1. Design

the objective of this project is not only to predict the houses sale. This project leverages the supervised/ unsupervised ML techniques, utilizes different exploratory data analysis and processing techniques, uses different features engineer methods, and then select the appropriate machine learning models.

The general approach that will be used for the core project design follows the standard machine learning life cycle that includes project planning phase, data preparation, exploratory data analysis, features engineering, modeling, and predictions as shown in Figure 1-1. Each step in the diagram shown in Figure 1-1 will be described in the next subsections.

The next step is to collect and consolidate the data, wrangle it, and conduct exploratory data analysis. Once the data is ready to go, the next steps are to select ML models, split the dataset, train the selected models with training data, fine-tune the models, and evaluate the models using the testing data based on the pre-determined success metrics. The last step is to select the best model and productionize it.

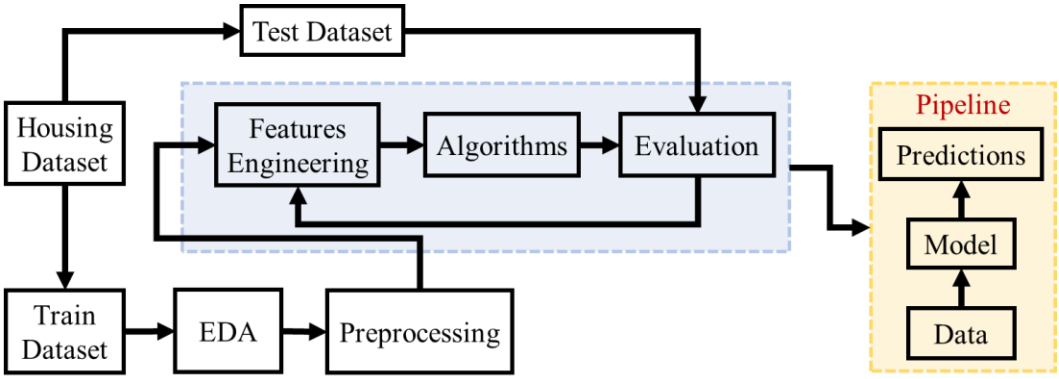


Figure 1-1: The machine learning project life cycle.

1.1. Dataset

The Ames housing dataset will be used to predict house prices [9]. This dataset contains 79 explanatory variables describing (almost) every aspect of residential homes in Ames. The variable number 80 is the Sale Price which is the target variable to be predicted using other variables. It is more complex to handle, containing too many features and variables, missing data, outliers, and both numerical and categorical features.

1.2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis is a very important process to perform initial investigations on dataset, understand the data, discover the patterns in the data to identify anomalies, test hypothesis and to check assumptions with the help of summary statistics and graphical representations. It is all about making sense of data in hand and getting important insights from it.

The dataset has two parts train dataset and test dataset. The size of train dataset is (118,260) with shape of (1460, 81) whereas the size of test set is (116,720) with shape of (1459, 80). Also, the type of data is heterogeneous containing both numerical (quantitative) and categorical (qualitative) data with number of 38 and 43 variables respectively.

After exploring the data, the next step is to explore the target variable which is the "SalePrice". Univariate analysis will be used to get more knowledge about the target variable and other important dependant variables using distribution plots (histogram/displot) and probability plot. Multivariate analysis will be used to understand the relationship between the target variable (SalesPrice) and other variables (predictors). For numerical variables, scatter plot, correlation matrix, and pair plot will be used to study the relationship between target variable and numerical variables. For categorical variables, bar plot and boxplot will be used to study the relationship between target variable and categorical variables. Figure 1-2 summarizes the workflow of exploratory data analysis.

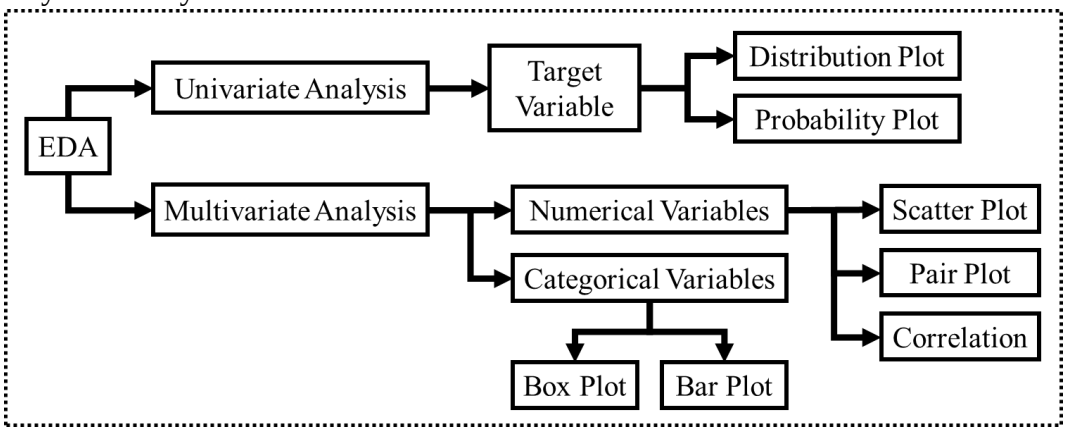


Figure 1-2: Exploratory Data Analysis Workflow.

1.3. Data Processing

Data preprocessing is an extremely important step in machine learning to enhance the quality of data to promote the extraction of meaningful insights from the data. Data preprocessing in machine learning refers to the technique of preparing (cleaning and organizing) the raw data to make it suitable for a building and training machine learning models. The steps in data preprocessing pipeline are shown in Figure 1-3, which includes:

- Outlier detection using visual method and interquartile range rule (IQR) technique.
- Identifying and handling the missing values
- Variable transformation using:
    - Log for target variable
    - Scaling (StandardScaler) for other numerical variables.
- Handling categorical variables by using dummy variables and encoding.

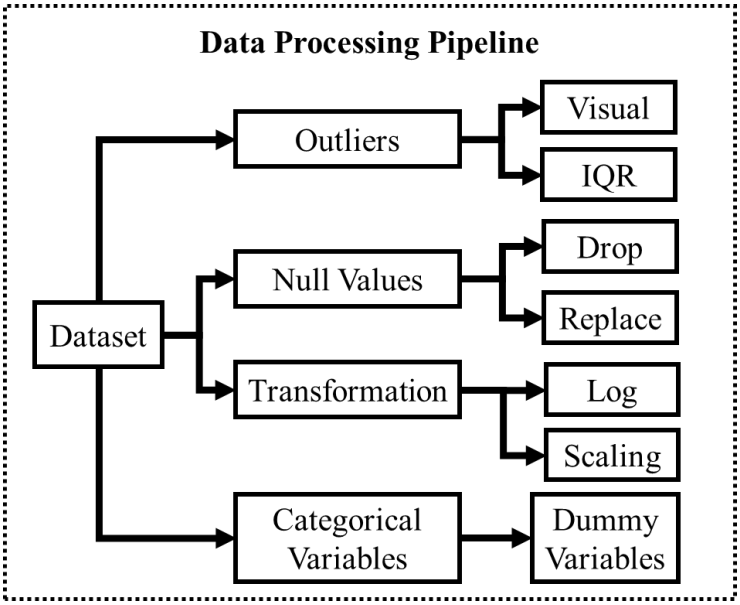


Figure 1-3: Data Processing Pipeline.

1.4. Features Engineering

One of the most important steps in machine learning life cycle is the features engineering where the most important features will be selected, new features may be created, or/and existing features may be simplified or eliminated. The goal of features engineering is to simplify and speed up data transformations while also enhancing the performance of machine learning models.

In this project, many features engineering techniques have been used to create features pipeline as shown in Figure 1-4.

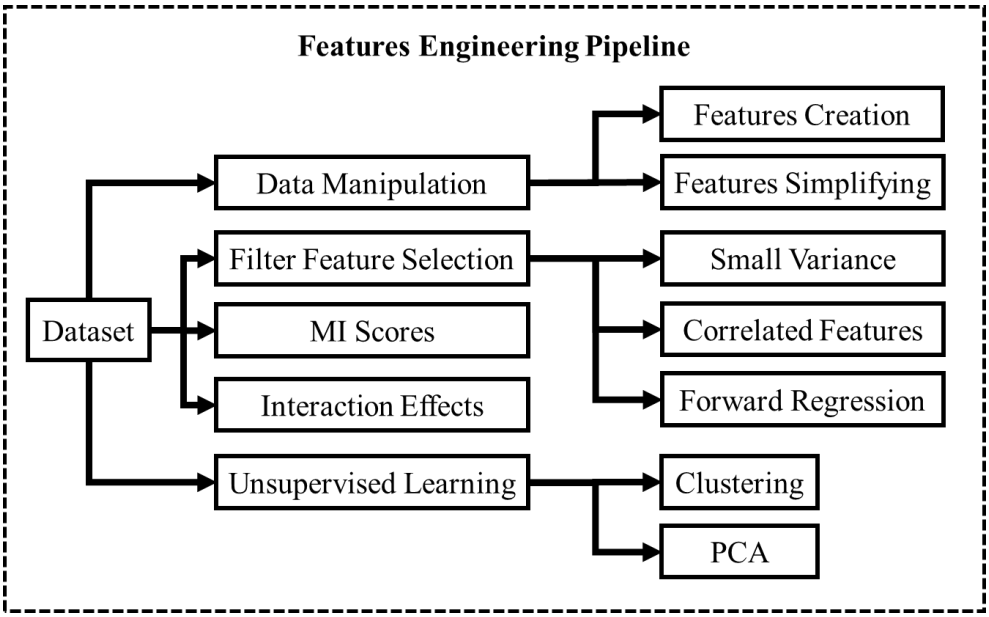


Figure 1-4: Features Engineering Pipeline.

The first step in features engineering is to leverage mathematical transforms and data manipulation using panadas to create new features and simplify existing features. Many techniques are used to create new features as follows:

- Combination of existing features.
- Ratios, creating new features based on the ratio between two variables can often lead to some easy performance gains, for example, creating two new features expressing important ratios using mathematical transformation as follows:
    - LivLotRatio: the ratio of GrLivArea to LotArea.
    - Spaciousness: the sum of FirstFlrSF and SecondFlrSF divided by TotRmsAbvGrd
- Counts, creating new features based on counting some important features, for example: Creating a new feature called PorchTypes that describes how many kinds of outdoor areas a dwelling has. We will count how many of WoodDeckSF, OpenPorchSF, EnclosedPorch, Threeseasonporch, and ScreenPorch are greater than 0.0. Creating another new feature TotalHalfBath that contains the sum of half-bathrooms within the property.
    - Creating new feature called TotalRoom that sums up the total number of rooms (including full and half bathrooms) in each property.
- Grouped transform, the value of a home often depends on how it compares to typical homes in its neighborhood. Therefore, let's create a new feature called MedNhbdArea that describes the median of GrLivArea grouped on Neighborhood.
- Simplification of the Existing Features
- Filter Feature Selection, the feature selection is the process of selecting a subset of relevant features for use in machine learning model construction. The filter feature selection method is one of features selection methods that ranks each feature based on some uni-variate metric and then selects the highest-ranking features. The criteria used in this project for filter feature selectors is based on the following:
    - Removing features with small variance, removing the columns with very little variance. Small variance equals small predictive power because all houses have very similar values. (VarianceThreshold)
    - Removing correlated features, the goal of this part is to remove one feature from each highly correlated pair. This can be done in 3 steps:
        - Calculate a correlation matrix
        - Get pairs of highly correlated features
        - Remove correlated columns
    - Forward Regression
      We have removed the features with no information and correlated features so far. The last thing we will do before modeling is to select the k-best features in terms of the relationship with the target variable. We will use the forward wrapper method for that.
- Mutual information scores, it is one of filter-based feature selection methods. We have a number of features that are highly informative and several that don't seem to be informative at all (at least by themselves). Therefore, we will focus our efforts on the top scoring

features. Training on uninformative features can lead to overfitting as well, so features with 0.0 MI scores are going to be dropped entirely.

- Interaction effects, they occur when the effect of an independent variable depends on the value of another independent variable, for example, checking the interaction effect between GrLivArea and BldgType.
- Unsupervised machine learning:
  The feature selection can leverage unsupervised algorithm to create new features. For this purpose, the clustering approach (i.e., k-mean clustering) and dimensionality-reduction methods (i.e., principal component analysis) can be used to create new features. The **clustering approach** can be used for feature selection. The formation of clusters reduces the dimensionality and helps in selection of the relevant features for the target class by using cluster labels as features. More information will be presented in results section [10].
  Principal Component Analysis (PCA) is another unsupervised learning method to create more new features. It is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets, by identifying important relationships in dataset, transforms the existing data based on these relationships, and then quantifies the importance of these relationships, keeping the most important relationships and drop the others. When reducing dimensionality through PCA, the most important information will be kept by selecting the principal components that explain most of the relationships among the features. More information will be presented in results section.

1.5. Machine Learning Algorithms

The Ames Housing dataset was chosen due to its richness and huge features that allow us to utilize many machine learning techniques at each stage of the ML project's life cycle.

The main goal behind using Ames Housing dataset is to predict the houses prices based on other housing features, therefore the appropriate ML approach is the regression algorithm which is one of supervised ML algorithms. However, unsupervised machine learning approaches can be applied to the dataset as well to create new features during features engineering stage.

The supervise ML algorithms used in this project for regression are Regression Tree Model, Extreme Gradient Boosting (XGBoost) Model, and Linear Regression. For unsupervised ML, Kmeans clustering, hierarchical clustering, PCA will be used to create new features. Figure 1-5 summarizes the ML algorithms used in this project.

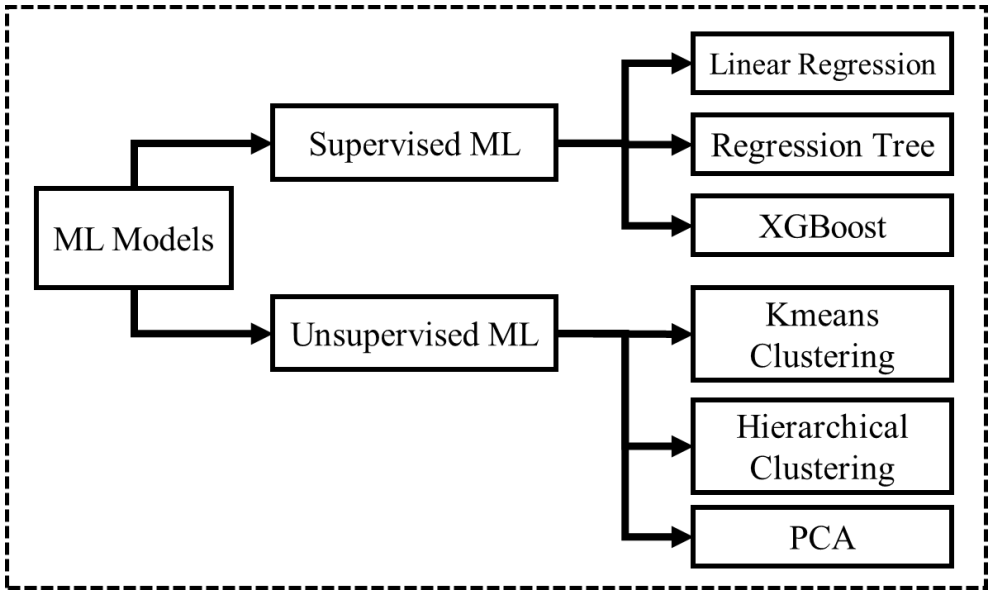


Figure 1-5: Machine learning algorithms used in the project.

1.6. Evaluation of Machine Learning Algorithms

To evaluate the performance of regression models, the root mean squared error (RMSE) will be used to measure and compare the performance of our models. Besides that, the bias and variance will be calculated as well to analyze the bias and variance errors for each model. For clustering models, silhouette score will be used to measure the goodness of a clustering models.

Figure 1-6 summarizes the evaluation metrics used for evaluating the performance of regression and clustering models.

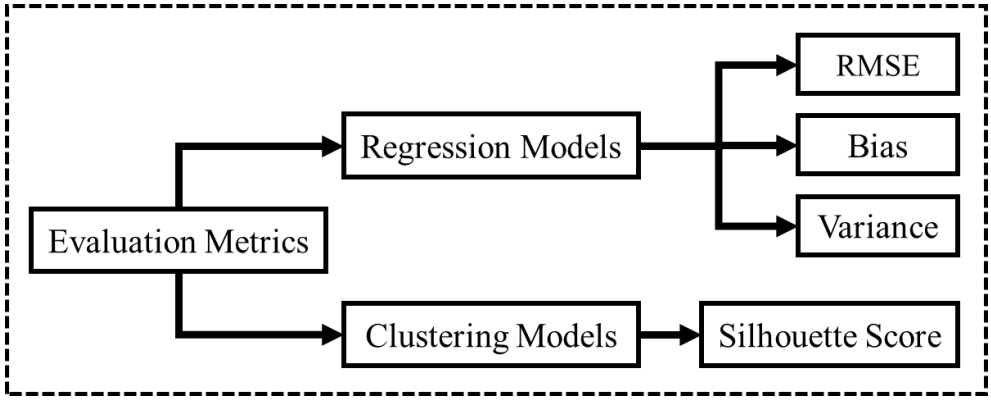


Figure 1-6: Evaluation metrics for regression and clustering models.

1.7. Best Model Selection and Hyperparameters Tuning

The best predictive regression model will be selected based on RSME score compared with other regression models. The model with lowest RSME score is the best model, then the hyperparameters tuning will be done on it for further enhancing its performance. After hyperparameters tuning and checking its performance again, the model will be ready for predictions.

2. Results

2.1. Exploratory Data Analysis (EDA)

2.1.1. Data Discovery

The dataset used in this project is Ames housing dataset that contains train set and test set. The size of trainset is (118,260) with shape of (1460, 81) whereas the size of test set is (116,720) with shape of (1459, 80).

The first step is to check the types of variables the train dataset has. By using Python, the type of variables has been identified. The dataset has 38 numerical (quantitative) variables and 43 categorical (qualitative) variables as well. Also, there are no duplicates in the dataset.

2.1.2. Exploring the Target Variable

The second step, we need to explore the target variable using descriptive statistics and the histogram. The goal of this project is to predict the price of a house. These values are stored in the SalePrice variable. Let's check the descriptive statistics and the histogram of SalePrice variable as shown in Table 2-1.

Table 2-1. Generating descriptive statistics for SalePrice variable.

```
count      1460.000000
mean     180921.195890
std       79442.502883
min       34900.000000
25%      129975.000000
50%      163000.000000
75%      214000.000000
max      755000.000000
Name: SalePrice, dtype: float64
```

Now, lets check the histogram distribution and probability plot for SalePrice variable as shown in Figure 2-1 and Figure 2-2.

Figure 2-1: SalePrice (Target Variable) distribution.

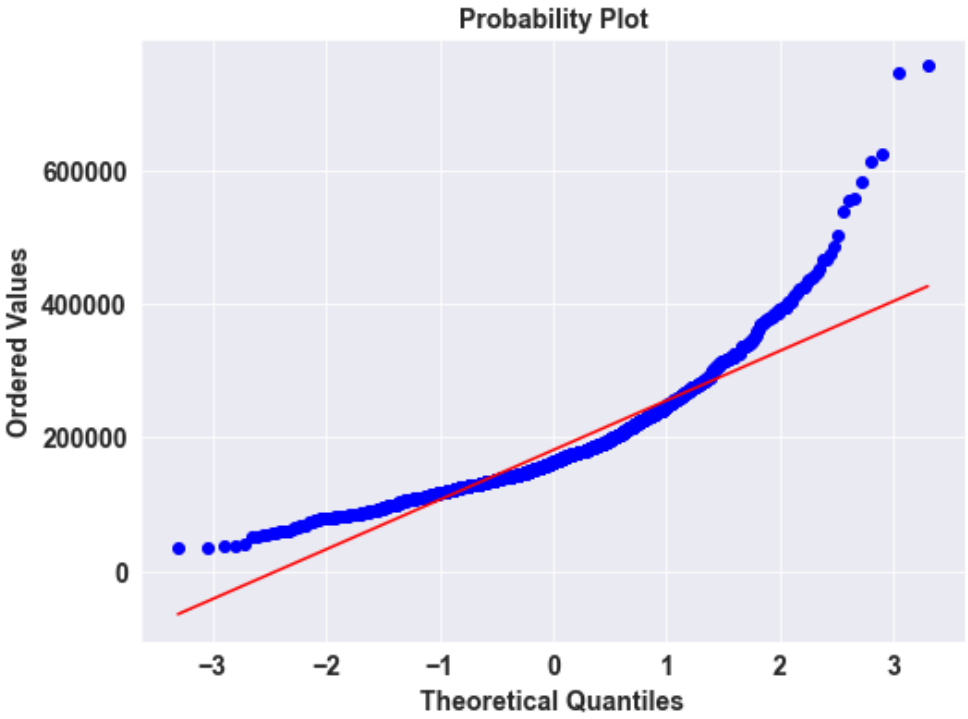


Figure 2-2: The probability plot for SalePrice variable.

Since the histogram is a good chart to check the distribution of variables, we need to calculate the skewness and kurtosis for SalePrice variable as well. The skewness represents the bias of the distribution, and kurtosis represents the sharpness of the distribution. Their calculated values for SalePrice variable are as follows: Skewness = 1.883 whereas Kurtosis = 6.536. The target variable (SalePrice in our case) should follow normality to ensure that the model performs well. The absolute value of skewness is 0-3 and kurtosis is 1-8.

From Figure 2-1 and Figure 2-2 above, we can observe that the distribution has a long tail and most of the density of sale's price lies between 100k and 200k. It means that most of the house are normally distributed but a couple of houses have a higher than normal value, resulting in slightly deviation from a normal distribution. Also, it's skewed and has some outliers. So. It is critical to take this peculiarity into account when designing predictive models.

2.1.3. Numerical and Categorical Variables

For simplicity, we will analyze the numerical and categorical variables separately. So, the third step is to generate descriptive statistics for both numerical and categorical variables using python. These descriptive statistics give us a general idea about the central tendency, dispersion and shape of a dataset's distribution, excluding NaN values.

Table 2-2 shows the summary of these descriptive statistics for numerical variables. Also, we can obtain a concise summary of them including the index data type and columns, non-null values and memory usage as shown in Table 2-3.

After looking at the Table 2-2 and Table 2-3, we can see that 36 numerical features are available, and that the dataset contains 1460 samples. However, some features contain missing values. When looking at min, max, and mean, we can see that some variables contain outliers, and the features are not normally distributed. We will examine the distribution, outliers, and missing values later in this section and in data preprocessing section.

Similarly, Table 2-4 shows the summary of descriptive statistics for categorical variables in term of count, top, unique values, and frequency. Also, we can obtain a concise summary of their information including the index data type and columns, non-null values, and memory usage as shown in Table 2-5.

After looking at the Table 2-4 and Table 2-5, we can see that 43 categorical features are available, and that the dataset contains 1460 samples. However, some features contain missing values. We will examine the distribution, outliers, and missing values later in this section and in data preprocessing section.

Table 2-2. Generating descriptive statistics for numerical variables.

	count	mean	std	min	25%	50%	75%	max
Id	1460.0	730.500000	421.610009	1.0	365.75	730.5	1095.25	1460.0
MSSubClass	1460.0	56.897260	42.300571	20.0	20.00	50.0	70.00	190.0
LotFrontage	1201.0	70.049958	24.284752	21.0	59.00	69.0	80.00	313.0
LotArea	1460.0	10516.828082	9981.264932	1300.0	7553.50	9478.5	11601.50	215245.0
OverallQual	1460.0	6.099315	1.382997	1.0	5.00	6.0	7.00	10.0
OverallCond	1460.0	5.575342	1.112799	1.0	5.00	5.0	6.00	9.0
YearBuilt	1460.0	1971.267808	30.202904	1872.0	1954.00	1973.0	2000.00	2010.0
YearRemodAdd	1460.0	1984.865753	20.645407	1950.0	1967.00	1994.0	2004.00	2010.0
MasVnrArea	1452.0	103.685262	181.066207	0.0	0.00	0.0	166.00	1600.0
BsmtFinSF1	1460.0	443.639726	456.098091	0.0	0.00	383.5	712.25	5644.0
BsmtFinSF2	1460.0	46.549315	161.319273	0.0	0.00	0.0	0.00	1474.0
BsmtUnfSF	1460.0	567.240411	441.866955	0.0	223.00	477.5	808.00	2336.0
TotalBsmtSF	1460.0	1057.429452	438.705324	0.0	795.75	991.5	1298.25	6110.0
1stFlrSF	1460.0	1162.626712	386.587738	334.0	882.00	1087.0	1391.25	4692.0
2ndFlrSF	1460.0	346.992466	436.528436	0.0	0.00	0.0	728.00	2065.0
LowQualFinSF	1460.0	5.844521	48.623081	0.0	0.00	0.0	0.00	572.0
GrLivArea	1460.0	1515.463699	525.480383	334.0	1129.50	1464.0	1776.75	5642.0
BsmtFullBath	1460.0	0.425342	0.518911	0.0	0.00	0.0	1.00	3.0
BsmtHalfBath	1460.0	0.057534	0.238753	0.0	0.00	0.0	0.00	2.0
FullBath	1460.0	1.565068	0.550916	0.0	1.00	2.0	2.00	3.0
HalfBath	1460.0	0.382877	0.502885	0.0	0.00	0.0	1.00	2.0
BedroomAbvGr	1460.0	2.866438	0.815778	0.0	2.00	3.0	3.00	8.0
KitchenAbvGr	1460.0	1.046575	0.220338	0.0	1.00	1.0	1.00	3.0
TotRmsAbvGrd	1460.0	6.517808	1.625393	2.0	5.00	6.0	7.00	14.0
Fireplaces	1460.0	0.613014	0.644666	0.0	0.00	1.0	1.00	3.0
GarageYrBlt	1379.0	1978.506164	24.689725	1900.0	1961.00	1980.0	2002.00	2010.0
GarageCars	1460.0	1.767123	0.747315	0.0	1.00	2.0	2.00	4.0
GarageArea	1460.0	472.980137	213.804841	0.0	334.50	480.0	576.00	1418.0
WoodDeckSF	1460.0	94.244521	125.338794	0.0	0.00	0.0	168.00	857.0
OpenPorchSF	1460.0	46.660274	66.256028	0.0	0.00	25.0	68.00	547.0
EnclosedPorch	1460.0	21.954110	61.119149	0.0	0.00	0.0	0.00	552.0
3SsnPorch	1460.0	3.409589	29.317331	0.0	0.00	0.0	0.00	508.0
ScreenPorch	1460.0	15.060959	55.757415	0.0	0.00	0.0	0.00	480.0
PoolArea	1460.0	2.758904	40.177307	0.0	0.00	0.0	0.00	738.0
MiscVal	1460.0	43.489041	496.123024	0.0	0.00	0.0	0.00	15500.0
MoSold	1460.0	6.321918	2.703626	1.0	5.00	6.0	8.00	12.0
YrSold	1460.0	2007.815753	1.328095	2006.0	2007.00	2008.0	2009.00	2010.0
SalePrice	1460.0	180921.195890	79442.502883	34900.0	129975.00	163000.0	214000.00	755000.0

Table 2-3. Generating concise information summary for numerical variables.

```
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 36 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   MSSubClass     1460 non-null   int64
 1   LotFrontage    1201 non-null   float64
 2   LotArea        1460 non-null   int64
 3   OverallQual    1460 non-null   int64
 4   OverallCond    1460 non-null   int64
 5   YearBuilt      1460 non-null   int64
 6   YearRemodAdd   1460 non-null   int64
 7   MasVnrArea     1452 non-null   float64
 8   BsmtFinSF1     1460 non-null   int64
 9   BsmtFinSF2     1460 non-null   int64
 10  BsmtUnfSF      1460 non-null   int64
 11  TotalBsmtSF    1460 non-null   int64
 12  1stFlrSF       1460 non-null   int64
 13  2ndFlrSF       1460 non-null   int64
 14  LowQualFinSF   1460 non-null   int64
 15  GrLivArea      1460 non-null   int64
 16  BsmtFullBath   1460 non-null   int64
 17  BsmtHalfBath   1460 non-null   int64
 18  FullBath       1460 non-null   int64
 19  HalfBath       1460 non-null   int64
 20  BedroomAbvGr   1460 non-null   int64
 21  KitchenAbvGr   1460 non-null   int64
 22  TotRmsAbvGrd   1460 non-null   int64
 23  Fireplaces     1460 non-null   int64
 24  GarageYrBlt    1379 non-null   float64
 25  GarageCars     1460 non-null   int64
 26  GarageArea     1460 non-null   int64
 27  WoodDeckSF     1460 non-null   int64
 28  OpenPorchSF    1460 non-null   int64
 29  EnclosedPorch  1460 non-null   int64
 30  3SsnPorch      1460 non-null   int64
 31  ScreenPorch    1460 non-null   int64
 32  PoolArea       1460 non-null   int64
 33  MiscVal        1460 non-null   int64
 34  MoSold         1460 non-null   int64
 35  YrSold         1460 non-null   int64
dtypes: float64(3), int64(33)
memory usage: 410.8 KB
```

Table 2-4. Generating descriptive statistics for categorical variables.

	count	unique	top	freq
MSZoning	1460	5	RL	1151
Street	1460	2	Pave	1454
Alley	91	2	Grvl	50
LotShape	1460	4	Reg	925
LandContour	1460	4	Lvl	1311
Utilities	1460	2	AllPub	1459
LotConfig	1460	5	Inside	1052
LandSlope	1460	3	Gtl	1382
Neighborhood	1460	25	NAmes	225
Condition1	1460	9	Norm	1260
Condition2	1460	8	Norm	1445
BldgType	1460	5	1Fam	1220
HouseStyle	1460	8	1Story	726
RoofStyle	1460	6	Gable	1141
RoofMatl	1460	8	CompShg	1434
Exterior1st	1460	15	VinylSd	515
Exterior2nd	1460	16	VinylSd	504
MasVnrType	1452	4	None	864
ExterQual	1460	4	TA	906
ExterCond	1460	5	TA	1282
Foundation	1460	6	PConc	647
BsmtQual	1423	4	TA	649
BsmtCond	1423	4	TA	1311
BsmtExposure	1422	4	No	953
BsmtFinType1	1423	6	Unf	430
BsmtFinType2	1422	6	Unf	1256
Heating	1460	6	GasA	1428
HeatingQC	1460	5	Ex	741
CentralAir	1460	2	Y	1365
Electrical	1459	5	SBrkr	1334
KitchenQual	1460	4	TA	735
Functional	1460	7	Typ	1360
FireplaceQu	770	5	Gd	380
GarageType	1379	6	Attchd	870
GarageFinish	1379	3	Unf	605
GarageQual	1379	5	TA	1311
GarageCond	1379	5	TA	1326
PavedDrive	1460	3	Y	1340
PoolQC	7	3	Gd	3
Fence	281	4	MnPrv	157
MiscFeature	54	4	Shed	49
SaleType	1460	9	WD	1267
SaleCondition	1460	6	Normal	1198

Table 2-5. Generating concise information summary for categorical variables.

```
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 43 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   MSZoning       1460 non-null   object
 1   Street         1460 non-null   object
 2   Alley          91 non-null     object
 3   LotShape       1460 non-null   object
 4   LandContour    1460 non-null   object
 5   Utilities      1460 non-null   object
 6   LotConfig      1460 non-null   object
 7   LandSlope      1460 non-null   object
 8   Neighborhood   1460 non-null   object
 9   Condition1     1460 non-null   object
 10  Condition2     1460 non-null   object
 11  BldgType       1460 non-null   object
 12  HouseStyle     1460 non-null   object
 13  RoofStyle      1460 non-null   object
 14  RoofMatl       1460 non-null   object
 15  Exterior1st    1460 non-null   object
 16  Exterior2nd    1460 non-null   object
 17  MasVnrType     1452 non-null   object
 18  ExterQual      1460 non-null   object
 19  ExterCond      1460 non-null   object
 20  Foundation     1460 non-null   object
 21  BsmtQual       1423 non-null   object
 22  BsmtCond       1423 non-null   object
 23  BsmtExposure   1422 non-null   object
 24  BsmtFinType1   1423 non-null   object
 25  BsmtFinType2   1422 non-null   object
 26  Heating        1460 non-null   object
 27  HeatingQC      1460 non-null   object
 28  CentralAir     1460 non-null   object
 29  Electrical     1459 non-null   object
 30  KitchenQual    1460 non-null   object
 31  Functional     1460 non-null   object
 32  FireplaceQu    770 non-null    object
 33  GarageType     1379 non-null   object
 34  GarageFinish   1379 non-null   object
 35  GarageQual     1379 non-null   object
 36  GarageCond     1379 non-null   object
 37  PavedDrive     1460 non-null   object
 38  PoolQC         7 non-null      object
 39  Fence          281 non-null    object
 40  MiscFeature    54 non-null     object
 41  SaleType       1460 non-null   object
 42  SaleCondition  1460 non-null   object
dtypes: object(43)
memory usage: 490.6+ KB
```

2.1.4. Univariate Analysis for Numerical Variables

To check the distribution of all numerical variables, we plotted them together as shown in Figure 2-3 and Figure 2-4. They show that some variables seem to be able to follow a normal distribution through log transformation or box cox transformation.

Some variables were extremely biased toward one value (0) since they have high picks for 0. It could be linked that this value was assigned when the criterion did not apply, for instance the area of the swimming pool when no swimming pools are available. So, they don't seem to be very important variables.

We also have some feature encoding some date (for instance year). This information is useful and should also be considered when designing a predictive model. Since these plots are packed together, we can take random variables and plot them separately to look at their distribution more clearly as shown in Figure 2-5 and Figure 2-6.

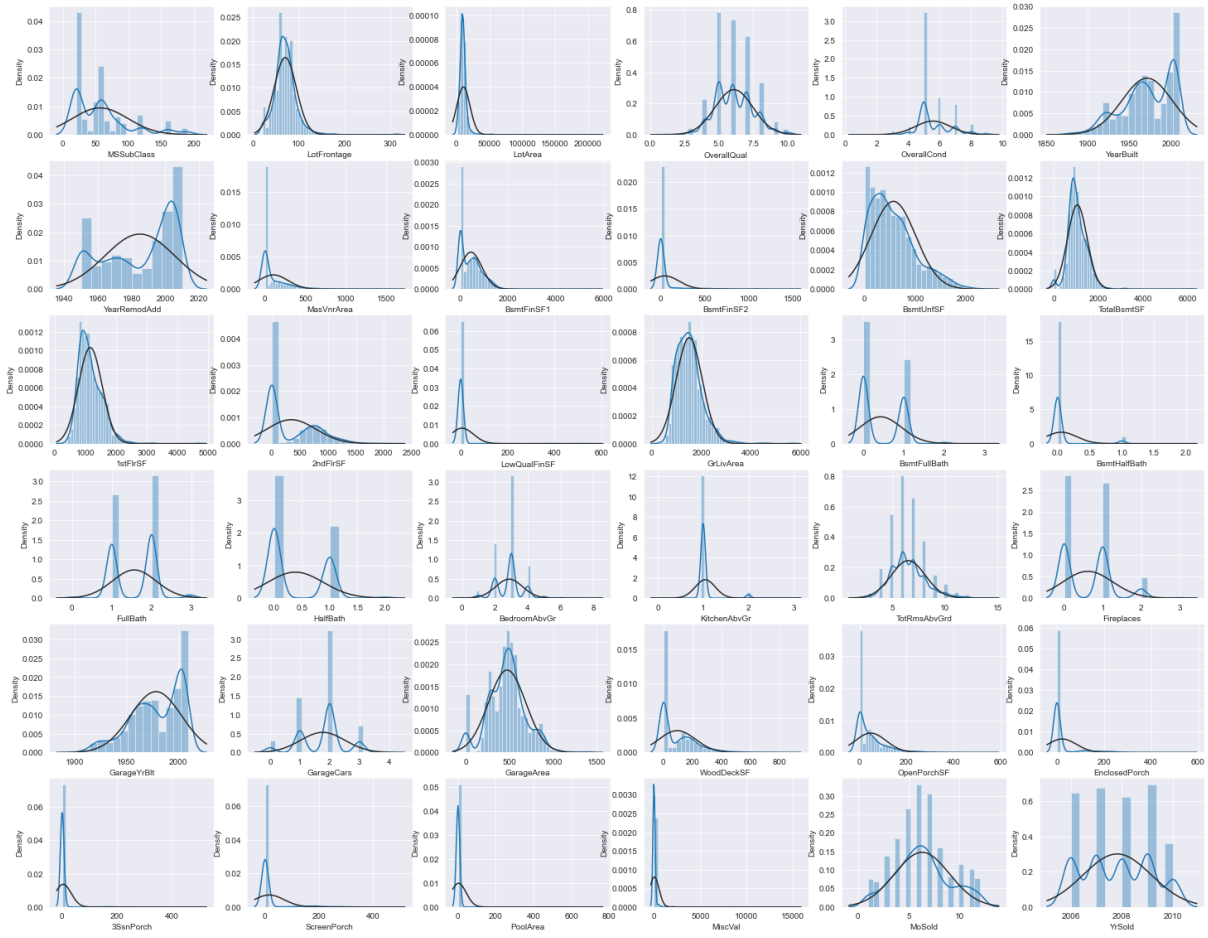
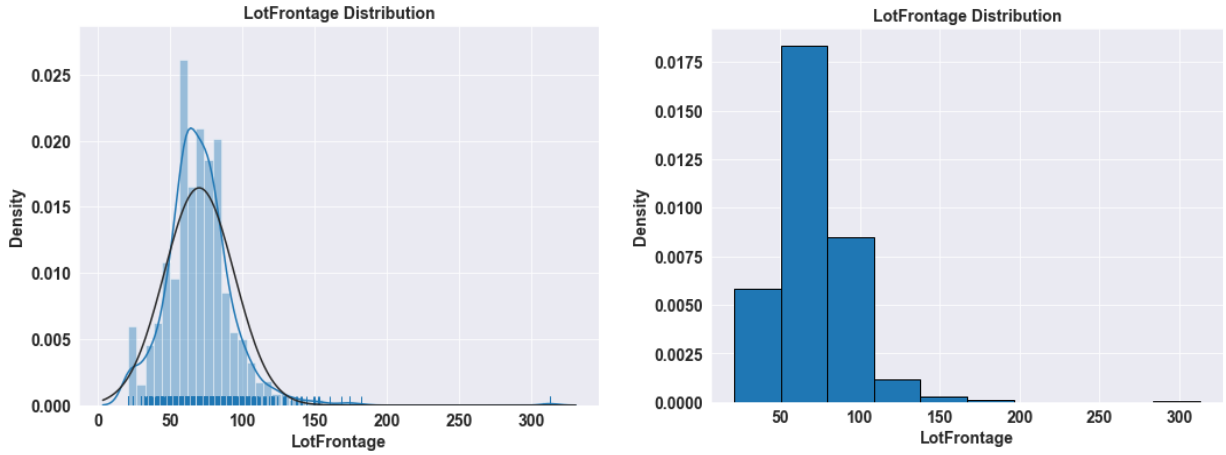


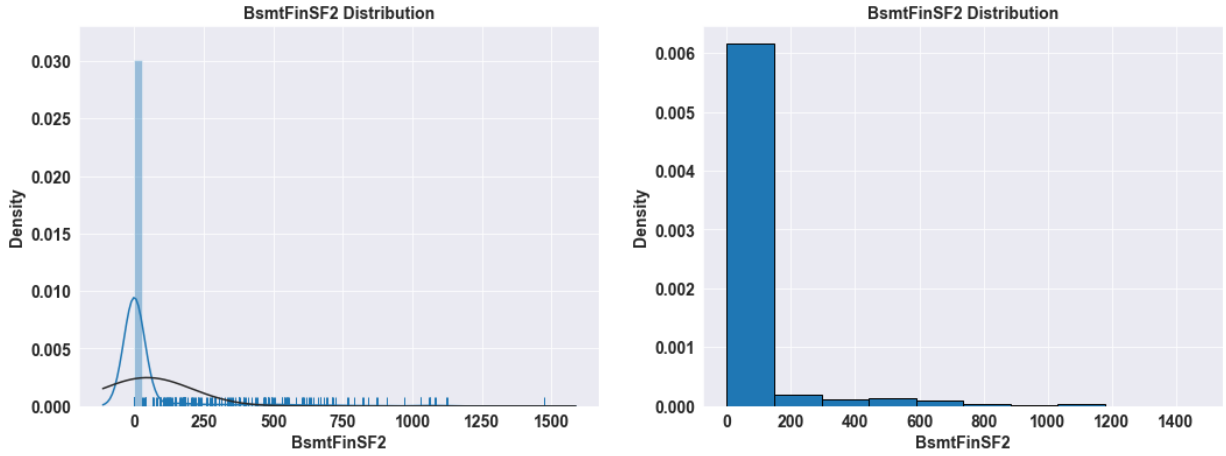
Figure 2-3: The distribution plots for numerical variables using distplot.

Figure 2-4: The distribution plots for numerical variables using histplot.



a) distplot b) hist plot

Figure 2-5: The distribution plots for LoftFrontage variable.



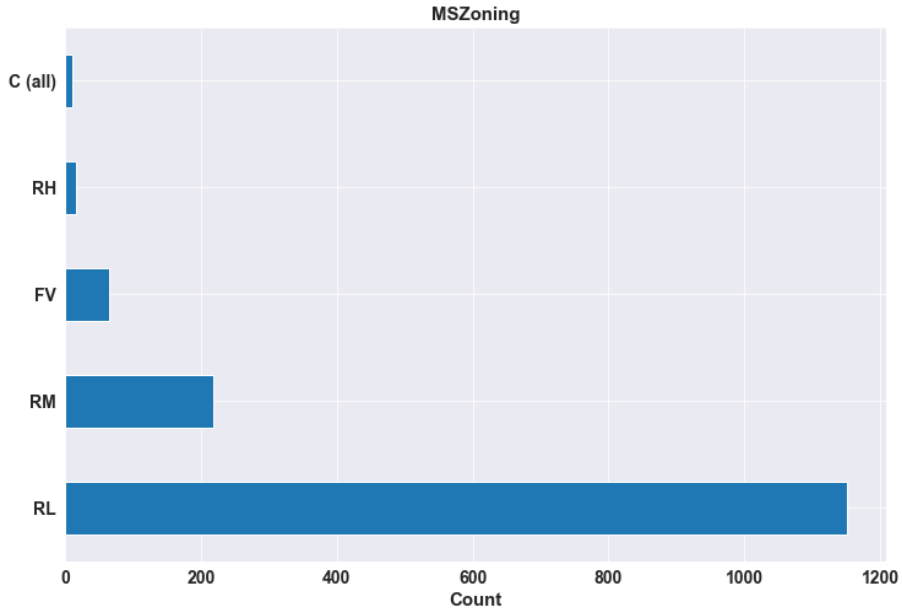
a) distplot b) histplot

Figure 2-6: The distribution plots for BsmtFinSF2 variable.

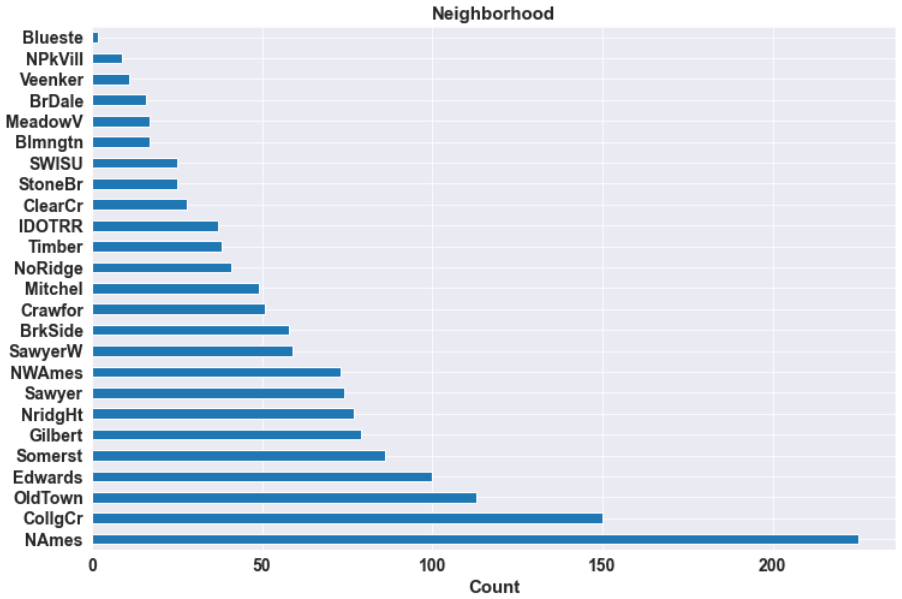
2.1.5. Univariate Analysis for Categorical Variables

For categorical variables, we created bar plots to see categories count for all categorical variables. Plotting this information allows us to find how many categories a given feature has and discover if there are rare categories for some features since knowing that would help at designing the predictive pipeline.

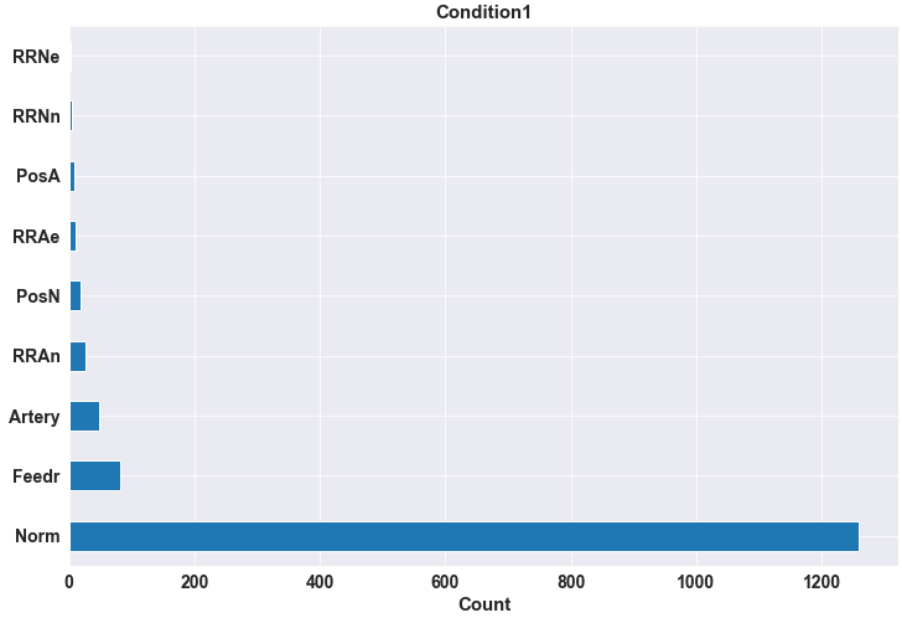
Since there are too many categorical variables, we provided plots for selected categorical variables that we think they have good disparity with respect to SalePrice. We will verify this hypothesis later when analysing the relationship between categorical variables to SalePrice in multivariate analysis. Figure 2-7 (a-m) shows multiple plots for categories count for selected categorical variables including MSZoning, Neighborhood, Condition1, Condition2, RoofMatl, MasVnrType, ExterQual, BsmtQual, BsmtCond, CentralAir, KitchenQual, SaleType, and SaleCondition respectively.


MSZoning
C (all)
RH
FV
RM
RL
0
200
400
600
800
1000
1200
Count

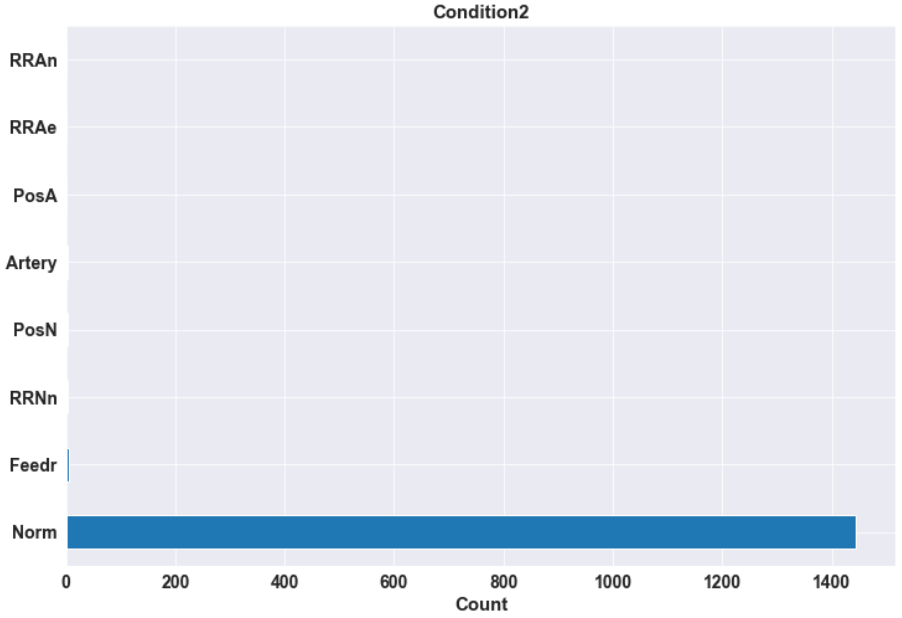
(a)


Neighborhood
Blueste
NPkVill
Veenker
BrDale
MeadowV
Blmngtn
SWISU
StoneBr
ClearCr
IDOTRR
Timber
NoRidge
Mitchel
Crawfor
BrkSide
SawyerW
NWAmes
Sawyer
NridgHt
Gilbert
Somerst
Edwards
OldTown
CollgCr
NAmes
0
50
100
150
200
Count

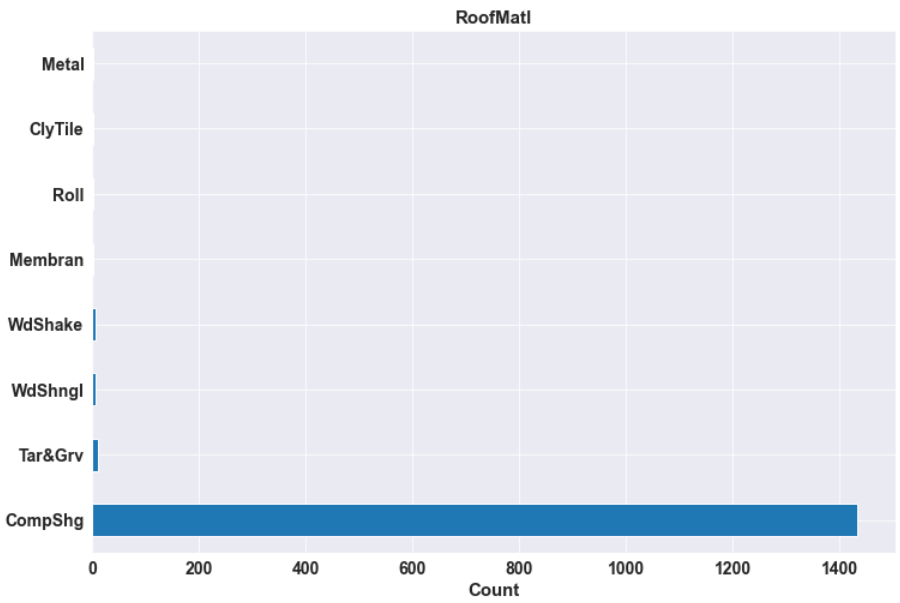
(b)


Condition1
RRNe
RRNn
PosA
RRAe
PosN
RRAn
Artery
Feedr
Norm
0
200
400
600
800
1000
1200
Count

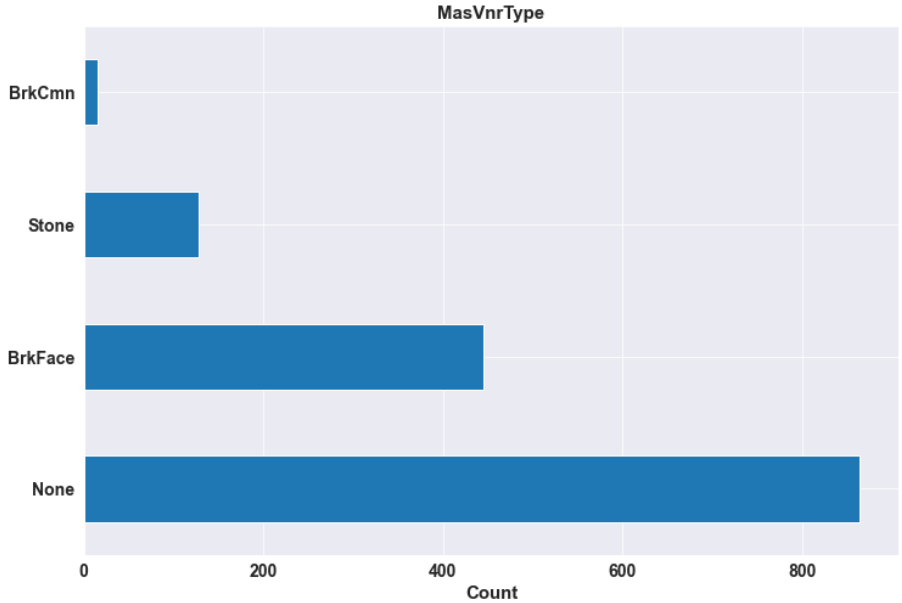
(c)


Condition2
RRAn
RRAe
PosA
Artery
PosN
RRNn
Feedr
Norm
0
200
400
600
800
1000
1200
1400
Count

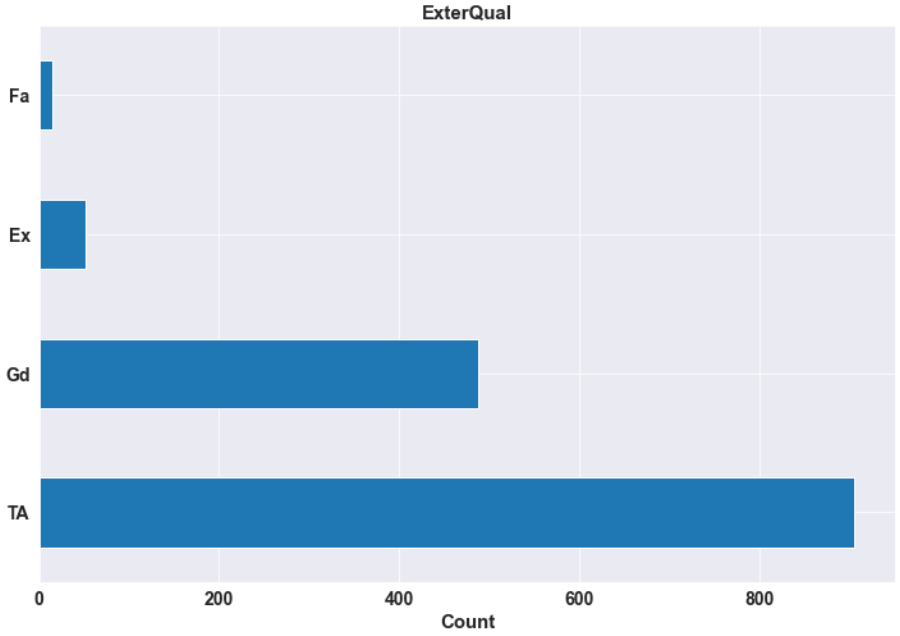
(d)


RoofMatl
Metal
ClyTile
Roll
Membran
WdShake
WdShngl
Tar&Grv
CompShg
0
200
400
600
800
1000
1200
1400
Count

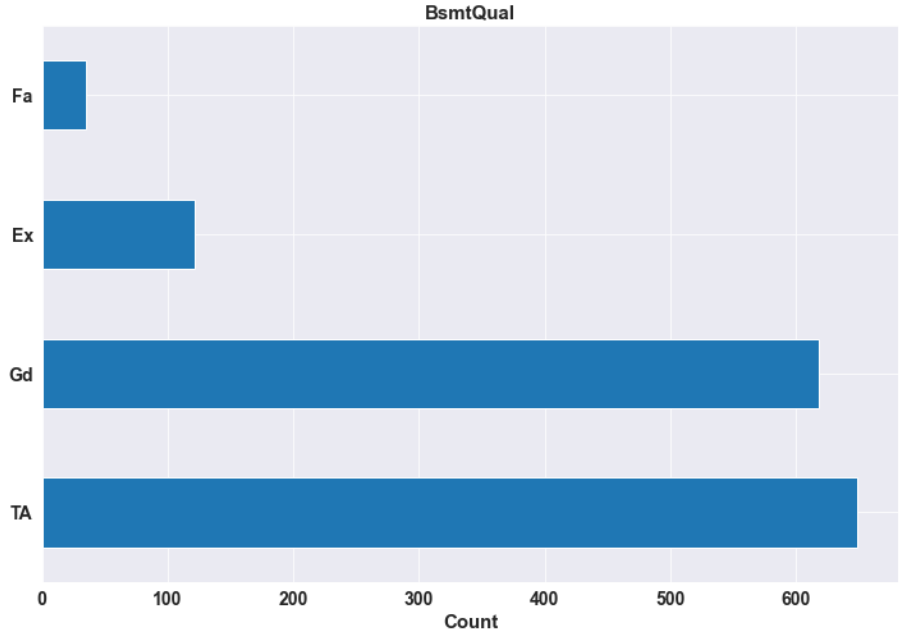
(e)


MasVnrType
BrkCmn
Stone
BrkFace
None
0
200
400
600
800
Count

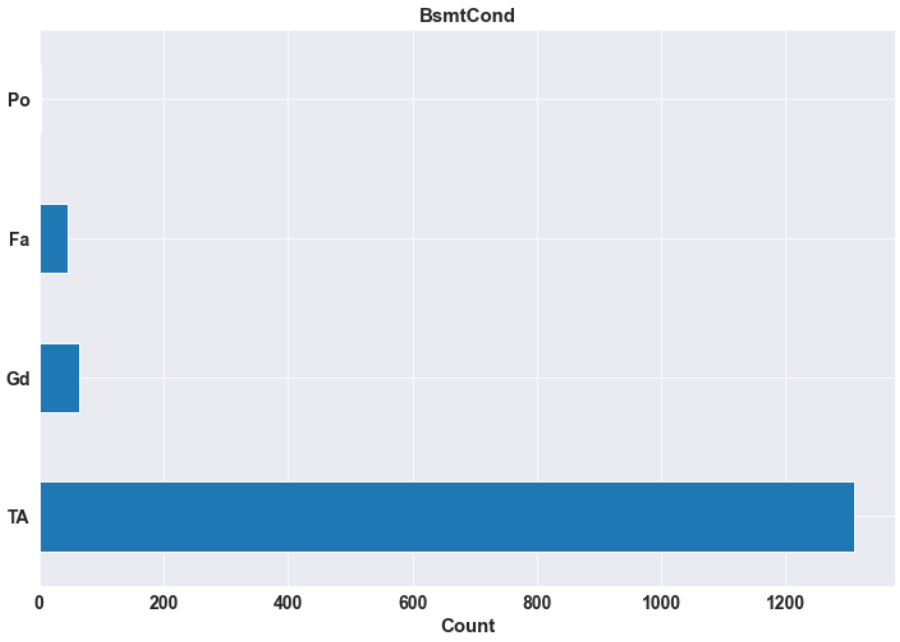
(f)


ExterQual
Fa
Ex
Gd
TA
0
200
400
600
800
Count

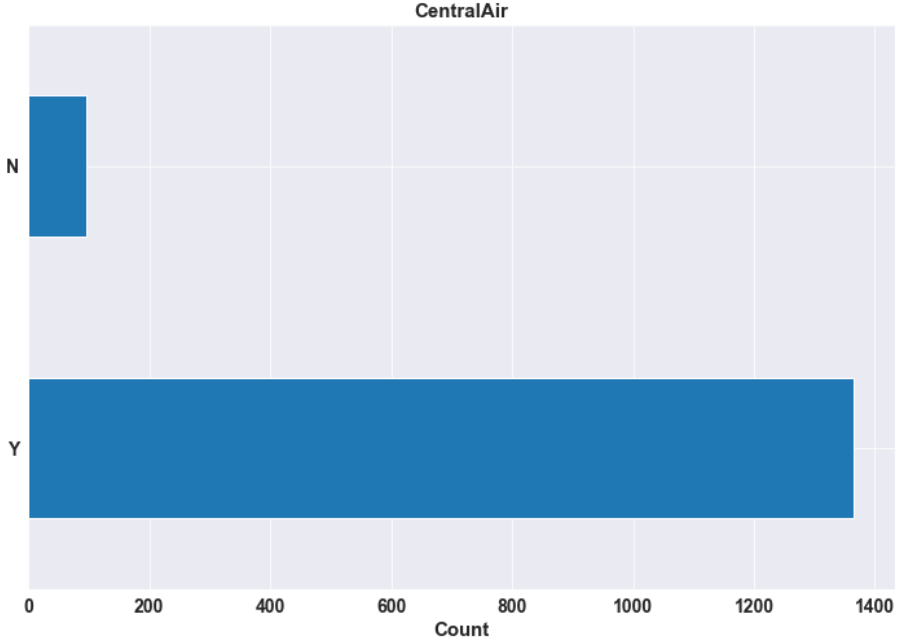
(g)


BsmtQual
Fa
Ex
Gd
TA
0
100
200
300
400
500
600
Count

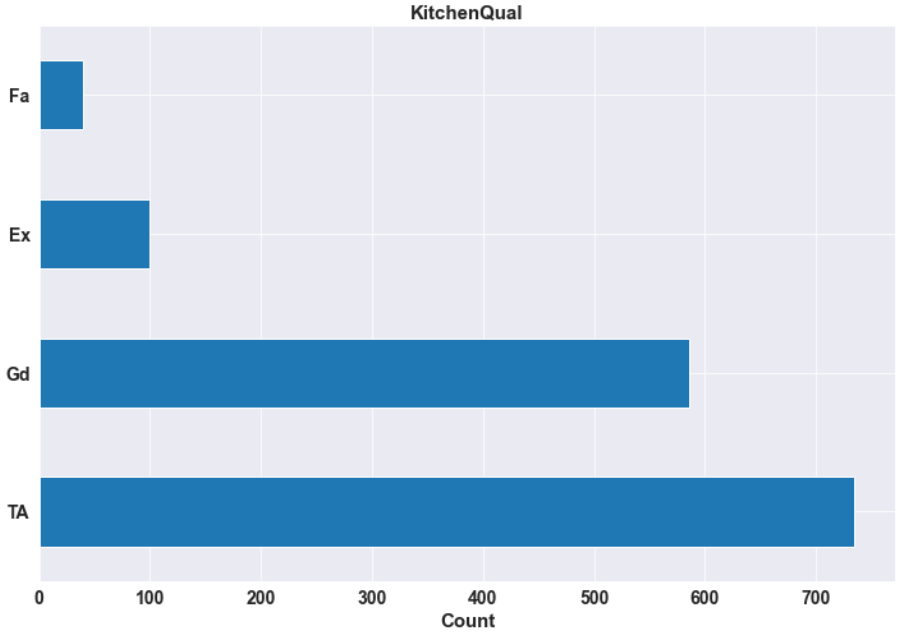
(h)

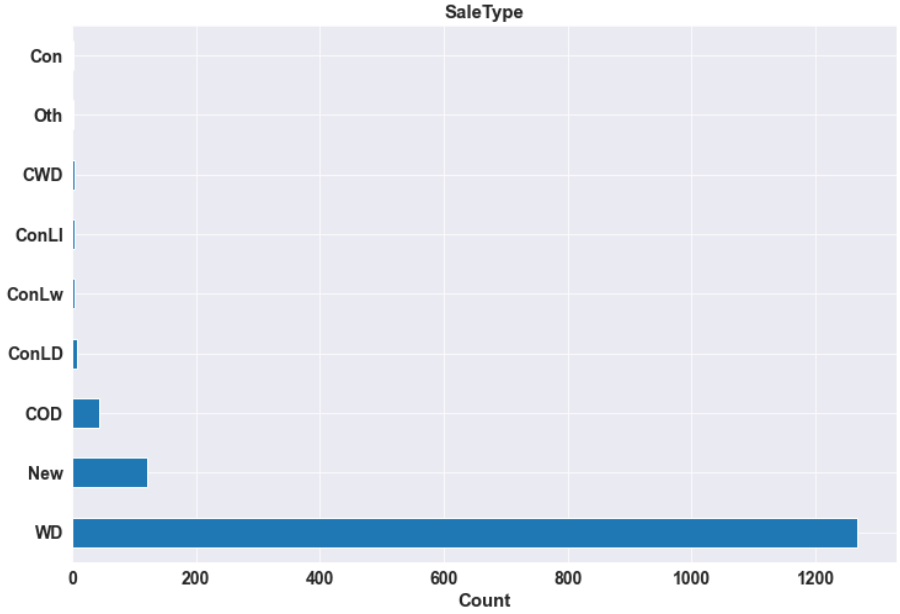

BsmtCond
Po
Fa
Gd
TA
0
200
400
600
800
1000
1200
Count

(i)

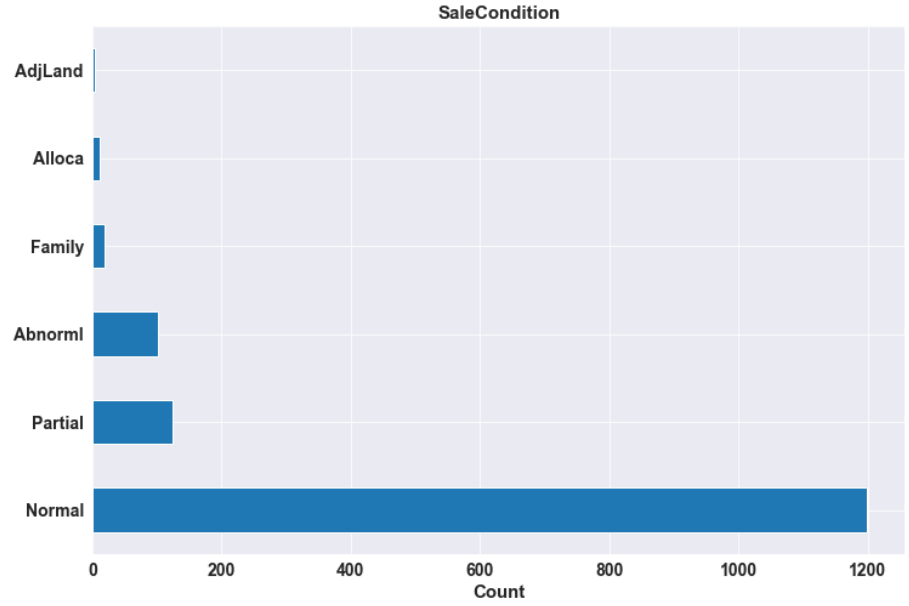

CentralAir
N
Y
0
200
400
600
800
1000
1200
1400
Count

(j)


KitchenQual
Fa
Ex
Gd
TA
0
100
200
300
400
500
600
700
Count


SaleType
Con
Oth
CWD
ConLI
ConLw
ConLD
COD
New
WD
0
200
400
600
800
1000
1200
Count

(k) (l)



(m)

Figure 2-7: The categories count for multiple selected categorical variables.

2.1.6. Multivariate Analysis for Numerical Variables

Now, we can explore the relationship between numerical variables and target variable using scatter plots, for example, Figure 2-8 and Figure 2-9 show the relationship of GrLivArea and TotalBsmtSF with regard to SalesPrice respectively. We can see a linear relationship in the Figure 2-8 and a quadratic relationship in Figure 2-9.

Figure 2-8: The relationship plot between SalePrice and GrLivArea.

Figure 2-9: The relationship plot between SalePrice and TotalBsmtSF.

We need to check the correlation of numerical variables with SalePrice variable as the target variable. Figure 2-10 shows the correlation of all numerical variables with SalePrice whereas Figure 2-11 shows the correlation matrix between numeric attributes.

	SalePrice
MSSubClass	0.084284
LotFrontage	0.351799
LotArea	0.263843
OverallQual	0.790982
OverallCond	0.077856
YearBuilt	0.522897
YearRemodAdd	0.507101
MasVnrArea	0.477493
BsmtFinSF1	0.386420
BsmtFinSF2	0.011378
BsmtUnfSF	0.214479
TotalBsmtSF	0.613581
1stFlrSF	0.605852
2ndFlrSF	0.319334
LowQualFinSF	0.025606
GrLivArea	0.708624
BsmtFullBath	0.227122
BsmtHalfBath	0.016844
FullBath	0.560664

	SalePrice
HalfBath	0.284108
BedroomAbvGr	0.168213
KitchenAbvGr	0.135907
TotRmsAbvGrd	0.533723
Fireplaces	0.466929
GarageYrBlt	0.486362
GarageCars	0.640409
GarageArea	0.623431
WoodDeckSF	0.324413
OpenPorchSF	0.315856
EnclosedPorch	0.128578
3SsnPorch	0.044584
ScreenPorch	0.111447
PoolArea	0.092404
MiscVal	0.021190
MoSold	0.046432
YrSold	0.028923
SalePrice	1.000000

Figure 2-10: The correlation of all numerical variables with SalePrice.

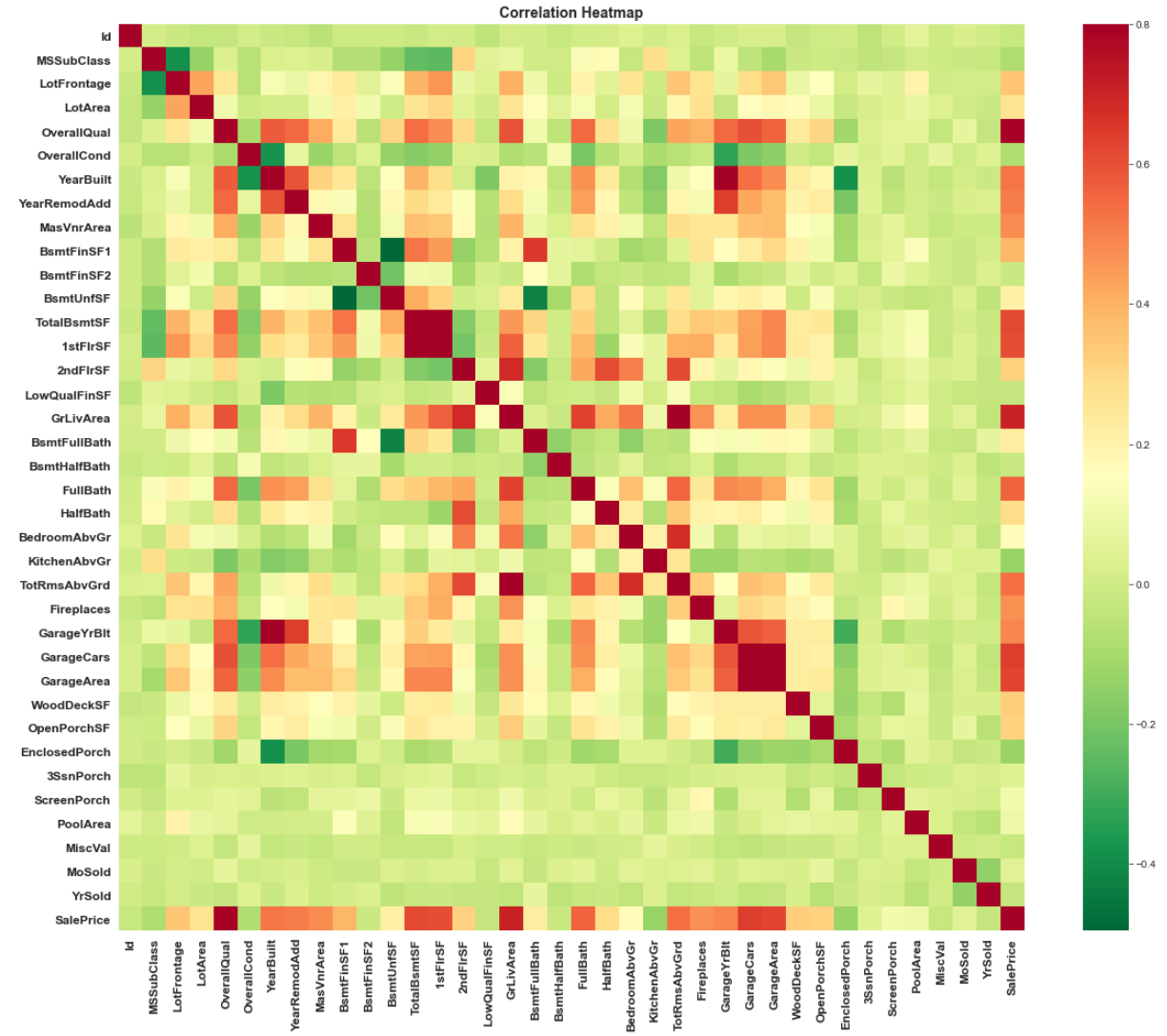


Figure 2-11: The correlation matrix between numeric attributes.

To make the correlation matrix more readable, we can plot only those variables that correlate with the target variable more than 0.5 to obtain the most correlated features with SalePrice. Figure 2-12 shows the features which the correlation is higher than 0.50 and Figure 2-13 shows the correlation matrix for most top correlated features.

Based on correltion values and correlation matrix, we can see that OverallQual, GrLivArea, TotalBsmtSF, GarageCars, GarageArea, YearBuilt, and FullBath are strongly correlated with SalePrice. GarageCars and GarageArea are like twin brothers. You'll never be able to distinguish them. Therefore, we just need one of these variables in our analysis (we can keep GarageCars since its correlation with SalePrice is higher). TotalBsmtSF and 1stFloor also seem to be twin brothers. TotRmsAbvGrd and GrLivArea are twin brothers again. However, 1stFlrSF, GarageCars, TotRmsAbvGrd are excluded because they are related to one of those most correlated ones. The relationship between these top correlated variables and target variable (SalePrice) is shown in Figure 2-14 (a-f).

	SalePrice
OverallQual	0.790982
YearBuilt	0.522897
YearRemodAdd	0.507101
TotalBsmtSF	0.613581
1stFlrSF	0.605852
GrLivArea	0.708624
FullBath	0.560664
TotRmsAbvGrd	0.533723
GarageCars	0.640409
GarageArea	0.623431
SalePrice	1.000000

Figure 2-12: The correlation of numerical variables with SalePrice when correlation > 0.5

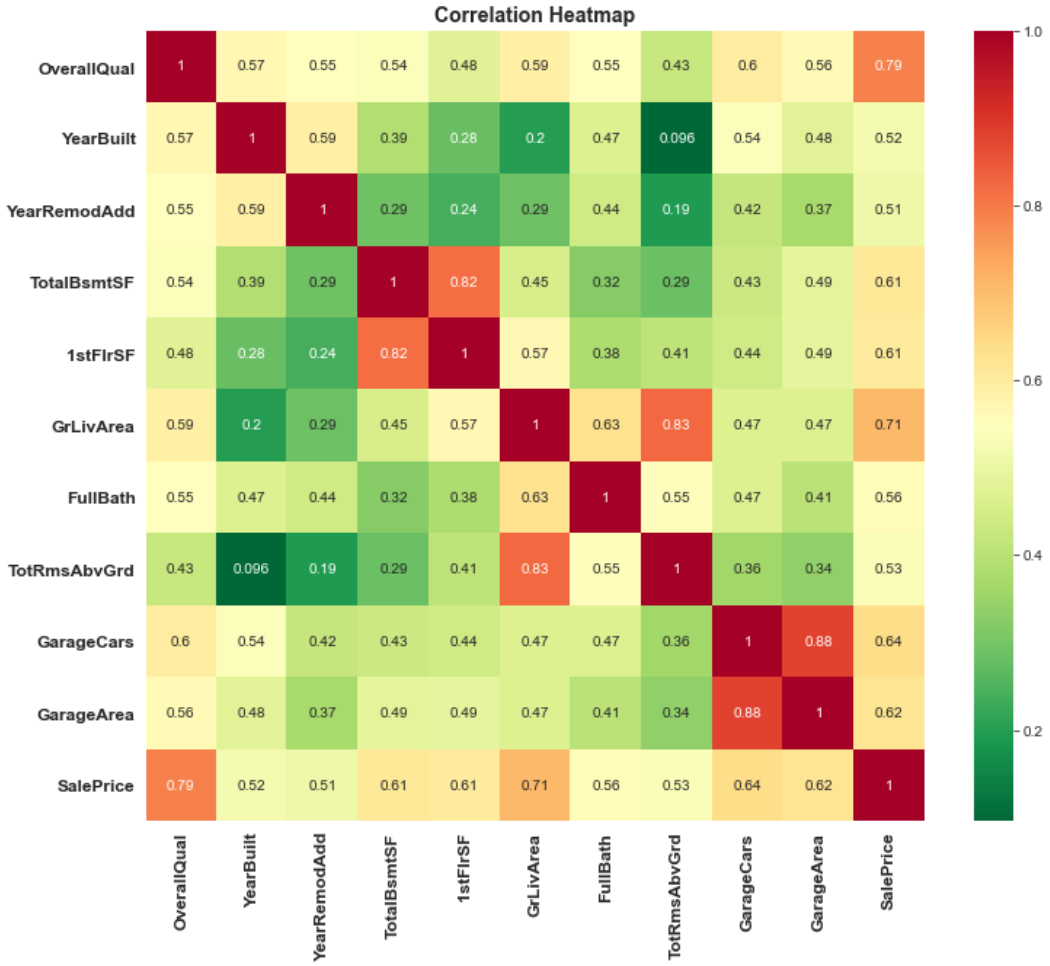


Figure 2-13: The correlation matrix for most top correlated features.


SalePrice vs TotalBsmtSF
SalePrice
TotalBsmtSF
0
100000
200000
300000
400000
500000
600000
700000
800000
0
1000
2000
3000
4000
5000
6000

(a)


SalePrice vs GarageCars
SalePrice
GarageCars
0
100000
200000
300000
400000
500000
600000
700000
800000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

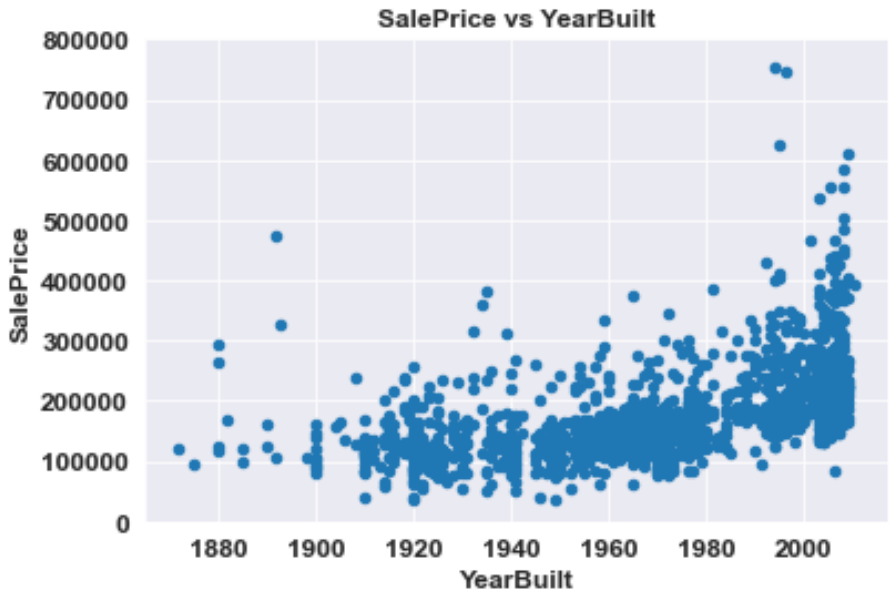
(b)


SalePrice vs OverallQual
SalePrice
OverallQual
0
100000
200000
300000
400000
500000
600000
700000
800000
2
4
6
8
10

(c)


SalePrice vs GrLivArea
SalePrice
GrLivArea
0
100000
200000
300000
400000
500000
600000
700000
800000
1000
2000
3000
4000
5000

(d)



(e)

(f)

Figure 2-14: The relationship between SlaPrice and top correlated variables.

From those charts, highly correlated variables mentioned above have sort of linear relationship with 'SalePrice'. Those charts also imply some exponential relationships, so we can log transform some features to get a better model. Candidates for log transformation: TotalBsmtSF, GrLivArea.

2.1.7. Multivariate Analysis for Categorical Variables

Also, we can explore the relationship between categorical variables and target variable using boxplot. We performed analysis for boxplot for all categorical variables and found that variables having good disparity with respect to SalePrice are: MSZoning, Neighborhood, Condition1, Condition2, RoofMatl, MatVnrType, ExterQual, BsmtQual, BsmtCond, CentralAir, KitchenQual, SaleType, SaleCondition. For simplicity, we will present only these plots in this report.

Figure 2-15 shows the relationship between Neighborhood and SalePrice. We can see that there are some outliers need to be eliminated. Similarly, Figure 2-16 (a-l) shows the relationship between selected categorical variables mentioned above and SalePrice. We can see that there are some outliers need to be eliminated as well.

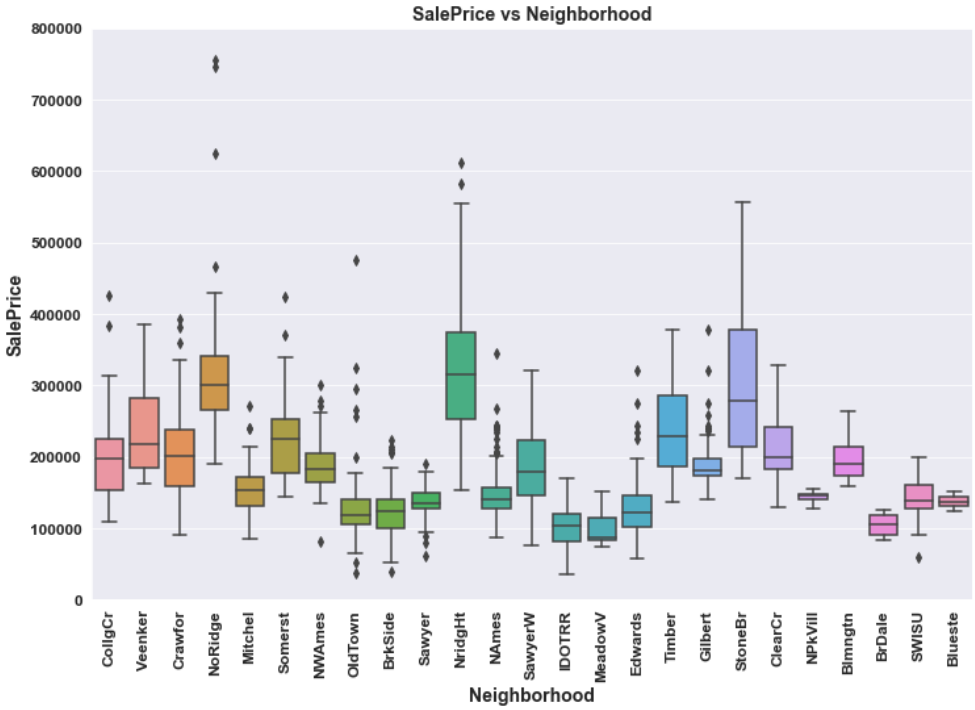
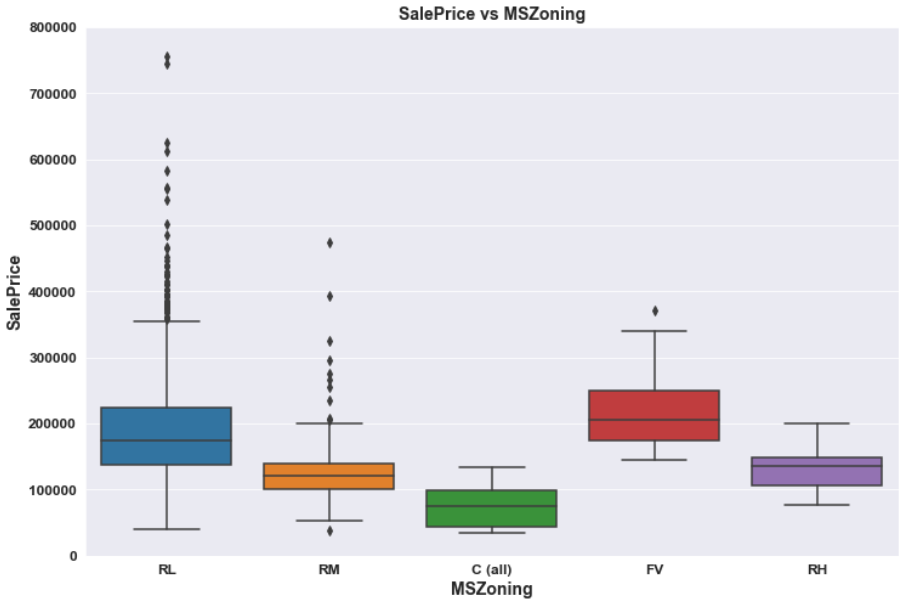
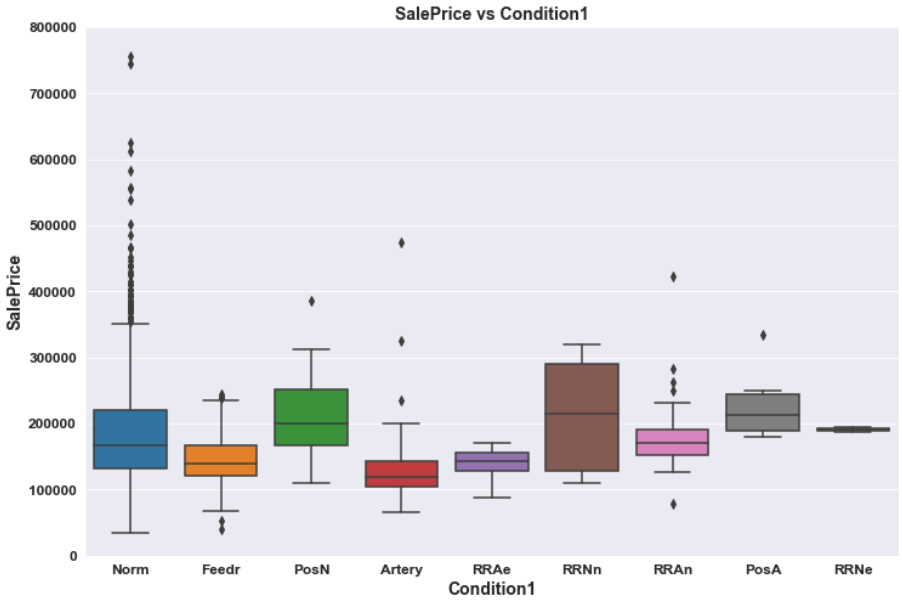


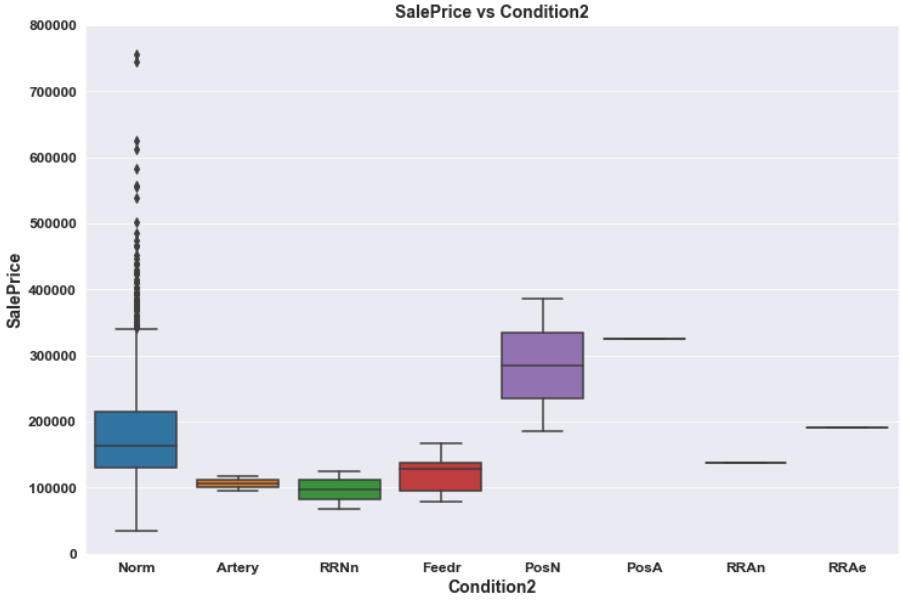
Figure 2-15: The relationship plot between SalePrice and Neighborhood.


SalePrice vs MSZoning
SalePrice
MSZoning
RL
RM
C (all)
FV
RH

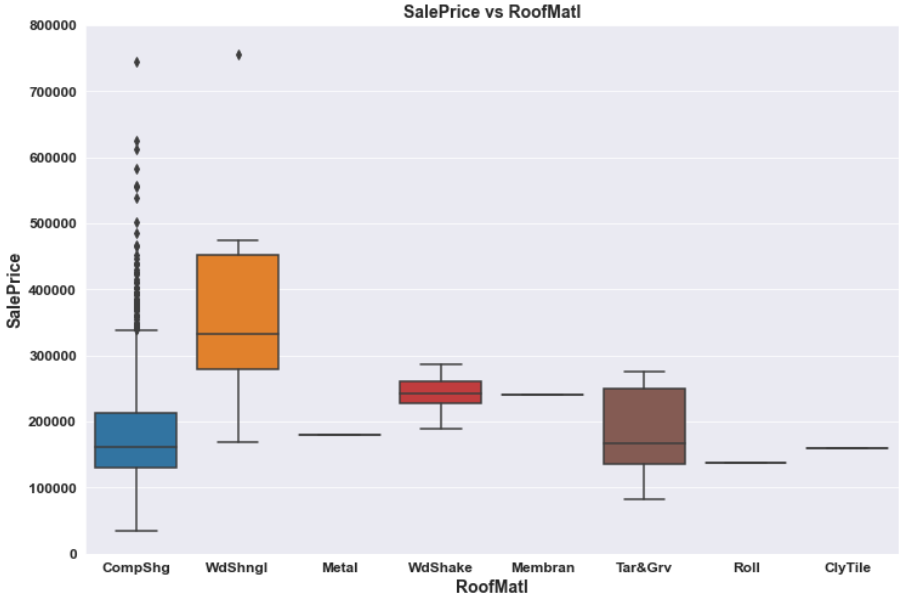
(a)


SalePrice vs Condition1
SalePrice
Condition1
Norm
Feedr
PosN
Artery
RRAe
RRNn
RRAn
PosA
RRNe

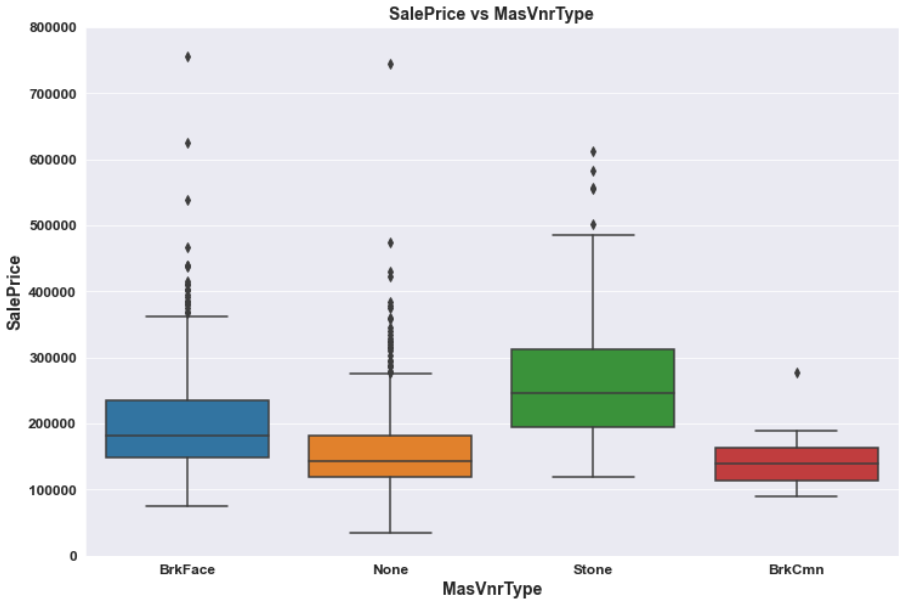
(b)


SalePrice vs Condition2
SalePrice
Condition2
Norm
Artery
RRNn
Feedr
PosN
PosA
RRAn
RRAe

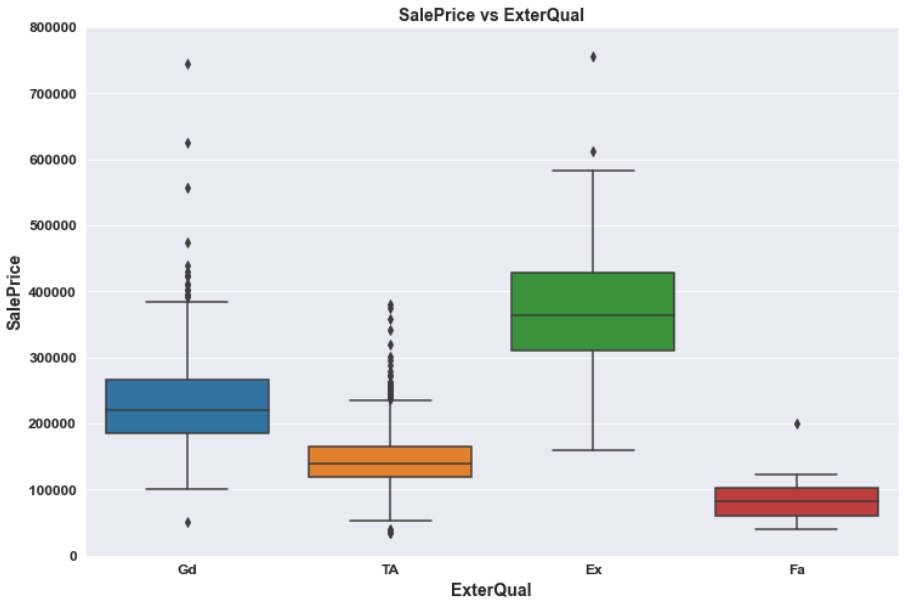
(c)


SalePrice vs RoofMatl
SalePrice
RoofMatl
CompShg
WdShngl
Metal
WdShake
Membran
Tar&Grv
Roll
ClyTile

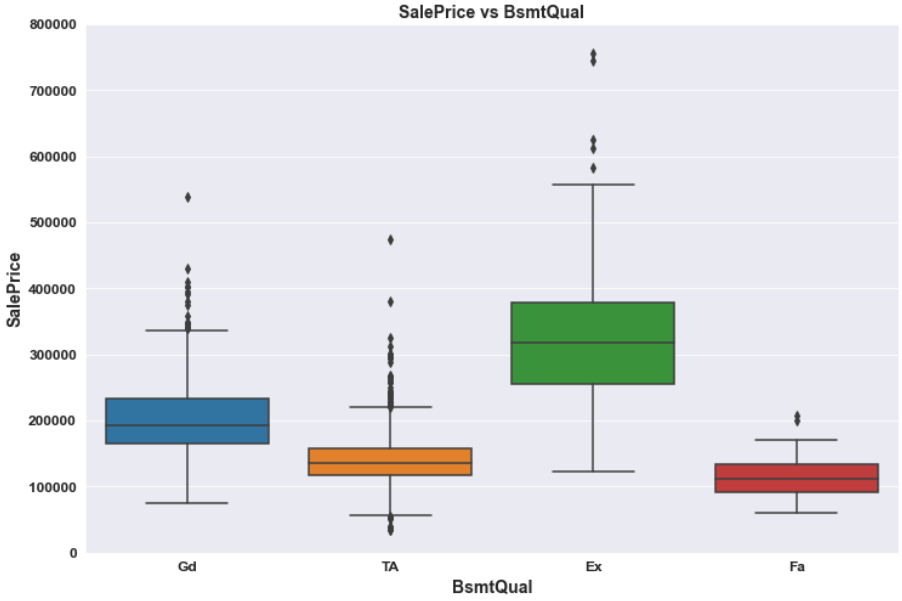
(d)


SalePrice vs MasVnrType
SalePrice
MasVnrType
BrkFace
None
Stone
BrkCmn
0
100000
200000
300000
400000
500000
600000
700000
800000

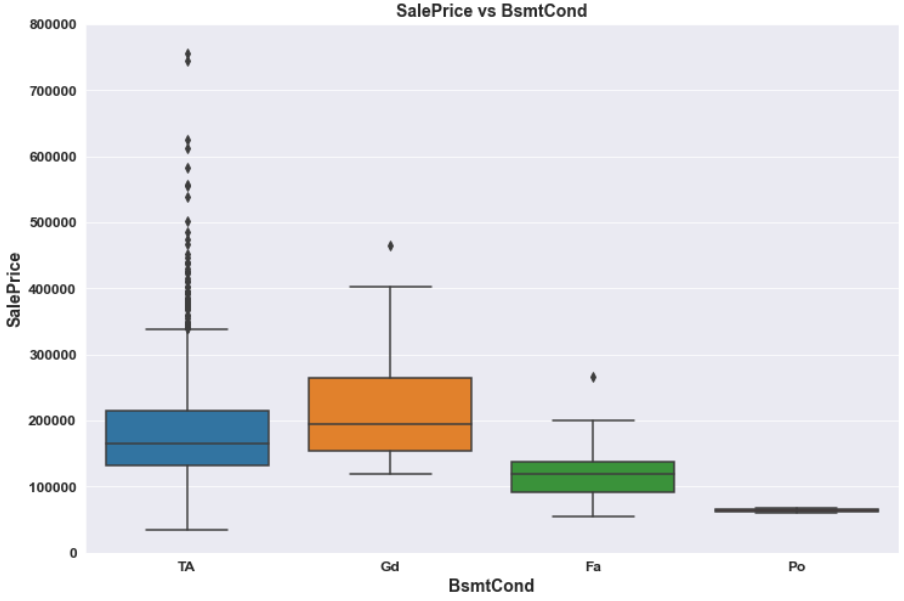
(e)


SalePrice vs ExterQual
SalePrice
ExterQual
Gd
TA
Ex
Fa
0
100000
200000
300000
400000
500000
600000
700000
800000

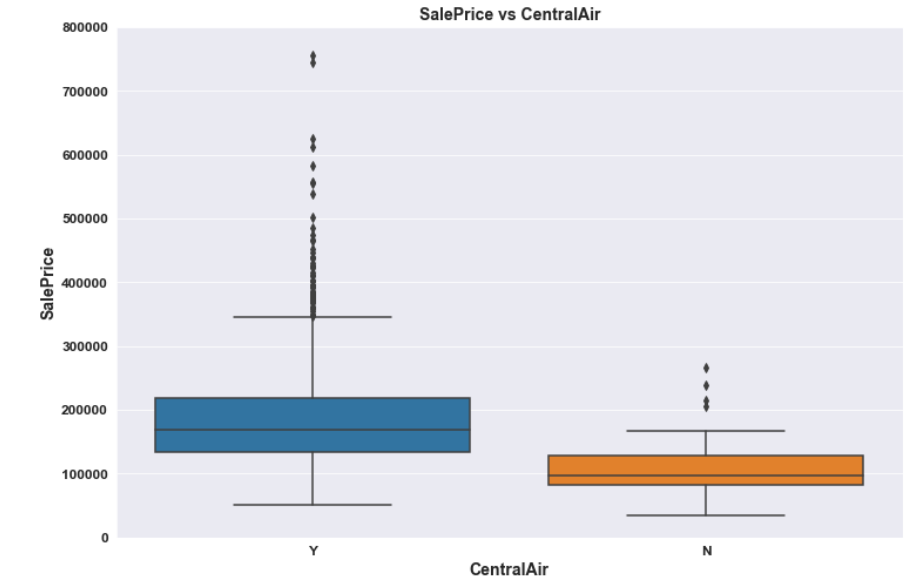
(f)


SalePrice vs BsmtQual
SalePrice
BsmtQual
Gd
TA
Ex
Fa
0
100000
200000
300000
400000
500000
600000
700000
800000

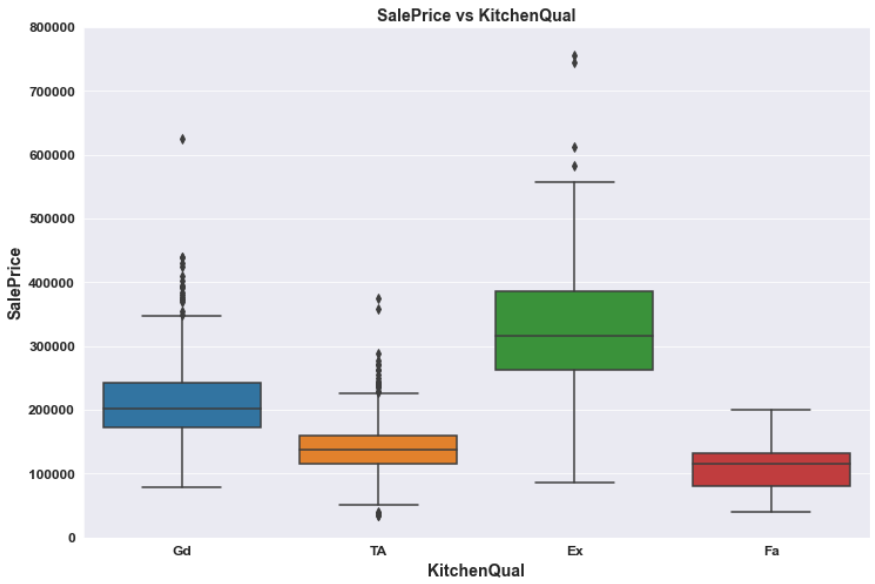
(g)


SalePrice vs BsmtCond
SalePrice
BsmtCond
TA
Gd
Fa
Po
0
100000
200000
300000
400000
500000
600000
700000
800000

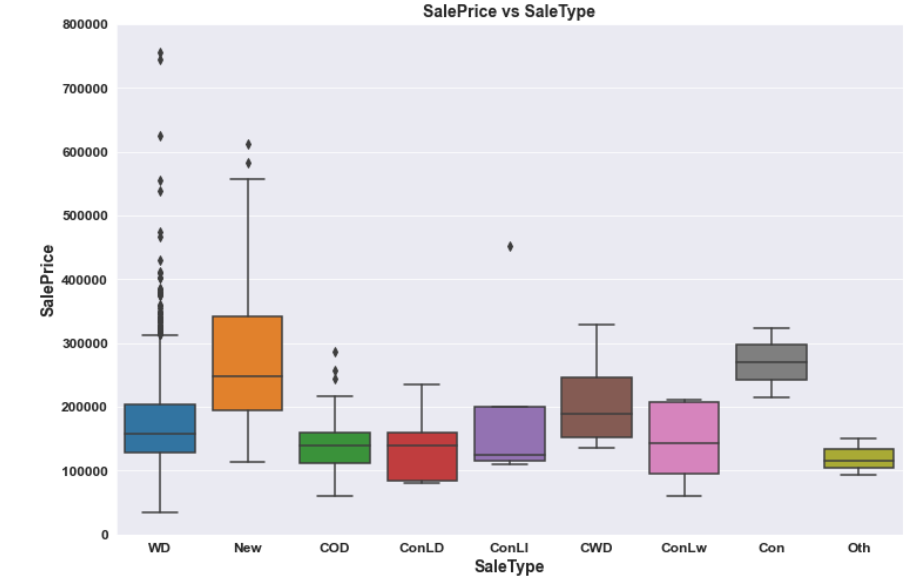
(h)



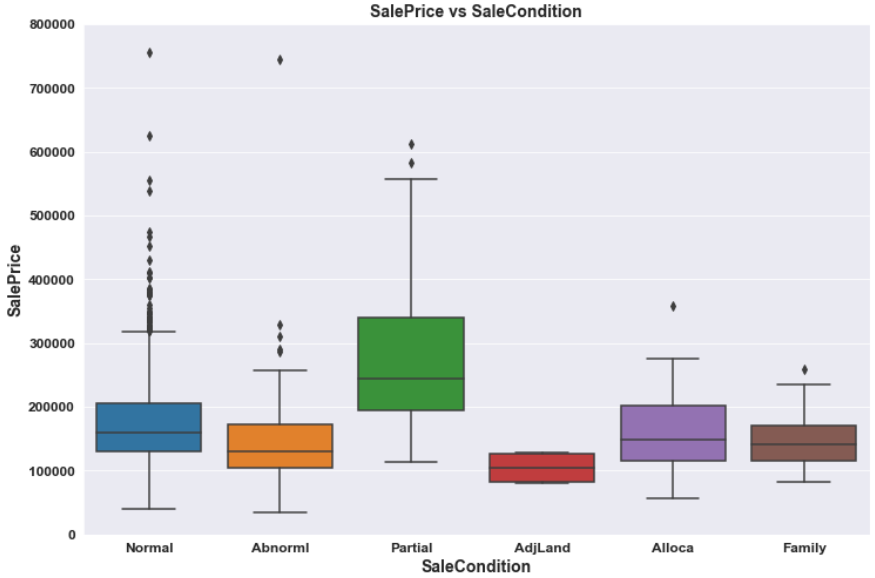
(i)



(j)



(k)



(l)

Figure 2-16: The relationship plot between SalePrice and top selected categorical variables.

2.2. Data Preprocessing

2.2.1. Outlier Treatment

Let's check the values of our target variable (SalePrice) together with the other numeric attributes. Figure 2-17 shows the scatter plot between SalePrice and TotalBsmtSF. We can see that there is one house with a huge basement and a cheap price, this is an outlier. This house would cause problems in modeling because, except for this one house, we can see a clear linear relationship between the size of a basement and the price of a house. So, we need to remove this outlier from the dataset.

We used two techniques to remove outliers, the first one is based on visual representation of variables of which we need to remove the outliers from and the target variable. Then, we can identify the outlier location and use Panadas to remove it. The second way is to use Interquartile range (IQR) method, which is stricter that the visual way.

For outliers shown in Figure 2-17, we will remove all observations that have more than 5,000 square feet of basement and a price lower than $300,000, Figure 2-18 shows the scatter plot between SalePrice and TotalBsmtSF after removing the outliers using visual methods. We can see that the relationship in the picture is much nicer now.

Figure 2-19 shows the scatter plot between SalePrice and TotalBsmtSF after outliers removal using IQR method. It is clear that IQR method is stricter that visual one because it removed low and high outliers.

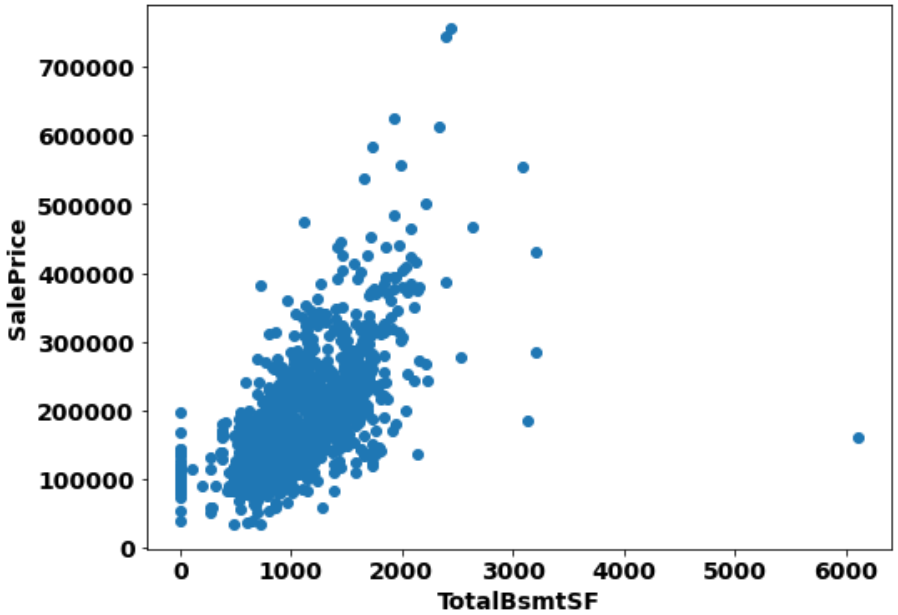


Figure 2-17: The scatter plot between SalePrice and TotalBsmtSF.

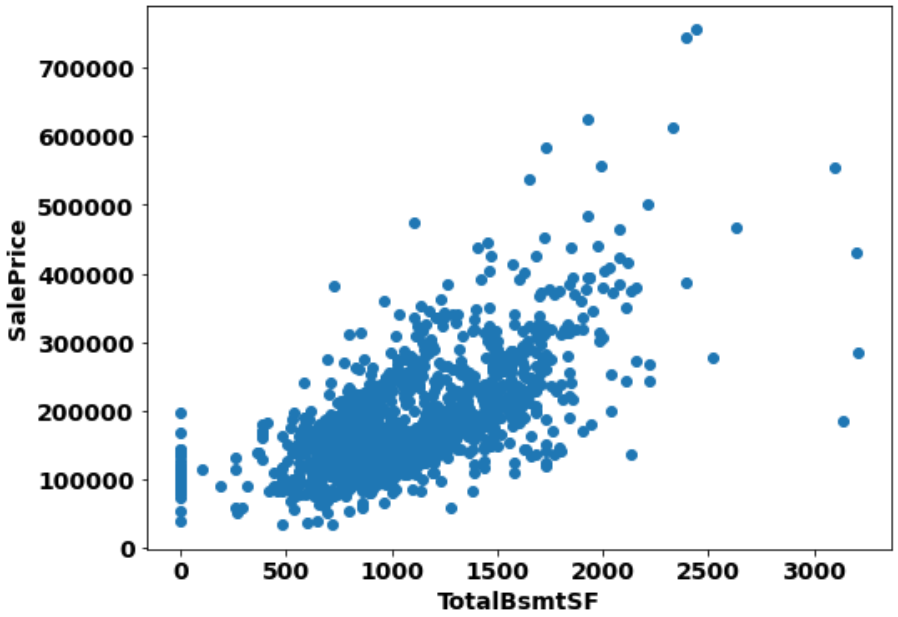


Figure 2-18: The scatter plot between SalePrice and TotalBsmtSF after outliers removal using visual method.

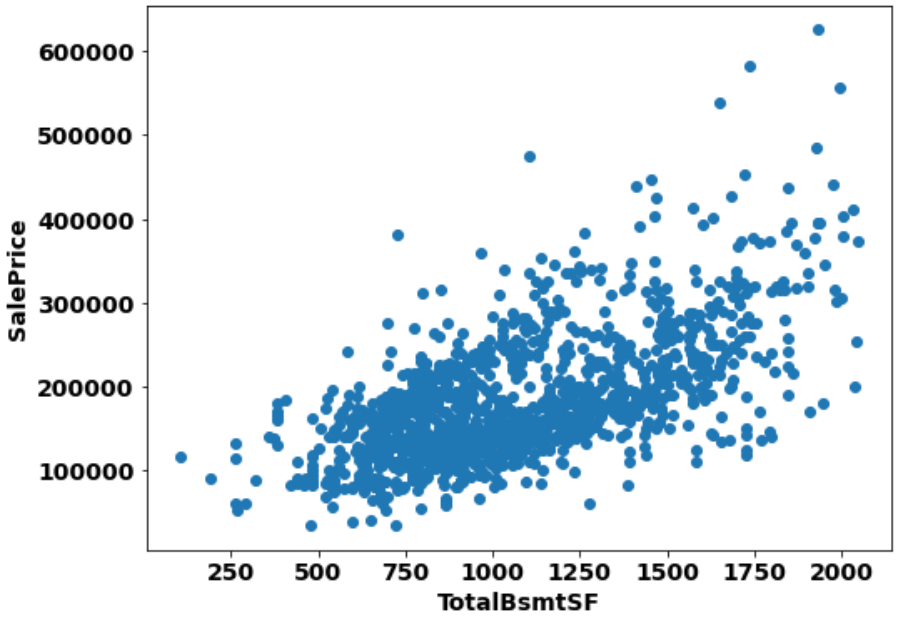


Figure 2-19: The scatter plot between SalePrice and TotalBsmtSF after outliers removal using IQR method.

Now let's take another variable and apply the two methods to remove its outliers. Figure 2-20 shows the scatter plot between SalePrice and GrLivArea. If we look only at GrLivArea there are no outliers because the largest area is quite close to the second and the third largest. However, if we take a look at SalePrice together with GrLivArea, we can see that the price of the largest

house is really small, which will again cause problems in the modeling step. Therefore, we should remove this observation as well.

Figure 2-21 shows the scatter plot between SalePrice and GrLivArea after outliers removal using Visual method whereas Figure 2-22 shows the removing outliers using IQR method. We followed the same way to check the outliers and remove them if available in other variables as a step in data preprocessing. So, we provide only two examples here to show how we treated the outliers in the dataset and make the report short instead of filling it with repeated examples.

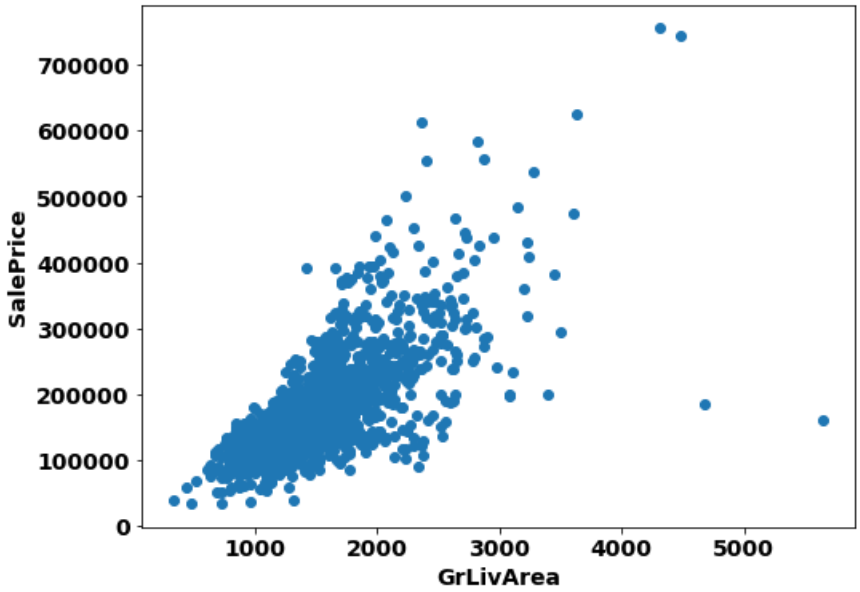


Figure 2-20: The scatter plot between SalePrice and GrLivArea.

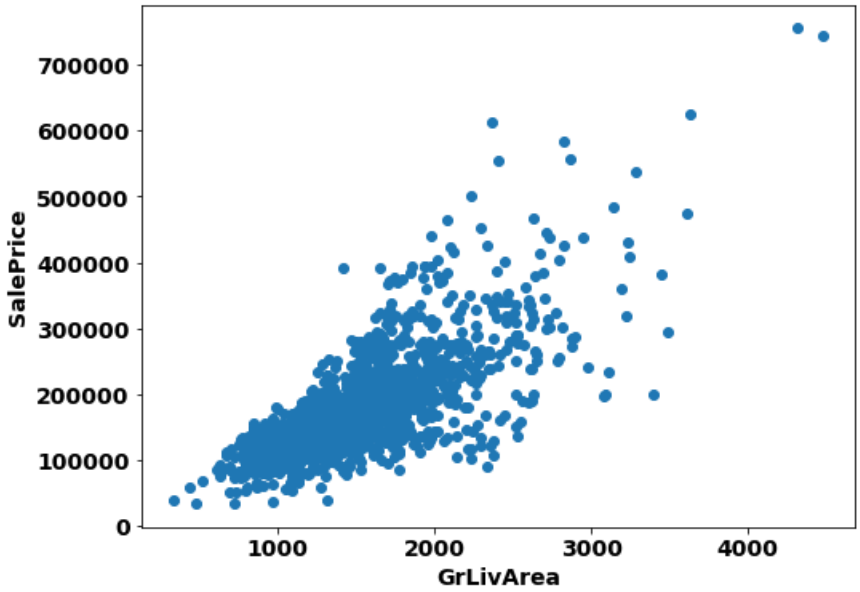


Figure 2-21: The scatter plot between SalePrice and GrLivArea after outliers removal using Visual method.

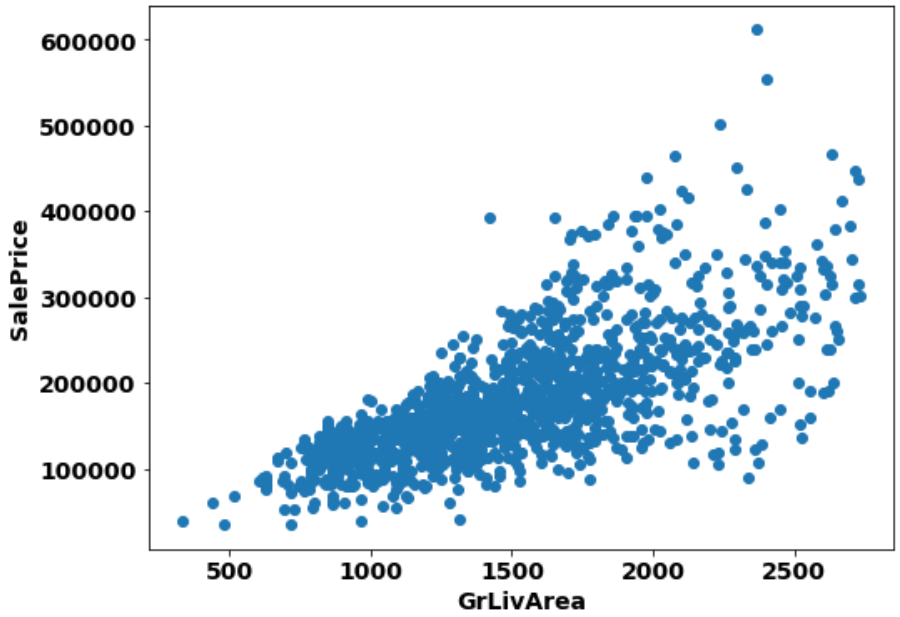


Figure 2-22: The scatter plot between SalePrice and GrLivArea after outliers removal using IQR method.

2.2.2. Null Values Replacement

As a first step, we will check whether we have columns with missing values. *Table 2-6* shows number of missing values and their percentages for each column has missing values. Figure 2-23 shows the percentages of missing values.

The first strategy is to drop the five columns with the biggest percentage of null values and check the missing values in dataset to make sure that the highest percentages have been dropped. Figure 2-24 shows the remaining percentages of missing vales after dropping the highest percentages.

Now we can work on the remaining Null values to replace them with appropriate values. We replaced the missing values of LotFrontage with the mean. For GarageYrBlt variable, if the house has a garage and the year is missing, we assume it's the minimum value available. If the veneer area (MasVnrArea) is missing, we assume it's 0.

With the help of the data documentation we have, we can figure out that the missing values in Garage and Basement variables mean no garage and no basement respectively. Therefore, we will replace the missing values with "None"

The information about Electrical and MasVnrType is missing in the documentation. Since we are dealing with categorical variables, we will create a new category for a missing value called ‘Empty’ and replace the missing values of Electrical and MasVnrType with it. After replacing the missing values, we checked the dataset again to ensure that there are zero missing values.

Table 2-6. Missing values and their percentages.

	Missing Values	Missing Values %
PoolQC	1452	99.588477
MiscFeature	1404	96.296296
Alley	1367	93.758573
Fence	1177	80.727023
FireplaceQu	690	47.325103
LotFrontage	259	17.764060
GarageCond	81	5.555556
GarageType	81	5.555556
GarageYrBlt	81	5.555556
GarageFinish	81	5.555556
GarageQual	81	5.555556
BsmtExposure	38	2.606310
BsmtFinType2	38	2.606310
BsmtFinType1	37	2.537723
BsmtCond	37	2.537723
BsmtQual	37	2.537723
MasVnrArea	8	0.548697
MasVnrType	8	0.548697
Electrical	1	0.068587

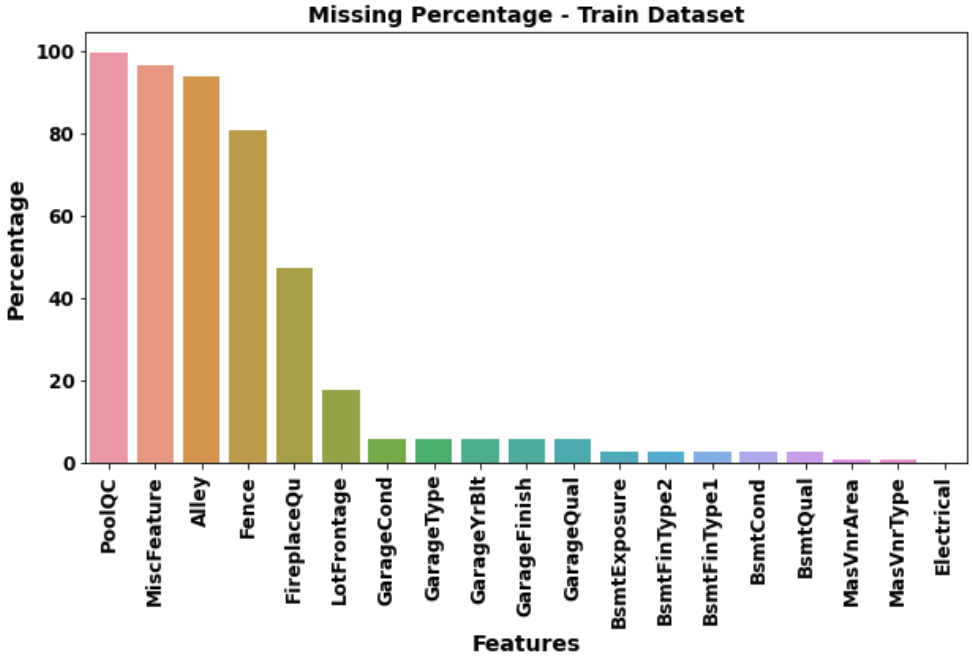


Figure 2-23: The percentages of missing vales

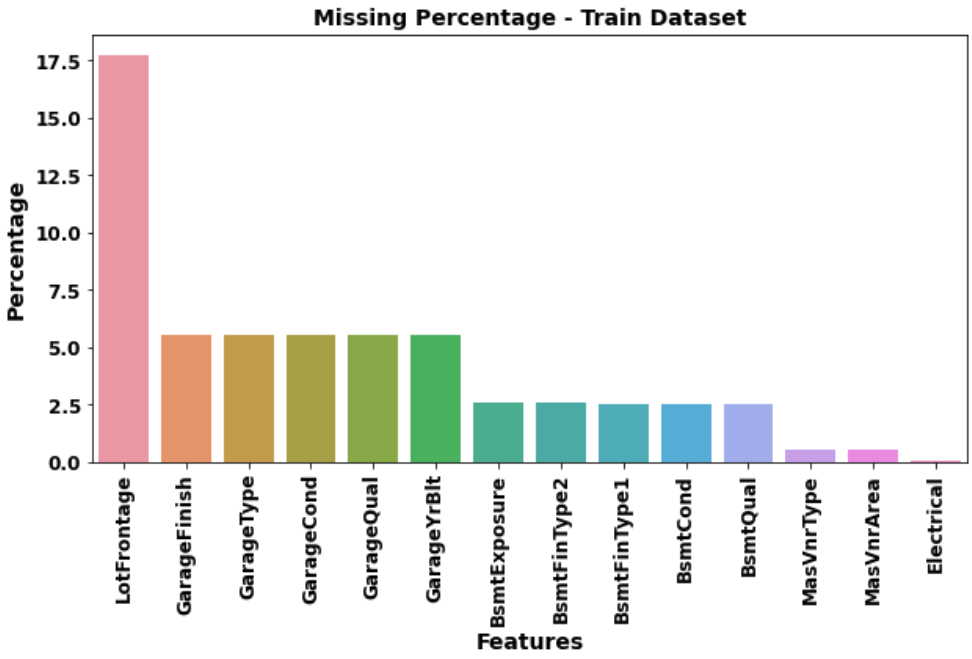


Figure 2-24: The percentages of missing vales after dropping the highest percentages.

2.2.3. Variable Transformation

Log transformation has been used to normalize the data that does not follow the normal distribution. Before performing the log transform, let's plot the distribution of our target (SalePrice) first and compare it to normalized one. Figure 2-25 and Figure 2-26 show the distribution and probability plots for SalePrice variable before using log transformation. We can observe that the distribution has a long tail and most of the density of sale's price lies between 100k and 200k. It means that most of the house are normally distributed but a couple of houses have a higher than normal value, resulting in slightly deviation from a normal distribution with skewness of 1.88.

Figure 2-27 and Figure 2-28 show the distribution and probability plots for SalePrice variable after applying log transformation. It is clear that the logarithmic transformation of the SalePrice is more normal and skewness has been reduced to 0.12. We applied log transformation for other numerical variables that do not follow the normal distribution.

Furthermore, the StandardScaler has been used from sklearn to scale variables that do not highly deviate from normal distribution. Scaling is important for some algorithms that require to have values with the same scale, for example between 0 and 1.

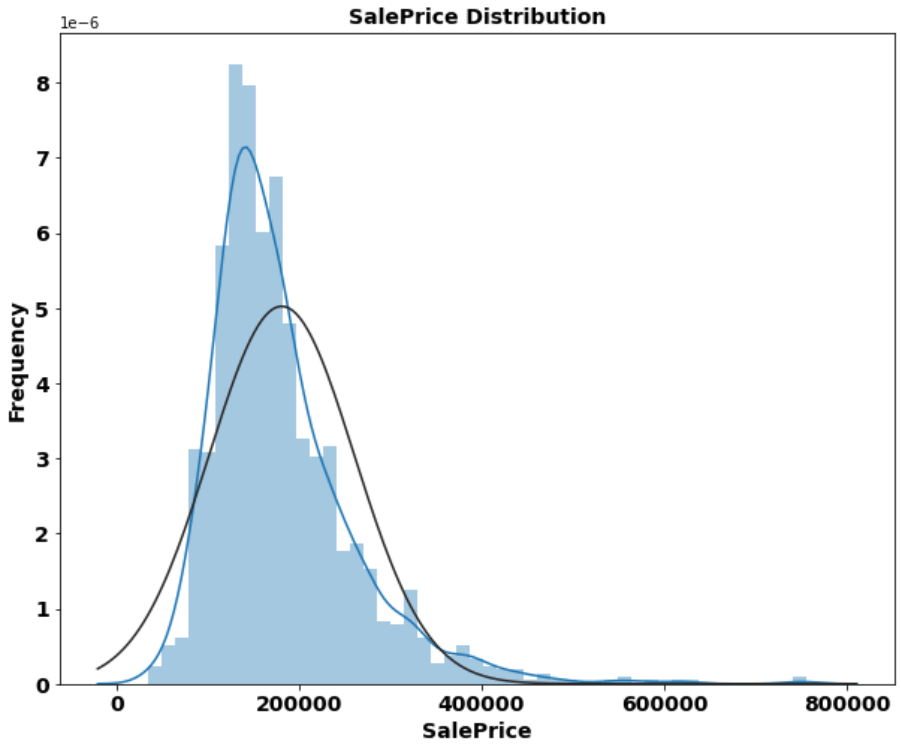


Figure 2-25: SalePrice (Target Variable) before log transformation.

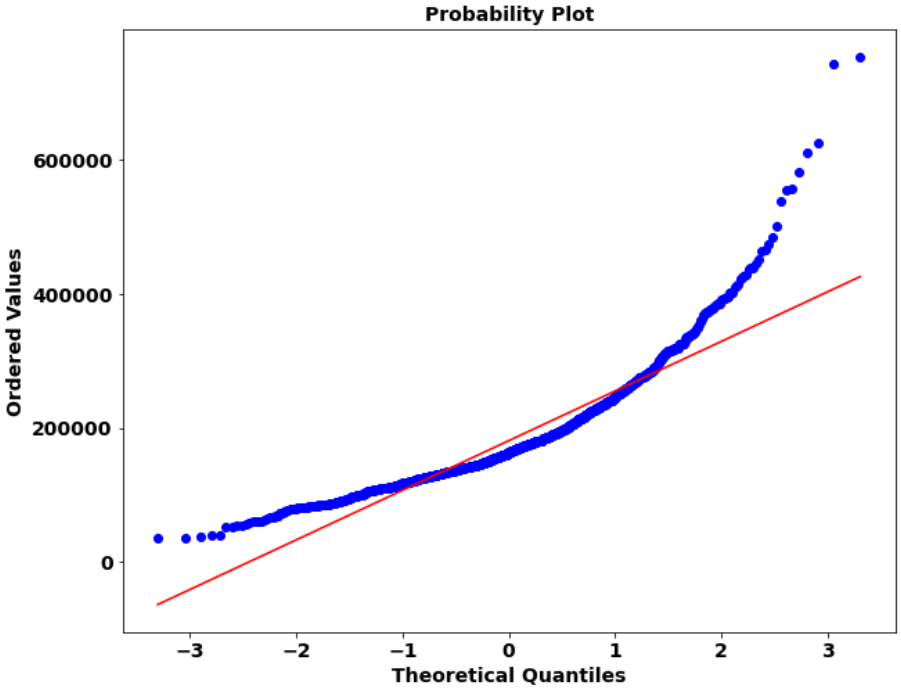


Figure 2-26: The probability plot for SalePrice variable before log transformation.

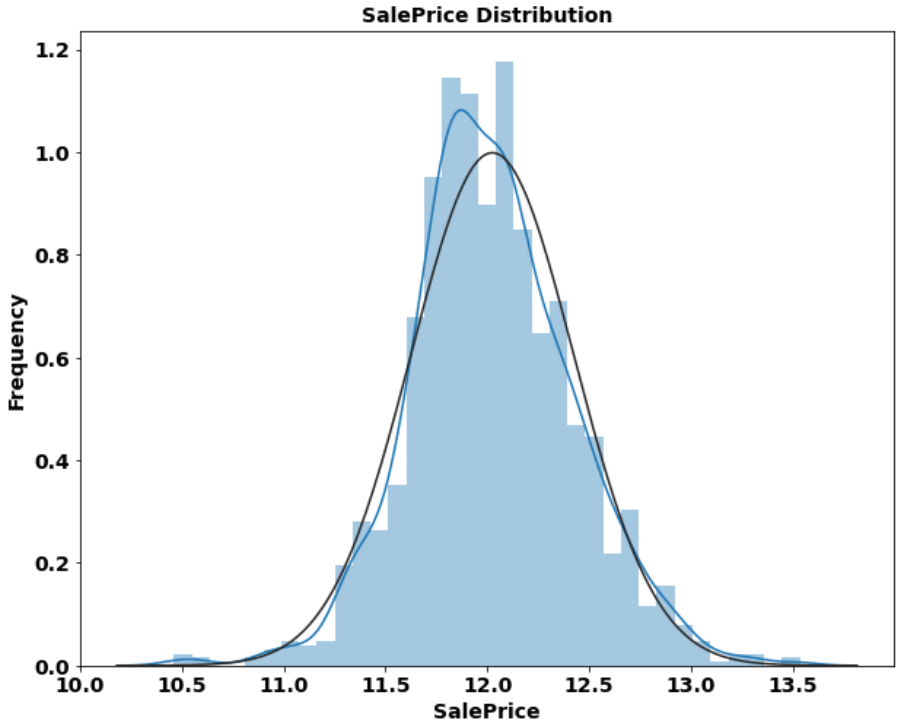


Figure 2-27: SalePrice (Target Variable) after log transformation.

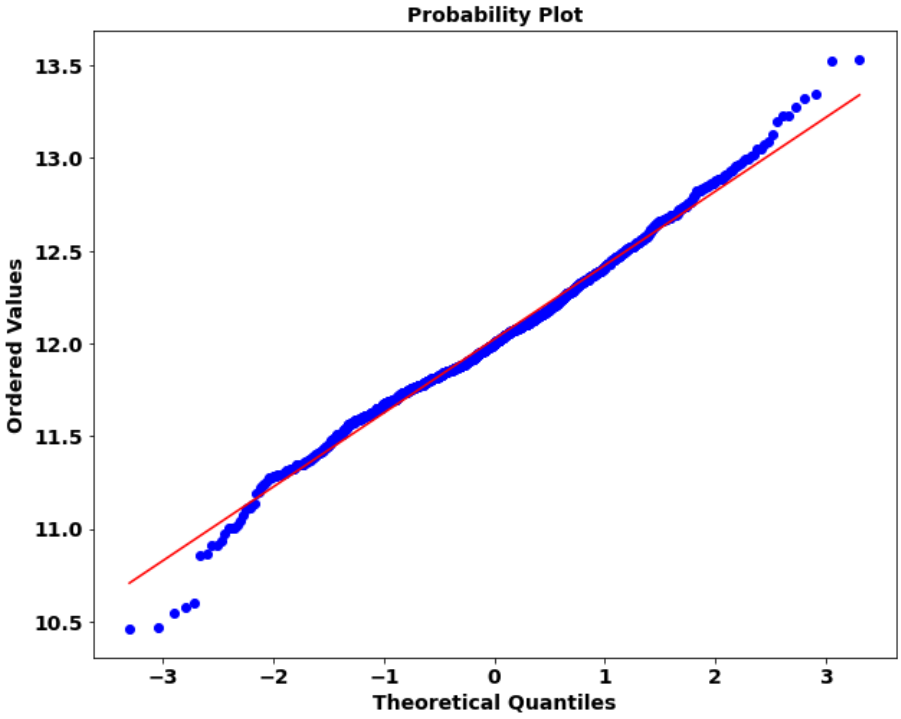


Figure 2-28: The probability plot for SalePrice variable after log transformation.

2.2.4. Categorical Variables Encoding

In our dataset, we have two types of categorical variables including nominal and ordinal variables. The difference is that with an ordinal variable, we can order the categories by importance/value/score, whereas a nominal variable has no intrinsic ordering to its categories. Therefore, we can transform the ordinal variables into numbers and create numeric variables out of them. We will use the help of our documentation to understand the sorting. For nominal variables, we will use dummy variables to encode them. These dummy variables will be created with one-hot encoding and each attribute will have a value of either 0 or 1, representing the presence or absence of that attribute.

2.3. Features Engineering

2.3.1. Combination of the Existing Features

An example of combining existing variables is to sum two variables. We have two variables 1stFlrSF and 2ndFlrSF but we don't have the total square footage. So, we can create a new feature which represents the sum of these two called 1stFlr_2ndFlrSF as shown in Figure 2-29. We can see that there is a significant relationship between the new variable and our target (SalePrice).

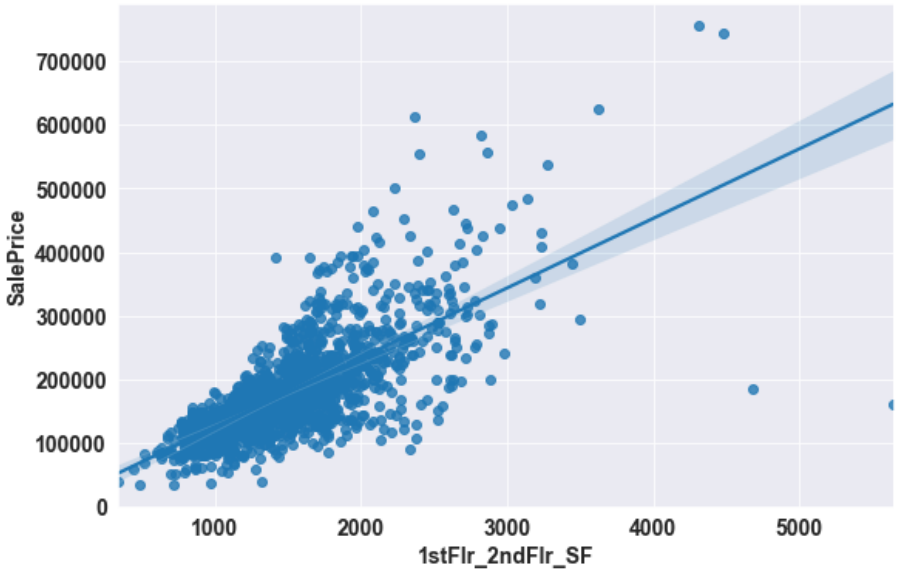


Figure 2-29: Creating new feature by summing two variables (1stFlrSF and 2ndFlrSF).

Another example is to create OverallGrade feature by multiplying two variables (OverallQual * OverallCond). The result of the new feature is shown in Figure 2-30.

In this project, we created new features based on ratios between two variables for example, creating two new features expressing important ratios using mathematical transformation as follows: LivLotRatio: the ratio of GrLivArea to LotArea. Spaciousness: the sum of FirstFlrSF and SecondFlrSF divided by TotRmsAbvGrd.

Also, we created new features based on counting some important features, for example:

- Creating a new feature called PorchTypes that describes how many kinds of outdoor areas a dwelling has. We will count how many of WoodDeckSF, OpenPorchSF, EnclosedPorch, Threeseasonporch, and ScreenPorch are greater than 0.0.
- Creating another new feature TotalHalfBath that contains the sum of half-bathrooms within the property.
- Creating new feature called TotalRoom that sums up the total number of rooms (including full and half bathrooms) in each property.
- Grouped Transform, the value of a home often depends on how it compares to typical homes in its neighborhood. Therefore, we created a new feature called MedNhbdArea that describes the median of GrLivArea grouped on Neighborhood.

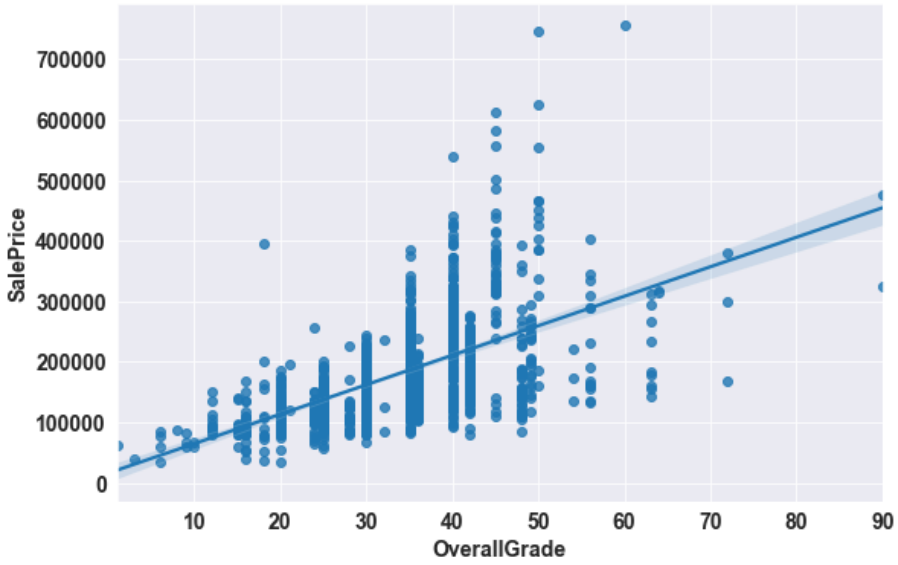


Figure 2-30: Creating new feature by multiplying two variables (OverallQual * OverallCond).

2.3.2. Simplification of the Existing Features

An example of simplification features is to simplify GarageQual which is an ordinal feature. Let's check the distribution per category of GarageQual as shown in Figure 2-31. We can see that there are categories for which the SalePrice is similar. If we move from the category 0 to 1 or 2, there is no change in SalePrice. Therefore, we can merge these categories into one as shown in Figure 2-32. We can see that the simplified feature has a "nicer" relationship. If we go from 0 to 1 or 2, the average SalePrice increases.

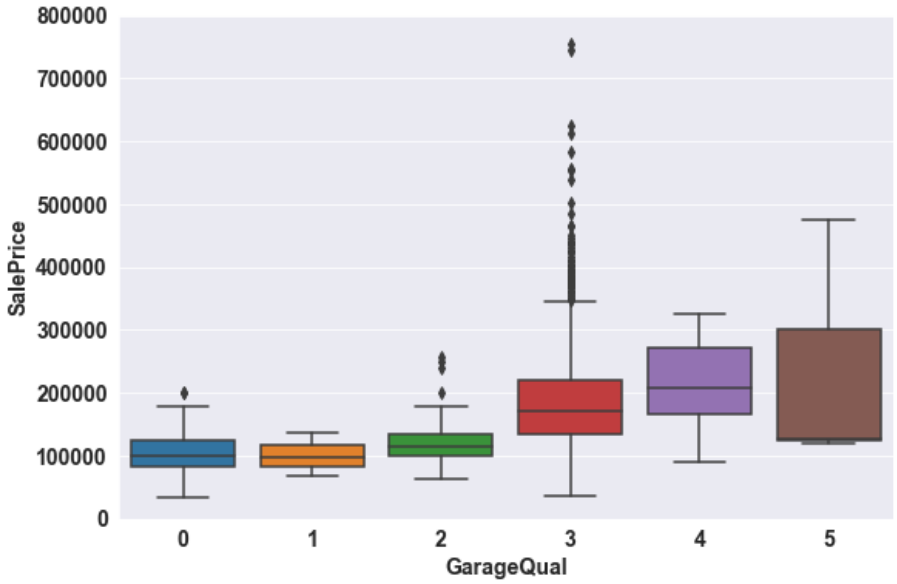


Figure 2-31: The distribution per category of GarageQual variable before simplification.

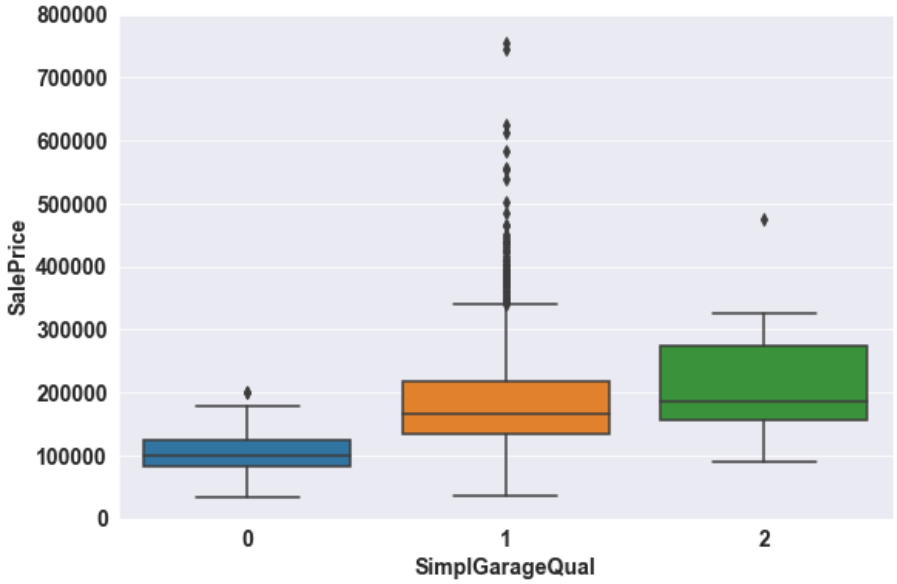


Figure 2-32: The distribution per category of GarageQual variable after simplification.

2.3.3. Filter Feature Selection

The criteria used for filter feature selectors is based on the following:

- Removing features with small variance, removing the columns with very little variance. Small variance equals small predictive power because all houses have very similar values.
- Removing correlated features, the goal of this part is to remove one feature from each highly correlated pair. This can be done in 3 steps:
    - Calculate a correlation matrix

- Get pairs of highly correlated features
- Remove correlated columns

- Forward Regression

  We have removed the features with no information and correlated features so far. The last thing we will do before modeling is to select the k-best features in terms of the relationship with the target variable. We will use the forward wrapper method for that. After performing filter feature selection, we came up with10 features which should be pretty good predictors of our target variable, SalePrice as shown in *Table 2-7*.

Table 2-7. The 10 top features obtained from filter feature selection.

	OverallQual	YearBuilt	ExterQual	BsmtQual	TotalBsmtSF	GrLivArea	FullBath	KitchenQual	GarageCars	OverallGrade
0	7.0	2003.0	4.0	4.0	856.0	1710.0	2.0	4.0	2.0	35.0
1	6.0	1976.0	3.0	4.0	1262.0	1262.0	2.0	3.0	2.0	48.0
2	7.0	2001.0	4.0	4.0	920.0	1786.0	2.0	4.0	2.0	35.0
3	7.0	1915.0	3.0	3.0	756.0	1717.0	1.0	4.0	3.0	35.0
4	8.0	2000.0	4.0	4.0	1145.0	2198.0	2.0	4.0	3.0	40.0

2.3.4. Mutual Information (MI) Scores

Mutual information score is one of filter-based feature selection methods. We have several features that are highly informative and several that don't seem to be informative at all. Therefore, we will focus our efforts on the top scoring features. Figure 2-33 shows the mutual information scores of our dataset. Training on uninformative features can lead to overfitting as well, so features with 0.0 MI scores have been dropped from the dataset.

2.3.5. Interaction Effects

Interaction effects occur when the effect of an independent variable depends on the value of another independent variable, for example, checking the interaction effect between GrLivArea and BldgType as shown in Figure 2-34. The trend lines being significantly different from one category to the next indicates an interaction effect between GrLivArea and BldgType that relates to a home's SalePrice. Several other detected interaction effects between categorical and numerical variables have been performed too.

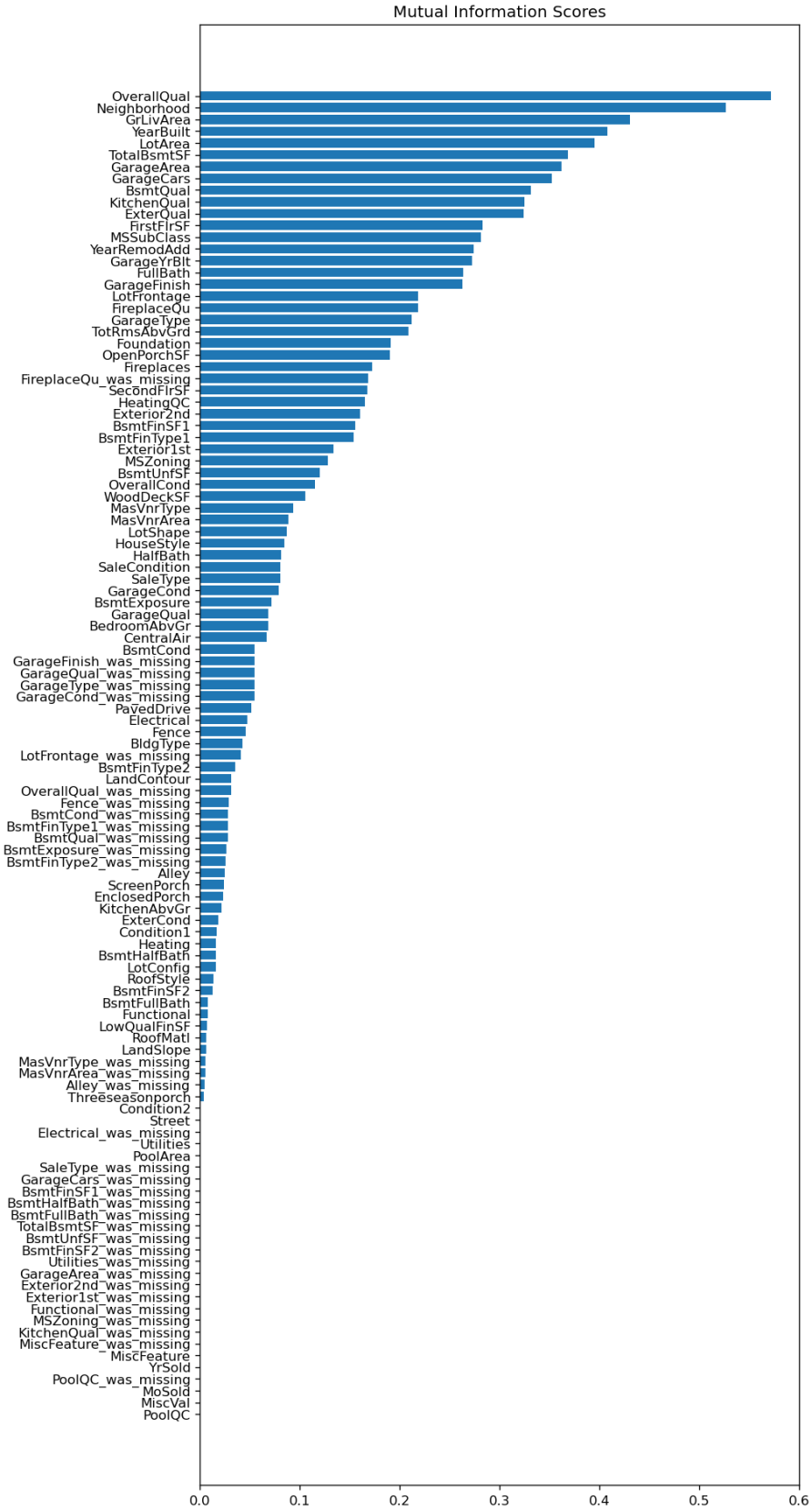


Figure 2-33: The mutual information scores.

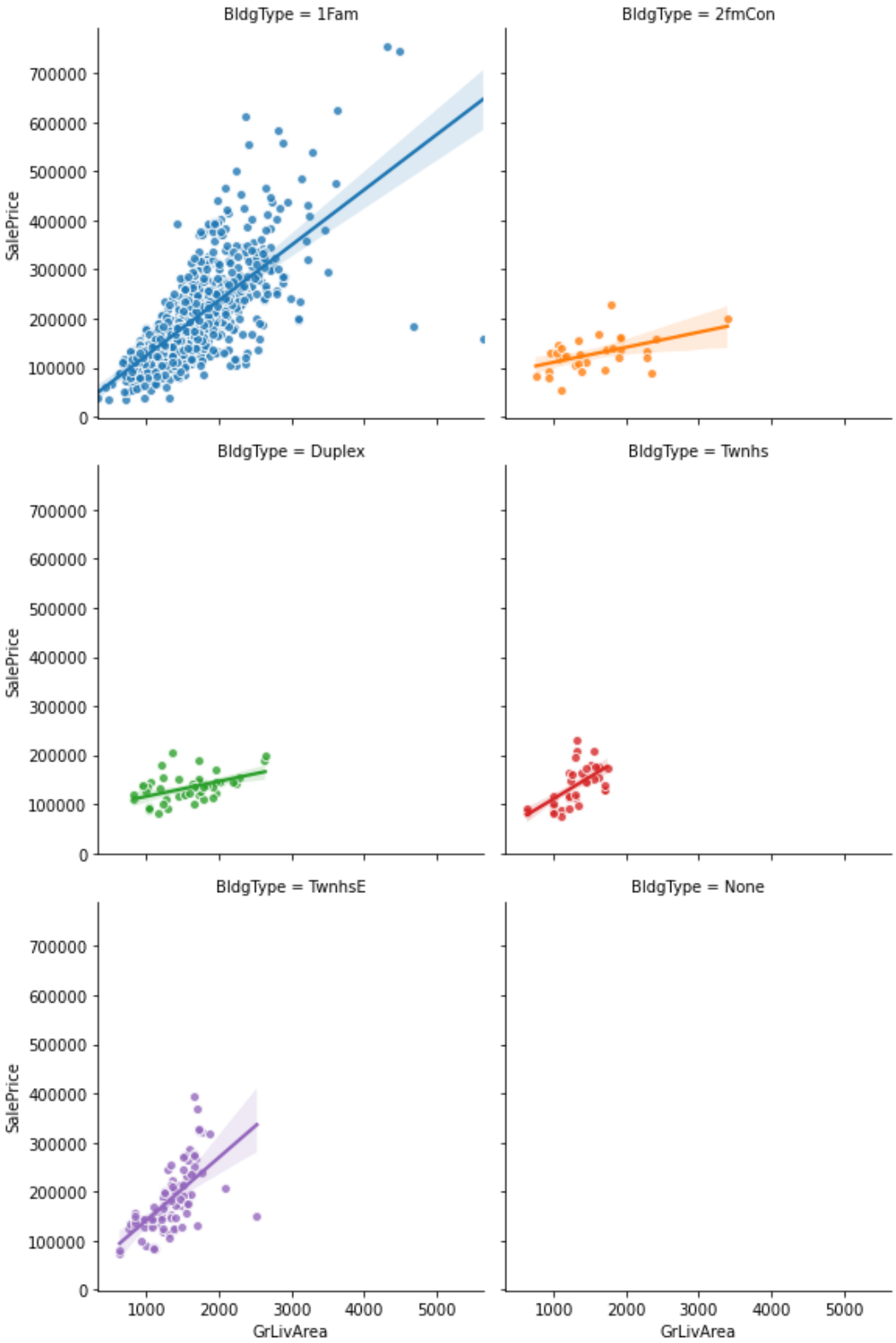


Figure 2-34: The interaction effect between GrLivArea and BldgType.

2.3.6. Creating New Feature Using Clustering

The clustering approach can be used for feature selection. The formation of clusters reduces the dimensionality and helps in selection of the relevant features for the target class by using cluster labels as features. We used k-means clustering and hierarchical clustering.

The first step in k-means clustering is to select the number of clusters first. We used the elbow method to find the appropriate number of clusters as shown in Figure 2-35. The elbow method runs k-means clustering on the dataset for a range of values for k (1-30) and then for each value of k computes an average score for all clusters. By default, the distortion score is computed, the sum of square distances from each point to its assigned center. We selected k = 5 at the corresponding small inflection since the distortion does not have high change after this point.

The k-means clustering model has been trained with k = 5, the clusters have been visualized in Figure 2-36. K-means model resulted with good clusters with average silhouette score of 0.474 for 5 clusters. Figure 2-37 shows the silhouette score visualization for k-means clustering model. The distance of the observations to each cluster has been calculated to be used as another new feature.

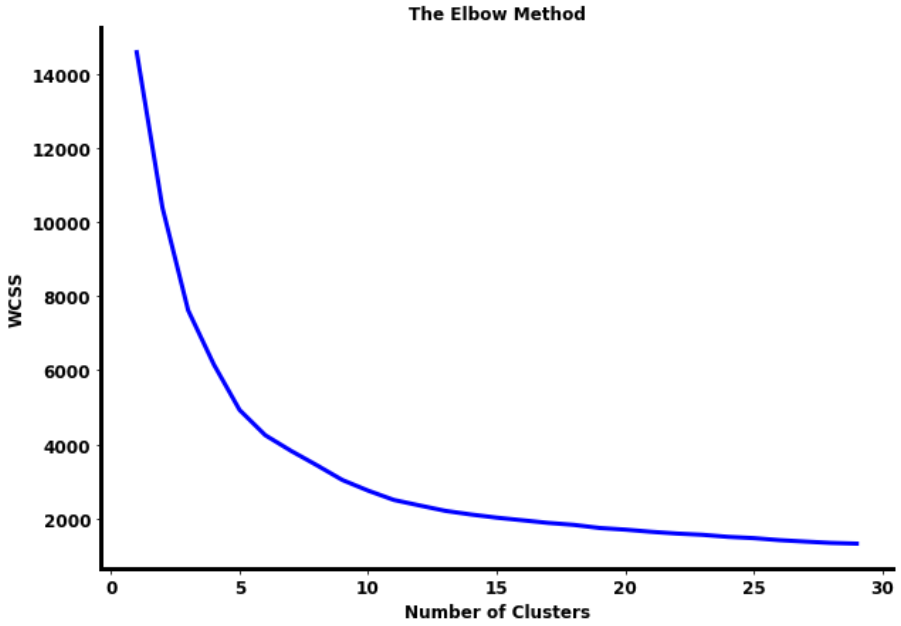


Figure 2-35: The elbow method to find the appropriate number of clusters.

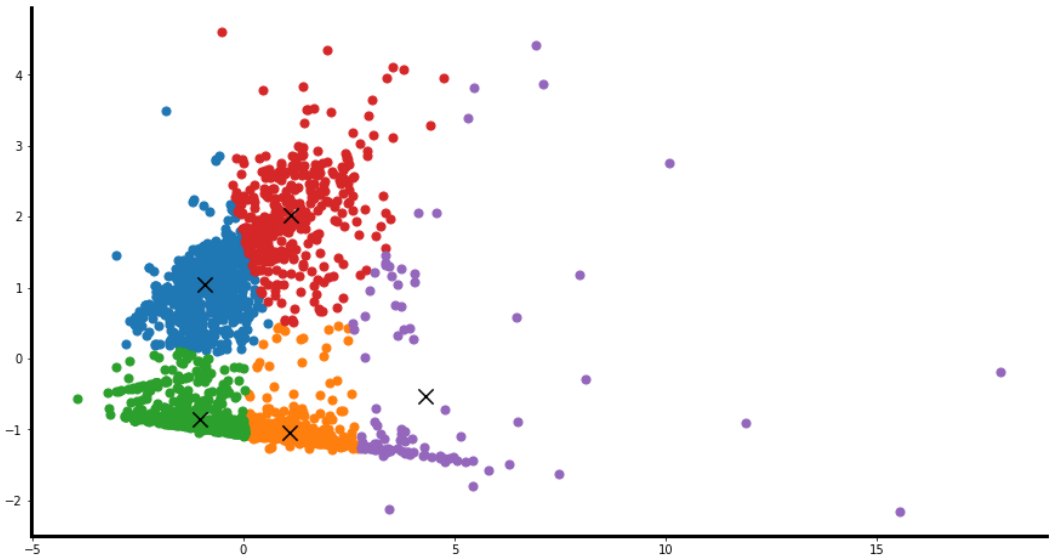


Figure 2-36: The clusters visualization of k-means model.

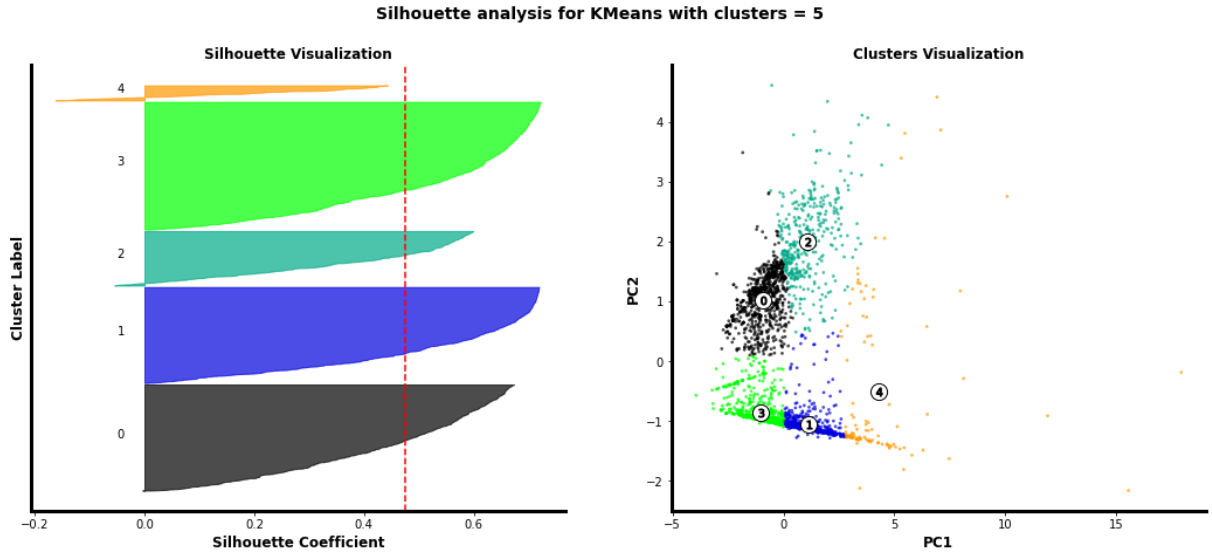


Figure 2-37: The silhouette score visualization for k-means with k = 5.

Moreover, we used hierarchical clustering (Agglomerative algorithm) as well. We plotted a dendrogram first to find out the number of clusters as shown Figure 2-38. To find the number of clusters, we added a horizontal line across the dendrogram that cuts the long vertical lines and not the small vertical lines (clusters). By counting the total vertical lines that this horizontal line cuts, we can find the number of clusters to pass to the hierarchical clustering model. From the dendrogram, we can find that the number of clusters = 6. After running the hierarchical clustering (Agglomerative algorithm) with n = 5, the resulted clusters are plotted as shown in Figure 2-39. The silhouette score is 0.431 which is almost close to silhouette score of k-means clustering.

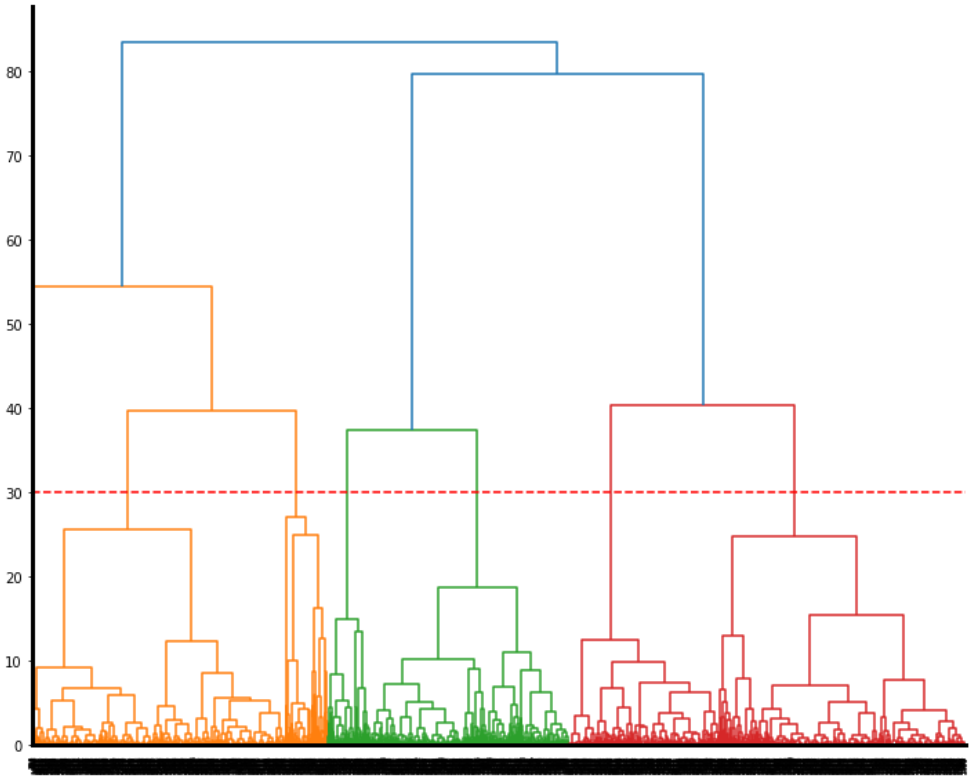


Figure 2-38: The dendrogram plot for hierarchical clustering.

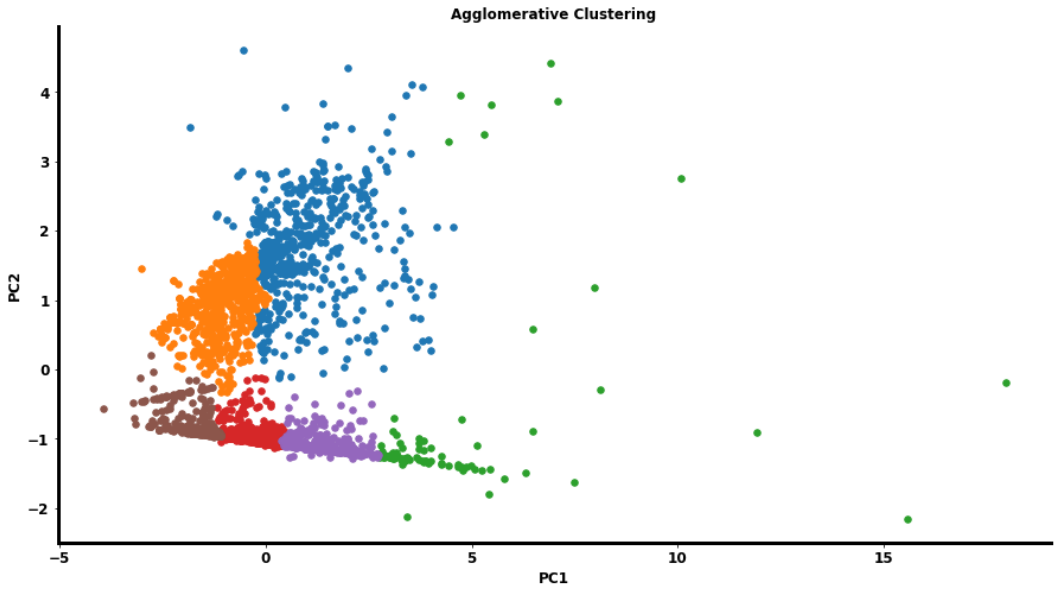


Figure 2-39: The clusters visualization of hierarchical clustering model.

2.3.7. Principal Component Analysis (PCA)

PCA is another unsupervised learning method to create more new features. It is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets, by identifying important relationships in dataset, transforms the existing data based on these relationships, and then quantifies the importance of these relationships, keeping the most important relationships and drop the others. When reducing dimensionality through PCA, the most important information will be kept by selecting the principal components that explain most of the relationships among the features. More information will be presented in results section.

First, we plotted a matrix dataset as a hierarchically clustered heatmap to order dataset by similarity as shown in Figure 2-40. This reorganizes the data for the rows and columns and displays similar content next to one another for even more depth of understanding the dataset.

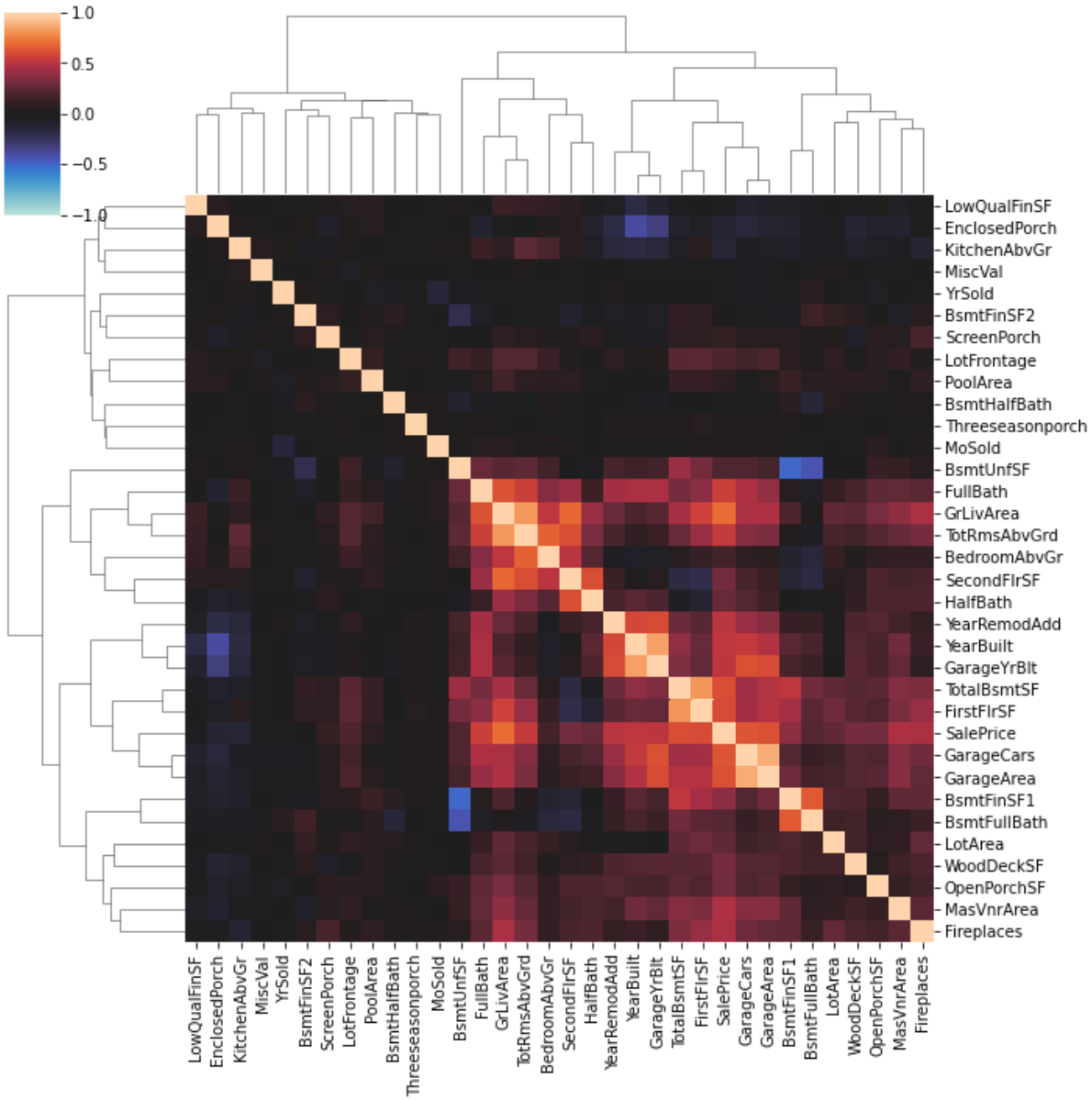


Figure 2-40: The hierarchically clustered heatmap

The PCA algorithm gives us *loadings* which describe each *component* of variation, and also the components which were the transformed datapoints. The loadings can suggest features to create. Additionally, we can use the components as features directly. After performing the PCA, the explained variance and cumulative variance based on components from PCA have been plotted as shown in Figure 2-41. The goal is to use the results of PCA to discover one or more new features that could improve the performance of our models, we can use PCA loadings to create features.

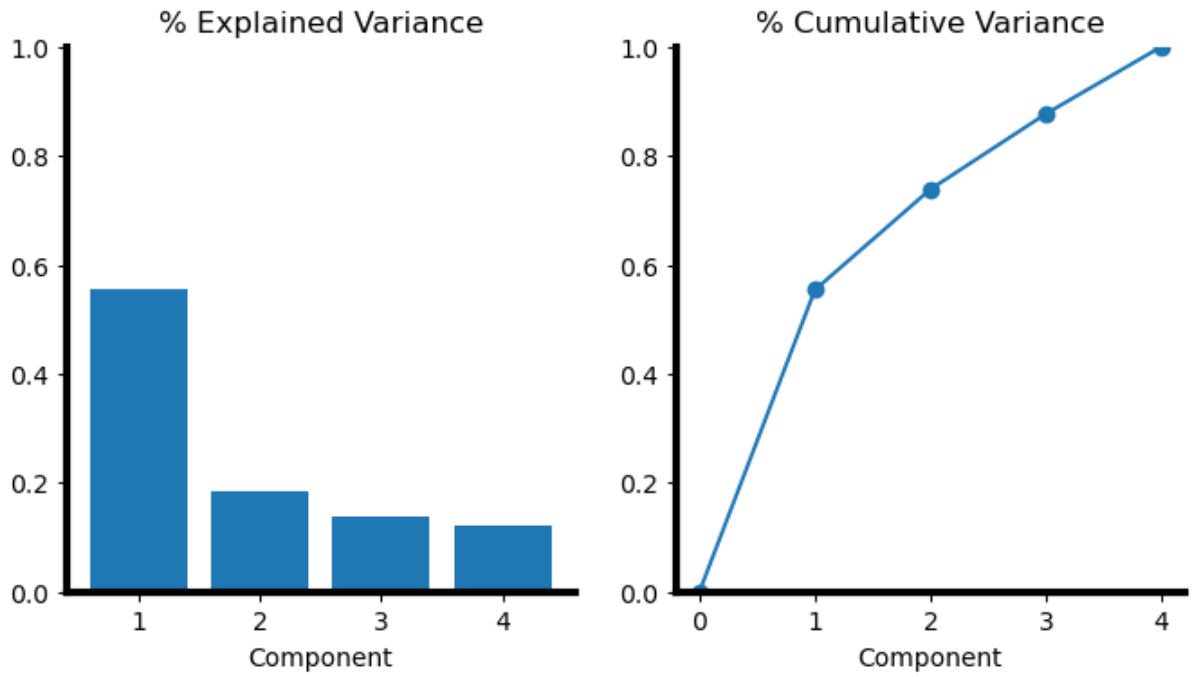


Figure 2-41: The explained variance and cumulative variance.

2.4. Regression Algorithms

As mentioned previously, the main goal behind using Ames Housing dataset is to predict the houses prices based on other housing features, therefore the appropriate ML approaches are the regression algorithms which are supervised ML algorithms. These algorithms include regression tree model, extreme gradient boosting (XGBoost) model, and linear regression. To compare the performance of these algorithms, we used Root Mean Square Error (RMSE).

2.4.1. Baseline Models

At first, these models were trained with dataset before applying the features engineering results. The goal here is to create baseline models to compare the performance of regression models with after applying the features engineering to see how much features engineering results improve the performance of regression models.

Figure 2-42 shows the RMSE for baseline models, it is clear that extreme gradient boosting (XGB) has the lowest RMSE value followed by regression tree (RT) model, whereas liner regression (LR) model has the highest value.

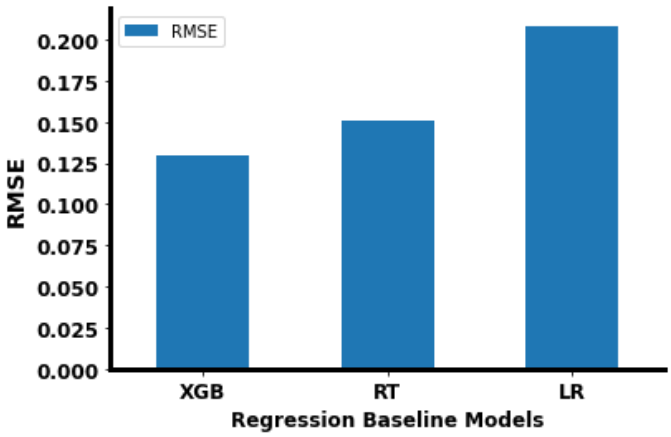


Figure 2-42: The RMSE for baseline models.

2.4.2. Regression Models with Engineered Features

After training the models with engineered features, the performance of extreme gradient boosting (XGB) and liner regression (LR) model have been improved as shown in Figure 2-43. The performance of regression tree (RT) model went to the worse. Figure 2-44 compares the performance of regression models in term of RMSE. Also, *Table 2-8* shows the exact RMSE values before and after features engineering. We can conclude that XGB and LR have been improved after applying features engineering, but XGB is still the best. Therefore, we selected XGB as our champion model and went further to tune its parameters by performing GridSearchCV to optimize XGB's hyperparameters. Figure 2-45 shows the RMSE plot before and after hyperparameter tuning for XGB model. The tuned XGB has lower RMSE, so the tuning improved XGB's performance.

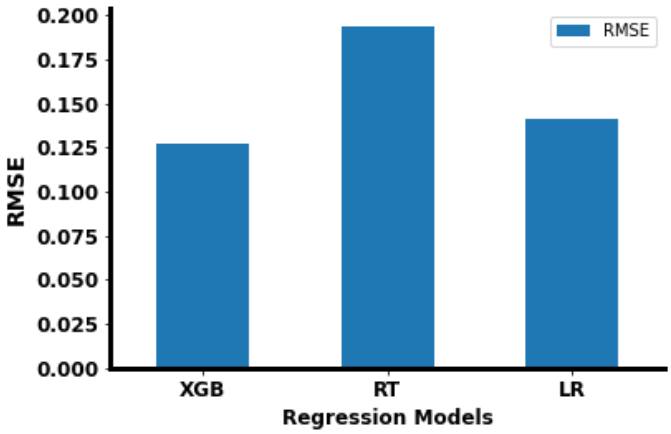


Figure 2-43: The RMSE for regression models after applying features engineering.

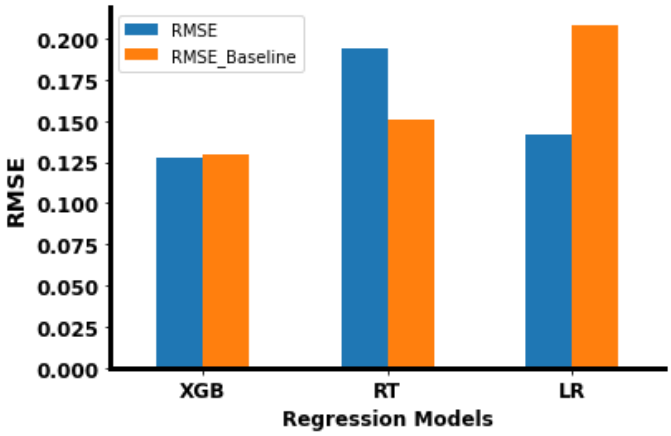


Figure 2-44: The RMSE of regression models before and after applying features engineering.

Table 2-8. RMSE values for regression models before and after applying features engineering.

	Regression Models	RMSE	RMSE_Baseline
0	XGB	0.127309	0.129941
1	RT	0.193836	0.150853
2	LR	0.141355	0.208400

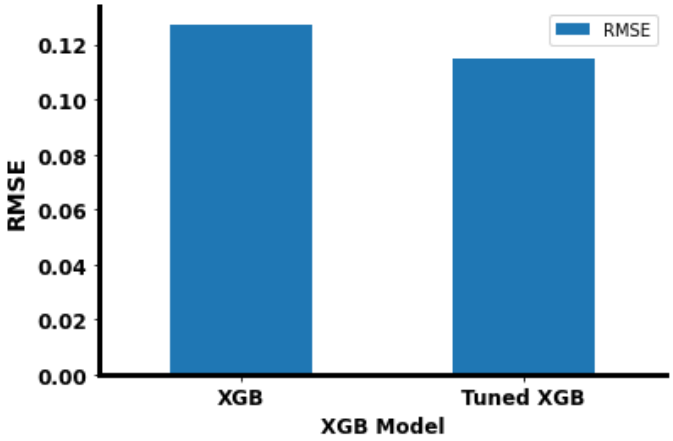


Figure 2-45: The RMSE of XGB model before and after parameters tuning.

The XGB model has been saved as pickle format to be used for prediction and avoid retraining. Also, the model has been scored with test dataset and provide predictions for houses prices as shown in *Table 2-9*.

Table 2-9. The sale price predictions for houses prices using XGB model.

	Id	SalePrice
947	2408	149302.781250
167	1628	270477.218750
620	2081	124235.437500
952	2413	132041.125000
1350	2811	157216.000000
1155	2616	140943.234375
1084	2545	136695.484375
84	1545	108411.140625
338	1799	114041.492188
88	1549	112806.640625

2.4.3. Feature Importance

We can get information about what features in the dataset the model think they are most important. Figure 2-46 shows the feature importance based on SHAP values, so the values have been ordered based on their importance to the model. We can check, for any single prediction from a model, how did each feature in the data affect that particular prediction as shown in Figure 2-47.

Finally, Figure 2-48 shows the feature impact on overall model prediction. So, we can get idea about how does each feature affect the model's predictions in a big-picture sense, i.e., what is its typical effect when considered over a large number of possible predictions.

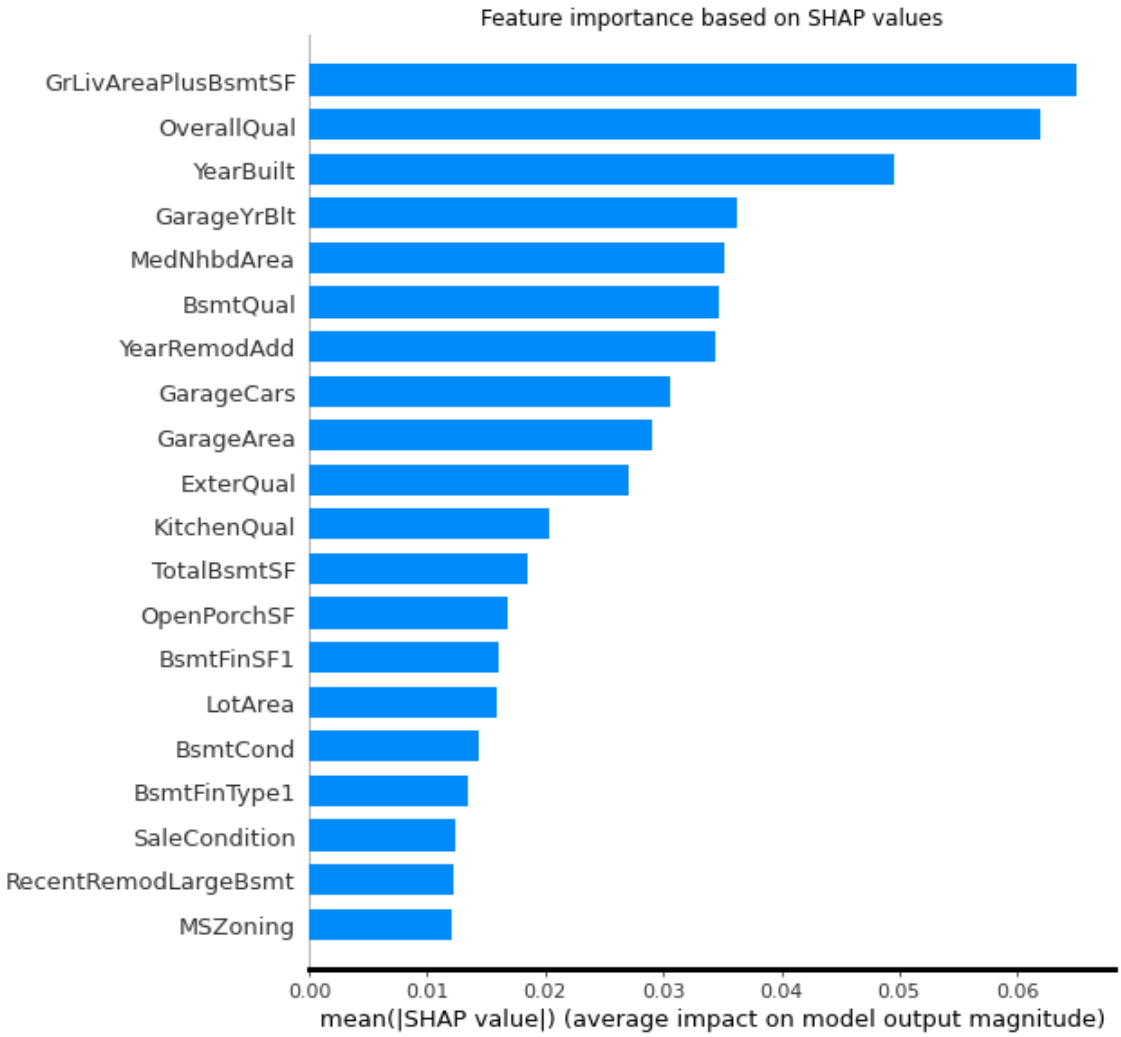


Figure 2-46: Feature importance based on SHAP values.

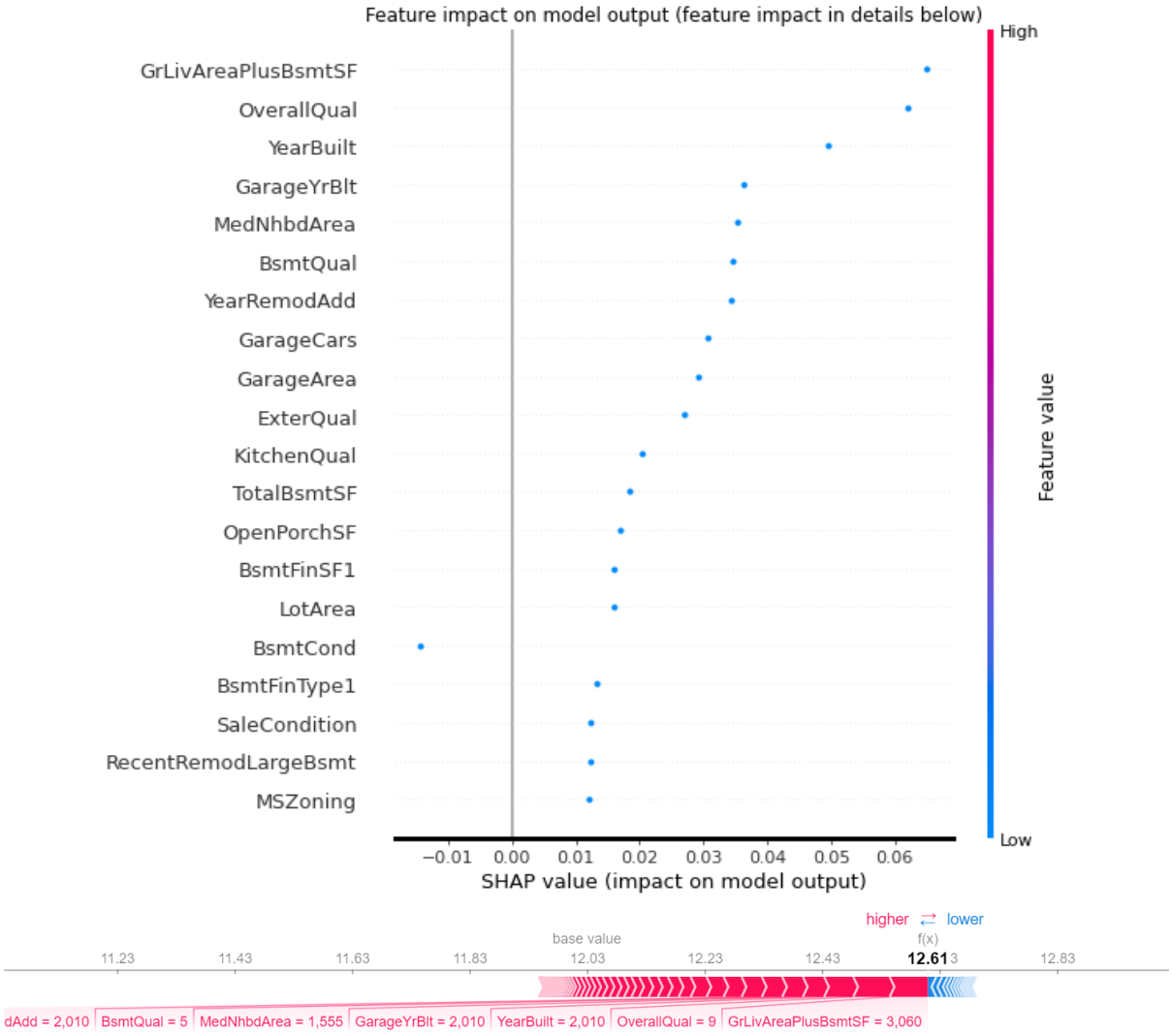


Figure 2-47: Feature impact on the model prediction.

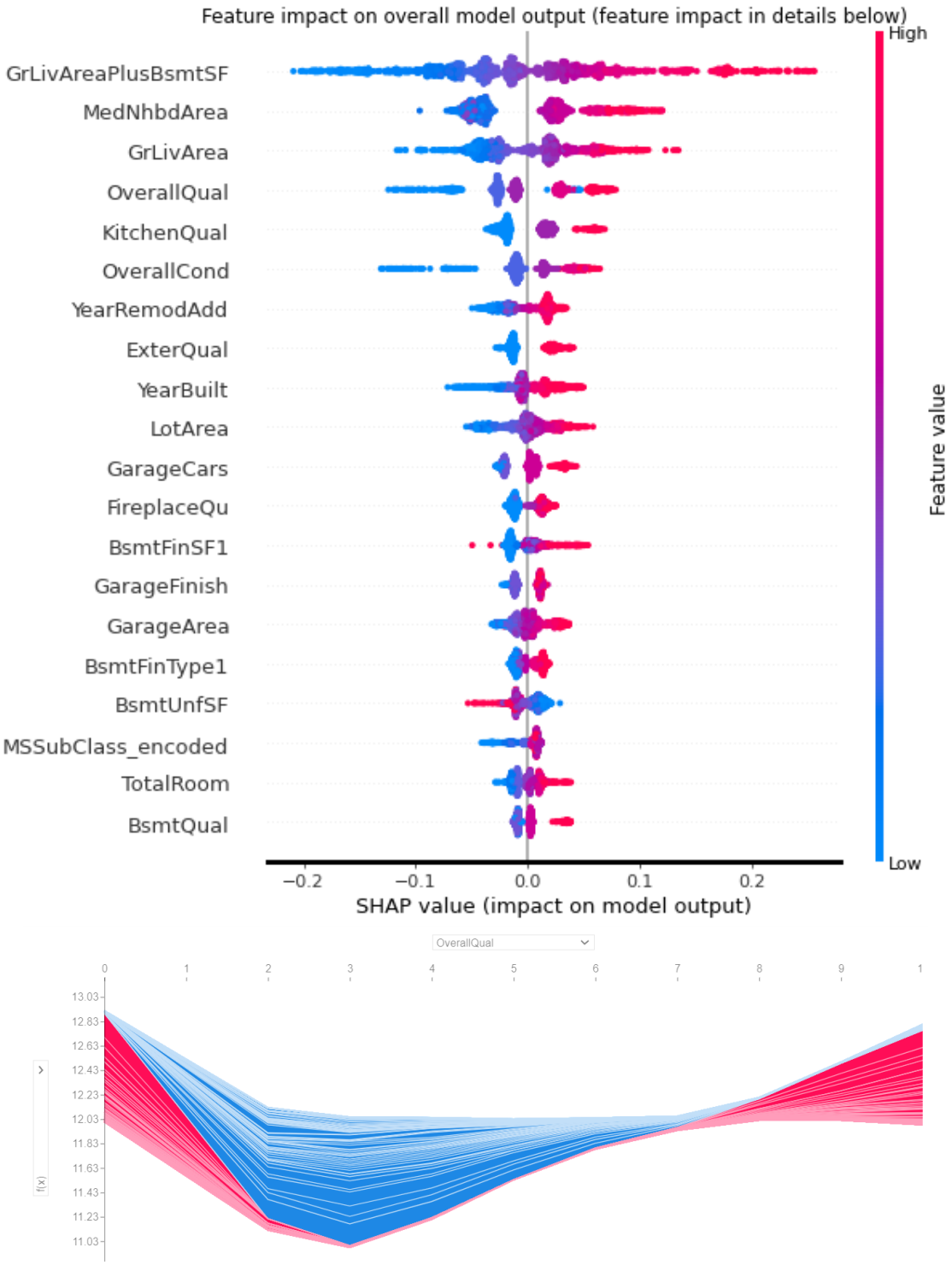


Figure 2-48: Feature impact on overall model prediction.