House Prices Prediction

Author: Irfan Ullah Khan

GITHUB PROFILE | KAGGLE PROFILE | LINKEDIN PROFILE

YOUTUBE PROFILE | EMAIL CONTACT ME | WEBSITE CONTACT ME

Introduction

Predicting the selling price of a home is an important topic in real estate. There are several elements that influence the price of a home. Some factors may produce an increase in price, others may cause a fall, and yet others are reliant on one or more factors, i.e. their combination with other factors determines whether they will increase or decrease the price. To assist us in determining the relationship between these attributes and sale prices, we have data from 1460 residences (sold). The dataset covers nearly all of the elements that influence a house's sales price, including overall quality, neighborhood, the presence of a basement and/or garage, and so on, in addition to the sale price. The aim is to, perform exploratory data analysis for finding out which factors affect the most. We will be using of multiple machine learning algorithms and choose the one which has the highest accuracy. Then training, evaluating and tuning the model with appropriate parameter values, we will try to keep the RMSE minimum. I will also create a web application having user frinedly interfece where one can easily get the sale price of their house just by giving the different attributes of the house as an input.

```
#filtering the warnings to keep the notebook clean
import warnings
warnings.filterwarnings('ignore')
```

1. Loading and Exploring Data

```
import pandas as pd
import numpy as np
#showing max rows
pd.options.display.max_columns = 200
pd.options.display.max_rows = 200
```

```
prices_df = pd.read_csv('/content/train.csv')
prices_df.head()
```

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

```
prices_df.shape
```

```
(1460, 81)
```

Let's explore the columns and data types within the dataset.

```
prices_df.info()
```

```
27  ExterQual      1460 non-null   object
28  ExterCond      1460 non-null   object
29  Foundation     1460 non-null   object
30  BsmtQual       1423 non-null   object
31  BsmtCond       1423 non-null   object
32  BsmtExposure   1422 non-null   object
33  BsmtFinType1   1423 non-null   object
34  BsmtFinSF1     1460 non-null   int64
35  BsmtFinType2   1422 non-null   object
36  BsmtFinSF2     1460 non-null   int64
37  BsmtUnfSF      1460 non-null   int64
38  TotalBsmtSF    1460 non-null   int64
39  Heating        1460 non-null   object
40  HeatingQC      1460 non-null   object
41  CentralAir     1460 non-null   object
42  Electrical     1459 non-null   object
43  1stFlrSF       1460 non-null   int64
44  2ndFlrSF       1460 non-null   int64
45  LowQualFinSF   1460 non-null   int64
46  GrLivArea      1460 non-null   int64
47  BsmtFullBath   1460 non-null   int64
48  BsmtHalfBath   1460 non-null   int64
49  FullBath       1460 non-null   int64
50  HalfBath       1460 non-null   int64
51  BedroomAbvGr   1460 non-null   int64
52  KitchenAbvGr   1460 non-null   int64
53  KitchenQual    1460 non-null   object
54  TotRmsAbvGrd   1460 non-null   int64
55  Functional     1460 non-null   object
56  Fireplaces     1460 non-null   int64
57  FireplaceQu    770 non-null    object
58  GarageType     1379 non-null   object
59  GarageYrBlt    1379 non-null   float64
60  GarageFinish   1379 non-null   object
61  GarageCars     1460 non-null   int64
62  GarageArea     1460 non-null   int64
63  GarageQual     1379 non-null   object
64  GarageCond     1379 non-null   object
65  PavedDrive     1460 non-null   object
66  WoodDeckSF     1460 non-null   int64
67  OpenPorchSF    1460 non-null   int64
68  EnclosedPorch  1460 non-null   int64
69  3SsnPorch      1460 non-null   int64
70  ScreenPorch    1460 non-null   int64
71  PoolArea       1460 non-null   int64
72  PoolQC         7 non-null      object
73  Fence          281 non-null    object
74  MiscFeature    54 non-null     object
75  MiscVal        1460 non-null   int64
76  MoSold         1460 non-null   int64
77  YrSold         1460 non-null   int64
78  SaleType       1460 non-null   object
79  SaleCondition  1460 non-null   object
```

```
#Statistical Information of the Numeric Columns with Transpose method
prices_df.describe().T
```

	count	mean	std	min	25%	50%
Id	1460.0	730.500000	421.610009	1.000000	365.750000	730.500000
MSSubClass	1460.0	56.897260	42.300571	20.000000	20.000000	50.000000
LotFrontage	1201.0	4.207109	0.346228	3.091042	4.094345	4.248495
LotArea	1460.0	9.110966	0.517369	7.170888	8.929898	9.156887
OverallQual	1460.0	6.099315	1.382997	1.000000	5.000000	6.000000
OverallCond	1460.0	5.575342	1.112799	1.000000	5.000000	5.000000
YearBuilt	1460.0	7.586821	0.015389	7.535297	7.578145	7.587817
YearRemodAdd	1460.0	7.593756	0.010424	7.576097	7.584773	7.598399
MasVnrArea	1452.0	2.131946	2.631265	0.000000	0.000000	0.000000
BsmtFinSF1	1460.0	4.229731	2.992052	0.000000	0.000000	5.951943
BsmtFinSF2	1460.0	0.655398	1.845045	0.000000	0.000000	0.000000
BsmtUnfSF	1460.0	5.648378	1.854020	0.000000	5.411646	6.170651
TotalBsmtSF	1460.0	6.750560	1.145712	0.000000	6.680541	6.900227
1stFlrSF	1460.0	7.008452	0.317431	5.814131	6.783325	6.992096
2ndFlrSF	1460.0	2.864586	3.293311	0.000000	0.000000	0.000000
LowQualFinSF	1460.0	0.099814	0.747354	0.000000	0.000000	0.000000
GrLivArea	1460.0	7.268512	0.333303	5.814131	7.030415	7.289611
BsmtFullBath	1460.0	0.425342	0.518911	0.000000	0.000000	0.000000
BsmtHalfBath	1460.0	0.057534	0.238753	0.000000	0.000000	0.000000
FullBath	1460.0	1.565068	0.550916	0.000000	1.000000	2.000000
HalfBath	1460.0	0.382877	0.502885	0.000000	0.000000	0.000000
BedroomAbvGr	1460.0	2.866438	0.815778	0.000000	2.000000	3.000000
KitchenAbvGr	1460.0	1.046575	0.220338	0.000000	1.000000	1.000000
TotRmsAbvGrd	1460.0	6.517808	1.625393	2.000000	5.000000	6.000000
Fireplaces	1460.0	0.613014	0.644666	0.000000	0.000000	1.000000
GarageYrBlt	1379.0	7.590525	0.012525	7.550135	7.581720	7.591357
GarageCars	1460.0	1.767123	0.747315	0.000000	1.000000	2.000000
GarageArea	1460.0	5.808156	1.455118	0.000000	5.815592	6.175867
WoodDeckSF	1460.0	2.457206	2.596435	0.000000	0.000000	0.000000
OpenPorchSF	1460.0	2.308541	2.152387	0.000000	0.000000	3.258097
EnclosedPorch	1460.0	0.698019	1.727317	0.000000	0.000000	0.000000
3SsnPorch	1460.0	0.085679	0.666876	0.000000	0.000000	0.000000
ScreenPorch	1460.0	0.410671	1.403194	0.000000	0.000000	0.000000
PoolArea	1460.0	0.030431	0.438685	0.000000	0.000000	0.000000
MiscVal	1460.0	0.233456	1.226030	0.000000	0.000000	0.000000
MoSold	1460.0	6.321918	2.703626	1.000000	5.000000	6.000000
YrSold	1460.0	2007.815753	1.328095	2006.000000	2007.000000	2008.000000
SalePrice	1460.0	180921.195890	79442.502883	34900.000000	129975.000000	163000.000000

```
#Statistical Information of the Categorical Columns
prices_df.describe(include=['O'])
```

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

Visualizing the dataset

```
#importing the dependencies for visualization
import plotly.express as px
import matplotlib
import matplotlib.pyplot as plt
```

```
import seaborn as sns
%matplotlib inline

#setting the style and background
sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (10, 6)
```

Sales Prices

```
px.histogram(prices_df, x='SalePrice', title='Distribution of Sales Price', color='SaleCondition')
```

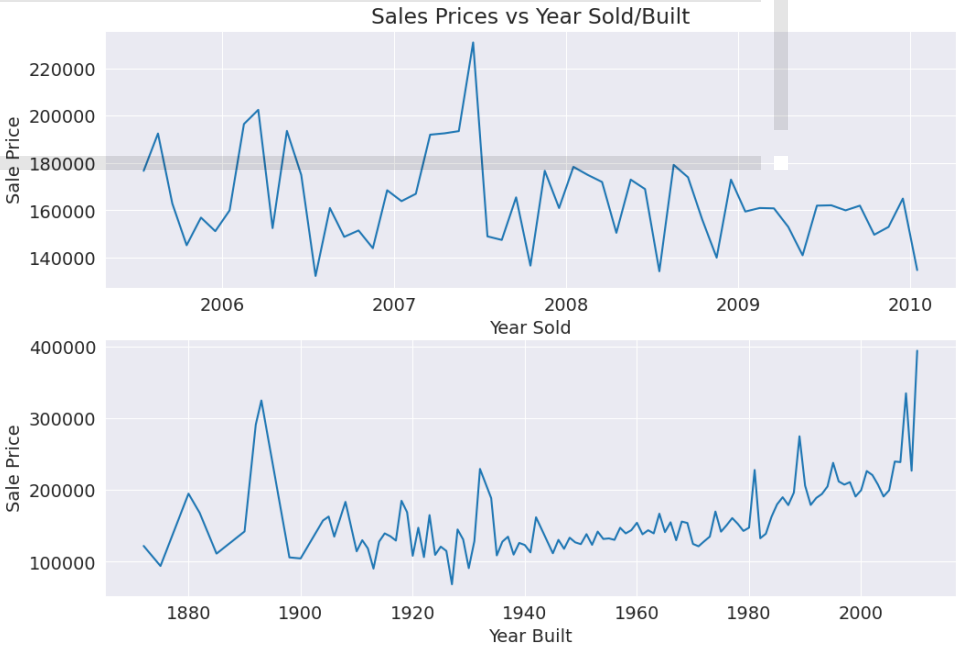

```
px.histogram(x=np.log(prices_df['SalePrice']), title='Logarithmic Distribution of Sales Price').update_layout(
    xaxis_title="Logarithmic SalePrice")
```

Majority of the data points lie in the range of 100k-300k and as we move further the number of expensive properties sold decreases. However the logarithmic distribution of sale prices appears to be normal.

Year Built and Year Sold

```
fig, axes = plt.subplots(2, 1, figsize=(12, 8))
axes[0].set_title('Sales Prices vs Year Sold/Built')

xdmd=prices_df.groupby(['YrSold', 'MoSold'], as_index=False)['SalePrice'].median(
axes[0].plot(xdmd.index, xdmd.SalePrice)
axes[0].set_xlabel('Year Sold')
```

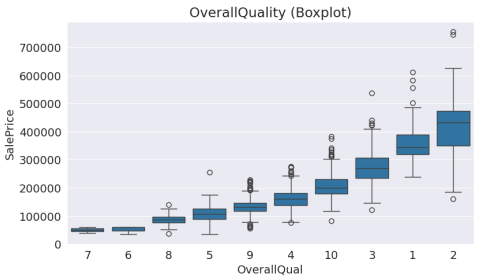
```
axes[0].set_ylabel('Sale Price')
axes[0].set_xticks([5.5, 17.5, 29.5, 41.5, 53.5])
axes[0].set_xticklabels([2006, 2007, 2008, 2009, 2010])


xdmdb=prices_df.groupby('YearBuilt')['SalePrice'].median()
axes[1].plot(xdmdb.index,xdmdb)
axes[1].set_xlabel('Year Built')
```

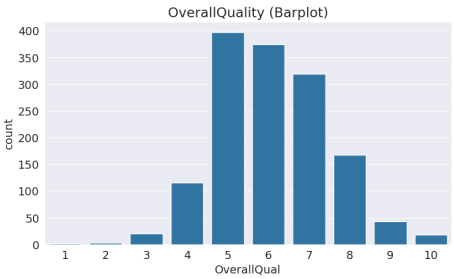


1. The sales prices are seem to be increasing from the end of the 2006 to till end of the 2007 with the a spike in Dec, 2022. Then they show sudden decrease in Jan 2008 and do not show much variation in the prices afterwards. The reasons behind this sudden fall were low interest rates, easy credit, insufficient regulation, and toxic subprime mortgages. You can read more about it over here : https://financialcomplete.com/why-did-house-prices-fall-in-2008.
2. After 1925, the sale price appears to be propotional to the year when the house was built i.e. old houses are sold for the lesser price than the new houses. The reason behind this could be the condition of houses getting bad over the period of time.

Overall Quality of the Houses

```
fig, axes = plt.subplots(1,2, figsize=(20, 5))
axes[0].set_title('OverallQuality (Boxplot)')
axes[0].set_xticklabels(labels=prices_df['OverallQual'].unique().tolist())
sns.boxplot(y=prices_df['SalePrice'], x=prices_df['OverallQual'], ax=axes[0])
axes[1].set_title('OverallQuality (Barplot)')
sns.countplot(x=prices_df['OverallQual'], ax=axes[1]);
```





Sale price is increasing with over all quality which is no surprise. However most houses sold have average (5-6) overall condition.

Quality, Condition and Year Built

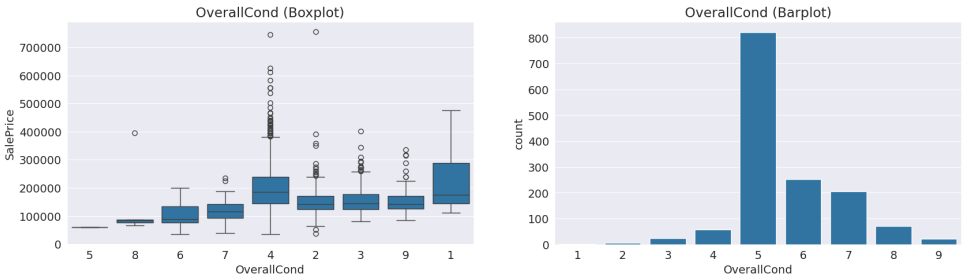
```
px.scatter_3d(prices_df, x='OverallQual', y='OverallCond', z='YearBuilt',
              title='Relation between Quality, Condition and Year Built')
```

Relation between Quality, Condition and Year Built

1. The relationship between over all quality and over all condition isn't linear (as someone might expect). It appears that houses can be in good condition even though their over all quality is poor.
2. The over all quality of house built between 1900 to 1960, falls below average whereas it's above average for houses built after 1960. And it's average for houses built before 1900.
3. The houses having over all condition below average were mostly built between 1900-1980.

Overall Condition of the Houses

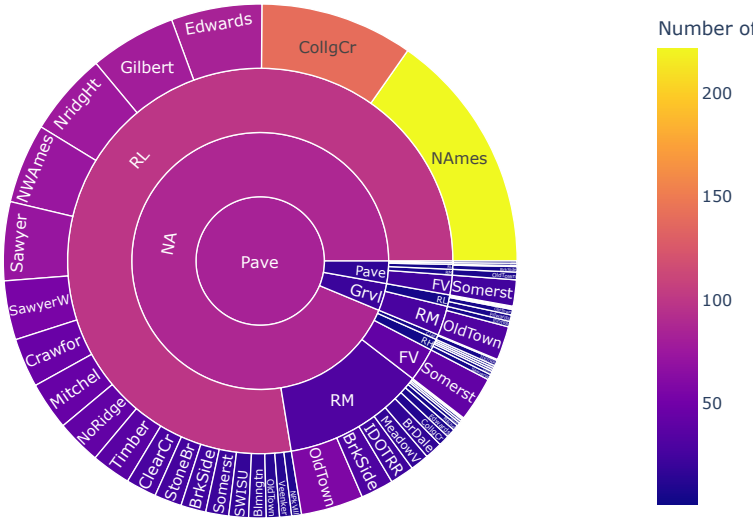
```
fig, axes = plt.subplots(1,2, figsize=(20, 5))
axes[0].set_title('OverallCond (Boxplot)')
axes[0].set_xticklabels(labels=prices_df['OverallCond'].unique().tolist())
sns.boxplot(y=prices_df['SalePrice'], x=prices_df['OverallCond'], ax=axes[0])
axes[1].set_title('OverallCond (Barplot)')
sns.countplot(x=prices_df['OverallCond'], ax=axes[1]);
```



It doesn't show linear increment in sale price with the over all condtion as we have already seen that good condition doesn't imply good quality (and this is the reason sales price isn't significantly increasing with overall condition rating getting increased from 6 to 7 till 8). The above graphs shows large variation in the sale price of the houses having average over all condition and also number of houses sold in that category are larger.

Streets and Allies

```
xdf=prices_df.groupby(['Street','Alley','MSZoning','Neighborhood'], as_index=False, dropna=False)['Id'].count()
xdf.fillna('NA', inplace=True)
xdf.rename(columns={'Id':'Number of Houses'}, inplace=True)
px.sunburst(xdf,path=['Street','Alley','MSZoning','Neighborhood'], values='Number of Houses', color='Number of Houses')
```

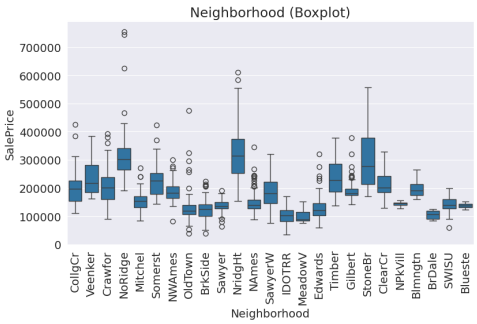


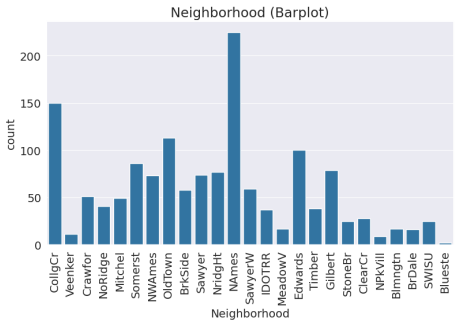
The above sunburst classifies the number of houses based on the `Street, Alley, MSZoning` and `Neighbourhood`. We can draw mainly two conclusions from this.

1. Almost all the streets are `Paved`(only six are `Gravel`)
2. Majority of the houses don't have alley access (indicated by `NA`)

Neighbourhood

```
fig, axes = plt.subplots(1,2, figsize=(20, 5))
axes[0].set_title('Neighborhood (Boxplot)')
axes[0].set_xticklabels(labels=prices_df['Neighborhood'].unique().tolist(),rotation=90)
sns.boxplot(y=prices_df['SalePrice'], x=prices_df['Neighborhood'], ax=axes[0])
axes[1].set_title('Neighborhood (Barplot)')
sns.countplot(x=prices_df['Neighborhood'], ax=axes[1])
plt.xticks(rotation=90);
```





The houses from neighbourhoods `NridgHt (Northridge Heights)` and `StoneBr (Stone Brook)` show large variation in the `SalePrice`. The `SalePrice` is in the `NoRidge (Northridge)` neighbourhood. However the maximum numbers of houses are sold in `NAmes (North Ames)` neighbourhood followed by `CollgCr (College Creek)` and the least number of houses are sold in `Blueste (Bluestem)` followed by `NPkVill (Northpark Villa)`. You may notice that the sale prices from `NAmes` neighbourhood do not vary much and are below average that could be the reason they are among the most sold houses.

Let's explore the North Ames neighbourhood some more

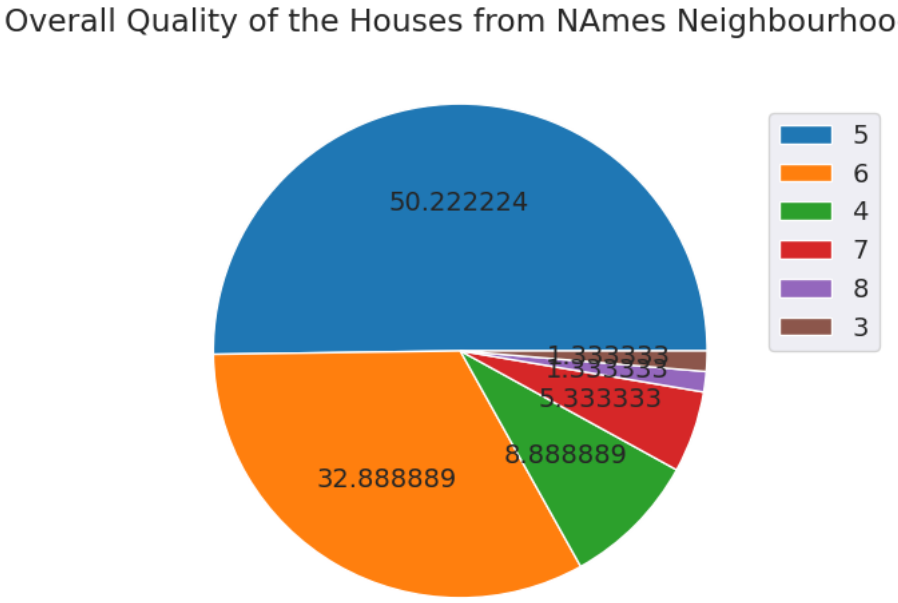
```
NAmes_df=prices_df[prices_df['Neighborhood']=='NAmes']
NAmes_df.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utili
14	15	20	RL	NaN	10920	Pave	NaN	IR1	Lvl	A
16	17	20	RL	NaN	11241	Pave	NaN	IR1	Lvl	A
19	20	20	RL	70.0	7560	Pave	NaN	Reg	Lvl	A
26	27	20	RL	60.0	7200	Pave	NaN	Reg	Lvl	A
28	29	20	RL	47.0	16321	Pave	NaN	IR1	Lvl	A

```
NAmes_df['SalePrice'].median(), prices_df['SalePrice'].median()
```

```
(140000.0, 163000.0)
```

```
NAmes_OverallQual=NAmes_df['OverallQual'].value_counts()/len(NAmes_df['OverallQual'])
plt.pie(NAmes_OverallQual, autopct='%f')
plt.legend(NAmes_OverallQual.index, loc=(1,0.5))
plt.title('Overall Quality of the Houses from NAmes Neighbourhood');
```

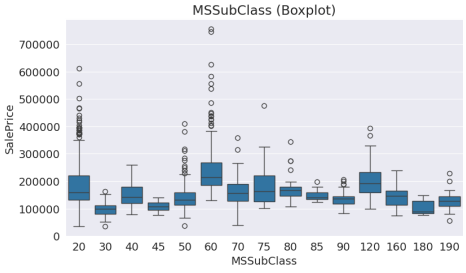


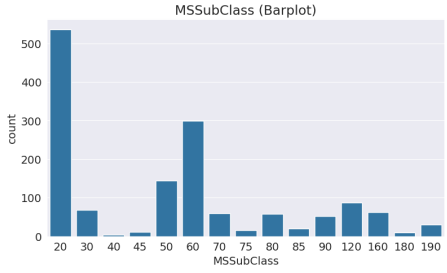
See that the average sale price in the `NAmes` neighbourhood is below average of the whole dataset. Most of the houses (nearly 82%) have average(5-6) overall quality that's what makes them affordable.

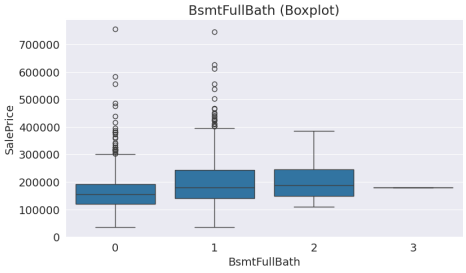
We can similiary use loop through all the other categorical variables and discrete variables to obtain their relationship with `SalePrice` using boxplot.

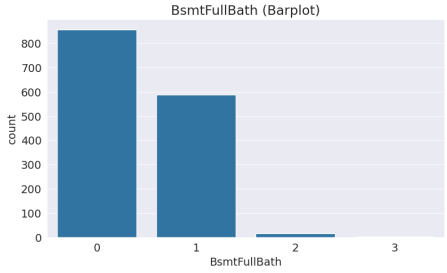
```
discrete = ['MSSubClass','BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath','BedroomAbvGr', 'KitchenAbvGr',
'TotRmsAbvGrd', 'Fireplaces','GarageCars', 'MoSold','YrSold']
categorical=['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilities',
 'LotConfig', 'LandSlope', 'Condition1', 'Condition2','HouseStyle', 'RoofStyle',
 'RoofMatl', 'Exterior1st', 'Exterior2nd','BldgType','MasVnrType', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating', 'HeatingQC', 'CentralAir', 'Electrical',
 'KitchenQual', 'Functional', 'FireplaceQu', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond', 'PavedDrive',
 'PoolQC', 'Fence', 'MiscFeature', 'SaleType', 'SaleCondition']
```

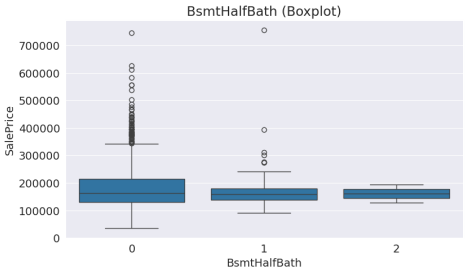
```
for i in discrete+categorical:
    fig, axes = plt.subplots(1,2, figsize=(20, 5))
    axes[0].set_title(i+' (Boxplot)')
    sns.boxplot(y=prices_df['SalePrice'], x=prices_df[i], ax=axes[0])
    axes[1].set_title(i+' (Barplot)')
    sns.countplot(x=prices_df[i], ax=axes[1])
```

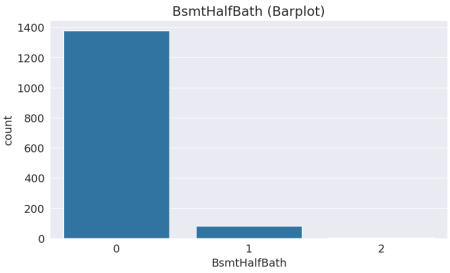

MSSubClass (Boxplot)
SalePrice
MSSubClass

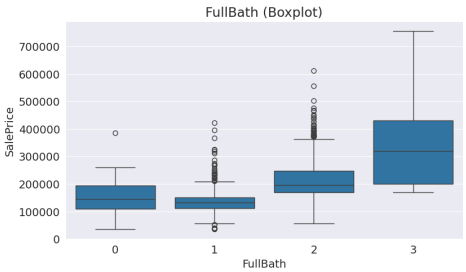

MSSubClass (Barplot)
count
MSSubClass

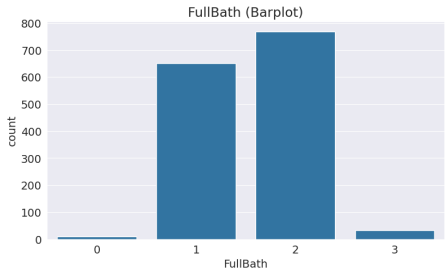

BsmtFullBath (Boxplot)
SalePrice
BsmtFullBath

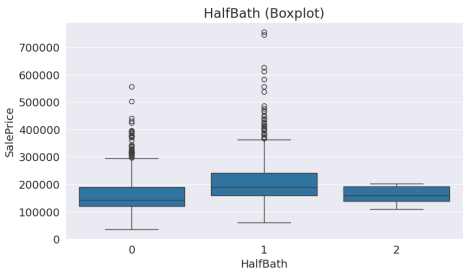

BsmtFullBath (Barplot)
count
BsmtFullBath

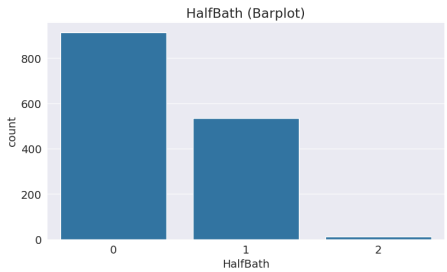

BsmtHalfBath (Boxplot)
SalePrice
BsmtHalfBath

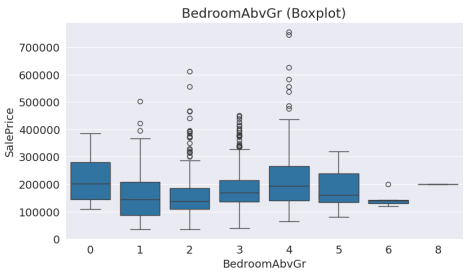

BsmtHalfBath (Barplot)
count
BsmtHalfBath

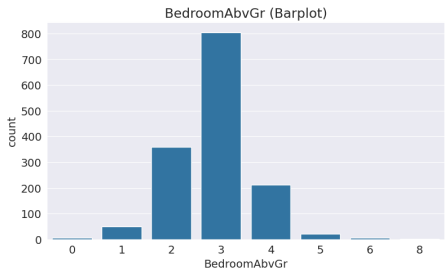

FullBath (Boxplot)
SalePrice
FullBath

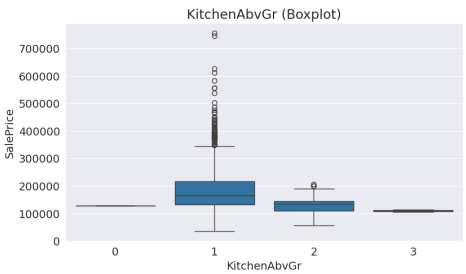

FullBath (Barplot)
count
FullBath

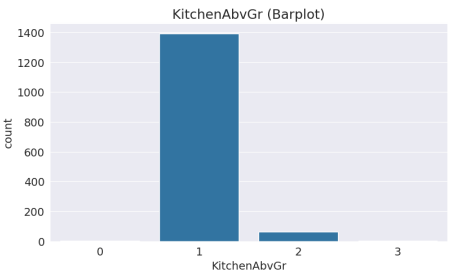

HalfBath (Boxplot)
SalePrice
HalfBath

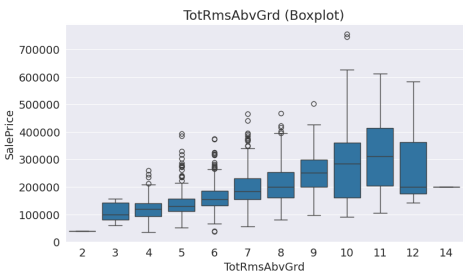

HalfBath (Barplot)
count
HalfBath

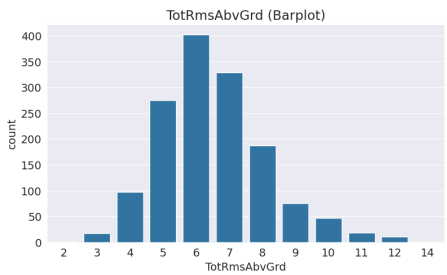

BedroomAbvGr (Boxplot)
SalePrice
BedroomAbvGr


BedroomAbvGr (Barplot)
count
BedroomAbvGr


KitchenAbvGr (Boxplot)
SalePrice
KitchenAbvGr

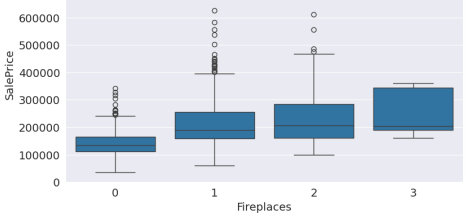

KitchenAbvGr (Barplot)
count
KitchenAbvGr

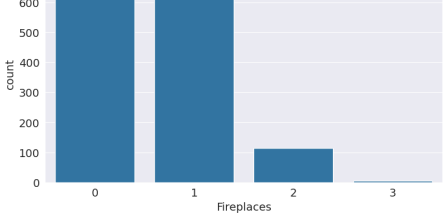

TotRmsAbvGrd (Boxplot)
SalePrice
TotRmsAbvGrd


TotRmsAbvGrd (Barplot)
count
TotRmsAbvGrd

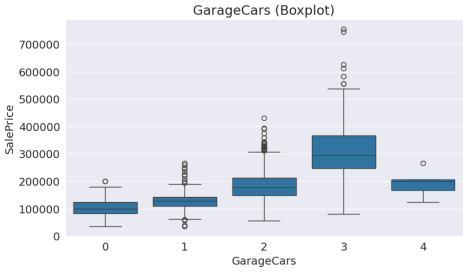

Fireplaces (Boxplot)
SalePrice


Fireplaces (Barplot)
count

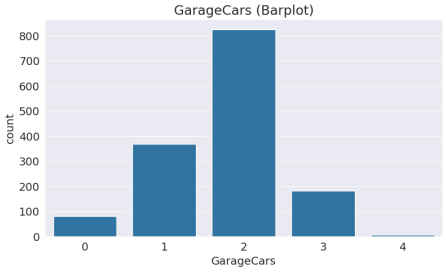




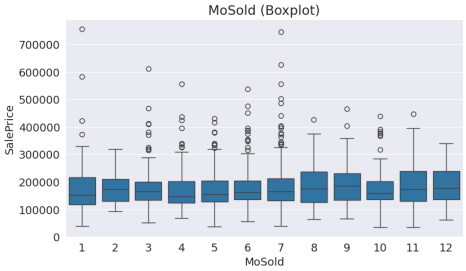
GarageCars (Boxplot)



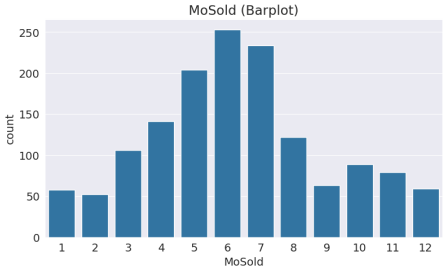
GarageCars (Barplot)



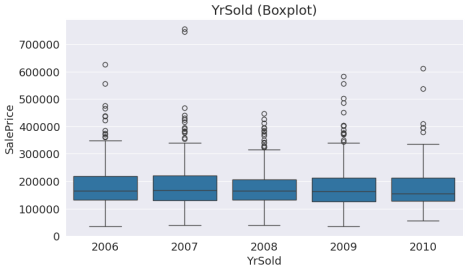
MoSold (Boxplot)



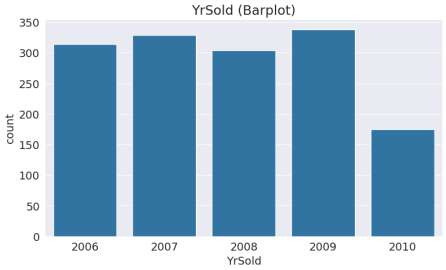
MoSold (Barplot)



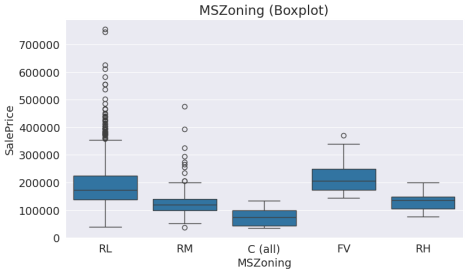
YrSold (Boxplot)



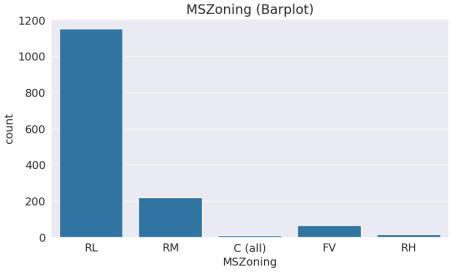
YrSold (Barplot)



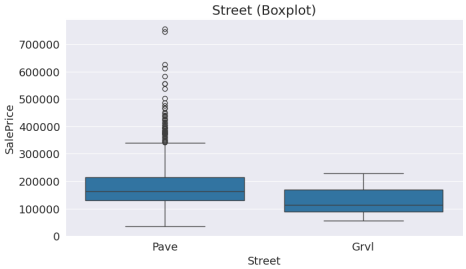
MSZoning (Boxplot)



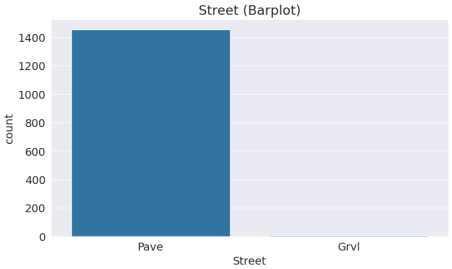
MSZoning (Barplot)



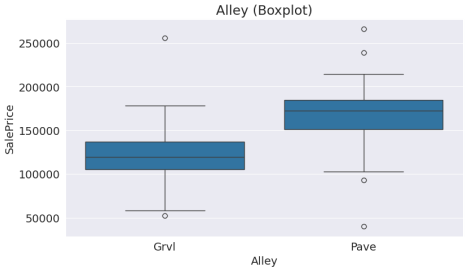
Street (Boxplot)



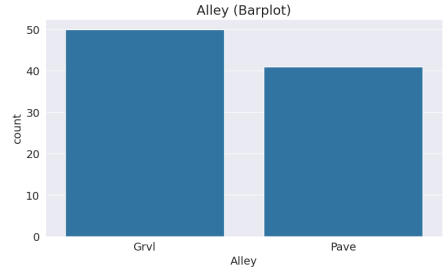
Street (Barplot)



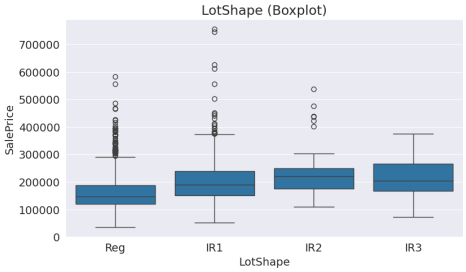
Alley (Boxplot)



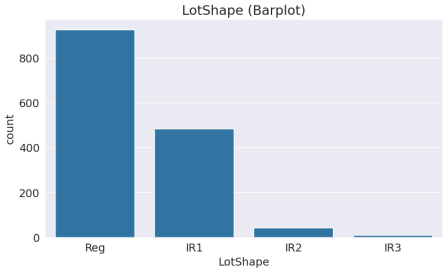
Alley (Barplot)



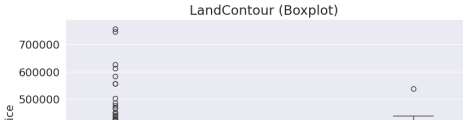
LotShape (Boxplot)



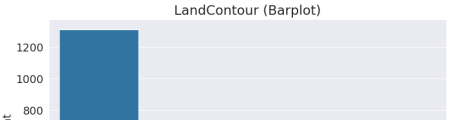
LotShape (Barplot)

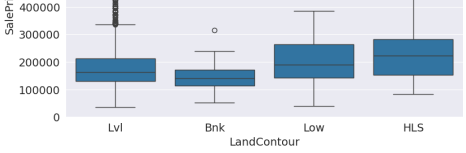


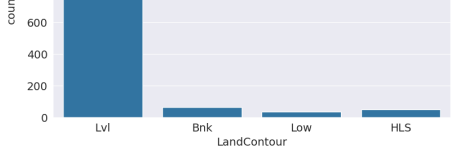
LandContour (Boxplot)



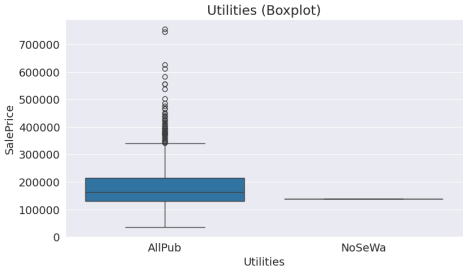
LandContour (Barplot)



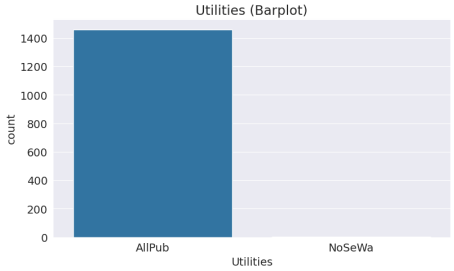




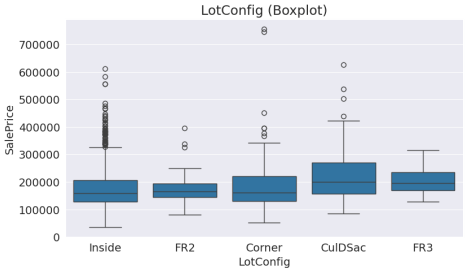
Utilities (Boxplot)



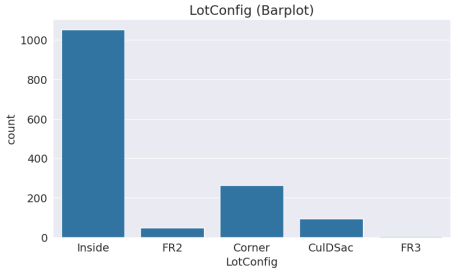
Utilities (Barplot)



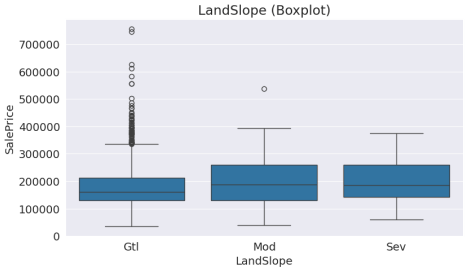
LotConfig (Boxplot)



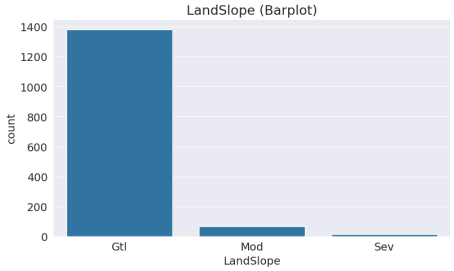
LotConfig (Barplot)



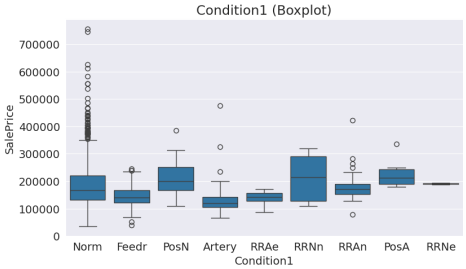
LandSlope (Boxplot)



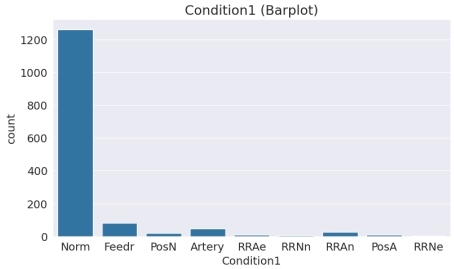
LandSlope (Barplot)



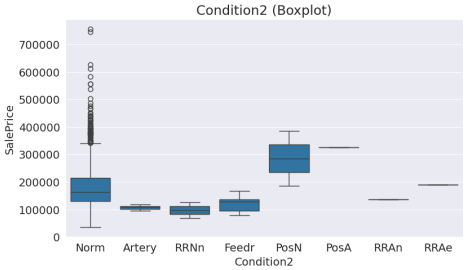
Condition1 (Boxplot)



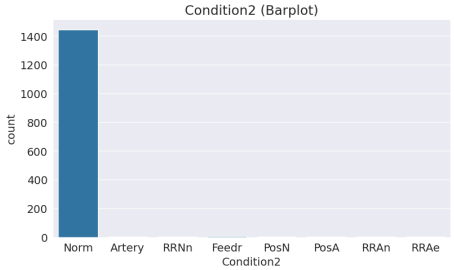
Condition1 (Barplot)



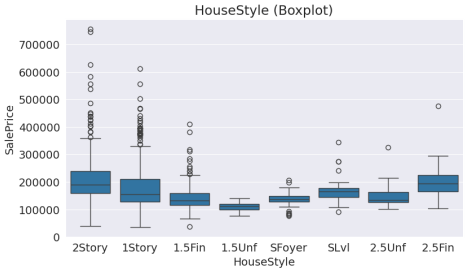
Condition2 (Boxplot)



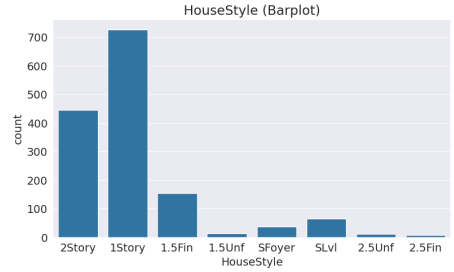
Condition2 (Barplot)



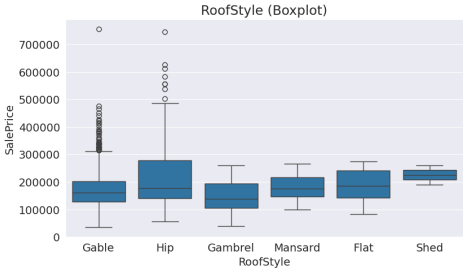
HouseStyle (Boxplot)



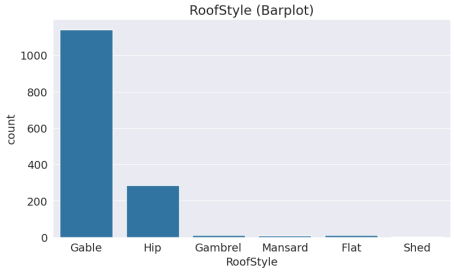
HouseStyle (Barplot)



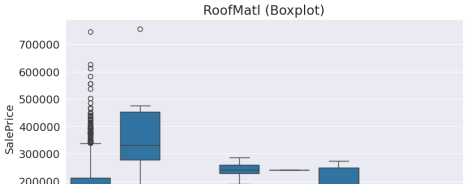
RoofStyle (Boxplot)



RoofStyle (Barplot)

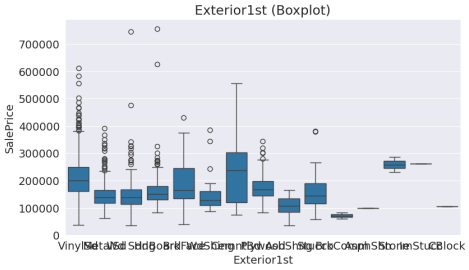


RoofMatl (Boxplot)

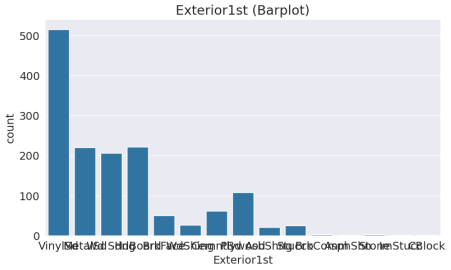


RoofMatl (Barplot)

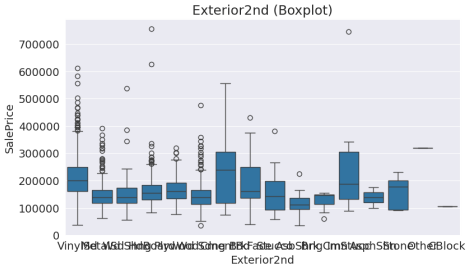
Exterior1st (Boxplot)



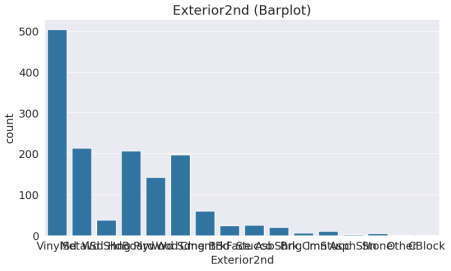
Exterior1st (Barplot)



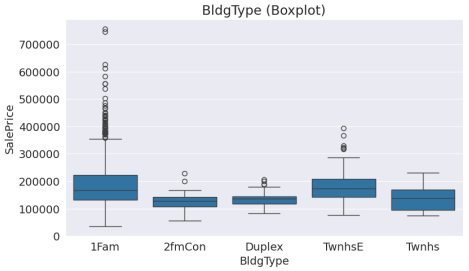
Exterior2nd (Boxplot)



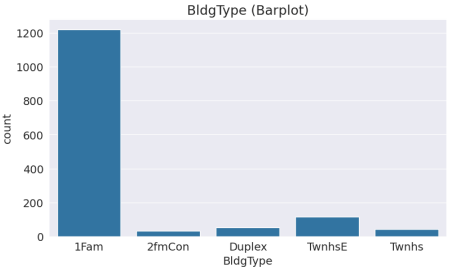
Exterior2nd (Barplot)



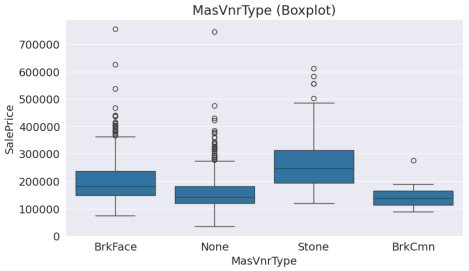
BldgType (Boxplot)



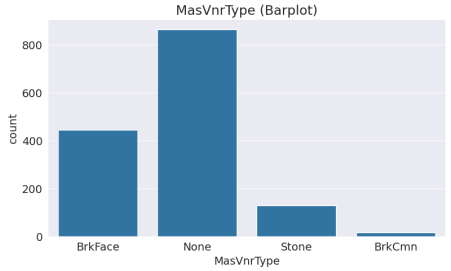
BldgType (Barplot)



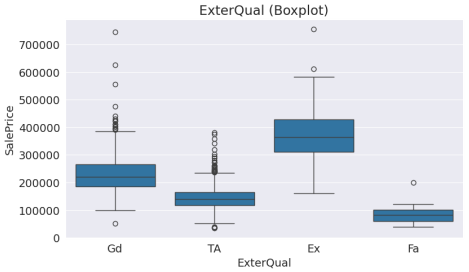
MasVnrType (Boxplot)



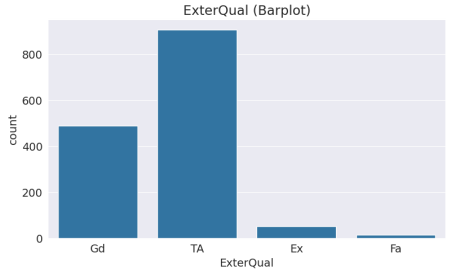
MasVnrType (Barplot)



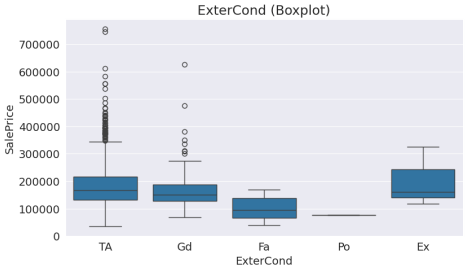
ExterQual (Boxplot)



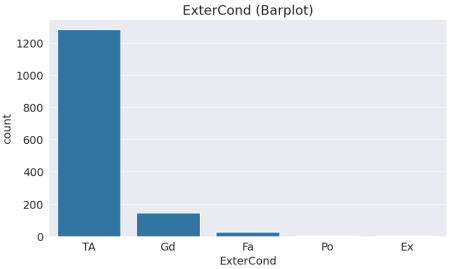
ExterQual (Barplot)



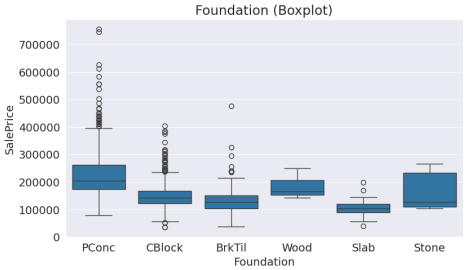
ExterCond (Boxplot)



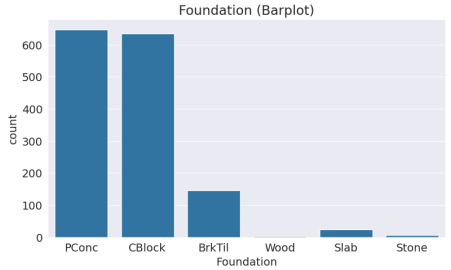
ExterCond (Barplot)



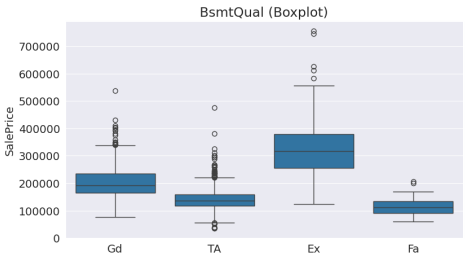
Foundation (Boxplot)



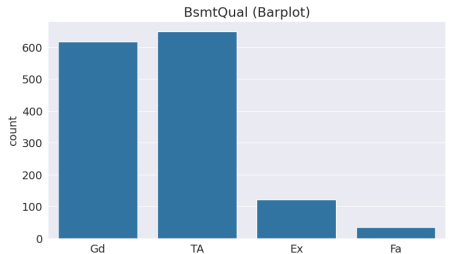
Foundation (Barplot)



BsmtQual (Boxplot)



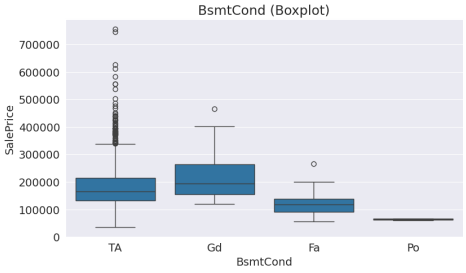
BsmtQual (Barplot)



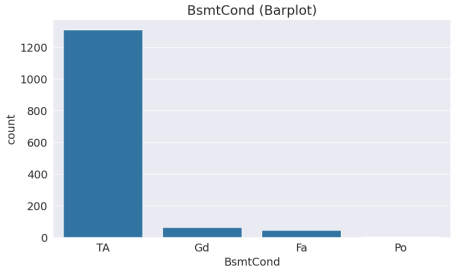
BsmtQual

BsmtQual

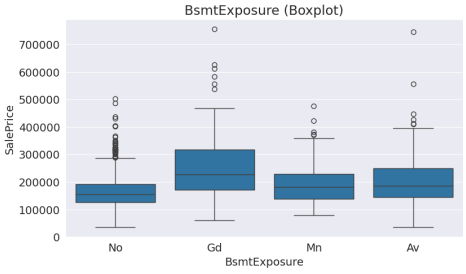
BsmtCond (Boxplot)



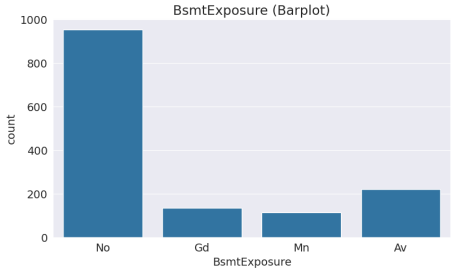
BsmtCond (Barplot)



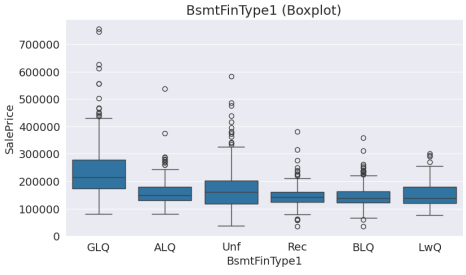
BsmtExposure (Boxplot)



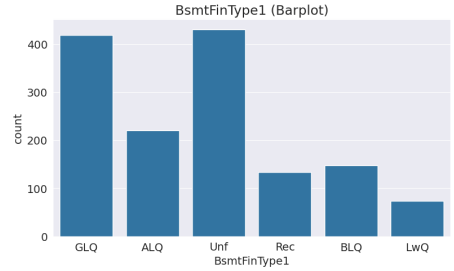
BsmtExposure (Barplot)



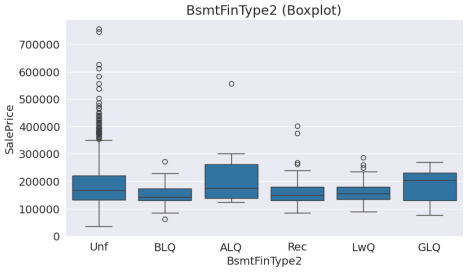
BsmtFinType1 (Boxplot)



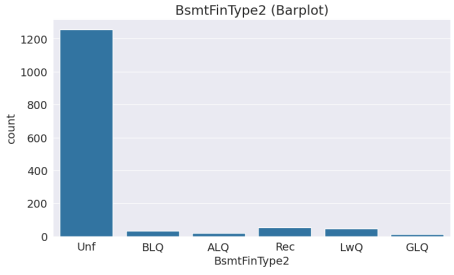
BsmtFinType1 (Barplot)



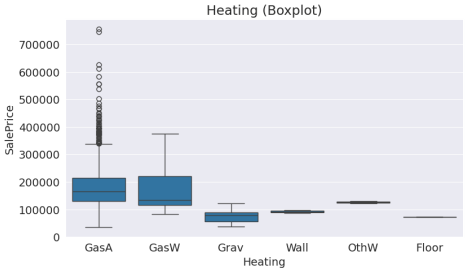
BsmtFinType2 (Boxplot)



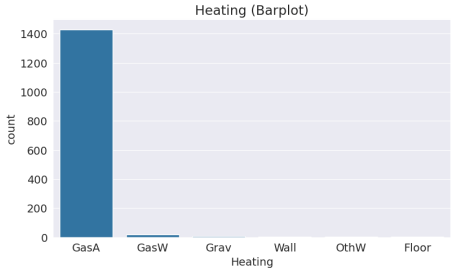
BsmtFinType2 (Barplot)



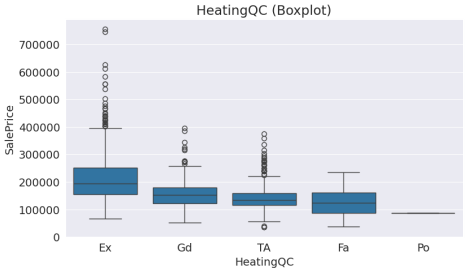
Heating (Boxplot)



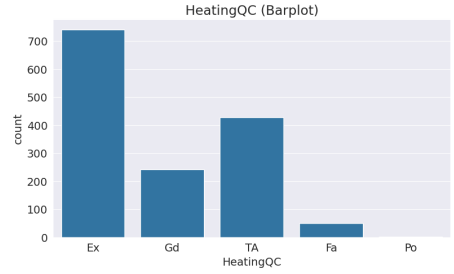
Heating (Barplot)



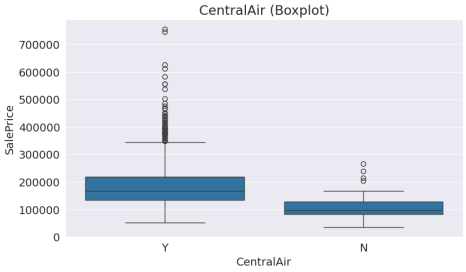
HeatingQC (Boxplot)



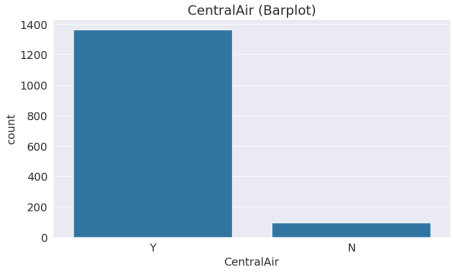
HeatingQC (Barplot)



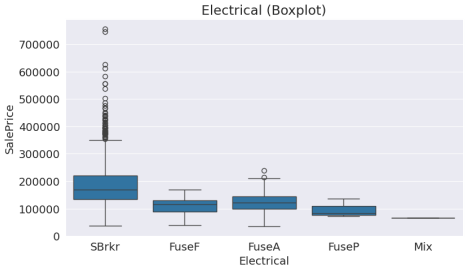
CentralAir (Boxplot)



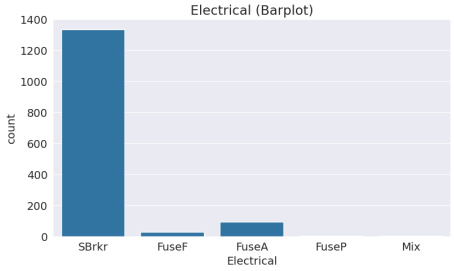
CentralAir (Barplot)



Electrical (Boxplot)

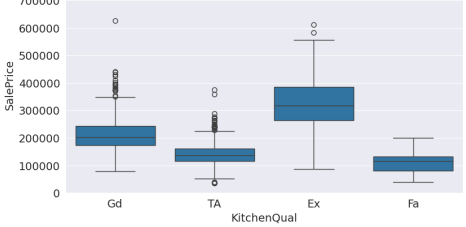


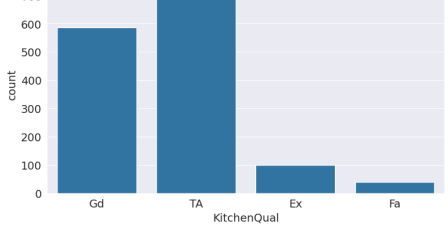
Electrical (Barplot)



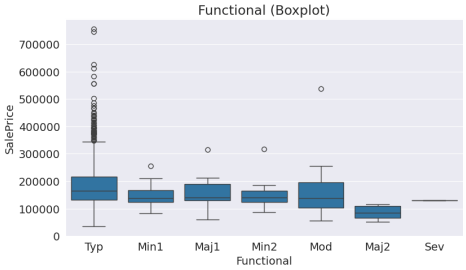
KitchenQual (Boxplot)

KitchenQual (Barplot)

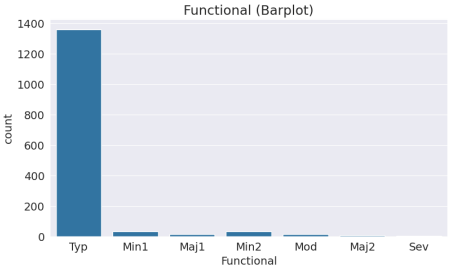




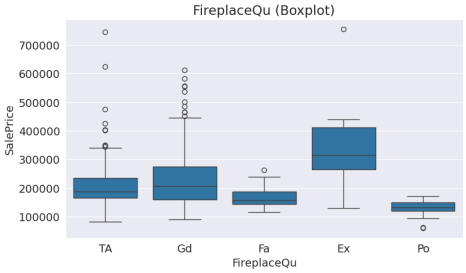
Functional (Boxplot)



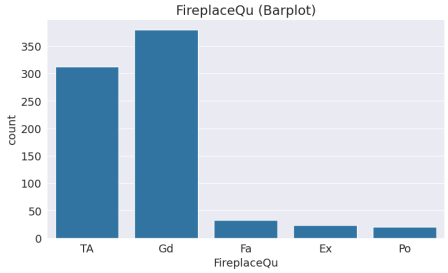
Functional (Barplot)



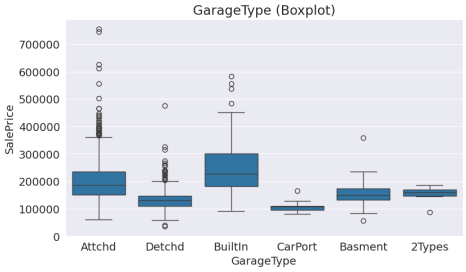
FireplaceQu (Boxplot)



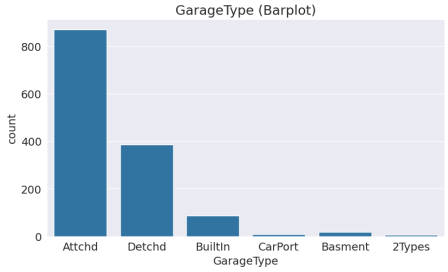
FireplaceQu (Barplot)



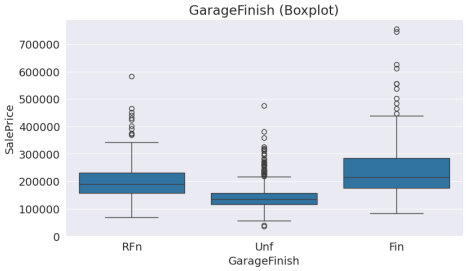
GarageType (Boxplot)



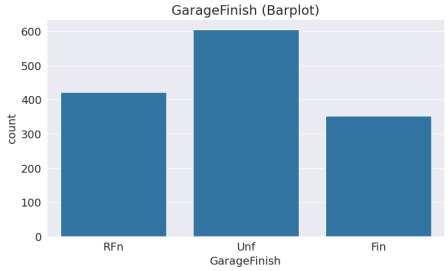
GarageType (Barplot)



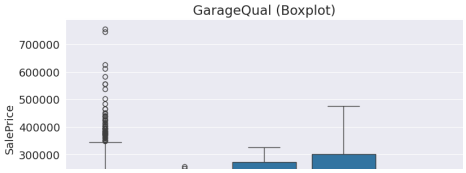
GarageFinish (Boxplot)



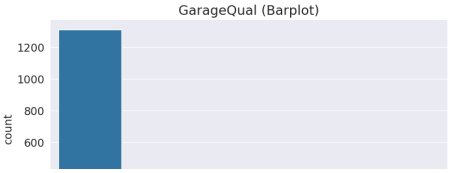
GarageFinish (Barplot)



GarageQual (Boxplot)



GarageQual (Barplot)

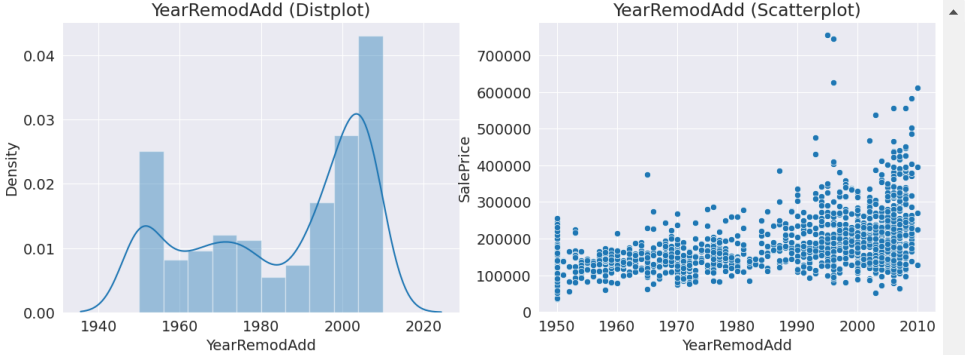


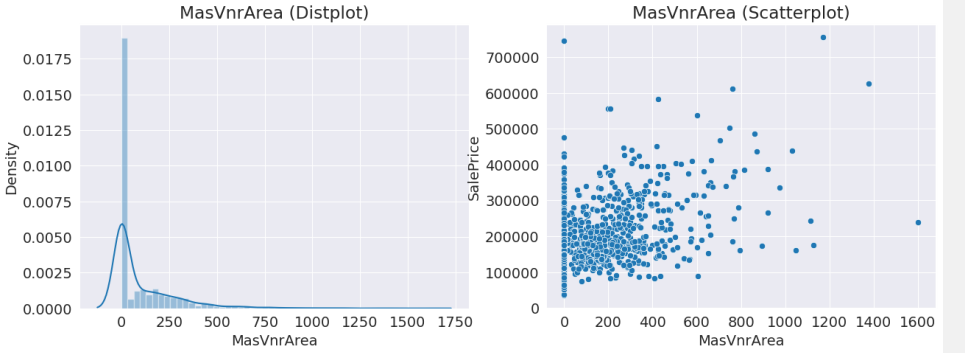
Here the countplot shows number of houses sold corresponding to the value of the attribute and the boxplot shows distribution of `SalePrice` corresponding to the value of the attribute. Observe that the number of houses sold corresponding to a value of the attribute can be less in number and still can cause wide distribution of corresponding `SalePrice`.

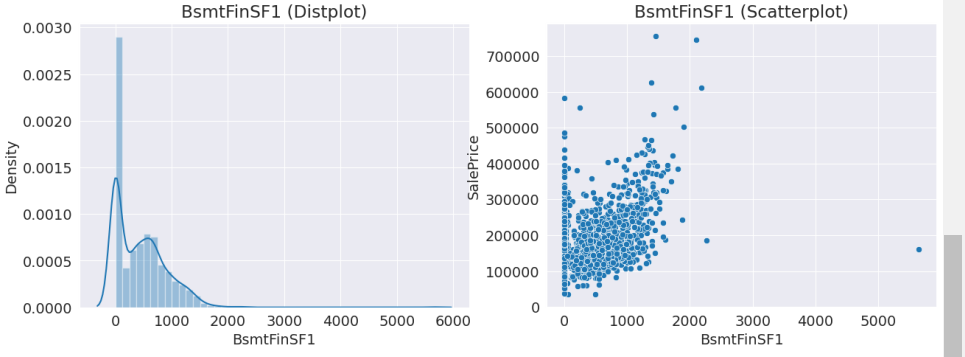
Now will find the distribution of numeric values while comparing the same attribute with SalePrice using scatterplot

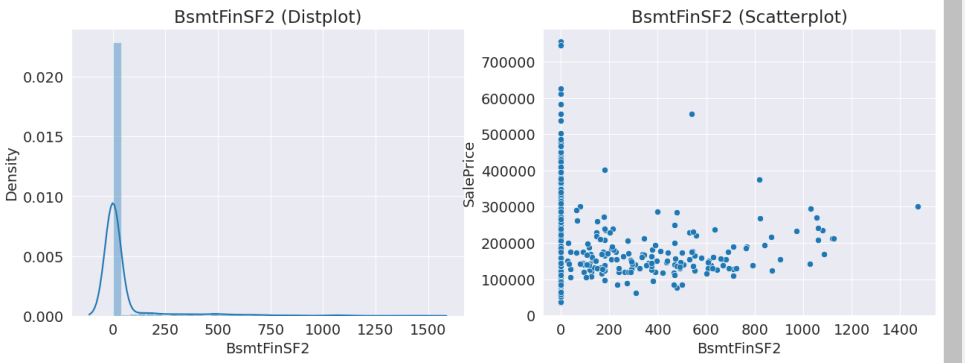
```
Numeric_Attributes=['LotFrontage','LotArea','YearBuilt',
 'YearRemodAdd', 'MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF',
 'LowQualFinSF', 'GrLivArea',   'GarageYrBlt', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch',
 '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal']
```

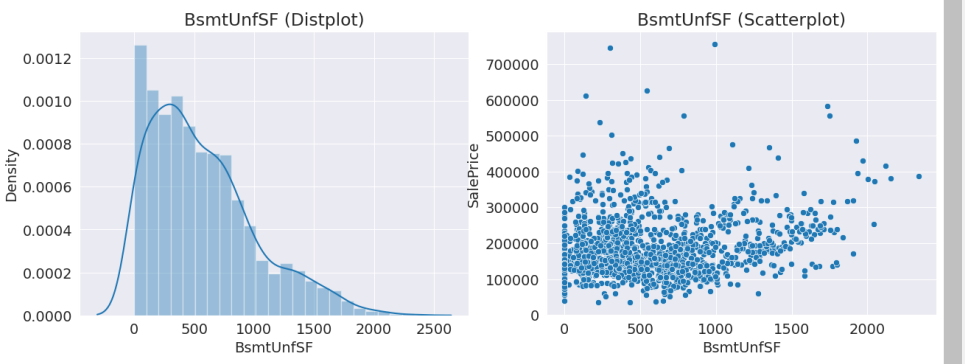
```
for i in Numeric_Attributes:
    fig, axes = plt.subplots(1,2, figsize=(15, 5))
    axes[0].set_title(i+' (Distplot)')
    sns.distplot(prices_df[i], ax=axes[0])
    axes[1].set_title(i+' (Scatterplot)')
    sns.scatterplot(y=prices_df['SalePrice'], x=prices_df[i], ax=axes[1])
```

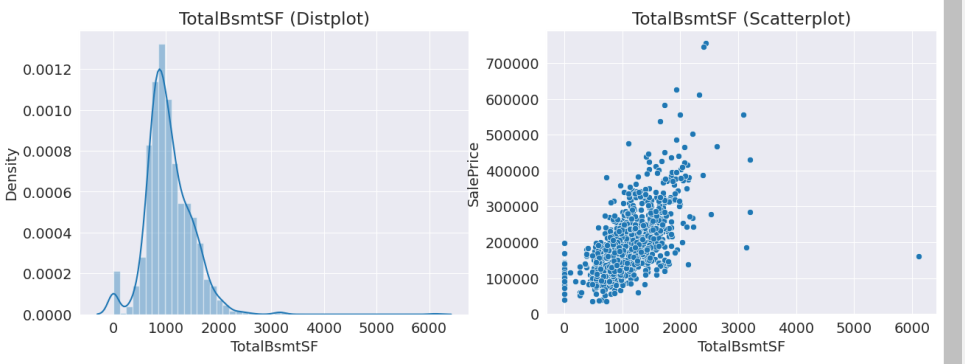

YearRemodAdd (Distplot)
Density
YearRemodAdd
YearRemodAdd (Scatterplot)
SalePrice
YearRemodAdd

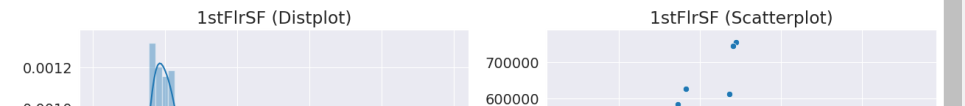

MasVnrArea (Distplot)
Density
MasVnrArea
MasVnrArea (Scatterplot)
SalePrice
MasVnrArea

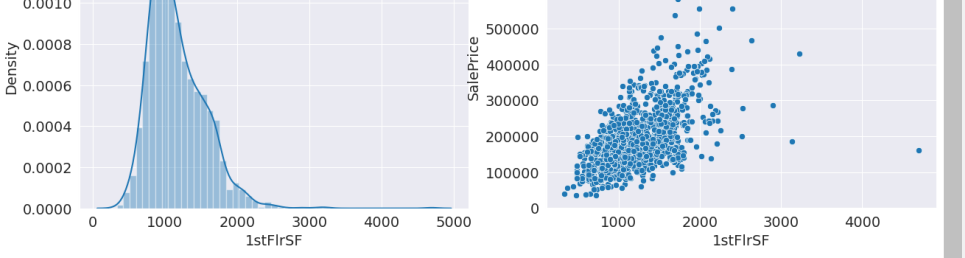

BsmtFinSF1 (Distplot)
Density
BsmtFinSF1
BsmtFinSF1 (Scatterplot)
SalePrice
BsmtFinSF1

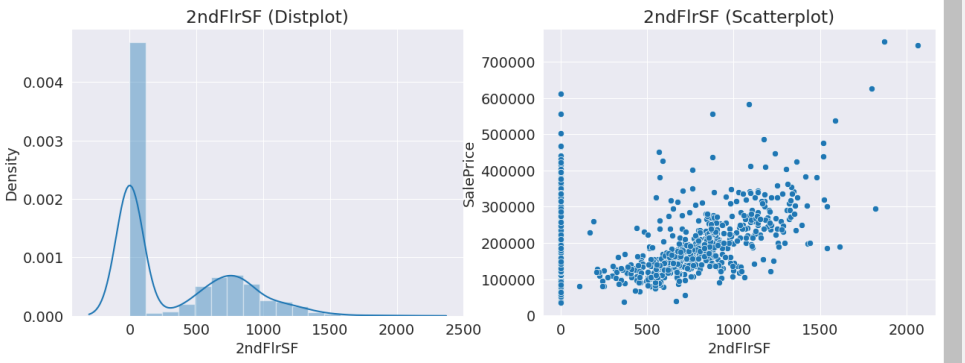

BsmtFinSF2 (Distplot)
Density
BsmtFinSF2
BsmtFinSF2 (Scatterplot)
SalePrice
BsmtFinSF2

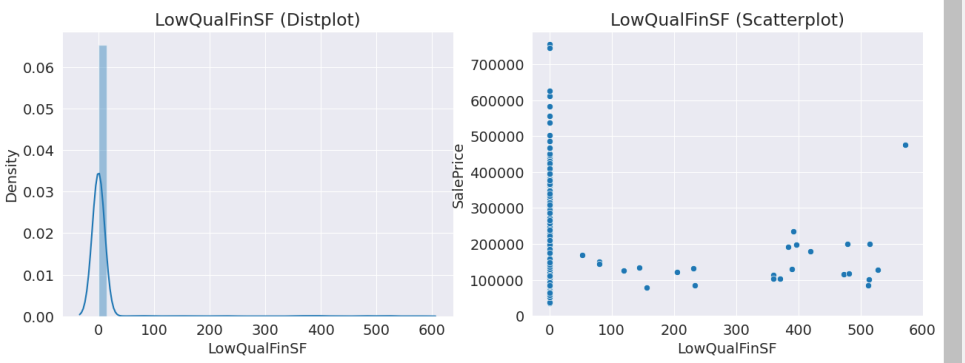

BsmtUnfSF (Distplot)
Density
BsmtUnfSF
BsmtUnfSF (Scatterplot)
SalePrice
BsmtUnfSF

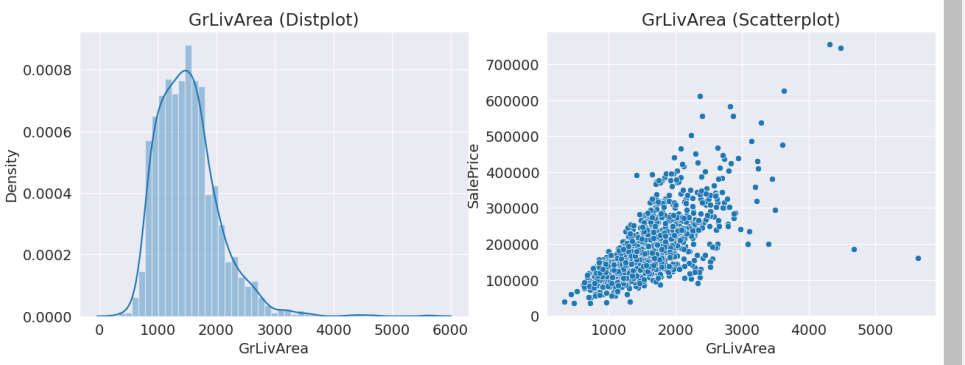

TotalBsmtSF (Distplot)
Density
TotalBsmtSF
TotalBsmtSF (Scatterplot)
SalePrice
TotalBsmtSF

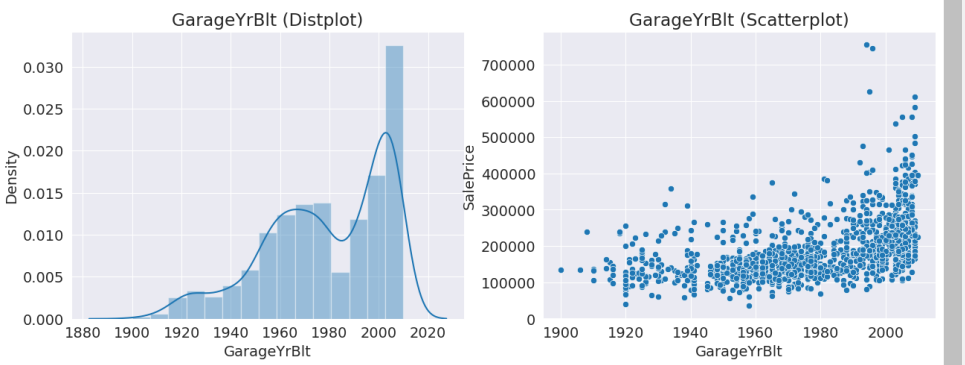

1stFlrSF (Distplot)
1stFlrSF (Scatterplot)

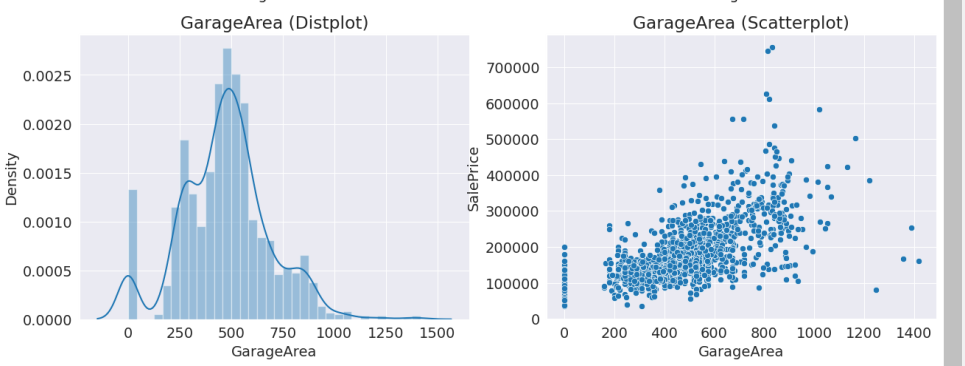

Density
1stFlrSF
SalePrice
1stFlrSF

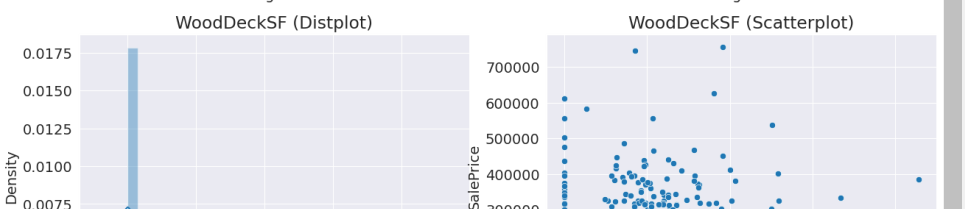

2ndFlrSF (Distplot)
2ndFlrSF (Scatterplot)
Density
2ndFlrSF
SalePrice
2ndFlrSF

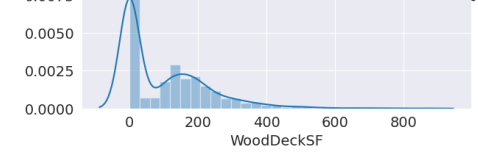

LowQualFinSF (Distplot)
LowQualFinSF (Scatterplot)
Density
LowQualFinSF
SalePrice
LowQualFinSF

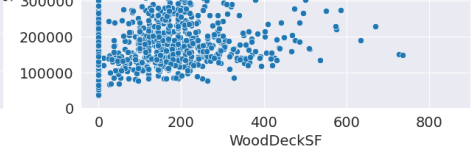

GrLivArea (Distplot)
GrLivArea (Scatterplot)
Density
GrLivArea
SalePrice
GrLivArea


GarageYrBlt (Distplot)
GarageYrBlt (Scatterplot)
Density
GarageYrBlt
SalePrice
GarageYrBlt

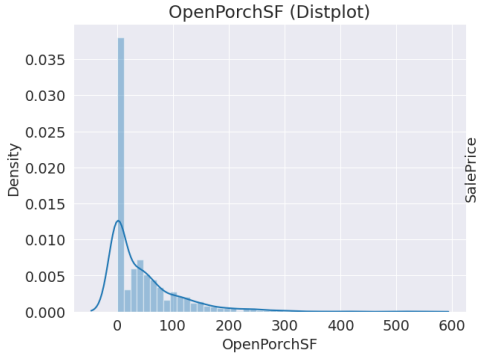

GarageArea (Distplot)
GarageArea (Scatterplot)
Density
GarageArea
SalePrice
GarageArea


WoodDeckSF (Distplot)
WoodDeckSF (Scatterplot)
Density
SalePrice

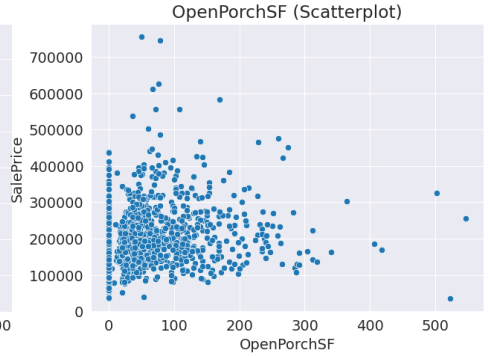




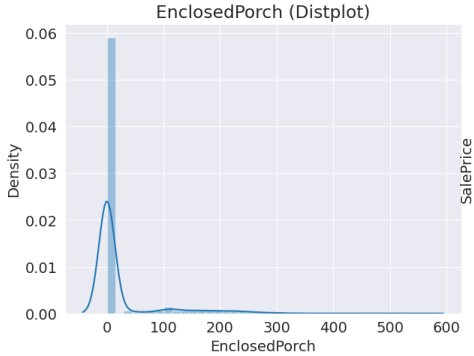
OpenPorchSF (Distplot)



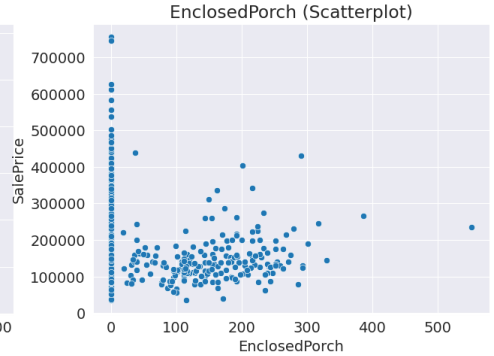
OpenPorchSF (Scatterplot)



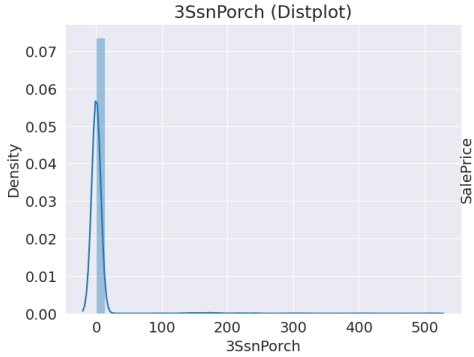
EnclosedPorch (Distplot)



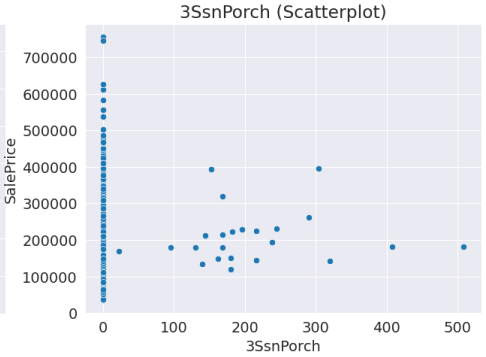
EnclosedPorch (Scatterplot)



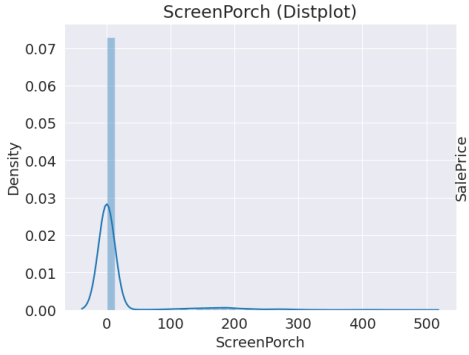
3SsnPorch (Distplot)



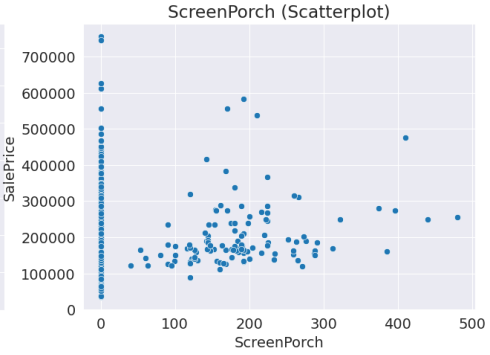
3SsnPorch (Scatterplot)



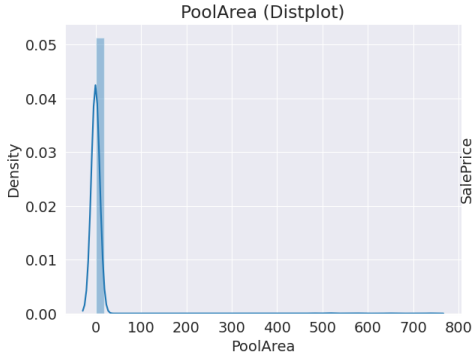
ScreenPorch (Distplot)



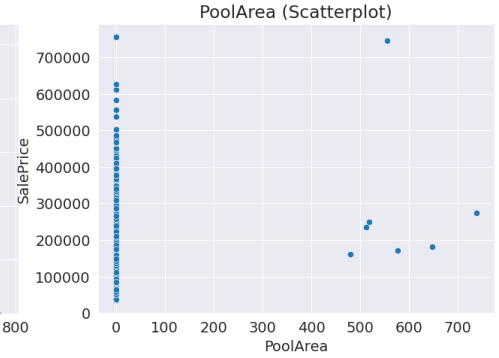
ScreenPorch (Scatterplot)



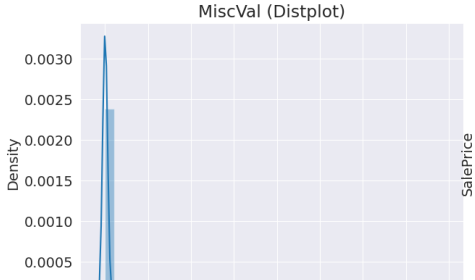
PoolArea (Distplot)



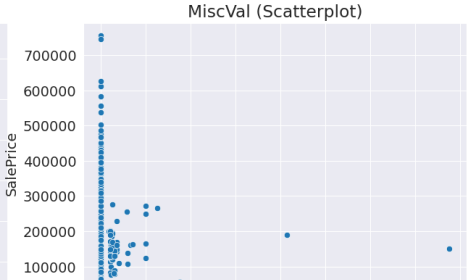
PoolArea (Scatterplot)



MiscVal (Distplot)



MiscVal (Scatterplot)




0.0000
0
2000
4000
6000
8000
10000
12000
14000
16000
MiscVal


0
0
2000
4000
6000
8000
10000
12000
14000
16000
MiscVal