HOUSING PRICE PREDICTION

Submitted By:

Junaid Shaikh

ACKNOWLEDGMENT

I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project from which I have learned a lot.

Some of the reference sources are as follows:

- Medium.com
- StackOverflow

INTRODUCTION

BUSINESS PROBLEM FRAMING

This is a real estate problem where a US based housing company named Surprise Housing has decided to invest in Australian Market. Their agenda is to buy houses in Australia at prices below their actual value in the market and sell them at high prices to gain profit. To do this this company uses data analytics to decide in which property they must invest.

Company has collected the data of previously sold houses in Australia and with the help of this data they want to know to the value of prospective properties to decide whether it will suitable to invest in the properties or not.

To know the value of properties company has provided data to us to do data analysis and to extract the information of attributes which are important to predict the price of the houses. They want a machine learning model which can predict the price of houses and also the significance of each important attribute in house prediction i.e., how and to what intensity each variable impacts the price of the house.

Conceptual Background OF The Domain Problem

In real estate the value of property usually increases with time as seen in many countries. One of the causes for this is due to rising population.

The value of property also depends on the proximity of the property, its size its neighborhood and audience for which the property is subjected to be sold. For example, if audience is mainly concerned of commercial purpose. Then the property which is located in densely populated area will be sold very fast and at high prices compared to the one located at remote place. Similarly, if audience is concerned only on living place, then property with less dense area having large area with all services will be sold at higher prices.

The company is looking at prospective properties to buy houses to enter the market. We are required to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not.

Review OF Literature

Houses are one of the necessary needs of each person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy.

A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price.

We are required to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not.

Motivation OF The Problem Undertaken

To understand real world problems where Machine Learning and Data Analysis can be applied to help organizations in various domains to make better decisions with the help of which they can gain profit or can be escaped from any loss which otherwise could be possible without the study of data. One of such domains is Real Estate.

Houses are one of the necessary needs of each person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies. Our problem is related to one such housing company.

Analytical Problem Framing

Mathematical/ Analytical Modeling OF The Problem

This particular problem contains two datasets train dataset and test dataset. Model are built using train dataset. This model is then used to predict the SalePrice for test dataset. By analyzing into the target column. After analysis it was concluded that the data entries of SalePrice column contains data points of continuous nature, it is a Regression problem, hence all regression algorithms were used while building the model. While checking for the null values in the datasets, many columns with nan values were found and null values were replaced with suitable entries like mean for numerical columns and specific value for categorical columns. For further analyses graph plot like distribution plot, bar plot, reg plot and scatter plot were used. With these plots, the relation between the feature columns and target column was visualized. Upon analyzing outliers and skewness were found in the dataset and were removed. Outliers were removed using percentile method and Skewness using Yeo-Johnson method. All the regression models were iterated to find the best model and then further Hyper-tune the best model and save the best model. Finally, SalePrice was predicted for test dataset using the saved model built from train dataset.

Data Sources & Data Formats

The data was provided in csv (comma separated values) format.

It contained two datasets train and test dataset. Model was built using train dataset and then used to predict SalePrice for test dataset. Train dataset contains 1168 rows and 81 columns including target variable, and test dataset was having 292 rows and 80 columns excluding target variable. Dataset's columns have data types objects, float and integer type s of data.

Dataset was imported using Panda's library and then transformed into data-frame.

Train Dataset

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1163	289	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1164	554	20	RL	67.0	8777	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1165	196	160	RL	24.0	2280	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1166	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1167	617	60	RL	NaN	7861	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0

1168 rows × 81 columns

Data Pre-processing

Current dataset is raw data. By proper Data Transformation methods, a lot of valuable insights can be gained. Also, we need to convert categorical info into numerical data type. Categorical columns having categorical data point needs to be assigned a unique integer data point.

In ID and Utilities column the unique counts were 1168 and 1 resp, which concludes that all the data entries in ID column is unique. ID is the unique identity number given to every asset. Utilities column contained only one data point in whole dataset, hence these two columns were dropped.

In this project we have performed various mathematical and statistical analysis such as description or statistical summary of the data using describe, checked correlation using corr and also visualized it using heatmap. Then we have used Z-Score to plot outliers and remove them.

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF
count	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000
mean	56.767979	70.990582	10484.749144	6.104452	5.595890	1970.930651	1984.758562	101.696918	444.726027	46.647260	569.721747
std	41.940650	22.437057	8957.442311	1.390153	1.124343	30.145255	20.785185	182.218483	462.664785	163.520016	449.375525
min	20.000000	21.000000	1300.000000	1.000000	1.000000	1875.000000	1950.000000	0.000000	0.000000	0.000000	0.000000
25%	20.000000	60.000000	7621.500000	5.000000	5.000000	1954.000000	1966.000000	0.000000	0.000000	0.000000	216.000000
50%	50.000000	71.000000	9522.500000	6.000000	5.000000	1972.000000	1993.000000	0.000000	385.500000	0.000000	474.000000
75%	70.000000	79.250000	11515.500000	7.000000	6.000000	2000.000000	2004.000000	160.000000	714.500000	0.000000	816.000000
max	190.000000	313.000000	164660.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000	1474.000000	2336.000000

TotalBsmtSF	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr
1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000
1061.095034	1169.860445	348.826199	6.380137	1525.066781	0.425514	0.055651	1.562500	0.388699	2.884418	1.045377
442.272249	391.161983	439.696370	50.892844	528.042957	0.521615	0.236699	0.551882	0.504929	0.817229	0.216292
0.000000	334.000000	0.000000	0.000000	334.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
799.000000	892.000000	0.000000	0.000000	1143.250000	0.000000	0.000000	1.000000	0.000000	2.000000	1.000000
1005.500000	1096.500000	0.000000	0.000000	1468.500000	0.000000	0.000000	2.000000	0.000000	3.000000	1.000000
1291.500000	1392.000000	729.000000	0.000000	1795.000000	1.000000	0.000000	2.000000	1.000000	3.000000	1.000000
6110.000000	4692.000000	2065.000000	572.000000	5642.000000	3.000000	2.000000	3.000000	2.000000	8.000000	3.000000

TotRmsAbvGrd	Fireplaces	GarageYrBlt	GarageCars	GarageArea	WoodDeckSF	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea
1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000
6.542808	0.617295	1869.799658	1.776541	476.860445	96.206336	46.559932	23.015411	3.639555	15.051370	3.448630
1.598484	0.650575	451.037303	0.745554	214.466769	126.158988	66.381023	63.191089	29.088867	55.080816	44.896939
2.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5.000000	0.000000	1957.750000	1.000000	338.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6.000000	1.000000	1977.000000	2.000000	480.000000	0.000000	24.000000	0.000000	0.000000	0.000000	0.000000
7.000000	1.000000	2001.000000	2.000000	576.000000	171.000000	70.000000	0.000000	0.000000	0.000000	0.000000
14.000000	3.000000	2010.000000	4.000000	1418.000000	857.000000	547.000000	552.000000	508.000000	480.000000	738.000000

MiscVal	MoSold	YrSold	SalePrice
1168.000000	1168.000000	1168.000000	1168.000000
47.315068	6.344178	2007.804795	181477.005993
543.264432	2.686352	1.329738	79105.586863
0.000000	1.000000	2006.000000	34900.000000
0.000000	5.000000	2007.000000	130375.000000
0.000000	6.000000	2008.000000	163995.000000
0.000000	8.000000	2009.000000	215000.000000
15500.000000	12.000000	2010.000000	755000.000000

From this statistical analysis we make some of the interpretations that,

- Maximum standard deviation of 8957.44 is observed in LotArea column.
- Maximum SalePrice of a house observed is 755000 and minimum is 34900.
- In the columns Id, MSSubclass, LotArea, MasVnrArea, BsmtFinSF1, BsmtFinSF2, BsmtUnfsF, TotalBsmtSF, 1stFlrSF, 2ndFlrSF, LowQualFinSF, GrLivArea, BsmtFullBath, HalfBath, TotRmsAbvGrd, WoodDeckSF, OpenPorchSF, EnclosedPorch, 3SsnPorch, ScreenPorch, PoolArea, Miscval, salePrice mean is considerably greater than median so the columns are positively skewed.
- In the columns FullBath, BedroomAbvGr, Fireplaces, Garagecars, GarageArea, YrSold Median is greater than mean so the columns are negatively skewed.

  In the columns Id, MSSubClass, LotFrontage, LotArea, MasVnrArea, BsmtFinSF1, BsmtFinSF2, BsmtUnfSF, TotalBsmtSF, 1stFlrSF, 2ndFlrSF, LowQualFinSF, GrLivArea, BsmtHalfBath, BedroomAbvGr, ToRmsAbvGrd, GarageArea, WoodDeckSF, OpenPorchSF, EnclosedPorch, 3SsnPorch, ScreenPorch, PoolArea, MiscVal, SalePrice there is considerable difference between the 75 percentile and maximum, so outliers are present.

Column Description

The variable features of this problem statement are as:

MSSubClass: Identifies the type of dwelling involved in the sale

MSZoning: Identifies the general zoning classification of the sale

LotFrontage: Linear feet of street connected to property

LotArea: Lot size in square feet

Street: Type of road access to property

Alley: Type of alley access to property

LotShape: General shape of property

LandContour: Flatness of the property

Utilities: Type of utilities available

LotConfig: Lot configuration

LandSlope: Slope of property

Neighborhood: Physical locations within Ames city limits

Condition1: Proximity to various conditions

Condition2: Proximity to various conditions (if more than one is present)

BldgType: Type of dwelling

HouseStyle: Style of dwelling

OverallQual: Rates the overall material and finish of the house

OverallCond: Rates the overall condition of the house

YearBuilt: Original construction date

YearRemodAdd: Remodel date (same as construction date if no remodeling or additions)

RoofStyle: Type of roof

RoofMatl: Roof material

Exterior1st: Exterior covering on house

Exterior2nd: Exterior covering on house (if more than one material)

MasVnrType: Masonry veneer type

MasVnrArea: Masonry veneer area in square feet

ExterQual: Evaluates the quality of the material on the exterior

ExterCond: Evaluates the present condition of the material on the exterior

Foundation: Type of foundation

BsmtQual: Evaluates the height of the basement

BsmtCond: Evaluates the general condition of the basement

BsmtExposure: Refers to walkout or garden level walls

BsmtFinType1: Rating of basement finished area

BsmtFinSF1: Type 1 finished square feet

BsmtFinType2: Rating of basement finished area (if multiple types)

BsmtFinSF2: Type 2 finished square feet

BsmtUnfSF: Unfinished square feet of basement area

TotalBsmtSF: Total square feet of basement area

Heating: Type of heating

HeatingQC: Heating quality and condition

CentralAir: Central air conditioning

Electrical: Electrical system

1stFlrSF: First Floor square feet

2ndFlrSF: Second floor square feet

LowQualFinSF: Low quality finished square feet (all floors)

GrLivArea: Above grade (ground) living area square feet

BsmtFullBath: Basement full bathrooms

BsmtHalfBath: Basement half bathrooms

FullBath: Full bathrooms above grade

HalfBath: Half baths above grade

Bedroom: Bedrooms above grade (does NOT include basement bedrooms)

Kitchen: Kitchens above grade

KitchenQual: Kitchen quality

TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)

Functional: Home functionality (Assume typical unless deductions are warranted)

Fireplaces: Number of fireplaces

FireplaceQu: Fireplace quality

GarageType: Garage location

GarageYrBlt: Year garage was built

GarageFinish: Interior finish of the garage

GarageCars: Size of garage in car capacity

GarageArea: Size of garage in square feet

GarageQual: Garage quality

GarageCond: Garage condition

PavedDrive: Paved driveway

WoodDeckSF: Wood deck area in square feet

OpenPorchSF: Open porch area in square feet

EnclosedPorch: Enclosed porch area in square feet

3SsnPorch: Three season porch area in square feet

ScreenPorch: Screen porch area in square feet

PoolArea: Pool area in square feet

PoolQC: Pool quality Fence: Fence quality

MiscFeature: Miscellaneous feature not covered in other categories

MiscVal: $Value of miscellaneous feature

MoSold: Month Sold (MM)

YrSold: Year Sold (YYYY)

SaleType: Type of sale

SaleCondition: Condition of sale

Data Inputs-Logic-Output Relationship

SalePrice is our target variable i.e., output column. Rest all columns are to be used as feature column i.e., input column. We need to find which feature columns have positive correlation and which have negative correlation, accordingly to train our model. Columns which have no correlation have to dropped.

Execute command info to obtain descriptive summary of the train dataset

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Id              1168 non-null   int64
 1   MSSubClass      1168 non-null   int64
 2   MSZoning        1168 non-null   object
 3   LotFrontage     954 non-null    float64
 4   LotArea         1168 non-null   int64
 5   Street          1168 non-null   object
 6   Alley           77 non-null     object
 7   LotShape        1168 non-null   object
 8   LandContour     1168 non-null   object
 9   Utilities       1168 non-null   object
 10  LotConfig       1168 non-null   object
 11  LandSlope       1168 non-null   object
 12  Neighborhood    1168 non-null   object
 13  Condition1      1168 non-null   object
 14  Condition2      1168 non-null   object
 15  BldgType        1168 non-null   object
 16  HouseStyle      1168 non-null   object
 17  OverallQual     1168 non-null   int64
 18  OverallCond     1168 non-null   int64
 19  YearBuilt       1168 non-null   int64
 20  YearRemodAdd    1168 non-null   int64
 21  RoofStyle       1168 non-null   object
 22  RoofMatl        1168 non-null   object
 23  Exterior1st     1168 non-null   object
 24  Exterior2nd     1168 non-null   object
 25  MasVnrType      1161 non-null   object
 26  MasVnrArea      1161 non-null   float64
 27  ExterQual       1168 non-null   object
 28  ExterCond       1168 non-null   object
 29  Foundation      1168 non-null   object
 30  BsmtQual        1138 non-null   object
 31  BsmtCond        1138 non-null   object
 32  BsmtExposure    1137 non-null   object
 33  BsmtFinType1    1138 non-null   object
 34  BsmtFinSF1      1168 non-null   int64
 35  BsmtFinType2    1137 non-null   object
 36  BsmtFinSF2      1168 non-null   int64
 37  BsmtUnfSF       1168 non-null   int64
 38  TotalBsmtSF     1168 non-null   int64
 39  Heating         1168 non-null   object
 40  HeatingQC       1168 non-null   object
 41  CentralAir      1168 non-null   object
 42  Electrical      1168 non-null   object
 43  1stFlrSF        1168 non-null   int64
 44  2ndFlrSF        1168 non-null   int64
 45  LowQualFinSF    1168 non-null   int64
 46  GrLivArea       1168 non-null   int64
 47  BsmtFullBath    1168 non-null   int64
 48  BsmtHalfBath    1168 non-null   int64
 49  FullBath        1168 non-null   int64
 50  HalfBath        1168 non-null   int64
 51  BedroomAbvGr    1168 non-null   int64
 52  KitchenAbvGr    1168 non-null   int64
 53  KitchenQual     1168 non-null   object
 54  TotRmsAbvGrd    1168 non-null   int64
 55  Functional      1168 non-null   object
 56  Fireplaces      1168 non-null   int64
 57  FireplaceQu     617 non-null    object
 58  GarageType      1104 non-null   object
 59  GarageYrBlt     1104 non-null   float64
 60  GarageFinish    1104 non-null   object
 61  GarageCars      1168 non-null   int64
 62  GarageArea      1168 non-null   int64
 63  GarageQual      1104 non-null   object
 64  GarageCond      1104 non-null   object
 65  PavedDrive      1168 non-null   object
 66  WoodDeckSF      1168 non-null   int64
 67  OpenPorchSF     1168 non-null   int64
 68  EnclosedPorch   1168 non-null   int64
 69  3SsnPorch       1168 non-null   int64
 70  ScreenPorch     1168 non-null   int64
 71  PoolArea        1168 non-null   int64
```

```
 71  PoolArea        1168 non-null   int64
 72  PoolQC          7 non-null      object
 73  Fence           237 non-null    object
 74  MiscFeature     44 non-null     object
 75  MiscVal         1168 non-null   int64
 76  MoSold          1168 non-null   int64
 77  YrSold          1168 non-null   int64
 78  SaleType        1168 non-null   object
 79  SaleCondition   1168 non-null   object
 80  SalePrice       1168 non-null   int64
dtypes: float64(3), int64(35), object(43)
```

These 81 columns comprise of both dimensions (categorical value) and measures (numeric value)

The dataset is not clean, i.e., consists of missing values as well

We can fill null values of categorical columns with 0 as it basically represents that, the feature is not available for the property.

For missing values in numerical column, we can fill them mean of the resp column.

Next step is to assign every categorical data a unique value.

Verify whether all categorical columns are converted into numerical column using info command.

```
#   Column         Non-Null Count  Dtype
---  ------         --------------  -----
0   MSSubClass     1168 non-null   int64
1   MSZoning       1168 non-null   int64
2   LotFrontage    1168 non-null   float64
3   LotArea        1168 non-null   int64
4   Street         1168 non-null   int64
5   Alley          1168 non-null   int64
6   LotShape       1168 non-null   int64
7   LandContour    1168 non-null   int64
8   Utilities      1168 non-null   int64
9   LotConfig      1168 non-null   int64
10  LandSlope      1168 non-null   int64
11  Neighborhood   1168 non-null   int64
12  Condition1     1168 non-null   int64
13  Condition2     1168 non-null   int64
14  BldgType       1168 non-null   int64
15  HouseStyle     1168 non-null   int64
16  OverallQual    1168 non-null   int64
17  OverallCond    1168 non-null   int64
18  YearBuilt      1168 non-null   int64
19  YearRemodAdd   1168 non-null   int64
20  RoofStyle      1168 non-null   int64
21  RoofMatl       1168 non-null   int64
22  Exterior1st    1168 non-null   int64
23  Exterior2nd    1168 non-null   int64
24  MasVnrType     1168 non-null   int64
25  MasVnrArea     1168 non-null   float64
26  ExterQual      1168 non-null   int64
27  ExterCond      1168 non-null   int64
28  Foundation     1168 non-null   int64
29  BsmtQual       1168 non-null   int64
30  BsmtCond       1168 non-null   int64
31  BsmtExposure   1168 non-null   int64
32  BsmtFinType1   1168 non-null   int64
33  BsmtFinSF1     1168 non-null   int64
34  BsmtFinType2   1168 non-null   int64
35  BsmtFinSF2     1168 non-null   int64
36  BsmtUnfSF      1168 non-null   int64
37  TotalBsmtSF    1168 non-null   int64
38  Heating        1168 non-null   int64
39  HeatingQC      1168 non-null   int64
40  CentralAir     1168 non-null   int64
41  Electrical     1168 non-null   int64
42  1stFlrSF       1168 non-null   int64
43  2ndFlrSF       1168 non-null   int64
44  LowQualFinSF   1168 non-null   int64
45  GrLivArea      1168 non-null   int64
46  BsmtFullBath   1168 non-null   int64
47  BsmtHalfBath   1168 non-null   int64
48  FullBath       1168 non-null   int64
49  HalfBath       1168 non-null   int64
50  BedroomAbvGr   1168 non-null   int64
51  KitchenAbvGr   1168 non-null   int64
52  KitchenQual    1168 non-null   int64
53  TotRmsAbvGrd   1168 non-null   int64
54  Functional     1168 non-null   int64
55  Fireplaces     1168 non-null   int64
56  FireplaceQu    1168 non-null   int64
57  GarageType     1168 non-null   int64
58  GarageYrBlt    1168 non-null   float64
59  GarageFinish   1168 non-null   int64
60  GarageCars     1168 non-null   int64
61  GarageArea     1168 non-null   int64
62  GarageQual     1168 non-null   int64
63  GarageCond     1168 non-null   int64
64  PavedDrive     1168 non-null   int64
65  WoodDeckSF     1168 non-null   int64
66  OpenPorchSF    1168 non-null   int64
67  EnclosedPorch  1168 non-null   int64
68  3SsnPorch      1168 non-null   int64
69  ScreenPorch    1168 non-null   int64
70  PoolArea       1168 non-null   int64
71  PoolQC         1168 non-null   int64
72  Fence          1168 non-null   int64
73  MiscFeature    1168 non-null   int64
74  MiscVal        1168 non-null   int64
75  MoSold         1168 non-null   int64
76  YrSold         1168 non-null   int64
77  SaleType       1168 non-null   int64
78  SaleCondition  1168 non-null   int64
79  SalePrice      1168 non-null   int64
dtypes: float64(3), int64(77)
memory usage: 730.1 KB
```

Hardware and Software Requirements and Tools Used

Hardware required:

- Processor — core i5 and above
- RAM — 8 GB or above
- SSD — 250GB or above

Software/s required: Anaconda

LIBRARIES:

The tools, libraries, and packages we used for accomplishing this project are pandas, numpy, matplotlib, seaborn, scipy, sklearn, mlxtend, xgboost, joblib.

Through pandas library we loaded our csv file 'Data file' into dataframe and performed data manipulation and analysis.

With the help of numpy we worked with arrays.

With the help of matplotlib and seaborn we did plot various graphs and figures and done data visualization.

Train_test_split is a function in Sklearn model selection for splitting data arrays into two subsets: for training data and for testing data. With this function, you don't need to divide the dataset manually. By default, Sklearn train_test_split will make random partitions for the two subsets.

With sklearn's standardscaler package we scaled all the feature variables onto single scale. As these columns are different in scale, they are standardized to have common scale while building machine learning model. This is useful when you want to compare data that correspond to different units.

With sklearn's package we imported many regression models, we could obtain cross_val_score which is an accuracy metric used to evaluate model, we could obtain best parameters of a model using GridSearchCV or RandomizedSearchCV, we could reduce skewness using power transform library of sklearn.

With mlxtend package we could stack low performing models to obtain a high accuracy model.

Model/s Development and Evaluation

Identification of possible problem-solving approaches

Null values of numerical columns can be filled by replacing null values with mean of respective column. Null values can of categorical column can be either replaced by using mode value or if it’s for a column having ordinal datapoint, we can perform ordinal encoding.

If outliers are present, we shall remove them using Z-Score, IQR method or by percentile method. If skewness exists, we shall remove them using Yeo-Johnson method.

If models have low accuracy, we shall fine tune them to improve accuracy but if accuracy is still low then we shall stack up our top performing models to boost accuracy by combining models.

Testing of Identified Approaches (Algorithms)

We can check null values using info function. Outliers can be detected using Boxplot. Skewness can be detected using skew function. Our target variable is SalePrice which has datapoints of continous in nature, hence it is a regression problem. For that we shall use all regression algorithms to find & build the optimized model. By looking into the difference of r2 score and cross validation score of each model we can find our best model with least difference. To get the best model we have to run through multiple models and to avoid the confusion of overfitting we have go through cross validation.

Run & evaluate selected models

```
models=[GradientBoostingRegressor(),NuSVR(),LinearRegression(),Ridge(),RidgeCV(),BayesianRidge(),SGDRegressor(),SVR(),
        AdaBoostRegressor(),LinearSVR(),KNeighborsRegressor(),RandomForestRegressor(),BaggingRegressor(),
       DecisionTreeRegressor(),LGBMRegressor(), XGBRFRegressor(),XGBRegressor(),LogisticRegression(),ExtraTreesRegressor()]
```

```
for i in models:
    x_train,x_test,y_train,y_test=train_test_split(x,y,random_state = 42,test_size=0.20)
    scores=cross_val_score(i,x_train,y_train,cv=5,scoring='r2',n_jobs=-1)
    score=np.mean(scores)
    i.fit(x_train,y_train)
    y_pred=i.predict(x_test)
    if r2_score(y_test,y_pred)>score:
        diff=r2_score(y_test,y_pred)-score
    else:
        diff=score-r2_score(y_test,y_pred)
    print('*'*10)
    print(i)
    print('score',score)
    print('r2',r2_score(y_test,y_pred))
    print('diff',diff)
    print('mae',mean_absolute_error(y_test, y_pred))
    print('rmse',np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
**********
GradientBoostingRegressor()
score 0.84202015532579
r2 0.817042062025452
diff 0.024978093300337956
mae 18491.01004254228
rmse 35728.1398169082
**********
NuSVR()
score -0.016386606493028 21
r2 -0.008689780769930655
diff 0.007696825723097555
mae 57484.218333046425
rmse 83890.65538631662
**********
LinearRegression()
score 0.8018427485473225
r2 0.7716364558845128
diff 0.03020629266280972
mae 21650.755540611055
rmse 39916.1057061981
**********
Ridge()
score 0.8022439202493714
r2 0.7717781004919595
diff 0.030465819757411916
mae 21623.92812544445
rmse 39903.72461853671
**********
RidgeCV(alphas=array([ 0.1,  1. , 10. ]))
score 0.8052810050772298
r2 0.7725759243164028
diff 0.03270508076082701
mae 21429.939870235718
rmse 39833.915335579455
```

```
**********
BayesianRidge()
score 0.8107655886398216
r2 0.7734656691261194
diff 0.03729991951370226
mae 21175.833242136607
rmse 39755.91842168027
**********
SGDRegressor()
score 0.798033691154122
r2 0.7696370802710762
diff 0.028396610883045792
mae 21396.82728517374
rmse 40090.46227077579
**********
SVR()
score -0.05498952035287763
r2 -0.038953484001147176
diff 0.016036036351730454
mae 57124.51260057353
rmse 85139.83980752791
**********
AdaBoostRegressor()
score 0.8002951100621216
r2 0.7298258513602652
diff 0.07046925870185639
mae 25362.19737536402
rmse 43416.68619858223
**********
LinearSVR()
score -5.475783044568629
r2 -4.646418602464995
diff 0.829364442103633 7
mae 180099.4332129291
rmse 198482.08468133438
```

```
**********
KNeighborsRegressor()
score 0.7504654468076735
r2 0.7570132491766333
diff 0.006547802368959799
mae 24228.020512820516
rmse 41174.285702271845
**********
RandomForestRegressor()
score 0.8395767611095944
r2 0.7877437306781712
diff 0.05183303043142329
mae 19383.014017094014
rmse 38482.65613968785
**********
BaggingRegressor()
score 0.8195482347886817
r2 0.7402572060350301
diff 0.07929102875365157
mae 21455.223931623932
rmse 42570.28235396726
**********
DecisionTreeRegressor()
score 0.6269853510812382
r2 0.571965174215431
diff 0.05502017686580718
mae 29763.594017094016
rmse 54648.00720536014
**********
LGBMRegressor()
score 0.8530894222284748
r2 0.786645022773341
diff 0.0664443994551338
mae 19818.048194043058
rmse 38582.126992681966
```

```
ExtraTreesRegressor()
score 0.8572843830878465
r2 0.7943592337095083
diff 0.06292514937833815
mae 19272.629316239316
rmse 37878.20442726365
XGBRFRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
               colsample_bytree=1, gamma=0, gpu_id=-1, importance_type='gain',
               interaction_constraints='', max_delta_step=0, max_depth=6,
               min_child_weight=1, missing=nan, monotone_constraints='()',
               n_estimators=100, n_jobs=8, num_parallel_tree=100,
               objective='reg:squarederror', random_state=0, reg_alpha=0,
               scale_pos_weight=1, tree_method='exact', validate_parameters=1,
               verbosity=None)
score 0.8177052093979029
r2 0.7795861269010174
diff 0.03811908249688556
mae 21243.909455128207
rmse 39215.182442261415
**********
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.300000012, max_delta_step=0, max_depth=6,
             min_child_weight=1, missing=nan, monotone_constraints='()',
             n_estimators=100, n_jobs=8, num_parallel_tree=1, random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
             tree_method='exact', validate_parameters=1, verbosity=None)
score 0.8294024669431674
r2 0.7711003857562838
diff 0.058302081186883625
mae 20288.5914463141
rmse 39962.92861123615
**********
LogisticRegression()
score 0.5548319963186098
r2 0.6081085027805024
diff 0.053276506461892637
mae 32626.581196581195
rmse 52289.88626679993
```

Key Metrics for success in solving problem under consideration

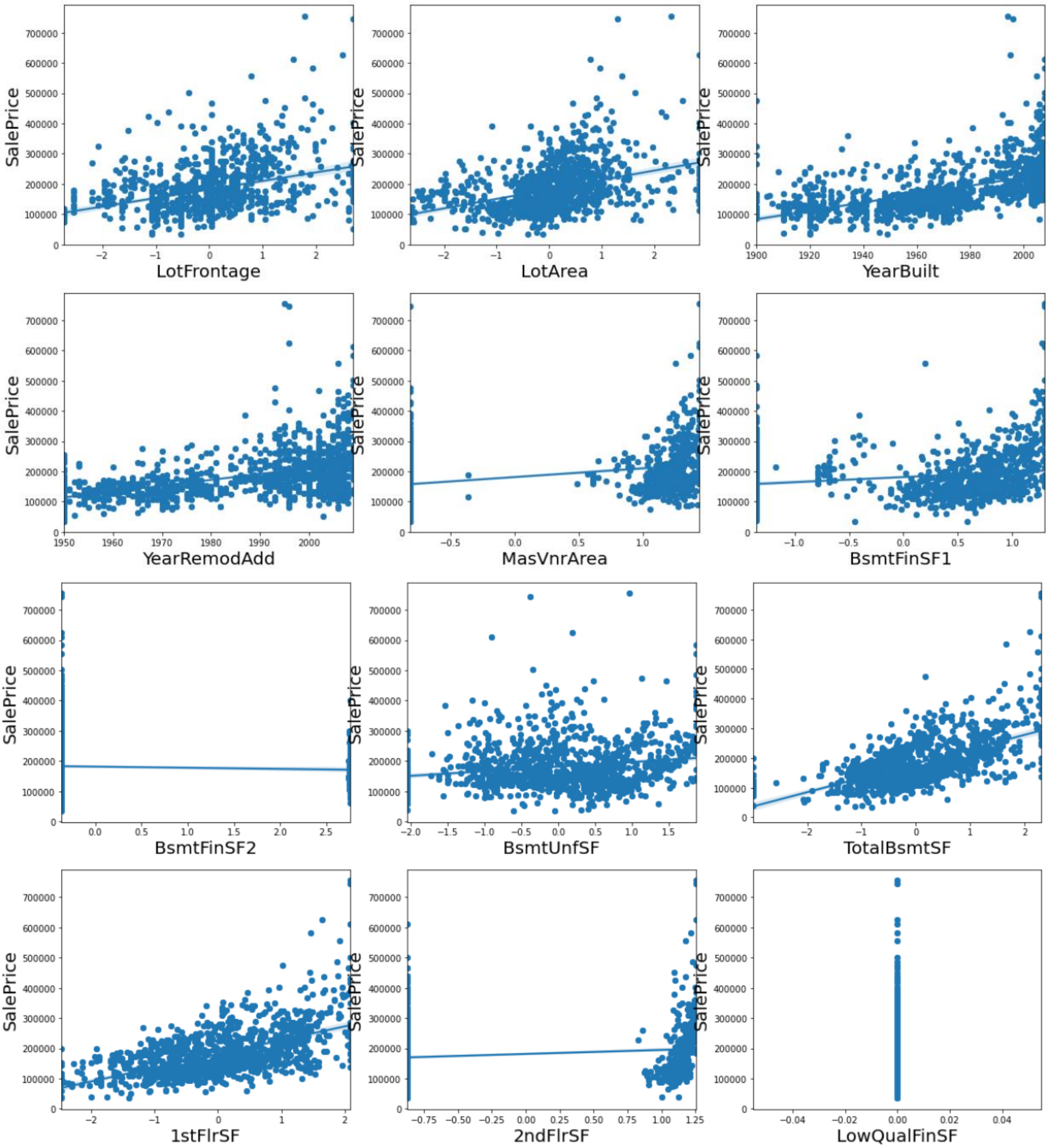
Following metrics were used to evaluate our model:

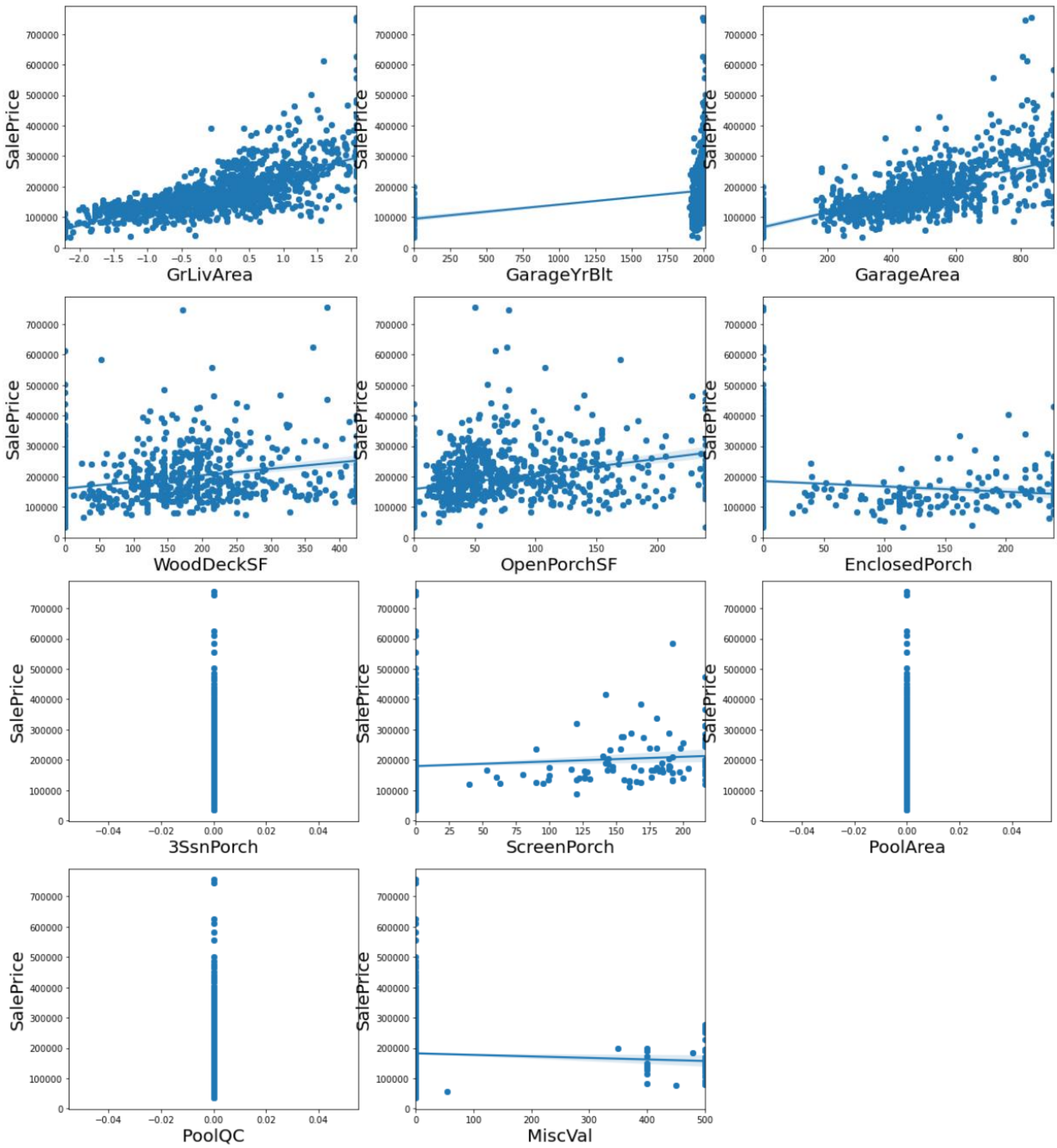
--- Cross Val Score
--- R2 Score
--- Mean absolute error
--- Mean squared error
--- Standard deviation error

Visualization

Scatter Plot & Regression Plot

Plot graph for all columns having datapoints of continuous nature w.r.t target variable to find the relation between the feature column and target variable.





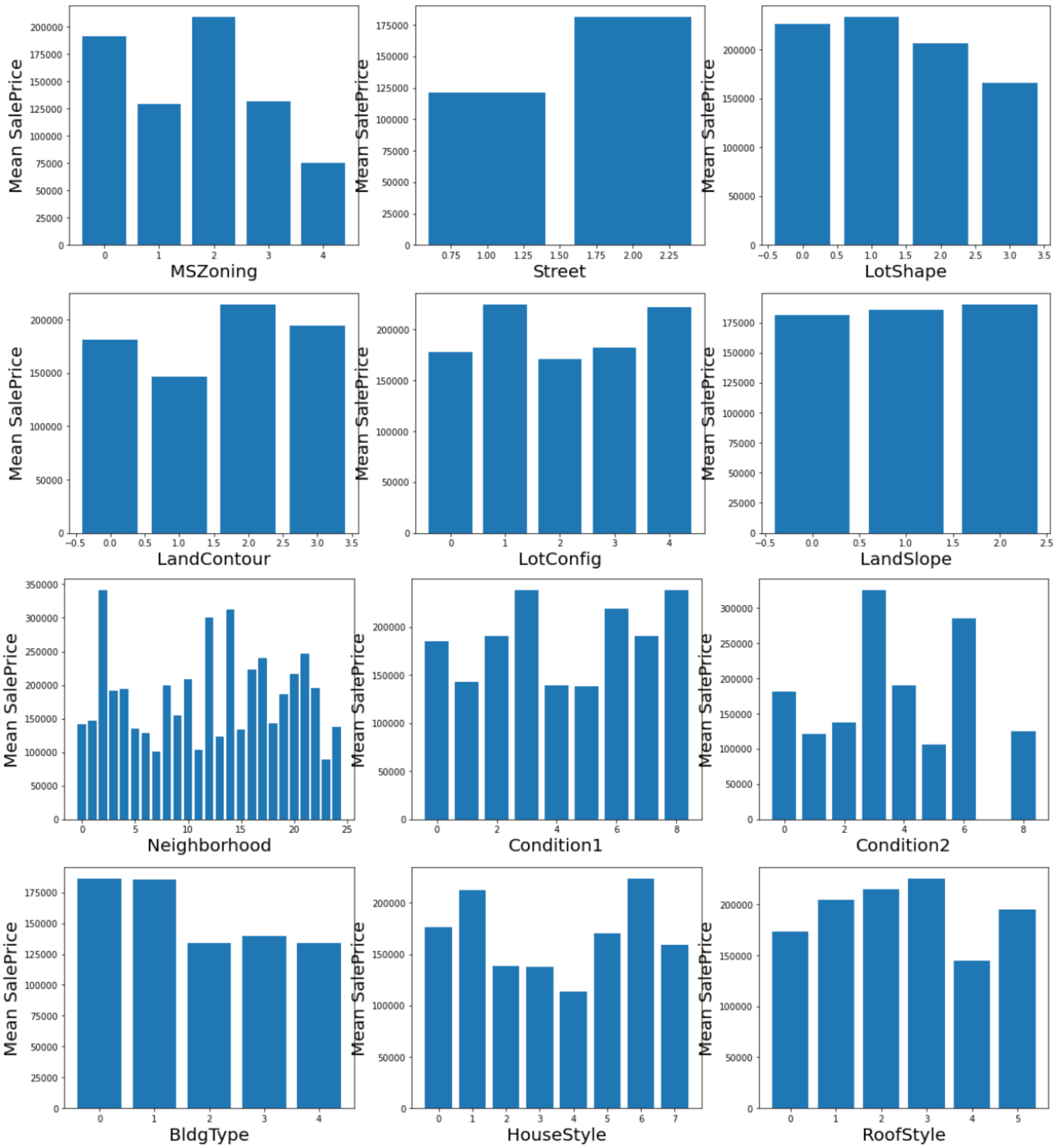
Observations:

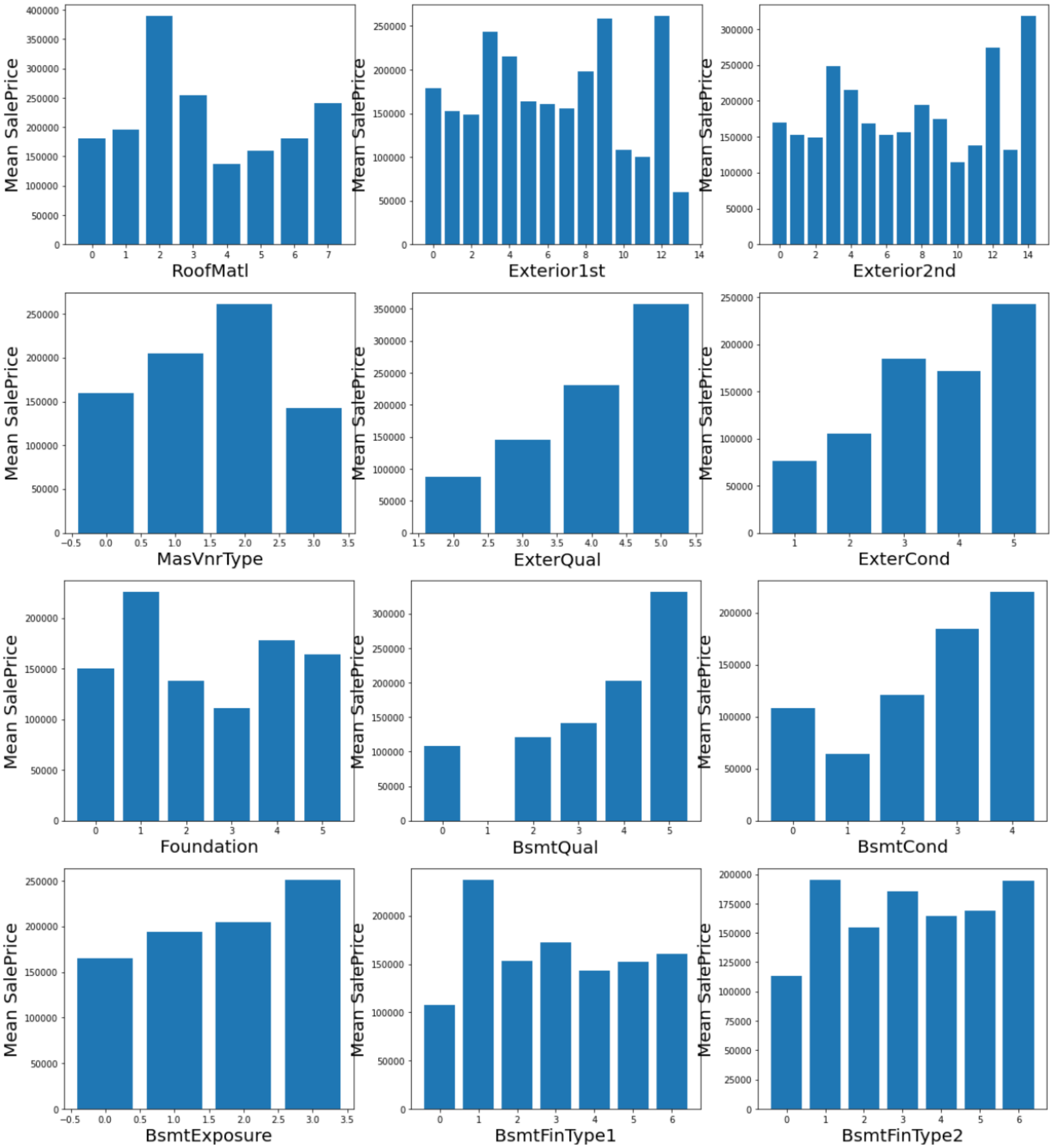
Columns such as OpenporchSF, WoodDeckSF, GarageArea, GarageYrBuilt, GrlivArea, 1stflrsf, 2ndflrsf, totalbsmtsf, YearremodAdd, Yearbuilt, lotfrontage, lotarea, masvnrarea, bsmntfinsf1 have positive correlation w.r.t salesprice

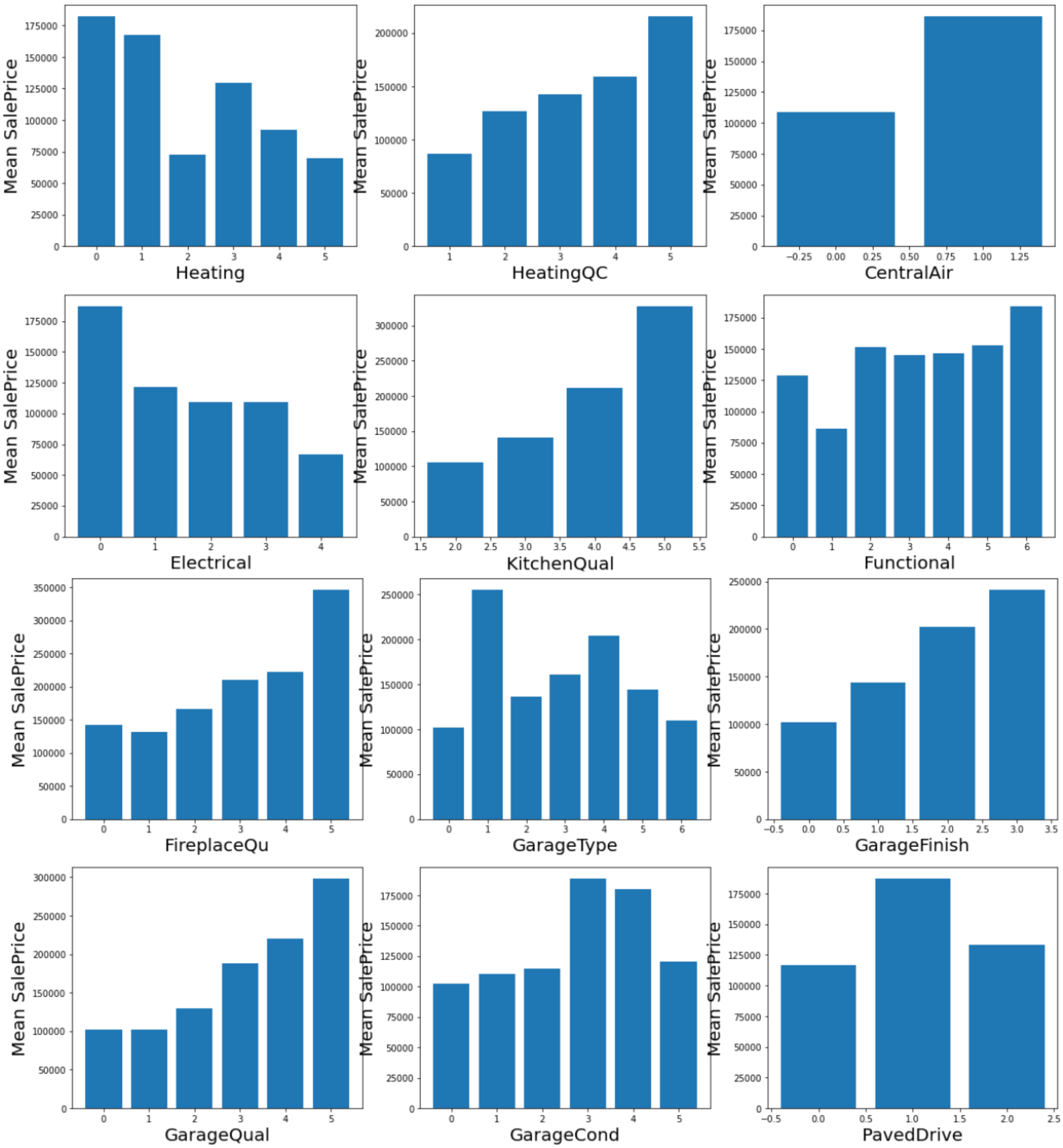
Columns such as MiscVal, Enclosedporch have slight negative correlation

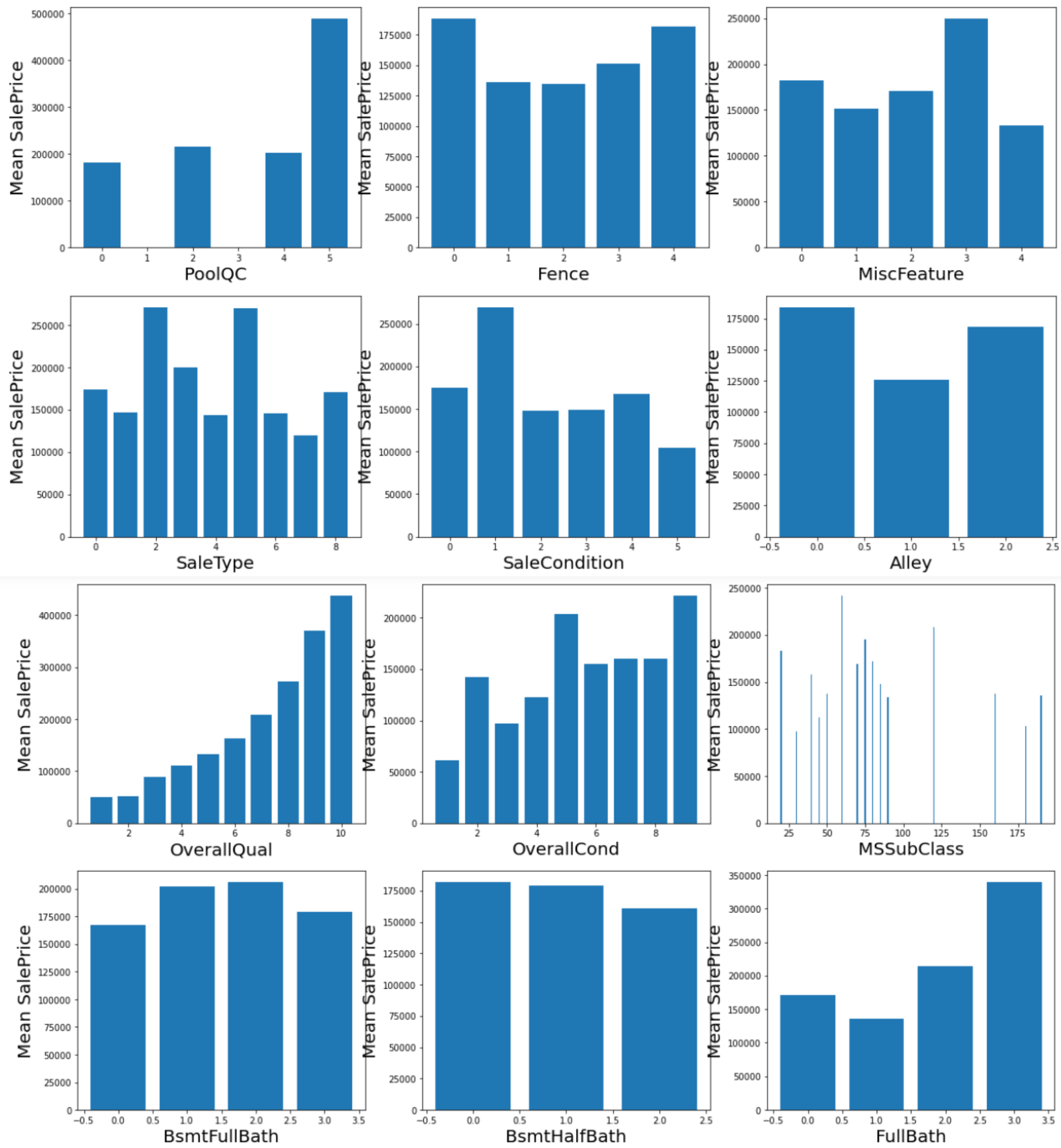
Remaining columns do not have any correlation w.r.t salesprice

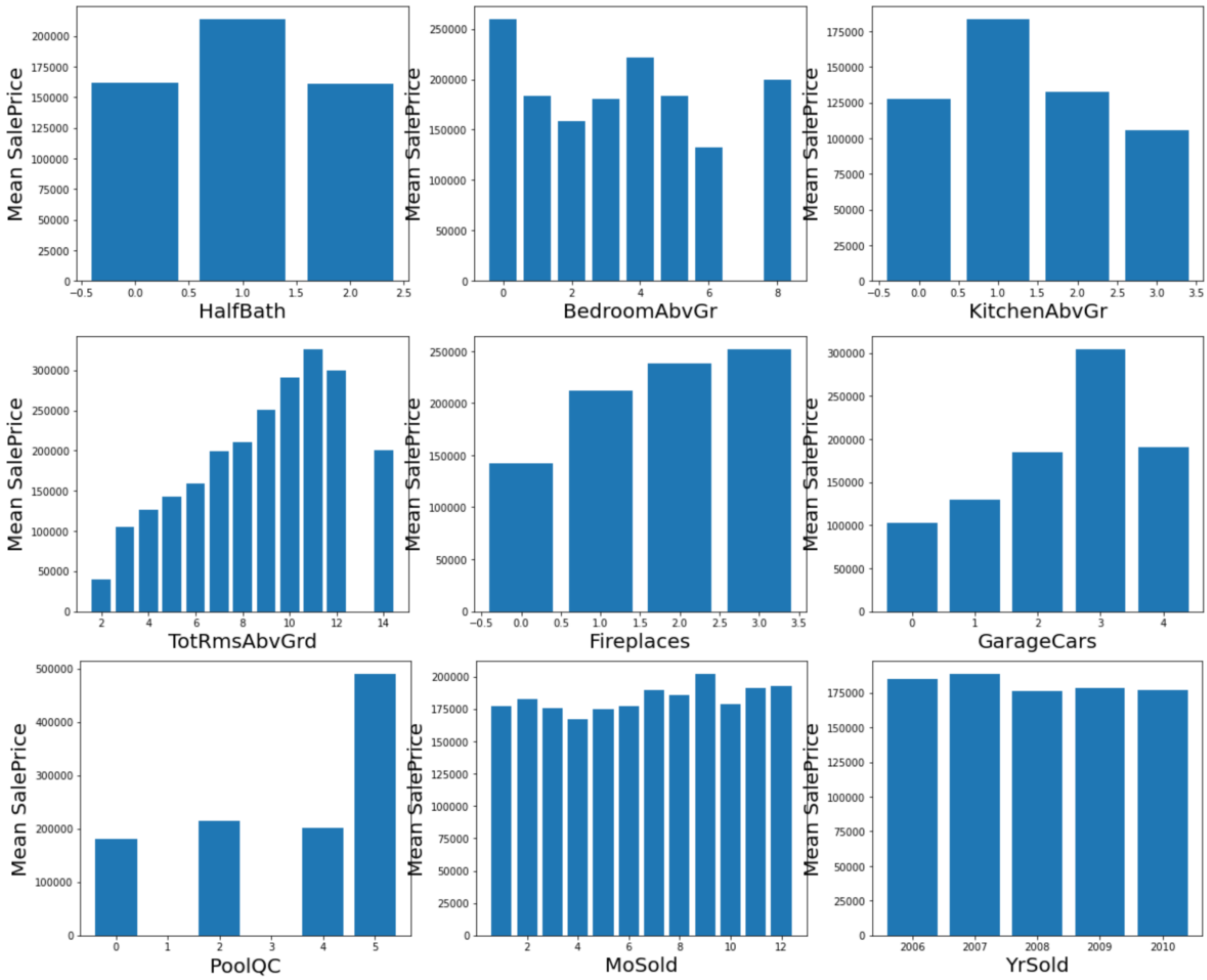
Plot graph showing categorical columns value point and the respective mean value of salesprice for that category




Mean SalePrice
RoofMatl
Mean SalePrice
Exterior1st
Mean SalePrice
Exterior2nd
Mean SalePrice
MasVnrType
Mean SalePrice
ExterQual
Mean SalePrice
ExterCond
Mean SalePrice
Foundation
Mean SalePrice
BsmtQual
Mean SalePrice
BsmtCond
Mean SalePrice
BsmtExposure
Mean SalePrice
BsmtFinType1
Mean SalePrice
BsmtFinType2


Mean SalePrice
Heating
HeatingQC
CentralAir
Electrical
KitchenQual
Functional
FireplaceQu
GarageType
GarageFinish
GarageQual
GarageCond
PavedDrive

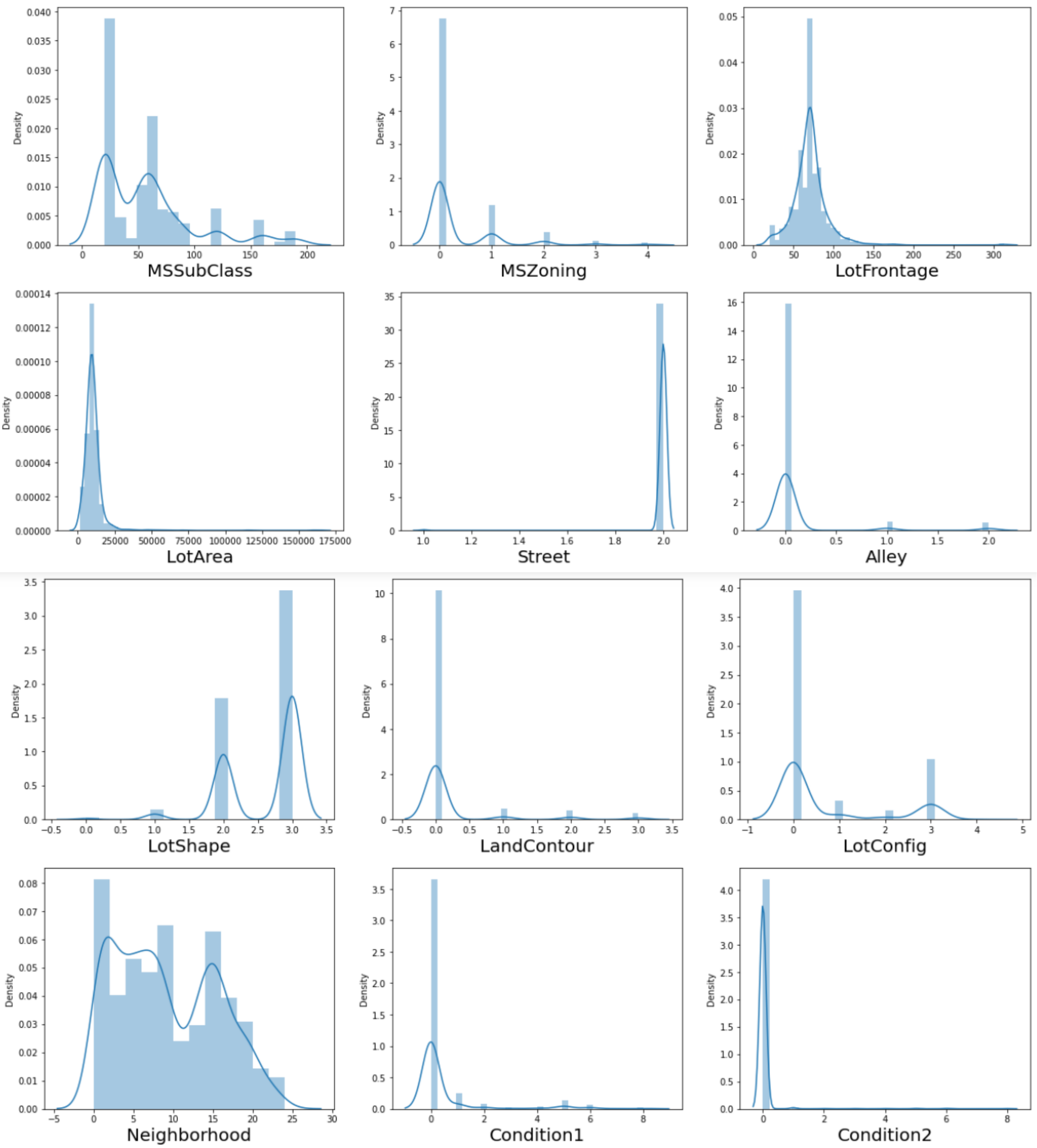

Mean SalePrice
PoolQC
Fence
MiscFeature
SaleType
SaleCondition
Alley
OverallQual
OverallCond
MSSubClass
BsmtFullBath
BsmtHalfBath
FullBath

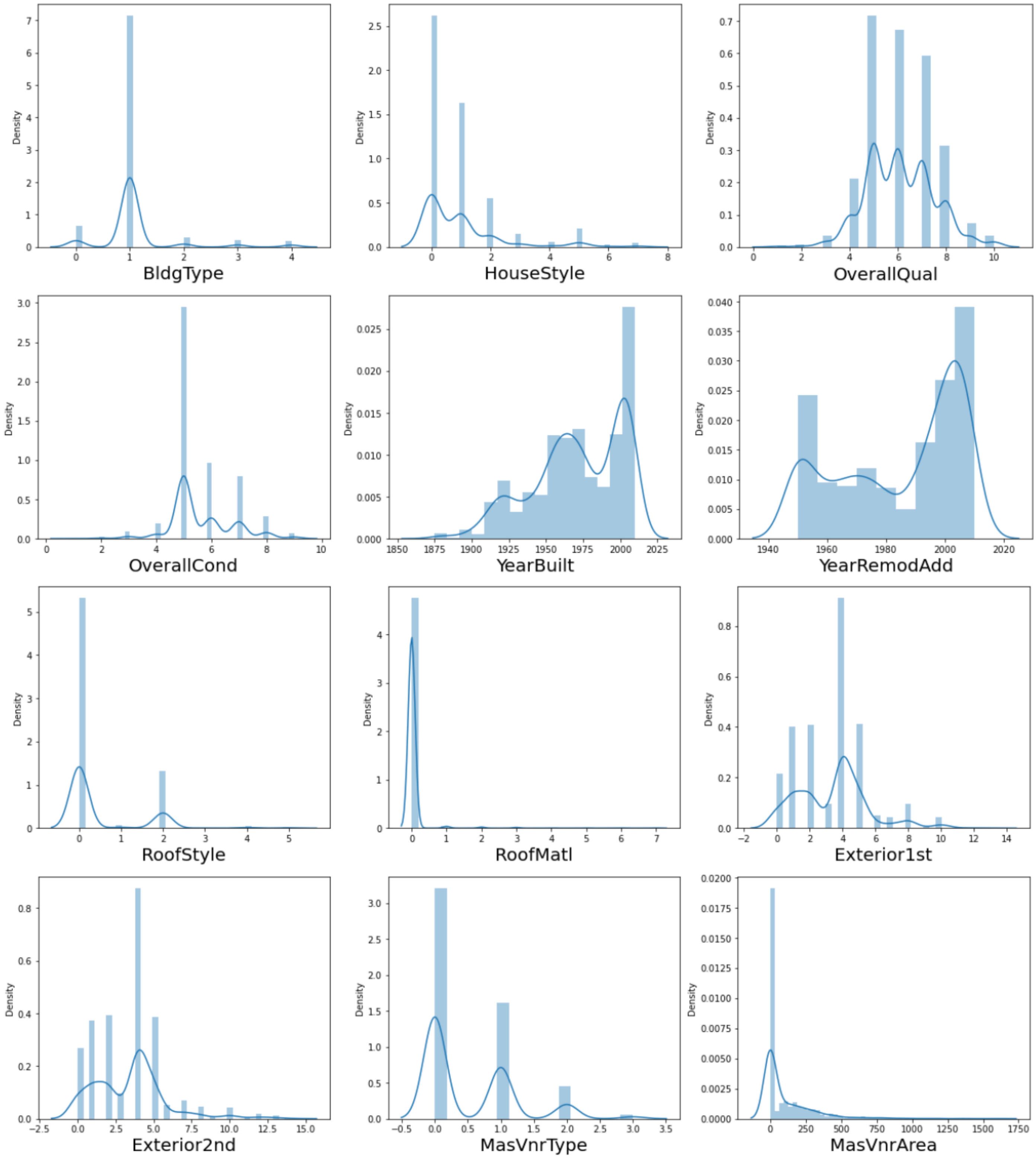


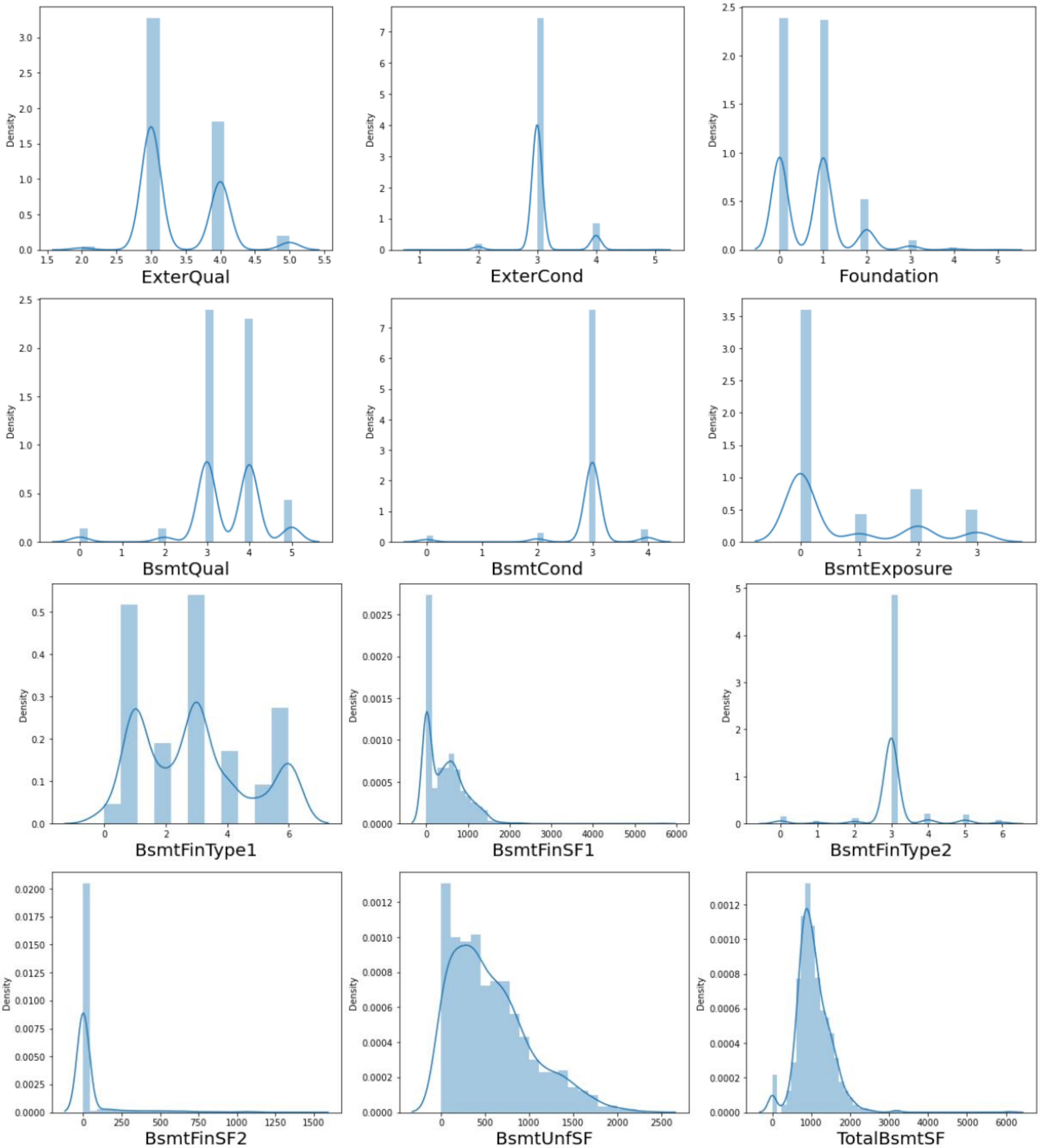
Observations:

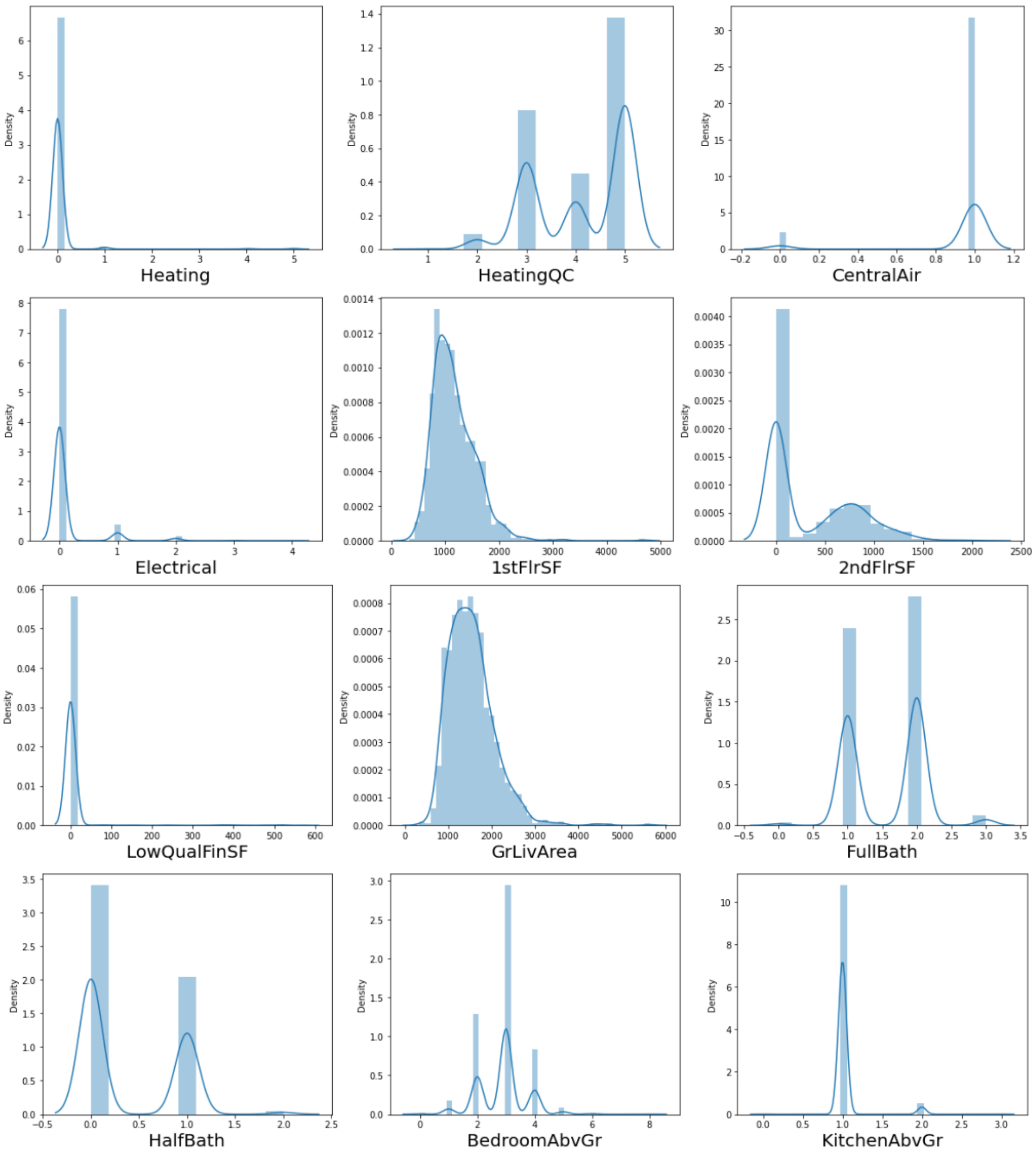
From above we can observe that categorical datapoints of columns LandSlope, MoSold, YrSold, BsmtFullBath, BsmtHalfBath have equal mean value of salesprice throughout their respective category, hence drop them.

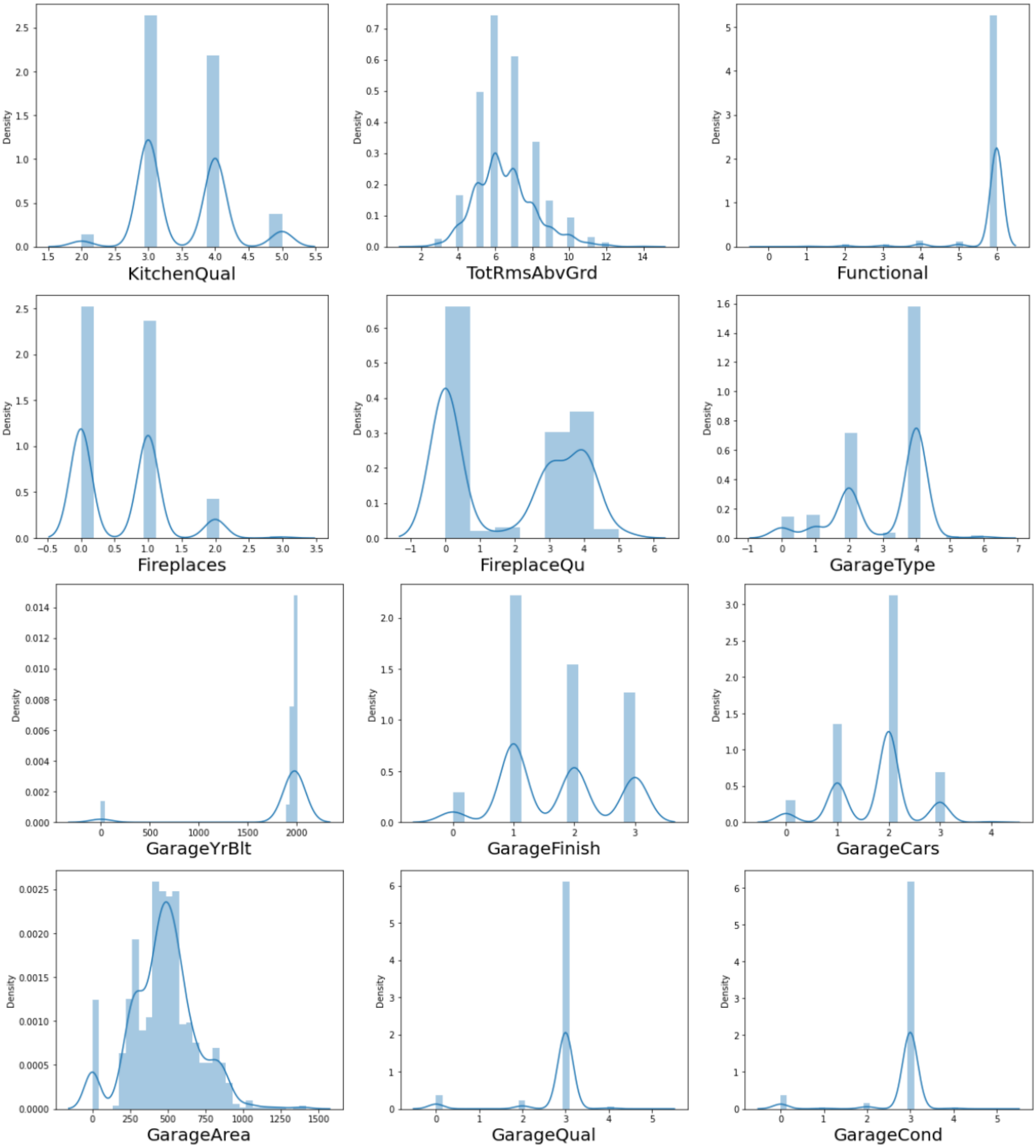
Plot Distribution plot of each column

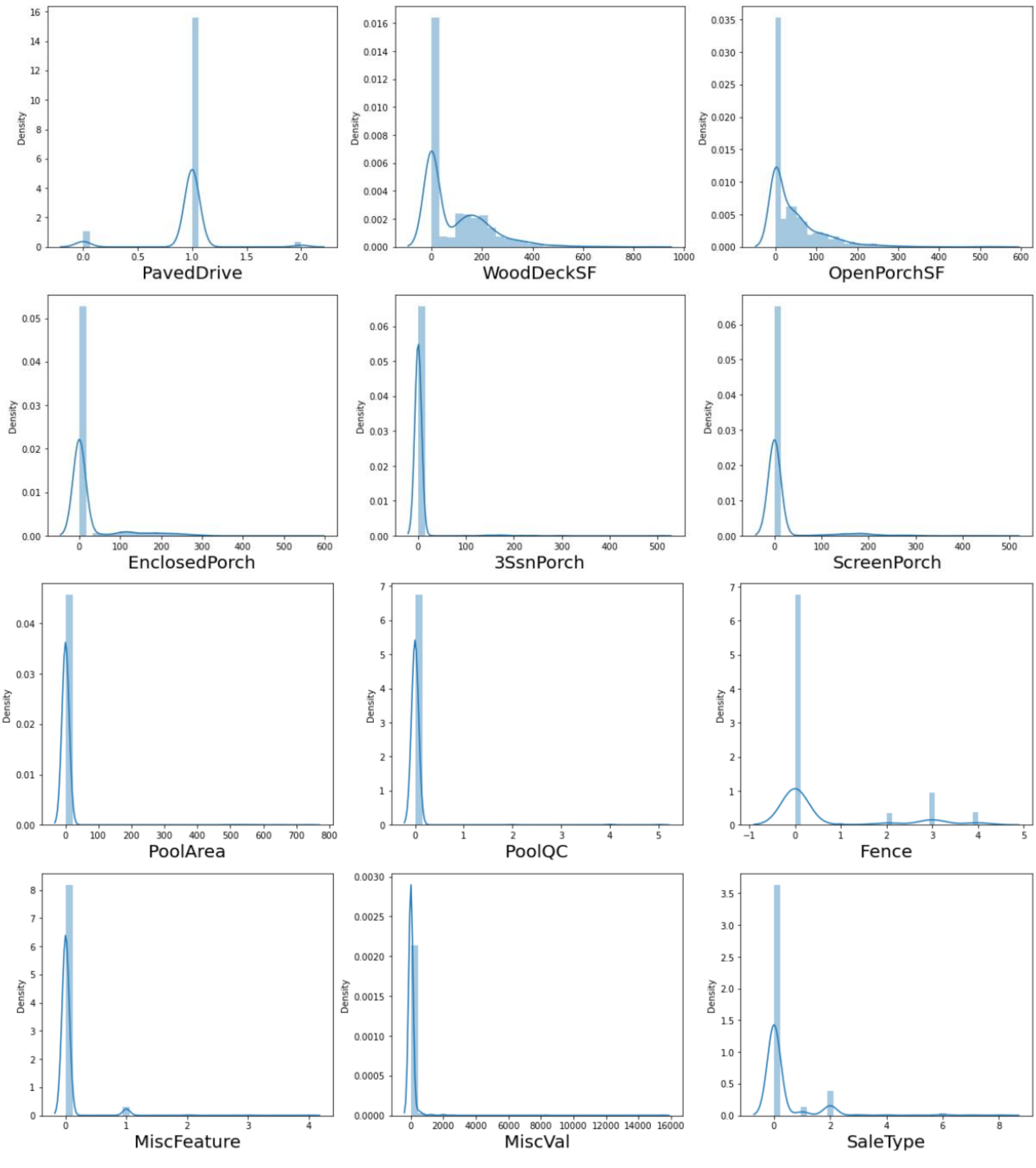


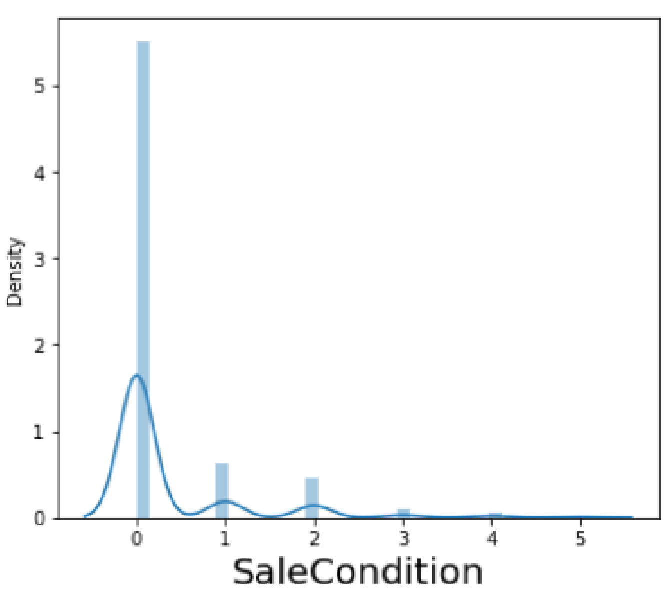

Density
BldgType
HouseStyle
OverallQual
OverallCond
YearBuilt
YearRemodAdd
RoofStyle
RoofMatl
Exterior1st
Exterior2nd
MasVnrType
MasVnrArea

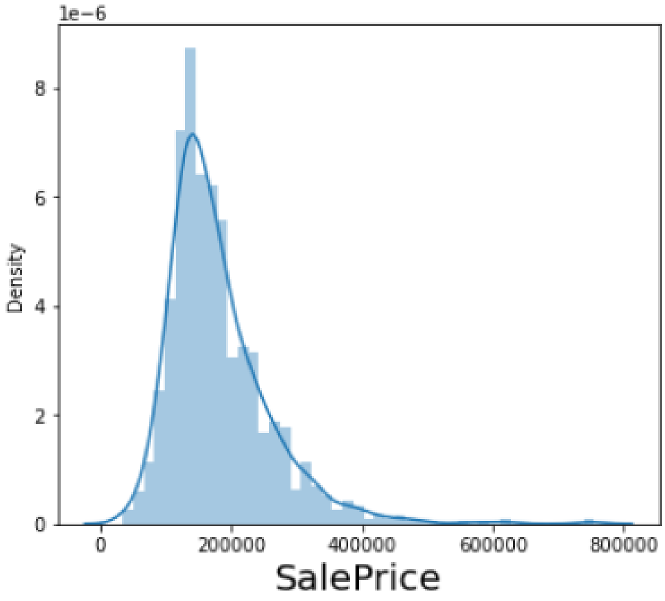

Density
ExterQual
ExterCond
Foundation
BsmtQual
BsmtCond
BsmtExposure
BsmtFinType1
BsmtFinSF1
BsmtFinType2
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF


Density
Heating
HeatingQC
CentralAir
Electrical
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr


Density
KitchenQual
TotRmsAbvGrd
Functional
Fireplaces
FireplaceQu
GarageType
GarageYrBlt
GarageFinish
GarageCars
GarageArea
GarageQual
GarageCond


PavedDrive
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
PoolQC
Fence
MiscFeature
MiscVal
SaleType
Density





Observation:

Columns having normal distribution plots are:

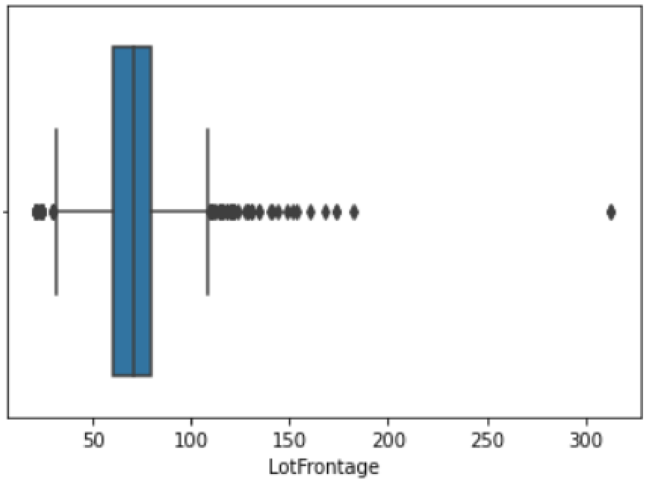
- LotFrontage
- Alley
- LandContour
- Condition1
- Conditional2
- BldgType
- RoofMatl
- MasVnrArea
- BsmtCond
- BsmtFinType2
- BsmtFinSF2
- LotArea
- BsmtUnfSF
- TotalBsmtSF
- Heating
- CentralAir
- Electrical
- 1stFlrSF
- LowQualFinSF
- GrLivArea
- KitchenAbvGr
- Functional
- GarageYrBlt
- Street
- GarageQual
- GarageCond
- PavedDrive
- OpenPorchSF

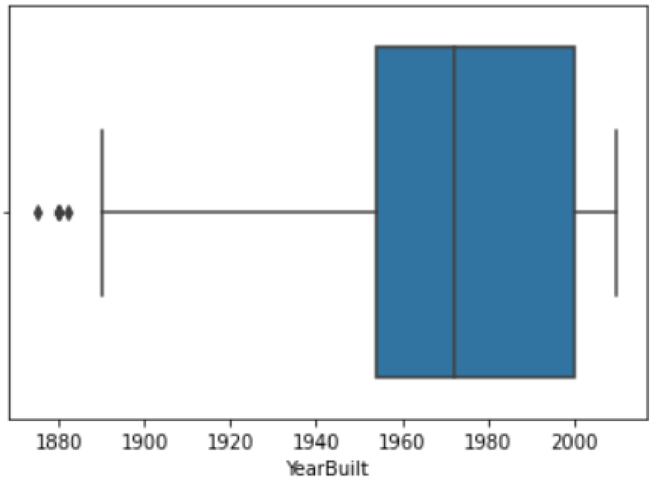
- EnclosedPorch
- 3SsnPorch
- ScreenPorch
- PoolArea
- PoolQC
- Fence
- MiscFeature
- MiscVal
- SaleType
- SalePrice

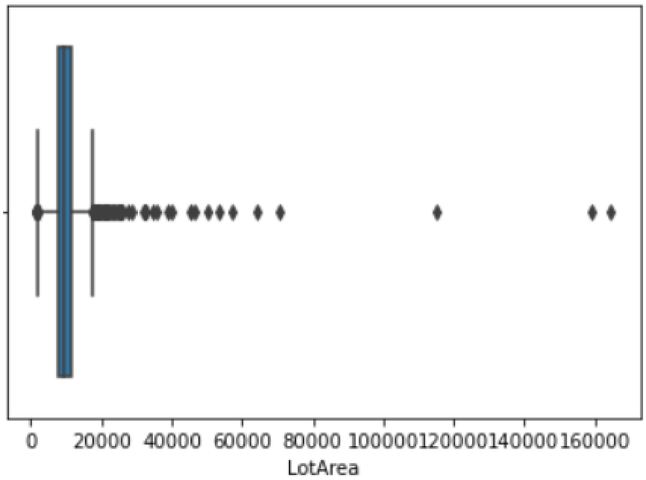
Rest all columns have bimodal type distribution plot

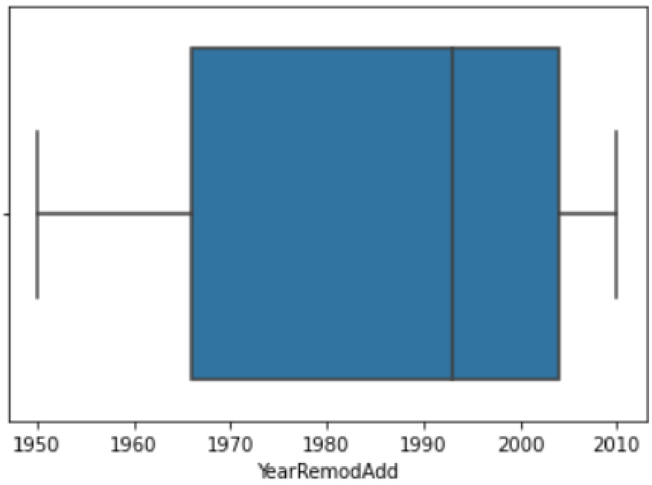
Check For Outliers

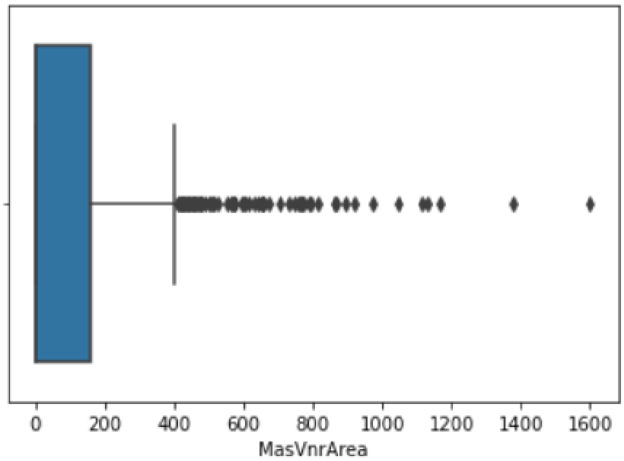
Visualize Boxplot of every column having datapoints of continuous nature

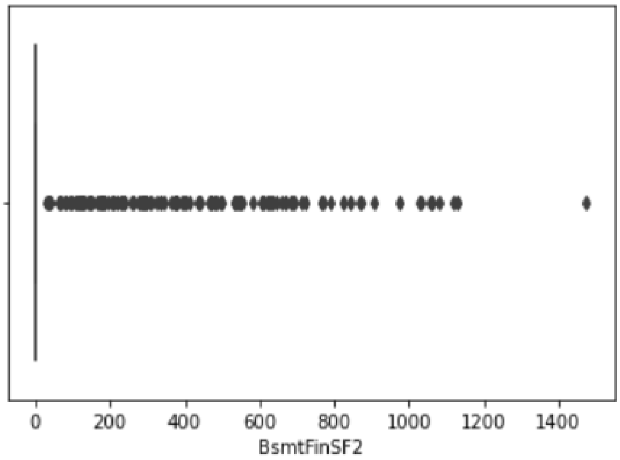


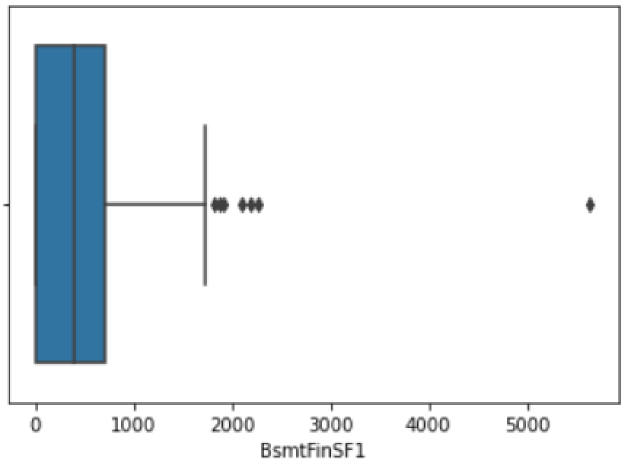


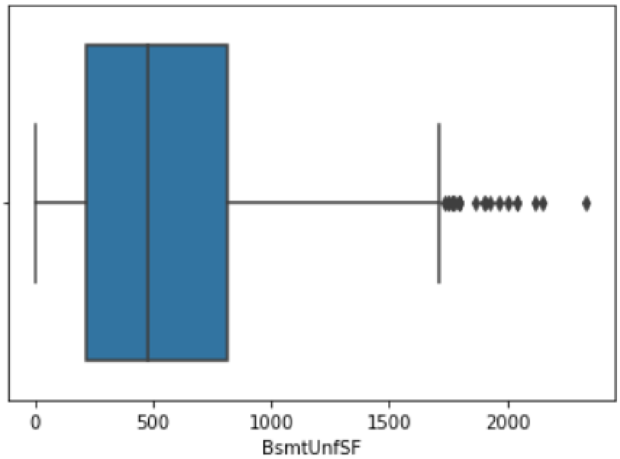


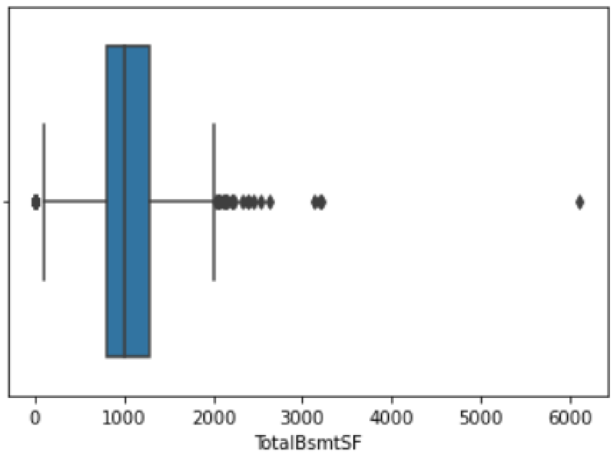


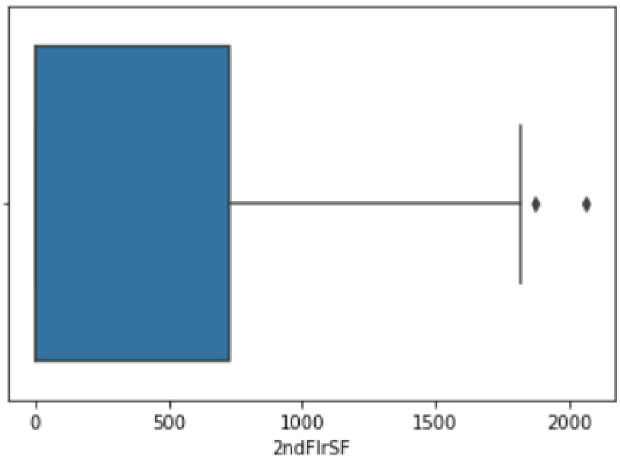

0 200 400 600 800 1000 1200 1400 1600
MasVnrArea

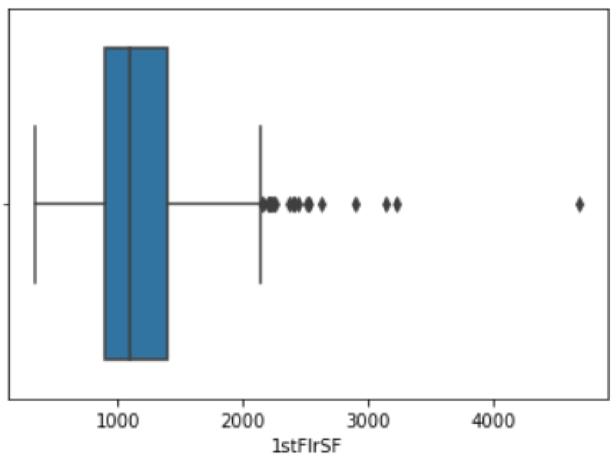

0 200 400 600 800 1000 1200 1400
BsmtFinSF2

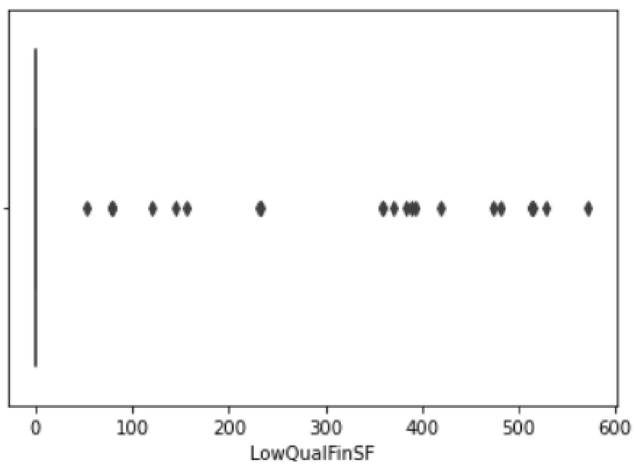

0 1000 2000 3000 4000 5000
BsmtFinSF1

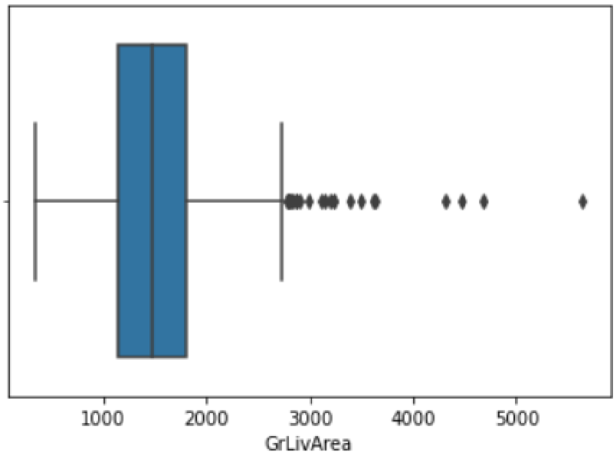

0 500 1000 1500 2000
BsmtUnfSF

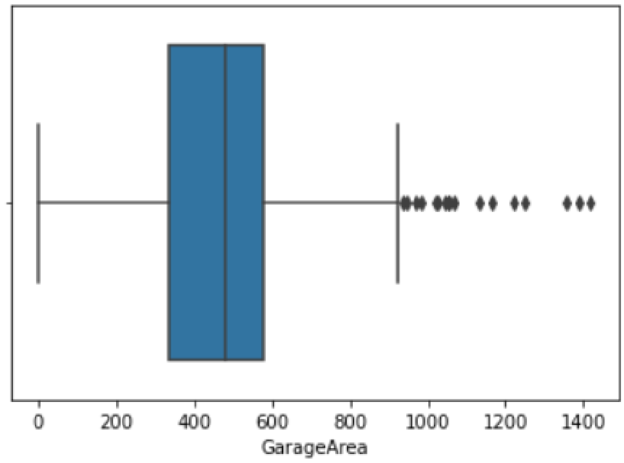

0 1000 2000 3000 4000 5000 6000
TotalBsmtSF

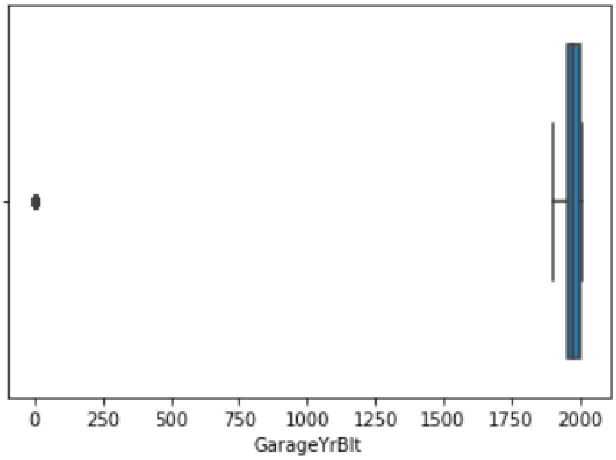

0 500 1000 1500 2000
2ndFlrSF

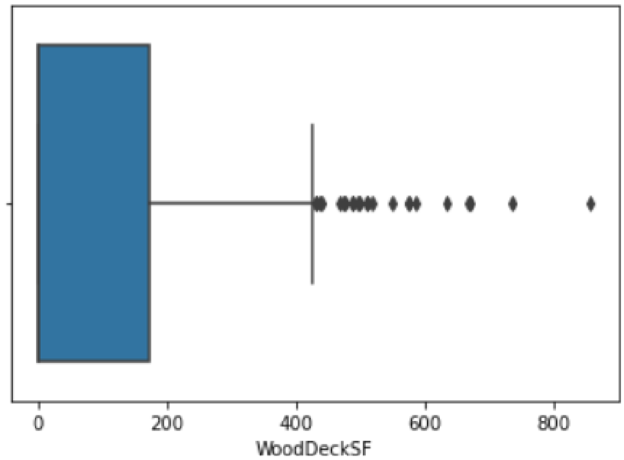

1000 2000 3000 4000
1stFlrSF

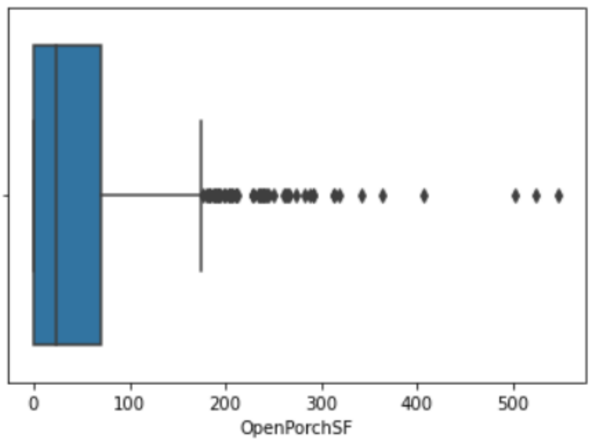

0 100 200 300 400 500 600
LowQualFinSF

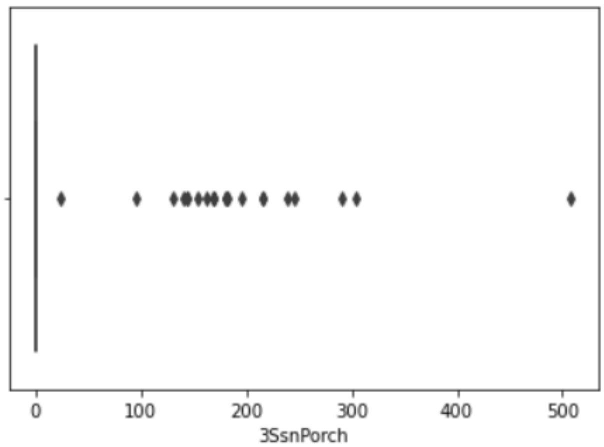

1000
2000
3000
4000
5000
GrLivArea

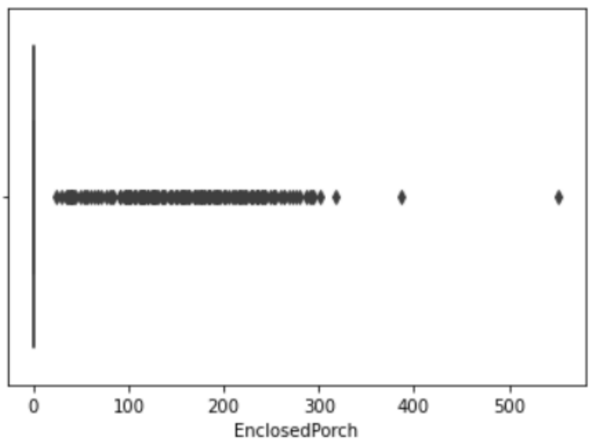

0
200
400
600
800
1000
1200
1400
GarageArea

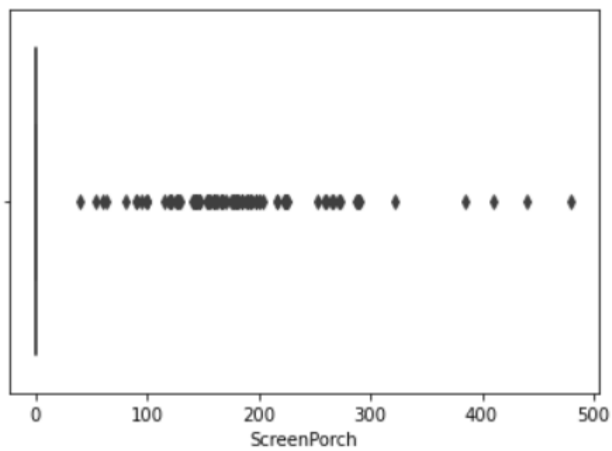

0
250
500
750
1000
1250
1500
1750
2000
GarageYrBlt

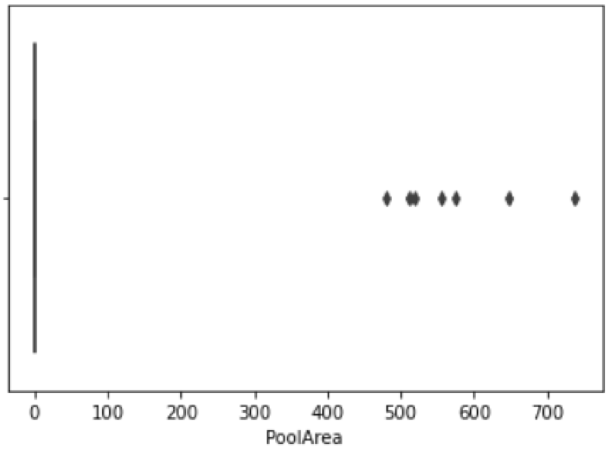

0
200
400
600
800
WoodDeckSF

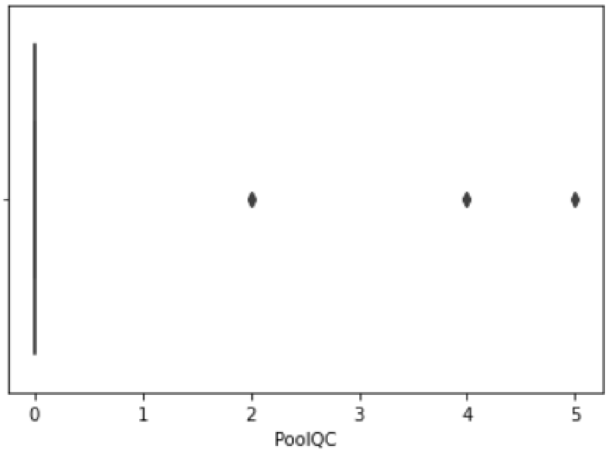

0
100
200
300
400
500
OpenPorchSF

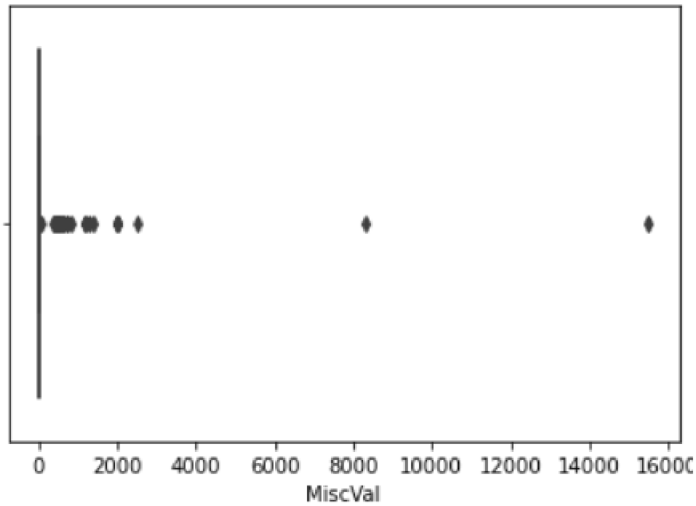

0
100
200
300
400
500
3SsnPorch


0
100
200
300
400
500
EnclosedPorch


0
100
200
300
400
500
ScreenPorch

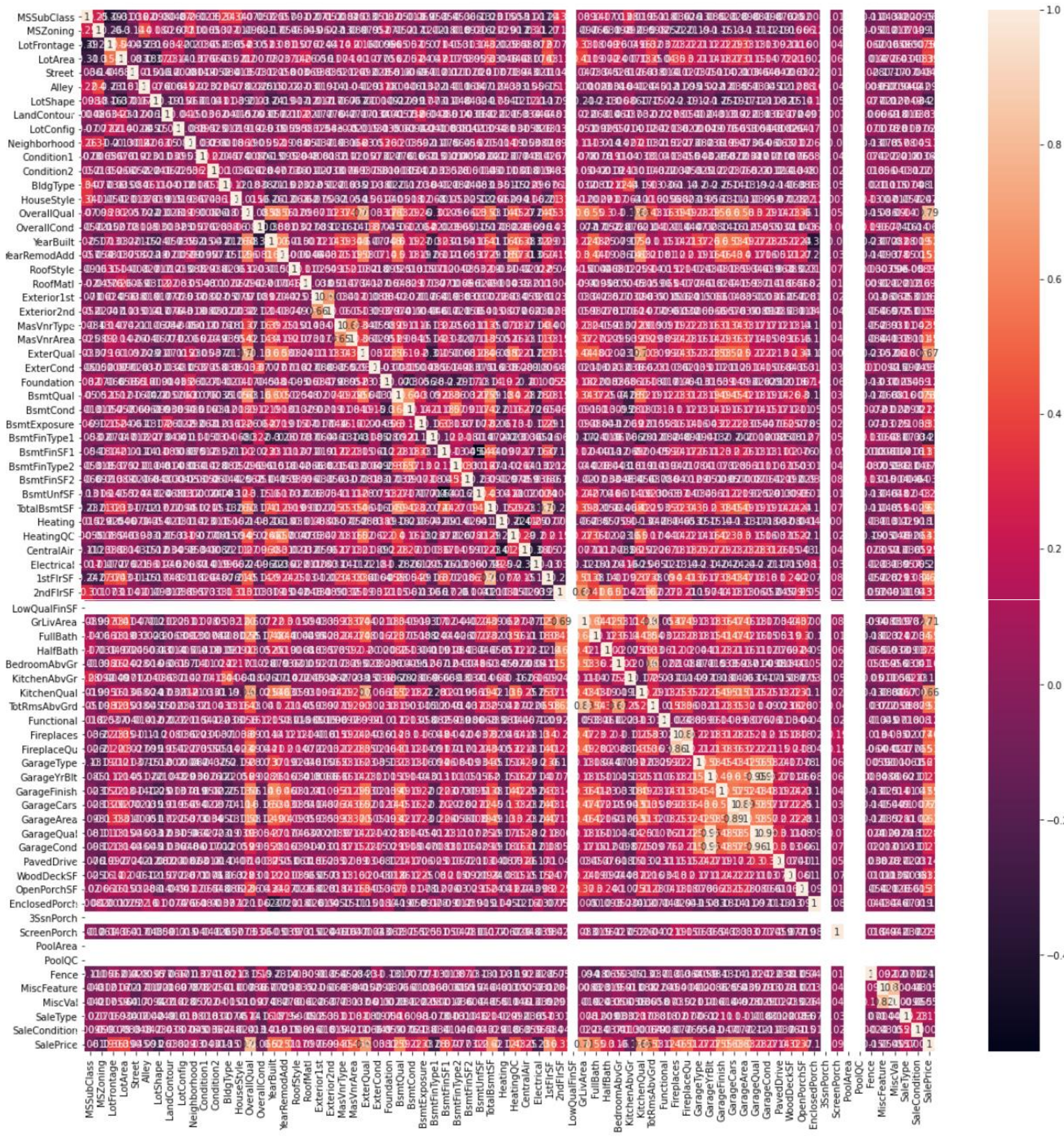






Outliers were detected in almost every column, remove them using percentile method

Check For Correlation



Observation:

Very few columns have high correlation with column SalePrice but further dropping of column is not possible as we cannot drop more than 10% of columns w.r.t the original dataset.

Model Building

Perform Feature Scaling

Before we start model building, we need to perform feature scaling on all columns, to avoid biasing of data.

Also check for skewness in data and remove it.

```
LotArea          1.191912
MasVnrArea       1.877102
BsmtFinSF1       0.639523
BsmtFinSF2       3.543134
BsmtUnfSF        0.777624
TotalBsmtSF      0.166773
1stFlrSF         0.645842
2ndFlrSF         0.717390
LowQualFinSF     0.000000
GrLivArea        0.592755
LotFrontage      0.187719
dtype: float64
8
```

```
LotArea          0.077862
MasVnrArea       0.438357
BsmtFinSF1      -0.418554
BsmtFinSF2       2.394737
BsmtUnfSF       -0.304290
TotalBsmtSF     -0.155420
1stFlrSF        -0.000731
2ndFlrSF         0.279883
LowQualFinSF     0.000000
GrLivArea       -0.005974
LotFrontage      0.088373
dtype: float64
1
```

Model Building

As we know, this is a regression problem we need to build a model using regression algorithm models.

First, we need to write a function which can find us best random state for train test split.

Then we shall iterate through all the models supporting regression algorithms to find the best models.

```
**********
GradientBoostingRegressor()
score 0.84202015532579
r2 0.817042062025452
diff 0.024978093300337956
mae 18491.01004254228
rmse 35728.1398169082
**********
NuSVR()
score -0.01638660649302821
r2 -0.008689780769930655
diff 0.007696825723097555
mae 57484.218333046425
rmse 83890.65538631662
**********
LinearRegression()
score 0.8018427485473225
r2 0.7716364558845128
diff 0.03020629266280972
mae 21650.755540611055
rmse 39916.1057061981
**********
Ridge()
score 0.8022439202493714
r2 0.7717781004919595
diff 0.030465819757411916
mae 21623.92812544445
rmse 39903.72461853671
**********
RidgeCV(alphas=array([ 0.1,  1. , 10. ]))
score 0.8052810050772298
r2 0.7725759243164028
diff 0.03270508076082701
mae 21429.939870235718
rmse 39833.915335579455
```

```
**********
BayesianRidge()
score 0.8107655886398216
r2 0.7734656691261194
diff 0.03729991951370226
mae 21175.833242136607
rmse 39755.91842168027
**********
SGDRegressor()
score 0.798033691154122
r2 0.7696370802710762
diff 0.028396610883045792
mae 21396.82728517374
rmse 40090.46227077579
**********
SVR()
score -0.05498952035287763
r2 -0.038953484001147176
diff 0.016036036351730454
mae 57124.51260057353
rmse 85139.83980752791
**********
AdaBoostRegressor()
score 0.8002951100621216
r2 0.7298258513602652
diff 0.07046925870185639
mae 25362.19737536402
rmse 43416.68619858223
**********
LinearSVR()
score -5.475783044568629
r2 -4.646418602464995
diff 0.8293644421036337
mae 180099.4332129291
rmse 198482.08468133438
```

```
**********
KNeighborsRegressor()
score 0.7504654468076735
r2 0.7570132491766333
diff 0.006547802368959799
mae 24228.020512820516
rmse 41174.285702271845
**********
RandomForestRegressor()
score 0.8395767611095944
r2 0.7877437306781712
diff 0.05183303043142329
mae 19383.014017094014
rmse 38482.65613968785
**********
BaggingRegressor()
score 0.8195482347886817
r2 0.7402572060350301
diff 0.07929102875365157
mae 21455.223931623932
rmse 42570.28235396726
**********
DecisionTreeRegressor()
score 0.6269853510812382
r2 0.571965174215431
diff 0.05502017686580718
mae 29763.594017094016
rmse 54648.00720536014
**********
LGBMRegressor()
score 0.8530894222284748
r2 0.786645022773341
diff 0.0664443994551338
mae 19818.048194043058
rmse 38582.126992681966
```

```
ExtraTreesRegressor()
score 0.8572843830878465
r2 0.794359233709508З
diff 0.06292514937833815
mae 19272.629316239316
rmse 37878.20442726365
XGBRFRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
               colsample_bytree=1, gamma=0, gpu_id=-1, importance_type='gain',
               interaction_constraints='', max_delta_step=0, max_depth=6,
               min_child_weight=1, missing=nan, monotone_constraints='()',
               n_estimators=100, n_jobs=8, num_parallel_tree=100,
               objective='reg:squarederror', random_state=0, reg_alpha=0,
               scale_pos_weight=1, tree_method='exact', validate_parameters=1,
               verbosity=None)
score 0.8177052093979029
r2 0.7795861269010174
diff 0.03811908249688556
mae 21243.909455128207
rmse 39215.182442261415
**********
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.300000012, max_delta_step=0, max_depth=6,
             min_child_weight=1, missing=nan, monotone_constraints='()',
             n_estimators=100, n_jobs=8, num_parallel_tree=1, random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
             tree_method='exact', validate_parameters=1, verbosity=None)
score 0.8294024669431674
r2 0.7711003857562838
diff 0.058302081186883625
mae 20288.5914463141
rmse 39962.92861123615
**********
LogisticRegression()
score 0.5548319963186098
r2 0.6081085027805024
diff 0.053276506461892637
mae 32626.581196581195
rmse 52289.88626679993
```

From above we get to know that the top 5 models are:

- ExtratreesRegressor
- LGMBRegressor
- GradientboostingRegressor
- RandomForestRegressor
- XGBRegressor

Fine tune all these models and find their best parameters to use.

Next, find the best random state for train test split.

As we know from above output that our top models do not have accuracy above 90%, hence we will stack our top 5 models to build one model to obtain higher accuracy.

To stack models, we must use StackCVRegressor to combine all our fine tune models.

After using StackCVRegressor we obtain test accuracy of more than 90%.

CV score of this model is more than 87%.

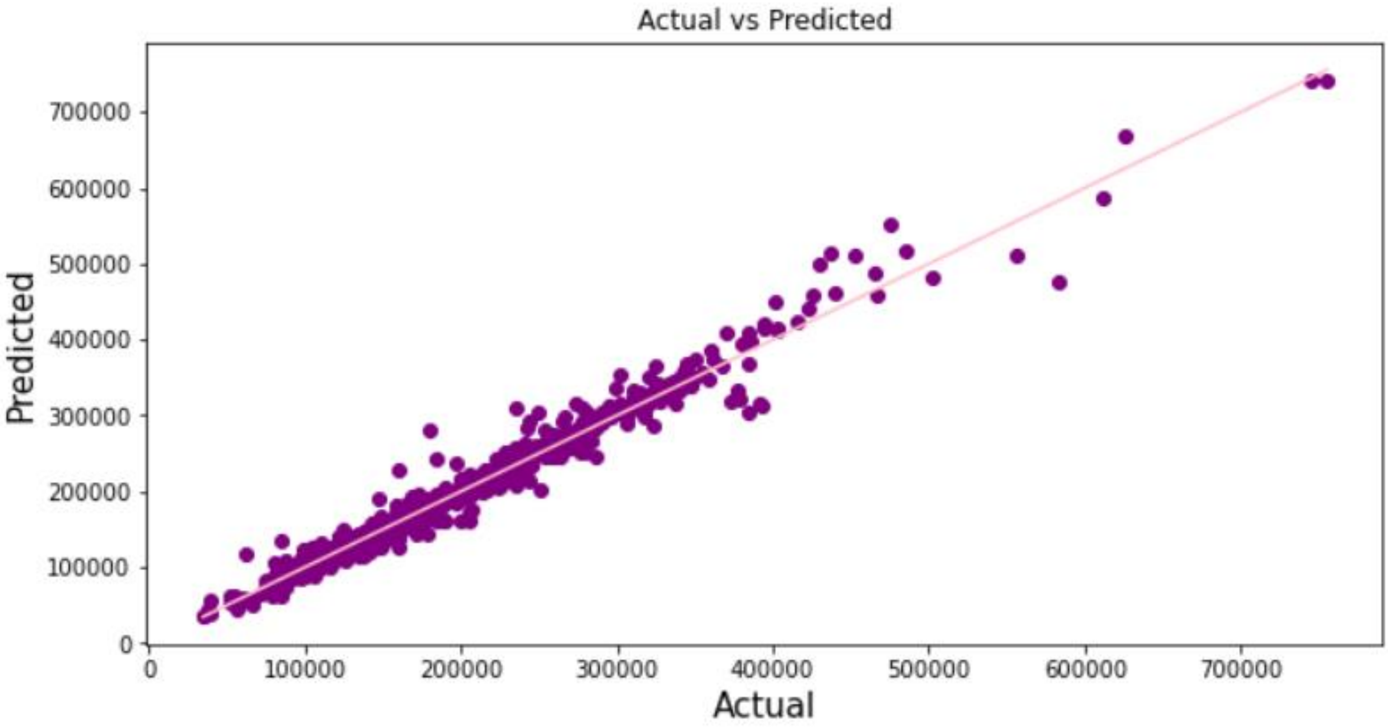
To analyze our model, we shall find the difference between actual and predicted value.

Value difference between actual and predicted value when actual value is greater than predicted value is 89259

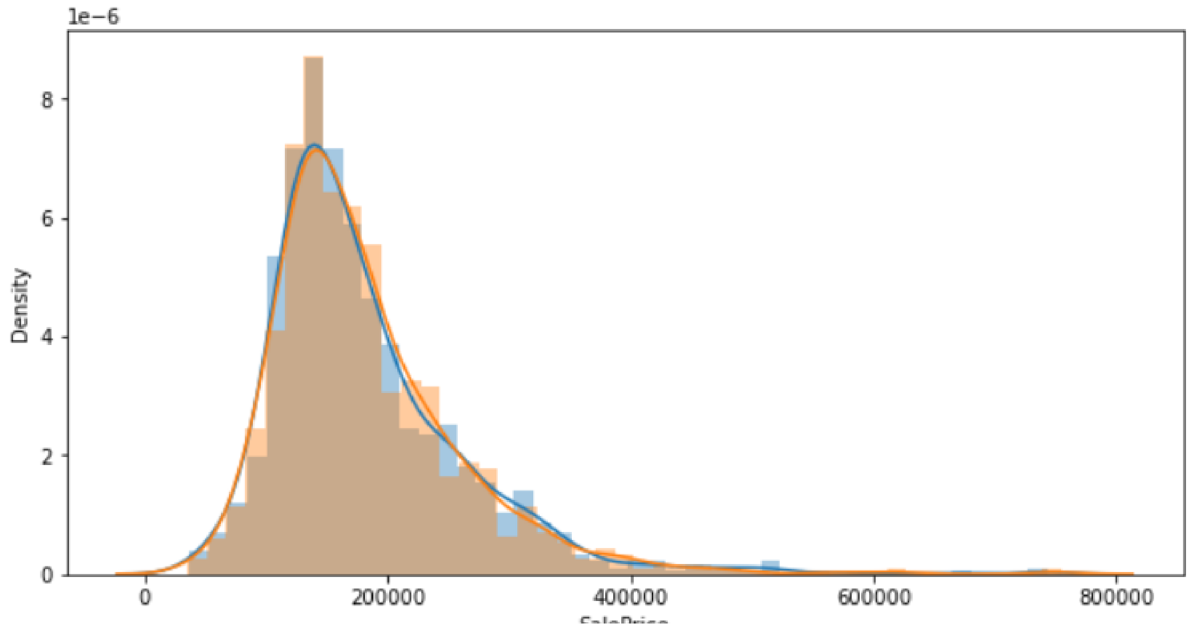
Value difference between actual and predicted value when actual value is less than predicted value is 95990

Avg difference between actual and predicted value is 193

Regression Plot of Actual vs Predicted value



Compare Distribution plot of Actual vs Predicted Value



Observation:

From above 2 plots we can observe the closeness between actual and predicted value.

Hence, we can verify that the model built is able to predict SalePrice with a high accuracy.

Save model for further use.

Perform the same data processing steps for test dataset & predict the value for target variable using the existing Stacked Model

Pred
405046
364184
377898
328445
346637
...
333210
325389
333114
332363
306176

Save this dataset in a CSV file

Interpretation of the Results

Here we check the correlation between all our feature variables with target variable label

1. The column OverallQual is most positively correlated with SalePrice.

2. The column KitchenAbvGrd is most negatively correlated with SalePrice.

Set of assumptions related to the problem under consideration

By looking into the target variable datapoints we assumed that it was a Regression type of problem.

We also observed that only one single unique value was present in Utilities column so we assumed that we will be dropping these columns. ID column was also dropped as it contained all unique vales.

Columns such as LandSlope, MoSold, YrSold, BsmtFullBath, BsmtHalfBath have equal mean value of salesprice throughout their respective category which basically concludes that whatever the datapoint is SalePrice is not affected by it, hence drop them also.

Outliers were removed using percentile method. Skewness was reduced using Yeo-Johnson method.

Final model built is actually a combination of top 5 fine tuned model to achieve high accuracy.

Conclusion

Key Features and conclusion of the study

In this project we have tried to show how the house prices vary and what are the factors related to the changing of house prices. The best accuracy score was achieved by stacking our top 5 fine-tuned models.

LEARNING OUTCOMES OF THE STUDY IN RESPECT OF DATA SCIENCE

Through different powerful tools of visualization, we were able to analyze and interpret different hidden insights about the data.

Through data cleaning we were able to remove unnecessary columns and outliers from our dataset due to which our model would have suffered from overfitting or underfitting.

The data was improper scaled, so we scaled it to a single scale using sklearns's package StandardScaler.

There were lot of missing values present in different columns which we imputed on the basis of our understanding.

The columns were skewed due to presence of outliers which we handled through percentile technique.

Stacked Model was then built having accuracy more than 90% using train dataset.

Finally, we predicted the SalePrice for Test dataset using our stacked model and saved the data frame into a CSV file.

LIMITATIONS OF THIS WORK AND SCOPE FOR FUTURE WORK

While we couldn’t reach out goal of minimum RMSE in house price prediction without letting the model to overfit, we did end up creating a system that can with enough time and data get very close to that goal.

As with any project there is room for improvement here.

The very nature of this project allows for multiple algorithms to be integrated together as modules and their results can be combined to increase the accuracy of the final result.

This model can further be improved with the addition of more algorithms into it.

However, the output of these algorithms needs to be in the same format as the others.

Once that condition is satisfied, the modules are easy to add as done in the code.

This provides a great degree of modularity and versatility to the project.