PRCP-1020-House Price Prediction using Advanced Regression

#Problem Statement Task 1:- Prepare a complete data analysis report on the given data. Task 2:-a) Create a robust machine learning algorithm to accurately predict the price of the house given the various factors across the market. b) Determine the relationship between the house features and how the price varies based on this. Task3:- Come up with suggestions for the customer to buy the house according to the area, price and other requirements.

Importing important libraires

In [1]:
```
# importing libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
#%matplotlibinline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
```

Read data set -

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

In [3]:
```
df
```

Out[3]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Ut
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	/
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	/
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	/
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	/
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	/
...	...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	/
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	/
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	/
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	/
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	/

1460 rows × 81 columns

In [6]:
```
df.shape # number of columns are 1460 and columns are 81
```

Out[6]:
```
(1460, 81)
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [7]:
```
df.head() # display first 5 rows
```

Out[7]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub

5 rows × 81 columns

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             1460 non-null   int64  
 1   MSSubClass     1460 non-null   int64  
 2   MSZoning       1460 non-null   object 
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64  
 5   Street         1460 non-null   object 
 6   Alley          91 non-null     object 
 7   LotShape       1460 non-null   object 
 8   LandContour    1460 non-null   object 
 9   Utilities      1460 non-null   object 
 10  LotConfig      1460 non-null   object 
 11  LandSlope      1460 non-null   object 
 12  Neighborhood   1460 non-null   object 
 13  Condition1     1460 non-null   object 
 14  Condition2     1460 non-null   object 
 15  BldgType       1460 non-null   object 
 16  HouseStyle     1460 non-null   object 
 17  OverallQual    1460 non-null   int64  
 18  OverallCond    1460 non-null   int64  
 19  YearBuilt      1460 non-null   int64  
 20  YearRemodAdd   1460 non-null   int64  
 21  RoofStyle      1460 non-null   object 
 22  RoofMatl       1460 non-null   object 
 23  Exterior1st    1460 non-null   object 
 24  Exterior2nd    1460 non-null   object 
 25  MasVnrType     1452 non-null   object 
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object 
 28  ExterCond      1460 non-null   object 
 29  Foundation     1460 non-null   object 
 30  BsmtQual       1423 non-null   object 
 31  BsmtCond       1423 non-null   object 
 32  BsmtExposure   1422 non-null   object 
 33  BsmtFinType1   1423 non-null   object 
 34  BsmtFinSF1     1460 non-null   int64  
 35  BsmtFinType2   1422 non-null   object 
 36  BsmtFinSF2     1460 non-null   int64  
 37  BsmtUnfSF      1460 non-null   int64  
 38  TotalBsmtSF    1460 non-null   int64  
 39  Heating        1460 non-null   object 
 40  HeatingQC      1460 non-null   object 
 41  CentralAir     1460 non-null   object 
 42  Electrical     1459 non-null   object 
 43  1stFlrSF       1460 non-null   int64  
 44  2ndFlrSF       1460 non-null   int64  
 45  LowQualFinSF   1460 non-null   int64  
 46  GrLivArea      1460 non-null   int64  
 47  BsmtFullBath   1460 non-null   int64  
 48  BsmtHalfBath   1460 non-null   int64  
 49  FullBath       1460 non-null   int64  
 50  HalfBath       1460 non-null   int64  
 51  BedroomAbvGr   1460 non-null   int64  
 52  KitchenAbvGr   1460 non-null   int64  
 53  KitchenQual    1460 non-null   object 
 54  TotRmsAbvGrd   1460 non-null   int64  
```

```
 55  Functional     1460 non-null   object
 56  Fireplaces     1460 non-null   int64
 57  FireplaceQu    770 non-null    object
 58  GarageType     1379 non-null   object
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
 69  3SsnPorch      1460 non-null   int64
 70  ScreenPorch    1460 non-null   int64
 71  PoolArea       1460 non-null   int64
 72  PoolQC         7 non-null      object
 73  Fence          281 non-null    object
 74  MiscFeature    54 non-null     object
 75  MiscVal        1460 non-null   int64
 76  MoSold         1460 non-null   int64
 77  YrSold         1460 non-null   int64
 78  SaleType       1460 non-null   object
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

In [9]:

```
# To show the all columns
pd.set_option("display.max_columns", 2000)
pd.set_option("display.max_rows", 85)
```

In [10]:

```
df
```

Out[10]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Ut
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	/
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	/
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	/
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	/
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	/
...	...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	/
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	/
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	/
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	/
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	/

1460 rows × 81 columns

In [9]:
```
df.head(6)
```

Out[9]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub
5	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub

In [10]:
```
df.shape
```

Out[10]:
```
(1460, 81)
```

In [11]:
```
df.describe() # shows numerical features
```

Out[11]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.000000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.267808
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.202904
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.000000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.000000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.000000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000

In [12]:
```
df.describe(include='O')# categorical columns
```

Out[12]:

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborh
count	1460	1460	91	1460	1460	1460	1460	1460	1
unique	5	2	2	4	4	2	5	3	
top	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	Gtl	NA
freq	1151	1454	50	925	1311	1459	1052	1382	

In [13]:
```
df.dtypes
```

Out[13]:
```
Id                 int64
MSSubClass         int64
MSZoning          object
LotFrontage      float64
LotArea            int64
Street            object
Alley             object
LotShape          object
LandContour       object
Utilities         object
LotConfig         object
LandSlope         object
Neighborhood      object
Condition1        object
Condition2        object
BldgType          object
HouseStyle        object
OverallQual        int64
OverallCond        int64
YearBuilt          int64
YearRemodAdd       int64
RoofStyle         object
RoofMatl          object
Exterior1st       object
Exterior2nd       object
MasVnrType        object
MasVnrArea       float64
ExterQual         object
ExterCond         object
Foundation        object
BsmtQual          object
BsmtCond          object
BsmtExposure      object
BsmtFinType1      object
BsmtFinSF1         int64
BsmtFinType2      object
BsmtFinSF2         int64
BsmtUnfSF          int64
TotalBsmtSF        int64
Heating           object
HeatingQC         object
CentralAir        object
Electrical        object
1stFlrSF           int64
2ndFlrSF           int64
LowQualFinSF       int64
GrLivArea          int64
BsmtFullBath       int64
BsmtHalfBath       int64
FullBath           int64
HalfBath           int64
BedroomAbvGr       int64
KitchenAbvGr       int64
KitchenQual       object
TotRmsAbvGrd       int64
Functional        object
Fireplaces         int64
FireplaceQu       object
GarageType        object
GarageYrBlt      float64
```

```
GarageFinish      object
GarageCars         int64
GarageArea         int64
GarageQual        object
GarageCond        object
PavedDrive        object
WoodDeckSF         int64
OpenPorchSF        int64
EnclosedPorch      int64
3SsnPorch          int64
ScreenPorch        int64
PoolArea           int64
PoolQC            object
Fence             object
MiscFeature       object
MiscVal            int64
MoSold             int64
YrSold             int64
SaleType          object
SaleCondition     object
SalePrice          int64
dtype: object
```

#df.describe(include='O').value_counts

In [15]:
```
# Set index as Id
df = df.set_index("Id")
```

In [16]:
```
df
```

Out[16]:

Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities
1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub
2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub
3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub
4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub
5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub
...	...	...	...	...	...	...	...	...	...
1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub
1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	AllPub
1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	AllPub
1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	AllPub
1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	AllPub

1460 rows × 80 columns

Id column set as index column so no nned to remobve it # not necessary step because we set index as id column in df # drop unimportant column ID : df.drop(df['Id'], axis = 1, inplace = True)

In [17]:
```
# drop duplicates
df = df.drop_duplicates()
print( df.shape )
# no duplicates in given datasets
```

```
(1460, 80)
```

In [18]:
```
# find unique value
df.nunique()
```

Out[18]:
```
MSSubClass         15
MSZoning            5
LotFrontage       110
LotArea          1073
Street              2
Alley               2
LotShape            4
LandContour         4
Utilities           2
LotConfig           5
LandSlope           3
Neighborhood       25
Condition1          9
Condition2          8
BldgType            5
HouseStyle          8
OverallQual        10
OverallCond         9
YearBuilt         112
YearRemodAdd       61
RoofStyle           6
RoofMatl            8
Exterior1st        15
Exterior2nd        16
MasVnrType          4
MasVnrArea        327
ExterQual           4
ExterCond           5
Foundation          6
BsmtQual            4
BsmtCond            4
BsmtExposure        4
BsmtFinType1        6
BsmtFinSF1        637
BsmtFinType2        6
BsmtFinSF2        144
BsmtUnfSF         780
TotalBsmtSF       721
Heating             6
HeatingQC           5
CentralAir          2
Electrical          5
1stFlrSF          753
2ndFlrSF          417
LowQualFinSF       24
GrLivArea         861
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       441
GarageQual         5
GarageCond         5
PavedDrive         3
WoodDeckSF       274
OpenPorchSF      202
EnclosedPorch    120
3SsnPorch         20
ScreenPorch       76
PoolArea           8
PoolQC             3
Fence              4
MiscFeature        4
MiscVal           21
MoSold            12
YrSold             5
SaleType           9
SaleCondition      6
SalePrice        663
dtype: int64
```

2. EXPLORATORY DATA ANALYSIS -with data analysis

-Using EDA :Visualize fesatures, insight /observation from the data

- Missing Values
- All The Numerical Variables
- Distribution of the Numerical Variables
- Categorical Variables
- Cardinality of Categorical Variables
- Outliers

use automated pandas profiling for EDA - Pandas profiling - goal is to a EDA like df.desribe() function

In [23]:
```
from pandas_profiling import ProfileReport  # file import
```

In [40]:
```
!pip install pandas-profiling
```

```
Collecting pandas-profiling
  Downloading pandas_profiling-3.6.6-py2.py3-none-any.whl (324 kB)
     ------------------------------------ 324.4/324.4 kB 872.6 kB/s eta 0:00:00
Collecting ydata-profiling
  Downloading ydata_profiling-4.3.1-py2.py3-none-any.whl (352 kB)
     -------------------------------------- 353.0/353.0 kB 2.4 MB/s eta 0:00:00
Requirement already satisfied: numpy<1.24,>=1.16.0 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (1.21.5)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (2.11.3)
Collecting wordcloud>=1.9.1
  Downloading wordcloud-1.9.2-cp39-cp39-win_amd64.whl (153 kB)
     -------------------------------------- 153.3/153.3 kB 1.3 MB/s eta 0:00:00
Collecting visions[type_image_path]==0.7.5
  Downloading visions-0.7.5-py3-none-any.whl (102 kB)
     ------------------------------------ 102.7/102.7 kB 211.5 kB/s eta 0:00:00
Requirement already satisfied: scipy<1.11,>=1.4.1 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (1.9.1)
Requirement already satisfied: pandas!=1.4.0,<2.1,>1.1 in c:\users\vijay shelke\anaco
nda3\lib\site-packages (from ydata-profiling->pandas-profiling) (1.4.4)
Requirement already satisfied: matplotlib<4,>=3.2 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (3.5.2)
Collecting htmlmin==0.1.12
  Downloading htmlmin-0.1.12.tar.gz (19 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Collecting phik<0.13,>=0.11.1
  Downloading phik-0.12.3-cp39-cp39-win_amd64.whl (663 kB)
     -------------------------------------- 663.5/663.5 kB 3.8 MB/s eta 0:00:00
Requirement already satisfied: tqdm<5,>=4.48.2 in c:\users\vijay shelke\anaconda3\lib
\site-packages (from ydata-profiling->pandas-profiling) (4.64.1)
Requirement already satisfied: seaborn<0.13,>=0.10.1 in c:\users\vijay shelke\anacond
a3\lib\site-packages (from ydata-profiling->pandas-profiling) (0.11.2)
Collecting multimethod<2,>=1.4
  Downloading multimethod-1.9.1-py3-none-any.whl (10 kB)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (6.0)
Collecting pydantic<2,>=1.8.1
  Downloading pydantic-1.10.11-cp39-cp39-win_amd64.whl (2.2 MB)
     ---------------------------------------- 2.2/2.2 MB 4.5 MB/s eta 0:00:00
Collecting typeguard<3,>=2.13.2
  Downloading typeguard-2.13.3-py3-none-any.whl (17 kB)
Collecting imagehash==4.3.1
  Downloading ImageHash-4.3.1-py2.py3-none-any.whl (296 kB)
     ------------------------------------ 296.5/296.5 kB 171.2 kB/s eta 0:00:00
Requirement already satisfied: requests<3,>=2.24.0 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from ydata-profiling->pandas-profiling) (2.28.1)
Collecting dacite>=1.8
  Downloading dacite-1.8.1-py3-none-any.whl (14 kB)
Requirement already satisfied: statsmodels<1,>=0.13.2 in c:\users\vijay shelke\anacon
da3\lib\site-packages (from ydata-profiling->pandas-profiling) (0.13.2)
Requirement already satisfied: pillow in c:\users\vijay shelke\anaconda3\lib\site-pac
kages (from imagehash==4.3.1->ydata-profiling->pandas-profiling) (9.2.0)
Requirement already satisfied: PyWavelets in c:\users\vijay shelke\anaconda3\lib\site
-packages (from imagehash==4.3.1->ydata-profiling->pandas-profiling) (1.3.0)
Collecting tangled-up-in-unicode>=0.0.4
  Downloading tangled_up_in_unicode-0.2.0-py3-none-any.whl (4.7 MB)
     ---------------------------------------- 4.7/4.7 MB 5.7 MB/s eta 0:00:00
Requirement already satisfied: attrs>=19.3.0 in c:\users\vijay shelke\anaconda3\lib\s
ite-packages (from visions[type_image_path]==0.7.5->ydata-profiling->pandas-profilin
```

```
g) (21.4.0)
Requirement already satisfied: networkx>=2.4 in c:\users\vijay shelke\anaconda3\lib\s
ite-packages (from visions[type_image_path]==0.7.5->ydata-profiling->pandas-profilin
g) (2.8.4)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\vijay shelke\anaconda3\li
b\site-packages (from jinja2<3.2,>=2.11.1->ydata-profiling->pandas-profiling) (2.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\vijay shelke\anaconda3\l
ib\site-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (1.4.2)
Requirement already satisfied: packaging>=20.0 in c:\users\vijay shelke\anaconda3\lib
\site-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (21.3)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\vijay shelke\anaconda3\l
ib\site-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (4.25.
0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\vijay shelke\anaconda
3\lib\site-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (2.
8.2)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\vijay shelke\anaconda3\li
b\site-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (3.0.9)
Requirement already satisfied: cycler>=0.10 in c:\users\vijay shelke\anaconda3\lib\si
te-packages (from matplotlib<4,>=3.2->ydata-profiling->pandas-profiling) (0.11.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\vijay shelke\anaconda3\lib\si
te-packages (from pandas!=1.4.0,<2.1,>1.1->ydata-profiling->pandas-profiling) (2022.
1)
Requirement already satisfied: joblib>=0.14.1 in c:\users\vijay shelke\anaconda3\lib
\site-packages (from phik<0.13,>=0.11.1->ydata-profiling->pandas-profiling) (1.2.0)
Requirement already satisfied: typing-extensions>=4.2.0 in c:\users\vijay shelke\anac
onda3\lib\site-packages (from pydantic<2,>=1.8.1->ydata-profiling->pandas-profiling)
(4.3.0)
Requirement already satisfied: idna<4,>=2.5 in c:\users\vijay shelke\anaconda3\lib\si
te-packages (from requests<3,>=2.24.0->ydata-profiling->pandas-profiling) (3.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\vijay shelke\anaconda3
\lib\site-packages (from requests<3,>=2.24.0->ydata-profiling->pandas-profiling) (202
2.9.14)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\vijay shelke\anac
onda3\lib\site-packages (from requests<3,>=2.24.0->ydata-profiling->pandas-profiling)
(2.0.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\vijay shelke\anacond
a3\lib\site-packages (from requests<3,>=2.24.0->ydata-profiling->pandas-profiling)
(1.26.11)
Requirement already satisfied: patsy>=0.5.2 in c:\users\vijay shelke\anaconda3\lib\si
te-packages (from statsmodels<1,>=0.13.2->ydata-profiling->pandas-profiling) (0.5.2)
Requirement already satisfied: colorama in c:\users\vijay shelke\anaconda3\lib\site-p
ackages (from tqdm<5,>=4.48.2->ydata-profiling->pandas-profiling) (0.4.5)
Requirement already satisfied: six in c:\users\vijay shelke\anaconda3\lib\site-packag
es (from patsy>=0.5.2->statsmodels<1,>=0.13.2->ydata-profiling->pandas-profiling) (1.
16.0)
Building wheels for collected packages: htmlmin
  Building wheel for htmlmin (setup.py): started
  Building wheel for htmlmin (setup.py): finished with status 'done'
  Created wheel for htmlmin: filename=htmlmin-0.1.12-py3-none-any.whl size=27082 sha2
56=9c3ba0be9f45119adaf9bbc0087789278be164ab100e2f538d6a77647431d3e5
  Stored in directory: c:\users\vijay shelke\appdata\local\pip\cache\wheels\1d\05\04
\c6d7d3b66539d9e659ac6dfe81e2d0fd4c1a8316cc5a403300
Successfully built htmlmin
Installing collected packages: htmlmin, typeguard, tangled-up-in-unicode, pydantic, m
ultimethod, dacite, imagehash, wordcloud, visions, phik, ydata-profiling, pandas-prof
iling
Successfully installed dacite-1.8.1 htmlmin-0.1.12 imagehash-4.3.1 multimethod-1.9.1
pandas-profiling-3.6.6 phik-0.12.3 pydantic-1.10.11 tangled-up-in-unicode-0.2.0 typeg
uard-2.13.3 visions-0.7.5 wordcloud-1.9.2 ydata-profiling-4.3.1
```

Generate the EDA report profile=ProfileReport(df,explorative=True) # EXPORATIVE TRUE- GIVES ALL INFORMATION FROM YOUR DATASET profile.to_file('house.html')

In [20]:
```
#know the data types of each columns
obj = (df.dtypes == 'object')
object_cols = list(obj[obj].index)
print("Categorical variables:",len(object_cols))
```

```
Categorical variables: 43
```

In [60]:
```
# most features are categorical varaibles
```

In [21]:
```
df.select_dtypes(include=['int64', 'float64']).columns
```

Out[21]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

In [23]:
```
# list of all objects column categorical
df.select_dtypes(include=['object']).columns
```

Out[23]:
```
Index(['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities',
       'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2',
       'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st',
       'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation',
       'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
       'Heating', 'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual',
       'Functional', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageQual',
       'GarageCond', 'PavedDrive', 'PoolQC', 'Fence', 'MiscFeature',
       'SaleType', 'SaleCondition'],
      dtype='object')
```

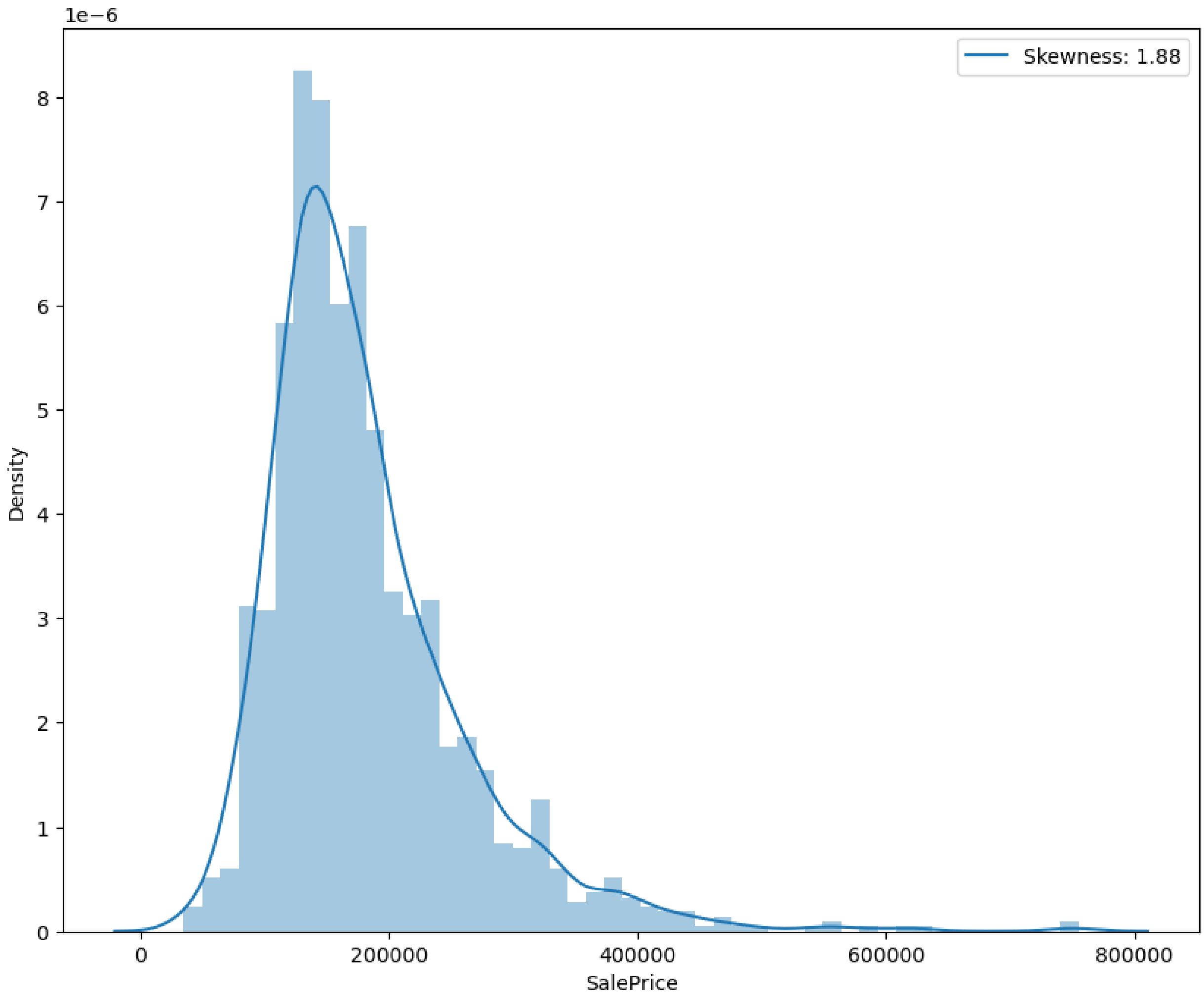
check the distribution of target column - do log transformation due to right positive skewness:

In [71]:
```
# check the distrubution of target varible i.e SalePrcie
df["SalePrice"].describe()
```

Out[71]:
```
count      1460.000000
mean     180921.195890
std       79442.502883
min       34900.000000
25%      129975.000000
50%      163000.000000
75%      214000.000000
max      755000.000000
Name: SalePrice, dtype: float64
```

In [24]:
```
# Plot the distplot of target
plt.figure(figsize=(10,8))
bar = sns.distplot(df["SalePrice"],kde=True)
bar.legend(["Skewness: {:.2f}".format(df['SalePrice'].skew())])
```

Out[24]: <matplotlib.legend.Legend at 0x2cd1631880>



here sales price normaly distrubeted with right skew

log transformation on sale price -right skewness

In [31]:
```
# this indicated output variabvle Sale price is right skewed so i applied log trabnsof
```

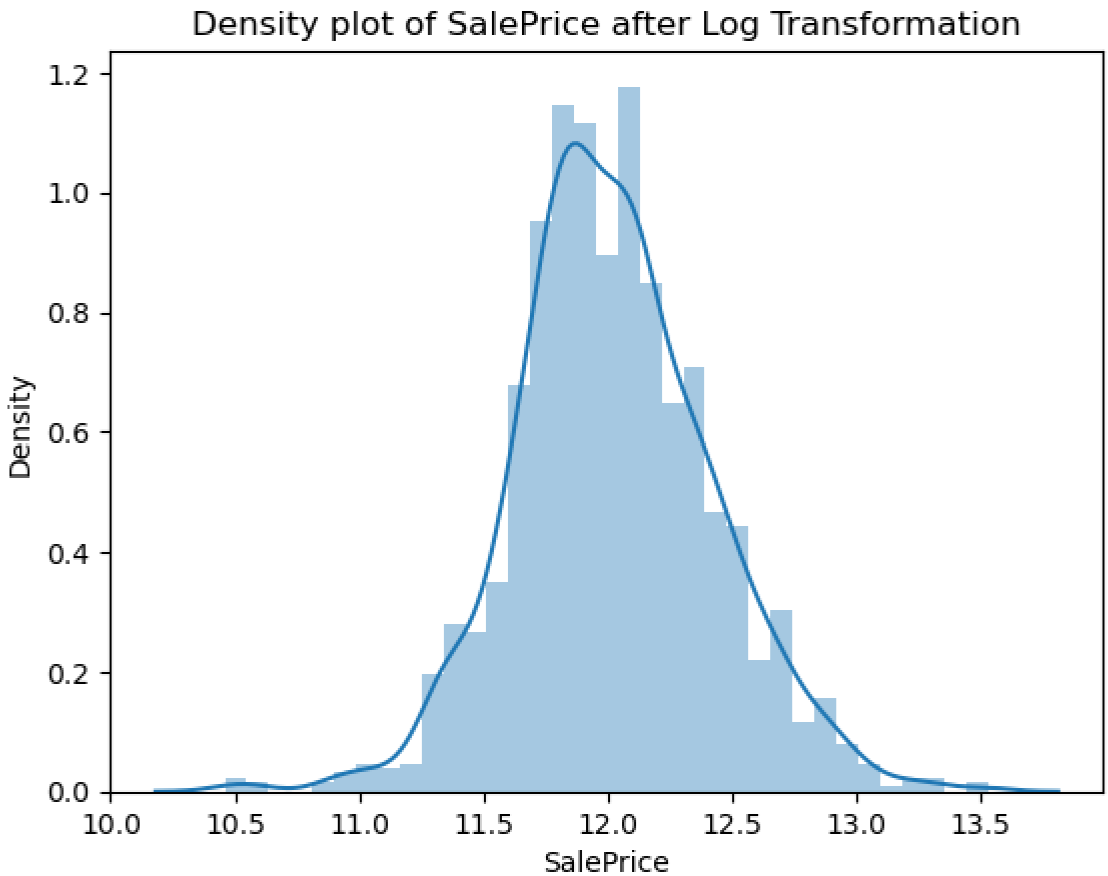
In [25]:
```
# Positive Skeweness:
df['SalePrice'].skew()
```

Out[25]: 1.8828757597682129

In [26]:
```
df["SalePrice"] = np.log1p(df["SalePrice"])
```

In [28]:
```
 #SalePrice after Log-transformation
sns.distplot(df["SalePrice"])
plt.title("Density plot of SalePrice after Log Transformation")
```

Out[28]: Text(0.5, 1.0, 'Density plot of SalePrice after Log Transformation')



In [29]:
```
# again check skewness :
# Positive Skeweness:
df['SalePrice'].skew()
```

Out[29]:
```
0.12134661989685333
```

3. Datapreprocessing

- handling missing values
- handling outliers
- drop duplicates
- scaling

In [31]:
```
#sum of missing data
df.isnull().sum().sort_values(ascending=False)
```

Out[31]:
```
PoolQC           1453
MiscFeature      1406
Alley            1369
Fence            1179
FireplaceQu       690
LotFrontage       259
GarageYrBlt        81
GarageCond         81
GarageType         81
GarageFinish       81
GarageQual         81
BsmtExposure       38
BsmtFinType2       38
BsmtCond           37
BsmtQual           37
BsmtFinType1       37
MasVnrArea          8
MasVnrType          8
Electrical          1
MSSubClass          0
Fireplaces          0
Functional          0
KitchenQual         0
KitchenAbvGr        0
BedroomAbvGr        0
HalfBath            0
FullBath            0
BsmtHalfBath        0
TotRmsAbvGrd        0
GarageCars          0
GrLivArea           0
GarageArea          0
PavedDrive          0
WoodDeckSF          0
OpenPorchSF         0
EnclosedPorch       0
3SsnPorch           0
ScreenPorch         0
PoolArea            0
MiscVal             0
MoSold              0
YrSold              0
SaleType            0
SaleCondition       0
BsmtFullBath        0
CentralAir          0
LowQualFinSF        0
Neighborhood        0
OverallCond         0
OverallQual         0
HouseStyle          0
BldgType            0
Condition2          0
Condition1          0
LandSlope           0
2ndFlrSF            0
LotConfig           0
Utilities           0
LandContour         0
LotShape            0
```

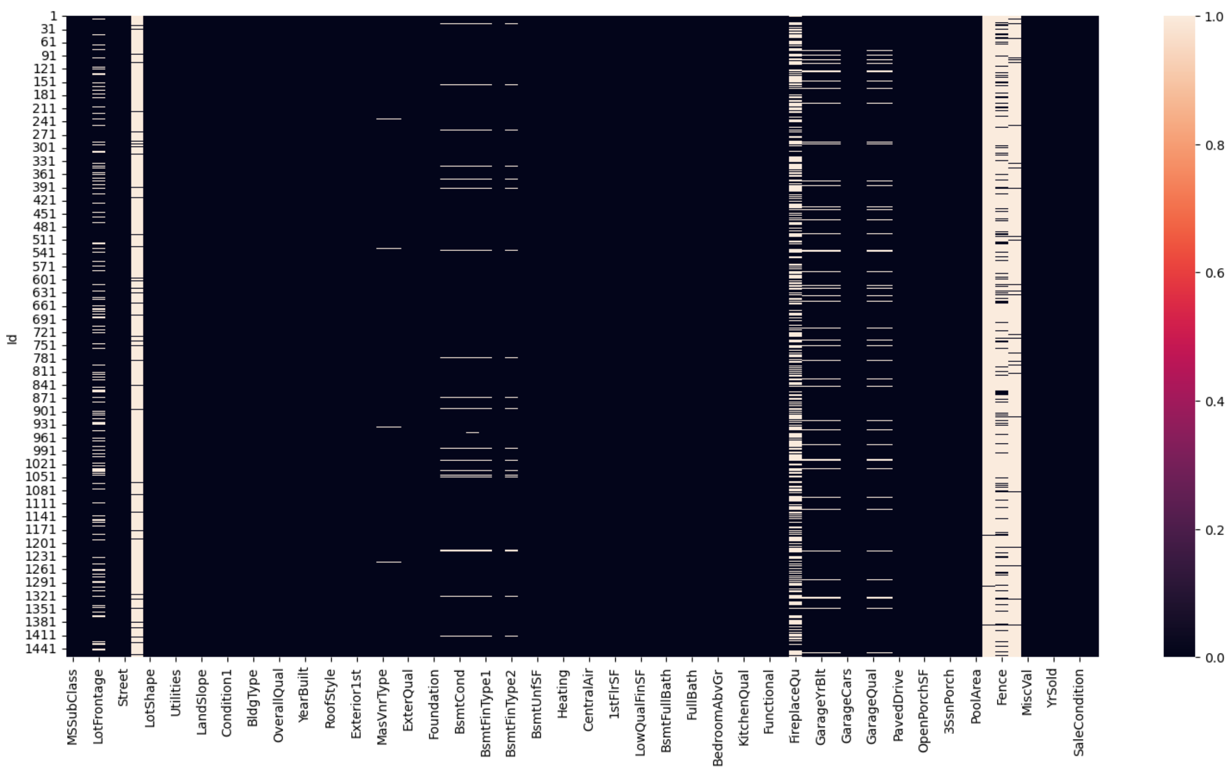
```
Street            0
LotArea           0
YearBuilt         0
YearRemodAdd      0
RoofStyle         0
RoofMatl          0
Exterior1st       0
Exterior2nd       0
ExterQual         0
ExterCond         0
Foundation        0
BsmtFinSF1        0
BsmtFinSF2        0
BsmtUnfSF         0
TotalBsmtSF       0
Heating           0
HeatingQC         0
MSZoning          0
1stFlrSF          0
SalePrice         0
dtype: int64
```

In [79]:
```
# Get the percentages of null values of each column
null_percent = df.isnull().sum()/df.shape[0]*100
null_percent
```

Out[79]:
```
MSSubClass        0.000000
MSZoning          0.000000
LotFrontage      17.739726
LotArea           0.000000
Street            0.000000
Alley            93.767123
LotShape          0.000000
LandContour       0.000000
Utilities         0.000000
LotConfig         0.000000
LandSlope         0.000000
Neighborhood      0.000000
Condition1        0.000000
Condition2        0.000000
BldgType          0.000000
HouseStyle        0.000000
OverallQual       0.000000
OverallCond       0.000000
YearBuilt         0.000000
YearRemodAdd      0.000000
RoofStyle         0.000000
RoofMatl          0.000000
Exterior1st       0.000000
Exterior2nd       0.000000
MasVnrType        0.547945
MasVnrArea        0.547945
ExterQual         0.000000
ExterCond         0.000000
Foundation        0.000000
BsmtQual          2.534247
BsmtCond          2.534247
BsmtExposure      2.602740
BsmtFinType1      2.534247
BsmtFinSF1        0.000000
BsmtFinType2      2.602740
BsmtFinSF2        0.000000
BsmtUnfSF         0.000000
TotalBsmtSF       0.000000
Heating           0.000000
HeatingQC         0.000000
CentralAir        0.000000
Electrical        0.068493
1stFlrSF          0.000000
2ndFlrSF          0.000000
LowQualFinSF      0.000000
GrLivArea         0.000000
BsmtFullBath      0.000000
BsmtHalfBath      0.000000
FullBath          0.000000
HalfBath          0.000000
BedroomAbvGr      0.000000
KitchenAbvGr      0.000000
KitchenQual       0.000000
TotRmsAbvGrd      0.000000
Functional        0.000000
Fireplaces        0.000000
FireplaceQu      47.260274
GarageType        5.547945
GarageYrBlt       5.547945
GarageFinish      5.547945
```

```
GarageCars        0.000000
GarageArea        0.000000
GarageQual        5.547945
GarageCond        5.547945
PavedDrive        0.000000
WoodDeckSF        0.000000
OpenPorchSF       0.000000
EnclosedPorch     0.000000
3SsnPorch         0.000000
ScreenPorch       0.000000
PoolArea          0.000000
PoolQC           99.520548
Fence            80.753425
MiscFeature      96.301370
MiscVal           0.000000
MoSold            0.000000
YrSold            0.000000
SaleType          0.000000
SaleCondition     0.000000
SalePrice         0.000000
dtype: float64
```

In [32]:
```
# Show the null values using heatmap
plt.figure(figsize=(18,9))
sns.heatmap(df.isnull())
plt.show()
```



In [33]:
```
# We have to drop some columns which contains large number of null values and features
drop_variables = ['Alley', 'FireplaceQu', 'PoolQC', 'Fence', 'MiscFeature']
df =df.drop(drop_variables, axis = 1)
df
```

Out[33]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotCo
Id									
1	60	RL	65.0	8450	Pave	Reg	Lvl	AllPub	In
2	20	RL	80.0	9600	Pave	Reg	Lvl	AllPub	
3	60	RL	68.0	11250	Pave	IR1	Lvl	AllPub	In
4	70	RL	60.0	9550	Pave	IR1	Lvl	AllPub	Co
5	60	RL	84.0	14260	Pave	IR1	Lvl	AllPub	
...	...	...	...	...	...	...	...	...	
1456	60	RL	62.0	7917	Pave	Reg	Lvl	AllPub	In
1457	20	RL	85.0	13175	Pave	Reg	Lvl	AllPub	In
1458	70	RL	66.0	9042	Pave	Reg	Lvl	AllPub	In
1459	20	RL	68.0	9717	Pave	Reg	Lvl	AllPub	In
1460	20	RL	75.0	9937	Pave	Reg	Lvl	AllPub	In

1460 rows × 75 columns

5 columns get droped # most null values columns are - ALLEY ,PoolQC ,MiscFeature, Fence

In [34]:

```
# some columns aremissing values less ampount of missing values
# after some delete missing values columns some columns have missing values
col_nan = df.isna().sum() / df.shape[0]
```

In [35]:

```
col_nan.sort_values(ascending=False)
```

Out[35]:

```
LotFrontage      0.177397
GarageType       0.055479
GarageYrBlt      0.055479
GarageFinish     0.055479
GarageQual       0.055479
GarageCond       0.055479
BsmtFinType2     0.026027
BsmtExposure     0.026027
BsmtQual         0.025342
BsmtCond         0.025342
BsmtFinType1     0.025342
MasVnrArea       0.005479
MasVnrType       0.005479
Electrical       0.000685
KitchenAbvGr     0.000000
BedroomAbvGr     0.000000
HalfBath         0.000000
FullBath         0.000000
BsmtHalfBath     0.000000
BsmtFullBath     0.000000
KitchenQual      0.000000
GrLivArea        0.000000
TotRmsAbvGrd     0.000000
Functional       0.000000
MSSubClass       0.000000
Fireplaces       0.000000
ScreenPorch      0.000000
SaleCondition    0.000000
SaleType         0.000000
YrSold           0.000000
MoSold           0.000000
MiscVal          0.000000
PoolArea         0.000000
3SsnPorch        0.000000
2ndFlrSF         0.000000
EnclosedPorch    0.000000
OpenPorchSF      0.000000
WoodDeckSF       0.000000
PavedDrive       0.000000
GarageArea       0.000000
GarageCars       0.000000
LowQualFinSF     0.000000
Heating          0.000000
1stFlrSF         0.000000
CentralAir       0.000000
LotArea          0.000000
Street           0.000000
LotShape         0.000000
LandContour      0.000000
Utilities        0.000000
LotConfig        0.000000
LandSlope        0.000000
Neighborhood     0.000000
Condition1       0.000000
Condition2       0.000000
BldgType         0.000000
HouseStyle       0.000000
OverallQual      0.000000
OverallCond      0.000000
YearBuilt        0.000000
```

```
YearRemodAdd     0.000000
RoofStyle        0.000000
RoofMatl         0.000000
Exterior1st      0.000000
Exterior2nd      0.000000
ExterQual        0.000000
ExterCond        0.000000
Foundation       0.000000
BsmtFinSF1       0.000000
BsmtFinSF2       0.000000
BsmtUnfSF        0.000000
TotalBsmtSF      0.000000
MSZoning         0.000000
HeatingQC        0.000000
SalePrice        0.000000
dtype: float64
```

these columns have missing values : LotFrontage 0.177397 GarageType 0.055479 GarageYrBlt 0.055479 GarageFinish 0.055479 GarageQual 0.055479 GarageCond 0.055479 BsmtFinType2 0.026027 BsmtExposure 0.026027 BsmtQual 0.025342 BsmtCond 0.025342 BsmtFinType1 0.025342 MasVnrArea 0.005479 MasVnrType 0.005479 Electrical 0.000685

In [36]:
```
df.columns[df.isnull().any()] # these columns having missing values
```

Out[36]:
```
Index(['LotFrontage', 'MasVnrType', 'MasVnrArea', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Electrical',
       'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageQual',
       'GarageCond'],
      dtype='object')
```

#these categorical columns missing values MasVnrType 8 BsmtQual 37 BsmtCond 37 BsmtExposure 38 BsmtFinType1 37 BsmtFinType2 38 GarageType 81 GarageFinish 81 GarageQual 81 GarageCond 81 Electrical 1 # NUMERICAL COLUMNS - MasVnrArea-8 , ' -1 # DATE AND TIME COLUMNS : GarageYrBlt', 81'

In [48]:
```
# filling missing value with median and mode of numerical as wlll as categorical varai
df['LotFrontage'] = df['LotFrontage'].fillna(df['LotFrontage'].mean())
```

cat_col=['BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2', 'MasVnrType','GarageType','GarageFinish','GarageQual', 'GarageCond'] # these are objectS 10 columns has missing vlaues .

In [49]:
```
# MasVnrType-
df['MasVnrType'] = df['MasVnrType'].fillna(df['MasVnrType'].mode()[0])
#MasVnrArea
df['MasVnrArea'] = df['MasVnrArea'].fillna(df['MasVnrArea'].median())
```

In [50]:
```
# electrical cat features
df['Electrical'] = df['Electrical'].fillna(df['Electrical'].mode()[0])
```

In [51]:
```
# basement features :
#BsmtQual     BsmtCond    BsmtExposure  BsmtFinType1  BsmtFinType2   -categorical feat
df['BsmtQual'] = df['BsmtQual'].fillna(df['BsmtQual'].mode()[0])
df['BsmtCond'] = df['BsmtCond'].fillna(df['BsmtCond'].mode()[0])
df['BsmtExposure'] = df['BsmtExposure'].fillna(df['BsmtExposure'].mode()[0])
df['BsmtFinType1'] = df['BsmtFinType1'].fillna(df['BsmtFinType1'].mode()[0])
df['BsmtFinType2'] = df['BsmtFinType2'].fillna(df['BsmtFinType2'].mode()[0])
```

In [52]:
```
#All garage features :are categorical -replace with mode
# GarageType    GarageFinish  GarageQual   GarageCond
print('Fill missing values of Garage features with medain or mode')
df['GarageType'] = df['GarageType'].fillna(df['GarageType'].mode()[0])
```

```
df['GarageFinish'] = df['GarageFinish'].fillna(df['GarageFinish'].mode()[0])
df['GarageCond'] = df['GarageCond'].fillna(df['GarageCond'].mode()[0])
df['GarageQual'] = df['GarageQual'].fillna(df['GarageQual'].mode()[0])
```

Fill missing values of Garage features with medain or mode

In [53]:
```
df['GarageYrBlt'] = df['GarageYrBlt'].fillna(int(0))
```

In [54]:
```
df.isnull().sum() # no null is present
```

Out[54]:
```
MSSubClass      0
MSZoning        0
LotFrontage     0
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      0
MasVnrArea      0
ExterQual       0
ExterCond       0
Foundation      0
BsmtQual        0
BsmtCond        0
BsmtExposure    0
BsmtFinType1    0
BsmtFinSF1      0
BsmtFinType2    0
BsmtFinSF2      0
BsmtUnfSF       0
TotalBsmtSF     0
Heating         0
HeatingQC       0
CentralAir      0
Electrical      0
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
GarageType      0
GarageYrBlt     0
GarageFinish    0
GarageCars      0
GarageArea      0
```

```
GarageQual       0
GarageCond       0
PavedDrive       0
WoodDeckSF       0
OpenPorchSF      0
EnclosedPorch    0
3SsnPorch        0
ScreenPorch      0
PoolArea         0
MiscVal          0
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
SalePrice        0
dtype: int64
```

In [55]:

```
# convert year column into another column and float column into int columns :


# LotFrontage,  MasVnrArea - float 64 type into int 64
df['LotFrontage'] = df['LotFrontage'].astype(np.int64)
df['MasVnrArea'] = df['MasVnrArea'].astype(np.int64)
```

In [56]:

```
# convert sale price into int 64 -SalePrice
df['SalePrice'] = df['SalePrice'].astype(np.int64)
```

In [57]:

```
df['GarageYrBlt']=df['GarageYrBlt'].astype(np.int64)
```

In [58]:

```
df['Utilities'].value_counts()
# these features totaly nosewa having only one value so it is no use for category
df.drop(columns='Utilities',inplace=True)
print('Drop Utilities \n')
```

```
Drop Utilities
```

In [59]:

```
# Convert year related columns to number of years, to find how old the house is, or hc
# house was sold.
# there 4 date time features -YearBuilt', 'YearRemodAdd', 'GarageYrBlt', 'YrSold']
df['YearBuilt'] = 2023 - df['YearBuilt']
df['YearRemodAdd'] = 2023 - df['YearRemodAdd']
df['GarageYrBlt'] = 2023 - df['GarageYrBlt']
df['YrSold'] = 2023 - df['YrSold']
```

In [60]:

```
print(df.isnull().sum().sum()) # no null values are present
```

```
0
```

no null values are present

In [61]:

```
df['Street'].value_counts() # only GRVL VALUE IS 6 PAVE MOST CATEGORY SO WE CAN DROP s
```

Out[61]:

```
Pave    1454
Grvl       6
Name: Street, dtype: int64
```

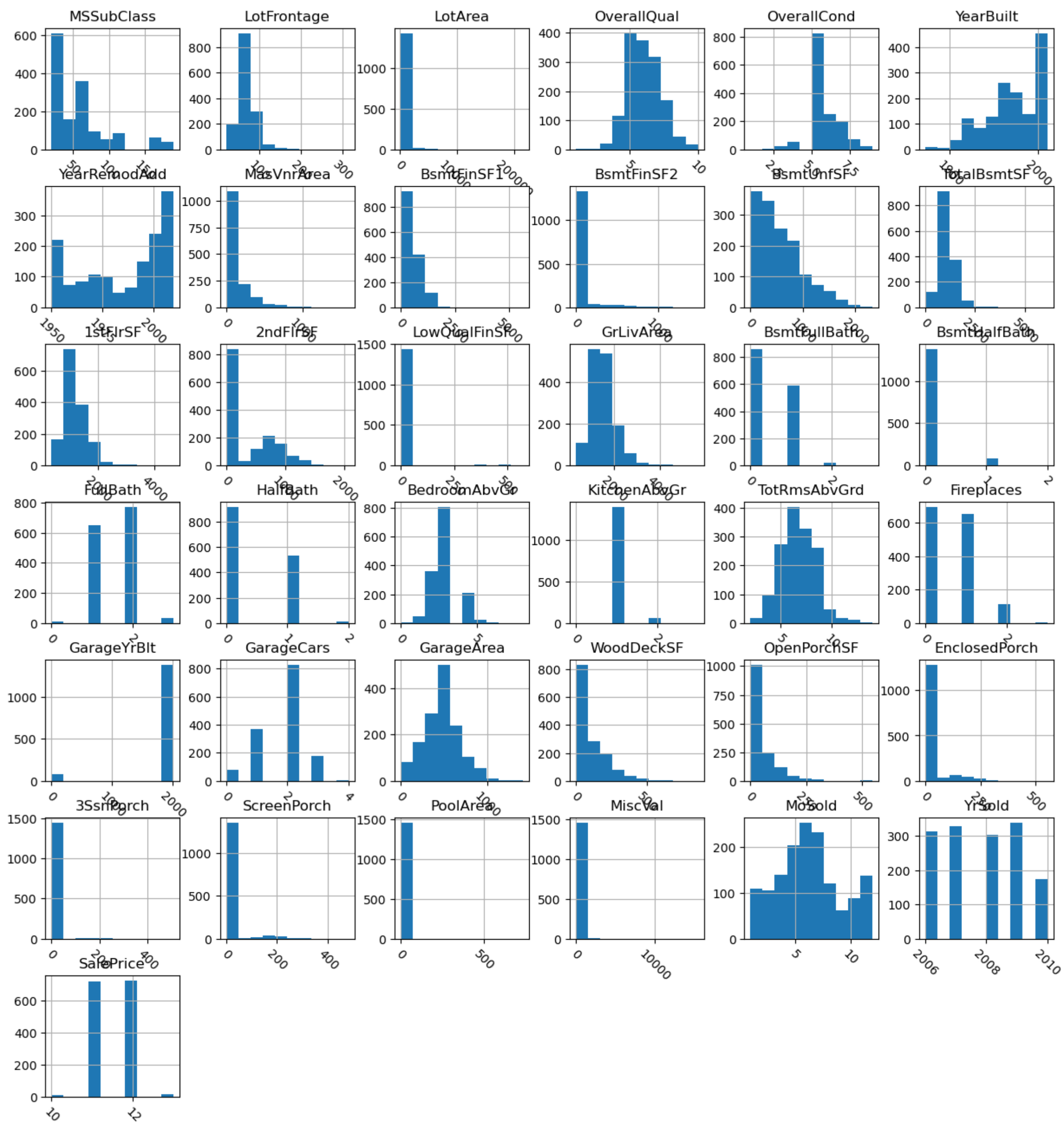
In [ ]:

OUTLIER DETECTION AND REMOVAL : MOST IMP

Removing outliers is important step in data analysis. However, while removing outliers in ML we should be careful, because we do not know if there are not any outliers in test set. I used two techniques: The first one was Z-score method. Z-scores are expressed in terms of standard deviations from their means. As a result, these z-scores have a distribution with a mean of 0 and a standard deviation of 1. I set threshold = 3 to identify outliers.

OUTLIER DETECTION AND REMOVAL :MOST IMP STEPS :

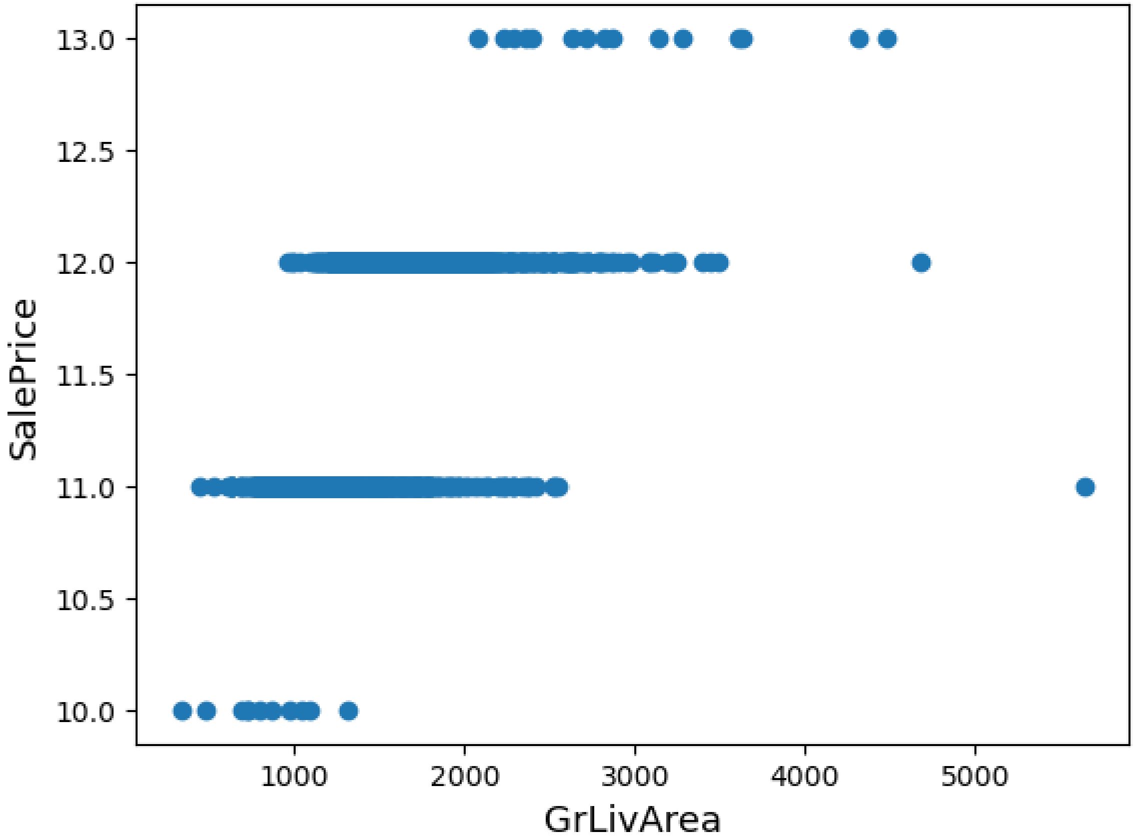
In [62]:
```
# DISTRIBUTIUTION OF NUMERICAL FEATURES
df.hist(figsize=(15,16), xrot=-45, bins=10) ## Display the labels rotated by 45 degres
plt.show()
```

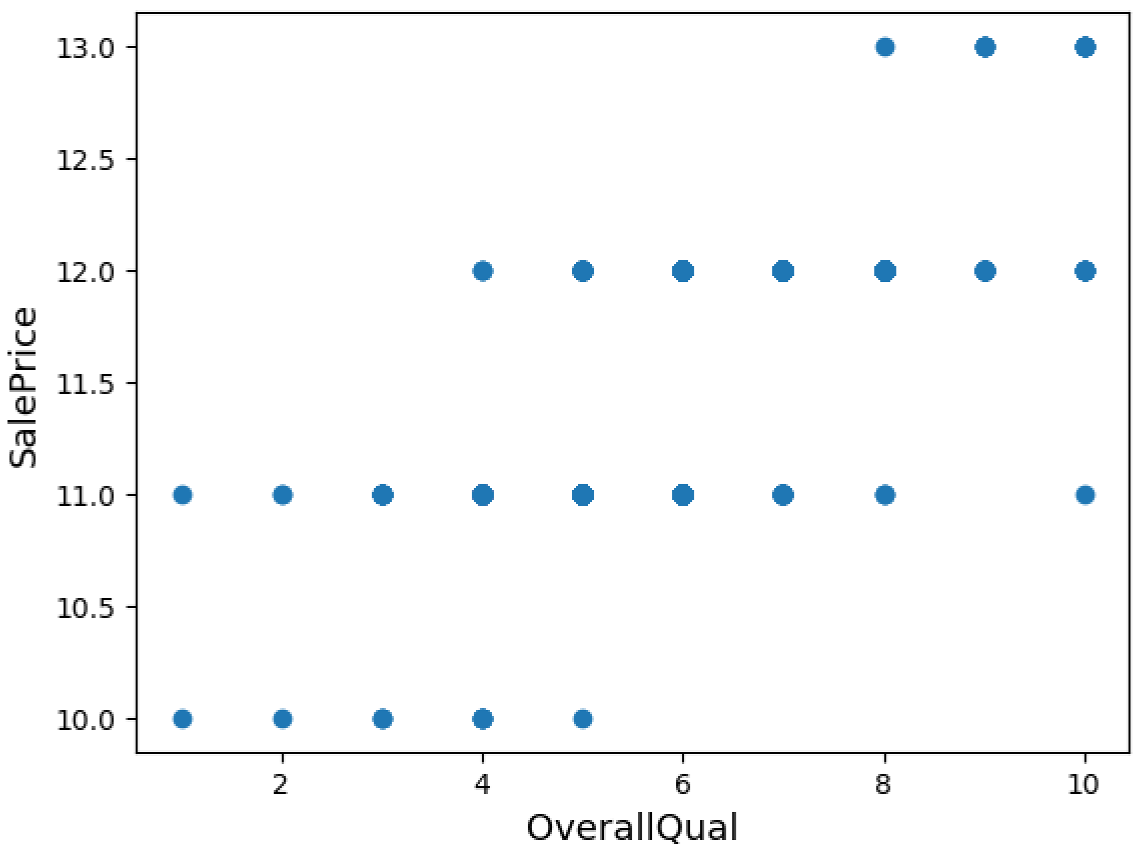


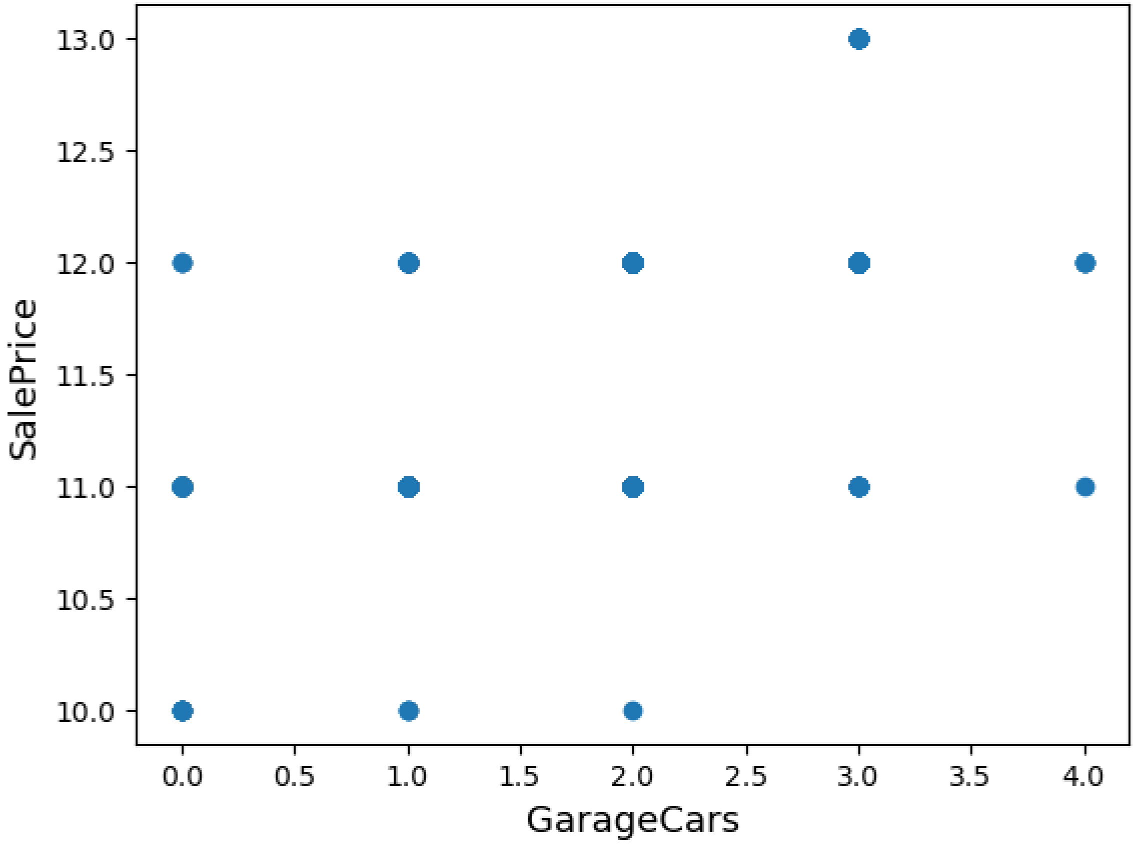
In [126…
```
# MOST COLUMNS ARE RIGHT SKEWED
```

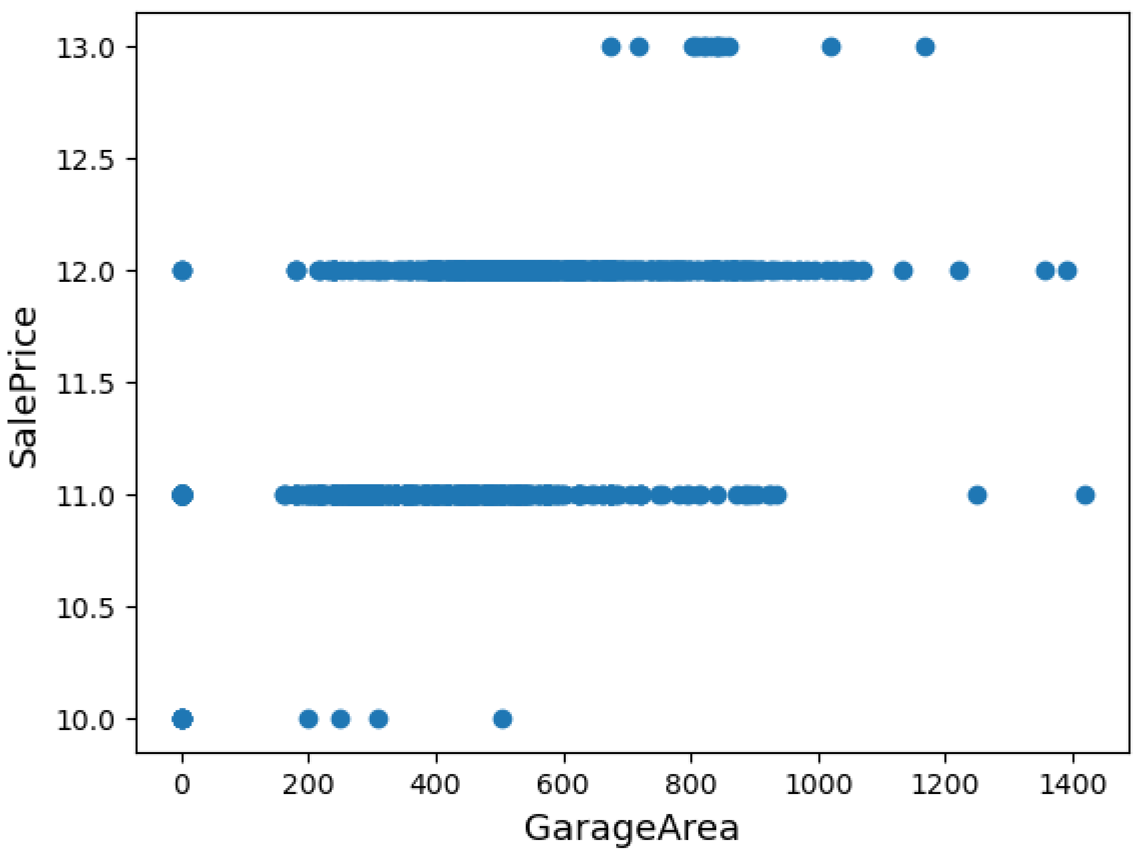
In [69]:
```
# plot outliers of different scatter plot
#now we are going detect outliers in whole dataset
```

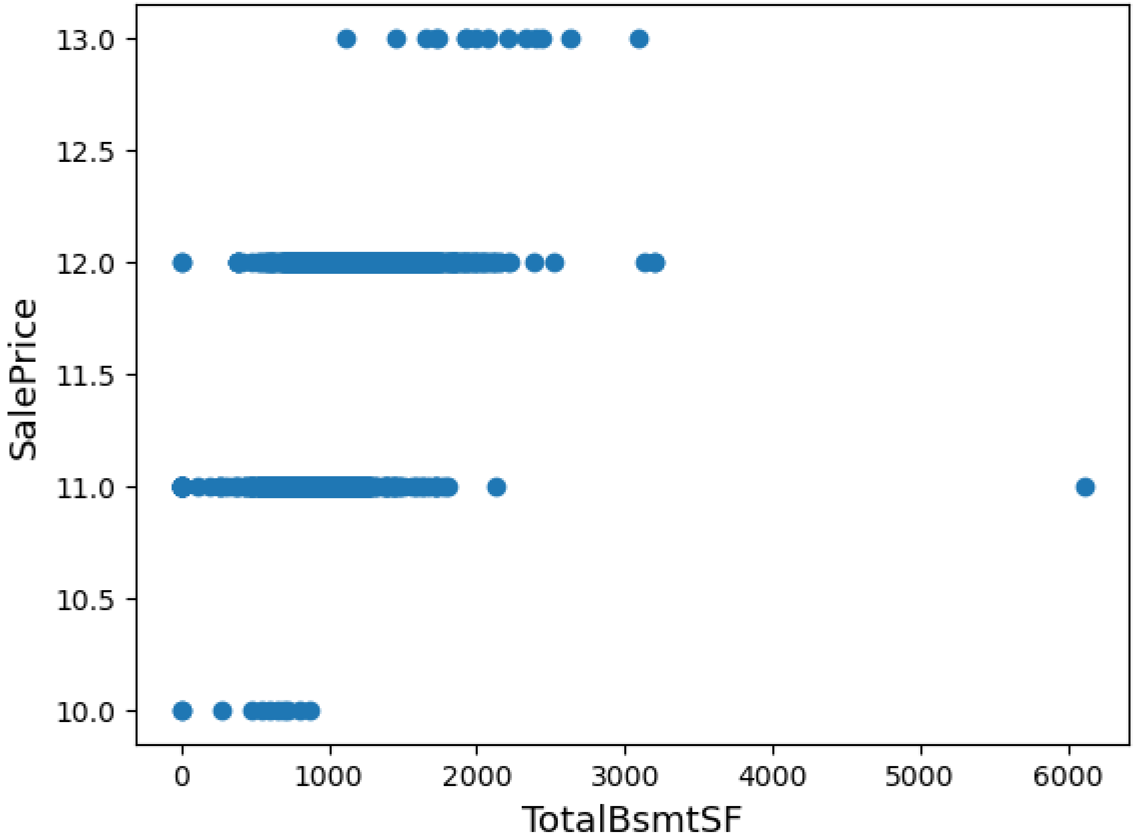
```
fig = plt.subplots()
plt.scatter(x = df['GrLivArea'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('GrLivArea', fontsize=13)
plt.show()

fig1= plt.subplots()
plt.scatter(x = df['OverallQual'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('OverallQual', fontsize=13)
plt.show()

fig2= plt.subplots()
plt.scatter(x = df['GarageCars'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('GarageCars', fontsize=13)
plt.show()

fig3= plt.subplots()
plt.scatter(x = df['GarageArea'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('GarageArea', fontsize=13)
plt.show()

fig4= plt.subplots()
plt.scatter(x = df['TotalBsmtSF'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('TotalBsmtSF', fontsize=13)
plt.show()

fig5= plt.subplots()
plt.scatter(x = df['1stFlrSF'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('1stFlrSF', fontsize=13)
plt.show()

fig6= plt.subplots()
plt.scatter(x = df['FullBath'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('FullBath', fontsize=13)
plt.show()

fig7= plt.subplots()
plt.scatter(x = df['TotRmsAbvGrd'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('TotRmsAbvGrd', fontsize=13)
plt.show()

fig8= plt.subplots()
plt.scatter(x = df['YearBuilt'], y = df['SalePrice'])
plt.ylabel('SalePrice', fontsize=13)
plt.xlabel('YearBuilt', fontsize=13)
plt.show()
```

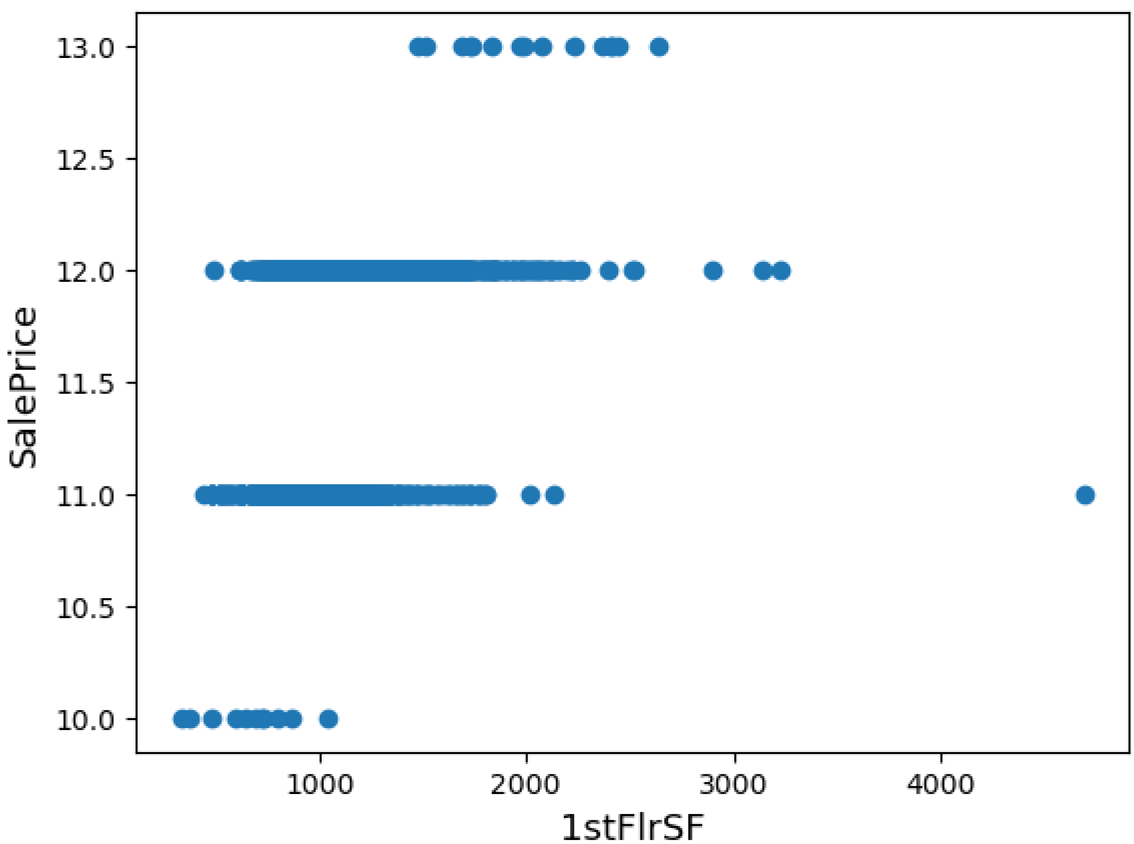

13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
1000
2000
3000
4000
5000
GrLivArea

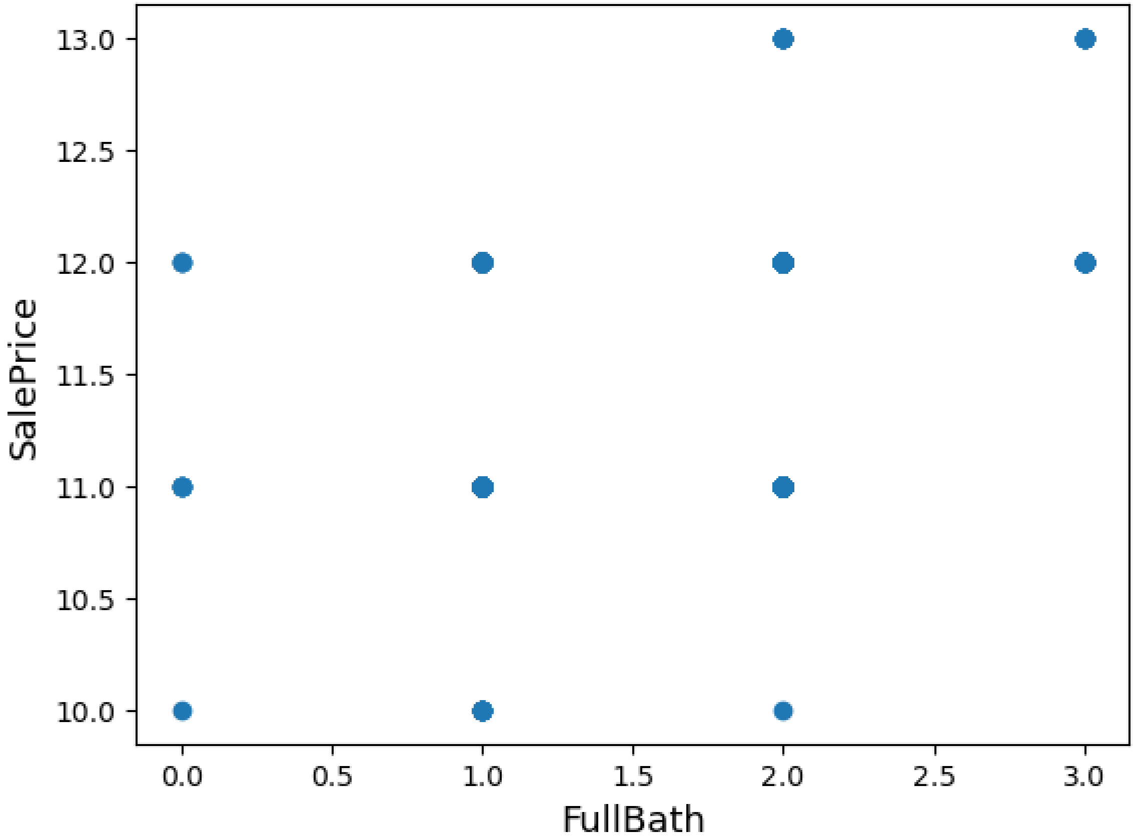

13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
2
4
6
8
10
OverallQual

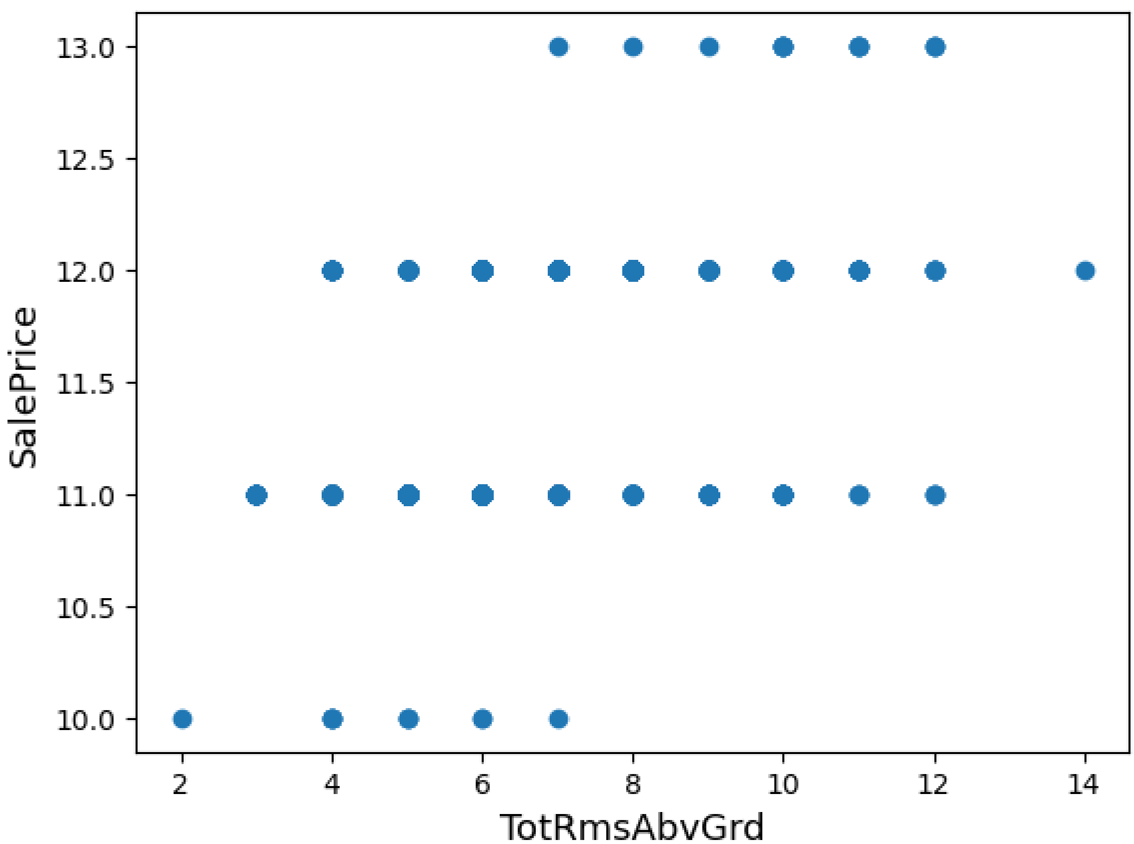

13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageCars

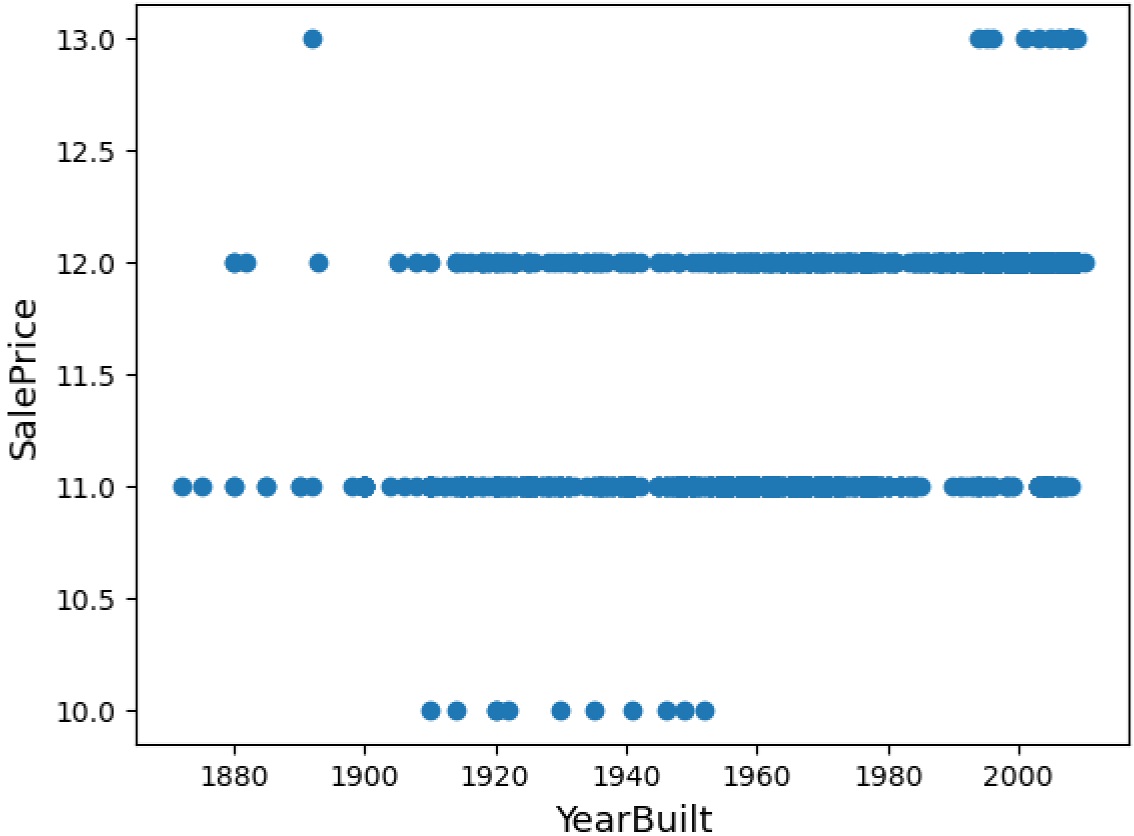

13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
0
200
400
600
800
1000
1200
1400
GarageArea


SalePrice
TotalBsmtSF
13.0
12.5
12.0
11.5
11.0
10.5
10.0
0
1000
2000
3000
4000
5000
6000


SalePrice
1stFlrSF
13.0
12.5
12.0
11.5
11.0
10.5
10.0
1000
2000
3000
4000


13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
FullBath

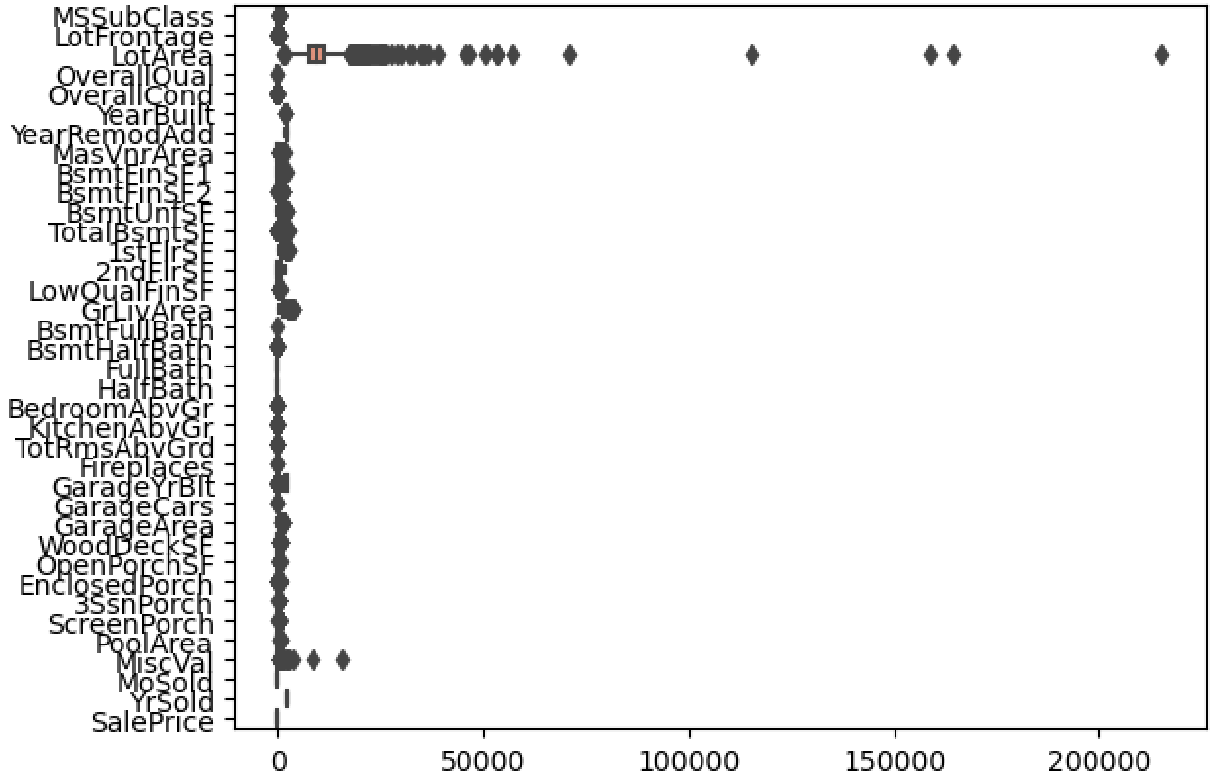

13.0
12.5
12.0
11.5
11.0
10.5
10.0
SalePrice
2
4
6
8
10
12
14
TotRmsAbvGrd



In [72]:
```
# above all highly correlated with output so i check thier outliers
df = df.drop(df[(df['GrLivArea']>4000) & (df['SalePrice']<300000)].index)
df = df.drop(df[(df['GarageArea']>1200) & (df['SalePrice']<500000)].index)
df = df.drop(df[(df['TotalBsmtSF']>3000) & (df['SalePrice']<700000)].index)
df = df.drop(df[(df['1stFlrSF']>2700) & (df['1stFlrSF']<700000)].index)
```

In [73]:
```
#check outliers
sns.boxplot(data=df,orient="h")
```

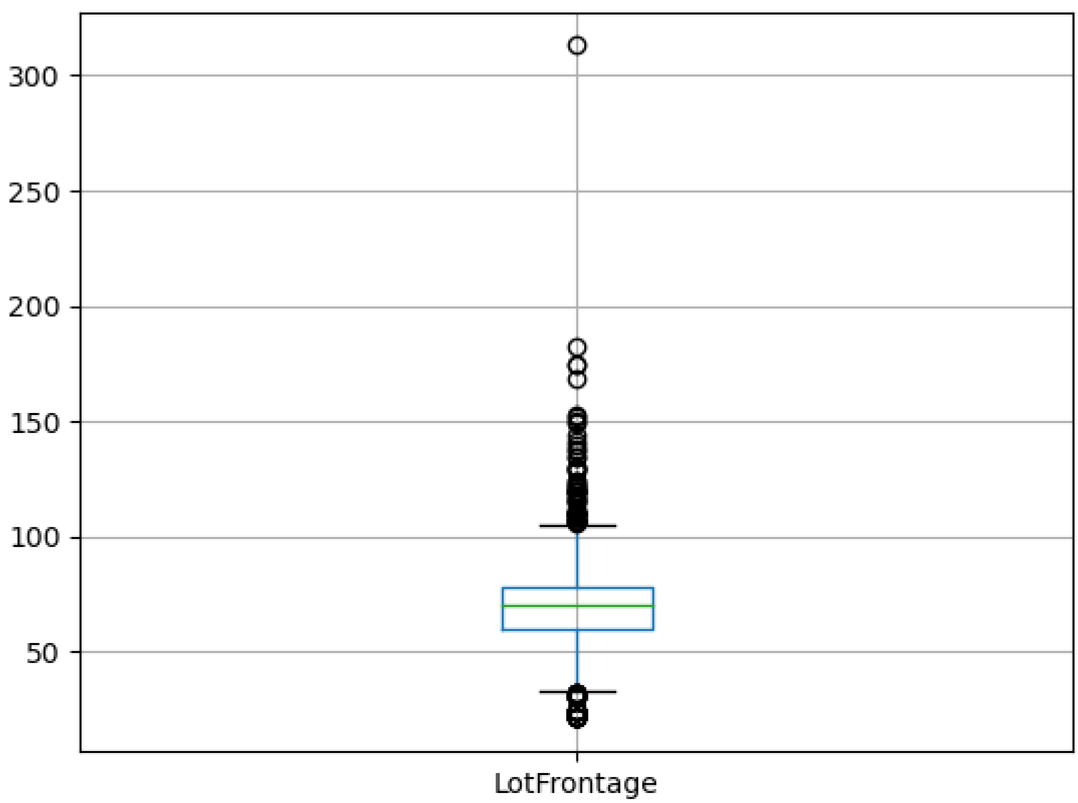
Out[73]: <AxesSubplot:>



In [83]:

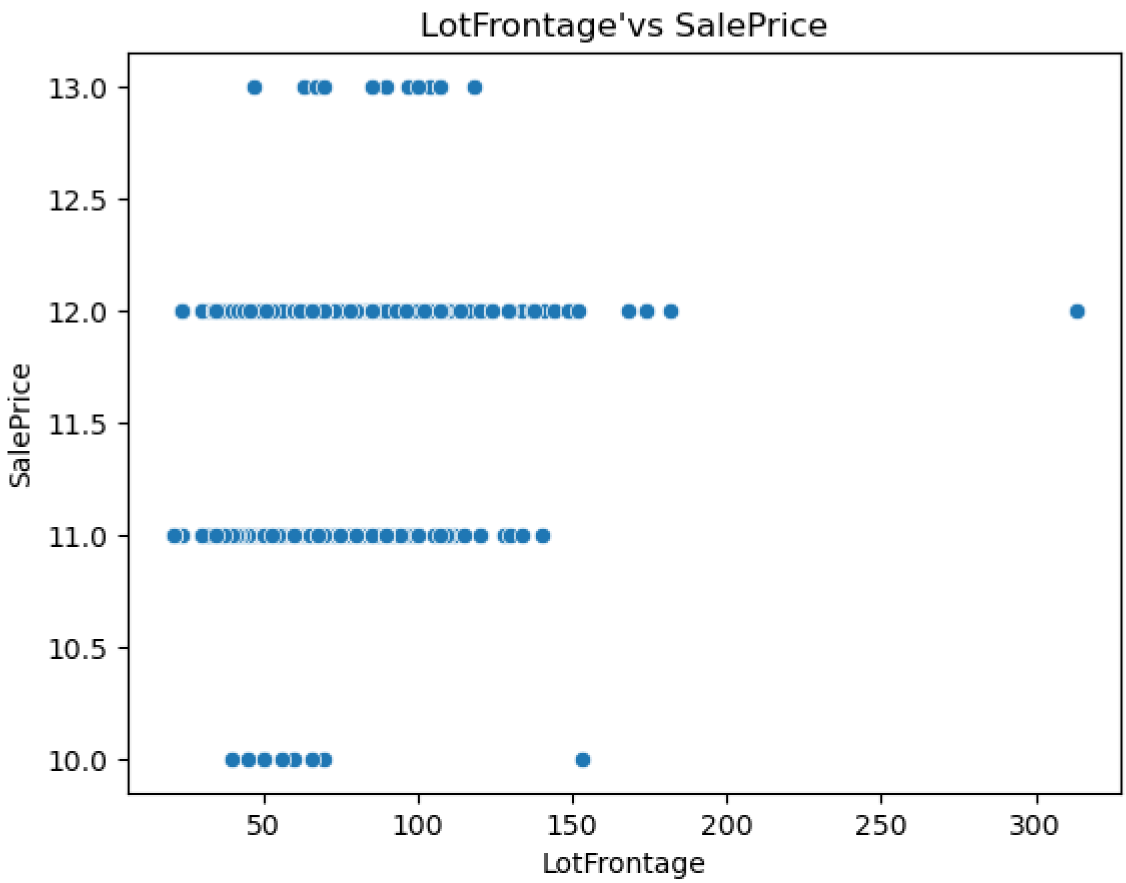
```
# box plot of numeric column only
df.boxplot(column =['LotFrontage'])
```

Out[83]: <AxesSubplot:>



not useful boxplot for outllier display sns.set_style("whitegrid") plt.figure(figsize=(20,10)) ax = sns.boxplot(x="LotFrontage", y="SalePrice" ,hue='MSZoning',data=df,width=0.8) plt.show()

In [88]:
```
sns.scatterplot(df['LotFrontage'], df["SalePrice"])

plt.title("LotFrontage'vs SalePrice")
plt.show()
```



In [113…
```
 #i decided to replace  lLotFrontage house  i impute value with median only
print('skewness value of : ',df['LotFrontage'].skew()) # skew value between -1 to 1 sc
# LotFrontage area increse sale prize also increase so need to remove LotFrontage outl
```

```
skewness value of :  1.7045256035173681
```

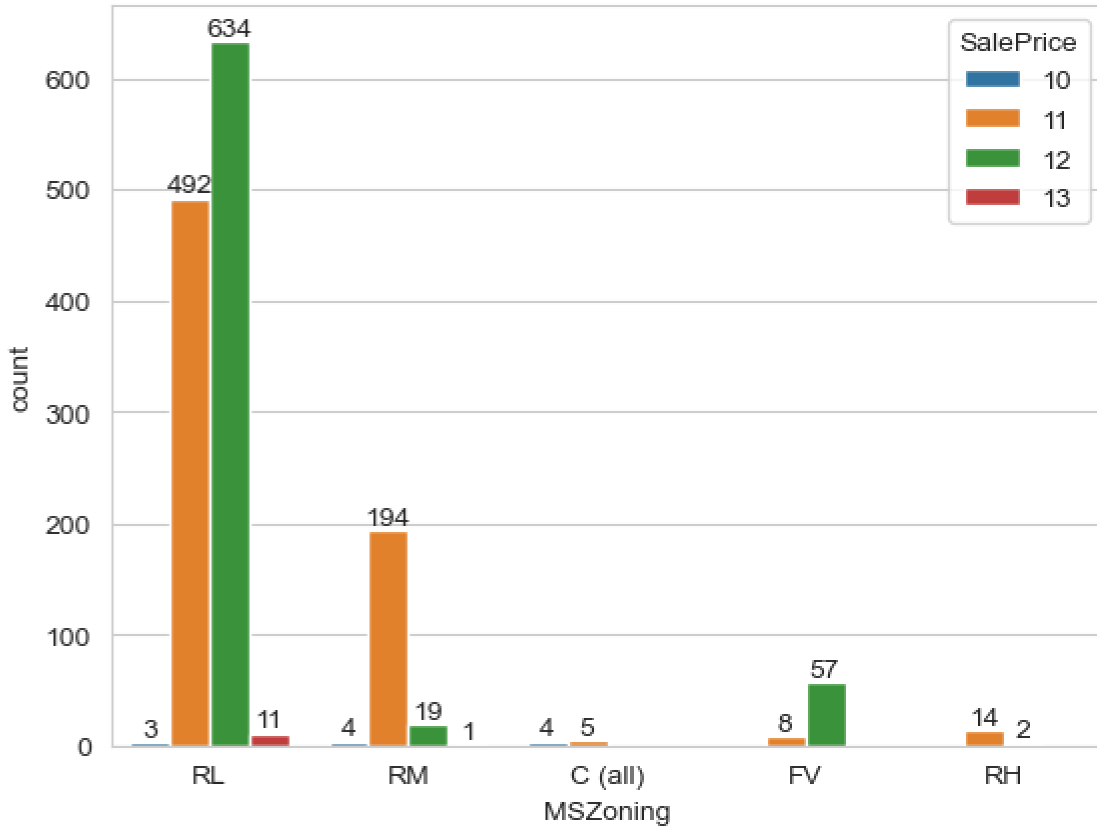
In [79]:
```
numeric
```

Out[79]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

3. EDA - DATA VISUALIZATION

In [118…
```
# first column msc=subclass vs street
ax = sns.countplot(data = df, x ='MSZoning', hue = 'SalePrice')
```

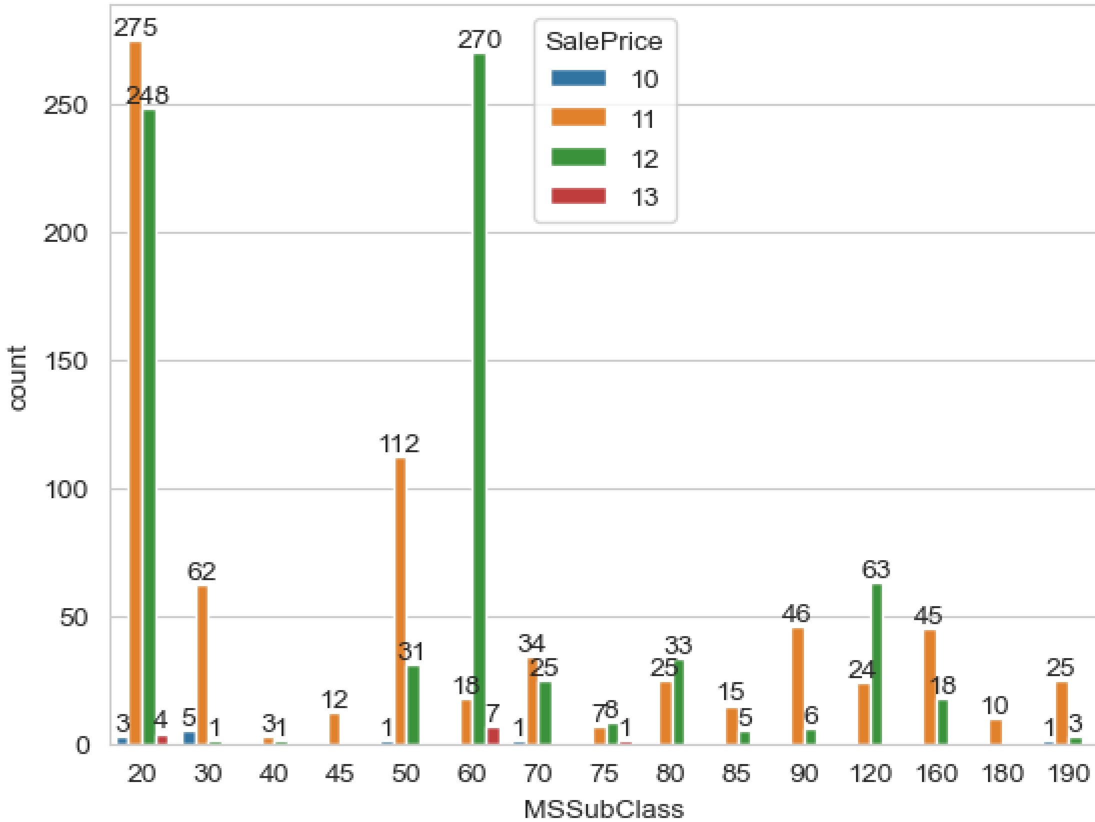
```
for bars in ax.containers:
    ax.bar_label(bars)
```



INSIGHT RL MSZoning have large sale count

In [121…

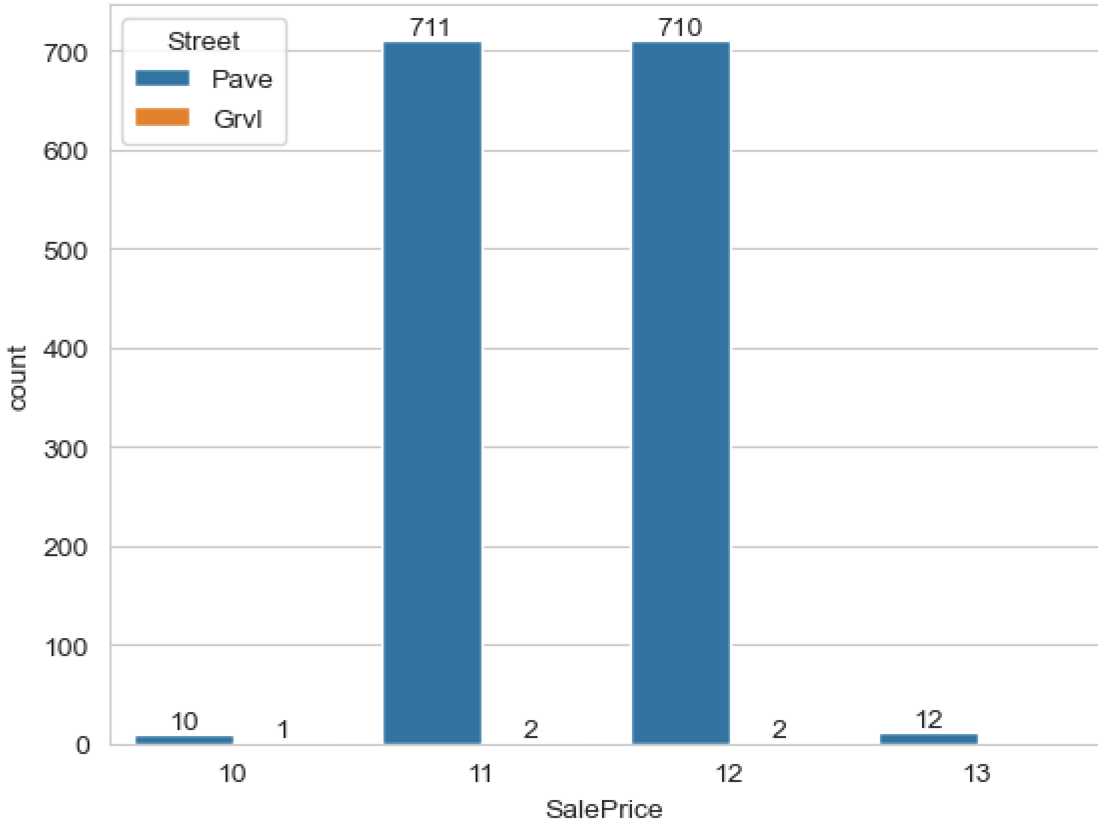
```
# first column msc=subclass vs street
ax = sns.countplot(data = df, x =  'MSSubClass', hue = 'SalePrice')

for bars in ax.containers:
    ax.bar_label(bars)
# at pave street more no. of building glass  grvl street dont have mssubclass
```



11 and 12 buliding class - have large count sale # 20 , 60 number also have more sale

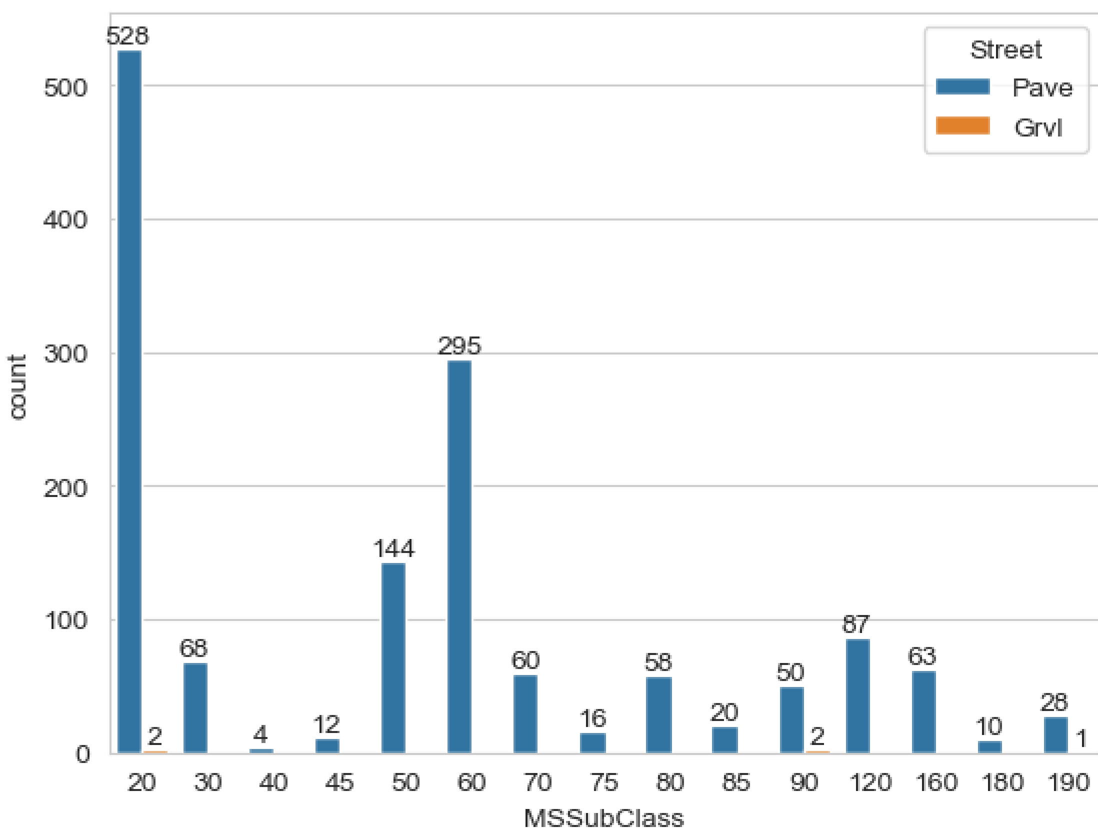
In [119...

```
# first column msc=subclass vs street
ax = sns.countplot(data = df, x =  'SalePrice', hue = 'Street')

for bars in ax.containers:
    ax.bar_label(bars)
# at pave street more no. of building glass  grvl street dont have mssubclass
# only pave street have more sales ,Grvi strret no sale at all
# only 11 and 12 large counts.
```



In [123…

```
# first column msc=subclass vs street
ax = sns.countplot(data = df, x = 'MSSubClass', hue = 'Street')

for bars in ax.containers:
    ax.bar_label(bars)
```



In [133…
```
## identifying numeric variables
numeric = df.select_dtypes(include=['float64','int64'])
numeric = numeric.columns
```

In [134…
```
numeric
```

Out[134]:
```
Index(['MSSubClass', 'LotFrontage', 'LotArea', 'OverallQual', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces',
       'GarageYrBlt', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice'],
      dtype='object')
```

taling too much time #sns.pairplot(df, hue ='SalePrice') # plt.show()## making pairplots to identify trends plt.figure(figsize = (20,50)) i = 1 for x in numeric : plt.subplot(len(numeric)//3+1,3,i) sns.scatterplot(y='SalePrice',x=x,data=df) plt.xticks(rotation = 'vertical') i = i+1 plt.tight_layout() plt.show()

plot categorical columns

In [131…
```
# plot categorical columns
##identifying categorical variables
categorical = df.select_dtypes(include=['object'])
categorical = categorical.columns
```

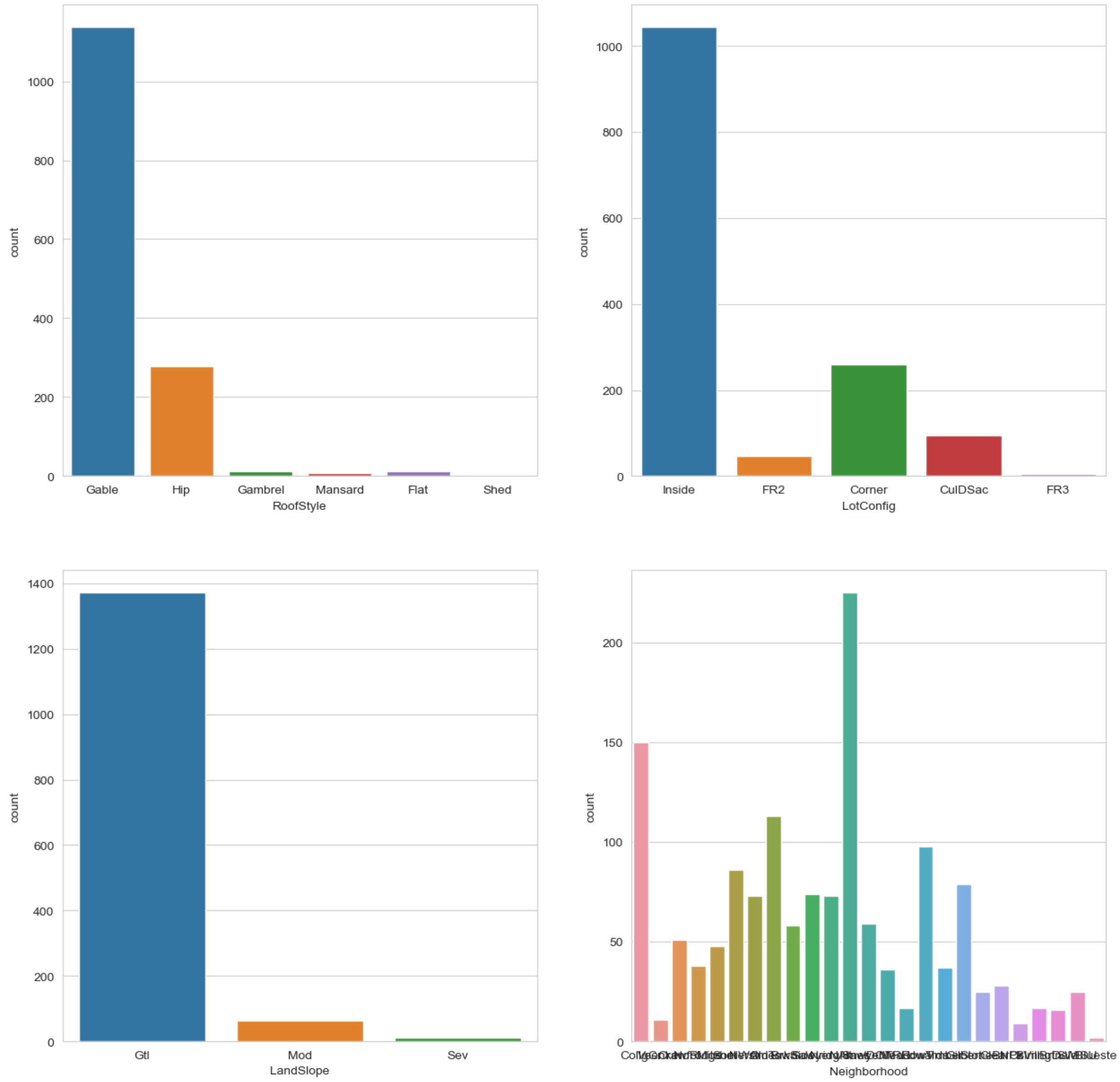
In [132…
```
categorical
```

Out[132]:
```
Index(['MSZoning', 'Street', 'LotShape', 'LandContour', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd',
       'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual', 'Functional',
       'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond', 'PavedDrive',
       'SaleType', 'SaleCondition'],
      dtype='object')
```

In [ ]:
```
plt.figure(figsize=(15,15))
plt.subplot(221)
sns.countplot(x='MSZoning', data=df)
plt.subplot(222)
sns.countplot(x='Street', data=df, hue="SalePrice")
plt.subplot(223)
sns.countplot(x='LotShape', data=df)
plt.subplot(224)
sns.countplot(x='LandContour', data=df)
```

In [128…
```
# outside part
plt.figure(figsize=(15,15))
plt.subplot(221)
sns.countplot(x='RoofStyle', data=df)
plt.subplot(222)
sns.countplot(x='LotConfig', data=df)
plt.subplot(223)
sns.countplot(x='LandSlope', data=df)
plt.subplot(224)
sns.countplot(x='Neighborhood', data=df)
```

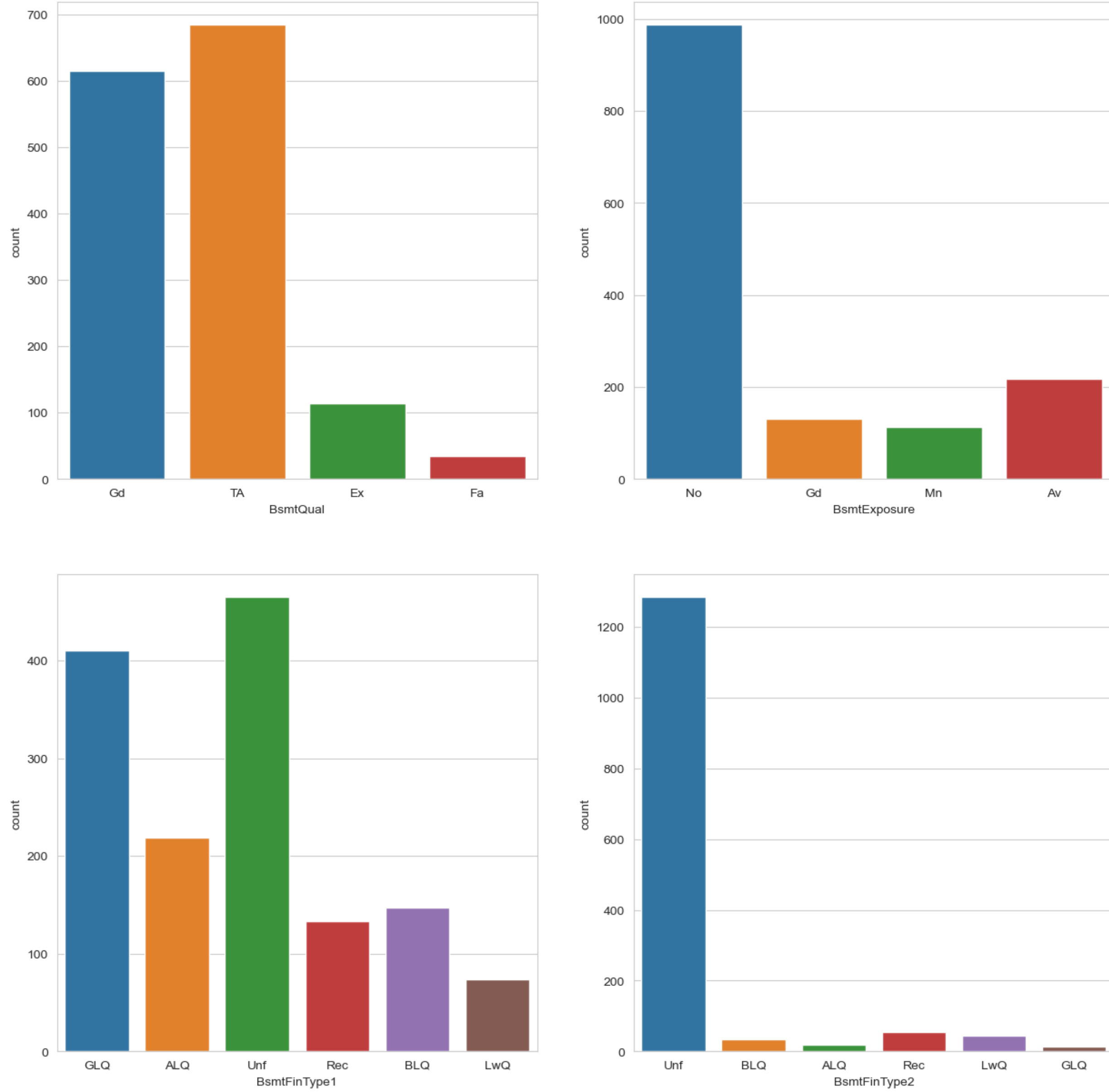
Out[128]: <AxesSubplot:xlabel='Neighborhood', ylabel='count'>



In [129…

```
# basement features -categorical
plt.figure(figsize=(15,15))
plt.subplot(221)
sns.countplot(x='BsmtQual', data=df)
plt.subplot(222)
sns.countplot(x='BsmtExposure', data=df)
plt.subplot(223)
sns.countplot(x='BsmtFinType1', data=df)
plt.subplot(224)
sns.countplot(x= 'BsmtFinType2', data=df)
```

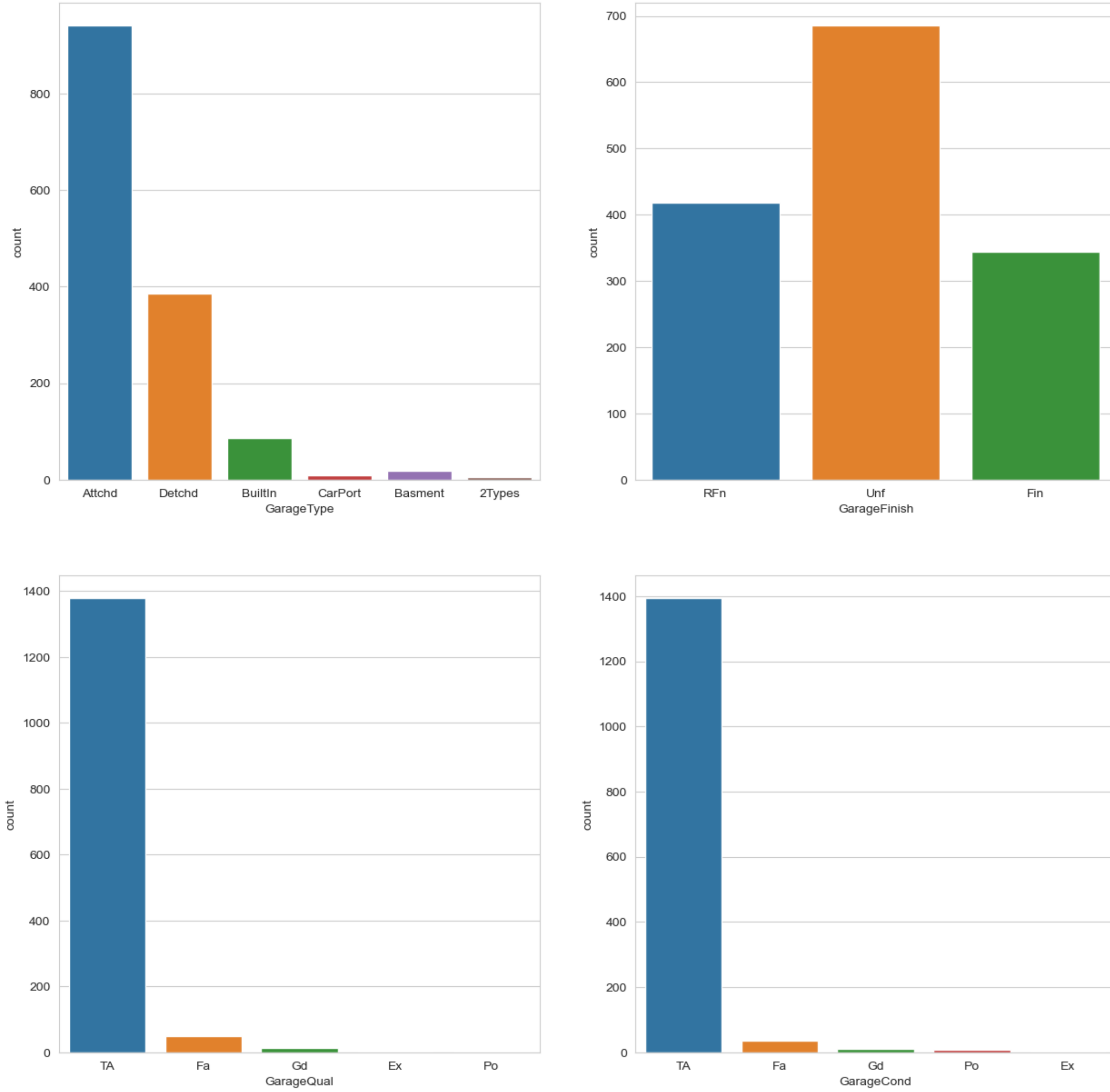
Out[129]: <AxesSubplot:xlabel='BsmtFinType2', ylabel='count'>



In [130…

```
# plot garage features - categorical
plt.figure(figsize=(15,15))
plt.subplot(221)
sns.countplot(x='GarageType', data=df)
plt.subplot(222)
sns.countplot(x='GarageFinish', data=df)
plt.subplot(223)
sns.countplot(x='GarageQual', data=df)
plt.subplot(224)
sns.countplot(x='GarageCond', data=df)
```

Out[130]: <AxesSubplot:xlabel='GarageCond', ylabel='count'>



Insights of univariate analysis - - RL MSZoning large amount of sale - write all above insights# GarageQual, GarageCond same values so we can drop any one # i will drop same columns charactersitcs -BldgType: Type of dwelling, HouseStyle: Style of dwelling drop any 1 # df.drop['GarageQual'] df.drop(columns=["Letter", "GDP per capita"], inplace=True) # check on tomarrow do encoding

In [ ]:
```
df.shape
```