the *lat* attribute however, suggests that there exists specific latitudinal bands where houses are densely clustered. These latitudinal bands could represent specific neighbourhoods or regions within the geographical area covered by the dataset.



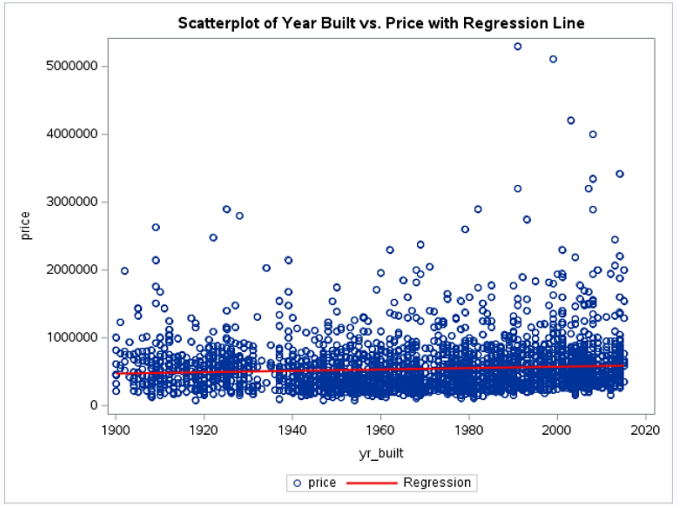




The distribution of the *long* attribute suggests that there is a concentration of houses in specific longitudinal bands, possibly pointing to certain densely populated areas or neighbourhoods within the geographical area covered by the dataset. The distribution for the *yr_built* attribute however, indicates that most houses within the dataset are from the latter half of the 20th century onwards. This suggests that there might be newer developments or perhaps an increased in popularity or growth within the King County area during recent decades. Lastly, the *year_of_sale* distribution is clearly temporal in nature, where house sales are seen to be focused primarily during the year 2014.

Visualization of Potential Relationship between Attributes

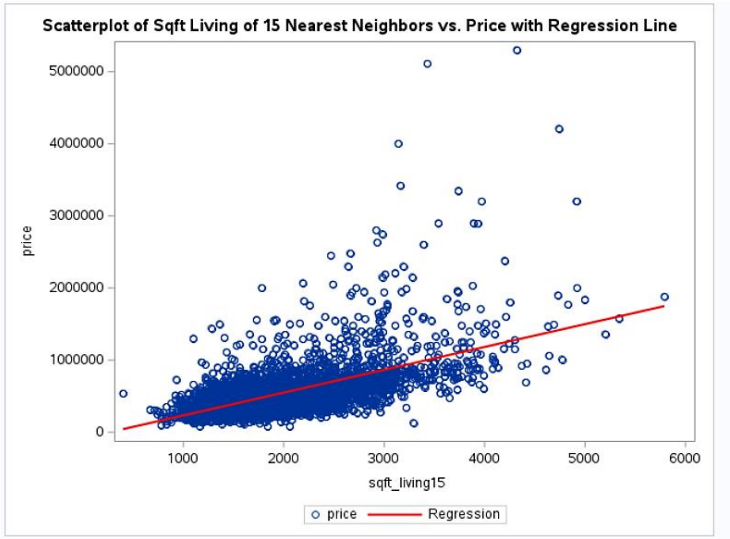
To ensure consistency with the five hypothesis statements identified in this paper, and subsequently the testing of those hypotheses, which will be further discussed in *Section 4.0*, note that only variables relevant to the hypotheses and their interrelationships will be discussed. Nonetheless, this is not to say that the potential relationships between attributes are only limited to those listed down below. In this section, scatterplots are used to determine the direction of the relationship between the pairs of continuous variables, without any mention of the respective magnitudes of their relationships. Note that, while scatterplots may be indicative of the magnitude of the relationship between the pairs, a correlation table will be referred to instead, because all of the values would then be explicitly computed in the table. Since one of the pairs is between a continuous variable ("*price*") and a categorical variable ("*view*"), a boxplot was then used in that case.

No.	Source Code:	Output:
1	690 /* Price vs House Age */ 691 /* Scatter plot with regression line for Price vs. Year Built*/ 692 proc sgplot data=house_imputed; 693 scatter x=yr_built y=price; 694 reg x=yr_built y=price / degree=1 lineattrs=(color=red); /* This adds a linear regression line in red color */ 695 title "Scatterplot of Year Built vs. Price with Regression Line"; 696 run;	

The Code Block above was executed to plot a scatter plot with regression line between the *price* and *yr_built* attribute. From the plot shown in the output column on the right, it appears that there is a positive relationship between the two variables. That is, for houses that are built more recently, prices tended to be higher too. Nevertheless, it is also worth noting that there are several older houses that have high prices, possibly due to other factors which are not shown in the plot above, for instance, lot size, location, or historical significance.

2

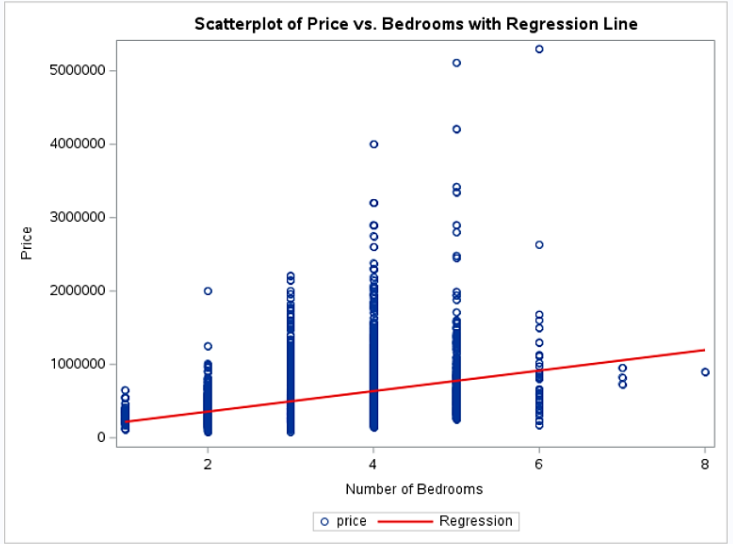
```
698 /* Price vs Neighbourhood Living Space */
699 /* Scatter plot with regression line for Price vs. Neighbourhood Living Space */
700 proc sgplot data=house_imputed;
701     scatter x=sqft_living15 y=price;
702     reg x=sqft_living15 y=price / degree=1 lineattrs=(color=red); /* This adds a linear regression line in red color */
703     title "Scatterplot of Sqft Living of 15 Nearest Neighbors vs. Price with Regression Line";
704 run;
```



The scatterplot above depicts the relationship between the total average living space of the 15 nearest neighbors (in square feet) and the respective prices of the houses sold. From the scatterplot, it is evident that there is a positive relationship between the two variables, implying that as the average interior living space of the 15 nearest neighbors increases, the prices tend to increase as well. This positive relationship observed possibly suggest that houses located in neighborhoods with larger properties generally command higher prices. Nevertheless, it is worth noting that there is still a notable spread in the data points, meaning, other factors which are not represented in the plot above, might also influence the house prices.

3

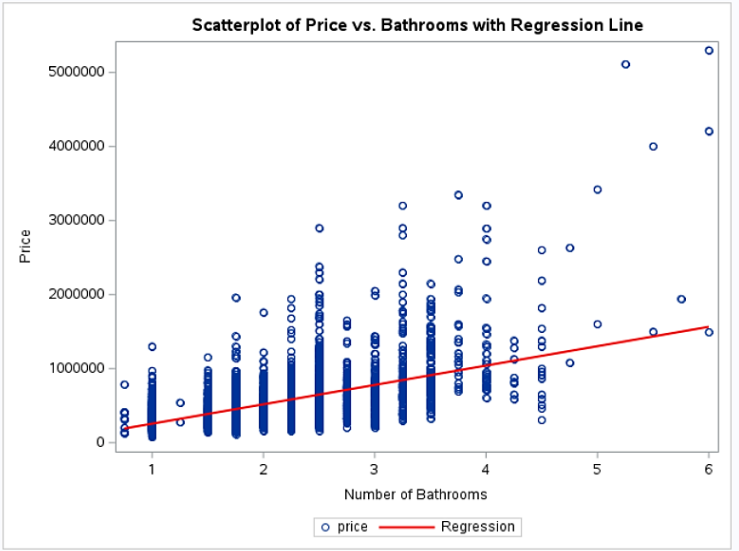
```
/* Price vs Bedrooms and Bathrooms */
/* Scatter plot with regression line for Price vs. Bedrooms */
proc sgplot data=house_imputed;
    scatter y=price x=bedrooms;
    reg y=price x=bedrooms / lineattrs=(color=red); /* Adds a regression line */
    xaxis label="Number of Bedrooms";
    yaxis label="Price";
    title "Scatterplot of Price vs. Bedrooms with Regression Line";
run;
```



Code Block Number 3 relates to the plotting of a scatterplot with regression line between the *price* and the *bedrooms* variable. Based on the regression line, there appeared to be a positive relationship between the two variables, where as the number of bedrooms increases, sale price of houses tends to increase as well. Having said that, the vertical spread of data points for each bedroom number suggests that there is considerable variability in the prices within each category. At the same time, it may be worth mentioning that while houses with 2,4, and 6 bedrooms showed wider range in terms of prices, those with 3 and 5 bedrooms however, seemed to have a more consistent price range. Possible implications could be that other factors which were not included in the scatterplot, might be affecting to the prices within these categories.

4

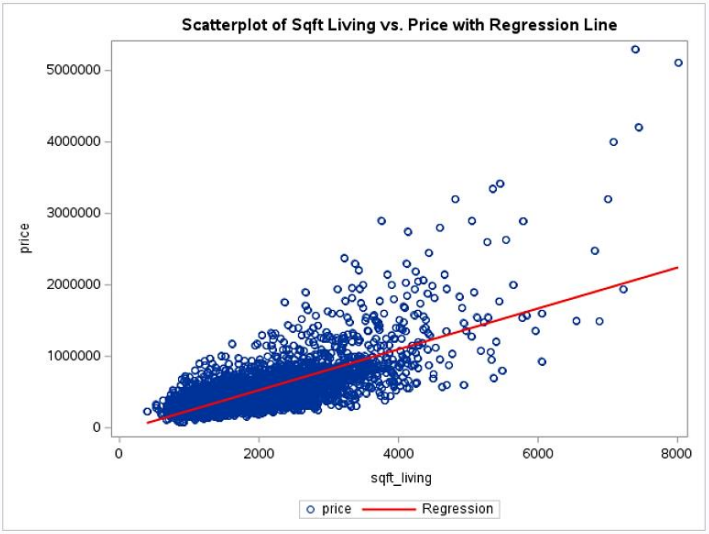
```
/* Scatter plot with regression line for Price vs. Bathrooms */
proc sgplot data=house_imputed;
    scatter y=price x=bathrooms;
    reg y=price x=bathrooms / lineattrs=(color=red); /* Adds a regression line */
    xaxis label="Number of Bathrooms";
    yaxis label="Price";
    title "Scatterplot of Price vs. Bathrooms with Regression Line";
run;
```



The scatterplot for the relationship between the sale price of a house ("*price*") and its respective number of bathrooms ("*bathrooms*") suggests that there exists a positive relationship between the two variables. That is, houses with more bathrooms typically command higher prices. Besides that, the density of the data points around 2 to 3 bathrooms proposes that these are the typical standards for most houses. Nonetheless, as more bathrooms are added beyond that point, the number of houses tended to decrease, but their sale prices still remained noticeably higher. Again, it is important to note that, given the wide distribution of prices for houses with the same number of bathrooms, it is very likely that while the number of bathrooms plays a role in determining the prices, other significant factors could be concurrently at play too.

5

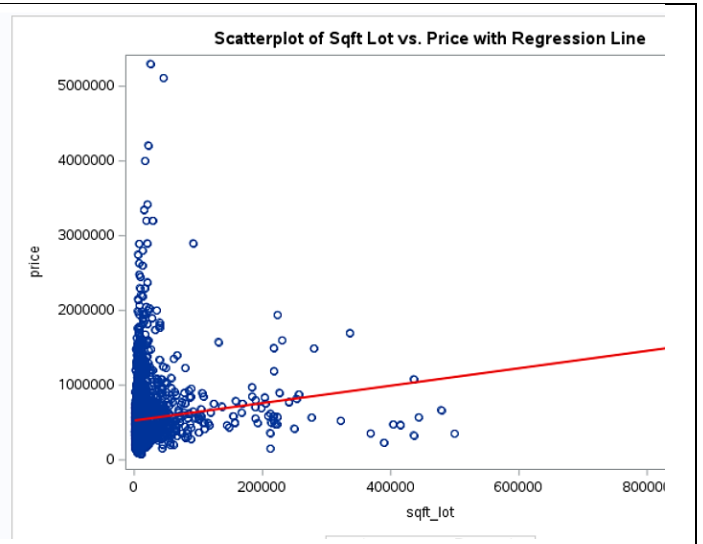
```
/*Price vs Living and Lot Utilization */
/* Scatterplot of sqft_living vs. Price with Regression Line */
proc sgplot data=house_imputed;
    scatter x=sqft_living y=price;
    reg x=sqft_living y=price / degree=1 lineattrs=(color=red); /* Add a linear regression line in red color */
    title "Scatterplot of Sqft Living vs. Price with Regression Line";
run;
```



The scatterplot shown above relates to the relationship between the square footage of the interior living space of a house ("*sqft_iving*") and its corresponding price ("*price*"). The regression line drawn suggest that there is a positive correlation between the two variables, whereby as the size of the interior living space of a house increases, the price of the house would then increase as well. The dense clustering of data points in the lower square footage ranges reveals that most houses within the dataset do fall within this range of house price and interior living space. Nevertheless, it is evident that as the living space increases in size, the spread in prices tended to be more prominent as well, essentially suggesting that there is greater variability in terms of prices for larger homes. This reveals that while the size of the interior living space of a house is an important factor when determining its sale price, other attributes are likely to have played a role too, more so when concerning properties with larger interior living space.

6

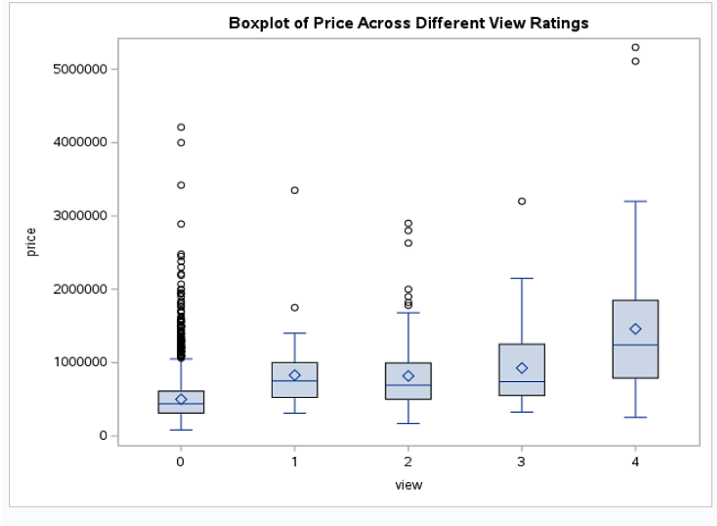
```
735 /* Scatterplot of sqft_lot vs. Price with Regression Line */
736 proc sgplot data=house_imputed;
737     scatter x=sqft_lot y=price;
738     reg x=sqft_lot y=price / degree=1 lineattrs=(color=red); /* Add a linear regression line in red color */
739     title "Scatterplot of Sqft Lot vs. Price with Regression Line";
740 run;
```



The Code Block above pertains to the plotting of a scatterplot between the total square footage of the land lot of a house (“sqft_lot”) and that of its sale price (“price”). The mild upward trend of the regression line demonstrates that there is a positive relationship between the lot size and its sale price. Having said that, the dense clustering of data points at the lower left end of the plot suggests that while many houses do have smaller lot sizes, their respective prices do however, vary quite considerably. At the same time, as the lot size increases, data points became more dispersed too, once again implying variability in prices for houses with larger lots. Put simply, the variability of data points observed within the scatterplot signals that while lot size of a house does influence its sale price, other attributes relating to the house could play a role as well in determining the property’s price, especially for those with smaller lot sizes.

7

```
743 /*Price vs View */
744 /* SGPLOT Procedure to Visualize Data Graphically */
745 proc sgplot data=house_imputed;  /* Use the 'house_imputed' dataset for the plot */
746
747     /* Generate a vertical boxplot of 'price' across different categories of 'view' */
748     vbox price / category=view;
749
750     /* Set the title for the plot */
751     title "Boxplot of Price Across Different View Ratings";
752
753 /* End the SGPLOT procedure */
754 run;
```



Code Block Number 7 was executed to graphically visualize the distribution of the sale price of houses across different view ratings. From the boxplot above, one observation made is that houses with a view rating of 0 tended to have the lowest median price, with a considerable number of outliers in the higher price ranges. As the view rating increases however, the median price of houses generally increases as well, ultimately implying that houses with better views tend to be on the pricier end. Besides, as view rating increases, the spread of the sale price, as represented by the interquartile range, was observed to have increased as well. Implication of this is that, while houses with a better view are associated with higher prices, there is however greater variability in prices within categories of higher ratings. Such implication is especially evident in the case of houses with a view rating of 4, which is the highest rating achievable. Houses in such category have had the broadest price range, from the lowest to some of the highest sale price. This suggests that diversity in house values do still exist even within the best view category. Generally, all observations made above points to the direction that, while the view of a house do play an important role in determining its sale price, other factors are likely to be at play too, which explains why there exist a broad price variability within each rating category.

Correlation Analysis

Source Code:

```
/* ------------------------- */
/* CORRELATION ANALYSIS */
/* ------------------------- */

/* Compute correlations for continuous and ordinal variables */
proc corr data=house_imputed out=corr_out noprob;
    var bathrooms bedrooms floors lat long price sqft_above sqft_basement sqft_living
    sqft_living15 sqft_lot sqft_lot15 yr_built year_of_sale view condition grade waterfront renovation;
run;


/* Sort the corr_out dataset based on _NAME_ and _TYPE_
   This is a necessary step before using PROC TRANSPOSE on data with BY groups. */
proc sort data=corr_out;
    by _NAME_ _TYPE_;
run;

/* Transpose the data from a wide format to a long format.
   In the original data, each row represented a variable's correlation with every other variable in columns.
   After transpose, each row will represent a correlation between two specific variables.

   The name= option specifies a new name for the column that holds the names of the original variables.
   The rename= option changes the default column name 'col1' to 'Value' which holds the correlation values. */
proc transpose data=corr_out out=long_format_correlations (rename=(col1=Value))
                name=Variable;
    by _NAME_ _TYPE_;
    /* List of variables for which correlations were computed */
    var bathrooms bedrooms floors lat long price sqft_above sqft_basement sqft_living
    sqft_living15 sqft_lot sqft_lot15 yr_built year_of_sale view condition grade waterfront renovation;
run;
```

```
/* Filter the correlations from the transposed data.
   Here, we are interested in correlations (specified by _TYPE_ = 'CORR')
   that have an absolute value greater than 0.7 and aren't self-correlations (i.e., variable with itself). */
proc sql;
    create table filtered_correlations as
    select
        _NAME_ as Var1,  /* Variable name from the original data */
        Variable as Var2,  /* Transposed variable name from the original data */
        Value as Correlation  /* Correlation value between Var1 and Var2 */
    from
        long_format_correlations
    where upcase(_TYPE_) = 'CORR'  /* Only pick correlation values, not N or other types */
    and abs(Value) > 0.7  /* Filter strong correlations (either positive or negative) */
    and _NAME_ ne Variable;  /* Exclude self-correlations */
quit;

/* Print the table of filtered correlations.
   This will display the correlations in a table format in the SAS output window. */
proc print data=filtered_correlations;
    title "Filtered Correlations (Greater than 0.7)";
run;
```

Output:

Filtered Correlations for Specific Pairs

Obs	Var1	Var2	Correlation
1	bathrooms	price	0.54136
2	bedrooms	price	0.33639
3	price	sqft_living	0.70914
4	price	sqft_living15	0.58989
5	price	sqft_lot	0.12041
6	price	yr_built	0.08160
7	price	view	0.40057

The source code above relates to the computation of the correlation for all continuous and ordinal variables within the dataset. Once the correlation has been computed, the *corr_out* dataset is then sorted based on the name of its variables, followed by their respective type. After that, and as indicated from code line 774 to 786, the sorted dataset is then transposed from a wide format to a long format. This is to ensure that after transposing, each row will now represent a correlation between two specific variables, as could be seen in the output above. While at it, it is important to note that data transposition is one form of feature engineering. Following that, only the correlation values of the attributes of interest will be filtered and stored into a dataset called "filtered_correlations". From the output, observation made is that, consistent with the findings from the scatterplots and boxplot above, all seven pairs of attributes are positively related to one another.

Taking a correlation value of 0.7 as the threshold for what is considered as a strong correlation, only the pair between *price* and *sqft_living* was found to be strongly correlated. The rest of the pairs were either moderately or weakly correlated to one another. Having said that, looking at the *bedrooms* and *bathrooms* variable separately might not be that intuitive, because both are often seen as a duo, meaning, in every house, it must not be without a bathroom nor could it be without a bedroom. Hence, feature creation in terms a bedroom-to -bathroom ratio might provide better insights as compared to evaluating both variables individually. Besides that, a correlation between *price* and *yr_built* might not be meaningful at all, because it does not make sense to strictly say that a house built in 2013 will have a higher price compared to a one which was built in 2015, or vice versa. Put simply, taking the *yr_built* variable at its face value would just prove to be redundant. However, when feature engineering is performed on it, the *yr_built* attribute will then be able to provide useful information, for instance, information about the age of the house at the time of the sale. Thus far, there has been two needs identified for feature creation. The last feature to be created would be that relating to how the spaces of the land of

the house ("*sqft_lot*") was utilized for interior living space ("*sqft_living*"). Again, this is because looking at lot utilization might not be that insightful when both variables were examined individually. Lastly, it is important to note that, even though the *view* and *sqft_living15* variables were found to only have a moderate positive relationship with the *price* variable, it is still necessary to evaluate them because, real estate valuation is inherently multifaceted. In other words, the value of a property is not just determined by an attribute alone, but instead, it is influenced by a combination of different factors working in tandem. Hence, even if the two variables mentioned are not highly correlated to the house price, they could still play a crucial role when combined with other variables.

3.3 Feature Engineering

This section includes the discussion on the six feature engineering processes performed, namely, date extraction, data transposition, feature creation, transformation, scaling, and one-hot encoding. Note that while certain feature engineering processes were performed in the beginning of the data analytics process (in an earlier section prior to this one), certain processes were however performed in later parts of the analytics process. Sequencing of these feature engineering processes would however, depend solely on the need, logic and appropriateness behind their execution.

3.3.1 Date Extraction

Refer to *Section 3.1.3* for the source code and the output. Reasonings as to why date extraction was performed at the beginning could also be found in the said section.

3.3.2 Data Transposition

Note that data transposition has been performed in conjunction with the correlation analysis documented in *Section 3.2 Exploratory Data Analysis*. Refer to the *Correlation Analysis* part under the said section for the source and the output.

3.3.3 Feature Creation

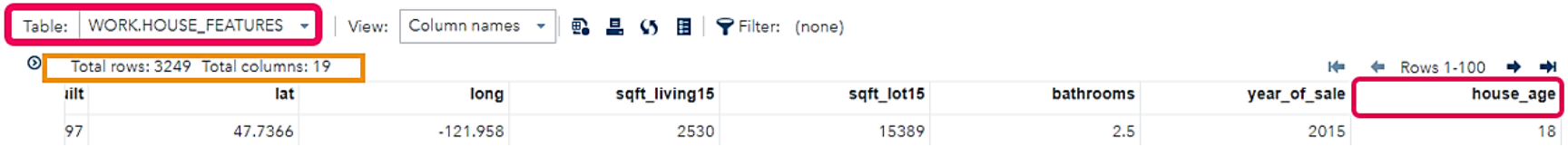
Three features as mentioned previously will be created. They are namely, *house_age*, *bed_bath_ratio*, and *lot_utilization*.

3.3.3.1 Feature Creation for Hypothesis 1

No.	Source Code:
1	

```
611 /* house_age VARIABLE */
612
613 data house_features;
614     set house_imputed;
615     /* Calculate the age of the house at the time of sale */
616     /* This will be used to test if older houses tend to have lower prices */
617     house_age = year_of_sale - yr_built;
618 run;
```

Output:



iilt	lat	long	sqft_living15	sqft_lot15	bathrooms	year_of_sale	house_age
97	47.7366	-121.958	2530	15389	2.5	2015	18

The Code Block above was executed to create a new variable called *house_age* and to store the newly created variable into a dataset called "house_features". The *house_age* attribute was created by finding the difference between the year in which the house was sold ("*year_of_sale*") and the year in which the house was built ("*yr_built*").

No.	Source Code:	Output:
2	See code block below	See output tables below

```
/* Summary Statistics */

proc means data=house_features mean median min max std;
    var house_age;
run;

/* Notice that the minimum value is -1 which is illogical given the context of the age of the house */

/* Identify which records have a negative house_age */
proc print data=house_features;
    where house_age < 0;
    var yr_built year_of_sale house_age;
    title "Houses with Negative Age";
run;
/* Since there is no way to determine whether the yr_built or year_of_sale ...*/
/*...is the incorrect value, these two records will be removed */
data house_features;
     set house_features;
     if house_age >= 0;
run;

/* Confirm Removal */
proc means data=house_features;
    var house_age;
run;
```

The MEANS Procedure

Analysis Variable : house_age

Mean	Median	Minimum	Maximum	Std Dev
43.5678670	39.0000000	-1.0000000	115.0000000	29.6015893

Houses with Negative Age

Obs	yr_built	year_of_sale	house_age
25	2015	2014	-1
2947	2015	2014	-1

The MEANS Procedure

Analysis Variable : house_age

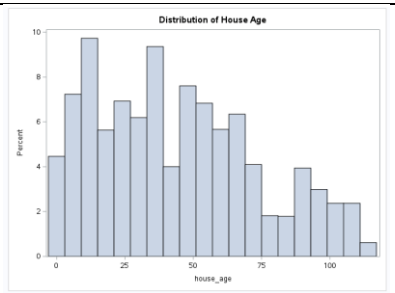
N	Mean	Std Dev	Minimum	Maximum
3247	43.5953188	29.5900218	0	115.0000000

The Code Block above provides the summary statistics for the *house_age* variable. The summarizing properties are as follows.

Summarizing Properties:

- There is no missing value.
- The mean value suggests that the age of most houses within the dataset are approximately 43.57 years old, on average.
- The standard deviation of 29.6015 indicates that there is significant variability within the *house_age* attribute.
- Inconsistency has been identified. That is, notice that the minimum value for the attribute is "-1", which is illogical given the context of the age of the house.

Code Block Number 2 proceeded to identify the location of the inconsistency. Following that, since there is no way to determine whether it was the *yr_built* attribute or the *year_of_sale* attribute which has caused the incorrect value, the entire observation will hence be removed. Removal was successful because the minimum value is now "0" instead of "-1".

3	648 /* Distribution */ 649 proc sgplot data=house_features; 650 histogram house_age; 651 title "Distribution of House Age"; 652 run;	

Code Block Number 3 was executed to display the distribution of the attribute. The distribution seemed to be skewed a bit to the right.

4

```
654 *Compute quartiles for the 'house_age' variable;
655 PROC UNIVARIATE DATA=house_features;
656     VAR house_age;
657     OUTPUT OUT=OutliersHouseAge (RENAME=(house_age=OriginalHouseAge))
658     Q1=Q1_house_age Q3=Q3_house_age;
659 RUN;
660
661 *Detect and store outliers for 'house_age' in the OutliersListHouseAge dataset;
662 DATA OutliersListHouseAge (keep=ObsNum OutlierValue);
663     SET house_features;
664     IF _N_ = 1 THEN SET OutliersHouseAge;
665
666     IQR = Q3_house_age - Q1_house_age;
667     LowerBound = Q1_house_age - 1.5 * IQR;
668     UpperBound = Q3_house_age + 1.5 * IQR;
669
670     /* Check if the house_age value is an outlier */
671     IF house_age < LowerBound OR house_age > UpperBound THEN DO;
672         ObsNum = _N_;
673         OutlierValue = house_age;
674         OUTPUT;
675     END;
676
677     DROP IQR LowerBound UpperBound Q1_house_age Q3_house_age;
678 RUN;
679
680 *Print detected outliers;
681 PROC PRINT DATA=OutliersListHouseAge;
682 RUN;
```

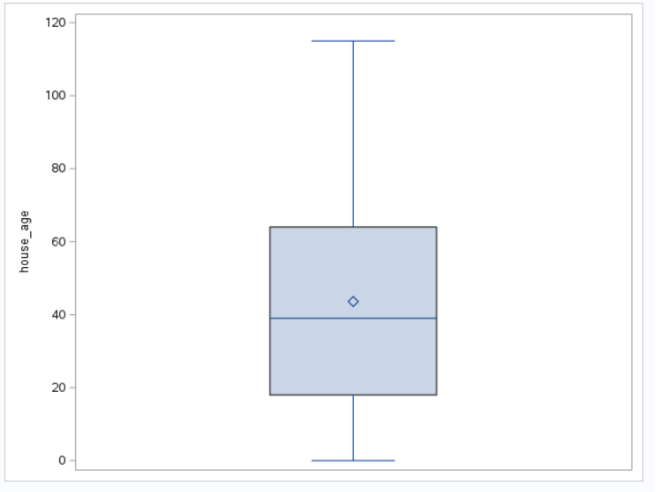
The UNIVARIATE Procedure
Variable: house_age

Moments			
N	3247	Sum Weights	3247
Mean	43.5953188	Sum Observations	141554
Std Deviation	29.5900218	Variance	875.569393
Skewness	0.459875	Kurtosis	-0.7157365
Uncorrected SS	9013190	Corrected SS	2842098.25
Coeff Variation	67.8743101	Std Error Mean	0.51928304

Basic Statistical Measures			
Location		Variability	
Mean	43.59532	Std Deviation	29.59002
Median	39.00000	Variance	875.56939
Mode	9.00000	Range	115.00000
		Interquartile Range	46.00000

Quantiles (Definition 5)	
Level	Quantile
100% Max	115
99%	109
95%	100
90%	89
75% Q3	64
50% Median	39
25% Q1	18
10%	8
5%	3
1%	0
0% Min	0

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0	3245	114	1964
0	3186	114	2274
0	3164	114	3158
0	3108	115	826
0	3101	115	1004

5	684 *Visualize 'house_age' distribution with a boxplot; 685 PROC SGPLOT DATA=house_features; 686 VBOX house_age; 687 RUN;	

Code Block Number 4 and 5 relates to the performing of outlier detection on the newly created variable. No outlier has been detected, as could be observed from the boxplot above.

3.3.3.2 Feature Creation for Hypothesis 3

No.	Source Code:
1	704 /* bed_bath_ratio VARIABLE */ 705 706 data house_features; 707 set house_features; 708 /* Calculate the ratio of bedrooms to bathrooms */ 709 /* This will be used to test if houses with more bedrooms per bathroom tend to be priced lower */ 710 bed_bath_ratio = bedrooms / bathrooms; 711 run;

Output:

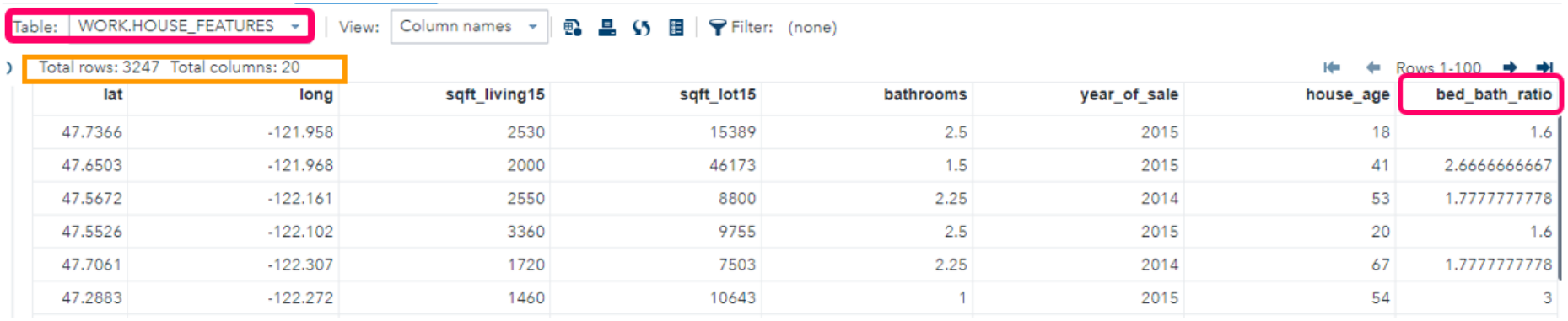


Table: WORK.HOUSE_FEATURES | View: Column names | Filter: (none)

Total rows: 3247 Total columns: 20 | Rows 1-100

lat	long	sqft_living15	sqft_lot15	bathrooms	year_of_sale	house_age	bed_bath_ratio
47.7366	-121.958	2530	15389	2.5	2015	18	1.6
47.6503	-121.968	2000	46173	1.5	2015	41	2.6666666667
47.5672	-122.161	2550	8800	2.25	2014	53	1.7777777778
47.5526	-122.102	3360	9755	2.5	2015	20	1.6
47.7061	-122.307	1720	7503	2.25	2014	67	1.7777777778
47.2883	-122.272	1460	10643	1	2015	54	3

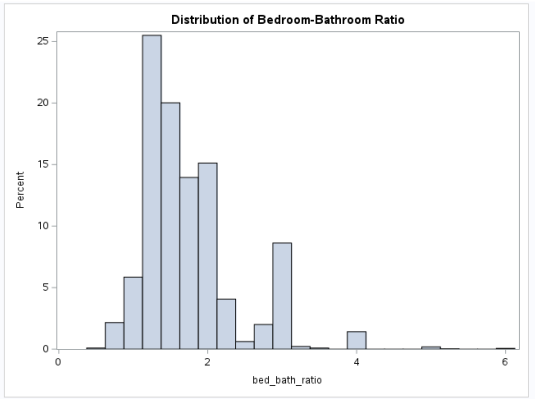
The Code Block above was executed to create a new variable called *bed_bath_ratio* and to store the newly created variable into a dataset called "house_features". The *bed_bath_ratio* attribute was created by computing the ratio of the respective number of bedrooms ("*bedrooms*") to the number of bathrooms ("*bathrooms*") of the houses within the dataset.

No.	Source Code:	Output:
1	713 /* Summary Statistics */ 714 proc means data=house_features mean median min max std; 715 var bed_bath_ratio; 716 run;	The MEANS Procedure Analysis Variable : bed_bath_ratio Mean: 1.7491793 | Median: 1.6000000 | Minimum: 0.5000000 | Maximum: 6.0000000 | Std Dev: 0.6502668

The Code Block above provides the summary statistics for the *bed_bath_ratio* variable. The summarizing properties are as follows.

Summarizing Properties:

- There is no missing value.
- The mean value suggests that the bedroom-to-bathroom ratio of most houses within the dataset is 1.75, on average.
- In the context of a mean value of 1.75, a standard deviation of 0.6502 suggests that there is a moderate spread within the distribution of the *bed_bath_ratio* attribute.
- No inconsistencies were identified.

2	718 /* Distribution */ 719 proc sgplot data=house_features; 720 histogram bed_bath_ratio; 721 title "Distribution of Bedroom-Bathroom Ratio"; 722 run;	

Code Block Number 2 was executed to display the distribution of the attribute. The distribution appeared to be considerably skewed to the right.

3

```
*Compute quartiles for the 'bed_bath_ratio' variable;
PROC UNIVARIATE DATA=house_features;
    VAR bed_bath_ratio;
    OUTPUT OUT=OutliersBedBathRatio (RENAME=(bed_bath_ratio=OriginalBedBathRatio))
    Q1=Q1_bed_bath_ratio Q3=Q3_bed_bath_ratio;
RUN;

*Detect and store outliers for 'bed_bath_ratio' in the OutliersListBedBathRatio dataset;
DATA OutliersListBedBathRatio (keep=ObsNum OutlierValue);
    SET house_features;
    IF _N_ = 1 THEN SET OutliersBedBathRatio;

    IQR = Q3_bed_bath_ratio - Q1_bed_bath_ratio;
    LowerBound = Q1_bed_bath_ratio - 1.5 * IQR;
    UpperBound = Q3_bed_bath_ratio + 1.5 * IQR;

    /* Check if the bed_bath_ratio value is an outlier */
    IF bed_bath_ratio < LowerBound OR bed_bath_ratio > UpperBound THEN DO;
        ObsNum = _N_;
        OutlierValue = bed_bath_ratio;
        OUTPUT;
    END;

    DROP IQR LowerBound UpperBound Q1_bed_bath_ratio Q3_bed_bath_ratio;
RUN;

*Print detected outliers;
PROC PRINT DATA=OutliersListBedBathRatio;
RUN;

*Visualize 'bed_bath_ratio' distribution with a boxplot;
PROC SGPLOT DATA=house_features;
    VBOX bed_bath_ratio;
RUN;
```

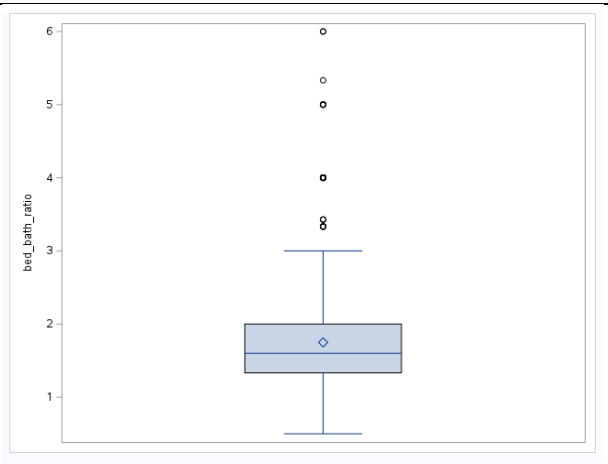
The UNIVARIATE Procedure
Variable: bed_bath_ratio

Moments			
N	3247	Sum Weights	3247
Mean	1.74917932	Sum Observations	5679.58526
Std Deviation	0.65026682	Variance	0.42284694
Skewness	1.50845329	Kurtosis	3.36871084
Uncorrected SS	11307.1743	Corrected SS	1372.56117
Coeff Variation	37.1755379	Std Error Mean	0.0114117

Basic Statistical Measures			
Location		Variability	
Mean	1.749179	Std Deviation	0.65027
Median	1.600000	Variance	0.42285
Mode	2.000000	Range	5.50000
		Interquartile Range	0.66667

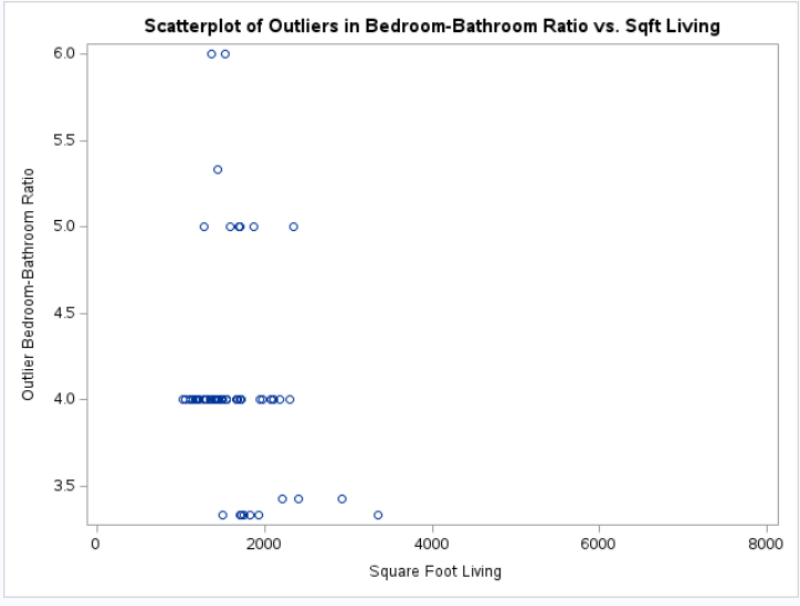
Quantiles (Definition 5)	
Level	Quantile
100% Max	6.00000
99%	4.00000
95%	3.00000
90%	3.00000
75% Q3	2.00000
50% Median	1.60000
25% Q1	1.33333
10%	1.14286
5%	1.00000
1%	0.80000
0% Min	0.50000

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0.500000	3007	5.00000	2188
0.571429	3142	5.00000	2793
0.571429	2791	5.33333	728
0.666667	3182	6.00000	1145
0.666667	2209	6.00000	3074

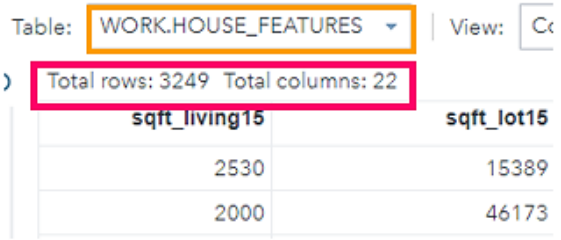


4

```sql
/* Add ObsNum to house_features */
data house_features;
    set house_features;
    ObsNum = _N_;
run;

/* Merge OutliersListBedBathRatio with house_features */
data OutliersWithSqft;
    merge house_features(rename=(bed_bath_ratio=OriginalBedBathRatio)) OutliersListBedBathRatio;
    by ObsNum;
run;

/* Scatterplot of OutlierValue vs. sqft_living */
proc sgplot data=OutliersWithSqft;
    scatter x=sqft_living y=OutlierValue;
    title "Scatterplot of Outliers in Bedroom-Bathroom Ratio vs. Sqft Living";
    xaxis label="Square Foot Living";
    yaxis label="Outlier Bedroom-Bathroom Ratio";
run;
```



```
1031 /* Drop the ObsNum column from house_features dataset */
1032 data house_features;
1033     set house_features(drop=ObsNum);
1034 run;
```



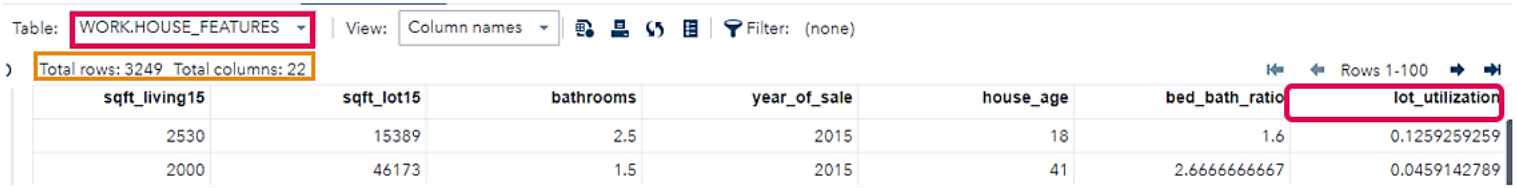
Code Block Number 3 relates to the performing of outlier detection on the newly created variable of *bed_bath_ratio*. According to the boxplot in the output column above, there seemed to be some outliers present within the attribute. To check if these outliers are genuine, Code Block Number 4 above proceeded to plot a scatterplot of the outliers contained within the *bed_bath_ratio* attribute against their respective *sqft_living* data. The rationale of doing so is to ensure that the total square footage of the interior living space is aligned with the bedroom-to-bathroom ratios of these outliers. That is, if they are aligned, then the outliers identified are then genuine. If it was otherwise, then these outliers must then be treated. Looking at the scatterplot plotted above, the outliers seemed to be clustered around the lower ranges of the total interior living space. Following this argument, the outliers have then proven themselves to be genuine, because it is only reasonable for smaller houses to have had more bedrooms share a single bathroom, as indicated by the higher bedroom-to-bathroom ratio. That said, the outliers detected does not require any treatments. Since the *ObsNum* column has served its purpose, which was to allow for the merging of the *house_features* dataset with the *OutliersListBedBathRatio* dataset, it was thereby dropped from the *house_features* dataset at the end of the feature creation process.

3.3.3.3 Feature Creation for Hypothesis 4

No.	Source Code:

```
788 /* lot_utilization VARIABLE */
789 data house_features;
790     set house_features;
791     /* Calculate the proportion of the lot that's occupied by living space */
792     /* This will be used to test if houses that utilize more of their lot for living space tend to have a higher price */
793     lot_utilization = sqft_living / sqft_lot;
794 run;
```

Output:



sqft_living15	sqft_lot15	bathrooms	year_of_sale	house_age	bed_bath_ratio	lot_utilization
2530	15389	2.5	2015	18	1.6	0.1259259259
2000	46173	1.5	2015	41	2.6666666667	0.0459142789

The Code Block above was executed to create a new variable called *lot_utilization* and to store the newly created variable into a dataset called "house_features". The *lot_utilization* attribute was created by computing the ratio of the respective square footage of the interior living space ("*sqft_living*") to the square footage of the land lots ("*sqft_lot*") of the houses within the dataset.

2

```
796 /* Summary Statistics */
797 proc means data=house_features mean median min max std;
798     var lot_utilization;
799 run;
800
```

The MEANS Procedure

Analysis Variable : lot_utilization

Mean	Median	Minimum	Maximum	Std Dev
0.3272402	0.2470294	0.0026031	2.1875000	0.2745702

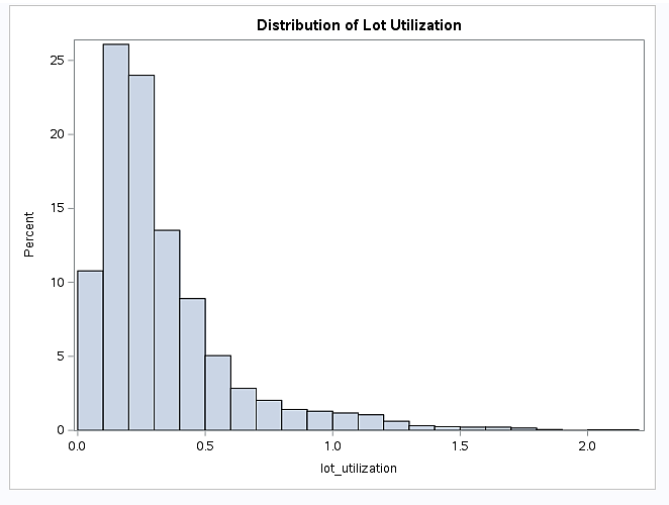
The Code Block above provides the summary statistics for the *lot_utilization* variable. The summarizing properties are as follows.

Summarizing Properties:

- There is no missing value.
- The mean value suggests that the ratio of total interior living space to the total land lot size of most houses within the dataset is 0.3272 square feet, on average.
- In the context of a mean value of 0.3272 square feet, a standard deviation of 0.2745 square feet suggests that there is a significant spread within the distribution of the *lot_utilization* attribute.
- No inconsistencies were identified.

3

```
801 /* Distribution */
802 proc sgplot data=house_features;
803     histogram lot_utilization;
804     title "Distribution of Lot Utilization";
805 run;
```



Code Block Number 3 was executed to display the distribution of the attribute. The distribution appeared to be considerably skewed to the right.

4

```
807 *Compute quartiles for the 'lot_utilization' variable;
808 PROC UNIVARIATE DATA=house_features;
809     VAR lot_utilization;
810     OUTPUT OUT=OutliersLotUtilization (RENAME=(lot_utilization=OriginalLotUtilization))
811     Q1=Q1_lot_utilization Q3=Q3_lot_utilization;
812 RUN;
```

The UNIVARIATE Procedure
Variable: lot_utilization

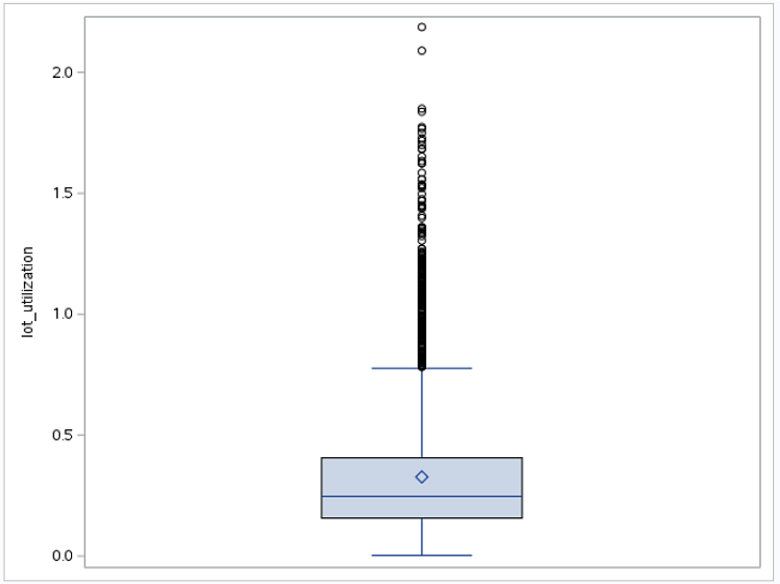
Moments			
N	3247	Sum Weights	3247
Mean	0.32724021	Sum Observations	1062.54895
Std Deviation	0.2745702	Variance	0.07538879
Skewness	2.18426963	Kurtosis	6.20397588
Uncorrected SS	592.420757	Corrected SS	244.712021
Coeff Variation	83.9047867	Std Error Mean	0.0048185

Basic Statistical Measures			
Location		Variability	
Mean	0.327240	Std Deviation	0.27457
Median	0.247029	Variance	0.07539
Mode	0.166667	Range	2.18490
		Interquartile Range	0.24931

```
814 *Detect and store outliers for 'lot_utilization' in the OutliersListLotUtilization dataset;
815 DATA OutliersListLotUtilization (keep=ObsNum OutlierValue);
816     SET house_features;
817     IF _N_ = 1 THEN SET OutliersLotUtilization;
818 
819     IQR = Q3_lot_utilization - Q1_lot_utilization;
820     LowerBound = Q1_lot_utilization - 1.5 * IQR;
821     UpperBound = Q3_lot_utilization + 1.5 * IQR;
822 
823     /* Check if the lot_utilization value is an outlier */
824     IF lot_utilization < LowerBound OR lot_utilization > UpperBound THEN DO;
825         ObsNum = _N_;
826         OutlierValue = lot_utilization;
827         OUTPUT;
828     END;
829 
830     DROP IQR LowerBound UpperBound Q1_lot_utilization Q3_lot_utilization;
831 RUN;
832 
833 *Print detected outliers;
834 PROC PRINT DATA=OutliersListLotUtilization;
835 RUN;
836 
837 *Visualize 'lot_utilization' distribution with a boxplot;
838 PROC SGPLOT DATA=house_features;
839     VBOX lot_utilization;
840 RUN;
```

Quantiles (Definition 5)	
Level	Quantile
100% Max	2.1875000
99%	1.3604767
95%	0.9244444
90%	0.6500000
75% Q3	0.4062437
50% Median	0.2470294
25% Q1	0.1569290
10%	0.0944901
5%	0.0517084
1%	0.0138470
0% Min	0.0026031

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
0.00260310	3055	1.77438	3027
0.00314270	2570	1.83761	269
0.00339403	369	1.85000	2098
0.00355831	2863	2.08976	605
0.00366333	1352	2.18750	2405



No.	Source Code:
5	

```
841
842 /* Filter observations where sqft_living is greater than sqft_lot and compute the difference */
843 data LivingGreaterLot;
844     set house_features;
845     where sqft_living > sqft_lot;
846
847     /* Compute the difference between sqft_living and sqft_lot */
848     living_lot_diff = sqft_living - sqft_lot;
849 run;
```

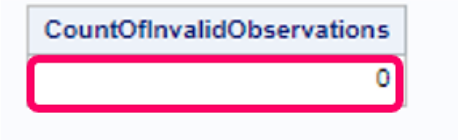
Output:

Table: WORK.LIVINGGREATERLOT | View: Column names | Filter: (none)

Total rows: 133 Total columns: 22 | Rows 1-100

15	sqft_lot15	bathrooms	year_of_sale	house_age	bed_bath_ratio	lot_utilization	living_lot_diff
20	1068	2.25	2014	2	1.3333333333	1.6248746239	623

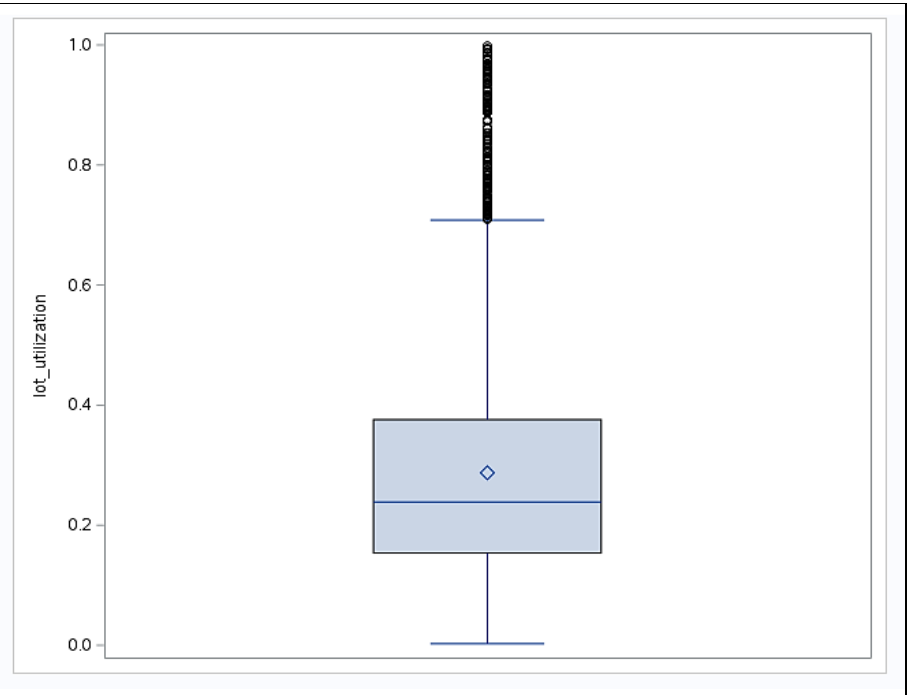
Code Block Number 4 relates to the performing of outlier detection on the newly created variable of *lot_utilization*. According to the boxplot in the output column above, there seemed to be a considerable number of outliers present within the attribute. Some of these outliers clearly did not make sense, because a *lot_utilization* value of above "1" simply means that the interior living space of the house has actually exceeded the size of the land lot, which is a scenario that is impractical. Following such observation, Code Block Number 5 above then proceeded to identify all observations where their sqft_living was greater than their sqft_lot, and a new column called "*living_lot_diff*" was added to the dataset, to store the results of the differences computed.

No.	Source Code:	Output:
6	851 /* Display filtered observations with the new difference column */ 852 proc print data=LivingGreaterLot; 853 run; 854 855 /* Filter out observations where sqft_living is greater than sqft_lot */ 856 data house_filtered; 857 set house_features; 858 if sqft_living <= sqft_lot; 859 run; 860 861 /* Confirm that the observations have been removed */ 862 proc sql; 863 select count(*) as CountOfInvalidObservations 864 from house_filtered 865 where sqft_living > sqft_lot; 866 quit;	

Code Block Number 6 then proceeded to display the problematic observations and subsequently to remove those observations from the dataset. Removal was successful because no more of such observations are now present in the dataset.

7

```
1116 *Visualize 'lot_utilization' distribution with a boxplot once again;
1117 PROC SGPLOT DATA=house_filtered;
1118     VBOX lot_utilization;
1119 RUN;
```



Once the removal was successful, the Code Block above was then executed to check if the rest of the outliers which remained were genuine or not. From the boxplot, all outliers identified were now within the reasonable range of below “1”. With that, none of these remaining outliers will be removed.

3.3.4 Transformation

In this section, transformation will be done for all continuous attributes whose distributions are skewed. These attributes are namely, *sqft_lot*, *sqft_lot15*, *price*, *sqft_basement*, *bed_bath_ratio*, *sqft_above*, *sqft_living*, *lot_utilization*, and *sqft_living15*. Since the transformation methods are repetitive for all variables except for that of the *sqft_basement* attribute, hence only the first instance for the *sqft_lot* attribute will be explained in depth, and subsequently applied to the rest of the attributes. The *sqft_basement* variable will require a separate explanation given that its data distribution is somewhat dissimilar with that of the rest of the attributes. Note also that the logarithmic transformation method was applied to transform all the attributes listed down below.

No.	Source Code:
1	885 /* Start by creating a copy of the original dataset to apply transformations */ 886 data house_transformed; 887 set house_filtered; 888 run;

Output:

Table: WORK.HOUSE_TRANSFORMED | View: Column na

Total rows: 3114 Total columns: 21

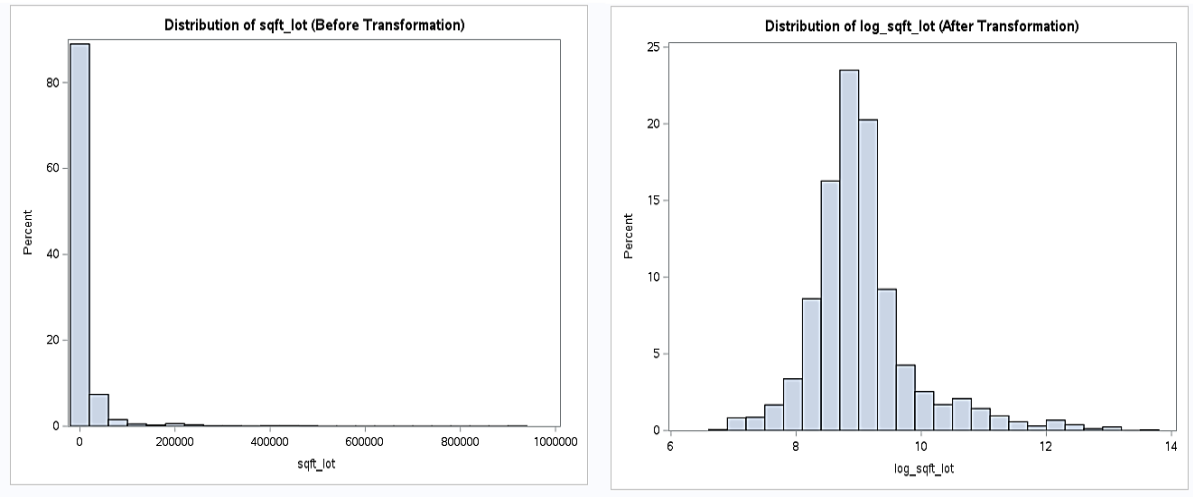
	price	bedrooms
1	475000	4
2	316000	4
3	802000	4
4	905000	4
5	700000	4

The source code above was executed to initiate a copy of the original dataset and have it named as “house_transformed”.

No.	Source Code:

```
/* ------------------------ */
/* sqft_lot */
/* ------------------------ */

/* Before Transformation */
proc sgplot data=house_filtered;
    histogram sqft_lot;
    title "Distribution of sqft_lot (Before Transformation)";
run;


/* Transformation */
data house_transformed;
    set house_transformed;
    log_sqft_lot = log(sqft_lot + 1);
run;


/* After Transformation */
proc sgplot data=house_transformed;
    histogram log_sqft_lot;
    title "Distribution of log_sqft_lot (After Transformation)";
run;
```

Output:

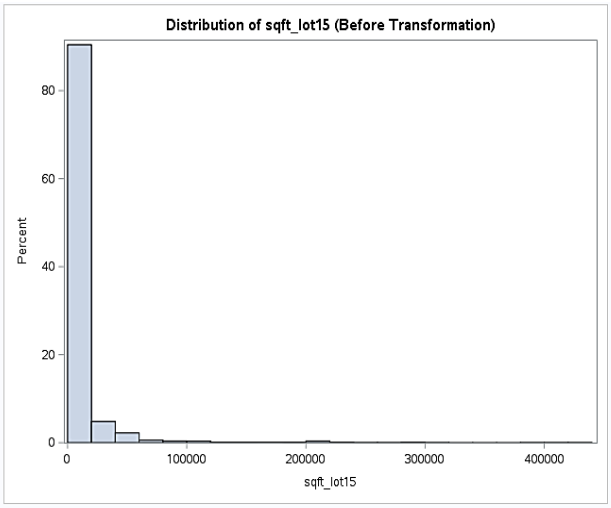


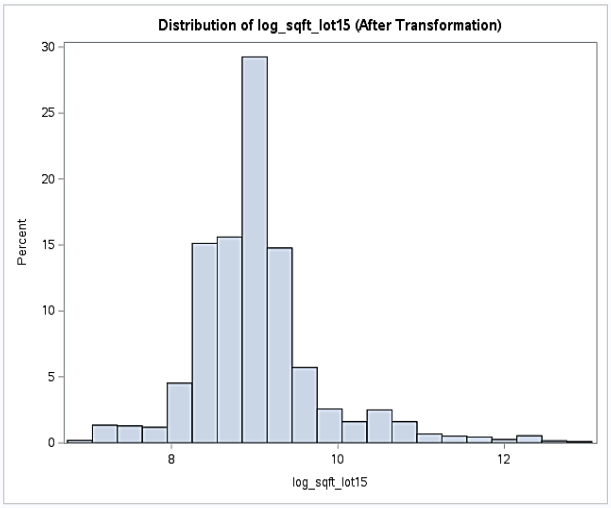
Code Block Number 2 above first showed the initial distribution of the *sqft_lot* attribute, as could be seen on the left side of the panel. The attribute is then log-transformed and the distribution of the attribute after transformation is then plotted and displayed on the right side of the panel. Notice that after transformation, the distribution has now approximated a normal distribution.

No.	Source Code:

```
916 /* ------------------------ */
917 /* sqft_lot15 */
918 /* ------------------------ */
919
920
921 /* Before Transformation */
922 proc sgplot data=house_filtered;
923     histogram sqft_lot15;
924     title "Distribution of sqft_lot15 (Before Transformation)";
925 run;
926
927
928 /* Transformation */
929 data house_transformed;
930     set house_transformed;
931     log_sqft_lot15 = log(sqft_lot15 + 1);
932 run;
933
934
935 /* After Transformation */
936 proc sgplot data=house_transformed;
937     histogram log_sqft_lot15;
938     title "Distribution of log_sqft_lot15 (After Transformation)";
939 run;
```

Output:



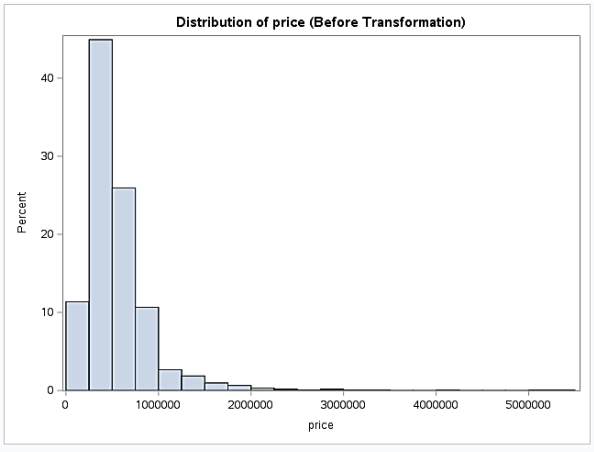


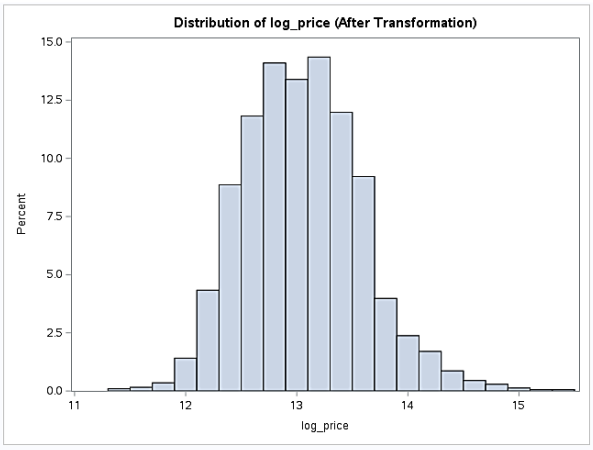
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
/* ------------------------ */
/* price */
/* ------------------------ */


/* Before Transformation */
proc sgplot data=house_filtered;
    histogram price;
    title "Distribution of price (Before Transformation)";
run;


/* Transformation */
data house_transformed;
    set house_transformed;
    log_price = log(price);
run;


/* After Transformation */
proc sgplot data=house_transformed;
    histogram log_price;
    title "Distribution of log_price (After Transformation)";
run;
```

Output:



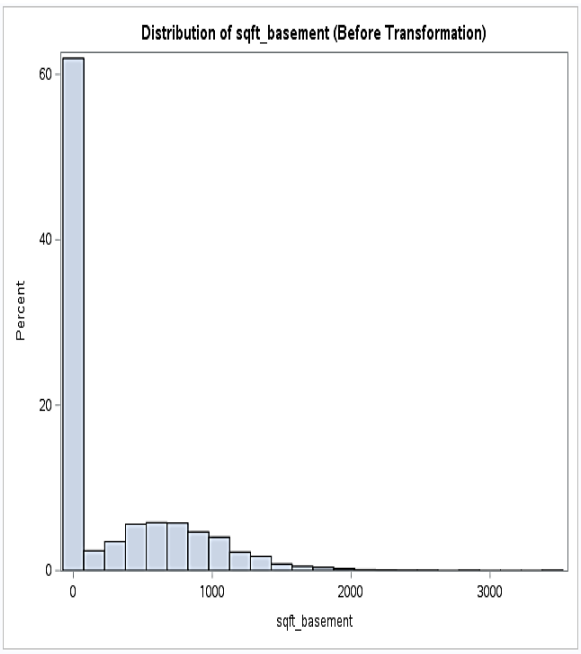


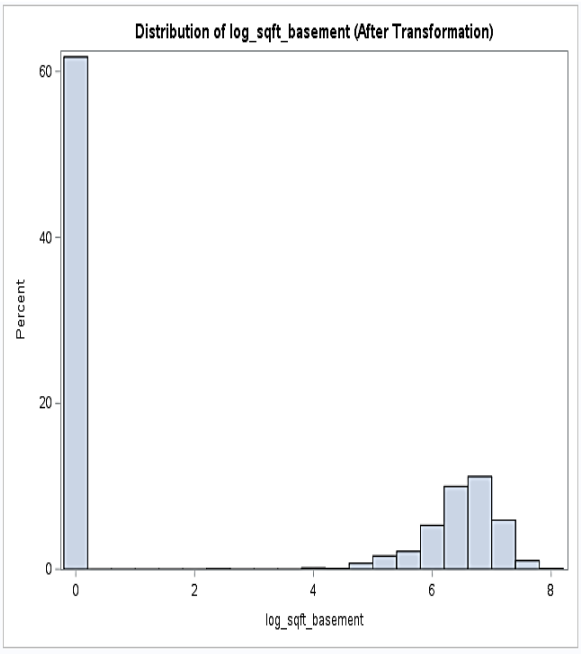
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
/* ------------------------ */
/* sqft_basement */
/* ------------------------ */


/* Before Transformation */
proc sgplot data=house_filtered;
    histogram sqft_basement;
    title "Distribution of sqft_basement (Before Transformation)";
run;


/* Transformation */
data house_transformed;
    set house_transformed;
    log_sqft_basement = log(sqft_basement + 1);
run;


/* After Transformation */
proc sgplot data=house_transformed;
    histogram log_sqft_basement;
    title "Distribution of log_sqft_basement (After Transformation)";
run;
```

Output:



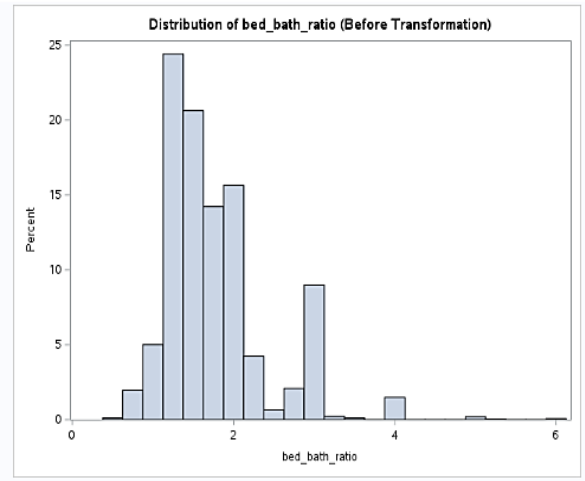


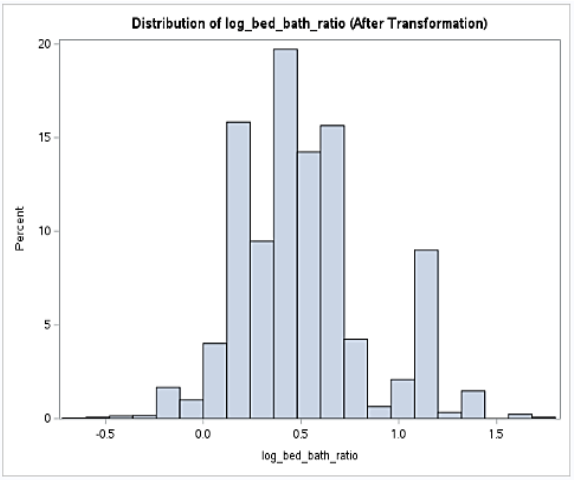
The Code Block above was executed to transform the *sqft_basement* attribute as a means to normalize its distribution. The code began by plotting the initial distribution of the variable. The plot could be seen on the left side of the panel above. After that, logarithmic transformation was applied to the attribute, and its post-transformation distribution was displayed (right side of the panel). Notice however that the distribution still looks unusual even after being log-transformed. As seen in the transformed histogram, the zero values dominated the distribution, essentially causing the distribution to be strange-looking. Having said that, and recognizing that the large peak at zero is legitimate and that it is representing houses without basements, it is then important to keep such distribution visible. In addition to that, it is important to recognize that while a variable may not have a perfect distribution, it can still however be predictive. The key is to understand the chosen model's assumptions (which is not known at this point of time) and to evaluate the variable's impact on the model's performance and interpretability. Given the above rationales, it was then decided to have the *sqft_basement* attribute kept in its log-transformed state.

No.	Source Code:

```
/* ------------------------ */
/* bed_bath_ratio */
/* ------------------------ */


/* Before Transformation */
proc sgplot data=house_filtered;
    histogram bed_bath_ratio;
    title "Distribution of bed_bath_ratio (Before Transformation)";
run;


/* Transformation */
data house_transformed;
    set house_transformed;
    log_bed_bath_ratio = log(bed_bath_ratio);
run;


/* After Transformation */
proc sgplot data=house_transformed;
    histogram log_bed_bath_ratio;
    title "Distribution of log_bed_bath_ratio (After Transformation)";
run;
```

Output:



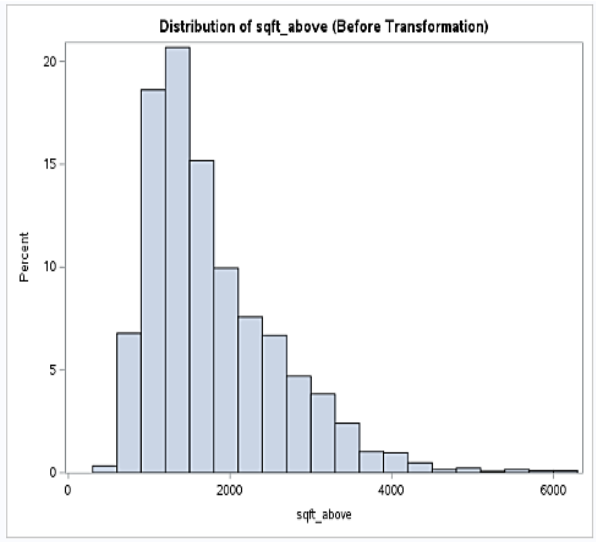


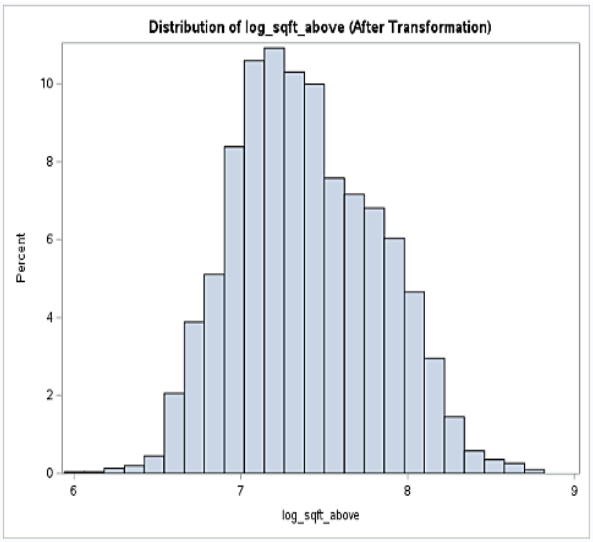
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
1058 /* ------------------------ */
1059 /* sqft_above */
1060 /* ------------------------ */
1061
1062
1063 /* Before Transformation */
1064 proc sgplot data=house_filtered;
1065     histogram sqft_above;
1066     title "Distribution of sqft_above (Before Transformation)";
1067 run;
1068
1069
1070 /* Transformation */
1071 data house_transformed;
1072     set house_transformed;
1073     log_sqft_above = log(sqft_above);
1074 run;
1075
1076
1077 /* After Transformation */
1078 proc sgplot data=house_transformed;
1079     histogram log_sqft_above;
1080     title "Distribution of log_sqft_above (After Transformation)";
1081 run;
```

Output:



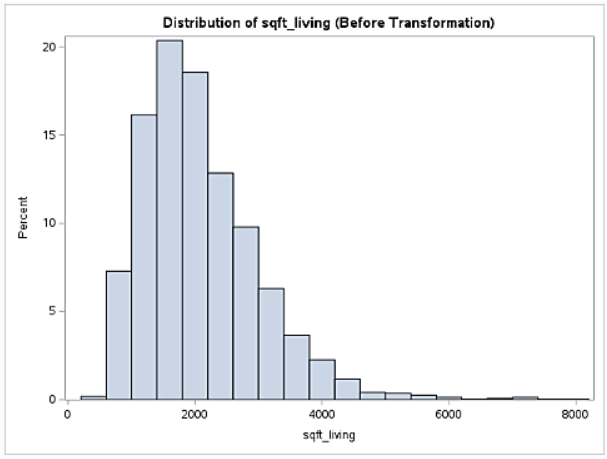


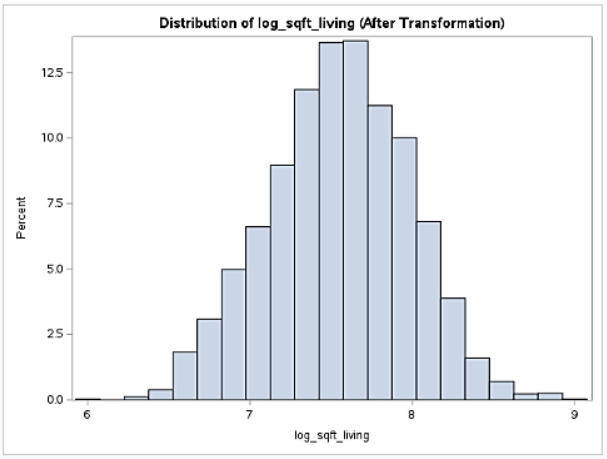
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
/* ------------------------ */
/* sqft_living */
/* ------------------------ */


/* Before Transformation */
proc sgplot data=house_filtered;
    histogram sqft_living;
    title "Distribution of sqft_living (Before Transformation)";
run;

/* Transformation */
data house_transformed;
    set house_transformed;
    log_sqft_living = log(sqft_living);
run;


/* After Transformation */
proc sgplot data=house_transformed;
    histogram log_sqft_living;
    title "Distribution of log_sqft_living (After Transformation)";
run;
```

Output:



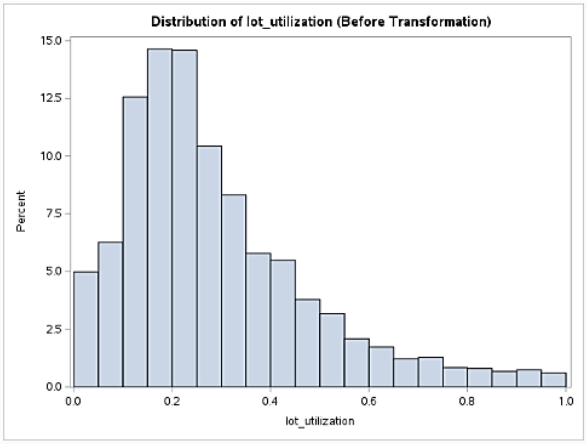


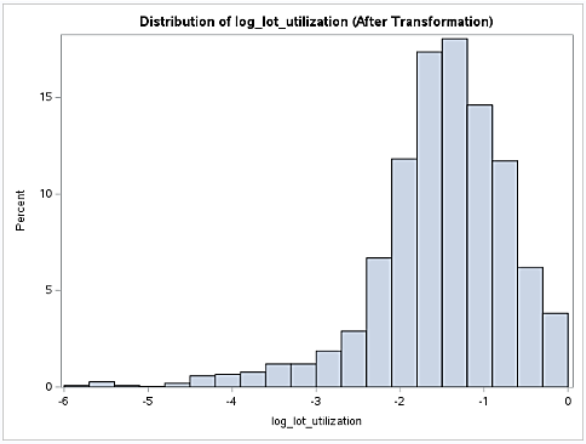
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
1109 /* ------------------------ */
1110 /* lot_utilization */
1111 /* ------------------------ */
1112
1113
1114 /* Before Transformation */
1115 proc sgplot data=house_filtered;
1116     histogram lot_utilization;
1117     title "Distribution of lot_utilization (Before Transformation)";
1118 run;
1119
1120
1121 /* Transformation */
1122 data house_transformed;
1123     set house_transformed;
1124     log_lot_utilization = log(lot_utilization);
1125 run;
1126
1127
1128 /* After Transformation */
1129 proc sgplot data=house_transformed;
1130     histogram log_lot_utilization;
1131     title "Distribution of log_lot_utilization (After Transformation)";
1132 run;
1133
```

Output:



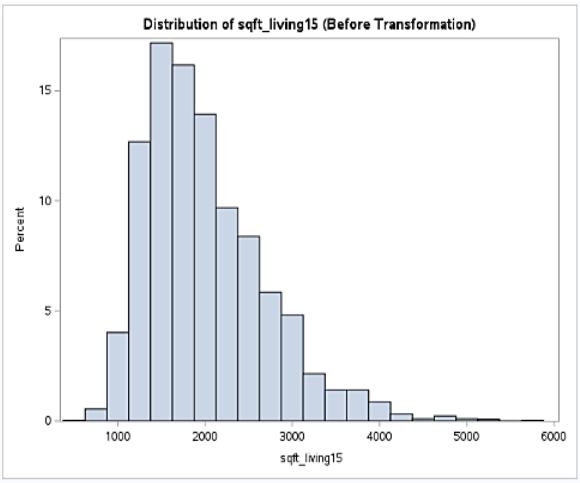


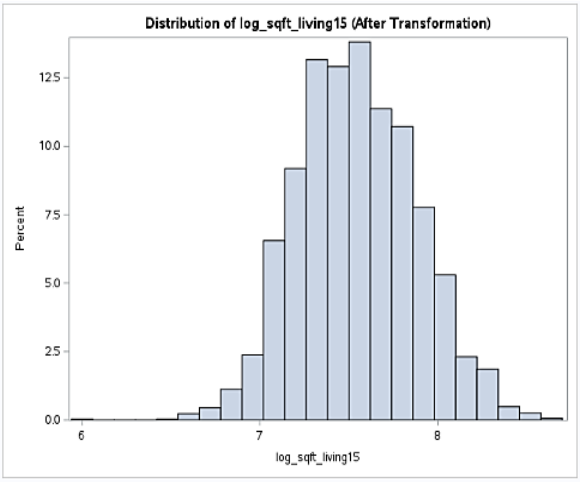
Similar explanation done for the *sqft_lot* attribute applies here too.

No.	Source Code:

```
1135 /* ------------------------- */
1136 /* sqft_living15 */
1137 /* ------------------------- */
1138
1139
1140 /* Before Transformation */
1141 proc sgplot data=house_filtered;
1142     histogram sqft_living15;
1143     title "Distribution of sqft_living15 (Before Transformation)";
1144 run;
1145
1146
1147 /* Transformation */
1148 data house_transformed;
1149     set house_transformed;
1150     log_sqft_living15 = log(sqft_living15);
1151 run;
1152
1153
1154 /* After Transformation */
1155 proc sgplot data=house_transformed;
1156     histogram log_sqft_living15;
1157     title "Distribution of log_sqft_living15 (After Transformation)";
1158 run;
```

Output:





Similar explanation done for the *sqft_lot* attribute applies here too.

3.3.5 Scaling

In this section, scaling will be done for the continuous attributes namely, *lat*, *long*, *log_bed_bath_ratio*, *house_age*, *log_sqft_lot*, *log_sqft_lot15*, *log_sqft_above*, *log_sqft_basement*, *log_sqft_living*, *log_sqft_living15*, and *log_lot_ultization*.

No.	Source Code:

```
/* ------------------------- */
/* Scaling */
/* ------------------------- */

/* Start by creating a copy of the original dataset to apply scaling */
data house_scaled;
    set house_transformed;
run;

/* Use PROC STANDARD to scale the variables to have a mean of 0 and standard deviation of 1 */
proc standard data=house_scaled out=house_scaled mean=0 std=1;
    var lat
        long
        log_bed_bath_ratio
        house_age
        log_sqft_lot
        log_sqft_lot15
        log_sqft_above
        log_sqft_basement
        log_sqft_living
        log_sqft_living15
        log_lot_utilization;
run;

The PROC STANDARD procedure is used to scale (standardize) the variables.
This ensures that the selected variables have a mean of 0 and a standard deviation of 1,
making them comparable in terms of scale.
The output dataset "house_scaled" now contains the scaled versions of the specified variables.
*/

/* Check mean and standard deviation of the scaled variables */
proc means data=house_scaled mean std;
    var lat long log_bed_bath_ratio house_age
        log_sqft_lot log_sqft_lot15 log_sqft_above log_sqft_basement
        log_sqft_living log_sqft_living15 log_lot_utilization;
run;
```

Output:

The MEANS Procedure

Variable	Mean	Std Dev
lat	3.541904E-14	1.0000000
long	1.073588E-12	1.0000000
log_bed_bath_ratio	1.000534E-14	1.0000000
house_age	-1.08874E-17	1.0000000
log_sqft_lot	2.897873E-15	1.0000000
log_sqft_lot15	1.126695E-15	1.0000000
log_sqft_above	4.581656E-14	1.0000000
log_sqft_basement	2.495257E-15	1.0000000
log_sqft_living	-2.32841E-15	1.0000000
log_sqft_living15	2.376418E-14	1.0000000
log_lot_utilization	3.929754E-15	1.0000000

The Code Block above began by initiating a copy of the original dataset and subsequently naming the dataset as “house_scaled”. Following that, all of the attributes listed above were then scaled to have a mean of “0” and a standard deviation of “1”. After that, the standard deviations and mean of the scaled variables were then printed out to ensure that scaling was successful. Scaling was indeed successful.

3.3.6 One-hot encoding

One-hot encoding is usually the standard way to handle categorical data. While mention about one-hot encoding could be too soon at this point of reading this report, the specific execution of such technique could be found in the testing of the last Hypothesis detailed in *Section 4.5 Hypothesis 5*. The exact source code and output could also be found in the said section.

4.0 Hypothesis

In this section, testing will be performed on five distinct hypothesis statements.

These five statements are as follows:

1. Older houses tend to have lower prices as compared to newer ones, all else being equal.
2. Houses where the total average living space of the 15 nearest neighbors is high tend to be in pricier neighborhoods, all else being equal.
3. Houses with a higher bedroom-to-bathroom ratio, meaning more bedrooms per bathroom, tend to have a lower price, all else being equal.
4. Houses that utilize a higher proportion of their lot for interior living space tend to have a higher price, all else being equal.
5. Houses with a higher view rating are significantly pricier than those with a lower view rating, all else being equal.

Also, it is important to note that, since the tests used for the hypothesis testing of the five statements mentioned above are linear regression and ANOVA test, having had scaled variables is thereby not necessary. For that reason, the dataset used for the following hypothesis testing will be the “house_transformed” dataset instead of the “house_scaled” dataset. The former dataset has no scaling applied to it while the latter dataset has scaling performed on it. The choice to defer scaling at this stage is because, scaling would change the magnitude of the coefficients of each attribute, essentially making interpretation harder for non-technical audience. Keeping coefficients in their original units rather than the standardized units would then make presenting findings to these group of audience more comprehensible.

4.1 Hypothesis 1

No.	Source Code:

```
1222 /* ------------------------ */
1223 /* Hypothesis 1: House Age and Price */
1224 /* ------------------------ */
1225
1226 /* Linear Regression for House Age and Price */
1227 /* Here is the modeling of the relationship between the price and the year the house was built */
1228 /* The yr_built will demonstatre if older homes are priced differently than newer ones */
1229
1230 proc reg data=house_transformed;
1231     model log_price = house_age; /* price is the dependent variable and yr_built is the independent variable */
1232     title "Linear Regression: Log-Transformed Price vs. House Age";
1233 run;
```

The source code above was executed to perform a linear regression analysis on the relationship between the *log_price* and *house_age* attribute.

Output:

Linear Regression: Log-Transformed Price vs. House Age

The REG Procedure
Model: MODEL1
Dependent Variable: log_price

Number of Observations Read	3114
Number of Observations Used	3114

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	8.93042	8.93042	31.67	<.0001
Error	3112	877.52496	0.28198		
Corrected Total	3113	886.45538			

Root MSE	0.53102	R-Square	0.0101
Dependent Mean	13.05262	Adj R-Sq	0.0098
Coeff Var	4.06829		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	13.13534	0.01751	750.20	<.0001
house_age	1	-0.00183	0.00032592	-5.63	<.0001

Hypothesis Statement:

Older houses tend to have lower prices as compared to newer ones, all else being equal.

Linear Regression Equation:

$$\widehat{\log\_price} = 13.13534 - 0.00183 house\_age$$

Null and Alternative Hypotheses:

$H_0: \beta_{house\_age} = 0$ (The age of the house has no effect on its price.)

$H_1: \beta_{house\_age} < 0$ (Older houses tend to have lower prices.)

The decision rule is to reject the null hypothesis if p-value is less than the significance level. In this case, since the p-value is smaller than 0.001, which is less than the significance level of 0.05, we thereby reject the null hypothesis. There is sufficient evidence to conclude that there is a significant negative relationship between the age of a house and its price. This relationship is statistically significant.

4.2 Hypothesis 2

No.	Source Code:
1	1235 /* ------------------------ */ 1236 /* Hypothesis 2: Larger Neighborhood Living Spaces Indicate Pricier Areas */ 1237 /* ------------------------ */ 1238 1239 /* Linear Regression for Neighborhood Living Space and Price */ 1240 /* Here is the modeling of the relationship between the price and the average living space of the 15 nearest neighbors (sqft_living15) */ 1241 /* This will help determine if houses in neighborhoods with larger average living spaces are pricier */ 1242 1243 proc reg data=house_transformed; 1244 model log_price = log_sqft_living15; /* price is the dependent variable and sqft_living15 is the independent variable */ 1245 title "Linear Regression: Log-Transformed Price vs. Average Living Space of 15 Nearest Neighbors"; 1246 run;

The source code above was executed to perform a linear regression analysis on the relationship between the *log_price* and *log_sqft_living15* attribute.

Output:

Linear Regression: Log-Transformed Price vs. Average Living Space of 15 Nearest Neighbors

The REG Procedure
Model: MODEL1
Dependent Variable: log_price

Number of Observations Read	3114
Number of Observations Used	3114

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	340.50539	340.50539	1940.93	<.0001
Error	3112	545.94999	0.17543		
Corrected Total	3113	886.45538			

Root MSE	0.41885	R-Square	0.3841
Dependent Mean	13.05262	Adj R-Sq	0.3839
Coeff Var	3.20892		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	5.51715	0.17121	32.22	<.0001
log_sqft_living15	1	0.99849	0.02266	44.06	<.0001

Hypothesis Statement:

Houses where the total average living space of the 15 nearest neighbors is high tend to be in pricier neighborhoods, all else being equal.

Linear Regression Equation:

$$\widehat{\log\_price} = 5.51715 + 0.99849 \log\_sqft\_living15$$

Null and Alternative Hypotheses:

$H_0: \beta_{\log\_sqft\_living15} = 0$ (The average living space of the 15 nearest neighbors has no effect on the price of a house.)

$H_1: \beta_{\log\_sqft\_living15} > 0$ (The average living space of the 15 nearest neighbors has a positive effect on the price of a house.)

The decision rule is to reject the null hypothesis if p-value is less than the significance level. In this case, since the p-value is smaller than 0.001, which is less than the significance level of 0.05, we thereby reject the null hypothesis. There is sufficient evidence to conclude that houses where the total average living space of the 15 nearest neighbors is high tend to be in pricier neighborhoods.

4.3 Hypothesis 3

No.	Source Code:

```
1248 /* ------------------------ */
1249 /* Hypothesis 3: Bedroom-Bathroom Ratio */
1250 /* ------------------------ */
1251
1252 /* Linear Regression for Bedroom-Bathroom Ratio and Price */
1253 /* Here is the modeling of the relationship between the price and the ratio of bedrooms to bathrooms (bed_bath_ratio) */
1254 /* This will test if homes with more bedrooms per bathroom are priced lower */
1255
1256 proc reg data=house_transformed;
1257     model log_price = log_bed_bath_ratio; /* price is the dependent variable and log_bed_bath_ratio is the independent variable */
1258     title "Linear Regression: Log- Transformed Price vs. Log-Transformed Bedroom-Bathroom Ratio";
1259 run;
```

The source code above was executed to perform a linear regression analysis on the relationship between the *log_price* and *log_bed_bath_ratio* attribute.

Output:

Linear Regression: Log- Transformed Price vs. Log-Transformed Bedroom-Bathroom Ratio

The REG Procedure
Model: MODEL1
Dependent Variable: log_price

Number of Observations Read	3114
Number of Observations Used	3114

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	92.83251	92.83251	364.02	<.0001
Error	3112	793.62286	0.25502		
Corrected Total	3113	886.45538			

Root MSE	0.50500	R-Square	0.1047
Dependent Mean	13.05262	Adj R-Sq	0.1044
Coeff Var	3.86892		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	13.31722	0.01656	804.19	<.0001
log_bed_bath_ratio	1	-0.51547	0.02702	-19.08	<.0001

Hypothesis Statement:

Houses with a higher bedroom-to-bathroom ratio, meaning more bedrooms per bathroom, tend to have a lower price, all else being equal.

Linear Regression Equation:

$$\widehat{\log\_price} = 13.31722 - 0.51547 \log\_bed\_bath\_ratio$$

Null and Alternative Hypotheses:

$H_0: \beta_{\log\_bed\_bath\_ratio} = 0$ (There is no relationship between the bedroom-to-bathroom ratio and the price of a house.)

$H_1: \beta_{\log\_bed\_bath\_ratio} < 0$ (Houses with a higher bedroom-to-bathroom ratio tend to have a lower price.)

The decision rule is to reject the null hypothesis if p-value is less than the significance level. In this case, since the p-value is smaller than 0.001, which is less than the significance level of 0.05, we thereby reject the null hypothesis. There is sufficient evidence to conclude that houses with a higher bedroom-to-bathroom ratio tend to have a lower price, holding all else constant.

4.4 Hypothesis 4

No.	Source Code:
1	1261 /* ------------------------ */ 1262 /* Hypothesis 4: Lot Utilization*/ 1263 /* ------------------------ */ 1264 1265 /* Linear Regression for Lot Utilization and Price */ 1266 /* Here is the modeling of the relationship between the price and lot utilization (ratio of living space to lot size) */ 1267 /* This tests if houses that utilize a higher proportion of their lot for living space tend to be pricier */ 1268 1269 proc reg data=house_transformed; 1270 model log_price = log_lot_utilization; /* price is the dependent variable and log_lot_utilization is the independent variable */ 1271 title "Linear Regression: Log-Transformed Price vs. Log Transformed Lot Utilization"; 1272 run;

The source code above was executed to perform a linear regression analysis on the relationship between the *log_price* and *log_lot_utilization* attribute.

Output:

Linear Regression: Log-Transformed Price vs. Log Transformed Lot Utilization

The REG Procedure
Model: MODEL1
Dependent Variable: log_price

Number of Observations Read	3114
Number of Observations Used	3114

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	30.25171	30.25171	109.95	<.0001
Error	3112	856.20367	0.27513		
Corrected Total	3113	886.45538			

Root MSE	0.52453	R-Square	0.0341
Dependent Mean	13.05262	Adj R-Sq	0.0338
Coeff Var	4.01856		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	13.23573	0.01983	667.42	<.0001
log_lot_utilization	1	0.12185	0.01162	10.49	<.0001

Hypothesis Statement:

Houses that utilize a higher proportion of their lot for interior living space tend to have a higher price, all else being equal.

Linear Regression Equation:

$$\widehat{\log\_price} = 13.23573 + 0.12185 \log\_lot\_utilization$$

Null and Alternative Hypotheses:

$H_0: \beta_{\log\_lot\_utilization} = 0$ (There is no relationship between the utilization and the price of the house.)

$H_1: \beta_{\log\_lot\_utilization} > 0$ (The lot utilization has a positive effect on the price of the house.)

The decision rule is to reject the null hypothesis if p-value is less than the significance level. In this case, since the p-value is smaller than 0.001, which is less than the significance level of 0.05, we thereby reject the null hypothesis. There is sufficient evidence to conclude that houses which utilize a higher proportion of their lot for interior living space tend to have a higher price when everything else is held constant.

4.5 Hypothesis 5

No.	Source Code:
1	

```
/* ------------------------ */
/* Hypothesis 5: Rare Views Command Premium Prices*/
/* ------------------------ */

/* Creating dummy variables for 'view' */
/* Since 'view' is a categorical variable with multiple categories, it should then be converted to multiple binary (0 or 1) dummy variables */
/* Each dummy variable represents one category of the 'view' variable */

data house_transformed_with_dummies;
    set house_transformed;
    if view = 0 then view_0 = 1; else view_0 = 0; /* Dummy for view rating 0 */
    if view = 1 then view_1 = 1; else view_1 = 0; /* Dummy for view rating 1 */
    if view = 2 then view_2 = 1; else view_2 = 0; /* Dummy for view rating 2 */
    if view = 3 then view_3 = 1; else view_3 = 0; /* Dummy for view rating 3 */
    if view = 4 then view_4 = 1; else view_4 = 0; /* Dummy for view rating 4 */
run;

/* ANOVA for Price and View Rating */
/* The ANOVA test is used to test if there's a significant difference in the house prices across different view ratings */
/* This will help determine if rare views indeed command premium prices */

proc glm data=house_transformed_with_dummies;
    class view; /* 'view' is treated as a categorical variable */
    model log_price = view; /* log-transformed price is the dependent variable and 'view' dummies are the independent variables */
    title "ANOVA: Log-Transformed Price vs. View Rating";
run;
```

The source code above was executed to perform an ANOVA test to compare if there are significant differences in the average log-transformed house prices across different view ratings. The Code Block above began by firstly creating dummy variables for each category within the *view* attribute. Each dummy variable created will represent one category of the *view* variable and these dummy variables were then stored into the “house_transformed_with_dummies” dataset. Following that, the ANOVA test is conducted using the General Linear Model, and the results from the test is displayed as could be seen in the output column below. On a side note, it is important to mention that the process of creating dummy variables for each category as mentioned above is actually one form of feature engineering better known as one-hot encoding. As one might have read about it already in previous section, mentions on the one-hot encoding technique could be found in *Section 3.3.6 One-hot encoding*.

Output:

ANOVA: Price vs. View Rating

The GLM Procedure

Class Level Information		
Class	Levels	Values
view	5	0 1 2 3 4

Number of Observations Read	3114
Number of Observations Used	3114

ANOVA: Price vs. View Rating

The GLM Procedure

Dependent Variable: log_price

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	116.8013221	29.2003305	117.95	<.0001
Error	3109	769.6540557	0.2475568		
Corrected Total	3113	886.4553778			

R-Square	Coeff Var	Root MSE	log_price Mean
0.131762	3.811883	0.497551	13.05262

Source	DF	Type I SS	Mean Square	F Value	Pr > F
view	4	116.8013221	29.2003305	117.95	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
view	4	116.8013221	29.2003305	117.95	<.0001

Hypothesis Statement:

Houses with a higher view rating are significantly pricier than those with a lower view rating, all else being equal.

Null and Alternative Hypotheses:

Let μ_0, μ_1, μ_2, μ_3, μ_4 represent the mean log-transformedμ prices for view ratings of 0,1,2,3,and 4 respectively.

H_0: μ0 = μ1 = μ2 = μ3 = μ4 (The means of the prices are the same across all view ratings)

H_1: At least one μ is different from the others. (At least one view rating's mean of price is different from the others.)

The decision rule is to reject the null hypothesis if p-value is less than the significance level. In this case, since the p-value is smaller than 0.001, which is less than the significance level of 0.05, we thereby reject the null hypothesis. There is sufficient evidence to conclude that there is a statistically significant difference in the mean log-transformed house prices across different view ratings. In other words, the view rating has a significant effect on the house price, essentially supporting the hypothesis that houses with a higher view rating are significantly pricier than those with a lower view rating, all else being equal.

5.0 Discussion

Given the testing carried out above, the results from all five of the testing could be concluded into two main points, the first one being that, attributes namely, the age of the house and the bedroom-to-bathroom ratio were found to be inversely related to the sale price of the house; secondly, attributes namely the neighborhood living space, lot utilization and view ratings of the house, on the other hand, were proven to have had a positive relationship with the sale price of the house itself. It is worth noting as well that all of these attributes listed were found to be statistically significant. The effects of these variables on the sale price were also in the expected direction as suggested by the literature work previously discussed in *Section 2.0*. Even when interpreting from a non-statistical point of view, the results from the hypothesis testing would still make perfect sense. This is because, and taking the example of the first hypothesis, houses that are older in age meant that they have gone through more wear and tear as compared to newer ones. When taking into account the need for refurbishment and the maintenance work required, homebuyers would naturally demand to be compensated in the form of a reduced sale price. The only exception to this would be, if the house was of a heritage status .

In terms of the second hypothesis which claimed that neighborhood with larger living space is indicative of a pricier area, the rationale is that, an area where most homes are larger might be a more upscale or desirable neighborhood. This is evident in the work done by Owusu-Ansah (2012), where in his work he segregated the residential area of his study into three different classes namely, first-, second-, and third-class residential area. It was shown that the average land price in the first-class residential areas took the value between GH¢300,000 to GH¢450,000 per acre and the properties within the area tended to be considerable large. Given the size, most properties were single detached properties and were dwelling places for the elites of the society. The average plot size in the second- and third-class residential area however, was approximated at 0.25 acres. Both classes consisted of a mix of property types ranging from semi-detached, flat, to what is known as traditional compound houses in Ghana. Once segregated, the author then proceeded to evaluate the impact of these three different residential areas on the value of their respective properties. Findings were that properties situated in the first-class residential area were approximately 234% and 149% pricier than those located in the third- and second-class residential areas, respectively. At the same time, properties situated in the second-class residential areas were about 85% pricier than their counterparts in the third-class residential areas. In short, while individual houses with larger plot size would command

higher prices, houses within a prestigious neighborhood, as indicated by the plot size of their neighbors would command higher prices too.

For the third hypothesis, houses with greater number of bedroom-to-bathroom ratio (a ratio of "3" means that 3 bedrooms are sharing 1 bathroom) will very likely appear non-appealing to home buyers. This is especially so if there are many occupants in the house, and if convenience and privacy were a concern to the home buyer. Applying an almost similar reasoning to the next hypothesis statement concerning lot utilization, Chin and Chau (2003) in their work explained that home buyers were willing to pay more in exchange for more usable space, or in other words, a greater built-up area. The authors attributed this identified preference as home buyers wanting to have that flexibility in terms of being able to switch from personal utility to functional utility whenever required. In a separate work done by Kam, Chuah, Lim & Ang (2016), the authors found that for every increment in m^2 of built-up area, Malaysian home buyers were willing to pay RM2392.76 more for a particular property. Lastly, concerning the positive relationship between the view that a property is overlooking and the property's price itself, even though not many research was done with regards to this, probably due to its highly subjective nature, one study done by Jayasekare et al. (2019) found that beach views have had the greatest influence on the value of properties in comparison to other views, such as sea, conservation area, parks, and water views which were included in the study. The authors demonstrated that every 1% increase in beach view within a 1-kilometer radius has resulted in a 2 to 3% elevation in property prices.

6.0 Conclusion

In this study, even though there were several discussions made on the multidimensionality of the residential estate valuation mechanism and the discussion on a few factors identified to have played an important role in predicting house prices, one should not be led to lose sight of the primary objective of this paper, which was to demonstrate the importance of appropriate data pre-processing and feature engineering techniques as part of the data analysis workflow. While the documentation of both processes has proved itself to be very lengthy, both processes could however be appropriately summarized into a single statement, which is that, the process of data pre-processing and feature engineering is more of an “art” than it is a “science”. The idea is that, both processes often involve many subjective decisions that require a good blend of domain expertise, intuition, experience, and creativity, rather than just strictly adhering to a fixed set and sequence of steps. Both processes are often iterative in nature, where an analyst might begin by pre-processing and feature engineering the data, followed by building a model, evaluate the model’s performance, and then go back to the pre-processing and feature engineering stages to make improvements if the model’s performance was suboptimal. This feedback loop continues until the analyst achieved results that are satisfactory. That said, while the process may be iterative and tedious, there is really no easier way out because data integrity must be upheld in any data analysis workflow, bearing in mind that with garbage in comes garbage out.

References

Adair, A., McGreal, S., Smyth, A., Cooper, J., & Ryley, T. (2000). House prices and accessibility: The testing of relationships within the Belfast Urban Area. *Housing Studies*, *15*(5), 699–716. https://doi.org/10.1080/02673030050134565

Babawale, G. K., & Adewunmi, Y. (2011). The impact of neighbourhood churches on house prices. *Journal of Sustainable Development*, *4*(1), 246–253. Retrieved from https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=3bc07102108772a3252ac84ff32747d85447e553

Bello, M. O., & Bello, V. A. (2007). The influence of consumers behavior on the variables determining residential property values in Lagos, Nigeria. *American Journal of Applied Sciences*, *4*(10), 774–778. https://doi.org/10.3844/ajassp.2007.774.778

Chiang, Y.-H., Peng, T.-C., & Chang, C.-O. (2015). The nonlinear effect of convenience stores on house prices: A case study of Taipei, Taiwan. *Habitat International*, *46*, 82–90. https://doi.org/10.1016/j.habitatint.2014.10.017

Chin, T. L., & Chau, K. W. (2003). A critical review of literature on the hedonic price model. *International Journal for Housing Science and its Applications*, *27*(2), 145 - 165. Retrieved from https://ssrn.com/abstract=2073594

Ge, X. J., & Du, Y. (2007). *Main variables influencing residential property values using the entropy method – The case of Auckland*. Retrieved from https://www.asres2007.um.edu.mo/papers/041%20-%20PAPER.pdf

Goodman, A. C., & Thibodeau, T. G. (1995). Age-related heteroskedasticity in hedonic house price. *Journal of Housing Research*, *6*(1), 25–42. Retrieved from https://allengoodman.wayne.edu/Research/PUBS/Deep/age_r_hetero.pdf

Iman, A. H. B. H., Hamidi, N., & Liew, S. (2009). The effects of environmental disamenities on house prices. *Malaysian Journal of Real Estate*, *4*(2), 31–44. Retrieved from https://api.semanticscholar.org/CorpusID:130061761

Jayasekare, A.S., Herath, S., Wickramasuriya, R., & Perez, P. (2019). The price of a view: Estimating the impact of view on house prices. *Pacific Rim Property Research Journal*, *25*(2), 141-158. doi: 10.1080/14445921.2019.1626543

Kain, J. F., & Quigley, J. M. (1970). Measuring the value of housing quality. *Journal of the American Statistical Association, 65*. 532 - 548. https://doi.org/10.1080/01621459.1970.10481102

Kam, K. J., Chuah, S. Y., Lim, T. S., & Ang, F. L. (2016). Modelling of property market: The structural and locational attributes towards Malaysian properties. *Pacific Rim Property Research Journal*, *22*(3), 203-216. doi:10.1080/14445921.2016.1234361

Kauko, T. O. M., Hooimeijer, P., & Hakfoort, J. (2002). Capturing housing market segmentation: An alternative approach based on neural network modelling. *Housing Studies, 17*(6), 875–894. https://doi.org/10.1080/02673030215999

Khoiry, M. A., Tawil, N. M., Hamzah, N., Ani, A. I. C., & Sood, S. (2012). Critical factors affecting double storey terrace houses prices in Bandar Baru Bangi. *Procedia – Social and Behavioral Sciences, 60*, 562–566. https://doi.org/10.1016/j.sbspro.2012.09.423

Oloke, O. C., Simon, F. R., & Adesulu, A. F. (2013). An examination of the factors affecting residential property values in Magodo neighbourhood, Lagos State. *International Journal of Economy, Management and Social Sciences, 2*(8), 639–643.

Ong, T. S. (2013). Factors affecting the price of housing in Malaysia. *Journal of Emerging Issues in Economics, Finance and Banking, 1*(5), 414–429.

Owusu-Ansah, A. (2012). Examination of the determinants of housing values in urban Ghana and implications for policy makers. *Journal of African Real Estate Research, 2*(1), 58 - 85. Retrieved from https://api.semanticscholar.org/CorpusID:85555872

Pashardes, P., & Savva, C. S. (2009). Factors affecting house prices in Cyprus: 1988-2008. *Cyprus Economic Policy Review*, *3*(1), 3-25. Retrieved from https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f2ffb6ea91ab2027a897701ad1303ceed43771da

Reibel, M., Chernobai, E., & Carney, M. (2008, April). *House price change and highway construction: Spatial and temporal heterogeneity*. [Paper presentation]. American Real Estate Society Conference. Retrieved from https://www.researchgate.net/profile/Michael-Reibel/publication/228354622_House_Price_Change_and_Highway_Construction_Spatial_and_Temporal_Heterogeneity/links/5417078a0cf2fa878ad43f50/House-Price-Change-and-Highway-Construction-Spatial-and-Temporal-Heterogeneity.pdf

Rodriguez, M., & Sirmans, C. F. (1994). Quantifying the value of a view in single-family housing markets. *Appraisal Journal, 62*. 600 - 603. Retrieved from http://www.sbuweb.tcu.edu/mrodriguez/research/viewppr.pdf

Türkoğlu, H. D. (1997). Residents' satisfaction of housing environments: The case of Istanbul, Turkey. *Landscape and Urban Planning*, *39*(1), 55–67. http://doi.org/10.1016/S0169-2046(97)00040-6

Vrbka, S. J., & Combs, E. R. (1993). Predictors of neighborhood and community satisfactions in rural communities. *Housing and Society, 20*(1), 41-49. https://doi.org/10.1080/08882746.1993.11430153

Wilhelmsson, M. (2002). Spatial models in real estate economics. *Housing, Theory and Society*, *19*(2), 92–101. doi:10.1080/140