

Building Machine Learning Pipelines: Data Analysis Phase

In this project we will focus on creating Machine Learning Pipelines considering all the life cycle of a Data Science Projects.

Project Name: House Prices: Advanced Regression Techniques

The main aim of this project is to predict the house price based on various features which we will discuss as we go ahead

All the Lifecycle In A Data Science Projects

1.Data Analysis

2.Feature Engineering

3.Feature Selection

4.Model Building

5.Model Deployment

```
In [92]: # Data Analysis Phase

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
## Display all the columns of the dataframe

pd.pandas.set_option('display.max_columns',None)
```

```
In [93]: dataset = pd.read_csv(r'C:\Users\sudha\Python Projects\House prices EDA\train.csv')

##print shape of the dataset with rows and columns
print(dataset.shape)

(1460, 81)
```

Data fields

Here's a brief version of what you'll find in the data description file.

- SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict.
- MSSubClass: The building class
- MSZoning: The general zoning classification
- LotFrontage: Linear feet of street connected to property

- LotArea: Lot size in square feet
- Street: Type of road access
- Alley: Type of alley access
- LotShape: General shape of property
- LandContour: Flatness of the property
- Utilities: Type of utilities available
- LotConfig: Lot configuration
- LandSlope: Slope of property
- Neighborhood: Physical locations within Ames city limits
- Condition1: Proximity to main road or railroad
- Condition2: Proximity to main road or railroad (if a second is present)
- BldgType: Type of dwelling
- HouseStyle: Style of dwelling
- OverallQual: Overall material and finish quality
- OverallCond: Overall condition rating
- YearBuilt: Original construction date
- YearRemodAdd: Remodel date
- RoofStyle: Type of roof
- RoofMatl: Roof material
- Exterior1st: Exterior covering on house
- Exterior2nd: Exterior covering on house (if more than one material)
- MasVnrType: Masonry veneer type
- MasVnrArea: Masonry veneer area in square feet
- ExterQual: Exterior material quality
- ExterCond: Present condition of the material on the exterior
- Foundation: Type of foundation
- BsmtQual: Height of the basement
- BsmtCond: General condition of the basement
- BsmtExposure: Walkout or garden level basement walls
- BsmtFinType1: Quality of basement finished area
- BsmtFinSF1: Type 1 finished square feet
- BsmtFinType2: Quality of second finished area (if present)
- BsmtFinSF2: Type 2 finished square feet
- BsmtUnfSF: Unfinished square feet of basement area
- TotalBsmtSF: Total square feet of basement area
- Heating: Type of heating
- HeatingQC: Heating quality and condition
- CentralAir: Central air conditioning
- Electrical: Electrical system
- 1stFlrSF: First Floor square feet
- 2ndFlrSF: Second floor square feet
- LowQualFinSF: Low quality finished square feet (all floors)
- GrLivArea: Above grade (ground) living area square feet
- BsmtFullBath: Basement full bathrooms
- BsmtHalfBath: Basement half bathrooms
- FullBath: Full bathrooms above grade
- HalfBath: Half baths above grade
- Bedroom: Number of bedrooms above basement level

- Kitchen: Number of kitchens
- KitchenQual: Kitchen quality
- TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
- Functional: Home functionality rating
- Fireplaces: Number of fireplaces
- FireplaceQu: Fireplace quality
- GarageType: Garage location
- GarageYrBlt: Year garage was built
- GarageFinish: Interior finish of the garage
- GarageCars: Size of garage in car capacity
- GarageArea: Size of garage in square feet
- GarageQual: Garage quality
- GarageCond: Garage condition
- PavedDrive: Paved driveway
- WoodDeckSF: Wood deck area in square feet
- OpenPorchSF: Open porch area in square feet
- EnclosedPorch: Enclosed porch area in square feet
- 3SsnPorch: Three season porch area in square feet
- ScreenPorch: Screen porch area in square feet
- PoolArea: Pool area in square feet
- PoolQC: Pool quality
- Fence: Fence quality
- MiscFeature: Miscellaneous feature not covered in other categories
- MiscVal: \$Value of miscellaneous feature
- MoSold: Month Sold
- YrSold: Year Sold
- SaleType: Type of sale
- SaleCondition: Condition of sale

In [94]: *#PRINT TOP 5 records*

```
dataset.head()
```

Out[94]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	Corner
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	FR2

In Data Analysis We will Analyze To Find out the below stuff

1. Missing Values
2. All The Numerical Variables
3. Distribution of the Numerical Variables
4. Categorical Variables
5. Cardinality of Categorical Variables

6. Outliers

7. Relationship between independent and dependent feature(SalePrice)

Missing Values

```
In [95]: ## 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 dataset.columns if dataset[features].isnull()]
## 2- step print the feature name and the percentage of missing values

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

LotFrontage 0.1774  % missing values
Alley 0.9377  % missing values
MasVnrType 0.0055  % missing values
MasVnrArea 0.0055  % missing values
BsmtQual 0.0253  % missing values
BsmtCond 0.0253  % missing values
BsmtExposure 0.026  % missing values
BsmtFinType1 0.0253  % missing values
BsmtFinType2 0.026  % missing values
FireplaceQu 0.4726  % missing values
GarageType 0.0555  % missing values
GarageYrBlt 0.0555  % missing values
GarageFinish 0.0555  % missing values
GarageQual 0.0555  % missing values
GarageCond 0.0555  % missing values
PoolQC 0.9952  % missing values
Fence 0.8075  % missing values
MiscFeature 0.963  % missing values
```

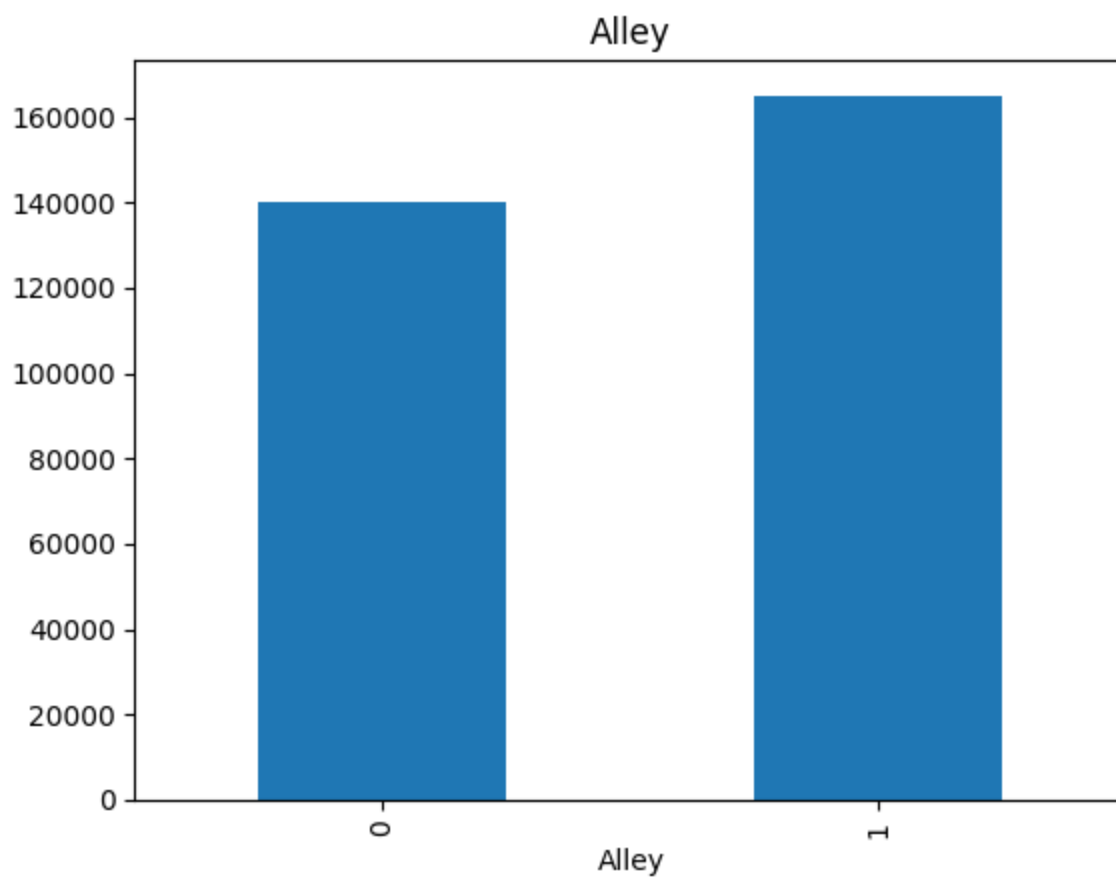
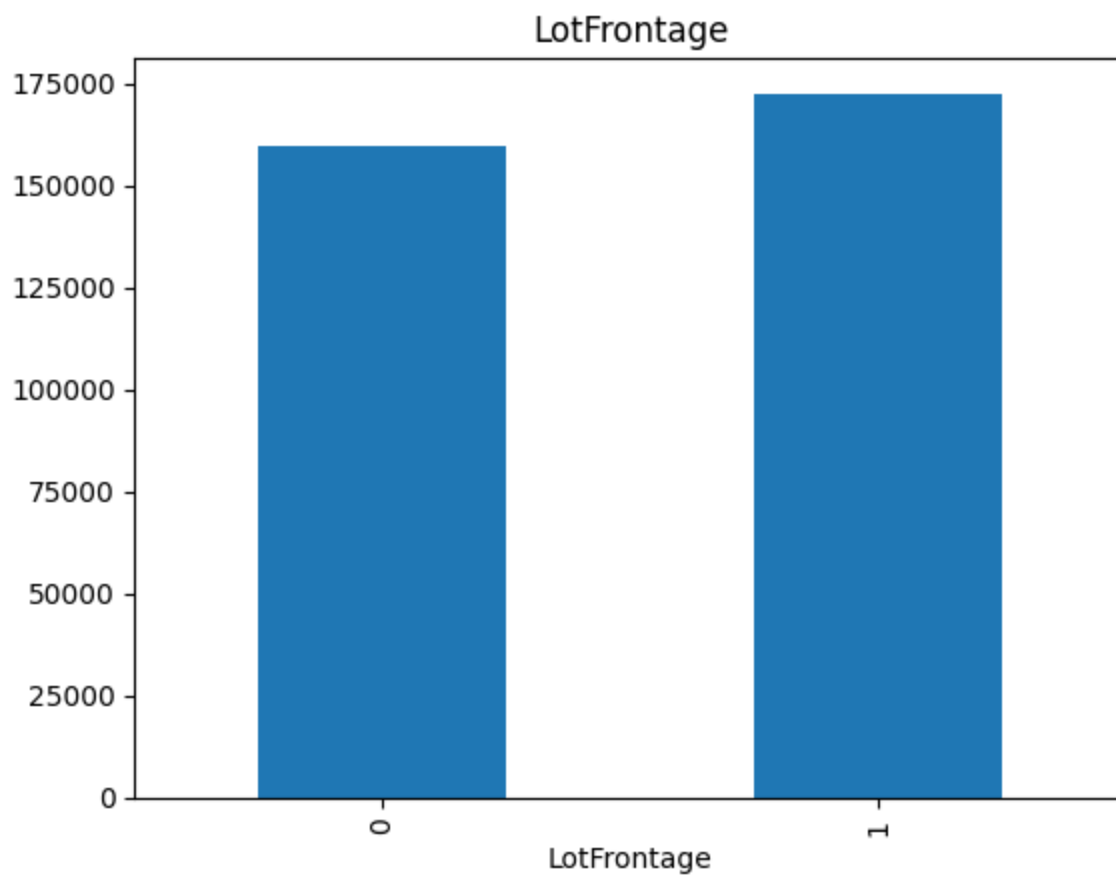
Since we have many missing values, we will find the relationship between missing values and Sales price

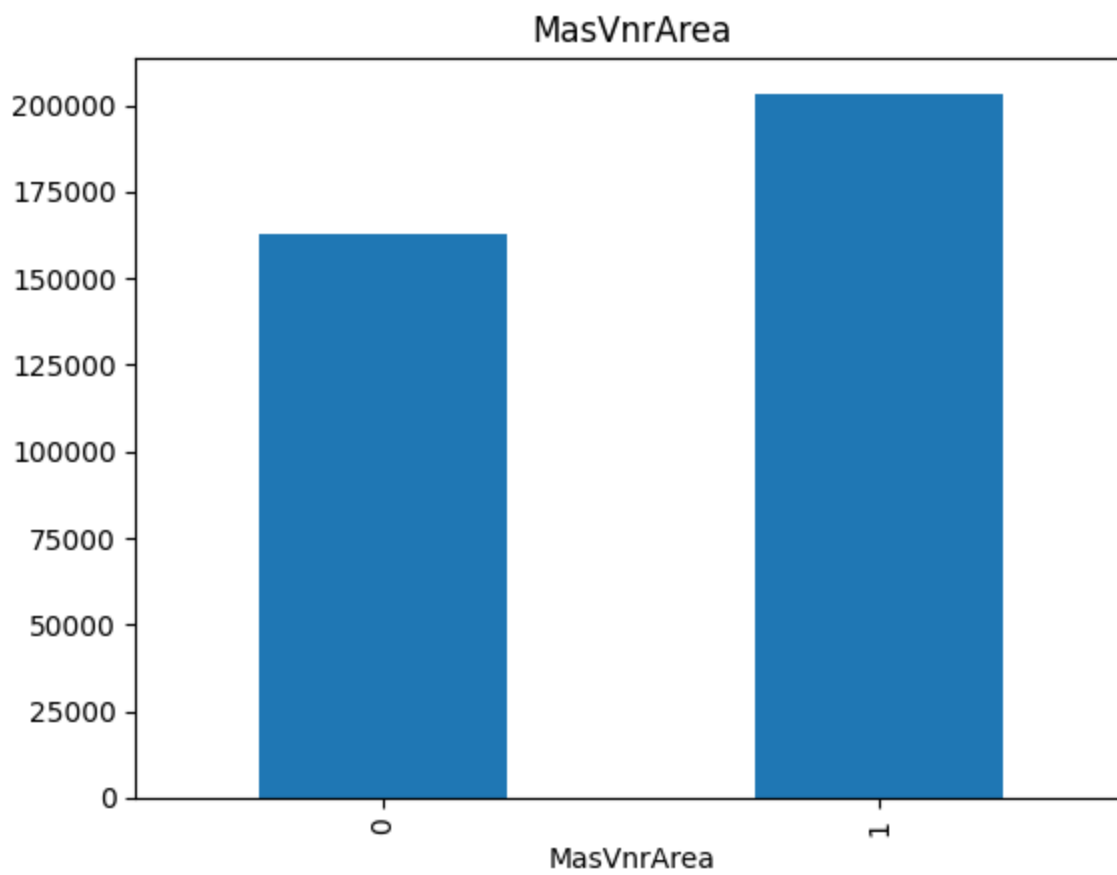
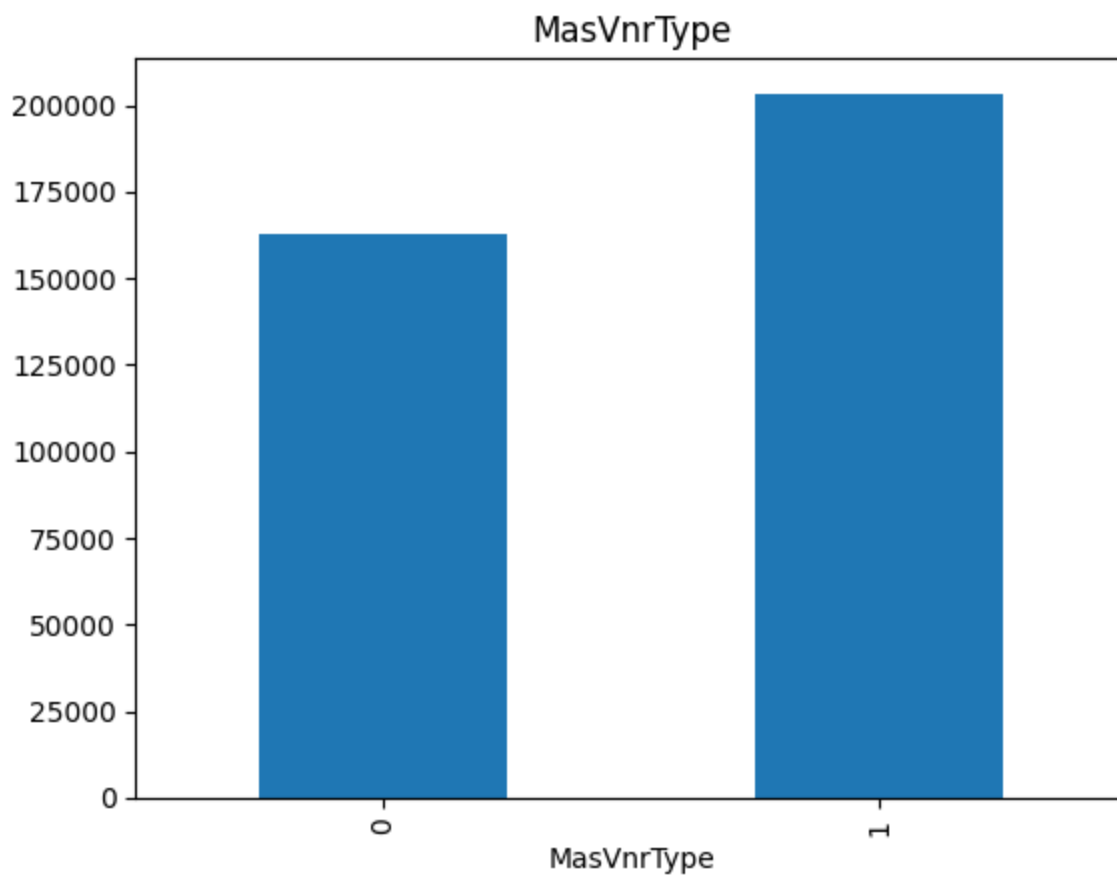
Plotting diagrams for relationships

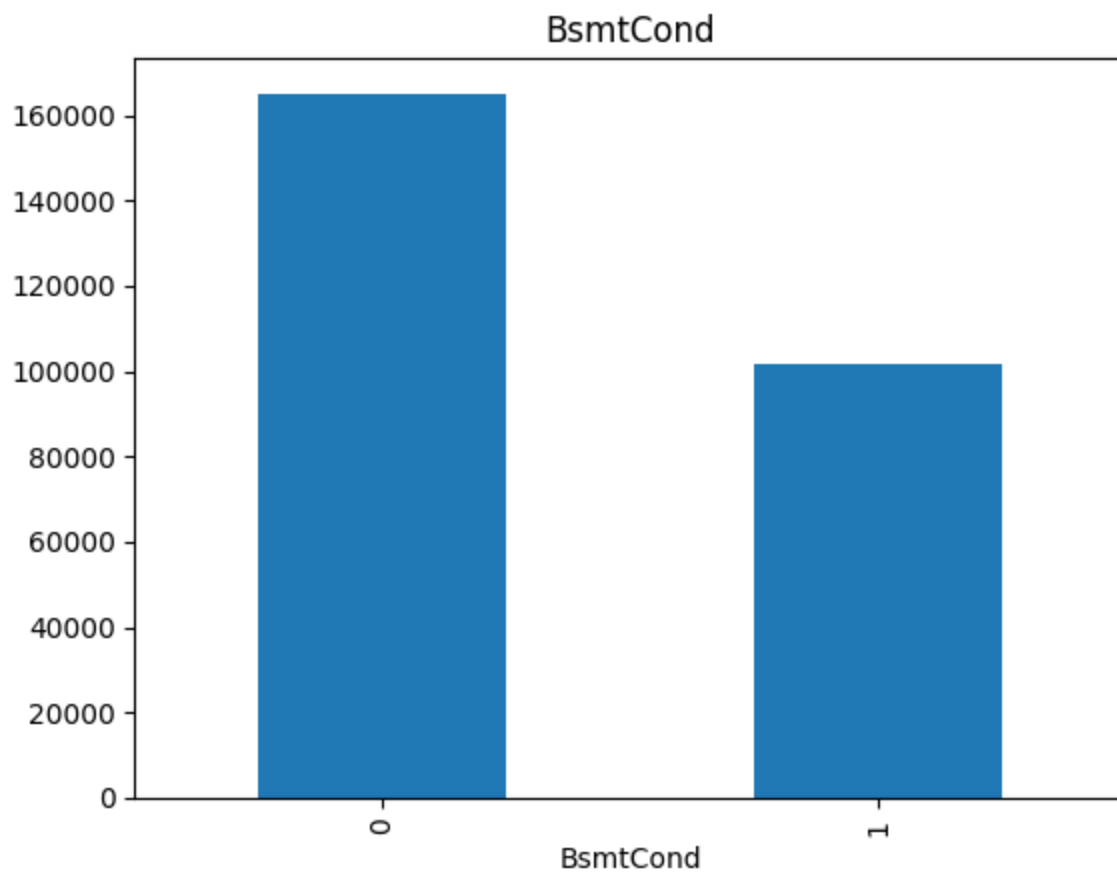
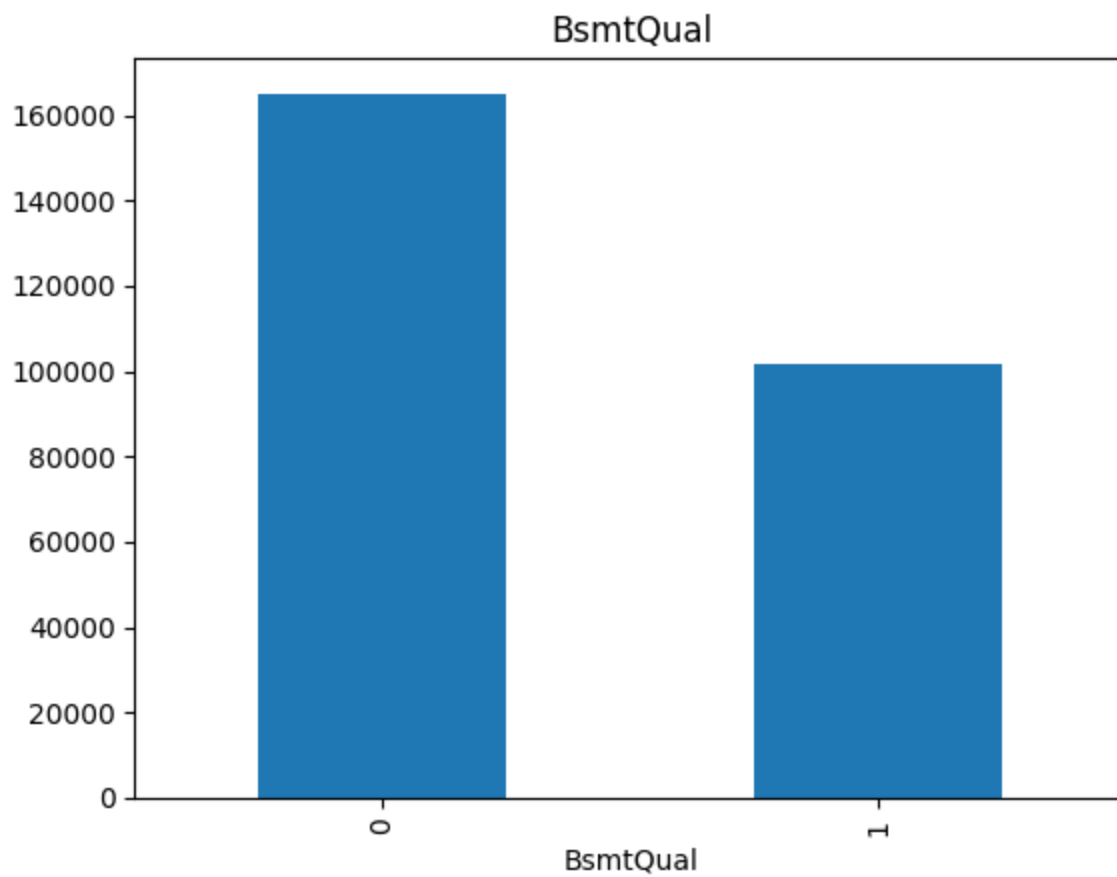
```
In [96]: for feature in features_with_na:
    data = dataset.copy()

    #making a var taht indicates 1 if observation was missing
    data[feature] = np.where(data[feature].isnull(), 1, 0)

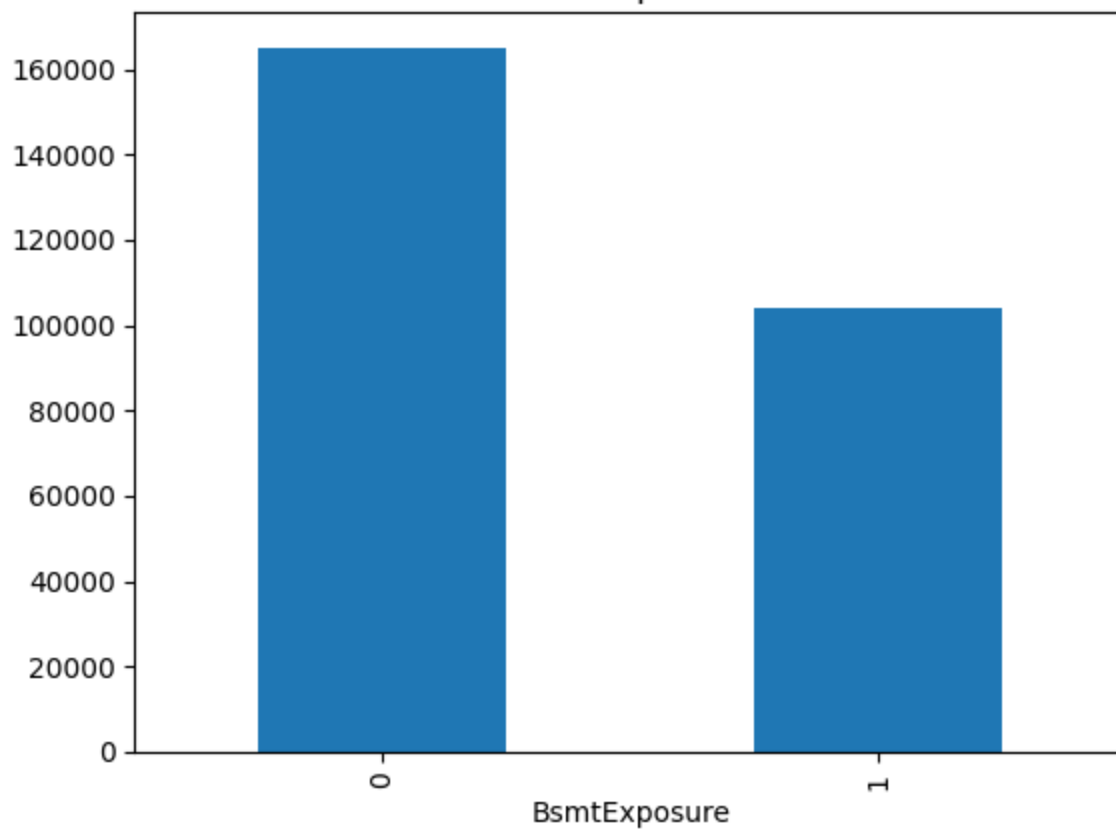
    # Caculating mean SalePrice where the information is missing or present
    data.groupby(feature) ['SalePrice'].median().plot.bar()
    plt.title(feature)
    plt.show()
```



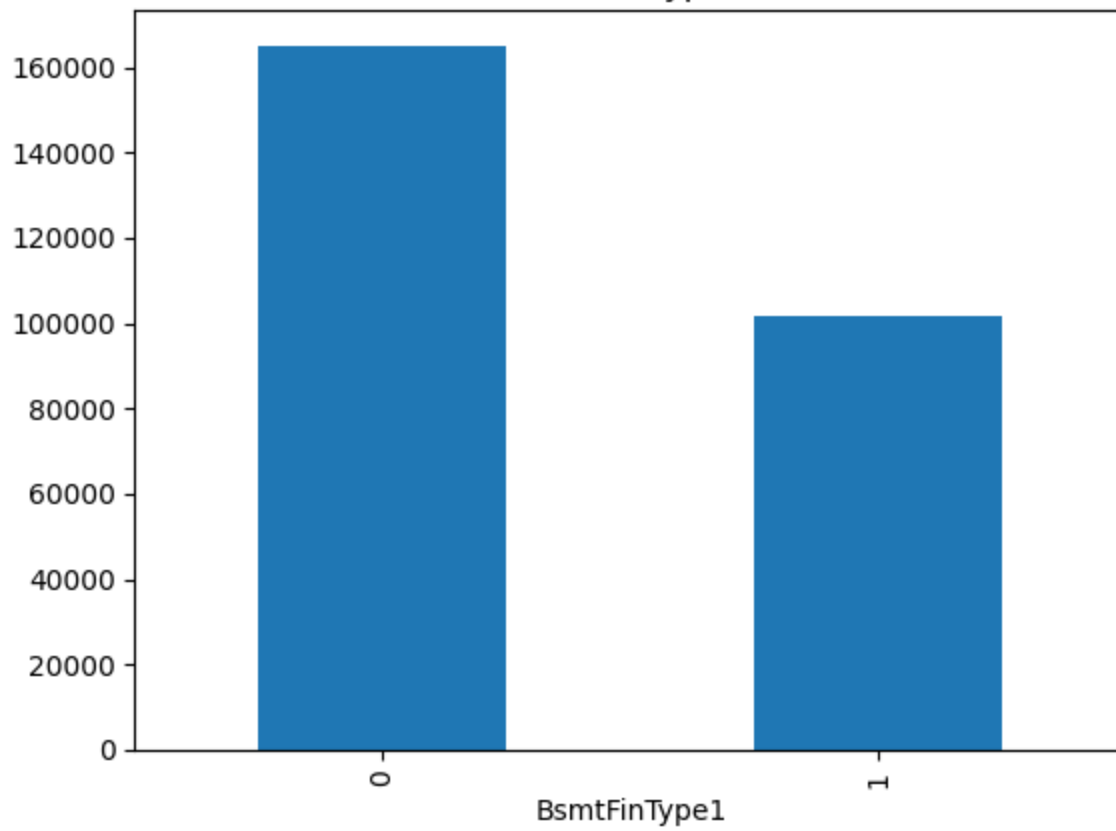




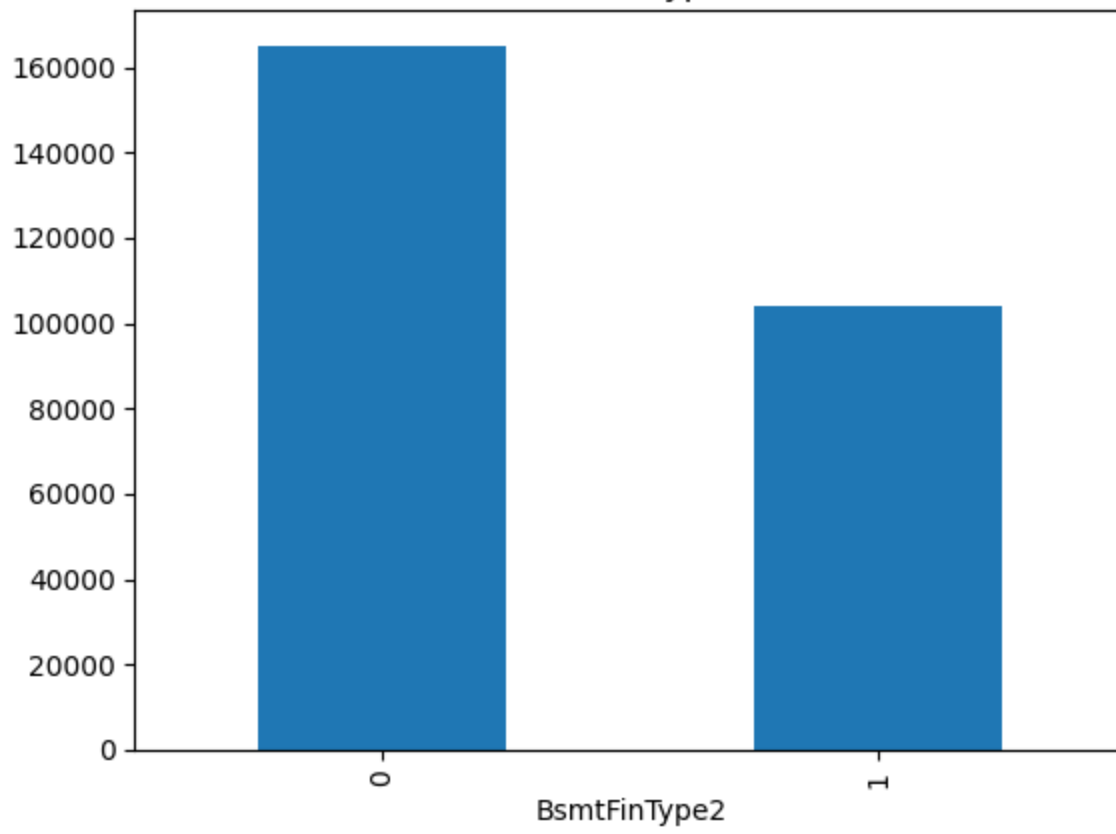
BsmtExposure



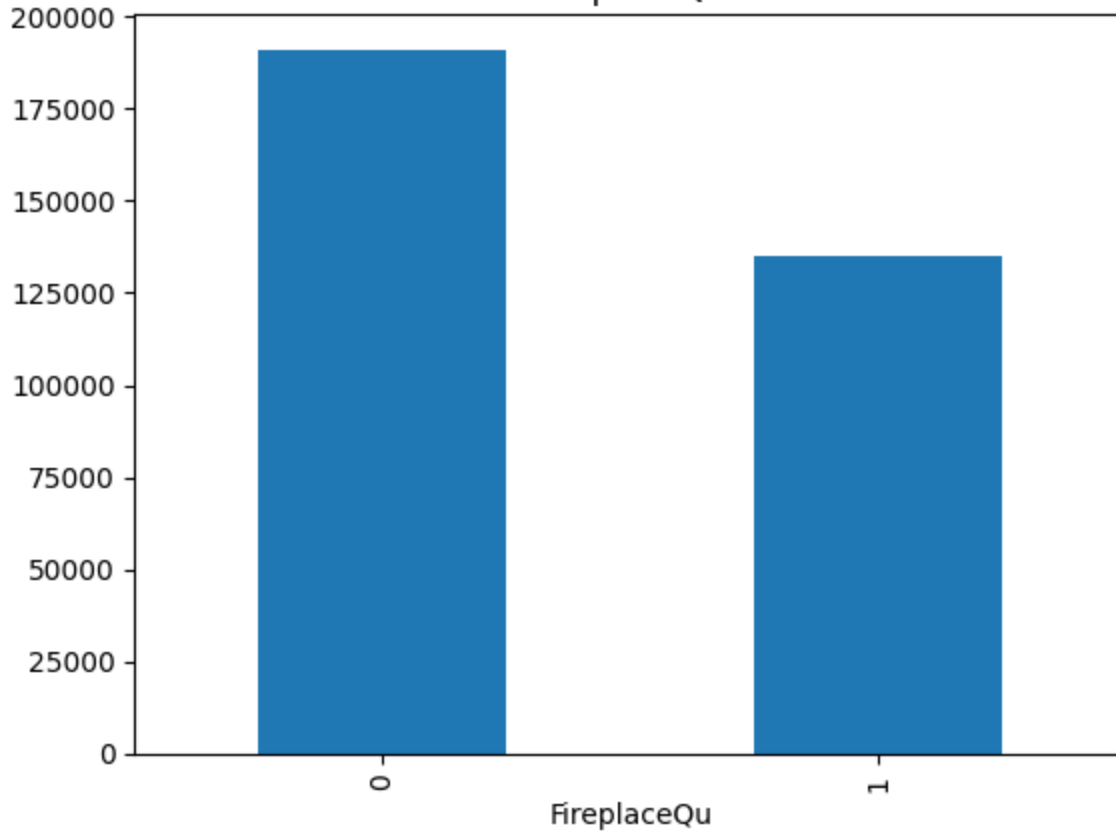
BsmtFinType1

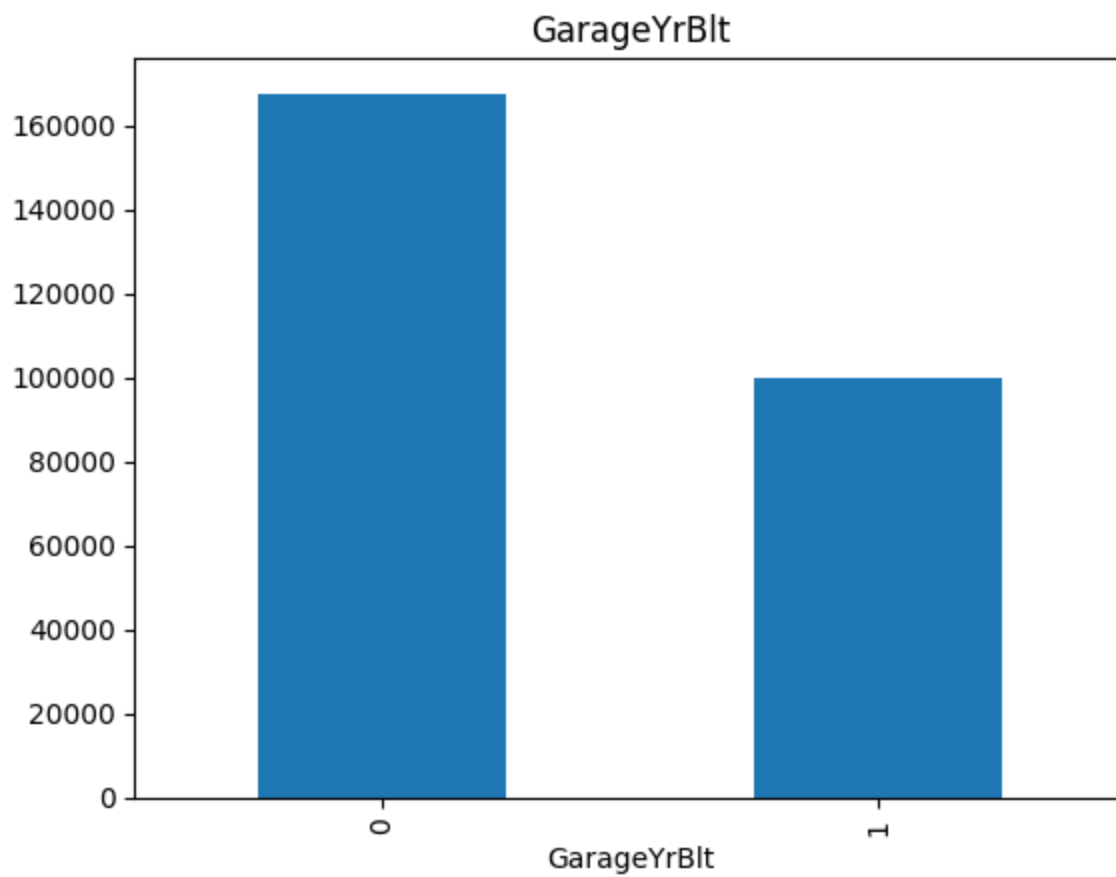
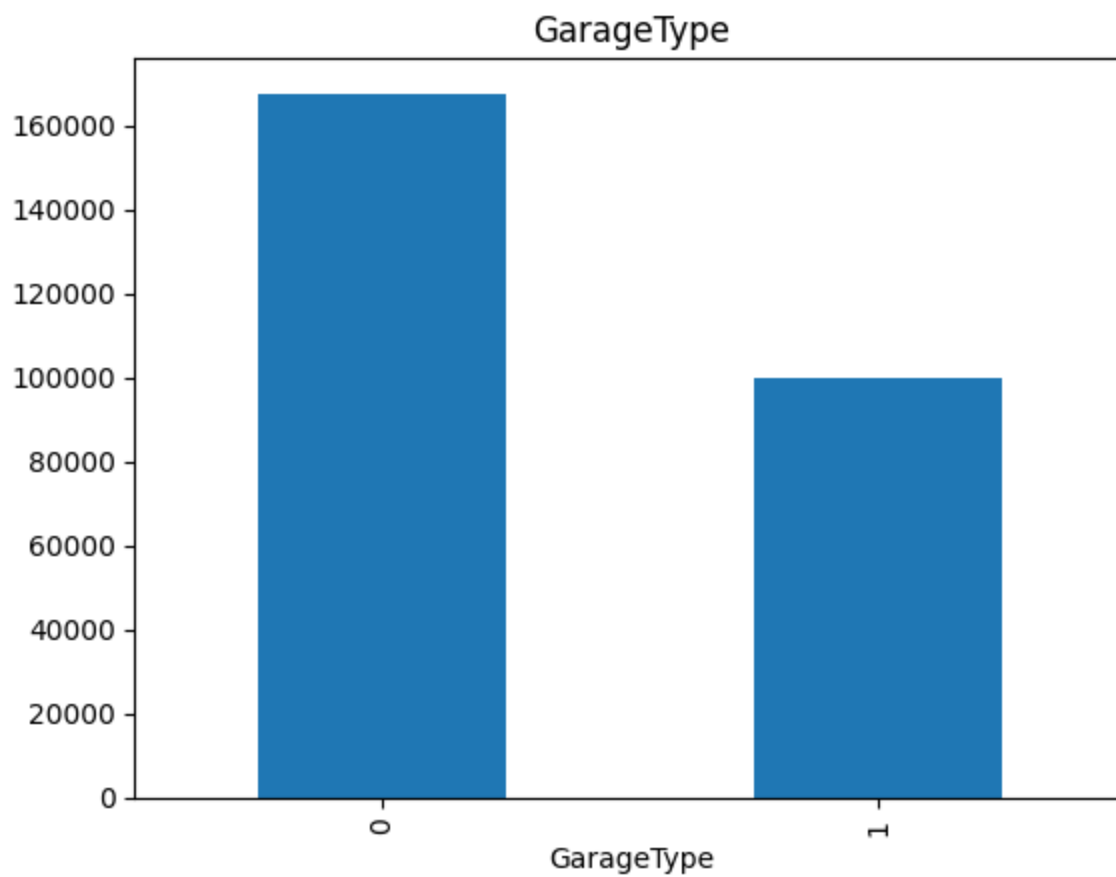


BsmtFinType2

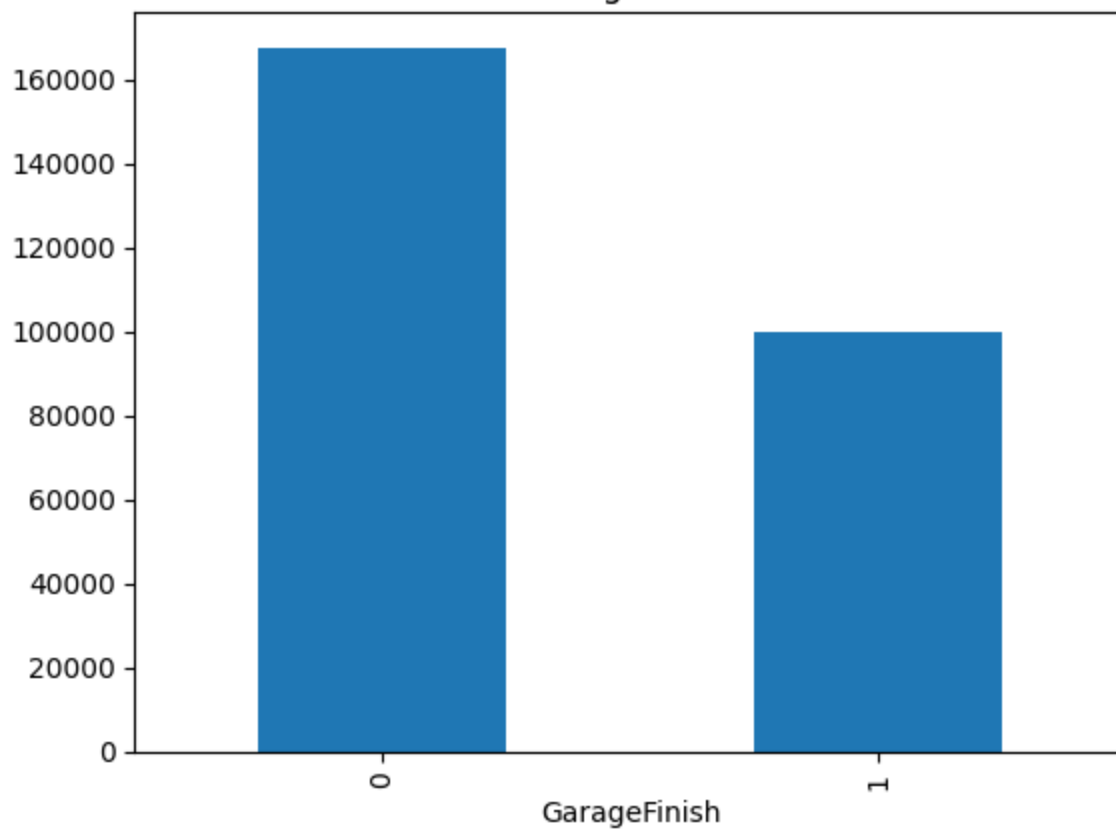


FireplaceQu

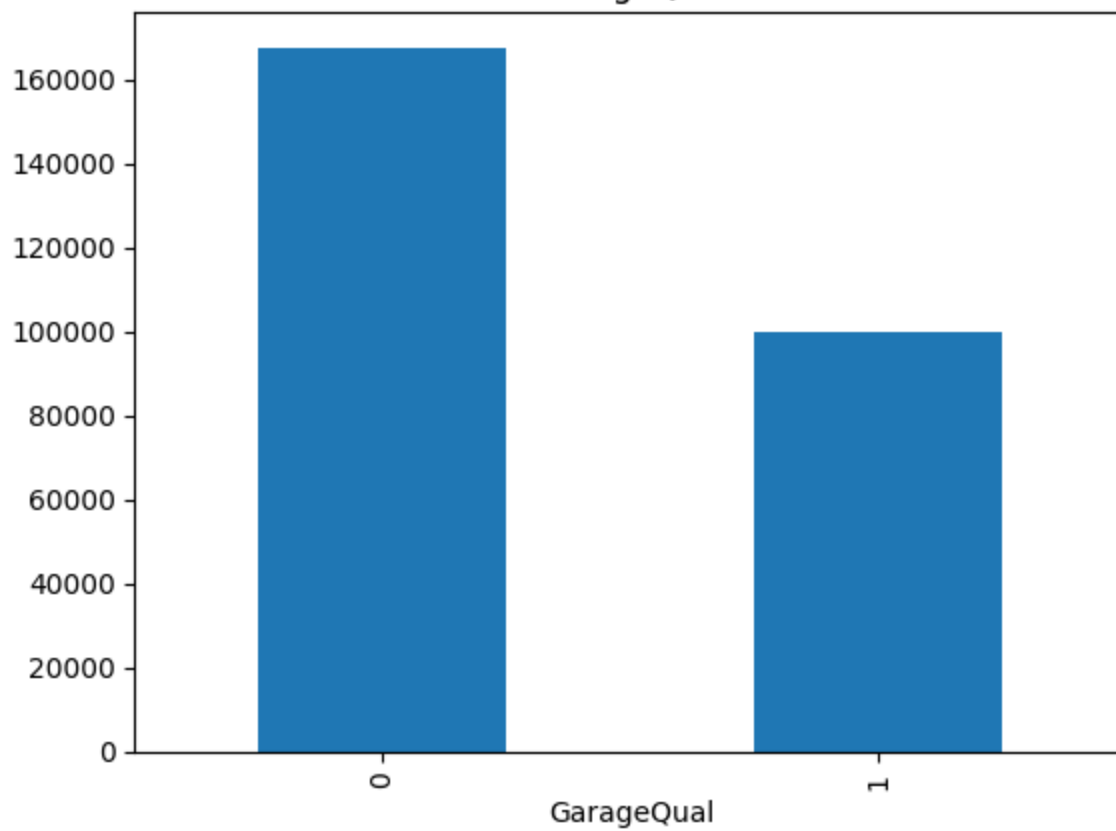




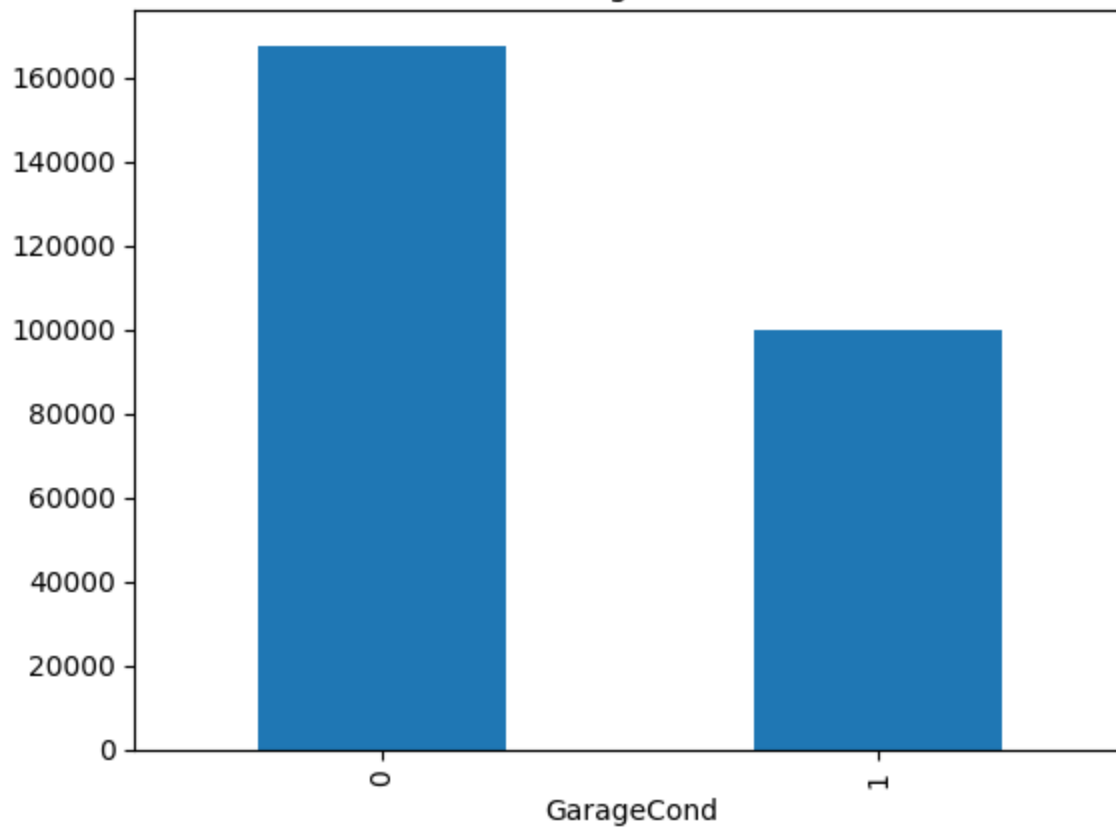
GarageFinish



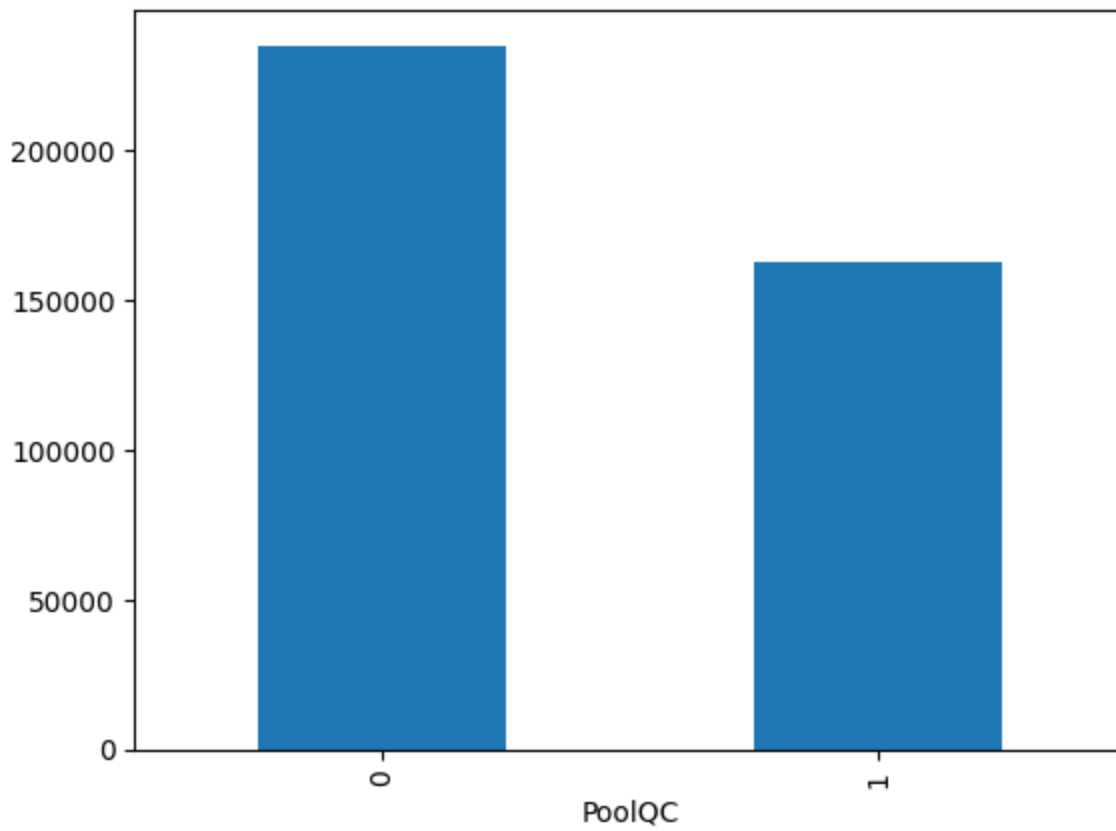
GarageQual

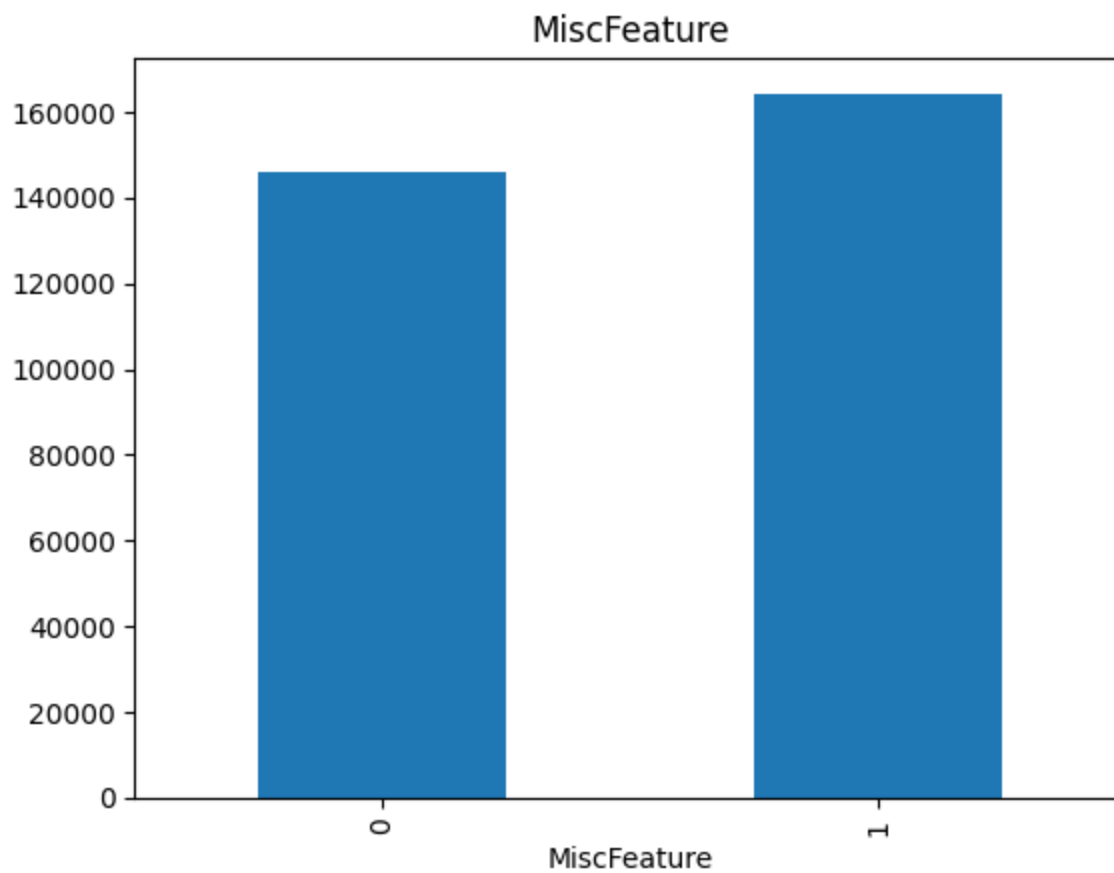
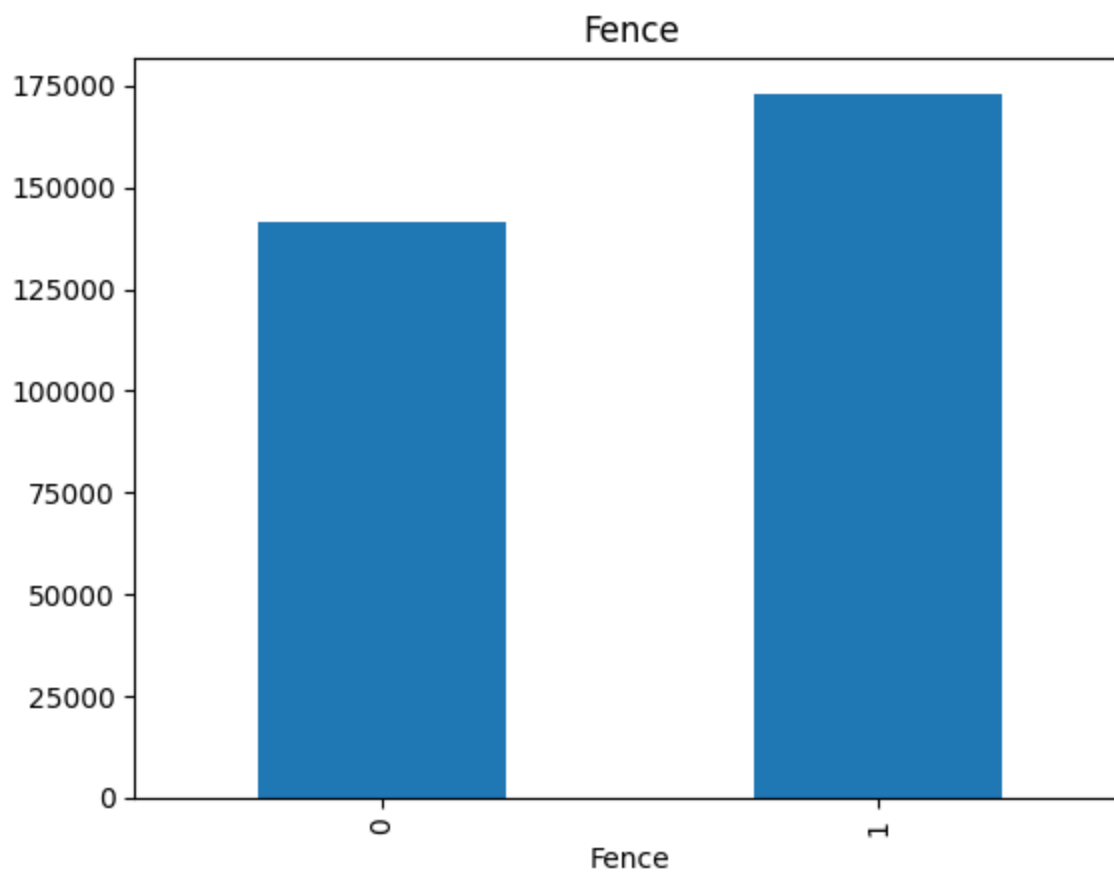


GarageCond



PoolQC





Here With the relation between the missing values and the dependent variable is clearly visible. We will replace these non values with something meaningful.

We don't need some features like ID

```
In [61]: print('ID of the houses {}'.format(len(dataset.Id)))
```

Numerical Variables

```
In [62]: #list of numerical variables

numerical_features = [feature for feature in dataset.columns if dataset[feature].dtypes

print('Number of numerical variables: ', len(numerical_features))

#visualising numerical variables
dataset[numerical_features].head()
```

Number of numerical variables: 38

```
Out[62]:
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	Bsm
0	1	60	65.0	8450	7	5	2003	2003	196.0	
1	2	20	80.0	9600	6	8	1976	1976	0.0	
2	3	60	68.0	11250	7	5	2001	2002	162.0	
3	4	70	60.0	9550	7	5	1915	1970	0.0	
4	5	60	84.0	14260	8	5	2000	2000	350.0	

Temporal Variables(i.e. Datetime variables)

```
In [63]: year_feature = [feature for feature in numerical_features if 'Yr' in feature or 'Year' i

year_feature
```

```
Out[63]: ['YearBuilt', 'YearRemodAdd', 'GarageYrBlt', 'YrSold']
```

```
In [64]: #now exploring the content of these variables

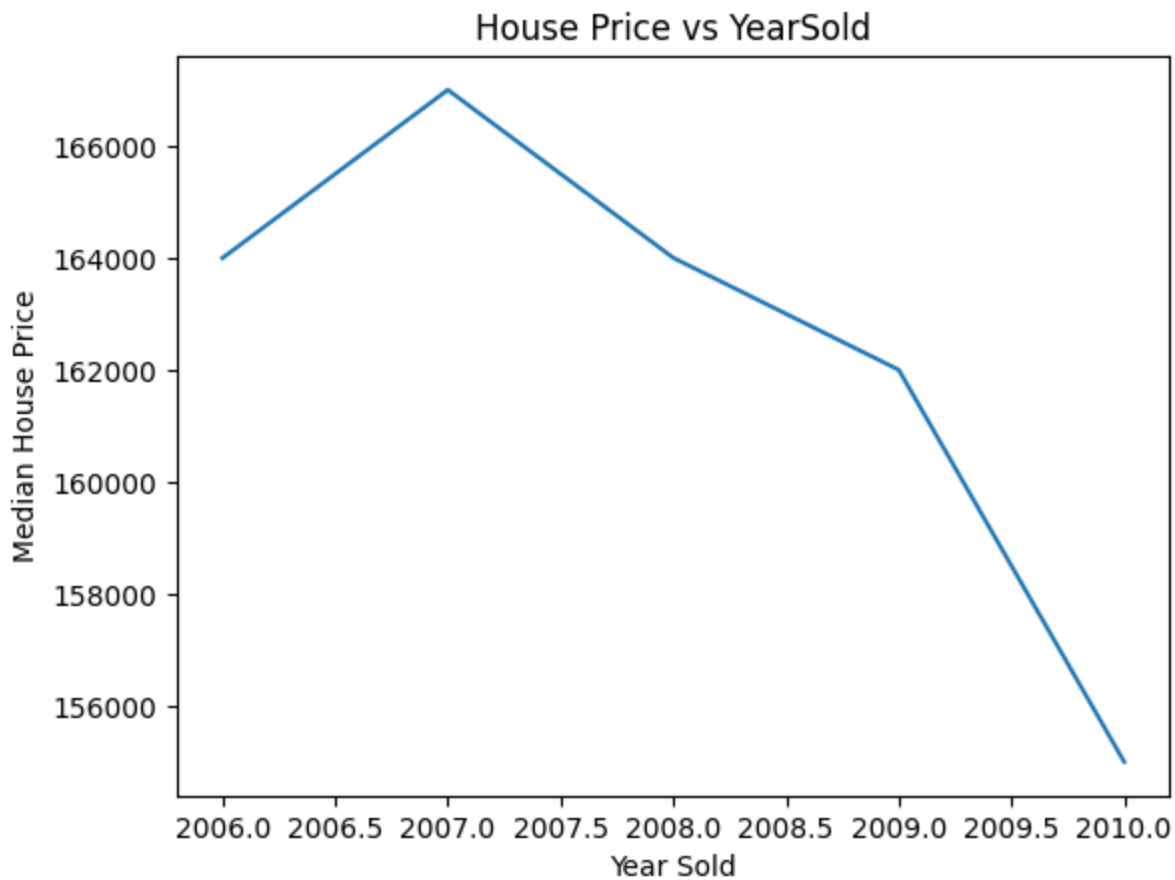
for feature in year_feature:
    print(feature, dataset[feature].unique())
```

```
YearBuilt [2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 1965 2005 1962 2006
1960 1929 1970 1967 1958 1930 2002 1968 2007 1951 1957 1927 1920 1966
1959 1994 1954 1953 1955 1983 1975 1997 1934 1963 1981 1964 1999 1972
1921 1945 1982 1998 1956 1948 1910 1995 1991 2009 1950 1961 1977 1985
1979 1885 1919 1990 1969 1935 1988 1971 1952 1936 1923 1924 1984 1926
1940 1941 1987 1986 2008 1908 1892 1916 1932 1918 1912 1947 1925 1900
1980 1989 1992 1949 1880 1928 1978 1922 1996 2010 1946 1913 1937 1942
1938 1974 1893 1914 1906 1890 1898 1904 1882 1875 1911 1917 1872 1905]
YearRemodAdd [2003 1976 2002 1970 2000 1995 2005 1973 1950 1965 2006 1962 2007 1960
2001 1967 2004 2008 1997 1959 1990 1955 1983 1980 1966 1963 1987 1964
1972 1996 1998 1989 1953 1956 1968 1981 1992 2009 1982 1961 1993 1999
1985 1979 1977 1969 1958 1991 1971 1952 1975 2010 1984 1986 1994 1988
1954 1957 1951 1978 1974]
GarageYrBlt [2003. 1976. 2001. 1998. 2000. 1993. 2004. 1973. 1931. 1939. 1965. 2005.
1962. 2006. 1960. 1991. 1970. 1967. 1958. 1930. 2002. 1968. 2007. 2008.
1957. 1920. 1966. 1959. 1995. 1954. 1953. nan 1983. 1977. 1997. 1985.
1963. 1981. 1964. 1999. 1935. 1990. 1945. 1987. 1989. 1915. 1956. 1948.
1974. 2009. 1950. 1961. 1921. 1900. 1979. 1951. 1969. 1936. 1975. 1971.
1923. 1984. 1926. 1955. 1986. 1988. 1916. 1932. 1972. 1918. 1980. 1924.
1996. 1940. 1949. 1994. 1910. 1978. 1982. 1992. 1925. 1941. 2010. 1927.
1947. 1937. 1942. 1938. 1952. 1928. 1922. 1934. 1906. 1914. 1946. 1908.
1929. 1933.]
YrSold [2008 2007 2006 2009 2010]
```

```
In [97]: #Now analyzing this temporal dataset variables  
#check whether there's a realation between year and the house is sold
```

```
dataset.groupby('YrSold')['SalePrice'].median().plot()  
plt.xlabel('Year Sold')  
plt.ylabel('Median House Price')  
plt.title("House Price vs YearSold")
```

```
Out[97]: Text(0.5, 1.0, 'House Price vs YearSold')
```

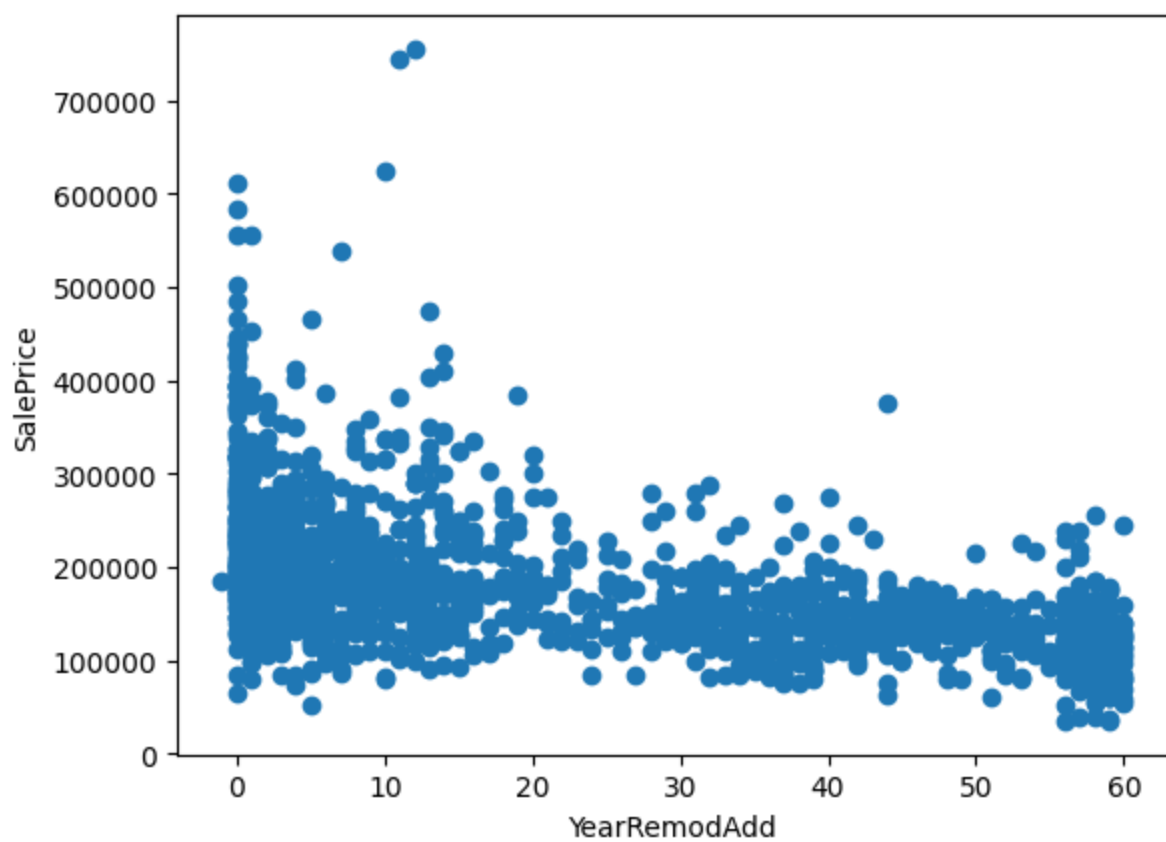
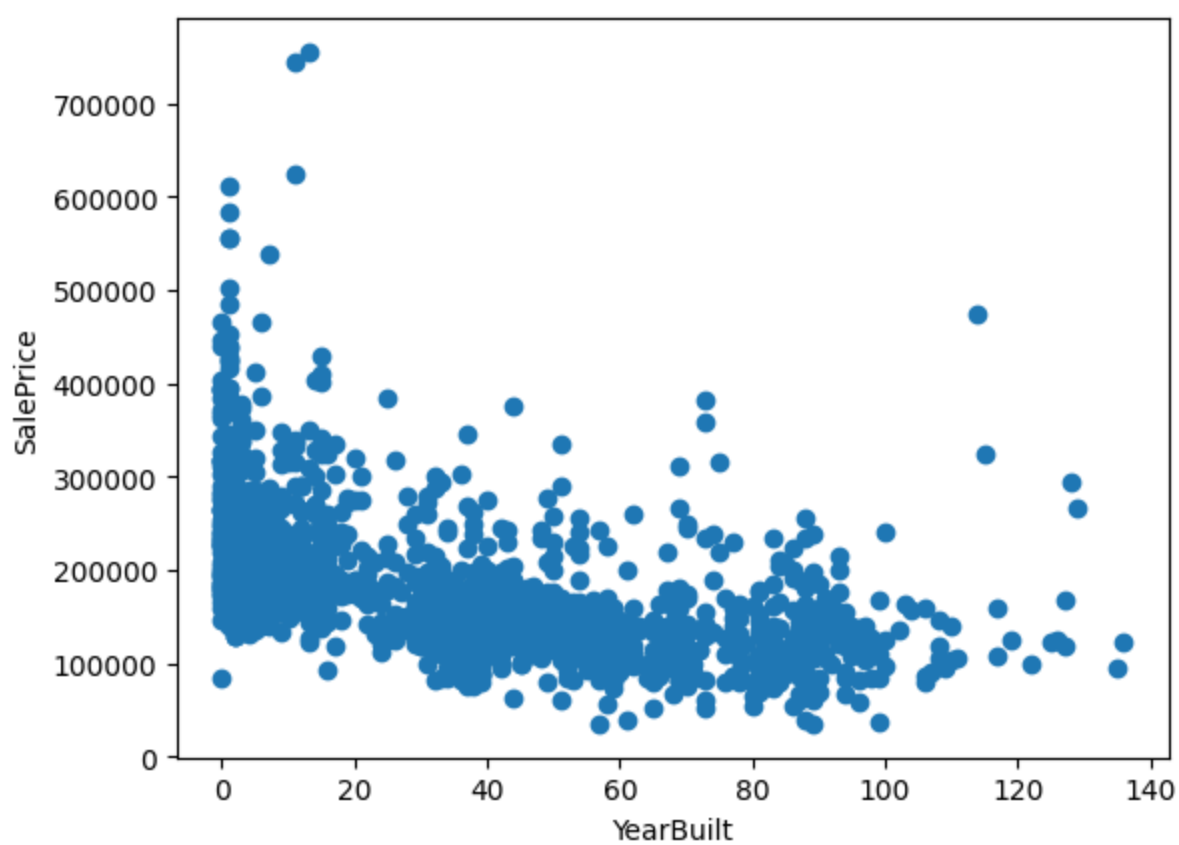


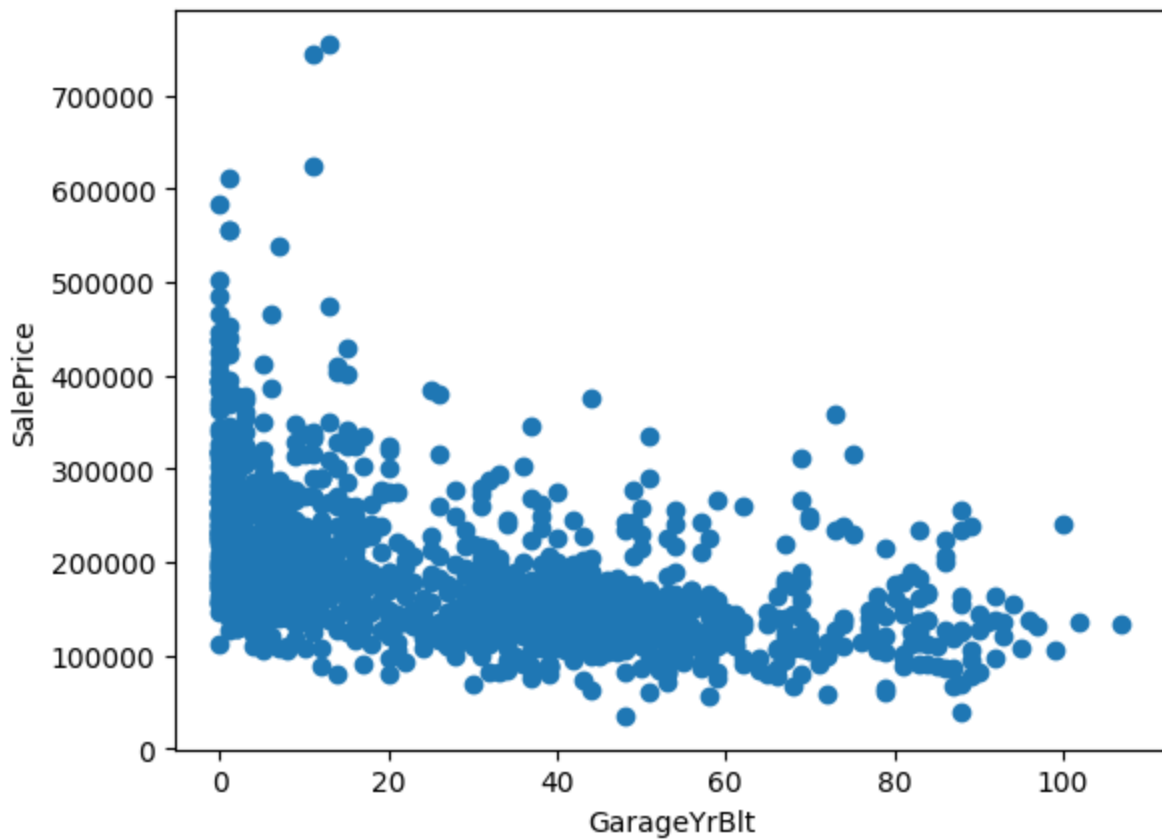
This info doesn't look very promising as with the years passing the sales price is dropping so we will try to find more info.

```
In [66]: year_feature
```

```
Out[66]: ['YearBuilt', 'YearRemodAdd', 'GarageYrBlt', 'YrSold']
```

```
In [98]: #now we will compare every year feature with Sales price  
  
for feature in year_feature:  
    if feature!='YrSold':  
        data=dataset.copy()  
        ## We will capture the difference between year variable and year the house was s  
        data[feature]=data['YrSold']-data[feature]  
  
        plt.scatter(data[feature],data['SalePrice'])  
        plt.xlabel(feature)  
        plt.ylabel('SalePrice')  
        plt.show()
```





```
In [68]: #Numerical variables are of two types 1. Continuous 2.Descrete

discrete_features=[feature for feature in numerical_features if len(dataset[feature].unique()) > 1]
print("Discrete Vatiables Count: {}".format(len(discrete_features)))
```

Discrete Vatiables Count: 17

```
In [99]: discrete_features
```

```
Out[99]: ['MSSubClass',
'OverallQual',
'OverallCond',
'LowQualFinSF',
'BsmtFullBath',
'BsmtHalfBath',
'FullBath',
'HalfBath',
'BedroomAbvGr',
'KitchenAbvGr',
'TotRmsAbvGrd',
'Fireplaces',
'GarageCars',
'3SsnPorch',
'PoolArea',
'MiscVal',
'MoSold']
```

```
In [69]: data[discrete_features].head()
```

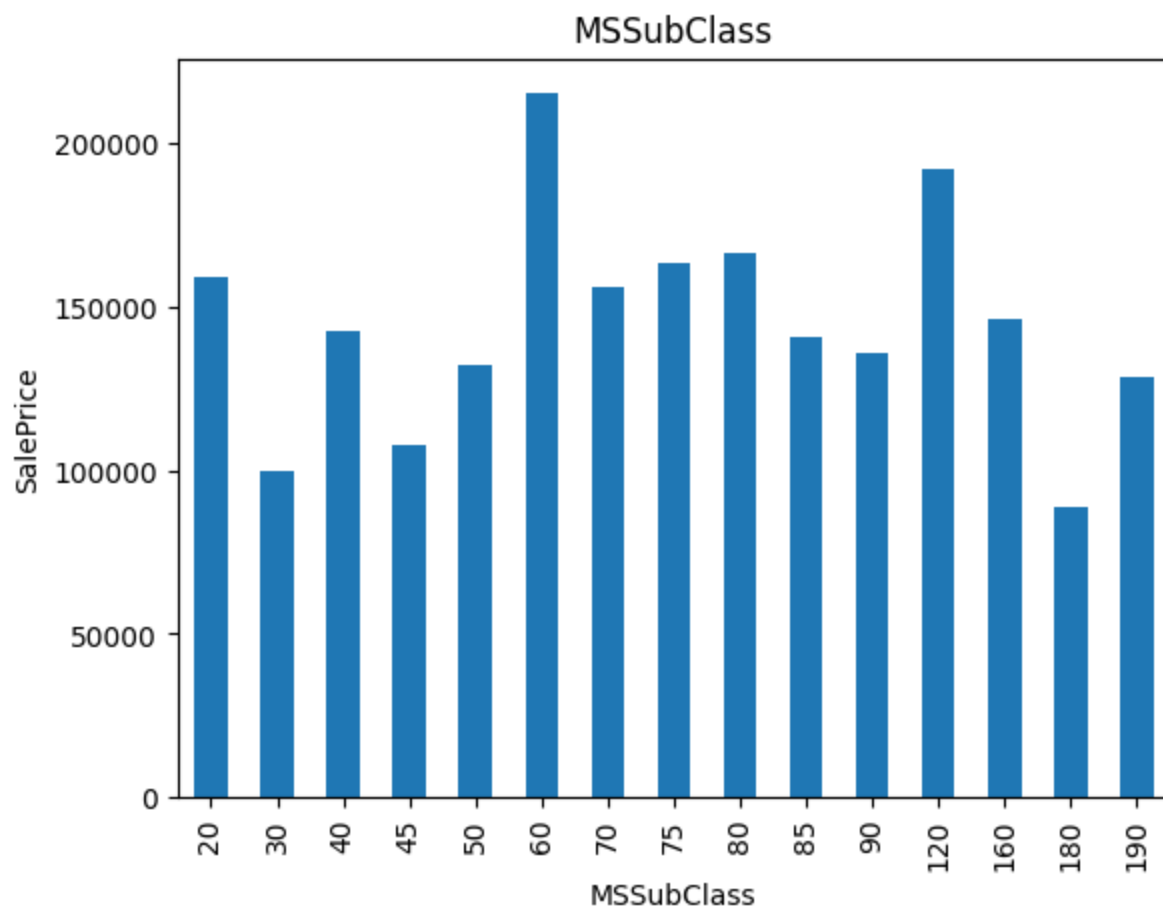
```
Out[69]:
```

	MSSubClass	OverallQual	OverallCond	LowQualFinSF	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	Bedroo
0	60	7	5	0	1	0	2	1	
1	20	6	8	0	0	1	2	0	
2	60	7	5	0	1	0	2	1	
3	70	7	5	0	1	0	1	0	

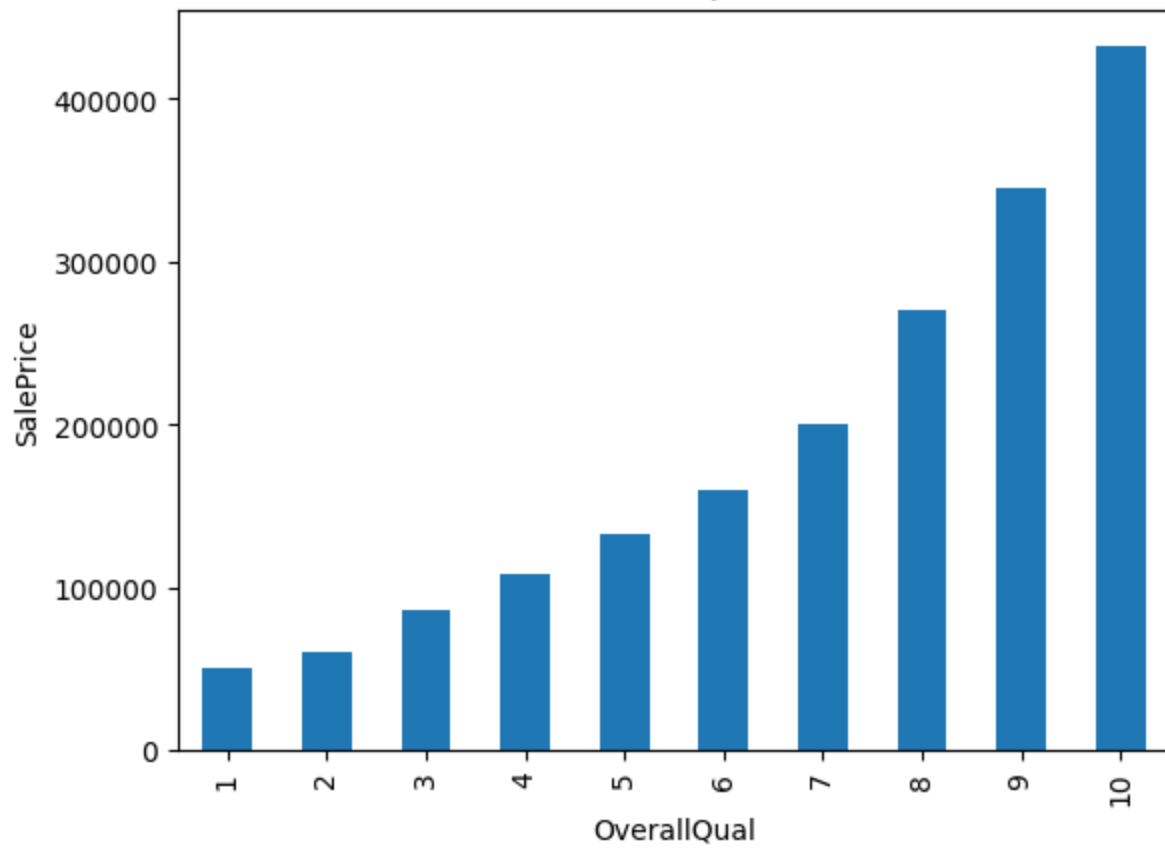
There is a relationship between them and Sales Price

```
In [70]: ## Lets Find the realtionship between them and Sale PRice

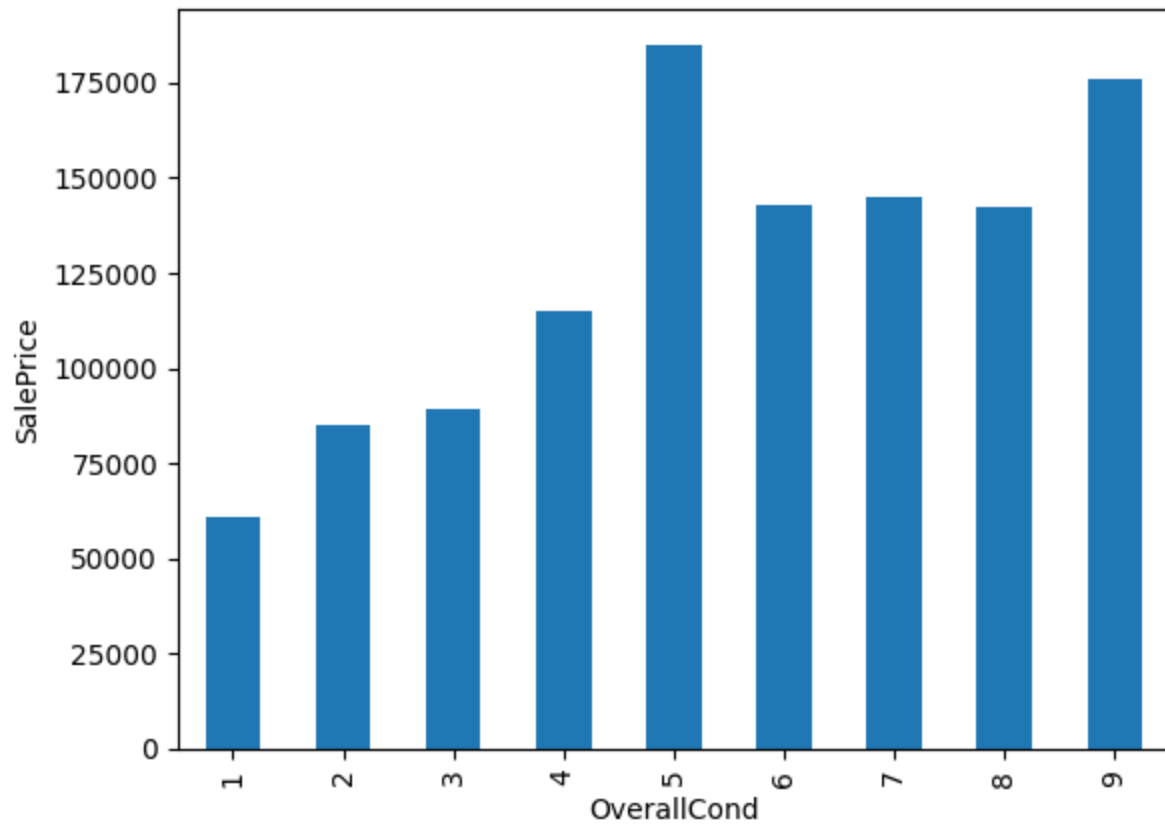
for feature in discrete_features:
    data=dataset.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```



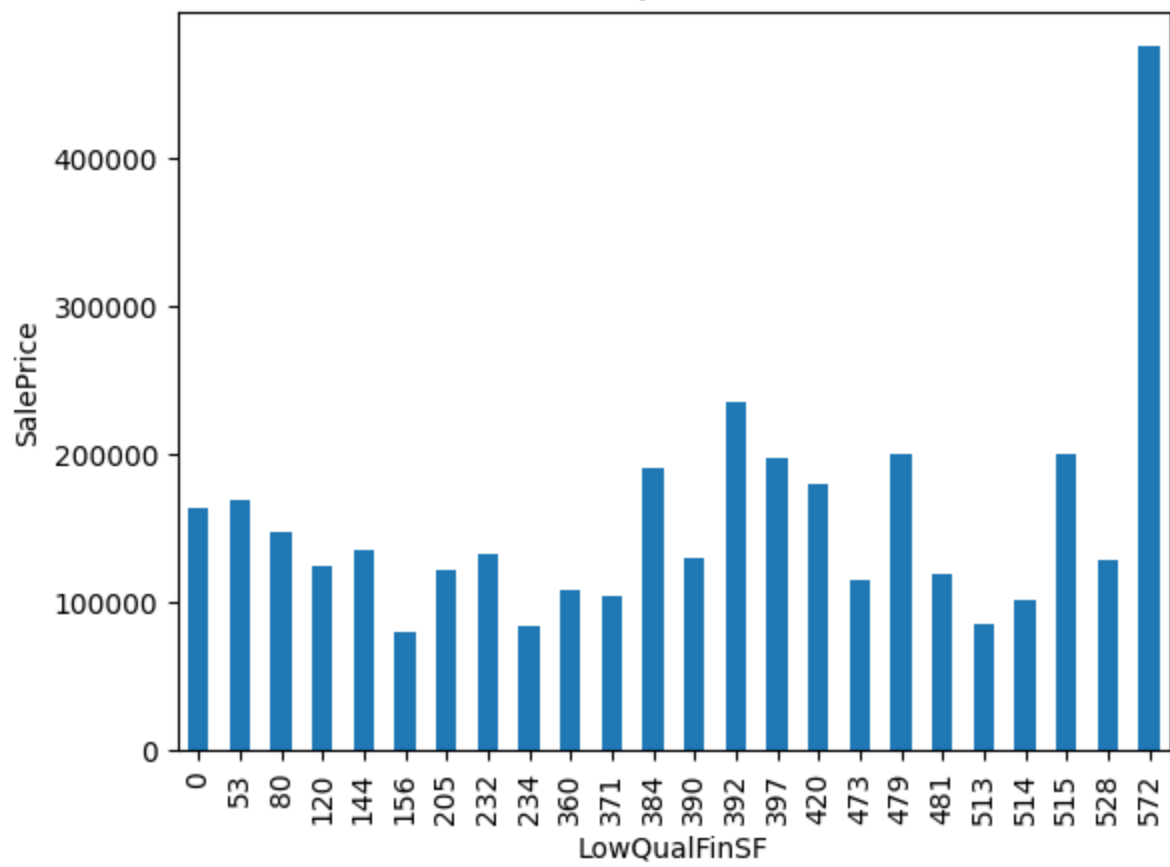
OverallQual



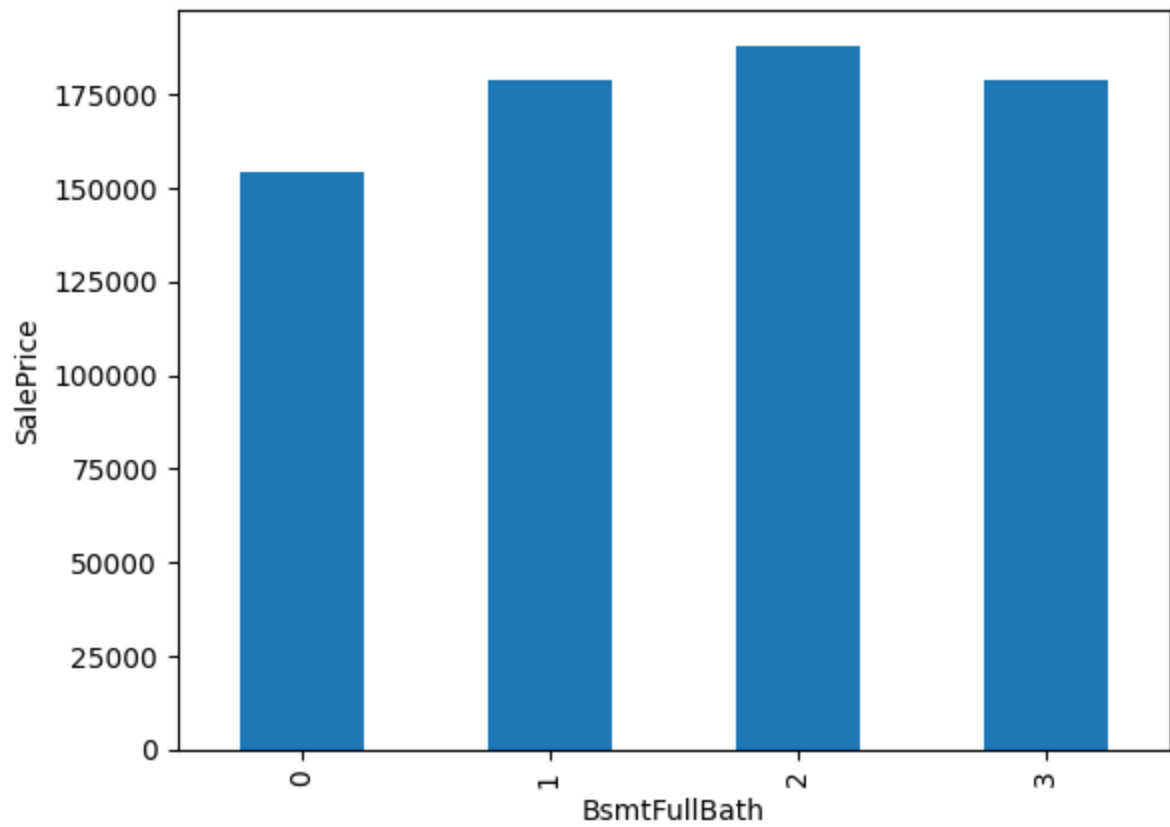
OverallCond



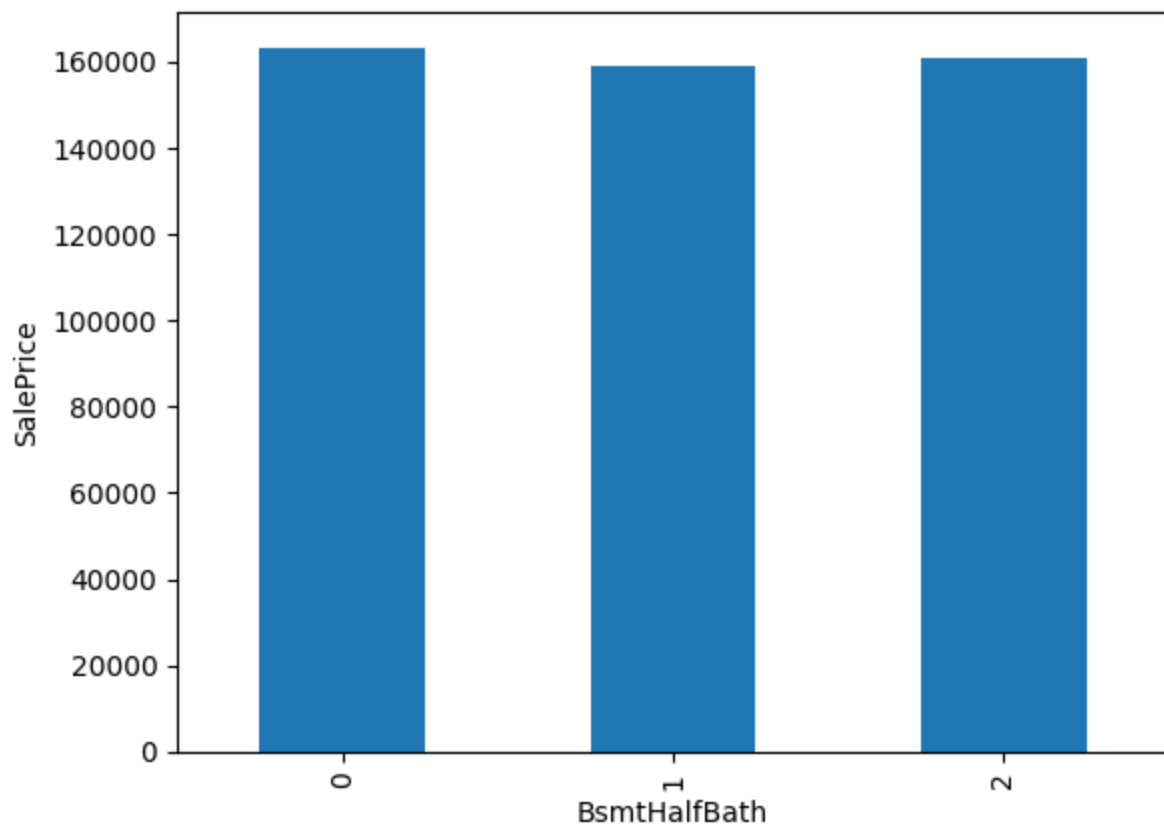
LowQualFinSF



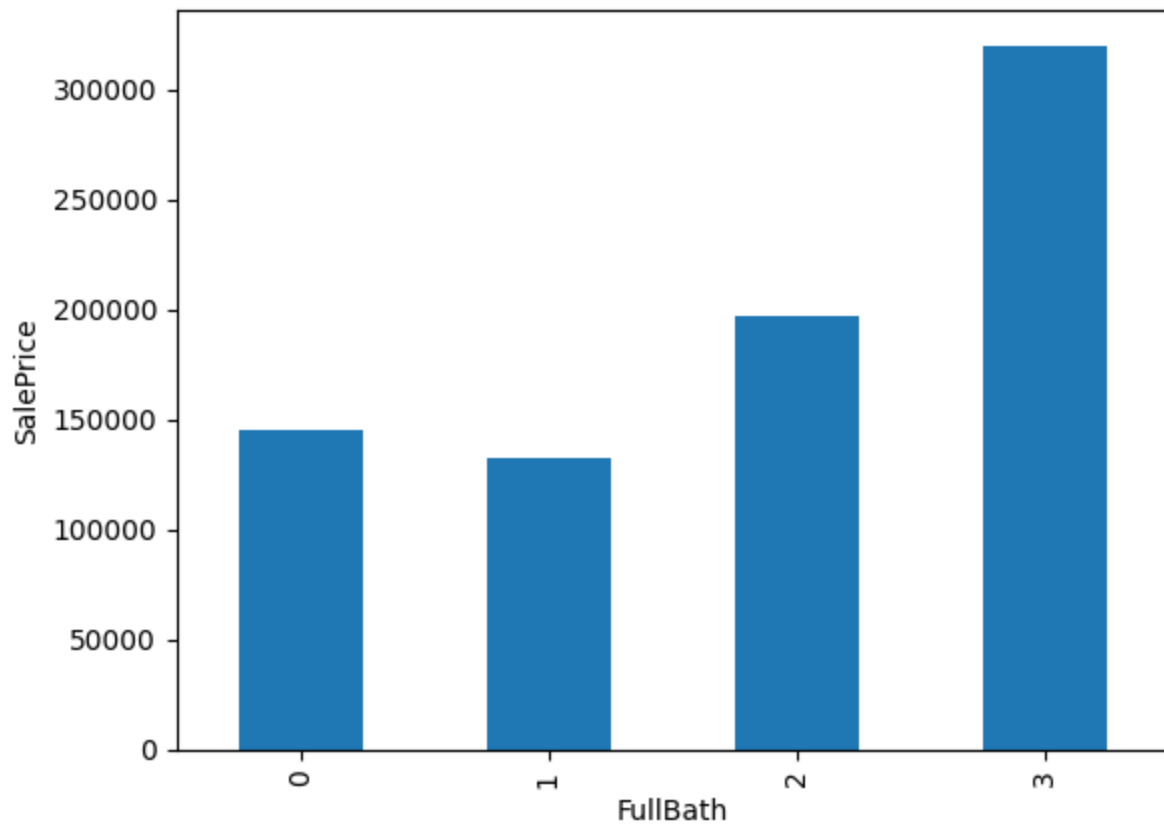
BsmtFullBath

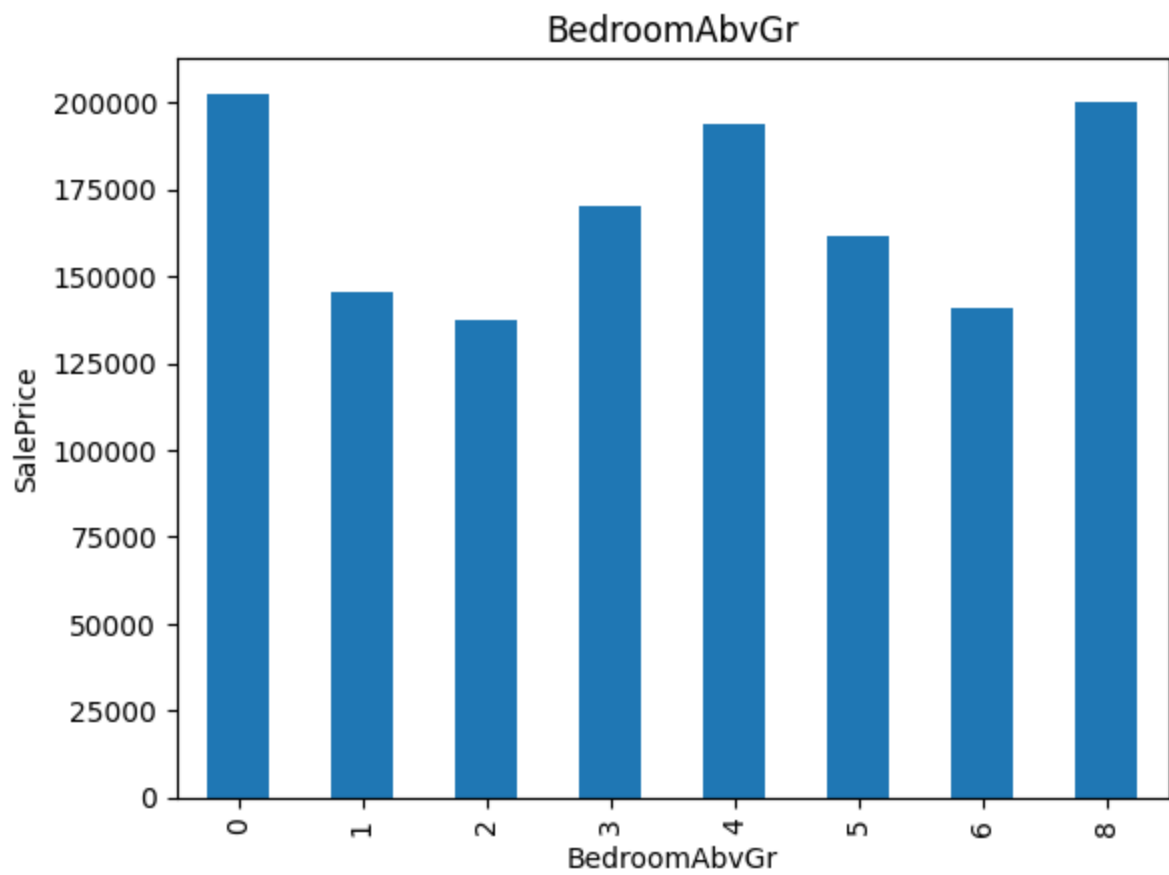
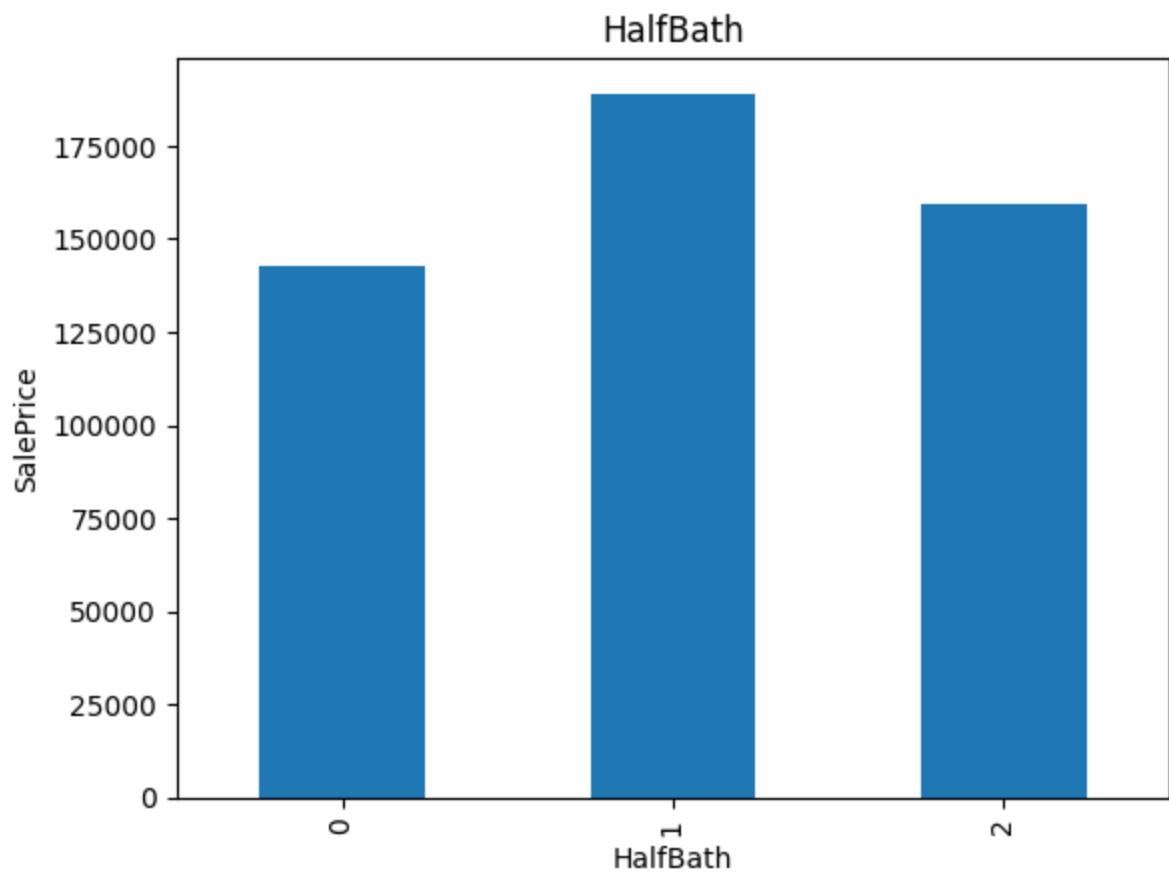


BsmtHalfBath

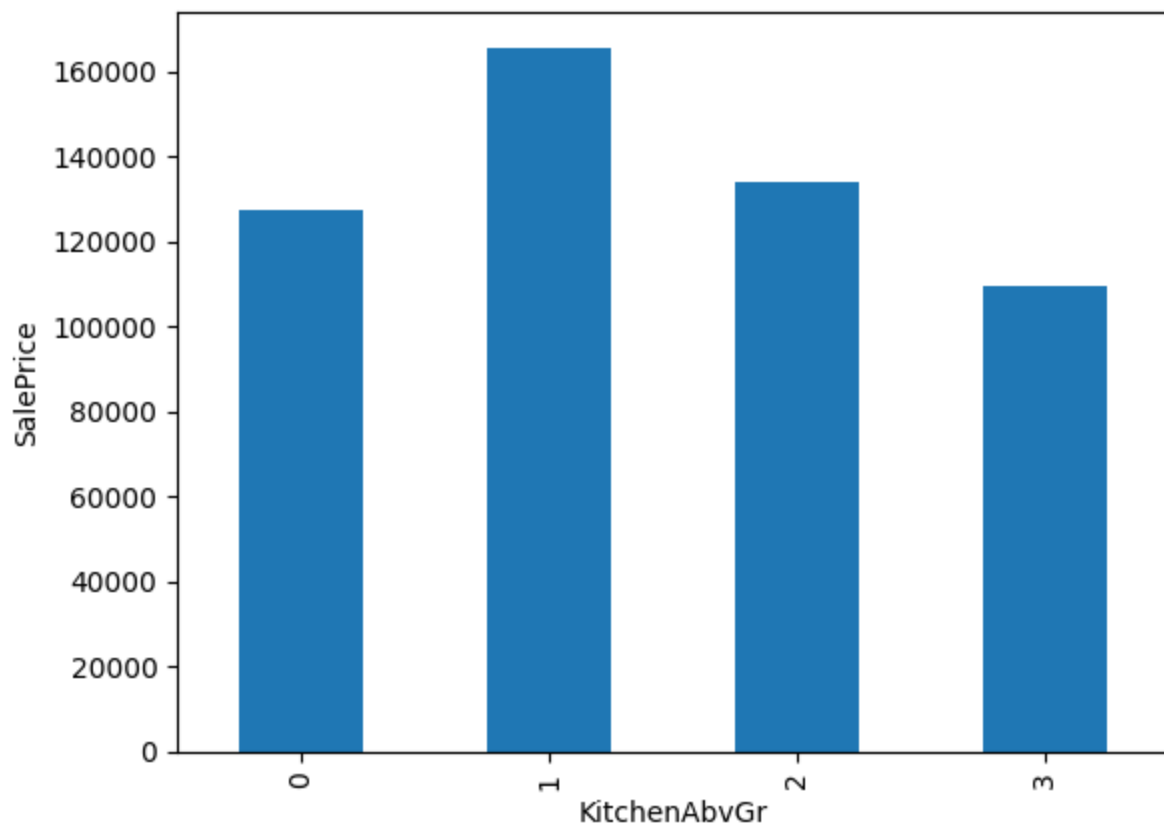


FullBath

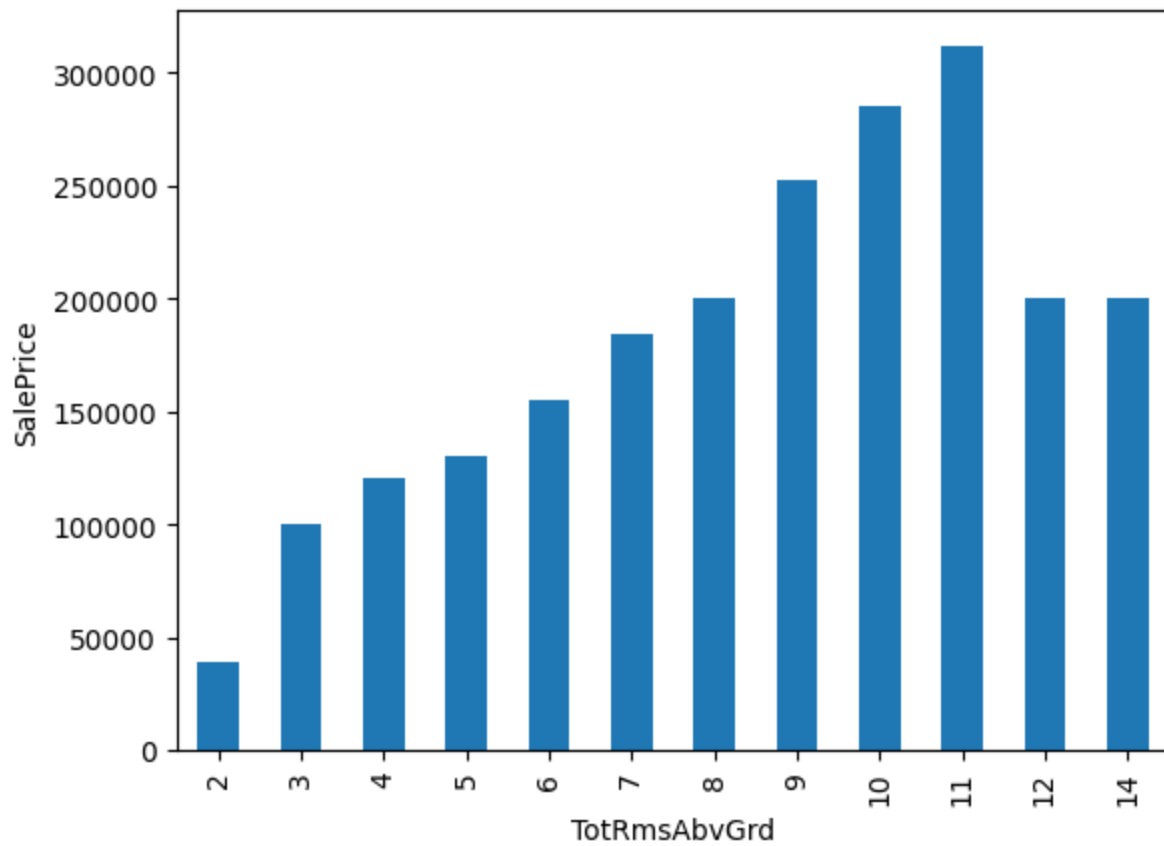


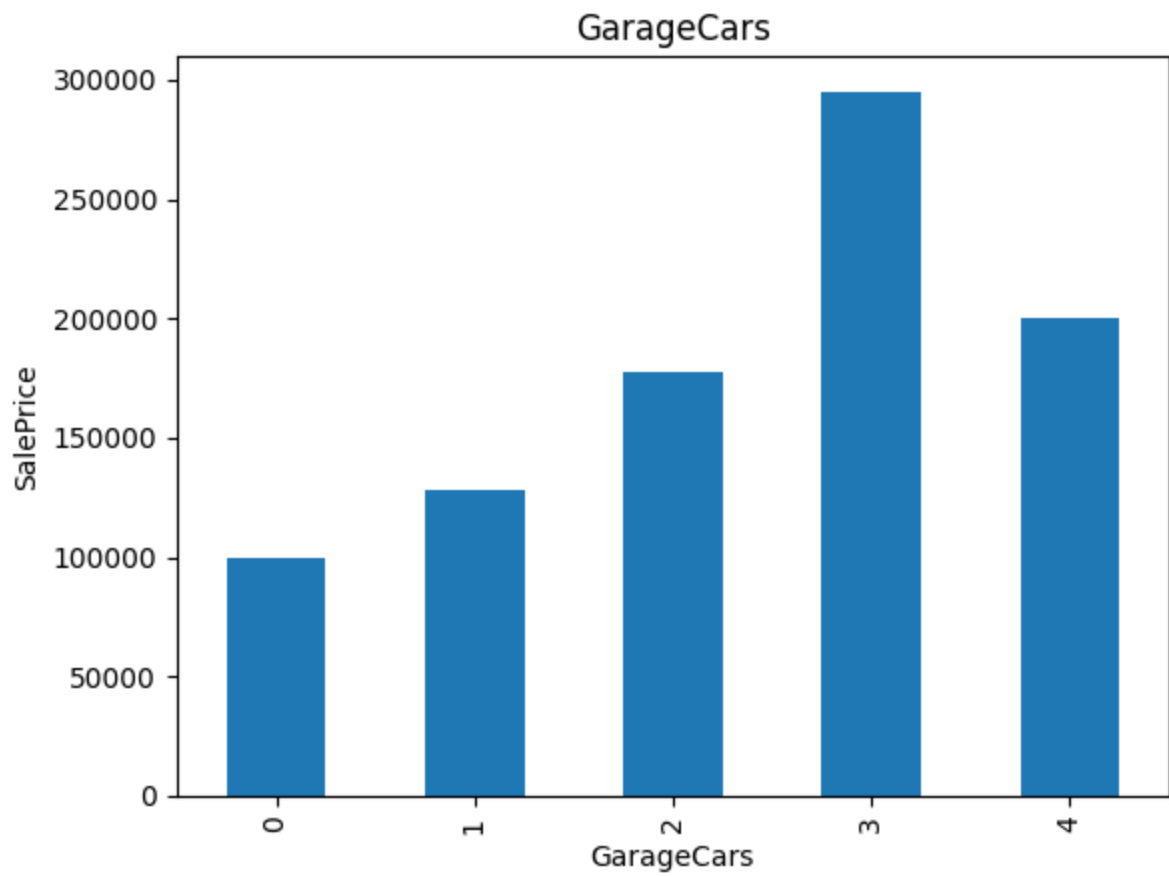
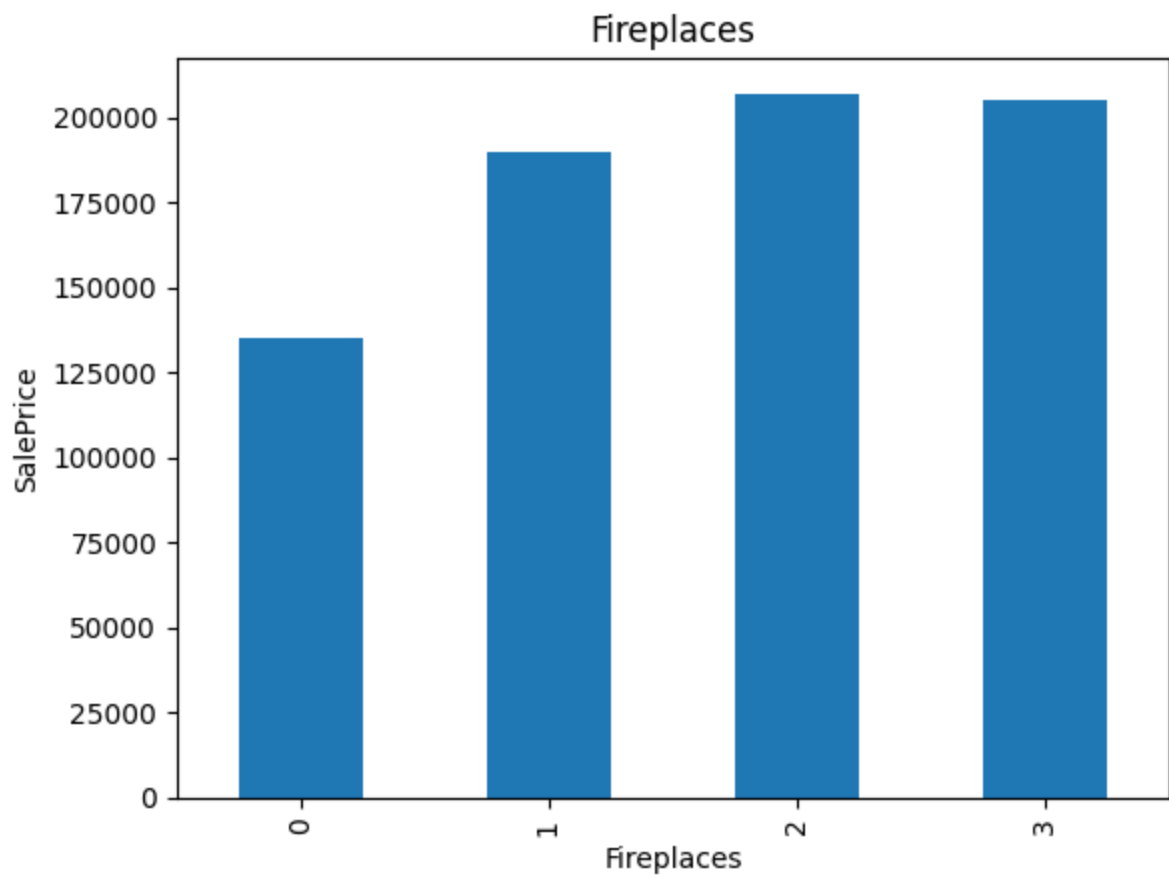


KitchenAbvGr

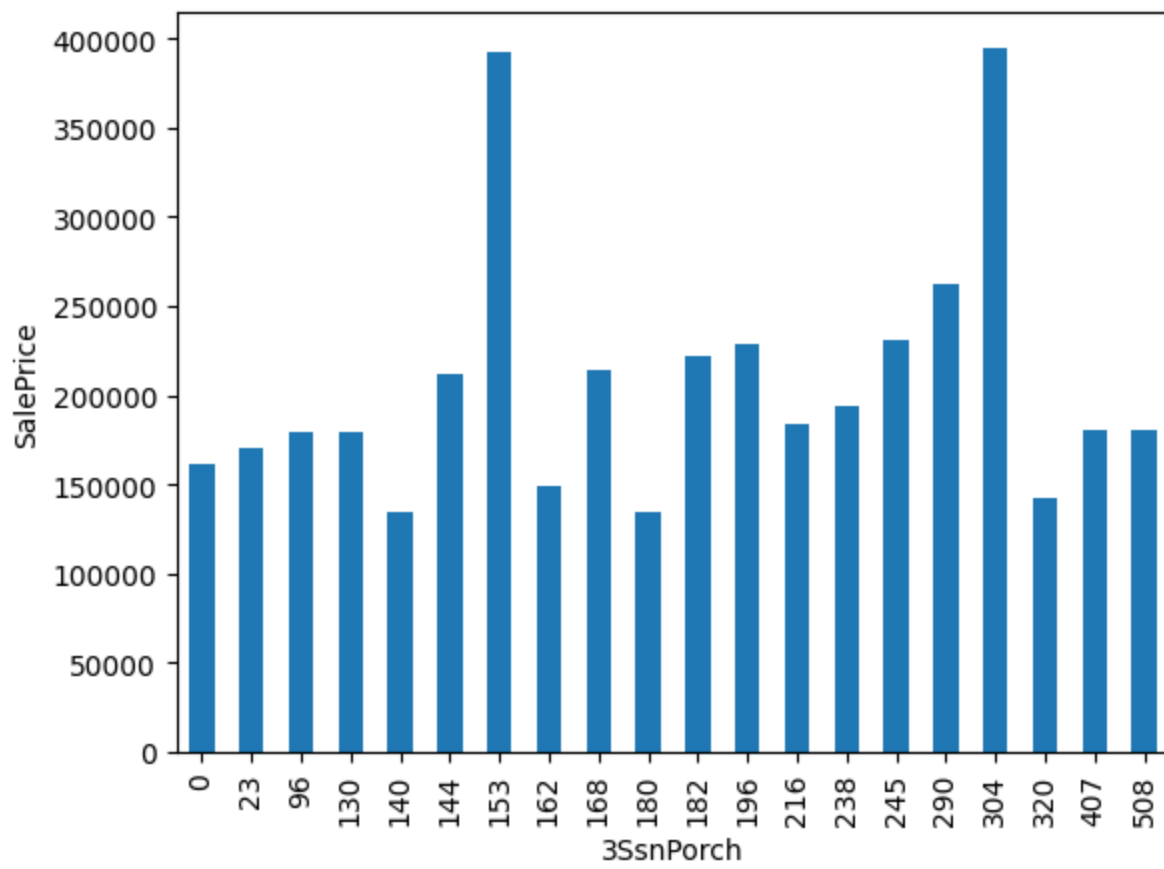


TotRmsAbvGrd

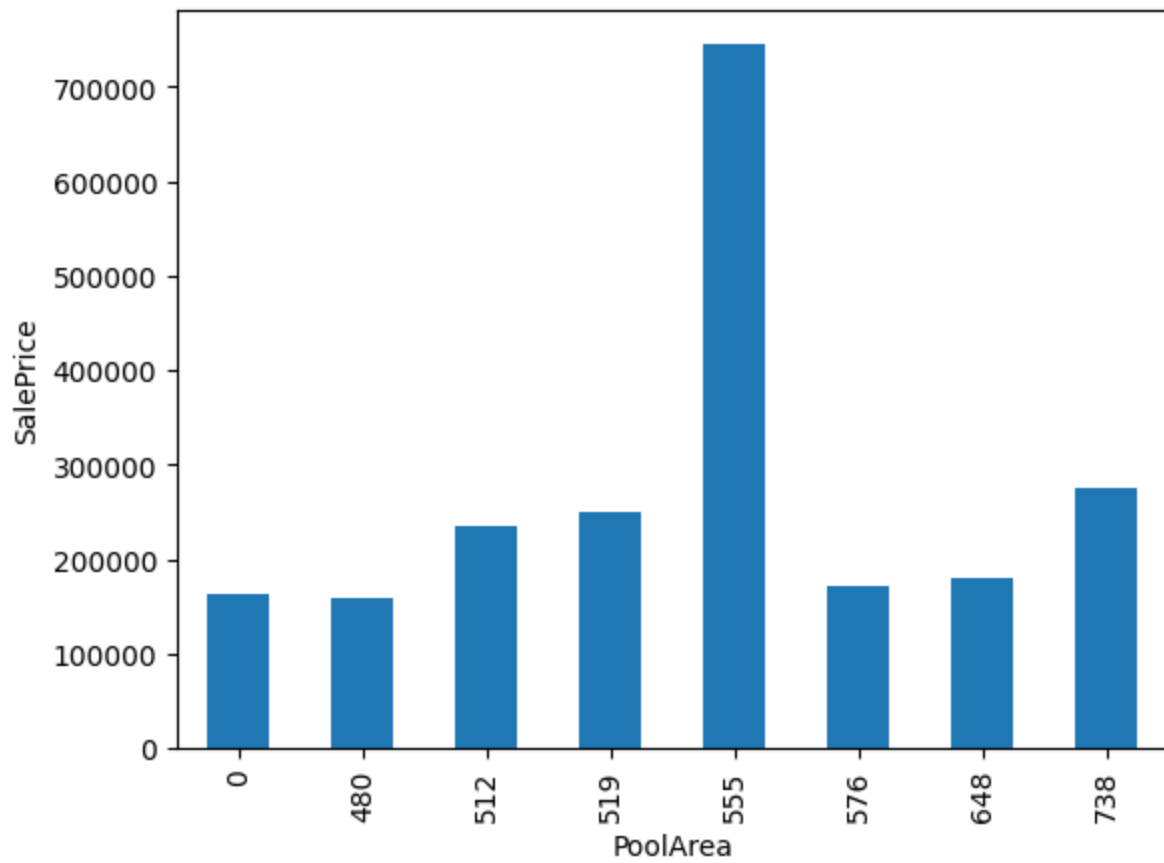


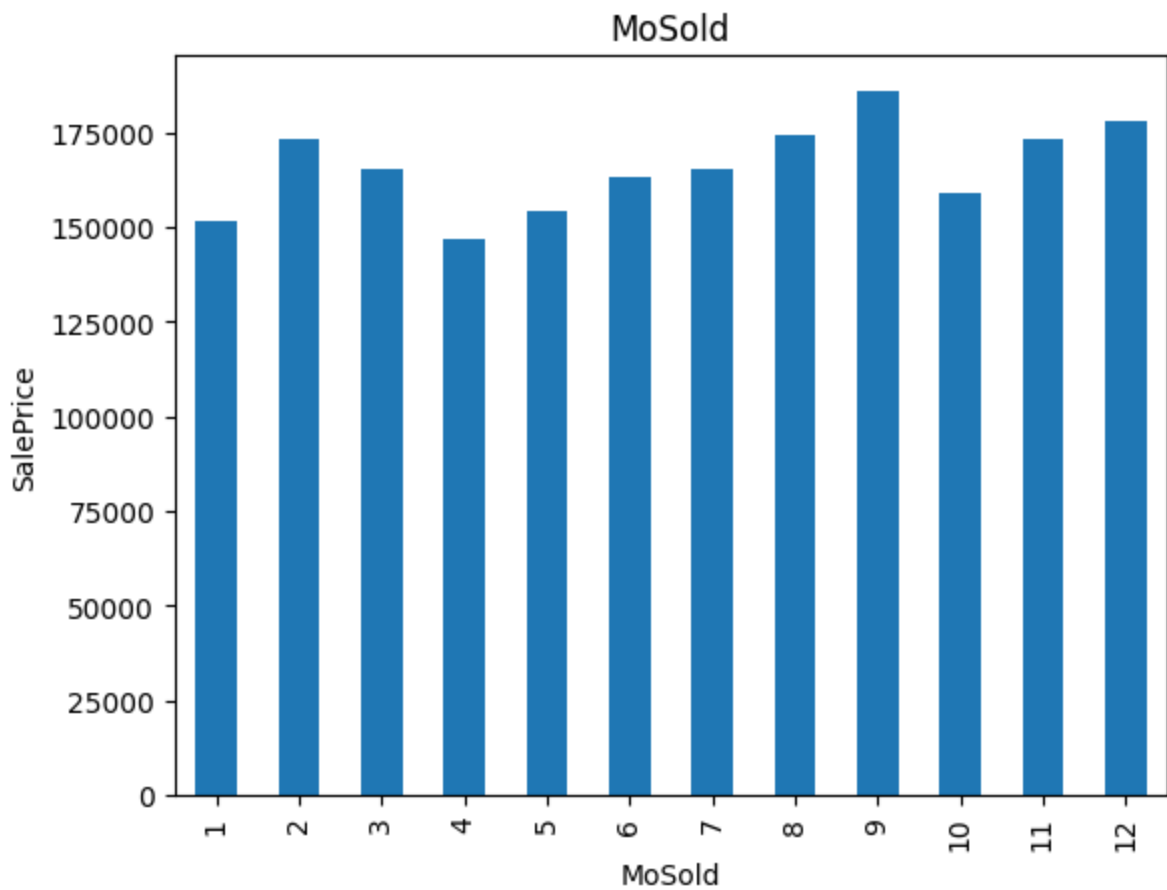
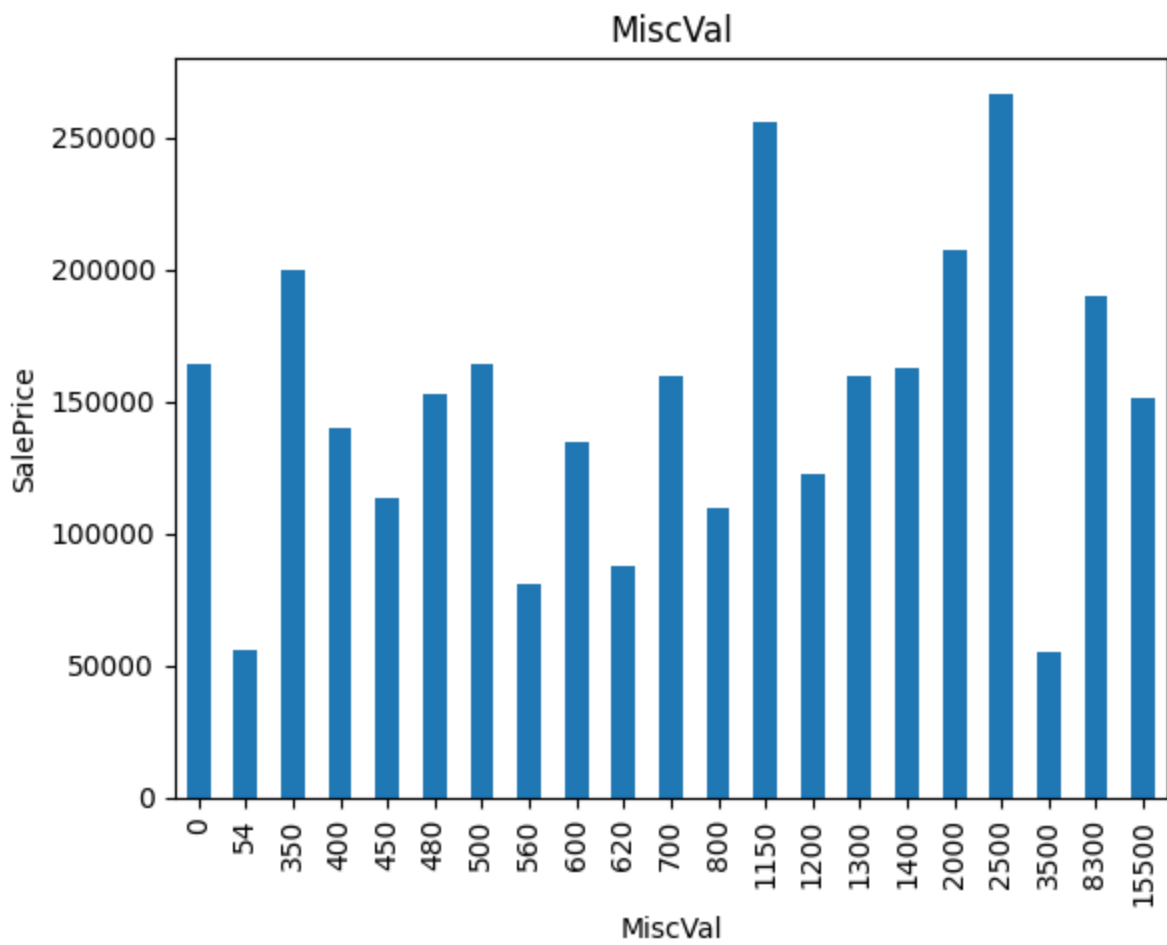


3SsnPorch



PoolArea





Continuous Variables

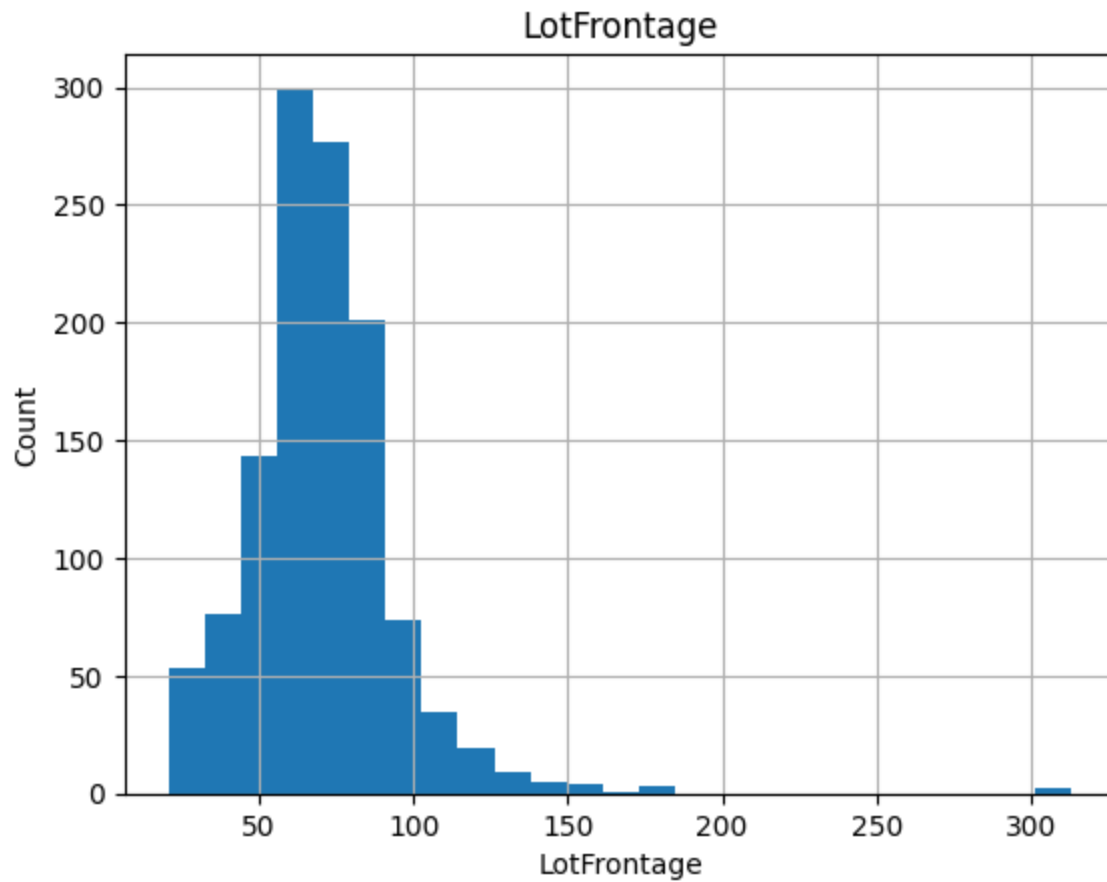
```
In [71]: continuous_features=[feature for feature in numerical_features if feature not in discret
```

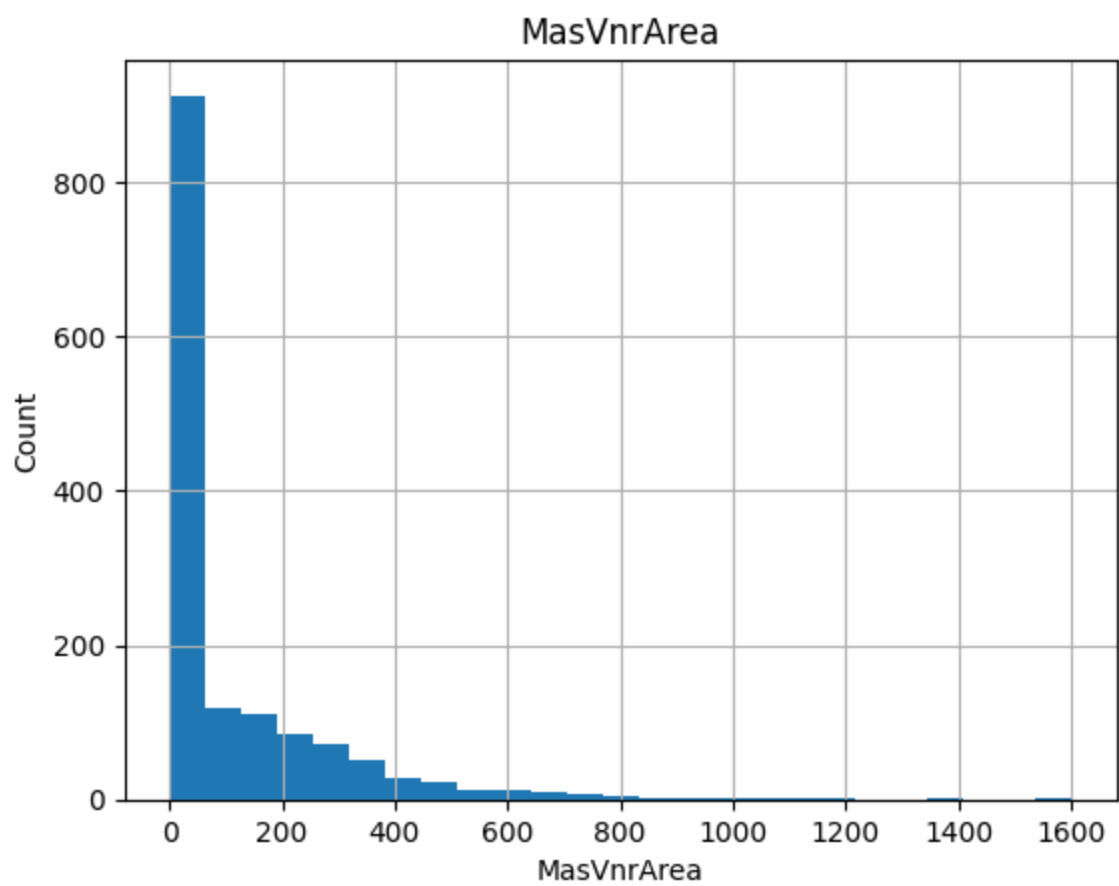
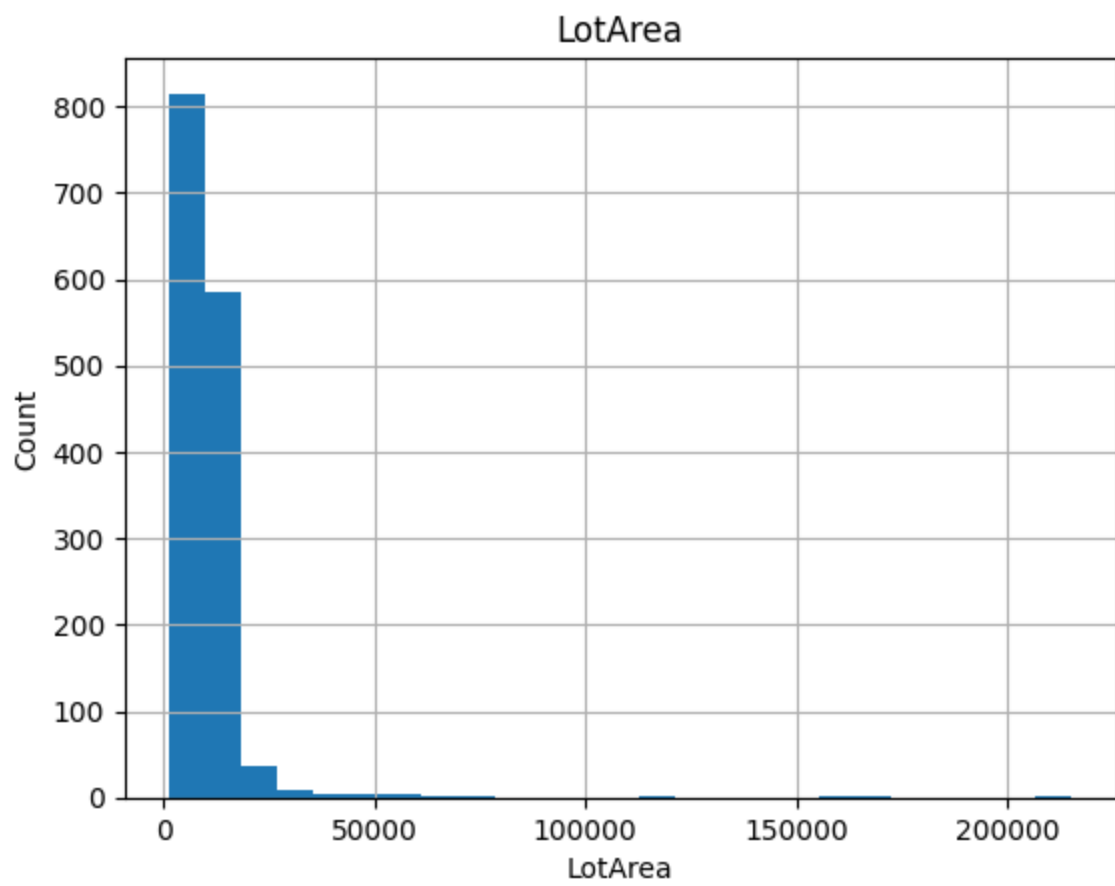
```
print("Continuous Feature Count {}".format(len(continuous_features)))
```

Continuous Feature Count 16

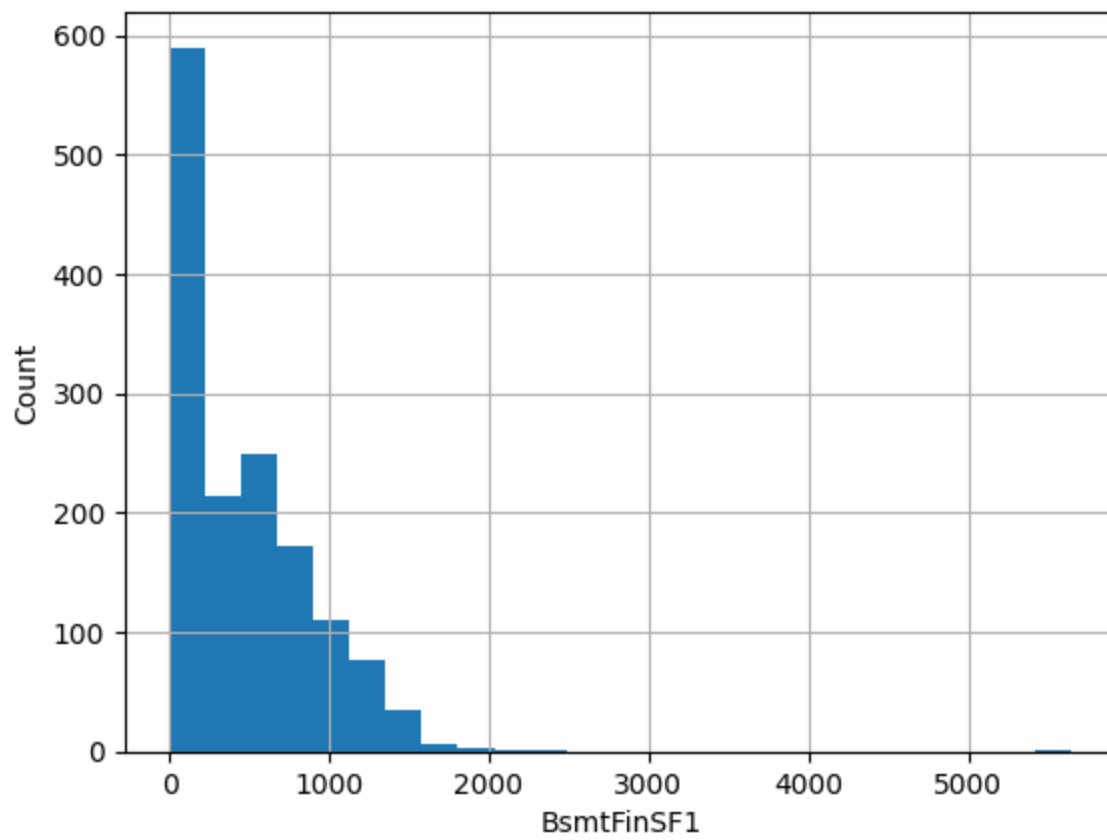
```
In [72]: # Analyzing continuous values by creating histograms to understand the

for feature in continuous_features:
    data=dataset.copy()
    data[feature].hist(bins=25)
    plt.xlabel(feature)
    plt.ylabel('Count')
    plt.title(feature)
    plt.show()
```

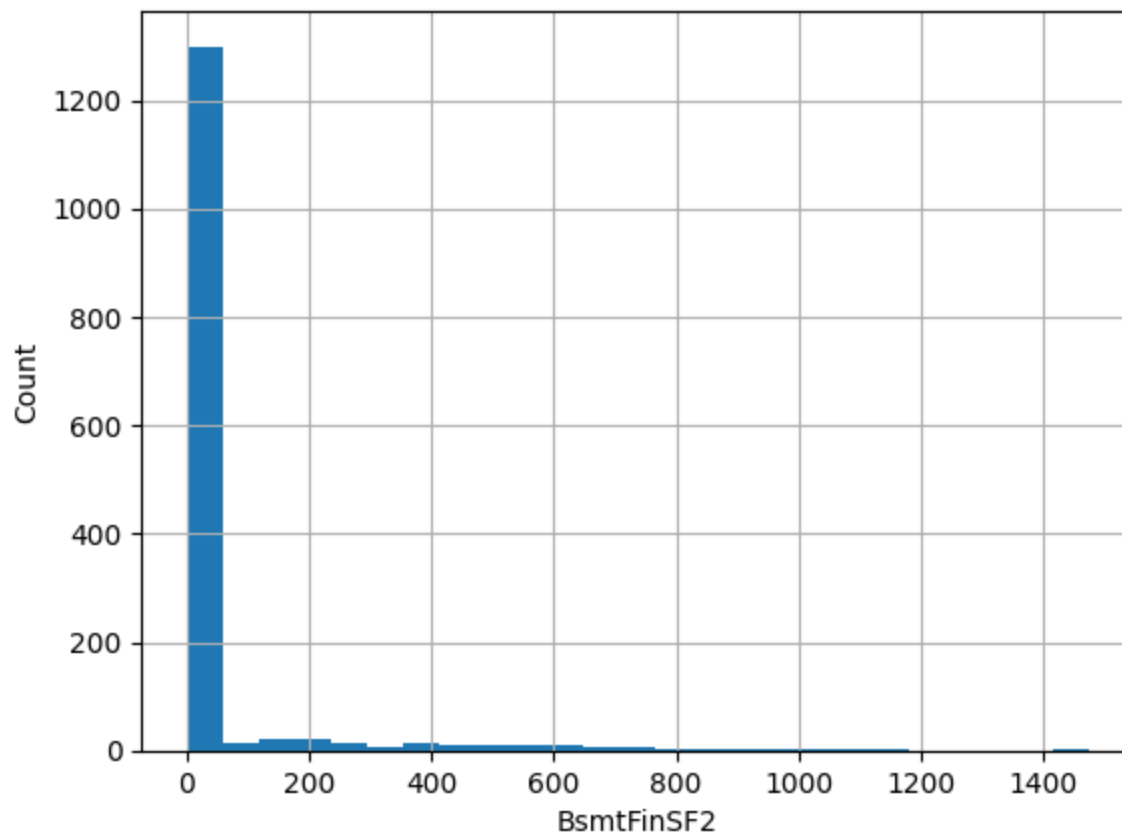




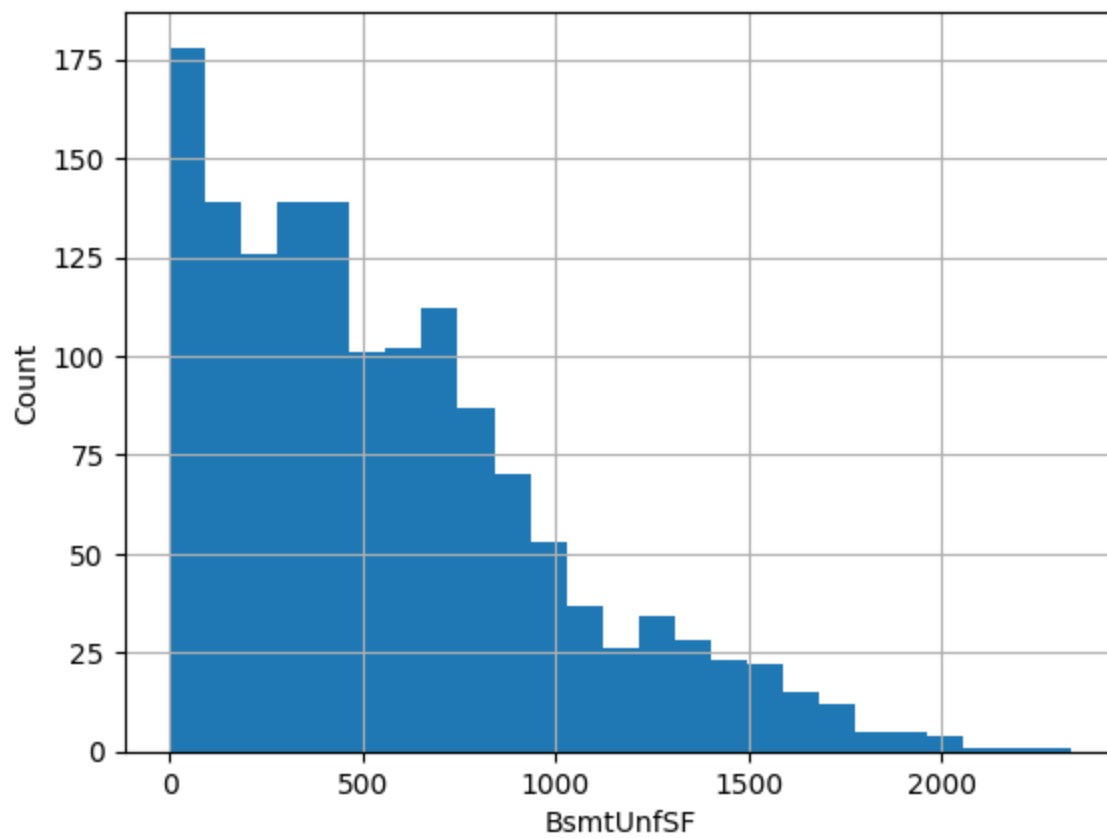
BsmtFinSF1



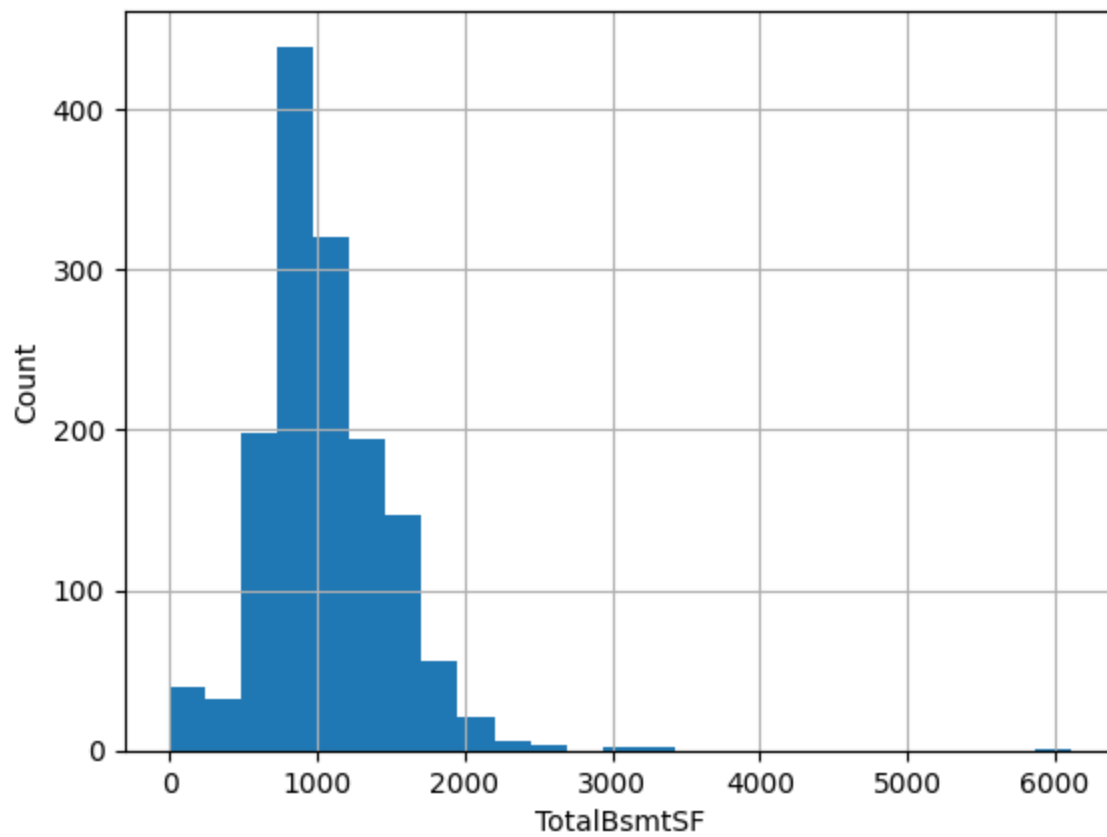
BsmtFinSF2



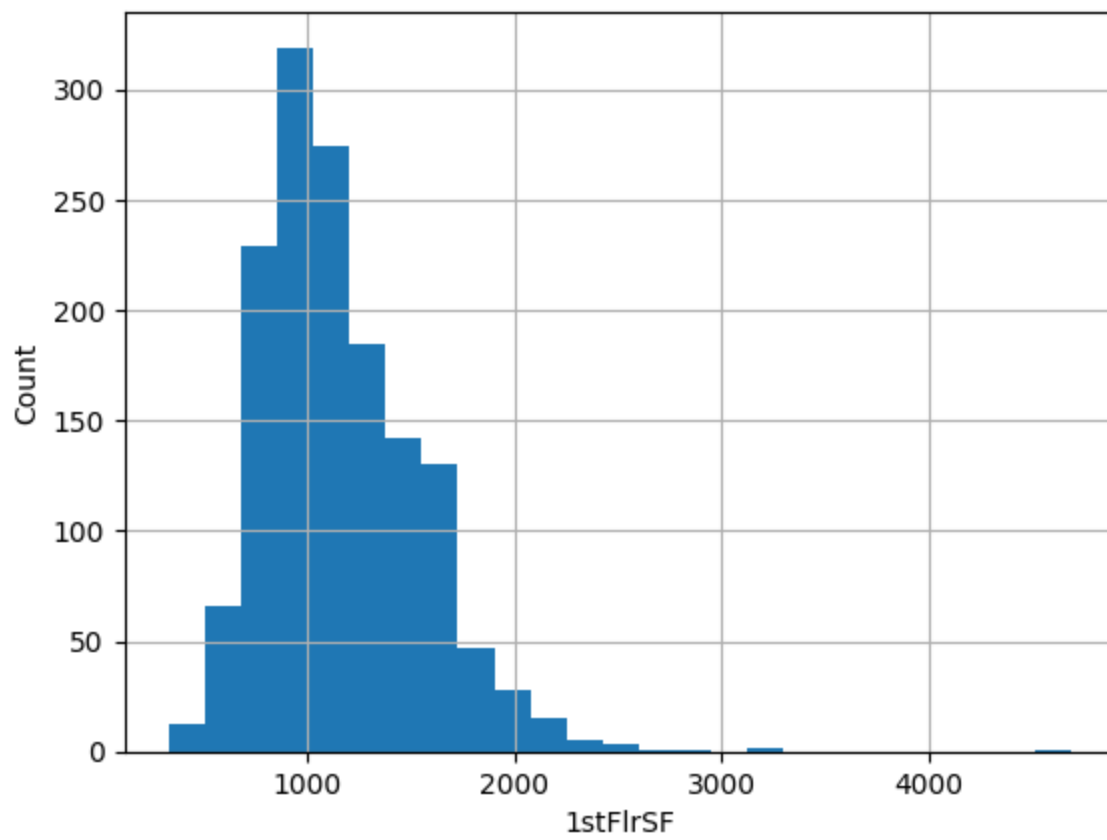
BsmtUnfSF



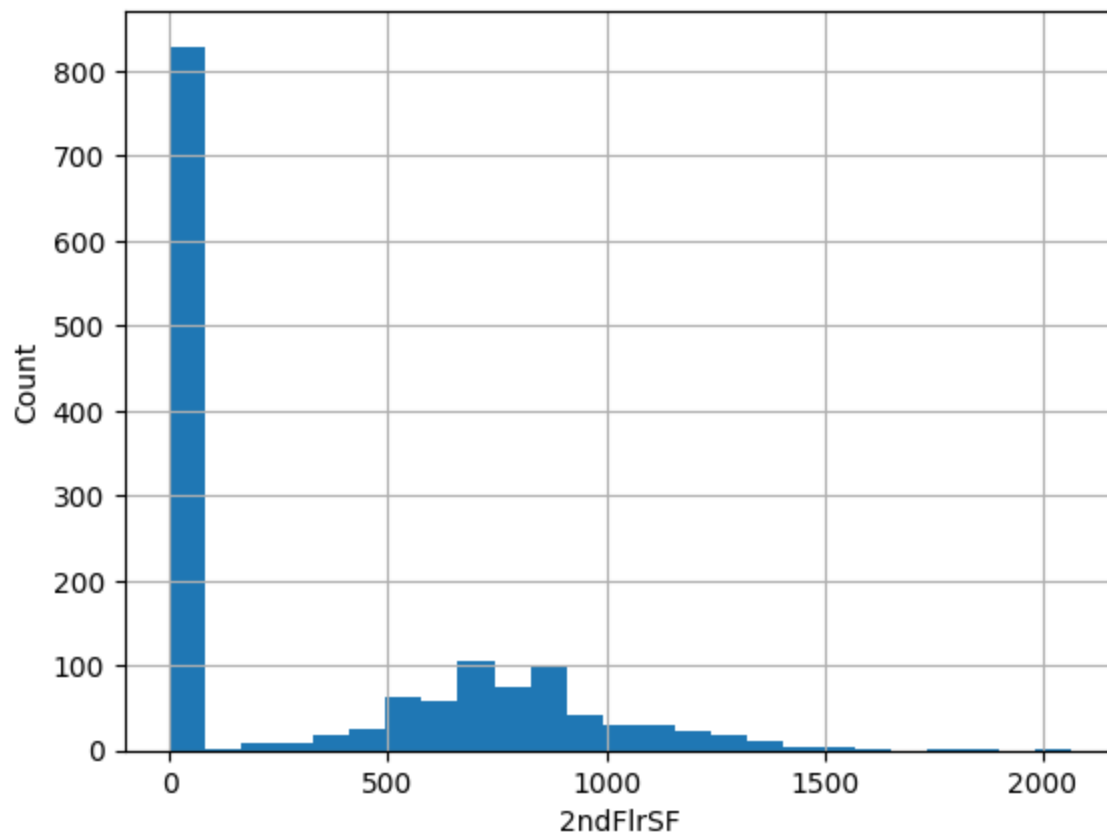
TotalBsmtSF

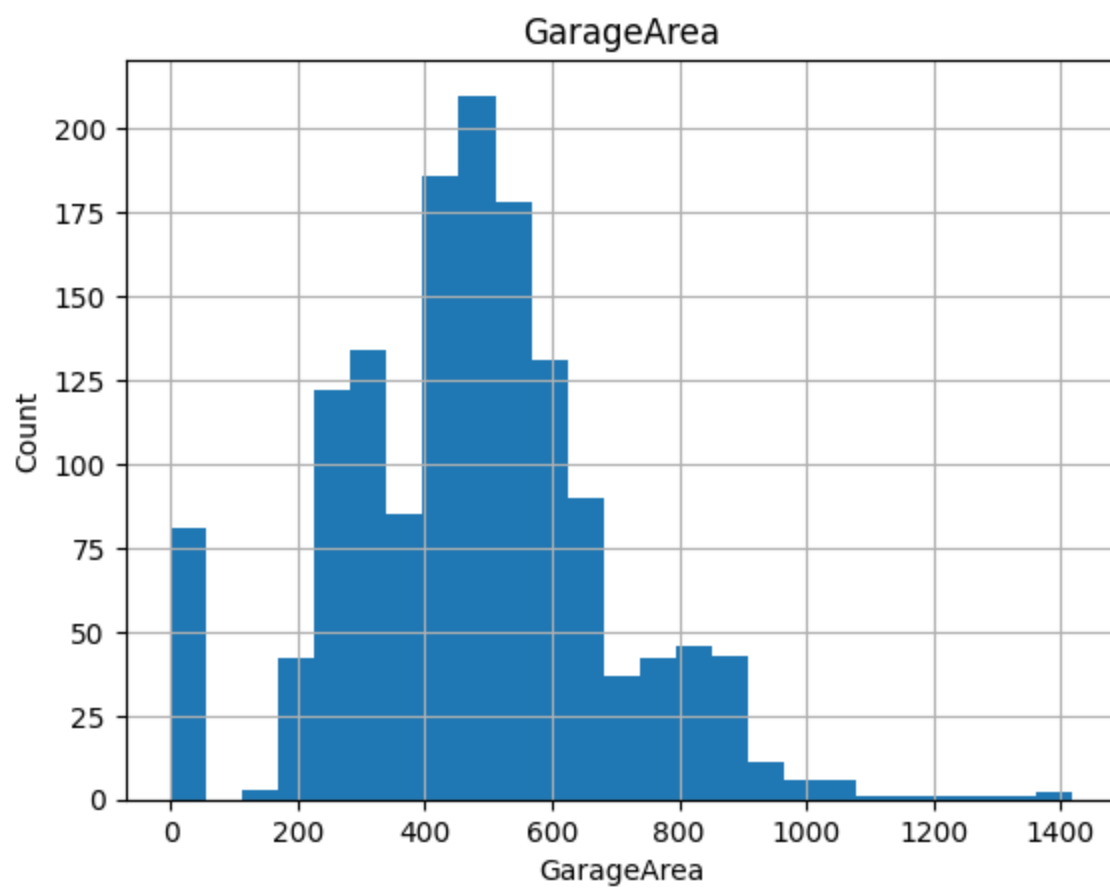
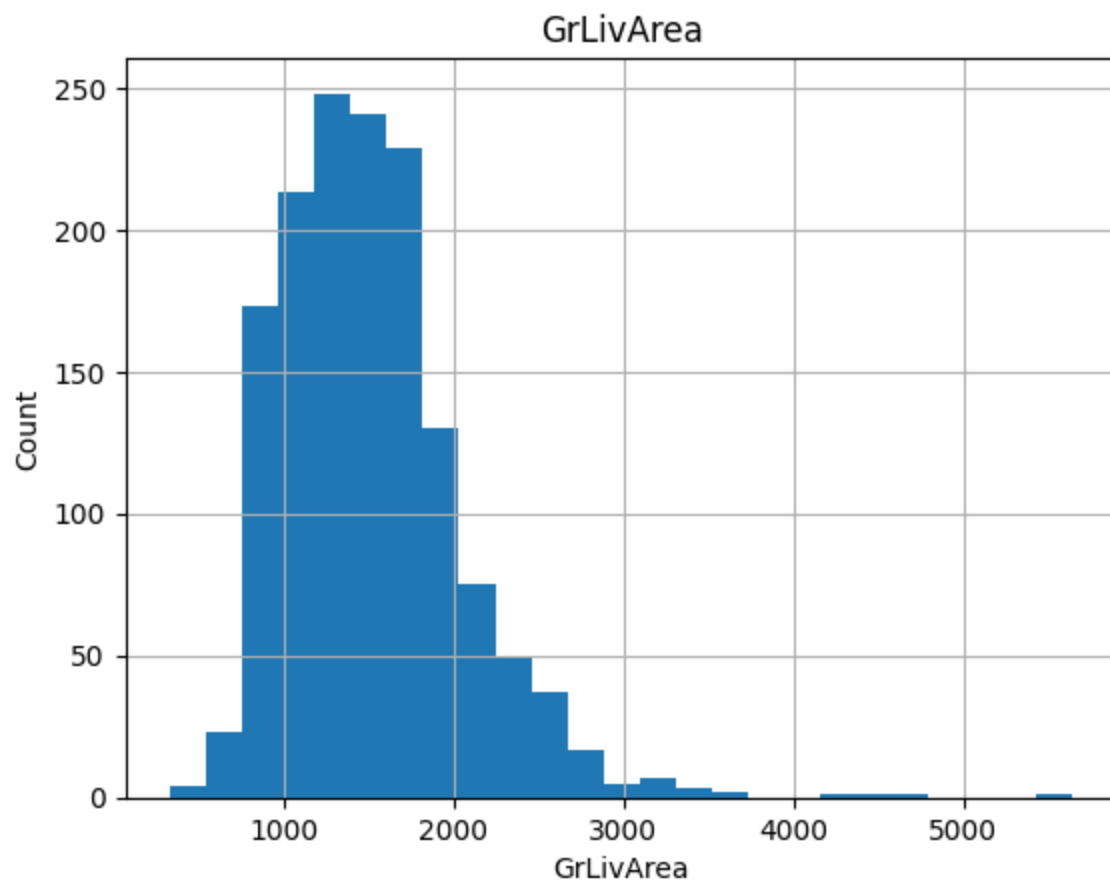


1stFlrSF

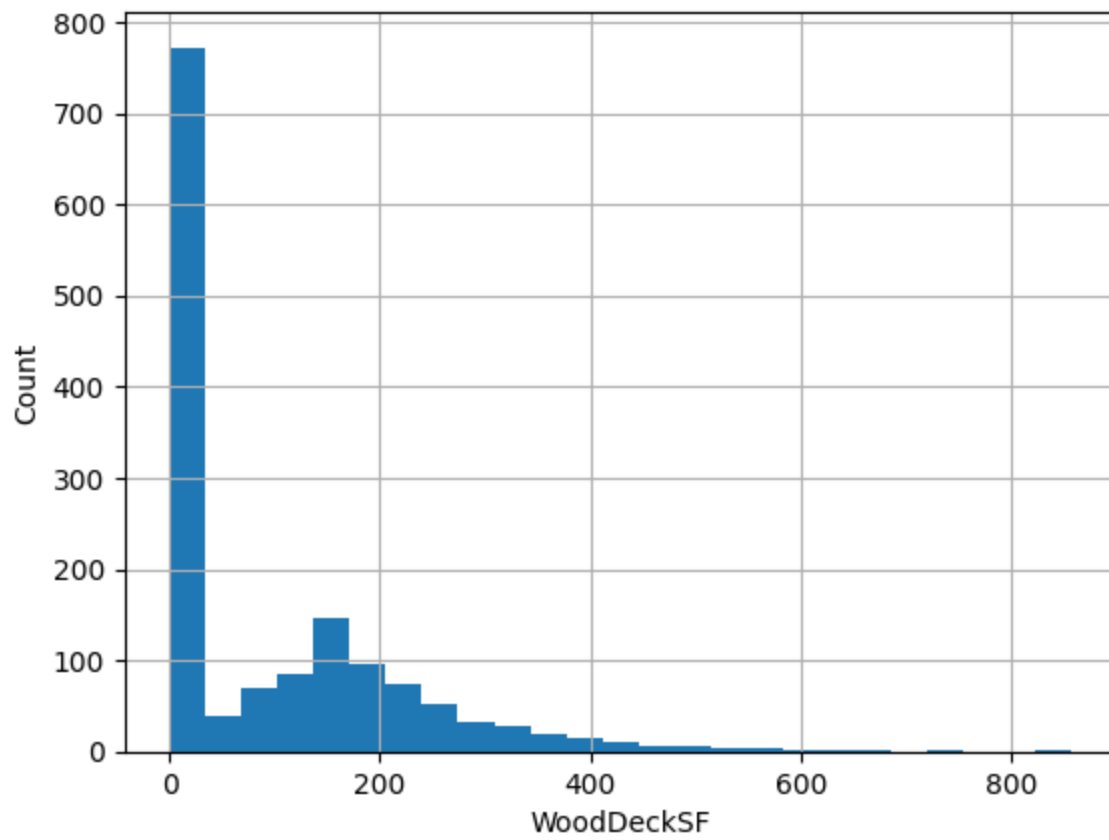


2ndFlrSF

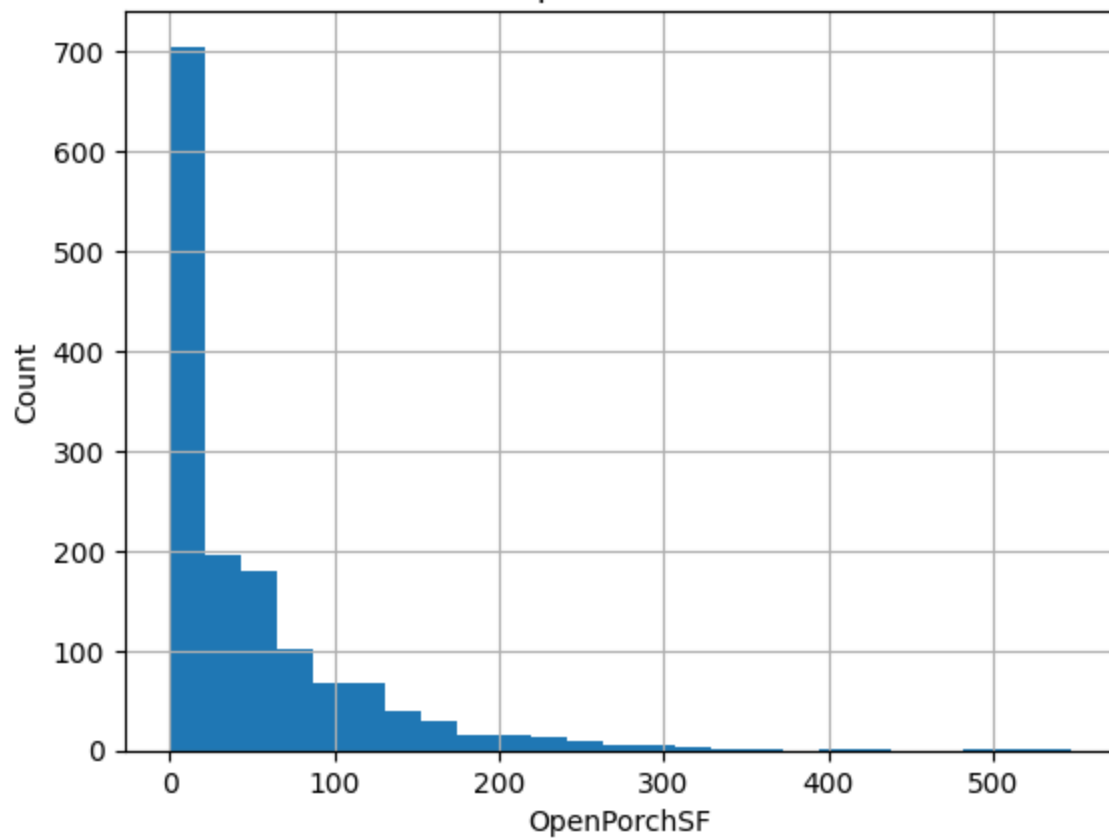




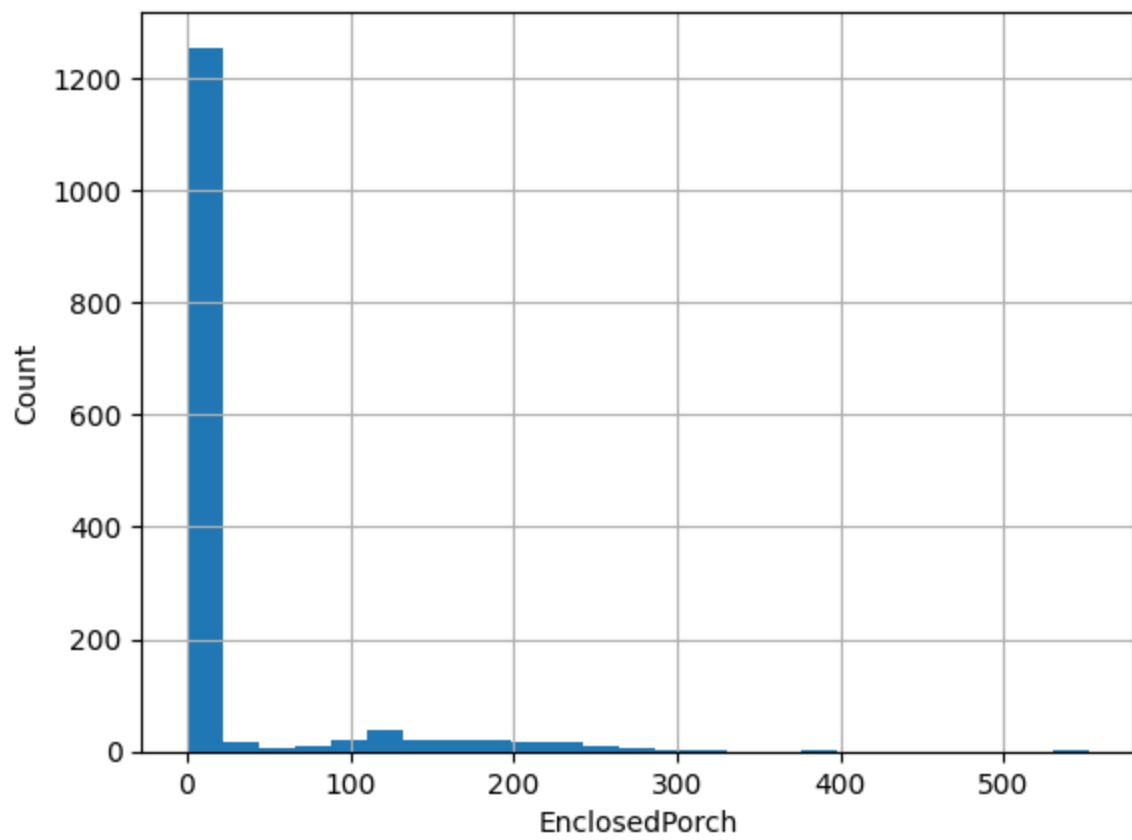
WoodDeckSF



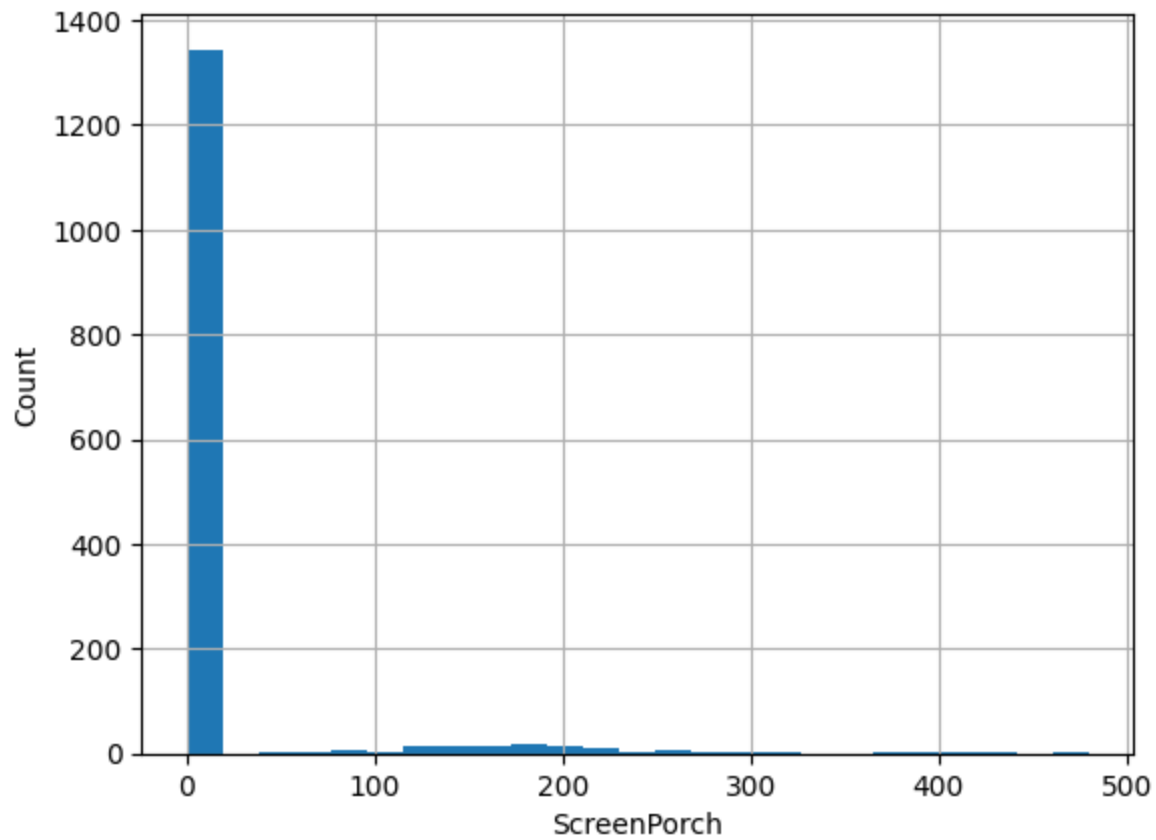
OpenPorchSF

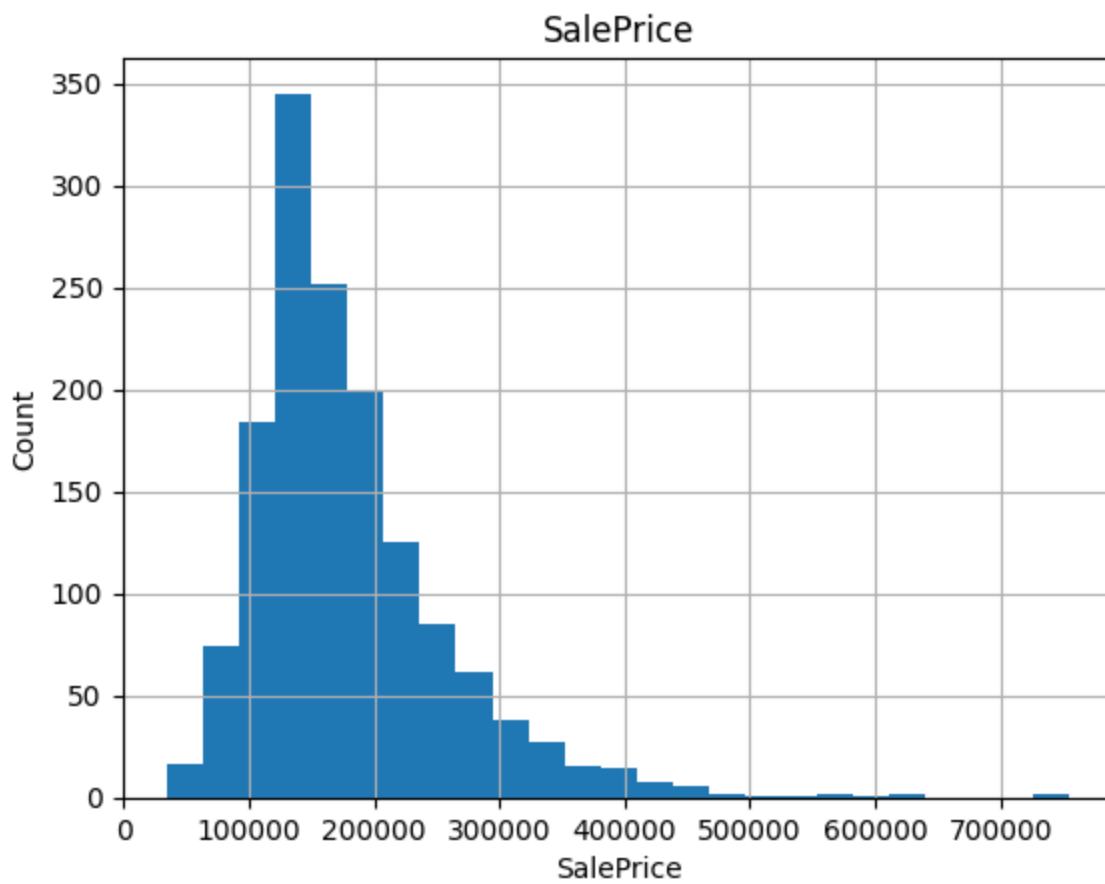


EnclosedPorch



ScreenPorch

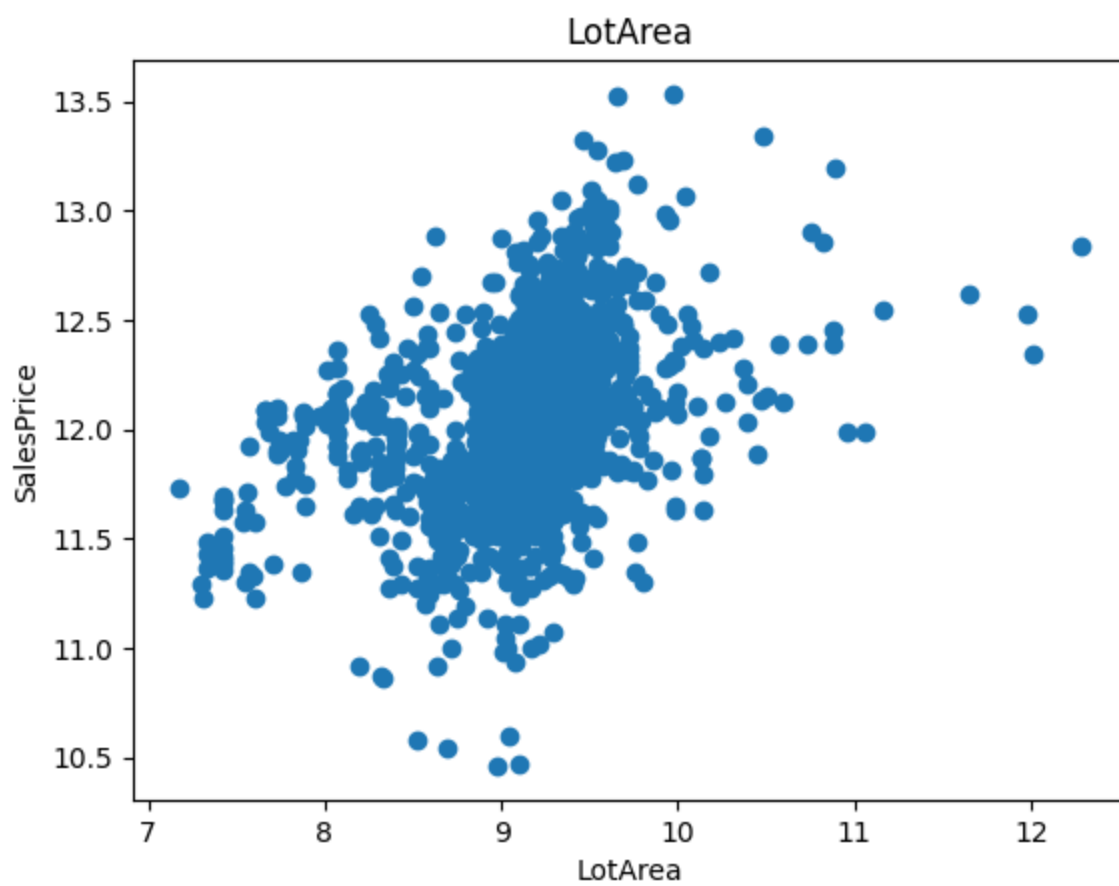
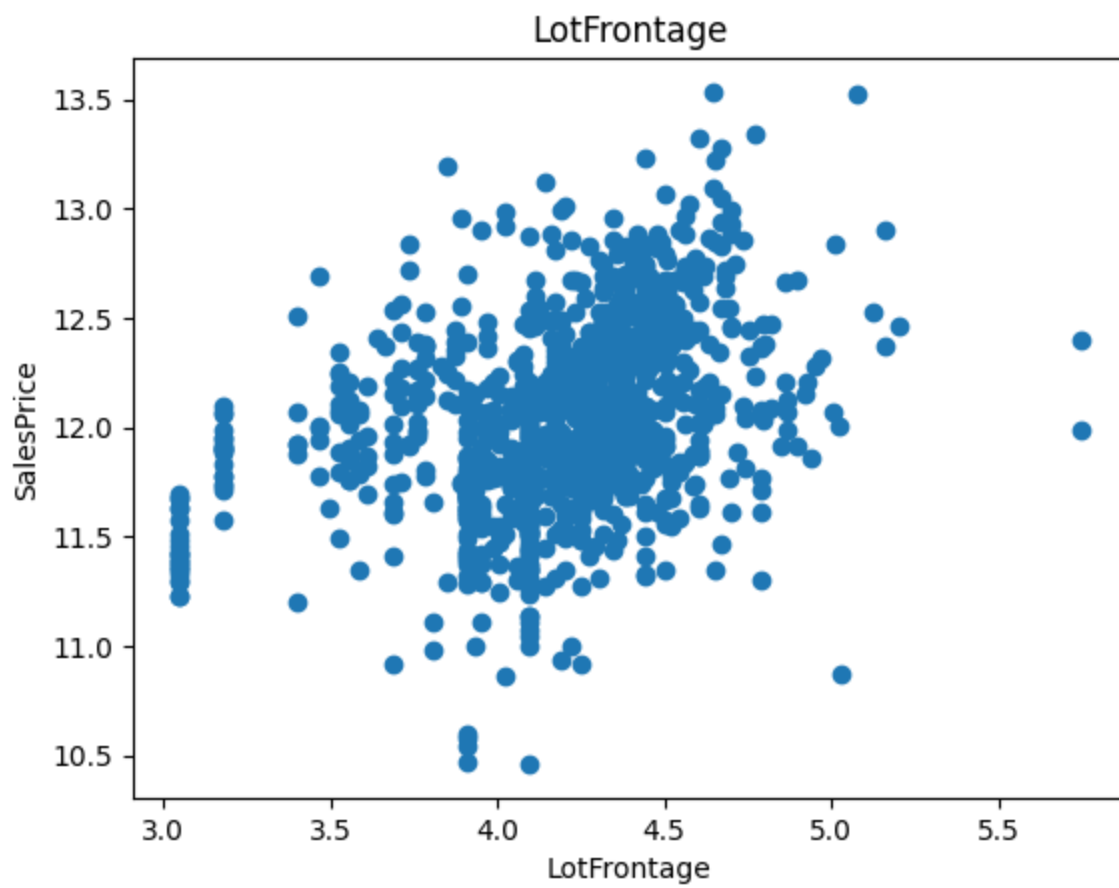




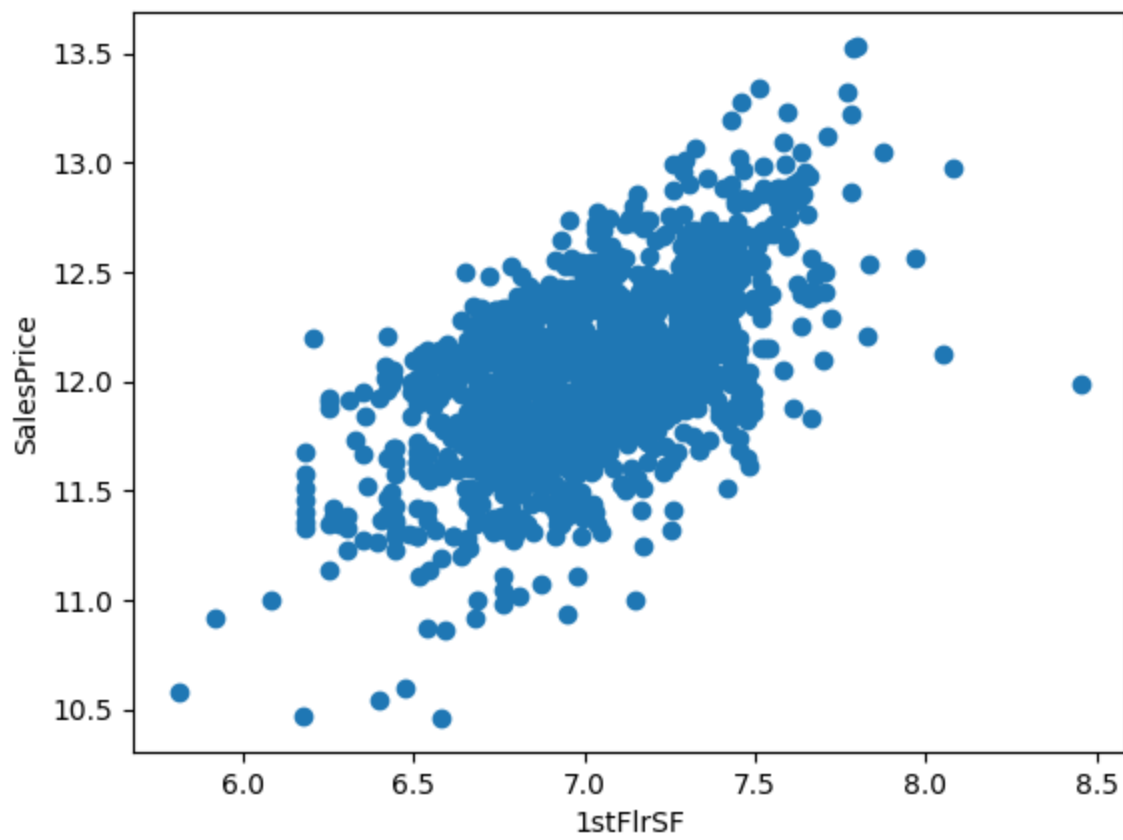
When analyzing the data turn non-gaussian distribution data into gaussian data.

```
In [75]: # we will now use logarithmic transformation

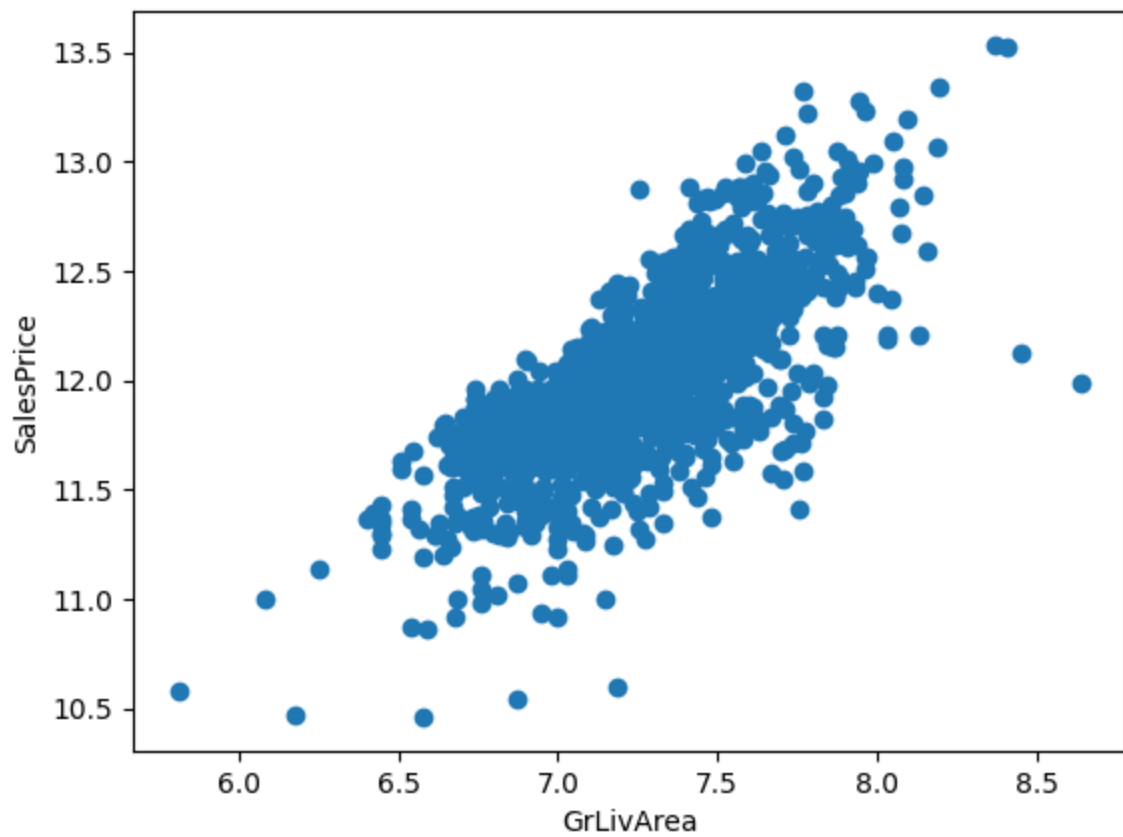
for feature in continuous_features:
    data=dataset.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data['SalePrice']=np.log(data['SalePrice'])
        plt.scatter(data[feature],data['SalePrice'])
        plt.xlabel(feature)
    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

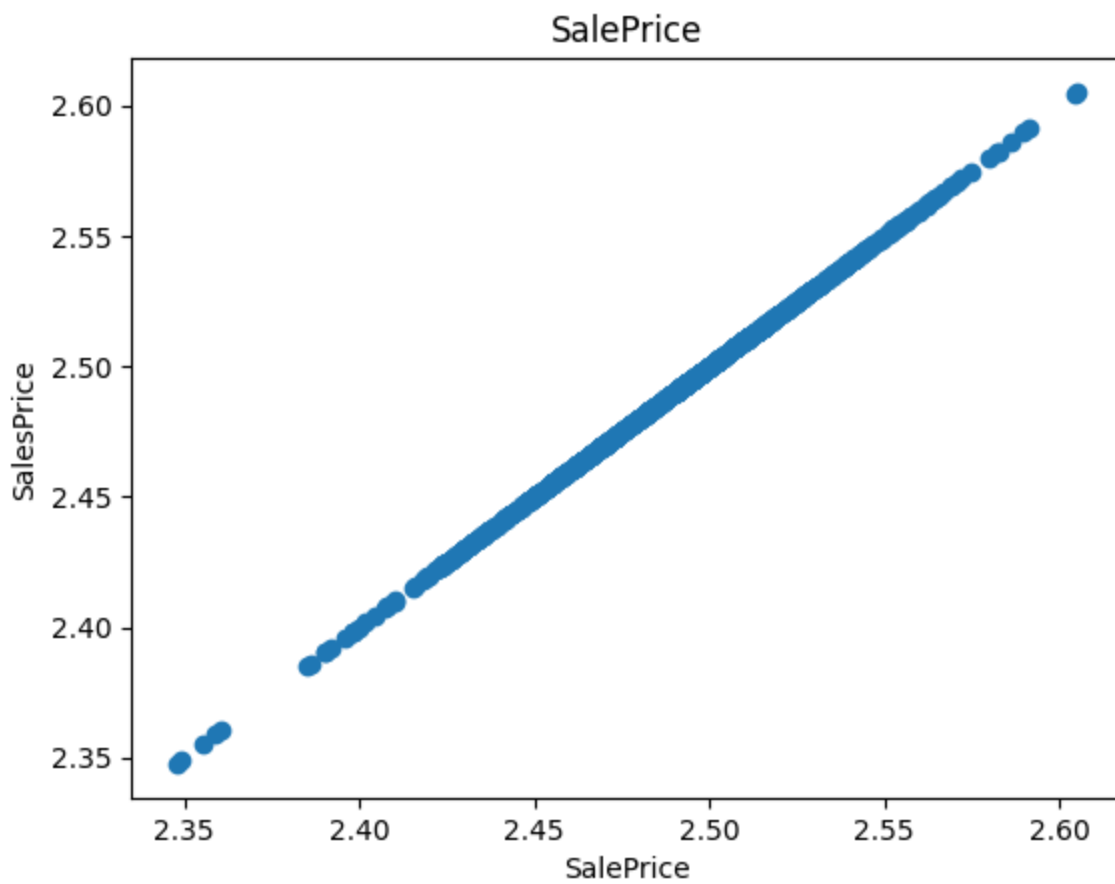


1stFlrSF



GrLivArea

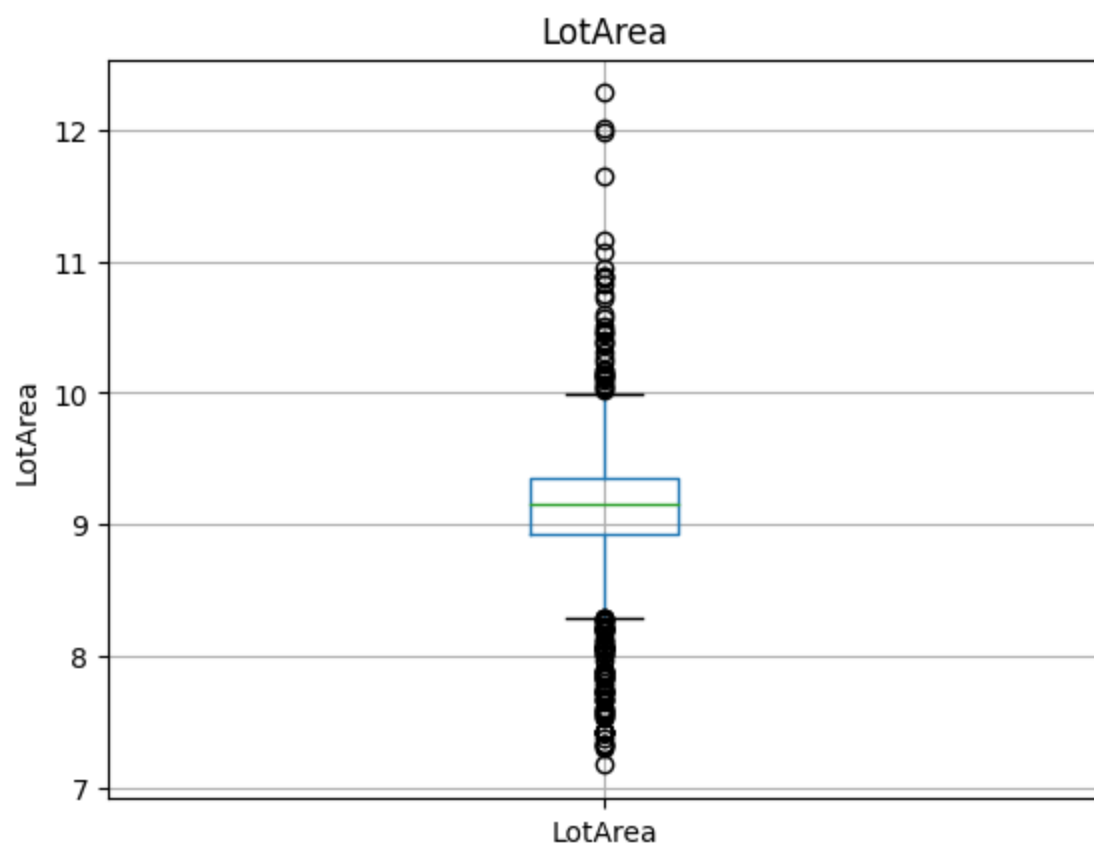
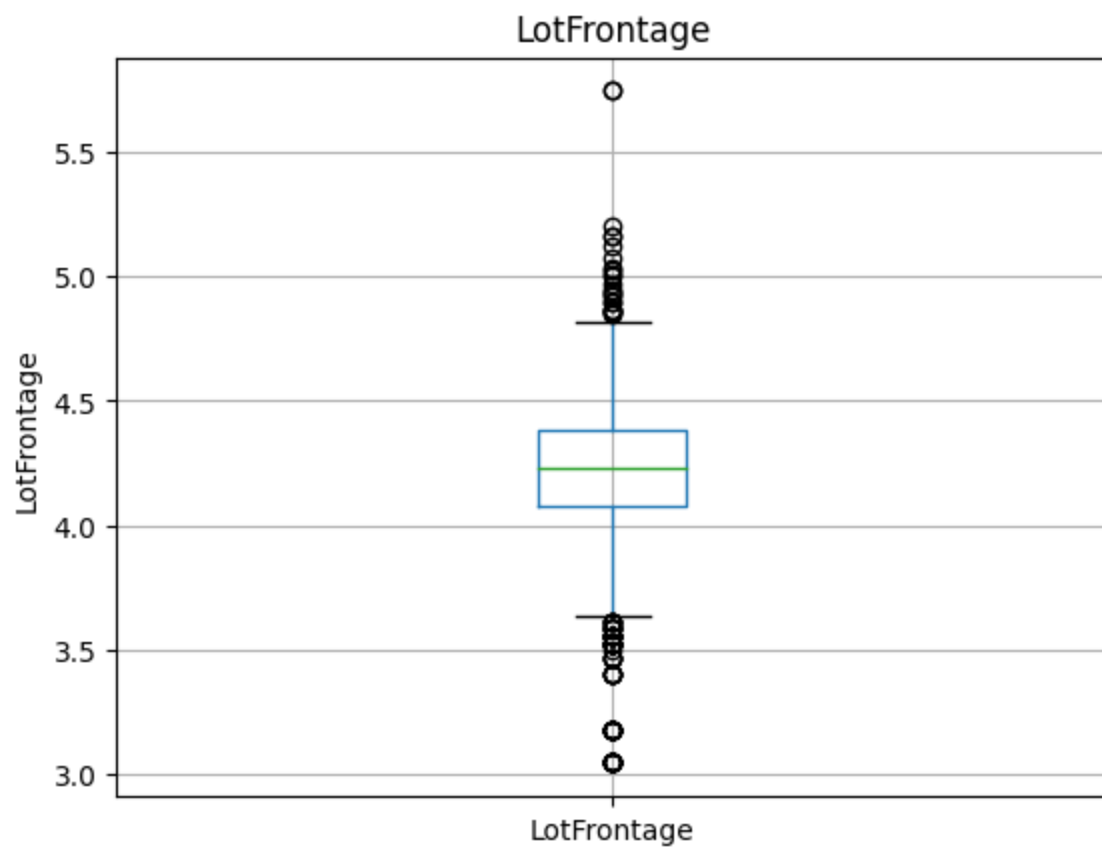


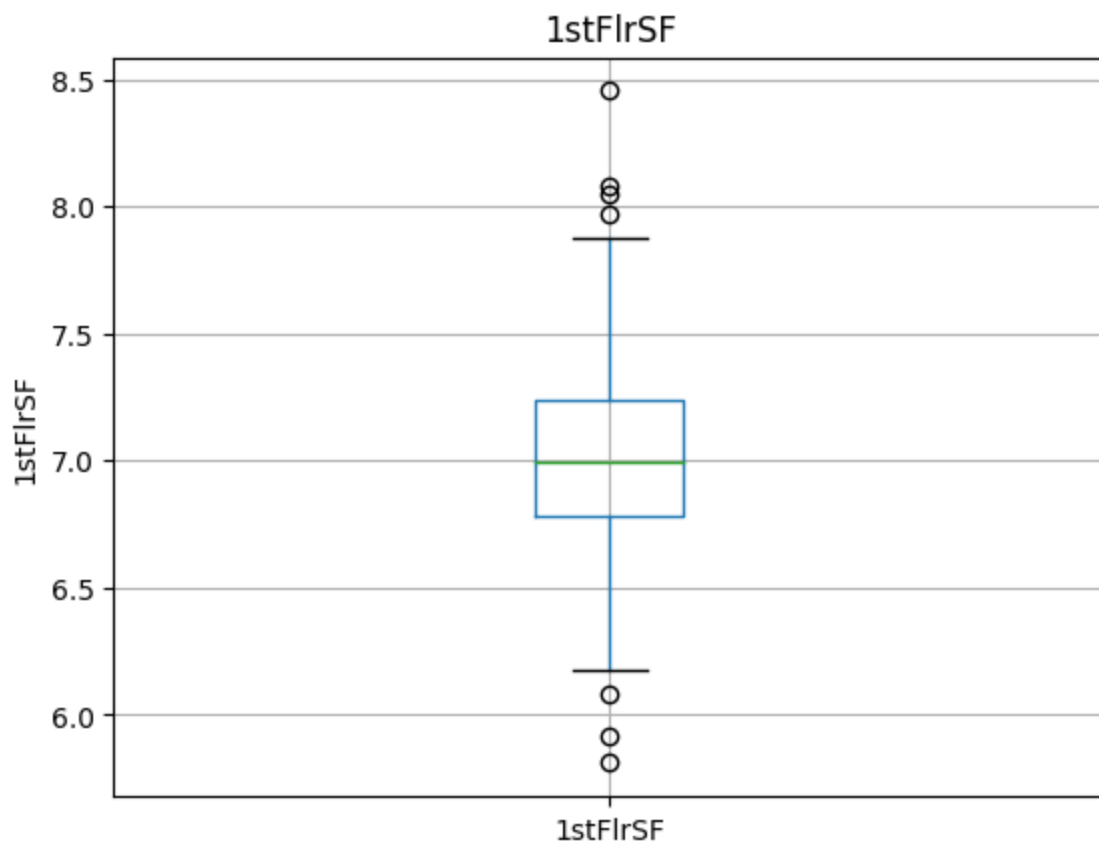


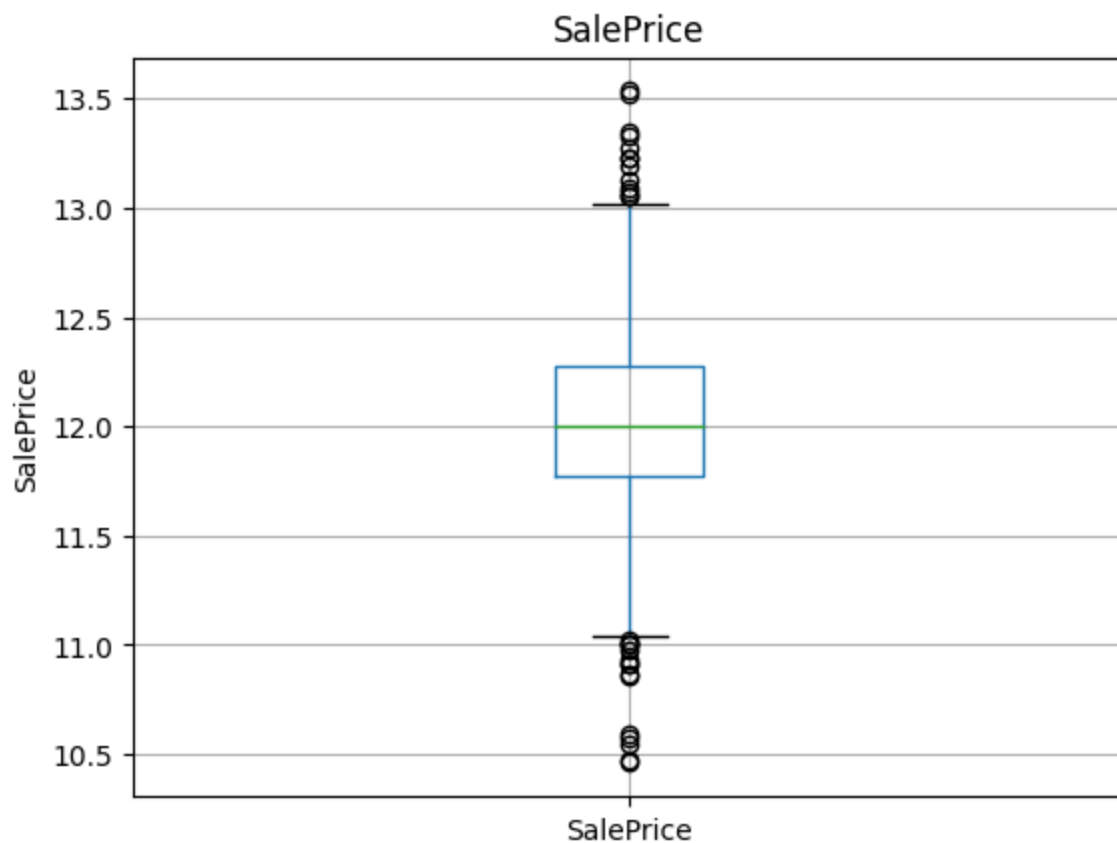
Outliers

particular values which are very high or low// out of line to find this we can use boxplot.

```
In [101... for feature in continuous_features:
    data=dataset.copy()
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        data.boxplot(column=feature)
        plt.ylabel(feature)
        plt.title(feature)
        plt.show()
```







Categorical variables

```
In [84]: categorical_features=[feature for feature in data.columns if data[feature].dtypes=='O']
categorical_features
```

```
Out[84]: ['MSZoning',
'Street',
'Alley',
'LotShape',
'LandContour',
'Utilities',
'LotConfig',
'LandSlope',
'Neighborhood',
'Condition1',
'Condition2',
'BldgType',
'HouseStyle',
'RoofStyle',
'RoofMatl',
'Exterior1st',
'Exterior2nd',
'MasVnrType',
'ExterQual',
'ExterCond',
'Foundation',
'BsmtQual',
'BsmtCond',
'BsmtExposure',
'BsmtFinType1',
'BsmtFinType2',
'Heating',
'HeatingQC',
'CentralAir',
'Electrical',
```

```

'KitchenQual',
'Functional',
'FireplaceQu',
'GarageType',
'GarageFinish',
'GarageQual',
'GarageCond',
'PavedDrive',
'PoolQC',
'Fence',
'MiscFeature',
'SaleType',
'SaleCondition']

```

```
In [85]: dataset[categorical_features].head()
```

```
Out[85]:
```

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1
0	RL	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm
1	RL	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr
2	RL	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	CollgCr	Norm
3	RL	Pave	NaN	IR1	Lvl	AllPub	Corner	Gtl	Crawfor	Norm
4	RL	Pave	NaN	IR1	Lvl	AllPub	FR2	Gtl	NoRidge	Norm

```
In [87]: for feature in categorical_features:
          print("The feature is {} and number of categories are {}".format(feature, len(dataset
```

```

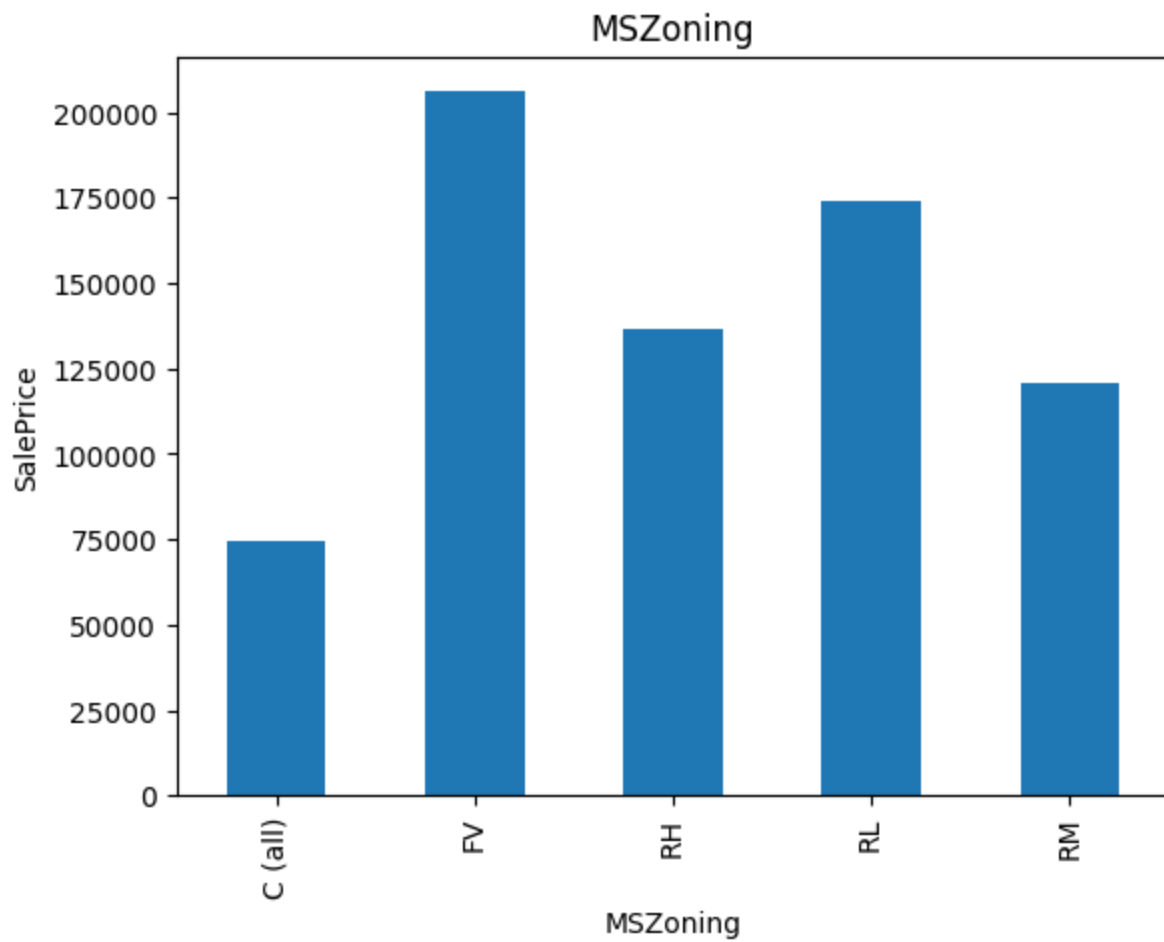
The feature is MSZoning and number of categories are 5
The feature is Street and number of categories are 2
The feature is Alley and number of categories are 3
The feature is LotShape and number of categories are 4
The feature is LandContour and number of categories are 4
The feature is Utilities and number of categories are 2
The feature is LotConfig and number of categories are 5
The feature is LandSlope and number of categories are 3
The feature is Neighborhood and number of categories are 25
The feature is Condition1 and number of categories are 9
The feature is Condition2 and number of categories are 8
The feature is BldgType and number of categories are 5
The feature is HouseStyle and number of categories are 8
The feature is RoofStyle and number of categories are 6
The feature is RoofMatl and number of categories are 8
The feature is Exterior1st and number of categories are 15
The feature is Exterior2nd and number of categories are 16
The feature is MasVnrType and number of categories are 5
The feature is ExterQual and number of categories are 4
The feature is ExterCond and number of categories are 5
The feature is Foundation and number of categories are 6
The feature is BsmtQual and number of categories are 5
The feature is BsmtCond and number of categories are 5
The feature is BsmtExposure and number of categories are 5
The feature is BsmtFinType1 and number of categories are 7
The feature is BsmtFinType2 and number of categories are 7
The feature is Heating and number of categories are 6
The feature is HeatingQC and number of categories are 5
The feature is CentralAir and number of categories are 2
The feature is Electrical and number of categories are 6
The feature is KitchenQual and number of categories are 4
The feature is Functional and number of categories are 7
The feature is FireplaceQu and number of categories are 6
The feature is GarageType and number of categories are 7

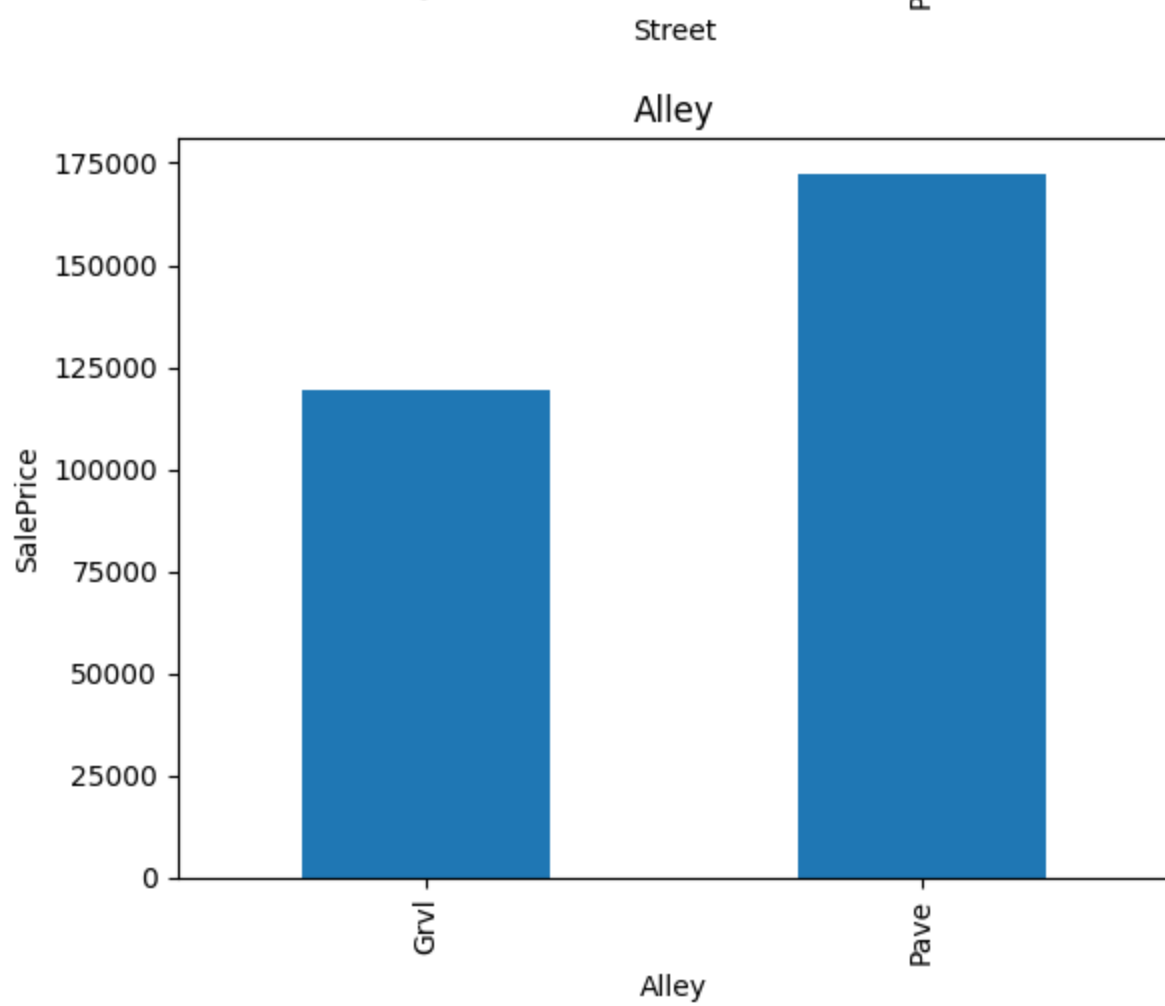
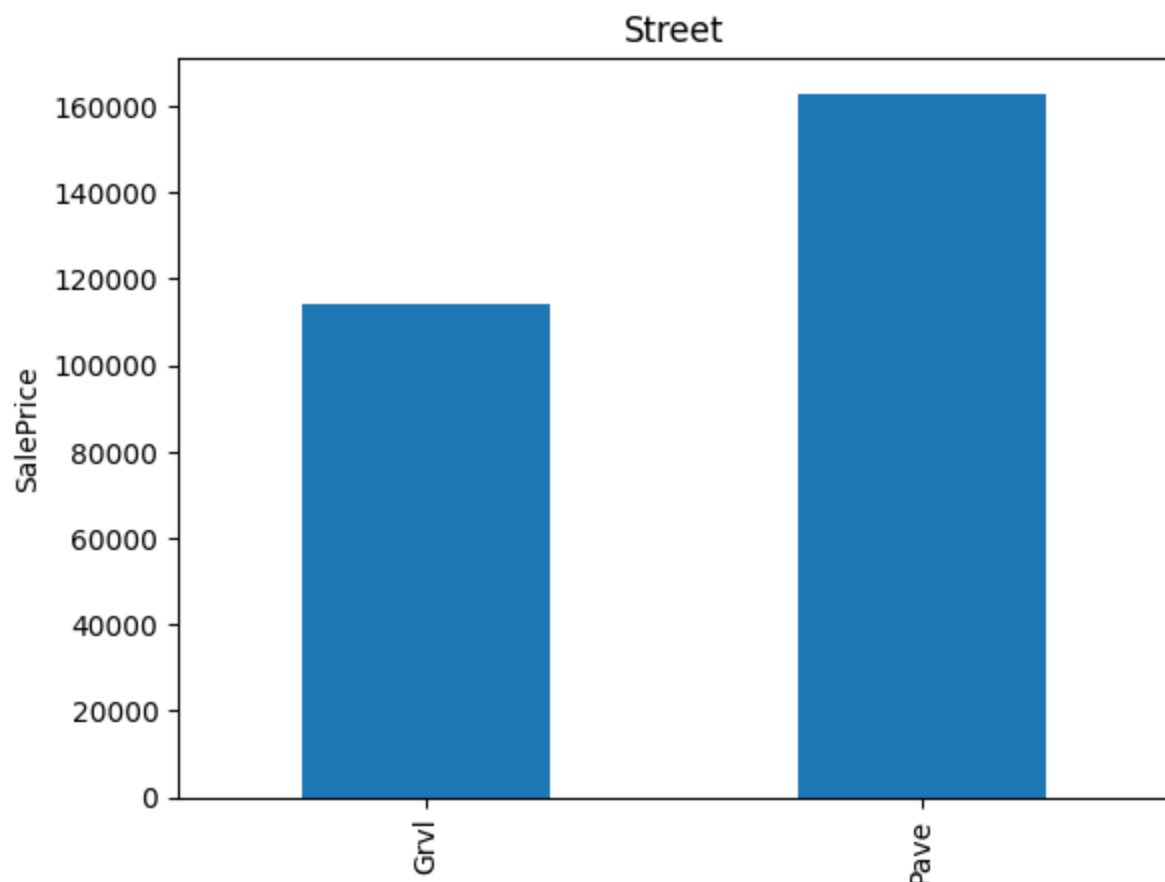
```

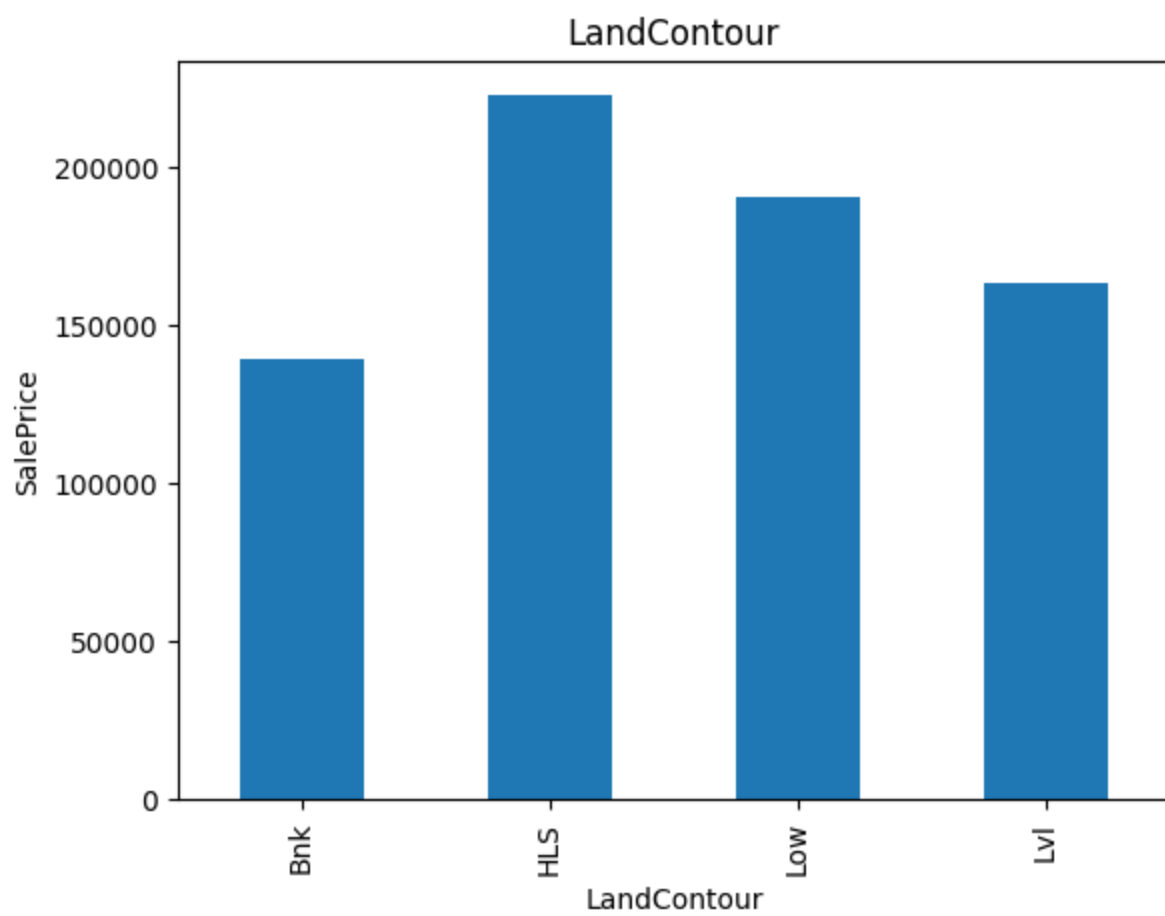
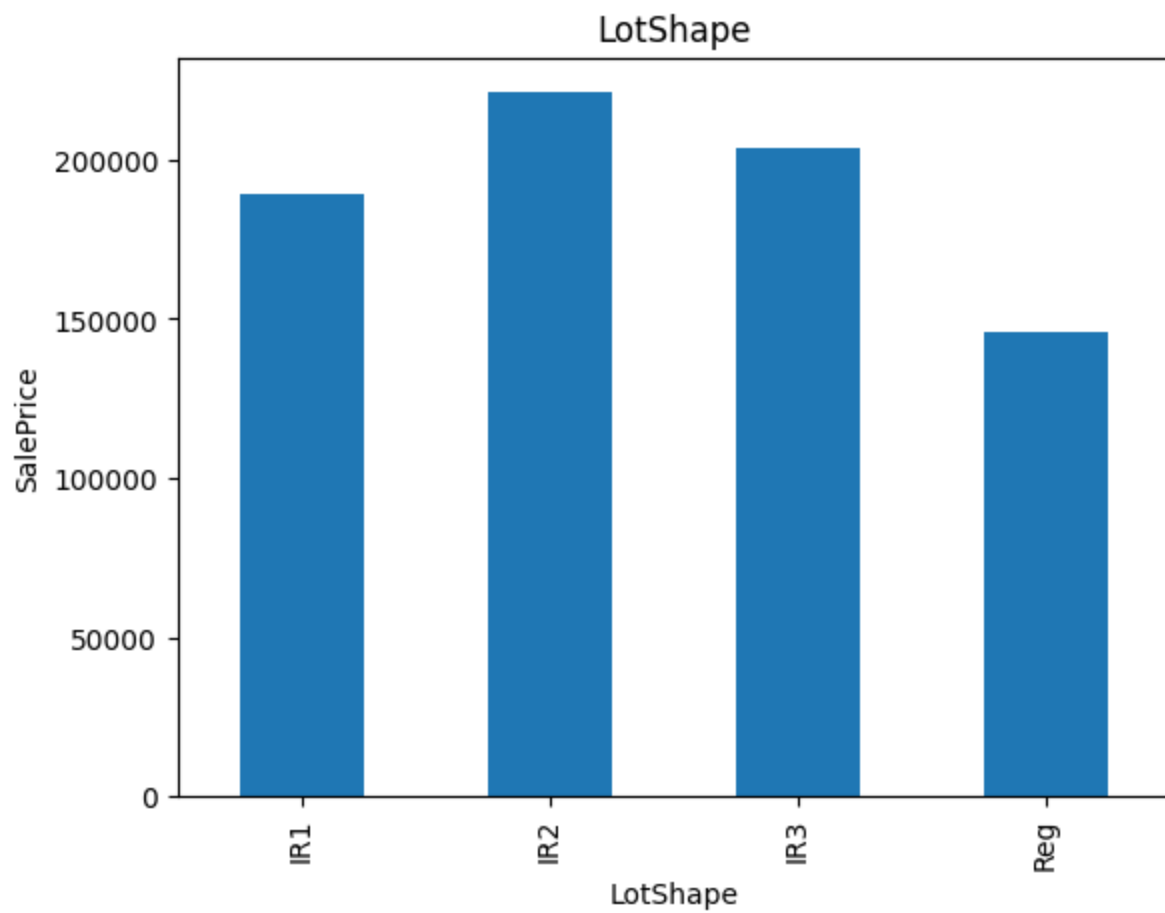
The feature is GarageFinish and number of categories are 4
The feature is GarageQual and number of categories are 6
The feature is GarageCond and number of categories are 6
The feature is PavedDrive and number of categories are 3
The feature is PoolQC and number of categories are 4
The feature is Fence and number of categories are 5
The feature is MiscFeature and number of categories are 5
The feature is SaleType and number of categories are 9
The feature is SaleCondition and number of categories are 6

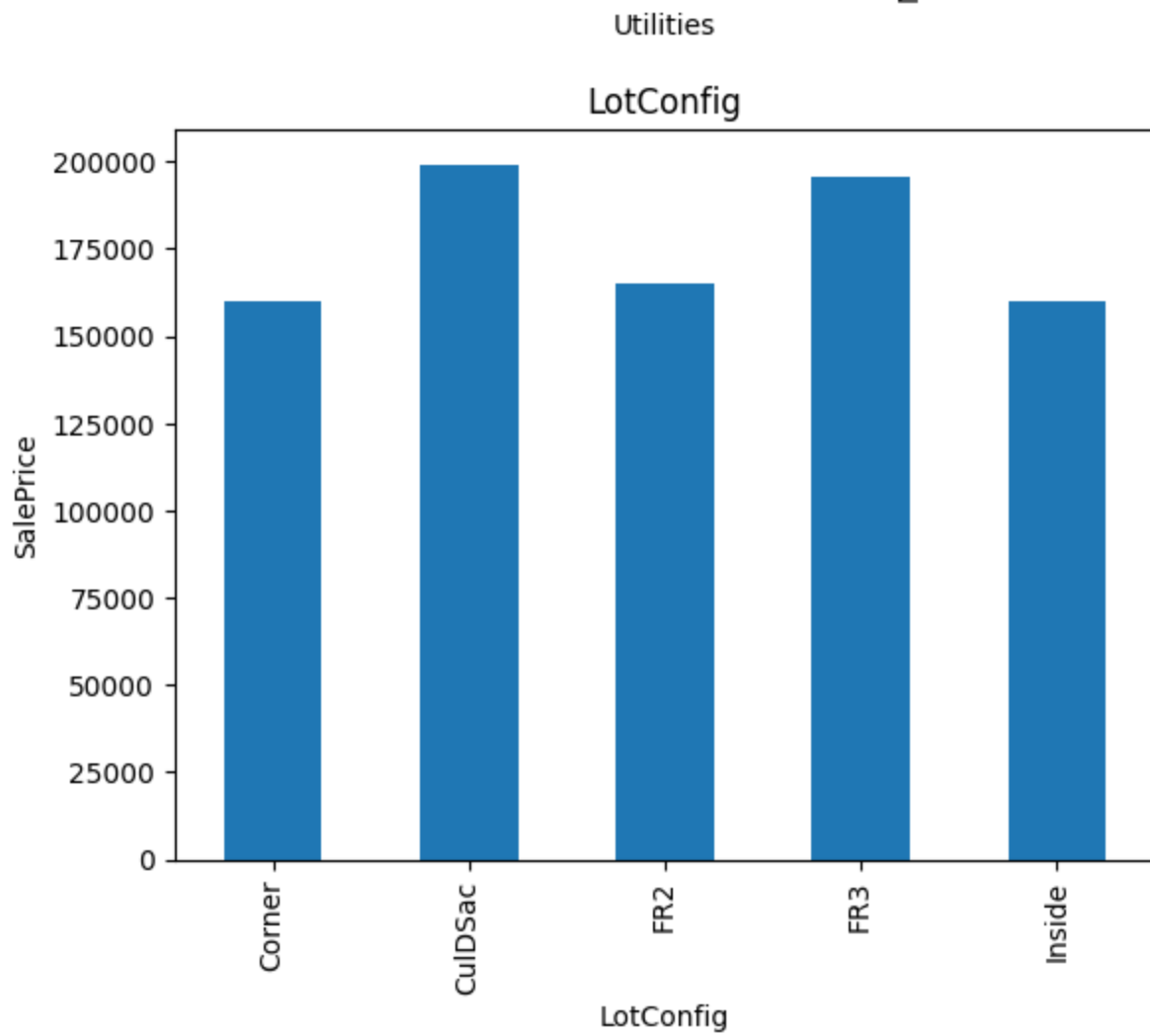
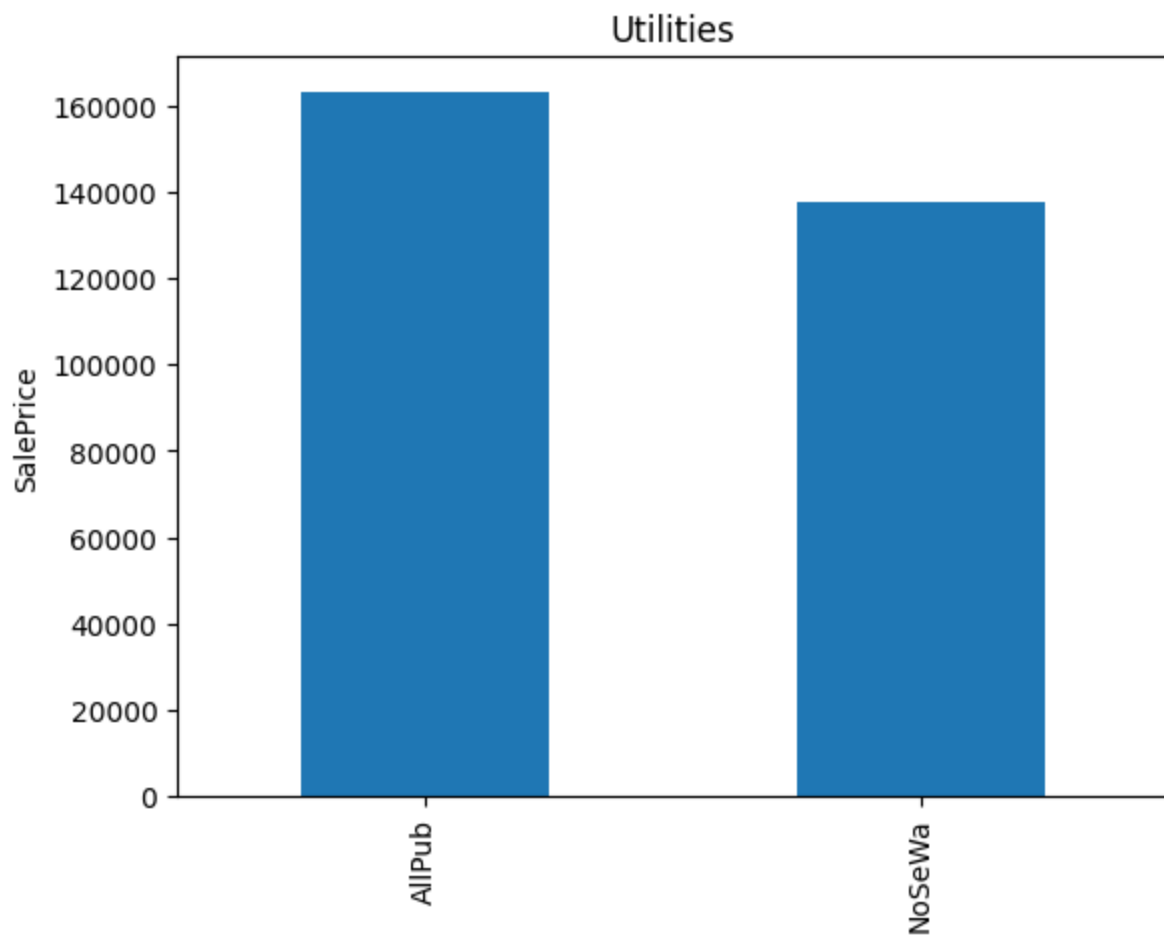
Now we will try to find out the relationship between categorical variables and dependent features

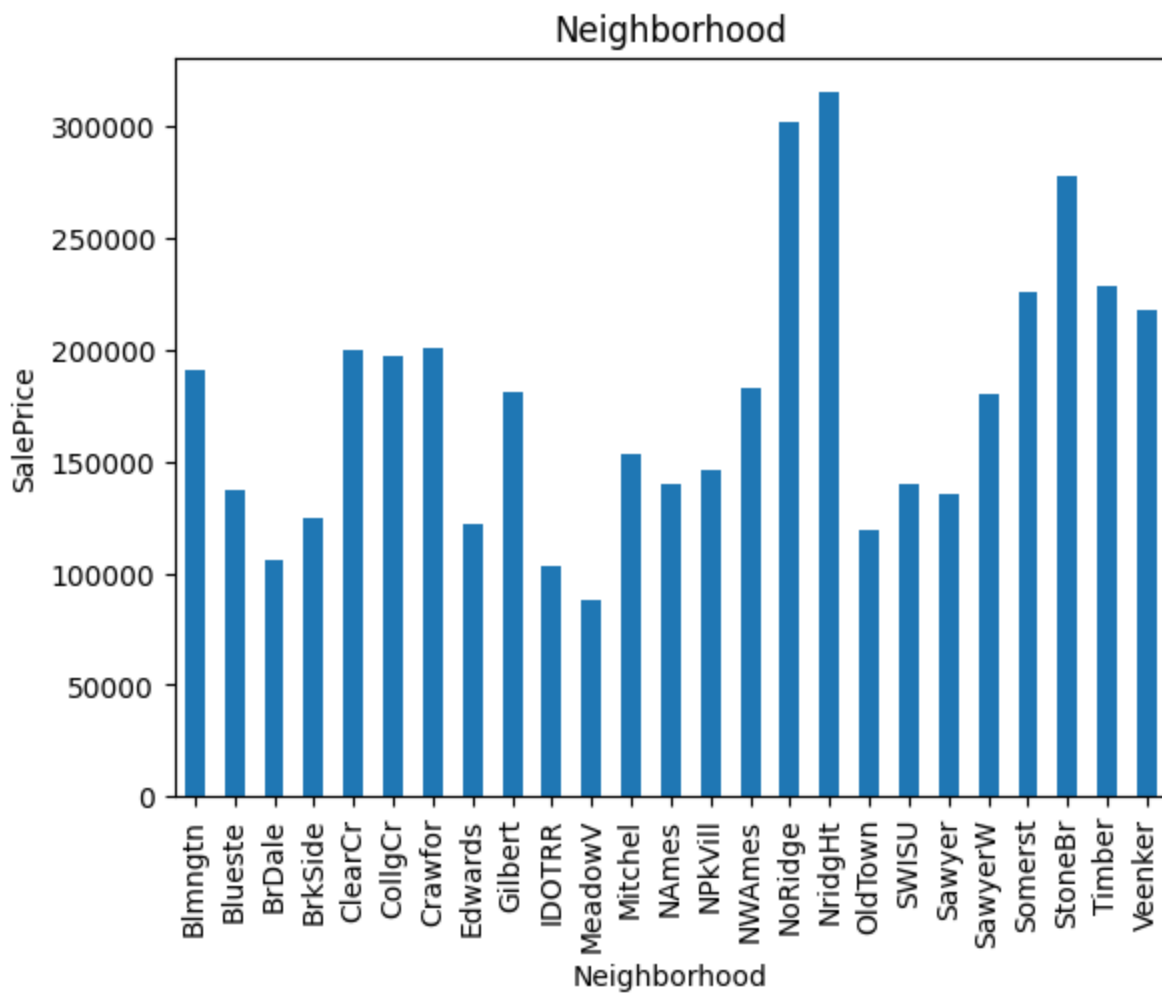
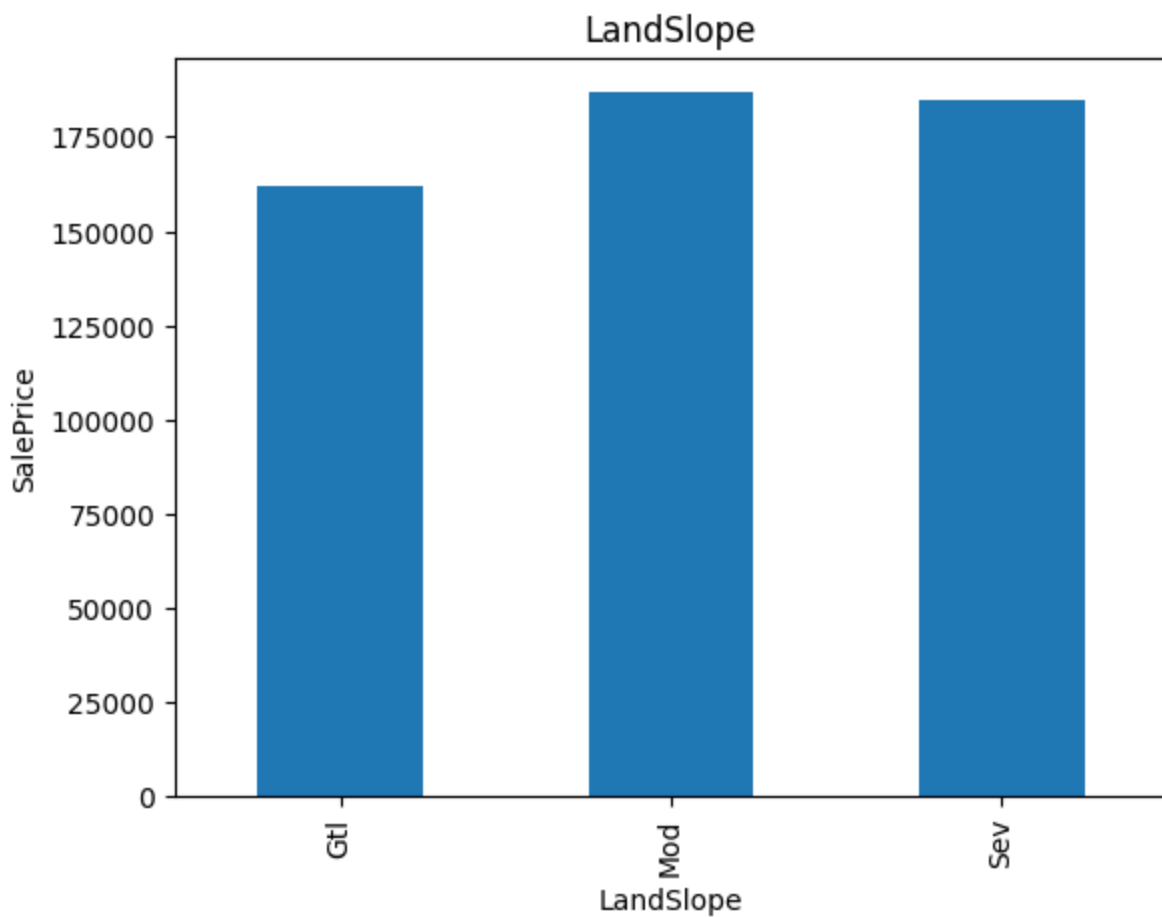
```
In [102... for feature in categorical_features:
    data=dataset.copy()
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```

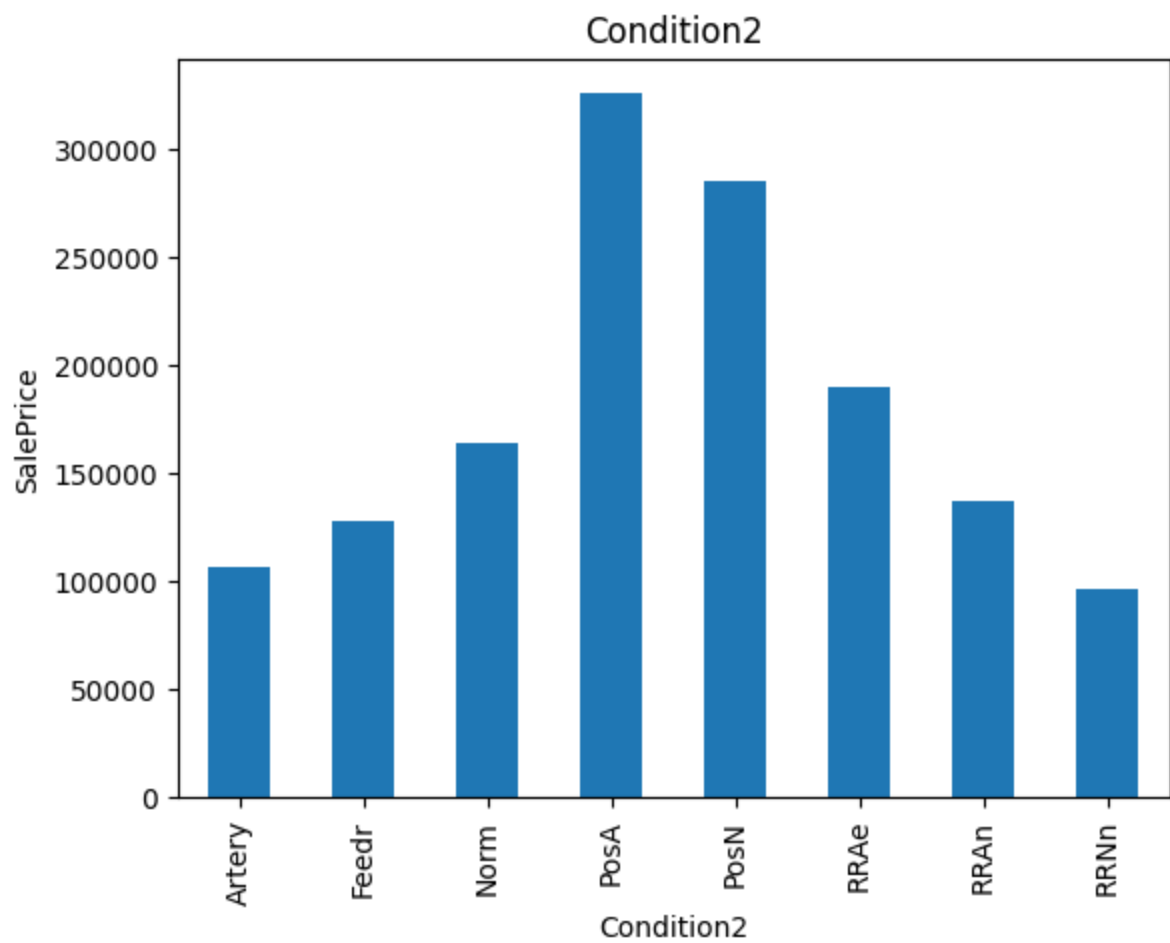
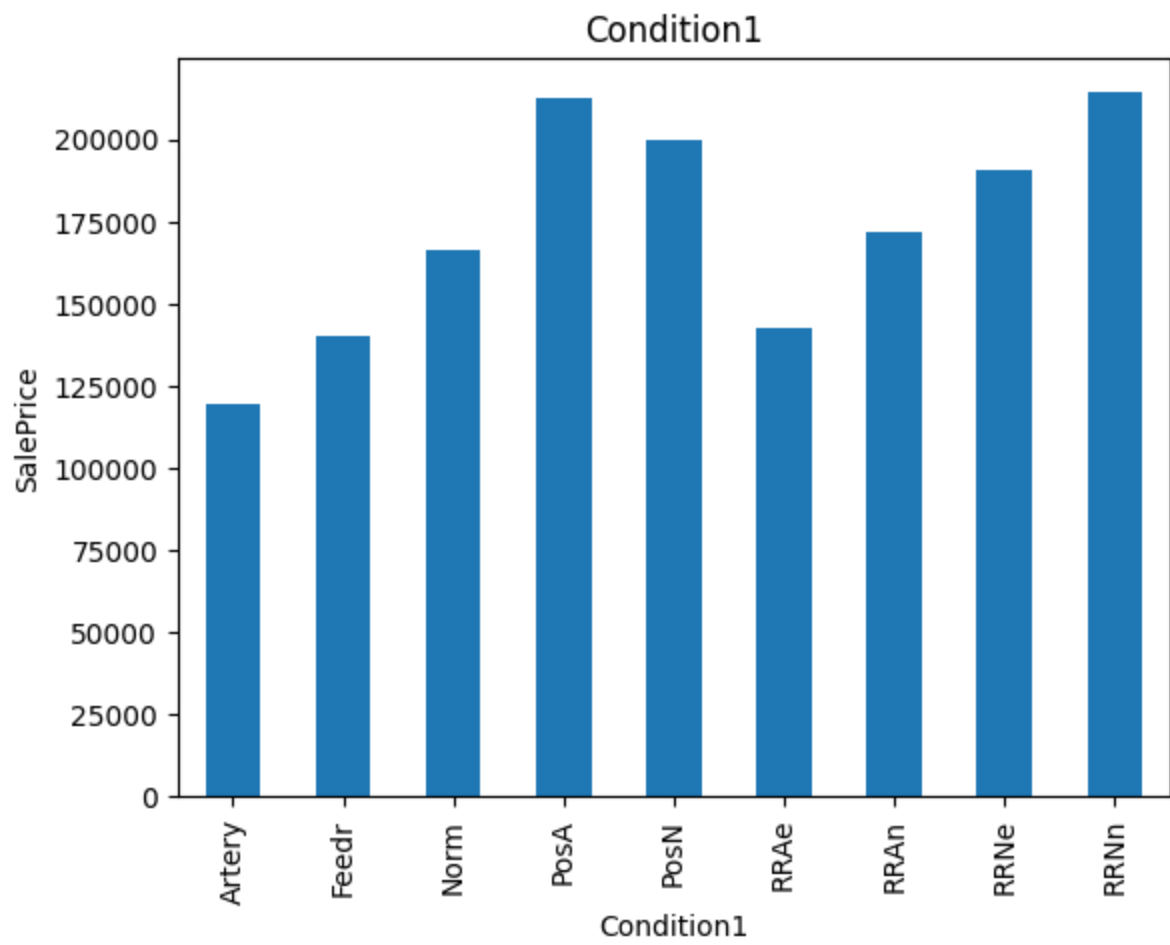


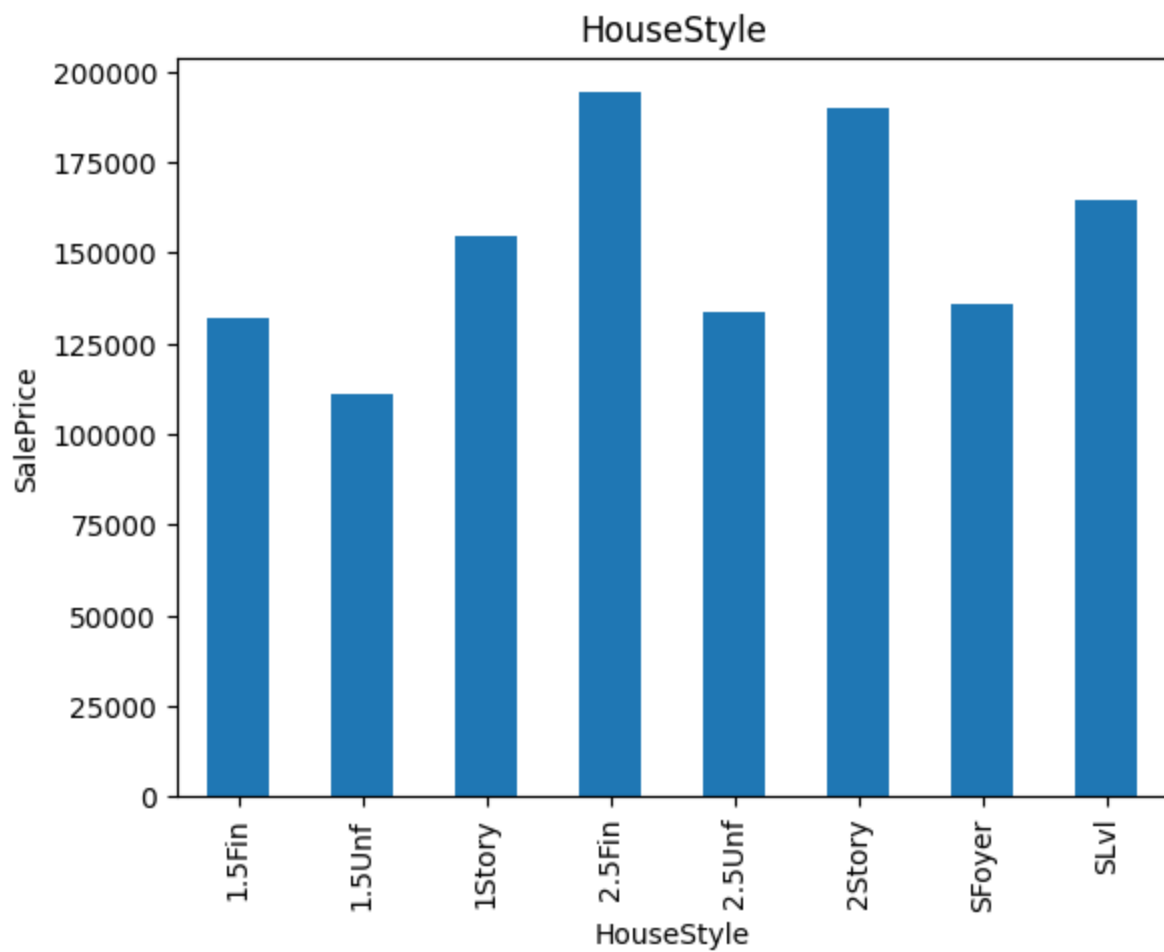
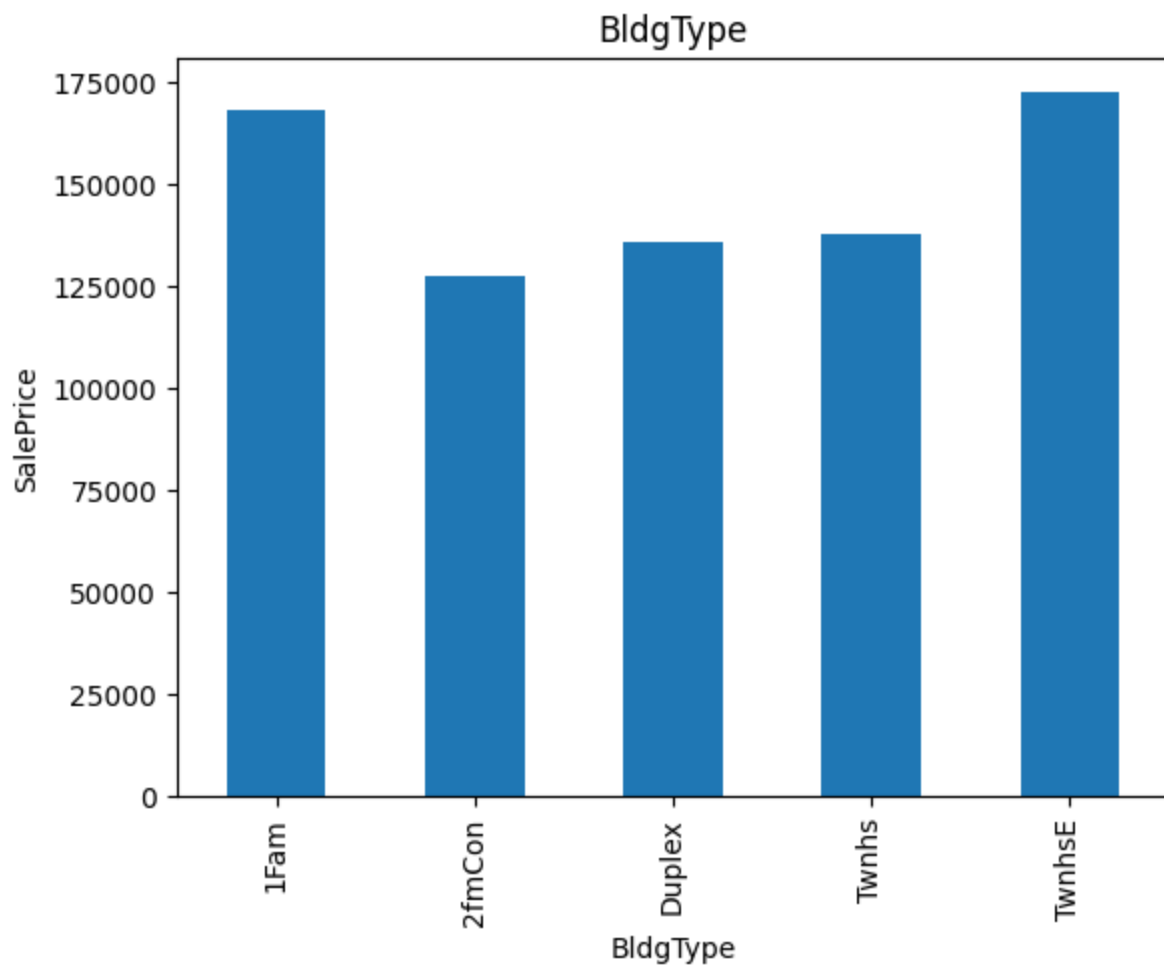


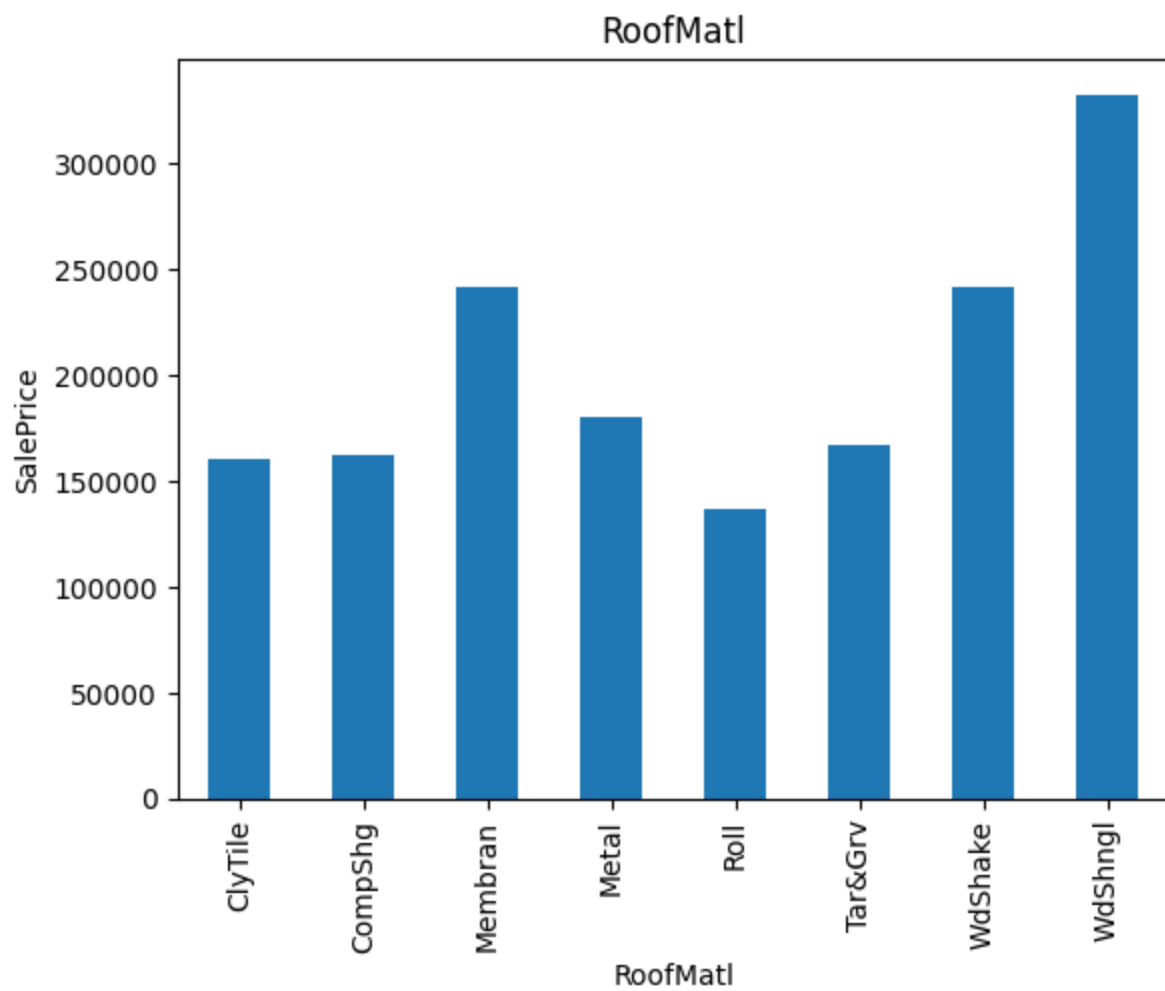
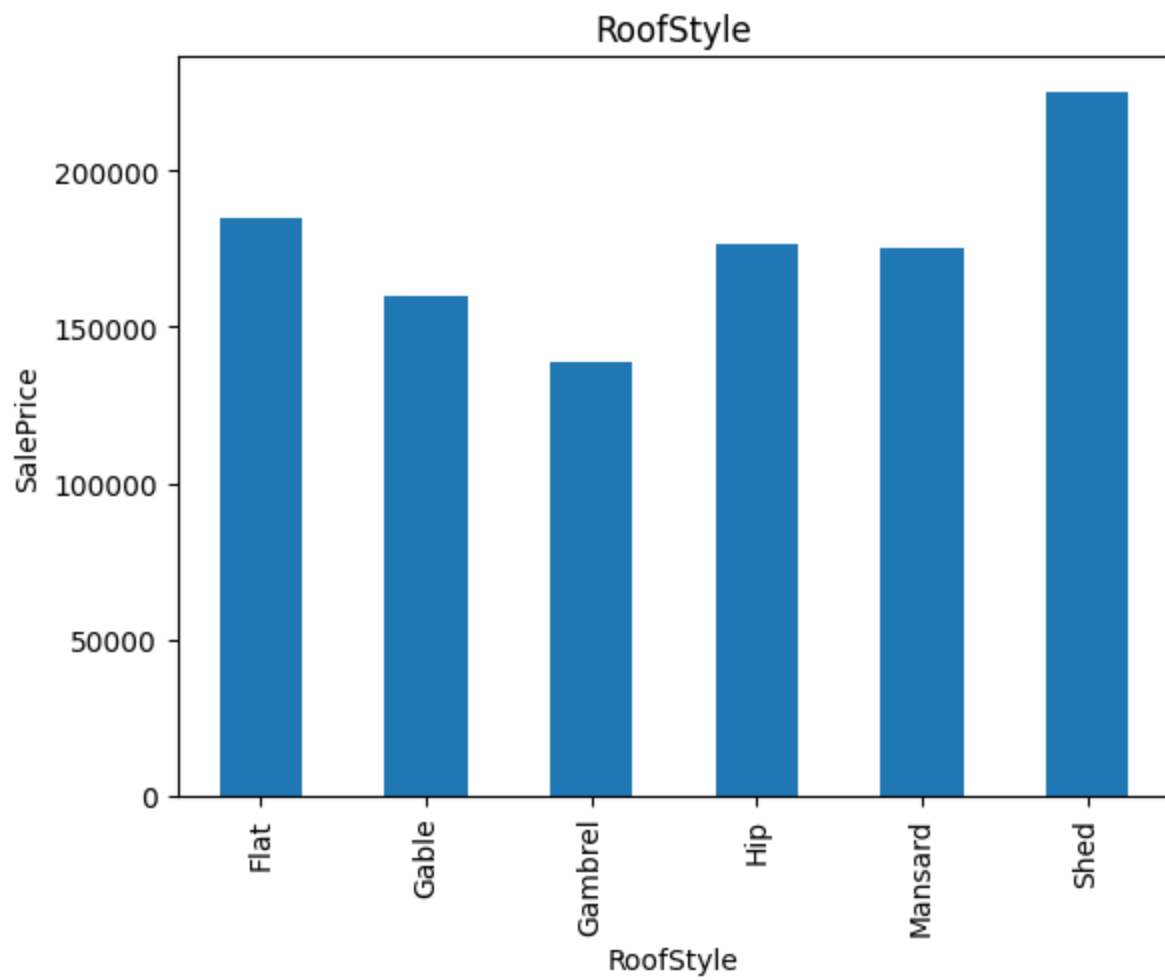


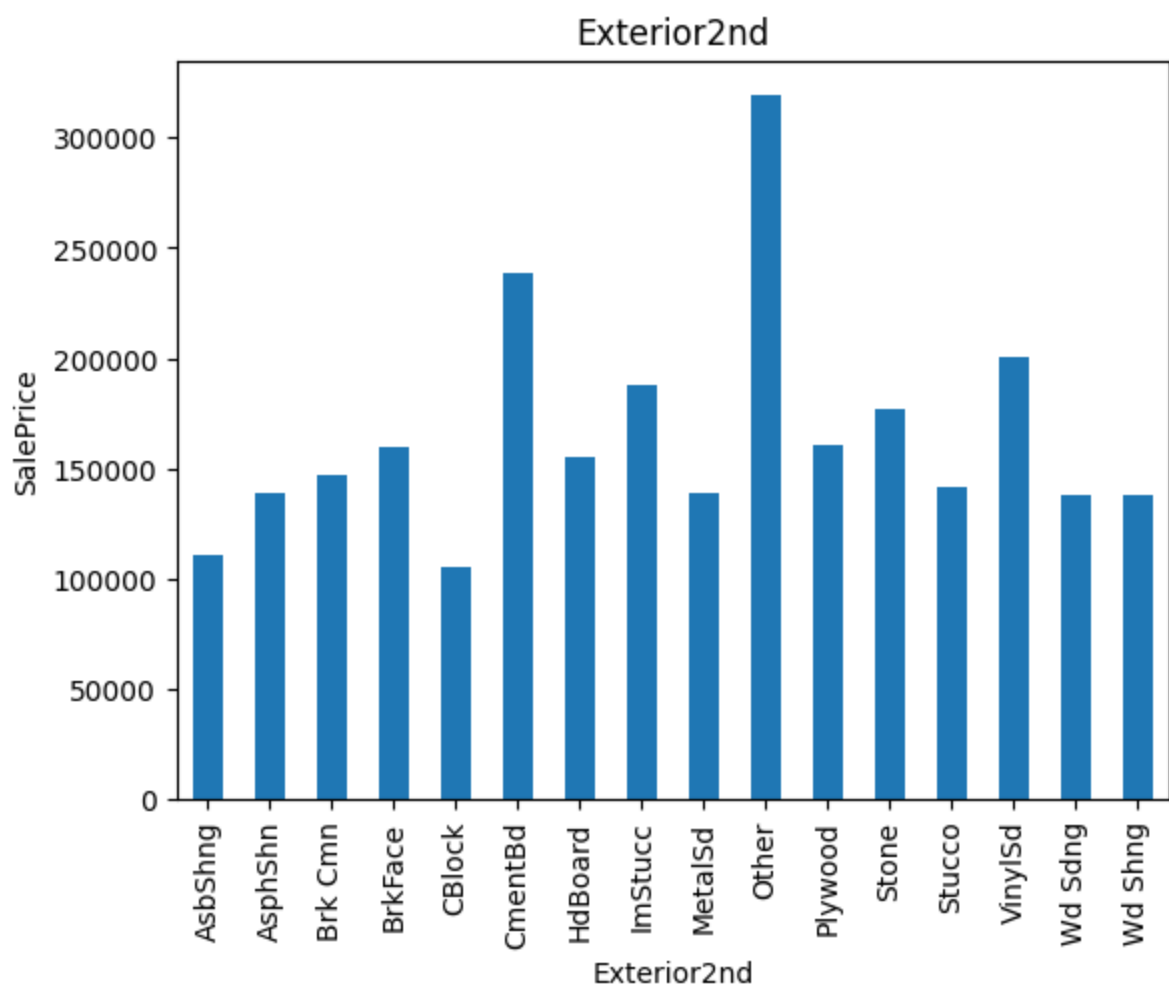
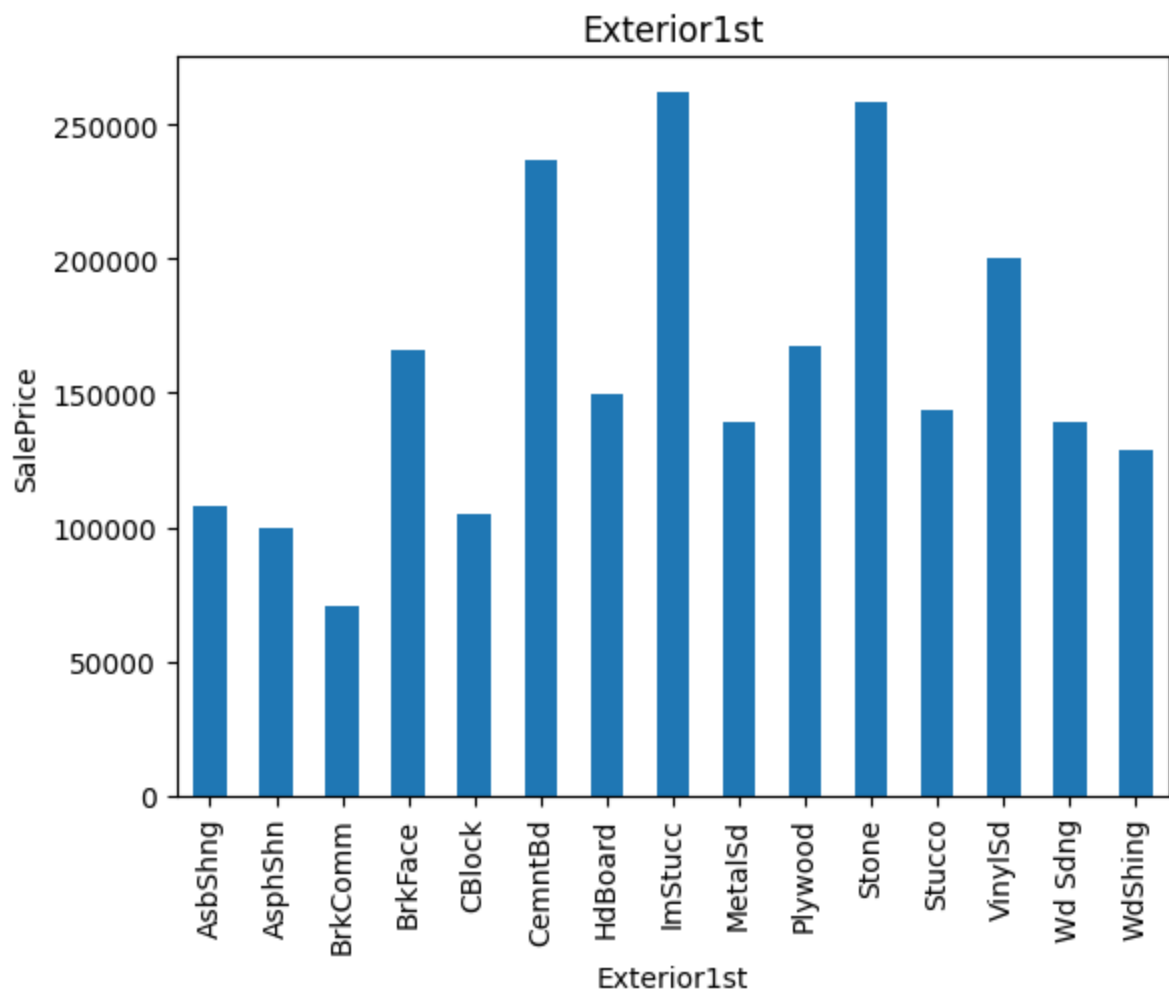


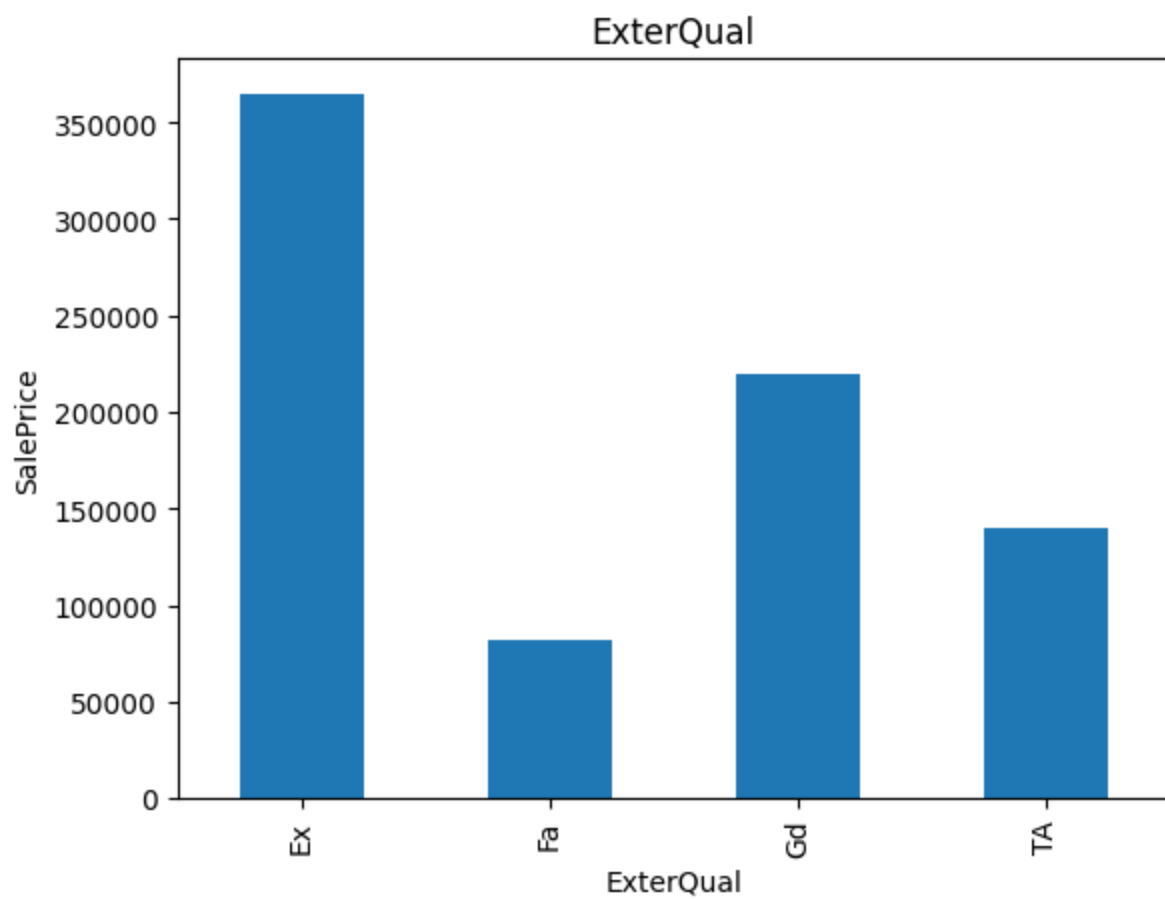
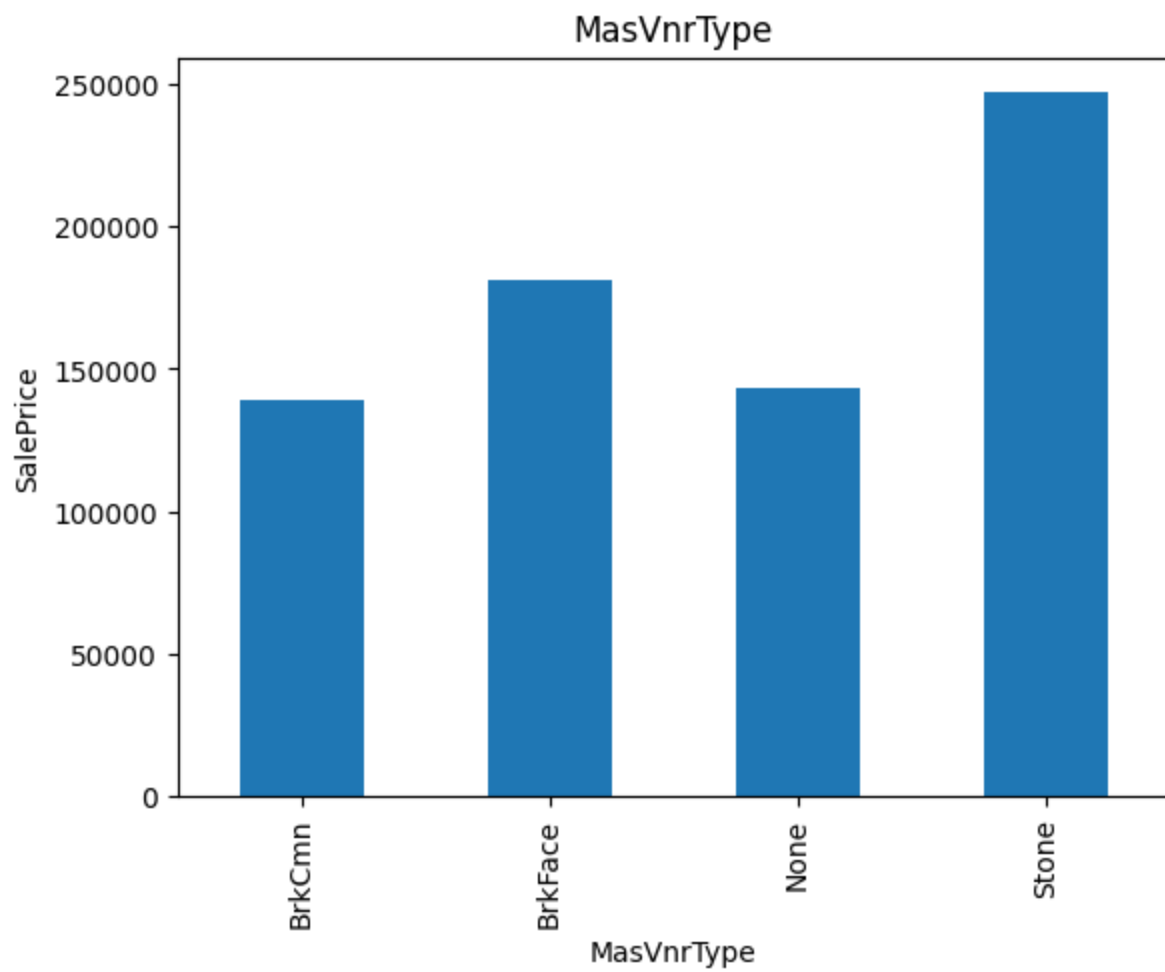


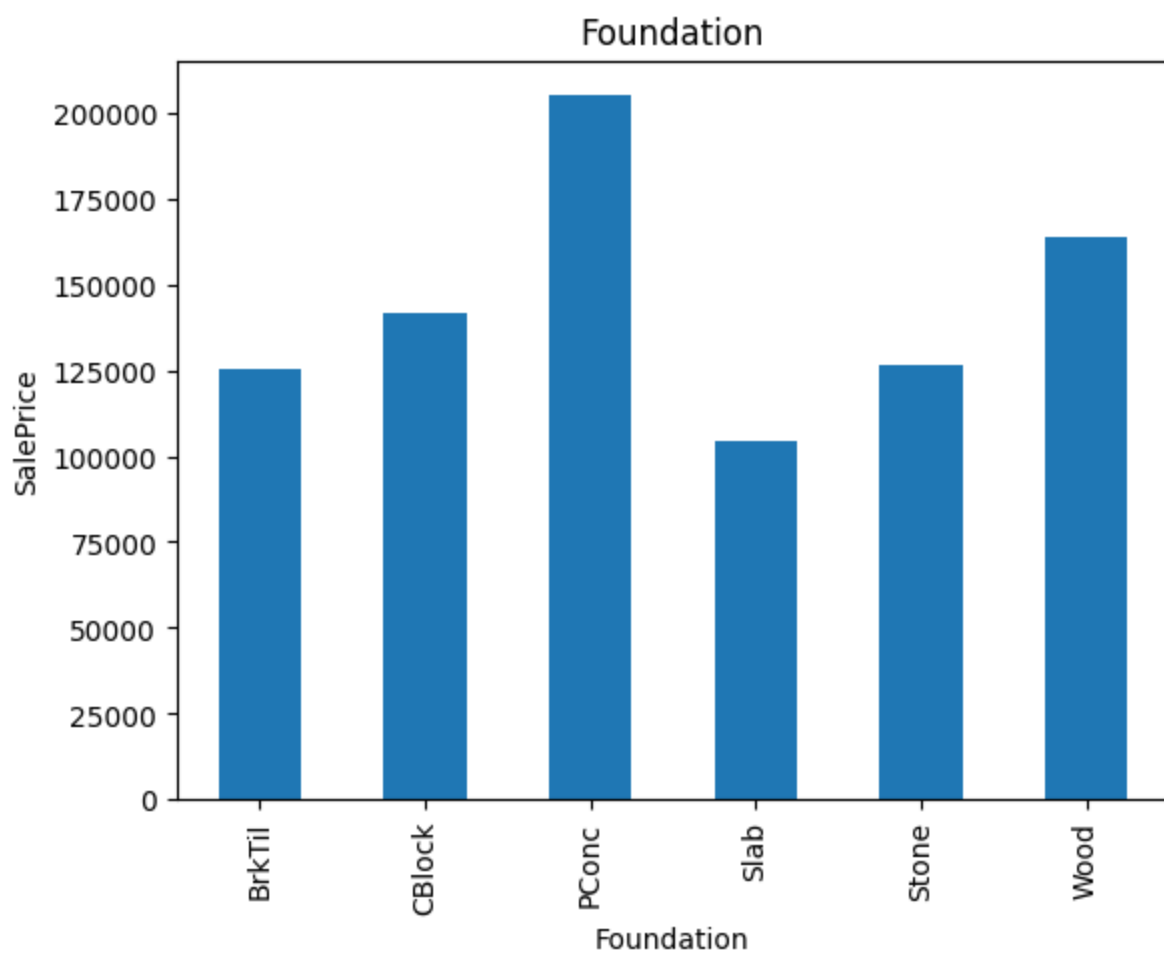
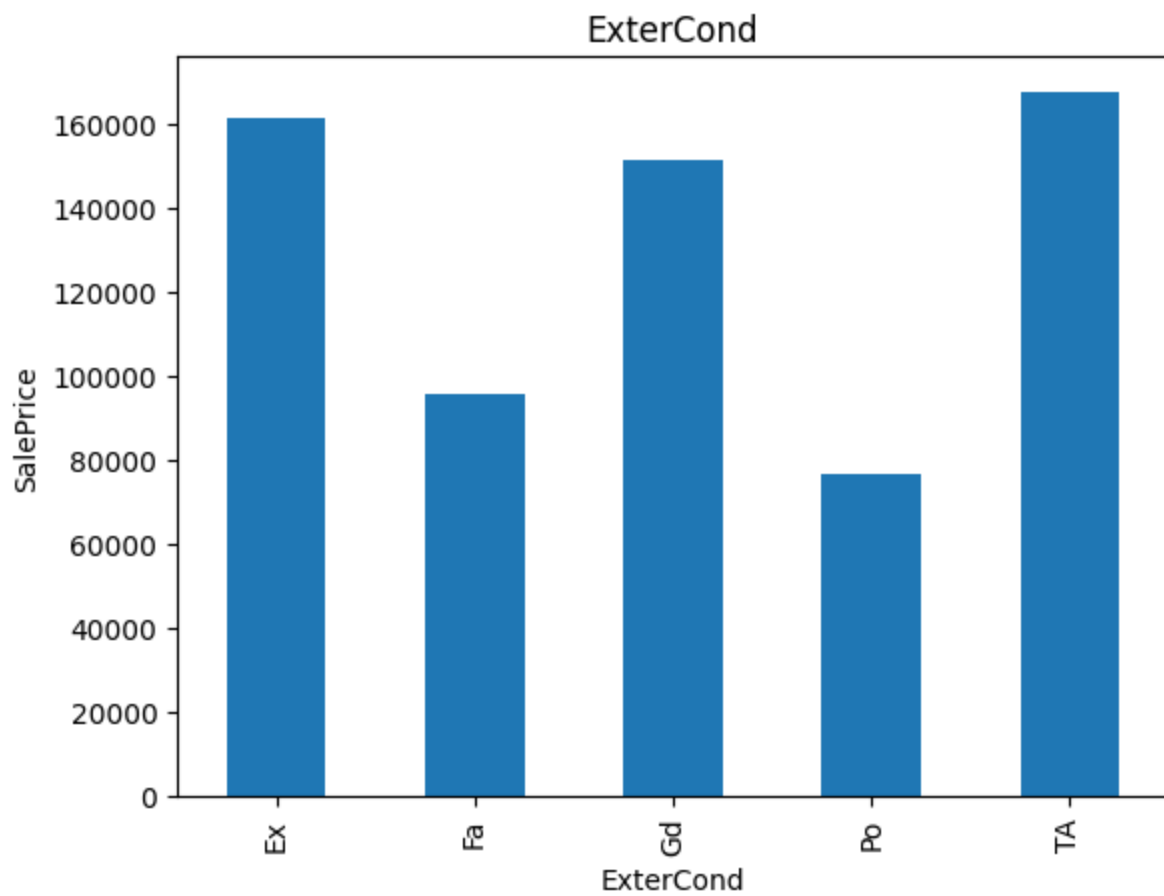


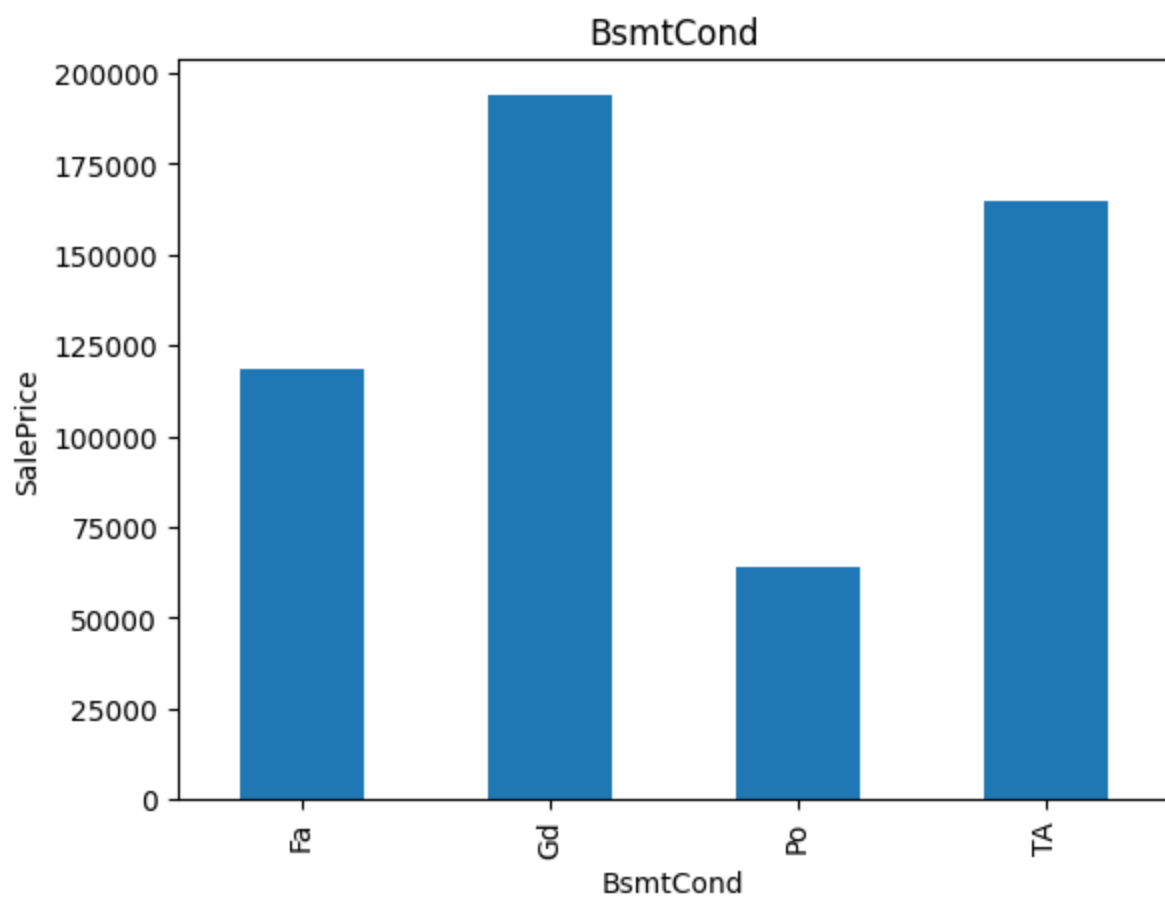
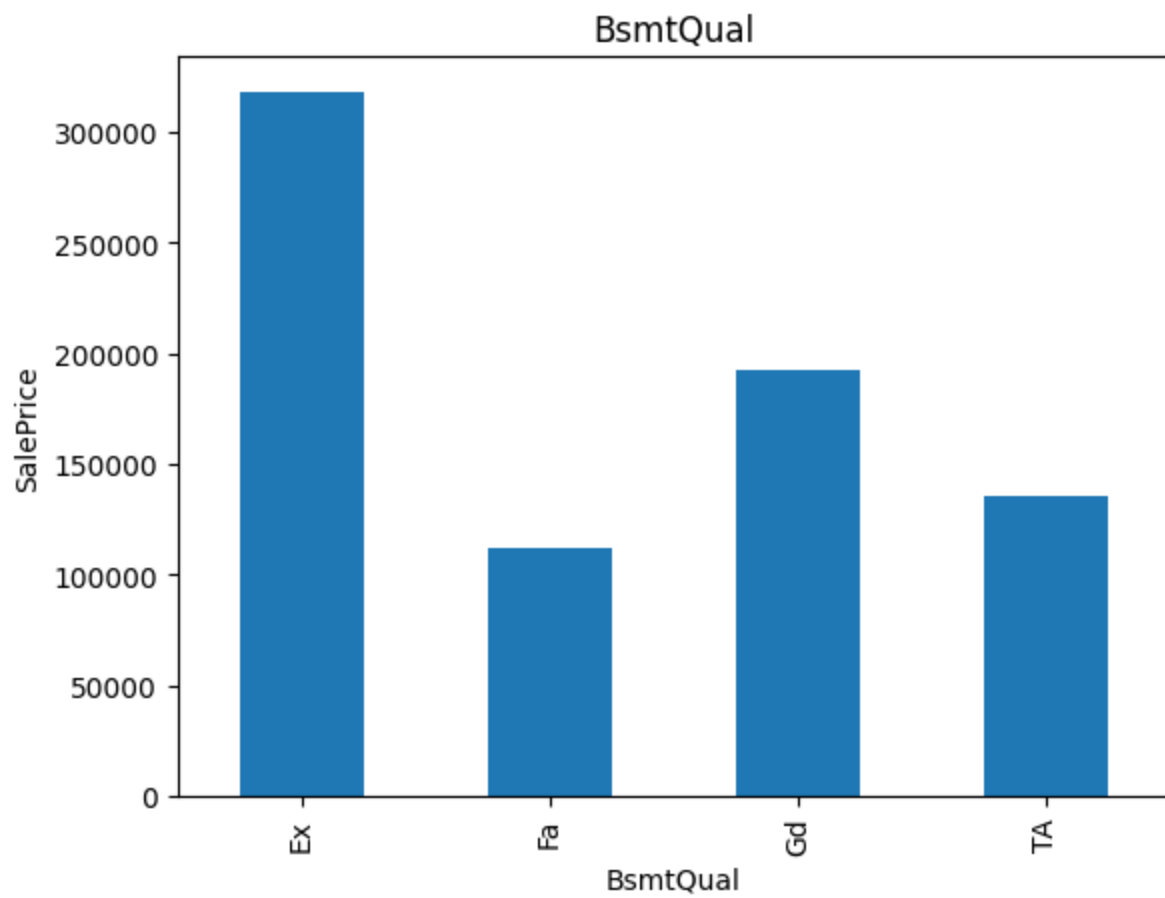


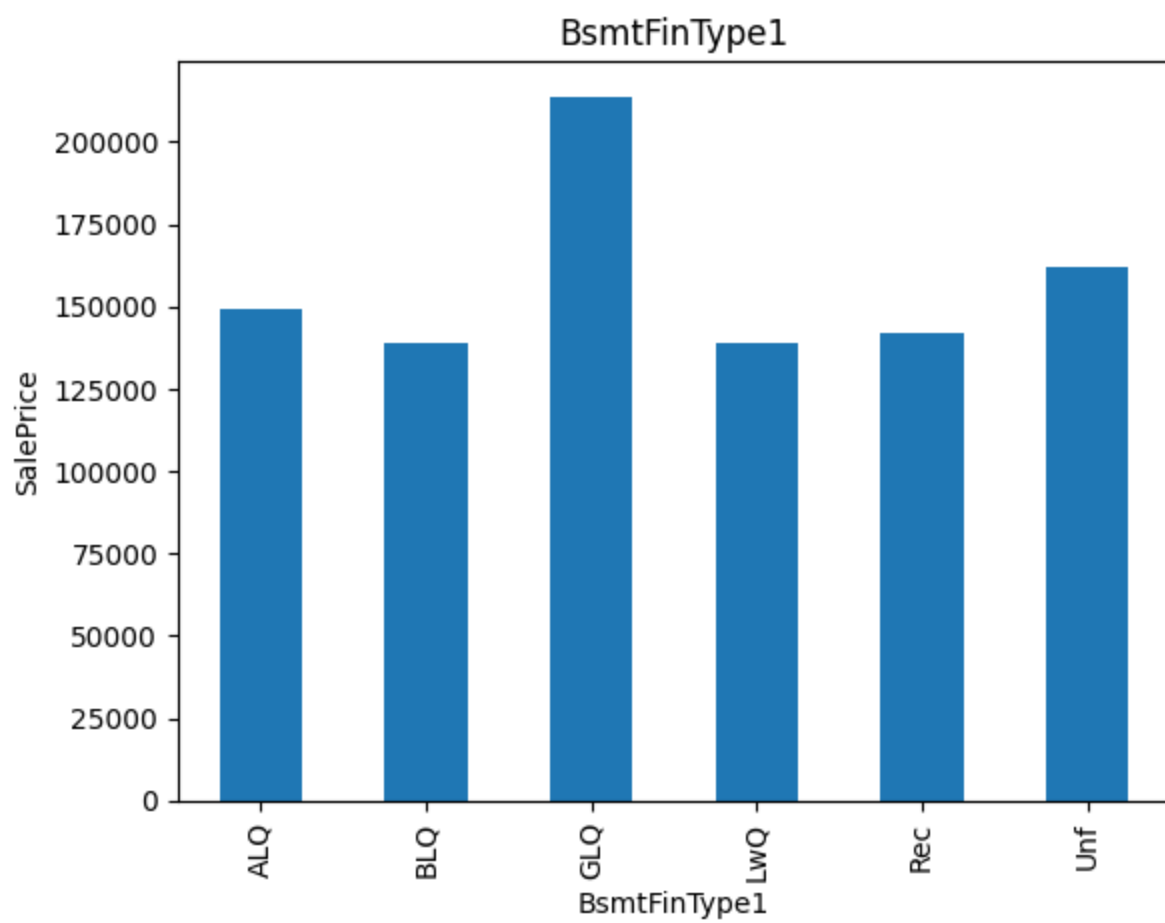
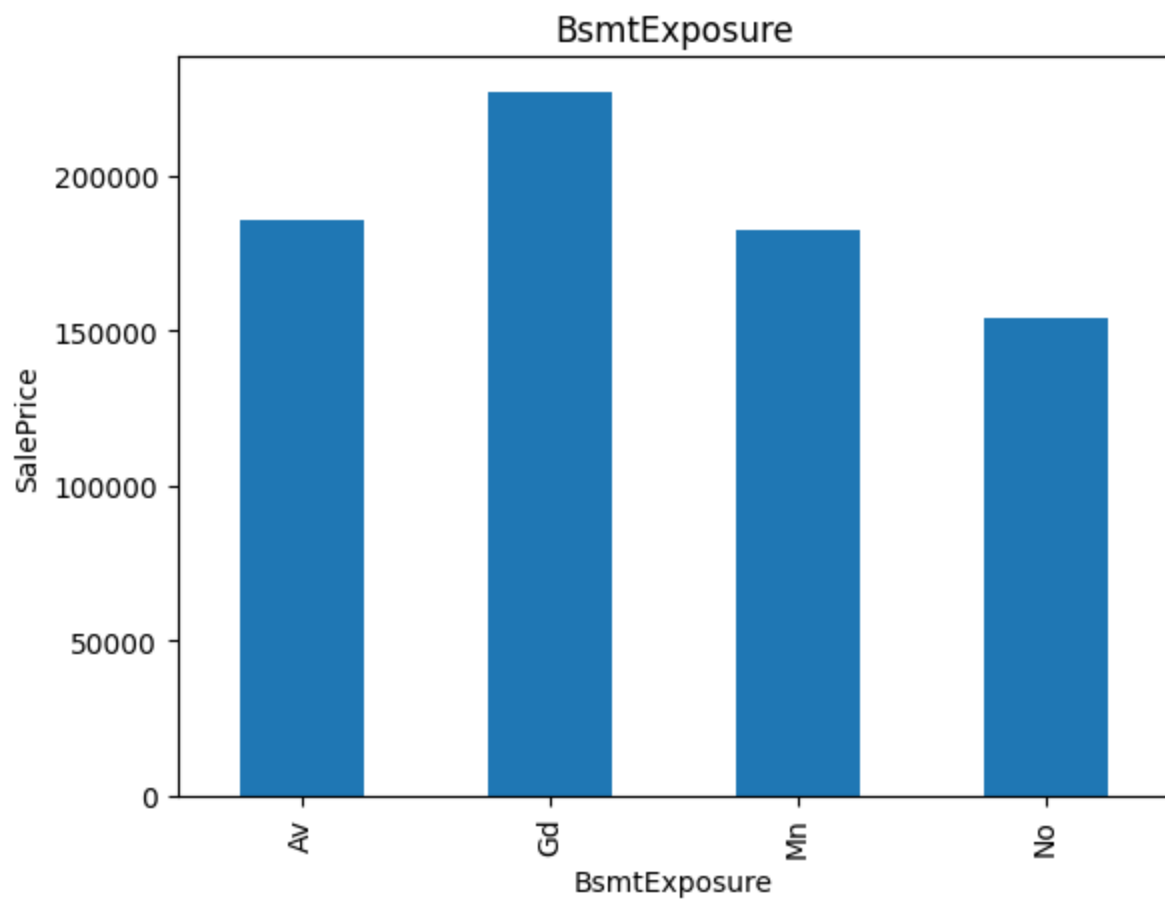


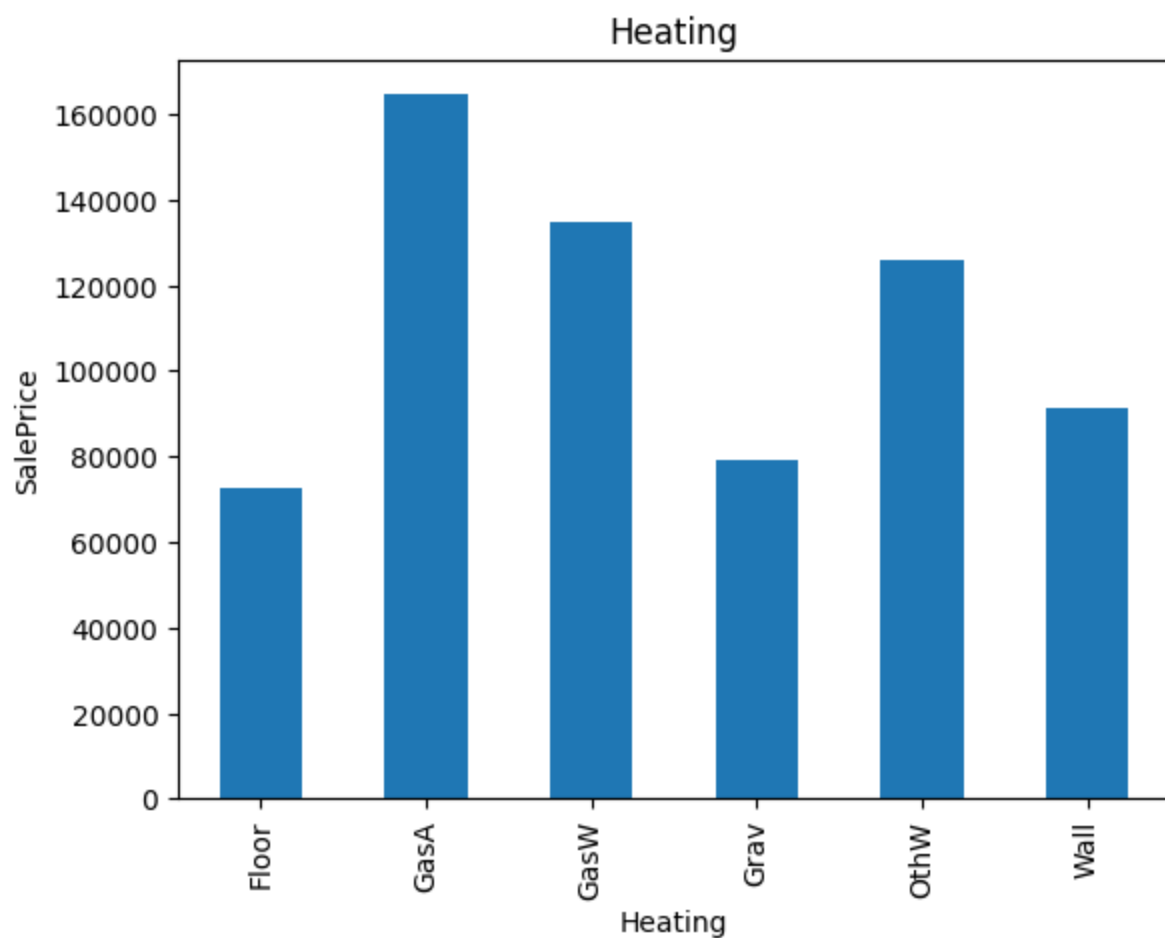
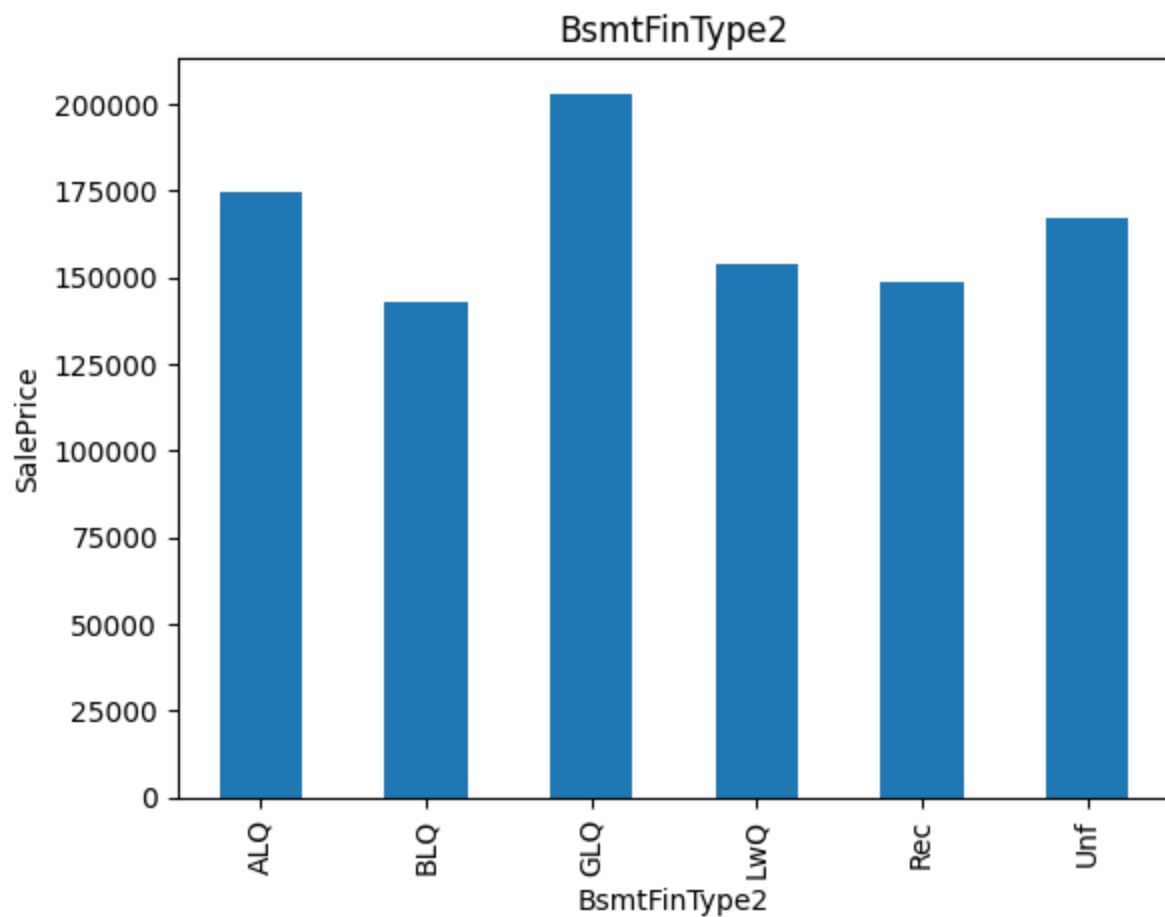


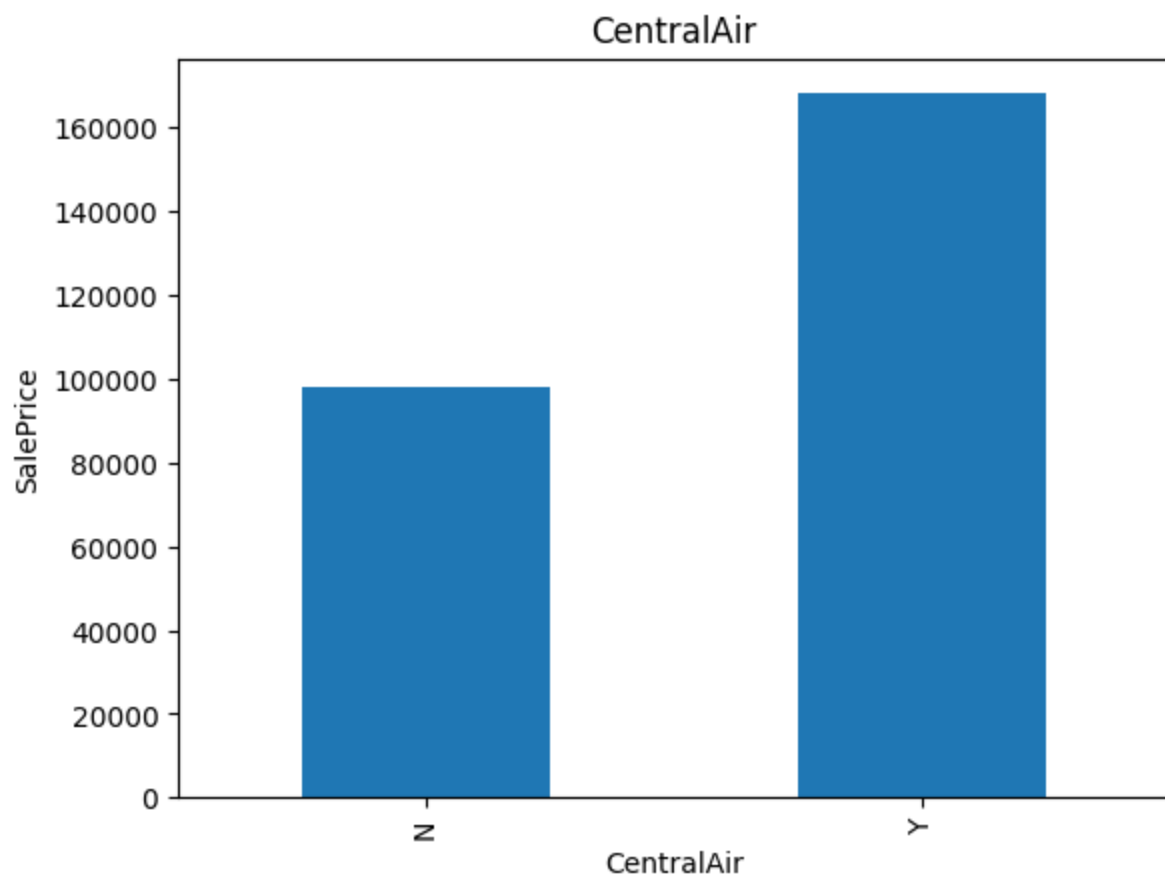
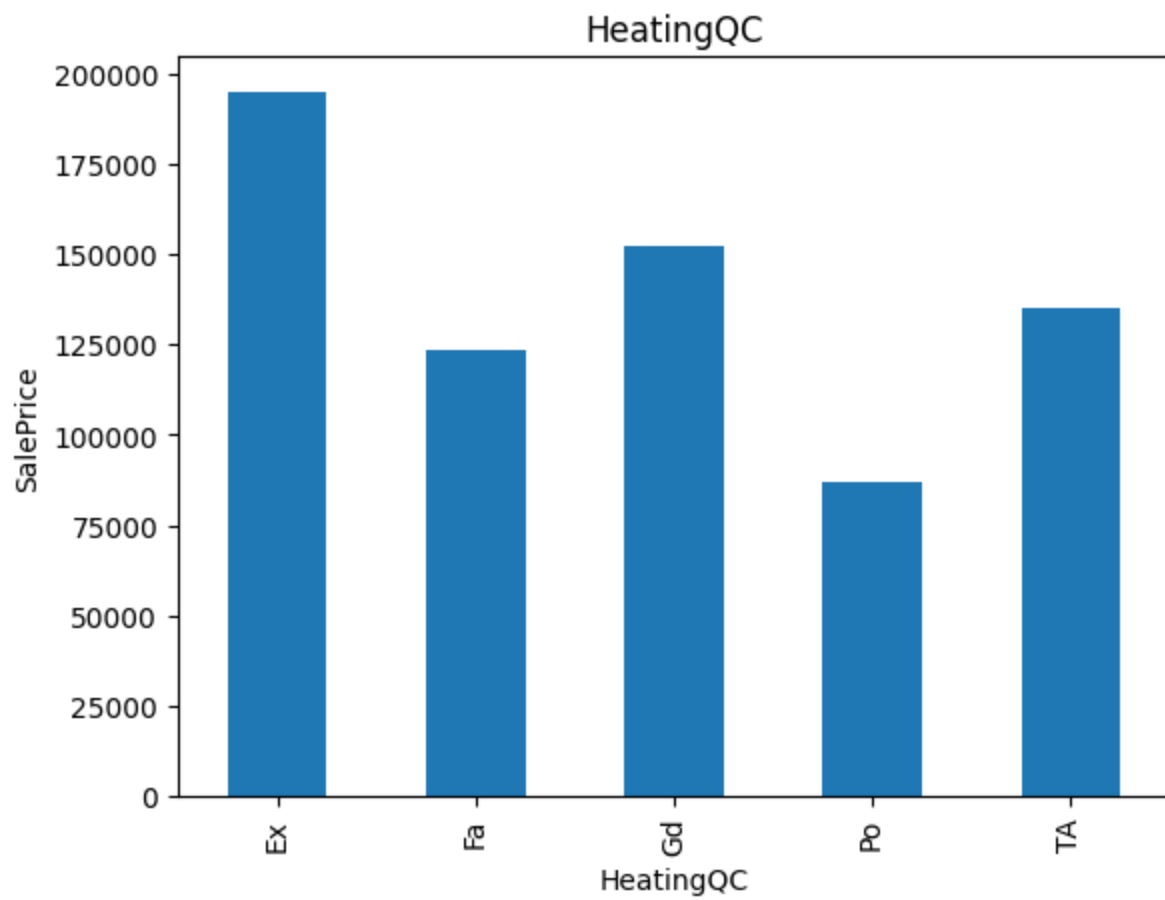


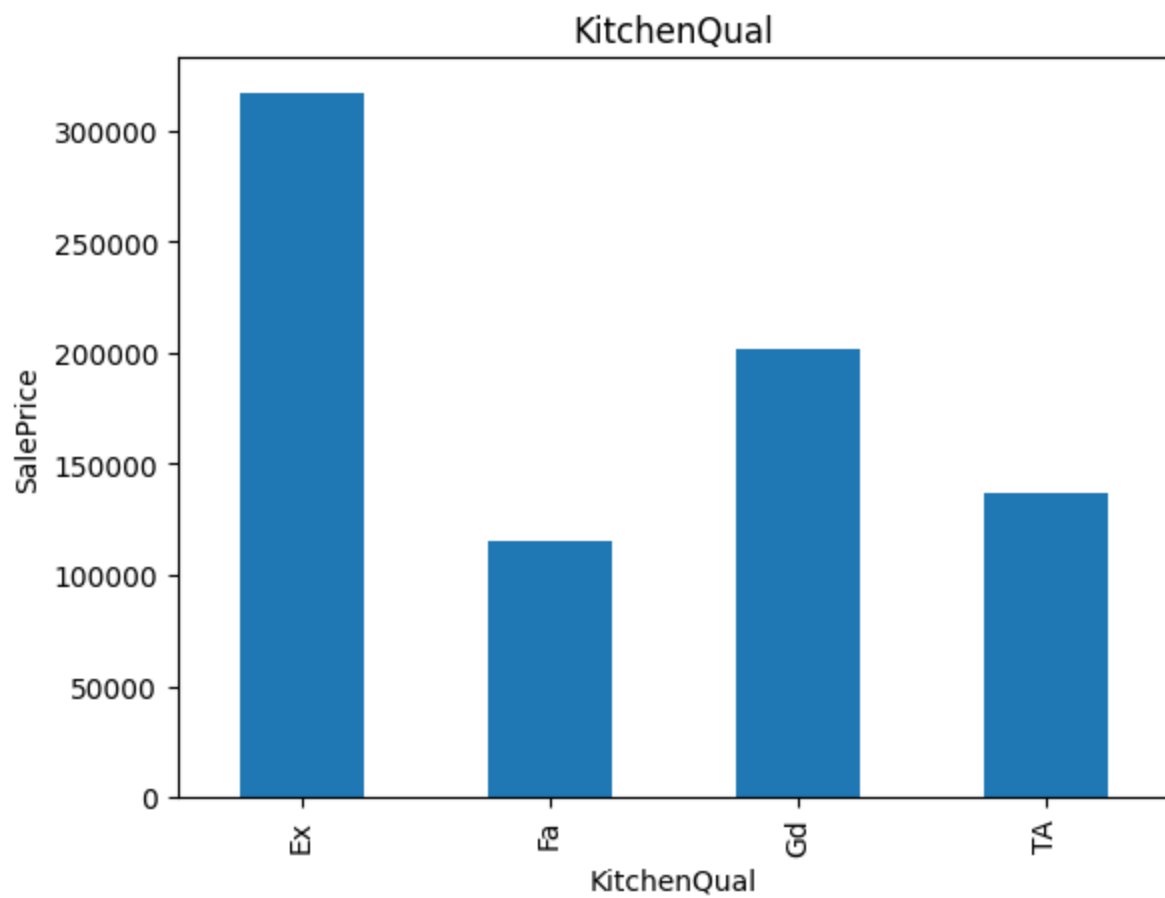
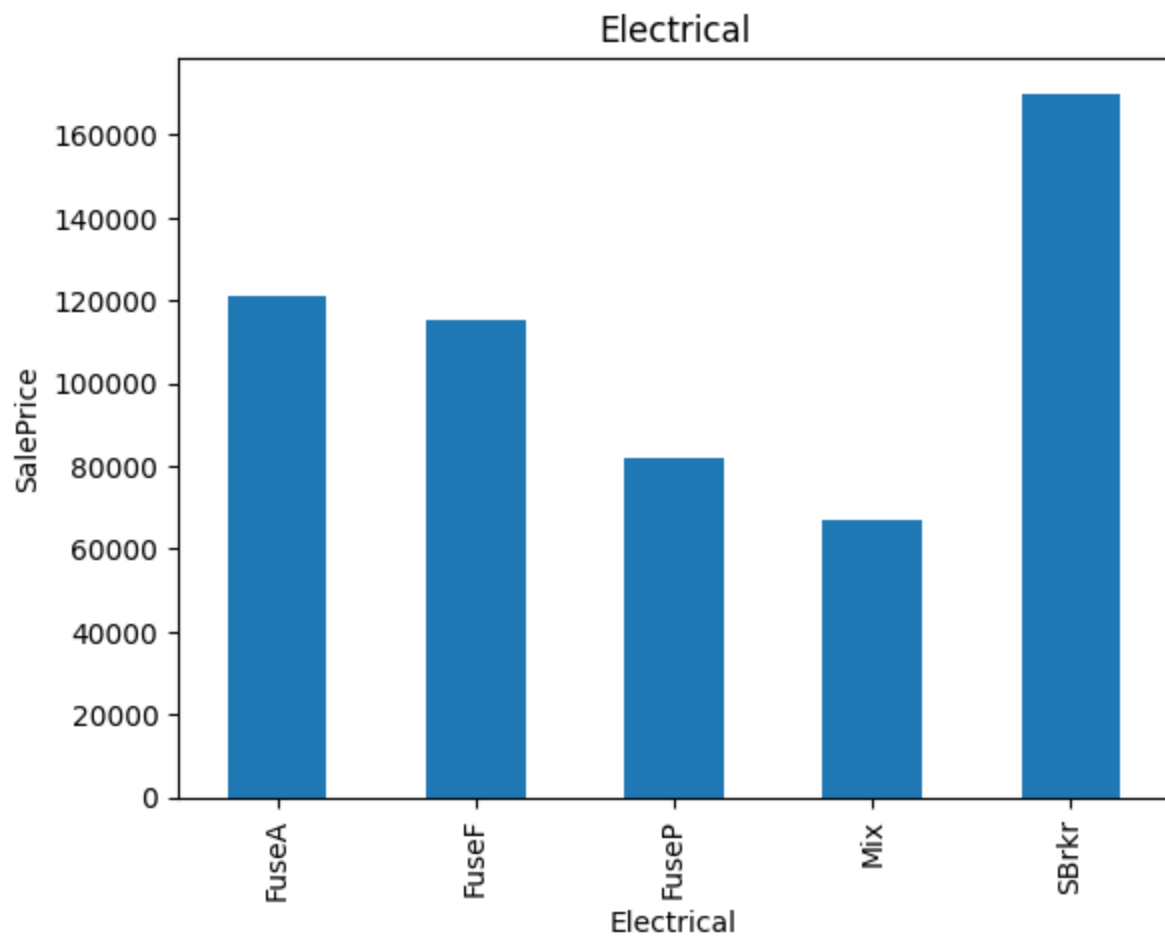


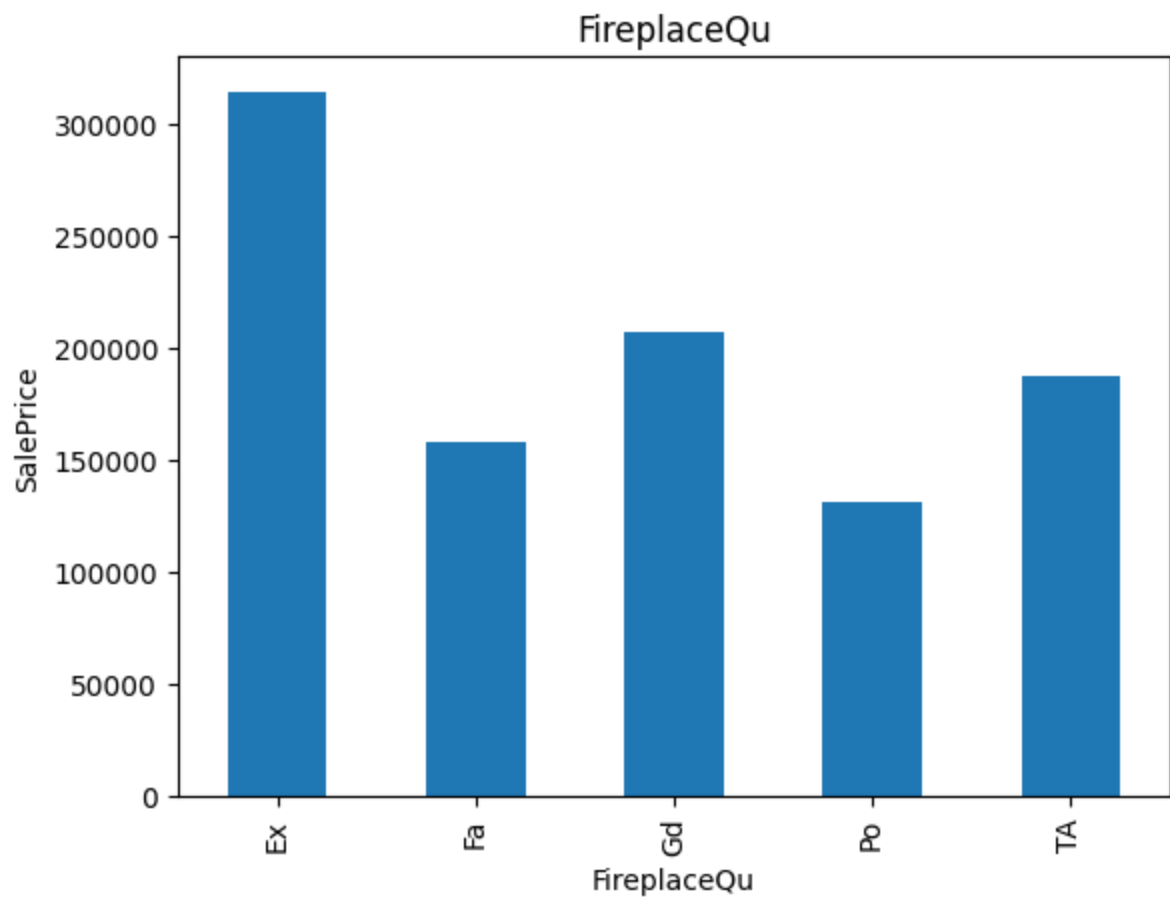
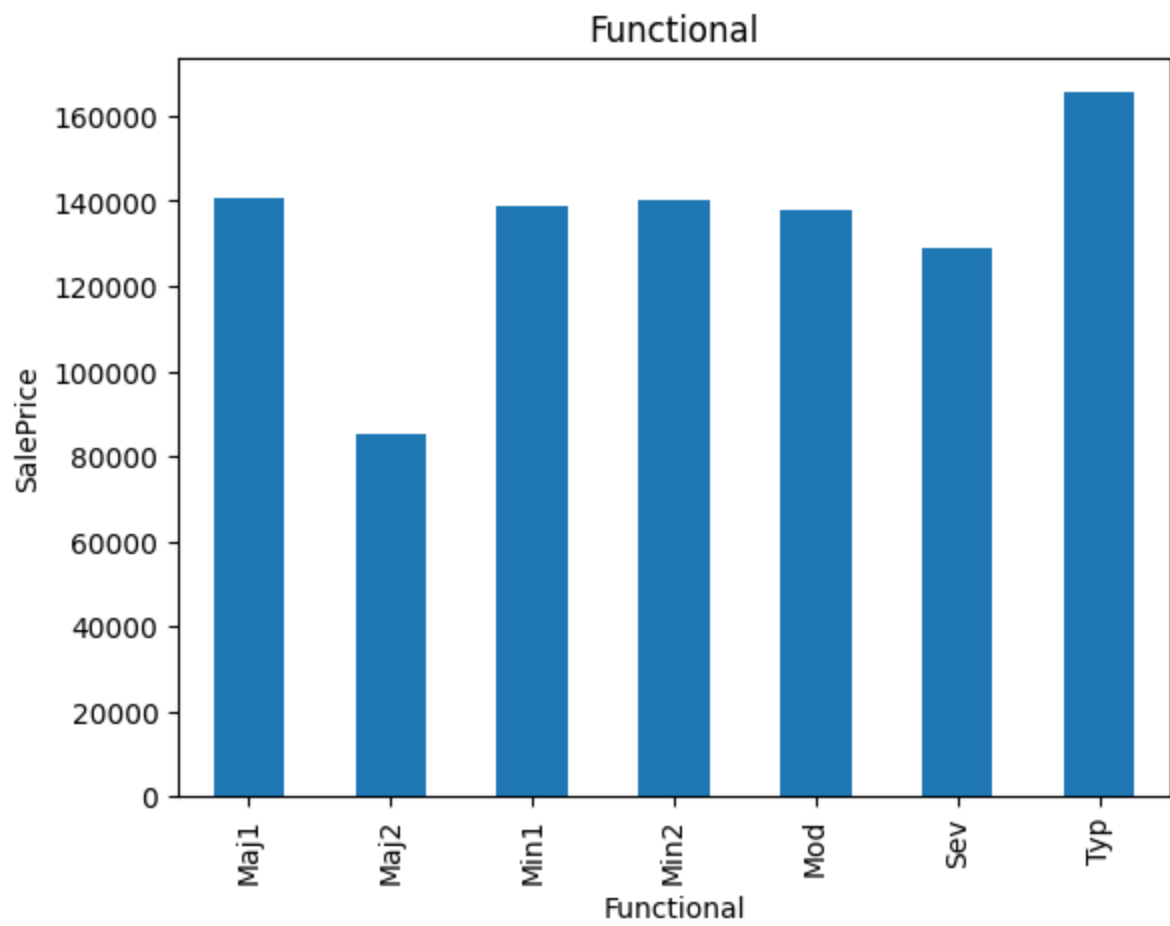


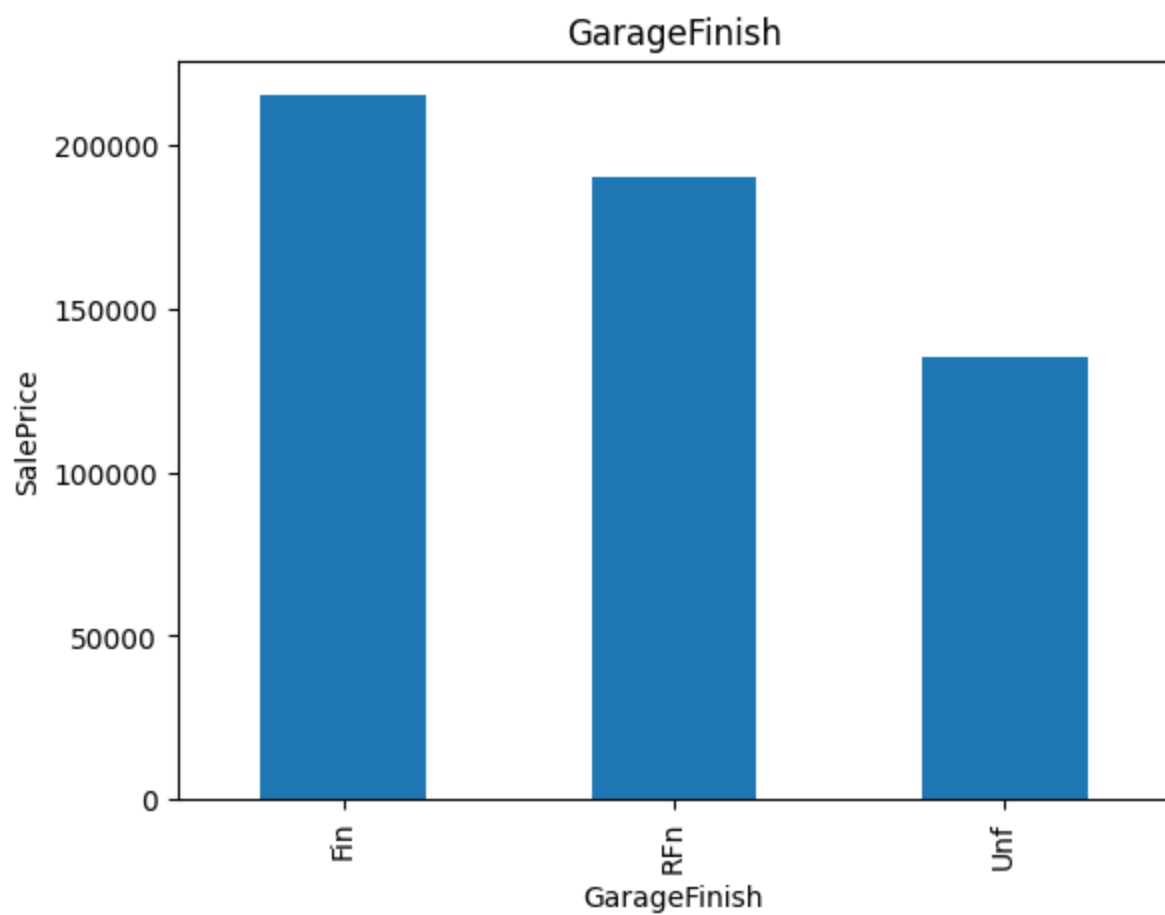
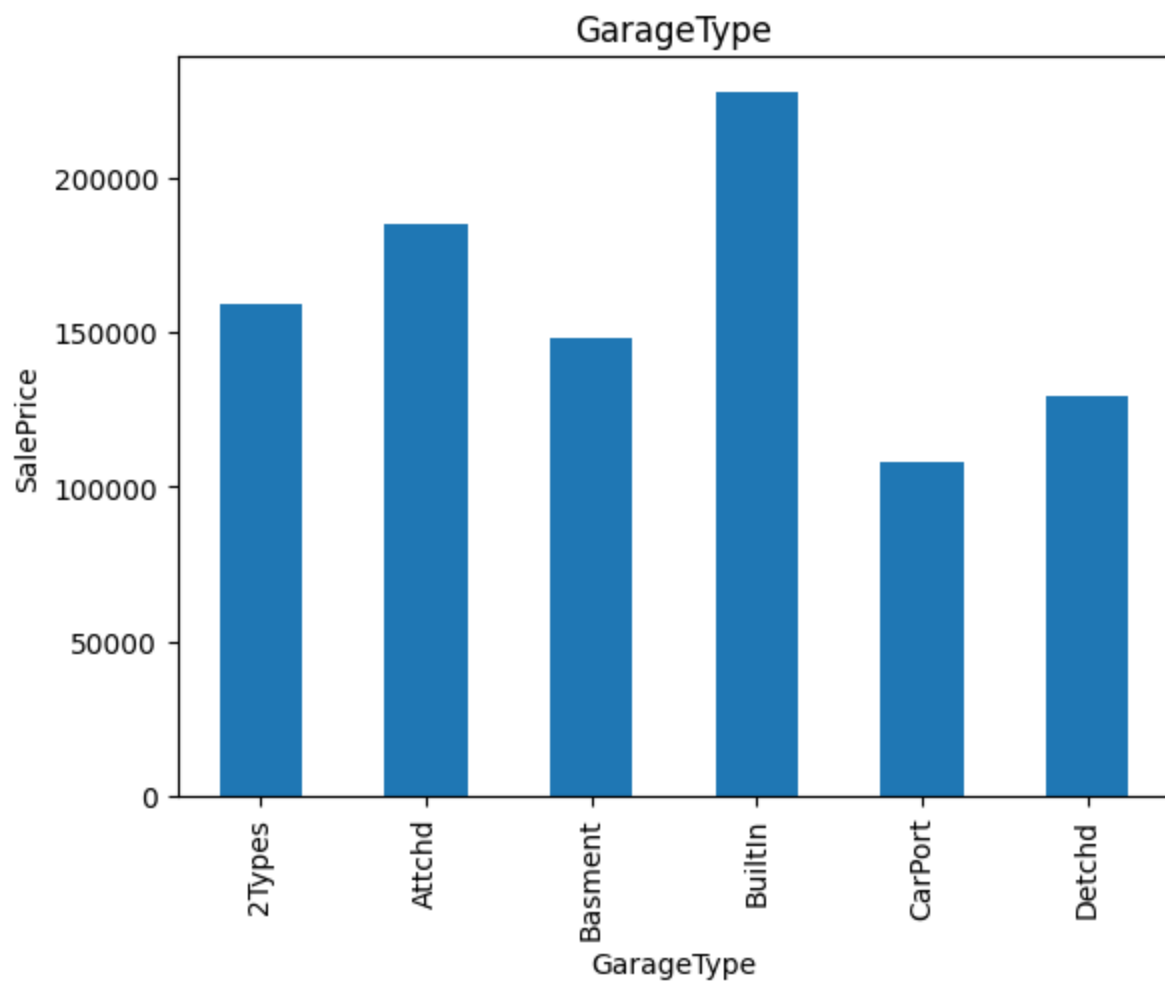


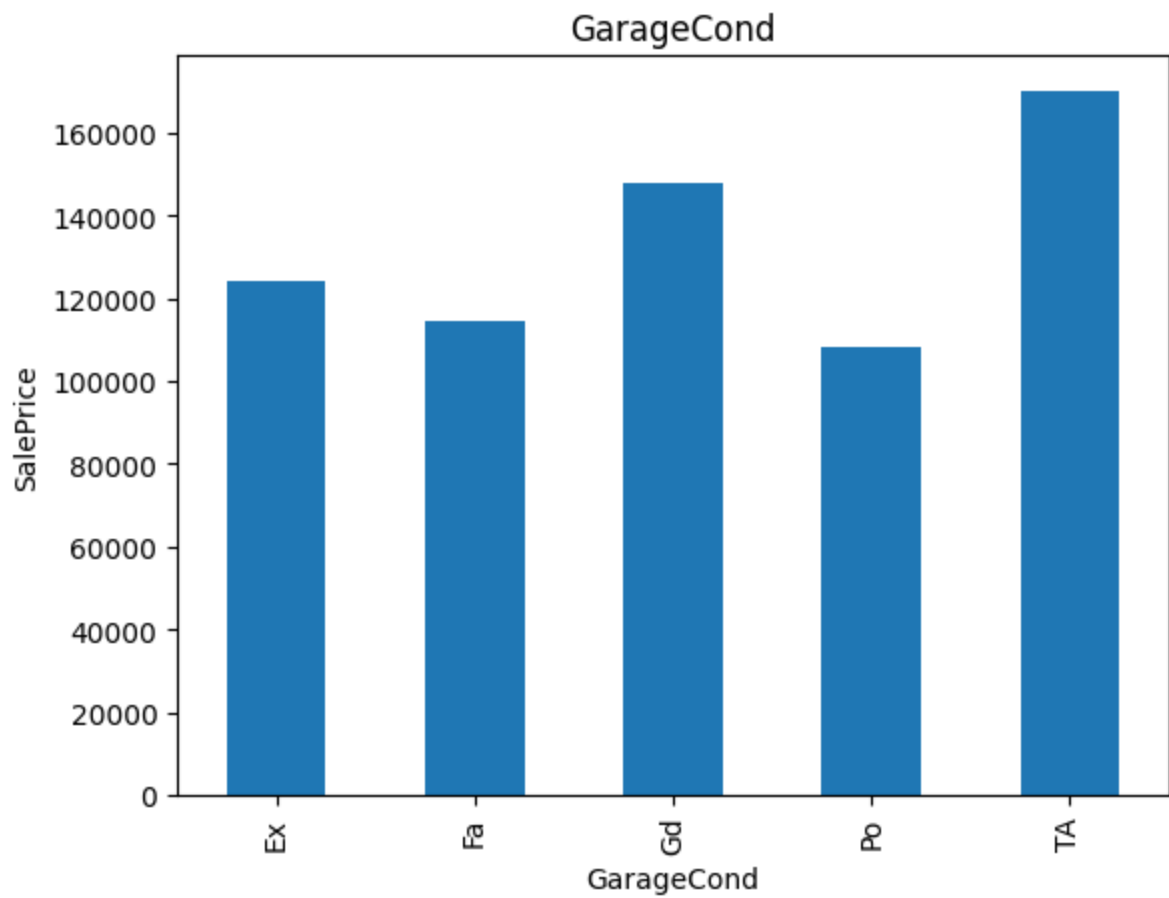
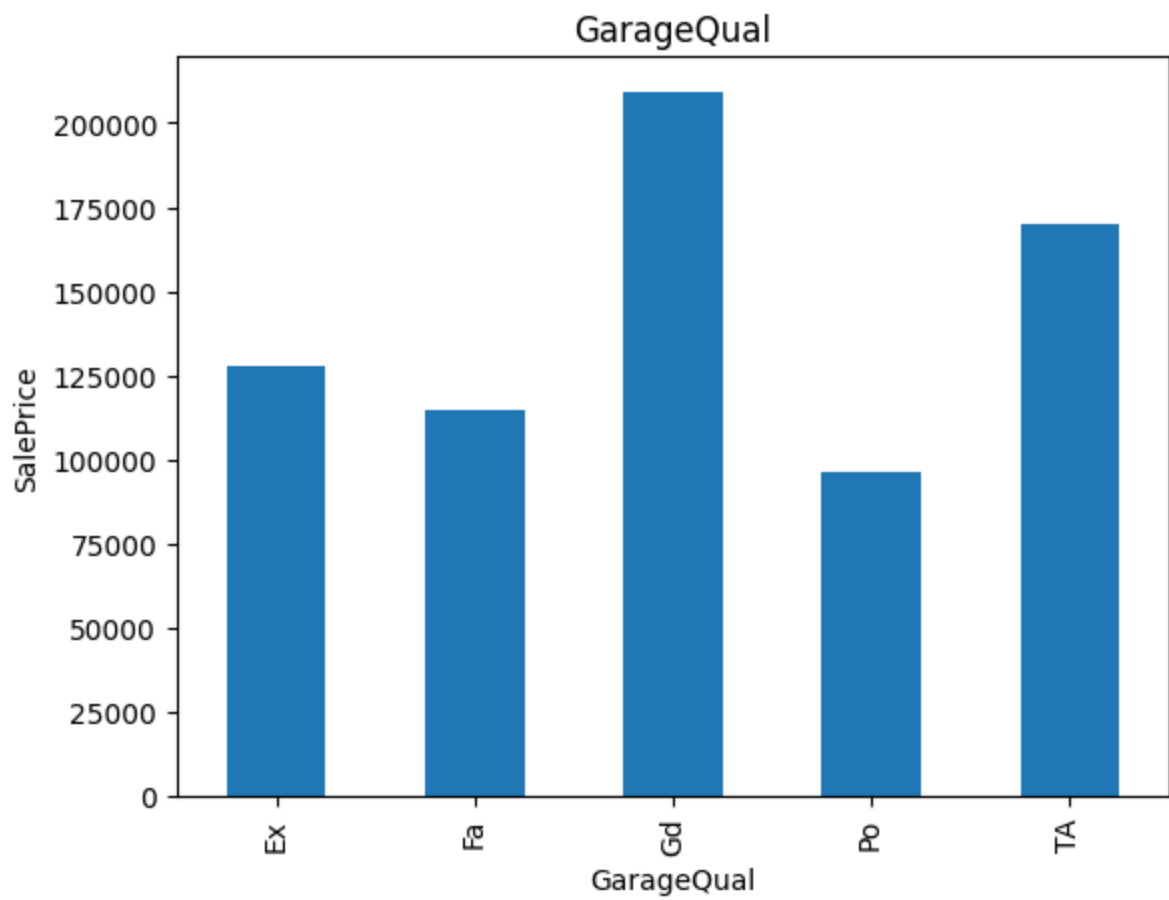


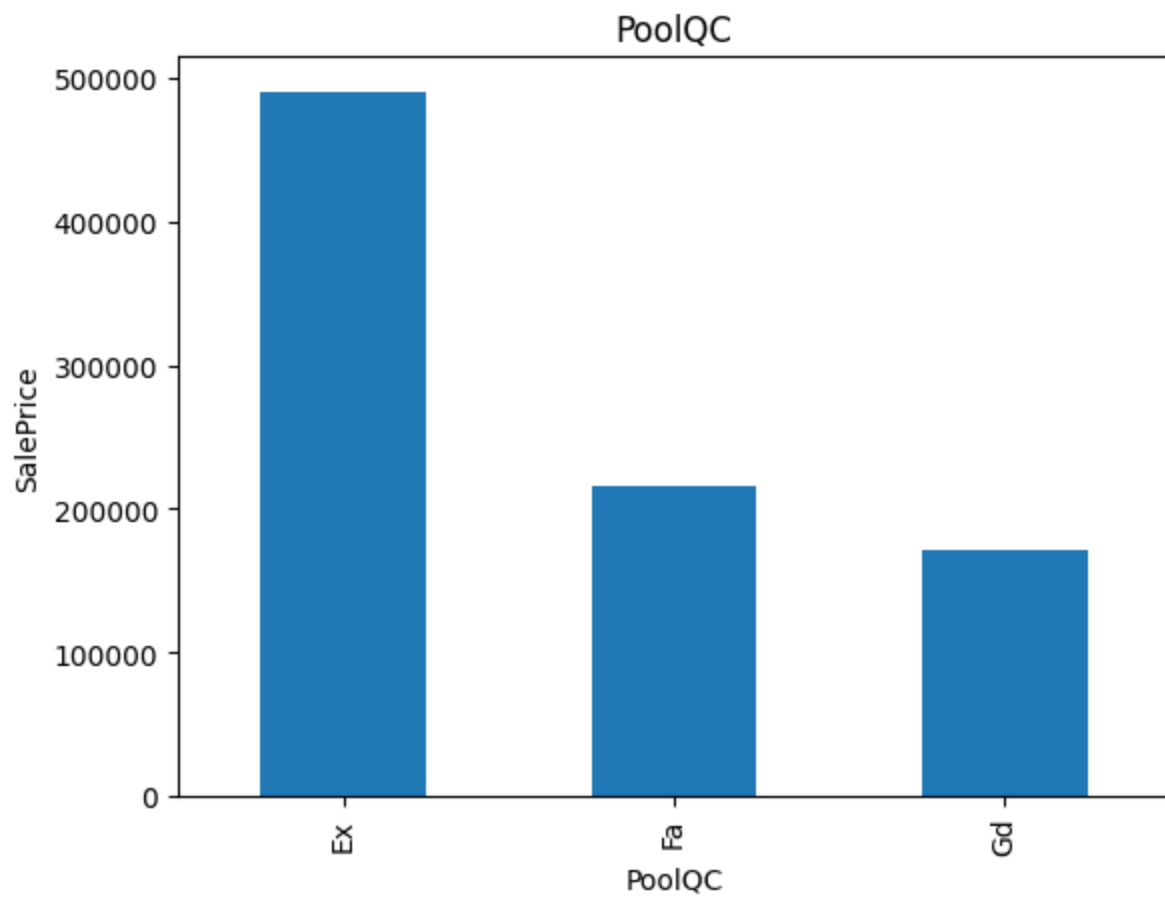
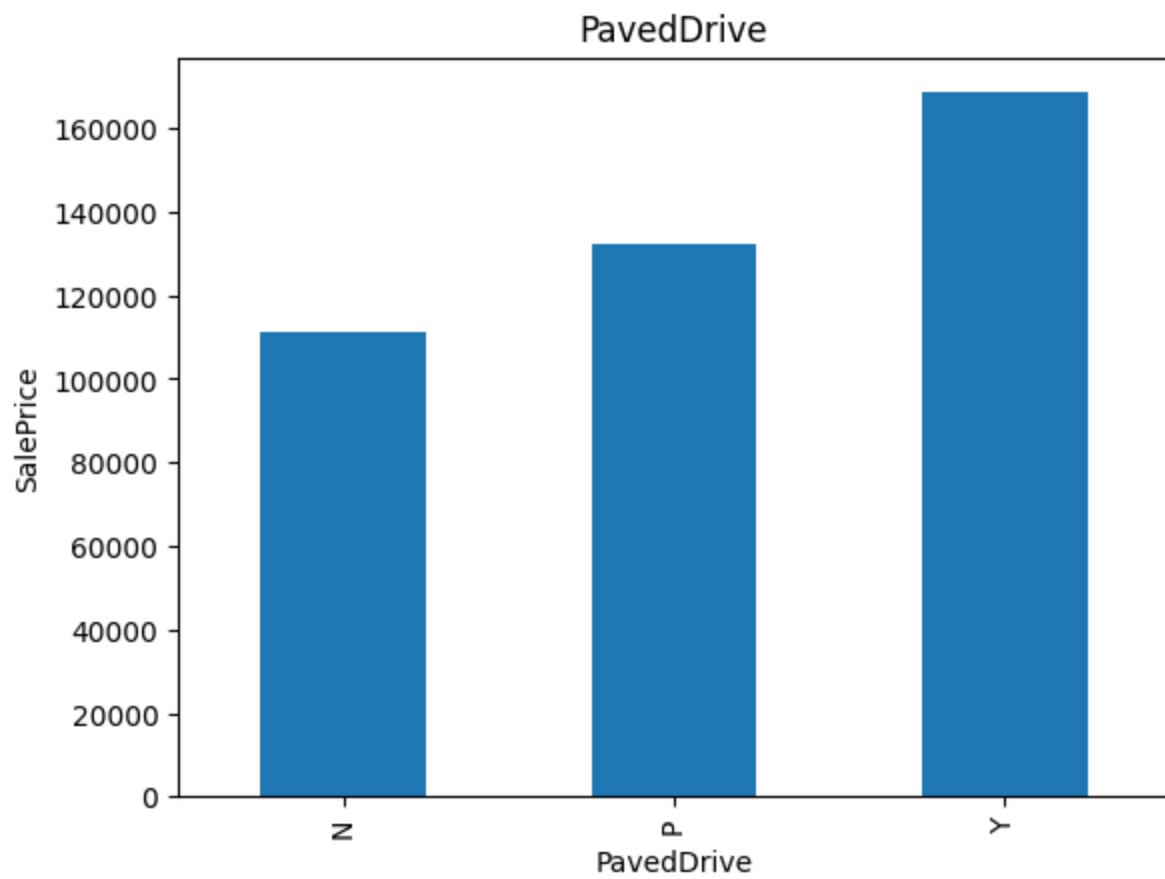


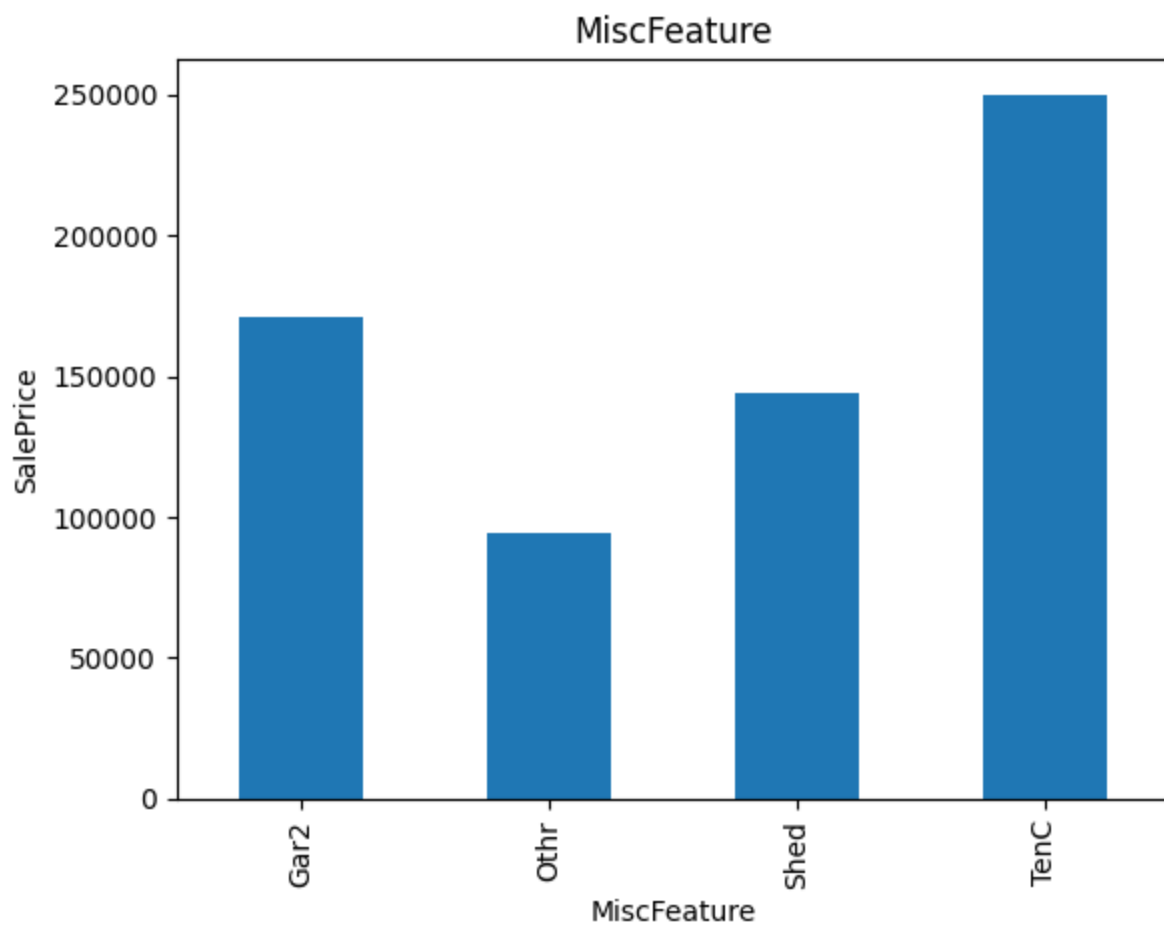
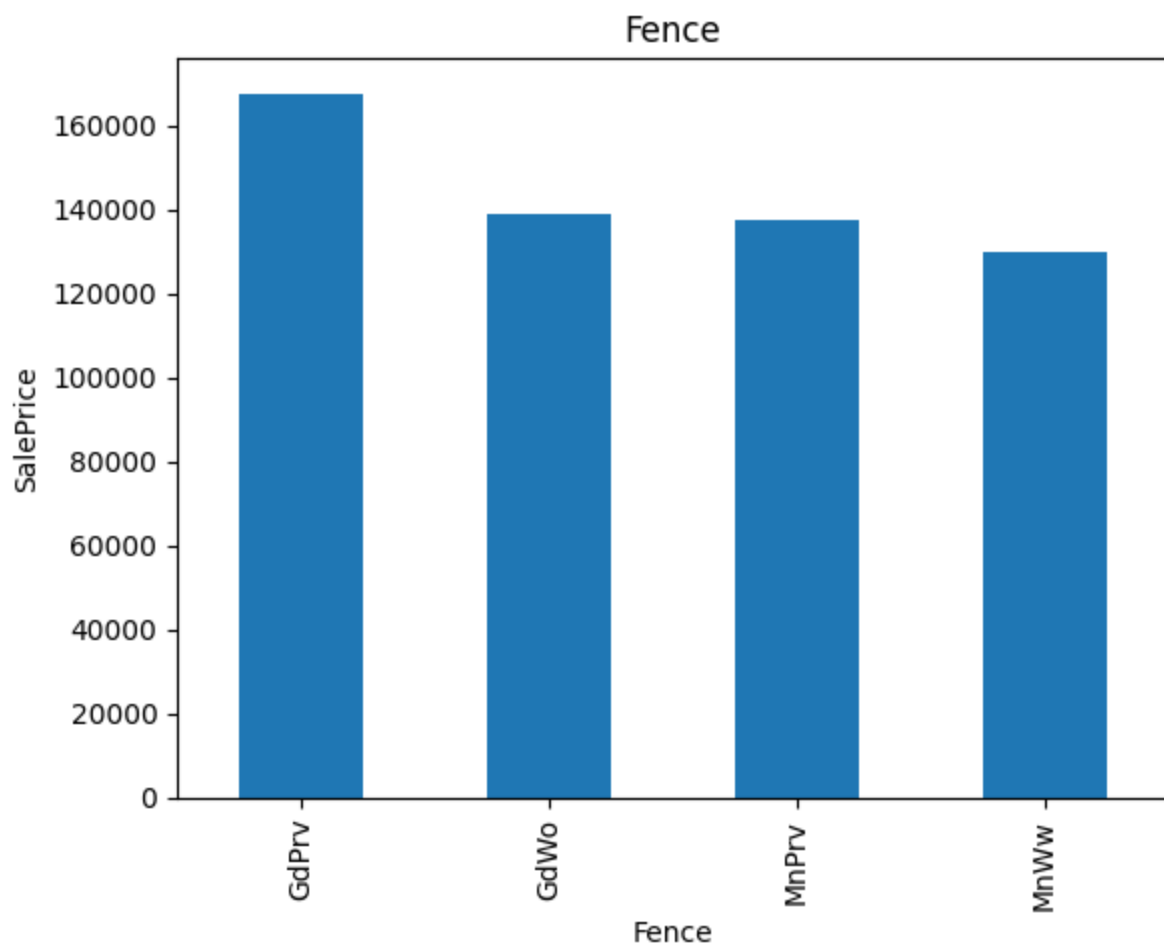


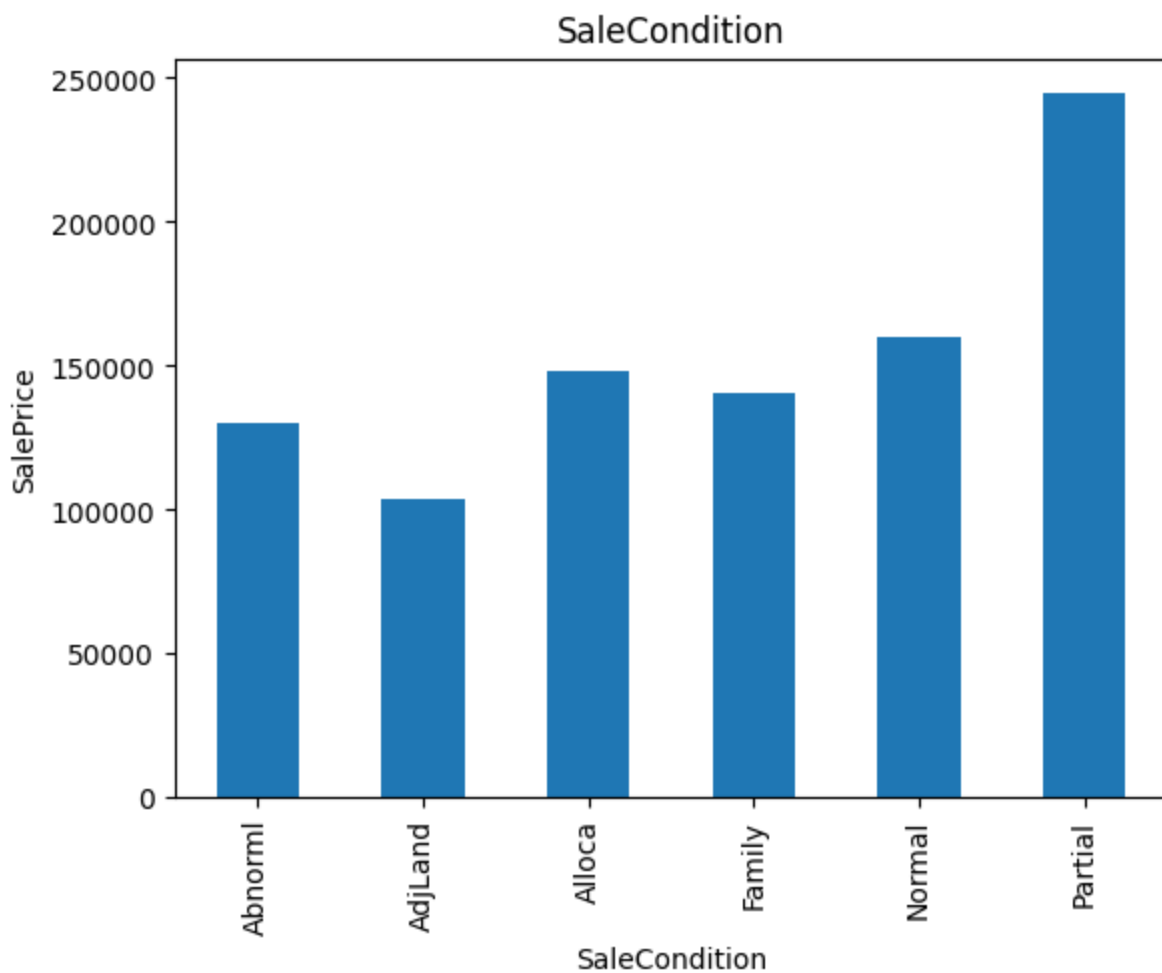
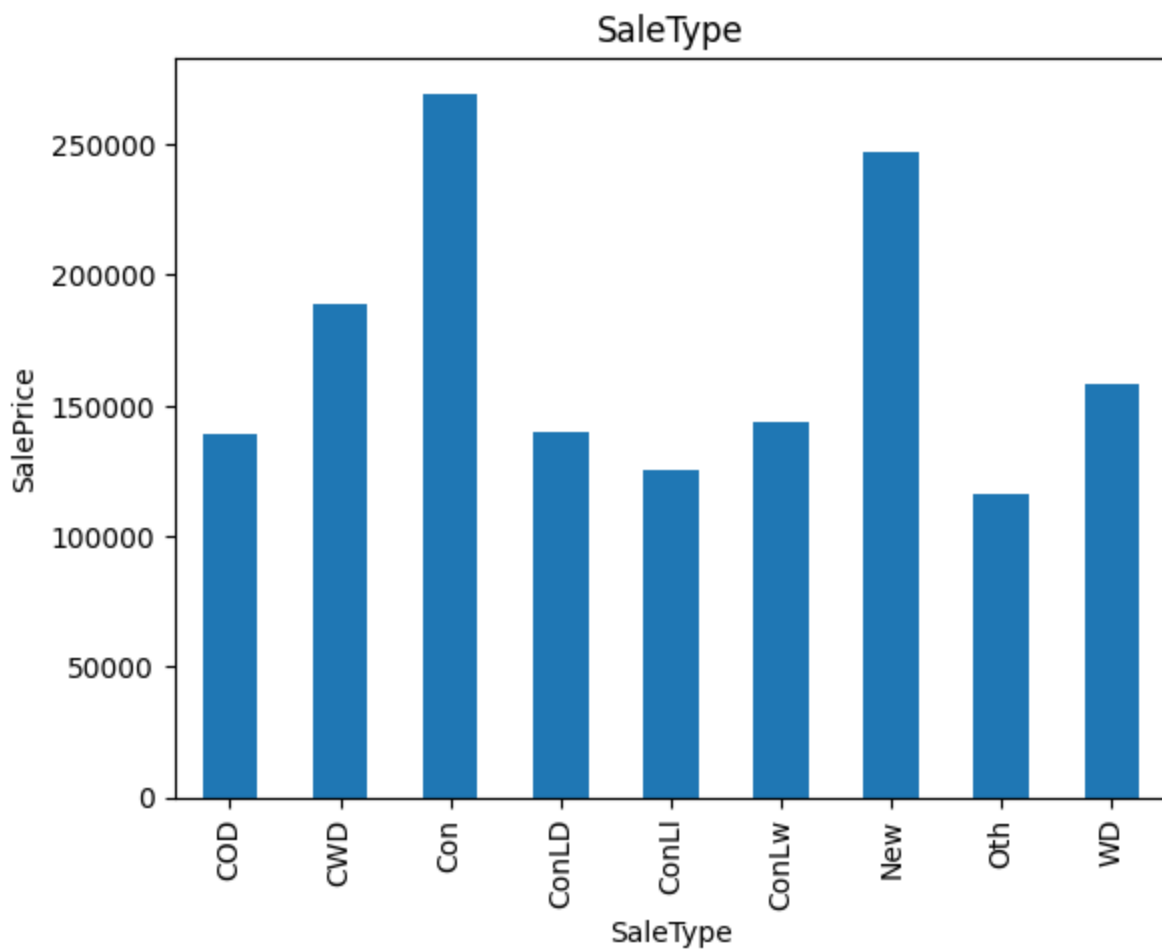












For feature engineering we will be follwing next notebook.

