In [303…
```
import pandas as pd
```

In [304…
```
df=pd.read_csv('training_set.csv')
df_test=pd.read_csv('testing_set.csv')
```

In [304…

In [305…
```
df
```

Out[305…

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

1460 rows × 81 columns

In [306…
```
df_test
```

Out[306…

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotSha
0	1461	20	RH	80.0	11622	Pave	NaN	
1	1462	20	RL	81.0	14267	Pave	NaN	
2	1463	60	RL	74.0	13830	Pave	NaN	
3	1464	60	RL	78.0	9978	Pave	NaN	
4	1465	120	RL	43.0	5005	Pave	NaN	
...	...	...	...	...	...	...	...	
1454	2915	160	RM	21.0	1936	Pave	NaN	
1455	2916	160	RM	21.0	1894	Pave	NaN	
1456	2917	20	RL	160.0	20000	Pave	NaN	
1457	2918	85	RL	62.0	10441	Pave	NaN	
1458	2919	60	RL	74.0	9627	Pave	NaN	

1459 rows × 80 columns

In [307…
```
df.columns
```

Out[307…
```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAd
d',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBat
h',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageTyp
e',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQua
l',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

In [308…
```
df.isna().sum()
```

Out[308…

	0
Id	0
MSSubClass	0
MSZoning	0
LotFrontage	259
LotArea	0
...	...
MoSold	0
YrSold	0
SaleType	0
SaleCondition	0
SalePrice	0

81 rows × 1 columns

dtype: int64

In [309…
```
for i in df.columns:
  if df[i].dtype==object:
    df[i]=df[i].fillna(df[i].mode()[0])
  else:
      df[i]=df[i].fillna(df[i].mean())
```

In [310…
```
cat=[]
con=[]
for i in df.columns:
  if df[i].dtype==object:
    cat.append(i)
  else:
    con.append(i)
```

In [311…
```
from sklearn.preprocessing import LabelEncoder,StandardScaler
from sklearn.metrics import r2_score,mean_absolute_error
from sklearn.model_selection import GridSearchCV,train_test_split
from sklearn.linear_model import LinearRegression,Ridge,Lasso
```

In [312…
```
#for col in cat:
 #   if df[col].isnull().sum() > 0:
  #      most_freq = df[col].mode()[0]
   #     df[col].fillna(most_freq, inplace=True)
    #    print(f"Filled missing in '{col}' with '{most_freq}'")
```

In [313…
```
le=LabelEncoder()
```

```
ss=StandardScaler()
lr=LinearRegression()
```

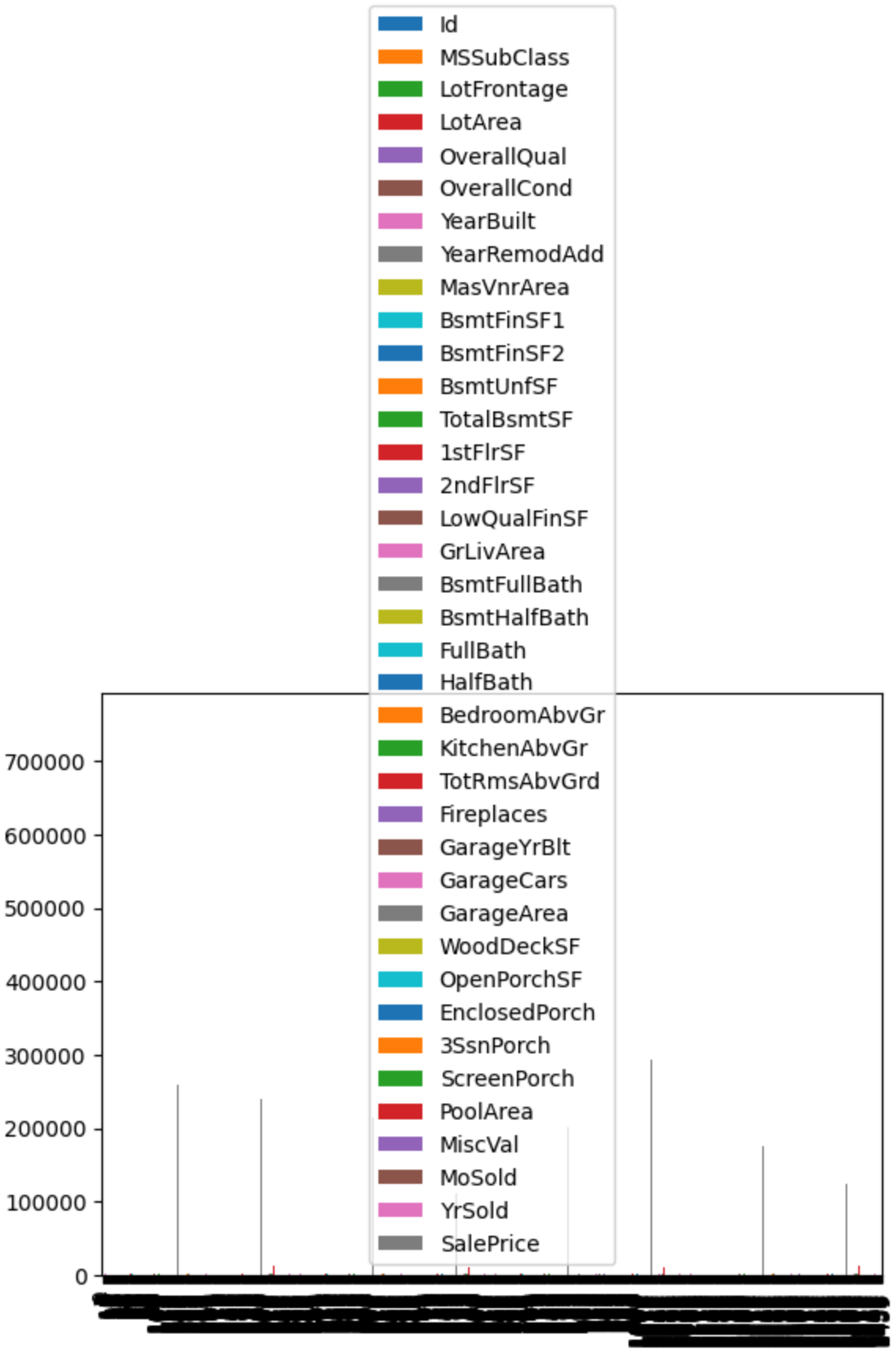
EDA

In [314…

```
df.plot(kind='bar')
```

Out[314…

```
<Axes: >
```



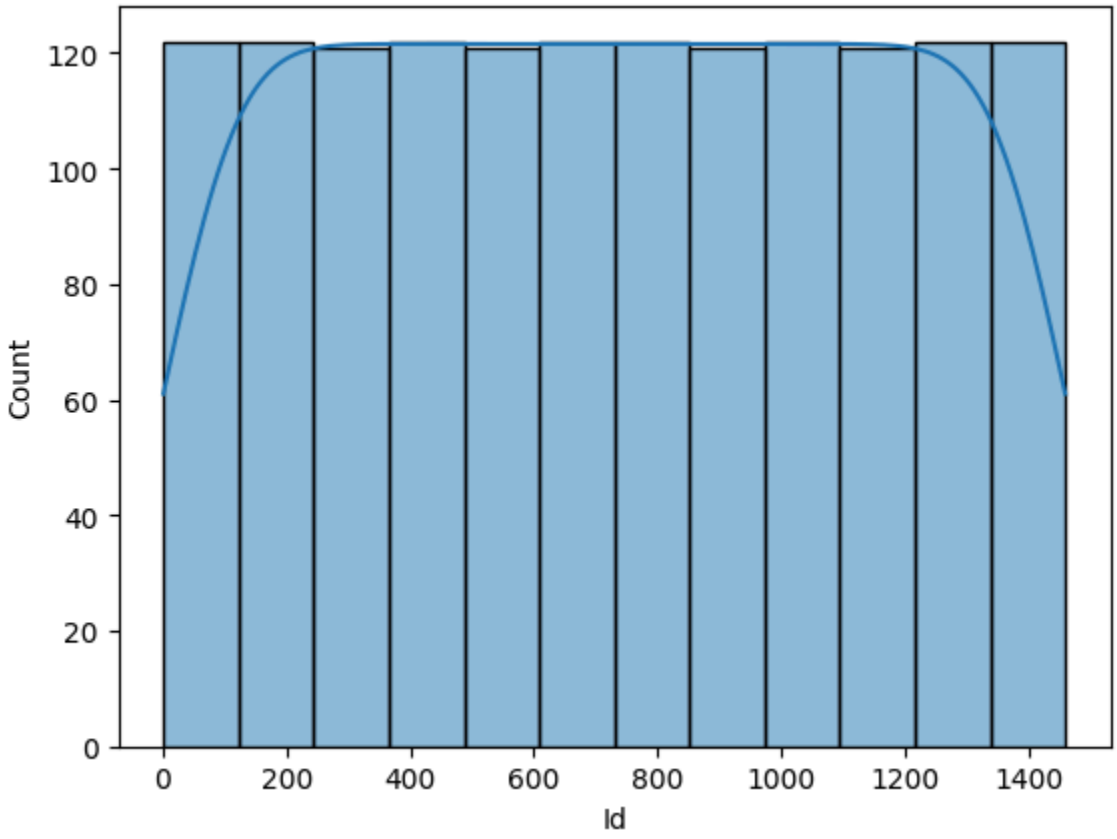
In [315…

```
from matplotlib.pyplot import show
```

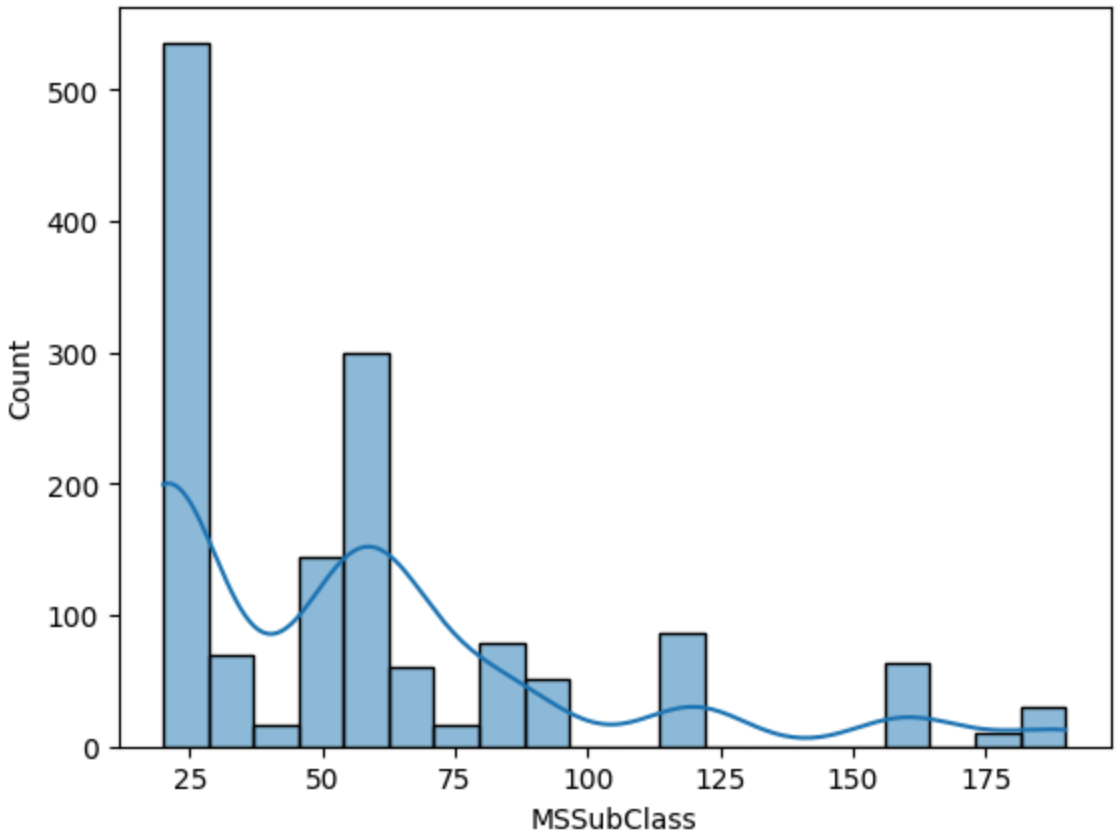
In [316…
```
import seaborn as sns
```

In [317…
```
for i in df[con].columns:
  sns.histplot(data=df,x=i,kde=True)
  print(i)
  show()
```

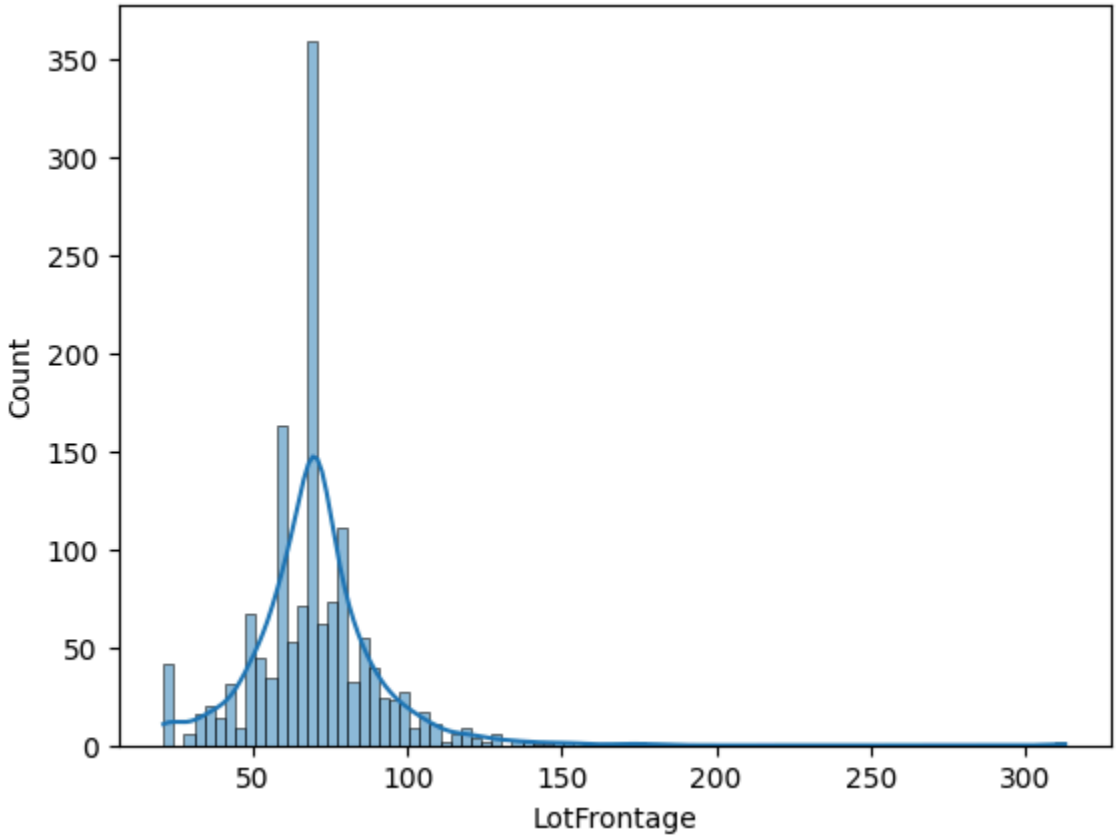
Id



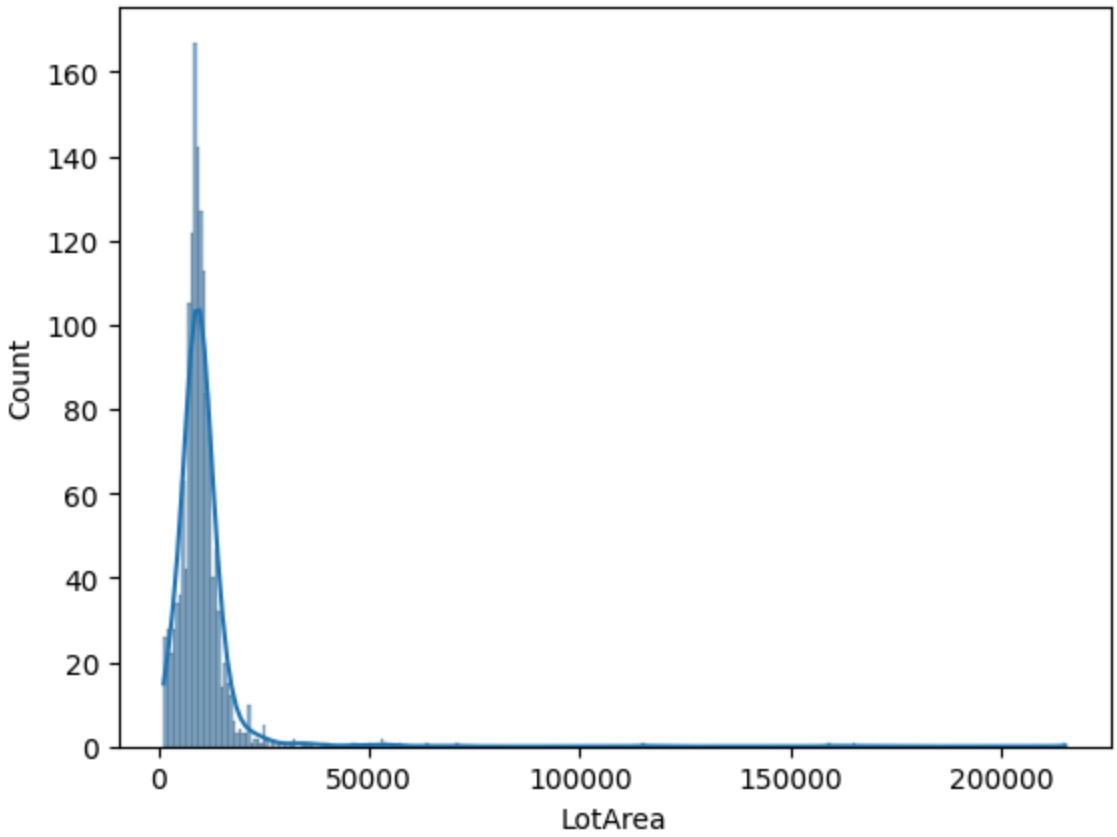
MSSubClass



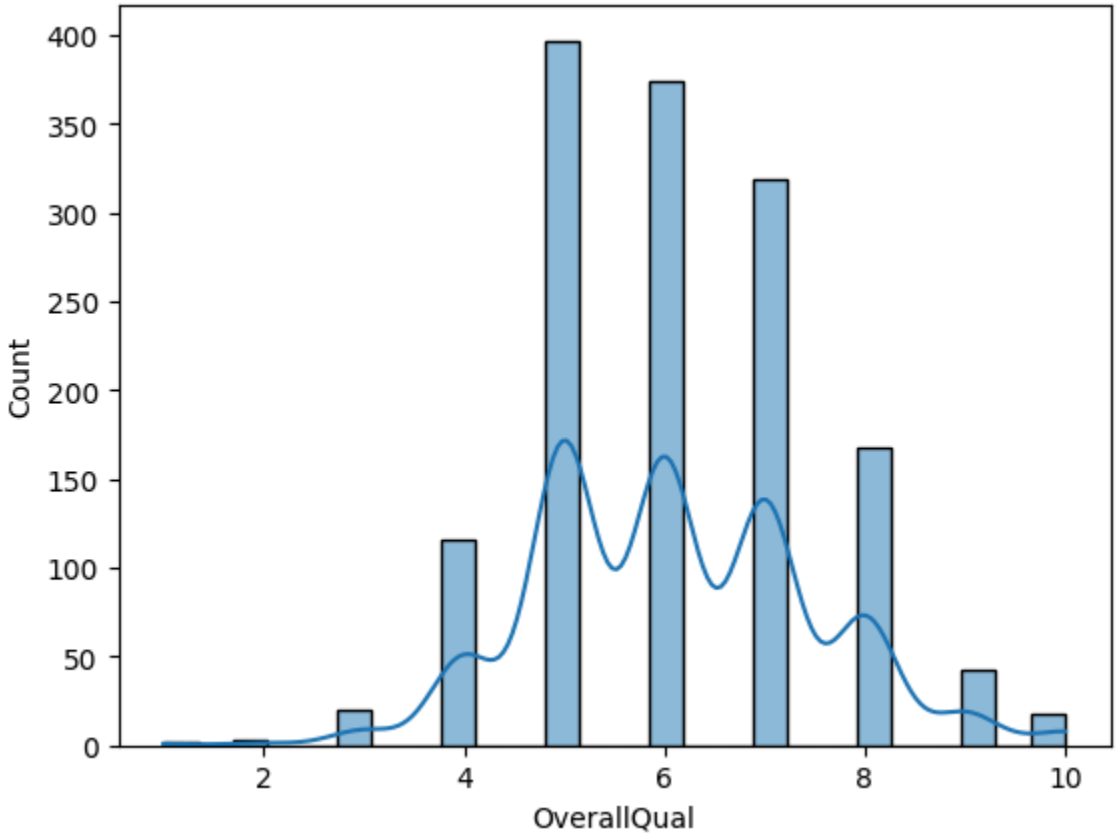
LotFrontage



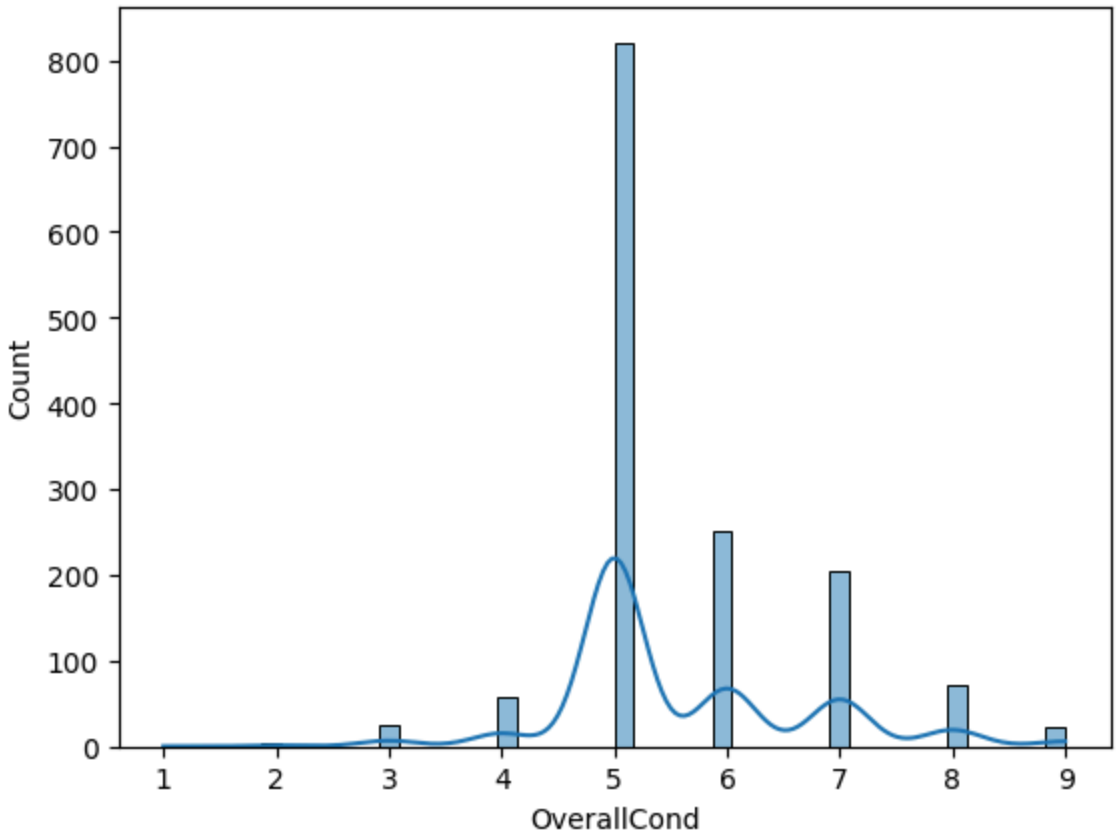
LotArea



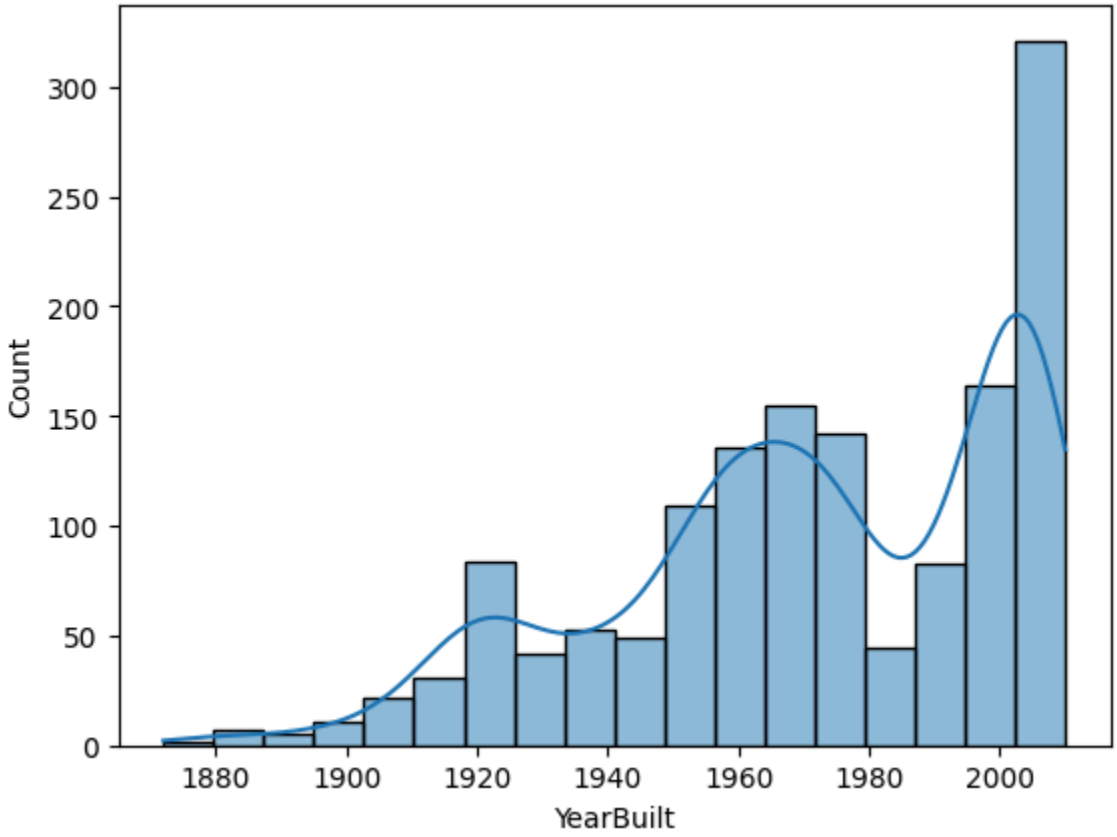
OverallQual



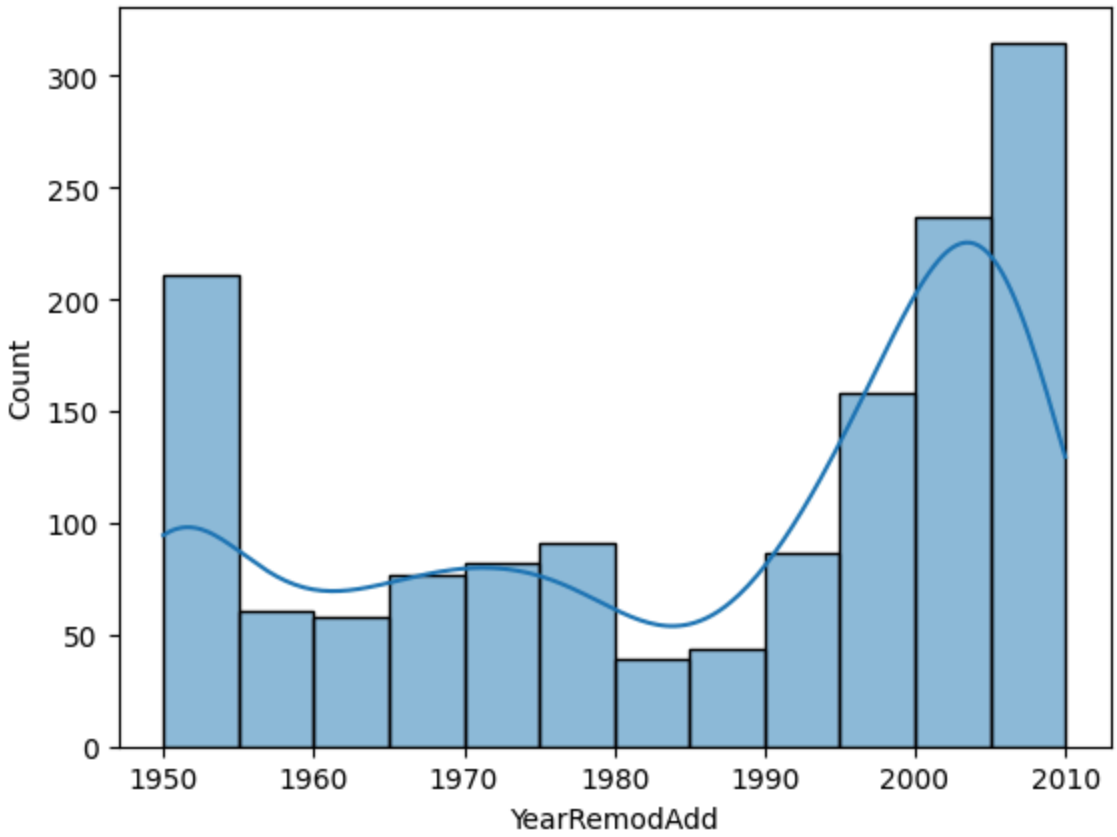
OverallCond



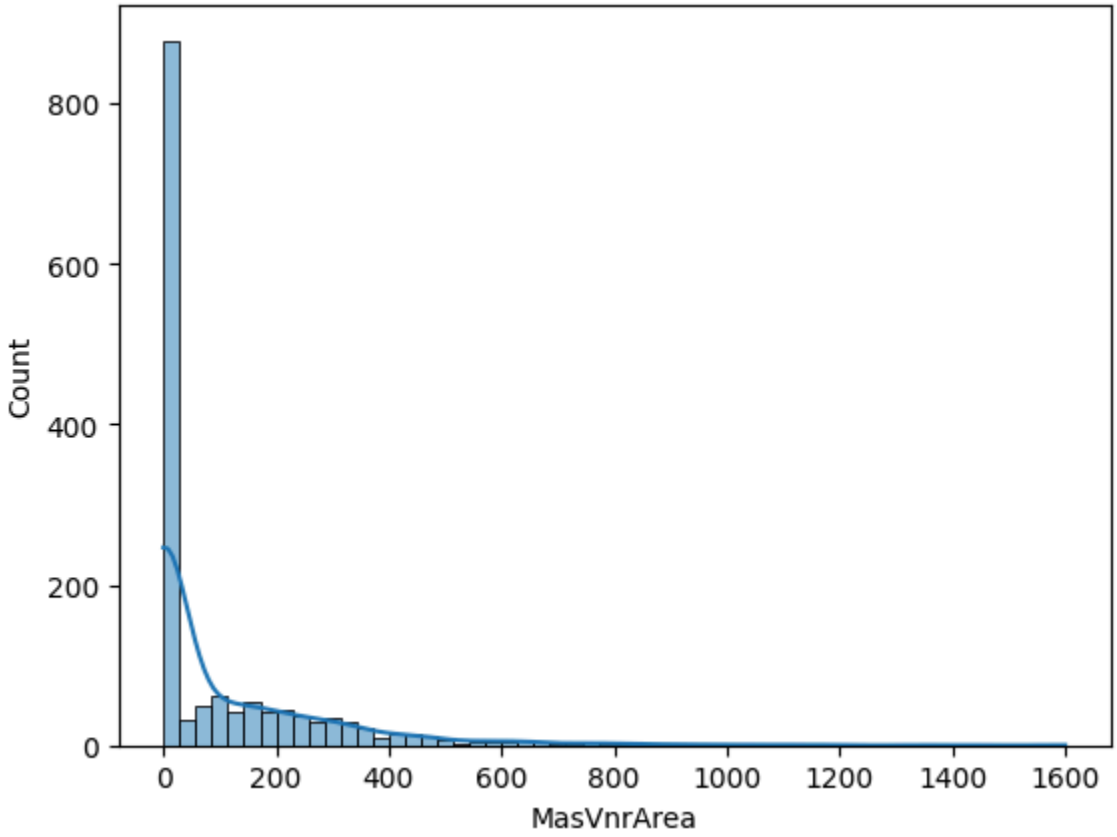
YearBuilt



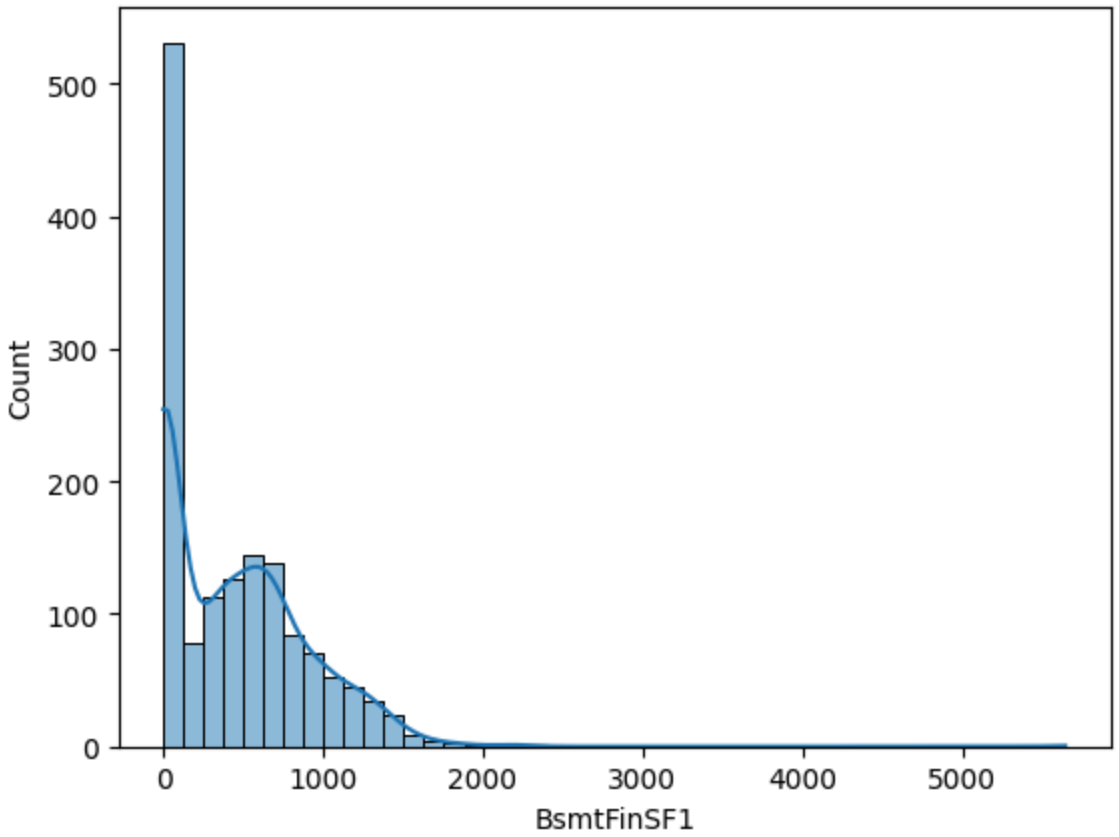
YearRemodAdd



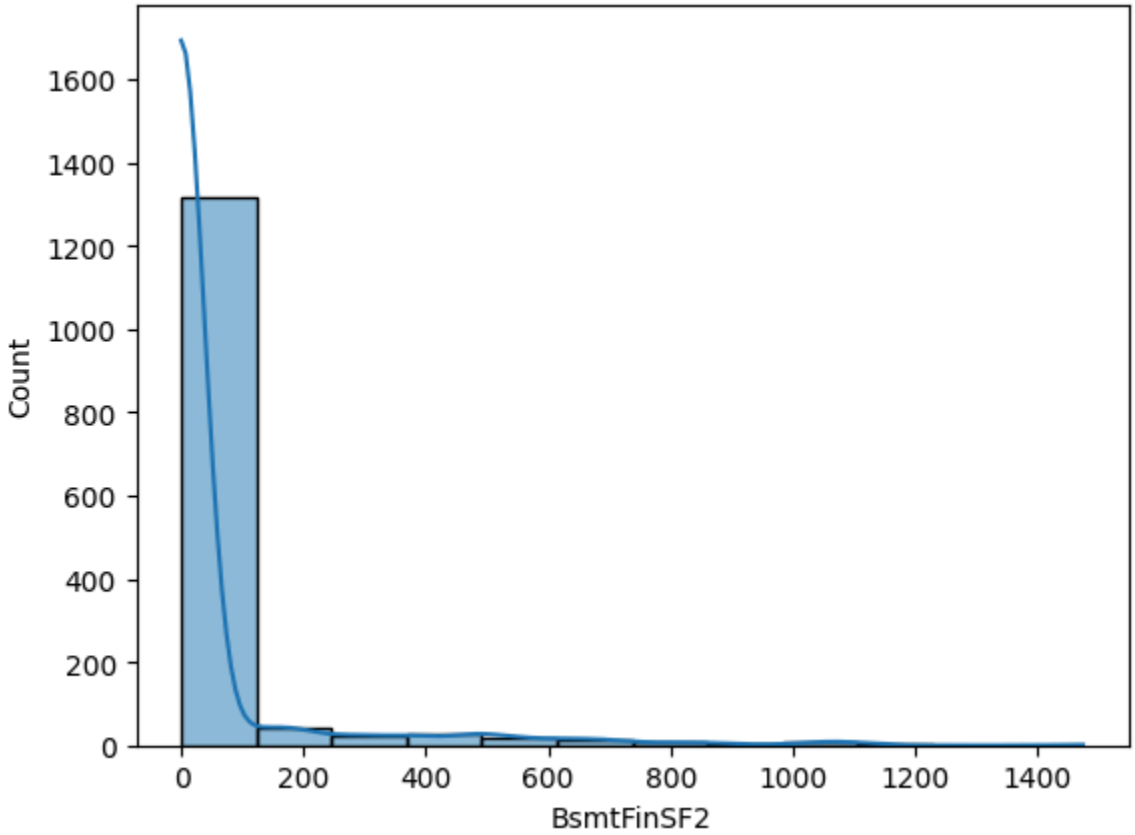
MasVnrArea



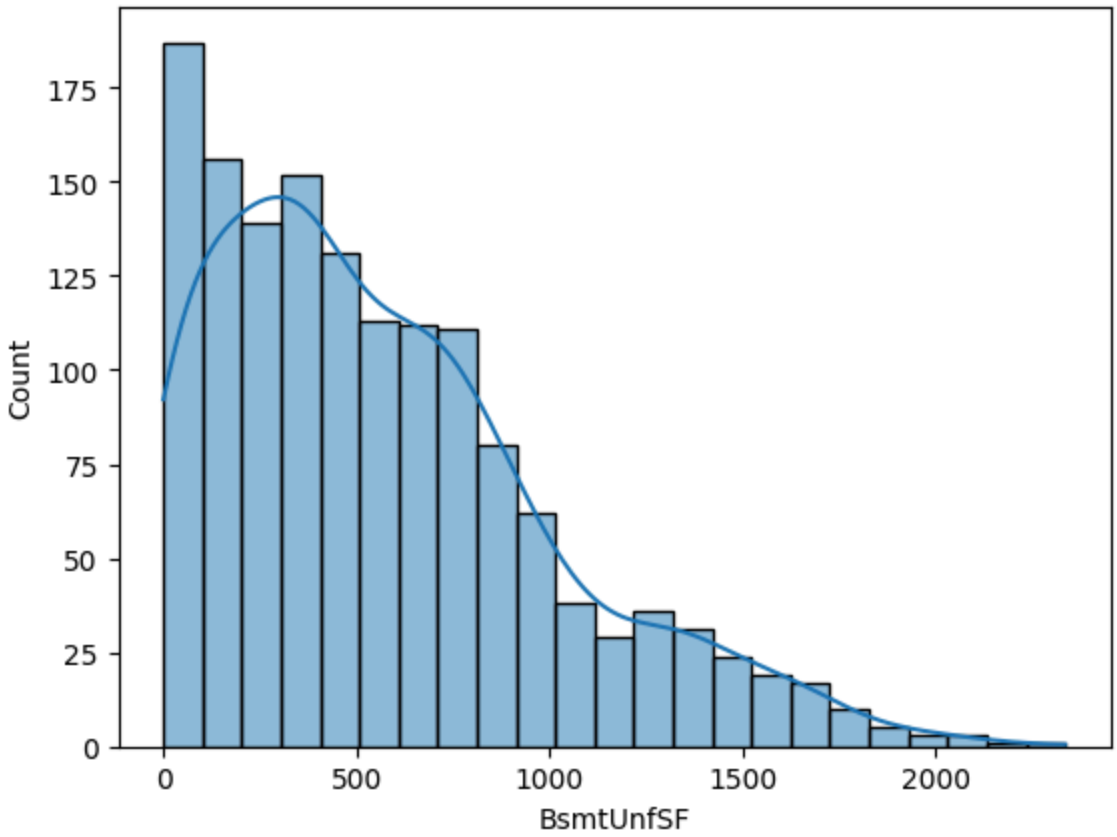
BsmtFinSF1



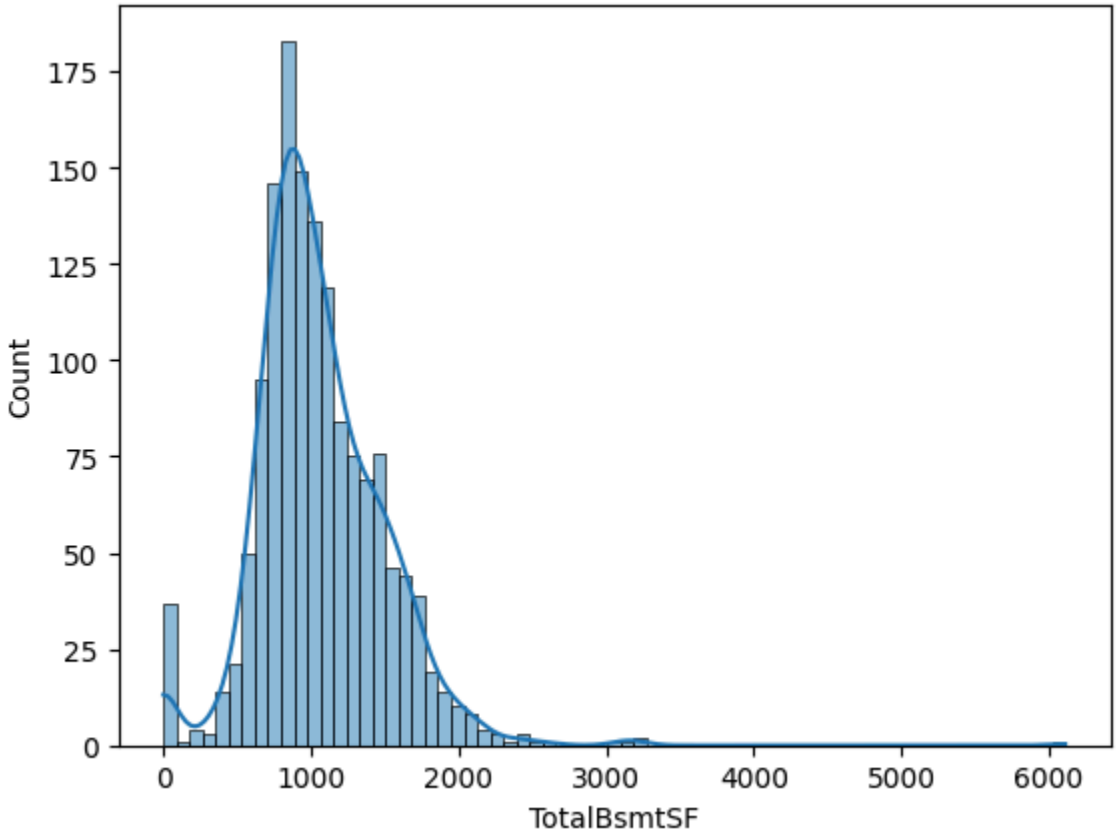
BsmtFinSF2



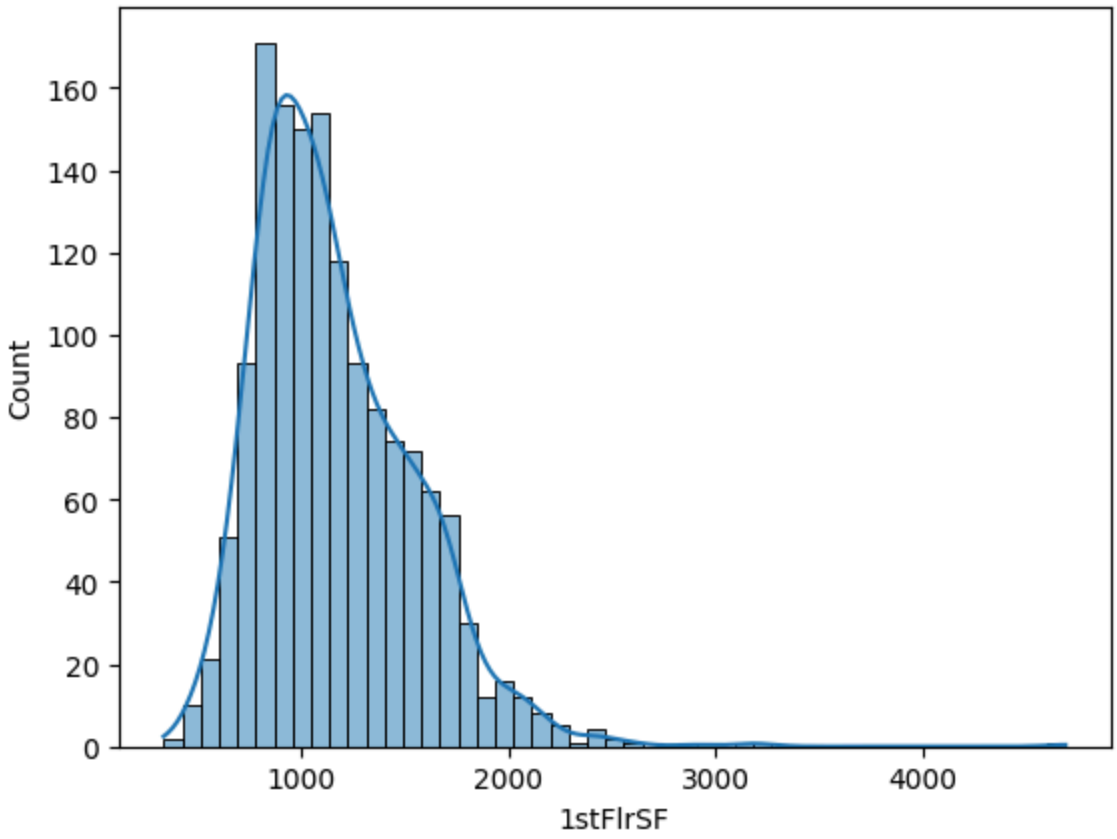
BsmtUnfSF



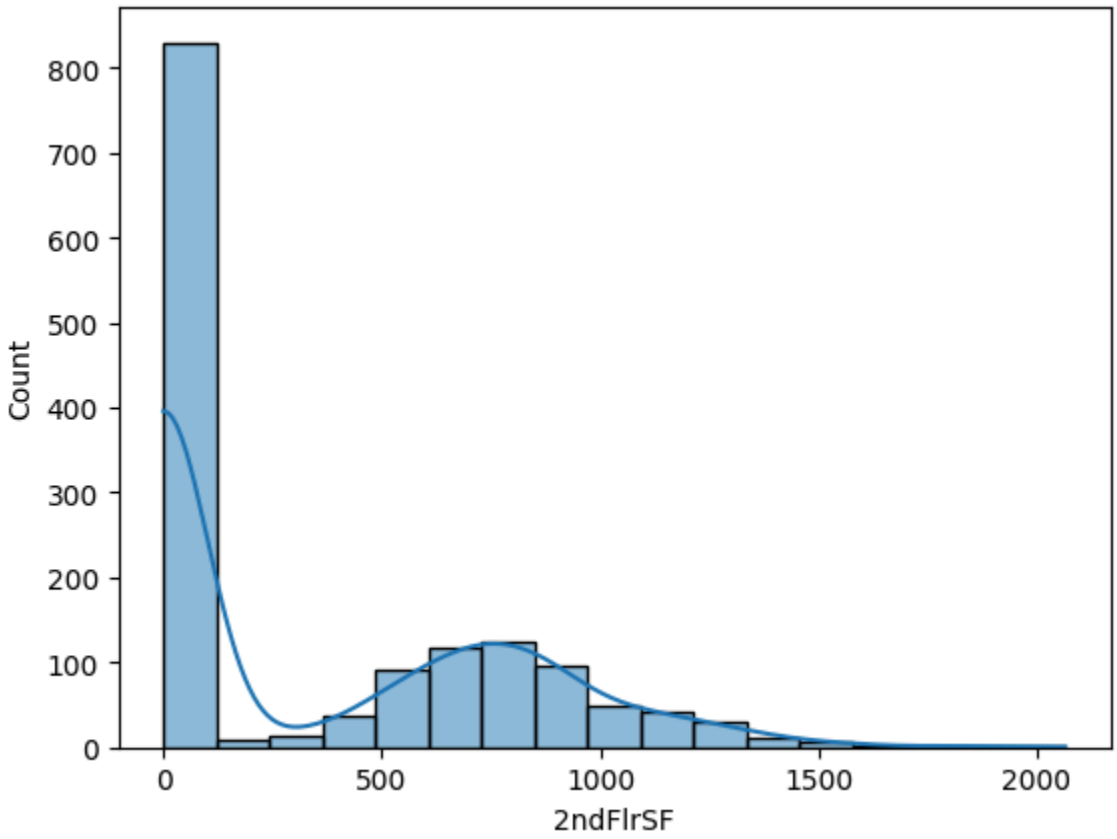
TotalBsmtSF



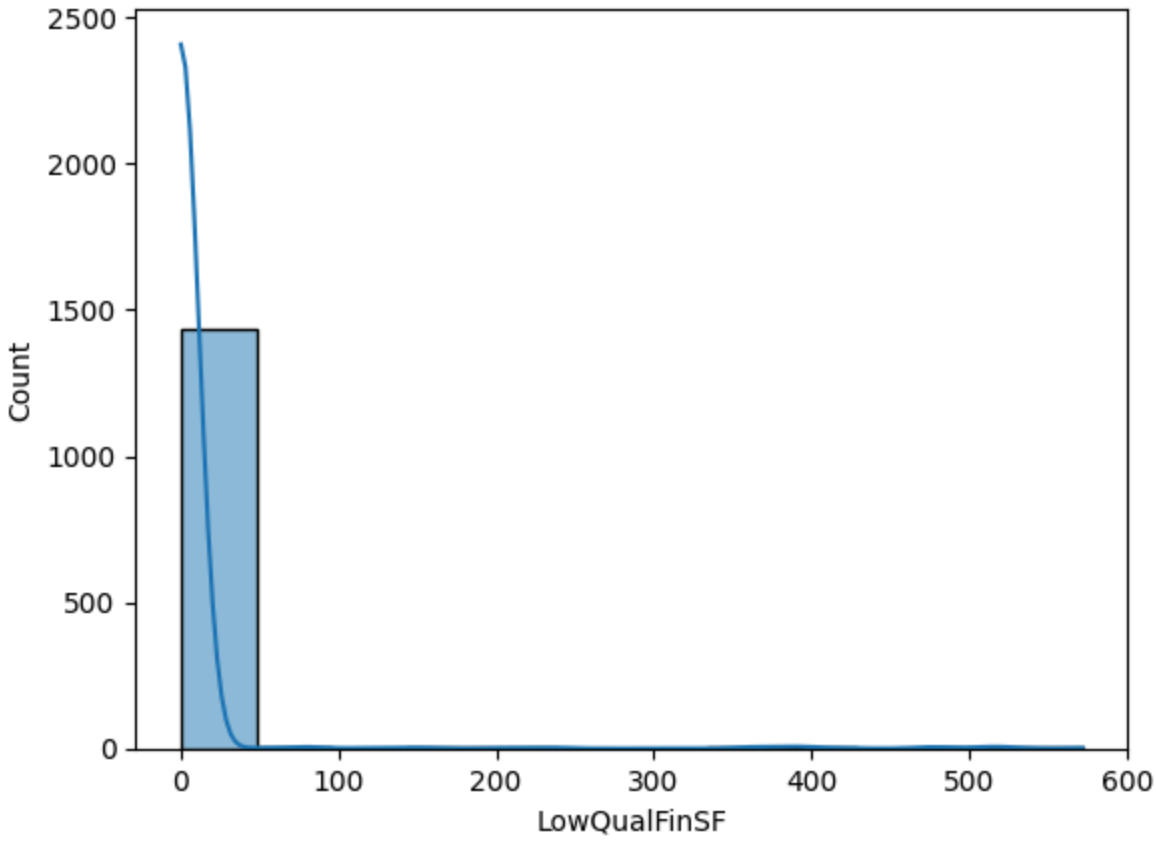
1stFlrSF



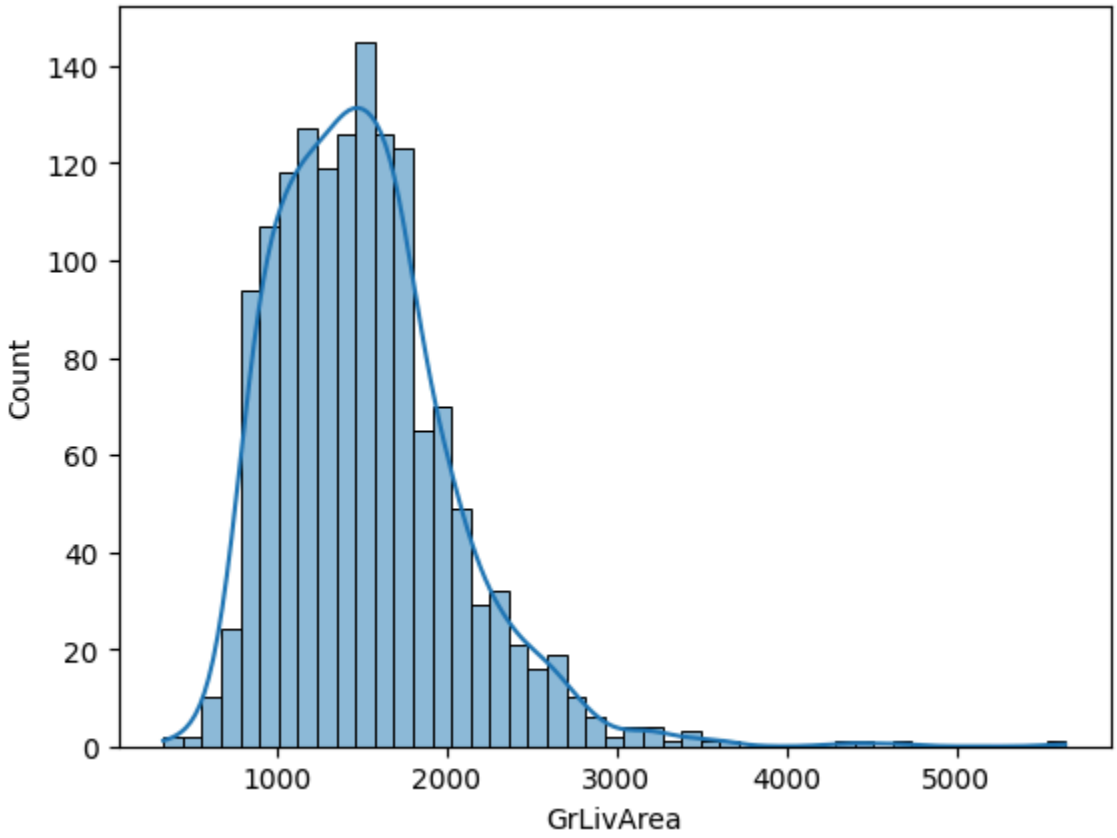
2ndFlrSF



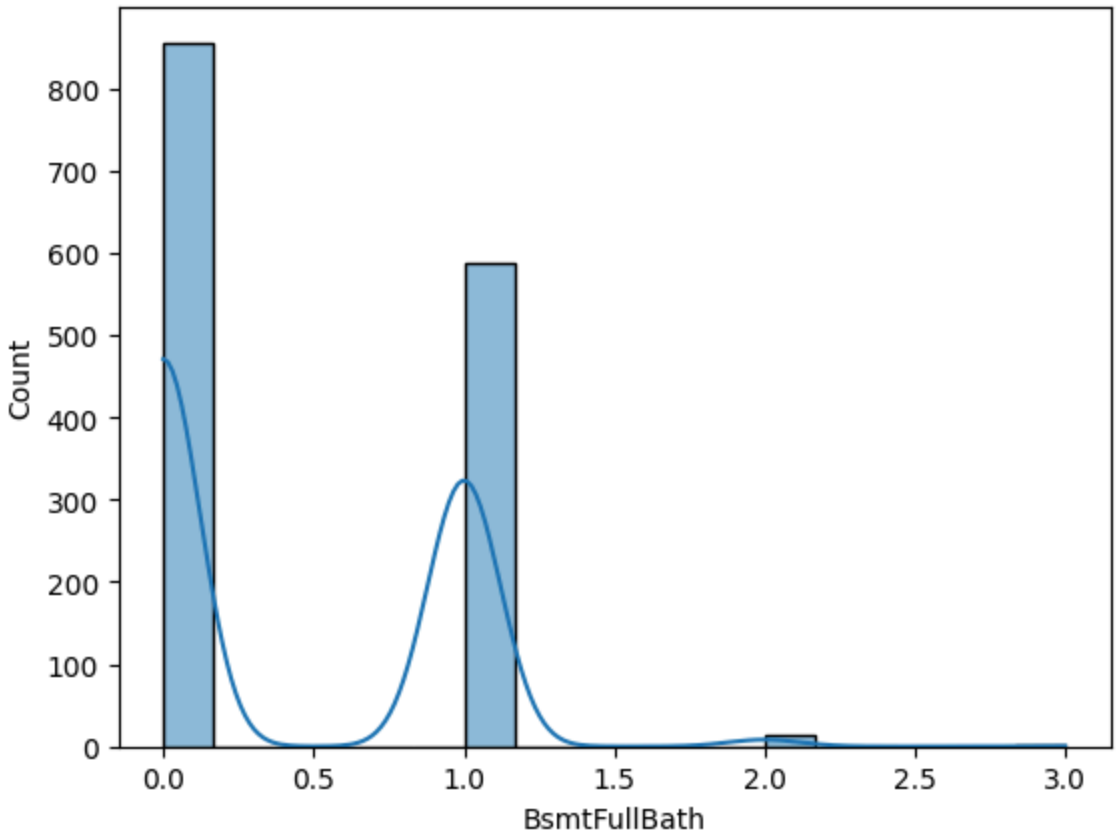
LowQualFinSF



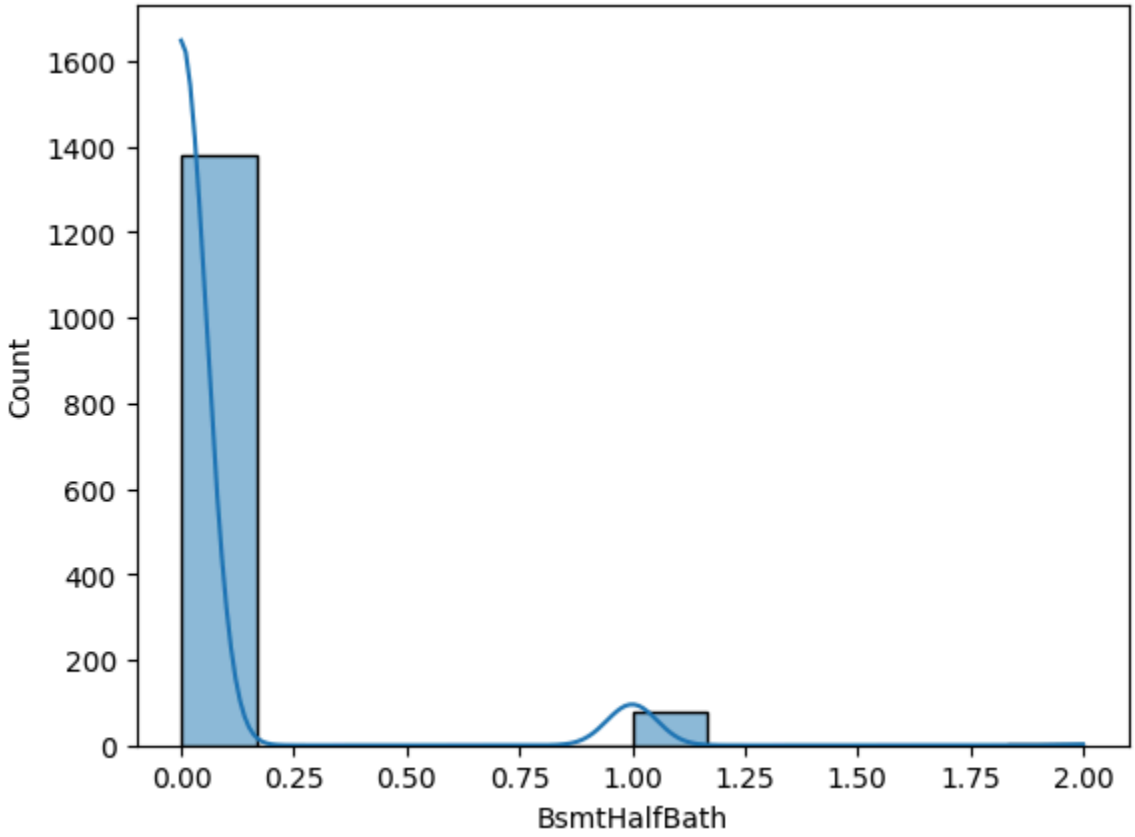
GrLivArea



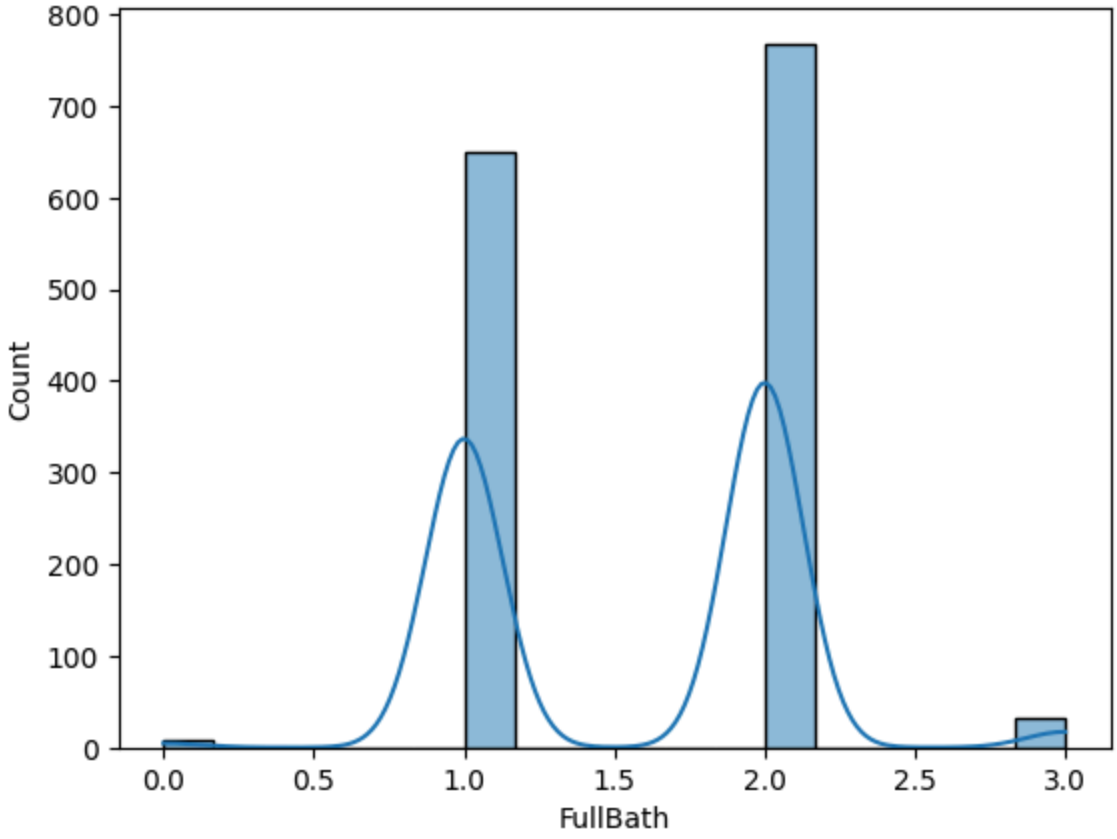
BsmtFullBath



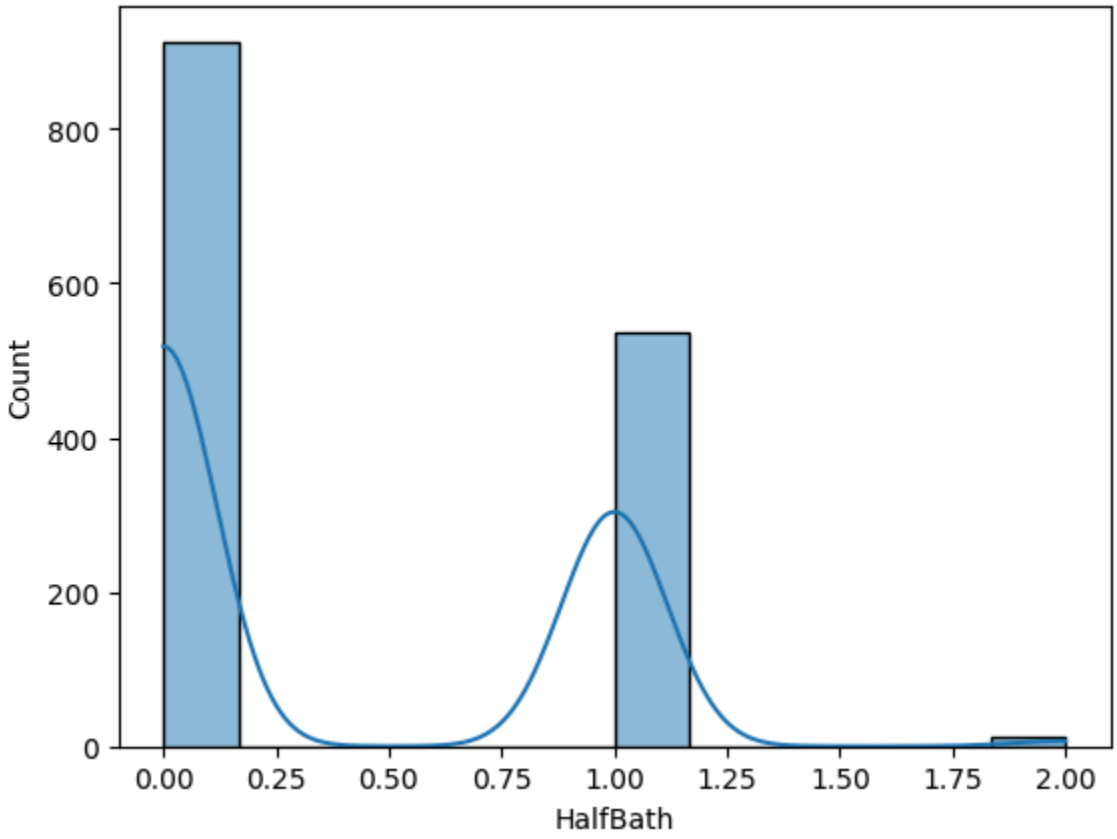
BsmtHalfBath



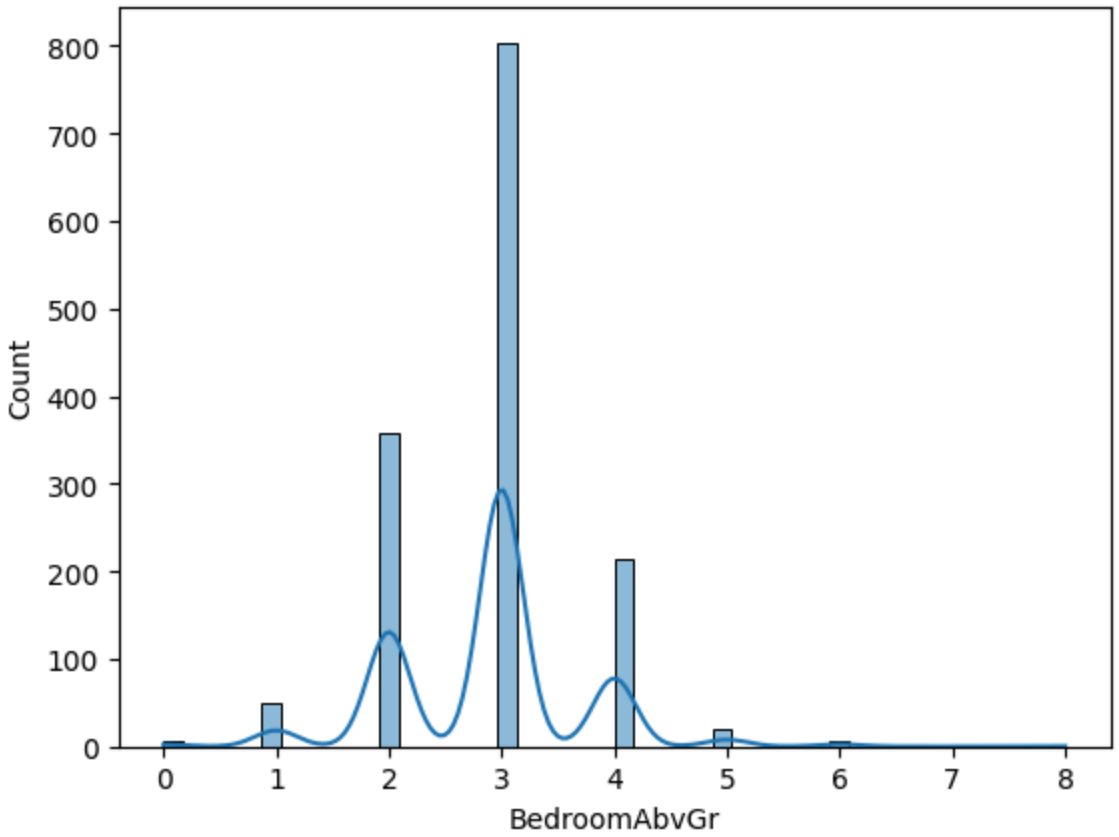
FullBath



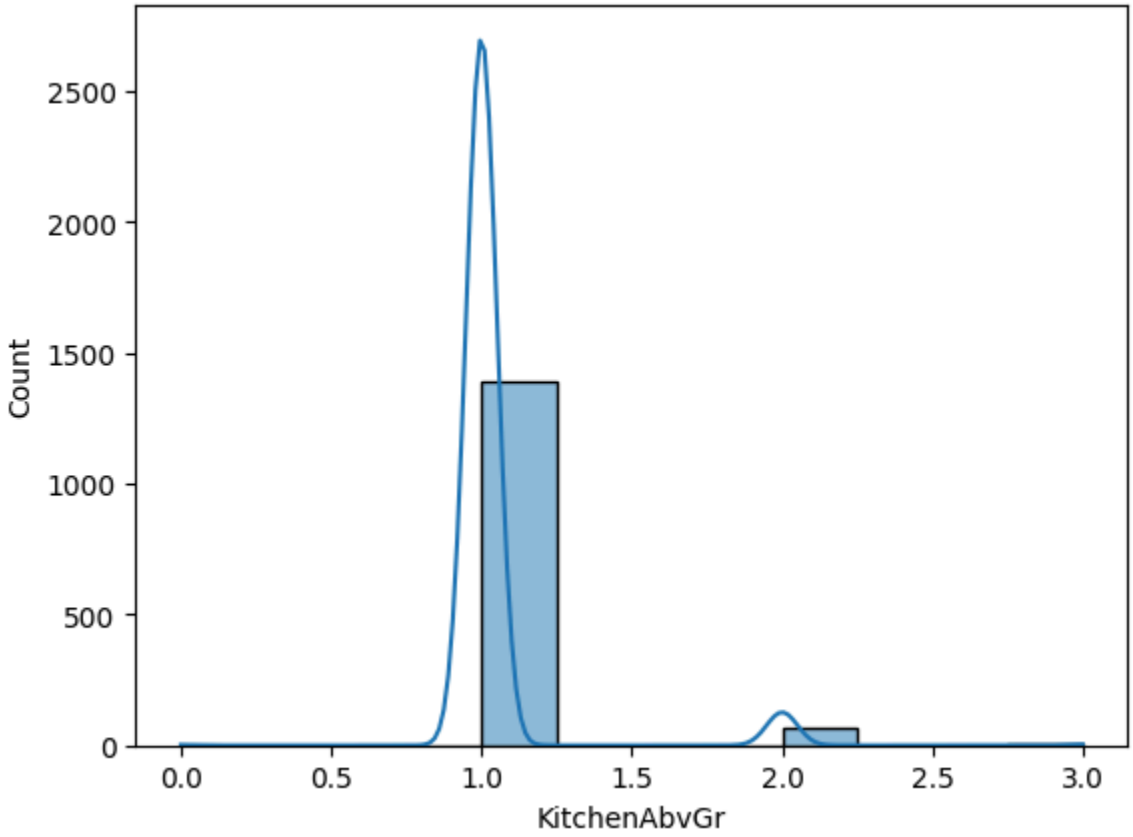
HalfBath



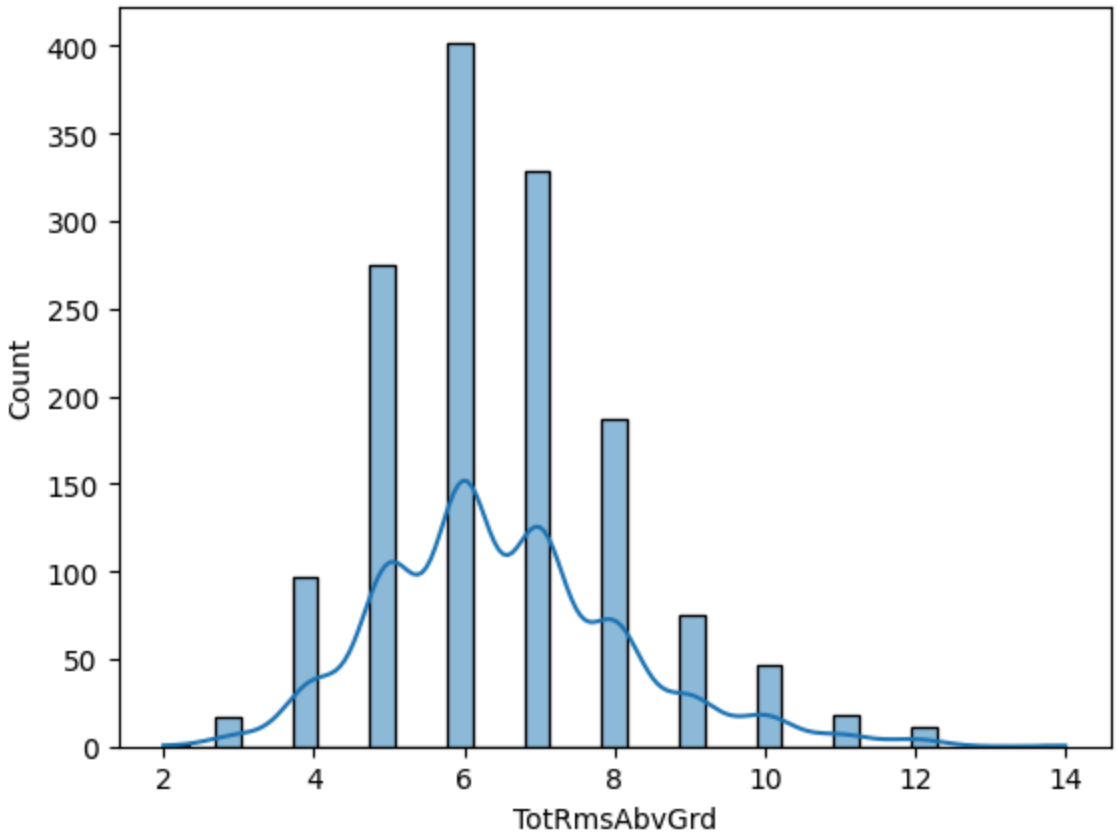
BedroomAbvGr



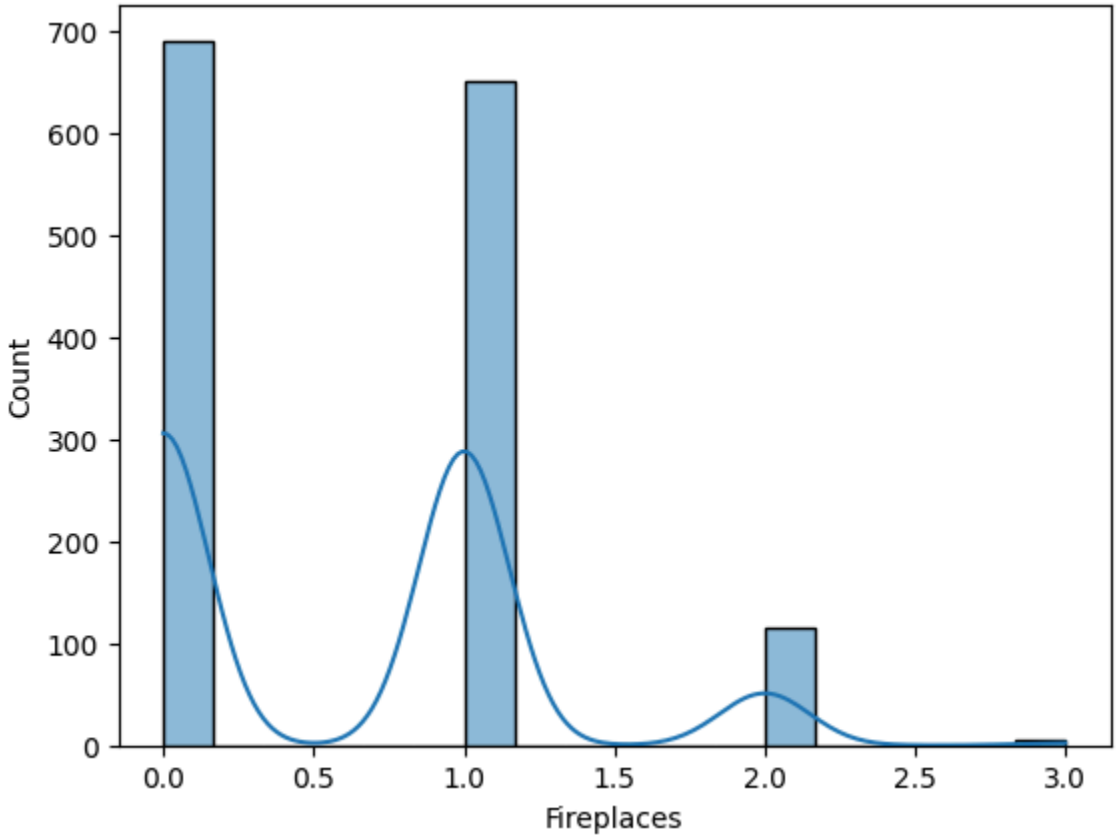
KitchenAbvGr



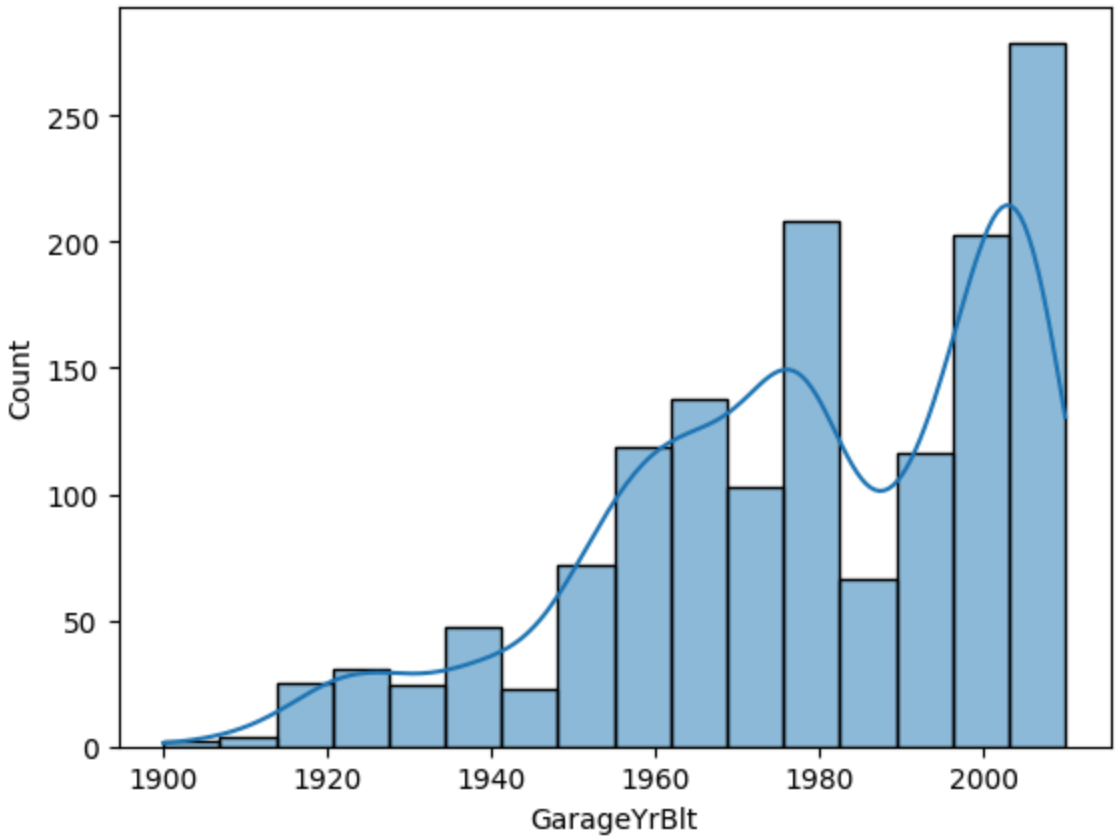
TotRmsAbvGrd



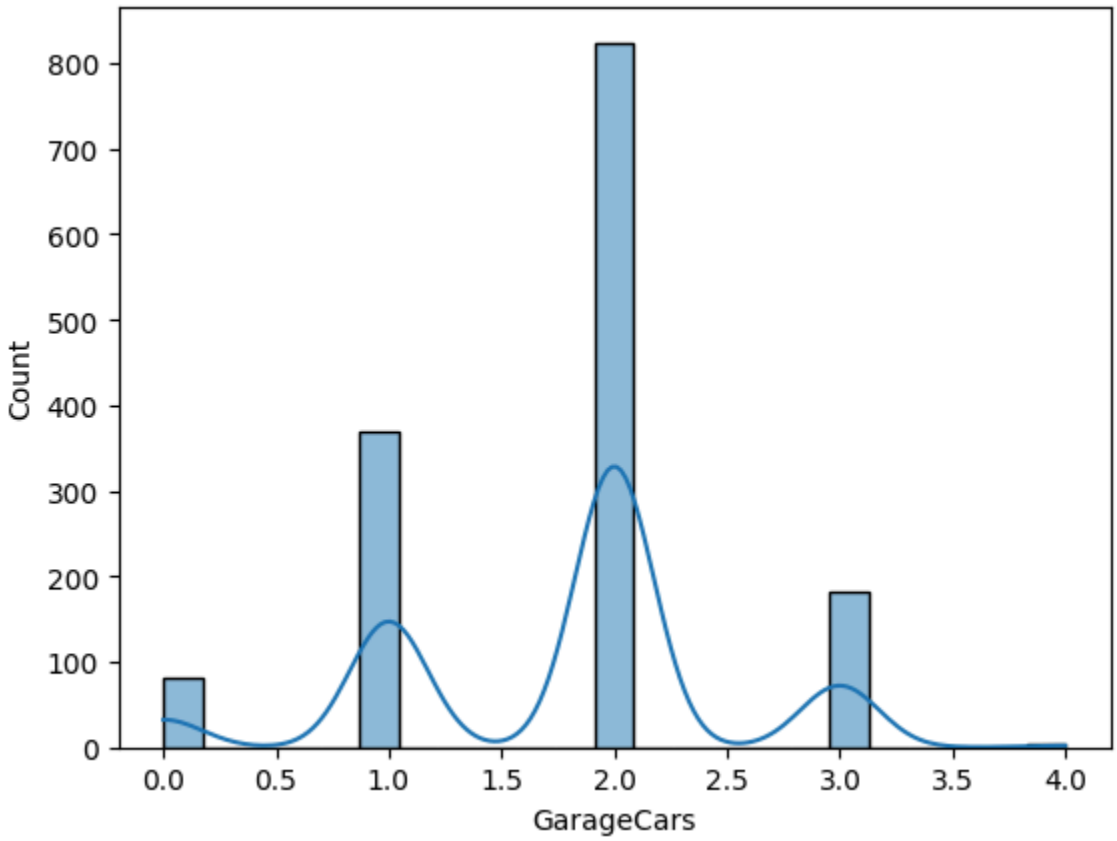
Fireplaces



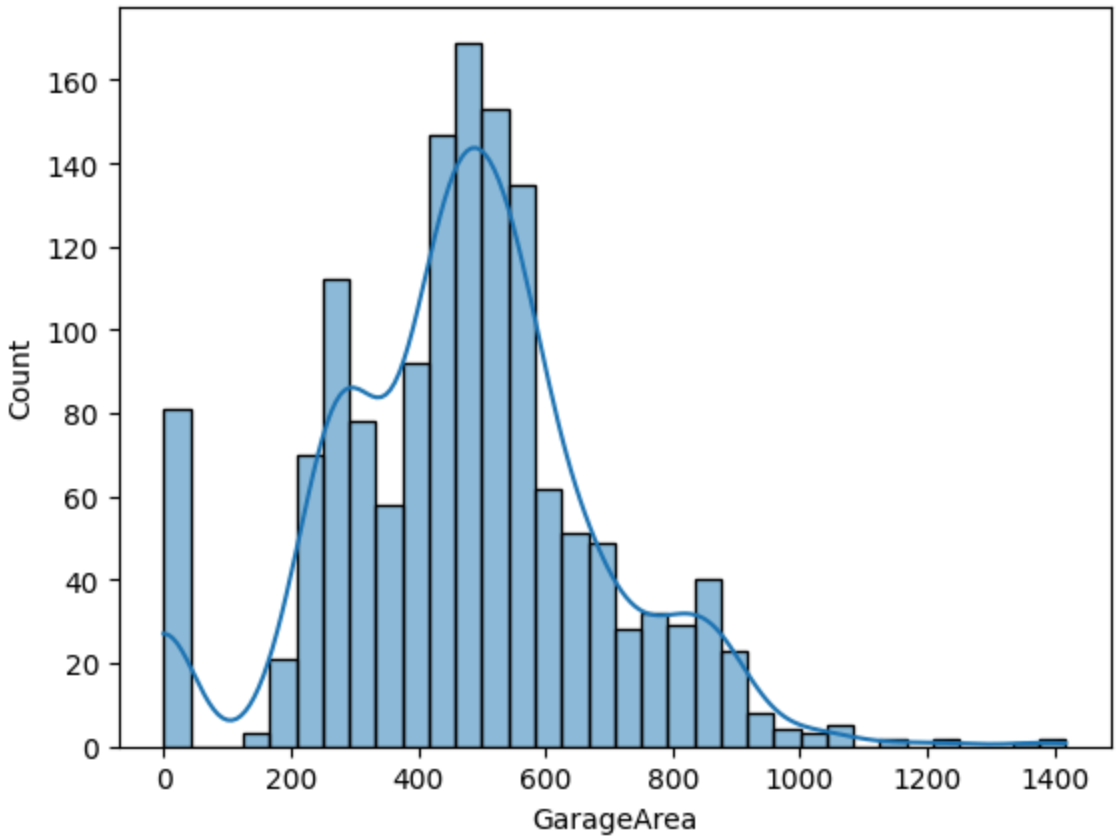
GarageYrBlt



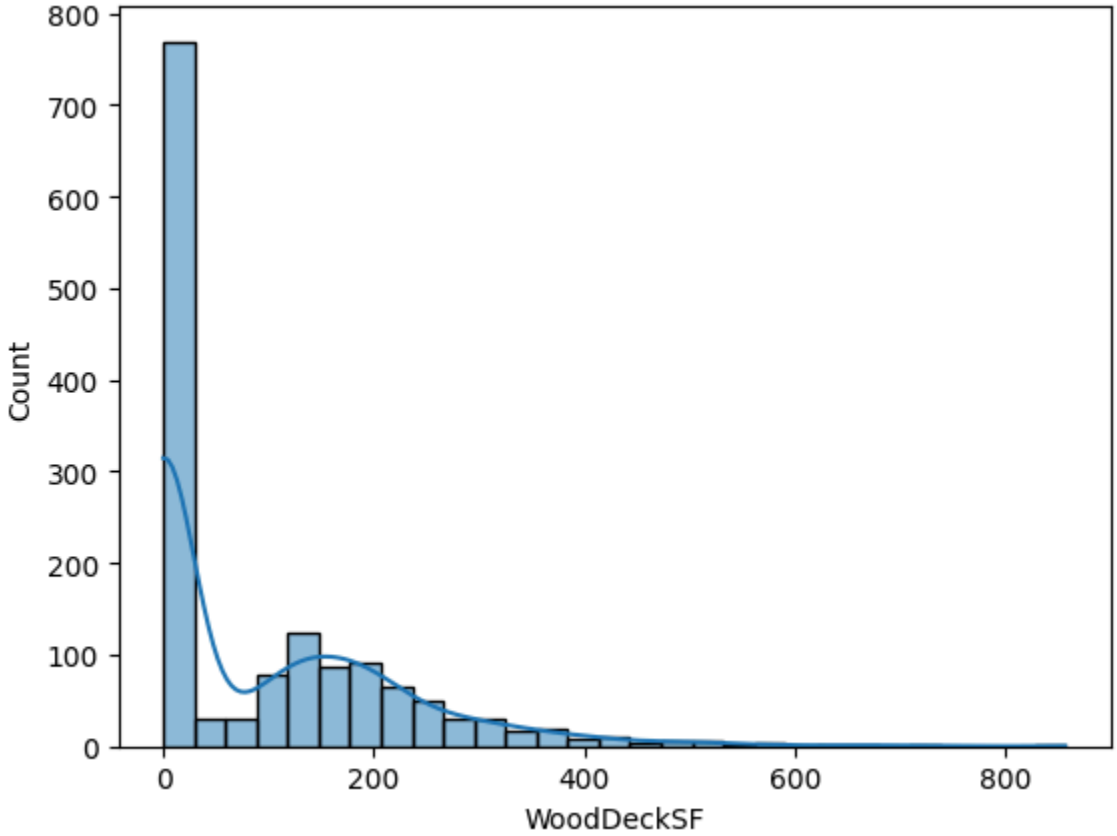
GarageCars



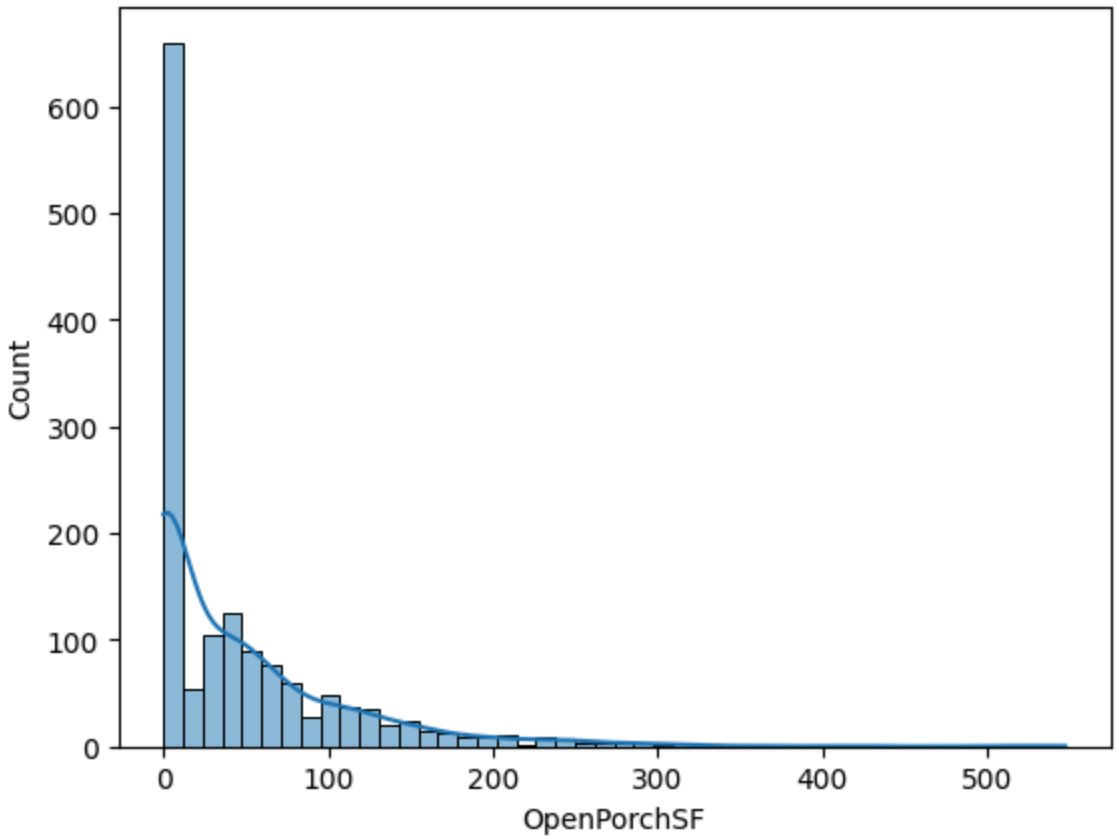
GarageArea



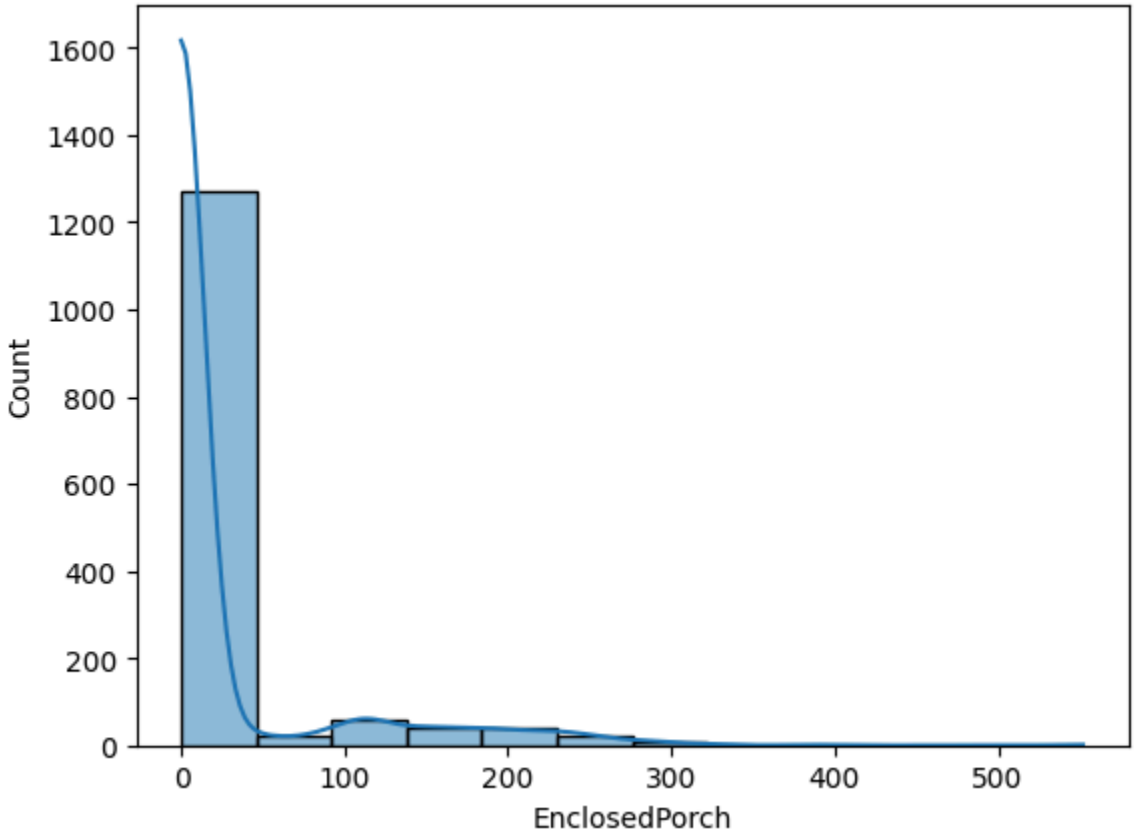
WoodDeckSF



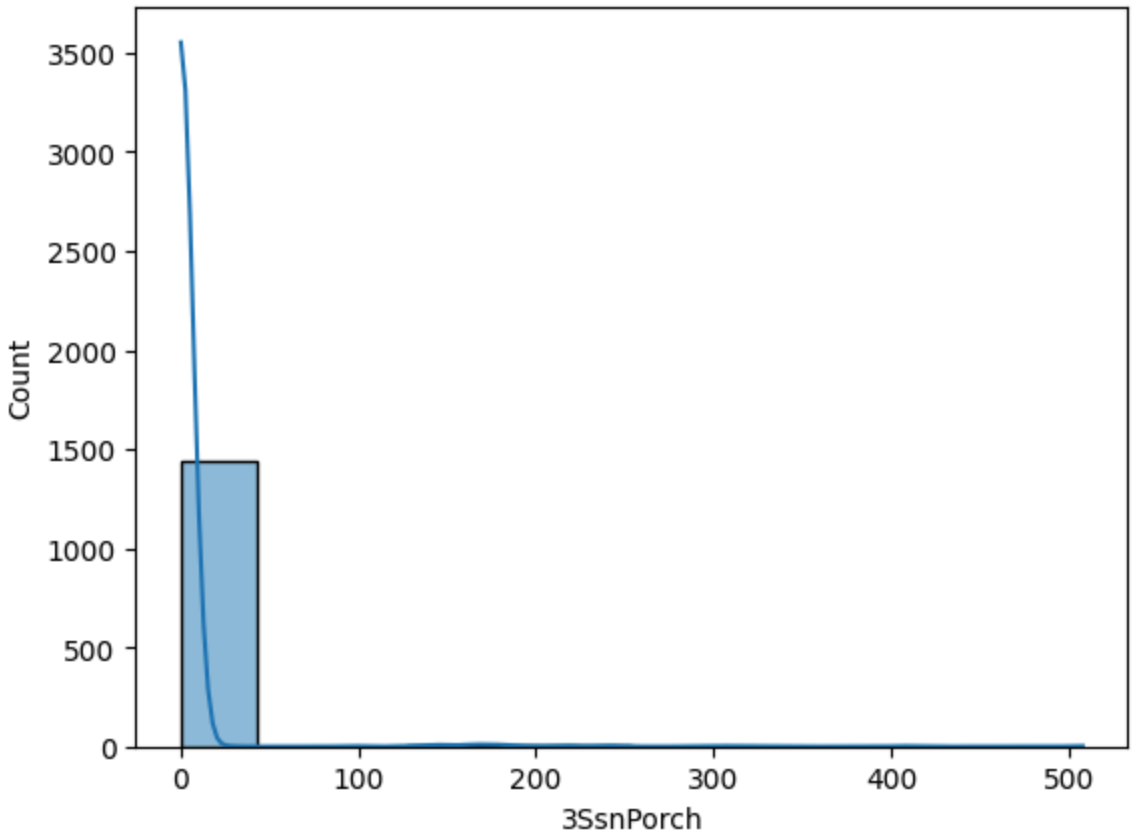
OpenPorchSF



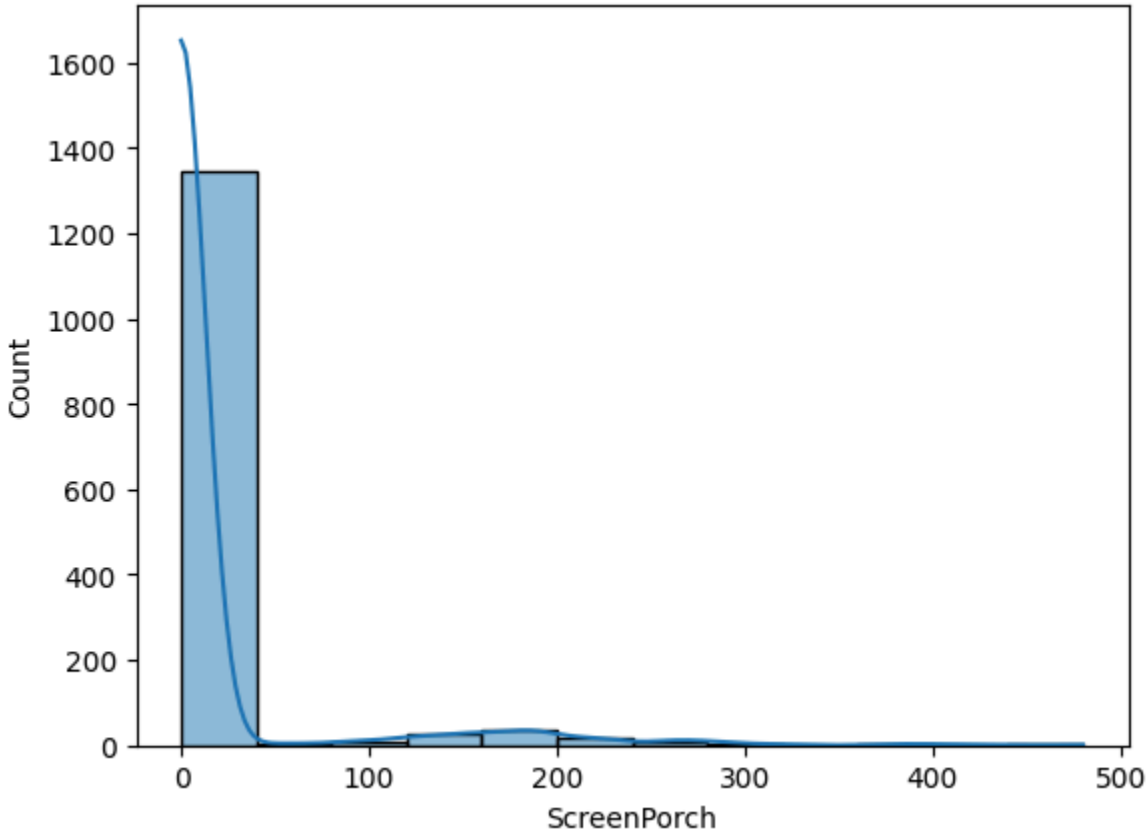
EnclosedPorch



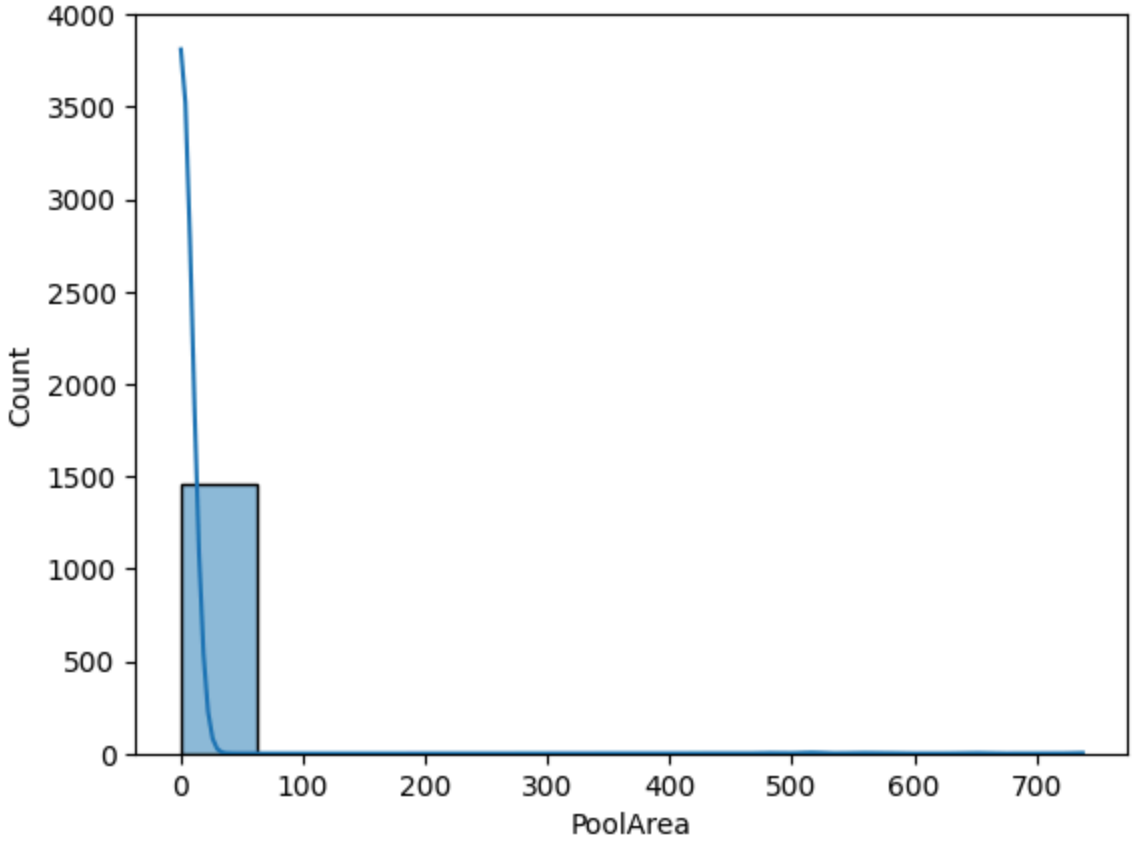
3SsnPorch



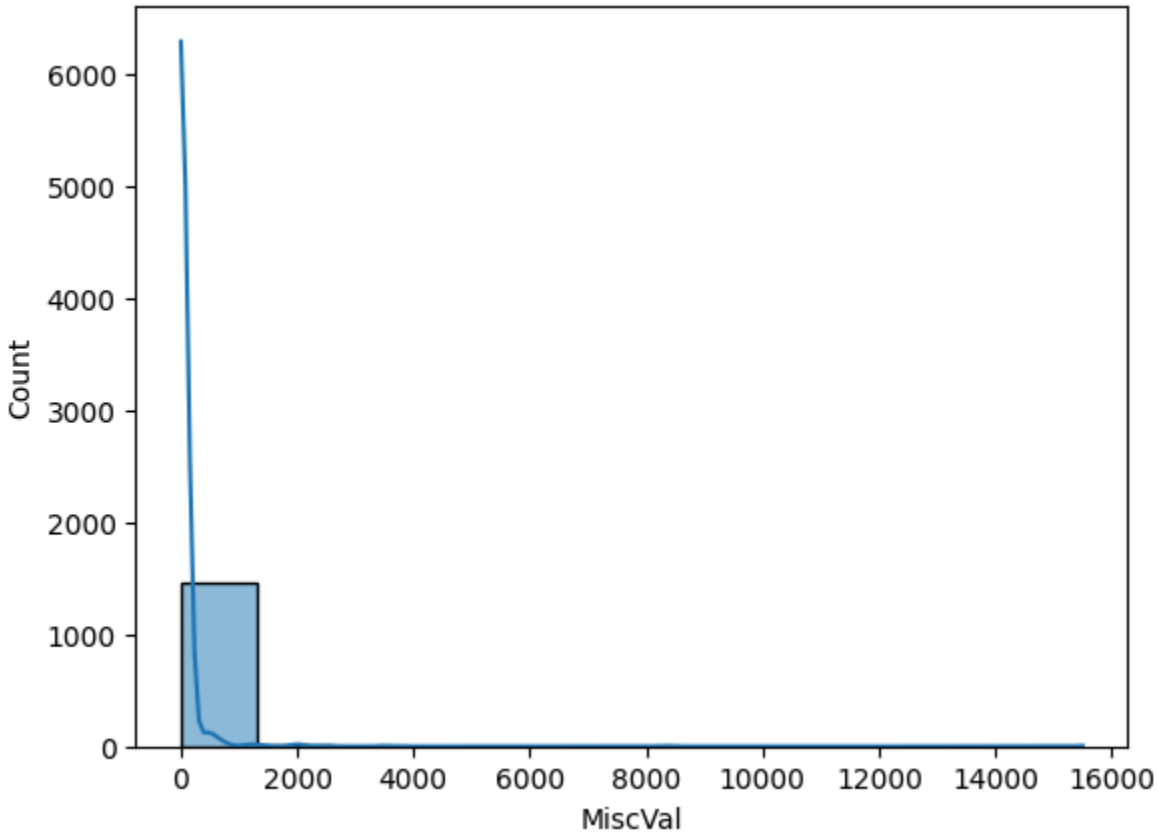
ScreenPorch



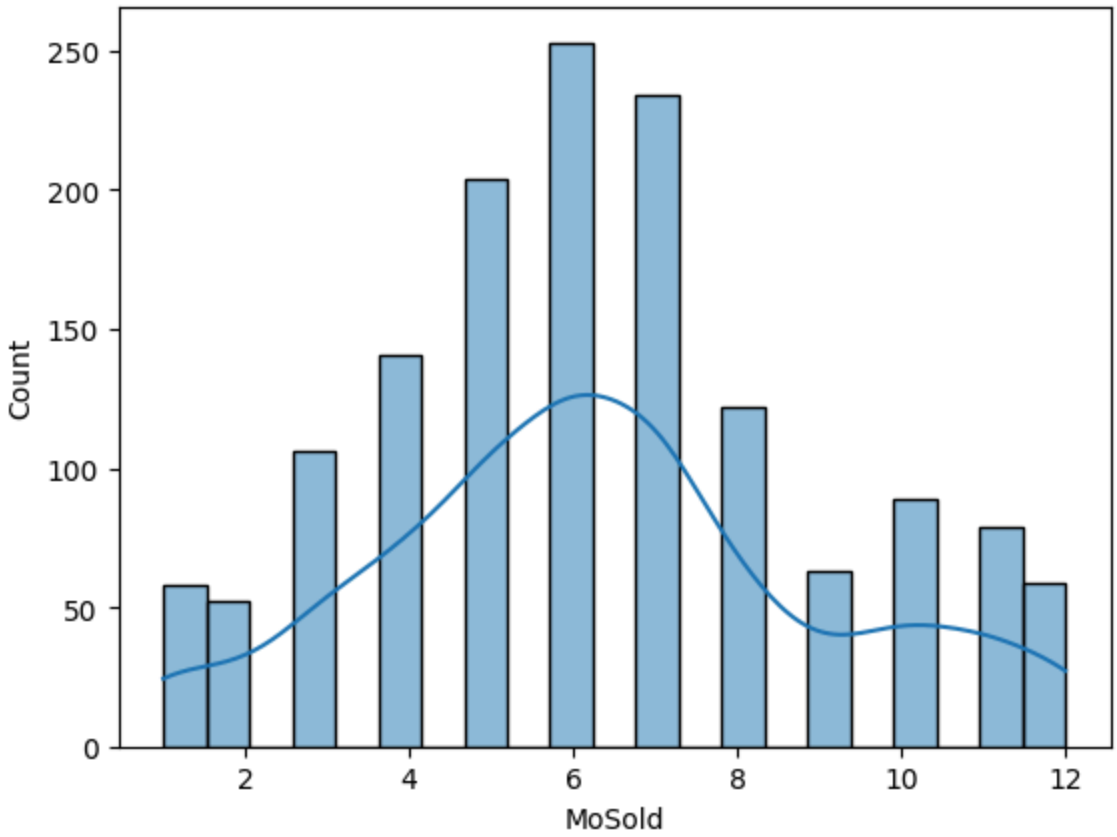
PoolArea



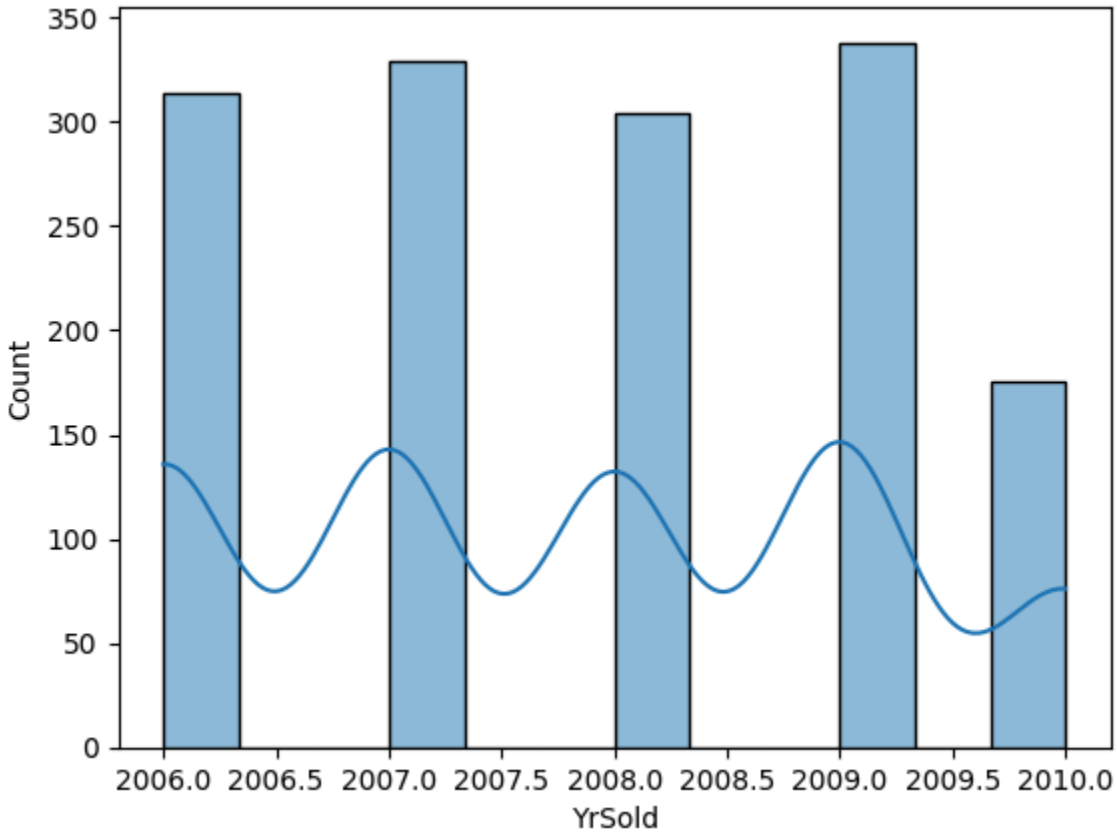
MiscVal



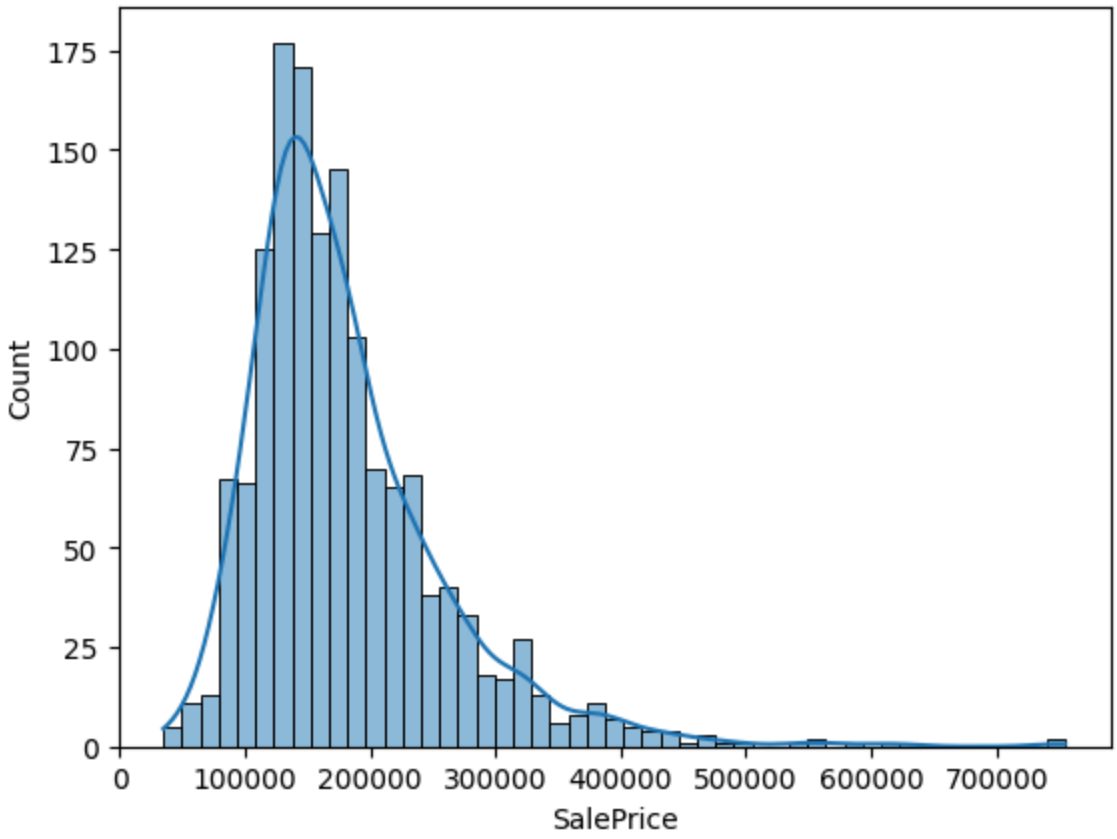
MoSold



YrSold



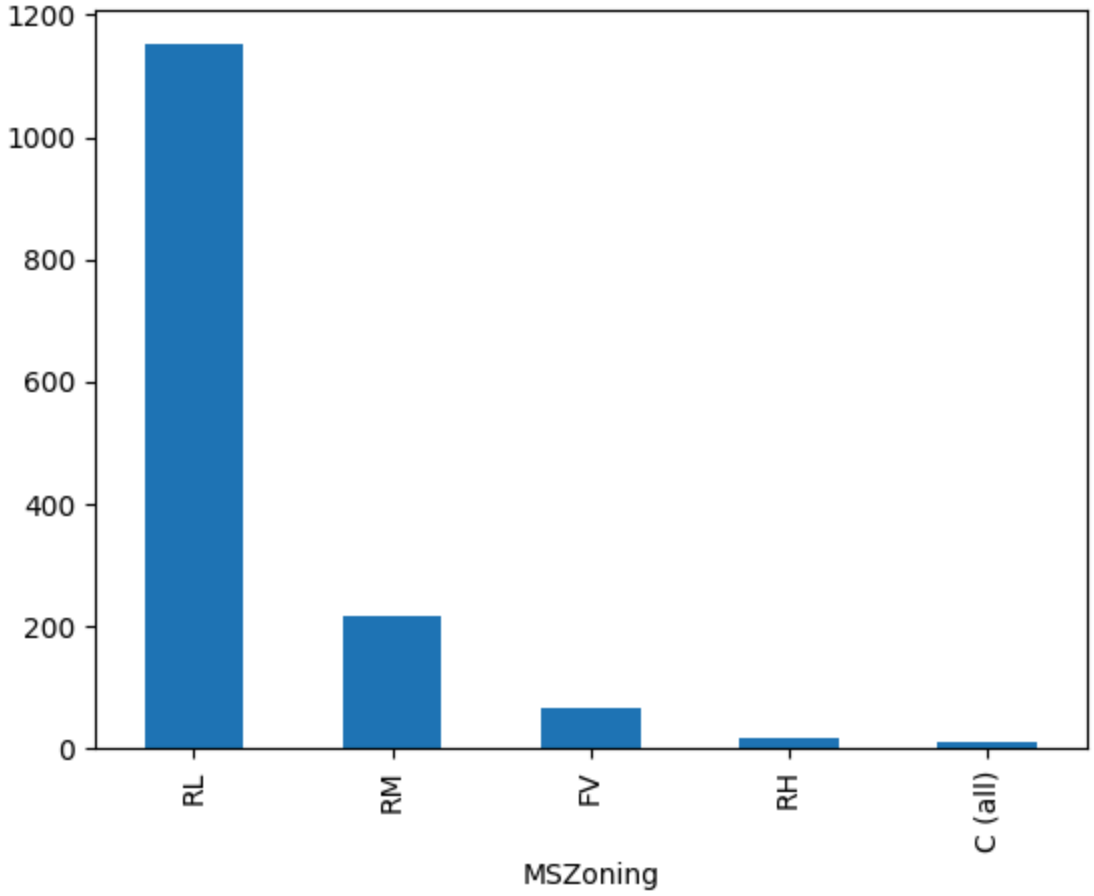
SalePrice



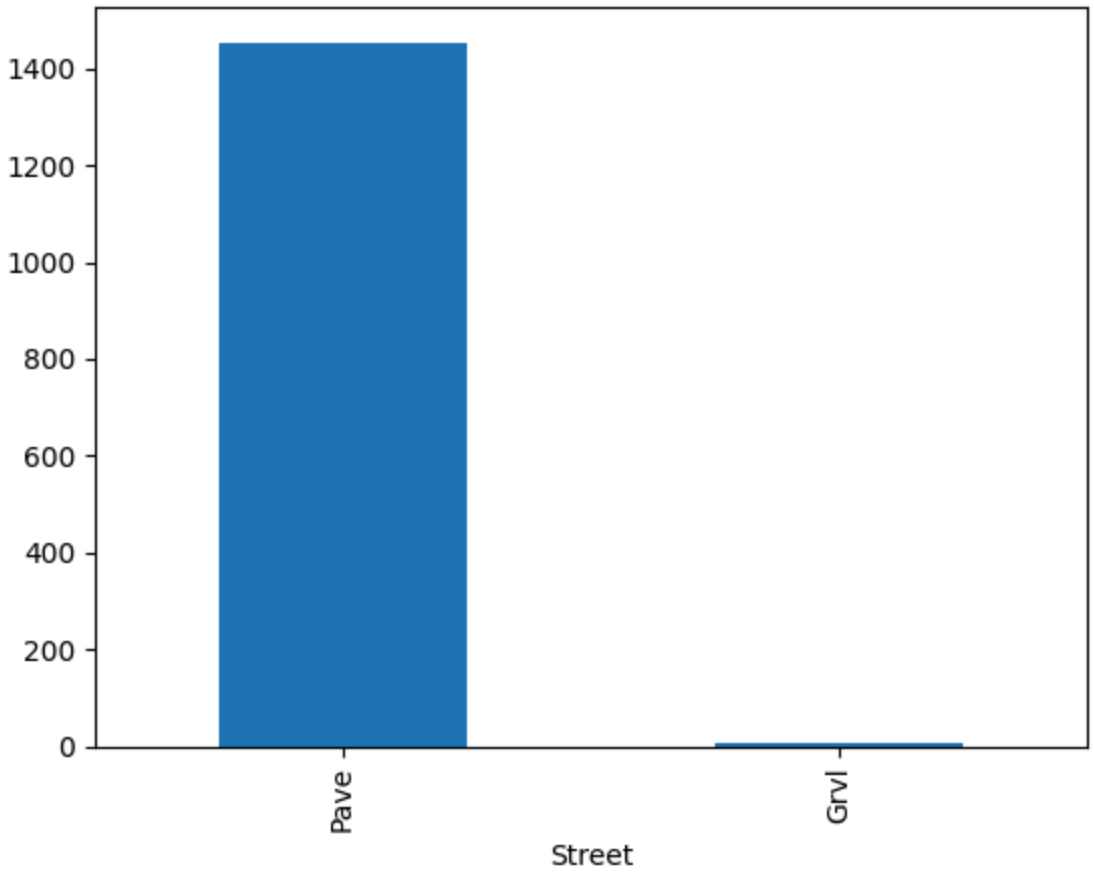
In [318…

```
for i in df[cat]:
  df[i].value_counts().plot(kind='bar')
  print(i)
  show()
```

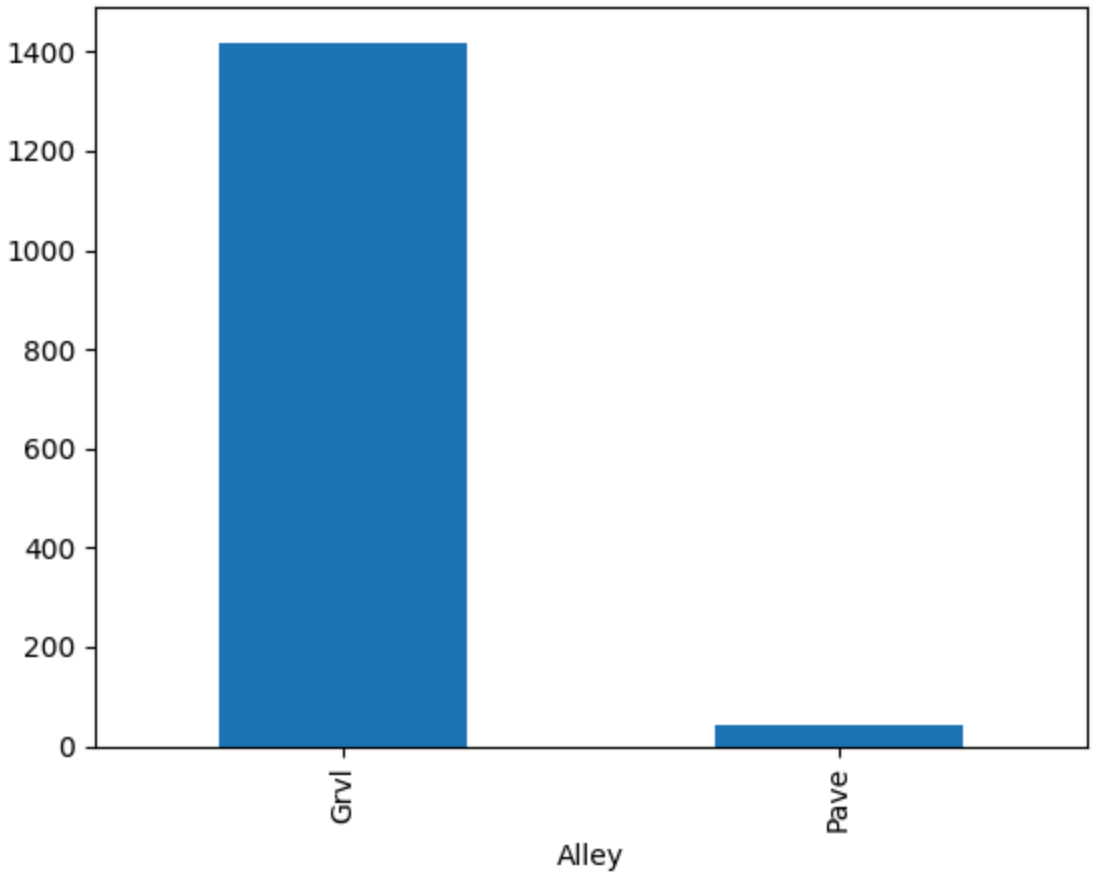
```
MSZoning
```



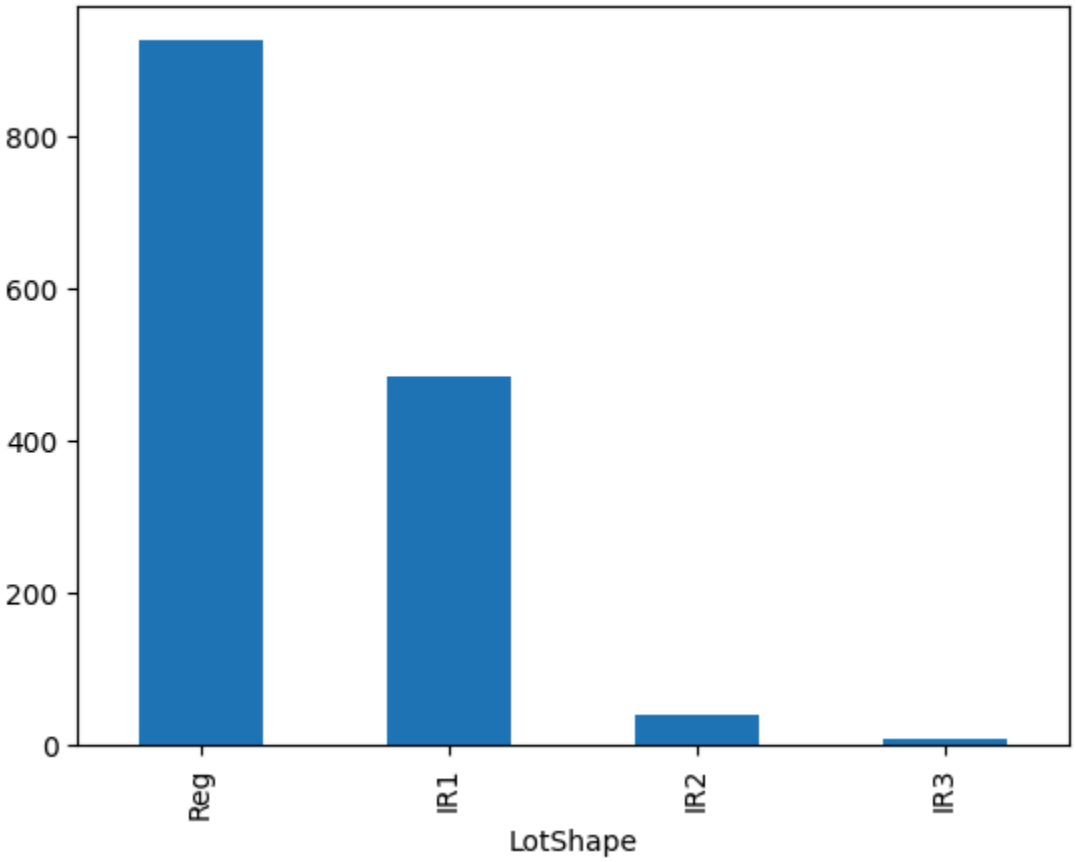
Street



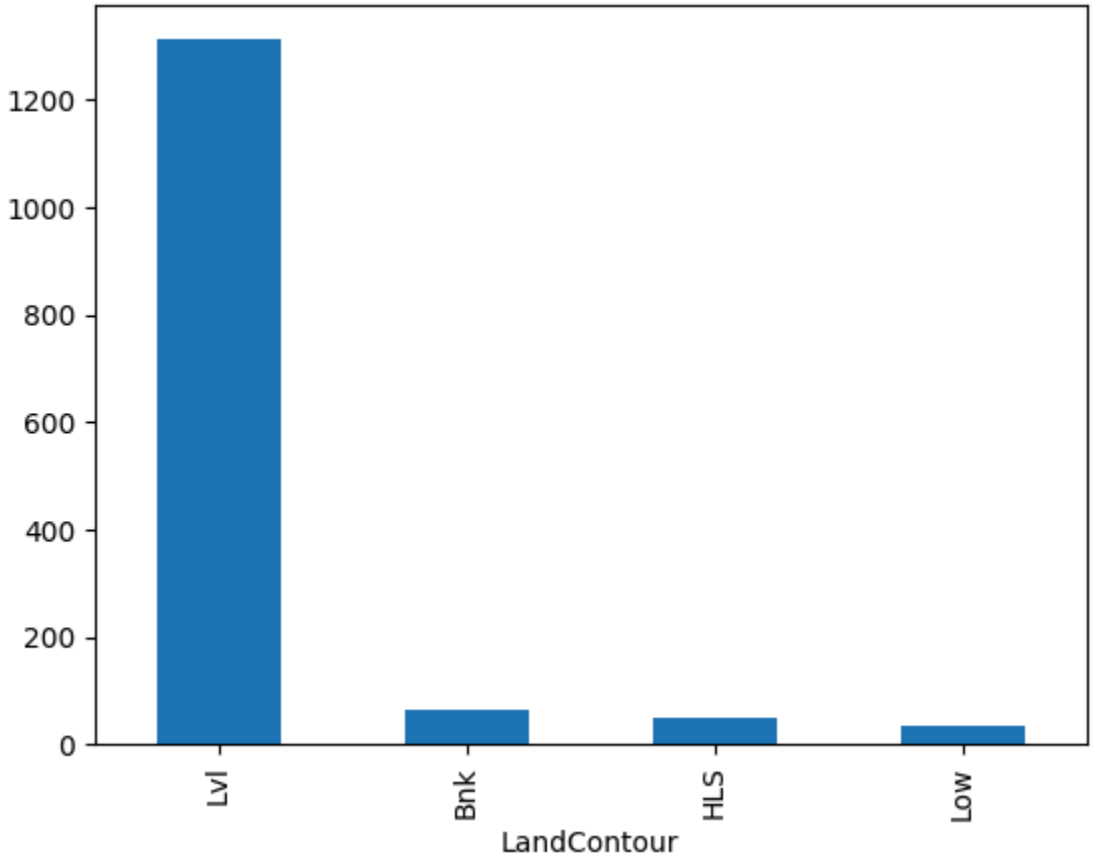

```
Alley
```


1400
1200
1000
800
600
400
200
0
Grvl
Pave
Alley

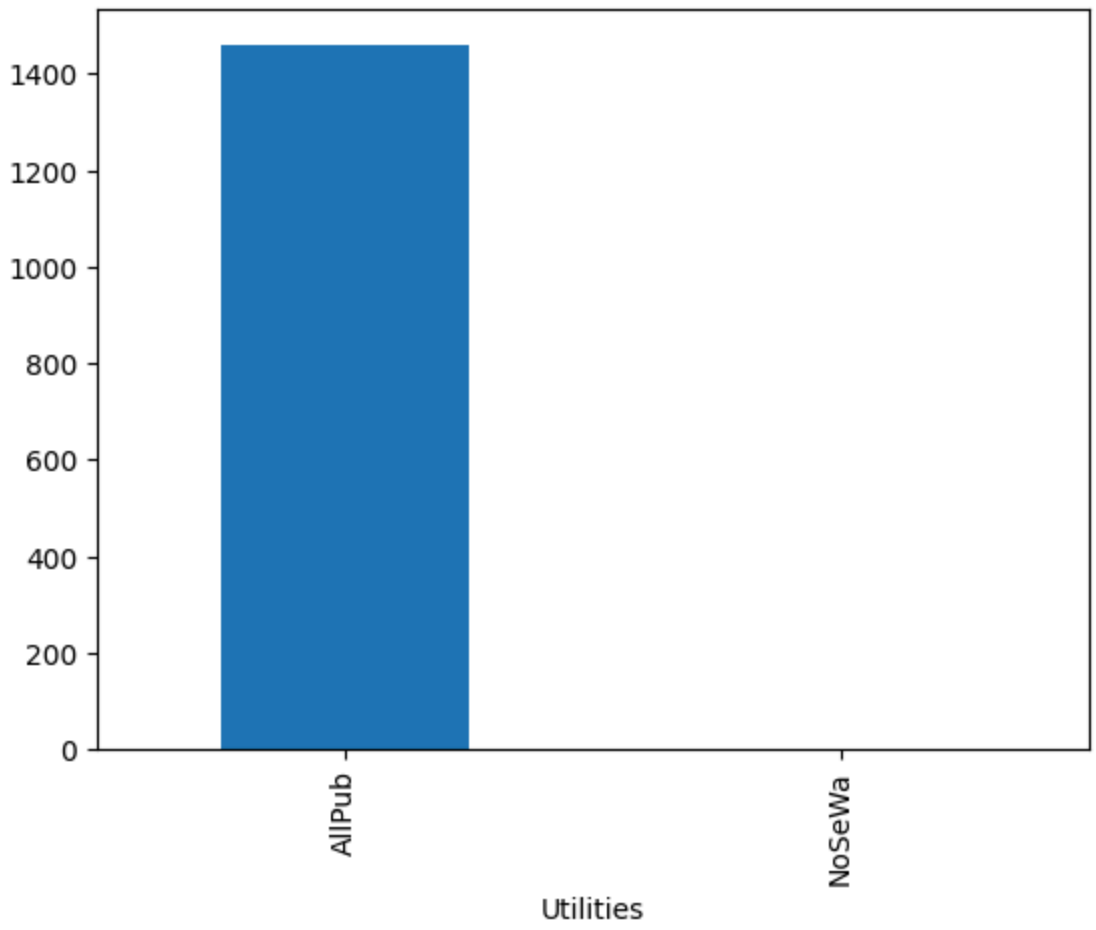
LotShape


800
600
400
200
0
Reg
IR1
IR2
IR3
LotShape

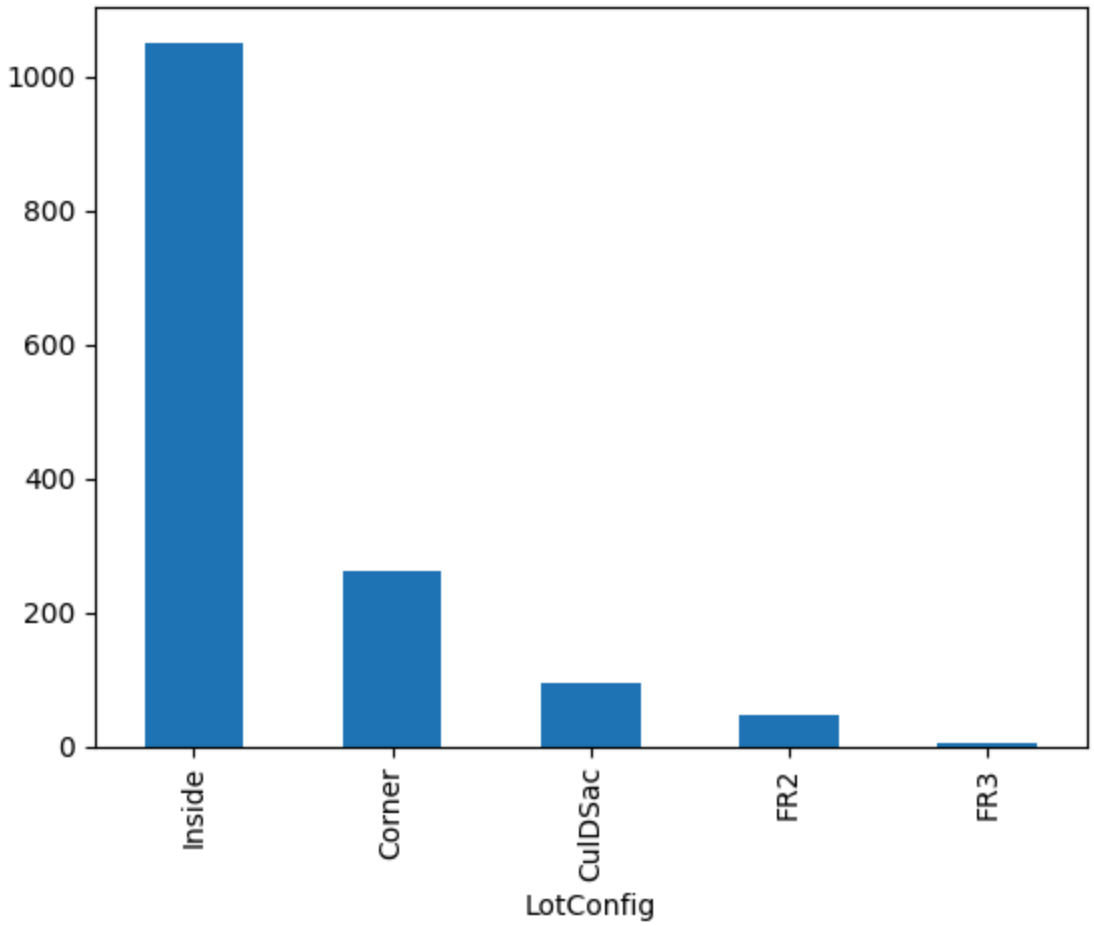
LandContour



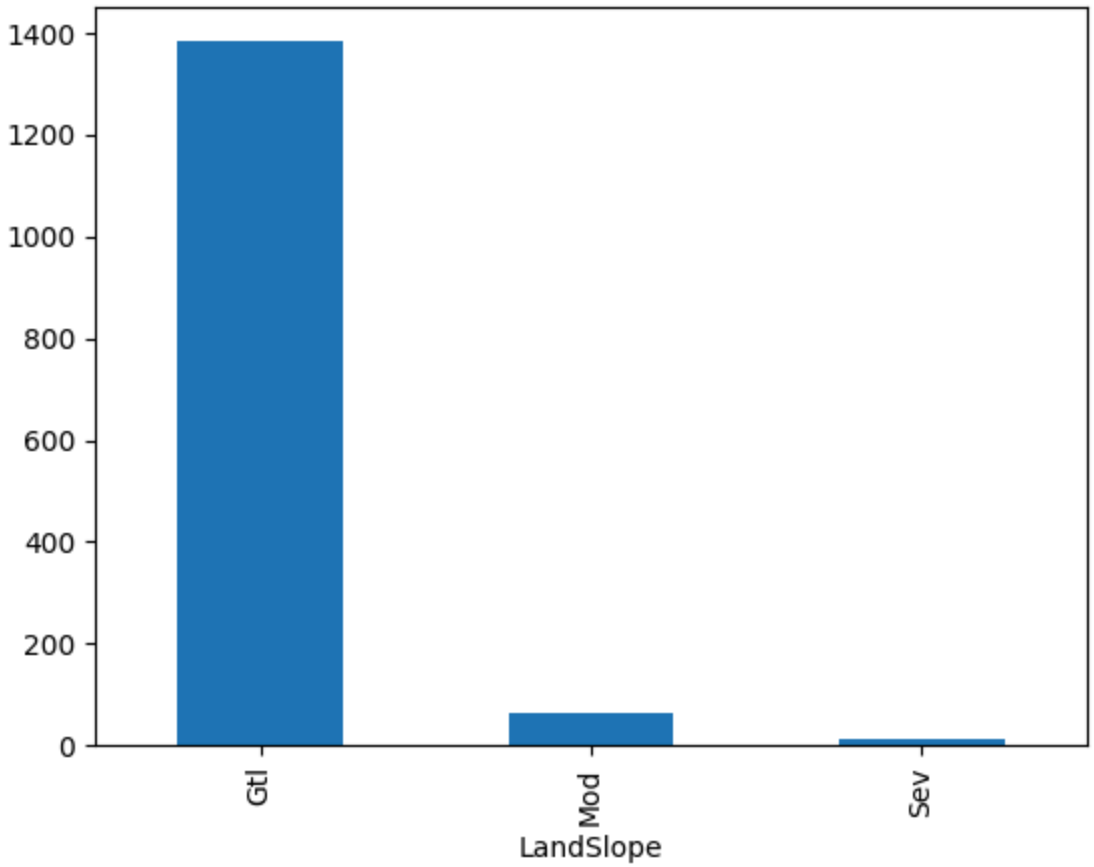
Utilities



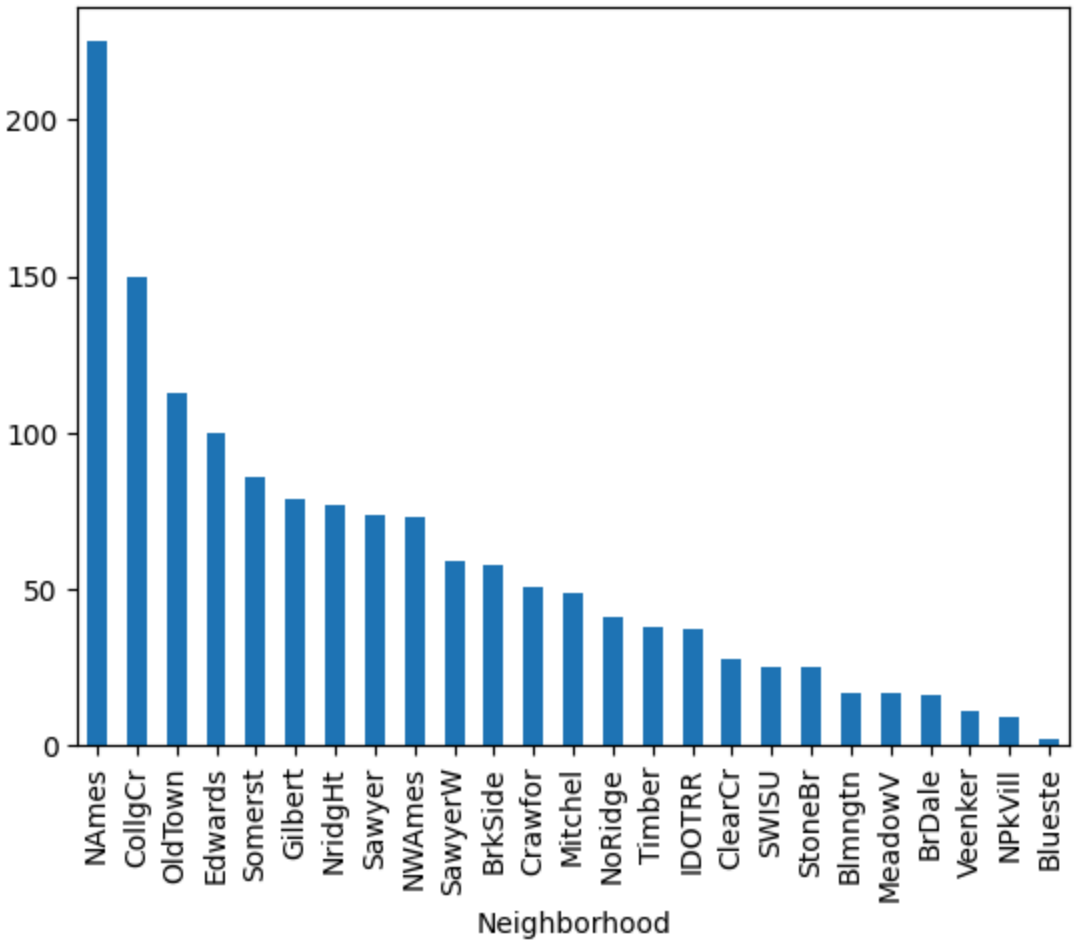
LotConfig


1000
800
600
400
200
0
Inside
Corner
CulDSac
FR2
FR3
LotConfig

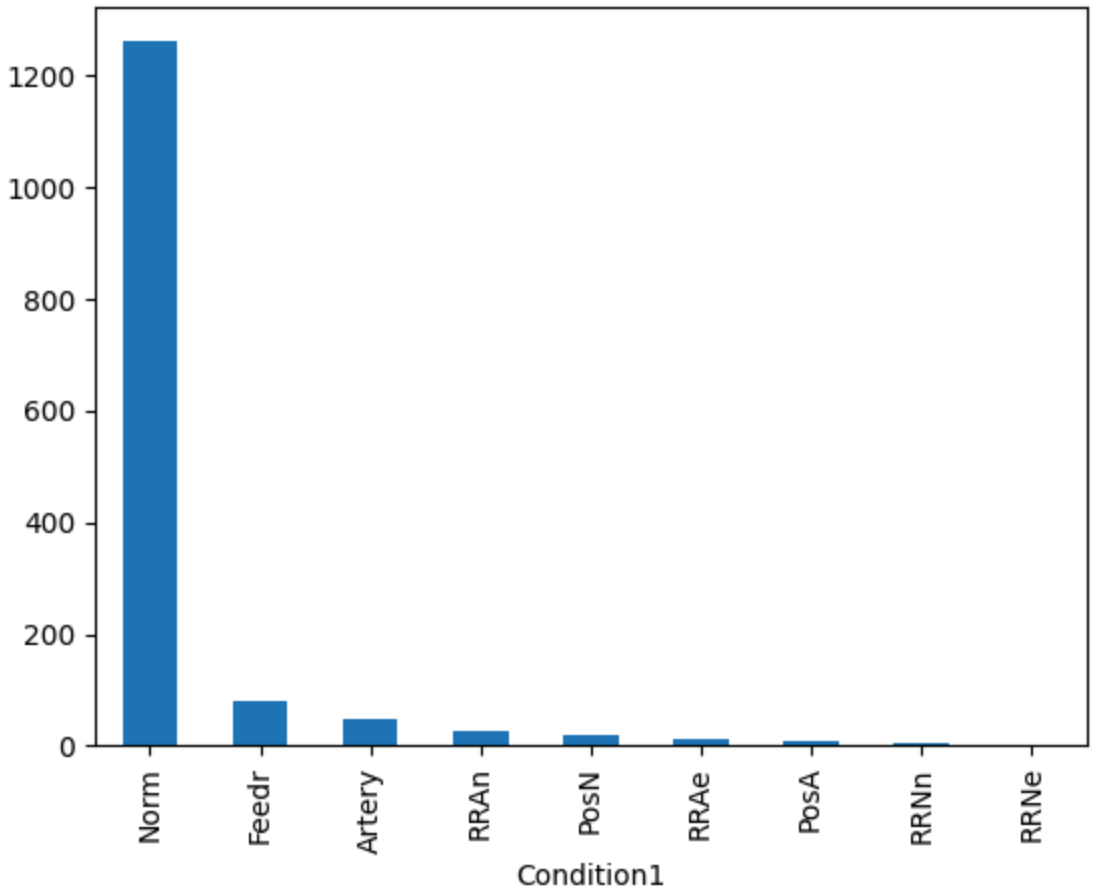
LandSlope



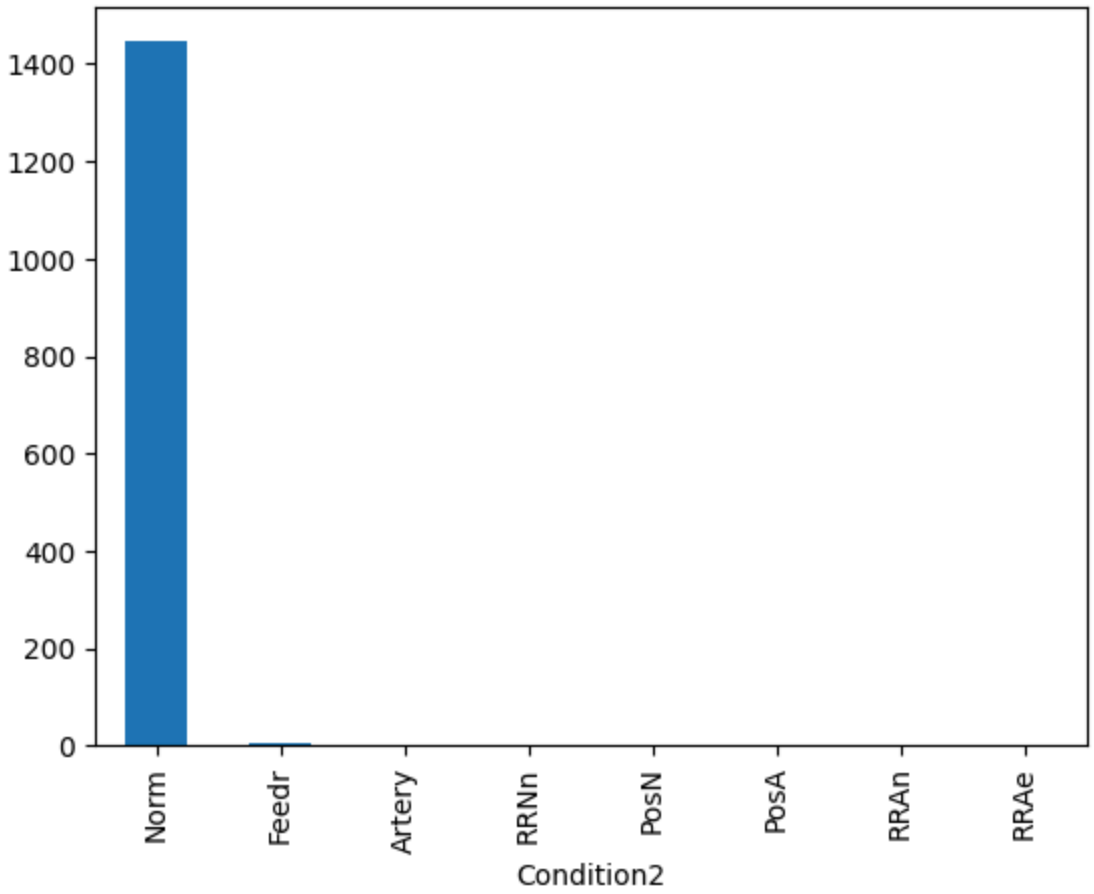

```
Neighborhood
```


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

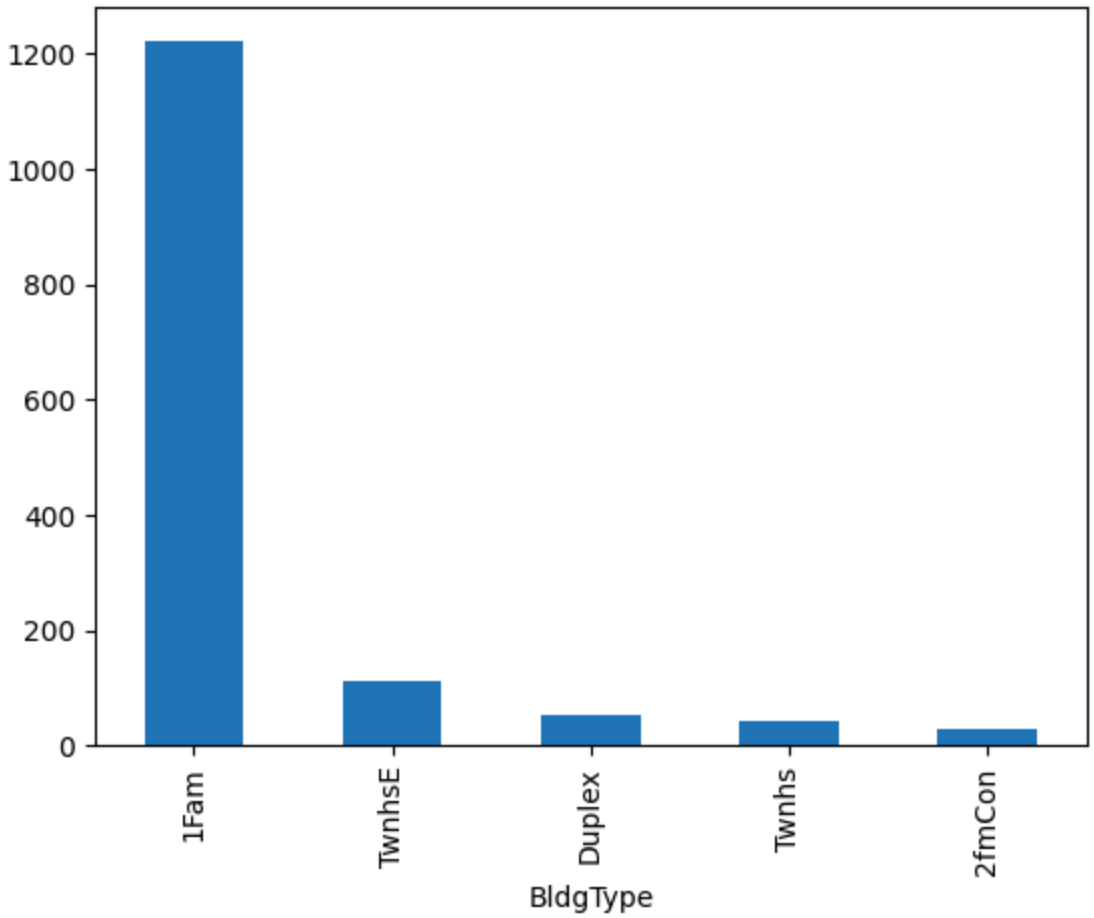
```
Condition1
```


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

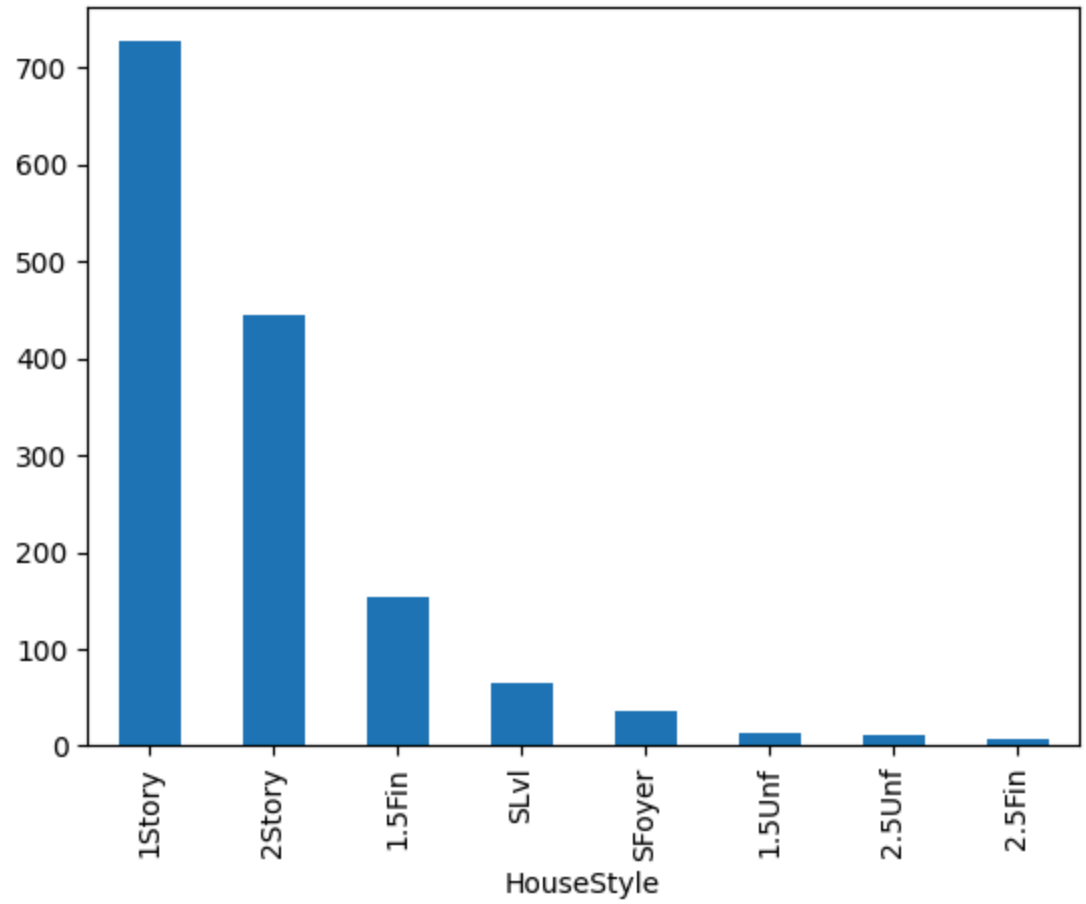
Condition2


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

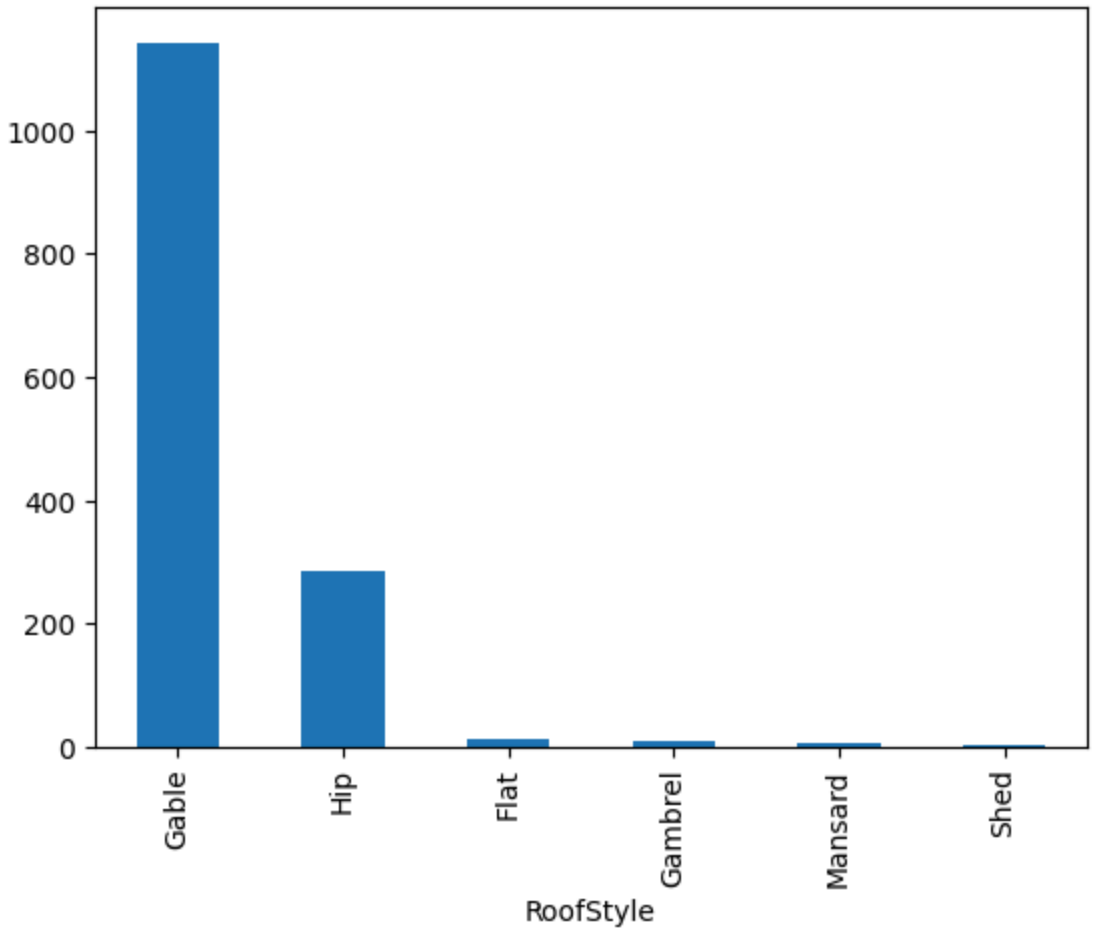
BldgType


1200
1000
800
600
400
200
0
1Fam
TwnhsE
Duplex
Twnhs
2fmCon
BldgType

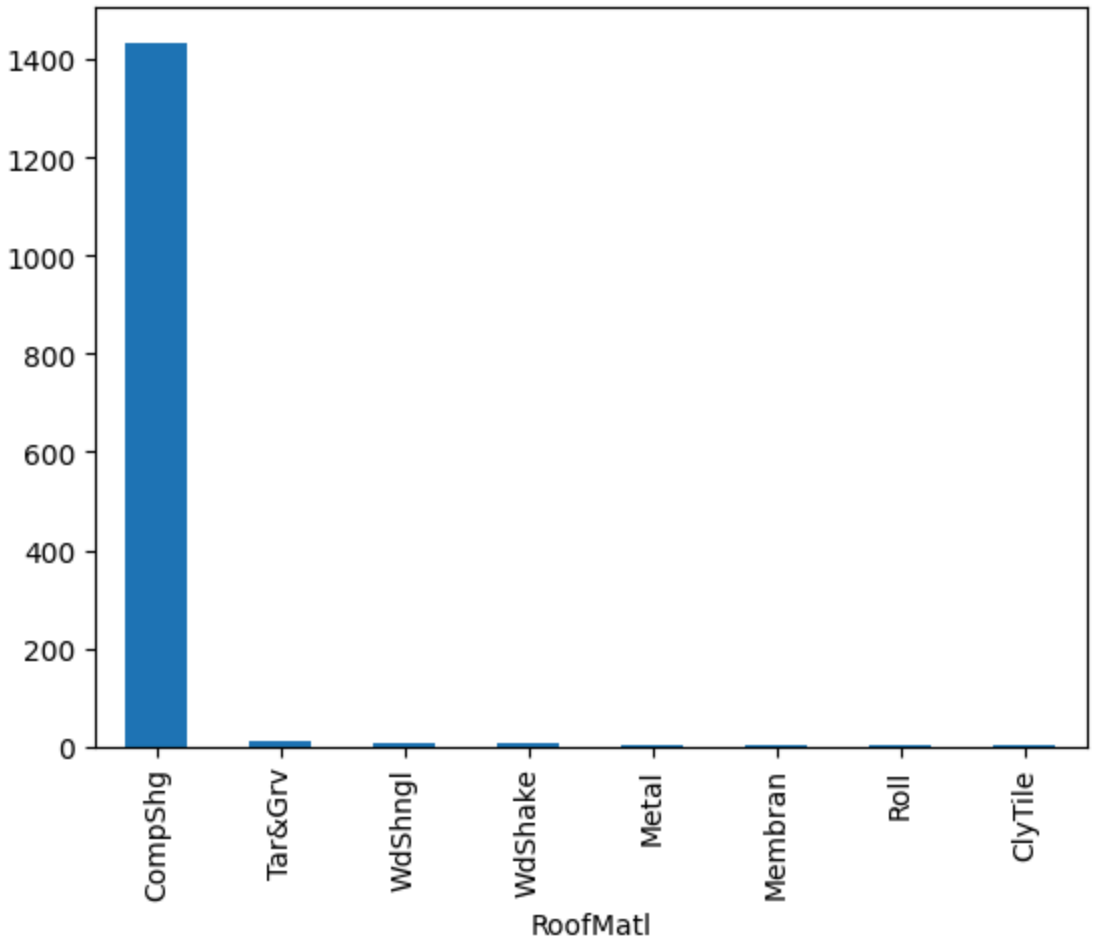
HouseStyle


700
600
500
400
300
200
100
0
1Story
2Story
1.5Fin
SLvl
SFoyer
1.5Unf
2.5Unf
2.5Fin
HouseStyle

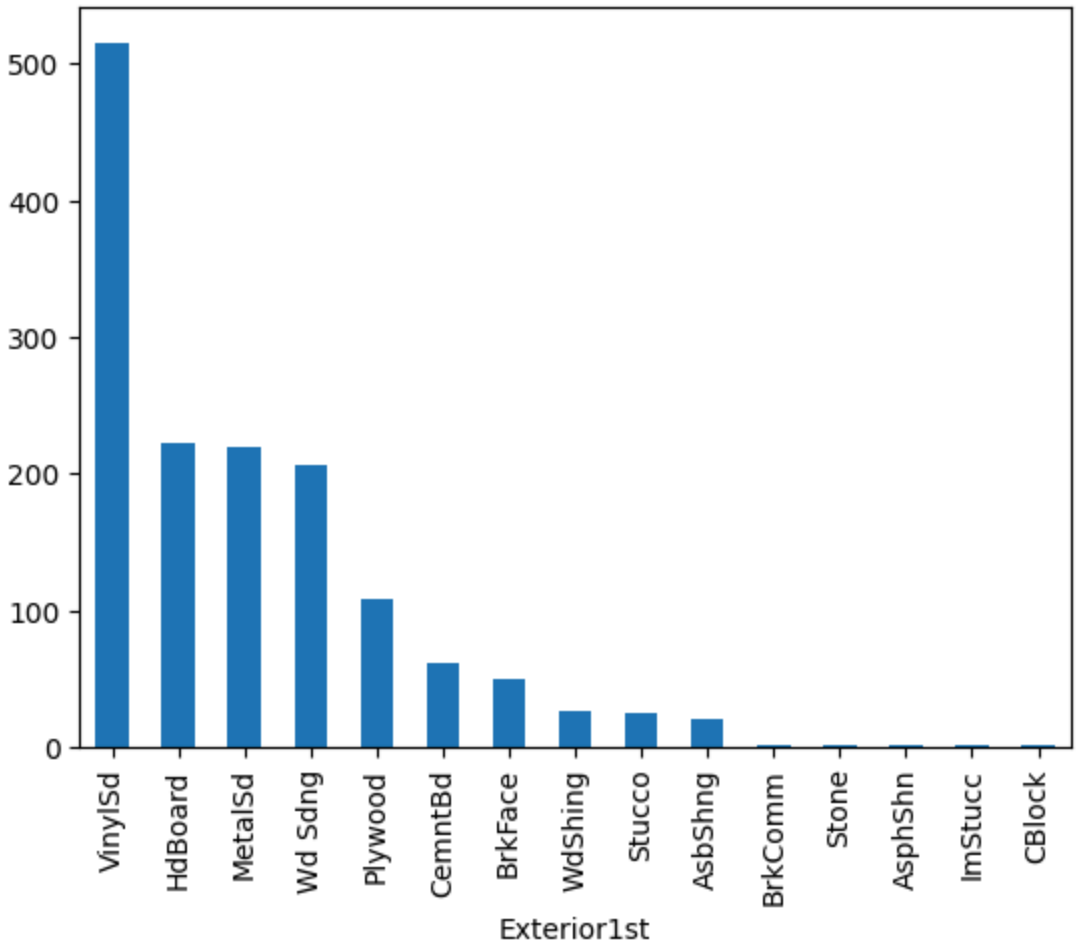
RoofStyle


1000
800
600
400
200
0
Gable
Hip
Flat
Gambrel
Mansard
Shed
RoofStyle

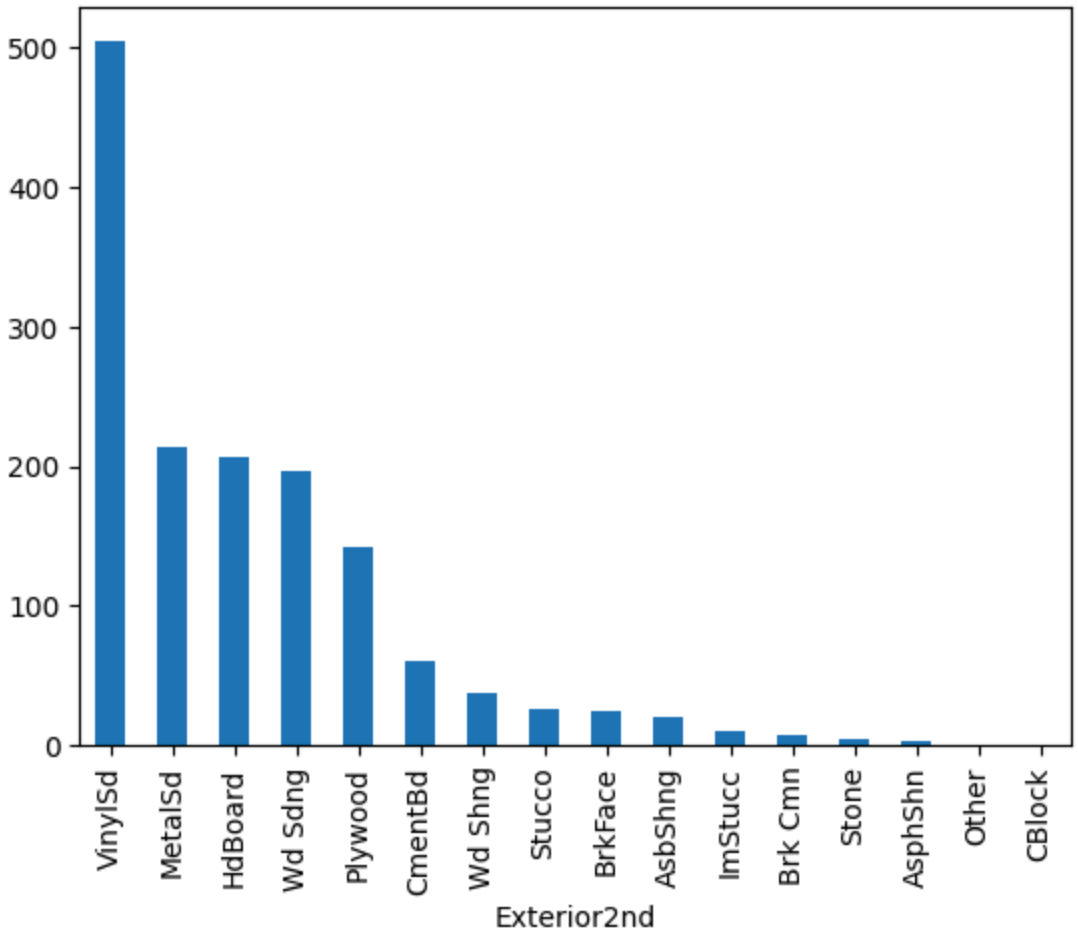
RoofMatl



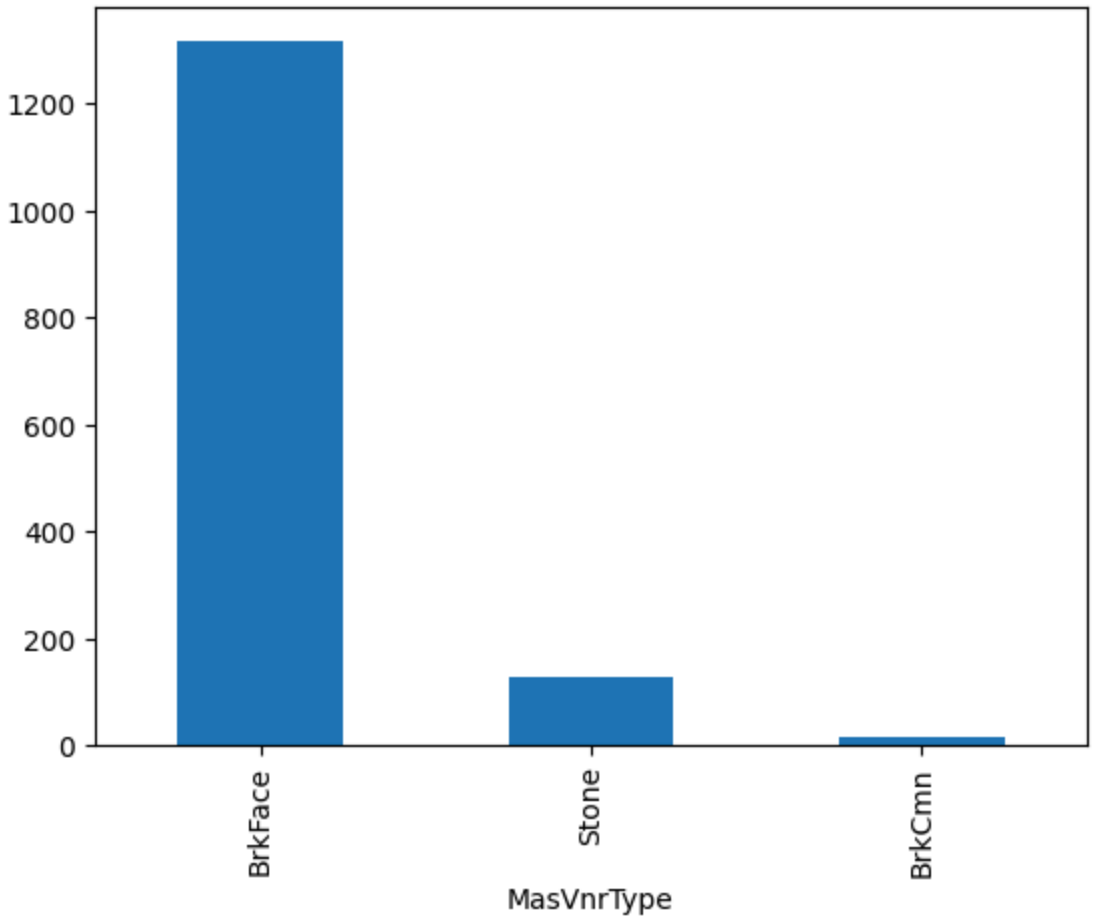
Exterior1st



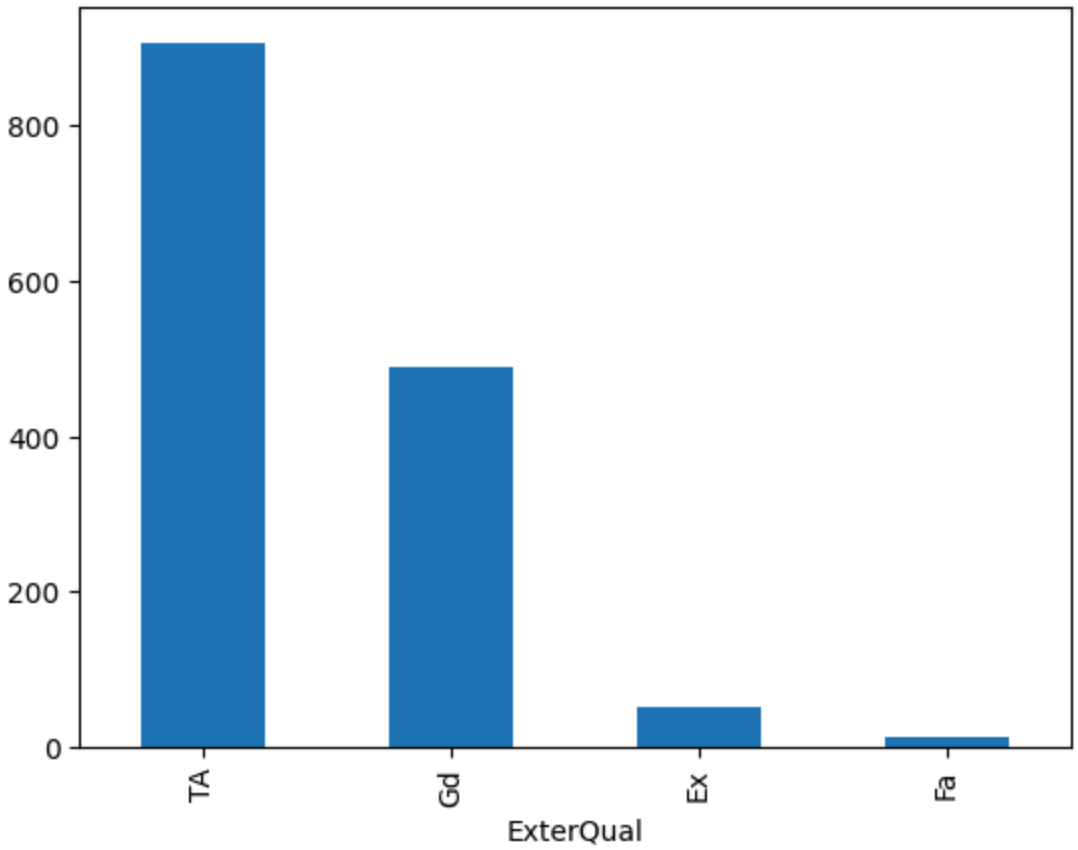
Exterior2nd


500
400
300
200
100
0
VinylSd
MetalSd
HdBoard
Wd Sdng
Plywood
CmentBd
Wd Shng
Stucco
BrkFace
AsbShng
ImStucc
Brk Cmn
Stone
AsphShn
Other
CBlock
Exterior2nd

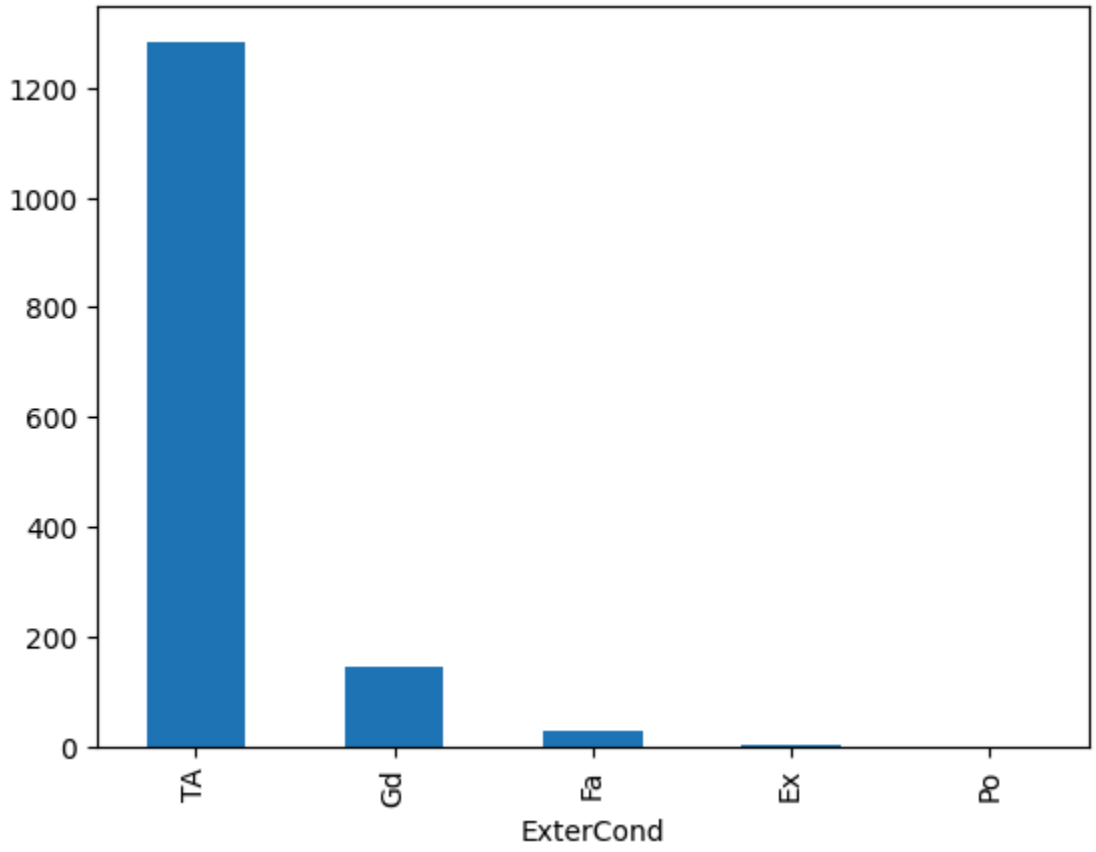
MasVnrType



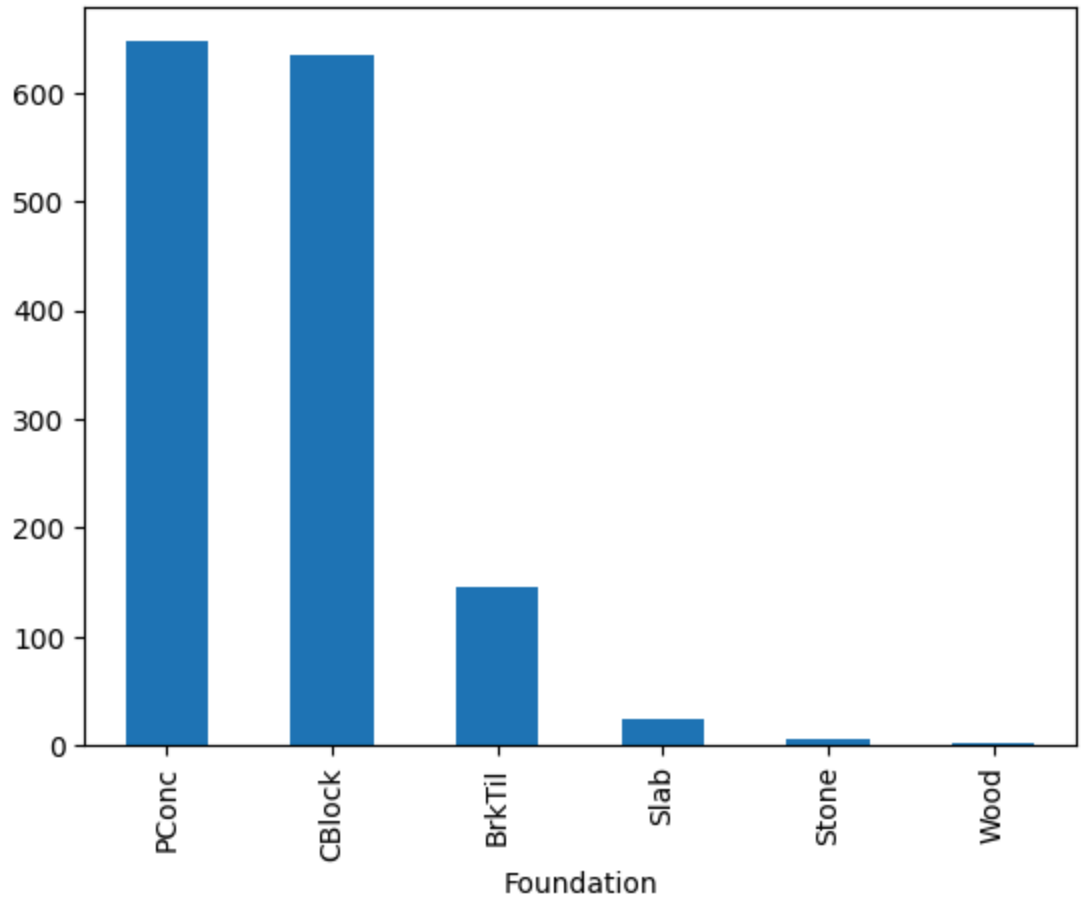
ExterQual



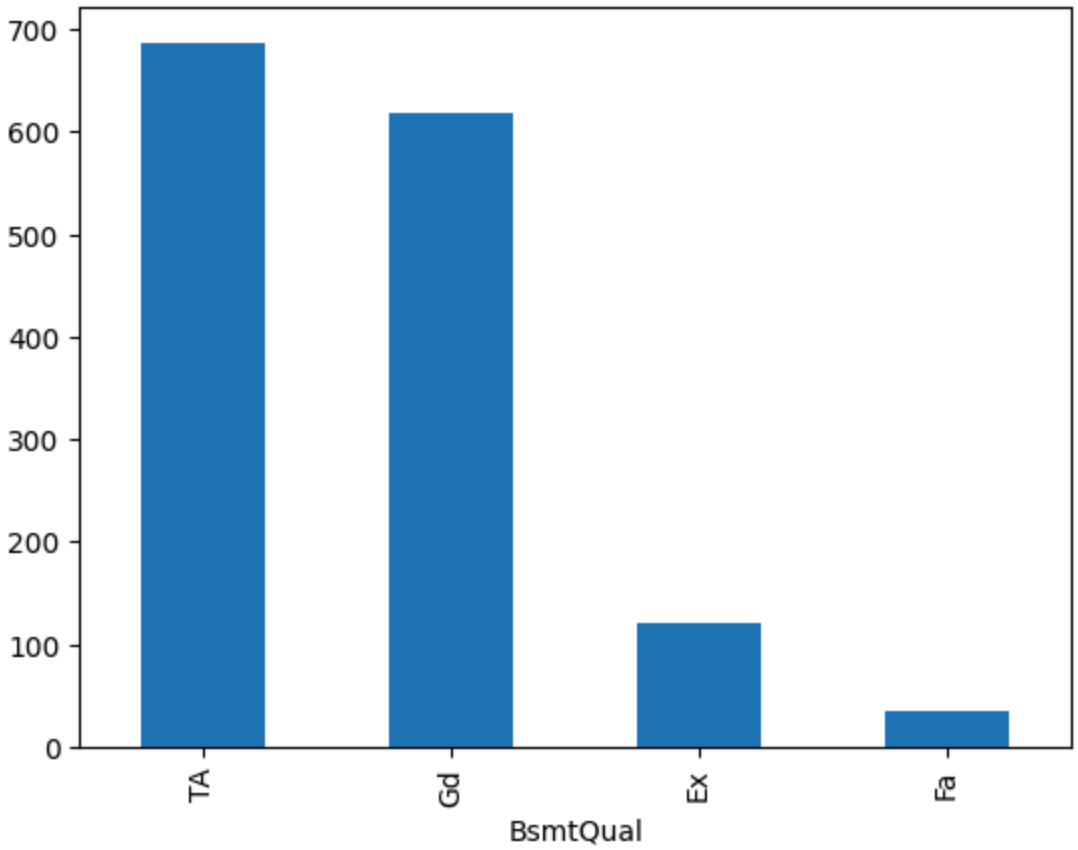
ExterCond



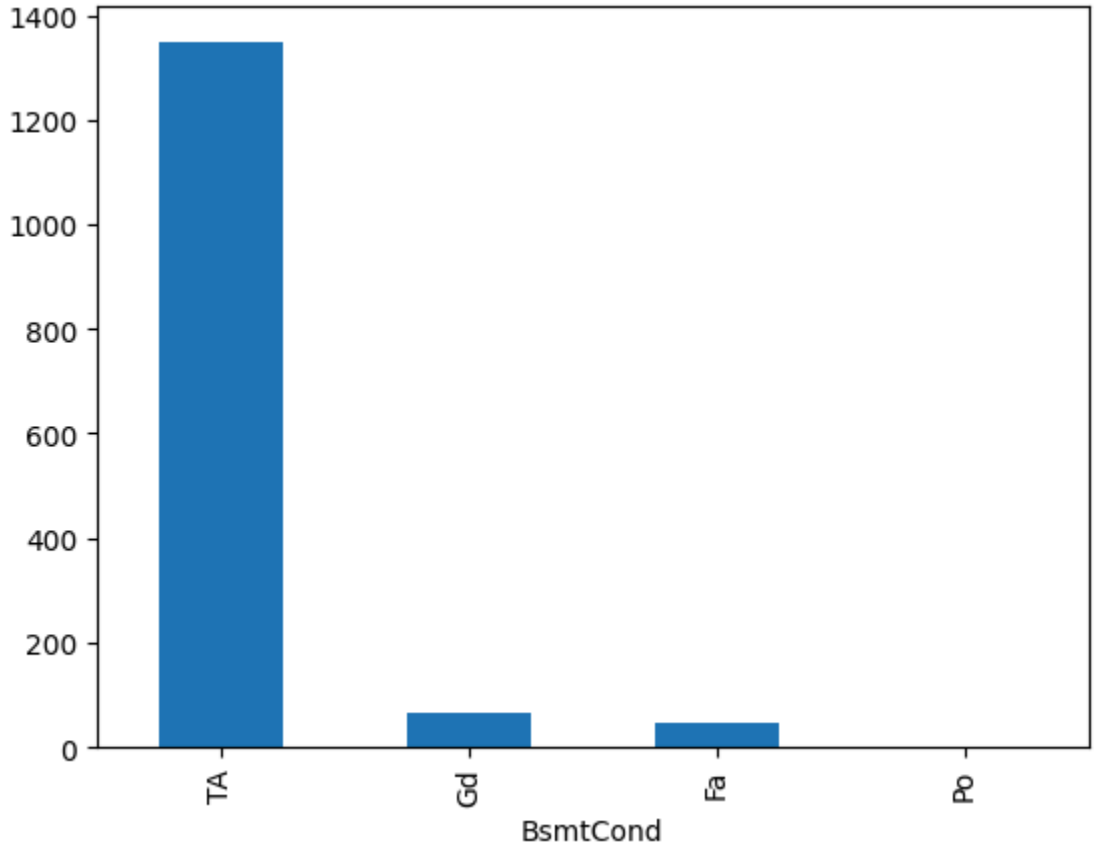
Foundation


600
500
400
300
200
100
0
PConc
CBlock
BrkTil
Slab
Stone
Wood
Foundation

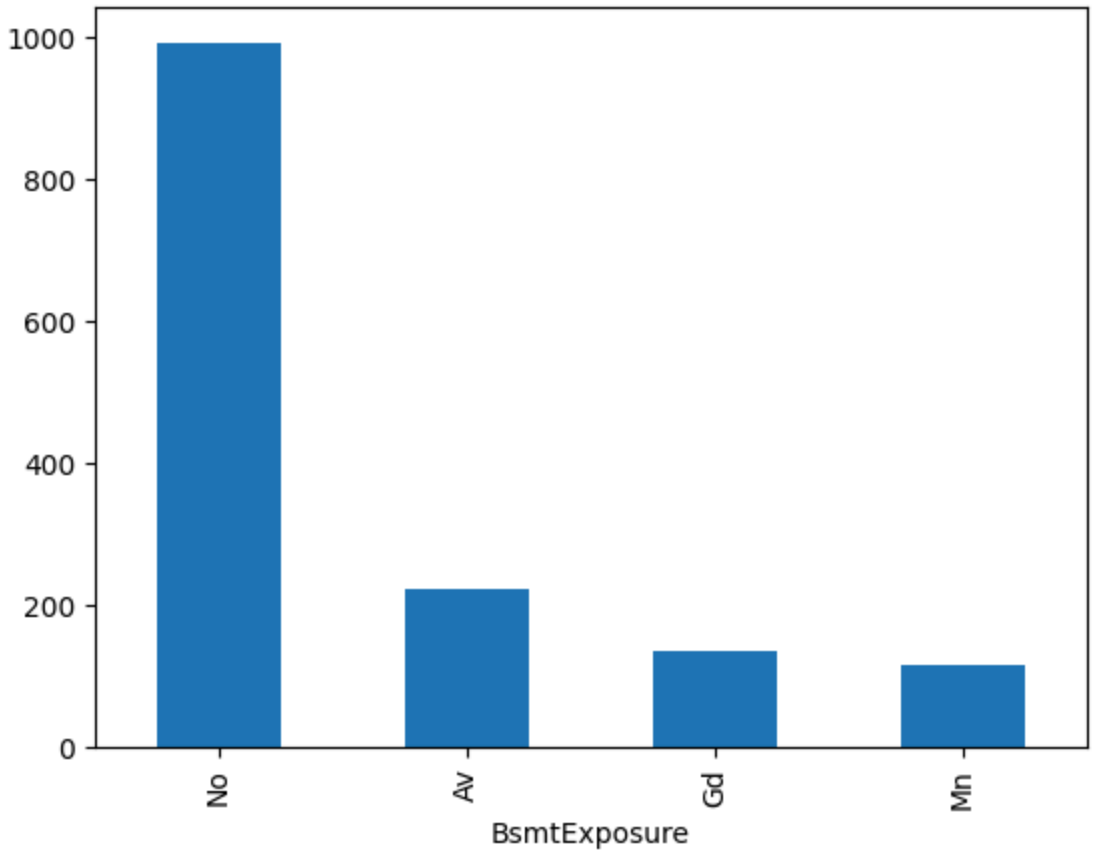
BsmtQual



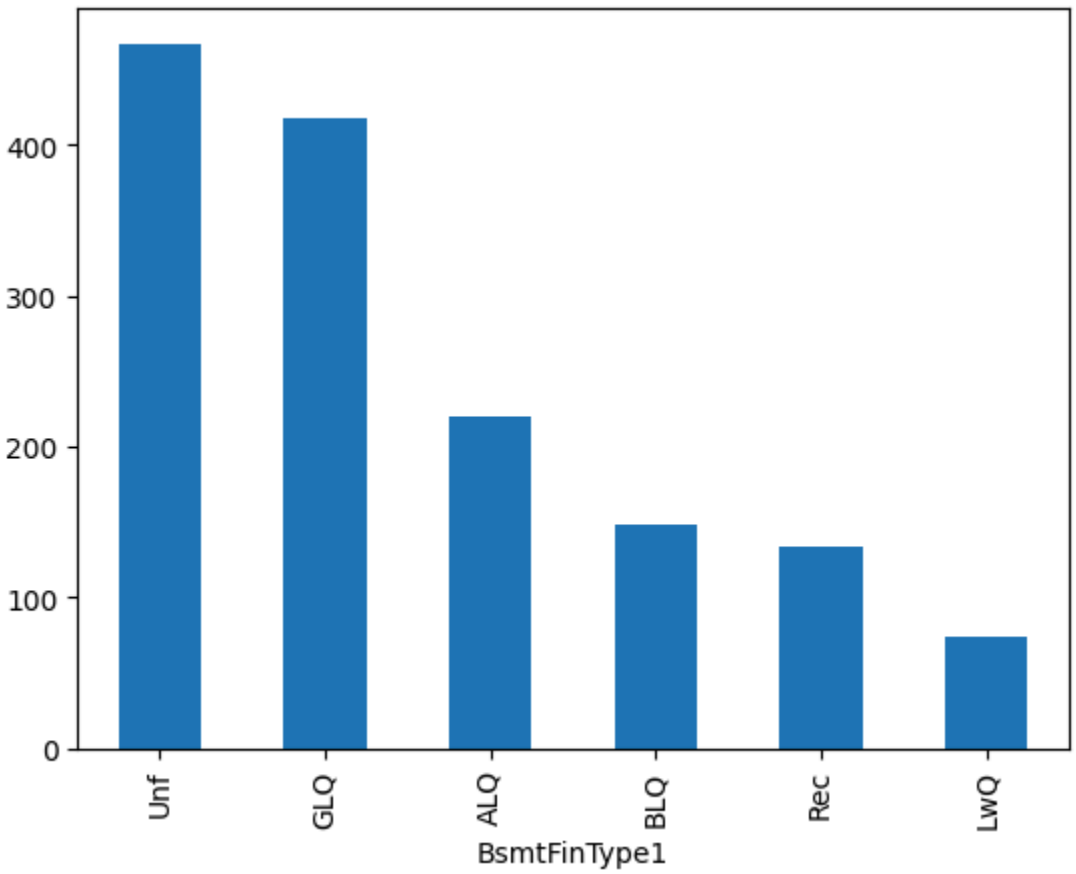
BsmtCond



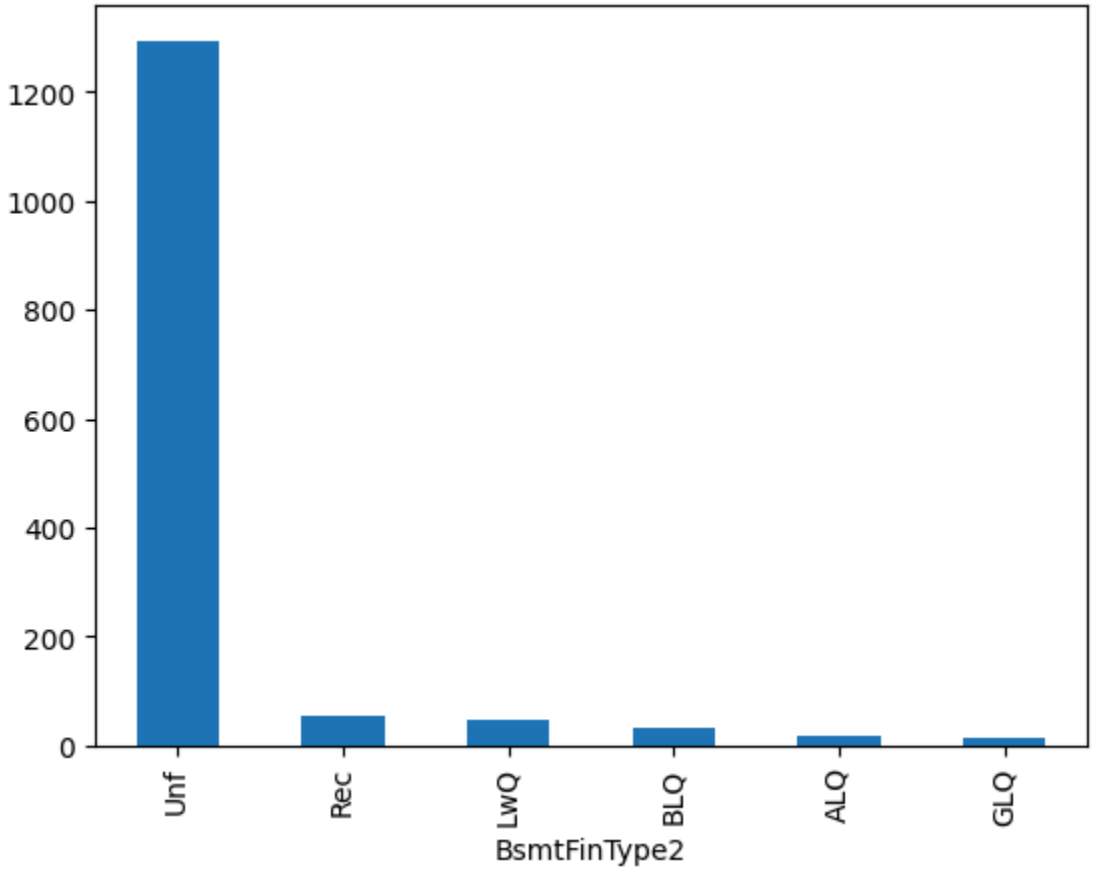
BsmtExposure



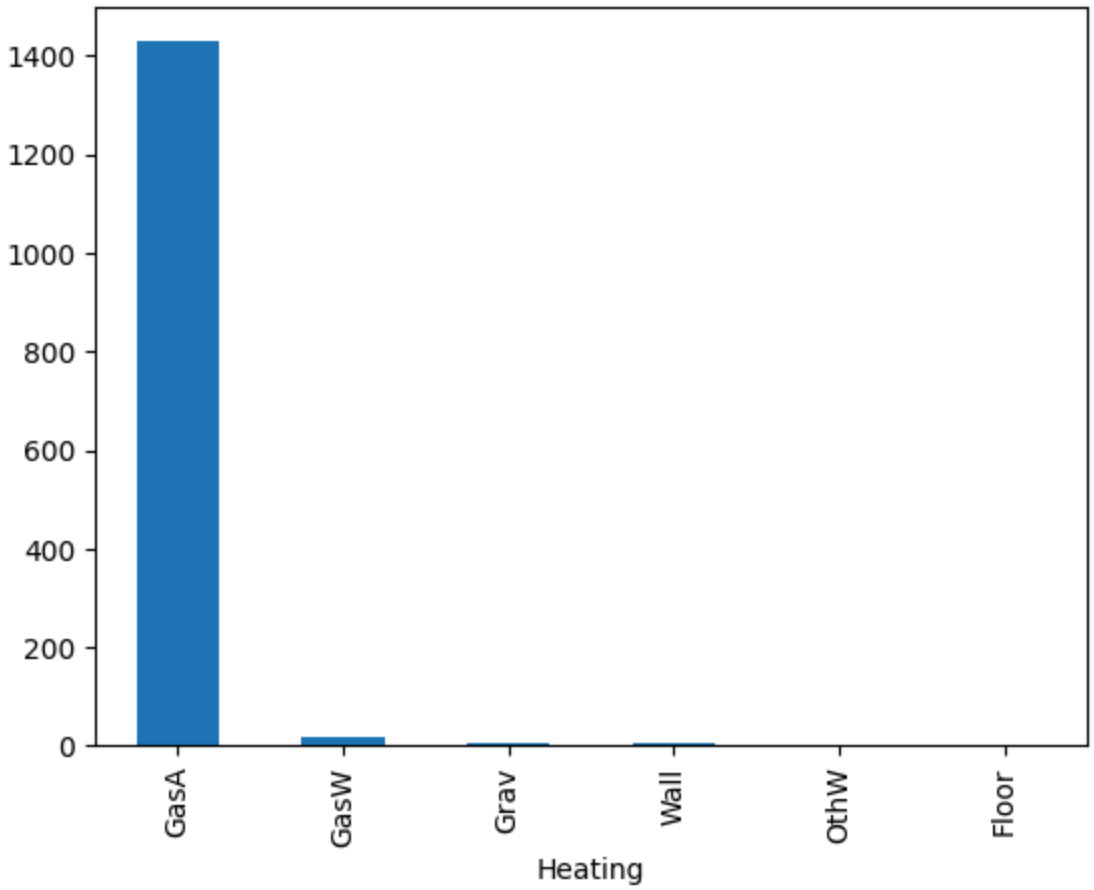
BsmtFinType1



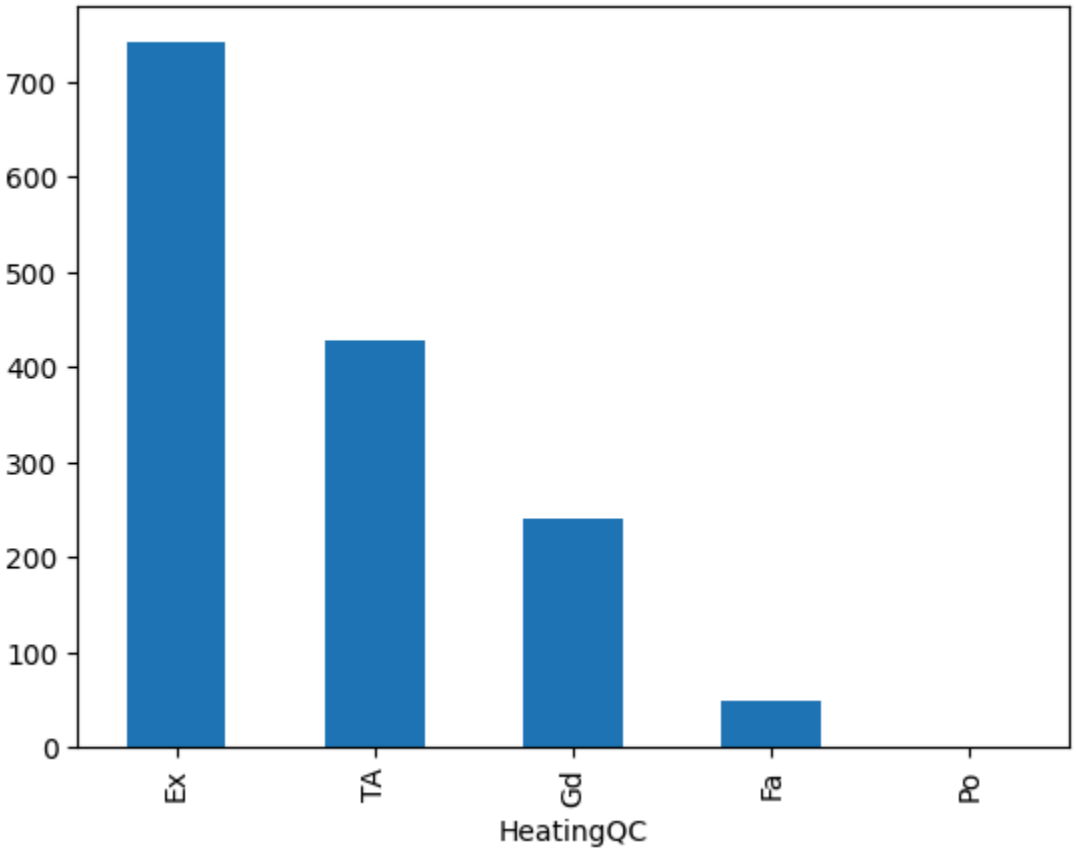
BsmtFinType2



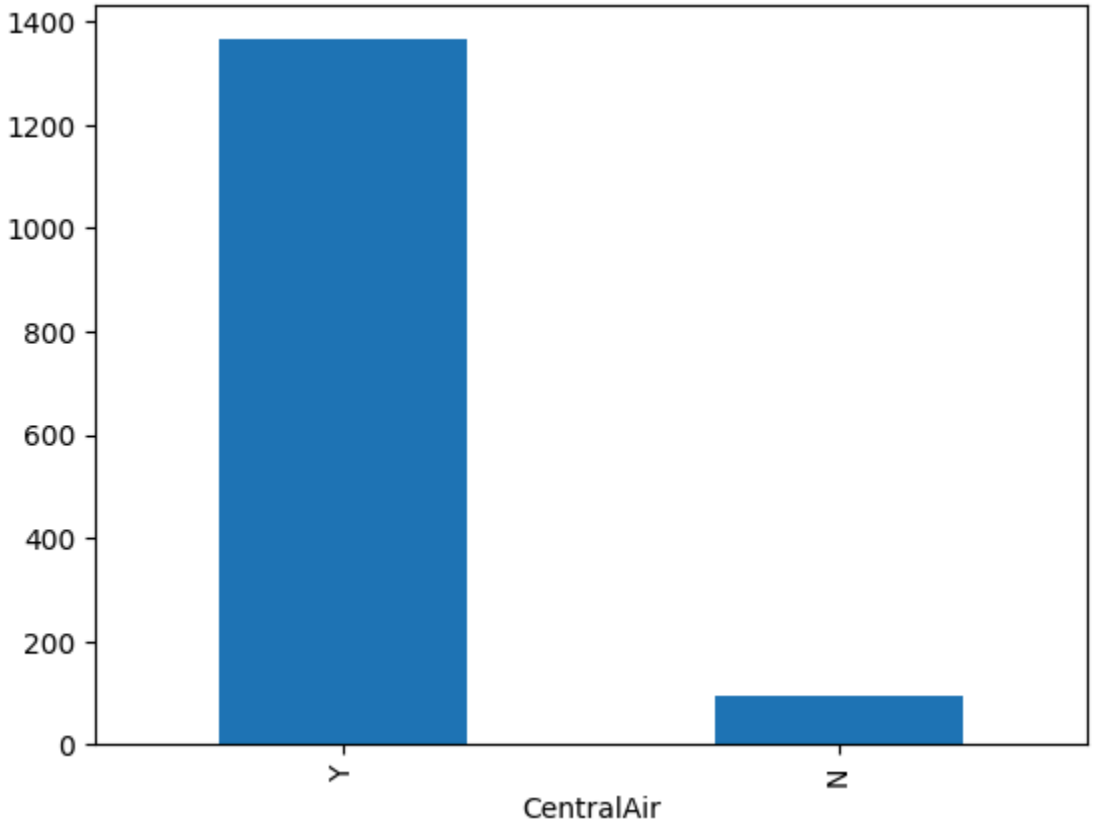
Heating


1400
1200
1000
800
600
400
200
0
GasA
GasW
Grav
Wall
OthW
Floor
Heating

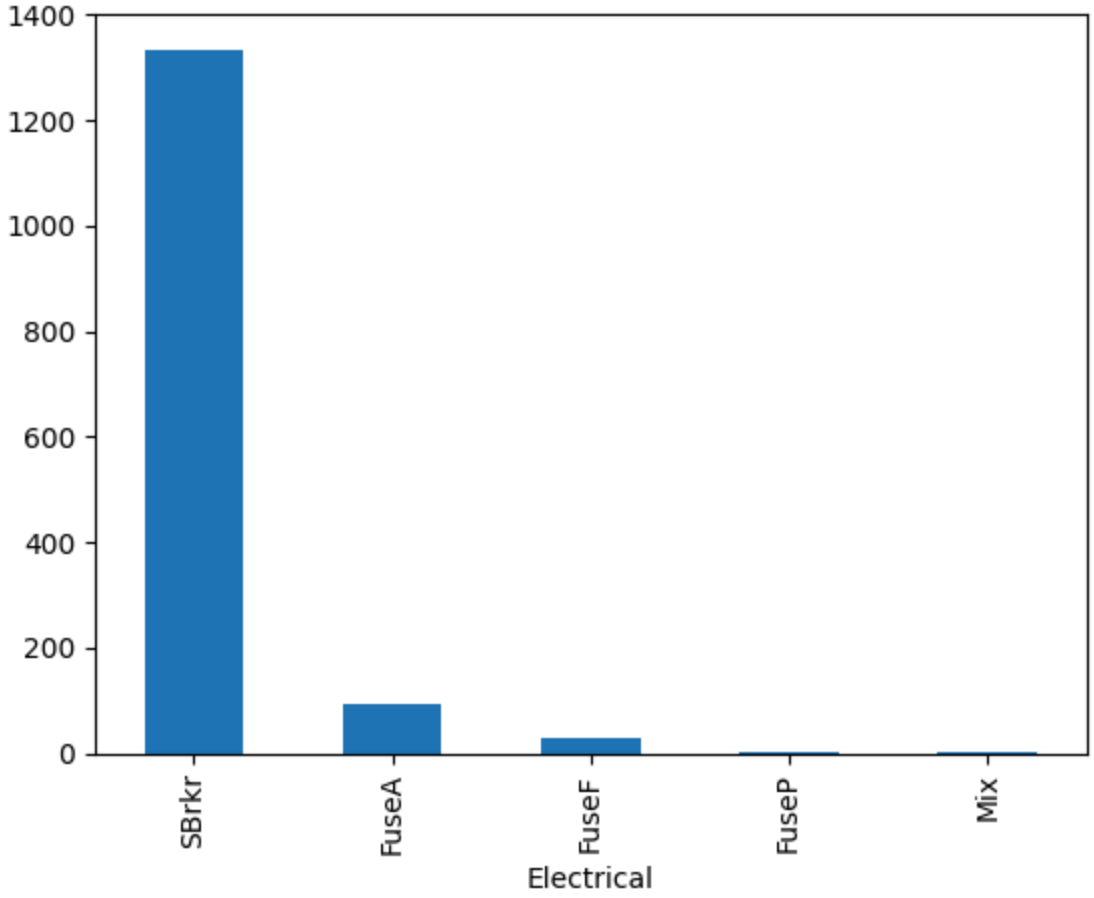
HeatingQC


700
600
500
400
300
200
100
0
Ex
TA
Gd
Fa
Po
HeatingQC

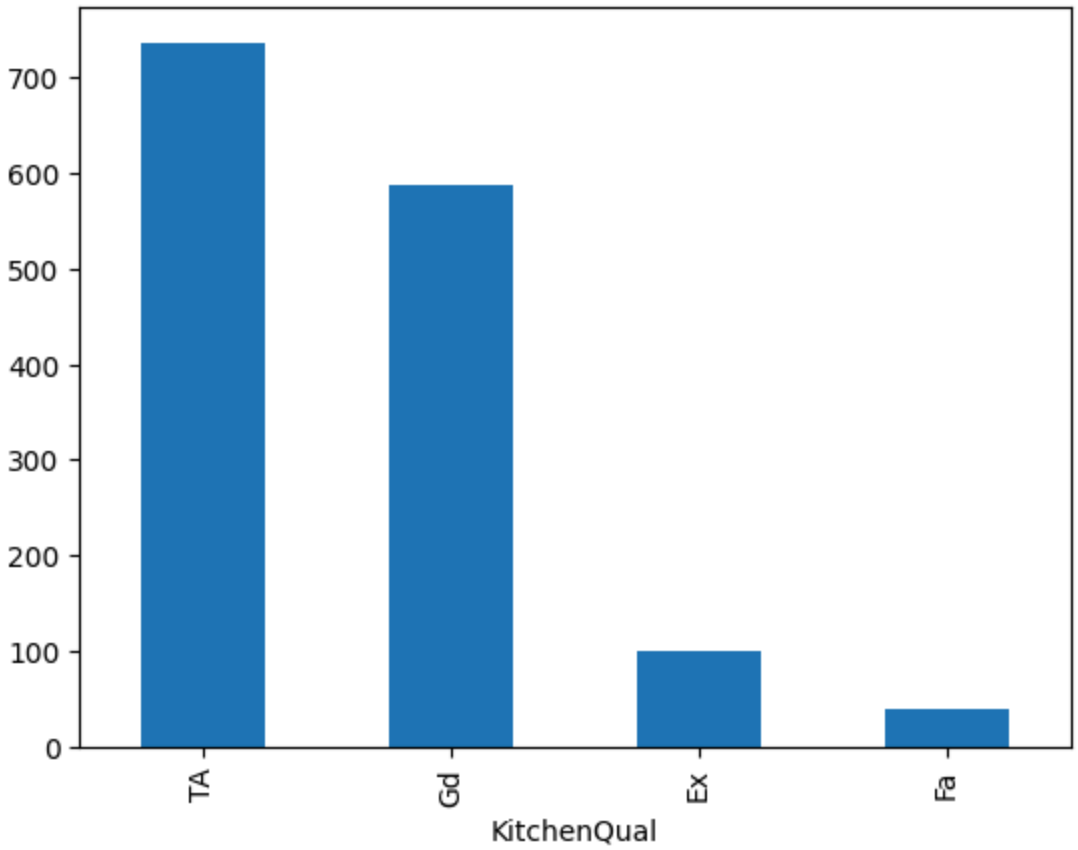
CentralAir



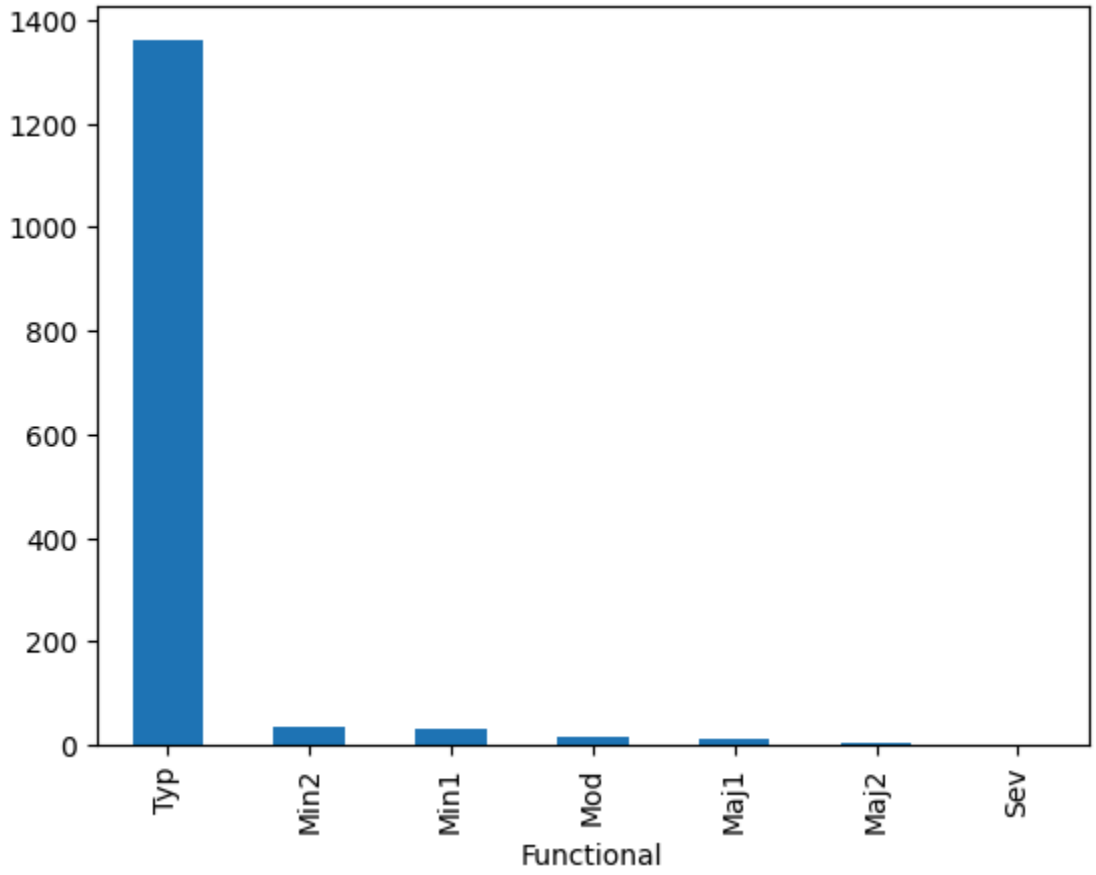
Electrical



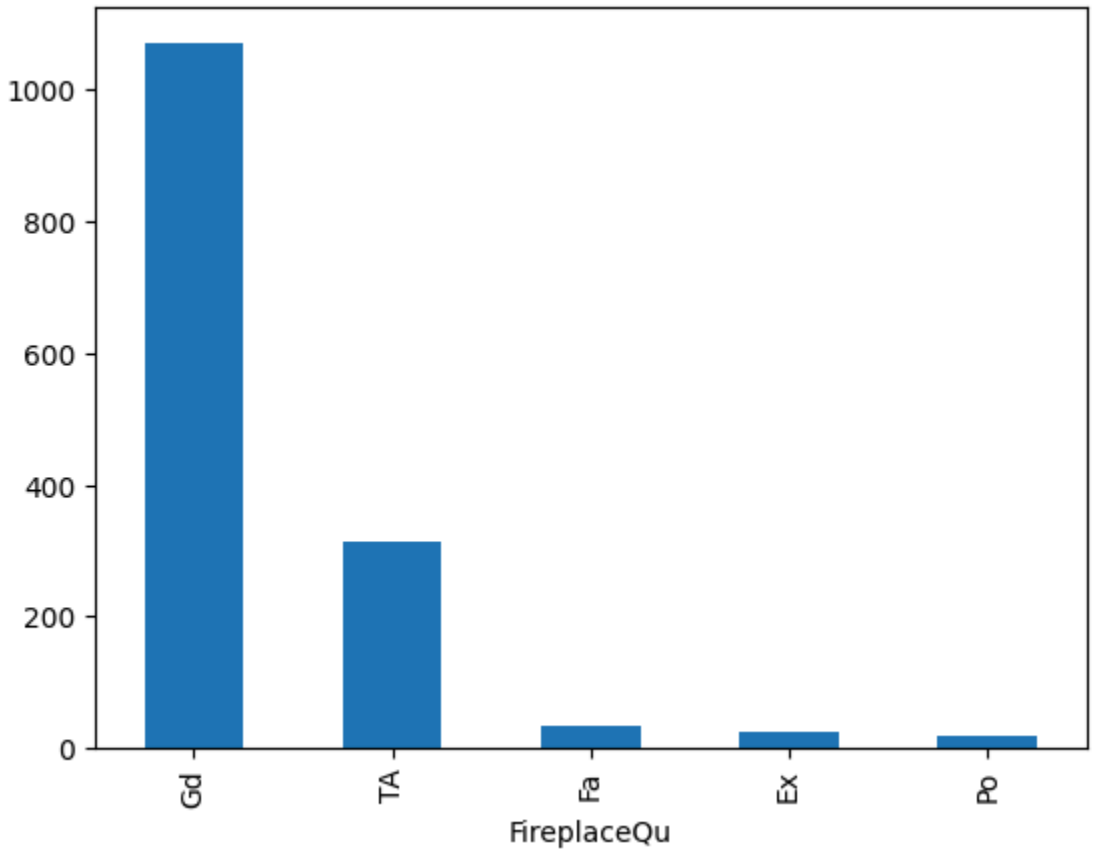
KitchenQual


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

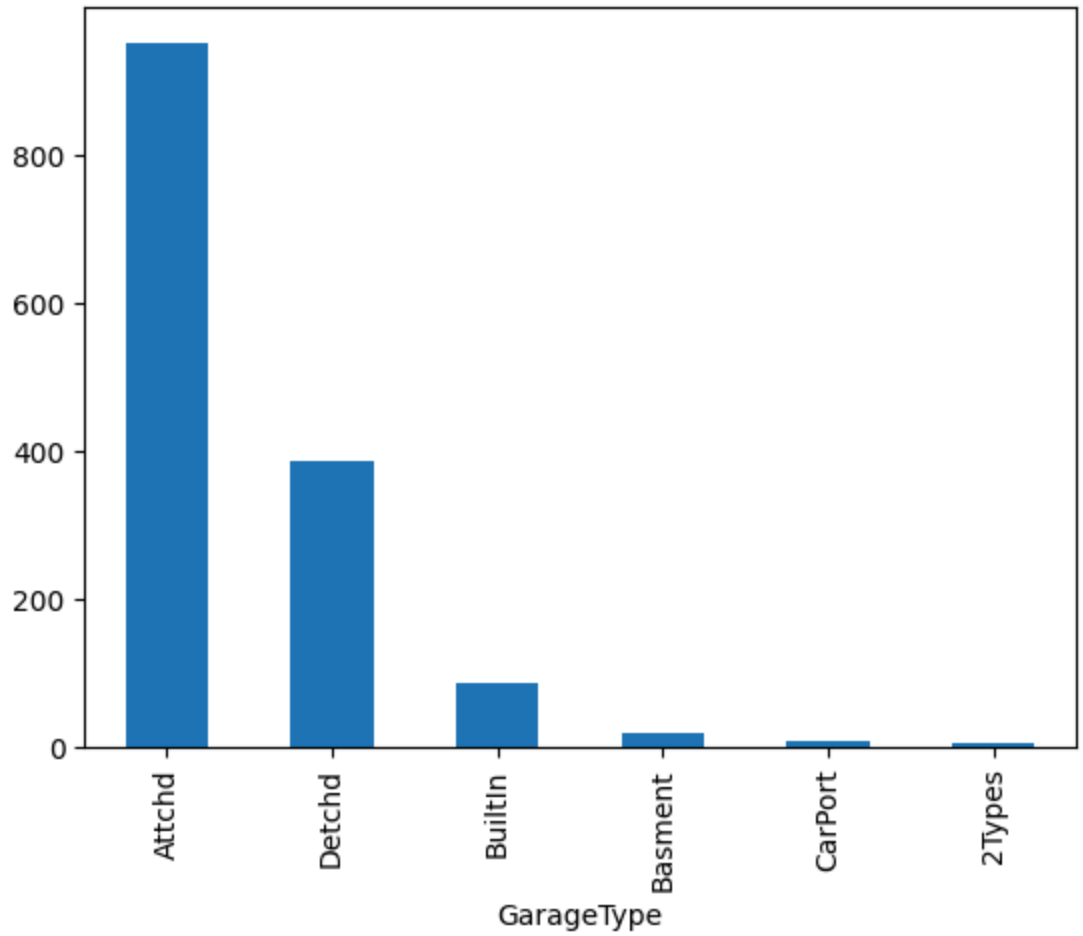
```
Functional
```


1400
1200
1000
800
600
400
200
0
Typ
Min2
Min1
Mod
Maj1
Maj2
Sev
Functional

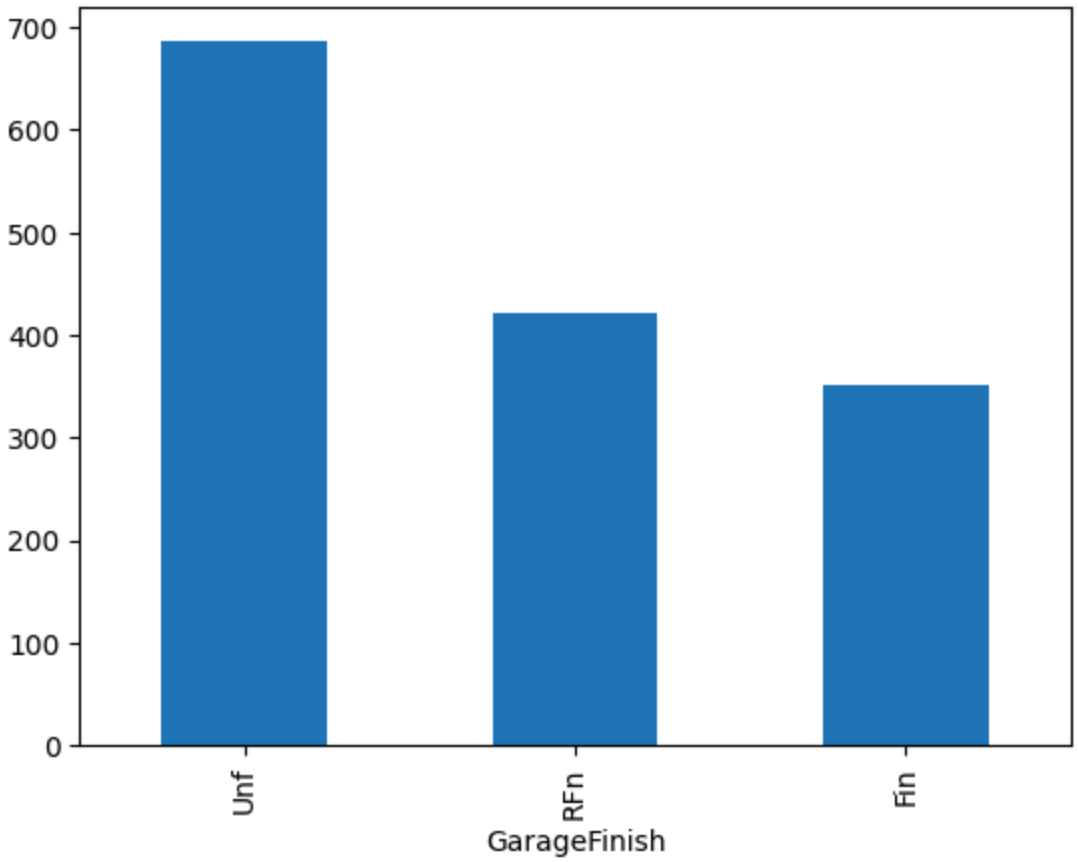
FireplaceQu



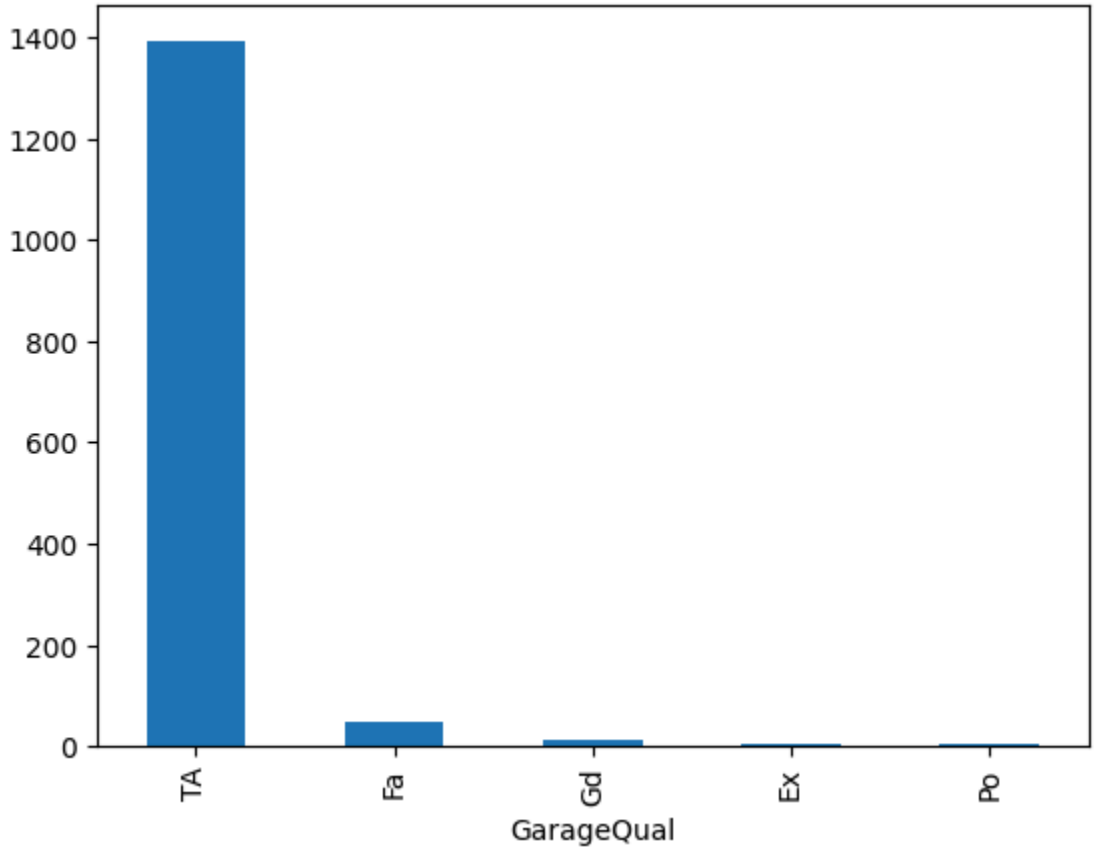
GarageType



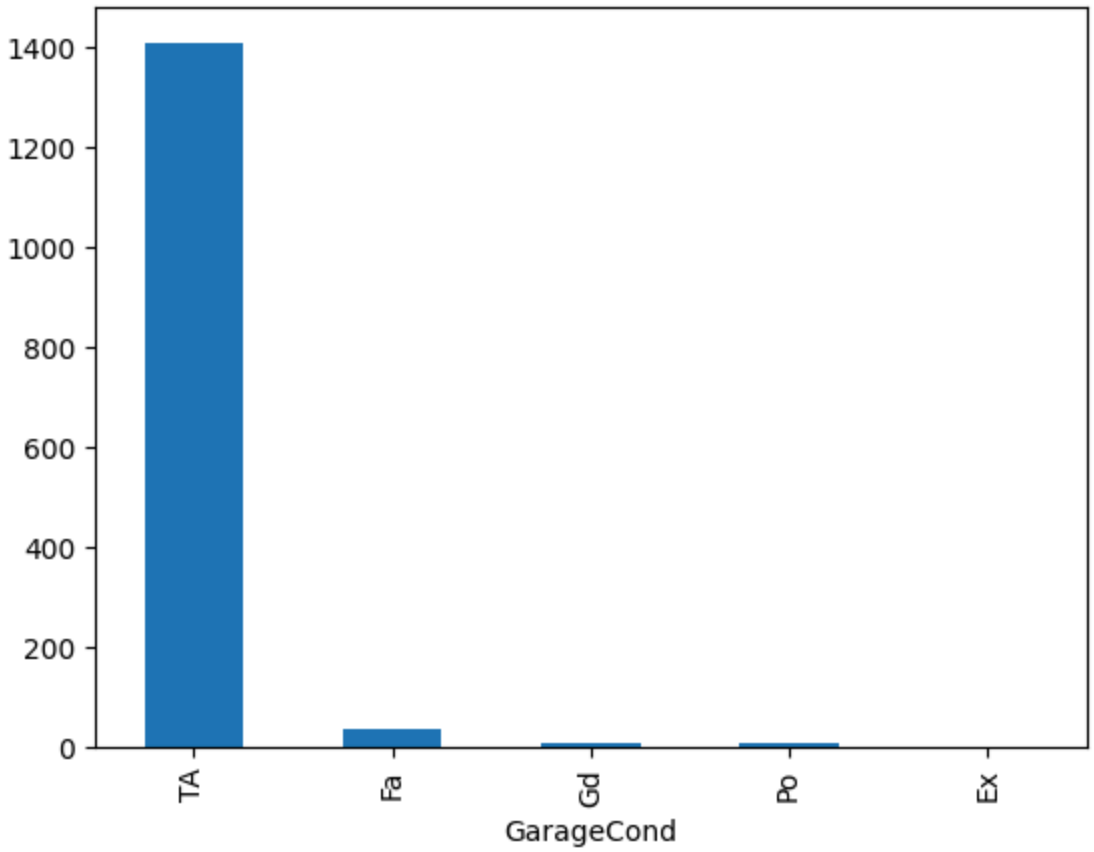
GarageFinish



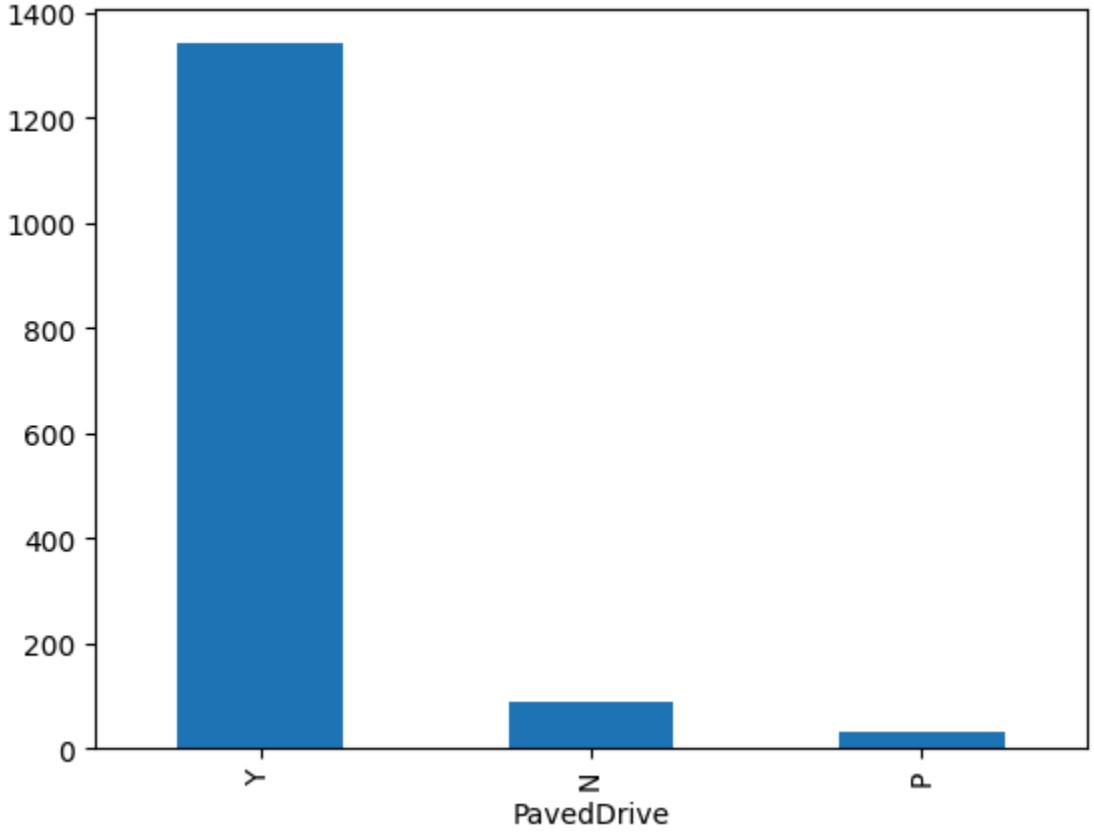
GarageQual



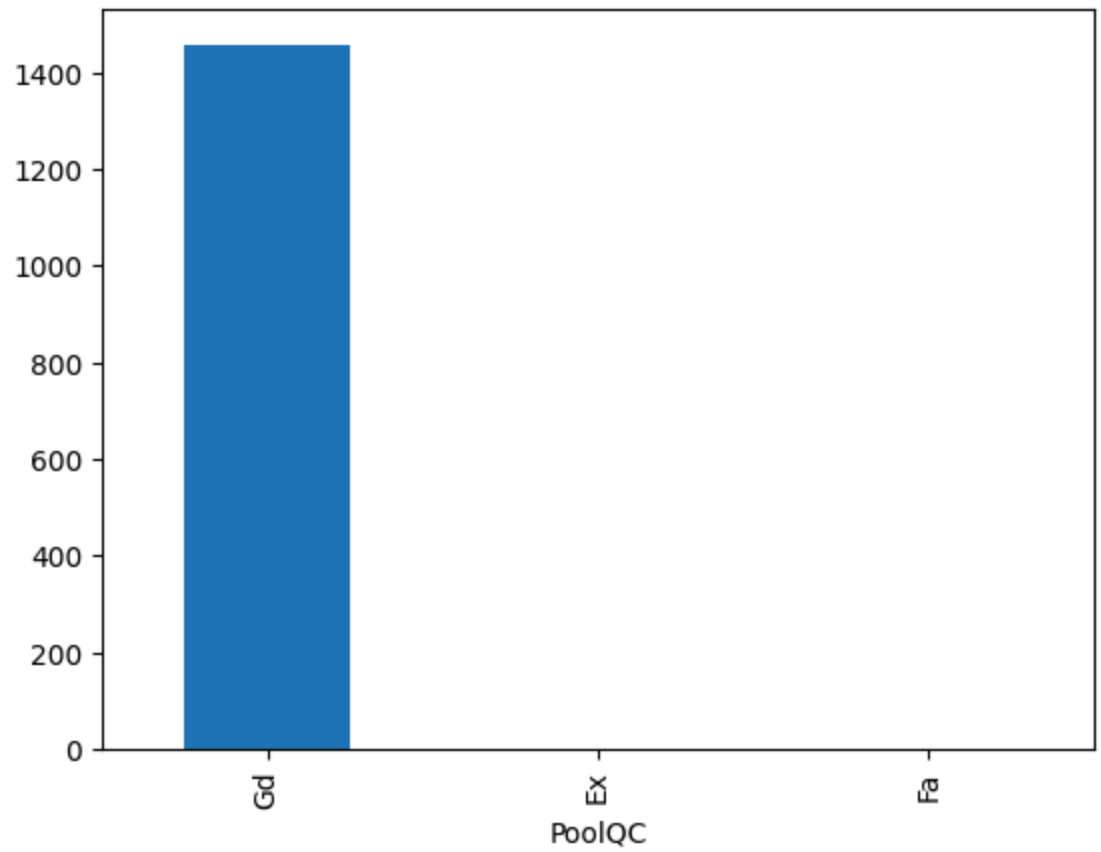
GarageCond



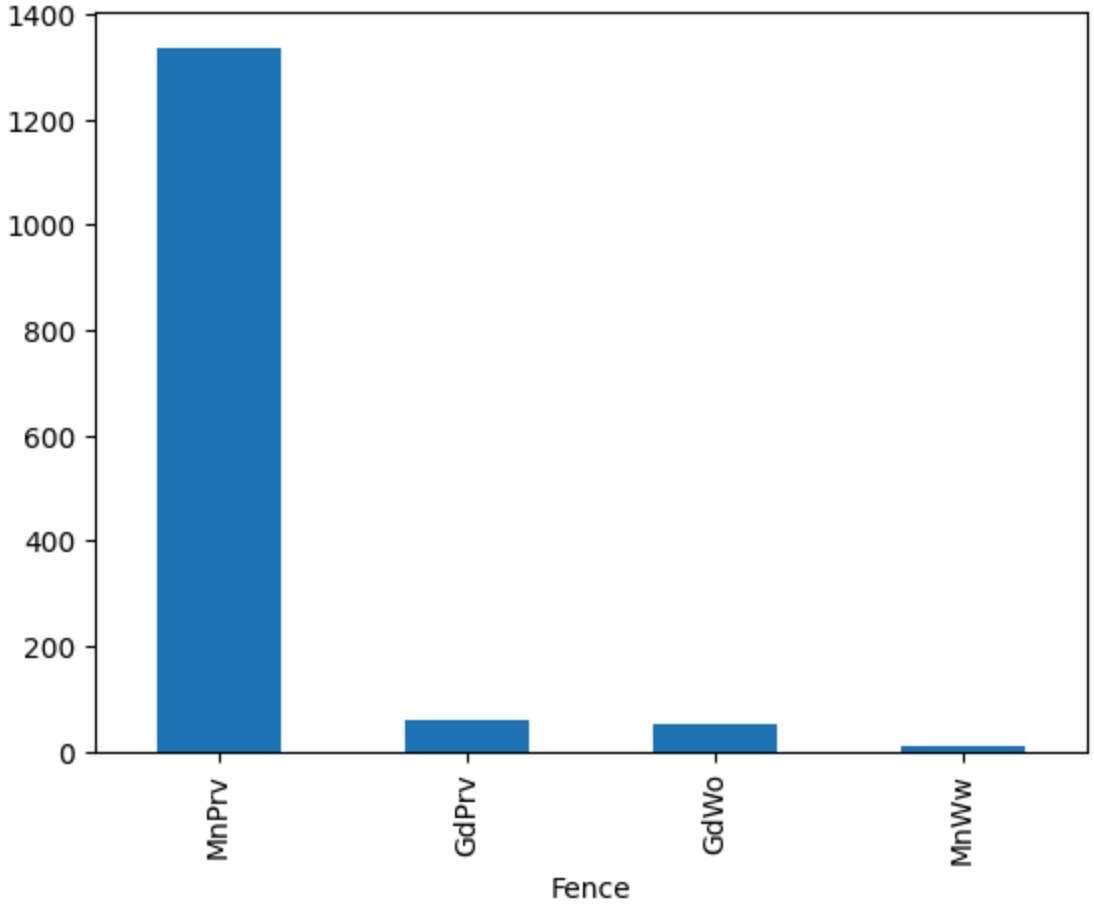
PavedDrive



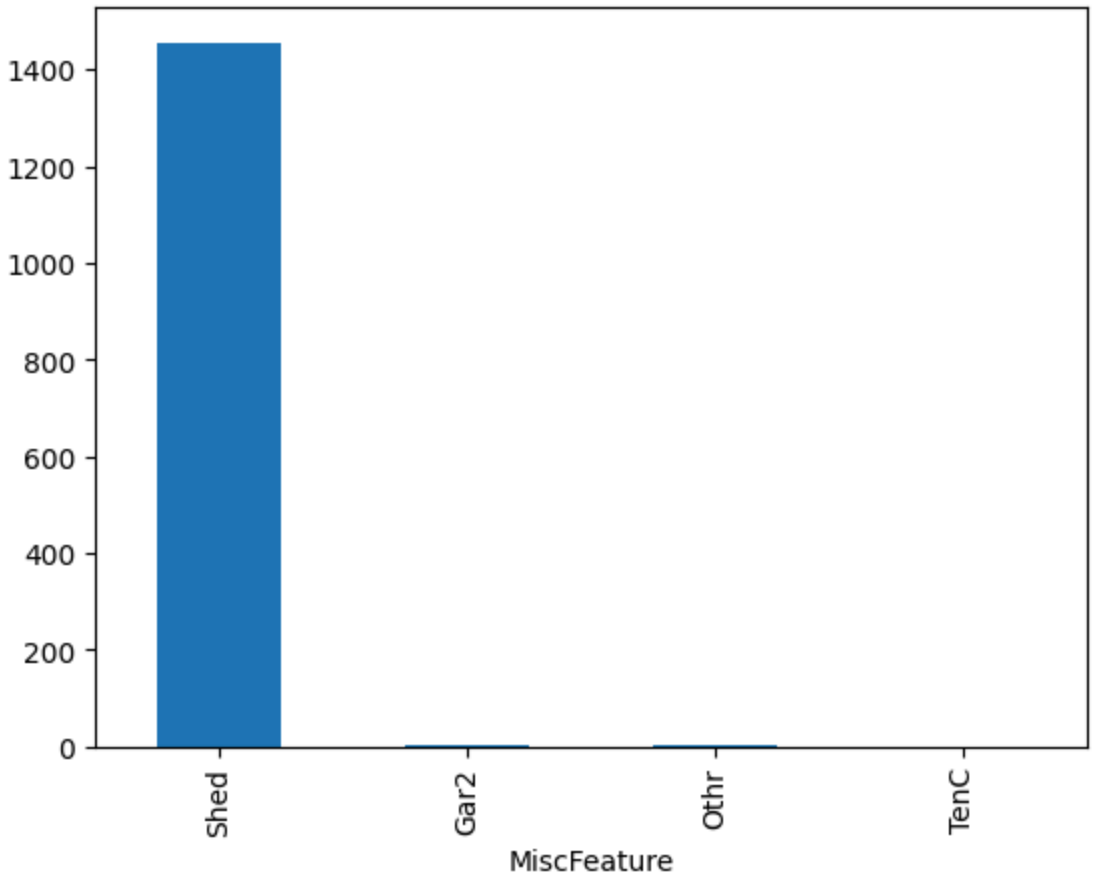
PoolQC



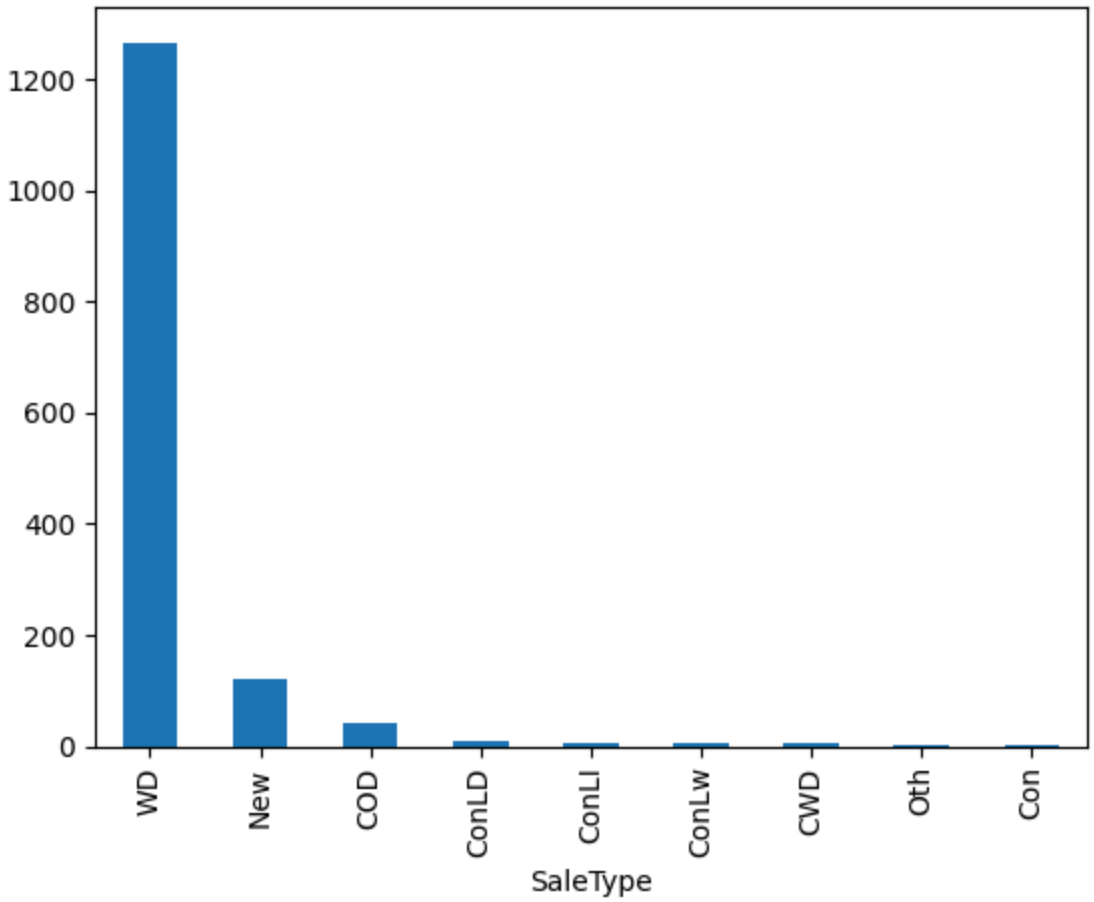
Fence



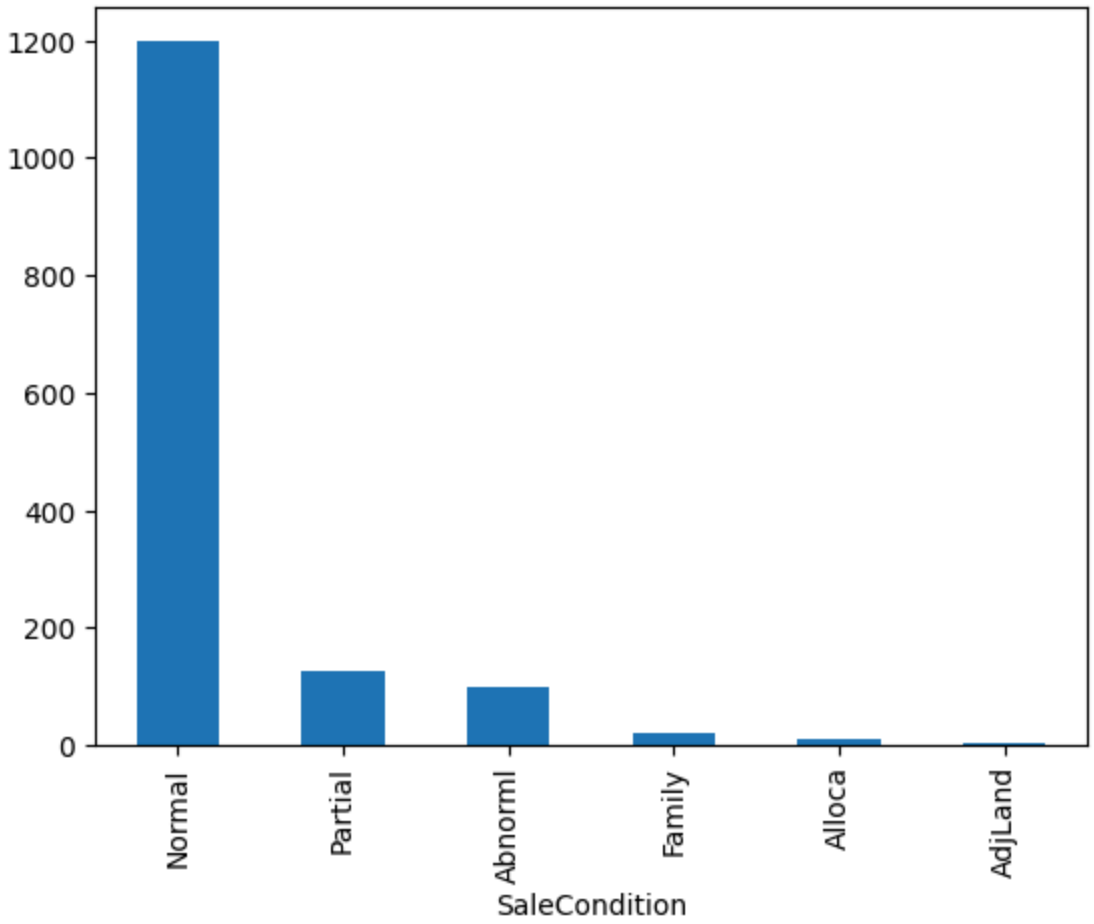
MiscFeature


1400
1200
1000
800
600
400
200
0
Shed
Gar2
Othr
TenC
MiscFeature

SaleType



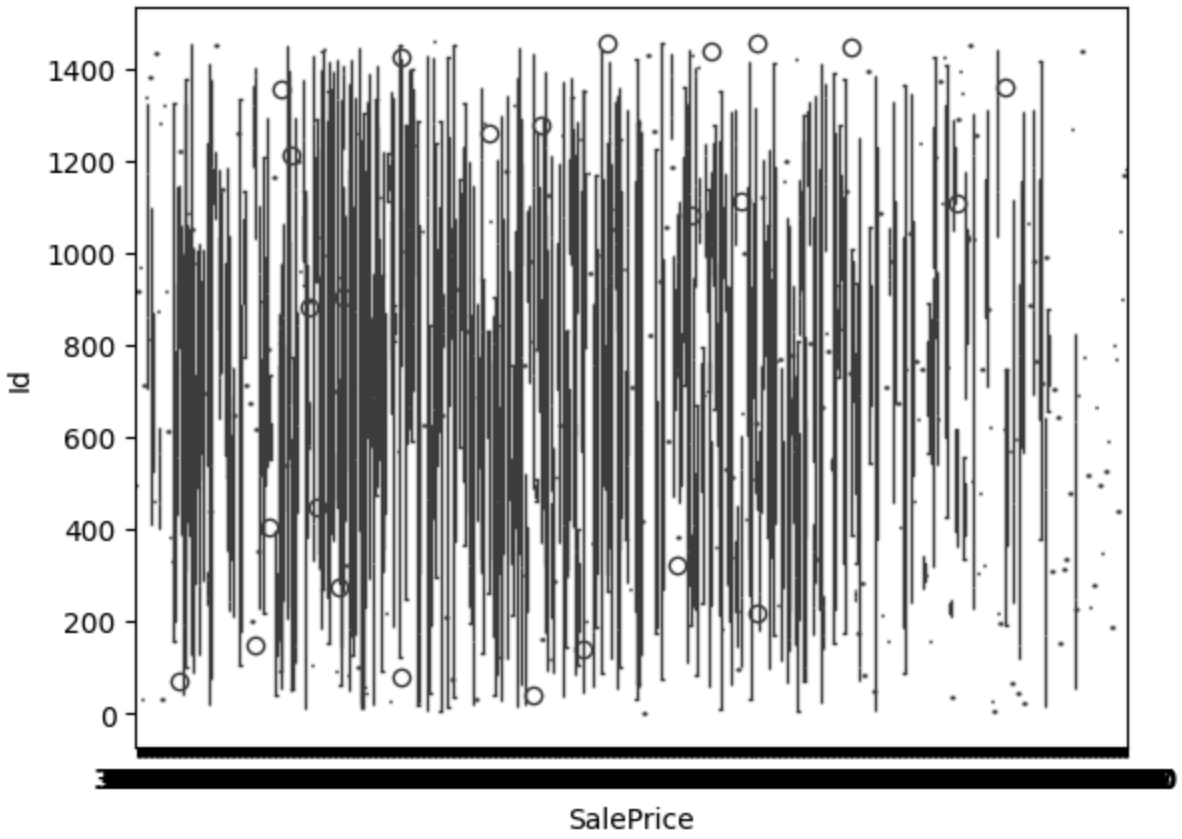
SaleCondition



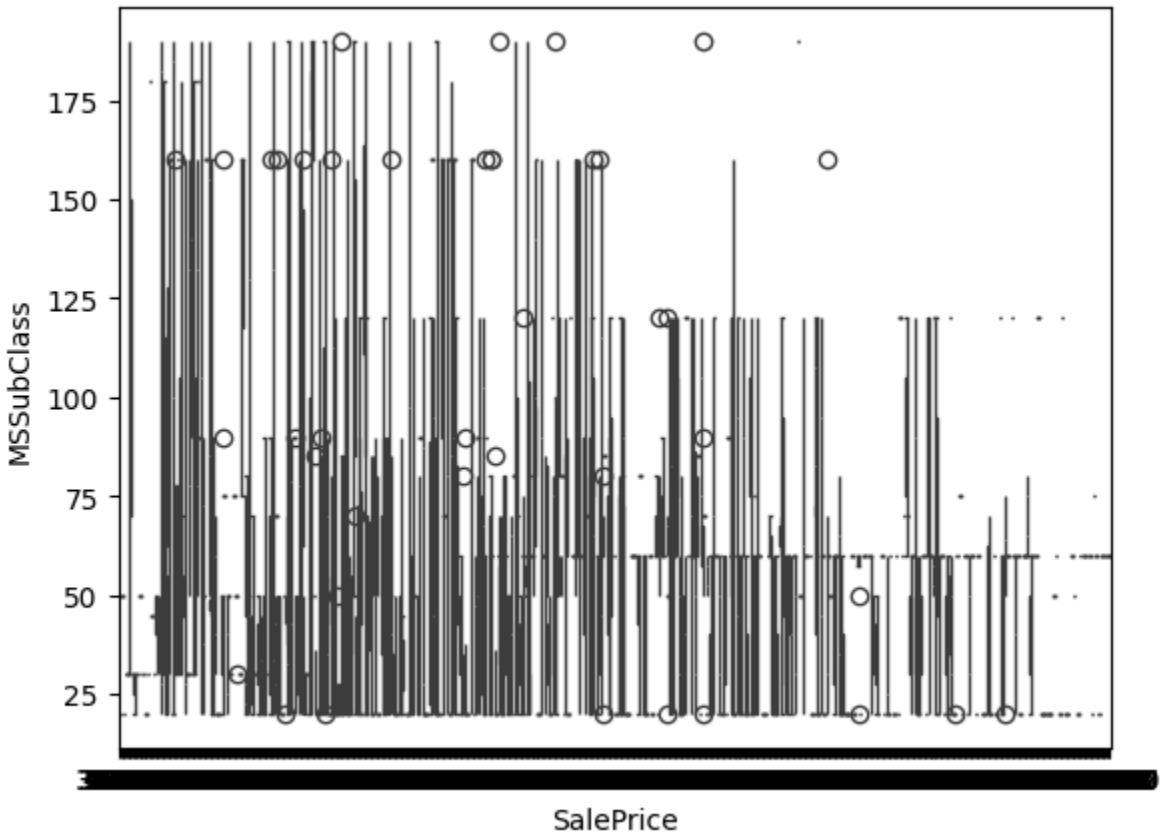
In [319…

```
for i in df.columns:
  sns.boxplot(data=df,x='SalePrice',y=i)
  print(i)
  show()
```

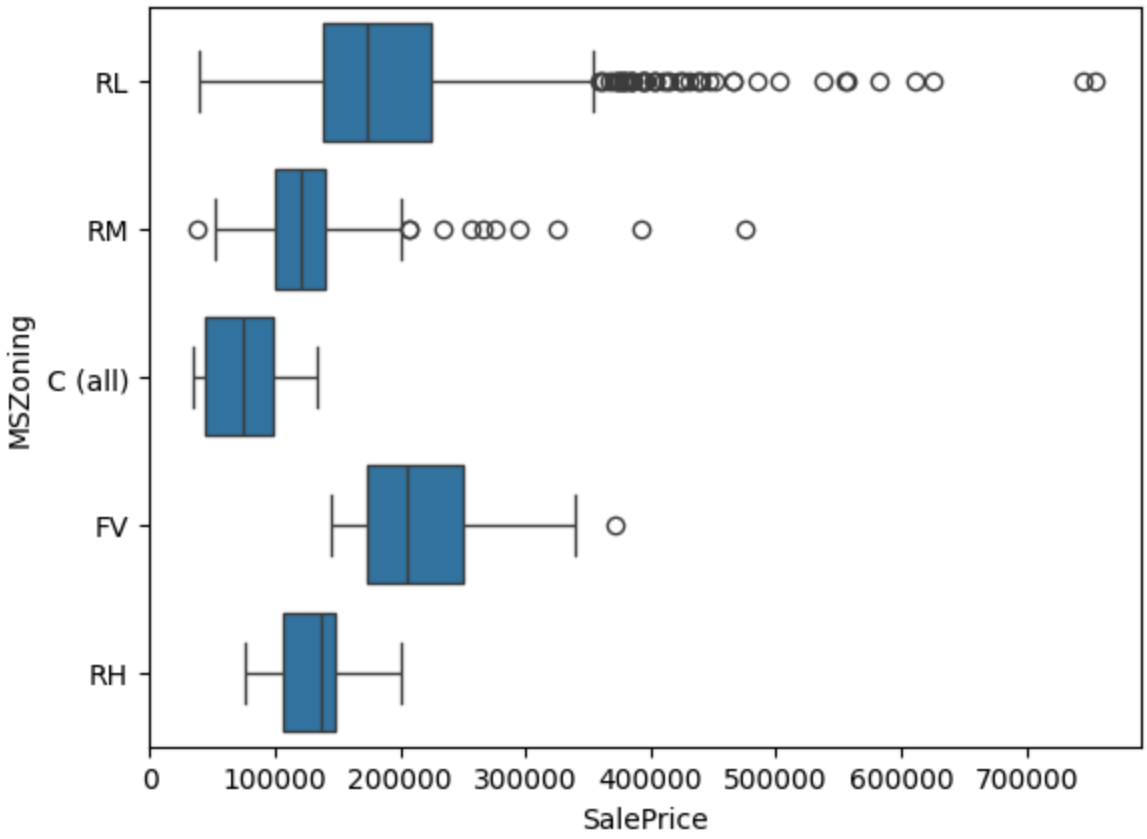
```
Id
```



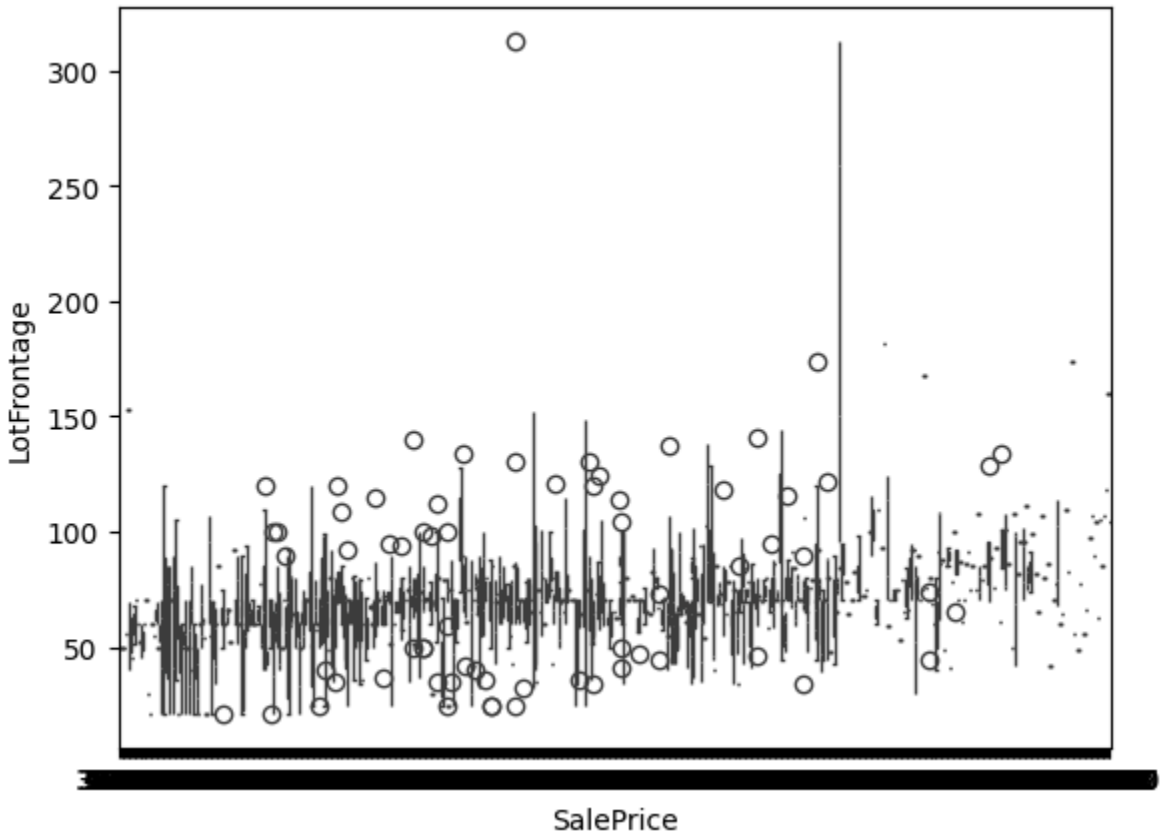
MSSubClass



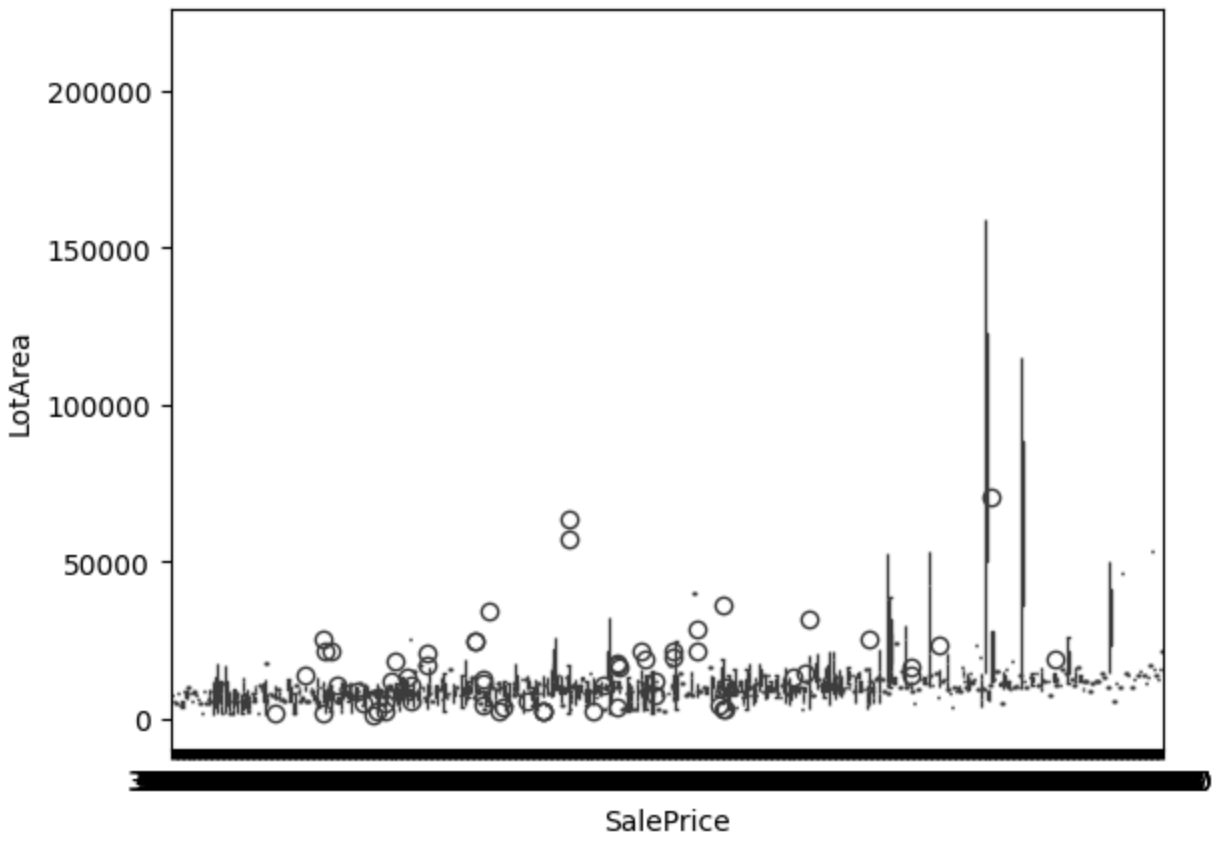
MSZoning



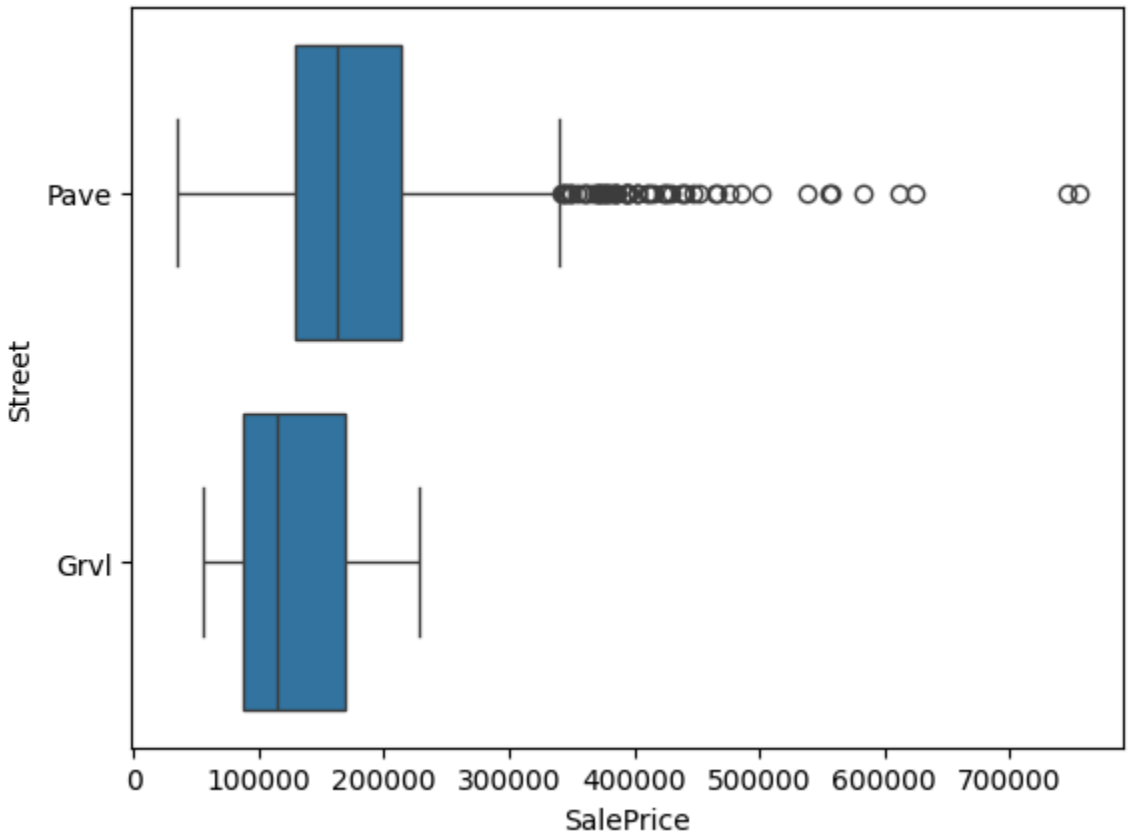
LotFrontage



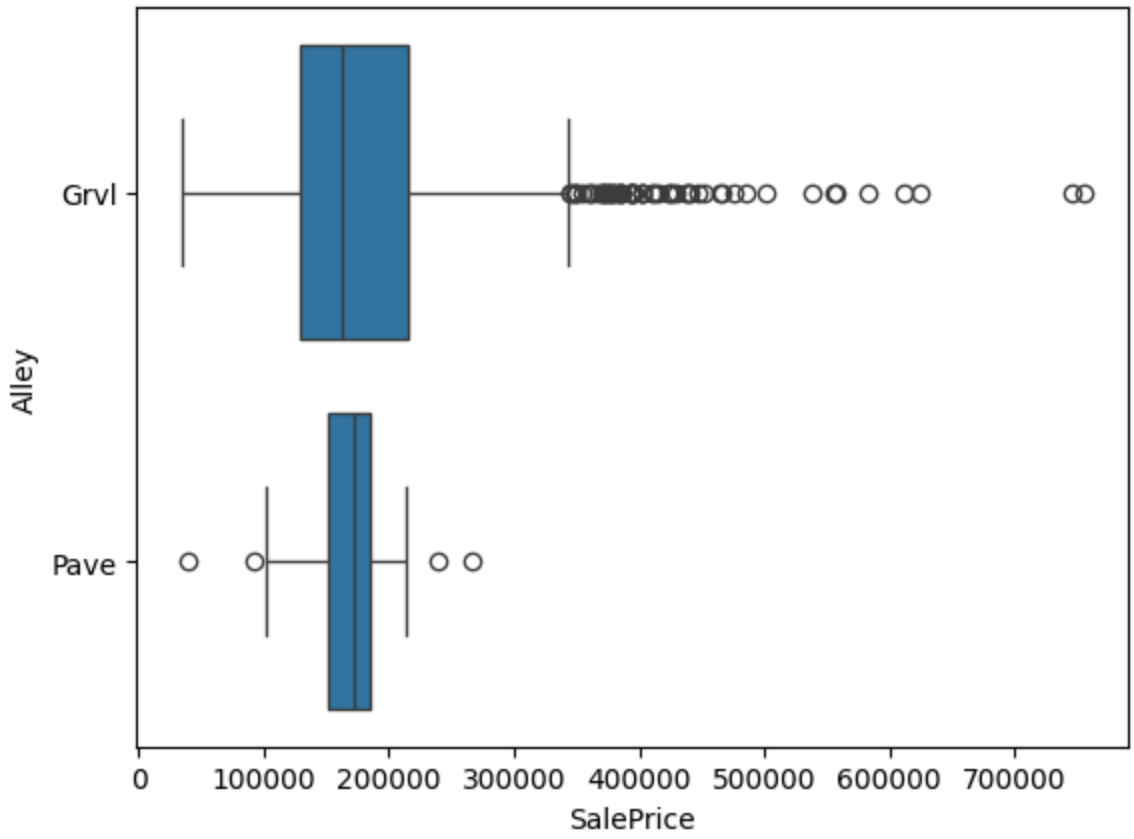
LotArea



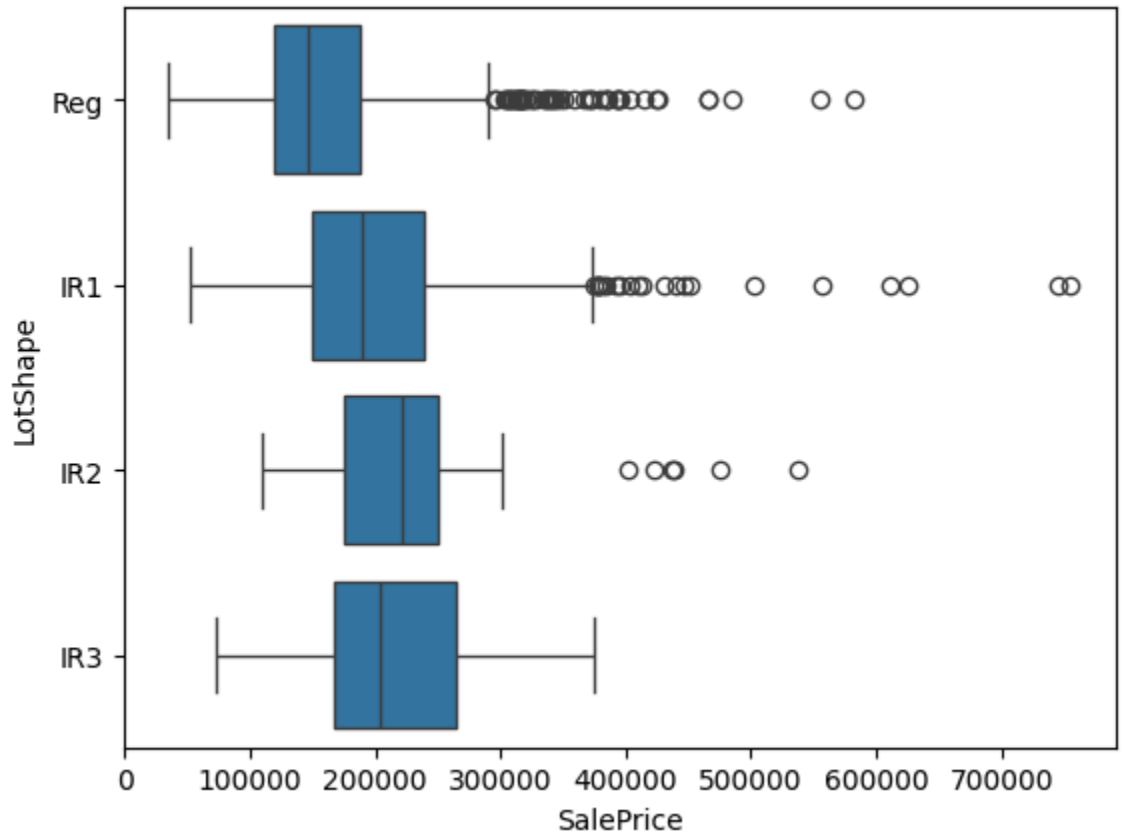
Street



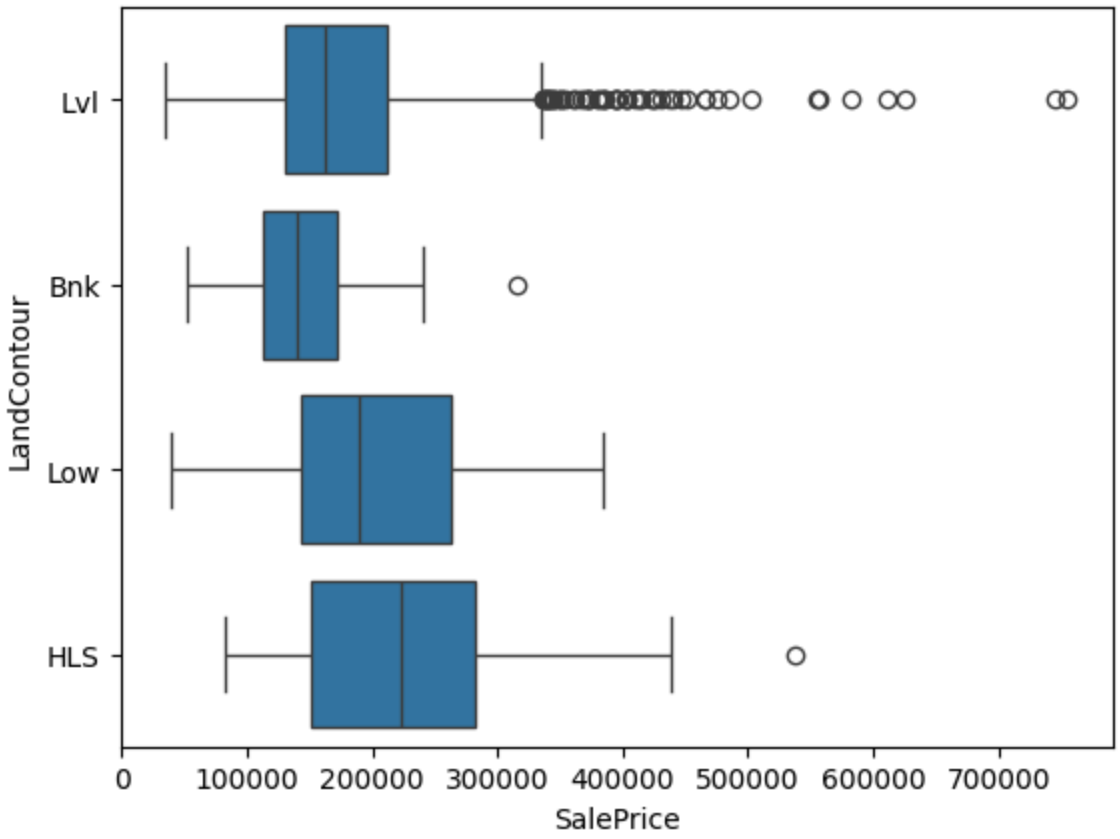
Alley



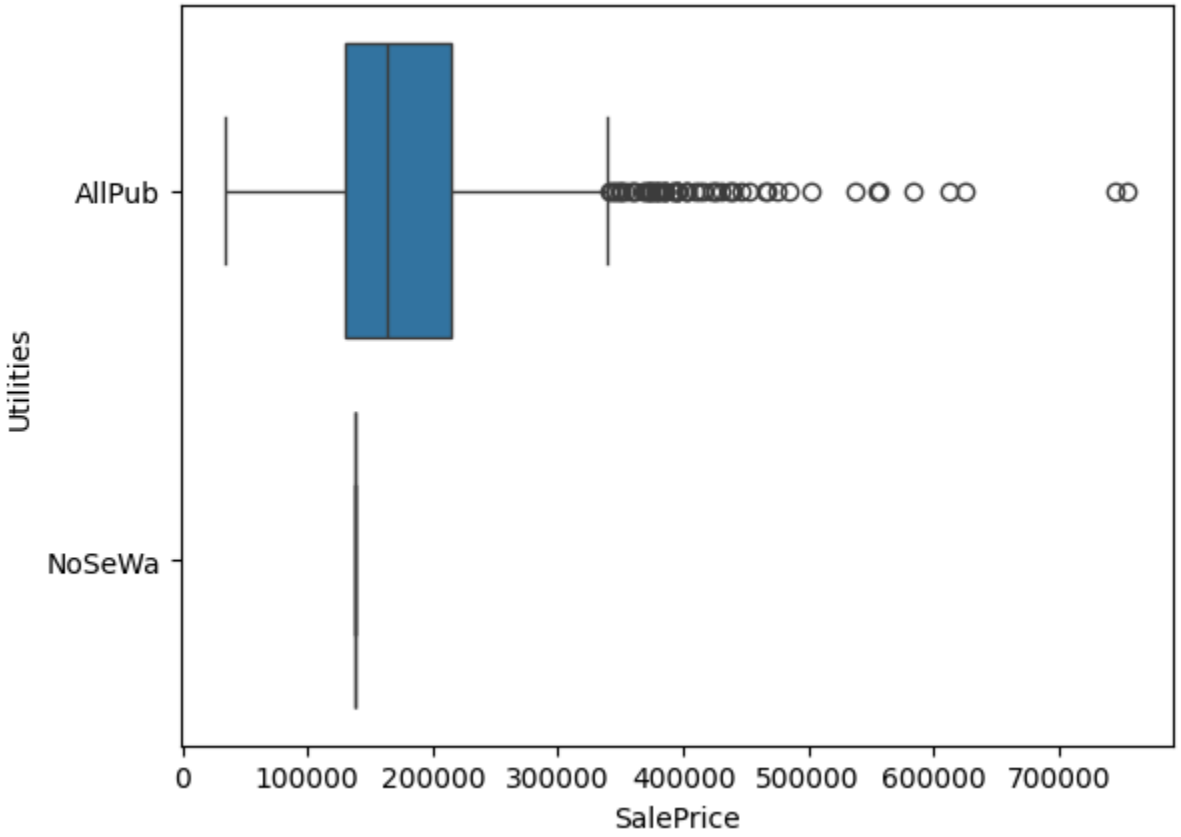
LotShape



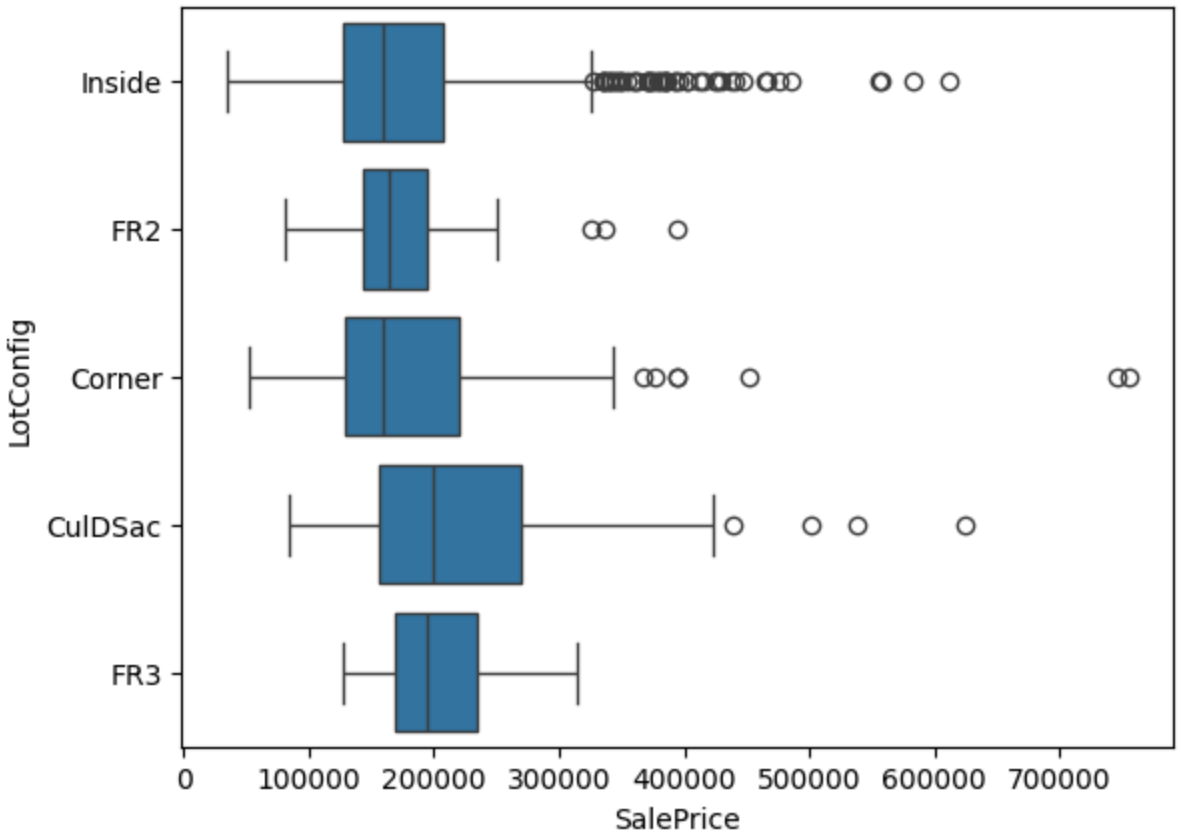
LandContour



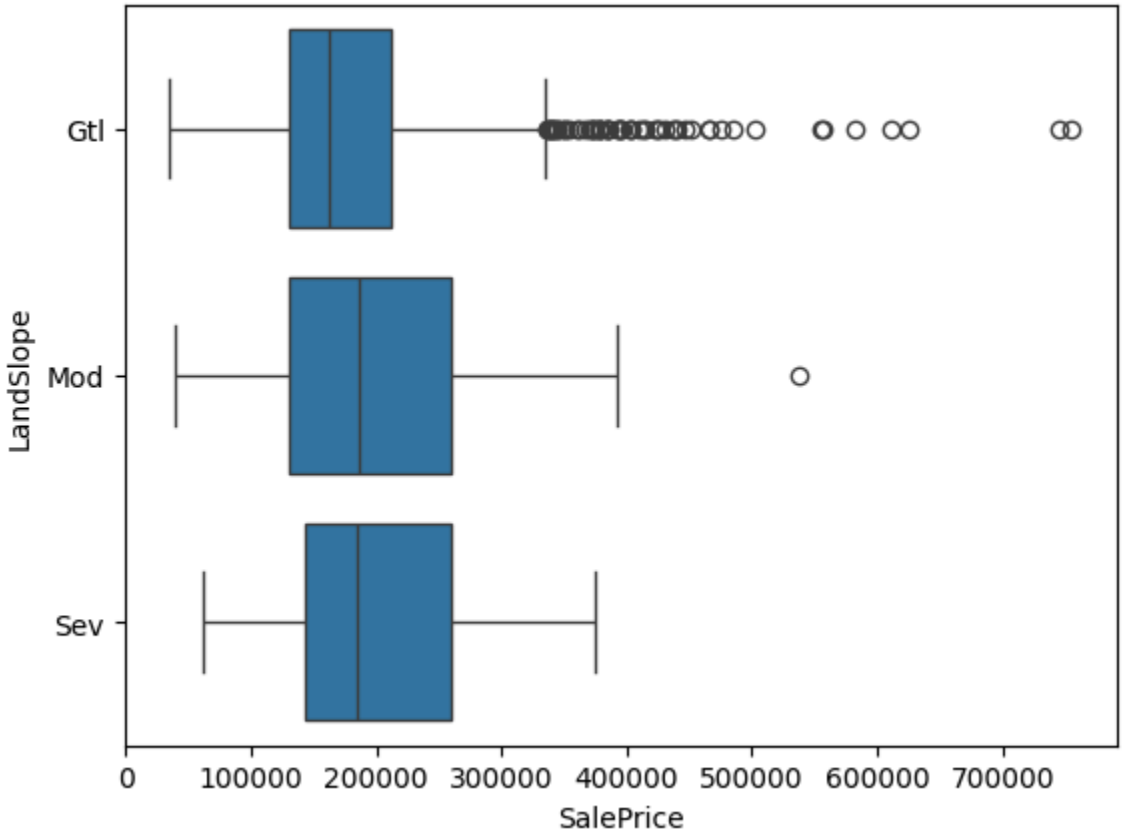
Utilities



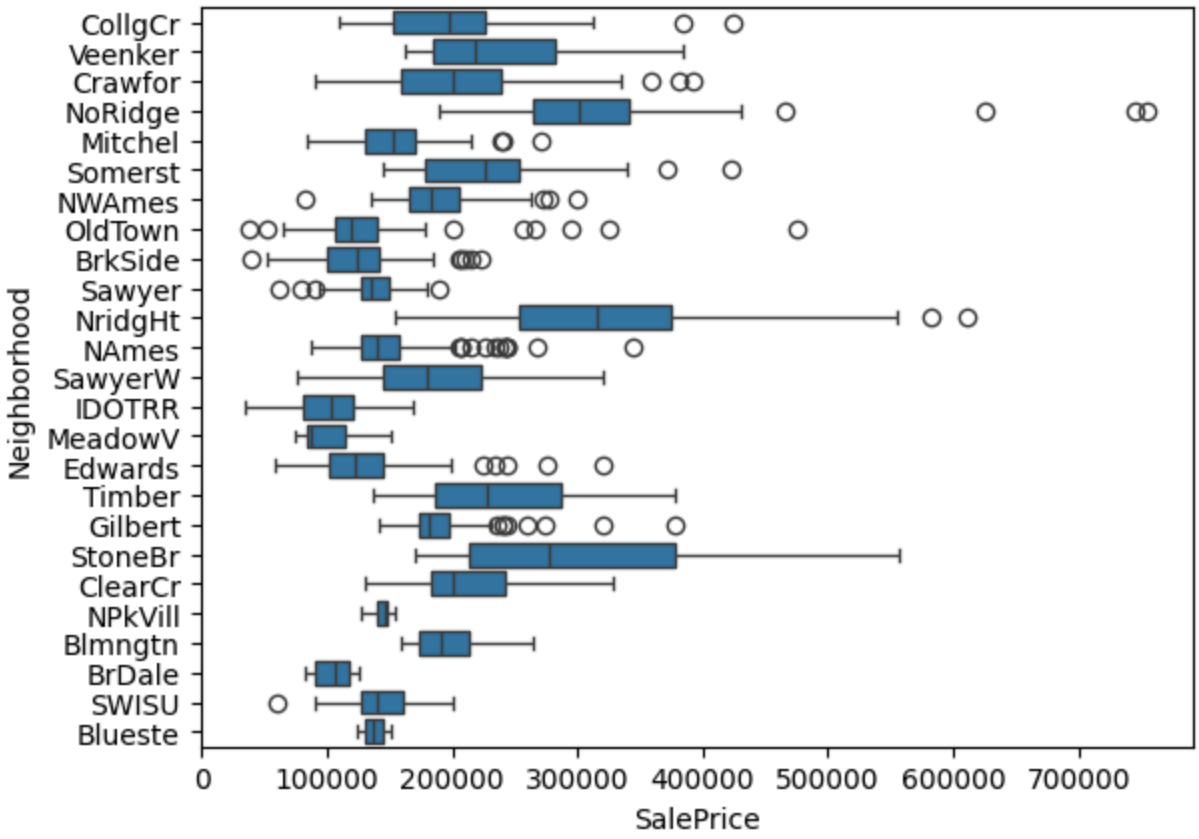
LotConfig



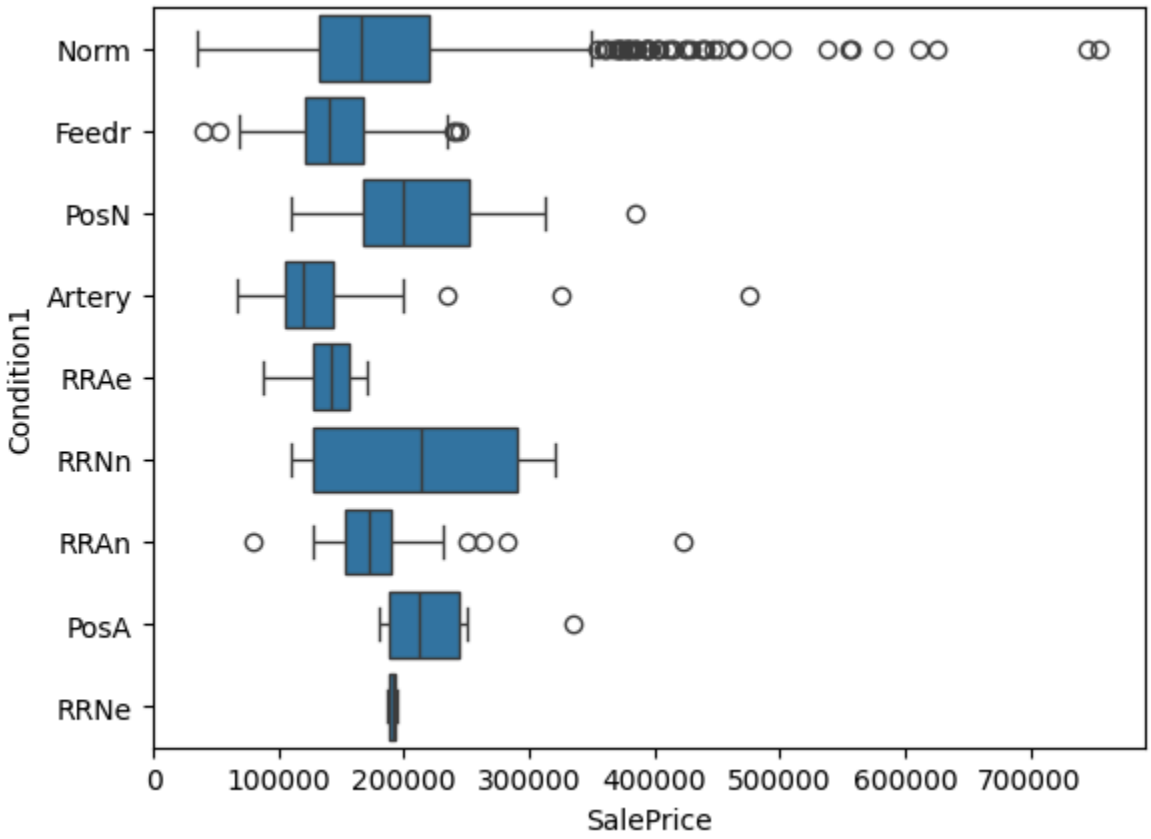
LandSlope



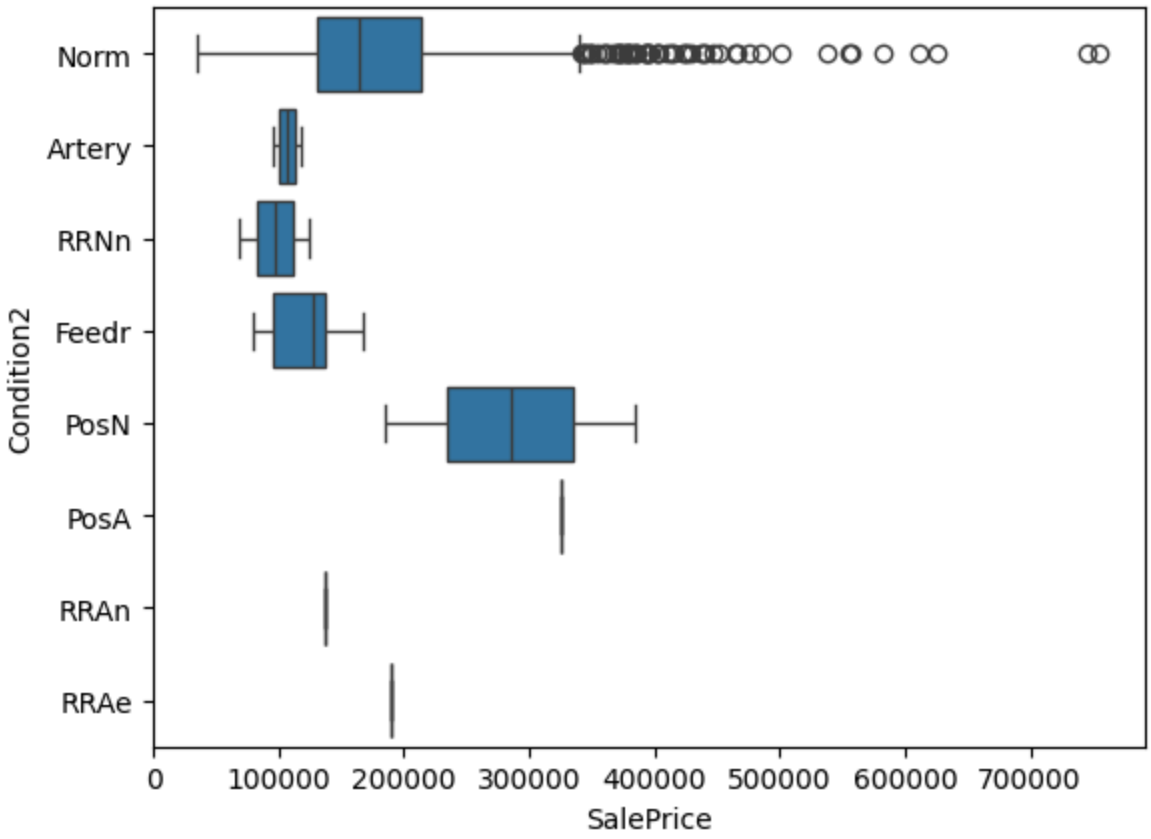
Neighborhood



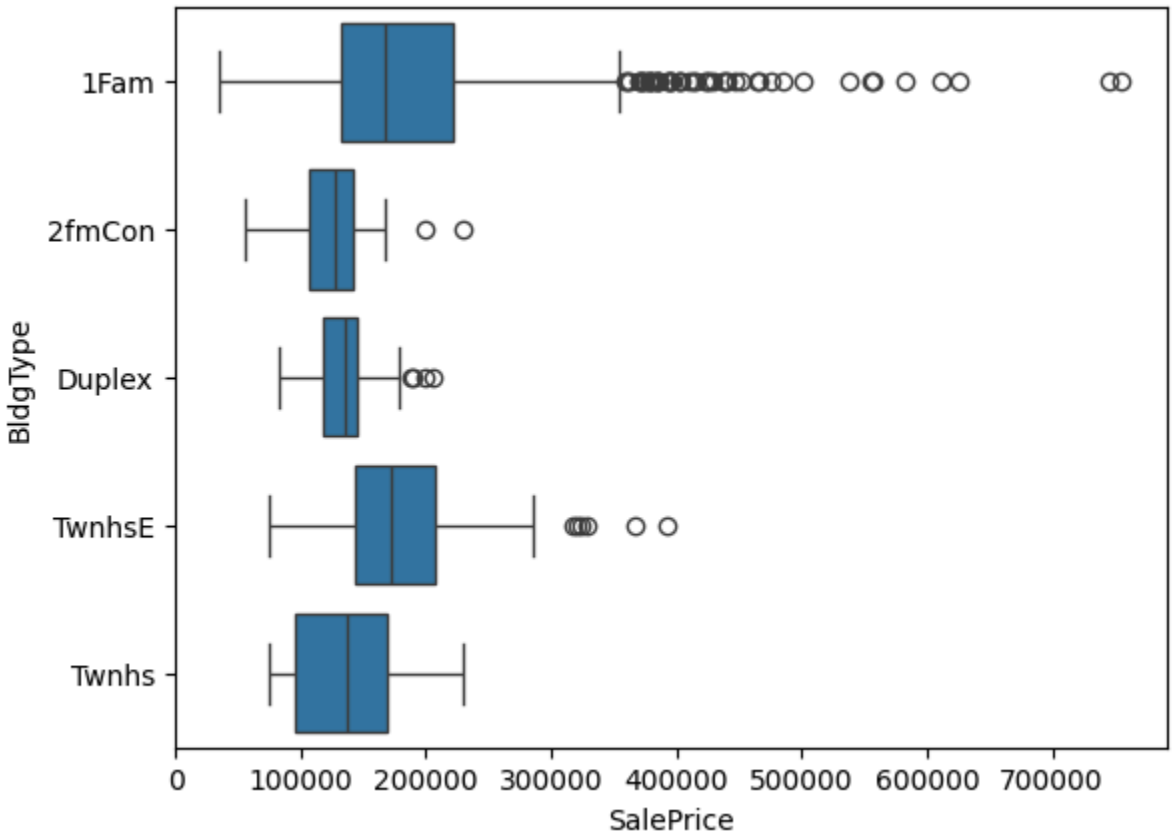
Condition1



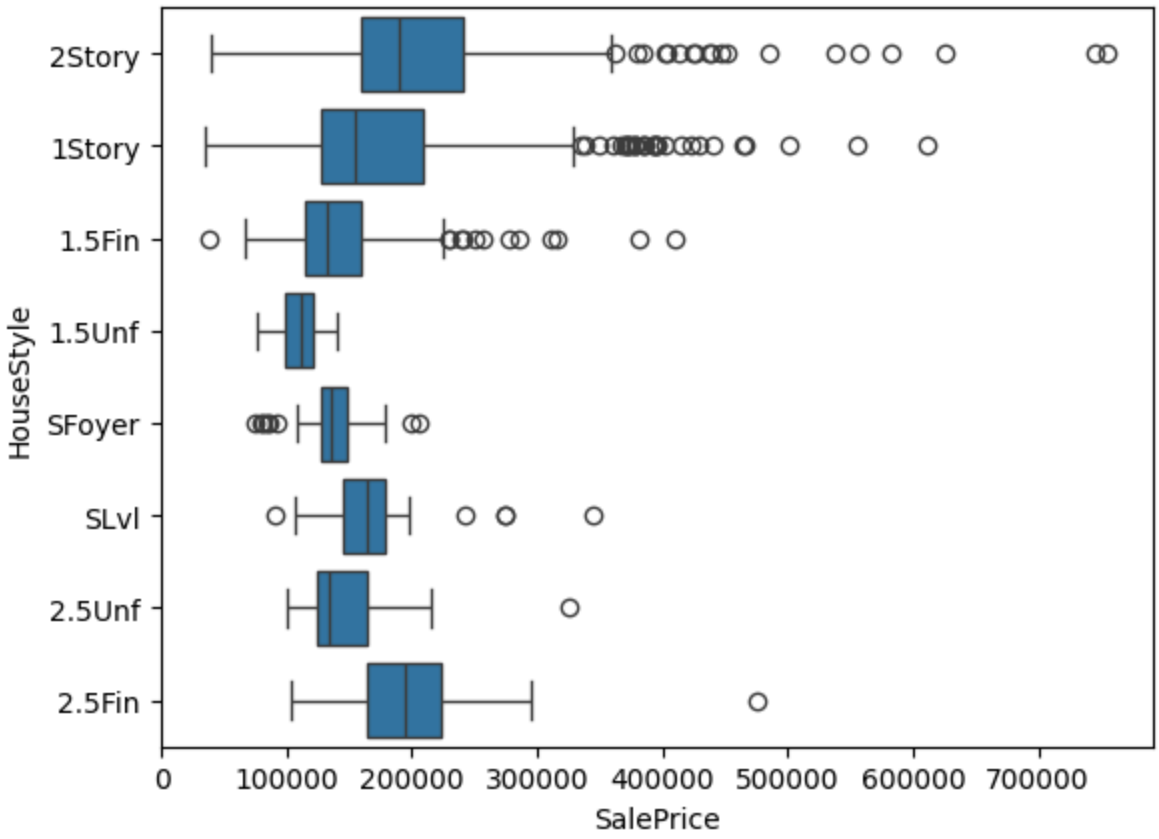
Condition2



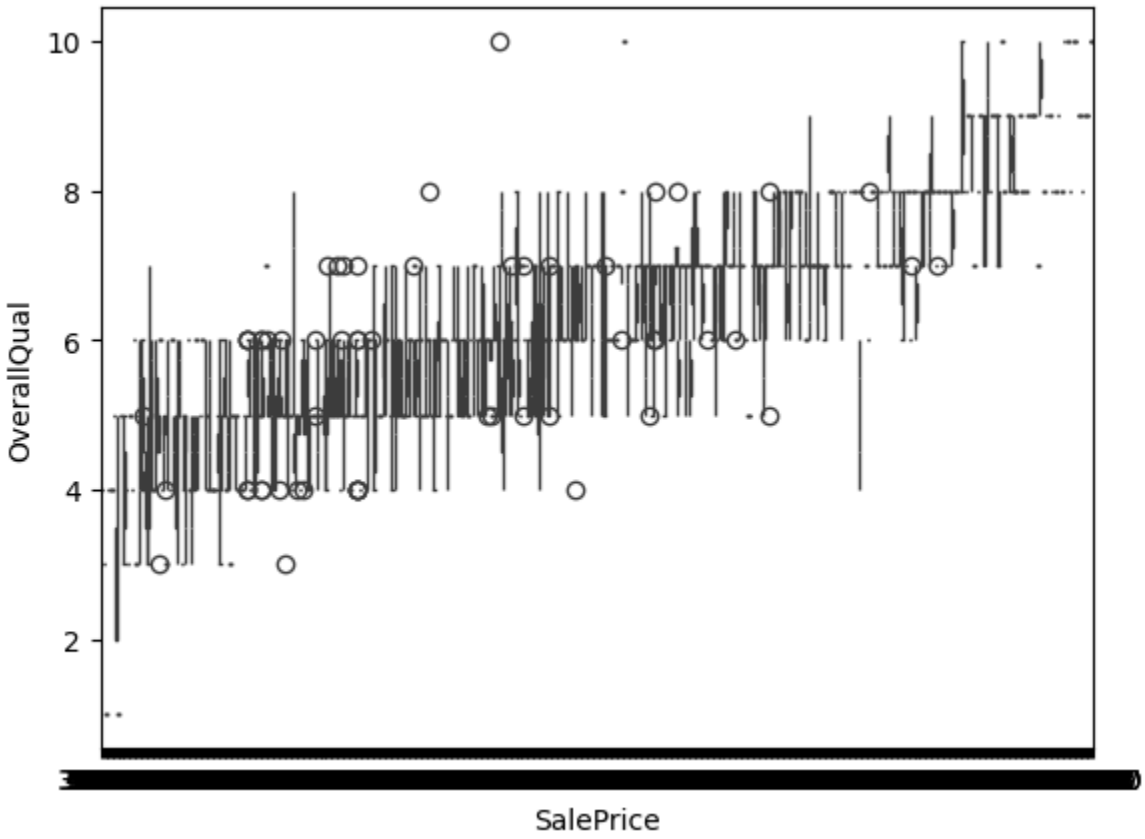
`BldgType`



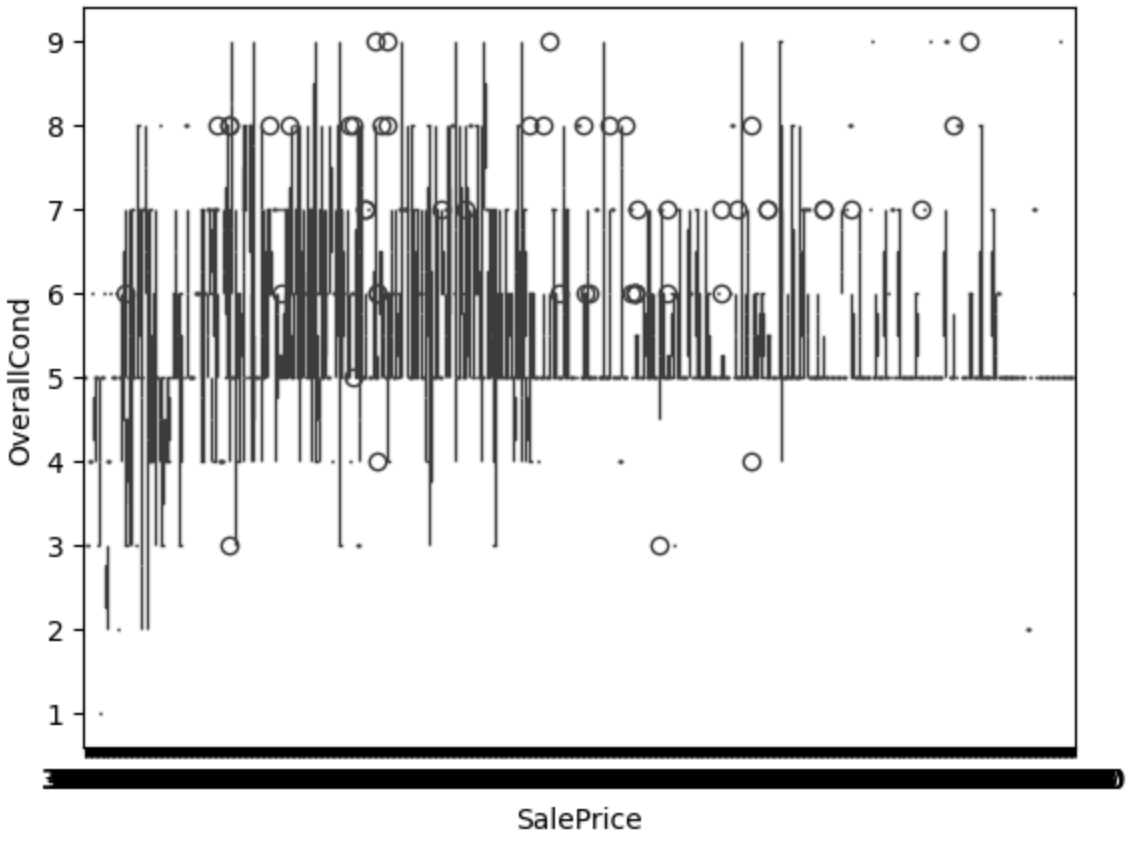
`HouseStyle`



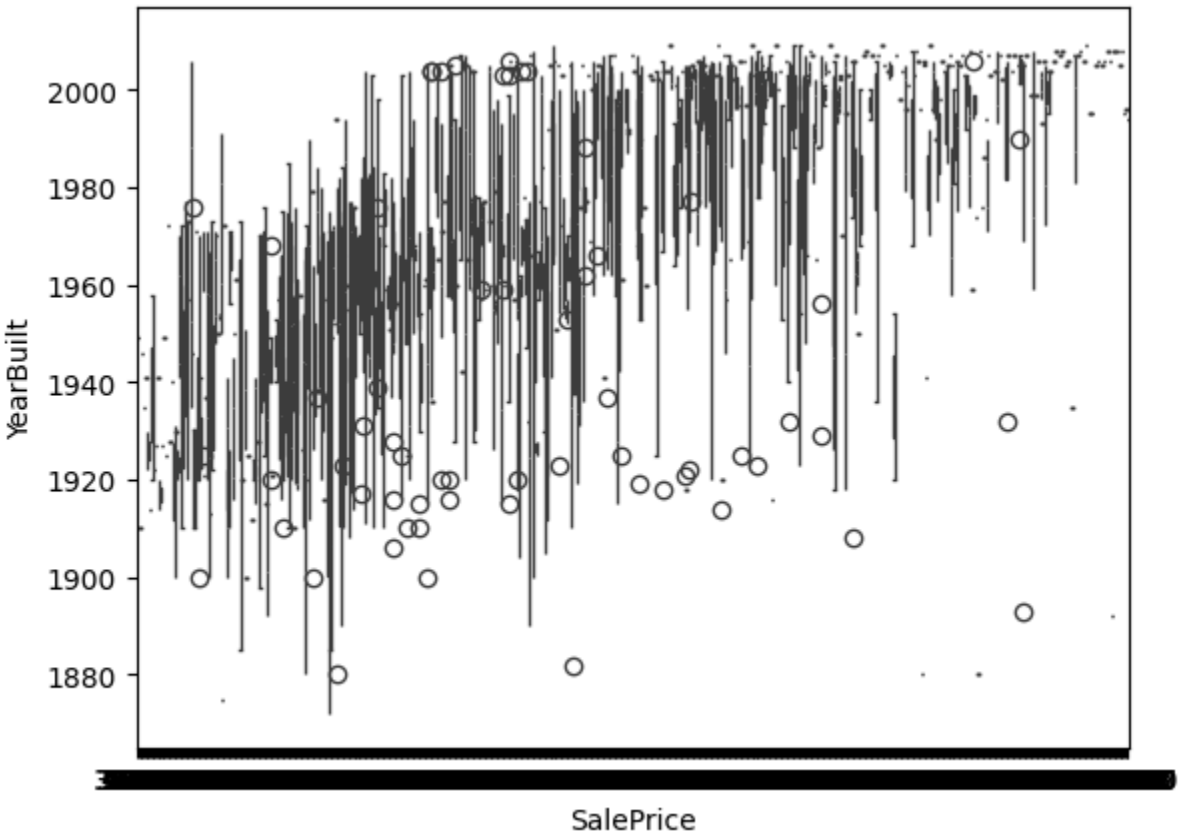
OverallQual



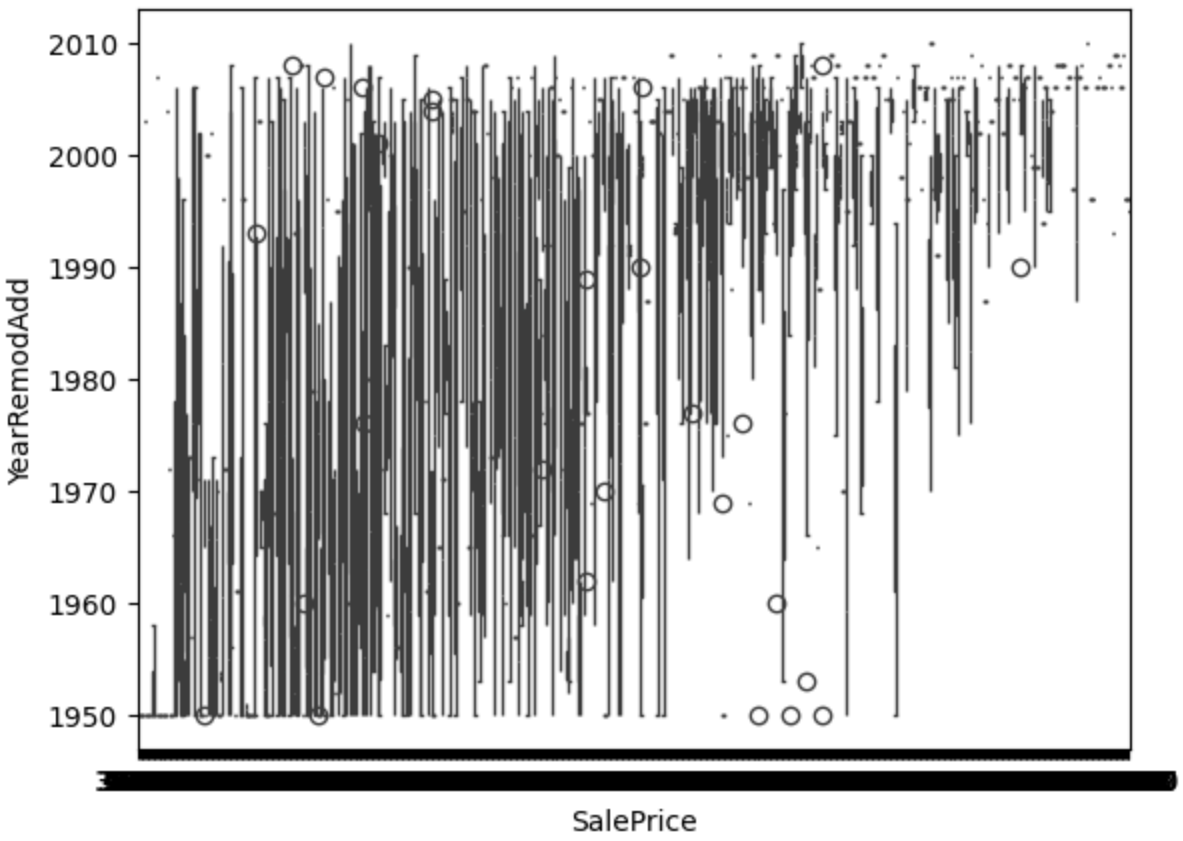
OverallCond



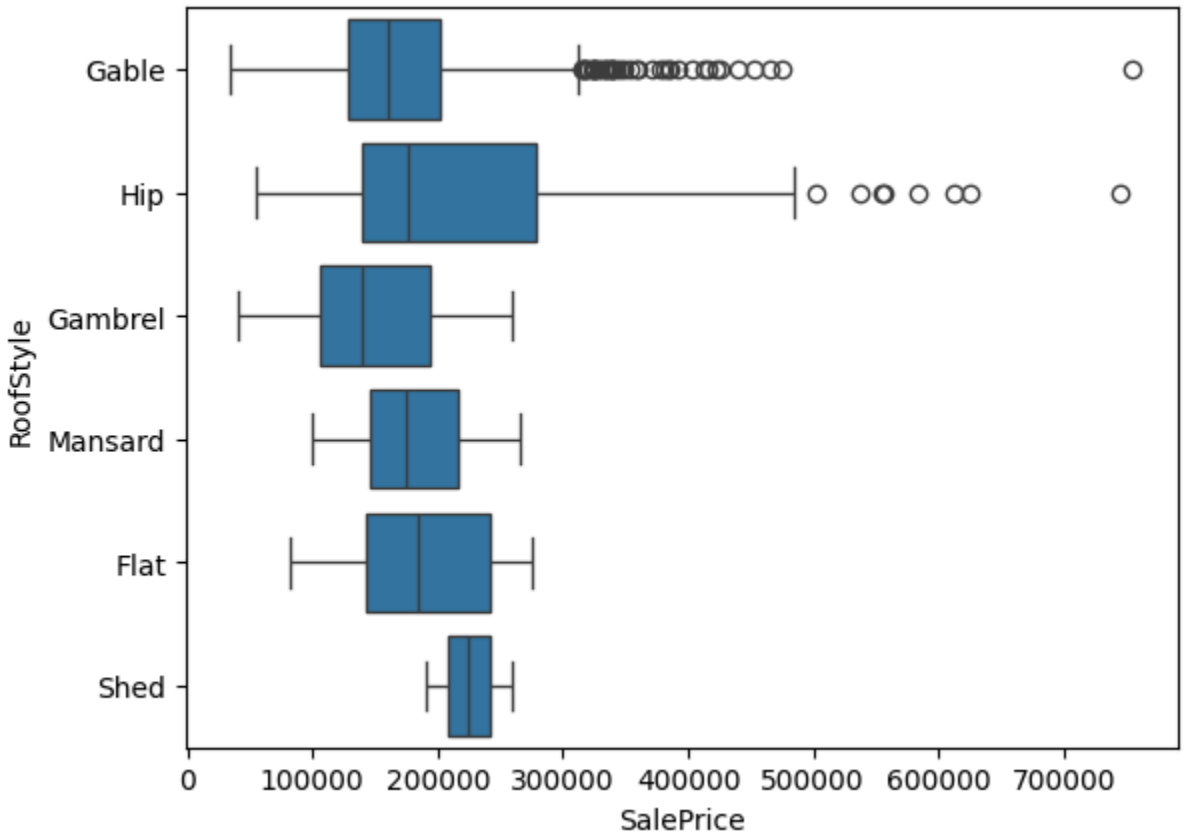
YearBuilt



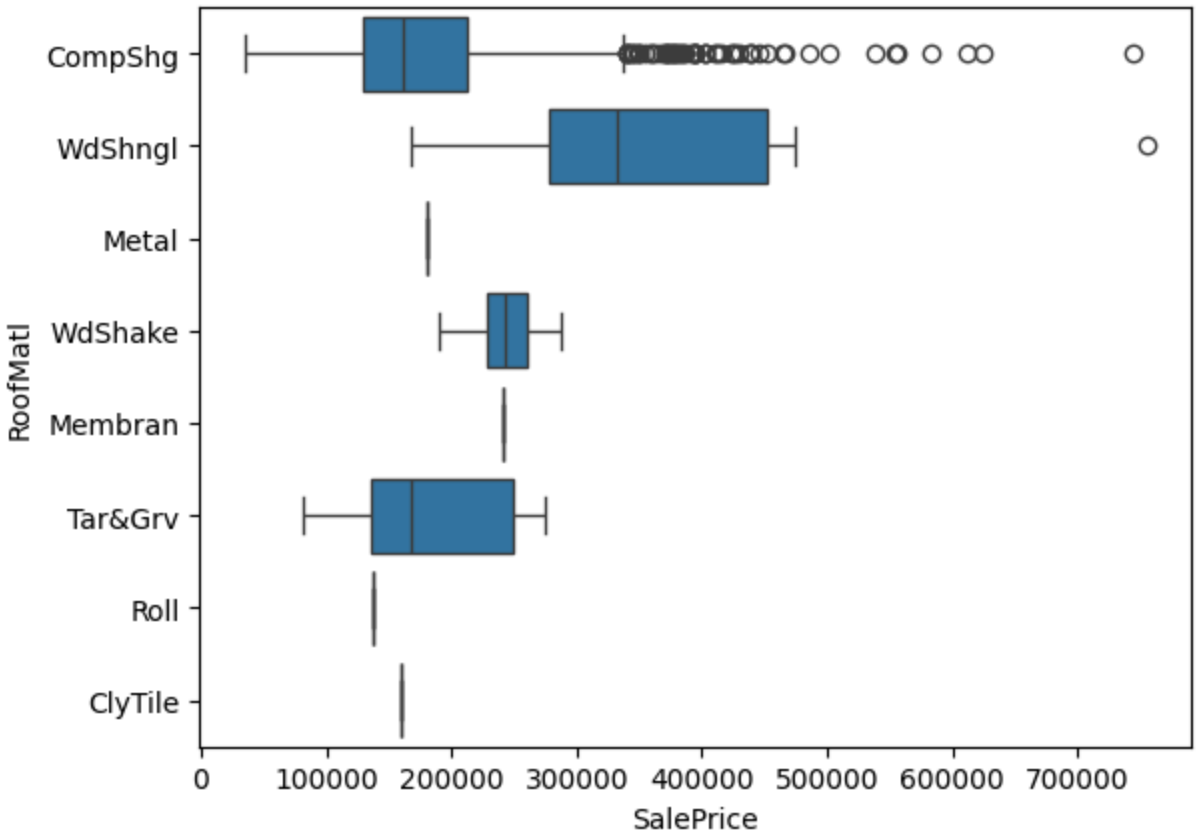
YearRemodAdd



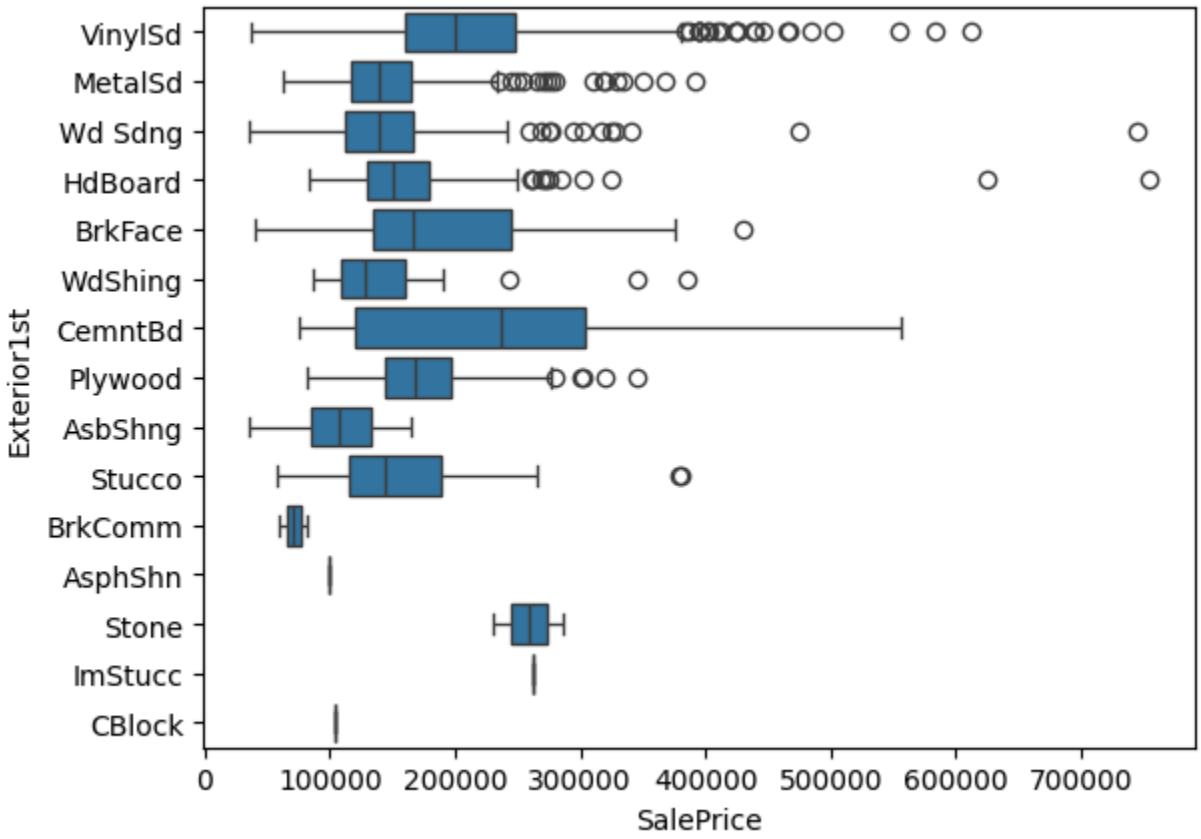
RoofStyle



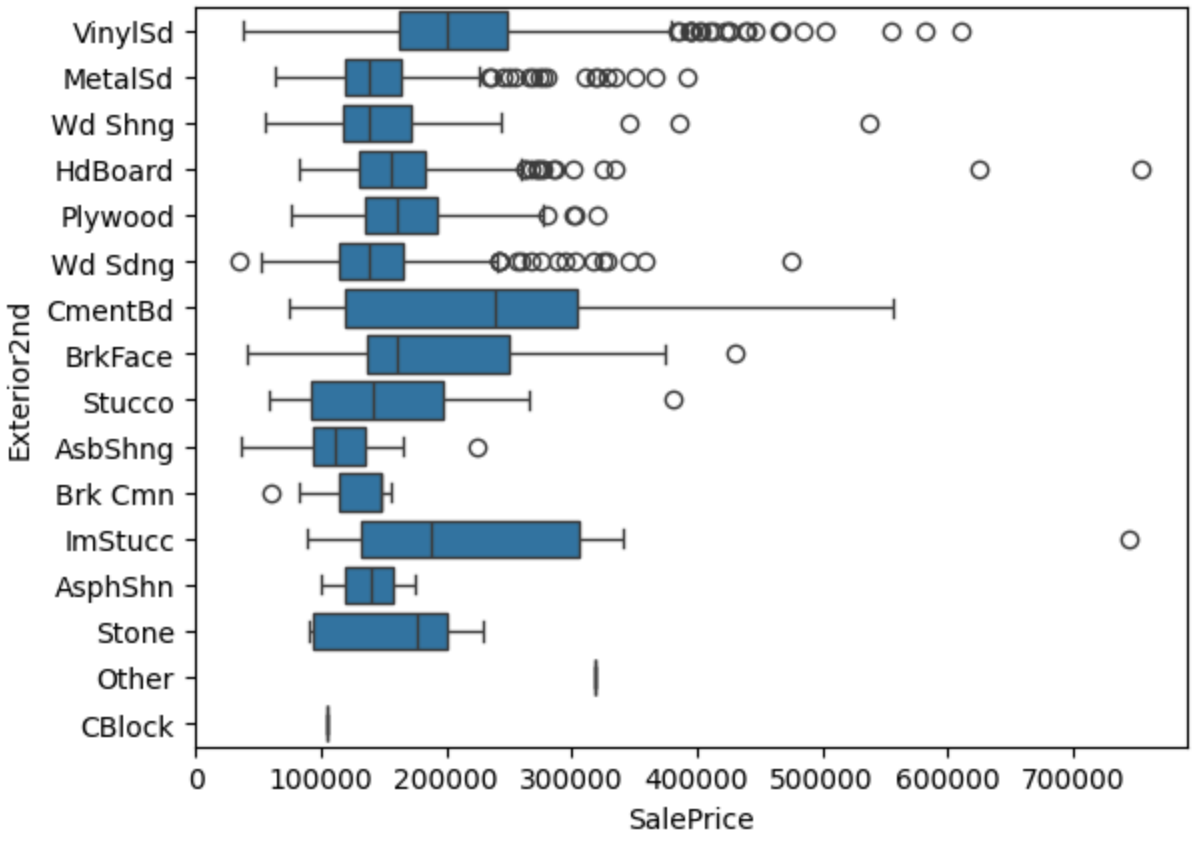
RoofMatl



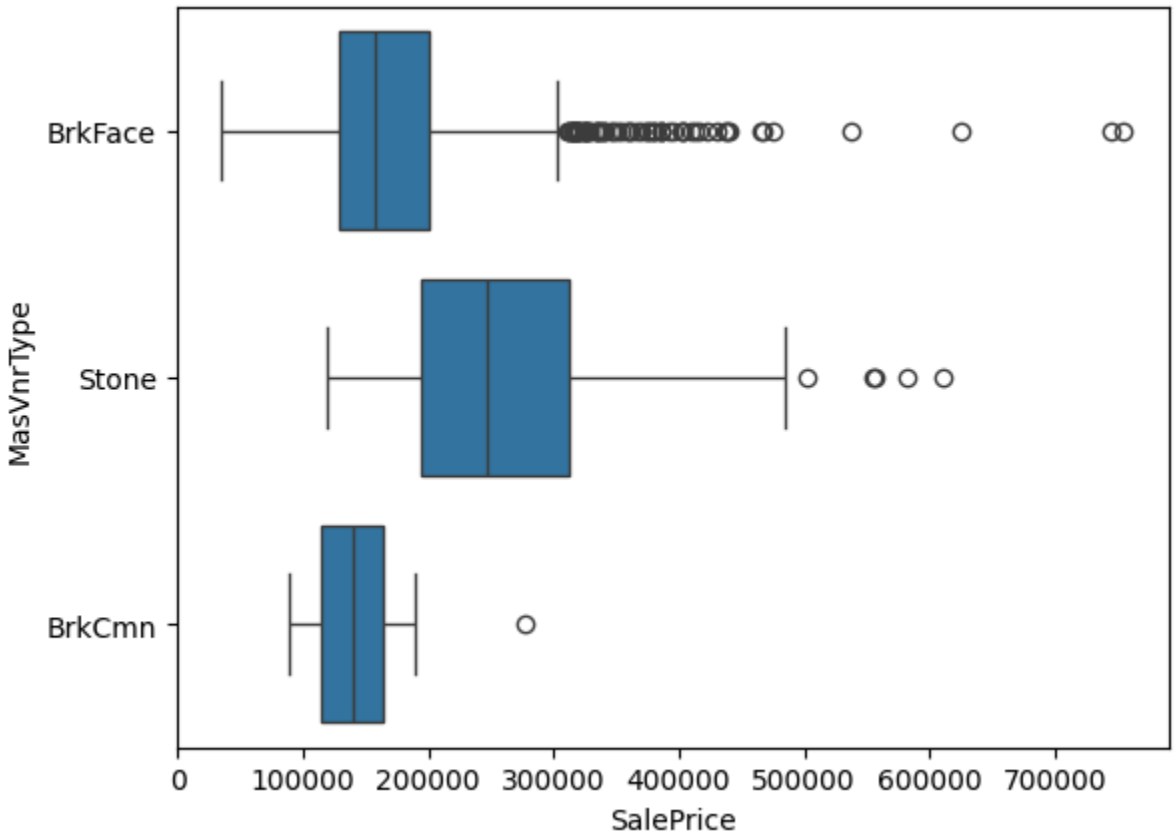
Exterior1st



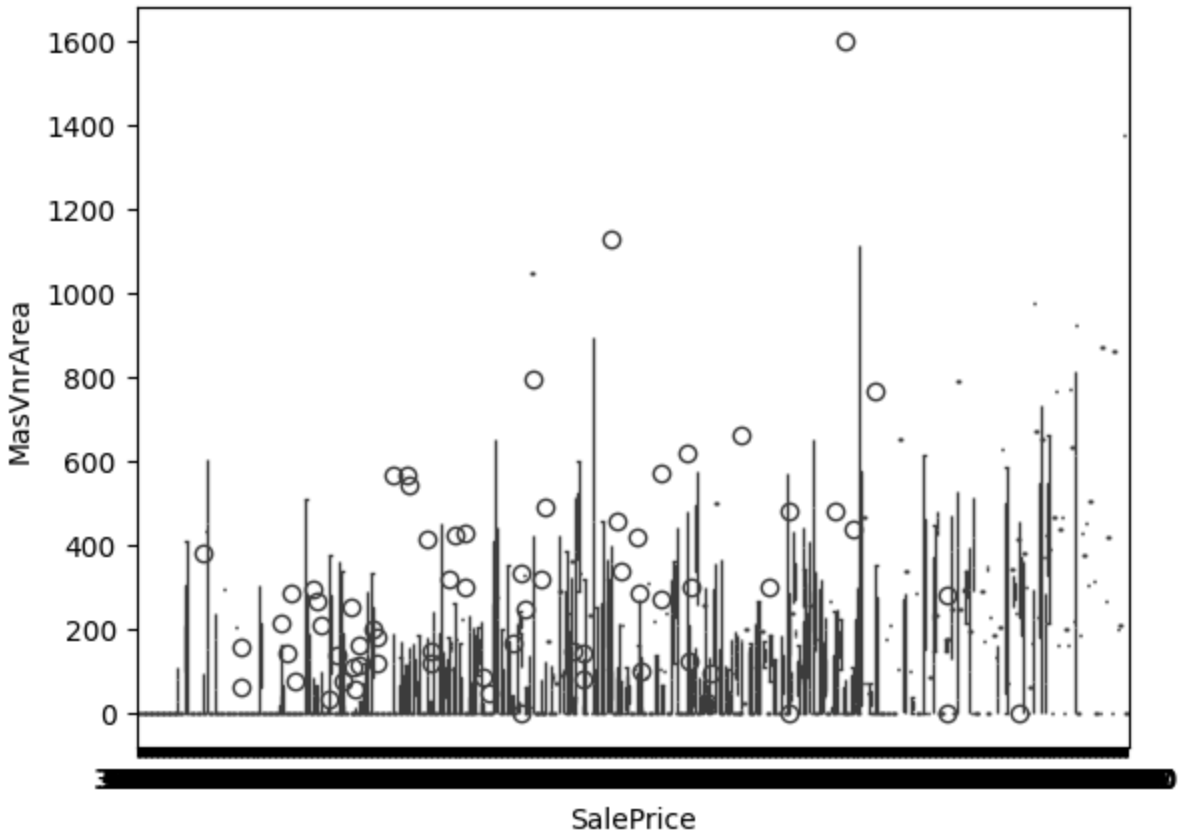
Exterior2nd



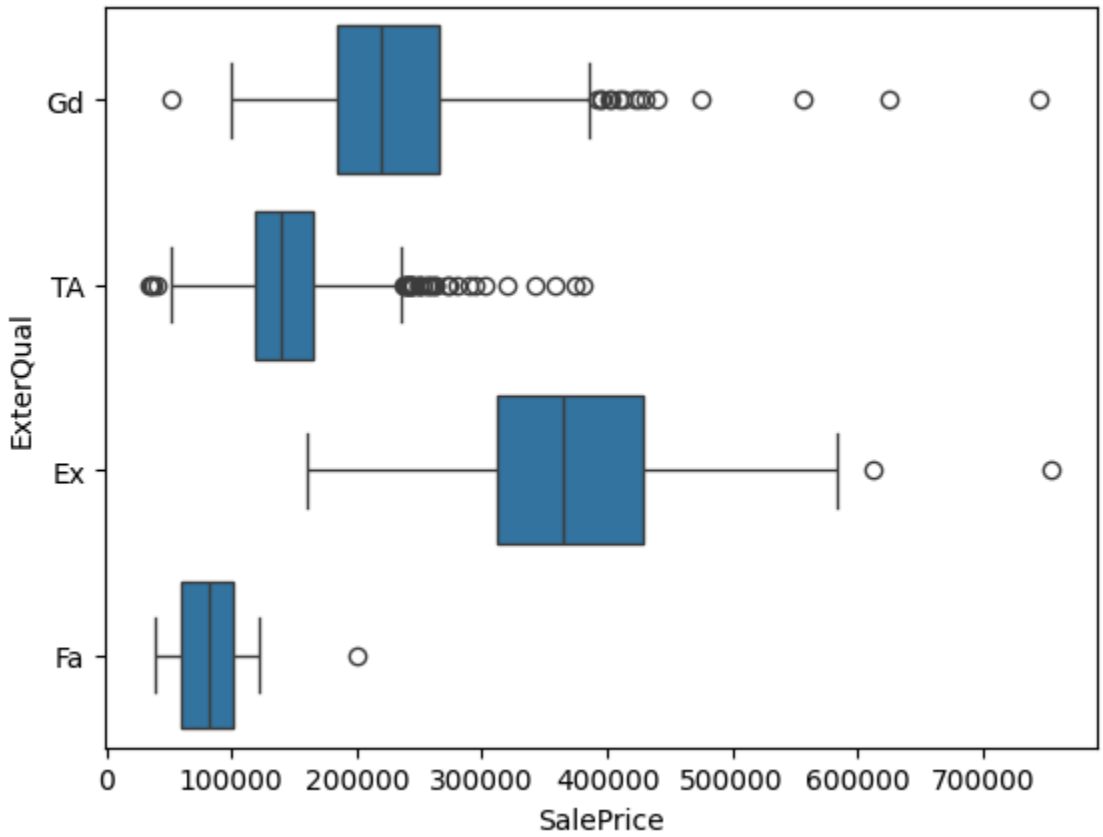
`MasVnrType`



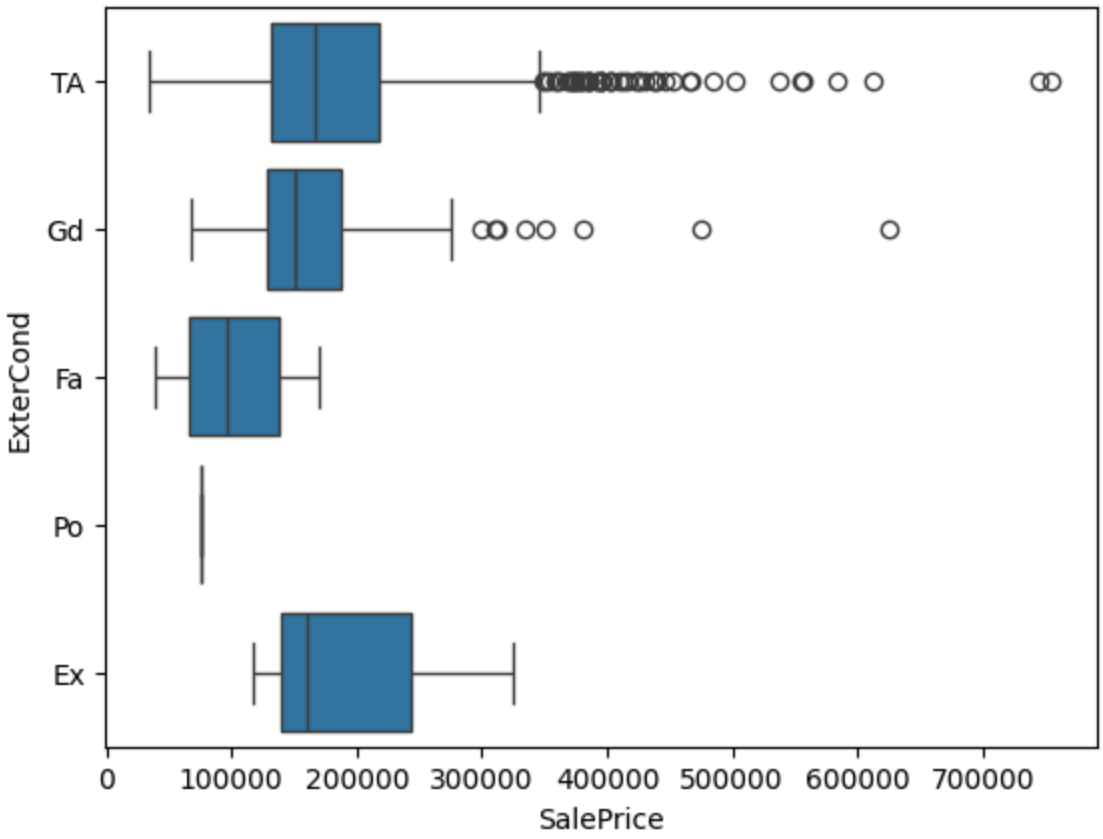
`MasVnrArea`



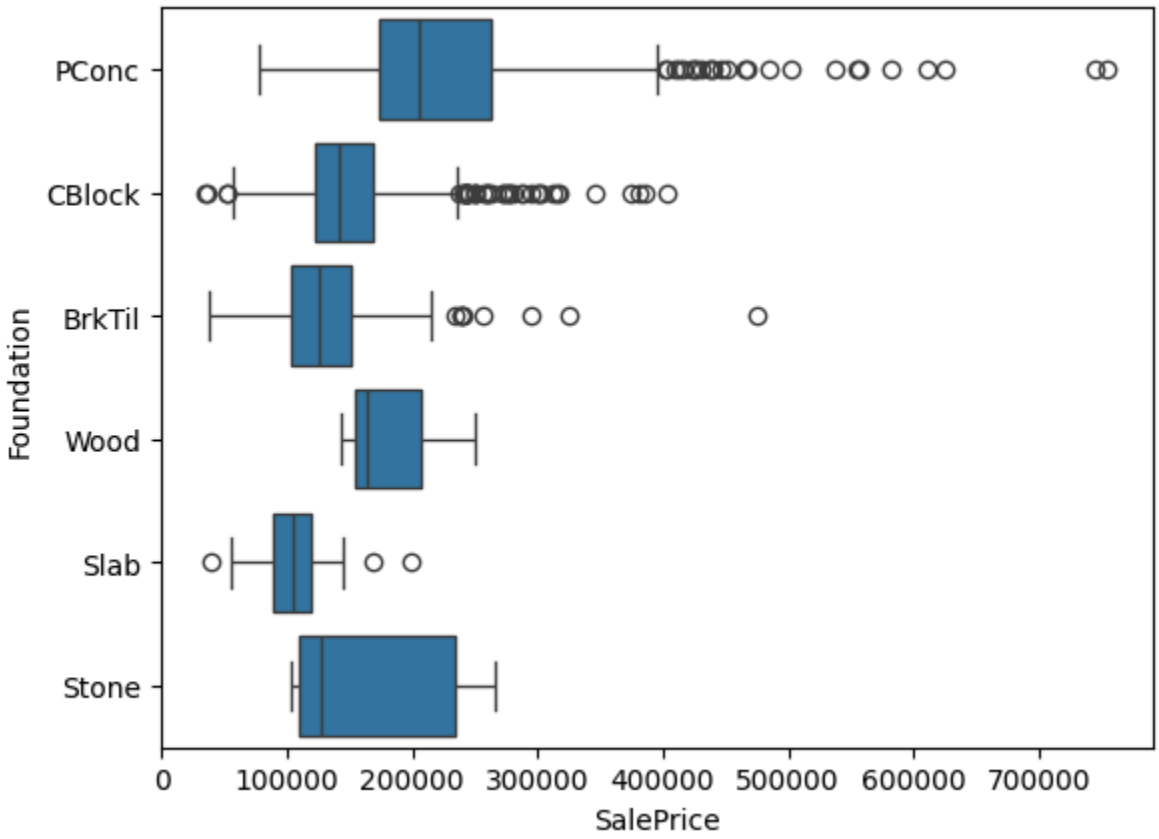
ExterQual



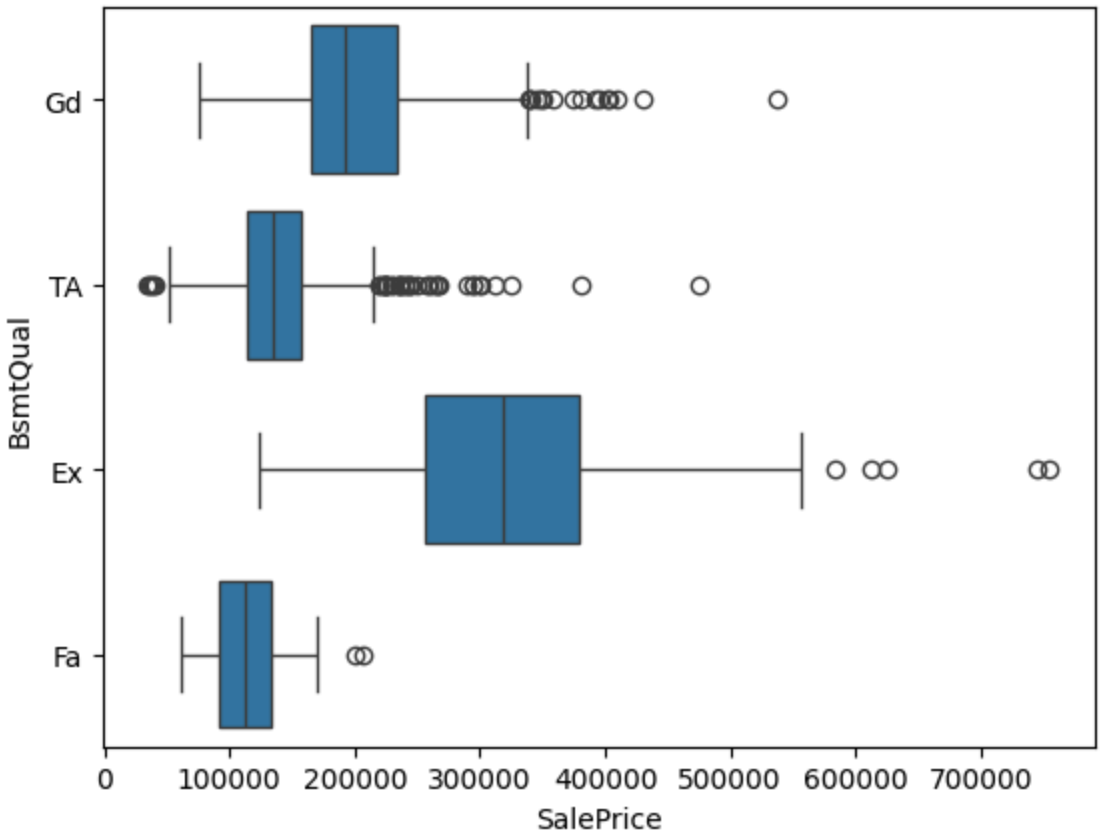
ExterCond



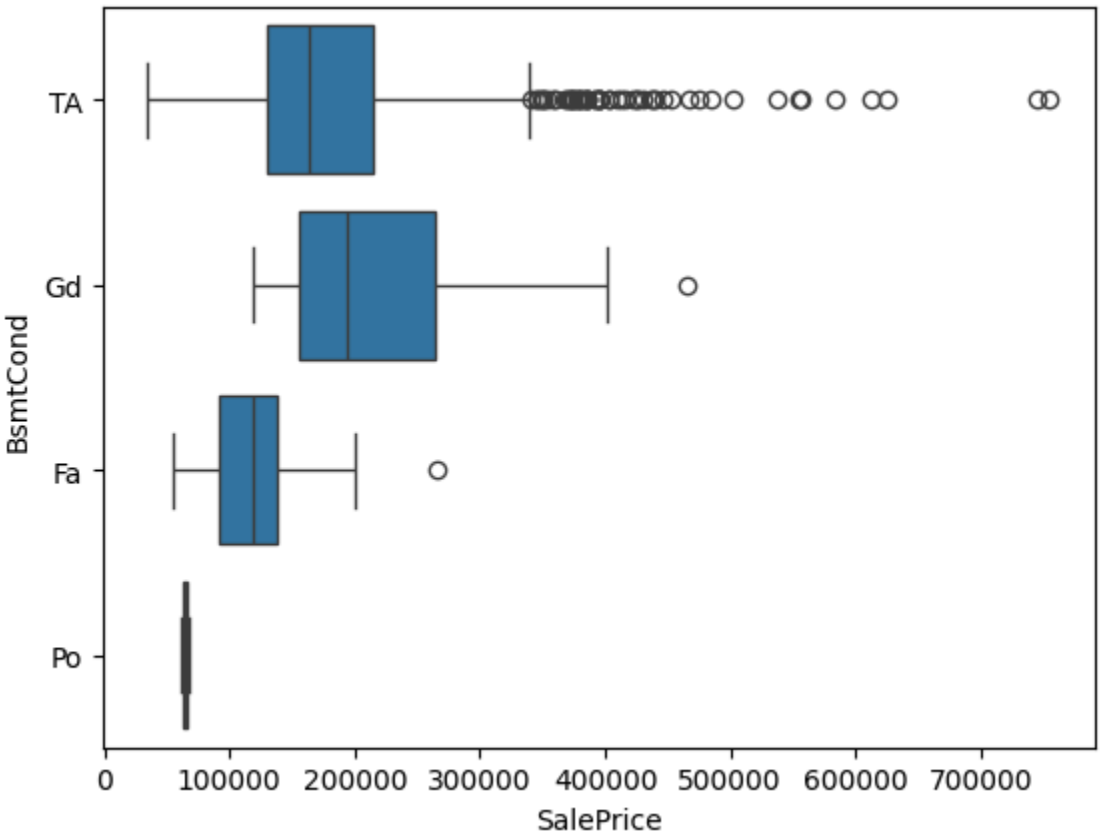
Foundation



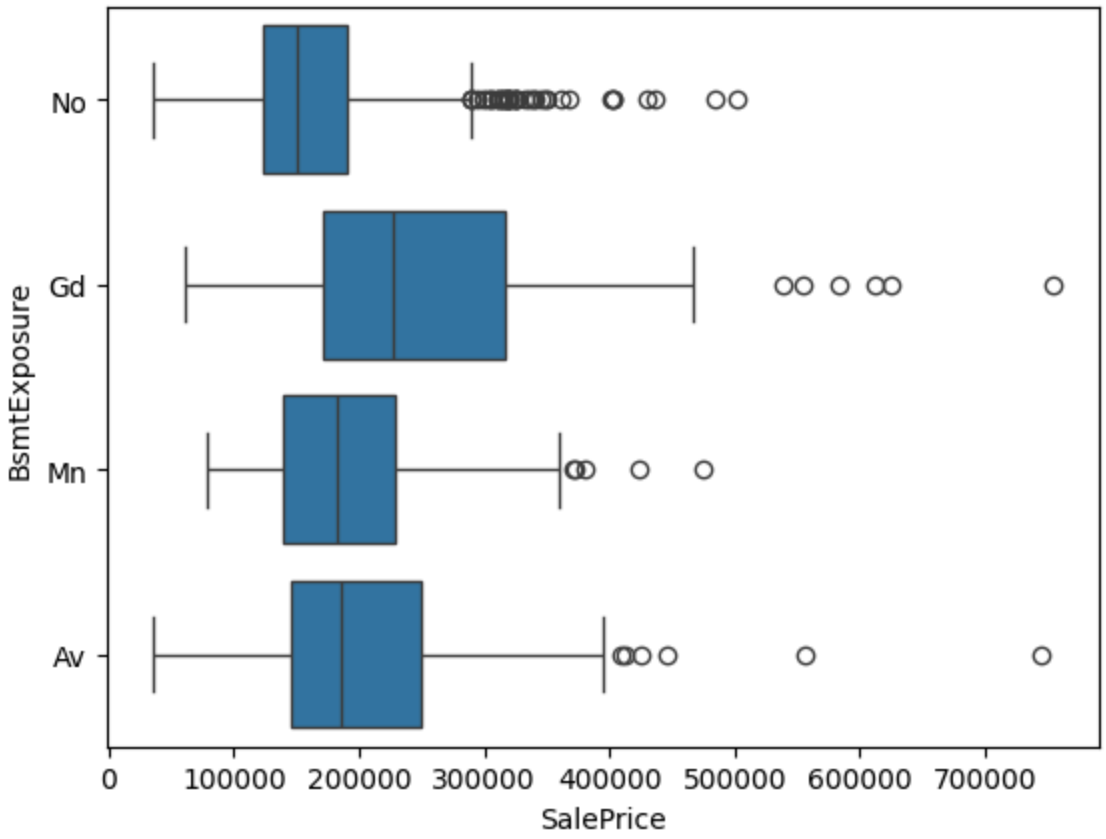
BsmtQual



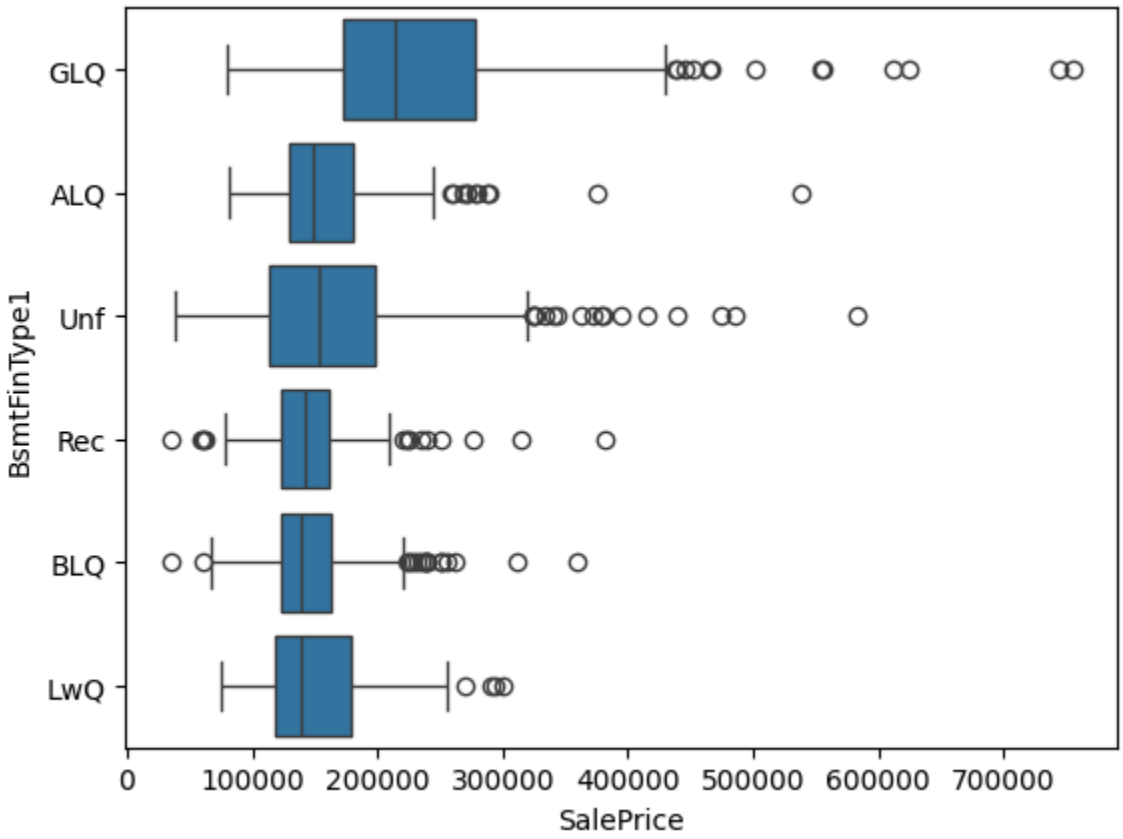
BsmtCond



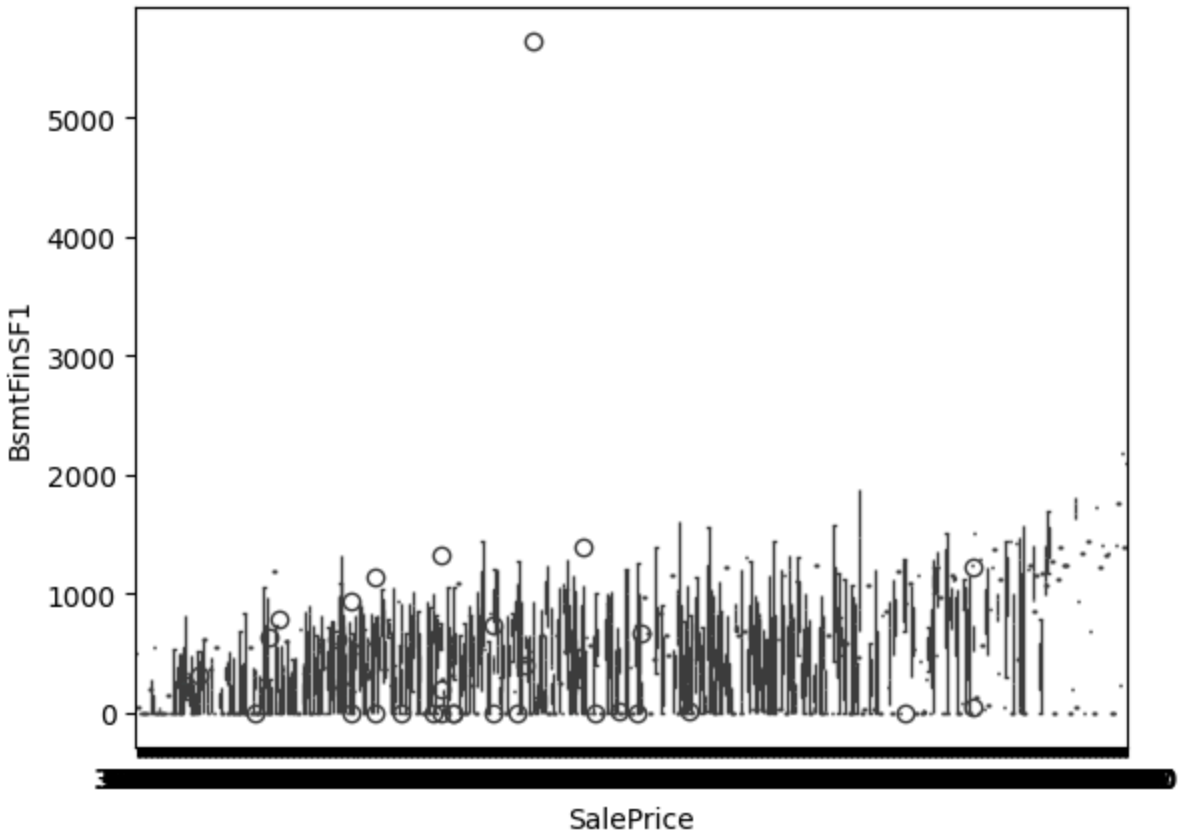
BsmtExposure



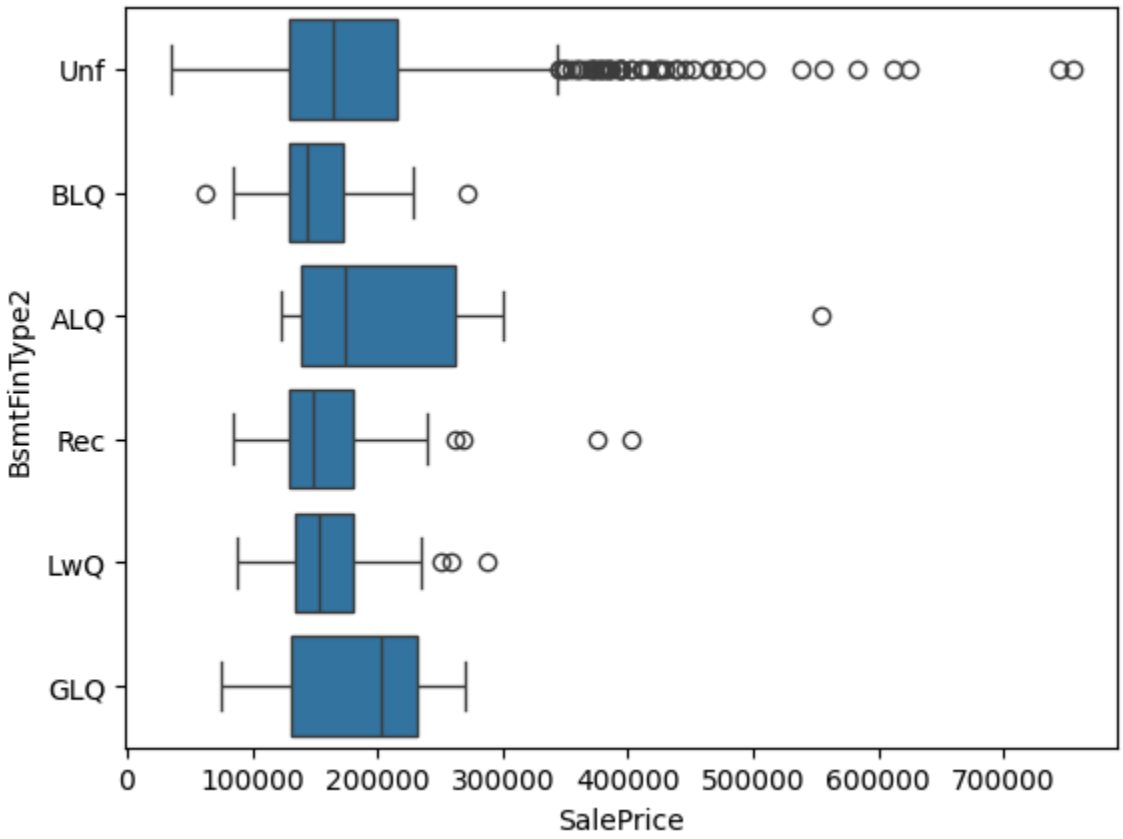
BsmtFinType1



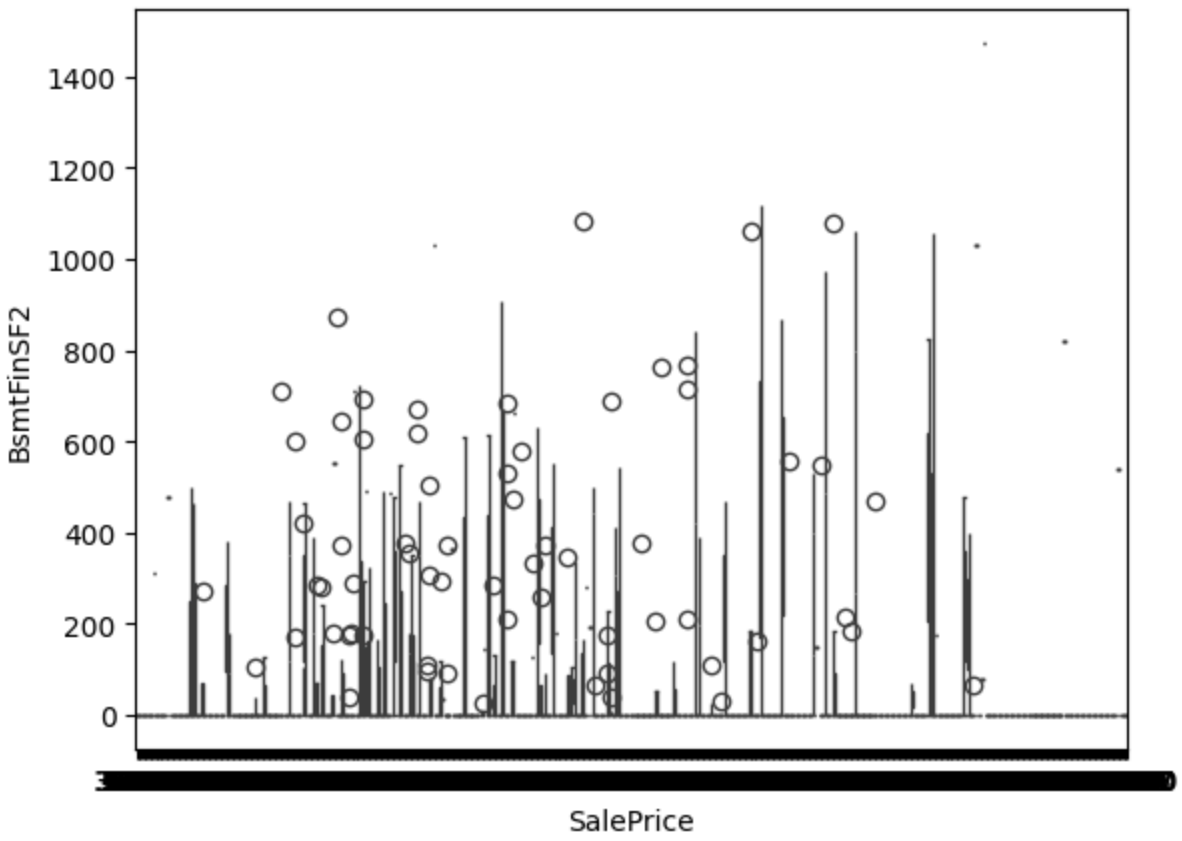
BsmtFinSF1



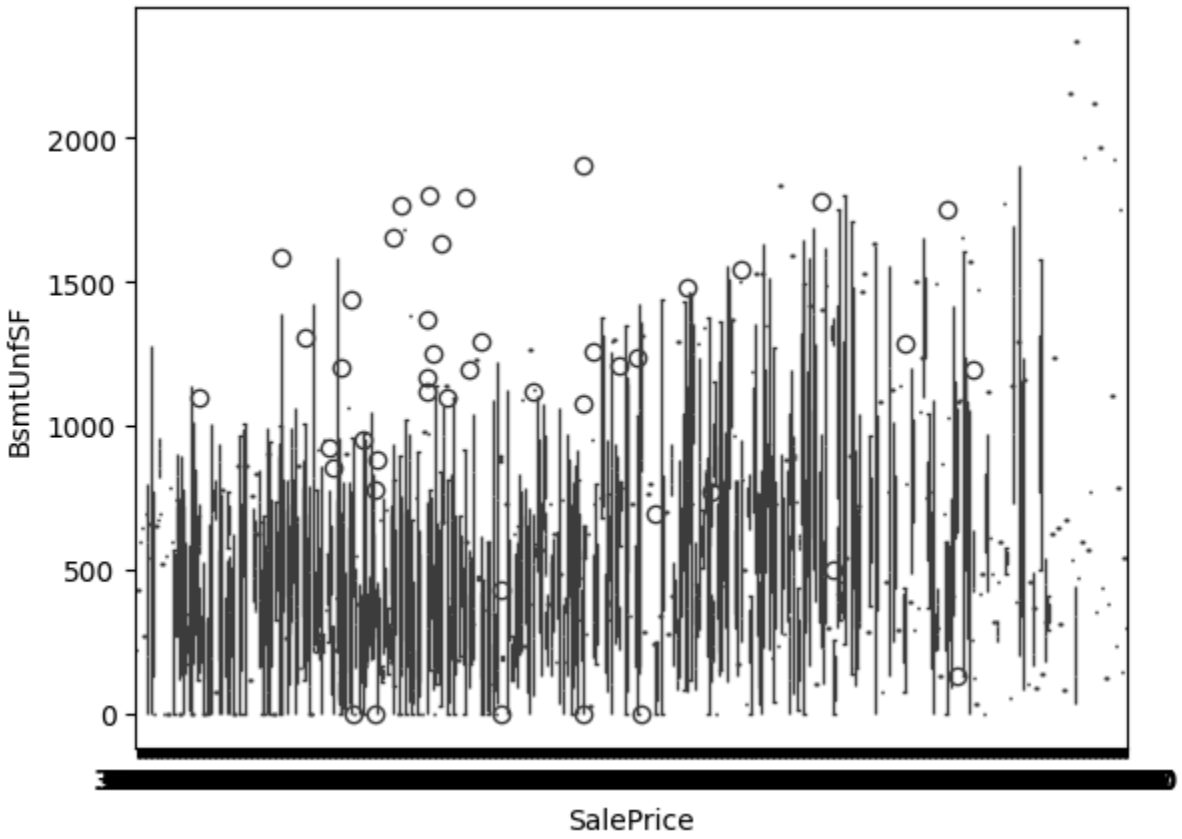
BsmtFinType2



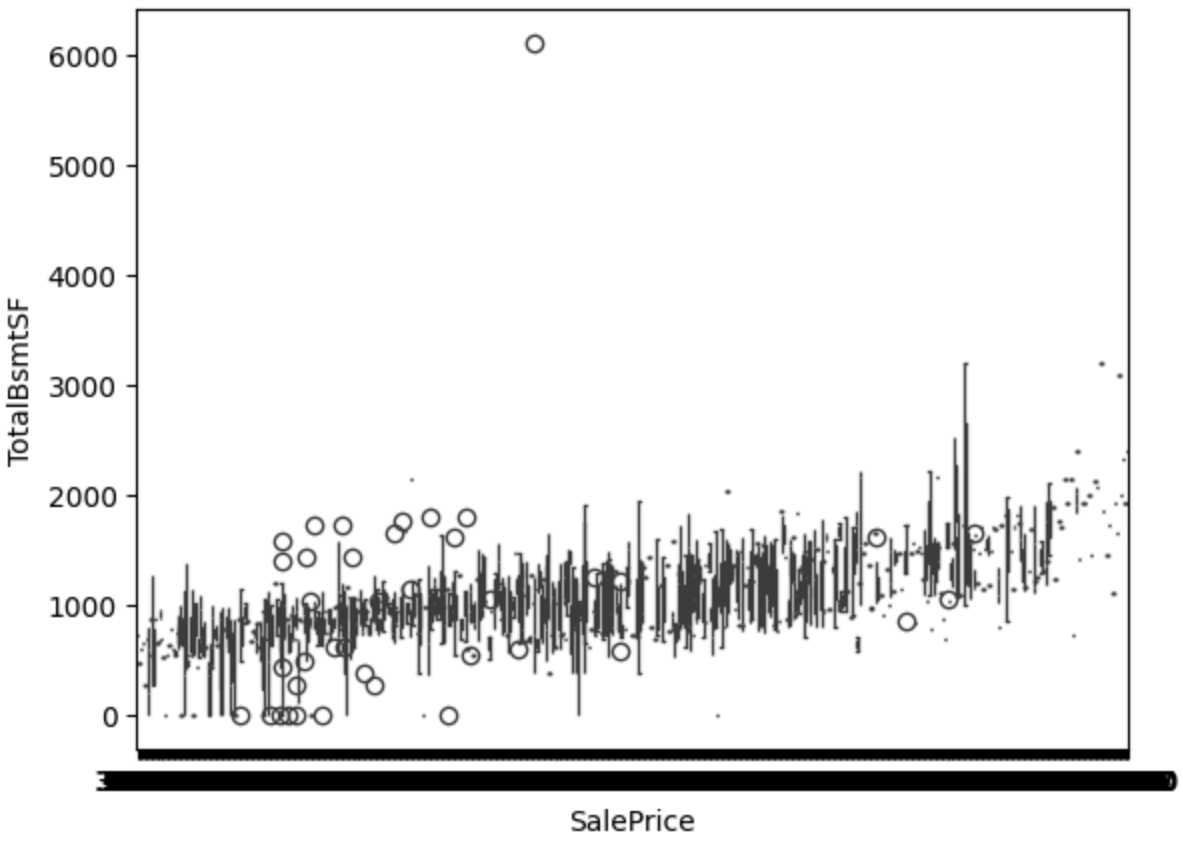
BsmtFinSF2



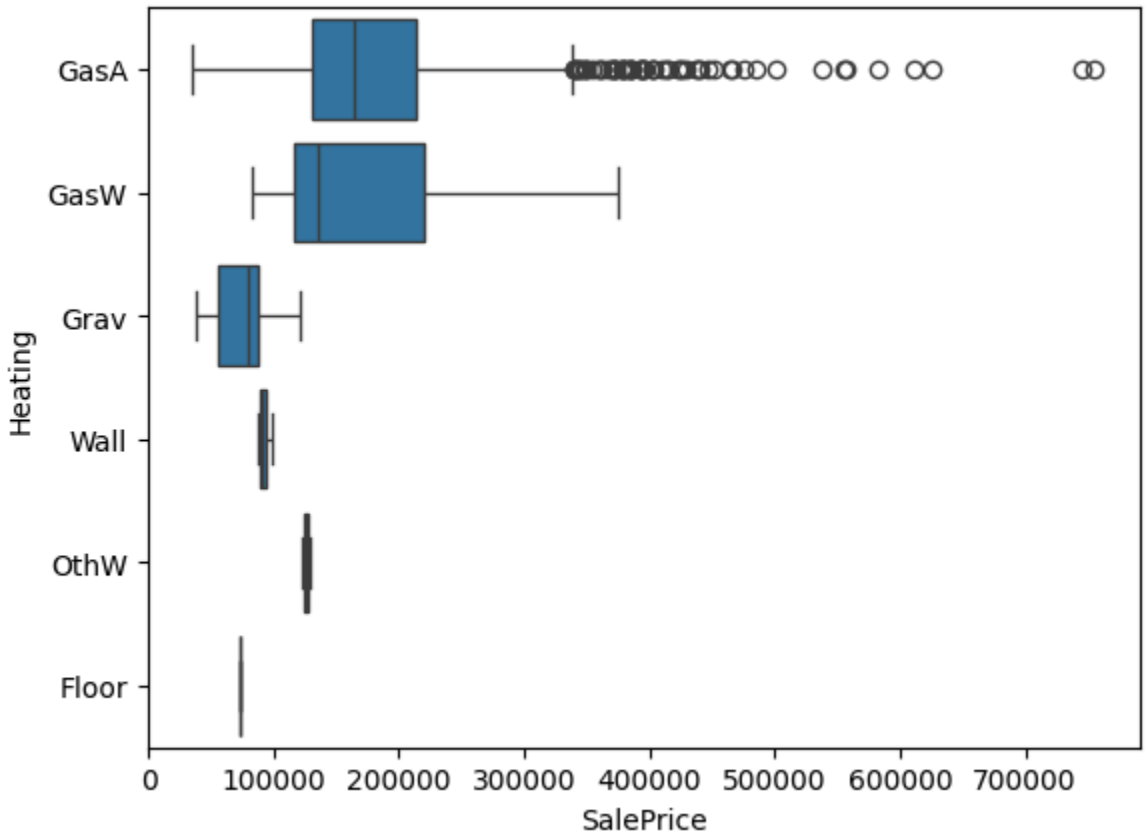
BsmtUnfSF



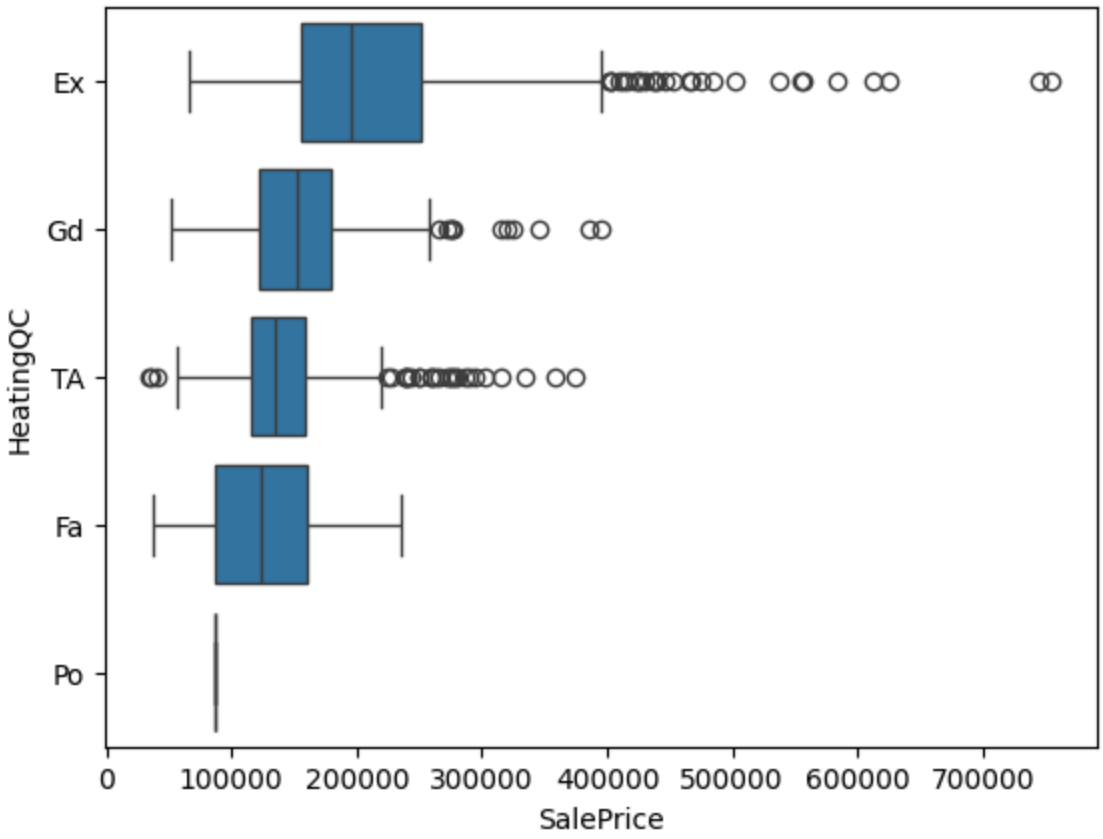
TotalBsmtSF



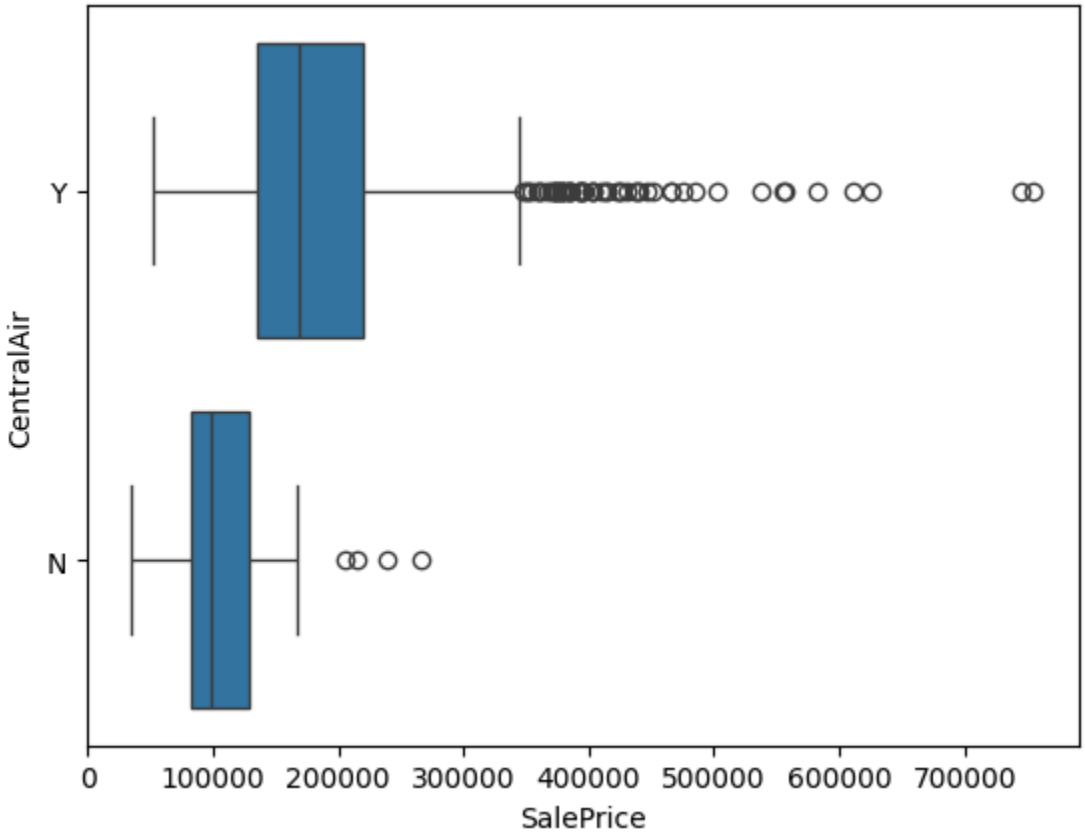
Heating



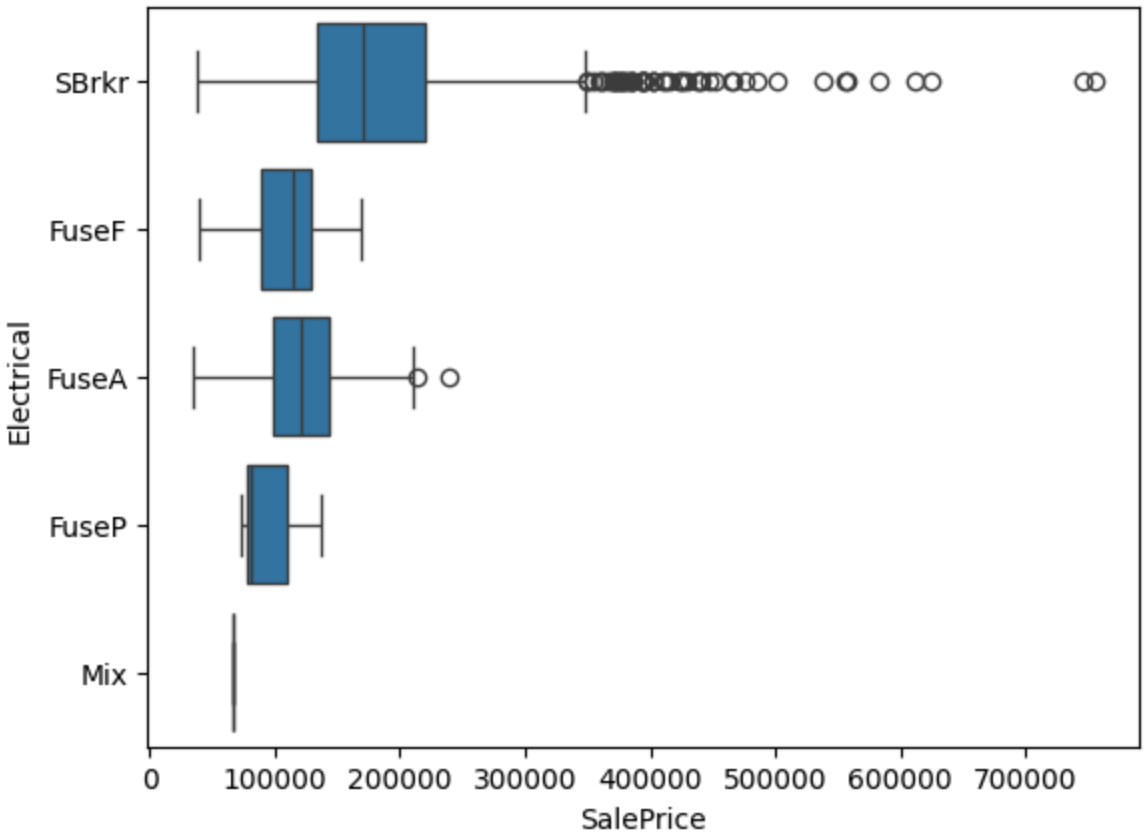
HeatingQC



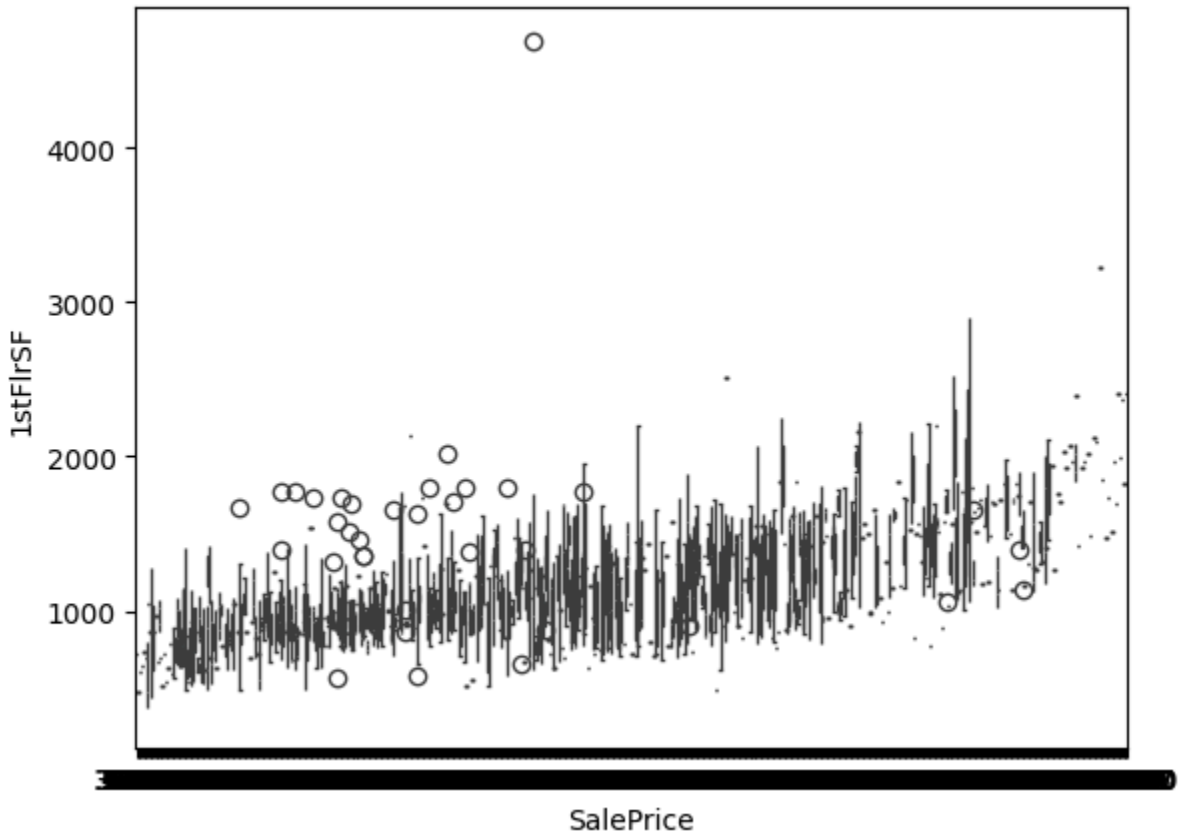
CentralAir



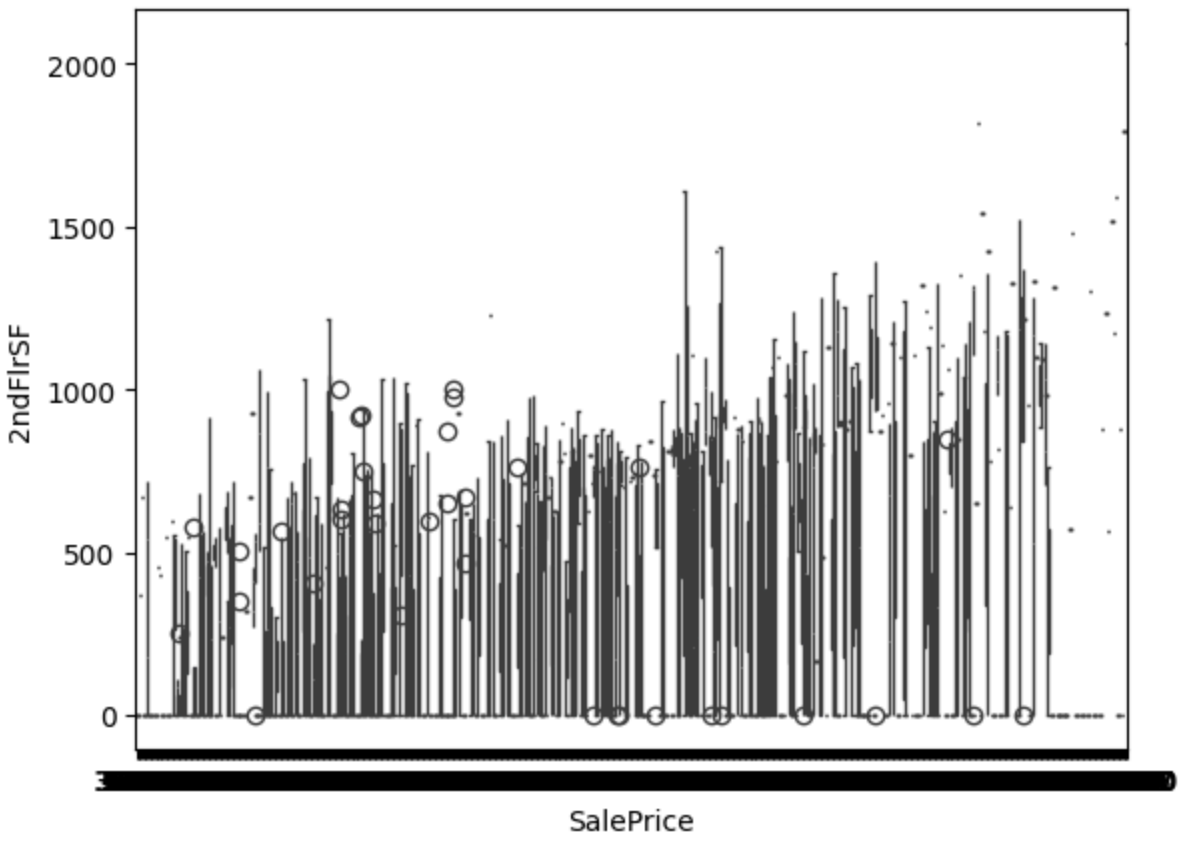
Electrical



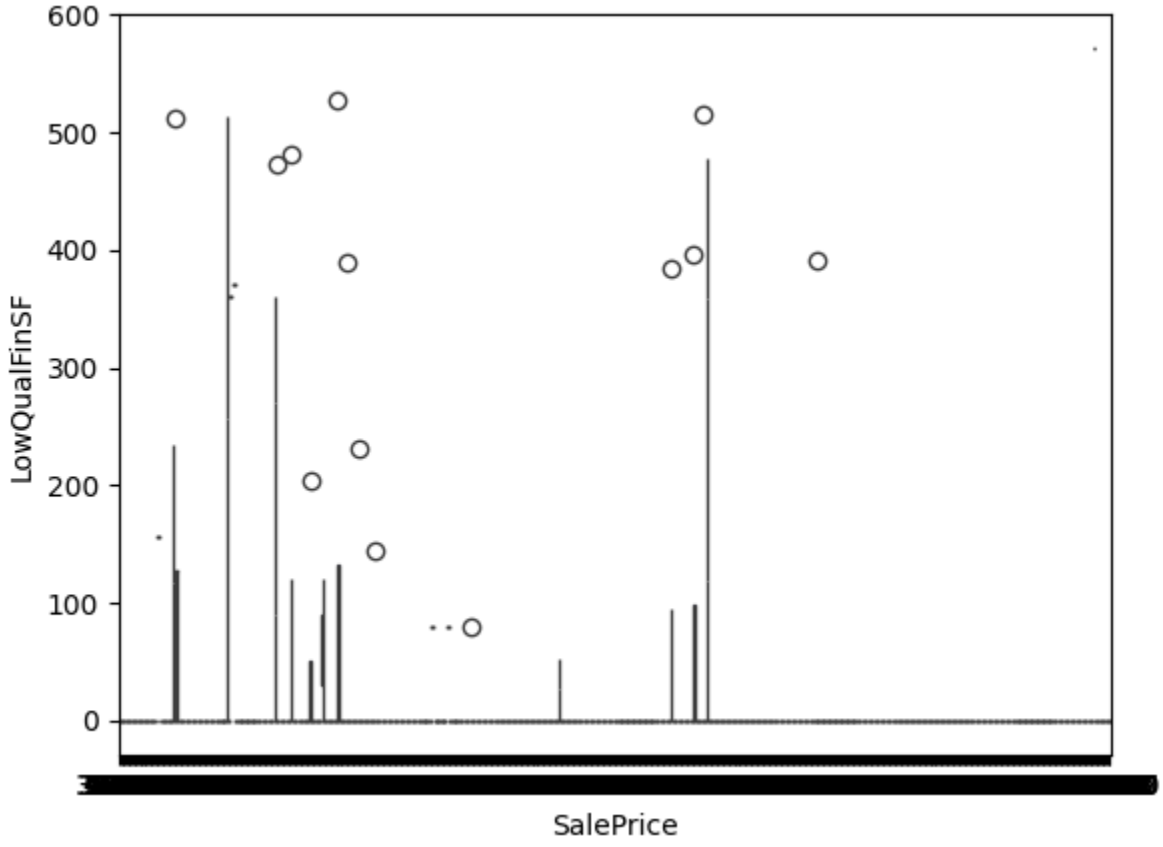
1stFlrSF



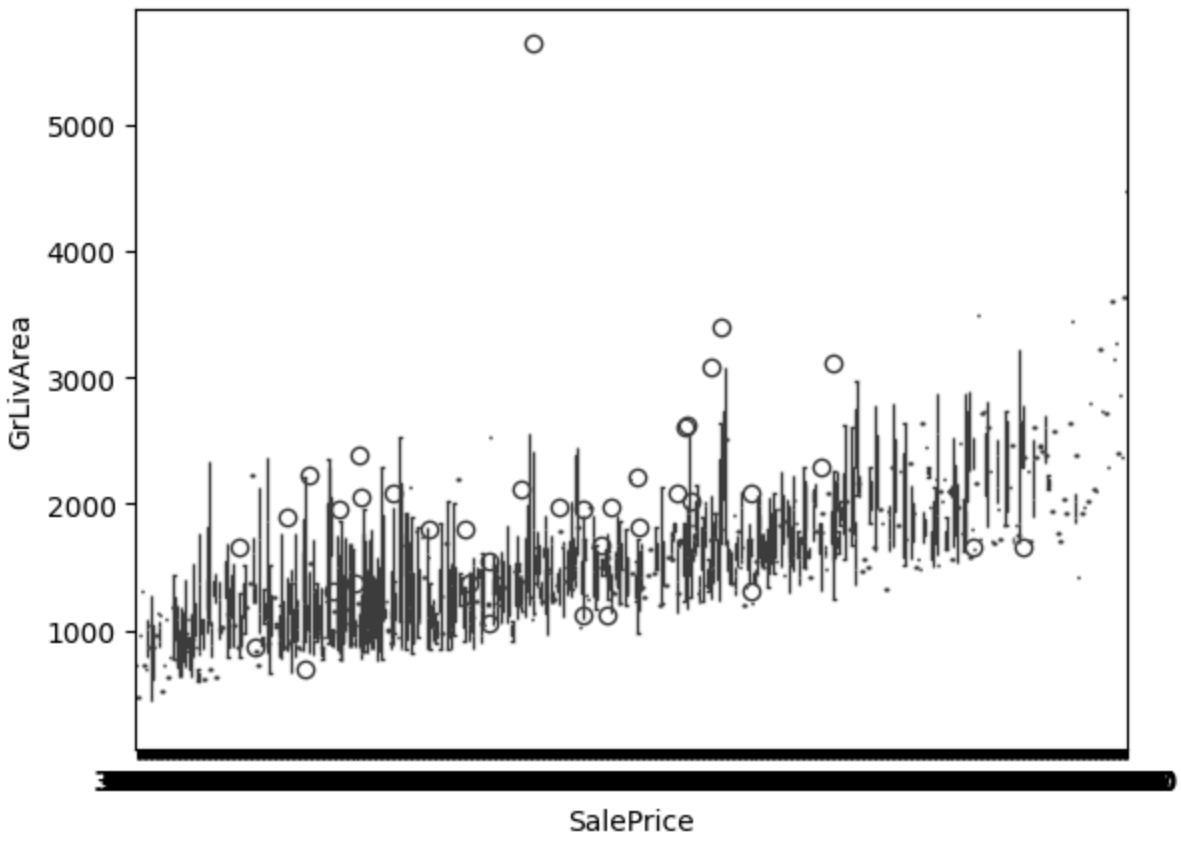
2ndFlrSF



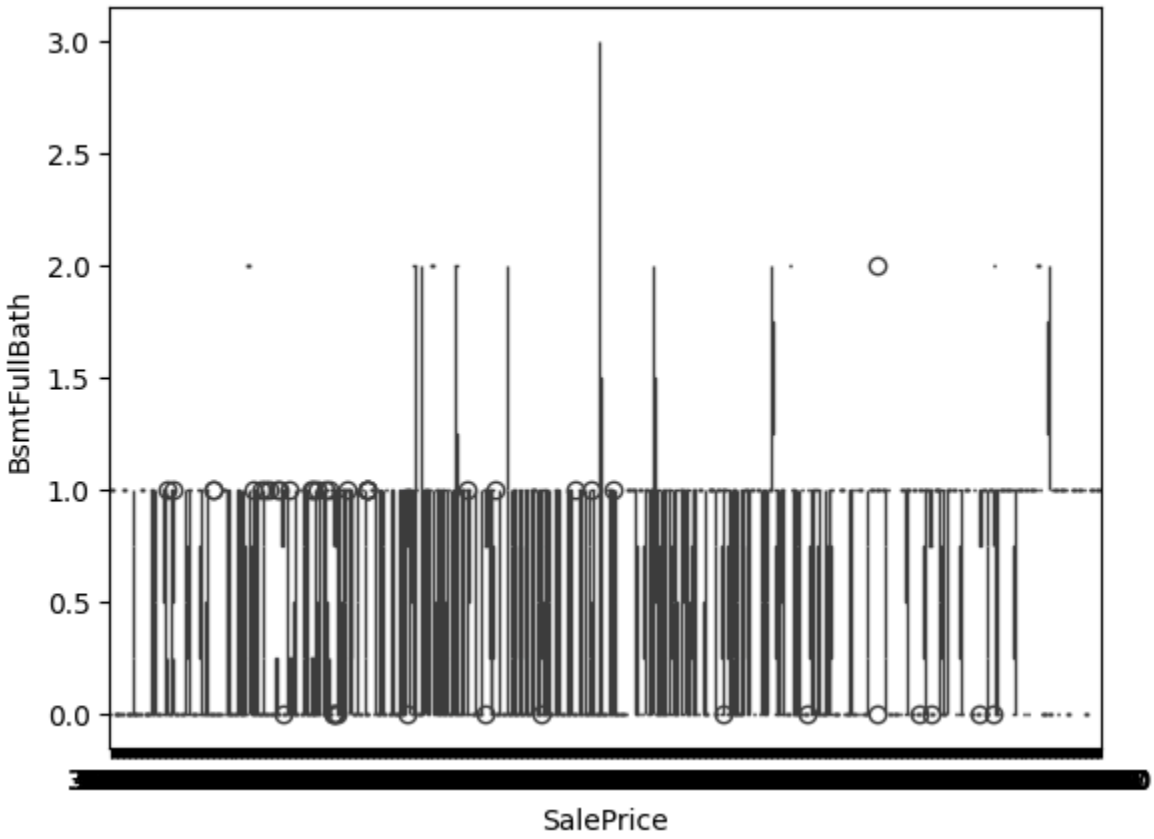
LowQualFinSF



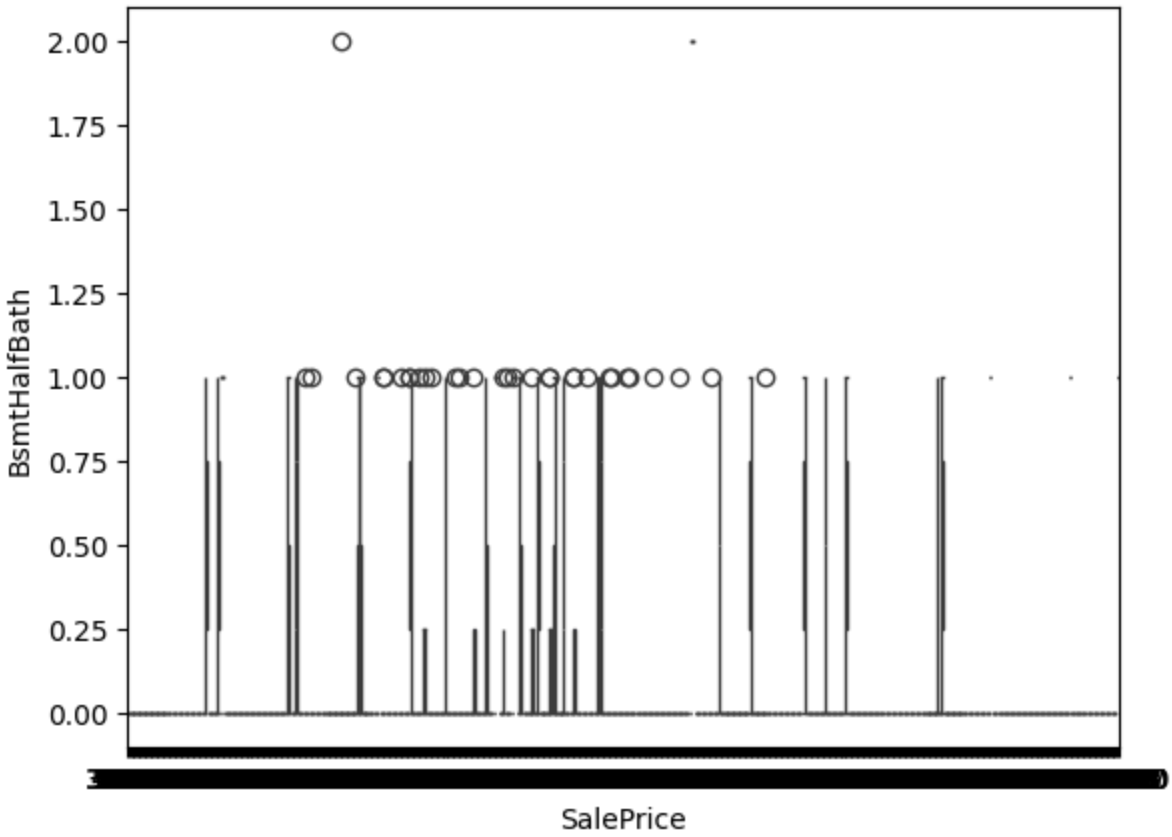
GrLivArea



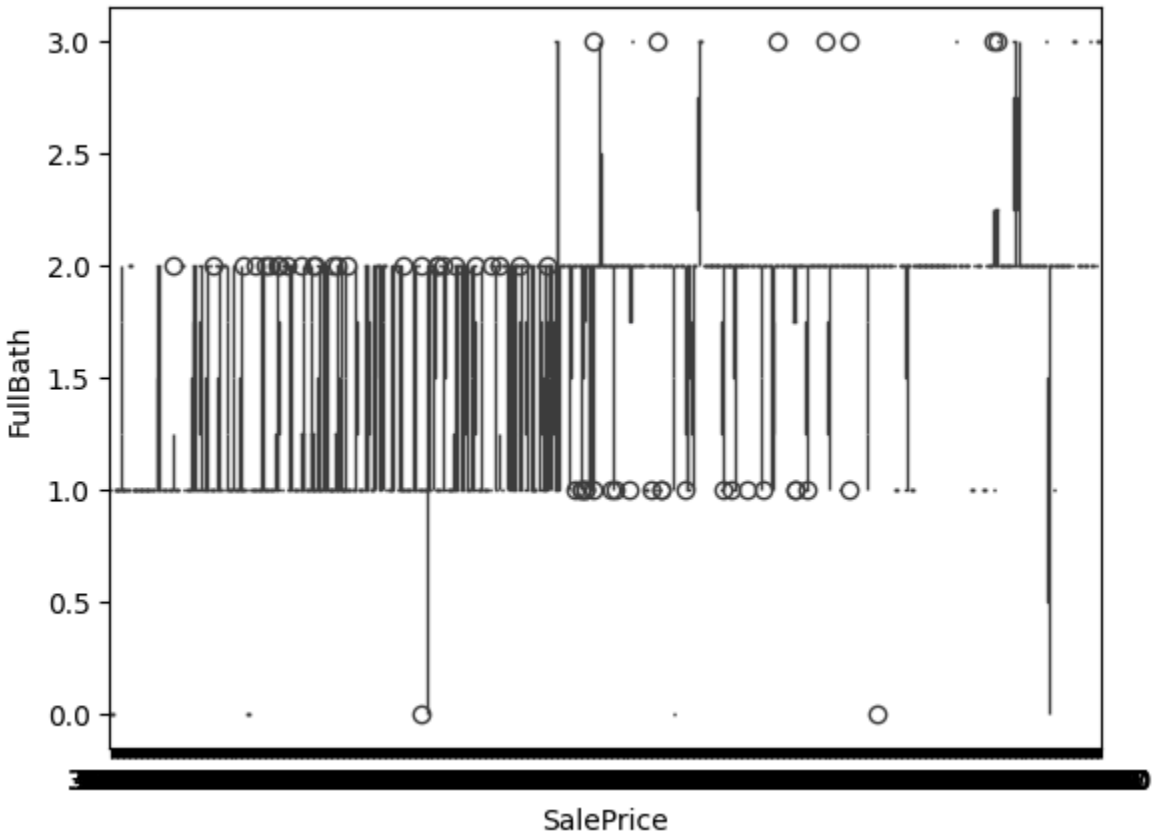
BsmtFullBath



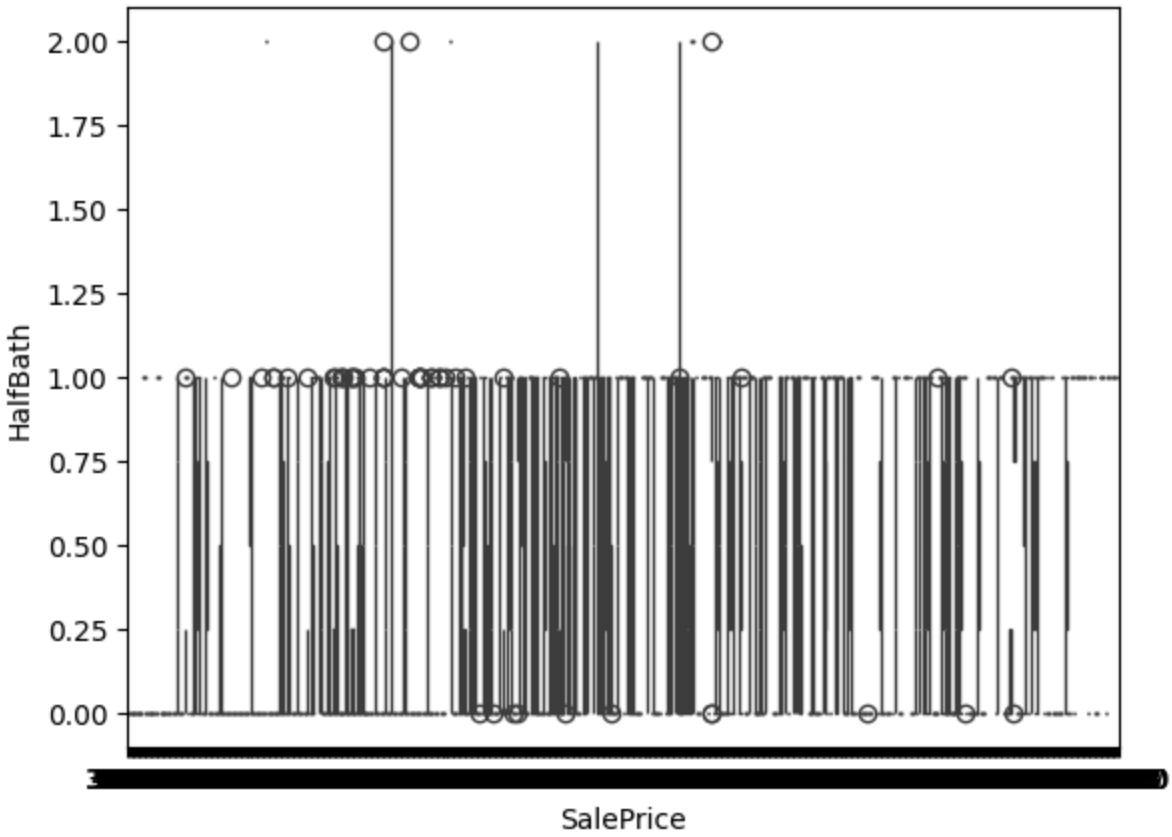
BsmtHalfBath



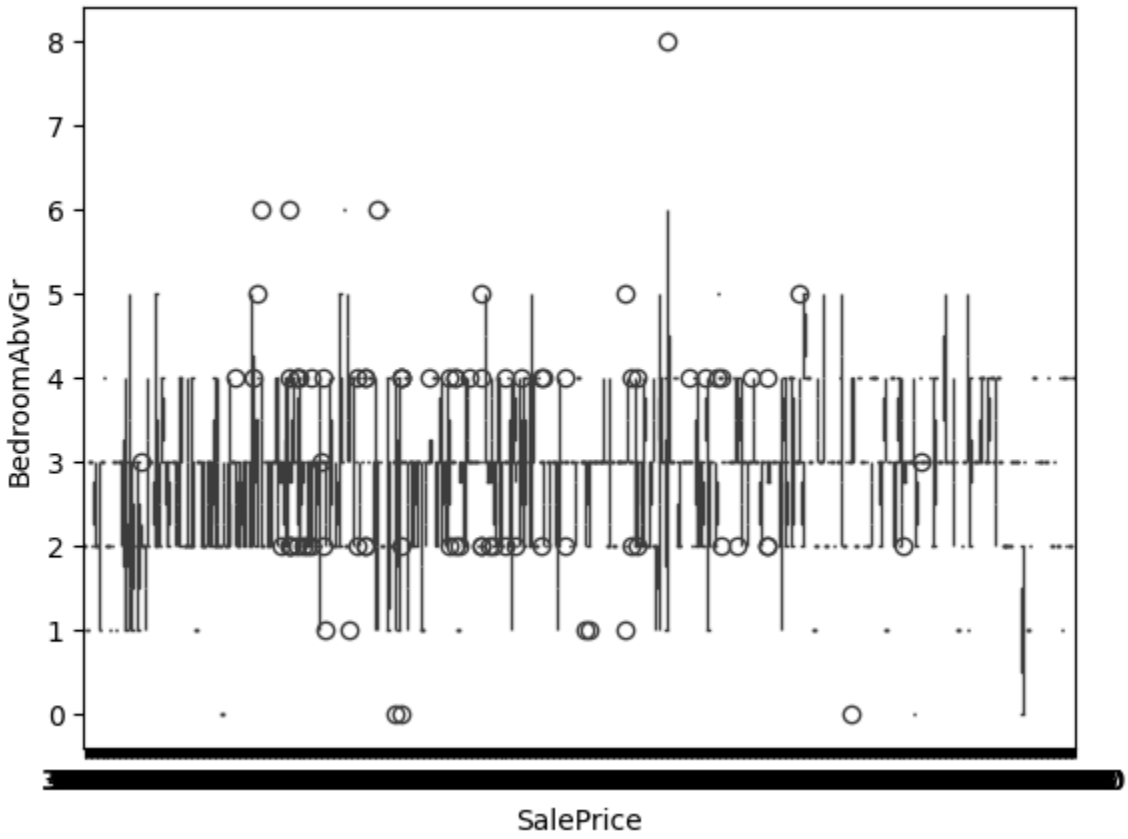
FullBath



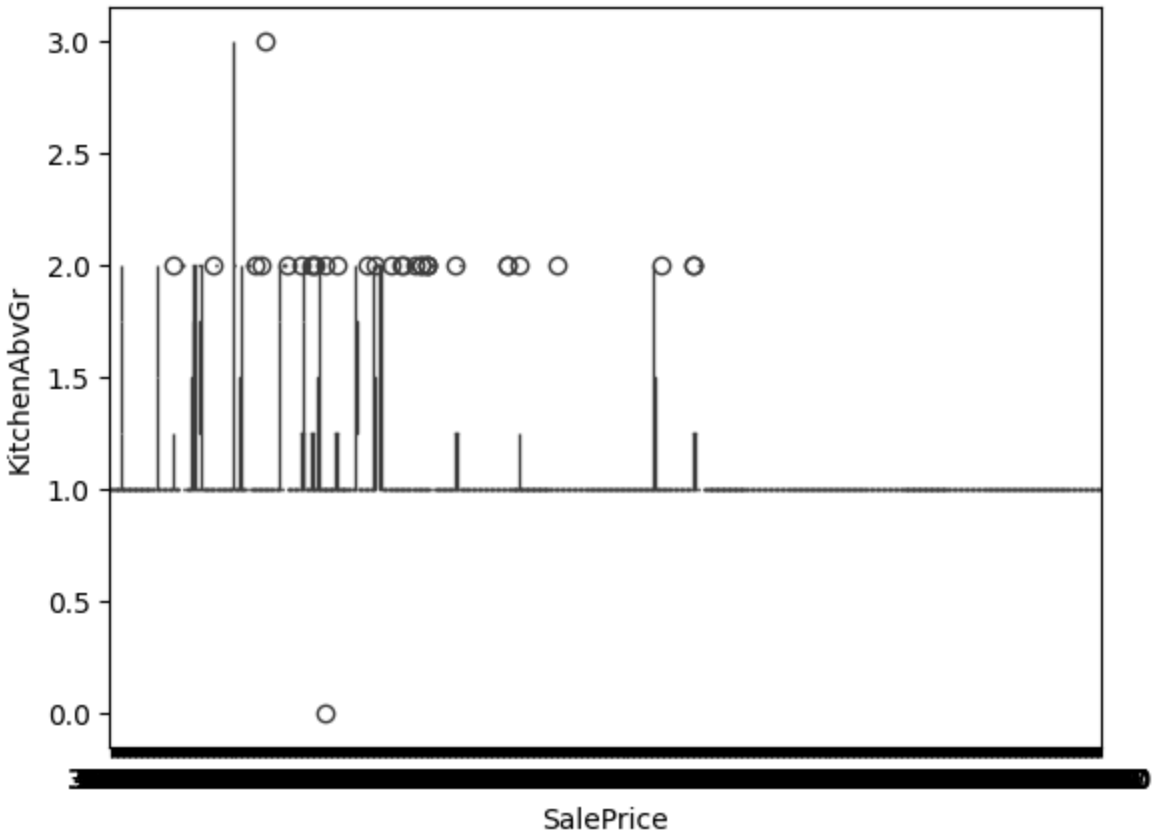
HalfBath



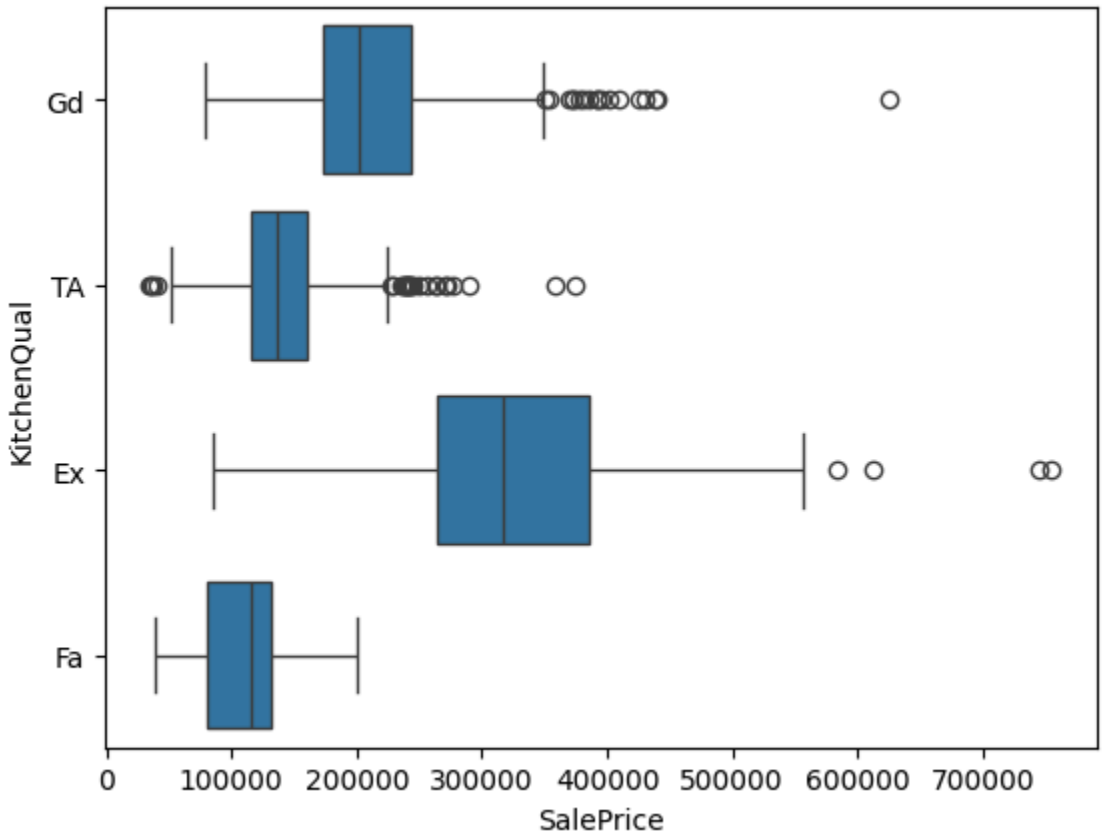
BedroomAbvGr



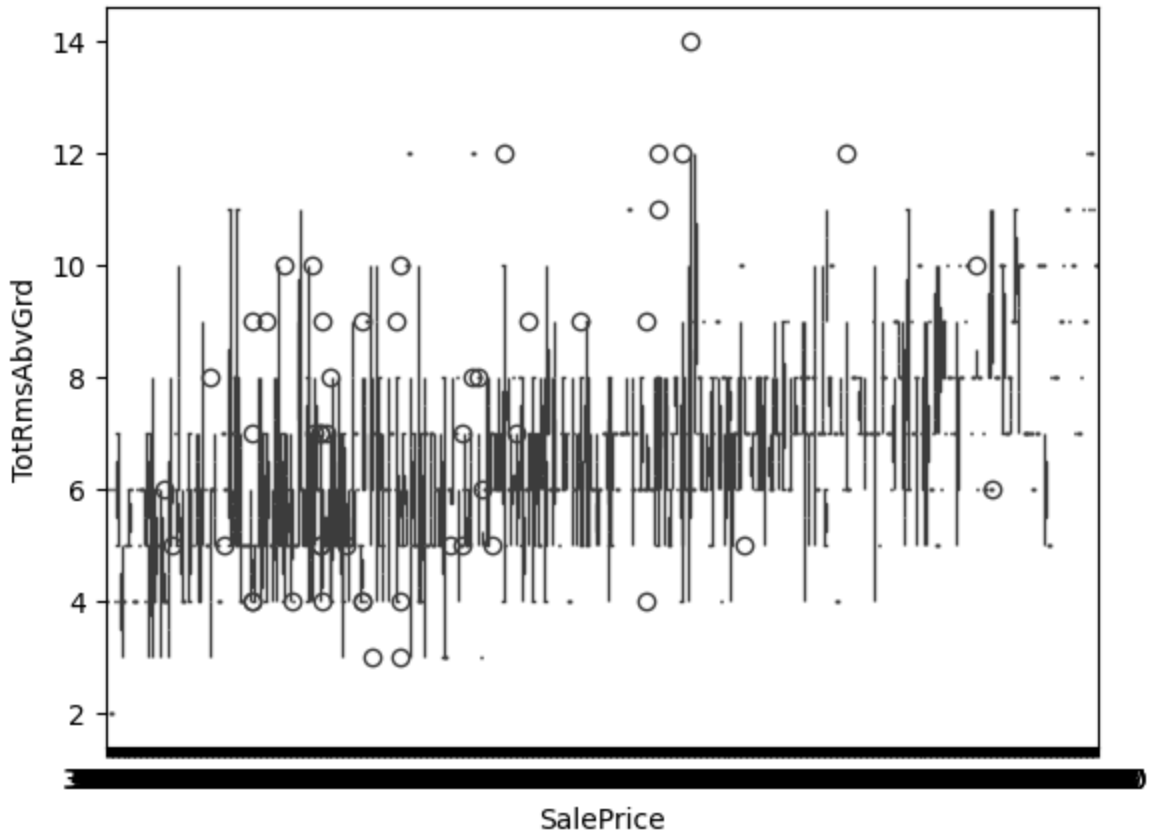
KitchenAbvGr



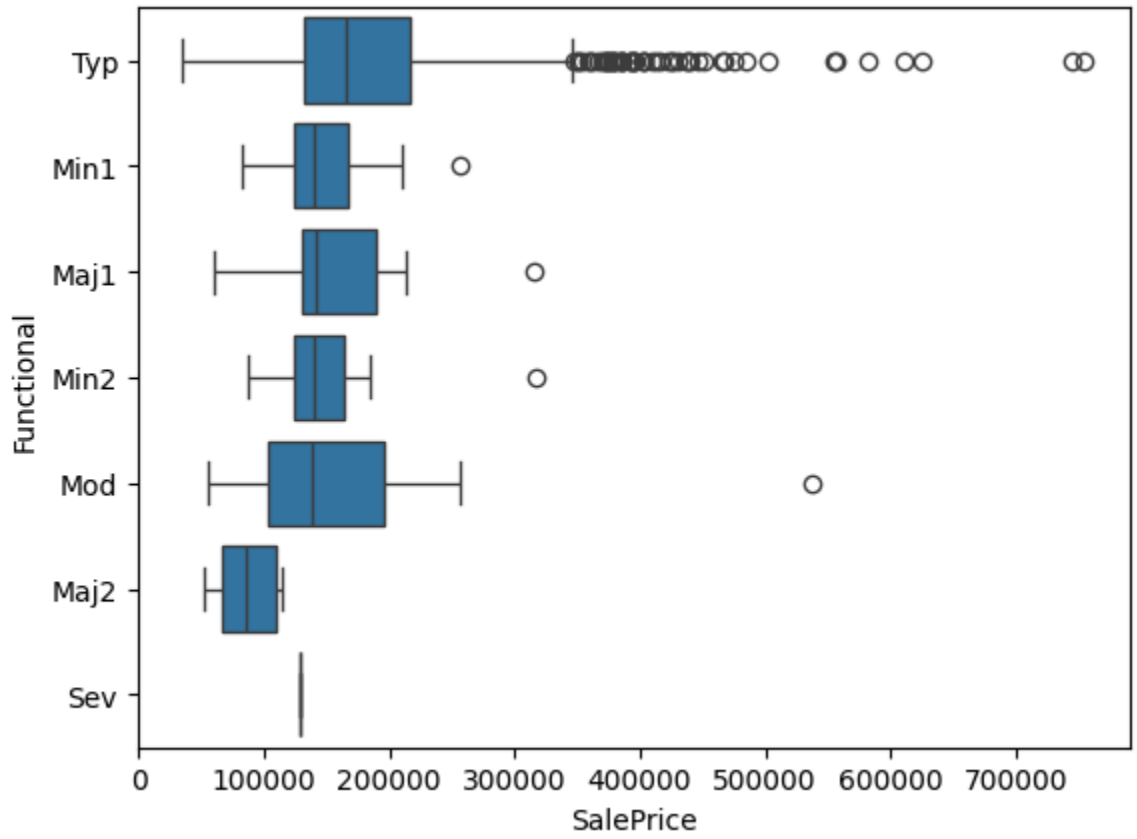
KitchenQual



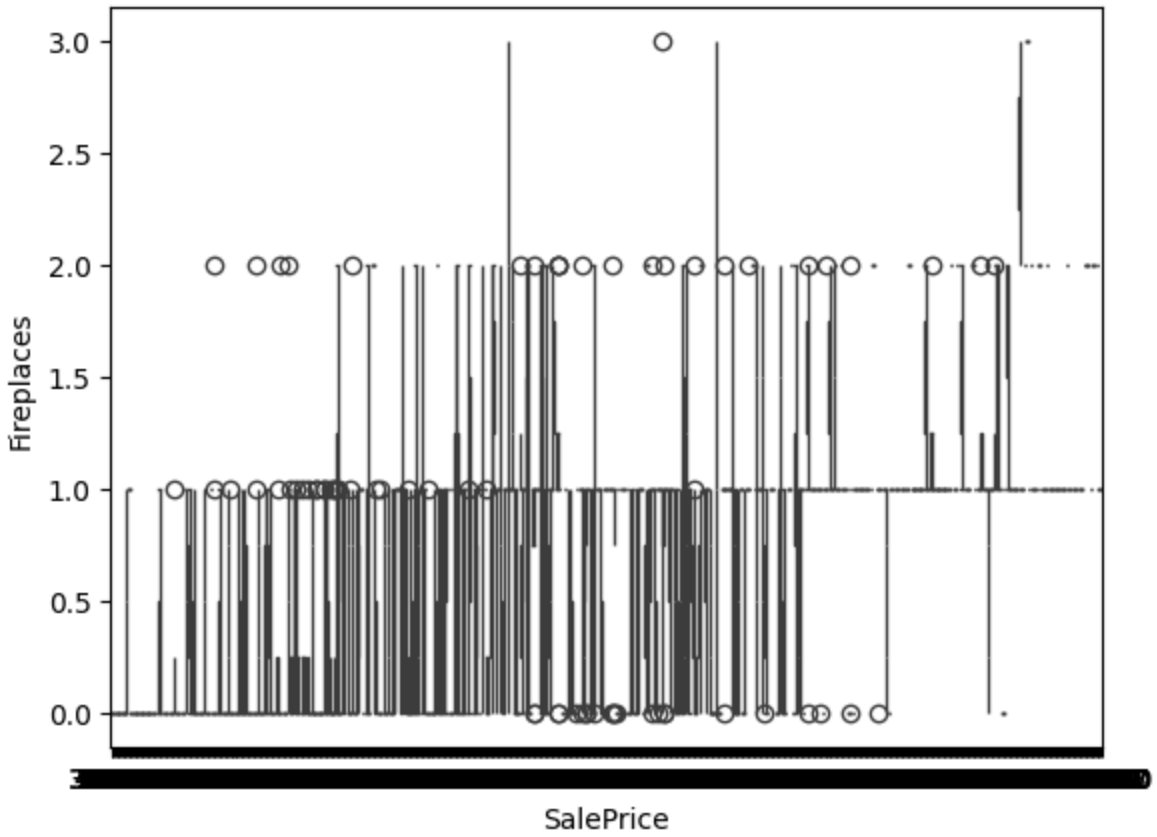
TotRmsAbvGrd



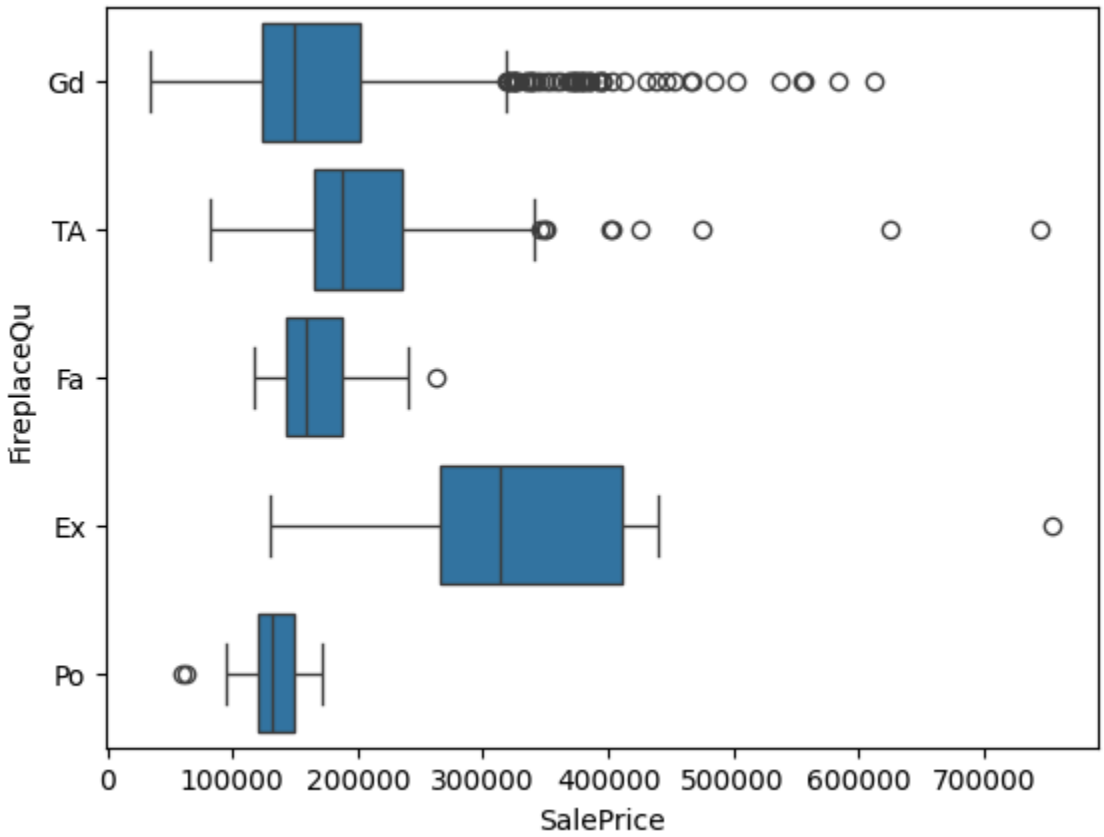
Functional



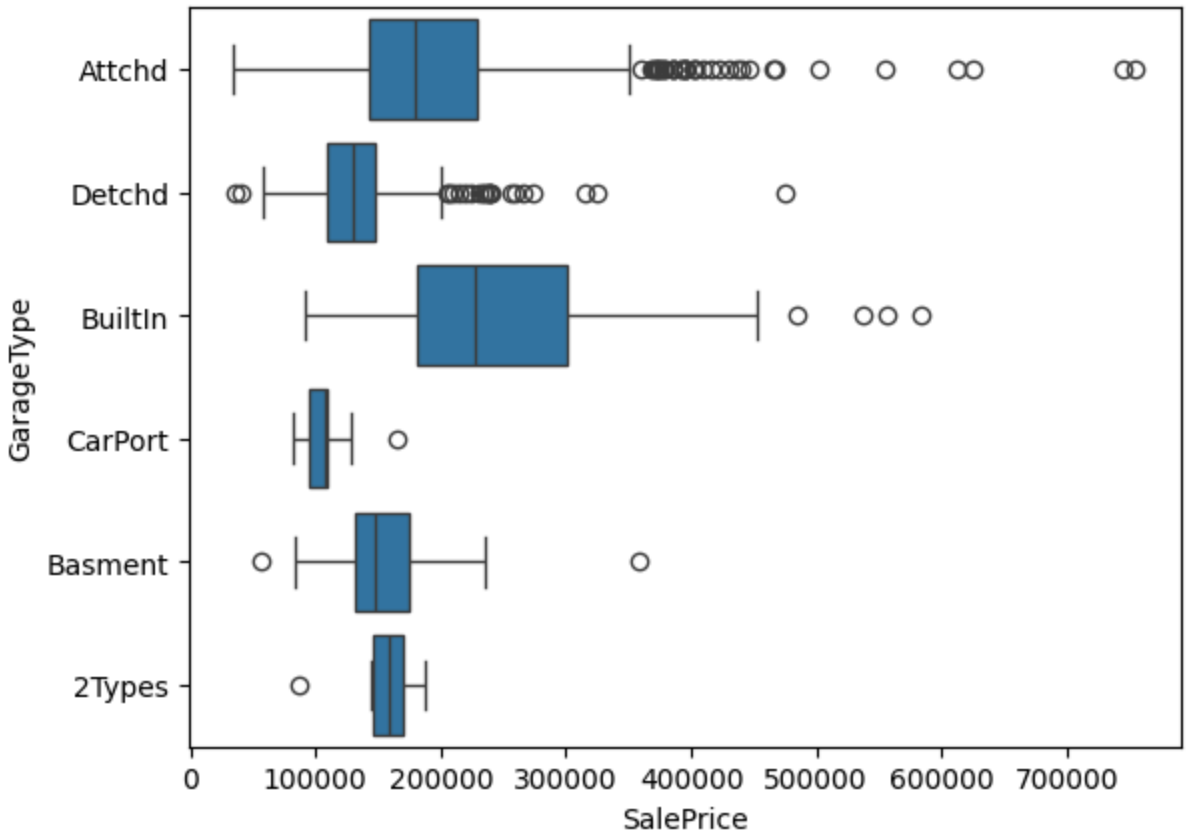
Fireplaces



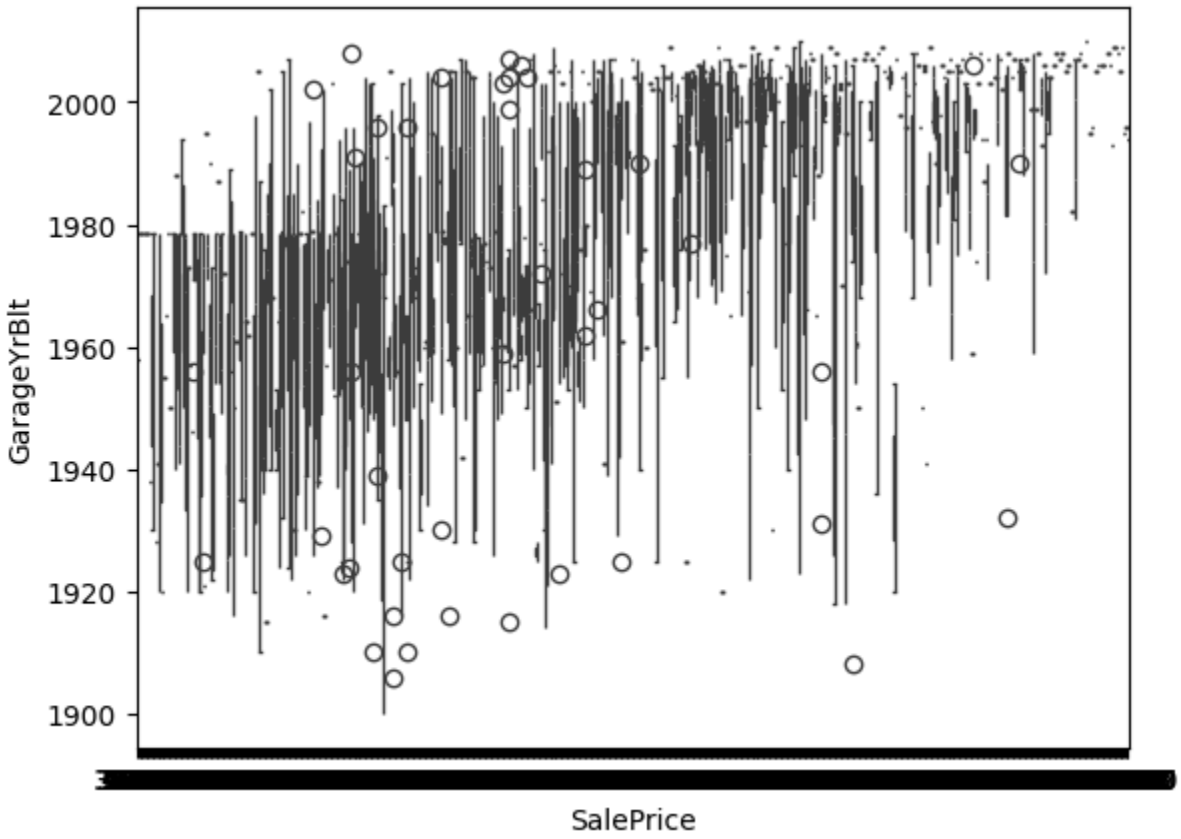
FireplaceQu



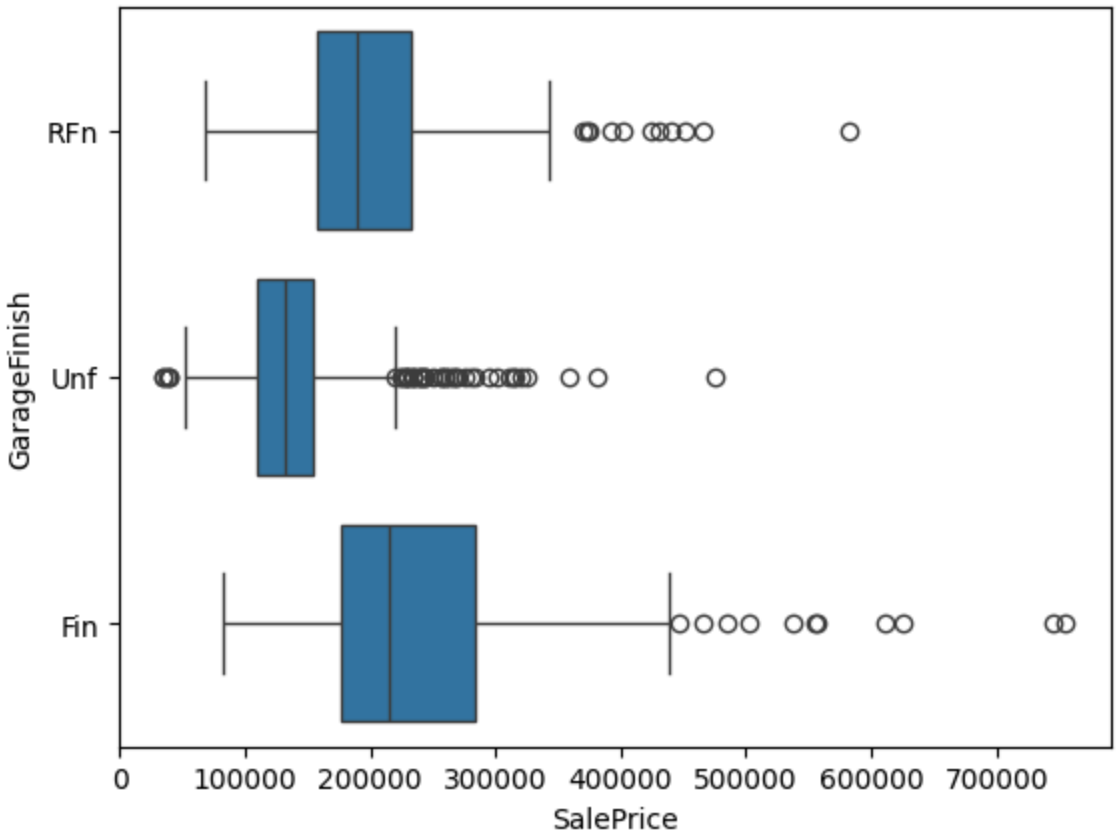
GarageType



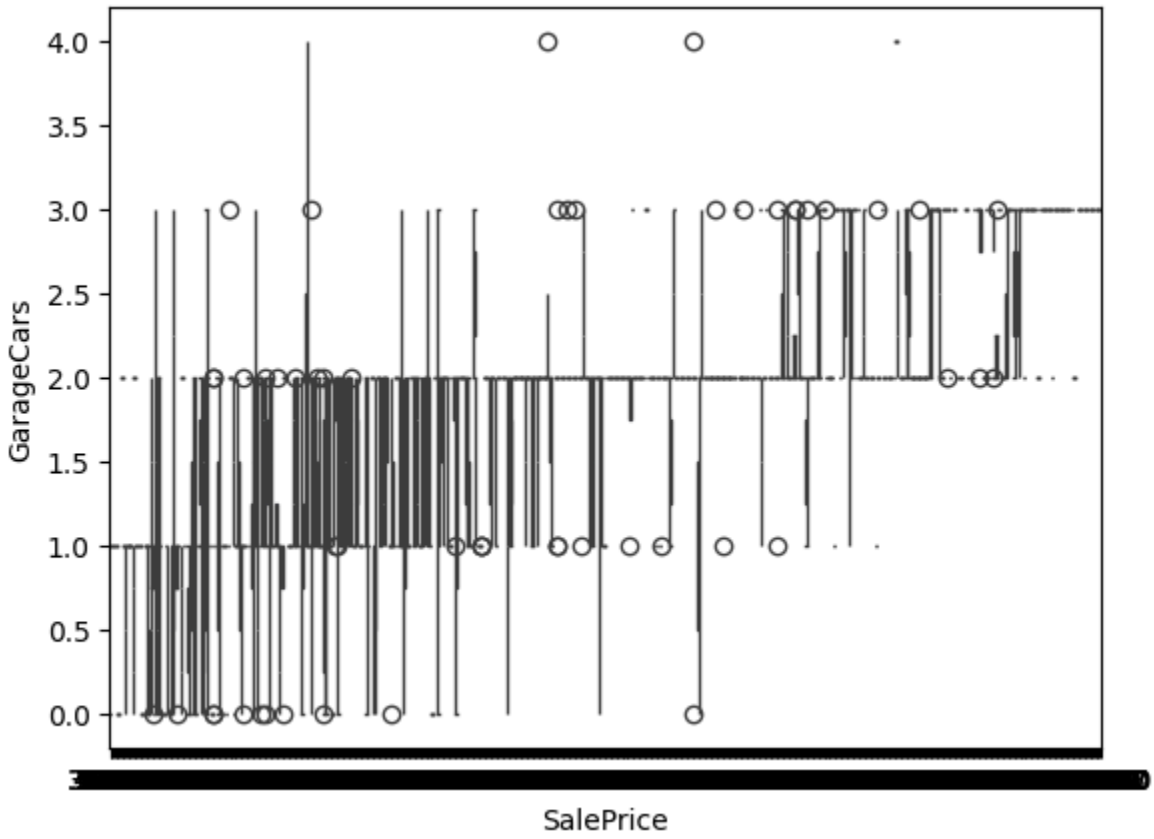
GarageYrBlt



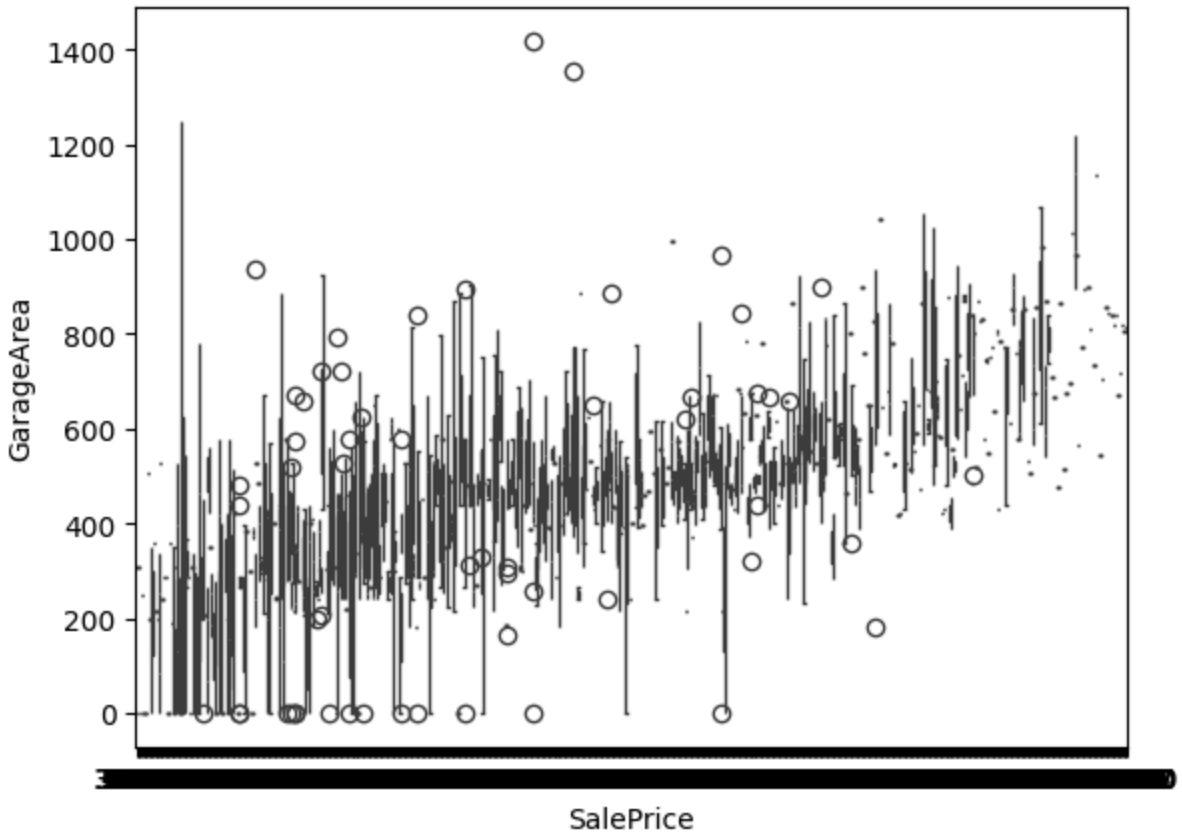
GarageFinish



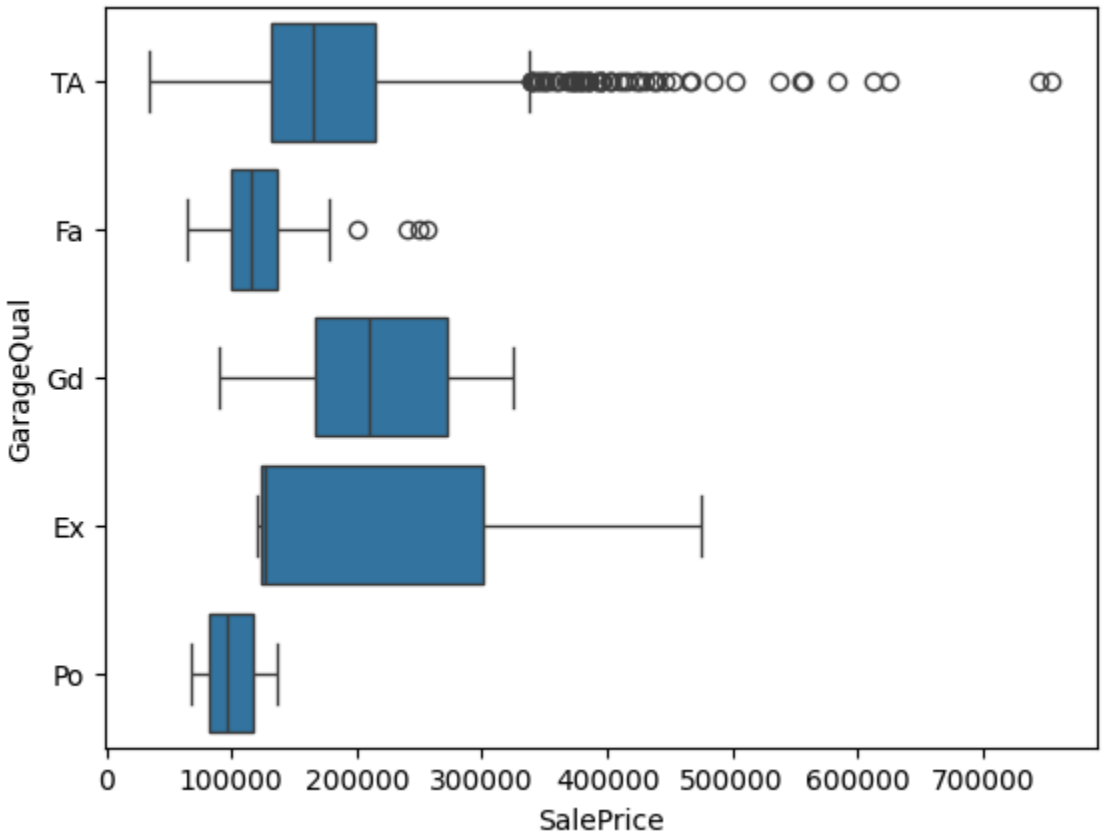
GarageCars



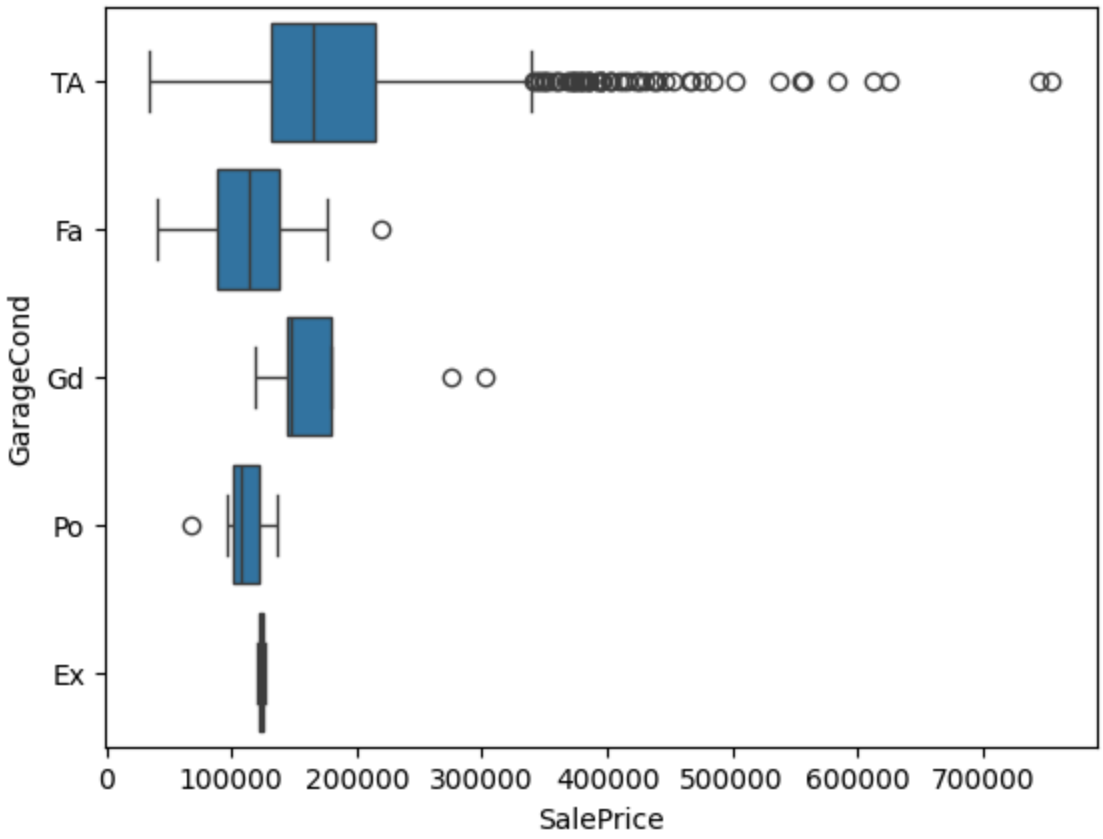
GarageArea



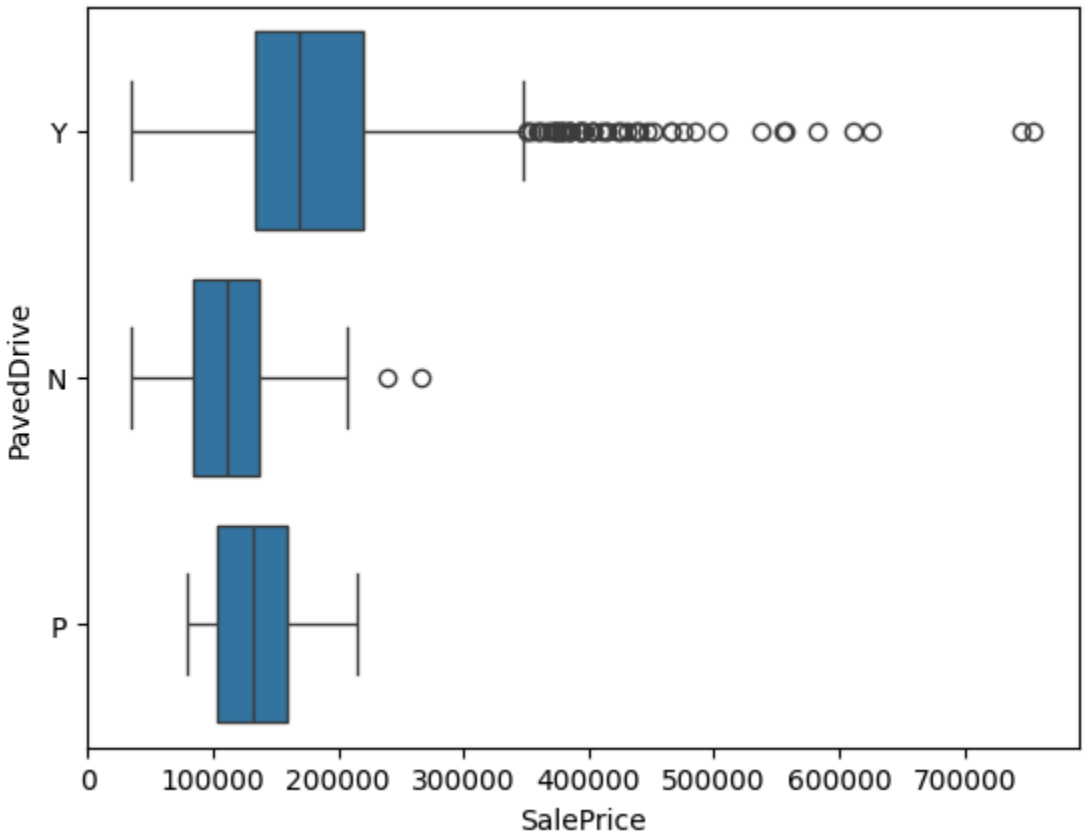
GarageQual



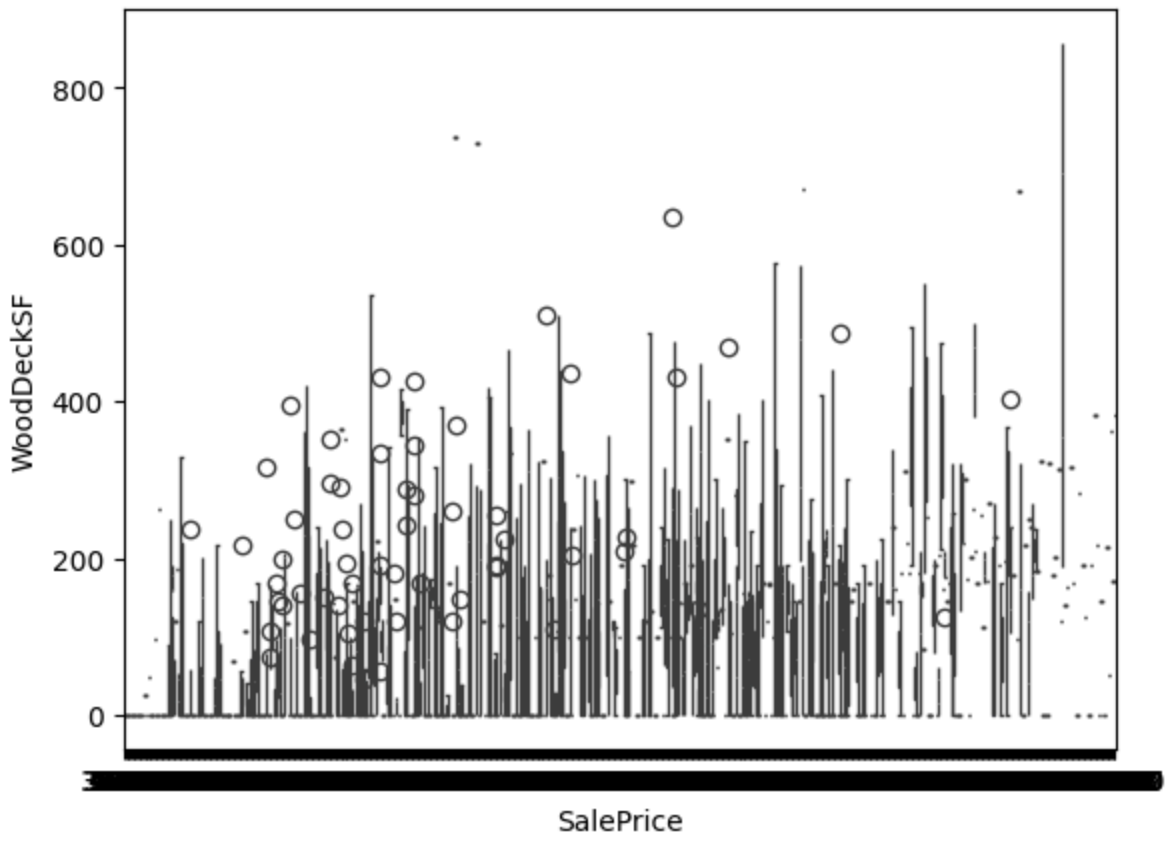
GarageCond



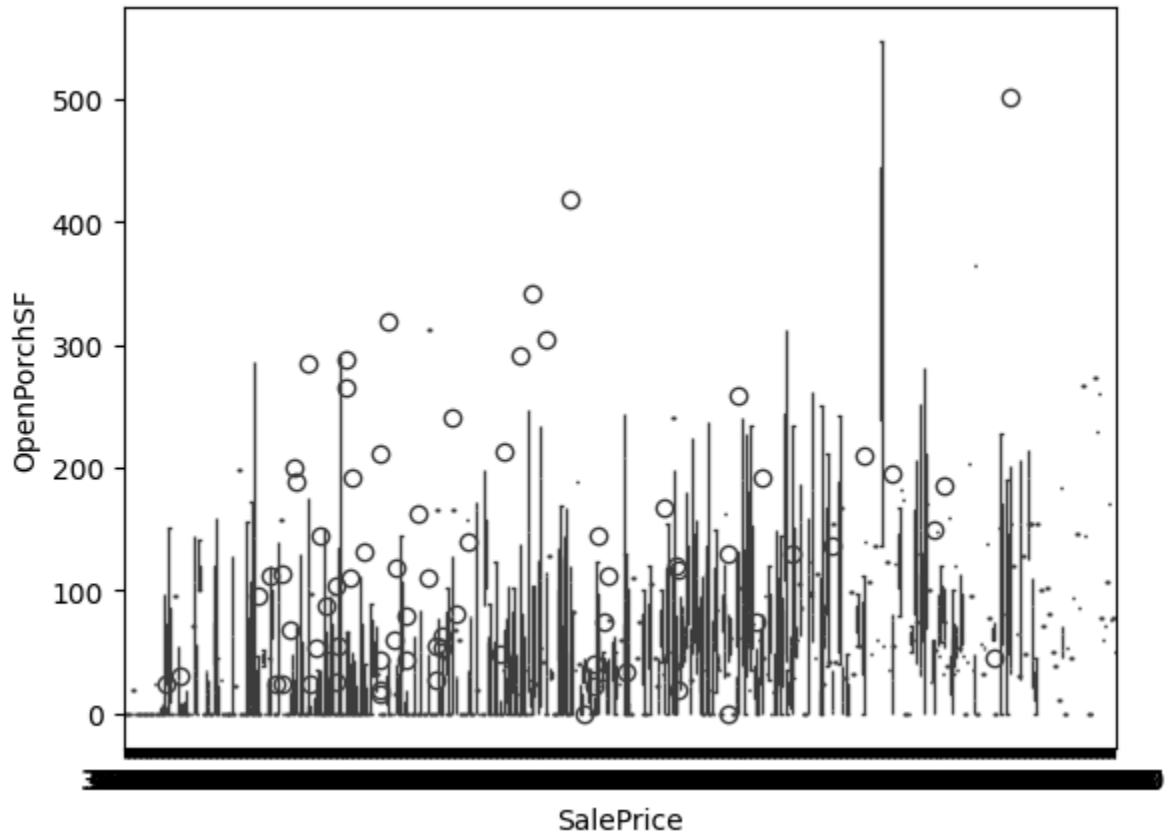
PavedDrive



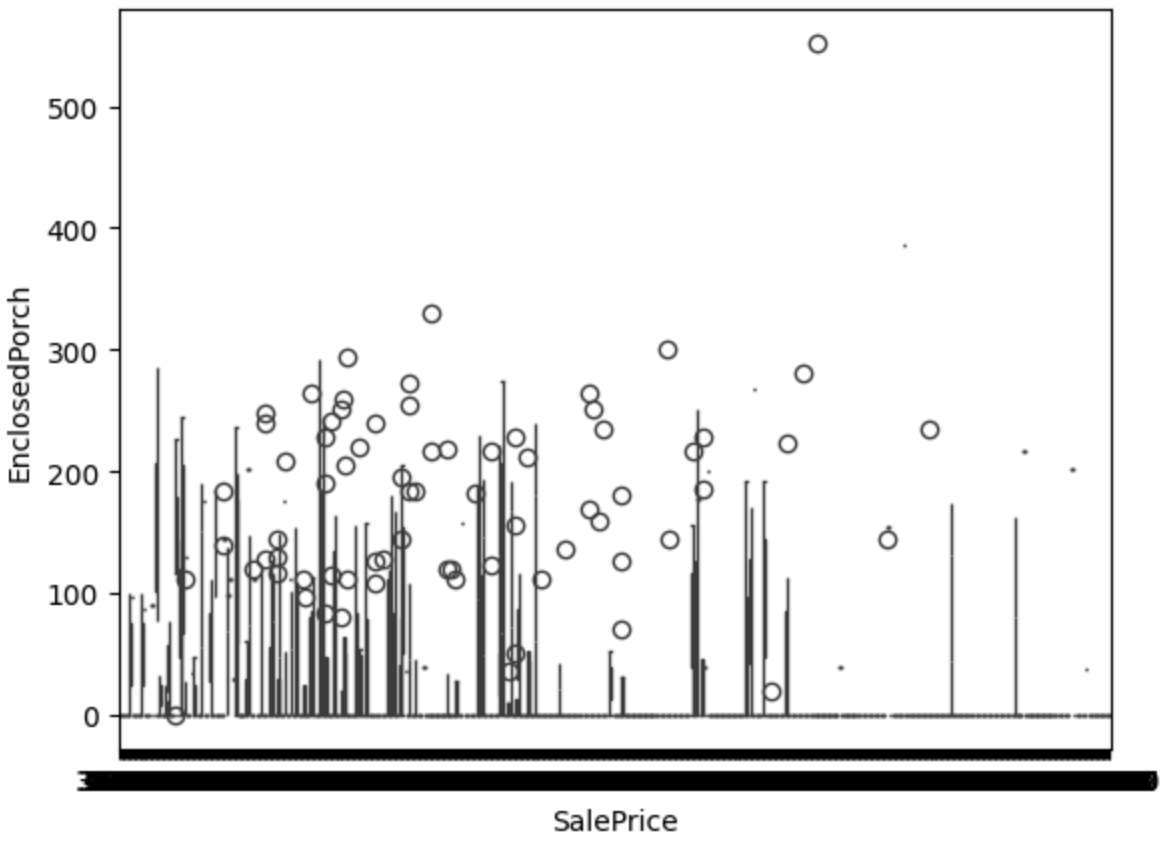
WoodDeckSF



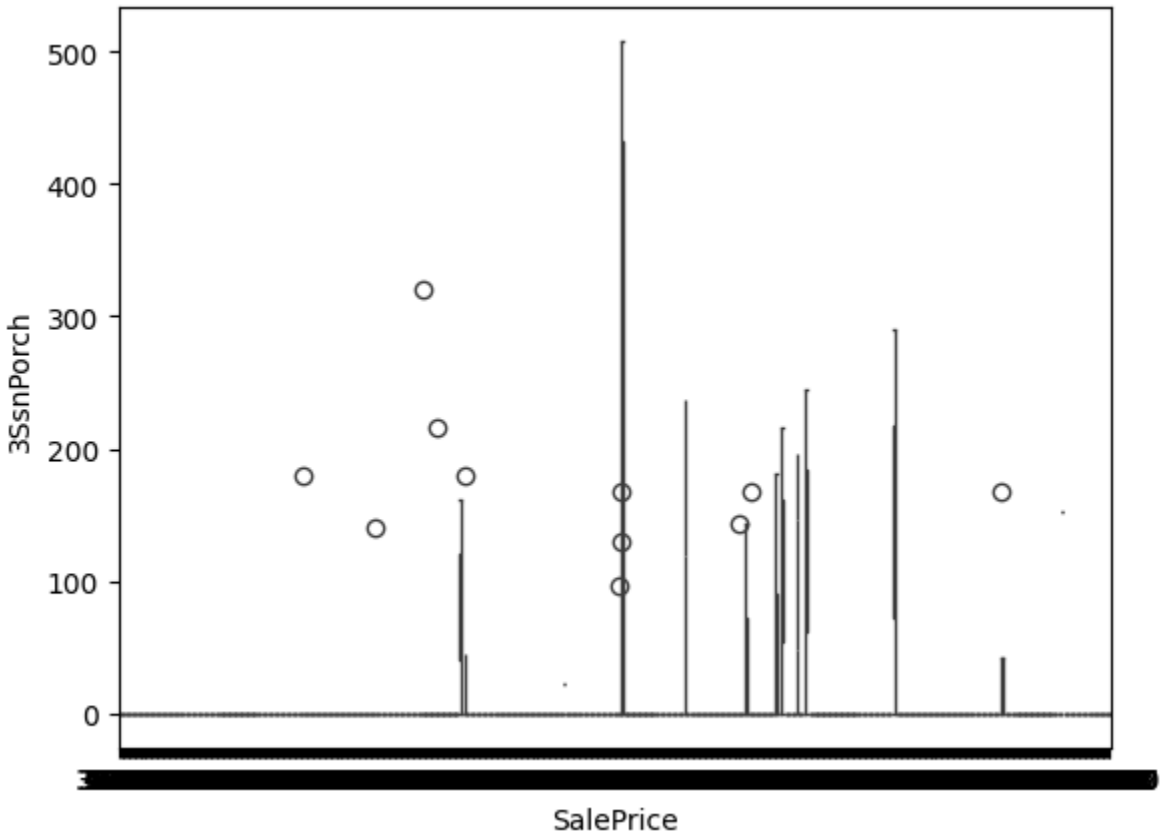
OpenPorchSF



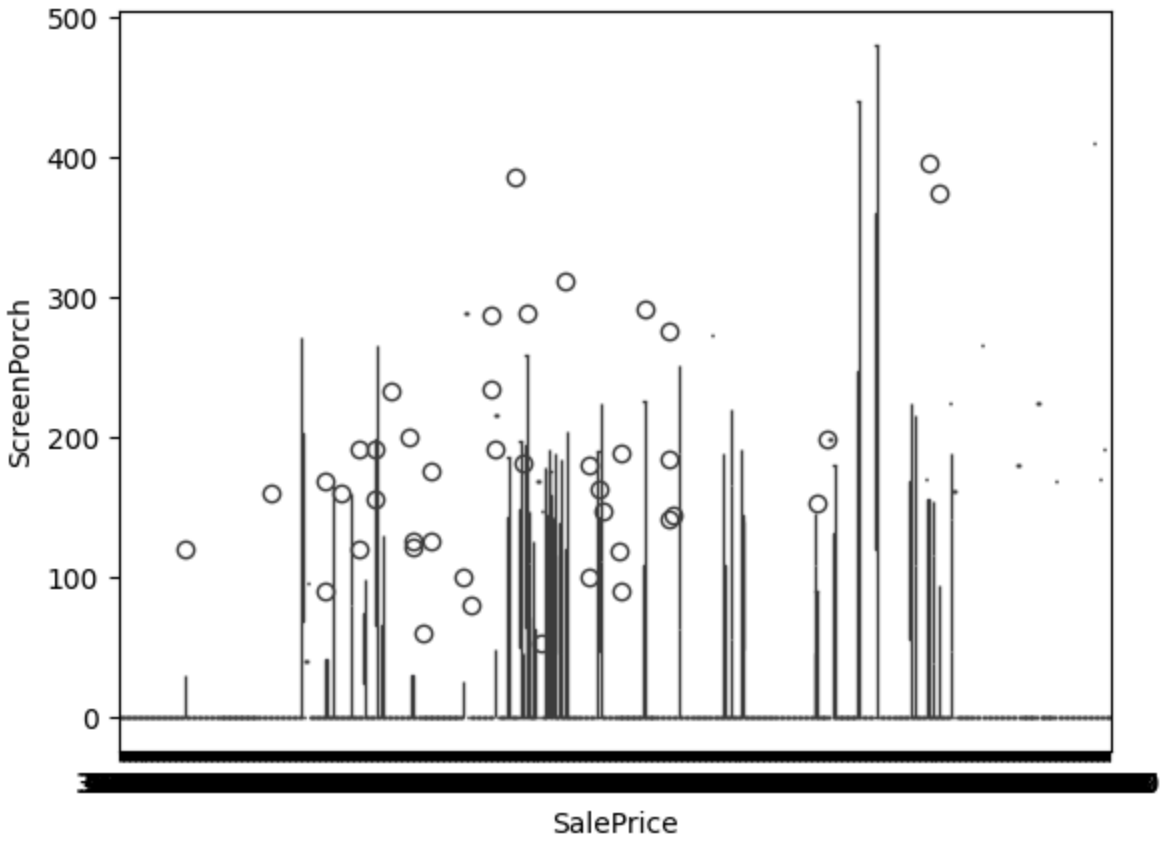
EnclosedPorch



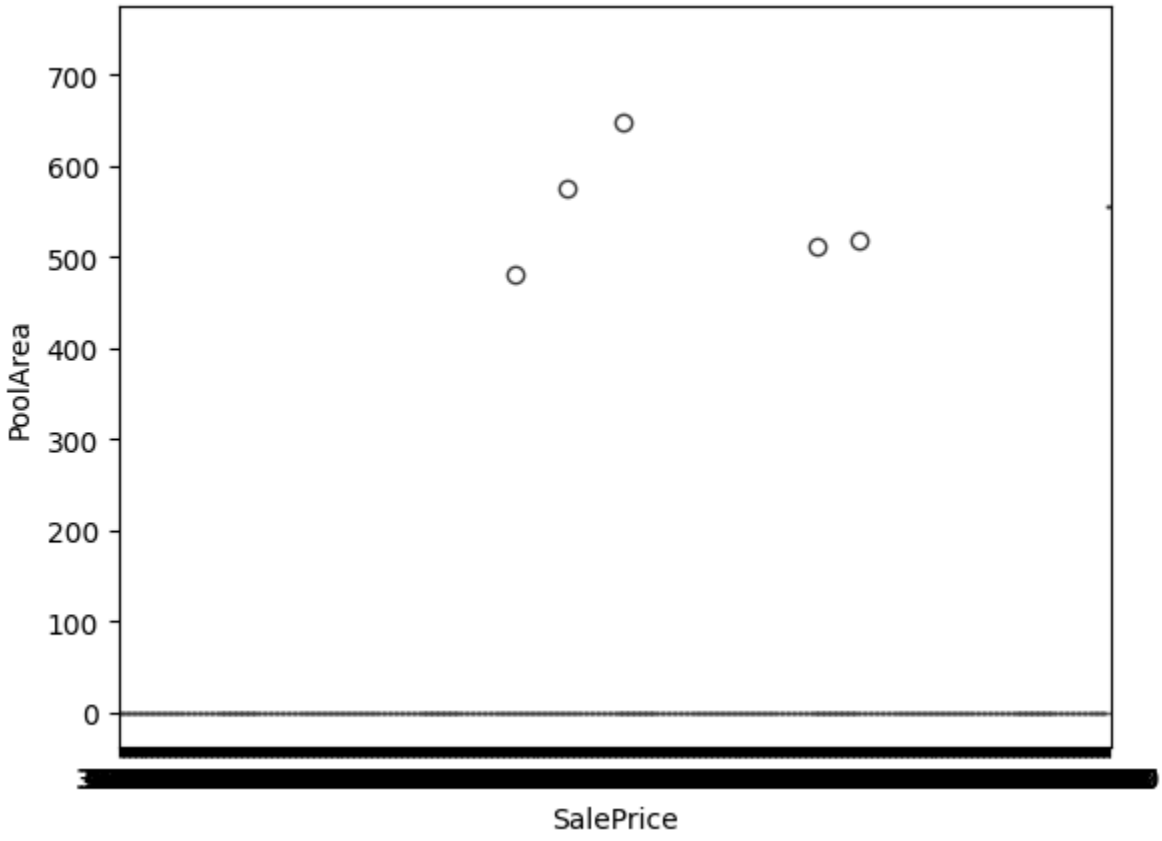
3SsnPorch



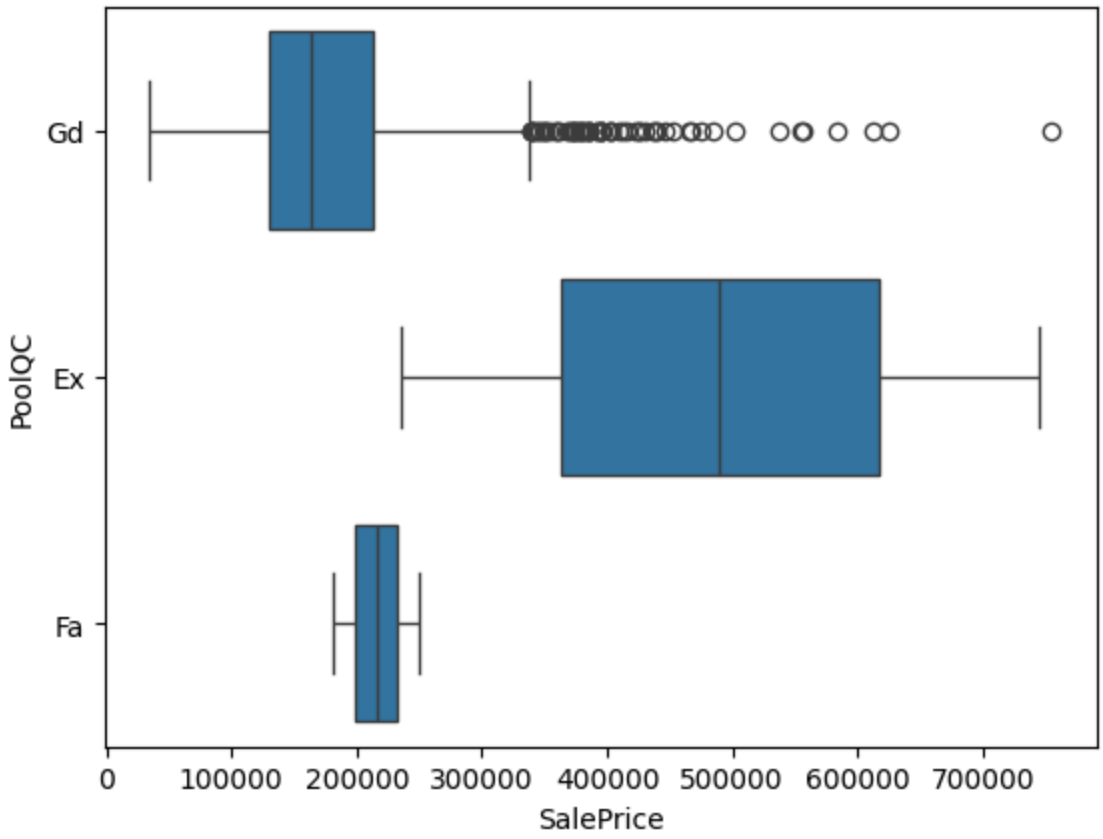
ScreenPorch



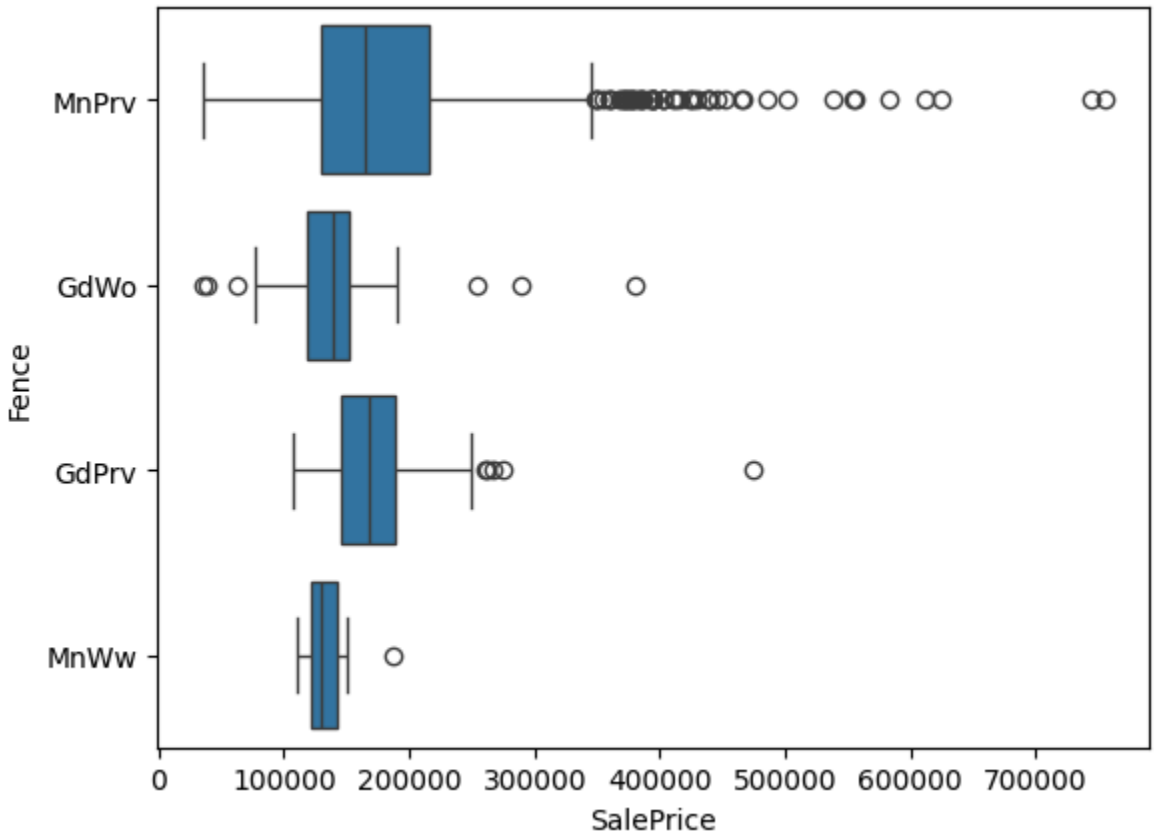
PoolArea



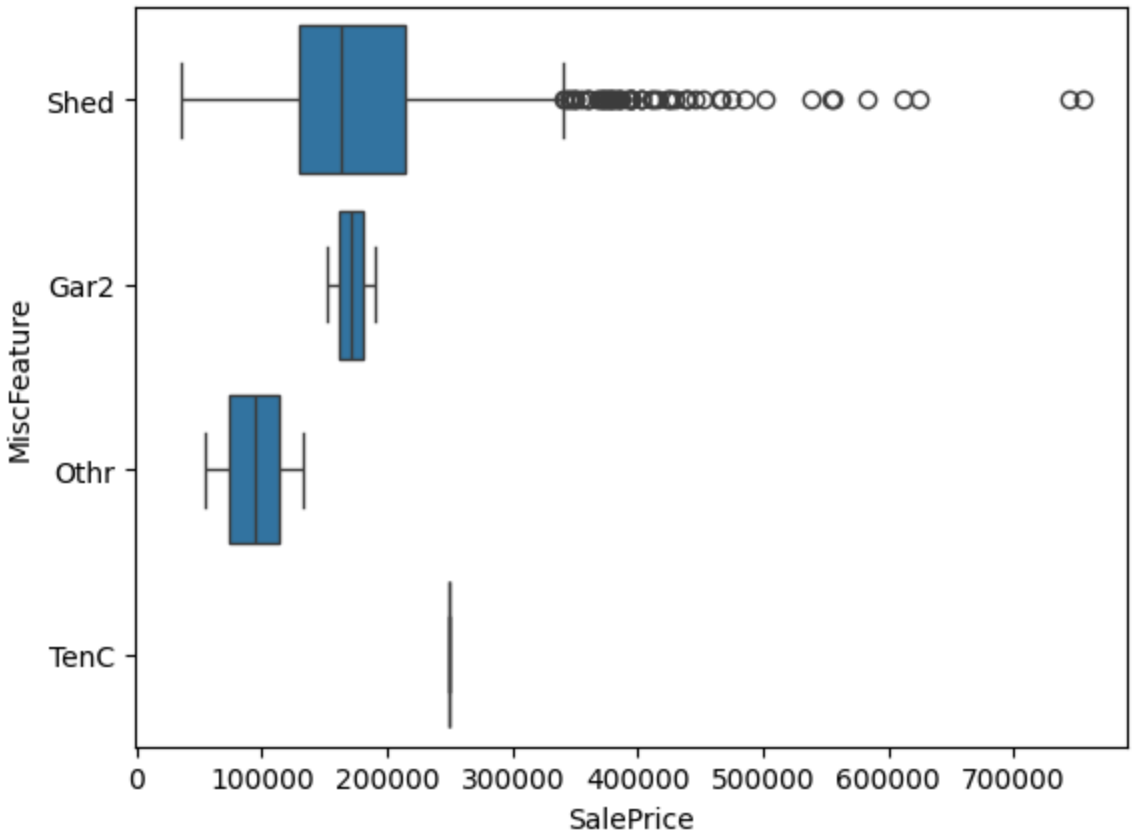
PoolQC



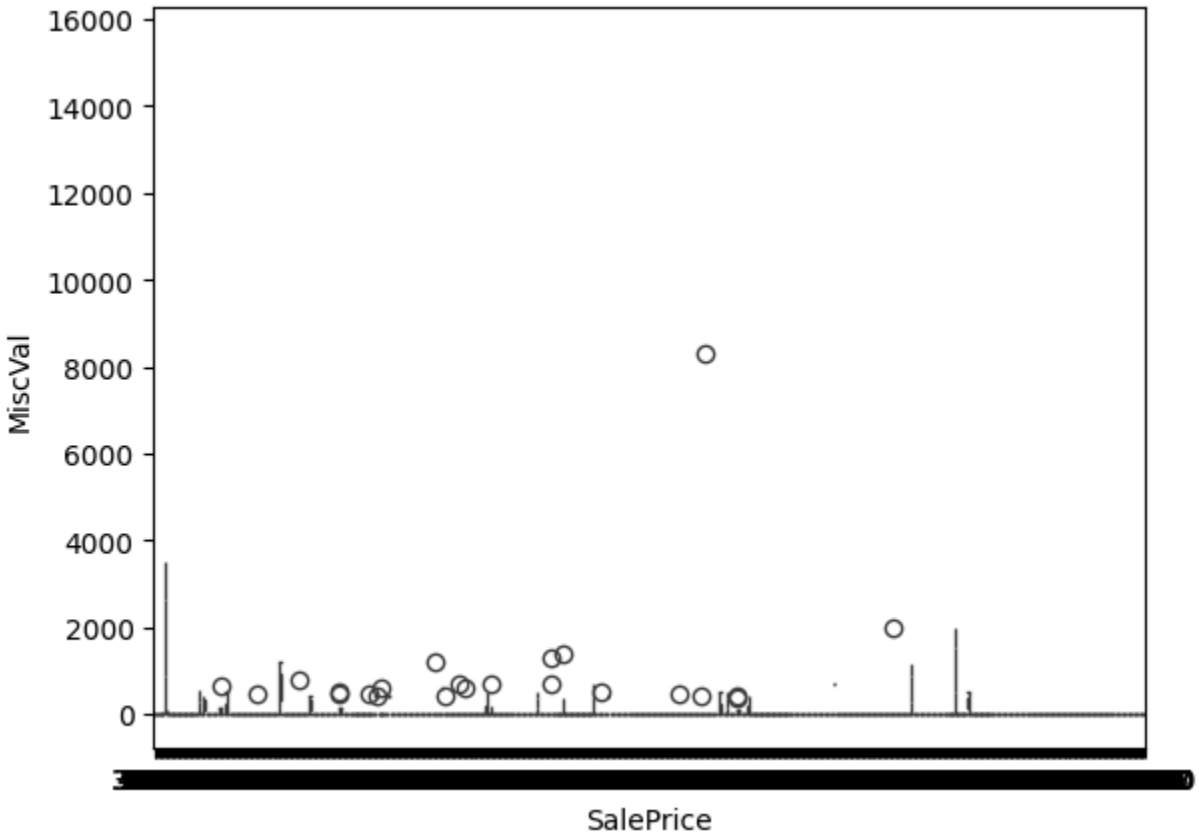
Fence



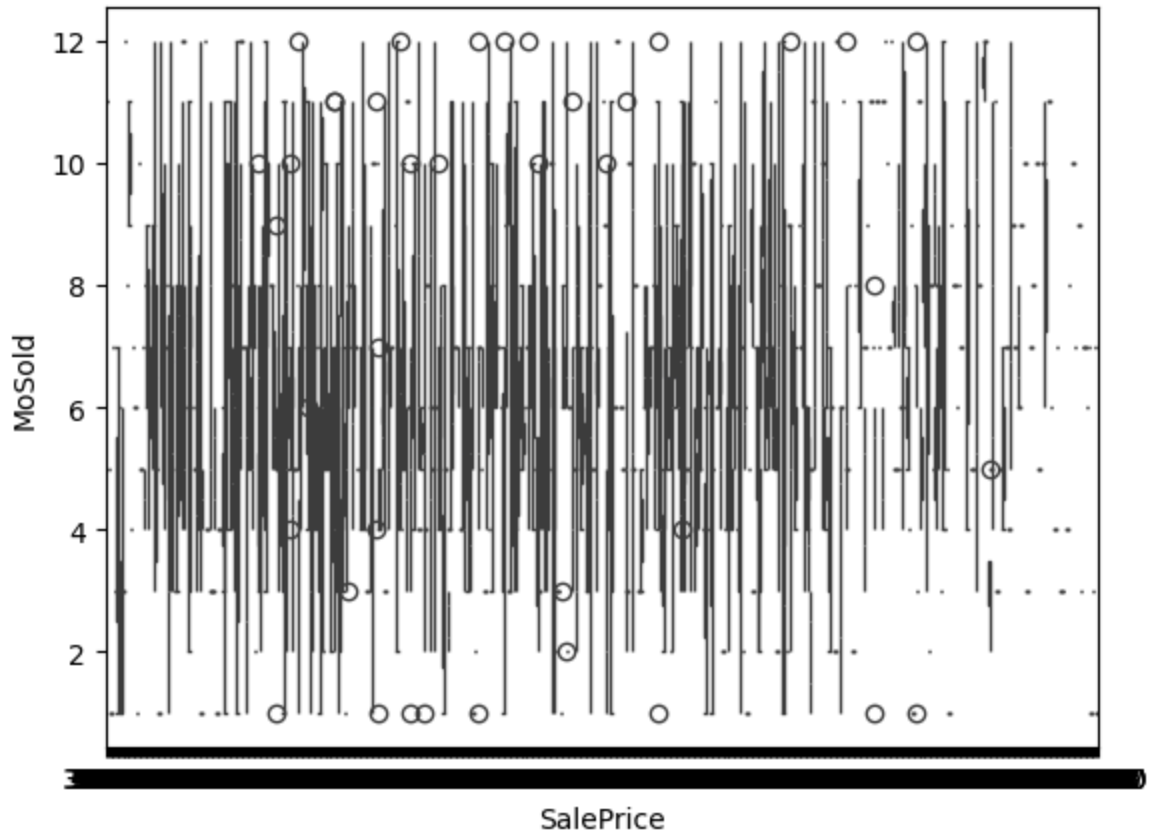
MiscFeature



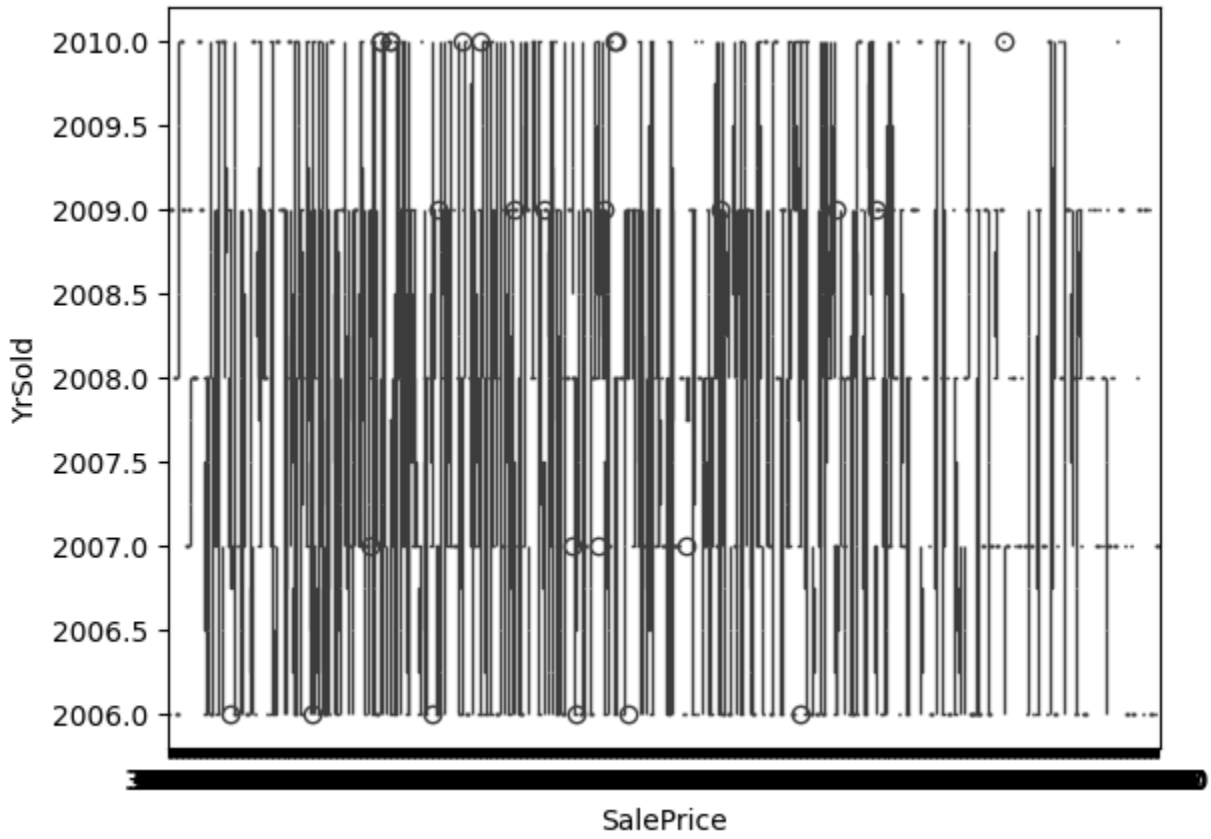
MiscVal



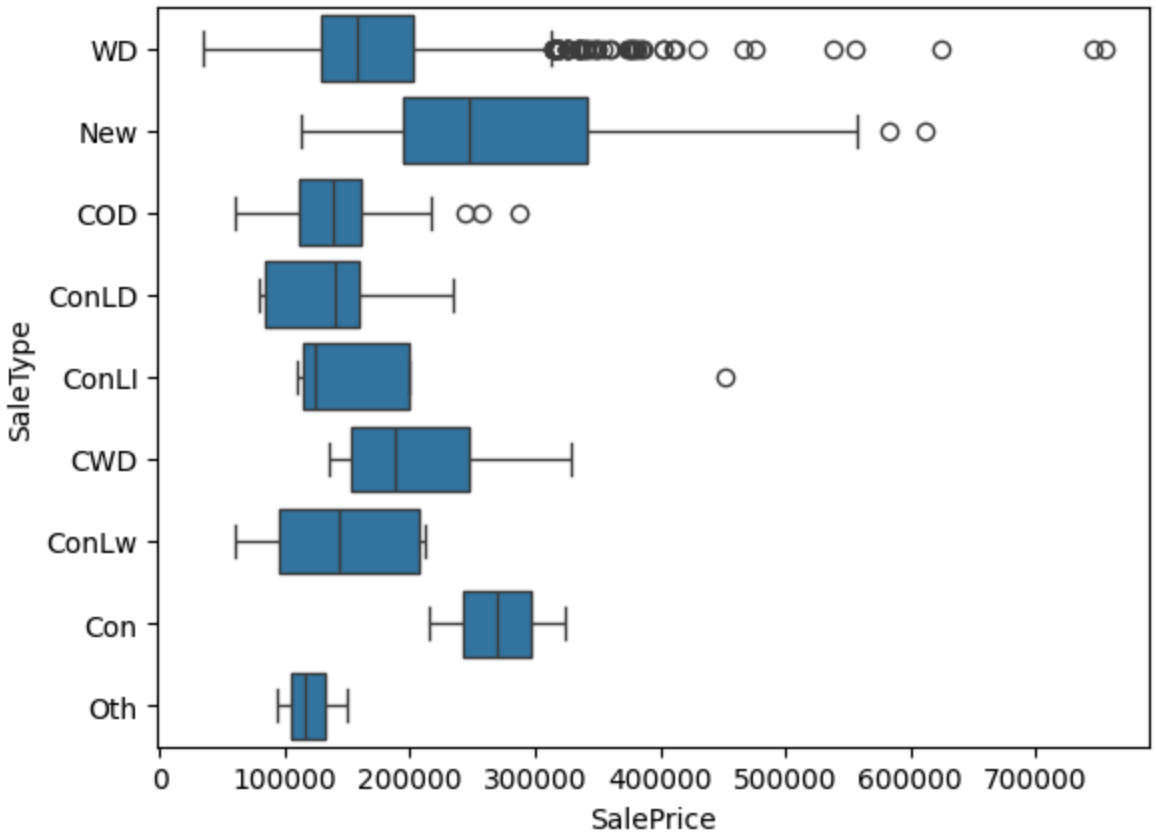
MoSold



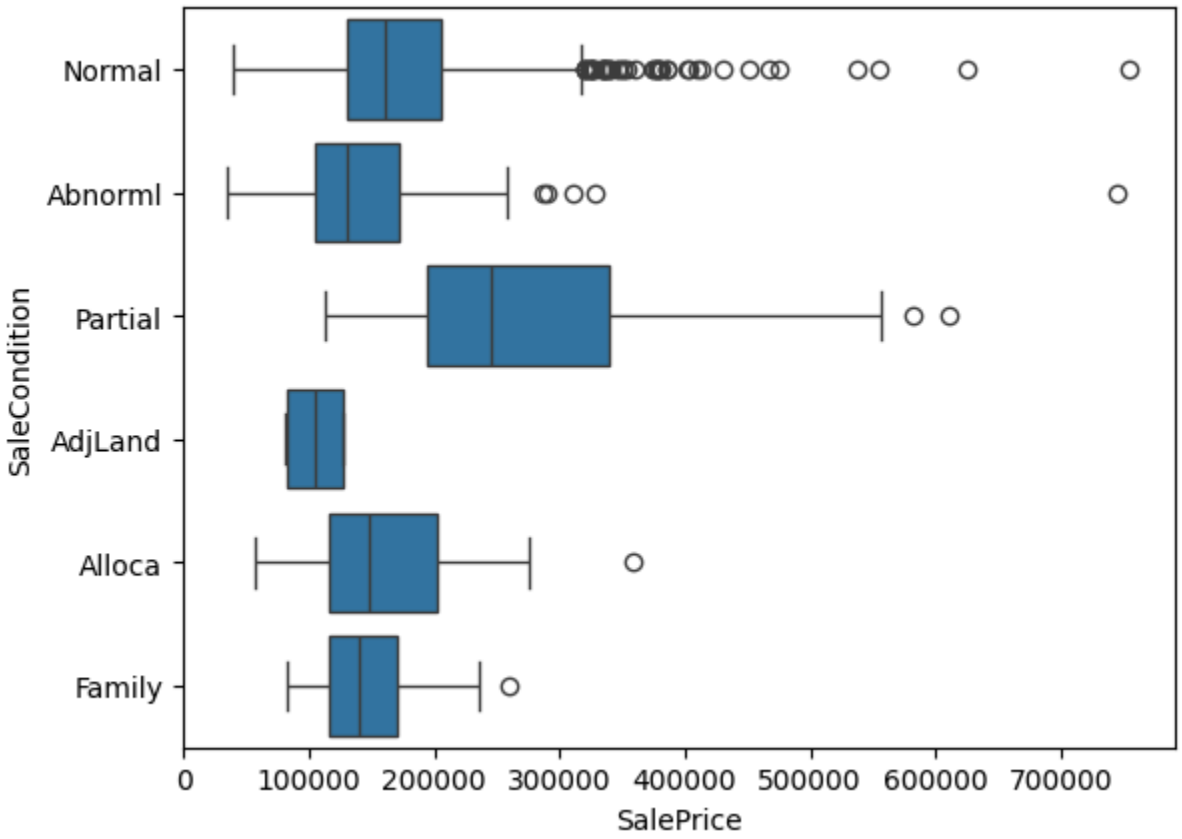
YrSold



SaleType



SaleCondition

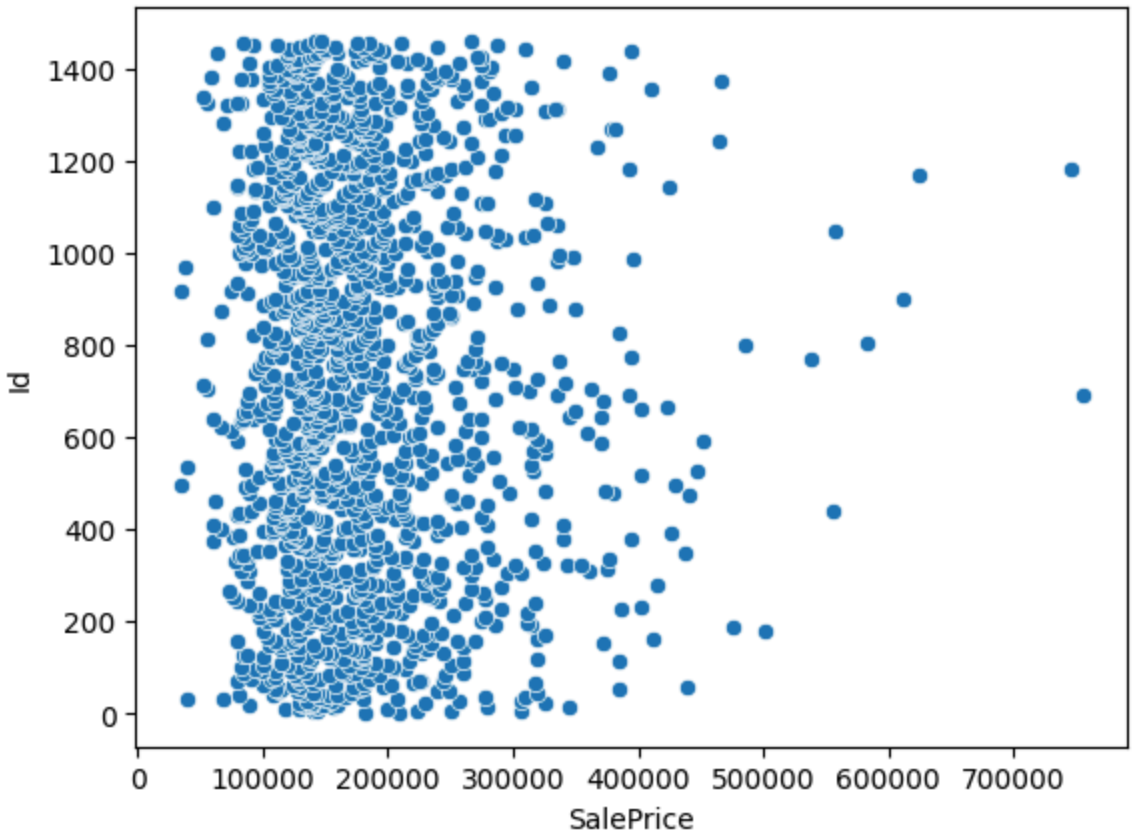


SalePrice

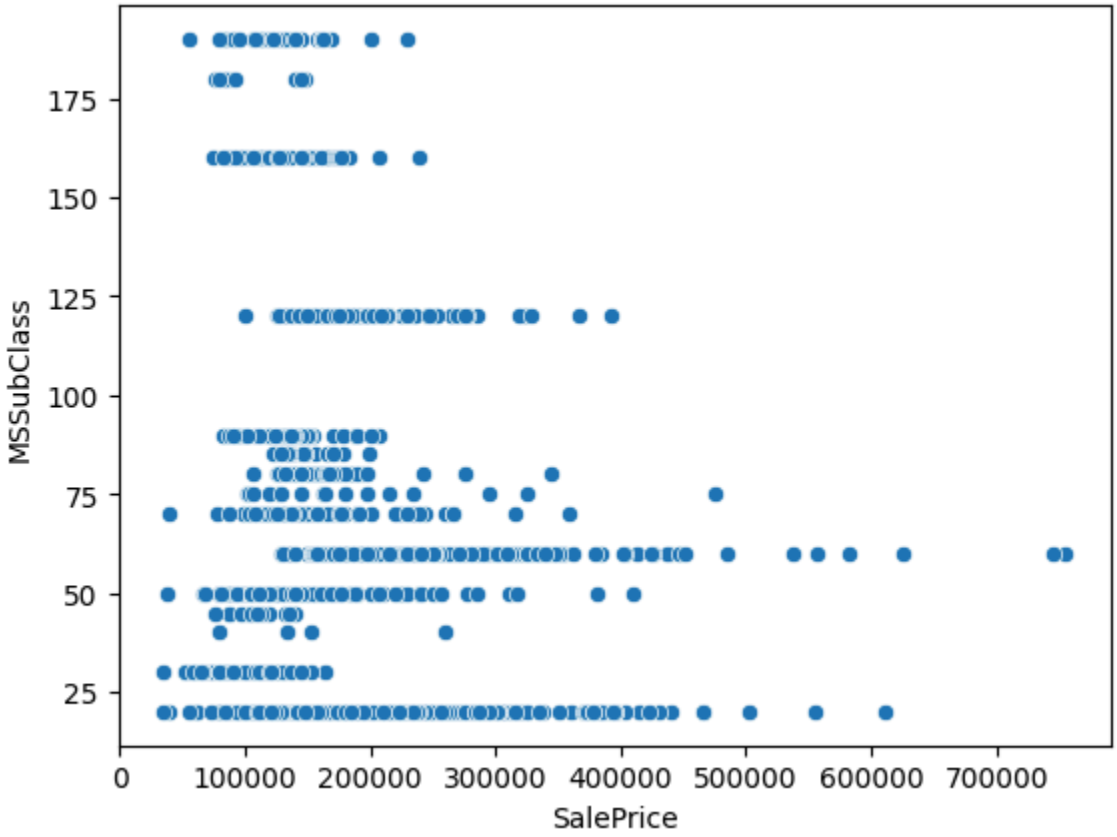
In [320…

```
for i in df[con].columns:
  sns.scatterplot(data=df,x='SalePrice',y=i)
  print(i)
  show()
```

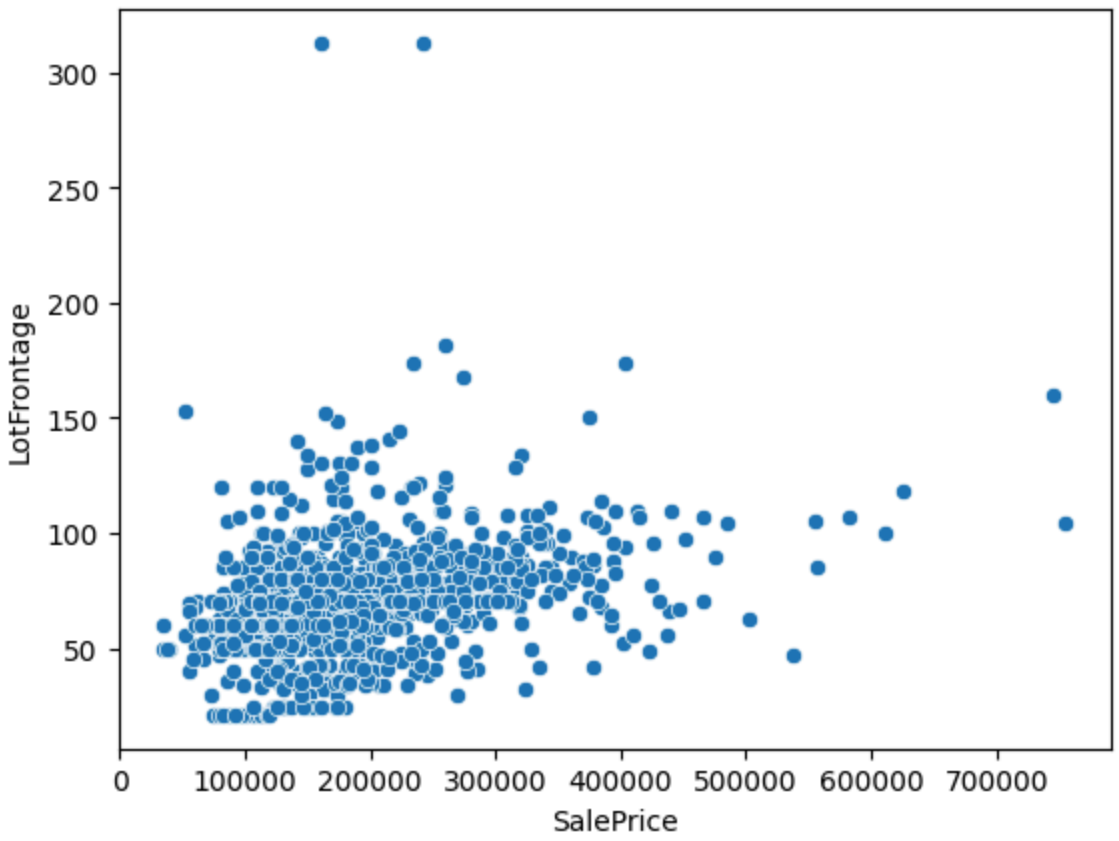
```
Id
```



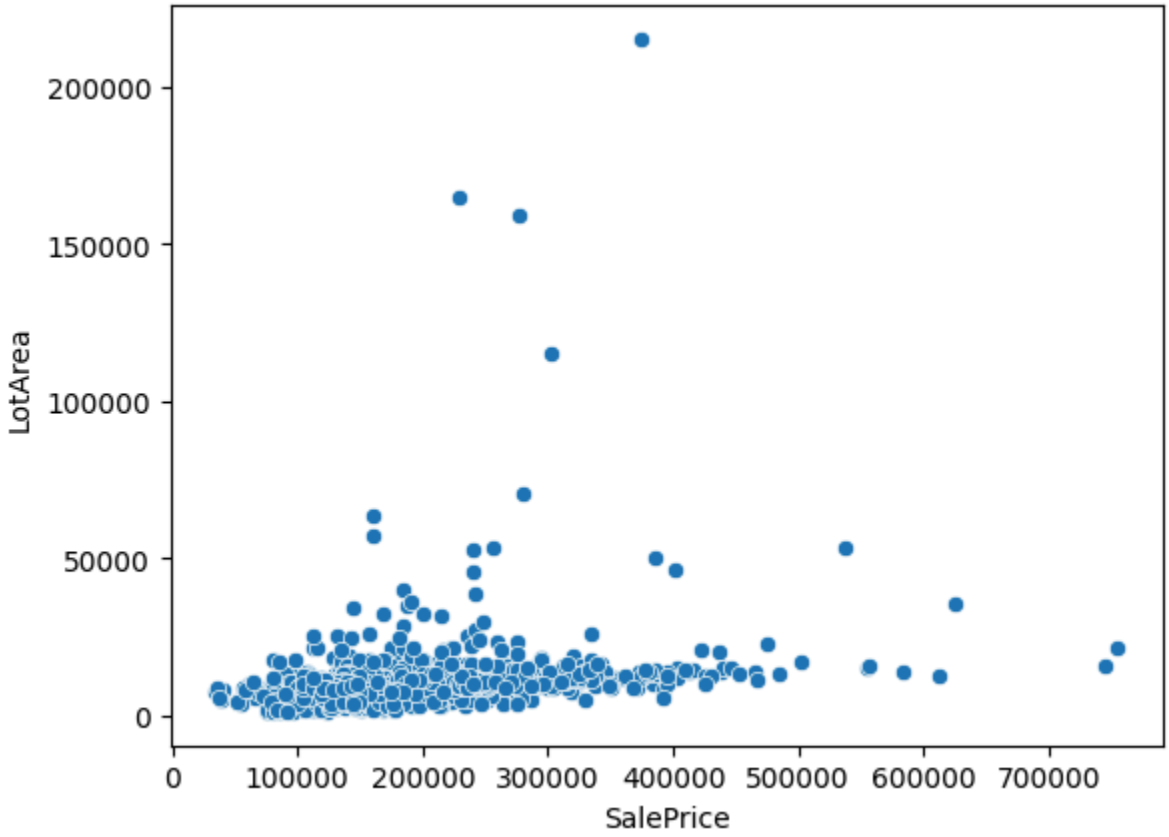
MSSubClass



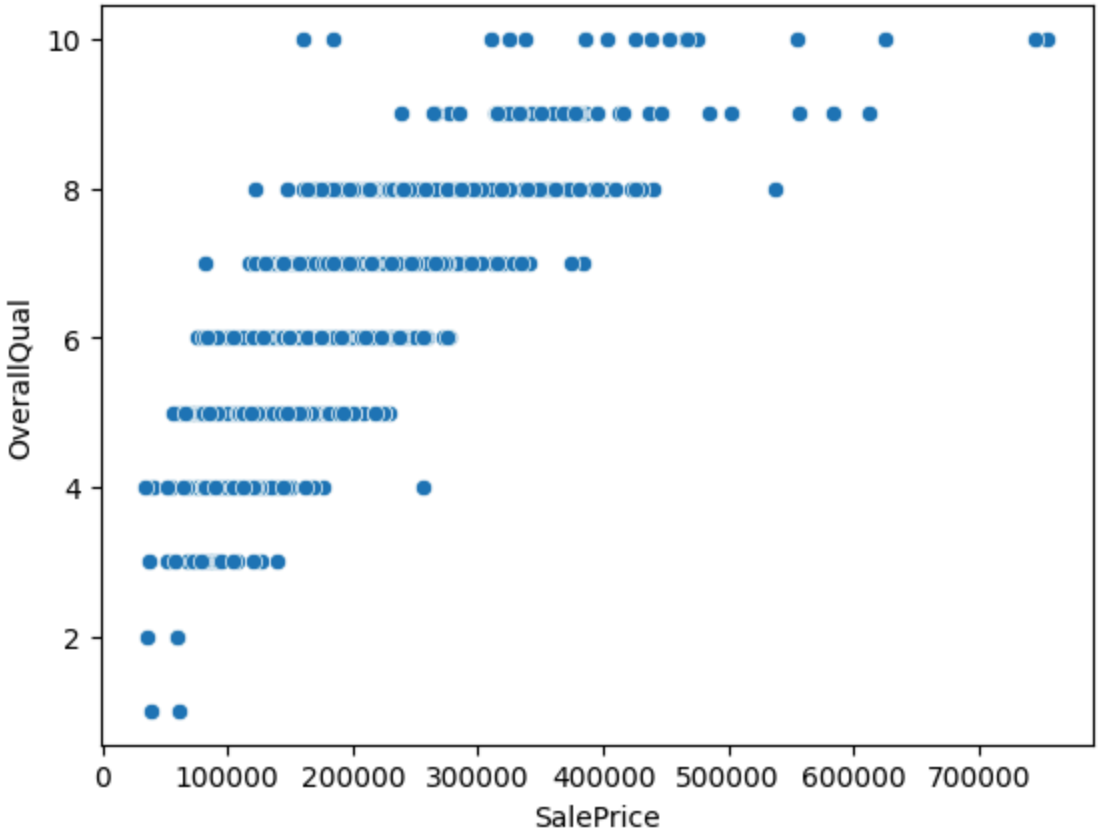
LotFrontage



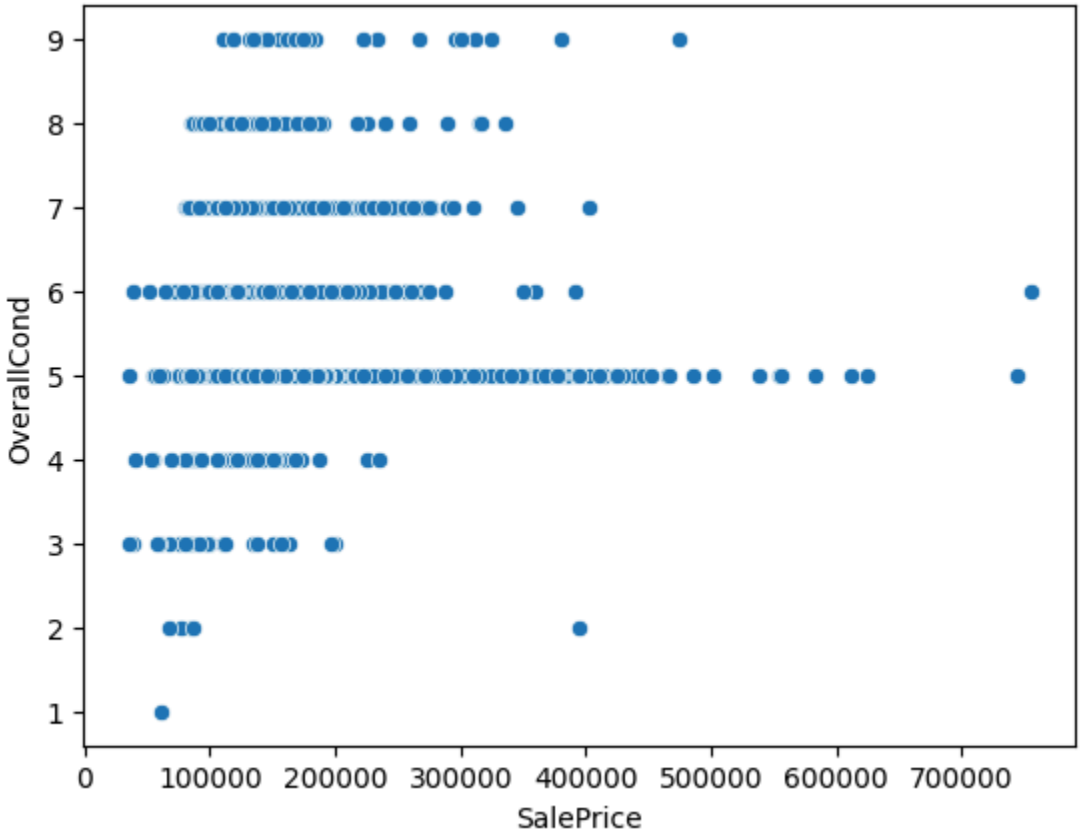
LotArea



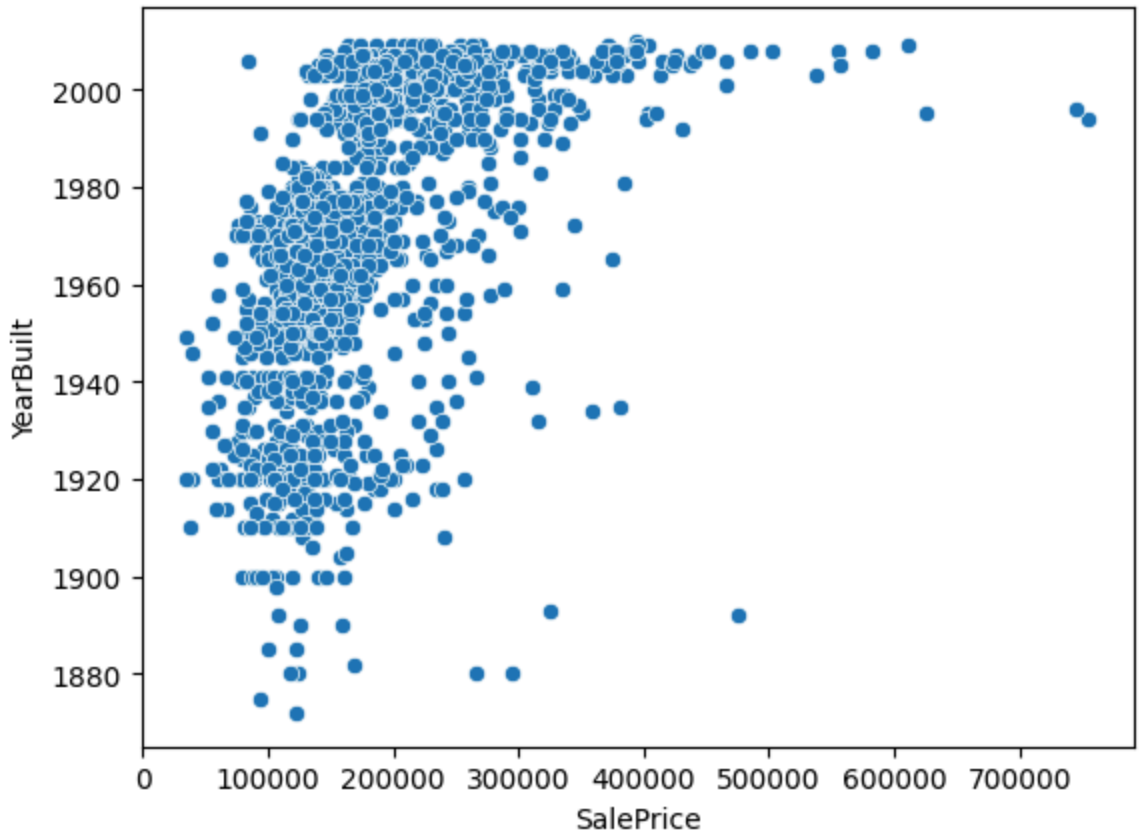
OverallQual



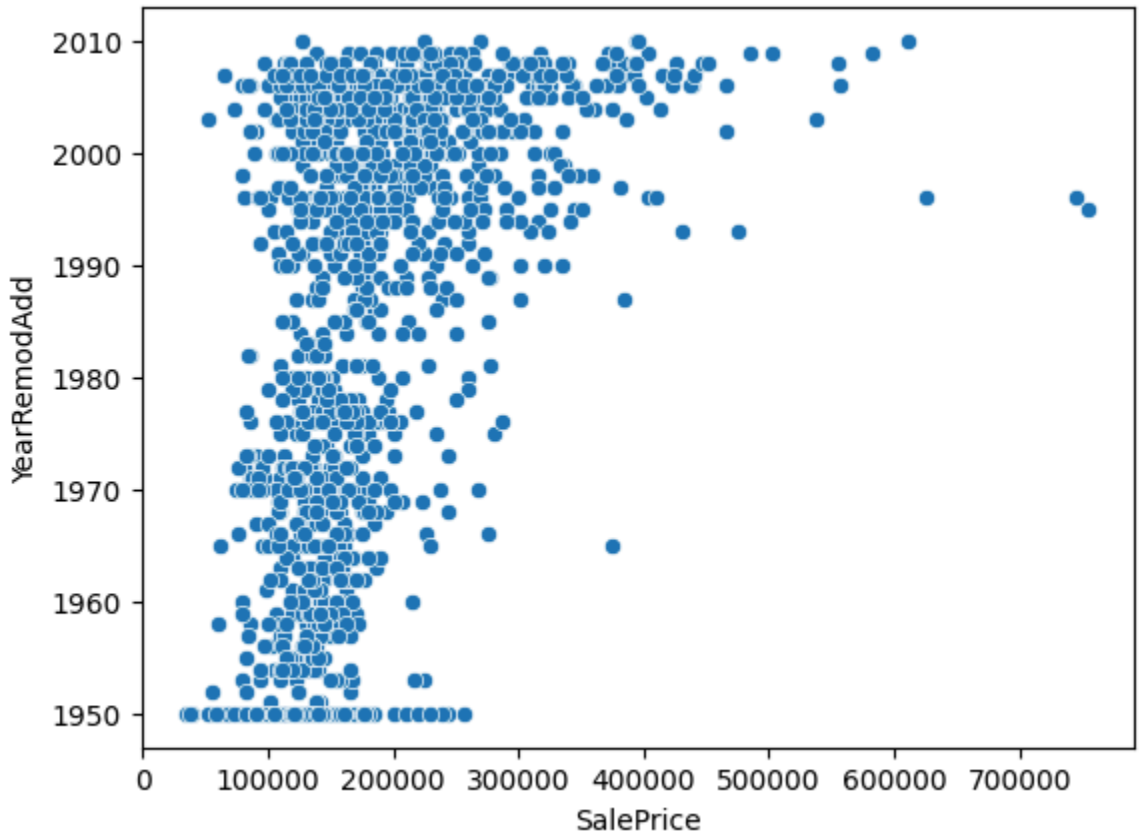
`OverallCond`



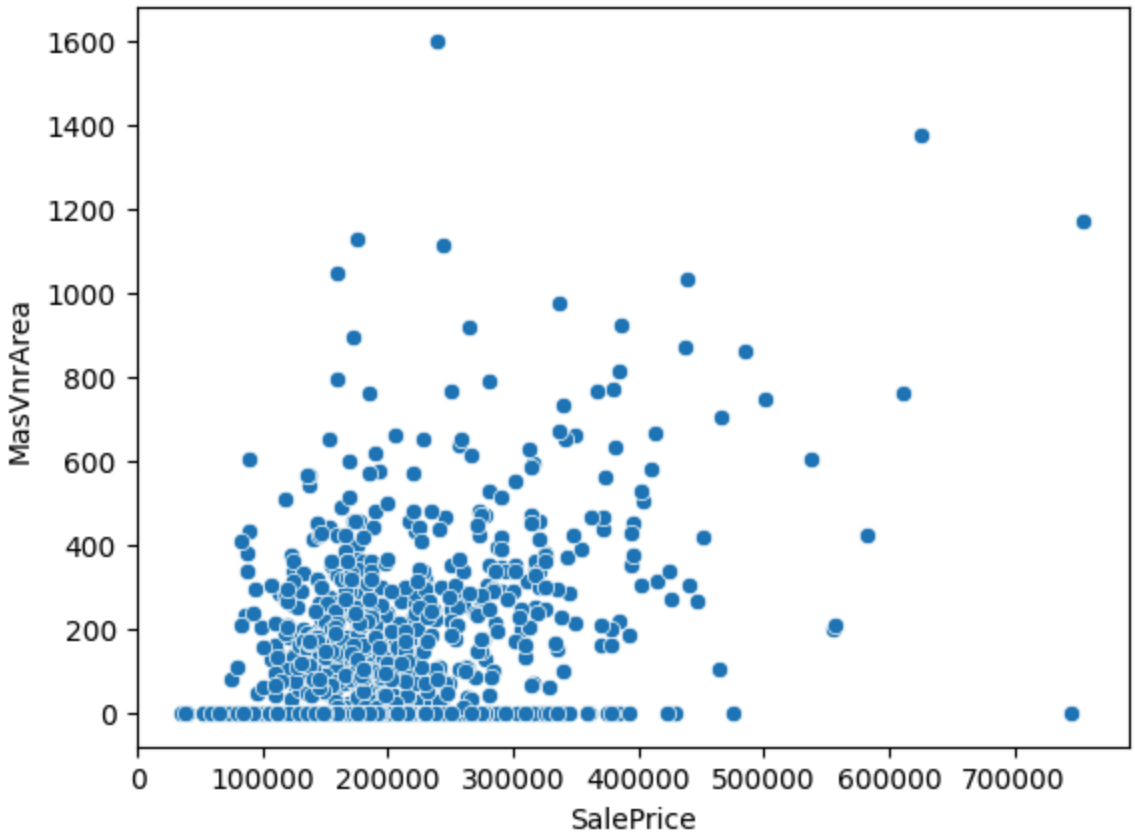
`YearBuilt`



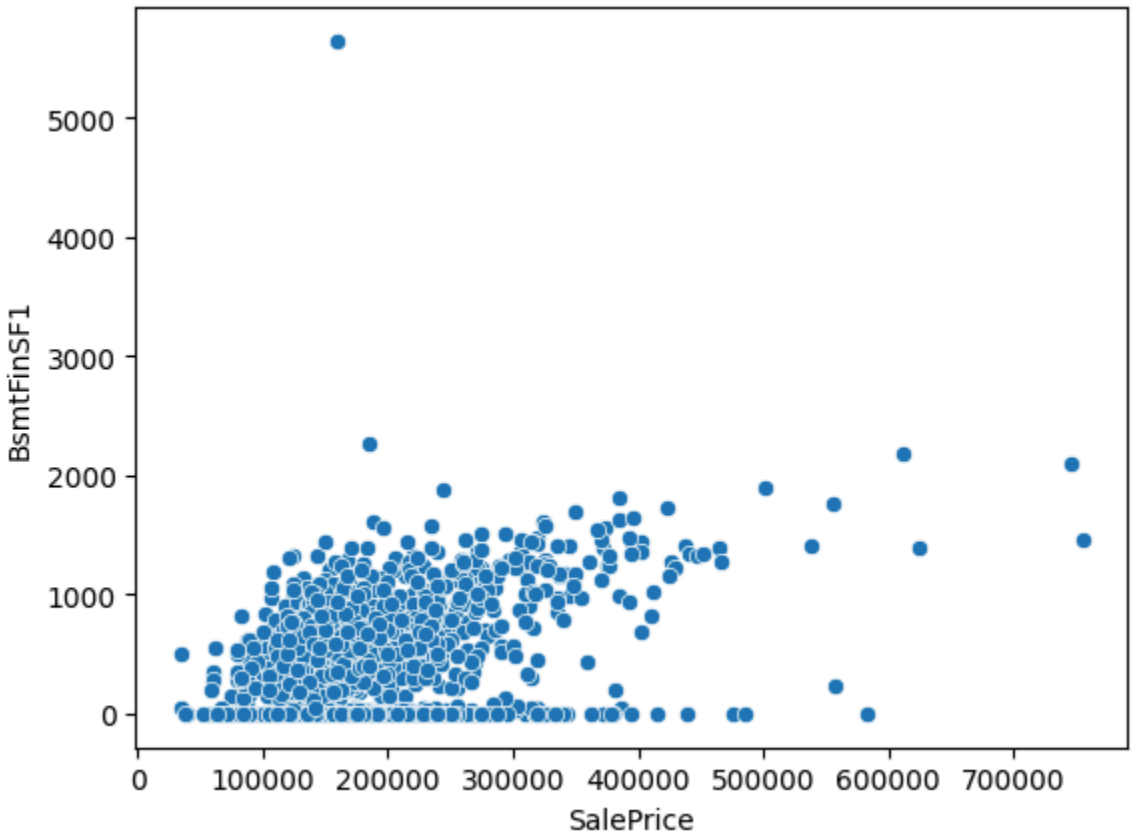
YearRemodAdd



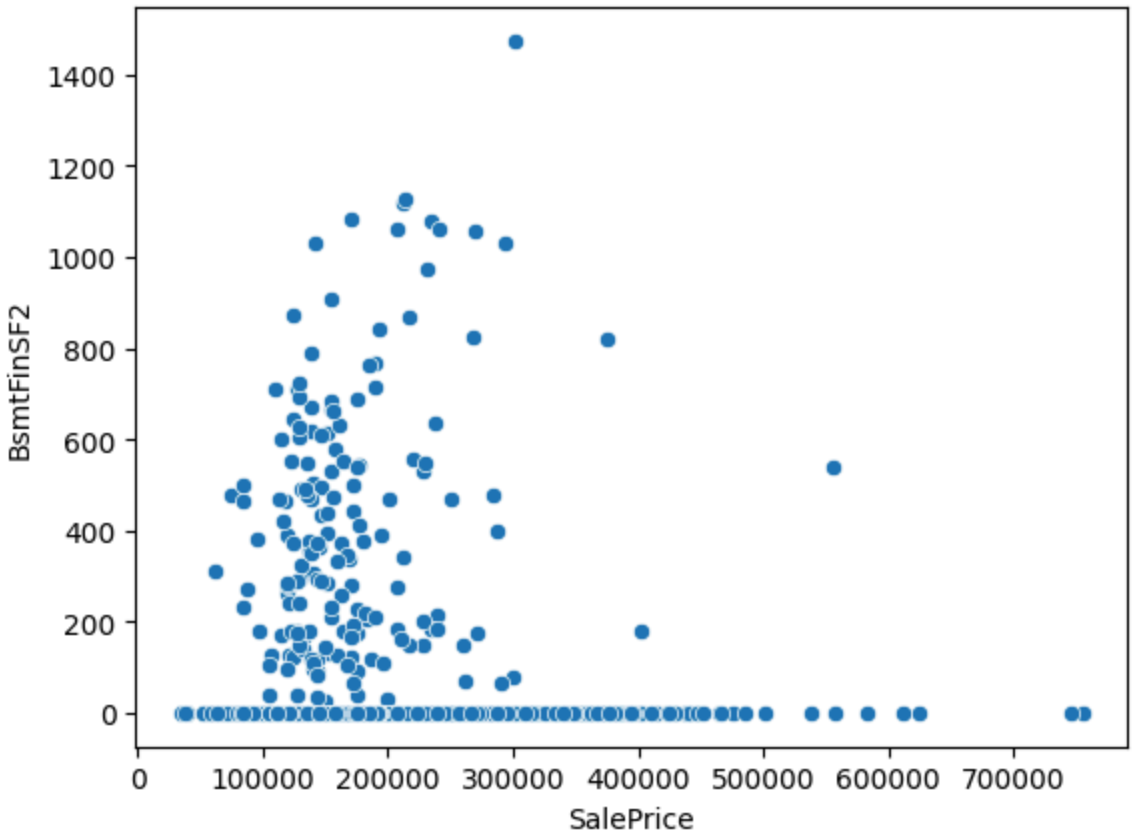
MasVnrArea



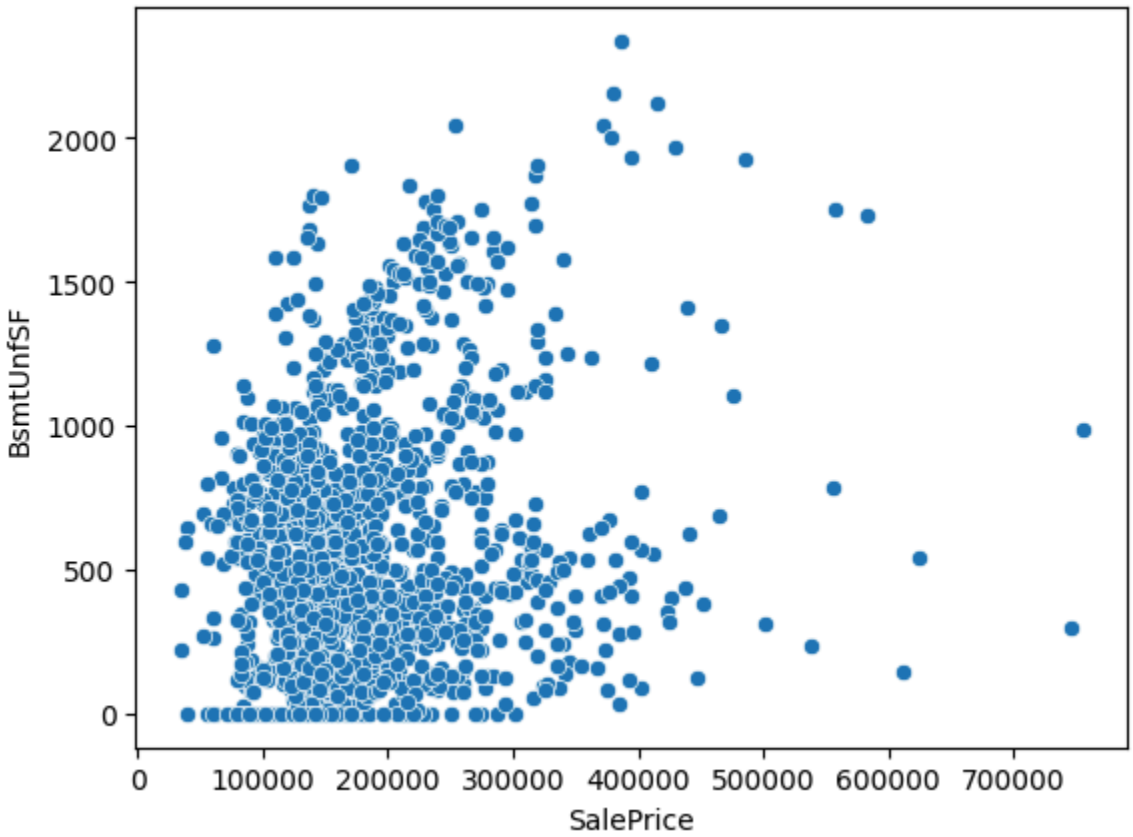
BsmtFinSF1



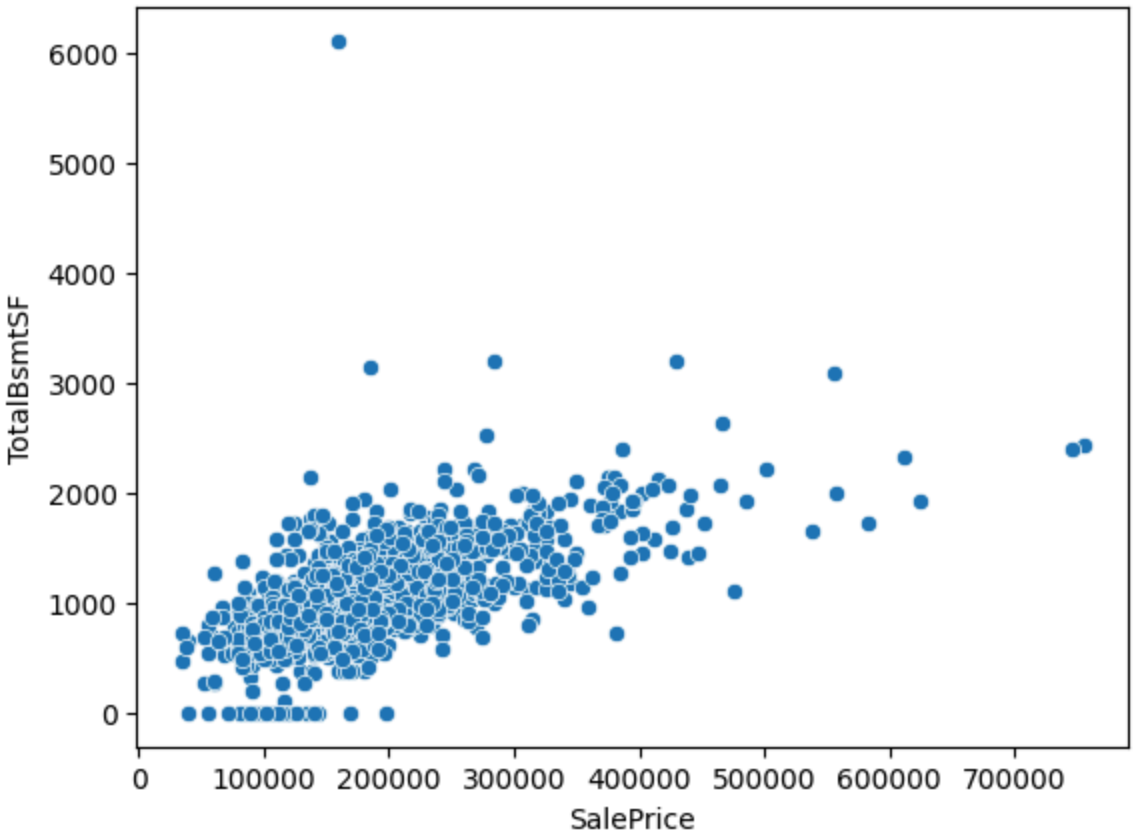
BsmtFinSF2



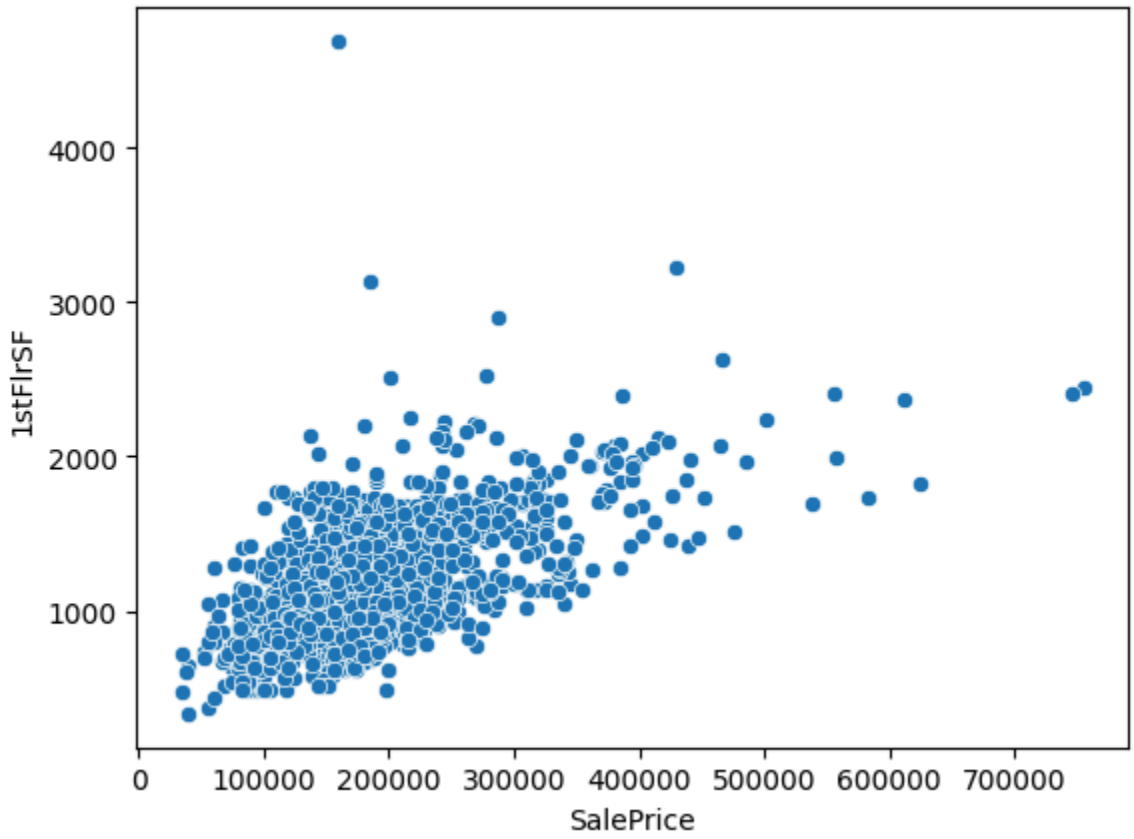
BsmtUnfSF



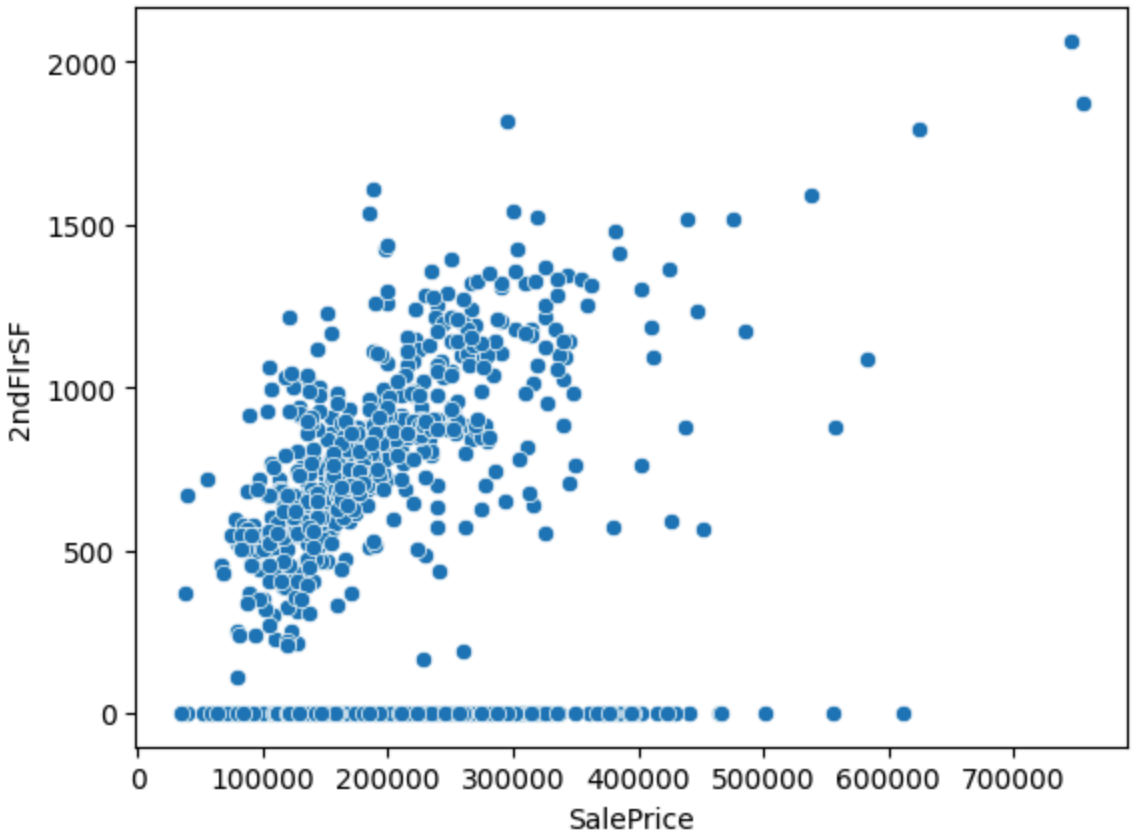
TotalBsmtSF



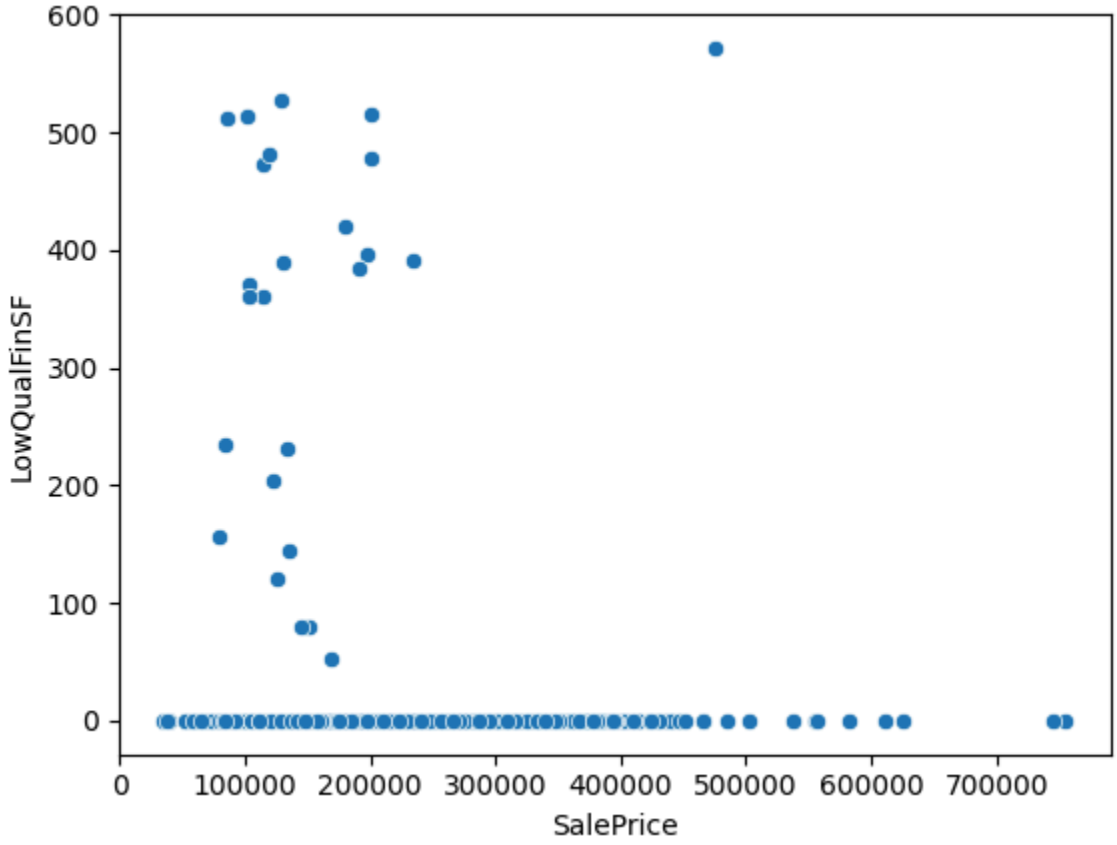
1stFlrSF



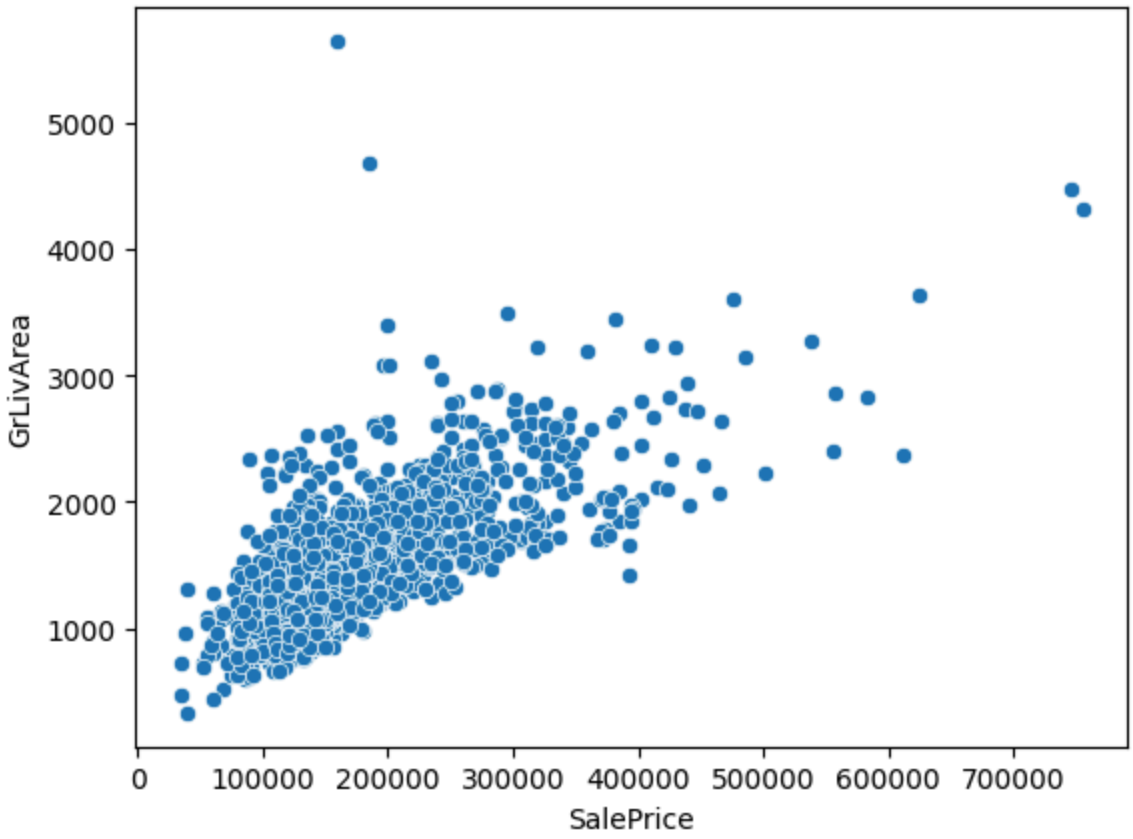
2ndFlrSF



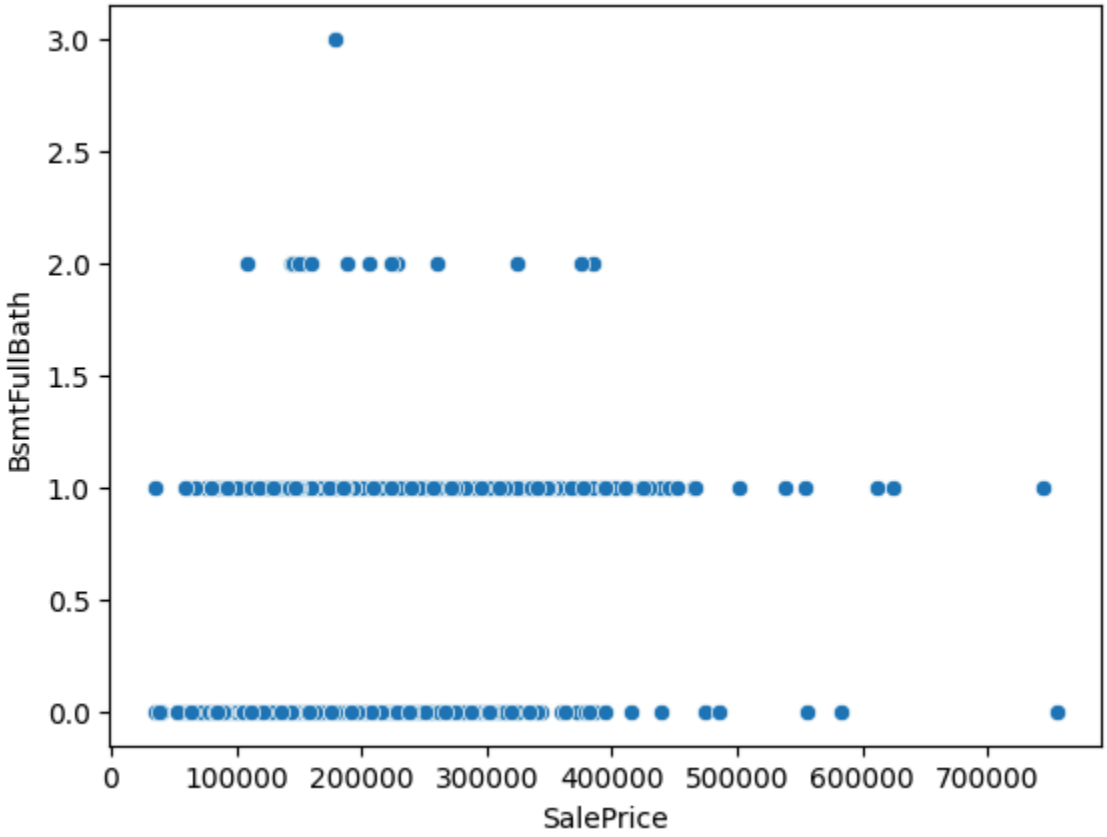
LowQualFinSF



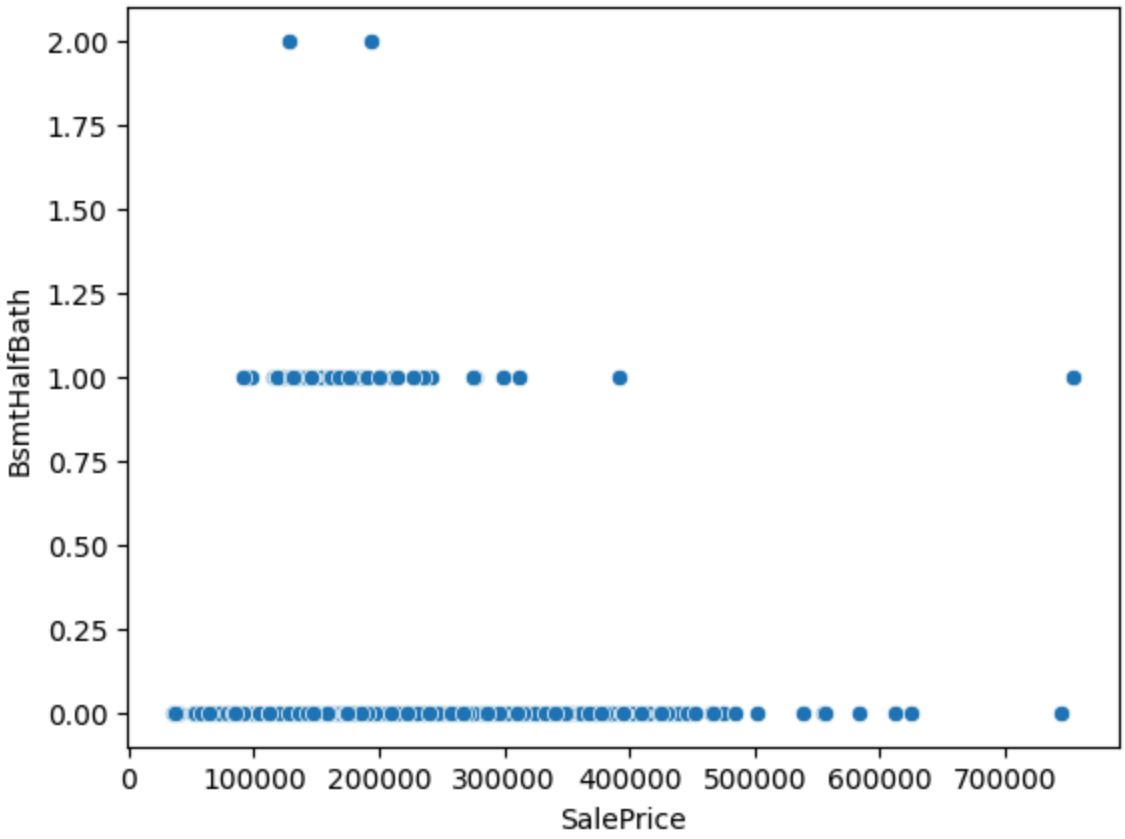
GrLivArea



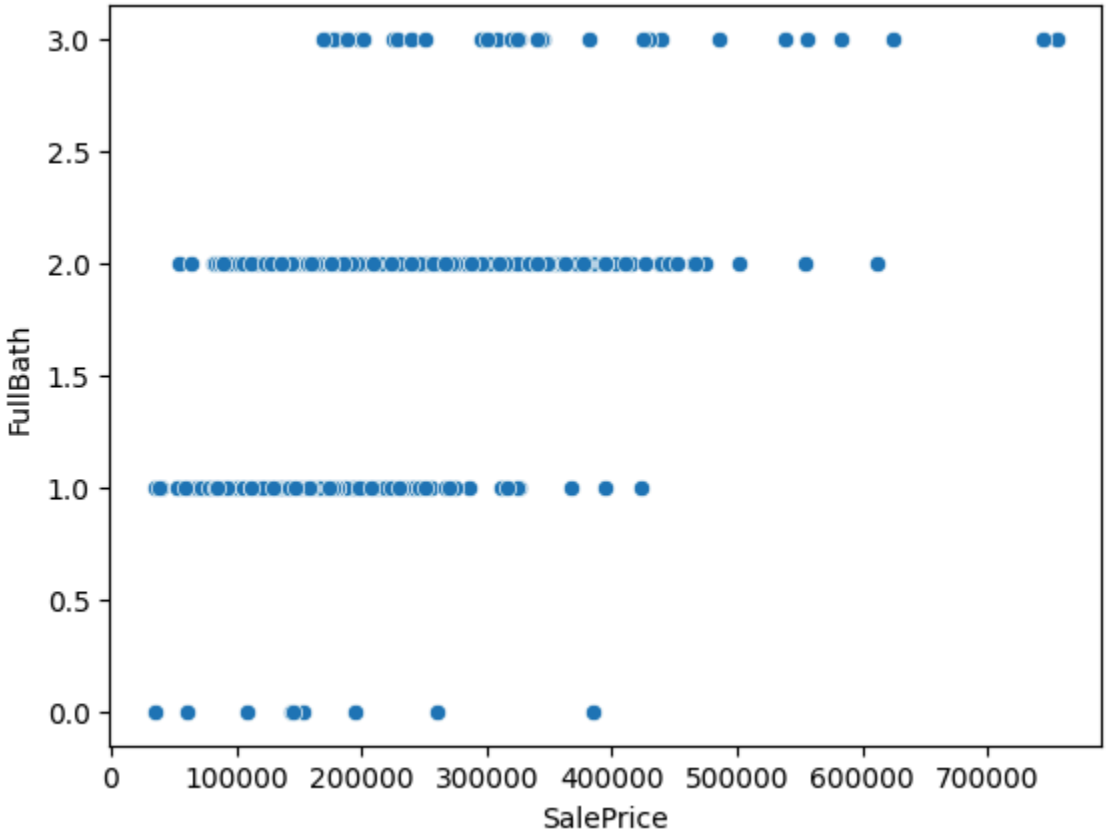
BsmtFullBath



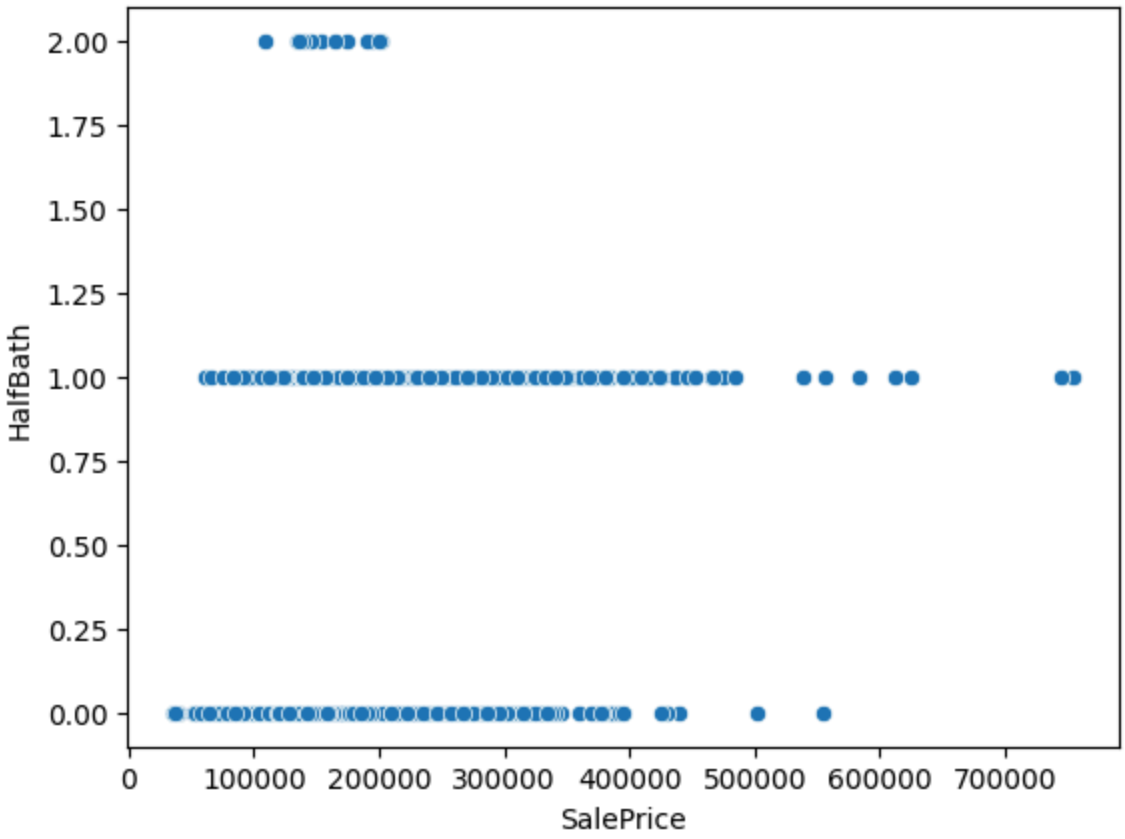
BsmtHalfBath



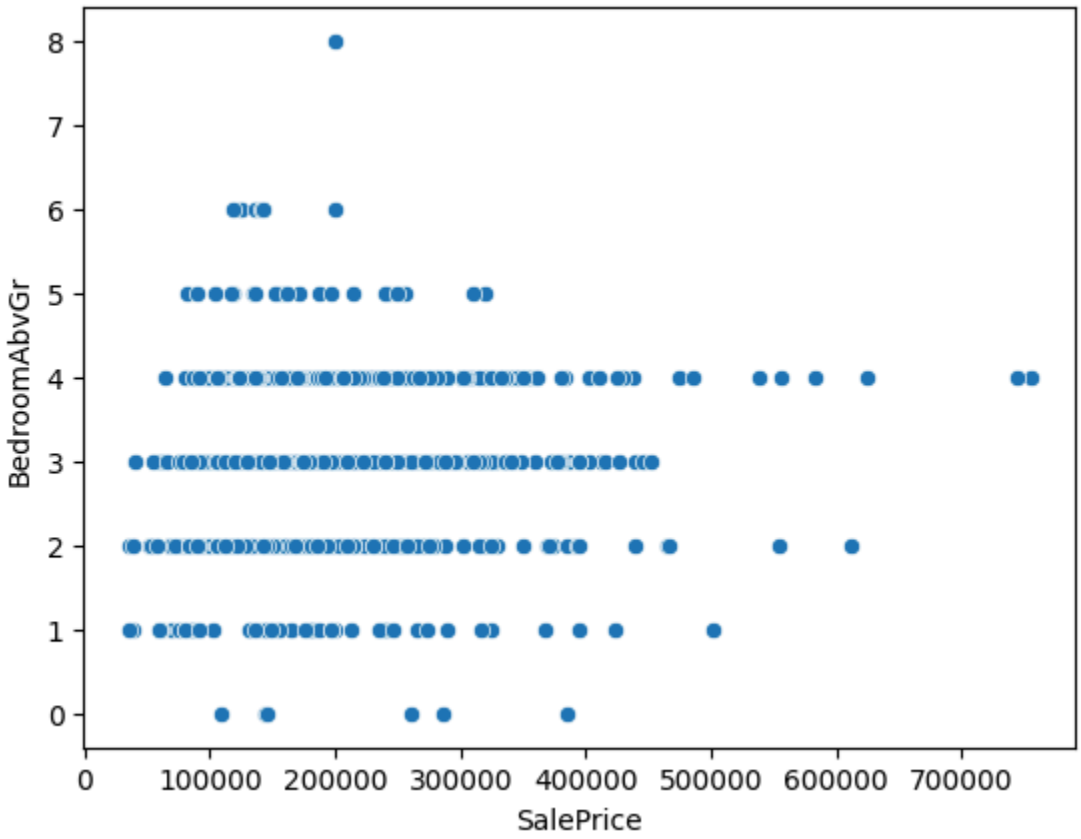
FullBath



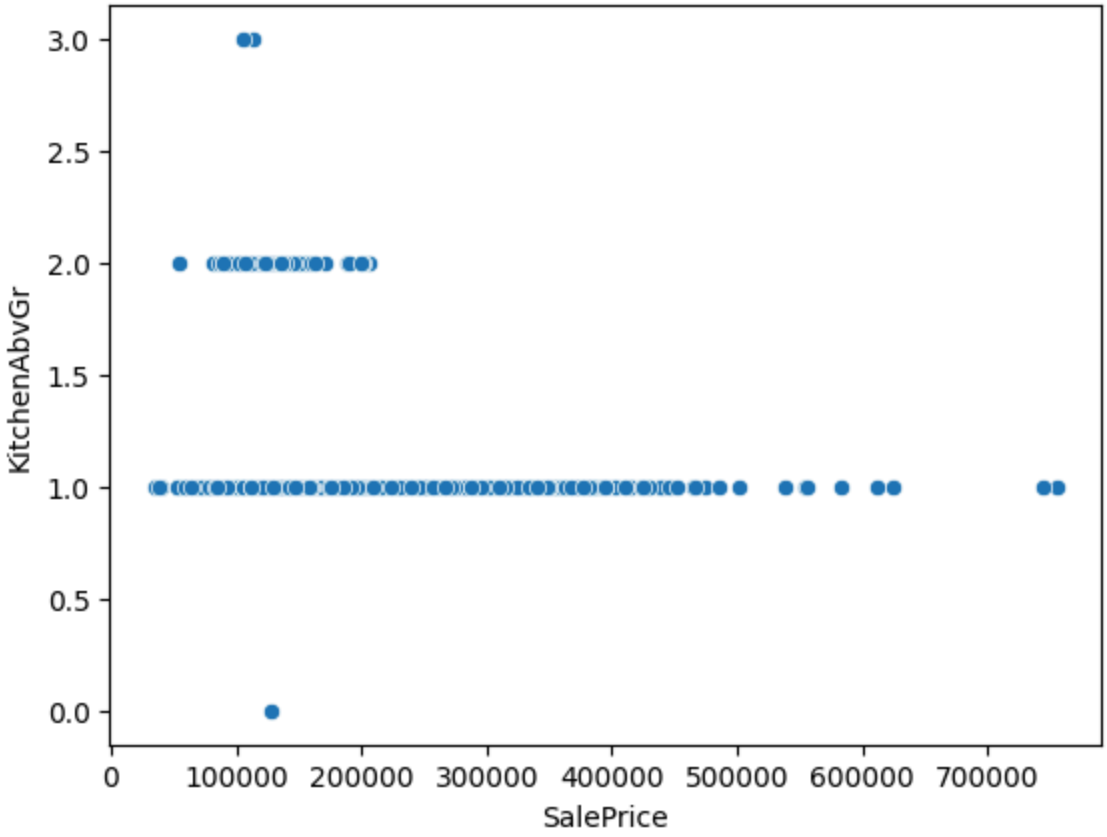
HalfBath



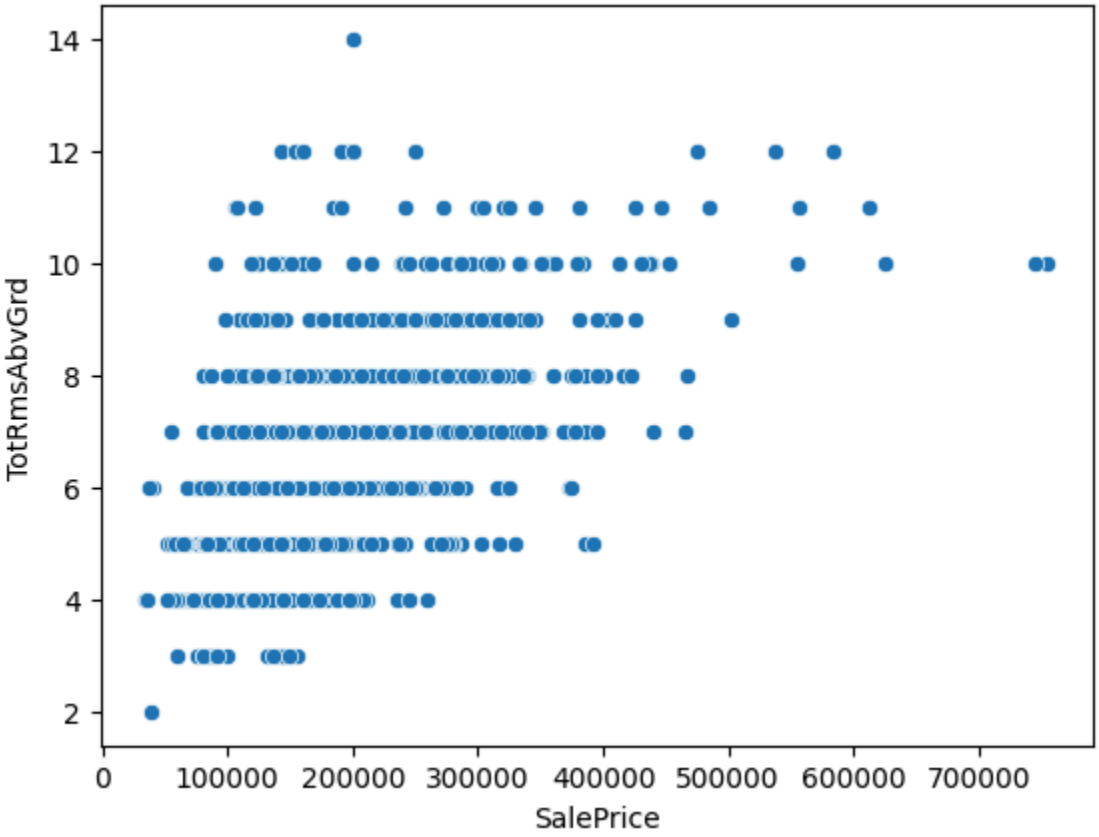
BedroomAbvGr



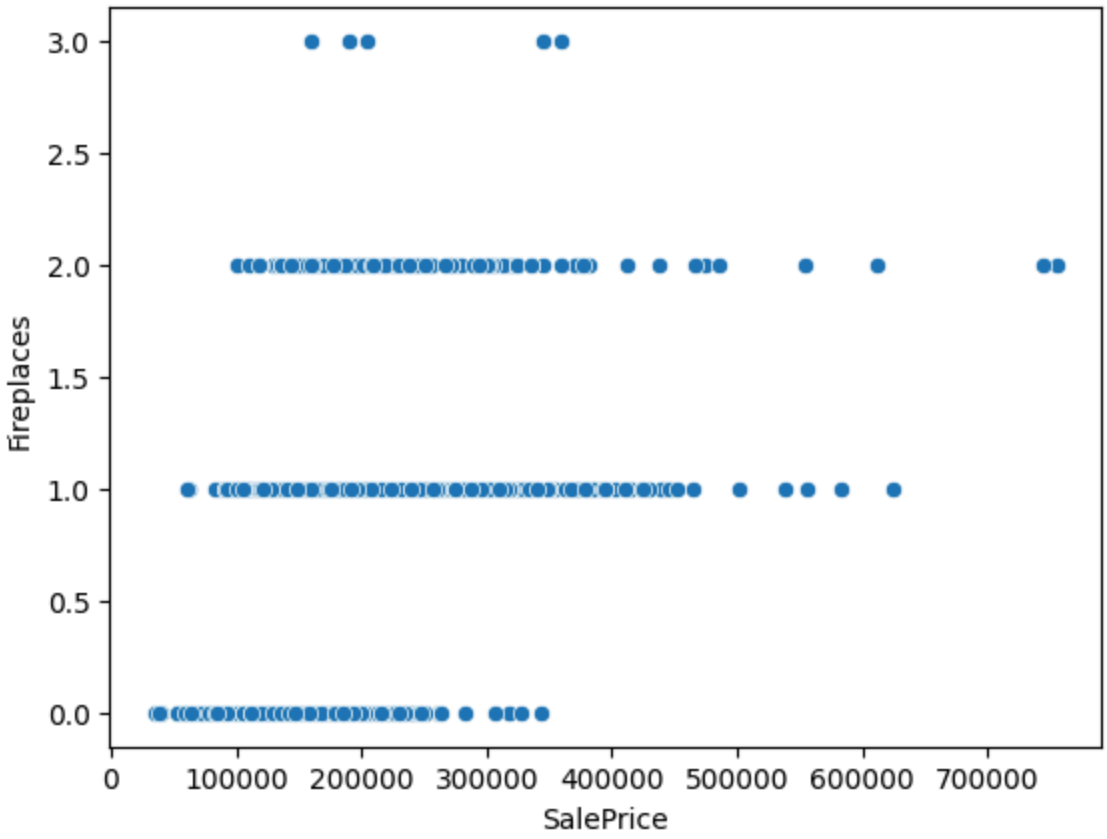
KitchenAbvGr



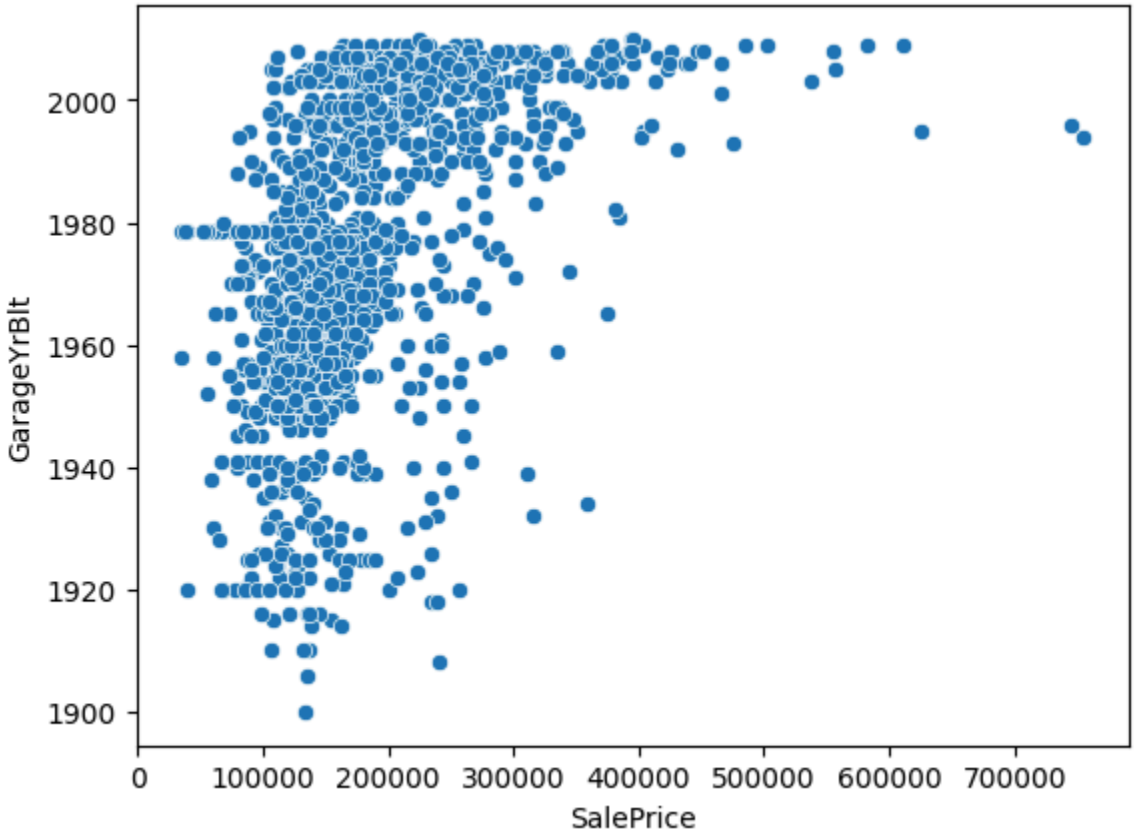
TotRmsAbvGrd



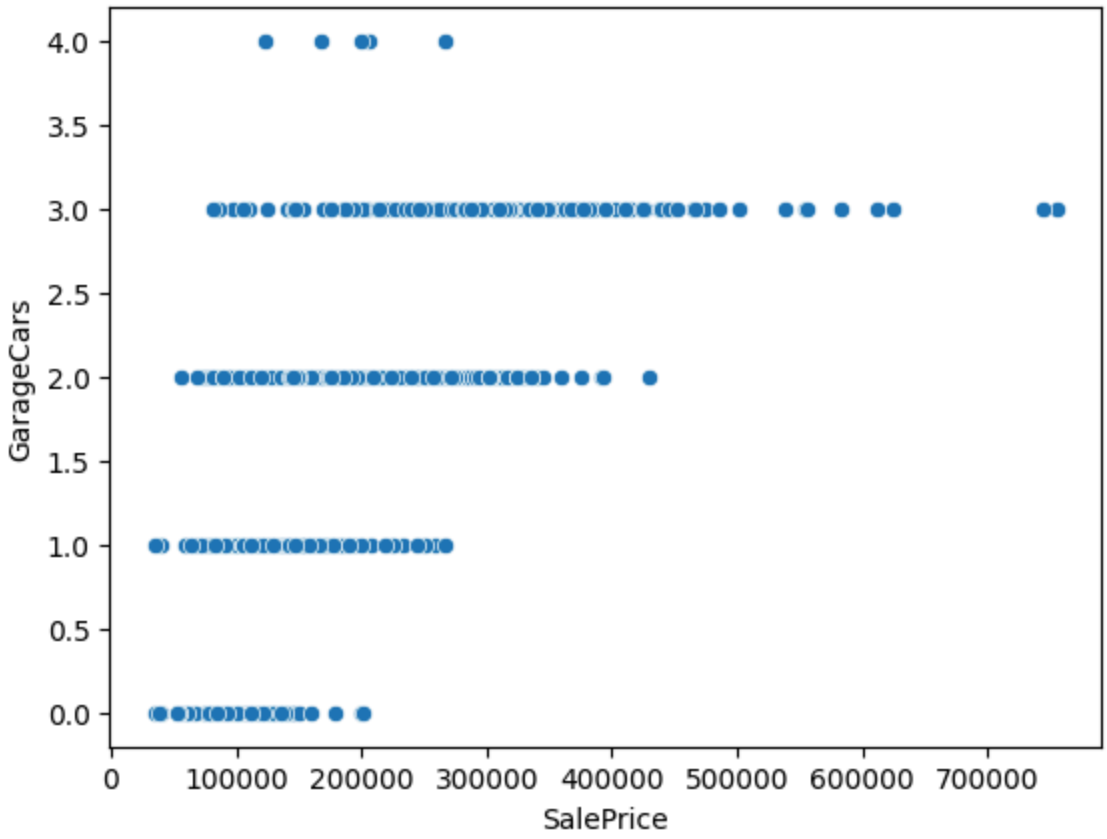
Fireplaces



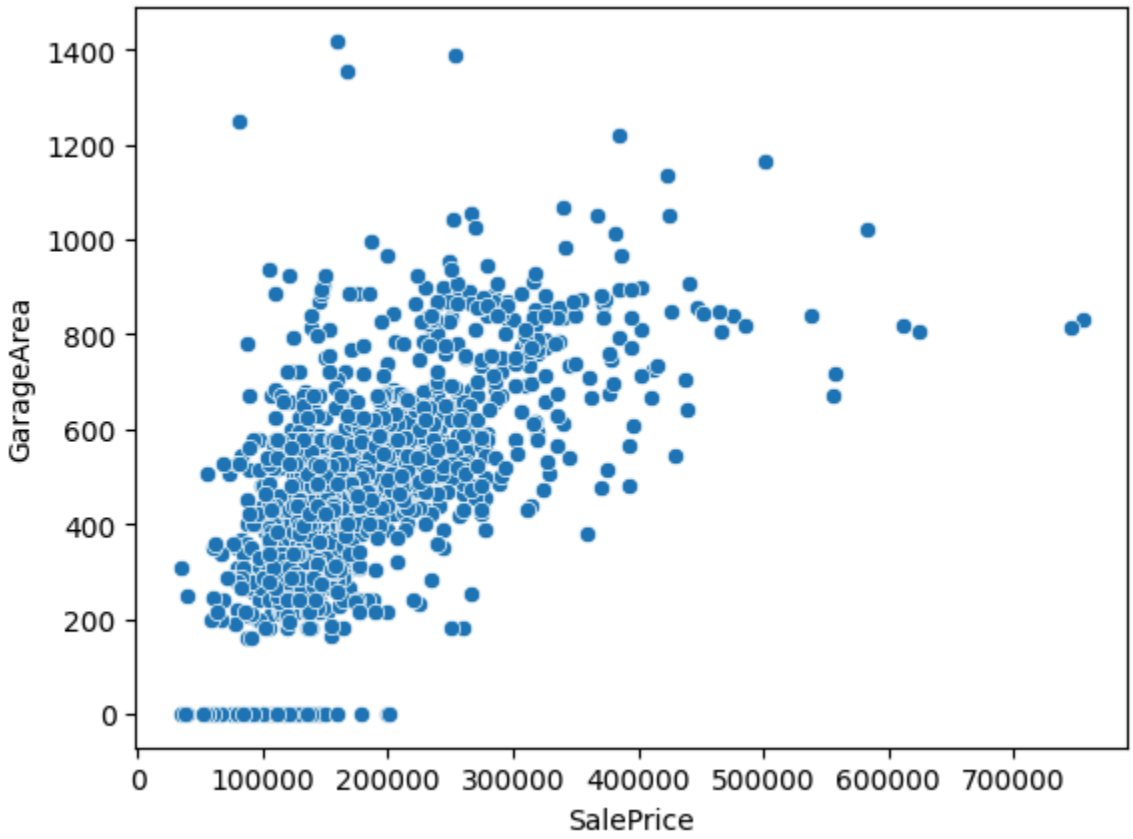
GarageYrBlt



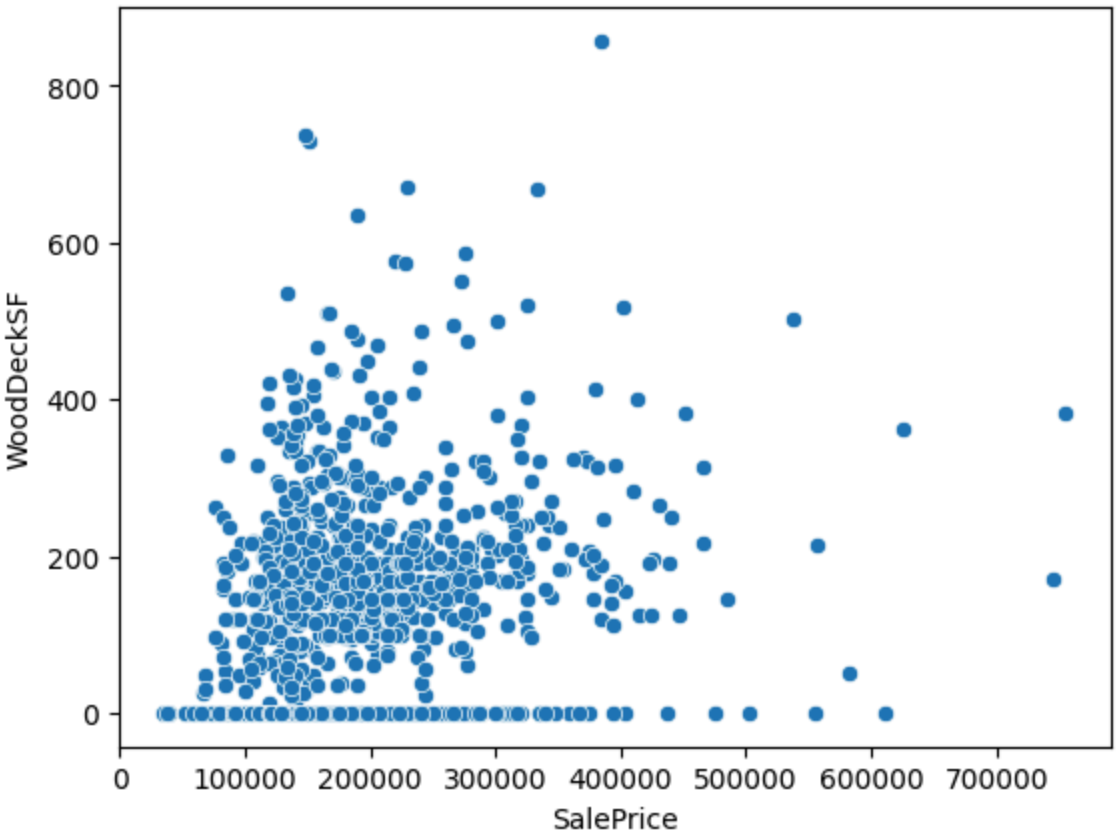
GarageCars



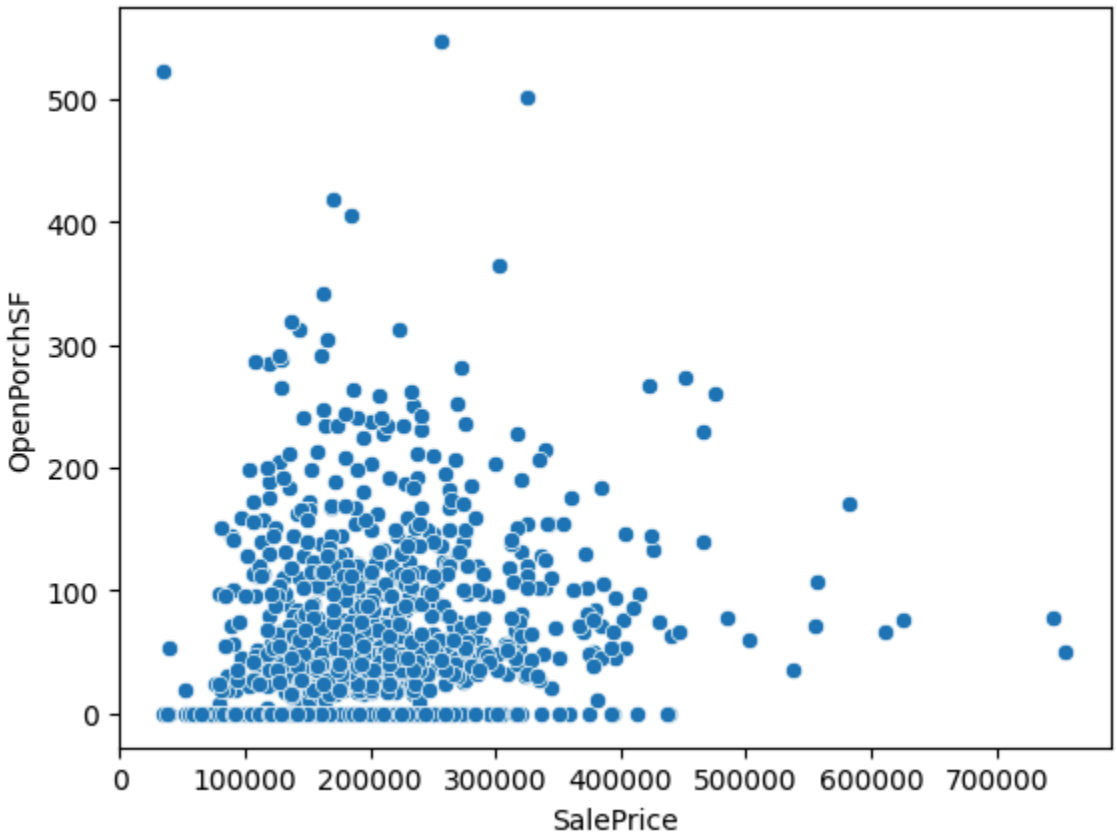
GarageArea



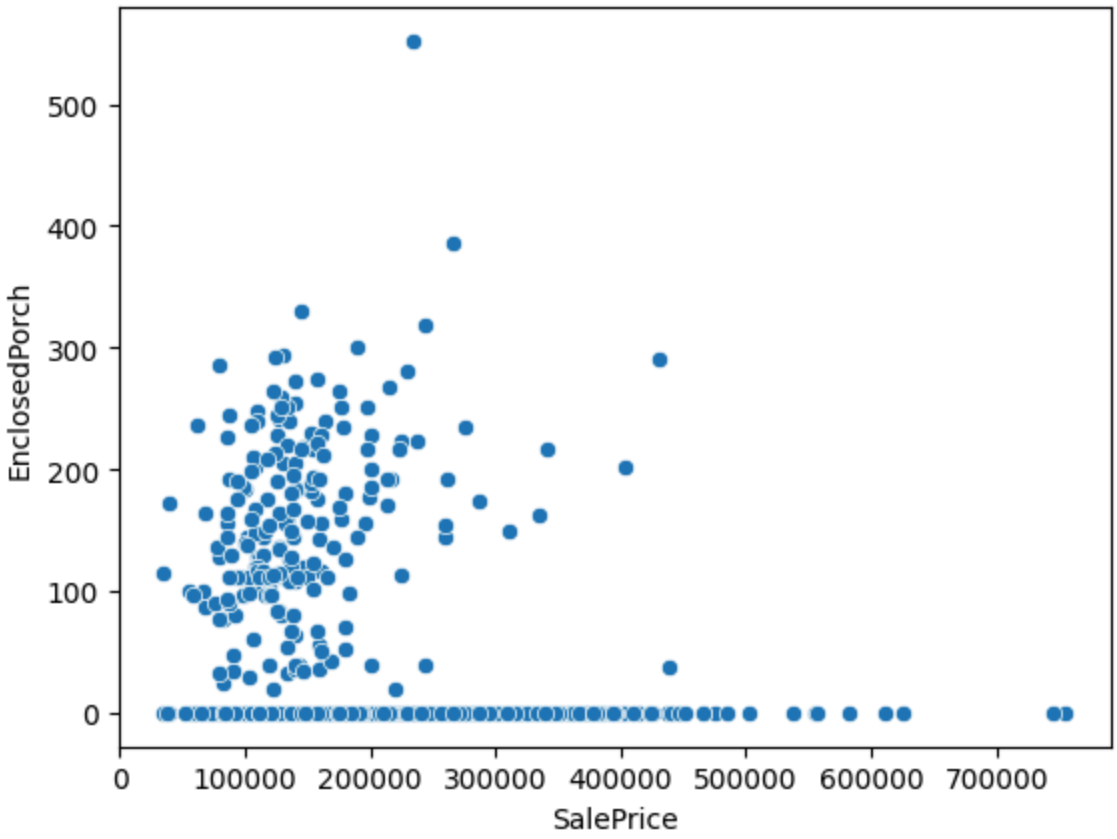
WoodDeckSF



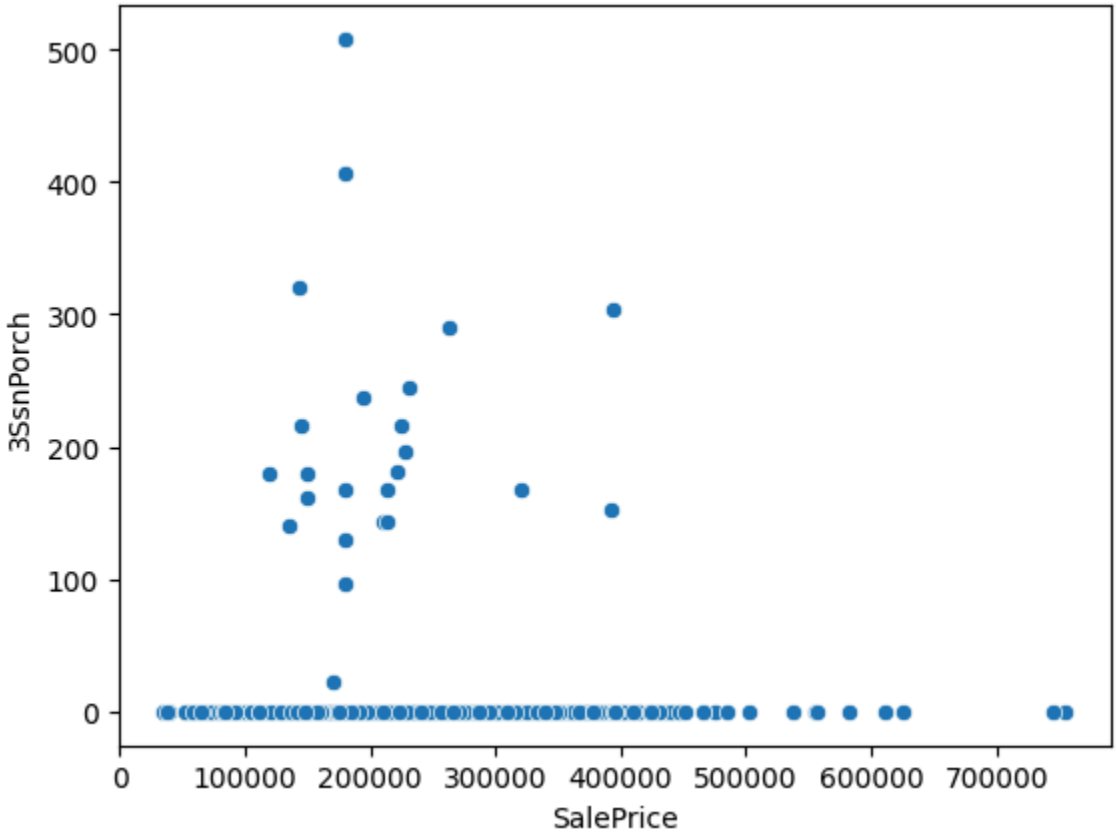
OpenPorchSF



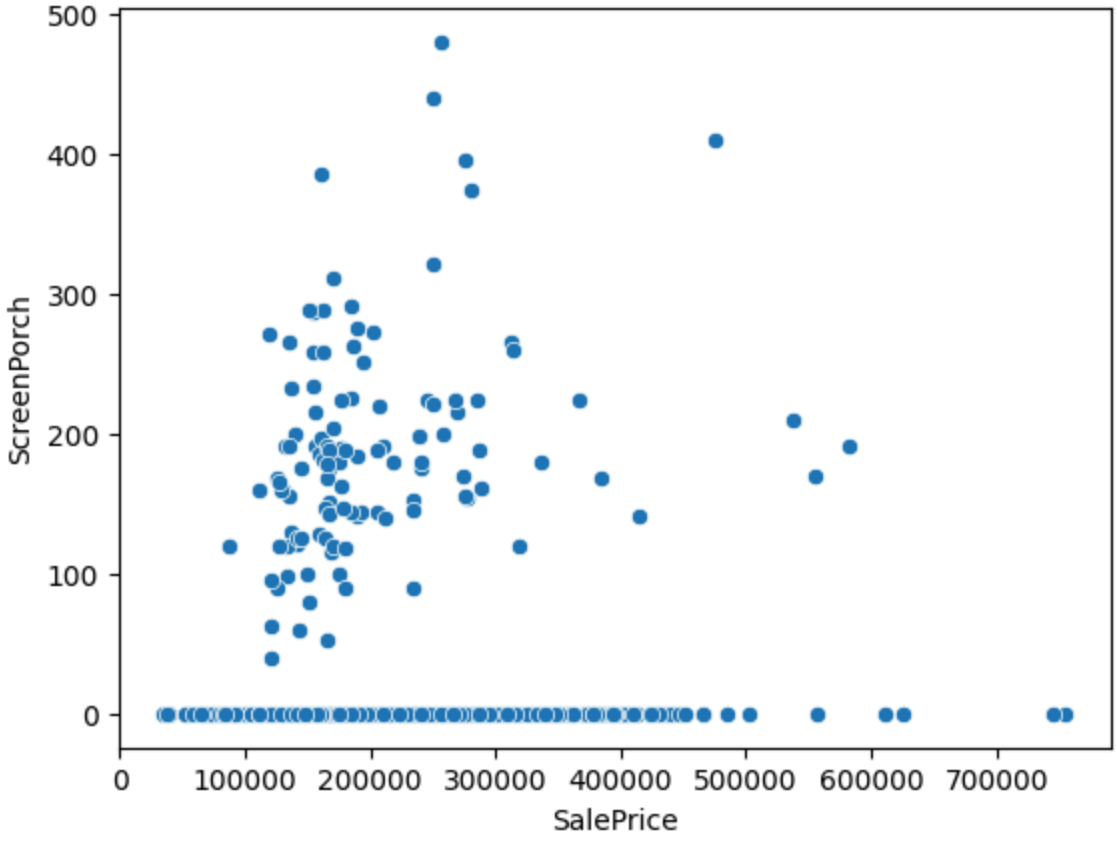
EnclosedPorch



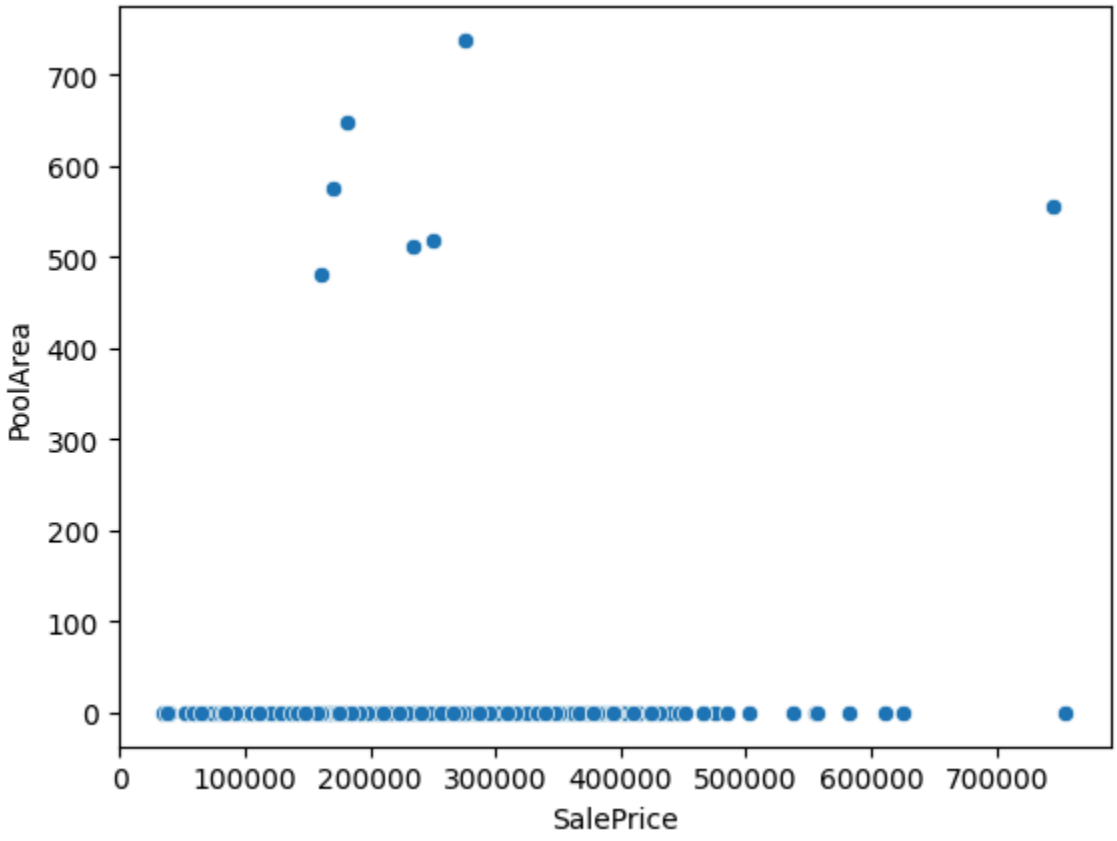
3SsnPorch



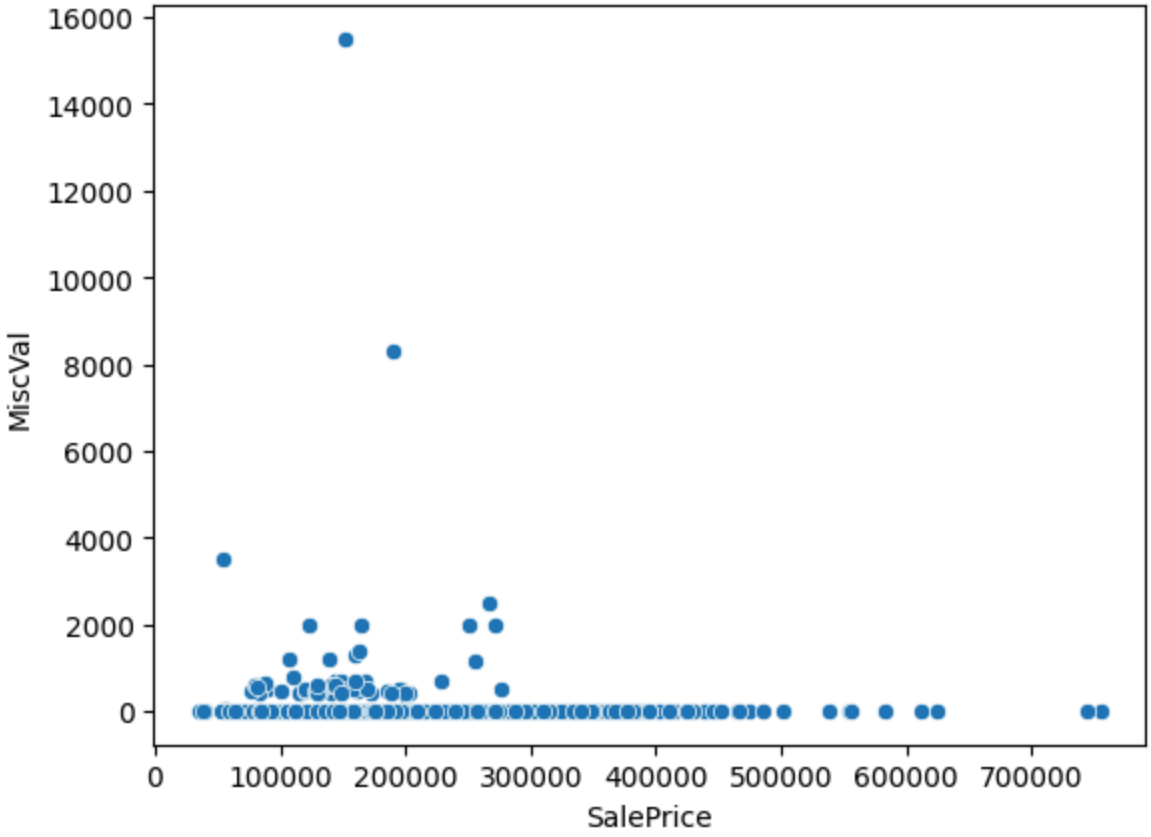
ScreenPorch



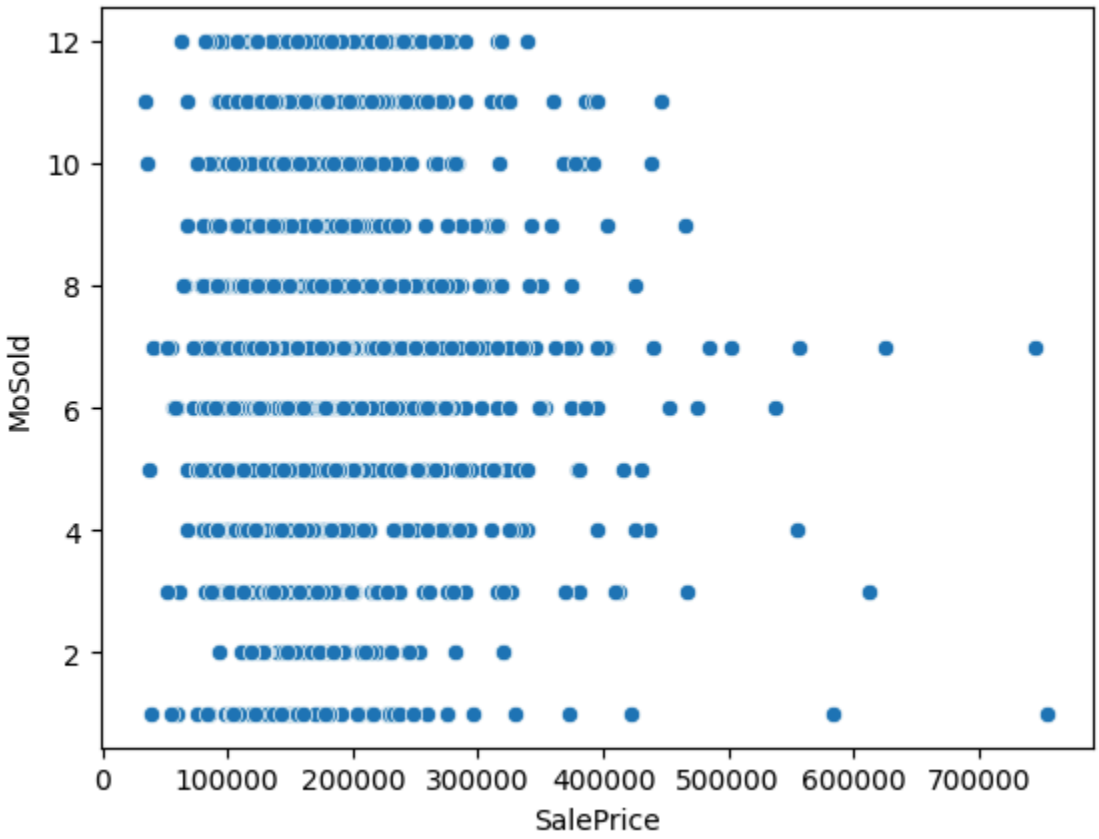
PoolArea



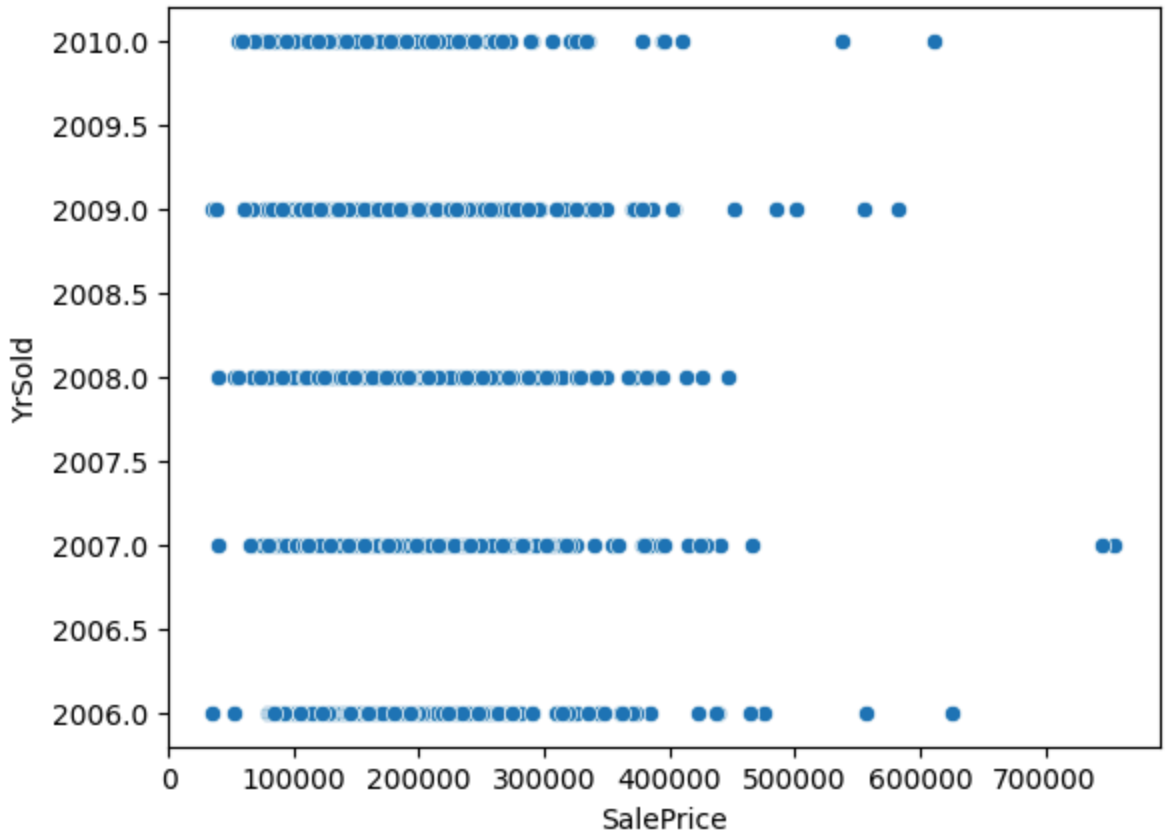
MiscVal



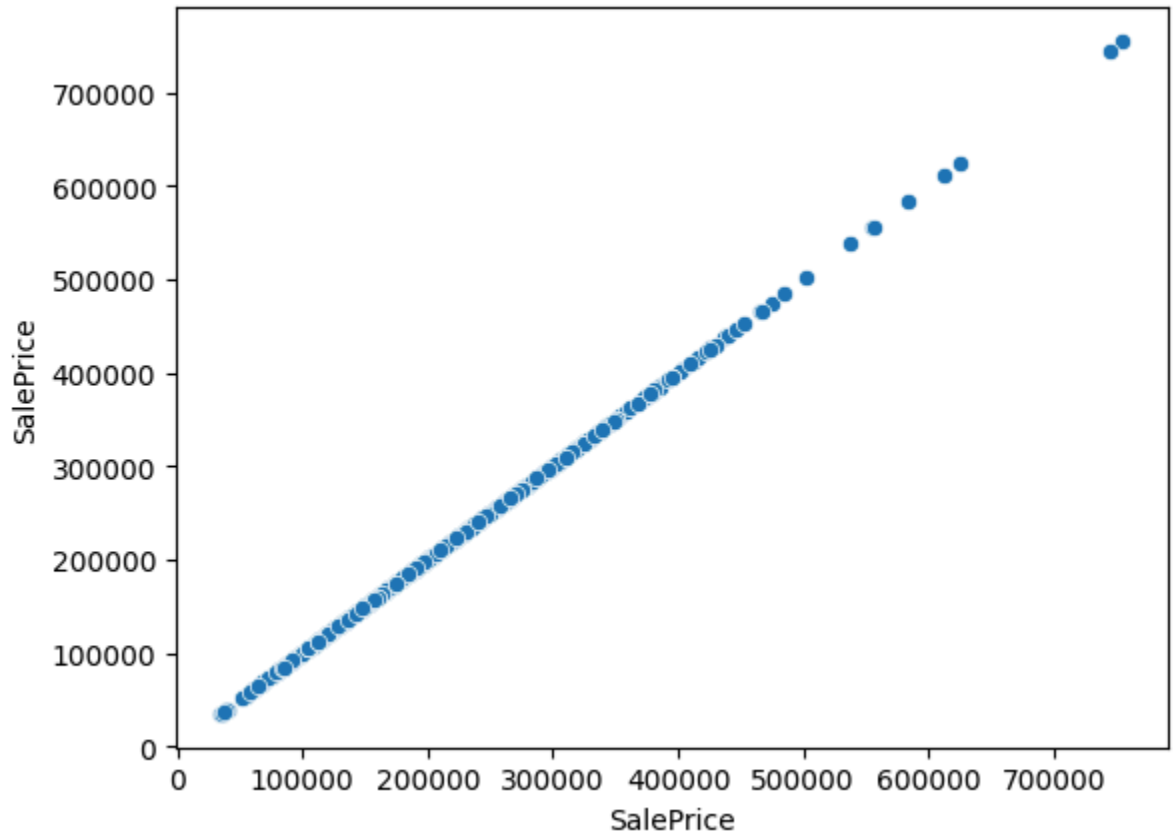
MoSold



YrSold


2010.0
2009.5
2009.0
2008.5
2008.0
2007.5
2007.0
2006.5
2006.0
YrSold
0
100000
200000
300000
400000
500000
600000
700000
SalePrice

SalePrice


700000
600000
500000
400000
300000
200000
100000
0
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
SalePrice

In [321…
```
drop_cols=['MsSubclass','BsmtFinF2','LowQualFinSF','BsmtHalfBath','KitchenAbvG
```

In [322…
```
df[con].corr()['SalePrice'].sort_values(ascending=False)
```

Out[322…

	SalePrice
SalePrice	1.000000
OverallQual	0.790982
GrLivArea	0.708624
GarageCars	0.640409
GarageArea	0.623431
TotalBsmtSF	0.613581
1stFlrSF	0.605852
FullBath	0.560664
TotRmsAbvGrd	0.533723
YearBuilt	0.522897
YearRemodAdd	0.507101
MasVnrArea	0.475241
GarageYrBlt	0.470177
Fireplaces	0.466929
BsmtFinSF1	0.386420
LotFrontage	0.334901
WoodDeckSF	0.324413
2ndFlrSF	0.319334
OpenPorchSF	0.315856
HalfBath	0.284108
LotArea	0.263843
BsmtFullBath	0.227122
BsmtUnfSF	0.214479
BedroomAbvGr	0.168213
ScreenPorch	0.111447
PoolArea	0.092404
MoSold	0.046432
3SsnPorch	0.044584
BsmtFinSF2	-0.011378
BsmtHalfBath	-0.016844
MiscVal	-0.021190

	SalePrice
Id	-0.021917
LowQualFinSF	-0.025606
YrSold	-0.028923
OverallCond	-0.077856
MSSubClass	-0.084284
EnclosedPorch	-0.128578
KitchenAbvGr	-0.135907

dtype: float64

In [322…

In [323…
```
correlations = df[con].corrwith(df['SalePrice']).sort_values(ascending=False)
for col, corr in correlations.items():
    if -0.5 < corr < 0.5 and col != 'SalePrice':
        print(col)
```

```
MasVnrArea
GarageYrBlt
Fireplaces
BsmtFinSF1
LotFrontage
WoodDeckSF
2ndFlrSF
OpenPorchSF
HalfBath
LotArea
BsmtFullBath
BsmtUnfSF
BedroomAbvGr
ScreenPorch
PoolArea
MoSold
3SsnPorch
BsmtFinSF2
BsmtHalfBath
MiscVal
Id
LowQualFinSF
YrSold
OverallCond
MSSubClass
EnclosedPorch
KitchenAbvGr
```

In [324…
```
drop_cols=['MSSubClass','BsmtFinSF2','LowQualFinSF','BsmtHalfBath','KitchenAbv
```

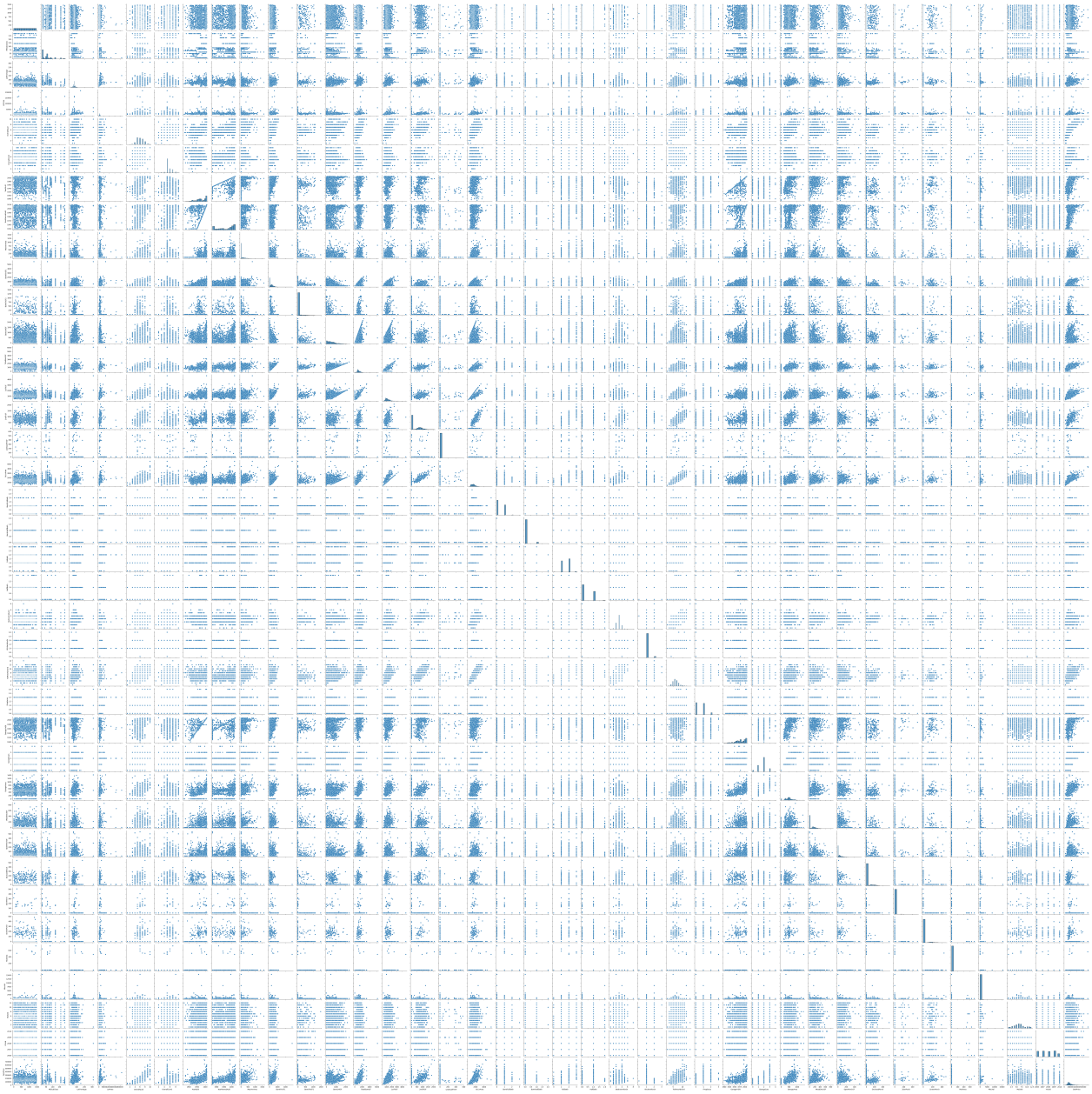
```
drop_cols.extend(['MasVnrArea','GarageYrBlt',
'Fireplaces',
'BsmtFinSF1',
'LotFrontage',
'WoodDeckSF',
'2ndFlrSF',
'OpenPorchSF',
'HalfBath',
'LotArea',
'BsmtFullBath',
'BsmtUnfSF',
'BedroomAbvGr',
'ScreenPorch',
'PoolArea',
'MoSold',
'3SsnPorch',
'BsmtFinSF2',
'BsmtHalfBath',
'MiscVal',
'Id',
'LowQualFinSF',
'YrSold',
'OverallCond',
'MSSubClass',
'EnclosedPorch',
'KitchenAbvGr'])
```

In [325…
```
len(drop_cols)
```

Out[325…
```
38
```

In [326…
```
sns.pairplot(data=df[con])
```

Out[326…
```
<seaborn.axisgrid.PairGrid at 0x7c221c5db200>
```



In [327…
```
drop_cols.append('SalePrice')
```

In [328…
```
X=df.drop(labels=drop_cols,axis=1)
Y=df[['SalePrice']]
```

In [329…
```
X.columns
```

Out[329…
```
Index(['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities',
       'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2',
       'BldgType', 'HouseStyle', 'OverallQual', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'TotalBsmtSF',
       'Heating', 'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF',
       'GrLivArea', 'FullBath', 'KitchenQual', 'TotRmsAbvGrd', 'Functional',
       'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageCars', 'GarageAre
a',
       'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence',
       'MiscFeature', 'SaleType', 'SaleCondition'],
      dtype='object')
```

In [330…
```
Y.columns
```

Out[330…
```
Index(['SalePrice'], dtype='object')
```

In [331…
```
X
```

Out[331…

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	Lar
0	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	
1	RL	Pave	Grvl	Reg	Lvl	AllPub	FR2	
2	RL	Pave	Grvl	IR1	Lvl	AllPub	Inside	
3	RL	Pave	Grvl	IR1	Lvl	AllPub	Corner	
4	RL	Pave	Grvl	IR1	Lvl	AllPub	FR2	
...	...	...	...	...	...	...	...	
1455	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	
1456	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	
1457	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	
1458	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	
1459	RL	Pave	Grvl	Reg	Lvl	AllPub	Inside	

1460 rows × 53 columns

In [332…
```
df['SalePrice'].unique()
```

Out[332… array([208500, 181500, 223500, 140000, 250000, 143000, 307000, 200000,
129900, 118000, 129500, 345000, 144000, 279500, 157000, 132000,
149000, 90000, 159000, 139000, 325300, 139400, 230000, 154000,
256300, 134800, 306000, 207500, 68500, 40000, 149350, 179900,
165500, 277500, 309000, 145000, 153000, 109000, 82000, 160000,
170000, 130250, 141000, 319900, 239686, 249700, 113000, 127000,
177000, 114500, 110000, 385000, 130000, 180500, 172500, 196500,
438780, 124900, 158000, 101000, 202500, 219500, 317000, 180000,
226000, 80000, 225000, 244000, 185000, 144900, 107400, 91000,
135750, 136500, 193500, 153500, 245000, 126500, 168500, 260000,
174000, 164500, 85000, 123600, 109900, 98600, 163500, 133900,
204750, 214000, 94750, 83000, 128950, 205000, 178000, 118964,
198900, 169500, 100000, 115000, 190000, 136900, 383970, 217000,
259500, 176000, 155000, 320000, 163990, 136000, 153900, 181000,
 84500, 128000, 87000, 150000, 150750, 220000, 171000, 231500,
166000, 204000, 125000, 105000, 222500, 122000, 372402, 235000,
 79000, 109500, 269500, 254900, 162500, 412500, 103200, 152000,
127500, 325624, 183500, 228000, 128500, 215000, 239000, 163000,
184000, 243000, 211000, 501837, 200100, 120000, 475000, 173000,
135000, 153337, 286000, 315000, 192000, 148500, 311872, 104000,
274900, 171500, 112000, 143900, 277000, 98000, 186000, 252678,
156000, 161750, 134450, 210000, 107000, 311500, 167240, 204900,
 97000, 386250, 290000, 106000, 192500, 148000, 403000, 94500,
128200, 216500, 89500, 185500, 194500, 318000, 262500, 110500,
241500, 137000, 76500, 276000, 151000, 73000, 175500, 179500,
120500, 266000, 124500, 201000, 415298, 228500, 244600, 179200,
164700, 88000, 153575, 233230, 135900, 131000, 167000, 142500,
175000, 158500, 267000, 149900, 295000, 305900, 82500, 360000,
165600, 119900, 375000, 188500, 270000, 187500, 342643, 354000,
301000, 126175, 242000, 324000, 145250, 214500, 78000, 119000,
284000, 207000, 228950, 377426, 202900, 87500, 140200, 151500,
157500, 437154, 318061, 95000, 105900, 177500, 134000, 280000,
198500, 147000, 165000, 162000, 172400, 134432, 123000, 61000,
340000, 394432, 179000, 187750, 213500, 76000, 240000, 81000,
191000, 426000, 106500, 129000, 67000, 241000, 245500, 164990,
108000, 258000, 168000, 339750, 60000, 222000, 181134, 149500,
126000, 142000, 206300, 275000, 109008, 195400, 85400, 79900,
122500, 212000, 116000, 90350, 555000, 162900, 199900, 119500,
188000, 256000, 161000, 263435, 62383, 188700, 124000, 178740,
146500, 187000, 440000, 251000, 132500, 208900, 380000, 297000,
 89471, 326000, 374000, 164000, 86000, 133000, 172785, 91300,
 34900, 430000, 226700, 289000, 208300, 164900, 202665, 96500,
402861, 265000, 234000, 106250, 184750, 315750, 446261, 200624,
107500, 39300, 111250, 272000, 248000, 213250, 179665, 229000,
263000, 112500, 255500, 121500, 268000, 325000, 316600, 135960,
142600, 224500, 118500, 146000, 131500, 181900, 253293, 369900,
 79500, 185900, 451950, 138000, 319000, 114504, 194201, 217500,
221000, 359100, 313000, 261500, 75500, 137500, 183200, 105500,
314813, 305000, 165150, 139900, 209500, 93000, 264561, 274000,
370878, 143250, 98300, 205950, 350000, 145500, 97500, 197900,
402000, 423000, 230500, 173500, 103600, 257500, 372500, 159434,
285000, 227875, 148800, 392000, 194700, 755000, 335000, 108480,
141500, 89000, 123500, 138500, 196000, 312500, 361919, 213000,
 55000, 302000, 254000, 179540, 52000, 102776, 189000, 130500,

```
159500, 341000, 103000, 236500, 131400,  93500, 239900, 299800,
236000, 265979, 260400, 275500, 158900, 179400, 215200, 337000,
264132, 216837, 538000, 134900, 102000, 395000, 221500, 175900,
187100, 161500, 233000, 107900, 160200, 146800, 269790, 143500,
485000, 582933, 227680, 135500, 159950, 144500,  55993, 157900,
224900, 271000, 224000, 183000, 139500, 232600, 147400, 237000,
139950, 174900, 133500, 189950, 250580, 248900, 169000, 200500,
 66500, 303477, 132250, 328900, 122900, 154500, 118858, 142953,
611657, 125500, 255000, 154300, 173733,  75000,  35311, 238000,
176500, 145900, 169990, 193000, 117500, 184900, 253000, 239799,
244400, 150900, 197500, 172000, 116500, 214900, 178900,  37900,
 99500, 182000, 167500,  85500, 178400, 336000, 159895, 255900,
117000, 395192, 195000, 197000, 348000, 173900, 337500, 121600,
206000, 232000, 136905, 119200, 227000, 203000, 213490, 194000,
287000, 293077, 310000, 119750,  84000, 315500, 262280, 278000,
139600, 556581,  84900, 176485, 200141, 185850, 328000, 167900,
151400,  91500, 138800, 155900,  83500, 252000,  92900, 176432,
274725, 134500, 184100, 133700, 118400, 212900, 163900, 259000,
239500,  94000, 424870, 174500, 116900, 201800, 218000, 235128,
108959, 233170, 245350, 625000, 171900, 154900, 392500, 745000,
186700, 104900, 262000, 219210, 116050, 271900, 229456,  80500,
137900, 367294, 101800, 138887, 265900, 248328, 465000, 186500,
169900, 171750, 294000, 165400, 301500,  99900, 128900, 183900,
378500, 381000, 185750,  68400, 150500, 281000, 333168, 206900,
295493, 111000, 156500,  72500,  52500, 155835, 108500, 283463,
410000, 156932, 144152, 216000, 274300, 466500,  58500, 237500,
377500, 246578, 281213, 137450, 193879, 282922, 257000, 223000,
274970, 182900, 192140, 143750,  64500, 394617, 149700, 149300,
121000, 179600,  92000, 287090, 266500, 142125, 147500])
```

In [333…
```
df['SalePrice']=df['SalePrice'].replace('rotary',0)
```

In [334…
```
df['SalePrice'].unique()
```

Out[334… array([208500, 181500, 223500, 140000, 250000, 143000, 307000, 200000,
129900, 118000, 129500, 345000, 144000, 279500, 157000, 132000,
149000, 90000, 159000, 139000, 325300, 139400, 230000, 154000,
256300, 134800, 306000, 207500, 68500, 40000, 149350, 179900,
165500, 277500, 309000, 145000, 153000, 109000, 82000, 160000,
170000, 130250, 141000, 319900, 239686, 249700, 113000, 127000,
177000, 114500, 110000, 385000, 130000, 180500, 172500, 196500,
438780, 124900, 158000, 101000, 202500, 219500, 317000, 180000,
226000, 80000, 225000, 244000, 185000, 144900, 107400, 91000,
135750, 136500, 193500, 153500, 245000, 126500, 168500, 260000,
174000, 164500, 85000, 123600, 109900, 98600, 163500, 133900,
204750, 214000, 94750, 83000, 128950, 205000, 178000, 118964,
198900, 169500, 100000, 115000, 190000, 136900, 383970, 217000,
259500, 176000, 155000, 320000, 163990, 136000, 153900, 181000,
 84500, 128000, 87000, 150000, 150750, 220000, 171000, 231500,
166000, 204000, 125000, 105000, 222500, 122000, 372402, 235000,
 79000, 109500, 269500, 254900, 162500, 412500, 103200, 152000,
127500, 325624, 183500, 228000, 128500, 215000, 239000, 163000,
184000, 243000, 211000, 501837, 200100, 120000, 475000, 173000,
135000, 153337, 286000, 315000, 192000, 148500, 311872, 104000,
274900, 171500, 112000, 143900, 277000, 98000, 186000, 252678,
156000, 161750, 134450, 210000, 107000, 311500, 167240, 204900,
 97000, 386250, 290000, 106000, 192500, 148000, 403000, 94500,
128200, 216500, 89500, 185500, 194500, 318000, 262500, 110500,
241500, 137000, 76500, 276000, 151000, 73000, 175500, 179500,
120500, 266000, 124500, 201000, 415298, 228500, 244600, 179200,
164700, 88000, 153575, 233230, 135900, 131000, 167000, 142500,
175000, 158500, 267000, 149900, 295000, 305900, 82500, 360000,
165600, 119900, 375000, 188500, 270000, 187500, 342643, 354000,
301000, 126175, 242000, 324000, 145250, 214500, 78000, 119000,
284000, 207000, 228950, 377426, 202900, 87500, 140200, 151500,
157500, 437154, 318061, 95000, 105900, 177500, 134000, 280000,
198500, 147000, 165000, 162000, 172400, 134432, 123000, 61000,
340000, 394432, 179000, 187750, 213500, 76000, 240000, 81000,
191000, 426000, 106500, 129000, 67000, 241000, 245500, 164990,
108000, 258000, 168000, 339750, 60000, 222000, 181134, 149500,
126000, 142000, 206300, 275000, 109008, 195400, 85400, 79900,
122500, 212000, 116000, 90350, 555000, 162900, 199900, 119500,
188000, 256000, 161000, 263435, 62383, 188700, 124000, 178740,
146500, 187000, 440000, 251000, 132500, 208900, 380000, 297000,
 89471, 326000, 374000, 164000, 86000, 133000, 172785, 91300,
 34900, 430000, 226700, 289000, 208300, 164900, 202665, 96500,
402861, 265000, 234000, 106250, 184750, 315750, 446261, 200624,
107500, 39300, 111250, 272000, 248000, 213250, 179665, 229000,
263000, 112500, 255500, 121500, 268000, 325000, 316600, 135960,
142600, 224500, 118500, 146000, 131500, 181900, 253293, 369900,
 79500, 185900, 451950, 138000, 319000, 114504, 194201, 217500,
221000, 359100, 313000, 261500, 75500, 137500, 183200, 105500,
314813, 305000, 165150, 139900, 209500, 93000, 264561, 274000,
370878, 143250, 98300, 205950, 350000, 145500, 97500, 197900,
402000, 423000, 230500, 173500, 103600, 257500, 372500, 159434,
285000, 227875, 148800, 392000, 194700, 755000, 335000, 108480,
141500, 89000, 123500, 138500, 196000, 312500, 361919, 213000,
 55000, 302000, 254000, 179540, 52000, 102776, 189000, 130500,

```
       159500, 341000, 103000, 236500, 131400,  93500, 239900, 299800,
       236000, 265979, 260400, 275500, 158900, 179400, 215200, 337000,
       264132, 216837, 538000, 134900, 102000, 395000, 221500, 175900,
       187100, 161500, 233000, 107900, 160200, 146800, 269790, 143500,
       485000, 582933, 227680, 135500, 159950, 144500,  55993, 157900,
       224900, 271000, 224000, 183000, 139500, 232600, 147400, 237000,
       139950, 174900, 133500, 189950, 250580, 248900, 169000, 200500,
        66500, 303477, 132250, 328900, 122900, 154500, 118858, 142953,
       611657, 125500, 255000, 154300, 173733,  75000,  35311, 238000,
       176500, 145900, 169990, 193000, 117500, 184900, 253000, 239799,
       244400, 150900, 197500, 172000, 116500, 214900, 178900,  37900,
        99500, 182000, 167500,  85500, 178400, 336000, 159895, 255900,
       117000, 395192, 195000, 197000, 348000, 173900, 337500, 121600,
       206000, 232000, 136905, 119200, 227000, 203000, 213490, 194000,
       287000, 293077, 310000, 119750,  84000, 315500, 262280, 278000,
       139600, 556581,  84900, 176485, 200141, 185850, 328000, 167900,
       151400,  91500, 138800, 155900,  83500, 252000,  92900, 176432,
       274725, 134500, 184100, 133700, 118400, 212900, 163900, 259000,
       239500,  94000, 424870, 174500, 116900, 201800, 218000, 235128,
       108959, 233170, 245350, 625000, 171900, 154900, 392500, 745000,
       186700, 104900, 262000, 219210, 116050, 271900, 229456,  80500,
       137900, 367294, 101800, 138887, 265900, 248328, 465000, 186500,
       169900, 171750, 294000, 165400, 301500,  99900, 128900, 183900,
       378500, 381000, 185750,  68400, 150500, 281000, 333168, 206900,
       295493, 111000, 156500,  72500,  52500, 155835, 108500, 283463,
       410000, 156932, 144152, 216000, 274300, 466500,  58500, 237500,
       377500, 246578, 281213, 137450, 193879, 282922, 257000, 223000,
       274970, 182900, 192140, 143750,  64500, 394617, 149700, 149300,
       121000, 179600,  92000, 287090, 266500, 142125, 147500])
```

In [335…
```
df['SalePrice']=df['SalePrice'].astype(int)
```

In [336…
```
df['SalePrice'].unique()
```

Out[336… array([208500, 181500, 223500, 140000, 250000, 143000, 307000, 200000,
129900, 118000, 129500, 345000, 144000, 279500, 157000, 132000,
149000, 90000, 159000, 139000, 325300, 139400, 230000, 154000,
256300, 134800, 306000, 207500, 68500, 40000, 149350, 179900,
165500, 277500, 309000, 145000, 153000, 109000, 82000, 160000,
170000, 130250, 141000, 319900, 239686, 249700, 113000, 127000,
177000, 114500, 110000, 385000, 130000, 180500, 172500, 196500,
438780, 124900, 158000, 101000, 202500, 219500, 317000, 180000,
226000, 80000, 225000, 244000, 185000, 144900, 107400, 91000,
135750, 136500, 193500, 153500, 245000, 126500, 168500, 260000,
174000, 164500, 85000, 123600, 109900, 98600, 163500, 133900,
204750, 214000, 94750, 83000, 128950, 205000, 178000, 118964,
198900, 169500, 100000, 115000, 190000, 136900, 383970, 217000,
259500, 176000, 155000, 320000, 163990, 136000, 153900, 181000,
 84500, 128000, 87000, 150000, 150750, 220000, 171000, 231500,
166000, 204000, 125000, 105000, 222500, 122000, 372402, 235000,
 79000, 109500, 269500, 254900, 162500, 412500, 103200, 152000,
127500, 325624, 183500, 228000, 128500, 215000, 239000, 163000,
184000, 243000, 211000, 501837, 200100, 120000, 475000, 173000,
135000, 153337, 286000, 315000, 192000, 148500, 311872, 104000,
274900, 171500, 112000, 143900, 277000, 98000, 186000, 252678,
156000, 161750, 134450, 210000, 107000, 311500, 167240, 204900,
 97000, 386250, 290000, 106000, 192500, 148000, 403000, 94500,
128200, 216500, 89500, 185500, 194500, 318000, 262500, 110500,
241500, 137000, 76500, 276000, 151000, 73000, 175500, 179500,
120500, 266000, 124500, 201000, 415298, 228500, 244600, 179200,
164700, 88000, 153575, 233230, 135900, 131000, 167000, 142500,
175000, 158500, 267000, 149900, 295000, 305900, 82500, 360000,
165600, 119900, 375000, 188500, 270000, 187500, 342643, 354000,
301000, 126175, 242000, 324000, 145250, 214500, 78000, 119000,
284000, 207000, 228950, 377426, 202900, 87500, 140200, 151500,
157500, 437154, 318061, 95000, 105900, 177500, 134000, 280000,
198500, 147000, 165000, 162000, 172400, 134432, 123000, 61000,
340000, 394432, 179000, 187750, 213500, 76000, 240000, 81000,
191000, 426000, 106500, 129000, 67000, 241000, 245500, 164990,
108000, 258000, 168000, 339750, 60000, 222000, 181134, 149500,
126000, 142000, 206300, 275000, 109008, 195400, 85400, 79900,
122500, 212000, 116000, 90350, 555000, 162900, 199900, 119500,
188000, 256000, 161000, 263435, 62383, 188700, 124000, 178740,
146500, 187000, 440000, 251000, 132500, 208900, 380000, 297000,
 89471, 326000, 374000, 164000, 86000, 133000, 172785, 91300,
 34900, 430000, 226700, 289000, 208300, 164900, 202665, 96500,
402861, 265000, 234000, 106250, 184750, 315750, 446261, 200624,
107500, 39300, 111250, 272000, 248000, 213250, 179665, 229000,
263000, 112500, 255500, 121500, 268000, 325000, 316600, 135960,
142600, 224500, 118500, 146000, 131500, 181900, 253293, 369900,
 79500, 185900, 451950, 138000, 319000, 114504, 194201, 217500,
221000, 359100, 313000, 261500, 75500, 137500, 183200, 105500,
314813, 305000, 165150, 139900, 209500, 93000, 264561, 274000,
370878, 143250, 98300, 205950, 350000, 145500, 97500, 197900,
402000, 423000, 230500, 173500, 103600, 257500, 372500, 159434,
285000, 227875, 148800, 392000, 194700, 755000, 335000, 108480,
141500, 89000, 123500, 138500, 196000, 312500, 361919, 213000,
 55000, 302000, 254000, 179540, 52000, 102776, 189000, 130500,

```
       159500, 341000, 103000, 236500, 131400,  93500, 239900, 299800,
       236000, 265979, 260400, 275500, 158900, 179400, 215200, 337000,
       264132, 216837, 538000, 134900, 102000, 395000, 221500, 175900,
       187100, 161500, 233000, 107900, 160200, 146800, 269790, 143500,
       485000, 582933, 227680, 135500, 159950, 144500,  55993, 157900,
       224900, 271000, 224000, 183000, 139500, 232600, 147400, 237000,
       139950, 174900, 133500, 189950, 250580, 248900, 169000, 200500,
        66500, 303477, 132250, 328900, 122900, 154500, 118858, 142953,
       611657, 125500, 255000, 154300, 173733,  75000,  35311, 238000,
       176500, 145900, 169990, 193000, 117500, 184900, 253000, 239799,
       244400, 150900, 197500, 172000, 116500, 214900, 178900,  37900,
        99500, 182000, 167500,  85500, 178400, 336000, 159895, 255900,
       117000, 395192, 195000, 197000, 348000, 173900, 337500, 121600,
       206000, 232000, 136905, 119200, 227000, 203000, 213490, 194000,
       287000, 293077, 310000, 119750,  84000, 315500, 262280, 278000,
       139600, 556581,  84900, 176485, 200141, 185850, 328000, 167900,
       151400,  91500, 138800, 155900,  83500, 252000,  92900, 176432,
       274725, 134500, 184100, 133700, 118400, 212900, 163900, 259000,
       239500,  94000, 424870, 174500, 116900, 201800, 218000, 235128,
       108959, 233170, 245350, 625000, 171900, 154900, 392500, 745000,
       186700, 104900, 262000, 219210, 116050, 271900, 229456,  80500,
       137900, 367294, 101800, 138887, 265900, 248328, 465000, 186500,
       169900, 171750, 294000, 165400, 301500,  99900, 128900, 183900,
       378500, 381000, 185750,  68400, 150500, 281000, 333168, 206900,
       295493, 111000, 156500,  72500,  52500, 155835, 108500, 283463,
       410000, 156932, 144152, 216000, 274300, 466500,  58500, 237500,
       377500, 246578, 281213, 137450, 193879, 282922, 257000, 223000,
       274970, 182900, 192140, 143750,  64500, 394617, 149700, 149300,
       121000, 179600,  92000, 287090, 266500, 142125, 147500])
```

In [337…

```
for i in df.columns:
  print(i)
  print(df[i].unique())
  print('****************************************8')
```

```
Id
[   1    2    3 ... 1458 1459 1460]
****************************************8
MSSubClass
[ 60  20  70  50 190  45  90 120  30  85  80 160  75 180  40]
****************************************8
MSZoning
['RL' 'RM' 'C (all)' 'FV' 'RH']
****************************************8
LotFrontage
[ 65.          80.          68.          60.          84.
  85.          75.          70.04995837  51.          50.
  70.          91.          72.          66.         101.
  57.          44.         110.          98.          47.
 108.         112.          74.         115.          61.
  48.          33.          52.         100.          24.
  89.          63.          76.          81.          95.
  69.          21.          32.          78.         121.
 122.          40.         105.          73.          77.
  64.          94.          34.          90.          55.
  88.          82.          71.         120.         107.
  92.         134.          62.          86.         141.
  97.          54.          41.          79.         174.
  99.          67.          83.          43.         103.
  93.          30.         129.         140.          35.
  37.         118.          87.         116.         150.
 111.          49.          96.          59.          36.
  56.         102.          58.          38.         109.
 130.          53.         137.          45.         106.
 104.          42.          39.         144.         114.
 128.         149.         313.         168.         182.
 138.         160.         152.         124.         153.
  46.        ]
****************************************8
LotArea
[ 8450  9600 11250 ... 17217 13175  9717]
****************************************8
Street
['Pave' 'Grvl']
****************************************8
Alley
['Grvl' 'Pave']
****************************************8
LotShape
['Reg' 'IR1' 'IR2' 'IR3']
****************************************8
LandContour
['Lvl' 'Bnk' 'Low' 'HLS']
****************************************8
Utilities
['AllPub' 'NoSeWa']
****************************************8
LotConfig
['Inside' 'FR2' 'Corner' 'CulDSac' 'FR3']
```

```
****************************************8
LandSlope
['Gtl' 'Mod' 'Sev']
****************************************8
Neighborhood
['CollgCr' 'Veenker' 'Crawfor' 'NoRidge' 'Mitchel' 'Somerst' 'NWAmes'
 'OldTown' 'BrkSide' 'Sawyer' 'NridgHt' 'NAmes' 'SawyerW' 'IDOTRR'
 'MeadowV' 'Edwards' 'Timber' 'Gilbert' 'StoneBr' 'ClearCr' 'NPkVill'
 'Blmngtn' 'BrDale' 'SWISU' 'Blueste']
****************************************8
Condition1
['Norm' 'Feedr' 'PosN' 'Artery' 'RRAe' 'RRNn' 'RRAn' 'PosA' 'RRNe']
****************************************8
Condition2
['Norm' 'Artery' 'RRNn' 'Feedr' 'PosN' 'PosA' 'RRAn' 'RRAe']
****************************************8
BldgType
['1Fam' '2fmCon' 'Duplex' 'TwnhsE' 'Twnhs']
****************************************8
HouseStyle
['2Story' '1Story' '1.5Fin' '1.5Unf' 'SFoyer' 'SLvl' '2.5Unf' '2.5Fin']
****************************************8
OverallQual
[ 7  6  8  5  9  4 10  3  1  2]
****************************************8
OverallCond
[5 8 6 7 4 2 3 9 1]
****************************************8
YearBuilt
[2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 1965 2005 1962 2006
 1960 1929 1970 1967 1958 1930 2002 1968 2007 1951 1957 1927 1920 1966
 1959 1994 1954 1953 1955 1983 1975 1997 1934 1963 1981 1964 1999 1972
 1921 1945 1982 1998 1956 1948 1910 1995 1991 2009 1950 1961 1977 1985
 1979 1885 1919 1990 1969 1935 1988 1971 1952 1936 1923 1924 1984 1926
 1940 1941 1987 1986 2008 1908 1892 1916 1932 1918 1912 1947 1925 1900
 1980 1989 1992 1949 1880 1928 1978 1922 1996 2010 1946 1913 1937 1942
 1938 1974 1893 1914 1906 1890 1898 1904 1882 1875 1911 1917 1872 1905]
****************************************8
YearRemodAdd
[2003 1976 2002 1970 2000 1995 2005 1973 1950 1965 2006 1962 2007 1960
 2001 1967 2004 2008 1997 1959 1990 1955 1983 1980 1966 1963 1987 1964
 1972 1996 1998 1989 1953 1956 1968 1981 1992 2009 1982 1961 1993 1999
 1985 1979 1977 1969 1958 1991 1971 1952 1975 2010 1984 1986 1994 1988
 1954 1957 1951 1978 1974]
****************************************8
RoofStyle
['Gable' 'Hip' 'Gambrel' 'Mansard' 'Flat' 'Shed']
****************************************8
RoofMatl
['CompShg' 'WdShngl' 'Metal' 'WdShake' 'Membran' 'Tar&Grv' 'Roll'
 'ClyTile']
****************************************8
Exterior1st
['VinylSd' 'MetalSd' 'Wd Sdng' 'HdBoard' 'BrkFace' 'WdShing' 'CemntBd'
```

```
 'Plywood' 'AsbShng' 'Stucco' 'BrkComm' 'AsphShn' 'Stone' 'ImStucc'
 'CBlock']
****************************************8
Exterior2nd
['VinylSd' 'MetalSd' 'Wd Shng' 'HdBoard' 'Plywood' 'Wd Sdng' 'CmentBd'
 'BrkFace' 'Stucco' 'AsbShng' 'Brk Cmn' 'ImStucc' 'AsphShn' 'Stone'
 'Other' 'CBlock']
****************************************8
MasVnrType
['BrkFace' 'Stone' 'BrkCmn']
****************************************8
MasVnrArea
[1.96000000e+02 0.00000000e+00 1.62000000e+02 3.50000000e+02
 1.86000000e+02 2.40000000e+02 2.86000000e+02 3.06000000e+02
 2.12000000e+02 1.80000000e+02 3.80000000e+02 2.81000000e+02
 6.40000000e+02 2.00000000e+02 2.46000000e+02 1.32000000e+02
 6.50000000e+02 1.01000000e+02 4.12000000e+02 2.72000000e+02
 4.56000000e+02 1.03100000e+03 1.78000000e+02 5.73000000e+02
 3.44000000e+02 2.87000000e+02 1.67000000e+02 1.11500000e+03
 4.00000000e+01 1.04000000e+02 5.76000000e+02 4.43000000e+02
 4.68000000e+02 6.60000000e+01 2.20000000e+01 2.84000000e+02
 7.60000000e+01 2.03000000e+02 6.80000000e+01 1.83000000e+02
 4.80000000e+01 2.80000000e+01 3.36000000e+02 6.00000000e+02
 7.68000000e+02 4.80000000e+02 2.20000000e+02 1.84000000e+02
 1.12900000e+03 1.16000000e+02 1.35000000e+02 2.66000000e+02
 8.50000000e+01 3.09000000e+02 1.36000000e+02 2.88000000e+02
 7.00000000e+01 3.20000000e+02 5.00000000e+01 1.20000000e+02
 4.36000000e+02 2.52000000e+02 8.40000000e+01 6.64000000e+02
 2.26000000e+02 3.00000000e+02 6.53000000e+02 1.12000000e+02
 4.91000000e+02 2.68000000e+02 7.48000000e+02 9.80000000e+01
 2.75000000e+02 1.38000000e+02 2.05000000e+02 2.62000000e+02
 1.28000000e+02 2.60000000e+02 1.53000000e+02 6.40000000e+01
 3.12000000e+02 1.60000000e+01 9.22000000e+02 1.42000000e+02
 2.90000000e+02 1.27000000e+02 5.06000000e+02 2.97000000e+02
 1.03685262e+02 6.04000000e+02 2.54000000e+02 3.60000000e+01
 1.02000000e+02 4.72000000e+02 4.81000000e+02 1.08000000e+02
 3.02000000e+02 1.72000000e+02 3.99000000e+02 2.70000000e+02
 4.60000000e+01 2.10000000e+02 1.74000000e+02 3.48000000e+02
 3.15000000e+02 2.99000000e+02 3.40000000e+02 1.66000000e+02
 7.20000000e+01 3.10000000e+01 3.40000000e+01 2.38000000e+02
 1.60000000e+03 3.65000000e+02 5.60000000e+01 1.50000000e+02
 2.78000000e+02 2.56000000e+02 2.25000000e+02 3.70000000e+02
 3.88000000e+02 1.75000000e+02 2.96000000e+02 1.46000000e+02
 1.13000000e+02 1.76000000e+02 6.16000000e+02 3.00000000e+01
 1.06000000e+02 8.70000000e+02 3.62000000e+02 5.30000000e+02
 5.00000000e+02 5.10000000e+02 2.47000000e+02 3.05000000e+02
 2.55000000e+02 1.25000000e+02 1.00000000e+02 4.32000000e+02
 1.26000000e+02 4.73000000e+02 7.40000000e+01 1.45000000e+02
 2.32000000e+02 3.76000000e+02 4.20000000e+01 1.61000000e+02
 1.10000000e+02 1.80000000e+01 2.24000000e+02 2.48000000e+02
 8.00000000e+01 3.04000000e+02 2.15000000e+02 7.72000000e+02
 4.35000000e+02 3.78000000e+02 5.62000000e+02 1.68000000e+02
 8.90000000e+01 2.85000000e+02 3.60000000e+02 9.40000000e+01
 3.33000000e+02 9.21000000e+02 7.62000000e+02 5.94000000e+02
```

```
 2.19000000e+02 1.88000000e+02 4.79000000e+02 5.84000000e+02
 1.82000000e+02 2.50000000e+02 2.92000000e+02 2.45000000e+02
 2.07000000e+02 8.20000000e+01 9.70000000e+01 3.35000000e+02
 2.08000000e+02 4.20000000e+02 1.70000000e+02 4.59000000e+02
 2.80000000e+02 9.90000000e+01 1.92000000e+02 2.04000000e+02
 2.33000000e+02 1.56000000e+02 4.52000000e+02 5.13000000e+02
 2.61000000e+02 1.64000000e+02 2.59000000e+02 2.09000000e+02
 2.63000000e+02 2.16000000e+02 3.51000000e+02 6.60000000e+02
 3.81000000e+02 5.40000000e+01 5.28000000e+02 2.58000000e+02
 4.64000000e+02 5.70000000e+01 1.47000000e+02 1.17000000e+03
 2.93000000e+02 6.30000000e+02 4.66000000e+02 1.09000000e+02
 4.10000000e+01 1.60000000e+02 2.89000000e+02 6.51000000e+02
 1.69000000e+02 9.50000000e+01 4.42000000e+02 2.02000000e+02
 3.38000000e+02 8.94000000e+02 3.28000000e+02 6.73000000e+02
 6.03000000e+02 1.00000000e+00 3.75000000e+02 9.00000000e+01
 3.80000000e+01 1.57000000e+02 1.10000000e+01 1.40000000e+02
 1.30000000e+02 1.48000000e+02 8.60000000e+02 4.24000000e+02
 1.04700000e+03 2.43000000e+02 8.16000000e+02 3.87000000e+02
 2.23000000e+02 1.58000000e+02 1.37000000e+02 1.15000000e+02
 1.89000000e+02 2.74000000e+02 1.17000000e+02 6.00000000e+01
 1.22000000e+02 9.20000000e+01 4.15000000e+02 7.60000000e+02
 2.70000000e+01 7.50000000e+01 3.61000000e+02 1.05000000e+02
 3.42000000e+02 2.98000000e+02 5.41000000e+02 2.36000000e+02
 1.44000000e+02 4.23000000e+02 4.40000000e+01 1.51000000e+02
 9.75000000e+02 4.50000000e+02 2.30000000e+02 5.71000000e+02
 2.40000000e+01 5.30000000e+01 2.06000000e+02 1.40000000e+01
 3.24000000e+02 2.95000000e+02 3.96000000e+02 6.70000000e+01
 1.54000000e+02 4.25000000e+02 4.50000000e+01 1.37800000e+03
 3.37000000e+02 1.49000000e+02 1.43000000e+02 5.10000000e+01
 1.71000000e+02 2.34000000e+02 6.30000000e+01 7.66000000e+02
 3.20000000e+01 8.10000000e+01 1.63000000e+02 5.54000000e+02
 2.18000000e+02 6.32000000e+02 1.14000000e+02 5.67000000e+02
 3.59000000e+02 4.51000000e+02 6.21000000e+02 7.88000000e+02
 8.60000000e+01 7.96000000e+02 3.91000000e+02 2.28000000e+02
 8.80000000e+01 1.65000000e+02 4.28000000e+02 4.10000000e+02
 5.64000000e+02 3.68000000e+02 3.18000000e+02 5.79000000e+02
 6.50000000e+01 7.05000000e+02 4.08000000e+02 2.44000000e+02
 1.23000000e+02 3.66000000e+02 7.31000000e+02 4.48000000e+02
 2.94000000e+02 3.10000000e+02 2.37000000e+02 4.26000000e+02
 9.60000000e+01 4.38000000e+02 1.94000000e+02 1.19000000e+02]
****************************************8
ExterQual
['Gd' 'TA' 'Ex' 'Fa']
****************************************8
ExterCond
['TA' 'Gd' 'Fa' 'Po' 'Ex']
****************************************8
Foundation
['PConc' 'CBlock' 'BrkTil' 'Wood' 'Slab' 'Stone']
****************************************8
BsmtQual
['Gd' 'TA' 'Ex' 'Fa']
****************************************8
BsmtCond
```

```
['TA' 'Gd' 'Fa' 'Po']
****************************************8
BsmtExposure
['No' 'Gd' 'Mn' 'Av']
****************************************8
BsmtFinType1
['GLQ' 'ALQ' 'Unf' 'Rec' 'BLQ' 'LwQ']
****************************************8
BsmtFinSF1
[ 706  978  486  216  655  732 1369  859    0  851  906  998  737  733
  578  646  504  840  188  234 1218 1277 1018 1153 1213  731  643  967
  747  280  179  456 1351   24  763  182  104 1810  384  490  649  632
  941  739  912 1013  603 1880  565  320  462  228  336  448 1201   33
  588  600  713 1046  648  310 1162  520  108  569 1200  224  705  444
  250  984   35  774  419  170 1470  938  570  300  120  116  512  567
  445  695  405 1005  668  821  432 1300  507  679 1332  209  680  716
 1400  416  429  222   57  660 1016  370  351  379 1288  360  639  495
  288 1398  477  831 1904  436  352  611 1086  297  626  560  390  566
 1126 1036 1088  641  617  662  312 1065  787  468   36  822  378  946
  341   16  550  524   56  321  842  689  625  358  402   94 1078  329
  929  697 1573  270  922  503 1334  361  672  506  714  403  751  226
  620  546  392  421  905  904  430  614  450  210  292  795 1285  819
  420  841  281  894 1464  700  262 1274  518 1236  425  692  987  970
   28  256 1619   40  846 1124  720  828 1249  810  213  585  129  498
 1270  573 1410 1082  236  388  334  874  956  773  399  162  712  609
  371  540   72  623  428  350  298 1445  218  985  631 1280  241  690
  266  777  812  786 1116  789 1056   50 1128  775 1309 1246  986  616
 1518  664  387  471  385  365 1767  133  642  247  331  742 1606  916
  185  544  553  326  778  386  426  368  459 1350 1196  630  994  168
 1261 1567  299  897  607  836  515  374 1231  111  356  400  698 1247
  257  380   27  141  991  650  521 1436 2260  719  377 1330  348 1219
  783  969  673 1358 1260  144  584  554 1002  619  180  559  308  866
  895  637  604 1302 1071  290  728    2 1441  943  231  414  349  442
  328  594  816 1460 1324 1338  685 1422 1283   81  454  903  605  990
  206  150  457   48  871   41  674  624  480 1154  738  493 1121  282
  500  131 1696  806 1361  920 1721  187 1138  988  193  551  767 1186
  892  311  827  543 1003 1059  239  945   20 1455  965  980  863  533
 1084 1173  523 1148  191 1234  375  808  724  152 1180  252  832  575
  919  439  381  438  549  612 1163  437  394 1416  422  762  975 1097
  251  686  656  568  539  862  197  516  663  608 1636  784  249 1040
  483  196  572  338  330  156 1390  513  460  659  364  564  306  505
  932  750   64  633 1170  899  902 1238  528 1024 1064  285 2188  465
  322  860  599  354   63  223  301  443  489  284  294  814  165  552
  833  464  936  772 1440  748  982  398  562  484  417  699  696  896
  556 1106  651  867  854 1646 1074  536 1172  915  595 1237  273  684
  324 1165  138 1513  317 1012 1022  509  900 1085 1104  240  383  644
  397  740  837  220  586  535  410   75  824  592 1039  510  423  661
  248  704  412 1032  219  708  415 1004  353  702  369  622  212  645
  852 1150 1258  275  176  296  538 1157  492 1198 1387  522  658 1216
 1480 2096 1159  440 1456  883  547  788  485  340 1220  427  344  756
 1540  666  803 1000  885 1386  319  534  125 1314  602  192  593  804
 1053  532 1158 1014  194  167  776 5644  694 1572  746 1406  925  482
  189  765   80 1443  259  735  734 1447  548  315 1282  408  309  203
  865  204  790 1320  769 1070  264  759 1373  976  781   25 1110  404
```

```
  580  678  958 1336 1079   49  830]
****************************************8
BsmtFinType2
['Unf' 'BLQ' 'ALQ' 'Rec' 'LwQ' 'GLQ']
****************************************8
BsmtFinSF2
[   0   32  668  486   93  491  506  712  362   41  169  869  150  670
   28 1080  181  768  215  374  208  441  184  279  306  180  580  690
  692  228  125 1063  620  175  820 1474  264  479  147  232  380  544
  294  258  121  391  531  344  539  713  210  311 1120  165  532   96
  495  174 1127  139  202  645  123  551  219  606  612  480  182  132
  336  468  287   35  499  723  119   40  117  239   80  472   64 1057
  127  630  128  377  764  345 1085  435  823  500  290  324  634  411
  841 1061  466  396  354  149  193  273  465  400  682  557  230  106
  791  240  547  469  177  108  600  492  211  168 1031  438  375  144
   81  906  608  276  661   68  173  972  105  420  546  334  352  872
  110  627  163 1029]
****************************************8
BsmtUnfSF
[ 150  284  434  540  490   64  317  216  952  140  134  177  175 1494
  520  832  426    0  468  525 1158  637 1777  200  204 1566  180  486
  207  649 1228 1234  380  408 1117 1097   84  326  445  383  167  465
 1296   83 1632  736  192  612  816   32  935  321  860 1410  148  217
  530 1346  576  318 1143 1035  440  747  701  343  280  404  840  724
  295 1768  448   36 1530 1065  384 1288  684 1013  402  635  163  168
  176  370  350  381  410  741 1226 1053  641  516  793 1139  550  905
  104  310  252 1125  203  728  732  510  899 1362   30  958  556  413
  479  297  658  262  891 1304  519 1907  336  107  432  403  811  396
  970  506  884  400  896  253  409   93 1200  572  774  769 1335  340
  882  779  112  470  294 1686  360  441  354  700  725  320  554  312
  968  504 1107  577  660   99  871  474  289  600  755  625 1121  276
  186 1424 1140  375   92  305 1176   78  274  311  710  686  457 1232
 1498 1010  160 2336  630  638  162   70 1357 1194  773  483  235  125
 1390  594 1694  488  357  626  916 1020 1367  798  452  392  975  361
  270  602 1482  680  606   88  342  212 1095   96  628 1560  744 2121
  768  386 1468 1145  244  698 1079  570  476  131  184  143 1092  324
 1541 1470  536  319  599  622  179  292  286   80  712  291  153 1088
 1249  166  906  604  100  818  844  596  210 1603  115  103  673  726
  995  967  721 1656  972  460  208  191  438 1869  371  624  552  322
  598  268  130  484  785  733  953  847  333 1580  411  982  808 1293
  939  784  595  229  114  522  735  405  117  961 1286  672 1141  806
  165 1064 1063  245 1276  892 1008  499 1316  463  242  444  281   35
  356  988  580  651  619  544  387  901  926  135  648   75  788 1307
 1078 1258  273 1436  557  930  780  813  878  122  248  588  524  288
  389  424 1375 1626  406  298 2153  417  739  225  611  237  290  264
  238  363  190 1969  697  414  316  466  420  254  960  397 1191  548
   50  178 1368  169  748  689 1264  467  605 1257  551  678  707  880
  378  223  578  969  379  765  149  912  620 1709  132  993  197 1374
   90  195  706 1163  367 1122 1515   55 1497  450  846   23  390  861
  285 1050  331 2042 1237  113  742  924  512  119  314  308  293  537
  126  427  309  914  173 1774  823  485 1116  978  636  564  108 1184
  796  366  300  542  645  664  756  247  776  849 1392   38 1406  111
  545  121 2046  161  261  567 1195  874 1342  151  989 1073  927  219
  224  526 1164  761  461  876  859  171  718  138  941  464  250   72
```

```
  508 1584  415   82  948  893  864 1349   76  487  652 1240  801  279
 1030  348  234 1198  740   89  586  323 1836  480  456 1935  338 1594
  102  374 1413  491 1129  255 1496  650 1926  154  999 1734  124 1417
   15  834 1649  936  778 1489  442 1434  352  458 1221 1099  416 1800
  227  907  528  189 1273  563  372  702 1090  435  198 1372  174 1638
  894  299  105  676 1120  431  218  110  795 1098 1043  481  666  142
  447  783 1670  277  412  794  239  662 1072  717  546  430  422  188
  266 1181 1753  964 1450 1905 1480  772 1032  220  187   29  495  640
  193  196  720  918 1428   77 1266 1128  692  770  750 1442 1007  501
  691 1550 1680 1330 1710  746  814  515  571  359  355  301  668  920
 1055 1420 1752  304 1302  833  133  549  705  722  799  462  429  810
  155  170  230 1459 1082  758 1290 1074  251  172  868  797  365  418
  730  533  671 1012 1528 1005 1373  500  762  752  399 1042   40   26
  932  278  459  568 1502  543  574  977  449  983  731  120  538  831
  994  341  879  815 1212  866 1630  328  141  364 1380   81  303  940
  764 1048  334 1689  690  792  585  473  246 1045 1405  201   14  841
 1104  241  925 2002   74  661  708 1152  256  804  812 1085  344  425
 1616  976  496  349  971 1393 1622 1352 1795 1017 1588  428  803  693
  858 1284 1203 1652   39  539 1217  257  715  616  240  315 1351 1026
 1571  156   61   95  482 1094   60  862  221  791  398  777  503  734
  709 1252  656 1319 1422  560 1573  589  877  136]
****************************************8
TotalBsmtSF
[ 856 1262  920  756 1145  796 1686 1107  952  991 1040 1175  912 1494
 1253  832 1004    0 1114 1029 1158  637 1777 1060 1566  900 1704 1484
  520  649 1228 1234 1398 1561 1117 1097 1297 1057 1088 1350  840  938
 1150 1752 1434 1656  736  955  794  816 1842  384 1425  970  860 1410
  780  530 1370  576 1143 1947 1453  747 1304 2223  845 1086  462  672
 1768  440  896 1237 1563 1065 1288  684  612 1013  990 1235  876 1214
  824  680 1588  960  458  950 1610  741 1226 1053  641  789  793 1844
  994 1264 1809 1028  729 1092 1125 1673  728  732 1080 1199 1362 1078
  660 1008  924  992 1063 1267 1461 1907  928  864 1734  910 1490 1728
  715  884  969 1710  825 1602 1200  572  774 1392 1232 1572 1541  882
 1149  644 1617 1582  720 1064 1606 1202 1151 1052 2216  968  504 1188
 1593  853  725 1431  855 1726 1360  755 1713 1121 1196  617  848 1424
 1140 1100 1157 1212  689 1070 1436  686  798 1248 1498 1010  713 2392
  630 1203  483 1373 1194 1462  894 1414  996 1694  735  540  626  948
 1845 1020 1367 1444 1573 1302 1314  975 1604  963 1482  506  926 1422
  802  740 1095 1385 1152 1240 1560 2121 1160  807 1468 1575  625  858
  698 1079  768  795 1416 1003  702 1165 1470 2000  700  319  861 1896
  697  972 2136  716 1347 1372 1249 1136 1502 1162  710 1719 1383  844
  596 1056 3206 1358  943 1499 1922 1536 1208 1215  967  721 1684  536
  958 1478  764 1848 1869  616  624  940 1142 1062  888  883 1394 1099
 1268  953  744  608  847  683  870 1580 1856  982 1026 1293  939  784
 1256  658 1041 1682  804  788 1144  961 1260 1310 1141  806 1281 1034
 1276 1340 1344  988  651 1518  907  901  765  799  648 3094 1440 1258
  915 1517  930  813 1533  872 1242 1364  588  709  560 1375 1277 1626
 1488  808  547 1976 2153 1705 1833 1792 1216  999 1113 1073  954  264
 1269  190 3200  866 1501  777 1218 1368 1084 2006 1244 3138 1379 1257
 1452  528 2035  611  707  880 1051 1581 1838 1650  723  654 1204 1069
 1709  998  993 1374 1389 1163 1122 1496  846  372 1164 1050 2042 1868
 1437  742  770 1722 1814 1430 1058  908  600  965 1032 1299 1120  936
  783 1822 1522  980 1116  978 1156  636 1554 1386  811 1520 1952 1766
  981 1094 2109  525  776 1486 1629 1138 2077 1406 1021 1408  738 1477
```

 2046 923 1291 1195 1190 874 551 1419 2444 1210 927 1112 1391 1800
  360 1473 1643 1324 270 859 718 1176 1311 971 1742 941 1698 1584
 1595 868 1153 893 1349 1337 1720 1479 1030 1318 1252 983 1860 836
 1935 1614 761 1413 956 712 650 773 1926 731 1417 1024 849 1442
 1649 1568 778 1489 2078 1454 1516 1067 1559 1127 1390 1273 918 1763
 1090 1054 1039 1148 1002 1638 105 676 1184 1109 892 2217 1505 1059
  951 2330 1670 1623 1017 1105 1001 546 480 1134 1104 1272 1316 1126
 1181 1753 964 1466 925 1905 1500 585 1632 819 1616 1161 828 945
  979 561 696 1330 817 1098 1428 673 1241 944 1225 1266 1128 485
 1930 1396 916 822 750 1700 1007 1187 691 1574 1680 1346 985 1657
  602 1022 1082 810 1504 1220 1132 1565 1338 1654 1620 1055 800 1306
 1475 2524 1992 1193 973 854 662 1103 1154 942 1048 727 690 1096
 1459 1251 1247 1074 1271 290 655 1463 1836 803 833 408 533 1012
 1552 1005 1530 974 1567 1006 1042 1298 704 932 1219 1296 1198 959
 1261 1598 1683 818 1600 2396 1624 831 1224 663 879 815 1630 2158
  931 1660 559 1300 1702 1075 1361 1106 1476 1689 2076 792 2110 1405
 1192 746 1986 841 2002 1332 935 1019 661 1309 1328 1085 6110 1246
  771 976 1652 1278 1902 1274 1393 1622 1352 420 1795 544 1510 911
  693 1284 1732 2033 570 1980 814 873 757 1108 2633 1571 984 1205
  714 1746 1525 482 1356 862 839 1286 1485 1594 622 791 708 1223
  913 656 1319 1932 539 1221 1542]
**8
Heating
['GasA' 'GasW' 'Grav' 'Wall' 'OthW' 'Floor']
**8
HeatingQC
['Ex' 'Gd' 'TA' 'Fa' 'Po']
**8
CentralAir
['Y' 'N']
**8
Electrical
['SBrkr' 'FuseF' 'FuseA' 'FuseP' 'Mix']
**8
1stFlrSF
[856 1262 920 961 1145 796 1694 1107 1022 1077 1040 1182 912 1494
 1253 854 1004 1296 1114 1339 1158 1108 1795 1060 1600 900 1704 520
  649 1228 1234 1700 1561 1132 1097 1297 1057 1152 1324 1328 884 938
 1150 1752 1518 1656 736 955 794 816 1842 1360 1425 983 860 1426
  780 581 1370 902 1143 2207 1479 747 1304 2223 845 885 1086 840
  526 952 1072 1768 682 1337 1563 1065 804 1301 684 612 1013 990
 1235 964 1260 905 680 1588 960 835 1225 1610 977 1535 1226 1053
 1047 789 997 1844 1216 774 1282 2259 1436 729 1092 1125 1699 728
  988 772 1080 1199 1586 958 660 1327 1721 1682 1214 1959 928 864
 1734 910 1501 1728 970 875 896 969 1710 1252 1200 572 991 1392
 1232 1572 1541 882 1149 808 1867 1707 1064 1362 1651 2158 1164 2234
  968 769 901 1340 936 1217 1224 1593 1549 725 1431 855 1726 929
 1713 1121 1279 865 848 720 1442 1696 1100 1180 1212 932 689 1236
  810 1137 1248 1498 1010 811 2392 630 483 1555 1194 1490 894 1414
 1014 798 1566 866 889 626 1222 1872 908 1375 1444 1306 1625 1302
 1314 1005 1604 963 1382 1482 926 764 1422 802 1052 778 1113 1095
 1363 1632 1560 2121 1156 1175 1468 1575 625 1085 858 698 1079 1148
 1644 1003 975 1041 1336 1210 1675 2000 1122 1035 861 1944 697 972
  793 2036 832 716 1153 1088 1372 1472 1249 1136 1553 1163 1898 803

```
 1719 1383 1445  596 1056 1629 1358  943 1619 1922 1536 1621 1215  993
  841 1684  536 1478 1848 1869 1453  616 1192 1167 1142 1352  495  790
  672 1394 1268 1287  953 1120  752 1319  847  904  914 1580 1856 1007
 1026  939  784 1269  658 1742  788  735 1144  876 1112 1288 1310 1165
  806 1620 1166 1071 1050 1276 1028  756 1344 1602 1470 1196  707  907
 1208 1412  765  827  734  694 2402 1440 1128 1258  933 1689 1888  956
  679  813 1533  888  786 1242  624 1663  833  979  575  849 1277 1634
 1502 1161 1976 1652 1493 2069 1718 1131 1850 1792  916  999 1073 1484
 1766  886 3228 1133  899 1801 1218 1368 2020 1378 1244 3138 1266 1476
  605 2515 1509  751  334  820  880 1159 1601 1838 1680  767  664 1377
  915  768  825 1069 1717 1126 1006 1048  897 1557 1389  996 1134 1496
  846  576  877 1320  703 1429 2042 1521  989 2028  838 1473  779  770
  924 1826 1402 1647 1058  927  600 1186 1940 1029 1032 1299 1054  807
 1828 1548  980 1012 1116 1520 1350 1089 1554 1411  800 1567  981 1094
 1051  822  755  909 2113  525  851 1486 1686 1181 2097 1454 1465 1679
 1437  738 1839  792 2046  923 1291 1668 1195 1190  874  551 1419 2444
 1238 1067 1391 1800 1264  372 1824  859 1576 1178 1325  971 1698 1776
 1616 1146  948 1349 1464 1720 1038  742  757 1506 1836 1690 1220 1117
 1973 1204 1614 1430 1110 1342  966  976 1062 1127 1285  773 1966 1428
 1075 1309 1044  686 1661 1008  944 1489 2084 1434 1160  941 1516 1559
 1099 1701 1307 1456  918 1779  702 1512 1039 1002 1646 1547 1036  676
 1184 1462 1155 1090 1187  954  892 1709 1712  872 2217 1505 1068  951
 2364 1670 1063 1636 1020 1105 1015 1001  546  480 1229 1272 1316 1617
 1098 1788 1466  925 1905 1500 1207 1188 1381  965 1168  561  696 1542
  824  783  673  869 1241 1118 1407  750  691 1574 1504  985 1657 1664
 1082 2898 1687 1654 1055 1803 1532 2524 1733 1992 1771  930 1526 1091
 1523 1364 1130 1096 1338 1103 1154  799  893  829 1240 1459 1251 1247
 1390  438  950  887 1021 1552  812 1530  974  986 1042 1298 1811 1265
 1640 1432  959 1831 1261 1170 2129  818 1124 2411  949 1624  831 1622
  842  663  879  815 1630 1074 2196 1283 1660 1318 1211 2136 1138 1702
 1507 1361 1024 1141 1173 2076 1140 1034 2110 1405  760 1987 1104  713
 2018 1968 1332  935 1357  661 1724 1573 1582 1659 4692 1246  753 1203
 1294 1902 1274 1787 1061  708 1584 1334  693 1284 1172 2156 2053  992
 1078 1980 1281  814 2633 1571  984  754 2117  998 1416 1746 1525 1221
  741 1569 1223  962 1537 1932 1423  913 1578 2073 1256]
****************************************8
2ndFlrSF
[ 854    0  866  756 1053  566  983  752 1142 1218  668 1320  631  716
  676  860 1519  530  808  977 1330  833  765  462  213  548  960  670
 1116  876  612 1031  881  790  755  592  939  520  639  656 1414  884
  729 1523  728  351  688  941 1032  848  836  475  739 1151  448  896
  524 1194  956 1070 1096  467  547  551  880  703  901  720  316 1518
  704 1178  754  601 1360  929  445  564  882  920  518  817 1257  741
  672 1306  504 1304 1100  730  689  591  888 1020  828  700  842 1286
  864  829 1092  709  844 1106  596  807  625  649  698  840  780  568
  795  648  975  702 1242 1818 1121  371  804  325  809 1200  871 1274
 1347 1332 1177 1080  695  167  915  576  605  862  495  403  838  517
 1427  784  711  468 1081  886  793  665  858  874  526  590  406 1157
  299  936  438 1098  766 1101 1028 1017 1254  378 1160  682  110  600
  678  834  384  512  930  868  224 1103  560  811  878  574  910  620
  687  546  902 1000  846 1067  914  660 1538 1015 1237  611  707  527
 1288  832  806 1182 1040  439  717  511 1129 1370  636  533  745  584
  812  684  595  988  800  677  573 1066  778  661 1440  872  788  843
  713  567  651  762  482  738  586  679  644  900  887 1872 1281  472
```

```
 1312  319  978 1093  473  664 1540 1276  441  348 1060  714  744 1203
  783 1097  734  767 1589  742  686 1128 1111 1174  787 1072 1088 1063
  545  966  623  432  581  540  769 1051  761  779  514  455 1426  785
  521  252  813 1120 1037 1169 1001 1215  928 1140 1243  571 1196 1038
  561  979  701  332  368  883 1336 1141  634  912  798  985  826  831
  750  456  602  855  336  408  980  998 1168 1208  797  850  898 1054
  895  954  772 1230  727  454  370  628  304  582 1122 1134  885  640
  580 1112  653  220  240 1362  534  539  650  918  933  712 1796  971
 1175  743  523 1216 2065  272  685  776  630  984  875  913  464 1039
 1259  940  892  725  924  764  925 1479  192  589  992  903  430  748
  587  994  950 1323  732 1357  557 1296  390 1185  873 1611  457  796
  908  550  989  932  358 1392  349  691 1349  768  208  622  857  556
 1044  708  626  904  510 1104  830  981  870  694 1152]
****************************************8
LowQualFinSF
[  0 360 513 234 528 572 144 392 371 390 420 473 156 515  80  53 232 481
 120 514 397 479 205 384]
****************************************8
GrLivArea
[1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 1040 2324  912 1494
 1253  854 1004 1296 1114 1339 2376 1108 1795 1060 1600  900 1704  520
 1317 1228 1234 1700 1561 2452 1097 1297 1057 1152 1324 1328  884  938
 1150 1752 2149 1656 1452  955 1470 1176  816 1842 1360 1425 1739 1720
 2945  780 1158 1111 1370 2034 2473 2207 1479  747 2287 2223  845 1718
 1086 1605  988  952 1285 1768 1230 2142 1337 1563 1065 1474 2417 1560
 1224 1526  990 1235  964 2291 1588  960  835 1225 1610 1732 1535 1226
 1818 1992 1047  789 1517 1844 1855 1430 2696 2259 2320 1458 1092 1125
 3222 1456 1123 1080 1199 1586  754  958  840 1348 1053 2157 2054 1327
 1721 1682 1214 1959 1852 1764  864 1734 1385 1501 1728 1709  875 2035
 1344  969 1993 1252 1200 1096 1968 1947 2462 1232 2668 1541  882 1616
 1355 1867 2161 1707 1382 1767 1651 2158 2060 1920 2234  968 1525 1802
 1340 2082 3608 1217 1593 2727 1431 1726 3112 2229 1713 1121 1279 1310
  848 1284 1442 1696 1100 2062 1212 1392 1236 1436 1954 1248 1498 2267
 1552 2392 1302 2520  987 1555 1194 2794  894 1960 1414 1744 1487 1566
  866 1440 2110 1872 1928 1375 1668 2144 1306 1625 1640 1314 1604 1792
 2574 1316  764 1422 1511 2192  778 1113 1939 1363 2270 1632 1548 2121
 2022 1982 1468 1575 1250  858 1396 1919 1716 2263 1644 1003 1558 1950
 1743 1336 3493 2000 2243 1406  861 1944  972 1118 2036 1641 1432 2353
 2646 1472 2596 2468 2730 1163 2978  803 1719 1383 2134 1192 1056 1629
 1358 1638 1922 1536 1621 1215 1908  841 1684 1112 1577 1478 1626 2728
 1869 1453  720 1595 1167 1142 1352 1924 1505 1574 1394 1268 1287 1664
  752 1319  904  914 2466 1856 1800 1691 1301 1797  784 1953 1269 1184
 2332 1367 1961  788 1034 1144 1812 1550 1288  672 1572 1620 1639 1680
 2172 2078 1276 1028 2097 1400 2624 1134 1602 2630 1196 1389  907 1208
 1412 1198 1365  630 1661  694 2402 1573 1258 1689 1888 1886 1376 1183
  813 1533 1756 1590 1242 1663 1666 1203 1935 1135 1660 1277 1634 1502
 1969 1072 1976 1652  970 1493 2643 1131 1850 1826 1216  999 1073 1484
 2414 1304 1578  886 3228 1820  899 1218 1801 1322 1911 1378 1041 1368
 2020 2119 2344 1796 2080 1294 1244 4676 2398 1266  928 2713  605 2515
 1509  827  334 1347 1724 1159 1601 1838 2285  767 1496 2183 1635  768
  825 2094 1069 1126 2046 1048 1446 1557  996 1674 2295 1647 2504 2132
  943 1692 1109 1477 1320 1429 2042 2775 2028  838  860 1473  935 1582
 2296  924 1402 1556 1904 1915 1986 2008 3194 1029 2153 1032 1120 1054
  832 1828 2262 2614  980 1512 1790 1116 1520 1350 1750 1554 1411 3395
```

```
  800 1387  796 1567 1518 1929 2704 1766  981 1094 1839 1665 1510 1469
 2113 1486 2448 1181 1936 2380 1679 1437 1180 1476 1369 1136 1441  792
  923 1291 1761 1102 1419 4316 2519 1539 1137  616 1148 1391 1164 2576
 1824  729 1178 2554 2418  971 1742 1698 1776 1146 2031  948 1349 1464
 2715 2256 2640 1529 1140 2098 1026 1471 1386 2531 1547 2365 1506 1714
 1836 3279 1220 1117 1973 1204 1614 1603 1110 1342 2084  901 2087 1145
 1062 2013 1895 1564  773 3140 1688 2822 1128 1428 1576 2138 1309 1044
 1008 1052  936 1733 1489 1434 2126 1223 1829 1516 1067 1559 1099 1482
 1165 1416 1701 1775 2358 1646 1445 1779 1481 2654 1426 1039 1372 1002
 1949  910 2610 2224 1155 1090 2230  892 1712 1393 2217 1683 1068  951
 2240 2364 1670  902 1063 1636 2057 2274 1015 2002  480 1229 2127 2200
 1617 1686 2374 1978 1788 2236 1466  925 1905 1500 2069 1971 1962 2403
 1381  965 1958 2872 1894 1308 1098 1095  918 2019  869 1241 2612 2290
 1940 2030 1851 1050  944  691 1504  985 1657 1522 1271 1022 1082 1132
 2898 1264 3082 1654  954 1803 2329 2524 2868 1771  930 1977 1989 1523
 1364 2184 1991 1338 2337 1103 1154 2260 1571 1611 2521  893 1240 1740
 1459 1251 1247 1088  438  950 2622 2021 1690 1658 1964  833 1012  698
 1005 1530 1981  974 2210  986 1020 1868 2828 1006 1298  932 1811 1265
 1580 1876 1671 2108 3627 1261 3086 2345 1343 1124 2514 4476 1130 1221
 1699 1624 1804 1622 1863 1630 1074 2196 1283 1845 1902 1211 1846 2136
 1490 1138 1933 1702 1507 2620 1190 1188 1784 1948 1141 1173 2076 1553
 2058 1405  874 2167 1987 1166 1675 1889 2018 3447 1524 1357 1395 2447
 1659 1970 2372 5642 1246 1983 2526 1708 1122 1274 2810 2599 2112 1787
 1923  708  774 2792 1334  693 1861  872 2169 1913 2156 2634 3238 1865
 1078 1980 2601 1738 1475 1374 2633  790 2117 1762 2784 1746 1584 1912
 2482 1687 1513 1608 2093 1840 1848 1569 2450 2201  804 1537 1932 1725
 2555 2007  913 1346 2073 2340 1256]
****************************************8
BsmtFullBath
[1 0 2 3]
****************************************8
BsmtHalfBath
[0 1 2]
****************************************8
FullBath
[2 1 3 0]
****************************************8
HalfBath
[1 0 2]
****************************************8
BedroomAbvGr
[3 4 1 2 0 5 6 8]
****************************************8
KitchenAbvGr
[1 2 3 0]
****************************************8
KitchenQual
['Gd' 'TA' 'Ex' 'Fa']
****************************************8
TotRmsAbvGrd
[ 8  6  7  9  5 11  4 10 12  3  2 14]
****************************************8
Functional
['Typ' 'Min1' 'Maj1' 'Min2' 'Mod' 'Maj2' 'Sev']
```

```
****************************************8
Fireplaces
[0 1 2 3]
****************************************8
FireplaceQu
['Gd' 'TA' 'Fa' 'Ex' 'Po']
****************************************8
GarageType
['Attchd' 'Detchd' 'BuiltIn' 'CarPort' 'Basment' '2Types']
****************************************8
GarageYrBlt
[2003.         1976.         2001.         1998.         2000.
 1993.         2004.         1973.         1931.         1939.
 1965.         2005.         1962.         2006.         1960.
 1991.         1970.         1967.         1958.         1930.
 2002.         1968.         2007.         2008.         1957.
 1920.         1966.         1959.         1995.         1954.
 1953.         1978.50616389 1983.         1977.         1997.
 1985.         1963.         1981.         1964.         1999.
 1935.         1990.         1945.         1987.         1989.
 1915.         1956.         1948.         1974.         2009.
 1950.         1961.         1921.         1900.         1979.
 1951.         1969.         1936.         1975.         1971.
 1923.         1984.         1926.         1955.         1986.
 1988.         1916.         1932.         1972.         1918.
 1980.         1924.         1996.         1940.         1949.
 1994.         1910.         1978.         1982.         1992.
 1925.         1941.         2010.         1927.         1947.
 1937.         1942.         1938.         1952.         1928.
 1922.         1934.         1906.         1914.         1946.
 1908.         1929.         1933.        ]
****************************************8
GarageFinish
['RFn' 'Unf' 'Fin']
****************************************8
GarageCars
[2 3 1 0 4]
****************************************8
GarageArea
[ 548  460  608  642  836  480  636  484  468  205  384  736  352  840
  576  516  294  853  280  534  572  270  890  772  319  240  250  271
  447  556  691  672  498  246    0  440  308  504  300  670  826  386
  388  528  894  565  641  288  645  852  558  220  667  360  427  490
  379  297  283  509  405  758  461  400  462  420  432  506  684  472
  366  476  410  740  648  273  546  325  792  450  180  430  594  390
  540  264  530  435  453  750  487  624  471  318  766  660  470  720
  577  380  434  866  495  564  312  625  680  678  726  532  216  303
  789  511  616  521  451 1166  252  497  682  666  786  795  856  473
  398  500  349  454  644  299  210  431  438  675  968  721  336  810
  494  457  818  463  604  389  538  520  309  429  673  884  868  492
  413  924 1053  439  671  338  573  732  505  575  626  898  529  685
  281  539  418  588  282  375  683  843  552  870  888  746  708  513
 1025  656  872  292  441  189  880  676  301  474  706  617  445  200
  592  566  514  296  244  610  834  639  501  846  560  596  600  373
```

```
  947  350  396  864  304  784  696  569  628  550  493  578  198  422
  228  526  525  908  499  508  694  874  164  402  515  286  603  900
  583  889  858  502  392  403  527  765  367  426  615  871  570  406
  590  612  650 1390  275  452  842  816  621  544  486  230  261  531
  393  774  749  364  627  260  256  478  442  562  512  839  330  711
 1134  416  779  702  567  832  326  551  606  739  408  475  704  983
  768  632  541  320  800  831  554  878  752  614  481  496  423  841
  895  412  865  630  605  602  618  444  397  455  409  820 1020  598
  857  595  433  776 1220  458  613  456  436  812  686  611  425  343
  479  619  902  574  523  414  738  354  483  327  756  690  284  833
  601  533  522  788  555  689  796  808  510  255  424  305  368  824
  328  160  437  665  290  912  905  542  716  586  467  582 1248 1043
  254  712  719  862  928  782  466  714 1052  225  234  324  306  830
  807  358  186  693  482  813  995  757 1356  459  701  322  315  668
  404  543  954  850  477  276  518 1014  753 1418  213  844  860  748
  248  287  825  647  342  770  663  377  804  936  722  208  662  754
  622  620  370 1069  372  923  192]
****************************************8
GarageQual
['TA' 'Fa' 'Gd' 'Ex' 'Po']
****************************************8
GarageCond
['TA' 'Fa' 'Gd' 'Po' 'Ex']
****************************************8
PavedDrive
['Y' 'N' 'P']
****************************************8
WoodDeckSF
[  0 298 192  40 255 235  90 147 140 160  48 240 171 100 406 222 288  49
 203 113 392 145 196 168 112 106 857 115 120  12 576 301 144 300  74 127
 232 158 352 182 180 166 224  80 367  53 188 105  24  98 276 200 409 239
 400 476 178 574 237 210 441 116 280 104  87 132 238 149 355  60 139 108
 351 209 216 248 143 365 370  58 197 263 123 138 333 250 292  95 262  81
 289 124 172 110 208 468 256 302 190 340 233 184 201 142 122 155 670 135
 495 536 306  64 364 353  66 159 146 296 125  44 215 264  88  89  96 414
 519 206 141 260 324 156 220  38 261 126  85 466 270  78 169 320 268  72
 349  42  35 326 382 161 179 103 253 148 335 176 390 328 312 185 269 195
  57 236 517 304 198 426  28 316 322 307 257 219 416 344 380  68 114 327
 165 187 181  92 228 245 503 315 241 303 133 403  36  52 265 207 150 290
 486 278  70 418 234  26 342  97 272 121 243 511 154 164 173 384 202  56
 321  86 194 421 305 117 550 509 153 394 371  63 252 136 186 170 474 214
 199 728 436  55 431 448 361 362 162 229 439 379 356  84 635 325  33 212
 314 242 294  30 128  45 177 227 218 309 404 500 668 402 283 183 175 586
 295  32 366 736]
****************************************8
OpenPorchSF
[ 61   0  42  35  84  30  57 204   4  21  33 213 112 102 154 159 110  90
  56  32  50 258  54  65  38  47  64  52 138 104  82  43 146  75  72  70
  49  11  36 151  29  94 101 199  99 234 162  63  68  46  45 122 184 120
  20  24 130 205 108  80  66  48  25  96 111 106  40 114   8 136 132  62
 228  60 238 260  27  74  16 198  26  83  34  55  22  98 172 119 208 105
 140 168  28  39 148  12  51 150 117 250  10  81  44 144 175 195 128  76
  17  59 214 121  53 231 134 192 123  78 187  85 133 176 113 137 125 523
 100 285  88 406 155  73 182 502 274 158 142 243 235 312 124 267 265  87
```

```
 288  23 152 341 116 160 174 247 291  18 170 156 166 129 418 240  77 364
 188 207  67  69 131 191  41 118 252 189 282 135  95 224 169 319  58  93
 244 185 200  92 180 263 304 229 103 211 287 292 241 547  91  86 262 210
 141  15 126 236]
****************************************8
EnclosedPorch
[  0 272 228 205 176  87 172 102  37 144  64 114 202 128 156  44  77 192
 140 180 183  39 184  40 552  30 126  96  60 150 120 112 252  52 224 234
 244 268 137  24 108 294 177 218 242  91 160 130 169 105  34 248 236  32
  80 115 291 116 158 210  36 200  84 148 136 240  54 100 189 293 164 216
 239  67  90  56 129  98 143  70 386 154 185 134 196 264 275 230 254  68
 194 318  48  94 138 226 174  19 170 220 214 280 190 330 208 145 259  81
  42 123 162 286 168  20 301 198 221 212  50  99]
****************************************8
3SsnPorch
[  0 320 407 130 180 168 140 508 238 245 196 144 182 162  23 216  96 153
 290 304]
****************************************8
ScreenPorch
[  0 176 198 291 252  99 184 168 130 142 192 410 224 266 170 154 153 144
 128 259 160 271 234 374 185 182  90 396 140 276 180 161 145 200 122  95
 120  60 126 189 260 147 385 287 156 100 216 210 197 204 225 152 175 312
 222 265 322 190 233  63  53 143 273 288 263  80 163 116 480 178 440 155
 220 119 165  40]
****************************************8
PoolArea
[  0 512 648 576 555 480 519 738]
****************************************8
PoolQC
['Gd' 'Ex' 'Fa']
****************************************8
Fence
['MnPrv' 'GdWo' 'GdPrv' 'MnWw']
****************************************8
MiscFeature
['Shed' 'Gar2' 'Othr' 'TenC']
****************************************8
MiscVal
[    0   700   350   500   400   480   450 15500  1200   800  2000   600
  3500  1300    54   620   560  1400  8300  1150  2500]
****************************************8
MoSold
[ 2  5  9 12 10  8 11  4  1  7  3  6]
****************************************8
YrSold
[2008 2007 2006 2009 2010]
****************************************8
SaleType
['WD' 'New' 'COD' 'ConLD' 'ConLI' 'CWD' 'ConLw' 'Con' 'Oth']
****************************************8
SaleCondition
['Normal' 'Abnorml' 'Partial' 'AdjLand' 'Alloca' 'Family']
****************************************8
SalePrice
```

```
[208500 181500 223500 140000 250000 143000 307000 200000 129900 118000
 129500 345000 144000 279500 157000 132000 149000  90000 159000 139000
 325300 139400 230000 154000 256300 134800 306000 207500  68500  40000
 149350 179900 165500 277500 309000 145000 153000 109000  82000 160000
 170000 130250 141000 319900 239686 249700 113000 127000 177000 114500
 110000 385000 130000 180500 172500 196500 438780 124900 158000 101000
 202500 219500 317000 180000 226000  80000 225000 244000 185000 144900
 107400  91000 135750 136500 193500 153500 245000 126500 168500 260000
 174000 164500  85000 123600 109900  98600 163500 133900 204750 214000
  94750  83000 128950 205000 178000 118964 198900 169500 100000 115000
 190000 136900 383970 217000 259500 176000 155000 320000 163990 136000
 153900 181000  84500 128000  87000 150000 150750 220000 171000 231500
 166000 204000 125000 105000 222500 122000 372402 235000  79000 109500
 269500 254900 162500 412500 103200 152000 127500 325624 183500 228000
 128500 215000 239000 163000 184000 243000 211000 501837 200100 120000
 475000 173000 135000 153337 286000 315000 192000 148500 311872 104000
 274900 171500 112000 143900 277000  98000 186000 252678 156000 161750
 134450 210000 107000 311500 167240 204900  97000 386250 290000 106000
 192500 148000 403000  94500 128200 216500  89500 185500 194500 318000
 262500 110500 241500 137000  76500 276000 151000  73000 175500 179500
 120500 266000 124500 201000 415298 228500 244600 179200 164700  88000
 153575 233230 135900 131000 167000 142500 175000 158500 267000 149900
 295000 305900  82500 360000 165600 119900 375000 188500 270000 187500
 342643 354000 301000 126175 242000 324000 145250 214500  78000 119000
 284000 207000 228950 377426 202900  87500 140200 151500 157500 437154
 318061  95000 105900 177500 134000 280000 198500 147000 165000 162000
 172400 134432 123000  61000 340000 394432 179000 187750 213500  76000
 240000  81000 191000 426000 106500 129000  67000 241000 245500 164990
 108000 258000 168000 339750  60000 222000 181134 149500 126000 142000
 206300 275000 109008 195400  85400  79900 122500 212000 116000  90350
 555000 162900 199900 119500 188000 256000 161000 263435  62383 188700
 124000 178740 146500 187000 440000 251000 132500 208900 380000 297000
  89471 326000 374000 164000  86000 133000 172785  91300  34900 430000
 226700 289000 208300 164900 202665  96500 402861 265000 234000 106250
 184750 315750 446261 200624 107500  39300 111250 272000 248000 213250
 179665 229000 263000 112500 255500 121500 268000 325000 316600 135960
 142600 224500 118500 146000 131500 181900 253293 369900  79500 185900
 451950 138000 319000 114504 194201 217500 221000 359100 313000 261500
  75500 137500 183200 105500 314813 305000 165150 139900 209500  93000
 264561 274000 370878 143250  98300 205950 350000 145500  97500 197900
 402000 423000 230500 173500 103600 257500 372500 159434 285000 227875
 148800 392000 194700 755000 335000 108480 141500  89000 123500 138500
 196000 312500 361919 213000  55000 302000 254000 179540  52000 102776
 189000 130500 159500 341000 103000 236500 131400  93500 239900 299800
 236000 265979 260400 275500 158900 179400 215200 337000 264132 216837
 538000 134900 102000 395000 221500 175900 187100 161500 233000 107900
 160200 146800 269790 143500 485000 582933 227680 135500 159950 144500
  55993 157900 224900 271000 224000 183000 139500 232600 147400 237000
 139950 174900 133500 189950 250580 248900 169000 200500  66500 303477
 132250 328900 122900 154500 118858 142953 611657 125500 255000 154300
 173733  75000  35311 238000 176500 145900 169990 193000 117500 184900
 253000 239799 244400 150900 197500 172000 116500 214900 178900  37900
  99500 182000 167500  85500 178400 336000 159895 255900 117000 395192
 195000 197000 348000 173900 337500 121600 206000 232000 136905 119200
```

```
227000 203000 213490 194000 287000 293077 310000 119750  84000 315500
262280 278000 139600 556581  84900 176485 200141 185850 328000 167900
151400  91500 138800 155900  83500 252000  92900 176432 274725 134500
184100 133700 118400 212900 163900 259000 239500  94000 424870 174500
116900 201800 218000 235128 108959 233170 245350 625000 171900 154900
392500 745000 186700 104900 262000 219210 116050 271900 229456  80500
137900 367294 101800 138887 265900 248328 465000 186500 169900 171750
294000 165400 301500  99900 128900 183900 378500 381000 185750  68400
150500 281000 333168 206900 295493 111000 156500  72500  52500 155835
108500 283463 410000 156932 144152 216000 274300 466500  58500 237500
377500 246578 281213 137450 193879 282922 257000 223000 274970 182900
192140 143750  64500 394617 149700 149300 121000 179600  92000 287090
266500 142125 147500]
****************************************8
```

In [338…
```
X.columns
```

Out[338…
```
Index(['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities',
       'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2',
       'BldgType', 'HouseStyle', 'OverallQual', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond',
       'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'TotalBsmtSF',
       'Heating', 'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF',
       'GrLivArea', 'FullBath', 'KitchenQual', 'TotRmsAbvGrd', 'Functional',
       'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageCars', 'GarageAre
a',
       'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence',
       'MiscFeature', 'SaleType', 'SaleCondition'],
      dtype='object')
```

In [339…
```
X['avg_Year']=(X['YearBuilt']+X['YearRemodAdd'])/2
```

In [340…
```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 54 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   MSZoning       1460 non-null   object
 1   Street         1460 non-null   object
 2   Alley          1460 non-null   object
 3   LotShape       1460 non-null   object
 4   LandContour    1460 non-null   object
 5   Utilities      1460 non-null   object
 6   LotConfig      1460 non-null   object
 7   LandSlope      1460 non-null   object
 8   Neighborhood   1460 non-null   object
 9   Condition1     1460 non-null   object
 10  Condition2     1460 non-null   object
 11  BldgType       1460 non-null   object
 12  HouseStyle     1460 non-null   object
 13  OverallQual    1460 non-null   int64
 14  YearBuilt      1460 non-null   int64
 15  YearRemodAdd   1460 non-null   int64
 16  RoofStyle      1460 non-null   object
 17  RoofMatl       1460 non-null   object
 18  Exterior1st    1460 non-null   object
 19  Exterior2nd    1460 non-null   object
 20  MasVnrType     1460 non-null   object
 21  ExterQual      1460 non-null   object
 22  ExterCond      1460 non-null   object
 23  Foundation     1460 non-null   object
 24  BsmtQual       1460 non-null   object
 25  BsmtCond       1460 non-null   object
 26  BsmtExposure   1460 non-null   object
 27  BsmtFinType1   1460 non-null   object
 28  BsmtFinType2   1460 non-null   object
 29  TotalBsmtSF    1460 non-null   int64
 30  Heating        1460 non-null   object
 31  HeatingQC      1460 non-null   object
 32  CentralAir     1460 non-null   object
 33  Electrical     1460 non-null   object
 34  1stFlrSF       1460 non-null   int64
 35  GrLivArea      1460 non-null   int64
 36  FullBath       1460 non-null   int64
 37  KitchenQual    1460 non-null   object
 38  TotRmsAbvGrd   1460 non-null   int64
 39  Functional     1460 non-null   object
 40  FireplaceQu    1460 non-null   object
 41  GarageType     1460 non-null   object
 42  GarageFinish   1460 non-null   object
 43  GarageCars     1460 non-null   int64
 44  GarageArea     1460 non-null   int64
 45  GarageQual     1460 non-null   object
 46  GarageCond     1460 non-null   object
 47  PavedDrive     1460 non-null   object
 48  PoolQC         1460 non-null   object
```

```
 49  Fence          1460 non-null   object
 50  MiscFeature    1460 non-null   object
 51  SaleType       1460 non-null   object
 52  SaleCondition  1460 non-null   object
 53  avg_Year       1460 non-null   float64
dtypes: float64(1), int64(10), object(43)
memory usage: 616.1+ KB
```

In [341…
```
X['avg_area']=(X['GrLivArea']+X['GarageCars'])/2
```

In [342…
```
X=X.drop(labels=['YearBuilt','YearRemodAdd','GrLivArea','GarageCars'],axis=1)
```

In [343…
```
len(X.columns)
```

Out[343… 51

In [344…
```
Y['SalePrice'].mean()
```

Out[344… np.float64(180921.19589041095)

In [345…
```
cat=[]
con=[]
for i in X.columns:
  if X[i].dtype==object:
      cat.append(i)
  else:
    con.append(i)
```

In [346…
```
Xcat=X[cat]
X[con]=X[con]
```

In [347…
```
for i in Xcat.columns:
  Xcat[i]=le.fit_transform(Xcat[i])
```

```
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
```

```
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-2553683435.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
```

In [348…

```
Xcon=pd.DataFrame(ss.fit_transform(X[con]),columns=con)
```

In [349…

```
X = Xcat.join(Xcon)
```

In [350…

```
out=[]
for i in Xcon.columns:
  a=Xcon[(Xcon[i]<-3)|(Xcon[i]>3)].index
  out.extend(a)


  out
```

In [351…

```
outlier=list(set(out))
```

In [352…

```
len(outlier)
```

Out[352…

```
40
```

In [353…

```
X=X.drop(index=outlier,axis=0)
Y=Y.drop(index=outlier,axis=0)
```

In [354…
```
X.index=range(X.shape[0])
Y.index=range(Y.shape[0])
```

In [355…
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_absolute_error
```

In [356…
```
lr=LinearRegression()
```

In [357…
```
xtrain,xtest,ytrain,ytest=train_test_split(X,Y,test_size=0.2)
```

In [358…
```
model=lr.fit(xtrain,ytrain)
```

In [359…
```
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)
```

In [360…
```
tr_score=r2_score(ytrain,tr_pred)
ts_score=r2_score(ytest,ts_pred)

print(tr_score)
print(ts_score)
```

```
0.8797437581826576
0.8646999337694159
```

In [361…
```
from statsmodels.api import OLS,add_constant
```

In [362…
```
for i in df.columns:
    if df[i].dtypes==object:
        df[i]=df[i].fillna(df[i].mode()[0])
    else:
        df[i]=df[i].fillna(df[i].mean())
```

In [363…
```
X
```

Out[363…

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	Lar
0	3	1	0	3	3	0	4	
1	3	1	0	3	3	0	2	
2	3	1	0	0	3	0	4	
3	3	1	0	0	3	0	0	
4	3	1	0	0	3	0	2	
...	...	...	...	...	...	...	...	
1415	3	1	0	3	3	0	4	
1416	3	1	0	3	3	0	4	
1417	3	1	0	3	3	0	4	
1418	3	1	0	3	3	0	4	
1419	3	1	0	3	3	0	4	

1420 rows × 51 columns

In [364…

```
X=df.drop(labels='SalePrice',axis=1)
Y=df[['SalePrice']]
```

In [365…

```
X
```

Out[365…

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

1460 rows × 80 columns

In [366…
```
cat=[]
con=[]
for i in X.columns:
    if X[i].dtype==object:
        cat.append(i)
    else:
        con.append(i)
```

In [367…
```
Xcat=X[cat]
Xcon=X[con]
```

In [368…
```
for i in Xcat.columns:
    Xcat[i]=le.fit_transform(Xcat[i])
```

```
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
```

```
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
```

```
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
/tmp/ipython-input-188807692.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  Xcat[i]=le.fit_transform(Xcat[i])
```

In [369…

```
Xcon=pd.DataFrame(ss.fit_transform(Xcon),columns=con)
```

In [370…

```
X=Xcat.join(Xcon)
```

In [371…

```
out=[]
for i in Xcon.columns:
    a=Xcon[(Xcon[i]<-3)|(Xcon[i]>3)].index
    out.extend(a)

out
```

Out[371… [9,
 48,
 93,
 125,
 165,
 246,
 291,
 300,
 312,
 335,
 411,
 488,
 520,
 535,
 635,
 637,
 703,
 705,
 713,
 861,
 969,
 985,
 1030,
 1062,
 1144,
 1186,
 1190,
 1266,
 1393,
 1416,
 171,
 197,
 231,
 277,
 313,
 446,
 807,
 909,
 934,
 1107,
 1127,
 1173,
 1182,
 1211,
 1298,
 1337,
 53,
 249,
 313,
 335,
 384,
 451,
 457,
 661,

706,
769,
848,
1298,
1396,
375,
533,
88,
185,
191,
218,
241,
250,
304,
375,
378,
398,
461,
508,
519,
583,
676,
703,
726,
745,
980,
991,
1123,
1149,
1213,
1268,
1327,
1352,
1435,
1457,
304,
630,
747,
1132,
1137,
1349,
37,
58,
70,
105,
115,
161,
169,
178,
224,
297,
349,
403,
477,

517,
523,
654,
691,
718,
755,
763,
798,
808,
825,
898,
981,
1111,
1169,
1228,
1289,
1298,
1373,
1417,
70,
178,
523,
898,
1182,
1298,
24,
52,
113,
116,
153,
166,
233,
253,
260,
263,
271,
273,
313,
322,
355,
414,
440,
446,
470,
493,
542,
548,
577,
586,
599,
666,
697,
764,
785,

828,
842,
854,
888,
918,
923,
924,
1040,
1059,
1077,
1152,
1220,
1253,
1299,
1308,
1320,
1369,
1387,
1418,
1445,
1458,
137,
224,
278,
477,
496,
581,
678,
774,
798,
932,
1267,
224,
332,
440,
496,
523,
691,
1044,
1182,
1298,
1373,
224,
440,
496,
523,
529,
691,
898,
1024,
1044,
1182,
1298,
1373,

304,
691,
1169,
1182,
51,
88,
125,
170,
185,
197,
198,
263,
267,
406,
589,
635,
729,
873,
883,
1009,
1031,
1173,
1349,
1440,
118,
185,
197,
304,
496,
523,
608,
635,
691,
769,
798,
1169,
1182,
1268,
1298,
1353,
53,
188,
313,
326,
335,
420,
454,
588,
634,
738,
807,
921,
942,
1163,

1270,
1298,
1,
26,
33,
37,
41,
50,
93,
116,
129,
176,
197,
201,
213,
215,
218,
245,
249,
251,
253,
298,
299,
314,
330,
352,
358,
367,
414,
421,
426,
499,
504,
558,
574,
576,
580,
597,
611,
628,
633,
658,
691,
697,
717,
741,
743,
745,
814,
828,
892,
920,
925,
931,

944,
952,
953,
954,
1006,
1029,
1041,
1047,
1052,
1055,
1069,
1072,
1076,
1080,
1103,
1118,
1123,
1149,
1156,
1181,
1213,
1225,
1276,
1287,
1327,
1335,
1350,
1389,
1405,
1415,
188,
298,
597,
624,
628,
921,
1154,
1163,
1230,
1283,
1350,
1450,
53,
144,
189,
291,
330,
570,
634,
635,
843,
897,
1163,
1213,

1270,
1350,
8,
9,
17,
39,
48,
74,
78,
93,
102,
137,
144,
165,
188,
246,
330,
342,
420,
441,
454,
488,
505,
520,
529,
570,
634,
635,
637,
676,
703,
705,
728,
736,
778,
809,
843,
886,
894,
897,
910,
913,
921,
940,
942,
943,
954,
955,
984,
1003,
1011,
1030,
1062,
1090,

1163,
1186,
1216,
1230,
1232,
1266,
1275,
1283,
1292,
1336,
1350,
1391,
1393,
1412,
1416,
1450,
185,
635,
769,
803,
897,
910,
1031,
1173,
1230,
1298,
1350,
1386,
166,
309,
605,
642,
1298,
93,
653,
178,
581,
664,
825,
1061,
1190,
1298,
53,
64,
166,
169,
335,
343,
357,
480,
661,
769,
828,
848,

893,
961,
974,
1044,
1068,
1210,
1312,
1313,
1423,
1459,
28,
185,
293,
495,
499,
523,
583,
591,
645,
664,
666,
713,
735,
775,
784,
807,
854,
875,
947,
961,
996,
1184,
1193,
1292,
1298,
1328,
1369,
3,
7,
154,
197,
260,
306,
314,
325,
328,
358,
365,
380,
459,
462,
496,
520,
577,

630,
648,
653,
660,
662,
718,
720,
747,
799,
813,
836,
840,
918,
939,
945,
1013,
1030,
1081,
1119,
1139,
1150,
1152,
1185,
1197,
1202,
1248,
1266,
1326,
1360,
1382,
1393,
1419,
1439,
1445,
5,
55,
120,
129,
159,
182,
187,
205,
237,
258,
280,
546,
704,
726,
744,
889,
941,
1080,
1156,
1161,

1181,
1346,
1437,
46,
72,
80,
104,
176,
185,
189,
196,
289,
297,
312,
339,
351,
359,
360,
366,
400,
426,
471,
475,
550,
605,
618,
625,
647,
673,
764,
769,
785,
795,
803,
828,
830,
854,
859,
887,
888,
907,
919,
944,
1037,
1055,
1067,
1070,
1154,
1171,
1184,
1228,
1282,
1293,
1301,

```
 1320,
 1328,
 1386,
 1414,
 197,
 810,
 1170,
 1182,
 1298,
 1386,
 1423,
 346,
 510,
 539,
 705,
 890,
 1230,
 1386,
 1457]
```

In [372…
```
len(out)
```

Out[372…
```
667
```

In [373…
```
outlier=list(set(out))
outlier
```

Out[373… [1,
 3,
 5,
 7,
 8,
 9,
 17,
 24,
 26,
 28,
 33,
 37,
 39,
 41,
 46,
 48,
 50,
 51,
 52,
 53,
 55,
 58,
 64,
 70,
 72,
 74,
 78,
 80,
 88,
 93,
 102,
 104,
 105,
 113,
 115,
 116,
 118,
 120,
 125,
 129,
 137,
 144,
 153,
 154,
 159,
 161,
 165,
 166,
 169,
 170,
 171,
 176,
 178,
 182,

185,
187,
188,
189,
191,
196,
197,
198,
201,
205,
213,
215,
218,
224,
231,
233,
237,
241,
245,
246,
249,
250,
251,
253,
258,
260,
263,
267,
271,
273,
277,
278,
280,
289,
291,
293,
297,
298,
299,
300,
304,
306,
309,
312,
313,
314,
322,
325,
326,
328,
330,
332,
335,
339,

342,
343,
346,
349,
351,
352,
355,
357,
358,
359,
360,
365,
366,
367,
375,
378,
380,
384,
398,
400,
403,
406,
411,
414,
420,
421,
426,
440,
441,
446,
451,
454,
457,
459,
461,
462,
470,
471,
475,
477,
480,
488,
493,
495,
496,
499,
504,
505,
508,
510,
517,
519,
520,
523,

529,
533,
535,
539,
542,
546,
548,
550,
558,
570,
574,
576,
577,
580,
581,
583,
586,
588,
589,
591,
597,
599,
605,
608,
611,
618,
624,
625,
628,
630,
633,
634,
635,
637,
642,
645,
647,
648,
653,
654,
658,
660,
661,
662,
664,
666,
673,
676,
678,
691,
697,
703,
704,
705,

706,
713,
717,
718,
720,
726,
728,
729,
735,
736,
738,
741,
743,
744,
745,
747,
755,
763,
764,
769,
774,
775,
778,
784,
785,
795,
798,
799,
803,
807,
808,
809,
810,
813,
814,
825,
828,
830,
836,
840,
842,
843,
848,
854,
859,
861,
873,
875,
883,
886,
887,
888,
889,
890,

892,
893,
894,
897,
898,
907,
909,
910,
913,
918,
919,
920,
921,
923,
924,
925,
931,
932,
934,
939,
940,
941,
942,
943,
944,
945,
947,
952,
953,
954,
955,
961,
969,
974,
980,
981,
984,
985,
991,
996,
1003,
1006,
1009,
1011,
1013,
1024,
1029,
1030,
1031,
1037,
1040,
1041,
1044,
1047,

1052,
1055,
1059,
1061,
1062,
1067,
1068,
1069,
1070,
1072,
1076,
1077,
1080,
1081,
1090,
1103,
1107,
1111,
1118,
1119,
1123,
1127,
1132,
1137,
1139,
1144,
1149,
1150,
1152,
1154,
1156,
1161,
1163,
1169,
1170,
1171,
1173,
1181,
1182,
1184,
1185,
1186,
1190,
1193,
1197,
1202,
1210,
1211,
1213,
1216,
1220,
1225,
1228,
1230,

1232,
1248,
1253,
1266,
1267,
1268,
1270,
1275,
1276,
1282,
1283,
1287,
1289,
1292,
1293,
1298,
1299,
1301,
1308,
1312,
1313,
1320,
1326,
1327,
1328,
1335,
1336,
1337,
1346,
1349,
1350,
1352,
1353,
1360,
1369,
1373,
1382,
1386,
1387,
1389,
1391,
1393,
1396,
1405,
1412,
1414,
1415,
1416,
1417,
1418,
1419,
1423,
1435,
1437,

```
1439,
1440,
1445,
1450,
1457,
1458,
1459]
```

In [374…
```
X=X.drop(index=outlier,axis=0)
Y=Y.drop(index=outlier,axis=0)
```

In [375…
```
X.index=range(0,X.shape[0])
Y.index=range(0,Y.shape[0])
```

In [376…
```
X=X.drop(labels='Id',axis=1)
```

In [377…
```
ols=OLS(ytrain,add_constant(xtrain))
```

In [378…
```
model=ols.fit()
```

In [379…
```
model.summary()
```

Out[379…

OLS Regression Results

Dep. Variable:	SalePrice	**R-squared:**	0.880
Model:	OLS	**Adj. R-squared:**	0.874
Method:	Least Squares	**F-statistic:**	155.5
Date:	Sun, 31 Aug 2025	**Prob (F-statistic):**	0.00
Time:	14:24:07	**Log-Likelihood:**	-13115.
No. Observations:	1136	**AIC:**	2.633e+04
Df Residuals:	1084	**BIC:**	2.660e+04
Df Model:	51		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.628e+05	6.07e+04	2.682	0.007	4.37e+04	2.82e+05
MSZoning	-1632.2106	1468.837	-1.111	0.267	-4514.296	1249.875
Street	2.897e+04	1.34e+04	2.163	0.031	2692.426	5.52e+04
Alley	-1.016e+04	5332.999	-1.905	0.057	-2.06e+04	303.887
LotShape	-949.4663	595.721	-1.594	0.111	-2118.363	219.431
LandContour	-1084.7631	1301.997	-0.833	0.405	-3639.483	1469.957
Utilities	-1.635e+04	2.65e+04	-0.618	0.537	-6.83e+04	3.56e+04
LotConfig	-235.2081	492.816	-0.477	0.633	-1202.189	731.773
LandSlope	1.144e+04	3392.046	3.374	0.001	4788.402	1.81e+04
Neighborhood	171.5622	140.550	1.221	0.222	-104.219	447.344
Condition1	325.9474	969.083	0.336	0.737	-1575.544	2227.439
Condition2	3975.7545	3757.775	1.058	0.290	-3397.582	1.13e+04
BldgType	-4555.0994	728.877	-6.249	0.000	-5985.268	-3124.930
HouseStyle	-645.9856	546.951	-1.181	0.238	-1719.189	427.218
RoofStyle	3563.5489	1029.184	3.462	0.001	1544.131	5582.967
RoofMatl	1281.4222	1453.221	0.882	0.378	-1570.022	4132.867
Exterior1st	-1169.8477	450.272	-2.598	0.010	-2053.351	-286.344
Exterior2nd	296.2540	413.997	0.716	0.474	-516.072	1108.581
MasVnrType	6269.9106	2806.408	2.234	0.026	763.303	1.18e+04
ExterQual	-1.056e+04	1804.994	-5.850	0.000	-1.41e+04	-7017.174
ExterCond	704.9064	1173.202	0.601	0.548	-1597.098	3006.911

Foundation	177.4043	1465.775	0.121	0.904	-2698.673	3053.481
BsmtQual	-8784.5939	1296.048	-6.778	0.000	-1.13e+04	-6241.547
BsmtCond	1306.0953	1200.989	1.088	0.277	-1050.431	3662.622
BsmtExposure	-3094.0371	795.736	-3.888	0.000	-4655.395	-1532.679
BsmtFinType1	-2752.6625	475.061	-5.794	0.000	-3684.805	-1820.520
BsmtFinType2	-572.9849	892.100	-0.642	0.521	-2323.424	1177.454
Heating	2879.2027	2733.891	1.053	0.293	-2485.114	8243.520
HeatingQC	-581.0257	559.240	-1.039	0.299	-1678.342	516.290
CentralAir	9729.2694	3955.300	2.460	0.014	1968.358	1.75e+04
Electrical	-152.1745	820.566	-0.185	0.853	-1762.252	1457.903
KitchenQual	-6601.4797	1290.747	-5.114	0.000	-9134.124	-4068.835
Functional	5389.6432	869.662	6.197	0.000	3683.232	7096.054
FireplaceQu	-761.2115	965.060	-0.789	0.430	-2654.808	1132.385
GarageType	162.9020	552.750	0.295	0.768	-921.678	1247.482
GarageFinish	-2883.7757	1312.125	-2.198	0.028	-5458.367	-309.184
GarageQual	1095.7379	1571.681	0.697	0.486	-1988.144	4179.620
GarageCond	765.7776	1769.211	0.433	0.665	-2705.688	4237.244
PavedDrive	-957.1764	1808.449	-0.529	0.597	-4505.634	2591.281
PoolQC	9832.1287	2.61e+04	0.377	0.706	-4.13e+04	6.1e+04
Fence	-1421.1016	1762.084	-0.806	0.420	-4878.584	2036.381
MiscFeature	-3328.7197	1.16e+04	-0.288	0.773	-2.6e+04	1.93e+04
SaleType	-391.9810	511.312	-0.767	0.443	-1395.254	611.292
SaleCondition	3136.8713	765.112	4.100	0.000	1635.602	4638.140
OverallQual	1.515e+04	1488.816	10.176	0.000	1.22e+04	1.81e+04
TotalBsmtSF	8365.4009	1724.371	4.851	0.000	4981.917	1.17e+04
1stFlrSF	3485.9582	1790.440	1.947	0.052	-27.163	6999.079
FullBath	-2439.0587	1204.309	-2.025	0.043	-4802.100	-76.018
TotRmsAbvGrd	-3010.4574	1581.863	-1.903	0.057	-6114.318	93.403
GarageArea	5781.3418	1157.017	4.997	0.000	3511.094	8051.589
avg_Year	6769.9750	1707.737	3.964	0.000	3419.131	1.01e+04
avg_area	3.217e+04	2041.422	15.757	0.000	2.82e+04	3.62e+04

Omnibus:	162.594	**Durbin-Watson:**	1.980
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1038.471
Skew:	0.473	**Prob(JB):**	3.15e-226
Kurtosis:	7.587	**Cond. No.**	2.22e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.22e+03. This might indicate that there are strong multicollinearity or other numerical problems.

In [380…

```
model.pvalues
```

Out[380…

	0
const	7.432471e-03
MSZoning	2.667173e-01
Street	3.074425e-02
Alley	5.702205e-02
LotShape	1.112701e-01
LandContour	4.049419e-01
Utilities	5.368662e-01
LotConfig	6.332634e-01
LandSlope	7.675793e-04
Neighborhood	2.224856e-01
Condition1	7.366750e-01
Condition2	2.902877e-01
BldgType	5.909135e-10
HouseStyle	2.378357e-01
RoofStyle	5.560616e-04
RoofMatl	3.780908e-01
Exterior1st	9.501492e-03
Exterior2nd	4.743959e-01
MasVnrType	2.567710e-02
ExterQual	6.511168e-09
ExterCond	5.480724e-01
Foundation	9.036888e-01
BsmtQual	1.997145e-11
BsmtCond	2.770504e-01
BsmtExposure	1.070981e-04
BsmtFinType1	8.983290e-09
BsmtFinType2	5.208222e-01
Heating	2.925060e-01
HeatingQC	2.990572e-01
CentralAir	1.405646e-02
Electrical	8.529103e-01

	0
KitchenQual	3.718355e-07
Functional	8.142588e-10
FireplaceQu	4.304180e-01
GarageType	7.682703e-01
GarageFinish	2.817490e-02
GarageQual	4.858423e-01
GarageCond	6.652203e-01
PavedDrive	5.967195e-01
PoolQC	7.061655e-01
Fence	4.201379e-01
MiscFeature	7.733710e-01
SaleType	4.434753e-01
SaleCondition	4.442347e-05
OverallQual	2.719369e-23
TotalBsmtSF	1.406098e-06
1stFlrSF	5.179466e-02
FullBath	4.308370e-02
TotRmsAbvGrd	5.729110e-02
GarageArea	6.792645e-07
avg_Year	7.843582e-05
avg_area	1.598466e-50

dtype: float64

In [381…

```
model.pvalues.sort_values()
```

Out[381…

	0
avg_area	1.598466e-50
OverallQual	2.719369e-23
BsmtQual	1.997145e-11
BldgType	5.909135e-10
Functional	8.142588e-10
ExterQual	6.511168e-09
BsmtFinType1	8.983290e-09
KitchenQual	3.718355e-07
GarageArea	6.792645e-07
TotalBsmtSF	1.406098e-06
SaleCondition	4.442347e-05
avg_Year	7.843582e-05
BsmtExposure	1.070981e-04
RoofStyle	5.560616e-04
LandSlope	7.675793e-04
const	7.432471e-03
Exterior1st	9.501492e-03
CentralAir	1.405646e-02
MasVnrType	2.567710e-02
GarageFinish	2.817490e-02
Street	3.074425e-02
FullBath	4.308370e-02
1stFlrSF	5.179466e-02
Alley	5.702205e-02
TotRmsAbvGrd	5.729110e-02
LotShape	1.112701e-01
Neighborhood	2.224856e-01
HouseStyle	2.378357e-01
MSZoning	2.667173e-01
BsmtCond	2.770504e-01
Condition2	2.902877e-01

	0
Heating	2.925060e-01
HeatingQC	2.990572e-01
RoofMatl	3.780908e-01
LandContour	4.049419e-01
Fence	4.201379e-01
FireplaceQu	4.304180e-01
SaleType	4.434753e-01
Exterior2nd	4.743959e-01
GarageQual	4.858423e-01
BsmtFinType2	5.208222e-01
Utilities	5.368662e-01
ExterCond	5.480724e-01
PavedDrive	5.967195e-01
LotConfig	6.332634e-01
GarageCond	6.652203e-01
PoolQC	7.061655e-01
Condition1	7.366750e-01
GarageType	7.682703e-01
MiscFeature	7.733710e-01
Electrical	8.529103e-01
Foundation	9.036888e-01

dtype: float64

In [382…
```
c=[ "OverallQual", "BsmtQual", "BldgType", "Functional", "GarageArea", "BsmtFi
```

In [383…
```
#"avg_area","avg_Year",const
```

In [384…
```
X=X.drop(labels=c,axis=1)
```

In [385…
```
X
```

Out[385…

	LandContour	ExterCond	Foundation	Electrical	FireplaceQu	GarageQual
0	3	4	2	4	2	4
1	3	4	2	4	4	4
2	3	4	2	4	4	4
3	3	4	2	4	2	4
4	3	4	1	4	2	4
...	...	...	...	...	...	...
1016	3	4	2	4	2	4
1017	3	4	2	4	2	4
1018	3	4	2	4	2	4
1019	3	4	2	4	4	4
1020	3	4	1	4	4	4

1021 rows × 41 columns

In [386…
```
xtrain,xtest,ytrain,ytest=train_test_split(X,Y,test_size=0.2,random_state=42)
# random state was compelsary or else error occurs
```

In [387…
```
model=lr.fit(xtrain,ytrain)
```

In [388…
```
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)
```

In [389…
```
tr_score=r2_score(ytrain,tr_pred)
ts_score=r2_score(ytest,ts_pred)
```

In [390…
```
print(tr_score)
print(ts_score)
```

```
0.8670730445150556
0.8474499245639848
```

ridge,Lasso

In [391…
```
xtrain,xtest,ytrain,ytest=train_test_split(X,Y,test_size=0.2)
```

In [392…
```
ll=Lasso(alpha=0.2)
model=ll.fit(xtrain,ytrain)
```

In [393…
```
r_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)
```

In [394…
```
tr_score=r2_score(ytrain,tr_pred)
ts_score=r2_score(ytest,ts_pred)
```

In [395…
```
print(tr_score)
print(ts_score)
```

```
-0.7613502881854366
0.8514622643883529
```

grid searchcv

In [396…
```
from sklearn.model_selection import GridSearchCV
import numpy as np
```

In [397…
```
a=[]
e=0.01
for i in range(0,1000):
  a.append(e)
  e=round(e+0.01,4)
```

In [398…
```
rr=Ridge()
hp={'alpha':a,'random_state':np.arange(1,40)} # hyyper parameters
```

In [399…
```
cv=GridSearchCV(rr,hp,scoring='neg_mean_squared_error',cv=4)
cvmodel=cv.fit(X,Y)
```

In [400…
```
from sklearn.metrics import mean_squared_error,mean_absolute_error
```

mean absolute error

In [401…
```
rr=Ridge(alpha=1,random_state=1)
model=rr.fit(xtrain,ytrain)
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)

tr_err=mean_absolute_error(ytrain,tr_pred)
ts_err=mean_absolute_error(ytest,ts_pred)


print(tr_err)
print(ts_err)
```

```
16979.70828740654
18201.877463275414
```

mean square error

In [402…

```
ll=Lasso(alpha=1,random_state=1)
model=ll.fit(xtrain,ytrain)

tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)

tr_err=mean_squared_error(ytrain,tr_pred)
ts_err=mean_squared_error(ytest,ts_pred)


print(tr_err)
print(ts_err)
```

```
599616014.7163607
678009340.3618506
```

R_mean square error

In [403…

```
model=lr.fit(xtrain,ytrain)
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)

tr_rmse=np.sqrt(mean_squared_error(ytrain,tr_pred))
ts_rmse=np.sqrt(mean_squared_error(ytest,ts_pred))


print(tr_rmse)
print(ts_rmse)
```

```
24487.02520364694
26039.969367127607
```

r2_score

In [404…

```
model=lr.fit(xtrain,ytrain)
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)

tr_score=r2_score(ytrain,tr_pred)
ts_score=r2_score(ytest,ts_pred)

print(tr_score)
print(ts_score)
```

```
0.8662277037283583
0.8514591617591415
```

r2_adj

In [405…

```
model=lr.fit(xtrain,ytrain)
tr_pred=model.predict(xtrain)
ts_pred=model.predict(xtest)

def adjusted_r2(y_true, y_pred, X):# ytrue n y pred are in formula
  r2=r2_score(y_true,y_pred)
  n,p=X.shape
  return 1-(1-r2)*((n-1)/(n-p-1))

tr_r2_score=adjusted_r2(ytrain,tr_pred,xtrain)
ts_r2_score=adjusted_r2(ytest,ts_pred,xtest)

print(tr_r2_score)
print(ts_r2_score)
```

```
0.8591415743392921
0.8140961288273918
```

In [406…

```
df_test
```

Out[406…

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotSha
0	1461	20	RH	80.0	11622	Pave	NaN	
1	1462	20	RL	81.0	14267	Pave	NaN	
2	1463	60	RL	74.0	13830	Pave	NaN	
3	1464	60	RL	78.0	9978	Pave	NaN	
4	1465	120	RL	43.0	5005	Pave	NaN	
...	...	...	...	...	...	...	...	
1454	2915	160	RM	21.0	1936	Pave	NaN	
1455	2916	160	RM	21.0	1894	Pave	NaN	
1456	2917	20	RL	160.0	20000	Pave	NaN	
1457	2918	85	RL	62.0	10441	Pave	NaN	
1458	2919	60	RL	74.0	9627	Pave	NaN	

1459 rows × 80 columns

In [407…

```
df_test.isna().sum()
```

Out[407…

	0
Id	0
MSSubClass	0
MSZoning	4
LotFrontage	227
LotArea	0
...	...
MiscVal	0
MoSold	0
YrSold	0
SaleType	1
SaleCondition	0

80 rows × 1 columns

dtype: int64

In [408…
```
df3.columns
```

Out[408…
```
Index(['LandContour', 'ExterCond', 'Foundation', 'Electrical', 'FireplaceQu',
       'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence',
       'MiscFeature', 'MSSubClass', 'LotFrontage', 'LotArea', 'OverallCond',
       'YearBuilt', 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2',
       'BsmtUnfSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath',
       'BsmtHalfBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr',
       'Fireplaces', 'GarageYrBlt', 'GarageCars', 'WoodDeckSF', 'OpenPorchS
F',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal',
       'MoSold', 'YrSold', 'SalePrice', 'Id'],
      dtype='object')
```

In [409…
```
from sklearn.preprocessing import LabelEncoder, StandardScaler
import pandas as pd

# 1) Keep only common columns with training data
df3 = df3[[c for c in X.columns if c in df_test.columns]]

# 2) Fill missing values (mean for numbers, mode for categorical)
for col in df3.columns:
    if df3[col].dtype == 'object':
        df3[col] = df3[col].fillna(df3[col].mode()[0])
    else:
        df3[col] = df3[col].fillna(df3[col].mean())
```

```
# 3) Encode categoricals
for col in df3.columns:
    if df3[col].dtype == 'object':
        le = LabelEncoder()
        df3[col] = le.fit_transform(df3[col].astype(str))

# 4) Scale numbers
scaler = StandardScaler()
df3[df3.columns] = scaler.fit_transform(df3)

# 5) Predict
preds = model.predict(df3)
```

In [410…
```
model.predict(df3)
```

Out[410…
```
array([[145686.55417016],
       [234266.94959077],
       [235165.33330418],
       ...,
       [229994.73291435],
       [151623.05665104],
       [280490.05402953]])
```

In [411…
```
df3['SalePrice'] =model.predict(df3)
```

In [412…
```
df3.head(20)
```

Out[412…

	LandContour	ExterCond	Foundation	Electrical	FireplaceQu	GarageQual	G
0	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
1	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
2	0.31870	0.388852	0.833702	0.296079	1.918553	0.250409	
3	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
4	-2.53396	0.388852	0.833702	0.296079	-0.401521	0.250409	
5	0.31870	0.388852	0.833702	0.296079	1.918553	0.250409	
6	0.31870	-2.072470	0.833702	0.296079	-0.401521	0.250409	
7	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
8	0.31870	0.388852	0.833702	0.296079	0.758516	0.250409	
9	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
10	0.31870	0.388852	0.833702	0.296079	-1.561558	0.250409	
11	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
12	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
13	0.31870	0.388852	-0.533007	0.296079	1.918553	0.250409	
14	0.31870	0.388852	-0.533007	0.296079	-0.401521	0.250409	
15	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
16	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
17	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
18	0.31870	0.388852	0.833702	0.296079	-0.401521	0.250409	
19	-2.53396	0.388852	0.833702	0.296079	-0.401521	0.250409	

20 rows × 42 columns

In [413…
```
df3['Id']=df['Id']
```

In [414…
```
result=df3[['Id','SalePrice']]
```

In [415…
```
result.head(20)
```

Out[415…

	Id	SalePrice
0	1	145686.554170
1	2	234266.949591
2	3	235165.333304
3	4	250925.431932
4	5	207103.111077
5	6	213939.568348
6	7	211409.006583
7	8	209703.004724
8	9	219599.900275
9	10	155497.600280
10	11	245457.879292
11	12	111190.829614
12	13	113202.432824
13	14	164302.317367
14	15	111437.753061
15	16	339168.895424
16	17	262935.784240
17	18	297898.893281
18	19	302393.585006
19	20	470485.058757

In [415…