HOUSING: PRICE PREDICTION

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. The "Data file.csv" and "Data description.txt" are enclosed with this file

Import Dependencies

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Exploratory Data Analysis

In [2]:
```
#Setting option to show max rows and max columns
pd.set_option("display.max_columns",None)
pd.set_option("display.max_rows", None)
```

Loading Data

In [3]:
```
test=pd.read_csv('housing_test.csv')
test.head()
```

Out[3]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfi
0	337	20	RL	86.0	14157	Pave	NaN	IR1	HLS	AllPub	Corne
1	1018	120	RL	NaN	5814	Pave	NaN	IR1	Lvl	AllPub	CulDSa
2	929	20	RL	NaN	11838	Pave	NaN	Reg	Lvl	AllPub	Insid
3	1148	70	RL	75.0	12000	Pave	NaN	Reg	Bnk	AllPub	Insid
4	1227	60	RL	86.0	14598	Pave	NaN	IR1	Lvl	AllPub	CulDSa

In [4]:
```
train=pd.read_csv('housing_train.csv')
train.head()
```

Out[4]:

	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

Shape of data

In [5]:
```
print(test.shape, train.shape)
```

```
(292, 80) (1168, 81)
```

Data Types

In [6]:
```
test .dtypes
```

Out[6]:

```
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           float64
Fence             object
MiscFeature       object
MiscVal            int64
MoSold             int64
```

```
YrSold            int64
SaleType         object
SaleCondition    object
dtype: object
```

In [7]:
```
train.dtypes
```

Out[7]:
```
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
```

Data Info

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 292 entries, 0 to 291
Data columns (total 80 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             292 non-null    int64
 1   MSSubClass     292 non-null    int64
 2   MSZoning       292 non-null    object
 3   LotFrontage    247 non-null    float64
 4   LotArea        292 non-null    int64
 5   Street         292 non-null    object
 6   Alley          14 non-null     object
 7   LotShape       292 non-null    object
 8   LandContour    292 non-null    object
 9   Utilities      292 non-null    object
 10  LotConfig      292 non-null    object
 11  LandSlope      292 non-null    object
 12  Neighborhood   292 non-null    object
 13  Condition1     292 non-null    object
 14  Condition2     292 non-null    object
 15  BldgType       292 non-null    object
 16  HouseStyle     292 non-null    object
 17  OverallQual    292 non-null    int64
 18  OverallCond    292 non-null    int64
 19  YearBuilt      292 non-null    int64
 20  YearRemodAdd   292 non-null    int64
 21  RoofStyle      292 non-null    object
 22  RoofMatl       292 non-null    object
 23  Exterior1st    292 non-null    object
 24  Exterior2nd    292 non-null    object
 25  MasVnrType     291 non-null    object
 26  MasVnrArea     291 non-null    float64
 27  ExterQual      292 non-null    object
 28  ExterCond      292 non-null    object
 29  Foundation     292 non-null    object
 30  BsmtQual       285 non-null    object
 31  BsmtCond       285 non-null    object
```

```
 32  BsmtExposure   285 non-null    object
 33  BsmtFinType1   285 non-null    object
 34  BsmtFinSF1     292 non-null    int64
 35  BsmtFinType2   285 non-null    object
 36  BsmtFinSF2     292 non-null    int64
 37  BsmtUnfSF      292 non-null    int64
 38  TotalBsmtSF    292 non-null    int64
 39  Heating        292 non-null    object
 40  HeatingQC      292 non-null    object
 41  CentralAir     292 non-null    object
 42  Electrical     291 non-null    object
 43  1stFlrSF       292 non-null    int64
 44  2ndFlrSF       292 non-null    int64
 45  LowQualFinSF   292 non-null    int64
 46  GrLivArea      292 non-null    int64
 47  BsmtFullBath   292 non-null    int64
 48  BsmtHalfBath   292 non-null    int64
 49  FullBath       292 non-null    int64
 50  HalfBath       292 non-null    int64
 51  BedroomAbvGr   292 non-null    int64
 52  KitchenAbvGr   292 non-null    int64
 53  KitchenQual    292 non-null    object
 54  TotRmsAbvGrd   292 non-null    int64
 55  Functional     292 non-null    object
 56  Fireplaces     292 non-null    int64
 57  FireplaceQu    153 non-null    object
 58  GarageType     275 non-null    object
 59  GarageYrBlt    275 non-null    float64
 60  GarageFinish   275 non-null    object
 61  GarageCars     292 non-null    int64
 62  GarageArea     292 non-null    int64
 63  GarageQual     275 non-null    object
 64  GarageCond     275 non-null    object
 65  PavedDrive     292 non-null    object
 66  WoodDeckSF     292 non-null    int64
 67  OpenPorchSF    292 non-null    int64
 68  EnclosedPorch  292 non-null    int64
 69  3SsnPorch      292 non-null    int64
 70  ScreenPorch    292 non-null    int64
 71  PoolArea       292 non-null    int64
 72  PoolQC         0 non-null      float64
 73  Fence          44 non-null     object
 74  MiscFeature    10 non-null     object
 75  MiscVal        292 non-null    int64
 76  MoSold         292 non-null    int64
 77  YrSold         292 non-null    int64
 78  SaleType       292 non-null    object
 79  SaleCondition  292 non-null    object
dtypes: float64(4), int64(34), object(42)
memory usage: 182.6+ KB
```

In [9]:
```
train.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
```

Checking Duplicates

In [10]:
```
## Checking for duplicates on train data
train.duplicated().sum()
```

Out[10]:
```
0
```

In [11]:
```
## Checking for duplicates on train data
test.duplicated().sum()
```

Out[11]:
```
0
```

We dont have any duplicates for both train and test data

Checking for Null values

In [12]:
```
train.isnull().sum()
```

Out[12]:
```
Id                   0
MSSubClass           0
MSZoning             0
LotFrontage        214
LotArea              0
Street               0
Alley             1091
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           7
MasVnrArea           7
ExterQual            0
ExterCond            0
Foundation           0
BsmtQual            30
BsmtCond            30
BsmtExposure        31
BsmtFinType1        30
```

```
BsmtFinSF1          0
BsmtFinType2       31
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
FireplaceQu       551
GarageType         64
GarageYrBlt        64
GarageFinish       64
GarageCars          0
GarageArea          0
GarageQual         64
GarageCond         64
PavedDrive          0
WoodDeckSF          0
OpenPorchSF         0
EnclosedPorch       0
3SsnPorch           0
ScreenPorch         0
PoolArea            0
PoolQC           1161
Fence             931
MiscFeature      1124
MiscVal             0
MoSold              0
YrSold              0
SaleType            0
SaleCondition       0
SalePrice           0
dtype: int64
```

In [13]:
```
test.isnull().sum()
```

Out[13]:
```
Id                0
MSSubClass        0
MSZoning          0
LotFrontage      45
LotArea           0
Street            0
Alley           278
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
PoolQC          292
Fence           248
MiscFeature     282
MiscVal           0
MoSold            0
YrSold            0
SaleType          0
SaleCondition     0
dtype: int64
```

We have a lot of null values. we need to handle them

Data Cleaning

In [14]:
```
# We will drop 'ID' column for both train and test data as it has no relavance in predic
train.drop('Id', axis=1, inplace=True)
test.drop('Id', axis=1, inplace=True)
```

Handling Null values

In [15]:
```
# We will drop the column with high null values for both train and test data
train.drop(['Alley','FireplaceQu','PoolQC','Fence','MiscFeature'], axis=1, inplace=True)
test.drop(['Alley','FireplaceQu','PoolQC','Fence','MiscFeature'], axis=1, inplace=True)
```

In [16]:
```
null_column=train.columns[train.isnull().any()]
null_column
```

Out[16]:
```
Index(['LotFrontage', 'MasVnrType', 'MasVnrArea', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageQual', 'GarageCond'],
      dtype='object')
```

In [17]:
```
for i in null_column:
    print(i,'\n', train[i].unique())
```

```
LotFrontage
 [ nan  95.  92. 105.  58.  88.  70.  80.  50.  44. 129.  59.  55.  64.
  24.  68.  71.  74.  61.  60. 120.  84. 141.  30.  65.  76. 100.  85.
  75. 107. 122.  82.  62.  73.  79.  77.  41.  69.  90.  96.  72.  34.
  78.  63.  40.  98. 160. 108. 128.  51.  81.  99.  66.  37. 174.  87.
  53. 152.  47.  86.  56.  89.  35.  52.  21. 104.  57.  83.  46. 101.
 112. 149.  93.  49.  43. 130.  54.  91.  67.  97. 110. 103. 115.  94.
  48.  36. 313. 109. 144. 121. 102. 116. 182.  32.  42. 168. 118.  38.
 140. 134. 114. 124.  39. 111.  45. 106. 153.]
MasVnrType
 ['None' 'BrkFace' 'Stone' 'BrkCmn' nan]
MasVnrArea
 [0.000e+00 4.800e+02 1.260e+02 1.800e+02 6.700e+01 2.230e+02 6.600e+01
 8.200e+01 1.740e+02 3.040e+02 1.720e+02 1.660e+02 1.840e+02 3.500e+02
 4.120e+02 1.000e+00 1.890e+02 1.120e+02 5.000e+01 2.200e+02 1.600e+01
 6.300e+02 2.810e+02 2.870e+02 3.400e+02 2.160e+02       nan 1.400e+02
 1.830e+02 3.360e+02 3.960e+02 2.320e+02 3.200e+02 8.500e+01 1.620e+02
 1.540e+02 1.760e+02 1.200e+02 1.080e+02 2.520e+02 1.300e+02 3.510e+02
 5.710e+02 5.300e+01 2.040e+02 9.750e+02 6.530e+02 8.000e+01 4.720e+02
 3.400e+01 1.650e+02 2.370e+02 1.130e+02 9.000e+01 1.600e+02 1.470e+02
 1.360e+02 3.760e+02 8.900e+01 4.150e+02 2.000e+02 5.060e+02 2.860e+02
 4.500e+01 2.450e+02 2.470e+02 2.400e+02 7.000e+01 9.800e+01 6.040e+02
 8.600e+02 2.700e+01 1.530e+02 2.120e+02 2.060e+02 7.600e+02 2.960e+02
 5.280e+02 2.990e+02 3.380e+02 1.230e+02 2.330e+02 6.800e+01 3.910e+02
 2.100e+02 7.500e+01 9.600e+01 3.800e+02 3.620e+02 1.560e+02 3.990e+02
 7.400e+01 1.860e+02 4.100e+01 6.500e+01 1.440e+02 8.400e+01 2.080e+02
 2.600e+02 4.810e+02 5.670e+02 1.940e+02 1.250e+02 1.800e+01 2.050e+02
 3.090e+02 6.510e+02 7.200e+01 1.450e+02 7.660e+02 7.880e+02 7.620e+02
 2.720e+02 2.560e+02 4.400e+01 3.000e+02 1.680e+02 2.880e+02 4.660e+02
 2.800e+01 3.660e+02 2.920e+02 9.900e+01 2.240e+02 4.200e+02 1.040e+02
 4.910e+02 2.890e+02 2.680e+02 6.400e+02 6.000e+02 3.020e+02 3.600e+02
 4.240e+02 1.780e+02 1.960e+02 5.000e+02 1.047e+03 2.590e+02 2.540e+02
 1.020e+02 1.370e+02 1.640e+02 4.320e+02 2.980e+02 1.280e+02 4.250e+02
 5.130e+02 2.700e+02 8.100e+01 1.700e+02 5.730e+02 3.100e+02 2.030e+02
 1.820e+02 7.050e+02 1.380e+02 6.300e+01 6.600e+02 6.500e+02 4.420e+02
 2.430e+02 3.000e+01 2.260e+02 1.060e+02 2.200e+01 4.000e+01 3.810e+02
```

```
 3.440e+02 2.150e+02 2.380e+02 2.460e+02 1.010e+02 4.280e+02 7.960e+02
 3.240e+02 1.000e+02 8.940e+02 1.160e+02 1.220e+02 3.120e+02 5.600e+01
 4.480e+02 3.330e+02 2.660e+02 2.750e+02 5.760e+02 1.430e+02 2.360e+02
 6.400e+01 1.400e+01 8.800e+01 3.480e+02 4.800e+01 5.640e+02 6.730e+02
 2.800e+02 2.580e+02 1.170e+03 2.340e+02 1.350e+02 1.115e+03 4.380e+02
 2.900e+02 1.050e+02 2.500e+02 1.129e+03 4.260e+02 5.100e+01 4.200e+01
 7.680e+02 9.200e+01 3.420e+02 3.880e+02 1.670e+02 2.950e+02 1.100e+02
 7.720e+02 5.100e+02 9.220e+02 2.440e+02 3.060e+02 4.600e+01 4.560e+02
 2.020e+02 2.780e+02 4.510e+02 1.690e+02 1.570e+02 4.520e+02 4.430e+02
 1.140e+02 2.630e+02 1.378e+03 1.610e+02 1.580e+02 8.700e+02 1.320e+02
 3.150e+02 1.270e+02 4.590e+02 7.310e+02 7.600e+01 2.280e+02 2.250e+02
 4.100e+02 1.920e+02 8.160e+02 4.230e+02 7.480e+02 4.790e+02 1.510e+02
 5.400e+01 1.100e+01 8.600e+01 1.490e+02 5.940e+02 5.700e+01 3.700e+02
 3.350e+02 1.170e+02 1.150e+02 1.500e+02 4.350e+02 2.610e+02 6.000e+01
 3.050e+02 1.630e+02 1.090e+02 2.620e+02 1.600e+03 1.480e+02 2.850e+02
 1.710e+02 9.400e+01 1.190e+02 2.840e+02 9.500e+01 3.600e+01 6.160e+02
 3.650e+02 5.540e+02 4.640e+02 3.100e+01]
BsmtQual
 ['Gd' 'TA' 'Ex' nan 'Fa']
BsmtCond
 ['TA' 'Gd' 'Fa' nan 'Po']
BsmtExposure
 ['No' 'Gd' 'Av' 'Mn' nan]
BsmtFinType1
 ['ALQ' 'GLQ' 'BLQ' 'Unf' 'Rec' 'LwQ' nan]
BsmtFinType2
 ['Unf' 'Rec' 'BLQ' 'GLQ' nan 'ALQ' 'LwQ']
GarageType
 ['Attchd' 'BuiltIn' 'Detchd' 'Basment' nan '2Types' 'CarPort']
GarageYrBlt
 [1977. 1970. 1997. 2006. 1957. 1965. 1947. 1937. 2003. 1974. 1955. 1923.
 2002. 2007. 1987. 2001. 1988. 1950. 1961. 1953. 2010. 1922. 1939. 2005.
 1991. 1979. 1975. 1976. 1978. 1960. 1956. 2004. 1982. 2000. 1948.   nan
 1964. 1920. 1930. 1968. 1946. 1992. 1936. 1967. 1989. 1959. 1966. 1916.
 1941. 1998. 1962. 1926. 1925. 1983. 1999. 1969. 1985. 1993. 2008. 1971.
 1980. 1945. 1995. 1981. 1994. 1949. 1996. 1921. 1963. 1938. 1958. 1935.
 1940. 1990. 1910. 1954. 1927. 2009. 1986. 1929. 1984. 1973. 1924. 1942.
 1900. 1931. 1951. 1934. 1972. 1932. 1928. 1918. 1908. 1933. 1906. 1914.
 1952. 1915.]
GarageFinish
 ['RFn' 'Unf' 'Fin' nan]
GarageQual
 ['TA' 'Fa' nan 'Gd' 'Ex' 'Po']
GarageCond
 ['TA' 'Fa' 'Gd' nan 'Po' 'Ex']
```

In [18]:
```
train['GarageYrBlt'].fillna(train['YearBuilt'],inplace=True)
test['GarageYrBlt'].fillna(test['YearBuilt'],inplace=True)
```

In [19]:
```
train['GarageYrBlt'].unique()
```

Out[19]:
```
array([1977., 1970., 1997., 2006., 1957., 1965., 1947., 1937., 2003.,
       1974., 1955., 1923., 2002., 2007., 1987., 2001., 1988., 1950.,
       1961., 1953., 2010., 1922., 1939., 2005., 1991., 1979., 1975.,
       1976., 1978., 1960., 1956., 2004., 1982., 2000., 1948., 1946.,
       1964., 1920., 1930., 1968., 1992., 1936., 1967., 1989., 1959.,
       1935., 1966., 1931., 1916., 1941., 1998., 1962., 1926., 1925.,
       1983., 1999., 1969., 1985., 1993., 2008., 1971., 1980., 1945.,
       1995., 1981., 1994., 1949., 1940., 1996., 1921., 1924., 1963.,
       1938., 1910., 1958., 1911., 1990., 1954., 1927., 2009., 1986.,
       1929., 1984., 1973., 1900., 1942., 1951., 1972., 1934., 1932.,
       1928., 1918., 1908., 1914., 1933., 1875., 1906., 1952., 1915.,
       1912.])
```

In [20]:
```
null=train.columns[train.isnull().any()]
```

```
null
```

Out[20]:
```
Index(['LotFrontage', 'MasVnrType', 'MasVnrArea', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'GarageType',
       'GarageFinish', 'GarageQual', 'GarageCond'],
      dtype='object')
```

In [21]:
```
#replacing missing values in for numerical culumns with their mean value

# for train data
for i in null:
    if train[i].dtypes!='O':
        train[i].fillna(train[i].mean(),inplace=True)

# for test data
for i in null:
    if test[i].dtypes!='O':
        test[i].fillna(test[i].mean(),inplace=True)
```

In [22]:
```
#replacing missing values in for categorical culumns with their mode value

# for train data
for i in null:
    if train[i].dtypes=='O':
        train[i].fillna(train[i].mode()[0],inplace=True)

# for test data
for i in null:
    if test[i].dtypes=='O':
        test[i].fillna(test[i].mode()[0],inplace=True)
```

In [23]:
```
train.isnull().sum().sum()
```

Out[23]:
```
0
```

In [24]:
```
test.isnull().sum().sum()
```

Out[24]:
```
1
```

we have 1 column with null value in it for our test data

In [25]:
```
test.columns[test.isnull().any()]
```

Out[25]:
```
Index(['Electrical'], dtype='object')
```

In [26]:
```
test['Electrical'].unique()
```

Out[26]:
```
array(['SBrkr', 'FuseA', 'FuseP', nan, 'FuseF'], dtype=object)
```

In [27]:
```
test['Electrical'].fillna(test['Electrical'].mode()[0],inplace=True)
```

In [28]:
```
test.isnull().sum().sum()
```

Out[28]:
```
0
```

We have no null value left in our both test and train data

Replacing whitespaces if any

In [29]:
```
train.replace([' ','  '], ['',''], inplace=True)
test.replace([' ','  '], ['',''], inplace=True)
```

In [30]:
```
train.isnull().sum().sum()
```

Out[30]:
```
0
```

In [31]:
```
test.isnull().sum().sum()
```

Out[31]:
```
0
```

We dont have any whitespaces in both train and test data

Statistical Analysis

In [32]:
```
train.describe().T
```

Out[32]:

	count	mean	std	min	25%	50%	75%	max
MSSubClass	1168.0	56.767979	41.940650	20.0	20.00	50.00000	70.00	190.0
LotFrontage	1168.0	70.988470	22.437056	21.0	60.00	70.98847	79.25	313.0
LotArea	1168.0	10484.749144	8957.442311	1300.0	7621.50	9522.50000	11515.50	164660.0
OverallQual	1168.0	6.104452	1.390153	1.0	5.00	6.00000	7.00	10.0
OverallCond	1168.0	5.595890	1.124343	1.0	5.00	5.00000	6.00	9.0
YearBuilt	1168.0	1970.930651	30.145255	1875.0	1954.00	1972.00000	2000.00	2010.0
YearRemodAdd	1168.0	1984.758562	20.785185	1950.0	1966.00	1993.00000	2004.00	2010.0
MasVnrArea	1168.0	102.310078	182.047152	0.0	0.00	0.00000	160.00	1600.0
BsmtFinSF1	1168.0	444.726027	462.664785	0.0	0.00	385.50000	714.50	5644.0
BsmtFinSF2	1168.0	46.647260	163.520016	0.0	0.00	0.00000	0.00	1474.0
BsmtUnfSF	1168.0	569.721747	449.375525	0.0	216.00	474.00000	816.00	2336.0
TotalBsmtSF	1168.0	1061.095034	442.272249	0.0	799.00	1005.50000	1291.50	6110.0
1stFlrSF	1168.0	1169.860445	391.161983	334.0	892.00	1096.50000	1392.00	4692.0
2ndFlrSF	1168.0	348.826199	439.696370	0.0	0.00	0.00000	729.00	2065.0
LowQualFinSF	1168.0	6.380137	50.892844	0.0	0.00	0.00000	0.00	572.0
GrLivArea	1168.0	1525.066781	528.042957	334.0	1143.25	1468.50000	1795.00	5642.0
BsmtFullBath	1168.0	0.425514	0.521615	0.0	0.00	0.00000	1.00	3.0
BsmtHalfBath	1168.0	0.055651	0.236699	0.0	0.00	0.00000	0.00	2.0
FullBath	1168.0	1.562500	0.551882	0.0	1.00	2.00000	2.00	3.0
HalfBath	1168.0	0.388699	0.504929	0.0	0.00	0.00000	1.00	2.0
BedroomAbvGr	1168.0	2.884418	0.817229	0.0	2.00	3.00000	3.00	8.0
KitchenAbvGr	1168.0	1.045377	0.216292	0.0	1.00	1.00000	1.00	3.0
TotRmsAbvGrd	1168.0	6.542808	1.598484	2.0	5.00	6.00000	7.00	14.0
Fireplaces	1168.0	0.617295	0.650575	0.0	0.00	1.00000	1.00	3.0
GarageYrBlt	1168.0	1976.287671	26.376864	1875.0	1959.00	1978.00000	2001.00	2010.0

GarageCars	1168.0	1.776541	0.745554	0.0	1.00	2.00000	2.00	4.0
GarageArea	1168.0	476.860445	214.466769	0.0	338.00	480.00000	576.00	1418.0
WoodDeckSF	1168.0	96.206336	126.158988	0.0	0.00	0.00000	171.00	857.0
OpenPorchSF	1168.0	46.559932	66.381023	0.0	0.00	24.00000	70.00	547.0
EnclosedPorch	1168.0	23.015411	63.191089	0.0	0.00	0.00000	0.00	552.0
3SsnPorch	1168.0	3.639555	29.088867	0.0	0.00	0.00000	0.00	508.0
ScreenPorch	1168.0	15.051370	55.080816	0.0	0.00	0.00000	0.00	480.0
PoolArea	1168.0	3.448630	44.896939	0.0	0.00	0.00000	0.00	738.0
MiscVal	1168.0	47.315068	543.264432	0.0	0.00	0.00000	0.00	15500.0
MoSold	1168.0	6.344178	2.686352	1.0	5.00	6.00000	8.00	12.0
YrSold	1168.0	2007.804795	1.329738	2006.0	2007.00	2008.00000	2009.00	2010.0
SalePrice	1168.0	181477.005993	79105.586863	34900.0	130375.00	163995.00000	215000.00	755000.0

In [33]:
```
test.describe().T
```

Out[33]:

	count	mean	std	min	25%	50%	75%	max
MSSubClass	292.0	57.414384	43.780649	20.0	20.00	50.000000	70.00	190.0
LotFrontage	292.0	66.425101	19.975962	21.0	57.75	66.425101	76.00	150.0
LotArea	292.0	10645.143836	13330.669795	1526.0	7200.00	9200.000000	11658.75	215245.0
OverallQual	292.0	6.078767	1.356147	3.0	5.00	6.000000	7.00	10.0
OverallCond	292.0	5.493151	1.063267	3.0	5.00	5.000000	6.00	9.0
YearBuilt	292.0	1972.616438	30.447016	1872.0	1954.00	1976.000000	2001.00	2009.0
YearRemodAdd	292.0	1985.294521	20.105792	1950.0	1968.00	1994.000000	2003.25	2010.0
MasVnrArea	292.0	109.171821	174.729023	0.0	0.00	0.000000	180.00	1031.0
BsmtFinSF1	292.0	439.294521	429.559675	0.0	0.00	369.500000	700.50	1767.0
BsmtFinSF2	292.0	46.157534	152.467119	0.0	0.00	0.000000	0.00	1085.0
BsmtUnfSF	292.0	557.315068	411.043768	0.0	255.00	487.000000	780.00	1935.0
TotalBsmtSF	292.0	1042.767123	424.561153	0.0	771.75	971.000000	1322.00	3094.0
1stFlrSF	292.0	1133.691781	366.941919	372.0	858.00	1047.500000	1370.50	2402.0
2ndFlrSF	292.0	339.657534	424.278825	0.0	0.00	0.000000	717.00	1589.0
LowQualFinSF	292.0	3.702055	38.219527	0.0	0.00	0.000000	0.00	479.0
GrLivArea	292.0	1477.051370	514.199429	520.0	1061.50	1440.000000	1720.25	3447.0
BsmtFullBath	292.0	0.424658	0.508831	0.0	0.00	0.000000	1.00	2.0
BsmtHalfBath	292.0	0.065068	0.247070	0.0	0.00	0.000000	0.00	1.0
FullBath	292.0	1.575342	0.547856	0.0	1.00	2.000000	2.00	3.0
HalfBath	292.0	0.359589	0.494795	0.0	0.00	0.000000	1.00	2.0
BedroomAbvGr	292.0	2.794521	0.807336	0.0	2.00	3.000000	3.00	6.0
KitchenAbvGr	292.0	1.051370	0.236160	1.0	1.00	1.000000	1.00	3.0
TotRmsAbvGrd	292.0	6.417808	1.728105	3.0	5.00	6.000000	7.00	12.0

Fireplaces	292.0	0.595890	0.621259	0.0	0.00	1.000000	1.00	2.0
GarageYrBlt	292.0	1977.386986	26.050656	1872.0	1962.00	1978.000000	2002.00	2010.0
GarageCars	292.0	1.729452	0.754430	0.0	1.00	2.000000	2.00	4.0
GarageArea	292.0	457.458904	210.785591	0.0	300.00	467.500000	569.75	1052.0
WoodDeckSF	292.0	86.397260	121.898836	0.0	0.00	0.000000	149.25	728.0
OpenPorchSF	292.0	47.061644	65.865449	0.0	0.00	28.500000	66.00	418.0
EnclosedPorch	292.0	17.708904	51.892906	0.0	0.00	0.000000	0.00	330.0
3SsnPorch	292.0	2.489726	30.247488	0.0	0.00	0.000000	0.00	407.0
ScreenPorch	292.0	15.099315	58.483473	0.0	0.00	0.000000	0.00	396.0
PoolArea	292.0	0.000000	0.000000	0.0	0.00	0.000000	0.00	0.0
MiscVal	292.0	28.184932	224.036218	0.0	0.00	0.000000	0.00	3500.0
MoSold	292.0	6.232877	2.774556	1.0	4.00	6.000000	8.00	12.0
YrSold	292.0	2007.859589	1.322867	2006.0	2007.00	2008.000000	2009.00	2010.0

Data Visualisation and graphical analysis

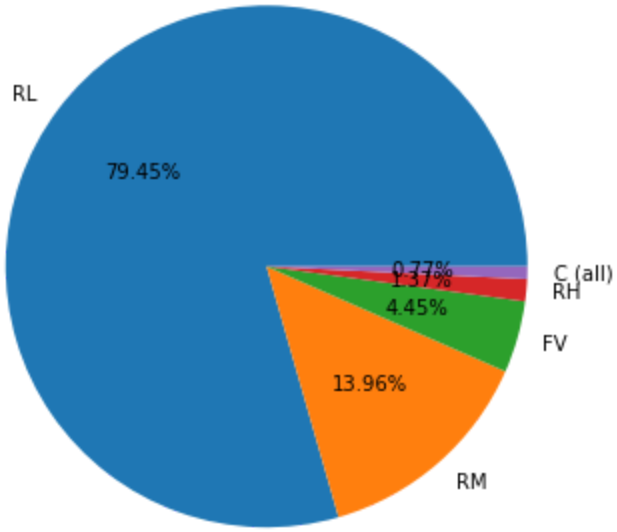
In [34]:
```
train.head()
```

Out[34]:

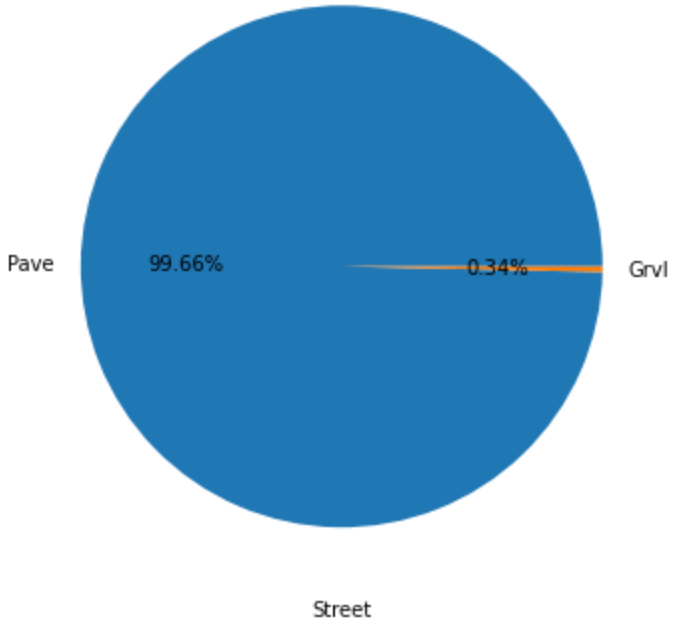
	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope
0	120	RL	70.98847	4928	Pave	IR1	Lvl	AllPub	Inside	Gtl
1	20	RL	95.00000	15865	Pave	IR1	Lvl	AllPub	Inside	Mod
2	60	RL	92.00000	9920	Pave	IR1	Lvl	AllPub	CulDSac	Gtl
3	20	RL	105.00000	11751	Pave	IR1	Lvl	AllPub	Inside	Gtl
4	20	RL	70.98847	16635	Pave	IR1	Lvl	AllPub	FR2	Gtl

We will use pie chart for all the categorical columns

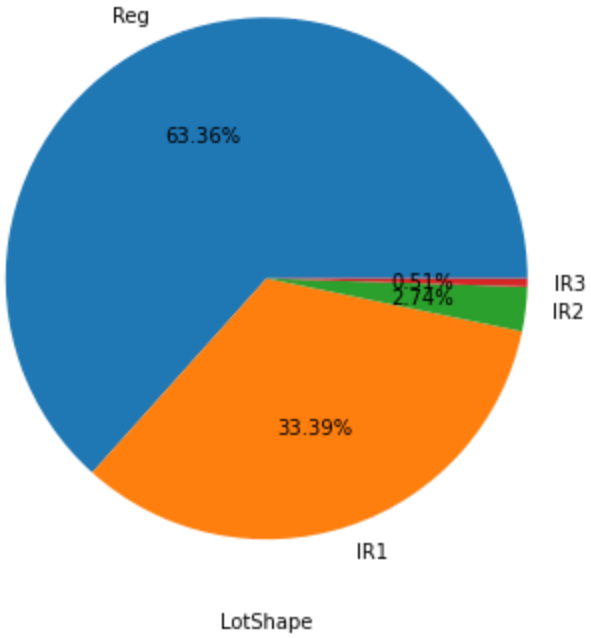
In [35]:
```
#Ploting pie chart for evry subject
for i in train:
    if train[i].dtypes=='O':
        plt.figure(figsize=(6,6))
        train[i].value_counts().plot.pie(autopct='%.2f%%')
        plt.xlabel(i)
        plt.ylabel('')
        plt.figure()
```



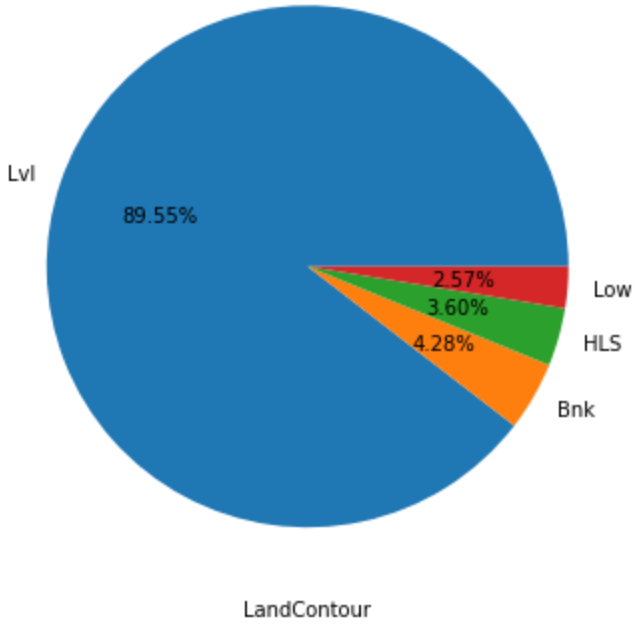
<Figure size 432x288 with 0 Axes>



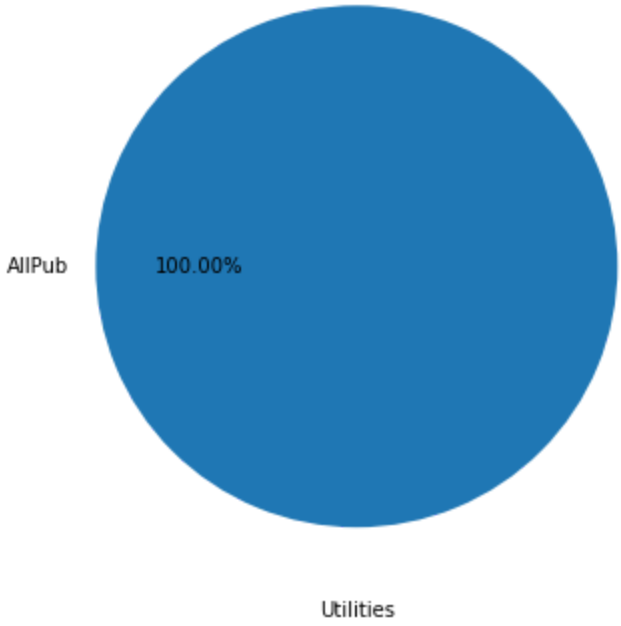
<Figure size 432x288 with 0 Axes>



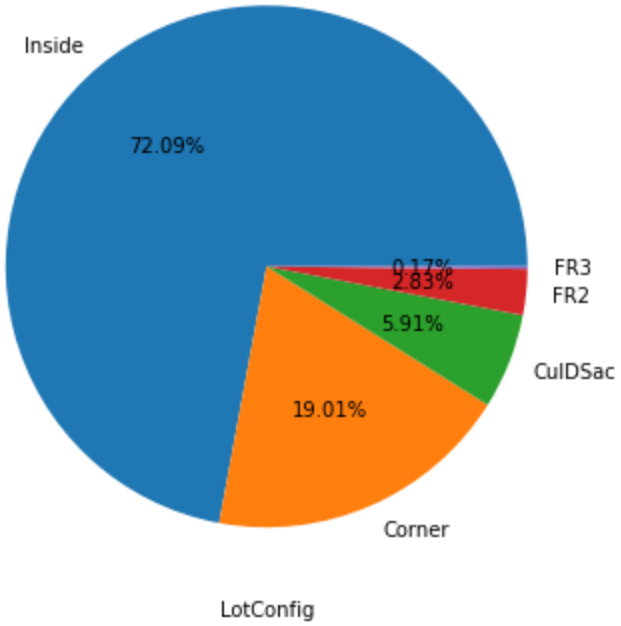
<Figure size 432x288 with 0 Axes>



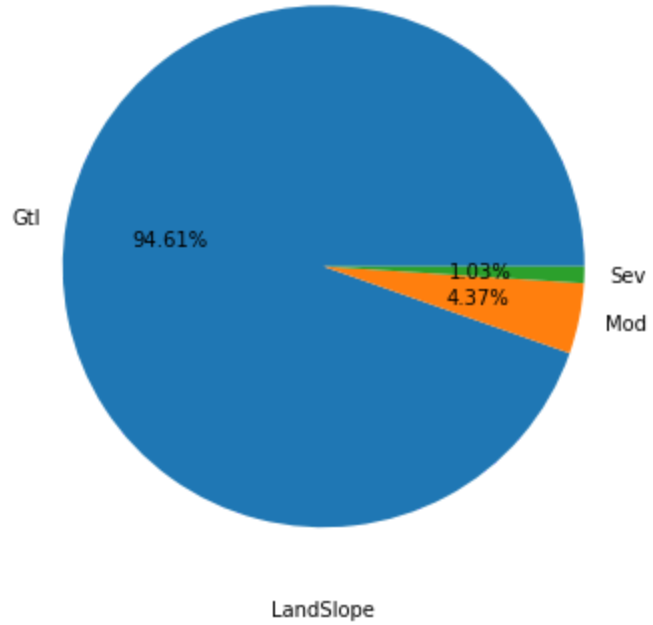
<Figure size 432x288 with 0 Axes>



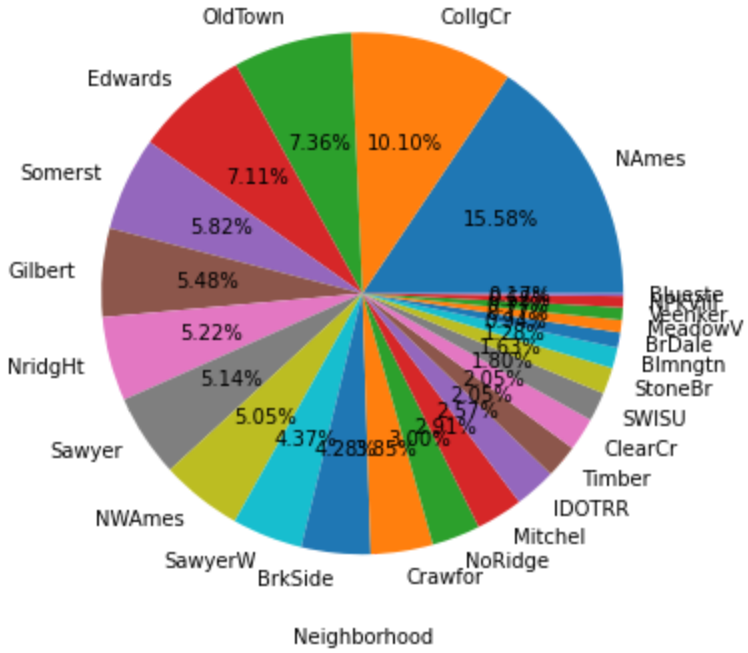
<Figure size 432x288 with 0 Axes>



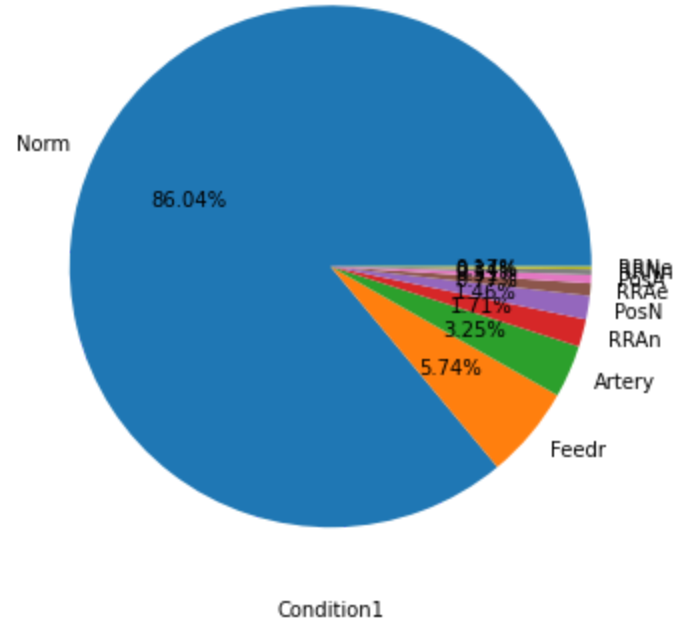
<Figure size 432x288 with 0 Axes>



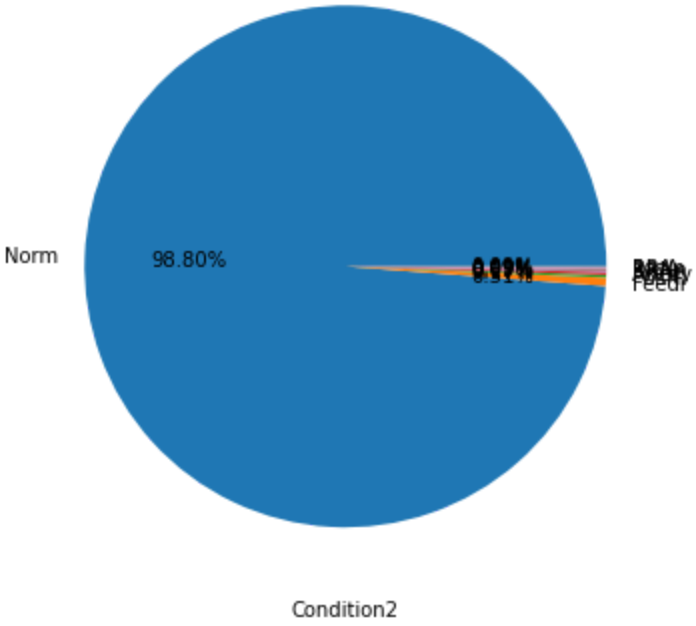
<Figure size 432x288 with 0 Axes>



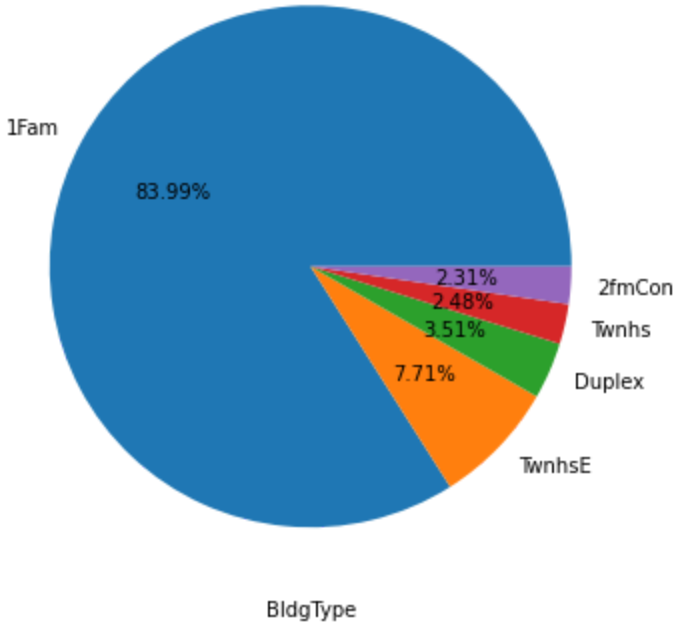
<Figure size 432x288 with 0 Axes>



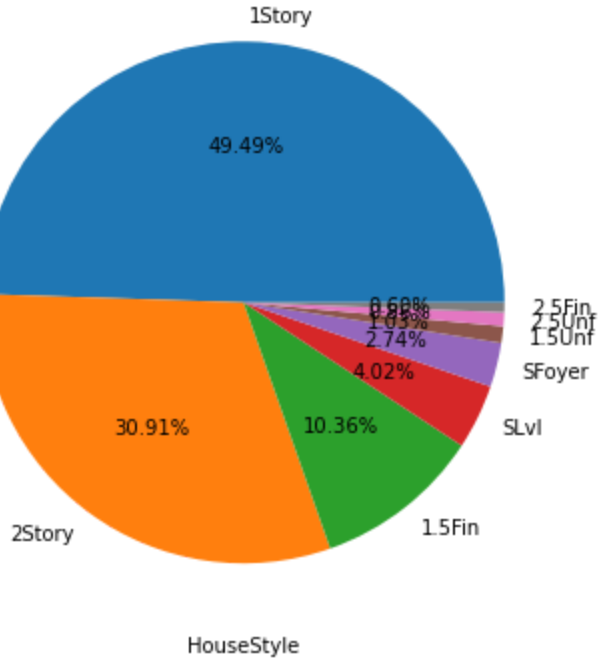
<Figure size 432x288 with 0 Axes>



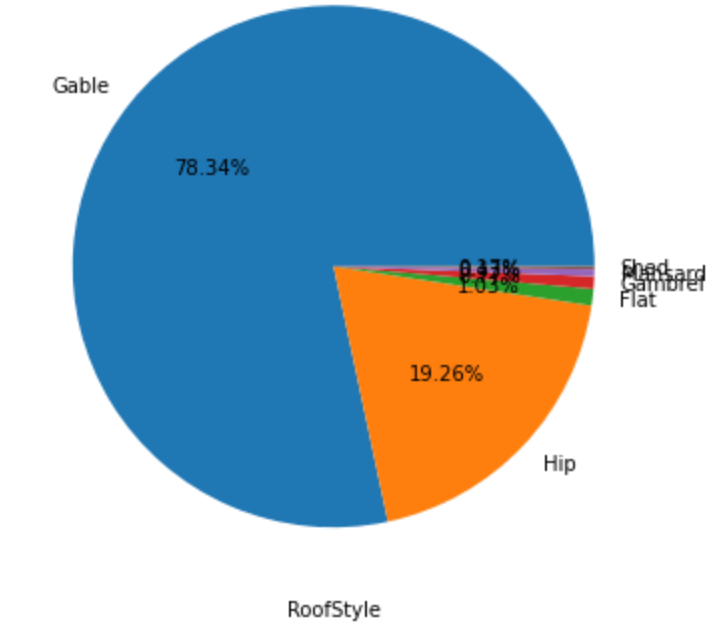
<Figure size 432x288 with 0 Axes>



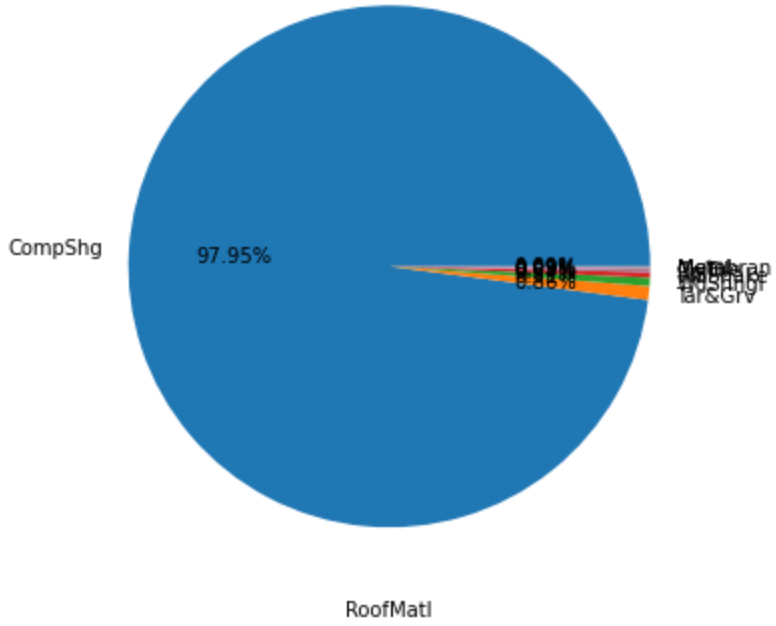
<Figure size 432x288 with 0 Axes>



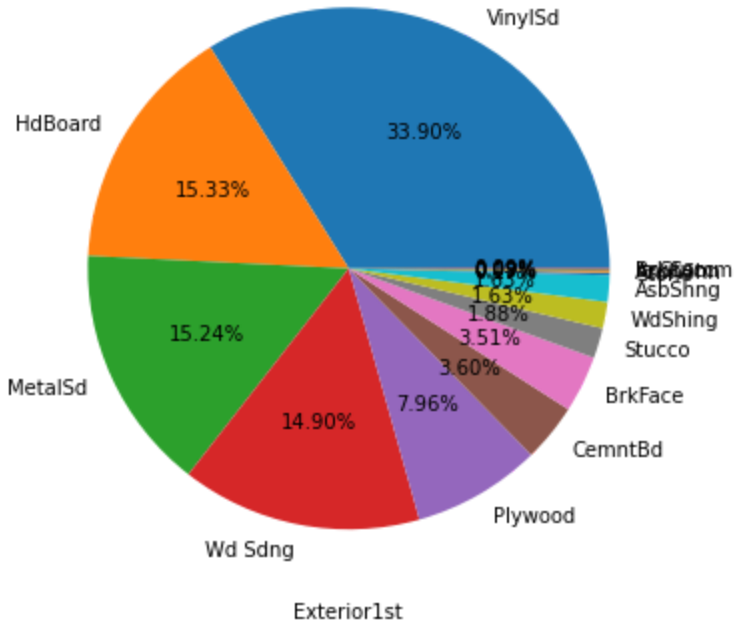
<Figure size 432x288 with 0 Axes>



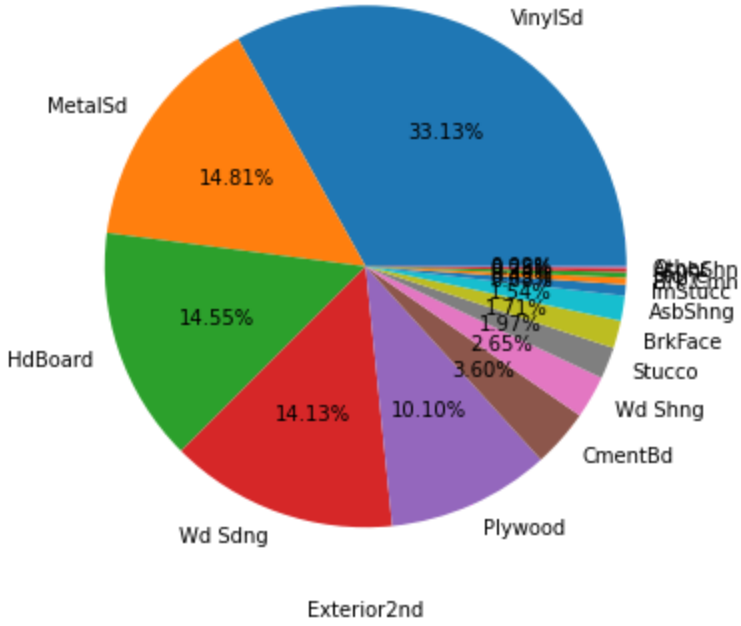
<Figure size 432x288 with 0 Axes>



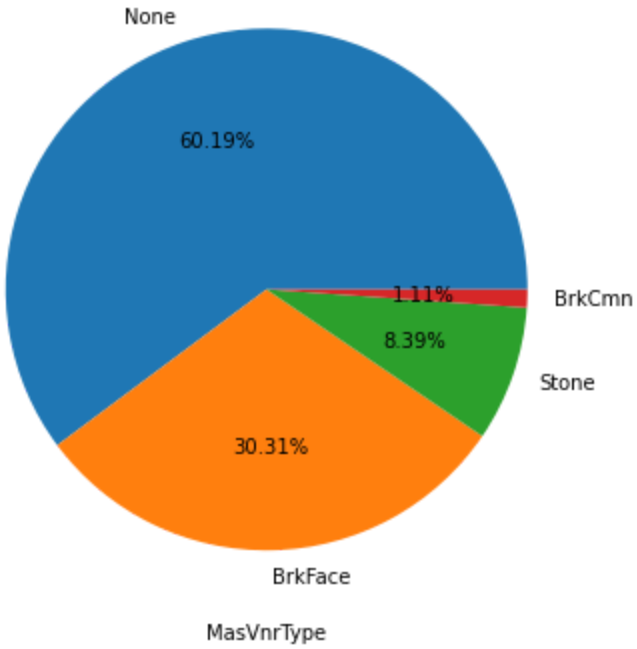
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

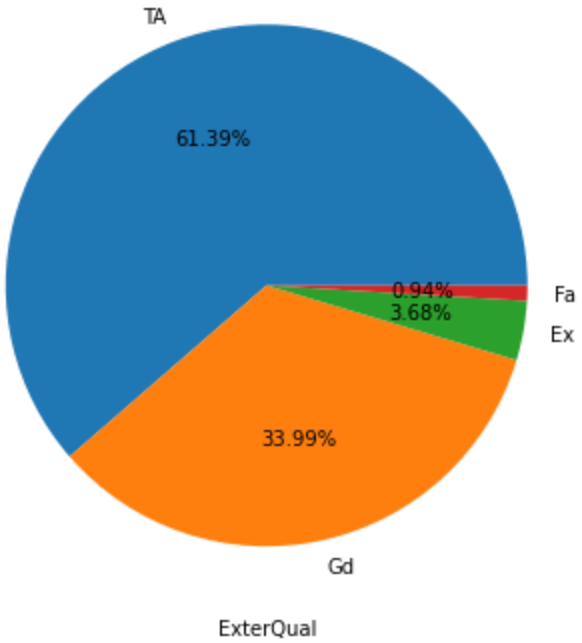


<Figure size 432x288 with 0 Axes>



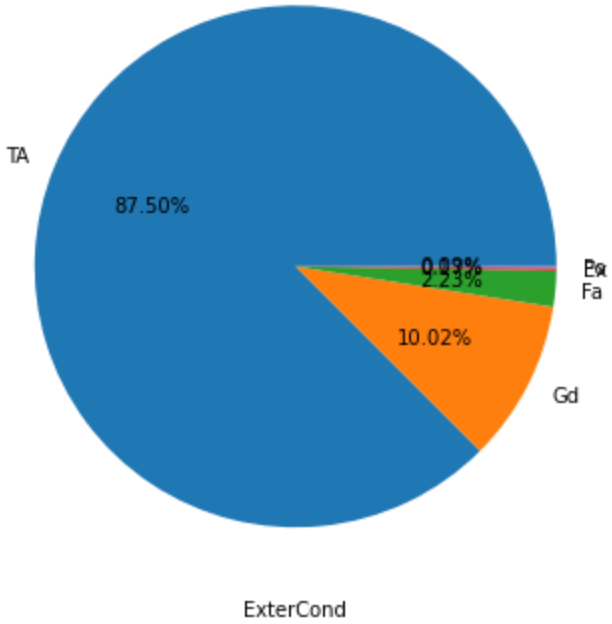
MasVnrType

<Figure size 432x288 with 0 Axes>

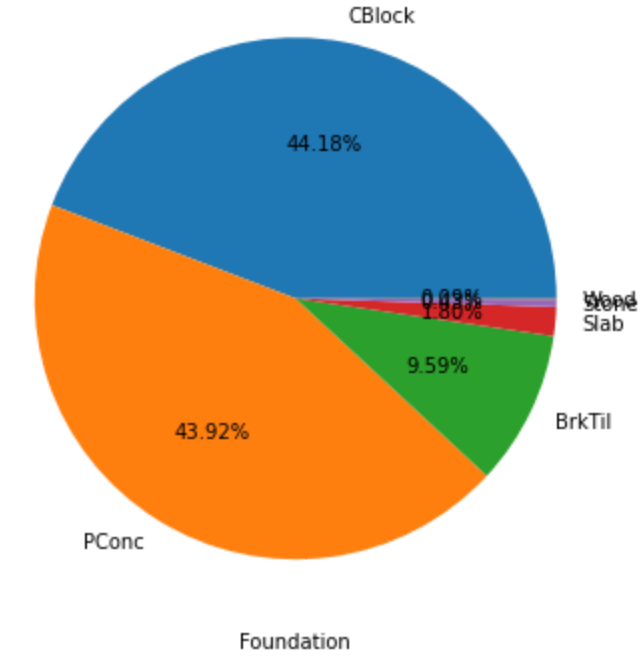


ExterQual

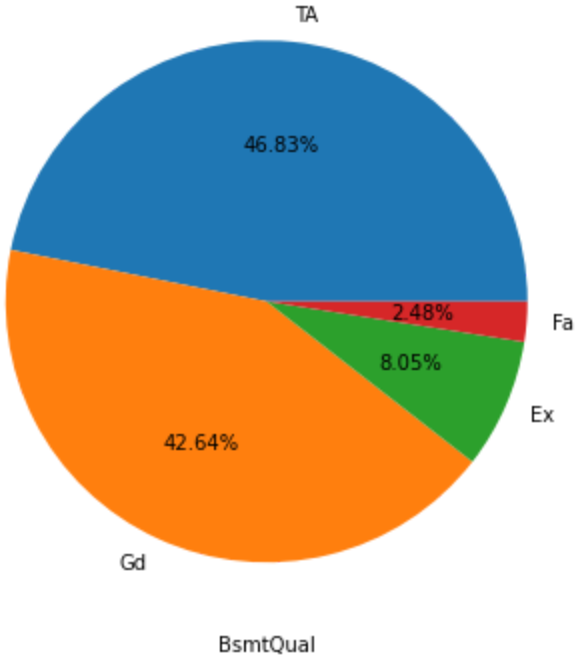
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

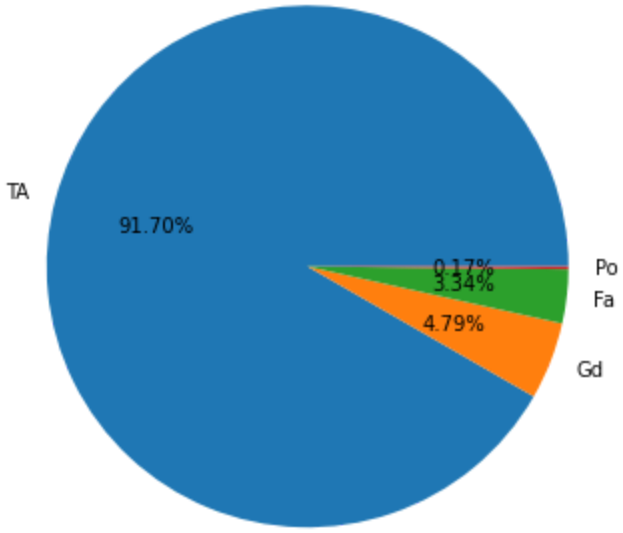


<Figure size 432x288 with 0 Axes>



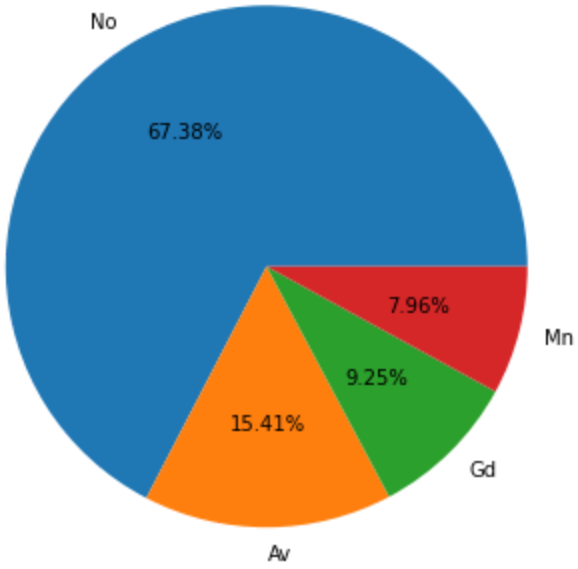
BsmtQual

<Figure size 432x288 with 0 Axes>



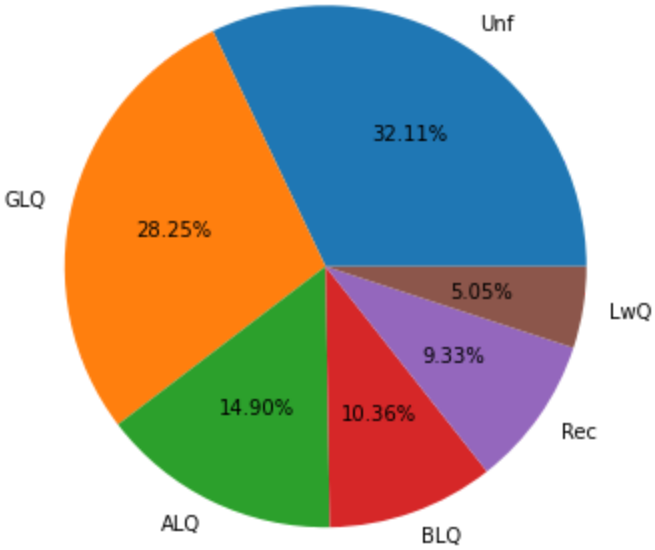
BsmtCond

<Figure size 432x288 with 0 Axes>



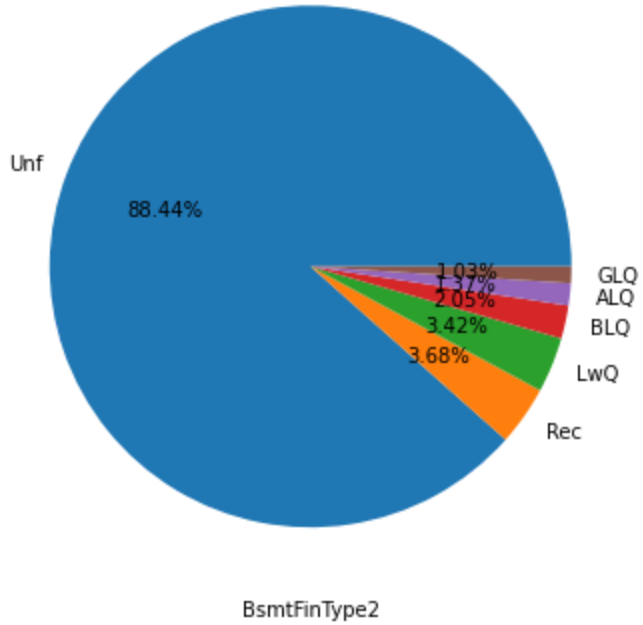
BsmtExposure

<Figure size 432x288 with 0 Axes>

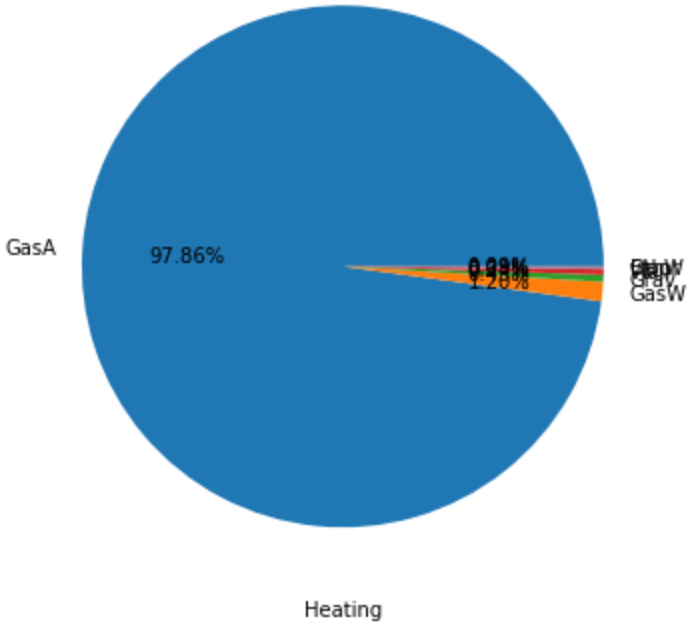


BsmtFinType1

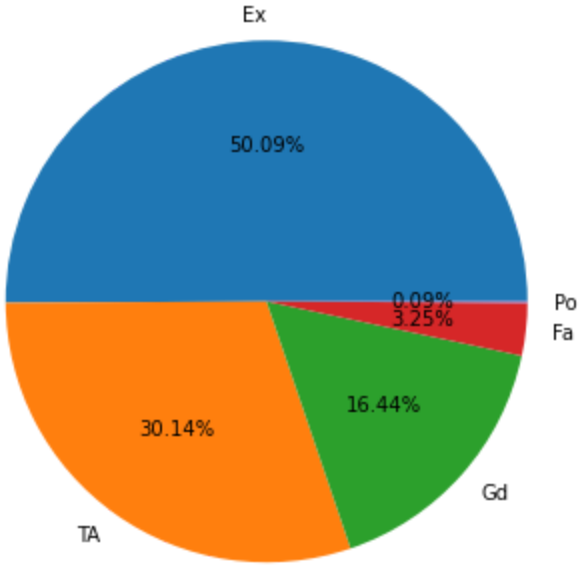
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

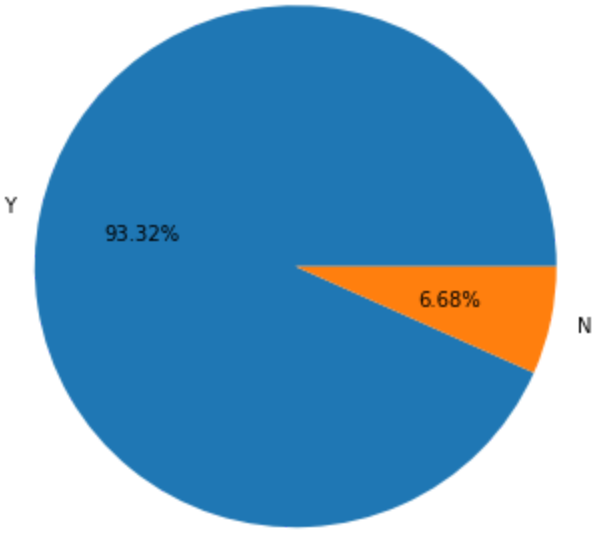


<Figure size 432x288 with 0 Axes>



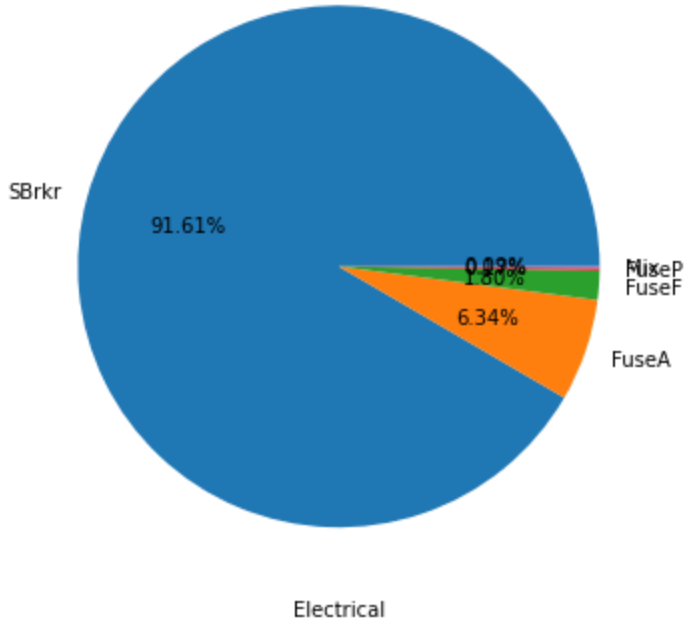
HeatingQC

<Figure size 432x288 with 0 Axes>

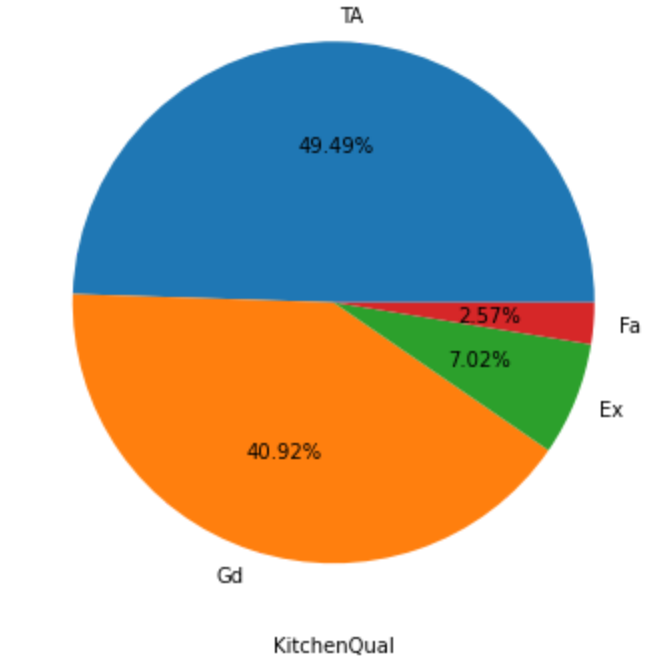


CentralAir

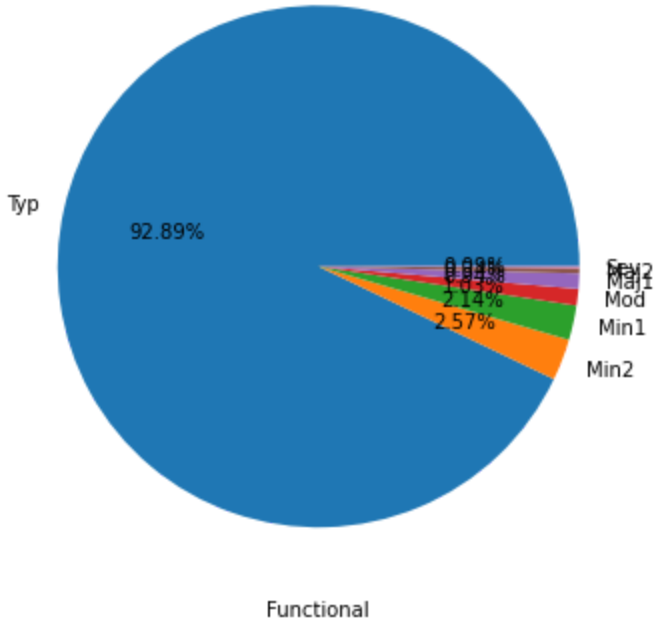
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>

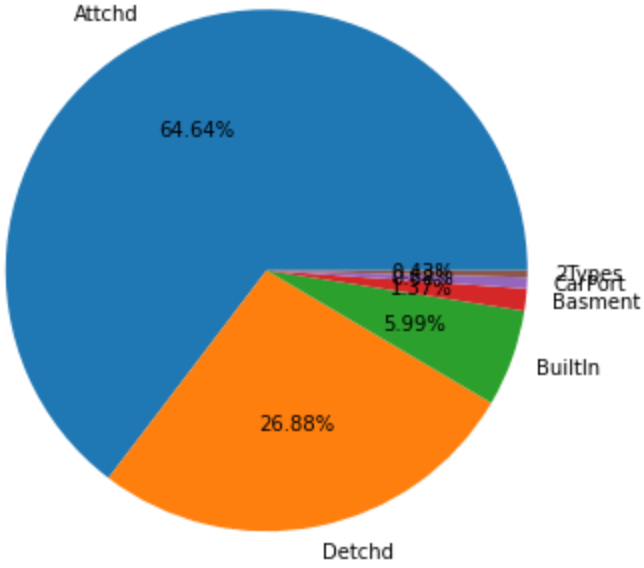


<Figure size 432x288 with 0 Axes>



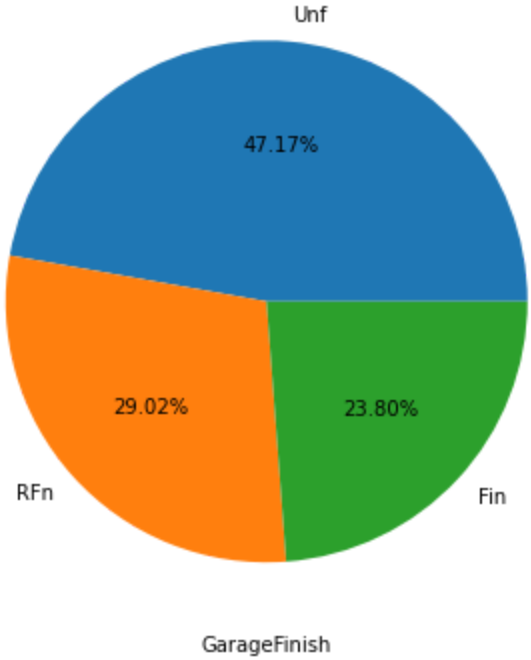
Functional

<Figure size 432x288 with 0 Axes>

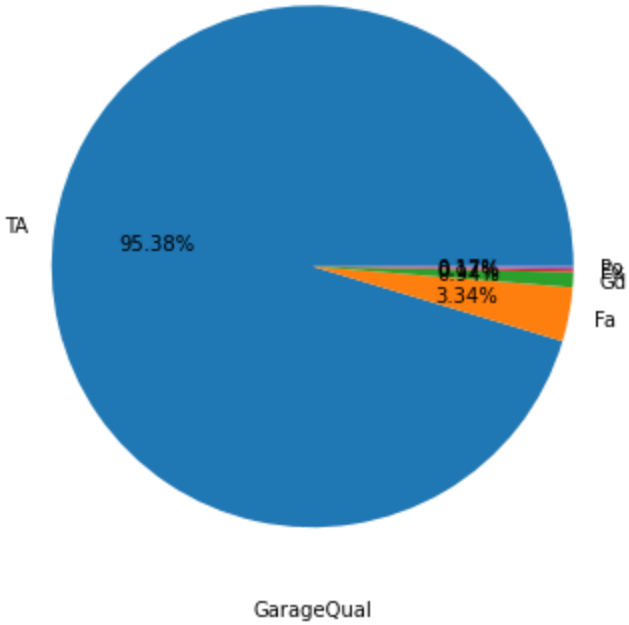


GarageType

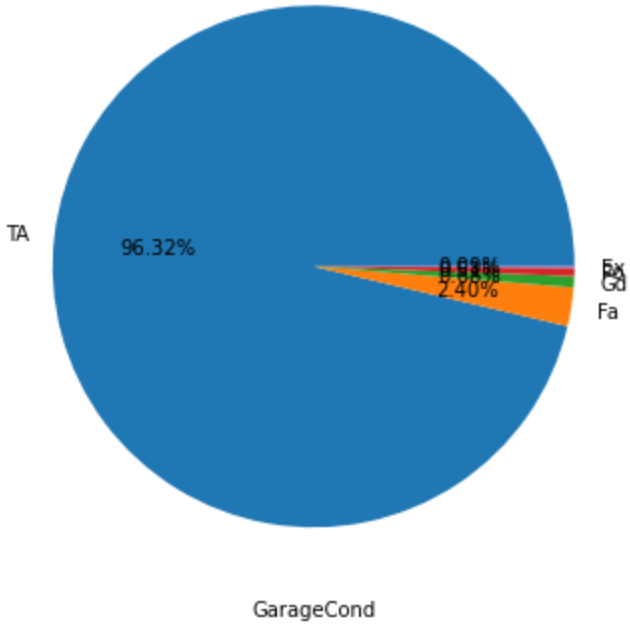
<Figure size 432x288 with 0 Axes>



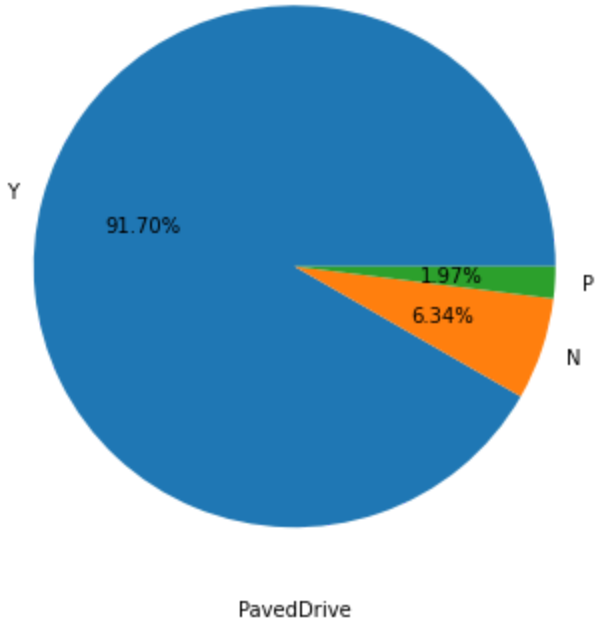
<Figure size 432x288 with 0 Axes>



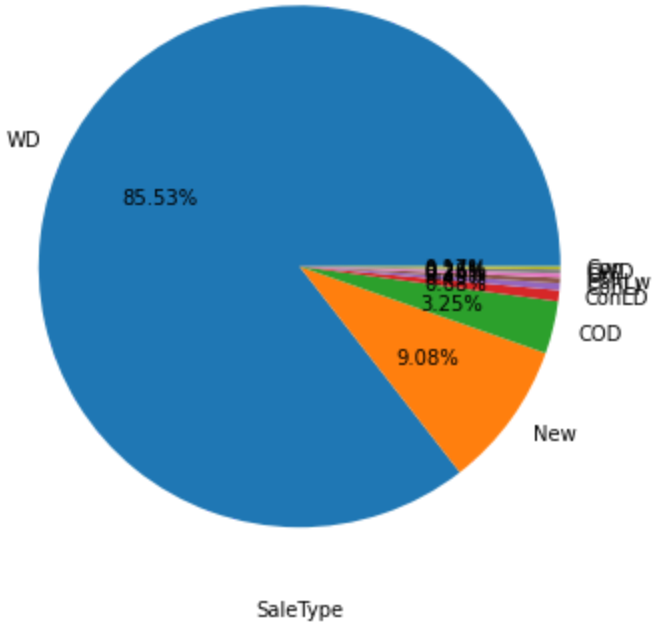
<Figure size 432x288 with 0 Axes>



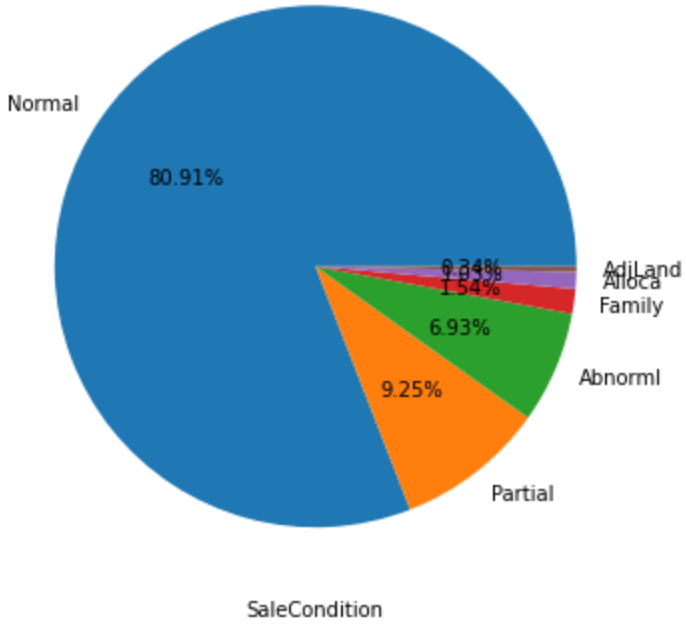
<Figure size 432x288 with 0 Axes>



<Figure size 432x288 with 0 Axes>




```
<Figure size 432x288 with 0 Axes>
```




```
<Figure size 432x288 with 0 Axes>
```

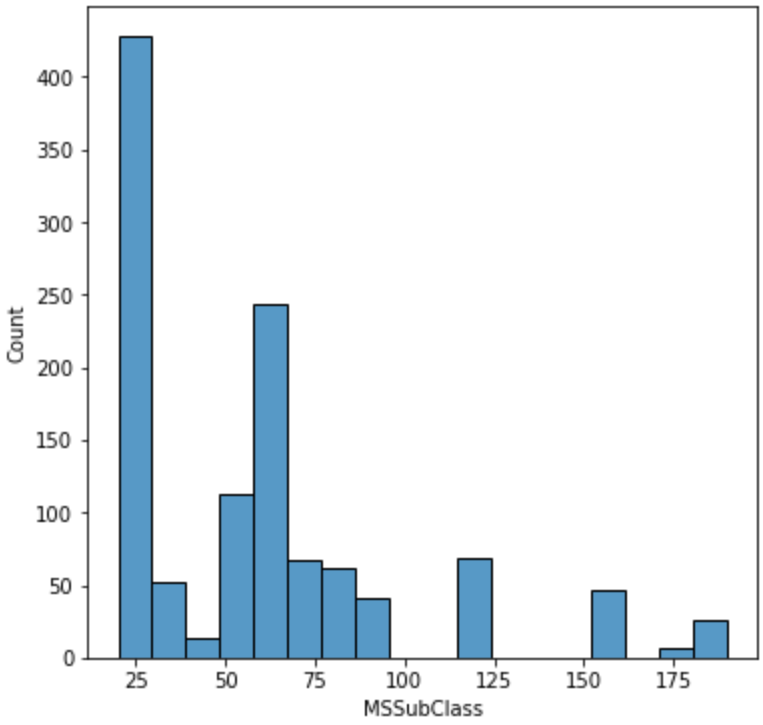
We have drop utilities column because is has only one value for all rows

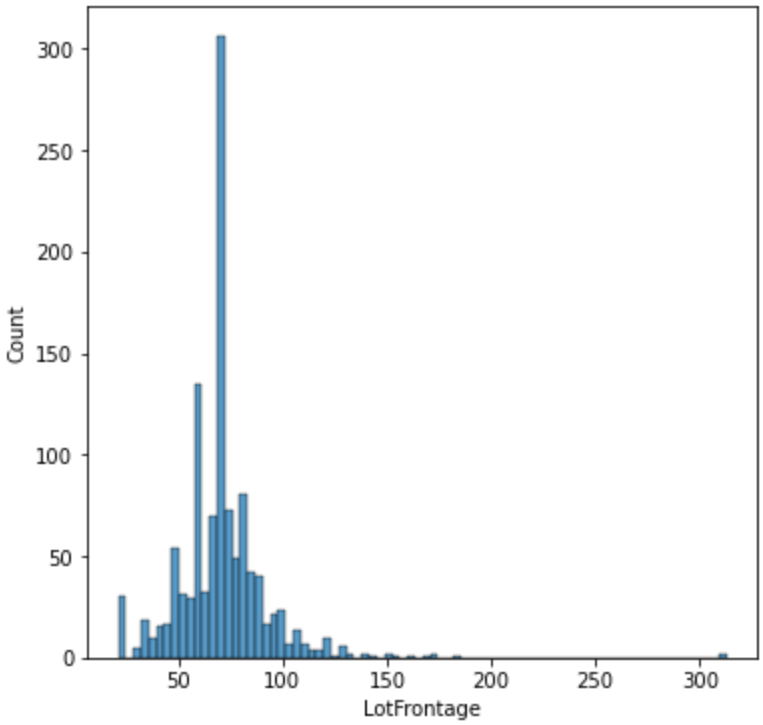
In [36]:
```
train.drop('Utilities', axis=1, inplace=True)
```

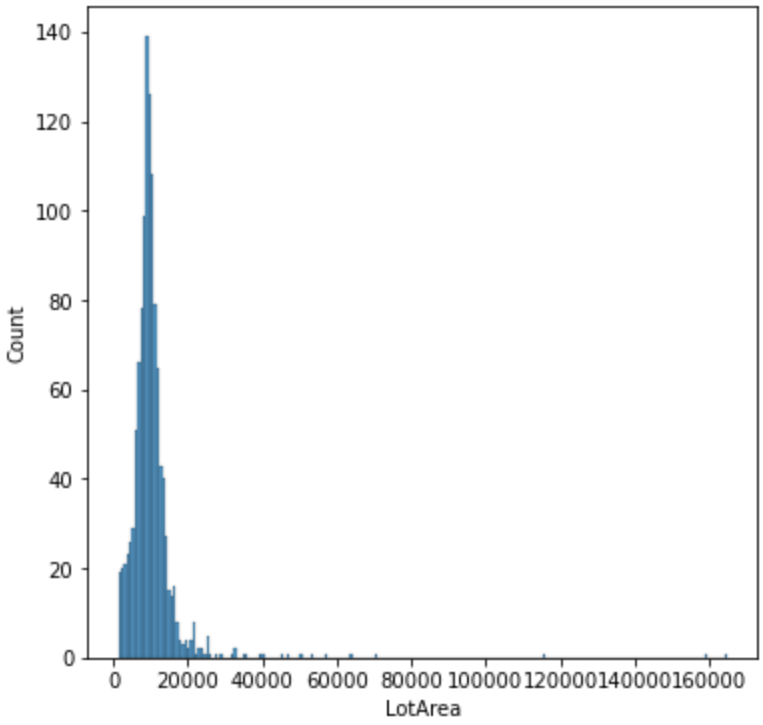
In [37]:
```
test.drop('Utilities', axis=1, inplace=True)
```

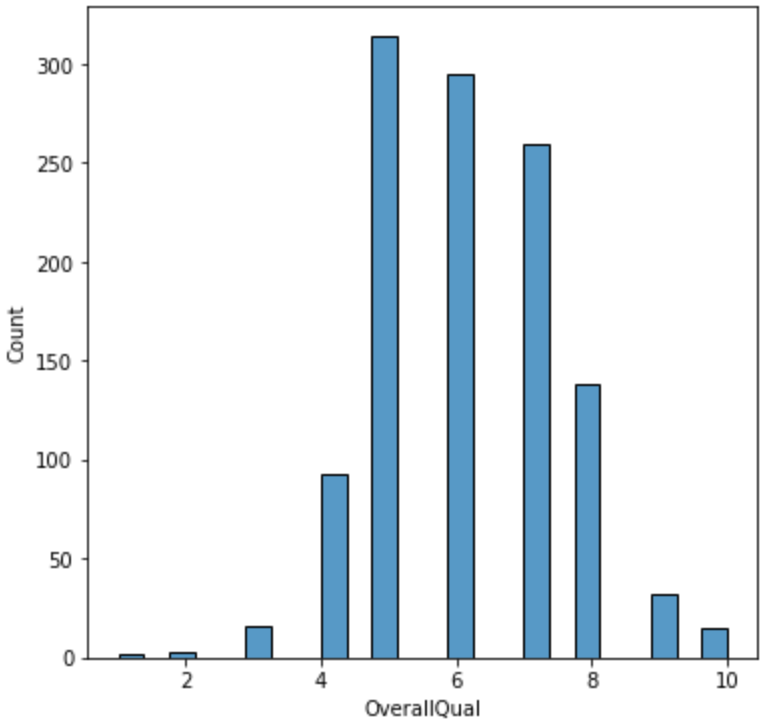
Ploting Histogram for numerical column

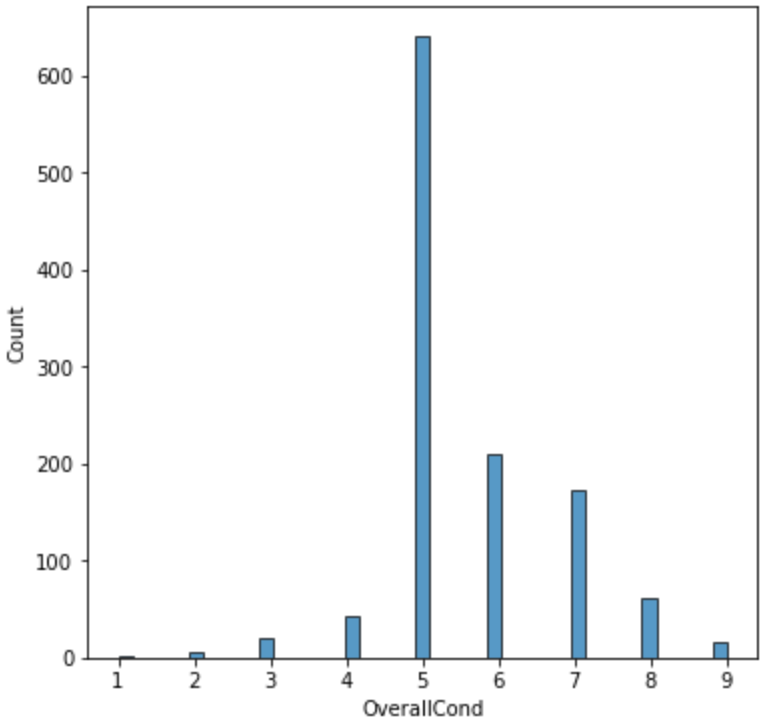
In [38]:
```
for i in train:
    if train[i].dtypes!='O':
        plt.figure(figsize=(6,6))
        sns.histplot(train[i])
```

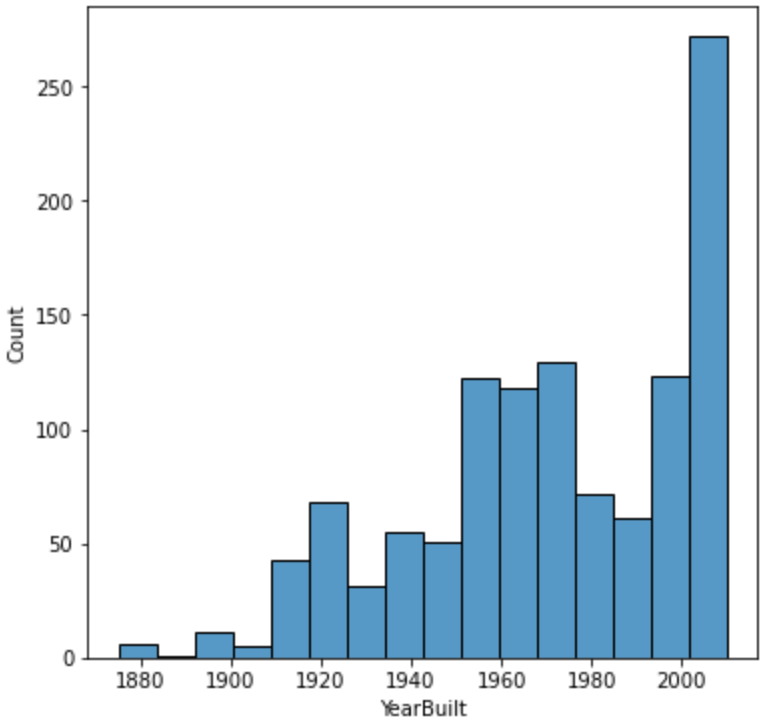

Count
MSSubClass
0
50
100
150
200
250
300
350
400
25
50
75
100
125
150
175

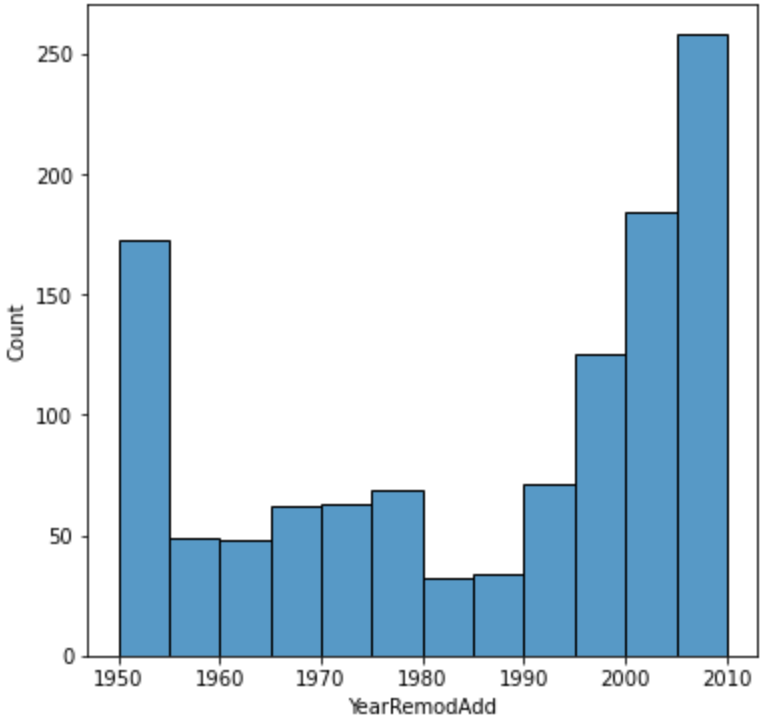

Count
LotFrontage
0
50
100
150
200
250
300
50
100
150
200
250
300

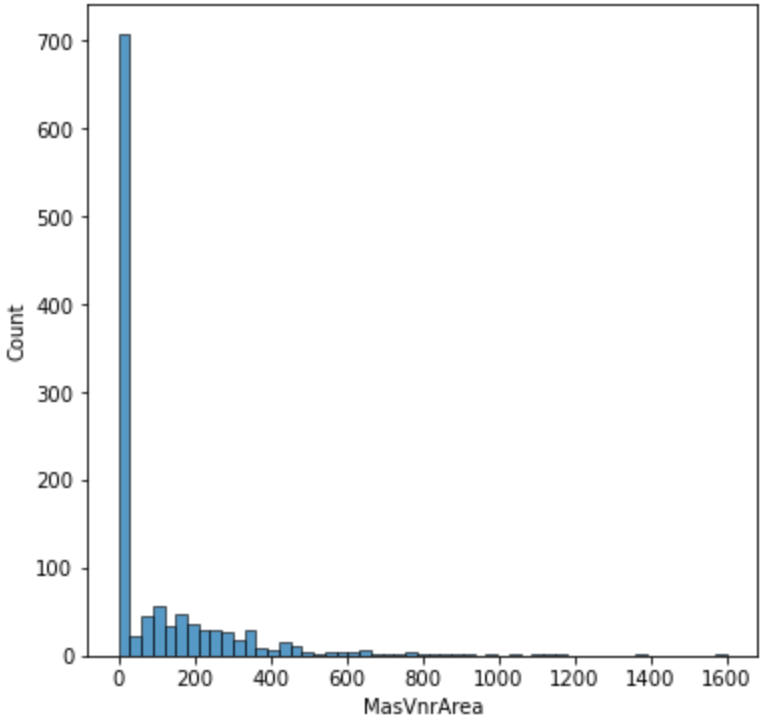

Count
LotArea
0
20
40
60
80
100
120
140
0
20000
40000
60000
80000
100000
120000
140000
160000

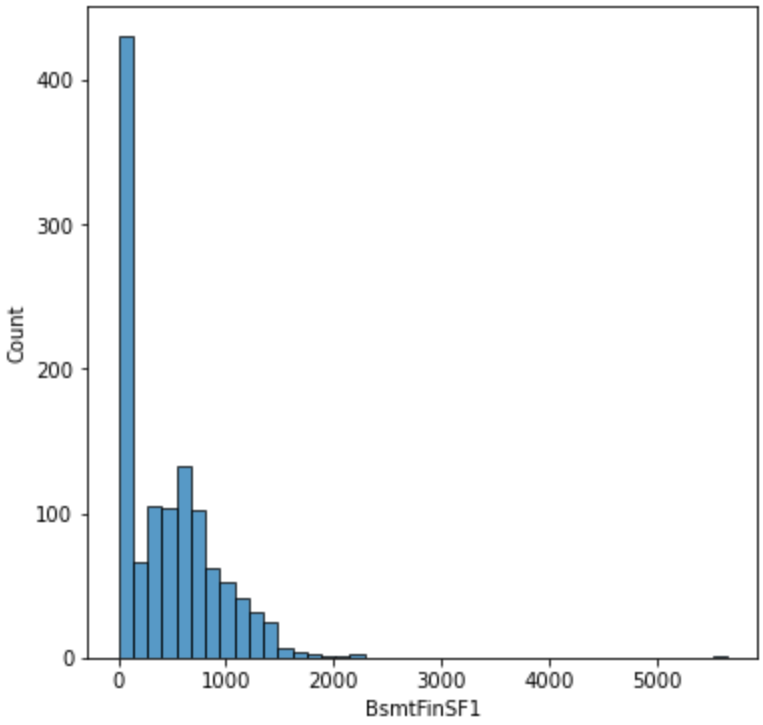

Count
OverallQual
0
50
100
150
200
250
300
2
4
6
8
10

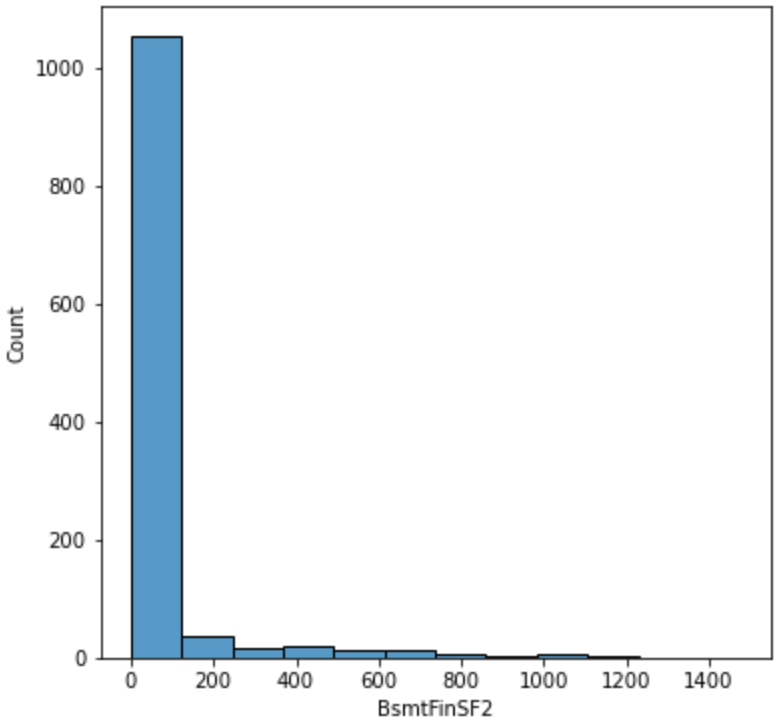

Count
600
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
OverallCond

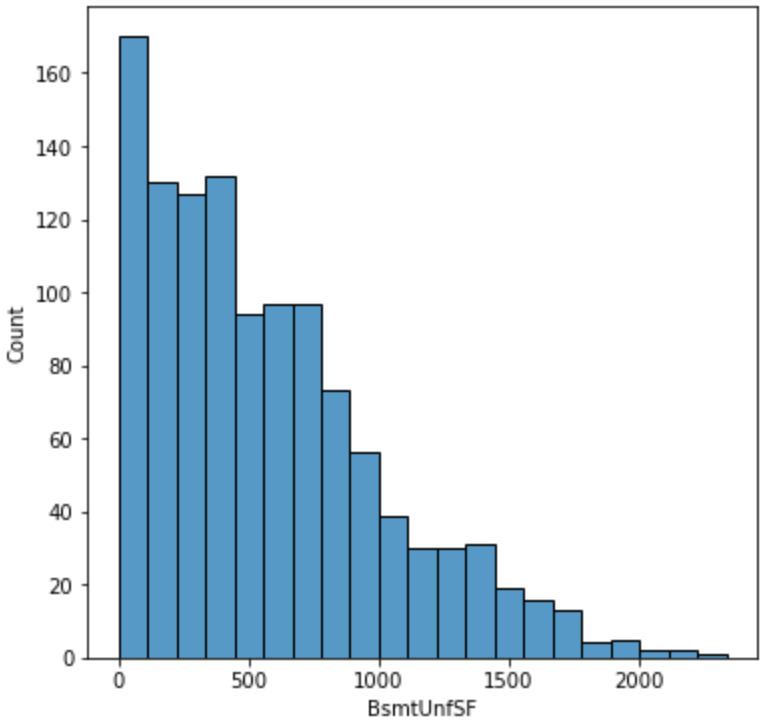

Count
250
200
150
100
50
0
1880
1900
1920
1940
1960
1980
2000
YearBuilt

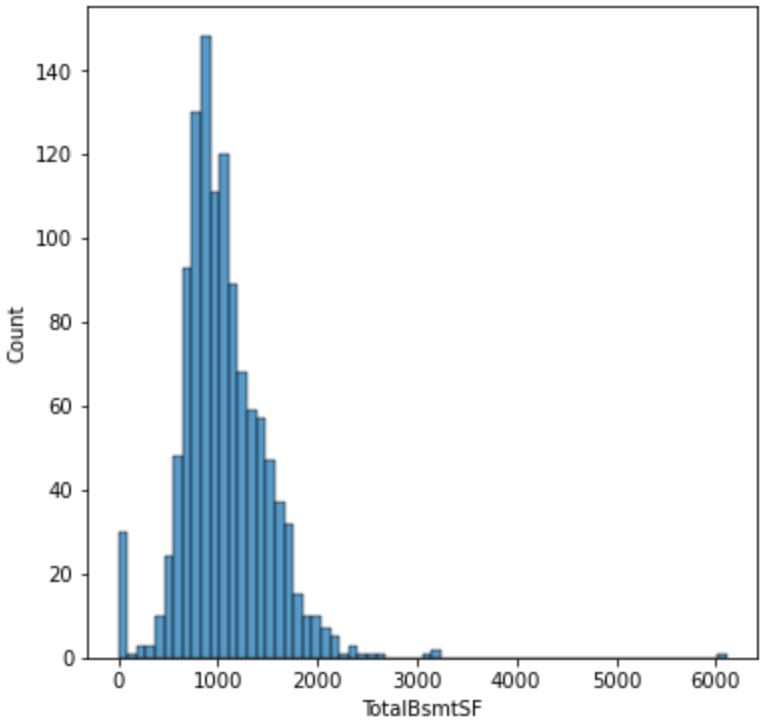

Count
YearRemodAdd
0
50
100
150
200
250
1950
1960
1970
1980
1990
2000
2010

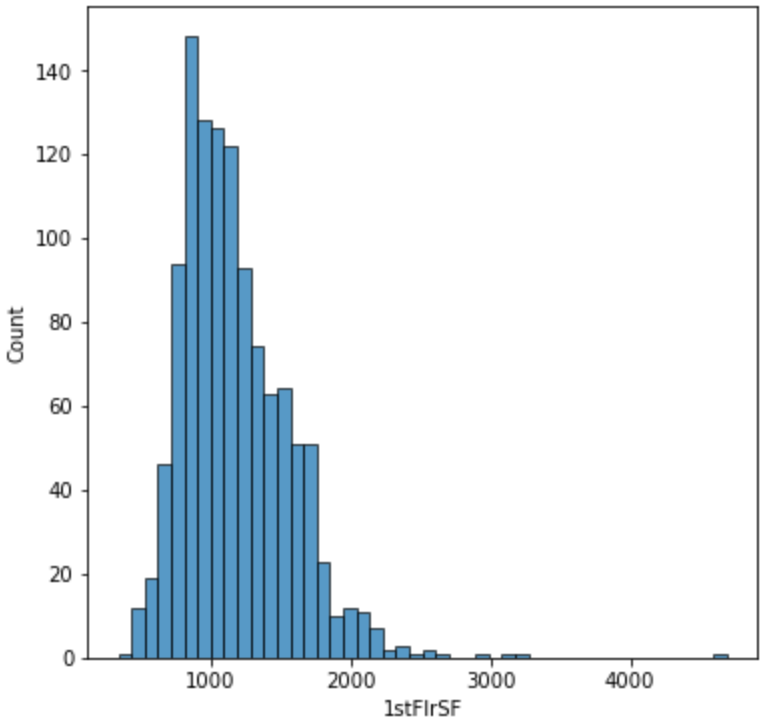

Count
MasVnrArea
0
100
200
300
400
500
600
700
0
200
400
600
800
1000
1200
1400
1600

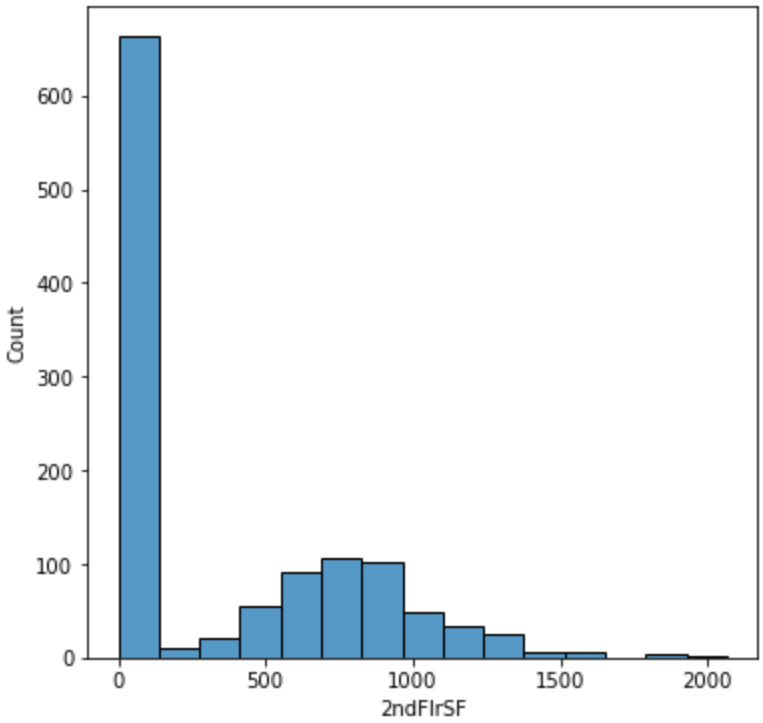

Count
400
300
200
100
0
0
1000
2000
3000
4000
5000
BsmtFinSF1

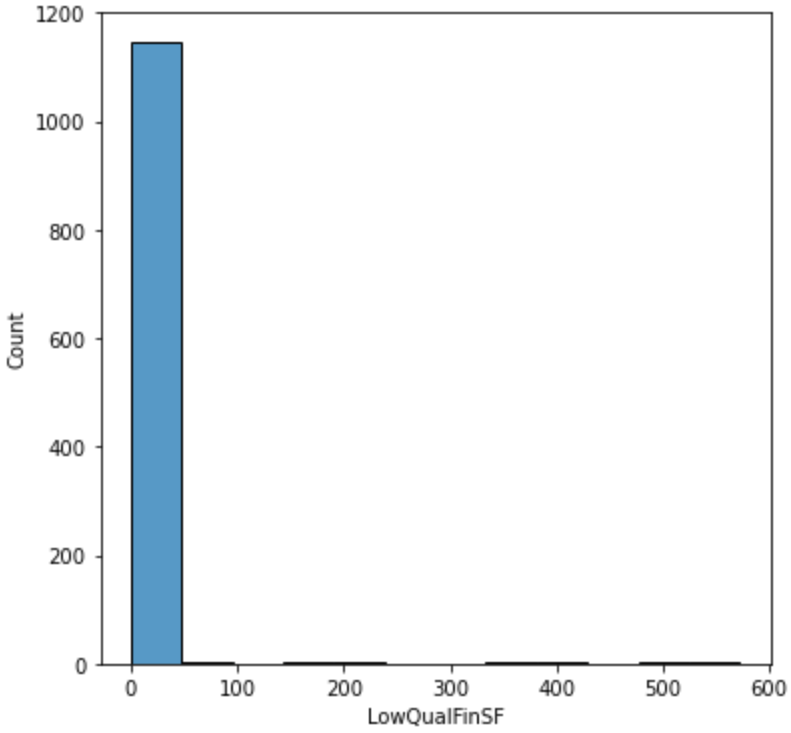

Count
1000
800
600
400
200
0
0
200
400
600
800
1000
1200
1400
BsmtFinSF2

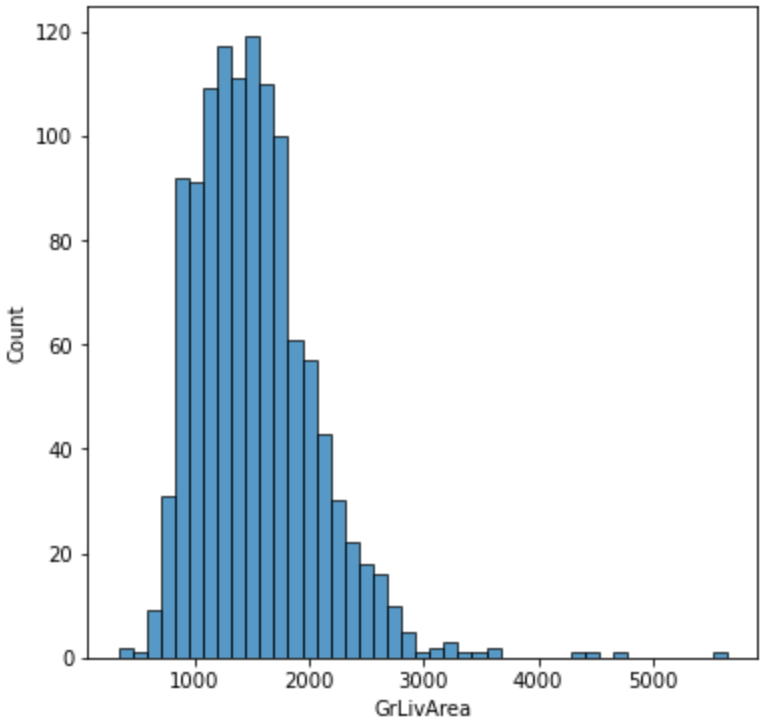

Count
BsmtUnfSF
0
20
40
60
80
100
120
140
160
0
500
1000
1500
2000

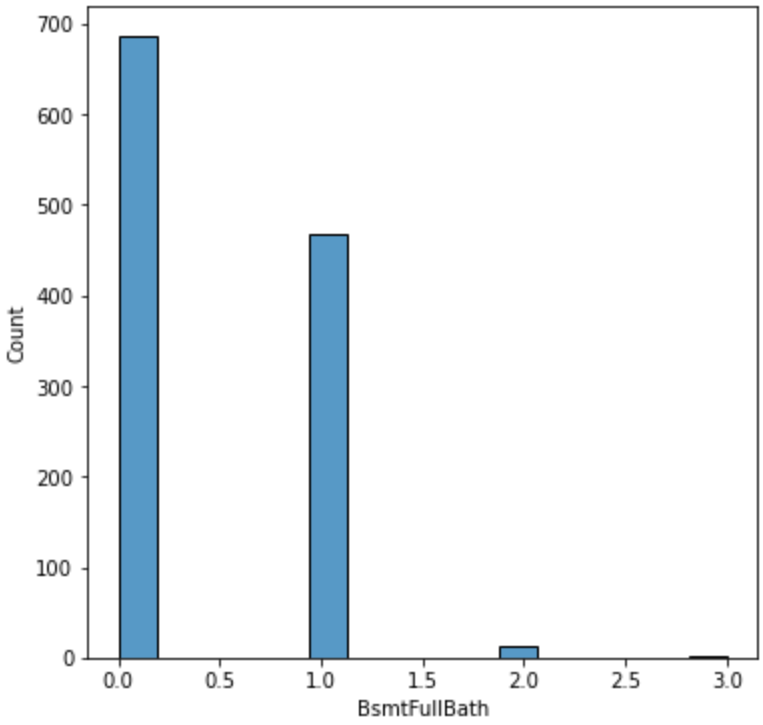

Count
TotalBsmtSF
0
20
40
60
80
100
120
140
0
1000
2000
3000
4000
5000
6000

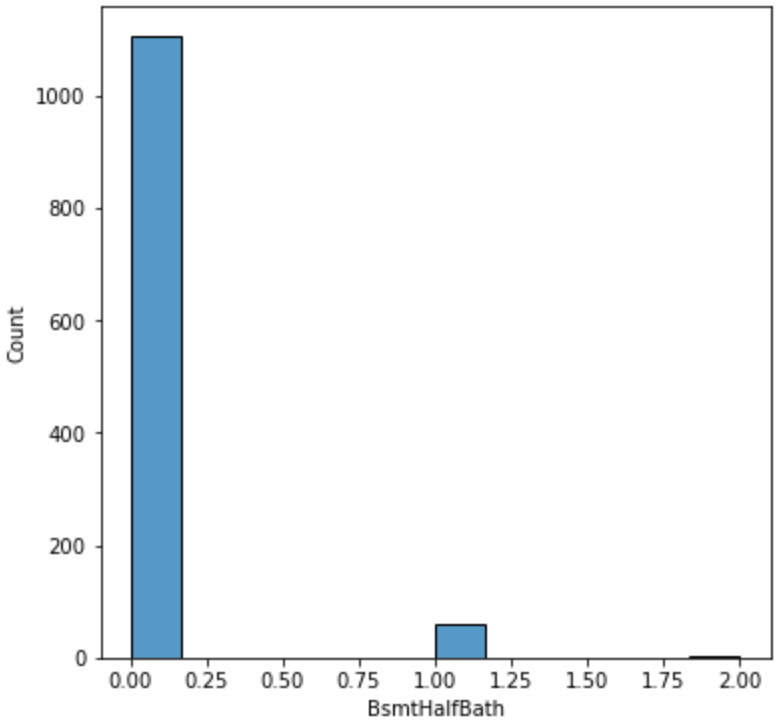

Count
1stFlrSF
0
20
40
60
80
100
120
140
1000
2000
3000
4000

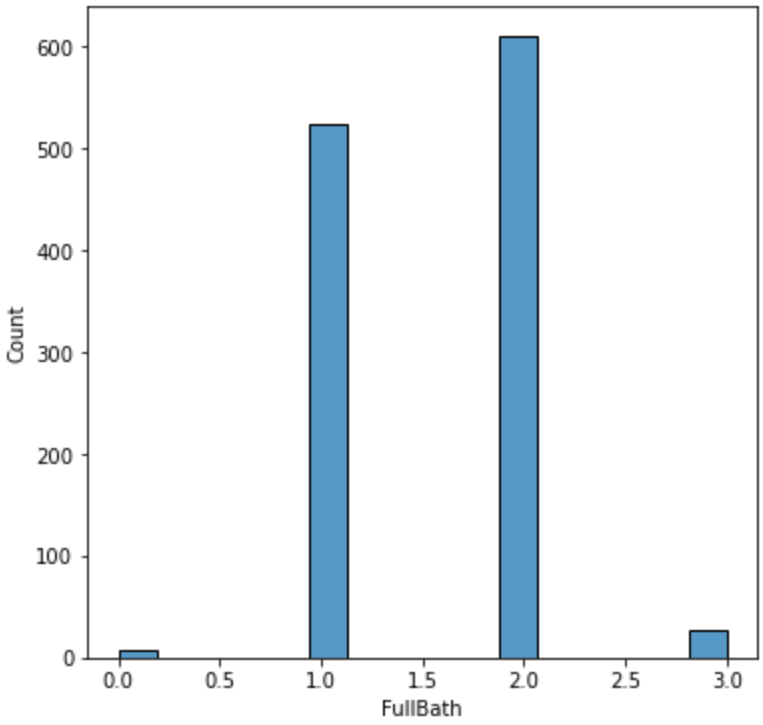

Count
2ndFlrSF
0
100
200
300
400
500
600
0
500
1000
1500
2000

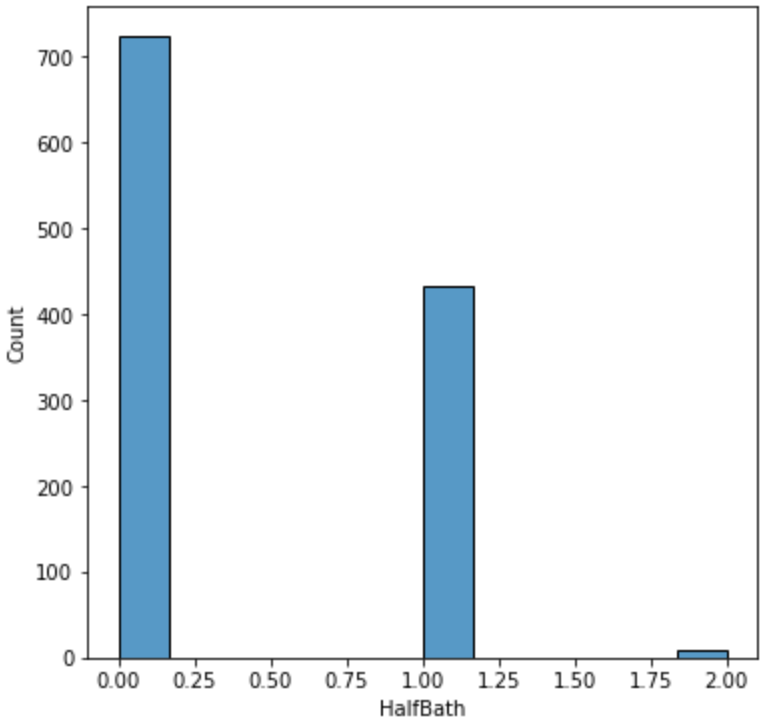

1200
1000
800
600
400
200
0
Count
0
100
200
300
400
500
600
LowQualFinSF

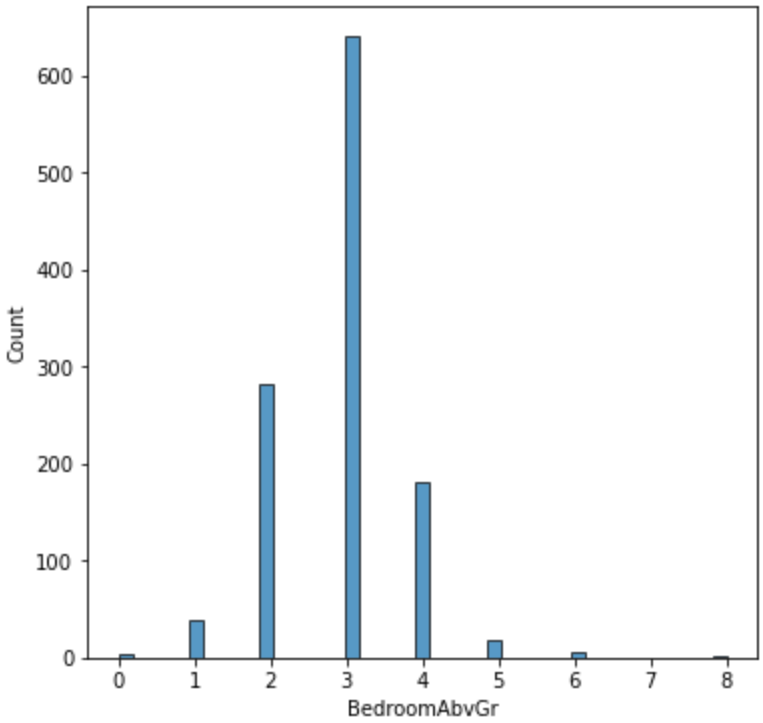

120
100
80
60
40
20
0
Count
1000
2000
3000
4000
5000
GrLivArea

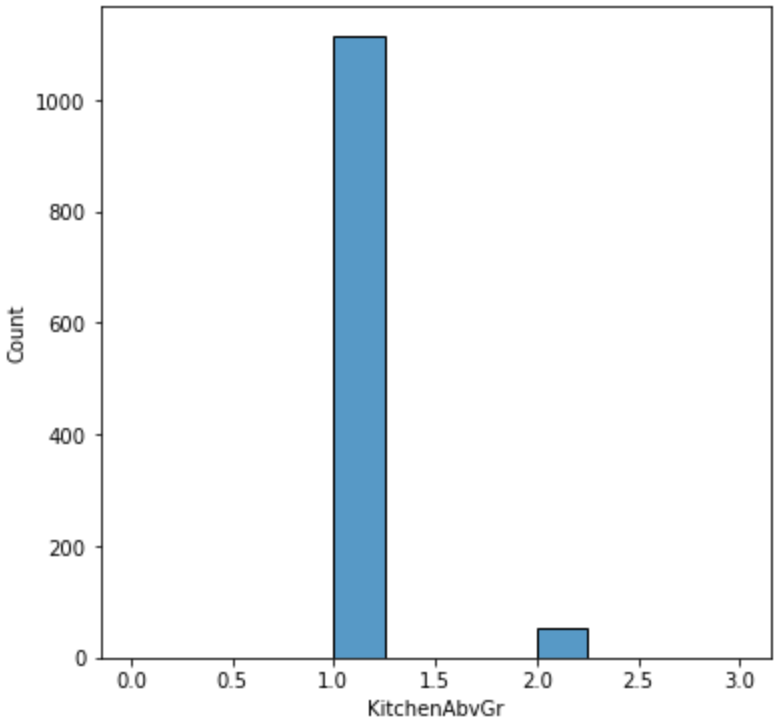

Count
BsmtFullBath
0
100
200
300
400
500
600
700
0.0
0.5
1.0
1.5
2.0
2.5
3.0

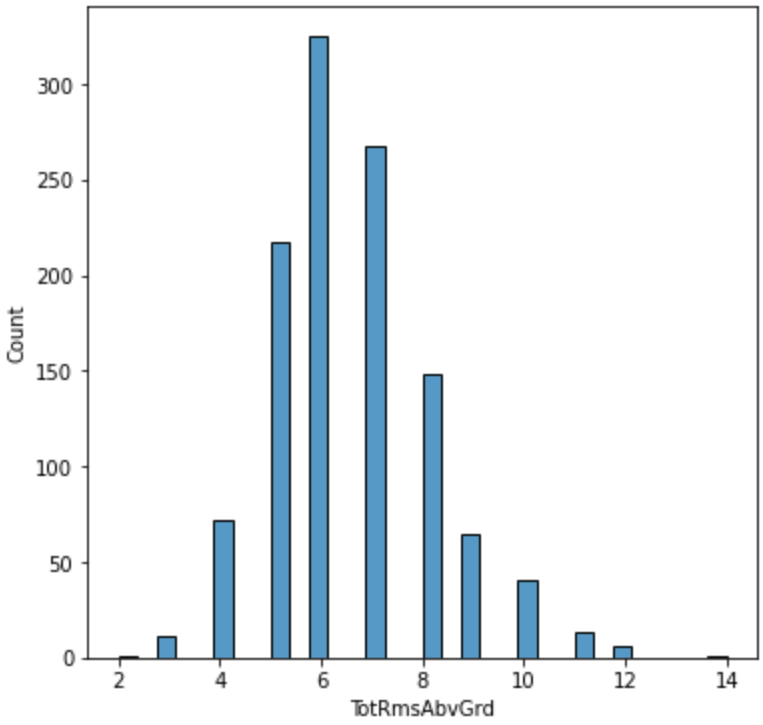

Count
BsmtHalfBath
0
200
400
600
800
1000
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

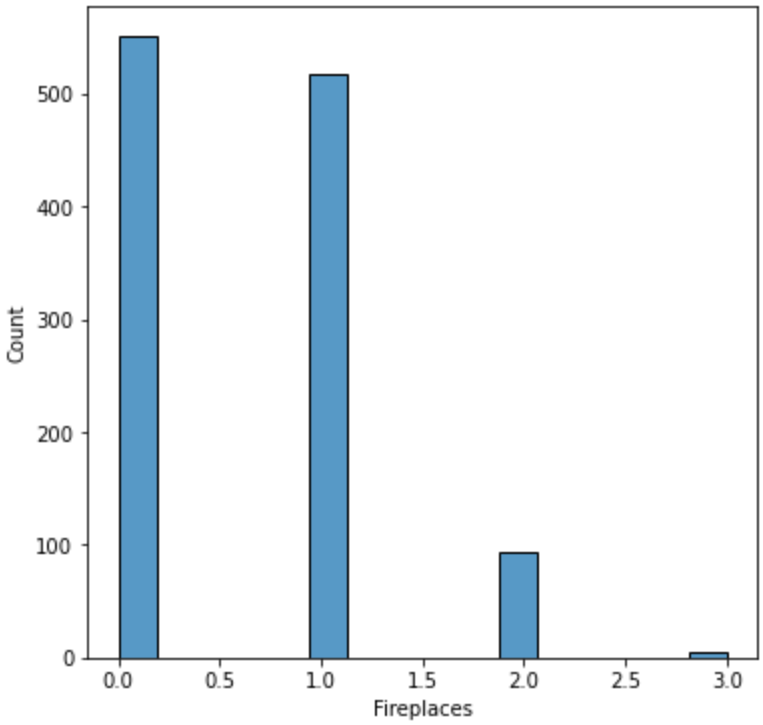

Count
FullBath
0
100
200
300
400
500
600
0.0
0.5
1.0
1.5
2.0
2.5
3.0

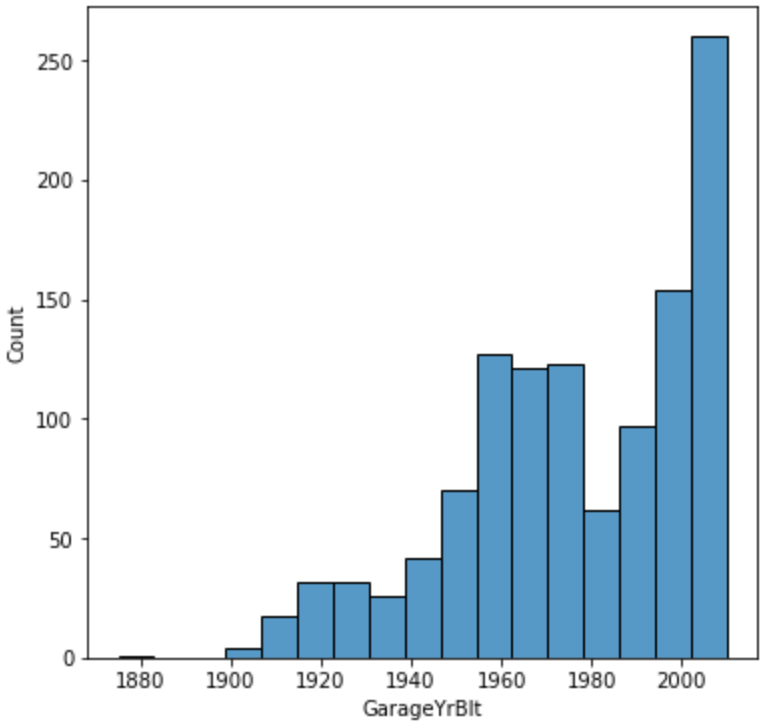

Count
HalfBath
0
100
200
300
400
500
600
700
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00

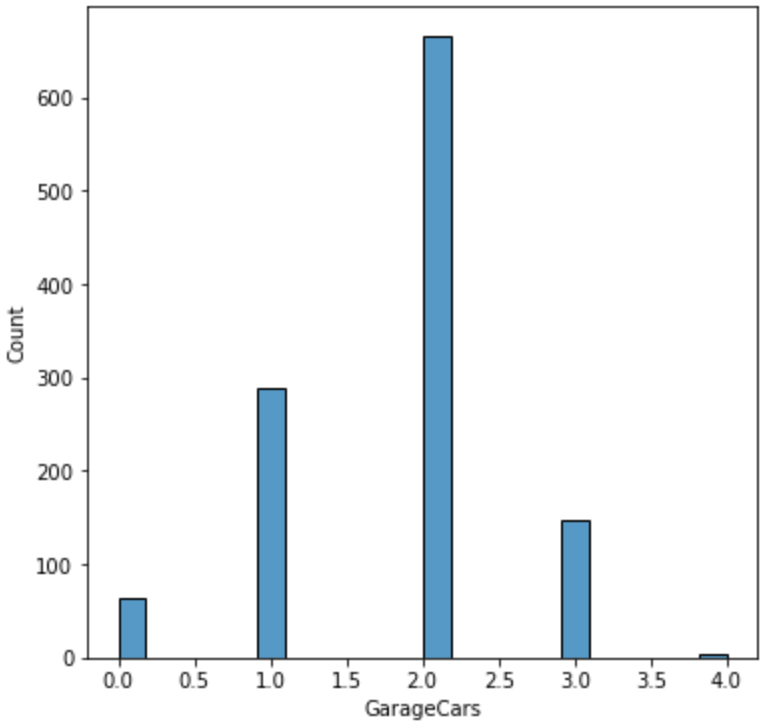

Count
BedroomAbvGr
0
100
200
300
400
500
600
0
1
2
3
4
5
6
7
8

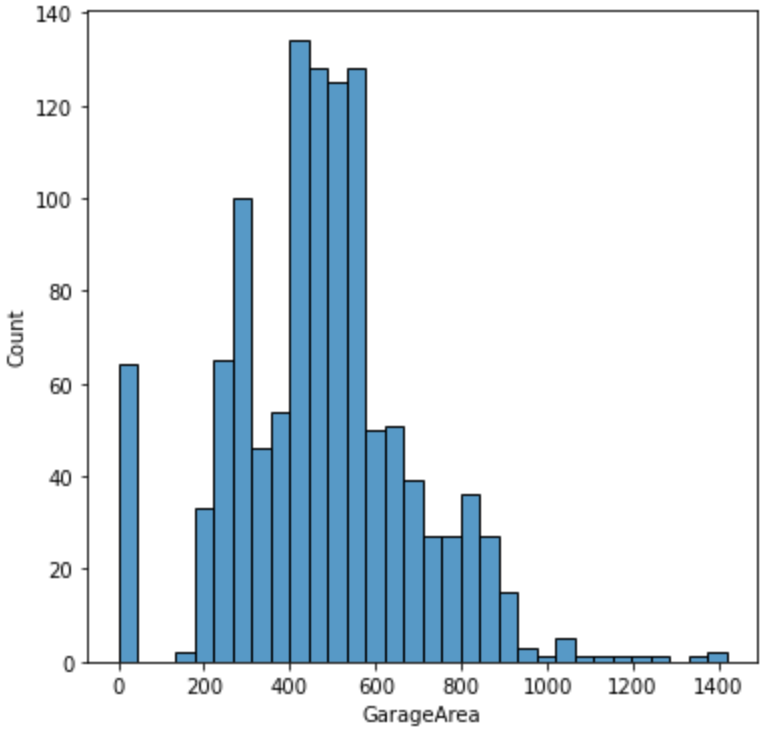

Count
KitchenAbvGr
0
200
400
600
800
1000
0.0
0.5
1.0
1.5
2.0
2.5
3.0

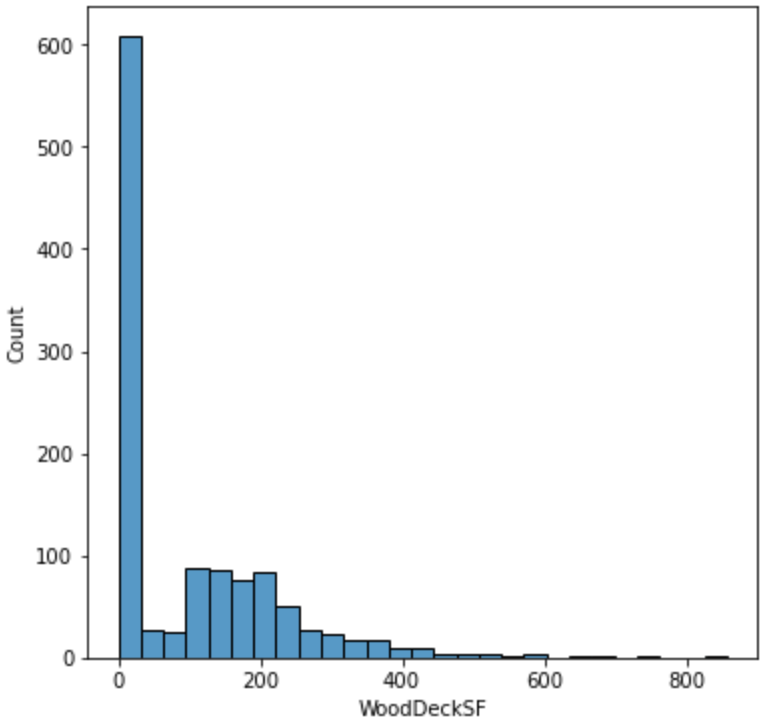

Count
300
250
200
150
100
50
0
2
4
6
8
10
12
14
TotRmsAbvGrd

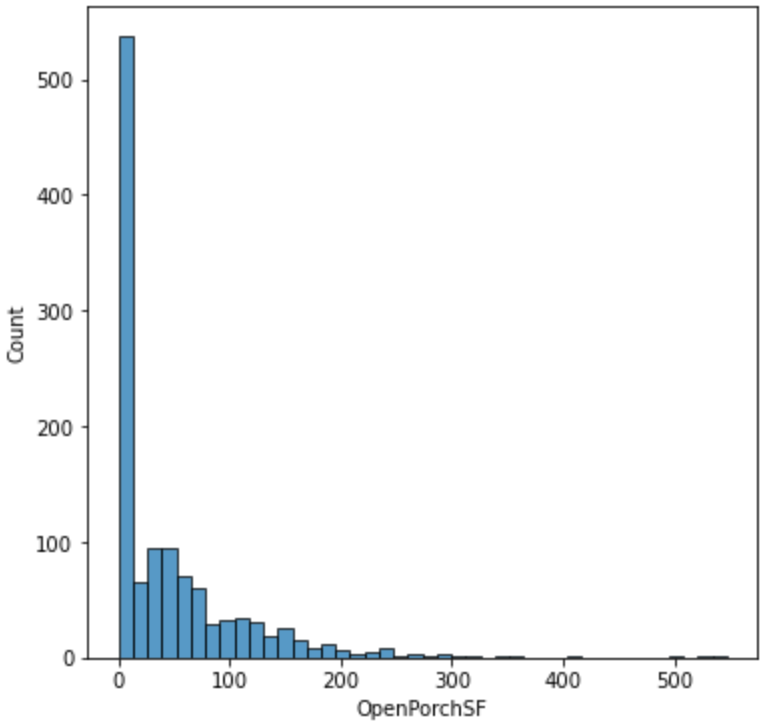

Count
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Fireplaces

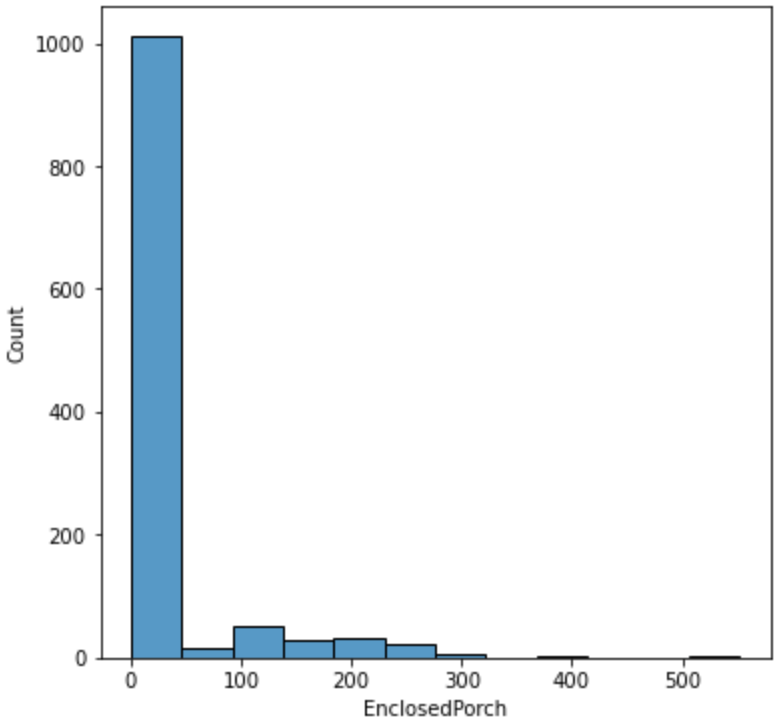

Count
250
200
150
100
50
0
1880
1900
1920
1940
1960
1980
2000
GarageYrBlt

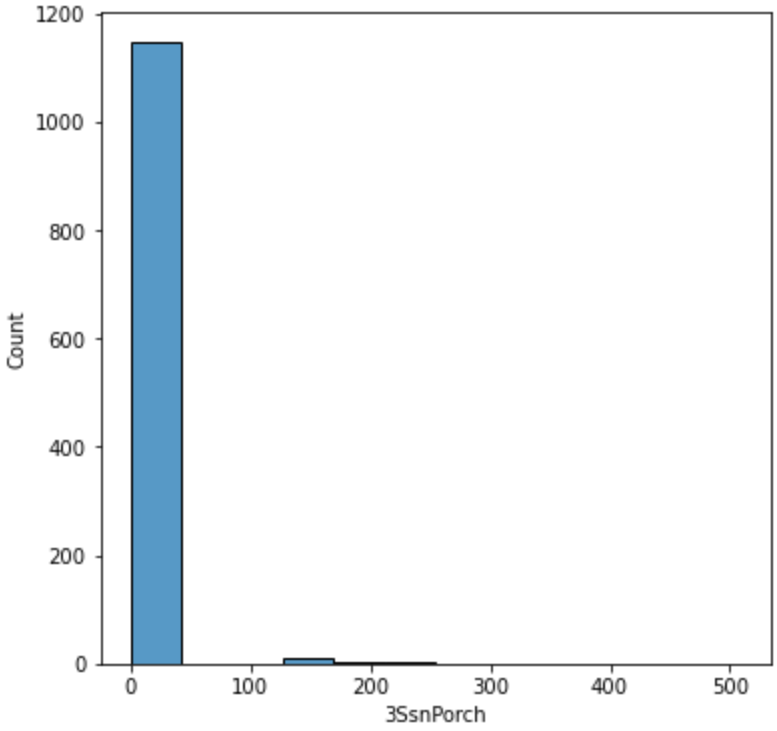

Count
600
500
400
300
200
100
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageCars

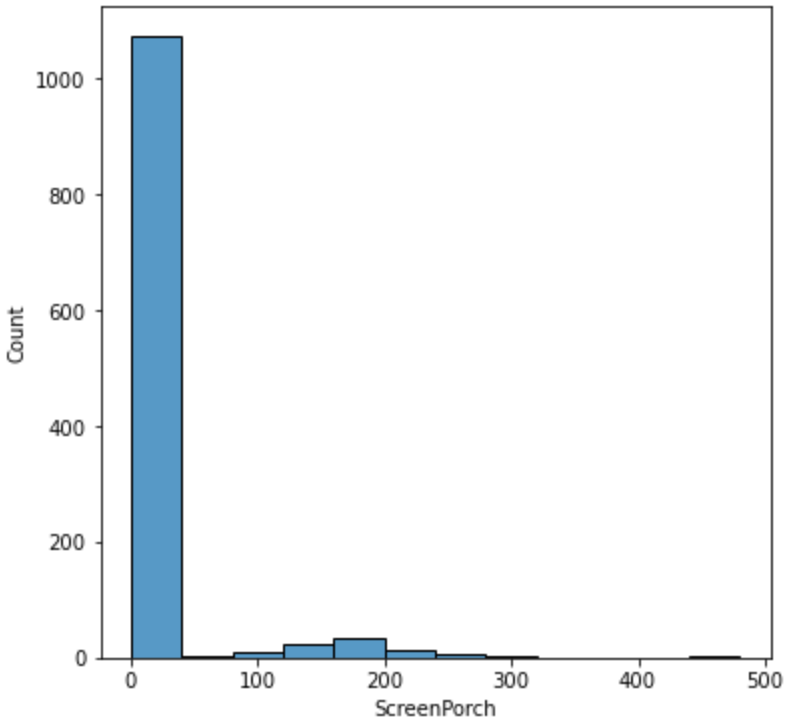

Count
GarageArea
0
20
40
60
80
100
120
140
0
200
400
600
800
1000
1200
1400

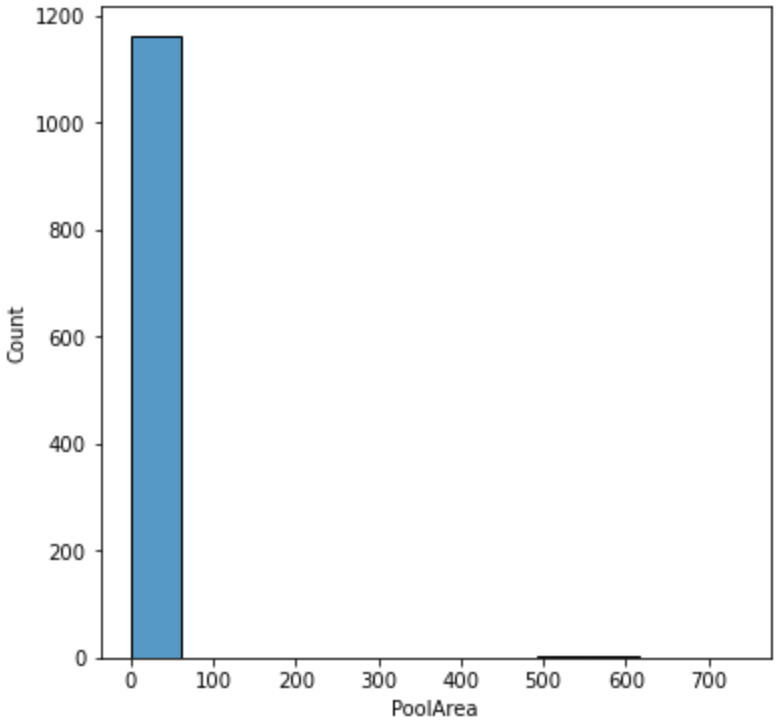

Count
WoodDeckSF
0
100
200
300
400
500
600
0
200
400
600
800

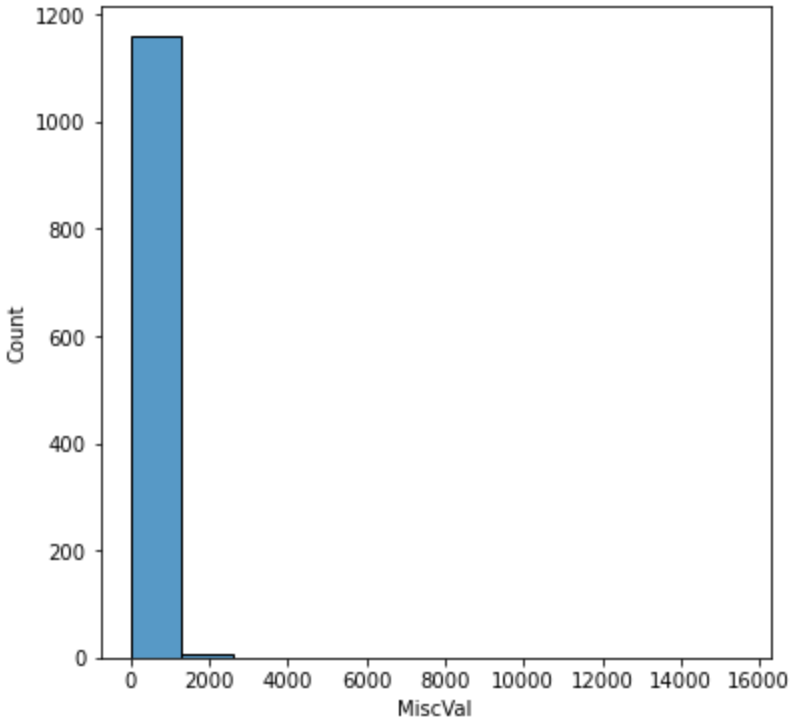

Count
OpenPorchSF
0
100
200
300
400
500

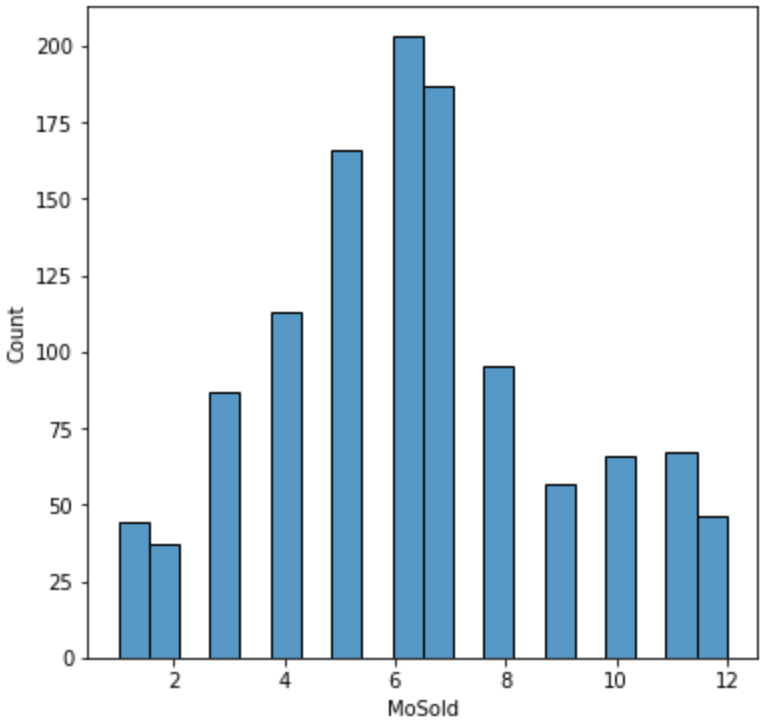

Count
EnclosedPorch
0
200
400
600
800
1000
100
300
500

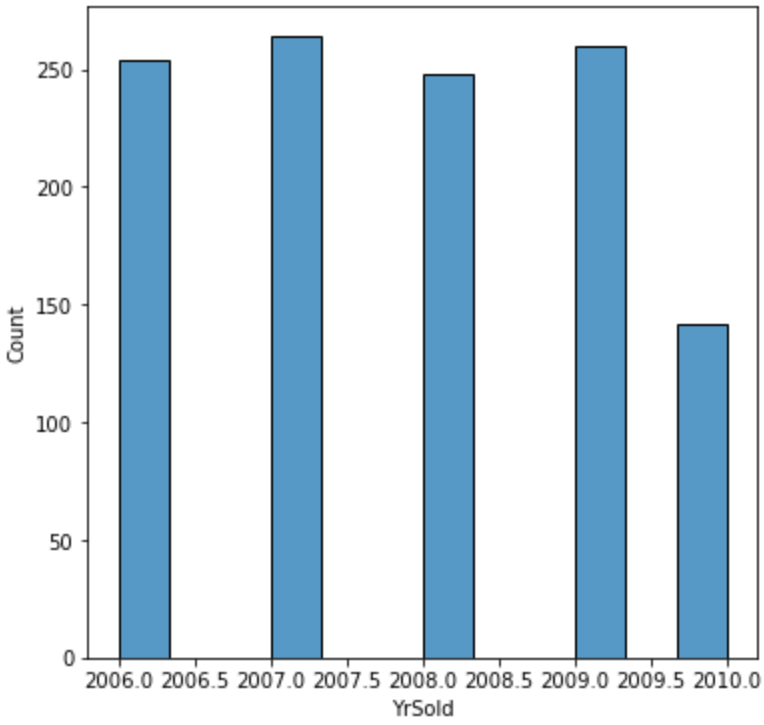

1200
1000
800
600
400
200
0
Count
0
100
200
300
400
500
3SsnPorch

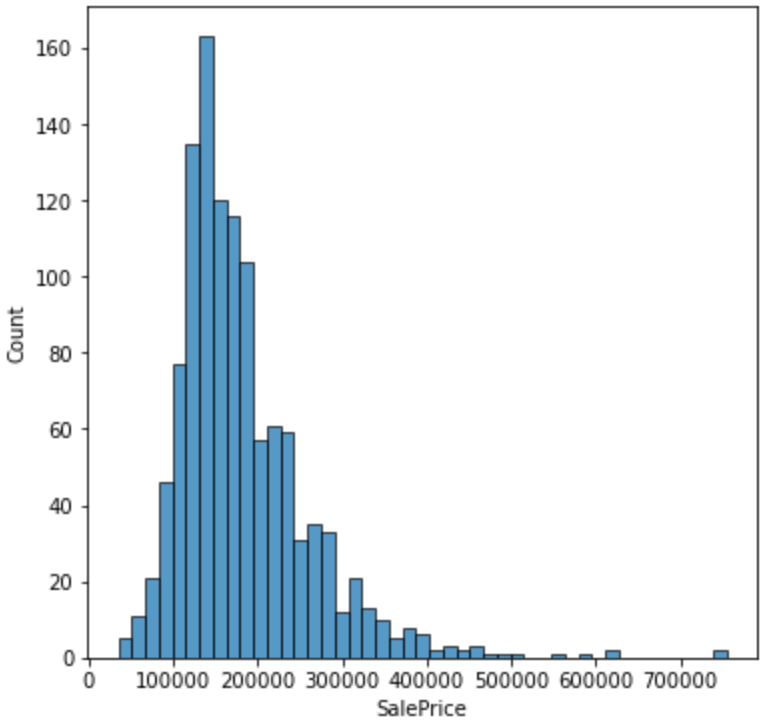

1000
800
600
400
200
0
Count
0
100
200
300
400
500
ScreenPorch


1200
1000
800
600
400
200
0
Count
0
100
200
300
400
500
600
700
PoolArea


1200
1000
800
600
400
200
0
Count
0
2000
4000
6000
8000
10000
12000
14000
16000
MiscVal


Count
MoSold
0
25
50
75
100
125
150
175
200
2
4
6
8
10
12

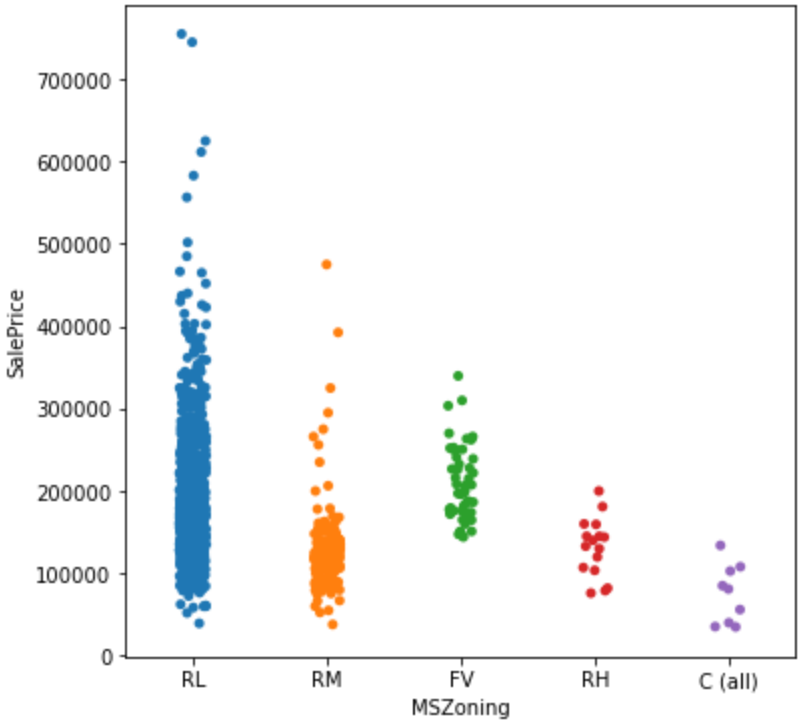

Count
YrSold
0
50
100
150
200
250
2006.0
2006.5
2007.0
2007.5
2008.0
2008.5
2009.0
2009.5
2010.0

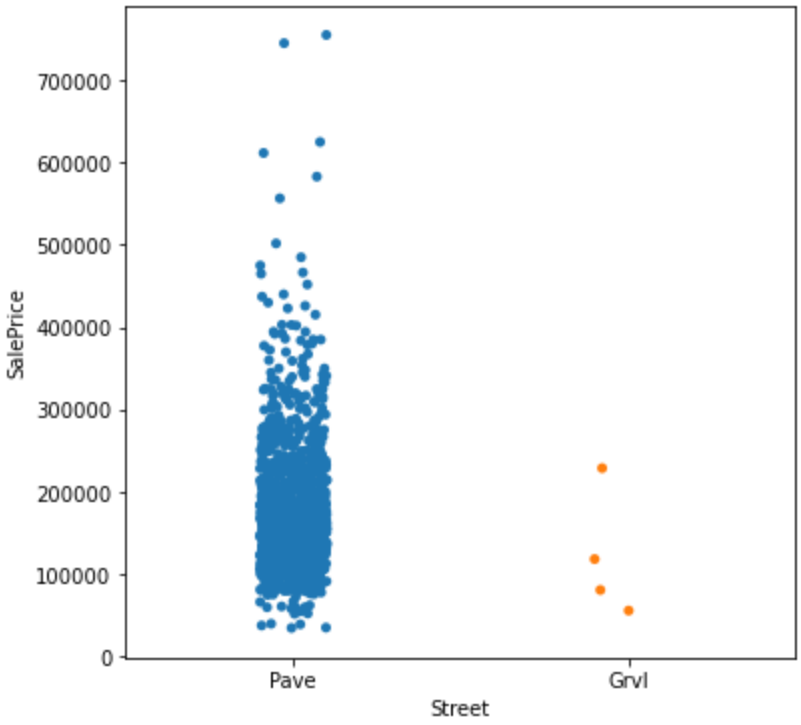


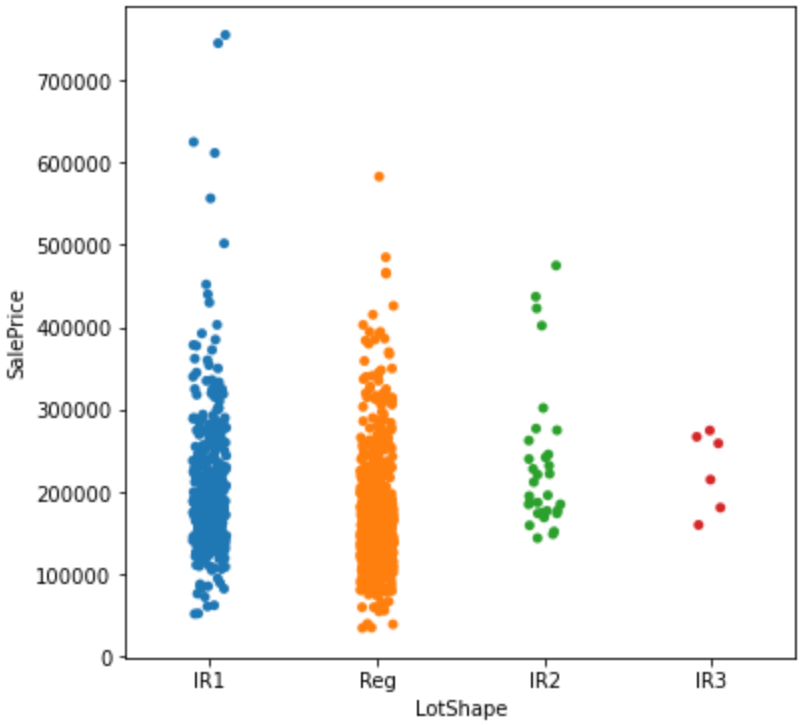
Bi-variate Analysis

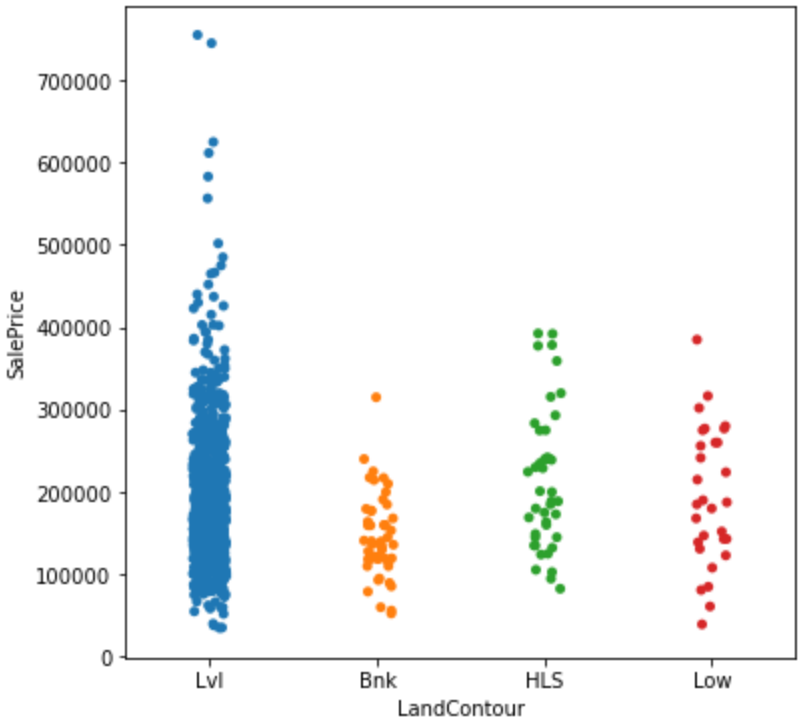
For Categorical Columns

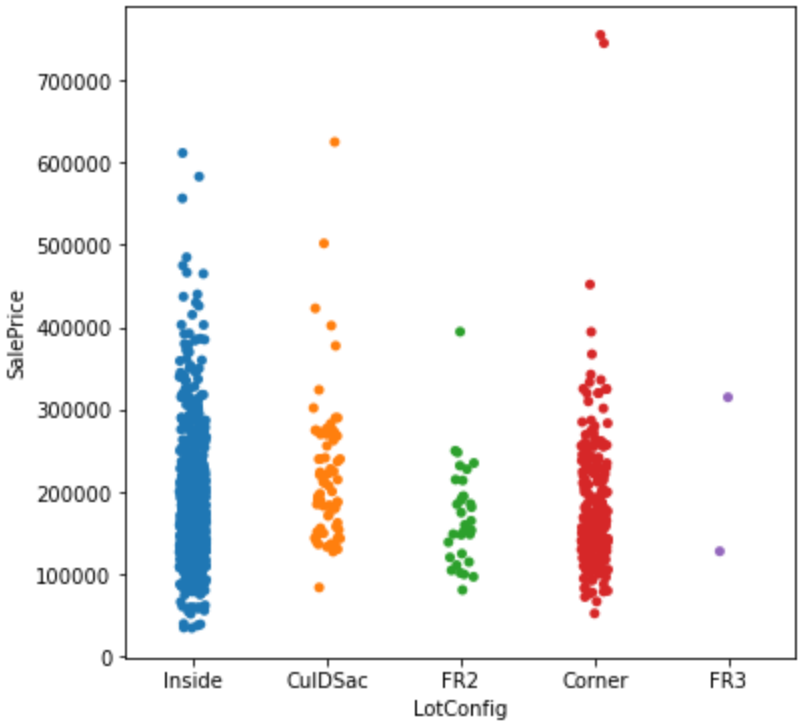
In [39]:
```
for i in train:
    if train[i].dtypes=='O':
        plt.figure(figsize=(6,6))
        sns.stripplot(x=i,y='SalePrice',data=train)
        plt.show()
```

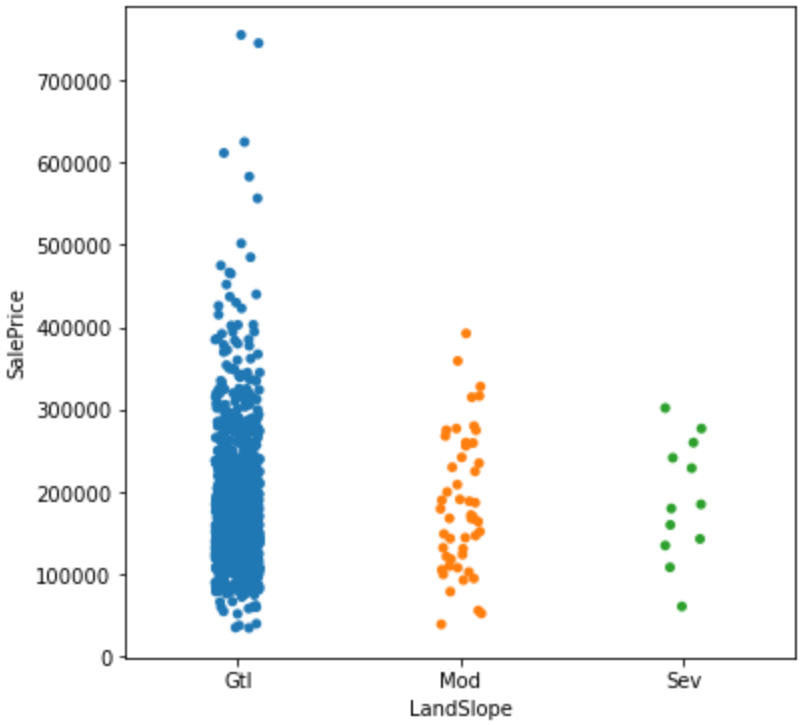


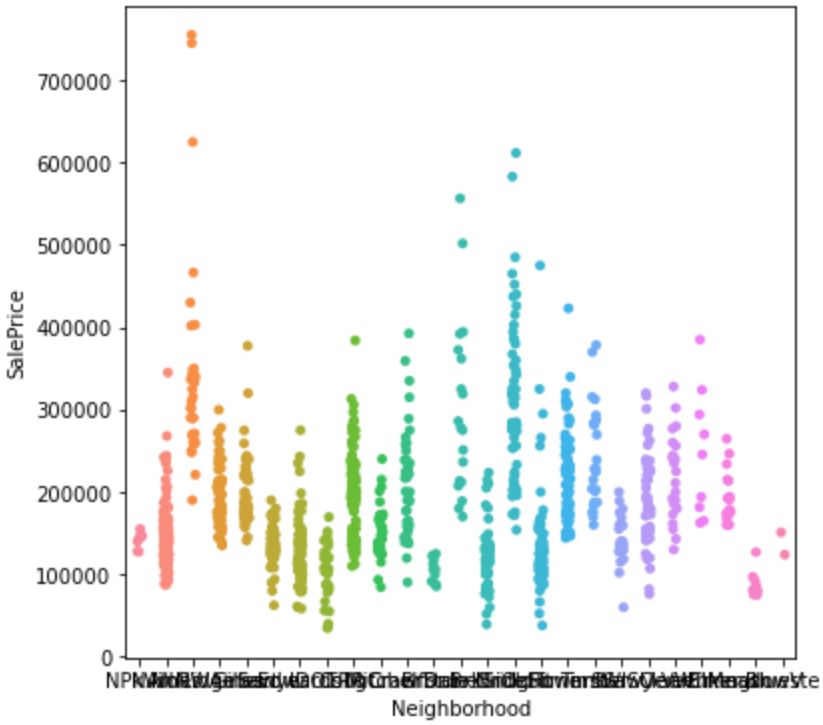

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
Pave
Grvl
Street

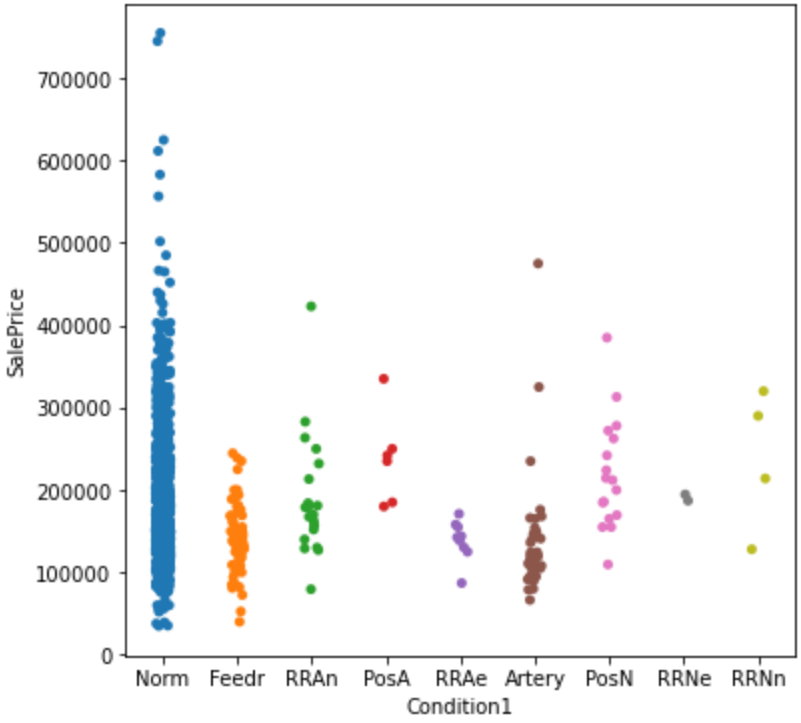

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
IR1
Reg
IR2
IR3
LotShape

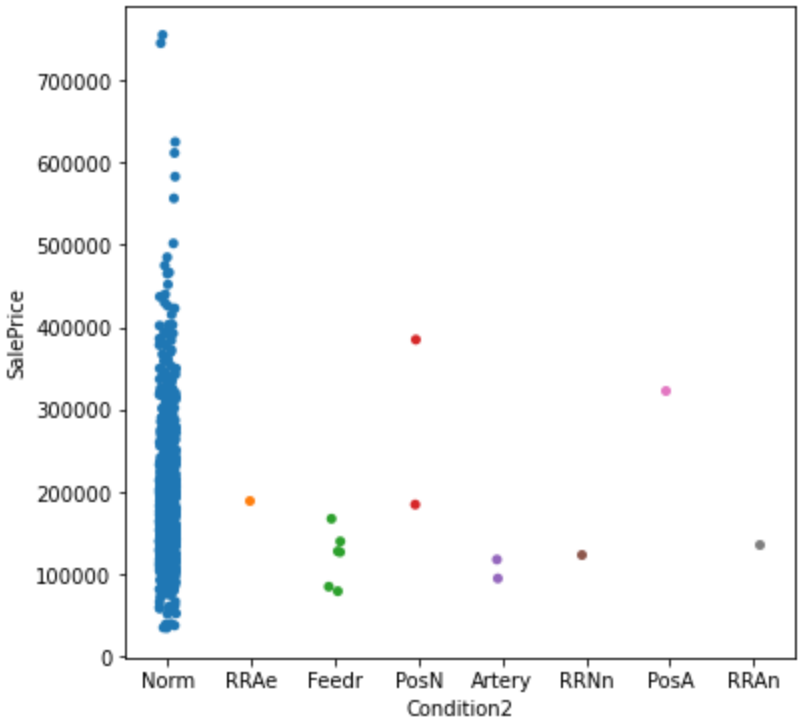

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Lvl
Bnk
HLS
Low
LandContour

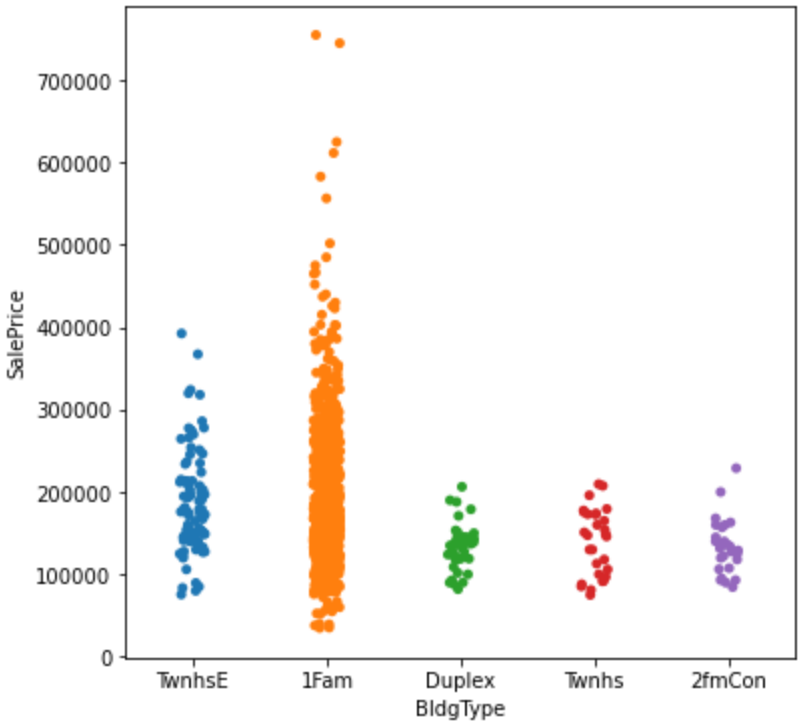

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Inside
CulDSac
FR2
Corner
FR3
LotConfig

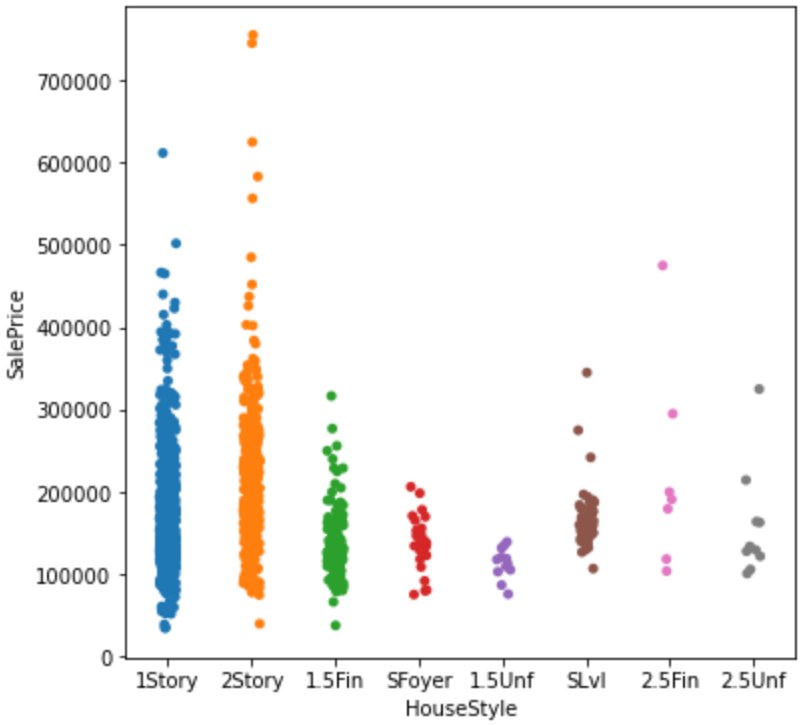

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Gtl
Mod
Sev
LandSlope

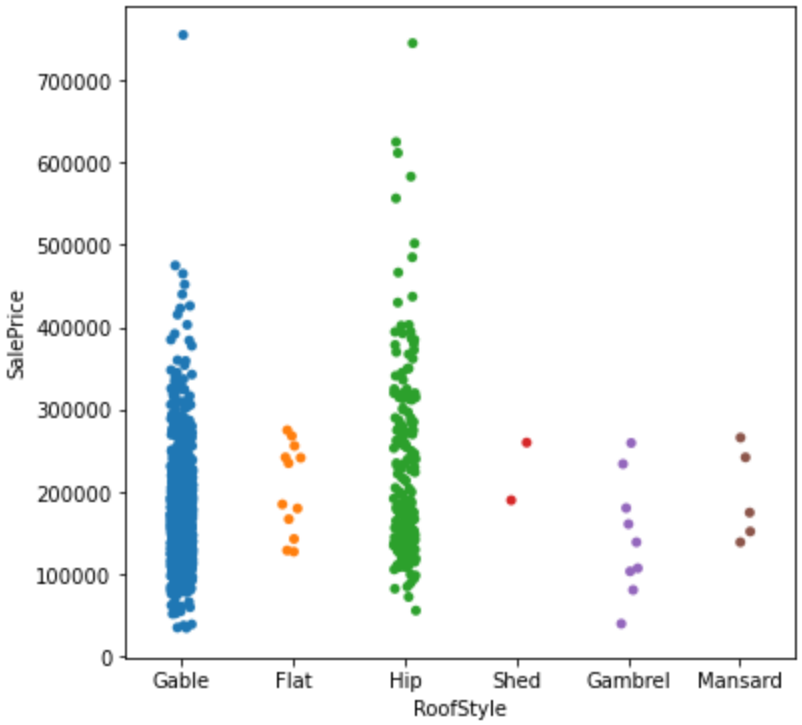

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Neighborhood

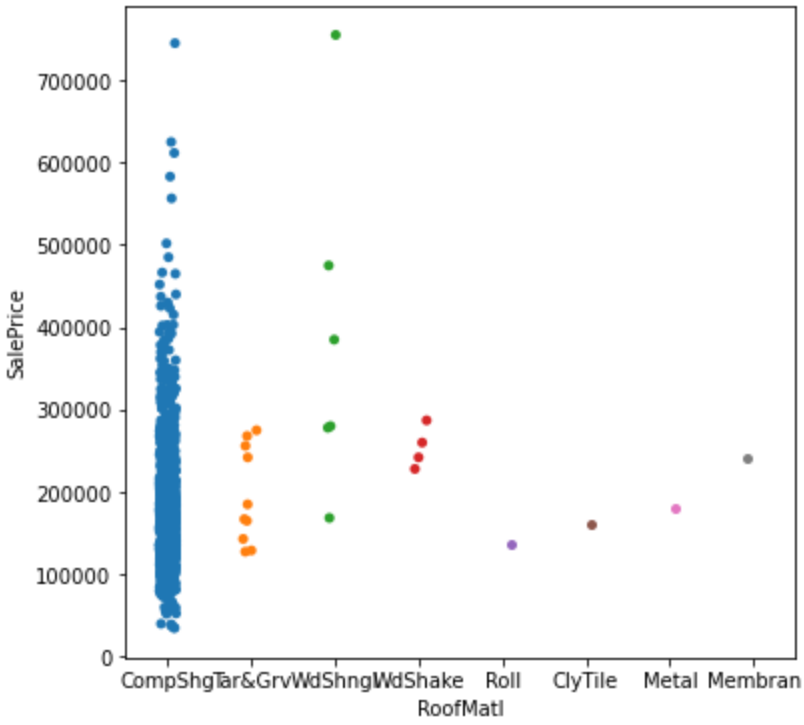

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Norm
Feedr
RRAn
PosA
RRAe
Artery
PosN
RRNe
RRNn
Condition1

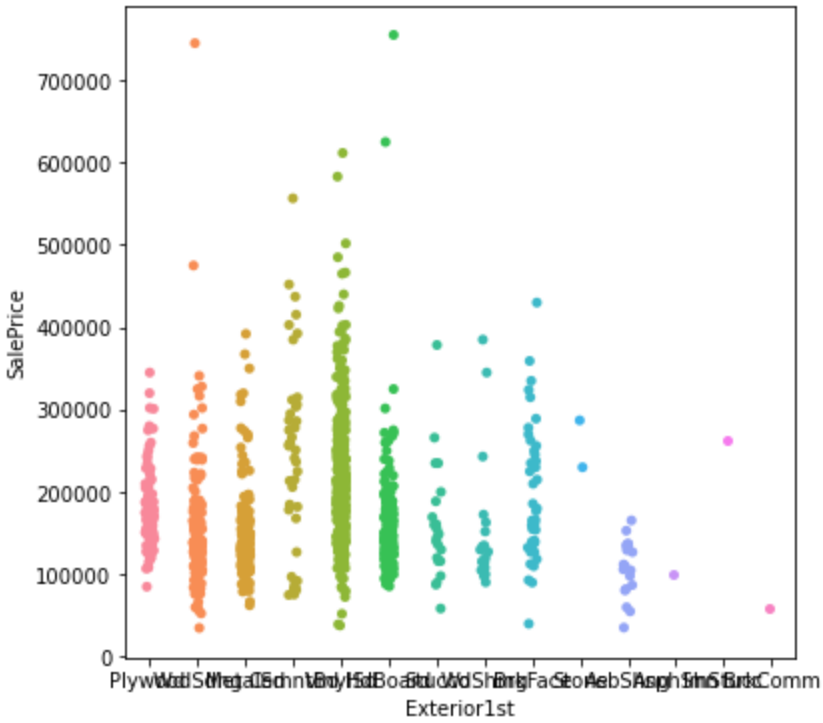

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Norm
RRAe
Feedr
PosN
Artery
RRNn
PosA
RRAn
Condition2

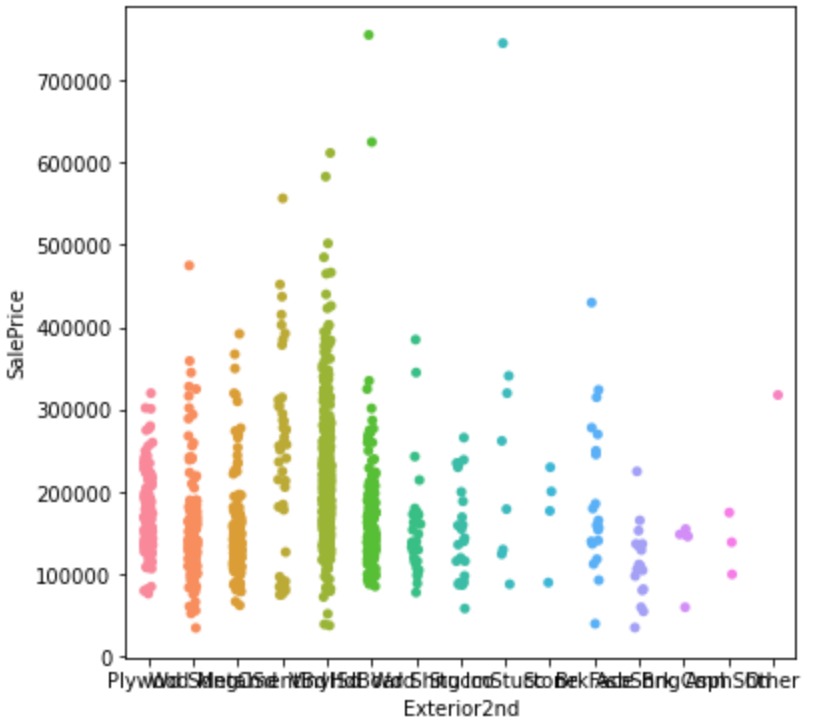

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
TwnhsE
1Fam
Duplex
Twnhs
2fmCon
BldgType

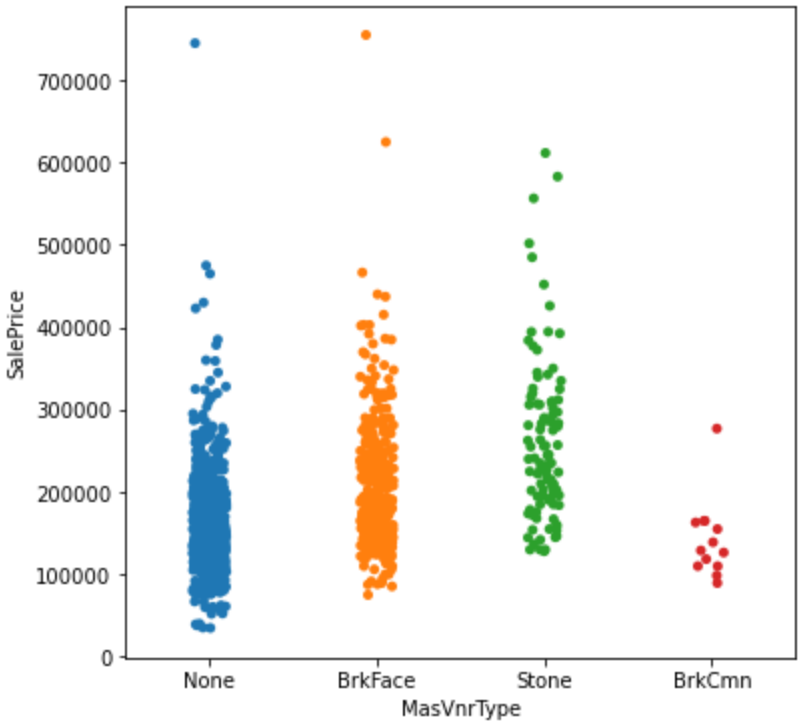

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
1Story
2Story
1.5Fin
SFoyer
1.5Unf
SLvl
2.5Fin
2.5Unf
HouseStyle

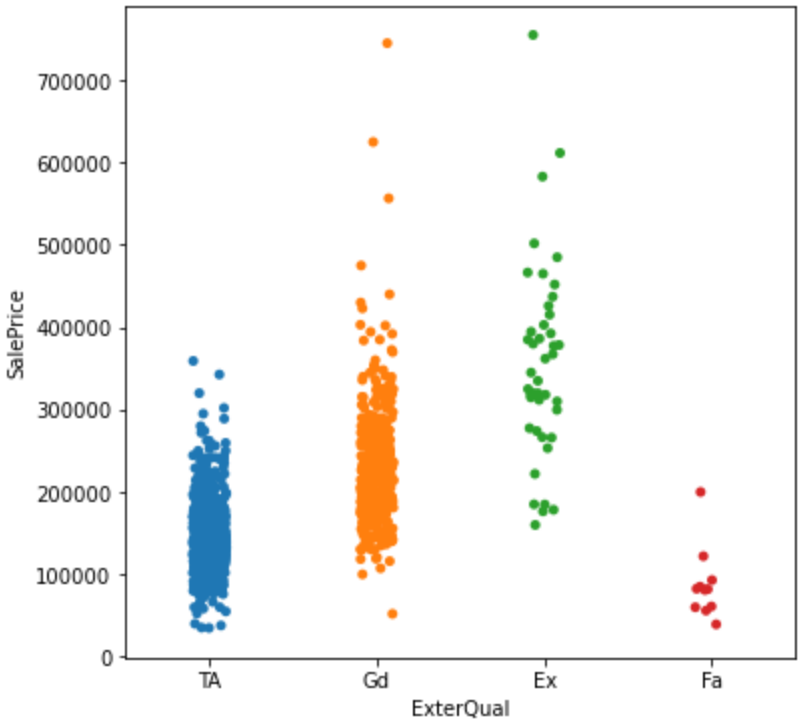

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Gable
Flat
Hip
Shed
Gambrel
Mansard
RoofStyle

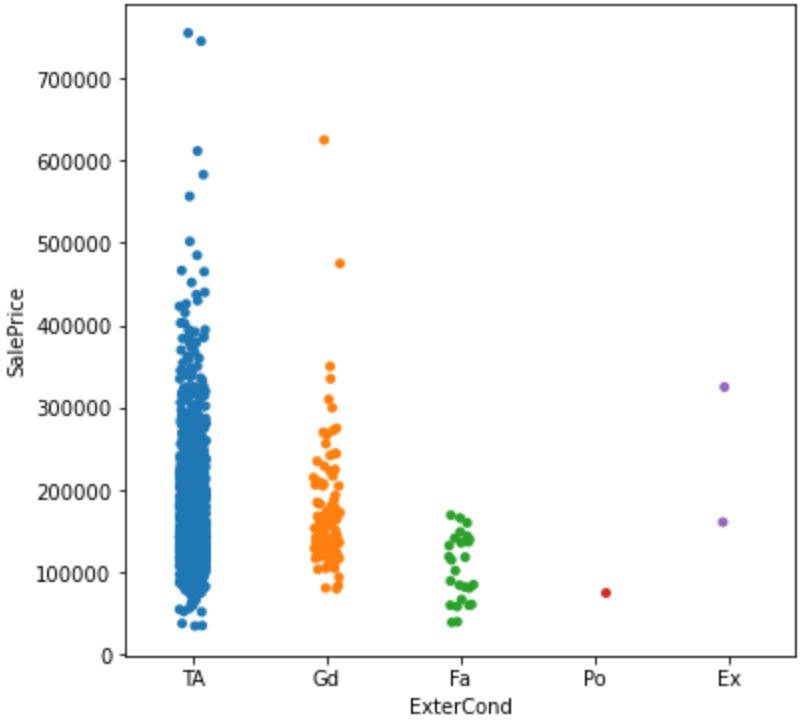

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
CompShg
Tar&Grv
WdShngl
WdShake
Roll
ClyTile
Metal
Membran
RoofMatl

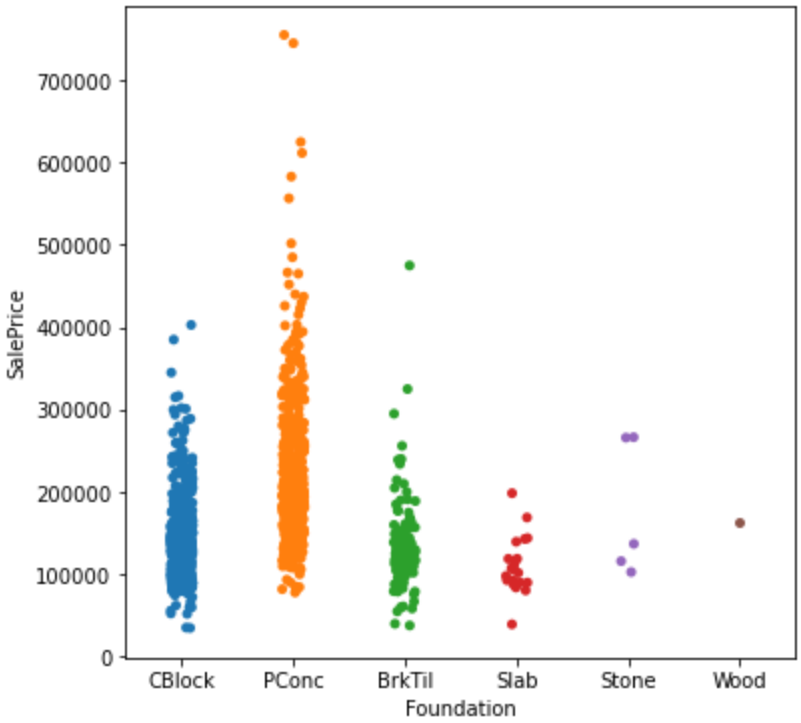

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Exterior1st

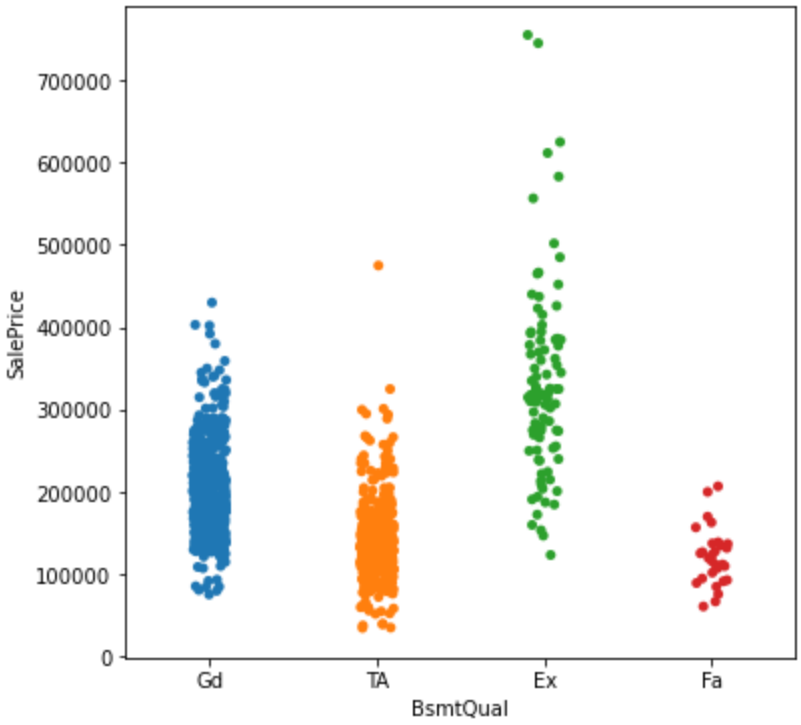

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Exterior2nd

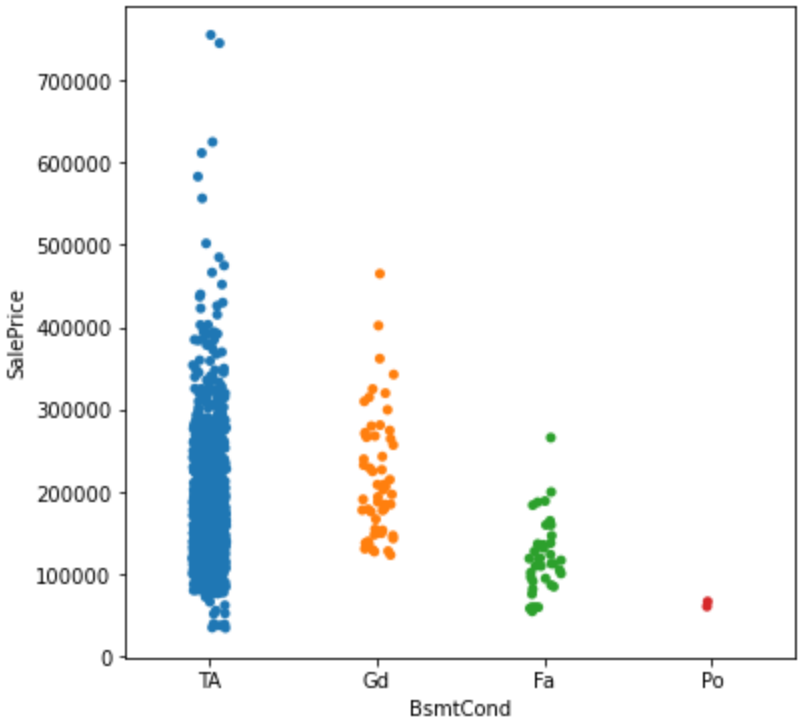

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
None
BrkFace
Stone
BrkCmn
MasVnrType

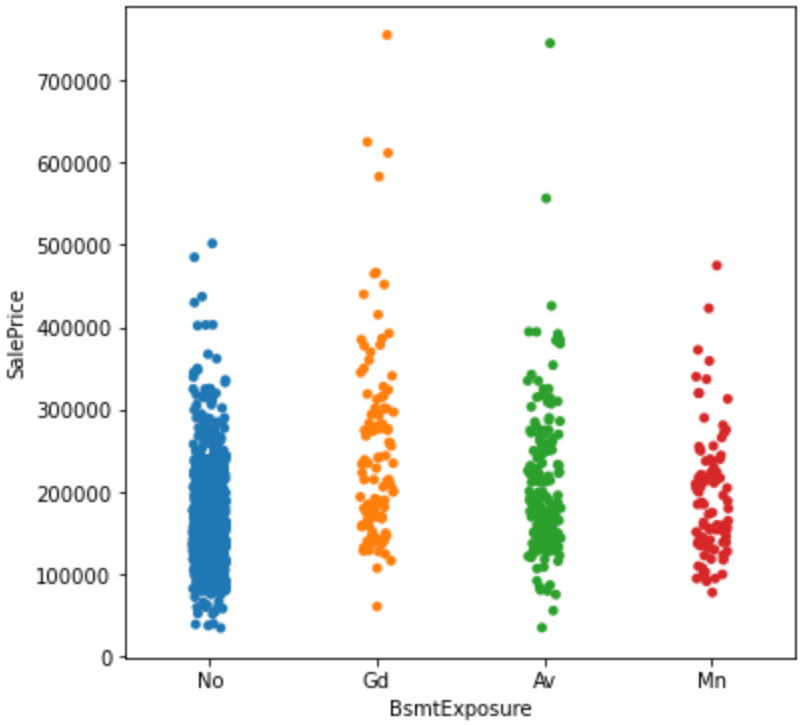

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
TA
Gd
Ex
Fa
ExterQual

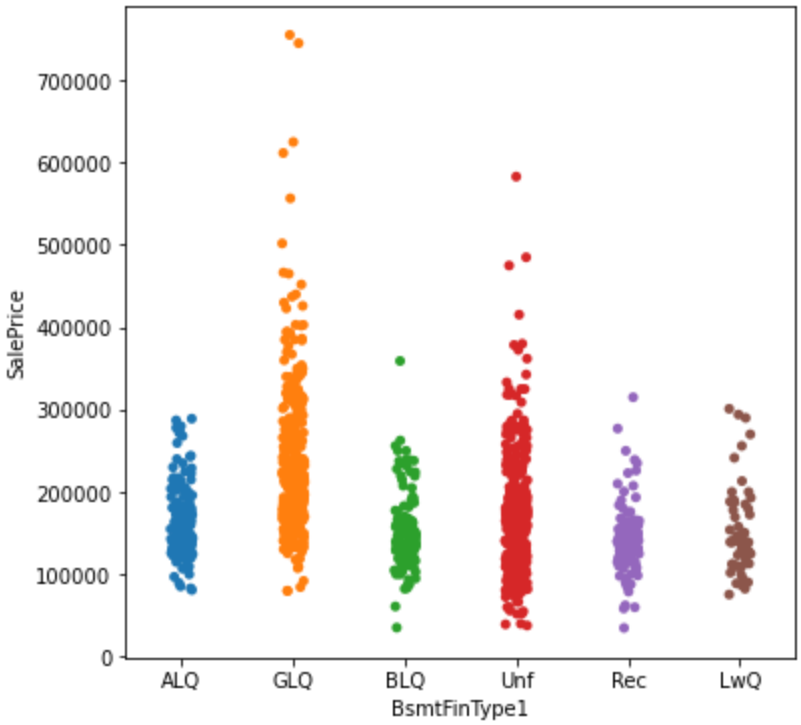

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
TA
Gd
Fa
Po
Ex
ExterCond

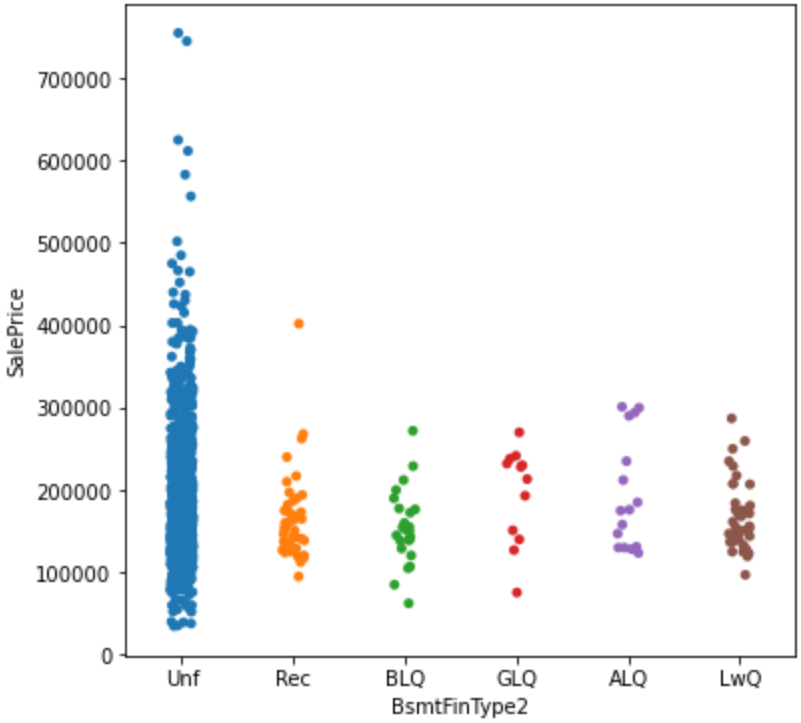

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
CBlock
PConc
BrkTil
Slab
Stone
Wood
Foundation

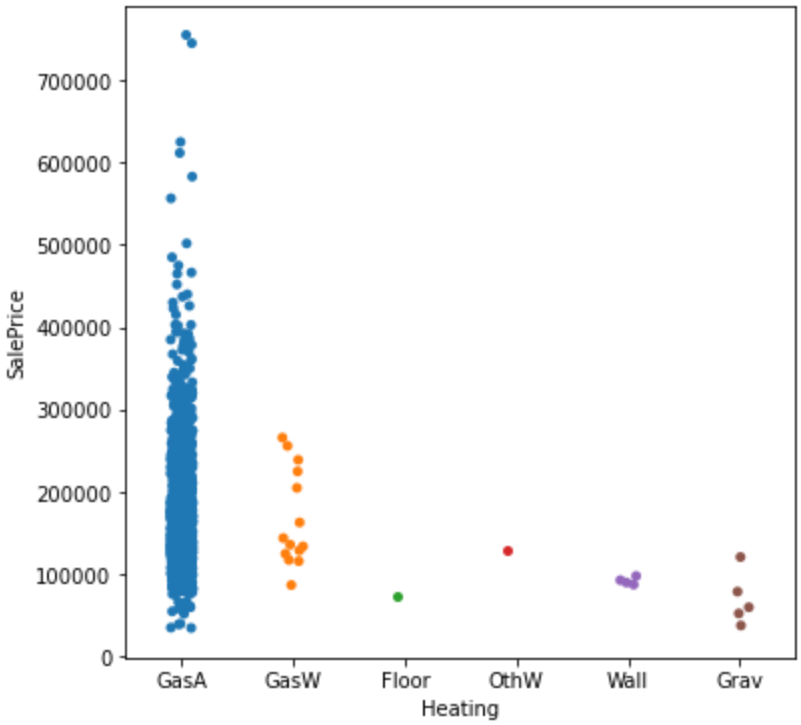

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Gd
TA
Ex
Fa
BsmtQual

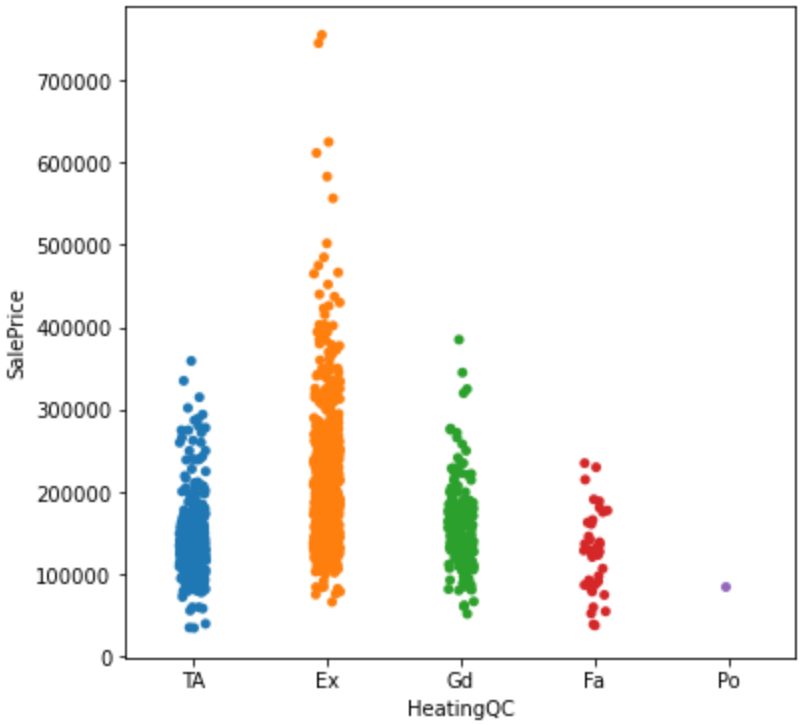

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
TA
Gd
Fa
Po
BsmtCond

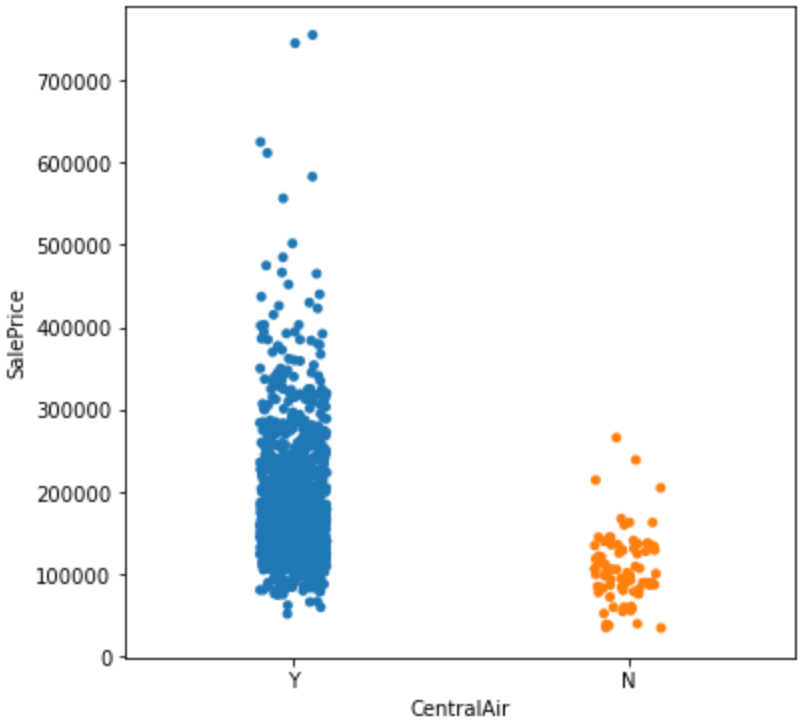

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
No
Gd
Av
Mn
BsmtExposure

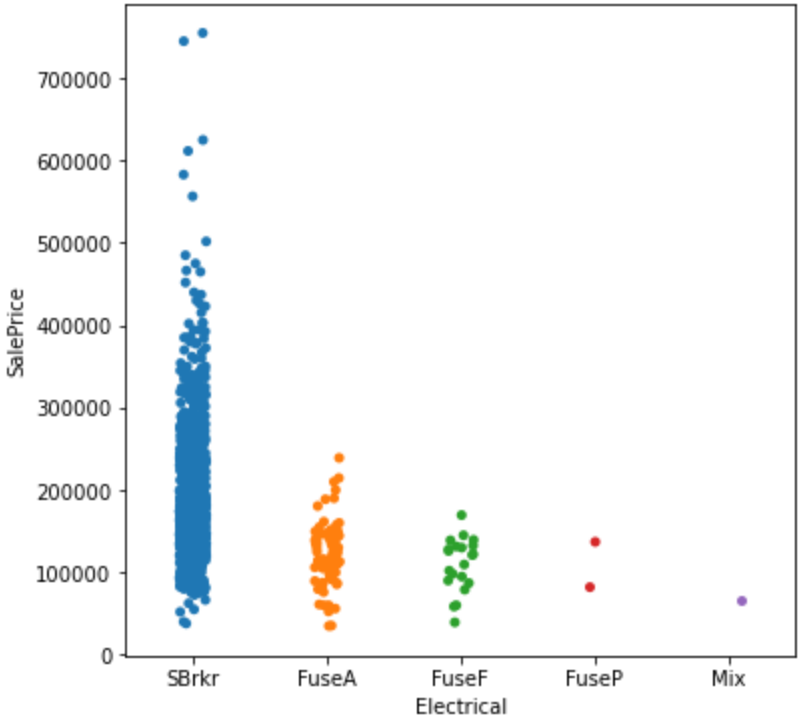

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
ALQ
GLQ
BLQ
Unf
Rec
LwQ
BsmtFinType1

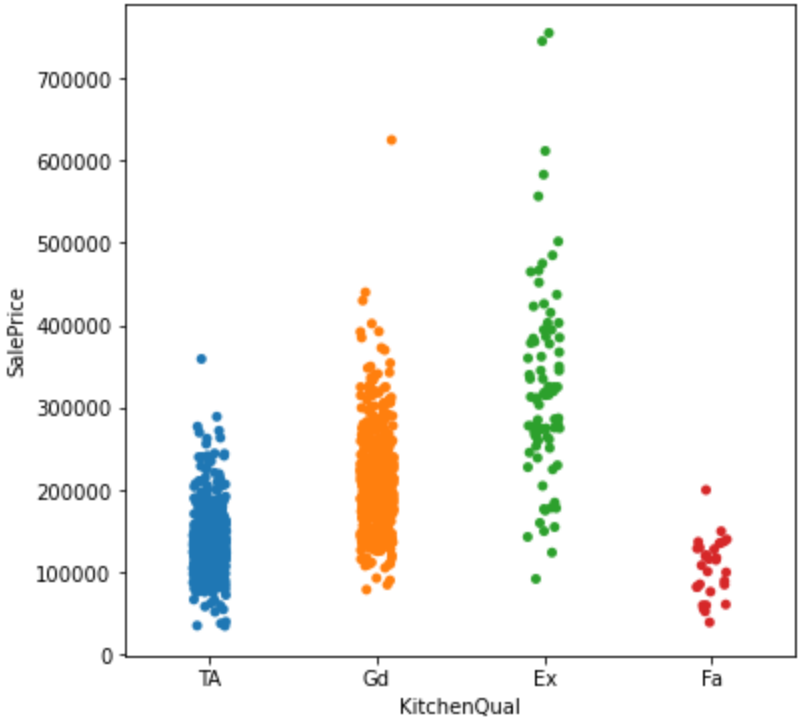

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Unf
Rec
BLQ
GLQ
ALQ
LwQ
BsmtFinType2

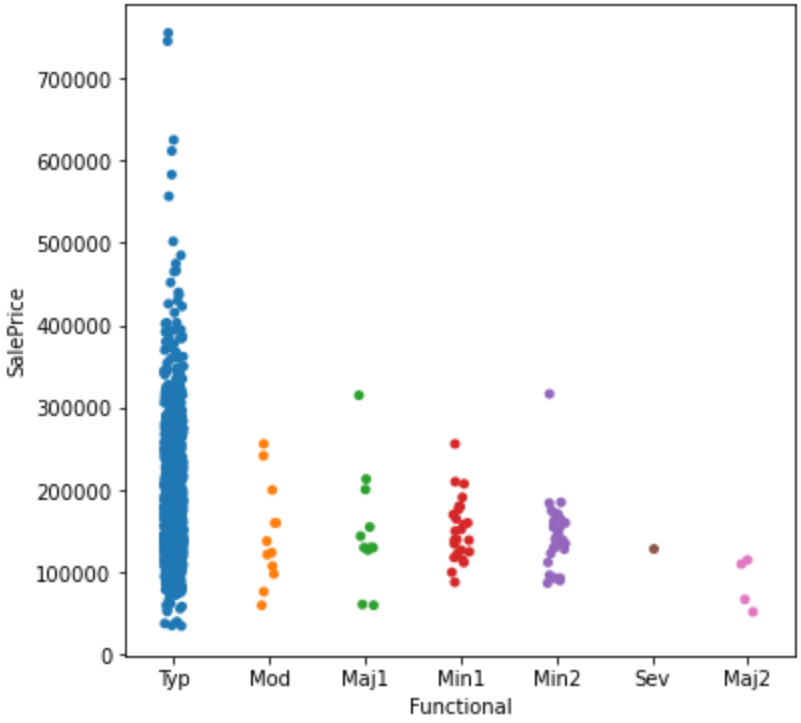

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
GasA
GasW
Floor
OthW
Wall
Grav
Heating

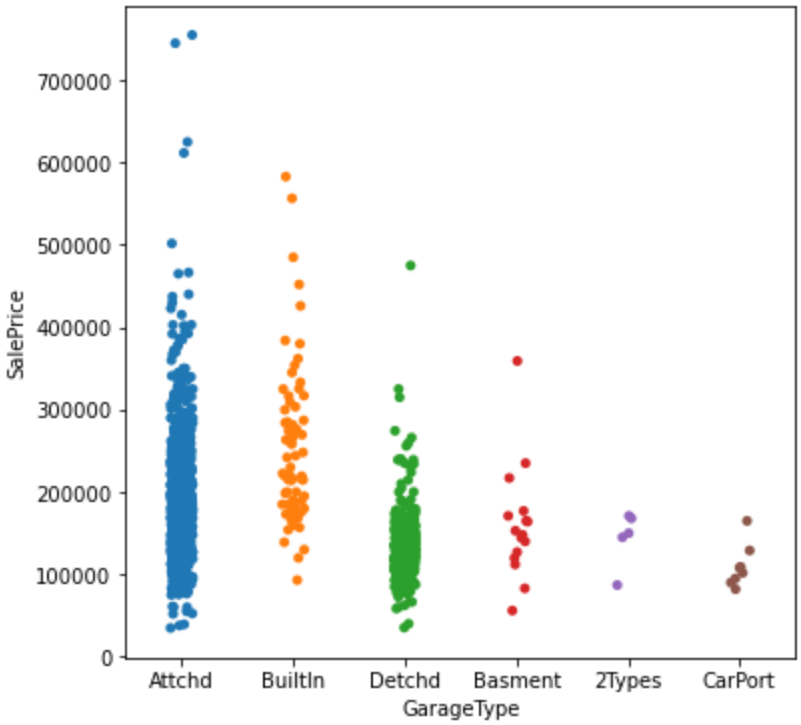

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
TA
Ex
Gd
Fa
Po
HeatingQC

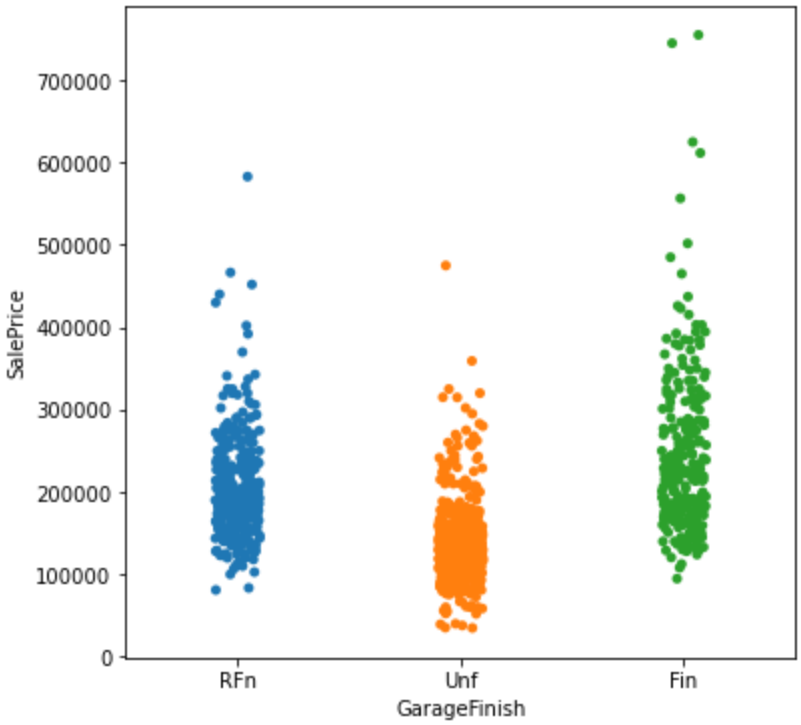

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Y
N
CentralAir

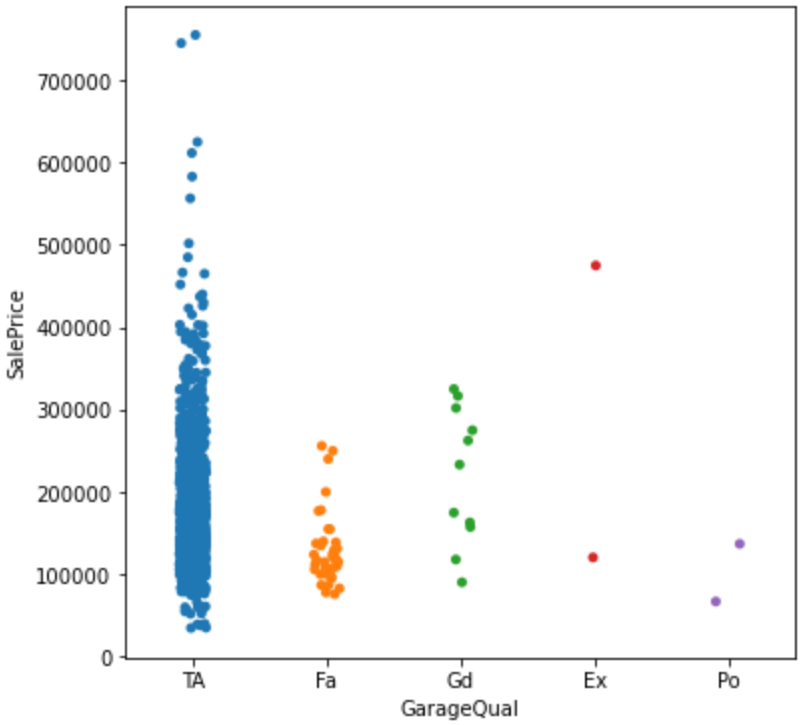

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
SBrkr
FuseA
FuseF
FuseP
Mix
Electrical

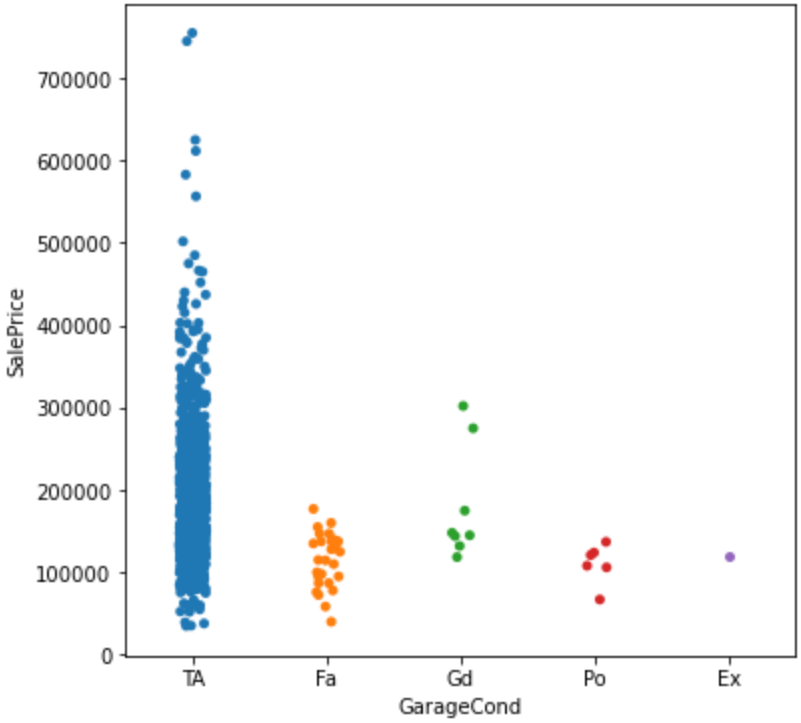

SalePrice
0
100000
200000
300000
400000
500000
600000
700000
TA
Gd
Ex
Fa
KitchenQual

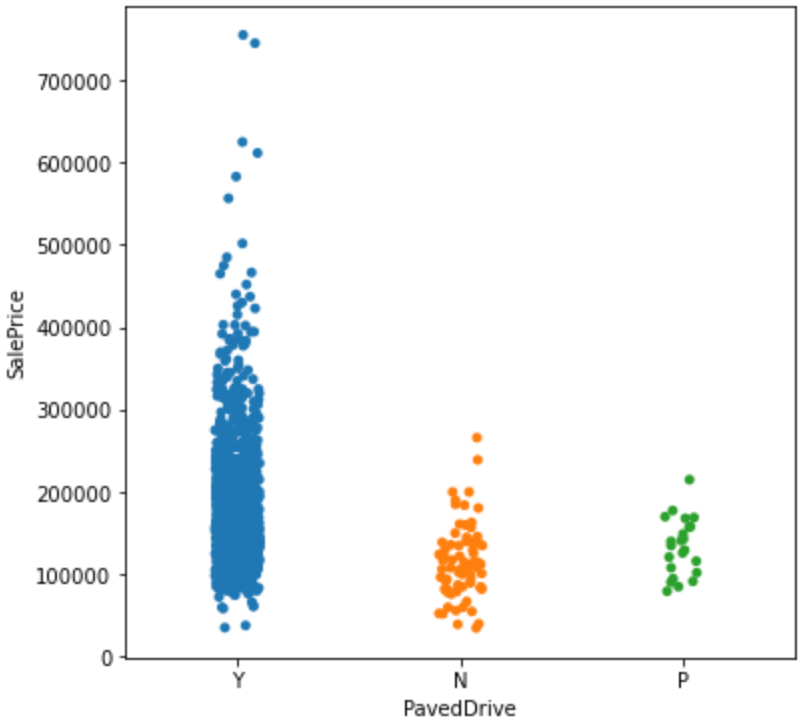

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Typ
Mod
Maj1
Min1
Min2
Sev
Maj2
Functional

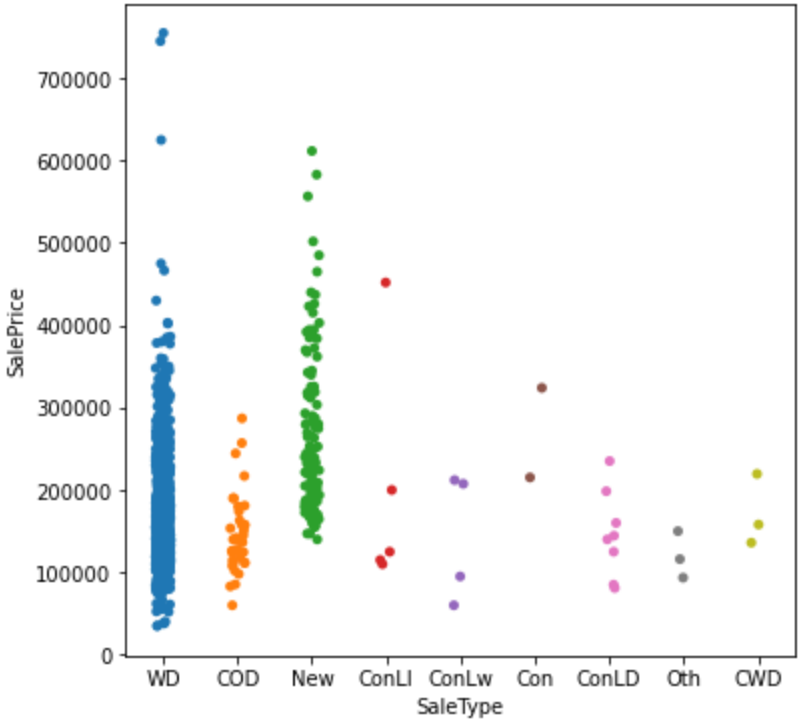

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Attchd
BuiltIn
Detchd
Basment
2Types
CarPort
GarageType

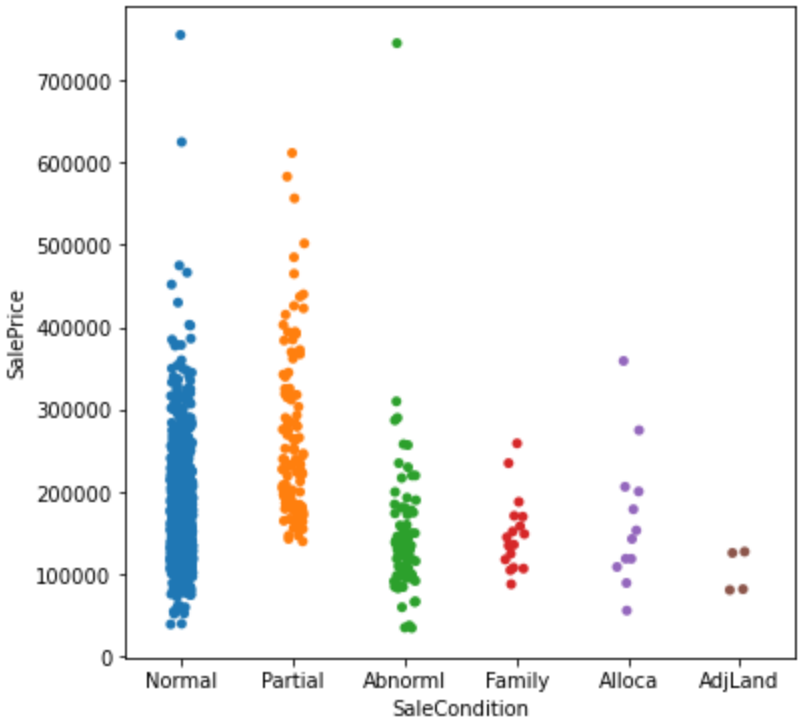

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
RFn
Unf
Fin
GarageFinish


SalePrice
700000
600000
500000
400000
300000
200000
100000
0
TA
Fa
Gd
Ex
Po
GarageQual


700000
600000
500000
400000
300000
200000
100000
0
SalePrice
TA
Fa
Gd
Po
Ex
GarageCond

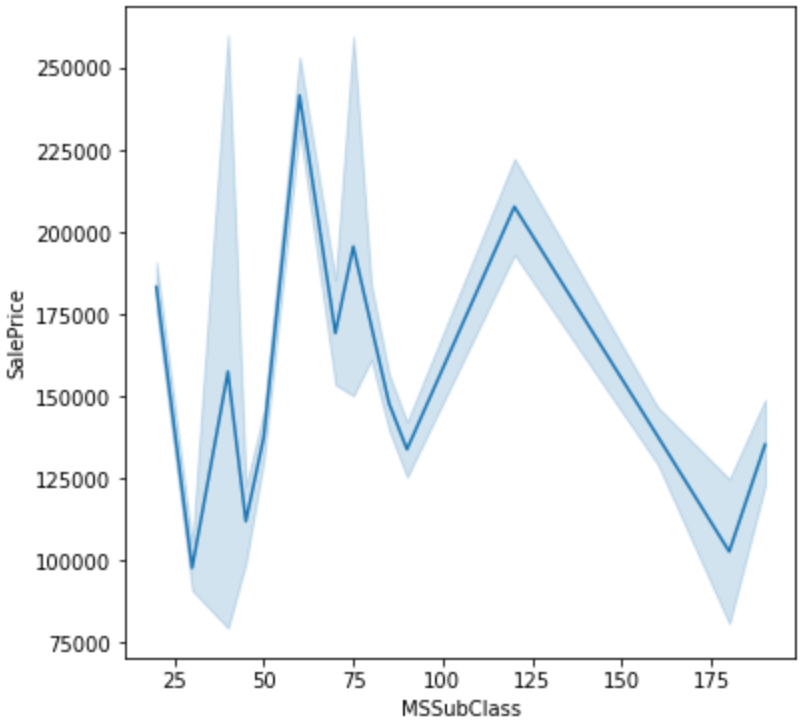

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
Y
N
P
PavedDrive

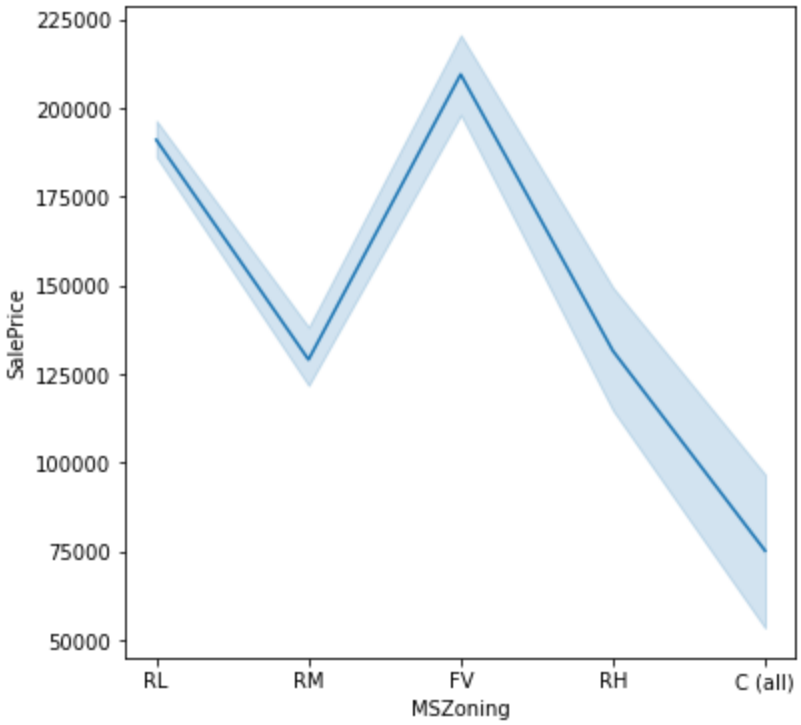


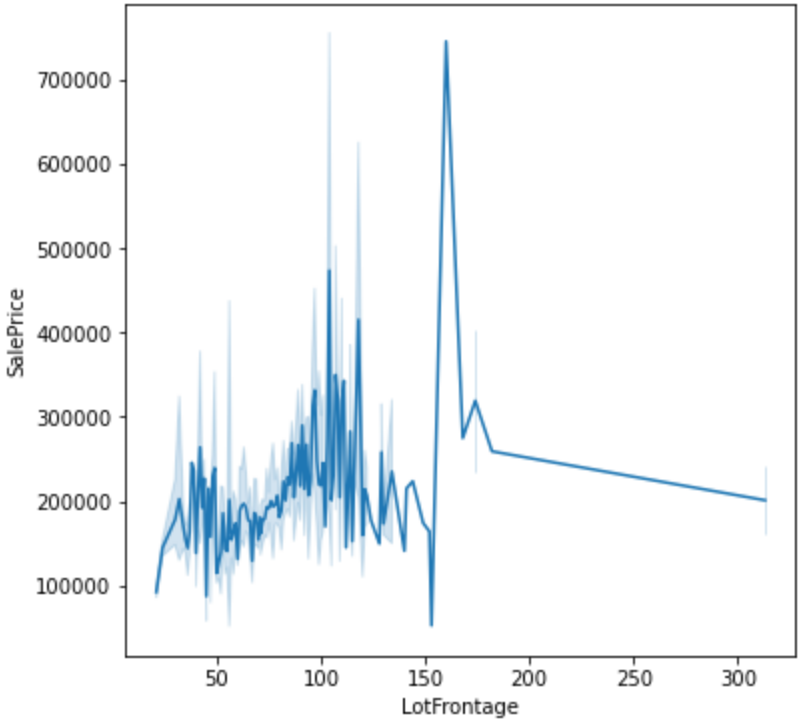


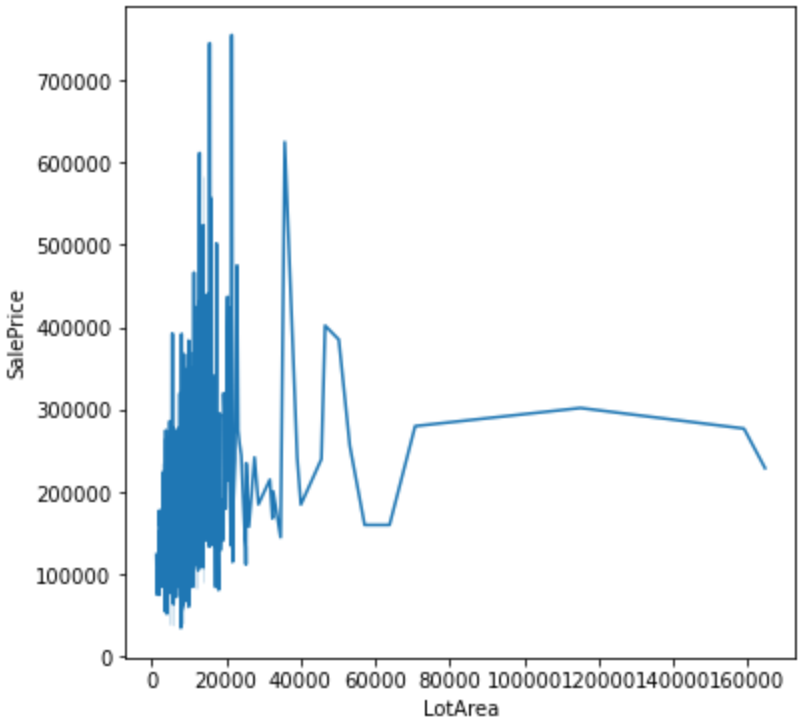
For numerical column

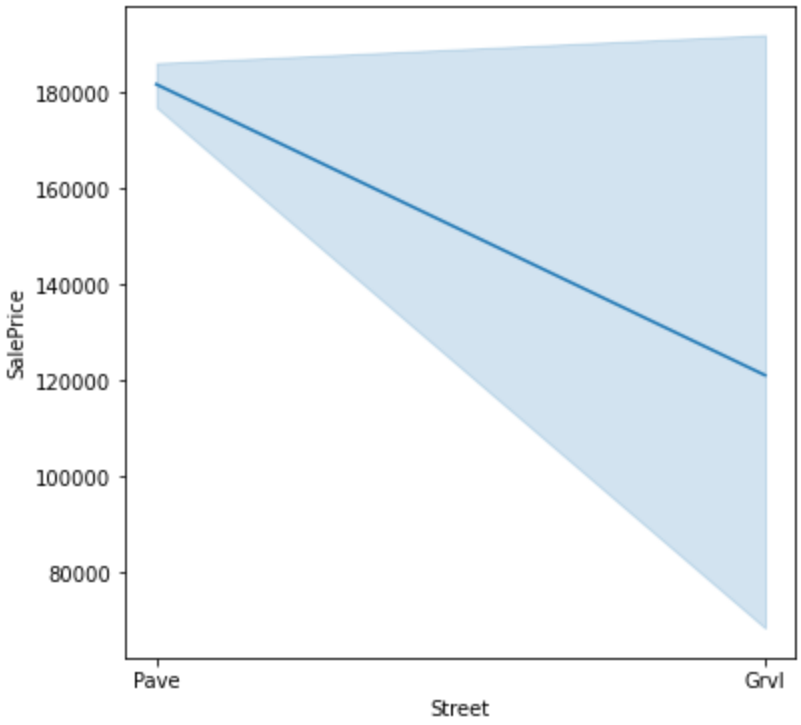
In [40]:
```
for i in train:
    if train[i].dtypes!='O' or i!='SalePrice':
        plt.figure(figsize=(6,6))
        sns.lineplot(x=i,y='SalePrice',data=train)
        plt.show()
```

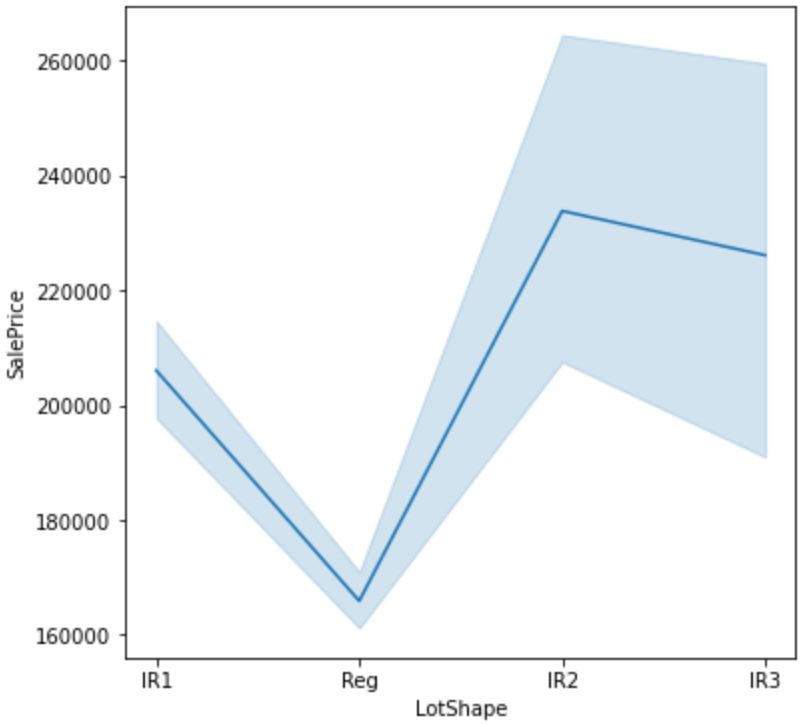

250000
225000
200000
175000
150000
125000
100000
75000
SalePrice
25
50
75
100
125
150
175
MSSubClass

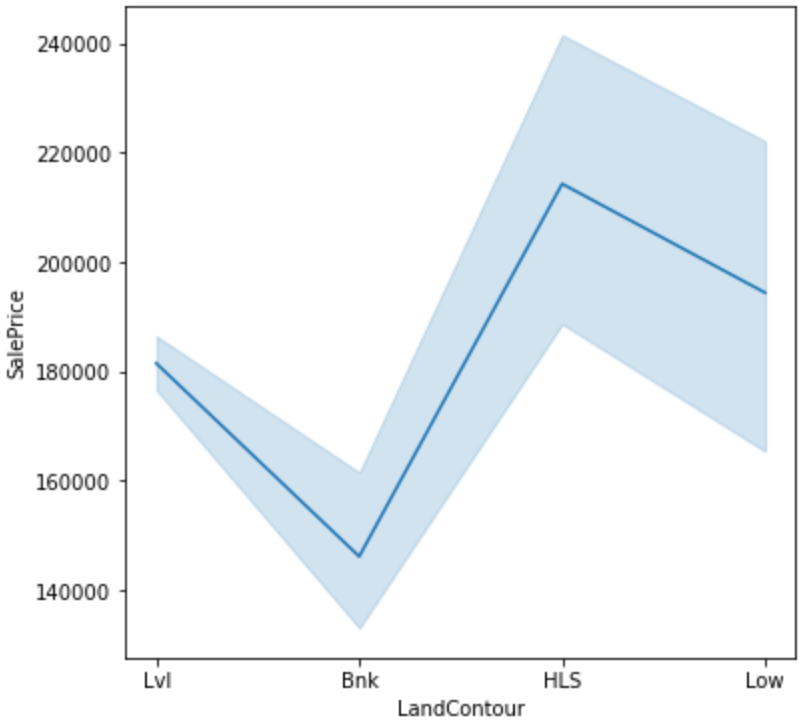

225000
200000
175000
150000
125000
100000
75000
50000
SalePrice
RL
RM
FV
RH
C (all)
MSZoning

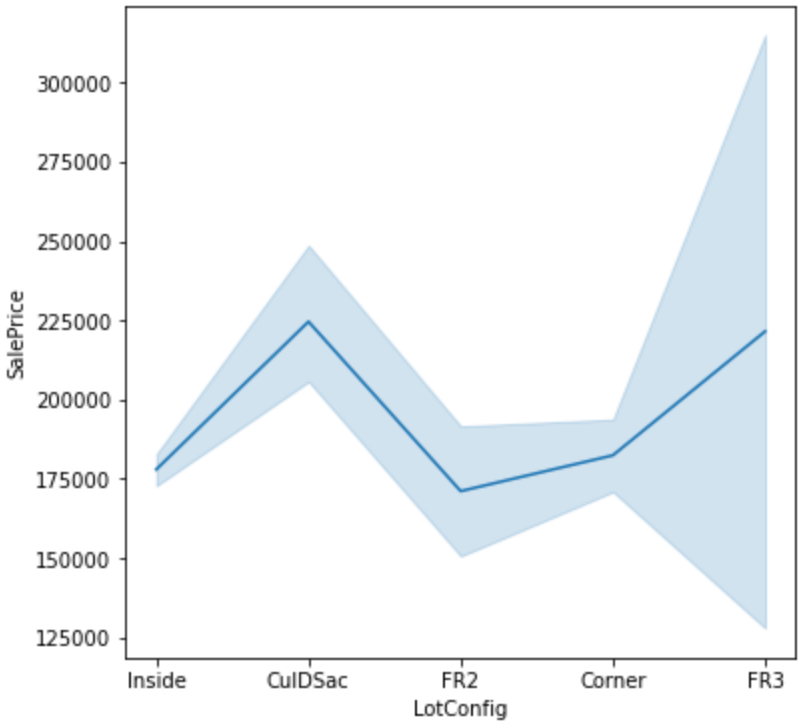

700000
600000
500000
400000
300000
200000
100000
SalePrice
50
100
150
200
250
300
LotFrontage

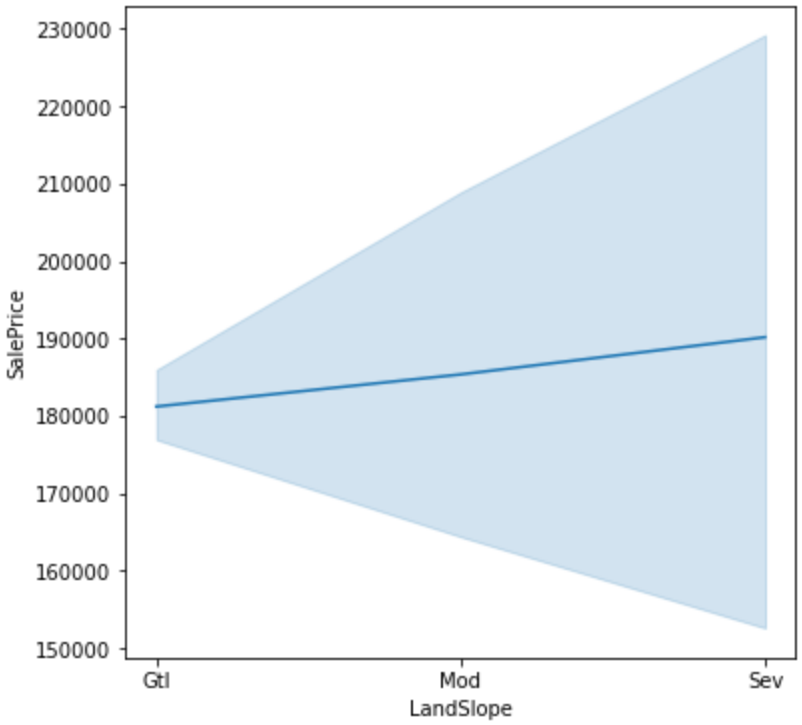

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
0
20000
40000
60000
80000
100000
120000
140000
160000
LotArea

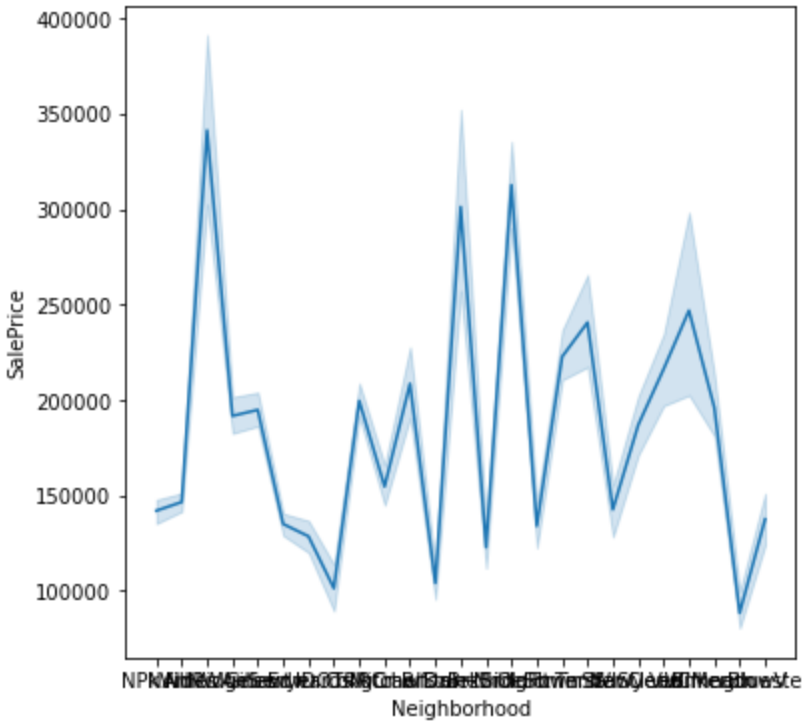

180000
160000
140000
120000
100000
80000
SalePrice
Pave
Grvl
Street

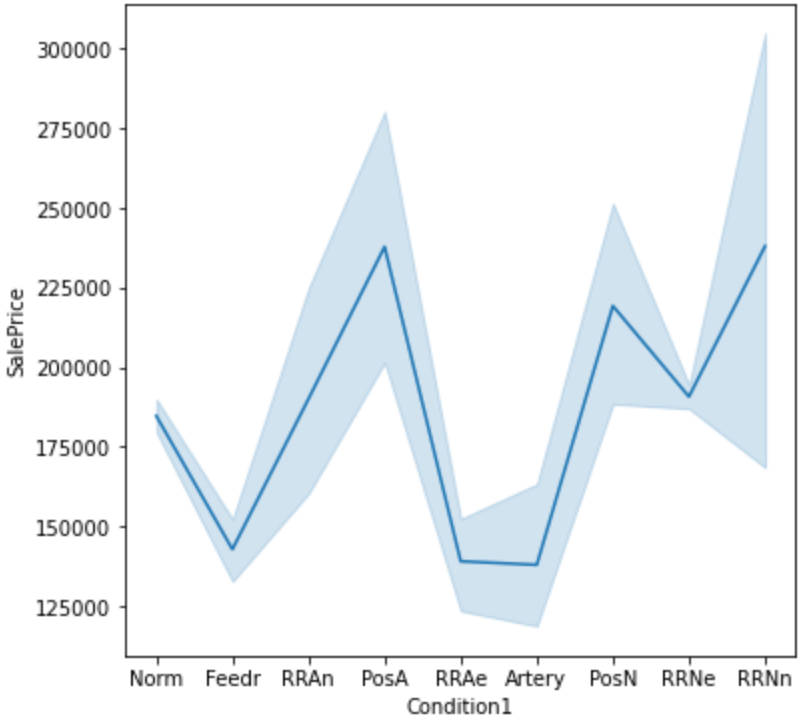

260000
240000
220000
200000
180000
160000
SalePrice
IR1
Reg
IR2
IR3
LotShape

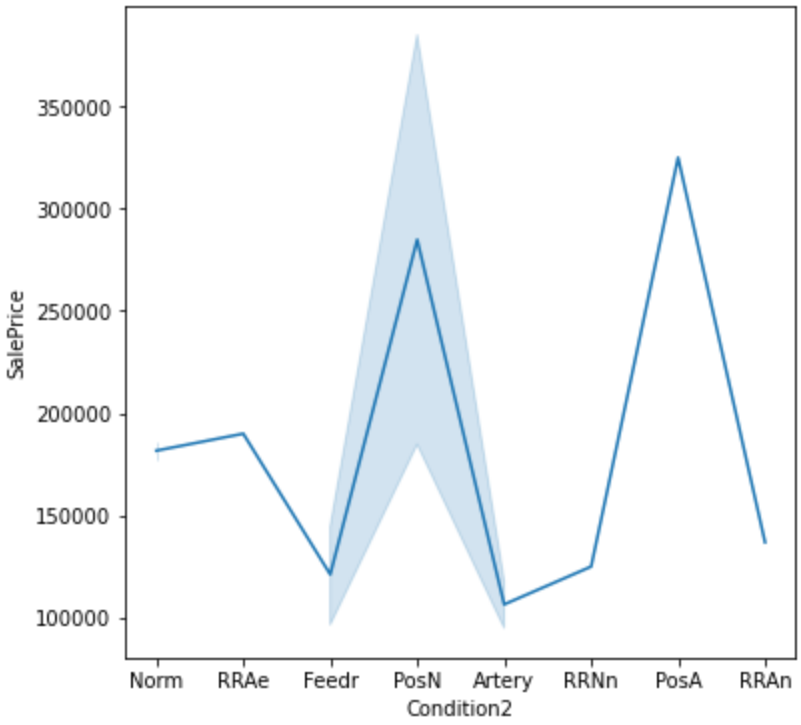

240000
220000
200000
180000
160000
140000
SalePrice
Lvl
Bnk
HLS
Low
LandContour

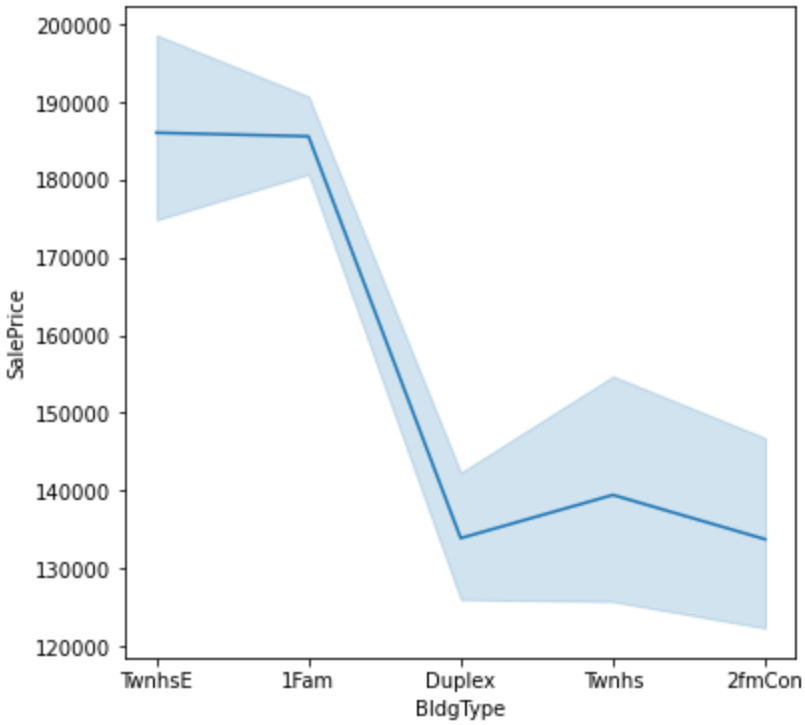

300000
275000
250000
225000
200000
175000
150000
125000
SalePrice
Inside
CulDSac
FR2
Corner
FR3
LotConfig

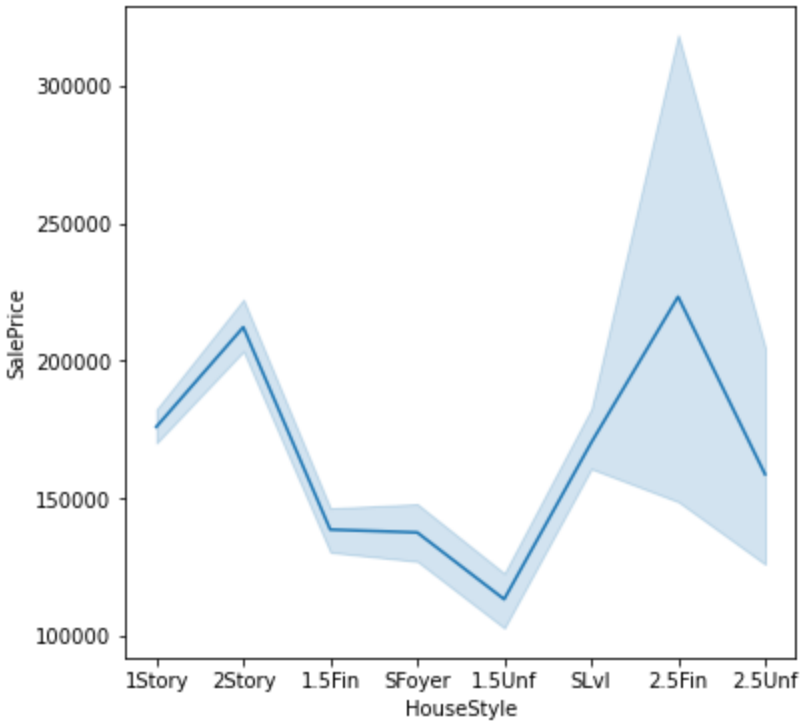

230000
220000
210000
200000
190000
180000
170000
160000
150000
SalePrice
Gtl
Mod
Sev
LandSlope

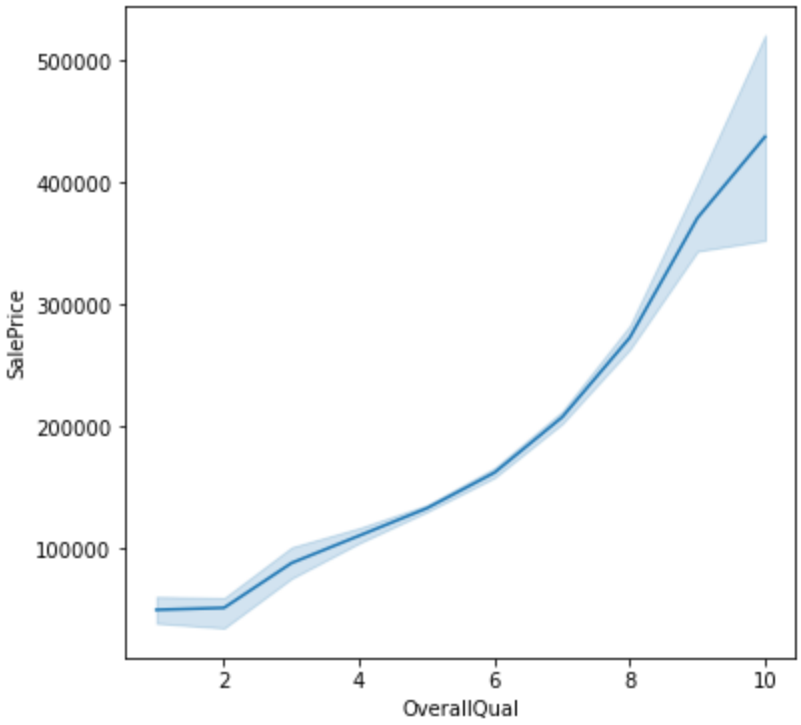

400000
350000
300000
250000
200000
150000
100000
SalePrice
Neighborhood

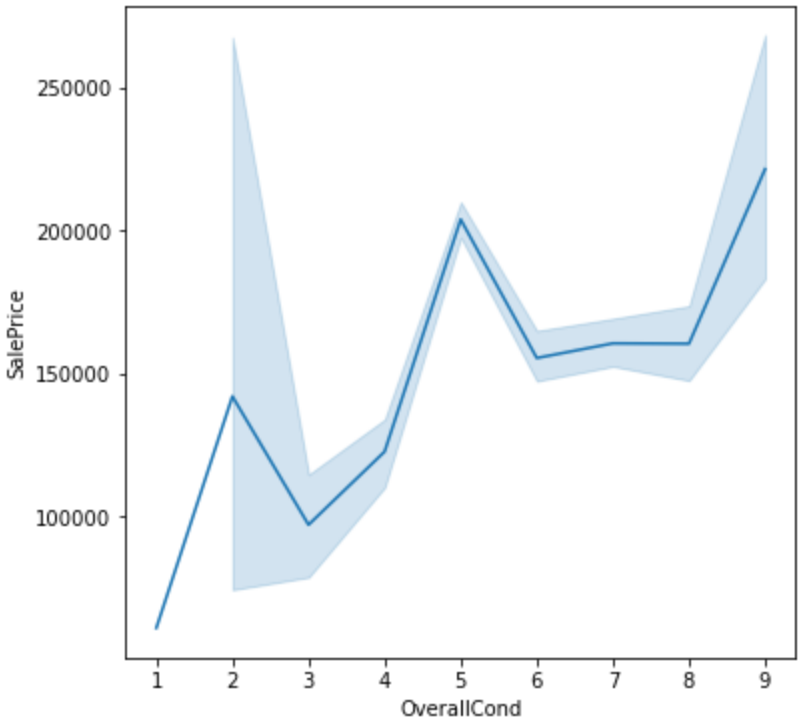

300000
275000
250000
225000
200000
175000
150000
125000
SalePrice
Norm
Feedr
RRAn
PosA
RRAe
Artery
PosN
RRNe
RRNn
Condition1

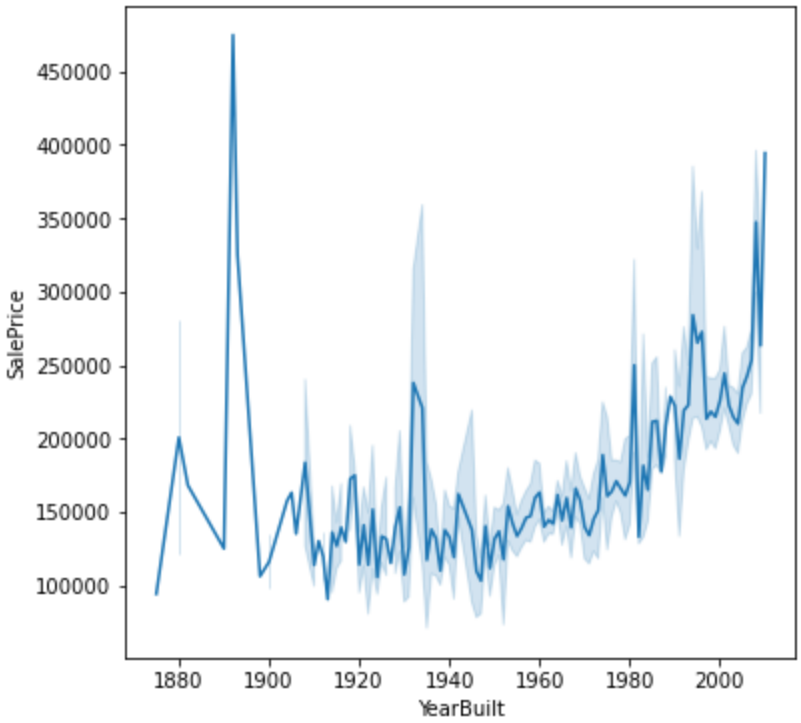

350000
300000
250000
200000
150000
100000
SalePrice
Norm
RRAe
Feedr
PosN
Artery
RRNn
PosA
RRAn
Condition2

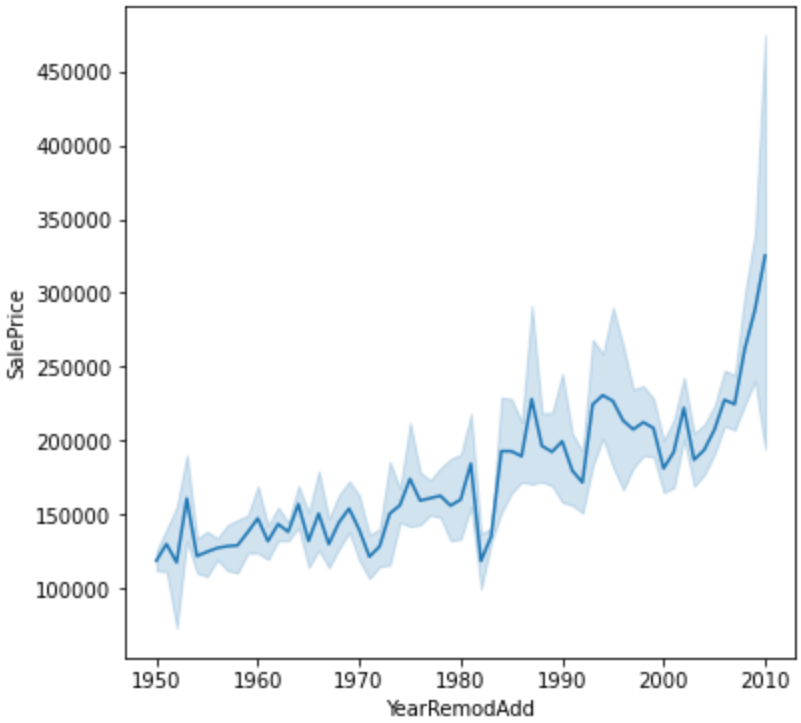

200000
190000
180000
170000
160000
150000
140000
130000
120000
SalePrice
TwnhsE
1Fam
Duplex
Twnhs
2fmCon
BldgType

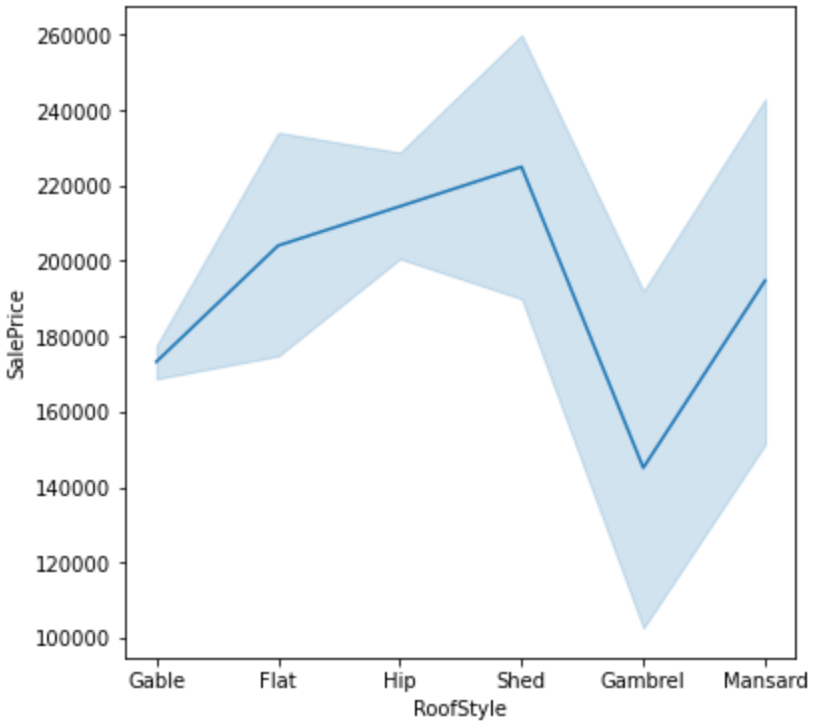

300000
250000
200000
150000
100000
SalePrice
1Story
2Story
1.5Fin
SFoyer
1.5Unf
SLvl
2.5Fin
2.5Unf
HouseStyle

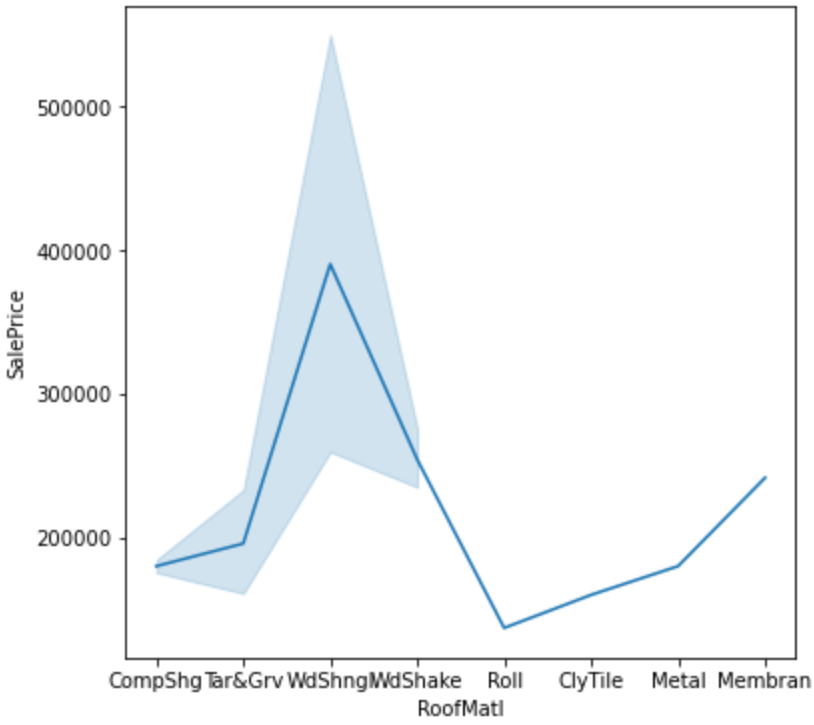

500000
400000
300000
200000
100000
SalePrice
2
4
6
8
10
OverallQual

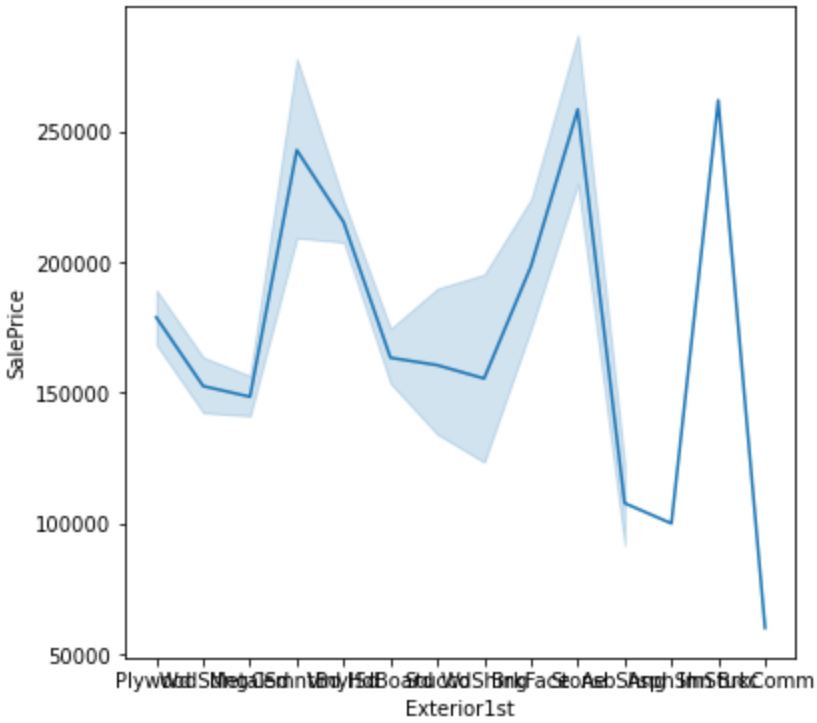

250000
200000
150000
100000
SalePrice
1
2
3
4
5
6
7
8
9
OverallCond

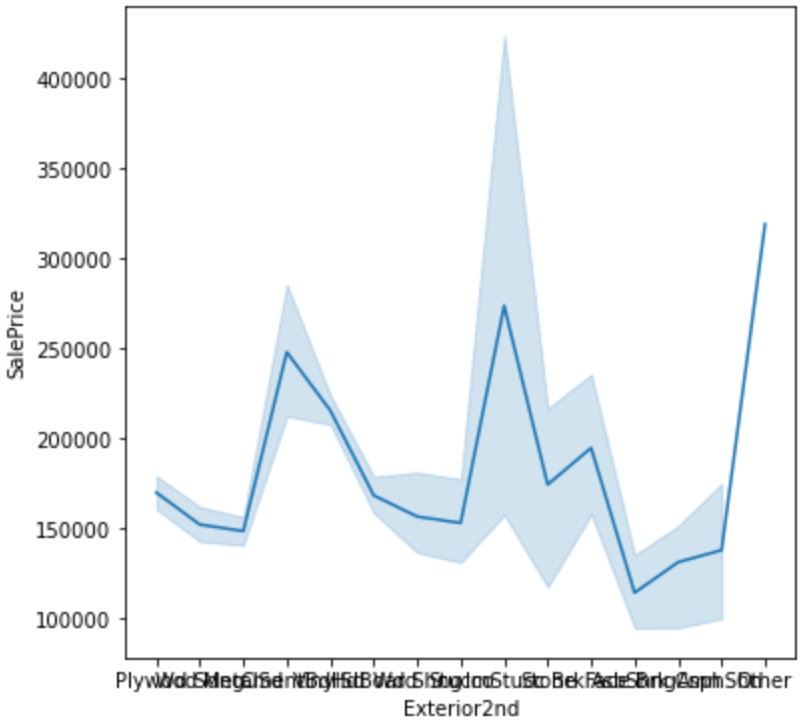

SalePrice
450000
400000
350000
300000
250000
200000
150000
100000
1880
1900
1920
1940
1960
1980
2000
YearBuilt

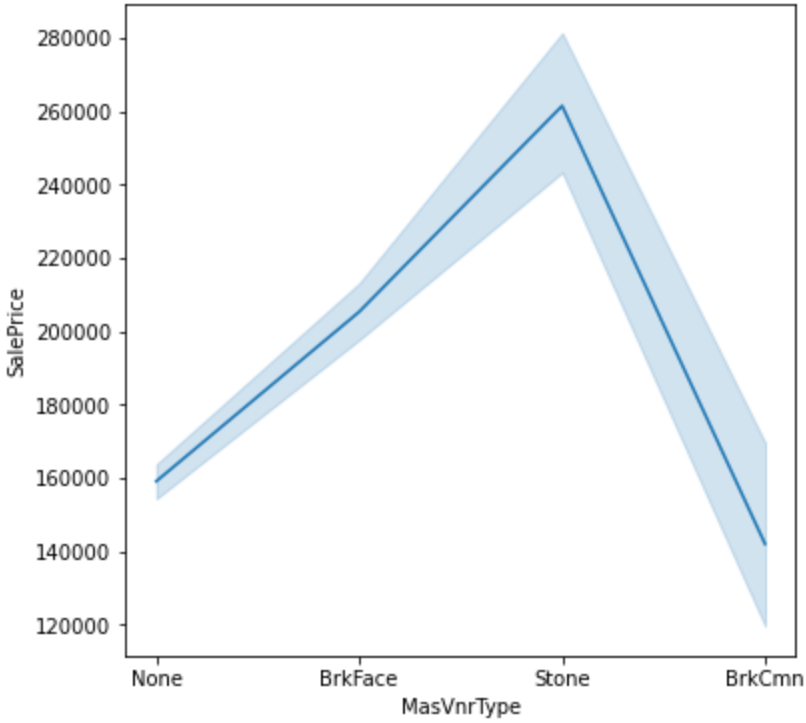

SalePrice
450000
400000
350000
300000
250000
200000
150000
100000
1950
1960
1970
1980
1990
2000
2010
YearRemodAdd

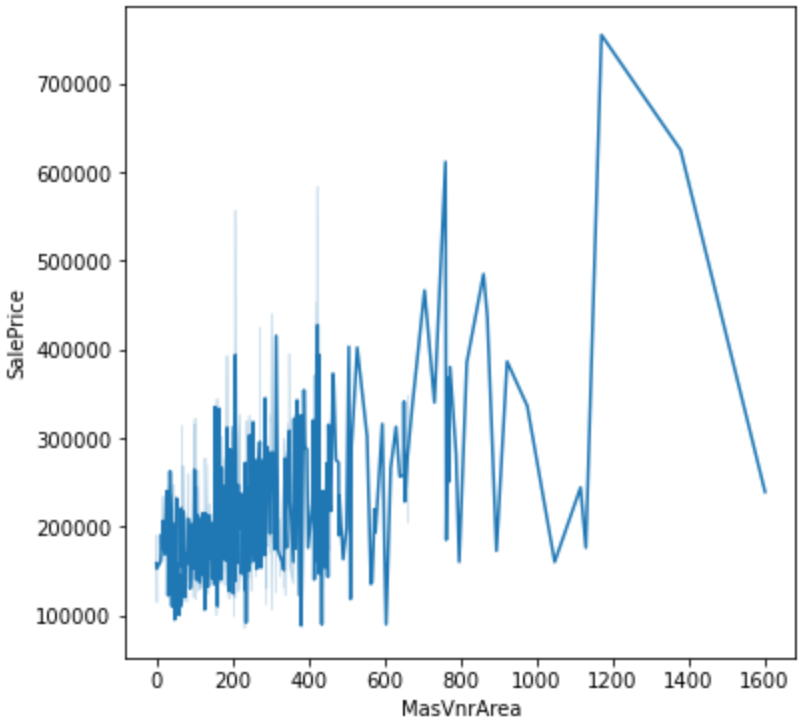

260000
240000
220000
200000
180000
160000
140000
120000
100000
SalePrice
Gable
Flat
Hip
Shed
Gambrel
Mansard
RoofStyle

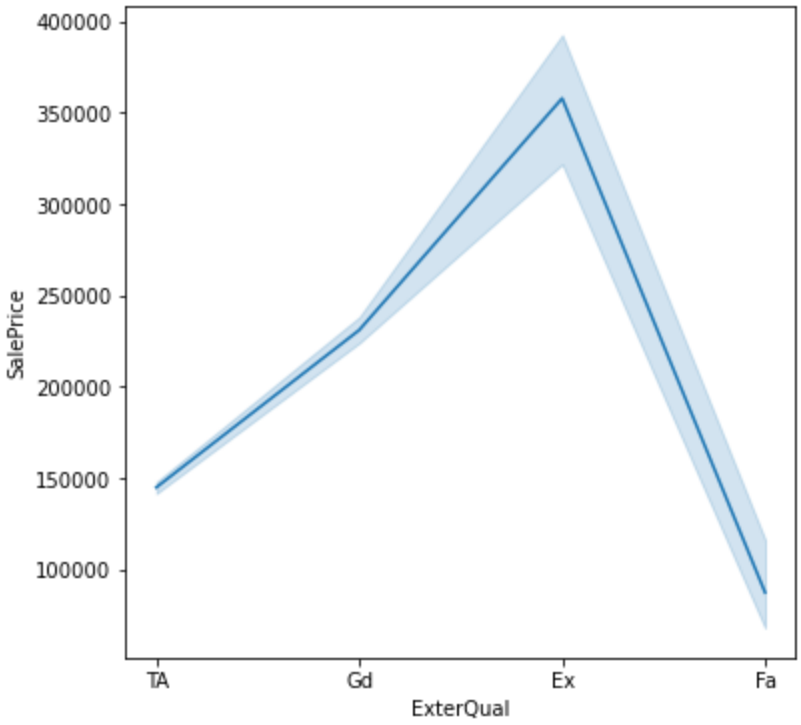

500000
400000
300000
200000
SalePrice
CompShg
Tar&Grv
WdShngl
WdShake
Roll
ClyTile
Metal
Membran
RoofMatl

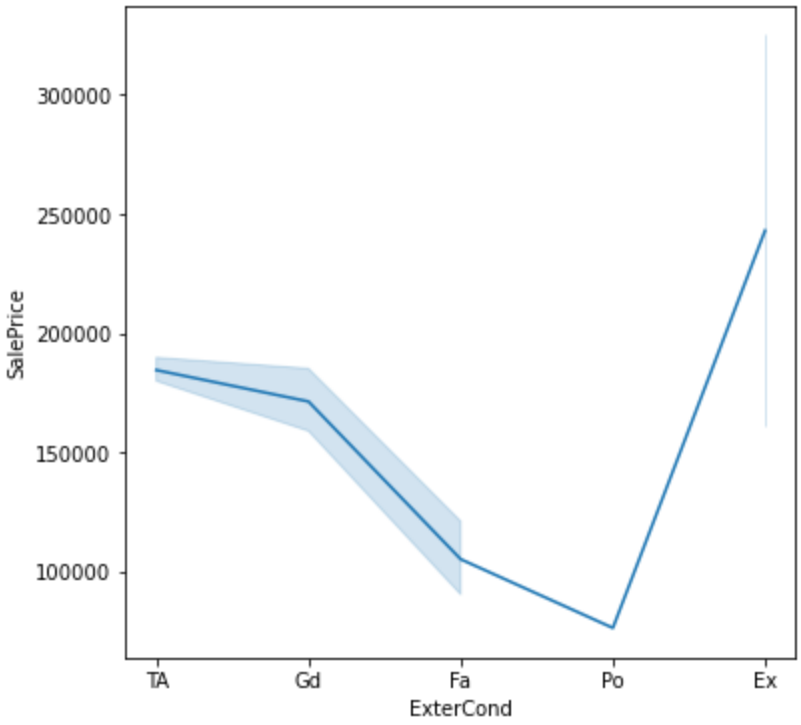

SalePrice
250000
200000
150000
100000
50000
Exterior1st

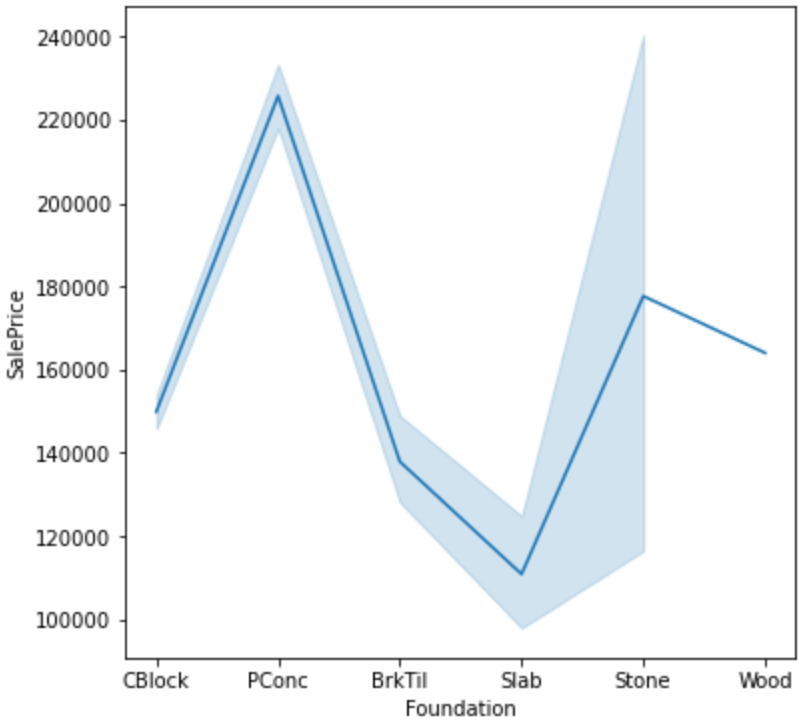

SalePrice
400000
350000
300000
250000
200000
150000
100000
Exterior2nd

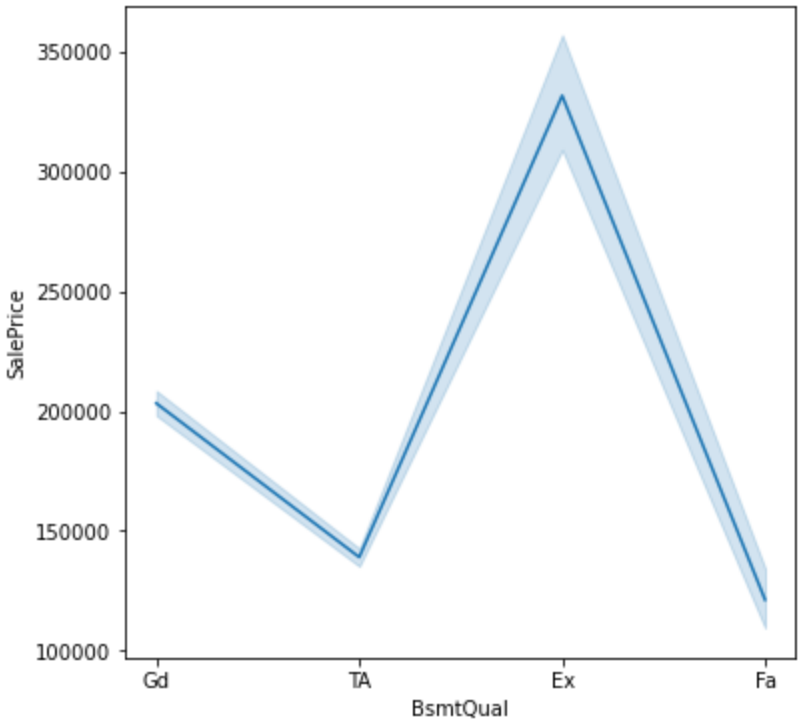

280000
260000
240000
220000
200000
180000
160000
140000
120000
SalePrice
None
BrkFace
Stone
BrkCmn
MasVnrType

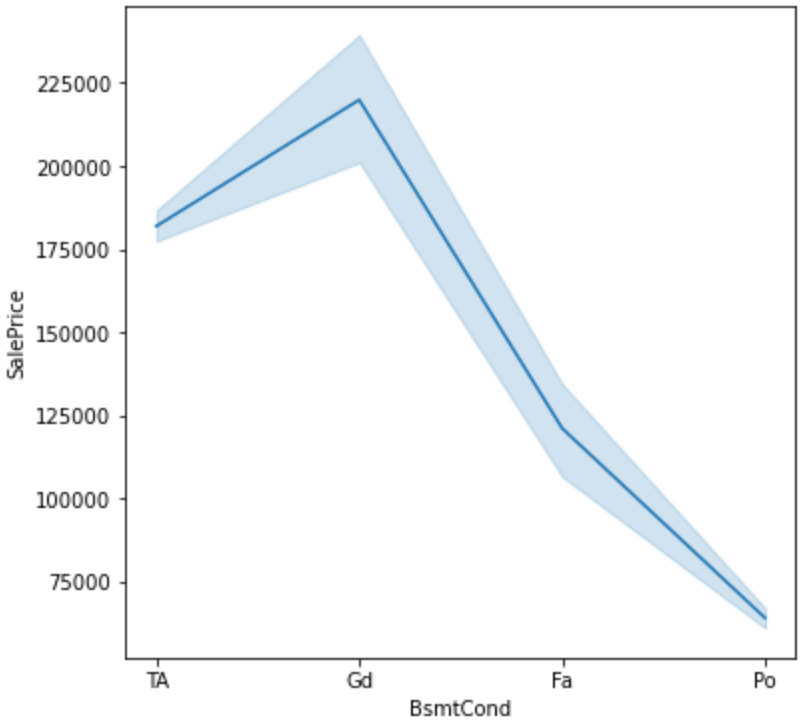

700000
600000
500000
400000
300000
200000
100000
SalePrice
0
200
400
600
800
1000
1200
1400
1600
MasVnrArea

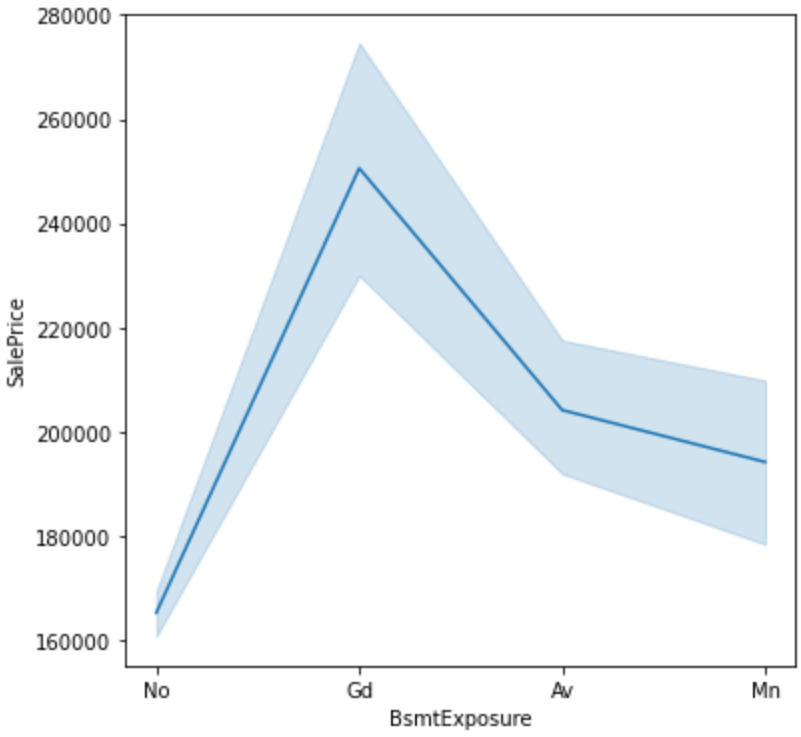

400000
350000
300000
250000
200000
150000
100000
SalePrice
TA
Gd
Ex
Fa
ExterQual

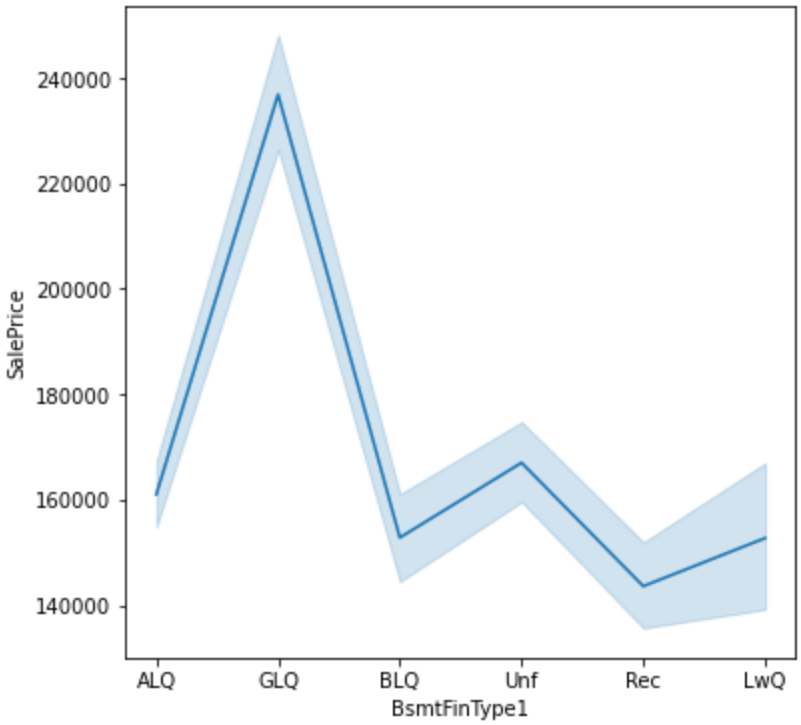

300000
250000
200000
150000
100000
SalePrice
TA
Gd
Fa
Po
Ex
ExterCond

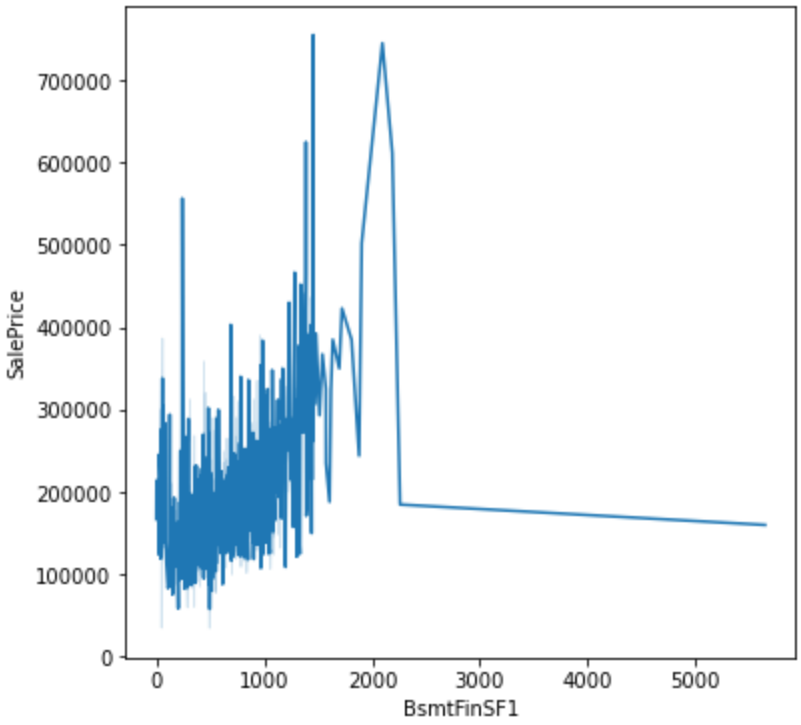

240000
220000
200000
180000
160000
140000
120000
100000
SalePrice
CBlock
PConc
BrkTil
Slab
Stone
Wood
Foundation

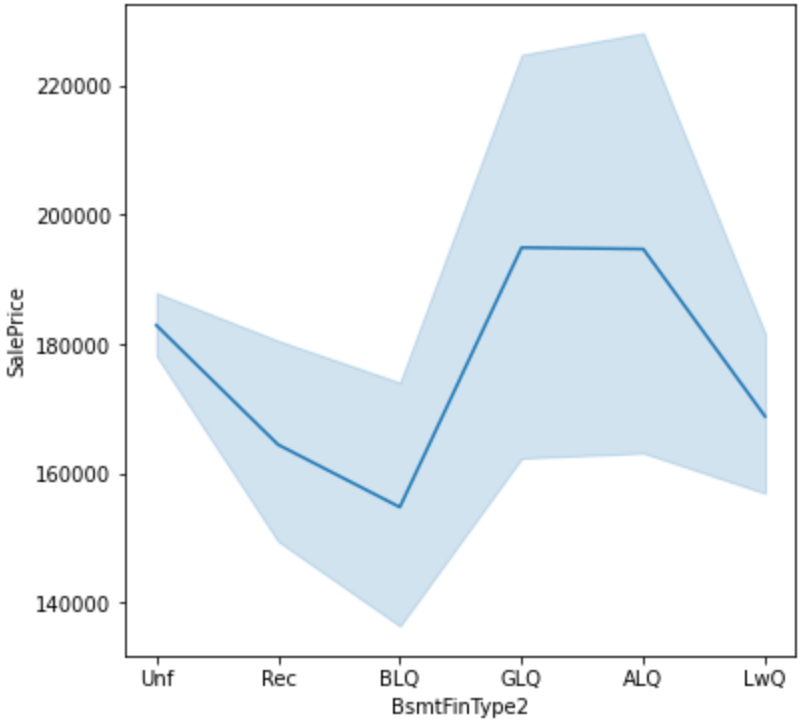

350000
300000
250000
200000
150000
100000
SalePrice
Gd
TA
Ex
Fa
BsmtQual

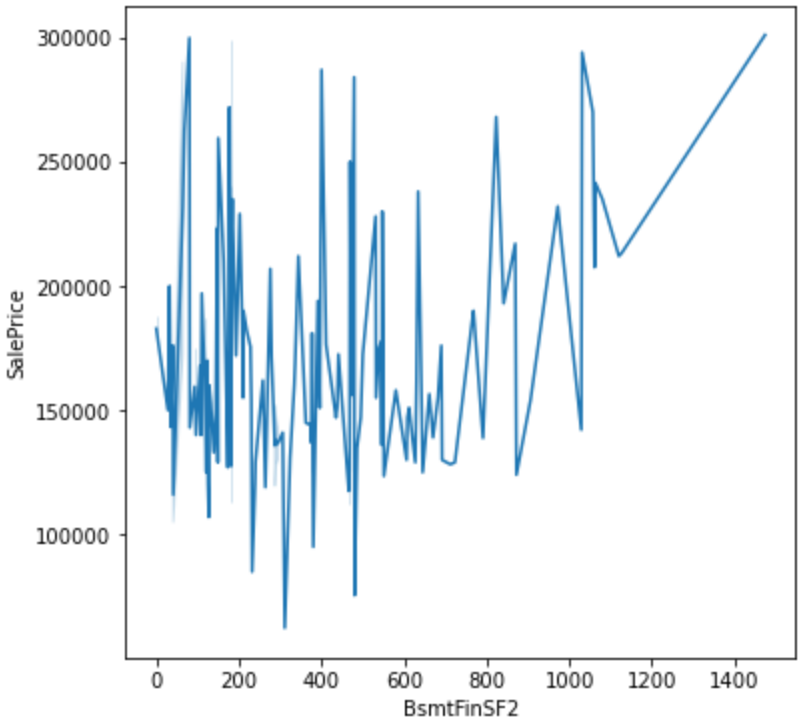

225000
200000
175000
150000
125000
100000
75000
SalePrice
TA
Gd
Fa
Po
BsmtCond

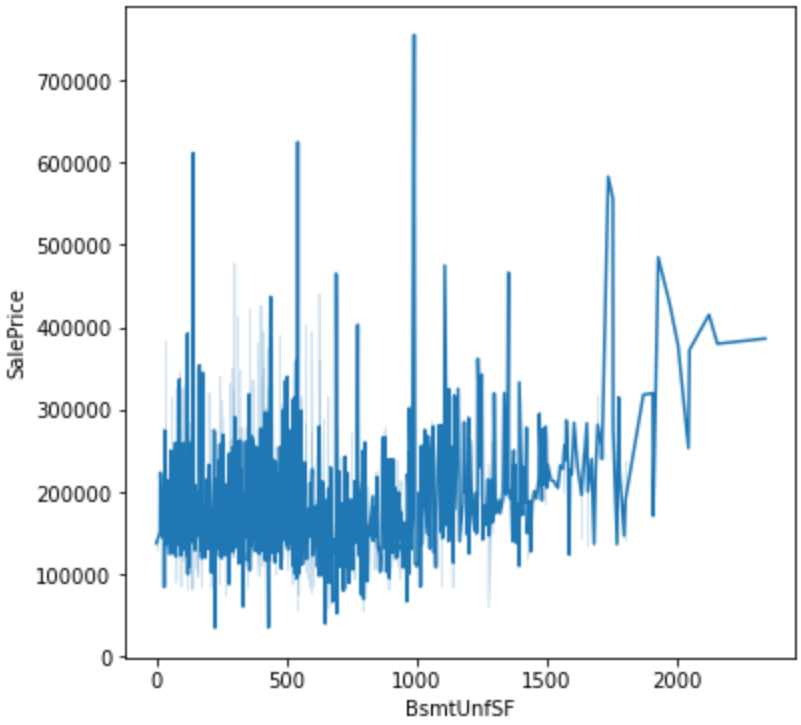

280000
260000
240000
220000
200000
180000
160000
SalePrice
No
Gd
Av
Mn
BsmtExposure

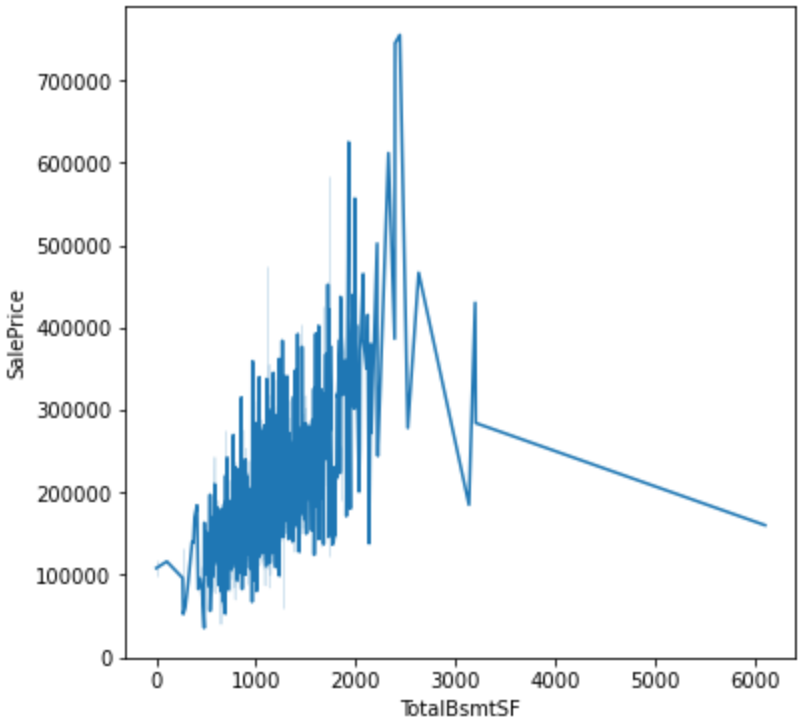

240000
220000
200000
180000
160000
140000
SalePrice
ALQ
GLQ
BLQ
Unf
Rec
LwQ
BsmtFinType1

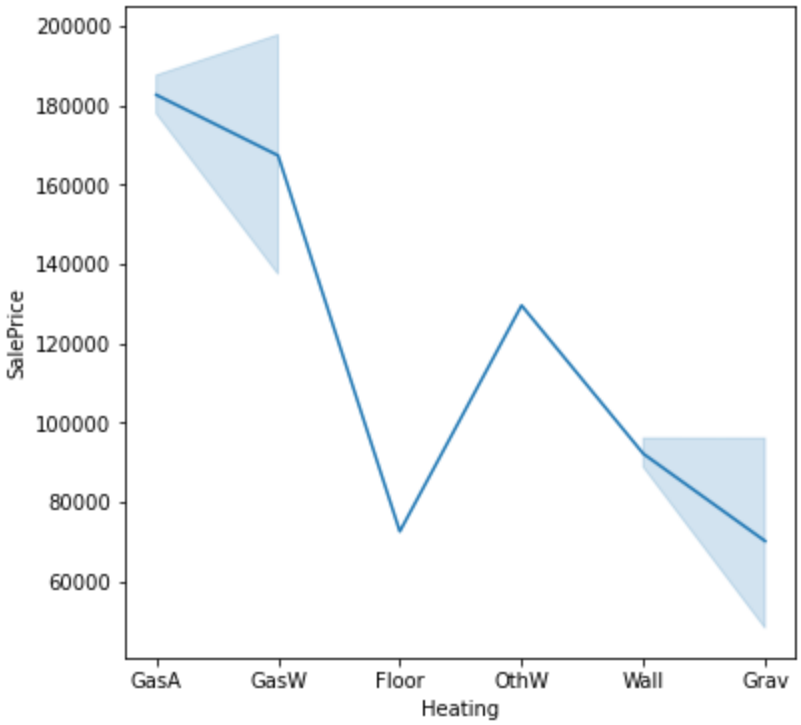

700000
600000
500000
400000
300000
200000
100000
0
SalePrice
0
1000
2000
3000
4000
5000
BsmtFinSF1

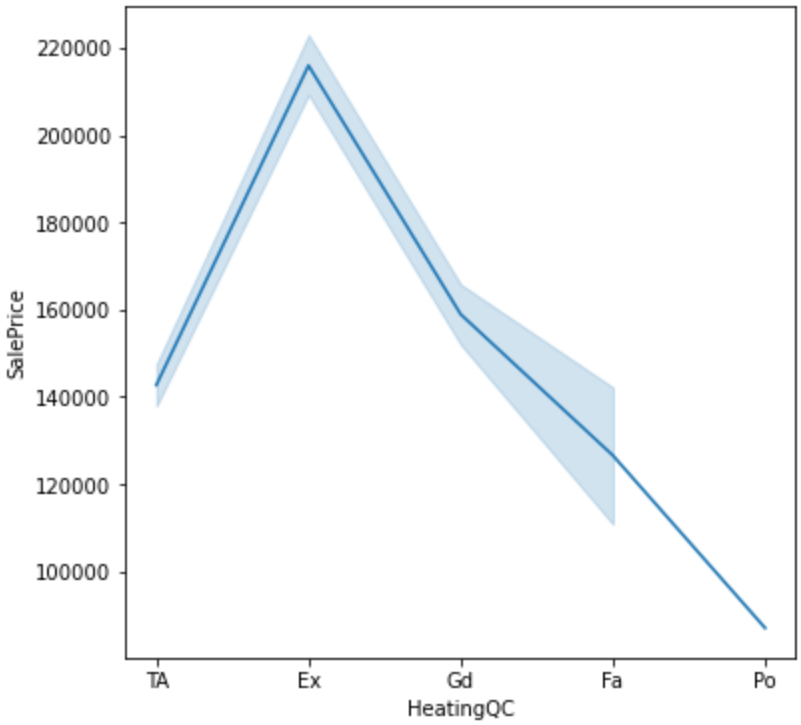

220000
200000
180000
160000
140000
SalePrice
Unf
Rec
BLQ
GLQ
ALQ
LwQ
BsmtFinType2

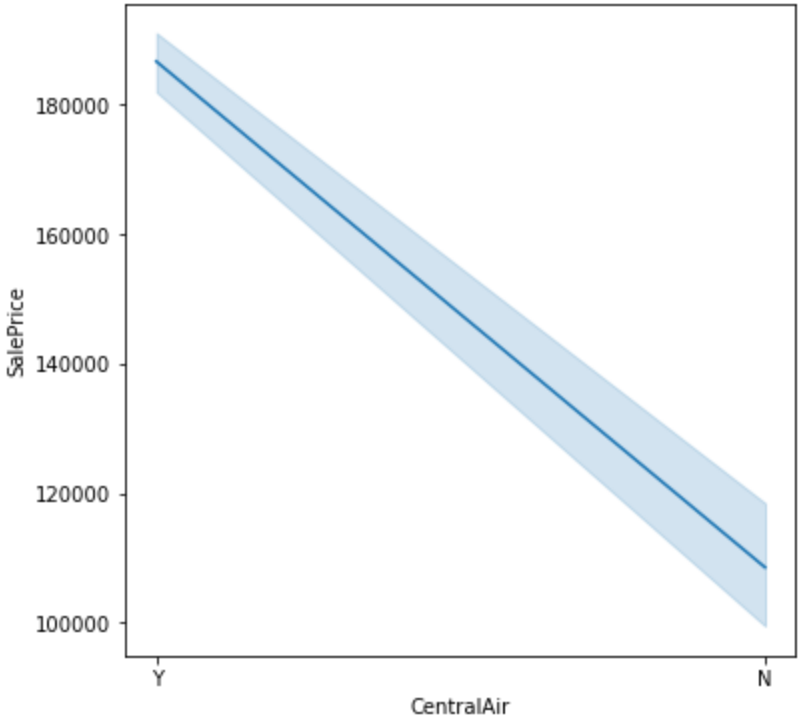

300000
250000
200000
150000
100000
SalePrice
0
200
400
600
800
1000
1200
1400
BsmtFinSF2

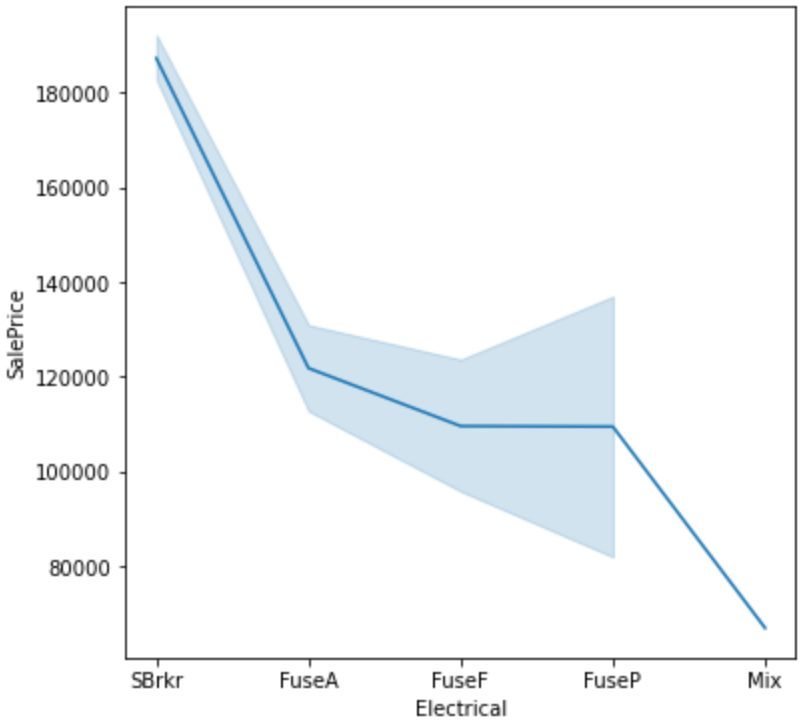

SalePrice
BsmtUnfSF
0
100000
200000
300000
400000
500000
600000
700000
0
500
1000
1500
2000

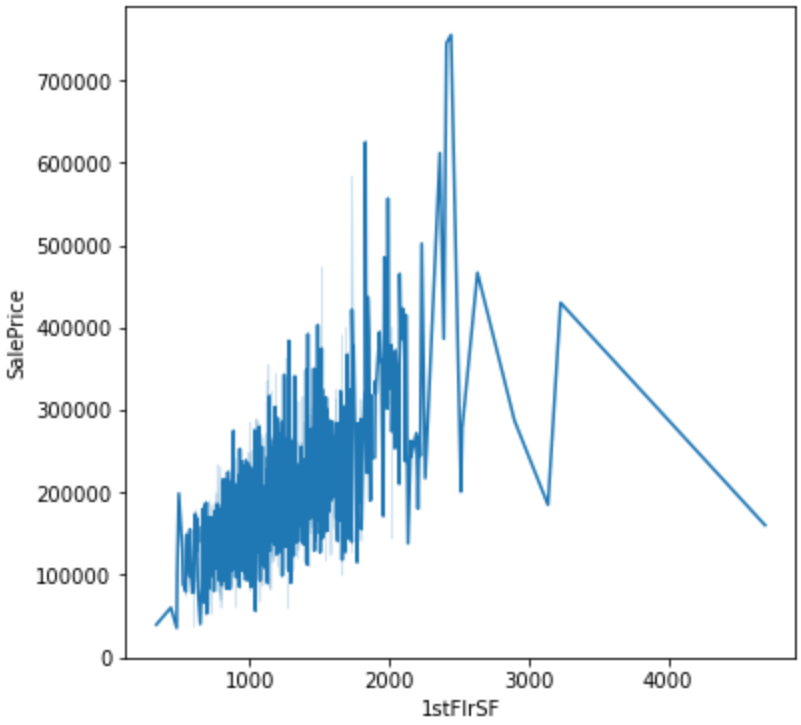

SalePrice
TotalBsmtSF
0
100000
200000
300000
400000
500000
600000
700000
0
1000
2000
3000
4000
5000
6000

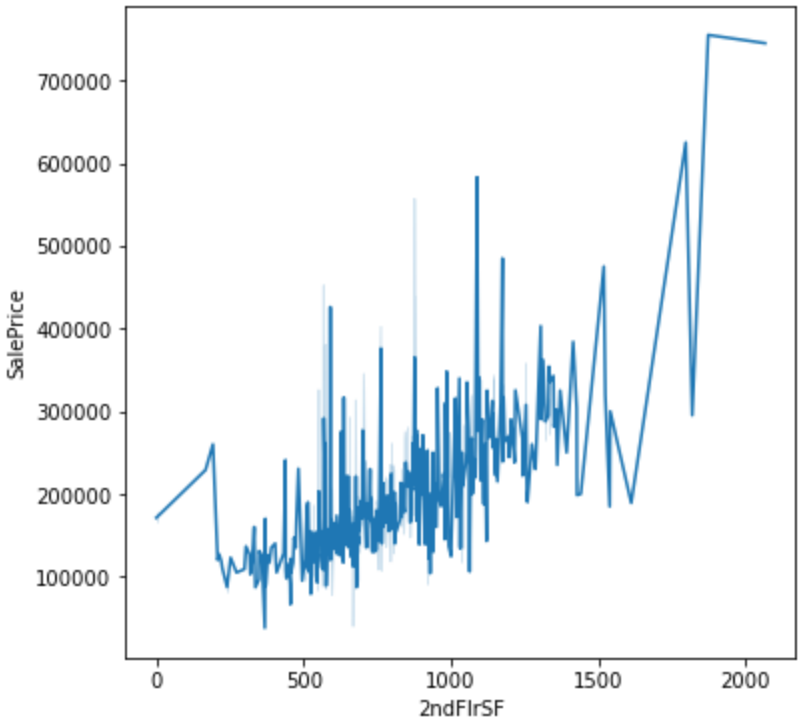

200000
180000
160000
140000
120000
100000
80000
60000
SalePrice
GasA
GasW
Floor
OthW
Wall
Grav
Heating

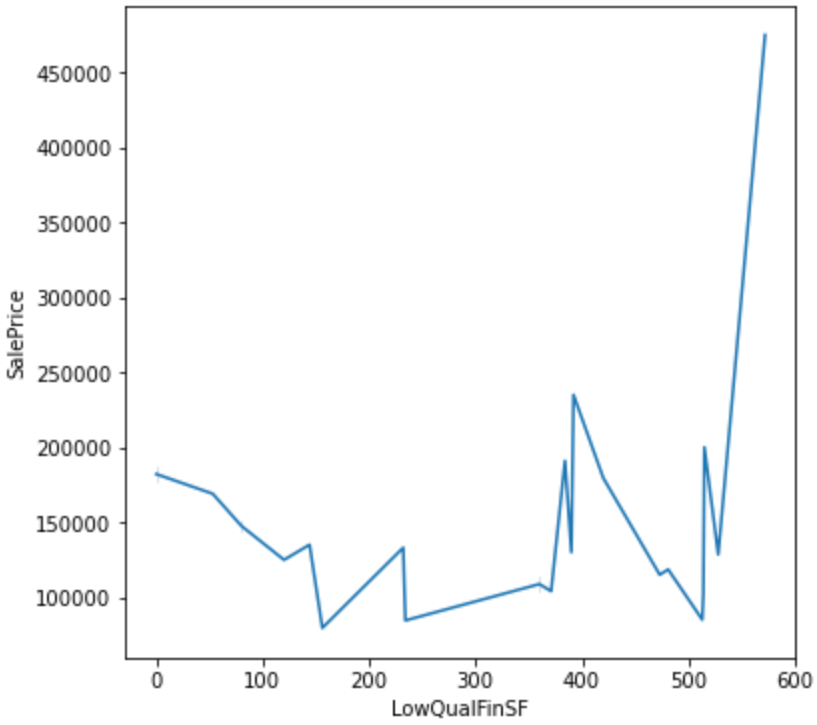

220000
200000
180000
160000
140000
120000
100000
SalePrice
TA
Ex
Gd
Fa
Po
HeatingQC

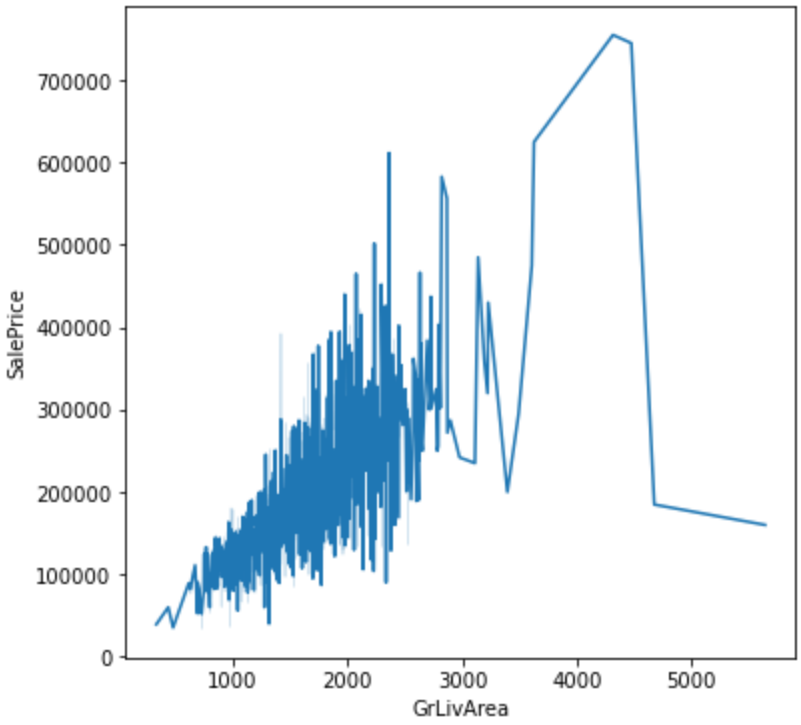

180000
160000
140000
120000
100000
SalePrice
Y
N
CentralAir

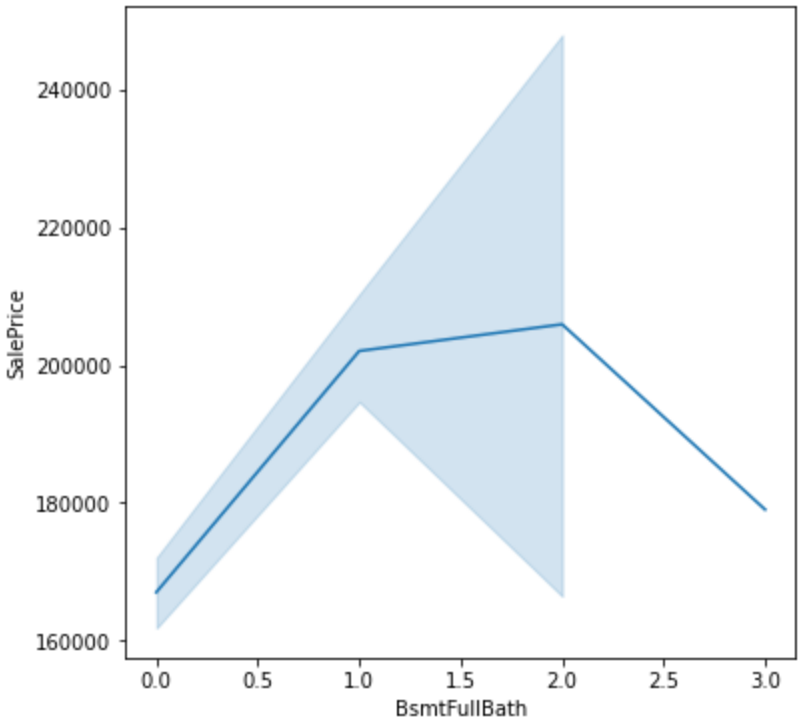

180000
160000
140000
120000
100000
80000
SalePrice
SBrkr
FuseA
FuseF
FuseP
Mix
Electrical

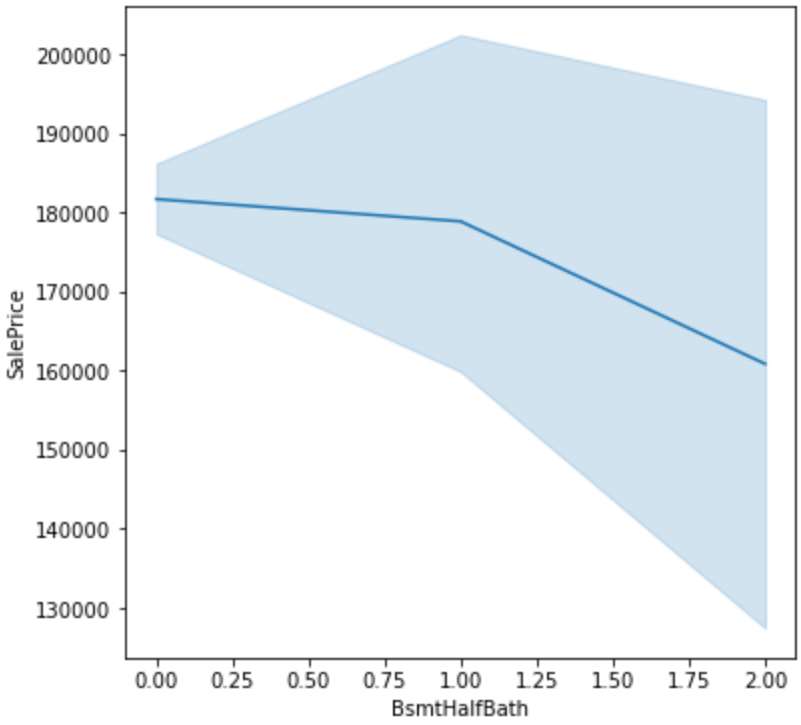

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
1000
2000
3000
4000
1stFlrSF

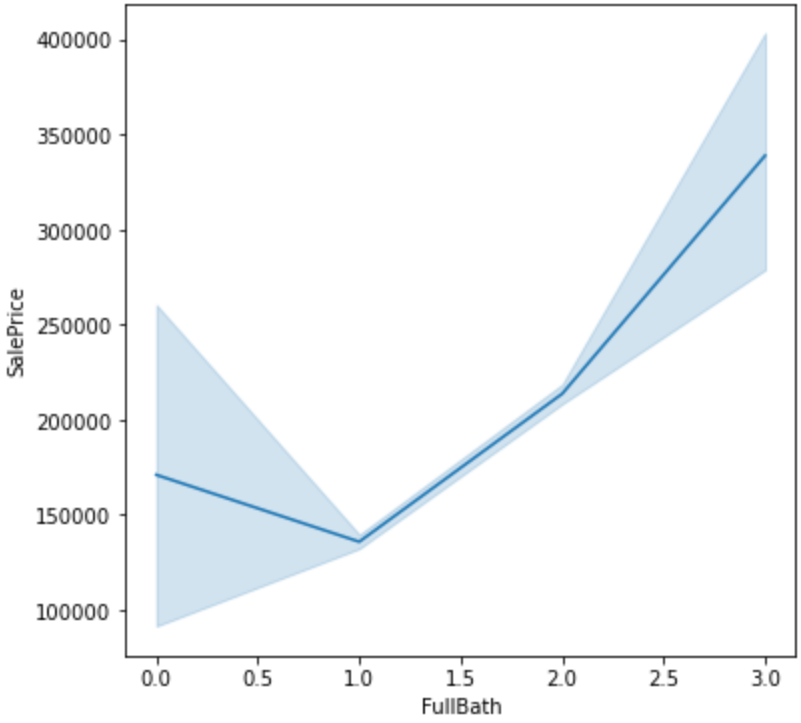

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
500
1000
1500
2000
2ndFlrSF

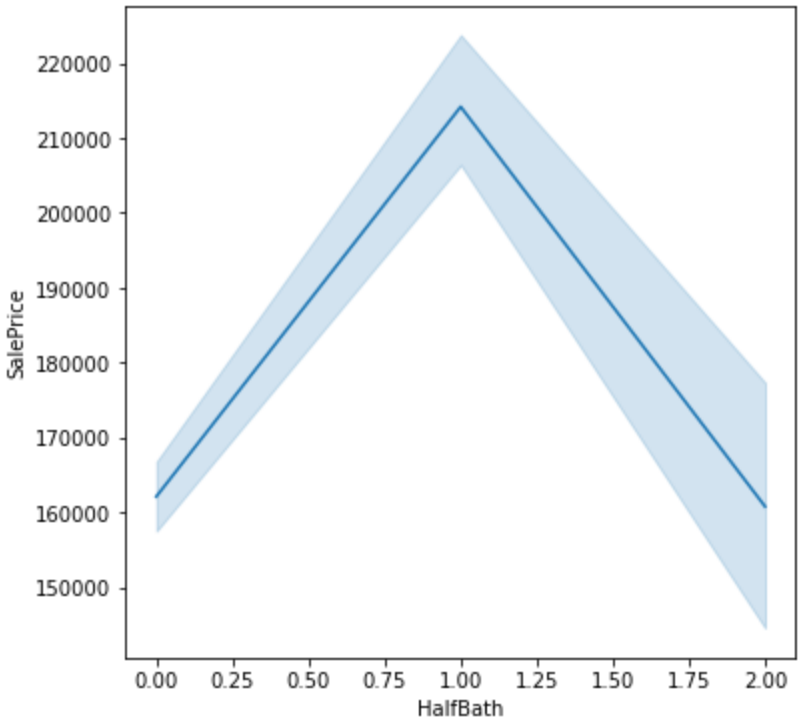

SalePrice
450000
400000
350000
300000
250000
200000
150000
100000
0
100
200
300
400
500
600
LowQualFinSF

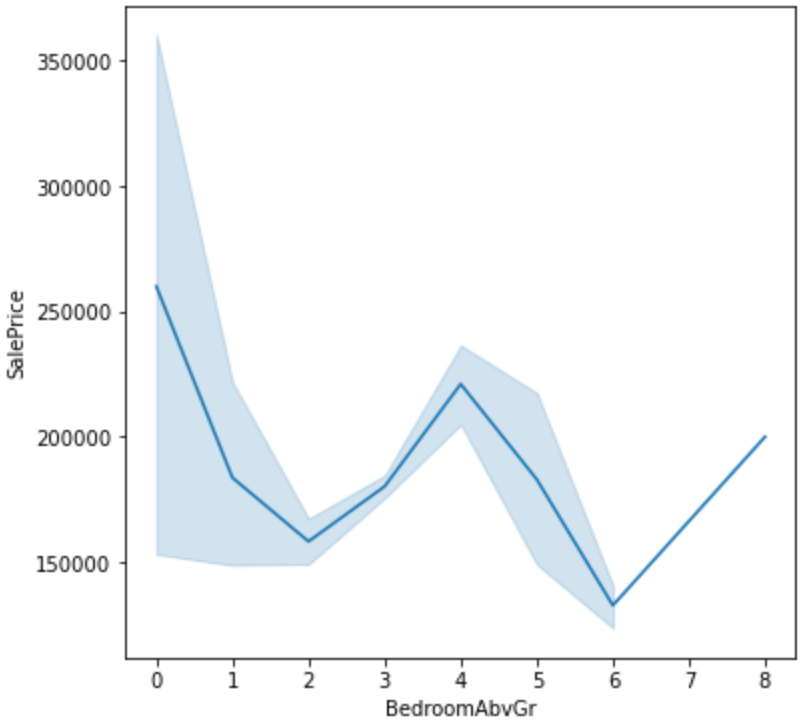

SalePrice
700000
600000
500000
400000
300000
200000
100000
0
1000
2000
3000
4000
5000
GrLivArea

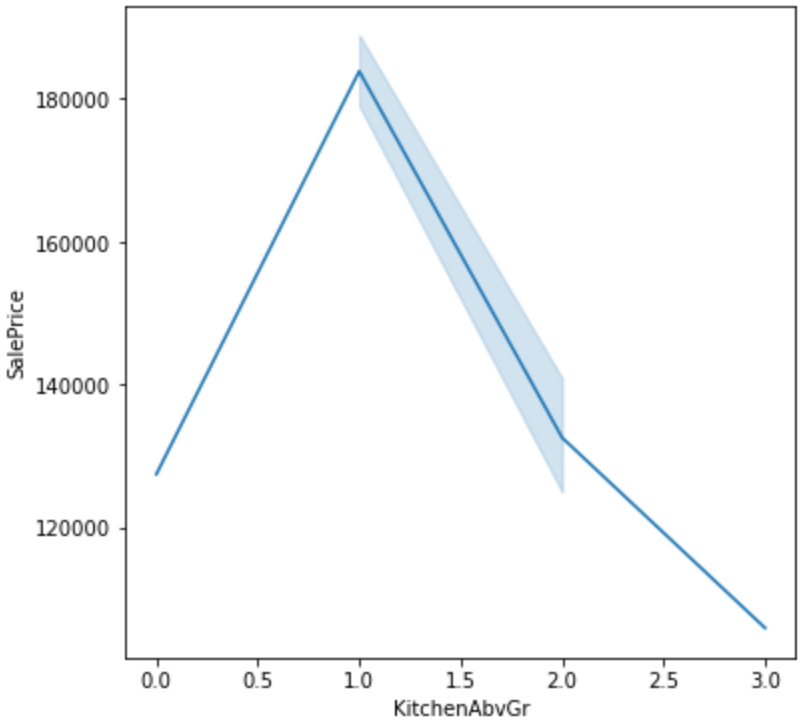

240000
220000
200000
180000
160000
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtFullBath

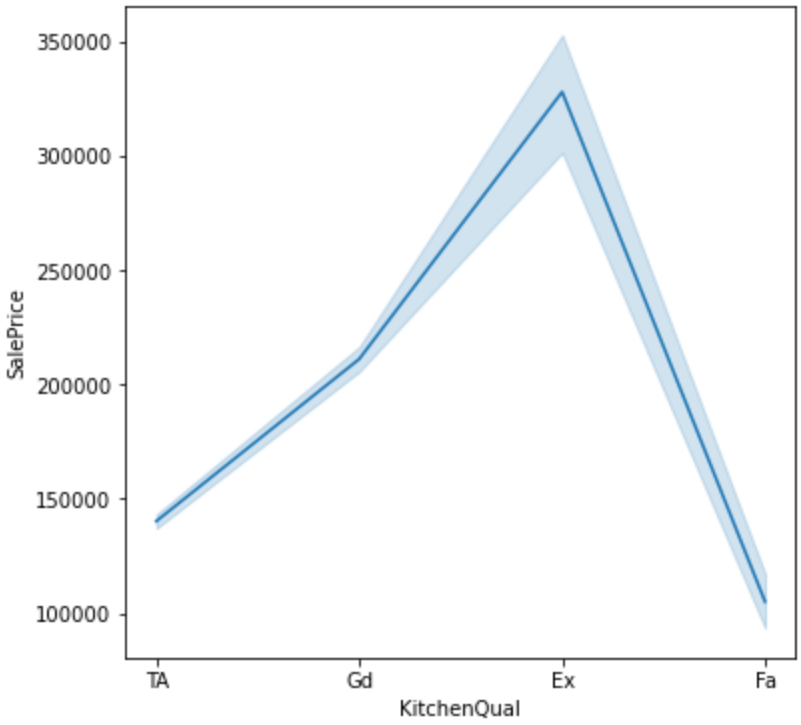

200000
190000
180000
170000
160000
150000
140000
130000
SalePrice
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
BsmtHalfBath

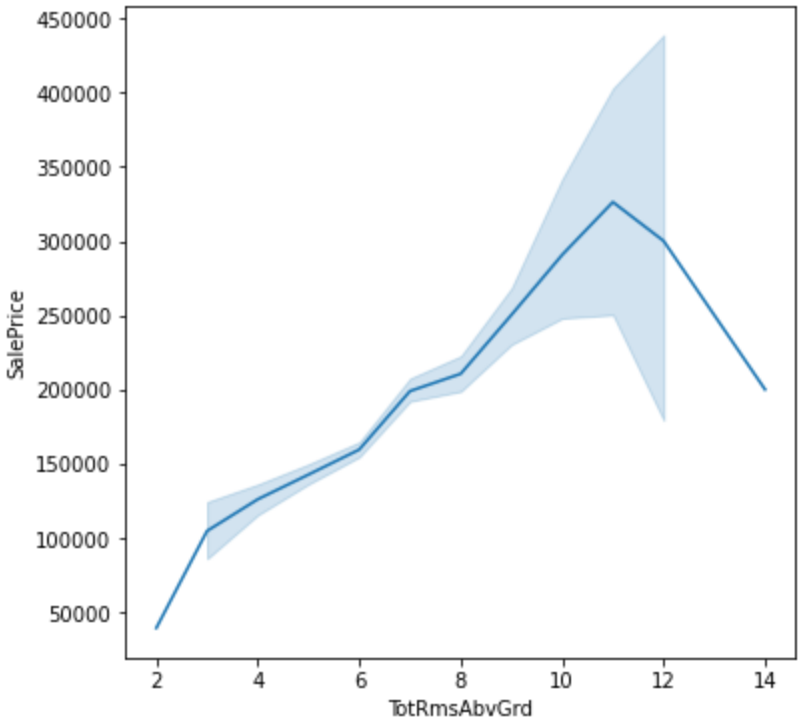

400000
350000
300000
250000
200000
150000
100000
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
FullBath

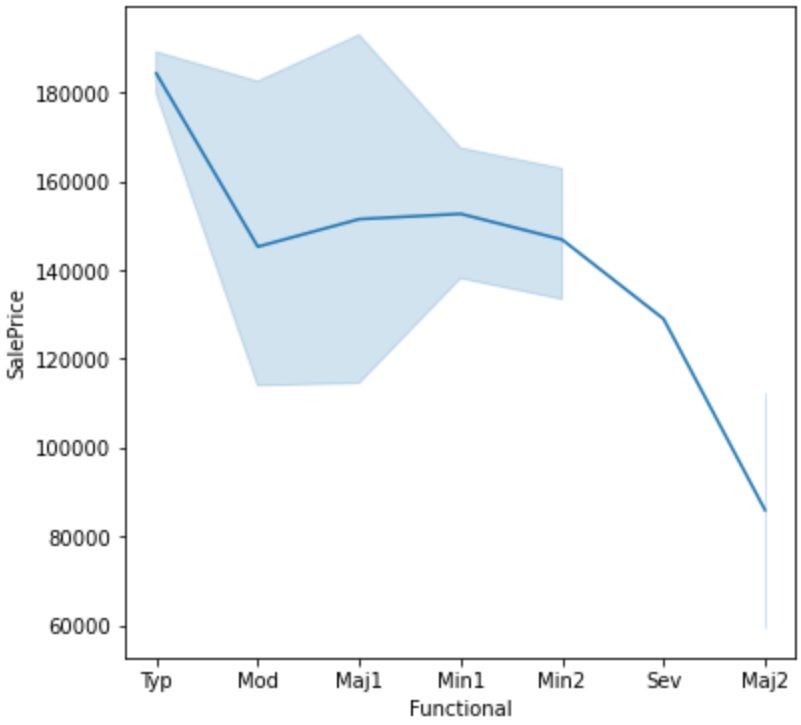

220000
210000
200000
190000
180000
170000
160000
150000
SalePrice
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
HalfBath

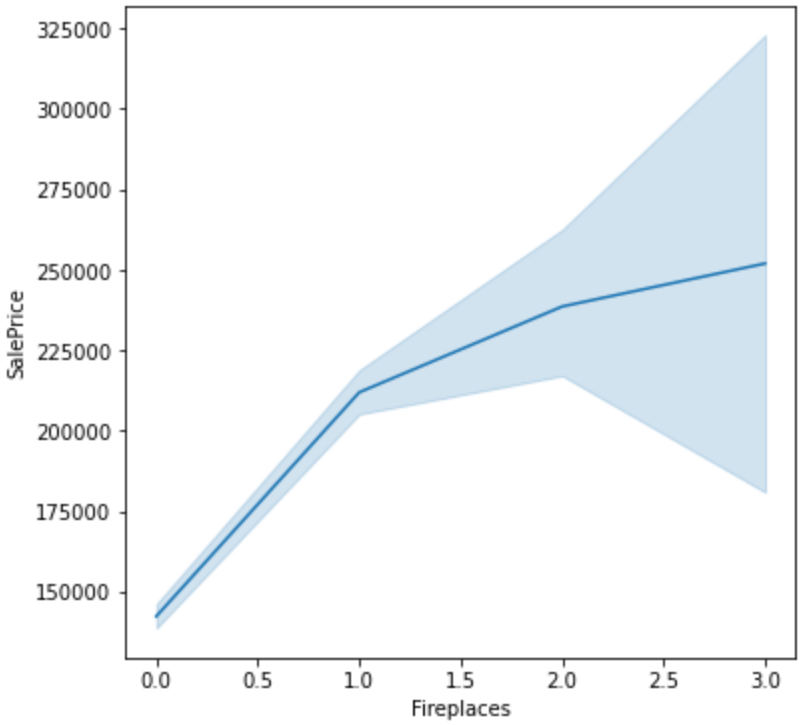

350000
300000
250000
200000
150000
SalePrice
0
1
2
3
4
5
6
7
8
BedroomAbvGr

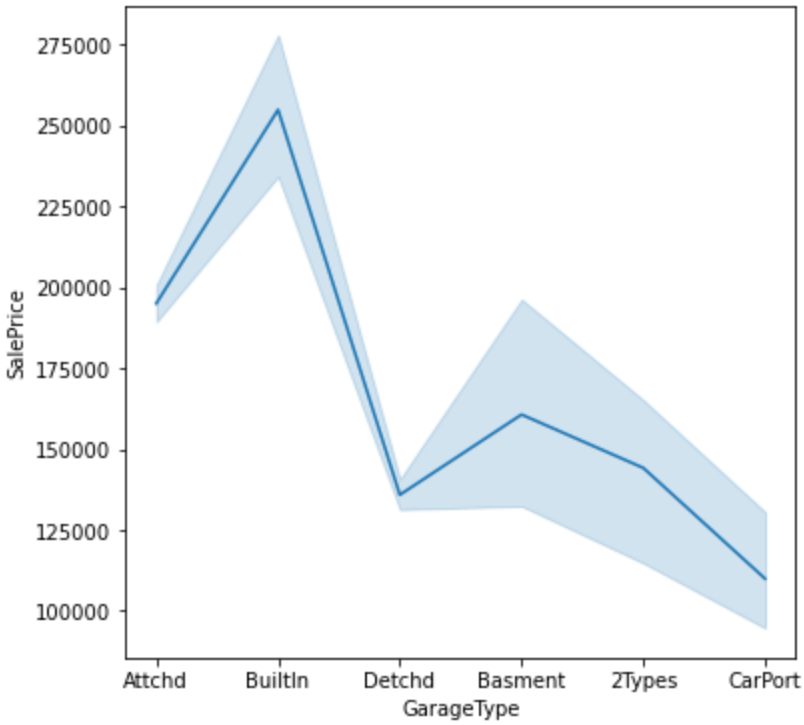

180000
160000
140000
120000
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
KitchenAbvGr

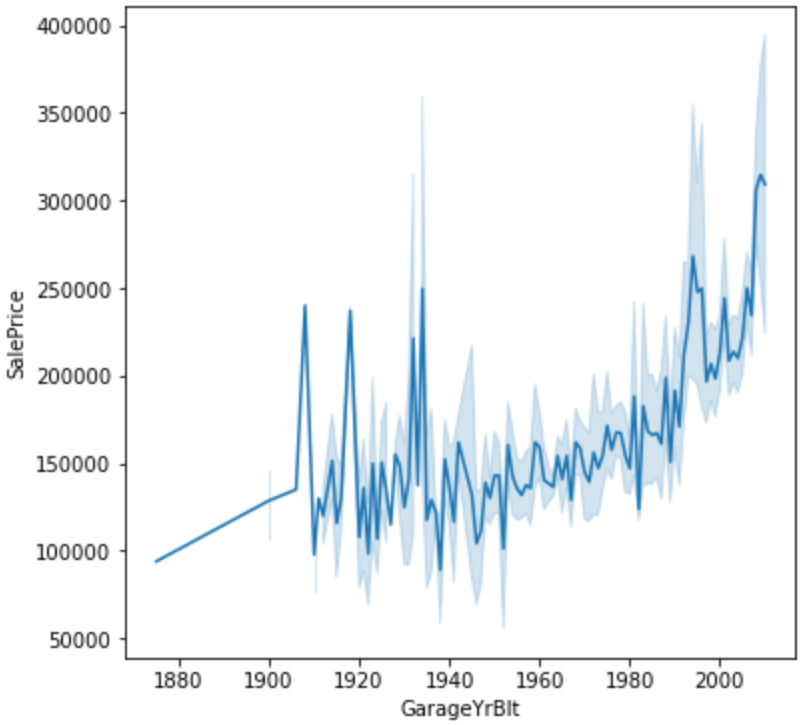

350000
300000
250000
200000
150000
100000
SalePrice
TA
Gd
Ex
Fa
KitchenQual

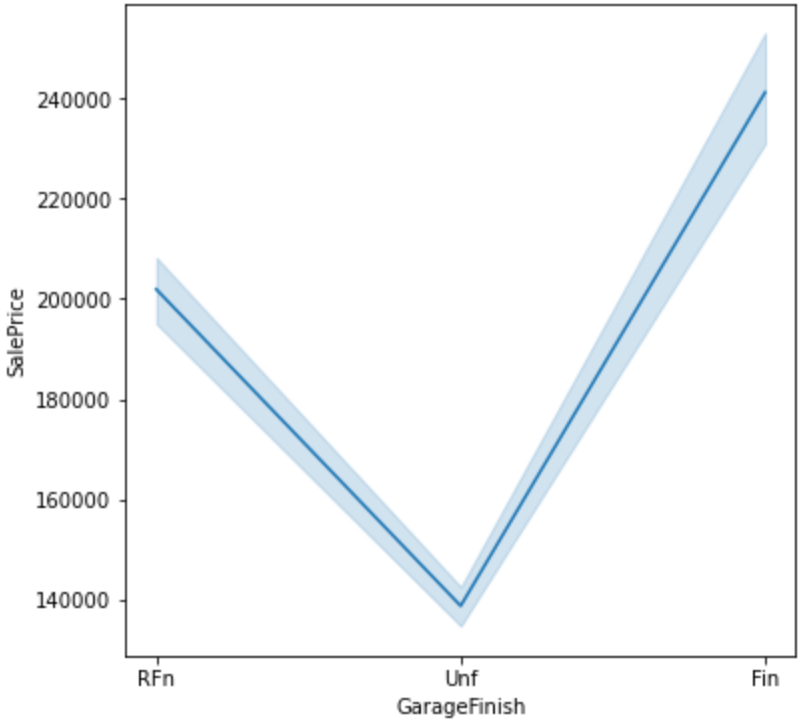

450000
400000
350000
300000
250000
200000
150000
100000
50000
SalePrice
2
4
6
8
10
12
14
TotRmsAbvGrd

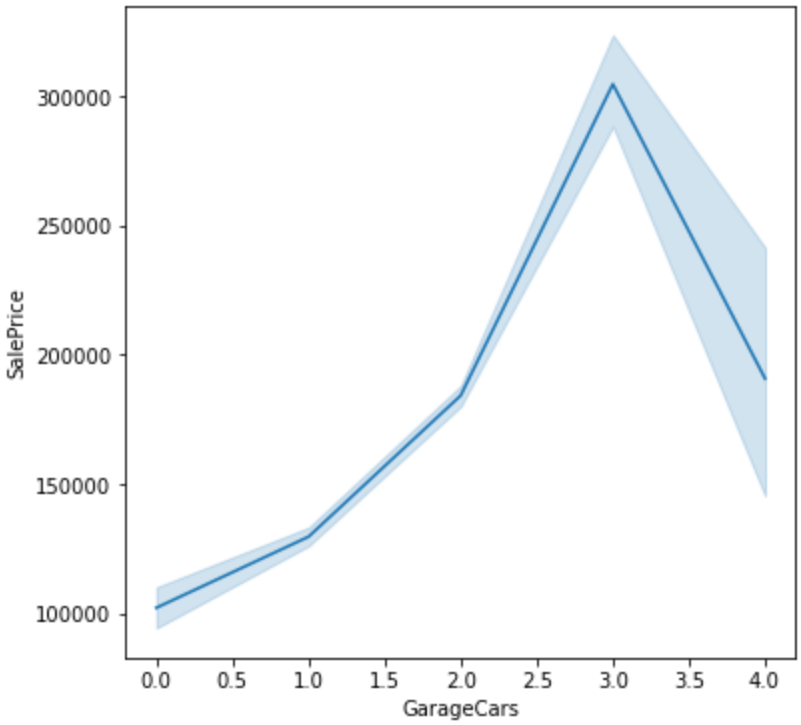

180000
160000
140000
120000
100000
80000
60000
SalePrice
Typ
Mod
Maj1
Min1
Min2
Sev
Maj2
Functional

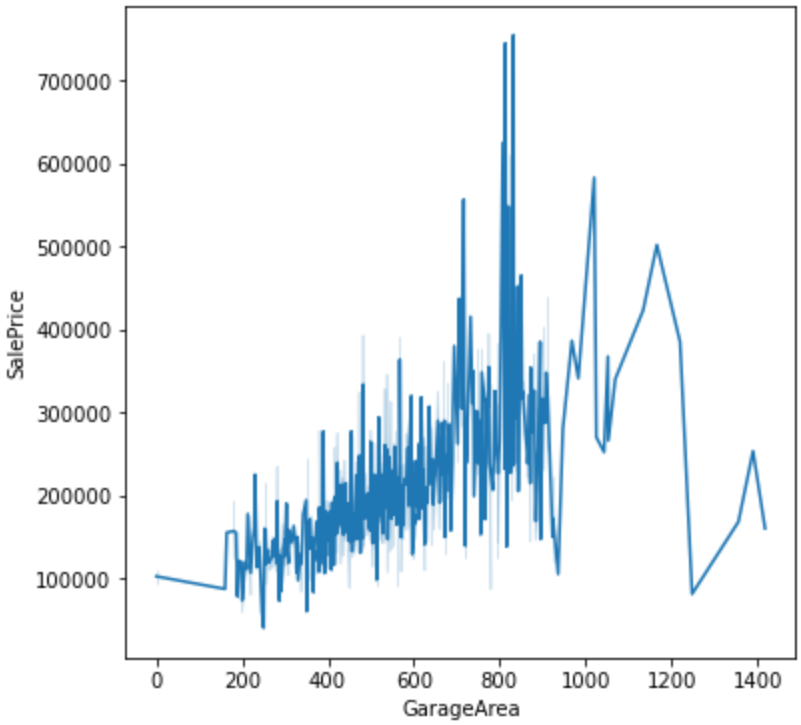

325000
300000
275000
250000
225000
200000
175000
150000
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Fireplaces

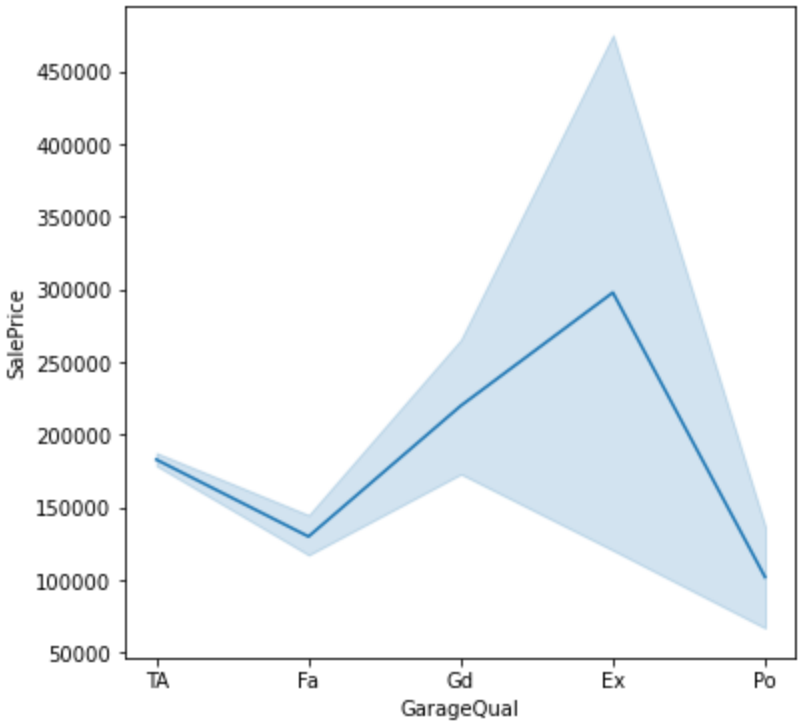

275000
250000
225000
200000
175000
150000
125000
100000
SalePrice
Attchd
BuiltIn
Detchd
Basment
2Types
CarPort
GarageType

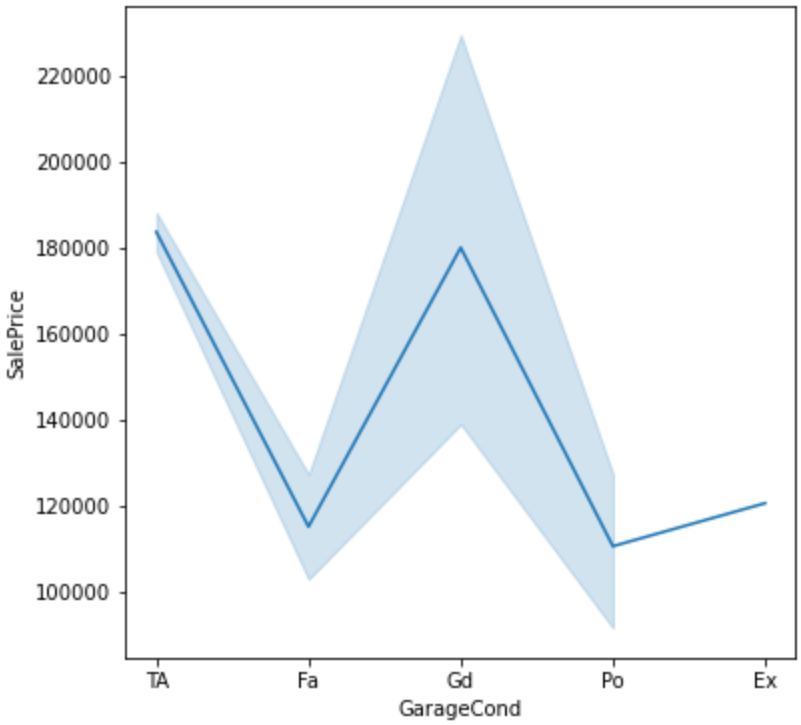

400000
350000
300000
250000
200000
150000
100000
50000
SalePrice
1880
1900
1920
1940
1960
1980
2000
GarageYrBlt

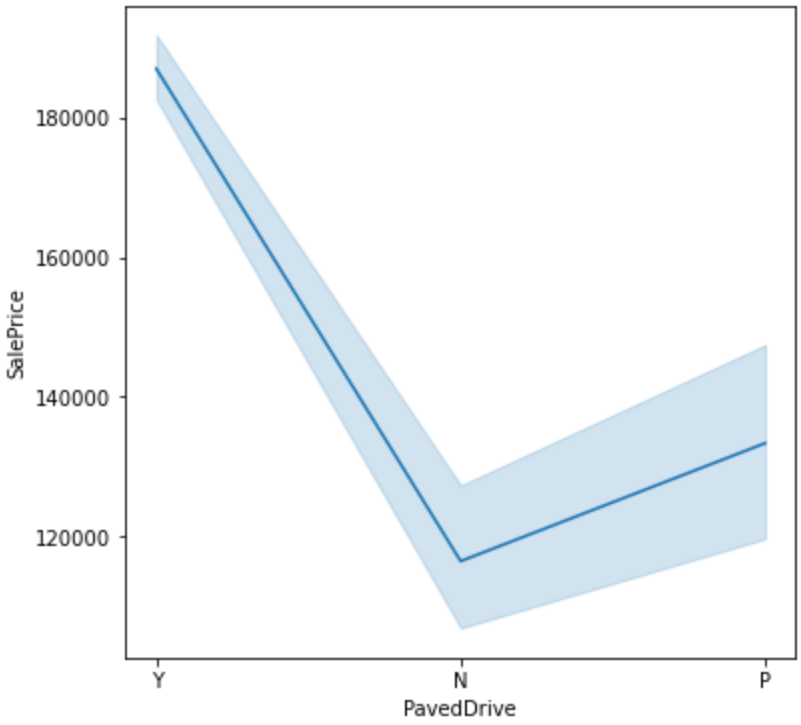

240000
220000
200000
180000
160000
140000
SalePrice
RFn
Unf
Fin
GarageFinish

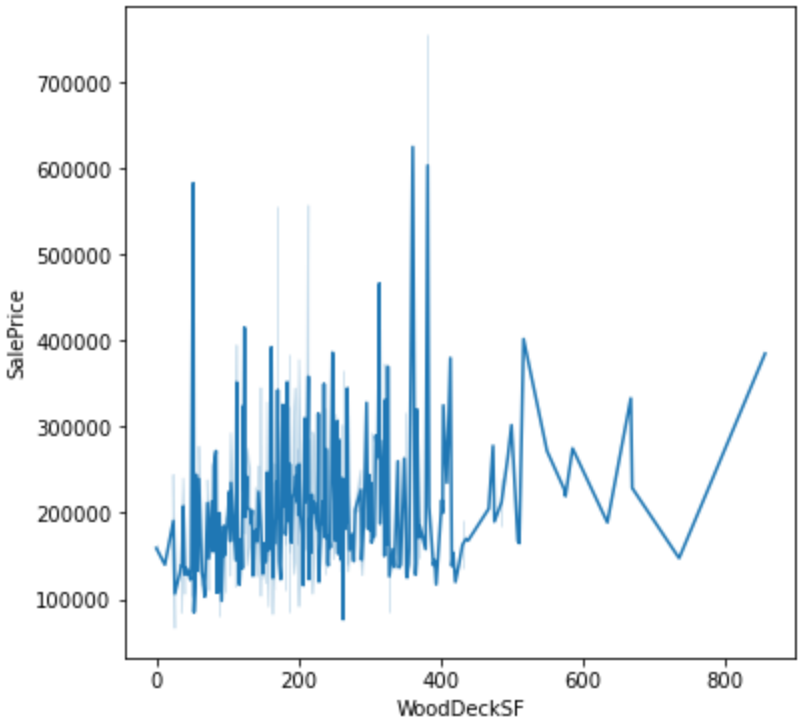

300000
250000
200000
150000
100000
SalePrice
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageCars

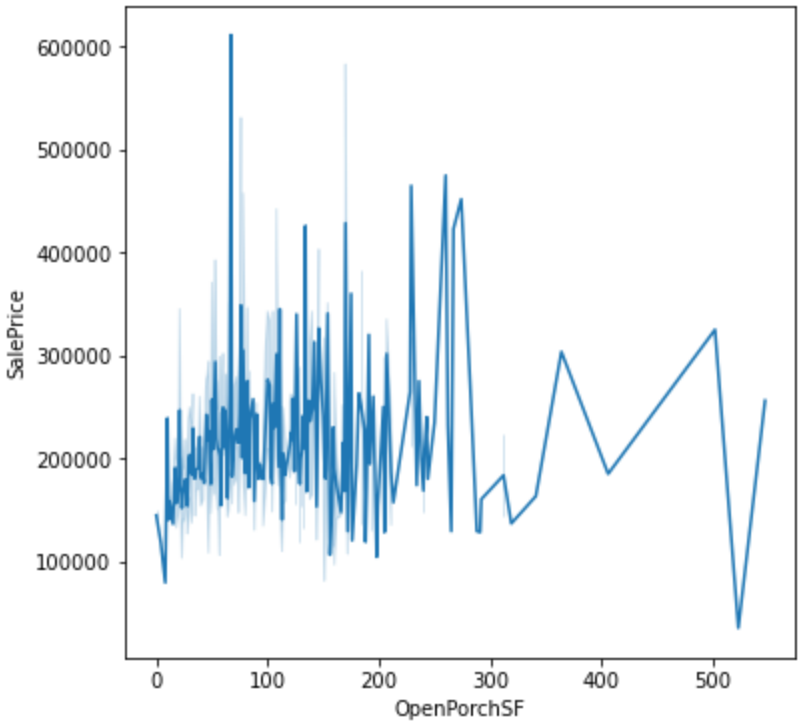

700000
600000
500000
400000
300000
200000
100000
SalePrice
0
200
400
600
800
1000
1200
1400
GarageArea

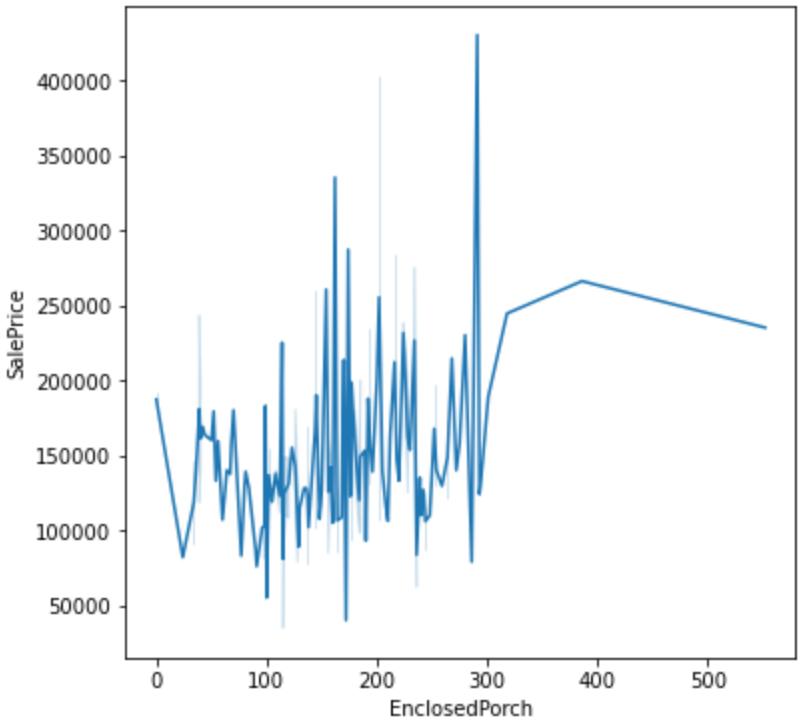

450000
400000
350000
300000
250000
200000
150000
100000
50000
SalePrice
TA
Fa
Gd
Ex
Po
GarageQual

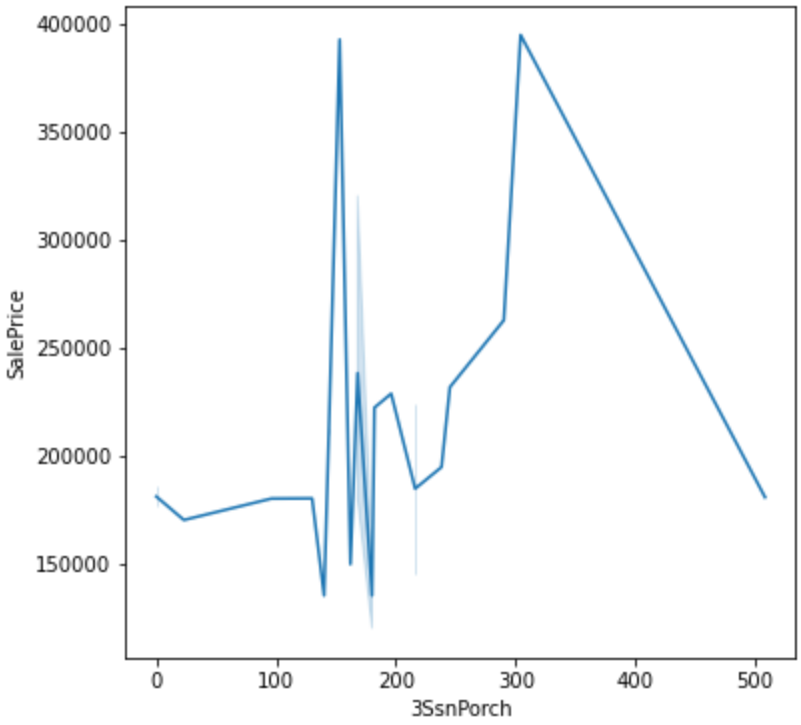

220000
200000
180000
160000
140000
120000
100000
SalePrice
TA
Fa
Gd
Po
Ex
GarageCond

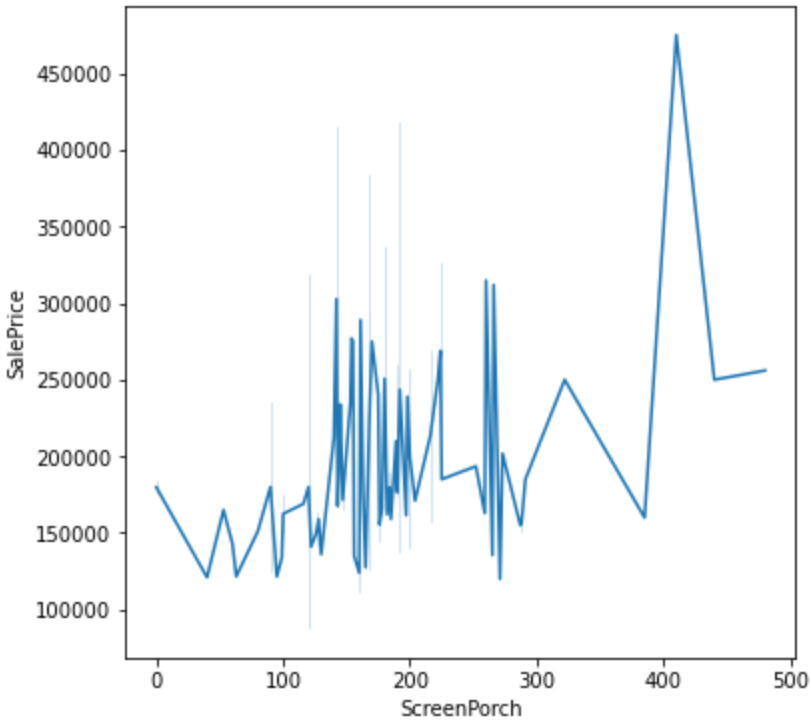

180000
160000
140000
120000
SalePrice
Y
N
P
PavedDrive

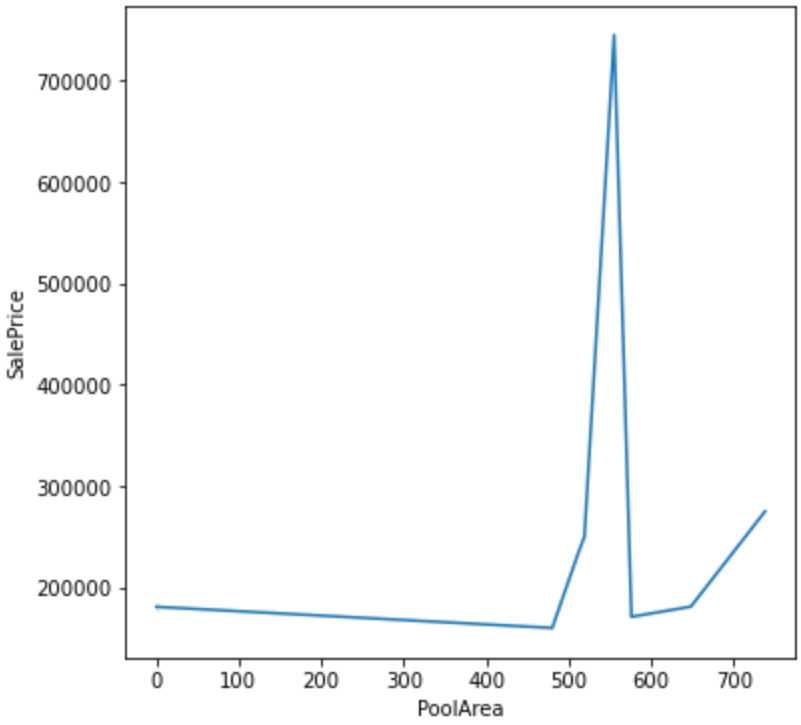

700000
600000
500000
400000
300000
200000
100000
SalePrice
0
200
400
600
800
WoodDeckSF

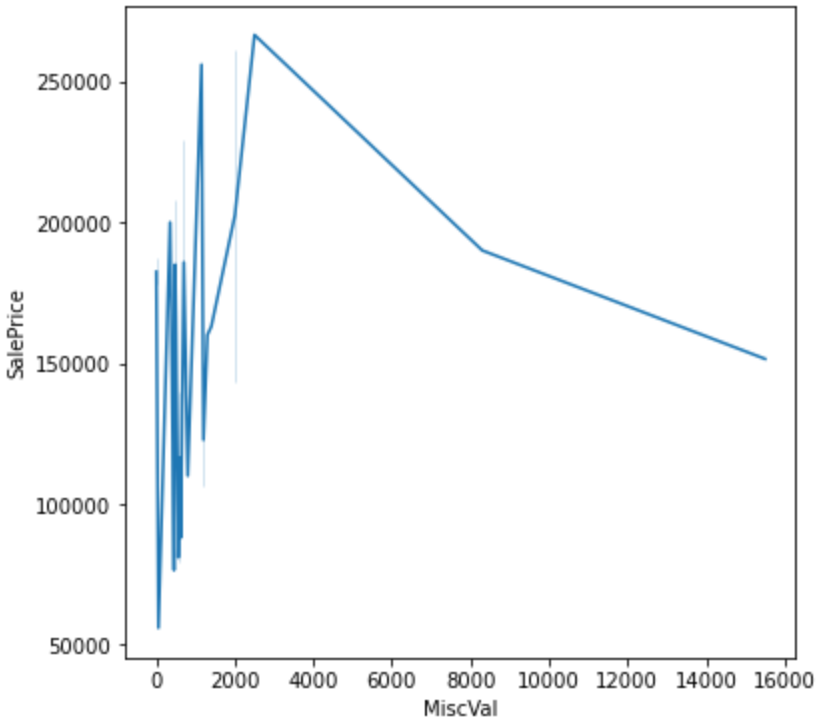

600000
500000
400000
300000
200000
100000
SalePrice
0
100
200
300
400
500
OpenPorchSF

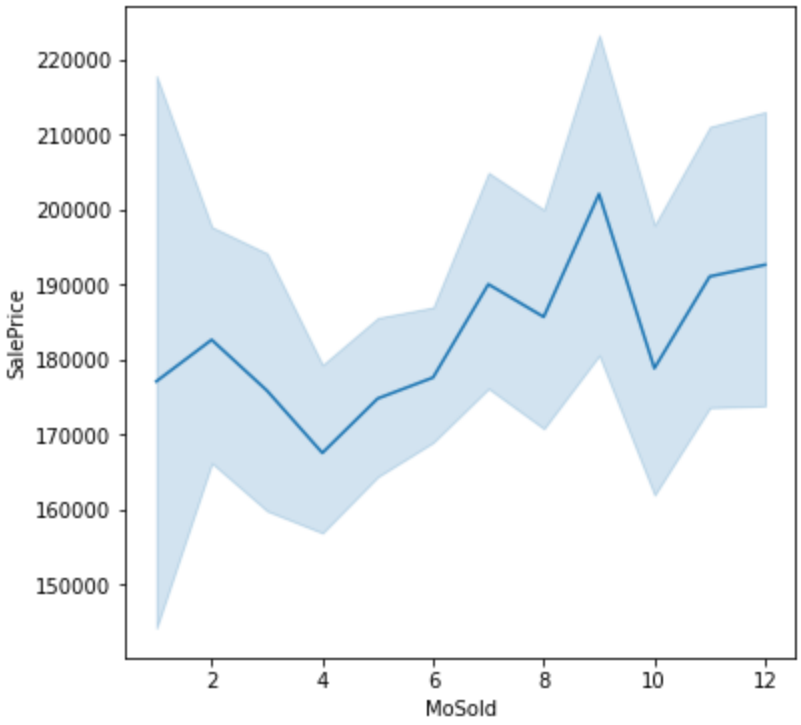

400000
350000
300000
250000
200000
150000
100000
50000
SalePrice
0
100
200
300
400
500
EnclosedPorch

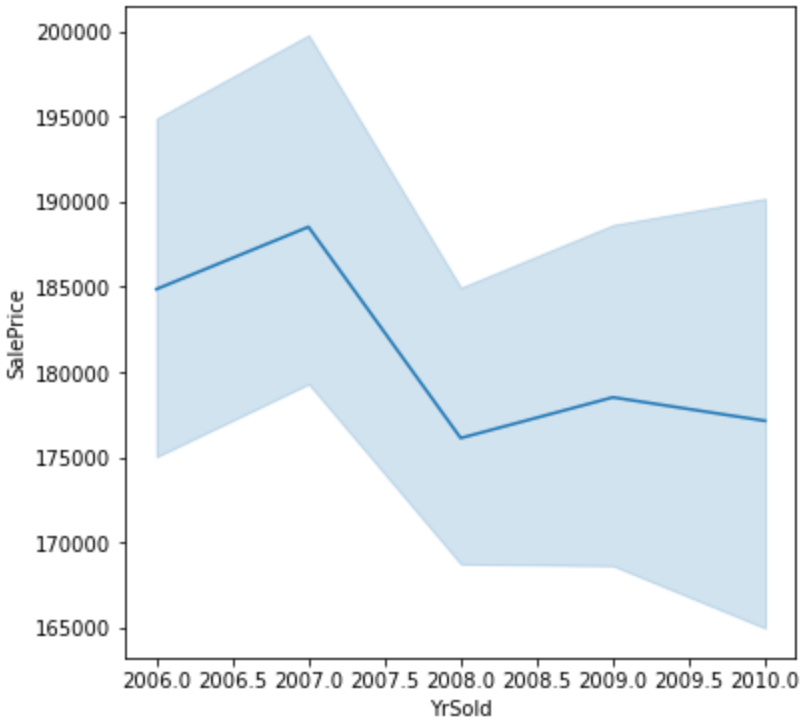

400000
350000
300000
250000
200000
150000
SalePrice
0
100
200
300
400
500
3SsnPorch

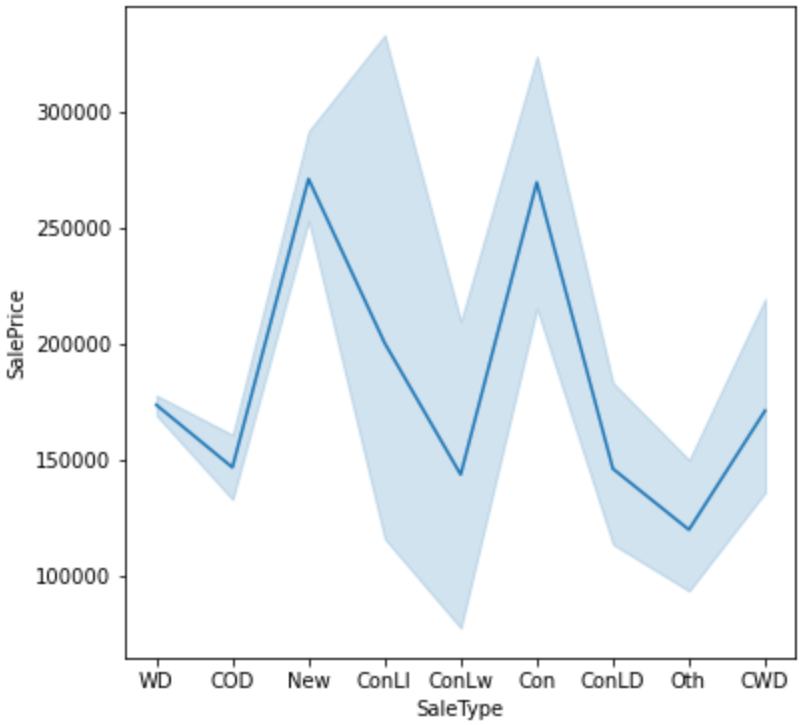

SalePrice
450000
400000
350000
300000
250000
200000
150000
100000
0
100
200
300
400
500
ScreenPorch

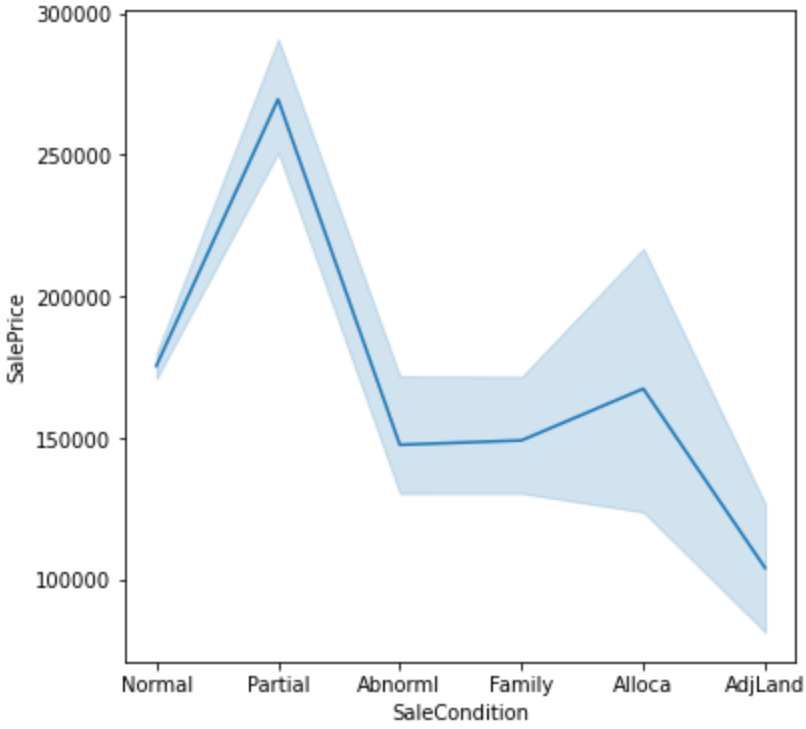

SalePrice
700000
600000
500000
400000
300000
200000
0
100
200
300
400
500
600
700
PoolArea

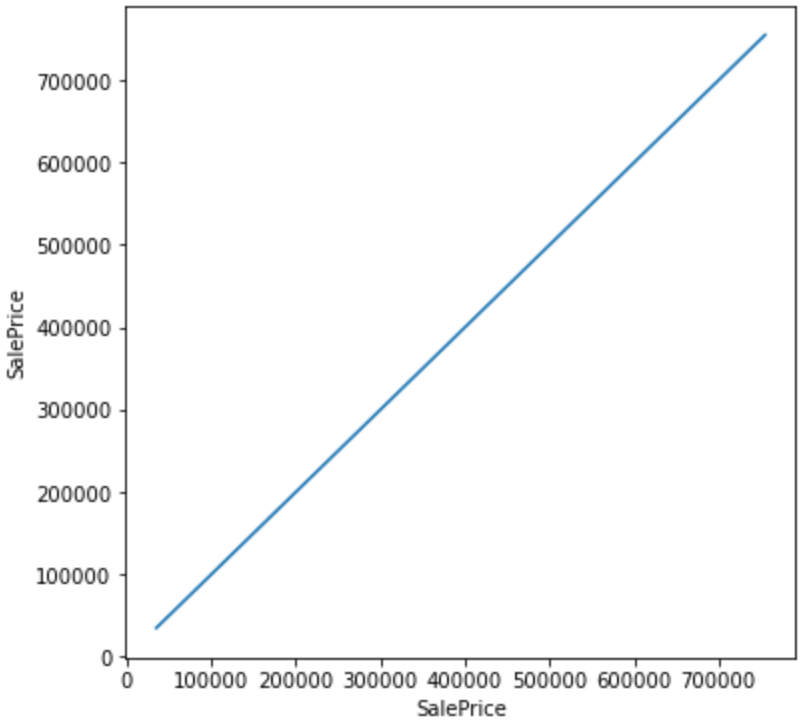

250000
200000
150000
100000
50000
SalePrice
0
2000
4000
6000
8000
10000
12000
14000
16000
MiscVal


220000
210000
200000
190000
180000
170000
160000
150000
SalePrice
2
4
6
8
10
12
MoSold


200000
195000
190000
185000
180000
175000
170000
165000
SalePrice
2006.0 2006.5 2007.0 2007.5 2008.0 2008.5 2009.0 2009.5 2010.0
YrSold


300000
250000
200000
150000
100000
SalePrice
WD
COD
New
ConLI
ConLw
Con
ConLD
Oth
CWD
SaleType





Data Preprocessing

Encoding

Dividing data into feature(x) and target(y)

In [41]:
```
x=train.drop('SalePrice',axis=1)
y=train['SalePrice']
```

In [42]:
```
cat_columns=[i for i in x.columns if x[i].dtypes=='O']
cat_columns
```

```
['MSZoning',
```

Out[42]:
```
'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']
```

In [43]:
```
from sklearn.preprocessing import OrdinalEncoder,LabelEncoder
label=LabelEncoder()

#using ordinal encoder for independent features
for i in cat_columns:
    x[i]=label.fit_transform(x[i].values.reshape(-1,1))
    train[i]=label.fit_transform(train[i].values.reshape(-1,1))
    test[i]=label.fit_transform(test[i].values.reshape(-1,1))
```

In [44]:
```
x.head()
```

Out[44]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighbo
0	120	3	70.98847	4928	1	0	3	4	0	
1	20	3	95.00000	15865	1	0	3	4	1	
2	60	3	92.00000	9920	1	0	3	1	0	
3	20	3	105.00000	11751	1	0	3	4	0	
4	20	3	70.98847	16635	1	0	3	2	0	

In [45]:
```
test.head()
```

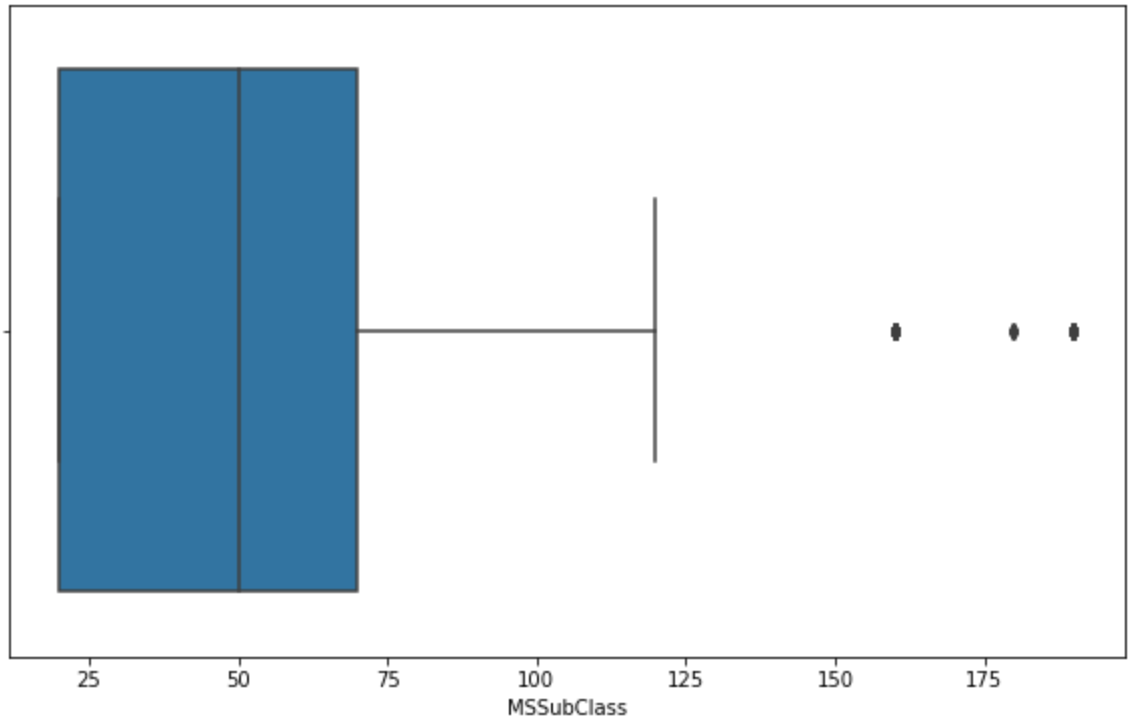
Out[45]:

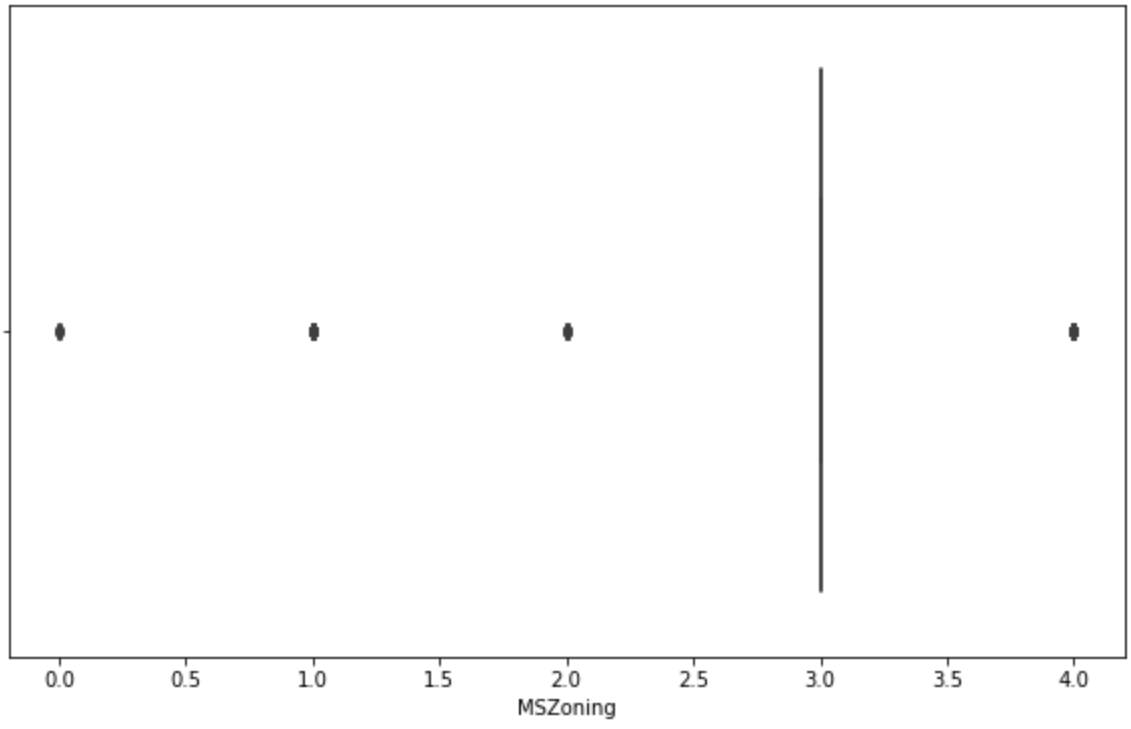
	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighbo
0	20	2	86.000000	14157	1	0	1	0	0	

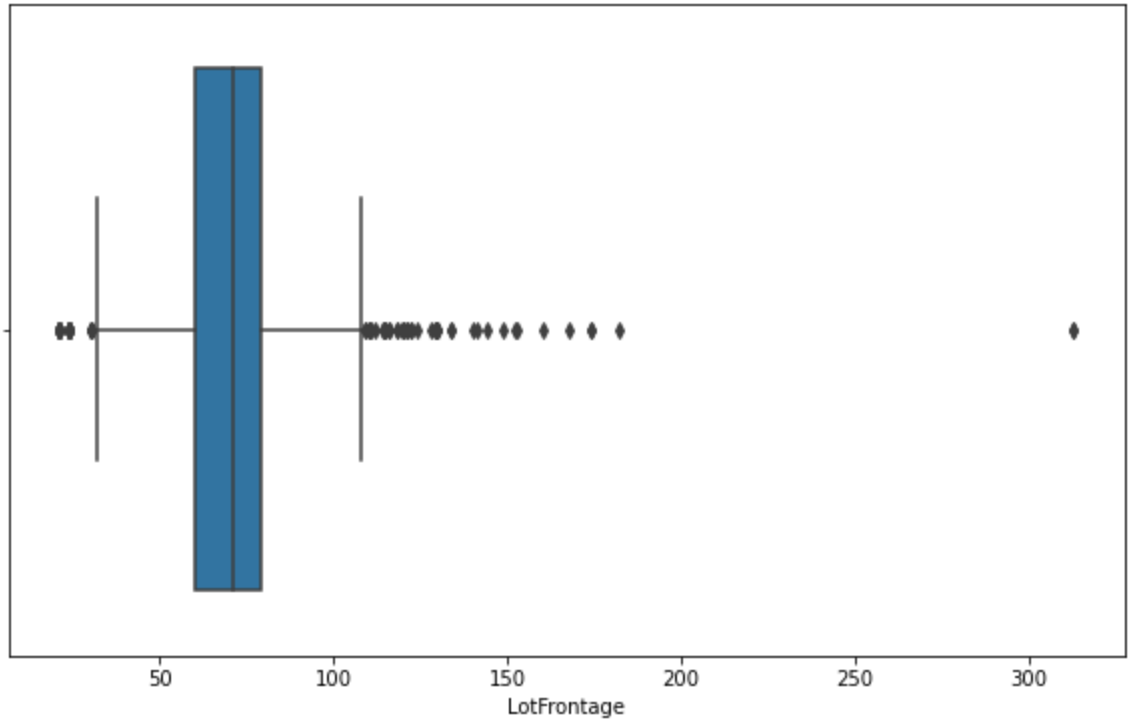
1	120	2	66.425101	5814	1	0	3	1	0
2	20	2	66.425101	11838	1	3	3	4	0
3	70	2	75.000000	12000	1	3	0	4	0
4	60	2	86.000000	14598	1	0	3	1	0

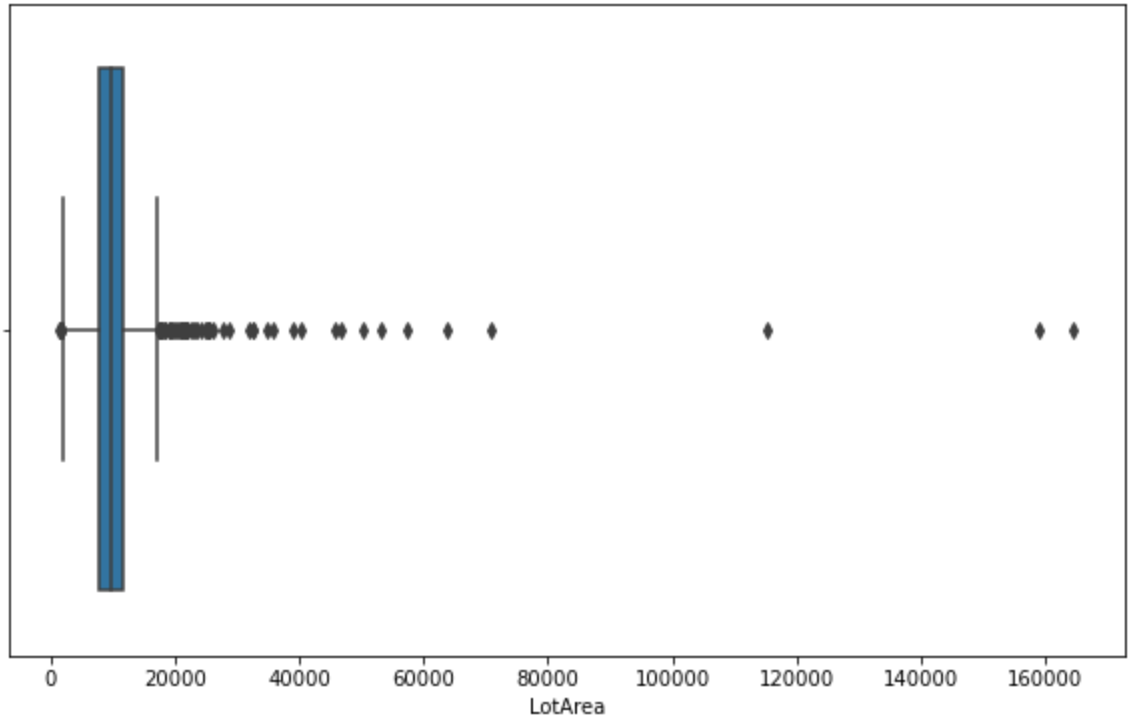
Outliers

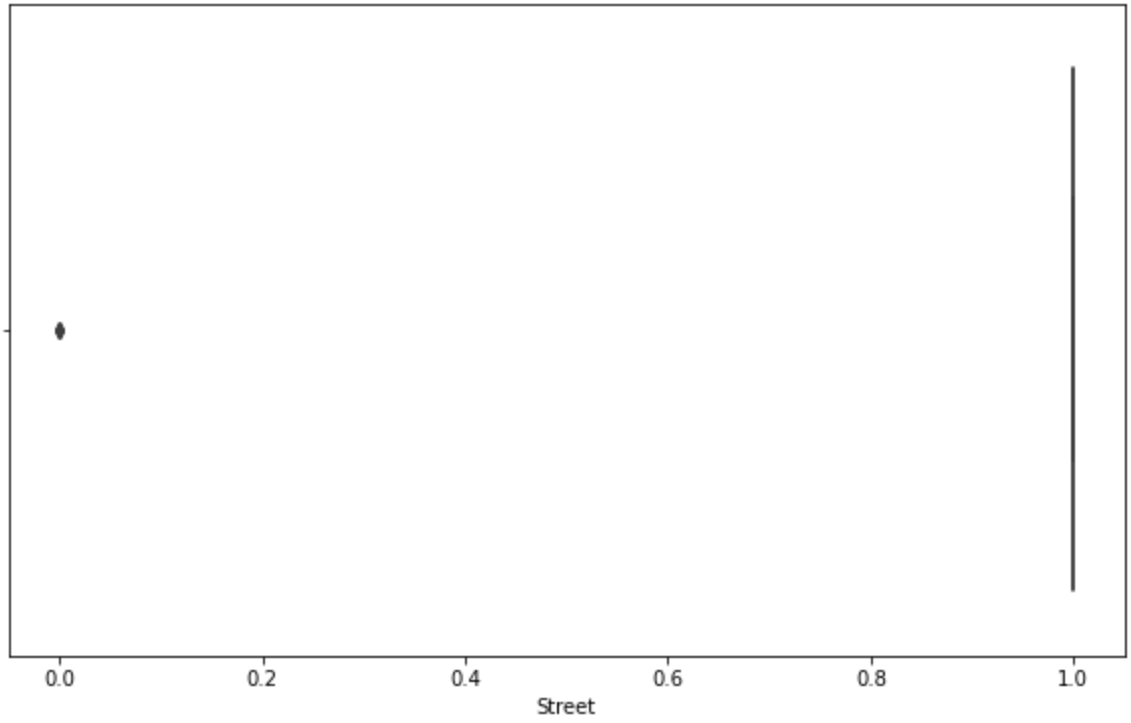
In [46]:
```
for i in x:
    plt.figure(figsize=(10,6))
    sns.boxplot(train[i])
```

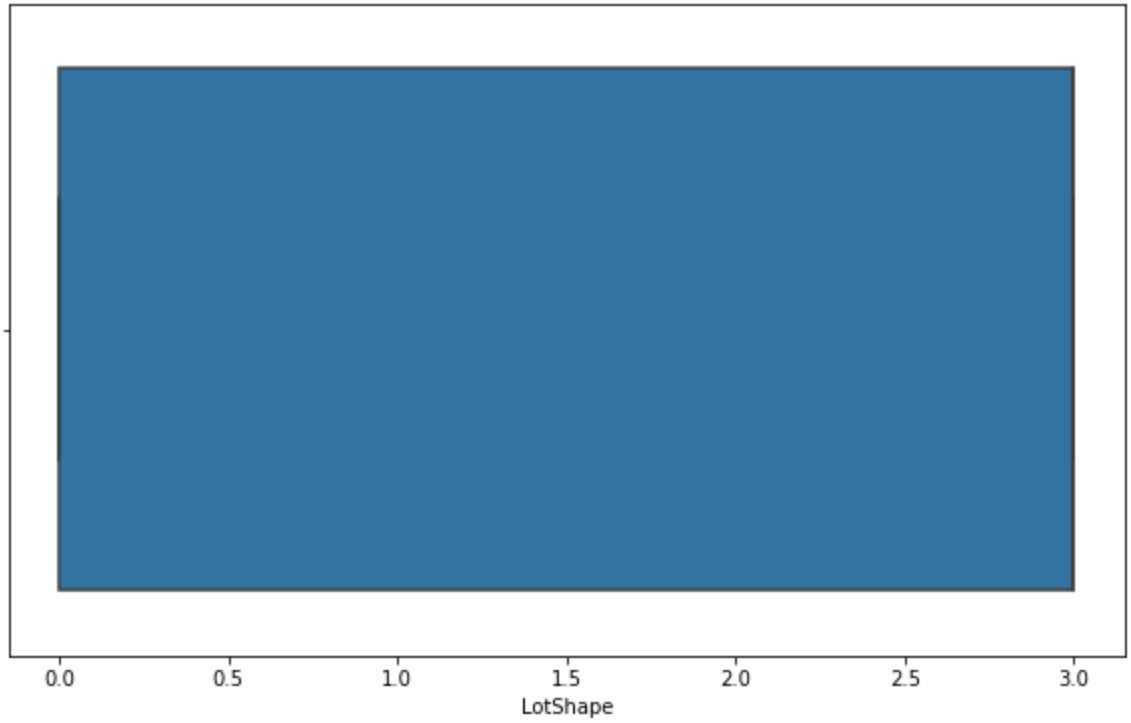


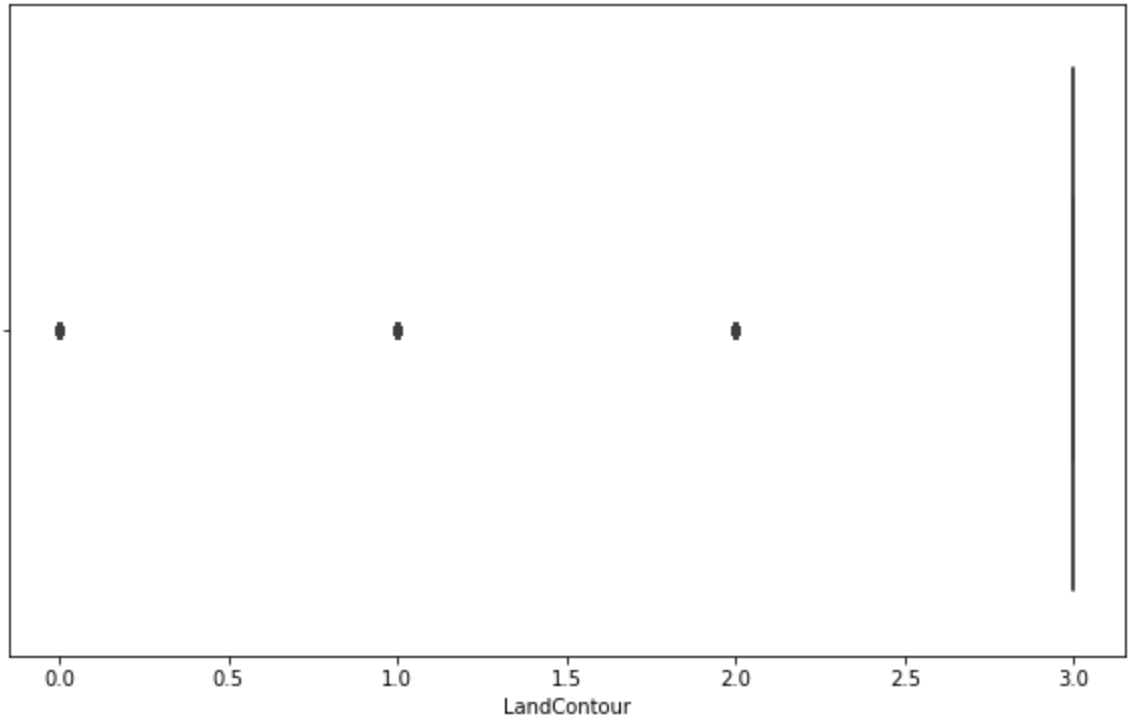


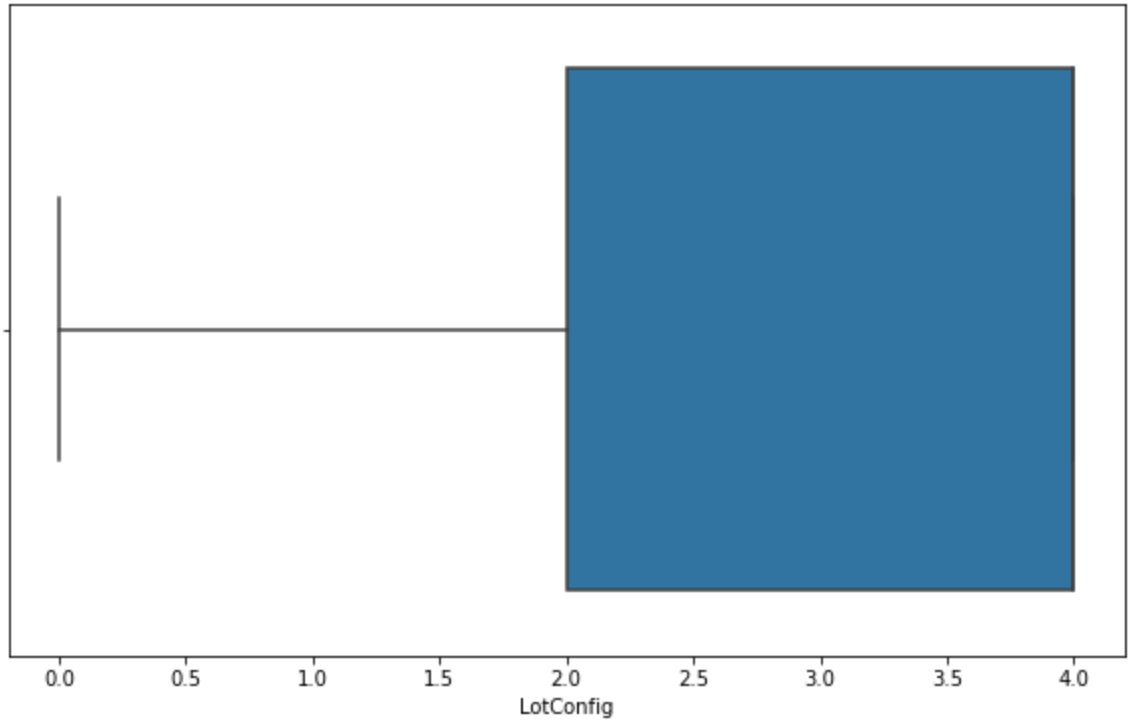

50
100
150
200
250
300
LotFrontage

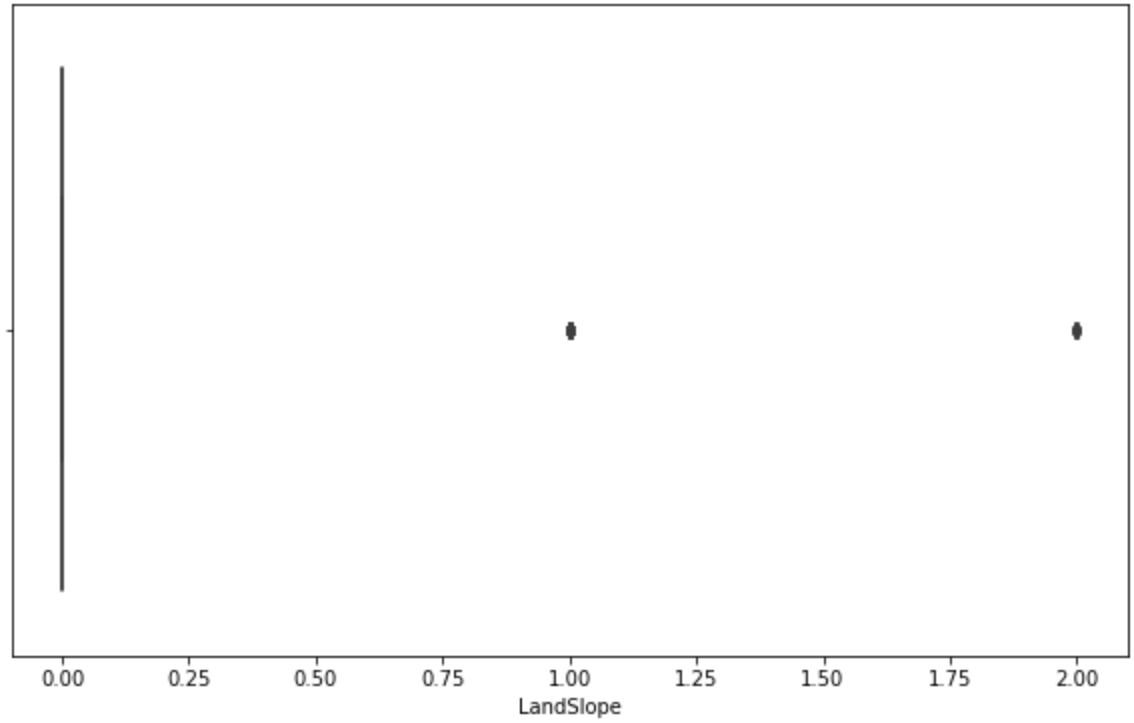

0
20000
40000
60000
80000
100000
120000
140000
160000
LotArea

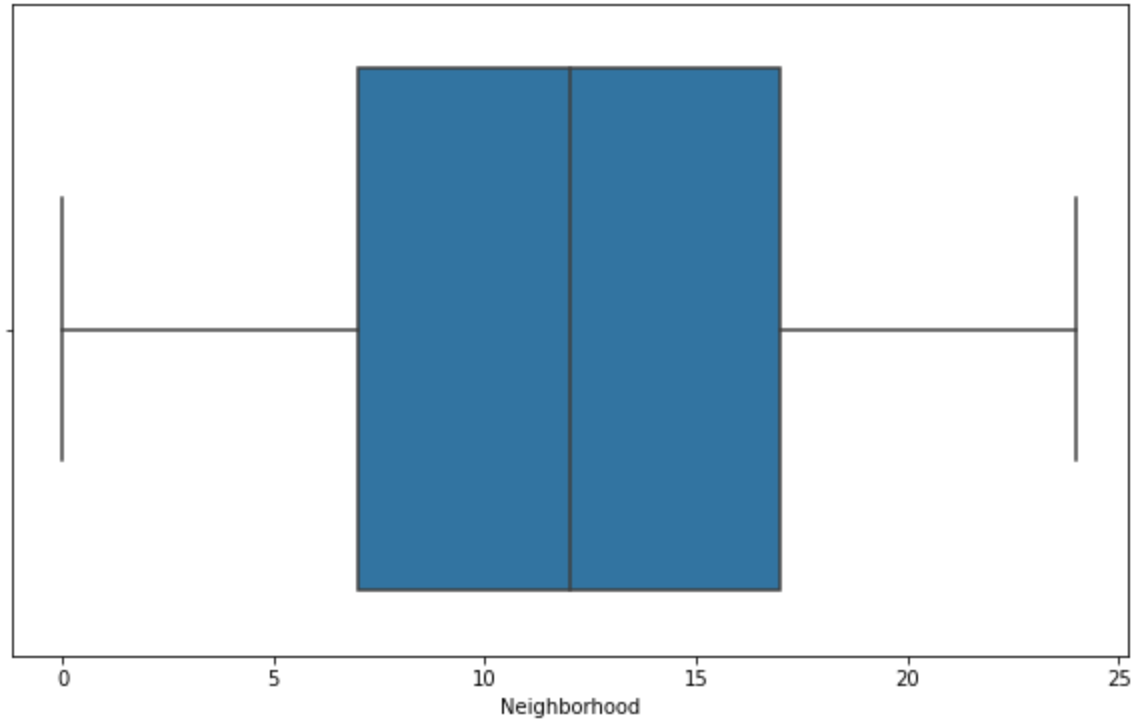

0.0
0.2
0.4
0.6
0.8
1.0
Street

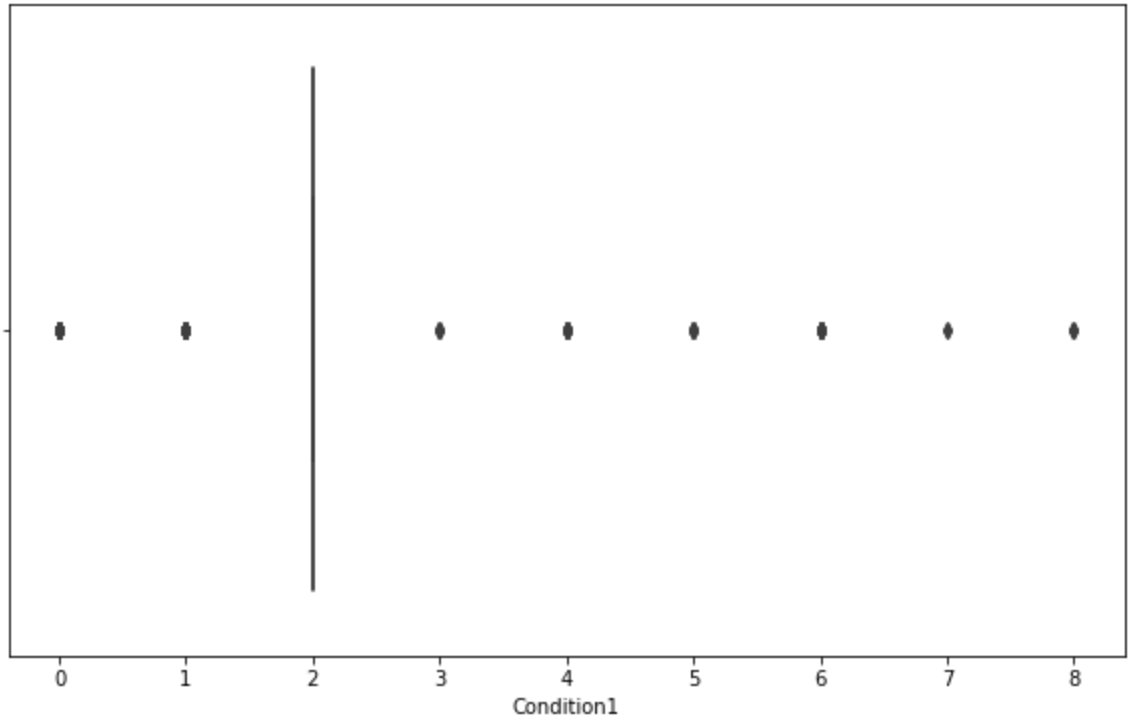

0.0
0.5
1.0
1.5
2.0
2.5
3.0
LotShape

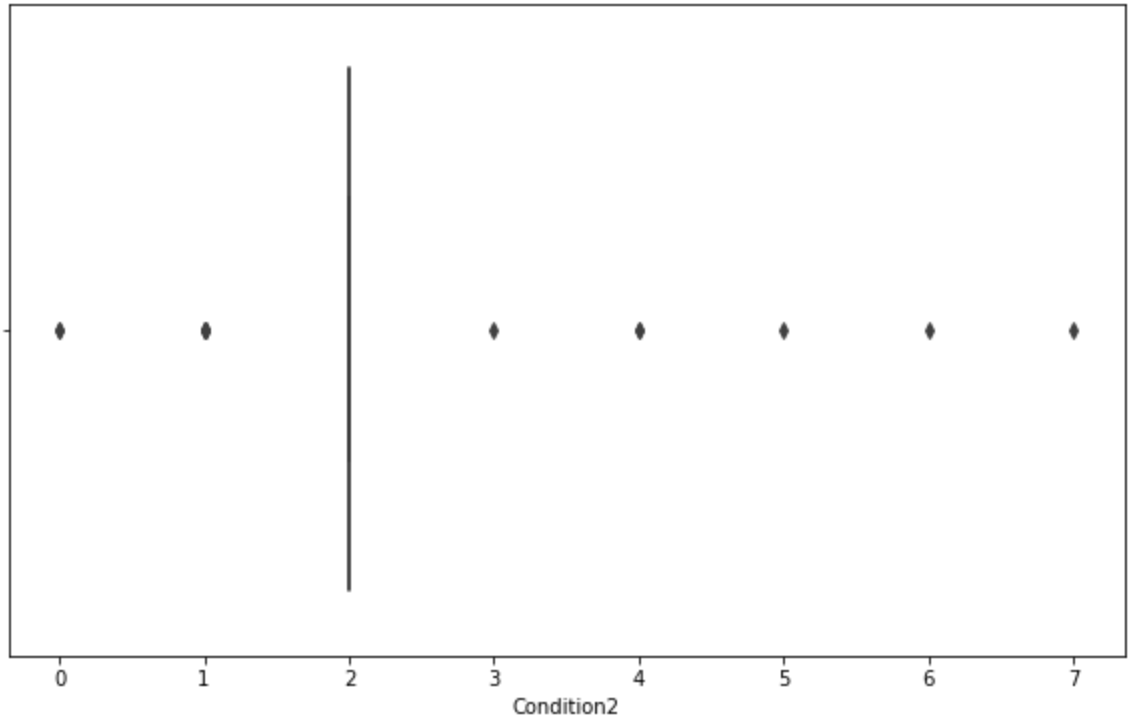

0.0
0.5
1.0
1.5
2.0
2.5
3.0
LandContour

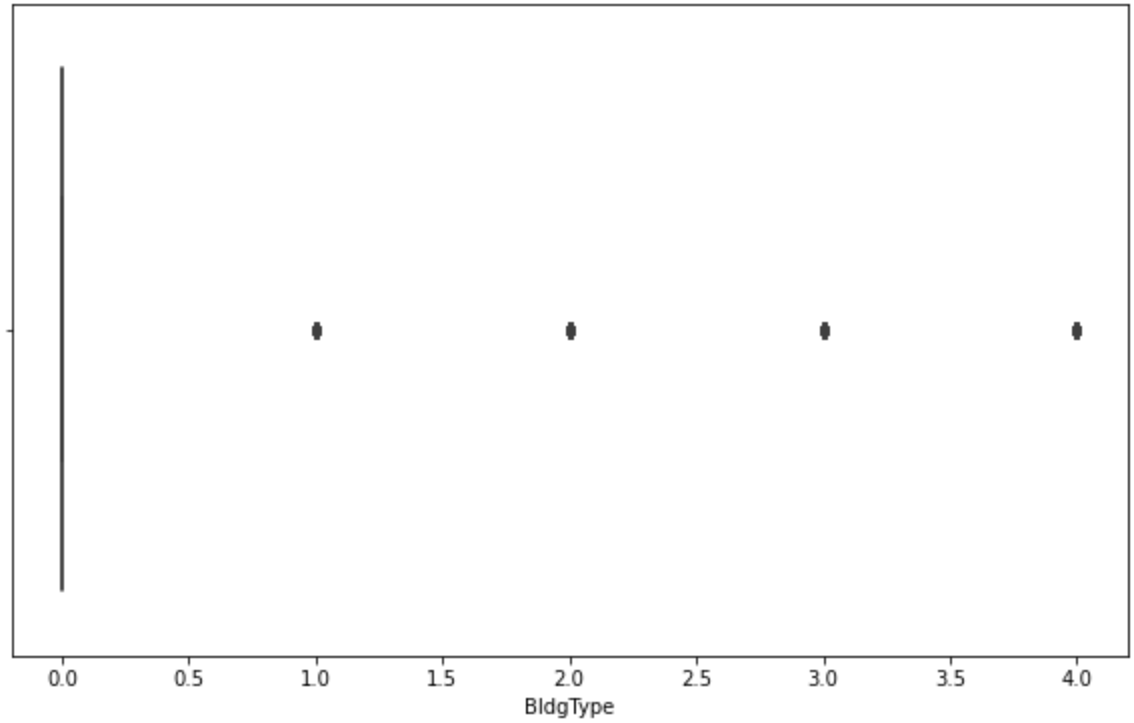

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
LotConfig

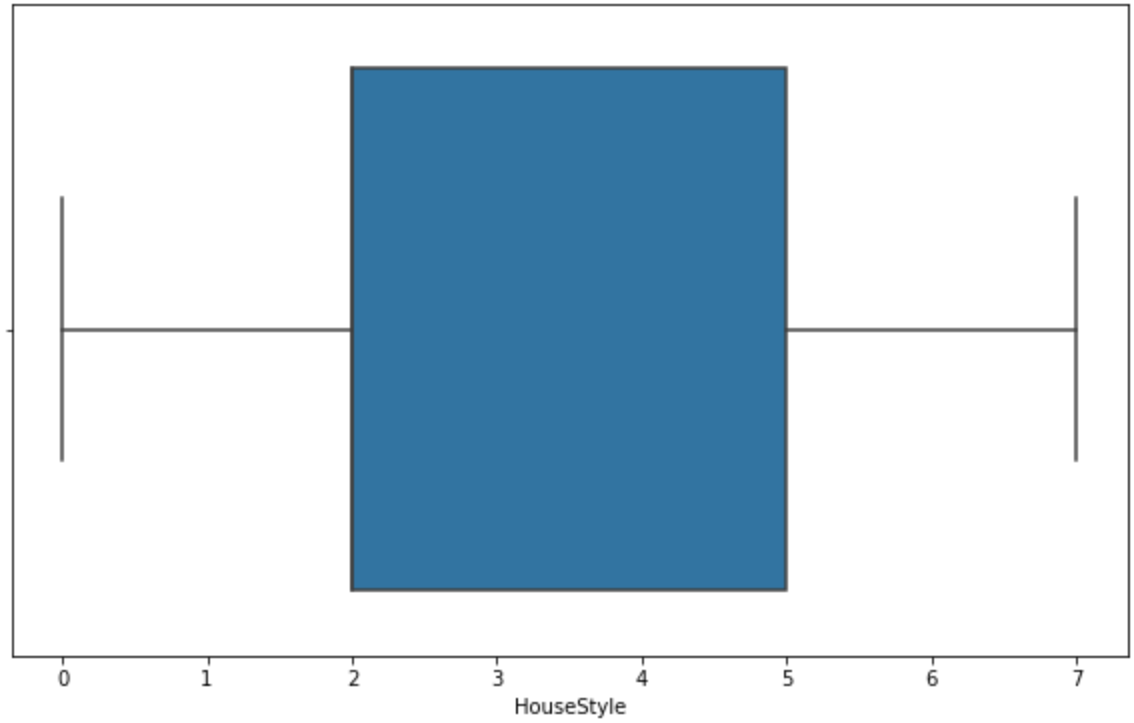

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
LandSlope

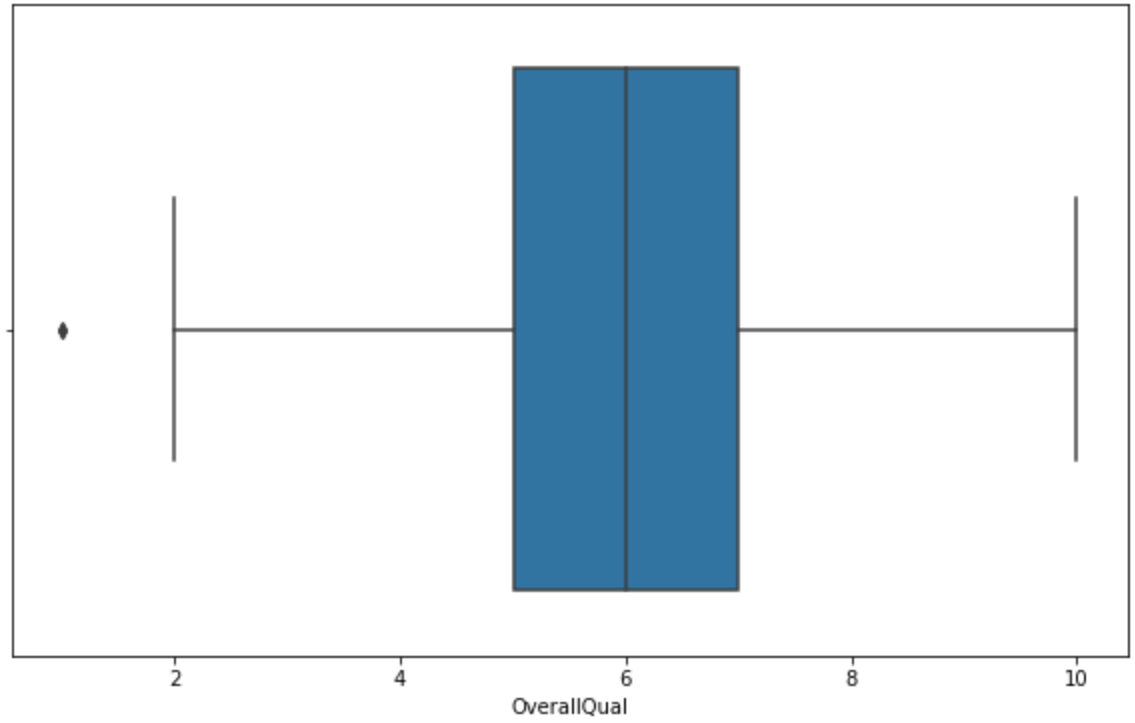

0
5
10
15
20
25
Neighborhood

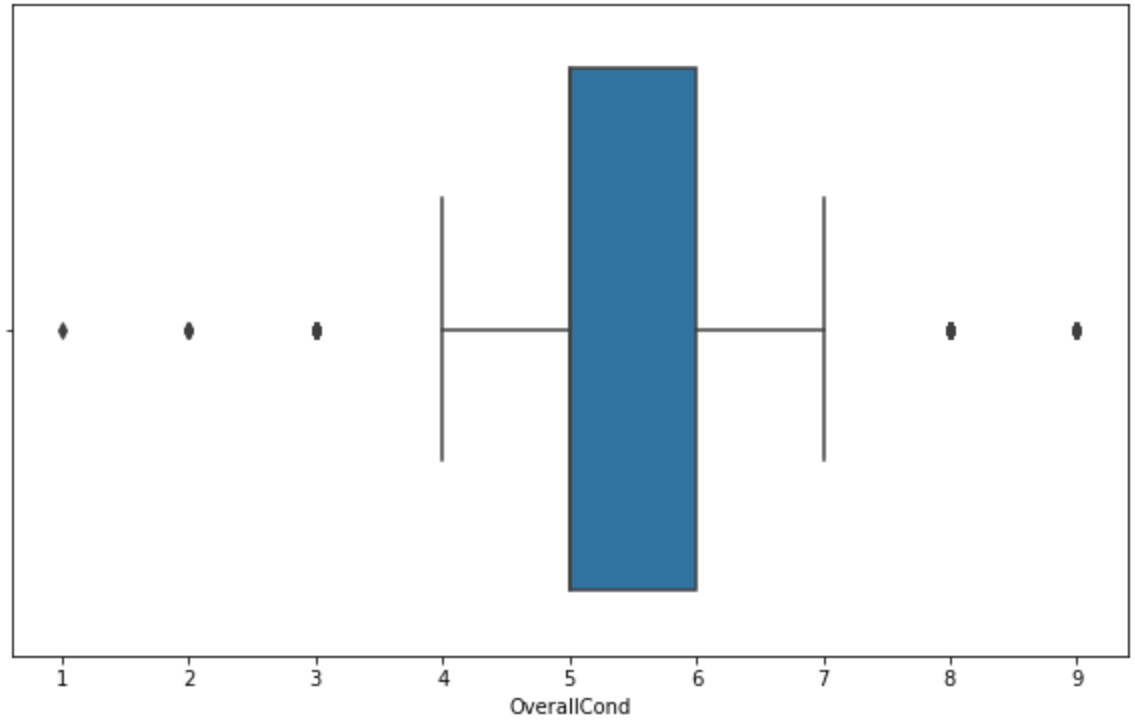

0
1
2
3
4
5
6
7
8
Condition1

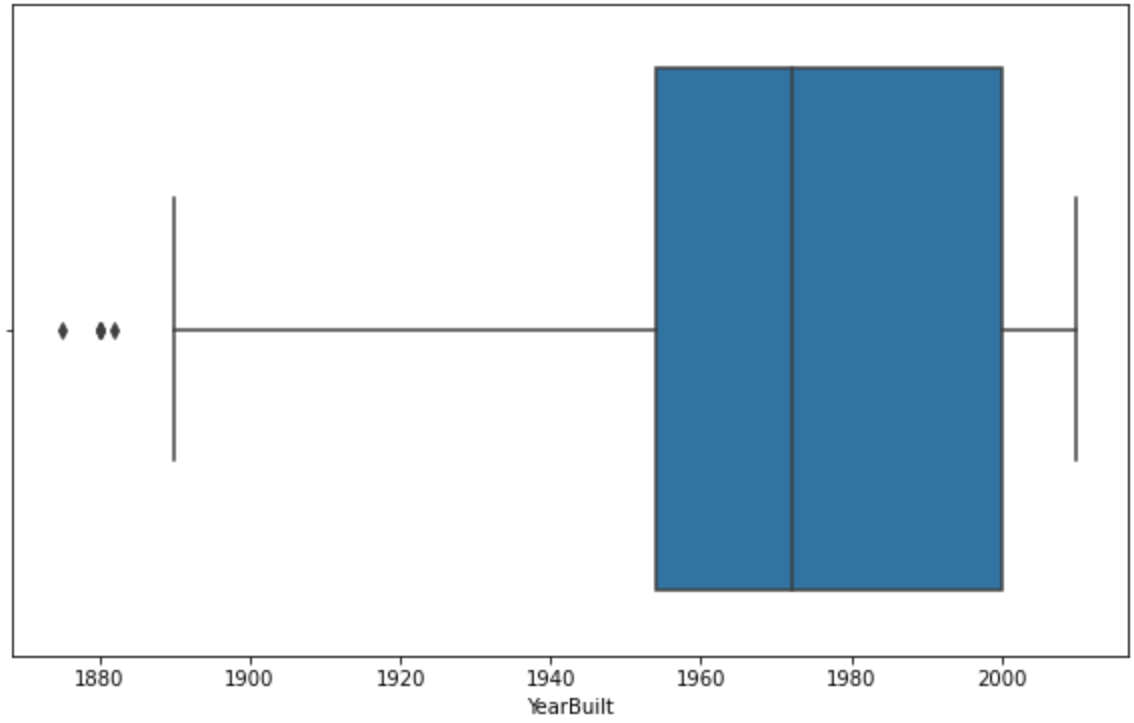

0
1
2
3
4
5
6
7
Condition2

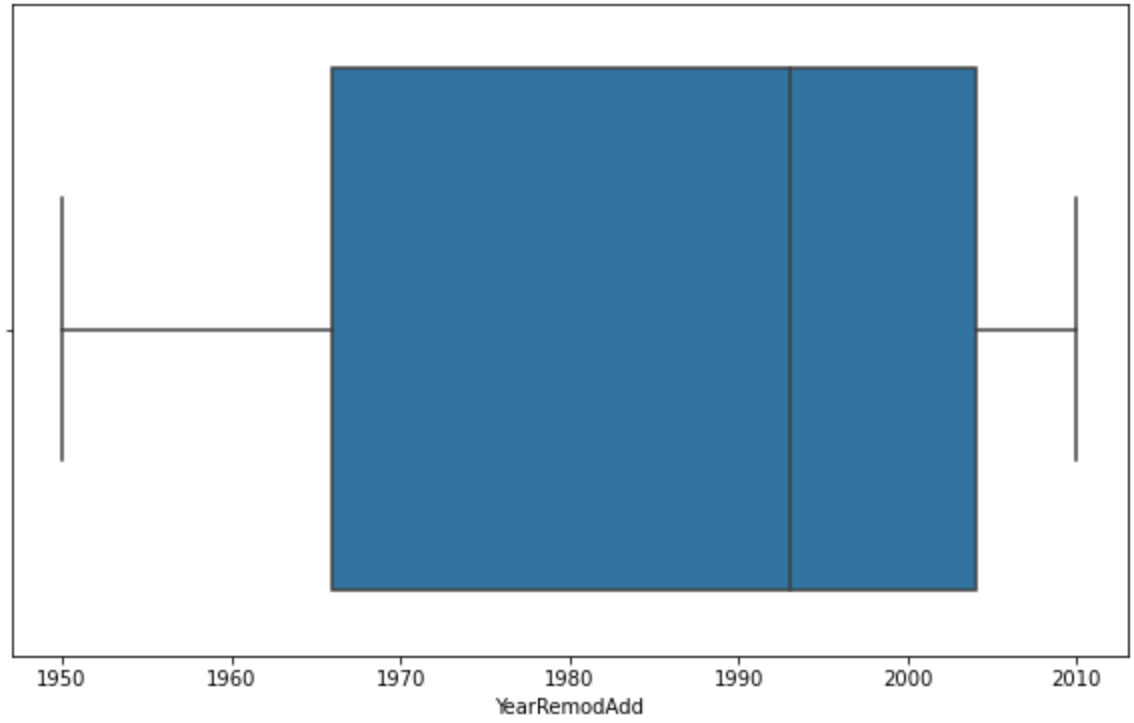

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
BldgType

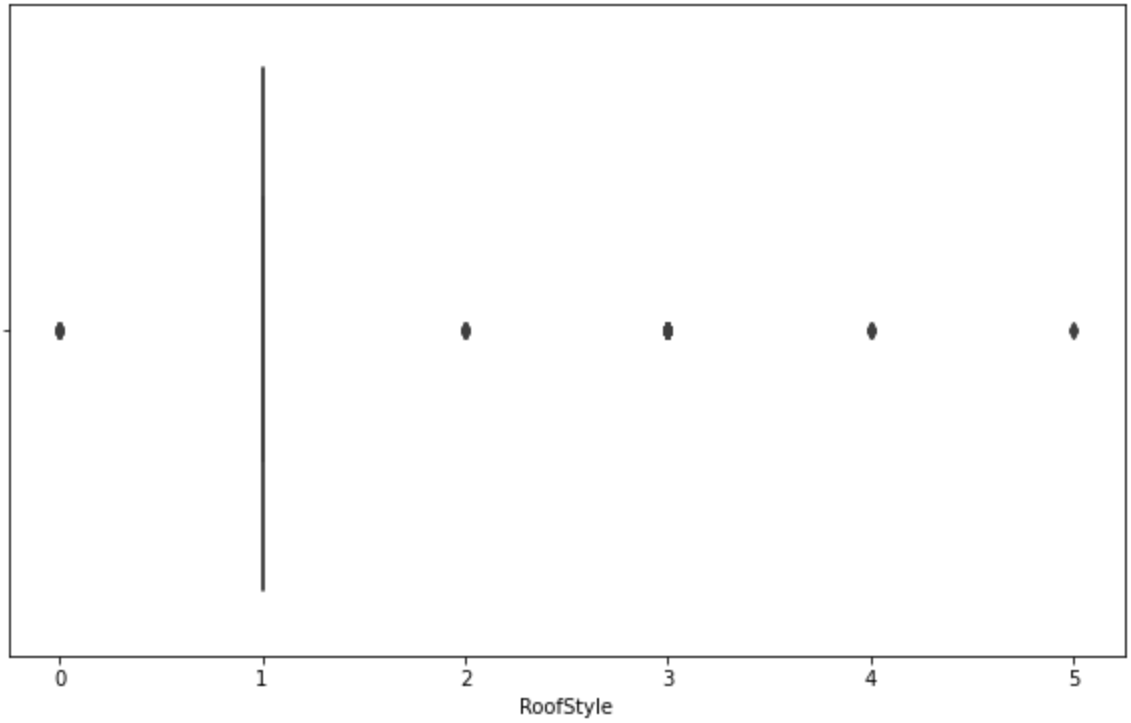

0
1
2
3
4
5
6
7
HouseStyle

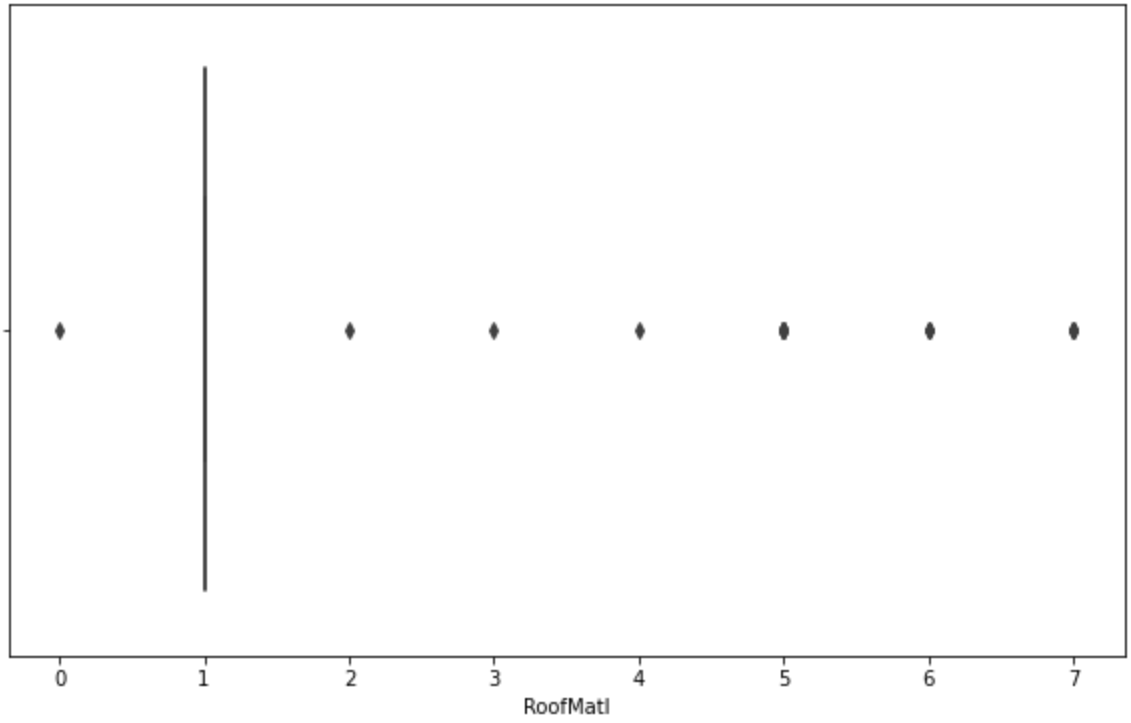

2
4
6
8
10
OverallQual

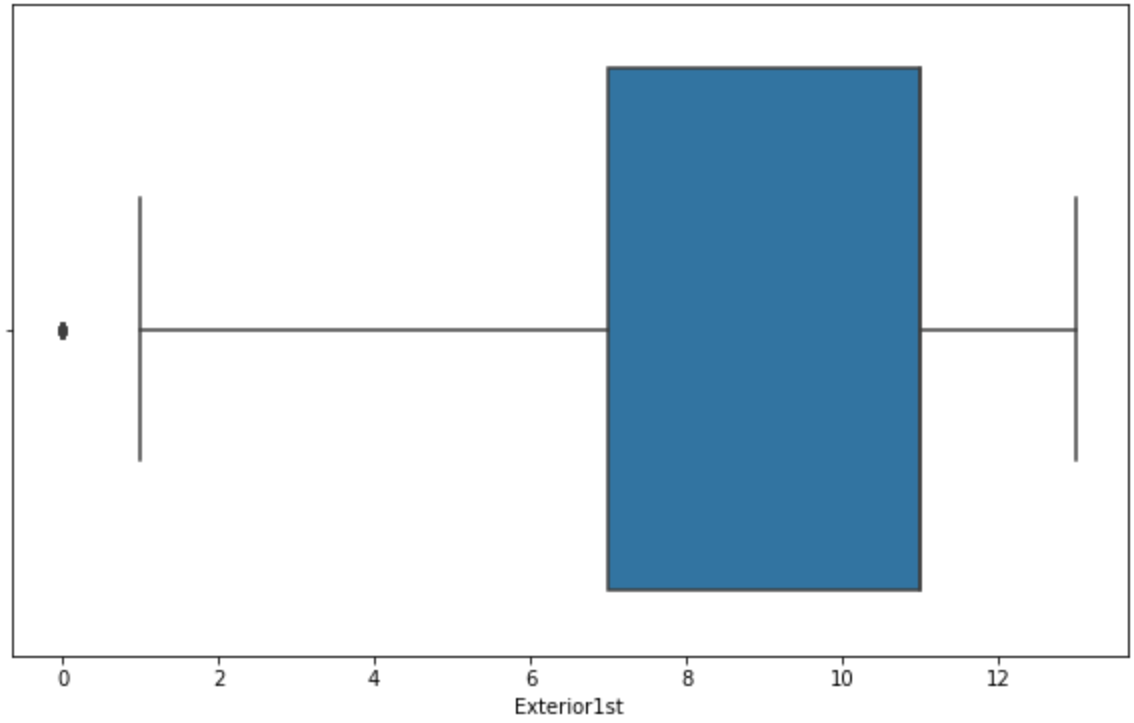

1
2
3
4
5
6
7
8
9
OverallCond

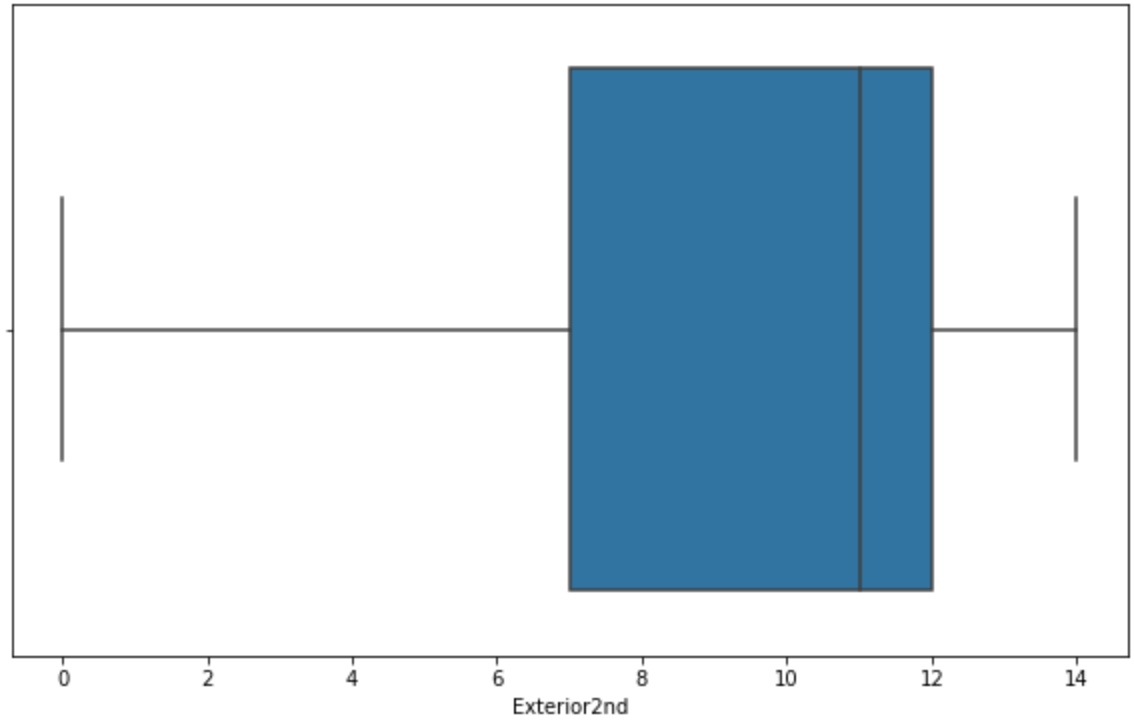

1880
1900
1920
1940
1960
1980
2000
YearBuilt

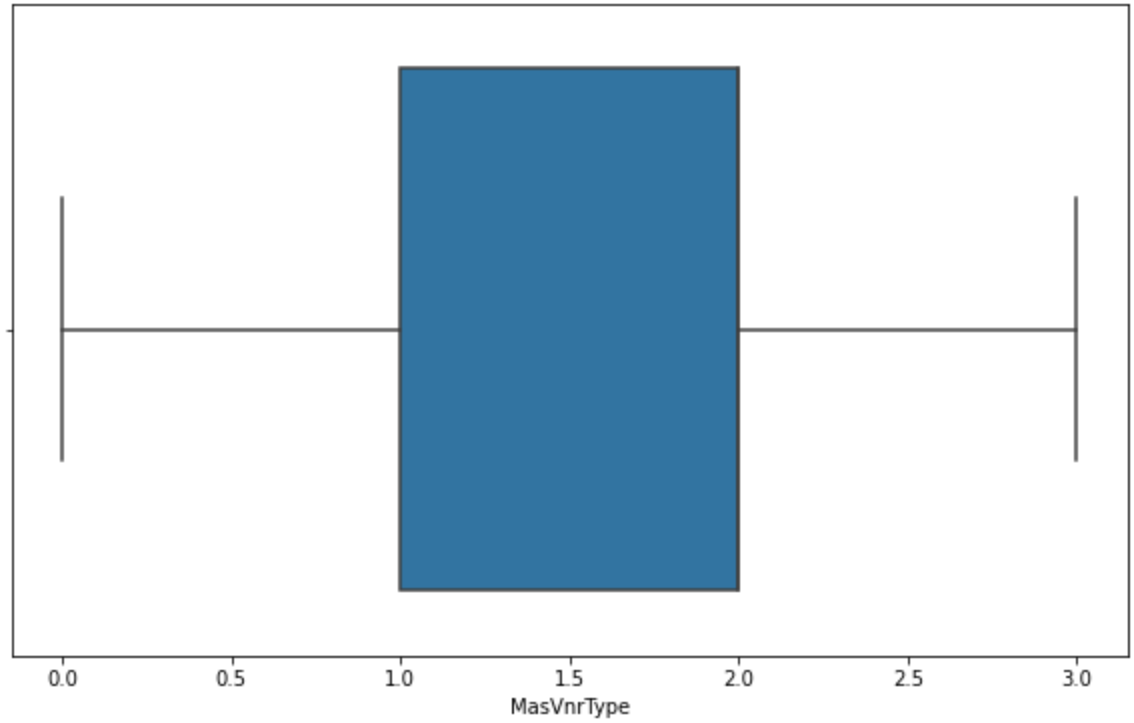

1950
1960
1970
1980
1990
2000
2010
YearRemodAdd

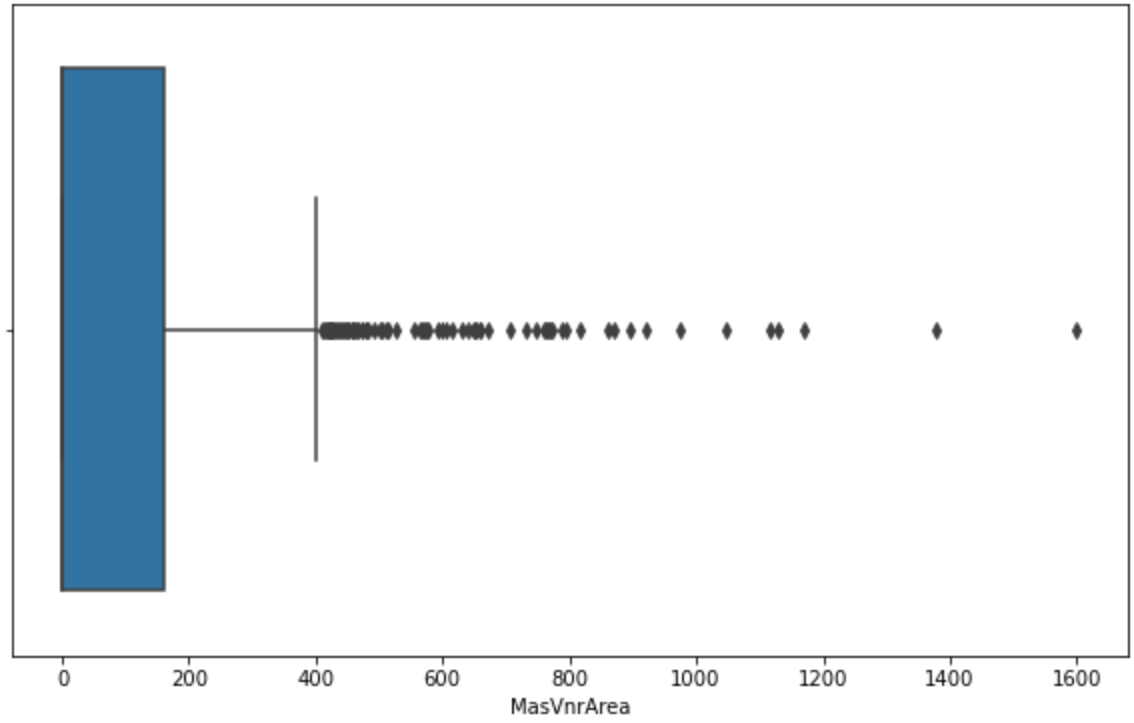

0
1
2
3
4
5
RoofStyle

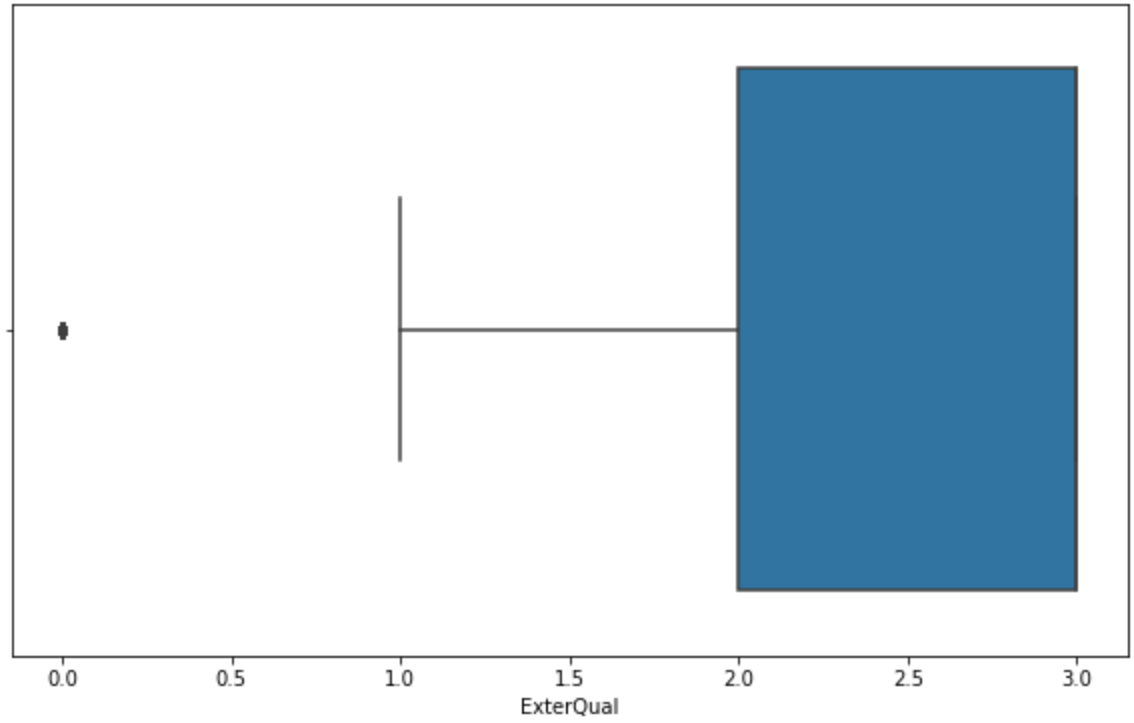

0
1
2
3
4
5
6
7
RoofMatl

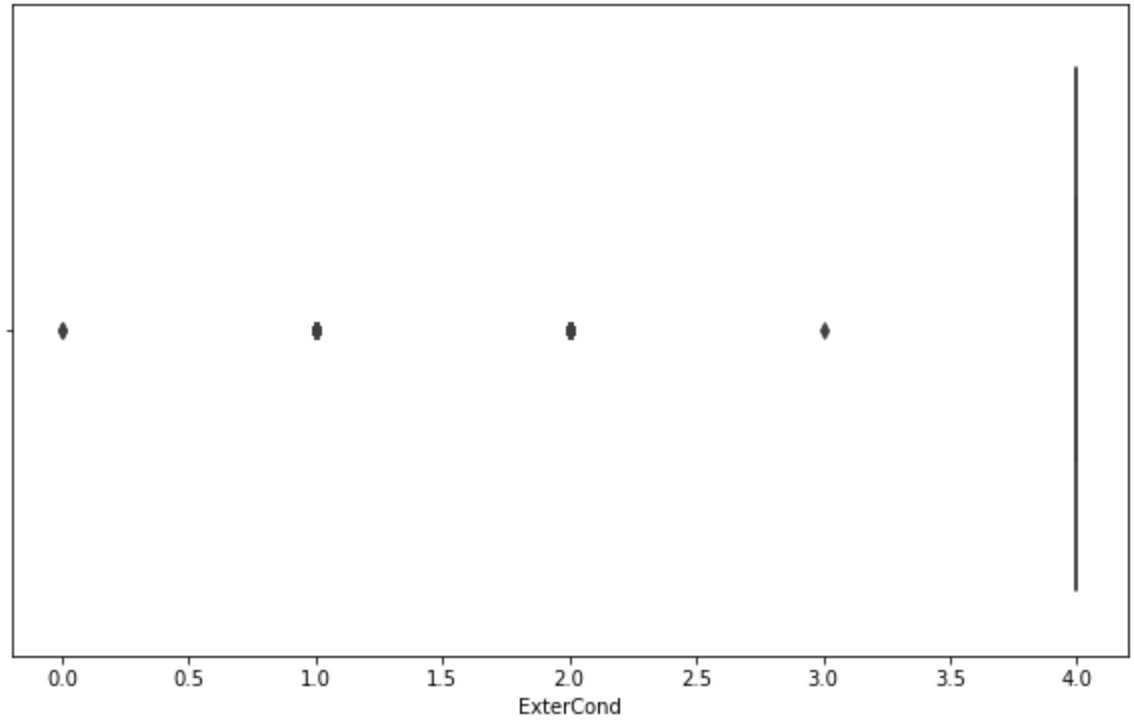

0
2
4
6
8
10
12
Exterior1st

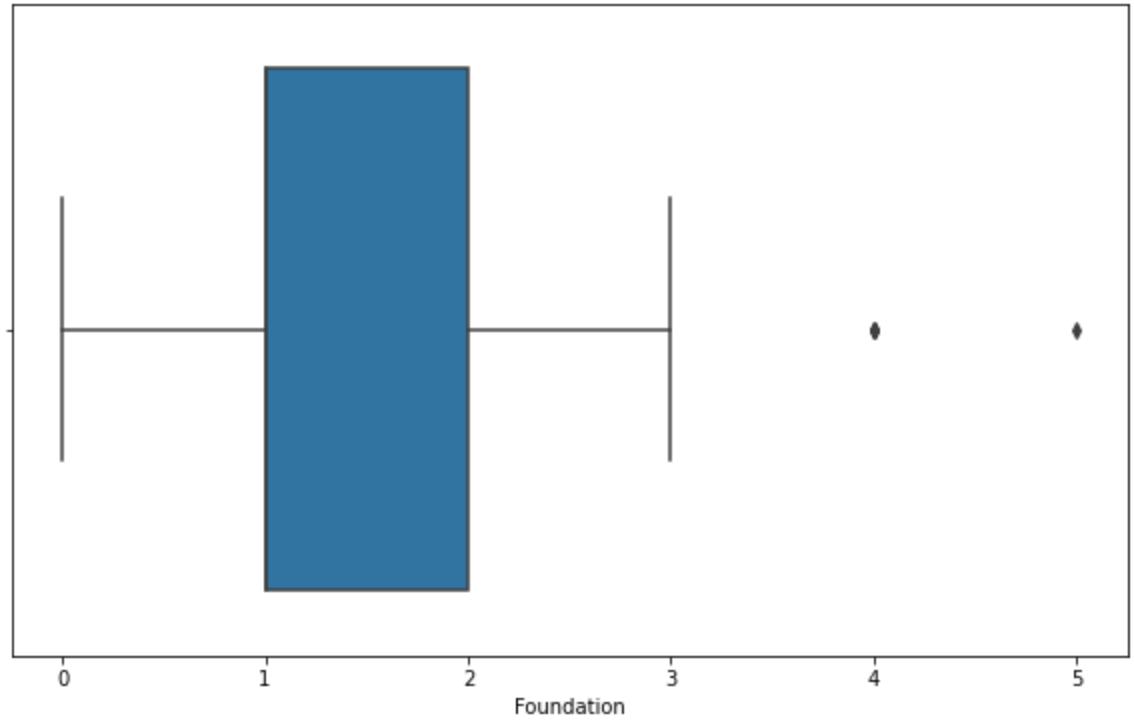

0
2
4
6
8
10
12
14
Exterior2nd

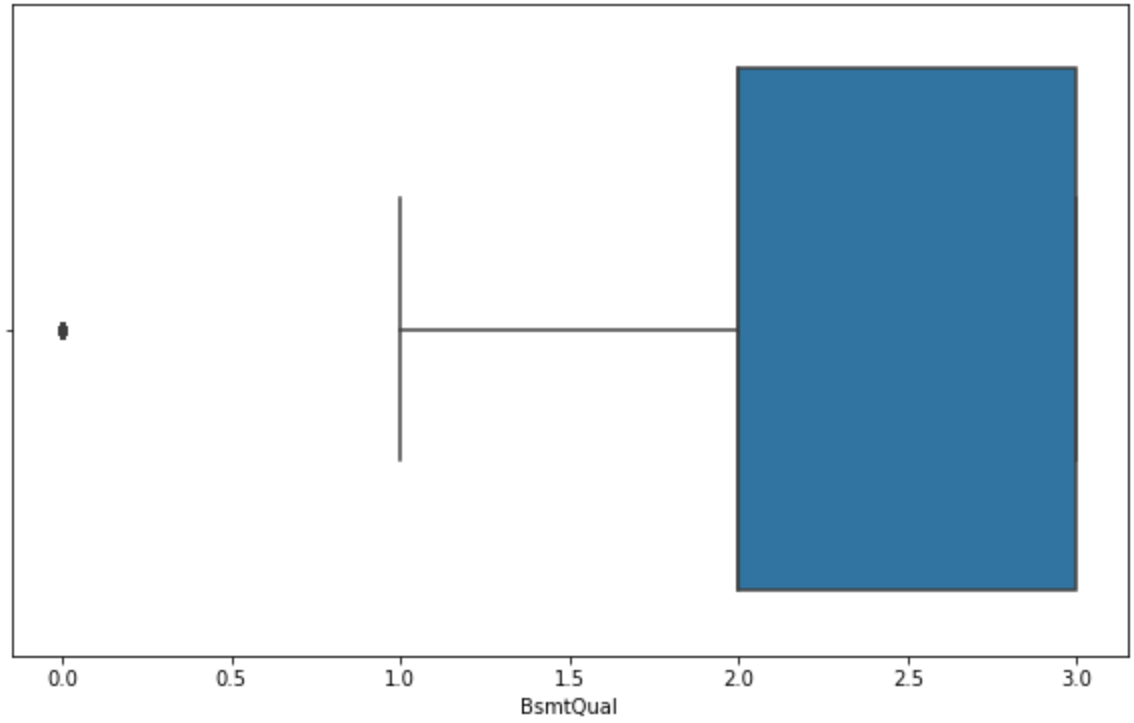

0.0
0.5
1.0
1.5
2.0
2.5
3.0
MasVnrType

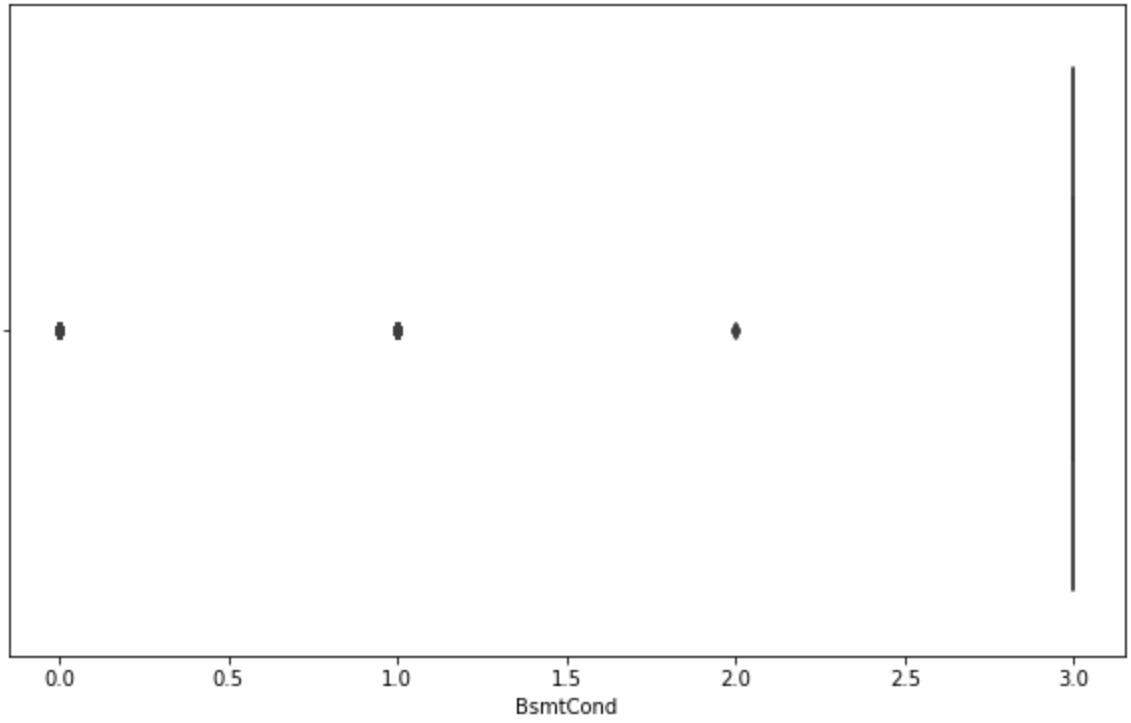

0
200
400
600
800
1000
1200
1400
1600
MasVnrArea

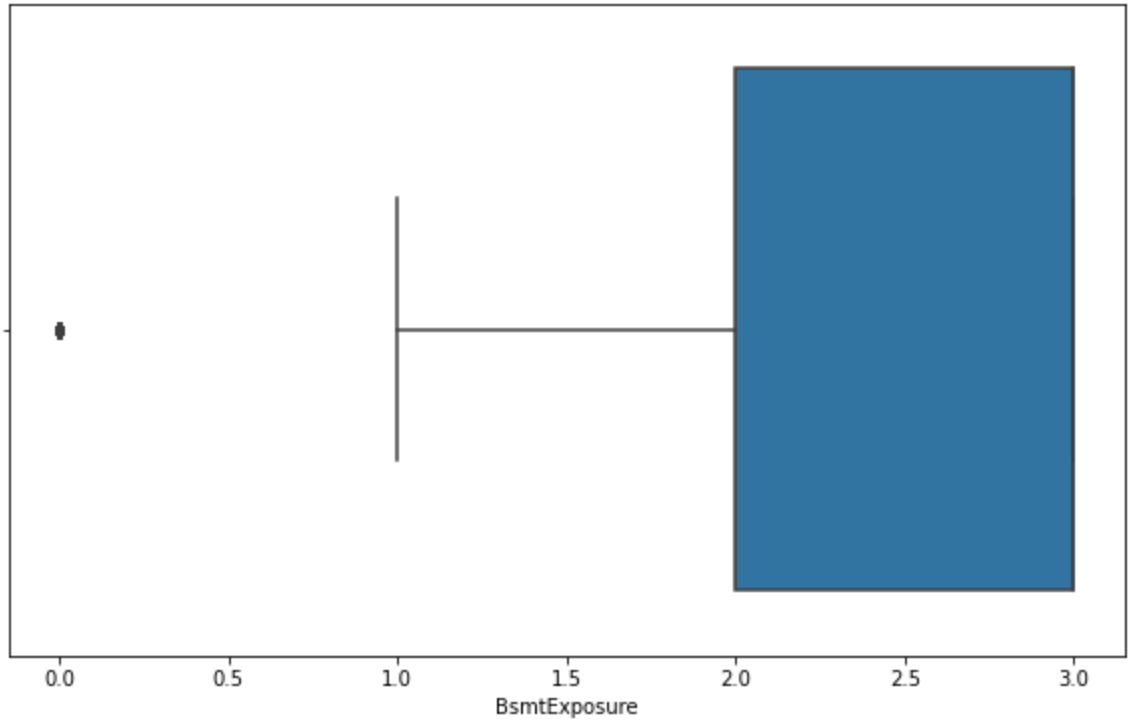

0.0
0.5
1.0
1.5
2.0
2.5
3.0
ExterQual

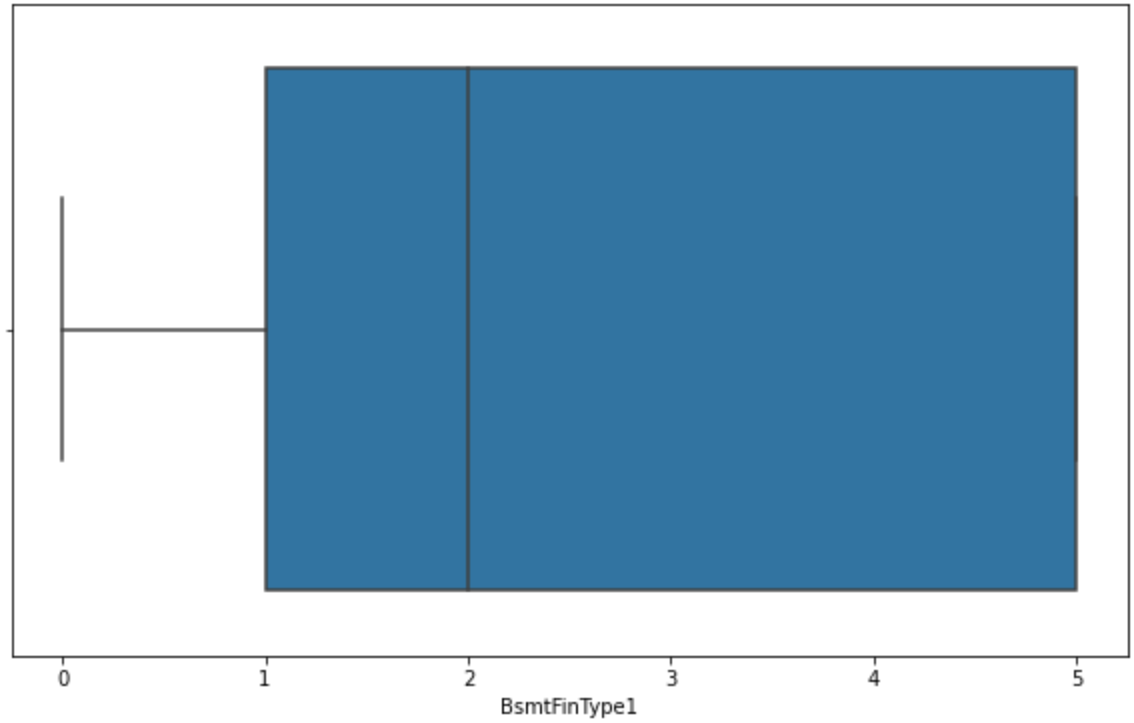

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
ExterCond

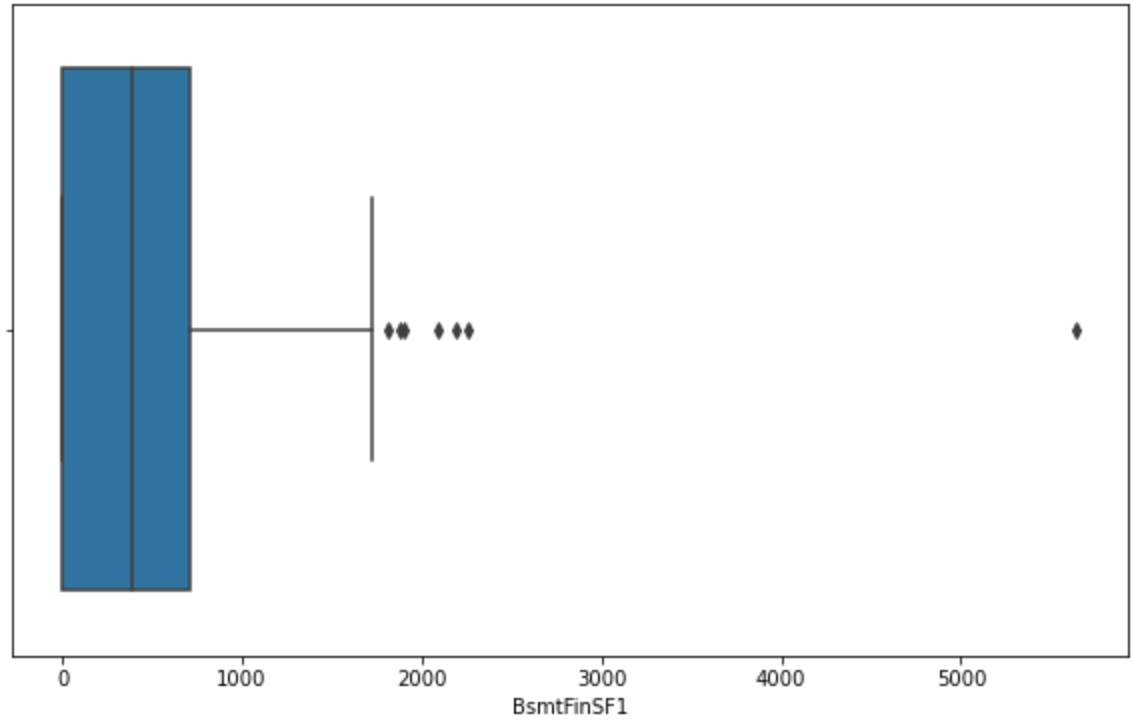

0
1
2
3
4
5
Foundation

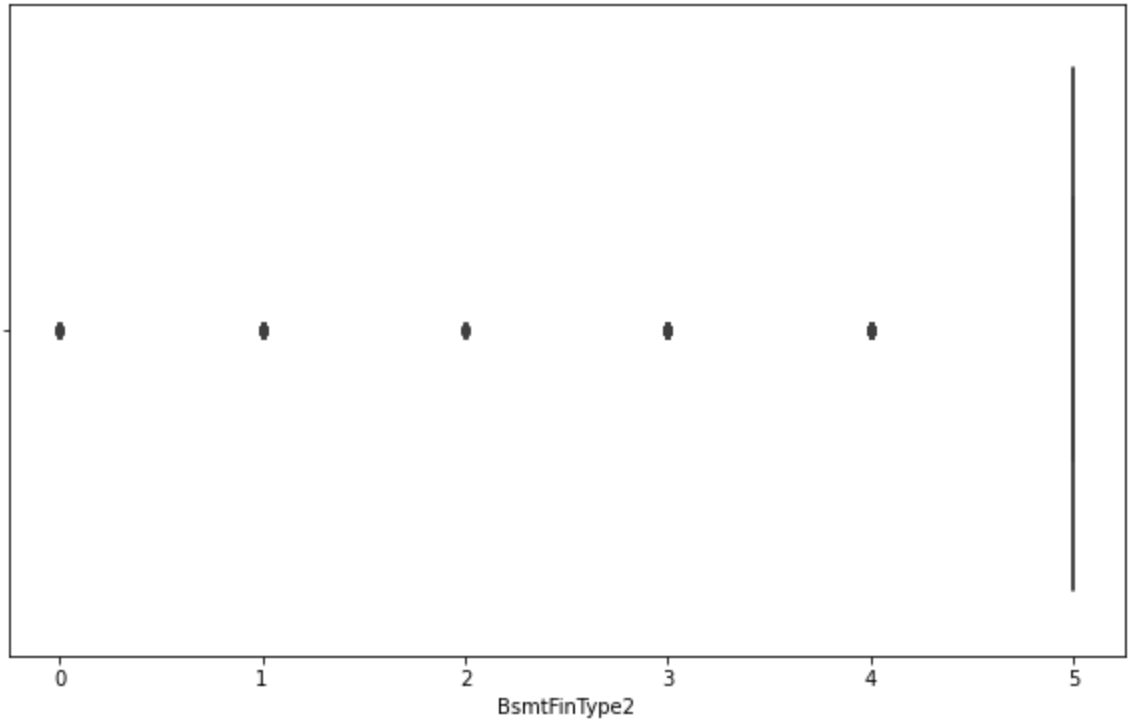

0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtQual

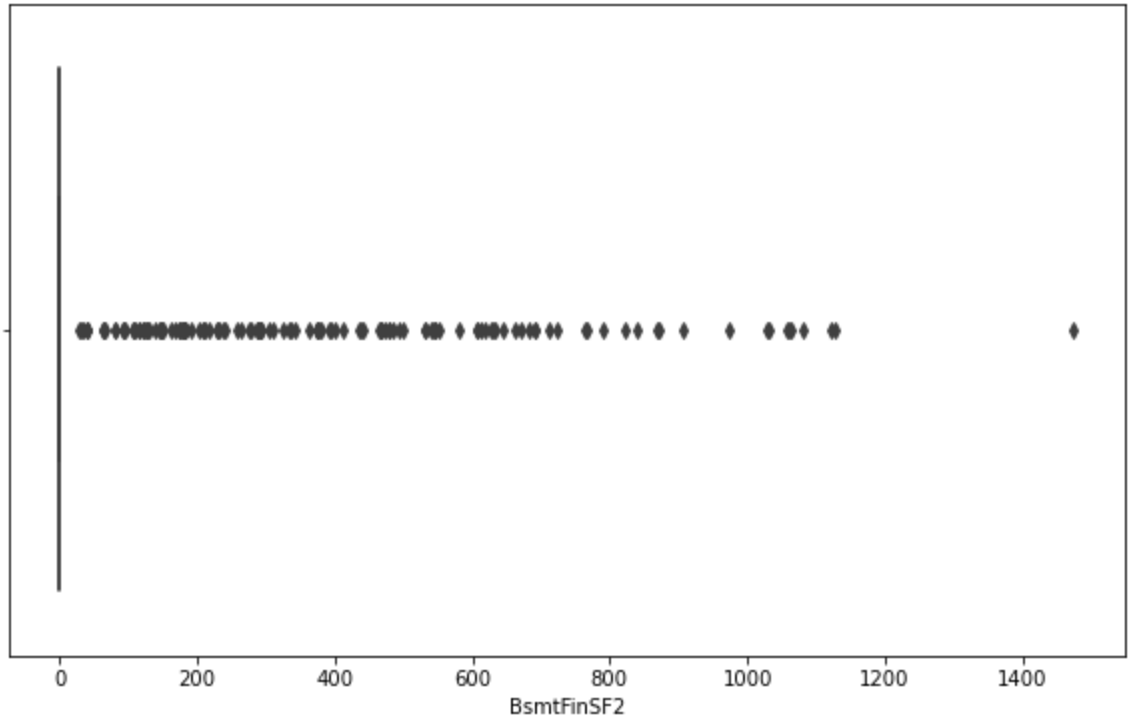

0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtCond

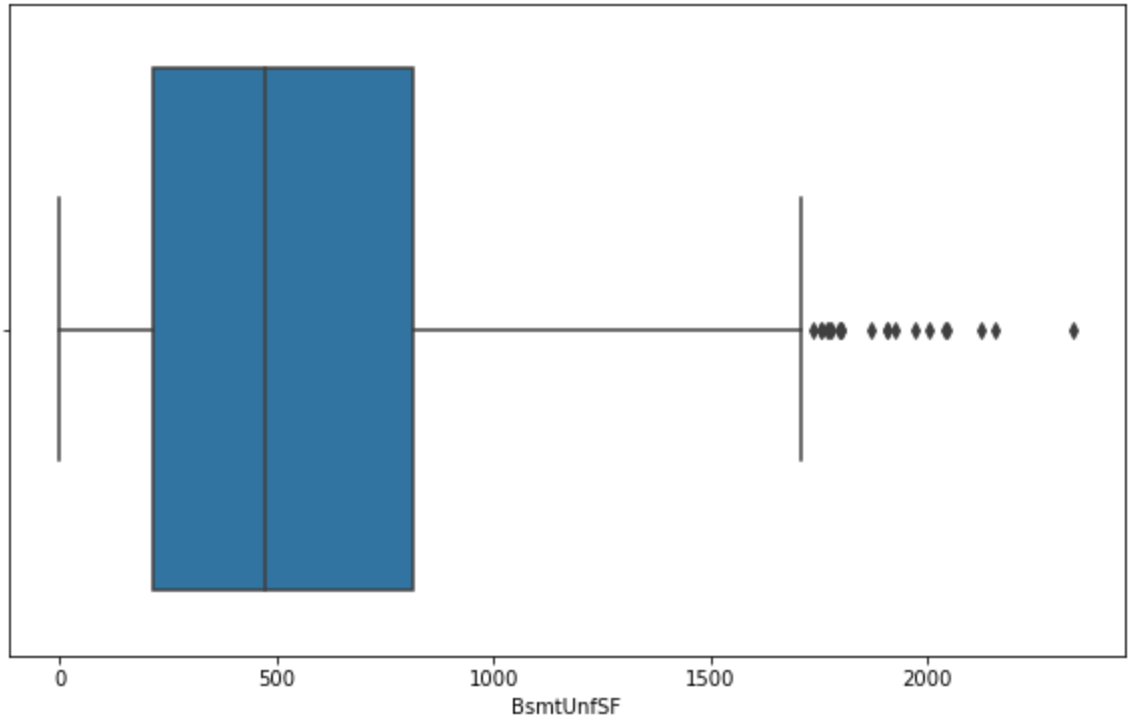

0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtExposure

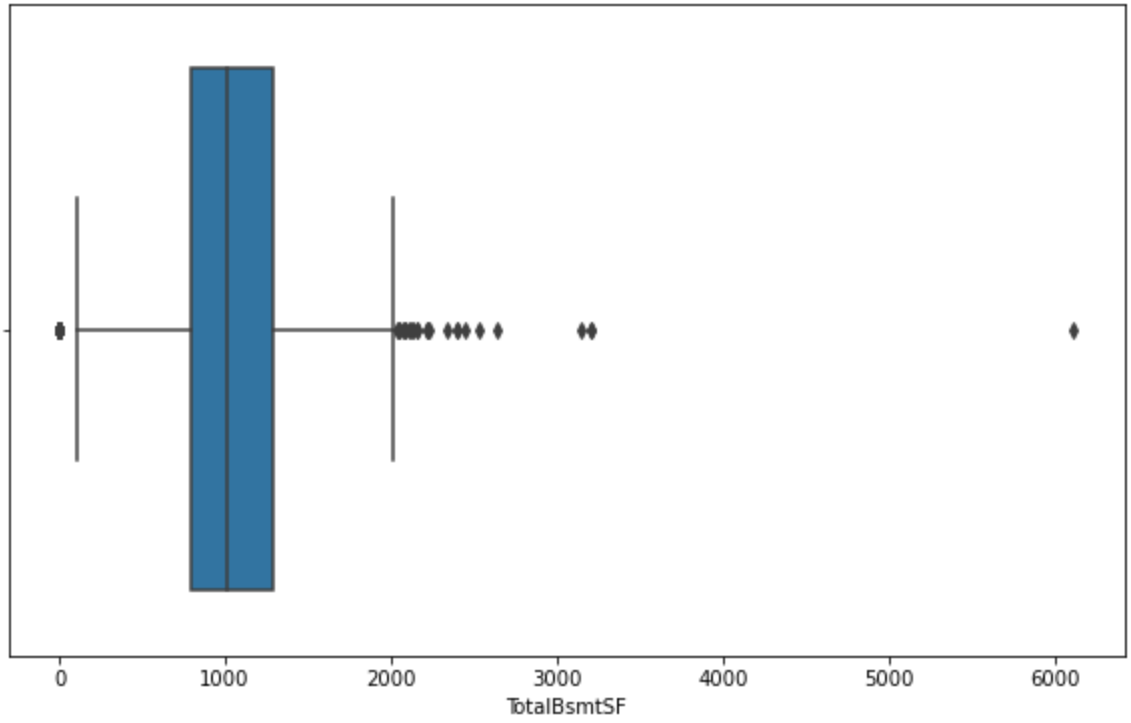

0
1
2
3
4
5
BsmtFinType1

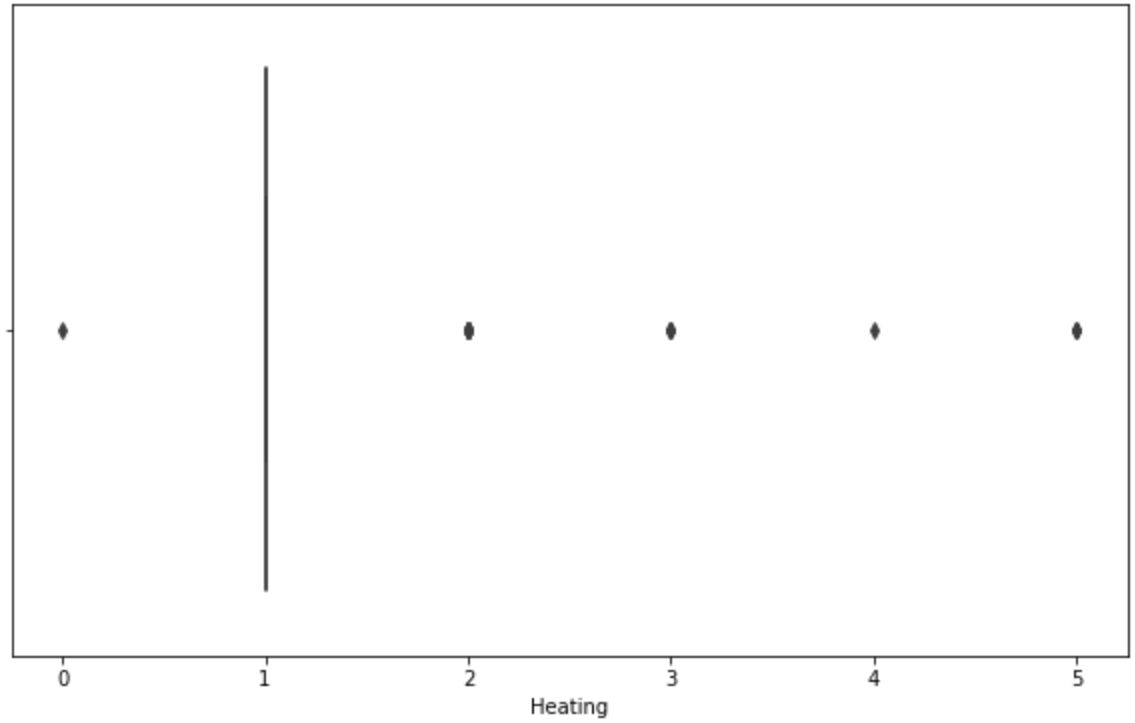

0
1000
2000
3000
4000
5000
BsmtFinSF1

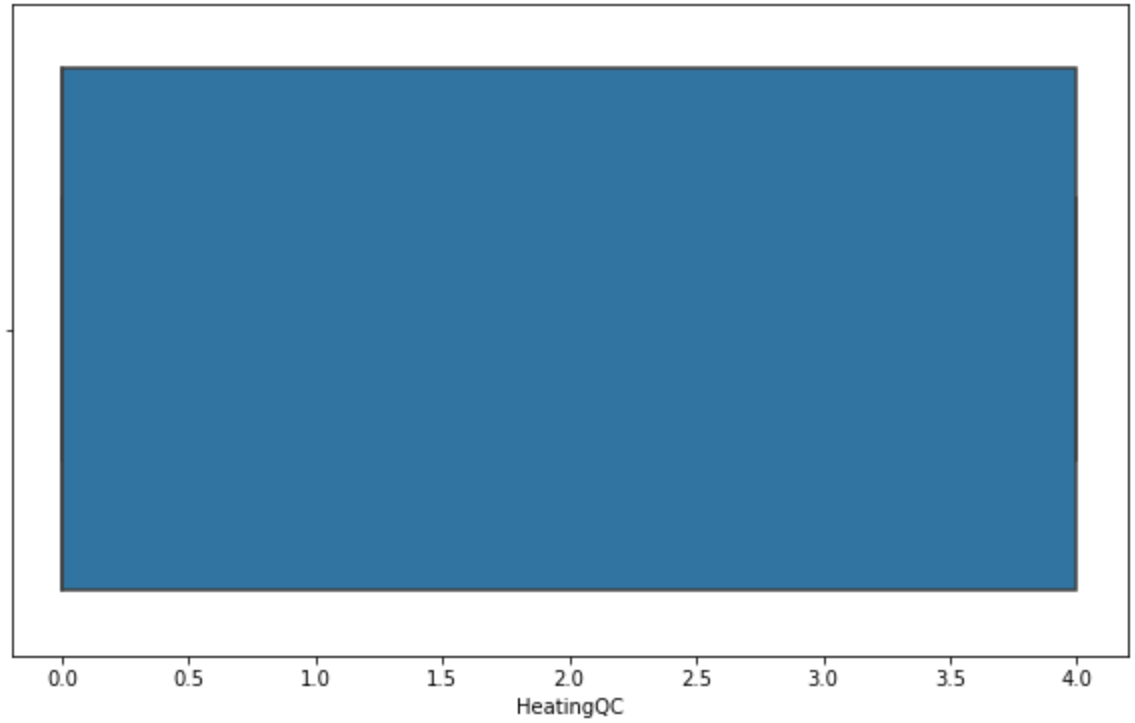

0
1
2
3
4
5
BsmtFinType2

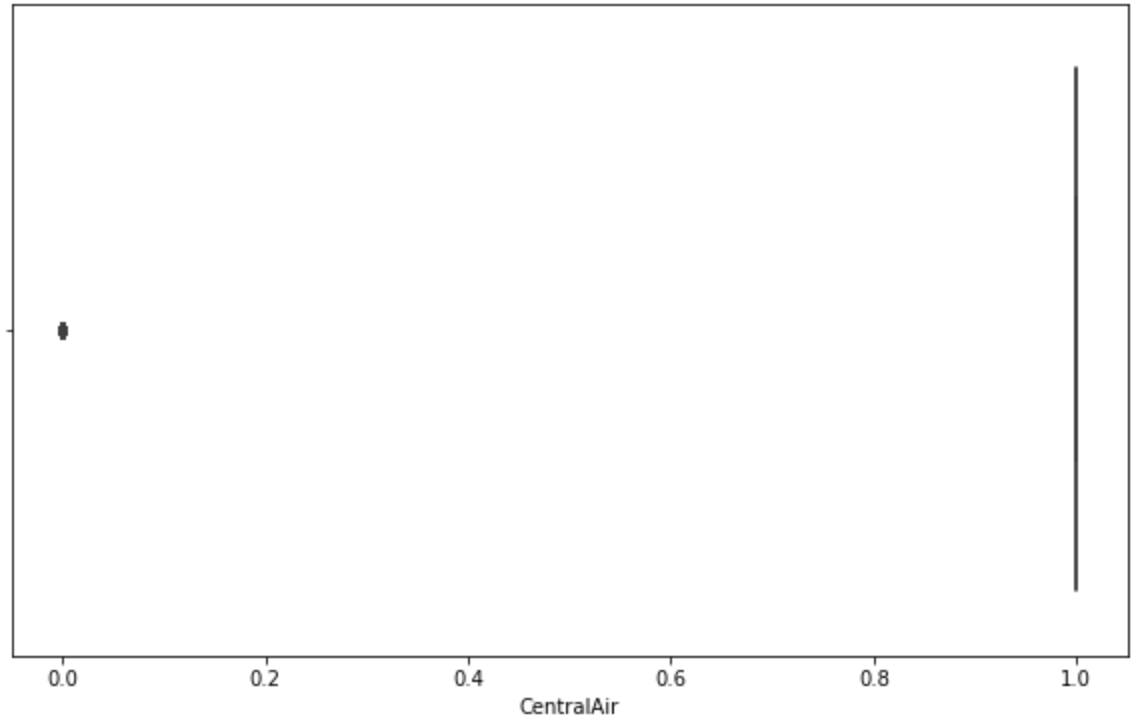

0
200
400
600
800
1000
1200
1400
BsmtFinSF2

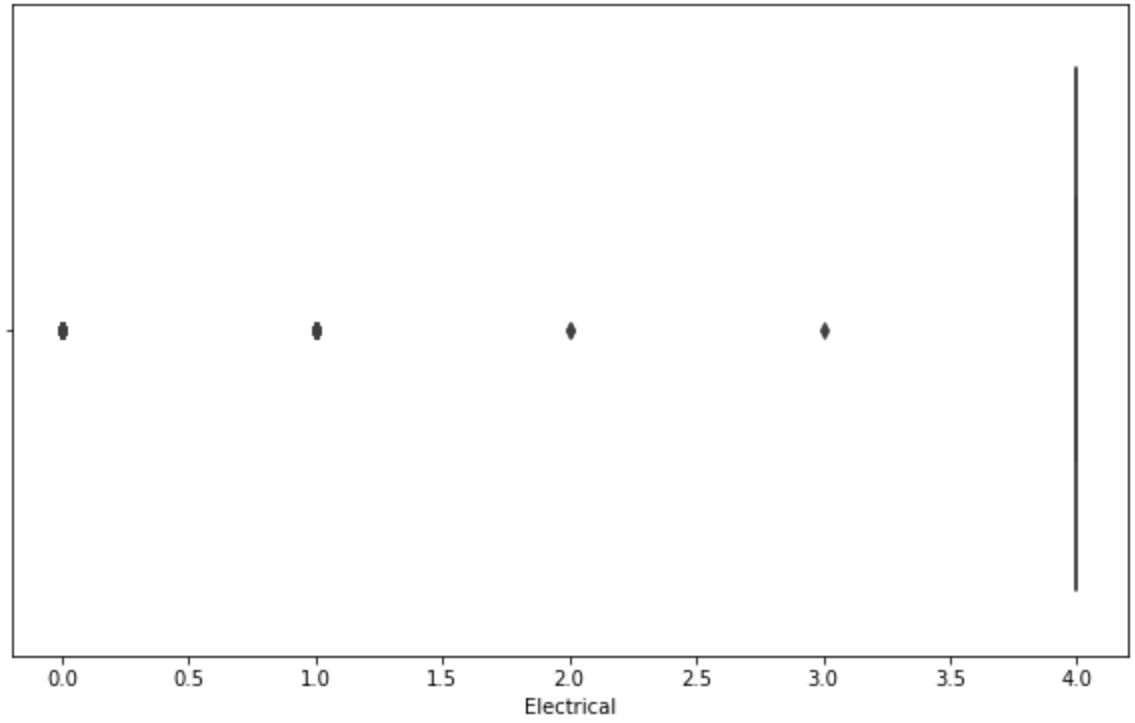

0
500
1000
1500
2000
BsmtUnfSF

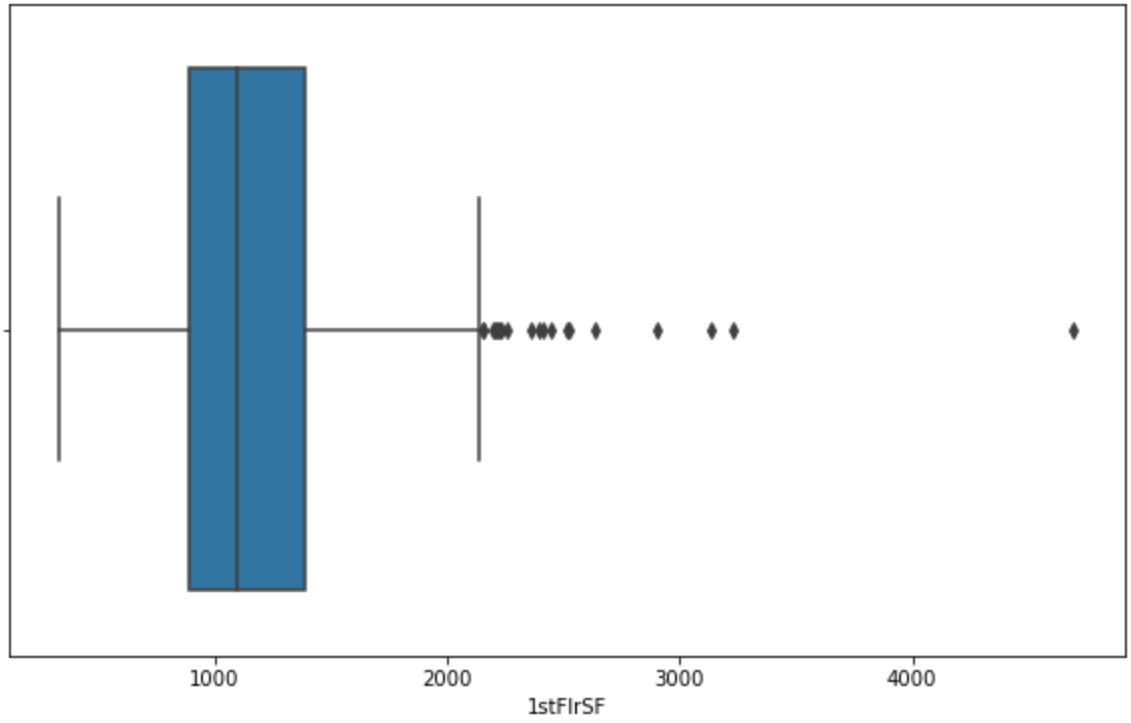

0
1000
2000
3000
4000
5000
6000
TotalBsmtSF

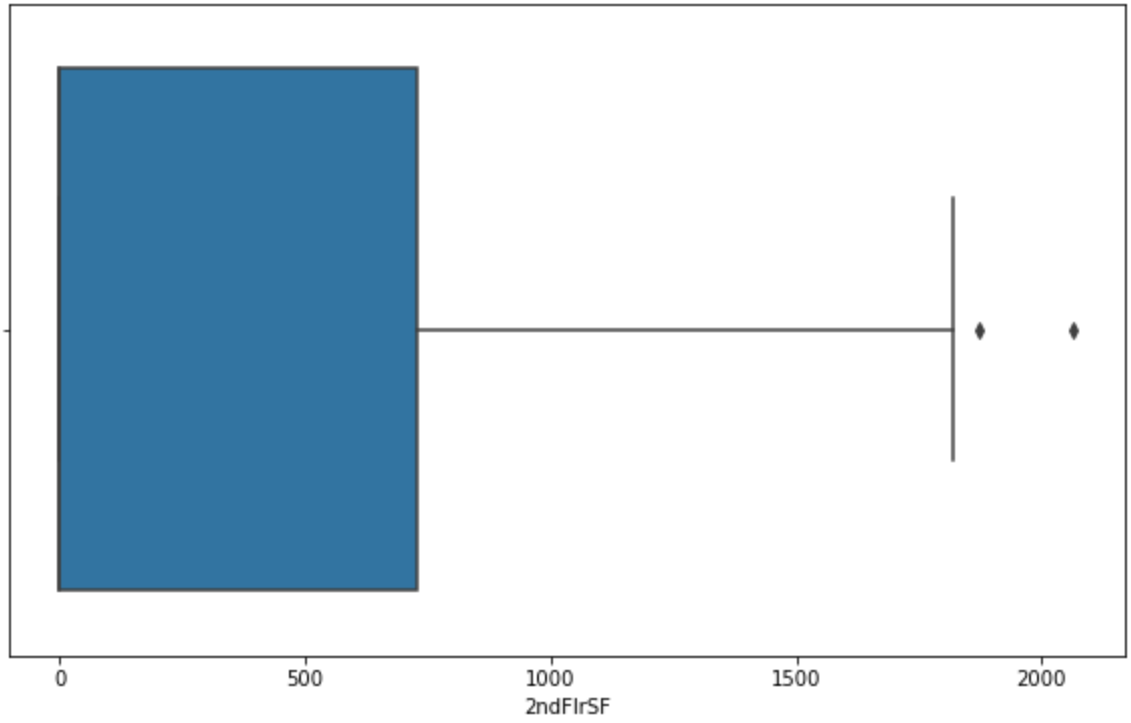

0
1
2
3
4
5
Heating

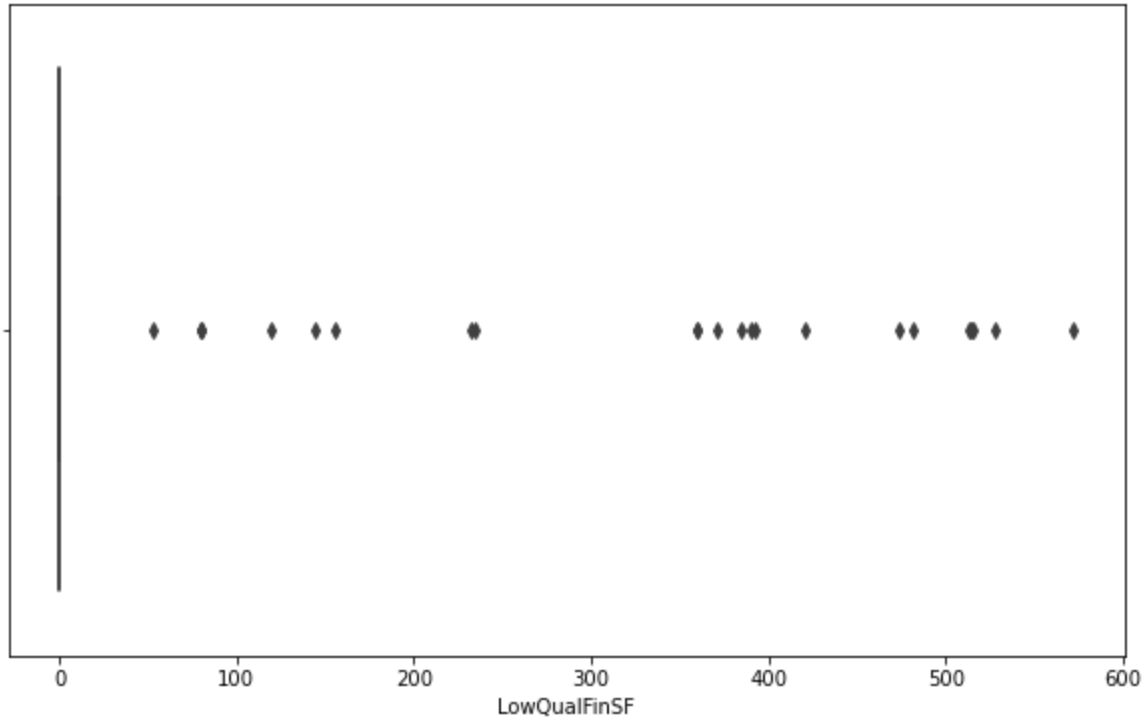

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
HeatingQC

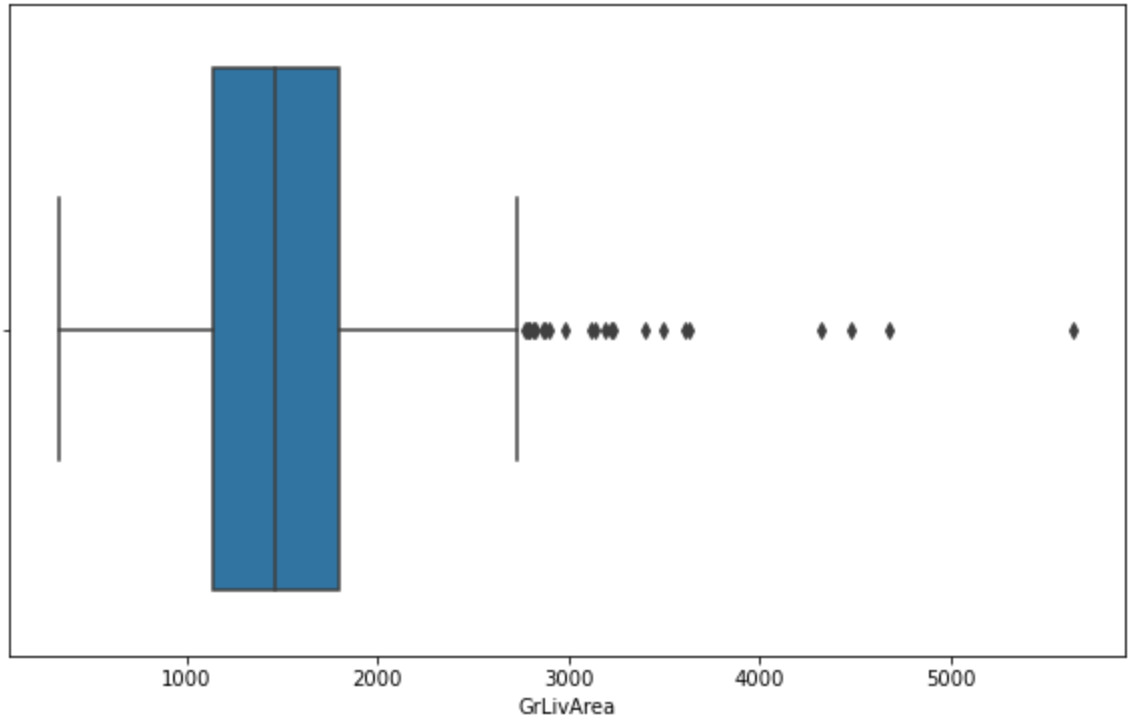

0.0
0.2
0.4
0.6
0.8
1.0
CentralAir

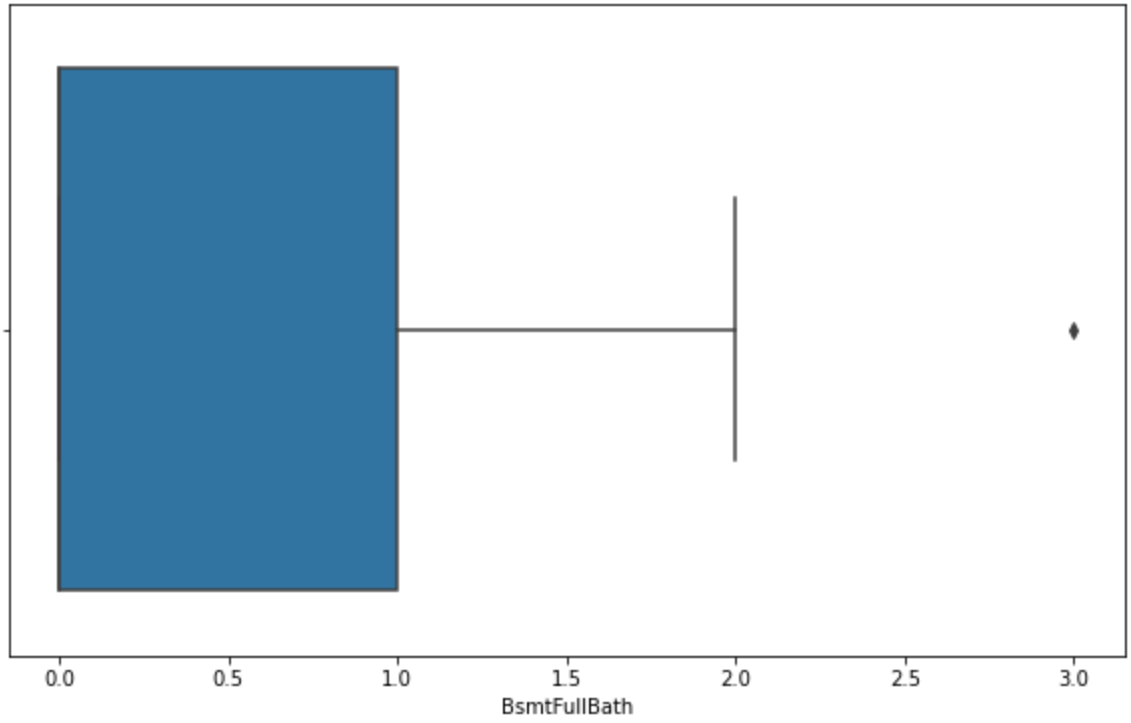

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Electrical

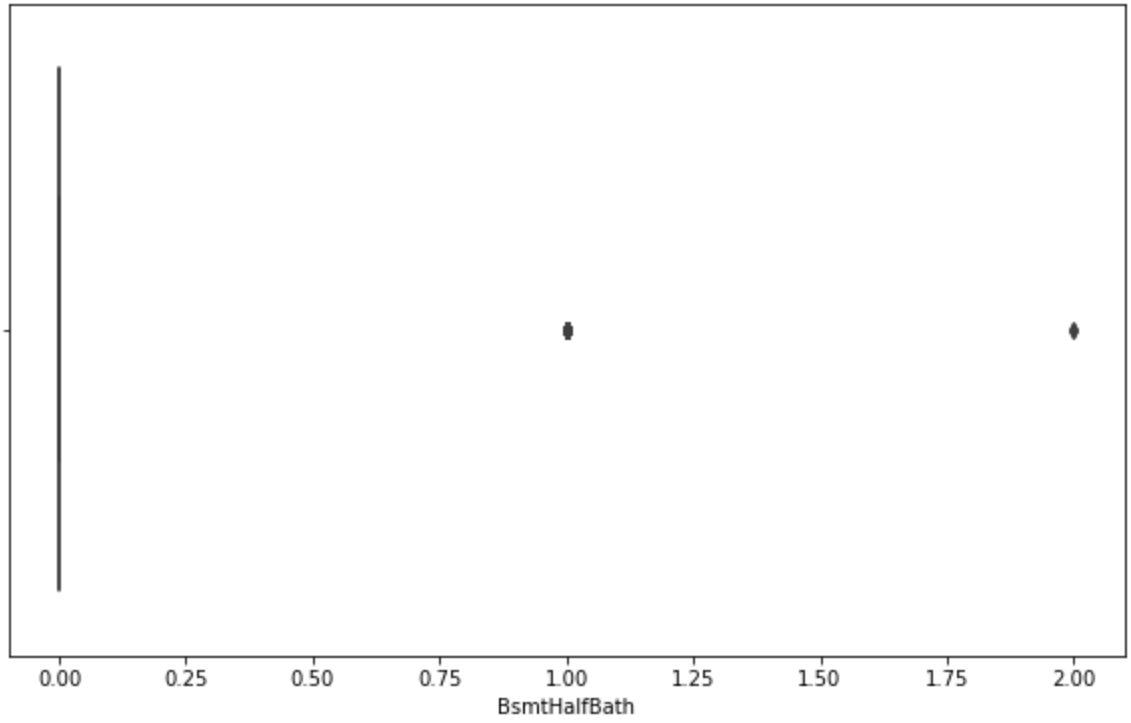

1000
2000
3000
4000
1stFlrSF

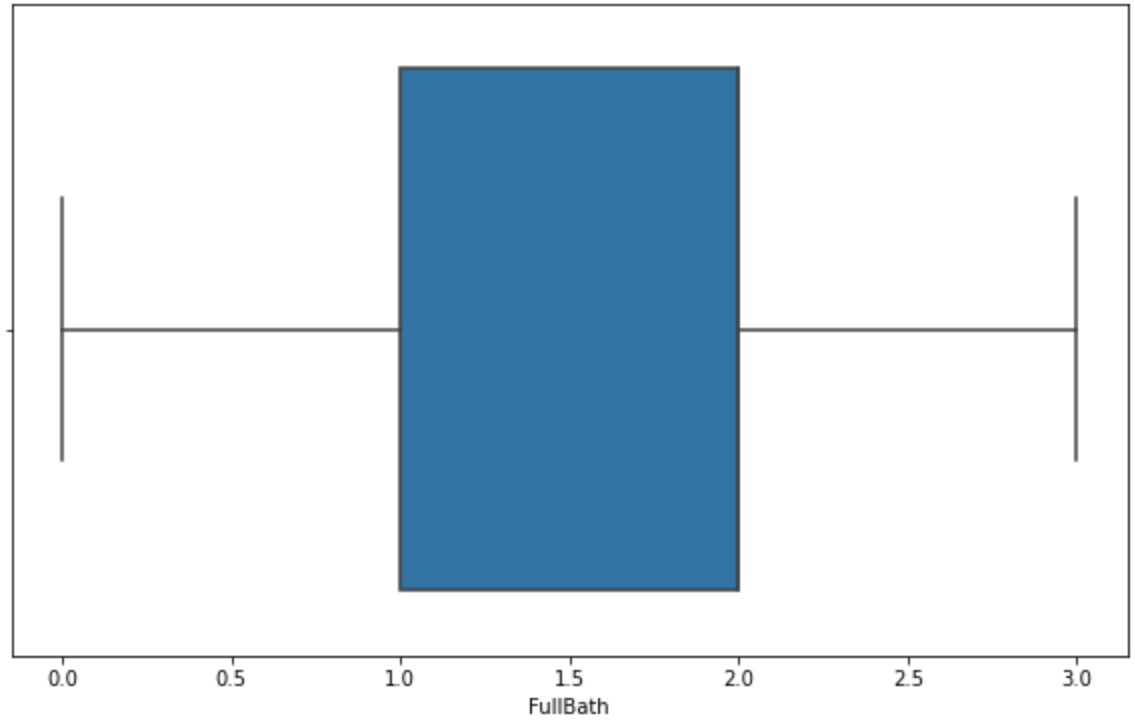

0
500
1000
1500
2000
2ndFlrSF

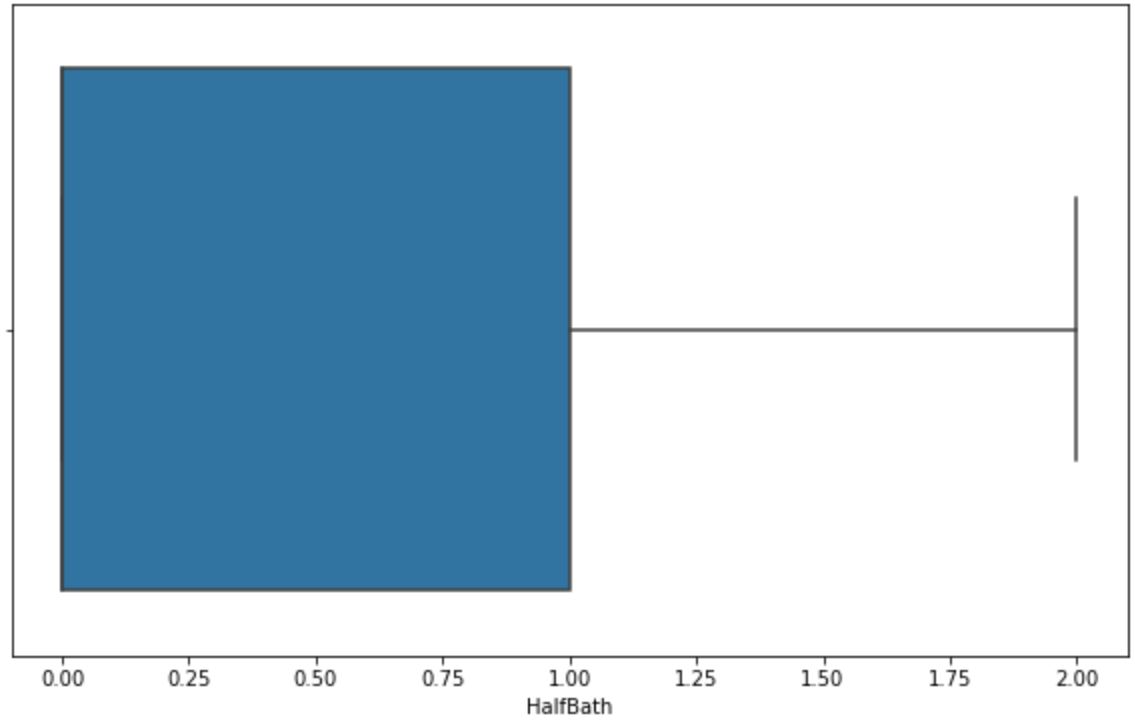

0
100
200
300
400
500
600
LowQualFinSF

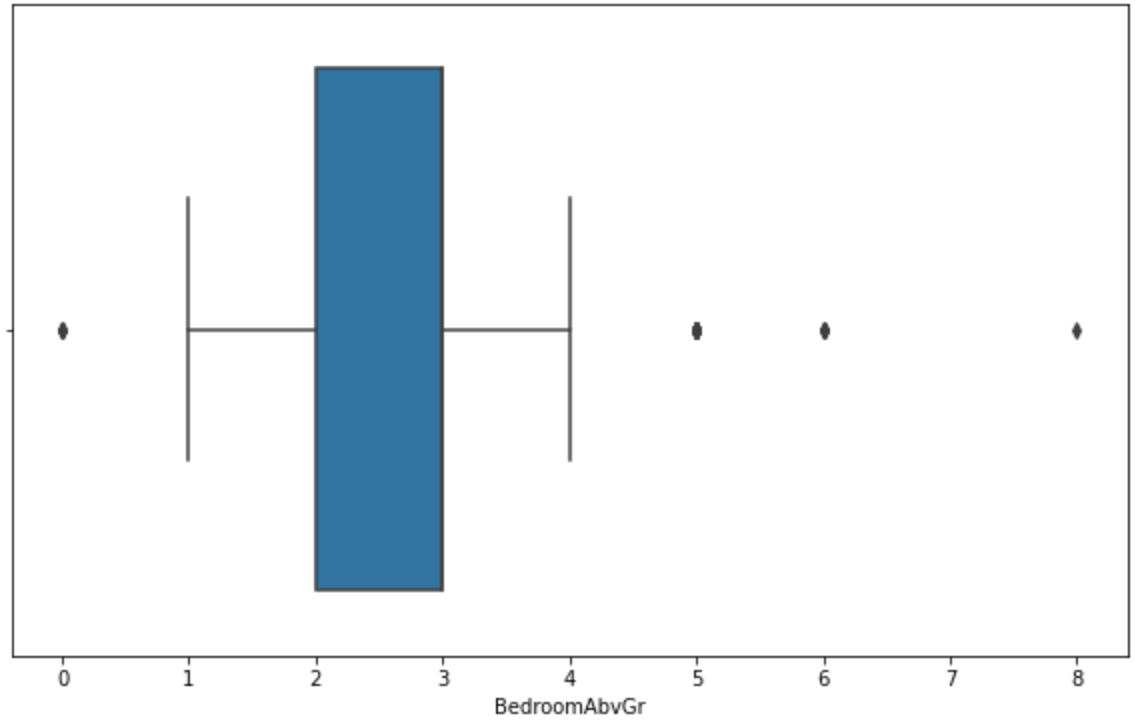

1000
2000
3000
4000
5000
GrLivArea

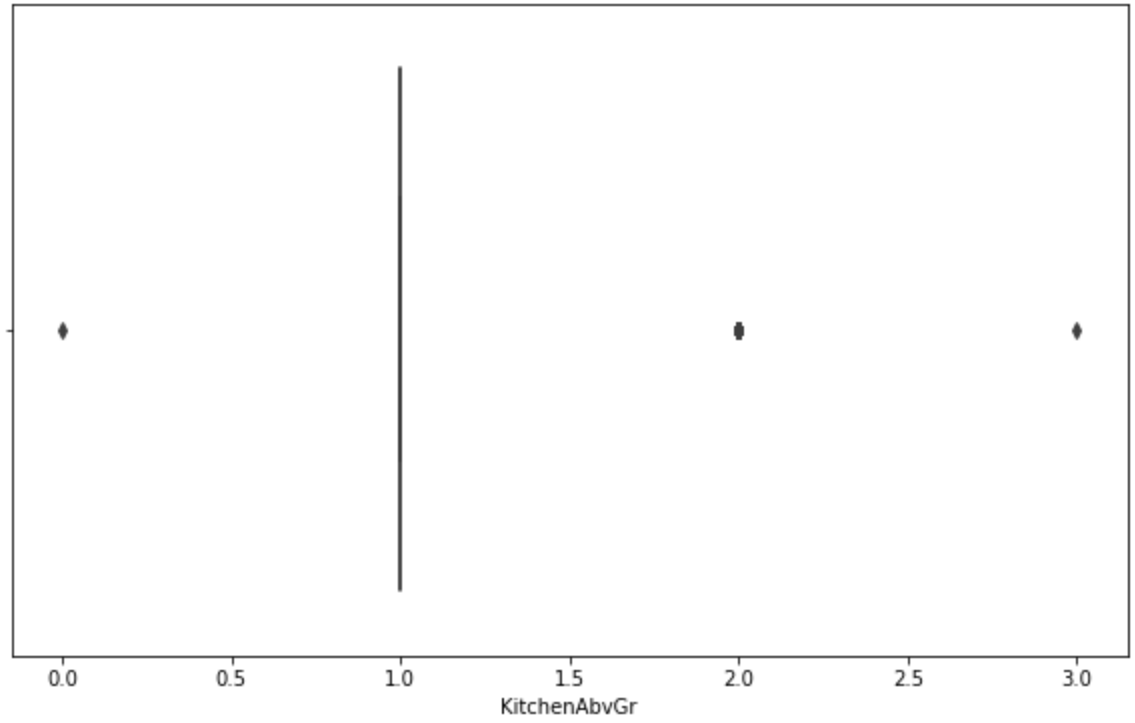

0.0
0.5
1.0
1.5
2.0
2.5
3.0
BsmtFullBath

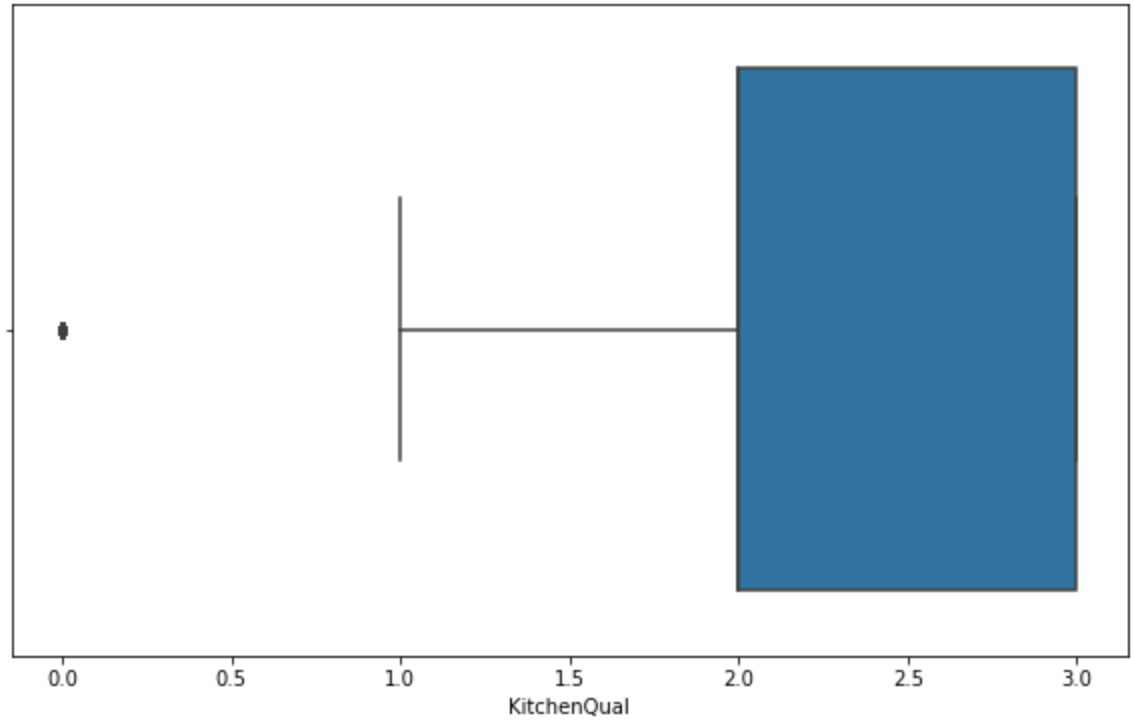

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
BsmtHalfBath

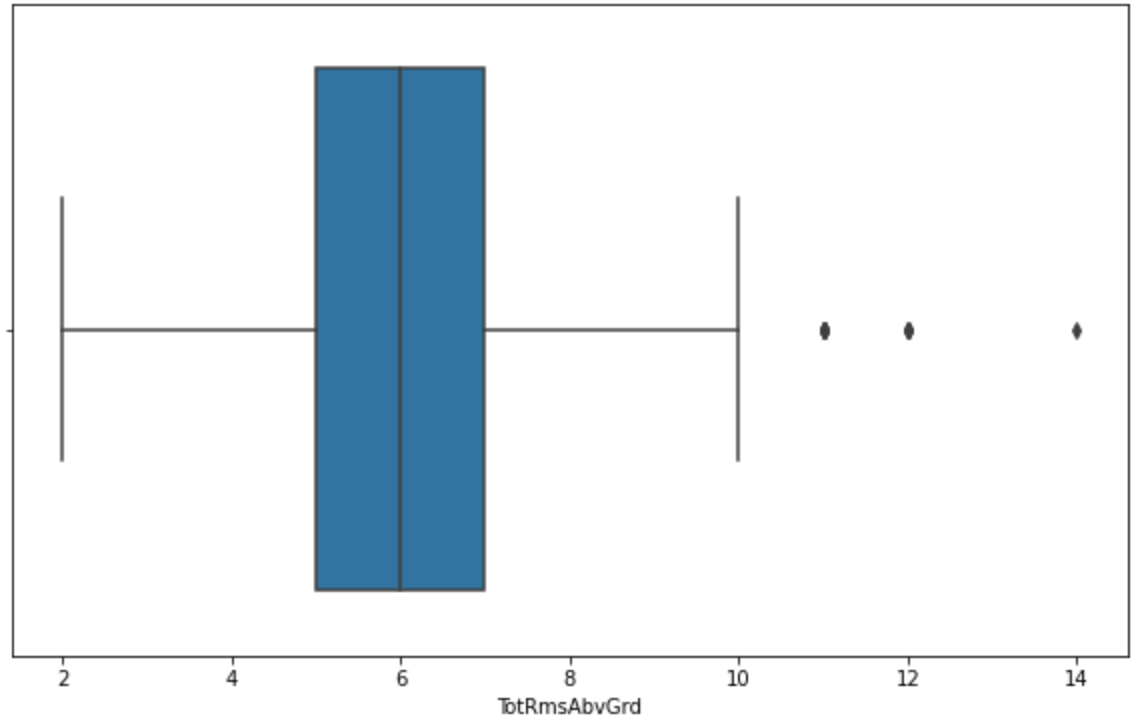

0.0
0.5
1.0
1.5
2.0
2.5
3.0
FullBath

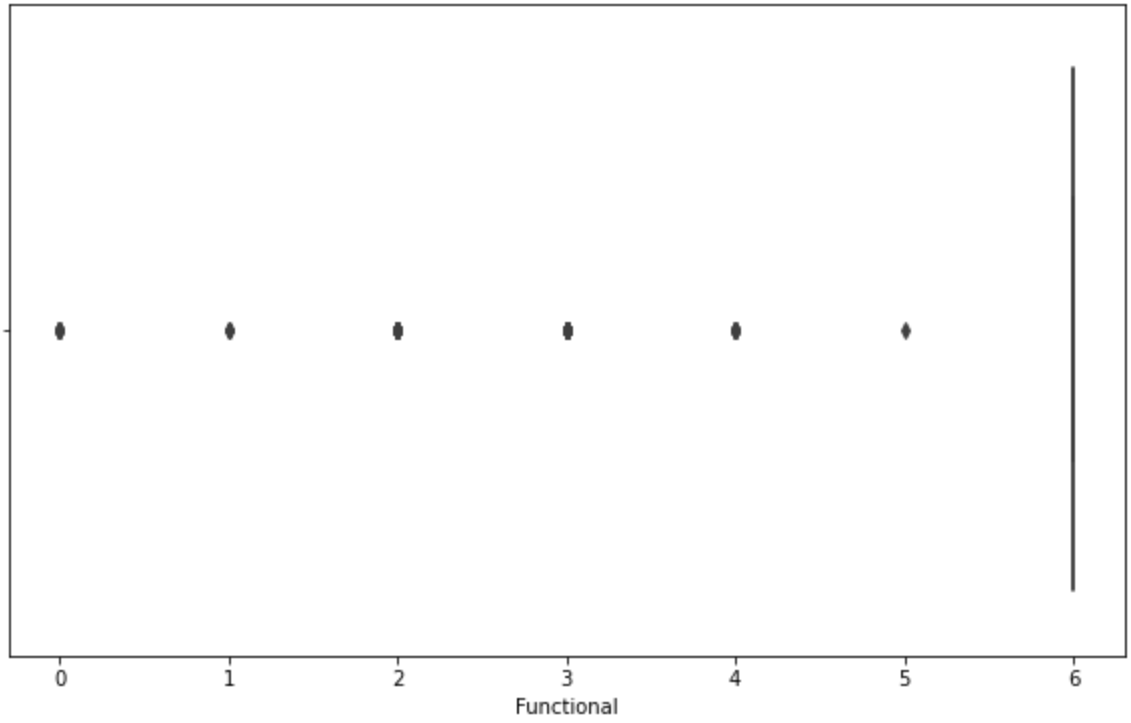

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
HalfBath

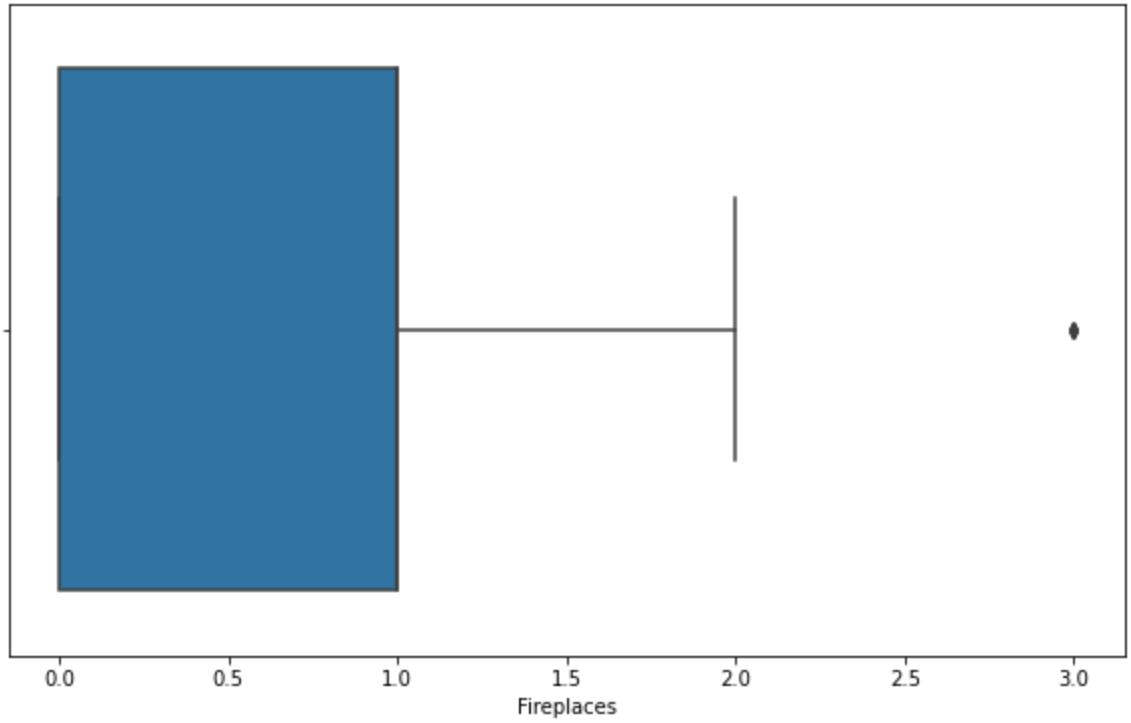

0
1
2
3
4
5
6
7
8
BedroomAbvGr

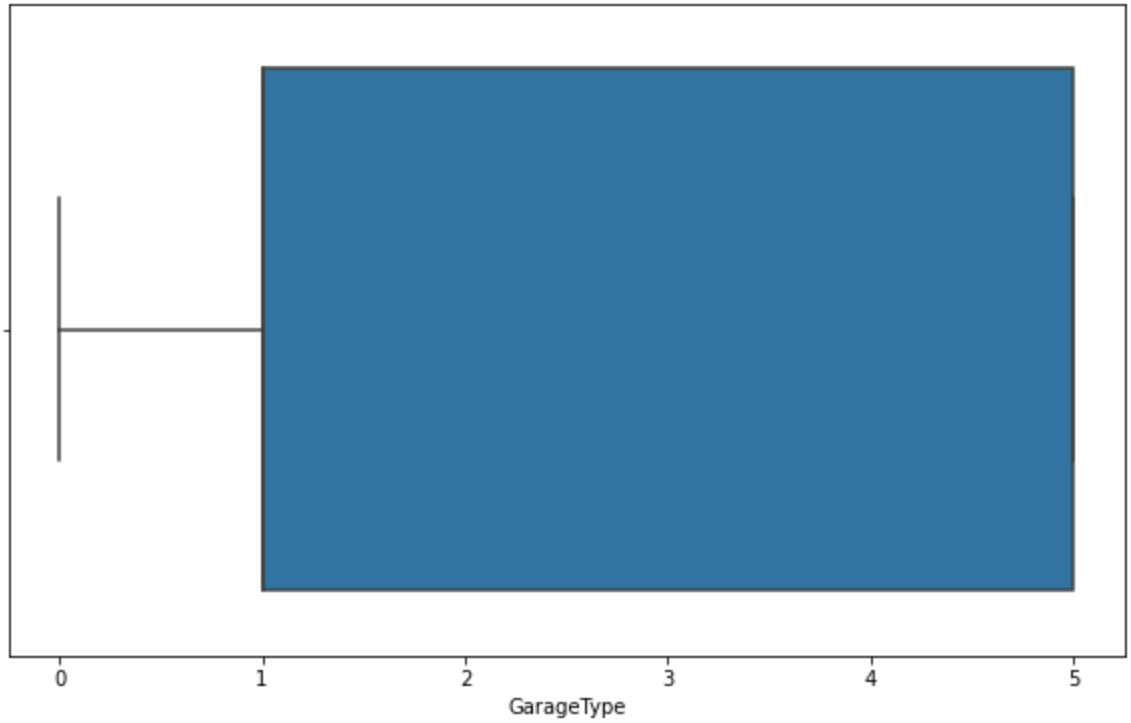

0.0
0.5
1.0
1.5
2.0
2.5
3.0
KitchenAbvGr

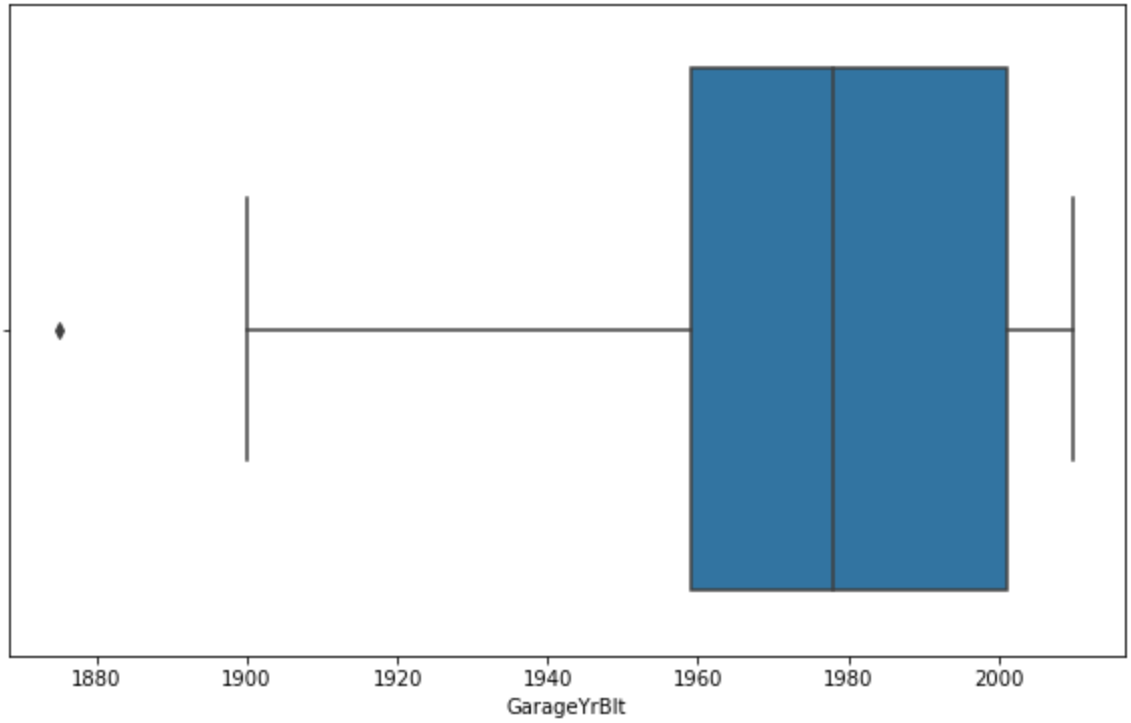

0.0
0.5
1.0
1.5
2.0
2.5
3.0
KitchenQual

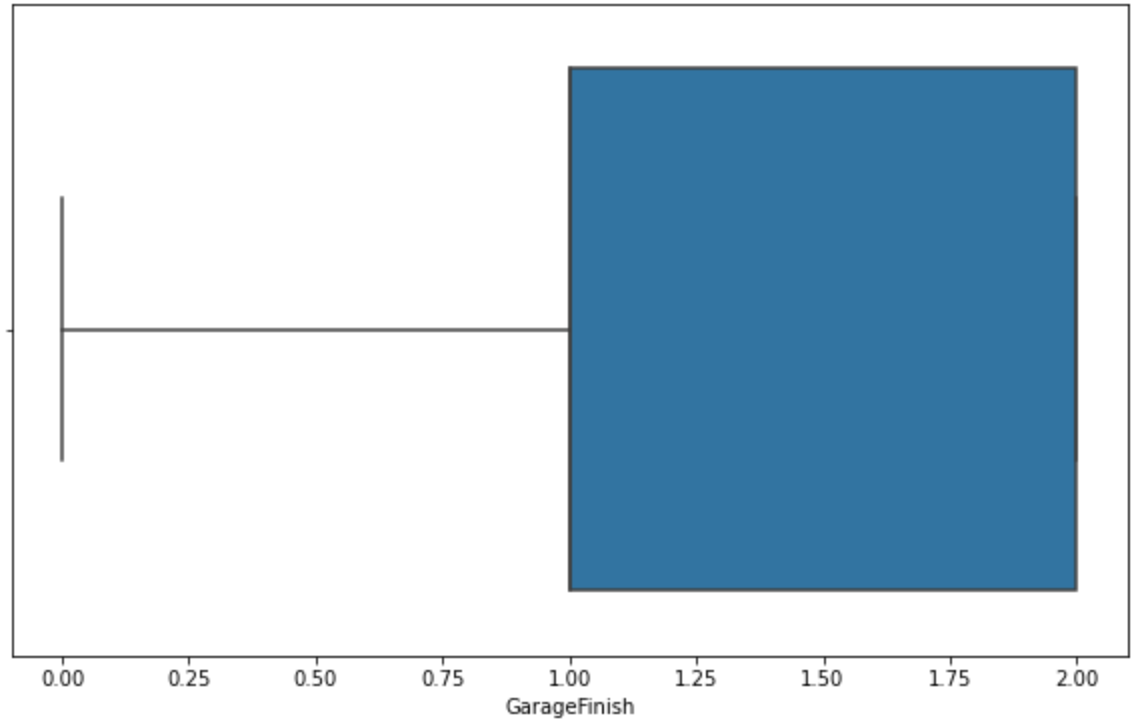

2
4
6
8
10
12
14
TotRmsAbvGrd

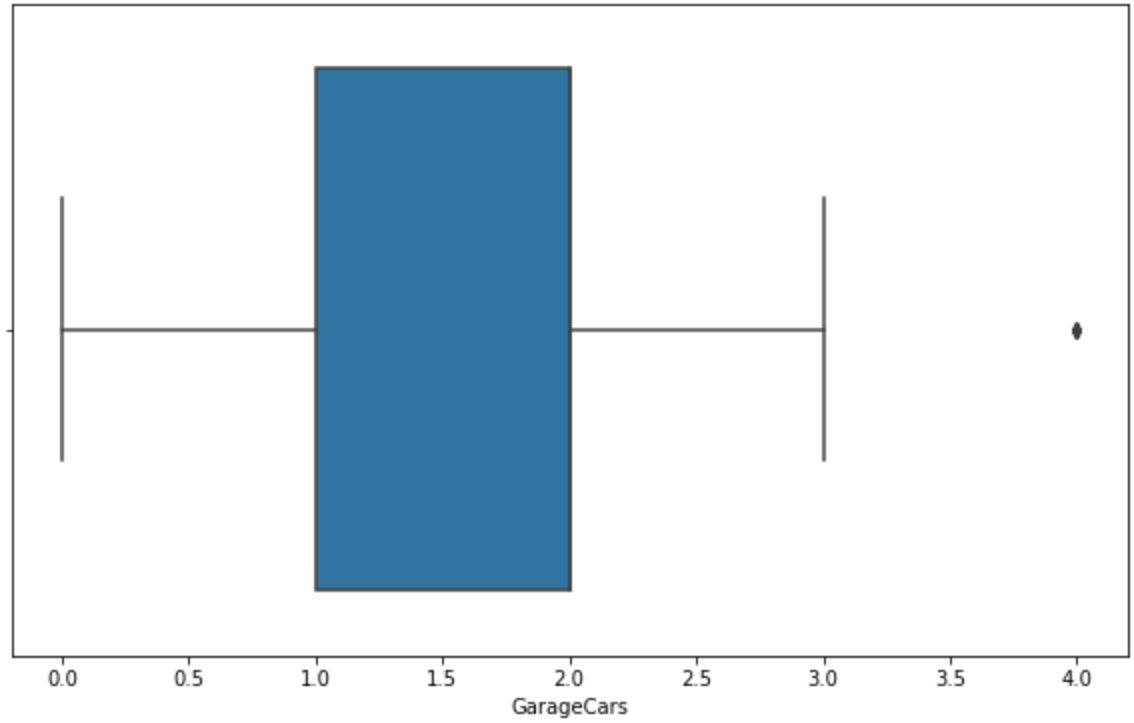

0
1
2
3
4
5
6
Functional

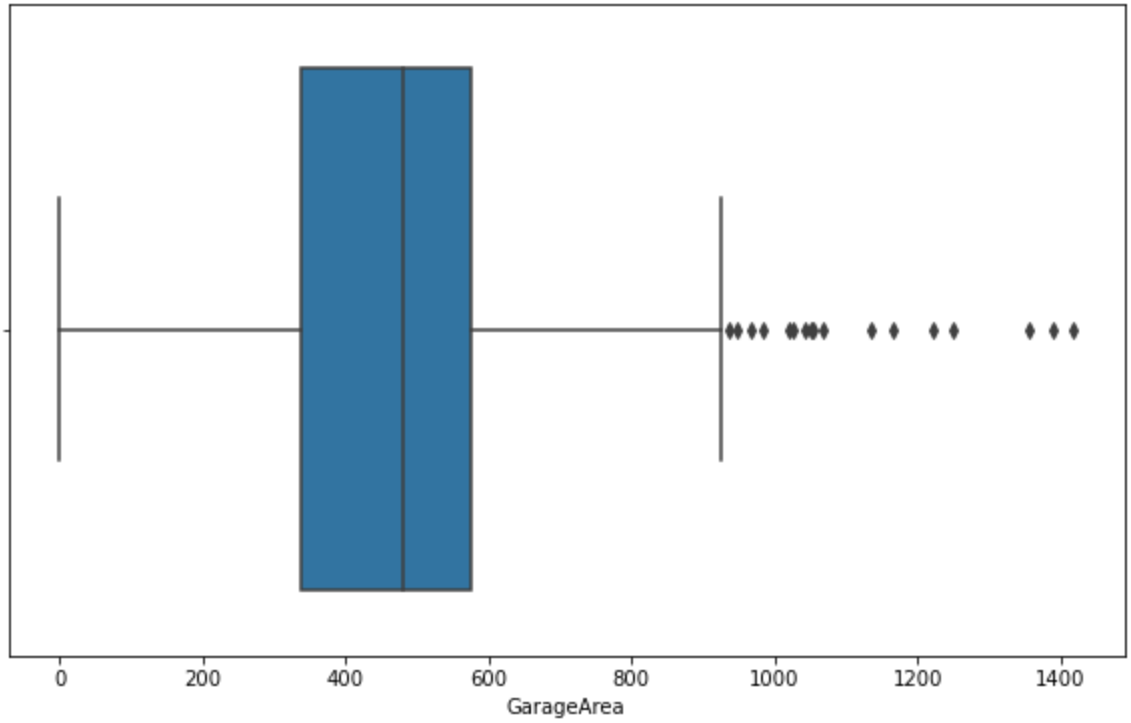

0.0
0.5
1.0
1.5
2.0
2.5
3.0
Fireplaces

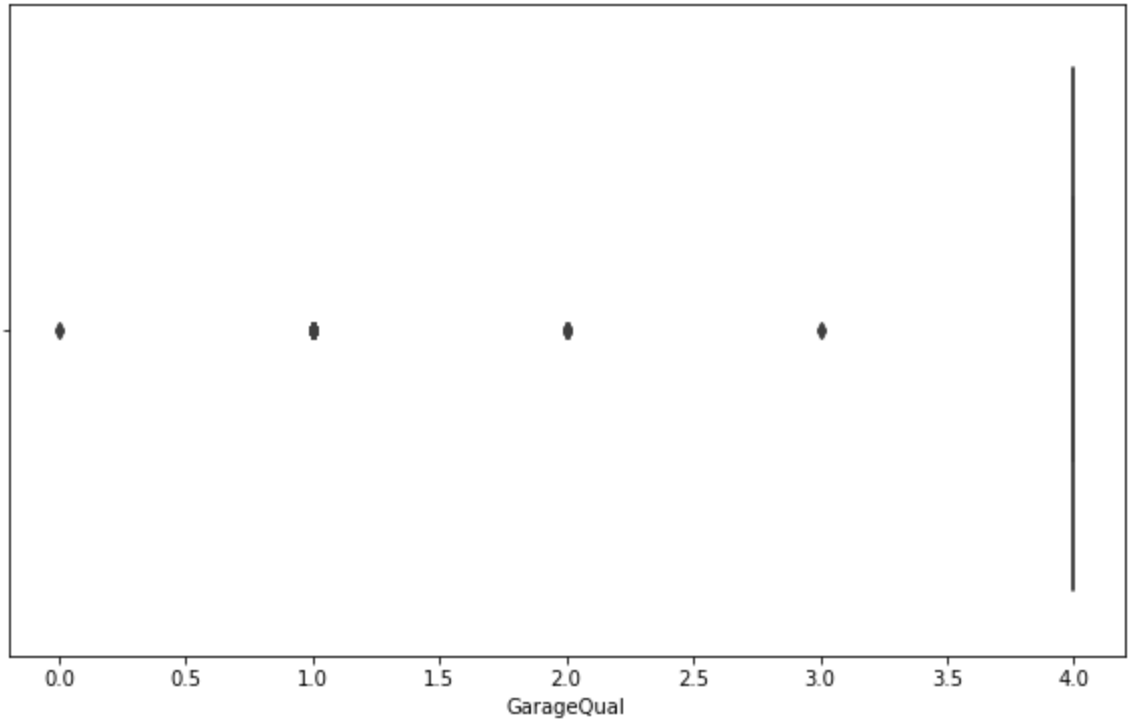

0
1
2
3
4
5
GarageType

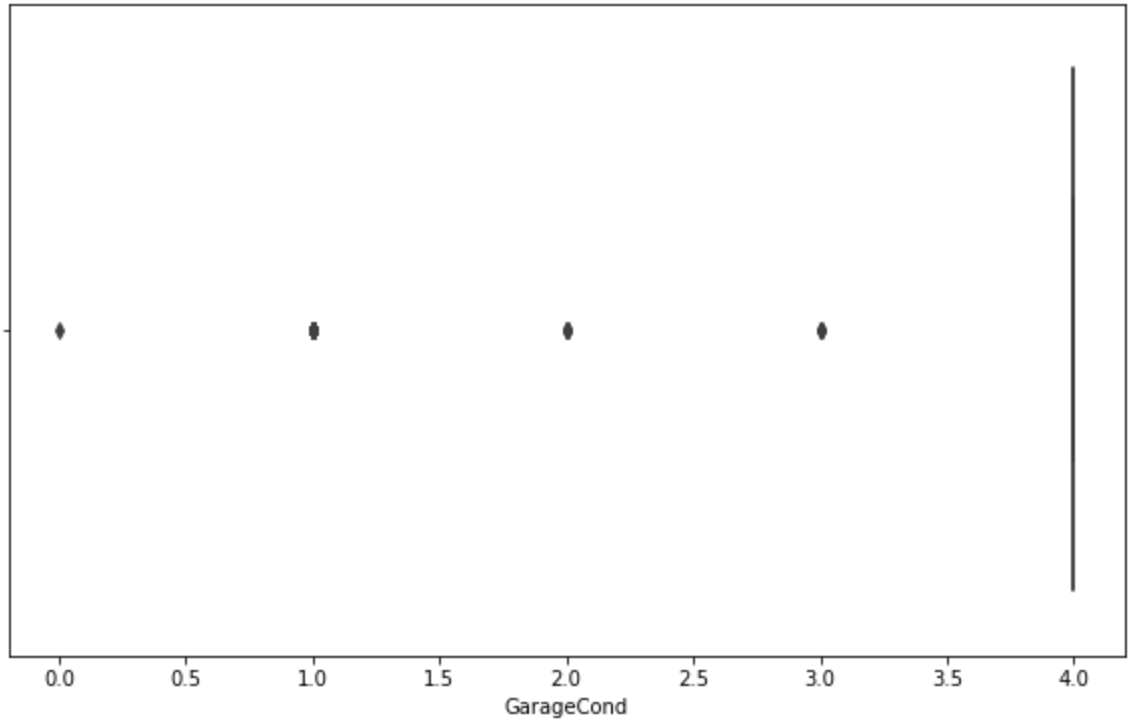

1880
1900
1920
1940
1960
1980
2000
GarageYrBlt

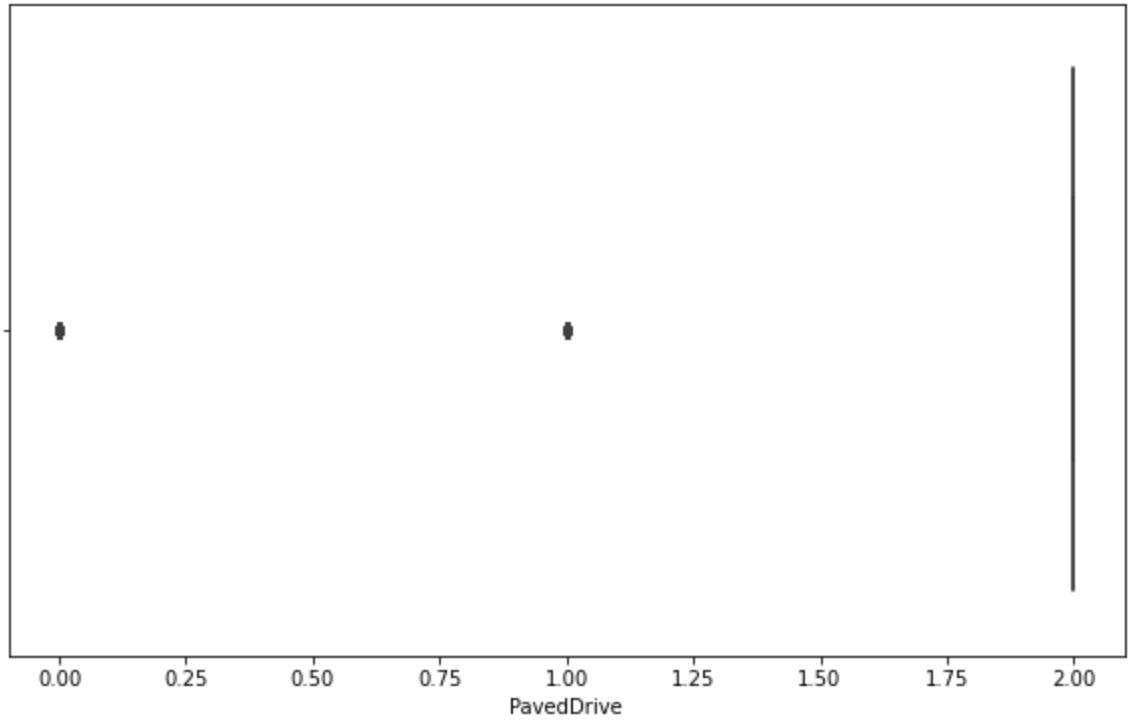

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
GarageFinish

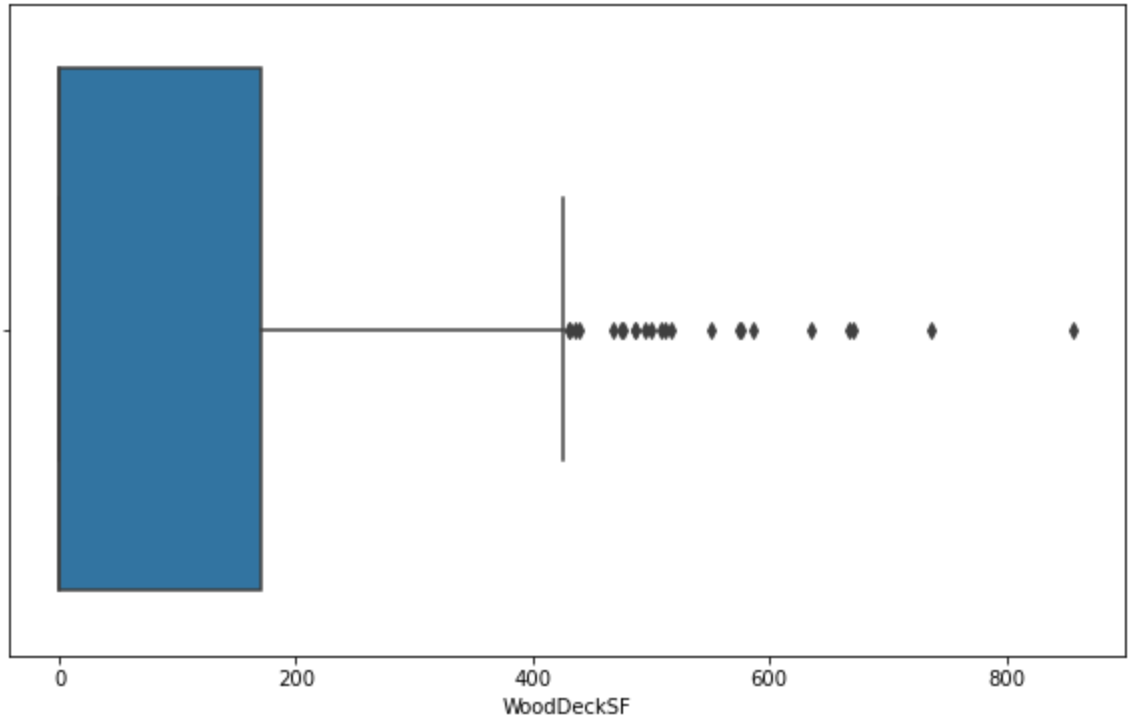

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageCars

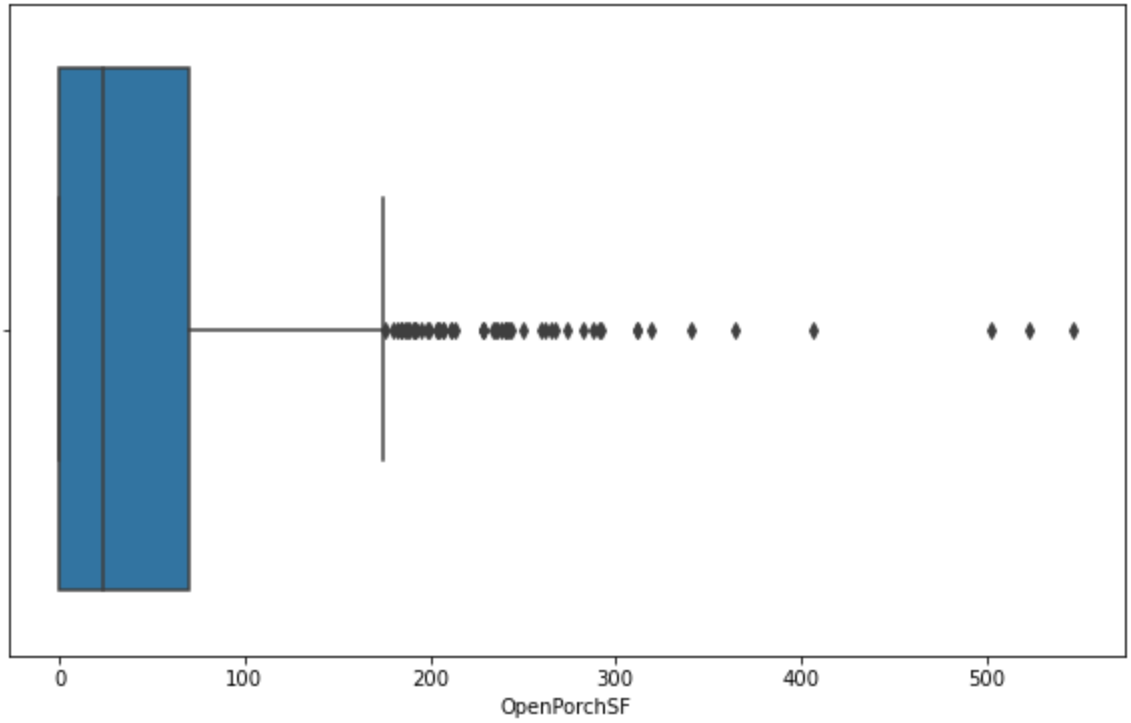

0
200
400
600
800
1000
1200
1400
GarageArea

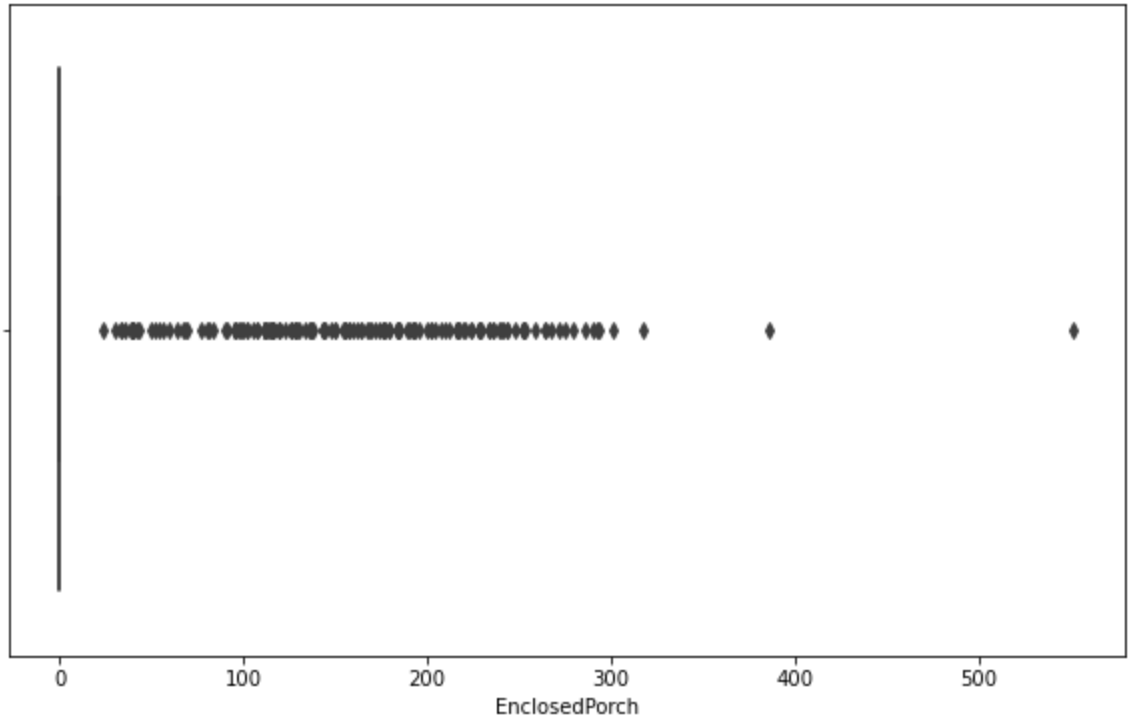

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageQual

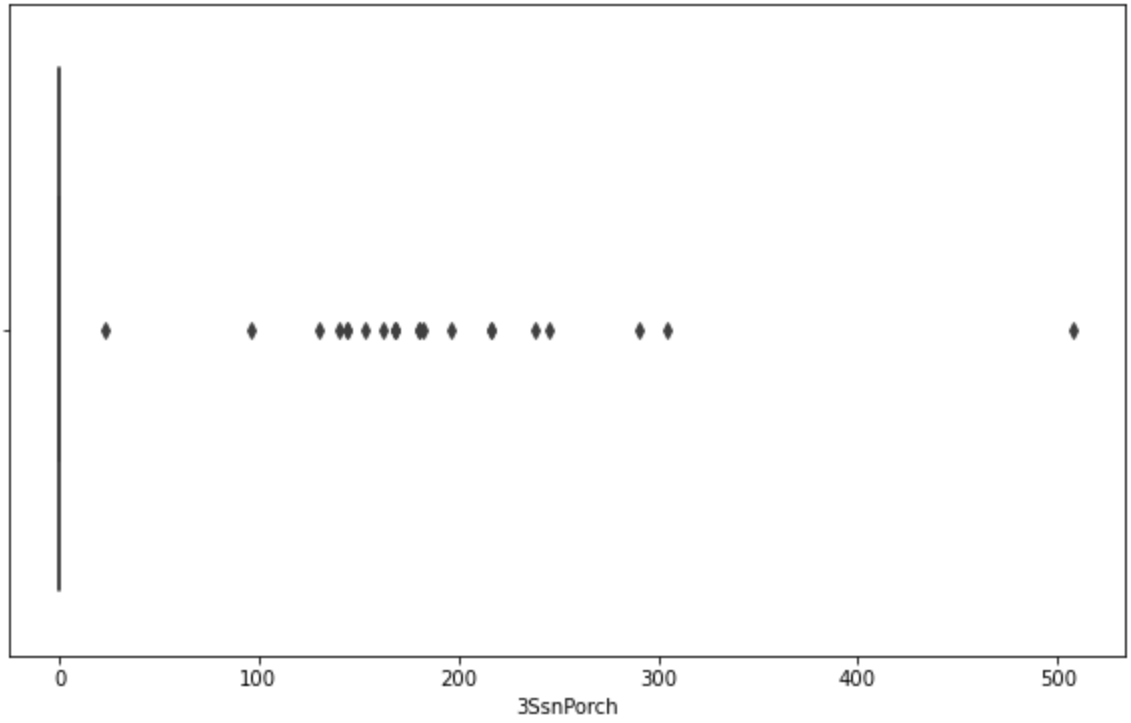

0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
GarageCond

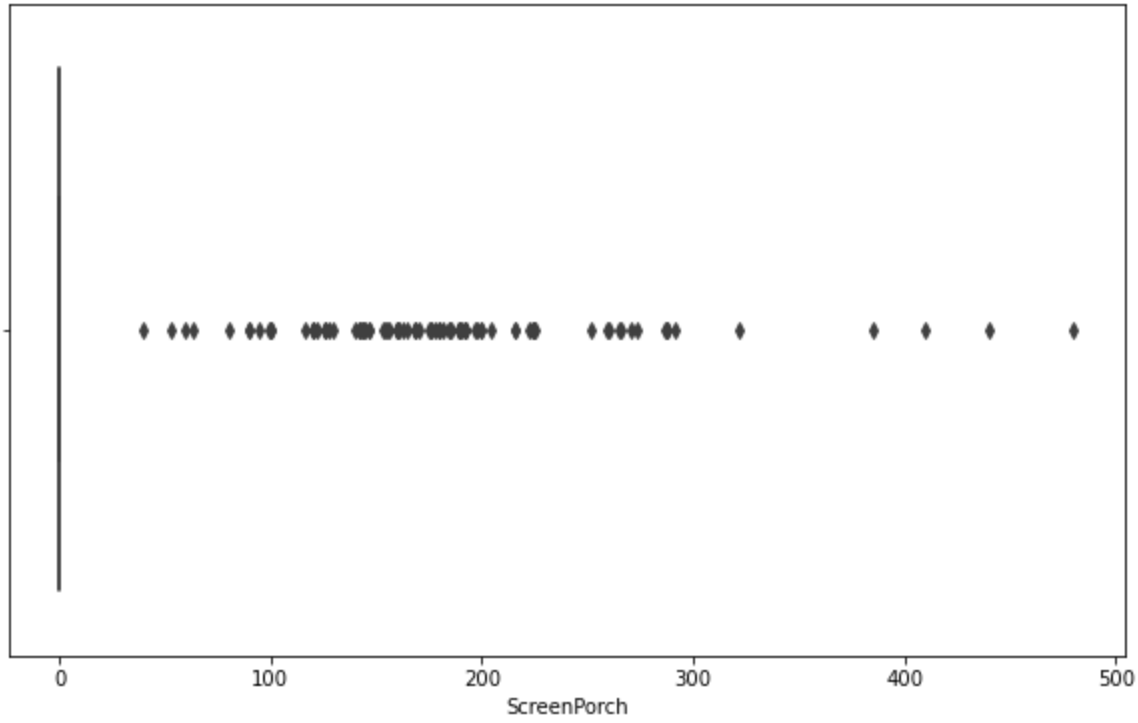

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
PavedDrive

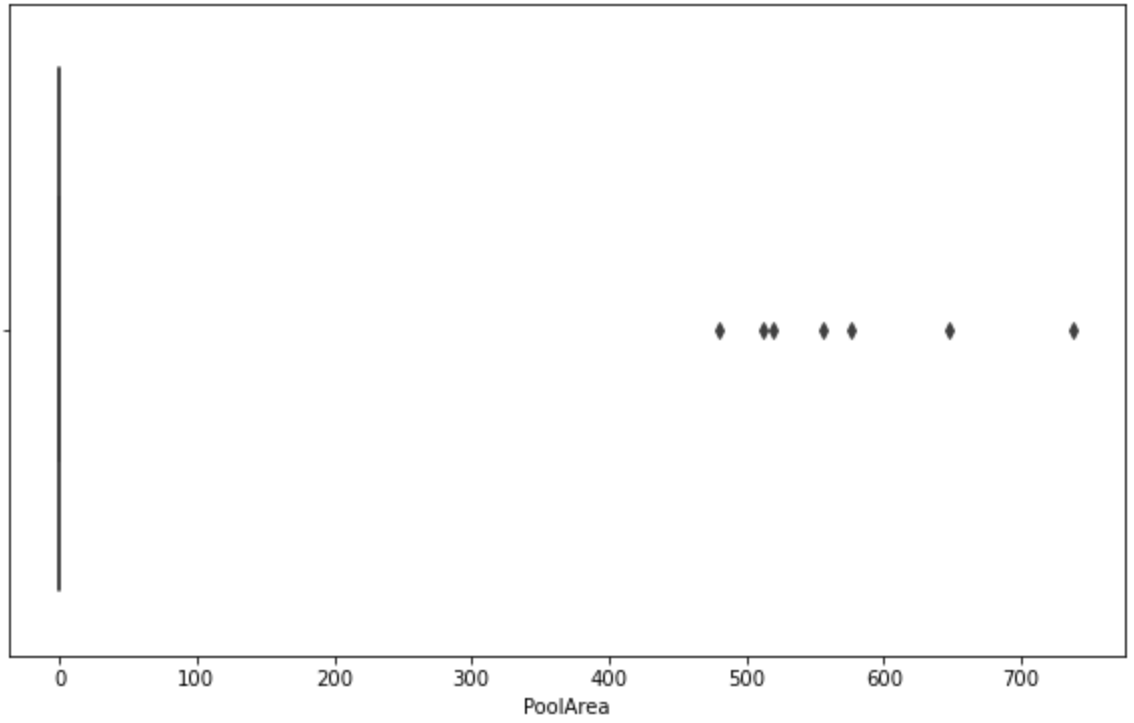

0
200
400
600
800
WoodDeckSF

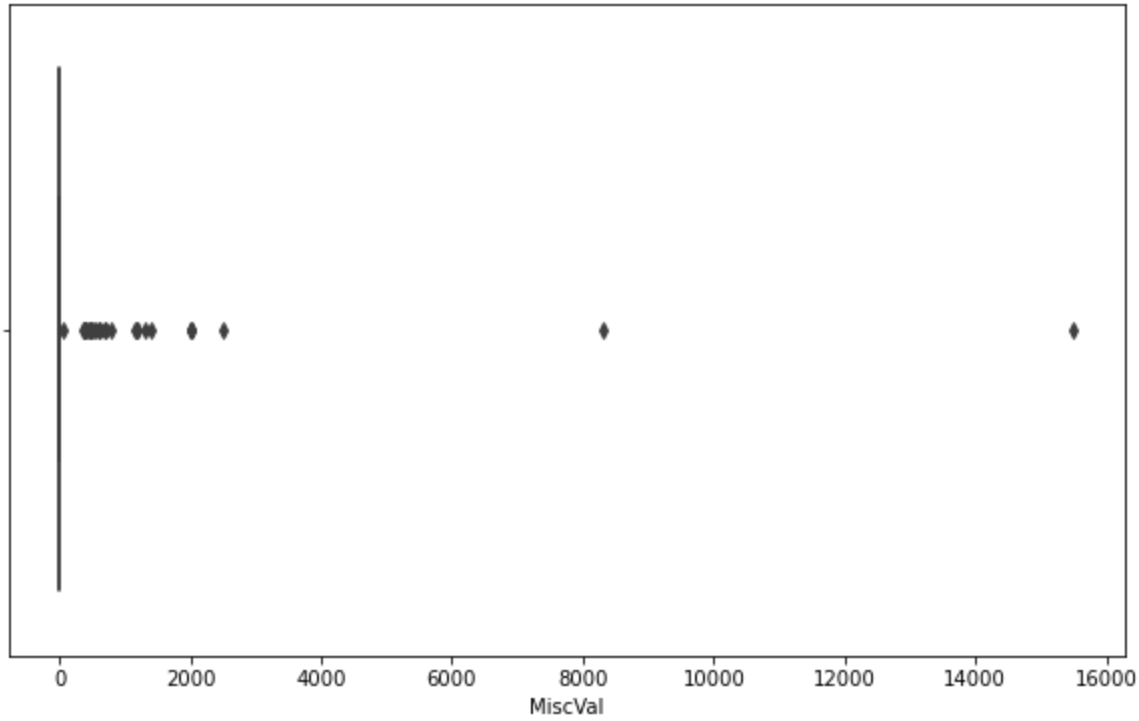

0
100
200
300
400
500
OpenPorchSF

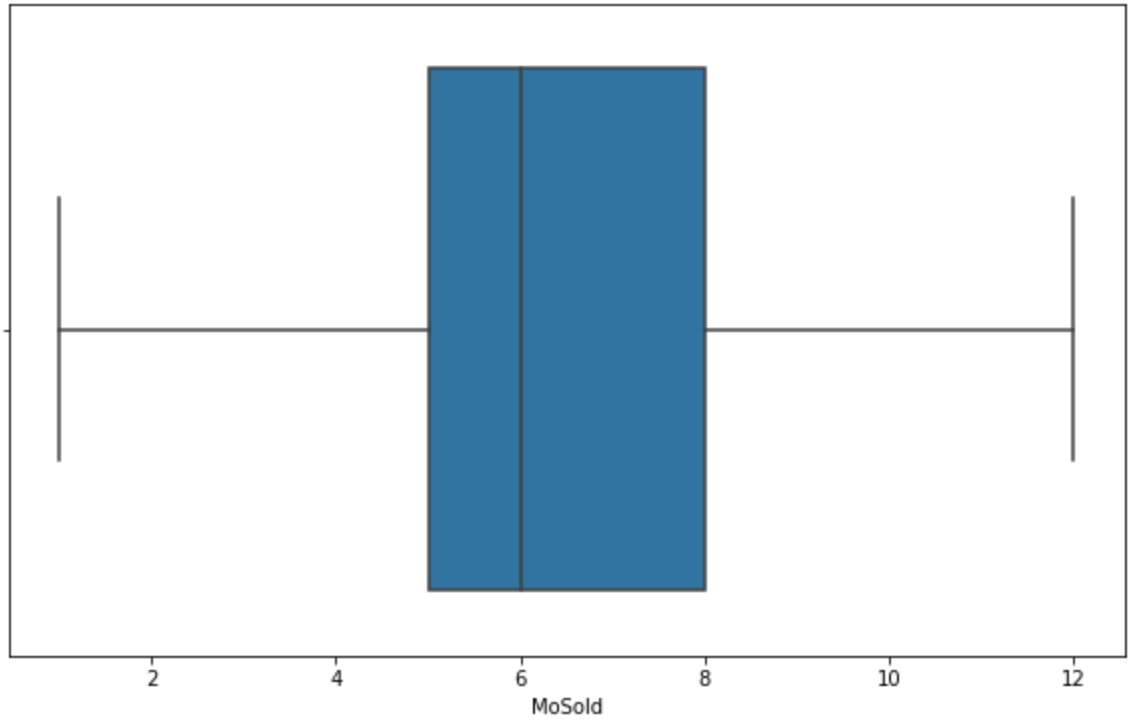

0
100
200
300
400
500
EnclosedPorch

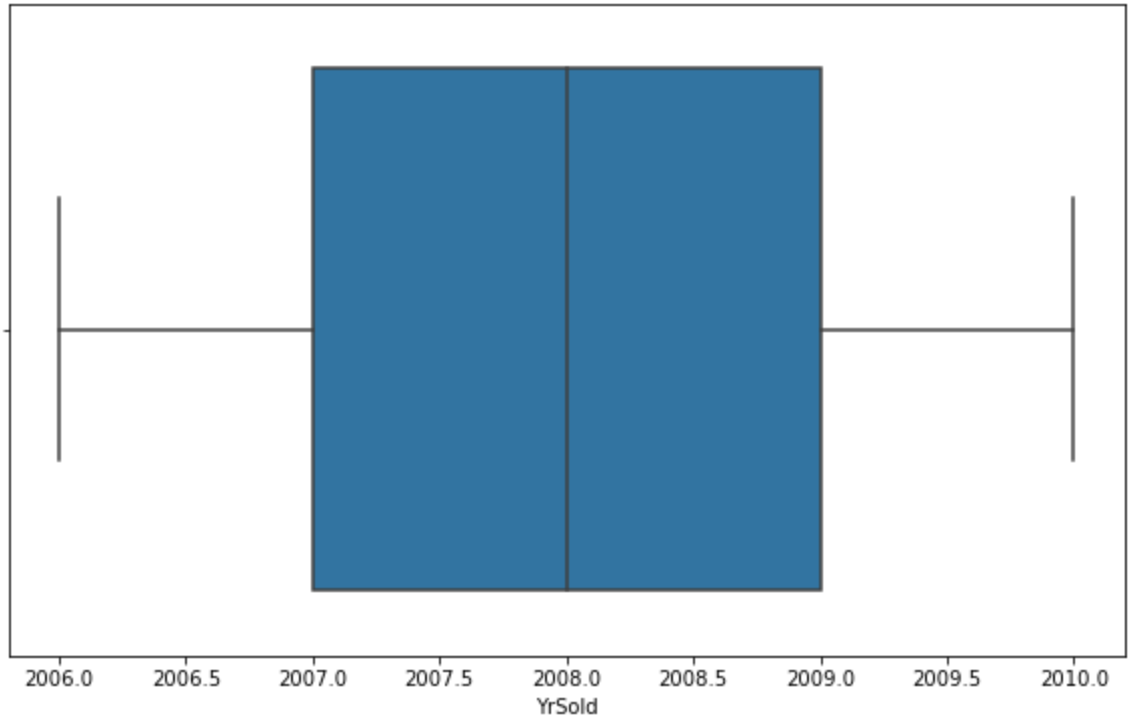

0
100
200
300
400
500
3SsnPorch

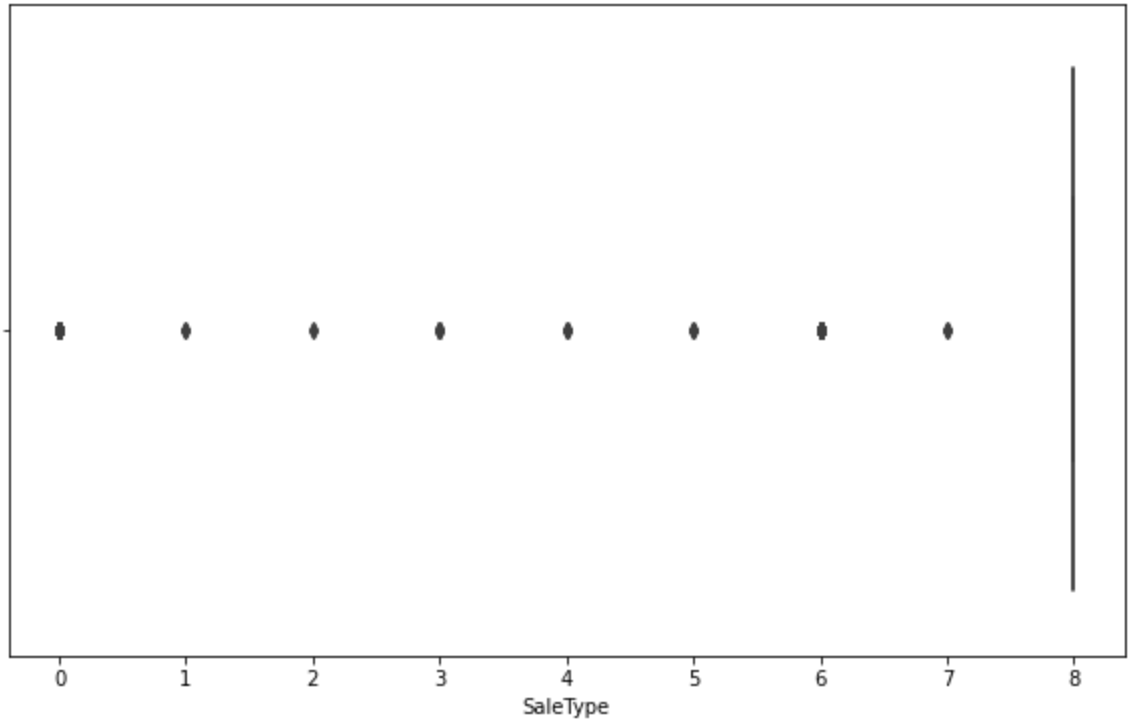

0
100
200
300
400
500
ScreenPorch

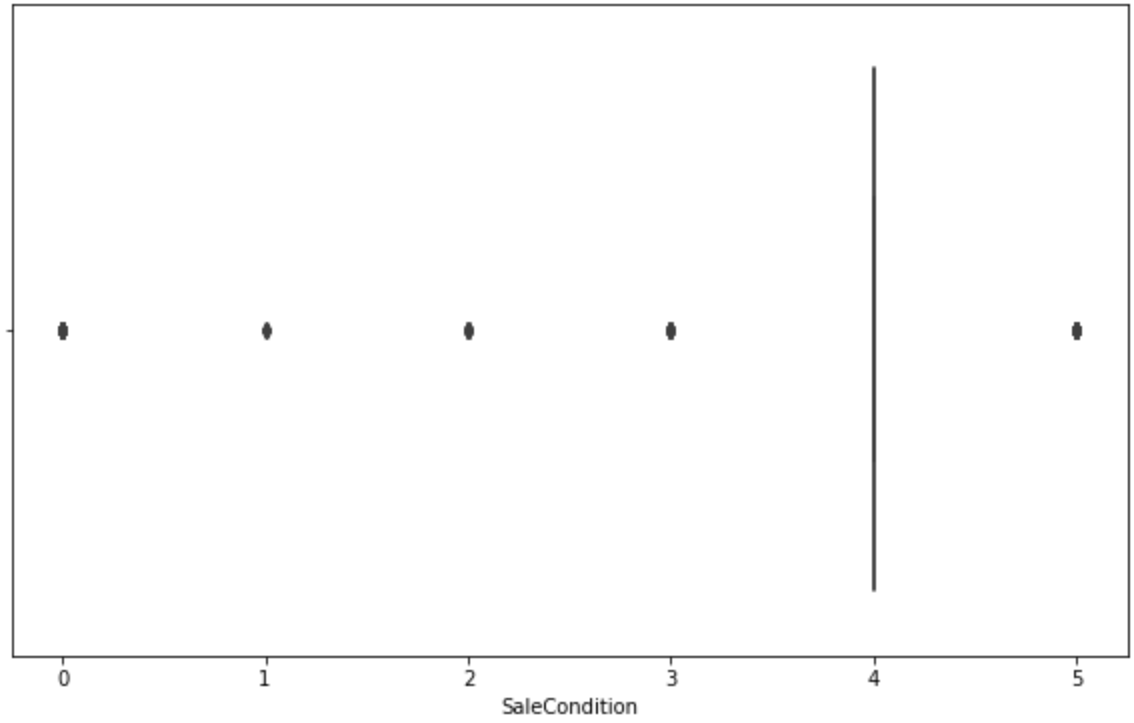

0
100
200
300
400
500
600
700
PoolArea


0
2000
4000
6000
8000
10000
12000
14000
16000
MiscVal


2
4
6
8
10
12
MoSold


2006.0
2006.5
2007.0
2007.5
2008.0
2008.5
2009.0
2009.5
2010.0
YrSold


0
1
2
3
4
5
6
7
8
SaleType



Oulier Removal

In [47]:
```
from scipy.stats import zscore
z=np.abs(zscore(x))
z.shape
```

Out[47]:
```
(1168, 73)
```

In [48]:
```
np.where(z>3)
```

Out[48]:
```
(array([   1,    1,    1, ..., 1166, 1166, 1166], dtype=int64),
 array([ 8, 19, 33, ..., 38, 60, 61], dtype=int64))
```

In [49]:
```
nd=x[(z<=3).all(axis=1)]
print(x.shape,'\n',nd.shape)
```

```
(1168, 73)
 (484, 73)
```

Data loss

In [50]:
```
loss=(1168-484)/1168*100
loss
```

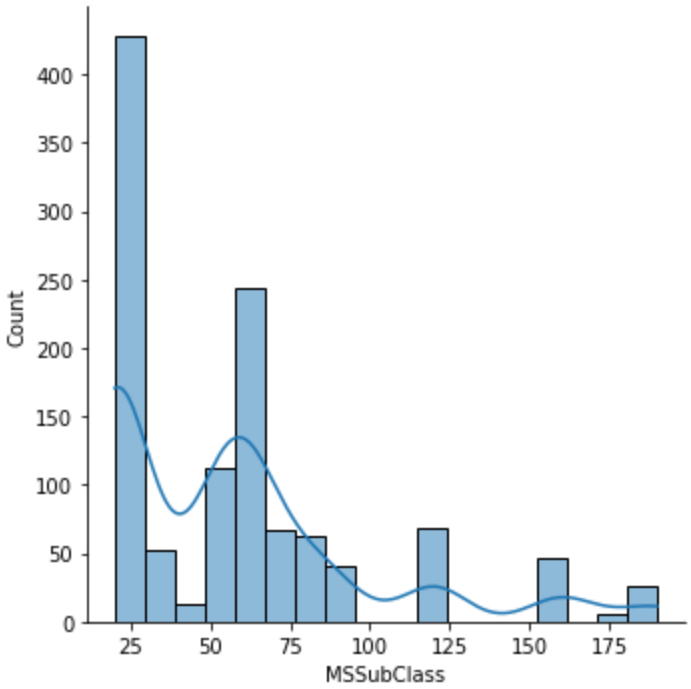
Out[50]:
```
58.56164383561644
```

59% of data loss is not acceptable. Hence we will not remove outliers.

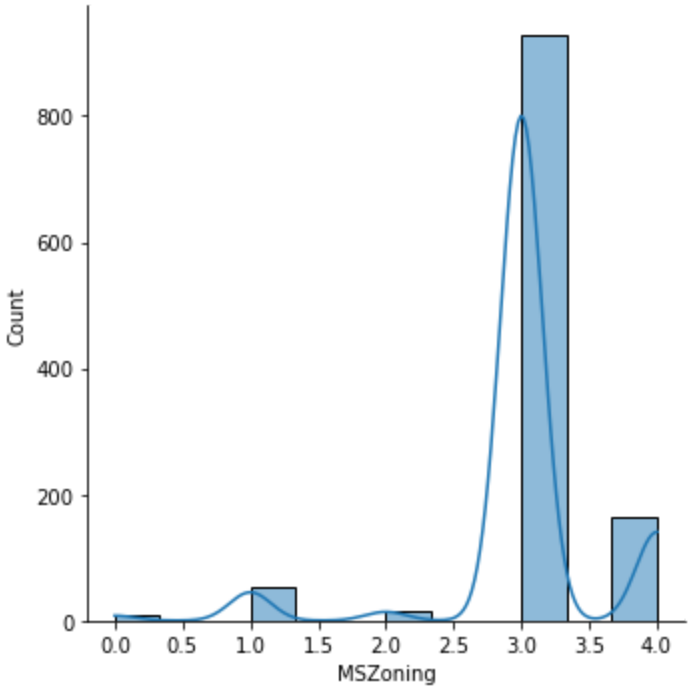
Skewness

In [51]:
```
for i in x:
    plt.figure(figsize=(20,8))
    sns.displot(x[i], kde=True)
```

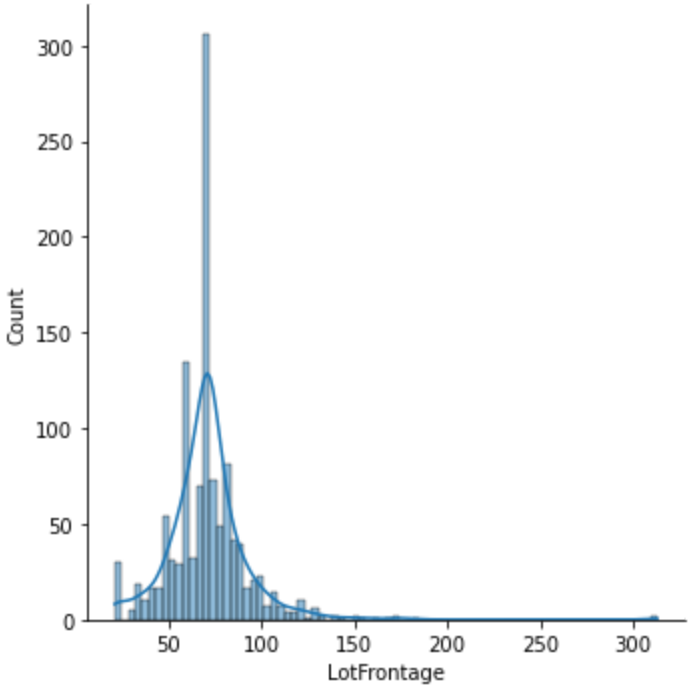
```
<Figure size 1440x576 with 0 Axes>
```



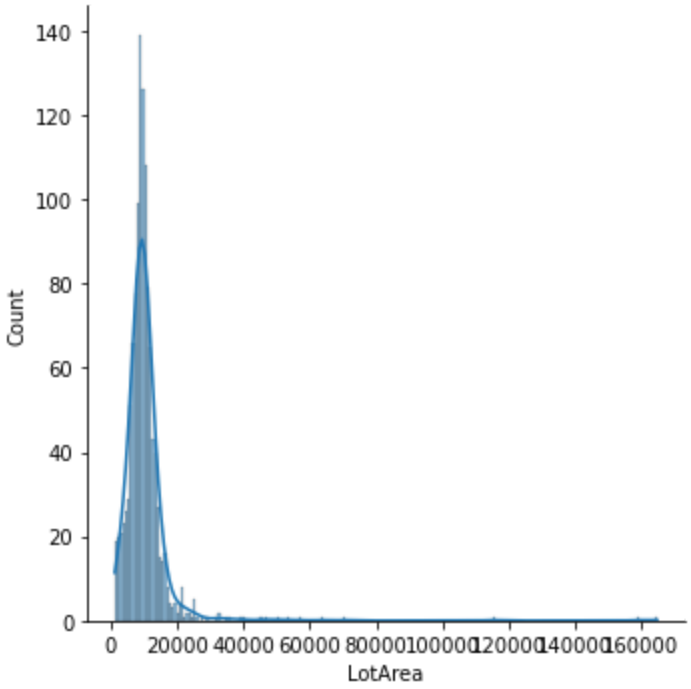
<Figure size 1440x576 with 0 Axes>



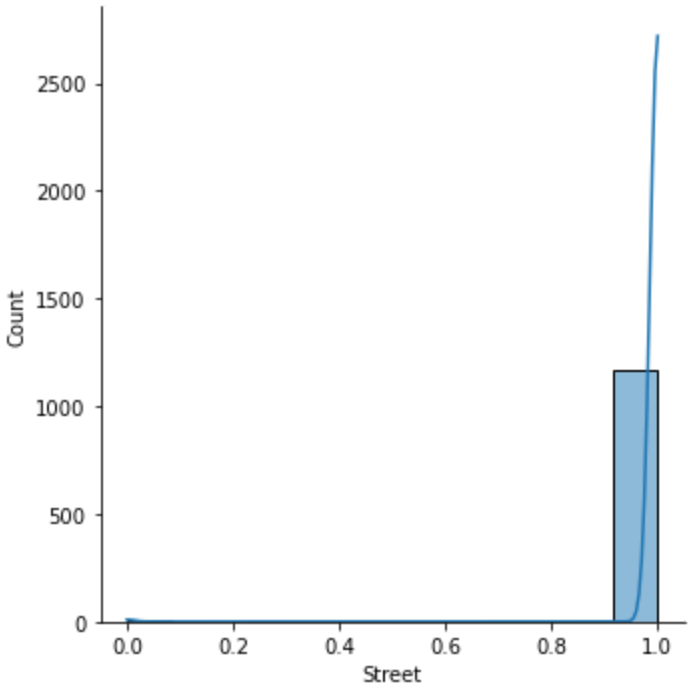
<Figure size 1440x576 with 0 Axes>



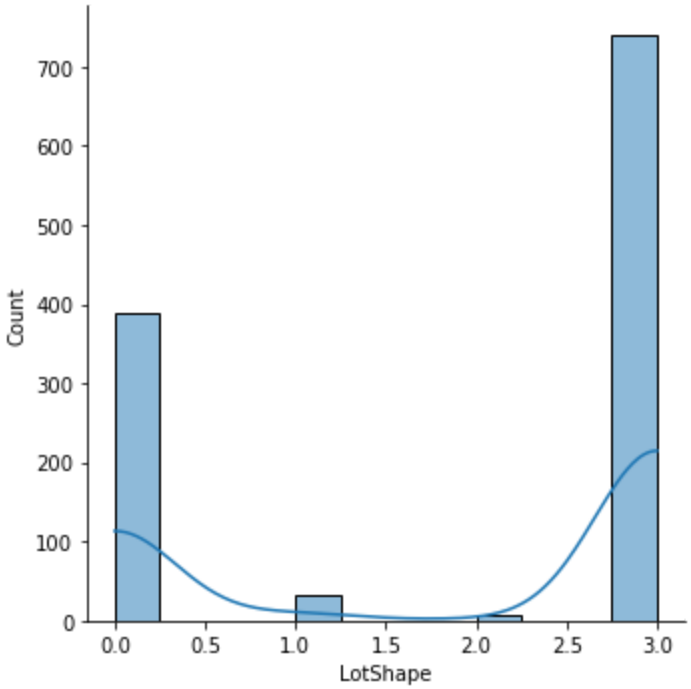
<Figure size 1440x576 with 0 Axes>



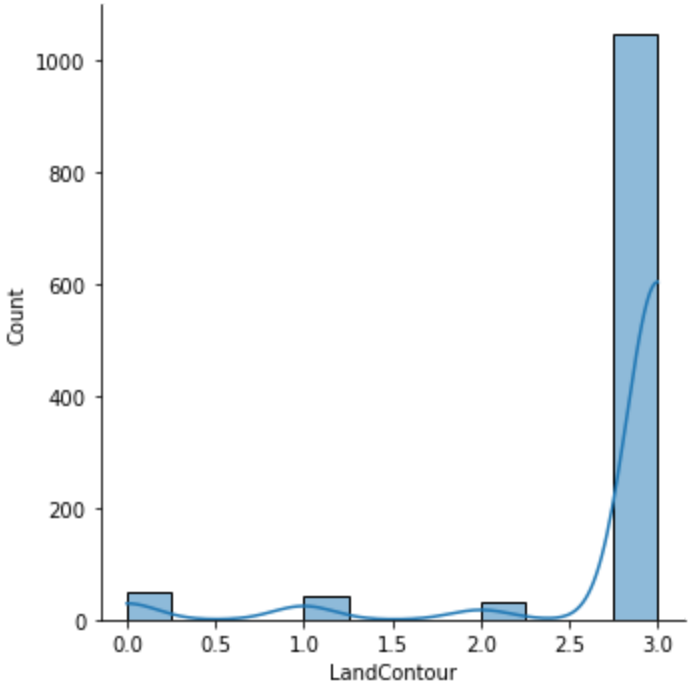
<Figure size 1440x576 with 0 Axes>



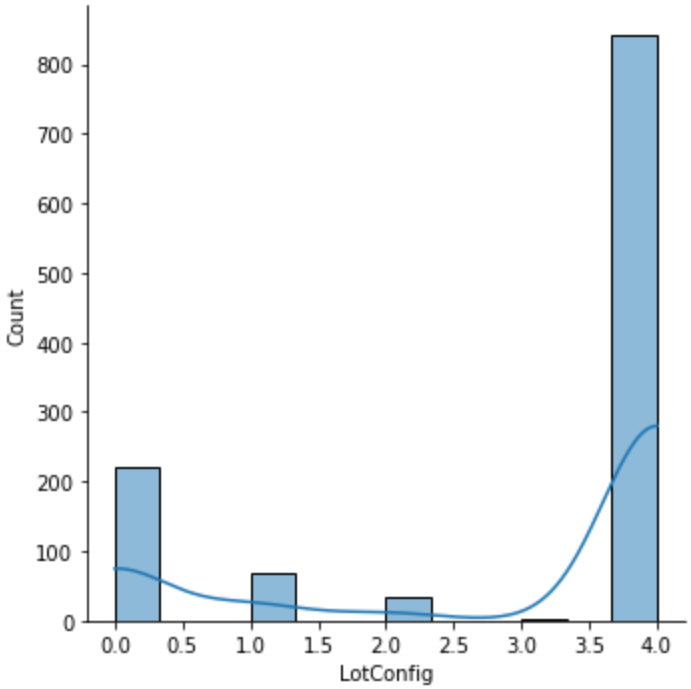
<Figure size 1440x576 with 0 Axes>



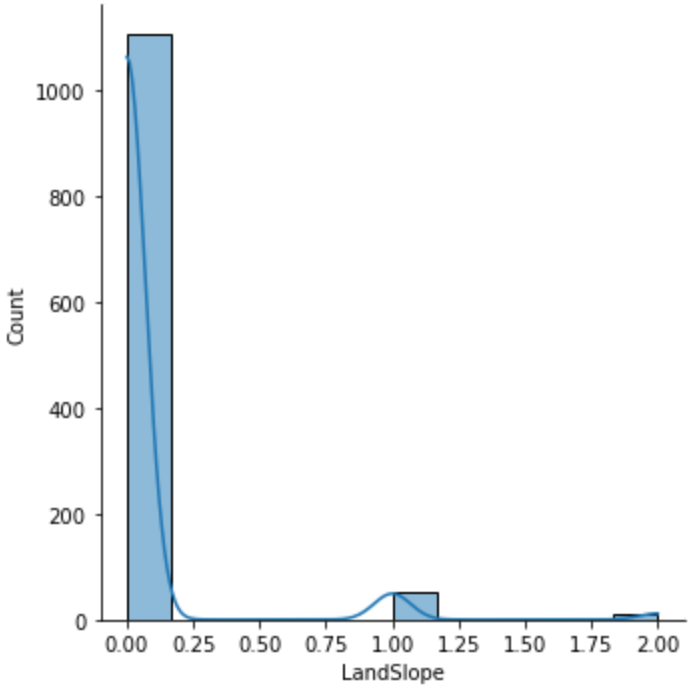
<Figure size 1440x576 with 0 Axes>



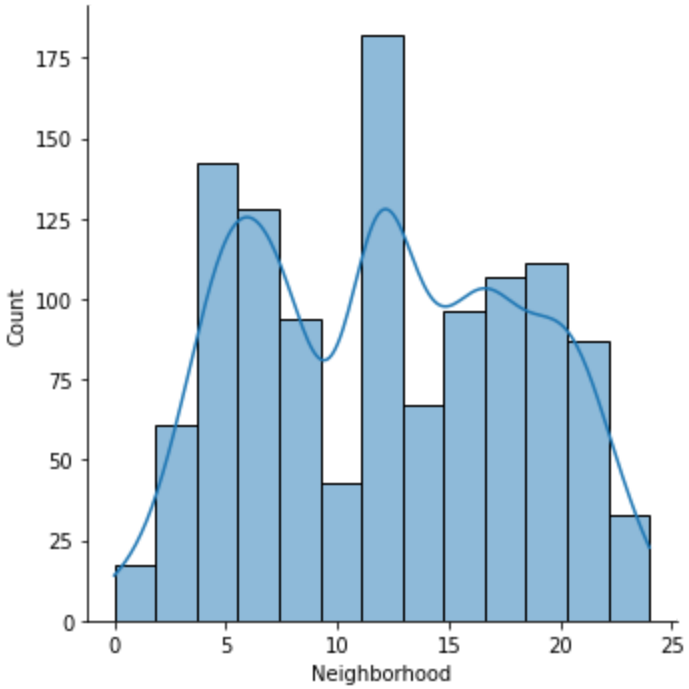

```
<Figure size 1440x576 with 0 Axes>
```



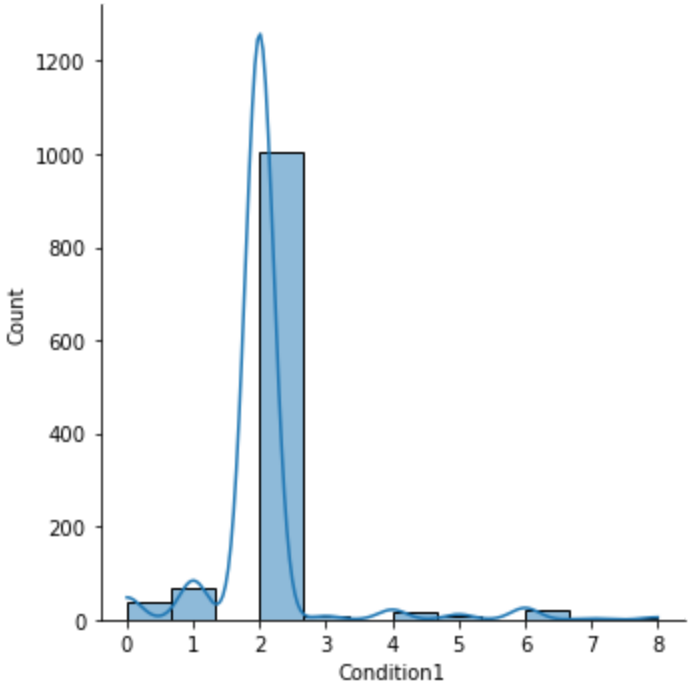

```
<Figure size 1440x576 with 0 Axes>
```



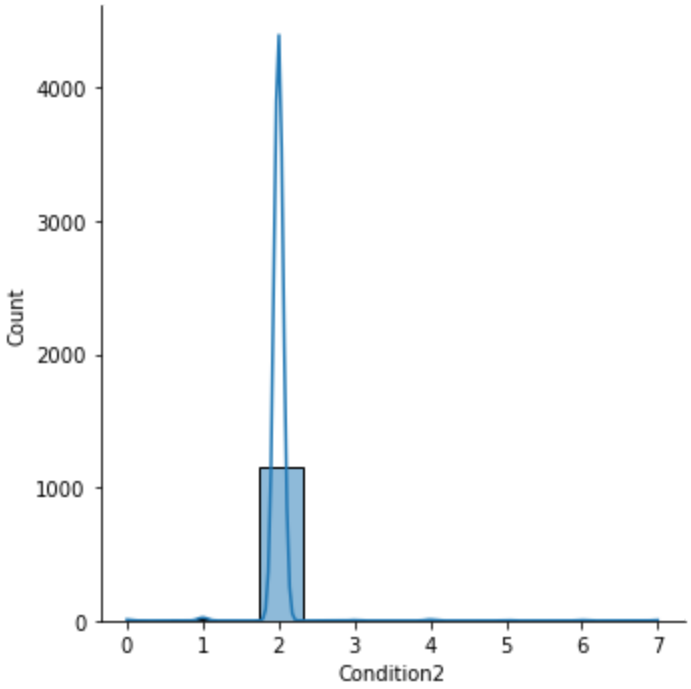
<Figure size 1440x576 with 0 Axes>



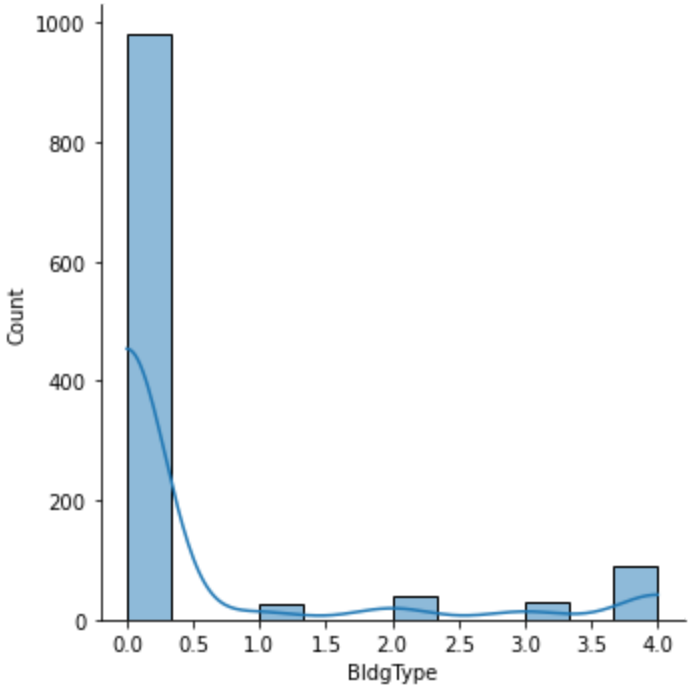
<Figure size 1440x576 with 0 Axes>



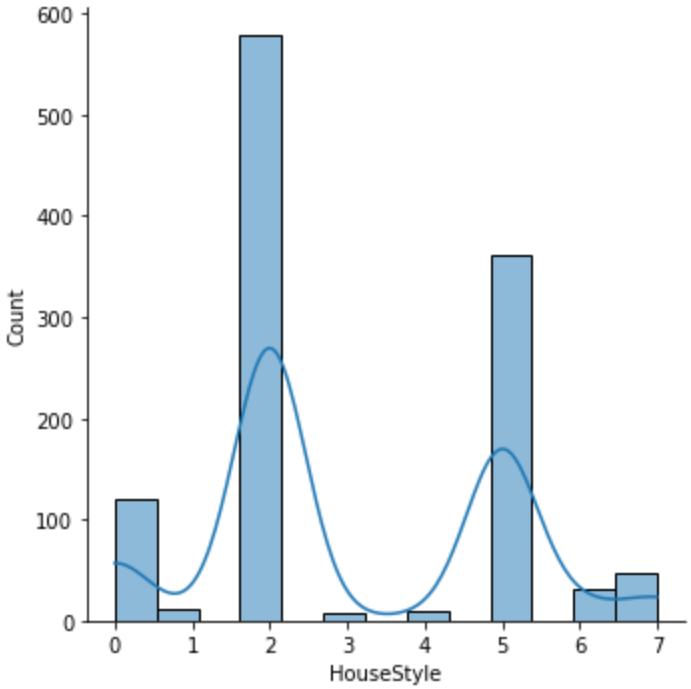
<Figure size 1440x576 with 0 Axes>



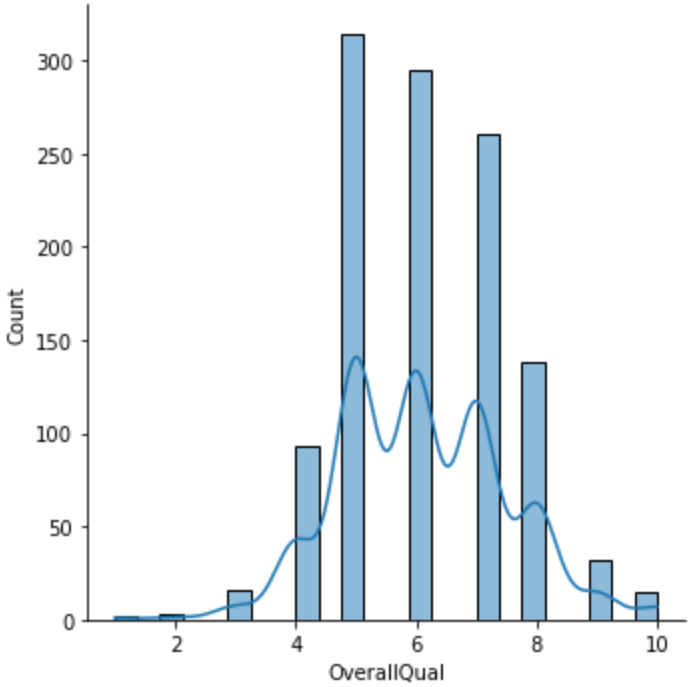
<Figure size 1440x576 with 0 Axes>



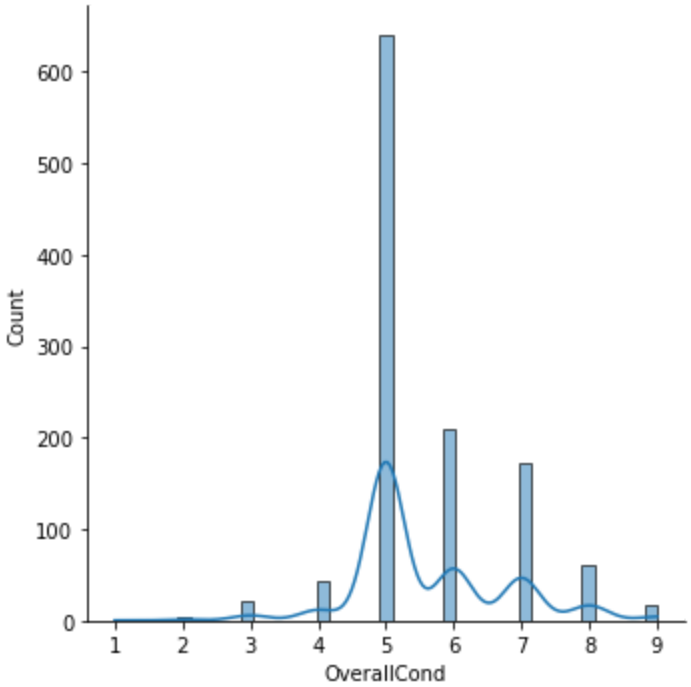

```
<Figure size 1440x576 with 0 Axes>
```



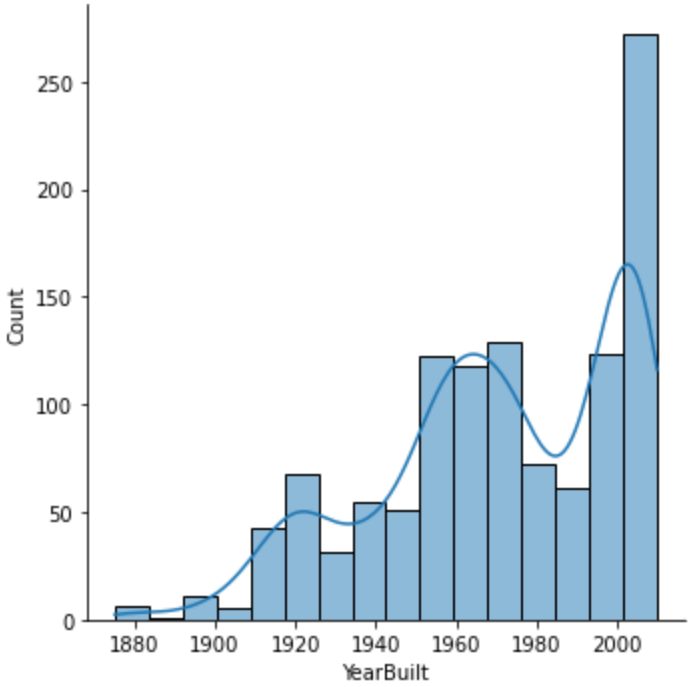

```
<Figure size 1440x576 with 0 Axes>
```



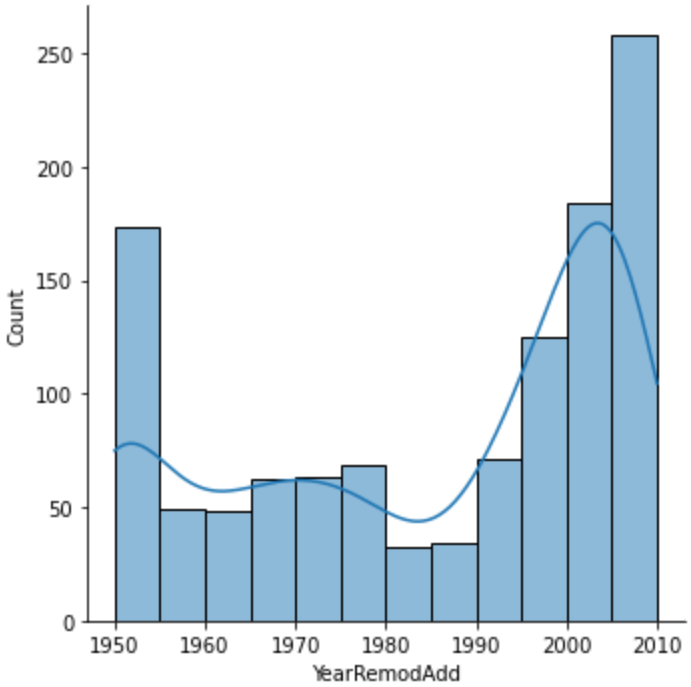
<Figure size 1440x576 with 0 Axes>



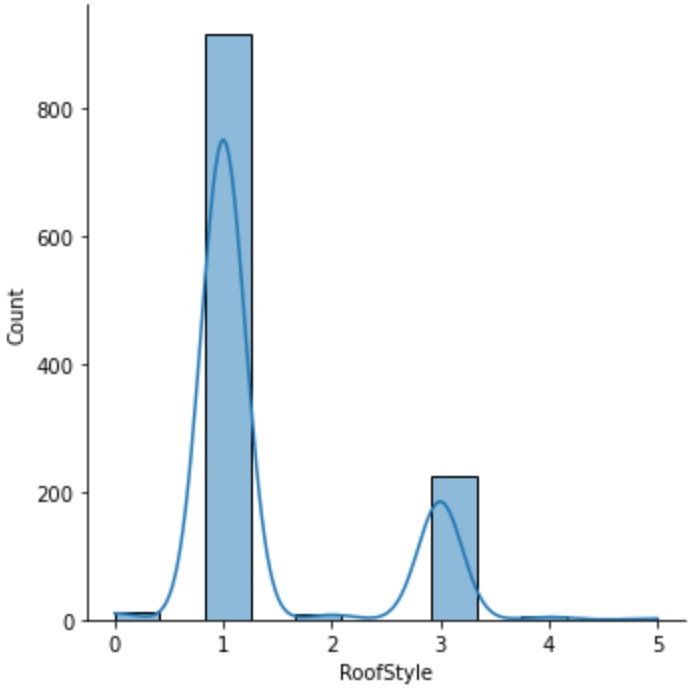
<Figure size 1440x576 with 0 Axes>



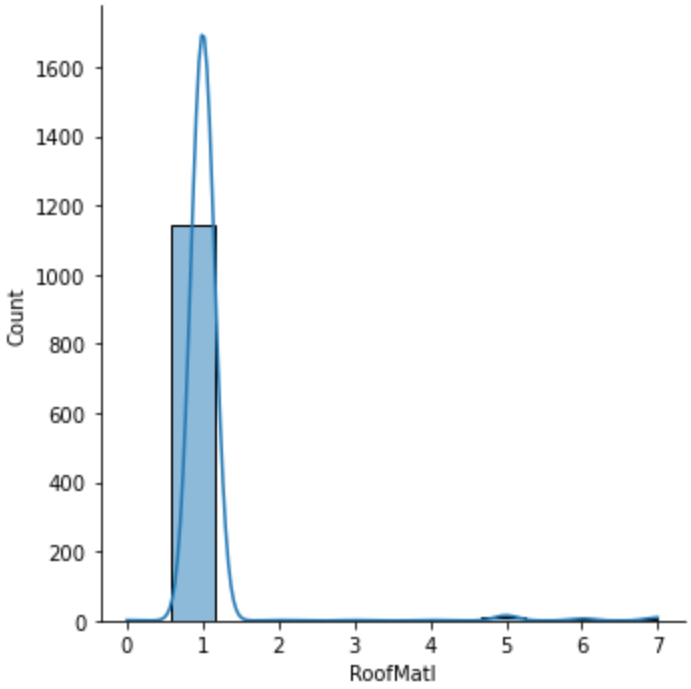
<Figure size 1440x576 with 0 Axes>



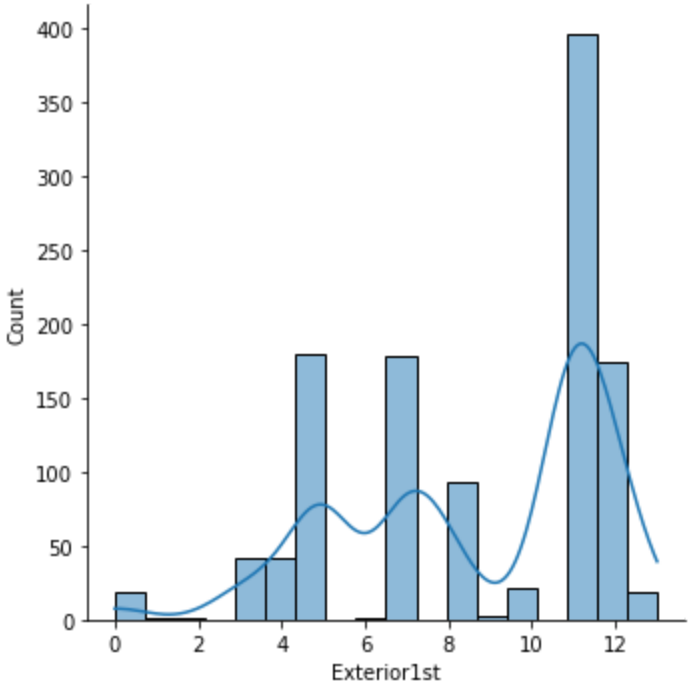
<Figure size 1440x576 with 0 Axes>



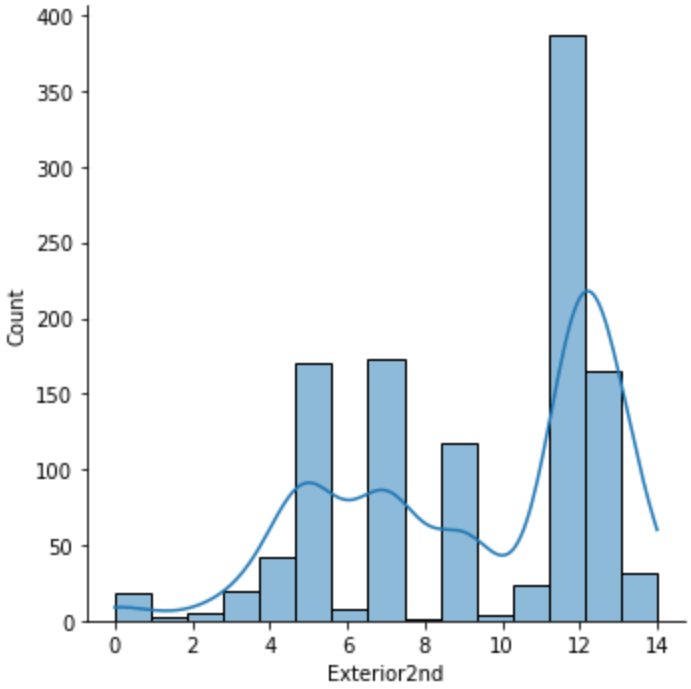
<Figure size 1440x576 with 0 Axes>



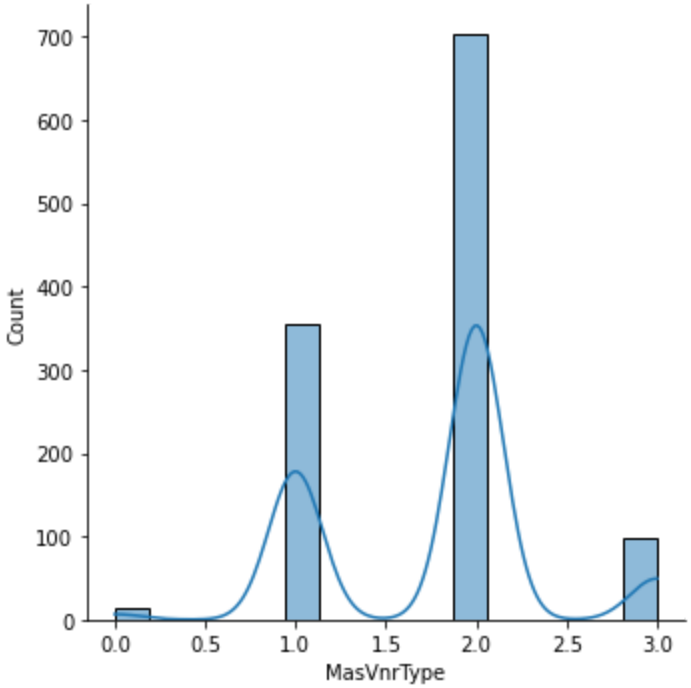
<Figure size 1440x576 with 0 Axes>



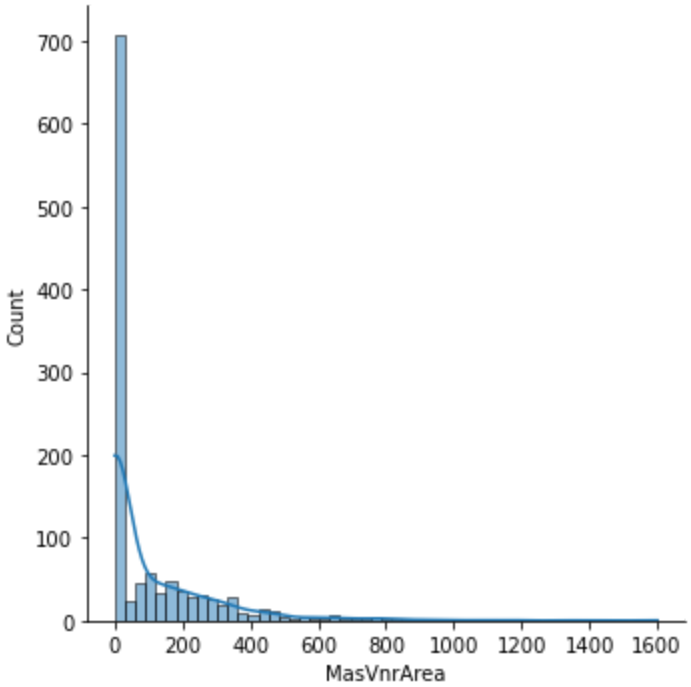

```
<Figure size 1440x576 with 0 Axes>
```



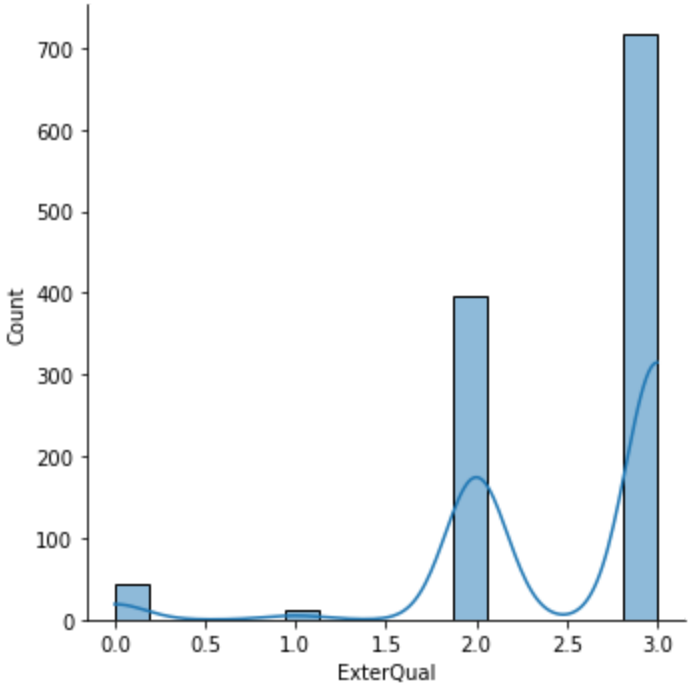

```
<Figure size 1440x576 with 0 Axes>
```



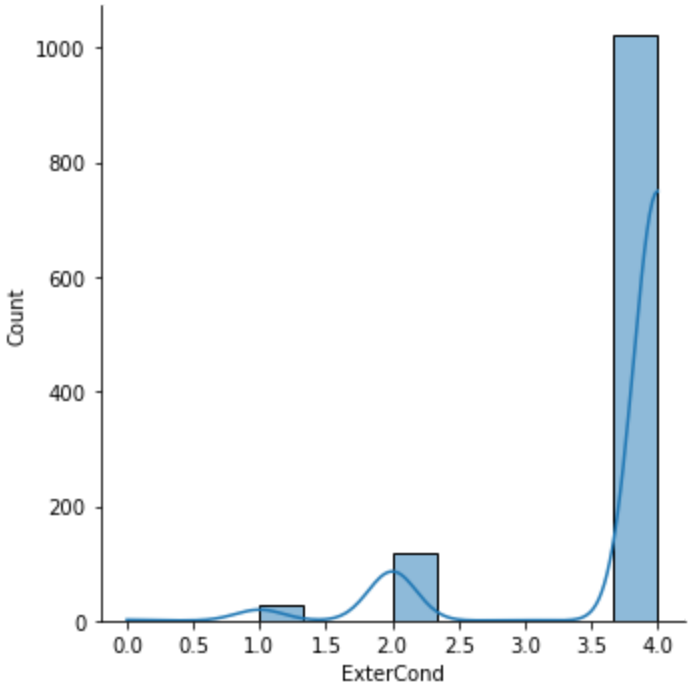

```
<Figure size 1440x576 with 0 Axes>
```



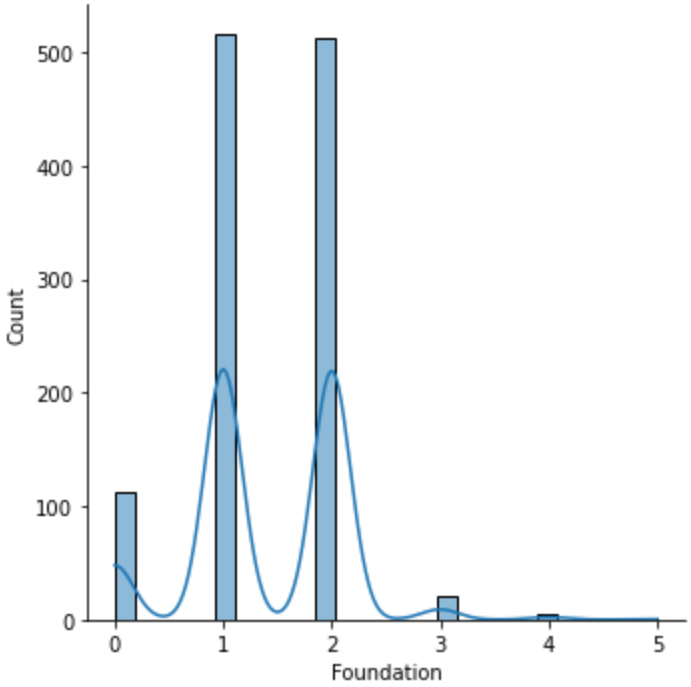

```
<Figure size 1440x576 with 0 Axes>
```



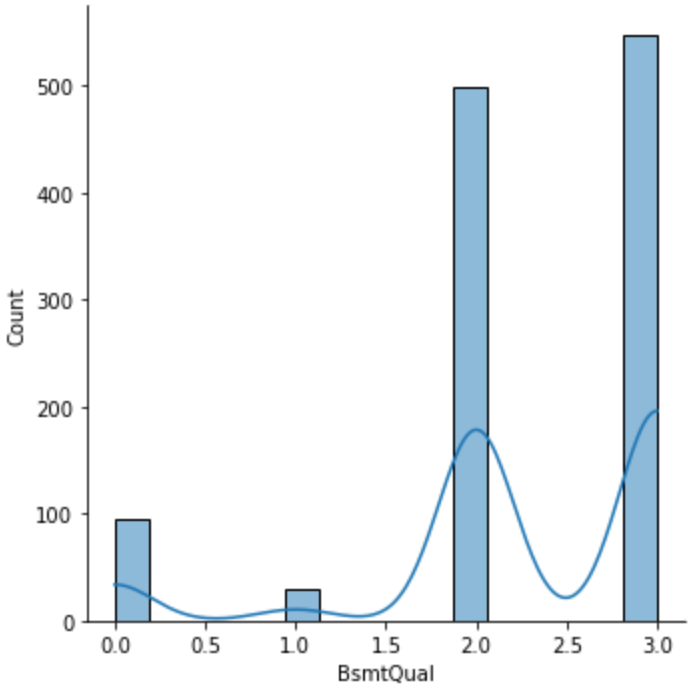
<Figure size 1440x576 with 0 Axes>



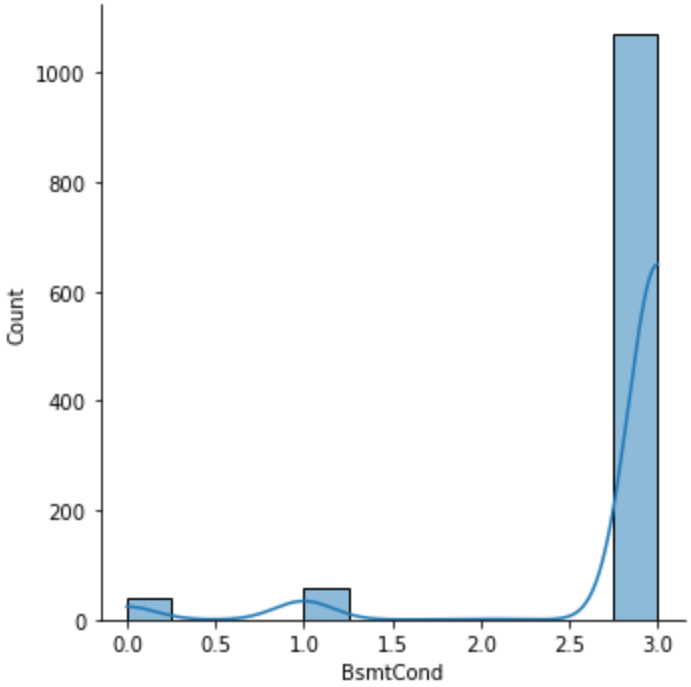
<Figure size 1440x576 with 0 Axes>



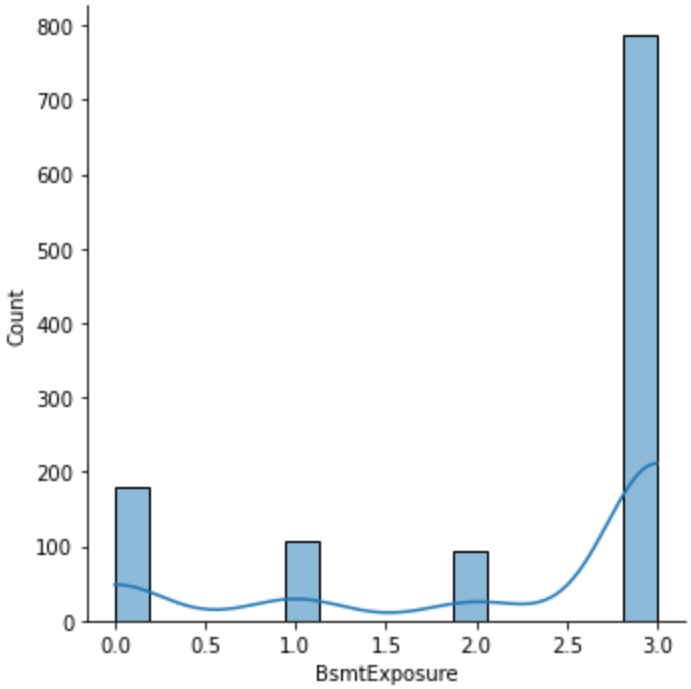
<Figure size 1440x576 with 0 Axes>



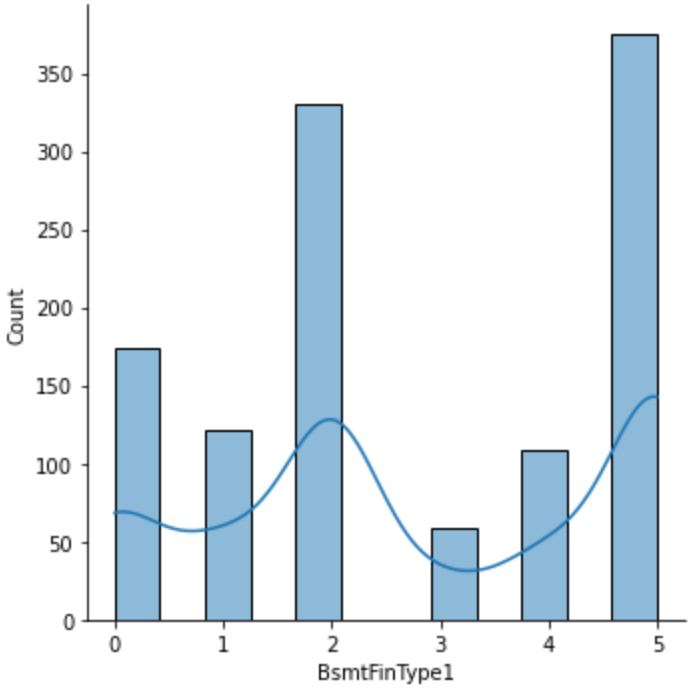
<Figure size 1440x576 with 0 Axes>



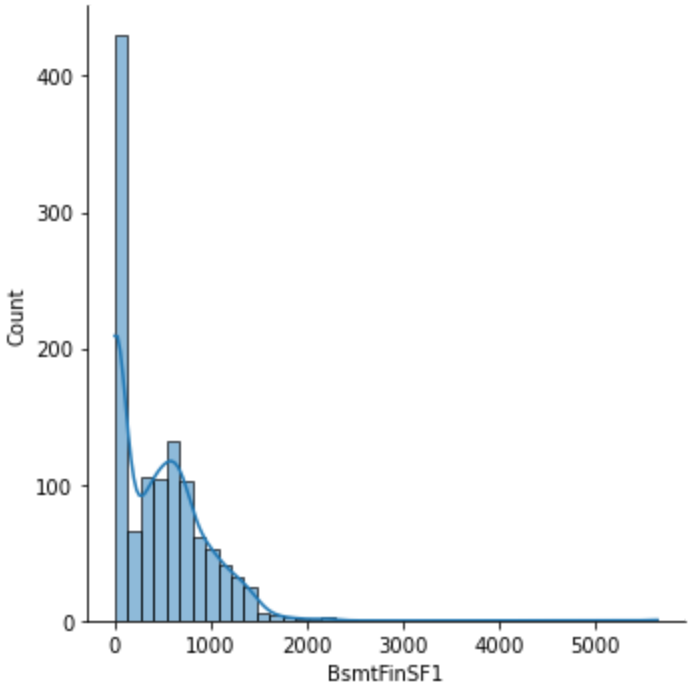

```
<Figure size 1440x576 with 0 Axes>
```



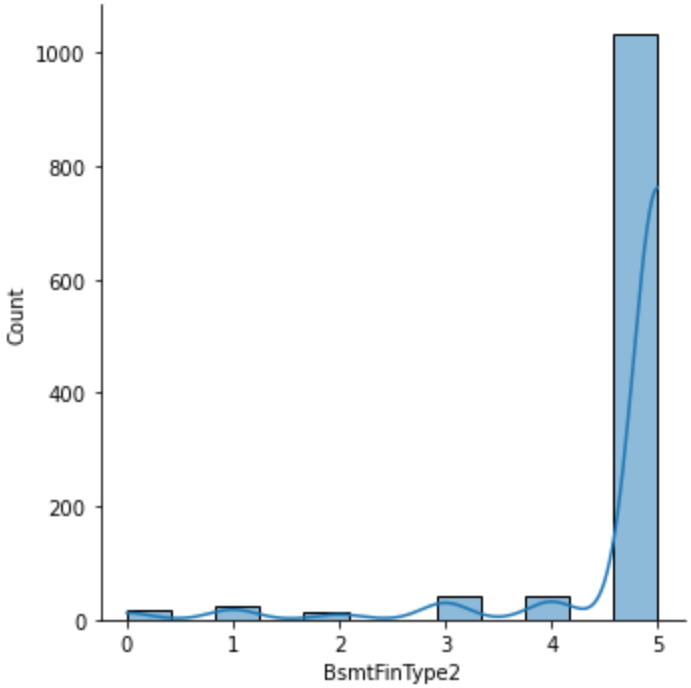

```
<Figure size 1440x576 with 0 Axes>
```



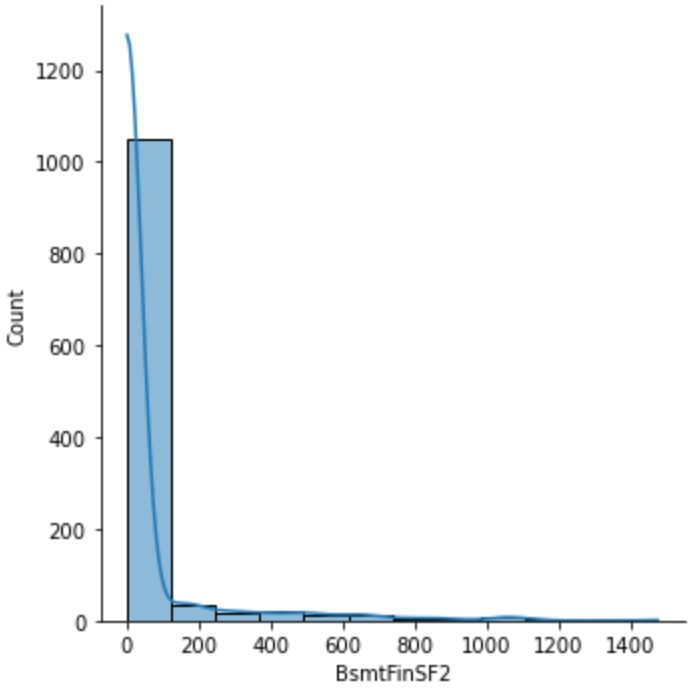
<Figure size 1440x576 with 0 Axes>



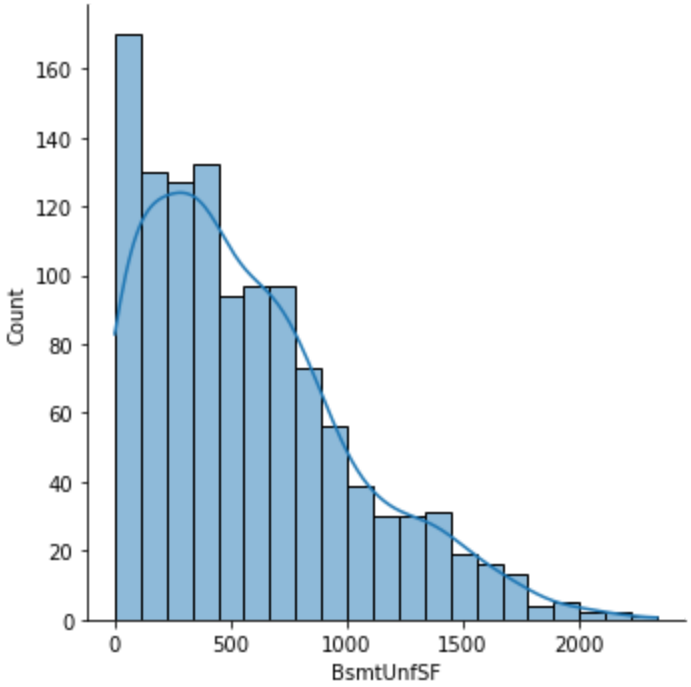
<Figure size 1440x576 with 0 Axes>



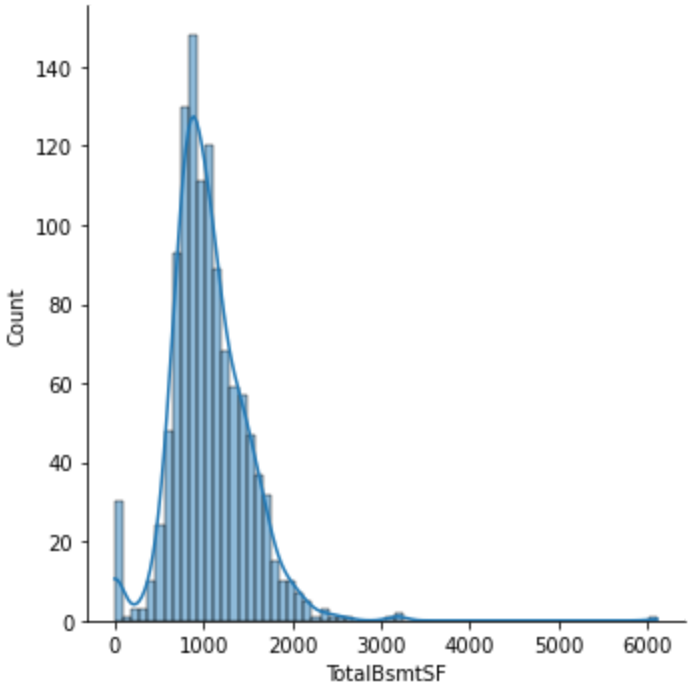
<Figure size 1440x576 with 0 Axes>



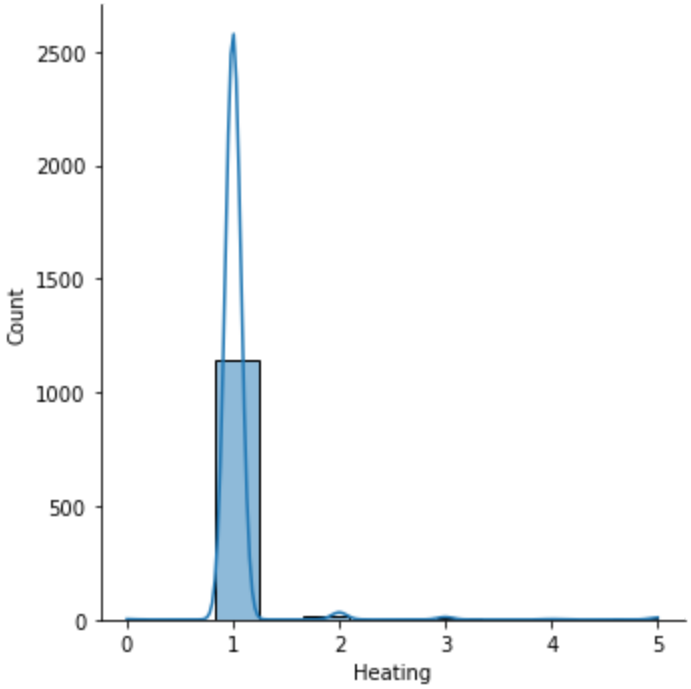
<Figure size 1440x576 with 0 Axes>



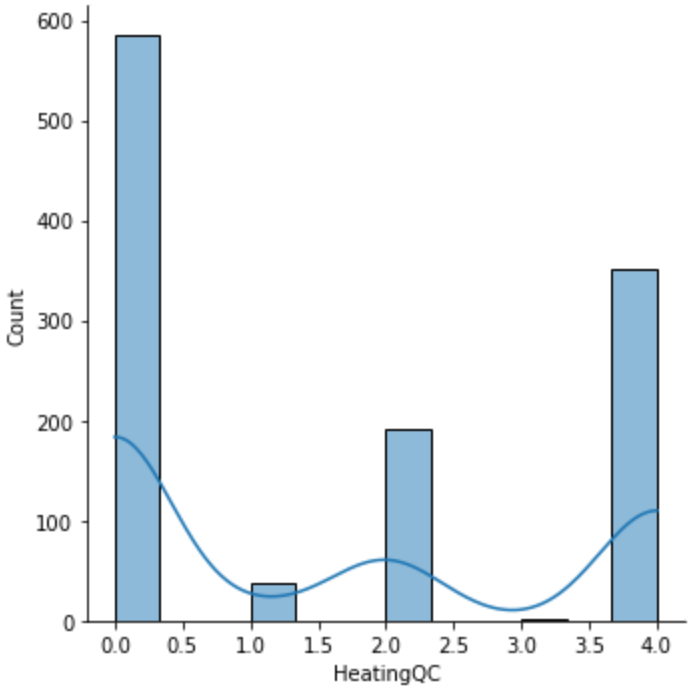
<Figure size 1440x576 with 0 Axes>



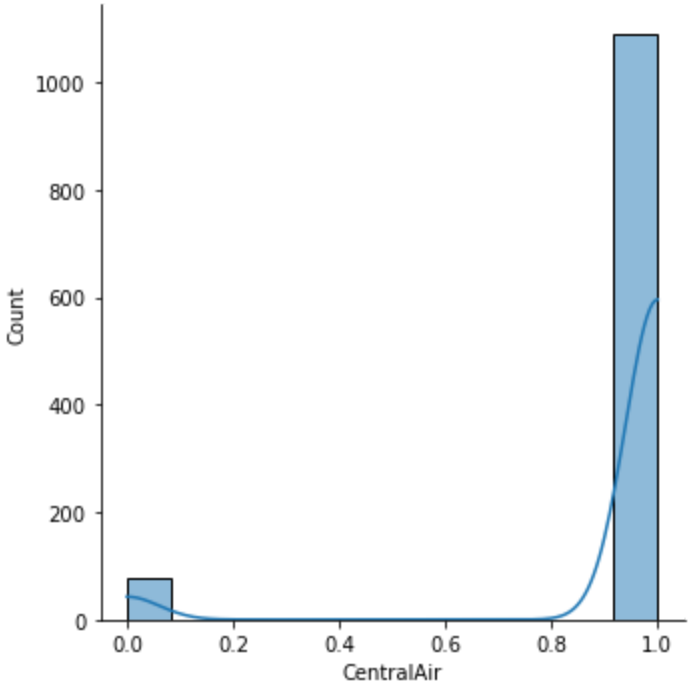
<Figure size 1440x576 with 0 Axes>



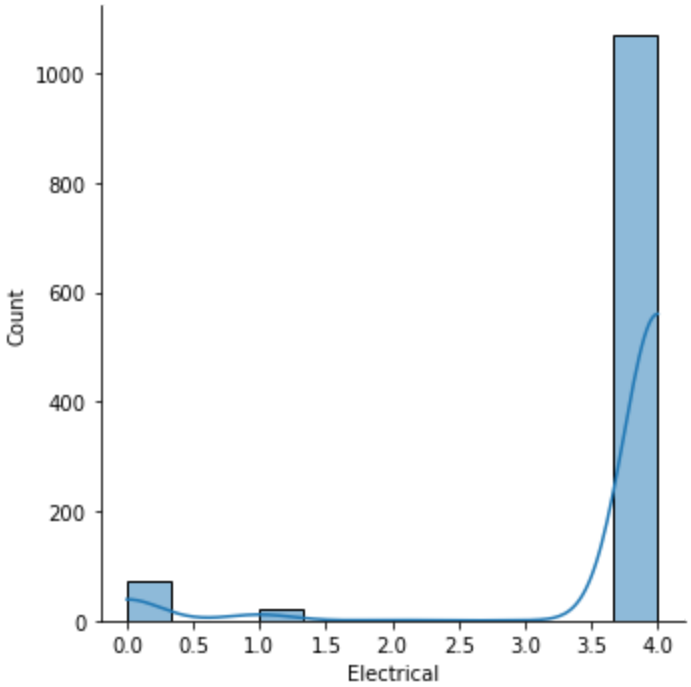
<Figure size 1440x576 with 0 Axes>



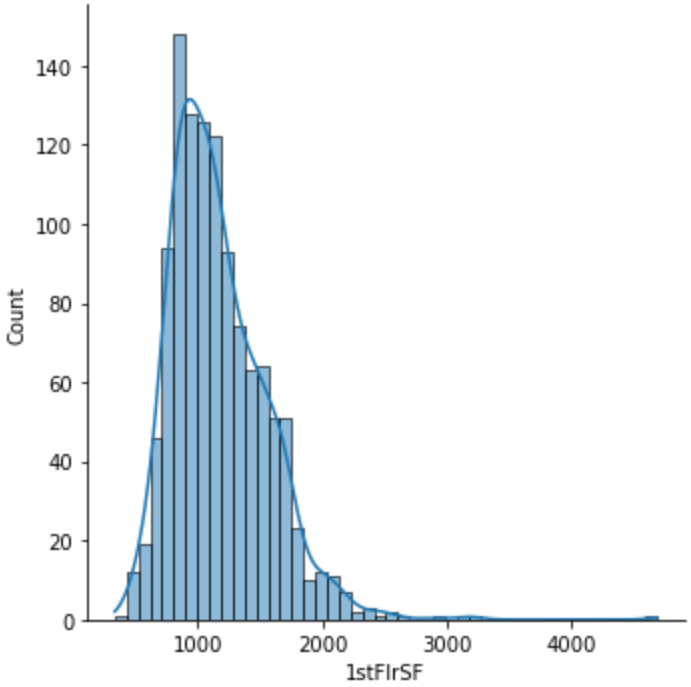
<Figure size 1440x576 with 0 Axes>



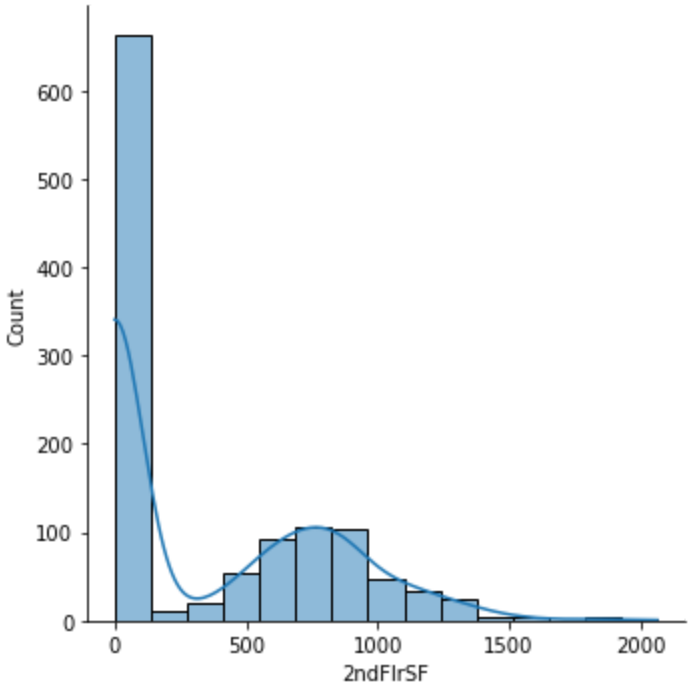
<Figure size 1440x576 with 0 Axes>



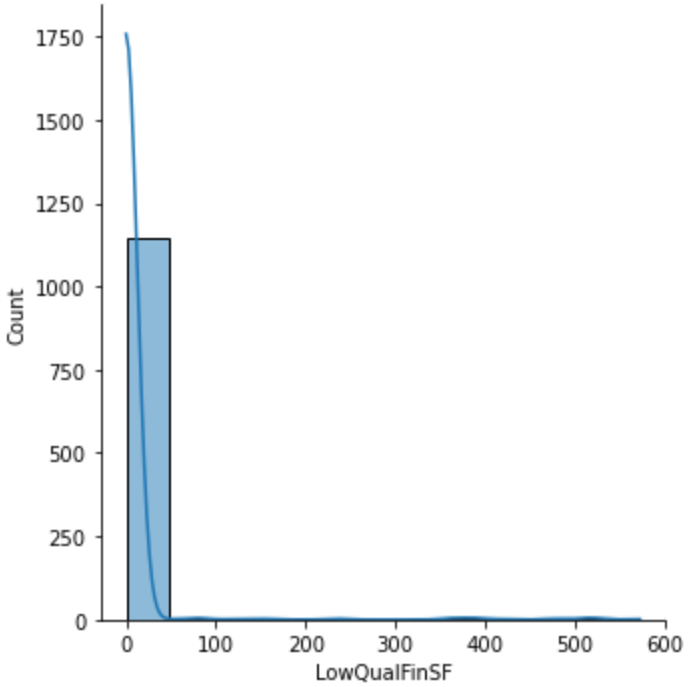
<Figure size 1440x576 with 0 Axes>



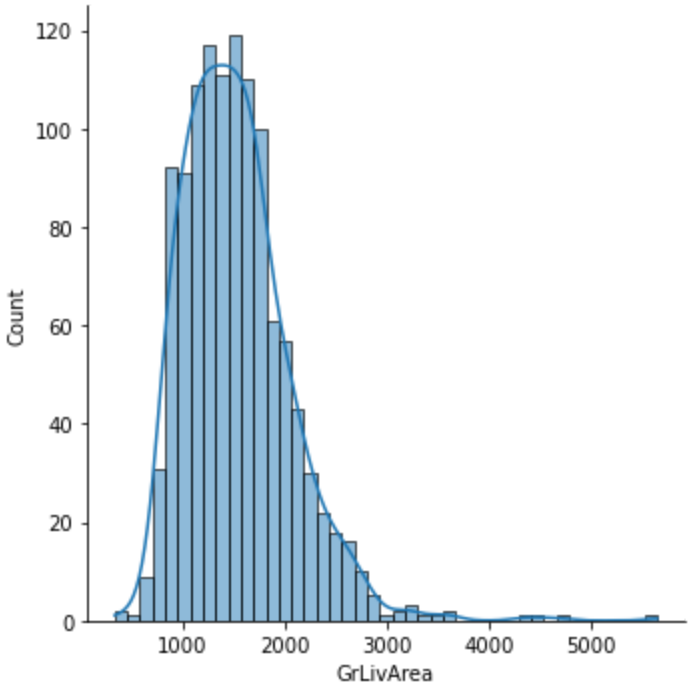
<Figure size 1440x576 with 0 Axes>



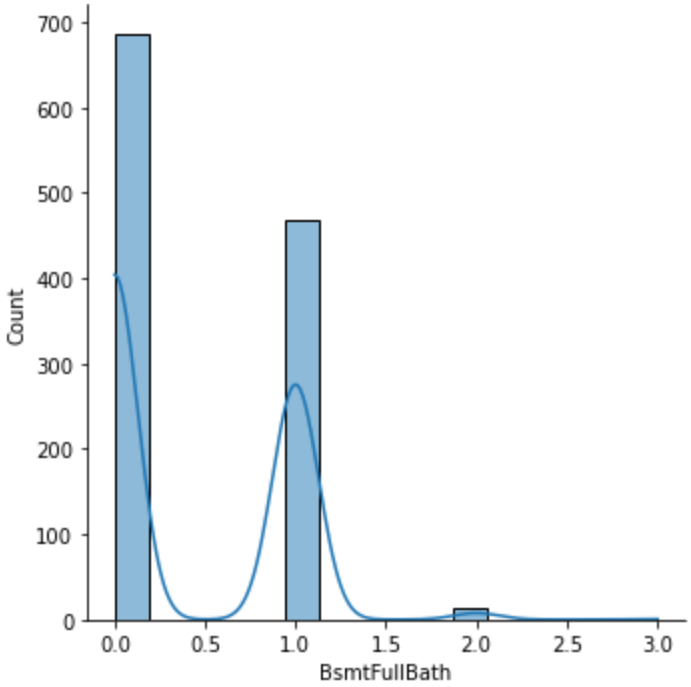
<Figure size 1440x576 with 0 Axes>



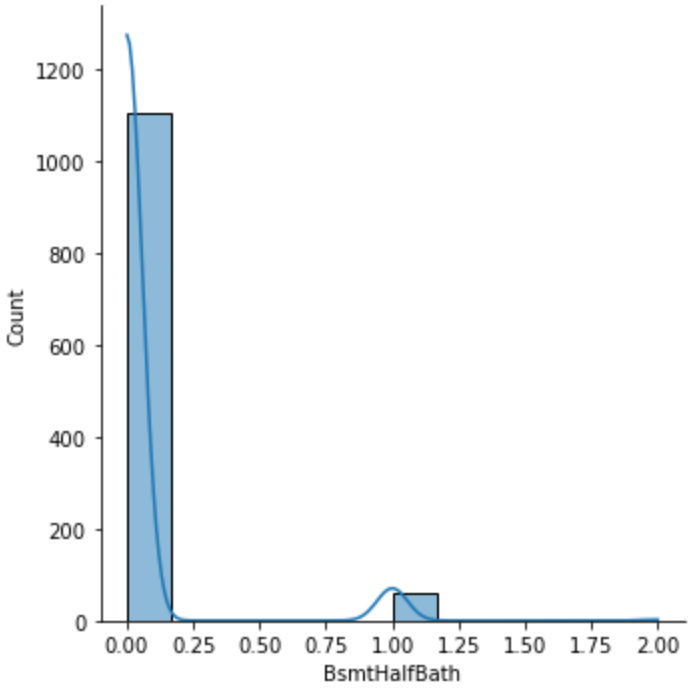
<Figure size 1440x576 with 0 Axes>



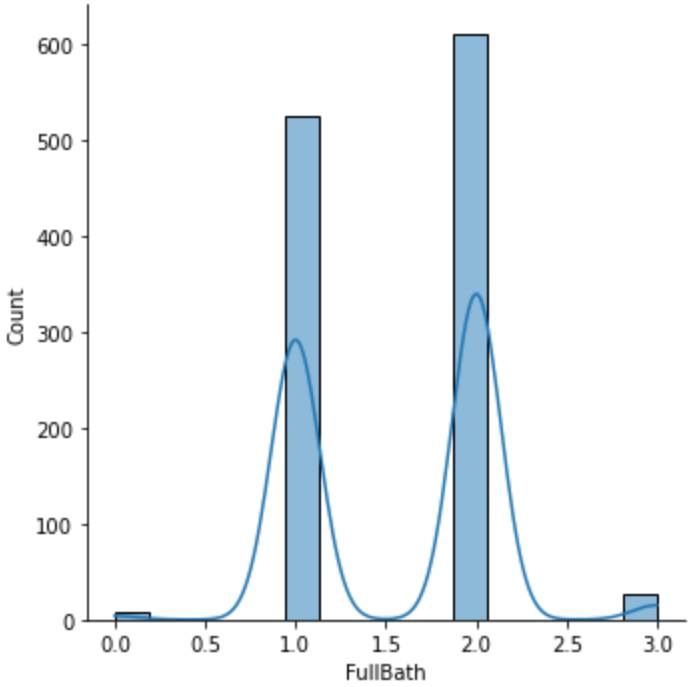
<Figure size 1440x576 with 0 Axes>



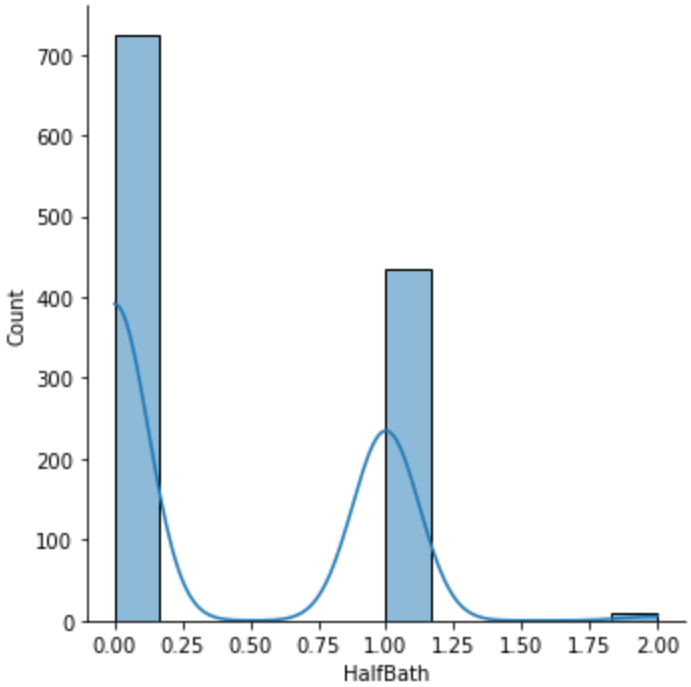
<Figure size 1440x576 with 0 Axes>



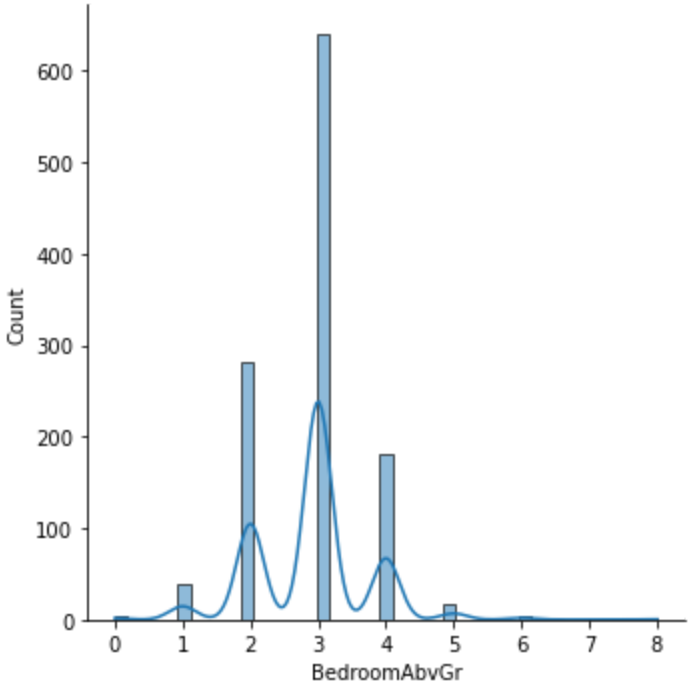
<Figure size 1440x576 with 0 Axes>



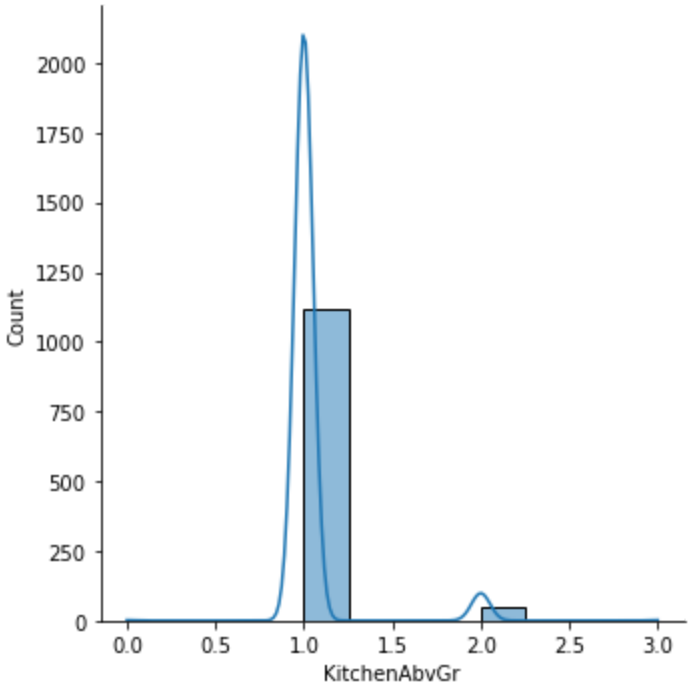

```
<Figure size 1440x576 with 0 Axes>
```



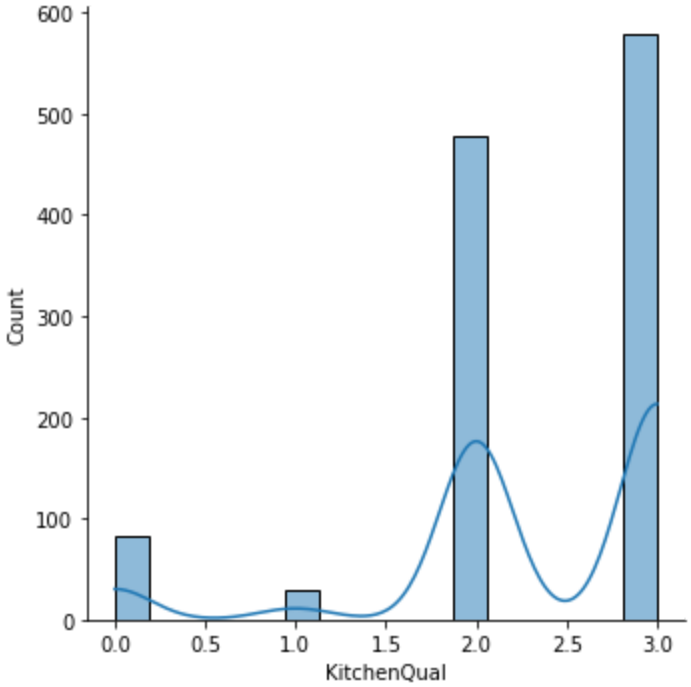

```
<Figure size 1440x576 with 0 Axes>
```



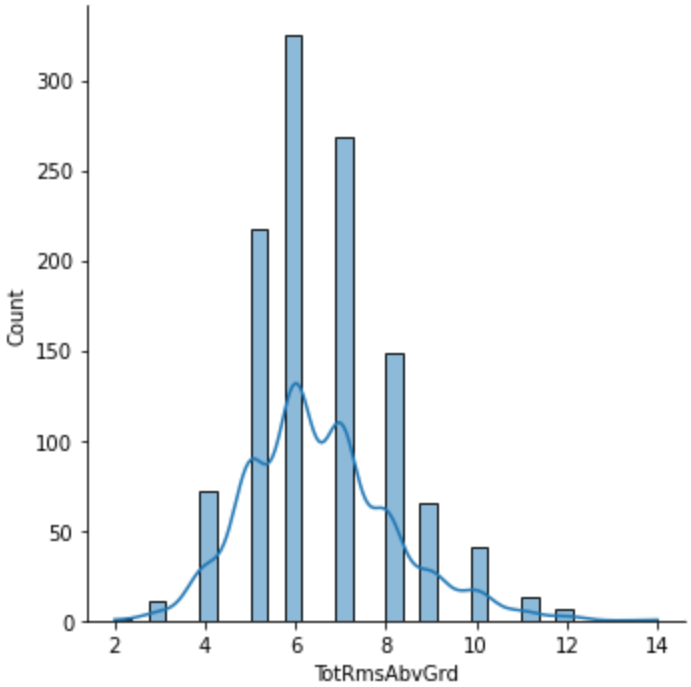
<Figure size 1440x576 with 0 Axes>



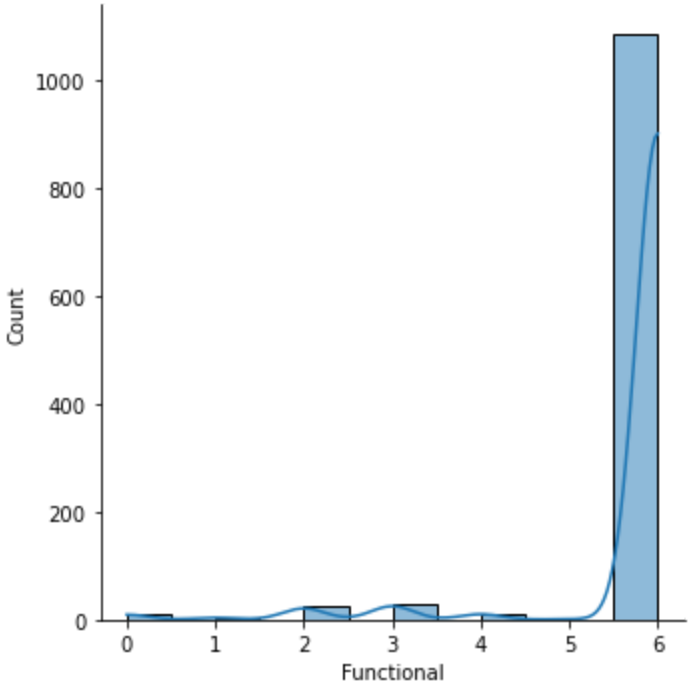
<Figure size 1440x576 with 0 Axes>



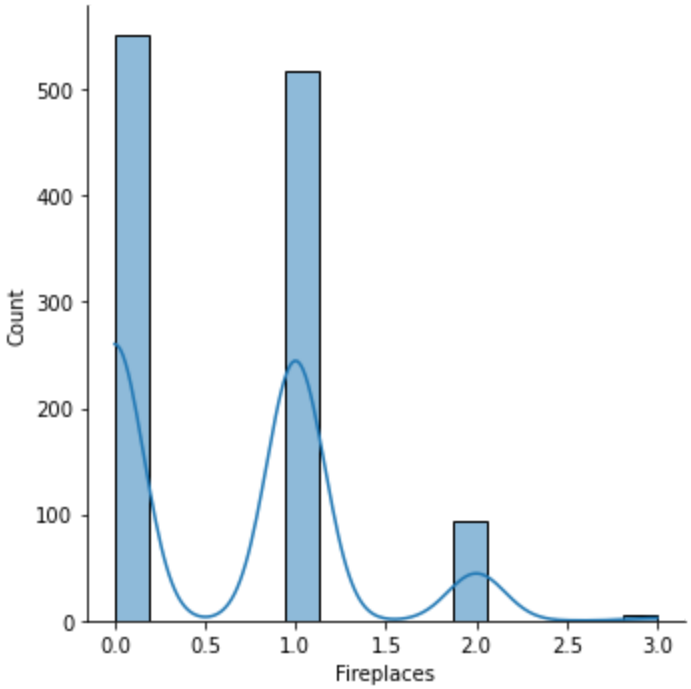

```
<Figure size 1440x576 with 0 Axes>
```



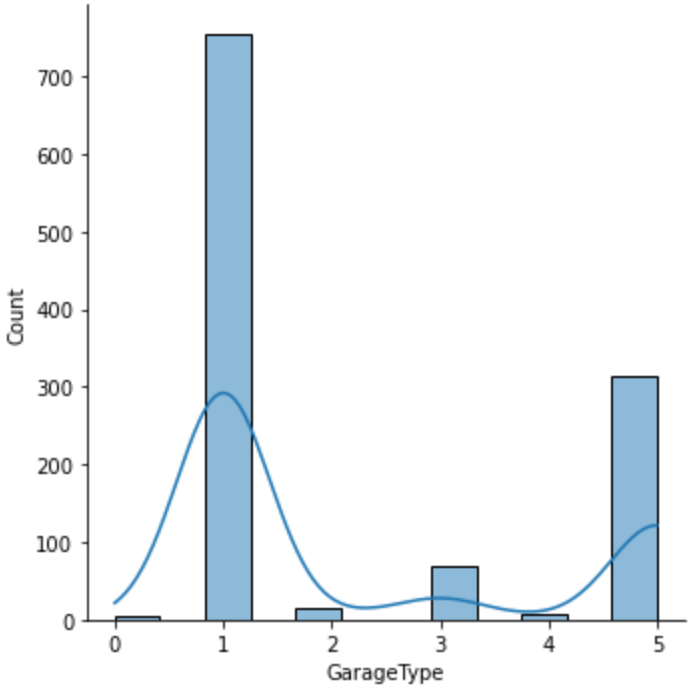

```
<Figure size 1440x576 with 0 Axes>
```



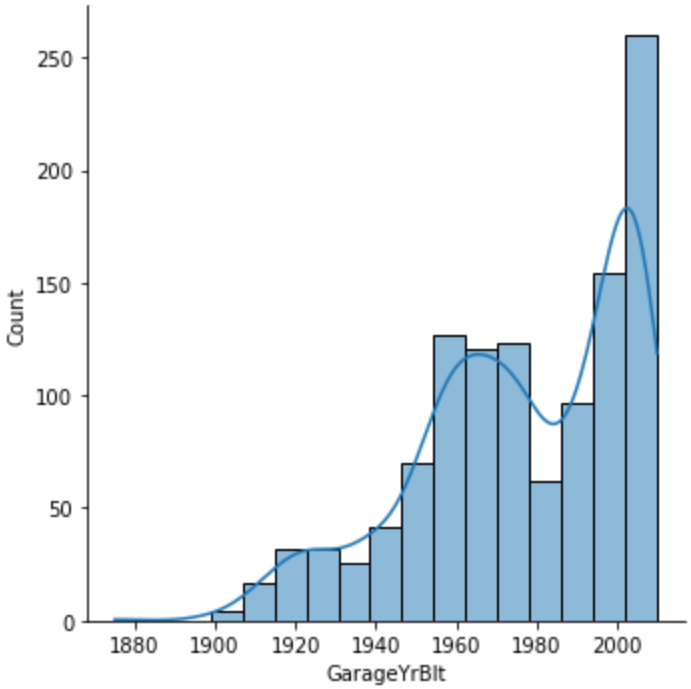
<Figure size 1440x576 with 0 Axes>



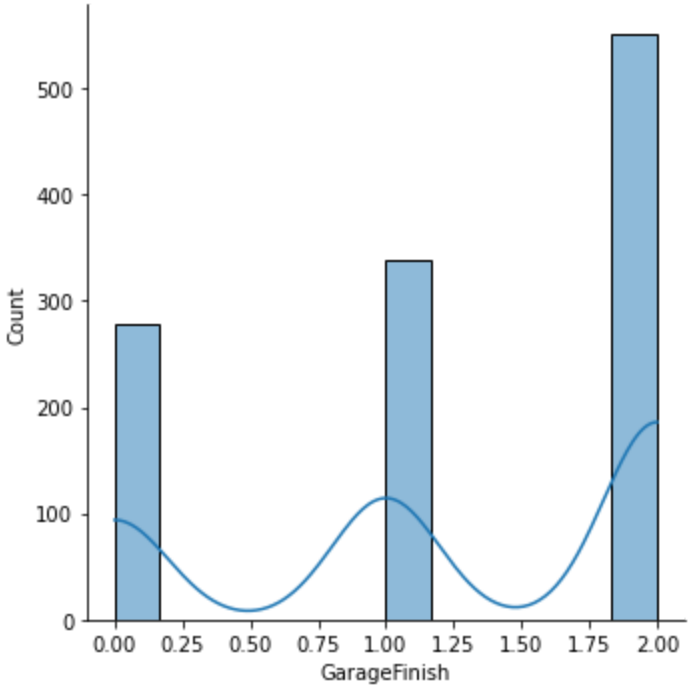
<Figure size 1440x576 with 0 Axes>



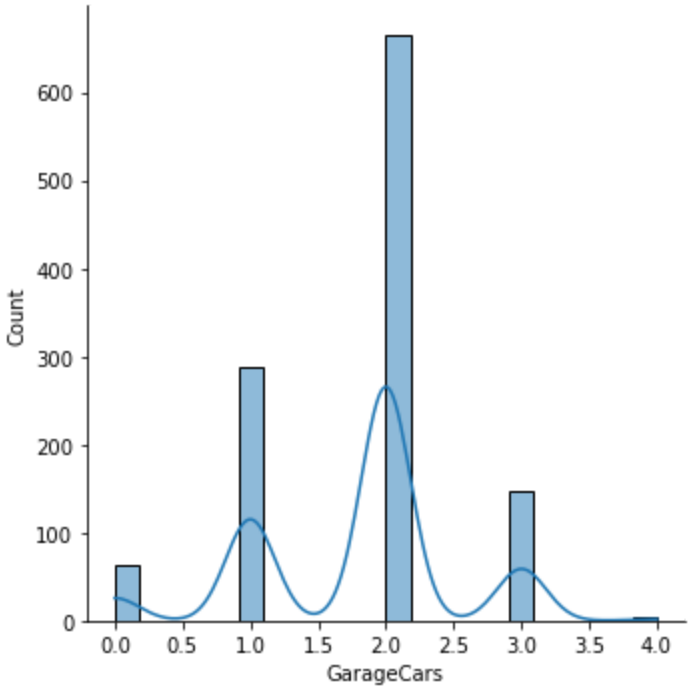
<Figure size 1440x576 with 0 Axes>



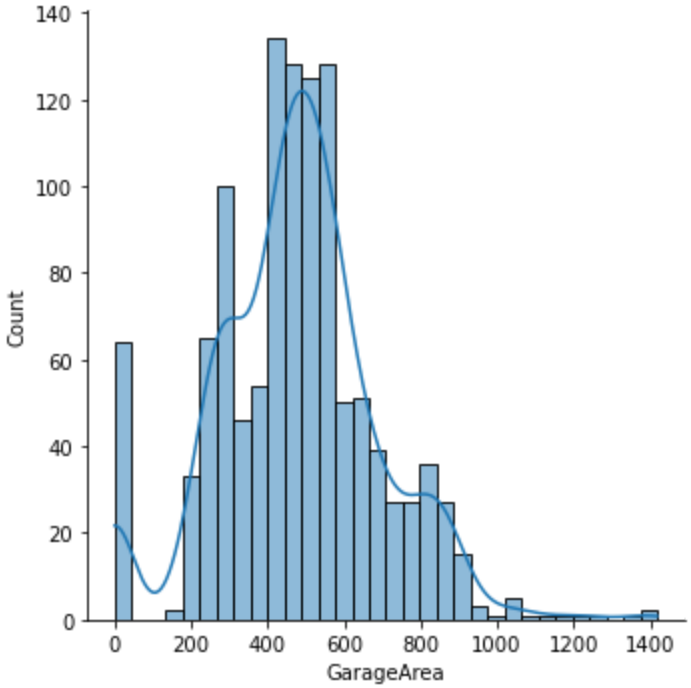
<Figure size 1440x576 with 0 Axes>



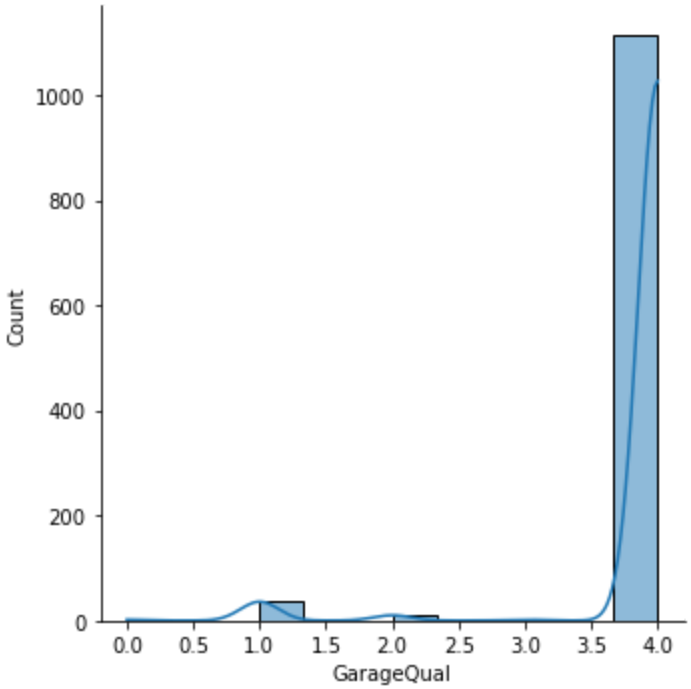

```
<Figure size 1440x576 with 0 Axes>
```



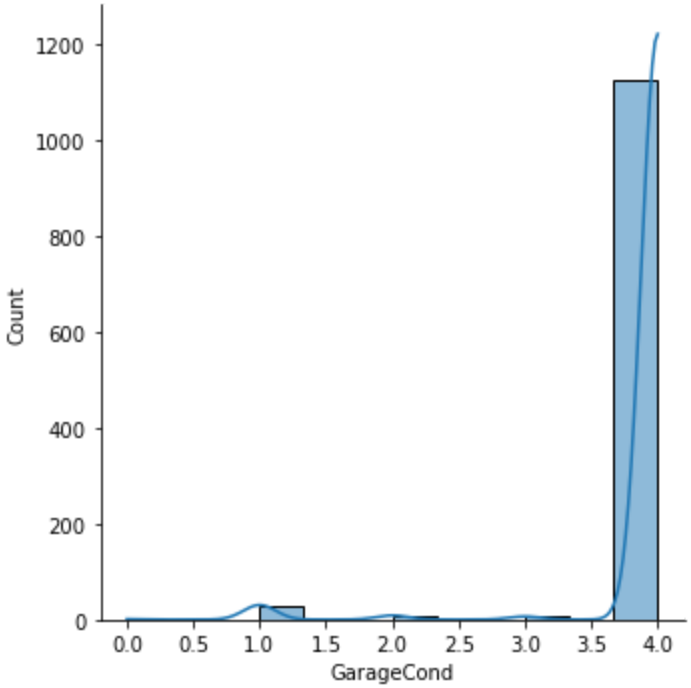

```
<Figure size 1440x576 with 0 Axes>
```



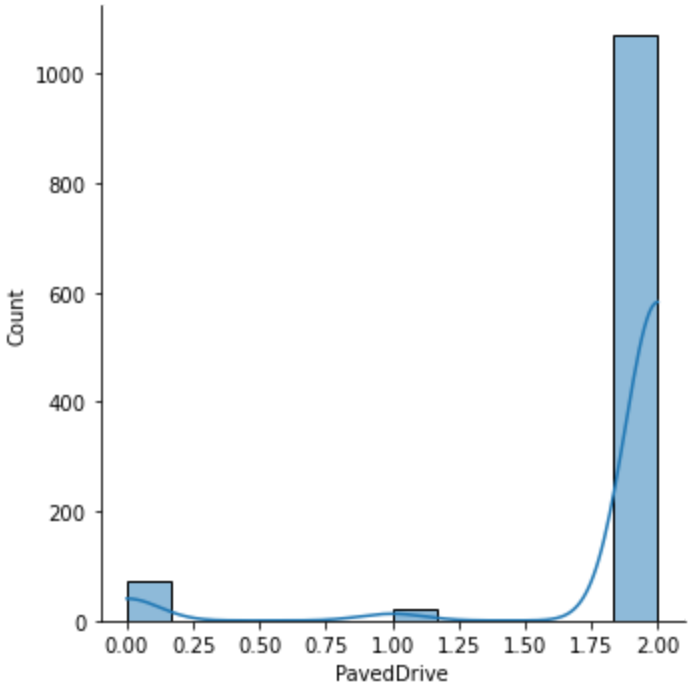
<Figure size 1440x576 with 0 Axes>



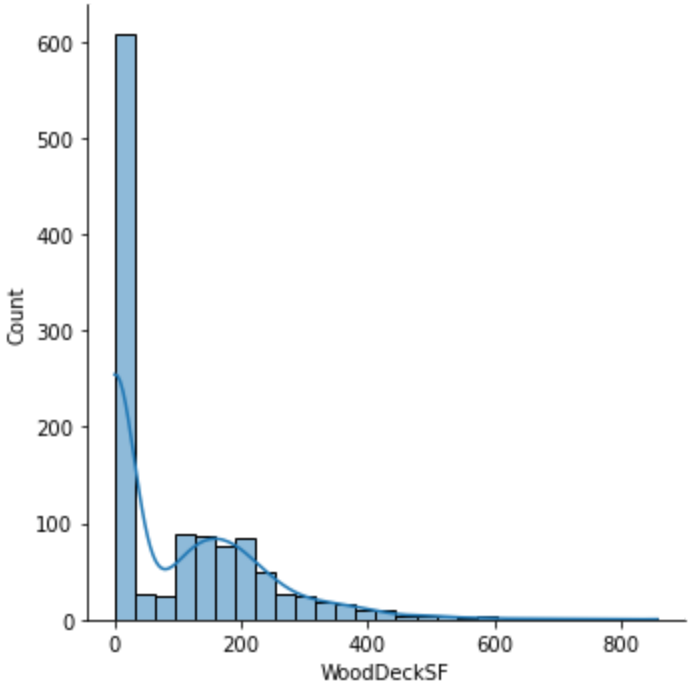
<Figure size 1440x576 with 0 Axes>



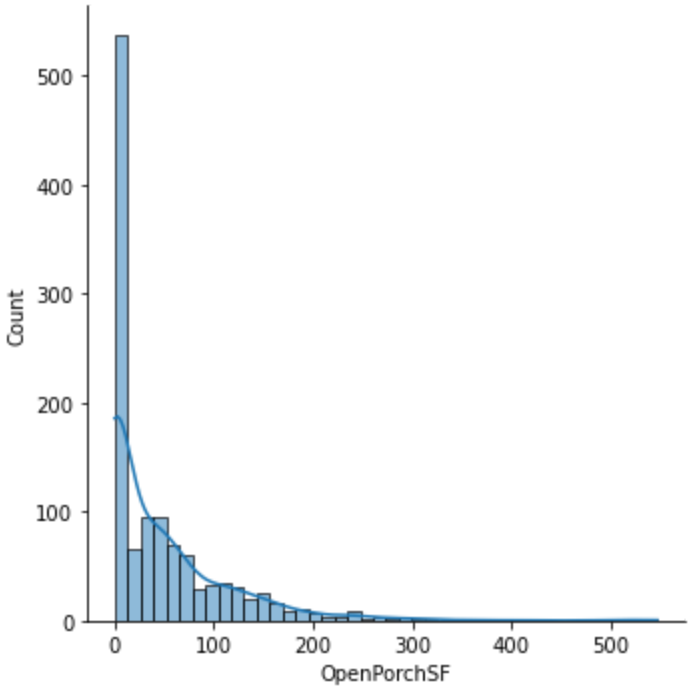
<Figure size 1440x576 with 0 Axes>



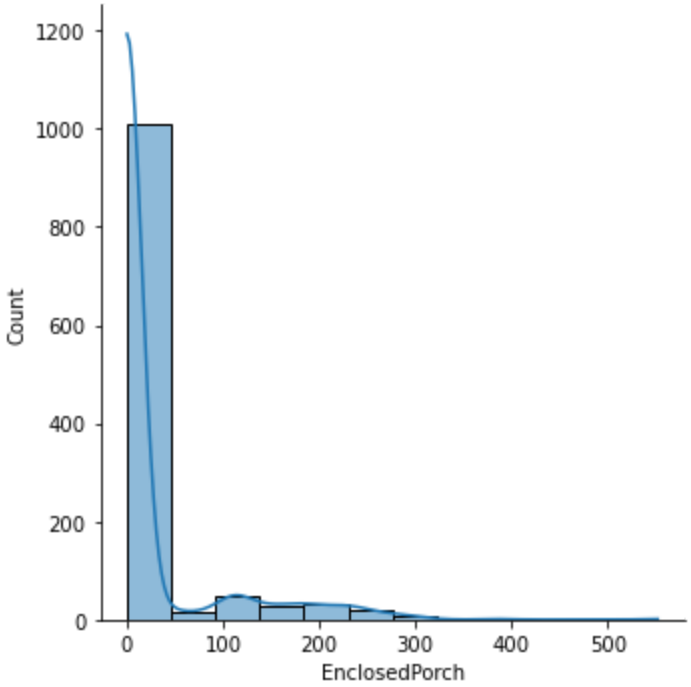
<Figure size 1440x576 with 0 Axes>



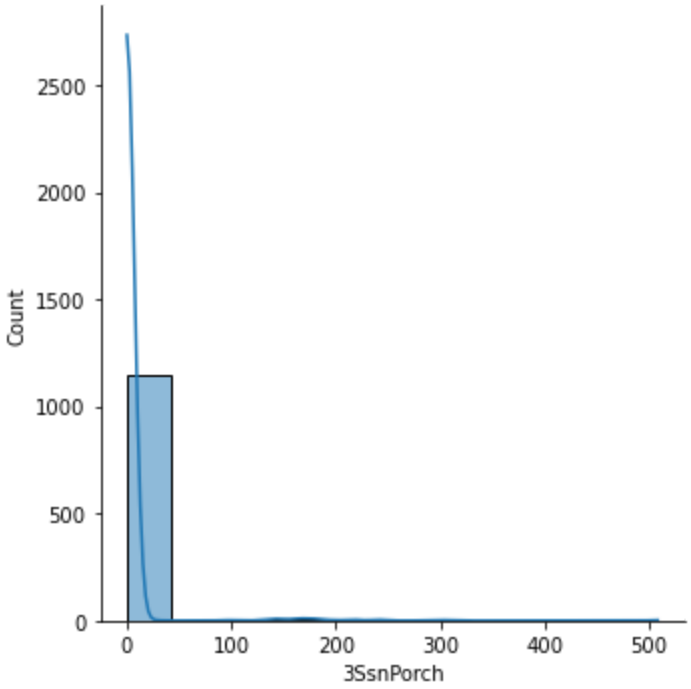
<Figure size 1440x576 with 0 Axes>



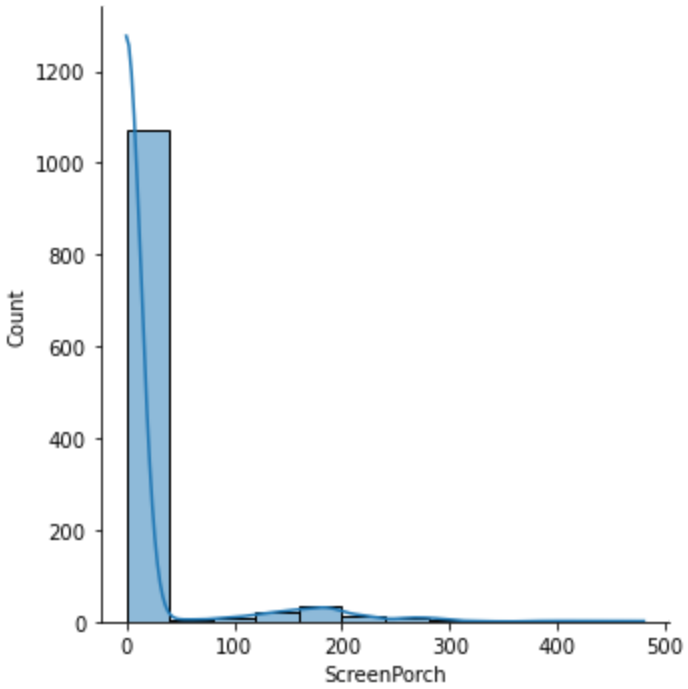
<Figure size 1440x576 with 0 Axes>



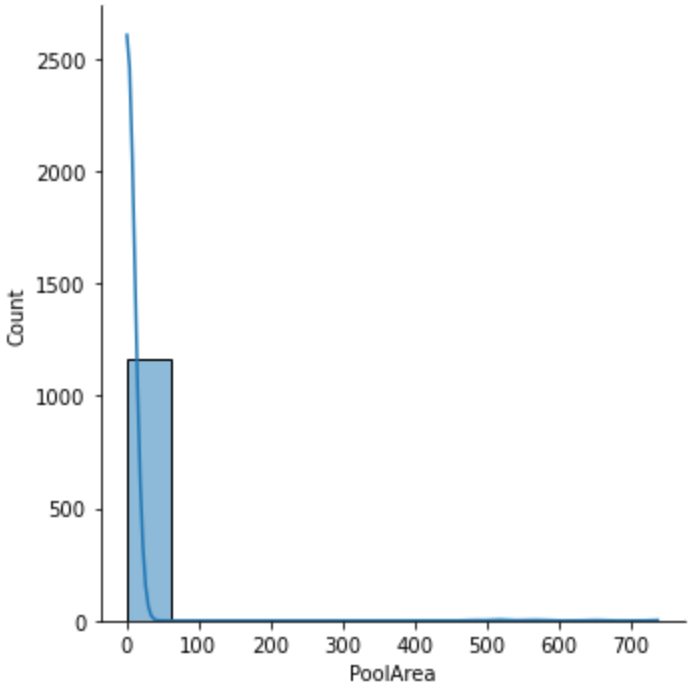

```
<Figure size 1440x576 with 0 Axes>
```



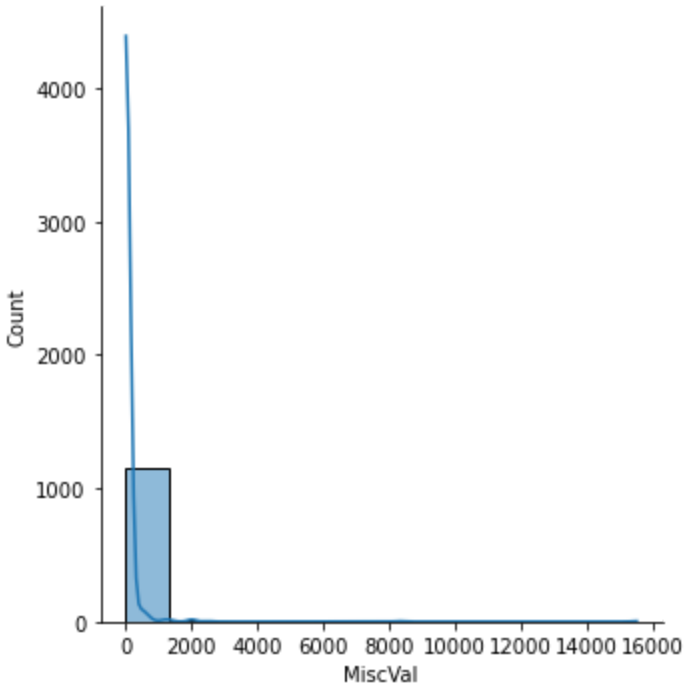

```
<Figure size 1440x576 with 0 Axes>
```



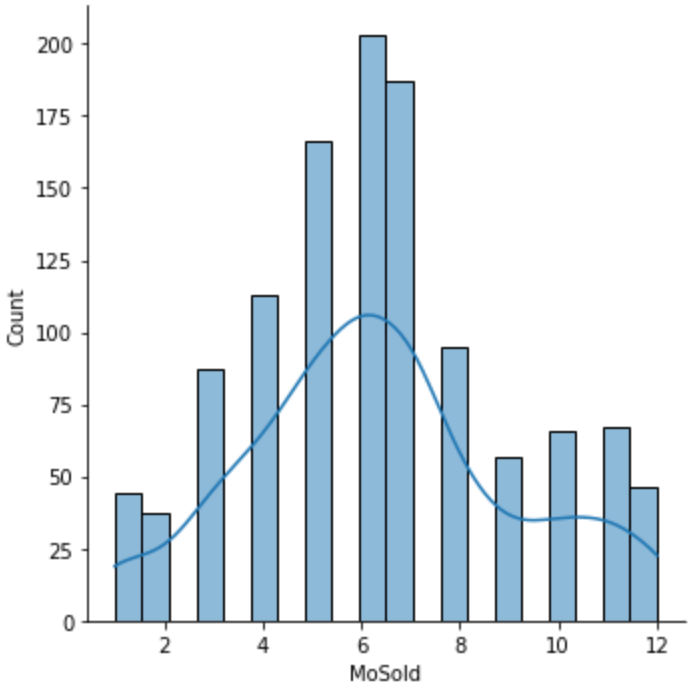
<Figure size 1440x576 with 0 Axes>



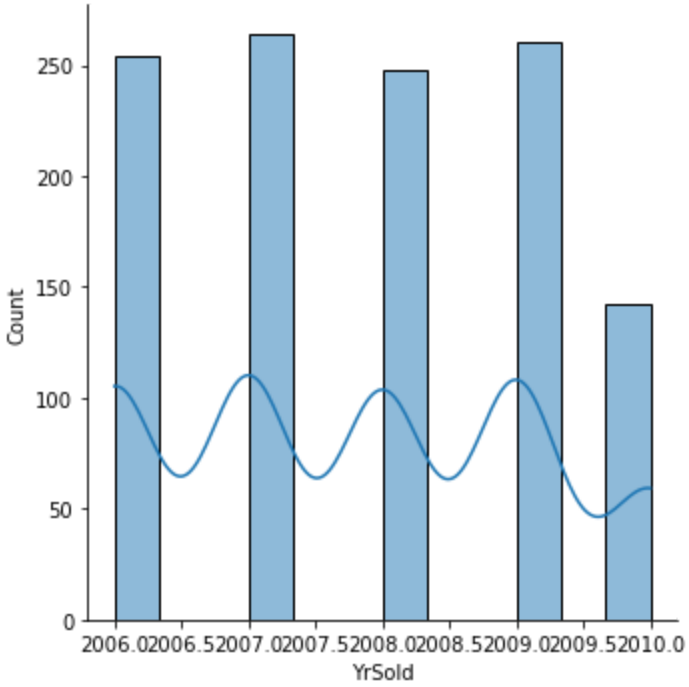
<Figure size 1440x576 with 0 Axes>



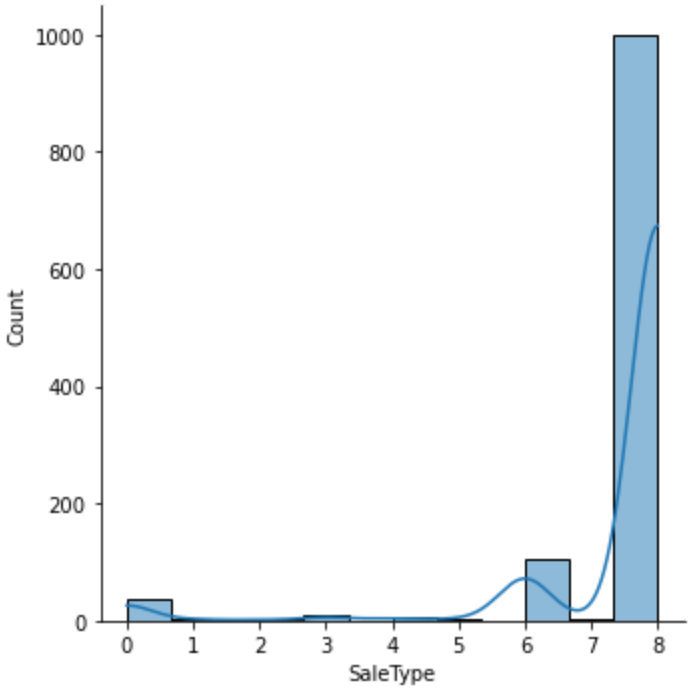
<Figure size 1440x576 with 0 Axes>



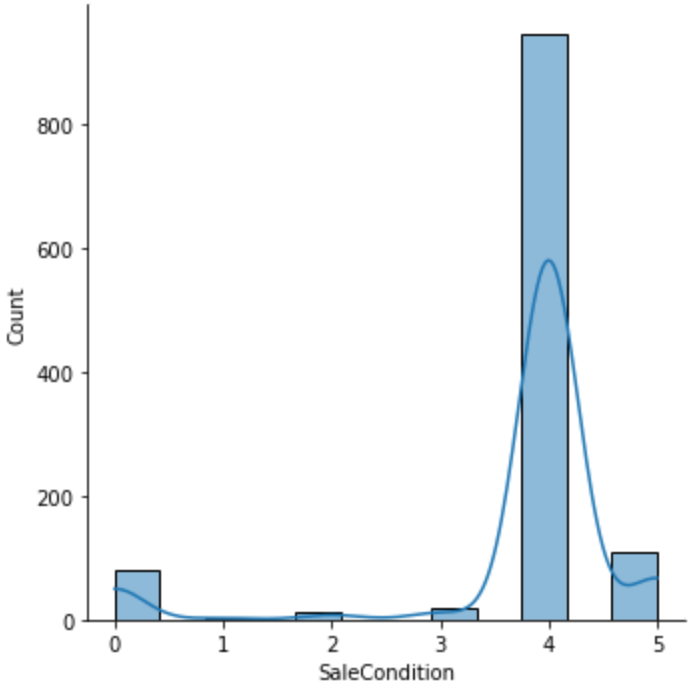
<Figure size 1440x576 with 0 Axes>



<Figure size 1440x576 with 0 Axes>



<Figure size 1440x576 with 0 Axes>



In [52]:
```
x.skew()
```

Out[52]:

MSSubClass	1.422019
MSZoning	-1.796785
LotFrontage	2.710383
LotArea	10.659285
Street	-17.021969
LotShape	-0.603775
LandContour	-3.125982
LotConfig	-1.118821
LandSlope	4.812568
Neighborhood	0.043735
Condition1	3.008289
Condition2	11.514458
BldgType	2.318657
HouseStyle	0.285680
OverallQual	0.175082
OverallCond	0.580714
YearBuilt	-0.579204
YearRemodAdd	-0.495864
RoofStyle	1.498560
RoofMatl	7.577352
Exterior1st	-0.612816
Exterior2nd	-0.592349
MasVnrType	-0.104609
MasVnrArea	2.834658
ExterQual	-1.810843
ExterCond	-2.516219
Foundation	-0.002761
BsmtQual	-1.343781
BsmtCond	-3.293554
BsmtExposure	-1.166987
BsmtFinType1	-0.068901
BsmtFinSF1	1.871606
BsmtFinType2	-3.615783
BsmtFinSF2	4.365829
BsmtUnfSF	0.909057
TotalBsmtSF	1.744591
Heating	10.103609
HeatingQC	0.449933
CentralAir	-3.475188
Electrical	-3.104209
1stFlrSF	1.513707

```
2ndFlrSF          0.823479
LowQualFinSF      8.666142
GrLivArea         1.449952
BsmtFullBath      0.627106
BsmtHalfBath      4.264403
FullBath          0.057809
HalfBath          0.656492
BedroomAbvGr      0.243855
KitchenAbvGr      4.365259
KitchenQual      -1.408106
TotRmsAbvGrd      0.644657
Functional       -3.999663
Fireplaces        0.671966
GarageType        0.831142
GarageYrBlt      -0.673163
GarageFinish     -0.450190
GarageCars       -0.358556
GarageArea        0.189665
GarageQual       -4.582386
GarageCond       -5.422472
PavedDrive       -3.274035
WoodDeckSF        1.504929
OpenPorchSF       2.410840
EnclosedPorch     3.043610
3SsnPorch         9.770611
ScreenPorch       4.105741
PoolArea         13.243711
MiscVal          23.065943
MoSold            0.220979
YrSold            0.115765
SaleType         -3.660513
SaleCondition    -2.671829
dtype: float64
```

Data is highly skewed. We need to reduce skewness.

Removing skewness

In [53]:
```
from sklearn.preprocessing import PowerTransformer

pt=PowerTransformer()

# removing skewness in train data
xs=pt.fit_transform(x)
x=pd.DataFrame(xs,columns=x.columns)

# removing skewness in train data
ts=pt.fit_transform(test)
test=pd.DataFrame(ts,columns=test.columns)
```

In [54]:
```
x.skew()
```

Out[54]:
```
MSSubClass        0.064007
MSZoning          0.233113
LotFrontage       0.161368
LotArea           0.032509
Street          -17.021969
LotShape         -0.594207
LandContour      -2.592303
LotConfig        -1.030401
LandSlope         3.954345
Neighborhood     -0.146541
Condition1        0.225468
Condition2        0.537277
```

```
BldgType          1.857194
HouseStyle       -0.080331
OverallQual       0.021658
OverallCond       0.048063
YearBuilt        -0.126641
YearRemodAdd     -0.225131
RoofStyle        -0.292233
RoofMatl         -6.314987
Exterior1st      -0.338023
Exterior2nd      -0.352793
MasVnrType       -0.016203
MasVnrArea        0.416370
ExterQual        -0.605112
ExterCond        -2.270791
Foundation        0.004296
BsmtQual         -0.413999
BsmtCond         -3.025865
BsmtExposure     -0.914214
BsmtFinType1     -0.206639
BsmtFinSF1       -0.404528
BsmtFinType2     -2.420885
BsmtFinSF2        2.394737
BsmtUnfSF        -0.284390
TotalBsmtSF       0.286779
Heating          -4.541694
HeatingQC         0.156511
CentralAir       -3.475188
Electrical       -3.006845
1stFlrSF         -0.002391
2ndFlrSF          0.280208
LowQualFinSF      6.922843
GrLivArea        -0.000054
BsmtFullBath      0.365488
BsmtHalfBath      3.954345
FullBath         -0.045944
HalfBath          0.498003
BedroomAbvGr      0.116498
KitchenAbvGr     -2.370593
KitchenQual      -0.435558
TotRmsAbvGrd      0.002332
Functional       -3.343664
Fireplaces        0.084950
GarageType        0.222501
GarageYrBlt      -0.147623
GarageFinish     -0.335248
GarageCars       -0.022970
GarageArea       -0.320370
GarageQual       -4.327379
GarageCond       -4.925781
PavedDrive       -3.025809
WoodDeckSF        0.113026
OpenPorchSF      -0.002749
EnclosedPorch     2.022616
3SsnPorch         7.087955
ScreenPorch       3.067153
PoolArea         12.817372
MiscVal           4.991071
MoSold           -0.035838
YrSold            0.112893
SaleType         -2.067563
SaleCondition    -0.353292
dtype: float64
```

Scaling

In [55]:
```
# scaling the data
from sklearn.preprocessing import MinMaxScaler, StandardScaler

scaler=MinMaxScaler()

# scaling train data
xd=scaler.fit_transform(x)
x=pd.DataFrame(xd,columns=x.columns)

# scaling test data
td=scaler.fit_transform(test)
test=pd.DataFrame(td,columns=test.columns)
```

In [56]:
```
x.head()
```

Out[56]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighb
0	0.830391	0.554691	0.328447	0.256316	1.0	0.0	1.0	1.000000	0.0	0
1	0.000000	0.554691	0.432505	0.492699	1.0	0.0	1.0	1.000000	1.0	0
2	0.542392	0.554691	0.420474	0.396442	1.0	0.0	1.0	0.103558	0.0	0
3	0.000000	0.554691	0.470981	0.430958	1.0	0.0	1.0	1.000000	0.0	0
4	0.000000	0.554691	0.328447	0.502514	1.0	0.0	1.0	0.298574	0.0	0

In [57]:
```
test.head()
```

Out[57]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighb
0	0.000000	0.673616	0.553513	0.447989	1.0	0.0	0.000421	0.000000	0.0	0
1	0.840802	0.673616	0.402786	0.268585	1.0	0.0	1.000000	0.084023	0.0	0
2	0.000000	0.673616	0.402786	0.411882	1.0	1.0	1.000000	1.000000	0.0	0
3	0.627085	0.673616	0.469927	0.414624	1.0	1.0	0.000000	1.000000	0.0	0
4	0.560136	0.673616	0.553513	0.454183	1.0	0.0	1.000000	0.084023	0.0	0

Co-relation

In [58]:
```
train.corr()
```

Out[58]:

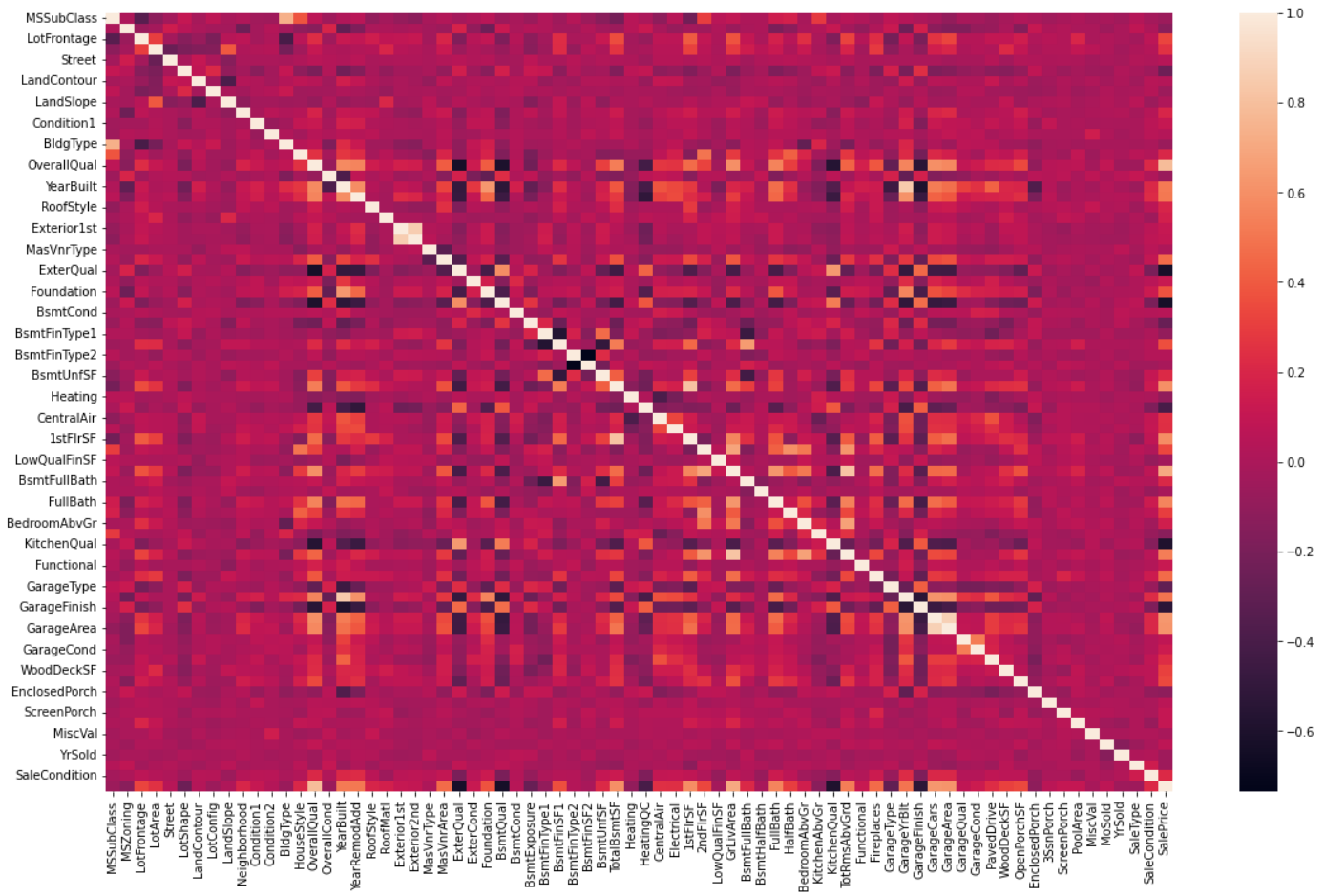
	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	L
MSSubClass	1.000000	0.007478	-0.336681	-0.124151	-0.035981	0.104485	-0.021387	0.076880	
MSZoning	0.007478	1.000000	-0.069661	-0.023328	0.140215	0.053655	0.001175	-0.027246	
LotFrontage	-0.336681	-0.069661	1.000000	0.299452	-0.035309	-0.144523	-0.073451	-0.192468	
LotArea	-0.124151	-0.023328	0.299452	1.000000	-0.263973	-0.189201	-0.159038	-0.152063	
Street	-0.035981	0.140215	-0.035309	-0.263973	1.000000	-0.012941	0.105226	0.000153	
LotShape	0.104485	0.053655	-0.144523	-0.189201	-0.012941	1.000000	0.081803	0.211395	
LandContour	-0.021387	0.001175	-0.073451	-0.159038	0.105226	0.081803	1.000000	-0.031496	
LotConfig	0.076880	-0.027246	-0.192468	-0.152063	0.000153	0.211395	-0.031496	1.000000	
LandSlope	-0.014930	-0.023952	0.046051	0.395410	-0.141572	-0.101187	-0.386787	-0.007934	

Neighborhood	0.013918	-0.251833	0.065824	0.010707	0.001420	-0.031852	0.037232	-0.030440
Condition1	-0.042474	-0.025651	-0.025581	0.029520	0.002189	-0.136885	0.021583	0.016659
Condition2	-0.044319	0.031959	0.011761	0.040096	0.001406	-0.032926	-0.026150	0.033318
BldgType	0.731815	-0.024776	-0.403220	-0.215345	-0.013606	0.099822	0.035434	0.112920
HouseStyle	0.381585	-0.110991	0.056321	-0.040637	0.016793	-0.116280	0.072799	-0.039075
OverallQual	0.070462	-0.134037	0.229218	0.107188	0.057140	-0.189636	0.045643	-0.031717
OverallCond	-0.056978	0.189553	-0.047573	0.017513	0.031082	0.026931	-0.038304	-0.027364
YearBuilt	0.023988	-0.299732	0.112655	0.005506	0.020292	-0.230080	0.153721	0.012933
YearRemodAdd	0.056618	-0.174586	0.088799	0.027228	0.057866	-0.155428	0.086936	-0.009281
RoofStyle	-0.100641	0.000913	0.146222	0.038615	-0.006875	-0.012410	0.009035	-0.025071
RoofMatl	-0.032214	0.009719	0.082566	0.194142	0.007889	-0.091341	-0.026604	-0.071760
Exterior1st	-0.090178	-0.012037	0.087727	0.048265	0.012482	-0.033798	-0.001932	0.001634
Exterior2nd	-0.120022	0.005548	0.110762	0.027328	0.014618	-0.034674	-0.027594	-0.011272
MasVnrType	-0.013252	-0.033521	-0.034388	-0.005204	0.000822	-0.006337	-0.090320	-0.009212
MasVnrArea	0.027813	-0.063862	0.188758	0.121086	0.024583	-0.086605	0.066604	-0.029937
ExterQual	-0.024133	0.184815	-0.169835	-0.057865	0.044509	0.158076	-0.010117	-0.018388
ExterCond	0.004186	-0.099269	0.044582	0.011438	0.017738	-0.047098	0.012069	0.036699
Foundation	0.053951	-0.244844	0.090734	-0.021195	0.032701	-0.136443	0.021138	-0.003660
BsmtQual	-0.052242	0.123822	-0.176402	-0.075262	-0.031959	0.162217	-0.000578	0.022857
BsmtCond	-0.013220	0.002336	0.043165	0.010165	-0.017190	-0.026293	0.040758	0.048697
BsmtExposure	-0.068192	0.037414	-0.123896	-0.136900	0.065253	0.123908	0.043353	-0.004270
BsmtFinType1	0.017515	0.021518	-0.031507	-0.053584	-0.006420	0.099214	-0.076907	0.008233
BsmtFinSF1	-0.052236	-0.034161	0.228996	0.221851	-0.009066	-0.120411	-0.029114	-0.021688
BsmtFinType2	0.040573	-0.031322	-0.007509	-0.084713	0.045208	0.029741	-0.050507	0.009830
BsmtFinSF2	-0.062403	0.027055	0.002159	0.056656	0.003550	-0.038124	0.030222	-0.015262
BsmtUnfSF	-0.134170	-0.032638	0.113924	0.006600	0.024205	-0.028545	0.029149	-0.006061
TotalBsmtSF	-0.214042	-0.058896	0.356107	0.259733	0.016422	-0.169062	0.010334	-0.034490
Heating	0.047734	0.051188	-0.026622	-0.020394	0.006981	0.071484	0.013956	-0.003764
HeatingQC	-0.001994	0.119852	-0.078262	-0.003337	-0.047968	0.092617	-0.076374	-0.012182
CentralAir	-0.113883	-0.015887	0.077404	0.051182	0.043024	-0.113719	0.103265	-0.013923
Electrical	0.045533	-0.079205	0.066868	0.050388	0.038720	-0.090459	0.098073	-0.043250
1stFlrSF	-0.227927	-0.033095	0.403436	0.312843	0.011486	-0.171928	-0.036866	-0.060678
2ndFlrSF	0.300366	-0.048968	0.089675	0.059803	0.040957	-0.066535	-0.034056	-0.045865
LowQualFinSF	0.053737	0.012098	0.007506	-0.001915	0.007352	0.028451	-0.088111	0.000657
GrLivArea	0.086448	-0.064125	0.374251	0.281360	0.043322	-0.180021	-0.064160	-0.083076
BsmtFullBath	0.004556	-0.007287	0.094046	0.142387	-0.036483	-0.067889	-0.006165	-0.006128
BsmtHalfBath	0.008207	0.006345	0.001389	0.059282	0.013788	-0.017926	0.013725	-0.016040
FullBath	0.140807	-0.188837	0.171773	0.123197	0.033208	-0.175359	0.047016	0.000177

HalfBath	0.168423	-0.121209	0.047816	0.007271	0.045146	-0.122586	0.030155	-0.020604
BedroomAbvGr	-0.013283	-0.001906	0.237199	0.117351	0.027587	-0.054438	-0.042108	-0.057080
KitchenAbvGr	0.283506	0.026744	-0.002729	-0.013075	0.012304	0.082102	-0.067071	-0.002959
KitchenQual	-0.011378	0.111689	-0.174713	-0.064278	-0.012056	0.122861	0.027230	-0.013554
TotRmsAbvGrd	0.051179	-0.031062	0.318771	0.184546	0.038259	-0.125990	-0.052622	-0.042657
Functional	0.022913	-0.091684	0.039272	-0.033165	-0.015309	-0.012018	0.019517	-0.016756
Fireplaces	-0.035792	0.010658	0.230610	0.285983	0.010574	-0.183316	-0.053860	-0.056197
GarageType	0.077469	0.125155	-0.216237	-0.123746	-0.000824	0.160087	-0.076191	0.016495
GarageYrBlt	0.032568	-0.242276	0.080386	-0.004754	0.002307	-0.202091	0.137992	-0.020992
GarageFinish	-0.000067	0.162566	-0.192961	-0.117373	-0.001178	0.239653	-0.077940	0.014064
GarageCars	-0.027639	-0.126031	0.260252	0.158313	0.002088	-0.183999	0.046954	-0.067087
GarageArea	-0.092408	-0.168364	0.322644	0.195162	-0.025537	-0.153079	0.043263	-0.063637
GarageQual	-0.010568	-0.168881	0.041522	0.005999	-0.012622	-0.097438	0.008800	0.033125
GarageCond	-0.025595	-0.087375	0.043699	0.035657	-0.010973	-0.068449	0.005124	0.034690
PavedDrive	-0.068702	-0.077280	0.092551	0.021907	0.041318	-0.122756	0.111451	-0.034578
WoodDeckSF	-0.022609	-0.004509	0.088334	0.216720	-0.033142	-0.142202	-0.011580	-0.042424
OpenPorchSF	0.017468	-0.152694	0.151328	0.093080	0.029649	-0.090534	0.034995	-0.052485
EnclosedPorch	-0.004252	0.111221	0.020902	-0.007446	0.021360	0.078660	-0.061782	-0.069765
3SsnPorch	-0.043210	0.004409	0.051084	0.025794	0.007338	-0.016018	-0.032990	-0.049588
ScreenPorch	-0.013291	0.030793	0.030405	0.025256	0.016026	-0.057748	0.016993	0.005699
PoolArea	0.009583	-0.001663	0.196001	0.097107	0.004505	-0.022160	-0.014235	-0.051021
MiscVal	-0.023503	0.003416	-0.001329	0.051679	-0.030354	-0.049577	0.020147	-0.023326
MoSold	-0.016015	-0.051646	0.022517	0.015141	-0.008860	-0.050418	-0.023872	0.019084
YrSold	-0.038595	-0.004964	-0.003885	-0.035399	-0.019635	0.021421	0.009499	-0.009817
SaleType	0.035050	0.079854	-0.035356	0.005421	0.025920	-0.015161	-0.041763	-0.002039
SaleCondition	-0.028981	0.004501	0.065091	0.034236	0.014176	-0.054905	0.047715	0.043692
SalePrice	-0.060775	-0.133221	0.323779	0.249499	0.044753	-0.248171	0.032836	-0.060452

In [59]:
```
plt.figure(figsize=(20,12))
sns.heatmap(train.corr())
```

Out[59]:
```
<AxesSubplot:>
```



In [60]:
```
train.corr()['SalePrice']
```

Out[60]:
```
MSSubClass      -0.060775
MSZoning        -0.133221
LotFrontage      0.323779
LotArea          0.249499
Street           0.044753
LotShape        -0.248171
LandContour      0.032836
LotConfig       -0.060452
LandSlope        0.015485
Neighborhood     0.198942
Condition1       0.105820
Condition2       0.033956
BldgType        -0.066028
HouseStyle       0.205502
OverallQual      0.789185
OverallCond     -0.065642
YearBuilt        0.514408
YearRemodAdd     0.507831
RoofStyle        0.192654
RoofMatl         0.159865
Exterior1st      0.108451
Exterior2nd      0.097541
MasVnrType       0.007732
MasVnrArea       0.463626
ExterQual       -0.624820
ExterCond        0.115167
Foundation       0.374169
BsmtQual        -0.626850
BsmtCond         0.048125
BsmtExposure    -0.268559
BsmtFinType1    -0.092109
BsmtFinSF1       0.362874
BsmtFinType2     0.032285
```

```
BsmtFinSF2      -0.010151
BsmtUnfSF        0.215724
TotalBsmtSF      0.595042
Heating         -0.100021
HeatingQC       -0.406604
CentralAir       0.246754
Electrical       0.234621
1stFlrSF         0.587642
2ndFlrSF         0.330386
LowQualFinSF    -0.032381
GrLivArea        0.707300
BsmtFullBath     0.212924
BsmtHalfBath    -0.011109
FullBath         0.554988
HalfBath         0.295592
BedroomAbvGr     0.158281
KitchenAbvGr    -0.132108
KitchenQual     -0.592468
TotRmsAbvGrd     0.528363
Functional       0.118673
Fireplaces       0.459611
GarageType      -0.299470
GarageYrBlt      0.497402
GarageFinish    -0.537121
GarageCars       0.628329
GarageArea       0.619000
GarageQual       0.080795
GarageCond       0.135071
PavedDrive       0.231707
WoodDeckSF       0.315444
OpenPorchSF      0.339500
EnclosedPorch   -0.115004
3SsnPorch        0.060119
ScreenPorch      0.100284
PoolArea         0.103280
MiscVal         -0.013071
MoSold           0.072764
YrSold          -0.045508
SaleType        -0.050851
SaleCondition    0.217687
SalePrice        1.000000
Name: SalePrice, dtype: float64
```

In [61]:
```
x.skew()
```

Out[61]:
```
MSSubClass        0.064007
MSZoning          0.233113
LotFrontage       0.161368
LotArea           0.032509
Street          -17.021969
LotShape         -0.594207
LandContour      -2.592303
LotConfig        -1.030401
LandSlope         3.954345
Neighborhood     -0.146541
Condition1        0.225468
Condition2        0.537277
BldgType          1.857194
HouseStyle       -0.080331
OverallQual       0.021658
OverallCond       0.048063
YearBuilt        -0.126641
YearRemodAdd     -0.225131
RoofStyle        -0.292233
RoofMatl         -6.314987
Exterior1st      -0.338023
```

```
Exterior2nd      -0.352793
MasVnrType       -0.016203
MasVnrArea        0.416370
ExterQual        -0.605112
ExterCond        -2.270791
Foundation        0.004296
BsmtQual         -0.413999
BsmtCond         -3.025865
BsmtExposure     -0.914214
BsmtFinType1     -0.206639
BsmtFinSF1       -0.404528
BsmtFinType2     -2.420885
BsmtFinSF2        2.394737
BsmtUnfSF        -0.284390
TotalBsmtSF       0.286779
Heating          -4.541694
HeatingQC         0.156511
CentralAir       -3.475188
Electrical       -3.006845
1stFlrSF         -0.002391
2ndFlrSF          0.280208
LowQualFinSF      6.922843
GrLivArea        -0.000054
BsmtFullBath      0.365488
BsmtHalfBath      3.954345
FullBath         -0.045944
HalfBath          0.498003
BedroomAbvGr      0.116498
KitchenAbvGr     -2.370593
KitchenQual      -0.435558
TotRmsAbvGrd      0.002332
Functional       -3.343664
Fireplaces        0.084950
GarageType        0.222501
GarageYrBlt      -0.147623
GarageFinish     -0.335248
GarageCars       -0.022970
GarageArea       -0.320370
GarageQual       -4.327379
GarageCond       -4.925781
PavedDrive       -3.025809
WoodDeckSF        0.113026
OpenPorchSF      -0.002749
EnclosedPorch     2.022616
3SsnPorch         7.087955
ScreenPorch       3.067153
PoolArea         12.817372
MiscVal           4.991071
MoSold           -0.035838
YrSold            0.112893
SaleType         -2.067563
SaleCondition    -0.353292
dtype: float64
```

We will drop columns with high skewness and lower corelation with target.

In [62]:
```
# for train data
x.drop(['Street','LandContour','LandSlope','BsmtCond','BsmtFinType2','BsmtFinSF2',
        'LowQualFinSF','BsmtHalfBath','GarageQual','3SsnPorch','MiscVal', 'SaleType','Po

# for test data
test.drop(['Street','LandContour','LandSlope','BsmtCond','BsmtFinType2','BsmtFinSF2',
        'LowQualFinSF','BsmtHalfBath','GarageQual','3SsnPorch','MiscVal','SaleType','Poo
```

PCA

In [63]:
```
from sklearn.decomposition import PCA
pca = PCA().fit(x)
```

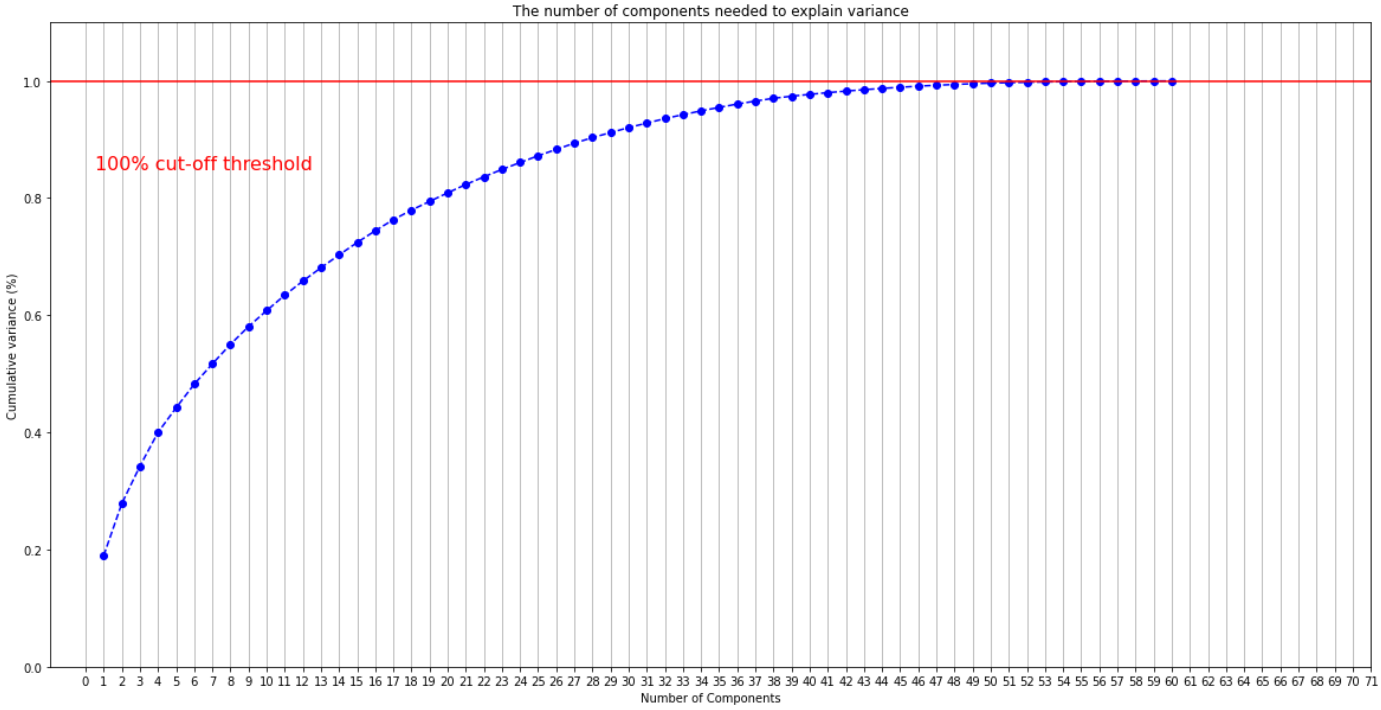
In [64]:
```
fig, ax = plt.subplots(figsize=(20,10))
xi = np.arange(1, 61, step=1)
yi = np.cumsum(pca.explained_variance_ratio_)

plt.ylim(0.0,1.1)
plt.plot(xi, yi, marker='o', linestyle='--', color='b')

plt.xlabel('Number of Components')
plt.xticks(np.arange(0, 72, step=1)) #change from 0-based array index to 1-based human-r
plt.ylabel('Cumulative variance (%)')
plt.title('The number of components needed to explain variance')

plt.axhline(y=1, color='r', linestyle='-')
plt.text(0.5, 0.85, '100% cut-off threshold', color = 'red', fontsize=16)

ax.grid(axis='x')
plt.show()
```



We can see with 55 feature we are able to retain most information.

In [65]:
```
pca=PCA(n_components=55)
xp=pca.fit_transform(x)
xn=pd.DataFrame(xp)
xn.head()
```

Out[65]:

	0	1	2	3	4	5	6	7	8	9	
0	-0.277890	-0.257006	0.322190	0.185692	0.032661	0.925131	0.230429	-0.796550	0.318061	-0.085202	-0.1
1	0.381486	-0.882948	-0.061733	-0.526325	0.609025	-0.641452	-0.196856	-0.464658	0.504783	0.183782	-0.2
2	0.917723	0.115078	0.859545	-0.494155	0.812493	-0.045460	0.143028	0.023608	-0.207912	-0.438158	0.0
3	0.257209	-0.578336	-0.092906	-0.407012	-0.860720	0.153112	0.098949	-0.355227	0.263063	0.548149	-0.1
4	0.537968	-0.855542	0.320952	-0.601114	-0.357694	0.400317	0.013799	0.005464	-0.222696	0.565547	0.5

In [66]:
```
pca=PCA(n_components=55)
testp=pca.fit_transform(test)
testn=pd.DataFrame(testp)
testn.head()
```

Out[66]:

	0	1	2	3	4	5	6	7	8	9	
0	1.914412	-0.892921	-0.152342	-0.495919	0.368212	-0.066282	0.279334	0.059075	-0.331545	-0.213135	0.0
1	0.522173	-0.780945	0.983392	0.123314	1.085831	-0.270471	0.342653	0.444628	-0.534270	0.248163	-0.3
2	0.980603	-0.151094	-1.233284	0.246157	-0.016305	-0.401129	-0.242060	0.656951	-0.282779	0.126195	0.1
3	-0.599523	0.887916	0.014310	-0.109241	-0.302985	0.556104	-0.021978	0.106593	0.447900	0.215933	-0.1
4	1.463070	1.031109	-0.258033	-0.613199	0.518961	-0.193346	0.188294	-0.198574	-0.291059	0.052083	-0.1

ML Modeling

In [67]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import r2_score, max_error, mean_squared_error

# Checking the best random state
lnr=LinearRegression()
for i in range(1,100):
    x_train, x_test, y_train, y_test=train_test_split(xn,y, test_size=0.2, random_state=
    lnr.fit(x_train,y_train)
    pred_train=lnr.predict(x_train)
    pred_test=lnr.predict(x_test)
    if r2_score(y_train,pred_train)*100>81.5 and r2_score(y_test,pred_test)*100>81.5:
        print('At random state',i,'the train accuracy is', r2_score(y_train,pred_train))
        print('At random state',i,'the test accuracy is', r2_score(y_test,pred_test))
        print('\n')
```

```
At random state 19 the train accuracy is 0.8155270605850127
At random state 19 the test accuracy is 0.8236330859092318


At random state 33 the train accuracy is 0.8158638327122185
At random state 33 the test accuracy is 0.8192472111033362


At random state 55 the train accuracy is 0.8157286956508712
At random state 55 the test accuracy is 0.8228124063475402


At random state 56 the train accuracy is 0.8153412103567637
At random state 56 the test accuracy is 0.8229140193114369


At random state 75 the train accuracy is 0.8170974191051683
At random state 75 the test accuracy is 0.8189827750839296


At random state 89 the train accuracy is 0.8154212763317712
At random state 89 the test accuracy is 0.8216359111681243
```

At random stae=75 we are getting the best accuracy of our model. So we will use

random state=75.

LinearRegression

In [68]:
```
x_train, x_test, y_train, y_test=train_test_split(xn,y, test_size=0.2, random_state=75)
lnr.fit(x_train,y_train)

pred_test_lnr=lnr.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_lnr))
print('\n max error', max_error(y_test,pred_test_lnr))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_lnr, squared=Fal
print('\n MAPE',(abs((y_test-pred_test_lnr)/y_test).mean())*100)
```

```
r2 score 0.8189827750839296

 max error 315386.43660979276

 Root mean squared error 33712.12569326364

 MAPE 13.925965928286562
```

In [69]:
```
# Checking for the best value of cross fold
for j in range(2,15):
    cv_score=cross_val_score(lnr,x,y,cv=j)
    cv_mean=cv_score.mean()
    print('At cv=',j, 'cross vallue score is', cv_mean)
    print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_te
```

```
At cv= 2 cross vallue score is 0.7701372373291364
Differnce between in r2_score and cross value score 0.04884553775479317
At cv= 3 cross vallue score is 0.7784977951070046
Differnce between in r2_score and cross value score 0.04048497997692502
At cv= 4 cross vallue score is 0.781641900936302
Differnce between in r2_score and cross value score 0.037340874147627545
At cv= 5 cross vallue score is 0.7782034618613342
Differnce between in r2_score and cross value score 0.040779313222595426
At cv= 6 cross vallue score is 0.7845277441038313
Differnce between in r2_score and cross value score 0.03445503098009828
At cv= 7 cross vallue score is 0.7843371908851765
Differnce between in r2_score and cross value score 0.03464558419875308
At cv= 8 cross vallue score is 0.7843303542288915
Differnce between in r2_score and cross value score 0.03465242085503806
At cv= 9 cross vallue score is 0.7881511404399691
Differnce between in r2_score and cross value score 0.030831634643960437
At cv= 10 cross vallue score is 0.7871754216406098
Differnce between in r2_score and cross value score 0.03180735344331975
At cv= 11 cross vallue score is 0.7850206672854589
Differnce between in r2_score and cross value score 0.03396210779847064
At cv= 12 cross vallue score is 0.7828614658830578
Differnce between in r2_score and cross value score 0.036121309200871754
At cv= 13 cross vallue score is 0.7891877459785804
Differnce between in r2_score and cross value score 0.029795029105349147
At cv= 14 cross vallue score is 0.7731944718015564
Differnce between in r2_score and cross value score 0.045788303282373155
```

At CV=13 we are getting minimum differnce between in r2_score and cross value score

In [86]:
```
# cross validation for lnr model
cv_score=cross_val_score(lnr,x,y,cv=13)
cv_mean=cv_score.mean()
```

```
print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross val score', r2_score(y_test,pred_test_lnr
```

```
cross val score is 0.7891877459785804
Differnce between in r2_score and cross val score 0.029795029105349147
```

Predicting output for test data

In [87]:
```
# predicting for test set we are given
prediction_lnr=lnr.predict(testn)
prediction_lnr
```

Out[87]:
```
array([311010.94936028, 212571.81719534, 268437.08751059, 180663.22647393,
       228029.74991285, 115512.15451622, 143259.63155506, 237889.74861919,
       237796.91706727, 172870.46982558, 111914.17414661, 227265.58027843,
        84029.953096  , 355999.05717656, 280299.64657687,  64388.15338745,
        72317.38641733,  85125.36254713, 228218.87209972, 244282.06942747,
        85992.51932638, 225135.97999311, 154326.66671259,  79694.91287859,
        93152.76867198, 133709.46496537, 205592.73374791, 144667.43161203,
       207461.24776541, 122735.67780557, -11127.22757839, 182274.80926792,
       276002.79134779, 219261.19411427, 127978.7493755 , 207577.46897565,
       281311.36392495, 104700.27318787, 177834.61257407, 191758.71492227,
        68162.40061979, 215983.38666502, 248854.0213722 , 194848.76453316,
       210255.38403061, 124178.86134719,  99796.36752231,  54738.67519935,
       255835.81561606, 289903.39111941, 140442.69725683, 177731.05696275,
        45687.42011663, 100137.01382886, 259472.63192177, 197011.10211271,
        96780.28303064, 230002.39773188,  72582.61166878, 234652.31423235,
       130689.70746712, 270080.33609652, 161930.06746951, 231864.14948259,
       266066.35039842, 116364.18004437, 205036.34638128, 192065.77188257,
       120754.5740771 , 141461.39830576, 259955.89452012, 169242.78903059,
       154666.01028218,  69182.69889698, 239845.2609189 , 202142.57365407,
       324343.25141212, 230686.94978223, 309420.13442048, 261050.13375934,
       269698.97783438, 151107.07953972, 154584.98228585, 158903.19851032,
       260350.70366758, 179498.91597294,  79840.21892679, 287159.18922651,
       173419.70218567, 219607.03927327, 221155.519686  , 144570.96792671,
       125610.47882736, 147781.24764147, 197113.93130786, 215138.41559659,
       334380.38767362, 227569.38791915, 270162.4659555 , 175570.76022374,
       273066.27157553, 188954.47166376, 175446.37010016, 152316.31891381,
       172437.72699772, 112280.66342916, 279567.06939073, 118957.03103861,
       185539.28963505, 126307.7862104 , 232638.34790705, 226192.83024807,
       108315.37248362, 312352.11236163, 125375.76278444,  33007.34158248,
        97585.17539059, 213237.18250716, 202643.92527086, 119486.15204661,
       166848.27134323, 257772.21682243, 148918.22882399, 112964.22352599,
        98383.45438599, 201604.84082506, 106114.37836463, 244193.08288621,
       -55114.51336592,  54582.85713975, 218632.87809685, 211413.25697063,
       203069.01526228, 223811.21836942, 112757.7527635 , 255987.13544644,
       302109.48744266, 109433.76745581, 252022.2048304 , 157125.23904223,
       160866.60642653, 339898.03948054,  36377.49668071, 310436.80215414,
       232246.50494428, 130324.83570814, 132448.90531857,  32569.34205113,
        23114.36798877, 246029.39332841, 163403.73834675, 166370.79194154,
       240149.04513919,  97738.61412708, 139864.63219137, 183719.25561095,
       241096.73006879, 219695.135407  , 141850.4169408 , 222176.36545532,
       260361.54321099, 196153.16606562, 243411.35142532, 134878.21178119,
        69169.01968407, 234899.62596096, 244147.92954345, 228153.79379218,
       124727.37979038, 172511.27934608, 149462.64985524, 115260.40015997,
       145084.67473863, 221225.40226786, 105255.22883873, 301822.45910099,
        79182.31976118,  95574.8625789 , 176383.37442617, 172921.56984009,
       188408.95148444, 171242.84306851, 292930.25561644, 156163.44451292,
       331040.43967955, 315915.95352364, 230557.57442923, 147251.74108187,
       146631.11756169, 126978.33599987, 140793.36675654, 249016.22176784,
       303094.11333204,  95090.38148648, 169515.82246699,  60661.11033246,
       157319.81114879, 198007.0498256 , 113705.06045401, 133513.85527801,
       230129.59167417, 181910.11969415, 306706.94641923, 268894.98302528,
       126878.21817328, 205560.61531517, 282863.61465067, 185621.30634922,
```

```
 55941.15774018, 233728.41339508, 151126.64448833, 122403.5859989 ,
 99180.87898941, 231299.99806393,  80411.50286816, 111699.1145084 ,
181172.07152371, 104475.00181295, 147042.63481013, 229539.7340492 ,
207954.99816348, 219595.82536354, 279054.07373852, 135313.70306476,
209221.34553339, 331898.03479739, 173098.10333788, 112281.54099224,
318471.67689648, 181090.61922941,  67062.90258348, 227611.36090072,
102995.53938681, 124529.58182494, 105538.54897245, 169395.6095319 ,
179045.20460432, 256009.50287437, 210381.00686053, 177862.67652888,
260466.06742238, 162922.34614501, 253688.89443255, 124028.56882164,
136435.87225459, 208712.74382718, 235421.30560194, 348564.0054205 ,
198732.68671183, 113380.88086988, 256206.49333475, 199352.6250201 ,
 77115.37063348, 105116.24118156,  99226.30969851, 190530.08266776,
 43938.62661089, 289884.63951006, 207726.17140845, 120220.10300397,
264438.20481071, 235219.69897555,  81073.62977217, 131156.81145664,
163185.55636955, 222598.78990384, 102317.38399961, 214420.83976673,
231942.19614637, 220118.75537263,  87490.84500629,  55543.53653331,
305102.60703085, 179796.26885389, 298414.04861986, 246025.09955564,
217182.75410334, 109874.9442708 , 226788.01294568, 329088.50397372,
133771.35545404, 324209.41183388, 155288.04705397, 282464.81456073,
209983.24704063,  88268.64018048, 153414.06632399, 262202.62609886,
131047.3704293 , 119609.1667111 , 174475.75711272, 158838.68285695])
```

DecisionTreeRegressor

In [88]:
```
from sklearn.tree import DecisionTreeRegressor

dtr=DecisionTreeRegressor()
dtr.fit(x_train, y_train)

pred_test_dtr=lnr.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_dtr))
print('\n max error', max_error(y_test,pred_test_dtr))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_dtr, squared=Fal
print('\n MAPE',(abs((y_test-pred_test_dtr)/y_test).mean())*100)
```

```
r2 score 0.8189827750839296

 max error 315386.43660979276

 Root mean squared error 33712.12569326364

 MAPE 13.925965928286562
```

In [90]:
```
# cross validation for dtr model
cv_score=cross_val_score(dtr, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross val score', r2_score(y_test,pred_test_dtr
```

```
cross val score is 0.7019363831231034
Differnce between in r2_score and cross val score 0.11704639196082622
```

In [91]:
```
# predicting for test set we are given
prediction_dtr=dtr.predict(testn)
prediction_dtr
```

Out[91]:
```
array([325624., 200000., 190000., 154900., 219210., 160000., 141000.,
       266000., 228000., 160200., 117000., 160000., 108480., 158000.,
       262500., 106000., 147000.,  97500., 181000., 262280., 105500.,
       120000., 143750., 127000., 115000., 169000., 155000., 154900.,
       167500.,  58500.,  80000., 195400., 215000., 295000., 121600.,
       272000., 195400., 110500., 171000., 129000., 105500., 242000.,
       187000., 193500., 256000., 120000., 132500., 123000., 235000.,
```

```
       380000., 135000., 165000., 104900.,  35311., 267000., 171000.,
       104000., 185000., 158000., 214000., 147000., 180000., 161000.,
       175500., 227000., 144000., 153500., 194500., 124000., 243000.,
       201000.,  75500., 136500.,  96500., 168500., 237000., 354000.,
       153500., 235000., 189950., 183500., 239000., 134500., 207500.,
       179900., 198900., 109000., 611657., 135000., 266000., 220000.,
        75000., 119900., 132000., 181900., 160200., 262500., 176500.,
       380000., 140000., 233170., 184000., 157900., 272000., 197000.,
       139000., 212000., 127000., 215000., 179200., 228000., 203000.,
       110000., 184000., 147000.,  87000., 105500., 197900., 250000.,
       155000.,  82500., 215000., 181000., 140000., 106000., 128000.,
       135750., 171000.,  72500., 115000., 163000., 181000., 139000.,
       191000., 147000., 383970., 316600., 205000., 275000., 115000.,
       177000., 611657., 128000., 556581., 189000., 191000.,  98600.,
       168000., 128000., 233170., 141000., 133000., 183500., 128200.,
       135000., 122500., 223500., 137900., 150000., 191000., 255900.,
       217000., 200500., 144000., 155000., 377500., 159000., 172500.,
       110000., 230000., 302000., 129000., 180000., 345000., 134500.,
       222000., 101000., 150000., 180000., 200000., 215200., 151000.,
       315750., 191000., 383970., 465000., 262000., 167000., 129000.,
       241500., 144000., 271900., 208300., 157900., 112500.,  72500.,
       206300., 221000., 109000., 129000., 174000., 108000., 485000.,
       465000., 106000., 167900., 235000., 137900., 102000., 168000.,
       128000., 107900.,  97500., 271900., 123000.,  55000., 174000.,
        97500., 127500., 179000., 193879., 212000., 380000., 171500.,
       150000., 360000., 173000.,  87000., 310000., 227680.,  80500.,
       197000., 126500., 162500.,  97500., 158000., 302000., 181900.,
       190000., 208500., 275000., 153000., 164700., 106000., 150000.,
       226000., 241500., 230000., 140000., 167000., 134900., 227000.,
       119000., 106000., 132000., 197000.,  93000., 394432., 180000.,
       142000., 211000., 173000., 124500., 104000., 208500., 191000.,
       124000., 160000., 227000., 207500., 124000., 110000., 415298.,
       120000., 277500., 227000., 211000., 140000., 175500., 360000.,
       134900., 394432.,  82500., 232000., 295000., 110000., 179500.,
       236500., 157000., 215000., 125500., 153500.])
```

SVR

In [92]:
```
from sklearn.svm import SVR

svr=SVR()
svr.fit(x_train, y_train)

pred_test_svr=svr.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_svr))
print('\n max error', max_error(y_test,pred_test_svr))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_svr, squared=Fal
print('\n MAPE',(abs((y_test-pred_test_svr)/y_test).mean())*100)
```

```
r2 score -0.05356266627398587

 max error 590935.9881206203

 Root mean squared error 81331.04875626473

 MAPE 32.35590782561008
```

In [93]:
```
# cross validation for svr model
cv_score=cross_val_score(svr, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross val score', r2_score(y_test,pred_test_svr
```

```
cross val score is -0.05971803951376769
Differnce between in r2_score and cross val score 0.006155373239781818
```

In [94]:
```
# predicting for test set we are given
prediction_svr=svr.predict(testn)
prediction_svr
```

Out[94]:
```
array([164076.04103621, 164019.43119046, 164051.24802183, 163984.67477543,
       164071.80683273, 163972.61467653, 164008.00034925, 164058.00265019,
       164036.55234085, 164017.4199148 , 163970.84016823, 163980.20989964,
       163963.9650187 , 164010.86960147, 164046.97447434, 163962.12274955,
       163977.66070765, 163953.15063272, 164026.0151052 , 164023.58081581,
       163970.1514408 , 164009.27675143, 163991.65315231, 163984.84330996,
       163954.00965301, 163978.62241641, 164022.2962231 , 163982.26454325,
       164006.61810165, 163965.74688537, 163964.7984876 , 164040.63867352,
       164050.18240006, 163997.15755203, 163979.02679031, 164031.96211053,
       164050.41916479, 163984.19845288, 164002.54534799, 163983.65438057,
       163950.78824059, 164059.38000142, 164063.95435695, 164031.71421298,
       164014.40726853, 163961.7975499 , 163958.35022095, 163963.55368966,
       164055.58676174, 164064.90330901, 163970.72140363, 164020.68421946,
       163951.34180554, 163958.56708313, 164065.37265442, 163998.63425657,
       163957.18381702, 164047.09648963, 163992.35487442, 164054.61708787,
       163983.22570031, 164029.3215208 , 163967.89227897, 164028.81199719,
       164055.09192344, 163976.16645501, 164004.58343965, 164039.21045941,
       163994.28555139, 163977.72740371, 164054.72066632, 163997.38209826,
       163977.05133839, 163980.24285515, 164010.4634471 , 164070.37399043,
       164074.51206226, 164031.09955608, 164067.41400202, 163998.20880878,
       164064.88491895, 163967.20085871, 163981.84714284, 163984.59589951,
       164032.11896338, 164023.19363595, 163949.59582856, 164069.62911415,
       163975.32638757, 164039.60468351, 164035.43961379, 163976.43835539,
       163973.7930531 , 163967.29648091, 164031.34390989, 164007.02128494,
       164072.0241679 , 164045.06800908, 164066.20278354, 163987.32989884,
       164067.80707447, 163977.61459857, 163973.94831446, 164017.27826095,
       164025.76072687, 163976.70464066, 164044.94280246, 163982.19391934,
       164045.9995318 , 163996.06721918, 164018.04362093, 164022.9094643 ,
       163981.38261475, 164065.71412001, 163977.9835781 , 163960.85318465,
       163967.75701311, 164040.62327186, 164000.37729093, 163949.86451741,
       163980.00724195, 164041.98903739, 164024.44779294, 163982.53238033,
       163966.25753064, 164006.19876266, 163938.88037232, 163990.32169252,
       163987.22463287, 163963.46851562, 164011.25399564, 164062.42696725,
       163982.76332848, 164024.58701277, 163992.0944688 , 164079.84847252,
       164061.29503728, 164000.3499321 , 164048.04917899, 163981.40244202,
       163968.93652256, 164042.65399404, 163955.66541057, 164045.86554568,
       164011.83877128, 164046.16456001, 163977.83051585, 163987.04507143,
       163950.08170464, 164055.18916976, 164028.41577198, 163968.26548855,
       164060.99788381, 163956.07511408, 163966.00825661, 164011.82819769,
       164025.28073361, 163994.3691347 , 163973.99714049, 164029.23357012,
       164051.08384419, 163996.27372218, 164037.08606999, 163966.51322114,
       163962.48099341, 164071.11631496, 164010.75357485, 164065.51525464,
       163960.48678369, 164035.81876318, 164005.52877908, 163983.53514355,
       163991.52286953, 164052.8894644 , 163973.28602561, 164073.22825757,
       163962.56866088, 163935.71919383, 163994.98131225, 164014.59965963,
       164042.16384822, 163991.58462726, 164062.06425348, 164012.34682285,
       164049.3549625 , 164078.83346482, 164015.2429009 , 163969.44785007,
       164007.53830735, 164020.61575243, 163979.81262744, 164041.53784794,
       164050.77025118, 163974.84180035, 163966.00159962, 163959.08179853,
       164019.17021112, 164053.58059727, 163963.646376  , 163997.58497024,
       164033.72370101, 163982.06020612, 164069.32408666, 164051.23213885,
       163958.35591894, 163990.11313827, 164058.24606638, 163978.52141486,
       163967.61576831, 164010.31319428, 163964.78676627, 163977.59238829,
       163974.46250441, 164015.50129678, 163968.91445956, 163969.94668411,
       164024.43188253, 163975.05668294, 163972.67727349, 163976.35647304,
       164053.07033193, 164069.01779162, 164063.92114894, 163961.69652969,
       164021.47050072, 164078.09711769, 164031.86017767, 163969.69376906,
       164071.11186286, 164019.29442078, 163952.32132678, 164014.95320275,
       163978.12154215, 163971.91379611, 163962.3994006 , 164019.71924853,
```

```
       164024.3775548 , 164023.74959296, 164019.37120976, 164051.09177972,
       164072.39387365, 163983.7266802 , 164029.47872663, 163977.84177699,
       163977.20952867, 164064.10961694, 164009.76651372, 164077.03057272,
       163978.63413918, 163963.71414486, 164001.73320958, 164047.46496268,
       163955.59276226, 163968.94320314, 163976.95504371, 164045.03171537,
       163977.3618452 , 164068.44396413, 163999.99624023, 163956.48301955,
       164054.93739929, 164045.15942542, 163963.25495428, 163969.80694632,
       164024.7205466 , 164025.59479828, 163957.50028766, 163992.27381669,
       164054.28816028, 164055.97524201, 163982.78901069, 163966.21856523,
       164065.47122149, 164000.79821007, 164064.69347472, 164054.03762935,
       164015.84230585, 163984.73559259, 164033.7356966 , 164071.72349924,
       163989.08229862, 164064.10109244, 163973.04514639, 164049.22392503,
       164004.00365269, 163961.01454817, 163991.56359251, 164051.73455786,
       163982.87826541, 164007.64891503, 163984.08852023, 163982.37332643])
```

KNN

In [95]:
```
from sklearn.neighbors import KNeighborsRegressor

knn=KNeighborsRegressor()
knn.fit(x_train, y_train)

pred_test_knn=knn.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_knn))
print('\n max error', max_error(y_test,pred_test_knn))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_knn, squared=Fal
print('\n MAPE',(abs((y_test-pred_test_knn)/y_test).mean())*100)
```

```
r2 score 0.7774662948231394

 max error 214293.80000000005

 Root mean squared error 37378.69076964831

 MAPE 15.111943905027786
```

In [96]:
```
# cross validation for knn model
cv_score=cross_val_score(knn, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_k
```

```
cross val score is 0.6931798133079993
Differnce between in r2_score and cross value score 0.08428648151514018
```

In [97]:
```
# predicting for test set we are given
prediction_knn=knn.predict(testn)
prediction_knn
```

Out[97]:
```
array([337300. , 180600. , 238629. , 188700. , 187100. , 156850. ,
       138791.8, 281480. , 245080. , 181560. , 134230. , 174380. ,
       114890. , 170780. , 243235.6, 126000. , 123531.6, 125280. ,
       154380. , 253756. , 127000. , 148850. , 137740. ,  86300. ,
       116980. , 139530. , 184698. , 151460. , 211000. ,  98700. ,
       106980. , 199620. , 225495.8, 129580. , 107640. , 192950. ,
       226150. , 109480. , 184560. , 149000. , 112280. , 253000. ,
       190588. , 203860. , 178475. , 116600. , 138940. , 120990. ,
       212180. , 310550. , 132494.2, 200700. , 103296. , 111320. ,
       206400. , 144680. , 123280. , 189178. , 142980. , 204750. ,
       136480. , 271000. , 147550. , 140800. , 182900. ,  95480. ,
       155100. , 207100. , 120711.6, 146100. , 285500. , 128750. ,
       121180. , 147900. , 172200. , 212300. , 316400. , 176220. ,
       265300. , 198560. , 239450. , 124325.2, 140400. , 137580. ,
```

```
193470. , 179457. , 112120. , 262300. , 146900. , 197042. ,
205763.8, 139100. , 109640. , 129560. , 206940. , 178220. ,
272780. , 230740. , 342800. , 162551.6, 247600. , 116750. ,
131980. , 192800. , 217980. , 162280. , 263100. , 158920. ,
186686.4, 170300. , 225100. , 190700. , 123896. , 203780. ,
132270. , 120630. , 143790. , 191780. , 172800. , 130880. ,
149491.8, 210350. , 179400. , 110300. , 135100. , 186180. ,
111880. , 144180. , 107260. , 116280. , 209580. , 235133.6,
175380. , 200300. , 135950. , 296900. , 332400. , 177580. ,
216075. , 137400. , 149300. , 286400. , 126880. , 216125. ,
179200. , 207400. , 140100. , 150380. , 121780. , 208800. ,
155890.4, 138390. , 302000. , 137150. , 131870. , 160500. ,
227100. , 155180. , 154650. , 211456. , 218615.8, 147080. ,
281800. , 133430. , 119080. , 297400. , 188100. , 235000. ,
115401.6, 184298. , 145700. , 141750. , 164400. , 203028. ,
132700. , 314500. , 107296. , 124170. , 141800. , 165480. ,
188200. , 182900. , 301900. , 169880. , 355790.8, 385500. ,
209760. ,  97876.6, 180200. , 192000. , 144700. , 250570. ,
225080. , 134080. , 156800. , 125760. , 181560. , 220480. ,
113580. , 185980. , 145380. , 215720. , 310590. , 194773. ,
116581.6, 158880. , 270300. , 151700. , 121080. , 179610.4,
147325. , 126760. , 129450. , 184740. , 137080. ,  97200. ,
143700. , 139200. , 129560. , 134270. , 185410. , 265580. ,
330080. , 111580. , 200400. , 363500. , 201715.8, 124800. ,
327823.4, 189200. , 108000. , 183400. , 109776. , 113820. ,
127825. , 182960. , 194300. , 234380. , 203061. , 194000. ,
256280. , 153350. , 282200. , 128700. , 128050. , 261880. ,
161500. , 282650. , 138200. , 123380. , 156665. , 194800. ,
122180. , 117500. , 151100. , 231500. ,  96776. , 302200. ,
147300. , 133900. , 230850. , 198520. , 135650. , 122726. ,
205660. , 175500. , 112440. , 141260. , 260080. , 218998. ,
137494.2, 123770. , 294800. , 149700. , 305497.8, 239300. ,
192996. , 127290.6, 175800. , 329150. , 128480. , 360322.2,
134550. , 215130. , 165600. , 122570. , 144000. , 236300. ,
123201.6, 158200. , 176080. , 144700. ])
```

Random Forest Regressor

In [98]:
```
from sklearn.ensemble import RandomForestRegressor

rfr=RandomForestRegressor()
rfr.fit(x_train,y_train)

pred_test_rfr=rfr.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_rfr))
print('\n max error', max_error(y_test,pred_test_rfr))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_rfr, squared=Fal
print('\n MAPE',(abs((y_test-pred_test_rfr)/y_test).mean())*100)
```

```
r2 score 0.721559509193331

 max error 347121.18

 Root mean squared error 41811.17586506602

 MAPE 15.564814835312543
```

In [99]:
```
# cross validation for rfr model
cv_score=cross_val_score(rfr, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_r
```

```
cross val score is 0.8412547924798951
Differnce between in r2_score and cross value score -0.11969528328656409
```

In [100…

```
# predicting for test set we are given
prediction_rfr=rfr.predict(testn)
prediction_rfr
```

Out[100]:

```
array([381825.91, 203065.13, 224245.29, 131739.44, 283518.02, 132767.14,
       160575.45, 285274.34, 224195.05, 181238.42, 116506.08, 137289.27,
       144263.7 , 208248.75, 268452.8 , 111280.98, 131137.52, 129073.5 ,
       183413.12, 204911.9 , 117876.21, 167723.65, 170435.61, 138577.98,
       126959.  , 141786.48, 182041.74, 152054.32, 210207.15, 100367.8 ,
       121596.81, 192258.81, 222402.27, 184150.22, 123474.  , 199249.39,
       230398.38, 115141.87, 173005.16, 133728.7 , 113432.58, 218559.35,
       224403.64, 195879.77, 211375.97, 114819.5 , 126252.92, 112745.04,
       251936.28, 404289.59, 133056.64, 197832.72, 101990.75,  81641.94,
       256063.78, 184033.57, 112336.92, 204092.09, 162212.87, 233970.39,
       132874.  , 249944.39, 132891.69, 184700.98, 245189.89, 119152.03,
       174172.01, 213009.1 , 161743.48, 135001.21, 235832.99, 165668.98,
       126700.49, 116825.37, 206578.48, 210211.64, 344609.61, 178272.92,
       264162.08, 183278.29, 232131.07, 143643.5 , 144253.65, 160833.79,
       225500.7 , 218056.42, 108906.75, 440745.03, 135044.09, 198141.35,
       229837.68, 148542.61, 113241.03, 139635.71, 181522.3 , 189069.98,
       262877.08, 196706.99, 361990.62, 150348.07, 232994.57, 156667.5 ,
       140185.12, 180918.  , 179691.95, 132978.33, 244457.04, 141438.53,
       209369.66, 184847.97, 214602.08, 186436.84, 122925.52, 283236.78,
       123105.54,  93024.77, 134528.18, 204714.24, 183765.24, 122986.04,
       133929.37, 219516.06, 182758.96, 134630.55, 120460.76, 175331.57,
       122893.5 , 179235.66,  87865.95,  95202.89, 174977.2 , 200187.86,
       148893.09, 189246.3 , 123392.66, 372983.49, 275744.91, 161758.66,
       234631.21, 133321.46, 133691.97, 494547.65,  87393.33, 451421.2 ,
       203709.44, 194125.24, 127724.56, 132126.61, 102091.37, 213043.39,
       188576.83, 128538.2 , 242754.53, 127144.21, 126696.58, 183967.51,
       204737.84, 167520.11, 129345.05, 189574.8 , 221584.32, 192173.47,
       217451.93, 130568.64, 127827.71, 321196.25, 191186.74, 225669.67,
       131759.  , 196196.53, 181220.55, 150539.5 , 183689.74, 262778.46,
       133661.5 , 338525.54, 102423.37, 125568.02, 165022.37, 208811.94,
       206589.27, 161281.24, 300927.71, 177280.13, 389769.01, 391926.97,
       211667.14, 127568.89, 168216.29, 182659.61, 122704.28, 219066.7 ,
       241050.83, 130820.26, 122676.83,  91535.55, 179835.12, 209751.98,
       109553.82, 166526.87, 187781.95, 132717.25, 347910.9 , 381776.01,
       122341.88, 176433.29, 248045.09, 140973.11, 105372.29, 185910.46,
       135078.83, 127683.55, 123268.3 , 190649.28, 141333.54,  89439.95,
       184340.55, 133595.83, 148522.26, 139951.56, 192712.82, 236686.2 ,
       384381.37, 131443.51, 182153.28, 403282.84, 211455.95, 102237.22,
       414077.85, 182181.51,  96731.21, 221060.  , 147529.59, 129206.6 ,
       118090.04, 201310.64, 187670.43, 216815.98, 191471.89, 208408.11,
       229945.87, 149220.06, 229285.08, 109920.83, 133438.  , 227118.42,
       226691.8 , 344138.37, 135693.08, 134414.62, 165957.37, 218896.25,
       109475.45, 116439.09, 140310.08, 198834.45,  88609.05, 401752.59,
       173729.29, 137269.29, 232672.3 , 216123.33, 121424.13, 118903.22,
       187346.73, 200005.13, 122197.74, 168847.42, 237174.5 , 217122.97,
       128111.54, 119812.04, 383721.21, 182005.94, 286863.31, 201190.01,
       187238.25, 136744.57, 183590.52, 351166.19, 163181.84, 376894.09,
       129112.87, 282340.85, 197386.75, 119153.15, 165689.03, 240779.83,
       176953.24, 178115.71, 126551.11, 127385.75])
```

Regularisation

In [101…

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge, Lasso
```

Ridge

In [102…
```
rg=Ridge()
rg.fit(x_train,y_train)

pred_test_rg=rg.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_rg))
print('\n max error', max_error(y_test,pred_test_rg))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_rg, squared=Fals
print('\n MAPE',(abs((y_test-pred_test_rg)/y_test).mean())*100)
```

```
r2 score 0.8187686141403463

 max error 328063.05790235195

 Root mean squared error 33732.06215642944

 MAPE 13.341463185508932
```

In [103…
```
# cross validation for rg
cv_score=cross_val_score(rg, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_r
```

```
cross val score is 0.7929581715329479
Differnce between in r2_score and cross value score 0.0258104426073984
```

In [104…
```
# predicting for test set we are given
prediction_rg=rg.predict(testn)
prediction_rg
```

Out[104]:
```
array([308397.87764544, 213763.53873147, 270036.54484664, 180037.52864129,
       227447.67912764, 116530.05749034, 147108.48785252, 242188.85466778,
       241887.73067821, 170997.53399631, 107268.8911182 , 223060.69568033,
        83477.10552839, 348911.31157249, 278310.51706011,  76184.7954334 ,
        80215.85136461,  88473.70725723, 230562.76571869, 246109.71508283,
        88846.58413559, 223432.46593357, 158473.64624628,  82362.9870234 ,
        96847.82283335, 137210.16403596, 203487.14334961, 140680.58865115,
       205319.12236665, 123311.21109841,  -5820.64625039, 181519.57534212,
       274340.74913113, 211703.18827898, 123040.69179798, 206102.66352464,
       277958.80751252, 107193.91299925, 178886.09153144, 177882.85381296,
        68008.86453793, 218913.2594733 , 247440.73535751, 190166.49746978,
       203295.99572229, 123407.46166631, 112265.80449463,  59202.16470866,
       255409.34914074, 290330.31445485, 134975.854682  , 182387.62432695,
        54720.27666351,  87188.15970486, 258673.20288076, 197047.80895084,
        93400.45988264, 226973.4361291 ,  73031.09802901, 234987.91736022,
       130091.1165981 , 262970.81222292, 165534.09611779, 229550.81925737,
       263494.4439799 , 116009.10922275, 196813.8035212 , 191538.17578218,
       119815.18158187, 146642.81327509, 261254.83389969, 163595.6331885 ,
       156461.17059328,  65935.11771457, 240120.03534984, 206349.32546407,
       319557.29438909, 231815.35563074, 306790.36533236, 257539.0019374 ,
       270306.15549011, 139090.76744357, 156483.3930577 , 158817.81048712,
       260787.68813988, 185287.47592394,  79477.92726715, 285283.19728686,
       162257.87149976, 221269.75980141, 217592.58852348, 144030.02881005,
       127488.66611134, 146983.09693031, 198229.73848482, 212842.15513434,
       329500.64257425, 228451.9532058 , 275083.57831235, 173856.18332062,
       276645.64337115, 185167.32717362, 170861.09665791, 150588.65560721,
       174037.00070954, 115733.53075635, 276579.11239176, 119195.31900793,
       186756.66126832, 148081.36019206, 231180.17196514, 222833.71066904,
       123500.26493294, 307280.83041352, 125059.34926935,  36771.48664586,
       102788.32349832, 210035.7090885 , 198393.18641347, 112069.63055421,
       162898.47545766, 259206.4831722 , 152037.72775529, 118354.55477162,
```

```
100336.638405  , 201423.49692665, 105269.85585982, 239079.69637374,
-35104.4964728 ,  56672.45933307, 218145.12339081, 210129.78798838,
198347.35286552, 217742.42191934, 118547.97798264, 258290.00657242,
300815.57423385, 119233.84341759, 255932.65414624, 156283.95019673,
158930.92540551, 345558.40425876,  35501.07109195, 308261.40307341,
233604.19146835, 135470.43147302, 140337.66898732,  39341.46615072,
 24340.29078008, 243595.63815098, 167464.52649222, 164328.14099336,
237129.39733032,  99178.12805304, 139298.06006407, 184217.17646243,
237699.1746404 , 215115.16628093, 144475.06248339, 227707.79833395,
257749.85371325, 191720.8483072 , 246721.60236849, 135570.99970955,
 74403.8603556 , 239065.99693578, 241468.32396159, 229411.35194454,
122567.94922312, 174406.34134479, 149609.41974843, 116384.7423896 ,
146813.61076851, 224457.16859238, 107055.16523069, 303497.34320079,
 75710.492097  ,  94841.48090634, 176215.92846807, 166784.68747862,
189342.91175016, 176243.03150569, 292259.77742034, 158411.96033017,
327655.75434899, 316331.33679233, 229497.85279907, 144488.13313007,
145697.77177506, 127097.31641657, 139887.4201672 , 250789.33316665,
301342.08632557,  95161.85816405, 168979.21272778,  61543.15563364,
157999.54846018, 198623.00307957, 113986.91045895, 135269.56339799,
231775.04583409, 180735.42194427, 310162.41361836, 275140.93417066,
128714.69361687, 203415.8598671 , 276720.27307971, 185679.57304557,
 46564.6115458 , 224704.63939612, 151169.79416202, 125107.76063272,
104207.62301124, 228725.01826304,  82980.05578936, 113246.2852513 ,
182296.78227361, 107150.84687064, 147227.76702867, 227421.06476728,
207076.56121208, 222656.27744216, 279261.00320886, 136783.09377934,
207870.24991041, 333156.16116041, 173741.64179603, 109138.12170325,
314749.77309479, 182222.10403277,  65006.56147325, 226307.05652127,
 98947.9083023 , 125174.97943392, 111207.3033725 , 173823.36503792,
177323.25603973, 257664.92760103, 212570.4397567 , 176824.63254499,
262354.79341969, 164844.51984323, 263349.7200613 , 110862.17369129,
139953.11626798, 209181.51488036, 228935.18426588, 348146.04255989,
191546.36839905, 110716.56184581, 254934.28983757, 198424.76760611,
 80036.9239678 , 107577.81066482, 103471.57892322, 191688.62051284,
 39320.39743516, 292605.17455811, 208916.9500996 , 119486.5341549 ,
257839.18013964, 235913.92697911,  80850.59276793, 126995.86363292,
165240.56034512, 226694.60003356, 101854.59654682, 211724.50061844,
233404.28733844, 218656.41724128,  82152.25899393,  72499.40865433,
304553.43883899, 176235.7949626 , 292069.02204908, 244717.77376774,
216396.71511289, 111556.0712565 , 224817.64002087, 327260.90644146,
137501.027484  , 320777.42080616, 148939.58654549, 280732.91750688,
208079.67628875,  91221.15568808, 158257.16190637, 260126.00880745,
131188.85602096, 117595.91012716, 172008.97709516, 158886.09784825])
```

In [ ]:

Parameter Tuning

In [105…

```
parameters={'alpha':[.001,.01,1,10], 'solver':['auto', 'svd', 'saga'],'random_state':lis

clf=GridSearchCV(Ridge(),parameters)
clf.fit(x_train, y_train)
print(clf.best_params_)
```

```
{'alpha': 10, 'random_state': 3, 'solver': 'saga'}
```

In [106…

```
rg_reg=Ridge(alpha=10,random_state=3, solver='saga')

rg_reg.fit(x_train, y_train)

pred_test_rg_reg=rg_reg.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_rg_reg))
print('\n max error', max_error(y_test,pred_test_rg_reg))
```

```
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_rg_reg, squared=
print('\n MAPE',(abs((y_test-pred_test_rg_reg)/y_test).mean())*100)
```

```
r2 score 0.7992347351862548

 max error 359359.99359577755

 Root mean squared error 35503.4432209391

 MAPE 12.771244541124732
```

In [107…
```
# cross validation for rg_reg
cv_score=cross_val_score(rg_reg, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_r
```

```
cross val score is 0.7848194836293392
Differnce between in r2_score and cross value score 0.014415251556915587
```

In [108…
```
# predicting for test set we are given
prediction_rg_reg=rg_reg.predict(testn)
prediction_rg_reg
```

Out[108]:
```
array([302365.23564917, 213358.96567413, 265320.14699136, 172083.08501628,
       231614.97037543, 119335.06230114, 159107.91025401, 250638.91947237,
       252597.07492827, 165743.54595507, 101200.10654624, 207921.68318535,
        83144.13308179, 312196.70373082, 274300.22783701,  93274.21718123,
        95545.74318469,  97139.28931439, 224844.60339325, 248090.9441378 ,
        97479.07630326, 222231.74761375, 166141.52834166,  79538.57279913,
       104302.18637104, 139912.3036354 , 199009.30002087, 140229.70773103,
       202326.40712546, 119259.52799914,  18999.63013317, 186047.71772513,
       265691.55861147, 204388.61028121, 118490.99627751, 207433.51309844,
       271021.63623955, 119259.12019551, 175618.79060749, 159275.33013454,
        77518.44775436, 228562.8160667 , 246810.08577043, 185474.57522515,
       188444.67788689, 123095.6486564 , 125383.88887455,  66228.01987717,
       250730.49237176, 286826.68383828, 123092.92119395, 197380.35078531,
        67451.20647652,  70584.39174347, 257308.86244506, 194908.91107204,
        95403.92683767, 221590.13972074,  88065.85814662, 238560.35659796,
       130910.06230549, 251053.58467896, 164693.88329411, 225340.64901221,
       259822.74175463, 112375.75611655, 184021.95309678, 200052.28330562,
       120131.218845  , 158887.36779974, 258309.22252647, 161959.77844956,
       153287.57419788,  69112.03781985, 234708.21865264, 216766.71284969,
       307403.12453683, 233422.79591605, 296011.36717893, 241001.9866733 ,
       270168.95594657, 116805.94226133, 154408.29206836, 156356.15939179,
       259804.54312826, 194953.58201317,  83092.63319866, 279352.79485066,
       142728.38782148, 223074.50250314, 212202.40615254, 140373.63722858,
       124062.98659371, 141002.62311708, 200003.9399395 , 201500.25705185,
       313956.93723214, 228427.56224554, 282368.52324756, 167594.42908634,
       279197.57995617, 172143.98257833, 159919.05951254, 151731.73243638,
       178374.44040842, 125587.14005545, 267763.24406225, 121849.404741  ,
       195084.0832681 , 175229.2032805 , 234768.52851298, 217206.81095874,
       146563.80394726, 292207.15784769, 124722.04826044,  44762.24653881,
       113636.92719283, 206481.2331646 , 184481.50654405, 101420.85147014,
       150599.18102396, 256745.12833372, 160868.18162938, 126387.26568906,
       106209.08322135, 200710.25402786, 106490.62033003, 229191.27844432,
        10345.74693383,  59355.11124687, 217531.60756454, 211246.24758859,
       187333.01173216, 202312.92780045, 133961.50514828, 267500.0293625 ,
       294075.68412537, 142189.7337295 , 260178.00121908, 152095.08767631,
       153532.8236427 , 339472.96747724,  44833.59134406, 306776.66101414,
       233104.57663937, 152477.9464581 , 147732.6372286 ,  59553.74419718,
        43293.61275608, 238262.84629206, 175870.55049873, 156941.05037109,
       236607.2836967 , 103129.80432477, 132071.12906069, 189395.02151441,
       227447.00604531, 204315.97209402, 151719.73663285, 235216.61829609,
       253589.0541198 , 183039.5966568 , 252089.0727727 , 131537.89935485,
```

```
 85341.74605417, 250092.89133721, 232941.53325031, 234943.42133648,
123350.12912849, 176208.39228993, 154316.22780388, 123907.78890128,
151250.34854909, 232898.48922006, 115504.18603974, 311814.81867406,
 73811.73789673,  92607.21579702, 177264.31693894, 159748.6042936 ,
190970.58941729, 186065.60204121, 292626.82850276, 157108.85789391,
321588.5834739 , 314815.24775742, 225102.67688175, 143305.08778528,
146980.31756994, 130886.58209735, 142412.4610585 , 253501.04670619,
293561.82849792,  96691.51171513, 160355.77585617,  58536.61716676,
160224.5071899 , 205591.14344396, 110475.11130683, 143749.112128  ,
234983.3681112 , 172818.10297281, 316647.40220472, 280690.39352518,
129363.33246075, 198136.47573018, 262359.75448815, 179923.14220865,
 41854.77852172, 209407.01747833, 152051.48553398, 132314.83671245,
116388.51034832, 224993.00190539,  90189.35173525, 118508.95744655,
187862.41110151, 109036.38436879, 146882.22249267, 215484.67907583,
207763.51163243, 232781.77304933, 278923.26245816, 136070.19468772,
201673.92276939, 329972.0632862 , 178530.21807945,  93490.53471086,
307109.44606351, 181336.31368895,  65776.8115684 , 217794.30436974,
 91840.72926388, 130431.21124403, 116745.2187986 , 183612.502096  ,
177718.29404351, 259191.60051201, 213397.41618826, 177510.18712992,
263360.03310552, 168002.12981096, 276946.57406433,  84346.65954835,
144800.64978056, 218188.90415251, 220342.31682185, 334592.22362847,
177475.82168552, 105975.23146635, 248607.71603189, 195007.42326378,
 93440.57966699, 101596.70437062, 111142.26742848, 193508.78458255,
 41236.73384158, 299636.21402183, 208454.68373379, 119259.24384834,
248959.62713791, 236346.01344621,  88269.27274772, 125537.16727876,
168446.49562397, 231199.23383852,  94573.55905555, 202523.05705937,
238691.34002697, 216267.01003535,  83542.66126127, 100584.52064174,
300047.8871774 , 173449.58472561, 281083.16146579, 245170.66042037,
217349.48524104, 106555.96380146, 221487.60972386, 320182.58948635,
143195.21050882, 318433.20743817, 138300.5611631 , 271368.66484218,
202341.18422108,  99493.058178  , 168141.47356084, 256361.68749875,
131614.83883398, 125252.21431171, 173358.54214601, 158373.22611816])
```

In [ ]:

Lasso

In [109…

```
ls=Ridge()
ls.fit(x_train,y_train)

pred_test_ls=ls.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_ls))
print('\n max error', max_error(y_test,pred_test_ls))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_ls, squared=Fals
print('\n MAPE',(abs((y_test-pred_test_ls)/y_test).mean())*100)
```

```
r2 score 0.8187686141403463

 max error 328063.05790235195

 Root mean squared error 33732.06215642944

 MAPE 13.341463185508932
```

In [110…

```
# cross validation for lasso
cv_score=cross_val_score(ls, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_l
```

```
cross val score is 0.7929581715329479
Differnce between in r2_score and cross value score 0.0258104426073984
```

In [111…

```
# predicting for test set we are given
prediction_ls=ls.predict(testn)
prediction_ls
```

Out[111]:

```
array([308397.87764544, 213763.53873147, 270036.54484664, 180037.52864129,
       227447.67912764, 116530.05749034, 147108.48785252, 242188.85466778,
       241887.73067821, 170997.53399631, 107268.8911182 , 223060.69568033,
        83477.10552839, 348911.31157249, 278310.51706011,  76184.7954334 ,
        80215.85136461,  88473.70725723, 230562.76571869, 246109.71508283,
        88846.58413559, 223432.46593357, 158473.64624628,  82362.9870234 ,
        96847.82283335, 137210.16403596, 203487.14334961, 140680.58865115,
       205319.12236665, 123311.21109841,  -5820.64625039, 181519.57534212,
       274340.74913113, 211703.18827898, 123040.69179798, 206102.66352464,
       277958.80751252, 107193.91299925, 178886.09153144, 177882.85381296,
        68008.86453793, 218913.2594733 , 247440.73535751, 190166.49746978,
       203295.99572229, 123407.46166631, 112265.80449463,  59202.16470866,
       255409.34914074, 290330.31445485, 134975.854682  , 182387.62432695,
        54720.27666351,  87188.15970486, 258673.20288076, 197047.80895084,
        93400.45988264, 226973.4361291 ,  73031.09802901, 234987.91736022,
       130091.1165981 , 262970.81222292, 165534.09611779, 229550.81925737,
       263494.4439799 , 116009.10922275, 196813.8035212 , 191538.17578218,
       119815.18158187, 146642.81327509, 261254.83389969, 163595.6331885 ,
       156461.17059328,  65935.11771457, 240120.03534984, 206349.32546407,
       319557.29438909, 231815.35563074, 306790.36533236, 257539.0019374 ,
       270306.15549011, 139090.76744357, 156483.3930577 , 158817.81048712,
       260787.68813988, 185287.47592394,  79477.92726715, 285283.19728686,
       162257.87149976, 221269.75980141, 217592.58852348, 144030.02881005,
       127488.66611134, 146983.09693031, 198229.73848482, 212842.15513434,
       329500.64257425, 228451.9532058 , 275083.57831235, 173856.18332062,
       276645.64337115, 185167.32717362, 170861.09665791, 150588.65560721,
       174037.00070954, 115733.53075635, 276579.11239176, 119195.31900793,
       186756.66126832, 148081.36019206, 231180.17196514, 222833.71066904,
       123500.26493294, 307280.83041352, 125059.34926935,  36771.48664586,
       102788.32349832, 210035.7090885 , 198393.18641347, 112069.63055421,
       162898.47545766, 259206.4831722 , 152037.72775529, 118354.55477162,
       100336.638405  , 201423.49692665, 105269.85585982, 239079.69637374,
       -35104.4964728 ,  56672.45933307, 218145.12339081, 210129.78798838,
       198347.35286552, 217742.42191934, 118547.97798264, 258290.00657242,
       300815.57423385, 119233.84341759, 255932.65414624, 156283.95019673,
       158930.92540551, 345558.40425876,  35501.07109195, 308261.40307341,
       233604.19146835, 135470.43147302, 140337.66898732,  39341.46615072,
        24340.29078008, 243595.63815098, 167464.52649222, 164328.14099336,
       237129.39733032,  99178.12805304, 139298.06006407, 184217.17646243,
       237699.1746404 , 215115.16628093, 144475.06248339, 227707.79833395,
       257749.85371325, 191720.8483072 , 246721.60236849, 135570.99970955,
        74403.8603556 , 239065.99693578, 241468.32396159, 229411.35194454,
       122567.94922312, 174406.34134479, 149609.41974843, 116384.7423896 ,
       146813.61076851, 224457.16859238, 107055.16523069, 303497.34320079,
        75710.492097  ,  94841.48090634, 176215.92846807, 166784.68747862,
       189342.91175016, 176243.03150569, 292259.77742034, 158411.96033017,
       327655.75434899, 316331.33679233, 229497.85279907, 144488.13313007,
       145697.77177506, 127097.31641657, 139887.4201672 , 250789.33316665,
       301342.08632557,  95161.85816405, 168979.21272778,  61543.15563364,
       157999.54846018, 198623.00307957, 113986.91045895, 135269.56339799,
       231775.04583409, 180735.42194427, 310162.41361836, 275140.93417066,
       128714.69361687, 203415.8598671 , 276720.27307971, 185679.57304557,
        46564.6115458 , 224704.63939612, 151169.79416202, 125107.76063272,
       104207.62301124, 228725.01826304,  82980.05578936, 113246.2852513 ,
       182296.78227361, 107150.84687064, 147227.76702867, 227421.06476728,
       207076.56121208, 222656.27744216, 279261.00320886, 136783.09377934,
       207870.24991041, 333156.16116041, 173741.64179603, 109138.12170325,
       314749.77309479, 182222.10403277,  65006.56147325, 226307.05652127,
        98947.9083023 , 125174.97943392, 111207.3033725 , 173823.36503792,
       177323.25603973, 257664.92760103, 212570.4397567 , 176824.63254499,
       262354.79341969, 164844.51984323, 263349.7200613 , 110862.17369129,
```

```
       139953.11626798, 209181.51488036, 228935.18426588, 348146.04255989,
       191546.36839905, 110716.56184581, 254934.28983757, 198424.76760611,
        80036.9239678 , 107577.81066482, 103471.57892322, 191688.62051284,
        39320.39743516, 292605.17455811, 208916.9500996 , 119486.5341549 ,
       257839.18013964, 235913.92697911,  80850.59276793, 126995.86363292,
       165240.56034512, 226694.60003356, 101854.59654682, 211724.50061844,
       233404.28733844, 218656.41724128,  82152.25899393,  72499.40865433,
       304553.43883899, 176235.7949626 , 292069.02204908, 244717.77376774,
       216396.71511289, 111556.0712565 , 224817.64002087, 327260.90644146,
       137501.027484  , 320777.42080616, 148939.58654549, 280732.91750688,
       208079.67628875,  91221.15568808, 158257.16190637, 260126.00880745,
       131188.85602096, 117595.91012716, 172008.97709516, 158886.09784825])
```

Parameter Tuning

In [112…

```
parameters={'alpha':[.001,.01,1,10],'random_state':list(range(0,10))}

clf=GridSearchCV(Lasso(),parameters)
clf.fit(x_train, y_train)
print(clf.best_params_)
```

```
{'alpha': 10, 'random_state': 0}
```

In [113…

```
ls_reg=Lasso(alpha=10,random_state=0)

ls_reg.fit(x_train, y_train)

pred_test_ls_reg=ls_reg.predict(x_test)

print('r2 score', r2_score(y_test,pred_test_ls_reg))
print('\n max error', max_error(y_test,pred_test_ls_reg))
print('\n Root mean squared error', mean_squared_error(y_test,pred_test_ls_reg, squared=
print('\n MAPE',(abs((y_test-pred_test_ls_reg)/y_test).mean())*100)
```

```
r2 score 0.818494155481501

 max error 317739.2415031061

 Root mean squared error 33757.59458290023

 MAPE 13.836524738387682
```

In [114…

```
# cross validation for lasso
cv_score=cross_val_score(ls_reg, x,y, cv=13)
cv_mean=cv_score.mean()

print('cross val score is', cv_mean)
print('Differnce between in r2_score and cross value score', r2_score(y_test,pred_test_l
```

```
cross val score is 0.7904603600580296
Differnce between in r2_score and cross value score 0.028033795423471397
```

In [115…

```
# predicting for test set we are given
prediction_ls_reg=ls_reg.predict(testn)
prediction_ls_reg
```

Out[115]:

```
array([310691.87162749, 212932.92929548, 268686.13251879, 180478.97498639,
       228344.68590229, 116170.09341347, 143896.06359528, 238643.09179773,
       238733.21926297, 172301.83981501, 109138.85410111, 226154.84507775,
        83869.04594092, 354026.24457639, 279880.83082725,  65794.78906157,
        73411.11437552,  85992.29123845, 228066.84913675, 244113.38021632,
        86084.82046089, 225305.78824248, 155144.01203331,  79713.58090865,
        92826.24958517, 134284.34377548, 205180.07613903, 144201.3056715 ,
       206923.86133722, 122360.84883079,  -9614.81915243, 182261.23035293,
       275602.48118468, 218333.39547986, 127513.07588394, 207299.08269979,
```

281076.63605797, 105657.31357946, 177693.23782328, 189752.56676461,
 68767.97713644, 216471.16041932, 248857.57656507, 193864.87256137,
209209.8469392 , 124152.44177313, 101333.25139864, 55091.75860846,
255768.05515596, 290121.41923337, 139466.76854237, 179263.04858347,
 47118.73462823, 98552.95519898, 259985.37959829, 196973.20436896,
 96724.41337754, 229501.83216365, 72896.82098385, 235103.91035021,
131364.63321969, 269321.45897653, 162417.15721106, 231289.30330791,
265486.96902834, 117284.25795516, 203274.62195156, 192163.46468363,
120505.71694133, 142348.95953557, 259800.21028466, 168810.47118253,
154949.79537529, 68770.16661381, 239530.65013018, 203054.76782554,
323514.82002776, 230716.16646624, 308717.62660741, 260319.45626283,
269811.52914236, 148970.98311377, 154281.59122084, 159065.3810835 ,
260416.07437918, 181445.59223659, 80004.02668223, 286729.31138229,
171516.29167417, 219804.84269829, 220581.38341692, 143885.40186098,
125668.68568466, 147110.793828 , 197184.40274998, 214266.74139022,
333158.23091718, 227756.91877881, 271314.15089567, 175044.01610451,
273797.57839813, 189141.66434347, 174781.70953647, 151997.2122201 ,
172675.91991392, 112705.95403942, 279425.90828244, 119377.91541204,
185835.17161279, 126115.20138152, 232597.91466314, 225424.75589124,
110952.3881214 , 310787.70835889, 125639.56868196, 33439.8429864 ,
 98372.52368724, 212557.88744397, 201834.46955929, 118793.16589554,
166414.2470617 , 257786.93925549, 149767.08093579, 114028.34601931,
 98413.44067829, 200911.6129958 , 106048.6023852 , 243624.57911215,
-52866.01031758, 55102.50566792, 218519.04604757, 210632.72571371,
201828.97301275, 222408.60610165, 114043.83190001, 256732.55188572,
301573.77564626, 111447.52424593, 252466.36170219, 157151.04658717,
160653.41028381, 339578.53333067, 36275.26061331, 309894.83583732,
232298.38069306, 131261.63574213, 133119.75452131, 34185.91396953,
 23994.06190528, 245693.90445397, 164419.01006692, 165124.92208328,
239783.35250822, 98201.16918188, 139049.24438985, 183707.75658004,
240618.81764046, 218557.50490198, 142185.53454214, 223094.23026514,
260420.54071608, 195094.39665479, 244223.07314535, 135136.59039179,
 70101.8195179 , 235697.31679734, 243928.52449855, 228900.69704575,
124283.07914781, 172660.03976509, 149695.35884033, 115495.09225942,
145649.63864053, 221831.20755413, 105849.90292444, 302309.25327506,
 78542.4377795 , 95750.20659714, 176840.79223818, 171703.69756064,
188843.61658083, 171712.689769 , 293113.85071976, 156557.28336568,
330502.29186014, 315669.7811464 , 230024.45694339, 147297.13024747,
146286.18420562, 127037.65891817, 140819.36522269, 249474.96746437,
302806.86169349, 94994.05691624, 169997.70050523, 60578.19993753,
157200.28717054, 198761.84580667, 113439.22939632, 133694.99962374,
230128.2882669 , 182052.97672   , 307292.73176311, 269684.128141 ,
126908.19174128, 205422.89230985, 281731.38476129, 185486.96881616,
 55079.82502914, 232360.73908313, 151147.00292491, 122855.89944236,
 99458.18978997, 230664.73177016, 81711.56651647, 112395.69367512,
181284.75845264, 103972.90769968, 146830.03537453, 229650.55321275,
207960.56283875, 220370.05085881, 279885.64688875, 135167.67529864,
208953.063065 , 332127.70868471, 173474.47090413, 112000.660636 ,
317680.65027687, 181108.25082711, 67220.56952803, 227096.10462011,
103471.54378878, 124389.67103698, 105981.21482194, 170556.50284422,
179021.76243657, 255907.2563328 , 211392.47743094, 177912.18043843,
260559.78057431, 162943.4091949 , 255408.19318697, 122399.14561225,
136573.17445439, 209190.24256694, 234153.44207527, 347854.37178355,
196931.41877945, 112836.70411551, 255618.93147596, 199108.40556184,
 77683.89753268, 105176.98704987, 100731.87880547, 190704.25650215,
 43708.80914415, 290921.84338606, 207650.01769838, 120834.10154171,
263048.9001238 , 235102.5994434 , 81315.54235609, 130549.33051799,
163932.16097553, 223328.34450766, 102888.17986094, 214020.36555666,
232010.18662782, 219805.32272953, 87362.12703771, 58328.26846395,
305122.41421935, 179520.76041765, 296954.14511184, 245904.97248319,
216753.64672037, 109672.23404126, 226771.32454767, 329217.88797986,
134632.24651527, 323872.41539515, 153624.61387076, 282256.86433098,
209691.099689 , 88406.79379755, 154380.47463239, 261777.9519678 ,
131114.08399782, 119444.15448386, 174020.2094007 , 159097.94685741])

Model Saving

Since rg_rg model (Tuned Ridge) is providing us the best accuracy(minimum difference between accuracy and validation score with high accuracy). So we will save the rg_reg model.

In [121…
```
import pickle
filename='House_Price.pkl'
pickle.dump(rg_reg, open(filename,'wb'))
```

Conclusion

In [124…
```
loaded_model=pickle.load(open('House_Price.pkl','rb'))


original=np.array(y_test)
predicted=np.array(loaded_model.predict(x_test))

r2_score(y_test,loaded_model.predict(x_test))
```

Out[124]:
```
0.7992347351862548
```

In [125…
```
df=pd.DataFrame({'Original':original,'Predicted':predicted})
df
```

Out[125]:

	Original	Predicted
0	133000	128483.493874
1	108000	117524.814672
2	220000	255821.744157
3	200000	251082.430709
4	253293	312227.869369
5	134900	124087.443747
6	144000	144201.695247
7	299800	285599.531664
8	109900	129593.547791
9	168000	175279.528205
10	102000	97042.227381
11	279500	257864.965743
12	225000	243831.407643
13	233230	254464.094609
14	107500	114625.770908
15	145000	136930.169969
16	84500	108816.353013
17	160000	158953.622226
18	102776	95937.417607

19	215000	238321.607652
20	165000	170517.118914
21	181134	218029.949070
22	125000	94733.295889
23	92900	77626.091028
24	271000	255460.178892
25	127500	150035.595967
26	124500	131267.259908
27	107500	84826.638548
28	181000	193771.477930
29	143000	140482.196421
30	141000	150640.599931
31	255500	281394.629278
32	287000	290621.813650
33	134450	135564.078915
34	128000	132468.268519
35	163000	194735.652376
36	269790	257052.107551
37	275000	289797.052905
38	165500	189092.616926
39	185000	212109.444125
40	108000	96282.360377
41	180000	182942.974451
42	188500	224016.733800
43	171000	192478.130094
44	287090	271924.233098
45	293077	293697.443089
46	274900	314821.016576
47	114500	105899.445769
48	118858	104211.322474
49	108000	108593.459618
50	117500	157364.038304
51	135500	132881.095795
52	169500	191708.209252
53	132500	125945.663645
54	310000	291019.590772
55	175900	213507.812064
56	260000	245791.945753

57	139900	111494.606581
58	109900	82570.948553
59	155835	216305.034891
60	140000	156356.002600
61	325300	308173.852603
62	110000	104740.274964
63	328000	257783.293146
64	246578	242993.003474
65	214900	246606.062293
66	189000	216276.682105
67	151000	151495.491328
68	175000	182880.495332
69	163500	135187.167708
70	194000	210246.890747
71	178740	188426.626546
72	187500	185776.660292
73	110000	138107.394748
74	257500	241759.103868
75	240000	199378.393779
76	254000	230139.316132
77	176000	160352.151945
78	216000	216323.038233
79	149500	170653.666910
80	66500	98016.753123
81	283463	305078.855393
82	55993	135281.706723
83	160000	164359.627040
84	315500	323096.561538
85	213250	239915.907487
86	137000	121979.035886
87	755000	395640.006404
88	140000	163598.710518
89	159000	206988.685864
90	191000	208913.436450
91	176500	214092.698396
92	131000	144677.172533
93	124900	108879.719092

94	325000	290619.024241
95	213000	241353.781968
96	378500	320692.165118
97	402000	304209.011509
98	106250	72637.941154
99	148000	150518.577886
100	193000	203505.927188
101	239000	196681.433297
102	147000	134331.621178
103	108500	116516.599262
104	297000	299745.633413
105	126000	118570.754247
106	149350	143481.479540
107	222500	255264.523344
108	152000	157927.598232
109	156000	173163.440454
110	117500	99451.920400
111	244000	252497.656208
112	168000	214802.144933
113	250000	297493.814557
114	205000	230778.308856
115	133000	129331.932133
116	192500	219181.970941
117	135960	148146.227760
118	172500	211826.506920
119	225000	193162.334772
120	350000	327773.860835
121	126175	119116.271415
122	112000	149366.838993
123	67000	86902.274310
124	189000	165969.870452
125	193000	187453.492675
126	158000	182333.355441
127	187100	193817.256583
128	176432	192217.360306
129	133900	118827.180864
130	192000	208368.236944
131	250580	211250.220564

132	223000	235041.403320
133	264561	264293.748364
134	242000	247149.516829
135	200100	173328.270401
136	125500	126206.929771
137	281213	281968.401856
138	80000	61880.426989
139	301000	256905.666348
140	80000	67548.167258
141	190000	200035.395279
142	101800	91804.139076
143	140000	132650.954319
144	133500	139142.121625
145	141000	154886.295684
146	260000	291160.300605
147	81000	76213.260861
148	107000	108804.722920
149	141000	97031.723740
150	145000	126016.442559
151	167000	196750.075114
152	157500	188087.870171
153	148000	165127.058491
154	466500	368146.096392
155	132500	130384.280056
156	155000	183647.696346
157	194700	215449.721639
158	230000	182401.893126
159	115000	89184.445238
160	135000	143335.359346
161	140000	133220.613228
162	214500	161921.935838
163	159950	146953.484854
164	148000	123006.005601
165	82500	93306.791807
166	248000	250378.865944
167	274300	252877.643830
168	141500	117490.239016

169	385000	296375.520836
170	173000	173133.615356
171	129000	139079.675345
172	131000	130325.315159
173	79500	70076.578328
174	281000	291478.721620
175	139000	122704.411921
176	140000	138355.427268
177	162900	204536.665057
178	116050	71783.399227
179	185500	210986.026813
180	140000	146911.479175
181	282922	269939.991731
182	119000	121635.575076
183	204900	208253.539091
184	234000	200213.608050
185	192000	229824.128086
186	157000	139824.475947
187	137000	173521.037322
188	212000	220329.067418
189	170000	168801.850754
190	214000	228180.159059
191	149900	170898.048967
192	120000	129030.749964
193	216500	244008.867223
194	215000	206074.925903
195	161500	163523.152125
196	106500	84061.755790
197	188000	196662.096180
198	135000	128792.130003
199	178000	211309.825746
200	269500	273836.144992
201	118500	107417.492774
202	155000	179822.031862
203	39300	4074.907491
204	306000	283591.248138
205	290000	302067.305178
206	335000	282584.989947

207	131400	154830.606016
208	251000	244370.276732
209	166000	130122.256239
210	110000	119333.148868
211	100000	87830.646727
212	240000	212101.024019
213	120500	136218.052738
214	221000	199405.631633
215	138000	129184.591202
216	173000	195155.435049
217	160000	204927.801190
218	325000	271741.314681
219	158500	145225.346430
220	97000	94846.453851
221	159434	105177.849113
222	197500	194168.612507
223	52500	71813.140471
224	130000	167120.724494
225	125000	161619.287533
226	140000	180875.521181
227	190000	214414.004928
228	215000	248622.775888
229	87000	104944.928458
230	255000	266892.597902
231	153900	169906.490078
232	211000	225692.528764
233	119200	127429.198044

Lets make data frame for results on test data for all the model

In [117…

```
df=pd.DataFrame({'LinearRegression':prediction_lnr,
                 'DecisionTreeRegressor':prediction_dtr,
                 'SVR':prediction_svr,
                 'KNN':prediction_knn,
                 'Random Forest Regressor':prediction_rfr,
                'Ridge':prediction_rg,
                'tuned_Ridge':prediction_rg_reg,
                'Lasso':prediction_ls,
                'tuned_Lasso':prediction_ls_reg})
df
```

Out[117]:

	LinearRegression	DecisionTreeRegressor	SVR	KNN	Random Forest Regressor	Ridge	tuned_Ridge
0	311010.949360	325624.0	164076.041036	337300.0	381825.91	308397.877645	302365.235649
1	212571.817195	200000.0	164019.431190	180600.0	203065.13	213763.538731	213358.965674
2	268437.087511	190000.0	164051.248022	238629.0	224245.29	270036.544847	265320.146991
3	180663.226474	154900.0	163984.674775	188700.0	131739.44	180037.528641	172083.085016
4	228029.749913	219210.0	164071.806833	187100.0	283518.02	227447.679128	231614.970375
5	115512.154516	160000.0	163972.614677	156850.0	132767.14	116530.057490	119335.062301
6	143259.631555	141000.0	164008.000349	138791.8	160575.45	147108.487853	159107.910254
7	237889.748619	266000.0	164058.002650	281480.0	285274.34	242188.854668	250638.919472
8	237796.917067	228000.0	164036.552341	245080.0	224195.05	241887.730678	252597.074928
9	172870.469826	160200.0	164017.419915	181560.0	181238.42	170997.533996	165743.545955
10	111914.174147	117000.0	163970.840168	134230.0	116506.08	107268.891118	101200.106546
11	227265.580278	160000.0	163980.209900	174380.0	137289.27	223060.695680	207921.683185
12	84029.953096	108480.0	163963.965019	114890.0	144263.70	83477.105528	83144.133082
13	355999.057177	158000.0	164010.869601	170780.0	208248.75	348911.311572	312196.703731
14	280299.646577	262500.0	164046.974474	243235.6	268452.80	278310.517060	274300.227837
15	64388.153387	106000.0	163962.122750	126000.0	111280.98	76184.795433	93274.217181
16	72317.386417	147000.0	163977.660708	123531.6	131137.52	80215.851365	95545.743185
17	85125.362547	97500.0	163953.150633	125280.0	129073.50	88473.707257	97139.289314
18	228218.872100	181000.0	164026.015105	154380.0	183413.12	230562.765719	224844.603393
19	244282.069427	262280.0	164023.580816	253756.0	204911.90	246109.715083	248090.944138
20	85992.519326	105500.0	163970.151441	127000.0	117876.21	88846.584136	97479.076303
21	225135.979993	120000.0	164009.276751	148850.0	167723.65	223432.465934	222231.747614
22	154326.666713	143750.0	163991.653152	137740.0	170435.61	158473.646246	166141.528342
23	79694.912879	127000.0	163984.843310	86300.0	138577.98	82362.987023	79538.572799
24	93152.768672	115000.0	163954.009653	116980.0	126959.00	96847.822833	104302.186371
25	133709.464965	169000.0	163978.622416	139530.0	141786.48	137210.164036	139912.303635
26	205592.733748	155000.0	164022.296223	184698.0	182041.74	203487.143350	199009.300021
27	144667.431612	154900.0	163982.264543	151460.0	152054.32	140680.588651	140229.707731
28	207461.247765	167500.0	164006.618102	211000.0	210207.15	205319.122367	202326.407125
29	122735.677806	58500.0	163965.746885	98700.0	100367.80	123311.211098	119259.527999
30	-11127.227578	80000.0	163964.798488	106980.0	121596.81	-5820.646250	18999.630133
31	182274.809268	195400.0	164040.638674	199620.0	192258.81	181519.575342	186047.717725
32	276002.791348	215000.0	164050.182400	225495.8	222402.27	274340.749131	265691.558611
33	219261.194114	295000.0	163997.157552	129580.0	184150.22	211703.188279	204388.610281
34	127978.749375	121600.0	163979.026790	107640.0	123474.00	123040.691798	118490.996278

35	207577.468976	272000.0	164031.962111	192950.0	199249.39	206102.663525	207433.513098
36	281311.363925	195400.0	164050.419165	226150.0	230398.38	277958.807513	271021.636240
37	104700.273188	110500.0	163984.198453	109480.0	115141.87	107193.912999	119259.120196
38	177834.612574	171000.0	164002.545348	184560.0	173005.16	178886.091531	175618.790607
39	191758.714922	129000.0	163983.654381	149000.0	133728.70	177882.853813	159275.330135
40	68162.400620	105500.0	163950.788241	112280.0	113432.58	68008.864538	77518.447754
41	215983.386665	242000.0	164059.380001	253000.0	218559.35	218913.259473	228562.816067
42	248854.021372	187000.0	164063.954357	190588.0	224403.64	247440.735358	246810.085770
43	194848.764533	193500.0	164031.714213	203860.0	195879.77	190166.497470	185474.575225
44	210255.384031	256000.0	164014.407269	178475.0	211375.97	203295.995722	188444.677887
45	124178.861347	120000.0	163961.797550	116600.0	114819.50	123407.461666	123095.648656
46	99796.367522	132500.0	163958.350221	138940.0	126252.92	112265.804495	125383.888875
47	54738.675199	123000.0	163963.553690	120990.0	112745.04	59202.164709	66228.019877
48	255835.815616	235000.0	164055.586762	212180.0	251936.28	255409.349141	250730.492372
49	289903.391119	380000.0	164064.903309	310550.0	404289.59	290330.314455	286826.683838
50	140442.697257	135000.0	163970.721404	132494.2	133056.64	134975.854682	123092.921194
51	177731.056963	165000.0	164020.684219	200700.0	197832.72	182387.624327	197380.350785
52	45687.420117	104900.0	163951.341806	103296.0	101990.75	54720.276664	67451.206477
53	100137.013829	35311.0	163958.567083	111320.0	81641.94	87188.159705	70584.391743
54	259472.631922	267000.0	164065.372654	206400.0	256063.78	258673.202881	257308.862445
55	197011.102113	171000.0	163998.634257	144680.0	184033.57	197047.808951	194908.911072
56	96780.283031	104000.0	163957.183817	123280.0	112336.92	93400.459883	95403.926838
57	230002.397732	185000.0	164047.096490	189178.0	204092.09	226973.436129	221590.139721
58	72582.611669	158000.0	163992.354874	142980.0	162212.87	73031.098029	88065.858147
59	234652.314232	214000.0	164054.617088	204750.0	233970.39	234987.917360	238560.356598
60	130689.707467	147000.0	163983.225700	136480.0	132874.00	130091.116598	130910.062305
61	270080.336097	180000.0	164029.321521	271000.0	249944.39	262970.812223	251053.584679
62	161930.067470	161000.0	163967.892279	147550.0	132891.69	165534.096118	164693.883294
63	231864.149483	175500.0	164028.811997	140800.0	184700.98	229550.819257	225340.649012
64	266066.350398	227000.0	164055.091923	182900.0	245189.89	263494.443980	259822.741755
65	116364.180044	144000.0	163976.166455	95480.0	119152.03	116009.109223	112375.756117
66	205036.346381	153500.0	164004.583440	155100.0	174172.01	196813.803521	184021.953097
67	192065.771883	194500.0	164039.210459	207100.0	213009.10	191538.175782	200052.283306
68	120754.574077	124000.0	163994.285551	120711.6	161743.48	119815.181582	120131.218845
69	141461.398306	243000.0	163977.727404	146100.0	135001.21	146642.813275	158887.367800
70	259955.894520	201000.0	164054.720666	285500.0	235832.99	261254.833900	258309.222526
71	169242.789031	75500.0	163997.382098	128750.0	165668.98	163595.633188	161959.778450
72	154666.010282	136500.0	163977.051338	121180.0	126700.49	156461.170593	153287.574198

73	69182.698897	96500.0	163980.242855	147900.0	116825.37	65935.117715	69112.037820
74	239845.260919	168500.0	164010.463447	172200.0	206578.48	240120.035350	234708.218653
75	202142.573654	237000.0	164070.373990	212300.0	210211.64	206349.325464	216766.712850
76	324343.251412	354000.0	164074.512062	316400.0	344609.61	319557.294389	307403.124537
77	230686.949782	153500.0	164031.099556	176220.0	178272.92	231815.355631	233422.795916
78	309420.134420	235000.0	164067.414002	265300.0	264162.08	306790.365332	296011.367179
79	261050.133759	189950.0	163998.208809	198560.0	183278.29	257539.001937	241001.986673
80	269698.977834	183500.0	164064.884919	239450.0	232131.07	270306.155490	270168.955947
81	151107.079540	239000.0	163967.200859	124325.2	143643.50	139090.767444	116805.942261
82	154584.982286	134500.0	163981.847143	140400.0	144253.65	156483.393058	154408.292068
83	158903.198510	207500.0	163984.595900	137580.0	160833.79	158817.810487	156356.159392
84	260350.703668	179900.0	164032.118963	193470.0	225500.70	260787.688140	259804.543128
85	179498.915973	198900.0	164023.193636	179457.0	218056.42	185287.475924	194953.582013
86	79840.218927	109000.0	163949.595829	112120.0	108906.75	79477.927267	83092.633199
87	287159.189227	611657.0	164069.629114	262300.0	440745.03	285283.197287	279352.794851
88	173419.702186	135000.0	163975.326388	146900.0	135044.09	162257.871500	142728.387821
89	219607.039273	266000.0	164039.604684	197042.0	198141.35	221269.759801	223074.502503
90	221155.519686	220000.0	164035.439614	205763.8	229837.68	217592.588523	212202.406153
91	144570.967927	75000.0	163976.438355	139100.0	148542.61	144030.028810	140373.637229
92	125610.478827	119900.0	163973.793053	109640.0	113241.03	127488.666111	124062.986594
93	147781.247641	132000.0	163967.296481	129560.0	139635.71	146983.096930	141002.623117
94	197113.931308	181900.0	164031.343910	206940.0	181522.30	198229.738485	200003.939940
95	215138.415597	160200.0	164007.021285	178220.0	189069.98	212842.155134	201500.257052
96	334380.387674	262500.0	164072.024168	272780.0	262877.08	329500.642574	313956.937232
97	227569.387919	176500.0	164045.068009	230740.0	196706.99	228451.953206	228427.562246
98	270162.465956	380000.0	164066.202784	342800.0	361990.62	275083.578312	282368.523248
99	175570.760224	140000.0	163987.329899	162551.6	150348.07	173856.183321	167594.429086
100	273066.271576	233170.0	164067.807074	247600.0	232994.57	276645.643371	279197.579956
101	188954.471664	184000.0	163977.614599	116750.0	156667.50	185167.327174	172143.982578
102	175446.370100	157900.0	163973.948314	131980.0	140185.12	170861.096658	159919.059513
103	152316.318914	272000.0	164017.278261	192800.0	180918.00	150588.655607	151731.732436
104	172437.726998	197000.0	164025.760727	217980.0	179691.95	174037.000710	178374.440408
105	112280.663429	139000.0	163976.704641	162280.0	132978.33	115733.530756	125587.140055
106	279567.069391	212000.0	164044.942802	263100.0	244457.04	276579.112392	267763.244062
107	118957.031039	127000.0	163982.193919	158920.0	141438.53	119195.319008	121849.404741
108	185539.289635	215000.0	164045.999532	186686.4	209369.66	186756.661268	195084.083268
109	126307.786210	179200.0	163996.067219	170300.0	184847.97	148081.360192	175229.203280

110	232638.347907	228000.0	164018.043621	225100.0	214602.08	231180.171965	234768.528513
111	226192.830248	203000.0	164022.909464	190700.0	186436.84	222833.710669	217206.810959
112	108315.372484	110000.0	163981.382615	123896.0	122925.52	123500.264933	146563.803947
113	312352.112362	184000.0	164065.714120	203780.0	283236.78	307280.830414	292207.157848
114	125375.762784	147000.0	163977.983578	132270.0	123105.54	125059.349269	124722.048260
115	33007.341582	87000.0	163960.853185	120630.0	93024.77	36771.486646	44762.246539
116	97585.175391	105500.0	163967.757013	143790.0	134528.18	102788.323498	113636.927193
117	213237.182507	197900.0	164040.623272	191780.0	204714.24	210035.709088	206481.233165
118	202643.925271	250000.0	164000.377291	172800.0	183765.24	198393.186413	184481.506544
119	119486.152047	155000.0	163949.864517	130880.0	122986.04	112069.630554	101420.851470
120	166848.271343	82500.0	163980.007242	149491.8	133929.37	162898.475458	150599.181024
121	257772.216822	215000.0	164041.989037	210350.0	219516.06	259206.483172	256745.128334
122	148918.228824	181000.0	164024.447793	179400.0	182758.96	152037.727755	160868.181629
123	112964.223526	140000.0	163982.532380	110300.0	134630.55	118354.554772	126387.265689
124	98383.454386	106000.0	163966.257531	135100.0	120460.76	100336.638405	106209.083221
125	201604.840825	128000.0	164006.198763	186180.0	175331.57	201423.496927	200710.254028
126	106114.378365	135750.0	163938.880372	111880.0	122893.50	105269.855860	106490.620330
127	244193.082886	171000.0	163990.321693	144180.0	179235.66	239079.696374	229191.278444
128	-55114.513366	72500.0	163987.224633	107260.0	87865.95	-35104.496473	10345.746934
129	54582.857140	115000.0	163963.468516	116280.0	95202.89	56672.459333	59355.111247
130	218632.878097	163000.0	164011.253996	209580.0	174977.20	218145.123391	217531.607565
131	211413.256971	181000.0	164062.426967	235133.6	200187.86	210129.787988	211246.247589
132	203069.015262	139000.0	163982.763328	175380.0	148893.09	198347.352866	187333.011732
133	223811.218369	191000.0	164024.587013	200300.0	189246.30	217742.421919	202312.927800
134	112757.752764	147000.0	163992.094469	135950.0	123392.66	118547.977983	133961.505148
135	255987.135446	383970.0	164079.848473	296900.0	372983.49	258290.006572	267500.029363
136	302109.487443	316600.0	164061.295037	332400.0	275744.91	300815.574234	294075.684125
137	109433.767456	205000.0	164000.349932	177580.0	161758.66	119233.843418	142189.733730
138	252022.204830	275000.0	164048.049179	216075.0	234631.21	255932.654146	260178.001219
139	157125.239042	115000.0	163981.402442	137400.0	133321.46	156283.950197	152095.087676
140	160866.606427	177000.0	163968.936523	149300.0	133691.97	158930.925406	153532.823643
141	339898.039481	611657.0	164042.653994	286400.0	494547.65	345558.404259	339472.967477
142	36377.496681	128000.0	163955.665411	126880.0	87393.33	35501.071092	44833.591344
143	310436.802154	556581.0	164045.865546	216125.0	451421.20	308261.403073	306776.661014
144	232246.504944	189000.0	164011.838771	179200.0	203709.44	233604.191468	233104.576639
145	130324.835708	191000.0	164046.164560	207400.0	194125.24	135470.431473	152477.946458
146	132448.905319	98600.0	163977.830516	140100.0	127724.56	140337.668987	147732.637229
147	32569.342051	168000.0	163987.045071	150380.0	132126.61	39341.466151	59553.744197

148	23114.367989	128000.0	163950.081705	121780.0	102091.37	24340.290780	43293.612756
149	246029.393328	233170.0	164055.189170	208800.0	213043.39	243595.638151	238262.846292
150	163403.738347	141000.0	164028.415772	155890.4	188576.83	167464.526492	175870.550499
151	166370.791942	133000.0	163968.265489	138390.0	128538.20	164328.140993	156941.050371
152	240149.045139	183500.0	164060.997884	302000.0	242754.53	237129.397330	236607.283697
153	97738.614127	128200.0	163956.075114	137150.0	127144.21	99178.128053	103129.804325
154	139864.632191	135000.0	163966.008257	131870.0	126696.58	139298.060064	132071.129061
155	183719.255611	122500.0	164011.828198	160500.0	183967.51	184217.176462	189395.021514
156	241096.730069	223500.0	164025.280734	227100.0	204737.84	237699.174640	227447.006045
157	219695.135407	137900.0	163994.369135	155180.0	167520.11	215115.166281	204315.972094
158	141850.416941	150000.0	163973.997140	154650.0	129345.05	144475.062483	151719.736633
159	222176.365455	191000.0	164029.233570	211456.0	189574.80	227707.798334	235216.618296
160	260361.543211	255900.0	164051.083844	218615.8	221584.32	257749.853713	253589.054120
161	196153.166066	217000.0	163996.273722	147080.0	192173.47	191720.848307	183039.596657
162	243411.351425	200500.0	164037.086070	281800.0	217451.93	246721.602368	252089.072773
163	134878.211781	144000.0	163966.513221	133430.0	130568.64	135570.999710	131537.899355
164	69169.019684	155000.0	163962.480993	119080.0	127827.71	74403.860356	85341.746054
165	234899.625961	377500.0	164071.116315	297400.0	321196.25	239065.996936	250092.891337
166	244147.929543	159000.0	164010.753575	188100.0	191186.74	241468.323962	232941.533250
167	228153.793792	172500.0	164065.515255	235000.0	225669.67	229411.351945	234943.421336
168	124727.379790	110000.0	163960.486784	115401.6	131759.00	122567.949223	123350.129128
169	172511.279346	230000.0	164035.818763	184298.0	196196.53	174406.341345	176208.392290
170	149462.649855	302000.0	164005.528779	145700.0	181220.55	149609.419748	154316.227804
171	115260.400160	129000.0	163983.535144	141750.0	150539.50	116384.742390	123907.788901
172	145084.674739	180000.0	163991.522870	164400.0	183689.74	146813.610769	151250.348549
173	221225.402268	345000.0	164052.889464	203028.0	262778.46	224457.168592	232898.489220
174	105255.228839	134500.0	163973.286026	132700.0	133661.50	107055.165231	115504.186040
175	301822.459101	222000.0	164073.228258	314500.0	338525.54	303497.343201	311814.818674
176	79182.319761	101000.0	163962.568661	107296.0	102423.37	75710.492097	73811.737897
177	95574.862579	150000.0	163935.719194	124170.0	125568.02	94841.480906	92607.215797
178	176383.374426	180000.0	163994.981312	141800.0	165022.37	176215.928468	177264.316939
179	172921.569840	200000.0	164014.599660	165480.0	208811.94	166784.687479	159748.604294
180	188408.951484	215200.0	164042.163848	188200.0	206589.27	189342.911750	190970.589417
181	171242.843069	151000.0	163991.584627	182900.0	161281.24	176243.031506	186065.602041
182	292930.255616	315750.0	164062.064253	301900.0	300927.71	292259.777420	292626.828503
183	156163.444513	191000.0	164012.346823	169880.0	177280.13	158411.960330	157108.857894
184	331040.439680	383970.0	164049.354963	355790.8	389769.01	327655.754349	321588.583474

185	315915.953524	465000.0	164078.833465	385500.0	391926.97	316331.336792	314815.247757
186	230557.574429	262000.0	164015.242901	209760.0	211667.14	229497.852799	225102.676882
187	147251.741082	167000.0	163969.447850	97876.6	127568.89	144488.133130	143305.087785
188	146631.117562	129000.0	164007.538307	180200.0	168216.29	145697.771775	146980.317570
189	126978.336000	241500.0	164020.615752	192000.0	182659.61	127097.316417	130886.582097
190	140793.366757	144000.0	163979.812627	144700.0	122704.28	139887.420167	142412.461058
191	249016.221768	271900.0	164041.537848	250570.0	219066.70	250789.333167	253501.046706
192	303094.113332	208300.0	164050.770251	225080.0	241050.83	301342.086326	293561.828498
193	95090.381486	157900.0	163974.841800	134080.0	130820.26	95161.858164	96691.511715
194	169515.822467	112500.0	163966.001600	156800.0	122676.83	168979.212728	160355.775856
195	60661.110332	72500.0	163959.081799	125760.0	91535.55	61543.155634	58536.617167
196	157319.811149	206300.0	164019.170211	181560.0	179835.12	157999.548460	160224.507190
197	198007.049826	221000.0	164053.580597	220480.0	209751.98	198623.003080	205591.143444
198	113705.060454	109000.0	163963.646376	113580.0	109553.82	113986.910459	110475.111307
199	133513.855278	129000.0	163997.584970	185980.0	166526.87	135269.563398	143749.112128
200	230129.591674	174000.0	164033.723701	145380.0	187781.95	231775.045834	234983.368111
201	181910.119694	108000.0	163982.060206	215720.0	132717.25	180735.421944	172818.102973
202	306706.946419	485000.0	164069.324087	310590.0	347910.90	310162.413618	316647.402205
203	268894.983025	465000.0	164051.232139	194773.0	381776.01	275140.934171	280690.393525
204	126878.218173	106000.0	163958.355919	116581.6	122341.88	128714.693617	129363.332461
205	205560.615315	167900.0	163990.113138	158880.0	176433.29	203415.859867	198136.475730
206	282863.614651	235000.0	164058.246066	270300.0	248045.09	276720.273080	262359.754488
207	185621.306349	137900.0	163978.521415	151700.0	140973.11	185679.573046	179923.142209
208	55941.157740	102000.0	163967.615768	121080.0	105372.29	46564.611546	41854.778522
209	233728.413395	168000.0	164010.313194	179610.4	185910.46	224704.639396	209407.017478
210	151126.644488	128000.0	163964.786766	147325.0	135078.83	151169.794162	152051.485534
211	122403.585999	107900.0	163977.592388	126760.0	127683.55	125107.760633	132314.836712
212	99180.878989	97500.0	163974.462504	129450.0	123268.30	104207.623011	116388.510348
213	231299.998064	271900.0	164015.501297	184740.0	190649.28	228725.018263	224993.001905
214	80411.502868	123000.0	163968.914460	137080.0	141333.54	82980.055789	90189.351735
215	111699.114508	55000.0	163969.946684	97200.0	89439.95	113246.285251	118508.957447
216	181172.071524	174000.0	164024.431883	143700.0	184340.55	182296.782274	187862.411102
217	104475.001813	97500.0	163975.056683	139200.0	133595.83	107150.846871	109036.384369
218	147042.634810	127500.0	163972.677273	129560.0	148522.26	147227.767029	146882.222493
219	229539.734049	179000.0	163976.356473	134270.0	139951.56	227421.064767	215484.679076
220	207954.998163	193879.0	164053.070332	185410.0	192712.82	207076.561212	207763.511632
221	219595.825364	212000.0	164069.017792	265580.0	236686.20	222656.277442	232781.773049
222	279054.073739	380000.0	164063.921149	330080.0	384381.37	279261.003209	278923.262458

223	135313.703065	171500.0	163961.696530	111580.0	131443.51	136783.093779	136070.194688
224	209221.345533	150000.0	164021.470501	200400.0	182153.28	207870.249910	201673.922769
225	331898.034797	360000.0	164078.097118	363500.0	403282.84	333156.161160	329972.063286
226	173098.103338	173000.0	164031.860178	201715.8	211455.95	173741.641796	178530.218079
227	112281.540992	87000.0	163969.693769	124800.0	102237.22	109138.121703	93490.534711
228	318471.676896	310000.0	164071.111863	327823.4	414077.85	314749.773095	307109.446064
229	181090.619229	227680.0	164019.294421	189200.0	182181.51	182222.104033	181336.313689
230	67062.902583	80500.0	163952.321327	108000.0	96731.21	65006.561473	65776.811568
231	227611.360901	197000.0	164014.953203	183400.0	221060.00	226307.056521	217794.304370
232	102995.539387	126500.0	163978.121542	109776.0	147529.59	98947.908302	91840.729264
233	124529.581825	162500.0	163971.913796	113820.0	129206.60	125174.979434	130431.211244
234	105538.548972	97500.0	163962.399401	127825.0	118090.04	111207.303373	116745.218799
235	169395.609532	158000.0	164019.719249	182960.0	201310.64	173823.365038	183612.502096
236	179045.204604	302000.0	164024.377555	194300.0	187670.43	177323.256040	177718.294044
237	256009.502874	181900.0	164023.749593	234380.0	216815.98	257664.927601	259191.600512
238	210381.006861	190000.0	164019.371210	203061.0	191471.89	212570.439757	213397.416188
239	177862.676529	208500.0	164051.091780	194000.0	208408.11	176824.632545	177510.187130
240	260466.067422	275000.0	164072.393874	256280.0	229945.87	262354.793420	263360.033106
241	162922.346145	153000.0	163983.726680	153350.0	149220.06	164844.519843	168002.129811
242	253688.894433	164700.0	164029.478727	282200.0	229285.08	263349.720061	276946.574064
243	124028.568822	106000.0	163977.841777	128700.0	109920.83	110862.173691	84346.659548
244	136435.872255	150000.0	163977.209529	128050.0	133438.00	139953.116268	144800.649781
245	208712.743827	226000.0	164064.109617	261880.0	227118.42	209181.514880	218188.904153
246	235421.305602	241500.0	164009.766514	161500.0	226691.80	228935.184266	220342.316822
247	348564.005421	230000.0	164077.030573	282650.0	344138.37	348146.042560	334592.223628
248	198732.686712	140000.0	163978.634139	138200.0	135693.08	191546.368399	177475.821686
249	113380.880870	167000.0	163963.714145	123380.0	134414.62	110716.561846	105975.231466
250	256206.493335	134900.0	164001.733210	156665.0	165957.37	254934.289838	248607.716032
251	199352.625020	227000.0	164047.464963	194800.0	218896.25	198424.767606	195007.423264
252	77115.370633	119000.0	163955.592762	122180.0	109475.45	80036.923968	93440.579667
253	105116.241182	106000.0	163968.943203	117500.0	116439.09	107577.810665	101596.704371
254	99226.309699	132000.0	163976.955044	151100.0	140310.08	103471.578923	111142.267428
255	190530.082668	197000.0	164045.031715	231500.0	198834.45	191688.620513	193508.784583
256	43938.626611	93000.0	163977.361845	96776.0	88609.05	39320.397435	41236.733842
257	289884.639510	394432.0	164068.443964	302200.0	401752.59	292605.174558	299636.214022
258	207726.171408	180000.0	163999.996240	147300.0	173729.29	208916.950100	208454.683734
259	120220.103004	142000.0	163956.483020	133900.0	137269.29	119486.534155	119259.243848

260	264438.204811	211000.0	164054.937399	230850.0	232672.30	257839.180140	248959.627138
261	235219.698976	173000.0	164045.159425	198520.0	216123.33	235913.926979	236346.013446
262	81073.629772	124500.0	163963.254954	135650.0	121424.13	80850.592768	88269.272748
263	131156.811457	104000.0	163969.806946	122726.0	118903.22	126995.863633	125537.167279
264	163185.556370	208500.0	164024.720547	205660.0	187346.73	165240.560345	168446.495624
265	222598.789904	191000.0	164025.594798	175500.0	200005.13	226694.600034	231199.233839
266	102317.384000	124000.0	163957.500288	112440.0	122197.74	101854.596547	94573.559056
267	214420.839767	160000.0	163992.273817	141260.0	168847.42	211724.500618	202523.057059
268	231942.196146	227000.0	164054.288160	260080.0	237174.50	233404.287338	238691.340027
269	220118.755373	207500.0	164055.975242	218998.0	217122.97	218656.417241	216267.010035
270	87490.845006	124000.0	163982.789011	137494.2	128111.54	82152.258994	83542.661261
271	55543.536533	110000.0	163966.218565	123770.0	119812.04	72499.408654	100584.520642
272	305102.607031	415298.0	164065.471221	294800.0	383721.21	304553.438839	300047.887177
273	179796.268854	120000.0	164000.798210	149700.0	182005.94	176235.794963	173449.584726
274	298414.048620	277500.0	164064.693475	305497.8	286863.31	292069.022049	281083.161466
275	246025.099556	227000.0	164054.037629	239300.0	201190.01	244717.773768	245170.660420
276	217182.754103	211000.0	164015.842306	192996.0	187238.25	216396.715113	217349.485241
277	109874.944271	140000.0	163984.735593	127290.6	136744.57	111556.071257	106555.963801
278	226788.012946	175500.0	164033.735697	175800.0	183590.52	224817.640021	221487.609724
279	329088.503974	360000.0	164071.723499	329150.0	351166.19	327260.906441	320182.589486
280	133771.355454	134900.0	163989.082299	128480.0	163181.84	137501.027484	143195.210509
281	324209.411834	394432.0	164064.101092	360322.2	376894.09	320777.420806	318433.207438
282	155288.047054	82500.0	163973.045146	134550.0	129112.87	148939.586545	138300.561163
283	282464.814561	232000.0	164049.223925	215130.0	282340.85	280732.917507	271368.664842
284	209983.247041	295000.0	164004.003653	165600.0	197386.75	208079.676289	202341.184221
285	88268.640180	110000.0	163961.014548	122570.0	119153.15	91221.155688	99493.058178
286	153414.066324	179500.0	163991.563593	144000.0	165689.03	158257.161906	168141.473561
287	262202.626099	236500.0	164051.734558	236300.0	240779.83	260126.008807	256361.687499
288	131047.370429	157000.0	163982.878265	123201.6	176953.24	131188.856021	131614.838834
289	119609.166711	215000.0	164007.648915	158200.0	178115.71	117595.910127	125252.214312
290	174475.757113	125500.0	163984.088520	176080.0	126551.11	172008.977095	173358.542146
291	158838.682857	153500.0	163982.373326	144700.0	127385.75	158886.097848	158373.226118

In [ ]: