



## **PFA Housing Project**

Submitted by:

Mohammad Shahid Khan

## **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:-

```
missing_values=[x for x in df.columns if df[x].isnull().sum()>1]
print('Number of missing variable columns:', len(missing_values))
print("Missing values in the dataset : \n ", missing_values)
print("-"*125)
df[missing_values].head()
```

Observation:-

Number of missing variable columns: 18

Missing values in the dataset :

['LotFrontage', 'Alley', 'MasVnrType', 'MasVnrArea', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'FireplaceQu', 'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageQual', 'GarageCond', 'PoolQC', 'Fence', 'MiscFeature']

### 2. Checking the percentage of the missing values

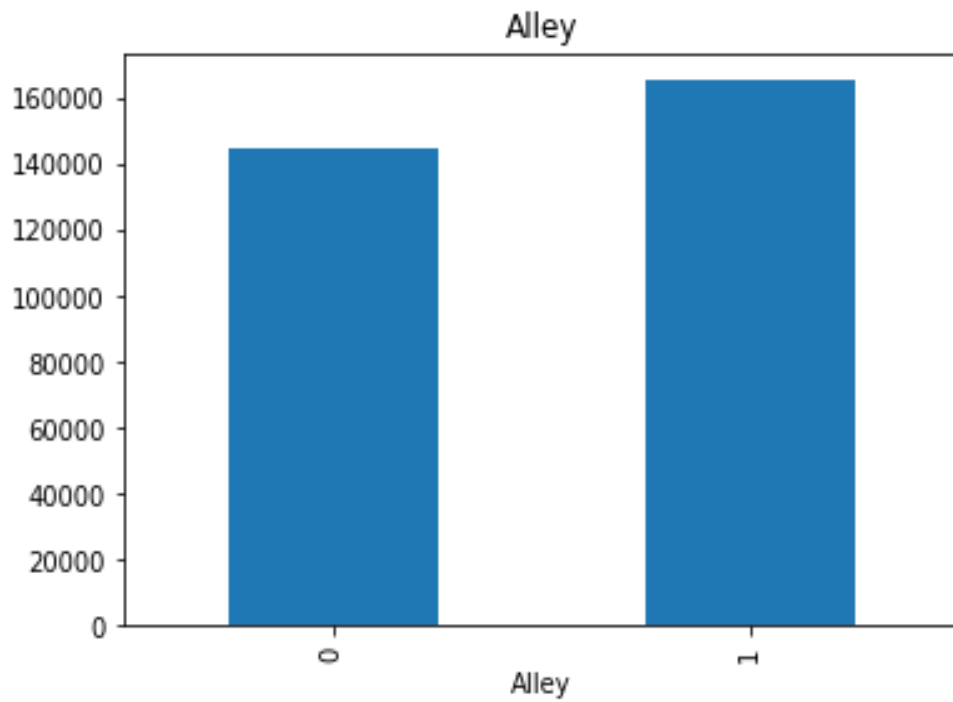
Missing values percentage can be checked by the below python code:-

```
for feature in missing_values:
    print(feature, np.round(df[feature].isnull().mean()*100,4), "% Missing Values")
```

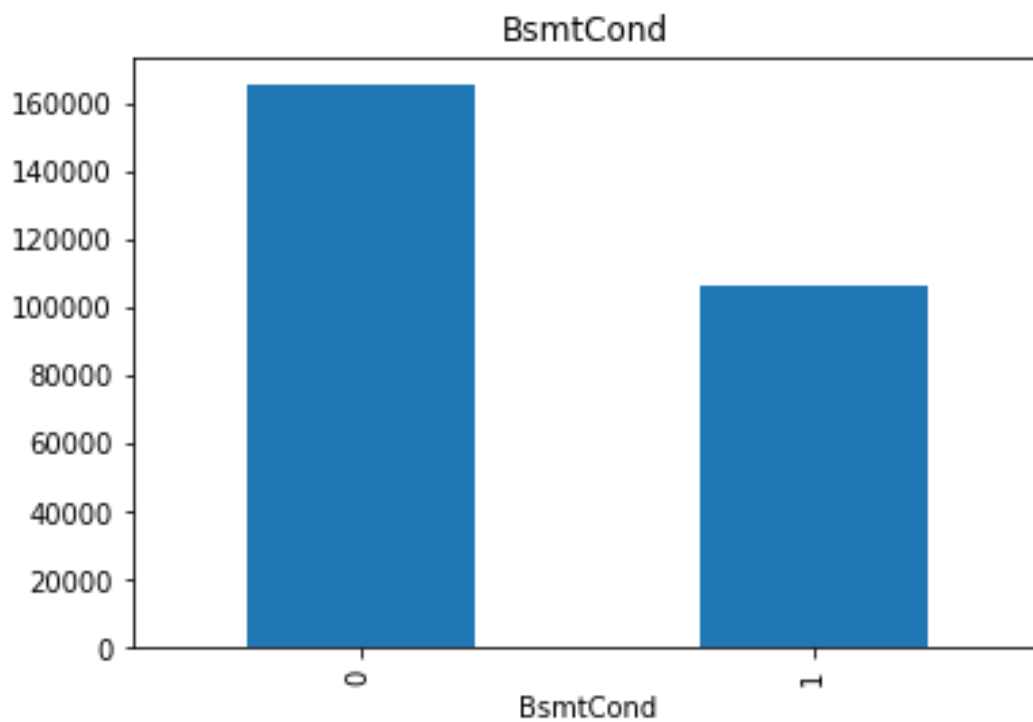
Observation:-

1. There are many missing values in the columns of the dataset
2. Hence need to check the relationship with sales price

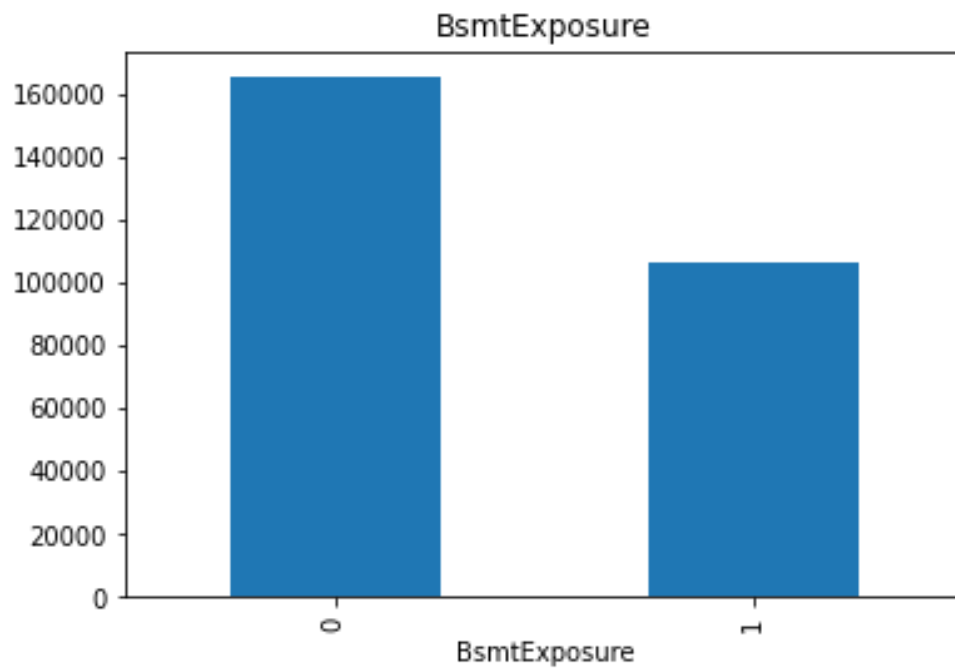
## Representation of Missing values vs Salesprice



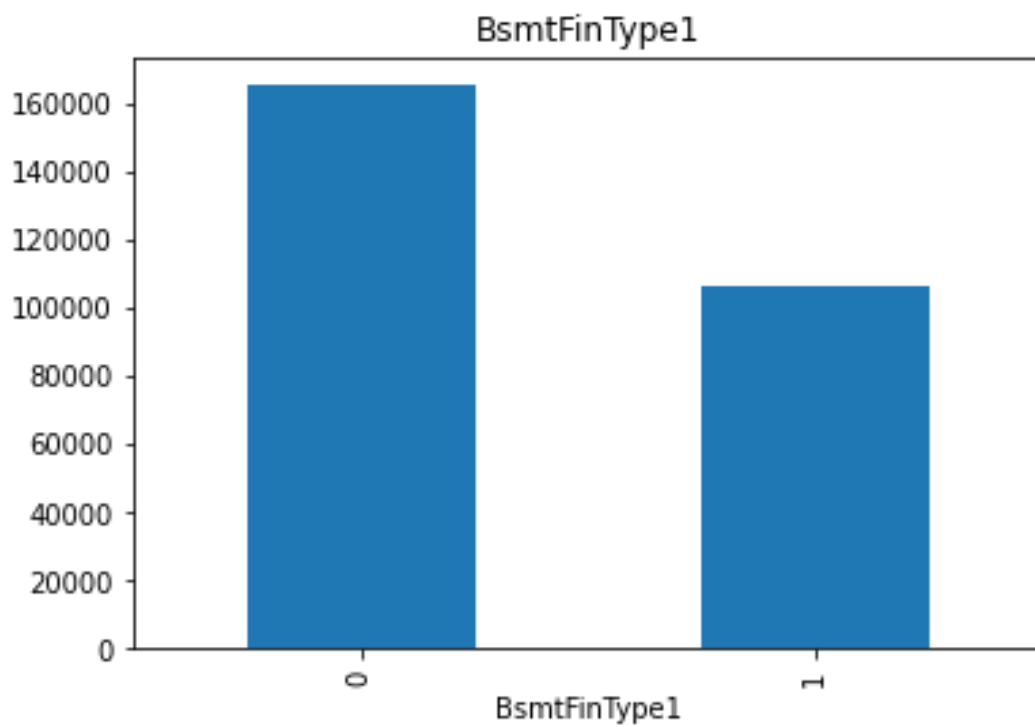
Missing value vs alley



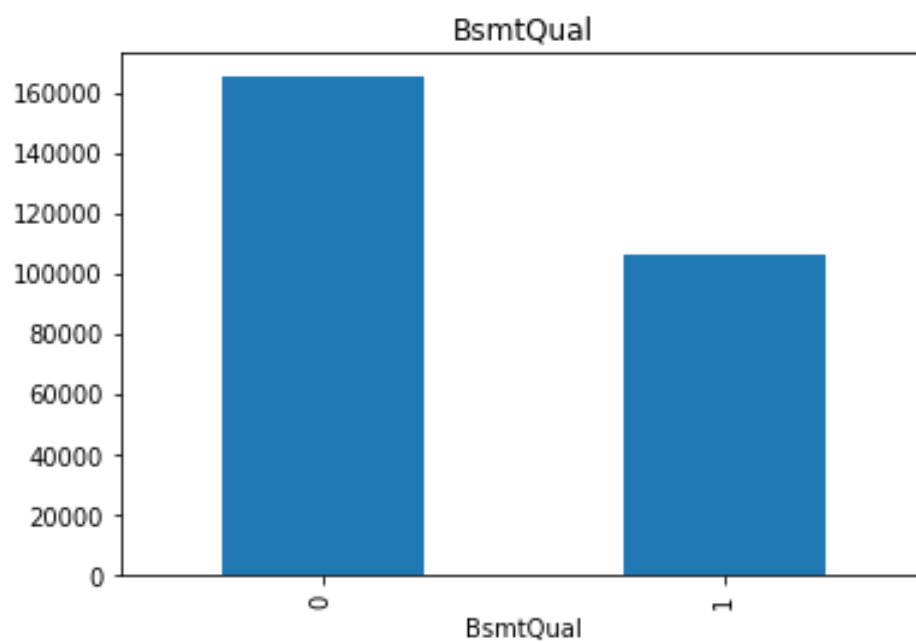
Missing value vs Bsmtcond



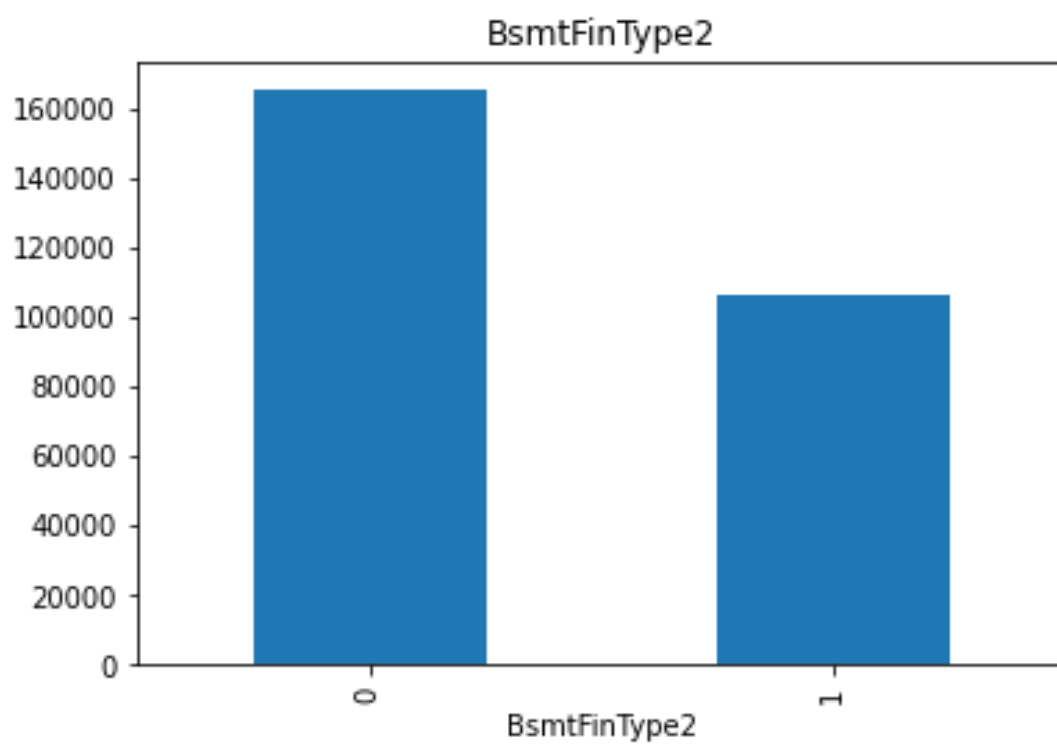
Missing value vs bsmt Exposure



Missing value vs BsmtFintype 1

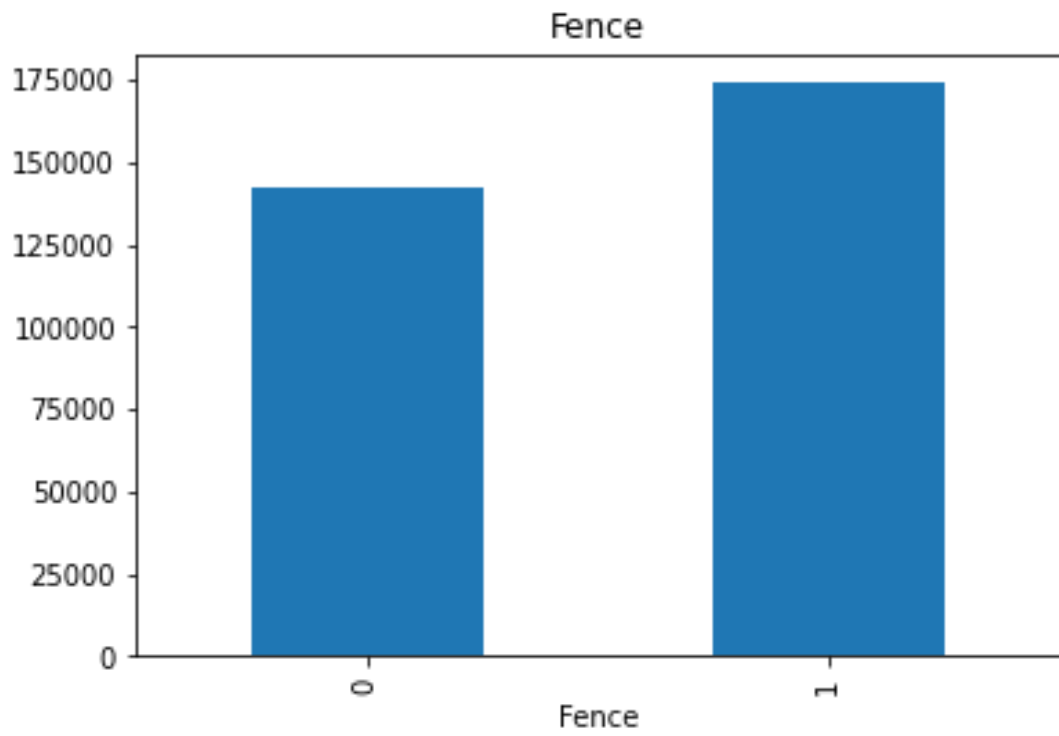


Missing value vs Bsmtqual

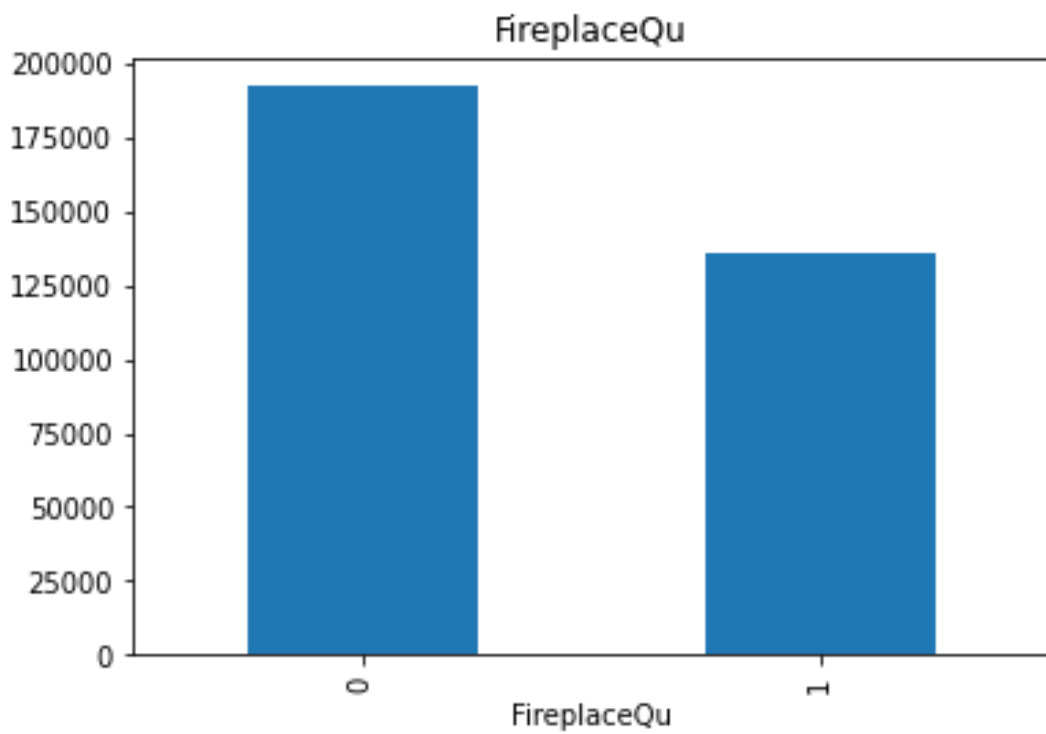


Missing value vs Bsmttype2

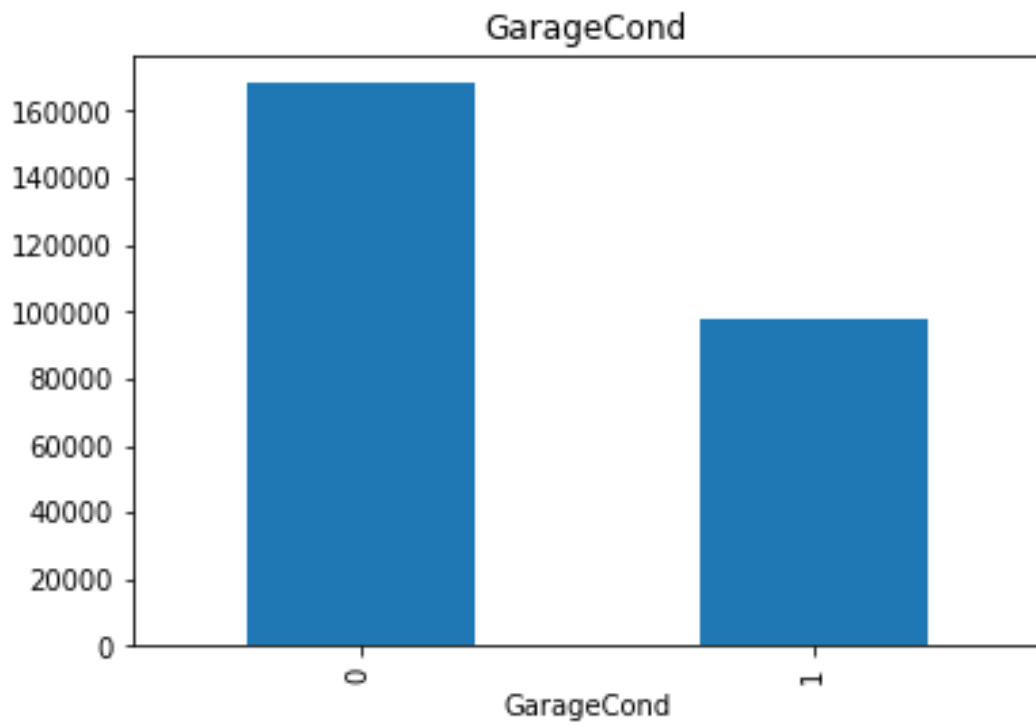




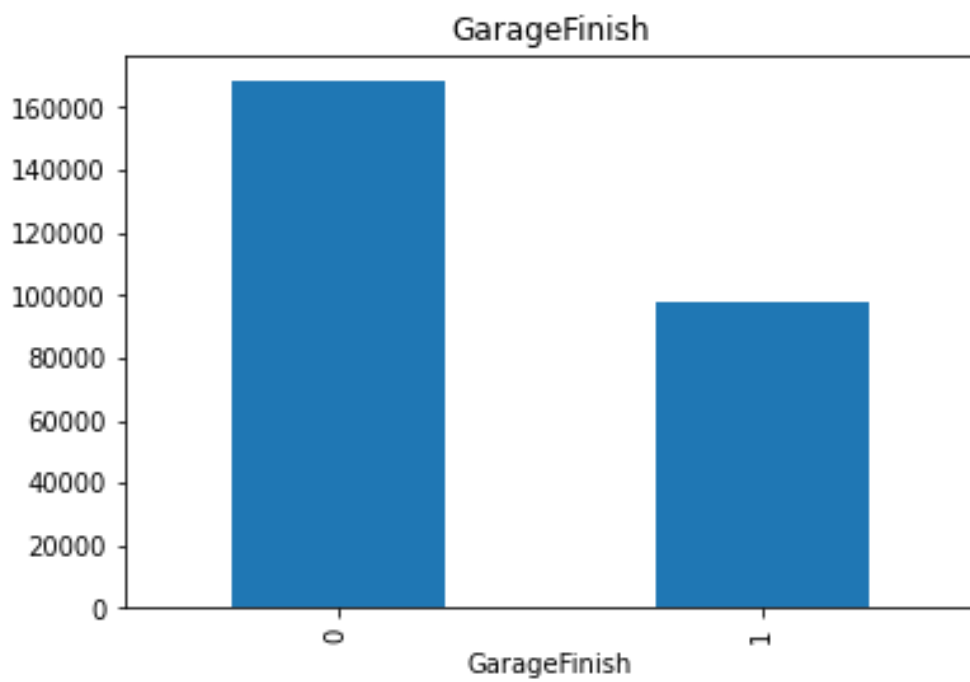
Missing value vs Fence



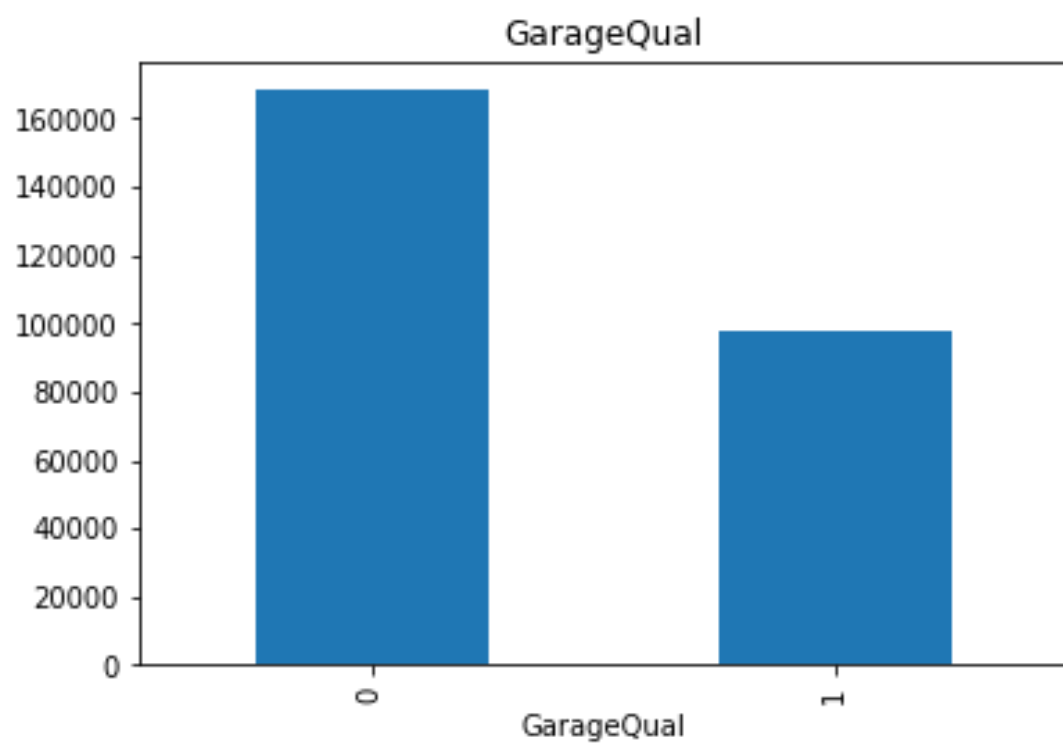
Missing value vs FireplaceQu



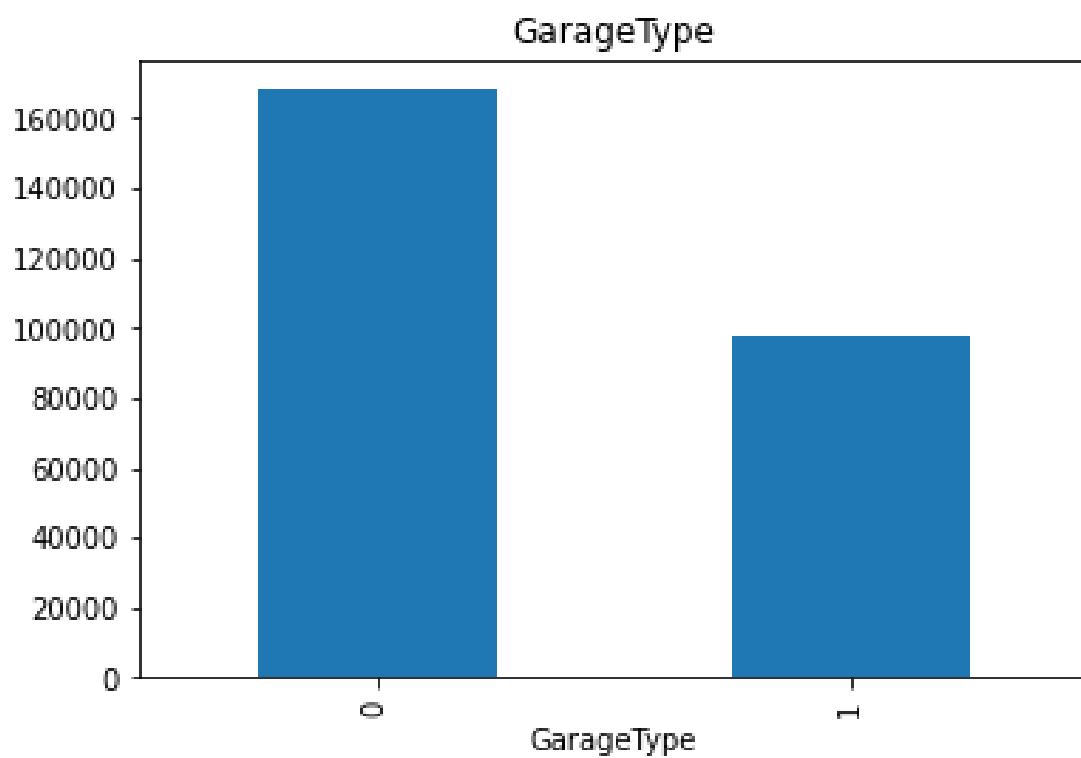
Missing value vs GarageCond



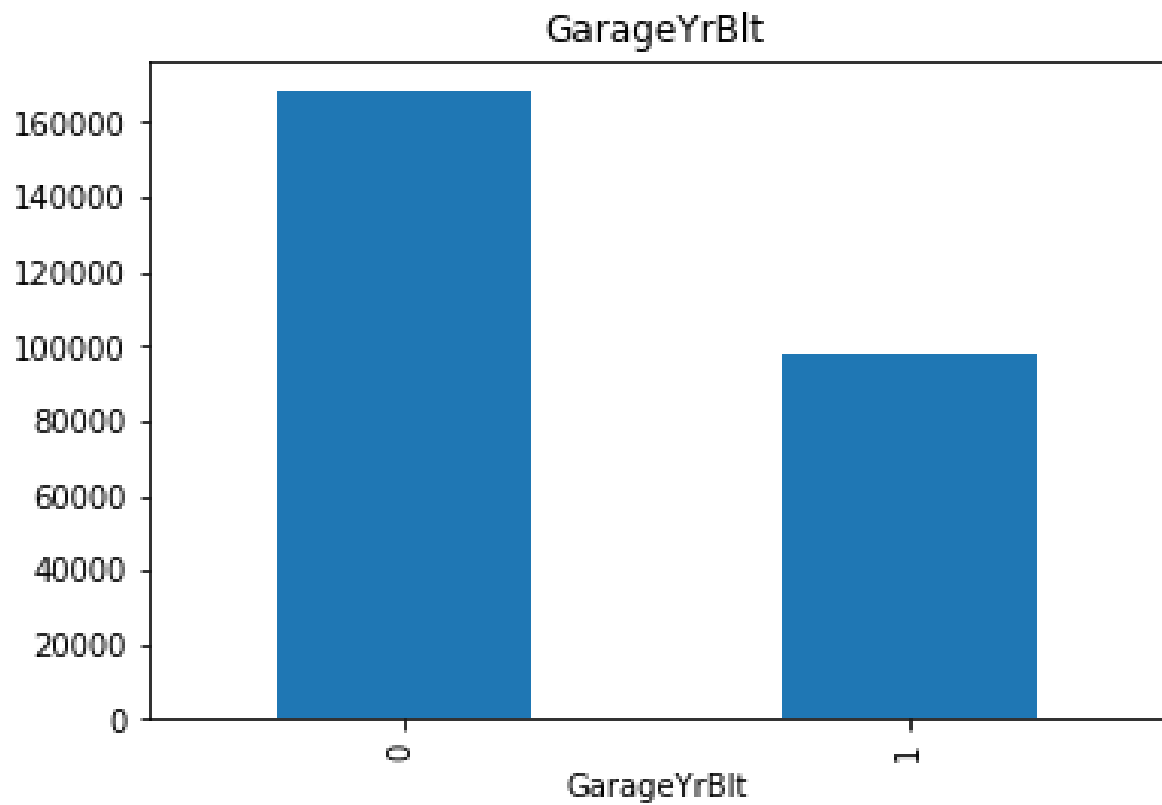
Missing value vs GarageFinish



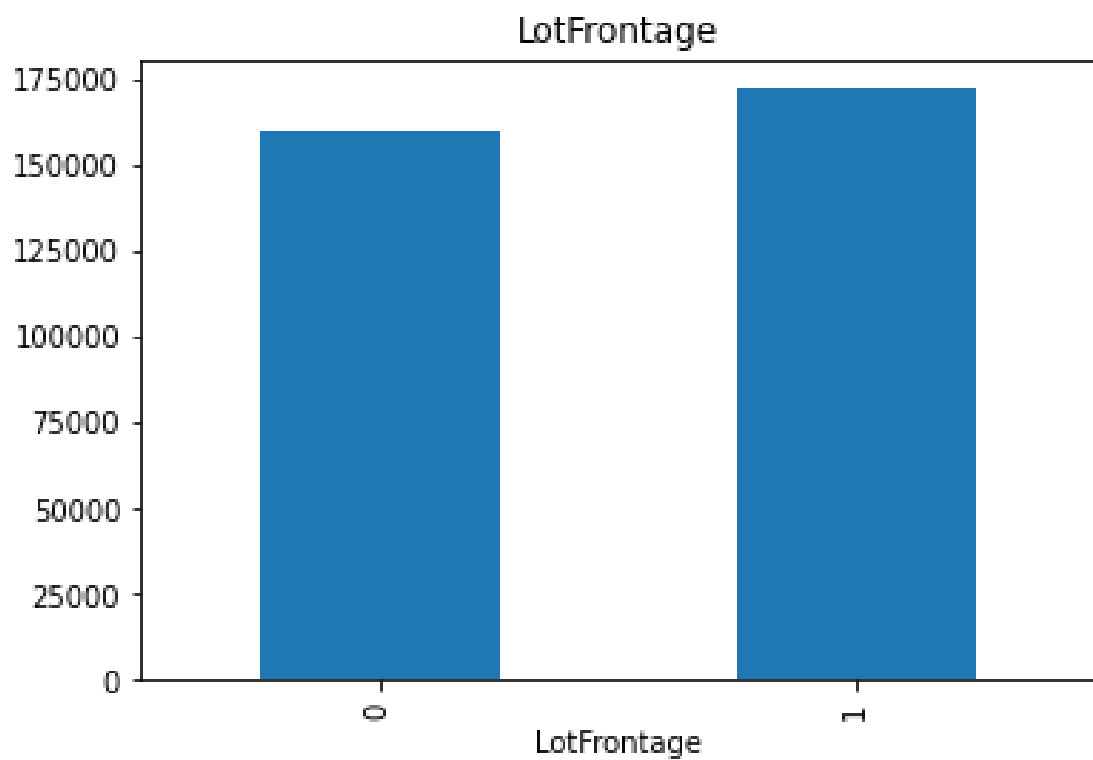
Missing value vs GarageQual



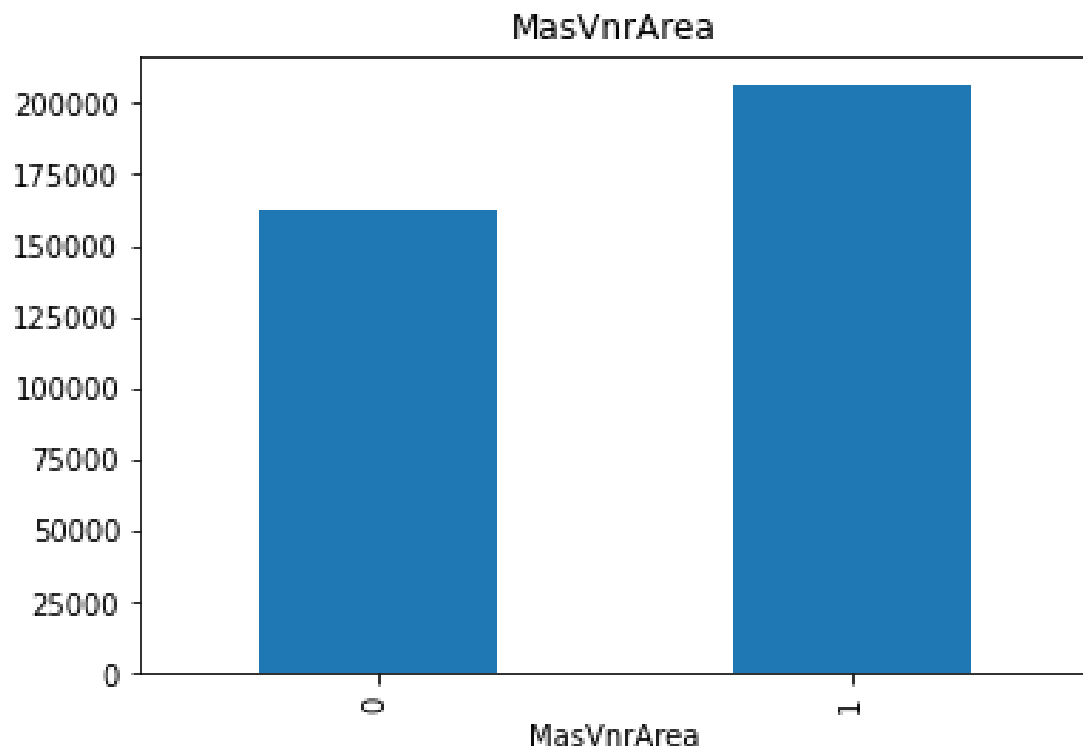
Missing value vs GarageType



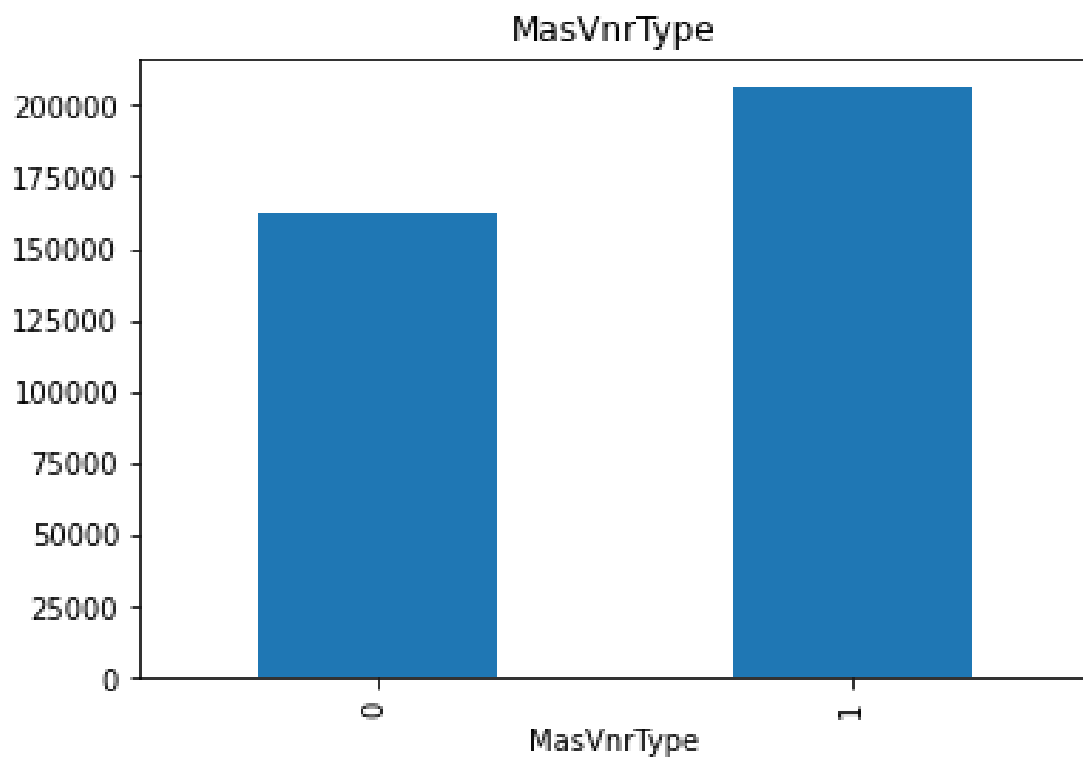
Missing value vs GarageYrBuilt



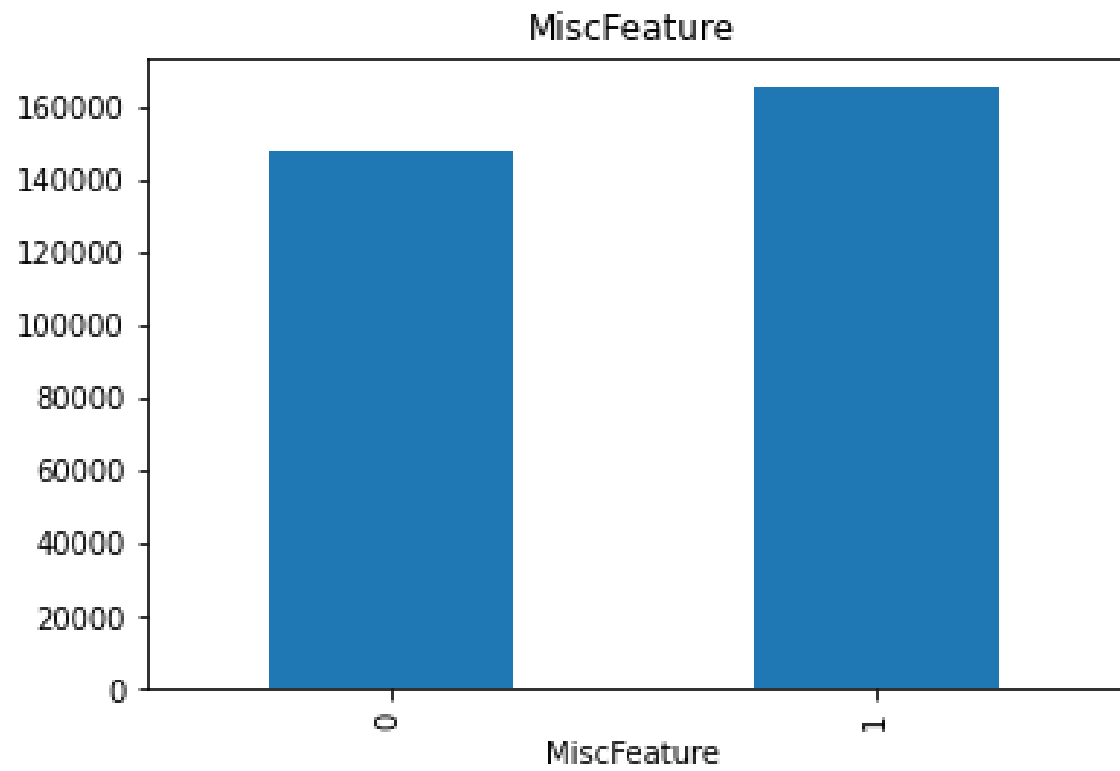
Missing value vs Lotfrontage



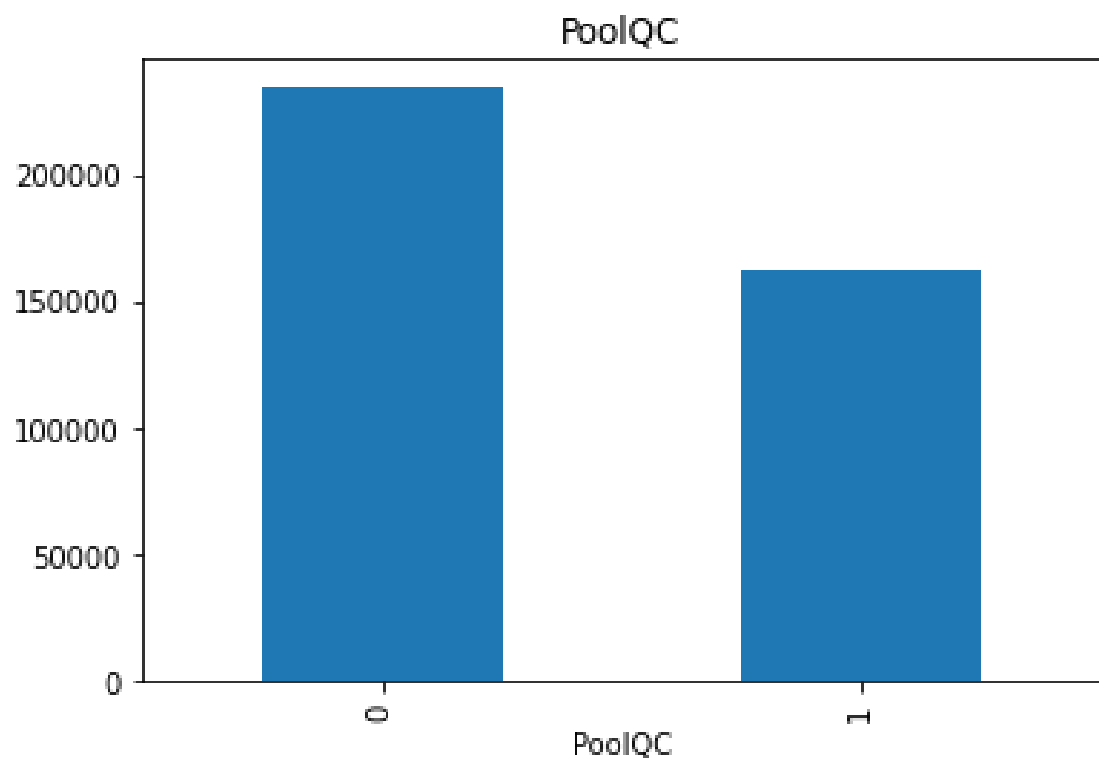
Missing value vs MassVnArea



Missing value vs MassVnrType



Missing value vs MiscFeature



Missing value vs PoolQC

### 3.Extracting all the numerical feature

Extracting all the numerical values using python code:-

```
numerical_features=[x for x in df.columns if df[x].dtypes != "O"]  
print("The number of the numerical columns in the dataset:",  
len(numerical_features))  
print("Numerical columns in the dataset:\n", numerical_features)  
print("-"*125)  
df[numerical_features].head()
```

Observation:-

1. 'YearBuilt','YearRemodAdd','GarageYrBlt','YrSold' are date columns we have in this dataset.
2. From the datetime column we usually extract the no of days, years, hours, minutes etc. hence this can be derived from the columns.
4. Extract the year column from the dataset:

Extract the year column from the dataset using the python code:-

```
year_feature=[x for x in df.columns if 'Yr' in x or 'Year' in x]  
print("The number of Year column in the dataset :",len(year_feature))  
print("Year columns in the dataset :\n",year_feature)  
print("-"*125)  
df[year_feature].head()
```

### 5.Checking the unique items in date time columns

Checking the unique items in datetime columns using the python code:-

# checking the unique items in the datetime columns

for feature in year\_feature:

```
print("The unique items in the columnn", feature, ":\n", df[feature].unique())
```

## 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))
```

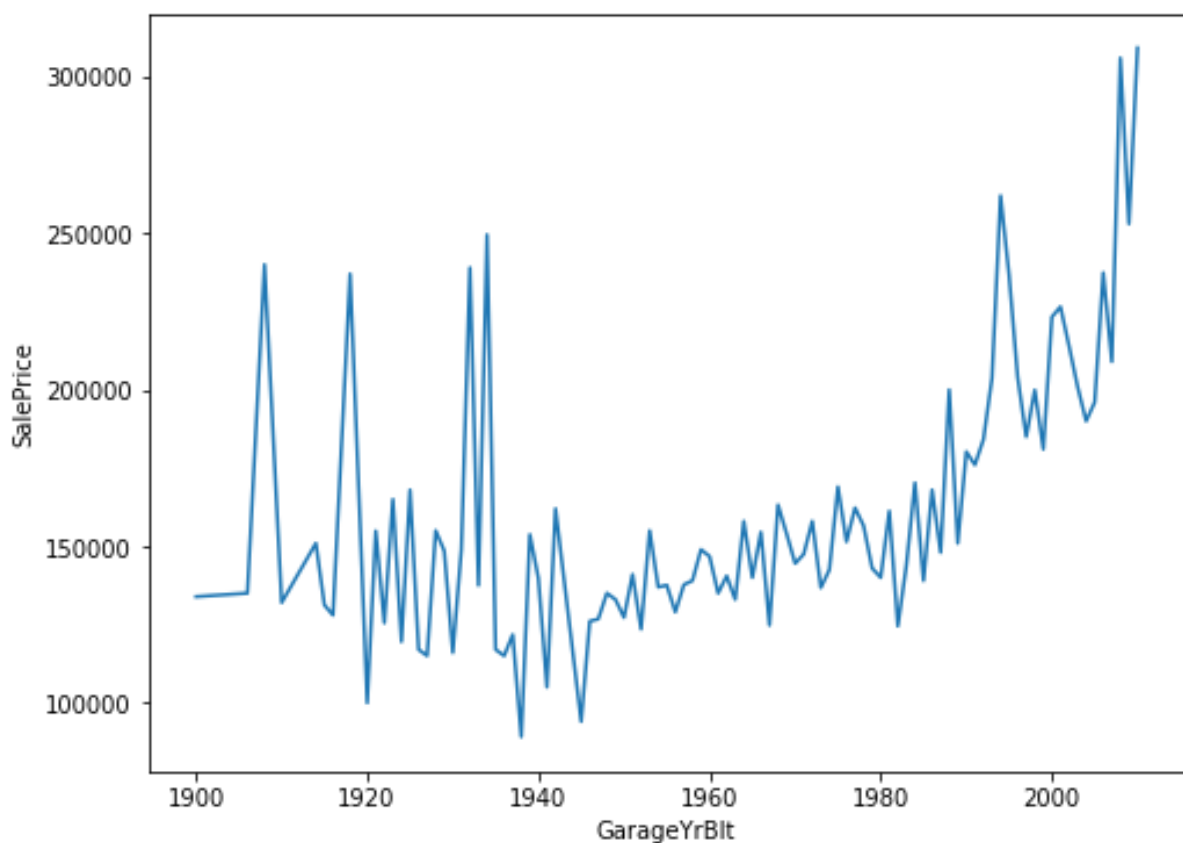
```
df.groupby(feature)['SalePrice'].median().plot()
```

```
plt.xlabel(feature)
```

```
plt.ylabel('SalePrice')
```

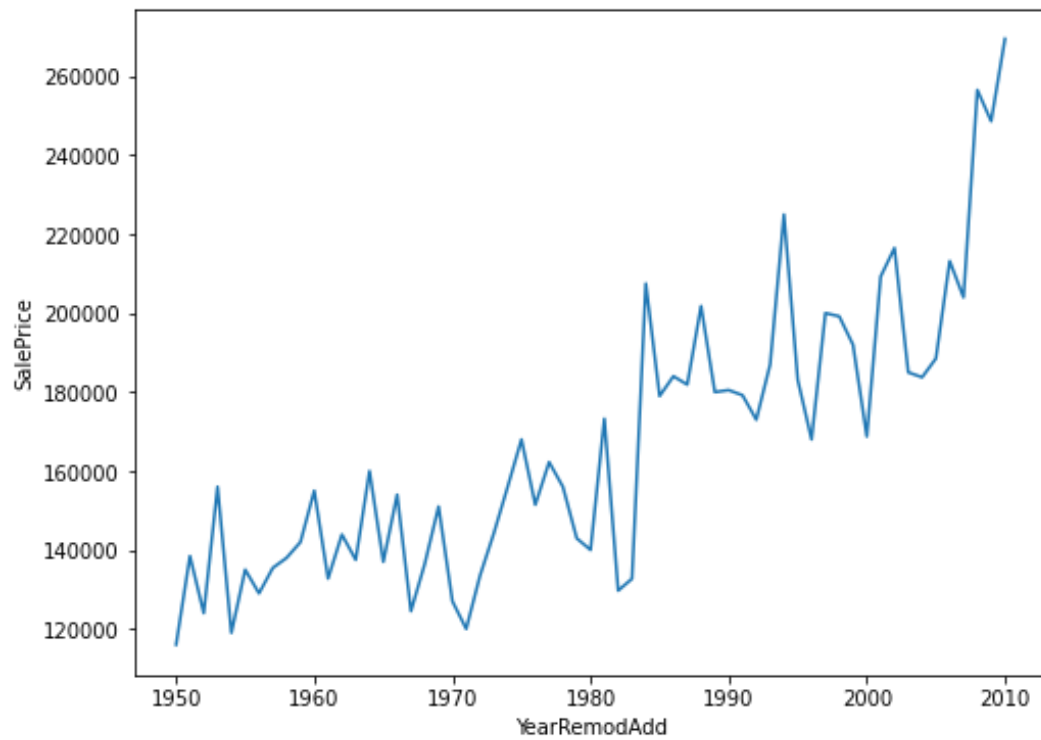
```
plt.show()
```

Data Visualization:-

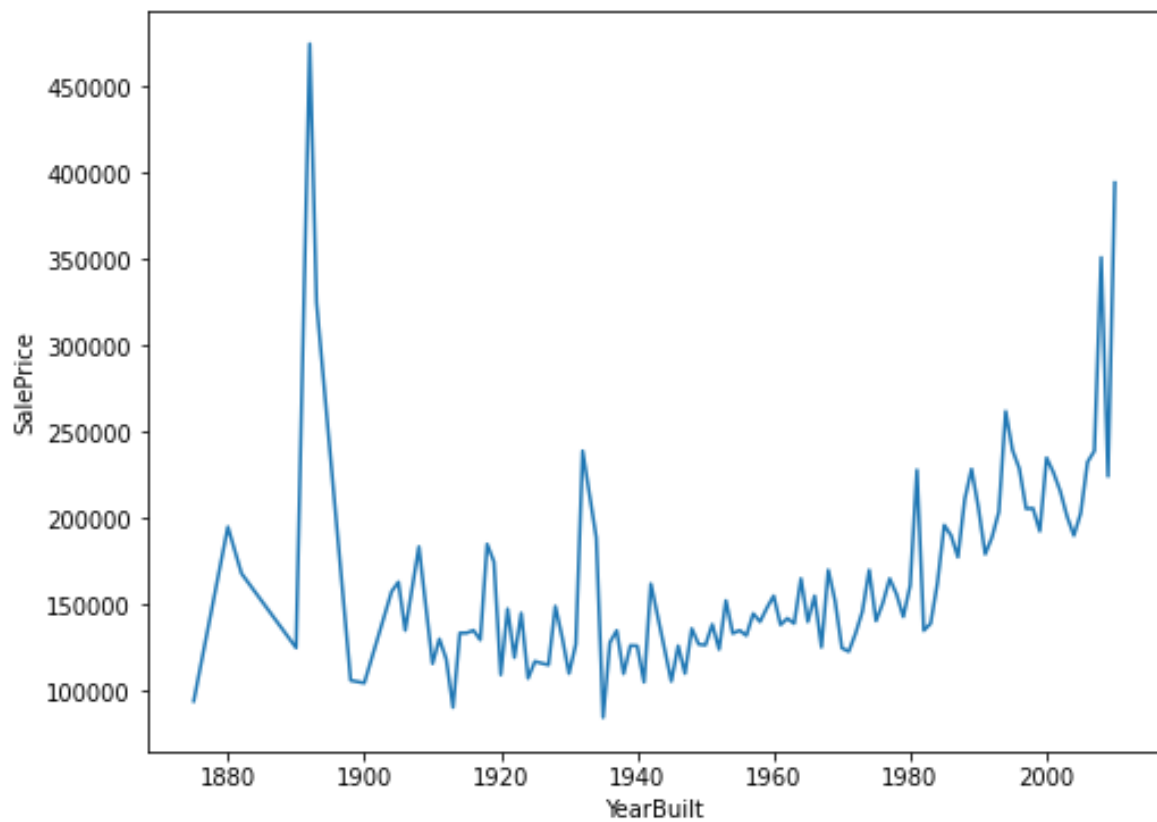


SalePrice vs GarageBelt

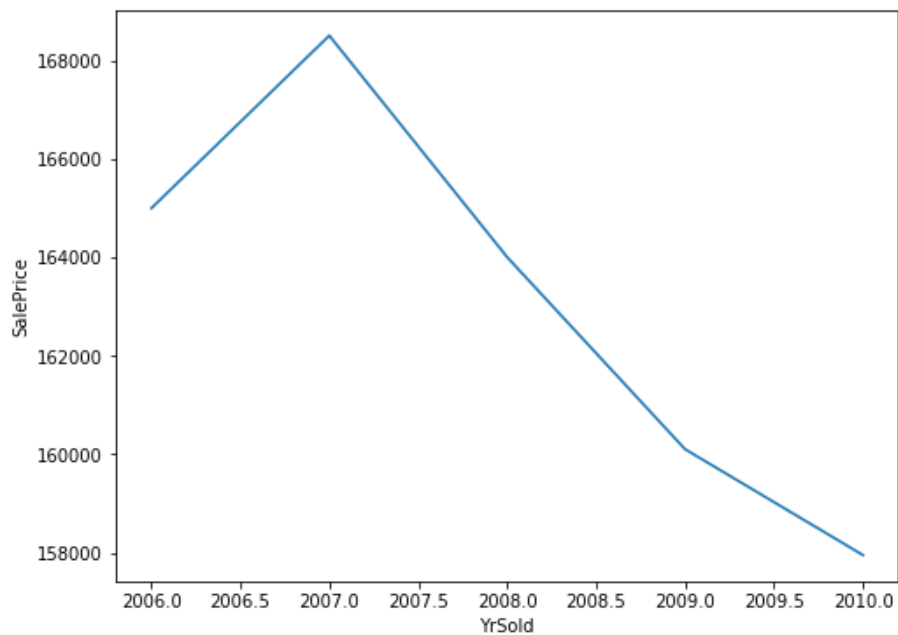




YearRemodAdd vs SalePrice



SalePrice vs YearBuilt



YrSold vs SalePrice

Extracting the discrete and continuous variable using the python code:-

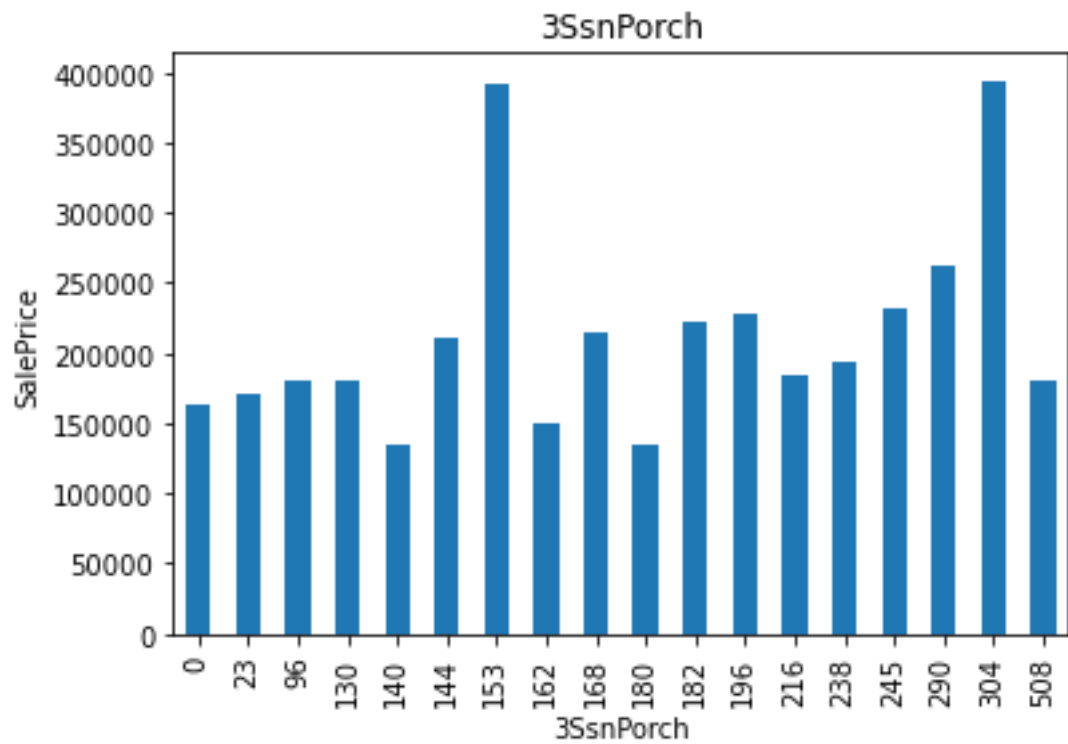
```
discrete_feature=[x for x in numerical_features if len(df[x].unique())<25 and x
not in year_feature+['Id']]

print("The number of discrete column in the dataset:", len(discrete_feature))

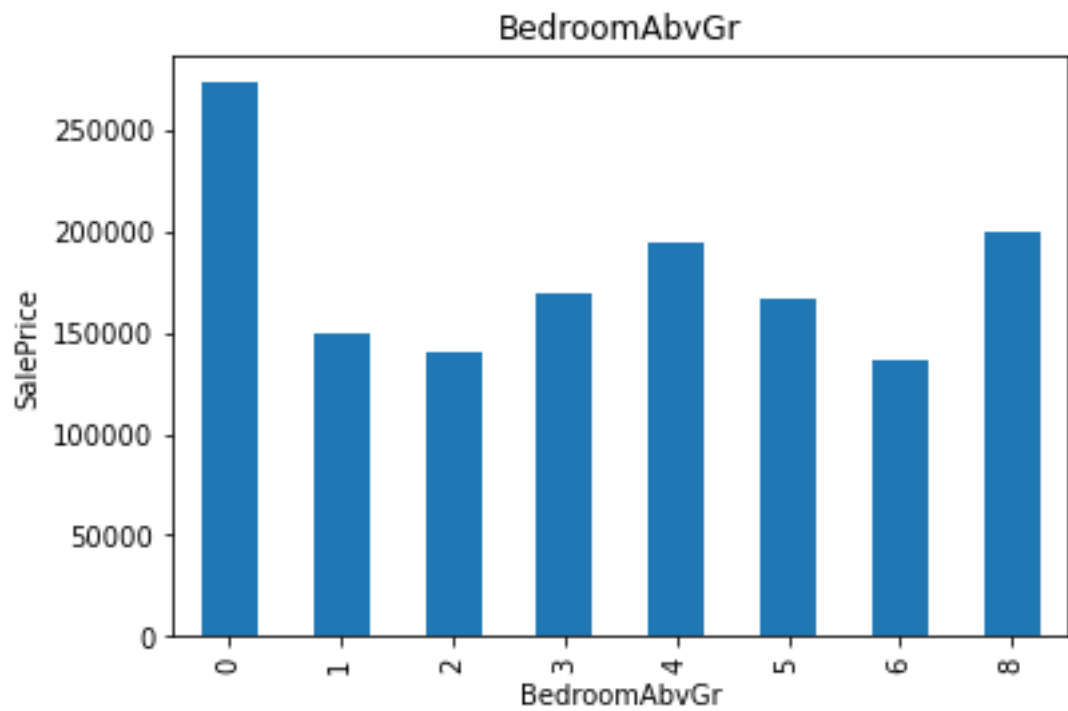
print("Discrete columns in the dataset: \n", discrete_feature)

print("-"*125)

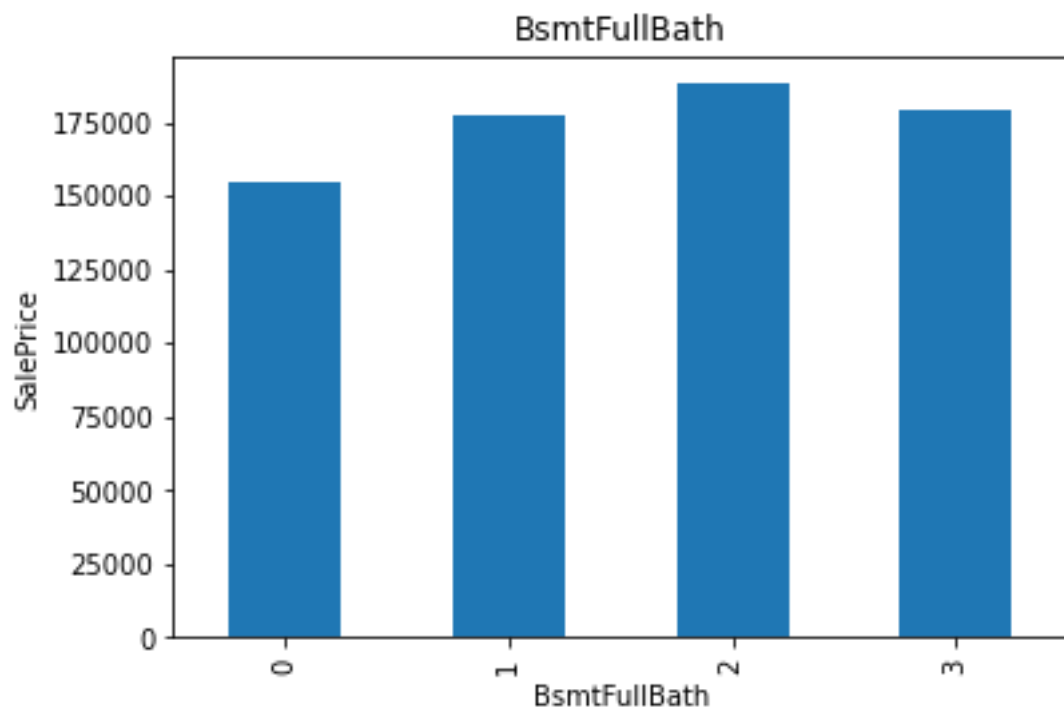
df[discrete_feature].head()
```



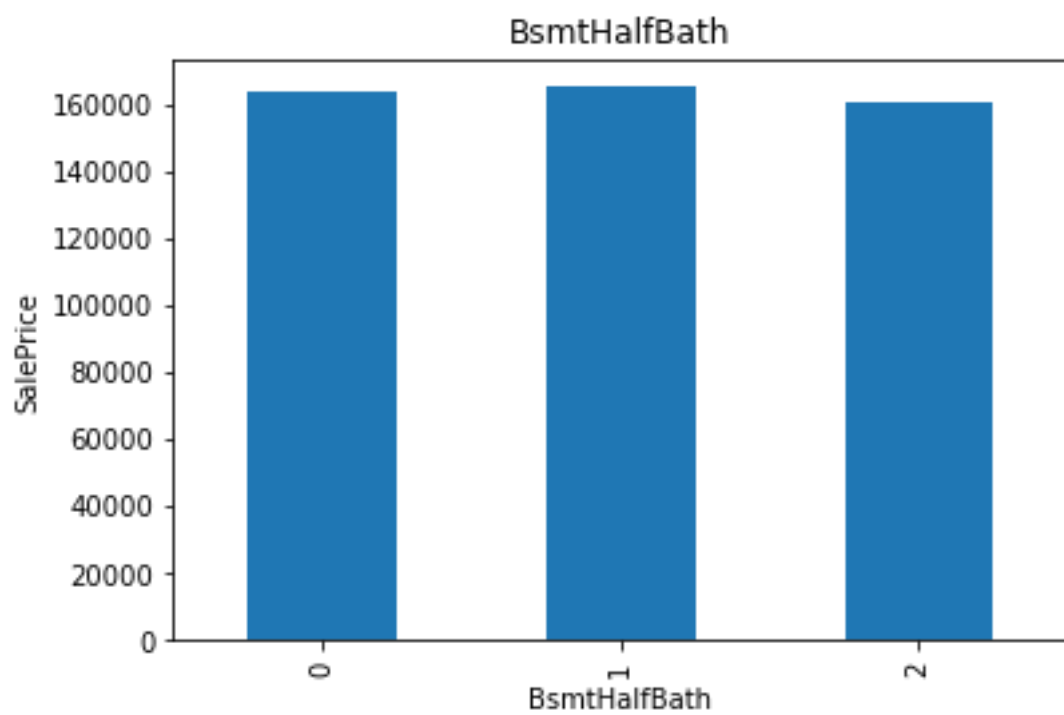
SalePrice vs 3SsnPorch



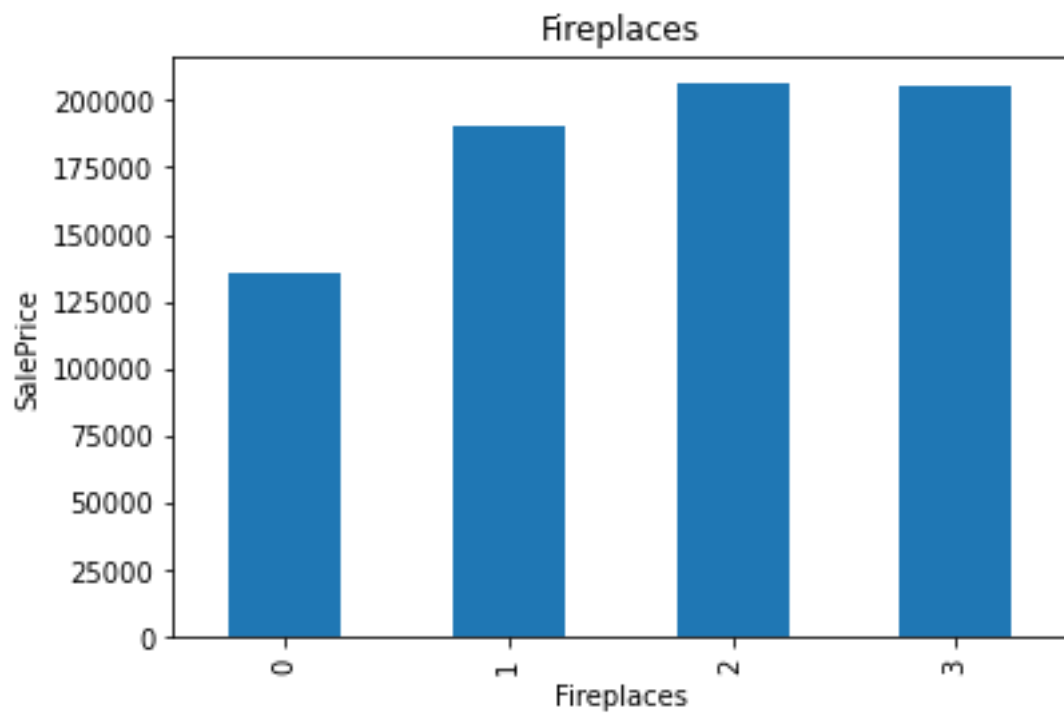
SalePrice vs BedroomAbvGr



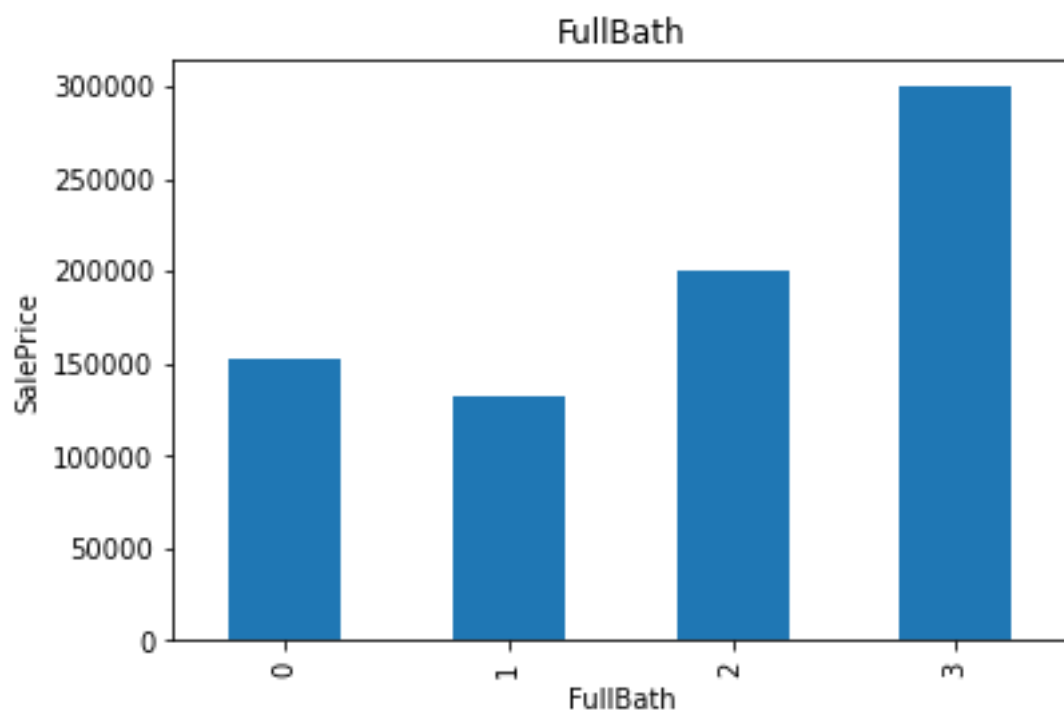
Saleprice vs bsmtFullBath



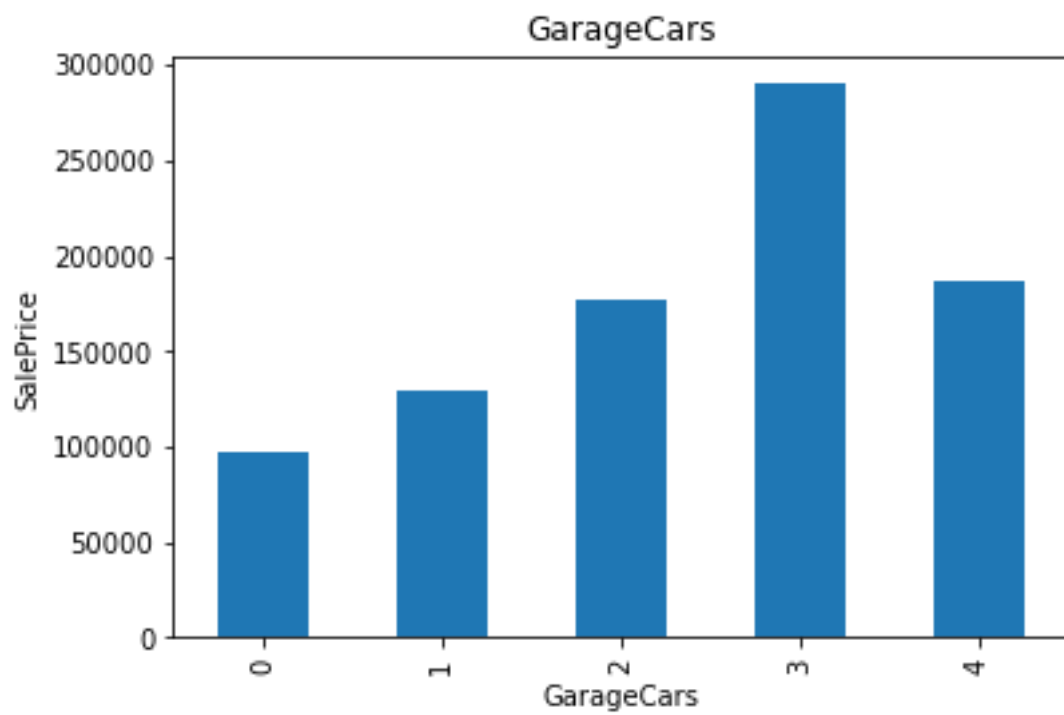
SalePrice vs BsmtHalf Bath



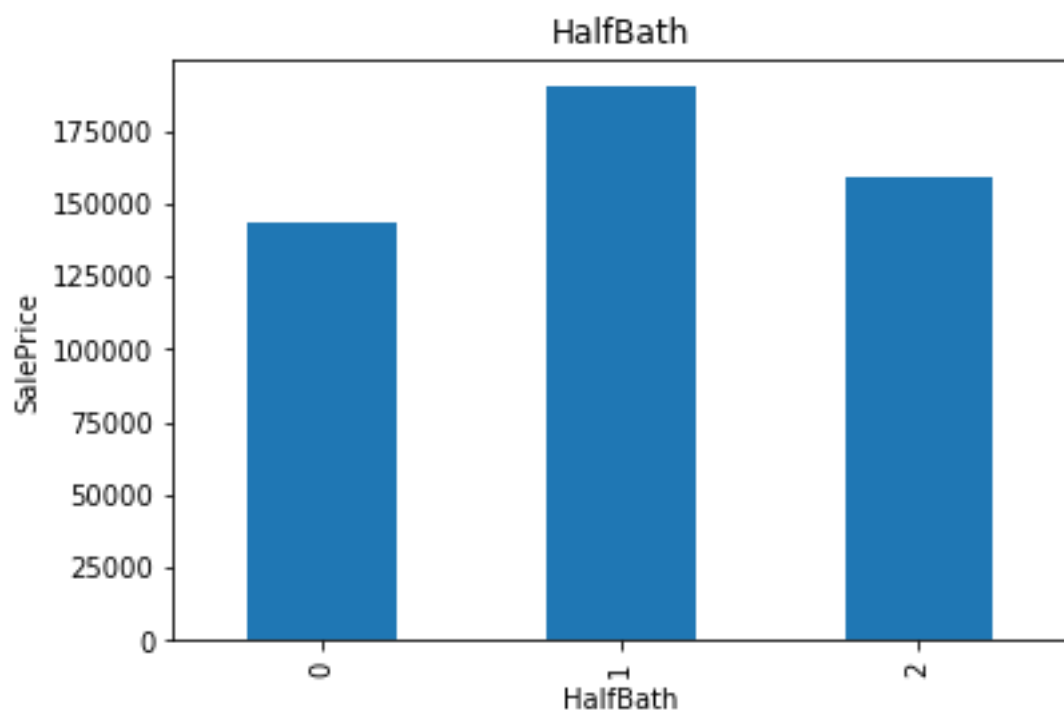
Saleprice vs Fireplaces



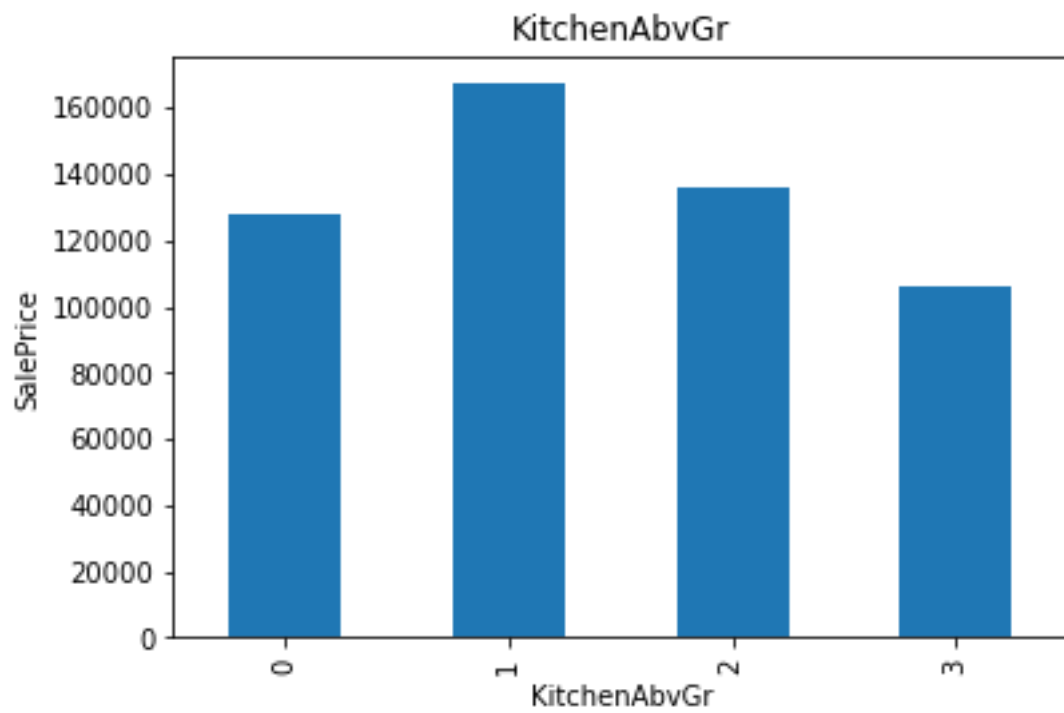
Saleprice vs FullBath



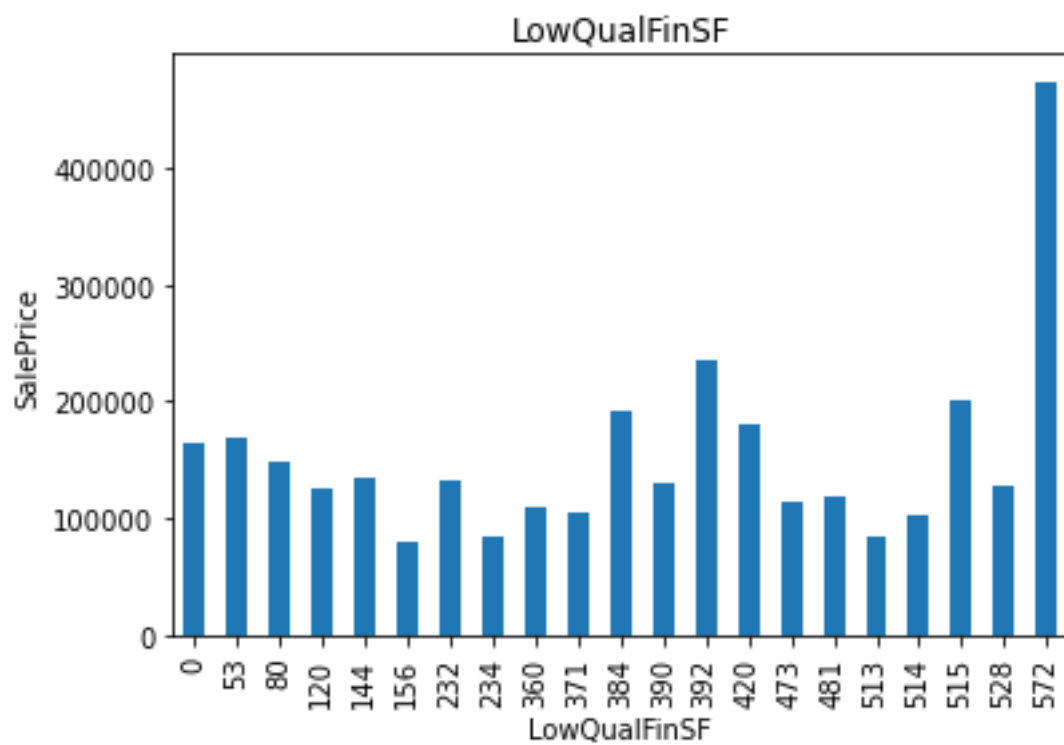
Salesprice vs GarageCars



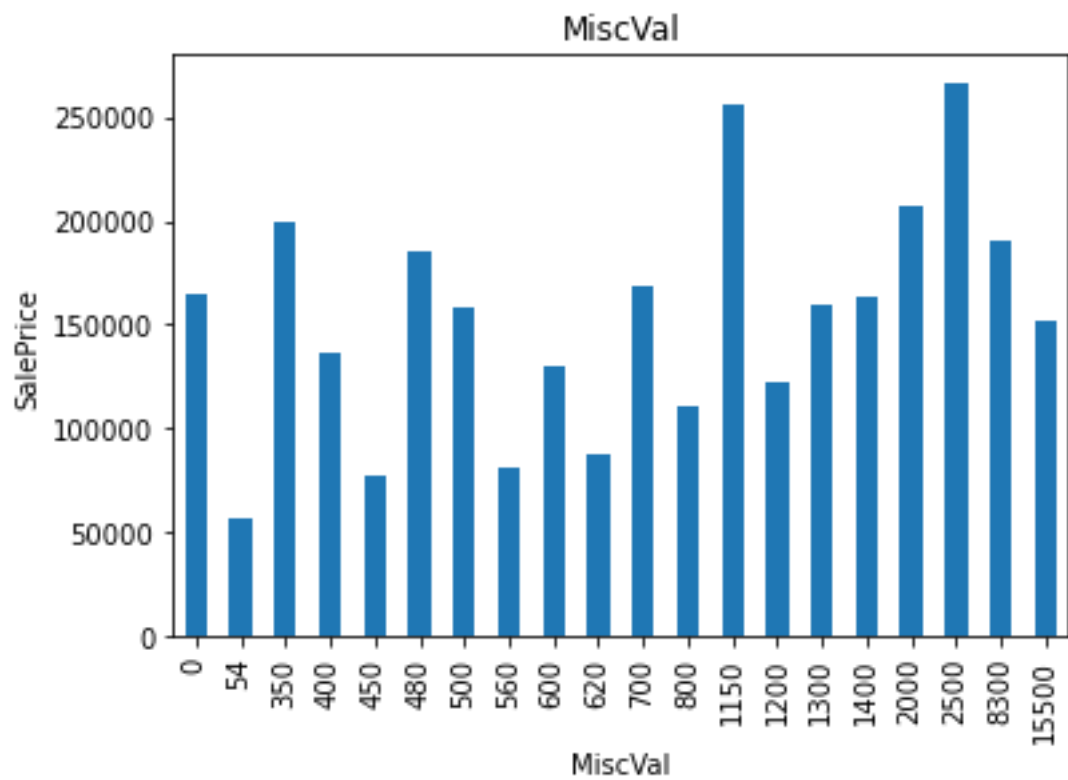
Salesprice vs HalfBath



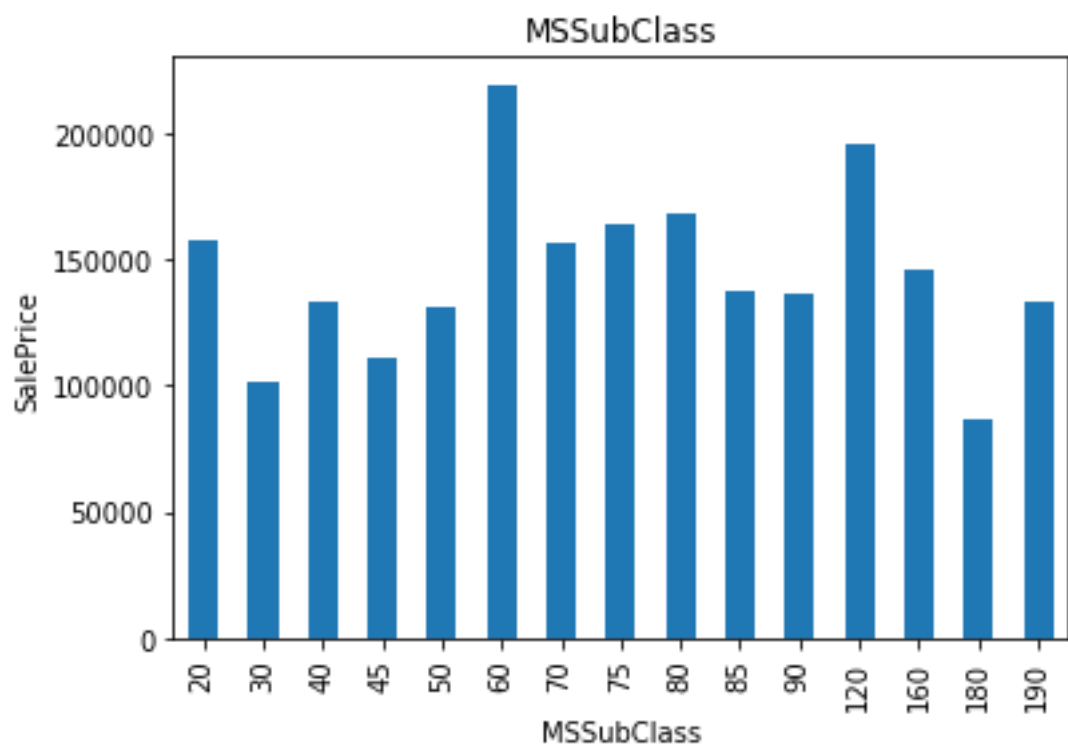
Salesprice vs KitchenAbvGr



Saleprice Vs LowQualinSf

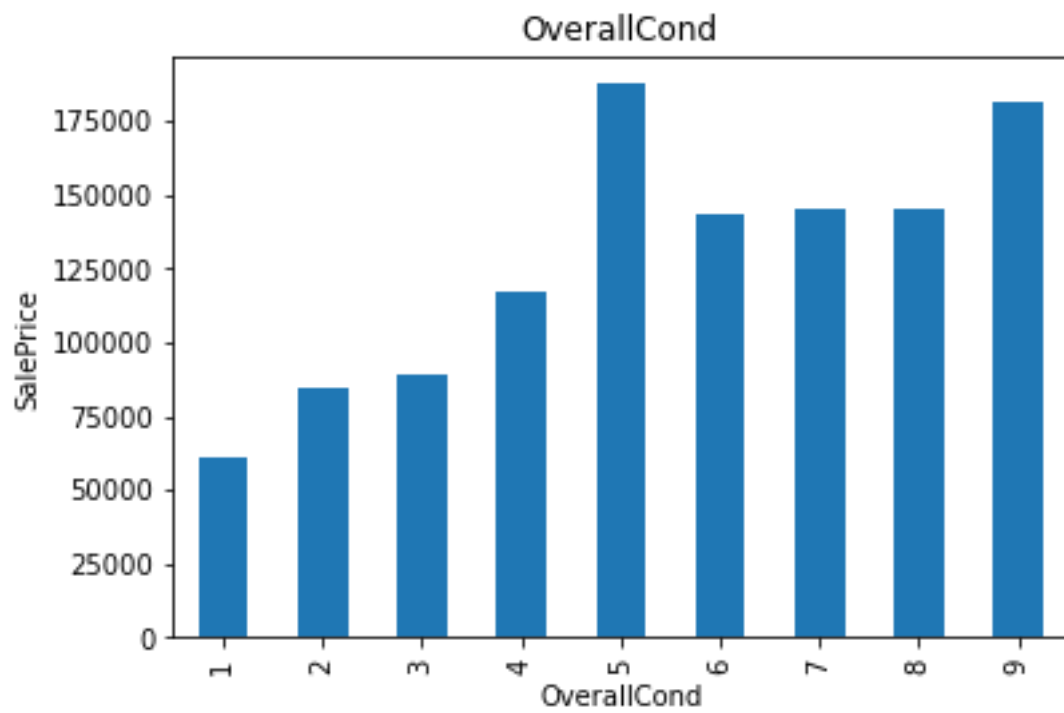


Saleprice vs MiscVal

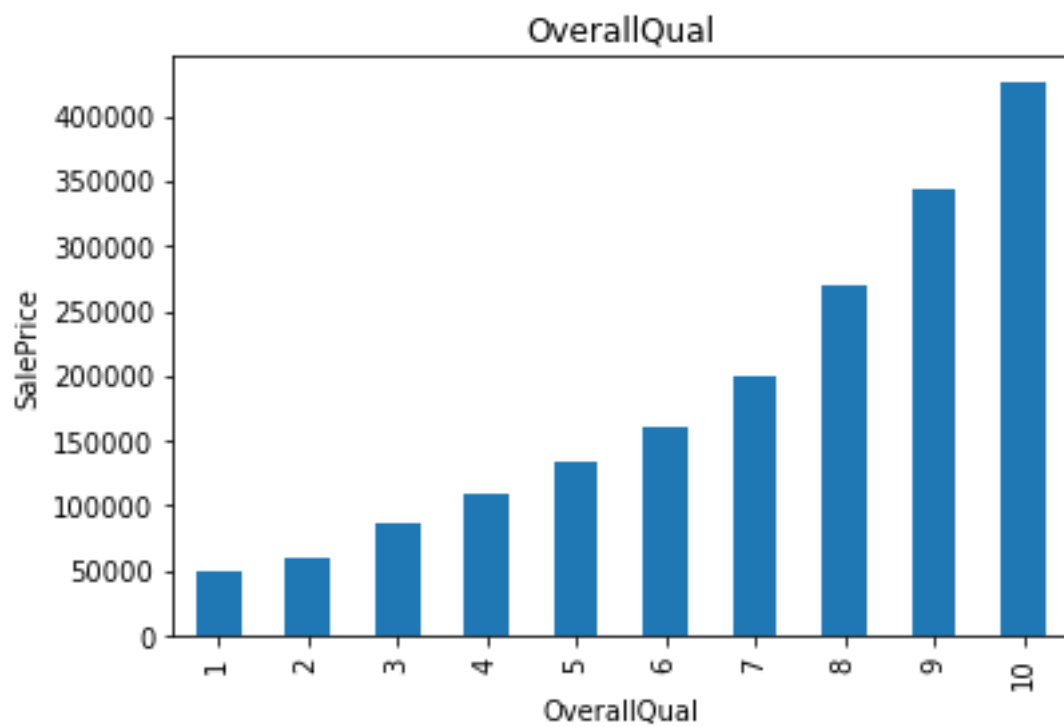


Saleprice vs MSSubclass

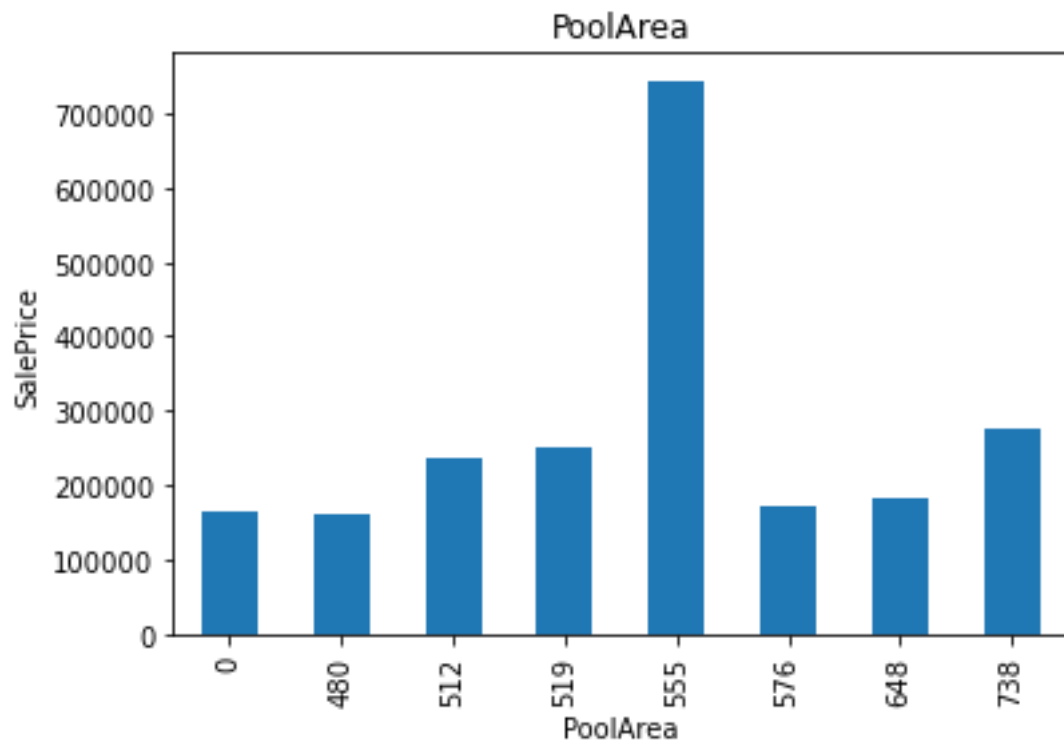




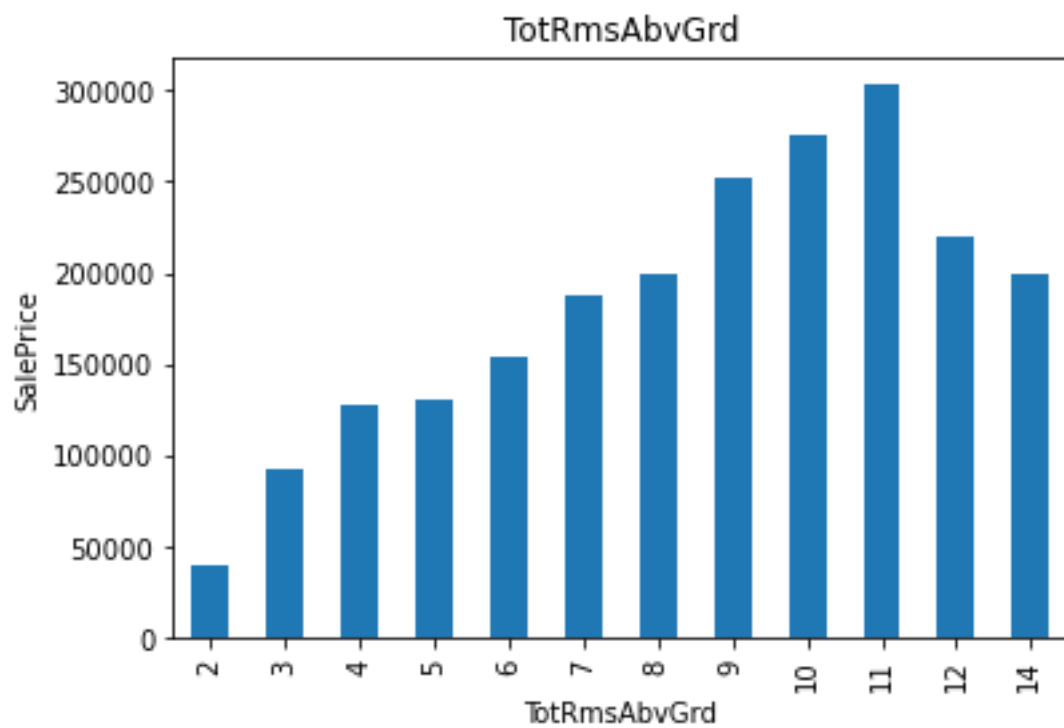
Saleprice vs overallcond



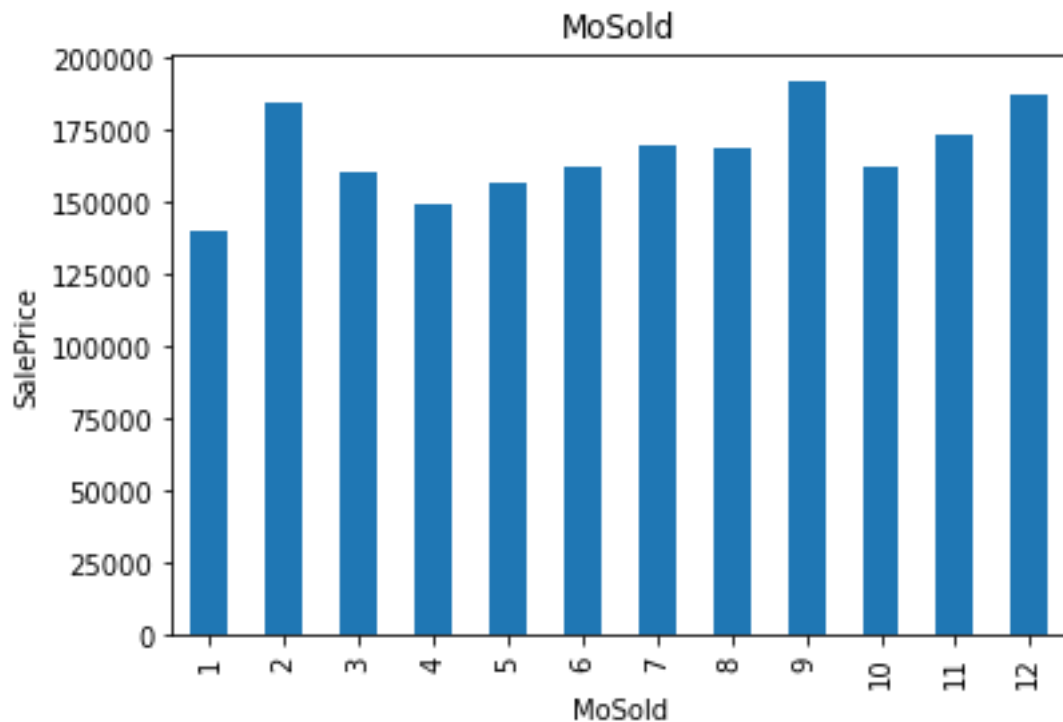
Saleprice vs OverallQaul



SalePrice vs PoolArea



SalePrice Vs TotRmsAbvGrd



Saleprice vs MOsold

Extracting the continuous variable

```
continuous_feature=[x for x in numerical_features if x not in
discrete_feature+year_feature+['Id']]
```

```
print("The number of continous feature column in the dataset
:",len(continuous_feature))
```

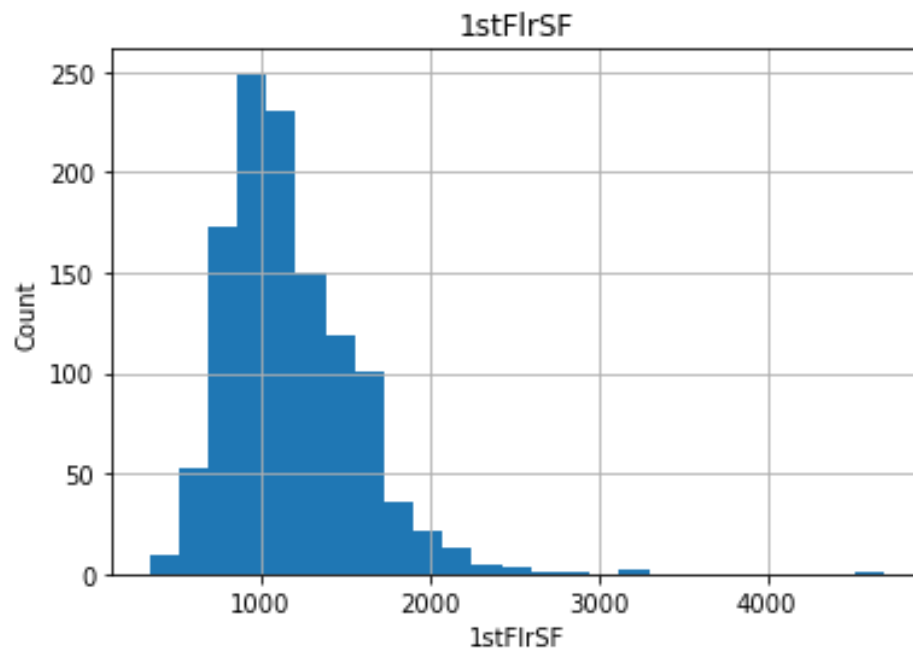
```
print("Continous feature columns in the dataset :\n",continuous_feature)
```

```
print("-"*125)
```

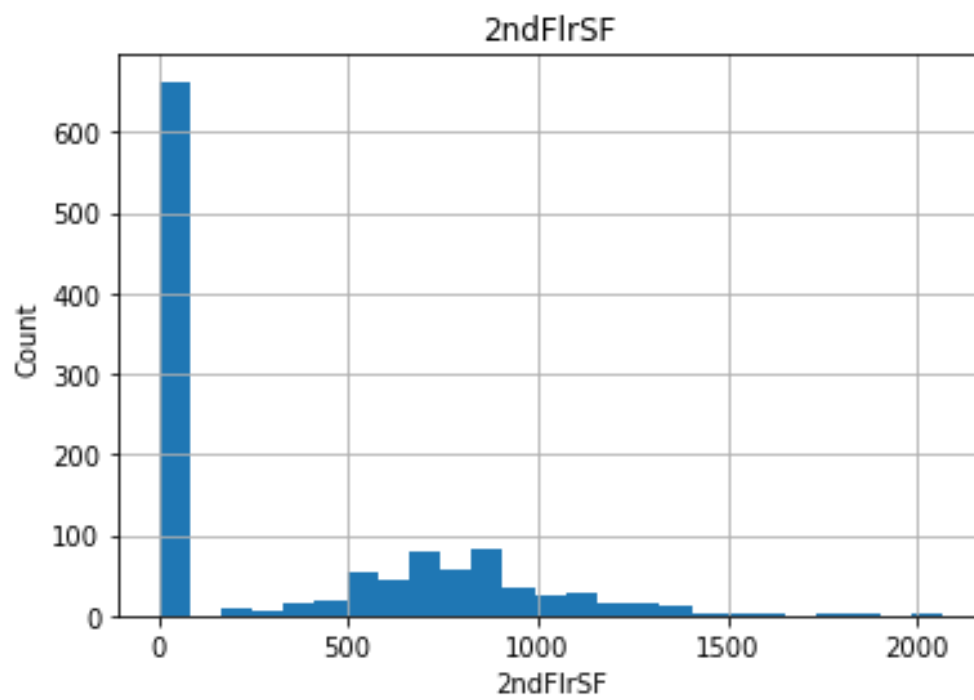
```
df[continuous_feature].head()
```

observation:-

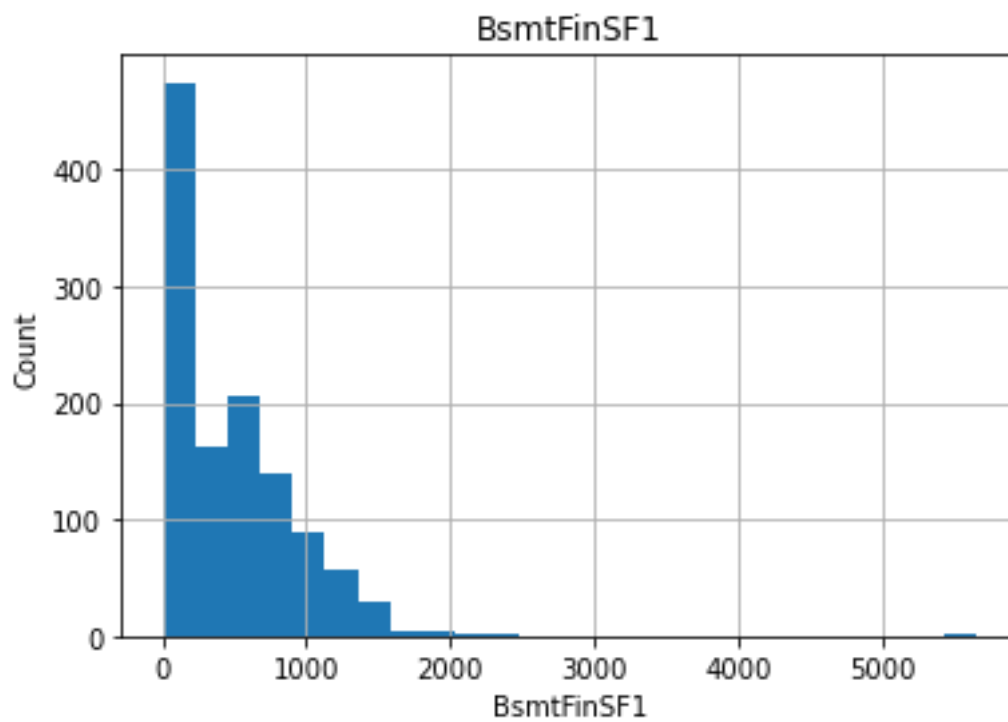
The number of continuous feature column in the dataset : 16



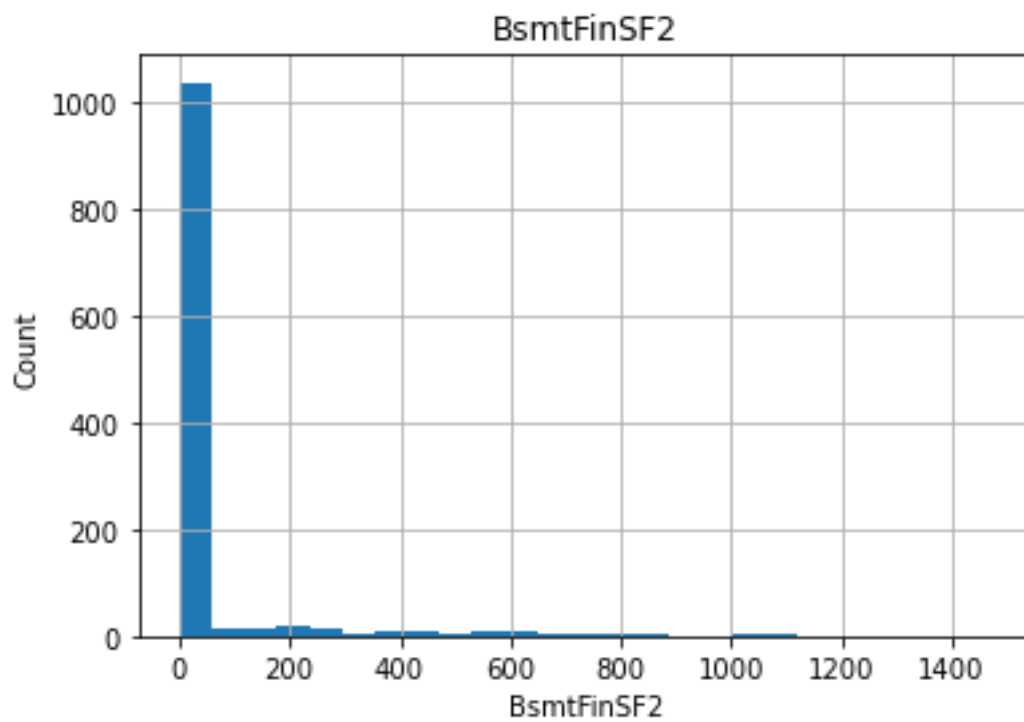
Histogram plot of 1stFlr



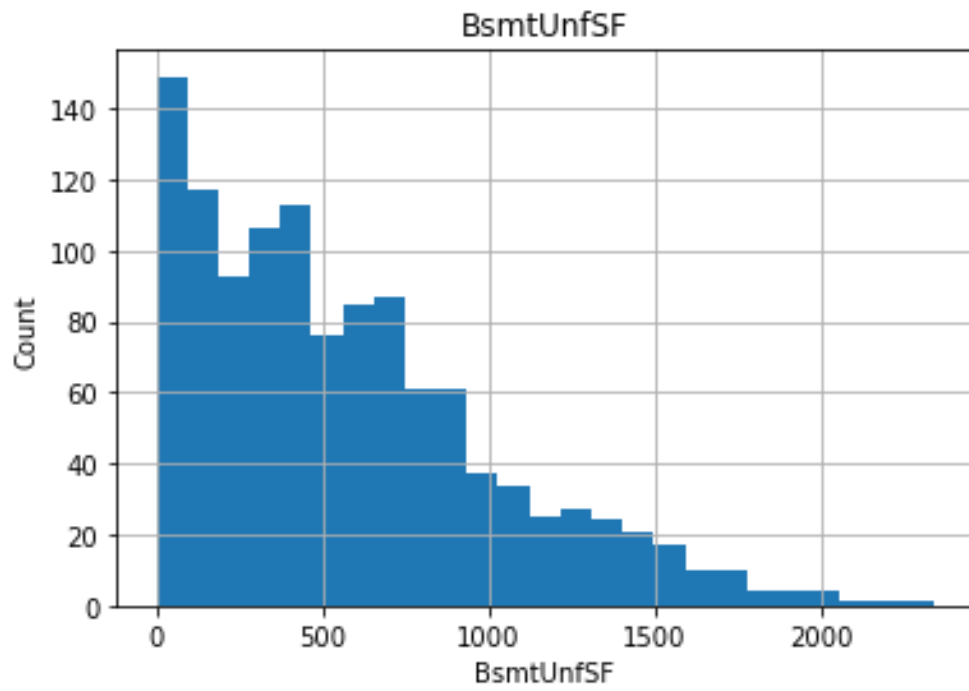
Histogram of 2ndFlrSF



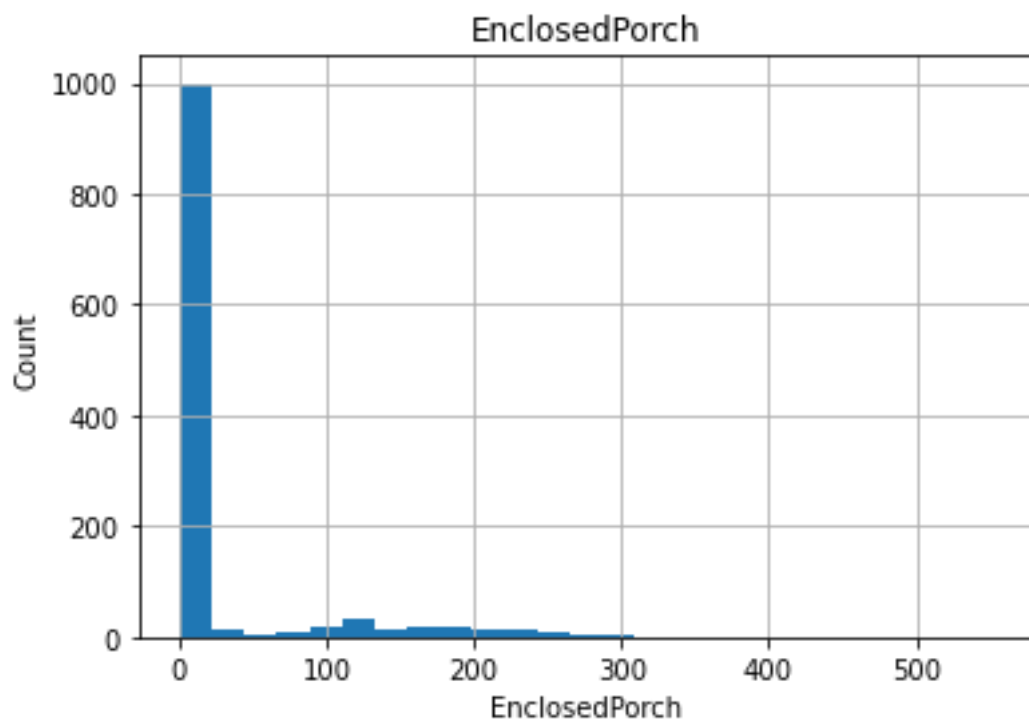
Histogram of BsmtFinSF1



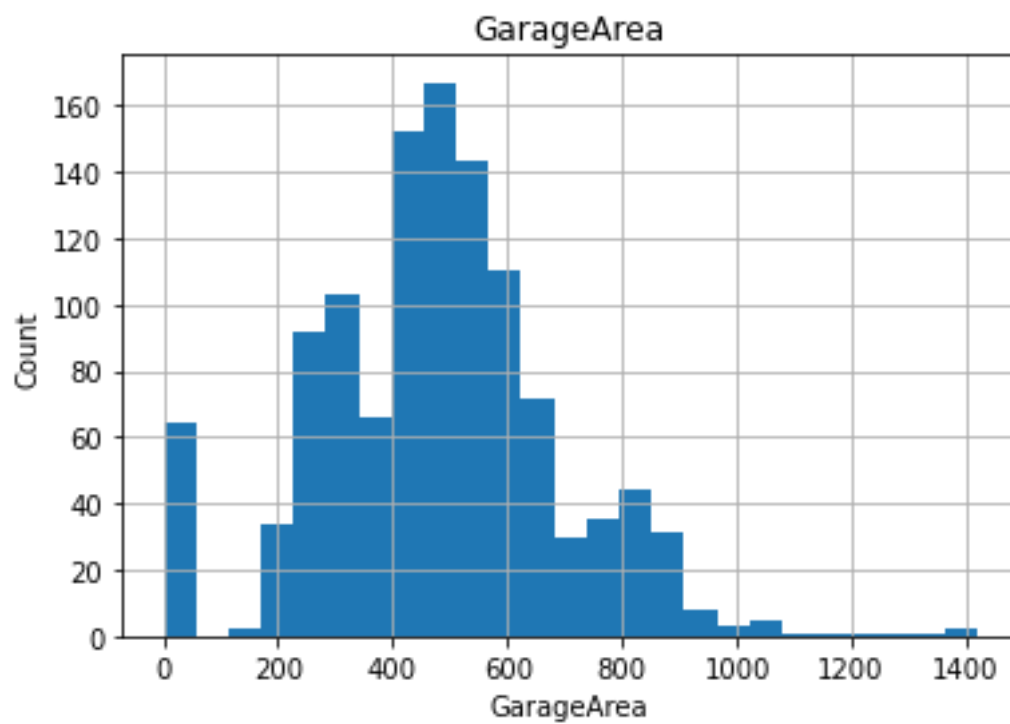
Histogram of BsmtfinSF2



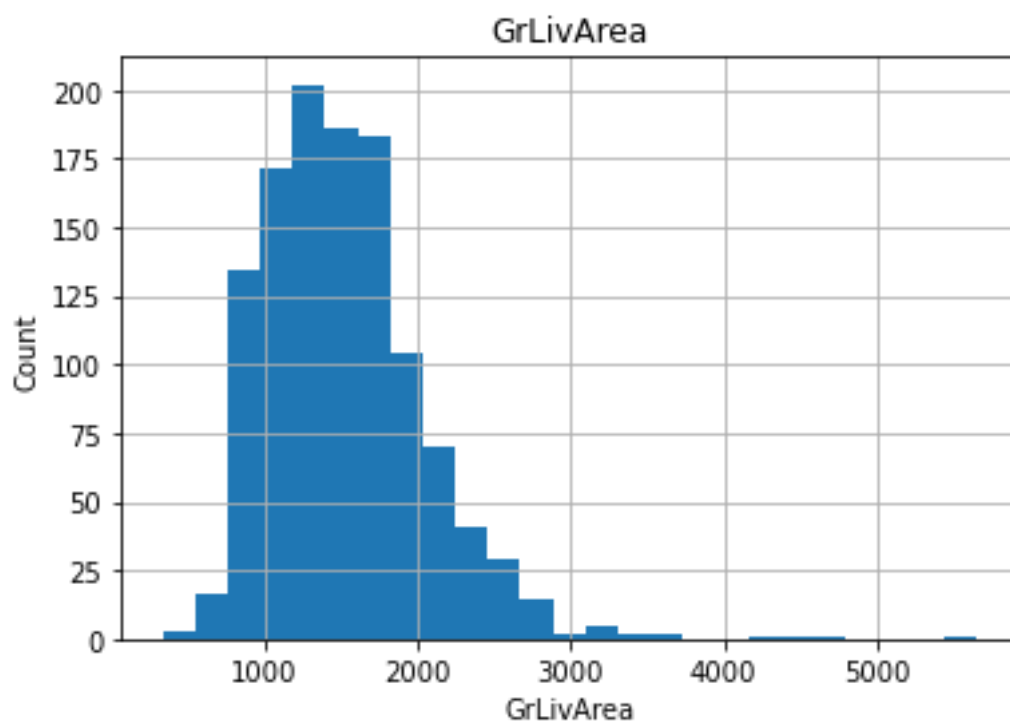
Histogram of BSmmtunfsf



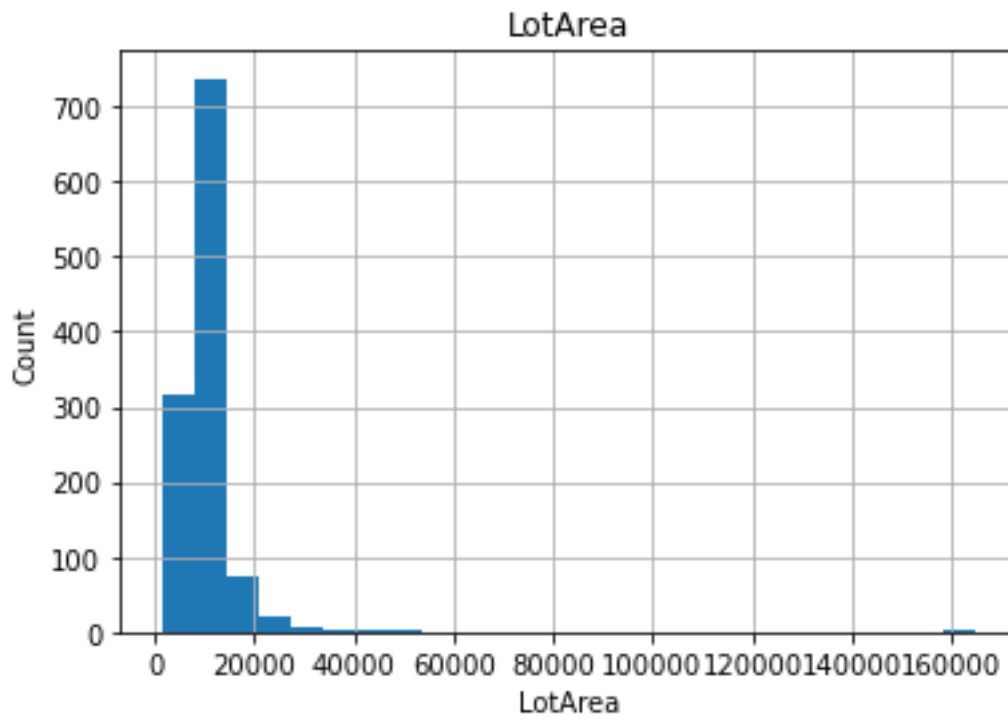
Histogram of EnclosedPorch



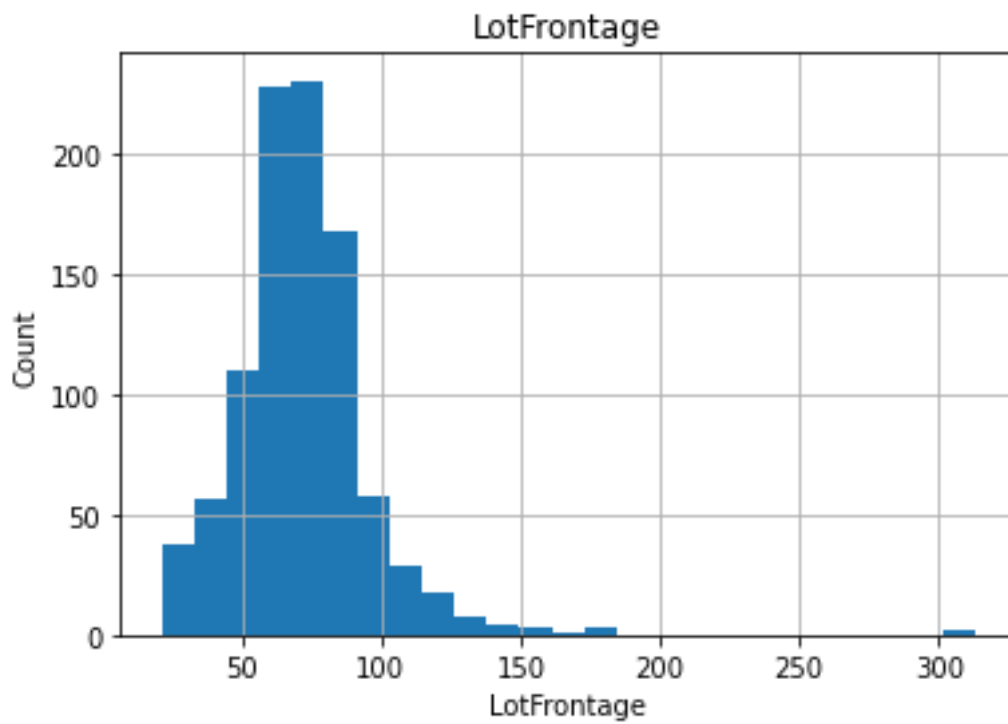
Histogram of GarageArea



Histogram of GrlivArea

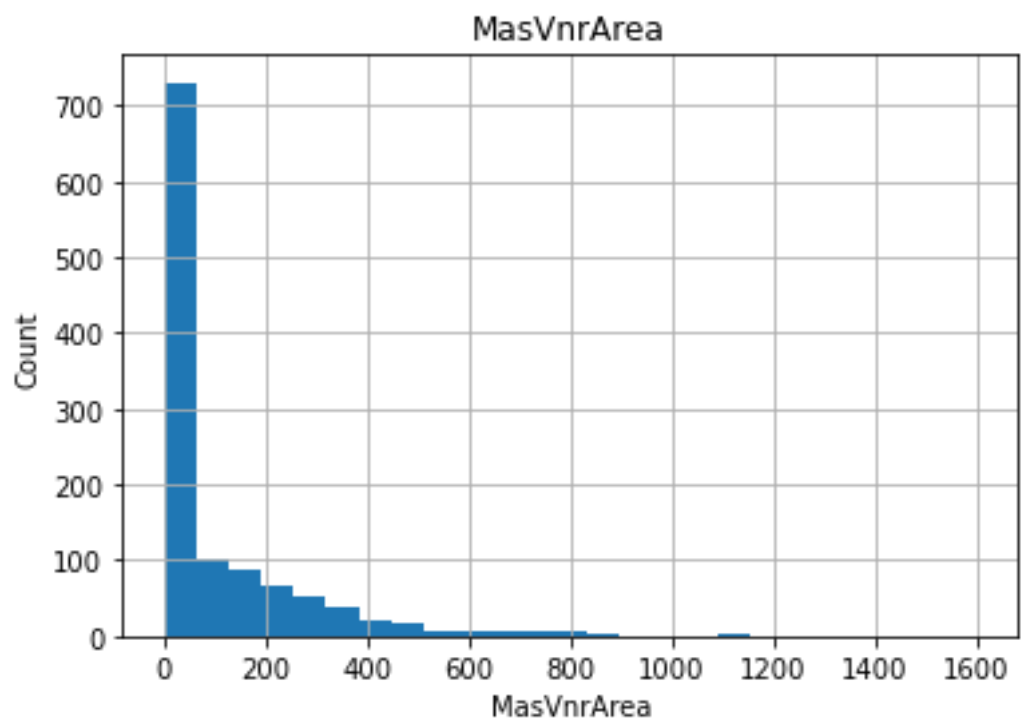


Histogram of LotArea

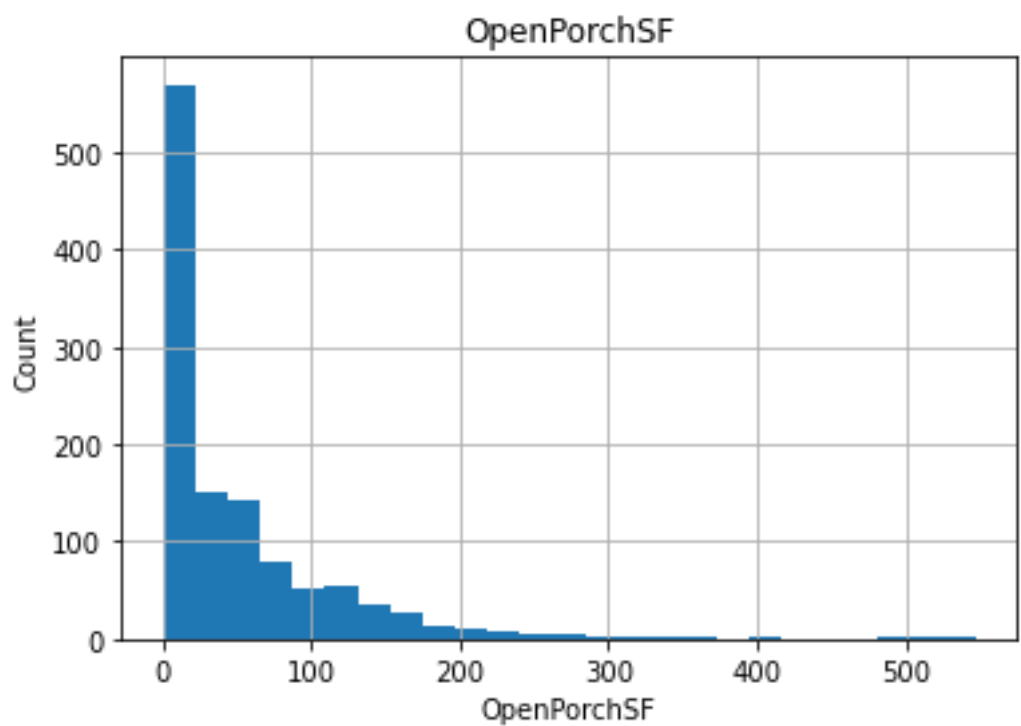


Histogram of LotFrontage

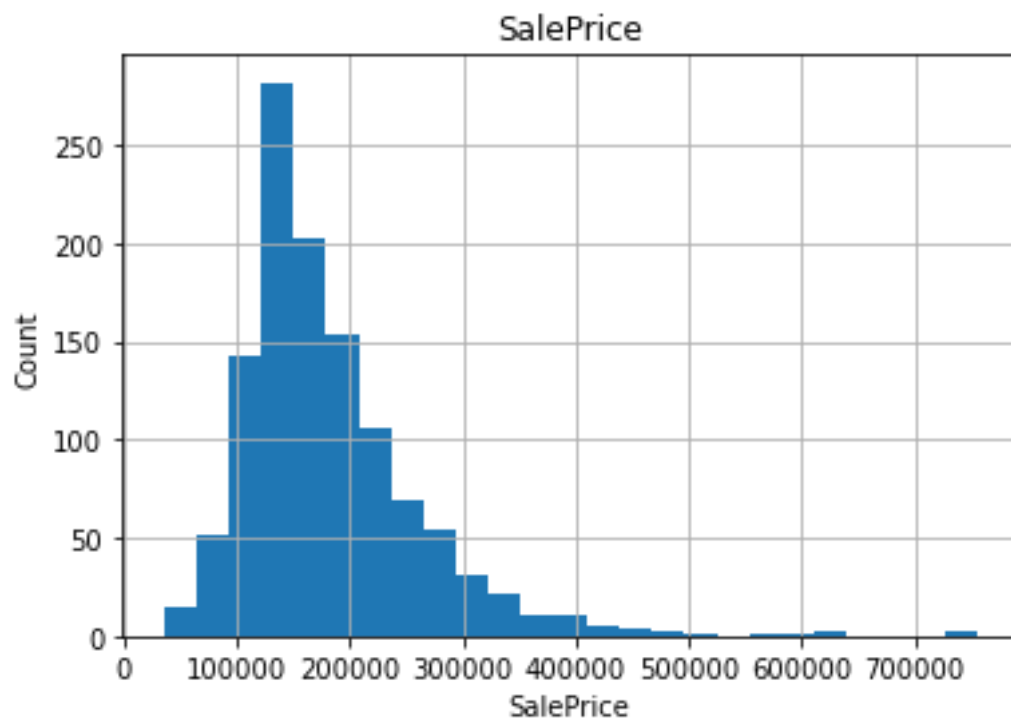




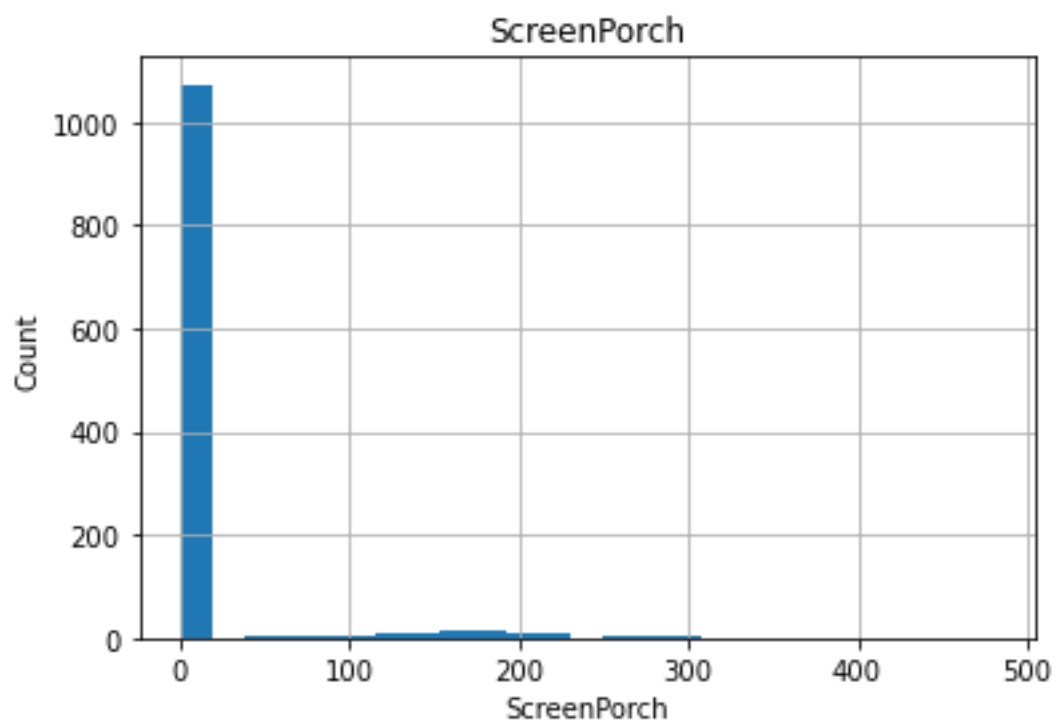
Histogram of MassvnArea



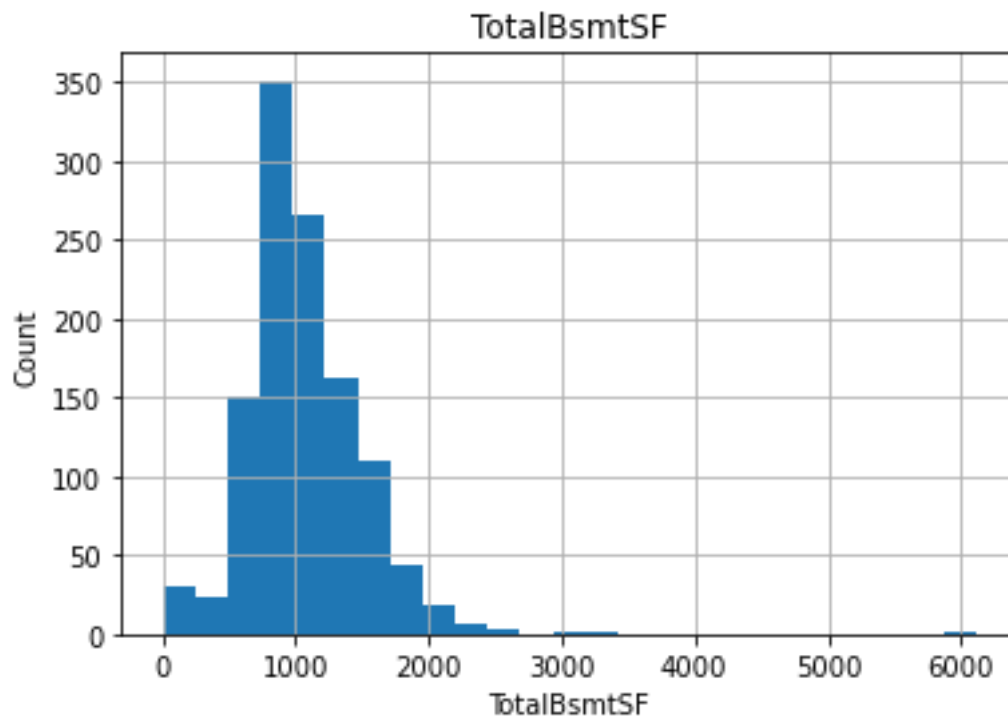
Histogram of OpenPorchSF



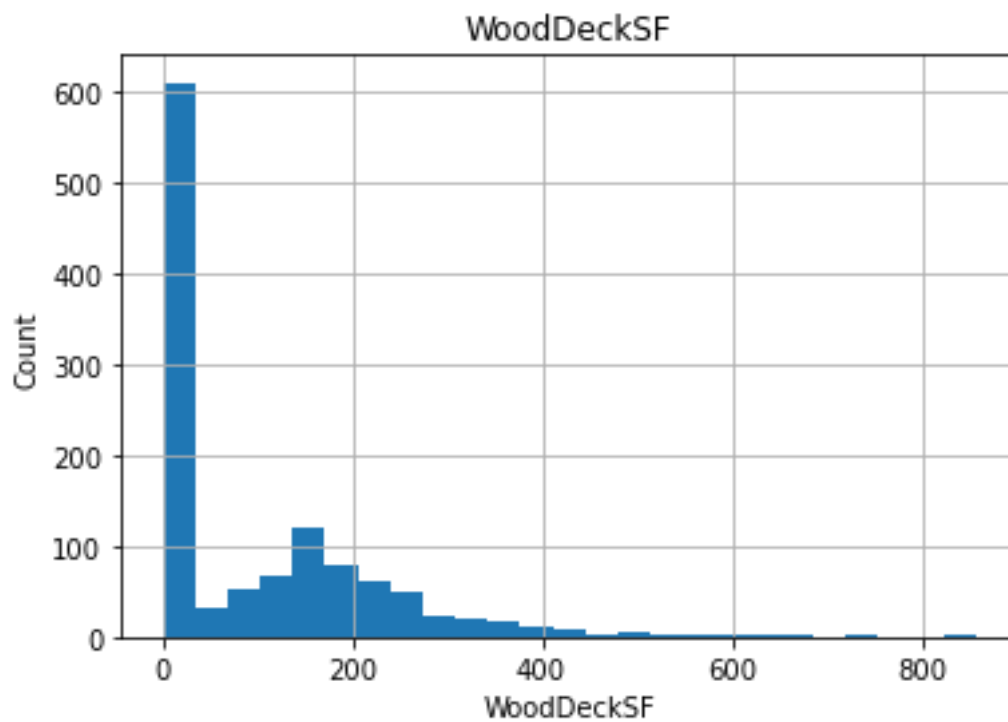
Histogram of SalePrice



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:-

```
for feature in continous_feature:
```

```
data=df.copy()
```

```
if 0 in data[feature].unique():
```

```
pass
```

```
else:
```

```
data[feature]=np.log(data[feature])
```

```
data['SalePrice']=np.log(data['SalePrice'])
```

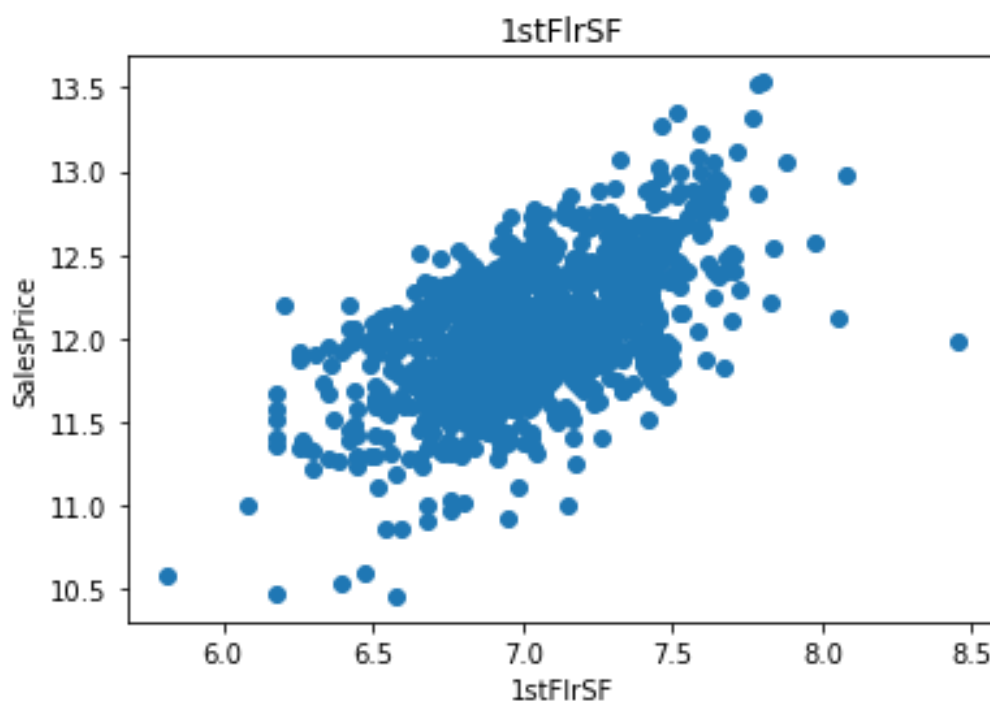
```
plt.scatter(data[feature],data['SalePrice'])
```

```
plt.xlabel(feature)
```

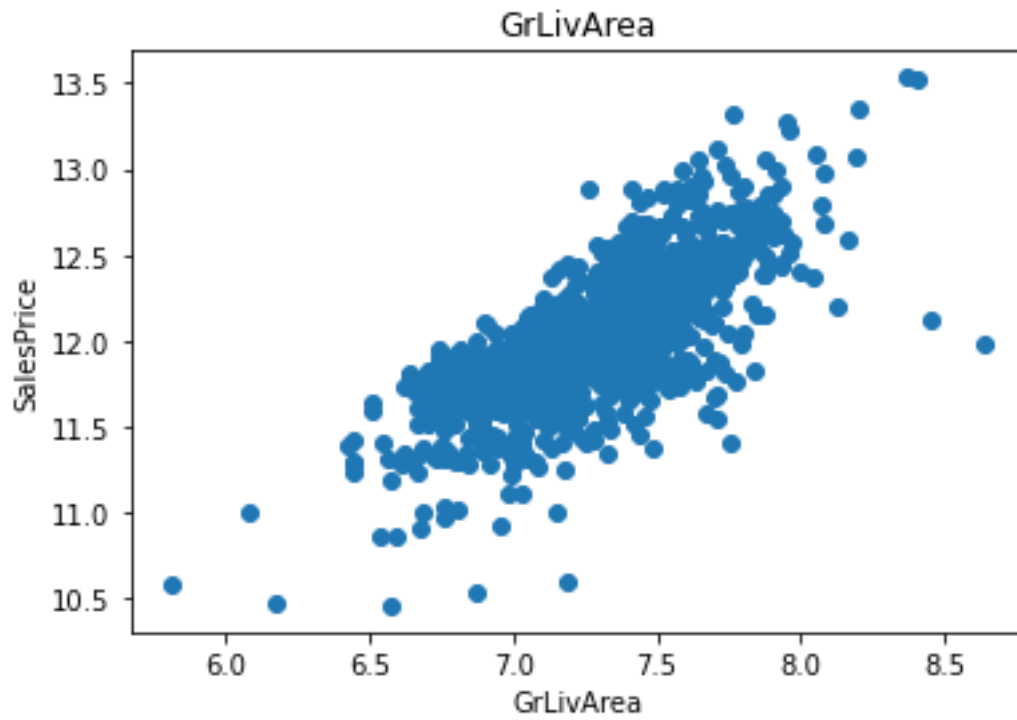
```
plt.ylabel('SalesPrice')
```

```
plt.title(feature)
```

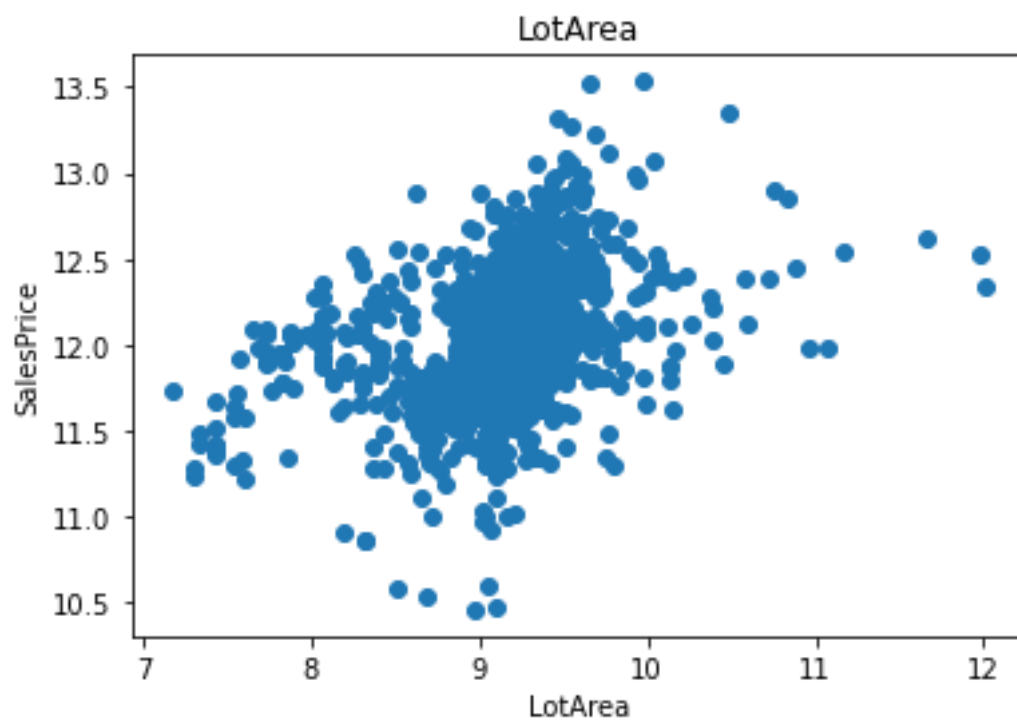
```
plt.show()
```



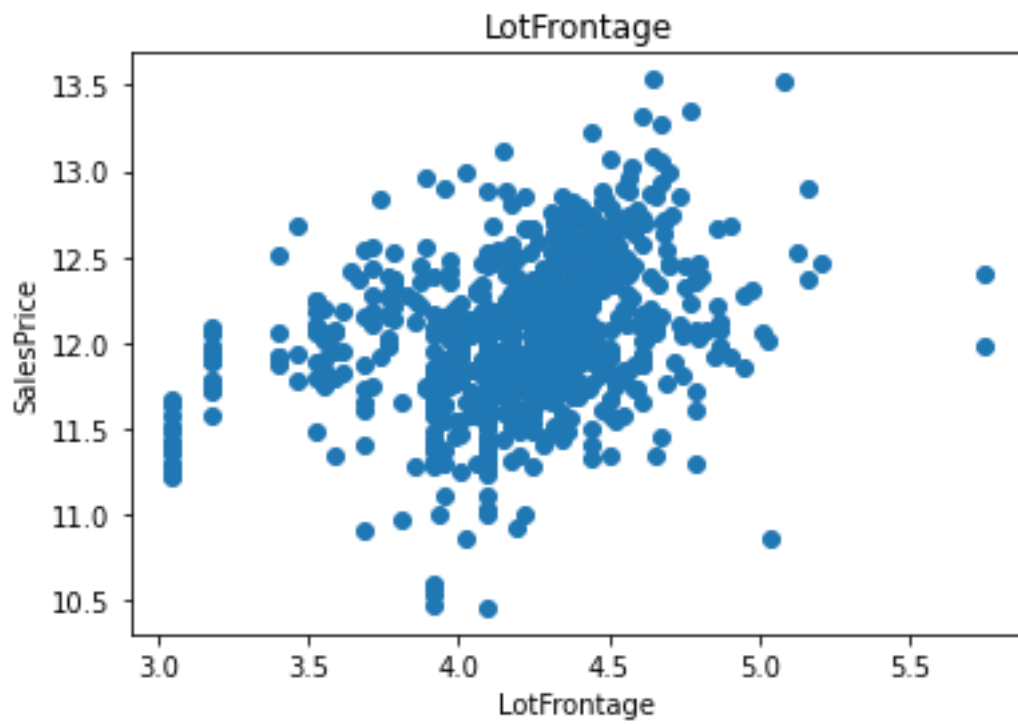
Saleprice vs 1StFlrSF



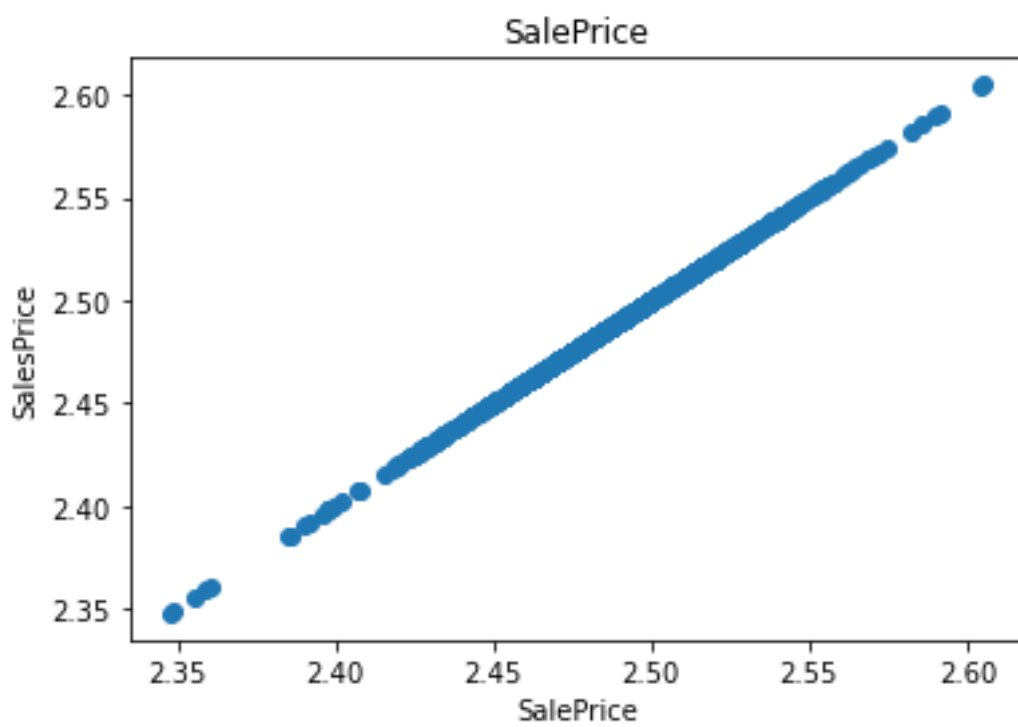
Saleprice vs GLivArea



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)
```

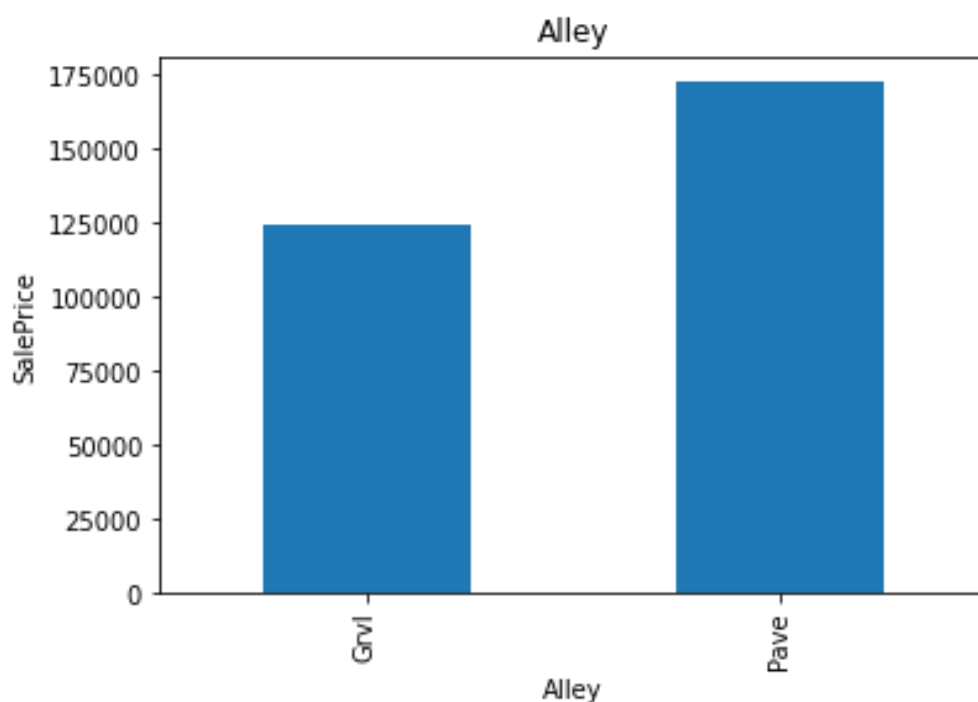
```
    plt.ylabel(feature)
```

```
    plt.title(feature)
```

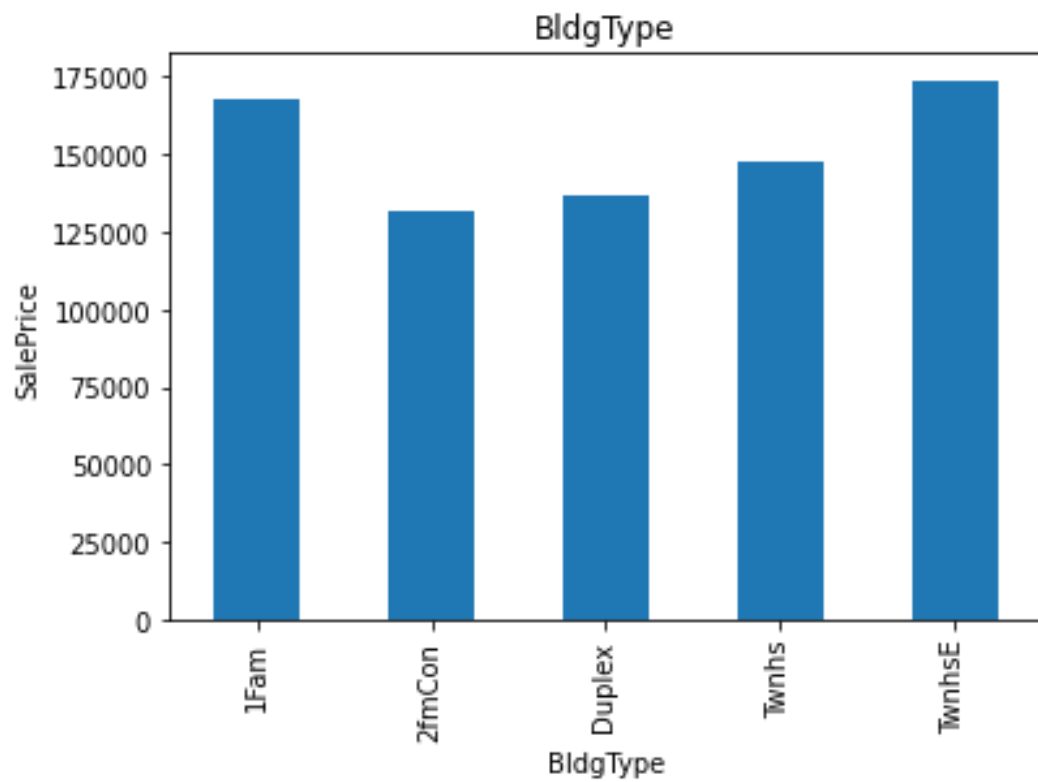
```
    plt.show()
```

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

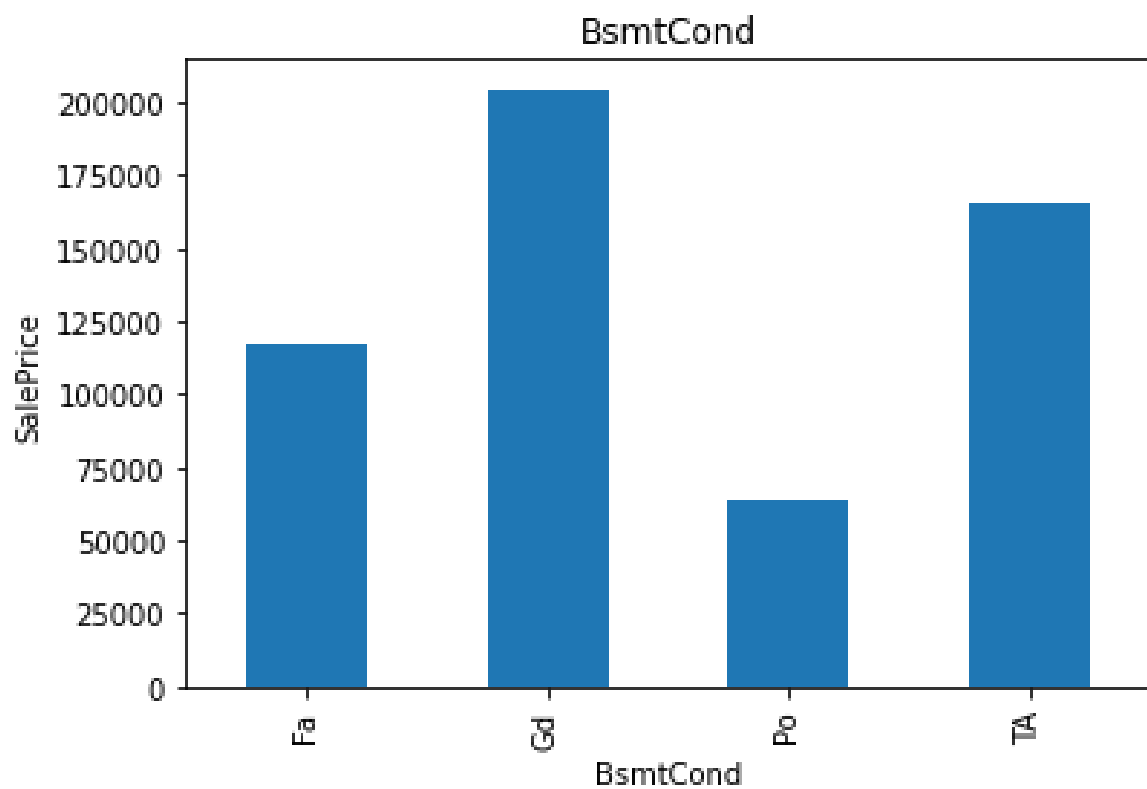
Realation Between categorical feature and SalePrice



Salepice Vs Alley

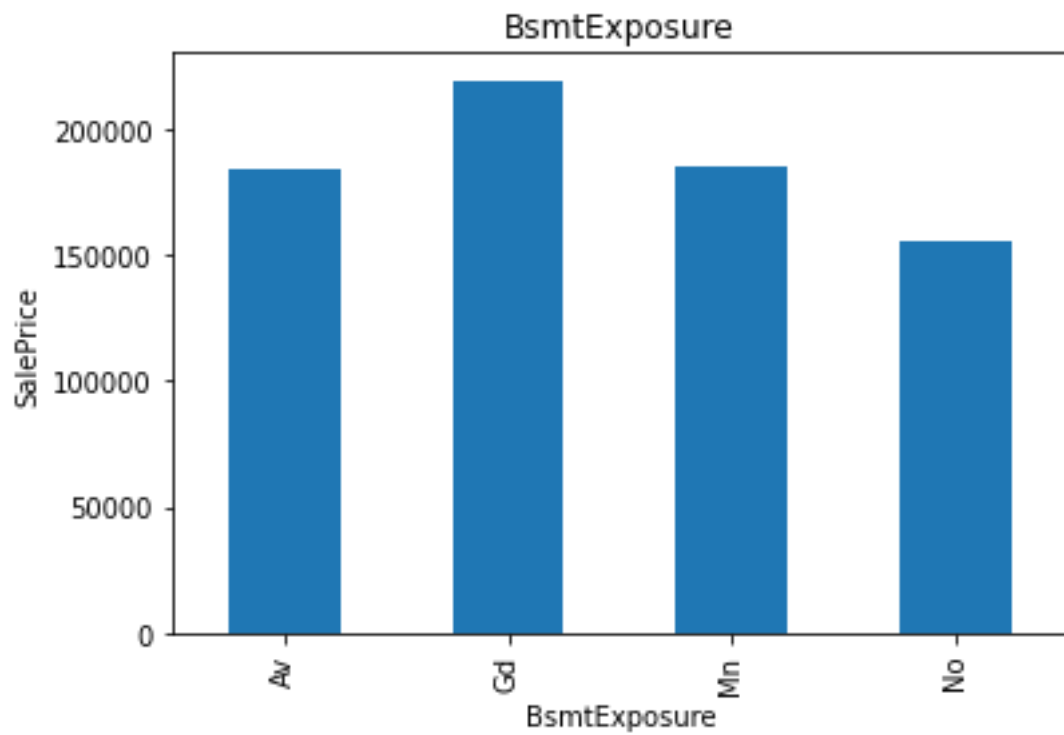


SalePrice Vs BldgType

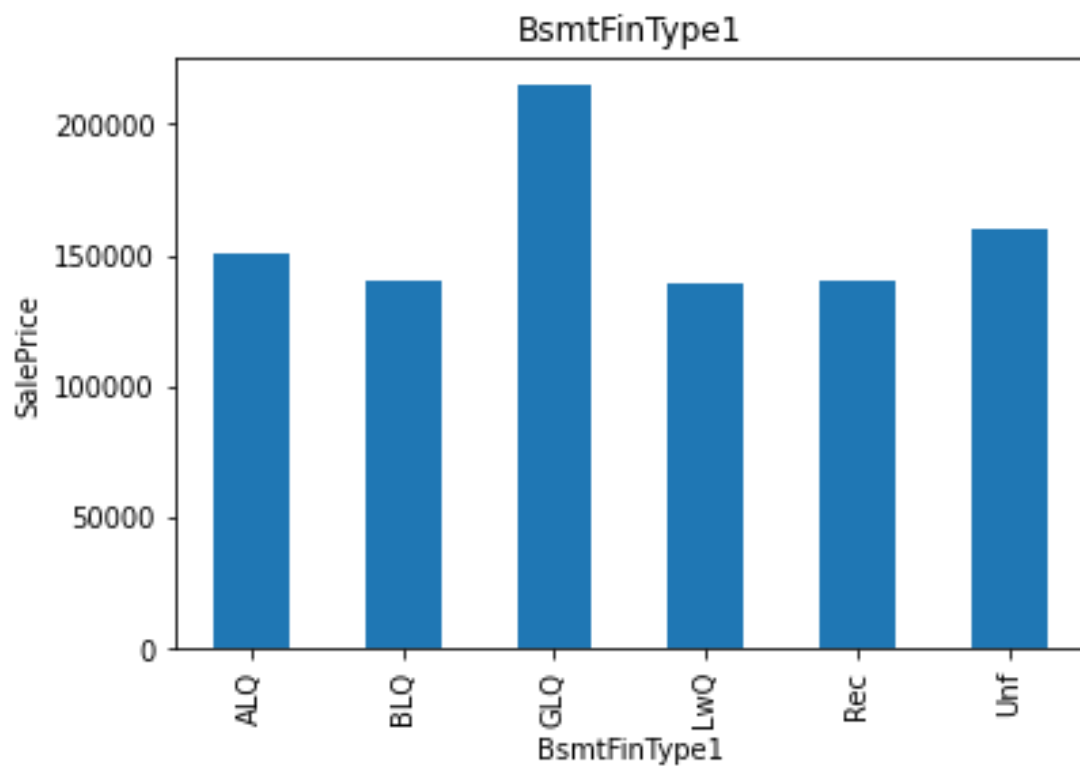


SalePrice Vs BSmt Cond

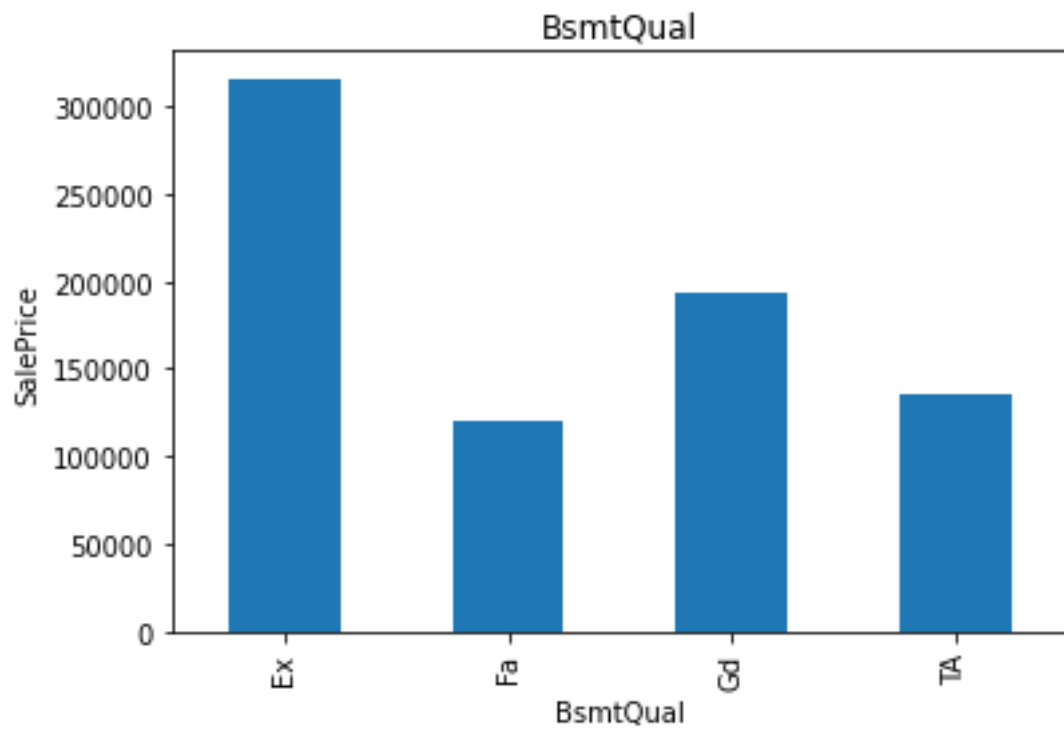




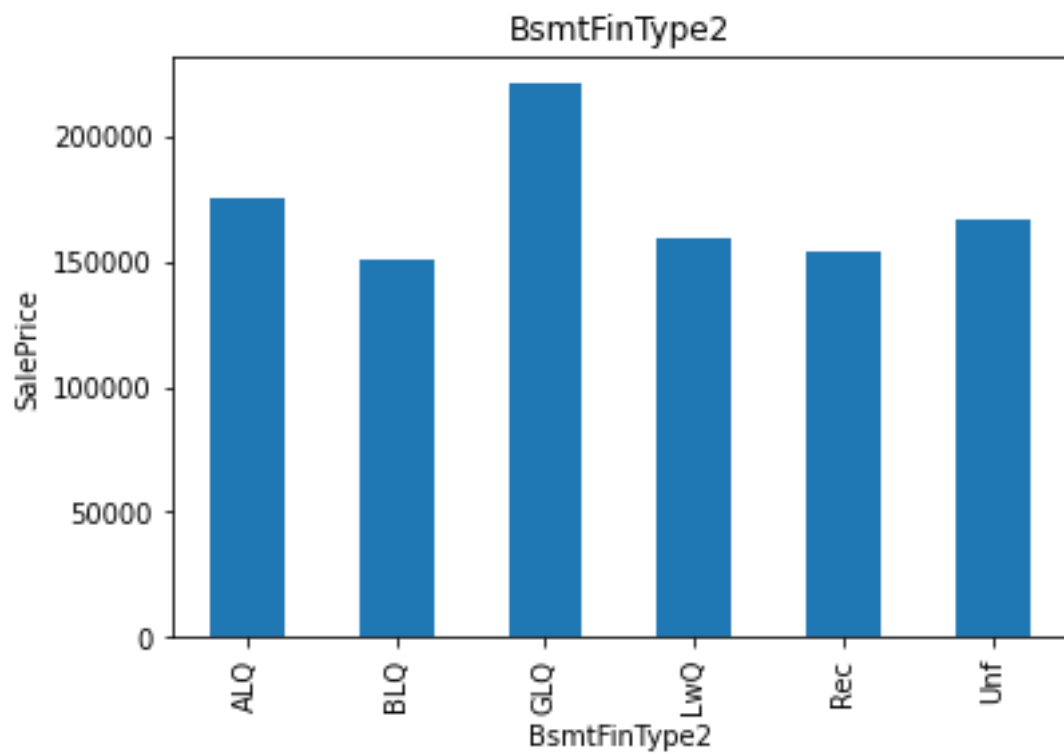
SalePrice Vs BsmtExposure



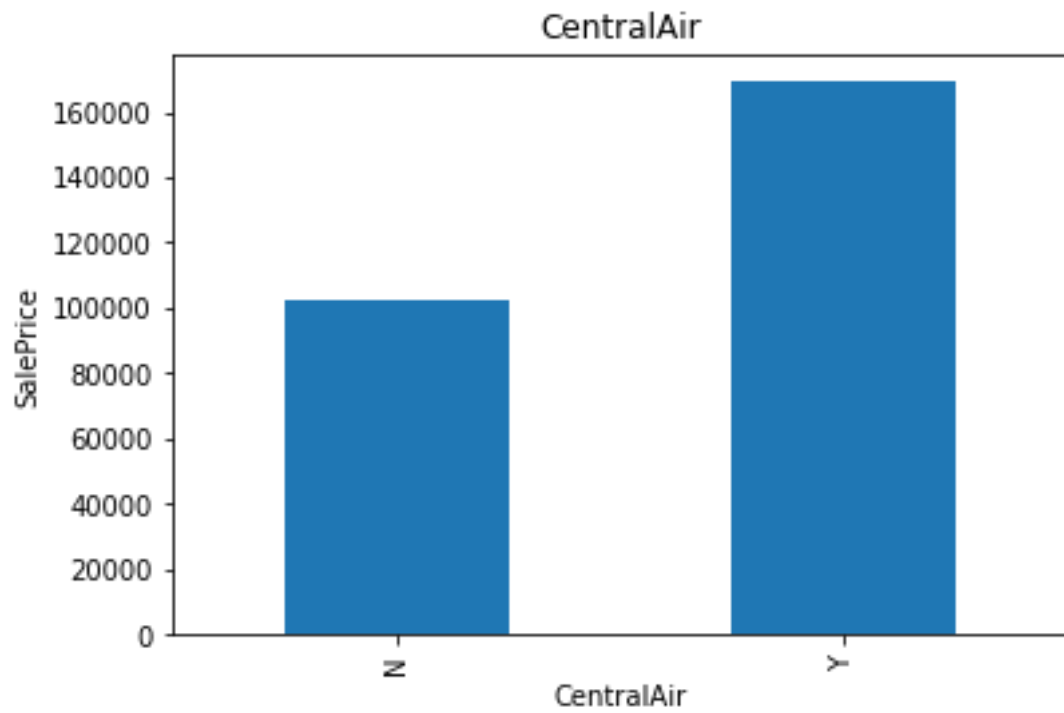
SalePrice VS BsmtFinType1



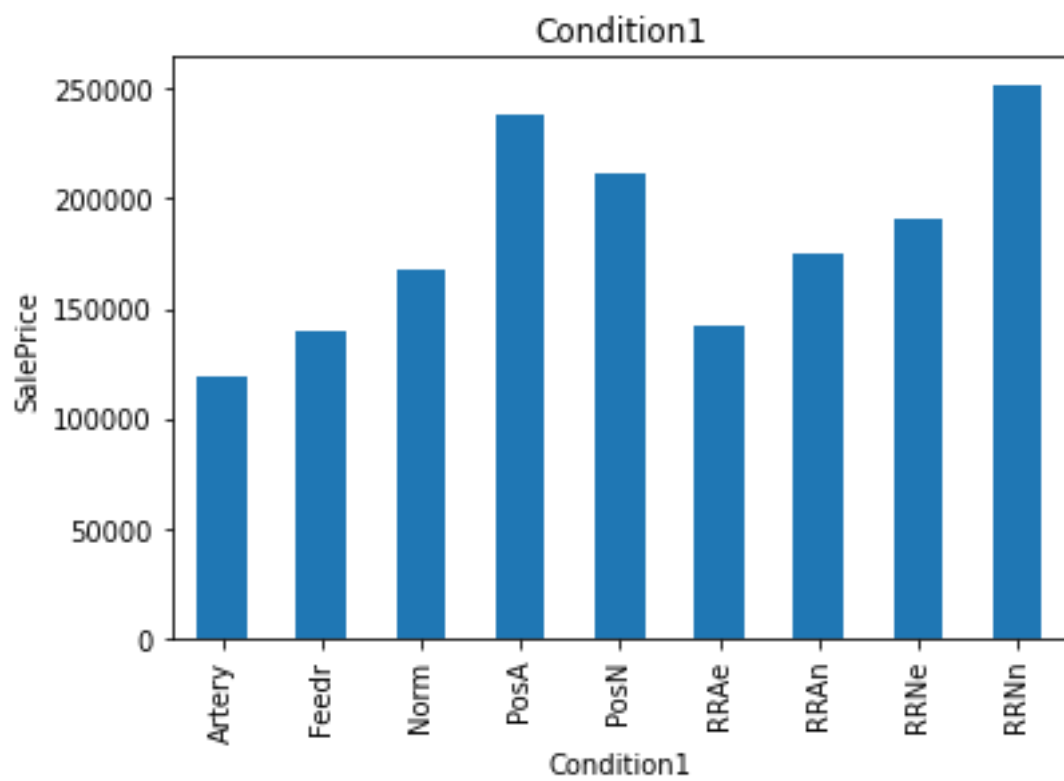
SalePrice VS BsmtQual



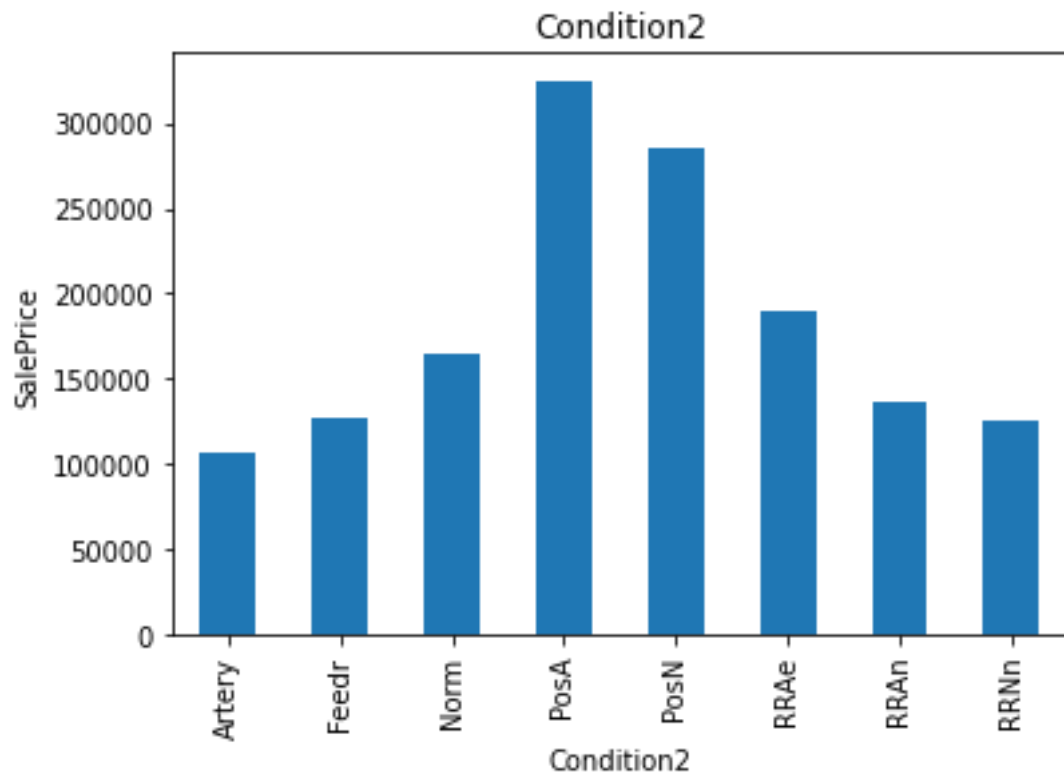
Saleprice vs BsmtFinType2



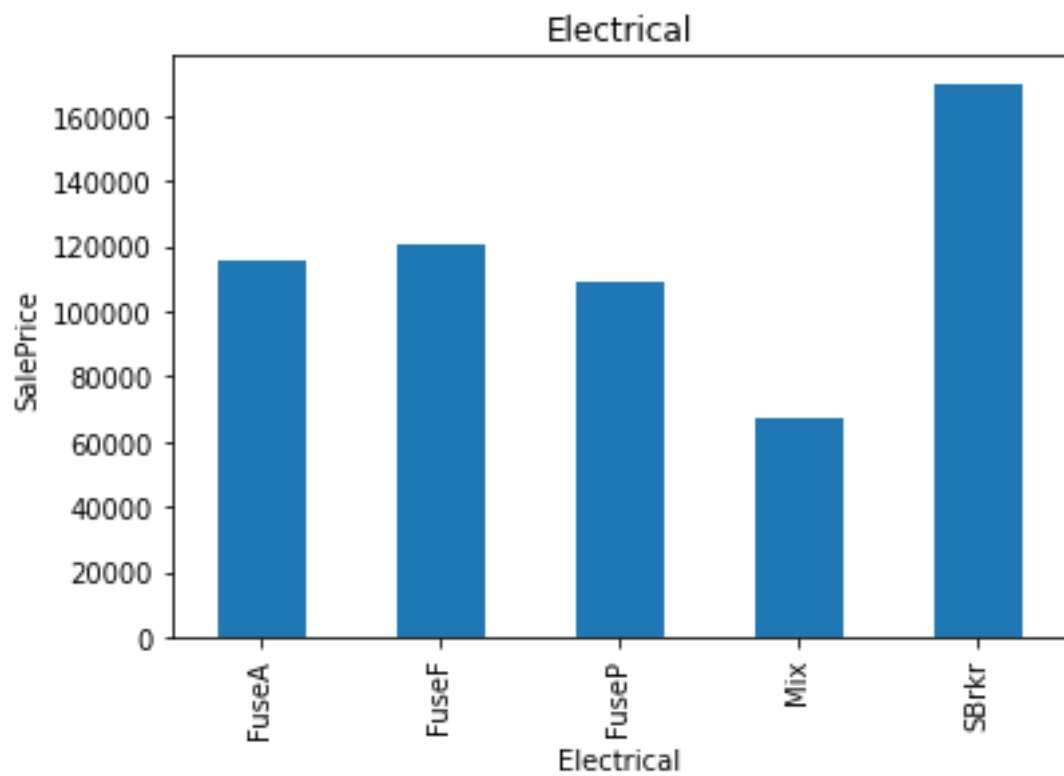
SalePrice Vs Central Air



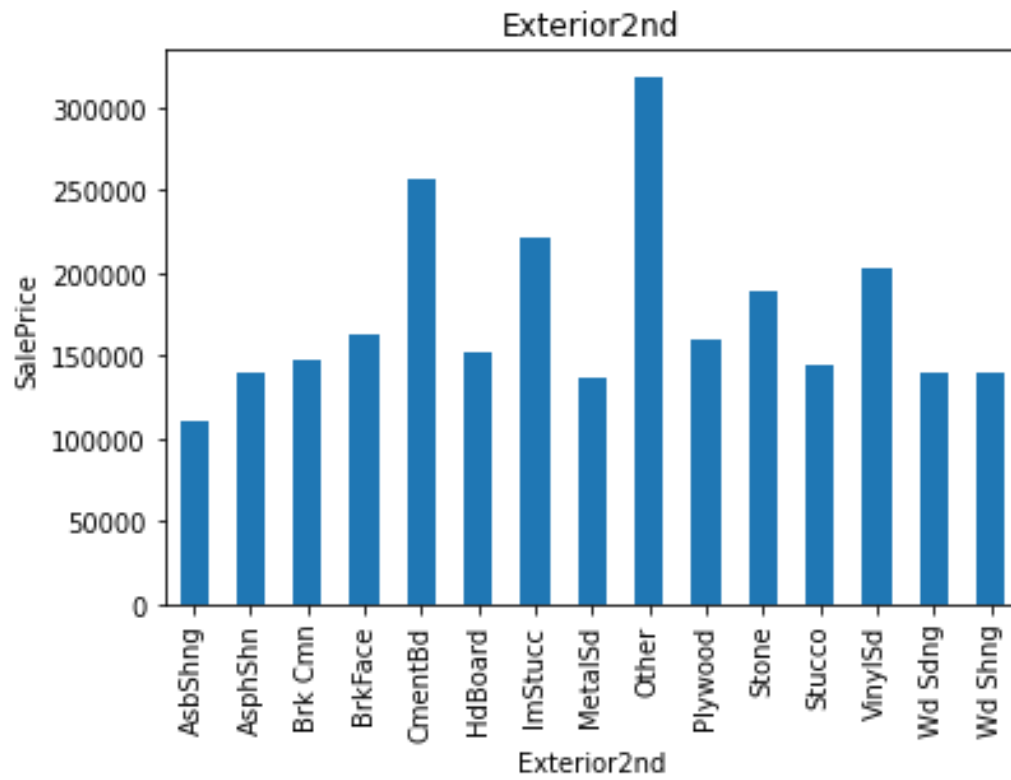
SalePrice Vs Condition1



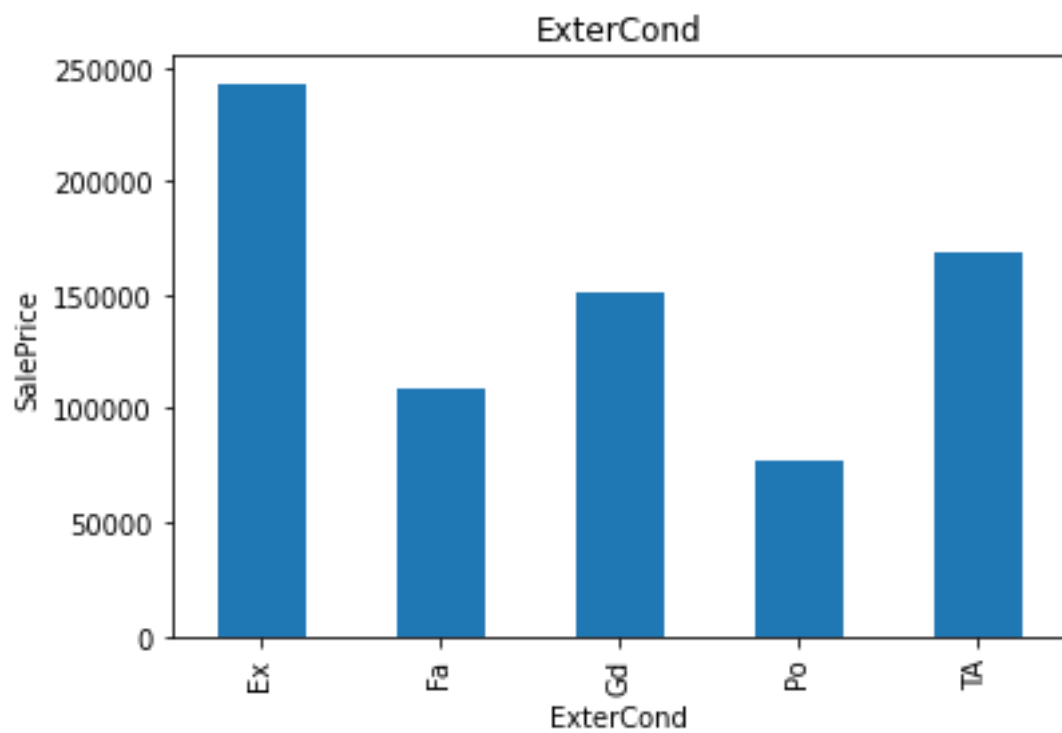
SalePrice Vs condition2



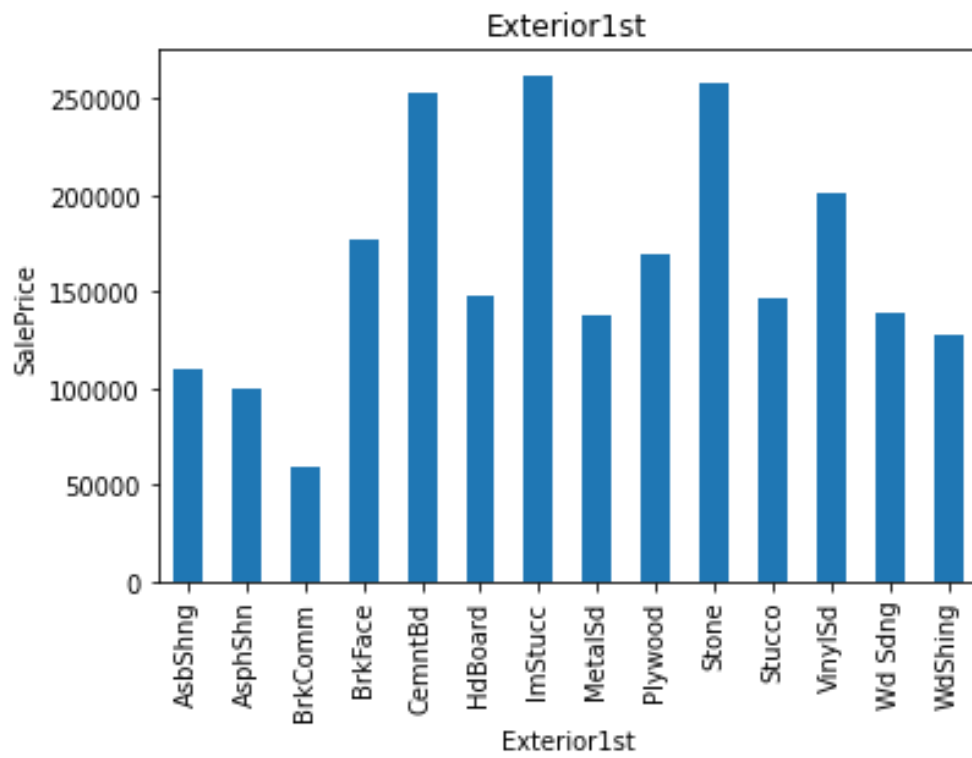
Saleprice Vs Electrical



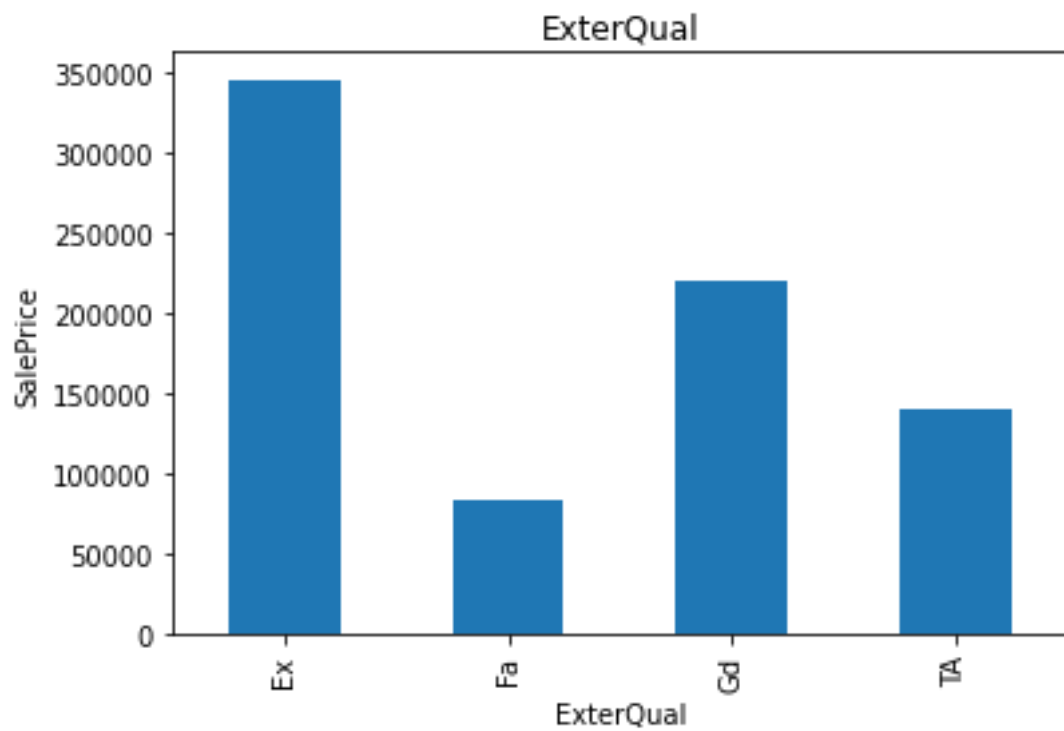
Saleprice Vs Exterior2nd



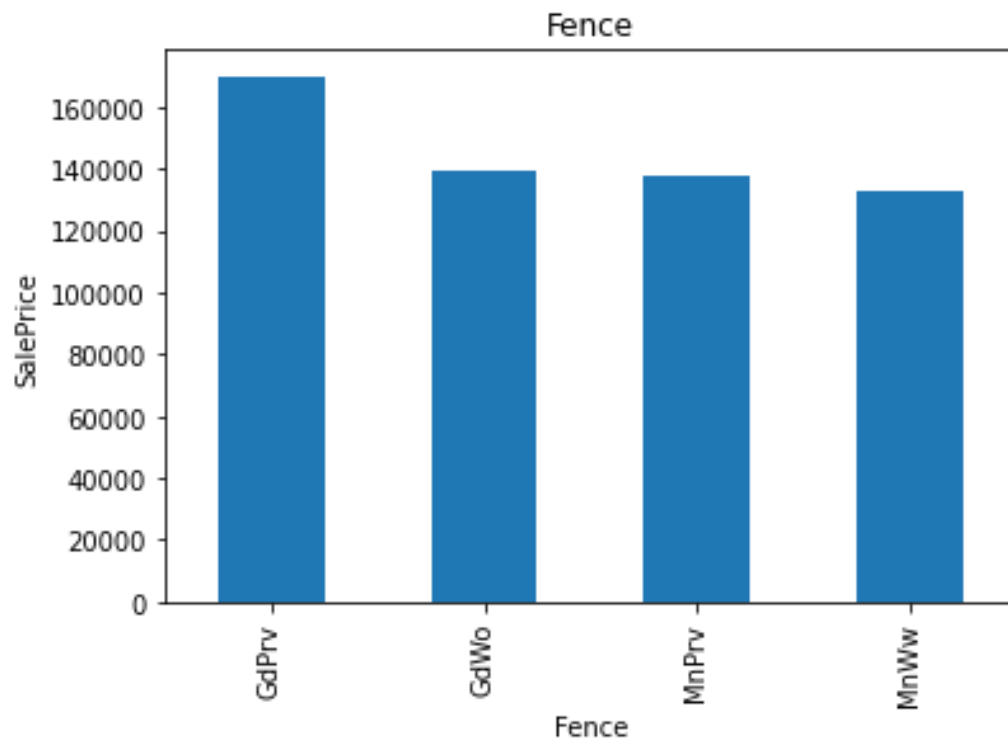
Saleprice Vs ExterCond



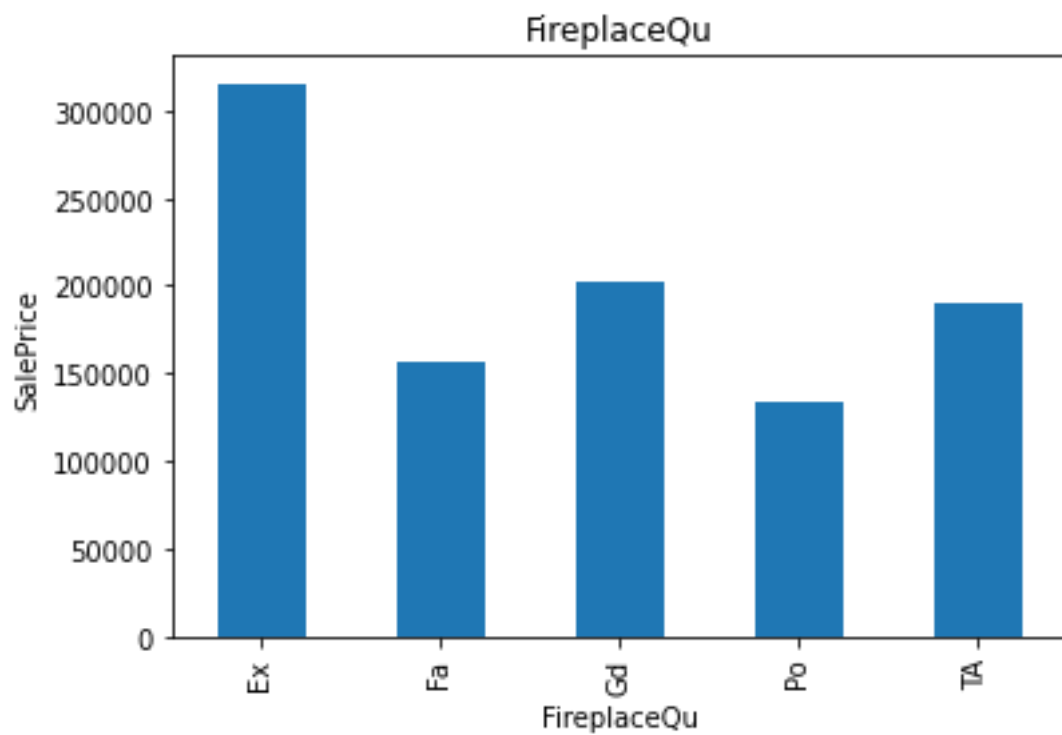
SalePrice Vs Exterior 1st



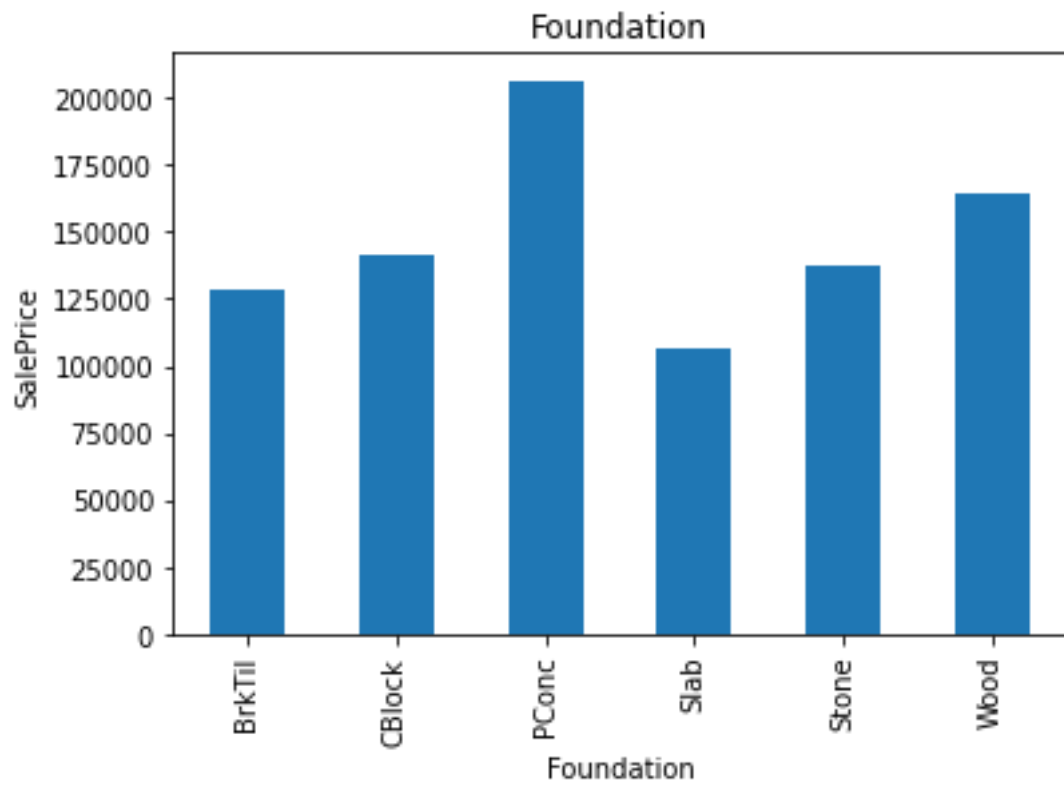
SalePrice VS ExteriorQual



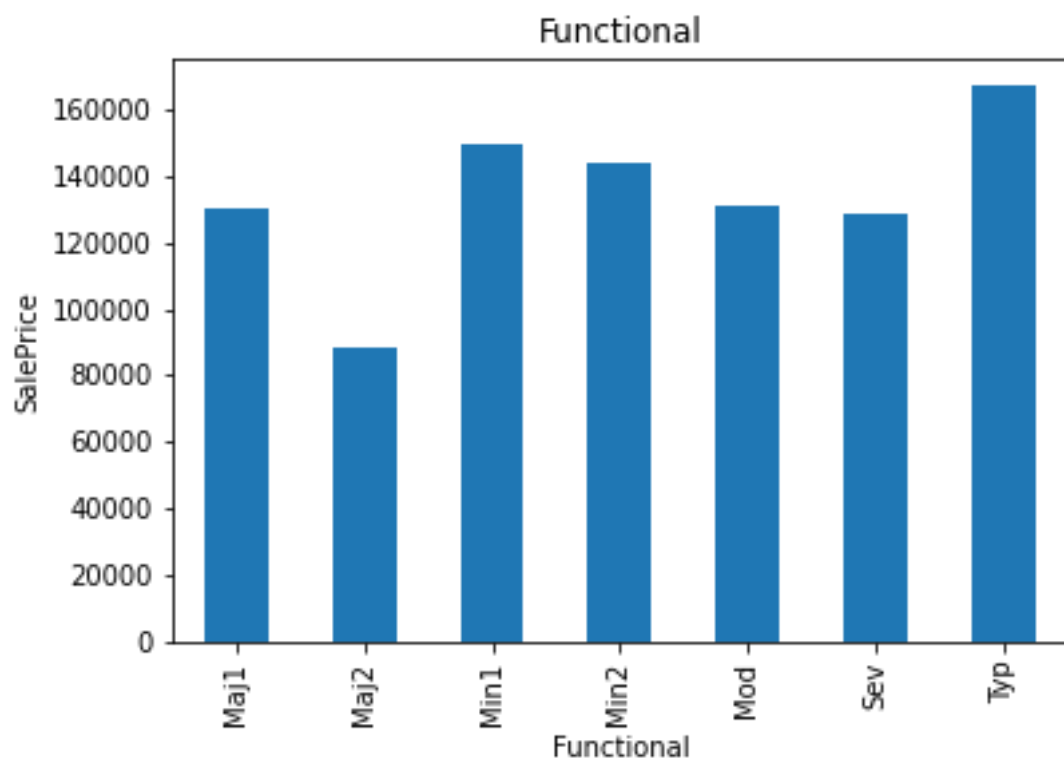
SalePrice Vs Fence



SalePrice Vs FirePlaceQU

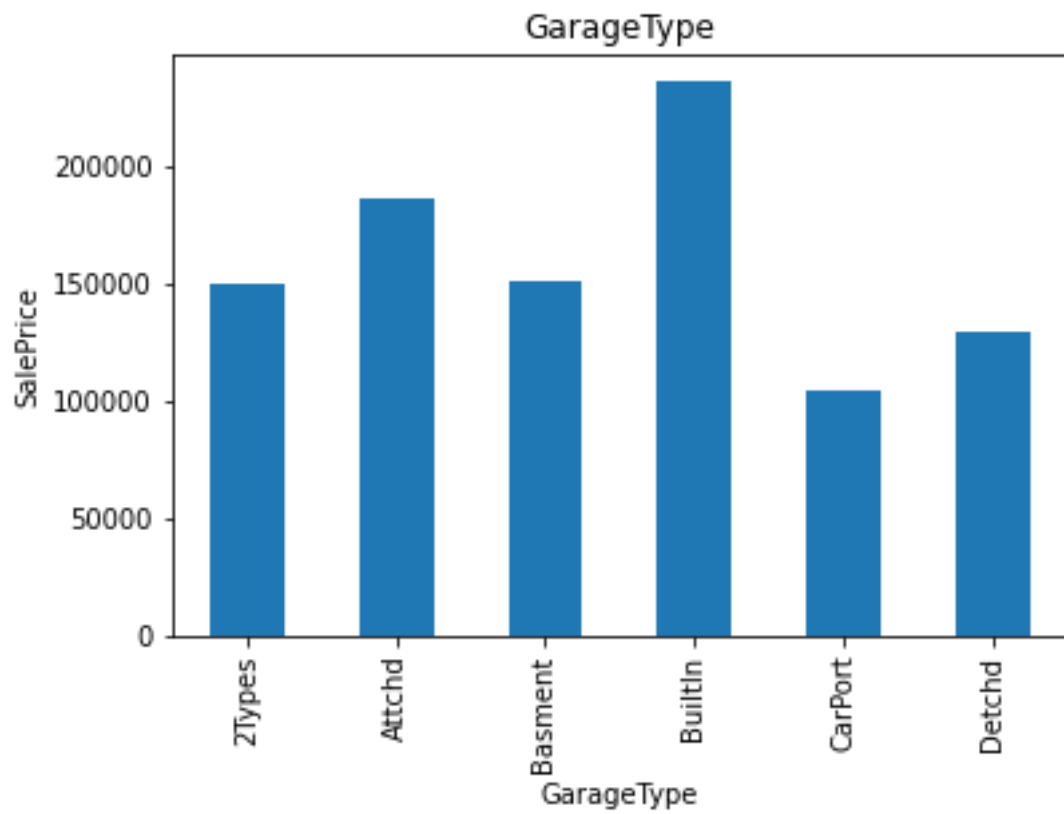


SalePrice Vs Foundation

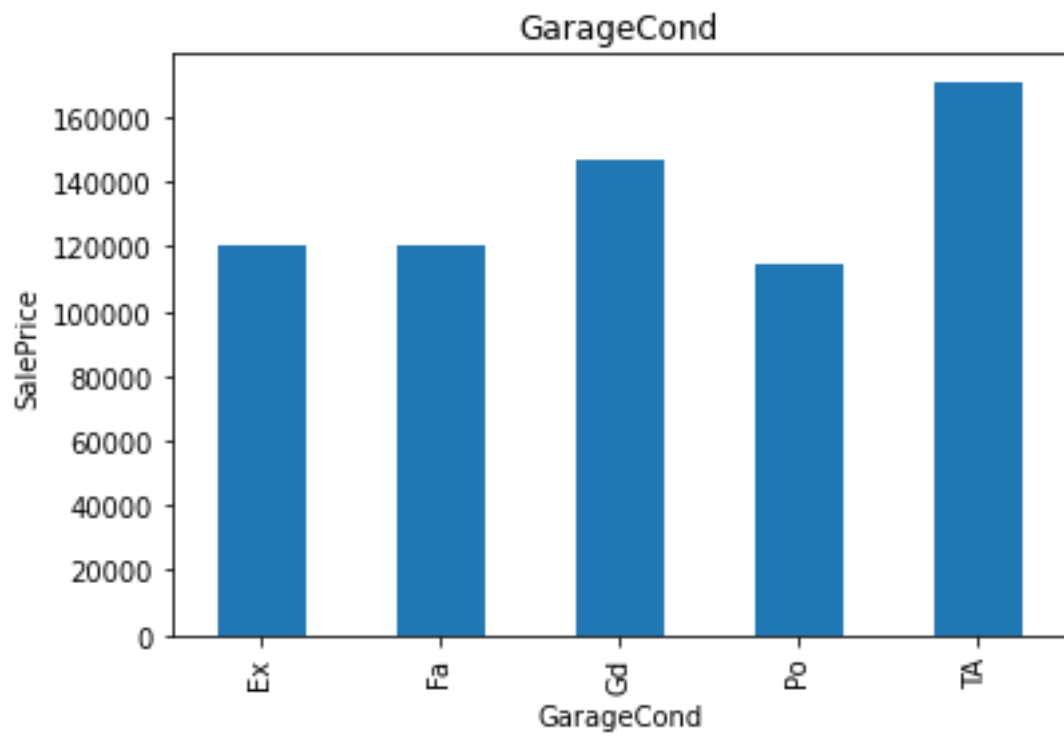


SalePrice Vs Functional

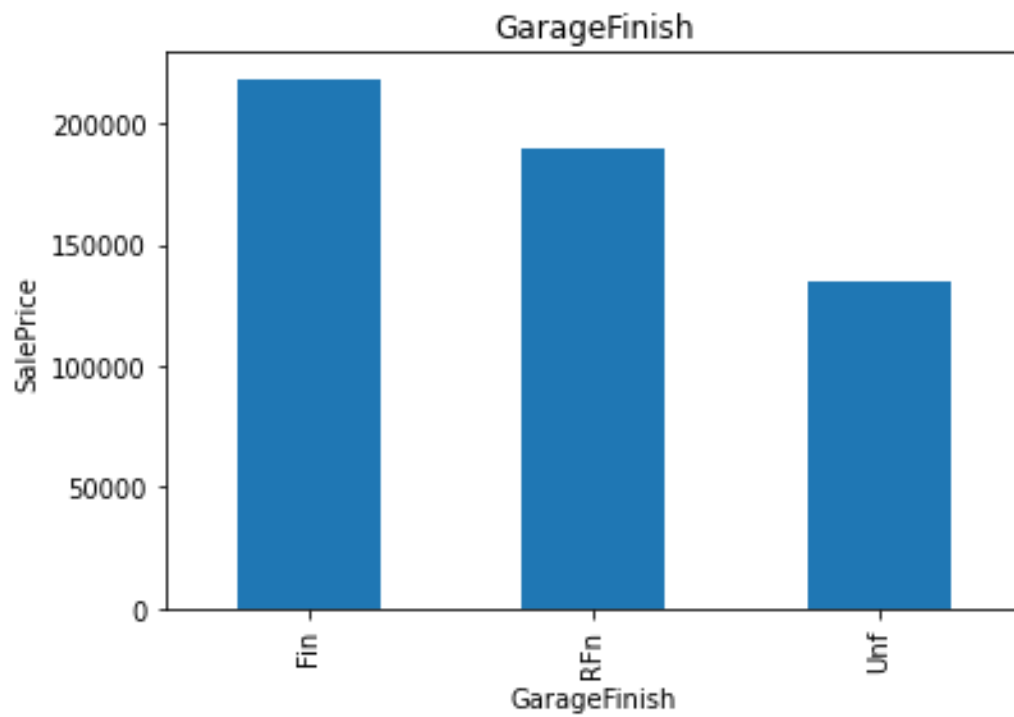




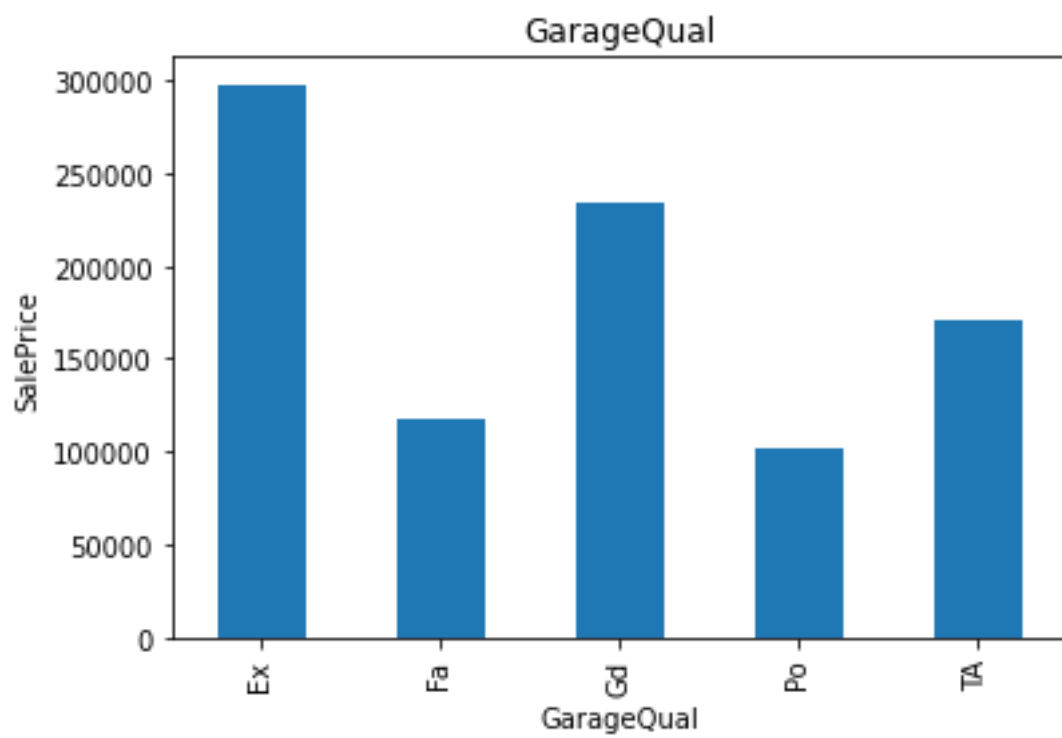
SalePrice Vs GarageType



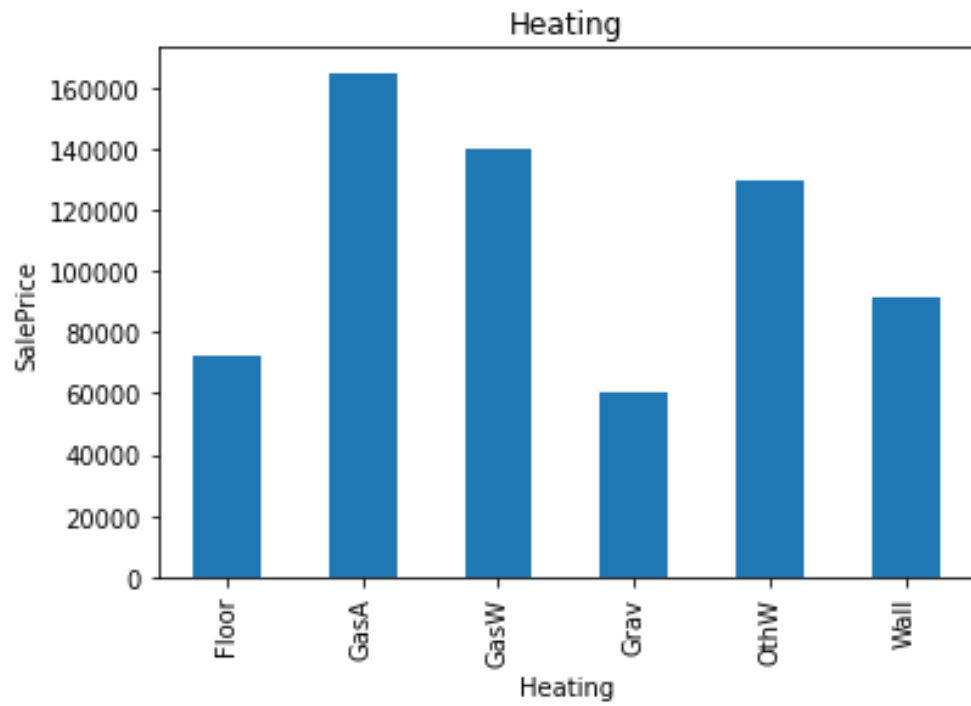
SalaryPrice Vs Garagecond



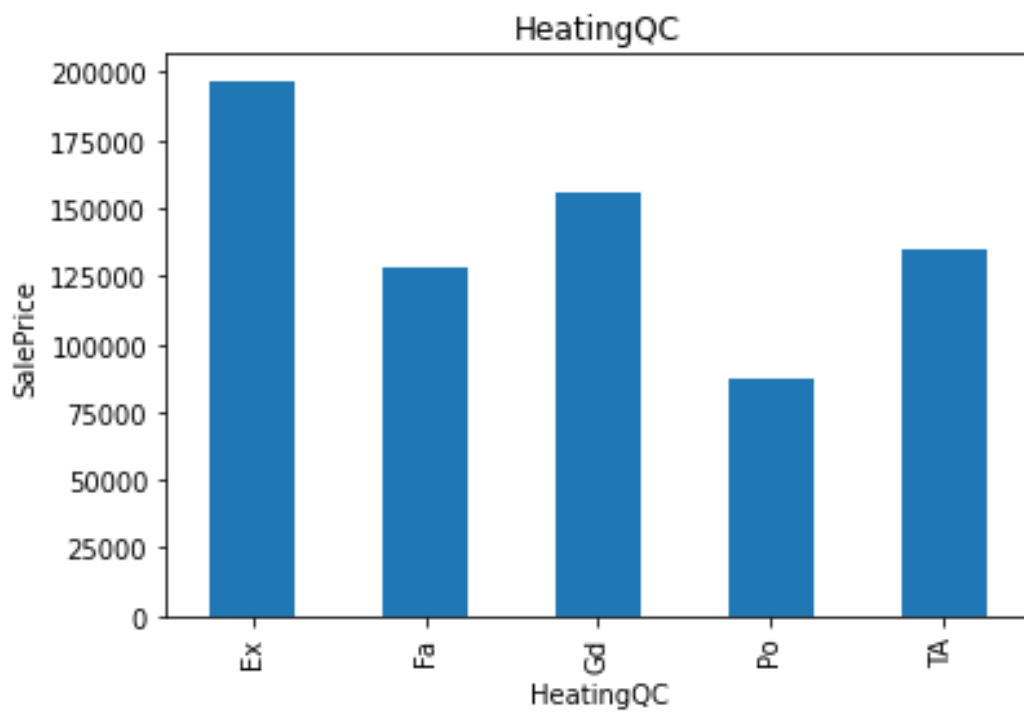
SalePrice Vs GargeFinish



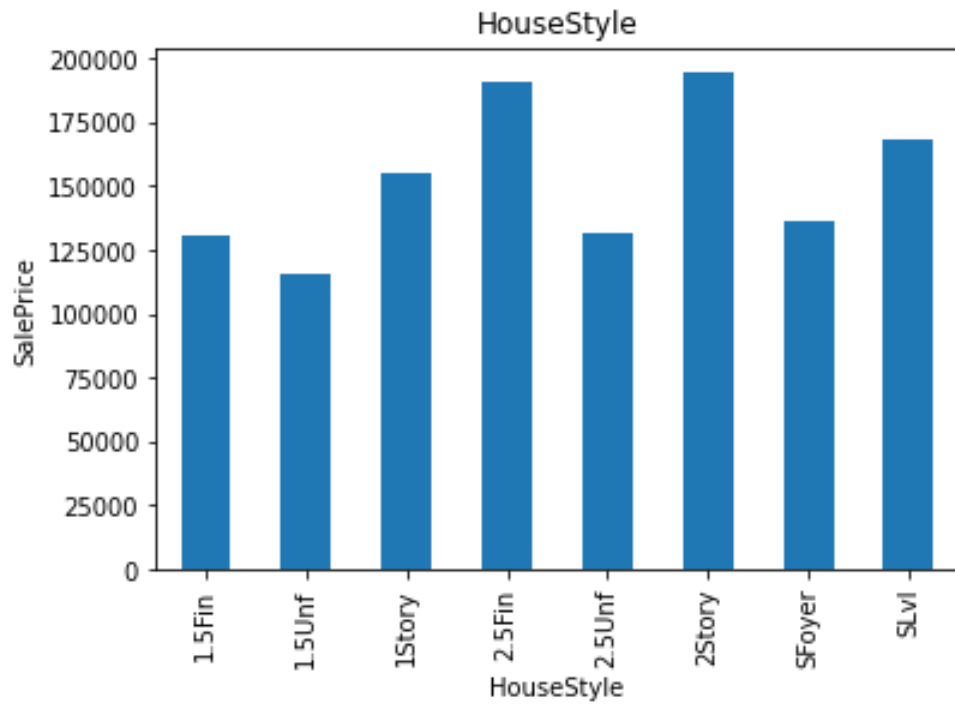
SalePrice Vs GarageQual



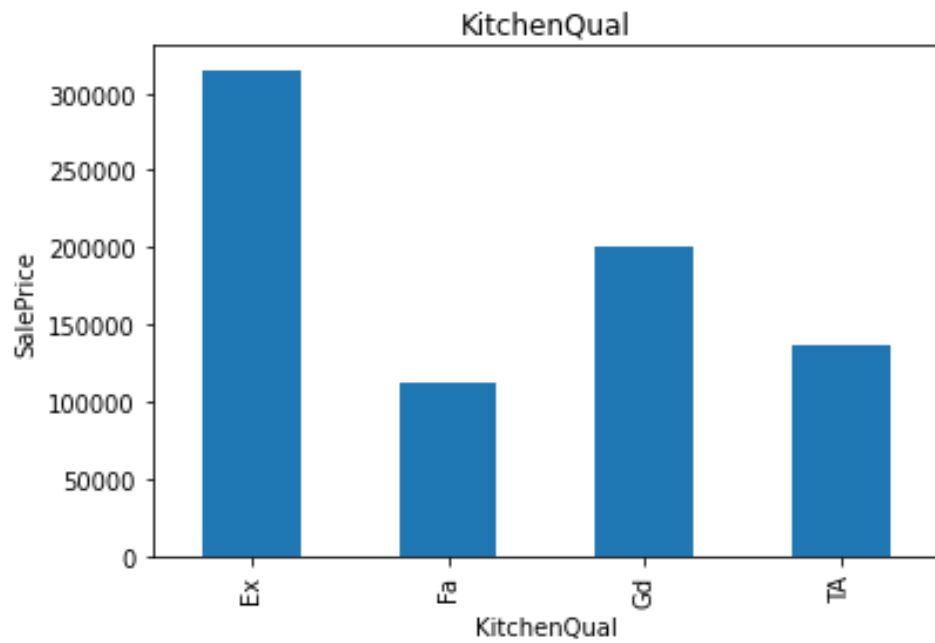
SalePrice Vs Heating



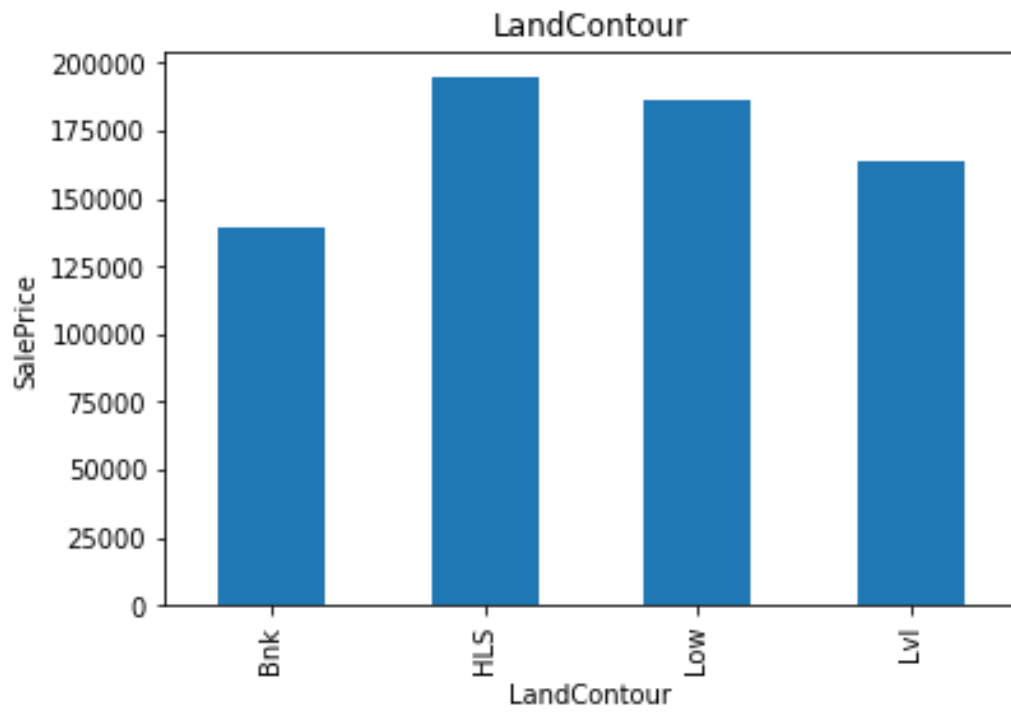
SalePrice Vs HeatingQC



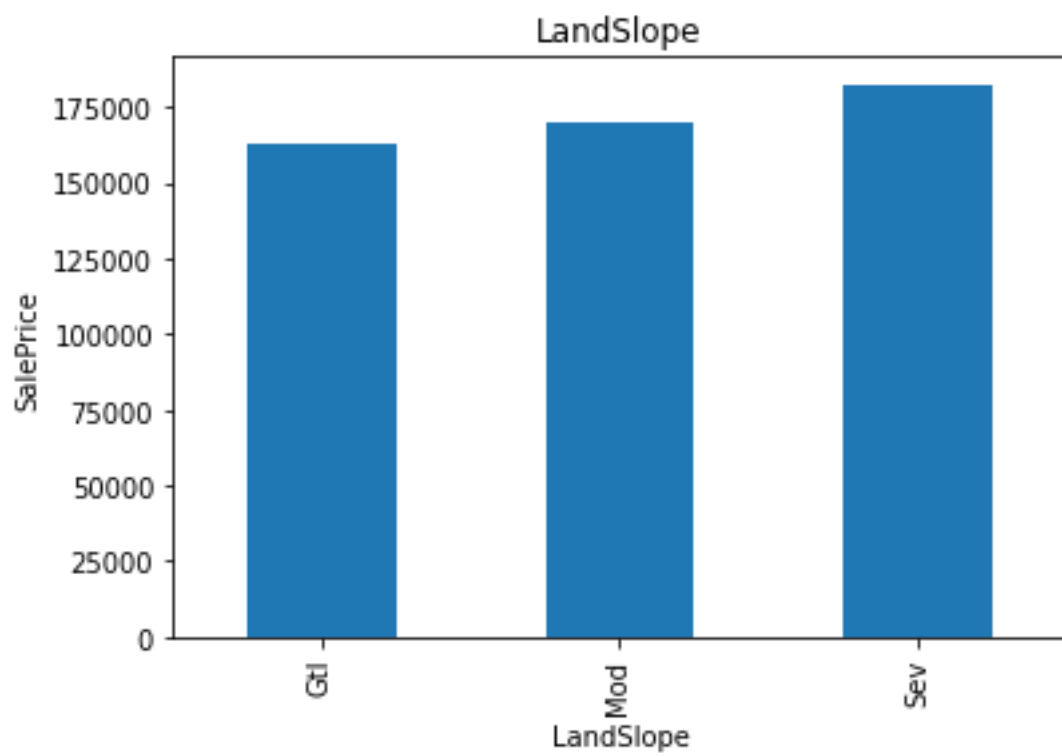
SalePrice Vs HouseStyle



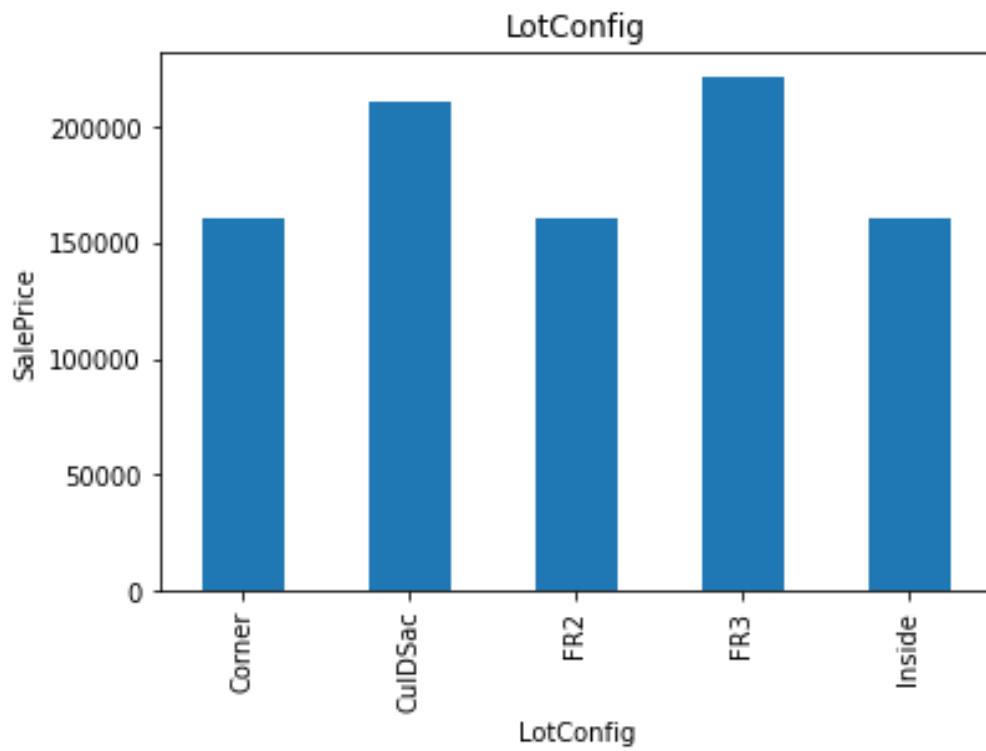
SalePrice VS KitchenQual



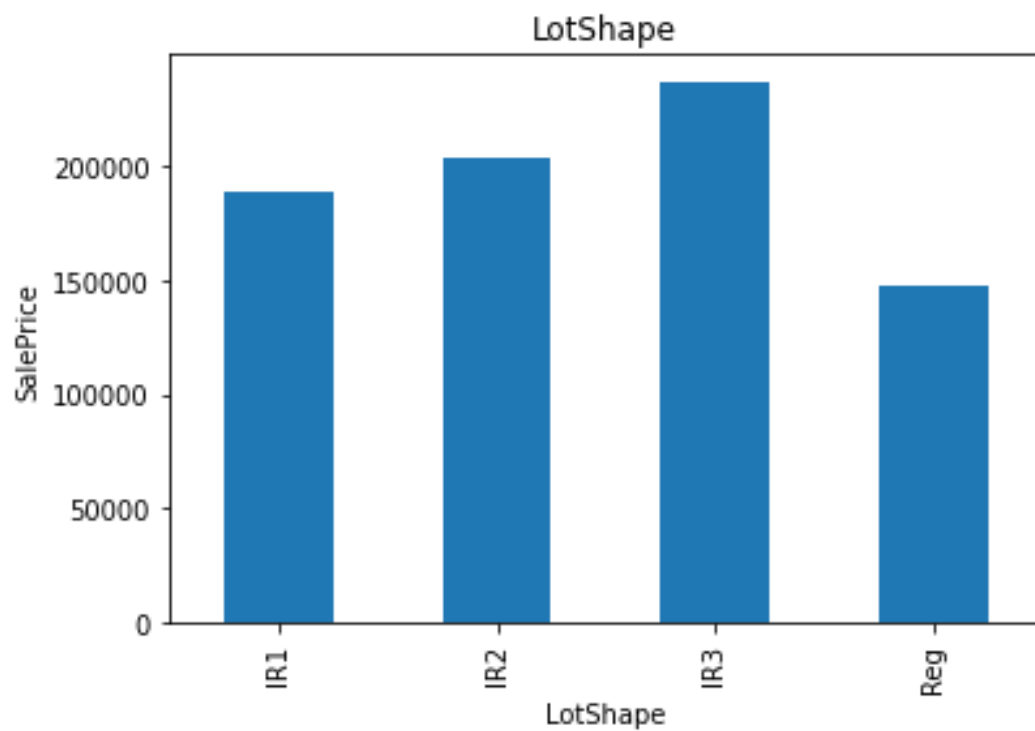
SalePrice Vs LandContour



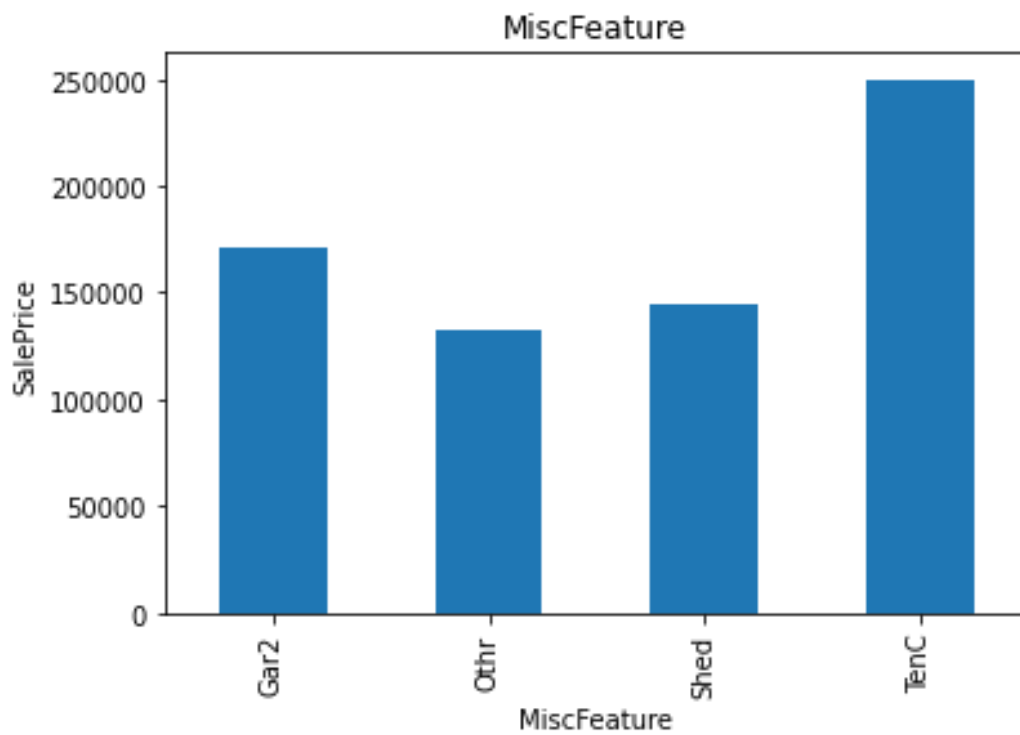
SalePrice VS LandSlope



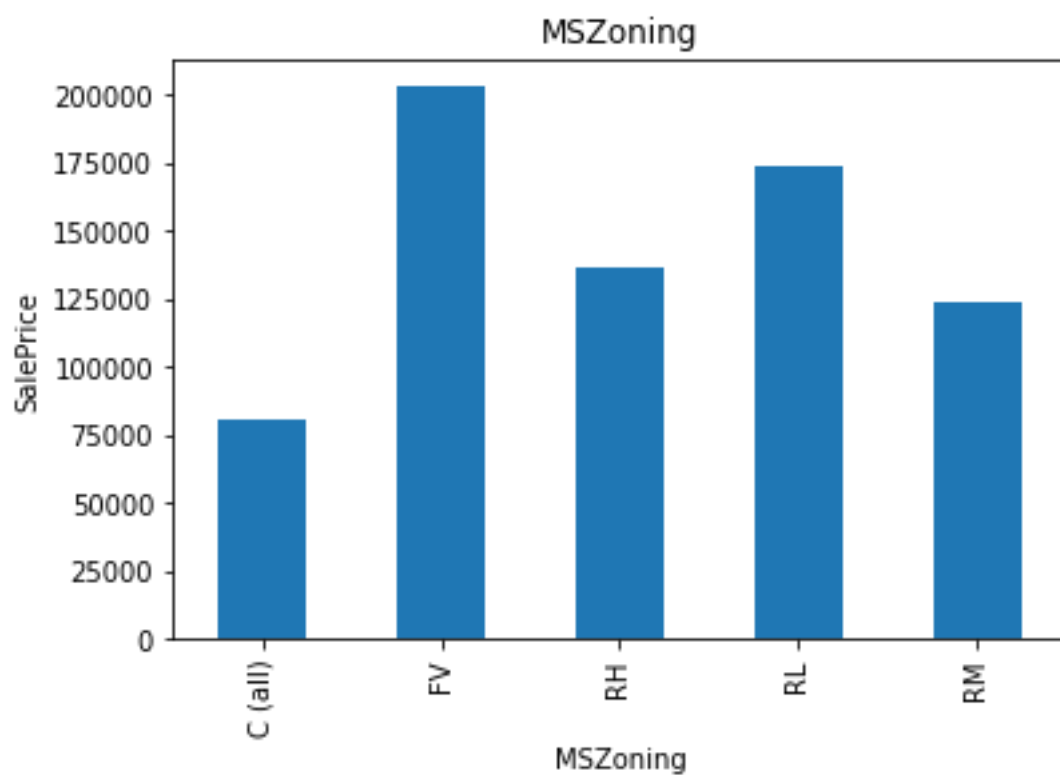
SalePrice Vs LotConfig



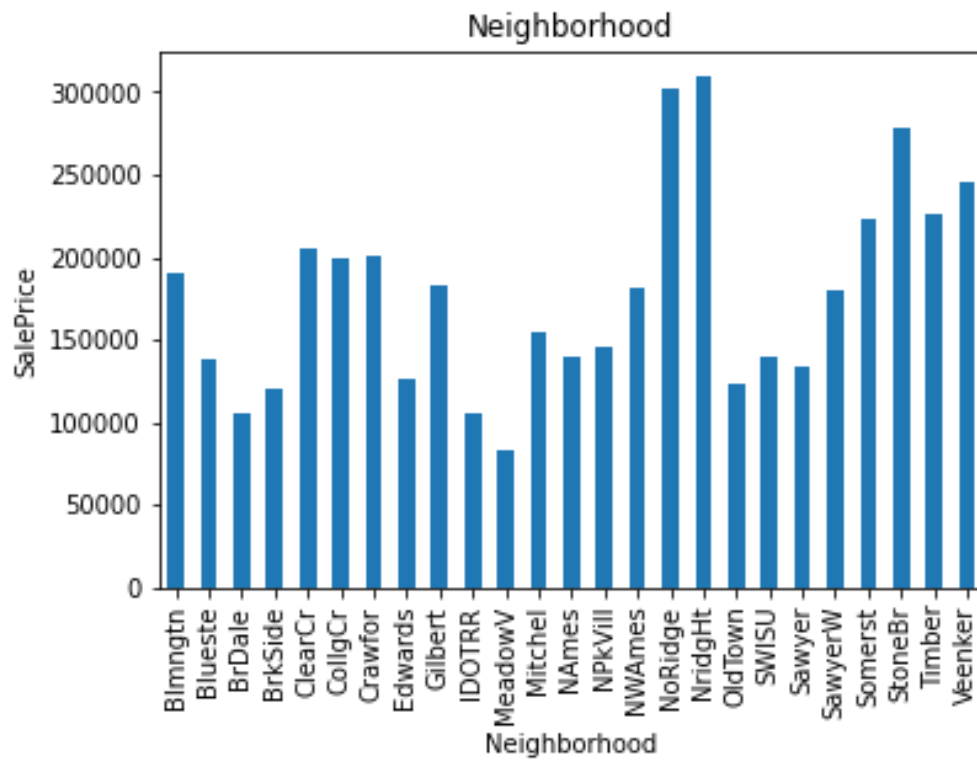
SalePrice Vs Lotshape



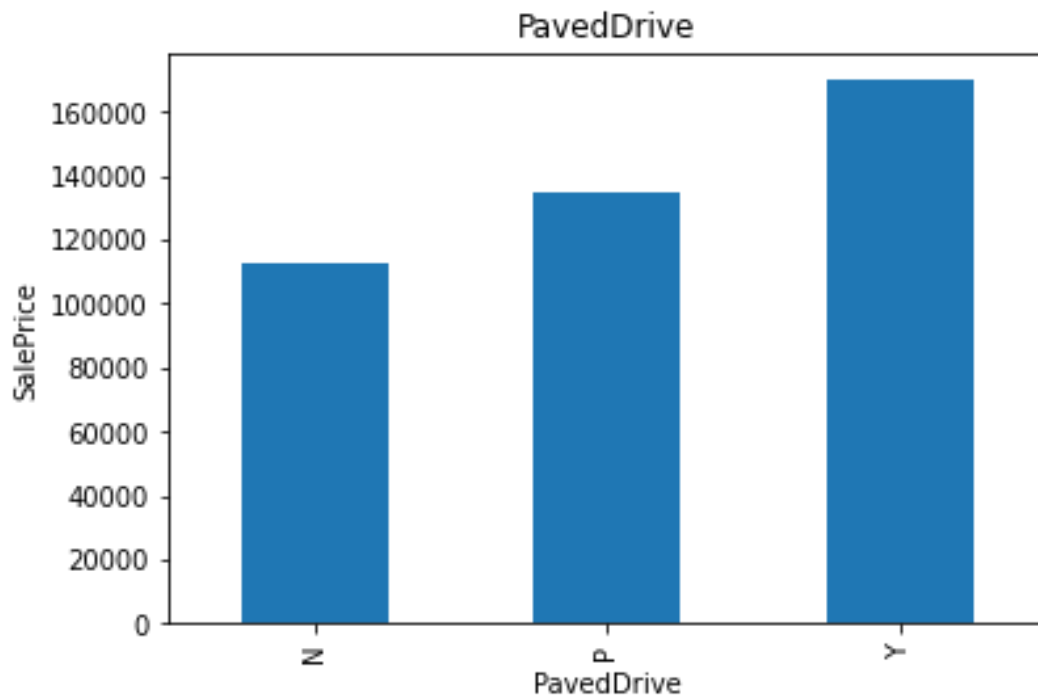
SalePrice Vs MiscFeature



SalePrice VS MSZoning

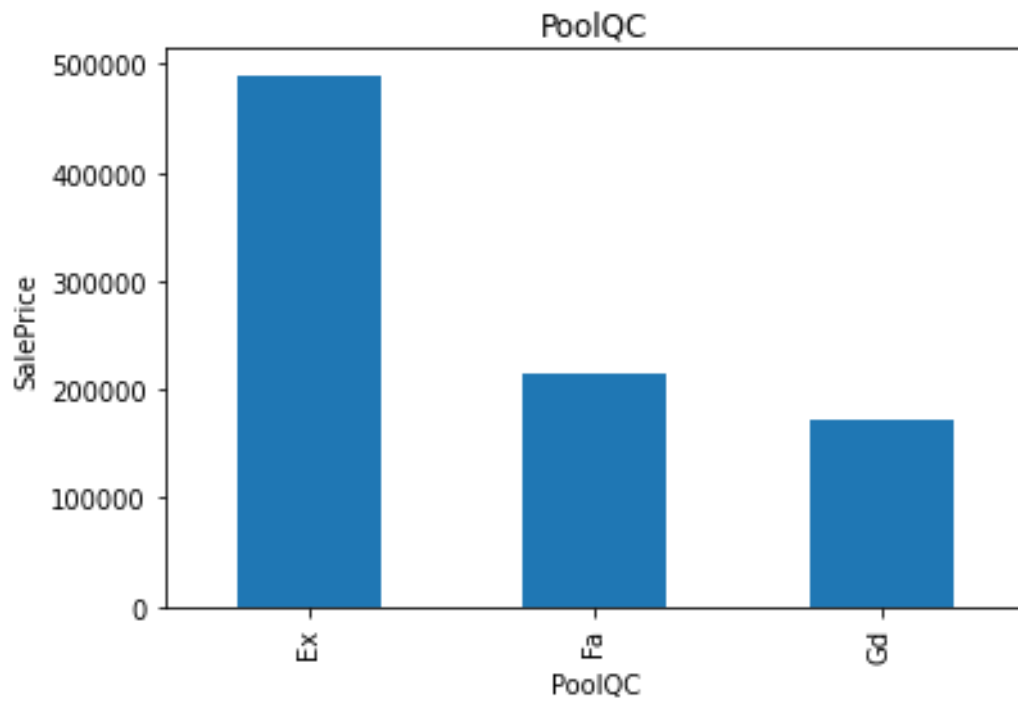


SalePrice Vs Neighborhood

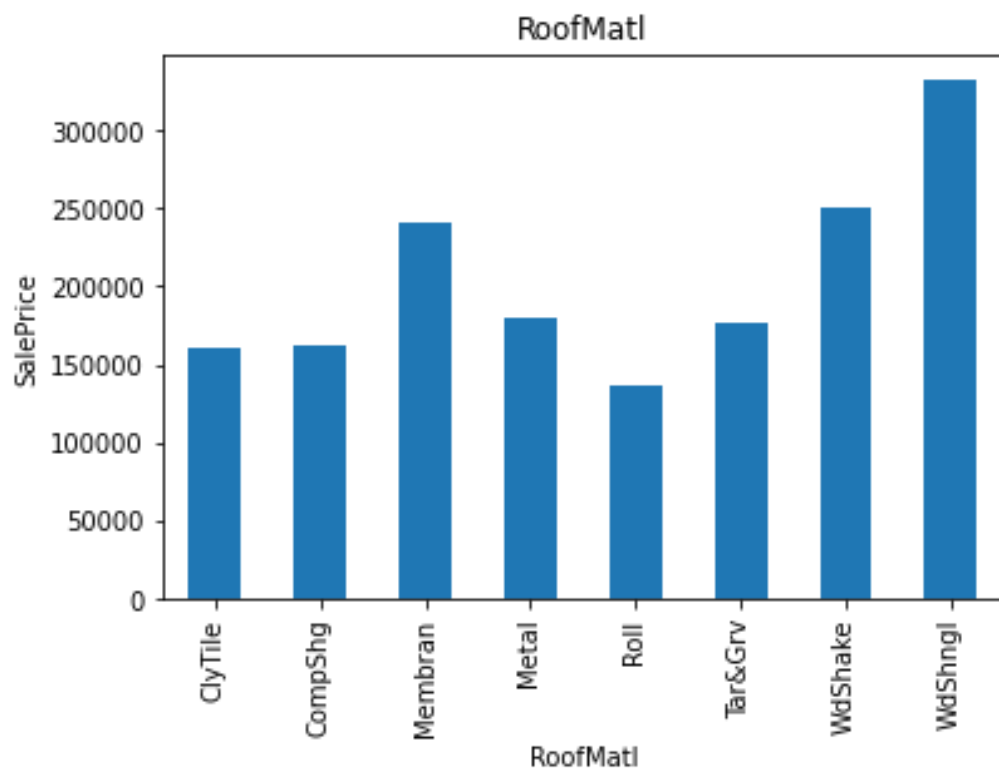


SalePrice Vs PavedDrive

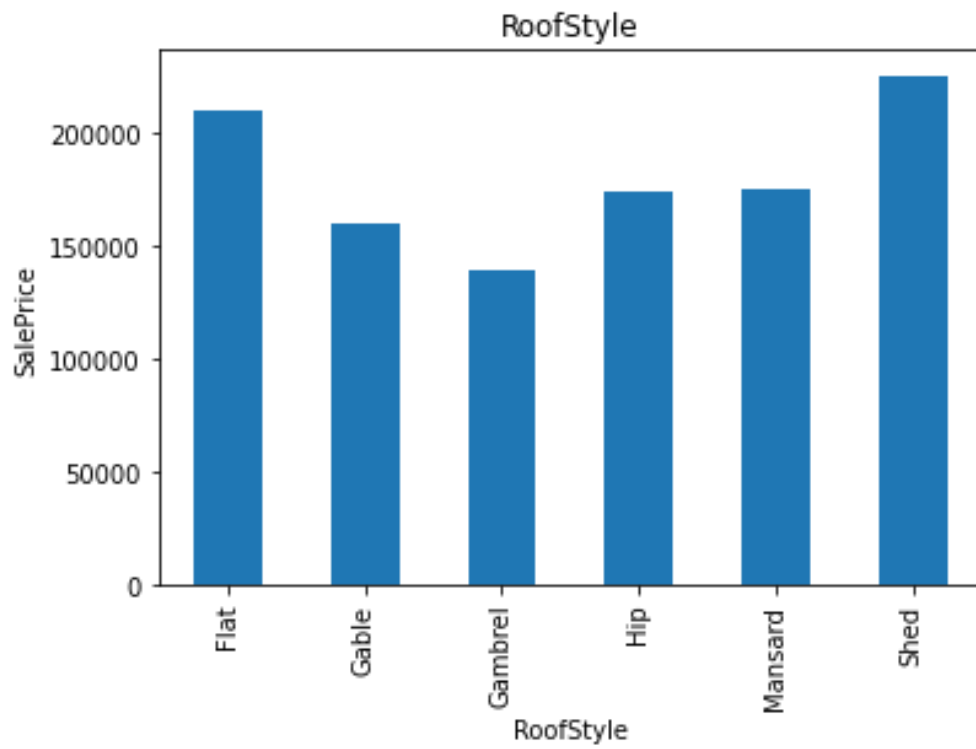




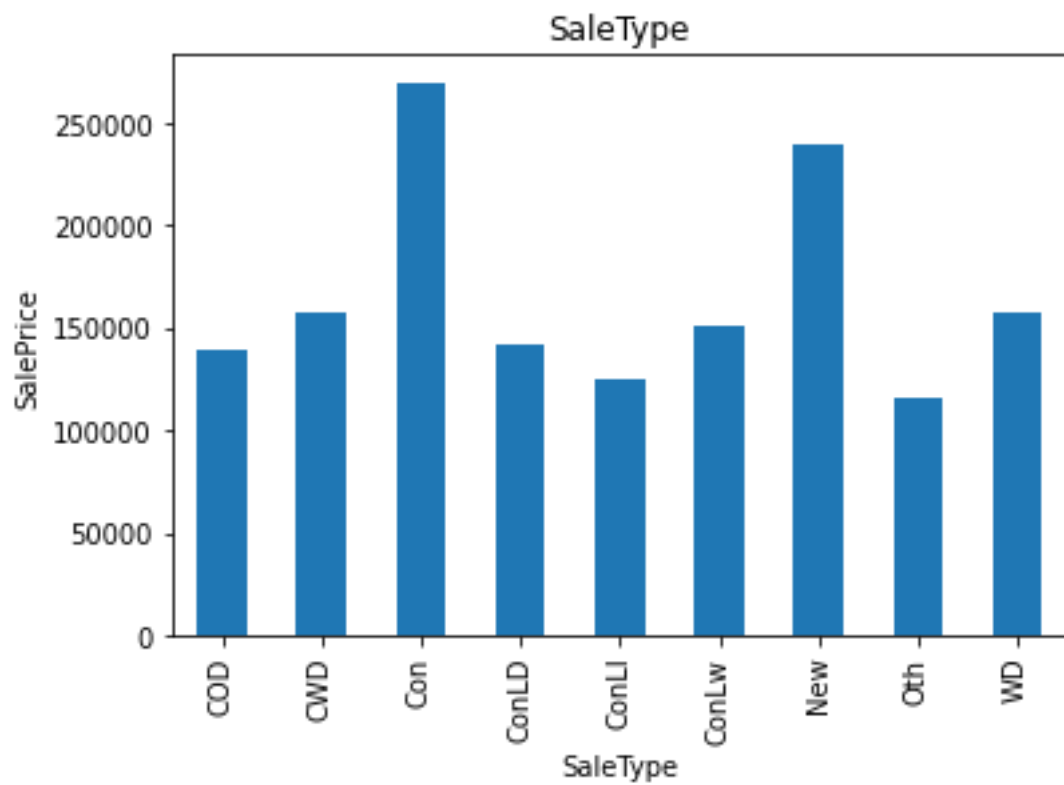
SalePrice VS PoolQC



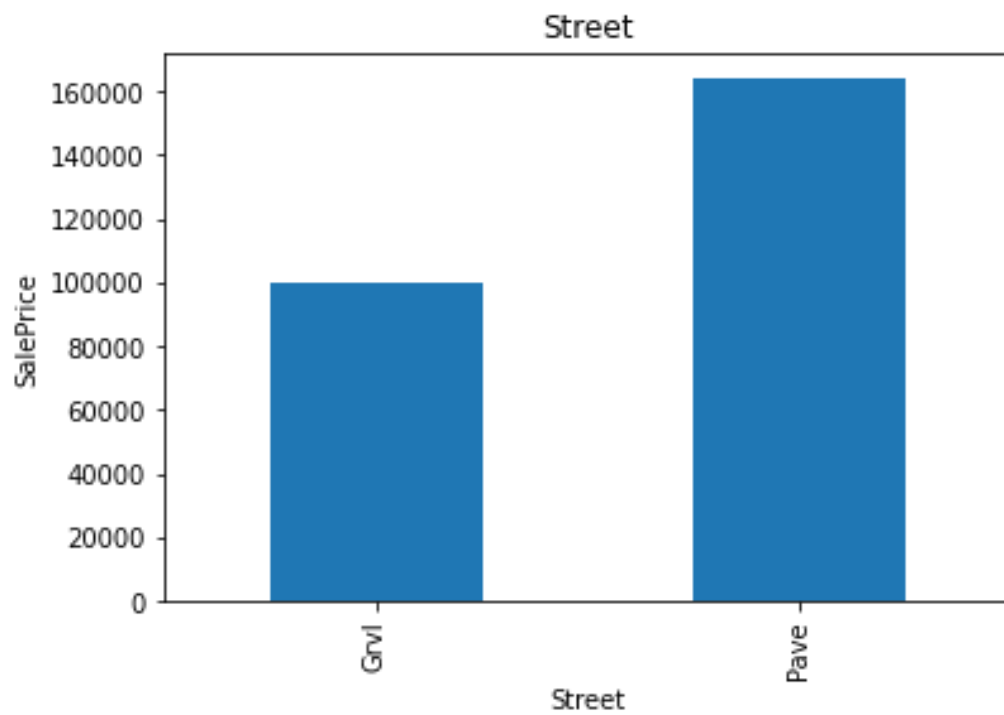
SalePrice VS RoofMati



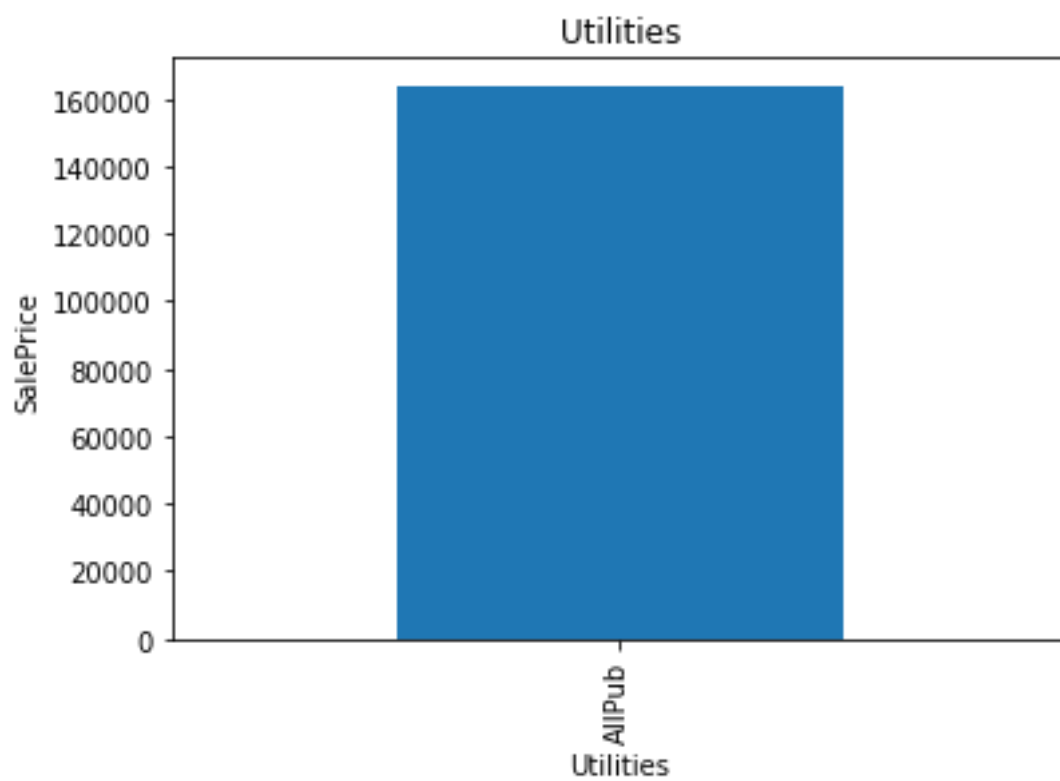
SalePrice VS RoofStyle



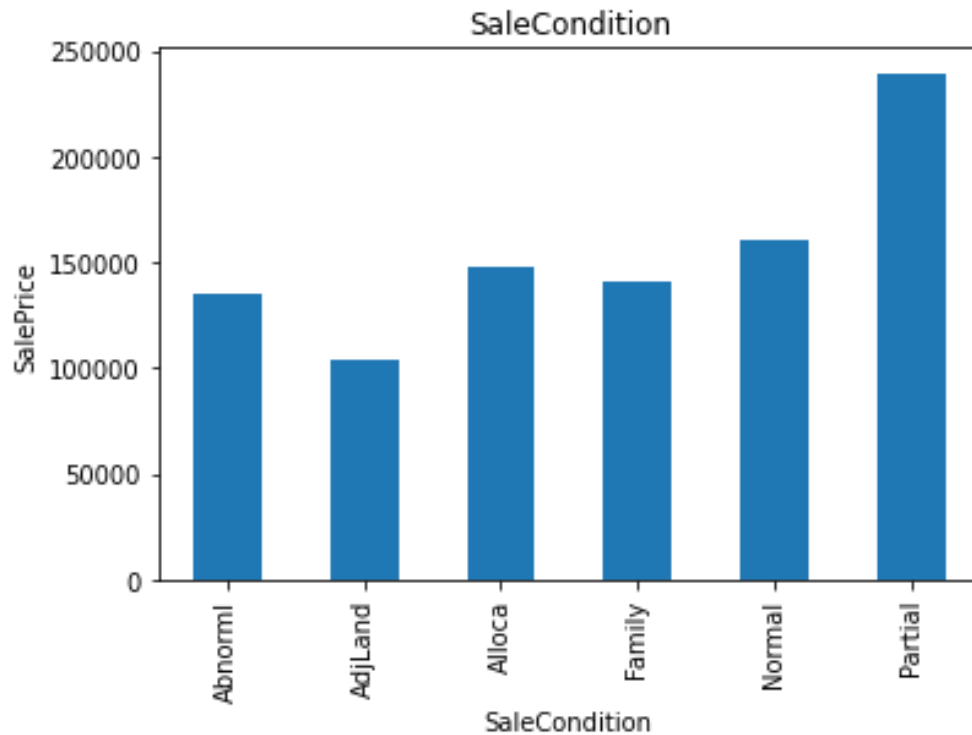
SalePrice VS SaleType



SalePrice Vs Street



SalePrice Vs utilities



SalePrice Vs SaleCondition

Missing value can be replaced by the word Missing in Feature Engineering using Python code

```
def replace_cat_feature(df, features_nan):
    data=df.copy()
    data[features_nan]=data[features_nan].fillna('Missing')
    return data

df=replace_cat_feature(df, features_nan)
df[features_nan].isnull().sum()
```

Missing values present in Numerical Variables can be replaced by the word Missing using the following python code:-

```
for feature in numerical_with_nan:
    median_value=df[feature].median()
    df[feature+'nan']=np.where(df[feature].isnull(),1,0)
    df[feature].fillna(median_value,inplace=True)
```

```
df[numerical_with_nan].isnull().sum()
```

Extracting the new Feature from Date time Variable using the following Python code:-

```
for feature in ['YearBuilt','YearRemodAdd','GarageYrBlt']:
```

```
    df[feature]=df['YrSold']-df[feature]
```

Make the logTransformation to remove the Right skewness in the histogram using the following python code:- in the features 'LotFrontage', 'LotArea', '1stFlrSF', 'GrLivArea', 'SalePrice', outliers are present

```
num_features=['LotFrontage', 'LotArea', '1stFlrSF', 'GrLivArea', 'SalePrice']
```

```
for feature in num_features:
```

```
    df[feature]=np.log(df[feature])
```

Categorical Encoding:- after outliers, skewness is removed using boxplot, log transformation, We are using Label Encoder to label from categorical to numerical using the following code:-

```
from sklearn.preprocessing import LabelEncoder
```

```
labelencoder=LabelEncoder()
```

```
for feature in categorical_features:
```

```
    df[feature]=labelencoder.fit_transform(df[feature])
```

Similarly the Missing data can be handled in test data using the following python code:-

```
##replce missing value with new value
```

```
def replace_cat_feature_test(df1, features_nan_test):
```

```
    data=df1.copy()
```

```
    data[features_nan_test]=data[features_nan_test].fillna('Missing')
```

```
    return data
```

```
df1=replace_cat_feature_test(df1, features_nan_test)
```

```
df1[features_nan_test].isnull().sum()
```

Missing value present in the test data can be removed using the python code:-

for feature in numerical\_with\_nan\_test:

```
median_value=df1[feature].median()
df1[feature+'nan']=np.where(df1[feature].isnull(),1,0)
df1[feature].fillna(median_value,inplace=True)
```

```
df1[numerical_with_nan_test].isnull().sum()
```

Similarly as in the train data, Extract the Date Time Variable using following the python code:-

```
## Date Time Variables
```

for feature in ['YearBuilt','YearRemodAdd','GarageYrBlt']:

```
df1[feature]=df1['YrSold']-df1[feature]
```

Feature 'LotFrontage', 'LotArea', '1stFlrSF', 'GrLivArea' have the missing value. It can be handled by using log transformation

```
num_features_test=['LotFrontage', 'LotArea', '1stFlrSF', 'GrLivArea']
```

for feature in num\_features\_test:

```
df1[feature]=np.log(df1[feature])
```

similarly after removing the skewness, we are using the LabelEncoder to convert categorical to numerical using the following python code:-

for feature in categorical\_features\_test:

```
df1[feature]=labelencoder.fit_transform(df1[feature])
```

Feature Scaling:-

We are using the Min Max scaler for Scaling purpose:-

```
from sklearn.preprocessing import MinMaxScaler
```

```
scaler=MinMaxScaler()
```

after applying the MinMax Scaler, we are dividing the train and test data using the follow python code:-

```
y_train=df[['SalePrice']]
```

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

Regression Techniques used:-

1.Linear Regression

2.Lasso Regression

3.Ridge Regression

4.Decision Tree Regression

5.Random Forest Regression

## **Conclusion**

Lasso regression model is considered as the best model among 5 because of less error 0.20 followed by ridge (0.22)