

Exploratory Data Analysis on House Pricing - Kaggle Dataset

```
In [1]: 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 dataframes
pd.pandas.set_option('display.max_columns',None)
```

```
In [2]: # Reading Dataset
dataset = pd.read_csv('train.csv')
dataset.head()
```

```
Out[2]:
```

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

In Data Analysis We will Analyze To Find OUT the below things

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

7. Relationship between independent and dependent feature (sale price)

Missing Values

```
In [3]: # We are checking the percentage of nan values present in each feature
# 1.Create a list of features having missing values
features_with_na = [feature for feature in dataset.columns if dataset[feature].isnull().sum()>0]
# 2.Print the feature name and percentage of missing values
for feature in features_with_na:
    print("The feature is {} and missing values in {}% ".format(feature,np.round(dataset[feature].isnull().mean(),4)))
```

```
The feature is LotFrontage and missing values in 0.1774%
The feature is Alley and missing values in 0.9377%
The feature is MasVnrType and missing values in 0.0055%
The feature is MasVnrArea and missing values in 0.0055%
The feature is BsmtQual and missing values in 0.0253%
The feature is BsmtCond and missing values in 0.0253%
The feature is BsmtExposure and missing values in 0.026%
The feature is BsmtFinType1 and missing values in 0.0253%
The feature is BsmtFinType2 and missing values in 0.026%
The feature is Electrical and missing values in 0.0007%
The feature is FireplaceQu and missing values in 0.4726%
The feature is GarageType and missing values in 0.0555%
The feature is GarageYrBlt and missing values in 0.0555%
The feature is GarageFinish and missing values in 0.0555%
The feature is GarageQual and missing values in 0.0555%
The feature is GarageCond and missing values in 0.0555%
The feature is PoolQC and missing values in 0.9952%
The feature is Fence and missing values in 0.8075%
The feature is MiscFeature and missing values in 0.963%
```

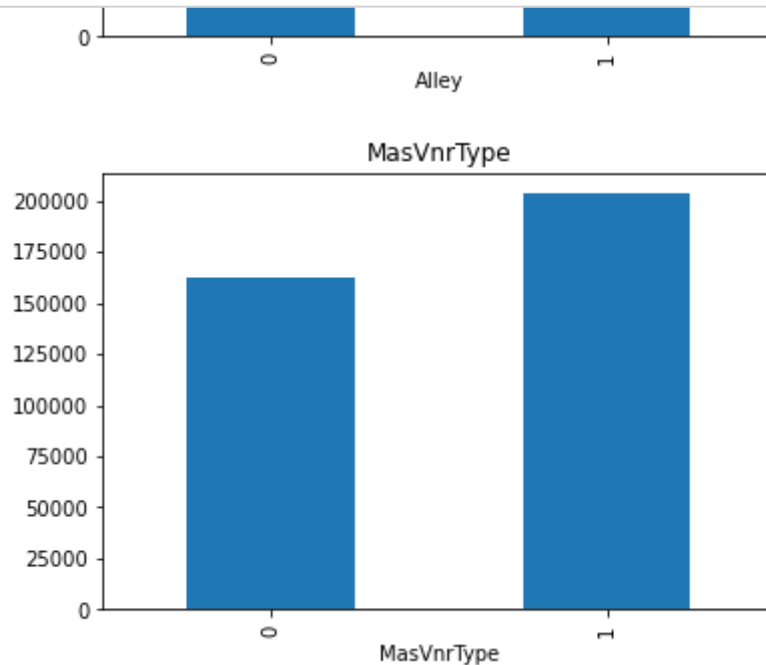
Since there are many missing values, We need to find the relationship between Missing Values and Sales Price

We can plot some diagram for visualize the Relationship

```
In [4]: for feature in features_with_na:
        data = dataset.copy()

        # We can make a variable that indicates 1 if the observation was missing else 0
        data[feature] = np.where(data[feature].isnull(),1,0)

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



Here with the relation between the missing values and the dependent variable is clearly visible. So we need to replace these nan values with something meaningful which we will do in the Feature Engineering Section

No need to consider the features like 'Id of Houses'

Numerical Variables

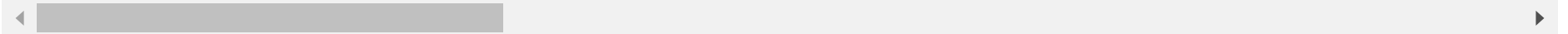
```
In [5]: # List of numerical variables
numerical_feature = [feature for feature in dataset.columns if dataset[feature].dtypes != 'O']
print('Number of numerical variables : ', len(numerical_feature))
```

Number of numerical variables : 38

```
In [6]: # visualize the numerical variables
dataset[numerical_feature].head(2)
```

```
Out[6]:
```

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF
0	1	60	65.0	8450	7	5	2003	2003	196.0	706	0	150
1	2	20	80.0	9600	6	8	1976	1976	0.0	978	0	284



Temporal Variable (Date , Time)

```
In [7]: # List of variables that contain year information
year_feature = [feature for feature in numerical_feature if 'Yr' in feature or 'Year' in feature]
year_feature
```

```
Out[7]: ['YearBuilt', 'YearRemodAdd', 'GarageYrBlt', 'YrSold']
```

From the Dataset we understood we have 4 year variables. We have extract information from the datetime variables like number of years or days. One example from the current dataset is years between the year which house buit and year which house was sold. We will consider this analysis in the Feature Engineering.

```
In [8]: # We can analyze the details of these year 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]
```

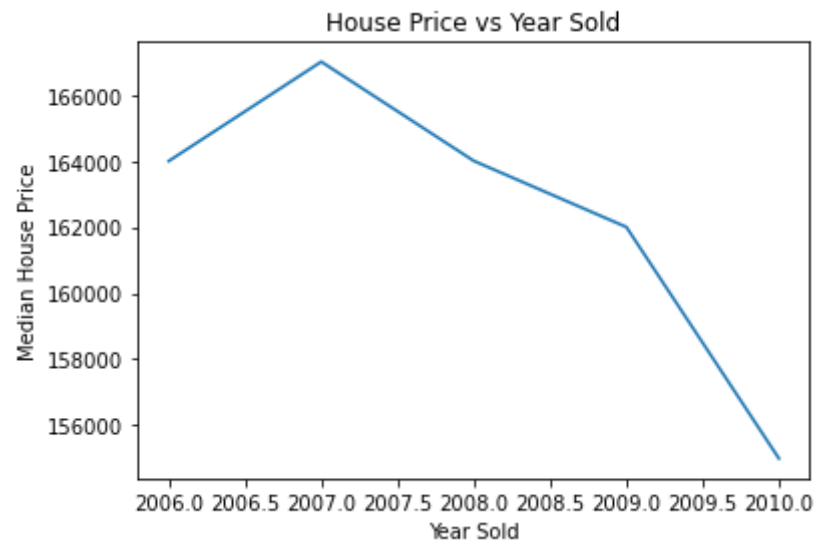
```
GarageYrBlt [2003. 1976. 2001. 1998. 2000. 1993. 2004. 1973. 1931. 1939. 1965. 2005.  
1962. 2006. 1960. 1991. 1970. 1967. 1958. 1930. 2002. 1968. 2007. 2008.  
1957. 1920. 1966. 1959. 1995. 1954. 1953. nan 1983. 1977. 1997. 1985.  
1963. 1981. 1964. 1999. 1935. 1990. 1945. 1987. 1989. 1915. 1956. 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 [9]: *# We can analyze the Temporal Date-Time Variables*
We can analyze whether there is a relation between year of house which sold and Sales price

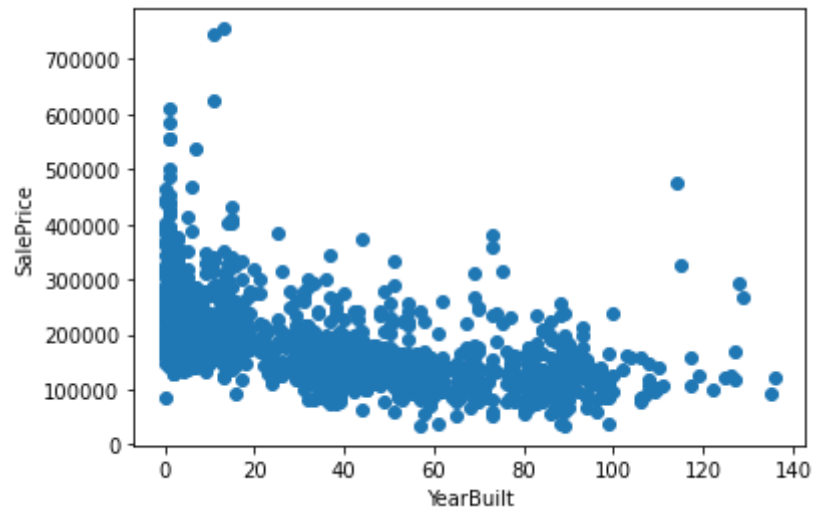
```
dataset.groupby('YrSold')['SalePrice'].median().plot()  
plt.xlabel('Year Sold')  
plt.ylabel('Median House Price')  
plt.title('House Price vs Year Sold')
```

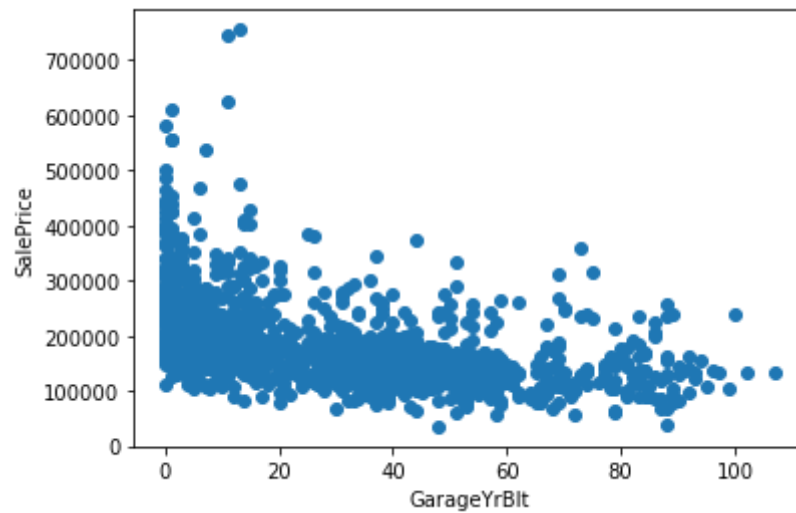
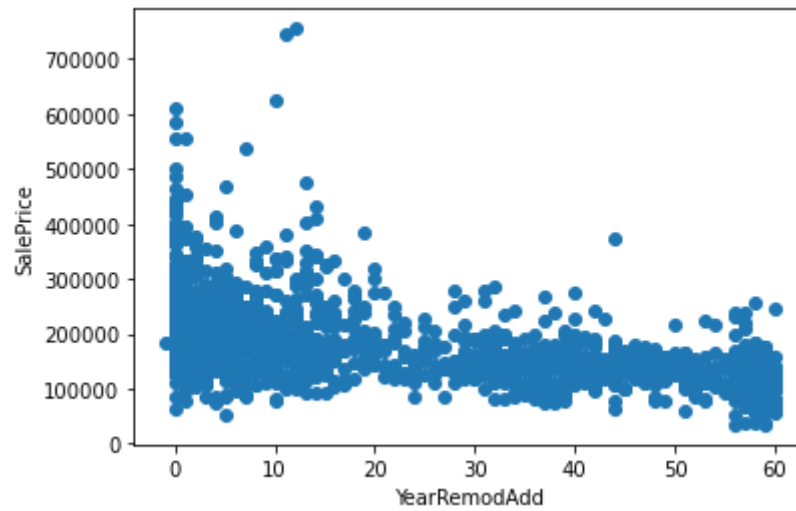
Out[9]: Text(0.5, 1.0, 'House Price vs Year Sold')



```
In [10]: ### Here we will compare the difference between all years features with salesprice
data = dataset.copy()
for feature in year_feature:
    if feature != 'YrSold':
        # We get the difference between year variable and year which house sold
        data[feature] = data['YrSold'] - data[feature]

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





we know numerical features are of 2 types

1. Continuous Variable
2. Discrete Variable

Discrete Variable


```
In [11]: discrete_feature = [feature for feature in numerical_feature if len(dataset[feature].unique())<=25 and feature not in y]
print("Discrete Variables Count: {}".format(len(discrete_feature)))
```

Discrete Variables Count: 17

```
In [12]: discrete_feature
```

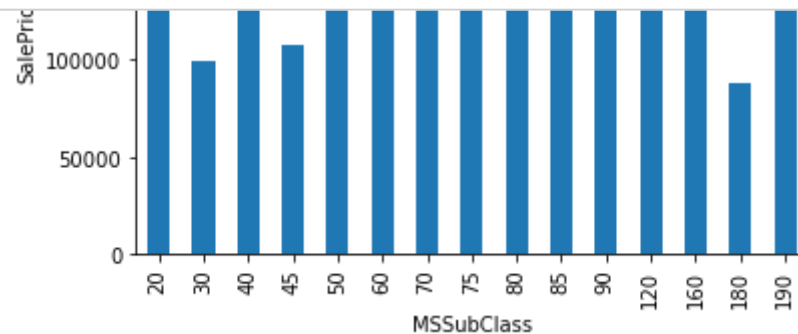
```
Out[12]: ['MSSubClass',
'OverallQual',
'OverallCond',
'LowQualFinSF',
'BsmtFullBath',
'BsmtHalfBath',
'FullBath',
'HalfBath',
'BedroomAbvGr',
'KitchenAbvGr',
'TotRmsAbvGrd',
'Fireplaces',
'GarageCars',
'3SsnPorch',
'PoolArea',
'MiscVal',
'MoSold']
```

```
In [13]: dataset[discrete_feature].head(2)
```

```
Out[13]:
```

	MSSubClass	OverallQual	OverallCond	LowQualFinSF	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	BedroomAbvGr	KitchenAbvGr	TotRmsAbvGrd
0	60	7	5	0	1	0	2	1	3	1	
1	20	6	8	0	0	1	2	0	3	1	

```
In [14]: # Lets Find the relationship between discrete_feature and SalePrice
# In EDA we have always compare with dependent feature for getting some analysis
data = dataset.copy()
for feature in discrete_feature:
    data.groupby(feature)['SalePrice'].median().plot.bar()
    plt.xlabel(feature)
    plt.ylabel('SalePrice')
    plt.title(feature)
    plt.show()
```



Continuous Variable

```
In [15]: continuous_feature = [feature for feature in numerical_feature if feature not in discrete_feature + year_feature + ['Id']]
print("Continuous feature Count: {}".format(len(continuous_feature)))
```

Continuous feature Count: 16

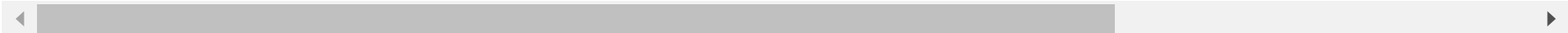
```
In [16]: continuous_feature
```

```
Out[16]: ['LotFrontage',  
          'LotArea',  
          'MasVnrArea',  
          'BsmtFinSF1',  
          'BsmtFinSF2',  
          'BsmtUnfSF',  
          'TotalBsmtSF',  
          '1stFlrSF',  
          '2ndFlrSF',  
          'GrLivArea',  
          'GarageArea',  
          'WoodDeckSF',  
          'OpenPorchSF',  
          'EnclosedPorch',  
          'ScreenPorch',  
          'SalePrice']
```

```
In [17]: dataset[continuous_feature].head(2)
```

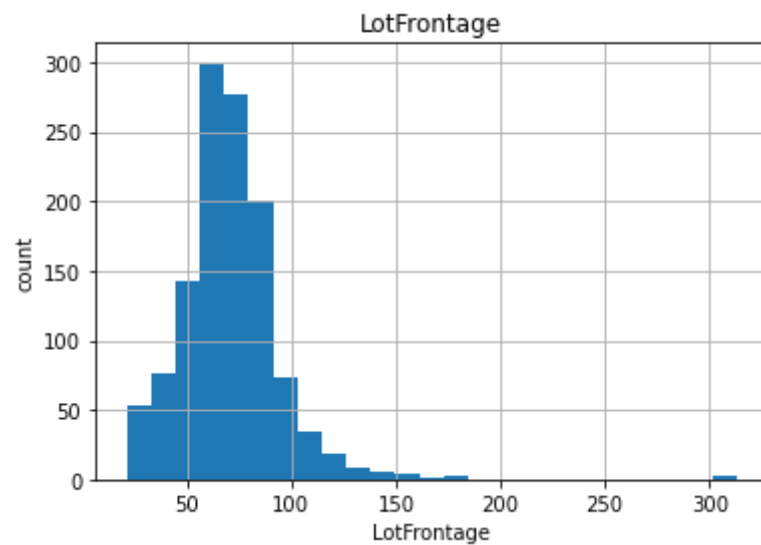
```
Out[17]:
```

	LotFrontage	LotArea	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	1stFlrSF	2ndFlrSF	GrLivArea	GarageArea	WoodDeckS
0	65.0	8450	196.0	706	0	150	856	856	854	1710	548	
1	80.0	9600	0.0	978	0	284	1262	1262	0	1262	460	29

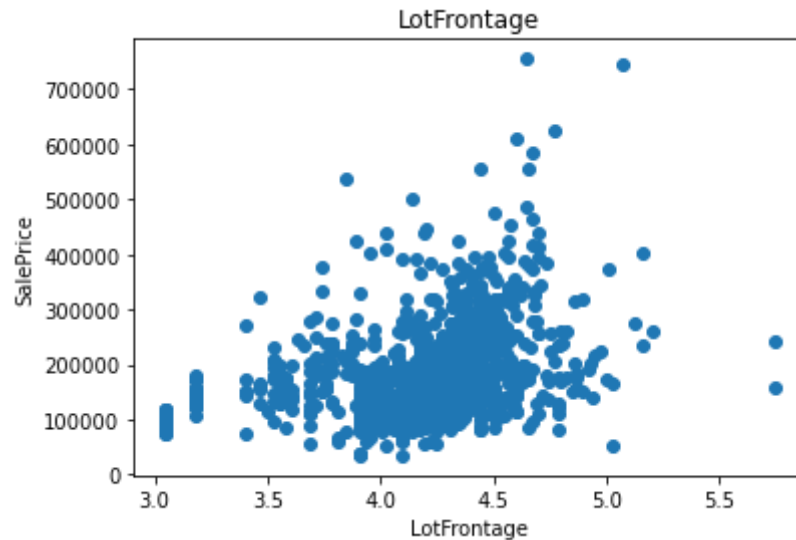


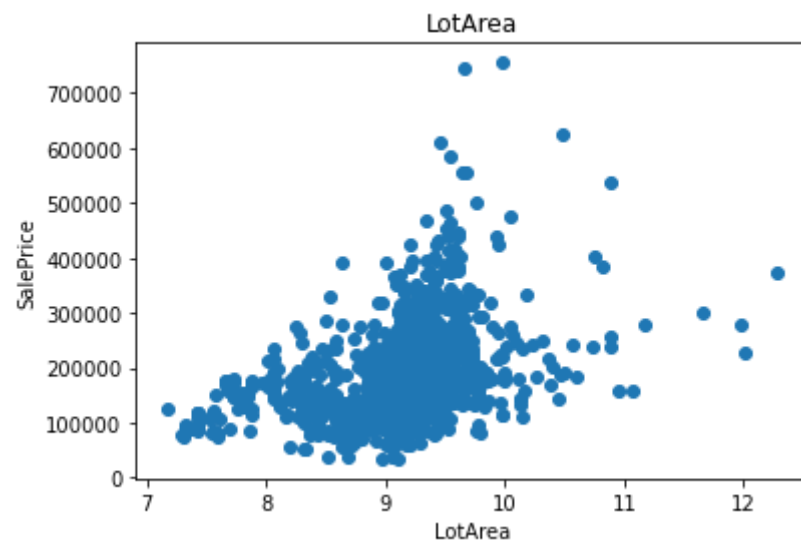
In [18]: *# we can analyze the continuous values by creating histogram to understand the distribution*

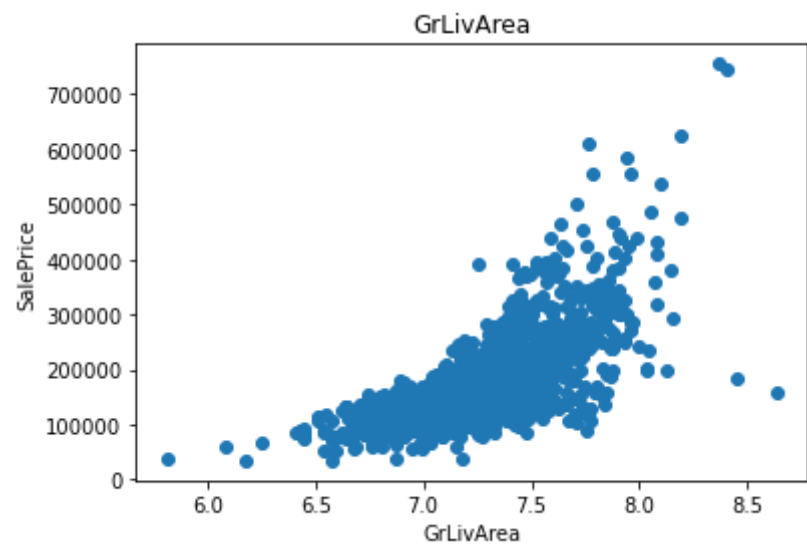
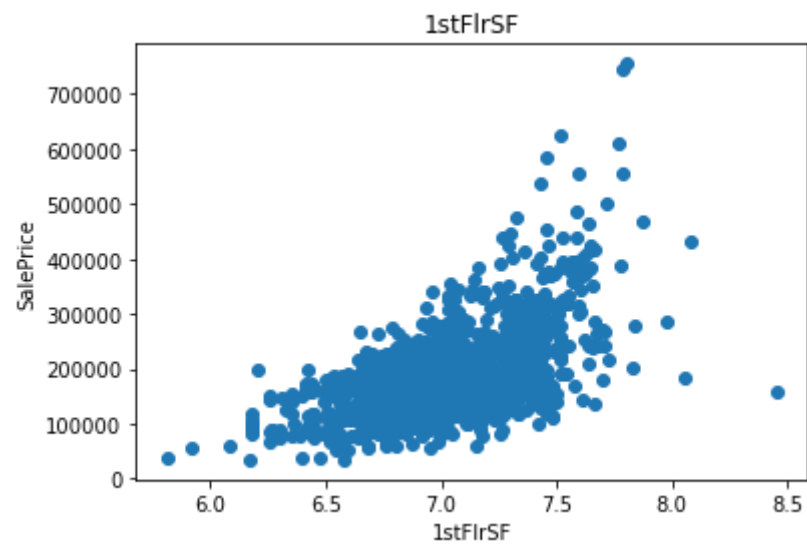
```
data = dataset.copy()
for feature in continuous_feature:
    data[feature].hist(bins=25)
    plt.xlabel(feature)
    plt.ylabel('count')
    plt.title(feature)
    plt.show()
```

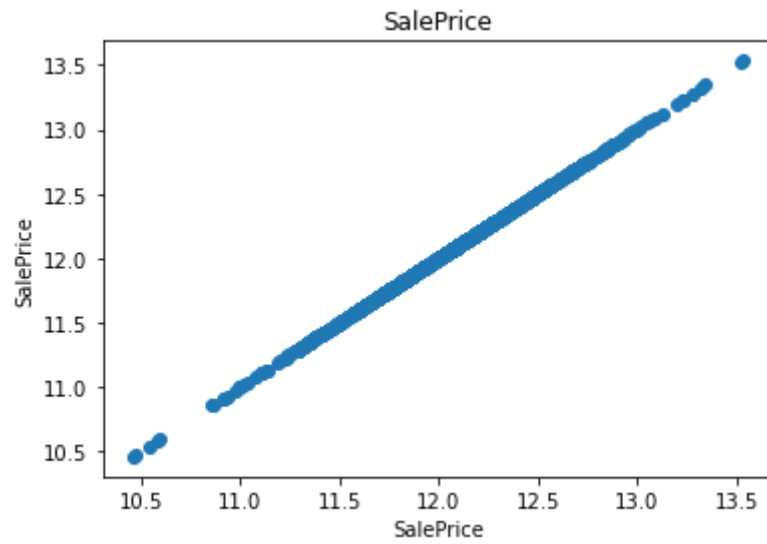


```
In [19]: # Transforming Log normal distribution to normal distribution using logarithmic transformation
data = dataset.copy()
for feature in continuous_feature:
    if 0 in data[feature].unique():
        pass
    else:
        data[feature]=np.log(data[feature])
        plt.scatter(data[feature],data['SalePrice'])
        plt.xlabel(feature)
        plt.ylabel('SalePrice')
        plt.title(feature)
        plt.show()
```



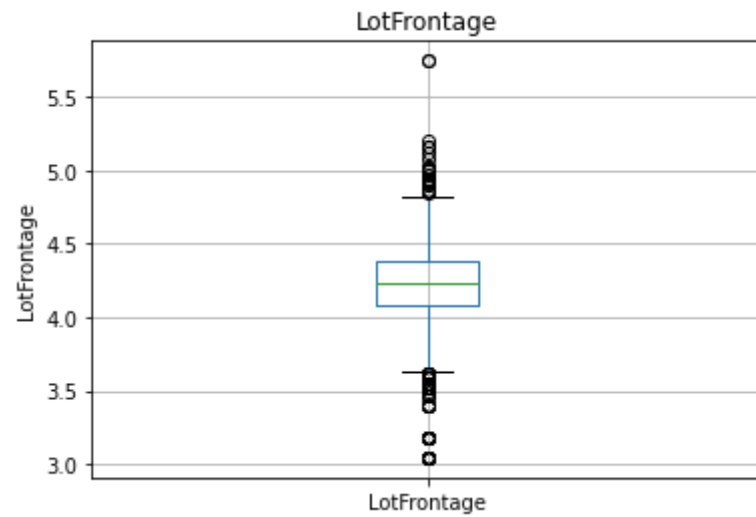


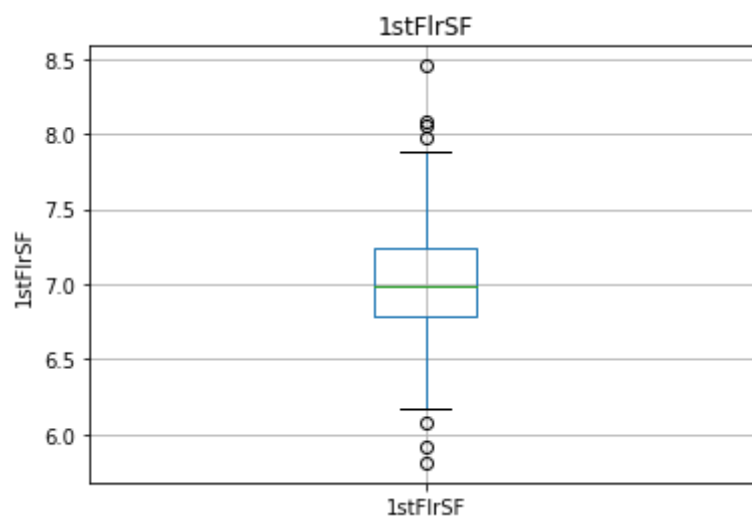
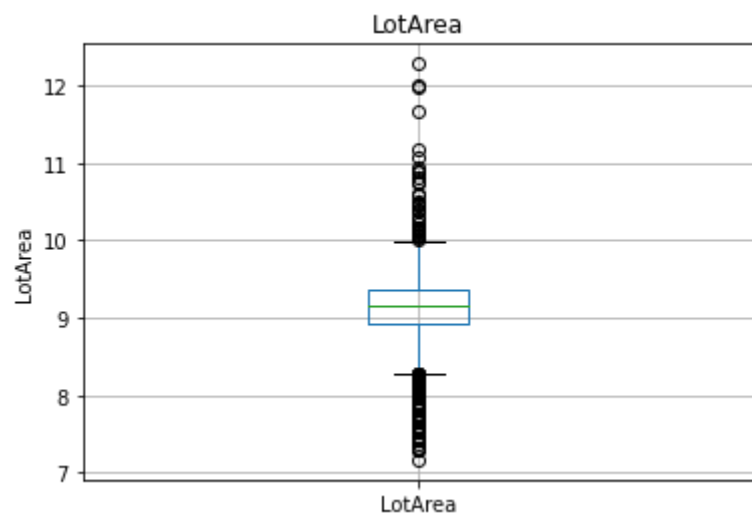


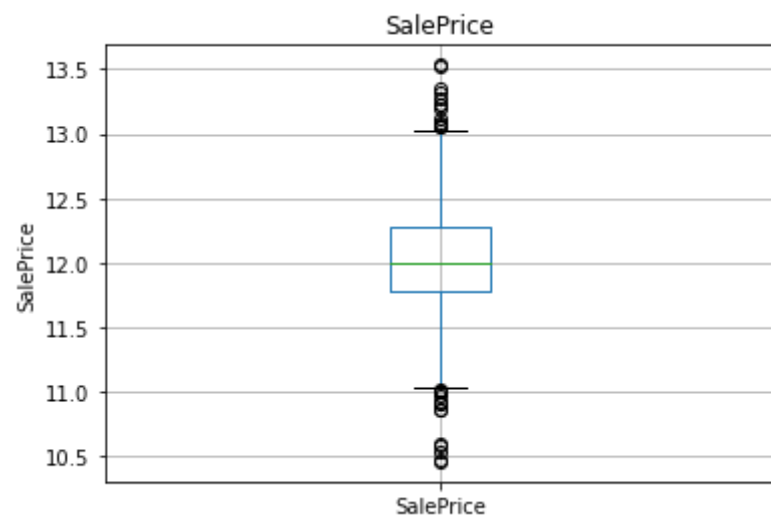
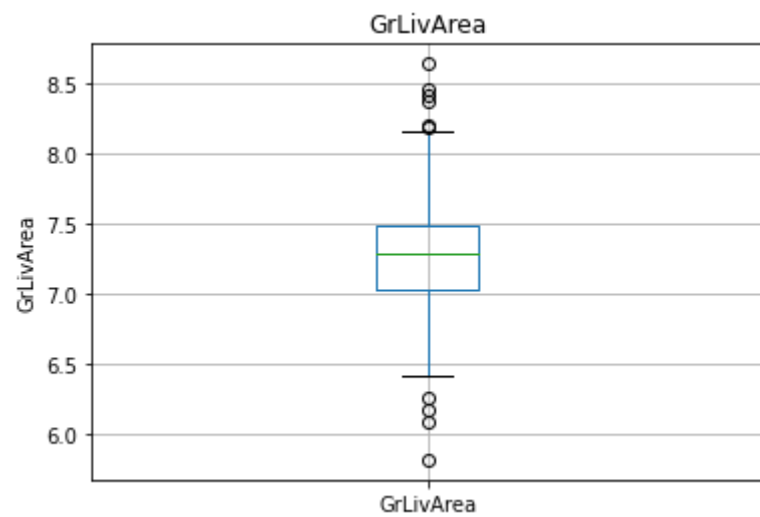


Checking Outliers


```
In [20]: data = dataset.copy()
for feature in continuous_feature:
    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 [21]: categorical_feature = [feature for feature in dataset.columns if dataset[feature].dtype=='O']
categorical_feature
```


```
Out[21]: ['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 [22]: dataset[categorical_feature].head(2)
```

Out[22]:

	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	BldgType	HouseStyle
0	RL	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm	Norm	1Fam	2Story
1	RL	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	Veenker	Feedr	Norm	1Fam	1Story



```
In [23]: for feature in categorical_feature:
          print("The feature name is {} and the number of categories are {}".format(feature, len(dataset[feature].unique())))
```

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

The feature name is PoolQC and the number of categories are 4
The feature name is Fence and the number of categories are 5
The feature name is MiscFeature and the number of categories are 5
The feature name is SaleType and the number of categories are 9
The feature name is SaleCondition and the number of categories are 6

In [24]: *## Analyze the relationship between categorical feature and saleprice*

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

