FLIP ROBO

PFA Housing Project

Submitted by:

RaviPrasad Bathina

ACKNOWLEDGMENT

“House Price Prediction Using Machine Learning” by G. Naga Satish, Ch.V. Raghavendran, M.D.Sugnana Rao, Ch.Srinivasulu in the Year july 2019

This paper tells that Machine learning plays a major role from past years in image detection, spam reorganization, normal speech command, product recommendation and medical diagnosis. Present machine learning algorithm helps us in enhancing security alerts, ensuring public safety and improve medical enhancements. Machine learning system also provides better customer service and safer automobile systems. In this paper They discuss about the prediction of future housing prices that is generated by machine learning algorithm. For the selection of prediction methods we compare and explore various prediction methods. We utilize lasso regression as our model because of its adaptable and probabilistic methodology on model selection. Their result exhibit that their approach of the issue need to be successful, and has the ability to process predictions that would be comparative with other house cost prediction models. More over on other hand housing value indices, the advancement of a housing cost prediction that tend to the advancement of real estate policies schemes. This study utilizes machine learning algorithms as a research method that develops housing price prediction models. We create a housing cost prediction model In view of machine learning algorithm models for example, XGBoost, lasso regression and neural system on look at their order precision execution. We in that point recommend a housing cost prediction model to support a house vender or a real estate agent for better information based on the valuation of house. Those examinations exhibit that lasso regression algorithm, in view of accuracy, reliably outperforms alternate models in the execution of housing cost prediction.

“Predicting House Price With a Memristor-Based Artificial Neural Network” by J.J Wangi, S.G. Hu, X.T Zhani, Q.LUO1, Q.YU1, Zhen Liu, T.P Chen , Y.Yin, Sumio Hosaka, Y. Liu in the year 2018

This paper tells about that Synaptic memristor has attracted much attention for its potential applications in artificial neural networks (ANNs). However useful applications in real life with such memristor-based networks have seldom been reported. In this paper, an ANN based on memristors is designed to learn a multi-variable regression model with a back-propagation algorithm. A weight unit circuit based on memristor, which can be programed as an excitatory synapse or inhibitory synapse, is introduced. The weight of the electronic synapse is determined by the conductance of the memristor, and the current of the synapse follows the charge-dependent relationship. The ANN has the ability to learn from labelled samples and make predictions after online training. As an example, the ANN was used to learn a regression model of the house prices of several Boston towns in the USA and the predicted results are found to be close to the target data

"House Planning and Price Prediction System using Machine Learning" by Mr. Rushikesh Naikare, Mr. Girish Gahandule, Mr. Akash Dumbre, Mr. Kaushal Agrawal, Prof. Chaitanya Mankar in the year December 2019

The housing sector has hike as it is the one of the basic need. Housing the main domain of real estate. In the major metropolitan cities and the cities with many prestigious Educational institutions and IT Parks have reasonable price increase in housing. Home buying plans can derails the family's financial planning and other goals. Now a day's house price changing rapidly according to various parameters. The buyer gets confused in choosing his dream home as difference in price making it challenging. Both the buyer and seller should satisfy so they do not overestimate or underestimate price. So to build the platform where buyer can find home according to its needs and friendly to its financial condition. House price prediction on different parameters is our goal. Doing that we are going to use regression algorithms using machine learning on dataset so it can extract features from dataset. Result of this approach provide maximum efficiency and minimum errors. We also propose to determine the plane for house building.

"House Price Prediction Using Machine Learning and RPA" by Prof. Pradnya Patil Assistant Professor, Computer Engineering Department Technology, Darshil Shah, Harshad Rajput, Jay Chheda in the year March 2020

In today's world, everyone wishes for a house that suits their lifestyle and provides amenities according to their needs. House prices keep on changing very frequently which proves that house prices are often exaggerated. There are many factors that have to be taken into consideration for predicting house prices such as location, number of rooms, carpet area, how old the property is? and other basic local amenities. We will be using CatBoost algorithm along with Robotic Process Automation for real-time data extraction. Robotic Process Automation involves the use of software robots to automate the tasks of data extraction while machine learning algorithm is used to predict house prices with respect to the dataset.

INTRODUCTION

Problem statement:-

Houses are one of the necessary need of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies. Our problem is related to one such housing company.

A US-based housing company named **Surprise Housing** has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia.

The company is looking at prospective properties to buy houses to enter the market. You are required to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not

Exploratory Data Analysis

1. Checking the Missing Values
2. All the numerical Variables
3. Distribution of the Numerical Variables
4. categorical variables
5. cardinality of the categorical variables
6. Outliers
7. Relationship between dependent and independent feature(SalePrice)

1.Checking the missing values

Missing values in the dataset can be checked by below python code:-

```
## Here we will check the percentage of nan values present in each feature
## 1 -step make the list of features which has missing values

features_with_na=[features for features in df.columns if
df[features].isnull().sum()>1]

## 2- step print the feature name and the percentage of missing values

for feature in features_with_na:
    print(feature, np.round(df[feature].isnull().mean(), 4),  ' % missing values')
```

Observation:-

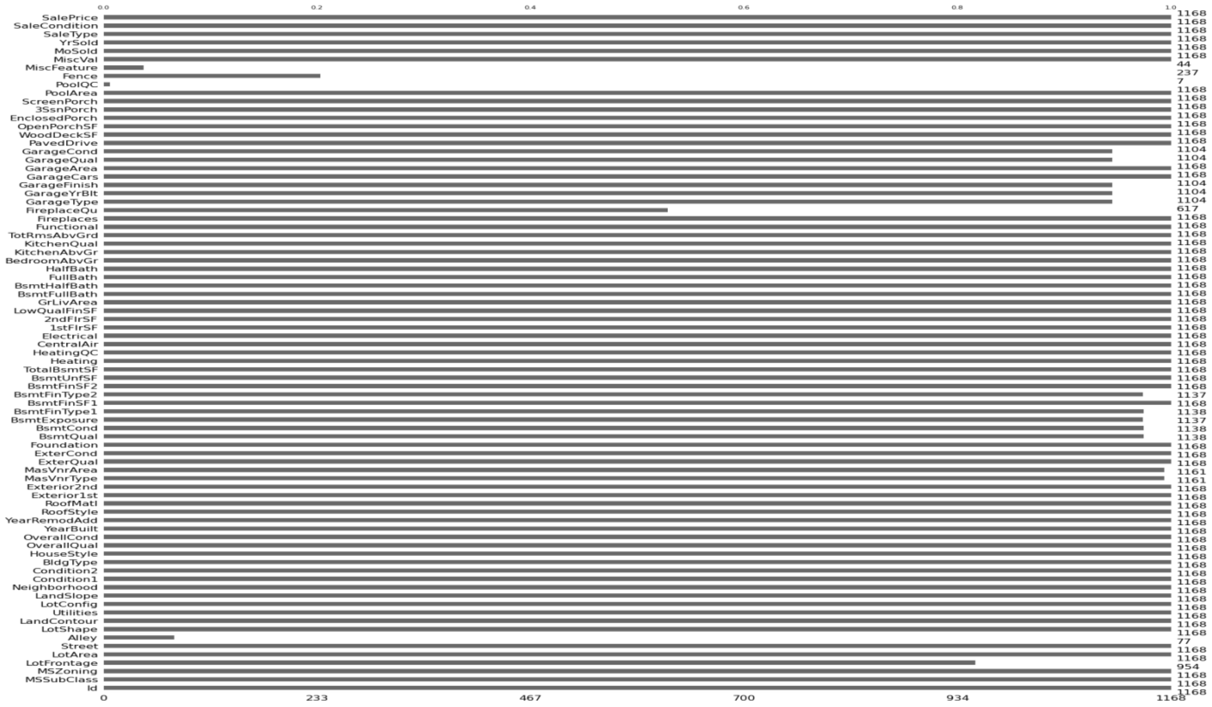
Number of missing variable columns: 18
Missing values in the dataset :

```
LotFrontage 0.1832  % missing values
Alley 0.9341  % missing values
MasVnrType 0.006  % missing values
MasVnrArea 0.006  % missing values
BsmtQual 0.0257  % missing values
BsmtCond 0.0257  % missing values
BsmtExposure 0.0265  % missing values
BsmtFinType1 0.0257  % missing values
BsmtFinType2 0.0265  % missing values
FireplaceQu 0.4717  % missing values
GarageType 0.0548  % missing values
GarageYrBlt 0.0548  % missing values
GarageFinish 0.0548  % missing values
```

GarageQual 0.0548 % missing values
GarageCond 0.0548 % missing values
PoolQC 0.994 % missing values
Fence 0.7971 % missing values
MiscFeature 0.9623 % missing values

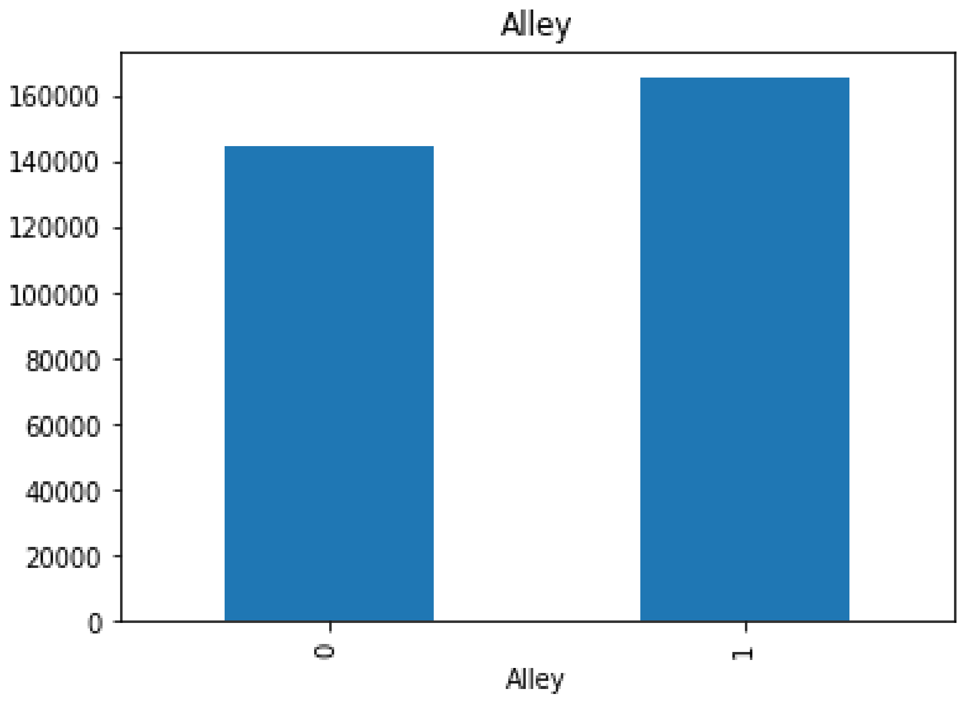
2. Checking the missing value with missingo library

1. There are many missing values in the columns of the dataset
2. Hence need to

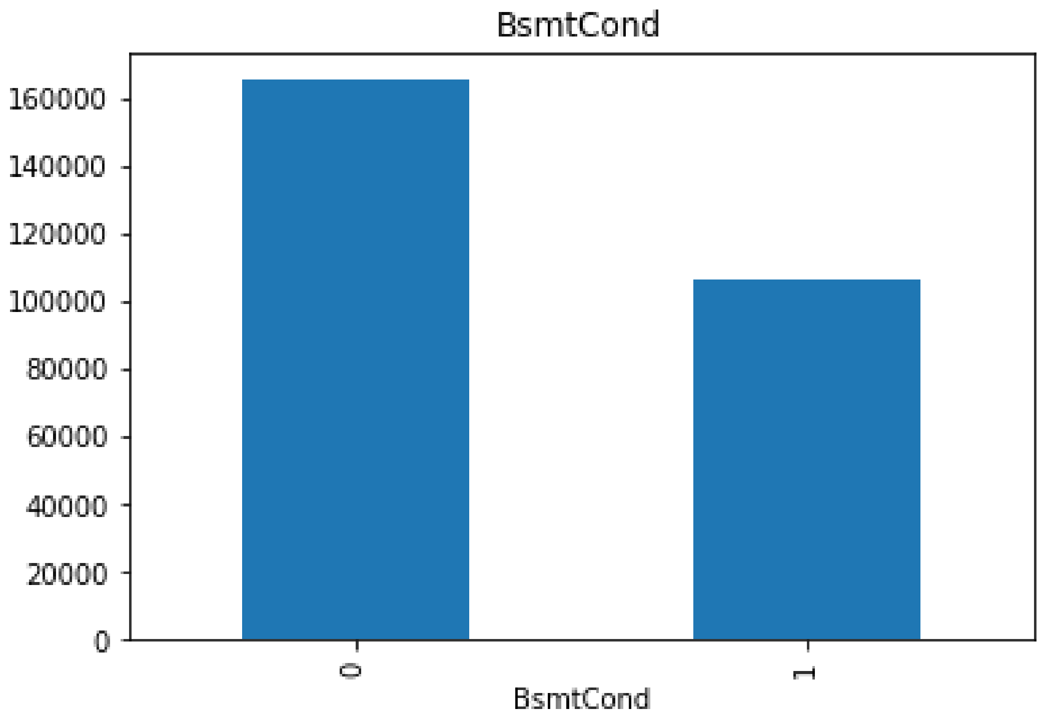


check the relationship with sales price

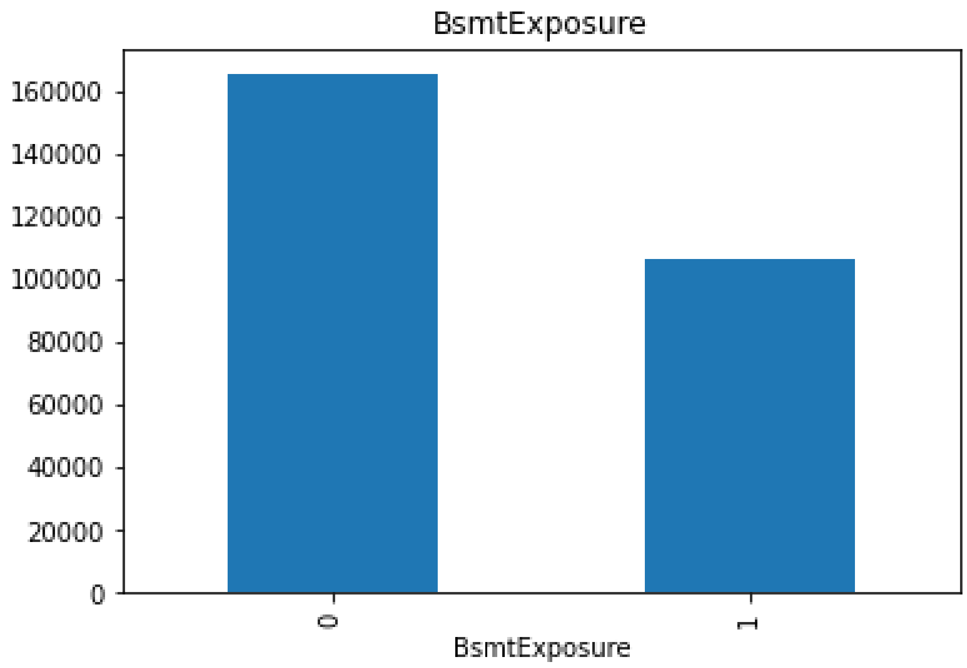
Representation of Missing values vs Salesprice



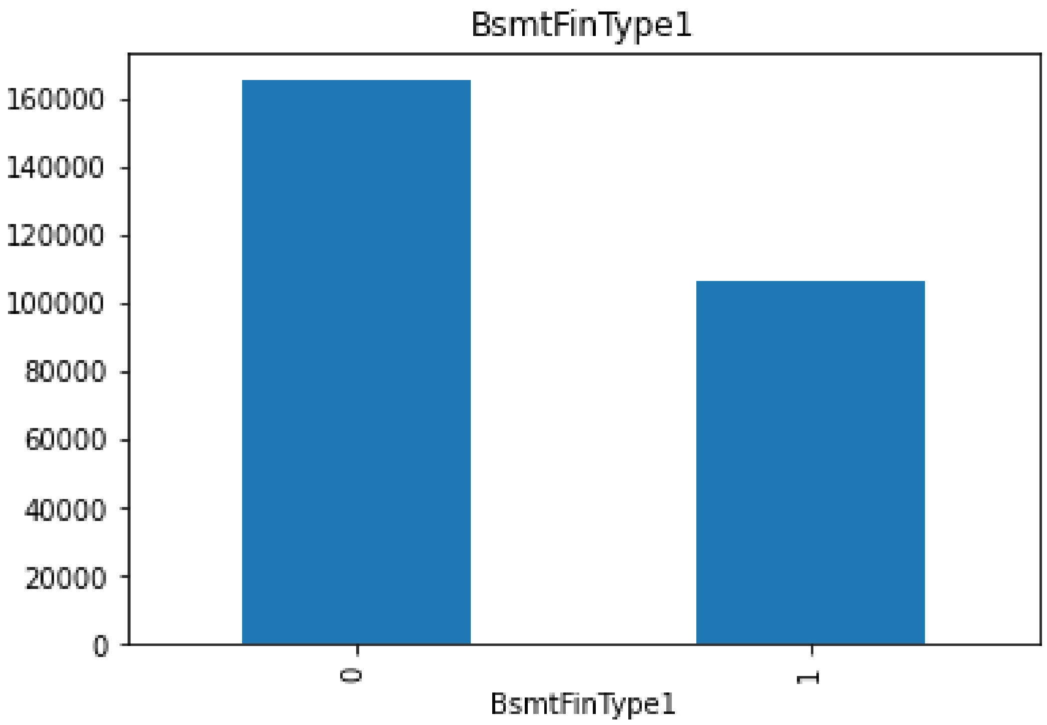
Missing value vs alley



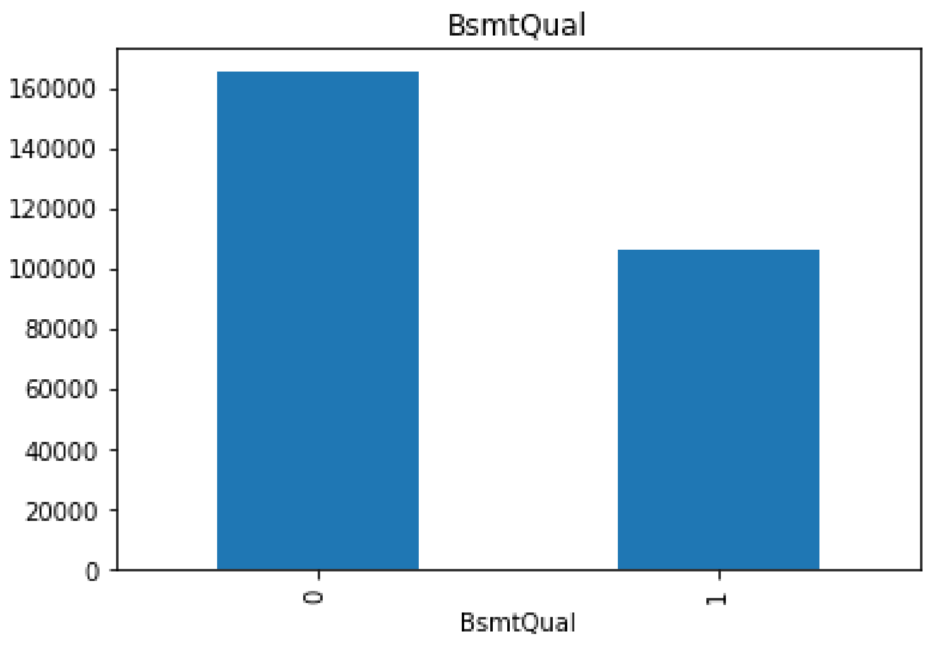
Missing value vs Bsmtcond



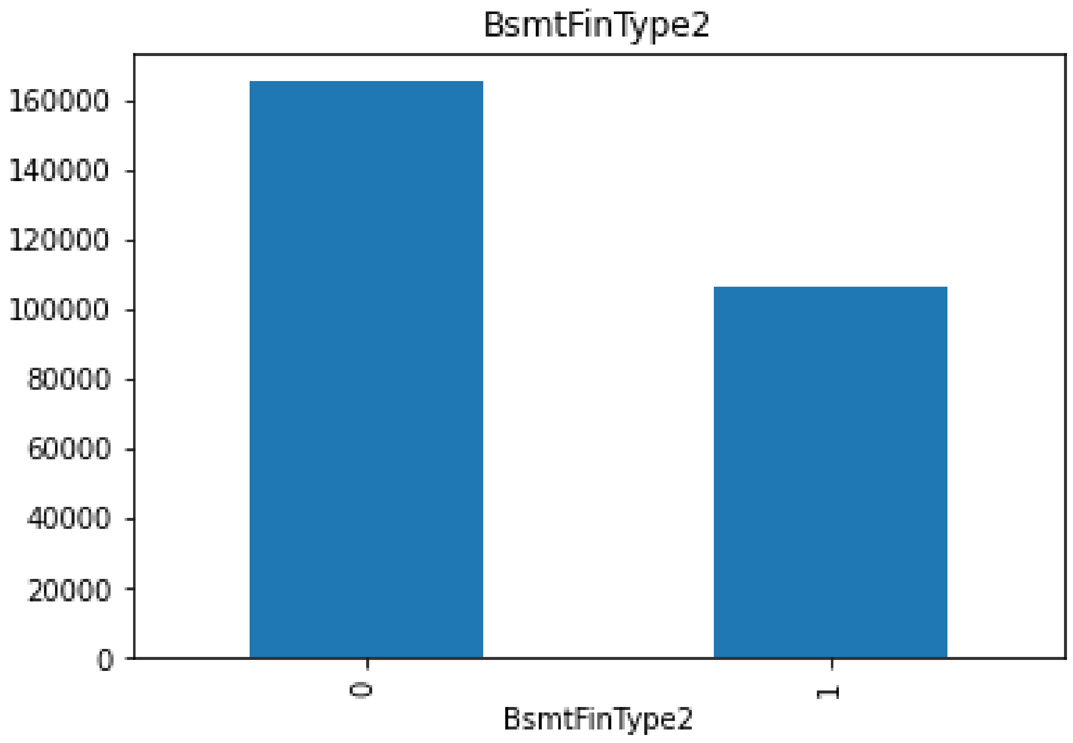
Missing value vs bsmt Exposure



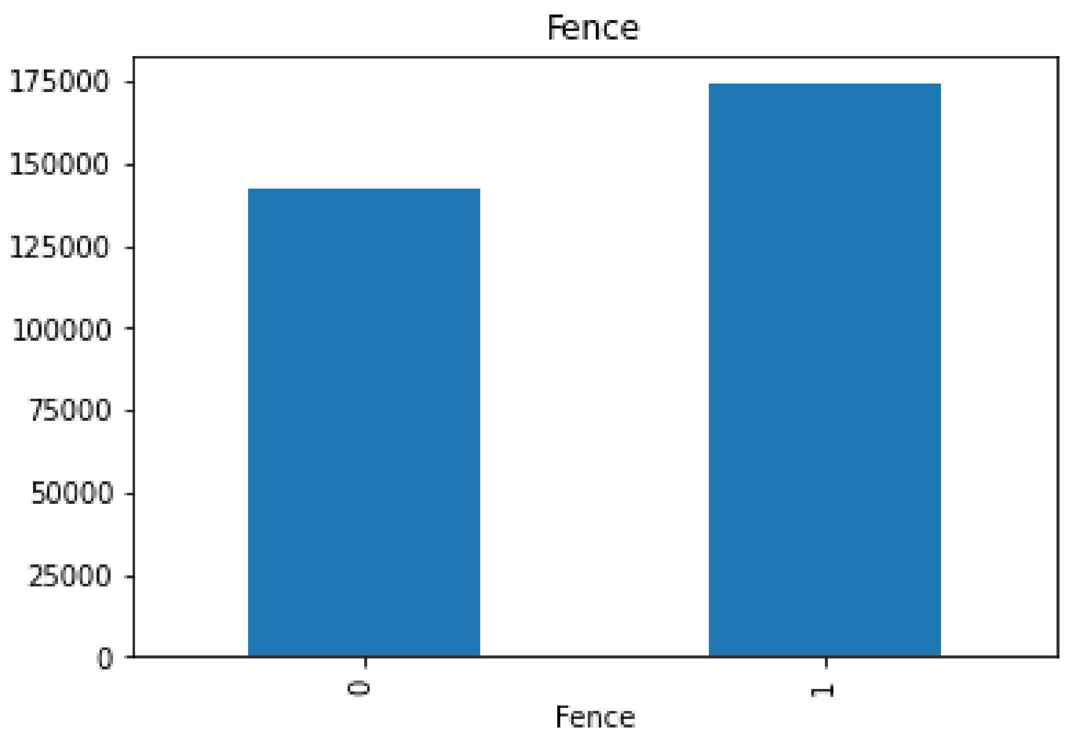
Missing value vs BsmtFintype 1



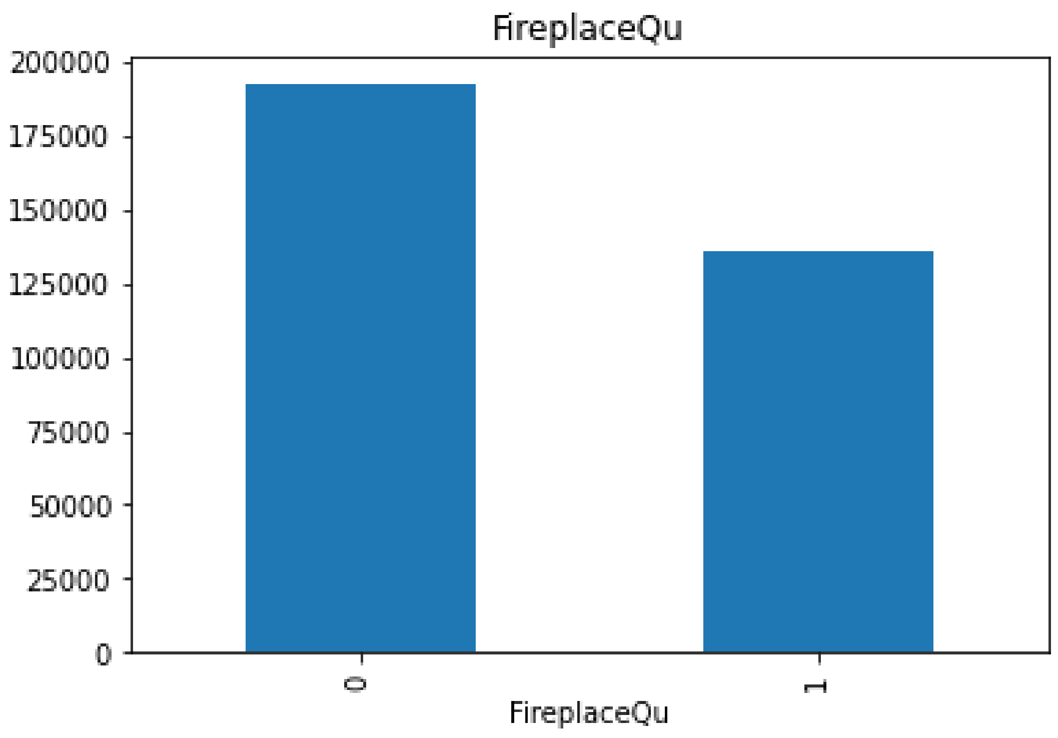
Missing value vs Bsmtqual



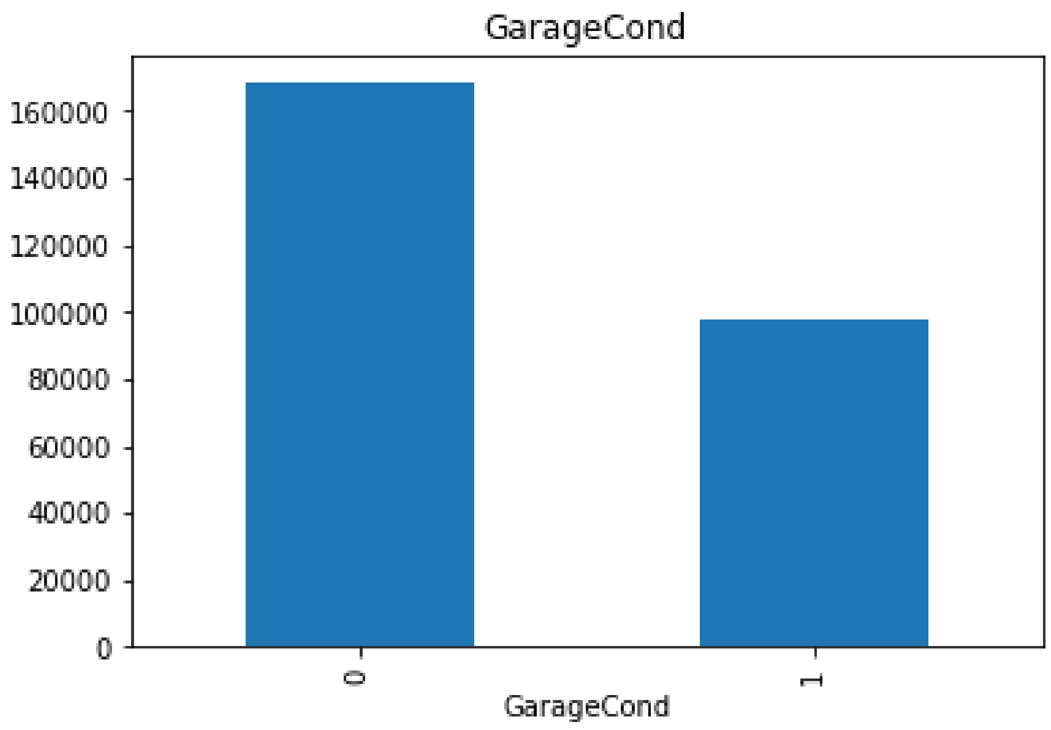
Missing value vs Bsmttype2



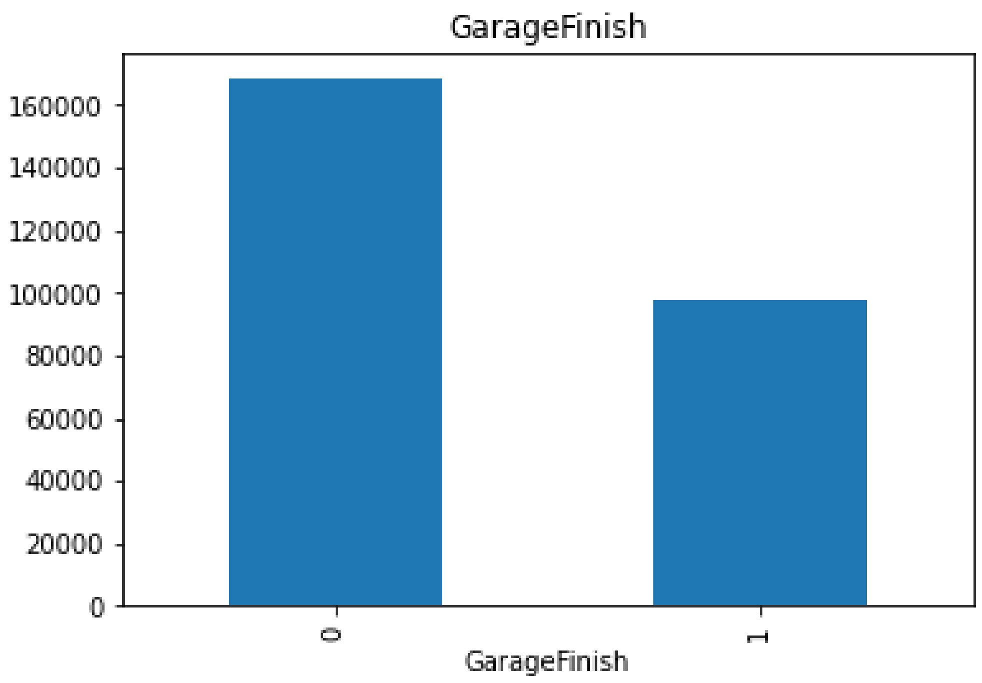
Missing value vs Fence



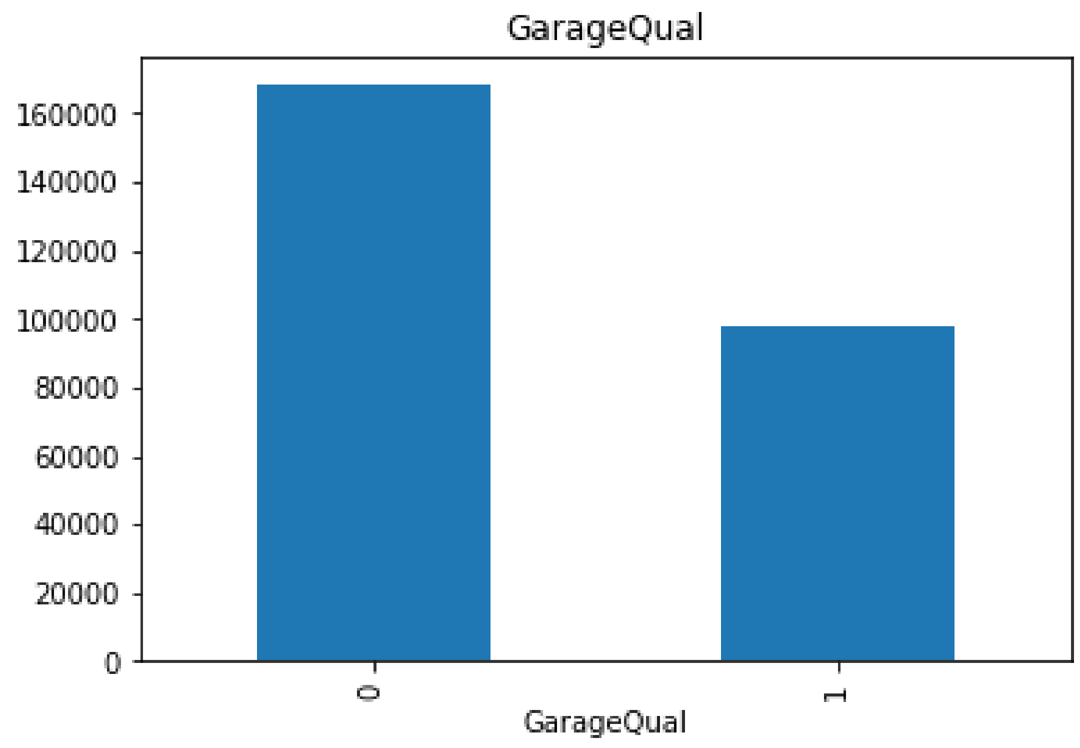
Missing value vs FireplaceQu



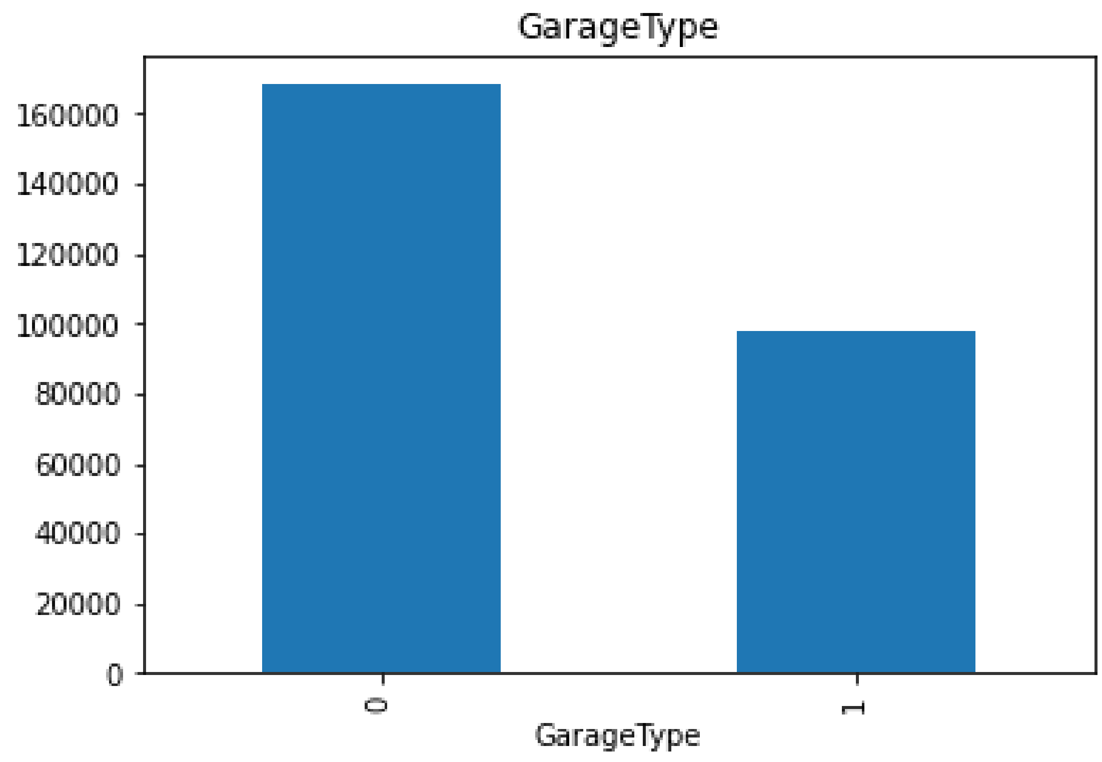
Missing value vs GarageCond



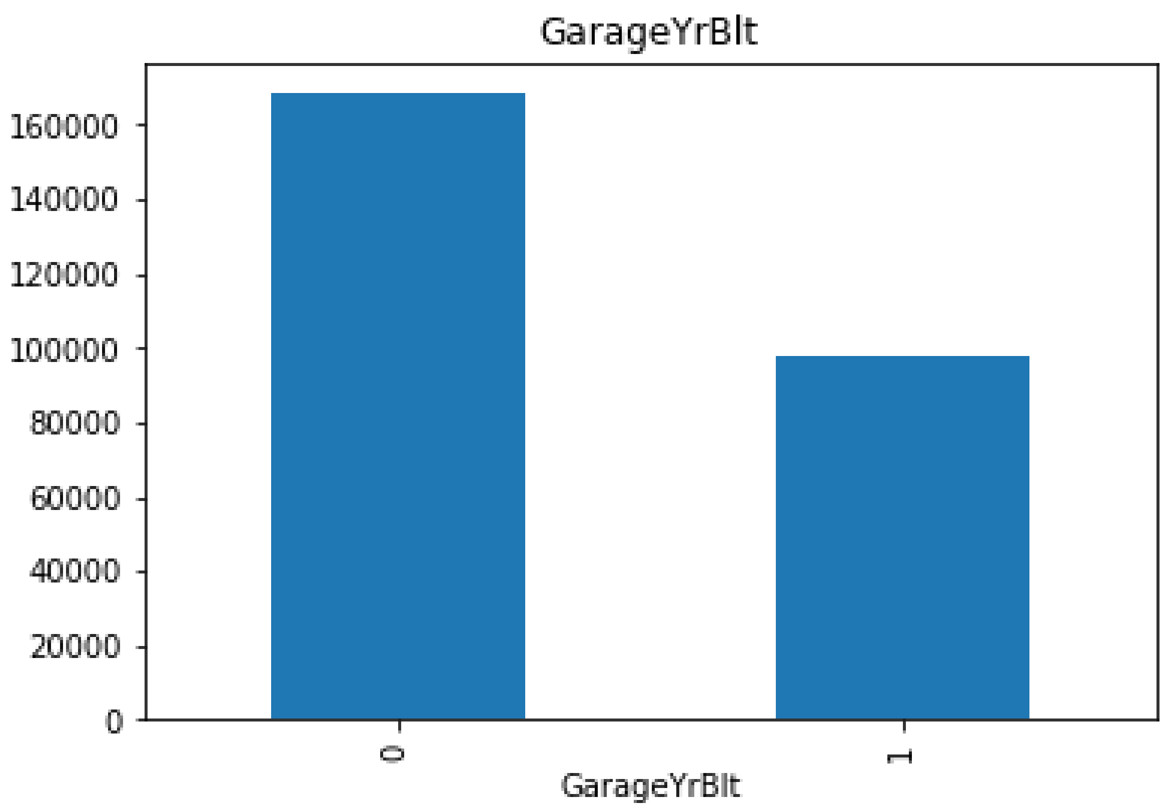
Missing value vs GarageFinish



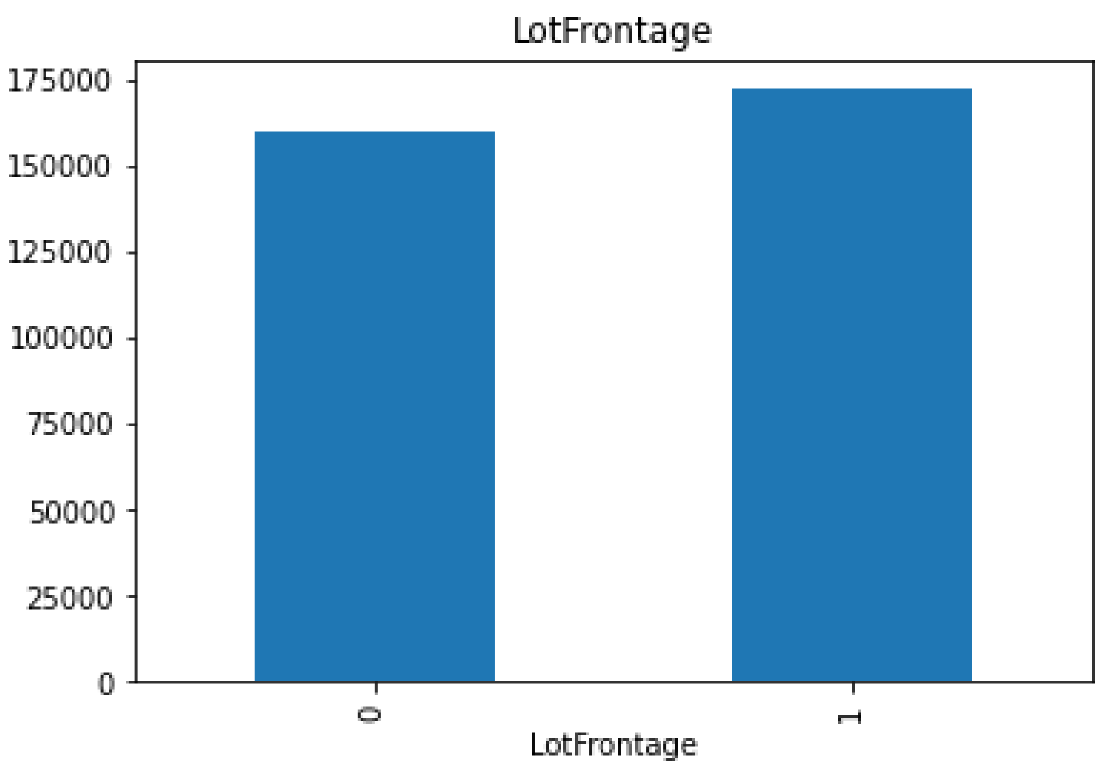
Missing value vs GarageQual



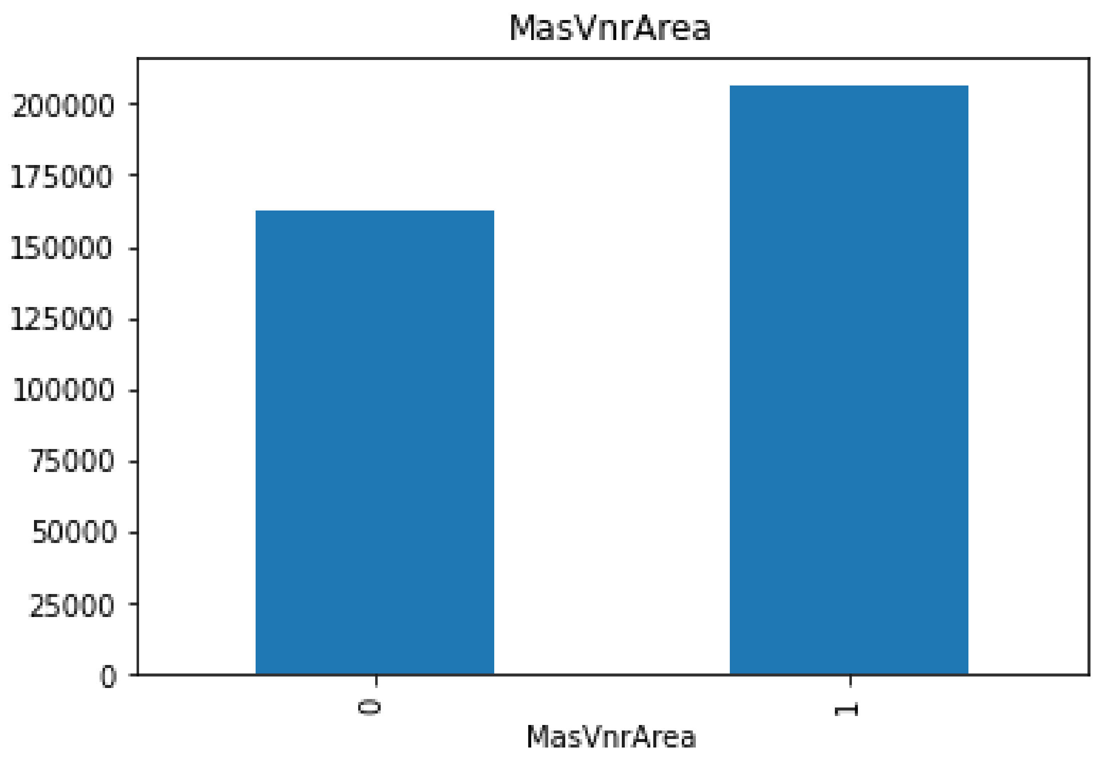
Missing value vs GarageType



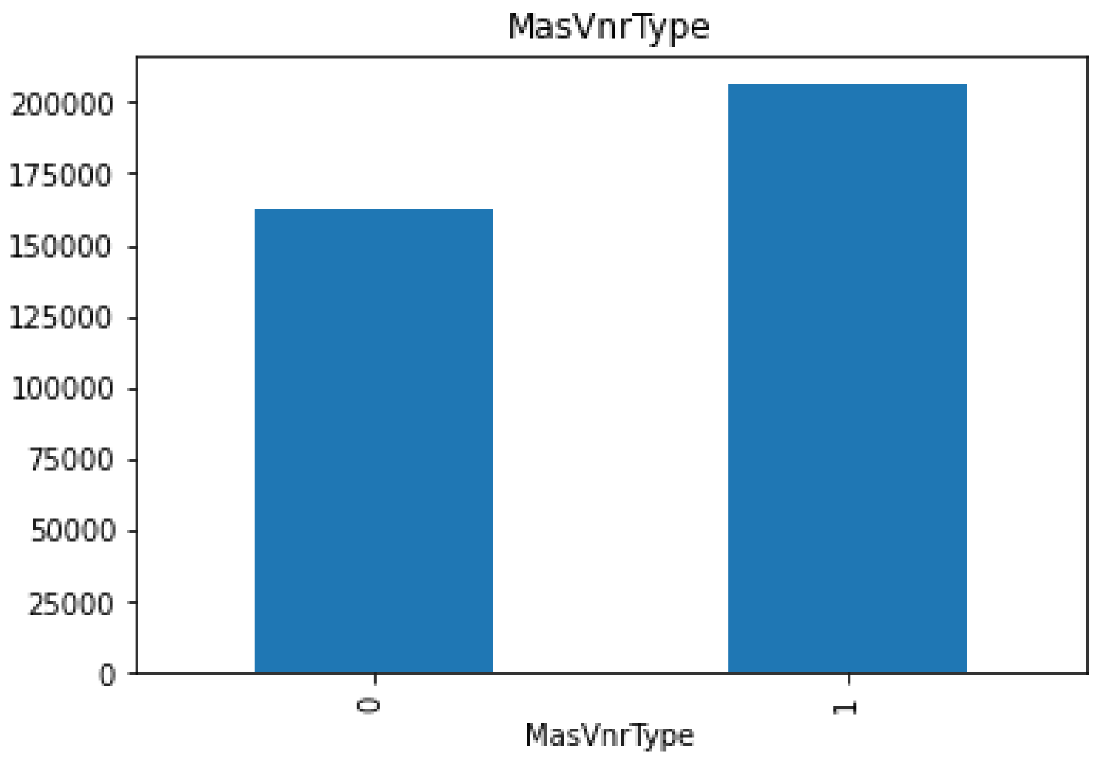
Missing value vs GarageYrBuilt



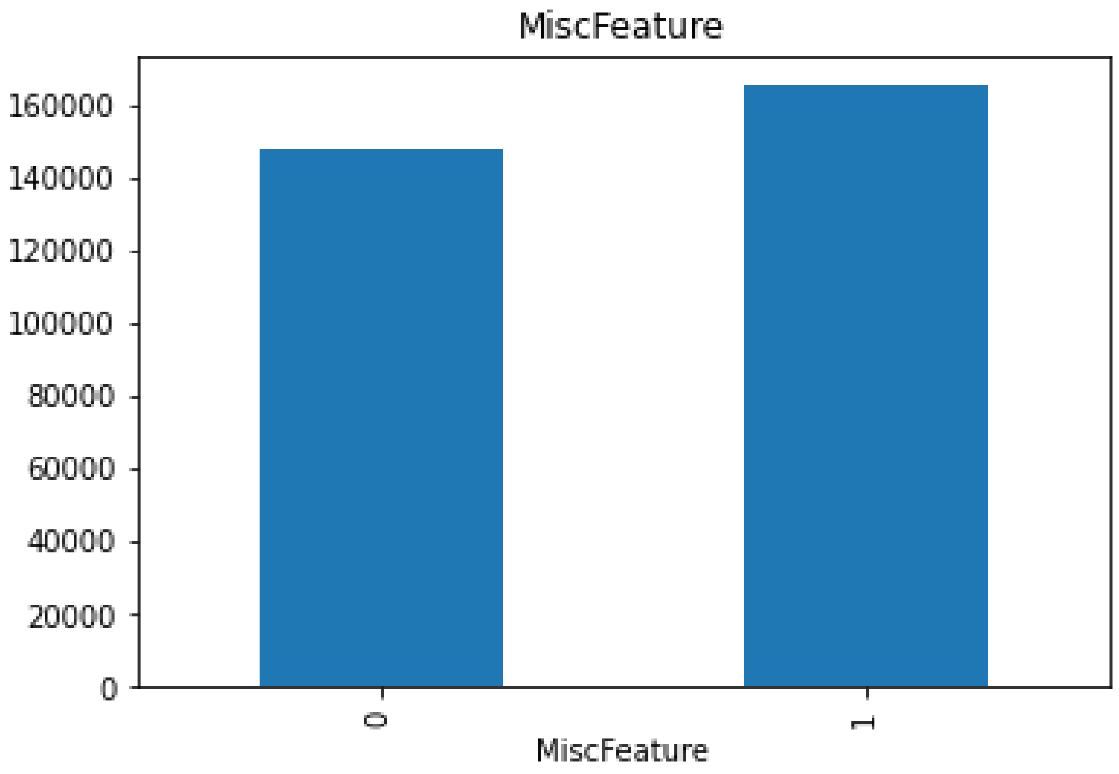
Missing value vs Lotfrontage



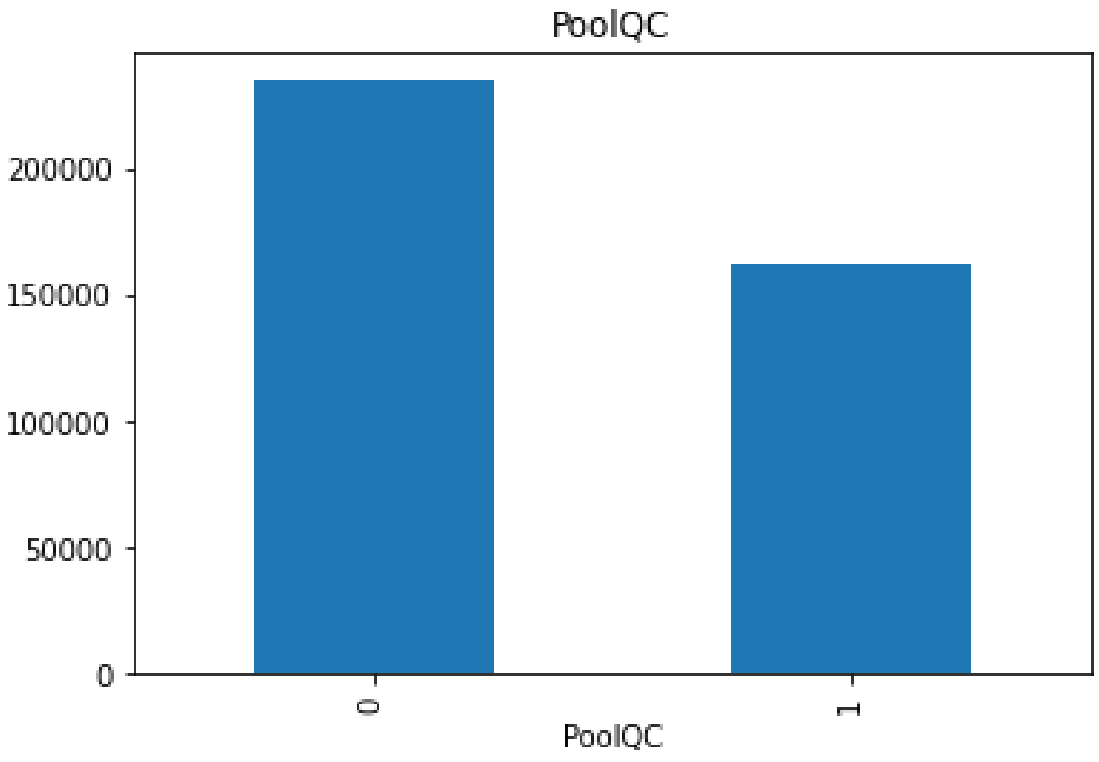
Missing value vs MassVnArea



Missing value vs MassVnrType



Missing value vs MiscFeature



Missing value vs PoolQC

Filling the missing values of Object and numeric columns.

```
#Filling the Object Columns
with mode.
for col in ['GarageType',
'GarageFinish', 'GarageQual',
'GarageCond','MasVnrType','
BsmtQual','BsmtCond','Bsmt
Exposure','BsmtFinType1','B
smtFinType2','Electrical']:
df[col].fillna(df[col].mode()[
0],inplace=True)
```

#Filling the Numeric columns with mean.

```
df['MasVnrArea'] = df['MasVnrArea'].fillna(df['MasVnrArea'].mean())
df['GarageYrBlt'] = df['GarageYrBlt'].fillna(df['GarageYrBlt'].mean())
```

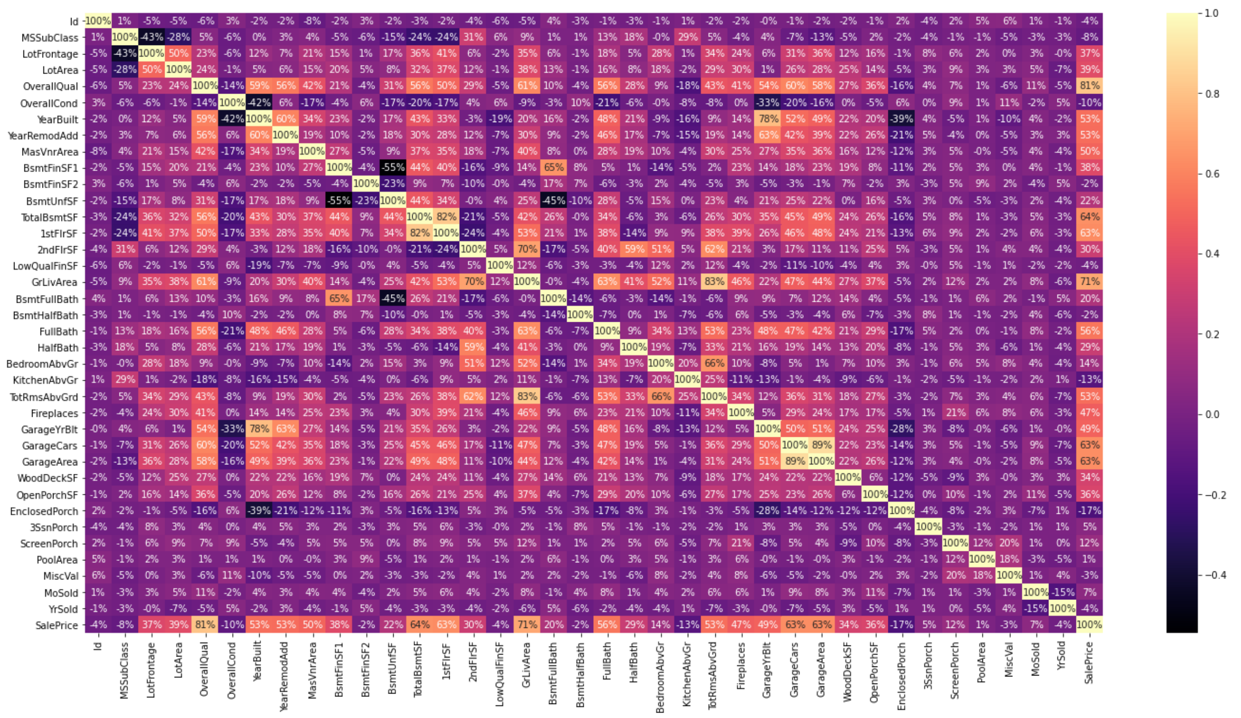
Now since all the data cleaning steps have been done. let us check the missing values in whole dataframe

```
Id              0
MSSubClass      0
MSZoning        0
LotFrontage     0
LotArea         0
Street          0
LotShape        0
LandContour     0
Utilities       0
LotConfig       0
LandSlope       0
Neighborhood    0
Condition1      0
Condition2      0
BldgType        0
HouseStyle      0
```

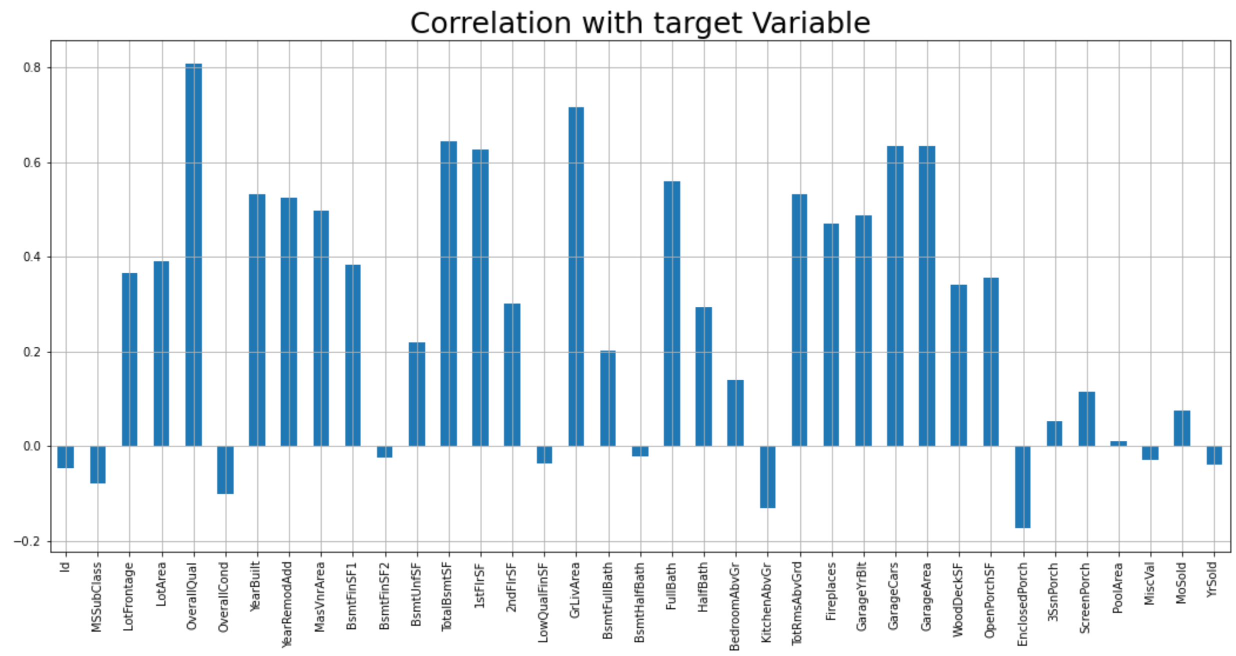
```
OverallQual      0
OverallCond      0
YearBuilt        0
YearRemodAdd     0
RoofStyle        0
RoofMatl         0
Exterior1st      0
Exterior2nd      0
MasVnrType       0
MasVnrArea       0
ExterQual        0
ExterCond        0
Foundation       0
BsmtQual         0
BsmtCond         0
BsmtExposure     0
BsmtFinType1     0
BsmtFinSF1       0
BsmtFinType2     0
BsmtFinSF2       0
BsmtUnfSF        0
TotalBsmtSF      0
Heating          0
HeatingQC        0
CentralAir       0
Electrical       0
1stFlrSF         0
2ndFlrSF         0
LowQualFinSF     0
GrLivArea        0
BsmtFullBath     0
BsmtHalfBath     0
FullBath         0
HalfBath         0
BedroomAbvGr     0
KitchenAbvGr     0
KitchenQual      0
TotRmsAbvGrd     0
Functional       0
Fireplaces       0
GarageType       0
GarageYrBlt      0
GarageFinish     0
GarageCars       0
GarageArea       0
GarageQual       0
GarageCond       0
```

```
PavedDrive       0
WoodDeckSF       0
OpenPorchSF      0
EnclosedPorch    0
3SsnPorch        0
ScreenPorch      0
PoolArea         0
MiscVal          0
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
SalePrice        0
```

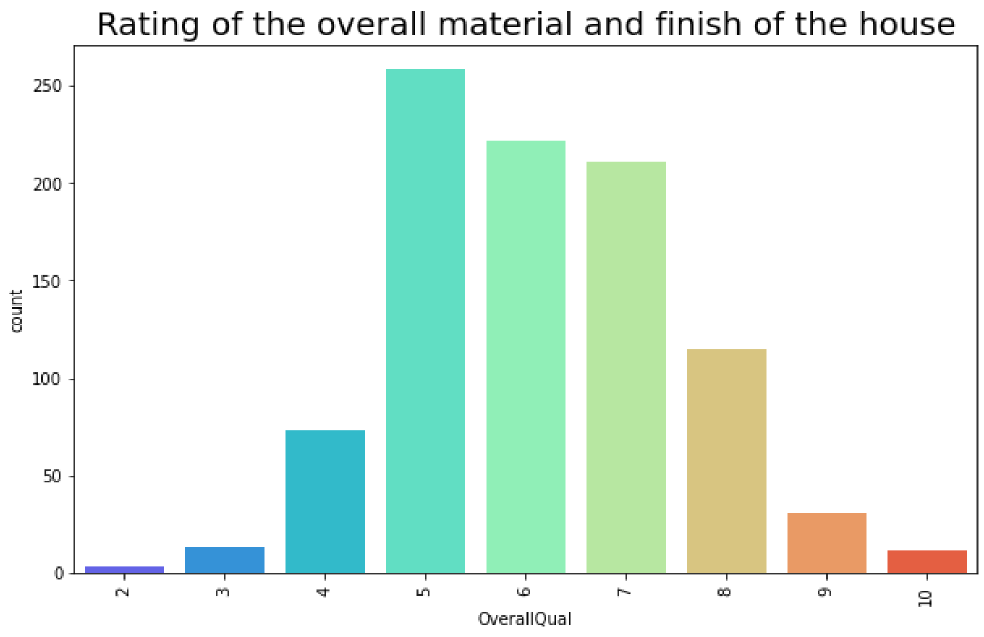
```
#Seeing the correlation via visualization
plt.figure(figsize=(24,12))
sns.heatmap(df.corr(),annot=True,fmt='.0%',cmap='magma')
plt.show()
```



```
#Checking the correlation with target variable that is SalePrice
plt.figure(figsize=(18,8))
df.drop('SalePrice', axis=1).corrwith(df['SalePrice']).plot(kind='bar',grid=True )
plt.xticks(rotation='vertical')
plt.title("Correlation with target Variable",fontsize=25)
plt.show()
```



```
#Rates the overall material and finish of the house.
plt.figure(figsize=(10,6))
sns.countplot(df['OverallQual'],palette= 'rainbow')
plt.title("Rating of the overall material and finish of the house",fontsize=20)
plt.xticks(rotation='vertical')
plt.show()

print(df.OverallQual.value_counts())
```



1-12 houses have Very excellent rating and 3 houses have very poor rating based on the overall material and finish of the house.

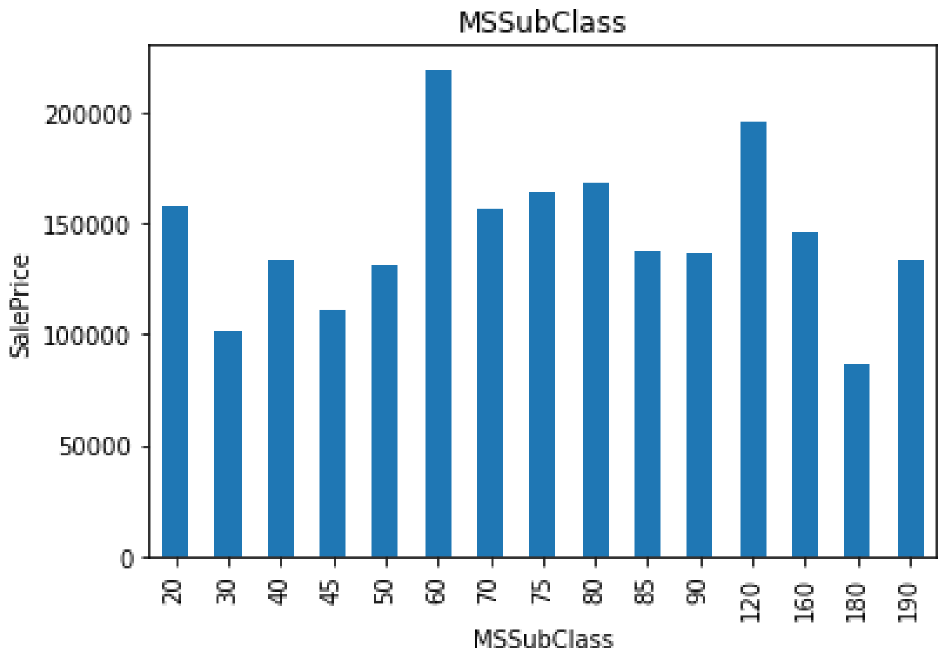
2-222 houses have above average rating and 211 houses have good rating based on the overall material and finish of the house.

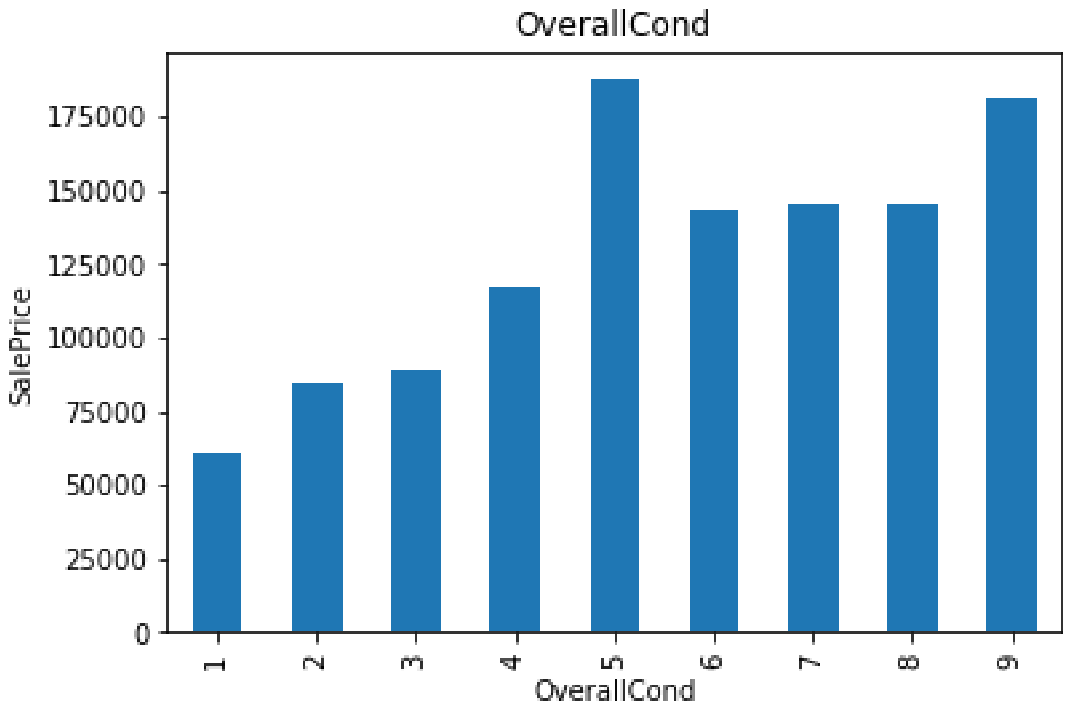
Relationship between feature vs Saleprice

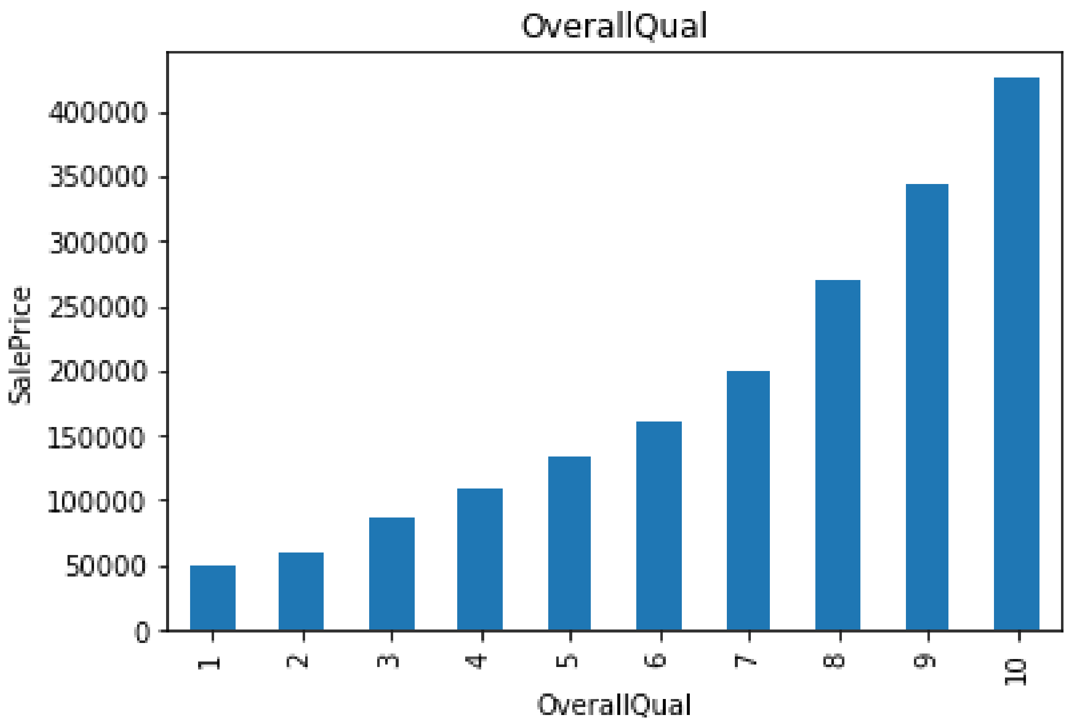
# relationship between year variables and SalePrice can be done using the python code

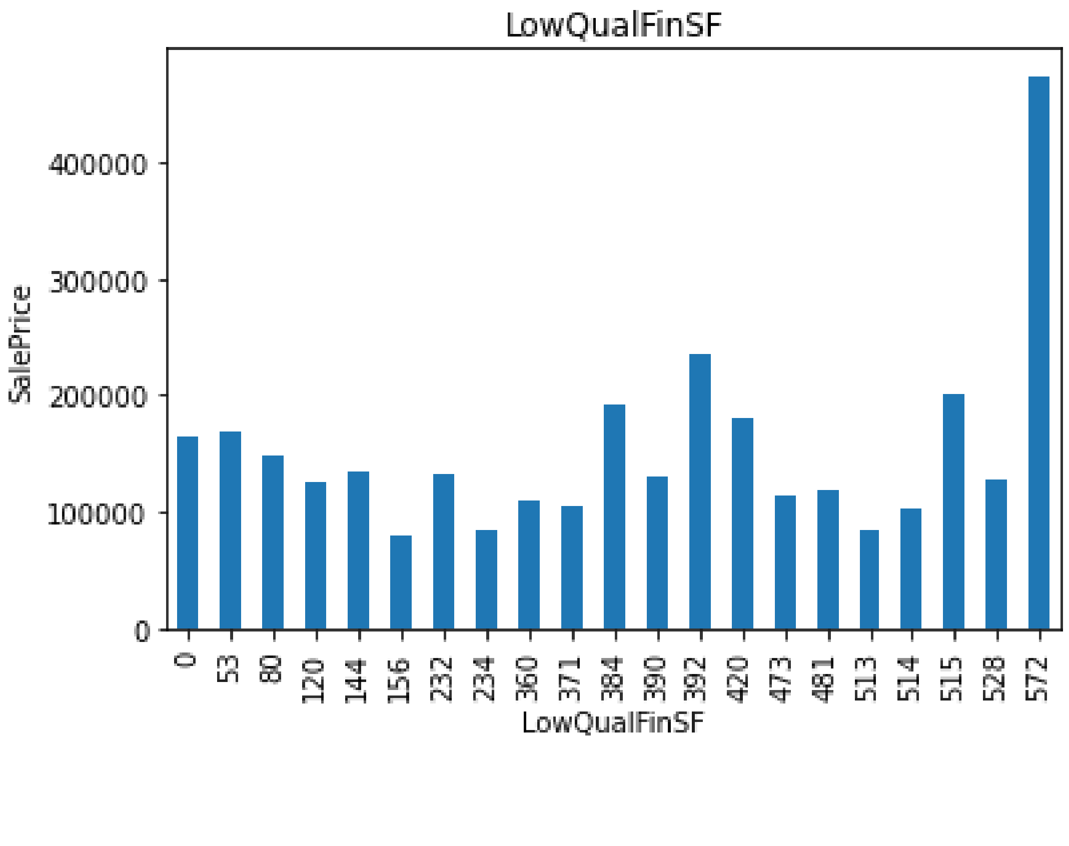
```
for feature in year_feature:
    plt.figure(figsize=(8,6))
    df.groupby(feature)['SalePrice'].median().plot()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.show()
```

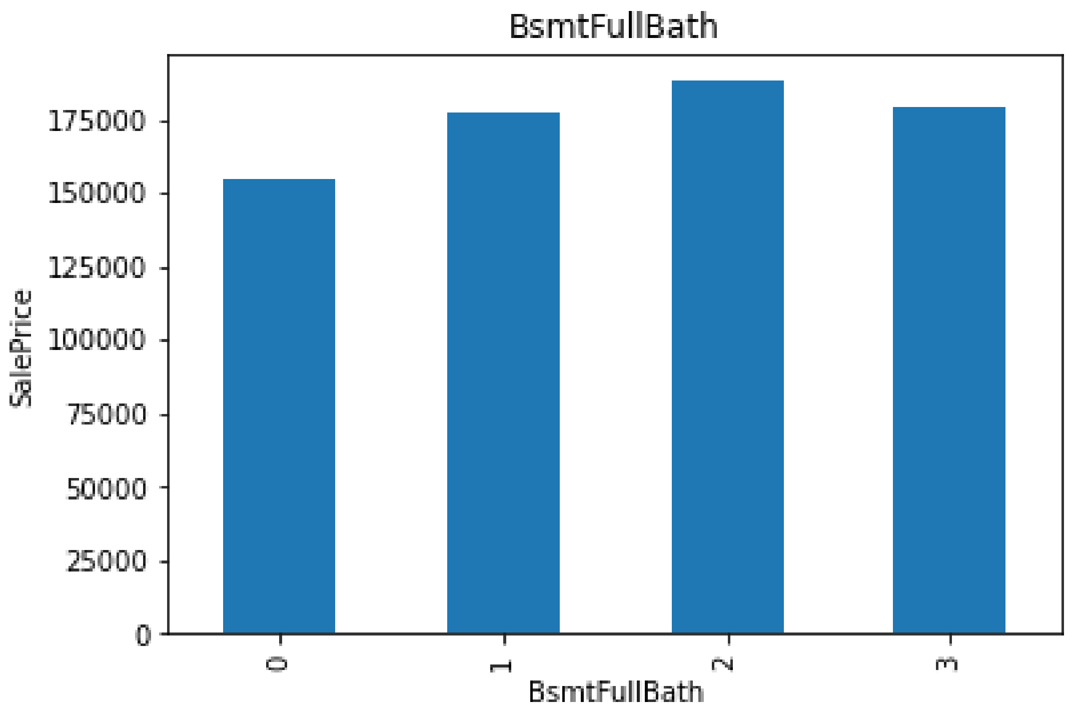
Data Visualization:-

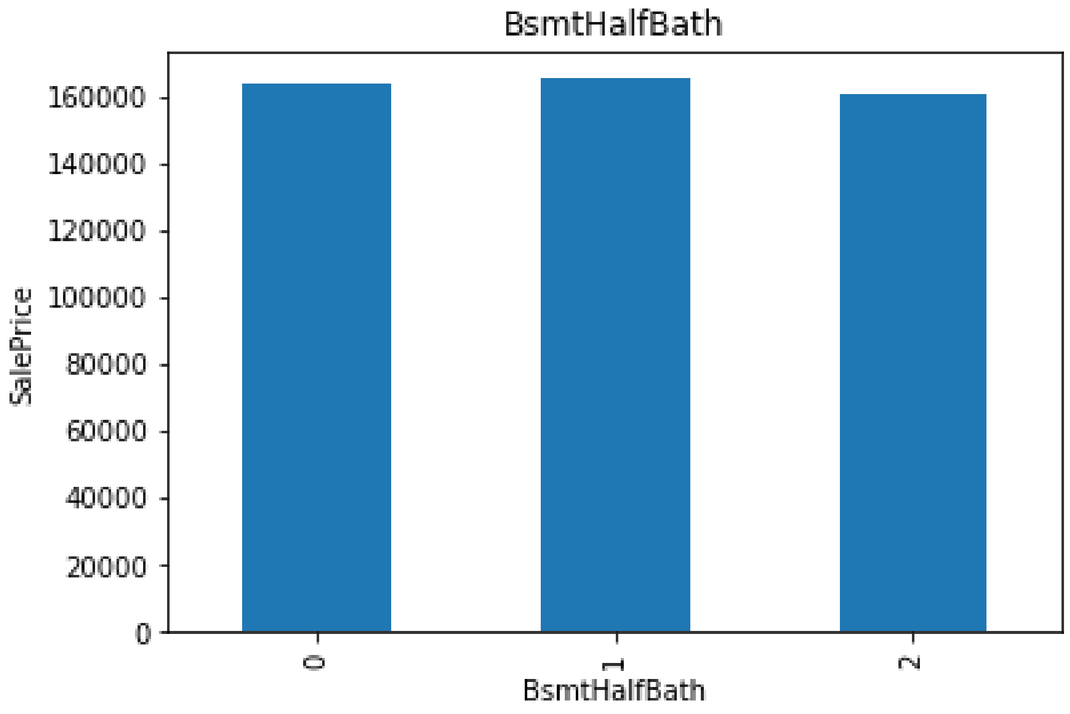

MSSubClass
SalePrice
200000
150000
100000
50000
0
20
30
40
45
50
60
70
75
80
85
90
120
160
180
190
MSSubClass

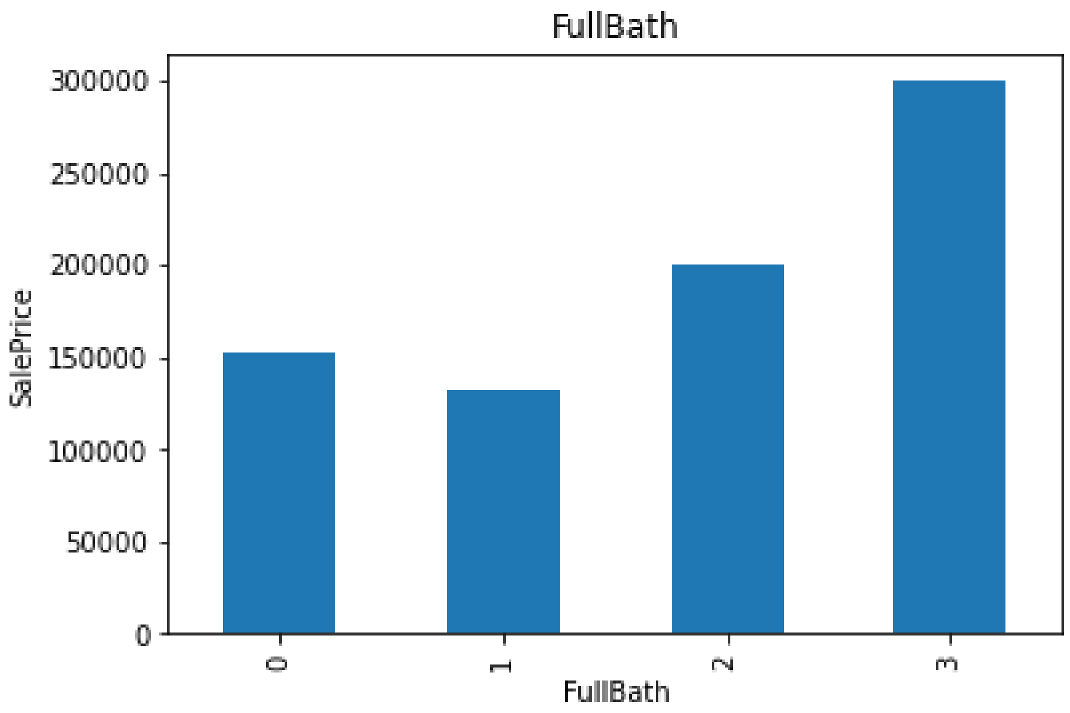

OverallCond
SalePrice
175000
150000
125000
100000
75000
50000
25000
0
1
2
3
4
5
6
7
8
9
OverallCond

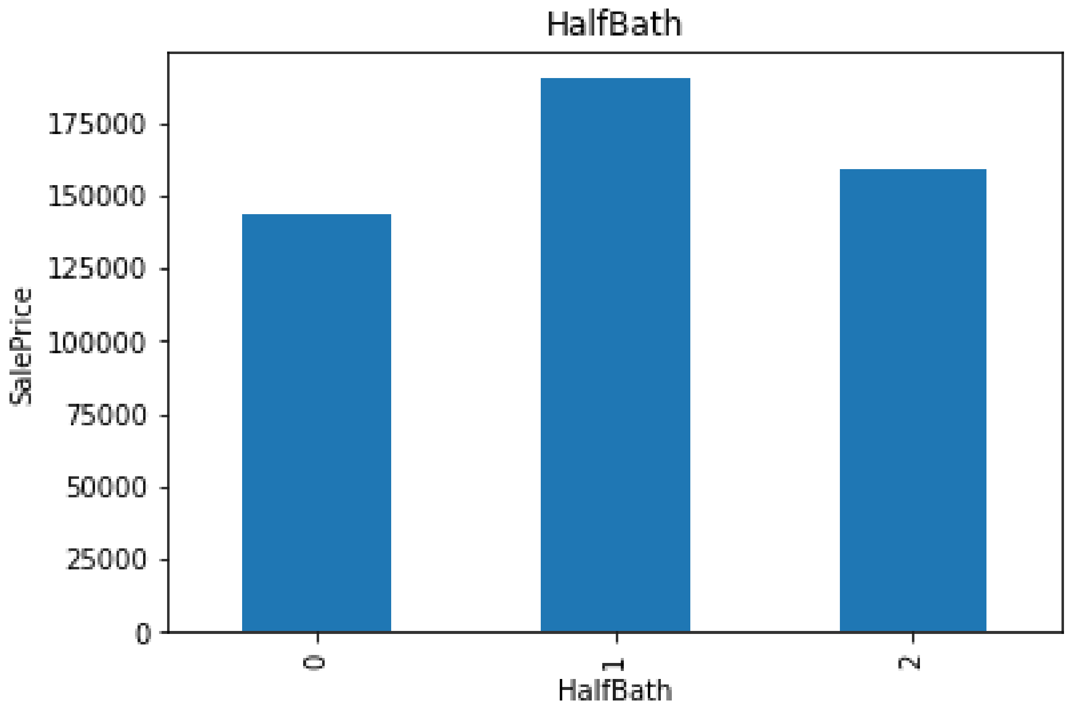

OverallQual
SalePrice
400000
350000
300000
250000
200000
150000
100000
50000
0
1
2
3
4
5
6
7
8
9
10
OverallQual

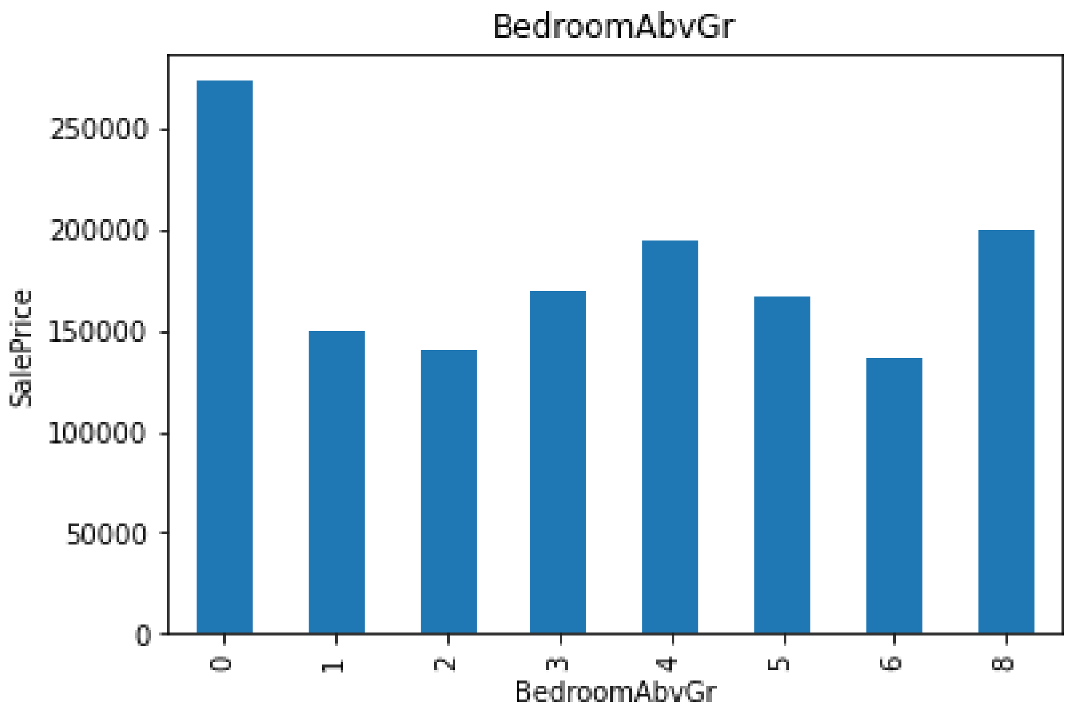

LowQualFinSF
SalePrice
400000
300000
200000
100000
0
0
53
80
120
144
156
232
234
360
371
384
390
392
420
473
481
513
514
515
528
572
LowQualFinSF

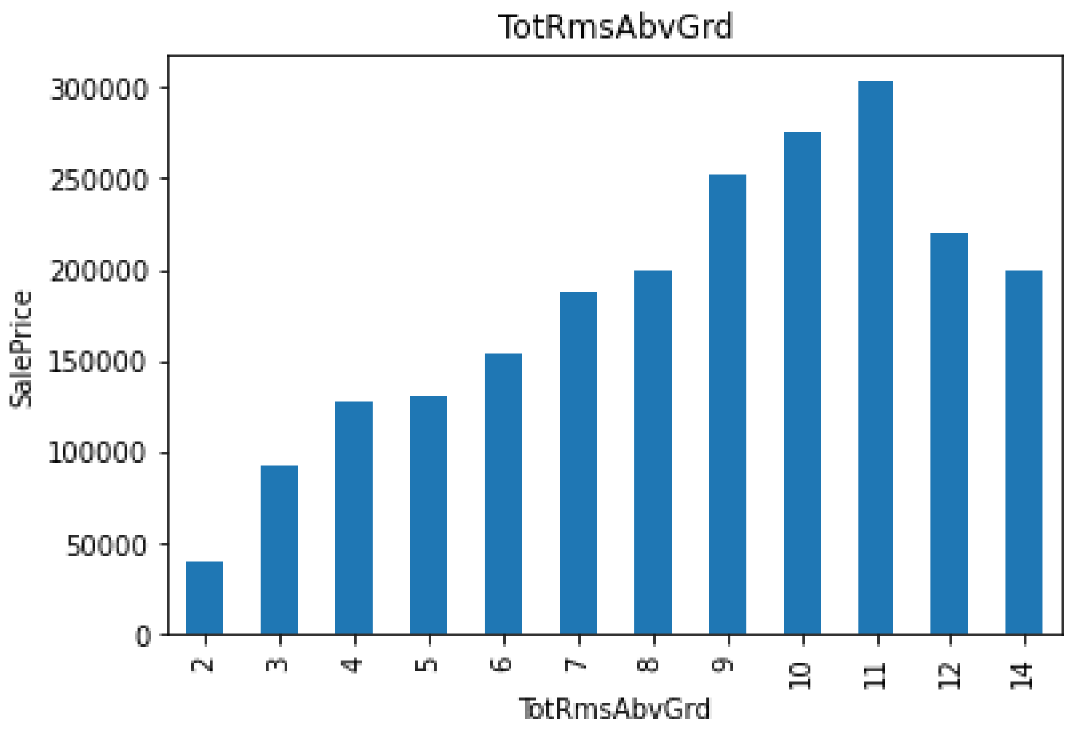

BsmtFullBath
175000
150000
125000
100000
75000
50000
25000
0
SalePrice
0
1
2
3
BsmtFullBath

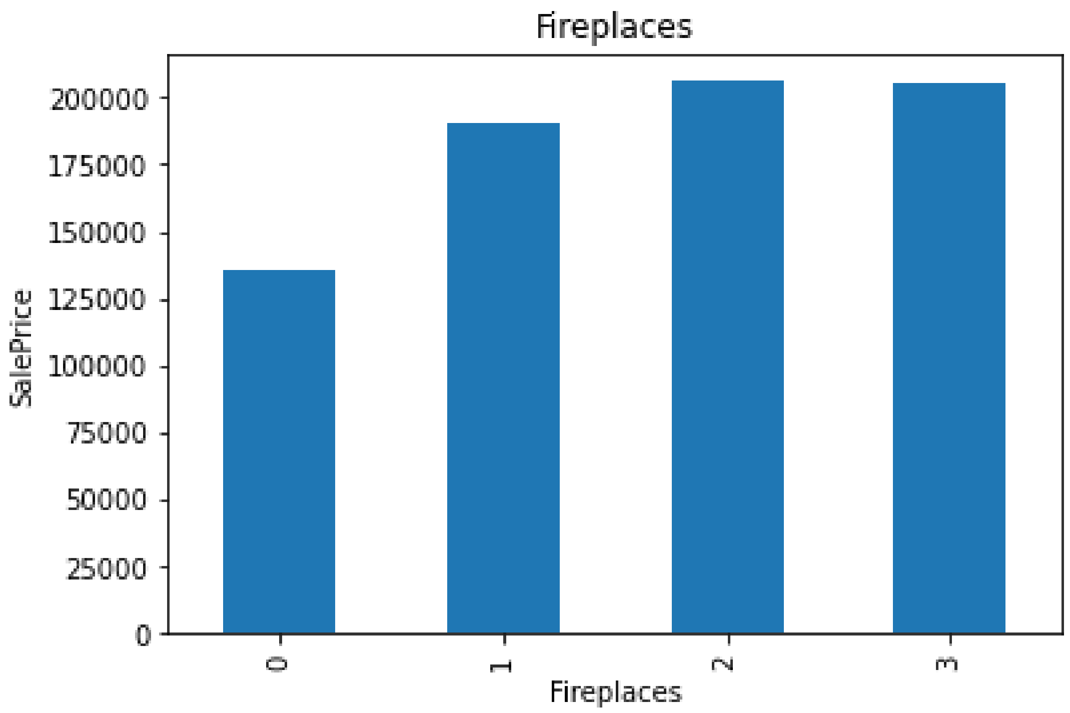

BsmtHalfBath
160000
140000
120000
100000
80000
60000
40000
20000
0
SalePrice
0
1
2
BsmtHalfBath

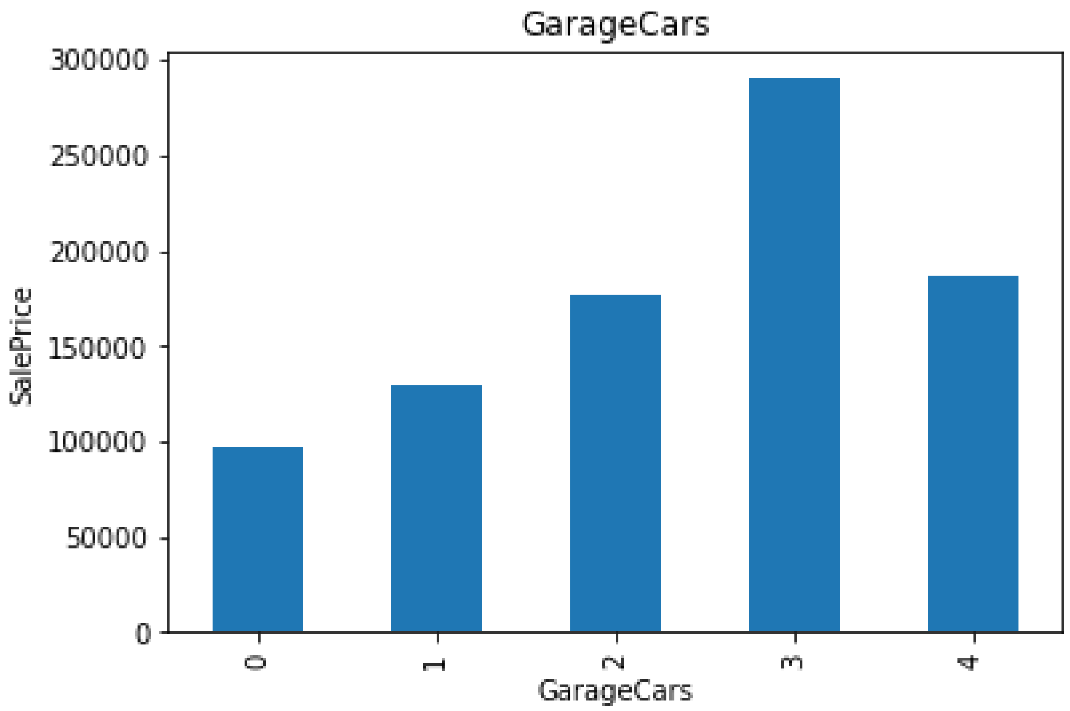

FullBath
SalePrice
300000
250000
200000
150000
100000
50000
0
0
1
2
3
FullBath

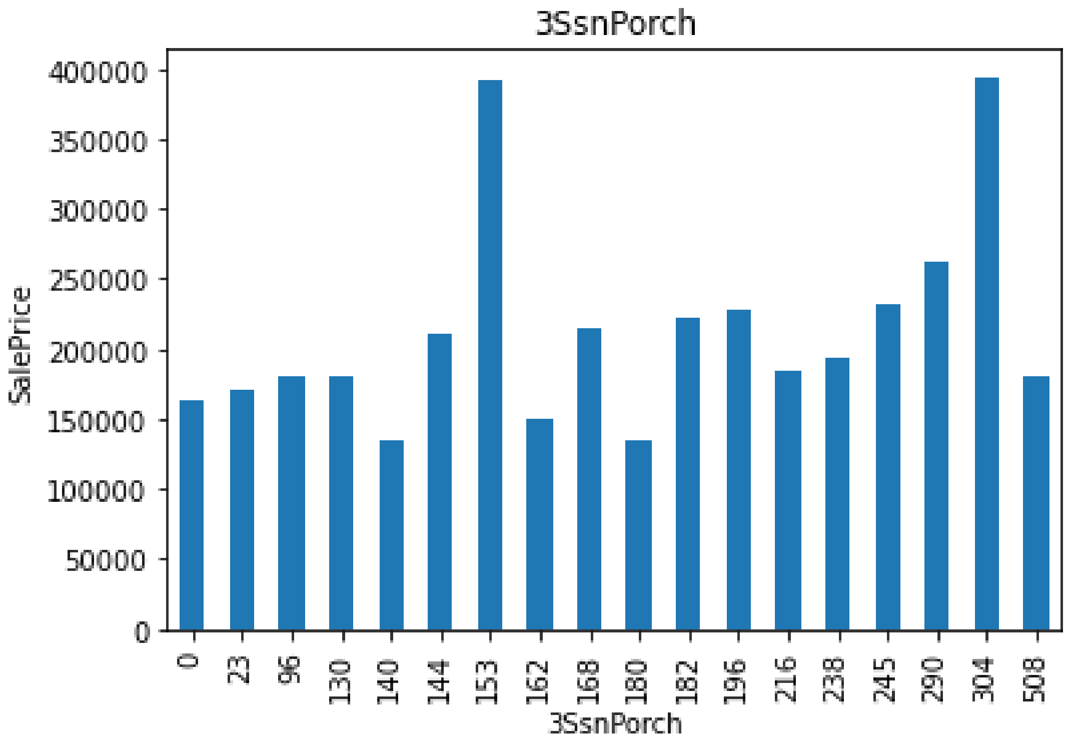

HalfBath
SalePrice
175000
150000
125000
100000
75000
50000
25000
0
0
1
2
HalfBath


BedroomAbvGr
SalePrice
0
50000
100000
150000
200000
250000
0
1
2
3
4
5
6
8
BedroomAbvGr

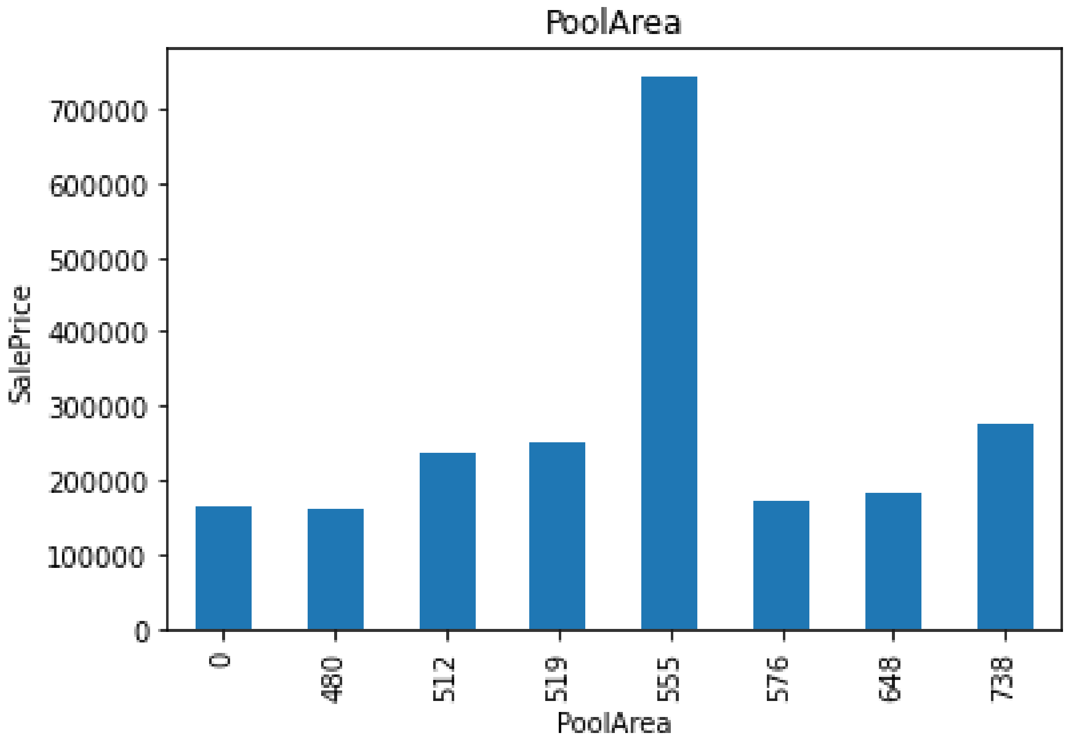

TotRmsAbvGrd
SalePrice
0
50000
100000
150000
200000
250000
300000
2
3
4
5
6
7
8
9
10
11
12
14
TotRmsAbvGrd


Fireplaces
SalePrice
200000
175000
150000
125000
100000
75000
50000
25000
0
0
1
2
3
Fireplaces

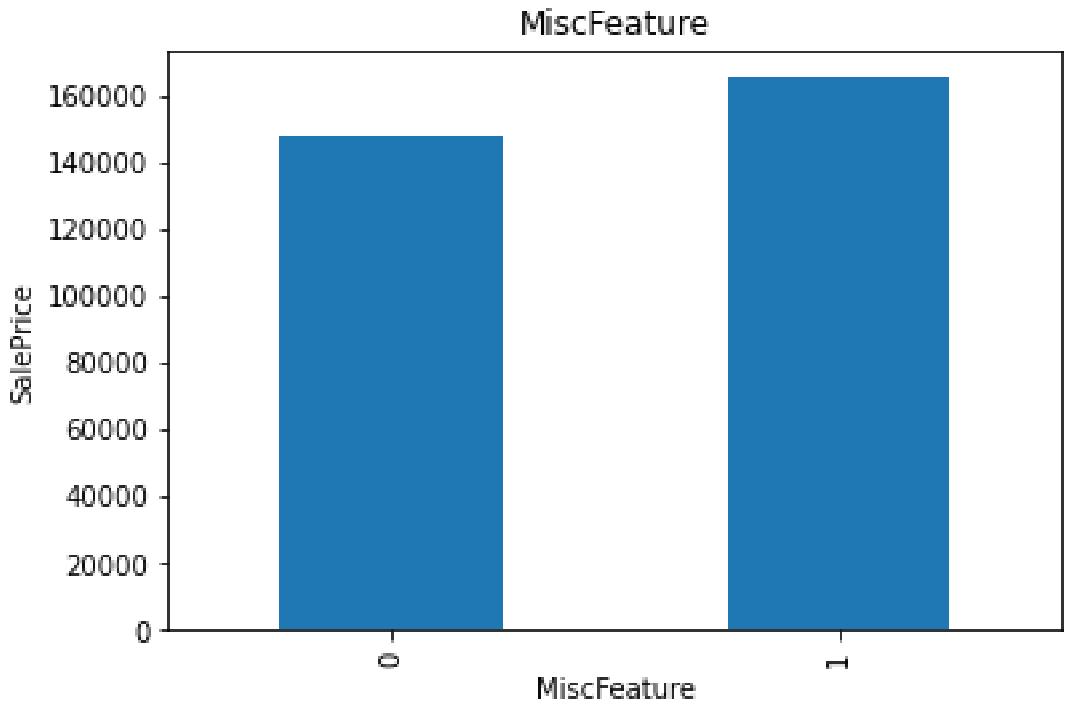

GarageCars
SalePrice
300000
250000
200000
150000
100000
50000
0
0
1
2
3
4
GarageCars

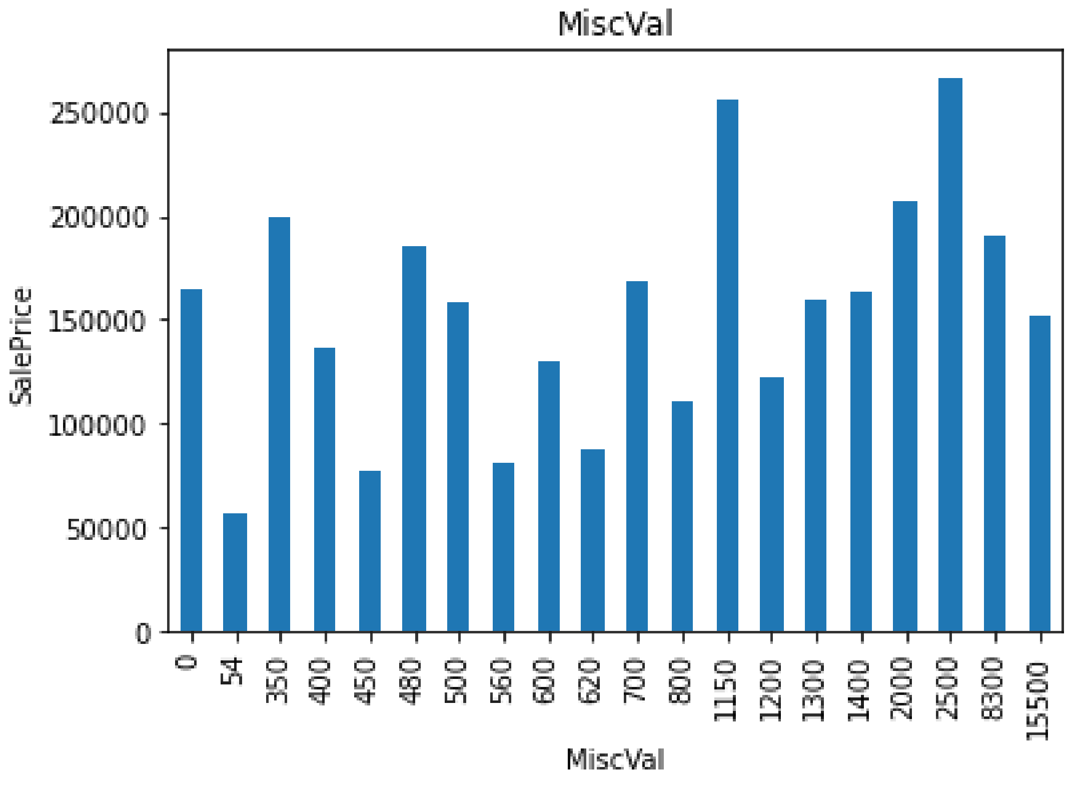


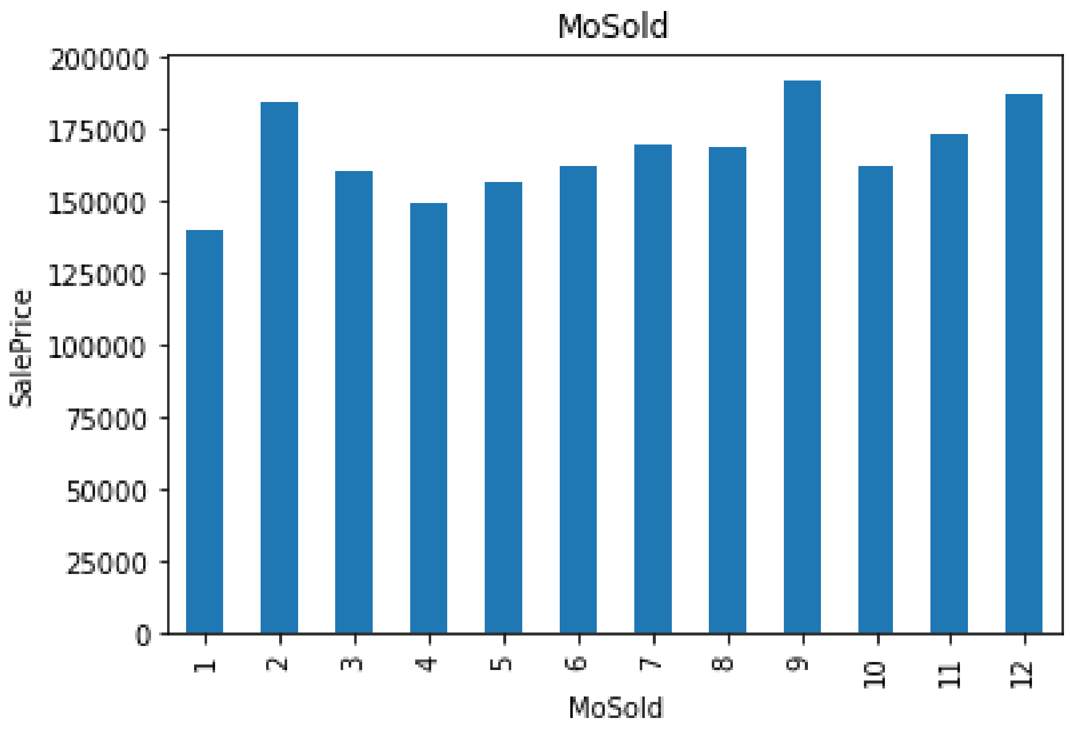
3SsnPorch
SalePrice
0
50000
100000
150000
200000
250000
300000
350000
400000
0
23
96
130
140
144
153
162
168
180
182
196
216
238
245
290
304
508
3SsnPorch



PoolArea
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
0
480
512
519
555
576
648
738
PoolArea

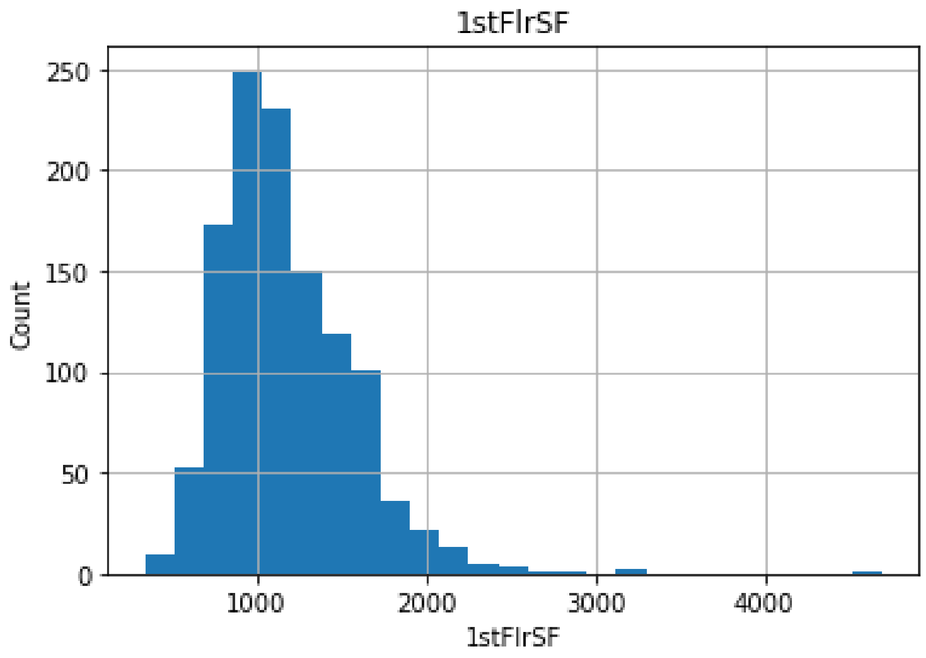

MiscFeature
SalePrice
0
20000
40000
60000
80000
100000
120000
140000
160000
0
1
MiscFeature


MiscVal
SalePrice
0
50000
100000
150000
200000
250000
0
54
350
400
450
480
500
560
600
620
700
800
1150
1200
1300
1400
2000
2500
8300
15500
MiscVal

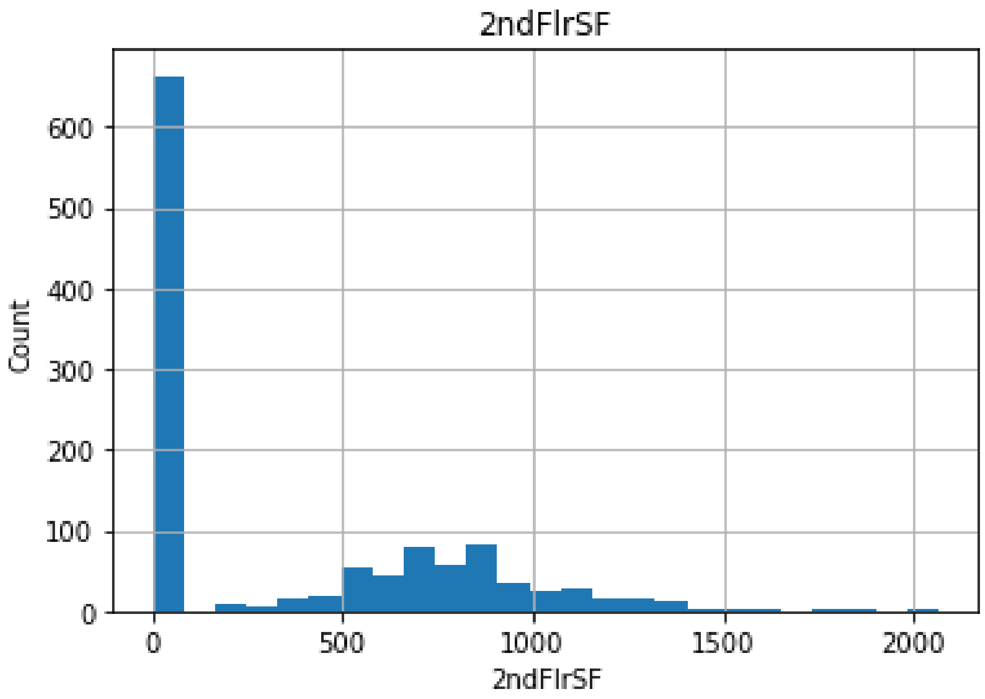


Extracting the continous variable

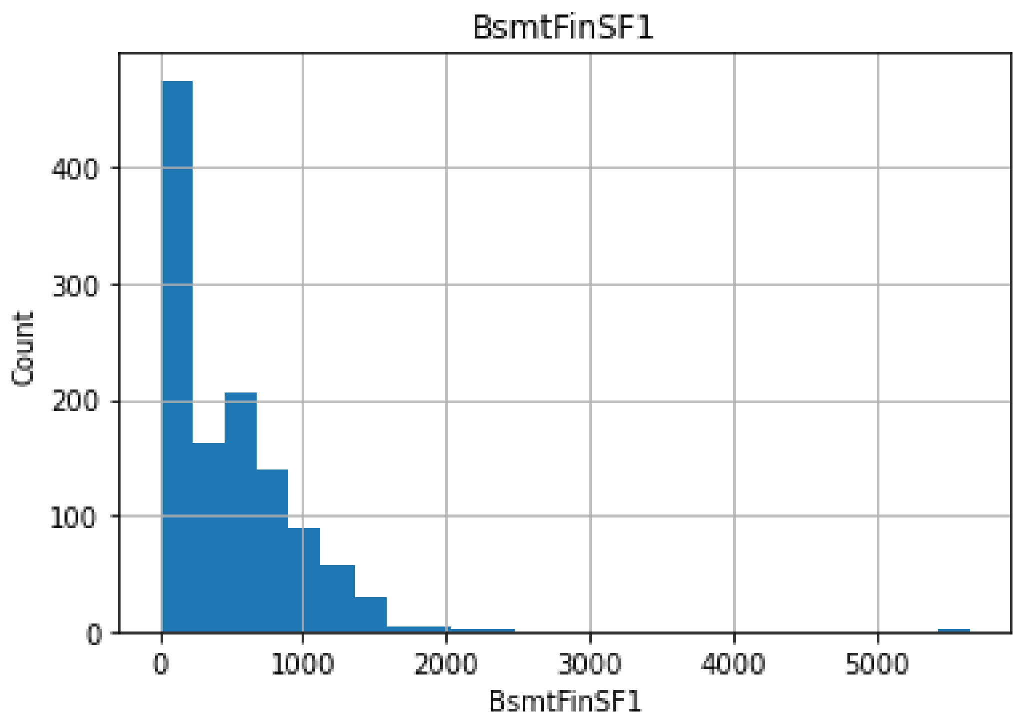
continuous_feature=[feature for feature in numerical_features if feature not in discrete_feature+year_feature+['Id']]
print("Continuous feature Count {}".format(len(continuous_feature))



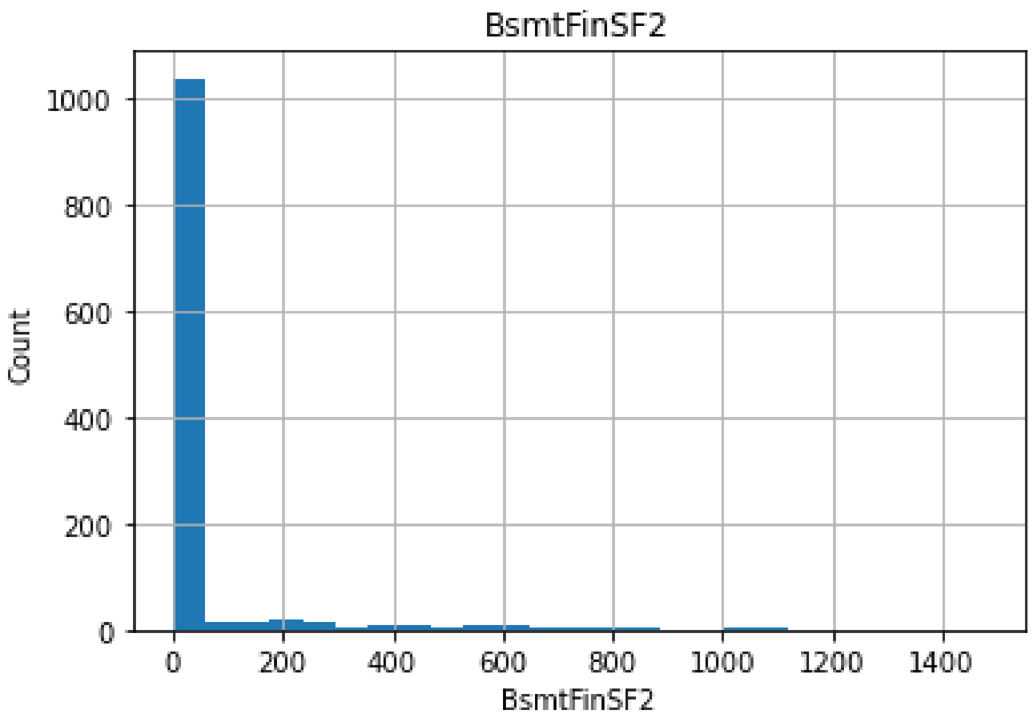
Histogram plot of 1stF1r



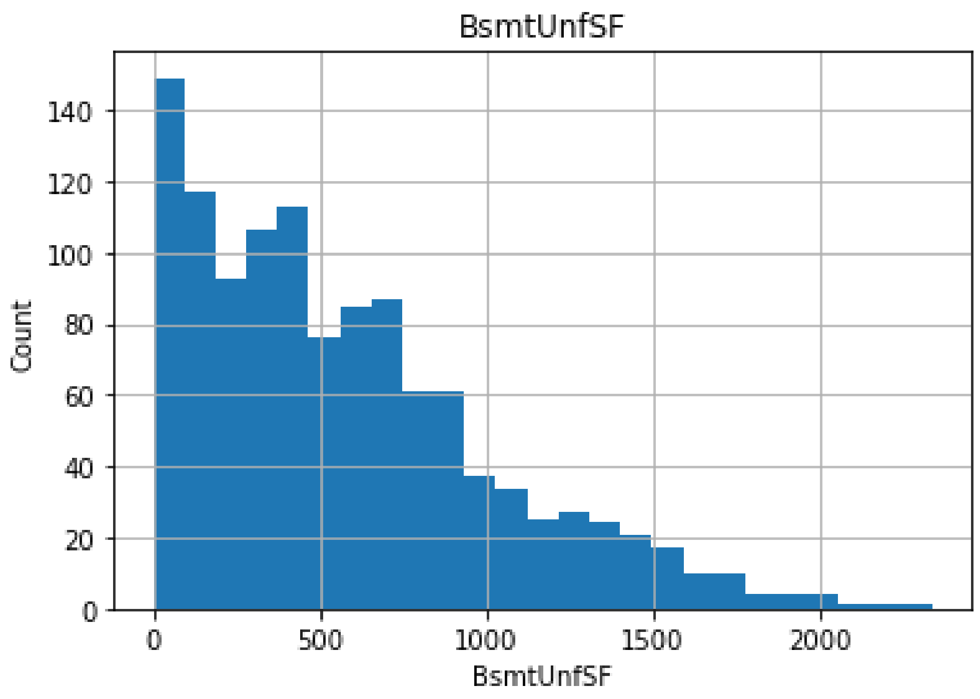
Histogram of 2ndF1Srf



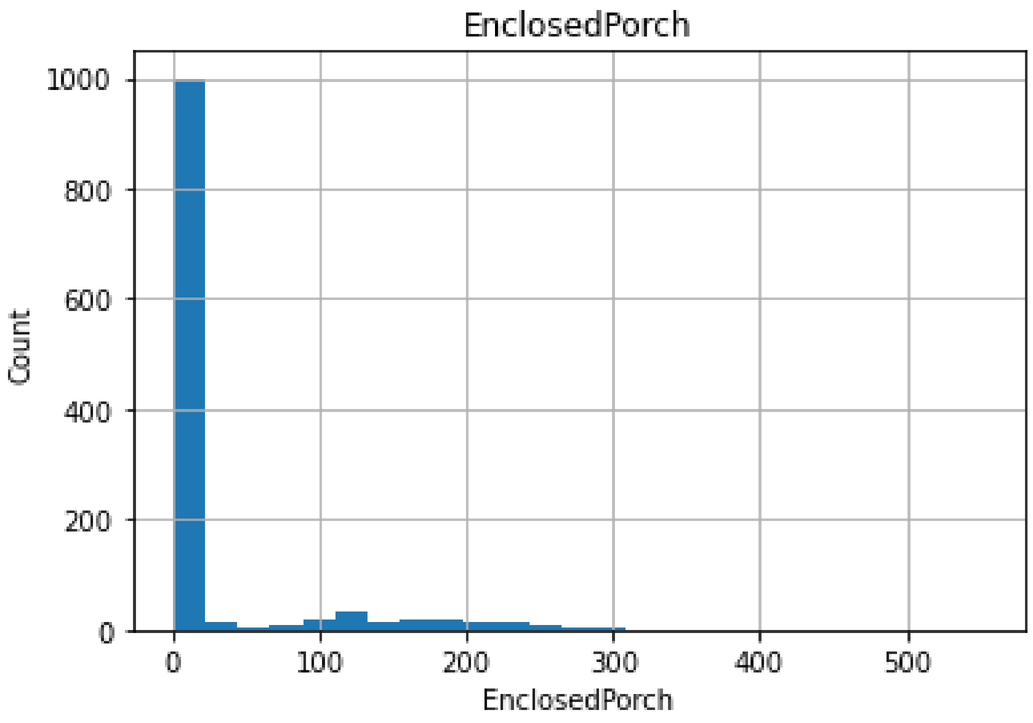
Histogram of BsmtFinSF1



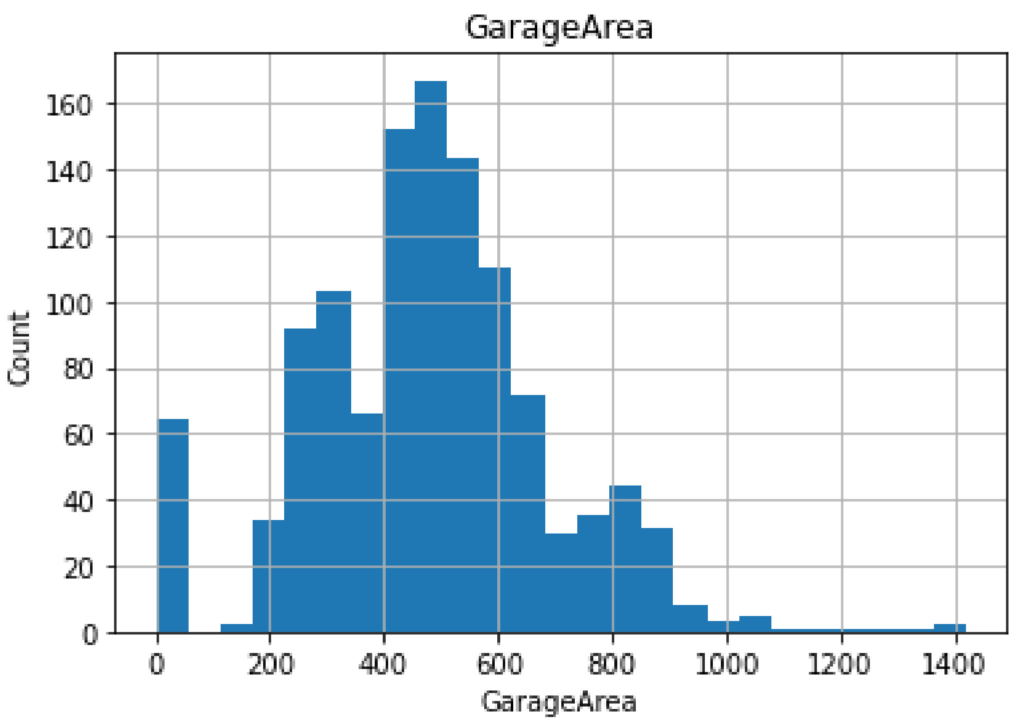
Histogram of BsmtfinSF2



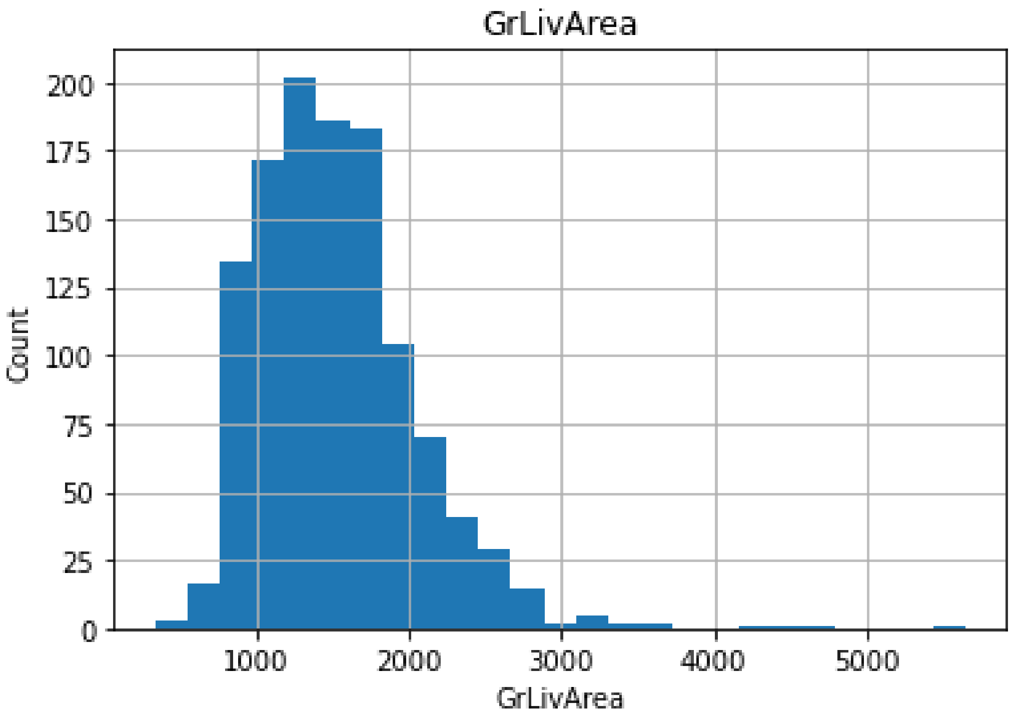
Histogram of BSmmtunfsf



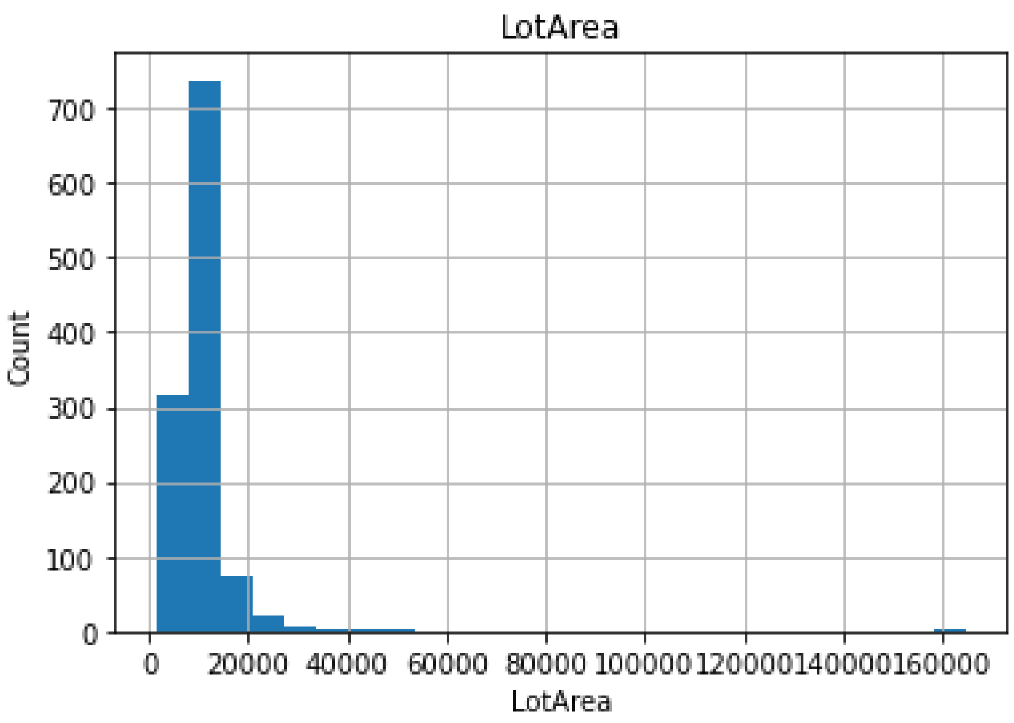
Histogram of EnclosedPorch



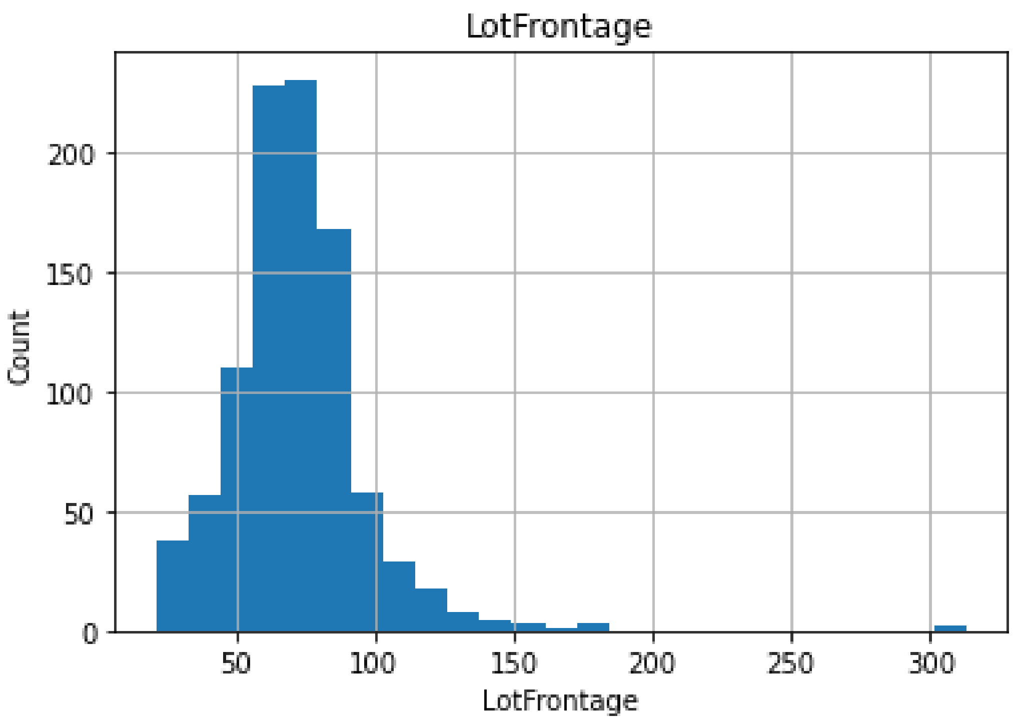
Histogram of GarageArea



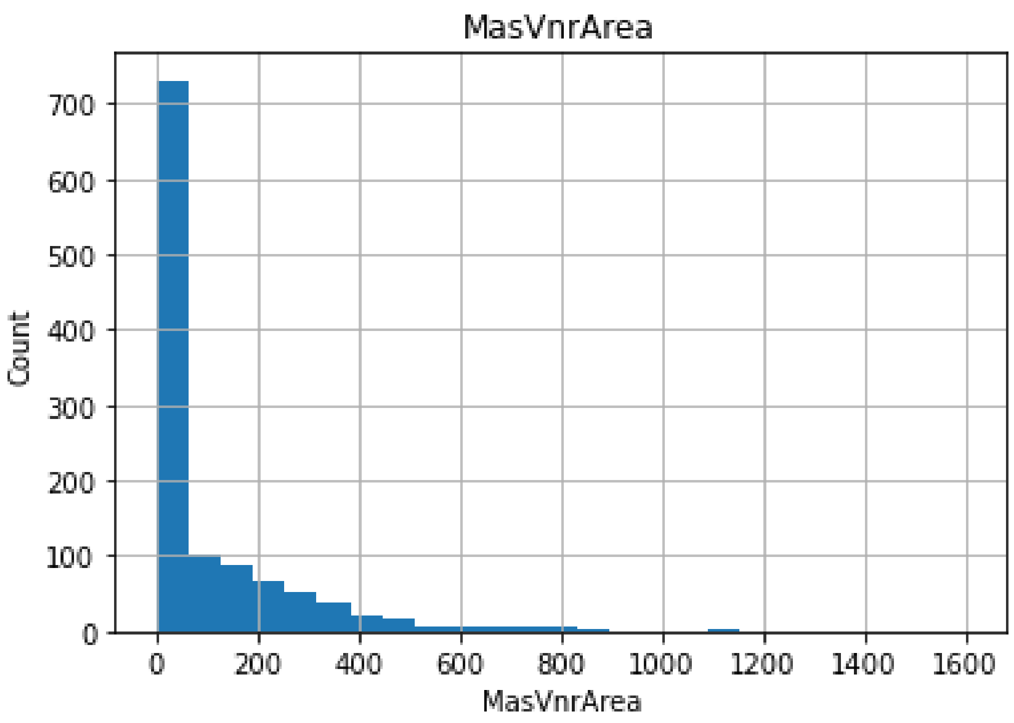
Histogram of GrlivArea



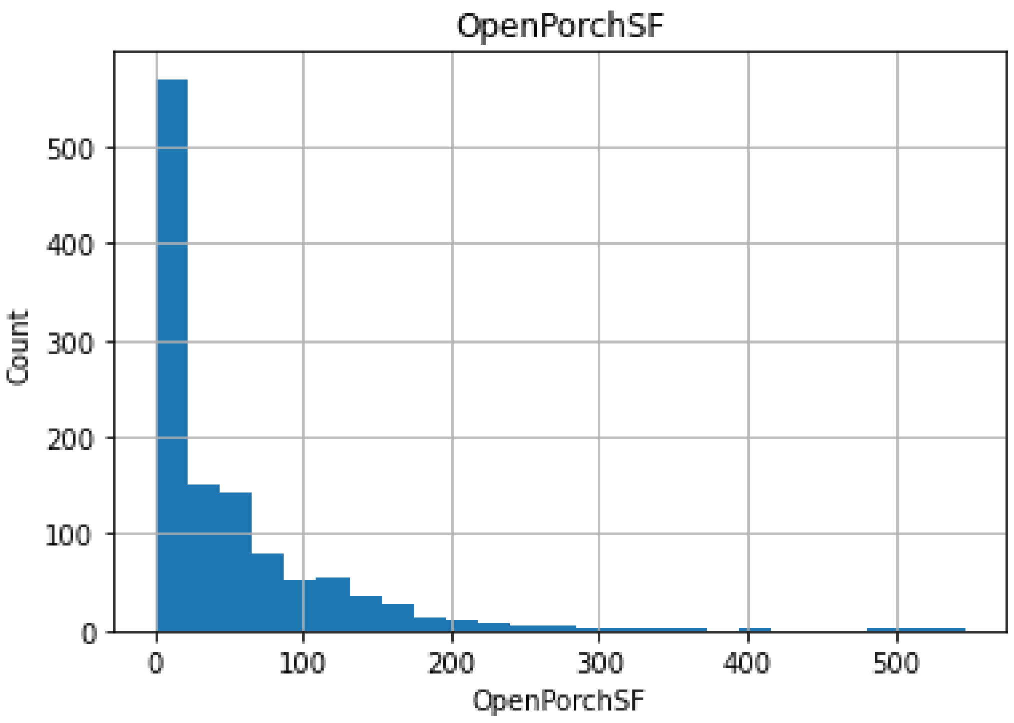
Histogram of LotArea



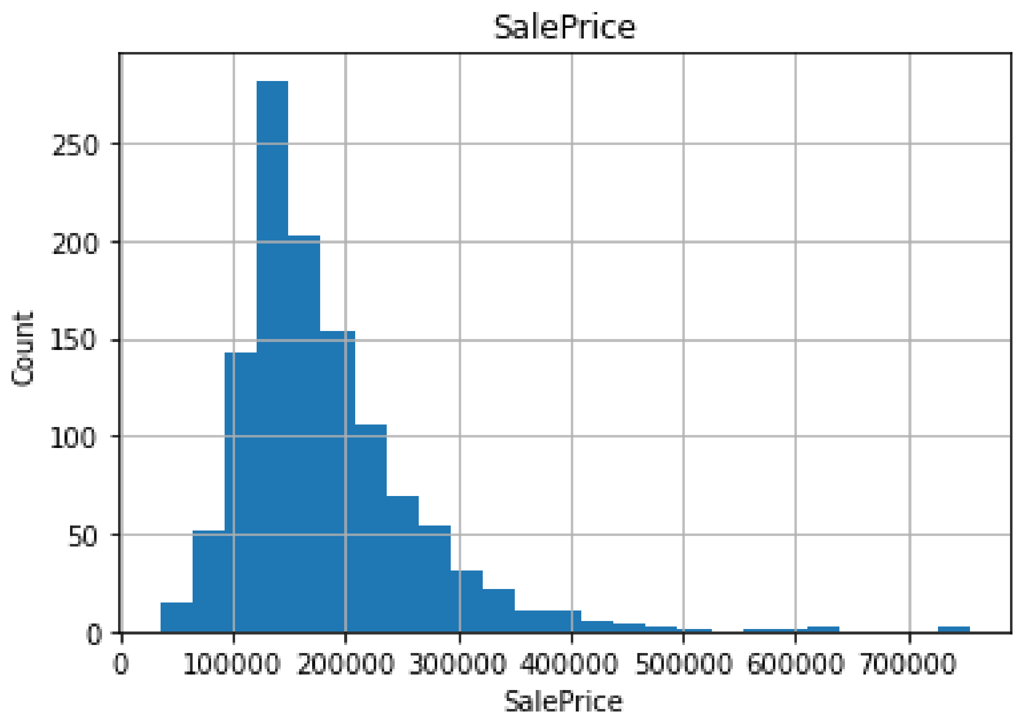
Histogram of LotFrontage



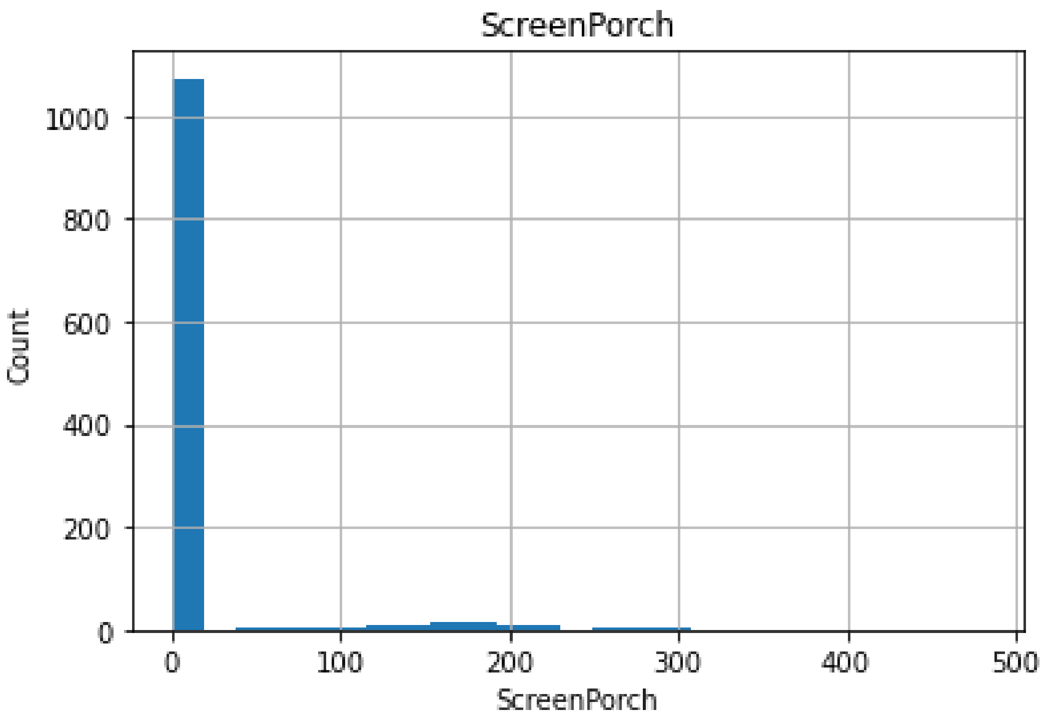
Histogram of MassvnArea



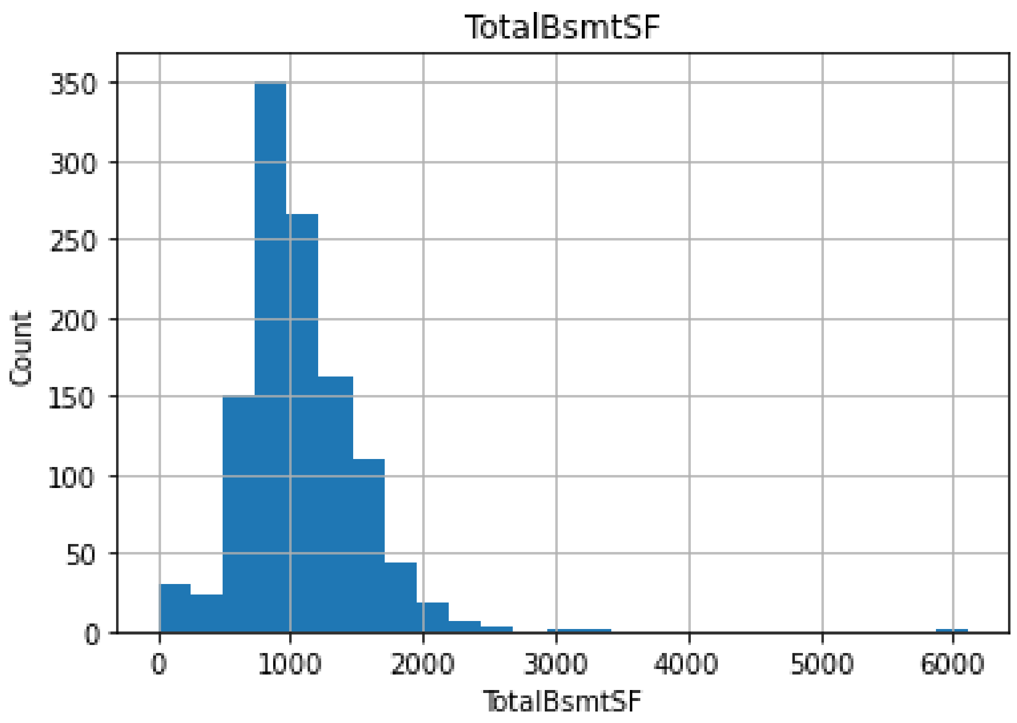
Histogram of OpenPorchSF



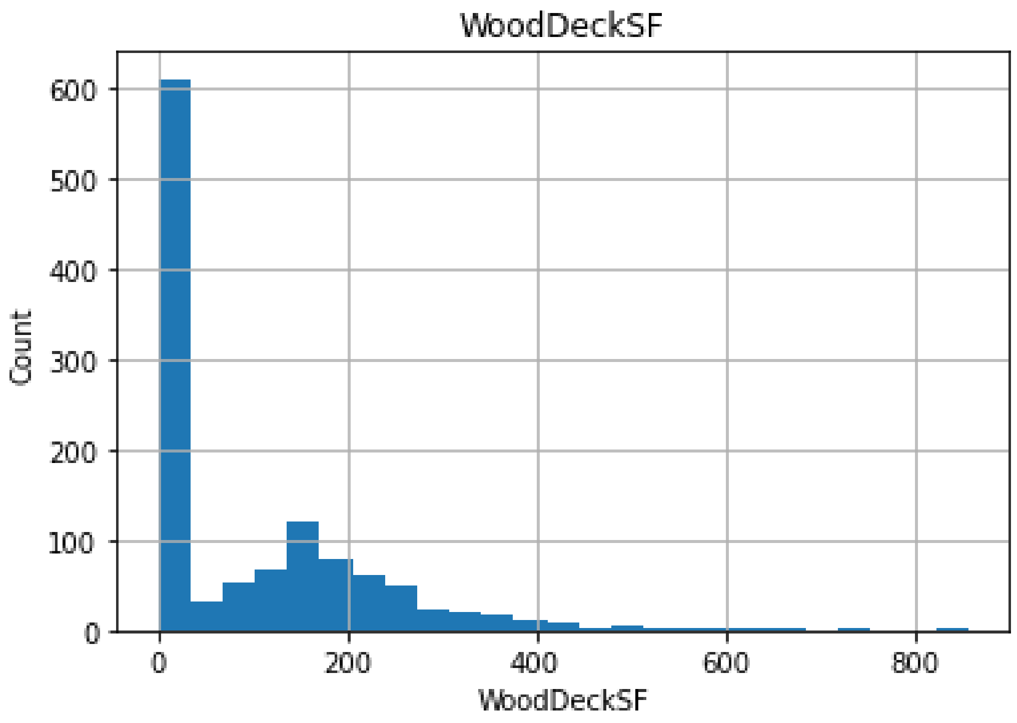
Histogram of SalePrice



Histogram of ScreenPorch



Histogram of TotalBsmtSF



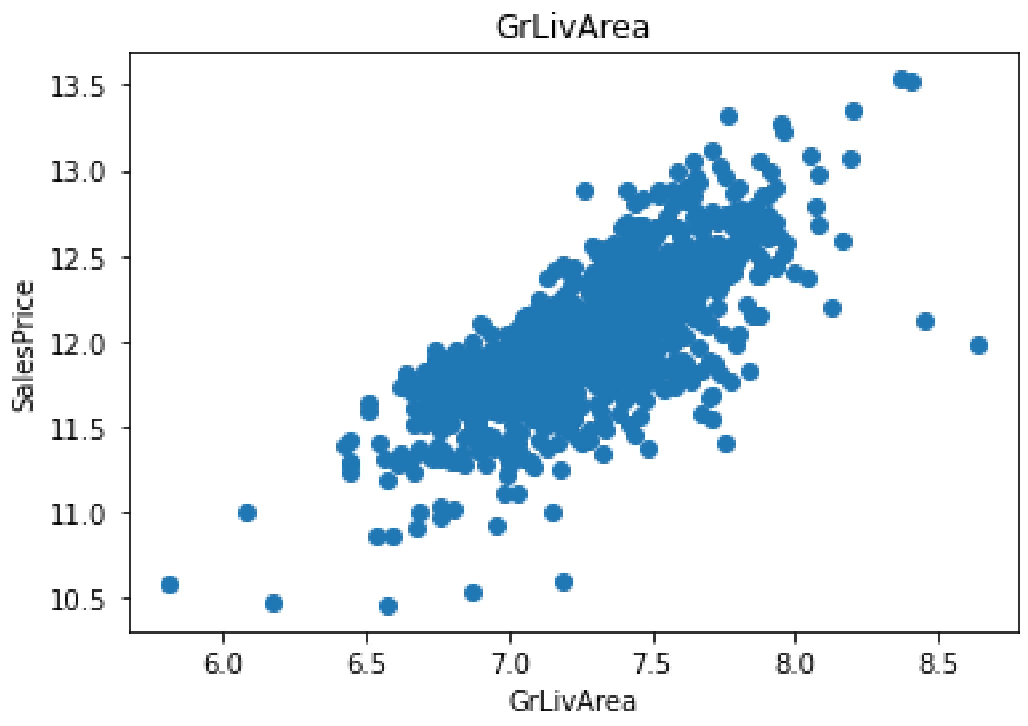
Histogram of WoodDeckSF

Observation:-1. Most of the features are right skewed

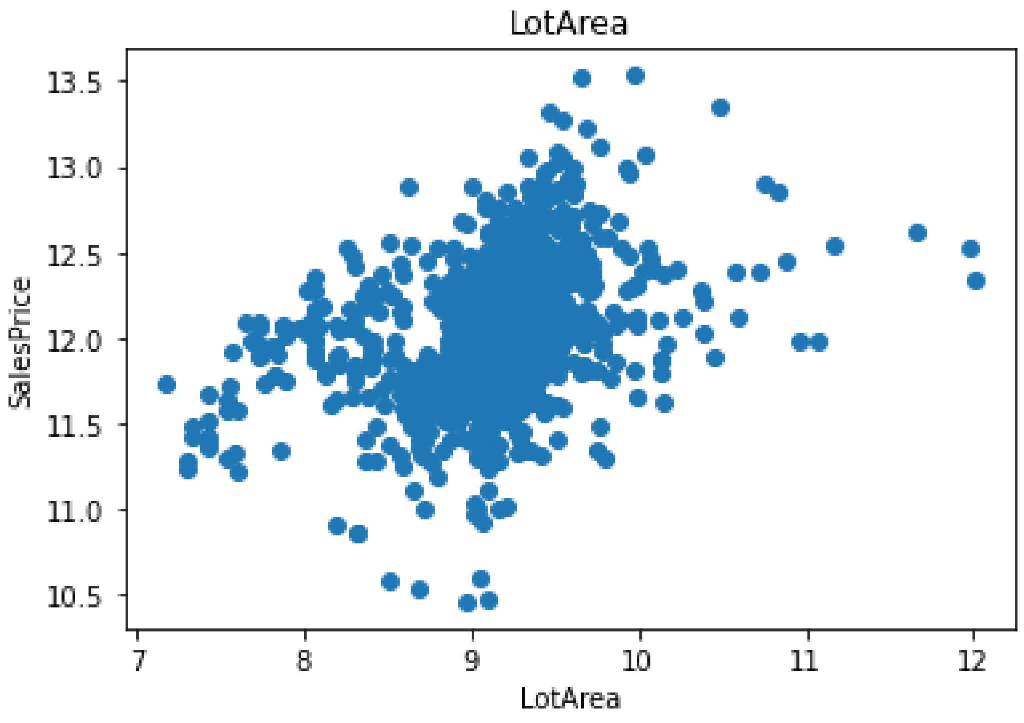
2.Need to go to transformation

Log transformation can be done using the following python code:-

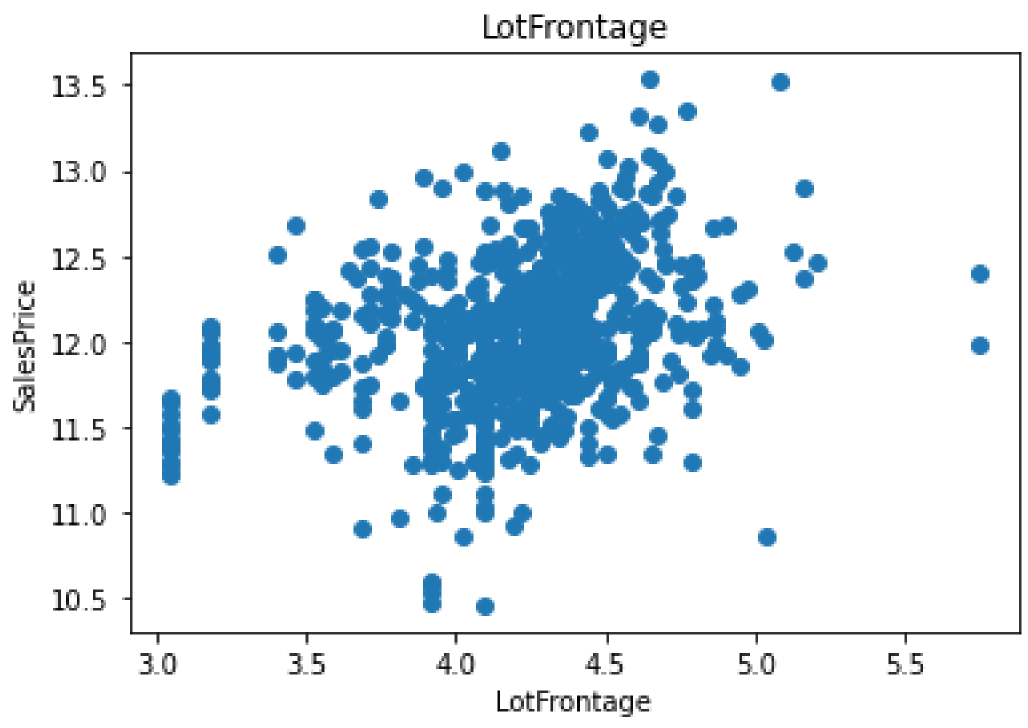
```
## We will be using logarithmic transformation



for feature in continuous_feature:
    if 0 in df[feature].unique():
        pass
    else:
        df[feature]=np.log(df[feature])
        plt.scatter(df[feature],df['SalePrice'])
        plt.xlabel(feature)
        plt.ylabel('SalesPrice')
        plt.title(feature)
        plt.show()
```



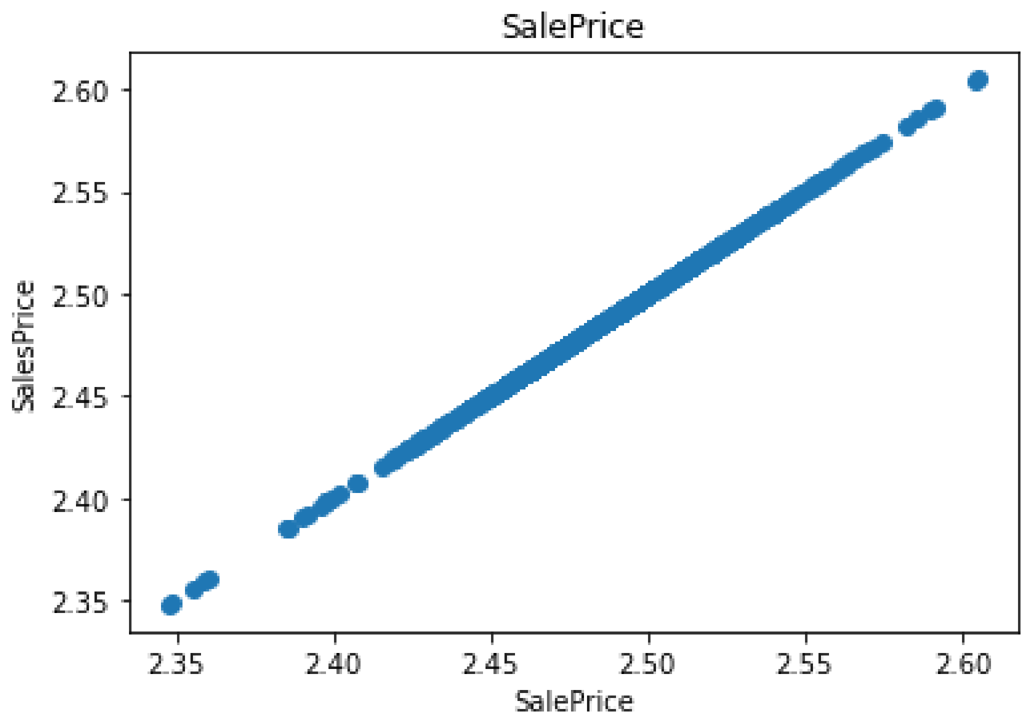
Saleprice vs GLivArea



Saleprice vs LotArea



SalePrice Vs LotFrontage



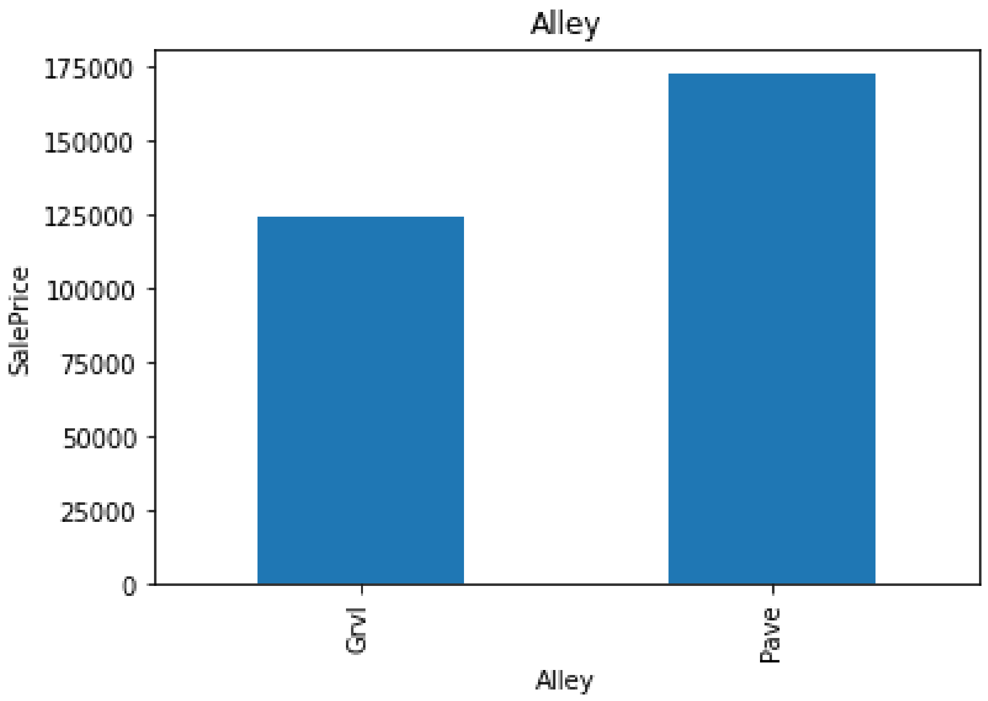
SalePrice vs SalePrice

To check the outliers we are using the box plot.

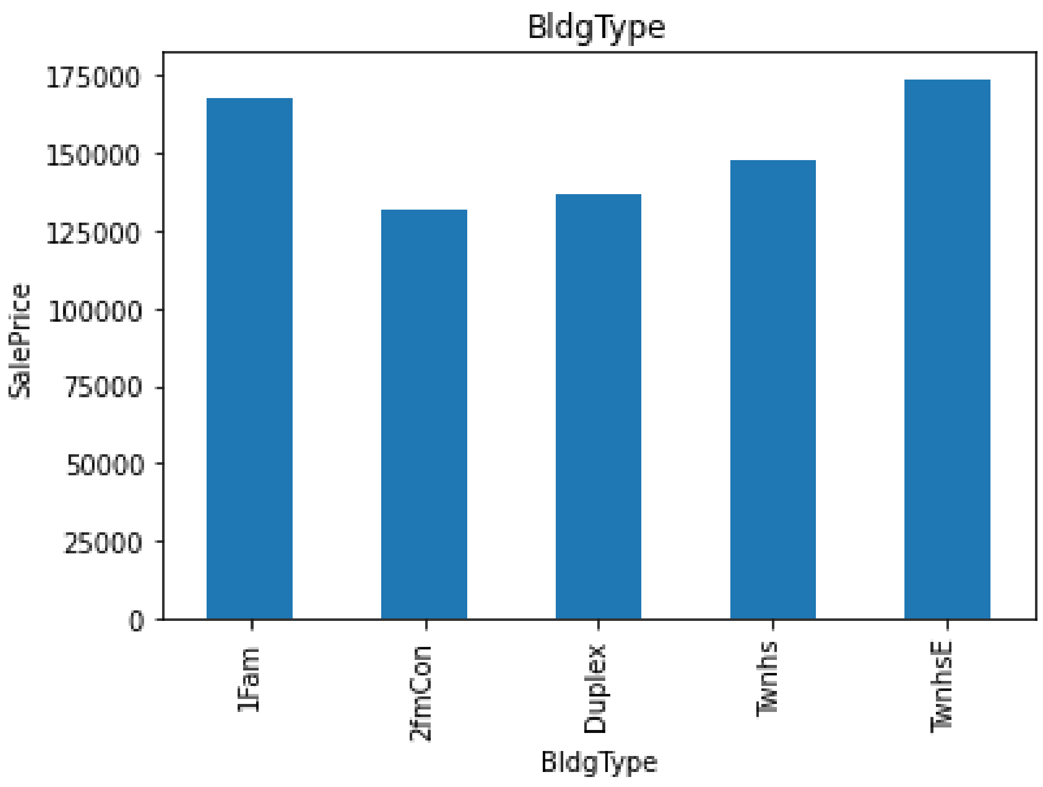
```
for feature in continous_feature:
    data=df.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data.boxplot(feature)
        plt.ylabel(feature)
        plt.title(feature)
        plt.show()
```

Observation:-There are lot of outliers therefore outlier treatment is required.

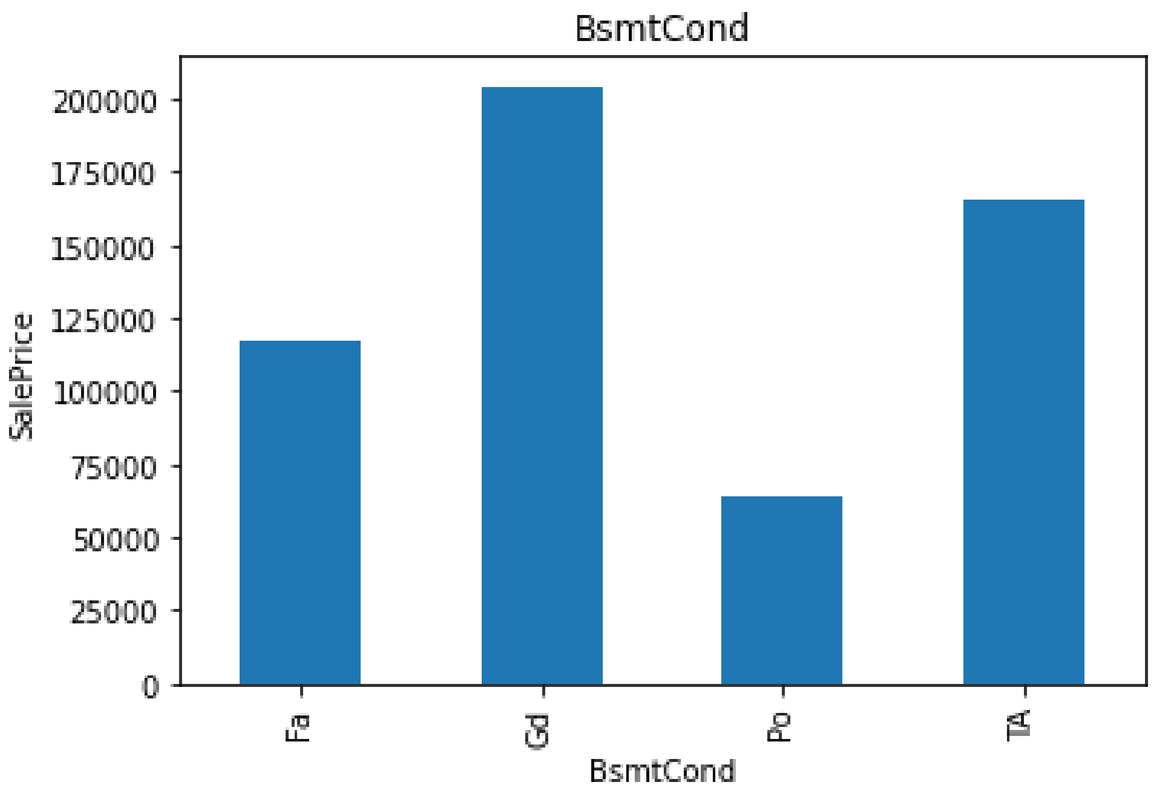
Realation Between categorical feature and SalePrice



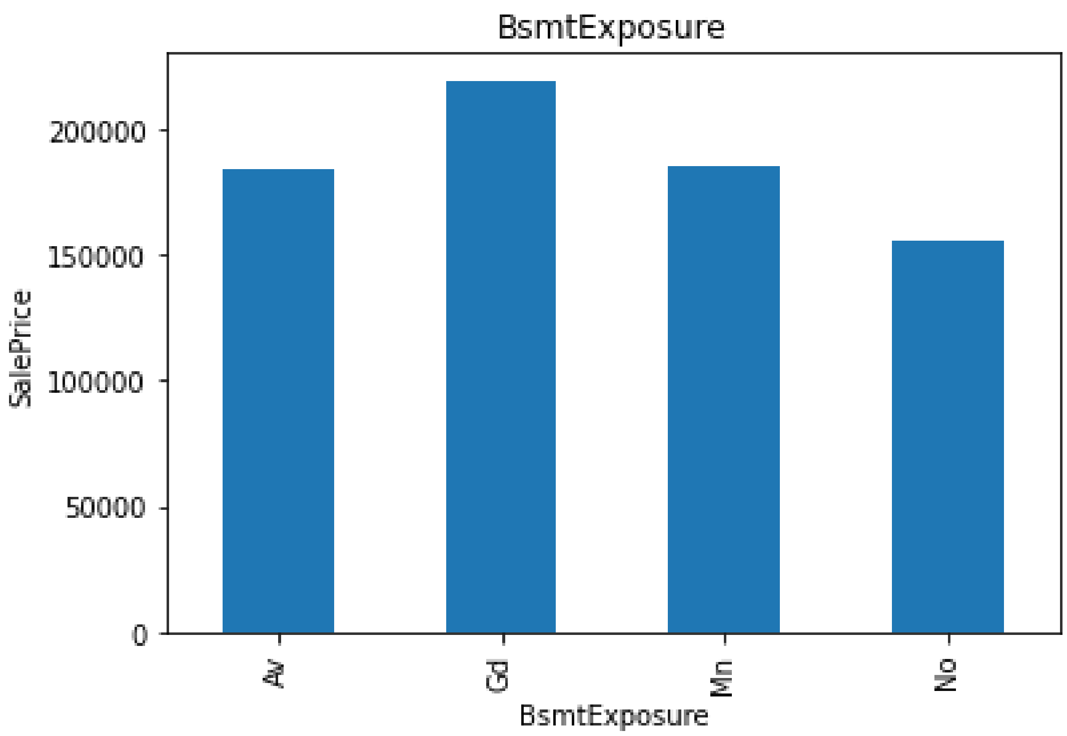
Salepice Vs Alley



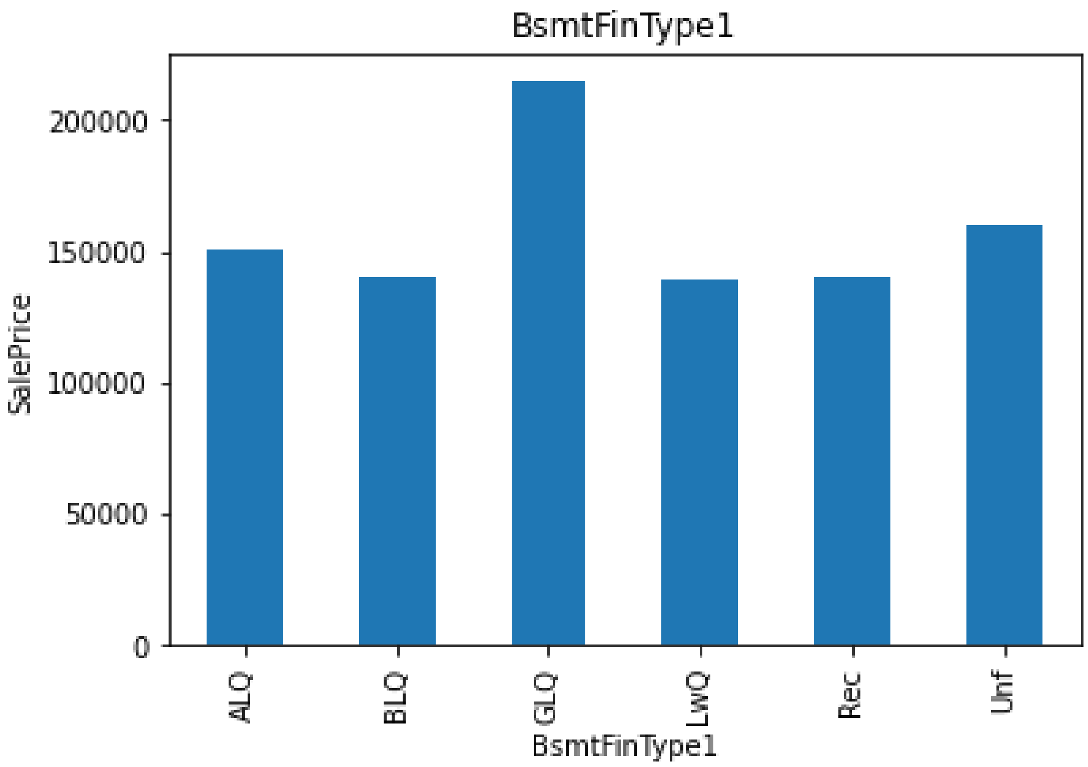
SalePrice Vs BldgType



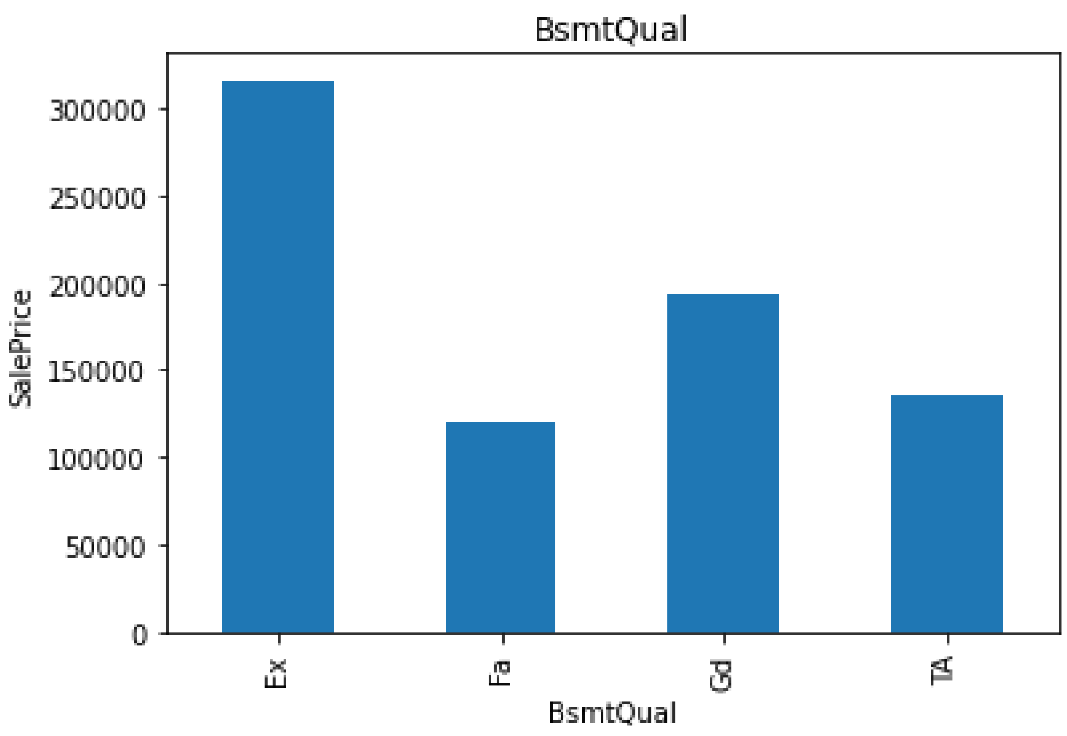
SalePrice Vs BSmt Cond



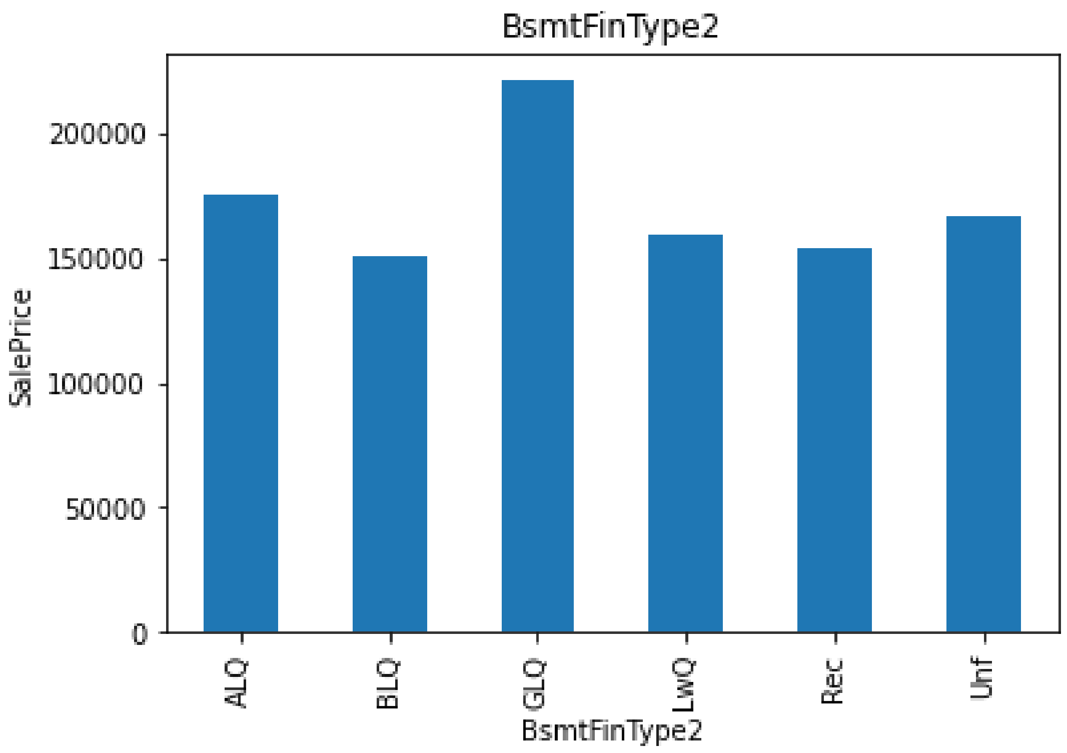
SalePrice Vs BsmtExposure



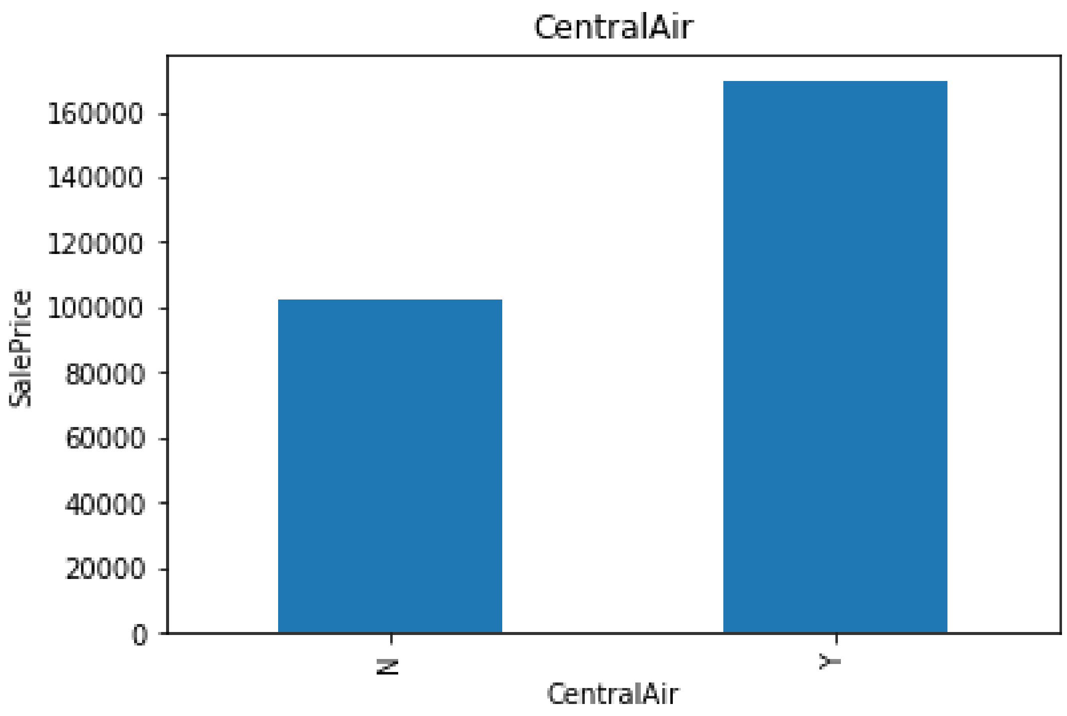
SalePrice VS BsmtFinType1



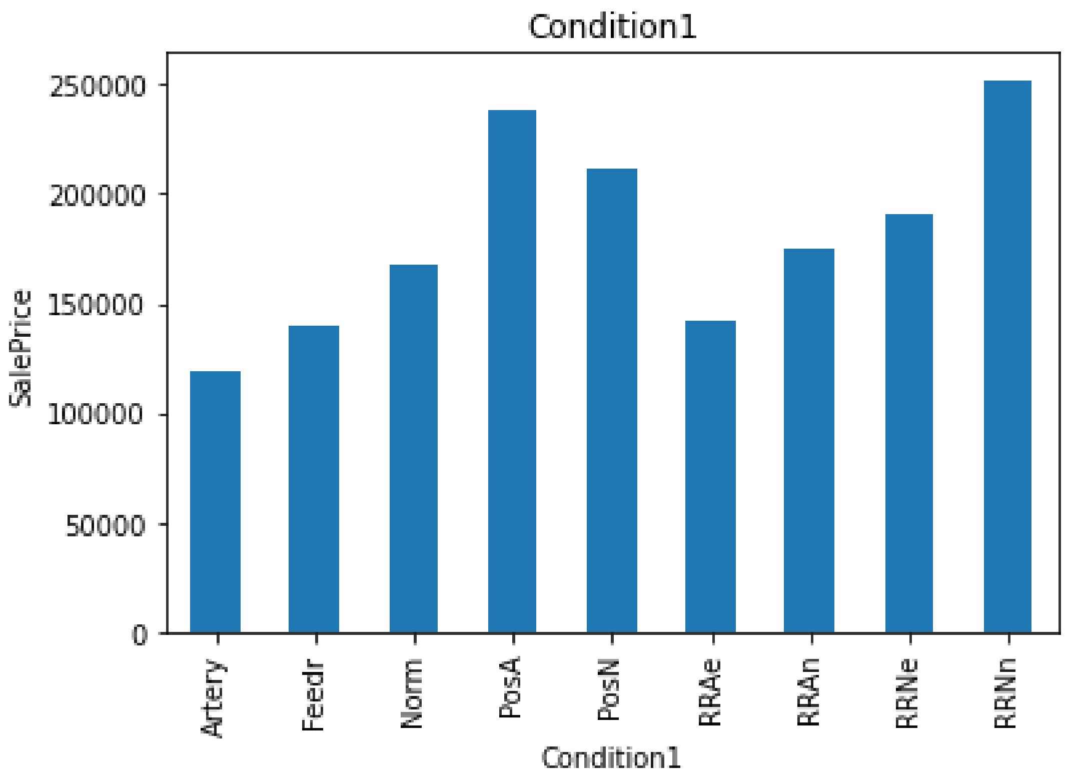
SalePrice VS BsmtQual



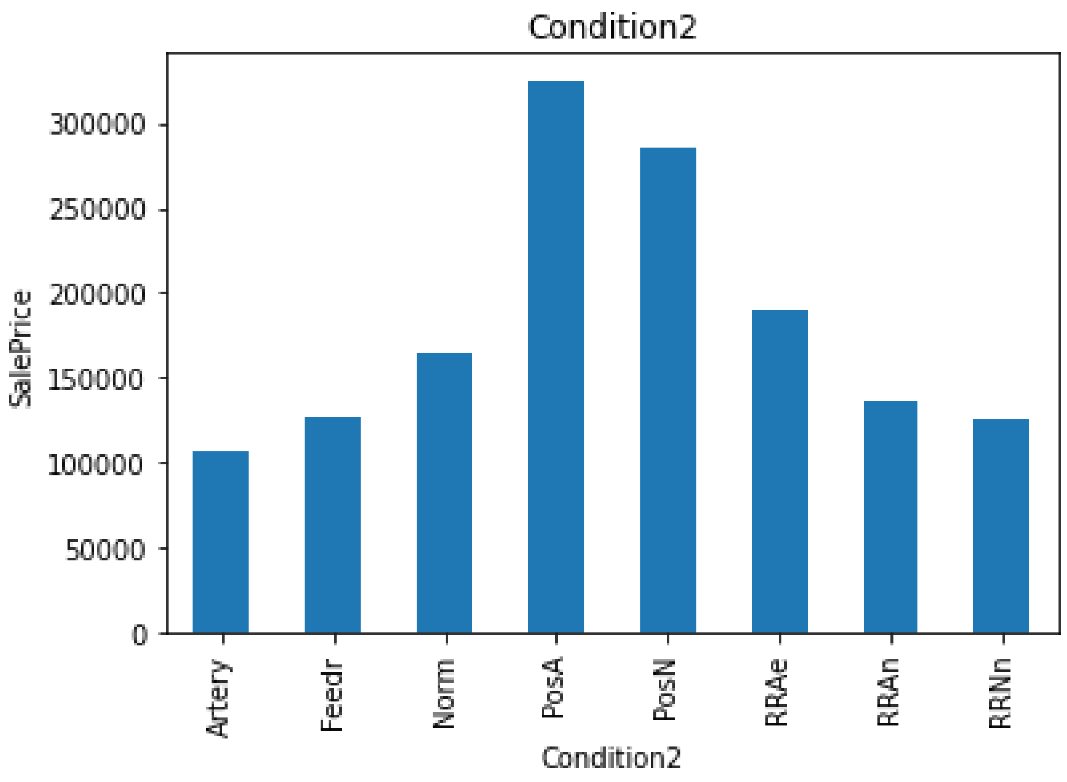
Saleprice vs BsmtFinType2



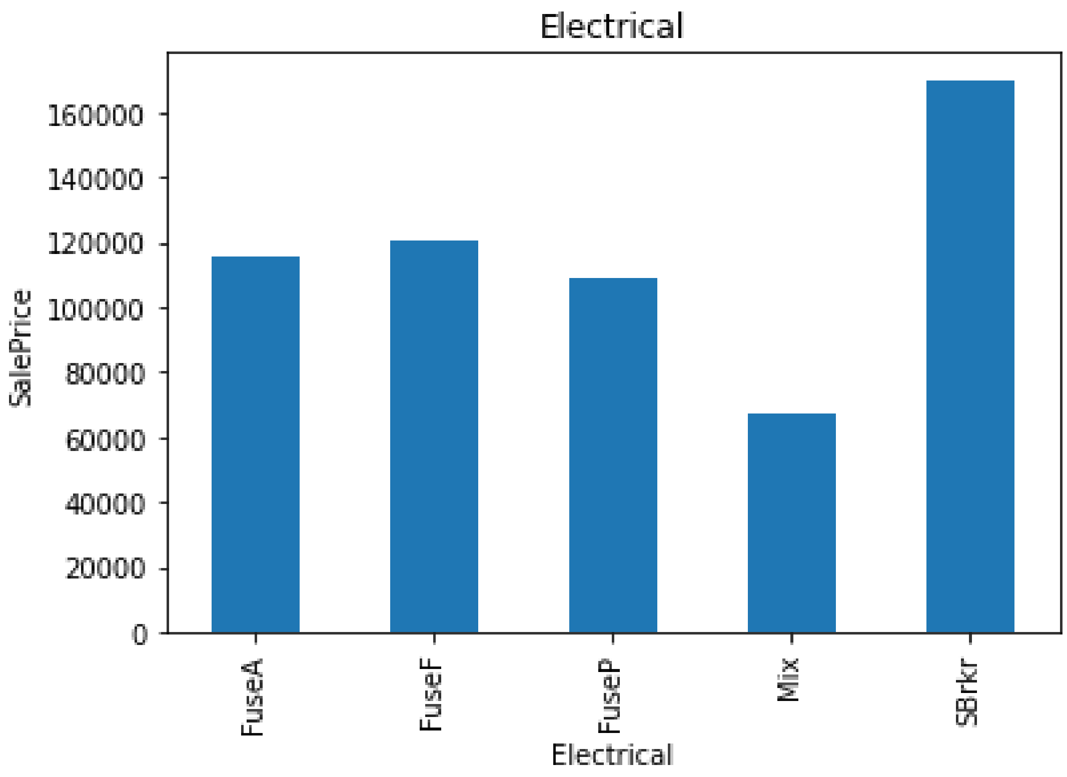
SalePrice Vs Central Air



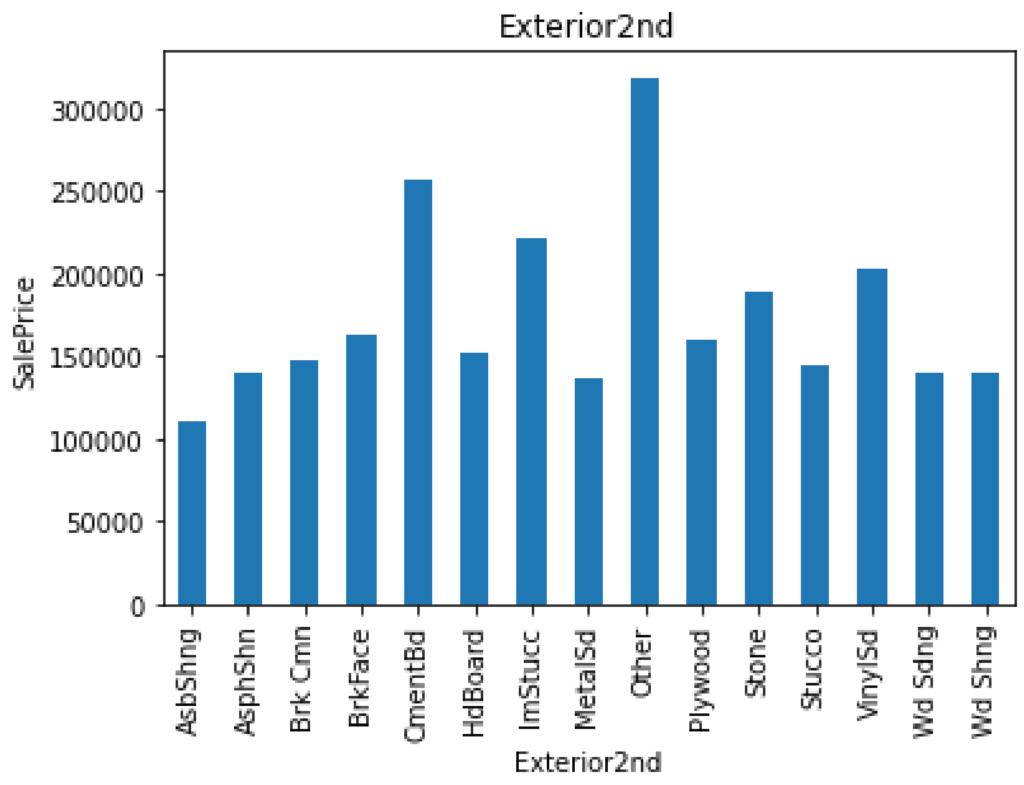
SalePrice Vs Condition1



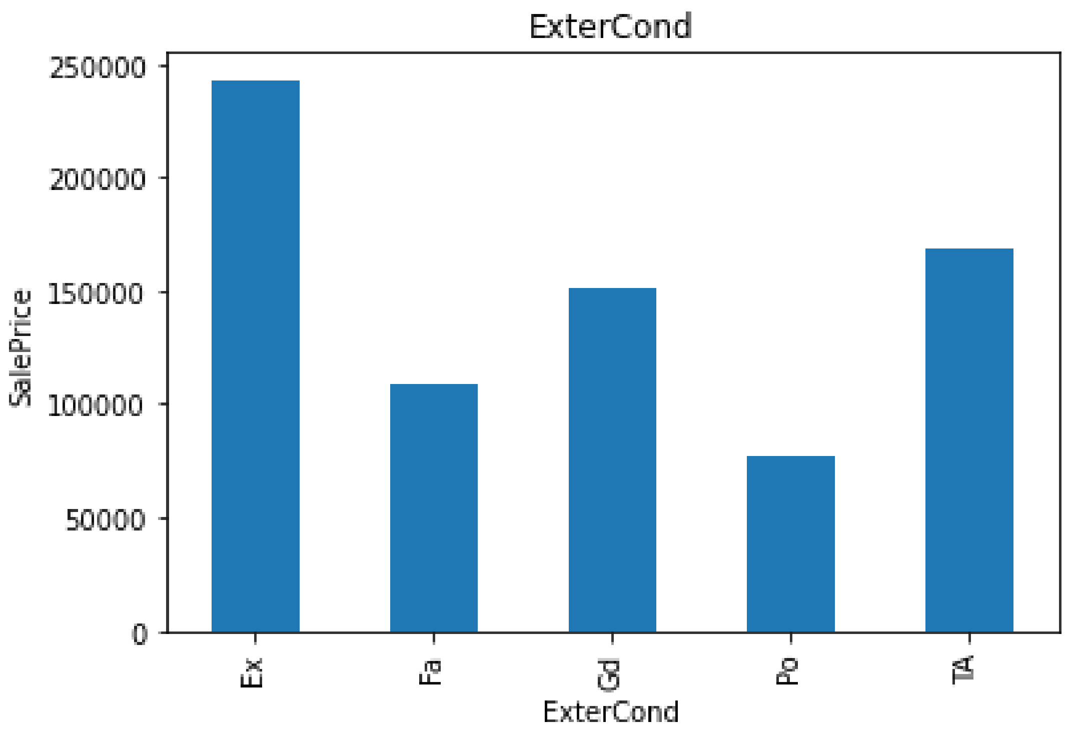
SalePrice Vs condition2



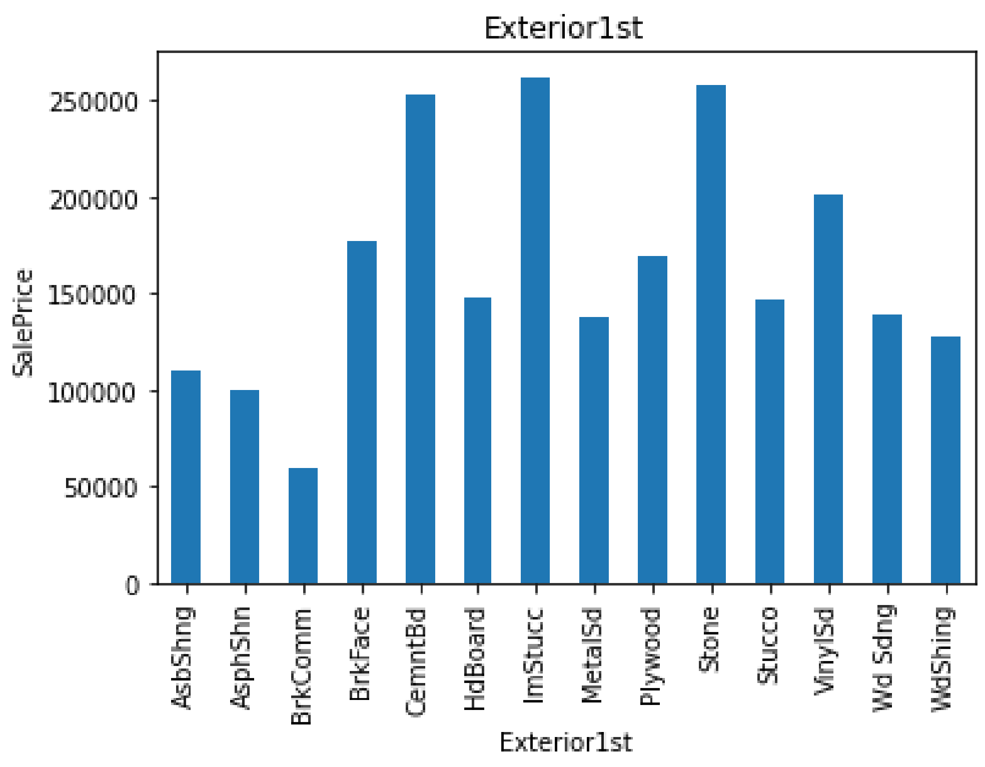
Saleprice Vs Electrical



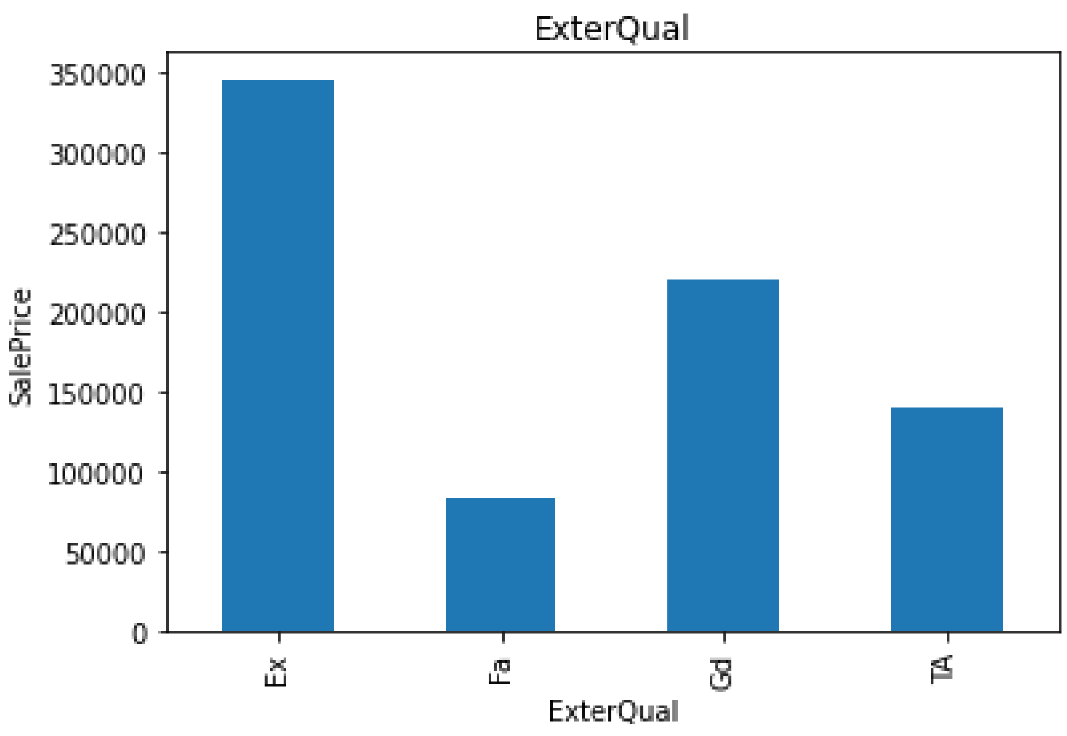
Saleprice Vs Exterior2nd



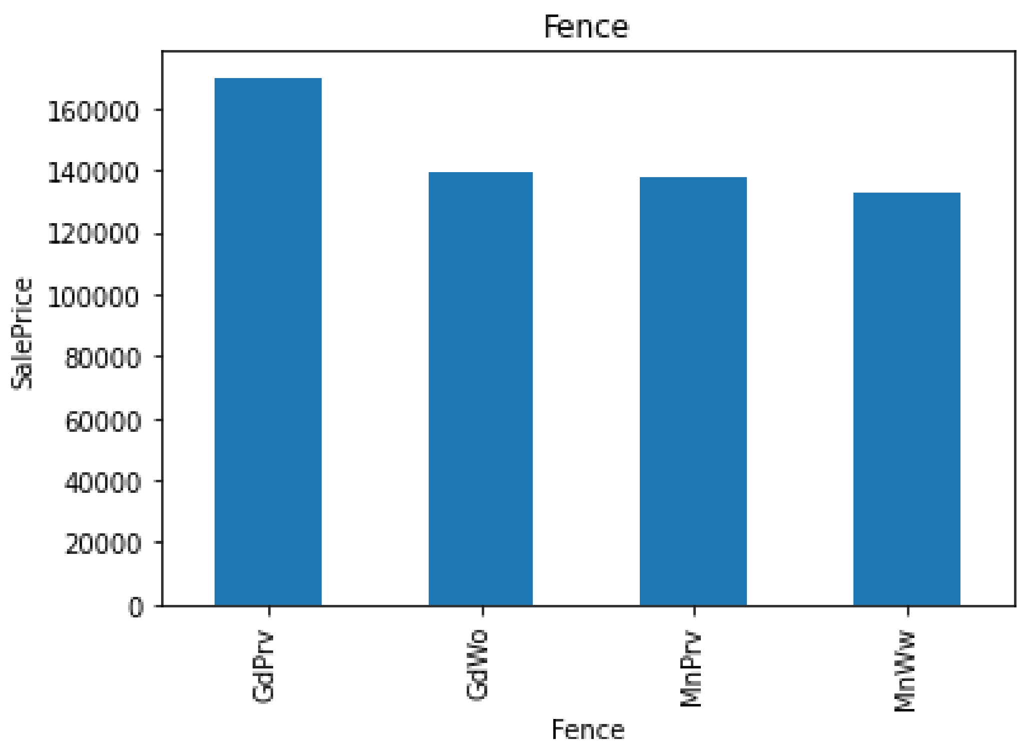
Saleprice Vs ExterCond



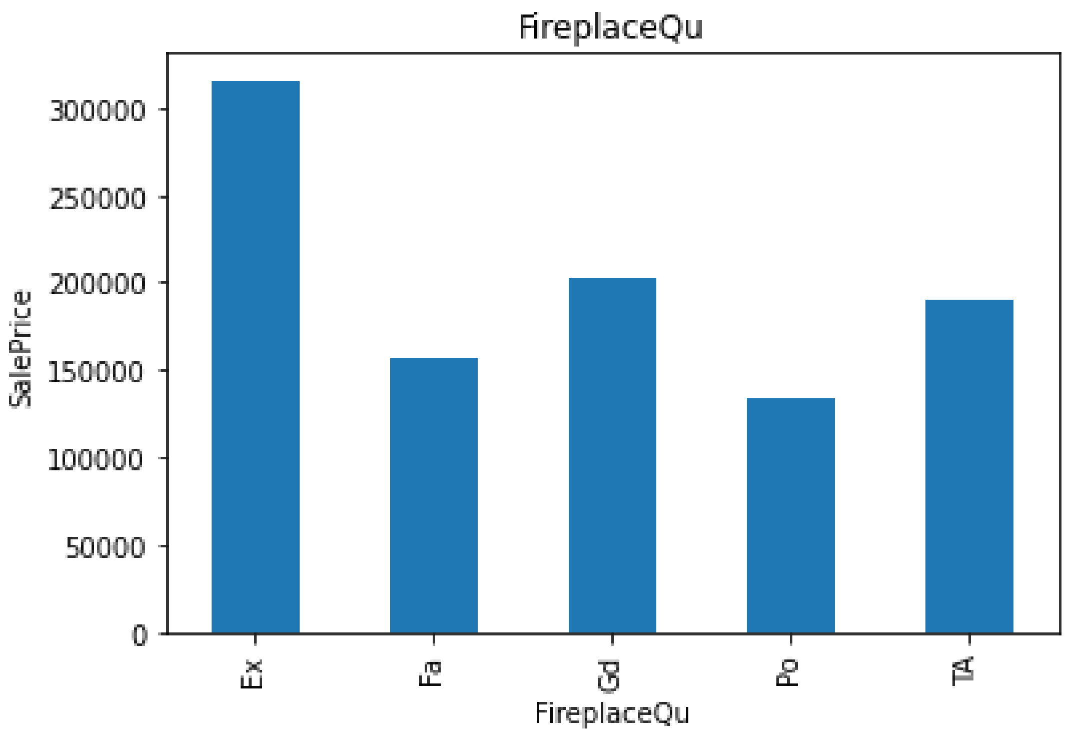
SalePrice Vs Exterior 1st



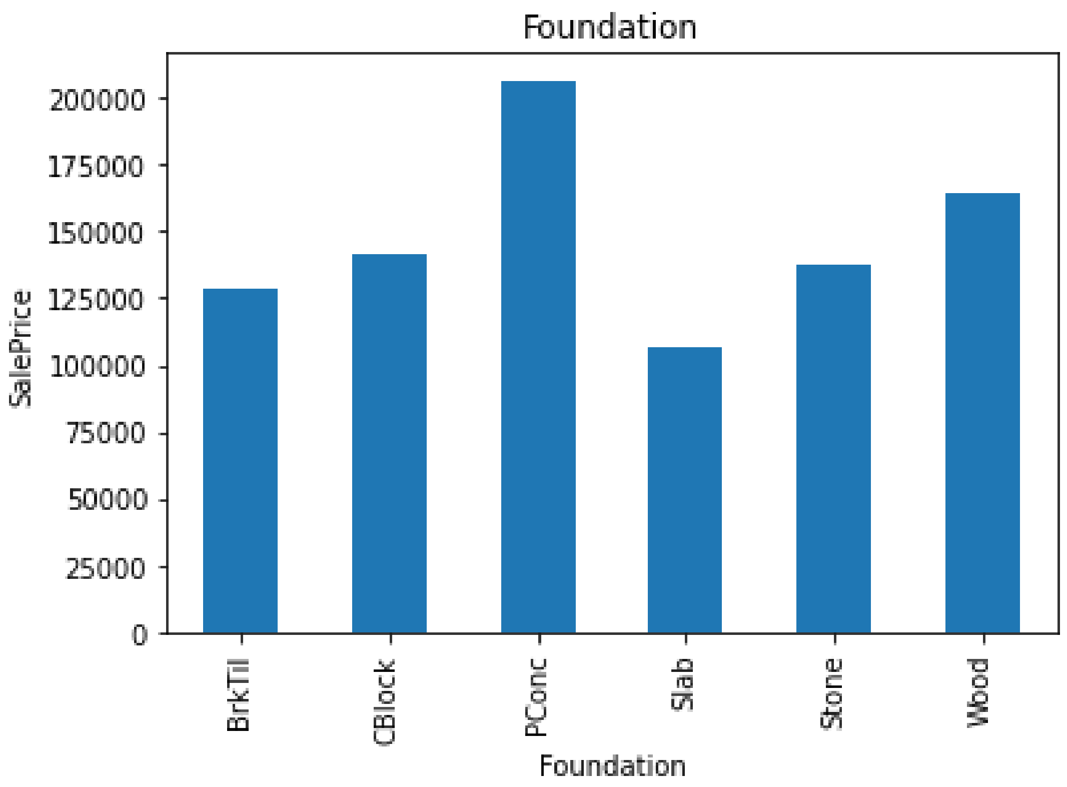
SalePrice VS ExteriorQual



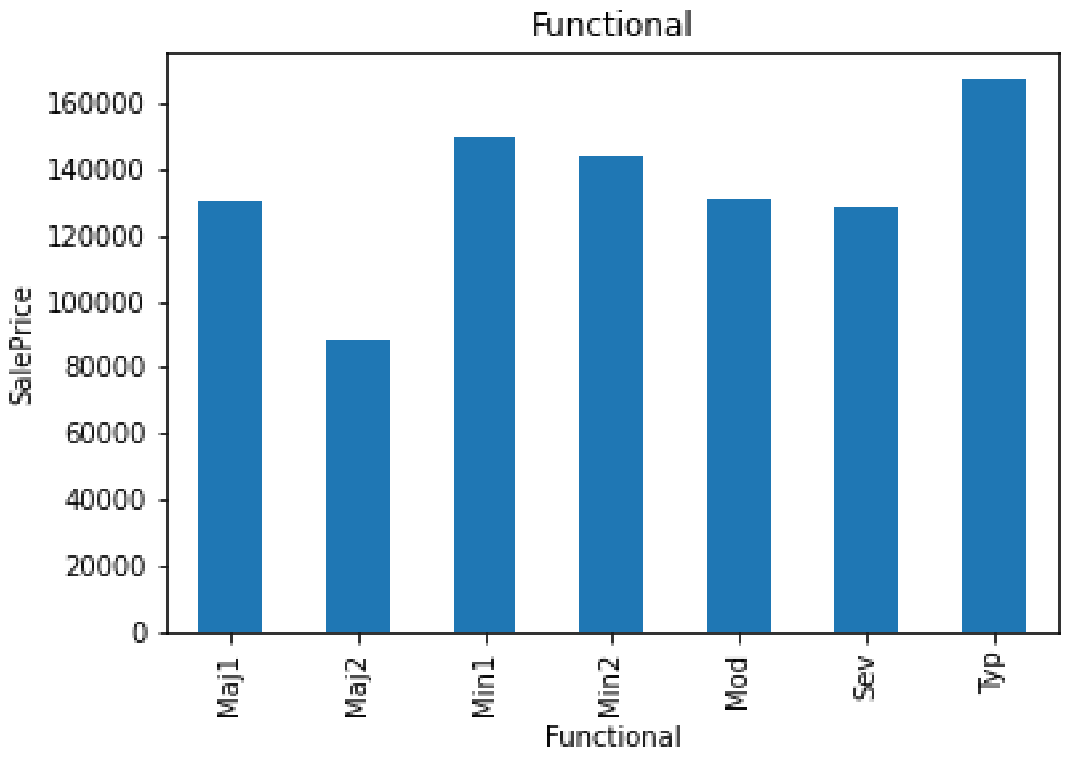
SalePrice Vs Fence



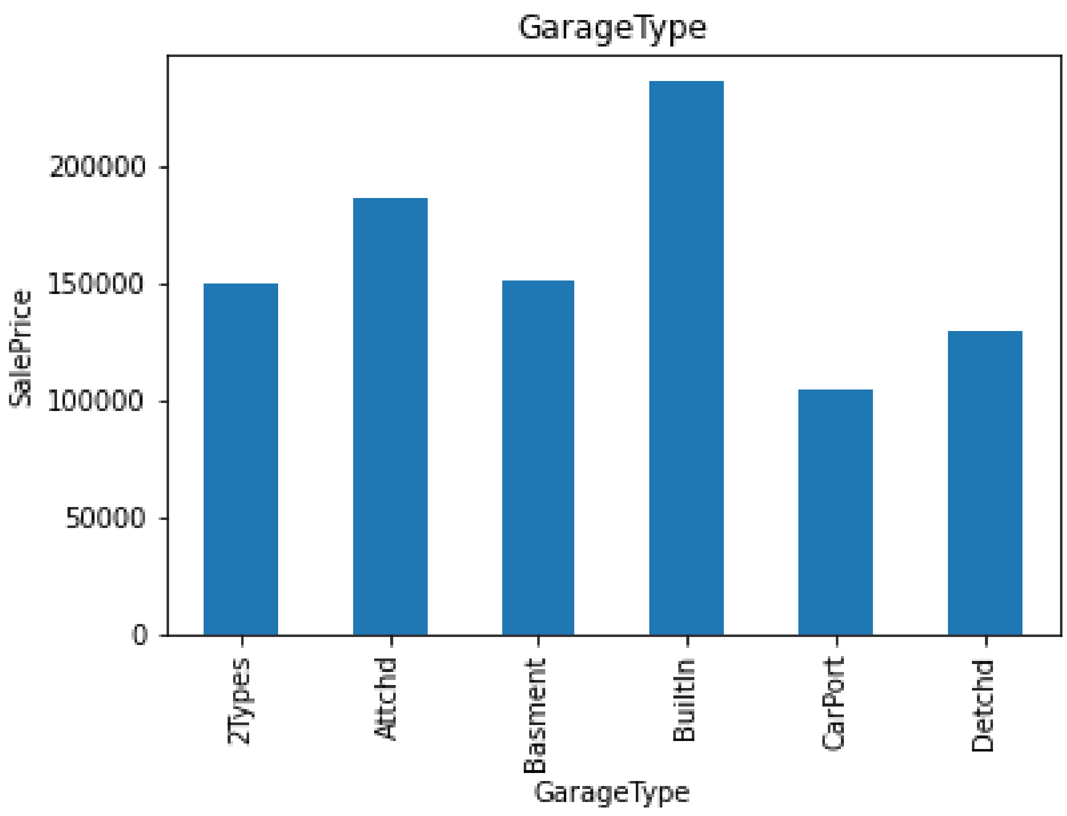
SalePrice Vs FirePlaceQU



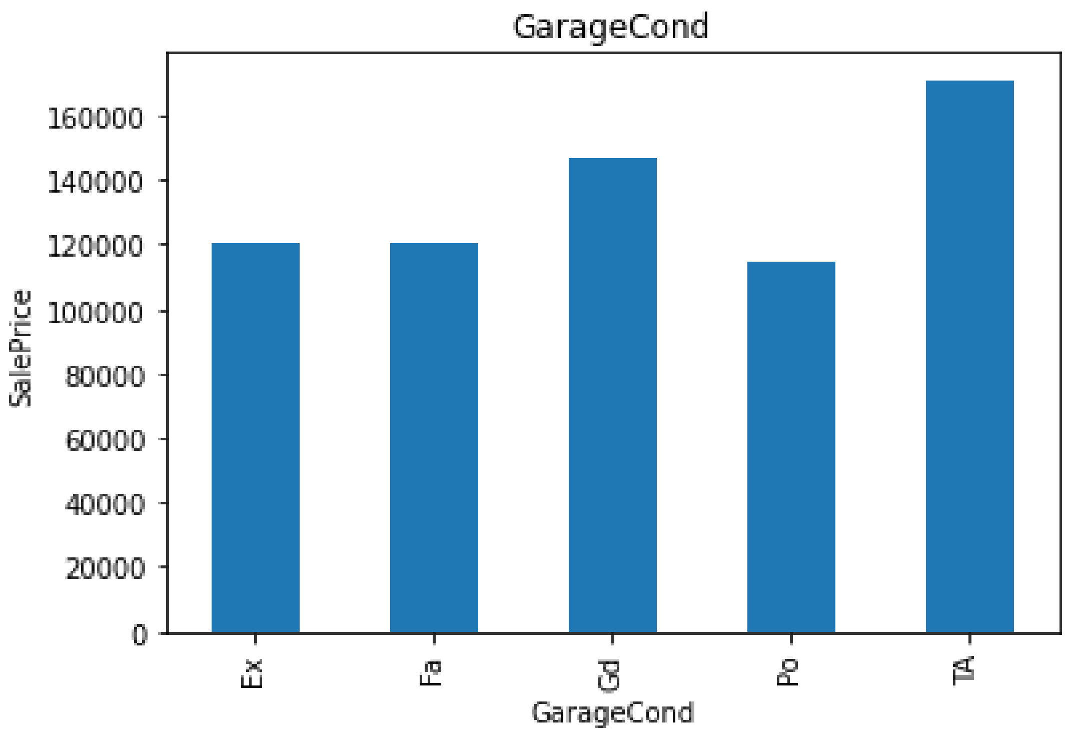
SalePrice Vs Foundation



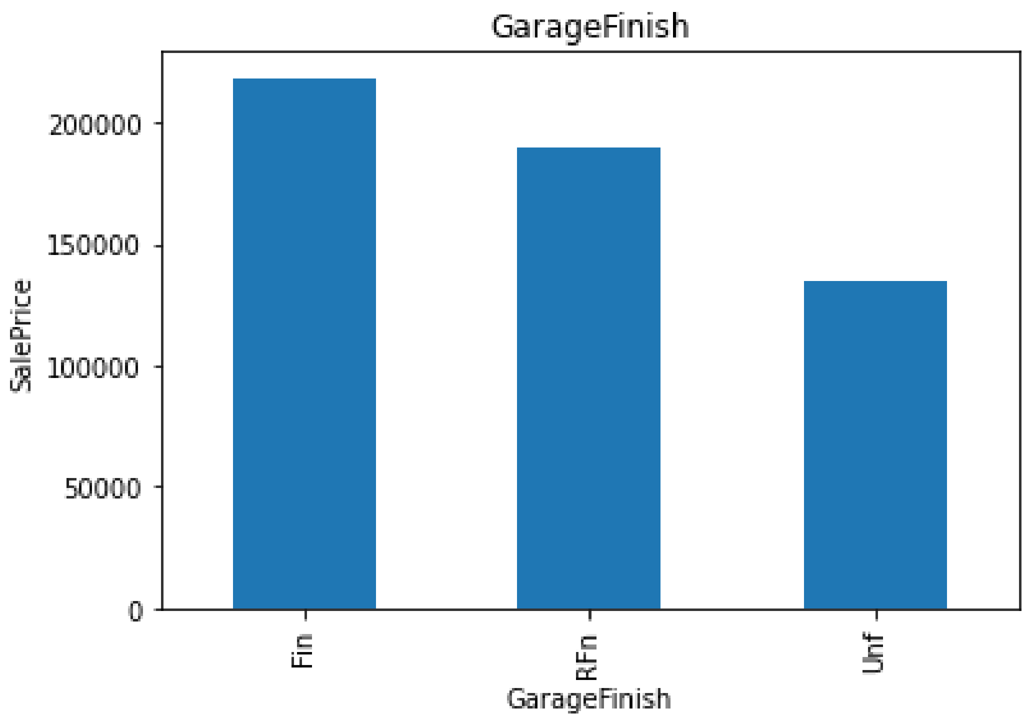
SalePrice Vs Functional



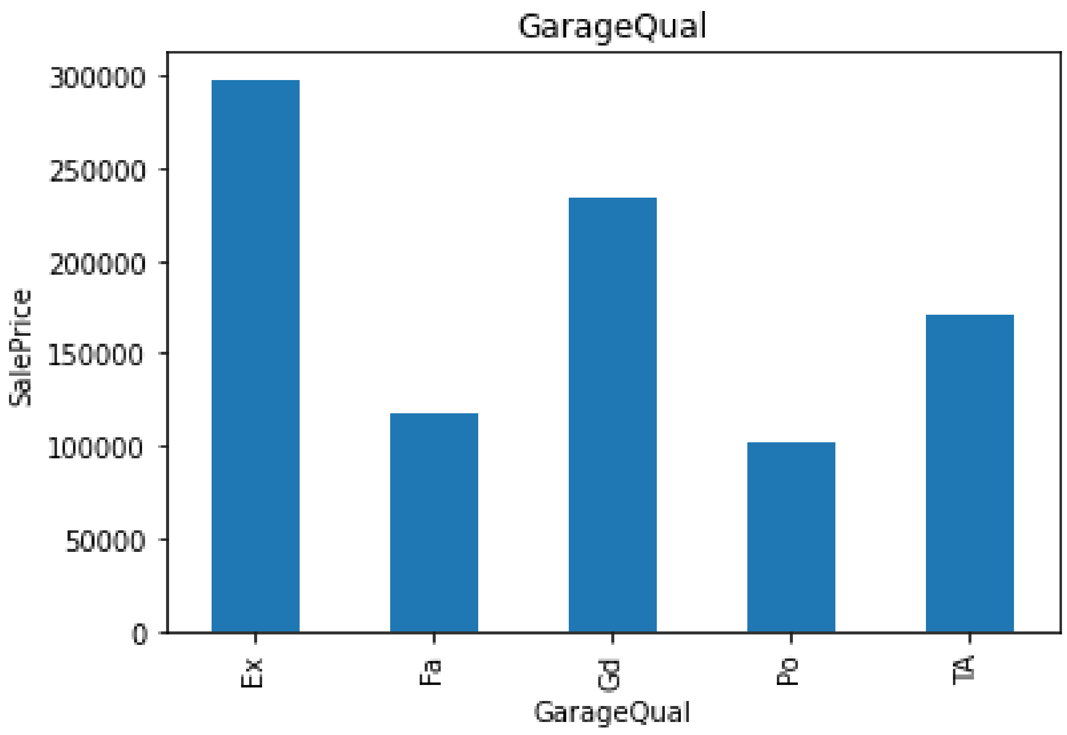
SalePrice Vs GarageType



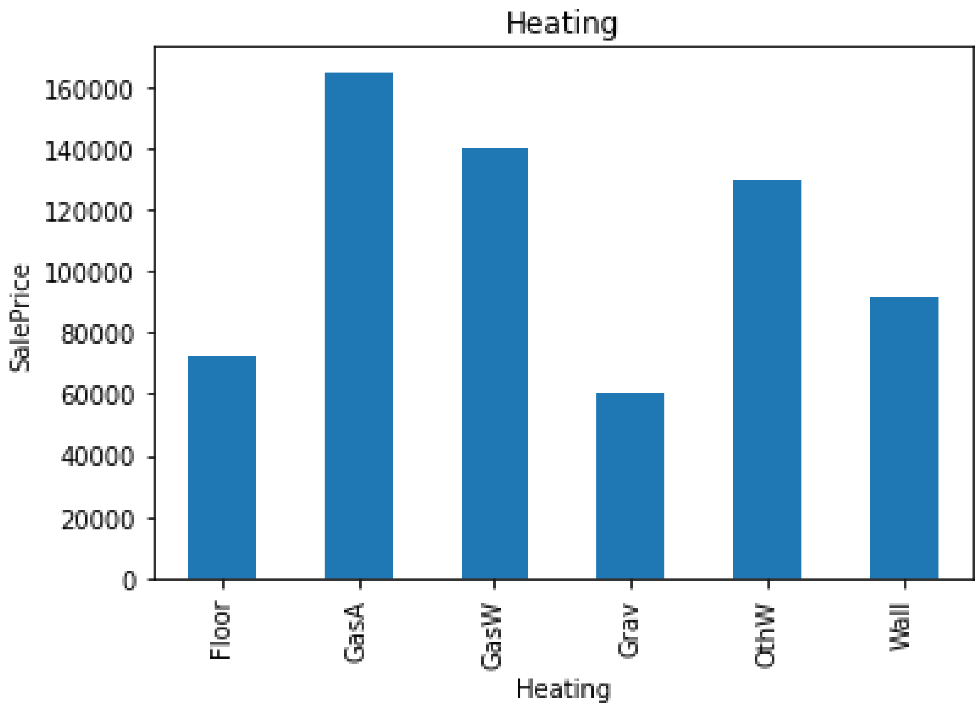
SalaryPrice Vs Garagecond



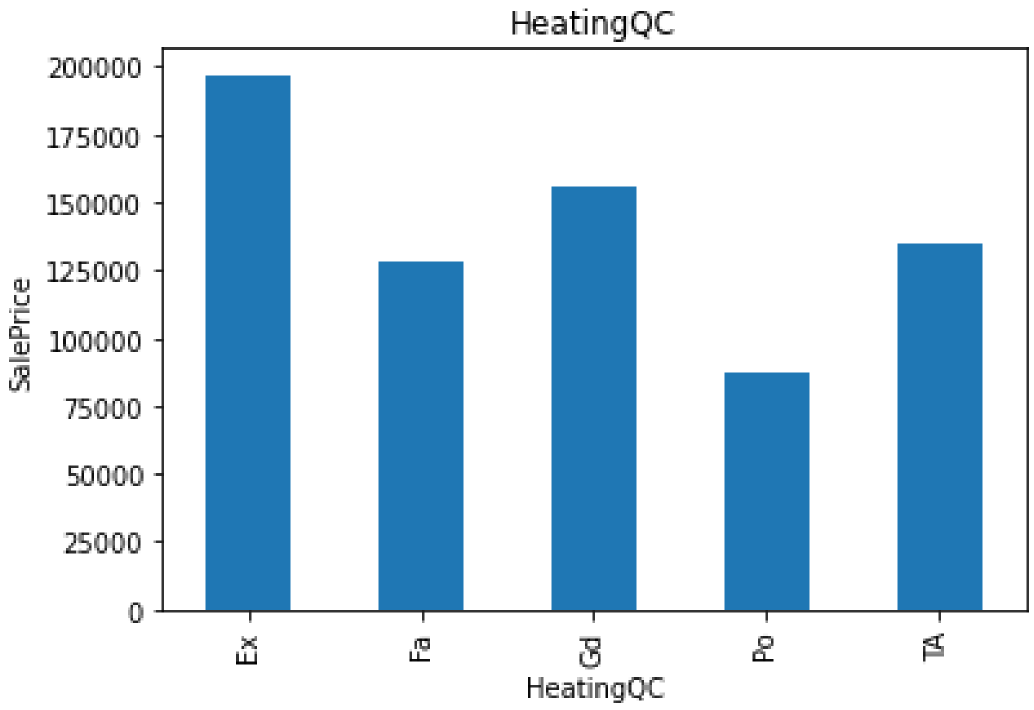
SalePrice Vs GargeFinish



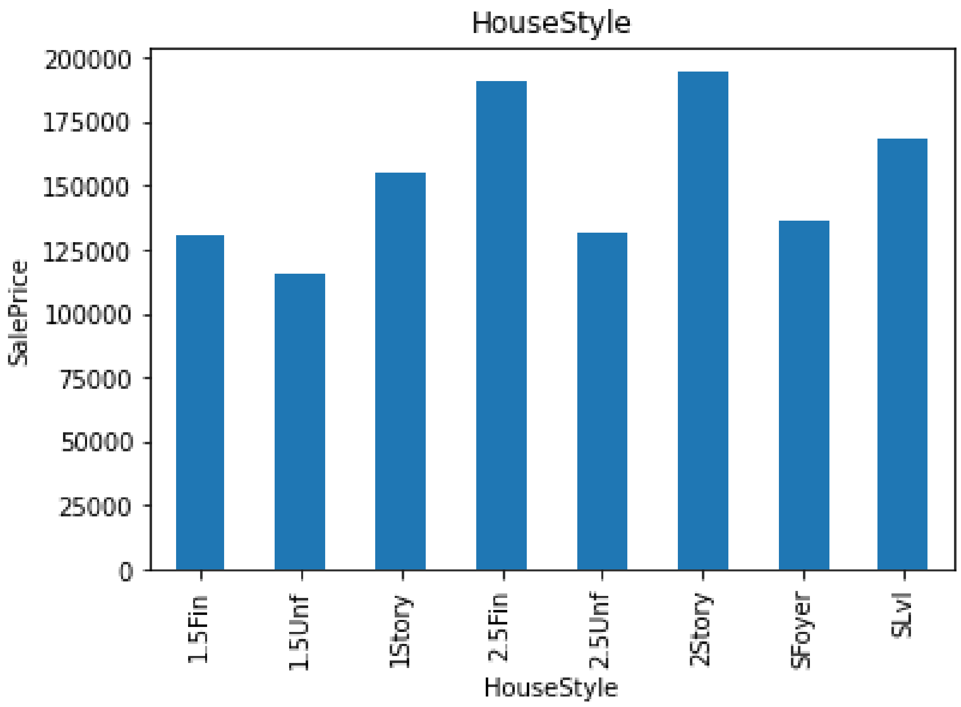
SalePrice Vs GarageQual



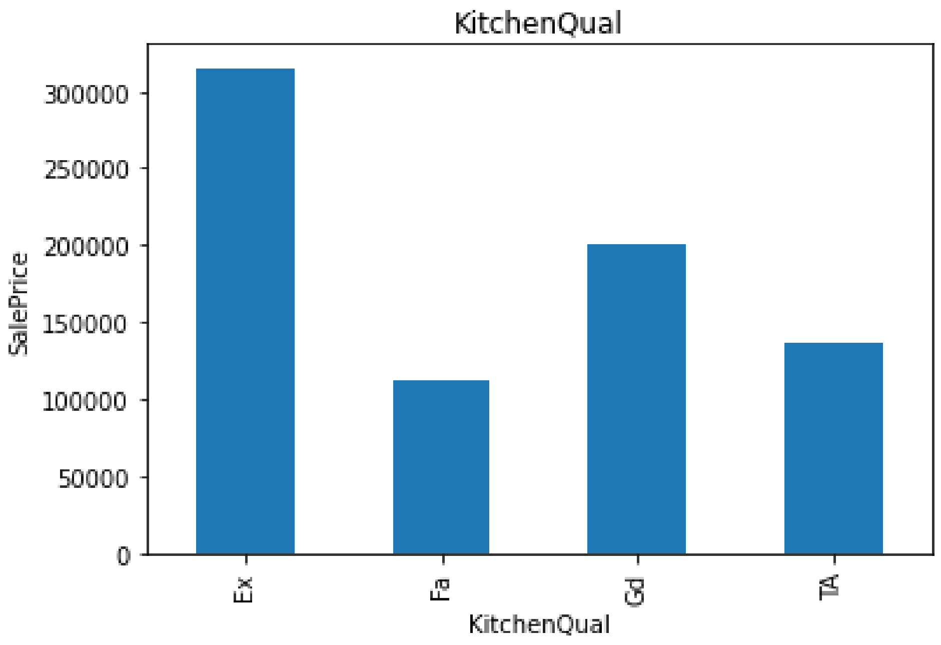
SalePrice Vs Heating



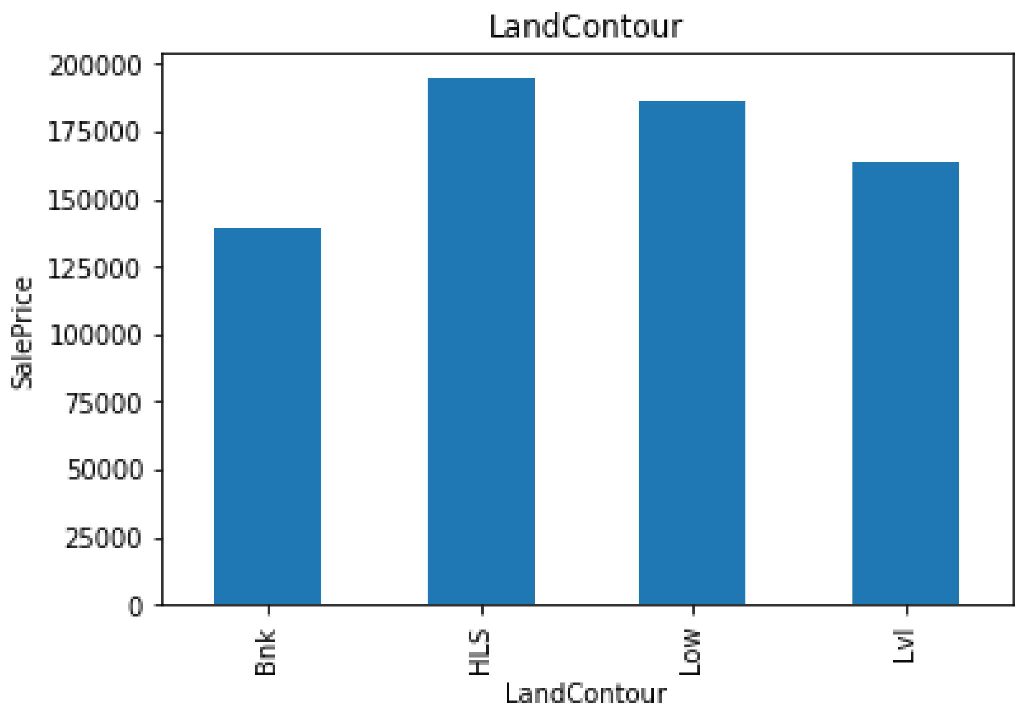
SalePrice Vs HeatingQC



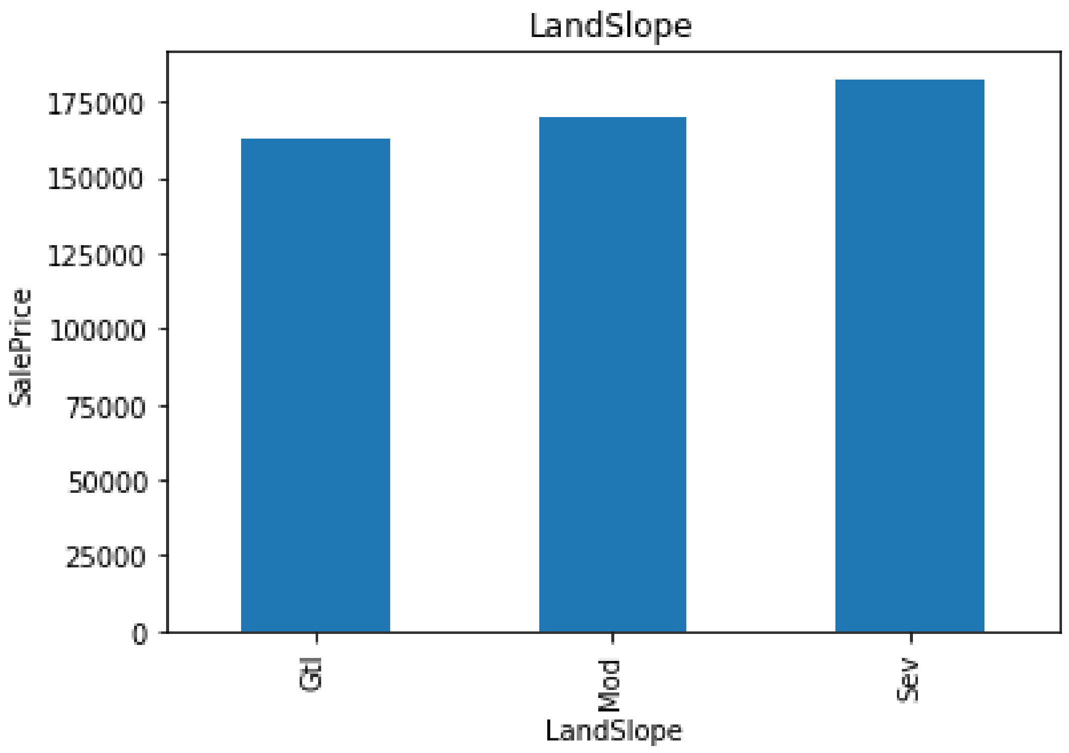
SalePrice Vs HouseStyle



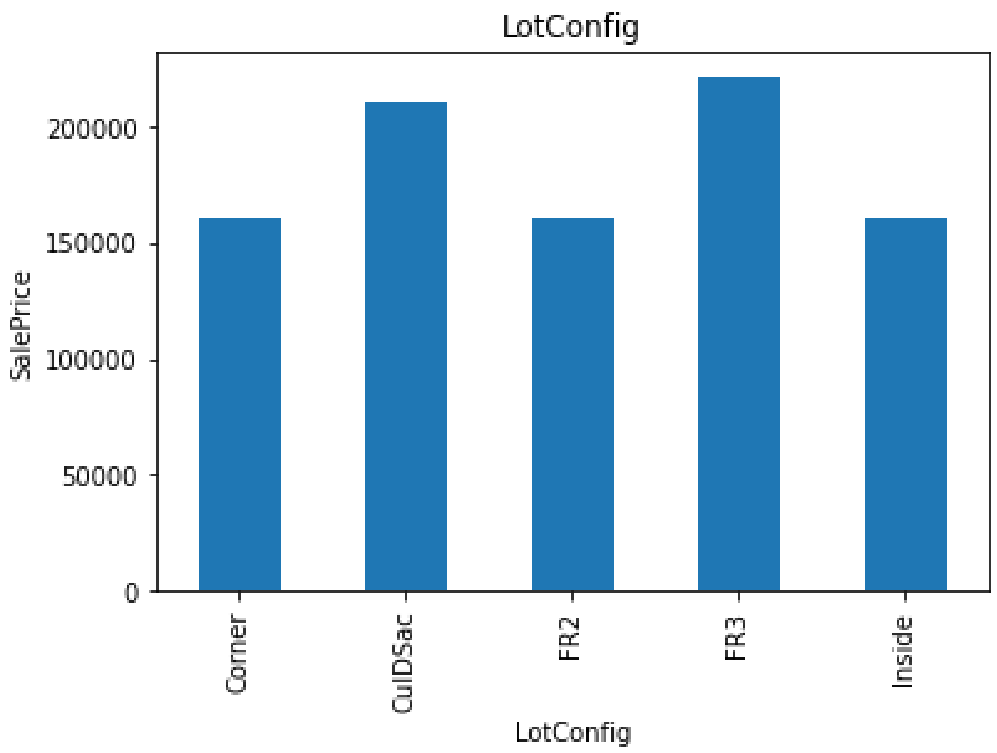
SalePrice VS KitchenQual



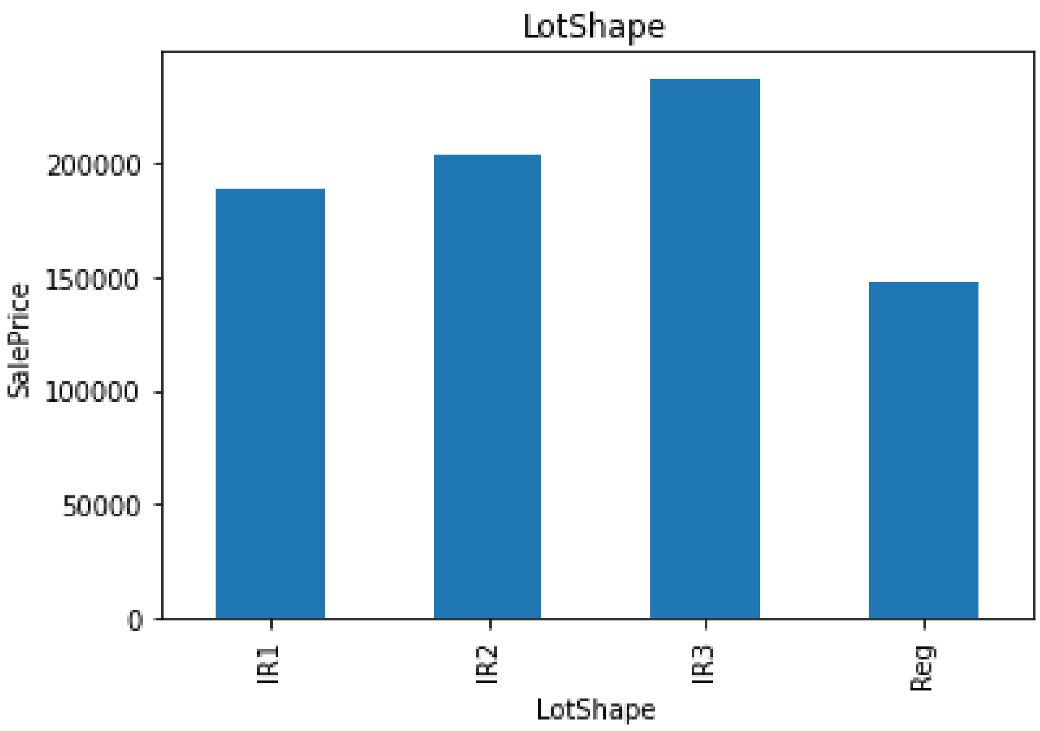
SalePrice Vs LandContour



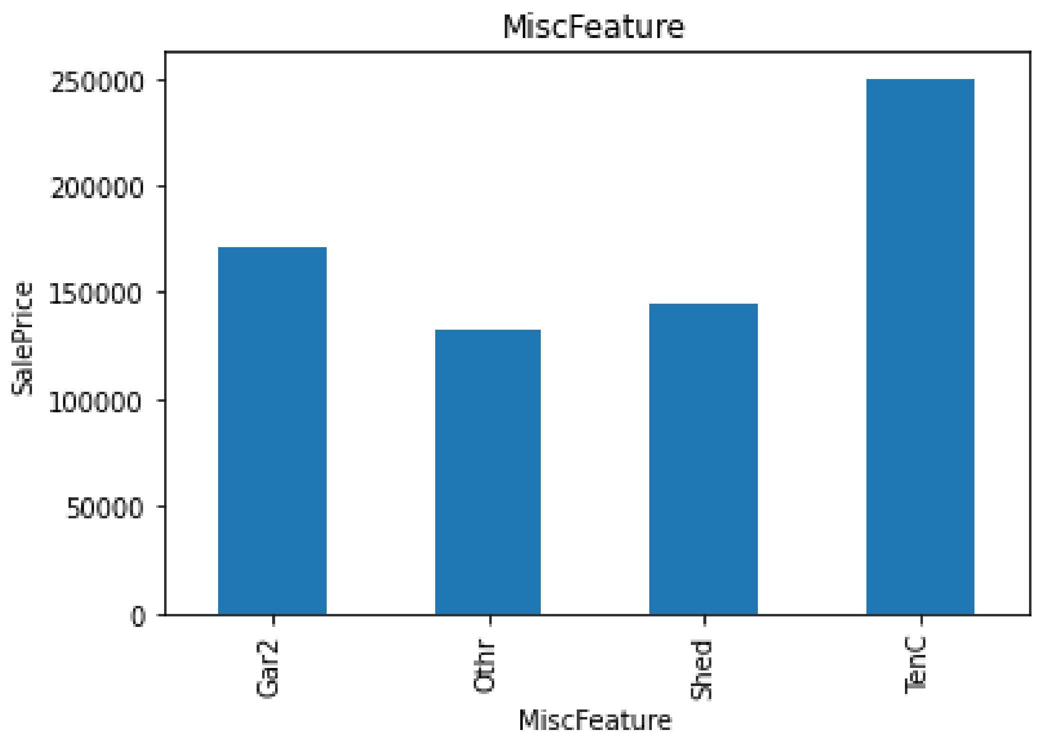
SalePrice VS LandSlope



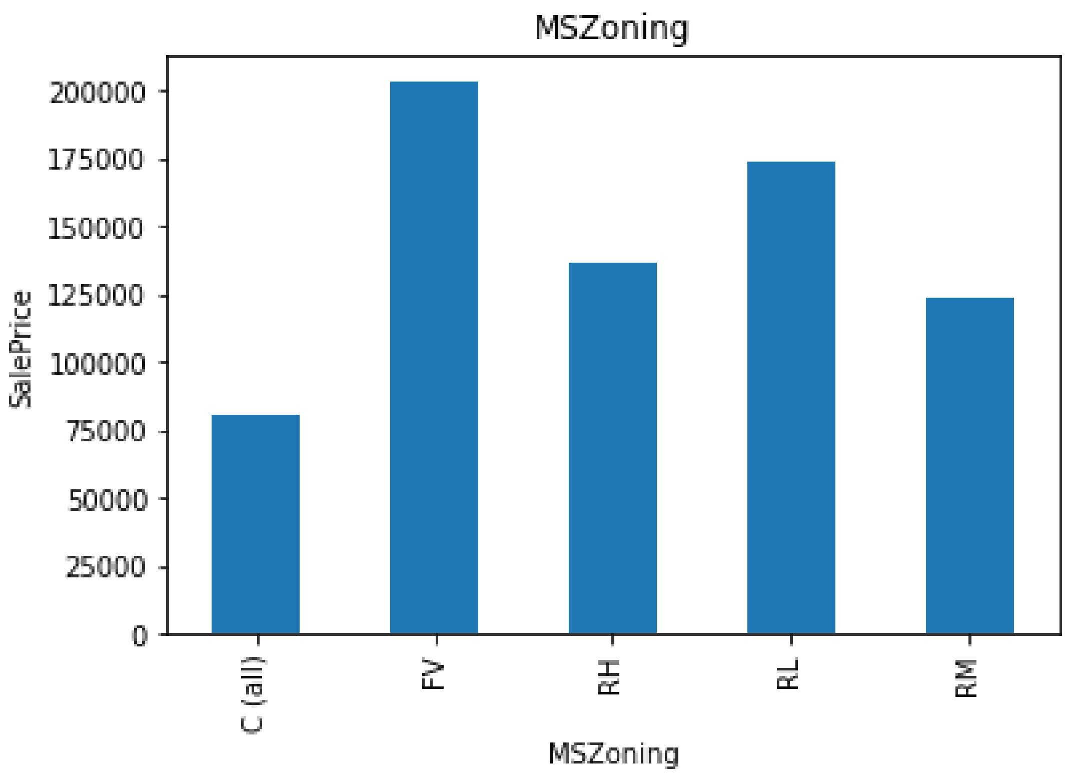
SalePrice Vs LotConfig



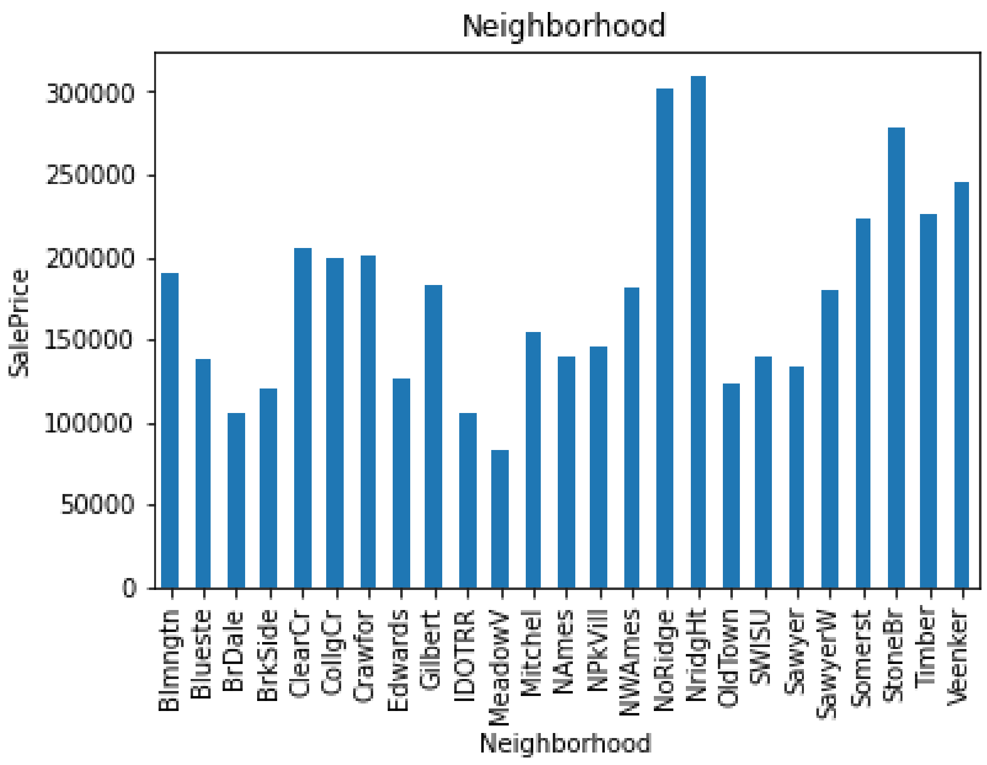
SalePrice Vs Lotshape



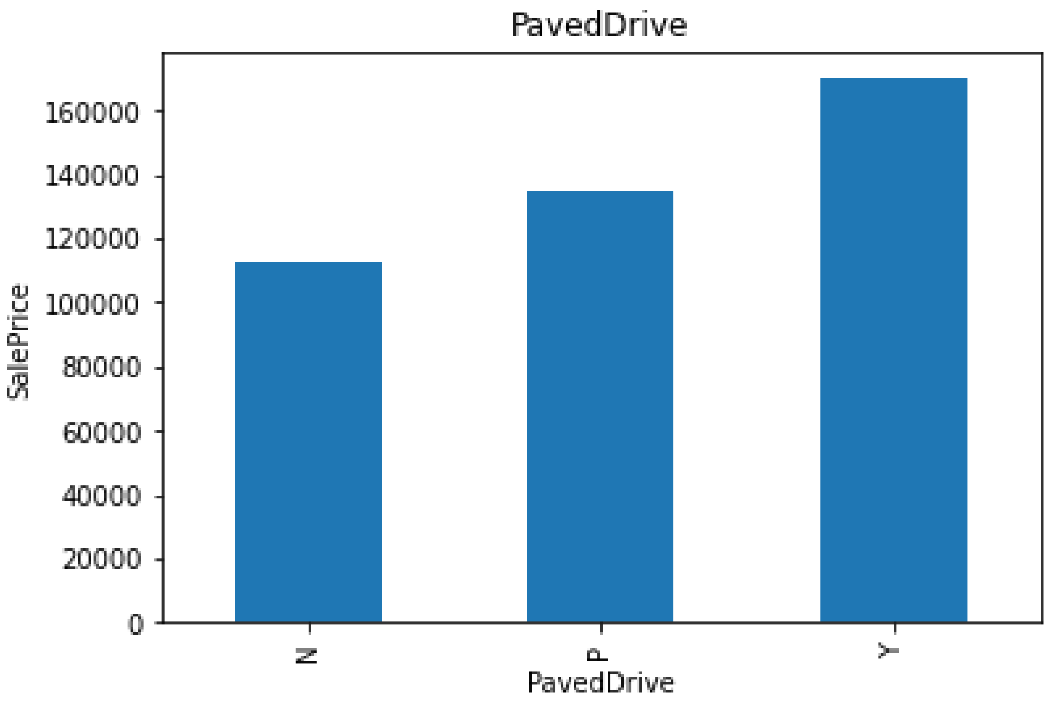
SalePrice Vs MiscFeature



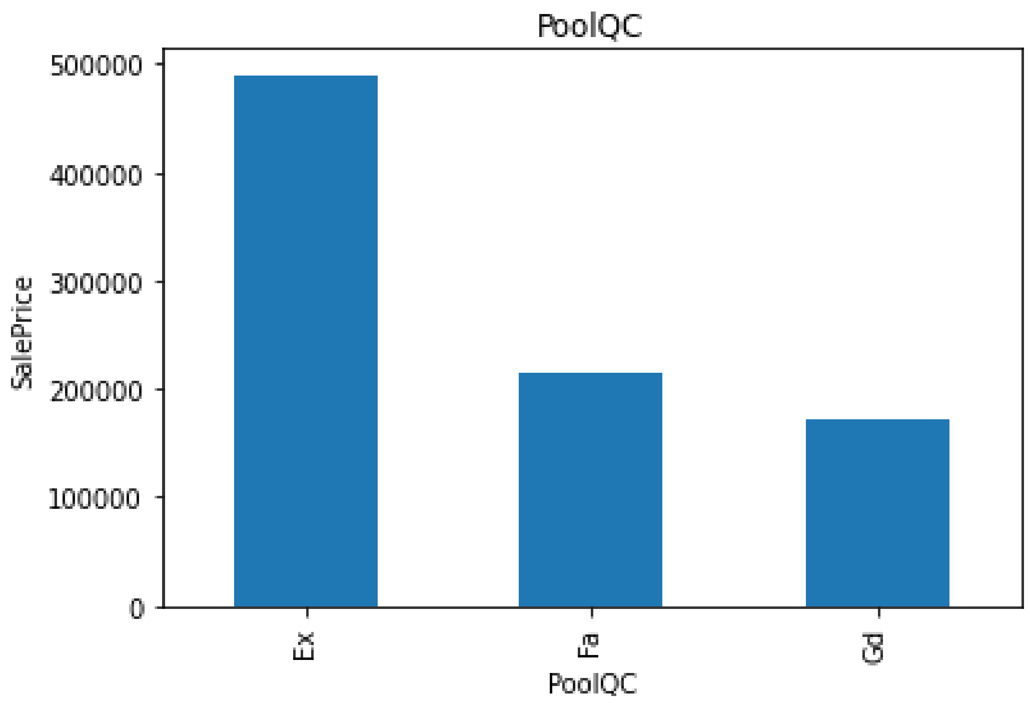
SalePrice VS MSZoning



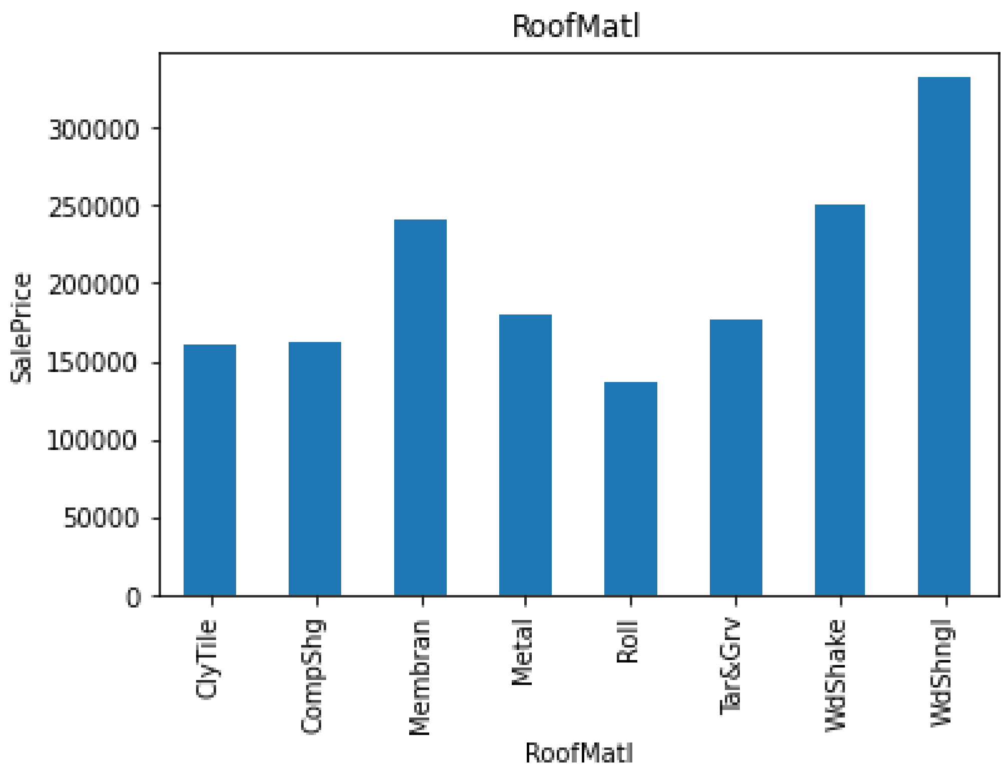
SalePrice Vs Neighborhood



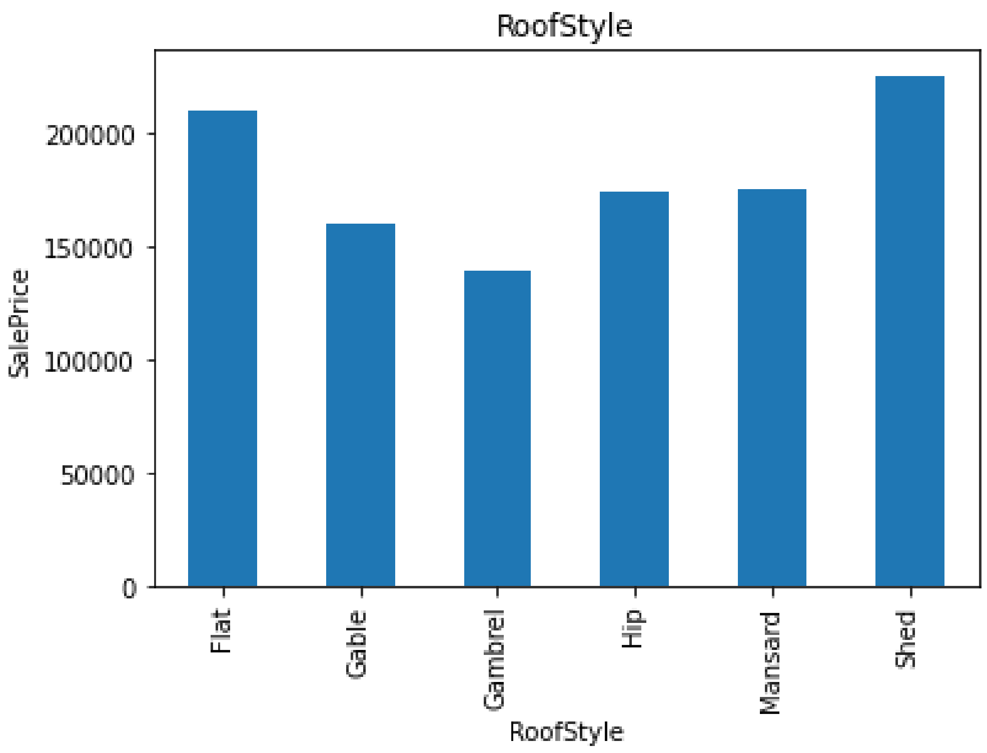
SalePrice Vs PavedDrive



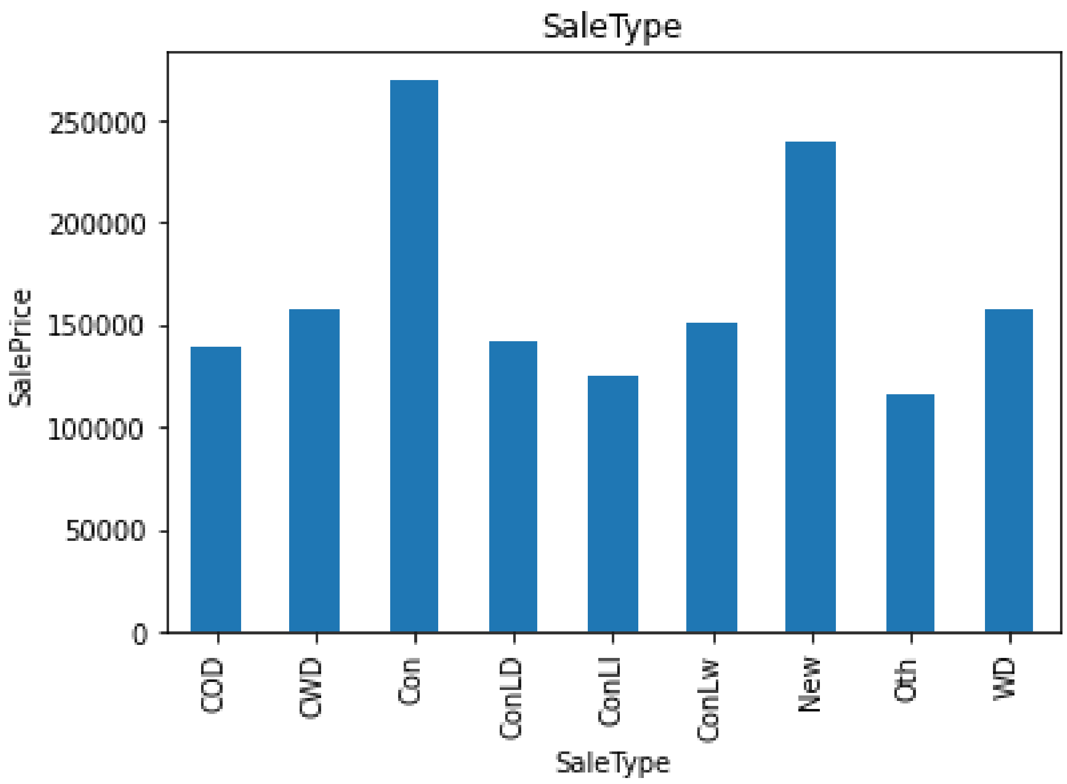
SalePrice VS PoolQC



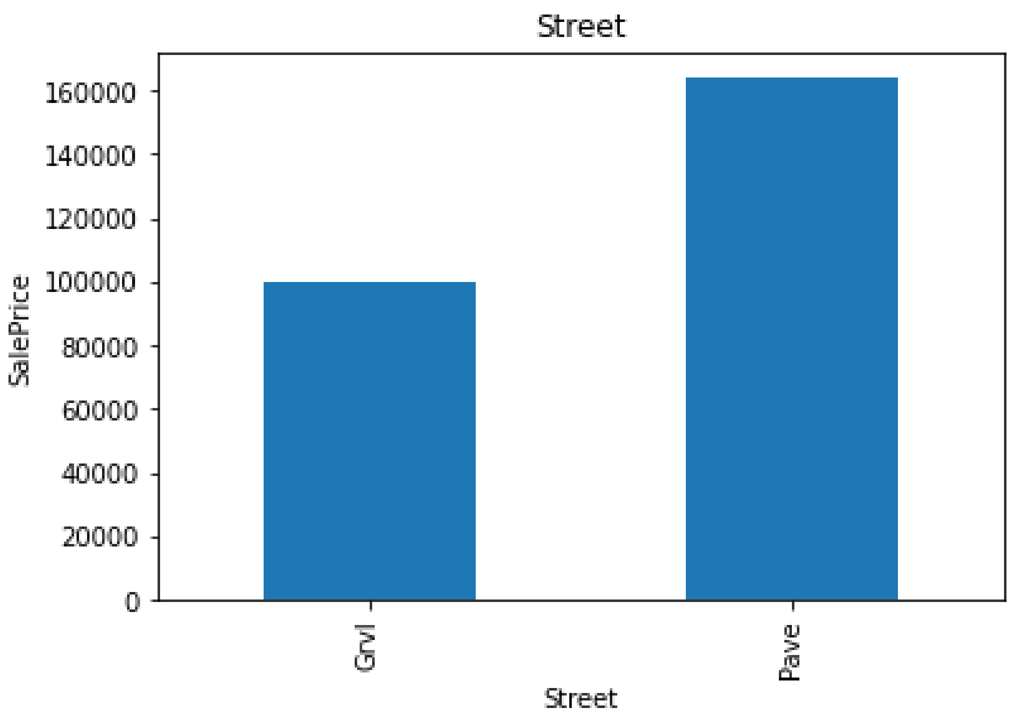
SalePrice VS RoofMati



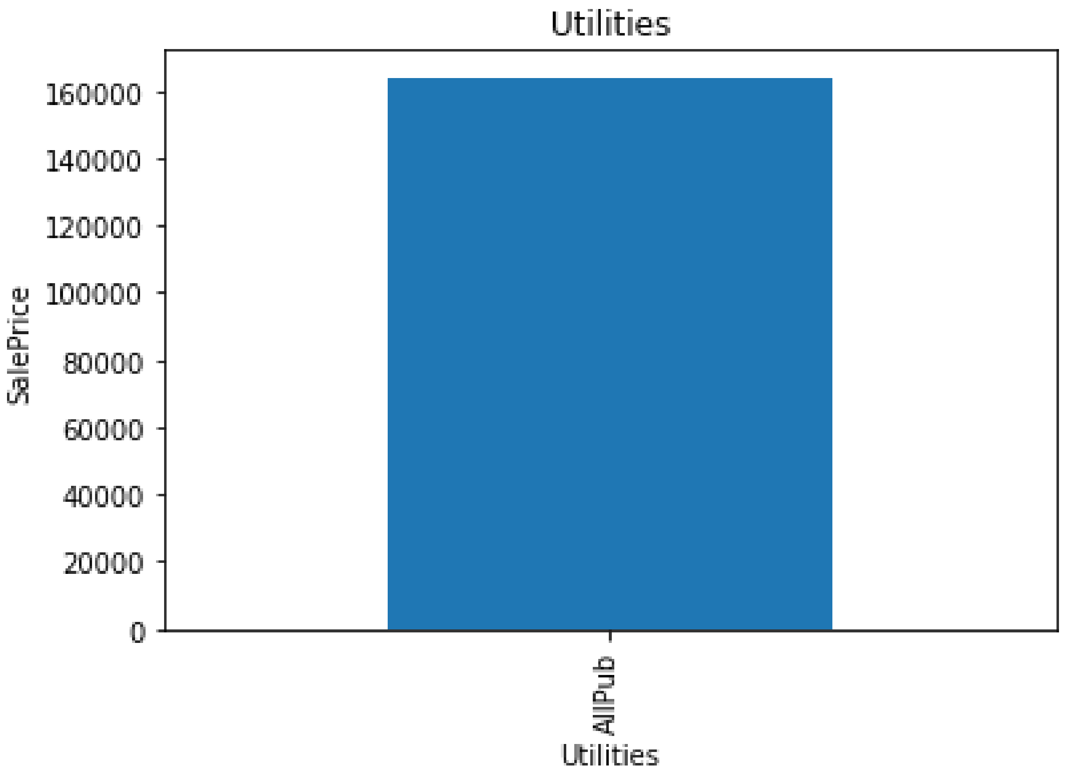
SalePrice VS RoofStyle



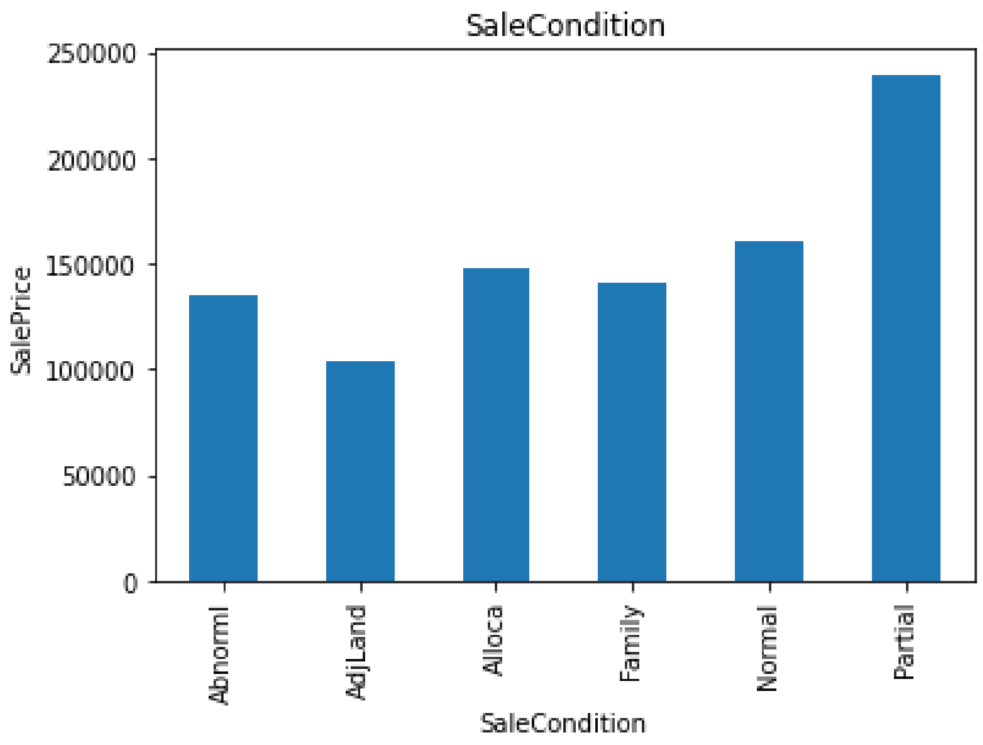
SalePrice VS SaleType



SalePrice Vs Street



SalePrice Vs utilities



```
# importing modules
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.linear_model import Ridge
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn import metrics
```

#Splitting the data into input and output variable.

```
x=df.drop(columns=['SalePrice','Id'],axis=1)

y=data['SalePrice']
```

#Splitting the data into training and testing data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.25,random_state=42)
```

Regression Techniques used:-

1.Linear Regression

2. SVR

3.Ridge Regression

4.AdaboostRegression

5.Random Forest Regression

Checking cross val score

Here our Ridge Regression model performs well. So we will do HyperParameter Tuning for Ridge Regression

```
# passing the created parameters to GridSearchCV
GCV = GridSearchCV(Ridge(), param_grid, cv=kfold)
GCV.fit(x_train, y_train)
```

```
# building the model with best parameters
tuned_model = Ridge(alpha=0.1, fit_intercept='True', normalize='True',
solver='sparse_cg')
tuned_model.fit(x_train, y_train)
prediction = tuned_model.predict(x_test)

# printing r2 score for the target variable Sale Price
print("r2 score: {:.4f}".format(metrics.r2_score(y_test, prediction)))
```

Saving our model

```
# importing joblib
import joblib
joblib.dump(tuned_model, 'PFAHOUSINGPROJECT.pkl')
```

Loading test csv file

```
# predicting the sale price of the test dataset using the model we used in
hyperparameter tuning.
prediction1 = tuned_model.predict(test_new)
```

Conclusion

Ridge Regression model is considered as the best model among 5 because of
Mean Absolute Error: 0.08378552638145562
Mean Squared Error: 0.014810336087915798
Root Mean Squared Error: 0.12169772425117817
r2 score: 0.9134148704413022