Problem Statement:

Houses are one of the necessary need of each and every 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. 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. For the same purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below.

The company is looking at prospective properties to buy houses to enter the market. You 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. For this company wants to know:

• Which variables are important to predict the price of variable?

• How do these variables describe the price of the house?

Business Goal:

You are required to model the price of houses with the available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.

Technical Requirements:

• Data contains 1460 entries each having 81 variables. • Data contains Null values. You need to treat them using the domain knowledge and your own understanding. • Extensive EDA has to be performed to gain relationships of important variable and price. • Data contains numerical as well as categorical variable. You need to handle them accordingly. • You have to build Machine Learning models, apply regularization and determine the optimal values of Hyper Parameters. • You need to find important features which affect the price positively or negatively. • Two datasets are being provided to you (test.csv, train.csv). You will train on train.csv dataset and predict on test.csv file.

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import scipy
from scipy import stats
from scipy.stats import zscore
import sklearn

from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
```

```python
from sklearn.preprocessing import StandardScaler

from sklearn import linear_model
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.datasets import make_regression
from sklearn import metrics
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```python
df=pd.read_csv("train.csv")
df
```

Out[2]:

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

1168 rows × 81 columns

Dataset Feature Description :

In [3]:
```python
print('No. of Rows :',df.shape[0])
print('No. of Columns :', df.shape[1])
pd.set_option('display.max_columns',None) # this will enable us to see truncated columns
df.head()
```

```
No. of Rows : 1168
No. of Columns : 81
```

Out[3]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	Inside
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	Inside
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	CulDSac
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	Inside
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	FR2

In [4]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1168 entries, 0 to 1167
Data columns (total 81 columns):
 #   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
 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)
memory usage: 739.2+ KB
```

In [5]:
```
df.describe().T
```

Out[5]:

	count	mean	std	min	25%	50%	75%	max
Id	1168.0	724.136130	416.159877	1.0	360.50	714.5	1079.5	1460.0
MSSubClass	1168.0	56.767979	41.940650	20.0	20.00	50.0	70.0	190.0
LotFrontage	954.0	70.988470	24.828750	21.0	60.00	70.0	80.0	313.0
LotArea	1168.0	10484.749144	8957.442311	1300.0	7621.50	9522.5	11515.5	164660.0
OverallQual	1168.0	6.104452	1.390153	1.0	5.00	6.0	7.0	10.0
OverallCond	1168.0	5.595890	1.124343	1.0	5.00	5.0	6.0	9.0
YearBuilt	1168.0	1970.930651	30.145255	1875.0	1954.00	1972.0	2000.0	2010.0
YearRemodAdd	1168.0	1984.758562	20.785185	1950.0	1966.00	1993.0	2004.0	2010.0
MasVnrArea	1161.0	102.310078	182.595606	0.0	0.00	0.0	160.0	1600.0
BsmtFinSF1	1168.0	444.726027	462.664785	0.0	0.00	385.5	714.5	5644.0
BsmtFinSF2	1168.0	46.647260	163.520016	0.0	0.00	0.0	0.0	1474.0
BsmtUnfSF	1168.0	569.721747	449.375525	0.0	216.00	474.0	816.0	2336.0
TotalBsmtSF	1168.0	1061.095034	442.272249	0.0	799.00	1005.5	1291.5	6110.0
1stFlrSF	1168.0	1169.860445	391.161983	334.0	892.00	1096.5	1392.0	4692.0
2ndFlrSF	1168.0	348.826199	439.696370	0.0	0.00	0.0	729.0	2065.0
LowQualFinSF	1168.0	6.380137	50.892844	0.0	0.00	0.0	0.0	572.0
GrLivArea	1168.0	1525.066781	528.042957	334.0	1143.25	1468.5	1795.0	5642.0
BsmtFullBath	1168.0	0.425514	0.521615	0.0	0.00	0.0	1.0	3.0
BsmtHalfBath	1168.0	0.055651	0.236699	0.0	0.00	0.0	0.0	2.0
FullBath	1168.0	1.562500	0.551882	0.0	1.00	2.0	2.0	3.0
HalfBath	1168.0	0.388699	0.504929	0.0	0.00	0.0	1.0	2.0
BedroomAbvGr	1168.0	2.884418	0.817229	0.0	2.00	3.0	3.0	8.0

KitchenAbvGr	1168.0	1.045377	0.216292	0.0	1.00	1.0	1.0	3.0
TotRmsAbvGrd	1168.0	6.542808	1.598484	2.0	5.00	6.0	7.0	14.0
Fireplaces	1168.0	0.617295	0.650575	0.0	0.00	1.0	1.0	3.0
GarageYrBlt	1104.0	1978.193841	24.890704	1900.0	1961.00	1980.0	2002.0	2010.0
GarageCars	1168.0	1.776541	0.745554	0.0	1.00	2.0	2.0	4.0
GarageArea	1168.0	476.860445	214.466769	0.0	338.00	480.0	576.0	1418.0
WoodDeckSF	1168.0	96.206336	126.158988	0.0	0.00	0.0	171.0	857.0
OpenPorchSF	1168.0	46.559932	66.381023	0.0	0.00	24.0	70.0	547.0
EnclosedPorch	1168.0	23.015411	63.191089	0.0	0.00	0.0	0.0	552.0
3SsnPorch	1168.0	3.639555	29.088867	0.0	0.00	0.0	0.0	508.0
ScreenPorch	1168.0	15.051370	55.080816	0.0	0.00	0.0	0.0	480.0
PoolArea	1168.0	3.448630	44.896939	0.0	0.00	0.0	0.0	738.0
MiscVal	1168.0	47.315068	543.264432	0.0	0.00	0.0	0.0	15500.0
MoSold	1168.0	6.344178	2.686352	1.0	5.00	6.0	8.0	12.0
YrSold	1168.0	2007.804795	1.329738	2006.0	2007.00	2008.0	2009.0	2010.0
SalePrice	1168.0	181477.005993	79105.586863	34900.0	130375.00	163995.0	215000.0	755000.0

Data Integrity check

checking for duplicates

In [6]:
```
df.duplicated().sum()
```

Out[6]:
```
0
```

In [7]:
```
df.columns[df.isnull().any()]
```

Out[7]:
```
Index(['LotFrontage', 'Alley', 'MasVnrType', 'MasVnrArea', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
       'FireplaceQu', 'GarageType', 'GarageYrBlt', 'GarageFinish',
       'GarageQual', 'GarageCond', 'PoolQC', 'Fence', 'MiscFeature'],
      dtype='object')
```

Checking for white spaces

In [8]:
```
df.isin(['NA','N/A','-',' ','?',' ?']).sum().any()
```

Out[8]:
```
False
```

Checking for null values

In [9]:
```
df.columns[df.isnull().any()]
```

Out[9]:
```
Index(['LotFrontage', 'Alley', 'MasVnrType', 'MasVnrArea', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
       'FireplaceQu', 'GarageType', 'GarageYrBlt', 'GarageFinish',
```

```
      'GarageQual', 'GarageCond', 'PoolQC', 'Fence', 'MiscFeature'],
     dtype='object')
```

These are the columns with null values.

In [10]:
```
# Finding what percentage of data is missing from dataset
pd.set_option('display.max_rows',None)
missing_values = df.isnull().sum().sort_values(ascending = False)
percentage_missing_values = (missing_values/len(df))*100
print(pd.concat([missing_values, percentage_missing_values], axis = 1, keys =['Missing V
```

```
               Missing Values  % Missing data
PoolQC                   1161       99.400685
MiscFeature              1124       96.232877
Alley                    1091       93.407534
Fence                     931       79.708904
FireplaceQu               551       47.174658
LotFrontage               214       18.321918
GarageYrBlt                64        5.479452
GarageFinish               64        5.479452
GarageType                 64        5.479452
GarageQual                 64        5.479452
GarageCond                 64        5.479452
BsmtExposure               31        2.654110
BsmtFinType2               31        2.654110
BsmtQual                   30        2.568493
BsmtCond                   30        2.568493
BsmtFinType1               30        2.568493
MasVnrType                  7        0.599315
MasVnrArea                  7        0.599315
Id                          0        0.000000
Functional                  0        0.000000
Fireplaces                  0        0.000000
KitchenQual                 0        0.000000
KitchenAbvGr                0        0.000000
BedroomAbvGr                0        0.000000
HalfBath                    0        0.000000
FullBath                    0        0.000000
BsmtHalfBath                0        0.000000
BsmtFullBath                0        0.000000
TotRmsAbvGrd                0        0.000000
GarageCars                  0        0.000000
LowQualFinSF                0        0.000000
GarageArea                  0        0.000000
PavedDrive                  0        0.000000
WoodDeckSF                  0        0.000000
OpenPorchSF                 0        0.000000
EnclosedPorch               0        0.000000
3SsnPorch                   0        0.000000
ScreenPorch                 0        0.000000
PoolArea                    0        0.000000
MiscVal                     0        0.000000
MoSold                      0        0.000000
YrSold                      0        0.000000
SaleType                    0        0.000000
SaleCondition               0        0.000000
GrLivArea                   0        0.000000
HeatingQC                   0        0.000000
2ndFlrSF                    0        0.000000
LandSlope                   0        0.000000
OverallQual                 0        0.000000
HouseStyle                  0        0.000000
BldgType                    0        0.000000
Condition2                  0        0.000000
Condition1                  0        0.000000
```

```
Neighborhood            0        0.000000
LotConfig               0        0.000000
YearBuilt               0        0.000000
Utilities               0        0.000000
LandContour             0        0.000000
LotShape                0        0.000000
Street                  0        0.000000
LotArea                 0        0.000000
MSZoning                0        0.000000
OverallCond             0        0.000000
YearRemodAdd            0        0.000000
1stFlrSF                0        0.000000
BsmtFinSF2              0        0.000000
Electrical              0        0.000000
CentralAir              0        0.000000
MSSubClass              0        0.000000
Heating                 0        0.000000
TotalBsmtSF             0        0.000000
BsmtUnfSF               0        0.000000
BsmtFinSF1              0        0.000000
RoofStyle               0        0.000000
Foundation              0        0.000000
ExterCond               0        0.000000
ExterQual               0        0.000000
Exterior2nd             0        0.000000
Exterior1st             0        0.000000
RoofMatl                0        0.000000
SalePrice               0        0.000000
```

In [11]:
```
df.drop(columns = ['MiscFeature','PoolQC','Alley','Fence'],axis = 1, inplace = True)
df.shape
```

Out[11]:
```
(1168, 77)
```

Dropping the Columns which has more than 70% of Null Values.

Filling the Null Values.

In [12]:
```
df['MasVnrArea'] = df['MasVnrArea'].fillna(df['MasVnrArea'].mean())
```

In [13]:
```
df['LotFrontage'] = df['LotFrontage'].fillna(df['LotFrontage'].median())
df['GarageYrBlt'] = df['GarageYrBlt'].fillna(df['GarageYrBlt'].median())
```

In [14]:
```
for x in ['MasVnrType','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2
'GarageFinish','GarageQual','GarageCond']:
     df[x] = df[x].fillna(df[x].mode()[0])
```

In [15]:
```
df.isnull().sum().sum()
```

Out[15]:
```
0
```

Now, No null values present in our dataset.

In [16]:
```
# Value counts for each feature data
for i in df.columns:
    print(i,df[i].nunique())
    print('*****************')
```

```
Id 1168
*****************
MSSubClass 15
```

```
*****************
MSZoning 5
*****************
LotFrontage 106
*****************
LotArea 892
*****************
Street 2
*****************
LotShape 4
*****************
LandContour 4
*****************
Utilities 1
*****************
LotConfig 5
*****************
LandSlope 3
*****************
Neighborhood 25
*****************
Condition1 9
*****************
Condition2 8
*****************
BldgType 5
*****************
HouseStyle 8
*****************
OverallQual 10
*****************
OverallCond 9
*****************
YearBuilt 110
*****************
YearRemodAdd 61
*****************
RoofStyle 6
*****************
RoofMatl 8
*****************
Exterior1st 14
*****************
Exterior2nd 15
*****************
MasVnrType 4
*****************
MasVnrArea 284
*****************
ExterQual 4
*****************
ExterCond 5
*****************
Foundation 6
*****************
BsmtQual 4
*****************
BsmtCond 4
*****************
BsmtExposure 4
*****************
BsmtFinType1 6
*****************
BsmtFinSF1 551
*****************
BsmtFinType2 6
```

```
*****************
BsmtFinSF2 122
*****************
BsmtUnfSF 681
*****************
TotalBsmtSF 636
*****************
Heating 6
*****************
HeatingQC 5
*****************
CentralAir 2
*****************
Electrical 5
*****************
1stFlrSF 669
*****************
2ndFlrSF 351
*****************
LowQualFinSF 21
*****************
GrLivArea 746
*****************
BsmtFullBath 4
*****************
BsmtHalfBath 3
*****************
FullBath 4
*****************
HalfBath 3
*****************
BedroomAbvGr 8
*****************
KitchenAbvGr 4
*****************
KitchenQual 4
*****************
TotRmsAbvGrd 12
*****************
Functional 7
*****************
Fireplaces 4
*****************
FireplaceQu 5
*****************
GarageType 6
*****************
GarageYrBlt 97
*****************
GarageFinish 3
*****************
GarageCars 5
*****************
GarageArea 392
*****************
GarageQual 5
*****************
GarageCond 5
*****************
PavedDrive 3
*****************
WoodDeckSF 244
*****************
OpenPorchSF 176
*****************
EnclosedPorch 106
```

```
****************
3SsnPorch 18
****************
ScreenPorch 65
****************
PoolArea 8
****************
MiscVal 20
****************
MoSold 12
****************
YrSold 5
****************
SaleType 9
****************
SaleCondition 6
****************
SalePrice 581
****************
```

Utilities column has only one unique value, so dropped the column.

In [17]:
```
df.drop("Utilities",axis=1,inplace=True)
```

As column ID has no contribution in the output logically, we dropped column "ID".

In [18]:
```
df.drop("Id",axis=1,inplace=True)
```

Statistical Matrix

In [19]:
```
df.describe().T
```

Out[19]:

	count	mean	std	min	25%	50%	75%	max
MSSubClass	1168.0	56.767979	41.940650	20.0	20.00	50.0	70.00	190.0
LotFrontage	1168.0	70.807363	22.440317	21.0	60.00	70.0	79.25	313.0
LotArea	1168.0	10484.749144	8957.442311	1300.0	7621.50	9522.5	11515.50	164660.0
OverallQual	1168.0	6.104452	1.390153	1.0	5.00	6.0	7.00	10.0
OverallCond	1168.0	5.595890	1.124343	1.0	5.00	5.0	6.00	9.0
YearBuilt	1168.0	1970.930651	30.145255	1875.0	1954.00	1972.0	2000.00	2010.0
YearRemodAdd	1168.0	1984.758562	20.785185	1950.0	1966.00	1993.0	2004.00	2010.0
MasVnrArea	1168.0	102.310078	182.047152	0.0	0.00	0.0	160.00	1600.0
BsmtFinSF1	1168.0	444.726027	462.664785	0.0	0.00	385.5	714.50	5644.0
BsmtFinSF2	1168.0	46.647260	163.520016	0.0	0.00	0.0	0.00	1474.0
BsmtUnfSF	1168.0	569.721747	449.375525	0.0	216.00	474.0	816.00	2336.0
TotalBsmtSF	1168.0	1061.095034	442.272249	0.0	799.00	1005.5	1291.50	6110.0
1stFlrSF	1168.0	1169.860445	391.161983	334.0	892.00	1096.5	1392.00	4692.0
2ndFlrSF	1168.0	348.826199	439.696370	0.0	0.00	0.0	729.00	2065.0
LowQualFinSF	1168.0	6.380137	50.892844	0.0	0.00	0.0	0.00	572.0
GrLivArea	1168.0	1525.066781	528.042957	334.0	1143.25	1468.5	1795.00	5642.0

BsmtFullBath	1168.0	0.425514	0.521615	0.0	0.00	0.0	1.00	3.0
BsmtHalfBath	1168.0	0.055651	0.236699	0.0	0.00	0.0	0.00	2.0
FullBath	1168.0	1.562500	0.551882	0.0	1.00	2.0	2.00	3.0
HalfBath	1168.0	0.388699	0.504929	0.0	0.00	0.0	1.00	2.0
BedroomAbvGr	1168.0	2.884418	0.817229	0.0	2.00	3.0	3.00	8.0
KitchenAbvGr	1168.0	1.045377	0.216292	0.0	1.00	1.0	1.00	3.0
TotRmsAbvGrd	1168.0	6.542808	1.598484	2.0	5.00	6.0	7.00	14.0
Fireplaces	1168.0	0.617295	0.650575	0.0	0.00	1.0	1.00	3.0
GarageYrBlt	1168.0	1978.292808	24.202053	1900.0	1962.00	1980.0	2001.00	2010.0
GarageCars	1168.0	1.776541	0.745554	0.0	1.00	2.0	2.00	4.0
GarageArea	1168.0	476.860445	214.466769	0.0	338.00	480.0	576.00	1418.0
WoodDeckSF	1168.0	96.206336	126.158988	0.0	0.00	0.0	171.00	857.0
OpenPorchSF	1168.0	46.559932	66.381023	0.0	0.00	24.0	70.00	547.0
EnclosedPorch	1168.0	23.015411	63.191089	0.0	0.00	0.0	0.00	552.0
3SsnPorch	1168.0	3.639555	29.088867	0.0	0.00	0.0	0.00	508.0
ScreenPorch	1168.0	15.051370	55.080816	0.0	0.00	0.0	0.00	480.0
PoolArea	1168.0	3.448630	44.896939	0.0	0.00	0.0	0.00	738.0
MiscVal	1168.0	47.315068	543.264432	0.0	0.00	0.0	0.00	15500.0
MoSold	1168.0	6.344178	2.686352	1.0	5.00	6.0	8.00	12.0
YrSold	1168.0	2007.804795	1.329738	2006.0	2007.00	2008.0	2009.00	2010.0
SalePrice	1168.0	181477.005993	79105.586863	34900.0	130375.00	163995.0	215000.00	755000.0

Observation

1. By comparing 75% and max column we can conclude that some of the feature contain outliers.

2. By looking at Mean & Median columns we can say that some of features are left skewed while others are right skewed.

3. Oldest Property is built in 1875 while recent property build in 2010.

Feature Extraction- age from year

In [20]:

```
# Converting years column to age column
df['Year_SinceBuilt'] = df['YearBuilt'].max() - df['YearBuilt']
df['Year_SinceRemodAdded'] = df['YearRemodAdd'].max() - df['YearRemodAdd']
df['Year_Since'] = df['YrSold'].max() - df['YrSold']
df['GarageAge'] = df['GarageYrBlt'].max() - df['GarageYrBlt']
```

In [21]:

```
# Dropping old columns in train dataset
df.drop(['YearBuilt','YearRemodAdd','YrSold','GarageYrBlt'], axis=1, inplace = True)
```

In [22]:

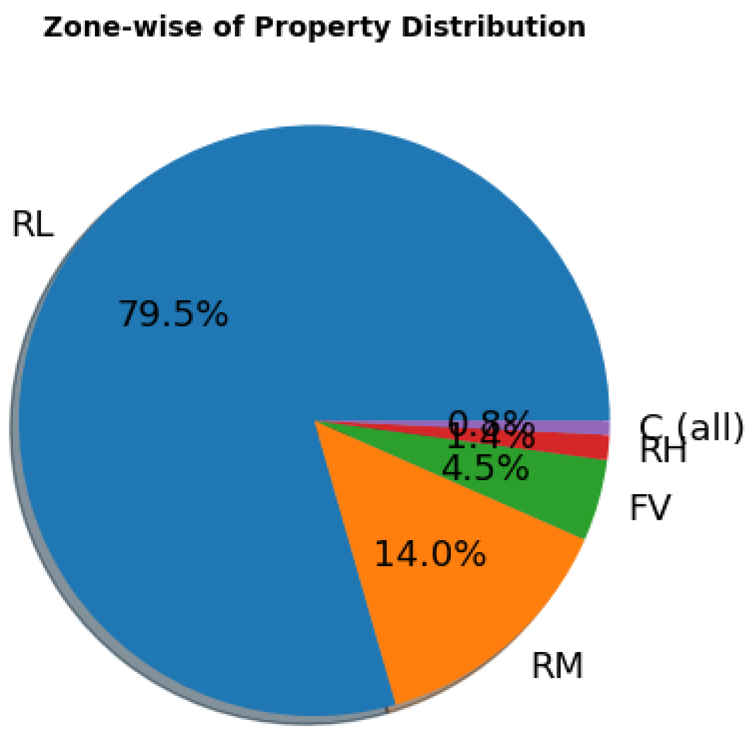
```
df.rename(columns= {'Year_Since' : 'Year_Since_Sold'}, inplace = True)
```

Exploratory Data Analysis

Zone-wise of Property Distribution

In [23]:
```
# Exploring MSZoning Type
ax=df['MSZoning'].value_counts().plot.pie(autopct='%2.1f%%',
                                          textprops ={ 'fontsize':13},shadow=True)
ax.set_title('Zone-wise of Property Distribution', fontsize=10,fontweight ='bold')
ax.set_ylabel('')
```

Out[23]:
```
Text(0, 0.5, '')
```



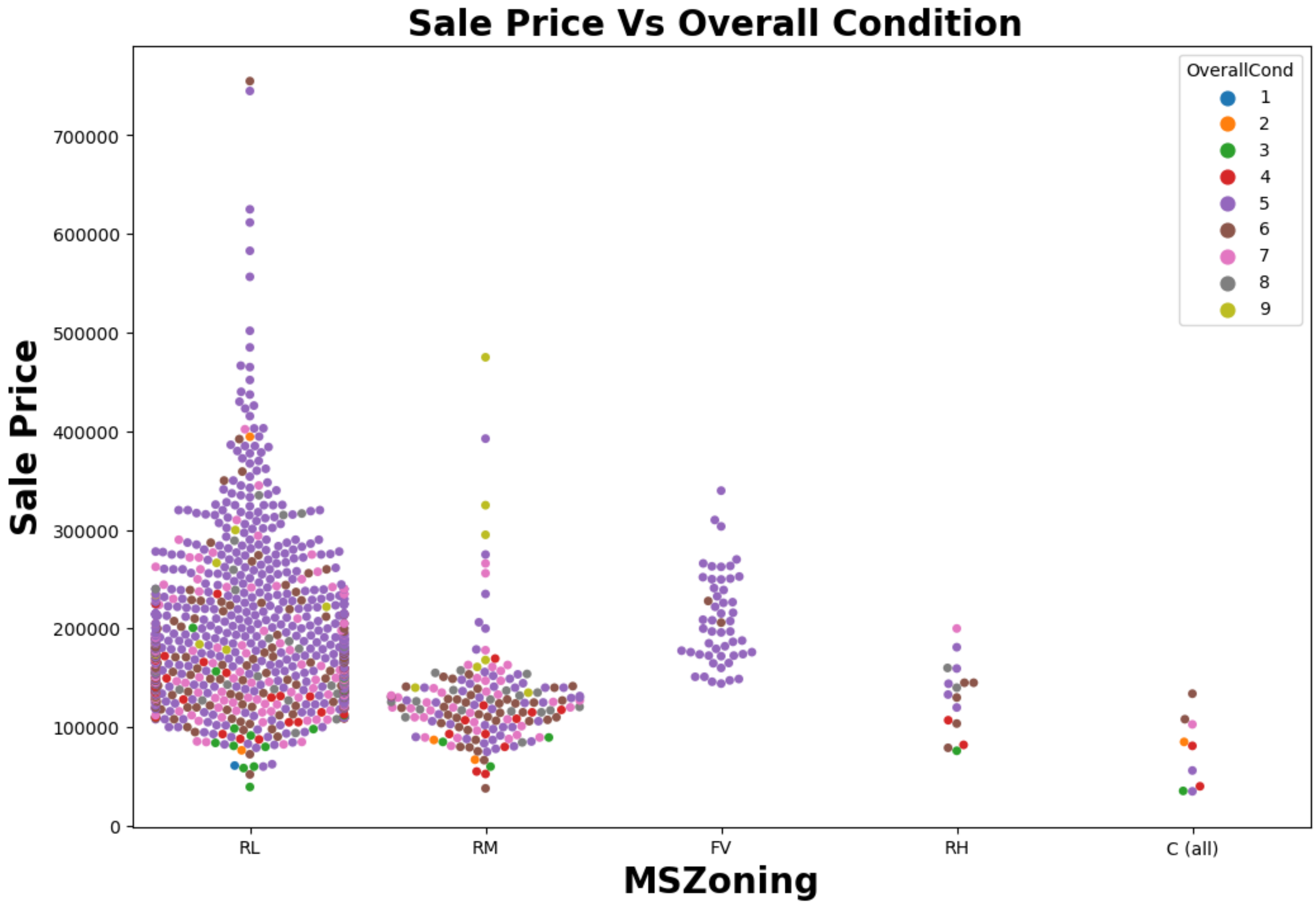
Observations:

1. 79.5% of House properties belongs to Low Density Residential Area followed by 14 % of properties belong to Medium Density Residential Area.
2. Very Few property (0.8%) belongs to Commerical zone.

Price relation with zone

In [24]:
```
plt.figure(figsize=(12,8))
sns.swarmplot(y=df['SalePrice'], x=df['MSZoning'], hue =df['OverallCond'])
plt.title("Sale Price Vs Overall Condition ",fontsize=20,fontweight ='bold')
plt.xlabel('MSZoning',fontsize = 20,fontweight ='bold')
plt.ylabel('Sale Price',fontsize = 20,fontweight ='bold')
```

Out[24]:
```
Text(0, 0.5, 'Sale Price')
```



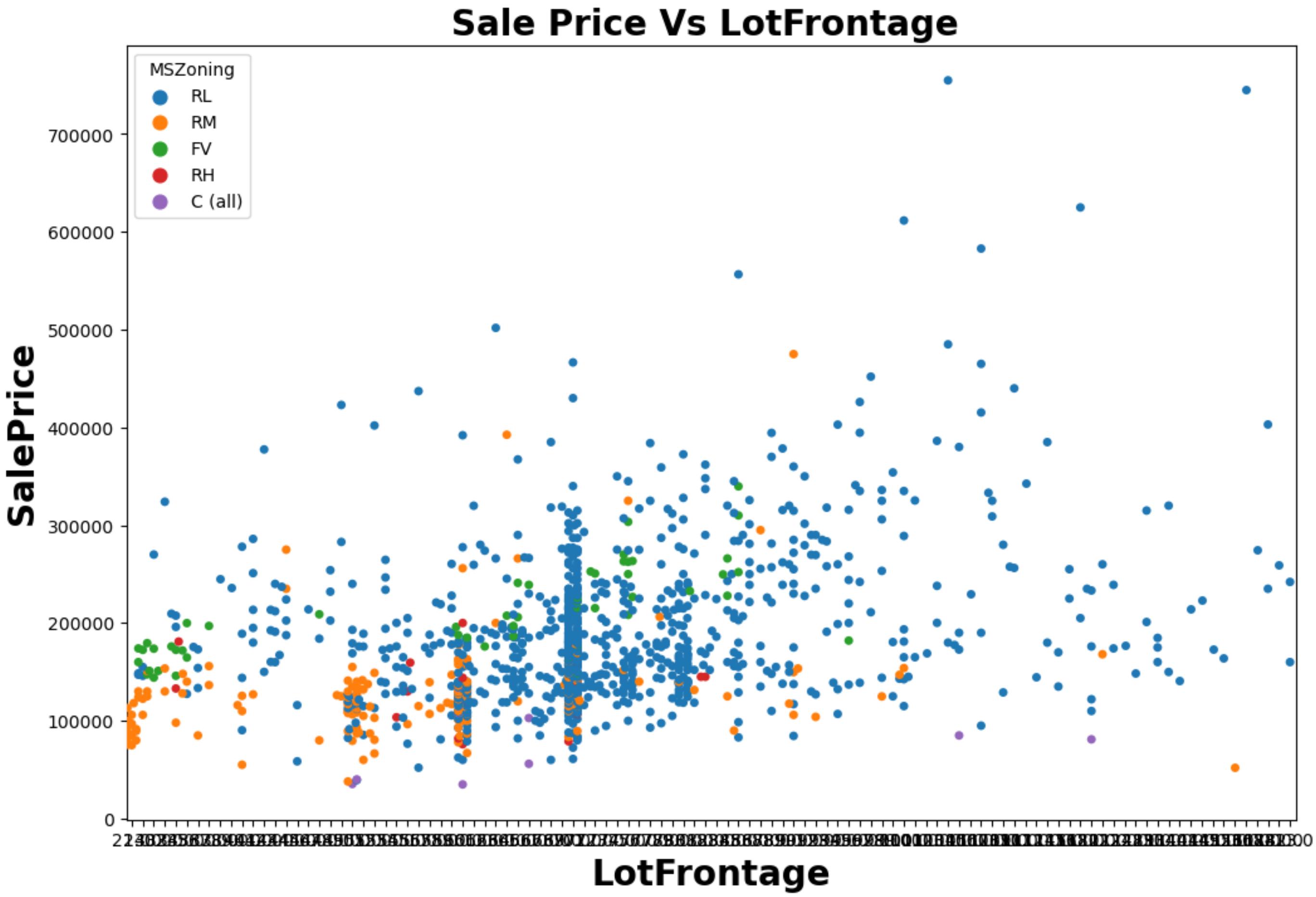
Observation :

1. Most of property for sale have overall condition rating of either 5 or 6.
2. Sale Price inside RL Zone is much higher than other remaining zone.
3. Cheapest properties are available in Commerical zone.
4. Another interesting observation we get here is for some house properties having Overall condition Rating of 8 & 9 have low price compare to others. This indicate that Overall Condition Rating is Not significant factor in determination of Sale price. Overall Condition Rating may helpful to buyer in taking decision of Buying property but not in determination of House Price.

LotFrontage(Linear feet of street connected to property) realtion with sale price

In [25]:
```
plt.figure(figsize=(12,8))
sns.swarmplot(y=df['SalePrice'], x=df['LotFrontage'], hue =df['MSZoning'])
plt.title("Sale Price Vs LotFrontage ",fontsize=20,fontweight ='bold')
plt.xlabel('LotFrontage',fontsize = 20,fontweight ='bold')
plt.ylabel('SalePrice',fontsize = 20,fontweight ='bold')
```

Out[25]:
```
Text(0, 0.5, 'SalePrice')
```

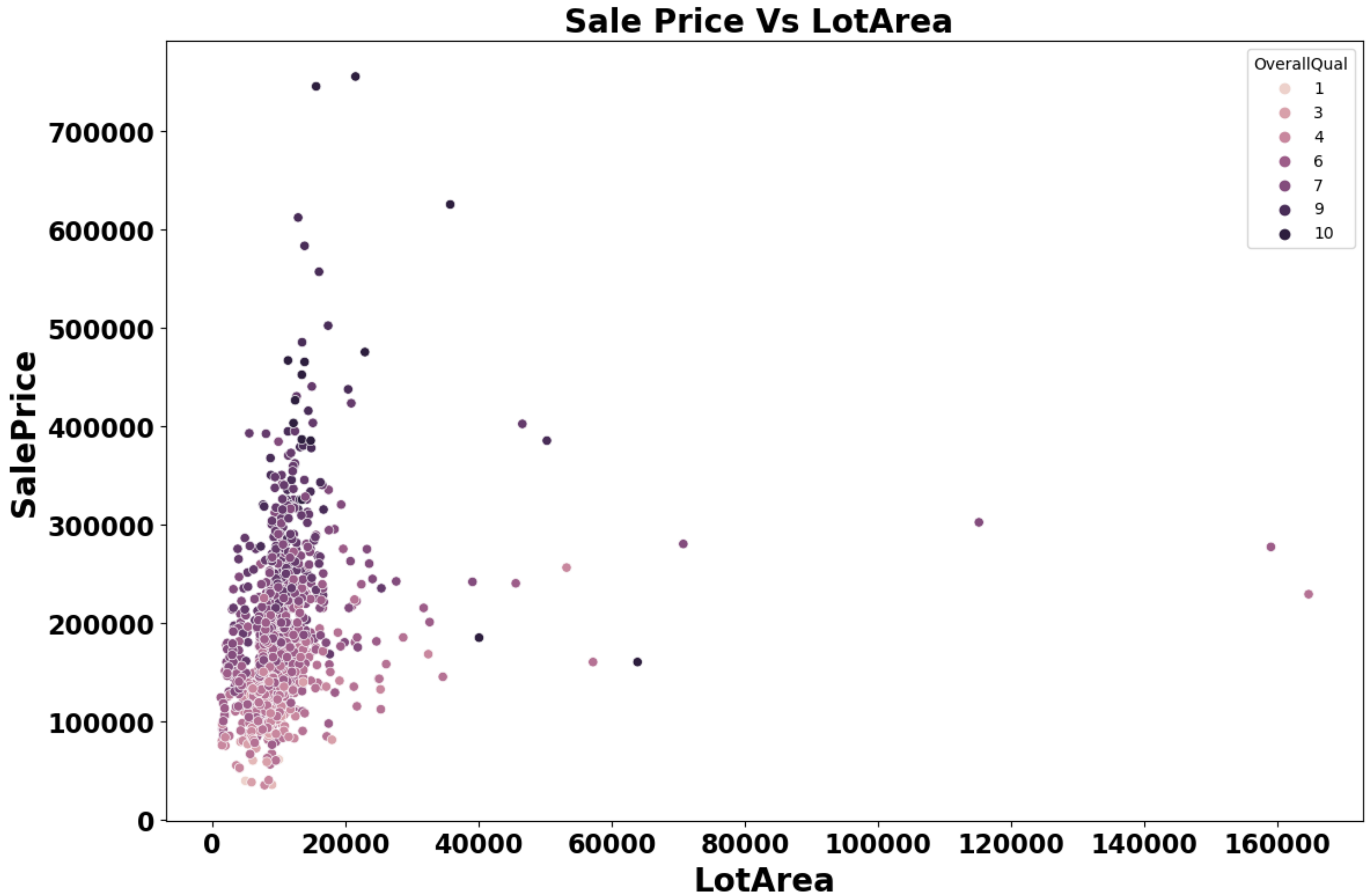


Observation:

With Exception of Commerical zone, As Lot Frontage area increase (which indicate Size of street connected to property) the Sale Price increases.

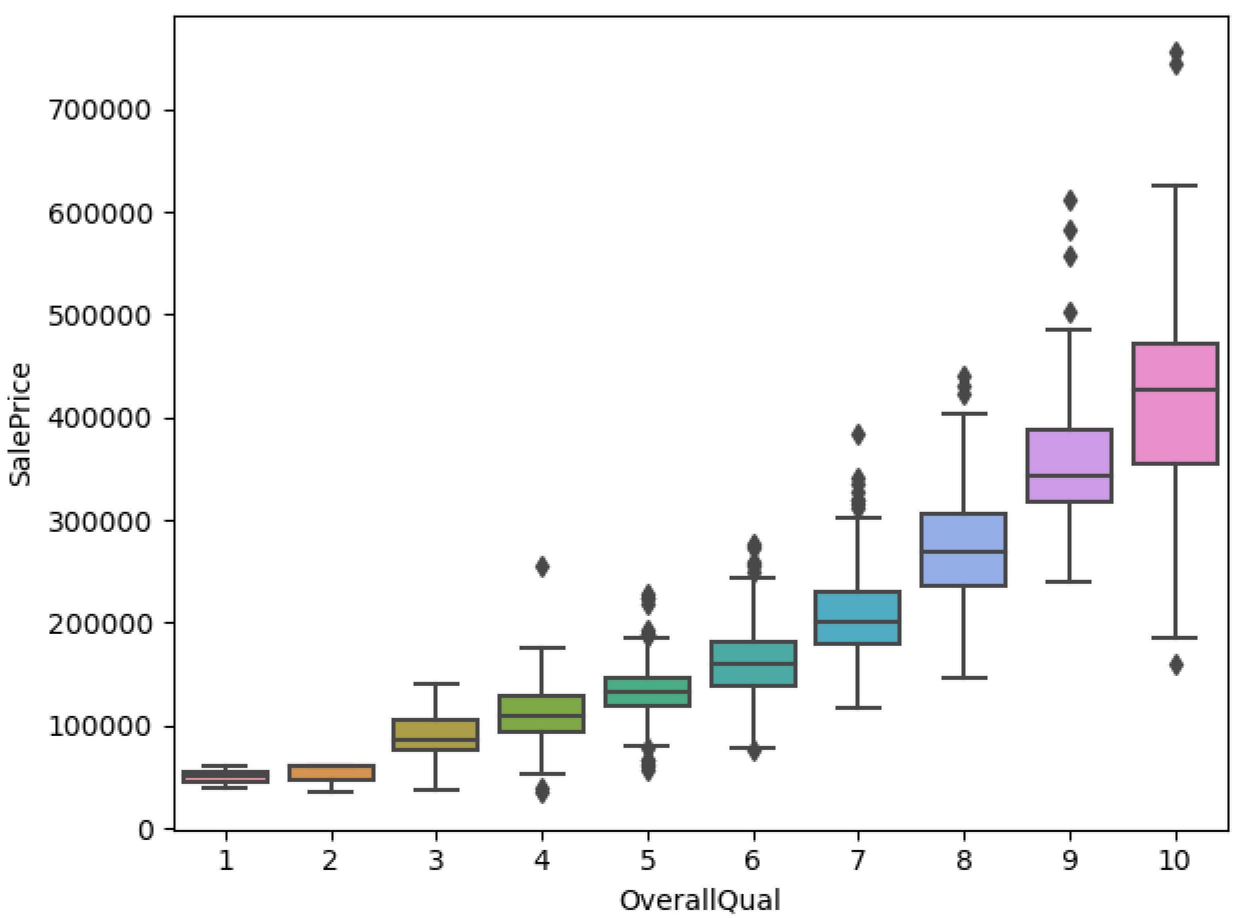
Quality & Area of house relation woth house Pricing

In [26]:
```
plt.rcParams['figure.autolayout']= True
sns.set_palette('rainbow')
plt.figure(figsize=(12,8))
sns.scatterplot(y=df['SalePrice'], x=df['LotArea'], hue =df['OverallQual'])
plt.title("Sale Price Vs LotArea ",fontsize=20,fontweight ='bold')
plt.xlabel('LotArea',fontsize = 20,fontweight ='bold')
plt.ylabel('SalePrice',fontsize = 20,fontweight ='bold')
plt.xticks(fontsize=16,fontweight ='bold')
plt.yticks(fontsize=16,fontweight ='bold')
plt.tight_layout()
plt.show()
```



In [27]:
```
sns.boxplot(y = df['SalePrice'], x= df['OverallQual'])
```

Out[27]:
```
<AxesSubplot:xlabel='OverallQual', ylabel='SalePrice'>
```



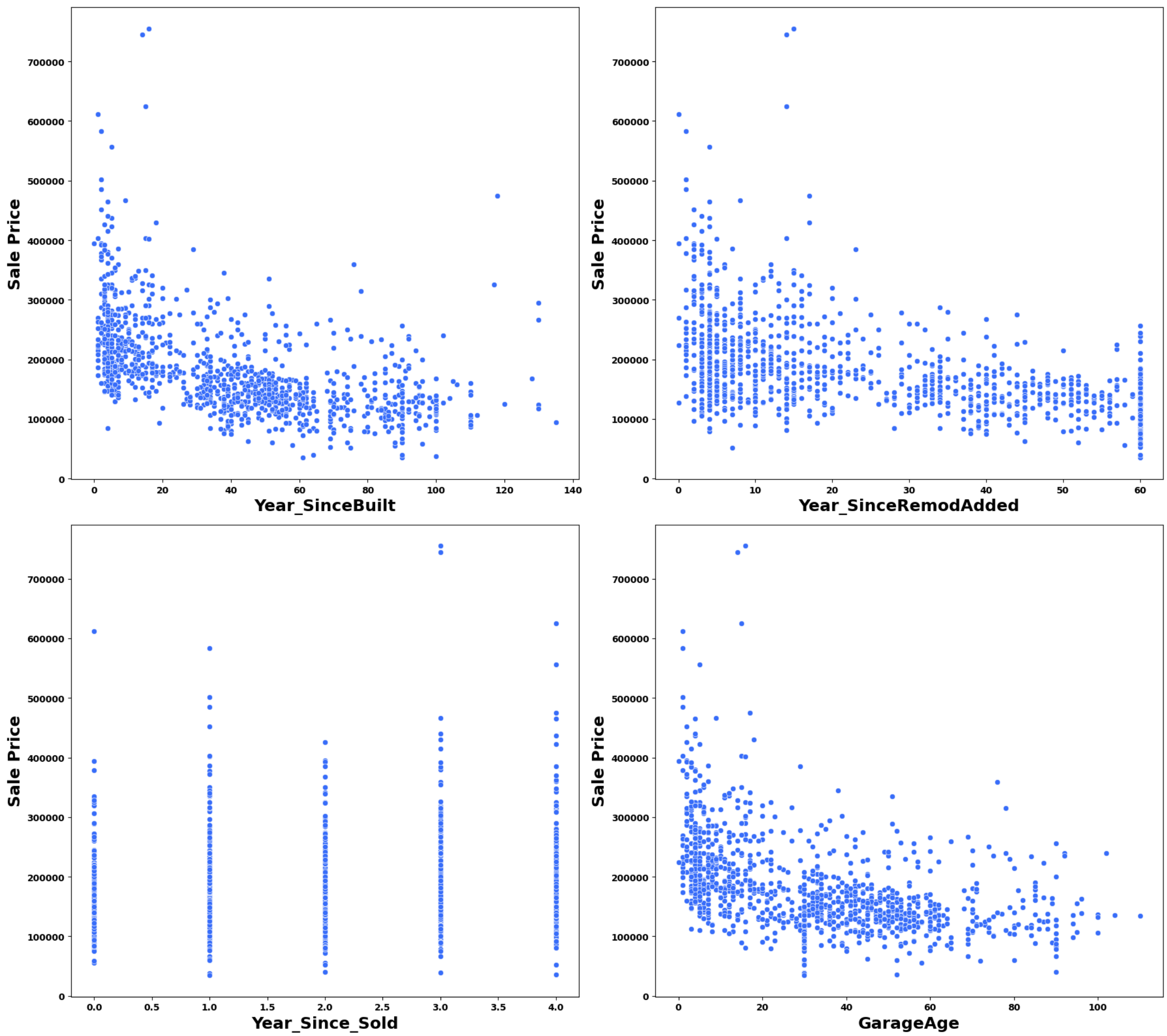
Observation:

1. There is No Significant relationship found between Sale price & Lot area.
2. Most houses fall under 40000 sqft. and the price varies based on overall quality.

In [28]:

```
plt.figure(figsize=(18,16),facecolor='white')
a=1

for i in ["Year_SinceBuilt", "Year_SinceRemodAdded", 'Year_Since_Sold', "GarageAge"]:
    if a<=4:
        ax=plt.subplot(2,2,a)
        sns.scatterplot(y = df['SalePrice'], x= df[i])
        plt.xlabel(i,fontsize=18,fontweight ='bold')
        plt.ylabel('Sale Price', fontsize =18, fontweight='bold')
        plt.xticks(fontweight ='bold')
        plt.yticks(fontweight ='bold')
    a+=1
plt.tight_layout()
plt.show()
```



Observation :-

1. We can see that as Property get older with time its sale Price get depricates.

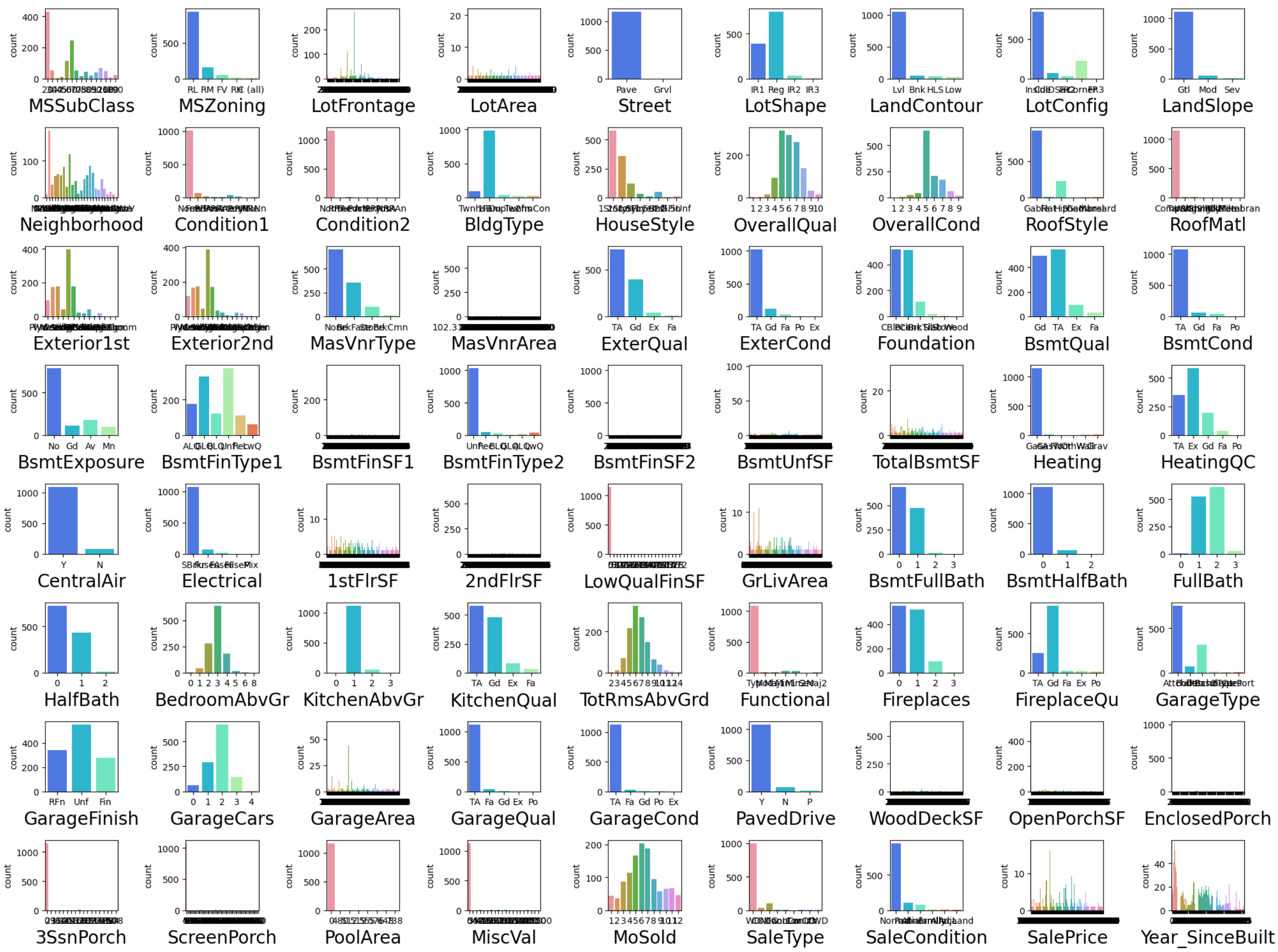
2. 20 years after Remodelling Price of properties start decreases.

3. Older the garage age less the price of Property.

In [29]:
```
plt.figure(figsize = (20,15))
plotnumber = 1

for column in df:
    if plotnumber <= 72:
        ax = plt.subplot(8,9,plotnumber)
        sns.countplot(df[column])
        plt.xlabel(column,fontsize = 20)

    plotnumber+=1
plt.tight_layout()
```



Encoding Categorical Features

In [30]:
```
Categorical_features = ['MSZoning', 'Street', 'LotShape', 'LandContour', 'LotConfig', 'L
                        'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'RoofStyle
                        'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundati
                        'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating', 'Heat
                        'Electrical', 'KitchenQual', 'Functional', 'GarageType', 'Garage
                        'GarageCond', 'PavedDrive', 'SaleType', 'SaleCondition','Firepla
```

In [31]:
```
# Using Label encoder for transforming Categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in Categorical_features:
    df[i] = le.fit_transform(df[i])
df.head()
```

Out[31]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighbo
0	120	3	70.0	4928	1	0	3	4	0	
1	20	3	95.0	15865	1	0	3	4	1	
2	60	3	92.0	9920	1	0	3	1	0	
3	20	3	105.0	11751	1	0	3	4	0	
4	20	3	70.0	16635	1	0	3	2	0	

In [32]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1168 entries, 0 to 1167
Data columns (total 75 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   MSSubClass      1168 non-null   int64
 1   MSZoning        1168 non-null   int32
 2   LotFrontage     1168 non-null   float64
 3   LotArea         1168 non-null   int64
 4   Street          1168 non-null   int32
 5   LotShape        1168 non-null   int32
 6   LandContour     1168 non-null   int32
 7   LotConfig       1168 non-null   int32
 8   LandSlope       1168 non-null   int32
 9   Neighborhood    1168 non-null   int32
 10  Condition1      1168 non-null   int32
 11  Condition2      1168 non-null   int32
 12  BldgType        1168 non-null   int32
 13  HouseStyle      1168 non-null   int32
 14  OverallQual     1168 non-null   int64
 15  OverallCond     1168 non-null   int64
 16  RoofStyle       1168 non-null   int32
 17  RoofMatl        1168 non-null   int32
 18  Exterior1st     1168 non-null   int32
 19  Exterior2nd     1168 non-null   int32
 20  MasVnrType      1168 non-null   int32
 21  MasVnrArea      1168 non-null   float64
 22  ExterQual       1168 non-null   int32
 23  ExterCond       1168 non-null   int32
 24  Foundation      1168 non-null   int32
 25  BsmtQual        1168 non-null   int32
 26  BsmtCond        1168 non-null   int32
 27  BsmtExposure    1168 non-null   int32
 28  BsmtFinType1    1168 non-null   int32
 29  BsmtFinSF1      1168 non-null   int64
 30  BsmtFinType2    1168 non-null   int32
 31  BsmtFinSF2      1168 non-null   int64
 32  BsmtUnfSF       1168 non-null   int64
 33  TotalBsmtSF     1168 non-null   int64
 34  Heating         1168 non-null   int32
 35  HeatingQC       1168 non-null   int32
 36  CentralAir      1168 non-null   int32
 37  Electrical      1168 non-null   int32
 38  1stFlrSF        1168 non-null   int64
 39  2ndFlrSF        1168 non-null   int64
 40  LowQualFinSF    1168 non-null   int64
 41  GrLivArea       1168 non-null   int64
 42  BsmtFullBath    1168 non-null   int64
 43  BsmtHalfBath    1168 non-null   int64
 44  FullBath        1168 non-null   int64
 45  HalfBath        1168 non-null   int64
```

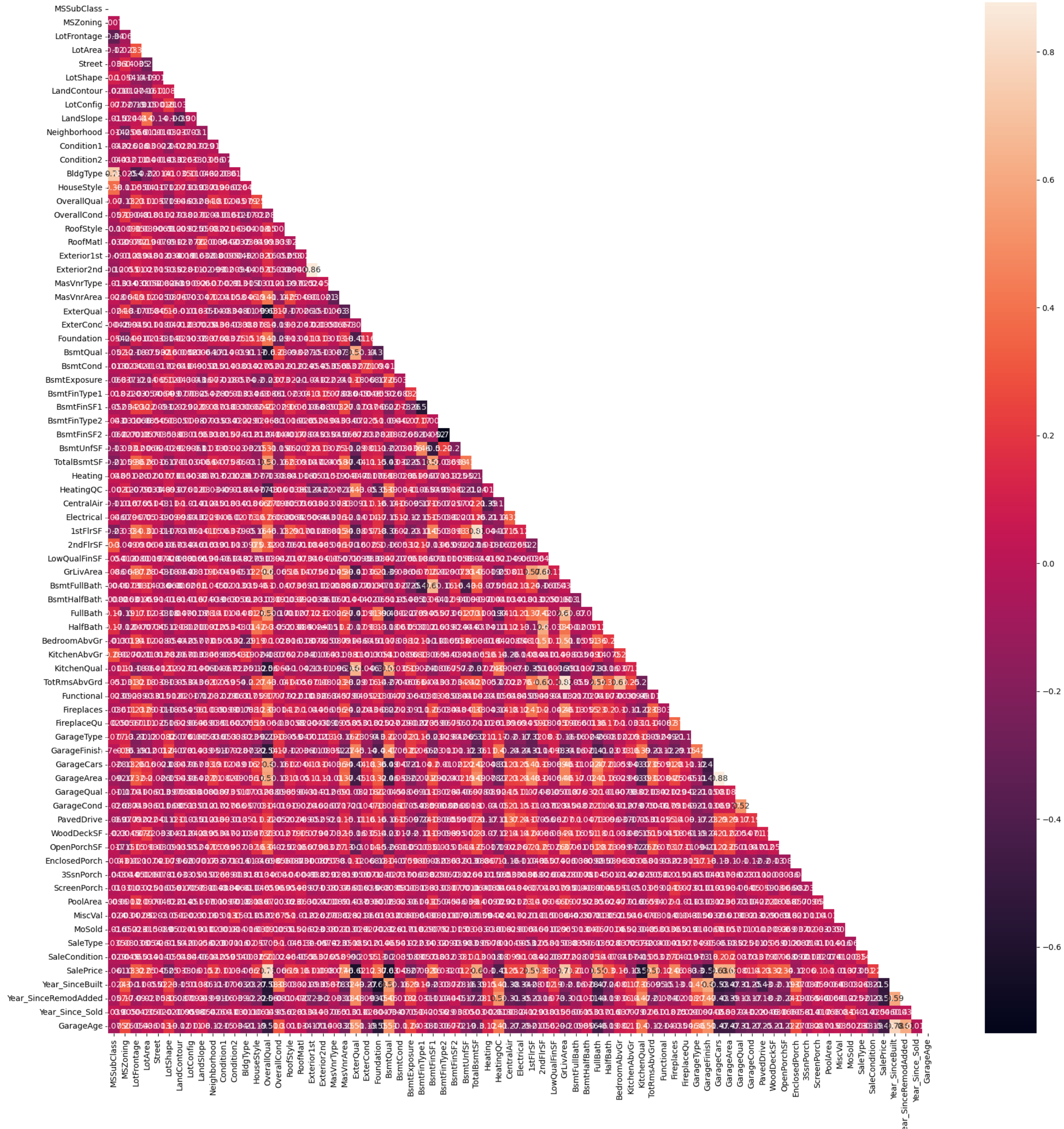
```
 46  BedroomAbvGr          1168 non-null   int64
 47  KitchenAbvGr          1168 non-null   int64
 48  KitchenQual           1168 non-null   int32
 49  TotRmsAbvGrd          1168 non-null   int64
 50  Functional            1168 non-null   int32
 51  Fireplaces            1168 non-null   int64
 52  FireplaceQu           1168 non-null   int32
 53  GarageType            1168 non-null   int32
 54  GarageFinish          1168 non-null   int32
 55  GarageCars            1168 non-null   int64
 56  GarageArea            1168 non-null   int64
 57  GarageQual            1168 non-null   int32
 58  GarageCond            1168 non-null   int32
 59  PavedDrive            1168 non-null   int32
 60  WoodDeckSF            1168 non-null   int64
 61  OpenPorchSF           1168 non-null   int64
 62  EnclosedPorch         1168 non-null   int64
 63  3SsnPorch             1168 non-null   int64
 64  ScreenPorch           1168 non-null   int64
 65  PoolArea              1168 non-null   int64
 66  MiscVal               1168 non-null   int64
 67  MoSold                1168 non-null   int64
 68  SaleType              1168 non-null   int32
 69  SaleCondition         1168 non-null   int32
 70  SalePrice             1168 non-null   int64
 71  Year_SinceBuilt       1168 non-null   int64
 72  Year_SinceRemodAdded  1168 non-null   int64
 73  Year_Since_Sold       1168 non-null   int64
 74  GarageAge             1168 non-null   float64
dtypes: float64(3), int32(38), int64(34)
memory usage: 511.1 KB
```

No columns with object datatype present.

Checking for Correlation.

In [33]:
```
plt.figure(figsize = (20,20))
corr=df.corr()
sns.heatmap(corr,annot = True,fmt = '.2g',mask=np.triu(np.ones_like(corr)))
```

Out[33]:
```
<AxesSubplot:>
```



Dropping the columns whose correlation is more than 80%.

In [34]:
```python
# To find index values where correlation is more than 0.8
corr.replace(1,0,inplace=True)
hh=corr.applymap(lambda x: x>0.8).to_numpy()
hh
indices = np.argwhere(hh)
indices
```

Out[34]:
```
array([[18, 19],
       [19, 18],
       [33, 38],
       [38, 33],
       [41, 49],
       [49, 41],
       [55, 56],
       [56, 55]], dtype=int64)
```

In [35]:
```
for i in indices:
    print(corr.iloc[0,i])
```

```
Exterior1st   -0.090178
Exterior2nd   -0.120022
Name: MSSubClass, dtype: float64
Exterior2nd   -0.120022
Exterior1st   -0.090178
Name: MSSubClass, dtype: float64
TotalBsmtSF   -0.214042
1stFlrSF      -0.227927
Name: MSSubClass, dtype: float64
1stFlrSF      -0.227927
TotalBsmtSF   -0.214042
Name: MSSubClass, dtype: float64
GrLivArea       0.086448
TotRmsAbvGrd    0.051179
Name: MSSubClass, dtype: float64
TotRmsAbvGrd    0.051179
GrLivArea       0.086448
Name: MSSubClass, dtype: float64
GarageCars   -0.027639
GarageArea   -0.092408
Name: MSSubClass, dtype: float64
GarageArea   -0.092408
GarageCars   -0.027639
Name: MSSubClass, dtype: float64
```

From the above list we can conclude that the foloowing columns have correlation is more than 80%.

1. GarageArea and GarageCars
2. TotRmsAbvGrd and GrLivArea
3. 1stFlrSF and TotalBsmtSF
4. Exterior2nd and Exterior1st

In [36]:
```
df.drop(columns = ['TotRmsAbvGrd','GarageArea','TotalBsmtSF','Exterior2nd'],axis = 1, in
```

Outlier treatment

Before the outlier treatment we must remove the columns with string values which have no significance in the output.

In [37]:
```
df.info()
```

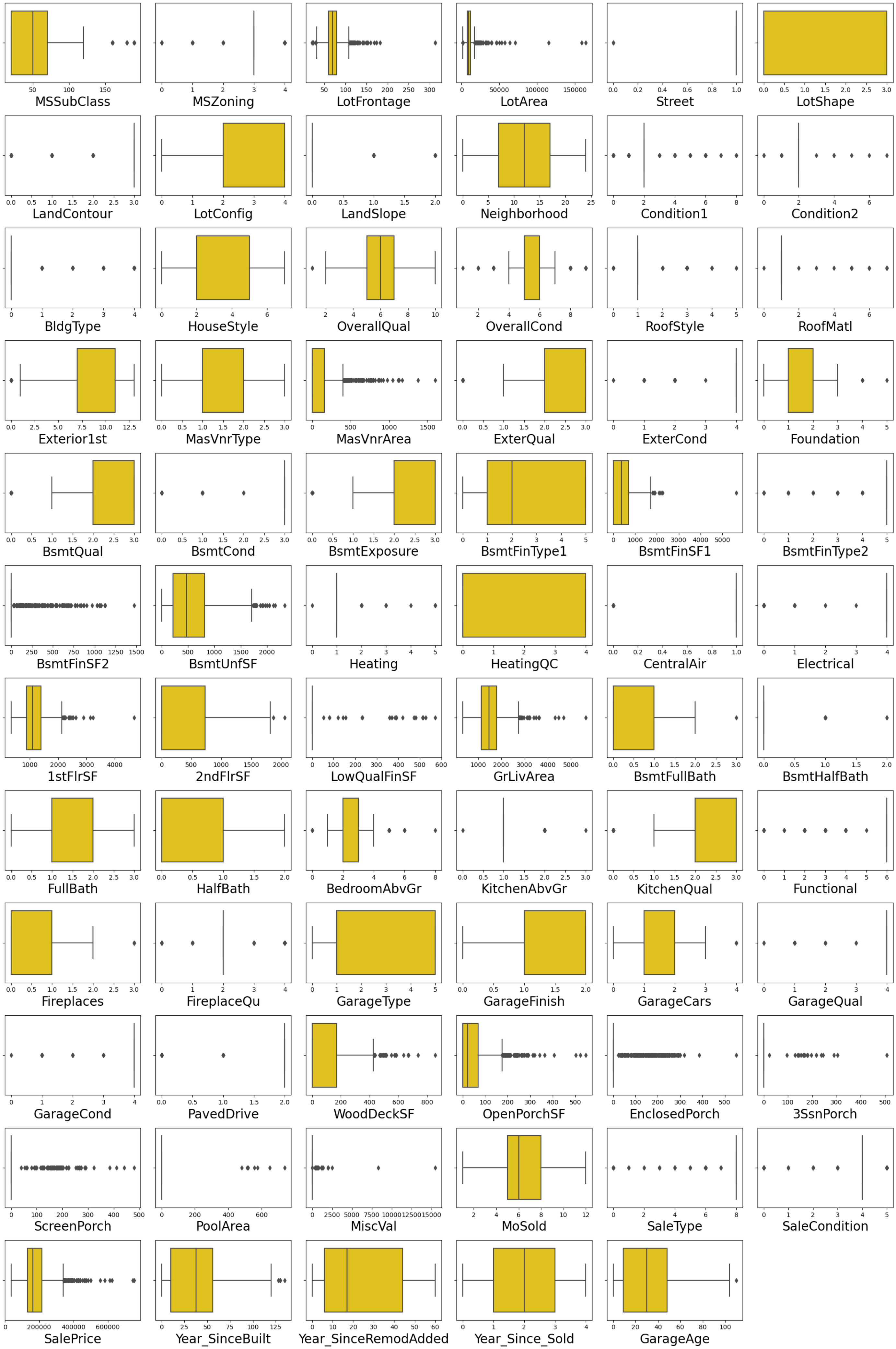
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1168 entries, 0 to 1167
Data columns (total 71 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   MSSubClass          1168 non-null   int64
 1   MSZoning            1168 non-null   int32
 2   LotFrontage         1168 non-null   float64
 3   LotArea             1168 non-null   int64
 4   Street              1168 non-null   int32
 5   LotShape            1168 non-null   int32
 6   LandContour         1168 non-null   int32
 7   LotConfig           1168 non-null   int32
 8   LandSlope           1168 non-null   int32
 9   Neighborhood        1168 non-null   int32
 10  Condition1          1168 non-null   int32
 11  Condition2          1168 non-null   int32
```

```
 12  BldgType             1168 non-null   int32
 13  HouseStyle           1168 non-null   int32
 14  OverallQual          1168 non-null   int64
 15  OverallCond          1168 non-null   int64
 16  RoofStyle            1168 non-null   int32
 17  RoofMatl             1168 non-null   int32
 18  Exterior1st          1168 non-null   int32
 19  MasVnrType           1168 non-null   int32
 20  MasVnrArea           1168 non-null   float64
 21  ExterQual            1168 non-null   int32
 22  ExterCond            1168 non-null   int32
 23  Foundation           1168 non-null   int32
 24  BsmtQual             1168 non-null   int32
 25  BsmtCond             1168 non-null   int32
 26  BsmtExposure         1168 non-null   int32
 27  BsmtFinType1         1168 non-null   int32
 28  BsmtFinSF1           1168 non-null   int64
 29  BsmtFinType2         1168 non-null   int32
 30  BsmtFinSF2           1168 non-null   int64
 31  BsmtUnfSF            1168 non-null   int64
 32  Heating              1168 non-null   int32
 33  HeatingQC            1168 non-null   int32
 34  CentralAir           1168 non-null   int32
 35  Electrical           1168 non-null   int32
 36  1stFlrSF             1168 non-null   int64
 37  2ndFlrSF             1168 non-null   int64
 38  LowQualFinSF         1168 non-null   int64
 39  GrLivArea            1168 non-null   int64
 40  BsmtFullBath         1168 non-null   int64
 41  BsmtHalfBath         1168 non-null   int64
 42  FullBath             1168 non-null   int64
 43  HalfBath             1168 non-null   int64
 44  BedroomAbvGr         1168 non-null   int64
 45  KitchenAbvGr         1168 non-null   int64
 46  KitchenQual          1168 non-null   int32
 47  Functional           1168 non-null   int32
 48  Fireplaces           1168 non-null   int64
 49  FireplaceQu          1168 non-null   int32
 50  GarageType           1168 non-null   int32
 51  GarageFinish         1168 non-null   int32
 52  GarageCars           1168 non-null   int64
 53  GarageQual           1168 non-null   int32
 54  GarageCond           1168 non-null   int32
 55  PavedDrive           1168 non-null   int32
 56  WoodDeckSF           1168 non-null   int64
 57  OpenPorchSF          1168 non-null   int64
 58  EnclosedPorch        1168 non-null   int64
 59  3SsnPorch            1168 non-null   int64
 60  ScreenPorch          1168 non-null   int64
 61  PoolArea             1168 non-null   int64
 62  MiscVal              1168 non-null   int64
 63  MoSold               1168 non-null   int64
 64  SaleType             1168 non-null   int32
 65  SaleCondition        1168 non-null   int32
 66  SalePrice            1168 non-null   int64
 67  Year_SinceBuilt      1168 non-null   int64
 68  Year_SinceRemodAdded 1168 non-null   int64
 69  Year_Since_Sold      1168 non-null   int64
 70  GarageAge            1168 non-null   float64
dtypes: float64(3), int32(37), int64(31)
memory usage: 479.2 KB
```

All coulmns have int and float datatypes.

In [38]:
```
# Identifying the outliers using boxplot in train dataset
```

```
plt.figure(figsize=(20,30),facecolor='white')
plotnumber=1
for i in df.columns:
    if plotnumber<=72:
        ax=plt.subplot(12,6,plotnumber)
        sns.boxplot(df[i],color='gold')
        plt.xlabel(i,fontsize=20)
    plotnumber+=1
plt.tight_layout()
```


MSSubClass
MSZoning
LotFrontage
LotArea
Street
LotShape
LandContour
LotConfig
LandSlope
Neighborhood
Condition1
Condition2
BldgType
HouseStyle
OverallQual
OverallCond
RoofStyle
RoofMatl
Exterior1st
MasVnrType
MasVnrArea
ExterQual
ExterCond
Foundation
BsmtQual
BsmtCond
BsmtExposure
BsmtFinType1
BsmtFinSF1
BsmtFinType2
BsmtFinSF2
BsmtUnfSF
Heating
HeatingQC
CentralAir
Electrical
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
KitchenQual
Functional
Fireplaces
FireplaceQu
GarageType
GarageFinish
GarageCars
GarageQual
GarageCond
PavedDrive
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold
SaleType
SaleCondition
SalePrice
Year_SinceBuilt
Year_SinceRemodAdded
Year_Since_Sold
GarageAge

```
In [39]: z=np.abs(zscore(df))
         dfn=df[(z<3).all(axis=1)]
```

Model Building.

Standard Scaling

```
In [40]: # Splitting data in target and dependent feature
         X = df.drop(['SalePrice'], axis =1)
         Y = df['SalePrice']
```

```
In [41]: scaler= StandardScaler()
         X_scale = scaler.fit_transform(X)
```

```
In [43]: train_x, test_x, train_y, test_y = train_test_split(X_scale, Y, random_state=99, test_si
         print('Training Feature Matrix Size:', train_x.shape)
         print('Training Target Vector Size :', train_y.shape)
         print('Test Feature Matrix Size:', test_x.shape)
         print('Test Target Vector Size:', test_y.shape)
```

```
Training Feature Matrix Size: (817, 70)
Training Target Vector Size : (817,)
Test Feature Matrix Size: (351, 70)
Test Target Vector Size: (351,)
```

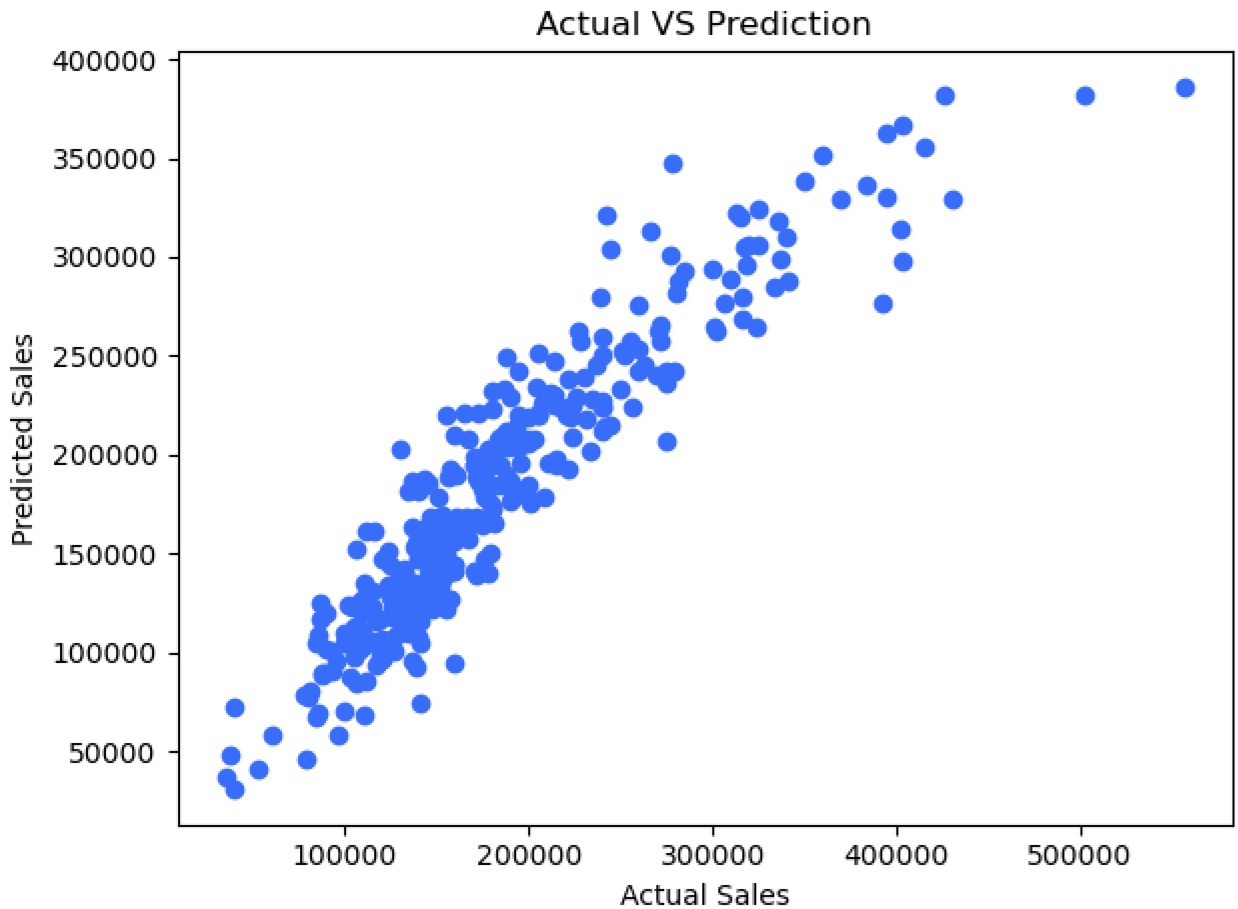
Finding the best model

```
In [44]: lr=LinearRegression()
         lasso=linear_model.Lasso()
         svr=SVR()
         dtr=DecisionTreeRegressor()
         rfr=RandomForestRegressor()
```

```
In [45]: model=[lr,lasso,svr,dtr,rfr]
         for m in model:
             m.fit(train_x,train_y)
             m.score(train_x,train_y)
             predm=m.predict(test_x)
             print(f"Scores for {m} are")
             print('Mean Absolute Error:', metrics.mean_absolute_error(test_y, predm))
             print('Mean Squared Error:', metrics.mean_squared_error(test_y, predm))
             print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(test_y, predm))
             print('R squared score: ', r2_score(test_y, predm))
             score=cross_val_score(m,train_x,train_y,cv=5)
             print("Cross Validation Score is :",score)
             print("Mean Score :",score.mean())
             print("Difference :",score.mean()-r2_score(test_y, predm))
             plt.scatter(test_y, predm)
             plt.xlabel("Actual Sales")
             plt.ylabel("Predicted Sales")
             plt.title("Actual VS Prediction")
             plt.show()
             print("==============================================\n\n\n========================
```

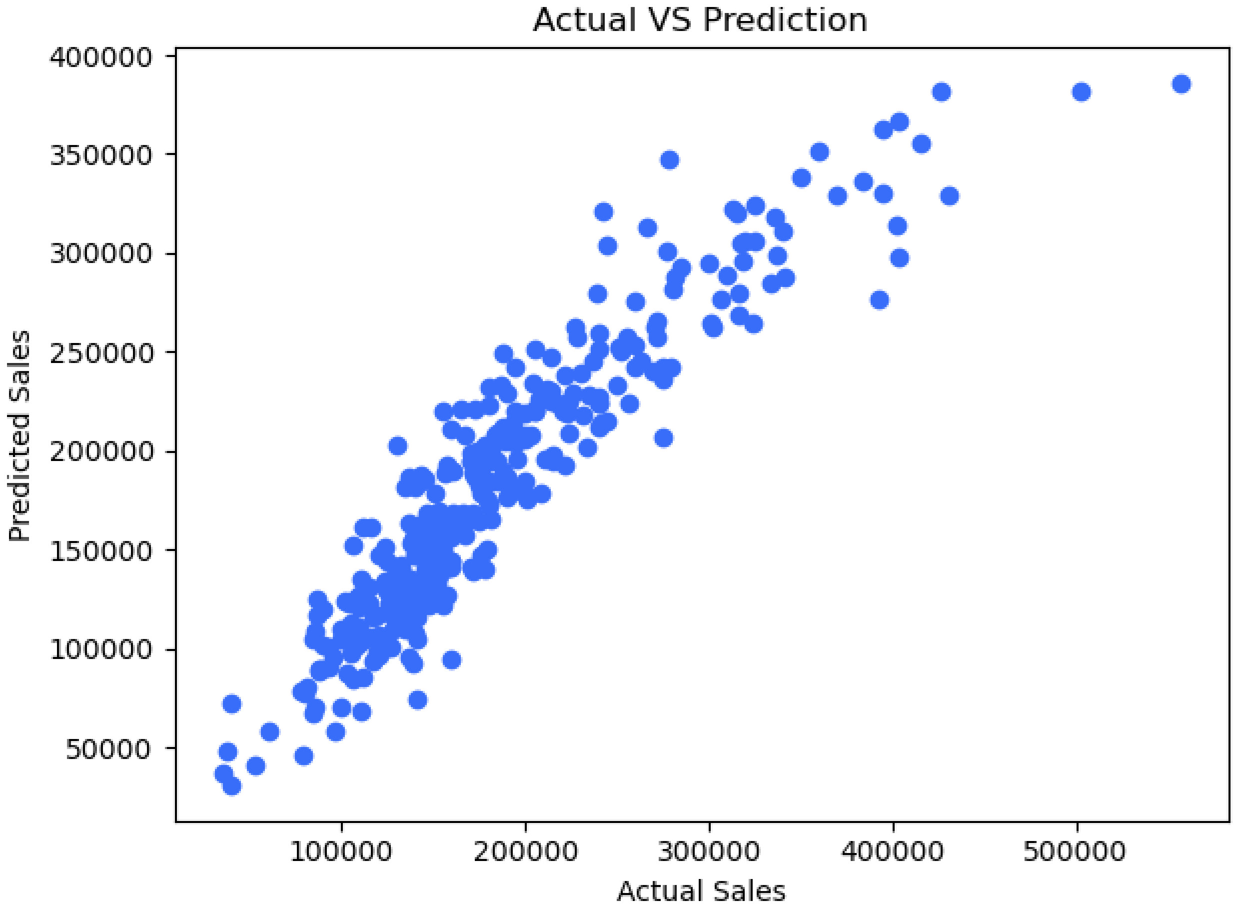
```
Scores for LinearRegression() are
Mean Absolute Error: 21170.39666608148
Mean Squared Error: 858383692.5971951
Root Mean Squared Error: 29298.185824333817
R squared score:  0.8596665675880402
```

```
Cross Validation Score is : [0.84275928 0.67878349 0.46247683 0.78869014 0.6728491 ]
Mean Score : 0.6891117685109865
Difference : -0.17055479907705362
```



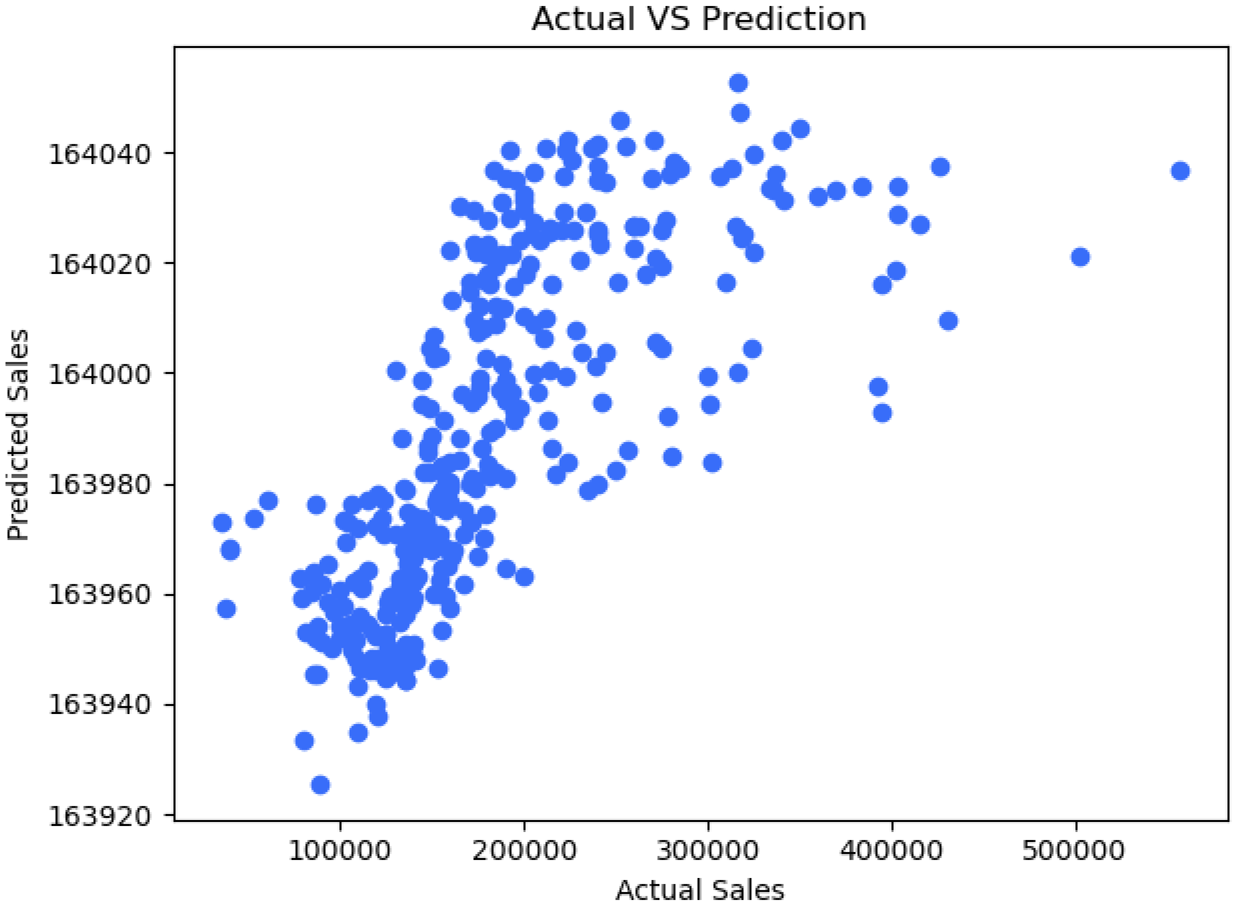

```
==============================================

=================================================
Scores for Lasso() are
Mean Absolute Error: 21168.395205360266
Mean Squared Error: 858289486.581479
Root Mean Squared Error: 29296.578069485844
R squared score:  0.8596819689215618
Cross Validation Score is : [0.84285582 0.67879124 0.46279577 0.78870934 0.67385926]
Mean Score : 0.6894022864677135
Difference : -0.17027968245384828
```



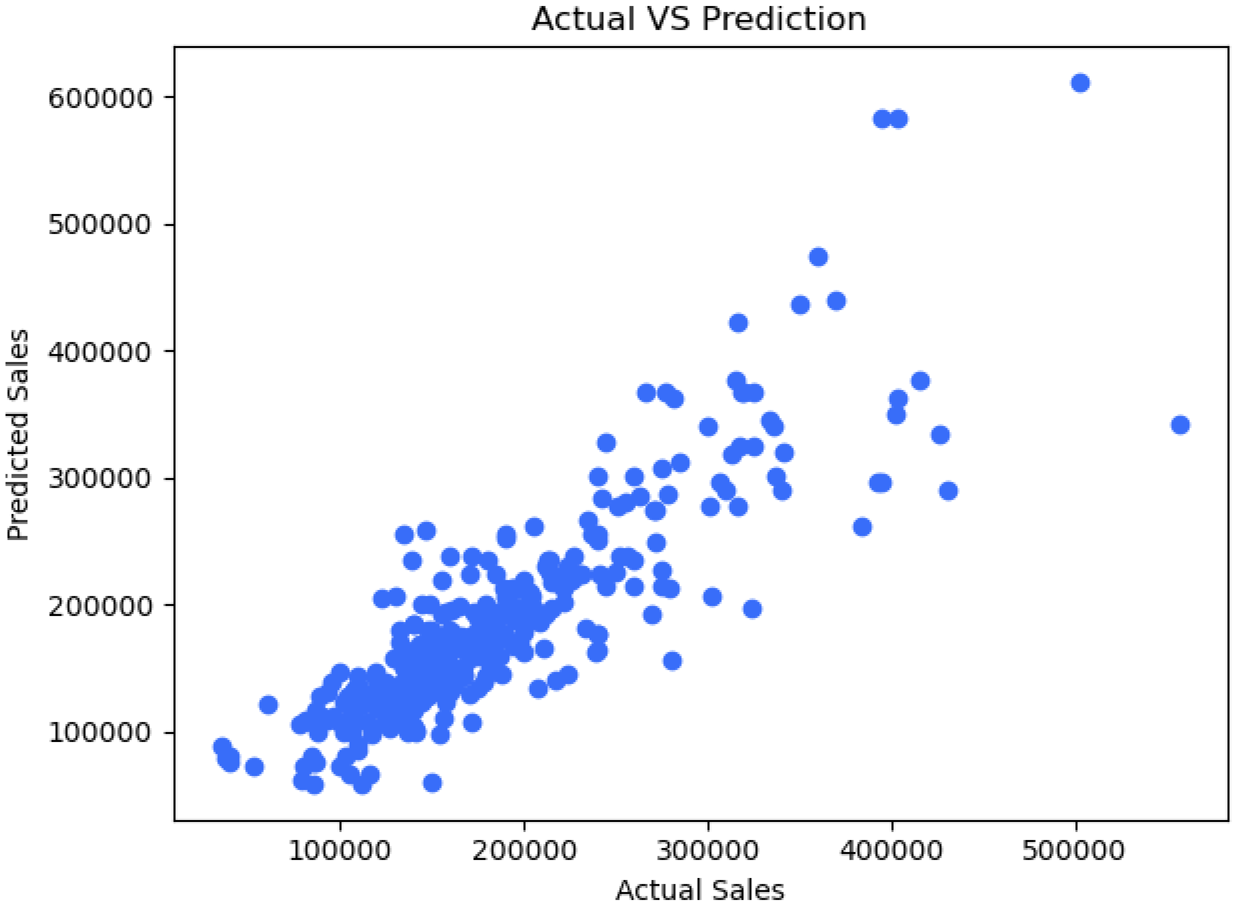

```
==============================================
```

```
====================================================
Scores for SVR() are
Mean Absolute Error: 56245.852821355744
Mean Squared Error: 6465090730.931606
Root Mean Squared Error: 80405.78791935073
R squared score:  -0.056949684565039016
Cross Validation Score is : [-0.05457259 -0.05994692 -0.02257844 -0.05773039 -0.0423051
6]
Mean Score : -0.04742669853437089
Difference : 0.009522986030668125
```



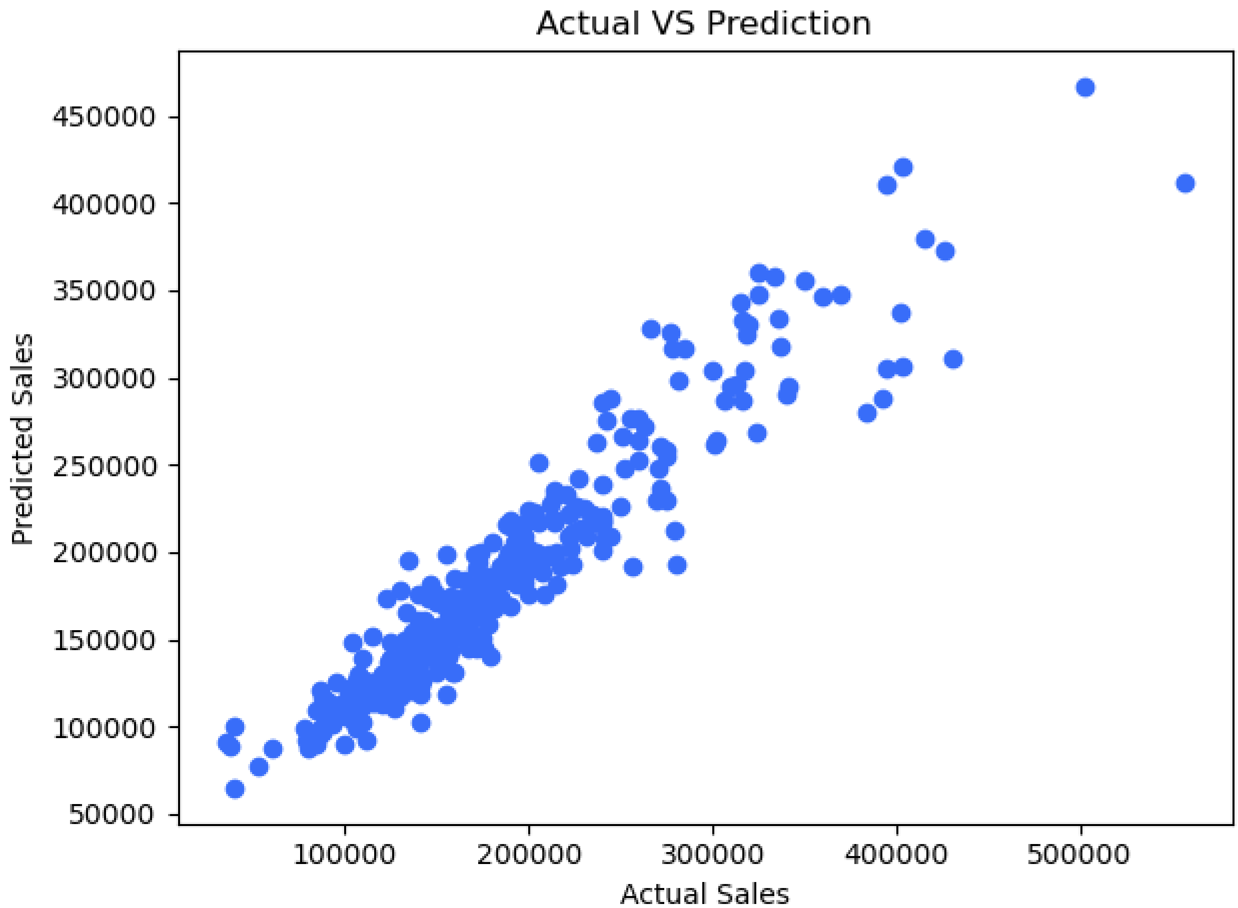

```
==============================================


=====================================================
Scores for DecisionTreeRegressor() are
Mean Absolute Error: 28722.390313390315
Mean Squared Error: 1754425529.974359
Root Mean Squared Error: 41885.86312796191
R squared score:  0.7131765681760132
Cross Validation Score is : [0.70603413 0.68612206 0.77645277 0.64930986 0.6233076 ]
Mean Score : 0.6882452832659839
Difference : -0.024931284910029272
```




```
===============================================


=====================================================
Scores for RandomForestRegressor() are
Mean Absolute Error: 17381.24358974359
Mean Squared Error: 653870812.5890375
Root Mean Squared Error: 25570.897766582963
R squared score:  0.8931014926355594
Cross Validation Score is : [0.88497492 0.82282216 0.76144476 0.83801195 0.7759498 ]
Mean Score : 0.8166407181087424
Difference : -0.07646077452681699
```




```
===============================================
```

```
=====================================================
```

Random forest regressor has given the best result with R2 score= 0.8962927546604891 and mean CV Score = 0.8213496578533995 so we will proceed with that.

Hyperparameter Tuning.

In [46]:
```
rfr.get_params()
```

Out[46]:
```
{'bootstrap': True,
 'ccp_alpha': 0.0,
 'criterion': 'squared_error',
 'max_depth': None,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': None,
 'verbose': 0,
 'warm_start': False}
```

In [47]:
```
param = {'n_estimators':range(0,100,10),
         'criterion':['friedman_mse', 'squared_error', 'absolute_error', 'poisson'],
         'random_state':range(1,10),
         'max_features':['auto', 'sqrt']}
```

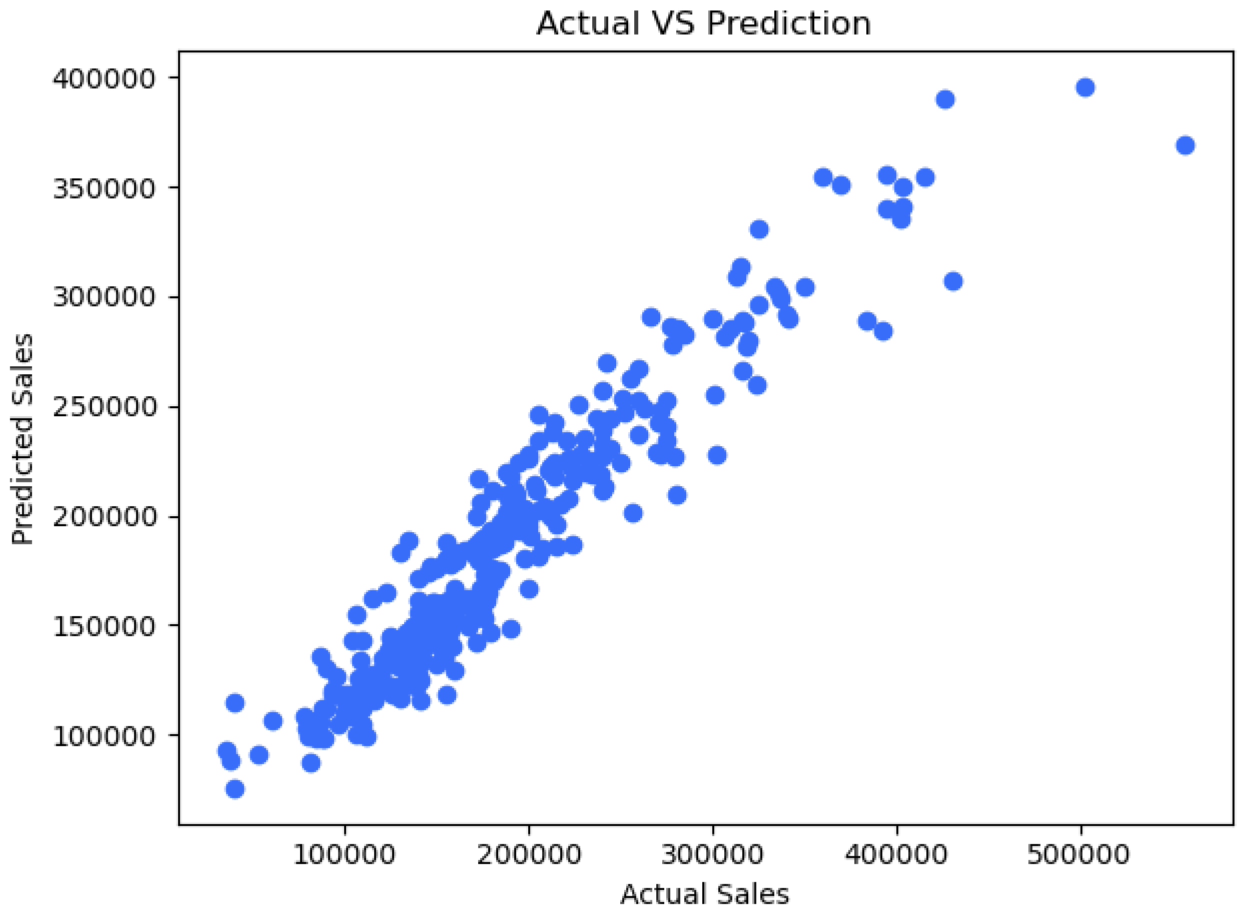
In [48]:
```
grid = GridSearchCV(rfr,param_grid = param)
grid.fit(train_x,train_y)
grid.best_params_
```

Out[48]:
```
{'criterion': 'absolute_error',
 'max_features': 'sqrt',
 'n_estimators': 70,
 'random_state': 2}
```

In [52]:
```
rfr_hyp=RandomForestRegressor(n_estimators= 70 ,criterion='absolute_error', random_state
```

In [53]:
```
rfr_hyp.fit(train_x,train_y)
rfr_hyp.score(train_x,train_y)
predm=rfr_hyp.predict(test_x)
print(f"Scores for {rfr_hyp} are")
print('Mean Absolute Error:', metrics.mean_absolute_error(test_y, predm))
print('Mean Squared Error:', metrics.mean_squared_error(test_y, predm))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(test_y, predm)))
print('R squared score: ', r2_score(test_y, predm))
score=cross_val_score(rfr_hyp,train_x,train_y,cv=5)
print("Cross Validation Score is :",score)
print("Mean Score :",score.mean())
print("Difference :",score.mean()-r2_score(test_y, predm))
plt.scatter(test_y, predm)
plt.xlabel("Actual Sales")
plt.ylabel("Predicted Sales")
plt.title("Actual VS Prediction")
plt.show()
```

```
Scores for RandomForestRegressor(criterion='absolute_error', max_features='sqrt',
                      n_estimators=70, random_state=2) are
Mean Absolute Error: 18035.69641839642
Mean Squared Error: 729711345.5527548
Root Mean Squared Error: 27013.16985384638
R squared score:  0.8807026523517364
Cross Validation Score is : [0.90310692 0.83471074 0.79509746 0.84475042 0.79309378]
Mean Score : 0.8341518638572382
Difference : -0.04655078849449823
```



In [73]:
```
import joblib
joblib.dump(rfr_hyp,"House_rfr.obj")
```

Out[73]:
```
['House_rfr.obj']
```

Testing Dataset.

In [82]:
```
df_test = pd.read_csv('test.csv')
df_test
```

Out[82]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotCo
0	337	20	RL	86.0	14157	Pave	NaN	IR1	HLS	AllPub	Cc
1	1018	120	RL	NaN	5814	Pave	NaN	IR1	Lvl	AllPub	CulI
2	929	20	RL	NaN	11838	Pave	NaN	Reg	Lvl	AllPub	Ir
3	1148	70	RL	75.0	12000	Pave	NaN	Reg	Bnk	AllPub	Ir
4	1227	60	RL	86.0	14598	Pave	NaN	IR1	Lvl	AllPub	CulI
5	650	180	RM	21.0	1936	Pave	NaN	Reg	Lvl	AllPub	Ir
6	1453	180	RM	35.0	3675	Pave	NaN	Reg	Lvl	AllPub	Ir
7	152	20	RL	107.0	13891	Pave	NaN	Reg	Lvl	AllPub	Ir
8	427	80	RL	NaN	12800	Pave	NaN	Reg	Low	AllPub	Ir
9	776	120	RM	32.0	4500	Pave	NaN	Reg	Lvl	AllPub	
10	30	30	RM	60.0	6324	Pave	NaN	IR1	Lvl	AllPub	Ir
11	1425	20	RL	NaN	9503	Pave	NaN	Reg	Lvl	AllPub	Ir

12	423	20	RL	100.0	21750	Pave	NaN	Reg	HLS	AllPub	Ir
13	1185	20	RL	50.0	35133	Grvl	NaN	Reg	Lvl	AllPub	Ir
14	775	20	RL	110.0	14226	Pave	NaN	Reg	Lvl	AllPub	Cc
15	391	50	RL	50.0	8405	Pave	Grvl	Reg	Lvl	AllPub	Ir
16	1408	20	RL	NaN	8780	Pave	NaN	IR1	Lvl	AllPub	Cc
17	513	20	RL	70.0	9100	Pave	NaN	Reg	Lvl	AllPub	Cc
18	1266	160	FV	35.0	3735	Pave	NaN	Reg	Lvl	AllPub	
19	173	160	RL	44.0	5306	Pave	NaN	IR1	Lvl	AllPub	Ir
20	1150	70	RM	50.0	9000	Pave	NaN	Reg	Lvl	AllPub	Ir
21	797	20	RL	71.0	8197	Pave	NaN	Reg	Lvl	AllPub	Ir
22	137	20	RL	NaN	10355	Pave	NaN	IR1	Lvl	AllPub	Cc
23	706	190	RM	70.0	5600	Pave	NaN	Reg	Lvl	AllPub	Ir
24	1377	30	RL	52.0	6292	Pave	NaN	Reg	Bnk	AllPub	Ir
25	1177	20	RL	37.0	6951	Pave	NaN	IR1	Lvl	AllPub	CulI
26	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	Ir
27	369	20	RL	78.0	7800	Pave	NaN	Reg	Lvl	AllPub	Ir
28	1421	60	RL	90.0	11700	Pave	NaN	Reg	Lvl	AllPub	Cc
29	999	30	RM	60.0	9786	Pave	NaN	Reg	Lvl	AllPub	Ir
30	1217	90	RM	68.0	8930	Pave	NaN	Reg	Lvl	AllPub	Ir
31	937	20	RL	67.0	10083	Pave	NaN	Reg	Lvl	AllPub	Ir
32	769	20	RL	70.0	9100	Pave	NaN	Reg	Lvl	AllPub	Ir
33	831	20	RL	80.0	11900	Pave	NaN	IR1	Lvl	AllPub	Cc
34	678	30	RL	52.0	9022	Pave	NaN	Reg	Lvl	AllPub	Ir
35	574	80	RL	76.0	9967	Pave	NaN	IR1	Lvl	AllPub	Ir
36	921	60	RL	70.0	8462	Pave	NaN	IR1	Lvl	AllPub	Ir
37	1292	160	RM	21.0	1680	Pave	NaN	Reg	Lvl	AllPub	Ir
38	1277	60	RL	NaN	12936	Pave	NaN	IR1	Lvl	AllPub	CulI
39	676	160	RL	24.0	2289	Pave	NaN	Reg	Lvl	AllPub	Ir
40	108	20	RM	50.0	6000	Pave	NaN	Reg	Lvl	AllPub	Ir
41	424	60	RL	80.0	9200	Pave	NaN	Reg	Lvl	AllPub	Ir
42	823	60	RL	NaN	12394	Pave	NaN	IR1	Lvl	AllPub	Cc
43	1455	20	FV	62.0	7500	Pave	Pave	Reg	Lvl	AllPub	Ir
44	377	85	RL	57.0	8846	Pave	NaN	IR1	Lvl	AllPub	CulI
45	1256	50	RM	52.0	6240	Pave	NaN	Reg	Lvl	AllPub	Ir
46	1120	20	RL	70.0	7560	Pave	NaN	Reg	Lvl	AllPub	Ir
47	265	30	RM	30.0	5232	Pave	Grvl	IR3	Bnk	AllPub	Ir
48	1158	120	RL	34.0	5001	Pave	NaN	IR1	Lvl	AllPub	Ir
49	725	20	RL	86.0	13286	Pave	NaN	IR1	Lvl	AllPub	Ir

50	1378	50	RL	60.0	10998	Pave	Grvl	Reg	Lvl	AllPub	Ir
51	1139	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	Ir
52	122	50	RM	50.0	6060	Pave	NaN	Reg	Lvl	AllPub	Ir
53	515	45	RL	55.0	10594	Pave	NaN	Reg	Lvl	AllPub	Ir
54	518	60	RL	79.0	10208	Pave	NaN	IR1	Lvl	AllPub	Ir
55	1214	80	RL	NaN	10246	Pave	NaN	IR1	Lvl	AllPub	CulD
56	443	50	RM	52.0	6240	Pave	NaN	Reg	Lvl	AllPub	Ir
57	903	60	RL	63.0	7875	Pave	NaN	Reg	Lvl	AllPub	Ir
58	53	90	RM	110.0	8472	Grvl	NaN	IR2	Bnk	AllPub	Co
59	469	20	RL	98.0	11428	Pave	NaN	IR1	Lvl	AllPub	Ir
60	76	180	RM	21.0	1596	Pave	NaN	Reg	Lvl	AllPub	Ir
61	1142	60	RL	NaN	10304	Pave	NaN	IR1	Lvl	AllPub	CulD
62	1222	20	RL	55.0	8250	Pave	NaN	Reg	Lvl	AllPub	Ir
63	851	120	RM	36.0	4435	Pave	NaN	Reg	Lvl	AllPub	Ir
64	334	120	RM	59.0	8198	Pave	NaN	Reg	Lvl	AllPub	
65	1008	160	RM	21.0	2217	Pave	NaN	Reg	Lvl	AllPub	Ir
66	19	20	RL	66.0	13695	Pave	NaN	Reg	Lvl	AllPub	Ir
67	339	20	RL	91.0	14145	Pave	NaN	Reg	Lvl	AllPub	Co
68	1118	20	RL	57.0	9764	Pave	NaN	IR1	Lvl	AllPub	
69	834	20	RL	100.0	10004	Pave	NaN	Reg	Lvl	AllPub	Ir
70	1176	50	RL	85.0	10678	Pave	NaN	Reg	Lvl	AllPub	Ir
71	945	20	RL	NaN	14375	Pave	NaN	IR1	Lvl	NoSeWa	CulD
72	1415	50	RL	64.0	13053	Pave	Pave	Reg	Bnk	AllPub	Ir
73	911	90	RL	80.0	11600	Pave	NaN	Reg	Lvl	AllPub	Co
74	388	80	RL	72.0	7200	Pave	NaN	Reg	Lvl	AllPub	Ir
75	747	60	RL	NaN	8795	Pave	NaN	IR1	Lvl	AllPub	Ir
76	620	60	RL	85.0	12244	Pave	NaN	Reg	Lvl	AllPub	Ir
77	386	120	RL	43.0	3182	Pave	NaN	Reg	Lvl	AllPub	Ir
78	818	20	RL	NaN	13265	Pave	NaN	IR1	Lvl	AllPub	CulD
79	539	20	RL	NaN	11553	Pave	NaN	IR1	Lvl	AllPub	Ir
80	535	60	RL	74.0	9056	Pave	NaN	IR1	Lvl	AllPub	Ir
81	1399	50	RL	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Ir
82	571	90	RL	74.0	13101	Pave	NaN	IR1	Lvl	AllPub	Ir
83	38	20	RL	74.0	8532	Pave	NaN	Reg	Lvl	AllPub	Ir
84	850	80	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	
85	1032	75	RL	102.0	15863	Pave	NaN	Reg	Lvl	AllPub	Co
86	1433	30	RL	60.0	10800	Pave	Grvl	Reg	Lvl	AllPub	Ir

87	1354	50	RL	56.0	14720	Pave	NaN	IR1	Lvl	AllPub	Cul
88	1072	60	RL	78.0	11700	Pave	NaN	Reg	Lvl	AllPub	Ir
89	371	60	RL	NaN	8121	Pave	NaN	IR1	Lvl	AllPub	Ir
90	1242	20	RL	83.0	9849	Pave	NaN	Reg	Lvl	AllPub	Ir
91	681	120	RL	50.0	8012	Pave	NaN	Reg	Lvl	AllPub	Ir
92	290	70	RL	60.0	8730	Pave	NaN	Reg	Lvl	AllPub	Ir
93	973	120	RL	55.0	7892	Pave	NaN	Reg	Lvl	AllPub	Ir
94	989	60	RL	NaN	12046	Pave	NaN	IR1	Lvl	AllPub	Ir
95	484	120	RM	32.0	4500	Pave	NaN	Reg	Lvl	AllPub	
96	1240	20	RL	64.0	9037	Pave	NaN	IR1	HLS	AllPub	Ir
97	1125	80	RL	NaN	9125	Pave	NaN	IR1	Lvl	AllPub	Ir
98	1143	60	RL	77.0	9965	Pave	NaN	Reg	Lvl	AllPub	Ir
99	1340	20	RL	120.0	9560	Pave	NaN	IR1	Lvl	AllPub	Cc
100	1343	60	RL	NaN	9375	Pave	NaN	Reg	Lvl	AllPub	Ir
101	936	30	RL	52.0	5825	Pave	NaN	IR1	Lvl	AllPub	Ir
102	1151	20	RL	57.0	8280	Pave	NaN	IR1	Lvl	AllPub	Ir
103	1380	80	RL	73.0	9735	Pave	NaN	Reg	Lvl	AllPub	Ir
104	1190	60	RL	60.0	7500	Pave	NaN	Reg	Lvl	AllPub	Ir
105	635	90	RL	64.0	6979	Pave	NaN	Reg	Lvl	AllPub	Ir
106	47	50	RL	48.0	12822	Pave	NaN	IR1	Lvl	AllPub	Cul
107	729	90	RL	85.0	11475	Pave	NaN	Reg	Lvl	AllPub	Cc
108	1434	60	RL	93.0	10261	Pave	NaN	IR1	Lvl	AllPub	Ir
109	472	60	RL	92.0	11952	Pave	NaN	Reg	Lvl	AllPub	Ir
110	1156	20	RL	90.0	10768	Pave	NaN	IR1	Lvl	AllPub	Cc
111	1352	60	RL	70.0	9247	Pave	NaN	IR1	Lvl	AllPub	Ir
112	717	70	RM	60.0	10800	Pave	Grvl	Reg	Bnk	AllPub	Ir
113	385	60	RL	NaN	53107	Pave	NaN	IR2	Low	AllPub	Cc
114	1334	50	RM	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Cc
115	243	50	RM	63.0	5000	Pave	NaN	Reg	Lvl	AllPub	Cc
116	39	20	RL	68.0	7922	Pave	NaN	Reg	Lvl	AllPub	Ir
117	1168	60	RL	58.0	10852	Pave	NaN	IR1	Lvl	AllPub	Ir
118	214	20	RL	43.0	13568	Pave	NaN	IR2	Lvl	AllPub	Cul
119	647	20	RL	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Ir
120	490	180	RM	21.0	1526	Pave	NaN	Reg	Lvl	AllPub	Ir
121	512	120	RL	40.0	6792	Pave	NaN	IR1	Lvl	AllPub	Ir
122	1181	60	RL	NaN	11170	Pave	NaN	IR2	Lvl	AllPub	Cc
123	1451	90	RL	60.0	9000	Pave	NaN	Reg	Lvl	AllPub	
124	1428	50	RL	60.0	10930	Pave	Grvl	Reg	Bnk	AllPub	Ir

125	767	60	RL	80.0	10421	Pave	NaN	Reg	Lvl	AllPub	Ir
126	1250	20	RL	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Ir
127	855	20	RL	102.0	17920	Pave	NaN	Reg	Lvl	AllPub	Ir
128	1001	20	RL	74.0	10206	Pave	NaN	Reg	Lvl	AllPub	Cc
129	49	190	RM	33.0	4456	Pave	NaN	Reg	Lvl	AllPub	Ir
130	25	20	RL	NaN	8246	Pave	NaN	IR1	Lvl	AllPub	Ir
131	1058	60	RL	NaN	29959	Pave	NaN	IR2	Lvl	AllPub	
132	24	120	RM	44.0	4224	Pave	NaN	Reg	Lvl	AllPub	Ir
133	758	60	RL	NaN	11616	Pave	NaN	IR1	Lvl	AllPub	CulD
134	1060	50	RL	NaN	11275	Pave	NaN	IR1	HLS	AllPub	Cc
135	1110	20	RL	107.0	11362	Pave	NaN	IR1	Lvl	AllPub	Ir
136	1057	120	RL	43.0	7052	Pave	NaN	IR1	Lvl	AllPub	Ir
137	491	160	RM	NaN	2665	Pave	NaN	Reg	Lvl	AllPub	Ir
138	378	60	FV	102.0	11143	Pave	NaN	IR1	Lvl	AllPub	Cc
139	1429	30	RM	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Cc
140	55	80	RL	60.0	7134	Pave	NaN	Reg	Bnk	AllPub	Ir
141	770	60	RL	47.0	53504	Pave	NaN	IR2	HLS	AllPub	CulD
142	737	90	RL	60.0	8544	Pave	NaN	Reg	Lvl	AllPub	Ir
143	59	60	RL	66.0	13682	Pave	NaN	IR2	HLS	AllPub	CulD
144	29	20	RL	47.0	16321	Pave	NaN	IR1	Lvl	AllPub	CulD
145	788	60	RL	76.0	10142	Pave	NaN	IR1	Lvl	AllPub	Ir
146	9	50	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	Ir
147	1371	50	RL	90.0	5400	Pave	NaN	Reg	Lvl	AllPub	Cc
148	260	20	RM	70.0	12702	Pave	NaN	Reg	Lvl	AllPub	Ir
149	249	60	RL	72.0	11317	Pave	NaN	Reg	Lvl	AllPub	Ir
150	88	160	FV	40.0	3951	Pave	Pave	Reg	Lvl	AllPub	Cc
151	893	20	RL	70.0	8414	Pave	NaN	Reg	Lvl	AllPub	Ir
152	803	60	RL	63.0	8199	Pave	NaN	Reg	Lvl	AllPub	Ir
153	901	20	RL	NaN	7340	Pave	NaN	IR1	Lvl	AllPub	Ir
154	1124	20	RL	50.0	9405	Pave	NaN	Reg	Lvl	AllPub	Ir
155	223	60	RL	85.0	11475	Pave	NaN	Reg	Lvl	AllPub	Ir
156	1278	80	RL	NaN	17871	Pave	NaN	IR1	Lvl	AllPub	CulD
157	1174	50	RL	138.0	18030	Pave	NaN	IR1	Bnk	AllPub	Ir
158	312	20	RL	50.0	8000	Pave	NaN	Reg	Lvl	AllPub	Ir
159	624	160	FV	NaN	2117	Pave	NaN	Reg	Lvl	AllPub	Ir
160	1445	20	RL	63.0	8500	Pave	NaN	Reg	Lvl	AllPub	
161	296	80	RL	37.0	7937	Pave	NaN	IR1	Lvl	AllPub	CulD

162	753	20	RL	79.0	9236	Pave	NaN	IR1	Lvl	AllPub	Ir
163	1320	20	RL	75.0	10215	Pave	NaN	Reg	Bnk	AllPub	Ir
164	432	50	RM	60.0	5586	Pave	NaN	IR1	Bnk	AllPub	Ir
165	1362	20	RL	124.0	16158	Pave	NaN	IR1	Low	AllPub	Ir
166	1069	160	RM	42.0	3964	Pave	NaN	Reg	Lvl	AllPub	Ir
167	1255	60	RL	60.0	6931	Pave	NaN	Reg	Lvl	AllPub	Ir
168	50	20	RL	66.0	7742	Pave	NaN	Reg	Lvl	AllPub	Ir
169	454	60	FV	75.0	9000	Pave	NaN	Reg	Lvl	AllPub	Ir
170	881	20	RL	60.0	7024	Pave	NaN	Reg	Lvl	AllPub	Ir
171	500	20	RL	70.0	7535	Pave	NaN	IR1	Lvl	AllPub	Ir
172	807	80	RL	75.0	9750	Pave	NaN	Reg	Lvl	AllPub	Ir
173	766	20	RL	75.0	14587	Pave	NaN	IR1	Lvl	AllPub	Ir
174	866	20	RL	NaN	8750	Pave	NaN	IR1	Lvl	AllPub	Ir
175	162	60	RL	110.0	13688	Pave	NaN	IR1	Lvl	AllPub	Ir
176	1193	50	RM	60.0	9600	Pave	Grvl	Reg	Lvl	AllPub	Ir
177	798	20	RL	57.0	7677	Pave	NaN	Reg	Lvl	AllPub	Ir
178	1400	50	RL	51.0	6171	Pave	NaN	Reg	Lvl	AllPub	Ir
179	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub	Ir
180	58	60	RL	89.0	11645	Pave	NaN	IR1	Lvl	AllPub	Cc
181	1435	20	RL	80.0	17400	Pave	NaN	Reg	Low	AllPub	Ir
182	867	20	RL	67.0	10656	Pave	NaN	IR1	HLS	AllPub	Ir
183	960	160	FV	24.0	2572	Pave	NaN	Reg	Lvl	AllPub	
184	441	20	RL	105.0	15431	Pave	NaN	Reg	Lvl	AllPub	Ir
185	481	20	RL	98.0	16033	Pave	NaN	IR1	Lvl	AllPub	
186	219	50	RL	NaN	15660	Pave	NaN	IR1	Lvl	AllPub	Cc
187	128	45	RM	55.0	4388	Pave	NaN	IR1	Bnk	AllPub	Ir
188	858	60	RL	65.0	8125	Pave	NaN	Reg	Lvl	AllPub	Ir
189	215	60	RL	NaN	10900	Pave	NaN	IR1	Lvl	AllPub	
190	1030	160	RM	21.0	1680	Pave	NaN	Reg	Lvl	AllPub	Ir
191	1204	20	RL	75.0	9750	Pave	NaN	Reg	Lvl	AllPub	Ir
192	531	80	RL	85.0	10200	Pave	NaN	Reg	Lvl	AllPub	Ir
193	529	30	RL	58.0	9098	Pave	NaN	IR1	Lvl	AllPub	Ir
194	346	50	RL	65.0	6435	Pave	NaN	Reg	Lvl	AllPub	Ir
195	621	30	RL	45.0	8248	Pave	Grvl	Reg	Lvl	AllPub	Ir
196	1194	120	RM	NaN	4500	Pave	NaN	Reg	Lvl	AllPub	
197	1000	20	RL	64.0	6762	Pave	NaN	Reg	Lvl	AllPub	Ir
198	761	20	RL	70.0	9100	Pave	NaN	Reg	Lvl	AllPub	Ir
199	1351	90	RL	91.0	11643	Pave	NaN	Reg	Lvl	AllPub	Ir

200	1442	120	RM	NaN	4426	Pave	NaN	Reg	Lvl	AllPub	Ir
201	233	160	RM	21.0	1680	Pave	NaN	Reg	Lvl	AllPub	Ir
202	360	60	RL	78.0	12011	Pave	NaN	IR1	Lvl	AllPub	Cul
203	886	120	FV	50.0	5119	Pave	NaN	IR1	Lvl	AllPub	Cul
204	11	20	RL	70.0	11200	Pave	NaN	Reg	Lvl	AllPub	Ir
205	572	20	RL	60.0	7332	Pave	NaN	Reg	Lvl	AllPub	Ir
206	1375	60	FV	85.0	10625	Pave	NaN	Reg	Lvl	AllPub	Ir
207	328	20	RL	80.0	11600	Pave	NaN	Reg	Lvl	AllPub	Ir
208	1417	190	RM	60.0	11340	Pave	NaN	Reg	Lvl	AllPub	Ir
209	1089	160	RM	24.0	2522	Pave	NaN	Reg	Lvl	AllPub	Ir
210	242	30	RM	40.0	3880	Pave	NaN	Reg	Lvl	AllPub	Ir
211	1198	75	RM	65.0	8850	Pave	NaN	IR1	Bnk	AllPub	Cc
212	492	50	RL	79.0	9490	Pave	NaN	Reg	Lvl	AllPub	Ir
213	743	20	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Ir
214	1258	30	RL	56.0	4060	Pave	NaN	Reg	Lvl	AllPub	Cc
215	1350	70	RM	50.0	5250	Pave	Pave	Reg	Lvl	AllPub	Ir
216	722	120	RM	NaN	4426	Pave	NaN	Reg	Lvl	AllPub	Ir
217	442	90	RL	92.0	12108	Pave	NaN	Reg	Lvl	AllPub	Ir
218	90	20	RL	60.0	8070	Pave	NaN	Reg	Lvl	AllPub	Ir
219	1007	20	RL	NaN	12155	Pave	NaN	IR3	Lvl	AllPub	Ir
220	1241	60	RL	65.0	8158	Pave	NaN	Reg	Lvl	AllPub	Ir
221	144	20	RL	78.0	10335	Pave	NaN	IR1	Lvl	AllPub	Ir
222	1317	20	RL	61.0	10226	Pave	NaN	IR1	Lvl	AllPub	Ir
223	1080	20	RL	65.0	8775	Pave	NaN	Reg	Lvl	AllPub	Ir
224	1160	60	RL	76.0	9120	Pave	NaN	Reg	Lvl	AllPub	Ir
225	482	20	RL	72.0	11846	Pave	NaN	IR1	HLS	AllPub	Ir
226	33	20	RL	85.0	11049	Pave	NaN	Reg	Lvl	AllPub	Cc
227	1280	50	C (all)	60.0	7500	Pave	NaN	Reg	Lvl	AllPub	Ir
228	541	20	RL	85.0	14601	Pave	NaN	Reg	Lvl	AllPub	Ir
229	994	60	RL	68.0	8846	Pave	NaN	Reg	Lvl	AllPub	Ir
230	1385	50	RL	60.0	9060	Pave	NaN	Reg	Lvl	AllPub	Ir
231	792	80	RL	NaN	11333	Pave	NaN	IR1	Lvl	AllPub	Cc
232	1002	30	RL	60.0	5400	Pave	NaN	Reg	Lvl	AllPub	Cc
233	420	20	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Ir
234	354	30	RM	60.0	8520	Pave	NaN	Reg	Lvl	AllPub	Ir
235	447	20	RL	137.0	16492	Pave	NaN	IR1	Lvl	AllPub	Cc
236	883	60	RL	NaN	9636	Pave	NaN	IR1	Lvl	AllPub	Cc

237	578	80	RL	96.0	11777	Pave	NaN	IR1	Lvl	AllPub	Ir
238	1251	20	RL	93.0	11160	Pave	NaN	Reg	Lvl	AllPub	Cc
239	671	60	RL	64.0	8633	Pave	NaN	Reg	Lvl	AllPub	
240	1017	20	RL	73.0	11883	Pave	NaN	Reg	Lvl	AllPub	Ir
241	780	90	RL	78.0	10530	Pave	NaN	Reg	Lvl	AllPub	Ir
242	1269	50	RL	NaN	14100	Pave	NaN	IR1	Lvl	AllPub	Ir
243	1145	190	RL	60.0	12180	Pave	NaN	Reg	Lvl	AllPub	Ir
244	486	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	Ir
245	134	20	RL	NaN	6853	Pave	NaN	IR1	Lvl	AllPub	Ir
246	1282	20	RL	50.0	8049	Pave	NaN	IR1	Lvl	AllPub	CulI
247	693	60	RL	42.0	26178	Pave	NaN	IR1	Lvl	AllPub	Ir
248	1119	80	RL	85.0	13825	Pave	NaN	Reg	Lvl	AllPub	Ir
249	408	70	RL	63.0	15576	Pave	NaN	Reg	Lvl	AllPub	Ir
250	1288	20	RL	NaN	36500	Pave	NaN	IR1	Low	AllPub	Ir
251	159	60	FV	100.0	12552	Pave	NaN	Reg	Lvl	AllPub	Cc
252	577	50	RL	52.0	6292	Pave	NaN	Reg	Lvl	AllPub	Ir
253	1014	30	RM	60.0	7200	Pave	NaN	Reg	Lvl	AllPub	Ir
254	226	160	RM	21.0	1680	Pave	NaN	Reg	Lvl	AllPub	Ir
255	140	60	RL	65.0	15426	Pave	NaN	IR1	Lvl	AllPub	Ir
256	109	50	RM	85.0	8500	Pave	NaN	Reg	Lvl	AllPub	Cc
257	988	20	RL	83.0	10159	Pave	NaN	IR1	Lvl	AllPub	Ir
258	633	20	RL	85.0	11900	Pave	NaN	Reg	Lvl	AllPub	Ir
259	1221	20	RL	66.0	7800	Pave	NaN	IR1	Lvl	AllPub	Ir
260	1431	60	RL	60.0	21930	Pave	NaN	IR3	Lvl	AllPub	Ir
261	365	60	RL	NaN	18800	Pave	NaN	IR1	Lvl	AllPub	
262	495	30	RM	50.0	5784	Pave	NaN	Reg	Lvl	AllPub	Ir
263	165	40	RM	40.0	5400	Pave	Pave	Reg	Lvl	AllPub	Cc
264	816	20	RL	48.0	12137	Pave	NaN	IR2	Lvl	AllPub	CulI
265	181	160	FV	NaN	2117	Pave	NaN	Reg	Lvl	AllPub	Ir
266	107	30	RM	60.0	10800	Pave	Grvl	Reg	Lvl	AllPub	Ir
267	125	20	RL	48.0	17043	Pave	NaN	IR1	Lvl	AllPub	CulI
268	1218	20	FV	72.0	8640	Pave	NaN	Reg	Lvl	AllPub	Ir
269	938	60	RL	75.0	9675	Pave	NaN	Reg	Lvl	AllPub	Ir
270	372	50	RL	80.0	17120	Pave	NaN	Reg	Lvl	AllPub	Ir
271	45	20	RL	70.0	7945	Pave	NaN	Reg	Lvl	AllPub	Ir
272	995	20	RL	96.0	12456	Pave	NaN	Reg	Lvl	AllPub	
273	314	20	RL	150.0	215245	Pave	NaN	IR3	Low	AllPub	Ir
274	948	20	RL	85.0	14536	Pave	NaN	Reg	Lvl	AllPub	Ir

275	217	20	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Ir
276	781	20	RL	63.0	7875	Pave	NaN	Reg	Lvl	AllPub	Ir
277	1293	70	RM	60.0	6600	Pave	NaN	Reg	Lvl	AllPub	Cc
278	1005	120	RL	43.0	3182	Pave	NaN	Reg	Lvl	AllPub	Ir
279	528	60	RL	67.0	14948	Pave	NaN	IR1	Lvl	AllPub	Ir
280	44	20	RL	NaN	9200	Pave	NaN	IR1	Lvl	AllPub	Cull
281	645	20	FV	85.0	9187	Pave	NaN	Reg	Lvl	AllPub	Ir
282	340	20	RL	66.0	12400	Pave	NaN	IR1	Lvl	AllPub	Ir
283	1116	20	RL	93.0	12085	Pave	NaN	Reg	Lvl	AllPub	Ir
284	358	120	RM	44.0	4224	Pave	NaN	Reg	Lvl	AllPub	Ir
285	72	20	RL	69.0	7599	Pave	NaN	Reg	Lvl	AllPub	Cc
286	56	20	RL	100.0	10175	Pave	NaN	IR1	Lvl	AllPub	Ir
287	83	20	RL	78.0	10206	Pave	NaN	Reg	Lvl	AllPub	Ir
288	1048	20	RL	57.0	9245	Pave	NaN	IR2	Lvl	AllPub	Ir
289	17	20	RL	NaN	11241	Pave	NaN	IR1	Lvl	AllPub	Cull
290	523	50	RM	50.0	5000	Pave	NaN	Reg	Lvl	AllPub	Cc
291	1379	160	RM	21.0	1953	Pave	NaN	Reg	Lvl	AllPub	Ir

Tweaking the test dataset according to train dataset so that it can fit the model.

In [83]:
```
df_test.drop(columns = ['MiscFeature','PoolQC','Alley','Fence'],axis = 1, inplace = True
```

In [84]:
```
df.isin(['NA','N/A','-',' ','?',' ?']).sum().any()
```

Out[84]:
```
False
```

In [85]:
```
df_test.isnull().sum()
```

Out[85]:
```
Id               0
MSSubClass       0
MSZoning         0
LotFrontage     45
LotArea          0
Street           0
LotShape         0
LandContour      0
Utilities        0
LotConfig        0
LandSlope        0
Neighborhood     0
Condition1       0
Condition2       0
BldgType         0
HouseStyle       0
OverallQual      0
OverallCond      0
YearBuilt        0
YearRemodAdd     0
RoofStyle        0
RoofMatl         0
Exterior1st      0
```

```
Exterior2nd        0
MasVnrType         1
MasVnrArea         1
ExterQual          0
ExterCond          0
Foundation         0
BsmtQual           7
BsmtCond           7
BsmtExposure       7
BsmtFinType1       7
BsmtFinSF1         0
BsmtFinType2       7
BsmtFinSF2         0
BsmtUnfSF          0
TotalBsmtSF        0
Heating            0
HeatingQC          0
CentralAir         0
Electrical         1
1stFlrSF           0
2ndFlrSF           0
LowQualFinSF       0
GrLivArea          0
BsmtFullBath       0
BsmtHalfBath       0
FullBath           0
HalfBath           0
BedroomAbvGr       0
KitchenAbvGr       0
KitchenQual        0
TotRmsAbvGrd       0
Functional         0
Fireplaces         0
FireplaceQu      139
GarageType        17
GarageYrBlt       17
GarageFinish      17
GarageCars         0
GarageArea         0
GarageQual        17
GarageCond        17
PavedDrive         0
WoodDeckSF         0
OpenPorchSF        0
EnclosedPorch      0
3SsnPorch          0
ScreenPorch        0
PoolArea           0
MiscVal            0
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
dtype: int64
```

In [86]:

```
df_test['MasVnrArea'] = df_test['MasVnrArea'].fillna(df_test['MasVnrArea'].mean())

df_test['LotFrontage'] = df_test['LotFrontage'].fillna(df_test['LotFrontage'].median())
df_test['GarageYrBlt'] = df_test['GarageYrBlt'].fillna(df_test['GarageYrBlt'].median())

for x in ['MasVnrType','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2
'GarageFinish','GarageQual','GarageCond','Electrical']:
    df_test[x] = df_test[x].fillna(df_test[x].mode()[0])
```

In [87]:

```
df_test.drop('Utilities',axis = 1, inplace = True )
```

In [88]:
```
df_test.drop('Id',axis = 1, inplace = True )
```

In [89]:
```
for i in Categorical_features:
    df_test[i] = le.fit_transform(df_test[i])
df_test.head()
```

Out[89]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighb
0	20	2	86.0	14157	1	0	1	0	0	
1	120	2	65.0	5814	1	0	3	1	0	
2	20	2	65.0	11838	1	3	3	4	0	
3	70	2	75.0	12000	1	3	0	4	0	
4	60	2	86.0	14598	1	0	3	1	0	

In [90]:
```
df_test.drop(['TotRmsAbvGrd','GarageArea','TotalBsmtSF','Exterior2nd'],axis = 1, inplace
```

In [91]:
```
# Converting years column to age column
df_test['Year_SinceBuilt'] = df_test['YearBuilt'].max() - df_test['YearBuilt']
df_test['Year_SinceRemodAdded'] = df_test['YearRemodAdd'].max() - df_test['YearRemodAdd'
df_test['Year_Since'] = df_test['YrSold'].max() - df_test['YrSold']
df_test['GarageAge'] = df_test['GarageYrBlt'].max() - df_test['GarageYrBlt']

# Dropping old columns in train dataset
df_test.drop(['YearBuilt','YearRemodAdd','YrSold','GarageYrBlt'], axis=1, inplace = True

df_test.rename(columns= {'Year_Since' : 'Year_Since_Sold'}, inplace = True)
```

In [92]:
```
scaler= StandardScaler()
test_scale = scaler.fit_transform(df_test)
```

In [93]:
```
rfr_l=joblib.load('House_rfr.obj')
result=rfr_l.predict(test_scale)
```

In [95]:
```
a = []
for i in result:
    a.append(i)

test = pd.DataFrame({'TEST' : a})
test.head(50)
```

Out[95]:

	TEST
0	340827.000000
1	201178.171429
2	254390.157143
3	169962.642857
4	232818.200000
5	88799.914286
6	145027.600000
7	333986.571429
8	249452.342857
9	174377.457143

10	91075.357143
11	148065.828571
12	117613.171429
13	173355.342857
14	319580.800000
15	117492.842857
16	121247.614286
17	128101.285714
18	174252.142857
19	197245.000000
20	164709.357143
21	154522.857143
22	156738.571429
23	118733.942857
24	110918.800000
25	129848.000000
26	180258.571429
27	141585.714286
28	174200.400000
29	110169.971429
30	146726.942857
31	199810.071429
32	227653.885714
33	156612.857143
34	118010.000000
35	184081.328571
36	202249.871429
37	124710.714286
38	164237.428571
39	147087.485714
40	106301.757143
41	301790.828571
42	207468.457143
43	195457.900000
44	145391.114286
45	126419.471429
46	125204.685714

47	108230.714286
48	214357.414286
49	340613.971429

```
In [ ]:
```