Regression Analysis

Problem Statement

There are 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, this competition challenges you to predict the final price of each home.

Goal To predict the sales price for each house. For each Id in the test set, you must predict the value of the SalePrice variable.

Metric Submissions are evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price. (Taking logs means that errors in predicting expensive houses and cheap houses will affect the result equally.)

Features

MSSubClass: Identifies the type of dwelling involved in the sale.

MSZoning: Identifies the general zoning classification of the sale.

LotFrontage: Linear feet of street connected to property

LotArea: Lot size in square feet

Street: Type of road access to property

Alley: Type of alley access to property

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 various conditions

Condition2: Proximity to various conditions (if more than one is present)

BldgType: Type of dwelling

HouseStyle: Style of dwelling

OverallQual: Rates the overall material and finish of the house

OverallCond: Rates the overall condition of the house

YearBuilt: Original construction date

YearRemodAdd: Remodel date (same as construction date if no remodeling or additions)

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: Evaluates the quality of the material on the exterior

ExterCond: Evaluates the present condition of the material on the exterior

Foundation: Type of foundation

BsmtQual: Evaluates the height of the basement

BsmtCond: Evaluates the general condition of the basement

BsmtExposure: Refers to walkout or garden level walls

BsmtFinType1: Rating of basement finished area

BsmtFinSF1: Type 1 finished square feet

BsmtFinType2: Rating of basement finished area (if multiple types)

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: Bedrooms above grade (does NOT include basement bedrooms)

Kitchen: Kitchens above grade

KitchenQual: Kitchen quality

TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)

Functional: Home functionality (Assume typical unless deductions are warranted)

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 (MM)

YrSold: Year Sold (YYYY)

SaleType: Type of sale

SaleCondition: Condition of sale

1. Importing the packages

In [1]:

```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set()

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

import warnings
warnings.filterwarnings('ignore')

```

In [2]:

```
df = pd.read_csv('train.csv')
```

In [3]:

```
df.head()
```

Out[3]:

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

In [4]:

```
print(df.shape)
```

```
(1460, 81)
```

In [5]:

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          91 non-null     object
 7   LotShape       1460 non-null   object
 8   LandContour    1460 non-null   object
 9   Utilities      1460 non-null   object
 10  LotConfig      1460 non-null   object
 11  LandSlope      1460 non-null   object
 12  Neighborhood   1460 non-null   object
 13  Condition1     1460 non-null   object
 14  Condition2     1460 non-null   object
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
 21  RoofStyle      1460 non-null   object
 22  RoofMatl       1460 non-null   object
 23  Exterior1st    1460 non-null   object
 24  Exterior2nd    1460 non-null   object
 25  MasVnrType     1452 non-null   object
 26  MasVnrArea     1452 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1423 non-null   object
 31  BsmtCond       1423 non-null   object
 32  BsmtExposure   1422 non-null   object
 33  BsmtFinType1   1423 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1422 non-null   object
 36  BsmtFinSF2     1460 non-null   int64
 37  BsmtUnfSF      1460 non-null   int64
 38  TotalBsmtSF    1460 non-null   int64
 39  Heating        1460 non-null   object
 40  HeatingQC      1460 non-null   object
 41  CentralAir     1460 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1460 non-null   int64
 44  2ndFlrSF       1460 non-null   int64
 45  LowQualFinSF   1460 non-null   int64
 46  GrLivArea      1460 non-null   int64
 47  BsmtFullBath   1460 non-null   int64
 48  BsmtHalfBath   1460 non-null   int64
 49  FullBath       1460 non-null   int64
 50  HalfBath       1460 non-null   int64
 51  BedroomAbvGr   1460 non-null   int64
 52  KitchenAbvGr   1460 non-null   int64
 53  KitchenQual    1460 non-null   object
 54  TotRmsAbvGrd   1460 non-null   int64
 55  Functional     1460 non-null   object
```

```
 56  Fireplaces     1460 non-null   int64
 57  FireplaceQu    770 non-null    object
 58  GarageType     1379 non-null   object
 59  GarageYrBlt    1379 non-null   float64
 60  GarageFinish   1379 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1379 non-null   object
 64  GarageCond     1379 non-null   object
 65  PavedDrive     1460 non-null   object
 66  WoodDeckSF     1460 non-null   int64
 67  OpenPorchSF    1460 non-null   int64
 68  EnclosedPorch  1460 non-null   int64
 69  3SsnPorch      1460 non-null   int64
 70  ScreenPorch    1460 non-null   int64
 71  PoolArea       1460 non-null   int64
 72  PoolQC         7 non-null      object
 73  Fence          281 non-null    object
 74  MiscFeature    54 non-null     object
 75  MiscVal        1460 non-null   int64
 76  MoSold         1460 non-null   int64
 77  YrSold         1460 non-null   int64
 78  SaleType       1460 non-null   object
 79  SaleCondition  1460 non-null   object
 80  SalePrice      1460 non-null   int64
dtypes: float64(3), int64(35), object(43)
memory usage: 924.0+ KB
```

In [6]:

```
1  # Dropping id which has no significance with price
```

In [7]:

```
1  df = df.drop('Id',axis = 1)
```

In [8]:

```
1  # MS Sub CLass is a dwelling type hence converting it to categorical
```

In [9]:

```
1  df['MSSubClass'] = df['MSSubClass'].astype(np.object)
```

In [10]:

```
1  # OverallQual and OverallCond are ordinal variables hence converting them as well to
```

In [11]:

```
1  df['OverallQual'] = df['OverallQual'].astype(np.object)
```

In [12]:

```
df['OverallCond'] = df['OverallCond'].astype(np.object)
```

2. Checking for missing values

In [13]:

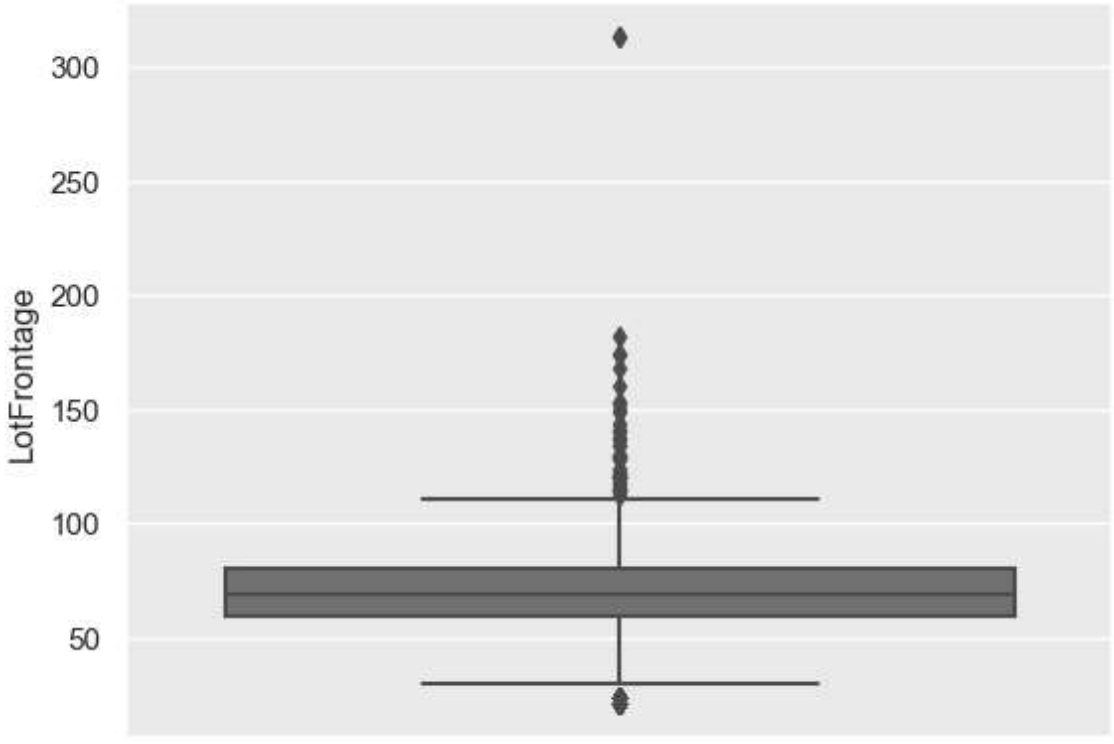
```
1 df.isnull().sum()/len(df)*100
```

Out[13]:

```
MSSubClass        0.000000
MSZoning          0.000000
LotFrontage      17.739726
LotArea           0.000000
Street            0.000000
Alley            93.767123
LotShape          0.000000
LandContour       0.000000
Utilities         0.000000
LotConfig         0.000000
LandSlope         0.000000
Neighborhood      0.000000
Condition1        0.000000
Condition2        0.000000
BldgType          0.000000
HouseStyle        0.000000
OverallQual       0.000000
OverallCond       0.000000
YearBuilt         0.000000
YearRemodAdd      0.000000
RoofStyle         0.000000
RoofMatl          0.000000
Exterior1st       0.000000
Exterior2nd       0.000000
MasVnrType        0.547945
MasVnrArea        0.547945
ExterQual         0.000000
ExterCond         0.000000
Foundation        0.000000
BsmtQual          2.534247
BsmtCond          2.534247
BsmtExposure      2.602740
BsmtFinType1      2.534247
BsmtFinSF1        0.000000
```

In [14]:
```
# Pulling the missing values
```

In [15]:
```
#Checking for outliers for 'LotFrontage'
sns.boxplot(y = 'LotFrontage',data = df )
plt.show()
```




```
GarageType        5.547945
GarageYrBlt       5.547945
GarageFinish      5.547945
GarageCars        0.000000
```

```
GarageArea          0.000000
GarageQual          5.547945
GarageCond          5.547945
PavedDrive          0.000000
WoodDeckSF          0.000000
OpenPorchSF         0.000000
EnclosedPorch       0.000000
3SsnPorch           0.000000
ScreenPorch         0.000000
PoolArea            0.000000
PoolQC             99.520548
Fence              80.753425
MiscFeature        96.301370
MiscVal             0.000000
MoSold              0.000000
YrSold              0.000000
SaleType            0.000000
SaleCondition       0.000000
SalePrice           0.000000
dtype: float64
```

In [16]:

```
df['LotFrontage'] = df['LotFrontage'].fillna(df['LotFrontage'].median())
```

In [17]:

```
# Since the data description mentions that NA is No alley access imputing missing va
df['Alley'] = df['Alley'].fillna('None')
```

In [18]:

```
#Cheking the max value count for 'MasVnrType'
df['MasVnrType'].value_counts()
```

Out[18]:

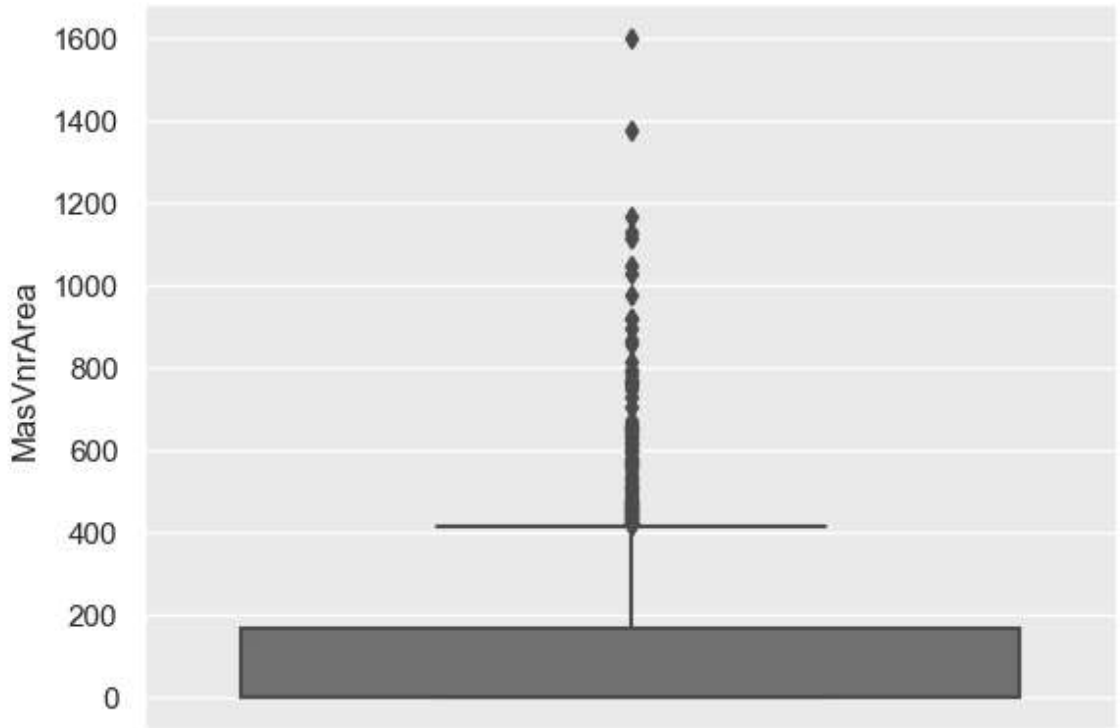
```
None       864
BrkFace    445
Stone      128
BrkCmn      15
Name: MasVnrType, dtype: int64
```

In [19]:

```
#Imputing with mode/max value of 'MasVnrType' - 'None'
df['MasVnrType'] = df['MasVnrType'].fillna('None')
```

In [20]:

```
#Checking for Outlier in 'MasVnrArea'
sns.boxplot(y = 'MasVnrArea' , data = df)
plt.show()
```



In [21]:

```
df['MasVnrArea'] = df['MasVnrArea'].fillna(df['MasVnrArea'].median())
```

In [22]:

```
# Since the data description mentions that NA is No basement imputing missing value
df['BsmtQual'] = df['BsmtQual'].fillna('None')
```

In [23]:

```
# Since the data description mentions that NA is No basement imputing missing value
df['BsmtCond'] = df['BsmtCond'].fillna('None')
```

In [24]:

```
# Since the data description mentions that NA is No basement imputing missing value
df['BsmtExposure'] = df['BsmtExposure'].fillna('None')
```

In [25]:

```
# Since the data description mentions that NA is No basement imputing missing value
df['BsmtFinType1'] = df['BsmtFinType1'].fillna('None')
```

In [26]:

```
# Since the data description mentions that NA is No basement imputing missing value
df['BsmtFinType2'] = df['BsmtFinType2'].fillna('None')
```

In [27]:

```
#Cheking the max value count for 'MasVnrType'
df['Electrical'].value_counts()
```

Out[27]:

```
SBrkr    1334
FuseA      94
FuseF      27
FuseP       3
Mix         1
Name: Electrical, dtype: int64
```

In [28]:

```
#Imputing with mode/max value of 'Electrical' - 'SBrkr'
df['Electrical'] = df['Electrical'].fillna('SBrkr')
```

In [29]:

```
# Since the data description mentions that NA is No Fireplace imputing missing value
df['FireplaceQu'] = df['FireplaceQu'].fillna('None')
```

In [30]:

```
# Since the data description mentions that NA is No Garage imputing missing value as
df['GarageType'] = df['GarageType'].fillna('None')
```

In [31]:

```
# Since the data description mentions that NA is No Garage imputing missing value 0
df['GarageYrBlt'] = df['GarageYrBlt'].fillna(0)
```

In [32]:

```
# Since the data description mentions that NA is No Garage imputing missing value as
df['GarageFinish'] = df['GarageFinish'].fillna('None')
```

In [33]:

```
# Since the data description mentions that NA is No Garage imputing missing value as
df['GarageQual'] = df['GarageQual'].fillna('None')
```

In [34]:

```
# Since the data description mentions that NA is No Garage imputing missing value as
df['GarageCond'] = df['GarageCond'].fillna('None')
```

In [35]:

```
# Since the data description mentions that NA is No Pool imputing missing value as N
df['PoolQC'] = df['PoolQC'].fillna('None')
```

In [36]:

```
# Since the data description mentions that NA is No Fence imputing missing value as
df['Fence'] = df['Fence'].fillna('None')
```

In [37]:

```
# Since the data description mentions that NA is None imputing missing value as None
df['MiscFeature'] = df['MiscFeature'].fillna('None')
```

In [38]:

```
1 df.isnull().sum()/len(df)*100
```

Out[38]:

```
MSSubClass       0.0
MSZoning         0.0
LotFrontage      0.0
LotArea          0.0
Street           0.0
Alley            0.0
LotShape         0.0
LandContour      0.0
Utilities        0.0
LotConfig        0.0
LandSlope        0.0
Neighborhood     0.0
Condition1       0.0
Condition2       0.0
BldgType         0.0
HouseStyle       0.0
OverallQual      0.0
OverallCond      0.0
YearBuilt        0.0
YearRemodAdd     0.0
RoofStyle        0.0
RoofMatl         0.0
Exterior1st      0.0
Exterior2nd      0.0
MasVnrType       0.0
MasVnrArea       0.0
ExterQual        0.0
ExterCond        0.0
Foundation       0.0
BsmtQual         0.0
BsmtCond         0.0
BsmtExposure     0.0
BsmtFinType1     0.0
BsmtFinSF1       0.0
BsmtFinType2     0.0
BsmtFinSF2       0.0
BsmtUnfSF        0.0
TotalBsmtSF      0.0
Heating          0.0
HeatingQC        0.0
CentralAir       0.0
Electrical       0.0
1stFlrSF         0.0
2ndFlrSF         0.0
LowQualFinSF     0.0
GrLivArea        0.0
BsmtFullBath     0.0
BsmtHalfBath     0.0
FullBath         0.0
HalfBath         0.0
BedroomAbvGr     0.0
KitchenAbvGr     0.0
KitchenQual      0.0
TotRmsAbvGrd     0.0
Functional       0.0
Fireplaces       0.0
FireplaceQu      0.0
GarageType       0.0
GarageYrBlt      0.0
GarageFinish     0.0
GarageCars       0.0
```

In [39]:

```
from datetime import datetime
df['CurrentYear'] = datetime.now().year
```

In [40]:

```
#Finding the age of the house
df['HouseBuildingAge'] = df['CurrentYear'] - df['YearBuilt']

#Finding the age of remodelling of the house
df['HouseRemodAge'] = df['CurrentYear'] - df['YearRemodAdd']

#Finding the age of garage in the house
df['GarageAge'] = df['CurrentYear'] - df['GarageYrBlt']

#Finding the age of sale of the house
df['SoldAge'] = df['CurrentYear'] - df['YrSold']

df['GarageAge'].replace(2023,0,inplace = True)

#Dropping 'CurrentYear','YearBuilt','YearRemodAdd','GarageYrBlt','MoSold' and 'YrSol
df = df.drop(['CurrentYear','YearBuilt','YearRemodAdd','GarageYrBlt','YrSold','MoSol
```

In [41]:

```
#Dropping 'BsmtFinSF1','BsmtFinSF2' and 'BsmtUnfSF' as their total is 'TotalBsmtSF'
# Dropping '1stFlrSF' and '2ndFlrSF', 'LowQualFinSF' as their total is 'GrLivArea'
```

In [42]:

```
df = df.drop(['BsmtFinSF1','BsmtFinSF2','BsmtUnfSF','1stFlrSF','2ndFlrSF', 'LowQualF
```

```
GarageArea        0.0
GarageQual        0.0
GarageCond        0.0
PavedDrive        0.0
WoodDeckSF        0.0
OpenPorchSF       0.0
EnclosedPorch     0.0
3SsnPorch         0.0
ScreenPorch       0.0
PoolArea          0.0
PoolQC            0.0
Fence             0.0
MiscFeature       0.0
MiscVal           0.0
MoSold            0.0
YrSold            0.0
SaleType          0.0
SaleCondition     0.0
SalePrice         0.0
dtype: float64
```

3. EDA

In [43]:

```
df.describe().T
```

Out[43]:

	count	mean	std	min	25%	50%	75%
LotFrontage	1460.0	69.863699	22.027677	21.0	60.00	69.0	79.0
LotArea	1460.0	10516.828082	9981.264932	1300.0	7553.50	9478.5	11601.5
MasVnrArea	1460.0	103.117123	180.731373	0.0	0.00	0.0	164.2
TotalBsmtSF	1460.0	1057.429452	438.705324	0.0	795.75	991.5	1298.2
GrLivArea	1460.0	1515.463699	525.480383	334.0	1129.50	1464.0	1776.7
BsmtFullBath	1460.0	0.425342	0.518911	0.0	0.00	0.0	1.0
BsmtHalfBath	1460.0	0.057534	0.238753	0.0	0.00	0.0	0.0
FullBath	1460.0	1.565068	0.550916	0.0	1.00	2.0	2.0
HalfBath	1460.0	0.382877	0.502885	0.0	0.00	0.0	1.0
BedroomAbvGr	1460.0	2.866438	0.815778	0.0	2.00	3.0	3.0
KitchenAbvGr	1460.0	1.046575	0.220338	0.0	1.00	1.0	1.0
TotRmsAbvGrd	1460.0	6.517808	1.625393	2.0	5.00	6.0	7.0
Fireplaces	1460.0	0.613014	0.644666	0.0	0.00	1.0	1.0
GarageCars	1460.0	1.767123	0.747315	0.0	1.00	2.0	2.0
GarageArea	1460.0	472.980137	213.804841	0.0	334.50	480.0	576.0
WoodDeckSF	1460.0	94.244521	125.338794	0.0	0.00	0.0	168.0
OpenPorchSF	1460.0	46.660274	66.256028	0.0	0.00	25.0	68.0
EnclosedPorch	1460.0	21.954110	61.119149	0.0	0.00	0.0	0.0
3SsnPorch	1460.0	3.409589	29.317331	0.0	0.00	0.0	0.0
ScreenPorch	1460.0	15.060959	55.757415	0.0	0.00	0.0	0.0
PoolArea	1460.0	2.758904	40.177307	0.0	0.00	0.0	0.0
MiscVal	1460.0	43.489041	496.123024	0.0	0.00	0.0	0.0
SalePrice	1460.0	180921.195890	79442.502883	34900.0	129975.00	163000.0	214000.0
BuildingAge	1460.0	51.732192	30.202904	13.0	23.00	50.0	69.0
RemodAge	1460.0	38.134247	20.645407	13.0	19.00	29.0	56.0
GarageAge	1460.0	42.025342	26.068188	0.0	20.00	38.5	61.0
AgeSold	1460.0	15.184247	1.328095	13.0	14.00	15.0	16.0

In [44]:

```
1 categorical_columns = (df.select_dtypes(include=['object']).copy())
```

In [45]:

```
1 numeric_columns = (df.select_dtypes(include=['int64', 'float64']).copy())
```

In [46]:

```
1 categorical_columns.head()
```

Out[46]:

	MSSubClass	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandS
0	60	RL	Pave	None	Reg	Lvl	AllPub	Inside	
1	20	RL	Pave	None	Reg	Lvl	AllPub	FR2	
2	60	RL	Pave	None	IR1	Lvl	AllPub	Inside	
3	70	RL	Pave	None	IR1	Lvl	AllPub	Corner	
4	60	RL	Pave	None	IR1	Lvl	AllPub	FR2	

In [47]:

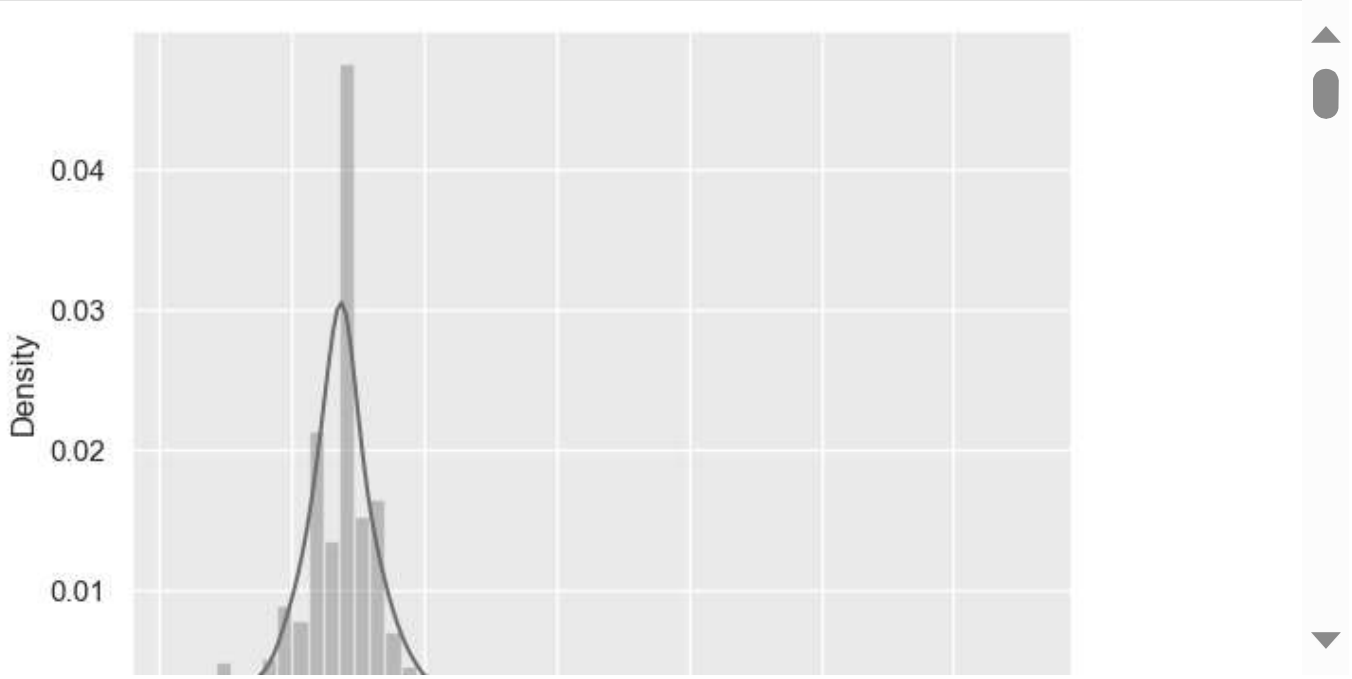
```
1 numeric_columns.head()
```

Out[47]:

	LotFrontage	LotArea	MasVnrArea	TotalBsmtSF	GrLivArea	BsmtFullBath	BsmtHalfBath
0	65.0	8450	196.0	856	1710	1	0
1	80.0	9600	0.0	1262	1262	0	1
2	68.0	11250	162.0	920	1786	1	0
3	60.0	9550	0.0	756	1717	1	0
4	84.0	14260	350.0	1145	2198	1	0

In [48]:

```
# Checking the distplot of the distribution
def distplots(col):
    sns.distplot(numeric_columns[col])
    plt.show()

for i in list(numeric_columns.columns)[0:]:
    distplots(i)
```

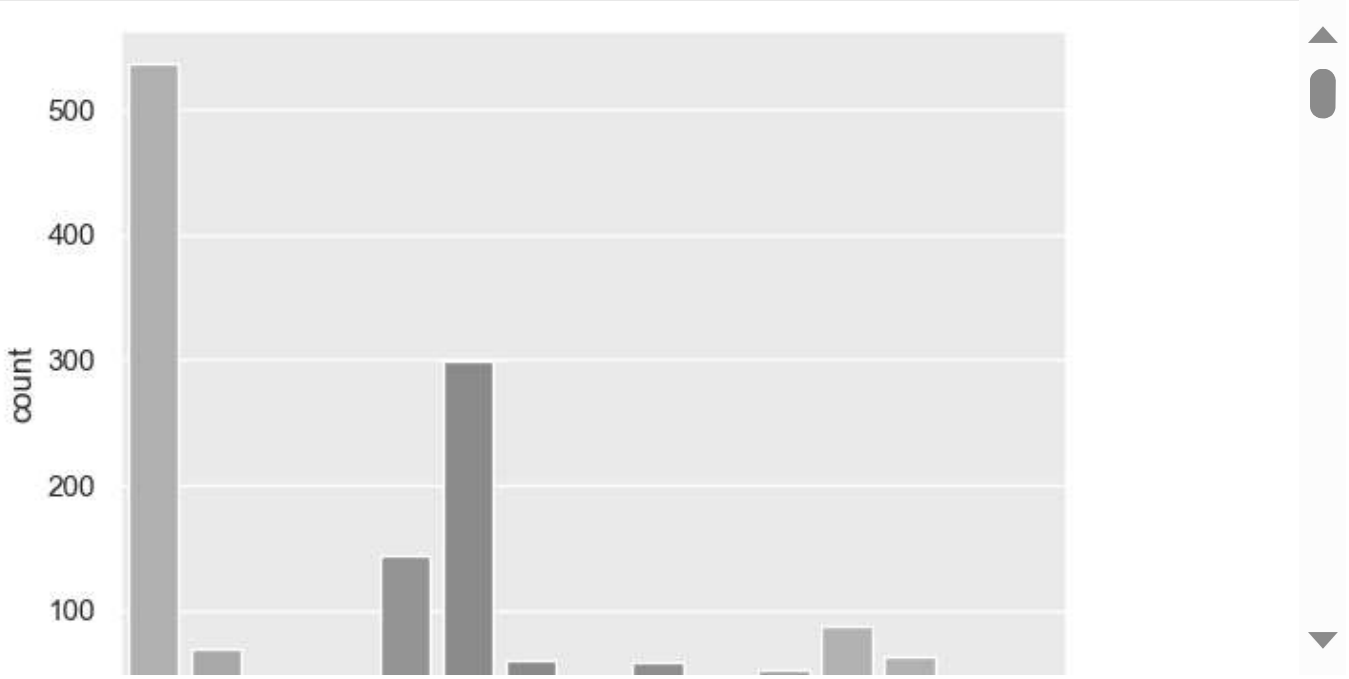


In [49]:

```
# The continuos data columns are skewed towards the right.
# TotalBsmtSF,Garage Area,WoodDesk SF, OpenPorchSF,EnclosedPorchSF,3SsnPorch,
#ScreenPOrch,PoolArea, MiscVal and MasVnrArea columns have a lot of values coming ur
# which means there are houses which do not have these provision provided.
```

In [50]:

```
# Checking the countplot of the categorical data
def countplots(col):

    sns.countplot(categorical_columns[col])
    plt.xticks(rotation=45)
    plt.show()

for i in list(categorical_columns.columns)[0:]:
    countplots(i)
```

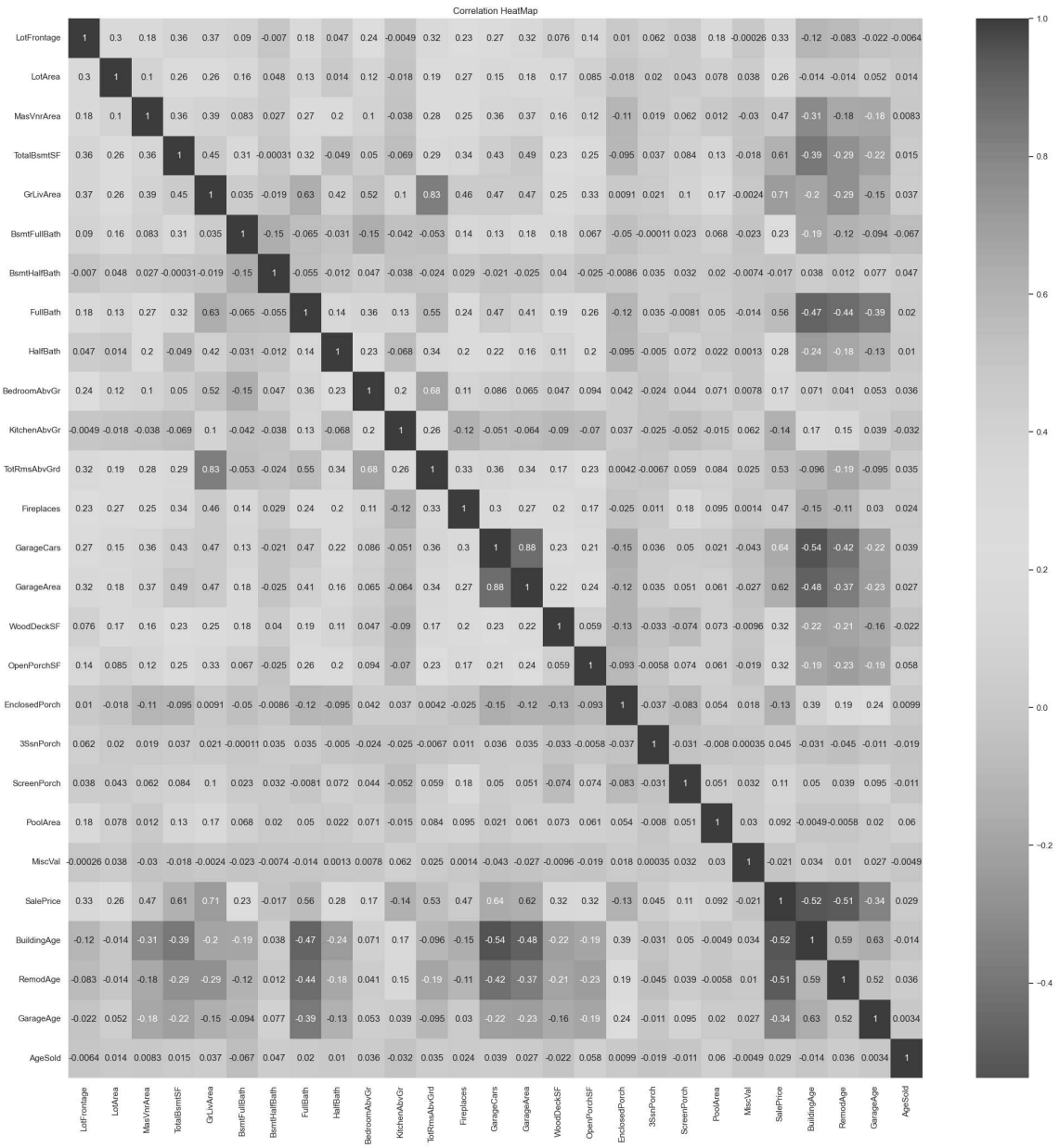


In [51]:

```
# The maximum dwelling count is   20 - 1-STORY 1946 & NEWER ALL STYLES
# The highest count is of Residential Low Density.
# most of the houses have Paved street access.
# 92% of the houses dont have any alley access.
# Most of The shape of the property is regular.
# Most of the houses have Flat Level of property.
# Most of the houses have All public utilities.
# THe max count of the property are situated near North Ames as a neighborhood.
# Most of the houses habe VinylSd as the exterior
# Max houses dont habe any Veneer type.
# Most of the houses with basement and exterior material have average condition
# Most of the houses have gas heating and central air conditining avalable.
# The sale type is Warraty Deed - Conventional Sale type.
```

In [52]:

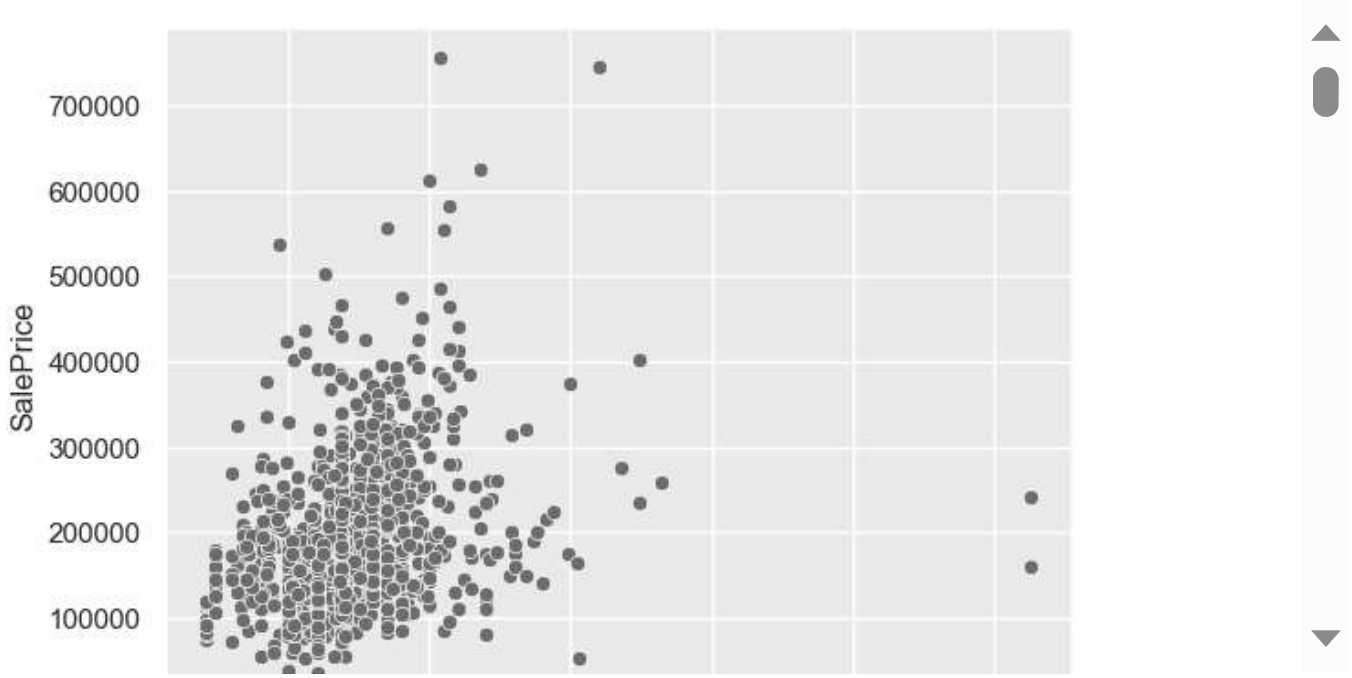
```
# Checking the correlation
df_corr = df.corr()
plt.figure(figsize = (25,25))
sns.heatmap(df_corr,annot=True,cmap = 'coolwarm')
plt.title('Correlation HeatMap')
plt.show()
```



GrLivArea is the most correlated with Sale Price. Next is GarageCars and Garage Area that are correlated with Sale Price. There are variables like LotArea, BsmtFullBath,HalfBAth,BedroomAboveGrd,KitchenAboveGrd,EnclosedPorch,3SsnPorch,ScreenPorch, PoolArea,MiscVal,BuildingAge,RemodAge,GArageAge,Agesold are all very less correlated with Price.

In [53]:

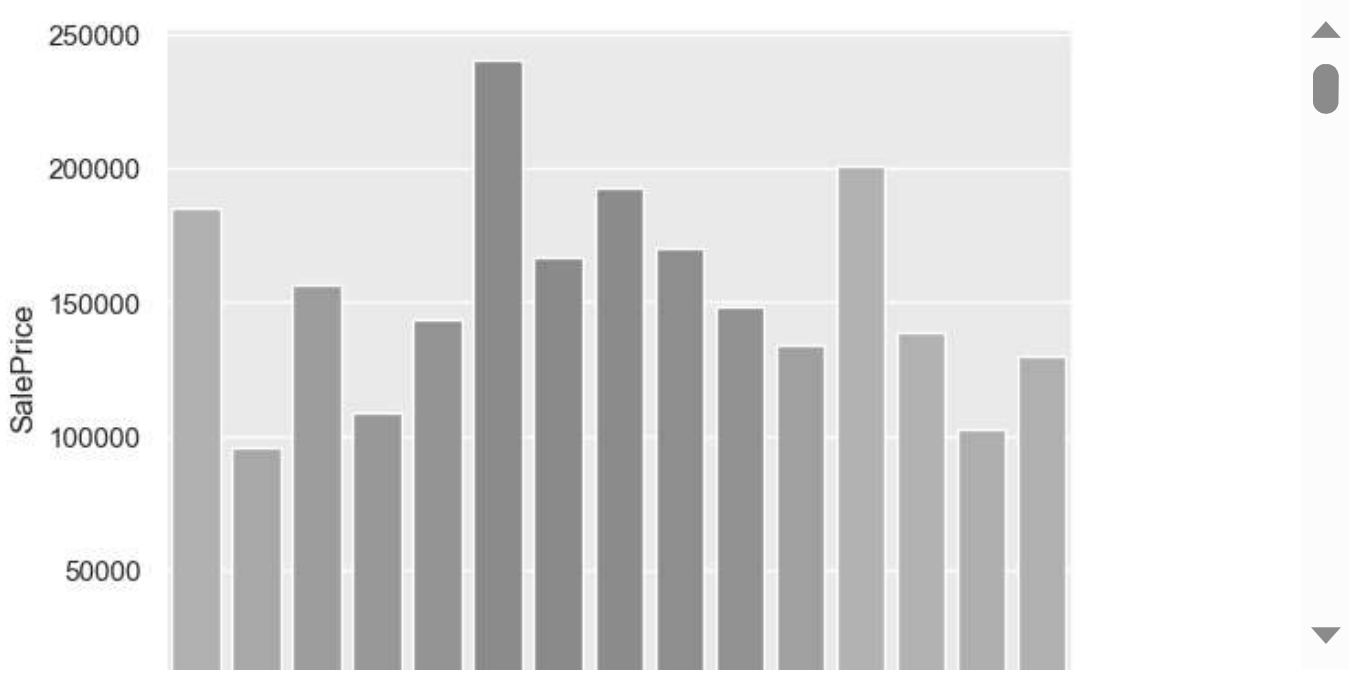
```
# Cheking the relationship between Numeric features and  Dep Variable 'Sale Price'

def scatterplots(col):
    sns.scatterplot(x = numeric_columns[col],y = df['SalePrice'])
    plt.show()

for i in list(numeric_columns.columns)[0:]:
    scatterplots(i)
```



Continous variables such as LotFrontage, MasVnrArea, TotalBsmtSF, GrLivArea,GarageAge,WoodDeckSF,OpenPorchSF have a positive relationship with Price. 3SSnPorch and Pool Area have no relationship with Price. Building Age, Remod Age, Garage Age all have a negative relationship with Price.

In [54]:

```
def barplots(col):

    sns.barplot(x = categorical_columns[col], y = df['SalePrice'],ci=None)
    plt.xticks(rotation=45)
    plt.show()

for i in list(categorical_columns.columns)[0:]:
    barplots(i)
```

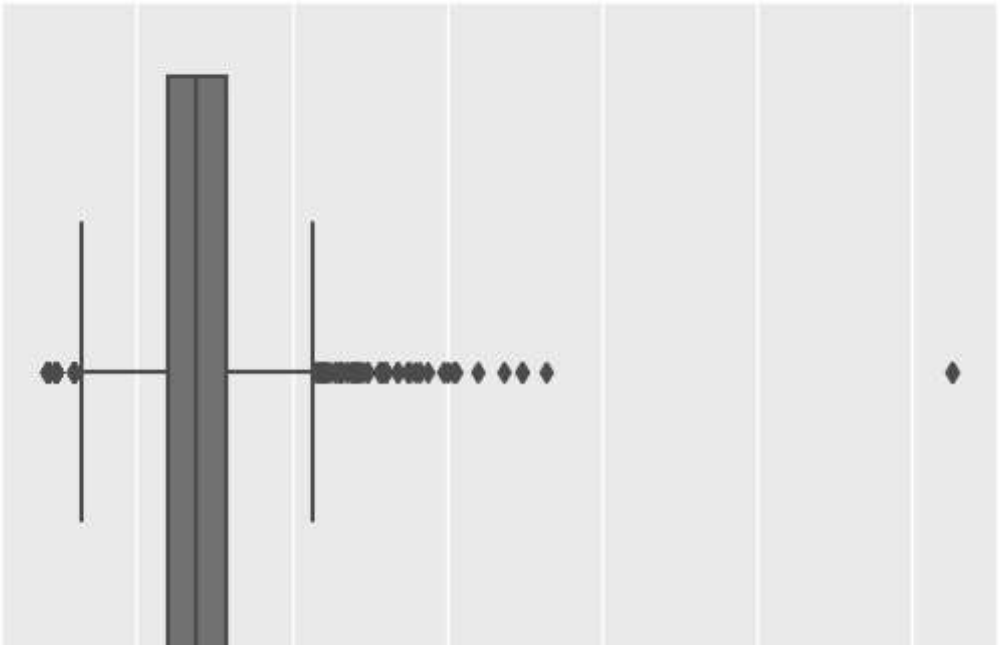


Houses with highest price have a Paved Street. Most of the houses have no Alley Acceess. Houses with highest price All Public Utilities available Houses with highest price are in theCrawford Neighborhood. The houses which are highest priced are 1Fam type houses. All the houses which are priced high have good quality amenities.

4. Check for outliers in the dataset

In [55]:

```
def boxplots(col):
    sns.boxplot(df[col])
    plt.show()

for i in list(df.select_dtypes(exclude = ['object']).columns)[0:]:
    boxplots(i)
```



Not handling the outliers as they seem to be natural outliers.

In [56]:

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

5. Encoding

In [57]:

```
#One hot Encoding for Categorical Variables
```

In [58]:

```
1 categorical_columns.head()
```

Out[58]:

	MSSubClass	MSZoning	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandS
0	60	RL	Pave	None	Reg	Lvl	AllPub	Inside	
1	20	RL	Pave	None	Reg	Lvl	AllPub	FR2	
2	60	RL	Pave	None	IR1	Lvl	AllPub	Inside	
3	70	RL	Pave	None	IR1	Lvl	AllPub	Corner	
4	60	RL	Pave	None	IR1	Lvl	AllPub	FR2	

In [59]:

```
1 categorical_columns = list(categorical_columns)
```

In [60]:

```
1 df_cleaned = pd.concat([data.drop(columns = categorical_columns),pd.get_dummies(data
```

In [61]:

```
1 df_cleaned.head()
```

Out[61]:

	LotFrontage	LotArea	MasVnrArea	TotalBsmtSF	GrLivArea	BsmtFullBath	BsmtHalfBath
0	65.0	8450	196.0	856	1710	1	0
1	80.0	9600	0.0	1262	1262	0	1
2	68.0	11250	162.0	920	1786	1	0
3	60.0	9550	0.0	756	1717	1	0
4	84.0	14260	350.0	1145	2198	1	0

In [62]:

```
#Dropping n+1 Dummy Variables
df_cleaned = df_cleaned.drop(['MSSubClass_20','MSZoning_C (all)','Street_Grvl','Alle
                              'LandContour_Bnk','Utilities_AllPub','LotConfig_Corner
                              'Neighborhood_Blmngtn','Condition1_Artery','Condition2
                              'HouseStyle_1.5Fin','OverallQual_1','OverallCond_1','H
                              'RoofMatl_ClyTile','Exterior1st_AsbShng','Exterior2nd_
                              'MasVnrType_BrkCmn','ExterQual_Ex','Foundation_BrkTil'
                              'BsmtExposure_Av','BsmtFinType1_ALQ','BsmtFinType2_ALQ
                              'CentralAir_N','Electrical_FuseA','KitchenQual_Ex','Fu
                              'GarageType_2Types','GarageFinish_Fin','GarageQual_Ex'
                              'PavedDrive_N','PoolQC_Ex','Fence_GdPrv','MiscFeature_
                              'SaleType_COD','SaleCondition_Abnorml'],axis = 1)
```

In [63]:

```
df_cleaned.shape
```

Out[63]:

```
(1460, 282)
```

In [64]:

```
df_cleaned.head()
```

Out[64]:

	LotFrontage	LotArea	MasVnrArea	TotalBsmtSF	GrLivArea	BsmtFullBath	BsmtHalfBath
0	65.0	8450	196.0	856	1710	1	0
1	80.0	9600	0.0	1262	1262	0	1
2	68.0	11250	162.0	920	1786	1	0
3	60.0	9550	0.0	756	1717	1	0
4	84.0	14260	350.0	1145	2198	1	0

In [65]:

```
df_cleaned.shape
```

Out[65]:

```
(1460, 282)
```

6. Splitting the data

In [66]:

```
x = df_cleaned.drop('SalePrice',axis = 1)
y = df_cleaned[['SalePrice']]
```

7. Train Test Split

In [67]:

```
from sklearn.model_selection import train_test_split
```

In [68]:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_sta
```

In [69]:

```
x_train.shape,x_test.shape,y_train.shape, y_test.shape
```

Out[69]:

```
((1095, 281), (365, 281), (1095, 1), (365, 1))
```

Importing Regression Tools

In [70]:

```
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_percentage_e
```

In [71]:

```
#Linear Regression

linear = LinearRegression()
lr = linear.fit(x_train,y_train)
y_pred_lr_test = linear.predict(x_test)
y_pred_lr_train = linear.predict(x_train)
r2_score_lr_test = r2_score(y_test, y_pred_lr_test)
r2_score_lr_train =  r2_score(y_train, y_pred_lr_train)
mape_lr_test = mean_absolute_percentage_error(y_test, y_pred_lr_test)

# Ridge

ridge = Ridge(alpha = 0.3)
rid = ridge.fit(x_train,y_train)
y_pred_rid_test = ridge.predict(x_test)
y_pred_rid_train = ridge.predict(x_train)
r2_score_rid_test = r2_score(y_test, y_pred_rid_test)
r2_score_rid_train =  r2_score(y_train, y_pred_rid_train)
mape_rid_test = mean_absolute_percentage_error(y_test, y_pred_rid_test)

#Lasso

lasso = Lasso(alpha = 0.1)
las = lasso.fit(x_train,y_train)
y_pred_las_test = lasso.predict(x_test)
y_pred_las_train = lasso.predict(x_train)
r2_score_las_test = r2_score(y_test, y_pred_las_test)
r2_score_las_train =  r2_score(y_train, y_pred_las_train)
mape_las_test = mean_absolute_percentage_error(y_test, y_pred_las_test)

# RandomForest
rfmodel = RandomForestRegressor()
rf = rfmodel.fit(x_train, y_train)
y_pred_rf_test = rfmodel.predict(x_test)
y_pred_rf_train = rfmodel.predict(x_train)
r2_score_rf_test = r2_score(y_test, y_pred_rf_test)
r2_score_rf_train =  r2_score(y_train, y_pred_rf_train)
mape_rf_test = mean_absolute_percentage_error(y_test, y_pred_rf_test)


#SVR

svr = SVR()
sv = svr.fit(x_train, y_train)
y_pred_sv_test = svr.predict(x_test)
y_pred_sv_train = svr.predict(x_train)
r2_score_sv_test = r2_score(y_test, y_pred_sv_test)
r2_score_sv_train =  r2_score(y_train, y_pred_sv_train)
mape_sv_test = mean_absolute_percentage_error(y_test, y_pred_sv_test)

#XGBRegressor
xgb = XGBRegressor()
xg = xgb.fit(x_train, y_train)
y_pred_xg_test = xgb.predict(x_test)
y_pred_xg_train = xgb.predict(x_train)
r2_score_xg_test = r2_score(y_test, y_pred_xg_test)
r2_score_xg_train =  r2_score(y_train, y_pred_xg_train)
mape_xg_test = mean_absolute_percentage_error(y_test, y_pred_xg_test)
```

```
59
```

In [72]:

```
print(r2_score_lr_train)
print (r2_score_lr_test)
print(mape_lr_test)
```

```
0.9407533865163138
0.5649252608784104
0.10990482860182876
```

In [73]:

```
print(r2_score_rid_train)
print (r2_score_rid_test)
print(mape_rid_test)
```

```
0.93810691340121
0.6734863778707227
0.11344739094933125
```

In [74]:

```
print(r2_score_las_train)
print (r2_score_las_test)
print(mape_las_test)
```

```
0.9397357636456941
0.7491308659566068
0.10894091500214093
```

In [75]:

```
print(r2_score_rf_train)
print (r2_score_rf_test)
print(mape_rf_test)
```

```
0.9769272051068553
0.8353640581966604
0.11298083717584533
```

In [76]:

```
print(r2_score_sv_train)
print (r2_score_sv_test)
print(mape_sv_test)
```

```
-0.04546049423139498
-0.024853598254753972
0.33733799529288466
```

In [77]:

```
print(r2_score_xg_train)
print (r2_score_xg_test)
print(mape_xg_test)
```

```
0.9995923129886186
0.8532604751081578
0.11191874852146544
```

8. Feature Selection using VIF

In [78]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [79]:

```
variable = x

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x.columns
```

In [80]:

```
vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[80]:

	Variance Inflation Factor	Features
100	inf	BldgType_Duplex
126	inf	OverallCond_9
247	inf	GarageQual_None
242	inf	GarageFinish_None
241	inf	GarageType_None
119	inf	OverallCond_2
120	inf	OverallCond_3
121	inf	OverallCond_4
122	inf	OverallCond_5
123	inf	OverallCond_6
124	inf	OverallCond_7

In [81]:

```
vif_threshold = 5
```

In [82]:

```
x_vif1 = x.drop('BldgType_Duplex',axis = 1)
```

In [83]:

```
variable = x_vif1

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif1.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[83]:

	Variance Inflation Factor	Features
124	inf	OverallCond_8
198	inf	BsmtFinType1_None
189	inf	BsmtCond_Po
125	inf	OverallCond_9
188	inf	BsmtCond_None
219	inf	Electrical_Mix
185	inf	BsmtQual_None
123	inf	OverallCond_7
122	inf	OverallCond_6
121	inf	OverallCond_5
177	inf	ExterCond_TA

In [84]:

```
x_vif2 = x_vif1.drop('OverallCond_8',axis = 1)
```

In [85]:

```
variable = x_vif2

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif2.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[85]:

	Variance Inflation Factor	Features
240	inf	GarageFinish_None
250	inf	GarageCond_None
154	inf	Exterior2nd_CBlock
187	inf	BsmtCond_None
140	inf	Exterior1st_CBlock
245	inf	GarageQual_None
197	inf	BsmtFinType1_None
184	inf	BsmtQual_None
239	inf	GarageType_None
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd

In [86]:

```
x_vif3 = x_vif2.drop('GarageFinish_None',axis = 1)
```

In [87]:

```
variable = x_vif3

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif3.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[87]:

	Variance Inflation Factor	Features
197	inf	BsmtFinType1_None
249	inf	GarageCond_None
154	inf	Exterior2nd_CBlock
187	inf	BsmtCond_None
239	inf	GarageType_None
184	inf	BsmtQual_None
244	inf	GarageQual_None
140	inf	Exterior1st_CBlock
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd
173	1.242618e+03	ExterCond_Fa

In [88]:

```
x_vif4 = x_vif3.drop('BsmtFinType1_None',axis = 1)
```

In [89]:

```
variable = x_vif4

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif4.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[89]:

	Variance Inflation Factor	Features
243	inf	GarageQual_None
187	inf	BsmtCond_None
248	inf	GarageCond_None
184	inf	BsmtQual_None
154	inf	Exterior2nd_CBlock
238	inf	GarageType_None
140	inf	Exterior1st_CBlock
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd
173	1.242618e+03	ExterCond_Fa
260	1.037134e+03	MiscFeature_None

In [90]:

```
x_vif5 = x_vif4.drop('GarageQual_None',axis = 1)
```

In [91]:

```
variable = x_vif5

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif5.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[91]:

	Variance Inflation Factor	Features
184	inf	BsmtQual_None
140	inf	Exterior1st_CBlock
247	inf	GarageCond_None
187	inf	BsmtCond_None
154	inf	Exterior2nd_CBlock
238	inf	GarageType_None
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd
173	1.242618e+03	ExterCond_Fa
259	1.037134e+03	MiscFeature_None
261	8.650604e+02	MiscFeature_Shed

In [92]:

```
x_vif6 = x_vif5.drop('BsmtQual_None',axis = 1)
```

In [93]:

```
variable = x_vif6

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif6.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[93]:

	Variance Inflation Factor	Features
246	inf	GarageCond_None
237	inf	GarageType_None
154	inf	Exterior2nd_CBlock
140	inf	Exterior1st_CBlock
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd
173	1.242618e+03	ExterCond_Fa
258	1.037134e+03	MiscFeature_None
260	8.650604e+02	MiscFeature_Shed
112	4.765304e+02	OverallQual_5
113	4.604885e+02	OverallQual_6

In [94]:

```
x_vif7 = x_vif6.drop('GarageCond_None',axis = 1)
```

In [95]:

```
variable = x_vif7

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif7.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[95]:

	Variance Inflation Factor	Features
140	inf	Exterior1st_CBlock
154	inf	Exterior2nd_CBlock
176	5.675726e+04	ExterCond_TA
174	6.477468e+03	ExterCond_Gd
173	1.242618e+03	ExterCond_Fa
257	1.037134e+03	MiscFeature_None
259	8.650604e+02	MiscFeature_Shed
112	4.765304e+02	OverallQual_5
113	4.604885e+02	OverallQual_6
114	4.127049e+02	OverallQual_7
247	3.044060e+02	GarageCond_TA

In [96]:

```
x_vif8 = x_vif7.drop('Exterior1st_CBlock',axis = 1)
```

In [97]:

```
variable = x_vif8

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif8.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[97]:

	Variance Inflation Factor	Features
175	56757.256491	ExterCond_TA
173	6477.468112	ExterCond_Gd
172	1242.617992	ExterCond_Fa
256	1037.134407	MiscFeature_None
258	865.060435	MiscFeature_Shed
112	476.530413	OverallQual_5
113	460.488512	OverallQual_6
114	412.704930	OverallQual_7
246	304.405985	GarageCond_TA
242	255.187536	GarageQual_TA
115	247.939615	OverallQual_8

In [98]:

```
x_vif9 = x_vif8.drop('ExterCond_TA',axis = 1)
```

In [99]:

```
variable = x_vif9

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif9.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[99]:

	Variance Inflation Factor	Features
250	29646.569678	PoolQC_None
255	17400.294107	MiscFeature_None
130	7524.781160	RoofMatl_CompShg
245	3310.775523	GarageCond_TA
241	2500.422651	GarageQual_TA
202	1850.224561	Heating_GasA
93	1677.162317	Condition2_Norm
125	734.139248	RoofStyle_Gable
112	631.896084	OverallQual_5
113	597.421254	OverallQual_6
257	555.790161	MiscFeature_Shed

In [100]:

```
x_vif10 = x_vif9.drop('PoolQC_None',axis = 1)
```

In [101]:

```
variable = x_vif10

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif10.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[101]:

	Variance Inflation Factor	Features
254	10734.366139	MiscFeature_None
130	3774.526693	RoofMatl_CompShg
245	3306.825441	GarageCond_TA
241	2500.109038	GarageQual_TA
202	1807.438011	Heating_GasA
93	1666.621254	Condition2_Norm
125	733.894499	RoofStyle_Gable
112	625.779736	OverallQual_5
113	591.990628	OverallQual_6
114	504.690099	OverallQual_7
44	477.583254	Street_Pave

In [102]:

```
x_vif11 = x_vif10.drop('MiscFeature_None',axis = 1)
```

In [103]:

```
variable = x_vif11

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif11.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[103]:

	Variance Inflation Factor	Features
245	3292.833088	GarageCond_TA
130	2950.427678	RoofMatl_CompShg
241	2496.094793	GarageQual_TA
202	1641.874333	Heating_GasA
93	1484.759637	Condition2_Norm
125	706.415686	RoofStyle_Gable
112	549.193961	OverallQual_5
113	520.864092	OverallQual_6
44	463.227460	Street_Pave
114	443.683831	OverallQual_7
42	251.704494	MSZoning_RL

In [104]:

```
x_vif12 = x_vif11.drop('GarageCond_TA',axis = 1)
```

In [105]:

```
variable = x_vif12

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif12.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[105]:

	Variance Inflation Factor	Features
130	2927.647437	RoofMatl_CompShg
202	1640.451458	Heating_GasA
93	1478.776107	Condition2_Norm
125	706.098874	RoofStyle_Gable
112	548.720480	OverallQual_5
241	545.351303	GarageQual_TA
113	520.499680	OverallQual_6
44	463.196972	Street_Pave
114	443.408280	OverallQual_7
42	251.341683	MSZoning_RL
115	236.067654	OverallQual_8

In [106]:

```
x_vif13 = x_vif12.drop('RoofMatl_CompShg',axis = 1)
```

In [107]:

```
variable = x_vif13

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif13.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[107]:

	Variance Inflation Factor	Features
201	1532.440229	Heating_GasA
93	1420.981016	Condition2_Norm
125	693.264523	RoofStyle_Gable
240	535.593382	GarageQual_TA
112	499.558230	OverallQual_5
113	472.537210	OverallQual_6
44	459.437037	Street_Pave
114	402.344369	OverallQual_7
42	246.115298	MSZoning_RL
229	217.517878	GarageType_Attchd
115	214.316517	OverallQual_8

In [108]:

```
x_vif14 = x_vif13.drop('Heating_GasA',axis = 1)
```

In [109]:

```
variable = x_vif14

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif14.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[109]:

	Variance Inflation Factor	Features
93	1401.245054	Condition2_Norm
125	677.989102	RoofStyle_Gable
239	525.976370	GarageQual_TA
112	461.637160	OverallQual_5
44	455.923747	Street_Pave
113	437.068640	OverallQual_6
114	372.722172	OverallQual_7
42	244.853343	MSZoning_RL
228	214.345186	GarageType_Attchd
115	198.971089	OverallQual_8
146	194.151362	Exterior1st_VinylSd

In [110]:

```
x_vif15 = x_vif14.drop('Condition2_Norm',axis = 1)
```

In [111]:

```
variable = x_vif15

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif15.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[111]:

	Variance Inflation Factor	Features
124	661.656962	RoofStyle_Gable
238	513.112676	GarageQual_TA
44	451.488726	Street_Pave
111	403.770854	OverallQual_5
112	381.186594	OverallQual_6
113	325.564798	OverallQual_7
42	244.417583	MSZoning_RL
227	212.977786	GarageType_Attchd
145	193.645445	Exterior1st_VinylSd
114	174.149243	OverallQual_8
160	172.960230	Exterior2nd_VinylSd

In [112]:

```
x_vif16 = x_vif15.drop('RoofStyle_Gable',axis = 1)
```

In [113]:

```
variable = x_vif16

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif16.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[113]:

	Variance Inflation Factor	Features
237	506.019902	GarageQual_TA
44	448.603653	Street_Pave
111	373.911203	OverallQual_5
112	353.552417	OverallQual_6
113	302.210577	OverallQual_7
42	241.970537	MSZoning_RL
226	212.157984	GarageType_Attchd
144	193.542059	Exterior1st_VinylSd
159	172.835282	Exterior2nd_VinylSd
25	170.528436	AgeSold
114	162.059493	OverallQual_8

In [114]:

```
x_vif17 = x_vif16.drop('GarageQual_TA',axis = 1)
```

In [115]:

```
variable = x_vif17

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif17.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[115]:

	Variance Inflation Factor	Features
44	446.826025	Street_Pave
111	334.680063	OverallQual_5
112	316.589436	OverallQual_6
113	270.645244	OverallQual_7
42	241.524182	MSZoning_RL
226	210.708106	GarageType_Attchd
144	193.462814	Exterior1st_VinylSd
159	172.714460	Exterior2nd_VinylSd
25	169.630848	AgeSold
220	149.573446	Functional_Typ
114	145.953336	OverallQual_8

In [116]:

```
x_vif18 = x_vif17.drop('Street_Pave',axis = 1)
```

In [117]:

```
variable = x_vif18

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif18.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[117]:

	Variance Inflation Factor	Features
110	333.883228	OverallQual_5
111	315.706652	OverallQual_6
112	269.791710	OverallQual_7
42	226.433950	MSZoning_RL
225	194.305043	GarageType_Attchd
143	192.000671	Exterior1st_VinylSd
158	172.299851	Exterior2nd_VinylSd
25	169.067641	AgeSold
219	149.306632	Functional_Typ
113	145.437353	OverallQual_8
4	144.896460	GrLivArea

In [118]:

```
x_vif19 = x_vif18.drop('OverallQual_5',axis = 1)
```

In [119]:

```
variable = x_vif19

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif19.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[119]:

	Variance Inflation Factor	Features
42	215.344229	MSZoning_RL
142	191.571524	Exterior1st_VinylSd
224	175.362945	GarageType_Attchd
157	172.142381	Exterior2nd_VinylSd
25	165.530861	AgeSold
4	144.853019	GrLivArea
218	144.531692	Functional_Typ
11	117.736742	TotRmsAbvGrd
10	117.642904	KitchenAbvGr
138	99.185655	Exterior1st_MetalSd
101	94.760109	HouseStyle_1Story

In [120]:

```
x_vif20 = x_vif19.drop('MSZoning_RL',axis = 1)
```

In [121]:

```
variable = x_vif20

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif20.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[121]:

	Variance Inflation Factor	Features
141	191.544268	Exterior1st_VinylSd
223	172.910847	GarageType_Attchd
156	171.922036	Exterior2nd_VinylSd
25	162.499324	AgeSold
4	144.840771	GrLivArea
217	141.205434	Functional_Typ
11	117.633048	TotRmsAbvGrd
10	116.087723	KitchenAbvGr
137	99.168188	Exterior1st_MetalSd
100	92.937202	HouseStyle_1Story
178	91.883592	BsmtCond_None

In [122]:

```
x_vif21 = x_vif20.drop('Exterior1st_VinylSd',axis = 1)
```

In [123]:

```
variable = x_vif21

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif21.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[123]:

	Variance Inflation Factor	Features
222	168.586491	GarageType_Attchd
25	162.247937	AgeSold
4	144.783205	GrLivArea
216	140.641361	Functional_Typ
11	117.365978	TotRmsAbvGrd
10	114.530683	KitchenAbvGr
100	92.923115	HouseStyle_1Story
177	91.851925	BsmtCond_None
194	84.973340	BsmtFinType2_Unf
159	77.704488	MasVnrType_None
226	75.041527	GarageType_Detchd

In [124]:

```
x_vif22 = x_vif21.drop('GarageType_Attchd',axis = 1)
```

In [125]:

```
variable = x_vif22

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif22.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[125]:

	Variance Inflation Factor	Features
25	160.952642	AgeSold
4	144.682074	GrLivArea
216	135.670235	Functional_Typ
11	115.859909	TotRmsAbvGrd
10	114.032695	KitchenAbvGr
100	92.415238	HouseStyle_1Story
177	91.822313	BsmtCond_None
194	83.818512	BsmtFinType2_Unf
159	76.782911	MasVnrType_None
22	72.065175	BuildingAge
150	58.867106	Exterior2nd_MetalSd

In [126]:

```
x_vif23 = x_vif22.drop('AgeSold',axis = 1)
```

In [127]:

```
variable = x_vif23

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif23.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[127]:

	Variance Inflation Factor	Features
4	144.506135	GrLivArea
215	135.515560	Functional_Typ
11	115.601656	TotRmsAbvGrd
10	114.004180	KitchenAbvGr
176	91.822250	BsmtCond_None
99	90.550536	HouseStyle_1Story
193	82.880514	BsmtFinType2_Unf
158	76.451021	MasVnrType_None
22	71.999100	BuildingAge
149	58.593924	Exterior2nd_MetalSd
250	57.366187	SaleType_New

In [128]:

```
x_vif24 = x_vif23.drop('Functional_Typ',axis = 1)
```

In [129]:

```
variable = x_vif24

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif24.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[129]:

	Variance Inflation Factor	Features
4	142.546584	GrLivArea
11	115.463546	TotRmsAbvGrd
10	113.923321	KitchenAbvGr
176	91.684573	BsmtCond_None
99	89.871707	HouseStyle_1Story
193	82.189010	BsmtFinType2_Unf
158	75.013608	MasVnrType_None
22	71.302706	BuildingAge
149	58.454077	Exterior2nd_MetalSd
249	57.366162	SaleType_New
3	55.738502	TotalBsmtSF

In [130]:

```
x_vif25 = x_vif24.drop('TotRmsAbvGrd',axis = 1)
```

In [131]:

```
variable = x_vif25

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif25.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[131]:

	Variance Inflation Factor	Features
4	114.556970	GrLivArea
10	110.009820	KitchenAbvGr
175	91.681093	BsmtCond_None
98	89.406841	HouseStyle_1Story
192	82.107633	BsmtFinType2_Unf
157	74.988207	MasVnrType_None
21	71.240725	BuildingAge
148	58.335558	Exterior2nd_MetalSd
248	57.341740	SaleType_New
3	55.431959	TotalBsmtSF
161	55.145908	ExterQual_TA

In [132]:

```
x_vif26 = x_vif25.drop('KitchenAbvGr',axis = 1)
```

In [133]:

```
variable = x_vif26

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif26.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[133]:

	Variance Inflation Factor	Features
4	113.746631	GrLivArea
174	91.589990	BsmtCond_None
97	89.403908	HouseStyle_1Story
191	81.836073	BsmtFinType2_Unf
156	74.220825	MasVnrType_None
20	70.048055	BuildingAge
147	58.289728	Exterior2nd_MetalSd
247	57.302001	SaleType_New
3	55.405618	TotalBsmtSF
160	55.141200	ExterQual_TA
254	54.212886	SaleCondition_Partial

In [134]:

```
x_vif27 = x_vif26.drop('BsmtCond_None',axis = 1)
```

In [135]:

```
variable = x_vif27

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif27.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[135]:

	Variance Inflation Factor	Features
4	111.199203	GrLivArea
97	89.028671	HouseStyle_1Story
190	81.805367	BsmtFinType2_Unf
156	74.213763	MasVnrType_None
20	70.047424	BuildingAge
147	58.231056	Exterior2nd_MetalSd
246	57.301854	SaleType_New
160	55.124599	ExterQual_TA
253	54.211781	SaleCondition_Partial
100	53.293734	HouseStyle_2Story
11	52.511566	GarageCars

In [136]:

```
x_vif28 = x_vif27.drop('HouseStyle_1Story',axis = 1)
```

In [137]:

```
variable = x_vif28

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif28.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[137]:

	Variance Inflation Factor	Features
4	108.175834	GrLivArea
189	80.436103	BsmtFinType2_Unf
155	73.392229	MasVnrType_None
20	69.875931	BuildingAge
146	58.122296	Exterior2nd_MetalSd
245	57.286839	SaleType_New
159	54.922637	ExterQual_TA
252	54.200592	SaleCondition_Partial
11	52.497410	GarageCars
3	49.093784	TotalBsmtSF
213	48.729938	FireplaceQu_None

In [138]:

```
x_vif29 = x_vif28.drop('BsmtFinType2_Unf',axis = 1)
```

In [139]:

```
variable = x_vif29

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif29.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[139]:

	Variance Inflation Factor	Features
4	108.049586	GrLivArea
155	73.151834	MasVnrType_None
20	69.671413	BuildingAge
146	57.960372	Exterior2nd_MetalSd
244	57.275291	SaleType_New
251	54.121430	SaleCondition_Partial
159	53.836466	ExterQual_TA
11	52.455936	GarageCars
3	48.991420	TotalBsmtSF
212	48.614052	FireplaceQu_None
95	48.127842	BldgType_TwnhsE

In [140]:

```
x_vif30 = x_vif29.drop('MasVnrType_None',axis = 1)
```

In [141]:

```
variable = x_vif30

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif30.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[141]:

	Variance Inflation Factor	Features
4	107.975997	GrLivArea
20	68.733760	BuildingAge
146	57.832638	Exterior2nd_MetalSd
243	57.257153	SaleType_New
250	54.084836	SaleCondition_Partial
158	53.534772	ExterQual_TA
11	52.446845	GarageCars
3	48.913541	TotalBsmtSF
95	48.122703	BldgType_TwnhsE
211	48.067759	FireplaceQu_None
133	47.846346	Exterior1st_MetalSd

In [142]:

```
x_vif31 = x_vif30.drop('BuildingAge',axis = 1)
```

In [143]:

```
variable = x_vif31

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif31.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[143]:

	Variance Inflation Factor	Features
4	104.261285	GrLivArea
145	57.780938	Exterior2nd_MetalSd
242	57.254919	SaleType_New
249	54.083672	SaleCondition_Partial
157	53.016281	ExterQual_TA
11	52.392740	GarageCars
3	48.365725	TotalBsmtSF
94	48.081379	BldgType_TwnhsE
210	47.830252	FireplaceQu_None
132	47.788995	Exterior1st_MetalSd
39	46.196020	Alley_None

In [144]:

```
x_vif32 = x_vif31.drop('Exterior2nd_MetalSd',axis = 1)
```

In [145]:

```
variable = x_vif32

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif32.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[145]:

	Variance Inflation Factor	Features
4	104.199279	GrLivArea
241	57.241698	SaleType_New
248	54.083148	SaleCondition_Partial
156	52.906428	ExterQual_TA
11	52.368074	GarageCars
3	48.265058	TotalBsmtSF
94	48.079385	BldgType_TwnhsE
209	47.640006	FireplaceQu_None
12	46.094958	GarageArea
39	46.074654	Alley_None
171	45.489191	BsmtCond_TA

In [146]:

```
x_vif33 = x_vif32.drop('SaleType_New',axis = 1)
```

In [147]:

```
variable = x_vif33

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif33.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[147]:

	Variance Inflation Factor	Features
4	104.106802	GrLivArea
156	52.905477	ExterQual_TA
11	52.351700	GarageCars
3	48.263666	TotalBsmtSF
94	48.019622	BldgType_TwnhsE
209	47.639380	FireplaceQu_None
12	46.092329	GarageArea
39	46.069969	Alley_None
171	45.488201	BsmtCond_TA
92	44.463318	BldgType_2fmCon
35	44.072246	MSSubClass_190

In [148]:

```
x_vif34 = x_vif33.drop('ExterQual_TA',axis = 1)
```

In [149]:

```
variable = x_vif34

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif34.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[149]:

	Variance Inflation Factor	Features
4	103.795866	GrLivArea
11	52.215572	GarageCars
3	48.127635	TotalBsmtSF
94	48.019619	BldgType_TwnhsE
208	47.553320	FireplaceQu_None
12	46.086123	GarageArea
39	45.991606	Alley_None
170	45.296122	BsmtCond_TA
92	44.462870	BldgType_2fmCon
35	44.071868	MSSubClass_190
9	40.264147	BedroomAbvGr

In [150]:

```
x_vif35 = x_vif34.drop('GarageCars',axis = 1)
```

In [151]:

```
variable = x_vif35

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif35.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[151]:

	Variance Inflation Factor	Features
4	103.785717	GrLivArea
3	48.081259	TotalBsmtSF
93	48.011023	BldgType_TwnhsE
207	47.326597	FireplaceQu_None
38	45.964950	Alley_None
169	45.260415	BsmtCond_TA
91	44.458703	BldgType_2fmCon
34	44.066805	MSSubClass_190
9	39.939287	BedroomAbvGr
240	38.606420	SaleType_WD
192	38.093606	CentralAir_Y

In [152]:

```
x_vif36 = x_vif35.drop('BldgType_TwnhsE',axis = 1)
```

In [153]:

```
variable = x_vif36

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif36.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[153]:

	Variance Inflation Factor	Features
4	103.750626	GrLivArea
3	48.055078	TotalBsmtSF
206	47.256430	FireplaceQu_None
38	45.932198	Alley_None
168	45.252416	BsmtCond_TA
91	44.448222	BldgType_2fmCon
34	44.059552	MSSubClass_190
9	39.927372	BedroomAbvGr
239	38.567777	SaleType_WD
191	38.077408	CentralAir_Y
78	37.970210	Condition1_Norm

In [154]:

```
x_vif37 = x_vif36.drop('FireplaceQu_None',axis = 1)
```

In [155]:

```
variable = x_vif37

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif37.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[155]:

	Variance Inflation Factor	Features
4	103.658753	GrLivArea
3	47.922063	TotalBsmtSF
38	45.853987	Alley_None
168	45.069803	BsmtCond_TA
91	44.445715	BldgType_2fmCon
34	44.057314	MSSubClass_190
9	39.867510	BedroomAbvGr
238	38.322362	SaleType_WD
191	38.045383	CentralAir_Y
78	37.639158	Condition1_Norm
7	37.508410	FullBath

In [156]:

```
x_vif38 = x_vif37.drop('Alley_None',axis = 1)
```

In [157]:

```
variable = x_vif38

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif38.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[157]:

	Variance Inflation Factor	Features
4	103.595966	GrLivArea
3	47.828544	TotalBsmtSF
167	44.635664	BsmtCond_TA
90	44.444109	BldgType_2fmCon
34	44.055004	MSSubClass_190
9	39.699653	BedroomAbvGr
237	38.110205	SaleType_WD
190	37.828349	CentralAir_Y
7	37.507123	FullBath
77	37.201016	Condition1_Norm
95	35.634183	HouseStyle_2Story

In [158]:

```
x_vif39 = x_vif38.drop('BsmtCond_TA',axis = 1)
```

In [159]:

```
variable = x_vif39

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif39.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[159]:

	Variance Inflation Factor	Features
4	103.340514	GrLivArea
3	47.588476	TotalBsmtSF
90	44.359971	BldgType_2fmCon
34	43.996083	MSSubClass_190
9	39.647329	BedroomAbvGr
236	38.045754	SaleType_WD
7	37.467227	FullBath
189	37.392579	CentralAir_Y
77	36.944654	Condition1_Norm
95	35.627631	HouseStyle_2Story
44	32.624688	LandContour_Lvl

In [160]:

```
x_vif40 = x_vif39.drop('BldgType_2fmCon',axis = 1)
```

In [161]:

```
variable = x_vif40

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif40.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[161]:

	Variance Inflation Factor	Features
4	103.297791	GrLivArea
3	47.573527	TotalBsmtSF
9	39.646317	BedroomAbvGr
235	38.041977	SaleType_WD
7	37.464732	FullBath
77	36.917855	Condition1_Norm
188	36.865267	CentralAir_Y
94	35.619681	HouseStyle_2Story
44	32.384874	LandContour_Lvl
145	30.472114	Exterior2nd_VinylSd
219	30.024500	PavedDrive_Y

In [162]:

```
x_vif41 = x_vif40.drop('BedroomAbvGr',axis = 1)
```

In [163]:

```
variable = x_vif41

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif41.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[163]:

	Variance Inflation Factor	Features
4	94.158747	GrLivArea
3	47.289139	TotalBsmtSF
234	37.903539	SaleType_WD
76	36.889439	Condition1_Norm
187	36.701205	CentralAir_Y
7	36.606332	FullBath
93	35.547355	HouseStyle_2Story
43	32.377500	LandContour_Lvl
144	30.453151	Exterior2nd_VinylSd
218	30.023757	PavedDrive_Y
124	29.900481	Exterior1st_CemntBd

In [164]:

```
x_vif42 = x_vif41.drop('SaleType_WD',axis = 1)
```

In [165]:

```
variable = x_vif42

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif42.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[165]:

	Variance Inflation Factor	Features
4	93.629784	GrLivArea
3	46.978611	TotalBsmtSF
76	36.819369	Condition1_Norm
187	36.692417	CentralAir_Y
7	36.562480	FullBath
93	35.541548	HouseStyle_2Story
43	32.363402	LandContour_Lvl
218	30.016962	PavedDrive_Y
144	29.943390	Exterior2nd_VinylSd
124	29.893096	Exterior1st_CemntBd
137	29.826501	Exterior2nd_CmentBd

In [166]:

```
x_vif43 = x_vif42.drop('Condition1_Norm',axis = 1)
```

In [167]:

```
variable = x_vif43

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif43.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[167]:

	Variance Inflation Factor	Features
4	93.626917	GrLivArea
3	46.761731	TotalBsmtSF
186	36.536226	CentralAir_Y
7	36.477363	FullBath
92	35.541436	HouseStyle_2Story
43	32.229306	LandContour_Lvl
123	29.879039	Exterior1st_CemntBd
136	29.825266	Exterior2nd_CmentBd
143	29.815878	Exterior2nd_VinylSd
217	29.813559	PavedDrive_Y
223	29.035791	Fence_None

In [168]:

```
x_vif44 = x_vif43.drop('CentralAir_Y',axis = 1)
```

In [169]:

```
variable = x_vif44

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif44.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[169]:

	Variance Inflation Factor	Features
4	93.350516	GrLivArea
3	46.760296	TotalBsmtSF
7	36.458619	FullBath
92	35.417581	HouseStyle_2Story
43	32.076306	LandContour_Lvl
136	29.822697	Exterior2nd_CmentBd
123	29.807497	Exterior1st_CemntBd
216	29.747340	PavedDrive_Y
143	29.336836	Exterior2nd_VinylSd
222	29.033229	Fence_None
0	26.741982	LotFrontage

In [170]:

```
x_vif45 = x_vif44.drop('FullBath',axis = 1)
```

In [171]:

```
variable = x_vif45

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif45.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[171]:

	Variance Inflation Factor	Features
4	77.236475	GrLivArea
3	46.653859	TotalBsmtSF
91	35.397551	HouseStyle_2Story
42	31.570056	LandContour_Lvl
135	29.803544	Exterior2nd_CmentBd
122	29.800665	Exterior1st_CemntBd
215	29.743528	PavedDrive_Y
142	29.153645	Exterior2nd_VinylSd
221	29.011307	Fence_None
0	26.735879	LotFrontage
23	25.510564	MSSubClass_60

In [172]:

```
x_vif46 = x_vif45.drop('HouseStyle_2Story',axis = 1)
```

In [173]:

```
variable = x_vif46

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif46.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[173]:

	Variance Inflation Factor	Features
4	72.617770	GrLivArea
3	45.227446	TotalBsmtSF
42	31.569797	LandContour_Lvl
134	29.772241	Exterior2nd_CmentBd
121	29.754007	Exterior1st_CemntBd
214	29.516738	PavedDrive_Y
141	29.148979	Exterior2nd_VinylSd
220	29.010884	Fence_None
0	26.728984	LotFrontage
18	23.937844	GarageAge
9	23.675093	GarageArea

In [174]:

```
x_vif47 = x_vif46.drop('LandContour_Lvl',axis = 1)
```

In [175]:

```
variable = x_vif47

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif47.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[175]:

	Variance Inflation Factor	Features
4	72.616631	GrLivArea
3	45.138098	TotalBsmtSF
120	29.658091	Exterior1st_CemntBd
133	29.563605	Exterior2nd_CmentBd
213	29.135278	PavedDrive_Y
140	28.972230	Exterior2nd_VinylSd
219	28.951782	Fence_None
0	26.724305	LotFrontage
18	23.868516	GarageAge
9	23.648802	GarageArea
164	22.657673	BsmtExposure_None

In [176]:

```
x_vif48 = x_vif47.drop('Exterior1st_CemntBd',axis = 1)
```

In [177]:

```
variable = x_vif48

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif48.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[177]:

	Variance Inflation Factor	Features
4	72.600311	GrLivArea
3	45.136985	TotalBsmtSF
212	29.135027	PavedDrive_Y
218	28.906115	Fence_None
139	28.305786	Exterior2nd_VinylSd
0	26.687250	LotFrontage
18	23.851317	GarageAge
9	23.648051	GarageArea
163	22.652135	BsmtExposure_None
185	22.331626	Electrical_SBrkr
103	21.607797	OverallCond_5

In [178]:

```
x_vif49 = x_vif48.drop('PavedDrive_Y',axis = 1)
```

In [179]:

```
variable = x_vif49

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif49.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[179]:

	Variance Inflation Factor	Features
4	72.595379	GrLivArea
3	44.996145	TotalBsmtSF
217	28.641677	Fence_None
139	27.814656	Exterior2nd_VinylSd
0	26.659547	LotFrontage
18	23.780524	GarageAge
9	23.571367	GarageArea
163	22.651132	BsmtExposure_None
185	22.321957	Electrical_SBrkr
103	21.506769	OverallCond_5
188	20.968998	KitchenQual_TA

In [180]:

```
x_vif50 = x_vif49.drop('Fence_None',axis = 1)
```

In [181]:

```
variable = x_vif50

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif50.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[181]:

	Variance Inflation Factor	Features
4	72.443711	GrLivArea
3	44.424659	TotalBsmtSF
139	27.176119	Exterior2nd_VinylSd
0	26.618344	LotFrontage
18	23.778140	GarageAge
9	23.477237	GarageArea
163	22.623944	BsmtExposure_None
185	22.168779	Electrical_SBrkr
103	21.388980	OverallCond_5
172	20.929807	BsmtFinType2_None
188	20.784295	KitchenQual_TA

In [182]:

```
x_vif51 = x_vif50.drop('Exterior2nd_VinylSd',axis = 1)
```

In [183]:

```
variable = x_vif51

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif51.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[183]:

	Variance Inflation Factor	Features
4	72.428573	GrLivArea
3	44.350654	TotalBsmtSF
0	26.216321	LotFrontage
18	23.699060	GarageAge
9	23.327907	GarageArea
162	22.610739	BsmtExposure_None
184	21.996593	Electrical_SBrkr
103	20.958476	OverallCond_5
171	20.926569	BsmtFinType2_None
187	20.524034	KitchenQual_TA
156	19.119650	BsmtQual_TA

In [184]:

```
x_vif52 = x_vif51.drop('LotFrontage',axis = 1)
```

In [185]:

```
variable = x_vif52

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif52.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[185]:

	Variance Inflation Factor	Features
3	71.645655	GrLivArea
2	44.085049	TotalBsmtSF
17	23.109441	GarageAge
8	22.831289	GarageArea
161	22.609689	BsmtExposure_None
183	21.846754	Electrical_SBrkr
170	20.926235	BsmtFinType2_None
102	20.861059	OverallCond_5
186	20.480609	KitchenQual_TA
155	19.080077	BsmtQual_TA
59	18.300884	Neighborhood_NAmes

In [186]:

```
x_vif53 = x_vif52.drop('GarageAge',axis = 1)
```

In [187]:

```
variable = x_vif53

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif53.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[187]:

	Variance Inflation Factor	Features
3	70.884323	GrLivArea
2	44.049051	TotalBsmtSF
160	22.591171	BsmtExposure_None
182	21.799339	Electrical_SBrkr
169	20.906704	BsmtFinType2_None
101	20.635874	OverallCond_5
185	20.380185	KitchenQual_TA
8	20.324408	GarageArea
154	18.659341	BsmtQual_TA
58	17.518913	Neighborhood_NAmes
89	17.429835	HouseStyle_SLvl

In [188]:

```
x_vif54 = x_vif53.drop('BsmtExposure_None',axis = 1)
```

In [189]:

```
variable = x_vif54

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif54.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[189]:

	Variance Inflation Factor	Features
3	69.766700	GrLivArea
2	41.587407	TotalBsmtSF
181	21.786162	Electrical_SBrkr
101	20.619998	OverallCond_5
184	20.380095	KitchenQual_TA
8	20.318077	GarageArea
154	18.462967	BsmtQual_TA
89	17.408714	HouseStyle_SLvl
58	17.266734	Neighborhood_NAmes
24	16.440702	MSSubClass_80
225	16.434425	SaleCondition_Normal

In [190]:

```
x_vif55 = x_vif54.drop('Electrical_SBrkr',axis = 1)
```

In [191]:

```
variable = x_vif55

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif55.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[191]:

	Variance Inflation Factor	Features
3	69.766696	GrLivArea
2	41.423908	TotalBsmtSF
101	20.573294	OverallCond_5
183	20.280310	KitchenQual_TA
8	20.191978	GarageArea
154	18.224876	BsmtQual_TA
89	17.403103	HouseStyle_SLvl
58	16.755891	Neighborhood_NAmes
24	16.440546	MSSubClass_80
224	16.300020	SaleCondition_Normal
118	16.097442	Exterior1st_HdBoard

In [192]:

```
x_vif56 = x_vif55.drop('OverallCond_5',axis = 1)
```

In [193]:

```
variable = x_vif56

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif56.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[193]:

	Variance Inflation Factor	Features
3	69.435459	GrLivArea
2	41.215359	TotalBsmtSF
8	20.098627	GarageArea
182	19.968233	KitchenQual_TA
153	18.213387	BsmtQual_TA
89	17.376071	HouseStyle_SLvl
58	16.524710	Neighborhood_NAmes
24	16.427963	MSSubClass_80
223	16.278425	SaleCondition_Normal
117	16.096155	Exterior1st_HdBoard
130	14.923386	Exterior2nd_HdBoard

In [194]:

```
x_vif57 = x_vif56.drop('KitchenQual_TA',axis = 1)
```

In [195]:

```
variable = x_vif57

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif57.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[195]:

	Variance Inflation Factor	Features
3	69.423233	GrLivArea
2	41.138991	TotalBsmtSF
8	20.098435	GarageArea
153	17.898843	BsmtQual_TA
89	17.361587	HouseStyle_SLvl
24	16.419756	MSSubClass_80
117	16.096069	Exterior1st_HdBoard
222	15.995748	SaleCondition_Normal
58	15.802085	Neighborhood_NAmes
130	14.923384	Exterior2nd_HdBoard
123	13.012981	Exterior1st_Wd Sdng

In [196]:

```
x_vif58 = x_vif57.drop('BsmtQual_TA',axis = 1)
```

In [197]:

```
variable = x_vif58

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif58.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[197]:

	Variance Inflation Factor	Features
3	69.338242	GrLivArea
2	40.725110	TotalBsmtSF
8	20.086660	GarageArea
89	17.360573	HouseStyle_SLvl
24	16.406662	MSSubClass_80
117	16.080301	Exterior1st_HdBoard
221	15.898486	SaleCondition_Normal
130	14.917321	Exterior2nd_HdBoard
58	14.320531	Neighborhood_NAmes
123	13.011267	Exterior1st_Wd Sdng
147	12.967659	Foundation_PConc

In [198]:

```
x_vif59 = x_vif58.drop('HouseStyle_SLvl',axis = 1)
```

In [199]:

```
variable = x_vif59

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif59.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[199]:

	Variance Inflation Factor	Features
3	69.112223	GrLivArea
2	40.549278	TotalBsmtSF
8	20.076122	GarageArea
116	16.079873	Exterior1st_HdBoard
220	15.872304	SaleCondition_Normal
129	14.917192	Exterior2nd_HdBoard
58	14.242567	Neighborhood_NAmes
122	13.011005	Exterior1st_Wd Sdng
146	12.959908	Foundation_PConc
16	12.873200	RemodAge
135	12.452731	Exterior2nd_Wd Sdng

In [200]:

```
x_vif60 = x_vif59.drop('Exterior1st_HdBoard',axis = 1)
```

In [201]:

```
variable = x_vif60

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif60.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[201]:

	Variance Inflation Factor	Features
3	69.015176	GrLivArea
2	40.519669	TotalBsmtSF
8	20.061826	GarageArea
219	15.846607	SaleCondition_Normal
58	14.223316	Neighborhood_NAmes
145	12.944762	Foundation_PConc
16	12.859887	RemodAge
63	11.747516	Neighborhood_OldTown
7	11.183262	Fireplaces
144	10.942265	Foundation_CBlock
85	10.575149	HouseStyle_1.5Unf

In [202]:

```
x_vif61 = x_vif60.drop('SaleCondition_Normal',axis = 1)
```

In [203]:

```
variable = x_vif61

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif61.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[203]:

	Variance Inflation Factor	Features
3	68.791916	GrLivArea
2	40.518052	TotalBsmtSF
8	19.991083	GarageArea
58	13.731063	Neighborhood_NAmes
16	12.810782	RemodAge
145	12.774497	Foundation_PConc
63	11.564743	Neighborhood_OldTown
7	11.180150	Fireplaces
144	10.816746	Foundation_CBlock
85	10.566953	HouseStyle_1.5Unf
19	10.152267	MSSubClass_45

In [204]:

```
x_vif62 = x_vif61.drop('Neighborhood_NAmes',axis = 1)
```

In [205]:

```
variable = x_vif62

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif62.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[205]:

	Variance Inflation Factor	Features
3	68.208880	GrLivArea
2	39.627213	TotalBsmtSF
8	19.697370	GarageArea
16	11.990165	RemodAge
144	11.862176	Foundation_PConc
7	11.164800	Fireplaces
84	10.541848	HouseStyle_1.5Unf
19	10.142207	MSSubClass_45
143	9.497345	Foundation_CBlock
133	9.170861	Exterior2nd_Wd Sdng
120	9.064247	Exterior1st_Wd Sdng

In [206]:

```
x_vif63 = x_vif62.drop('RemodAge',axis = 1)
```

In [207]:

```
variable = x_vif63

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif63.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[207]:

	Variance Inflation Factor	Features
3	68.167430	GrLivArea
2	39.052241	TotalBsmtSF
8	19.662278	GarageArea
143	11.741257	Foundation_PConc
7	11.164018	Fireplaces
83	10.525145	HouseStyle_1.5Unf
18	10.142111	MSSubClass_45
142	9.217240	Foundation_CBlock
132	9.142227	Exterior2nd_Wd Sdng
119	9.063571	Exterior1st_Wd Sdng
153	7.703761	BsmtExposure_No

In [208]:

```
x_vif64 = x_vif63.drop('Foundation_PConc',axis = 1)
```

In [209]:

```
variable = x_vif64

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif64.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[209]:

	Variance Inflation Factor	Features
3	68.147089	GrLivArea
2	37.806049	TotalBsmtSF
8	19.412579	GarageArea
7	11.160259	Fireplaces
83	10.501760	HouseStyle_1.5Unf
18	10.135699	MSSubClass_45
132	9.109224	Exterior2nd_Wd Sdng
119	9.052062	Exterior1st_Wd Sdng
152	7.550549	BsmtExposure_No
65	6.873475	Neighborhood_Somerst
20	6.521841	MSSubClass_60

In [210]:

```
x_vif65 = x_vif64.drop('Fireplaces',axis = 1)
```

In [211]:

```
variable = x_vif65

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif65.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[211]:

	Variance Inflation Factor	Features
3	67.291255	GrLivArea
2	37.764759	TotalBsmtSF
7	19.296568	GarageArea
82	10.462705	HouseStyle_1.5Unf
17	10.106918	MSSubClass_45
131	9.108487	Exterior2nd_Wd Sdng
118	9.048988	Exterior1st_Wd Sdng
151	7.544417	BsmtExposure_No
64	6.864947	Neighborhood_Somerst
19	6.513072	MSSubClass_60
0	6.337739	LotArea

In [212]:

```
x_vif66 = x_vif65.drop('HouseStyle_1.5Unf',axis = 1)
```

In [213]:

```
variable = x_vif66

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif66.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[213]:

	Variance Inflation Factor	Features
3	67.143553	GrLivArea
2	37.649710	TotalBsmtSF
7	19.261611	GarageArea
130	9.107477	Exterior2nd_Wd Sdng
117	9.047648	Exterior1st_Wd Sdng
150	7.536427	BsmtExposure_No
64	6.864900	Neighborhood_Somerst
19	6.511333	MSSubClass_60
0	6.337140	LotArea
21	6.252610	MSSubClass_75
189	6.153803	GarageFinish_Unf

In [214]:

```
x_vif67 = x_vif66.drop('Exterior2nd_Wd Sdng',axis = 1)
```

In [215]:

```
variable = x_vif67

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif67.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[215]:

	Variance Inflation Factor	Features
3	67.076717	GrLivArea
2	37.622054	TotalBsmtSF
7	19.185235	GarageArea
149	7.528629	BsmtExposure_No
64	6.863811	Neighborhood_Somerst
19	6.502963	MSSubClass_60
0	6.323274	LotArea
21	6.252489	MSSubClass_75
188	6.130515	GarageFinish_Unf
13	5.820319	PoolArea
42	5.785388	LotConfig_Inside

In [216]:

```
x_vif68 = x_vif67.drop('BsmtExposure_No',axis = 1)
```

In [217]:

```
variable = x_vif68

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif68.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[217]:

	Variance Inflation Factor	Features
3	66.453917	GrLivArea
2	37.443923	TotalBsmtSF
7	19.127560	GarageArea
64	6.863787	Neighborhood_Somerst
19	6.502778	MSSubClass_60
0	6.318723	LotArea
21	6.252489	MSSubClass_75
187	6.090504	GarageFinish_Unf
13	5.814224	PoolArea
42	5.758405	LotConfig_Inside
29	5.573197	MSZoning_FV

In [218]:

```
x_vif69 = x_vif68.drop('Neighborhood_Somerst',axis = 1)
```

In [219]:

```
variable = x_vif69

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif69.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[219]:

	Variance Inflation Factor	Features
3	66.340565	GrLivArea
2	37.417577	TotalBsmtSF
7	18.771303	GarageArea
19	6.501671	MSSubClass_60
0	6.307970	LotArea
21	6.249814	MSSubClass_75
186	6.069084	GarageFinish_Unf
13	5.813107	PoolArea
42	5.747121	LotConfig_Inside
60	5.120501	Neighborhood_OldTown
138	5.094039	Foundation_CBlock

In [220]:

```
x_vif70 = x_vif69.drop('MSSubClass_60',axis = 1)
```

In [221]:

```
variable = x_vif70

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif70.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[221]:

	Variance Inflation Factor	Features
3	45.792998	GrLivArea
2	25.640095	TotalBsmtSF
7	18.770694	GarageArea
0	6.307961	LotArea
20	6.194875	MSSubClass_75
185	6.059001	GarageFinish_Unf
13	5.769013	PoolArea
41	5.736466	LotConfig_Inside
59	5.120419	Neighborhood_OldTown
137	5.088993	Foundation_CBlock
30	4.873940	MSZoning_RM

In [222]:

```
x_vif71 = x_vif70.drop('MSSubClass_75',axis = 1)
```

In [223]:

```
variable = x_vif71

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif71.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[223]:

	Variance Inflation Factor	Features
3	45.792911	GrLivArea
2	25.564757	TotalBsmtSF
7	18.768310	GarageArea
0	6.271409	LotArea
184	6.044261	GarageFinish_Unf
40	5.734086	LotConfig_Inside
13	5.175443	PoolArea
58	5.111264	Neighborhood_OldTown
136	5.088429	Foundation_CBlock
29	4.869837	MSZoning_RM
147	4.822698	BsmtFinType1_GLQ

In [224]:

```
x_vif72 = x_vif71.drop('GarageFinish_Unf',axis = 1)
```

In [225]:

```
variable = x_vif72

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif72.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[225]:

	Variance Inflation Factor	Features
3	45.656222	GrLivArea
2	25.360574	TotalBsmtSF
7	18.767959	GarageArea
0	6.271191	LotArea
40	5.705044	LotConfig_Inside
13	5.175437	PoolArea
58	5.080005	Neighborhood_OldTown
136	4.966924	Foundation_CBlock
29	4.867264	MSZoning_RM
147	4.812141	BsmtFinType1_GLQ
131	4.773235	ExterQual_Gd

In [226]:

```
x_vif73 = x_vif72.drop('LotConfig_Inside',axis = 1)
```

In [227]:

```
variable = x_vif73

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif73.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[227]:

	Variance Inflation Factor	Features
3	45.520889	GrLivArea
2	25.130821	TotalBsmtSF
7	18.754143	GarageArea
0	6.266782	LotArea
13	5.161359	PoolArea
57	5.079664	Neighborhood_OldTown
135	4.923257	Foundation_CBlock
29	4.865033	MSZoning_RM
130	4.772440	ExterQual_Gd
85	4.755115	OverallQual_7
146	4.743194	BsmtFinType1_GLQ

In [228]:

```
x_vif74 = x_vif73.drop('Neighborhood_OldTown',axis = 1)
```

In [229]:

```
variable = x_vif74

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif74.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[229]:

	Variance Inflation Factor	Features
3	45.387013	GrLivArea
2	25.128477	TotalBsmtSF
7	18.722877	GarageArea
0	6.220575	LotArea
13	5.139554	PoolArea
134	4.921888	Foundation_CBlock
129	4.771866	ExterQual_Gd
84	4.751786	OverallQual_7
139	4.729437	BsmtQual_Gd
145	4.729233	BsmtFinType1_GLQ
148	4.726287	BsmtFinType1_Unf

In [230]:

```
x_vif75 = x_vif74.drop('Foundation_CBlock',axis = 1)
```

In [231]:

```
variable = x_vif75

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif75.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[231]:

	Variance Inflation Factor	Features
3	45.365100	GrLivArea
2	24.883674	TotalBsmtSF
7	18.720293	GarageArea
0	6.155643	LotArea
13	5.136269	PoolArea
129	4.771022	ExterQual_Gd
84	4.733217	OverallQual_7
147	4.725389	BsmtFinType1_Unf
144	4.724848	BsmtFinType1_GLQ
138	4.652243	BsmtQual_Gd
79	4.530910	HouseStyle_SFoyer

In [232]:

```
x_vif76 = x_vif75.drop('PoolArea',axis = 1)
```

In [233]:

```
variable = x_vif76

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif76.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[233]:

	Variance Inflation Factor	Features
3	43.836393	GrLivArea
2	24.484436	TotalBsmtSF
7	18.681549	GarageArea
0	6.134109	LotArea
128	4.744427	ExterQual_Gd
83	4.733112	OverallQual_7
143	4.724613	BsmtFinType1_GLQ
146	4.724457	BsmtFinType1_Unf
137	4.649833	BsmtQual_Gd
78	4.528294	HouseStyle_SFoyer
96	4.377962	RoofStyle_Shed

In [234]:

```
x_vif77 = x_vif76.drop('GarageArea',axis = 1)
```

In [235]:

```
variable = x_vif77

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif77.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[235]:

	Variance Inflation Factor	Features
3	40.938601	GrLivArea
2	23.483408	TotalBsmtSF
0	6.121575	LotArea
127	4.741953	ExterQual_Gd
82	4.723680	OverallQual_7
142	4.693889	BsmtFinType1_GLQ
136	4.649627	BsmtQual_Gd
145	4.648257	BsmtFinType1_Unf
77	4.525861	HouseStyle_SFoyer
95	4.364535	RoofStyle_Shed
83	4.307206	OverallQual_8

In [236]:

```
x_vif78 = x_vif77.drop('LotArea',axis = 1)
```

In [237]:

```
variable = x_vif78

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif78.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[237]:

	Variance Inflation Factor	Features
2	40.302238	GrLivArea
1	23.384011	TotalBsmtSF
126	4.741174	ExterQual_Gd
81	4.721758	OverallQual_7
141	4.680170	BsmtFinType1_GLQ
144	4.645016	BsmtFinType1_Unf
135	4.634940	BsmtQual_Gd
76	4.523186	HouseStyle_SFoyer
82	4.306573	OverallQual_8
94	4.205910	RoofStyle_Shed
21	4.077261	MSSubClass_160

In [238]:

```
x_vif79 = x_vif78.drop('TotalBsmtSF',axis = 1)
```

In [239]:

```
variable = x_vif79

vif = pd.DataFrame()

vif['Variance Inflation Factor'] = [variance_inflation_factor(variable,i) for i in r

vif['Features'] = x_vif79.columns

vif.sort_values(by = 'Variance Inflation Factor',ascending=False)
```

Out[239]:

	Variance Inflation Factor	Features
1	29.125207	GrLivArea
125	4.725149	ExterQual_Gd
80	4.718809	OverallQual_7
140	4.667055	BsmtFinType1_GLQ
143	4.577710	BsmtFinType1_Unf
134	4.558945	BsmtQual_Gd
75	4.522914	HouseStyle_SFoyer
81	4.278504	OverallQual_8
93	4.205067	RoofStyle_Shed
159	4.006732	Electrical_Mix
20	4.003754	MSSubClass_160

In [240]:

```
X_selected = x_vif79
```

In [241]:

```
X_selected.shape
```

Out[241]:

```
(1460, 202)
```

In [242]:

```
1 X_selected.head()
```

Out[242]:

	MasVnrArea	GrLivArea	BsmtFullBath	BsmtHalfBath	HalfBath	WoodDeckSF	OpenPorch
0	196.0	1710	1	0	1	0	
1	0.0	1262	0	1	0	298	
2	162.0	1786	1	0	1	0	
3	0.0	1717	1	0	0	0	
4	350.0	2198	1	0	1	192	

In [243]:

```
1 # Running regression again.
```

In [244]:

```
1 x_train1, x_test1, y_train1, y_test1 = train_test_split(X_selected, y, test_size=0.3
```

In [245]:

```
1 x_train1.shape,x_test1.shape,y_train1.shape, y_test1.shape
```

Out[245]:

```
((1022, 202), (438, 202), (1022, 1), (438, 1))
```

In [246]:

```
#Linear Regression

linear1 = LinearRegression()
lr1 = linear1.fit(x_train1,y_train1)
y_pred_lr1_test = linear1.predict(x_test1)
y_pred_lr1_train = linear1.predict(x_train1)
r2_score_lr1_test = r2_score(y_test1, y_pred_lr1_test)
r2_score_lr1_train =  r2_score(y_train1, y_pred_lr1_train)
mape_lr1_test = mean_absolute_percentage_error(y_test1, y_pred_lr1_test)

# Ridge

ridge1 = Ridge(alpha = 0.3)
rid1 = ridge.fit(x_train1,y_train1)
y_pred_rid1_test = ridge.predict(x_test1)
y_pred_rid1_train = ridge.predict(x_train1)
r2_score_rid1_test = r2_score(y_test1, y_pred_rid1_test)
r2_score_rid1_train =  r2_score(y_train1, y_pred_rid1_train)
mape_rid1_test = mean_absolute_percentage_error(y_test1, y_pred_rid1_test)

#Lasso

lasso1 = Lasso(alpha = 0.1)
las1 = lasso1.fit(x_train1,y_train1)
y_pred_las1_test = lasso1.predict(x_test1)
y_pred_las1_train = lasso1.predict(x_train1)
r2_score_las1_test = r2_score(y_test1, y_pred_las1_test)
r2_score_las1_train =  r2_score(y_train1, y_pred_las1_train)
mape_las1_test = mean_absolute_percentage_error(y_test1, y_pred_las1_test)

# RandomForest
rfmodel1 = RandomForestRegressor()
rf1 = rfmodel1.fit(x_train1, y_train1)
y_pred_rf1_test = rfmodel1.predict(x_test1)
y_pred_rf1_train = rfmodel1.predict(x_train1)
r2_score_rf1_test = r2_score(y_test1, y_pred_rf1_test)
r2_score_rf1_train =  r2_score(y_train1, y_pred_rf1_train)
mape_rf1_test = mean_absolute_percentage_error(y_test1, y_pred_rf1_test)

#XGBRegressor
xgb1 = XGBRegressor()
xg1 = xgb1.fit(x_train1, y_train1)
y_pred_xg1_test = xgb1.predict(x_test1)
y_pred_xg1_train = xgb1.predict(x_train1)
r2_score_xg1_test = r2_score(y_test1, y_pred_xg1_test)
r2_score_xg1_train =  r2_score(y_train1, y_pred_xg1_train)
mape_xg1_test = mean_absolute_percentage_error(y_test1, y_pred_xg1_test)

```

In [262]:

```
print(r2_score_lr1_train)
print (r2_score_lr1_test)
print(mape_lr1_test)
```

```
0.898551701330835
0.8216228015161838
0.13974840936132626
```

In [263]:

```
print(r2_score_rid1_train)
print (r2_score_rid1_test)
print(mape_rid1_test)
```

```
0.896901563557416
0.8426100964546428
0.13422366156328258
```

In [297]:

```
print(r2_score_las1_train)
print (r2_score_las1_test)
print(mape_las1_test)
```

```
0.8985515060700859
0.8216428823787221
0.13973567443620152
```

In [265]:

```
print(r2_score_rf1_train)
print (r2_score_rf1_test)
print(mape_rf1_test)
```

```
0.973896969430925
0.8519816440845851
0.11434463735705999
```

In [267]:

```
print(r2_score_xg1_train)
print (r2_score_xg1_test)
print(mape_xg1_test)
```

```
0.9983077168758933
0.8648391485651581
0.11446179545993934
```

In [268]:

```
X_selected.shape
```

Out[268]:

```
(1460, 202)
```

Applying OLS and removing insignificant variables

In [247]:

```
# addition of constant
```

In [248]:

```
from statsmodels.regression.linear_model import OLS
import statsmodels.regression.linear_model as smf
import statsmodels.api as sm
```

In [249]:

```
x1 = sm.add_constant(X_selected)
```

In [256]:

```
reg_model = smf.OLS(endog = y, exog = x1).fit()
results = reg_model.summary()
```

In [251]:

```
d = {}
for i in x1.columns.tolist():
    d[f'{i}'] = reg_model.pvalues[i]

df_pvalue= pd.DataFrame(d.items(), columns=['Var_name', 'p-Value']).sort_values(by =
```

In [252]:

```
df_pvalue[df_pvalue['p-Value'] > 0.05]
```

Out[252]:

	Var_name	p-Value
0	EnclosedPorch	0.992271
1	Condition1_RRNn	0.976761
2	SaleType_ConLI	0.966675
3	Exterior1st_WdShing	0.962392
4	BsmtQual_Fa	0.933940
5	Fence_MnPrv	0.908697
6	ExterCond_Fa	0.906397
7	Exterior2nd_Brk Cmn	0.906295
8	Condition2_Feedr	0.897346
9	Exterior1st_AsphShn	0.890815
10	HouseStyle_SFoyer	0.888078

In [279]:

```
def backward_elimination(X,y, significance_level = 0.05):
    X = sm.add_constant(X)
    model = sm.OLS(y, X).fit()

    num_iterations = 0

    while True:
        max_pvalue = model.pvalues[1:].max()  # Skip the constant term
        if max_pvalue > significance_level:
            # Find the variable with the highest p-value and remove it
            remove_variable = model.pvalues[1:].idxmax()
            X = X.drop(remove_variable, axis=1)
            model = sm.OLS(y, X).fit()
            num_iterations += 1
        else:
            break

    selected_features = X.columns[1:]  # Exclude the constant term

    return selected_features, model.summary(),num_iterations


```

In [280]:

```
final_model = backward_elimination(x1,y)
final_model
```

Out[280]:

```
(Index(['MasVnrArea', 'GrLivArea', 'BsmtFullBath', 'WoodDeckSF', 'Scre
enPorch',
       'MSSubClass_30', 'MSSubClass_50', 'MSSubClass_70', 'MSSubClass
_90',
       'MSSubClass_120', 'MSSubClass_160', 'MSSubClass_180', 'MSSubCl
ass_190',
       'MSZoning_FV', 'MSZoning_RM', 'LotShape_IR2', 'Utilities_NoSeW
a',
       'LotConfig_CulDSac', 'Neighborhood_BrkSide', 'Neighborhood_Col
lgCr',
       'Neighborhood_Crawfor', 'Neighborhood_Edwards', 'Neighborhood_
NPkVill',
       'Neighborhood_NoRidge', 'Neighborhood_NridgHt', 'Neighborhood_
StoneBr',
       'Neighborhood_Timber', 'Neighborhood_Veenker', 'Condition1_Fee
dr',
       'Condition2_PosN', 'HouseStyle_2.5Fin', 'HouseStyle_2.5Unf',
       'OverallQual_6', 'OverallQual_7', 'OverallQual_8', 'OverallQua
```

In [281]:

```
x_ols = x1[['MasVnrArea', 'GrLivArea', 'BsmtFullBath', 'WoodDeckSF', 'ScreenPorch',
       'MSSubClass_30', 'MSSubClass_50', 'MSSubClass_70', 'MSSubClass_90',
       'MSSubClass_120', 'MSSubClass_160', 'MSSubClass_180', 'MSSubClass_190',
       'MSZoning_FV', 'MSZoning_RM', 'LotShape_IR2', 'Utilities_NoSeWa',
       'LotConfig_CulDSac', 'Neighborhood_BrkSide', 'Neighborhood_CollgCr',
       'Neighborhood_Crawfor', 'Neighborhood_Edwards', 'Neighborhood_NPkVill',
       'Neighborhood_NoRidge', 'Neighborhood_NridgHt', 'Neighborhood_StoneBr',
       'Neighborhood_Timber', 'Neighborhood_Veenker', 'Condition1_Feedr',
       'Condition2_PosN', 'HouseStyle_2.5Fin', 'HouseStyle_2.5Unf',
       'OverallQual_6', 'OverallQual_7', 'OverallQual_8', 'OverallQual_9',
       'OverallQual_10', 'OverallCond_3', 'OverallCond_4', 'OverallCond_7',
       'OverallCond_9', 'RoofMatl_WdShngl', 'Exterior1st_BrkFace',
       'Exterior2nd_CmentBd', 'Exterior2nd_ImStucc', 'Exterior2nd_Stucco',
       'BsmtExposure_Gd', 'BsmtFinType1_GLQ', 'BsmtFinType1_Unf',
       'BsmtFinType2_None', 'Heating_GasW', 'Heating_OthW', 'HeatingQC_Gd',
       'HeatingQC_TA', 'KitchenQual_Fa', 'Functional_Maj2', 'Functional_Min2',
       'Functional_Sev', 'GarageType_None', 'GarageQual_Fa', 'PoolQC_Gd',
       'SaleCondition_Alloca', 'SaleCondition_Partial']]
```

In [282]:

```
x_ols.shape
```

Out[282]:

```
(1460, 63)
```

In [283]:

```
x_train2, x_test2, y_train2, y_test2 = train_test_split(x_ols, y, test_size=0.30, ra
```

In [284]:

```
x_train2.shape,x_test2.shape,y_train2.shape, y_test2.shape
```

Out[284]:

```
((1022, 63), (438, 63), (1022, 1), (438, 1))
```

In [295]:

```
from math import sqrt
```

In [315]:

```
#Linear Regression

linear2 = LinearRegression()
lr2 = linear2.fit(x_train2,y_train2)
y_pred_lr2_test = linear2.predict(x_test2)
y_pred_lr2_train = linear2.predict(x_train2)
r2_score_lr2_test = r2_score(y_test1, y_pred_lr2_test)
r2_score_lr2_train =  r2_score(y_train1, y_pred_lr2_train)
mape_lr2_test = mean_absolute_percentage_error(y_test2, y_pred_lr2_test)
mape_lr2_train = mean_absolute_percentage_error(y_train2, y_pred_lr2_train)
mse_lr2_test = sqrt(mean_squared_error(y_test2, y_pred_lr2_test))
mse_lr2_train = sqrt(mean_squared_error(y_train2, y_pred_lr2_train))



# Ridge

ridge2 = Ridge(alpha = 0.3)
rid2 = ridge2.fit(x_train2,y_train2)
y_pred_rid2_test = ridge2.predict(x_test2)
y_pred_rid2_train = ridge2.predict(x_train2)
r2_score_rid2_test = r2_score(y_test2, y_pred_rid2_test)
r2_score_rid2_train =  r2_score(y_train2, y_pred_rid2_train)
mape_rid2_test = mean_absolute_percentage_error(y_test2, y_pred_rid2_test)
mape_rid2_train = mean_absolute_percentage_error(y_train2, y_pred_rid2_train)
mse_rid2_test = sqrt(mean_squared_error(y_test2, y_pred_rid2_test))
mse_rid2_train = sqrt(mean_squared_error(y_train2, y_pred_rid2_train))

#Lasso

lasso2 = Lasso(alpha = 0.1)
las2 = lasso2.fit(x_train2,y_train2)
y_pred_las2_test = lasso2.predict(x_test2)
y_pred_las2_train = lasso2.predict(x_train2)
r2_score_las2_test = r2_score(y_test2, y_pred_las2_test)
r2_score_las2_train =  r2_score(y_train2, y_pred_las2_train)
mape_las2_test = mean_absolute_percentage_error(y_test2, y_pred_las2_test)
mape_las2_train = mean_absolute_percentage_error(y_train2, y_pred_las2_train)
mse_las2_test = sqrt(mean_squared_error(y_test2, y_pred_las2_test))
mse_las2_train = sqrt(mean_squared_error(y_train2, y_pred_las2_train))

# RandomForest
rfmodel2 = RandomForestRegressor()
rf2 = rfmodel2.fit(x_train2, y_train2)
y_pred_rf2_test = rfmodel2.predict(x_test2)
y_pred_rf2_train = rfmodel2.predict(x_train2)
r2_score_rf2_test = r2_score(y_test2, y_pred_rf2_test)
r2_score_rf2_train =  r2_score(y_train2, y_pred_rf2_train)
mape_rf2_test = mean_absolute_percentage_error(y_test2, y_pred_rf2_test)
mape_rf2_train = mean_absolute_percentage_error(y_train2, y_pred_rf2_train)
mse_rf2_test = sqrt(mean_squared_error(y_test2, y_pred_rf2_test))
mse_rf2_train = sqrt(mean_squared_error(y_train2, y_pred_rf2_train))


#XGBRegressor
xgb2 = XGBRegressor()
xg2 = xgb2.fit(x_train2, y_train2)
y_pred_xg2_test = xgb2.predict(x_test2)
y_pred_xg2_train = xgb2.predict(x_train2)
```

```
r2_score_xg2_test = r2_score(y_test2, y_pred_xg2_test)
r2_score_xg2_train =  r2_score(y_train2, y_pred_xg2_train)
mape_xg2_test = mean_absolute_percentage_error(y_test2, y_pred_xg2_test)
mape_xg2_train = mean_absolute_percentage_error(y_train2, y_pred_xg2_train)
mse_xg2_test = sqrt(mean_squared_error(y_test2, y_pred_xg2_test))
mse_xg2_train = sqrt(mean_squared_error(y_train2, y_pred_xg2_train))

```

In [316]:

```
print('r2_score:')
print(r2_score_lr2_train)
print (r2_score_lr2_test)
print('MAPE:')
print(mape_lr2_train)
print(mape_lr2_test)
print('RMSE')
print(mse_lr2_train)
print(mse_lr2_test)
```

```
r2_score:
0.8837931739696465
0.8475263257015939
MAPE:
0.10335579131975903
0.12495764502177349
RMSE
26446.16328222568
32618.638771861697
```

In [317]:

```
print('r2_score:')
print(r2_score_rid2_train)
print (r2_score_rid2_test)
print('MAPE:')
print(mape_rid2_train)
print(mape_rid2_test)
print('RMSE')
print(mse_rid2_train)
print(mse_rid2_test)
```

```
r2_score:
0.8825917898761063
0.8614821425144459
MAPE:
0.10347350907370631
0.12171064915189714
RMSE
26582.516306457183
31090.040159167136
```

In [304]:

```
print('r2_score:')
print(r2_score_las2_train)
print (r2_score_las2_test)
print('MAPE:')
print(mape_las2_train)
print(mape_las2_test)
print('RMSE')
print(mse_las2_train)
print(mse_las2_test)
```

```
r2_score:
0.883793160151977
0.8475644867170216
MAPE:
0.10335245651276984
0.12494339381702226
RMSE
26446.16485452719
32614.556630215477
```

In [305]:

```
print('r2_score:')
print(r2_score_rf2_train)
print (r2_score_rf2_test)
print('MAPE:')
print(mape_rf2_train)
print(mape_rf2_test)
print('RMSE')
print(mse_rf2_train)
print(mse_rf2_test)
```

```
r2_score:
0.9693622278505661
0.8576078241489518
MAPE:
0.04642514290150511
0.11436005889237429
RMSE
13579.239442989587
31521.832971969958
```

In [306]:

```
print('r2_score:')
print(r2_score_xg2_train)
print (r2_score_xg2_test)
print('MAPE:')
print(mape_xg2_train)
print(mape_xg2_test)
print('RMSE')
print(mse_xg2_train)
print(mse_xg2_test)
```

```
r2_score:
0.9956675925739631
0.8628076453189819
MAPE:
0.02306446198168645
0.11829468886161817
RMSE
5106.360842257748
30940.929434136026
```

Feature Selection with Lasso

In [433]:

```
for alpha in [0.001,0.01,0.1,1,10,100,200,500,1000]:
    lasso_reg = Lasso(alpha = alpha)
    lasso_reg.fit(x_train2,y_train2)
    selected_features = np.array(x_train2.columns)[np.abs(lasso_reg.coef_) != 0]

    print('alpha:', alpha)
    print('Selected Features:',selected_features)
```

```
alpha: 0.001
Selected Features: ['MasVnrArea' 'GrLivArea' 'BsmtFullBath' 'WoodDeckS
F' 'ScreenPorch'
 'MSSubClass_30' 'MSSubClass_50' 'MSSubClass_70' 'MSSubClass_90'
 'MSSubClass_120' 'MSSubClass_160' 'MSSubClass_180' 'MSSubClass_190'
 'MSZoning_FV' 'MSZoning_RM' 'LotShape_IR2' 'Utilities_NoSeWa'
 'LotConfig_CulDSac' 'Neighborhood_BrkSide' 'Neighborhood_CollgCr'
 'Neighborhood_Crawfor' 'Neighborhood_Edwards' 'Neighborhood_NPkVill'
 'Neighborhood_NoRidge' 'Neighborhood_NridgHt' 'Neighborhood_StoneBr'
 'Neighborhood_Timber' 'Neighborhood_Veenker' 'Condition1_Feedr'
 'Condition2_PosN' 'HouseStyle_2.5Fin' 'HouseStyle_2.5Unf' 'OverallQua
l_6'
 'OverallQual_7' 'OverallQual_8' 'OverallQual_9' 'OverallQual_10'
 'OverallCond_3' 'OverallCond_4' 'OverallCond_7' 'OverallCond_9'
 'RoofMatl_WdShngl' 'Exterior1st_BrkFace' 'Exterior2nd_CmentBd'
 'Exterior2nd_ImStucc' 'Exterior2nd_Stucco' 'BsmtExposure_Gd'
 'BsmtFinType1_GLQ' 'BsmtFinType1_Unf' 'BsmtFinType2_None' 'Heating_Ga
sW'
 'Heating_OthW' 'HeatingQC_Gd' 'HeatingQC_TA' 'KitchenQual_Fa'
```

In [ ]:

```
# Taking variables where at alpha = 500
```

In [434]:

```
x_las = x_ols[['MasVnrArea', 'GrLivArea', 'BsmtFullBath', 'WoodDeckSF', 'ScreenPorch
 'MSSubClass_50', 'MSSubClass_90', 'MSSubClass_160' ,'MSSubClass_190',
 'MSZoning_RM' ,'LotShape_IR2' ,'LotConfig_CulDSac' ,'Neighborhood_Crawfor',
 'Neighborhood_Edwards' ,'Neighborhood_NoRidge' ,'Neighborhood_NridgHt',
 'Neighborhood_StoneBr' ,'Condition1_Feedr' ,'OverallQual_7', 'OverallQual_8',
 'OverallQual_9' ,'OverallQual_10' ,'OverallCond_3', 'OverallCond_4',
 'OverallCond_7', 'Exterior1st_BrkFace', 'BsmtExposure_Gd',
 'BsmtFinType1_GLQ', 'HeatingQC_Gd' ,'HeatingQC_TA', 'Functional_Min2',
 'GarageType_None', 'SaleCondition_Partial']]
```

In [435]:

```
x_train3, x_test3, y_train3, y_test3 = train_test_split(x_las, y, test_size=0.30, ra
```

In [436]:

```
x_train3.shape,x_test3.shape,y_train3.shape, y_test3.shape
```

Out[436]:

```
((1022, 33), (438, 33), (1022, 1), (438, 1))
```

In [437]:

```
#Linear Regression

linear3 = LinearRegression()
lr3 = linear3.fit(x_train3,y_train3)
y_pred_lr3_test = linear3.predict(x_test3)
y_pred_lr3_train = linear3.predict(x_train3)
r2_score_lr3_test = r2_score(y_test3, y_pred_lr3_test)
r2_score_lr3_train =  r2_score(y_train3, y_pred_lr3_train)
mape_lr3_test = mean_absolute_percentage_error(y_test3, y_pred_lr3_test)
mape_lr3_train = mean_absolute_percentage_error(y_train3, y_pred_lr3_train)
mse_lr3_test = sqrt(mean_squared_error(y_test3, y_pred_lr3_test))
mse_lr3_train = sqrt(mean_squared_error(y_train3, y_pred_lr3_train))



# Ridge

ridge3 = Ridge(alpha = 0.3)
rid3 = ridge3.fit(x_train3,y_train3)
y_pred_rid3_test = ridge3.predict(x_test3)
y_pred_rid3_train = ridge3.predict(x_train3)
r2_score_rid3_test = r2_score(y_test3, y_pred_rid3_test)
r2_score_rid3_train =  r2_score(y_train2, y_pred_rid3_train)
mape_rid3_test = mean_absolute_percentage_error(y_test3, y_pred_rid3_test)
mape_rid3_train = mean_absolute_percentage_error(y_train3, y_pred_rid3_train)
mse_rid3_test = sqrt(mean_squared_error(y_test3, y_pred_rid2_test))
mse_rid3_train = sqrt(mean_squared_error(y_train3, y_pred_rid3_train))

#Lasso

lasso3 = Lasso(alpha = 0.1)
las3 = lasso3.fit(x_train3,y_train3)
y_pred_las3_test = lasso3.predict(x_test3)
y_pred_las3_train = lasso3.predict(x_train3)
r2_score_las3_test = r2_score(y_test3, y_pred_las3_test)
r2_score_las3_train =  r2_score(y_train3, y_pred_las3_train)
mape_las3_test = mean_absolute_percentage_error(y_test3, y_pred_las3_test)
mape_las3_train = mean_absolute_percentage_error(y_train3, y_pred_las3_train)
mse_las3_test = sqrt(mean_squared_error(y_test3, y_pred_las3_test))
mse_las3_train = sqrt(mean_squared_error(y_train3, y_pred_las3_train))

# RandomForest
rfmodel3 = RandomForestRegressor()
rf3 = rfmodel3.fit(x_train3, y_train3)
y_pred_rf3_test = rfmodel3.predict(x_test3)
y_pred_rf3_train = rfmodel3.predict(x_train3)
r2_score_rf3_test = r2_score(y_test3, y_pred_rf3_test)
r2_score_rf3_train =  r2_score(y_train3, y_pred_rf3_train)
mape_rf3_test = mean_absolute_percentage_error(y_test3, y_pred_rf3_test)
mape_rf3_train = mean_absolute_percentage_error(y_train3, y_pred_rf3_train)
mse_rf3_test = sqrt(mean_squared_error(y_test3, y_pred_rf3_test))
mse_rf3_train = sqrt(mean_squared_error(y_train3, y_pred_rf3_train))


#XGBRegressor
xgb3 = XGBRegressor()
xg3 = xgb3.fit(x_train3, y_train3)
y_pred_xg3_test = xgb3.predict(x_test3)
y_pred_xg3_train = xgb3.predict(x_train3)
```

```
r2_score_xg3_test = r2_score(y_test3, y_pred_xg3_test)
r2_score_xg3_train =  r2_score(y_train3, y_pred_xg3_train)
mape_xg3_test = mean_absolute_percentage_error(y_test3, y_pred_xg3_test)
mape_xg3_train = mean_absolute_percentage_error(y_train3, y_pred_xg3_train)
mse_xg3_test = sqrt(mean_squared_error(y_test3, y_pred_xg3_test))
mse_xg3_train = sqrt(mean_squared_error(y_train3, y_pred_xg3_train))

```

In [438]:

```
print('r2_score:')
print(r2_score_lr3_train)
print (r2_score_lr3_test)
print('MAPE:')
print(mape_lr3_train)
print(mape_lr3_test)
print('RMSE')
print(mse_lr3_train)
print(mse_lr3_test)
```

```
r2_score:
0.8368350602901418
0.8752357594368299
MAPE:
0.11712644054250343
0.11908029493843823
RMSE
31337.20637454793
29506.212374044022
```

In [439]:

```
print('r2_score:')
print(r2_score_rid3_train)
print (r2_score_rid3_test)
print('MAPE:')
print(mape_rid3_train)
print(mape_rid3_test)
print('RMSE')
print(mse_rid3_train)
print(mse_rid3_test)
```

```
r2_score:
0.8367918738586384
0.8749915351347937
MAPE:
0.11695574451802665
0.11869448403789197
RMSE
31341.353259841864
31090.040159167136
```

In [440]:

```
print('r2_score:')
print(r2_score_las3_train)
print (r2_score_las3_test)
print('MAPE:')
print(mape_las3_train)
print(mape_las3_test)
print('RMSE')
print(mse_las3_train)
print(mse_las3_test)
```

```
r2_score:
0.8368350592418133
0.8752359799399606
MAPE:
0.11712485947844016
0.1190770624629746
RMSE
31337.206475218096
29506.18630000609
```

In [441]:

```
print('r2_score:')
print(r2_score_rf3_train)
print (r2_score_rf3_test)
print('MAPE:')
print(mape_rf3_train)
print(mape_rf3_test)
print('RMSE')
print(mse_rf3_train)
print(mse_rf3_test)
```

```
r2_score:
0.9713223715010775
0.8633582061875607
MAPE:
0.046273797536248974
0.11790202807479461
RMSE
13137.673742390394
30878.783150994437
```

In [442]:

```
print('r2_score:')
print(r2_score_xg3_train)
print (r2_score_xg3_test)
print('MAPE:')
print(mape_xg3_train)
print(mape_xg3_test)
print('RMSE')
print(mse_xg3_train)
print(mse_xg3_test)
```

```
r2_score:
0.9956772133521572
0.8727173544716427
MAPE:
0.023526698283995885
0.1189969299703845
RMSE
5100.687960630612
29802.520608788767
```

After the feature selection with VIF, OLS elimination and Lasso from the above model we can see that Linear Regression, Ridge and Lasso are giving a better result. Random Forest and Xgboost are building an overfitting model. Ridge is better than LR and Lasso.

In [456]:

```
y_test3 = np.array(y_test3)
y_pred_lr3_test = np.array(y_pred_rid3_test)
x_test3 = np.array(x_test3)
```

In [457]:

```
len(y_pred_rid3_test), len(y_test3)
```

Out[457]:

```
(438, 438)
```

In [458]:

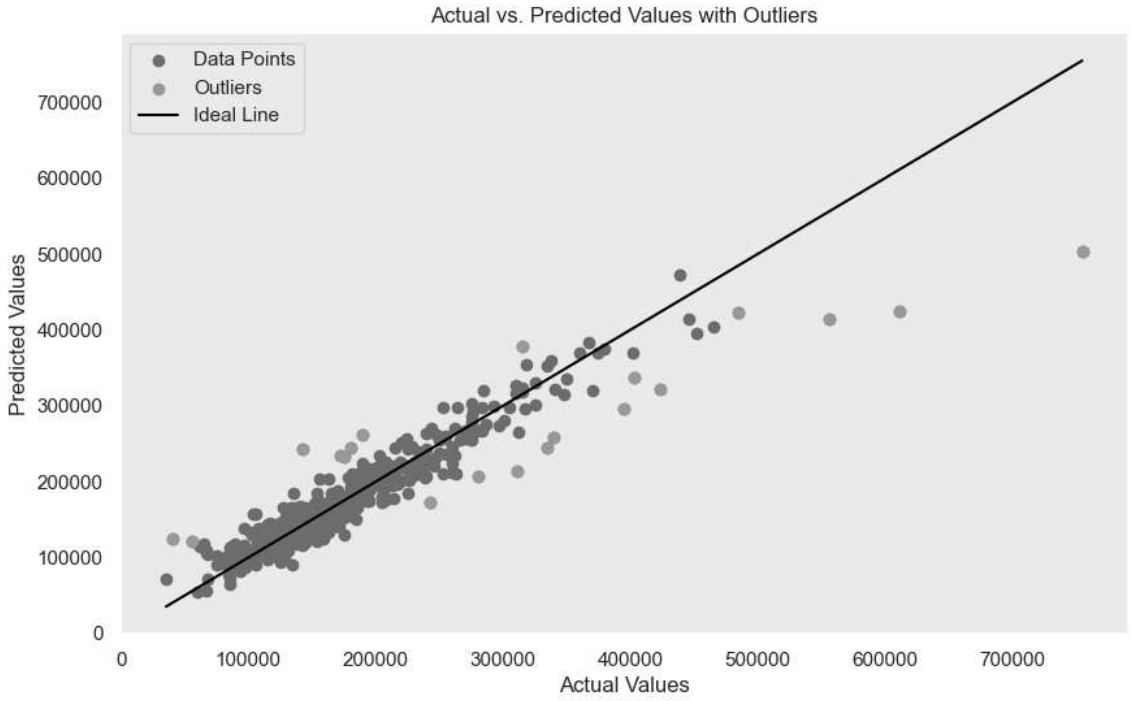
```
residuals = y_test3 - y_pred_rid3_test
mean_residual = np.mean(residuals)
std_residual = np.std(residuals)
threshold = 2 * std_residual
outliers = np.where(np.abs(mean_residual - residuals) > threshold)[0]
len(outliers)

```

Out[458]:

```
20
```

In [459]:

```
plt.figure(figsize=(10, 6))
plt.scatter(y_test3, y_pred_rid3_test, label='Data Points')
plt.scatter(y_test3[outliers], y_pred_rid3_test[outliers], label='Outliers')
plt.plot([min(y_test3), max(y_test3)], [min(y_test3), max(y_test3)], color='black',
plt.title('Actual vs. Predicted Values with Outliers')
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.legend()
plt.grid()
plt.show()
```



In [460]:

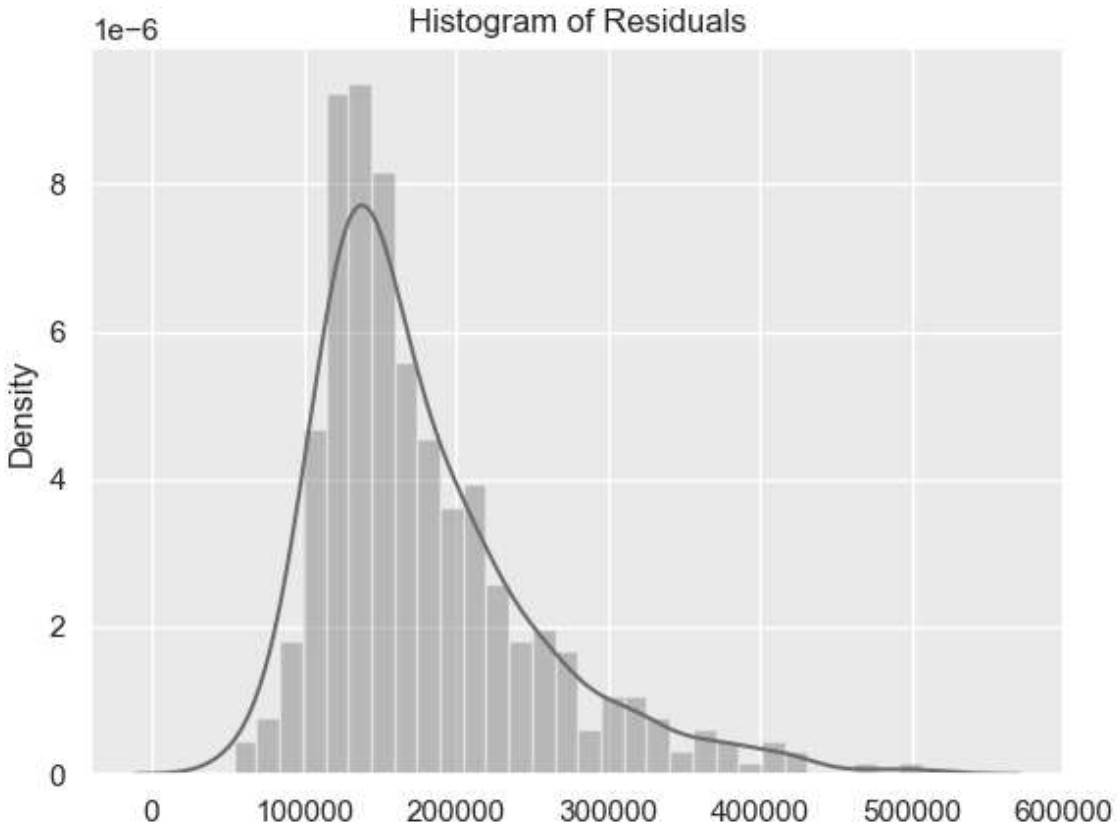
```
# We have 20 outliers out of 438 test datapoints
```

In [461]:

```
# Checking the normality of residuals
```

In [453]:

```
sns.distplot(y_pred_lr3_test,bins = 30)
plt.title('Histogram of Residuals')
plt.show()
```



In [462]:

```
y_test3_outliers = pd.DataFrame(y_test3[outliers], columns = ['Actual Price'])
y_pred_rid_test3_outliers = pd.DataFrame(y_pred_rid3_test[outliers], columns = ['Pre
```

In [463]:

```
x_test3_outliers = pd.DataFrame(x_test3[outliers], columns = x_las.columns)
```

In [464]:

```
res_df = pd.concat([x_test3_outliers,y_test3_outliers,y_pred_rid_test3_outliers],axi
```

In [465]:

```
res_df.sort_values(by = 'Predicted Price', ascending = False)
```

Out[465]:

	MasVnrArea	GrLivArea	BsmtFullBath	WoodDeckSF	ScreenPorch	MSSubClass_50	MSSubClass_90
3	1170.0	4316.0	0.0	382.0	0.0	0.0	0.0
2	760.0	2364.0	1.0	0.0	0.0	0.0	0.0
19	860.0	3140.0	0.0	144.0	0.0	0.0	0.0
5	208.0	2868.0	0.0	214.0	0.0	0.0	0.0
16	452.0	1828.0	0.0	0.0	260.0	0.0	0.0
4	506.0	2794.0	1.0	0.0	0.0	0.0	0.0
18	0.0	2097.0	1.0	192.0	0.0	0.0	0.0
11	375.0	1973.0	0.0	315.0	0.0	0.0	0.0
17	0.0	1923.0	1.0	240.0	0.0	0.0	0.0

Analysing the residual outliers we can see that some datapoints dont have MasVnrArea or WoodDeckSF or BsmtFullBath for the price prediction. Also these maybe outliers as we have kept only Significant variables for model Building.

In [ ]:

```
# Our Final Model has a test r2 score of 87%
# MAPE is 11% and RMSE is at 31000
# Normality of residuals is checked.
# There is no auto-correlation found as the Durbin Watson test is 1.89 which is clos
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```

```