Project: House Prices

Goal: Apply analytics to predict the sales price for each house. For each Id in the test set, must predict the value of the SalePrice variable.

Initial Setup

In [889…

```
# Required libraries
%pip install numpy
%pip install pandas
%pip install matplotlib.pyplot
%pip install statsmodels.api


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import seaborn as sns
```

```
Requirement already satisfied: numpy in c:\users\taban\anaconda3\lib\site-packages
(1.24.3)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: pandas in c:\users\taban\anaconda3\lib\site-package
s (2.0.3)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\taban\appdata\ro
aming\python\python311\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\taban\anaconda3\lib\site-p
ackages (from pandas) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in c:\users\taban\anaconda3\lib\site
-packages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.21.0 in c:\users\taban\anaconda3\lib\site-
packages (from pandas) (1.24.3)
Requirement already satisfied: six>=1.5 in c:\users\taban\appdata\roaming\python\p
ython311\site-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
Note: you may need to restart the kernel to use updated packages.
ERROR: Could not find a version that satisfies the requirement matplotlib.pyplot
(from versions: none)
ERROR: No matching distribution found for matplotlib.pyplot
Note: you may need to restart the kernel to use updated packages.
ERROR: Could not find a version that satisfies the requirement statsmodels.api (fr
om versions: none)
ERROR: No matching distribution found for statsmodels.api
```

In [890…

```
#Read data from file
data=pd.read_csv('train.csv')
data.head(5)
```

Out[890]:

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

5 rows × 81 columns

In [891…

```
data.shape
```

Out[891]:

```
(1460, 81)
```

Exploratory Data Analysis Check List

1: Understand data collection process

2: Document data set description (meta data)

3: Check for missing values

4: Univariate data analysis

5: Bivariate data analysis

6: Inferential Statistics (optional)

Data understanding- Phase1

1: Understand data collection process

Ref:Anna Montoya, DataCanary. (2016). House Prices - Advanced Regression Techniques. Kaggle. https://kaggle.com/competitions/house-prices-advanced-regression-techniques

2: Document Data Set Description (Meta Data)

In [892…

```
data.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     588 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
```

3: Check for Missing Values

In [893…

```
#Step 1: Determine the type of MVs
#Know the cause
np.sum(data.isnull())
```

Out[893]:

```
Id                 0
MSSubClass         0
MSZoning           0
LotFrontage      259
LotArea            0
                ... 
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64
```

In [894…

```
# Step 2: Determine the extent of MVs
# Summary of MVs in each column
mvs_summary = pd.DataFrame({'freq':np.sum(data.isnull())})
mvs_summary['pct'] = round(mvs_summary['freq']/data.shape[0]* 100, 1)
mvs_summary.sort_values(by = 'pct', ascending = False)
```

Out[894]:

	freq	pct
PoolQC	1453	99.5
MiscFeature	1406	96.3
Alley	1369	93.8
Fence	1179	80.8
MasVnrType	872	59.7
...	...	...
ExterQual	0	0.0
Exterior2nd	0	0.0
Exterior1st	0	0.0
RoofMatl	0	0.0
SalePrice	0	0.0

81 rows × 2 columns

In [895…

```
# Summary of MVs for each case
data.loc[:,'mvs']=data.apply(lambda row:np.sum(row.isnull()), axis = 1)
data.sort_values(by='mvs', ascending = False)
data.loc[:,'mvs_pct']=round( data.apply(lambda row:np.sum(row.isnull())/(data.shape
#(data.shape[1]-3), 3 refers to 3 extra columns (id, mvs, mvs-pct)
data.sort_values(by='mvs',ascending = False)
```

Out[895]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
1218	1219	50	RM	52.0	6240	Pave	NaN	Reg	Lvl
533	534	20	RL	50.0	5000	Pave	NaN	Reg	Low
39	40	90	RL	65.0	6040	Pave	NaN	Reg	Lvl
1011	1012	90	RL	75.0	9825	Pave	NaN	Reg	Lvl
1179	1180	20	RL	77.0	8335	Pave	NaN	Reg	Lvl
...	...	...	...	...	...	...	...	...	...
810	811	20	RL	78.0	10140	Pave	NaN	Reg	Lvl
1328	1329	50	RM	60.0	10440	Pave	Grvl	Reg	Lvl
766	767	60	RL	80.0	10421	Pave	NaN	Reg	Lvl
1083	1084	20	RL	80.0	8800	Pave	NaN	Reg	Lvl
1386	1387	60	RL	80.0	16692	Pave	NaN	IR1	Lvl

1460 rows × 83 columns

#Decision: Modify elements that are known as null but still contain information (Descriptive Statistics).

In [896…

```
data[['MasVnrArea','MasVnrType']]
```

Out[896]:

	MasVnrArea	MasVnrType
0	196.0	BrkFace
1	0.0	NaN
2	162.0	BrkFace
3	0.0	NaN
4	350.0	BrkFace
...	...	...
1455	0.0	NaN
1456	119.0	Stone
1457	0.0	NaN
1458	0.0	NaN
1459	0.0	NaN

1460 rows × 2 columns

In [ ]:

```

```

In [897…

```
# Modifying'MasVnrType'
data.loc[data['MasVnrArea']==0].shape[0]
```

Out[897]:

```
861
```

In [898…

```
data.loc[data['MasVnrArea'] == 0,'MasVnrType'] ='none'
np.sum(data['MasVnrType'].isnull())
```

Out[898]:

```
13
```

In [899…

```
np.sum(data['MasVnrArea'].isnull())
```

Out[899]:

```
8
```

In [900…

```
data.loc[data['MasVnrArea'].isnull()]
```

Out[900]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
234	235	60	RL	NaN	7851	Pave	NaN	Reg	Lvl
529	530	20	RL	NaN	32668	Pave	NaN	IR1	Lvl
650	651	60	FV	65.0	8125	Pave	NaN	Reg	Lvl
936	937	20	RL	67.0	10083	Pave	NaN	Reg	Lvl
973	974	20	FV	95.0	11639	Pave	NaN	Reg	Lvl
977	978	120	FV	35.0	4274	Pave	Pave	IR1	Lvl
1243	1244	20	RL	107.0	13891	Pave	NaN	Reg	Lvl
1278	1279	60	RL	75.0	9473	Pave	NaN	Reg	Lvl

8 rows × 83 columns

In [901…
```
data.loc[data['MasVnrType'].isnull()]
```

Out[901]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
234	235	60	RL	NaN	7851	Pave	NaN	Reg	Lvl
529	530	20	RL	NaN	32668	Pave	NaN	IR1	Lvl
624	625	60	RL	80.0	10400	Pave	NaN	Reg	Lvl
650	651	60	FV	65.0	8125	Pave	NaN	Reg	Lvl
773	774	20	RL	70.0	10150	Pave	NaN	Reg	Lvl
936	937	20	RL	67.0	10083	Pave	NaN	Reg	Lvl
973	974	20	FV	95.0	11639	Pave	NaN	Reg	Lvl
977	978	120	FV	35.0	4274	Pave	Pave	IR1	Lvl
1230	1231	90	RL	NaN	18890	Pave	NaN	IR1	Lvl
1243	1244	20	RL	107.0	13891	Pave	NaN	Reg	Lvl
1278	1279	60	RL	75.0	9473	Pave	NaN	Reg	Lvl
1300	1301	60	RL	NaN	10762	Pave	NaN	IR1	Lvl
1334	1335	160	RM	24.0	2368	Pave	NaN	Reg	Lvl

13 rows × 83 columns

In [902…
```
np.sum(data['MasVnrType']=='none')
```

Out[902]:
```
861
```

In [903…
```
data[['MasVnrArea','MasVnrType']]
```

Out[903]:

	MasVnrArea	MasVnrType
0	196.0	BrkFace
1	0.0	none
2	162.0	BrkFace
3	0.0	none
4	350.0	BrkFace
...	...	...
1455	0.0	none
1456	119.0	Stone
1457	0.0	none
1458	0.0	none
1459	0.0	none

1460 rows × 2 columns

In [904…
```python
# Modifying'FireplaceQu'
data.loc[data['Fireplaces']==0,'FireplaceQu']='NA'
```

In [905…
```python
np.sum(data['Fireplaces']==0)
```

Out[905]:
```
690
```

In [906…
```python
np.sum(data['FireplaceQu']=='NA')
```

Out[906]:
```
690
```

In [907…
```python
np.sum(data['FireplaceQu'].isnull())
```

Out[907]:
```
0
```

In [908…
```python
# Modifying'PoolQC'
data.loc[data['PoolArea']==0,'PoolQC']='NA'
```

In [909…
```python
np.sum(data['PoolArea']==0)
```

Out[909]:
```
1453
```

In [910…
```python
np.sum(data['PoolQC']=='NA')
```

Out[910]:
```
1453
```

In [911…
```python
np.sum(data['PoolQC'].isnull())
```

Out[911]:
```
0
```

In [912…
```python
# data[['PoolArea', 'PoolQC']]
```

In [913…
```python
np.sum(data['PoolArea']!=0)
```

Out[913]:
```
7
```

In [914…
```python
np.sum(data['PoolQC']!='NA')
```

Out[914]:
```
7
```

In [915…
```python
data.loc[(data['PoolArea']!= 0)]
```

Out[915]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
197	198	75	RL	174.0	25419	Pave	NaN	Reg	Lvl
810	811	20	RL	78.0	10140	Pave	NaN	Reg	Lvl
1170	1171	80	RL	76.0	9880	Pave	NaN	Reg	Lvl
1182	1183	60	RL	160.0	15623	Pave	NaN	IR1	Lvl
1298	1299	60	RL	313.0	63887	Pave	NaN	IR3	Bnk
1386	1387	60	RL	80.0	16692	Pave	NaN	IR1	Lvl
1423	1424	80	RL	NaN	19690	Pave	NaN	IR1	Lvl

7 rows × 83 columns

In [916…

```
data.loc[(data['PoolArea']!= 0)&(data['PoolQC']=='NA'),'PoolQC']=np.nan
```

In [917…

```
data.loc[data['PoolArea' ]!= 0]
```

Out[917]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
197	198	75	RL	174.0	25419	Pave	NaN	Reg	Lvl
810	811	20	RL	78.0	10140	Pave	NaN	Reg	Lvl
1170	1171	80	RL	76.0	9880	Pave	NaN	Reg	Lvl
1182	1183	60	RL	160.0	15623	Pave	NaN	IR1	Lvl
1298	1299	60	RL	313.0	63887	Pave	NaN	IR3	Bnk
1386	1387	60	RL	80.0	16692	Pave	NaN	IR1	Lvl
1423	1424	80	RL	NaN	19690	Pave	NaN	IR1	Lvl

7 rows × 83 columns

In [918…

```
data.loc[1298, 'PoolQC']=np.nan
```

In [919…

```
np.sum(data['PoolQC'].isnull())
```

Out[919]: 1

In [920…

```
# data['PoolQC']
```

In [921…

```
data.loc[data['PoolQC'].isnull()]
```

Out[921]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
1298	1299	60	RL	313.0	63887	Pave	NaN	IR3	Bnk

1 rows × 83 columns

In [922…

```
data.loc[1298,'PoolQC']
```

Out[922]: nan

In [923…
```
# Modifying'Fence'
np.sum(data['Fence'].isnull())
```

Out[923]: 1179

In [924…
```
data.loc[data['Fence'].isnull(),'Fence']='NA'
```

In [925…
```
np.sum(data['Fence'].isnull())
```

Out[925]: 0

In [926…
```
# data['Fence']
```

In [927…
```
# Modifying 'MiscFeature'
np.sum(data['MiscFeature'].isnull())
```

Out[927]: 1406

In [928…
```
data.loc[data['MiscFeature'].isnull(),'MiscFeature']='NA'
```

In [929…
```
np.sum(data['MiscFeature'].isnull())
```

Out[929]: 0

In [930…
```
# Modifying 'GarageType','GarageYrBlt','GarageFinish','GarageQual'and 'GarageCond'
np.sum(data['GarageArea']==0)
```

Out[930]: 81

In [931…
```
data.loc[data['GarageArea']==0,['GarageType','GarageFinish','GarageQual','GarageCor
data.loc[data['GarageArea']==0,['GarageYrBlt']]=0
data[['GarageArea','GarageType','GarageYrBlt','GarageFinish','GarageQual','GarageCo
```

Out[931]:

	GarageArea	GarageType	GarageYrBlt	GarageFinish	GarageQual	GarageCond
0	548	Attchd	2003.0	RFn	TA	TA
1	460	Attchd	1976.0	RFn	TA	TA
2	608	Attchd	2001.0	RFn	TA	TA
3	642	Detchd	1998.0	Unf	TA	TA
4	836	Attchd	2000.0	RFn	TA	TA
...	...	...	...	...	...	...
1455	460	Attchd	1999.0	RFn	TA	TA
1456	500	Attchd	1978.0	Unf	TA	TA
1457	252	Attchd	1941.0	RFn	TA	TA
1458	240	Attchd	1950.0	Unf	TA	TA
1459	276	Attchd	1965.0	Fin	TA	TA

1460 rows × 6 columns

In [932…

```
np.sum(data['GarageType'].isnull())
```

Out[932]: 0

In [933…

```
np.sum(data['GarageYrBlt'].isnull())
```

Out[933]: 0

In [934…

```
np.sum(data['GarageFinish'].isnull())
```

Out[934]: 0

In [935…

```
np.sum(data['GarageQual'].isnull())
```

Out[935]: 0

In [936…

```
np.sum(data['GarageCond'].isnull())
```

Out[936]: 0

In [937…

```
# Modifying'BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2'
np.sum(data['TotalBsmtSF']==0)
```

Out[937]: 37

In [938…

```
data.loc[data['TotalBsmtSF']==0,['BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1
data[['TotalBsmtSF','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinTyp
```

Out[938]:

	TotalBsmtSF	BsmtQual	BsmtCond	BsmtExposure	BsmtFinType1	BsmtFinType2
0	856	Gd	TA	No	GLQ	Unf
1	1262	Gd	TA	Gd	ALQ	Unf
2	920	Gd	TA	Mn	GLQ	Unf
3	756	TA	Gd	No	ALQ	Unf
4	1145	Gd	TA	Av	GLQ	Unf
...	...	...	...	...	...	...
1455	953	Gd	TA	No	Unf	Unf
1456	1542	Gd	TA	No	ALQ	Rec
1457	1152	TA	Gd	No	GLQ	Unf
1458	1078	TA	TA	Mn	GLQ	Rec
1459	1256	TA	TA	No	BLQ	LwQ

1460 rows × 6 columns

In [939…

```
np.sum(data['BsmtQual'].isnull())
```

Out[939]: 0

In [940…

```
np.sum(data['BsmtCond'].isnull())
```

Out[940]: 0

In [941…
```
np.sum(data['BsmtExposure'].isnull())
```
Out[941]:
```
1
```
In [942…
```
data.loc[data['BsmtExposure'].isnull()]
```
Out[942]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	U
948	949	60	RL	65.0	14006	Pave	NaN	IR1	Lvl	

1 rows × 83 columns

In [943…
```
np.sum(data['BsmtFinType1'].isnull())
```
Out[943]:
```
0
```
In [944…
```
np.sum(data['BsmtFinType2'].isnull())
```
Out[944]:
```
1
```
In [945…
```
data.loc[data['BsmtFinType2'].isnull()]
```
Out[945]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	U
332	333	20	RL	85.0	10655	Pave	NaN	IR1	Lvl	

1 rows × 83 columns

In [946…
```
# Modifying'LotFrontage'
np.sum(data['LotFrontage'].isnull())
```
Out[946]:
```
259
```
In [947…
```
# data['LotFrontage']
```
In [948…
```
np.sum(data['LotFrontage']==0)
```
Out[948]:
```
0
```
In [949…
```
data.loc[data['LotFrontage'].isnull(),'LotFrontage']=0
# data['LotFrontage']
```
In [950…
```
np.sum(data['LotFrontage'].isnull())
```
Out[950]:
```
0
```
In [951…
```
np.sum(data['LotFrontage']==0)
```
Out[951]:
```
259
```
In [952…
```
# Modifying 'Alley'
np.sum(data['Alley'].isnull())
```
Out[952]:
```
1369
```

In [953…

```
data.loc[data['Alley'].isnull(),'Alley']='NA'
# data['Alley']
```

In [954…

```
np.sum(data['Alley'].isnull())
```

Out[954]:

```
0
```

In [955…

```
np.sum(data['Alley']=='NA')
```

Out[955]:

```
1369
```

In [956…

```
#Step 2: Determine the extent of MVs after Modifying
#Summary of MVs in each column after Modifying
mvs_summary=pd.DataFrame({'freq':np.sum(data.isnull())})
mvs_summary['pct']=round(mvs_summary['freq']/data.shape[0]*100,1)
mvs_summary.sort_values(by='pct',ascending = False)
```

Out[956]:

	freq	pct
MasVnrType	13	0.9
MasVnrArea	8	0.5
PoolQC	1	0.1
Electrical	1	0.1
BsmtFinType2	1	0.1
...	...	...
ExterQual	0	0.0
Exterior2nd	0	0.0
Exterior1st	0	0.0
RoofMatl	0	0.0
mvs_pct	0	0.0

83 rows × 2 columns

In [957…

```
# Summary of MVs for each case after Modifying
data.loc[:,'mvs']=data.apply(lambda row: np.sum(row.isnull()),axis = 1)
data.sort_values(by='mvs',ascending = False)
data.loc[:,'mvs_pct'] =round(data.apply(lambda row: np.sum(row.isnull())/(data.shap
#(data.shape[1]-3), 3 refers to 3 extra columns (id, mvs, mvs-pct)
data.sort_values(by='mvs',ascending = False)
```

Out[957]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
529	530	20	RL	0.0	32668	Pave	NA	IR1	Lvl
936	937	20	RL	67.0	10083	Pave	NA	Reg	Lvl
1243	1244	20	RL	107.0	13891	Pave	NA	Reg	Lvl
234	235	60	RL	0.0	7851	Pave	NA	Reg	Lvl
977	978	120	FV	35.0	4274	Pave	Pave	IR1	Lvl
...	...	...	...	...	...	...	...	...	...
485	486	20	RL	80.0	9600	Pave	NA	Reg	Lvl
484	485	20	RL	0.0	7758	Pave	NA	IR1	Lvl
483	484	120	RM	32.0	4500	Pave	NA	Reg	Lvl
482	483	70	RM	50.0	2500	Pave	Pave	Reg	Lvl
1459	1460	20	RL	75.0	9937	Pave	NA	Reg	Lvl

1460 rows × 83 columns

In [958…

```
data['mvs_pct'].max()
```

Out[958]:

```
2.5
```

In [959…

```
# Step 3: Diagnose the randomness of the missing values processes
# Given the low percentage of missing values in columns and records (in a large dat
# It is acceptable to proceed with a simple imputation strategy without a detailed
# This imputation will not significantly impact the overall results.
```

In [960…

```
np.sum(data['MasVnrArea'].isnull())
```

Out[960]:

```
8
```

In [961…

```
float_columns=data.select_dtypes(include=['float64'])
float_columns
```

Out[961]:

	LotFrontage	MasVnrArea	GarageYrBlt	mvs_pct
0	65.0	196.0	2003.0	0.0
1	80.0	0.0	1976.0	0.0
2	68.0	162.0	2001.0	0.0
3	60.0	0.0	1998.0	0.0
4	84.0	350.0	2000.0	0.0
...	...	...	...	...
1455	62.0	0.0	1999.0	0.0
1456	85.0	119.0	1978.0	0.0
1457	66.0	0.0	1941.0	0.0
1458	68.0	0.0	1950.0	0.0
1459	75.0	0.0	1965.0	0.0

1460 rows × 4 columns

In [962…

```
# Step 4: Select the imputation method
# Method: mean substitution
# Substiude missing values in numeric column ('MasVnrArea') with the mean of columr
data.loc[:,'MasVnrArea']=data.loc[:,'MasVnrArea'].fillna(data.loc[:,'MasVnrArea'].n
print(data.shape)
np.sum(data['MasVnrArea'].isnull())
```

```
(1460, 83)
```

Out[962]:

```
0
```

In [963…

```
# Method: mode substitution
# Substiude missing values in categorical columns with the mode of each column.
categorical_columns = ['MasVnrType','BsmtExposure','BsmtFinType2','Electrical','Poc

for i in categorical_columns:
    mode_value=data[i].mode()[0]
    data[i]=data[i].fillna(mode_value)

print(data.shape)
np.sum(data[['MasVnrType','BsmtExposure','BsmtFinType2','Electrical','PoolQC']].isr
```

```
(1460, 83)
```

Out[963]:

```
MasVnrType      0
BsmtExposure    0
BsmtFinType2    0
Electrical      0
PoolQC          0
dtype: int64
```

In [964…

```
# np.sum(data.isnull())
```

In [965…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 83 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    1460 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          1460 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     1460 non-null   object
 26  MasVnrArea     1460 non-null   float64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1460 non-null   object
 31  BsmtCond       1460 non-null   object
 32  BsmtExposure   1460 non-null   object
 33  BsmtFinType1   1460 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1460 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     1460 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    1460 non-null   object
 58  GarageType     1460 non-null   object
```

```
 59  GarageYrBlt    1460 non-null   float64
 60  GarageFinish   1460 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1460 non-null   object
 64  GarageCond     1460 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         1460 non-null   object
 73  Fence          1460 non-null   object
 74  MiscFeature    1460 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
 81  mvs            1460 non-null   int64
 82  mvs_pct        1460 non-null   float64
dtypes: float64(4), int64(36), object(43)
memory usage: 946.8+ KB
```

In [966…

```
# Convert the column from float to int
data['LotFrontage']=data['LotFrontage'].astype('int64')
data['GarageYrBlt']=data['GarageYrBlt'].astype('int64')
data['MasVnrArea']=data['MasVnrArea'].astype('int64')

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 83 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    1460 non-null   int64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          1460 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     1460 non-null   object
 26  MasVnrArea     1460 non-null   int64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1460 non-null   object
 31  BsmtCond       1460 non-null   object
 32  BsmtExposure   1460 non-null   object
 33  BsmtFinType1   1460 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1460 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     1460 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    1460 non-null   object
 58  GarageType     1460 non-null   object
```

```
 59  GarageYrBlt    1460 non-null   int64
 60  GarageFinish   1460 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1460 non-null   object
 64  GarageCond     1460 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         1460 non-null   object
 73  Fence          1460 non-null   object
 74  MiscFeature    1460 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
 81  mvs            1460 non-null   int64
 82  mvs_pct        1460 non-null   float64
dtypes: float64(1), int64(39), object(43)
memory usage: 946.8+ KB
```

In [967…

```
data.loc[data['YearRemodAdd'] < data['YearBuilt'],['YearRemodAdd','YearBuilt']]
```

Out[967]:

YearRemodAdd	YearBuilt

In [968…

```
data.loc[data['MasVnrArea'] < 0,'MasVnrArea']
```

Out[968]:

```
Series([], Name: MasVnrArea, dtype: int64)
```

4: Univariate data analysis

In [969…

```
data['Id'].nunique()
```

Out[969]:

```
1460
```

In [970…

```
np.sum(data.duplicated())
```

Out[970]:

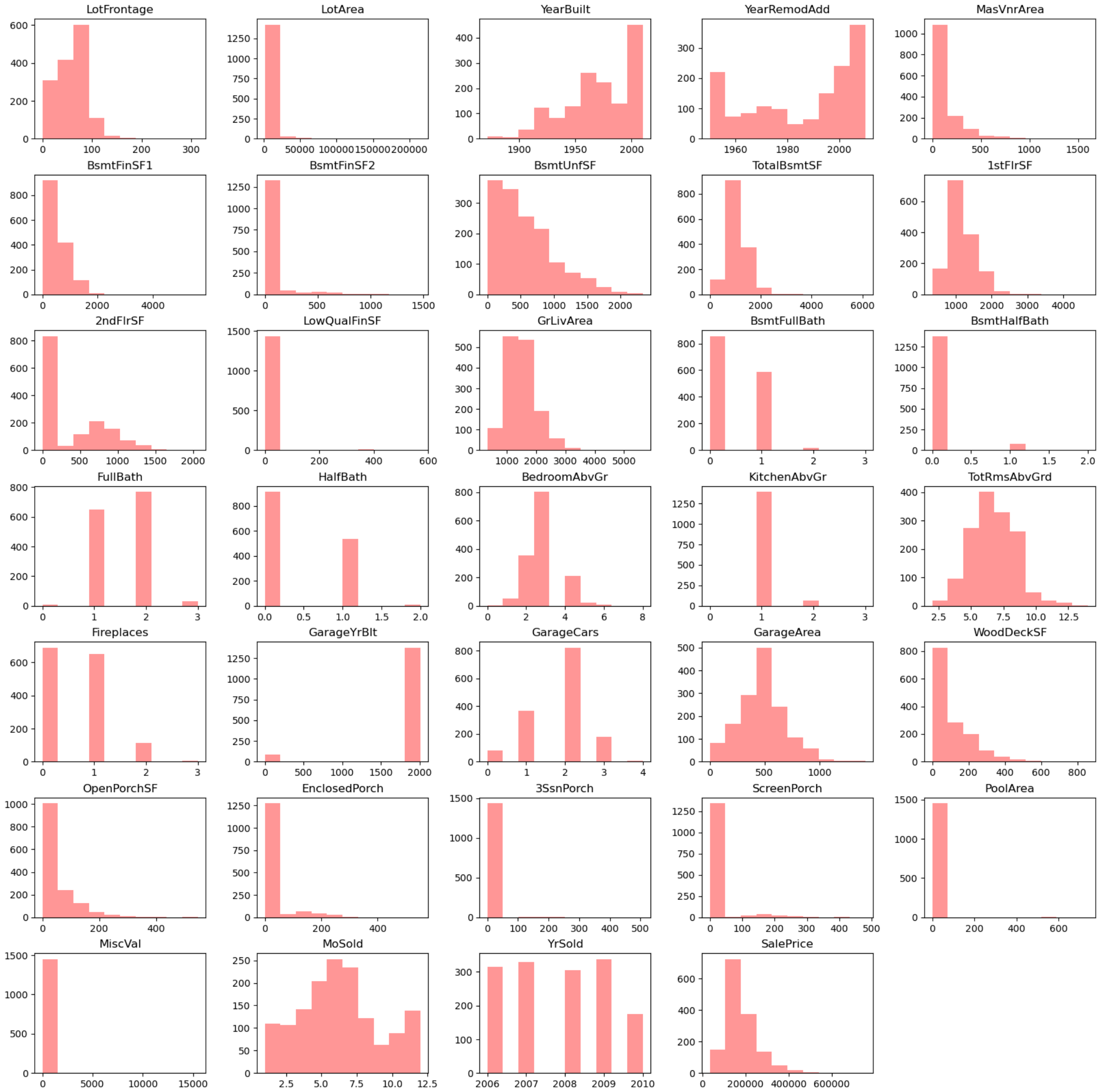
```
0
```

In [971…

```
var_ind_num1=[3,4,19,20,26,34,36,37,38]+ list(range(43,53))+[54,56,59,61,62]+list(r
var_ind_cat1=[1,2]+ list(range(5,19))+list(range(21,26))+list(range(27,34))+[35]+li
```

In [972…

```
# Histogram of numeric variables
plt.figure(figsize=(20,20))
plt.subplots_adjust(hspace =0.3, wspace =0.3)
for i in range(1,35):
    plt.subplot(7,5,i)
    plt.hist(x = data.iloc[: ,var_ind_num1[i- 1]],
             alpha = 0.4, color = 'red')
    plt.title(data.columns[var_ind_num1[i - 1]])
plt.show()
```



In [973…

```
#summary statistics of numeric variables
data.iloc[:,var_ind_num1].describe()
```

Out[973]:

	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtF
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.0
mean	57.623288	10516.828082	1971.267808	1984.865753	103.681507	443.639726	46.5
std	34.664304	9981.264932	30.202904	20.645407	180.569120	456.098091	161.3
min	0.000000	1300.000000	1872.000000	1950.000000	0.000000	0.000000	0.0
25%	42.000000	7553.500000	1954.000000	1967.000000	0.000000	0.000000	0.0
50%	63.000000	9478.500000	1973.000000	1994.000000	0.000000	383.500000	0.0
75%	79.000000	11601.500000	2000.000000	2004.000000	164.250000	712.250000	0.0
max	313.000000	215245.000000	2010.000000	2010.000000	1600.000000	5644.000000	1474.0

8 rows × 34 columns

In [974…

```
# Calculate frequency and percentage
for i in var_ind_cat1:
    print('--- Frequency Table of '+data.columns[i] +' ---')
```

```
    frq_summary=pd.DataFrame({'category':(data.iloc[:, i].value_counts()).index,'fr
    frq_summary['pct']=round(frq_summary['freq']/data.shape[0]* 100, 1)
    frq_summary.sort_values(by ='pct',ascending=False)
    print(frq_summary)
```

```
--- Frequency Table of MSSubClass ---
    category  freq   pct
0         20   536  36.7
1         60   299  20.5
2         50   144   9.9
3        120    87   6.0
4         30    69   4.7
5        160    63   4.3
6         70    60   4.1
7         80    58   4.0
8         90    52   3.6
9        190    30   2.1
10        85    20   1.4
11        75    16   1.1
12        45    12   0.8
13       180    10   0.7
14        40     4   0.3
--- Frequency Table of MSZoning ---
  category  freq   pct
0       RL  1151  78.8
1       RM   218  14.9
2       FV    65   4.5
3       RH    16   1.1
4  C (all)    10   0.7
--- Frequency Table of Street ---
  category  freq   pct
0     Pave  1454  99.6
1     Grvl     6   0.4
--- Frequency Table of Alley ---
  category  freq   pct
0       NA  1369  93.8
1     Grvl    50   3.4
2     Pave    41   2.8
--- Frequency Table of LotShape ---
  category  freq   pct
0      Reg   925  63.4
1      IR1   484  33.2
2      IR2    41   2.8
3      IR3    10   0.7
--- Frequency Table of LandContour ---
  category  freq   pct
0      Lvl  1311  89.8
1      Bnk    63   4.3
2      HLS    50   3.4
3      Low    36   2.5
--- Frequency Table of Utilities ---
  category  freq   pct
0   AllPub  1459  99.9
1   NoSeWa     1   0.1
--- Frequency Table of LotConfig ---
  category  freq   pct
0   Inside  1052  72.1
1   Corner   263  18.0
2  CulDSac    94   6.4
3      FR2    47   3.2
4      FR3     4   0.3
--- Frequency Table of LandSlope ---
  category  freq   pct
0      Gtl  1382  94.7
1      Mod    65   4.5
2      Sev    13   0.9
--- Frequency Table of Neighborhood ---
   category  freq   pct
0     NAmes   225  15.4
```

```
1   CollgCr   150  10.3
2   OldTown   113   7.7
3   Edwards   100   6.8
4   Somerst    86   5.9
5   Gilbert    79   5.4
6   NridgHt    77   5.3
7    Sawyer    74   5.1
8    NWAmes    73   5.0
9   SawyerW    59   4.0
10  BrkSide    58   4.0
11  Crawfor    51   3.5
12  Mitchel    49   3.4
13  NoRidge    41   2.8
14   Timber    38   2.6
15   IDOTRR    37   2.5
16  ClearCr    28   1.9
17  StoneBr    25   1.7
18    SWISU    25   1.7
19  MeadowV    17   1.2
20  Blmngtn    17   1.2
21   BrDale    16   1.1
22  Veenker    11   0.8
23  NPkVill     9   0.6
24  Blueste     2   0.1
--- Frequency Table of Condition1 ---
  category  freq   pct
0     Norm  1260  86.3
1    Feedr    81   5.5
2   Artery    48   3.3
3     RRAn    26   1.8
4     PosN    19   1.3
5     RRAe    11   0.8
6     PosA     8   0.5
7     RRNn     5   0.3
8     RRNe     2   0.1
--- Frequency Table of Condition2 ---
  category  freq   pct
0     Norm  1445  99.0
1    Feedr     6   0.4
2   Artery     2   0.1
3     RRNn     2   0.1
4     PosN     2   0.1
5     PosA     1   0.1
6     RRAn     1   0.1
7     RRAe     1   0.1
--- Frequency Table of BldgType ---
  category  freq   pct
0     1Fam  1220  83.6
1   TwnhsE   114   7.8
2   Duplex    52   3.6
3    Twnhs    43   2.9
4   2fmCon    31   2.1
--- Frequency Table of HouseStyle ---
  category  freq   pct
0   1Story   726  49.7
1   2Story   445  30.5
2   1.5Fin   154  10.5
3     SLvl    65   4.5
4   SFoyer    37   2.5
5   1.5Unf    14   1.0
6   2.5Unf    11   0.8
7   2.5Fin     8   0.5
--- Frequency Table of OverallQual ---
   category  freq   pct
```

```
0         5   397  27.2
1         6   374  25.6
2         7   319  21.8
3         8   168  11.5
4         4   116   7.9
5         9    43   2.9
6         3    20   1.4
7        10    18   1.2
8         2     3   0.2
9         1     2   0.1
--- Frequency Table of OverallCond ---
   category  freq   pct
0         5   821  56.2
1         6   252  17.3
2         7   205  14.0
3         8    72   4.9
4         4    57   3.9
5         3    25   1.7
6         9    22   1.5
7         2     5   0.3
8         1     1   0.1
--- Frequency Table of RoofStyle ---
  category  freq   pct
0    Gable  1141  78.2
1      Hip   286  19.6
2     Flat    13   0.9
3  Gambrel    11   0.8
4  Mansard     7   0.5
5     Shed     2   0.1
--- Frequency Table of RoofMatl ---
  category  freq   pct
0  CompShg  1434  98.2
1  Tar&Grv    11   0.8
2  WdShngl     6   0.4
3  WdShake     5   0.3
4    Metal     1   0.1
5  Membran     1   0.1
6     Roll     1   0.1
7  ClyTile     1   0.1
--- Frequency Table of Exterior1st ---
   category  freq   pct
0   VinylSd   515  35.3
1   HdBoard   222  15.2
2   MetalSd   220  15.1
3   Wd Sdng   206  14.1
4   Plywood   108   7.4
5   CemntBd    61   4.2
6   BrkFace    50   3.4
7   WdShing    26   1.8
8    Stucco    25   1.7
9   AsbShng    20   1.4
10  BrkComm     2   0.1
11    Stone     2   0.1
12  AsphShn     1   0.1
13  ImStucc     1   0.1
14   CBlock     1   0.1
--- Frequency Table of Exterior2nd ---
   category  freq   pct
0   VinylSd   504  34.5
1   MetalSd   214  14.7
2   HdBoard   207  14.2
3   Wd Sdng   197  13.5
4   Plywood   142   9.7
5   CmentBd    60   4.1
```

```
6   Wd Shng    38   2.6
7    Stucco    26   1.8
8   BrkFace    25   1.7
9   AsbShng    20   1.4
10  ImStucc    10   0.7
11  Brk Cmn     7   0.5
12    Stone     5   0.3
13  AsphShn     3   0.2
14    Other     1   0.1
15   CBlock     1   0.1
--- Frequency Table of MasVnrType ---
  category  freq   pct
0     none   874  59.9
1  BrkFace   444  30.4
2    Stone   127   8.7
3   BrkCmn    15   1.0
--- Frequency Table of ExterQual ---
  category  freq   pct
0       TA   906  62.1
1       Gd   488  33.4
2       Ex    52   3.6
3       Fa    14   1.0
--- Frequency Table of ExterCond ---
  category  freq   pct
0       TA  1282  87.8
1       Gd   146  10.0
2       Fa    28   1.9
3       Ex     3   0.2
4       Po     1   0.1
--- Frequency Table of Foundation ---
  category  freq   pct
0    PConc   647  44.3
1   CBlock   634  43.4
2   BrkTil   146  10.0
3     Slab    24   1.6
4    Stone     6   0.4
5     Wood     3   0.2
--- Frequency Table of BsmtQual ---
  category  freq   pct
0       TA   649  44.5
1       Gd   618  42.3
2       Ex   121   8.3
3       NA    37   2.5
4       Fa    35   2.4
--- Frequency Table of BsmtCond ---
  category  freq   pct
0       TA  1311  89.8
1       Gd    65   4.5
2       Fa    45   3.1
3       NA    37   2.5
4       Po     2   0.1
--- Frequency Table of BsmtExposure ---
  category  freq   pct
0       No   954  65.3
1       Av   221  15.1
2       Gd   134   9.2
3       Mn   114   7.8
4       NA    37   2.5
--- Frequency Table of BsmtFinType1 ---
  category  freq   pct
0      Unf   430  29.5
1      GLQ   418  28.6
2      ALQ   220  15.1
3      BLQ   148  10.1
```

```
4      Rec   133   9.1
5      LwQ    74   5.1
6       NA    37   2.5
--- Frequency Table of BsmtFinType2 ---
  category  freq   pct
0      Unf  1257  86.1
1      Rec    54   3.7
2      LwQ    46   3.2
3       NA    37   2.5
4      BLQ    33   2.3
5      ALQ    19   1.3
6      GLQ    14   1.0
--- Frequency Table of Heating ---
  category  freq   pct
0     GasA  1428  97.8
1     GasW    18   1.2
2     Grav     7   0.5
3     Wall     4   0.3
4     OthW     2   0.1
5    Floor     1   0.1
--- Frequency Table of HeatingQC ---
  category  freq   pct
0       Ex   741  50.8
1       TA   428  29.3
2       Gd   241  16.5
3       Fa    49   3.4
4       Po     1   0.1
--- Frequency Table of CentralAir ---
  category  freq   pct
0        Y  1365  93.5
1        N    95   6.5
--- Frequency Table of Electrical ---
  category  freq   pct
0    SBrkr  1335  91.4
1    FuseA    94   6.4
2    FuseF    27   1.8
3    FuseP     3   0.2
4      Mix     1   0.1
--- Frequency Table of KitchenQual ---
  category  freq   pct
0       TA   735  50.3
1       Gd   586  40.1
2       Ex   100   6.8
3       Fa    39   2.7
--- Frequency Table of Functional ---
  category  freq   pct
0      Typ  1360  93.2
1     Min2    34   2.3
2     Min1    31   2.1
3      Mod    15   1.0
4     Maj1    14   1.0
5     Maj2     5   0.3
6      Sev     1   0.1
--- Frequency Table of FireplaceQu ---
  category  freq   pct
0       NA   690  47.3
1       Gd   380  26.0
2       TA   313  21.4
3       Fa    33   2.3
4       Ex    24   1.6
5       Po    20   1.4
--- Frequency Table of GarageType ---
  category  freq   pct
0   Attchd   870  59.6
```

```
1   Detchd   387  26.5
2  BuiltIn    88   6.0
3       NA    81   5.5
4  Basment    19   1.3
5  CarPort     9   0.6
6   2Types     6   0.4
--- Frequency Table of GarageFinish ---
  category  freq   pct
0      Unf   605  41.4
1      RFn   422  28.9
2      Fin   352  24.1
3       NA    81   5.5
--- Frequency Table of GarageQual ---
  category  freq   pct
0       TA  1311  89.8
1       NA    81   5.5
2       Fa    48   3.3
3       Gd    14   1.0
4       Ex     3   0.2
5       Po     3   0.2
--- Frequency Table of GarageCond ---
  category  freq   pct
0       TA  1326  90.8
1       NA    81   5.5
2       Fa    35   2.4
3       Gd     9   0.6
4       Po     7   0.5
5       Ex     2   0.1
--- Frequency Table of PavedDrive ---
  category  freq   pct
0        Y  1340  91.8
1        N    90   6.2
2        P    30   2.1
--- Frequency Table of PoolQC ---
  category  freq   pct
0       NA  1454  99.6
1       Ex     2   0.1
2       Fa     2   0.1
3       Gd     2   0.1
--- Frequency Table of Fence ---
  category  freq   pct
0       NA  1179  80.8
1    MnPrv   157  10.8
2    GdPrv    59   4.0
3     GdWo    54   3.7
4     MnWw    11   0.8
--- Frequency Table of MiscFeature ---
  category  freq   pct
0       NA  1406  96.3
1     Shed    49   3.4
2     Gar2     2   0.1
3     Othr     2   0.1
4     TenC     1   0.1
--- Frequency Table of SaleType ---
  category  freq   pct
0       WD  1267  86.8
1      New   122   8.4
2      COD    43   2.9
3    ConLD     9   0.6
4    ConLI     5   0.3
5    ConLw     5   0.3
6      CWD     4   0.3
7      Oth     3   0.2
8      Con     2   0.1
```

```
--- Frequency Table of SaleCondition ---
  category  freq   pct
0   Normal  1198  82.1
1  Partial   125   8.6
2  Abnorml   101   6.9
3   Family    20   1.4
4   Alloca    12   0.8
5  AdjLand     4   0.3
```

In [975…

```
len(var_ind_cat1)
```

Out[975]:

```
46
```

In [976…

```
len(var_ind_num1)
```

Out[976]:

```
34
```

In [977…

```
len(var_ind_cat1)+len(var_ind_num1)
```

Out[977]:

```
80
```

In [978…

```
data.head(4)
```

Out[978]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilit
0	1	60	RL	65	8450	Pave	NA	Reg	Lvl	AllP
1	2	20	RL	80	9600	Pave	NA	Reg	Lvl	AllP
2	3	60	RL	68	11250	Pave	NA	IR1	Lvl	AllP
3	4	70	RL	60	9550	Pave	NA	IR1	Lvl	AllP

4 rows × 83 columns

Data Prepration

In [979…

```
# Remove added coloumns
data.drop(columns=['mvs','mvs_pct'],inplace=True)
data.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    1460 non-null   int64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          1460 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     1460 non-null   object
 26  MasVnrArea     1460 non-null   int64
 27  ExterQual      1460 non-null   object
 28  ExterCond      1460 non-null   object
 29  Foundation     1460 non-null   object
 30  BsmtQual       1460 non-null   object
 31  BsmtCond       1460 non-null   object
 32  BsmtExposure   1460 non-null   object
 33  BsmtFinType1   1460 non-null   object
 34  BsmtFinSF1     1460 non-null   int64
 35  BsmtFinType2   1460 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     1460 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    1460 non-null   object
 58  GarageType     1460 non-null   object
```

```
 59  GarageYrBlt    1460 non-null   int64
 60  GarageFinish   1460 non-null   object
 61  GarageCars     1460 non-null   int64
 62  GarageArea     1460 non-null   int64
 63  GarageQual     1460 non-null   object
 64  GarageCond     1460 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         1460 non-null   object
 73  Fence          1460 non-null   object
 74  MiscFeature    1460 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: int64(38), object(43)
memory usage: 924.0+ KB
```

Convert Ordinal Variables in to Numeric (Label Encoding)

In [980…

```python
# Convert 'LotShape' into numeric variable
data['cnvrt_LotShape']=data['LotShape'].replace(['Reg','IR1','IR2','IR3'],
                                                list(range(4, 0,-1)),inplace = Fal
#Check the results
print(np.sum(data['cnvrt_LotShape'].isnull()))
data['cnvrt_LotShape'].describe()
# data['cnvrt_LotShape']
```

```
0
```

Out[980]:

```
count    1460.000000
mean        3.591781
std         0.582296
min         1.000000
25%         3.000000
50%         4.000000
75%         4.000000
max         4.000000
Name: cnvrt_LotShape, dtype: float64
```

In [981…

```python
# Convert 'LandSlope' into numeric variable
data['cnvrt_LandSlope']=data['LandSlope'].replace(['Gtl','Mod','Sev'],
                                                list(range(3, 0,-1)), inplace = Fa
#Check the results
print(np.sum(data['cnvrt_LandSlope'].isnull()))
data['cnvrt_LandSlope'].describe()
# data['cnvrt_LandSlope']
```

```
0
```

Out[981]:
```
count    1460.000000
mean        2.937671
std         0.276232
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         3.000000
Name: cnvrt_LandSlope, dtype: float64
```

In [982…
```
# Convert 'ExterQua' into numeric variable
data['cnvrt_ExterQual']=data['ExterQual'].replace(['Ex','Gd','TA','Fa','Po'],
                                                  list(range(5, 0,-1)), inplace = Fa
#Check the results
print(np.sum(data['cnvrt_ExterQual'].isnull()))
data['cnvrt_ExterQual'].describe()
# data['cnvrt_ExterQual']
```

```
0
```

Out[982]:
```
count    1460.00000
mean        3.39589
std         0.57428
min         2.00000
25%         3.00000
50%         3.00000
75%         4.00000
max         5.00000
Name: cnvrt_ExterQual, dtype: float64
```

In [983…
```
# Convert 'ExterCond' into numeric variable
data['cnvrt_ExterCond']=data['ExterCond'].replace(['Ex','Gd','TA','Fa','Po'],
                                                  list(range(5, 0,-1)), inplace = Fa
#Check the results
print(np.sum(data['cnvrt_ExterCond'].isnull()))
data['cnvrt_ExterCond'].describe()
# data['cnvrt_ExterCond']
```

```
0
```

Out[983]:
```
count    1460.000000
mean        3.083562
std         0.351054
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         5.000000
Name: cnvrt_ExterCond, dtype: float64
```

In [984…
```
# Convert 'BsmtQual' into numeric variable
data['cnvrt_BsmtQual']=data['BsmtQual'].replace(['Ex','Gd','TA','Fa','Po','NA'],
                                                  list(range(6, 0,-1)), inplace = Fa
#Check the results
print(np.sum(data['cnvrt_BsmtQual'].isnull()))
data['cnvrt_BsmtQual'].describe()
# data['cnvrt_BsmtQual']
```

```
0
```

Out[984]:
```
count    1460.000000
mean        4.489041
std         0.876478
min         1.000000
25%         4.000000
50%         5.000000
75%         5.000000
max         6.000000
Name: cnvrt_BsmtQual, dtype: float64
```

In [985…
```
# Convert 'BsmtCond' into numeric variable
data['cnvrt_BsmtCond']=data['BsmtCond'].replace(['Ex','Gd','TA','Fa','Po','NA'],
                                                 list(range(6,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_BsmtCond'].isnull()))
data['cnvrt_BsmtCond'].describe()
# data['cnvrt_BsmtCond']
```

```
0
```

Out[985]:
```
count    1460.000000
mean        3.934932
std         0.552159
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         5.000000
Name: cnvrt_BsmtCond, dtype: float64
```

In [986…
```
# Convert 'BsmtExposure' into numeric variable
data['cnvrt_BsmtExposure']=data['BsmtExposure'].replace(['Gd','Av','Mn','No','NA'],
                                                 list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_BsmtExposure'].isnull()))
data['cnvrt_BsmtExposure'].describe()
# data['cnvrt_BsmtExposure']
```

```
0
```

Out[986]:
```
count    1460.000000
mean        2.630822
std         1.066665
min         1.000000
25%         2.000000
50%         2.000000
75%         3.000000
max         5.000000
Name: cnvrt_BsmtExposure, dtype: float64
```

In [987…
```
# Convert 'BsmtFinType1' into numeric variable
data['cnvrt_BsmtFinType1']=data['BsmtFinType1'].replace(['GLQ','ALQ','BLQ','Rec','L
                                                 list(range(7,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_BsmtFinType1'].isnull()))
data['cnvrt_BsmtFinType1'].describe()
# data['cnvrt_BsmtFinType1']
```

```
0
```

Out[987]:

```
count    1460.000000
mean        4.545890
std         2.107776
min         1.000000
25%         2.000000
50%         5.000000
75%         7.000000
max         7.000000
Name: cnvrt_BsmtFinType1, dtype: float64
```

In [988…

```
# Convert 'BsmtFinType2' into numeric variable
data['cnvrt_BsmtFinType2']=data['BsmtFinType2'].replace(['GLQ','ALQ','BLQ','Rec','L
                                                  list(range(7,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_BsmtFinType2'].isnull()))
data['cnvrt_BsmtFinType2'].describe()
# data['cnvrt_BsmtFinType2']
```

```
0
```

Out[988]:

```
count    1460.000000
mean        2.247945
std         0.891758
min         1.000000
25%         2.000000
50%         2.000000
75%         2.000000
max         7.000000
Name: cnvrt_BsmtFinType2, dtype: float64
```

In [989…

```
# Convert 'HeatingQC' into numeric variable
data['cnvrt_HeatingQC']=data['HeatingQC'].replace(['Ex','Gd','TA','Fa','Po'],
                                                  list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_HeatingQC'].isnull()))
data['cnvrt_HeatingQC'].describe()
# data['cnvrt_HeatingQC']
```

```
0
```

Out[989]:

```
count    1460.000000
mean        4.145205
std         0.959501
min         1.000000
25%         3.000000
50%         5.000000
75%         5.000000
max         5.000000
Name: cnvrt_HeatingQC, dtype: float64
```

In [990…

```
# Convert 'KitchenQual' into numeric variable
data['cnvrt_KitchenQual']=data['KitchenQual'].replace(['Ex','Gd','TA','Fa','Po'],
                                                  list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(data['cnvrt_KitchenQual'].isnull()))
data['cnvrt_KitchenQual'].describe()
# data['cnvrt_KitchenQual']
```

```
0
```

Out[990]:
```
count    1460.000000
mean        3.511644
std         0.663760
min         2.000000
25%         3.000000
50%         3.000000
75%         4.000000
max         5.000000
Name: cnvrt_KitchenQual, dtype: float64
```

In [991…
```
# Convert 'Functional' into numeric variable
data['cnvrt_Functional']=data['Functional'].replace(['Typ','Min1','Min2','Mod','Ma
                                                    list(range(8,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_Functional'].isnull()))
data['cnvrt_Functional'].describe()
# data['cnvrt_Functional']
```

```
0
```

Out[991]:
```
count    1460.000000
mean        7.841781
std         0.667698
min         2.000000
25%         8.000000
50%         8.000000
75%         8.000000
max         8.000000
Name: cnvrt_Functional, dtype: float64
```

In [992…
```
# Convert 'FireplaceQu' into numeric variable
data['cnvrt_FireplaceQu']=data['FireplaceQu'].replace(['Ex','Gd','TA','Fa','Po','NA
                                                    list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_FireplaceQu'].isnull()))
data['cnvrt_FireplaceQu'].describe()
# data['cnvrt_FireplaceQu']
```

```
0
```

Out[992]:
```
count    1460.000000
mean        2.825342
std         1.810877
min         1.000000
25%         1.000000
50%         3.000000
75%         5.000000
max         6.000000
Name: cnvrt_FireplaceQu, dtype: float64
```

In [993…
```
# Convert 'GarageFinish' into numeric variable
data['cnvrt_GarageFinish']=data['GarageFinish'].replace(['Fin','RFn','Unf','NA'],
                                                    list(range(4,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_GarageFinish'].isnull()))
data['cnvrt_GarageFinish'].describe()
# data['cnvrt_GarageFinish']
```

```
0
```

Out[993]:
```
count    1460.000000
mean        2.715753
std         0.892831
min         1.000000
25%         2.000000
50%         3.000000
75%         3.000000
max         4.000000
Name: cnvrt_GarageFinish, dtype: float64
```

In [994…
```
# Convert 'GararageQual' into numeric variable
data['cnvrt_GarageQual']=data['GarageQual'].replace(['Ex','Gd','TA','Fa','Po','NA']
                                                   list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_GarageQual'].isnull()))
data['cnvrt_GarageQual'].describe()
# data['cnvrt_GarageQual']
```

```
0
```

Out[994]:
```
count    1460.000000
mean        3.810274
std         0.722898
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         6.000000
Name: cnvrt_GarageQual, dtype: float64
```

In [995…
```
# Convert 'GararageCond' into numeric variable
data['cnvrt_GarageCond']=data['GarageCond'].replace(['Ex','Gd','TA','Fa','Po','NA']
                                                   list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_GarageCond'].isnull()))
data['cnvrt_GarageCond'].describe()
# data['cnvrt_GarageCond']
```

```
0
```

Out[995]:
```
count    1460.000000
mean        3.808904
std         0.719685
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         6.000000
Name: cnvrt_GarageCond, dtype: float64
```

In [996…
```
# Convert 'PavedDrive' into numeric variable
data['cnvrt_PavedDrive']=data['PavedDrive'].replace(['Y','P','N'],
                                                   list(range(3,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_PavedDrive'].isnull()))
data['cnvrt_PavedDrive'].describe()
# data['cnvrt_PavedDrive']
```

```
0
```

Out[996]:
```
count    1460.000000
mean        2.856164
std         0.496592
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         3.000000
Name: cnvrt_PavedDrive, dtype: float64
```

In [997…
```
# Convert 'PoolQC' into numeric variable
data['cnvrt_PoolQC']=data['PoolQC'].replace(['Ex','Gd','TA','Fa','NA'],
                                            list(range(5,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_PoolQC'].isnull()))
data['cnvrt_PoolQC'].describe()
# data['cnvrt_PoolQC']
```

Out[997]:
```
0
count    1460.000000
mean        1.010959
std         0.188469
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         5.000000
Name: cnvrt_PoolQC, dtype: float64
```

In [998…
```
# Convert 'Fence' into numeric variable
data['cnvrt_Fence']=data['Fence'].replace(['GdPrv','MnPrv','GdWo','MnWw','NA'],
                                            list(range(5,0,-1)), inplace = Fal

#Check the results
print(np.sum(data['cnvrt_Fence'].isnull()))
data['cnvrt_Fence'].describe()
# data['cnvrt_Fence']
```

Out[998]:
```
0
count    1460.000000
mean        1.565753
std         1.204483
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         5.000000
Name: cnvrt_Fence, dtype: float64
```

In [999…
```
np.sum(data.isnull())
```

Out[999]:
```
Id                  0
MSSubClass          0
MSZoning            0
LotFrontage         0
LotArea             0
                   ..
cnvrt_GarageQual    0
cnvrt_GarageCond    0
cnvrt_PavedDrive    0
cnvrt_PoolQC        0
cnvrt_Fence         0
Length: 100, dtype: int64
```

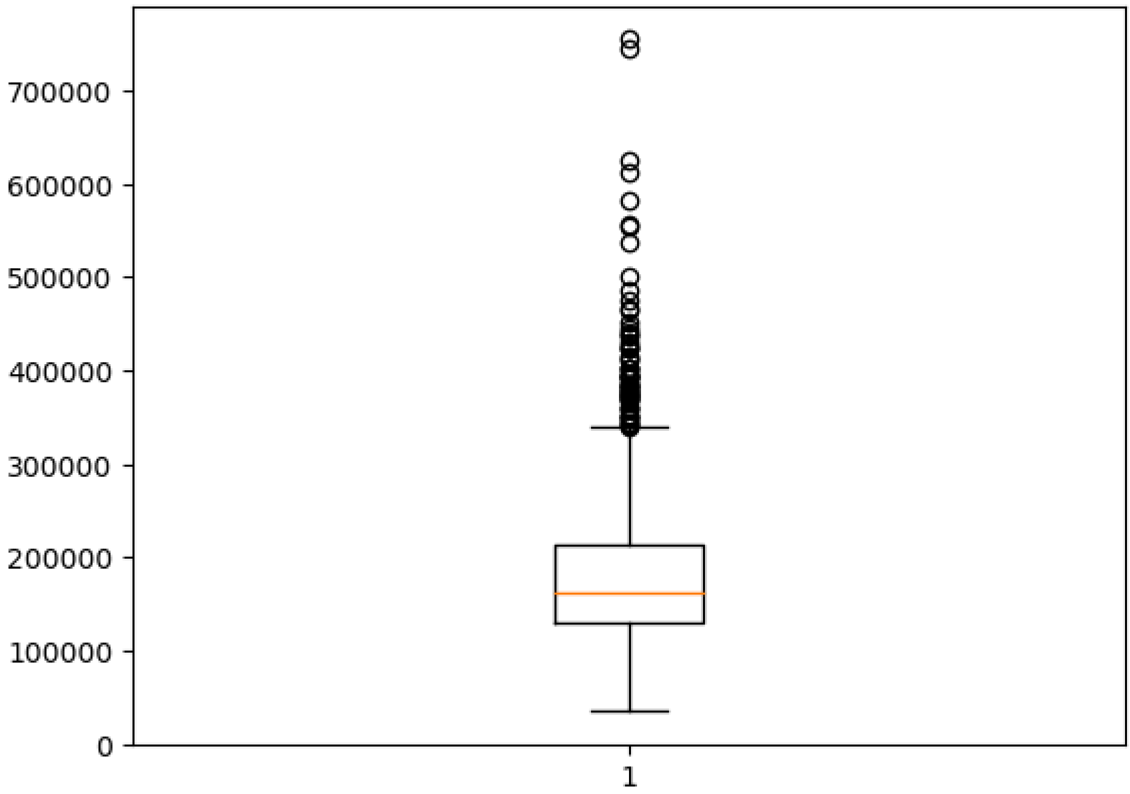
Outliers

In [100…

```
# data.info()
```

In [100…

```
# Check for outliers
plt.boxplot(data['SalePrice'])
plt.show()
```



In [100…

```
from scipy.stats import iqr
df=data
# Indices of variables to check for outliers
var_ind_num1=[3, 4, 19, 20, 26, 34, 36, 37, 38]+list(range(43, 53))+[54, 56, 59, 61

# Function to detect outliers
def outlier_detector(df,var_ind_num1):
    outlier_flags=pd.DataFrame(0,index=df.index,columns=[df.columns[i] for i in var
    for i in var_ind_num1:
        if i<len(df.columns) and np.issubdtype(df.iloc[:, i].dtype,np.number):
            col = df.iloc[:, i]
            lower_bound=np.quantile(col, 0.25) - 1.5 * iqr(col)
            upper_bound=np.quantile(col, 0.75) + 1.5 * iqr(col)
            outliers=(col<lower_bound) | (col>upper_bound)
            outlier_flags.loc[outliers, df.columns[i]] = 1
    return outlier_flags

# Detect outliers
outlier_flags=outlier_detector(df,var_ind_num1)

# Calculate the number of outliers per column (feature)
outlier_counts_per_feature=outlier_flags.sum(axis=0)

# Calculate the percentage of outliers per column (feature)
total_data_points=len(df)
percentage_outliers_per_feature=(outlier_counts_per_feature/total_data_points)*100
```

```
# Combine the counts and percentages into a single DataFrame
outlier_summary=pd.DataFrame({
    'Number of Outliers':outlier_counts_per_feature,
    'Percentage of Outliers':percentage_outliers_per_feature
})

# Find the feature with the most outliers
most_outliers_feature=outlier_counts_per_feature.idxmax()
most_outliers_count=outlier_counts_per_feature.max()
percentage_outliers_most_feature=percentage_outliers_per_feature[most_outliers_feat

# Sort and display the outlier summary DataFrame
sorted_outlier_summary=outlier_summary.sort_values(by='Number of Outliers', ascendi
print("\nSorted outlier summary:")
print(sorted_outlier_summary)
```

```
Sorted outlier summary:
               Number of Outliers  Percentage of Outliers
EnclosedPorch                 208               14.246575
BsmtFinSF2                    167               11.438356
ScreenPorch                   116                7.945205
MasVnrArea                     98                6.712329
BsmtHalfBath                   82                5.616438
GarageYrBlt                    81                5.547945
OpenPorchSF                    77                5.273973
LotArea                        69                4.726027
KitchenAbvGr                   68                4.657534
SalePrice                      61                4.178082
TotalBsmtSF                    61                4.178082
MiscVal                        52                3.561644
BedroomAbvGr                   35                2.397260
WoodDeckSF                     32                2.191781
GrLivArea                      31                2.123288
TotRmsAbvGrd                   30                2.054795
BsmtUnfSF                      29                1.986301
LowQualFinSF                   26                1.780822
3SsnPorch                      24                1.643836
GarageArea                     21                1.438356
1stFlrSF                       20                1.369863
LotFrontage                    16                1.095890
BsmtFinSF1                      7                0.479452
PoolArea                        7                0.479452
YearBuilt                       7                0.479452
GarageCars                      5                0.342466
Fireplaces                      5                0.342466
2ndFlrSF                        2                0.136986
BsmtFullBath                    1                0.068493
HalfBath                        0                0.000000
FullBath                        0                0.000000
YearRemodAdd                    0                0.000000
MoSold                          0                0.000000
YrSold                          0                0.000000
```

In [100…

```
# data.drop(['EnclosedPorch','BsmtFinSF2'],axis=1,inplace=True)
print(data.shape)
```

```
(1460, 100)
```

In [100…

```
# Indices of variables to check for outliers
var_ind_num1 =[3,4,19,20,26,34,36,37,38]+list(range(43,53))+[54,56,59,61,62]+list(r

# Function to detect outliers
def outlier_detector(df,var_ind_num1):
    outlier_flags = pd.DataFrame(0,index=df.index,columns=[df.columns[i] for i in v
    for i in var_ind_num1:
```

```
        if i < len(df.columns)and np.issubdtype(df.iloc[:,i].dtype, np.number):
            col=df.iloc[:, i]
            lower_bound=np.quantile(col,0.25)-1.5*iqr(col)
            upper_bound=np.quantile(col,0.75)+1.5*iqr(col)
            outliers=(col<lower_bound)| (col>upper_bound)
            outlier_flags.loc[outliers,df.columns[i]] = 1
    return outlier_flags

# Detect outliers
outlier_flags=outlier_detector(df,var_ind_num1)

# Calculate the number of outliers per row
outlier_counts=outlier_flags.sum(axis=1)

# Sort the outlier counts
sorted_outlier_counts=outlier_counts.sort_values(ascending=False)
print(sorted_outlier_counts)
```

```
1298    12
1182     8
523      8
197      8
691      8
        ..
715      0
719      0
721      0
722      0
730      0
Length: 1460, dtype: int64
```

In [100…

```
# data.iloc[[1298,1182,523], :]
```

In [100…

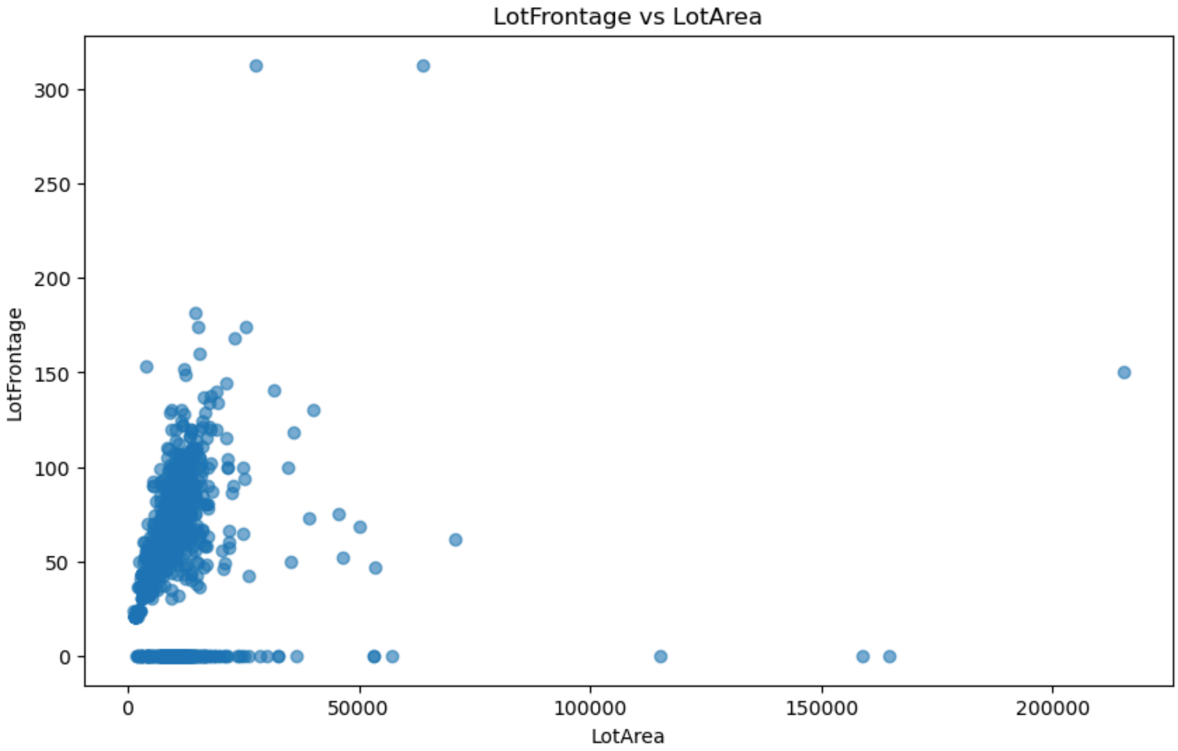
```
data.shape
```

Out[1006]:

```
(1460, 100)
```

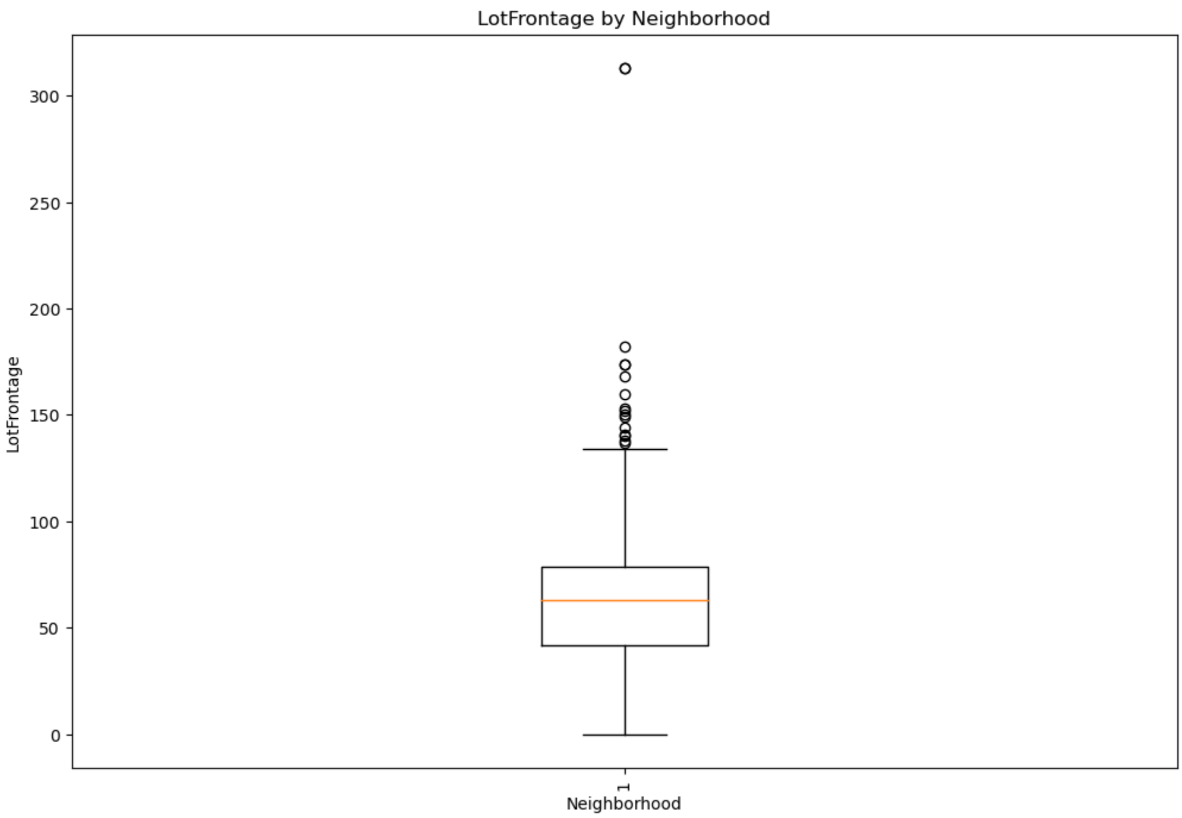
In [100…

```
# Scatter plot of LotFrontage vs LotArea
plt.figure(figsize=(10, 6))

# Enhanced scatter plot with color and marker size
plt.scatter(data['LotArea'],data['LotFrontage'],alpha=0.6)

plt.title('LotFrontage vs LotArea')
plt.xlabel('LotArea')
plt.ylabel('LotFrontage')
plt.show()
```



In [100…

```python
# Box plot of LotFrontage by Neighborhood
plt.figure(figsize=(12, 8))
plt.boxplot(data['LotFrontage'])
# plt.box(data['LotArea'], data['LotFrontage'])
plt.title('LotFrontage by Neighborhood')
plt.xlabel('Neighborhood')
plt.ylabel('LotFrontage')
plt.xticks(rotation=90)
plt.show()
```



In [100…

```python
summary_stats=data.groupby('Neighborhood')['LotFrontage'].describe()
print(summary_stats)
```

```
              count       mean        std   min    25%   50%     75%    max
Neighborhood
Blmngtn        17.0  38.823529  19.063169   0.0  43.00  43.0   53.00   53.0
Blueste         2.0  24.000000   0.000000  24.0  24.00  24.0   24.00   24.0
BrDale         16.0  21.562500   1.209339  21.0  21.00  21.0   21.00   24.0
BrkSide        58.0  50.568966  23.846815   0.0  50.00  51.0   60.00  144.0
ClearCr        28.0  38.750000  44.216324   0.0   0.00   0.0   80.00  138.0
CollgCr       150.0  60.213333  30.327427   0.0  57.75  68.0   75.00  122.0
Crawfor        51.0  57.725490  33.633363   0.0  47.00  61.0   80.00  130.0
Edwards       100.0  62.760000  36.232448   0.0  51.50  61.0   73.25  313.0
Gilbert        79.0  49.544304  46.712651   0.0   0.00  59.0   71.50  182.0
IDOTRR         37.0  57.432432  24.607520   0.0  50.00  60.0   60.00  120.0
MeadowV        17.0  24.529412  13.215132   0.0  21.00  21.0   36.00   44.0
Mitchel        49.0  51.489796  36.516619   0.0   0.00  62.0   77.00  129.0
NAmes         225.0  63.208889  36.076876   0.0  60.00  70.0   80.00  313.0
NPkVill         9.0  25.111111  18.784598   0.0  24.00  24.0   24.00   53.0
NWAmes         73.0  50.109589  40.656270   0.0   0.00  78.0   80.00  105.0
NoRidge        41.0  73.951220  43.264276   0.0  58.00  84.0   95.00  174.0
NridgHt        77.0  80.818182  26.521951   0.0  62.00  87.0  101.00  129.0
OldTown       113.0  60.566372  21.311414   0.0  50.00  60.0   63.00  153.0
SWISU          25.0  54.200000  19.181154   0.0  51.00  60.0   60.00  102.0
Sawyer         74.0  48.283784  37.679805   0.0   0.00  65.0   74.75  115.0
SawyerW        59.0  60.593220  29.246349   0.0  57.00  65.0   78.00  116.0
Somerst        86.0  58.651163  28.602660   0.0  35.00  72.0   78.00  116.0
StoneBr        25.0  50.160000  33.021054   0.0  40.00  44.0   77.00  124.0
Timber         38.0  63.263158  39.243305   0.0  44.75  73.0   87.25  150.0
Veenker        11.0  38.000000  36.066605   0.0   0.00  32.0   74.00   90.0
```

In [101…

```
#summary statistics of numeric variables
data.iloc[:,var_ind_num1].describe()
```

Out[1010]:

	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	Bsm
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.
mean	57.623288	10516.828082	1971.267808	1984.865753	103.681507	443.639726	46.
std	34.664304	9981.264932	30.202904	20.645407	180.569120	456.098091	161.
min	0.000000	1300.000000	1872.000000	1950.000000	0.000000	0.000000	0.
25%	42.000000	7553.500000	1954.000000	1967.000000	0.000000	0.000000	0.
50%	63.000000	9478.500000	1973.000000	1994.000000	0.000000	383.500000	0.
75%	79.000000	11601.500000	2000.000000	2004.000000	164.250000	712.250000	0.
max	313.000000	215245.000000	2010.000000	2010.000000	1600.000000	5644.000000	1474.

8 rows × 34 columns

Data understanding- Phase2

In [101…

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 100 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        1460 non-null   int64
 4   LotArea            1460 non-null   int64
 5   Street             1460 non-null   object
 6   Alley              1460 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         1460 non-null   object
 26  MasVnrArea         1460 non-null   int64
 27  ExterQual          1460 non-null   object
 28  ExterCond          1460 non-null   object
 29  Foundation         1460 non-null   object
 30  BsmtQual           1460 non-null   object
 31  BsmtCond           1460 non-null   object
 32  BsmtExposure       1460 non-null   object
 33  BsmtFinType1       1460 non-null   object
 34  BsmtFinSF1         1460 non-null   int64
 35  BsmtFinType2       1460 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         1460 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        1460 non-null   object
 58  GarageType         1460 non-null   object
```

```
 59  GarageYrBlt        1460 non-null   int64
 60  GarageFinish       1460 non-null   object
 61  GarageCars         1460 non-null   int64
 62  GarageArea         1460 non-null   int64
 63  GarageQual         1460 non-null   object
 64  GarageCond         1460 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             1460 non-null   object
 73  Fence              1460 non-null   object
 74  MiscFeature        1460 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
 81  cnvrt_LotShape     1460 non-null   int64
 82  cnvrt_LandSlope    1460 non-null   int64
 83  cnvrt_ExterQual    1460 non-null   int64
 84  cnvrt_ExterCond    1460 non-null   int64
 85  cnvrt_BsmtQual     1460 non-null   int64
 86  cnvrt_BsmtCond     1460 non-null   int64
 87  cnvrt_BsmtExposure 1460 non-null   int64
 88  cnvrt_BsmtFinType1 1460 non-null   int64
 89  cnvrt_BsmtFinType2 1460 non-null   int64
 90  cnvrt_HeatingQC    1460 non-null   int64
 91  cnvrt_KitchenQual  1460 non-null   int64
 92  cnvrt_Functional   1460 non-null   int64
 93  cnvrt_FireplaceQu  1460 non-null   int64
 94  cnvrt_GarageFinish 1460 non-null   int64
 95  cnvrt_GarageQual   1460 non-null   int64
 96  cnvrt_GarageCond   1460 non-null   int64
 97  cnvrt_PavedDrive   1460 non-null   int64
 98  cnvrt_PoolQC       1460 non-null   int64
 99  cnvrt_Fence        1460 non-null   int64
dtypes: int64(57), object(43)
memory usage: 1.1+ MB
```

In [101…

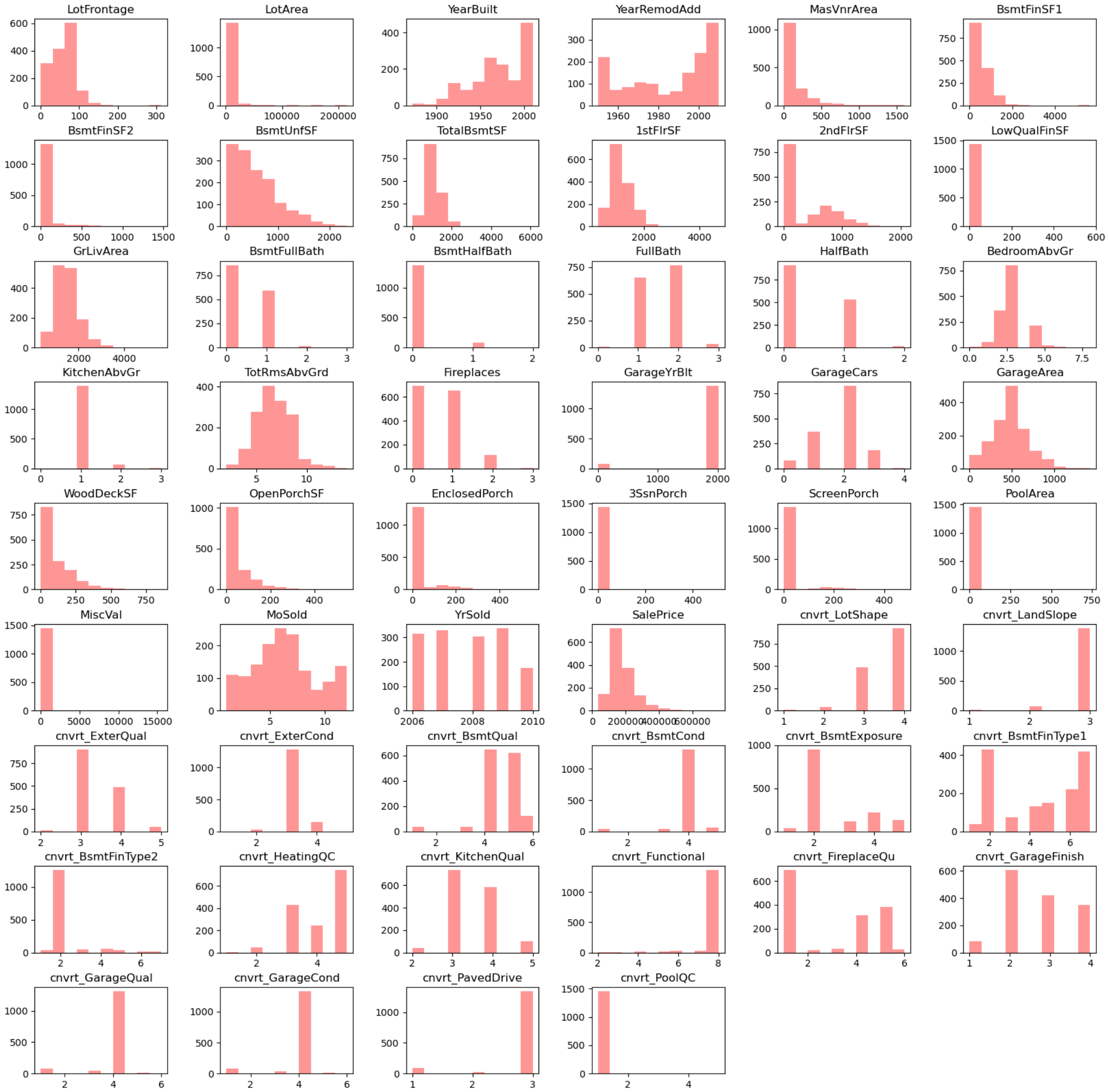
```
# List of all numeric variables after converting ordinal variables to numeric
var_ind_num2=[3,4,19,20,26,34,36,37,38]+list(range(43,53))+[54,56,59,61,62]+list(ra
# var_ind_num2
len(var_ind_num2)
```

Out[1012]:

```
53
```

In [101…

```
# Histogram of numeric variables
plt.figure(figsize=(20,20))
plt.subplots_adjust(hspace=0.4,wspace=0.4)
for i in range(1,53):
    plt.subplot(9,6,i)
    plt.hist(x=data.iloc[: ,var_ind_num2[i- 1]],
             alpha=0.4,color = 'red')
    plt.title(data.columns[var_ind_num2[i - 1]])
plt.show()
```



5: Bivariate data analysis

In [101…

```
#Correlation analysis after converting ordinal variables to numeric
corr_table=round(data.iloc[:,var_ind_num2].corr(method='pearson'),2)
corr_table
```

Out[1014]:

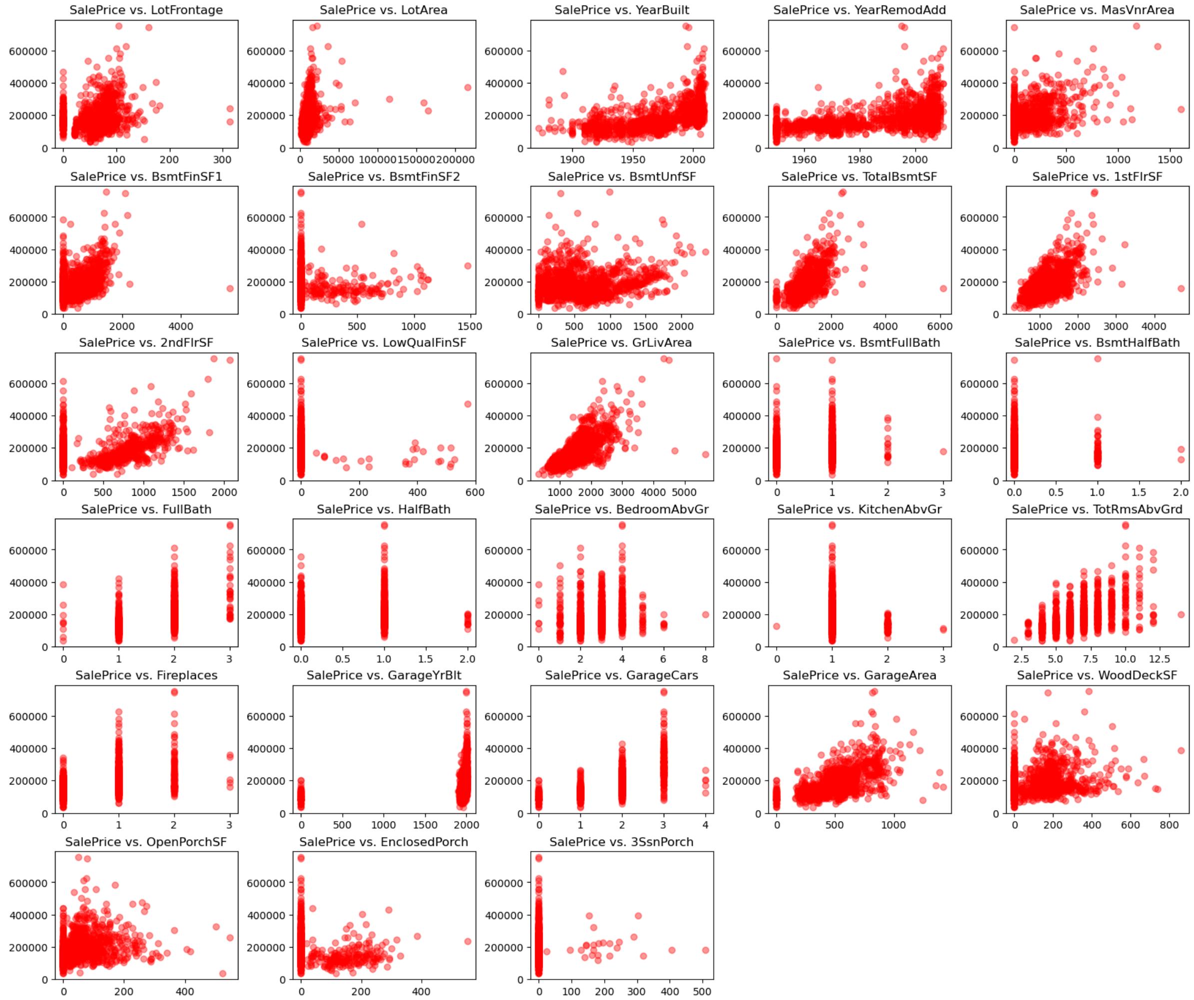
	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1
LotFrontage	1.00	0.10	0.04	0.08	0.10	0.08
LotArea	0.10	1.00	0.01	0.01	0.10	0.21
YearBuilt	0.04	0.01	1.00	0.59	0.31	0.25
YearRemodAdd	0.08	0.01	0.59	1.00	0.18	0.13
MasVnrArea	0.10	0.10	0.31	0.18	1.00	0.26
BsmtFinSF1	0.08	0.21	0.25	0.13	0.26	1.00
BsmtFinSF2	-0.01	0.11	-0.05	-0.07	-0.07	-0.05
BsmtUnfSF	0.16	-0.00	0.15	0.18	0.11	-0.50
TotalBsmtSF	0.24	0.26	0.39	0.29	0.36	0.52
1stFlrSF	0.25	0.30	0.28	0.24	0.34	0.45
2ndFlrSF	0.04	0.05	0.01	0.14	0.17	-0.14
LowQualFinSF	0.05	0.00	-0.18	-0.06	-0.07	-0.06
GrLivArea	0.22	0.26	0.20	0.29	0.39	0.21
BsmtFullBath	0.01	0.16	0.19	0.12	0.09	0.65
BsmtHalfBath	-0.03	0.05	-0.04	-0.01	0.03	0.07
FullBath	0.12	0.13	0.47	0.44	0.28	0.06
HalfBath	-0.01	0.01	0.24	0.18	0.20	0.00
BedroomAbvGr	0.14	0.12	-0.07	-0.04	0.10	-0.11
KitchenAbvGr	0.03	-0.02	-0.17	-0.15	-0.04	-0.08
TotRmsAbvGrd	0.22	0.19	0.10	0.19	0.28	0.04
Fireplaces	0.04	0.27	0.15	0.11	0.25	0.26
GarageYrBlt	0.02	0.07	0.27	0.15	0.13	0.12
GarageCars	0.17	0.15	0.54	0.42	0.36	0.22
GarageArea	0.20	0.18	0.48	0.37	0.37	0.30
WoodDeckSF	-0.02	0.17	0.22	0.21	0.16	0.20
OpenPorchSF	0.07	0.08	0.19	0.23	0.12	0.11
EnclosedPorch	0.03	-0.02	-0.39	-0.19	-0.11	-0.10
3SsnPorch	0.02	0.02	0.03	0.05	0.02	0.03
ScreenPorch	0.02	0.04	-0.05	-0.04	0.06	0.06
PoolArea	0.11	0.08	0.00	0.01	0.01	0.14
MiscVal	-0.06	0.04	-0.03	-0.01	-0.03	0.00
MoSold	0.02	0.00	0.01	0.02	-0.01	-0.02
YrSold	-0.01	-0.01	-0.01	0.04	-0.01	0.01
SalePrice	0.21	0.26	0.52	0.51	0.48	0.39
cnvrt_LotShape	0.11	-0.32	-0.23	-0.18	-0.09	-0.16
cnvrt_LandSlope	0.04	-0.44	0.07	0.06	0.02	-0.11

	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1
cnvrt_ExterQual	0.13	0.06	0.60	0.59	0.35	0.20
cnvrt_ExterCond	-0.05	0.01	-0.10	0.07	-0.02	0.04
cnvrt_BsmtQual	0.09	0.07	0.60	0.52	0.28	0.30
cnvrt_BsmtCond	0.02	0.03	0.18	0.19	0.07	0.17
cnvrt_BsmtExposure	0.07	0.23	0.29	0.22	0.17	0.37
cnvrt_BsmtFinType1	-0.02	0.06	0.35	0.22	0.19	0.70
cnvrt_BsmtFinType2	-0.02	0.09	-0.04	-0.04	-0.05	-0.01
cnvrt_HeatingQC	0.10	0.00	0.45	0.55	0.16	0.09
cnvrt_KitchenQual	0.13	0.07	0.53	0.63	0.29	0.23
cnvrt_Functional	0.00	-0.03	0.15	0.07	0.08	0.06
cnvrt_FireplaceQu	0.08	0.19	0.22	0.20	0.28	0.18
cnvrt_GarageFinish	0.09	0.12	0.60	0.44	0.28	0.25
cnvrt_GarageQual	0.01	0.08	0.29	0.15	0.14	0.14
cnvrt_GarageCond	0.01	0.08	0.29	0.14	0.13	0.14
cnvrt_PavedDrive	0.02	0.02	0.43	0.17	0.15	0.19
cnvrt_PoolQC	0.08	0.04	-0.01	-0.00	-0.03	0.07
cnvrt_Fence	0.01	-0.04	-0.21	-0.14	-0.10	-0.02

52 rows × 52 columns

In [101…

```
#Scatter plot for numeric variables after converting ordinal variables to numeric
plt.figure(figsize=(20,20))
plt.subplots_adjust(hspace=0.3,wspace=0.3)
for i in range(1,29):
    plt.subplot(7,5,i)
    plt.scatter(x=data.iloc[: ,var_ind_num2[i - 1]],y=data['SalePrice'],alpha=0.4,c
    plt.title('SalePrice vs. '+data.columns[var_ind_num2[i - 1]])
plt.show()
```

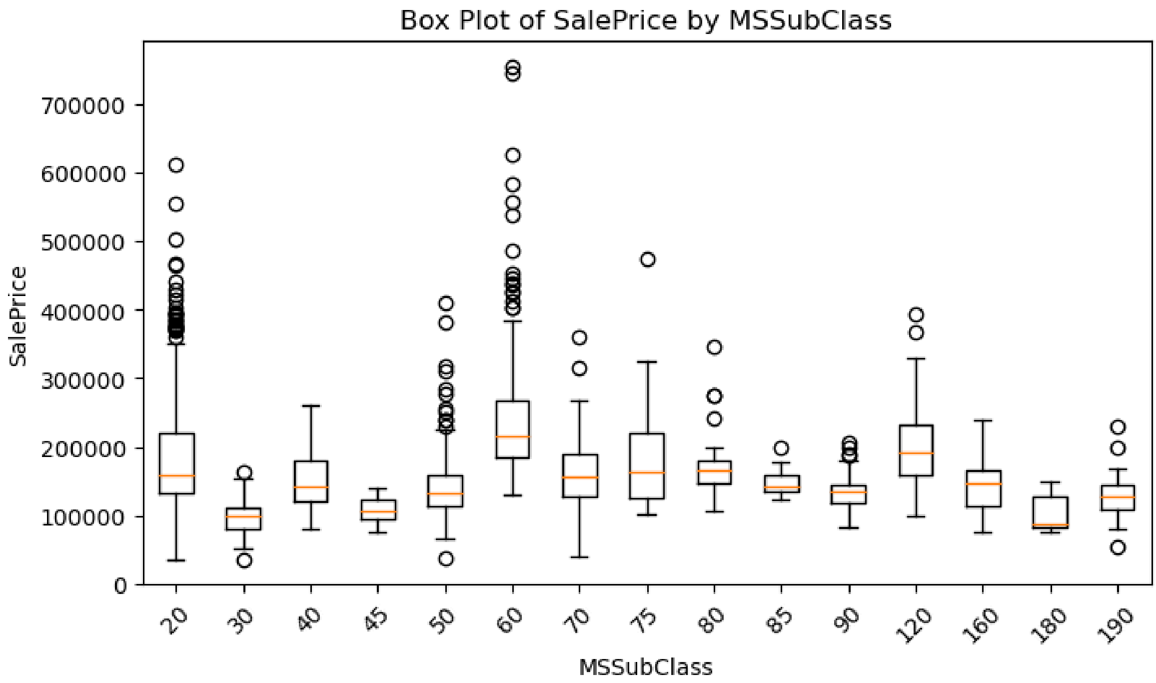


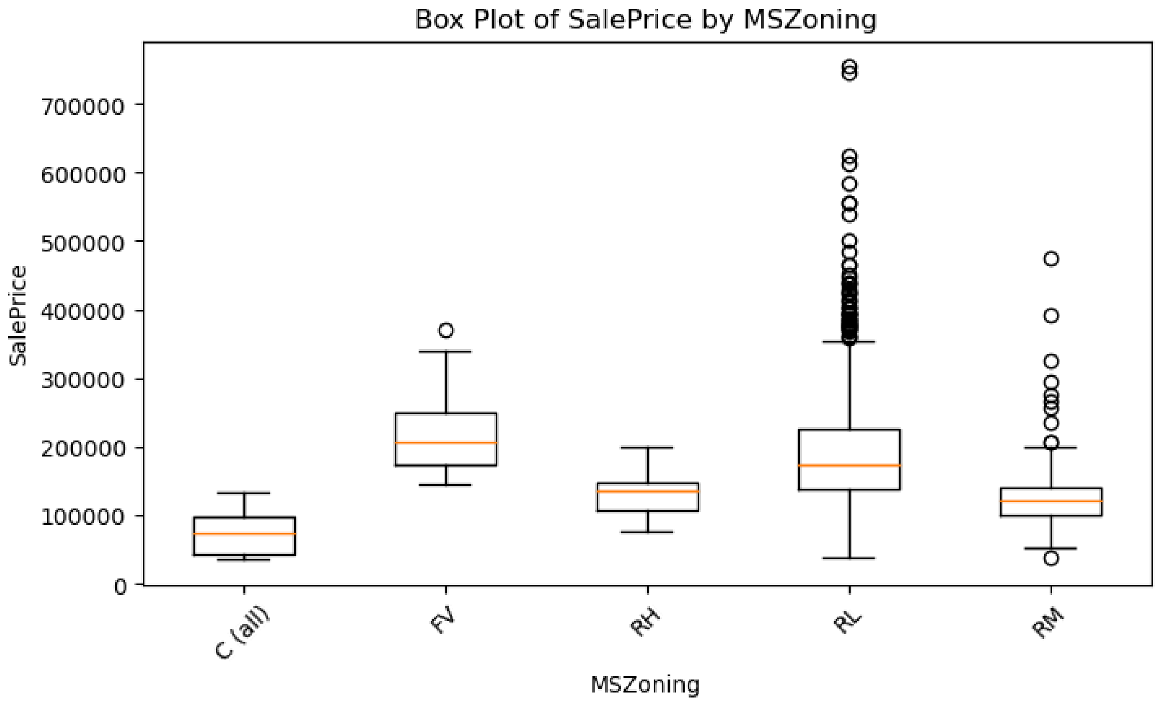
In [101…

```
var_ind_cat2=[1,2,5,6,8,9,10]+list(range(12,17))+list(range(21,26))+[29,39,41,42,58
```

In [101…

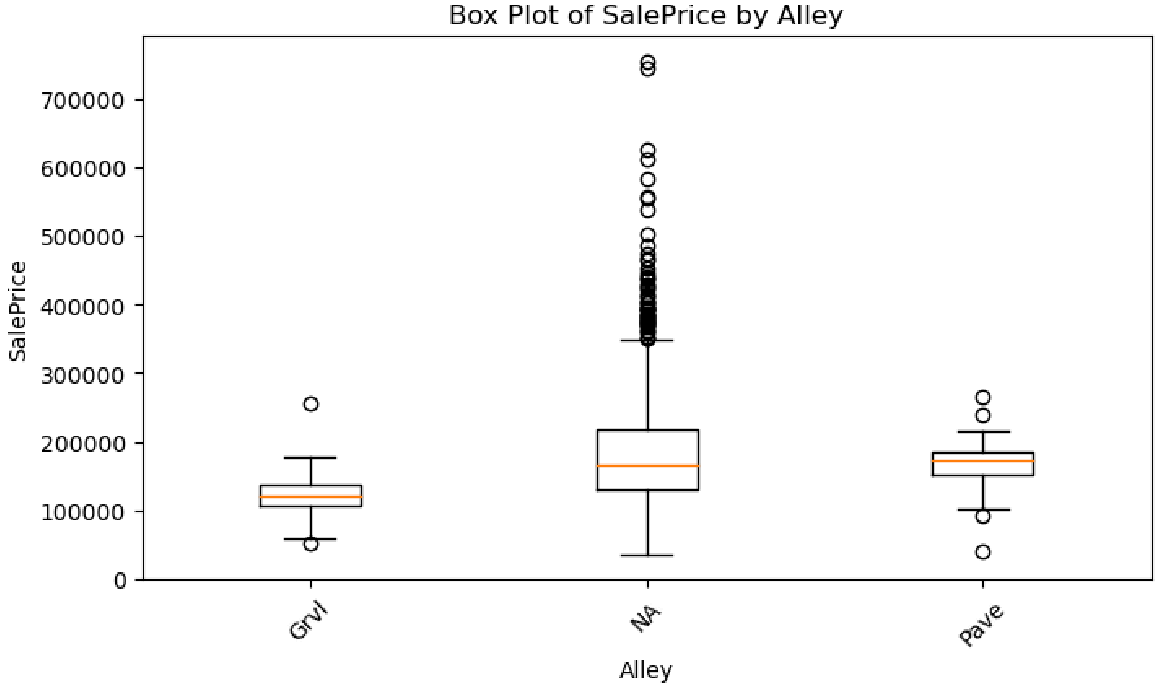
```
# list of categorical column
col_cat2=[data.columns[i] for i in var_ind_cat2]

# subplots
fig,axes=plt.subplots(len(col_cat2),1,figsize=(8,6*len(col_cat2)))

# If there's only one plot
if len(col_cat2)==1:
    axes=[axes]

# Loop over each categorical column and plot the boxplot
for ax,col in zip(axes,col_cat2):
    # Ensure categories are sorted to maintain consistent order
    categories=sorted(data[col].dropna().unique())

    # ensuring non-empty categories
    data_to_plot=[data[data[col]==category]['SalePrice'] for category in categories

    # Plot the boxplot
    ax.boxplot(data_to_plot,labels=categories)
    ax.set_title(f'Box Plot of SalePrice by {col}')
    ax.set_xlabel(col)
    ax.set_ylabel('SalePrice')
    ax.tick_params(axis='x',rotation=45)
plt.subplots_adjust(hspace=0.4)
plt.show()
```

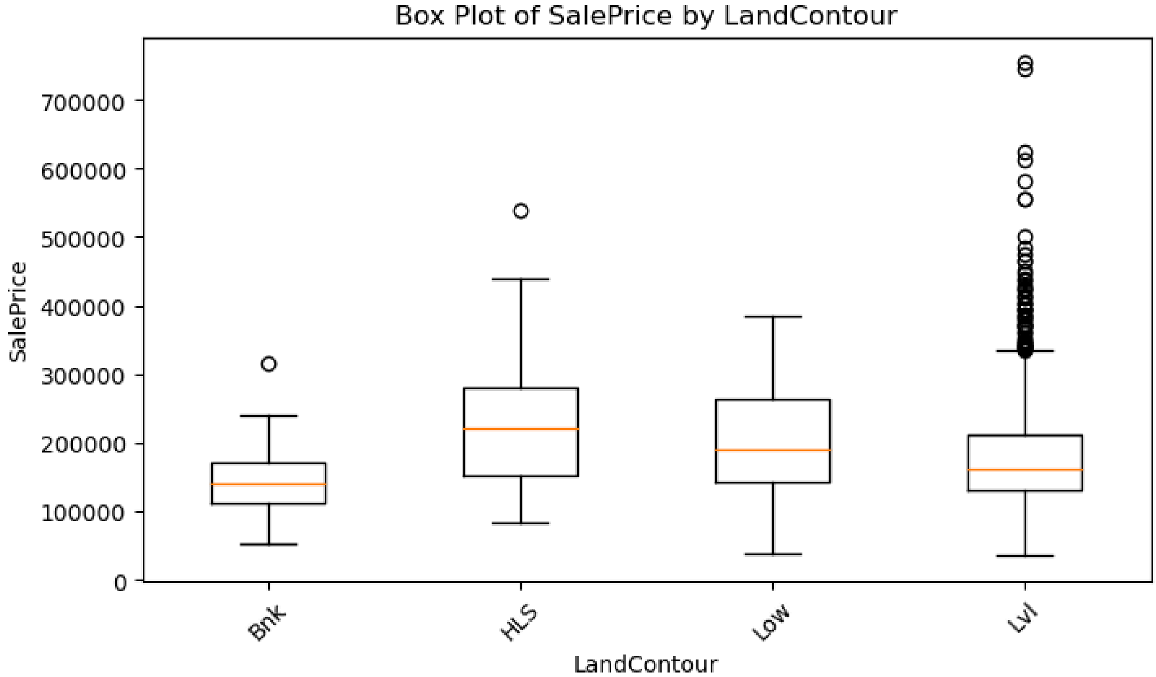

Box Plot of SalePrice by MSSubClass
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
20
30
40
45
50
60
70
75
80
85
90
120
160
180
190
MSSubClass

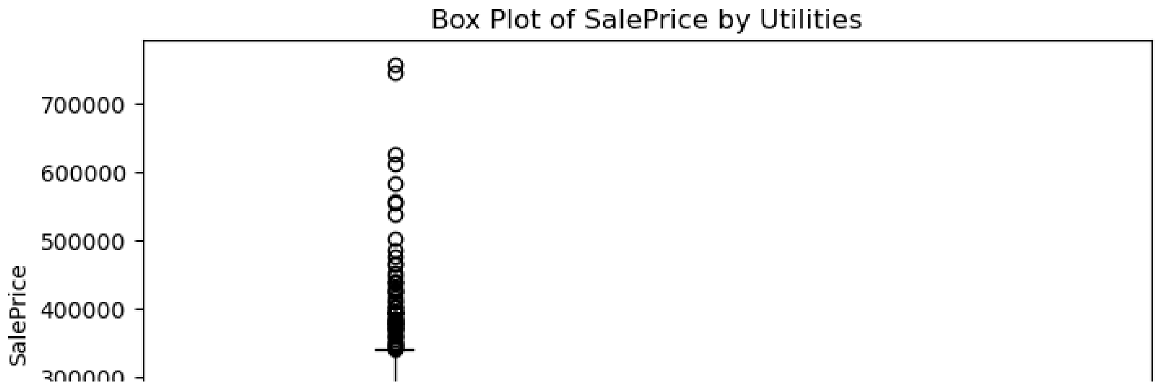

Box Plot of SalePrice by MSZoning
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
C (all)
FV
RH
RL
RM
MSZoning

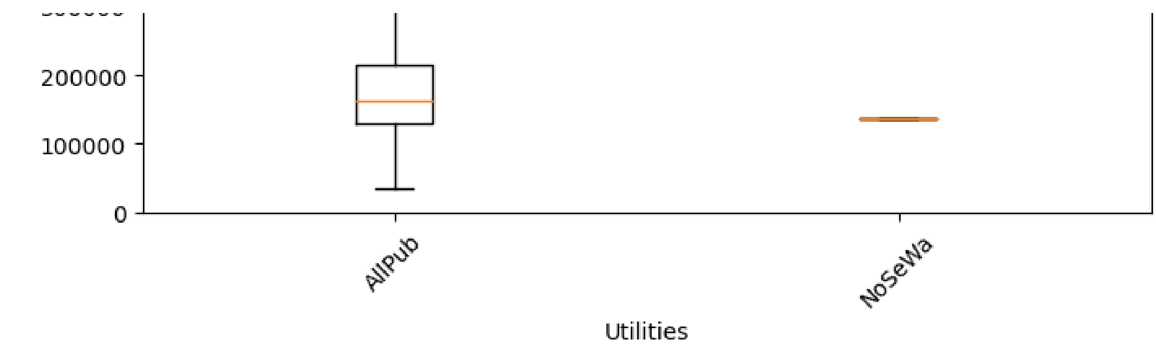

Box Plot of SalePrice by Street
SalePrice
100000
200000
300000
400000
500000
600000
700000


0
Grvl
Pave
Street

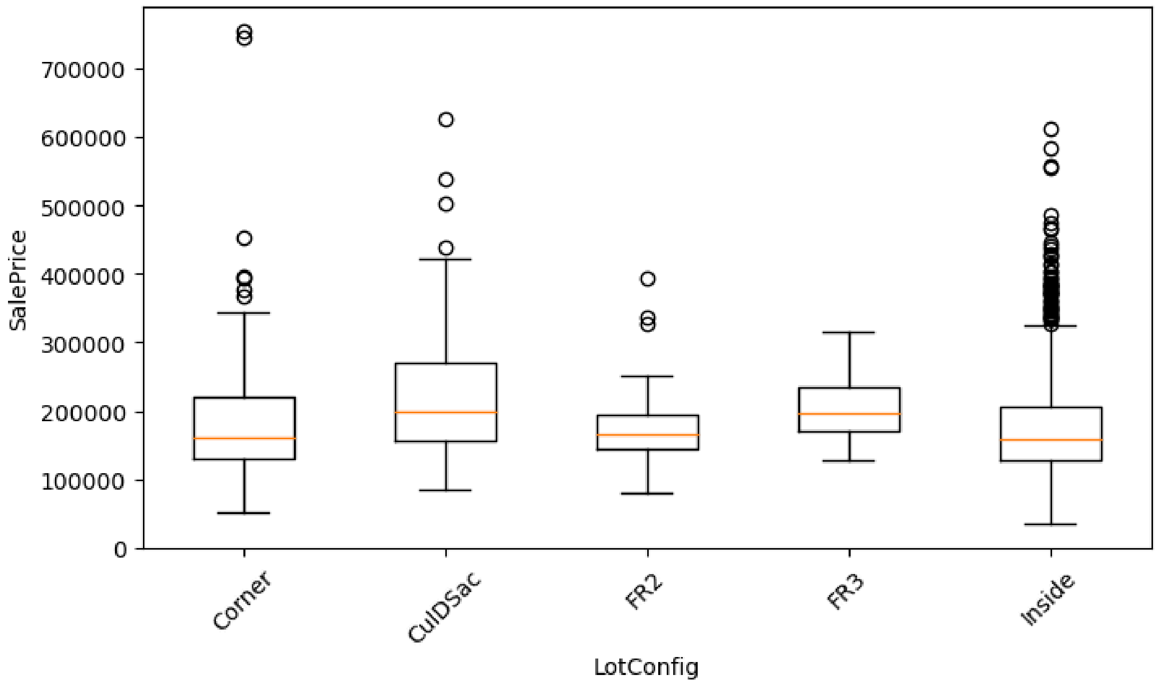

Box Plot of SalePrice by Alley
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Grvl
NA
Pave
Alley


Box Plot of SalePrice by LandContour
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Bnk
HLS
Low
Lvl
LandContour

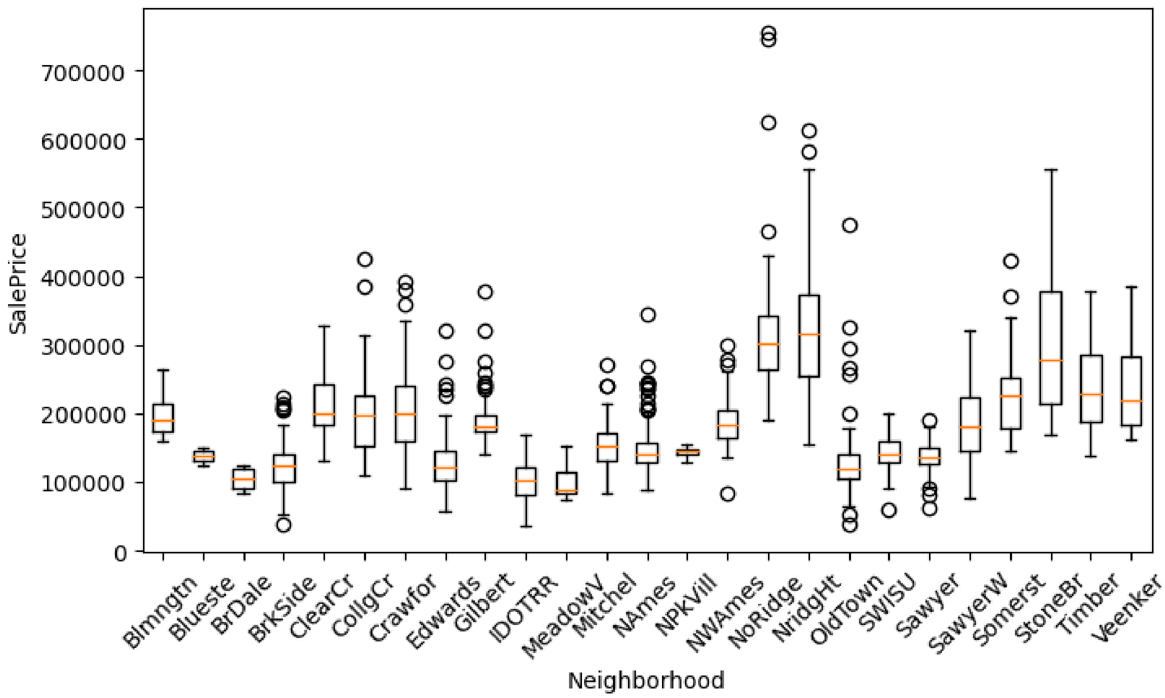

Box Plot of SalePrice by Utilities
SalePrice
400000
500000
600000
700000



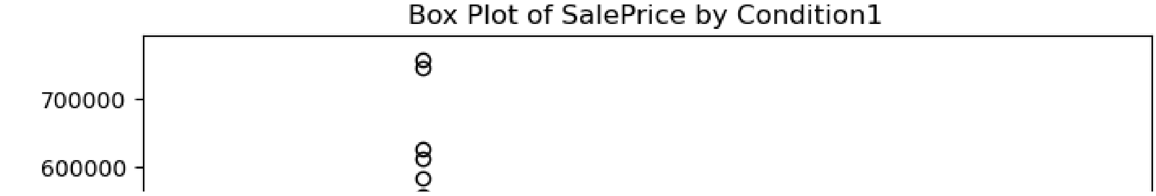
Box Plot of SalePrice by LotConfig

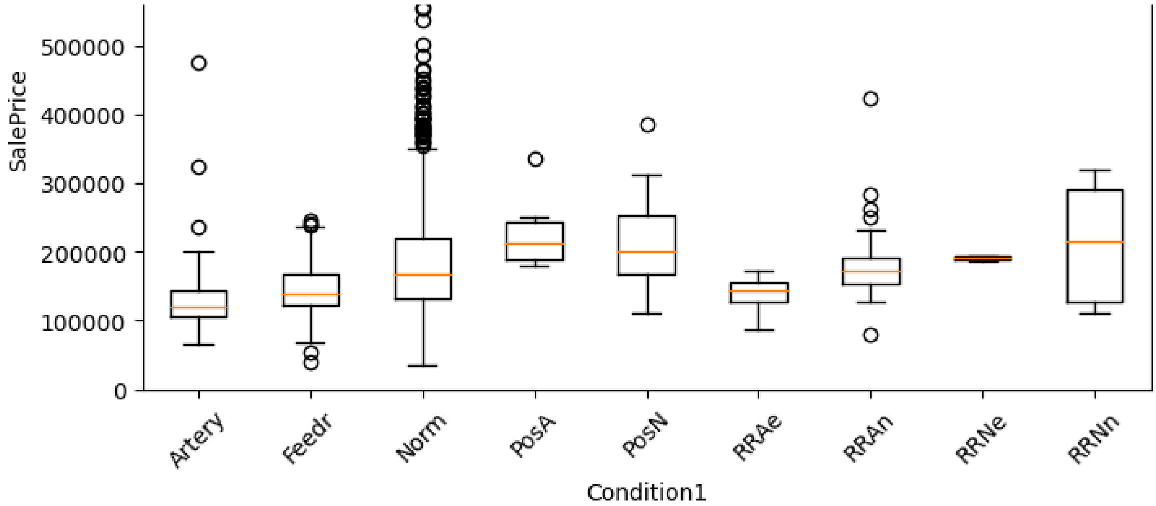


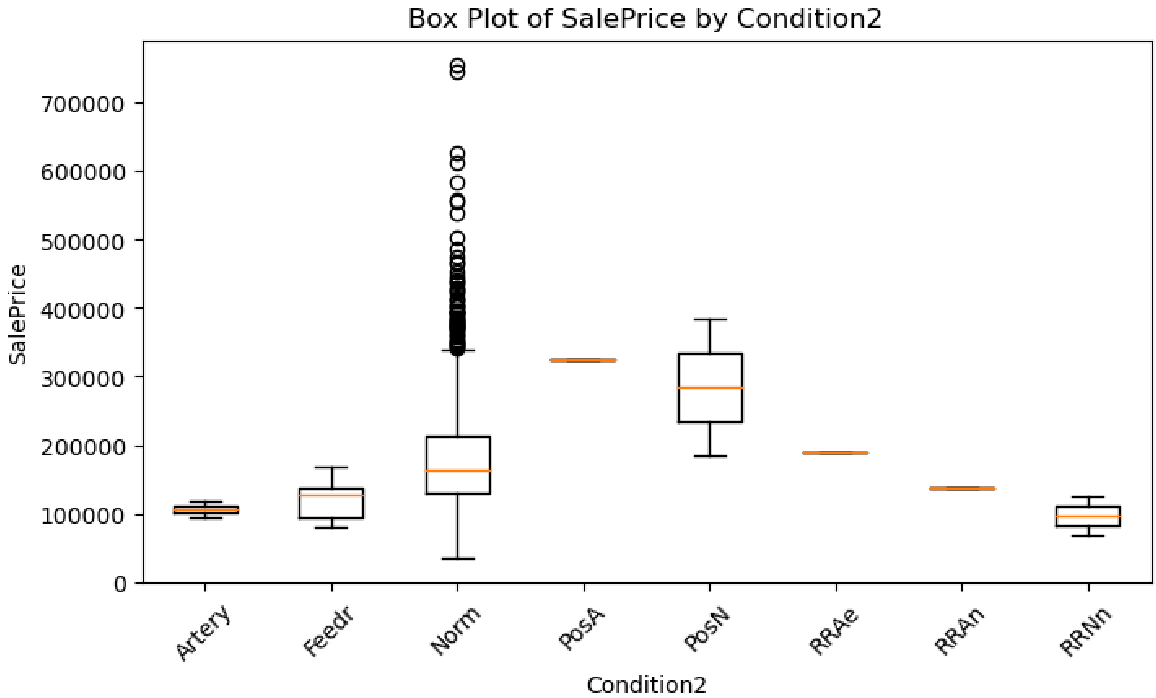
Box Plot of SalePrice by Neighborhood

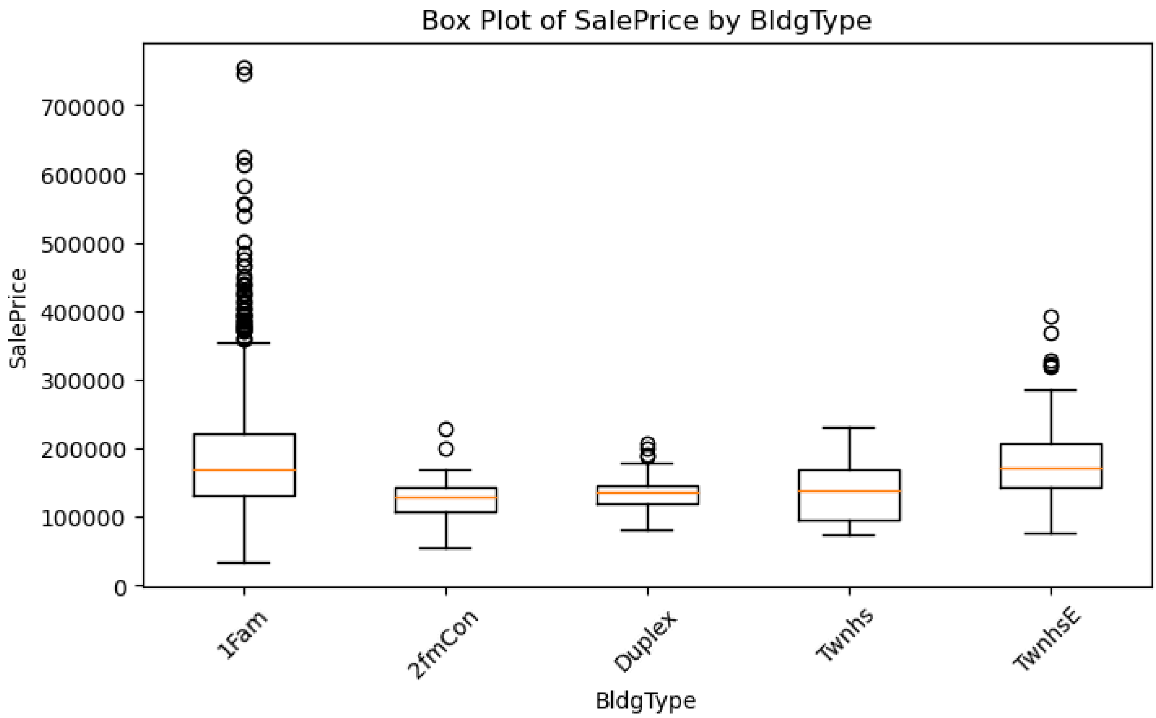


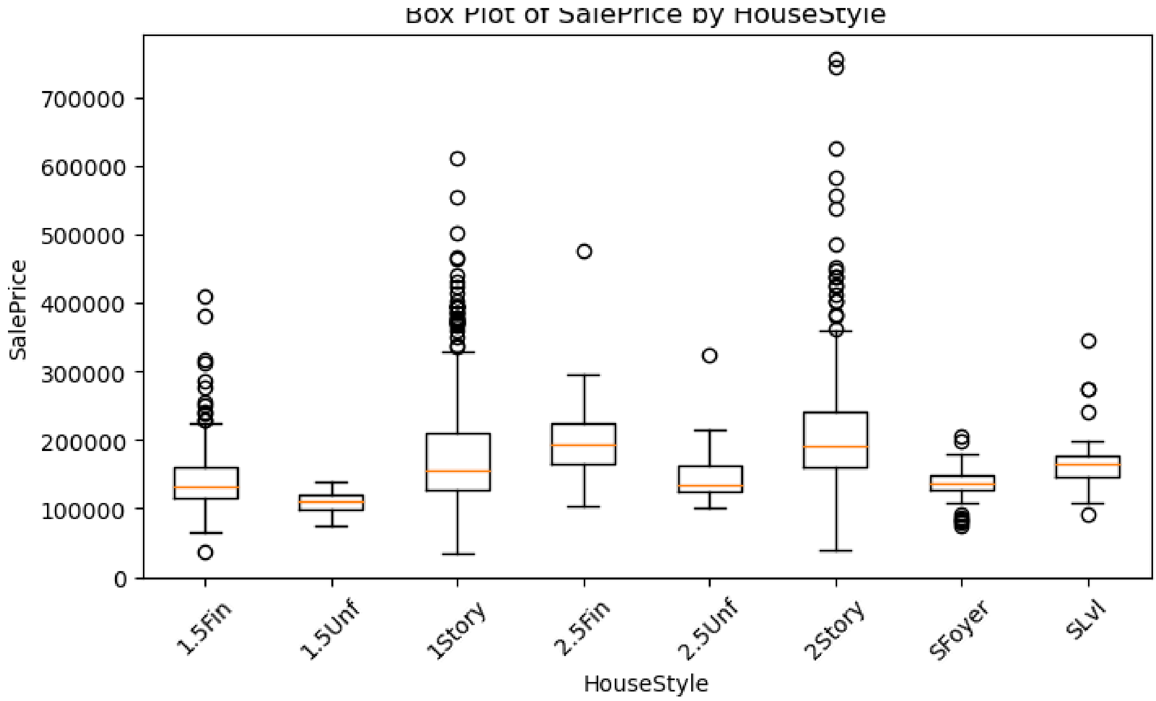
Box Plot of SalePrice by Condition1

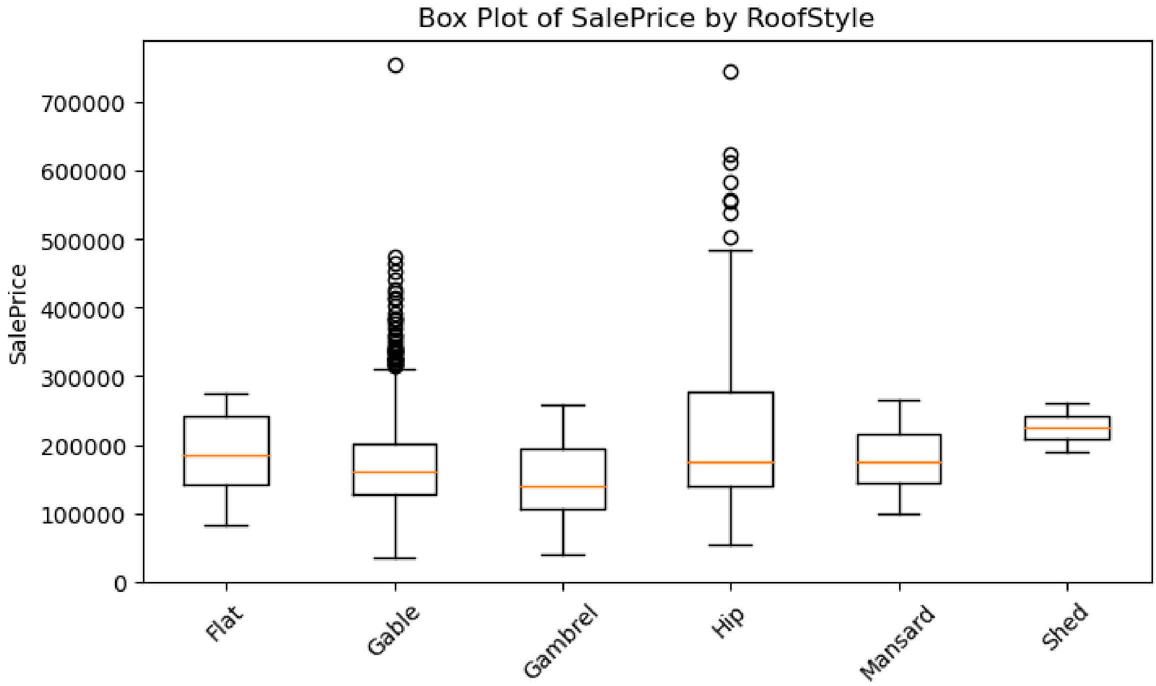


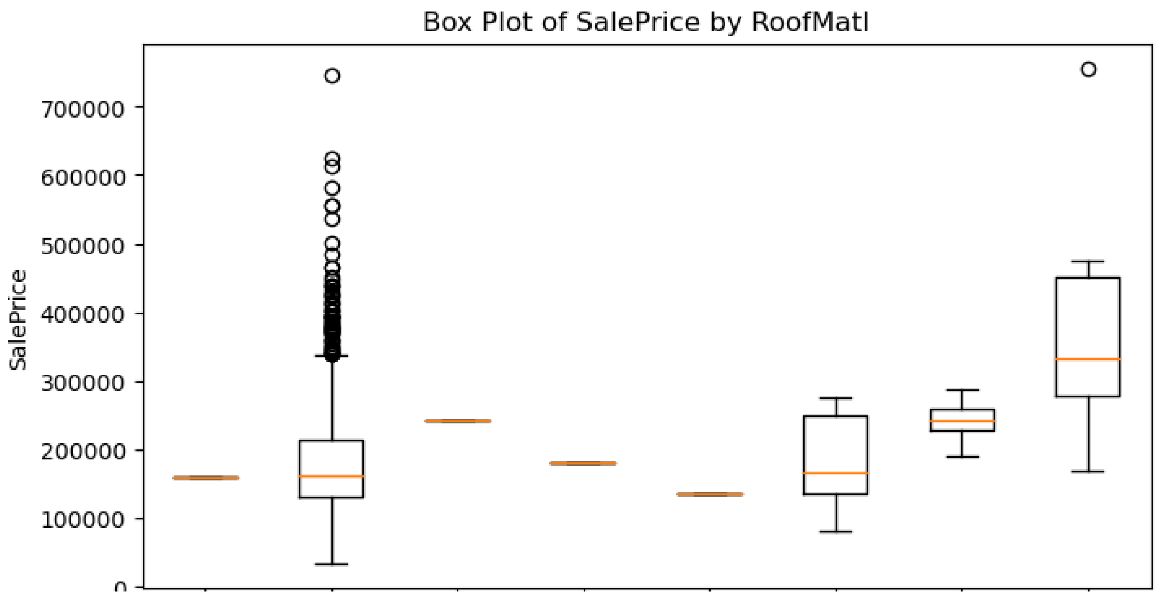

SalePrice
500000
400000
300000
200000
100000
0
Artery
Feedr
Norm
PosA
PosN
RRAe
RRAn
RRNe
RRNn
Condition1

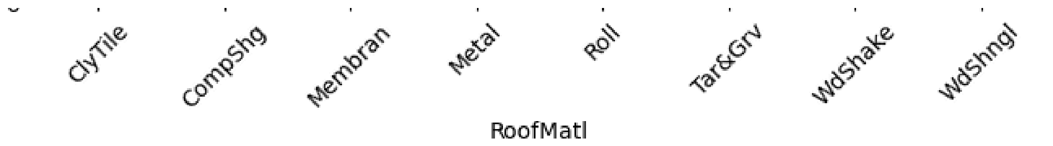

Box Plot of SalePrice by Condition2
SalePrice
700000
600000
500000
400000
300000
200000
100000
0
Artery
Feedr
Norm
PosA
PosN
RRAe
RRAn
RRNn
Condition2

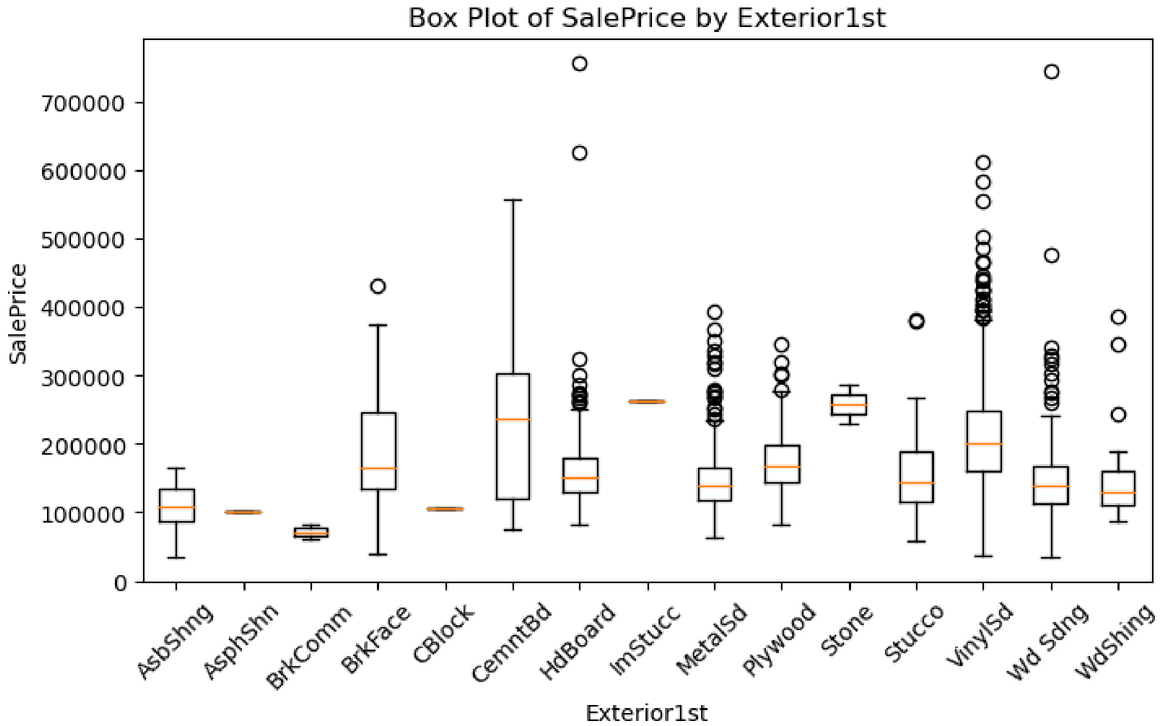

Box Plot of SalePrice by BldgType
SalePrice
700000
600000
500000
400000
300000
200000
100000
0
1Fam
2fmCon
Duplex
Twnhs
TwnhsE
BldgType

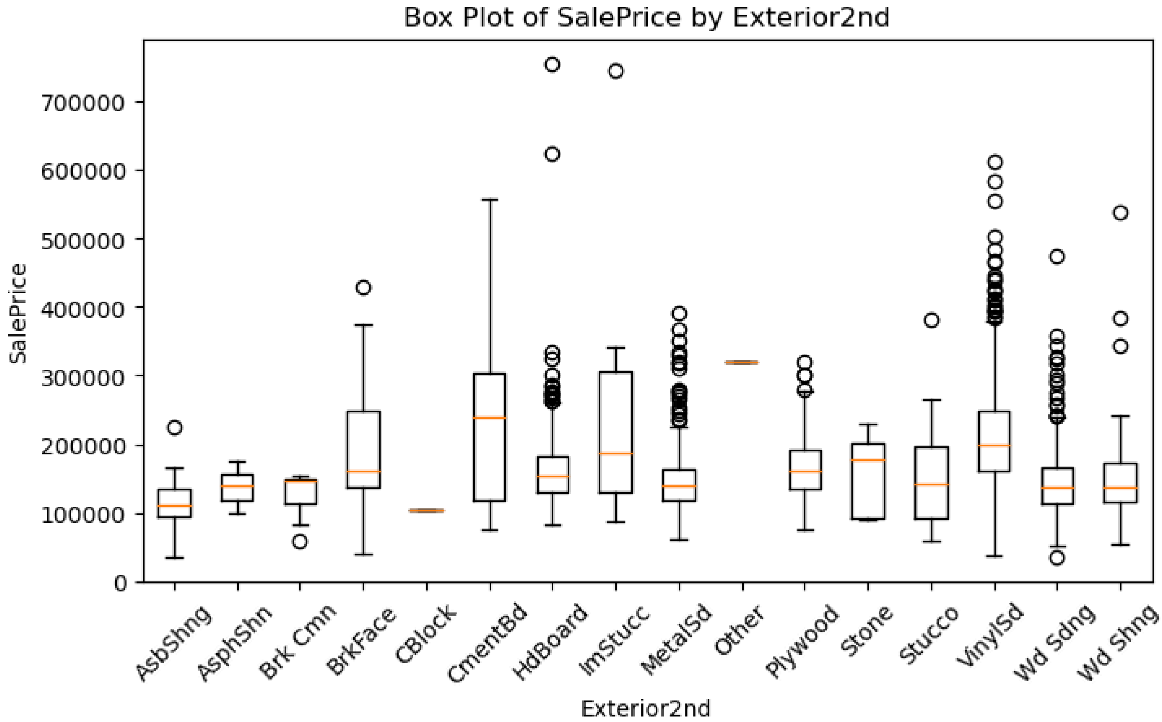

Box Plot of SalePrice by HouseStyle
SalePrice
HouseStyle
0
100000
200000
300000
400000
500000
600000
700000
1.5Fin
1.5Unf
1Story
2.5Fin
2.5Unf
2Story
SFoyer
SLvl

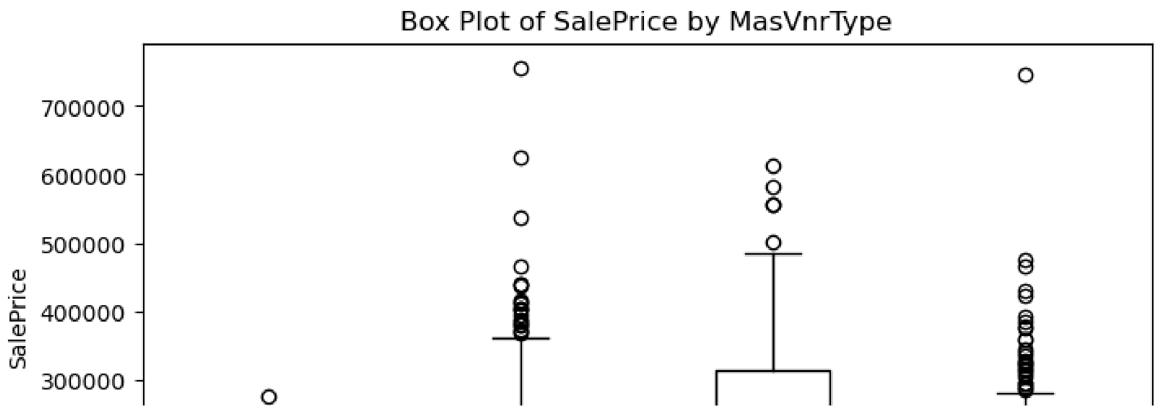

Box Plot of SalePrice by RoofStyle
SalePrice
RoofStyle
0
100000
200000
300000
400000
500000
600000
700000
Flat
Gable
Gambrel
Hip
Mansard
Shed

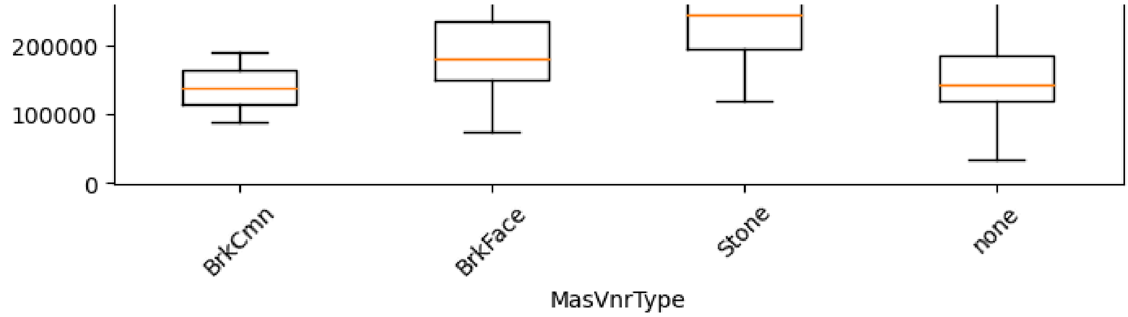

Box Plot of SalePrice by RoofMatl
SalePrice
100000
200000
300000
400000
500000
600000
700000


ClyTile
CompShg
Membran
Metal
Roll
Tar&Grv
WdShake
WdShngl
RoofMatl

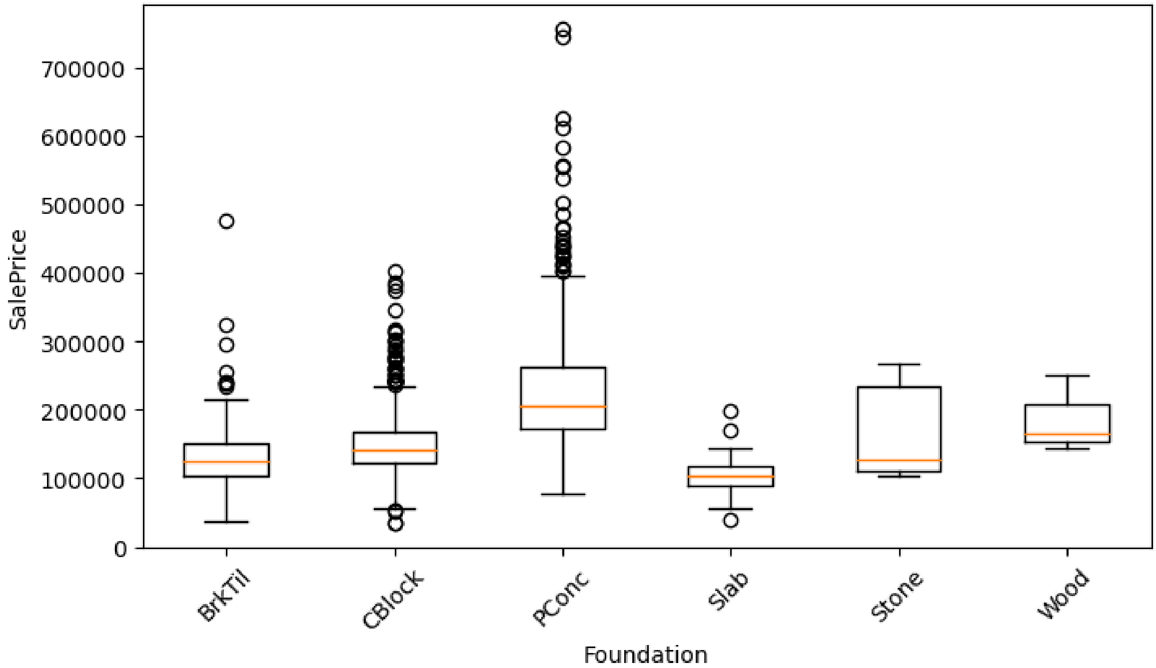

Box Plot of SalePrice by Exterior1st
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
AsbShng
AsphShn
BrkComm
BrkFace
CBlock
CemntBd
HdBoard
ImStucc
MetalSd
Plywood
Stone
Stucco
VinylSd
Wd Sdng
WdShing
Exterior1st


Box Plot of SalePrice by Exterior2nd
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
AsbShng
AsphShn
Brk Cmn
BrkFace
CBlock
CmentBd
HdBoard
ImStucc
MetalSd
Other
Plywood
Stone
Stucco
VinylSd
Wd Sdng
Wd Shng
Exterior2nd

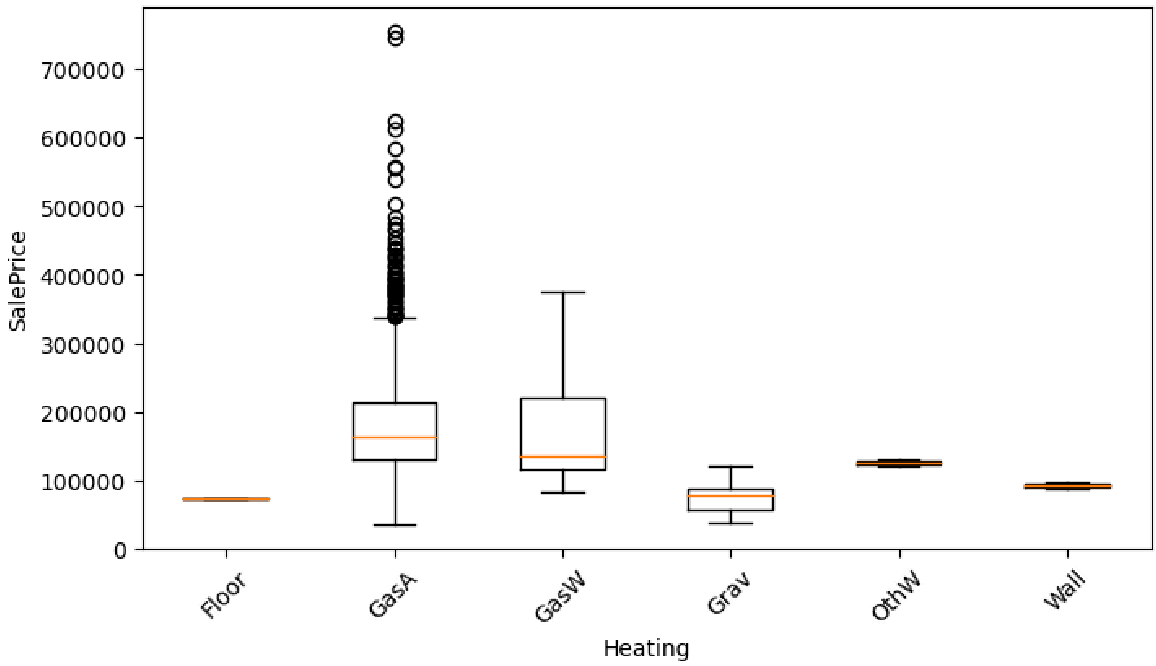

Box Plot of SalePrice by MasVnrType
SalePrice
300000
400000
500000
600000
700000



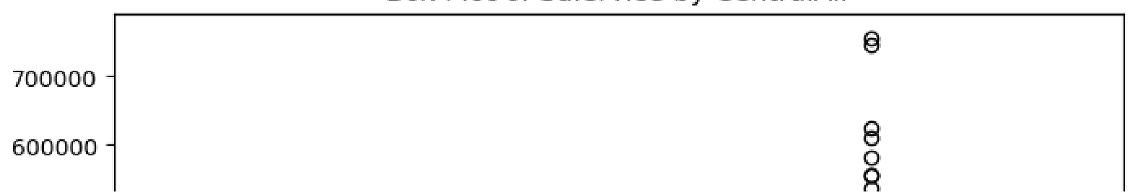
Box Plot of SalePrice by Foundation

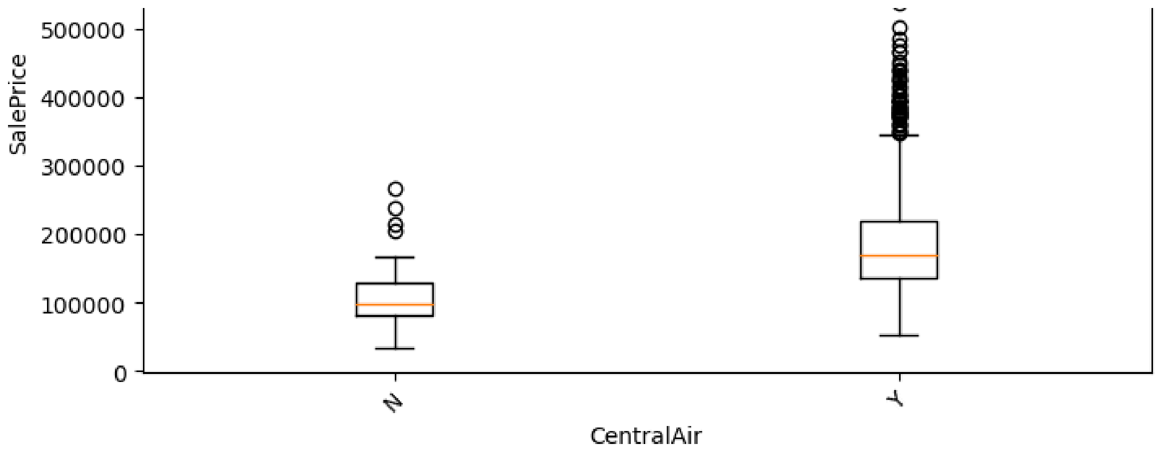


Box Plot of SalePrice by Heating

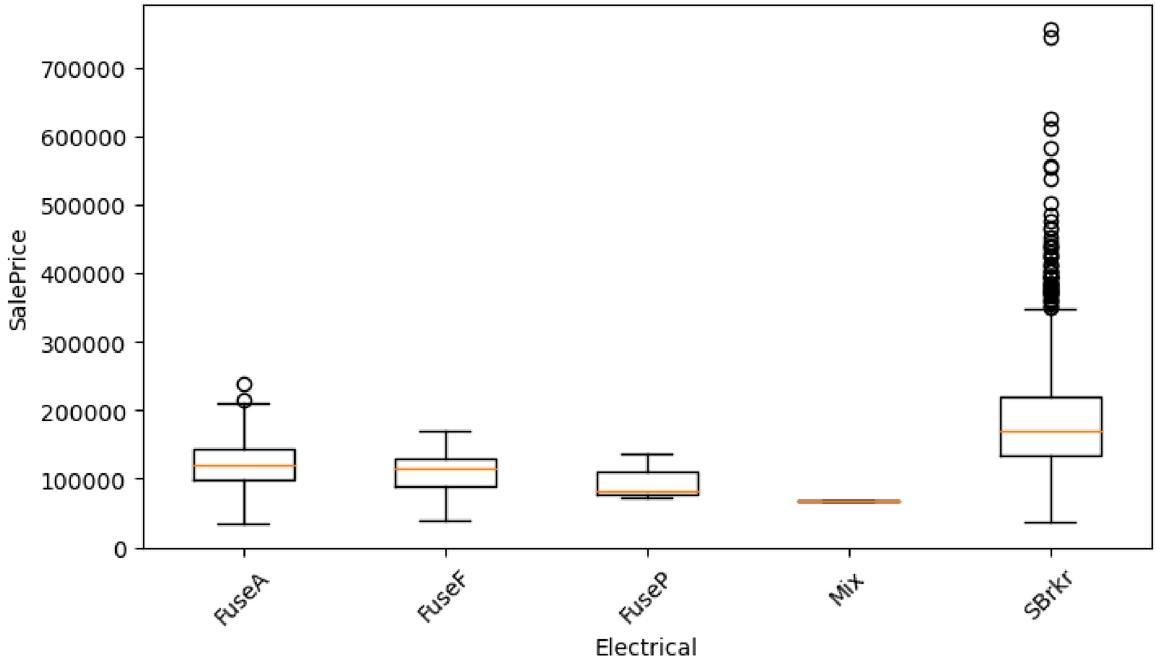


Box Plot of SalePrice by CentralAir

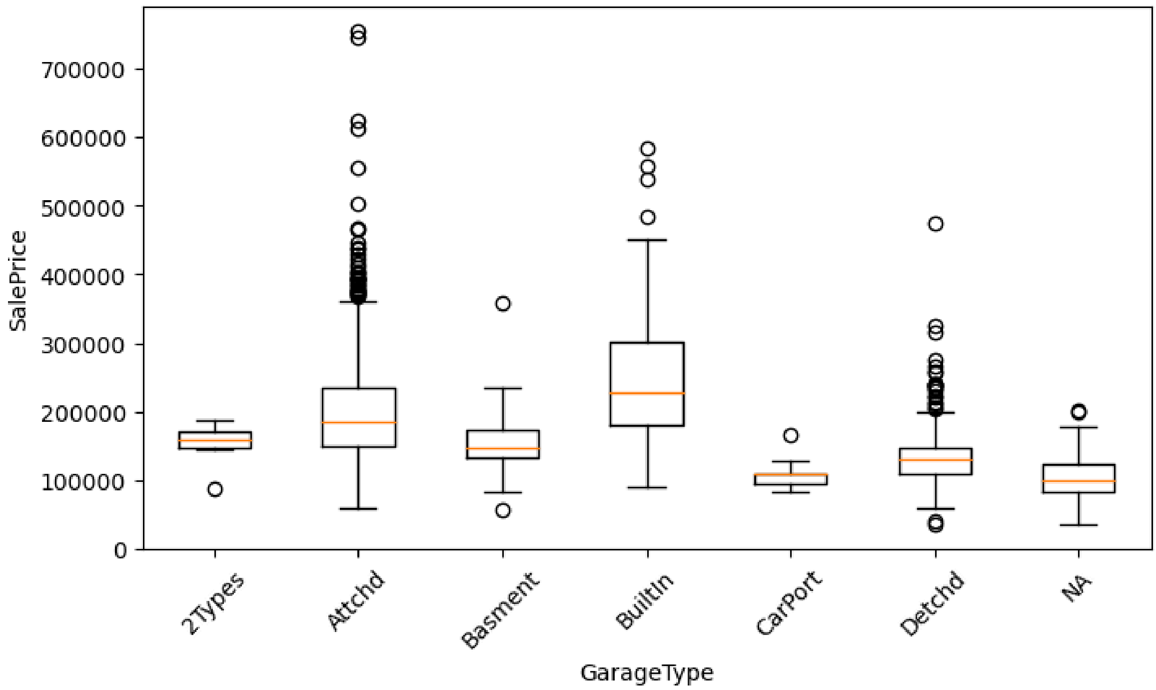




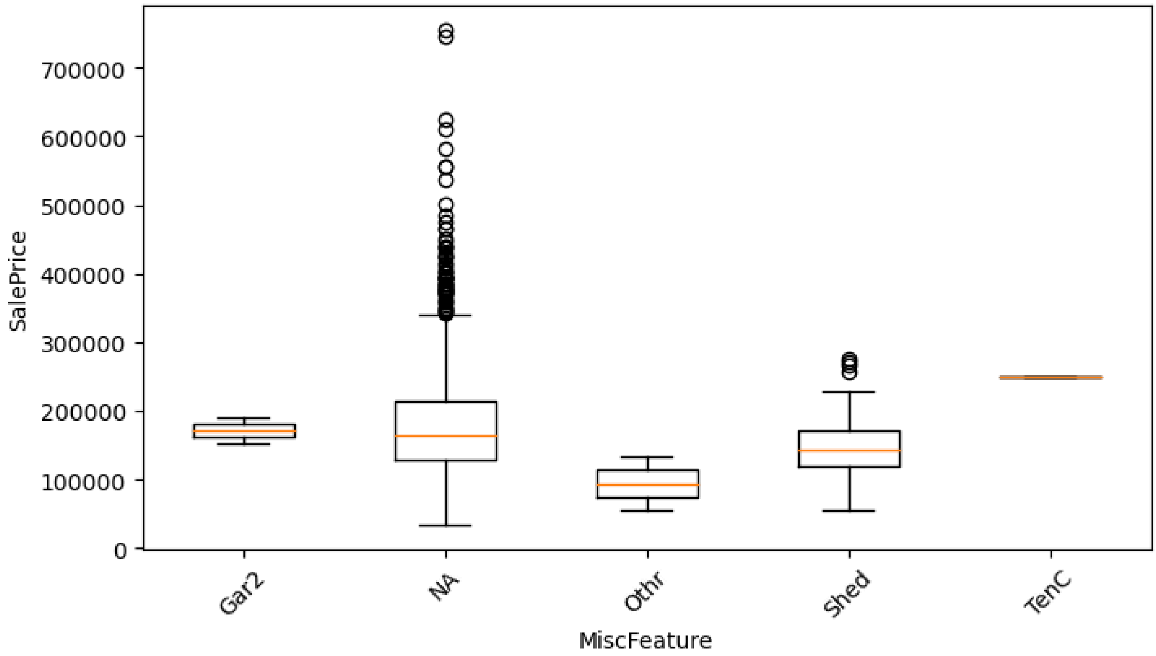
Box Plot of SalePrice by Electrical

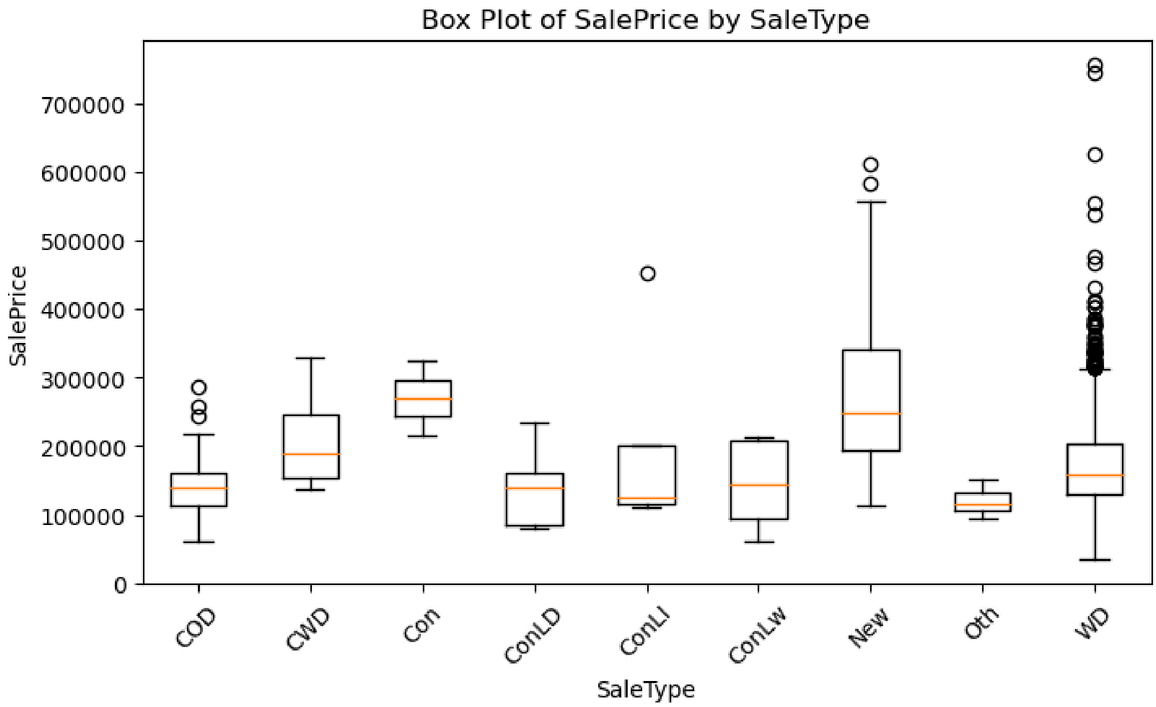


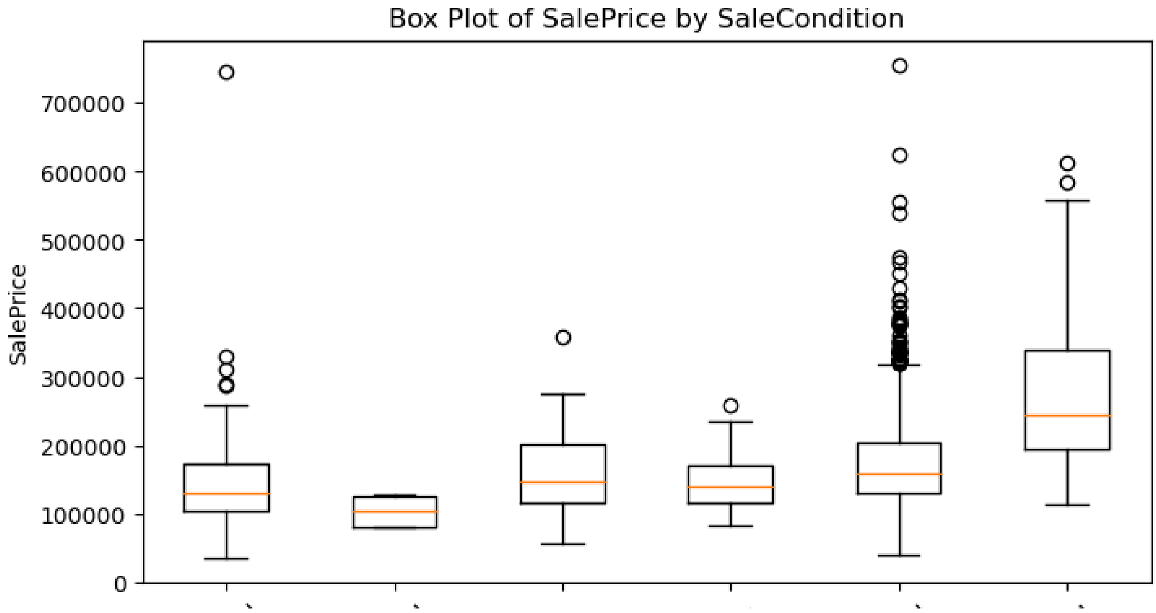
Box Plot of SalePrice by GarageType



Box Plot of SalePrice by MiscFeature


SalePrice
0
100000
200000
300000
400000
500000
600000
700000
Gar2
NA
Othr
Shed
TenC
MiscFeature


Box Plot of SalePrice by SaleType
SalePrice
0
100000
200000
300000
400000
500000
600000
700000
COD
CWD
Con
ConLD
ConLI
ConLw
New
Oth
WD
SaleType


Box Plot of SalePrice by SaleCondition
SalePrice
0
100000
200000
300000
400000
500000
600000
700000

Model I: Biuld Classic Linear Regression Model

In [101…

```
# Train data
train=data
train.info()
# print(train.head(4))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 100 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        1460 non-null   int64
 4   LotArea            1460 non-null   int64
 5   Street             1460 non-null   object
 6   Alley              1460 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         1460 non-null   object
 26  MasVnrArea         1460 non-null   int64
 27  ExterQual          1460 non-null   object
 28  ExterCond          1460 non-null   object
 29  Foundation         1460 non-null   object
 30  BsmtQual           1460 non-null   object
 31  BsmtCond           1460 non-null   object
 32  BsmtExposure       1460 non-null   object
 33  BsmtFinType1       1460 non-null   object
 34  BsmtFinSF1         1460 non-null   int64
 35  BsmtFinType2       1460 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         1460 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        1460 non-null   object
 58  GarageType         1460 non-null   object
```

```
 59  GarageYrBlt        1460 non-null   int64
 60  GarageFinish       1460 non-null   object
 61  GarageCars         1460 non-null   int64
 62  GarageArea         1460 non-null   int64
 63  GarageQual         1460 non-null   object
 64  GarageCond         1460 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             1460 non-null   object
 73  Fence              1460 non-null   object
 74  MiscFeature        1460 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
 81  cnvrt_LotShape     1460 non-null   int64
 82  cnvrt_LandSlope    1460 non-null   int64
 83  cnvrt_ExterQual    1460 non-null   int64
 84  cnvrt_ExterCond    1460 non-null   int64
 85  cnvrt_BsmtQual     1460 non-null   int64
 86  cnvrt_BsmtCond     1460 non-null   int64
 87  cnvrt_BsmtExposure 1460 non-null   int64
 88  cnvrt_BsmtFinType1 1460 non-null   int64
 89  cnvrt_BsmtFinType2 1460 non-null   int64
 90  cnvrt_HeatingQC    1460 non-null   int64
 91  cnvrt_KitchenQual  1460 non-null   int64
 92  cnvrt_Functional   1460 non-null   int64
 93  cnvrt_FireplaceQu  1460 non-null   int64
 94  cnvrt_GarageFinish 1460 non-null   int64
 95  cnvrt_GarageQual   1460 non-null   int64
 96  cnvrt_GarageCond   1460 non-null   int64
 97  cnvrt_PavedDrive   1460 non-null   int64
 98  cnvrt_PoolQC       1460 non-null   int64
 99  cnvrt_Fence        1460 non-null   int64
dtypes: int64(57), object(43)
memory usage: 1.1+ MB
```

In [101…

```
# Convert 'MSSubClass' to string type
train['MSSubClass']=train['MSSubClass'].astype(str)
```

In [102…

```
#Split Train data into test and train for validation (named trainv & testv)
from sklearn.model_selection import train_test_split
trainv, testv = train_test_split(data, test_size = 0.3, random_state = 1234)
print(trainv.shape)
print(testv.shape)
```

```
(1022, 100)
(438, 100)
```

In [102…

```
#Create dummy variables for categorical variables
dummy_vars_trv=pd.get_dummies(trainv[['MSSubClass','MSZoning','Street','Alley','Lar
                                      'Utilities','LotConfig','Neighborhood','Conditic
                                      'BldgType','HouseStyle','RoofStyle','Heating','F
                                      'Exterior1st','Exterior2nd','MasVnrType','Founda
                                      'Electrical','GarageType','MiscFeature','SaleTyp


# dummy_vars containing only 0 and 1
```

```
dummy_vars_trv=dummy_vars_trv.astype(int)

print(dummy_vars_trv.head(2))
dummy_vars_trv.info()
```

```
      MSSubClass_120  MSSubClass_160  MSSubClass_180  MSSubClass_190  \
1017               1               0               0               0
405                0               0               0               0

      MSSubClass_20  MSSubClass_30  MSSubClass_40  MSSubClass_45  \
1017              0              0              0              0
405               1              0              0              0

      MSSubClass_50  MSSubClass_60  ...  SaleType_ConLw  SaleType_New  \
1017              0              0  ...               0             0
405               0              0  ...               0             0

      SaleType_Oth  SaleType_WD  SaleCondition_Abnorml  SaleCondition_AdjLand  \
1017             0            0                      1                      0
405              0            1                      0                      0

      SaleCondition_Alloca  SaleCondition_Family  SaleCondition_Normal  \
1017                     0                     0                     0
405                      0                     0                     1

      SaleCondition_Partial
1017                      0
405                       0

[2 rows x 182 columns]
<class 'pandas.core.frame.DataFrame'>
Index: 1022 entries, 1017 to 815
Columns: 182 entries, MSSubClass_120 to SaleCondition_Partial
dtypes: int32(182)
memory usage: 734.6 KB
```

In [102…

```
dummy_vars_trv.shape
```

Out[1022]:

```
(1022, 182)
```

In [102…

```
# Define base level for each categorical variable According to frequency percentage
dummy_vars_trv.drop(columns = ['MSSubClass_20','MSZoning_RL','Street_Pave','Alley_N
                               'LotConfig_Inside', 'Neighborhood_NAmes', 'Condition1_No
                               'HouseStyle_1Story','RoofStyle_Gable','RoofMatl_CompShg'
                               'MasVnrType_none','Foundation_PConc','CentralAir_Y','Hea
                               'GarageType_Attchd','SaleType_WD','SaleCondition_Normal'
dummy_vars_trv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1022 entries, 1017 to 815
Columns: 157 entries, MSSubClass_120 to SaleCondition_Partial
dtypes: int32(157)
memory usage: 634.8 KB
```

In [102…

```
print(dummy_vars_trv.shape)
print(trainv.shape)
```

```
(1022, 157)
(1022, 100)
```

In [102…

```
dummy_vars_trv
```

Out[1025]:

	MSSubClass_120	MSSubClass_160	MSSubClass_180	MSSubClass_190	MSSubClass_30	MSS
1017	1	0	0	0	0	
405	0	0	0	0	0	
6	0	0	0	0	0	
388	0	0	0	0	0	
501	0	0	0	0	0	
...	...	...	...	...	...	
1228	1	0	0	0	0	
1077	0	0	0	0	0	
1318	0	0	0	0	0	
723	0	0	0	0	0	
815	0	0	0	0	0	

1022 rows × 157 columns

In [102…

```
#Define feature matrix
# Train All columns except 'SalePrice'
X_X=trainv.iloc[:,list(range(0,80))+list(range(81,100))]
print(X_X.columns)
X_trainv=pd.concat([X_X,dummy_vars_trv], axis = 1)
# X_trainv.info()
# #add constant
X_trainv=sm.add_constant(X_trainv)
X_trainv.head(2)
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'cnvrt_LotShape', 'cnvrt_LandSlope', 'cnvrt_ExterQual',
       'cnvrt_ExterCond', 'cnvrt_BsmtQual', 'cnvrt_BsmtCond',
       'cnvrt_BsmtExposure', 'cnvrt_BsmtFinType1', 'cnvrt_BsmtFinType2',
       'cnvrt_HeatingQC', 'cnvrt_KitchenQual', 'cnvrt_Functional',
       'cnvrt_FireplaceQu', 'cnvrt_GarageFinish', 'cnvrt_GarageQual',
       'cnvrt_GarageCond', 'cnvrt_PavedDrive', 'cnvrt_PoolQC', 'cnvrt_Fence'],
      dtype='object')
```

Out[1026]:

	const	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	Land
1017	1.0	1018	120	RL	0	5814	Pave	NA	IR1	
405	1.0	406	20	RL	0	9991	Pave	NA	IR1	

2 rows × 257 columns

In [102…

```
X_trainv.shape
```

Out[1027]:

```
(1022, 257)
```

In [102…

```
# Remove coloumn'Id'
X_trainv.drop(columns=['Id'],inplace=True)
print(X_trainv.shape)
X_trainv.head()
```

```
(1022, 256)
```

Out[1028]:

	const	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
1017	1.0	120	RL	0	5814	Pave	NA	IR1	L
405	1.0	20	RL	0	9991	Pave	NA	IR1	L
6	1.0	20	RL	75	10084	Pave	NA	Reg	L
388	1.0	20	RL	93	9382	Pave	NA	IR1	L
501	1.0	60	FV	75	9803	Pave	NA	Reg	L

5 rows × 256 columns

In [102…

```
X_trainv.shape
```

Out[1029]:

```
(1022, 256)
```

In [103…

```
# Define response matrix
y_trainv=trainv['SalePrice']
print(len(y_trainv))
y_trainv
```

```
1022
```

Out[1030]:

```
1017    187500
405     150000
6       307000
388     191000
501     226700
         ...
1228    367294
1077    138800
1318    275000
723     135000
815     224900
Name: SalePrice, Length: 1022, dtype: int64
```

In [103…

```
X_trainv=X_trainv.drop(columns=X_trainv.select_dtypes(include=['object']).columns)
# Check the data types of the cleaned DataFrame
print(X_trainv.dtypes)
X_trainv.info()
```

```
const                        float64
LotFrontage                    int64
LotArea                        int64
OverallQual                    int64
OverallCond                    int64
                              ...
SaleCondition_Abnorml          int32
SaleCondition_AdjLand          int32
SaleCondition_Alloca           int32
SaleCondition_Family           int32
SaleCondition_Partial          int32
Length: 212, dtype: object
<class 'pandas.core.frame.DataFrame'>
Index: 1022 entries, 1017 to 815
Columns: 212 entries, const to SaleCondition_Partial
dtypes: float64(1), int32(157), int64(54)
memory usage: 1.0 MB
```

In [103…

```
# Convert float64 and int32 columns to int64
X_trainv=X_trainv.astype({col:'int64' for col in X_trainv.select_dtypes(include=['f

# Check the data types of the columns to verify the conversion
print(X_trainv.dtypes)
```

```
const                        int64
LotFrontage                  int64
LotArea                      int64
OverallQual                  int64
OverallCond                  int64
                             ...
SaleCondition_Abnorml        int64
SaleCondition_AdjLand        int64
SaleCondition_Alloca         int64
SaleCondition_Family         int64
SaleCondition_Partial        int64
Length: 212, dtype: object
```

In [103…

```
X_trainv.shape
```

Out[1033]:

```
(1022, 212)
```

In [103…

```
#Linear regression- model 1
model_1=sm.OLS(y_trainv,X_trainv).fit()
model_1.summary()
```

Out[1034]:

OLS Regression Results

Dep. Variable:	SalePrice	**R-squared:**	0.924
Model:	OLS	**Adj. R-squared:**	0.904
Method:	Least Squares	**F-statistic:**	47.83
Date:	Fri, 31 Jan 2025	**Prob (F-statistic):**	0.00
Time:	22:33:15	**Log-Likelihood:**	-11710.
No. Observations:	1022	**AIC:**	2.383e+04
Df Residuals:	815	**BIC:**	2.485e+04
Df Model:	206		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	-4.244e+05	1.45e+06	-0.292	0.770	-3.28e+06	2.43e+06
LotFrontage	30.9974	31.021	0.999	0.318	-29.893	91.888
LotArea	0.4095	0.144	2.848	0.005	0.127	0.692
OverallQual	8137.3197	1414.696	5.752	0.000	5360.442	1.09e+04
OverallCond	5217.8559	1177.711	4.431	0.000	2906.151	7529.561
YearBuilt	231.4660	116.936	1.979	0.048	1.935	460.998
YearRemodAdd	6.1239	78.115	0.078	0.938	-147.207	159.455
MasVnrArea	29.2115	7.700	3.793	0.000	14.097	44.326
BsmtFinSF1	16.4604	3.590	4.585	0.000	9.413	23.508
BsmtFinSF2	9.0956	6.716	1.354	0.176	-4.087	22.278
BsmtUnfSF	-4.1291	3.204	-1.289	0.198	-10.419	2.160
TotalBsmtSF	21.4268	4.667	4.591	0.000	12.266	30.588
1stFlrSF	17.3976	8.434	2.063	0.039	0.844	33.952
2ndFlrSF	38.3211	7.824	4.898	0.000	22.963	53.679
LowQualFinSF	-32.9508	19.362	-1.702	0.089	-70.956	5.054
GrLivArea	22.7679	7.974	2.855	0.004	7.116	38.420
BsmtFullBath	248.5334	2787.416	0.089	0.929	-5222.828	5719.894
BsmtHalfBath	-2624.7522	4274.509	-0.614	0.539	-1.1e+04	5765.593
FullBath	2011.9828	3049.787	0.660	0.510	-3974.380	7998.345
HalfBath	898.0215	2926.238	0.307	0.759	-4845.829	6641.872
BedroomAbvGr	-5757.9152	1897.734	-3.034	0.002	-9482.937	-2032.893
KitchenAbvGr	-1.42e+04	9338.639	-1.520	0.129	-3.25e+04	4135.342
TotRmsAbvGrd	4408.5240	1295.386	3.403	0.001	1865.837	6951.211
Fireplaces	8041.6138	3126.828	2.572	0.010	1904.029	1.42e+04
GarageYrBlt	43.6328	82.728	0.527	0.598	-118.753	206.019
GarageCars	2203.6150	3124.466	0.705	0.481	-3929.333	8336.563

GarageArea	25.1661	10.557	2.384	0.017	4.443	45.889
WoodDeckSF	9.5207	7.821	1.217	0.224	-5.832	24.873
OpenPorchSF	11.6915	15.505	0.754	0.451	-18.742	42.125
EnclosedPorch	-18.1047	16.709	-1.084	0.279	-50.902	14.692
3SsnPorch	41.9458	30.440	1.378	0.169	-17.803	101.695
ScreenPorch	55.9820	16.326	3.429	0.001	23.936	88.028
PoolArea	-5.8170	53.739	-0.108	0.914	-111.301	99.667
MiscVal	1.5750	7.496	0.210	0.834	-13.139	16.289
MoSold	-608.1725	332.354	-1.830	0.068	-1260.543	44.198
YrSold	-136.5629	705.372	-0.194	0.847	-1521.124	1247.998
cnvrt_LotShape	-1037.1232	1916.672	-0.541	0.589	-4799.318	2725.072
cnvrt_LandSlope	83.0474	4629.713	0.018	0.986	-9004.519	9170.613
cnvrt_ExterQual	4101.2613	2934.089	1.398	0.163	-1658.001	9860.524
cnvrt_ExterCond	-3237.8501	2881.164	-1.124	0.261	-8893.227	2417.527
cnvrt_BsmtQual	5258.2567	2415.377	2.177	0.030	517.164	9999.350
cnvrt_BsmtCond	-5138.8116	3072.563	-1.672	0.095	-1.12e+04	892.258
cnvrt_BsmtExposure	5412.4052	1146.410	4.721	0.000	3162.141	7662.669
cnvrt_BsmtFinType1	286.3164	699.323	0.409	0.682	-1086.369	1659.002
cnvrt_BsmtFinType2	-437.7369	1663.466	-0.263	0.793	-3702.918	2827.445
cnvrt_HeatingQC	1529.4951	1313.893	1.164	0.245	-1049.517	4108.507
cnvrt_KitchenQual	6966.5623	2303.960	3.024	0.003	2444.168	1.15e+04
cnvrt_Functional	7498.8255	1683.017	4.456	0.000	4195.267	1.08e+04
cnvrt_FireplaceQu	-1844.7722	1157.794	-1.593	0.111	-4117.382	427.837
cnvrt_GarageFinish	174.3452	1633.436	0.107	0.915	-3031.892	3380.582
cnvrt_GarageQual	6390.5274	5060.584	1.263	0.207	-3542.787	1.63e+04
cnvrt_GarageCond	-8458.8973	5378.206	-1.573	0.116	-1.9e+04	2097.870
cnvrt_PavedDrive	-1188.3134	2309.442	-0.515	0.607	-5721.469	3344.842
cnvrt_PoolQC	3.064e+04	1.07e+04	2.867	0.004	9662.781	5.16e+04
cnvrt_Fence	-179.5060	847.944	-0.212	0.832	-1843.917	1484.905
MSSubClass_120	-2.251e+04	2e+04	-1.127	0.260	-6.17e+04	1.67e+04
MSSubClass_160	-1.366e+04	2.37e+04	-0.576	0.565	-6.03e+04	3.29e+04
MSSubClass_180	-1.816e+04	2.69e+04	-0.675	0.500	-7.1e+04	3.47e+04
MSSubClass_190	-4.63e+04	3.02e+04	-1.533	0.126	-1.06e+05	1.3e+04
MSSubClass_30	-3191.6131	6536.078	-0.488	0.625	-1.6e+04	9637.918
MSSubClass_40	-624.9925	2.53e+04	-0.025	0.980	-5.03e+04	4.9e+04
MSSubClass_45	-1.142e+04	2.69e+04	-0.425	0.671	-6.42e+04	4.13e+04
MSSubClass_50	-8932.8537	1.32e+04	-0.677	0.499	-3.48e+04	1.7e+04

MSSubClass_60	-4989.6995	1.03e+04	-0.485	0.627	-2.52e+04	1.52e+04
MSSubClass_70	-6532.6236	1.14e+04	-0.574	0.566	-2.89e+04	1.58e+04
MSSubClass_75	-3.344e+04	2e+04	-1.673	0.095	-7.27e+04	5791.477
MSSubClass_80	-8782.8741	1.58e+04	-0.554	0.580	-3.99e+04	2.23e+04
MSSubClass_85	-4553.6327	1.65e+04	-0.276	0.783	-3.7e+04	2.79e+04
MSSubClass_90	-4977.9353	5360.789	-0.929	0.353	-1.55e+04	5544.645
MSZoning_C (all)	-2.824e+04	1.42e+04	-1.993	0.047	-5.61e+04	-426.714
MSZoning_FV	1.097e+04	9292.990	1.180	0.238	-7270.644	2.92e+04
MSZoning_RH	8092.3510	1.03e+04	0.788	0.431	-1.21e+04	2.83e+04
MSZoning_RM	3560.1245	5140.418	0.693	0.489	-6529.893	1.37e+04
Street_Grvl	-2.617e+04	1.45e+04	-1.806	0.071	-5.46e+04	2277.445
Alley_Grvl	-3012.0735	5917.407	-0.509	0.611	-1.46e+04	8603.080
Alley_Pave	1378.6031	6887.950	0.200	0.841	-1.21e+04	1.49e+04
LandContour_Bnk	-1.158e+04	5594.111	-2.071	0.039	-2.26e+04	-603.976
LandContour_HLS	2115.3830	5503.382	0.384	0.701	-8687.089	1.29e+04
LandContour_Low	-1.372e+04	7211.122	-1.903	0.057	-2.79e+04	433.215
Utilities_NoSeWa	-6.382e+04	3.43e+04	-1.862	0.063	-1.31e+05	3467.196
LotConfig_Corner	1488.6619	2435.811	0.611	0.541	-3292.540	6269.864
LotConfig_CulDSac	1.481e+04	4324.134	3.425	0.001	6324.562	2.33e+04
LotConfig_FR2	-4482.4065	5296.092	-0.846	0.398	-1.49e+04	5913.181
LotConfig_FR3	-1.846e+04	1.44e+04	-1.285	0.199	-4.67e+04	9741.109
Neighborhood_Blmngtn	2.225e+04	1.06e+04	2.098	0.036	1430.473	4.31e+04
Neighborhood_Blueste	1.982e+04	2.93e+04	0.676	0.499	-3.77e+04	7.74e+04
Neighborhood_BrDale	1.372e+04	1.24e+04	1.109	0.268	-1.06e+04	3.8e+04
Neighborhood_BrkSide	1.088e+04	7064.938	1.540	0.124	-2988.870	2.47e+04
Neighborhood_ClearCr	1.033e+04	8656.309	1.193	0.233	-6664.276	2.73e+04
Neighborhood_CollgCr	1812.4929	5116.623	0.354	0.723	-8230.818	1.19e+04
Neighborhood_Crawfor	2.394e+04	6786.785	3.527	0.000	1.06e+04	3.73e+04
Neighborhood_Edwards	-2121.5937	4745.863	-0.447	0.655	-1.14e+04	7193.961
Neighborhood_Gilbert	2596.4765	6692.118	0.388	0.698	-1.05e+04	1.57e+04
Neighborhood_IDOTRR	4904.3165	9627.535	0.509	0.611	-1.4e+04	2.38e+04
Neighborhood_MeadowV	-1131.5750	1.27e+04	-0.089	0.929	-2.61e+04	2.38e+04
Neighborhood_Mitchel	-7383.8782	6541.595	-1.129	0.259	-2.02e+04	5456.482
Neighborhood_NPkVill	3.519e+04	1.54e+04	2.288	0.022	4998.320	6.54e+04
Neighborhood_NWAmes	-3567.1427	5363.017	-0.665	0.506	-1.41e+04	6959.810
Neighborhood_NoRidge	4.291e+04	7892.445	5.437	0.000	2.74e+04	5.84e+04
Neighborhood_NridgHt	4.679e+04	6775.490	6.906	0.000	3.35e+04	6.01e+04

Neighborhood_OldTown	2078.5037	7085.453	0.293	0.769	-1.18e+04	1.6e+04
Neighborhood_SWISU	9777.5104	8898.715	1.099	0.272	-7689.590	2.72e+04
Neighborhood_Sawyer	7110.5399	4537.854	1.567	0.118	-1796.719	1.6e+04
Neighborhood_SawyerW	8887.8779	6334.532	1.403	0.161	-3546.042	2.13e+04
Neighborhood_Somerst	3715.8111	9166.779	0.405	0.685	-1.43e+04	2.17e+04
Neighborhood_StoneBr	5.605e+04	9388.323	5.970	0.000	3.76e+04	7.45e+04
Neighborhood_Timber	1063.7499	6994.876	0.152	0.879	-1.27e+04	1.48e+04
Neighborhood_Veenker	1.722e+04	1.13e+04	1.523	0.128	-4972.216	3.94e+04
Condition1_Artery	-1.265e+04	5821.895	-2.173	0.030	-2.41e+04	-1220.476
Condition1_Feedr	-1.092e+04	4155.401	-2.627	0.009	-1.91e+04	-2760.024
Condition1_PosA	1.508e+04	1.38e+04	1.097	0.273	-1.19e+04	4.21e+04
Condition1_PosN	-5972.7109	8793.184	-0.679	0.497	-2.32e+04	1.13e+04
Condition1_RRAe	-2.369e+04	9862.930	-2.402	0.017	-4.31e+04	-4332.305
Condition1_RRAn	2100.0346	7285.720	0.288	0.773	-1.22e+04	1.64e+04
Condition1_RRNe	-8206.9738	2.66e+04	-0.309	0.758	-6.04e+04	4.4e+04
Condition1_RRNn	-4326.4871	1.42e+04	-0.305	0.761	-3.22e+04	2.35e+04
Condition2_Artery	2.091e+04	2.66e+04	0.785	0.433	-3.14e+04	7.32e+04
Condition2_Feedr	-1230.4377	1.55e+04	-0.079	0.937	-3.17e+04	2.92e+04
Condition2_PosA	5.803e+04	3.25e+04	1.787	0.074	-5718.202	1.22e+05
Condition2_PosN	-2.135e+05	2.27e+04	-9.394	0.000	-2.58e+05	-1.69e+05
Condition2_RRAe	-1.088e+04	3.35e+04	-0.325	0.745	-7.67e+04	5.49e+04
Condition2_RRNn	2.098e+04	2.84e+04	0.740	0.460	-3.47e+04	7.66e+04
BldgType_2fmCon	2.475e+04	2.88e+04	0.861	0.390	-3.17e+04	8.12e+04
BldgType_Duplex	-4977.9353	5360.789	-0.929	0.353	-1.55e+04	5544.645
BldgType_Twnhs	-1.936e+04	2.13e+04	-0.910	0.363	-6.11e+04	2.24e+04
BldgType_TwnhsE	-1.231e+04	2.01e+04	-0.613	0.540	-5.17e+04	2.71e+04
HouseStyle_1.5Fin	-766.6887	1.32e+04	-0.058	0.954	-2.68e+04	2.52e+04
HouseStyle_1.5Unf	6004.5859	2.62e+04	0.230	0.818	-4.53e+04	5.73e+04
HouseStyle_2.5Fin	1.527e+04	2.42e+04	0.630	0.529	-3.23e+04	6.28e+04
HouseStyle_2.5Unf	1783.8526	2.03e+04	0.088	0.930	-3.81e+04	4.17e+04
HouseStyle_2Story	-1.335e+04	1.14e+04	-1.171	0.242	-3.57e+04	9029.652
HouseStyle_SFoyer	-8020.3959	1.44e+04	-0.557	0.577	-3.63e+04	2.02e+04
HouseStyle_SLvl	1806.8809	1.56e+04	0.116	0.908	-2.87e+04	3.23e+04
RoofStyle_Flat	250.7998	2.17e+04	0.012	0.991	-4.24e+04	4.29e+04
RoofStyle_Gambrel	5461.8392	1.18e+04	0.463	0.643	-1.77e+04	2.86e+04
RoofStyle_Hip	3637.3008	2479.281	1.467	0.143	-1229.227	8503.829
RoofStyle_Mansard	1.568e+04	1.38e+04	1.138	0.255	-1.14e+04	4.27e+04

RoofStyle_Shed	-1.088e+04	3.35e+04	-0.325	0.745	-7.67e+04	5.49e+04
Heating_Floor	9008.9316	2.96e+04	0.304	0.761	-4.91e+04	6.72e+04
Heating_GasW	-4804.1284	1.1e+04	-0.438	0.662	-2.63e+04	1.67e+04
Heating_Grav	-1071.7788	1.64e+04	-0.065	0.948	-3.32e+04	3.11e+04
Heating_OthW	-3.398e+04	2.22e+04	-1.527	0.127	-7.76e+04	9688.311
Heating_Wall	2.016e+04	1.85e+04	1.087	0.277	-1.62e+04	5.66e+04
RoofMatl_ClyTile	-5.786e+05	4.84e+04	-11.966	0.000	-6.74e+05	-4.84e+05
RoofMatl_Membran	3.989e+04	3.44e+04	1.158	0.247	-2.77e+04	1.08e+05
RoofMatl_Metal	4711.6781	3.56e+04	0.132	0.895	-6.51e+04	7.46e+04
RoofMatl_Tar&Grv	-4813.7120	2.17e+04	-0.222	0.825	-4.74e+04	3.78e+04
RoofMatl_WdShake	-5.405e+04	2.73e+04	-1.982	0.048	-1.08e+05	-534.445
RoofMatl_WdShngl	7.649e+04	1.26e+04	6.088	0.000	5.18e+04	1.01e+05
Exterior1st_AsbShng	1.348e+04	1.74e+04	0.776	0.438	-2.06e+04	4.76e+04
Exterior1st_BrkComm	-3251.6649	3.81e+04	-0.085	0.932	-7.8e+04	7.15e+04
Exterior1st_BrkFace	1.696e+04	1.12e+04	1.514	0.130	-5023.691	3.89e+04
Exterior1st_CBlock	1971.6700	1.42e+04	0.139	0.890	-2.59e+04	2.98e+04
Exterior1st_CemntBd	5725.7455	2.34e+04	0.245	0.807	-4.02e+04	5.17e+04
Exterior1st_HdBoard	-4644.3440	1.08e+04	-0.432	0.666	-2.58e+04	1.65e+04
Exterior1st_ImStucc	-4.763e+04	3.08e+04	-1.545	0.123	-1.08e+05	1.29e+04
Exterior1st_MetalSd	9111.5404	1.46e+04	0.624	0.533	-1.96e+04	3.78e+04
Exterior1st_Plywood	-8011.7539	1.07e+04	-0.750	0.453	-2.9e+04	1.29e+04
Exterior1st_Stone	3.417e+04	2.95e+04	1.157	0.247	-2.38e+04	9.21e+04
Exterior1st_Stucco	6191.1002	1.56e+04	0.398	0.691	-2.44e+04	3.68e+04
Exterior1st_Wd Sdng	-2642.9020	1.01e+04	-0.262	0.793	-2.24e+04	1.71e+04
Exterior1st_WdShing	8032.8534	1.21e+04	0.663	0.507	-1.57e+04	3.18e+04
Exterior2nd_AsbShng	-1.083e+04	1.7e+04	-0.638	0.524	-4.42e+04	2.25e+04
Exterior2nd_AsphShn	5254.1457	2.4e+04	0.219	0.827	-4.19e+04	5.24e+04
Exterior2nd_Brk Cmn	-1.146e+04	2.6e+04	-0.440	0.660	-6.25e+04	3.96e+04
Exterior2nd_BrkFace	-5178.8772	1.29e+04	-0.401	0.689	-3.05e+04	2.02e+04
Exterior2nd_CBlock	1971.6700	1.42e+04	0.139	0.890	-2.59e+04	2.98e+04
Exterior2nd_CmentBd	7638.4132	2.33e+04	0.328	0.743	-3.81e+04	5.34e+04
Exterior2nd_HdBoard	3721.5285	1.08e+04	0.346	0.730	-1.74e+04	2.48e+04
Exterior2nd_ImStucc	2.367e+04	1.49e+04	1.594	0.111	-5480.733	5.28e+04
Exterior2nd_MetalSd	-3052.3310	1.49e+04	-0.205	0.838	-3.23e+04	2.62e+04
Exterior2nd_Other	-4.028e+04	2.75e+04	-1.467	0.143	-9.42e+04	1.36e+04
Exterior2nd_Plywood	2691.7141	1.02e+04	0.264	0.792	-1.74e+04	2.27e+04
Exterior2nd_Stone	-2.159e+04	2.18e+04	-0.990	0.323	-6.44e+04	2.12e+04

Exterior2nd_Stucco	-5074.4203	1.51e+04	-0.335	0.737	-3.48e+04	2.46e+04
Exterior2nd_Wd Sdng	7277.0055	1.02e+04	0.715	0.475	-1.27e+04	2.73e+04
Exterior2nd_Wd Shng	-7912.3343	1.09e+04	-0.725	0.469	-2.93e+04	1.35e+04
MasVnrType_BrkCmn	-1.621e+04	8826.706	-1.837	0.067	-3.35e+04	1112.723
MasVnrType_BrkFace	-6885.5112	3051.112	-2.257	0.024	-1.29e+04	-896.548
MasVnrType_Stone	766.3786	4322.355	0.177	0.859	-7717.881	9250.638
Foundation_BrkTil	-9816.5612	4864.043	-2.018	0.044	-1.94e+04	-269.033
Foundation_CBlock	-1374.7359	3263.829	-0.421	0.674	-7781.238	5031.766
Foundation_Slab	1.534e+04	1.3e+04	1.180	0.238	-1.02e+04	4.08e+04
Foundation_Stone	-3706.4812	1.35e+04	-0.274	0.784	-3.03e+04	2.29e+04
Foundation_Wood	-3.398e+04	1.68e+04	-2.017	0.044	-6.7e+04	-919.078
CentralAir_N	4276.1898	5735.418	0.746	0.456	-6981.742	1.55e+04
Electrical_FuseA	-1245.5074	4373.413	-0.285	0.776	-9829.988	7338.973
Electrical_FuseF	264.9285	7938.805	0.033	0.973	-1.53e+04	1.58e+04
Electrical_FuseP	1.207e+04	2.06e+04	0.586	0.558	-2.83e+04	5.25e+04
GarageType_2Types	-604.1757	1.41e+04	-0.043	0.966	-2.82e+04	2.7e+04
GarageType_Basment	1.806e+04	9987.403	1.808	0.071	-1546.361	3.77e+04
GarageType_BuiltIn	3707.5974	4863.287	0.762	0.446	-5838.446	1.33e+04
GarageType_CarPort	7598.4283	1.13e+04	0.673	0.501	-1.46e+04	2.98e+04
GarageType_Detchd	3118.4720	3118.450	1.000	0.318	-3002.669	9239.613
GarageType_NA	9.927e+04	1.57e+05	0.631	0.528	-2.1e+05	4.08e+05
MiscFeature_Gar2	-3.233e+04	1.19e+05	-0.272	0.786	-2.66e+05	2.01e+05
MiscFeature_Othr	1.6e+04	2.98e+04	0.537	0.591	-4.24e+04	7.44e+04
MiscFeature_Shed	87.5304	7979.633	0.011	0.991	-1.56e+04	1.58e+04
MiscFeature_TenC	-8.409e+04	3.75e+04	-2.241	0.025	-1.58e+05	-1.04e+04
SaleType_COD	-560.7315	5638.201	-0.099	0.921	-1.16e+04	1.05e+04
SaleType_CWD	1.389e+04	1.91e+04	0.727	0.467	-2.36e+04	5.14e+04
SaleType_Con	2.753e+04	1.96e+04	1.406	0.160	-1.09e+04	6.6e+04
SaleType_ConLD	3.722e+04	1.7e+04	2.195	0.028	3932.524	7.05e+04
SaleType_ConLI	-4894.1980	1.65e+04	-0.297	0.767	-3.73e+04	2.75e+04
SaleType_ConLw	-8102.4673	2.72e+04	-0.298	0.766	-6.15e+04	4.53e+04
SaleType_New	346.5753	2.74e+04	0.013	0.990	-5.34e+04	5.4e+04
SaleType_Oth	4616.1248	1.66e+04	0.278	0.781	-2.8e+04	3.72e+04
SaleCondition_Abnorml	-5683.5938	4049.331	-1.404	0.161	-1.36e+04	2264.753
SaleCondition_AdjLand	-1.503e+04	2.75e+04	-0.546	0.585	-6.9e+04	3.9e+04
SaleCondition_Alloca	257.2113	1.2e+04	0.021	0.983	-2.33e+04	2.38e+04
SaleCondition_Family	-4007.3717	8470.627	-0.473	0.636	-2.06e+04	1.26e+04

SaleCondition_Partial	1.831e+04	2.7e+04	0.677	0.499	-3.48e+04	7.14e+04

Omnibus:	263.746	**Durbin-Watson:**	1.919
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	5610.209
Skew:	0.637	**Prob(JB):**	0.00
Kurtosis:	14.407	**Cond. No.**	7.65e+16

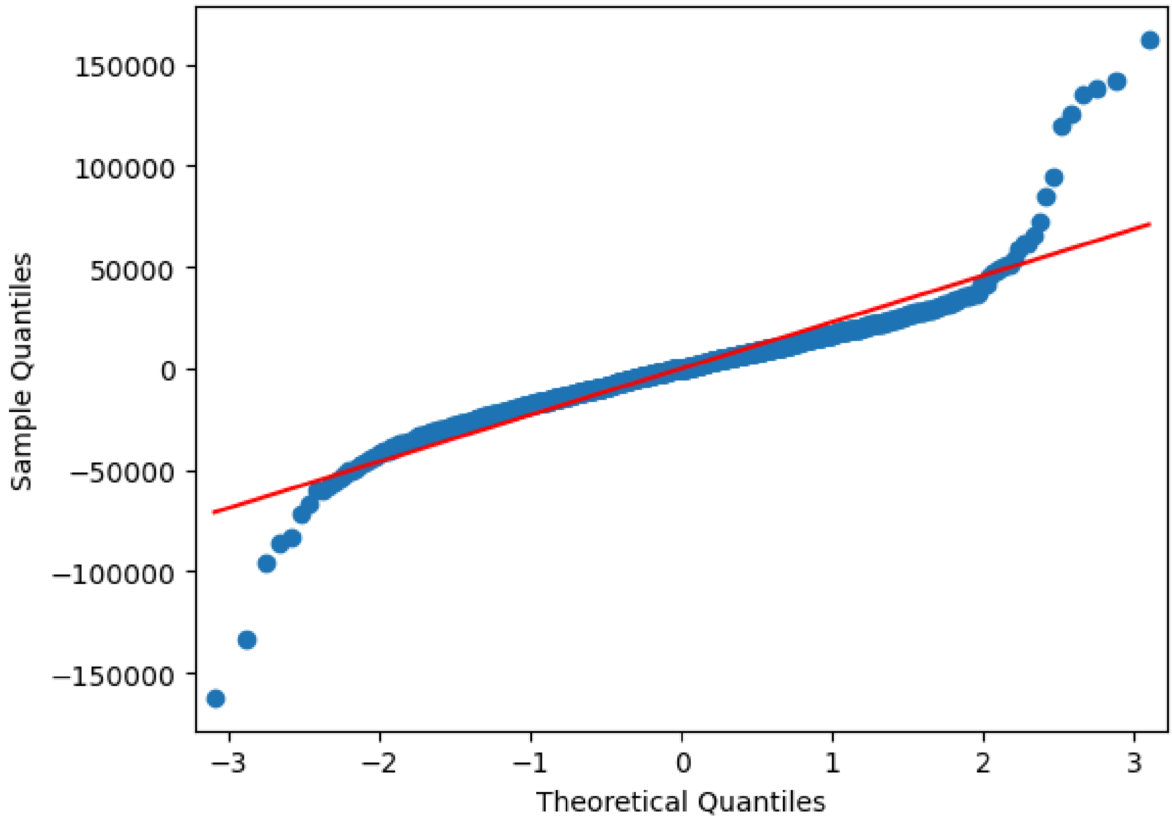
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.83e-23. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

In [103…

```
# Function to plot histogram of residuals
def hist_residuals(model,bins = 50):
    #Calculate density
    from scipy import stats
    density=stats.gaussian_kde(model.resid)
    xp=np.linspace(model.resid.min(),model.resid.max(), 100)
    yp=density.pdf(xp)

    #Histogram
    plt.hist(model.resid,bins=bins,
             color='red',alpha = 0.7,density = True)
    plt.axvline(model.resid.mean(), color='black',
                linewidth=2, linestyle='--', label = "Average")
    plt.title('Histogram of Residuals')
    plt.xlabel('Residuals')
    plt.ylabel('Density')
    plt.plot(xp,yp,color='black',linewidth = 2)
    plt.legend()

    return plt.show()
```

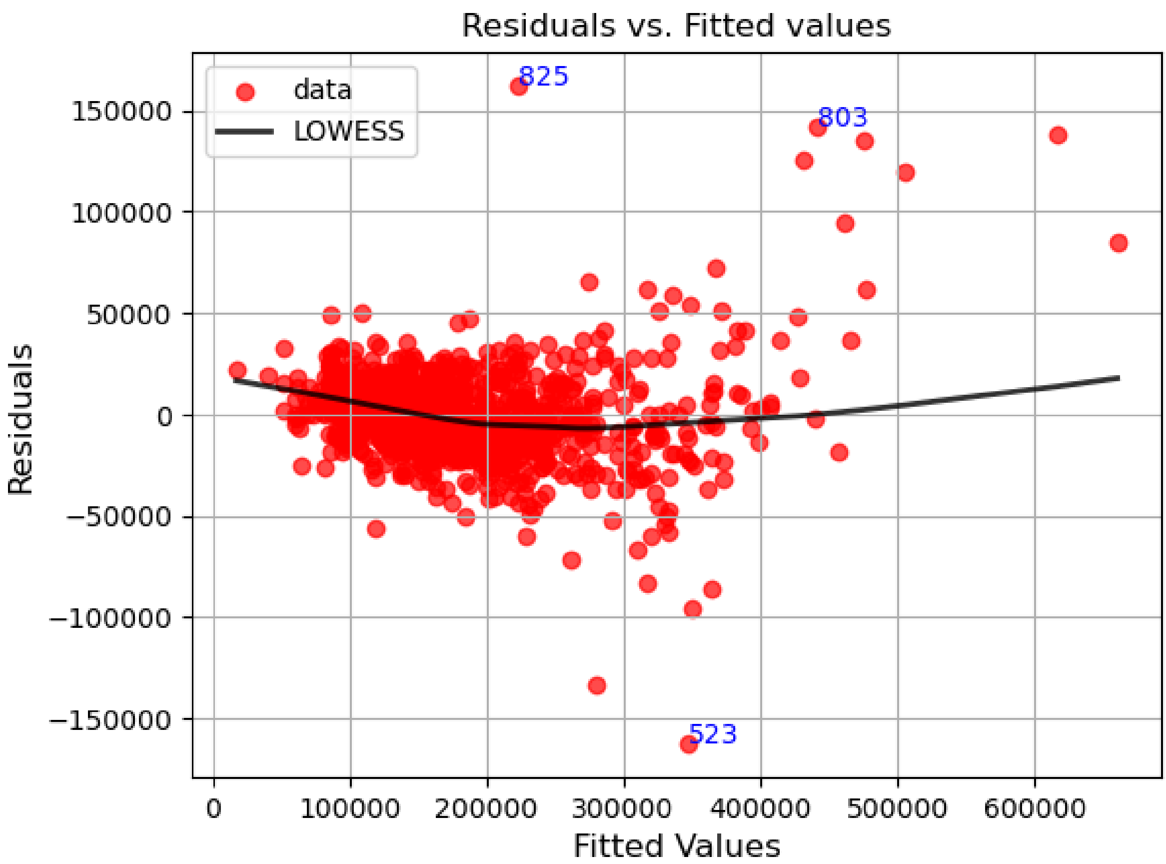
In [103…

```
#QQ-plot- model 1
sm.qqplot(model_1.resid,line = 's')
plt.show()
```



In [103…
```
# Function to plot residuals vs. fitted values
def residuals_fittedvalues_plot(model):
    # Implement Lowess algorithm
    lowess_res=sm.nonparametric.lowess(model.resid, model.fittedvalues)

    # Scatter plot: residuals vs. fitted values
    plt.scatter(x=model.fittedvalues,y = model.resid,
                color='red',alpha = 0.7,label = 'data')
    plt.title('Residuals vs. Fitted values')
    plt.xlabel('Fitted Values',fontsize = 12)
    plt.ylabel('Residuals',fontsize = 12)
    plt.grid()

    # Add LOWESS line
    plt.plot(lowess_res[:, 0],lowess_res[:, 1],'black',
             alpha = 0.8,linewidth = 2,label='LOWESS')
    plt.legend()

    # Top 3 observations with greates absolute value of the residual
    top3=abs(model.resid).sort_values(ascending=False)[:3]
    for i in top3.index:
        plt.annotate(i,xy=(model.fittedvalues[i],model.resid[i]),color='blue')

    return plt.show()
```

In [103…
```
# Scatter plot of residuals vs. fitted values- model 1
residuals_fittedvalues_plot(model_1)
```



In [103…
```
# trainv.loc[[825,523,691],:]
```

In [104…
```
#Function to check Cook's distance
def influencer_detector(model,thershold = 1):

    #create instance of influence
    influence=model.get_influence()

    #Obtain Cook's distance for each observation
    cooks=influence.cooks_distance

    #Check observations w/ Cook's distance greater than thershold
    return np.where(cooks[0] > 1)
```

In [104…
```
# #Check Cook's ditance- model 1
# influencer_detector(model_1)
```

In [104…
```
# trainv.loc[[65,   83,   91,  761,  819,  897, 1059, 1118], :]
```

In [104…
```
# #List of all numeric variables after converting ordinal variables to numeric (not
# var_ind_num3=[3,4,19,20,26,34,36,37,38]+ list(range(43,53))+[54,56,59,61,62]+list
# # var_ind_num2
# len(var_ind_num3)
```

In [104…
```
#Function to check multicollinearity
from statsmodels.stats.outliers_influence import variance_inflation_factor

def calc_vif(X):
    #Calculating VIF
    vif = pd.DataFrame()
    vif["variables"]=X.columns
    vif["VIF"]=[variance_inflation_factor(X, i) for i in range(X.shape[1])]
    return(vif)
```

In [104…

```
#Check multicollinearity for numeric variables
calc_vif(X_trainv)
#Note: If VIF > 10 then multicollinearity is high
```

```
C:\Users\Taban\anaconda3\Lib\site-packages\statsmodels\stats\outliers_influence.p
y:198: RuntimeWarning: divide by zero encountered in scalar divide
  vif = 1. / (1. - r_squared_i)
```

Out[1045]:

	variables	VIF
0	const	3.280421e+06
1	LotFrontage	1.865926e+00
2	LotArea	2.997268e+00
3	OverallQual	6.122441e+00
4	OverallCond	2.768753e+00
...	...	...
207	SaleCondition_Abnorml	1.691455e+00
208	SaleCondition_AdjLand	2.299044e+00
209	SaleCondition_Alloca	1.740580e+00
210	SaleCondition_Family	1.188173e+00
211	SaleCondition_Partial	8.205152e+01

212 rows × 2 columns

In [104…

```
# Regularization: Techniques like Ridge or Lasso regression can help address multic
# ], which can mitigate the impact of collinear variables.
# very large: Extreme collinearity, meaning the variable is nearly a perfect linear
```

In [104…

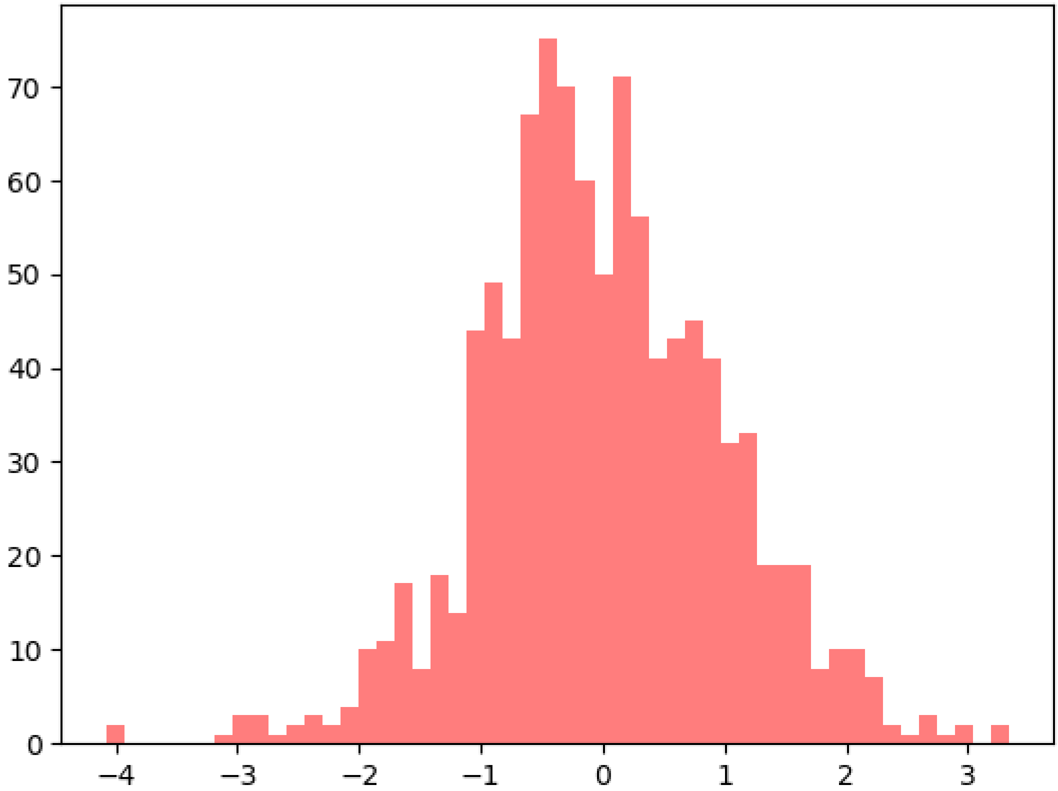
```
# Box-Cox transformation
from sklearn.preprocessing import PowerTransformer
boxcox=PowerTransformer(method='box-cox')
boxcox.fit(y_trainv.values.reshape(-1, 1))
boxcox.lambdas_
```

Out[1047]:

```
array([-0.17292736])
```

In [104…

```
#Linear regression-model 1 with Box-Cox transformation
trans_y_trainv=boxcox.fit_transform(y_trainv.values.reshape(-1, 1))
plt.hist(trans_y_trainv,color='red',alpha = 0.5,bins = 50)
plt.show()
```



In [104…

```
model_1=sm.OLS(trans_y_trainv,X_trainv).fit()
model_1.summary()
```

Out[1049]:

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.946
Model:	OLS	**Adj. R-squared:**	0.932
Method:	Least Squares	**F-statistic:**	69.07
Date:	Fri, 31 Jan 2025	**Prob (F-statistic):**	0.00
Time:	22:33:23	**Log-Likelihood:**	39.667
No. Observations:	1022	**AIC:**	334.7
Df Residuals:	815	**BIC:**	1355.
Df Model:	206		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0321	14.767	-0.002	0.998	-29.018	28.954
LotFrontage	0.0002	0.000	0.660	0.509	-0.000	0.001
LotArea	4.766e-06	1.46e-06	3.261	0.001	1.9e-06	7.64e-06
OverallQual	0.1081	0.014	7.514	0.000	0.080	0.136
OverallCond	0.0892	0.012	7.447	0.000	0.066	0.113
YearBuilt	0.0031	0.001	2.601	0.009	0.001	0.005
YearRemodAdd	0.0021	0.001	2.587	0.010	0.000	0.004
MasVnrArea	5.976e-05	7.83e-05	0.763	0.446	-9.39e-05	0.000
BsmtFinSF1	0.0001	3.65e-05	2.844	0.005	3.22e-05	0.000
BsmtFinSF2	0.0001	6.83e-05	1.772	0.077	-1.3e-05	0.000
BsmtUnfSF	-3.518e-05	3.26e-05	-1.080	0.281	-9.91e-05	2.88e-05
TotalBsmtSF	0.0002	4.75e-05	3.997	0.000	9.65e-05	0.000
1stFlrSF	0.0001	8.58e-05	1.544	0.123	-3.59e-05	0.000
2ndFlrSF	9.517e-05	7.96e-05	1.196	0.232	-6.1e-05	0.000
LowQualFinSF	0.0001	0.000	0.534	0.593	-0.000	0.000
GrLivArea	0.0003	8.11e-05	4.104	0.000	0.000	0.000
BsmtFullBath	0.0590	0.028	2.083	0.038	0.003	0.115
BsmtHalfBath	-0.0160	0.043	-0.367	0.713	-0.101	0.069
FullBath	0.0364	0.031	1.173	0.241	-0.024	0.097
HalfBath	0.0680	0.030	2.284	0.023	0.010	0.126
BedroomAbvGr	0.0146	0.019	0.758	0.449	-0.023	0.053
KitchenAbvGr	-0.0888	0.095	-0.936	0.350	-0.275	0.098
TotRmsAbvGrd	0.0336	0.013	2.550	0.011	0.008	0.059
Fireplaces	0.0359	0.032	1.128	0.260	-0.027	0.098
GarageYrBlt	-1.009e-05	0.001	-0.012	0.990	-0.002	0.002
GarageCars	0.0217	0.032	0.684	0.494	-0.041	0.084

GarageArea	0.0004	0.000	4.185	0.000	0.000	0.001
WoodDeckSF	0.0002	7.95e-05	2.815	0.005	6.78e-05	0.000
OpenPorchSF	0.0003	0.000	1.755	0.080	-3.28e-05	0.001
EnclosedPorch	0.0002	0.000	0.887	0.375	-0.000	0.000
3SsnPorch	0.0004	0.000	1.387	0.166	-0.000	0.001
ScreenPorch	0.0008	0.000	4.982	0.000	0.001	0.001
PoolArea	0.0004	0.001	0.663	0.508	-0.001	0.001
MiscVal	-3.171e-05	7.62e-05	-0.416	0.678	-0.000	0.000
MoSold	0.0011	0.003	0.335	0.737	-0.005	0.008
YrSold	-0.0070	0.007	-0.980	0.328	-0.021	0.007
cnvrt_LotShape	-0.0035	0.019	-0.180	0.858	-0.042	0.035
cnvrt_LandSlope	-0.0015	0.047	-0.033	0.974	-0.094	0.091
cnvrt_ExterQual	-0.0069	0.030	-0.233	0.816	-0.065	0.052
cnvrt_ExterCond	-0.0422	0.029	-1.440	0.150	-0.100	0.015
cnvrt_BsmtQual	0.0118	0.025	0.480	0.631	-0.036	0.060
cnvrt_BsmtCond	0.0321	0.031	1.028	0.304	-0.029	0.093
cnvrt_BsmtExposure	0.0325	0.012	2.790	0.005	0.010	0.055
cnvrt_BsmtFinType1	0.0129	0.007	1.809	0.071	-0.001	0.027
cnvrt_BsmtFinType2	-0.0237	0.017	-1.403	0.161	-0.057	0.009
cnvrt_HeatingQC	0.0427	0.013	3.198	0.001	0.017	0.069
cnvrt_KitchenQual	0.0473	0.023	2.018	0.044	0.001	0.093
cnvrt_Functional	0.0733	0.017	4.283	0.000	0.040	0.107
cnvrt_FireplaceQu	0.0199	0.012	1.690	0.091	-0.003	0.043
cnvrt_GarageFinish	0.0176	0.017	1.058	0.291	-0.015	0.050
cnvrt_GarageQual	0.0798	0.051	1.552	0.121	-0.021	0.181
cnvrt_GarageCond	-0.0467	0.055	-0.854	0.394	-0.154	0.061
cnvrt_PavedDrive	0.0221	0.023	0.940	0.348	-0.024	0.068
cnvrt_PoolQC	-0.0173	0.109	-0.159	0.874	-0.231	0.196
cnvrt_Fence	-0.0056	0.009	-0.652	0.515	-0.023	0.011
MSSubClass_120	-0.1295	0.203	-0.638	0.524	-0.528	0.269
MSSubClass_160	-0.3943	0.241	-1.634	0.103	-0.868	0.079
MSSubClass_180	-0.2370	0.274	-0.866	0.387	-0.774	0.300
MSSubClass_190	-0.3851	0.307	-1.254	0.210	-0.988	0.218
MSSubClass_30	-0.3028	0.066	-4.557	0.000	-0.433	-0.172
MSSubClass_40	0.3148	0.257	1.224	0.221	-0.190	0.820
MSSubClass_45	-0.8807	0.273	-3.224	0.001	-1.417	-0.344
MSSubClass_50	-0.1376	0.134	-1.026	0.305	-0.401	0.126

MSSubClass_60	-0.2727	0.105	-2.610	0.009	-0.478	-0.068
MSSubClass_70	-0.1854	0.116	-1.603	0.109	-0.412	0.042
MSSubClass_75	-0.3267	0.203	-1.608	0.108	-0.726	0.072
MSSubClass_80	-0.1189	0.161	-0.738	0.461	-0.435	0.197
MSSubClass_85	-0.0708	0.168	-0.422	0.673	-0.400	0.259
MSSubClass_90	-0.0474	0.055	-0.870	0.385	-0.154	0.060
MSZoning_C (all)	-1.0446	0.144	-7.250	0.000	-1.327	-0.762
MSZoning_FV	0.1576	0.094	1.668	0.096	-0.028	0.343
MSZoning_RH	-0.0331	0.104	-0.317	0.751	-0.238	0.172
MSZoning_RM	-0.0260	0.052	-0.498	0.619	-0.129	0.077
Street_Grvl	-0.2674	0.147	-1.814	0.070	-0.557	0.022
Alley_Grvl	-0.1421	0.060	-2.361	0.018	-0.260	-0.024
Alley_Pave	0.0364	0.070	0.519	0.604	-0.101	0.174
LandContour_Bnk	-0.0940	0.057	-1.653	0.099	-0.206	0.018
LandContour_HLS	-0.0280	0.056	-0.500	0.617	-0.138	0.082
LandContour_Low	-0.0812	0.073	-1.107	0.269	-0.225	0.063
Utilities_NoSeWa	-0.8332	0.349	-2.390	0.017	-1.517	-0.149
LotConfig_Corner	0.0280	0.025	1.132	0.258	-0.021	0.077
LotConfig_CulDSac	0.1090	0.044	2.480	0.013	0.023	0.195
LotConfig_FR2	-0.0530	0.054	-0.984	0.325	-0.159	0.053
LotConfig_FR3	-0.2238	0.146	-1.532	0.126	-0.511	0.063
Neighborhood_Blmngtn	0.1111	0.108	1.030	0.303	-0.101	0.323
Neighborhood_Blueste	0.2563	0.298	0.860	0.390	-0.329	0.841
Neighborhood_BrDale	-0.0737	0.126	-0.586	0.558	-0.321	0.173
Neighborhood_BrkSide	0.0772	0.072	1.075	0.283	-0.064	0.218
Neighborhood_ClearCr	0.3000	0.088	3.408	0.001	0.127	0.473
Neighborhood_CollgCr	0.0769	0.052	1.478	0.140	-0.025	0.179
Neighborhood_Crawfor	0.3626	0.069	5.254	0.000	0.227	0.498
Neighborhood_Edwards	-0.0597	0.048	-1.237	0.217	-0.154	0.035
Neighborhood_Gilbert	0.0828	0.068	1.216	0.224	-0.051	0.216
Neighborhood_IDOTRR	0.0373	0.098	0.381	0.703	-0.155	0.229
Neighborhood_MeadowV	-0.4050	0.129	-3.135	0.002	-0.659	-0.151
Neighborhood_Mitchel	-0.0297	0.067	-0.446	0.656	-0.160	0.101
Neighborhood_NPkVill	0.1461	0.156	0.934	0.351	-0.161	0.453
Neighborhood_NWAmes	0.0540	0.055	0.989	0.323	-0.053	0.161
Neighborhood_NoRidge	0.1993	0.080	2.484	0.013	0.042	0.357
Neighborhood_NridgHt	0.3197	0.069	4.641	0.000	0.185	0.455

Neighborhood_OldTown	-0.0358	0.072	-0.497	0.619	-0.177	0.106
Neighborhood_SWISU	0.1504	0.090	1.662	0.097	-0.027	0.328
Neighborhood_Sawyer	0.0342	0.046	0.742	0.459	-0.056	0.125
Neighborhood_SawyerW	0.1426	0.064	2.214	0.027	0.016	0.269
Neighborhood_Somerst	0.0906	0.093	0.972	0.332	-0.092	0.274
Neighborhood_StoneBr	0.4278	0.095	4.481	0.000	0.240	0.615
Neighborhood_Timber	0.1193	0.071	1.677	0.094	-0.020	0.259
Neighborhood_Veenker	0.2950	0.115	2.567	0.010	0.069	0.521
Condition1_Artery	-0.1682	0.059	-2.841	0.005	-0.284	-0.052
Condition1_Feedr	-0.1358	0.042	-3.215	0.001	-0.219	-0.053
Condition1_PosA	0.1083	0.140	0.774	0.439	-0.166	0.383
Condition1_PosN	0.0052	0.089	0.059	0.953	-0.170	0.181
Condition1_RRAe	-0.2784	0.100	-2.777	0.006	-0.475	-0.082
Condition1_RRAn	-0.0034	0.074	-0.046	0.963	-0.149	0.142
Condition1_RRNe	0.0399	0.270	0.148	0.883	-0.491	0.570
Condition1_RRNn	0.0374	0.144	0.259	0.796	-0.246	0.321
Condition2_Artery	-0.5176	0.271	-1.912	0.056	-1.049	0.014
Condition2_Feedr	0.1068	0.158	0.677	0.499	-0.203	0.416
Condition2_PosA	0.3422	0.330	1.036	0.300	-0.306	0.990
Condition2_PosN	-1.9339	0.231	-8.368	0.000	-2.387	-1.480
Condition2_RRAe	-0.3072	0.341	-0.901	0.368	-0.976	0.362
Condition2_RRNn	0.0069	0.288	0.024	0.981	-0.559	0.573
BldgType_2fmCon	0.1771	0.292	0.606	0.545	-0.397	0.751
BldgType_Duplex	-0.0474	0.055	-0.870	0.385	-0.154	0.060
BldgType_Twnhs	-0.1501	0.216	-0.694	0.488	-0.575	0.275
BldgType_TwnhsE	-0.0449	0.204	-0.220	0.826	-0.446	0.356
HouseStyle_1.5Fin	0.1616	0.135	1.200	0.231	-0.103	0.426
HouseStyle_1.5Unf	0.7876	0.266	2.961	0.003	0.266	1.310
HouseStyle_2.5Fin	0.1899	0.246	0.771	0.441	-0.294	0.673
HouseStyle_2.5Unf	0.4390	0.207	2.125	0.034	0.034	0.844
HouseStyle_2Story	0.2057	0.116	1.775	0.076	-0.022	0.433
HouseStyle_SFoyer	0.0145	0.146	0.099	0.921	-0.273	0.302
HouseStyle_SLvl	0.1124	0.158	0.710	0.478	-0.198	0.423
RoofStyle_Flat	-0.0151	0.221	-0.068	0.945	-0.449	0.418
RoofStyle_Gambrel	-0.0954	0.120	-0.796	0.426	-0.331	0.140
RoofStyle_Hip	0.0093	0.025	0.368	0.713	-0.040	0.059
RoofStyle_Mansard	0.1692	0.140	1.208	0.227	-0.106	0.444

RoofStyle_Shed	-0.3072	0.341	-0.901	0.368	-0.976	0.362
Heating_Floor	-0.2804	0.301	-0.931	0.352	-0.872	0.311
Heating_GasW	0.1499	0.112	1.344	0.179	-0.069	0.369
Heating_Grav	-0.1027	0.167	-0.617	0.538	-0.430	0.224
Heating_OthW	0.0175	0.226	0.077	0.938	-0.427	0.461
Heating_Wall	0.5553	0.189	2.946	0.003	0.185	0.925
RoofMatl_ClyTile	-5.6421	0.492	-11.476	0.000	-6.607	-4.677
RoofMatl_Membran	0.4879	0.350	1.393	0.164	-0.200	1.175
RoofMatl_Metal	0.0560	0.362	0.155	0.877	-0.654	0.766
RoofMatl_Tar&Grv	-0.0436	0.221	-0.198	0.843	-0.477	0.390
RoofMatl_WdShake	-0.3993	0.277	-1.440	0.150	-0.943	0.145
RoofMatl_WdShngl	0.2035	0.128	1.592	0.112	-0.047	0.454
Exterior1st_AsbShng	-0.0225	0.177	-0.127	0.899	-0.369	0.324
Exterior1st_BrkComm	-1.4896	0.387	-3.845	0.000	-2.250	-0.729
Exterior1st_BrkFace	0.2079	0.114	1.826	0.068	-0.016	0.431
Exterior1st_CBlock	-0.1218	0.144	-0.844	0.399	-0.405	0.161
Exterior1st_CemntBd	-0.3859	0.238	-1.621	0.105	-0.853	0.081
Exterior1st_HdBoard	-0.0630	0.109	-0.575	0.565	-0.278	0.152
Exterior1st_ImStucc	-0.0384	0.314	-0.122	0.903	-0.654	0.577
Exterior1st_MetalSd	0.0986	0.149	0.664	0.507	-0.193	0.390
Exterior1st_Plywood	-0.0468	0.109	-0.431	0.666	-0.260	0.166
Exterior1st_Stone	0.1593	0.300	0.531	0.596	-0.430	0.749
Exterior1st_Stucco	-0.1948	0.158	-1.230	0.219	-0.506	0.116
Exterior1st_Wd Sdng	-0.1348	0.102	-1.317	0.188	-0.336	0.066
Exterior1st_WdShing	-0.0126	0.123	-0.102	0.919	-0.254	0.229
Exterior2nd_AsbShng	-0.0854	0.173	-0.494	0.621	-0.424	0.254
Exterior2nd_AsphShn	0.0584	0.244	0.239	0.811	-0.421	0.538
Exterior2nd_Brk Cmn	0.0377	0.265	0.143	0.887	-0.482	0.557
Exterior2nd_BrkFace	-0.2067	0.131	-1.573	0.116	-0.465	0.051
Exterior2nd_CBlock	-0.1218	0.144	-0.844	0.399	-0.405	0.161
Exterior2nd_CmentBd	0.4387	0.237	1.852	0.064	-0.026	0.904
Exterior2nd_HdBoard	-0.0183	0.109	-0.167	0.867	-0.233	0.197
Exterior2nd_ImStucc	0.0728	0.151	0.482	0.630	-0.224	0.369
Exterior2nd_MetalSd	-0.0763	0.151	-0.504	0.615	-0.374	0.221
Exterior2nd_Other	-0.3294	0.279	-1.180	0.239	-0.877	0.219
Exterior2nd_Plywood	-0.0143	0.104	-0.137	0.891	-0.218	0.190
Exterior2nd_Stone	-0.0382	0.222	-0.172	0.863	-0.474	0.397

Exterior2nd_Stucco	0.1952	0.154	1.268	0.205	-0.107	0.497
Exterior2nd_Wd Sdng	0.1167	0.103	1.128	0.260	-0.086	0.320
Exterior2nd_Wd Shng	-0.0937	0.111	-0.844	0.399	-0.312	0.124
MasVnrType_BrkCmn	-0.1013	0.090	-1.129	0.259	-0.277	0.075
MasVnrType_BrkFace	0.0081	0.031	0.260	0.795	-0.053	0.069
MasVnrType_Stone	0.0382	0.044	0.869	0.385	-0.048	0.124
Foundation_BrkTil	-0.1383	0.049	-2.796	0.005	-0.235	-0.041
Foundation_CBlock	-0.0688	0.033	-2.072	0.039	-0.134	-0.004
Foundation_Slab	-0.1562	0.132	-1.182	0.238	-0.416	0.103
Foundation_Stone	0.0916	0.138	0.665	0.506	-0.179	0.362
Foundation_Wood	-0.3256	0.171	-1.902	0.058	-0.662	0.010
CentralAir_N	-0.2024	0.058	-3.470	0.001	-0.317	-0.088
Electrical_FuseA	0.0244	0.044	0.548	0.584	-0.063	0.112
Electrical_FuseF	-0.1157	0.081	-1.433	0.152	-0.274	0.043
Electrical_FuseP	0.1259	0.209	0.601	0.548	-0.285	0.537
GarageType_2Types	-0.1784	0.143	-1.247	0.213	-0.459	0.102
GarageType_Basment	0.0811	0.102	0.798	0.425	-0.118	0.280
GarageType_BuiltIn	-0.0066	0.049	-0.133	0.894	-0.104	0.090
GarageType_CarPort	0.0907	0.115	0.790	0.430	-0.135	0.316
GarageType_Detchd	-0.0478	0.032	-1.509	0.132	-0.110	0.014
GarageType_NA	0.1485	1.599	0.093	0.926	-2.991	3.288
MiscFeature_Gar2	0.6033	1.209	0.499	0.618	-1.771	2.977
MiscFeature_Othr	-0.4330	0.303	-1.430	0.153	-1.027	0.161
MiscFeature_Shed	0.0108	0.081	0.133	0.894	-0.148	0.170
MiscFeature_TenC	-0.6315	0.382	-1.655	0.098	-1.380	0.117
SaleType_COD	0.0277	0.057	0.483	0.629	-0.085	0.140
SaleType_CWD	0.2138	0.194	1.101	0.271	-0.167	0.595
SaleType_Con	0.2296	0.199	1.153	0.249	-0.161	0.620
SaleType_ConLD	0.6047	0.172	3.506	0.000	0.266	0.943
SaleType_ConLI	-0.2184	0.168	-1.302	0.193	-0.548	0.111
SaleType_ConLw	-0.0046	0.277	-0.017	0.987	-0.548	0.539
SaleType_New	-0.2485	0.278	-0.893	0.372	-0.795	0.297
SaleType_Oth	0.0307	0.169	0.182	0.856	-0.301	0.362
SaleCondition_Abnorml	-0.0852	0.041	-2.069	0.039	-0.166	-0.004
SaleCondition_AdjLand	-0.0407	0.280	-0.146	0.884	-0.590	0.508
SaleCondition_Alloca	-0.1366	0.122	-1.119	0.263	-0.376	0.103
SaleCondition_Family	-0.1123	0.086	-1.304	0.193	-0.281	0.057

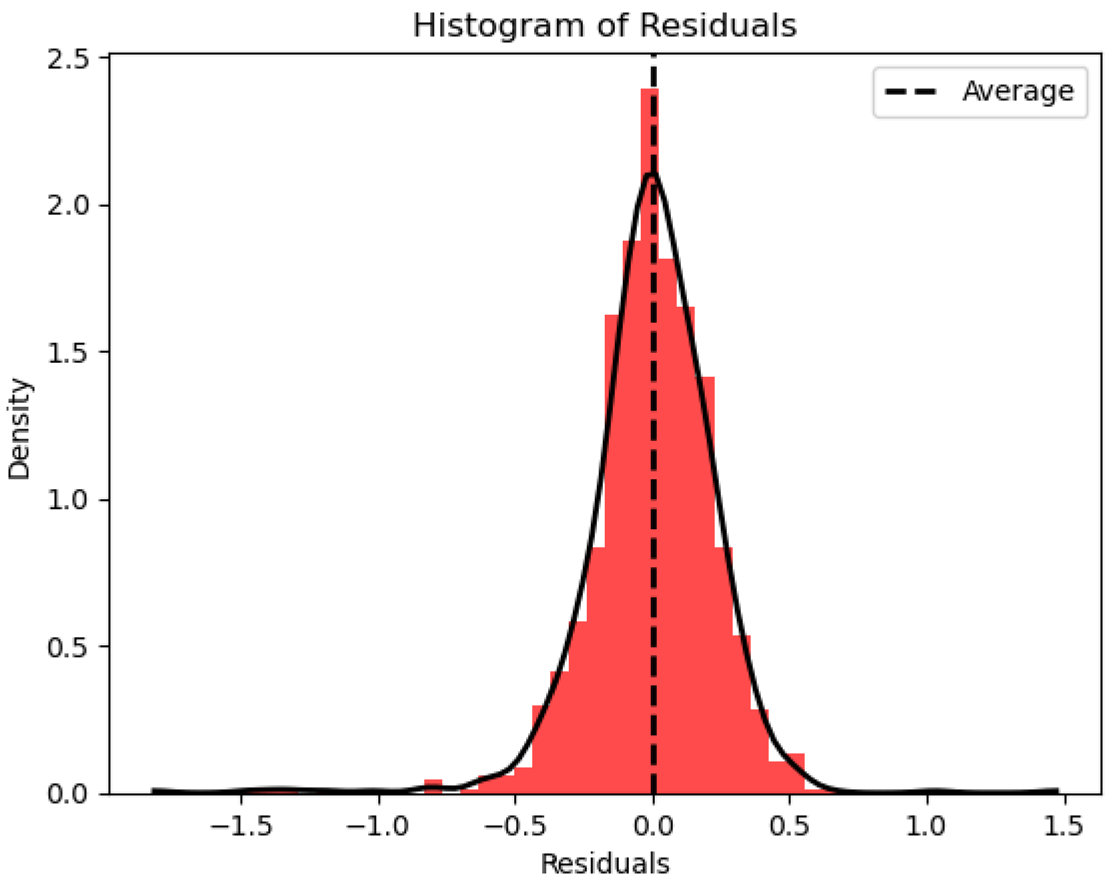
SaleCondition_Partial	0.3265	0.275	1.188	0.235	-0.213	0.866

Omnibus:	325.596	**Durbin-Watson:**	1.911
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	3970.454
Skew:	-1.101	**Prob(JB):**	0.00
Kurtosis:	12.402	**Cond. No.**	7.65e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.83e-23. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

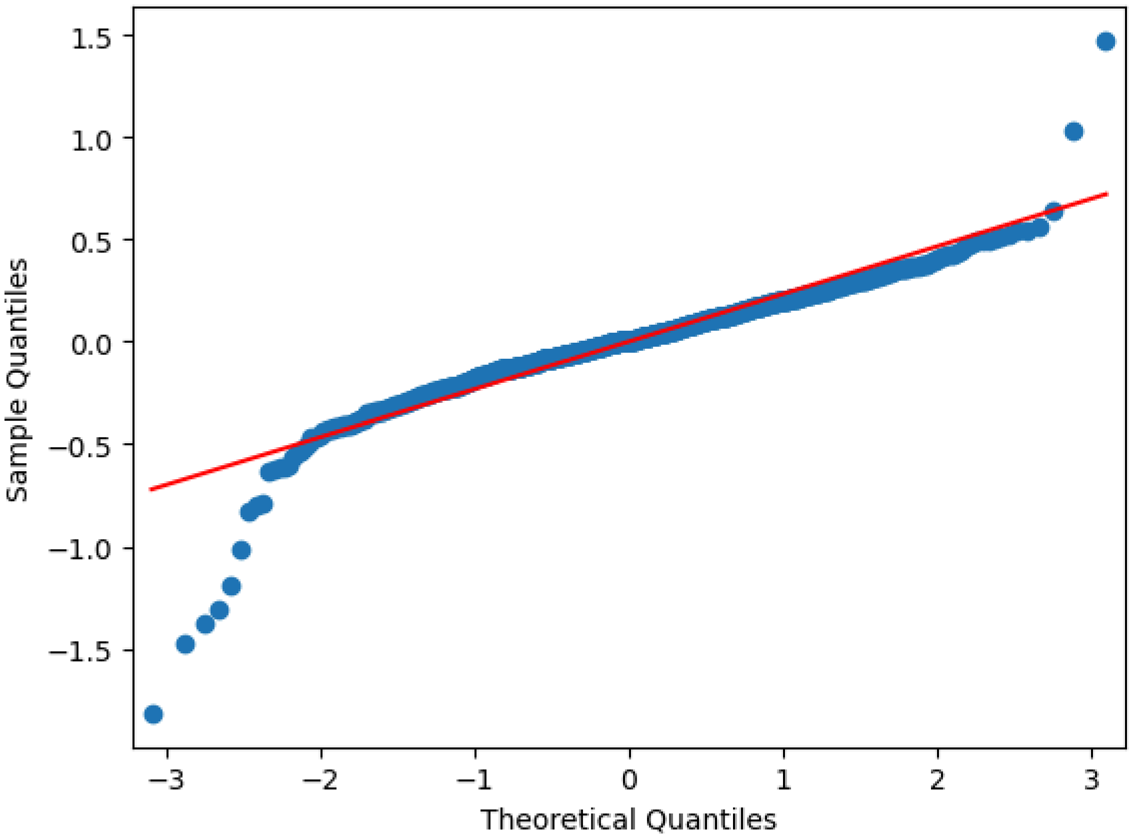
In [105…

```
# Histogram of residuals- model 1 with Box-Cox transformation
hist_residuals(model_1)
```



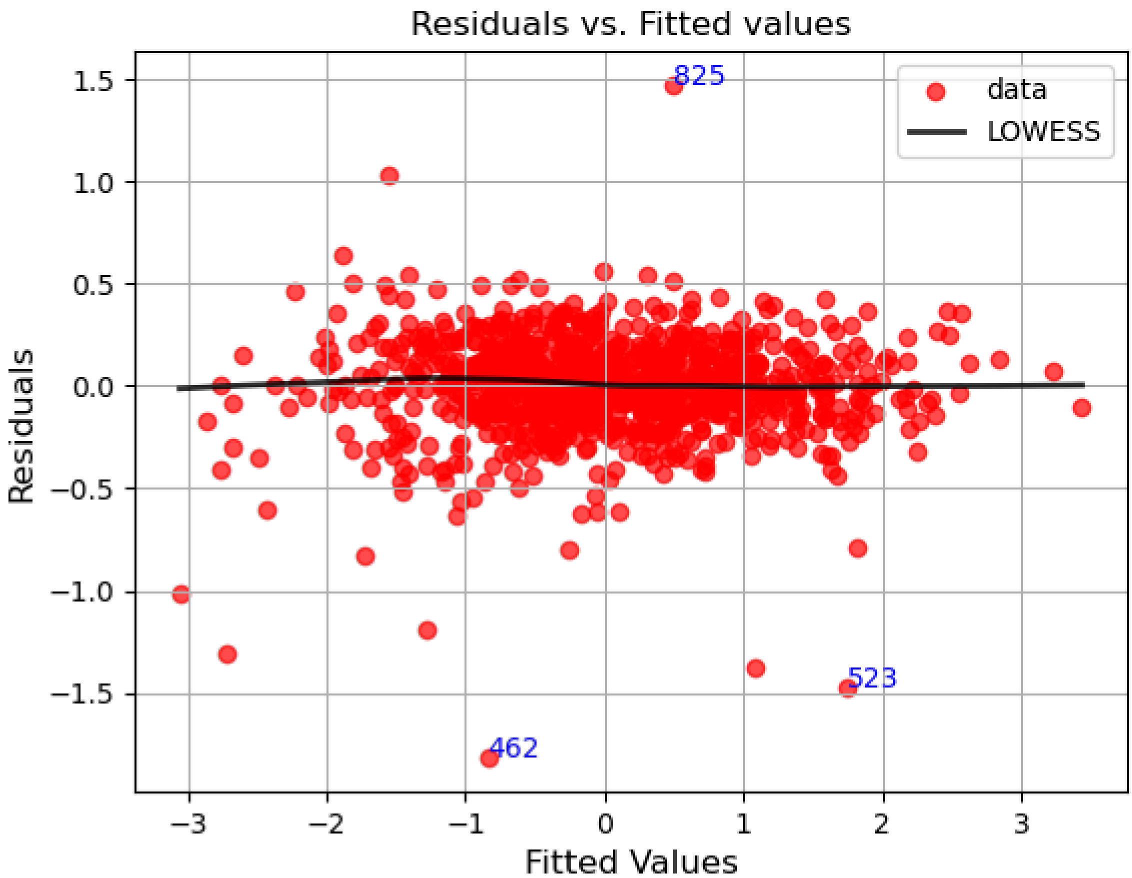
In [105…

```
# QQ-plot- model 1 with Box-Cox transformation
sm.qqplot(model_1.resid,line='s')
plt.show()
```



In [105…

```
# Scatter plot of residuals vs. fitted values- model 1 with Box-Cox transformation
residuals_fittedvalues_plot(model_1)
```



In [105…

```
# trainv.loc[[462,632,523], :]
```

In [105…

```
# Check Cook's ditance- model 1 with Box-Cox transformation
# influencer_detector(model_1)
```

In [105…

```
# X_trainv.loc[[ 22,   65,   83,   91,  761,  819,  897, 1059, 1118], :]
```

In [105…

```
# Note: Remove observations
 # X_train.drop(index =[1322,816,970], inplace = True)
 # y_train.drop(index =[591,816,970], inplace = True)
# trans_y_train=PowerTransformer(method='box-cox').fit_transform(y_train.values.res
```

In [105…

```
model_1 = sm.OLS(trans_y_trainv,X_trainv).fit()
model_1.summary()
```

Out[1057]:

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.946
Model:	OLS	**Adj. R-squared:**	0.932
Method:	Least Squares	**F-statistic:**	69.07
Date:	Fri, 31 Jan 2025	**Prob (F-statistic):**	0.00
Time:	22:33:25	**Log-Likelihood:**	39.667
No. Observations:	1022	**AIC:**	334.7
Df Residuals:	815	**BIC:**	1355.
Df Model:	206		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-0.0321	14.767	-0.002	0.998	-29.018	28.954
LotFrontage	0.0002	0.000	0.660	0.509	-0.000	0.001
LotArea	4.766e-06	1.46e-06	3.261	0.001	1.9e-06	7.64e-06
OverallQual	0.1081	0.014	7.514	0.000	0.080	0.136
OverallCond	0.0892	0.012	7.447	0.000	0.066	0.113
YearBuilt	0.0031	0.001	2.601	0.009	0.001	0.005
YearRemodAdd	0.0021	0.001	2.587	0.010	0.000	0.004
MasVnrArea	5.976e-05	7.83e-05	0.763	0.446	-9.39e-05	0.000
BsmtFinSF1	0.0001	3.65e-05	2.844	0.005	3.22e-05	0.000
BsmtFinSF2	0.0001	6.83e-05	1.772	0.077	-1.3e-05	0.000
BsmtUnfSF	-3.518e-05	3.26e-05	-1.080	0.281	-9.91e-05	2.88e-05
TotalBsmtSF	0.0002	4.75e-05	3.997	0.000	9.65e-05	0.000
1stFlrSF	0.0001	8.58e-05	1.544	0.123	-3.59e-05	0.000
2ndFlrSF	9.517e-05	7.96e-05	1.196	0.232	-6.1e-05	0.000
LowQualFinSF	0.0001	0.000	0.534	0.593	-0.000	0.000
GrLivArea	0.0003	8.11e-05	4.104	0.000	0.000	0.000
BsmtFullBath	0.0590	0.028	2.083	0.038	0.003	0.115
BsmtHalfBath	-0.0160	0.043	-0.367	0.713	-0.101	0.069
FullBath	0.0364	0.031	1.173	0.241	-0.024	0.097
HalfBath	0.0680	0.030	2.284	0.023	0.010	0.126
BedroomAbvGr	0.0146	0.019	0.758	0.449	-0.023	0.053
KitchenAbvGr	-0.0888	0.095	-0.936	0.350	-0.275	0.098
TotRmsAbvGrd	0.0336	0.013	2.550	0.011	0.008	0.059
Fireplaces	0.0359	0.032	1.128	0.260	-0.027	0.098
GarageYrBlt	-1.009e-05	0.001	-0.012	0.990	-0.002	0.002
GarageCars	0.0217	0.032	0.684	0.494	-0.041	0.084

GarageArea	0.0004	0.000	4.185	0.000	0.000	0.001
WoodDeckSF	0.0002	7.95e-05	2.815	0.005	6.78e-05	0.000
OpenPorchSF	0.0003	0.000	1.755	0.080	-3.28e-05	0.001
EnclosedPorch	0.0002	0.000	0.887	0.375	-0.000	0.000
3SsnPorch	0.0004	0.000	1.387	0.166	-0.000	0.001
ScreenPorch	0.0008	0.000	4.982	0.000	0.001	0.001
PoolArea	0.0004	0.001	0.663	0.508	-0.001	0.001
MiscVal	-3.171e-05	7.62e-05	-0.416	0.678	-0.000	0.000
MoSold	0.0011	0.003	0.335	0.737	-0.005	0.008
YrSold	-0.0070	0.007	-0.980	0.328	-0.021	0.007
cnvrt_LotShape	-0.0035	0.019	-0.180	0.858	-0.042	0.035
cnvrt_LandSlope	-0.0015	0.047	-0.033	0.974	-0.094	0.091
cnvrt_ExterQual	-0.0069	0.030	-0.233	0.816	-0.065	0.052
cnvrt_ExterCond	-0.0422	0.029	-1.440	0.150	-0.100	0.015
cnvrt_BsmtQual	0.0118	0.025	0.480	0.631	-0.036	0.060
cnvrt_BsmtCond	0.0321	0.031	1.028	0.304	-0.029	0.093
cnvrt_BsmtExposure	0.0325	0.012	2.790	0.005	0.010	0.055
cnvrt_BsmtFinType1	0.0129	0.007	1.809	0.071	-0.001	0.027
cnvrt_BsmtFinType2	-0.0237	0.017	-1.403	0.161	-0.057	0.009
cnvrt_HeatingQC	0.0427	0.013	3.198	0.001	0.017	0.069
cnvrt_KitchenQual	0.0473	0.023	2.018	0.044	0.001	0.093
cnvrt_Functional	0.0733	0.017	4.283	0.000	0.040	0.107
cnvrt_FireplaceQu	0.0199	0.012	1.690	0.091	-0.003	0.043
cnvrt_GarageFinish	0.0176	0.017	1.058	0.291	-0.015	0.050
cnvrt_GarageQual	0.0798	0.051	1.552	0.121	-0.021	0.181
cnvrt_GarageCond	-0.0467	0.055	-0.854	0.394	-0.154	0.061
cnvrt_PavedDrive	0.0221	0.023	0.940	0.348	-0.024	0.068
cnvrt_PoolQC	-0.0173	0.109	-0.159	0.874	-0.231	0.196
cnvrt_Fence	-0.0056	0.009	-0.652	0.515	-0.023	0.011
MSSubClass_120	-0.1295	0.203	-0.638	0.524	-0.528	0.269
MSSubClass_160	-0.3943	0.241	-1.634	0.103	-0.868	0.079
MSSubClass_180	-0.2370	0.274	-0.866	0.387	-0.774	0.300
MSSubClass_190	-0.3851	0.307	-1.254	0.210	-0.988	0.218
MSSubClass_30	-0.3028	0.066	-4.557	0.000	-0.433	-0.172
MSSubClass_40	0.3148	0.257	1.224	0.221	-0.190	0.820
MSSubClass_45	-0.8807	0.273	-3.224	0.001	-1.417	-0.344
MSSubClass_50	-0.1376	0.134	-1.026	0.305	-0.401	0.126

MSSubClass_60	-0.2727	0.105	-2.610	0.009	-0.478	-0.068
MSSubClass_70	-0.1854	0.116	-1.603	0.109	-0.412	0.042
MSSubClass_75	-0.3267	0.203	-1.608	0.108	-0.726	0.072
MSSubClass_80	-0.1189	0.161	-0.738	0.461	-0.435	0.197
MSSubClass_85	-0.0708	0.168	-0.422	0.673	-0.400	0.259
MSSubClass_90	-0.0474	0.055	-0.870	0.385	-0.154	0.060
MSZoning_C (all)	-1.0446	0.144	-7.250	0.000	-1.327	-0.762
MSZoning_FV	0.1576	0.094	1.668	0.096	-0.028	0.343
MSZoning_RH	-0.0331	0.104	-0.317	0.751	-0.238	0.172
MSZoning_RM	-0.0260	0.052	-0.498	0.619	-0.129	0.077
Street_Grvl	-0.2674	0.147	-1.814	0.070	-0.557	0.022
Alley_Grvl	-0.1421	0.060	-2.361	0.018	-0.260	-0.024
Alley_Pave	0.0364	0.070	0.519	0.604	-0.101	0.174
LandContour_Bnk	-0.0940	0.057	-1.653	0.099	-0.206	0.018
LandContour_HLS	-0.0280	0.056	-0.500	0.617	-0.138	0.082
LandContour_Low	-0.0812	0.073	-1.107	0.269	-0.225	0.063
Utilities_NoSeWa	-0.8332	0.349	-2.390	0.017	-1.517	-0.149
LotConfig_Corner	0.0280	0.025	1.132	0.258	-0.021	0.077
LotConfig_CulDSac	0.1090	0.044	2.480	0.013	0.023	0.195
LotConfig_FR2	-0.0530	0.054	-0.984	0.325	-0.159	0.053
LotConfig_FR3	-0.2238	0.146	-1.532	0.126	-0.511	0.063
Neighborhood_Blmngtn	0.1111	0.108	1.030	0.303	-0.101	0.323
Neighborhood_Blueste	0.2563	0.298	0.860	0.390	-0.329	0.841
Neighborhood_BrDale	-0.0737	0.126	-0.586	0.558	-0.321	0.173
Neighborhood_BrkSide	0.0772	0.072	1.075	0.283	-0.064	0.218
Neighborhood_ClearCr	0.3000	0.088	3.408	0.001	0.127	0.473
Neighborhood_CollgCr	0.0769	0.052	1.478	0.140	-0.025	0.179
Neighborhood_Crawfor	0.3626	0.069	5.254	0.000	0.227	0.498
Neighborhood_Edwards	-0.0597	0.048	-1.237	0.217	-0.154	0.035
Neighborhood_Gilbert	0.0828	0.068	1.216	0.224	-0.051	0.216
Neighborhood_IDOTRR	0.0373	0.098	0.381	0.703	-0.155	0.229
Neighborhood_MeadowV	-0.4050	0.129	-3.135	0.002	-0.659	-0.151
Neighborhood_Mitchel	-0.0297	0.067	-0.446	0.656	-0.160	0.101
Neighborhood_NPkVill	0.1461	0.156	0.934	0.351	-0.161	0.453
Neighborhood_NWAmes	0.0540	0.055	0.989	0.323	-0.053	0.161
Neighborhood_NoRidge	0.1993	0.080	2.484	0.013	0.042	0.357
Neighborhood_NridgHt	0.3197	0.069	4.641	0.000	0.185	0.455

Neighborhood_OldTown	-0.0358	0.072	-0.497	0.619	-0.177	0.106
Neighborhood_SWISU	0.1504	0.090	1.662	0.097	-0.027	0.328
Neighborhood_Sawyer	0.0342	0.046	0.742	0.459	-0.056	0.125
Neighborhood_SawyerW	0.1426	0.064	2.214	0.027	0.016	0.269
Neighborhood_Somerst	0.0906	0.093	0.972	0.332	-0.092	0.274
Neighborhood_StoneBr	0.4278	0.095	4.481	0.000	0.240	0.615
Neighborhood_Timber	0.1193	0.071	1.677	0.094	-0.020	0.259
Neighborhood_Veenker	0.2950	0.115	2.567	0.010	0.069	0.521
Condition1_Artery	-0.1682	0.059	-2.841	0.005	-0.284	-0.052
Condition1_Feedr	-0.1358	0.042	-3.215	0.001	-0.219	-0.053
Condition1_PosA	0.1083	0.140	0.774	0.439	-0.166	0.383
Condition1_PosN	0.0052	0.089	0.059	0.953	-0.170	0.181
Condition1_RRAe	-0.2784	0.100	-2.777	0.006	-0.475	-0.082
Condition1_RRAn	-0.0034	0.074	-0.046	0.963	-0.149	0.142
Condition1_RRNe	0.0399	0.270	0.148	0.883	-0.491	0.570
Condition1_RRNn	0.0374	0.144	0.259	0.796	-0.246	0.321
Condition2_Artery	-0.5176	0.271	-1.912	0.056	-1.049	0.014
Condition2_Feedr	0.1068	0.158	0.677	0.499	-0.203	0.416
Condition2_PosA	0.3422	0.330	1.036	0.300	-0.306	0.990
Condition2_PosN	-1.9339	0.231	-8.368	0.000	-2.387	-1.480
Condition2_RRAe	-0.3072	0.341	-0.901	0.368	-0.976	0.362
Condition2_RRNn	0.0069	0.288	0.024	0.981	-0.559	0.573
BldgType_2fmCon	0.1771	0.292	0.606	0.545	-0.397	0.751
BldgType_Duplex	-0.0474	0.055	-0.870	0.385	-0.154	0.060
BldgType_Twnhs	-0.1501	0.216	-0.694	0.488	-0.575	0.275
BldgType_TwnhsE	-0.0449	0.204	-0.220	0.826	-0.446	0.356
HouseStyle_1.5Fin	0.1616	0.135	1.200	0.231	-0.103	0.426
HouseStyle_1.5Unf	0.7876	0.266	2.961	0.003	0.266	1.310
HouseStyle_2.5Fin	0.1899	0.246	0.771	0.441	-0.294	0.673
HouseStyle_2.5Unf	0.4390	0.207	2.125	0.034	0.034	0.844
HouseStyle_2Story	0.2057	0.116	1.775	0.076	-0.022	0.433
HouseStyle_SFoyer	0.0145	0.146	0.099	0.921	-0.273	0.302
HouseStyle_SLvl	0.1124	0.158	0.710	0.478	-0.198	0.423
RoofStyle_Flat	-0.0151	0.221	-0.068	0.945	-0.449	0.418
RoofStyle_Gambrel	-0.0954	0.120	-0.796	0.426	-0.331	0.140
RoofStyle_Hip	0.0093	0.025	0.368	0.713	-0.040	0.059
RoofStyle_Mansard	0.1692	0.140	1.208	0.227	-0.106	0.444

RoofStyle_Shed	-0.3072	0.341	-0.901	0.368	-0.976	0.362
Heating_Floor	-0.2804	0.301	-0.931	0.352	-0.872	0.311
Heating_GasW	0.1499	0.112	1.344	0.179	-0.069	0.369
Heating_Grav	-0.1027	0.167	-0.617	0.538	-0.430	0.224
Heating_OthW	0.0175	0.226	0.077	0.938	-0.427	0.461
Heating_Wall	0.5553	0.189	2.946	0.003	0.185	0.925
RoofMatl_ClyTile	-5.6421	0.492	-11.476	0.000	-6.607	-4.677
RoofMatl_Membran	0.4879	0.350	1.393	0.164	-0.200	1.175
RoofMatl_Metal	0.0560	0.362	0.155	0.877	-0.654	0.766
RoofMatl_Tar&Grv	-0.0436	0.221	-0.198	0.843	-0.477	0.390
RoofMatl_WdShake	-0.3993	0.277	-1.440	0.150	-0.943	0.145
RoofMatl_WdShngl	0.2035	0.128	1.592	0.112	-0.047	0.454
Exterior1st_AsbShng	-0.0225	0.177	-0.127	0.899	-0.369	0.324
Exterior1st_BrkComm	-1.4896	0.387	-3.845	0.000	-2.250	-0.729
Exterior1st_BrkFace	0.2079	0.114	1.826	0.068	-0.016	0.431
Exterior1st_CBlock	-0.1218	0.144	-0.844	0.399	-0.405	0.161
Exterior1st_CemntBd	-0.3859	0.238	-1.621	0.105	-0.853	0.081
Exterior1st_HdBoard	-0.0630	0.109	-0.575	0.565	-0.278	0.152
Exterior1st_ImStucc	-0.0384	0.314	-0.122	0.903	-0.654	0.577
Exterior1st_MetalSd	0.0986	0.149	0.664	0.507	-0.193	0.390
Exterior1st_Plywood	-0.0468	0.109	-0.431	0.666	-0.260	0.166
Exterior1st_Stone	0.1593	0.300	0.531	0.596	-0.430	0.749
Exterior1st_Stucco	-0.1948	0.158	-1.230	0.219	-0.506	0.116
Exterior1st_Wd Sdng	-0.1348	0.102	-1.317	0.188	-0.336	0.066
Exterior1st_WdShing	-0.0126	0.123	-0.102	0.919	-0.254	0.229
Exterior2nd_AsbShng	-0.0854	0.173	-0.494	0.621	-0.424	0.254
Exterior2nd_AsphShn	0.0584	0.244	0.239	0.811	-0.421	0.538
Exterior2nd_Brk Cmn	0.0377	0.265	0.143	0.887	-0.482	0.557
Exterior2nd_BrkFace	-0.2067	0.131	-1.573	0.116	-0.465	0.051
Exterior2nd_CBlock	-0.1218	0.144	-0.844	0.399	-0.405	0.161
Exterior2nd_CmentBd	0.4387	0.237	1.852	0.064	-0.026	0.904
Exterior2nd_HdBoard	-0.0183	0.109	-0.167	0.867	-0.233	0.197
Exterior2nd_ImStucc	0.0728	0.151	0.482	0.630	-0.224	0.369
Exterior2nd_MetalSd	-0.0763	0.151	-0.504	0.615	-0.374	0.221
Exterior2nd_Other	-0.3294	0.279	-1.180	0.239	-0.877	0.219
Exterior2nd_Plywood	-0.0143	0.104	-0.137	0.891	-0.218	0.190
Exterior2nd_Stone	-0.0382	0.222	-0.172	0.863	-0.474	0.397

Exterior2nd_Stucco	0.1952	0.154	1.268	0.205	-0.107	0.497
Exterior2nd_Wd Sdng	0.1167	0.103	1.128	0.260	-0.086	0.320
Exterior2nd_Wd Shng	-0.0937	0.111	-0.844	0.399	-0.312	0.124
MasVnrType_BrkCmn	-0.1013	0.090	-1.129	0.259	-0.277	0.075
MasVnrType_BrkFace	0.0081	0.031	0.260	0.795	-0.053	0.069
MasVnrType_Stone	0.0382	0.044	0.869	0.385	-0.048	0.124
Foundation_BrkTil	-0.1383	0.049	-2.796	0.005	-0.235	-0.041
Foundation_CBlock	-0.0688	0.033	-2.072	0.039	-0.134	-0.004
Foundation_Slab	-0.1562	0.132	-1.182	0.238	-0.416	0.103
Foundation_Stone	0.0916	0.138	0.665	0.506	-0.179	0.362
Foundation_Wood	-0.3256	0.171	-1.902	0.058	-0.662	0.010
CentralAir_N	-0.2024	0.058	-3.470	0.001	-0.317	-0.088
Electrical_FuseA	0.0244	0.044	0.548	0.584	-0.063	0.112
Electrical_FuseF	-0.1157	0.081	-1.433	0.152	-0.274	0.043
Electrical_FuseP	0.1259	0.209	0.601	0.548	-0.285	0.537
GarageType_2Types	-0.1784	0.143	-1.247	0.213	-0.459	0.102
GarageType_Basment	0.0811	0.102	0.798	0.425	-0.118	0.280
GarageType_BuiltIn	-0.0066	0.049	-0.133	0.894	-0.104	0.090
GarageType_CarPort	0.0907	0.115	0.790	0.430	-0.135	0.316
GarageType_Detchd	-0.0478	0.032	-1.509	0.132	-0.110	0.014
GarageType_NA	0.1485	1.599	0.093	0.926	-2.991	3.288
MiscFeature_Gar2	0.6033	1.209	0.499	0.618	-1.771	2.977
MiscFeature_Othr	-0.4330	0.303	-1.430	0.153	-1.027	0.161
MiscFeature_Shed	0.0108	0.081	0.133	0.894	-0.148	0.170
MiscFeature_TenC	-0.6315	0.382	-1.655	0.098	-1.380	0.117
SaleType_COD	0.0277	0.057	0.483	0.629	-0.085	0.140
SaleType_CWD	0.2138	0.194	1.101	0.271	-0.167	0.595
SaleType_Con	0.2296	0.199	1.153	0.249	-0.161	0.620
SaleType_ConLD	0.6047	0.172	3.506	0.000	0.266	0.943
SaleType_ConLI	-0.2184	0.168	-1.302	0.193	-0.548	0.111
SaleType_ConLw	-0.0046	0.277	-0.017	0.987	-0.548	0.539
SaleType_New	-0.2485	0.278	-0.893	0.372	-0.795	0.297
SaleType_Oth	0.0307	0.169	0.182	0.856	-0.301	0.362
SaleCondition_Abnorml	-0.0852	0.041	-2.069	0.039	-0.166	-0.004
SaleCondition_AdjLand	-0.0407	0.280	-0.146	0.884	-0.590	0.508
SaleCondition_Alloca	-0.1366	0.122	-1.119	0.263	-0.376	0.103
SaleCondition_Family	-0.1123	0.086	-1.304	0.193	-0.281	0.057

SaleCondition_Partial	0.3265	0.275	1.188	0.235	-0.213	0.866

Omnibus:	325.596	**Durbin-Watson:**	1.911
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	3970.454
Skew:	-1.101	**Prob(JB):**	0.00
Kurtosis:	12.402	**Cond. No.**	7.65e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.83e-23. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

In [105…

```
# Get the p-values from the model
pvalues=model_1.pvalues

# Define the significance level (alpha)
alpha=0.05

# Identify columns with p-value greater than alpha
non_significant_columns=pvalues[pvalues>alpha].index
print("Non-significant columns to drop:",non_significant_columns)
# Drop non-significant columns from the original X_trainv DataFrame
X_trainv_significant=X_trainv.drop(columns=non_significant_columns)

# Refit the model with the remaining significant columns
model_1_significant=sm.OLS(trans_y_trainv, X_trainv_significant).fit()

# Summary of the new model
print(model_1_significant.summary())
# Start with the original X_trainv
X_trainv_current=X_trainv.copy()

while True:
    model = sm.OLS(trans_y_trainv, X_trainv_current).fit()
    pvalues = model.pvalues

    # Check if all p-values are below the significance level
    if pvalues.max()>alpha:
        # Drop the column with the highest p-value
        non_significant_column = pvalues.idxmax()
        print(f"Dropping non-significant column: {non_significant_column} with p-va
        X_trainv_current = X_trainv_current.drop(columns=[non_significant_column])
    else:
        break
```

```
Non-significant columns to drop: Index(['const', 'LotFrontage', 'MasVnrArea', 'Bsm
tFinSF2', 'BsmtUnfSF',
       '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'BsmtHalfBath', 'FullBath',
       ...
       'SaleType_CWD', 'SaleType_Con', 'SaleType_ConLI', 'SaleType_ConLw',
       'SaleType_New', 'SaleType_Oth', 'SaleCondition_AdjLand',
       'SaleCondition_Alloca', 'SaleCondition_Family',
       'SaleCondition_Partial'],
      dtype='object', length=165)
                                 OLS Regression Results
==================================================================================
=====
Dep. Variable:                      y   R-squared (uncentered):
0.917
Model:                            OLS   Adj. R-squared (uncentered):
0.913
Method:                 Least Squares   F-statistic:
229.2
Date:                Fri, 31 Jan 2025   Prob (F-statistic):
0.00
Time:                        22:33:25   Log-Likelihood:                         -1
78.31
No. Observations:                1022   AIC:
450.6
Df Residuals:                     975   BIC:
682.3
Df Model:                          47
Covariance Type:            nonrobust
==================================================================================
=======
                            coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------------------
-------
LotArea                 4.07e-06   1.18e-06      3.437      0.001    1.75e-06
6.39e-06
OverallQual               0.1785      0.013     14.222      0.000       0.154
0.203
OverallCond               0.0778      0.011      6.878      0.000       0.056
0.100
YearBuilt                -0.0006      0.001     -0.980      0.327      -0.002
0.001
YearRemodAdd             -0.0017      0.001     -2.702      0.007      -0.003
-0.000
BsmtFinSF1                0.0001   3.17e-05      4.544      0.000    8.18e-05
0.000
TotalBsmtSF               0.0004    3.7e-05     10.356      0.000       0.000
0.000
GrLivArea                 0.0004    4.6e-05      9.668      0.000       0.000
0.001
BsmtFullBath              0.0457      0.025      1.813      0.070      -0.004
0.095
HalfBath                  0.0615      0.025      2.428      0.015       0.012
0.111
TotRmsAbvGrd              0.0153      0.011      1.405      0.160      -0.006
0.037
GarageArea                0.0006   5.79e-05      9.987      0.000       0.000
0.001
WoodDeckSF                0.0003   7.97e-05      3.320      0.001       0.000
0.000
ScreenPorch               0.0007      0.000      3.990      0.000       0.000
0.001
cnvrt_BsmtExposure        0.0357      0.010      3.464      0.001       0.015
0.056
```

cnvrt_HeatingQC	0.0722	0.013	5.690	0.000	0.047	0.097
cnvrt_KitchenQual	0.1185	0.022	5.492	0.000	0.076	0.161
cnvrt_Functional	0.0829	0.016	5.226	0.000	0.052	0.114
MSSubClass_30	-0.3736	0.053	-7.042	0.000	-0.478	-0.269
MSSubClass_45	-0.7139	0.231	-3.084	0.002	-1.168	-0.260
MSSubClass_60	0.1530	0.036	4.199	0.000	0.081	0.224
MSZoning_C (all)	-1.0558	0.117	-9.027	0.000	-1.285	-0.826
Alley_Grvl	-0.2381	0.056	-4.263	0.000	-0.348	-0.129
Utilities_NoSeWa	-0.4121	0.303	-1.358	0.175	-1.007	0.183
LotConfig_CulDSac	0.1168	0.041	2.847	0.005	0.036	0.197
Neighborhood_ClearCr	0.2437	0.075	3.264	0.001	0.097	0.390
Neighborhood_Crawfor	0.3272	0.055	5.936	0.000	0.219	0.435
Neighborhood_MeadowV	-0.4941	0.081	-6.128	0.000	-0.652	-0.336
Neighborhood_NoRidge	-0.0067	0.067	-0.101	0.920	-0.137	0.124
Neighborhood_NridgHt	0.1953	0.047	4.135	0.000	0.103	0.288
Neighborhood_SawyerW	0.0277	0.052	0.535	0.593	-0.074	0.129
Neighborhood_StoneBr	0.1660	0.077	2.161	0.031	0.015	0.317
Neighborhood_Veenker	0.1894	0.110	1.727	0.084	-0.026	0.404
Condition1_Artery	-0.2857	0.055	-5.239	0.000	-0.393	-0.179
Condition1_Feedr	-0.1105	0.042	-2.622	0.009	-0.193	-0.028
Condition1_RRAe	-0.2691	0.102	-2.638	0.008	-0.469	-0.069
Condition2_PosN	-2.1949	0.217	-10.094	0.000	-2.622	-1.768
HouseStyle_1.5Unf	0.5104	0.215	2.371	0.018	0.088	0.933
HouseStyle_2.5Unf	0.1061	0.119	0.894	0.371	-0.127	0.339
Heating_Wall	0.3375	0.181	1.861	0.063	-0.018	0.693
RoofMatl_ClyTile	-6.6499	0.355	-18.744	0.000	-7.346	-5.954
Exterior1st_BrkComm	-1.5693	0.302	-5.201	0.000	-2.161	-0.977
Foundation_BrkTil	-0.2913	0.043	-6.758	0.000	-0.376	-0.207
Foundation_CBlock	-0.1489	0.027	-5.514	0.000	-0.202	-0.096
CentralAir_N	-0.3153	0.048	-6.508	0.000	-0.410	-0.220
SaleType_ConLD	0.5385	0.160	3.367	0.001	0.225	0.852
SaleCondition_Abnorml	-0.1402	0.038	-3.732	0.000	-0.214	-0.066

```
==============================================================================
Omnibus:                      170.331   Durbin-Watson:                   1.956
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1051.554
Skew:                          -0.602   Prob(JB):                    4.55e-229
Kurtosis:                       7.821   Cond. No.                     5.67e+05
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain
a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
[3] The condition number is large, 5.67e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
Dropping non-significant column: const with p-value: 0.9982652699058888
Dropping non-significant column: GarageYrBlt with p-value: 0.9902556887454413
Dropping non-significant column: SaleType_ConLw with p-value: 0.986673392627895
Dropping non-significant column: Condition2_RRNn with p-value: 0.9803529657510406
Dropping non-significant column: cnvrt_LandSlope with p-value: 0.9737641968210782
Dropping non-significant column: Condition1_RRAn with p-value: 0.9625698041603705
Dropping non-significant column: Condition1_PosN with p-value: 0.9516429165290141
Dropping non-significant column: RoofStyle_Flat with p-value: 0.9470354473550069
Dropping non-significant column: Heating_OthW with p-value: 0.9393113859706568
Dropping non-significant column: HouseStyle_SFoyer with p-value: 0.92171724850102
Dropping non-significant column: Exterior1st_WdShing with p-value: 0.9140557220380
436
Dropping non-significant column: Exterior1st_AsbShng with p-value: 0.9216664929257
257
Dropping non-significant column: Exterior1st_ImStucc with p-value: 0.9133674285800
941
Dropping non-significant column: Exterior2nd_Brk Cmn with p-value: 0.9102675557078
718
Dropping non-significant column: MiscFeature_Shed with p-value: 0.8998650920988018
Dropping non-significant column: Condition1_RRNe with p-value: 0.8841202464784453
Dropping non-significant column: RoofMatl_Metal with p-value: 0.8760071468065673
Dropping non-significant column: GarageType_BuiltIn with p-value: 0.87646712373698
25
Dropping non-significant column: cnvrt_PoolQC with p-value: 0.8739095551579034
Dropping non-significant column: SaleCondition_AdjLand with p-value: 0.86227359150
17716
Dropping non-significant column: Exterior2nd_AsphShn with p-value: 0.8574240122230
874
Dropping non-significant column: cnvrt_LotShape with p-value: 0.831930192147754
Dropping non-significant column: BldgType_TwnhsE with p-value: 0.834550600504326
Dropping non-significant column: SaleType_Oth with p-value: 0.8295121501445334
Dropping non-significant column: cnvrt_ExterQual with p-value: 0.8208193358569102
Dropping non-significant column: MasVnrType_BrkFace with p-value: 0.82268176849742
08
Dropping non-significant column: Exterior2nd_Stone with p-value: 0.770108598278568
1
Dropping non-significant column: Condition1_RRNn with p-value: 0.7722099105381106
Dropping non-significant column: MSZoning_RH with p-value: 0.7571652803205811
Dropping non-significant column: RoofStyle_Hip with p-value: 0.7168748585849956
Dropping non-significant column: MoSold with p-value: 0.7016444133517559
Dropping non-significant column: BsmtHalfBath with p-value: 0.6977607464902499
Dropping non-significant column: Exterior2nd_HdBoard with p-value: 0.6967718485260
024
Dropping non-significant column: Exterior2nd_Plywood with p-value: 0.8534174116441
02
Dropping non-significant column: Neighborhood_IDOTRR with p-value: 0.6980213547887
526
Dropping non-significant column: MSZoning_RM with p-value: 0.757589291984859
Dropping non-significant column: SaleType_COD with p-value: 0.6518086793483162
Dropping non-significant column: Neighborhood_Mitchel with p-value: 0.645346288384
```

```
4263
Dropping non-significant column: LandContour_HLS with p-value: 0.6469597091415127
Dropping non-significant column: cnvrt_BsmtQual with p-value: 0.6581300575681144
Dropping non-significant column: Electrical_FuseP with p-value: 0.6045414122770248
Dropping non-significant column: BldgType_2fmCon with p-value: 0.5887119800871876
Dropping non-significant column: LowQualFinSF with p-value: 0.6026912697718041
Dropping non-significant column: 2ndFlrSF with p-value: 0.9891318116854159
Dropping non-significant column: 1stFlrSF with p-value: 0.6693968025645225
Dropping non-significant column: RoofMatl_Tar&Grv with p-value: 0.5930183826407616
Dropping non-significant column: Electrical_FuseA with p-value: 0.5785442379168921
Dropping non-significant column: Exterior2nd_MetalSd with p-value: 0.5793487384067
288
Dropping non-significant column: MiscVal with p-value: 0.5600905113867879
Dropping non-significant column: MiscFeature_Gar2 with p-value: 0.6564218660056855
Dropping non-significant column: Alley_Pave with p-value: 0.557936676537815
Dropping non-significant column: Exterior2nd_ImStucc with p-value: 0.5205658452478
366
Dropping non-significant column: BedroomAbvGr with p-value: 0.5448701425479019
Dropping non-significant column: Exterior1st_Stone with p-value: 0.519011417257620
8
Dropping non-significant column: Condition1_PosA with p-value: 0.5930796850504743
Dropping non-significant column: Foundation_Stone with p-value: 0.5066573675833065
Dropping non-significant column: GarageType_CarPort with p-value: 0.50358335597841
55
Dropping non-significant column: GarageCars with p-value: 0.48667553165327304
Dropping non-significant column: MSSubClass_85 with p-value: 0.4838056758075222
Dropping non-significant column: Heating_Grav with p-value: 0.4625367141013741
Dropping non-significant column: RoofStyle_Gambrel with p-value: 0.462836026150349
Dropping non-significant column: HouseStyle_SLvl with p-value: 0.45860261846034567
Dropping non-significant column: MSSubClass_80 with p-value: 0.7230143152081019
Dropping non-significant column: MSSubClass_90 with p-value: 0.4738315981437643
Dropping non-significant column: BldgType_Duplex with p-value: 0.47383159814424225
Dropping non-significant column: LotFrontage with p-value: 0.4320068645990405
Dropping non-significant column: Condition2_Feedr with p-value: 0.4286863191321309
6
Dropping non-significant column: cnvrt_Fence with p-value: 0.4867617552638126
Dropping non-significant column: Neighborhood_Sawyer with p-value: 0.4206552105641
551
Dropping non-significant column: Exterior1st_CBlock with p-value: 0.42078017298879
555
Dropping non-significant column: Exterior2nd_CBlock with p-value: 0.42078017298871
717
Dropping non-significant column: Neighborhood_Blueste with p-value: 0.372212379750
7807
Dropping non-significant column: GarageType_NA with p-value: 0.3663351392666082
Dropping non-significant column: GarageType_Basment with p-value: 0.37538367922074
4
Dropping non-significant column: Exterior2nd_AsbShng with p-value: 0.4175730779924
295
Dropping non-significant column: MasVnrType_Stone with p-value: 0.3711804189895044
Dropping non-significant column: SaleType_New with p-value: 0.36124763249118585
Dropping non-significant column: BsmtUnfSF with p-value: 0.34545486337739817
Dropping non-significant column: HouseStyle_2.5Fin with p-value: 0.340061353907929
46
Dropping non-significant column: PoolArea with p-value: 0.4048436824349043
Dropping non-significant column: Condition2_PosA with p-value: 0.3371598320823539
Dropping non-significant column: EnclosedPorch with p-value: 0.376645665043713
Dropping non-significant column: MasVnrArea with p-value: 0.3469205610599362
Dropping non-significant column: SaleType_CWD with p-value: 0.3140009383440344
Dropping non-significant column: cnvrt_BsmtCond with p-value: 0.3035565112575708
Dropping non-significant column: Neighborhood_Blmngtn with p-value: 0.298880445626
23844
Dropping non-significant column: Neighborhood_NWAmes with p-value: 0.3247607107857
1467
```

```
Dropping non-significant column: Neighborhood_NPkVill with p-value: 0.408642413083
06193
Dropping non-significant column: Exterior1st_MetalSd with p-value: 0.3198816409580
218
Dropping non-significant column: HouseStyle_1.5Fin with p-value: 0.315102208876849
25
Dropping non-significant column: MSSubClass_50 with p-value: 0.6074374164019294
Dropping non-significant column: RoofStyle_Shed with p-value: 0.3875299208623465
Dropping non-significant column: Condition2_RRAe with p-value: 0.3875299208631313
Dropping non-significant column: Neighborhood_Gilbert with p-value: 0.309258610783
21665
Dropping non-significant column: Neighborhood_CollgCr with p-value: 0.307203731012
2265
Dropping non-significant column: Neighborhood_Somerst with p-value: 0.386783178252
0599
Dropping non-significant column: cnvrt_GarageFinish with p-value: 0.29385576801171
875
Dropping non-significant column: SaleType_Con with p-value: 0.27338074076644747
Dropping non-significant column: MSSubClass_180 with p-value: 0.2524183443745507
Dropping non-significant column: Exterior2nd_Other with p-value: 0.258633379738164
5
Dropping non-significant column: Neighborhood_BrkSide with p-value: 0.236596608357
89317
Dropping non-significant column: MSSubClass_190 with p-value: 0.2436897401949199
Dropping non-significant column: MSSubClass_70 with p-value: 0.39935288118231116
Dropping non-significant column: HouseStyle_2Story with p-value: 0.293951933961191
46
Dropping non-significant column: LandContour_Low with p-value: 0.28812087416625654
Dropping non-significant column: LotConfig_FR2 with p-value: 0.26486368987321957
Dropping non-significant column: SaleCondition_Alloca with p-value: 0.238871173881
02416
Dropping non-significant column: 3SsnPorch with p-value: 0.22138347507497752
Dropping non-significant column: Heating_GasW with p-value: 0.2138618151126488
Dropping non-significant column: Neighborhood_SWISU with p-value: 0.21111970739386
59
Dropping non-significant column: RoofMatl_WdShngl with p-value: 0.2140173884212989
4
Dropping non-significant column: LandContour_Bnk with p-value: 0.20697580679487054
Dropping non-significant column: Exterior2nd_Wd Shng with p-value: 0.2017618030959
829
Dropping non-significant column: Neighborhood_Timber with p-value: 0.1956772321356
9538
Dropping non-significant column: Neighborhood_SawyerW with p-value: 0.215224260773
44436
Dropping non-significant column: cnvrt_PavedDrive with p-value: 0.1790683045273980
3
Dropping non-significant column: Heating_Floor with p-value: 0.19762777080897603
Dropping non-significant column: cnvrt_BsmtFinType2 with p-value: 0.21883665551973
405
Dropping non-significant column: BsmtFinSF2 with p-value: 0.45823032684995313
Dropping non-significant column: Fireplaces with p-value: 0.17787368269742998
Dropping non-significant column: cnvrt_GarageCond with p-value: 0.1690509625905961
3
Dropping non-significant column: cnvrt_GarageQual with p-value: 0.4250486622222290
3
Dropping non-significant column: LotConfig_FR3 with p-value: 0.1594111724235155
Dropping non-significant column: RoofMatl_WdShake with p-value: 0.1478611276417940
7
Dropping non-significant column: RoofStyle_Mansard with p-value: 0.262301242879858
Dropping non-significant column: Exterior1st_Plywood with p-value: 0.1423123379702
558
Dropping non-significant column: MSSubClass_75 with p-value: 0.1144172334694386
Dropping non-significant column: HouseStyle_2.5Unf with p-value: 0.228058540413374
56
```

```
Dropping non-significant column: MasVnrType_BrkCmn with p-value: 0.10989777476768482
Dropping non-significant column: Electrical_FuseF with p-value: 0.10813089821757625
Dropping non-significant column: cnvrt_ExterCond with p-value: 0.12407731464952222
Dropping non-significant column: MSSubClass_40 with p-value: 0.11270225490602294
Dropping non-significant column: SaleType_ConLI with p-value: 0.10305687314988009
Dropping non-significant column: Neighborhood_BrDale with p-value: 0.098077253545671
Dropping non-significant column: Condition2_Artery with p-value: 0.08506381086870338
Dropping non-significant column: LotConfig_Corner with p-value: 0.09951266669443959
Dropping non-significant column: SaleCondition_Family with p-value: 0.10370737762440158
Dropping non-significant column: Utilities_NoSeWa with p-value: 0.07925013159451175
Dropping non-significant column: MiscFeature_Othr with p-value: 0.07946401942312567
Dropping non-significant column: Neighborhood_NoRidge with p-value: 0.05712707173523553
Dropping non-significant column: OpenPorchSF with p-value: 0.07487018414860602
Dropping non-significant column: Neighborhood_OldTown with p-value: 0.05460707542570495
Dropping non-significant column: GarageType_2Types with p-value: 0.05889125420821802
Dropping non-significant column: GarageType_Detchd with p-value: 0.06532611718595543
Dropping non-significant column: Exterior2nd_Wd Sdng with p-value: 0.06192063751502011
Dropping non-significant column: Exterior1st_Wd Sdng with p-value: 0.3438826201859365
Dropping non-significant column: Exterior1st_Stucco with p-value: 0.08135200550441114
Dropping non-significant column: Exterior2nd_Stucco with p-value: 0.5346212642961974
Dropping non-significant column: Exterior1st_CemntBd with p-value: 0.08902971711141007
Dropping non-significant column: Exterior2nd_CmentBd with p-value: 0.5972253601103321
Dropping non-significant column: MSSubClass_60 with p-value: 0.053974605909521155
Dropping non-significant column: Foundation_Wood with p-value: 0.06780725058832704
```

In [105…

```
 # X_train.drop(index=[1322,816,970],inplace = True)
y_trainv_current=y_trainv.drop(columns=[non_significant_column])
# X_trainv_current=X_trainv_current.drop(columns=[non_significant_column])
```

In [106…

```
y_trainv_current.shape
```

Out[1060]:

```
(1022,)
```

In [106…

```
X_trainv.shape
```

Out[1061]:

```
(1022, 212)
```

In [106…

```
X_trainv_current.shape
```

Out[1062]:

```
(1022, 60)
```

In [106…

```
X_trainv_current=sm.add_constant(X_trainv_current)
```

In [106…

```
trans_y_trainv_current=PowerTransformer(method='box-cox').fit_transform(y_trainv_cu
```

In [106…

```
#Linear regression - model 1 with Box-Cox transformation and significant columns (t
model_final=sm.OLS(trans_y_trainv_current,X_trainv_current).fit()
print(model_final.summary())
```

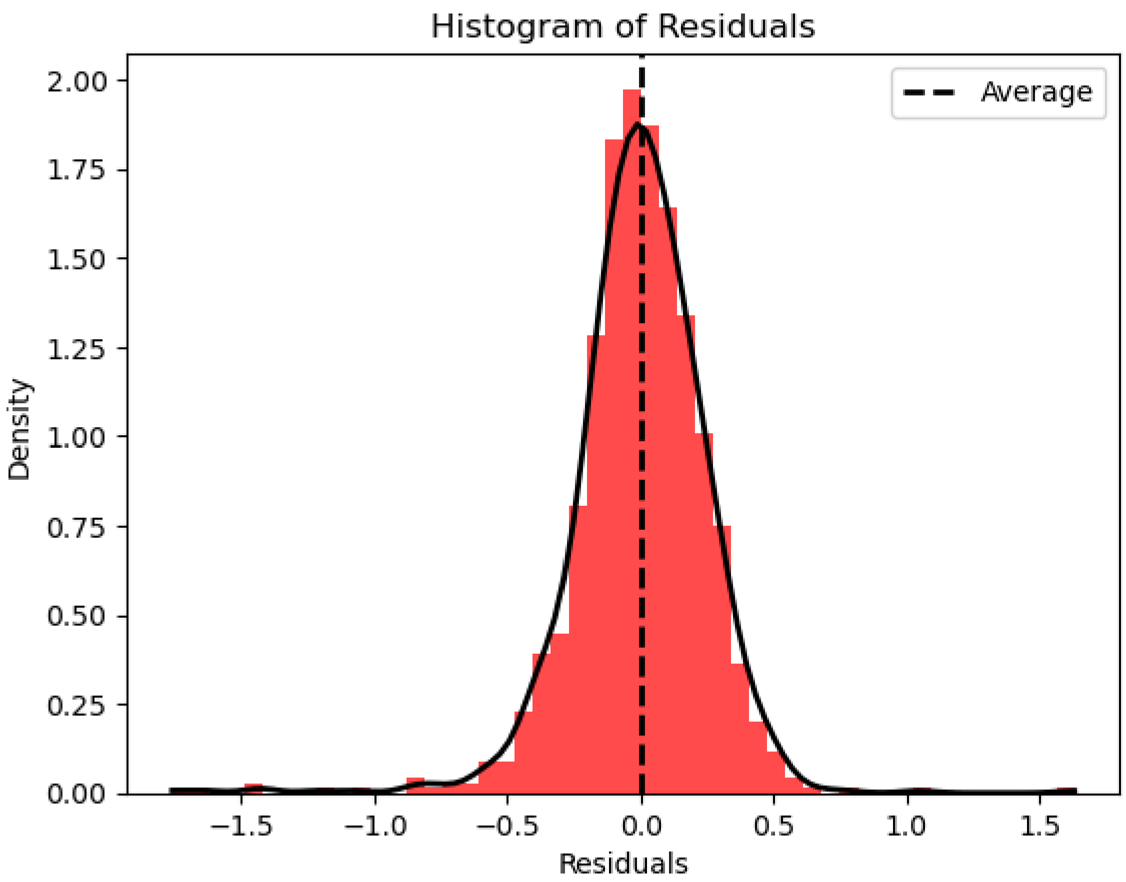
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.937
Model:                            OLS   Adj. R-squared:                  0.933
Method:                 Least Squares   F-statistic:                     236.6
Date:                Fri, 31 Jan 2025   Prob (F-statistic):               0.00
Time:                        22:33:31   Log-Likelihood:                -40.759
No. Observations:                1022   AIC:                             203.5
Df Residuals:                     961   BIC:                             504.2
Df Model:                          60
Covariance Type:            nonrobust
==================================================================================
=======
                            coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------------------
-------
const                    -4.4806     13.075     -0.343      0.732     -30.139
21.178
LotArea                4.224e-06   1.11e-06      3.791      0.000    2.04e-06
6.41e-06
OverallQual               0.1289      0.012     10.892      0.000       0.106
0.152
OverallCond               0.0841      0.010      8.279      0.000       0.064
0.104
YearBuilt                 0.0035      0.001      5.092      0.000       0.002
0.005
YearRemodAdd              0.0023      0.001      3.406      0.001       0.001
0.004
BsmtFinSF1                0.0001   3.26e-05      3.689      0.000    5.63e-05
0.000
TotalBsmtSF               0.0003   3.38e-05      7.782      0.000       0.000
0.000
GrLivArea                 0.0005   4.24e-05     11.759      0.000       0.000
0.001
BsmtFullBath              0.0671      0.023      2.867      0.004       0.021
0.113
FullBath                  0.0528      0.026      2.059      0.040       0.002
0.103
HalfBath                  0.0679      0.023      2.910      0.004       0.022
0.114
KitchenAbvGr             -0.1944      0.046     -4.207      0.000      -0.285
-0.104
TotRmsAbvGrd              0.0261      0.010      2.567      0.010       0.006
0.046
GarageArea                0.0005     5.3e-05      8.597      0.000       0.000
0.001
WoodDeckSF                0.0002   7.13e-05      3.226      0.001    9.01e-05
0.000
ScreenPorch               0.0008      0.000      5.171      0.000       0.000
0.001
YrSold                   -0.0055      0.006     -0.850      0.396      -0.018
0.007
cnvrt_BsmtExposure        0.0253      0.009      2.716      0.007       0.007
0.044
cnvrt_BsmtFinType1        0.0138      0.006      2.168      0.030       0.001
0.026
cnvrt_HeatingQC           0.0400      0.012      3.428      0.001       0.017
0.063
cnvrt_KitchenQual         0.0532      0.020      2.680      0.007       0.014
0.092
cnvrt_Functional          0.0929      0.014      6.429      0.000       0.065
0.121
cnvrt_FireplaceQu         0.0326      0.006      5.470      0.000       0.021
```

						0.044
MSSubClass_120	-0.1368	0.041	-3.302	0.001	-0.218	-0.055
MSSubClass_160	-0.1990	0.057	-3.518	0.000	-0.310	-0.088
MSSubClass_30	-0.3243	0.047	-6.943	0.000	-0.416	-0.233
MSSubClass_45	-0.6217	0.206	-3.014	0.003	-1.026	-0.217
MSZoning_C (all)	-0.8933	0.107	-8.367	0.000	-1.103	-0.684
MSZoning_FV	0.1621	0.046	3.508	0.000	0.071	0.253
Street_Grvl	-0.2953	0.121	-2.447	0.015	-0.532	-0.059
Alley_Grvl	-0.1510	0.050	-3.021	0.003	-0.249	-0.053
LotConfig_CulDSac	0.0811	0.037	2.215	0.027	0.009	0.153
Neighborhood_ClearCr	0.1988	0.068	2.929	0.003	0.066	0.332
Neighborhood_Crawfor	0.2819	0.049	5.727	0.000	0.185	0.378
Neighborhood_Edwards	-0.0770	0.036	-2.160	0.031	-0.147	-0.007
Neighborhood_MeadowV	-0.4429	0.075	-5.882	0.000	-0.591	-0.295
Neighborhood_NridgHt	0.2059	0.043	4.753	0.000	0.121	0.291
Neighborhood_StoneBr	0.2546	0.070	3.643	0.000	0.117	0.392
Neighborhood_Veenker	0.2353	0.100	2.349	0.019	0.039	0.432
Condition1_Artery	-0.2129	0.049	-4.333	0.000	-0.309	-0.116
Condition1_Feedr	-0.1065	0.037	-2.863	0.004	-0.179	-0.034
Condition1_RRAe	-0.2759	0.089	-3.090	0.002	-0.451	-0.101
Condition2_PosN	-2.0842	0.194	-10.736	0.000	-2.465	-1.703
BldgType_Twnhs	-0.1786	0.064	-2.810	0.005	-0.303	-0.054
HouseStyle_1.5Unf	0.5211	0.192	2.716	0.007	0.145	0.898
Heating_Wall	0.5159	0.171	3.021	0.003	0.181	0.851
RoofMatl_ClyTile	-6.0223	0.325	-18.555	0.000	-6.659	-5.385
RoofMatl_Membran	0.5562	0.274	2.030	0.043	0.018	1.094
Exterior1st_BrkComm	-1.3649	0.268	-5.093	0.000	-1.891	-0.839
Exterior1st_BrkFace	0.3217	0.060	5.358	0.000	0.204	0.440
Exterior1st_HdBoard	-0.0641	0.026	-2.499	0.013	-0.115	-0.014
Exterior2nd_BrkFace	-0.3345	0.085	-3.934	0.000	-0.501	-0.168
Foundation_BrkTil	-0.1521	0.042	-3.645	0.000	-0.234	-0.070
Foundation_CBlock	-0.0719	0.026	-2.769	0.006	-0.123	-0.021
Foundation_Slab	-0.1760	0.085	-2.069	0.039	-0.343	

```
-0.009
CentralAir_N              -0.1887      0.044     -4.331      0.000      -0.274
-0.103
MiscFeature_TenC          -0.5697      0.272     -2.092      0.037      -1.104
-0.035
SaleType_ConLD             0.5661      0.142      3.991      0.000       0.288
0.844
SaleCondition_Abnorml     -0.0947      0.034     -2.825      0.005      -0.161
-0.029
SaleCondition_Partial      0.1034      0.036      2.881      0.004       0.033
0.174
==============================================================================
Omnibus:                      277.308   Durbin-Watson:                   1.919
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2935.754
Skew:                          -0.926   Prob(JB):                         0.00
Kurtosis:                      11.094   Cond. No.                     2.36e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
[2] The condition number is large, 2.36e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

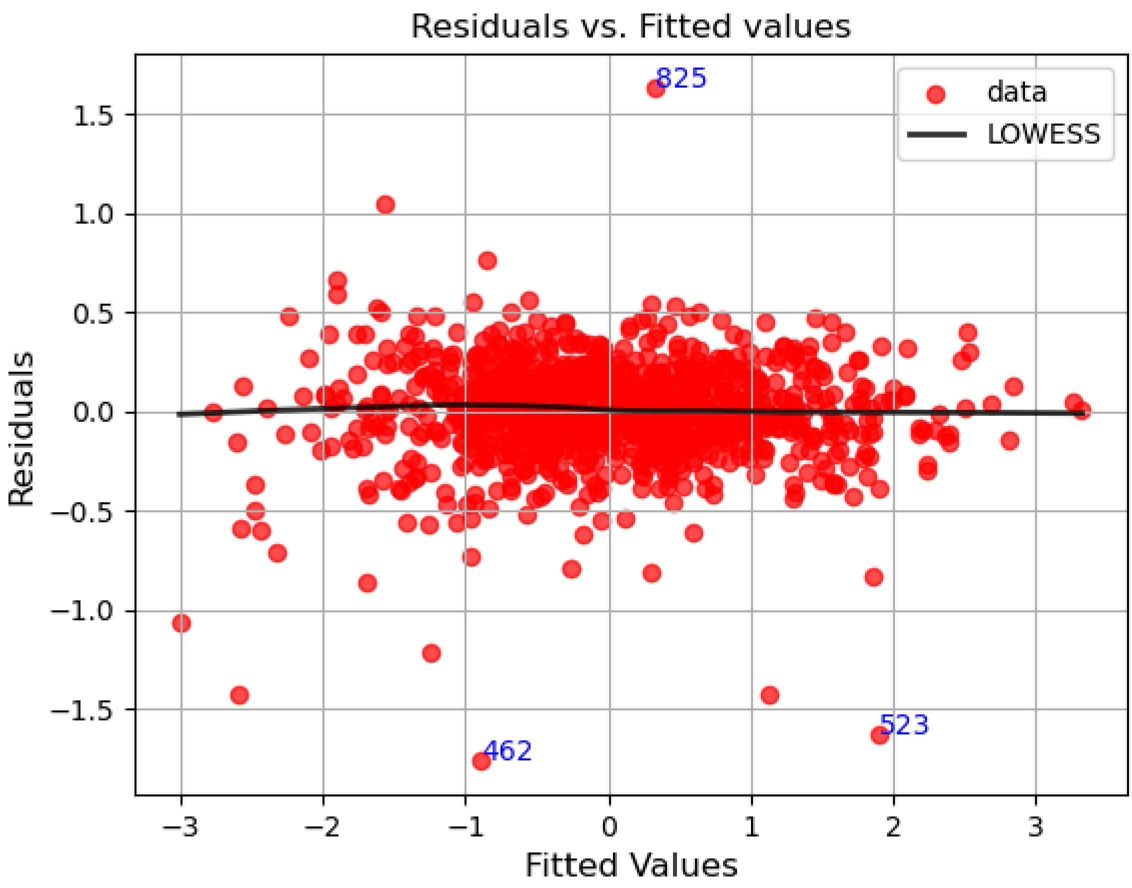
In [106…

```
# Histogram of residuals- mode_final
hist_residuals(model_final)
```



In [106…

```
#Scatter plot of residuals vs. fitted values- mode_final
residuals_fittedvalues_plot(model_final)
```



In [106…
```
# trainv.loc[[825,523,632], :]
```

In [106…
```
# #Check Cook's ditance- model_final
influencer_detector(model_final)
```

```
C:\Users\Taban\anaconda3\Lib\site-packages\statsmodels\stats\outliers_influence.p
y:848: RuntimeWarning: invalid value encountered in sqrt
  return self.resid / sigma / np.sqrt(1 - hii)
```

Out[1069]:
```
(array([114, 265, 416, 599, 829], dtype=int64),)
```

In [107…
```
# trainv.loc[[ 173, 185, 507, 995], :]
```

In [107…
```
# # Filter out variables with high VIF (>10)
# filtered = vif_df[vif_df['VIF'] > 10]
# # filtered_vif_df = vif_df[vif_df['VIF'] <= 10]
# print(filtered)
# # print(filtered_vif_df)
```

Evaluation on the Validation Set (Classic Regression)

In [107…
```
testv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 438 entries, 605 to 622
Data columns (total 100 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Id                 438 non-null    int64
 1   MSSubClass         438 non-null    object
 2   MSZoning           438 non-null    object
 3   LotFrontage        438 non-null    int64
 4   LotArea            438 non-null    int64
 5   Street             438 non-null    object
 6   Alley              438 non-null    object
 7   LotShape           438 non-null    object
 8   LandContour        438 non-null    object
 9   Utilities          438 non-null    object
 10  LotConfig          438 non-null    object
 11  LandSlope          438 non-null    object
 12  Neighborhood       438 non-null    object
 13  Condition1         438 non-null    object
 14  Condition2         438 non-null    object
 15  BldgType           438 non-null    object
 16  HouseStyle         438 non-null    object
 17  OverallQual        438 non-null    int64
 18  OverallCond        438 non-null    int64
 19  YearBuilt          438 non-null    int64
 20  YearRemodAdd       438 non-null    int64
 21  RoofStyle          438 non-null    object
 22  RoofMatl           438 non-null    object
 23  Exterior1st        438 non-null    object
 24  Exterior2nd        438 non-null    object
 25  MasVnrType         438 non-null    object
 26  MasVnrArea         438 non-null    int64
 27  ExterQual          438 non-null    object
 28  ExterCond          438 non-null    object
 29  Foundation         438 non-null    object
 30  BsmtQual           438 non-null    object
 31  BsmtCond           438 non-null    object
 32  BsmtExposure       438 non-null    object
 33  BsmtFinType1       438 non-null    object
 34  BsmtFinSF1         438 non-null    int64
 35  BsmtFinType2       438 non-null    object
 36  BsmtFinSF2         438 non-null    int64
 37  BsmtUnfSF          438 non-null    int64
 38  TotalBsmtSF        438 non-null    int64
 39  Heating            438 non-null    object
 40  HeatingQC          438 non-null    object
 41  CentralAir         438 non-null    object
 42  Electrical         438 non-null    object
 43  1stFlrSF           438 non-null    int64
 44  2ndFlrSF           438 non-null    int64
 45  LowQualFinSF       438 non-null    int64
 46  GrLivArea          438 non-null    int64
 47  BsmtFullBath       438 non-null    int64
 48  BsmtHalfBath       438 non-null    int64
 49  FullBath           438 non-null    int64
 50  HalfBath           438 non-null    int64
 51  BedroomAbvGr       438 non-null    int64
 52  KitchenAbvGr       438 non-null    int64
 53  KitchenQual        438 non-null    object
 54  TotRmsAbvGrd       438 non-null    int64
 55  Functional         438 non-null    object
 56  Fireplaces         438 non-null    int64
 57  FireplaceQu        438 non-null    object
 58  GarageType         438 non-null    object
```

```
 59  GarageYrBlt        438 non-null    int64
 60  GarageFinish       438 non-null    object
 61  GarageCars         438 non-null    int64
 62  GarageArea         438 non-null    int64
 63  GarageQual         438 non-null    object
 64  GarageCond         438 non-null    object
 65  PavedDrive         438 non-null    object
 66  WoodDeckSF         438 non-null    int64
 67  OpenPorchSF        438 non-null    int64
 68  EnclosedPorch      438 non-null    int64
 69  3SsnPorch          438 non-null    int64
 70  ScreenPorch        438 non-null    int64
 71  PoolArea           438 non-null    int64
 72  PoolQC             438 non-null    object
 73  Fence              438 non-null    object
 74  MiscFeature        438 non-null    object
 75  MiscVal            438 non-null    int64
 76  MoSold             438 non-null    int64
 77  YrSold             438 non-null    int64
 78  SaleType           438 non-null    object
 79  SaleCondition      438 non-null    object
 80  SalePrice          438 non-null    int64
 81  cnvrt_LotShape     438 non-null    int64
 82  cnvrt_LandSlope    438 non-null    int64
 83  cnvrt_ExterQual    438 non-null    int64
 84  cnvrt_ExterCond    438 non-null    int64
 85  cnvrt_BsmtQual     438 non-null    int64
 86  cnvrt_BsmtCond     438 non-null    int64
 87  cnvrt_BsmtExposure 438 non-null    int64
 88  cnvrt_BsmtFinType1 438 non-null    int64
 89  cnvrt_BsmtFinType2 438 non-null    int64
 90  cnvrt_HeatingQC    438 non-null    int64
 91  cnvrt_KitchenQual  438 non-null    int64
 92  cnvrt_Functional   438 non-null    int64
 93  cnvrt_FireplaceQu  438 non-null    int64
 94  cnvrt_GarageFinish 438 non-null    int64
 95  cnvrt_GarageQual   438 non-null    int64
 96  cnvrt_GarageCond   438 non-null    int64
 97  cnvrt_PavedDrive   438 non-null    int64
 98  cnvrt_PoolQC       438 non-null    int64
 99  cnvrt_Fence        438 non-null    int64
dtypes: int64(56), object(44)
memory usage: 345.6+ KB
```

In [107…

```
 #Create dummy variables for categorical variables
dummy_vars_tev=pd.get_dummies(testv[['MSSubClass','MSZoning','Street','Alley','Land
                                   'Utilities','LotConfig','Neighborhood','Conditio
                                   'BldgType','HouseStyle','RoofStyle','Heating','F
                                   'Exterior1st','Exterior2nd','MasVnrType','Founda
                                   'Electrical','GarageType','MiscFeature','SaleTyp

# dummy_vars containing only 0 and 1
dummy_vars_tev=dummy_vars_tev.astype(int)

# print(dummy_vars_tev.head(2))
dummy_vars_tev.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 438 entries, 605 to 622
Columns: 158 entries, MSSubClass_120 to SaleCondition_Partial
dtypes: int32(158)
memory usage: 273.8 KB
```

In [107…

```
dummy_vars_trv.shape
```

Out[1074]: (1022, 157)

In [107…
```
dummy_vars_tev.shape
```

Out[1075]: (438, 158)

In [107…
```
# Define base level for each categorical variable Acco8rding to train data set
dummy_vars_tev.drop(columns=['MSSubClass_20','MSZoning_RL','Street_Pave','Alley_NA'
                           'LotConfig_Inside', 'Neighborhood_NAmes', 'Condition1_No
                           'HouseStyle_1Story','RoofStyle_Gable','RoofMatl_CompShg'
                           'MasVnrType_none','Foundation_PConc','CentralAir_Y','Hea
                           'GarageType_Attchd','SaleType_WD','SaleCondition_Normal'
dummy_vars_tev.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 438 entries, 605 to 622
Columns: 133 entries, MSSubClass_120 to SaleCondition_Partial
dtypes: int32(133)
memory usage: 231.0 KB
```

In [107…
```
dummy_vars_tev=dummy_vars_tev.reindex(columns=dummy_vars_trv.columns,fill_value=0)
```

In [107…
```
print(dummy_vars_trv.shape)
print(trainv.shape)


print(dummy_vars_tev.shape)
print(testv.shape)
# print(testv.info)
```

```
(1022, 157)
(1022, 100)
(438, 157)
(438, 100)
```

In [107…
```
# Define feature matrix
# Train All columns except 'SalePrice'
X_X=testv.iloc[:,list(range(0,80))+list(range(81,100))]
# print(X_X.columns)
X_testv=pd.concat([X_X,dummy_vars_tev],axis = 1)
#add constant
# X_testv=sm.add_constant(X_testv)
X_testv = sm.add_constant(X_testv,has_constant='add')
X_testv.shape
print(X_testv.columns)
```

```
Index(['const', 'Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea',
       'Street', 'Alley', 'LotShape', 'LandContour',
       ...
       'SaleType_ConLD', 'SaleType_ConLI', 'SaleType_ConLw', 'SaleType_New',
       'SaleType_Oth', 'SaleCondition_Abnorml', 'SaleCondition_AdjLand',
       'SaleCondition_Alloca', 'SaleCondition_Family',
       'SaleCondition_Partial'],
      dtype='object', length=257)
```

In [108…
```
# Remove coloumn'Id'
X_testv.drop(columns=['Id'],inplace=True)
X_testv.shape
```

Out[1080]: (438, 256)

In [108…
```
X_testv=X_testv.drop(columns=X_testv.select_dtypes(include=['object']).columns)
# Check the data types of the cleaned DataFrame
```

```
print(X_testv.dtypes)
print(X_testv.shape)
```

```
const                      float64
LotFrontage                  int64
LotArea                      int64
OverallQual                  int64
OverallCond                  int64
                            ...
SaleCondition_Abnorml        int32
SaleCondition_AdjLand        int32
SaleCondition_Alloca         int32
SaleCondition_Family         int32
SaleCondition_Partial        int32
Length: 212, dtype: object
(438, 212)
```

In [108…

```
# Convert float64 and int32 columns to int64
X_testv=X_testv.astype({col:'int64' for col in X_testv.select_dtypes(include=['floa

# Check the data types of the columns to verify the conversion
print(X_testv.dtypes)
```

```
const                      int64
LotFrontage                int64
LotArea                    int64
OverallQual                int64
OverallCond                int64
                          ...
SaleCondition_Abnorml      int64
SaleCondition_AdjLand      int64
SaleCondition_Alloca       int64
SaleCondition_Family       int64
SaleCondition_Partial      int64
Length: 212, dtype: object
```

In [108…

```
#Linear regression - model 1 with Box-Cox transformation and significant columns (t
model_final=sm.OLS(trans_y_trainv_current,X_trainv_current).fit()
# print(model_final.summary())
```

In [108…

```
trans_y_trainv_current=PowerTransformer(method = 'box-cox').fit_transform(y_trainv_
```

In [108…

```
X_trainv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1022 entries, 1017 to 815
Columns: 212 entries, const to SaleCondition_Partial
dtypes: int64(212)
memory usage: 1.7 MB
```

In [108…

```
X_testv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 438 entries, 605 to 622
Columns: 212 entries, const to SaleCondition_Partial
dtypes: int64(212)
memory usage: 728.9 KB
```

In [108…

```
trans_y_trainv_current.shape
```

Out[1087]:

```
(1022, 1)
```

In [108…

```
#Linear regression- model 1 with Box-Cox transformation and significant columns (t_
model_final=sm.OLS(trans_y_trainv_current,X_trainv_current).fit()
```

```
print(model_final.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.937
Model:                            OLS   Adj. R-squared:                  0.933
Method:                 Least Squares   F-statistic:                     236.6
Date:                Fri, 31 Jan 2025   Prob (F-statistic):               0.00
Time:                        22:33:33   Log-Likelihood:                -40.759
No. Observations:                1022   AIC:                             203.5
Df Residuals:                     961   BIC:                             504.2
Df Model:                          60
Covariance Type:            nonrobust
==============================================================================
=======
                            coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-------
const                    -4.4806     13.075     -0.343      0.732     -30.139
21.178
LotArea                4.224e-06   1.11e-06      3.791      0.000    2.04e-06
6.41e-06
OverallQual               0.1289      0.012     10.892      0.000       0.106
0.152
OverallCond               0.0841      0.010      8.279      0.000       0.064
0.104
YearBuilt                 0.0035      0.001      5.092      0.000       0.002
0.005
YearRemodAdd              0.0023      0.001      3.406      0.001       0.001
0.004
BsmtFinSF1                0.0001   3.26e-05      3.689      0.000    5.63e-05
0.000
TotalBsmtSF               0.0003   3.38e-05      7.782      0.000       0.000
0.000
GrLivArea                 0.0005   4.24e-05     11.759      0.000       0.000
0.001
BsmtFullBath              0.0671      0.023      2.867      0.004       0.021
0.113
FullBath                  0.0528      0.026      2.059      0.040       0.002
0.103
HalfBath                  0.0679      0.023      2.910      0.004       0.022
0.114
KitchenAbvGr             -0.1944      0.046     -4.207      0.000      -0.285
-0.104
TotRmsAbvGrd              0.0261      0.010      2.567      0.010       0.006
0.046
GarageArea                0.0005     5.3e-05     8.597      0.000       0.000
0.001
WoodDeckSF                0.0002   7.13e-05      3.226      0.001    9.01e-05
0.000
ScreenPorch               0.0008      0.000      5.171      0.000       0.000
0.001
YrSold                   -0.0055      0.006     -0.850      0.396      -0.018
0.007
cnvrt_BsmtExposure        0.0253      0.009      2.716      0.007       0.007
0.044
cnvrt_BsmtFinType1        0.0138      0.006      2.168      0.030       0.001
0.026
cnvrt_HeatingQC           0.0400      0.012      3.428      0.001       0.017
0.063
cnvrt_KitchenQual         0.0532      0.020      2.680      0.007       0.014
0.092
cnvrt_Functional          0.0929      0.014      6.429      0.000       0.065
0.121
cnvrt_FireplaceQu         0.0326      0.006      5.470      0.000       0.021
```

```
0.044
MSSubClass_120          -0.1368      0.041     -3.302      0.001      -0.218
-0.055
MSSubClass_160          -0.1990      0.057     -3.518      0.000      -0.310
-0.088
MSSubClass_30           -0.3243      0.047     -6.943      0.000      -0.416
-0.233
MSSubClass_45           -0.6217      0.206     -3.014      0.003      -1.026
-0.217
MSZoning_C (all)        -0.8933      0.107     -8.367      0.000      -1.103
-0.684
MSZoning_FV              0.1621      0.046      3.508      0.000       0.071
0.253
Street_Grvl             -0.2953      0.121     -2.447      0.015      -0.532
-0.059
Alley_Grvl              -0.1510      0.050     -3.021      0.003      -0.249
-0.053
LotConfig_CulDSac        0.0811      0.037      2.215      0.027       0.009
0.153
Neighborhood_ClearCr     0.1988      0.068      2.929      0.003       0.066
0.332
Neighborhood_Crawfor     0.2819      0.049      5.727      0.000       0.185
0.378
Neighborhood_Edwards    -0.0770      0.036     -2.160      0.031      -0.147
-0.007
Neighborhood_MeadowV    -0.4429      0.075     -5.882      0.000      -0.591
-0.295
Neighborhood_NridgHt     0.2059      0.043      4.753      0.000       0.121
0.291
Neighborhood_StoneBr     0.2546      0.070      3.643      0.000       0.117
0.392
Neighborhood_Veenker     0.2353      0.100      2.349      0.019       0.039
0.432
Condition1_Artery       -0.2129      0.049     -4.333      0.000      -0.309
-0.116
Condition1_Feedr        -0.1065      0.037     -2.863      0.004      -0.179
-0.034
Condition1_RRAe         -0.2759      0.089     -3.090      0.002      -0.451
-0.101
Condition2_PosN         -2.0842      0.194    -10.736      0.000      -2.465
-1.703
BldgType_Twnhs          -0.1786      0.064     -2.810      0.005      -0.303
-0.054
HouseStyle_1.5Unf        0.5211      0.192      2.716      0.007       0.145
0.898
Heating_Wall             0.5159      0.171      3.021      0.003       0.181
0.851
RoofMatl_ClyTile        -6.0223      0.325    -18.555      0.000      -6.659
-5.385
RoofMatl_Membran         0.5562      0.274      2.030      0.043       0.018
1.094
Exterior1st_BrkComm     -1.3649      0.268     -5.093      0.000      -1.891
-0.839
Exterior1st_BrkFace      0.3217      0.060      5.358      0.000       0.204
0.440
Exterior1st_HdBoard     -0.0641      0.026     -2.499      0.013      -0.115
-0.014
Exterior2nd_BrkFace     -0.3345      0.085     -3.934      0.000      -0.501
-0.168
Foundation_BrkTil       -0.1521      0.042     -3.645      0.000      -0.234
-0.070
Foundation_CBlock       -0.0719      0.026     -2.769      0.006      -0.123
-0.021
Foundation_Slab         -0.1760      0.085     -2.069      0.039      -0.343
```

```
-0.009
CentralAir_N             -0.1887      0.044     -4.331      0.000      -0.274
-0.103
MiscFeature_TenC         -0.5697      0.272     -2.092      0.037      -1.104
-0.035
SaleType_ConLD            0.5661      0.142      3.991      0.000       0.288
0.844
SaleCondition_Abnorml    -0.0947      0.034     -2.825      0.005      -0.161
-0.029
SaleCondition_Partial     0.1034      0.036      2.881      0.004       0.033
0.174
==============================================================================
Omnibus:                      277.308   Durbin-Watson:                   1.919
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2935.754
Skew:                          -0.926   Prob(JB):                         0.00
Kurtosis:                      11.094   Cond. No.                     2.36e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly s
pecified.
[2] The condition number is large, 2.36e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [108…

```
X_trainv_current
```

Out[1089]:

	const	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	BsmtFinSF1	TotalBsn
1017	1.0	5814	8	5	1984	1984	1036	
405	1.0	9991	4	4	1976	1993	1116	
6	1.0	10084	8	5	2004	2005	1369	
388	1.0	9382	7	5	1999	2000	0	
501	1.0	9803	7	5	2005	2005	400	
...	...	...	...	...	...	...	...	
1228	1.0	8769	9	5	2008	2008	1540	
1077	1.0	15870	5	5	1969	1969	75	
1318	1.0	14781	8	5	2001	2002	0	
723	1.0	8172	4	6	1954	1972	0	
815	1.0	12137	7	5	1998	1998	0	

1022 rows × 61 columns

In [109…

```
X_testv.shape
```

Out[1090]:

```
(438, 212)
```

In [109…

```
# Ensure X_testv only contains the significant columns
X_testv_significant=X_testv[X_trainv_current.columns]

# Predict using the final model
testv_pred=model_final.predict(X_testv_significant)

# Output the predictions
print(testv_pred)
```

```
605     0.369858
642     1.810056
993     0.159041
736    -1.895730
1239    0.768088
          ...
805     0.427172
112     1.882165
348     0.193731
205    -0.000803
622    -0.586256
Length: 438, dtype: float64
```

In [109…

```
# X_testv_significant
```

In [109…

```
# X_testv_current=X_testv.drop(columns=[non_significant_column])
```

In [109…

```
# X_testv_current.info()
```

In [109…

```
# Predict using the final model
testv_pred=model_final.predict(X_testv_significant)

# Output the predictions
print(testv_pred)
```

```
605     0.369858
642     1.810056
993     0.159041
736    -1.895730
1239    0.768088
          ...
805     0.427172
112     1.882165
348     0.193731
205    -0.000803
622    -0.586256
Length: 438, dtype: float64
```

In [109…

```
# Inverse transformation of predicted values (Linear Model)
testv['pred_lm']=pd.Series(boxcox.inverse_transform(testv_pred.values.reshape(-1, 1
                         index = testv_pred.index)
testv['pred_lm']
```

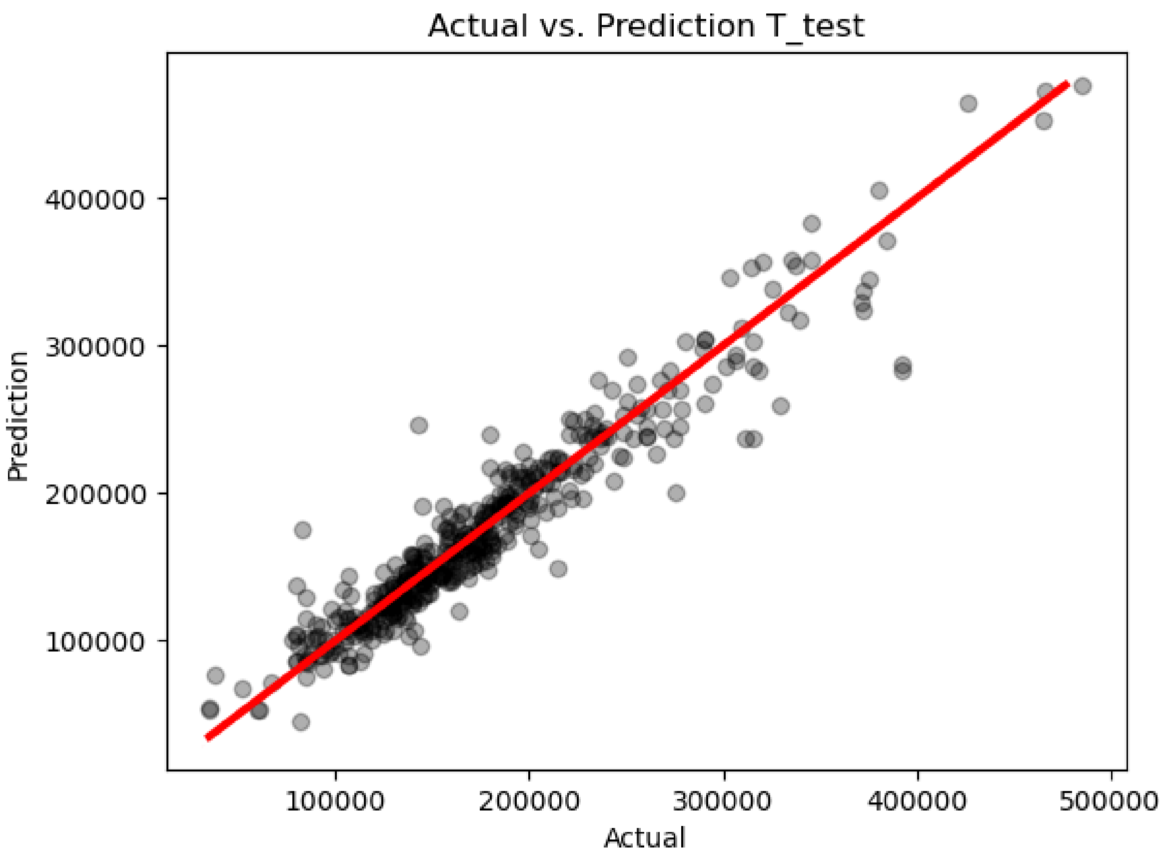
Out[1096]:

```
605     192167.948934
642     358727.495110
993     176337.599023
736      81070.141268
1239    226860.671459
            ...
805     196755.954753
112     370798.570874
348     178834.246434
205     165349.142918
622     131410.946240
Name: pred_lm, Length: 438, dtype: float64
```

In [109…

```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y = testv['pred_lm'],
            c='black',alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction T_test')
```

```
#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),testv['pred_lm'].max(),100)
plt.plot(xp,xp,c ='red',linewidth = 3)
plt.show()
```



In [109…

```
from scipy.stats import iqr
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Absolute error
abs_err_lm=abs(testv['SalePrice']-testv['pred_lm'])

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae=mean_absolute_error(testv['SalePrice'],testv['pred_lm'])
mse=mean_squared_error(testv['SalePrice'],testv['pred_lm'])
rmse=np.sqrt(mse)

# Display the models comparison DataFrame
models_comp=pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_lm.mean(),
                         'MSE' :                   mse,
                         'RMSE' :                  rmse,
                         'Median of AbsErrors' : abs_err_lm.median(),
                         'SD of AbsErrors' :     abs_err_lm.std(),
                         'IQR of AbsErrors':     iqr(abs_err_lm),
                         'Min of AbsErrors':     abs_err_lm.min(),
                         'Max of AbsErrors':     abs_err_lm.max()},
                        index = ['LM_t-Test'])
models_comp
```

Out[1098]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR of AbsErrors	A
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.086048	

Forward Selection

In [109…
```
# Define function to fit linear regression
def fit_lm(feature_set,y,X):
    reg_model=sm.OLS(y, X[['const']+list(feature_set)]).fit()
    return{'model':reg_model,
           'r2':reg_model.rsquared,
           'adj_r2':reg_model.rsquared_adj,
           'aic':reg_model.aic,
           'bic':reg_model.bic}
```

In [110…
```
# Define function to do forward selection
def fwd_selection(features, y, X):
    res = []
    # Pull out features still needed to process
    remaining_features = [ _ for _  in X.iloc[:, 1:].columns if _ not in features]

    # Fit linear model and save the results
    for f in remaining_features:
        res.append(fit_lm(features+[f], y, X))

    models=pd.DataFrame(res)

    # Choose the model with the highest R squared
    best_model=models.iloc[models['r2'].argmax()]

    # Return the best model
    return best_model
```

In [110…
```
X_trainv.shape
```

Out[1101]:
```
(1022, 212)
```

In [110…
```
len(trans_y_trainv)
```

Out[1102]:
```
1022
```

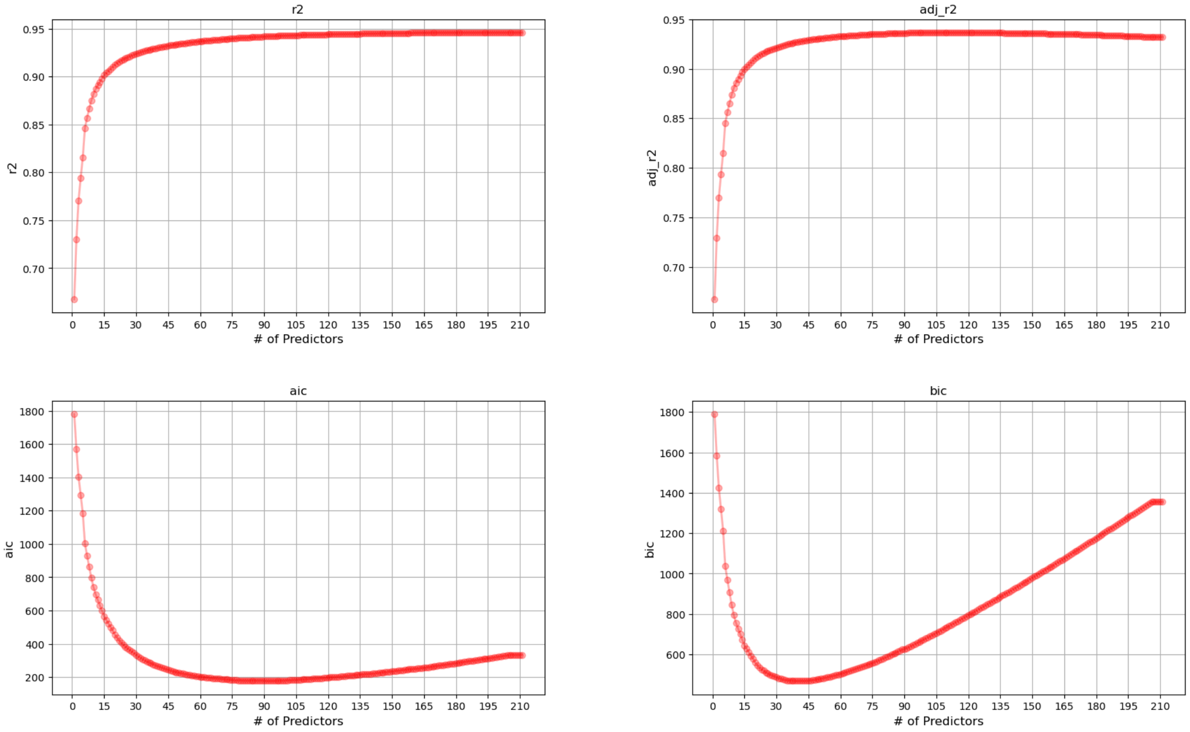
In [110…
```
#Forward selection implementation
import time #to measure the processing time
fwd_models=pd.DataFrame(columns=['model','r2','adj_r2','aic','bic'])
start_time=time.time()
features=[]
for i in range(1,len(X_trainv.iloc[:, 1:].columns)+1):
    fwd_models.loc[i]=fwd_selection(features,trans_y_trainv,X_trainv)
    features=fwd_models.loc[i,'model'].model.exog_names[1:]
end_time=time.time()
print('The Processing time is: ',end_time-start_time,'seconds')
```

```
The Processing time is:  315.2712743282318 seconds
```

Evaluation on the Validation Set (Forward selection)

In [110…
```
#Models evaluation
plt.figure(figsize = (20,12))
plt.subplots_adjust(hspace = 0.3,wspace = 0.3)
for i in range(1,5):
    plt.subplot(2,2,i)
    plt.plot(fwd_models.iloc[:, i],'r-o',alpha=0.3,linewidth=2,)
    plt.title(fwd_models.columns[i])
```

```python
    plt.xlabel('# of Predictors',fontsize=12)
    plt.xticks(range(0,214,15))
    plt.ylabel(fwd_models.columns[i],fontsize=12)
    plt.grid(True)
# Show the plot
plt.show()
```



In [110…

```python
#Liner model with 105 Variables (based on r2 and observing other 3 plots )
fwd_models.loc[105,'model'].params
```

Out[1105]:

```
const                      -14.873968
OverallQual                  0.119426
GrLivArea                    0.000454
YearBuilt                    0.003595
OverallCond                  0.085502
                               ...
Neighborhood_OldTown        -0.062657
Neighborhood_Timber          0.062409
SaleCondition_Alloca        -0.116266
Neighborhood_SawyerW         0.049366
Exterior2nd_AsbShng         -0.079085
Length: 106, dtype: float64
```

In [110…

```python
#Selected features
fwd_models.loc[105,'model'].model.exog_names
```

```
Out[1106]: ['const',
 'OverallQual',
 'GrLivArea',
 'YearBuilt',
 'OverallCond',
 'RoofMatl_ClyTile',
 'TotalBsmtSF',
 'Fireplaces',
 'GarageArea',
 'Condition2_PosN',
 'BsmtUnfSF',
 'MSSubClass_160',
 'MSZoning_C (all)',
 'MSZoning_RM',
 'YearRemodAdd',
 'cnvrt_Functional',
 'Neighborhood_Crawfor',
 'Exterior1st_BrkComm',
 'Neighborhood_ClearCr',
 'MSSubClass_30',
 'cnvrt_HeatingQC',
 'KitchenAbvGr',
 'SaleCondition_Partial',
 'Neighborhood_MeadowV',
 'ScreenPorch',
 'WoodDeckSF',
 'Exterior1st_BrkFace',
 'SaleType_ConLD',
 'Condition1_Artery',
 'CentralAir_N',
 'Neighborhood_NridgHt',
 'cnvrt_BsmtExposure',
 'MSZoning_FV',
 'Neighborhood_StoneBr',
 'LotArea',
 'SaleCondition_Abnorml',
 'Street_Grvl',
 'Exterior2nd_BrkFace',
 'Heating_Wall',
 'cnvrt_BsmtQual',
 'BsmtFullBath',
 'Foundation_Stone',
 'BldgType_Twnhs',
 'BldgType_TwnhsE',
 'Condition1_RRAe',
 'Exterior1st_MetalSd',
 'cnvrt_KitchenQual',
 'Condition1_Feedr',
 'Alley_Grvl',
 'MiscFeature_TenC',
 'TotRmsAbvGrd',
 'Neighborhood_Veenker',
 'Neighborhood_Edwards',
 'LandContour_Bnk',
 'Neighborhood_Mitchel',
 'LotConfig_CulDSac',
 'cnvrt_BsmtFinType1',
 'Exterior2nd_HdBoard',
 'Exterior2nd_Wd Shng',
 'Exterior2nd_MetalSd',
 'Neighborhood_SWISU',
 'GarageType_Detchd',
 'cnvrt_FireplaceQu',
 'Utilities_NoSeWa',
```

```
 'OpenPorchSF',
 'Neighborhood_NoRidge',
 'Exterior2nd_Plywood',
 'MasVnrType_BrkCmn',
 'RoofStyle_Mansard',
 'Neighborhood_BrkSide',
 'Foundation_BrkTil',
 'Foundation_CBlock',
 'Foundation_Slab',
 'MSSubClass_60',
 'HalfBath',
 'FullBath',
 'Foundation_Wood',
 'SaleCondition_Family',
 'RoofMatl_WdShngl',
 'RoofMatl_Membran',
 'cnvrt_BsmtFinType2',
 'Heating_GasW',
 'LotConfig_Corner',
 'SaleType_ConLI',
 'Condition2_PosA',
 'cnvrt_ExterCond',
 'MiscFeature_Othr',
 'MSSubClass_40',
 'RoofMatl_WdShake',
 'MSSubClass_45',
 'HouseStyle_1.5Unf',
 'Condition2_Artery',
 'Electrical_FuseF',
 'HouseStyle_SFoyer',
 'LotConfig_FR3',
 'Condition2_Feedr',
 'LotConfig_FR2',
 '3SsnPorch',
 'GarageType_2Types',
 'Neighborhood_BrDale',
 'Condition2_RRAe',
 'Neighborhood_OldTown',
 'Neighborhood_Timber',
 'SaleCondition_Alloca',
 'Neighborhood_SawyerW',
 'Exterior2nd_AsbShng']
```

In [110…

```
#Predict on test- model 2
pred_fwd=fwd_models.loc[105, 'model'].predict(X_testv[fwd_models.loc[105,'model'].m
pred_fwd=pd.Series(boxcox.inverse_transform(pred_fwd.values.reshape(-1, 1)).reshape
# pred_fwd
```

In [110…

```
len(pred_fwd)
```

Out[1108]:

```
438
```

In [110…

```
len(testv['SalePrice'])
```

Out[1109]:

```
438
```

In [111…

```
#Absolute error
abs_err_fwd=abs(testv['SalePrice']-pred_fwd)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_fwd =mean_absolute_error(testv['SalePrice'], pred_fwd)
mse_fwd=mean_squared_error(testv['SalePrice'], pred_fwd)
rmse_fwd=np.sqrt(mse_fwd)
```

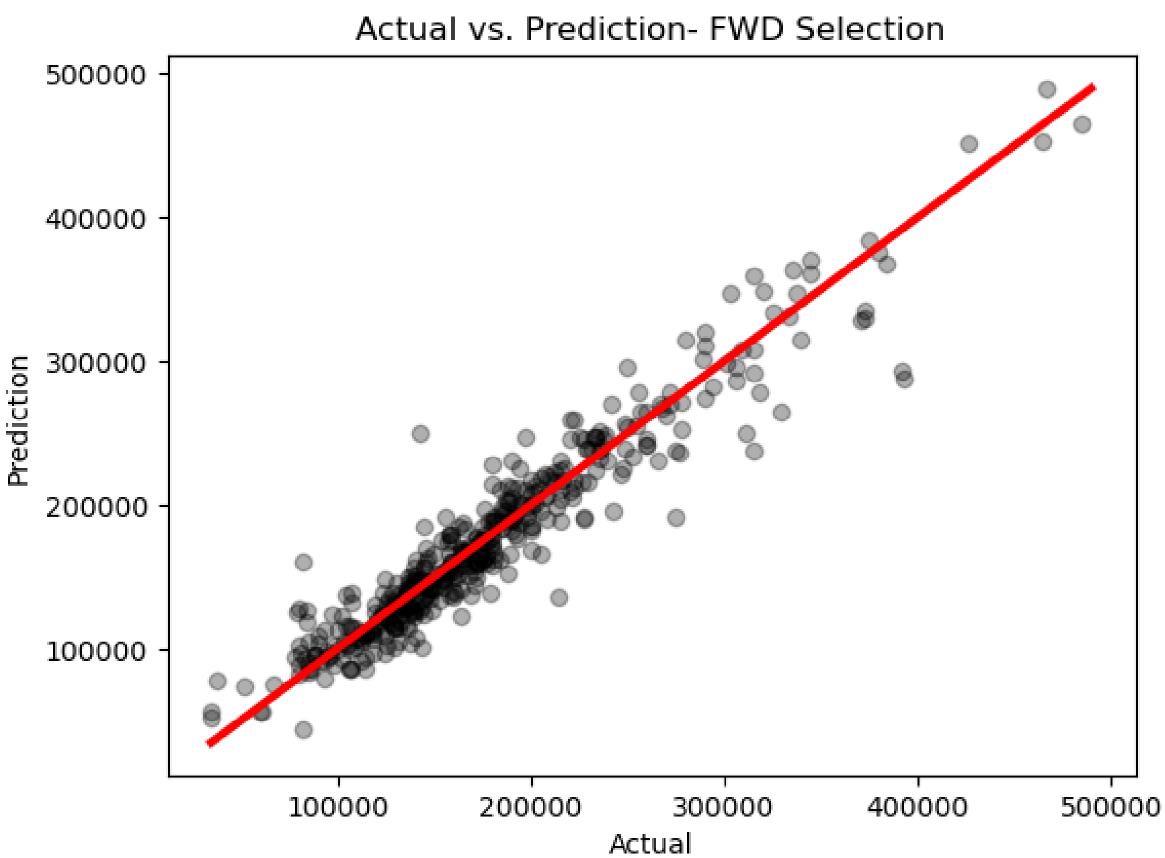
```python
#Absolute error mean, median, sd, IQR, max, min
models_comp = pd.concat([models_comp,
                         pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_fwd.mean()
                            'MSE' :                     mse_fwd,
                            'RMSE' :                   rmse_fwd,
                            'Median of AbsErrors' : abs_err_fwd.median(),
                            'SD of AbsErrors' :     abs_err_fwd.std(),
                            'IQR of AbsErrors':     iqr(abs_err_fwd),
                            'Min of AbsErrors':     abs_err_fwd.min(),
                            'Max of AbsErrors':     abs_err_fwd.max()},
                            index = ['LM_FWD'])])
models_comp
```

Out[1110]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR of AbsErrors
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.086048
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.043120

In [111…

```python
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y = pred_fwd,
            c='black',alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- FWD Selection')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_fwd.max(),100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



Backward Elimination

In [111…

```
import itertools

#Define function to do backward elimination
def bwd_elimination(features,y,X):
    res = []

    #Fit linear model and save the results
    for f in itertools.combinations(features,len(features)-1):
        res.append(fit_lm(f,y,X))

    models=pd.DataFrame(res)

    #Choose the model with the highest R squared
    best_model=models.iloc[models['r2'].argmax()]

    #Return the best model
    return best_model
```

In [111…

```
X_trainv.shape
```

Out[1113]:

```
(1022, 212)
```

In [111…

```
#Backward elimination implementation
bwd_models=pd.DataFrame(columns=['model','r2','adj_r2','aic','bic'])
start_time=time.time()
features=X_trainv.columns
while(len(features)>1):
    bwd_models.loc[len(features)-1]=bwd_elimination(features,trans_y_trainv,X_trair
    features=bwd_models.loc[len(features)-1]['model'].model.exog_names[1:]
end_time=time.time()
print('The Processing time is: ',end_time-start_time,'seconds')
```

```
The Processing time is:  661.1729784011841 seconds
```

In [111…

```
bwd_models
```

Out[1115]:

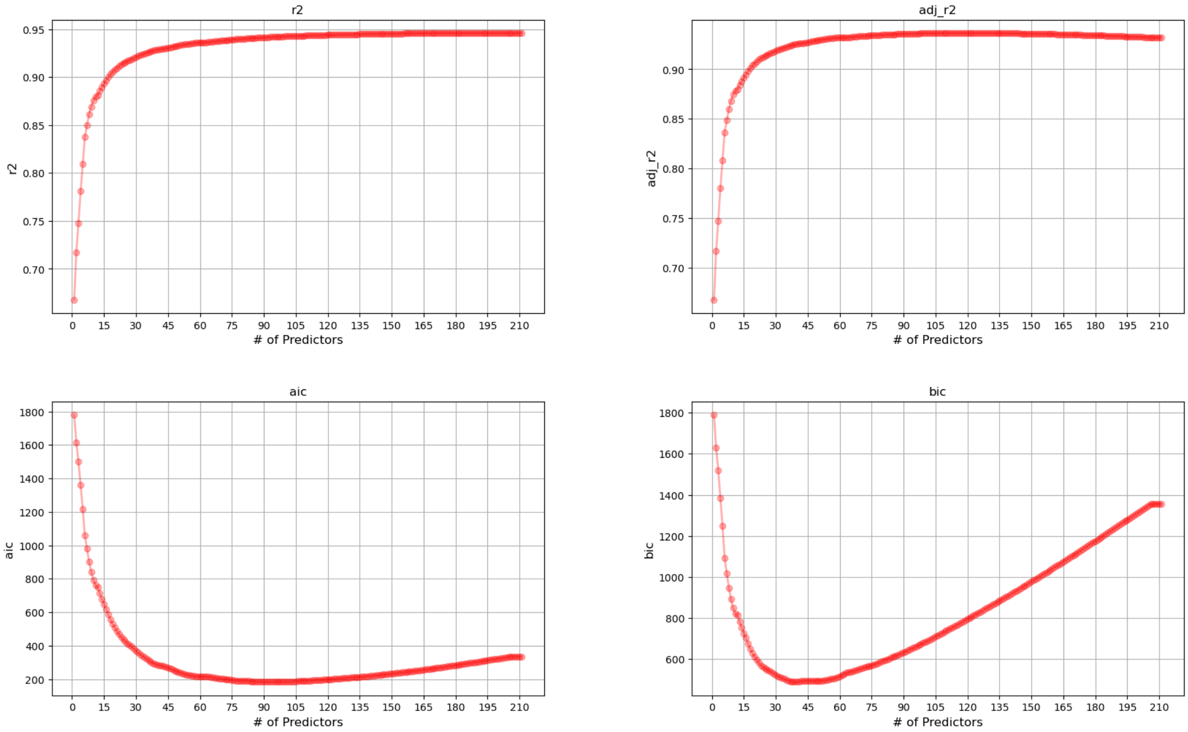
	model	r2	adj_r2	aic	bic
211	<statsmodels.regression.linear_model.Regressio...	0.945823	0.93213	334.665265	1355.075237
210	<statsmodels.regression.linear_model.Regressio...	0.945823	0.93213	334.665265	1355.075237
209	<statsmodels.regression.linear_model.Regressio...	0.945823	0.93213	334.665265	1355.075237
208	<statsmodels.regression.linear_model.Regressio...	0.945823	0.93213	334.665265	1355.075237
207	<statsmodels.regression.linear_model.Regressio...	0.945823	0.93213	334.665265	1355.075237
...	...	...	...	...	...
5	<statsmodels.regression.linear_model.Regressio...	0.809312	0.808373	1218.73921	1248.316311
4	<statsmodels.regression.linear_model.Regressio...	0.780783	0.77992	1359.229428	1383.877012
3	<statsmodels.regression.linear_model.Regressio...	0.747674	0.74693	1500.983547	1520.701615
2	<statsmodels.regression.linear_model.Regressio...	0.7174	0.716845	1614.786256	1629.574806
1	<statsmodels.regression.linear_model.Regressio...	0.667358	0.667032	1779.407768	1789.266801

211 rows × 5 columns

Evaluation on the Validation Set (Backward Elimination)

In [111…

```
#Models evaluation
plt.figure(figsize = (20,12))
plt.subplots_adjust(hspace = 0.3, wspace = 0.3)
for i in range(1, 5):
    plt.subplot(2, 2, i)
    plt.plot(bwd_models.iloc[:, i], 'r-o', alpha = 0.3, linewidth=2)
    plt.title(bwd_models.columns[i])
    plt.xlabel('# of Predictors', fontsize=12)
    plt.xticks(range(0, 214,15))
    plt.ylabel(bwd_models.columns[i],fontsize=12)
    plt.grid(True)
# Show the plot
plt.show()
```



In [111…

```
#Liner model with 120 Variables (based on r2)
bwd_models.loc[120,'model'].params
```

Out[1117]:

```
const                     -13.818466
LotArea                     0.000005
OverallQual                 0.117600
OverallCond                 0.087608
YearBuilt                   0.002715
                             ...
SaleType_ConLI             -0.239721
SaleCondition_Abnorml      -0.071215
SaleCondition_Alloca       -0.147320
SaleCondition_Family       -0.130016
SaleCondition_Partial       0.096610
Length: 121, dtype: float64
```

In [111…

```
#Selected features
bwd_models.loc[120,'model'].model.exog_names
```

Out[1118]:
```
['const',
 'LotArea',
 'OverallQual',
 'OverallCond',
 'YearBuilt',
 'YearRemodAdd',
 'BsmtFinSF1',
 'BsmtFinSF2',
 'BsmtUnfSF',
 '1stFlrSF',
 '2ndFlrSF',
 'LowQualFinSF',
 'BsmtFullBath',
 'FullBath',
 'HalfBath',
 'KitchenAbvGr',
 'TotRmsAbvGrd',
 'Fireplaces',
 'GarageArea',
 'WoodDeckSF',
 'OpenPorchSF',
 '3SsnPorch',
 'ScreenPorch',
 'cnvrt_ExterCond',
 'cnvrt_BsmtCond',
 'cnvrt_BsmtExposure',
 'cnvrt_BsmtFinType1',
 'cnvrt_BsmtFinType2',
 'cnvrt_HeatingQC',
 'cnvrt_KitchenQual',
 'cnvrt_Functional',
 'cnvrt_FireplaceQu',
 'cnvrt_GarageFinish',
 'cnvrt_GarageQual',
 'cnvrt_PavedDrive',
 'MSSubClass_120',
 'MSSubClass_160',
 'MSSubClass_180',
 'MSSubClass_190',
 'MSSubClass_30',
 'MSSubClass_40',
 'MSSubClass_45',
 'MSSubClass_60',
 'MSSubClass_70',
 'MSSubClass_75',
 'MSZoning_C (all)',
 'MSZoning_FV',
 'Street_Grvl',
 'Alley_Grvl',
 'LandContour_Bnk',
 'LandContour_Low',
 'Utilities_NoSeWa',
 'LotConfig_Corner',
 'LotConfig_CulDSac',
 'LotConfig_FR2',
 'LotConfig_FR3',
 'Neighborhood_BrDale',
 'Neighborhood_BrkSide',
 'Neighborhood_ClearCr',
 'Neighborhood_CollgCr',
 'Neighborhood_Crawfor',
 'Neighborhood_Edwards',
 'Neighborhood_Gilbert',
 'Neighborhood_MeadowV',
```

```
 'Neighborhood_NoRidge',
 'Neighborhood_NridgHt',
 'Neighborhood_OldTown',
 'Neighborhood_SWISU',
 'Neighborhood_SawyerW',
 'Neighborhood_Somerst',
 'Neighborhood_StoneBr',
 'Neighborhood_Timber',
 'Neighborhood_Veenker',
 'Condition1_Artery',
 'Condition1_Feedr',
 'Condition1_RRAe',
 'Condition2_Artery',
 'Condition2_PosN',
 'BldgType_Twnhs',
 'HouseStyle_1.5Unf',
 'HouseStyle_2.5Unf',
 'HouseStyle_2Story',
 'RoofStyle_Mansard',
 'Heating_GasW',
 'Heating_Wall',
 'RoofMatl_ClyTile',
 'RoofMatl_Membran',
 'RoofMatl_WdShake',
 'RoofMatl_WdShngl',
 'Exterior1st_BrkComm',
 'Exterior1st_BrkFace',
 'Exterior1st_CemntBd',
 'Exterior1st_HdBoard',
 'Exterior1st_Plywood',
 'Exterior1st_Stucco',
 'Exterior1st_Wd Sdng',
 'Exterior2nd_BrkFace',
 'Exterior2nd_CmentBd',
 'Exterior2nd_Other',
 'Exterior2nd_Stucco',
 'Exterior2nd_Wd Sdng',
 'Exterior2nd_Wd Shng',
 'MasVnrType_BrkCmn',
 'Foundation_BrkTil',
 'Foundation_CBlock',
 'Foundation_Slab',
 'Foundation_Wood',
 'CentralAir_N',
 'Electrical_FuseF',
 'GarageType_2Types',
 'GarageType_Detchd',
 'GarageType_NA',
 'MiscFeature_Othr',
 'MiscFeature_TenC',
 'SaleType_Con',
 'SaleType_ConLD',
 'SaleType_ConLI',
 'SaleCondition_Abnorml',
 'SaleCondition_Alloca',
 'SaleCondition_Family',
 'SaleCondition_Partial']
```

In [111…

```
#Predict on test- model 3
pred_bwd=bwd_models.loc[120,'model'].predict(X_testv[bwd_models.loc[120,'model'].mo
pred_bwd=pd.Series(boxcox.inverse_transform(pred_bwd.values.reshape(-1, 1)).reshape
pred_bwd
```

Out[1119]:

```
605     195541.170154
642     380334.058708
993     173118.382864
736      73048.926007
1239    227191.471159
            ...
805     194737.694005
112     364740.035920
348     181826.401494
205     180350.407169
622     127688.549965
Length: 438, dtype: float64
```

In [112…

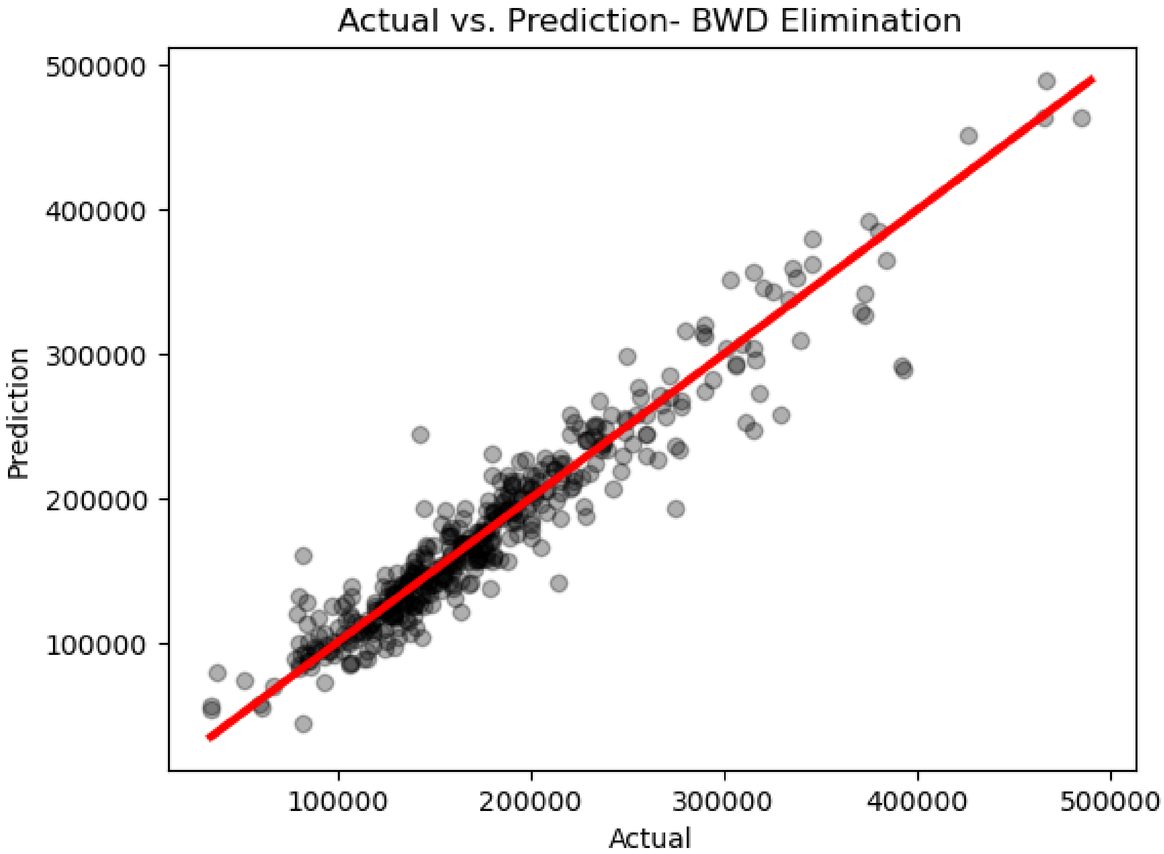
```
#Absolute error
abs_err_bwd=abs(testv['SalePrice']-pred_bwd)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_bwd=mean_absolute_error(testv['SalePrice'],pred_bwd)
mse_bwd=mean_squared_error(testv['SalePrice'],pred_bwd)
rmse_bwd=np.sqrt(mse_bwd)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                         pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_bwd.mean(
                            'MSE' :                      mse_bwd,
                            'RMSE' :                    rmse_bwd,
                            'Median of AbsErrors' : abs_err_bwd.median(),
                            'SD of AbsErrors' :     abs_err_bwd.std(),
                            'IQR of AbsErrors':     iqr(abs_err_bwd),
                            'Min of AbsErrors':     abs_err_bwd.min(),
                            'Max of AbsErrors':     abs_err_bwd.max()},
                            index = ['LM_BWD'])])
models_comp
```

Out[1120]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR of AbsErrors
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.086048
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.043120
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.349154

In [112…

```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_bwd,
            c='black', alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- BWD Elimination')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_bwd.max(), 100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



Ridge Regression

In [112…

```
#Scale data before implementing Ridge regression
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
X_trainv_scaled=scaler.fit_transform(X_trainv)
X_trainv_scaled
```

Out[1122]:

```
array([[ 0.        , -1.63557295, -0.4985911 , ..., -0.08882312,
        -0.10430876, -0.29142013],
       [ 0.        , -1.63557295, -0.06602189, ..., -0.08882312,
        -0.10430876, -0.29142013],
       [ 0.        ,  0.48852977, -0.05639083, ..., -0.08882312,
        -0.10430876, -0.29142013],
       ...,
       [ 0.        , -1.63557295,  0.43002946, ..., -0.08882312,
        -0.10430876, -0.29142013],
       [ 0.        ,  0.06370923, -0.25439713, ..., -0.08882312,
        -0.10430876, -0.29142013],
       [ 0.        , -0.27614721,  0.1562174 , ..., -0.08882312,
        -0.10430876, -0.29142013]])
```

In [112…

```
#Implement Ridge regression - model 4
from sklearn.linear_model import Ridge
ridge_reg=Ridge(alpha=0.1) #lambda = 0.1 (regularization hyperparameter)
model_4=ridge_reg.fit(X_trainv_scaled,trans_y_trainv)
```

In [112…

```
model_4.coef_
```

```
Out[1124]: array([[ 0.00000000e+00,  7.32238570e-03,  4.59805904e-02,
                   1.51595078e-01,  1.00970791e-01,  9.41006219e-02,
                   4.20683852e-02,  1.12900642e-02,  6.30239583e-02,
                   2.49101488e-02, -2.82690322e-03,  7.27390975e-02,
                   8.85069892e-02,  8.13985249e-02,  9.61708399e-03,
                   1.31344081e-01,  3.01160408e-02, -3.73133192e-03,
                   2.01734765e-02,  3.39859921e-02,  1.19729844e-02,
                  -1.95249330e-02,  5.54813564e-02,  2.31497156e-02,
                  -8.18392510e-03,  1.63402633e-02,  9.84372658e-02,
                   2.88520567e-02,  1.88935673e-02,  9.54237203e-03,
                   1.28618861e-02,  4.87257798e-02,  1.37644102e-02,
                  -1.84125411e-02,  3.06519774e-03, -9.25902930e-03,
                  -2.00906090e-03, -4.43173961e-04, -3.96198519e-03,
                  -1.49985845e-02,  1.02980334e-02,  1.69025248e-02,
                   3.51347110e-02,  2.69615589e-02, -2.16826042e-02,
                   4.08420378e-02,  3.17978503e-02,  4.65217495e-02,
                   3.60381170e-02,  1.55645586e-02,  5.58563620e-02,
                  -3.23090507e-02,  1.09909764e-02, -3.14623306e-03,
                  -6.70752301e-03, -3.01587592e-02, -7.75790947e-02,
                  -2.07018010e-02, -5.55361917e-02, -6.08047409e-02,
                   1.38931386e-02, -9.06102251e-02, -4.21092088e-02,
                  -1.08507149e-01, -3.76157532e-02, -3.19348836e-02,
                  -2.31030023e-02, -6.90604667e-03, -8.60190314e-03,
                  -9.20148294e-02,  3.22874750e-02, -3.23426532e-03,
                  -9.34019671e-03, -2.03877848e-02, -2.58142184e-02,
                   6.15463233e-03, -1.88667101e-02, -5.01399743e-03,
                  -1.32183741e-02, -2.60152236e-02,  1.07310972e-02,
                   2.62176680e-02, -9.22977750e-03, -1.39759384e-02,
                   1.24094184e-02,  7.99351115e-03, -7.28392862e-03,
                   1.53050134e-02,  4.15103643e-02,  2.37038409e-02,
                   6.49730569e-02, -1.48894316e-02,  1.74685244e-02,
                   5.61821530e-03, -4.87579907e-02, -5.11746253e-03,
                   1.11243869e-02,  1.19267608e-02,  3.19624116e-02,
                   7.27371360e-02, -9.70791594e-03,  2.07994919e-02,
                   7.82635865e-03,  2.66069132e-02,  2.12847867e-02,
                   5.46540303e-02,  1.94744374e-02,  2.59936838e-02,
                  -3.01782650e-02, -3.16580814e-02,  7.54972854e-03,
                   5.78971486e-04, -2.59932092e-02, -4.63787749e-04,
                   1.24683126e-03,  2.59547052e-03, -2.28408609e-02,
                   7.46015595e-03,  1.07032480e-02, -8.54372107e-02,
                  -9.53250769e-03,  2.01269850e-04,  2.61062770e-02,
                  -8.60190314e-03, -2.51084712e-02, -1.22749987e-02,
                   5.00812388e-02,  8.79591880e-02,  1.43089029e-02,
                   3.60085682e-02,  9.35265233e-02,  1.96079856e-03,
                   2.35426512e-02, -1.39872955e-03, -7.89201142e-03,
                   3.78712905e-03,  1.29537223e-02, -9.53250769e-03,
                  -8.76028570e-03,  1.47222516e-02, -6.36289493e-03,
                   7.56558416e-04,  3.00078830e-02, -1.76314689e-01,
                   1.52459734e-02,  1.74595352e-03, -3.59579610e-03,
                  -1.76609834e-02,  1.55549584e-02, -2.55929172e-03,
                  -4.65499055e-02,  3.94265414e-02, -3.80960375e-03,
                  -8.01502850e-02, -2.23603435e-02, -1.18968275e-03,
                   3.51950634e-02, -1.17882756e-02,  7.07426891e-03,
                  -2.54733777e-02, -4.70785281e-02, -1.56733605e-03,
                  -9.60845548e-03,  2.55298297e-03,  2.00126767e-03,
                  -2.80013192e-02, -3.80960375e-03,  9.11999503e-02,
                  -6.62568958e-03,  5.98779492e-03, -2.70870805e-02,
                  -1.02782287e-02, -4.37061679e-03, -2.11322717e-03,
                   2.68676010e-02,  3.98606724e-02, -1.40230803e-02,
                  -1.09132847e-02,  3.69749516e-03,  1.05358389e-02,
                  -4.10952088e-02, -3.40425009e-02, -1.99301652e-02,
                   6.97272924e-03, -1.76049260e-02, -4.72214634e-02,
                   5.55280692e-03, -1.47945472e-02,  6.81039073e-03,
                  -1.24790843e-02,  9.06036409e-03, -1.45521928e-03,
```

```
         8.41965528e-03, -2.13134377e-02,  2.89277221e-02,
         2.63388364e-02, -1.91389555e-02,  1.87181459e-03,
        -1.97557698e-02,  4.96633330e-03,  9.44370992e-03,
         1.01491636e-02,  3.77045795e-02, -1.18223182e-02,
        -1.44884678e-04, -6.50970864e-02,  1.66338037e-03,
        -2.19530918e-02, -1.79471892e-03, -1.20151337e-02,
        -1.15871528e-02,  8.64411525e-02]])
```

In [112…

```python
# Grid
lambda_grid=10**np.linspace(3,-3,100)
lambda_grid
```

Out[1125]:

```
array([1.00000000e+03, 8.69749003e+02, 7.56463328e+02, 6.57933225e+02,
       5.72236766e+02, 4.97702356e+02, 4.32876128e+02, 3.76493581e+02,
       3.27454916e+02, 2.84803587e+02, 2.47707636e+02, 2.15443469e+02,
       1.87381742e+02, 1.62975083e+02, 1.41747416e+02, 1.23284674e+02,
       1.07226722e+02, 9.32603347e+01, 8.11130831e+01, 7.05480231e+01,
       6.13590727e+01, 5.33669923e+01, 4.64158883e+01, 4.03701726e+01,
       3.51119173e+01, 3.05385551e+01, 2.65608778e+01, 2.31012970e+01,
       2.00923300e+01, 1.74752840e+01, 1.51991108e+01, 1.32194115e+01,
       1.14975700e+01, 1.00000000e+01, 8.69749003e+00, 7.56463328e+00,
       6.57933225e+00, 5.72236766e+00, 4.97702356e+00, 4.32876128e+00,
       3.76493581e+00, 3.27454916e+00, 2.84803587e+00, 2.47707636e+00,
       2.15443469e+00, 1.87381742e+00, 1.62975083e+00, 1.41747416e+00,
       1.23284674e+00, 1.07226722e+00, 9.32603347e-01, 8.11130831e-01,
       7.05480231e-01, 6.13590727e-01, 5.33669923e-01, 4.64158883e-01,
       4.03701726e-01, 3.51119173e-01, 3.05385551e-01, 2.65608778e-01,
       2.31012970e-01, 2.00923300e-01, 1.74752840e-01, 1.51991108e-01,
       1.32194115e-01, 1.14975700e-01, 1.00000000e-01, 8.69749003e-02,
       7.56463328e-02, 6.57933225e-02, 5.72236766e-02, 4.97702356e-02,
       4.32876128e-02, 3.76493581e-02, 3.27454916e-02, 2.84803587e-02,
       2.47707636e-02, 2.15443469e-02, 1.87381742e-02, 1.62975083e-02,
       1.41747416e-02, 1.23284674e-02, 1.07226722e-02, 9.32603347e-03,
       8.11130831e-03, 7.05480231e-03, 6.13590727e-03, 5.33669923e-03,
       4.64158883e-03, 4.03701726e-03, 3.51119173e-03, 3.05385551e-03,
       2.65608778e-03, 2.31012970e-03, 2.00923300e-03, 1.74752840e-03,
       1.51991108e-03, 1.32194115e-03, 1.14975700e-03, 1.00000000e-03])
```

In [112…

```python
# K-fold cross validation to choose the best model
from sklearn.model_selection import cross_val_score

cv_errors=np.zeros(shape=len(lambda_grid)) #to save cv results

for i in range(len(lambda_grid)):
    ridge_reg = Ridge(alpha = lambda_grid[i])
    scores=cross_val_score(estimator = ridge_reg,
                            X=X_trainv_scaled,
                            y=trans_y_trainv,
                            scoring='neg_root_mean_squared_error',
                            cv=5, n_jobs = -1)
    cv_errors[i]=scores.mean()


cv_errors
```

Out[1126]:
```
array([-0.3667995 , -0.36287383, -0.35966796, -0.3571202 , -0.35517322,
       -0.35377342, -0.35287034, -0.35241614, -0.3523652 , -0.35267382,
       -0.35330005, -0.35420366, -0.35534616, -0.35669098, -0.35820364,
       -0.35985193, -0.36160613, -0.36343917, -0.36532667, -0.367247  ,
       -0.36918116, -0.37111269, -0.37302745, -0.37491341, -0.37676038,
       -0.37855986, -0.38030476, -0.38198928, -0.38360879, -0.38515968,
       -0.38663934, -0.38804604, -0.38937888, -0.39063774, -0.39182318,
       -0.39293635, -0.39397895, -0.39495312, -0.39586141, -0.39670669,
       -0.39749214, -0.39822121, -0.39889761, -0.39952528, -0.4001084 ,
       -0.40065137, -0.4011588 , -0.40163547, -0.40208633, -0.40251647,
       -0.40293103, -0.40333519, -0.40373406, -0.40413264, -0.40453561,
       -0.4049473 , -0.40537147, -0.40581116, -0.40626855, -0.40674481,
       -0.40724001, -0.40775305, -0.4082817 , -0.4088227 , -0.40937189,
       -0.40992448, -0.4104753 , -0.41101907, -0.41155066, -0.41206539,
       -0.41255913, -0.41302853, -0.41347098, -0.41388471, -0.41426871,
       -0.41462268, -0.41494691, -0.41524219, -0.41550971, -0.41575093,
       -0.41596753, -0.41616129, -0.41633402, -0.41648756, -0.41662366,
       -0.41674403, -0.41685027, -0.41694385, -0.41702617, -0.41709846,
       -0.41716188, -0.41721745, -0.4172661 , -0.41730865, -0.41734584,
       -0.41737833, -0.41740669, -0.41743144, -0.41745303, -0.41747185])
```

In [112…
```
# Best lambda
best_lamda=lambda_grid[np.argmax(cv_errors)]
best_lamda
```

Out[1127]:
```
327.45491628777285
```

In [112…
```
# Best model coefs:
ridge_reg=Ridge(alpha = best_lamda)
model_4=ridge_reg.fit(X_trainv_scaled, trans_y_trainv)
model_4.coef_
```

Out[1128]:
```
array([[ 0.        ,  0.00174329,  0.02568706,  0.10723167,  0.06223939,
         0.03592919,  0.0495579 ,  0.02023223,  0.038852  ,  0.01175145,
         0.00484678,  0.04993221,  0.06833731,  0.03998357,  0.00954872,
         0.0831437 ,  0.03061545, -0.00345734,  0.04561003,  0.03384147,
         0.0229963 , -0.0132366 ,  0.06133767,  0.03756153, -0.00105271,
         0.04989307,  0.05565956,  0.02588079,  0.01931915,  0.00189787,
         0.01014623,  0.035487  , -0.01376503,  0.00094134,  0.00343951,
        -0.00562802, -0.01015032, -0.00245177,  0.03103153, -0.00080777,
         0.04147529,  0.01530245,  0.03642117,  0.03926574, -0.00493459,
         0.03775288,  0.04794026,  0.0346697 ,  0.04527004,  0.02233548,
         0.01348109,  0.00288914,  0.01827462,  0.01904978, -0.0042031 ,
        -0.01298441, -0.02357889, -0.01107988, -0.00149405, -0.05121887,
         0.00751329, -0.01358006,  0.0079985 ,  0.00330132,  0.01579974,
         0.00288616, -0.00126987, -0.00154423, -0.00384491, -0.06179222,
         0.01675725,  0.00457085, -0.02218707, -0.01014279, -0.01945708,
         0.00722369, -0.01953328,  0.00284694,  0.00111233, -0.01463245,
         0.00629027,  0.02273049, -0.00793482, -0.00694703, -0.00486765,
         0.00286818, -0.01036363,  0.00259329,  0.02667536,  0.00429986,
         0.03990482, -0.02976773, -0.00881072, -0.0157345 , -0.0430167 ,
        -0.00936352,  0.00145498,  0.00485088,  0.02974919,  0.04559798,
        -0.02062491,  0.00284406, -0.00947884,  0.00527816,  0.01548794,
         0.03462395,  0.00926316,  0.01756404, -0.02438442, -0.02195453,
         0.006525  , -0.00644296, -0.0159815 , -0.00076507,  0.00037862,
         0.00055533, -0.01049658,  0.00535539,  0.01223517, -0.04933065,
        -0.00230133, -0.00649203,  0.00156167, -0.00384491, -0.02392093,
        -0.01808973,  0.01062339,  0.00613071,  0.00525385,  0.00747338,
         0.00690282, -0.01084188,  0.00022955,  0.00515813, -0.00815459,
         0.01371513,  0.01029907, -0.00230133, -0.00811724,  0.01135996,
        -0.00045691, -0.00079532,  0.01115012, -0.09749519,  0.00976891,
        -0.00075977, -0.00482177, -0.00586782,  0.02278748, -0.0040911 ,
        -0.03224808,  0.03001909, -0.00359158, -0.00098945, -0.01743609,
         0.00141509,  0.00316357, -0.00668603,  0.00834855, -0.00868175,
        -0.01658095, -0.00559715, -0.004637  ,  0.00177511, -0.00587819,
        -0.01853512, -0.00359158,  0.00459649, -0.01243646,  0.00721608,
        -0.004595  , -0.00236857, -0.00577988, -0.00063764,  0.00015269,
         0.00197245, -0.01548349, -0.01099366,  0.00509601,  0.01083467,
        -0.0259395 , -0.02578684, -0.00766186,  0.00366072, -0.00537785,
        -0.0395555 ,  0.00251775, -0.01110515,  0.00386669, -0.01406353,
         0.00232808,  0.00384023, -0.00767327, -0.02749945,  0.00273054,
         0.00176743, -0.01411351, -0.00619432, -0.00733922, -0.0024027 ,
         0.00732542,  0.01039495,  0.01548926, -0.00838621, -0.00064387,
         0.01085702,  0.00154296, -0.02117047,  0.00272589, -0.00564405,
        -0.01127287,  0.01266997]])
```

Evaluation on the Validation Set (Ridge Regression)

In [112…
```
# Predict on test- model 4
# Sacle test data set
X_testv_scaled=scaler.transform(X_testv)
pred_ridge=model_4.predict(X_testv_scaled)
pred_ridge=pd.Series(boxcox.inverse_transform(pred_ridge).reshape(-1),index = testv
pred_ridge
```

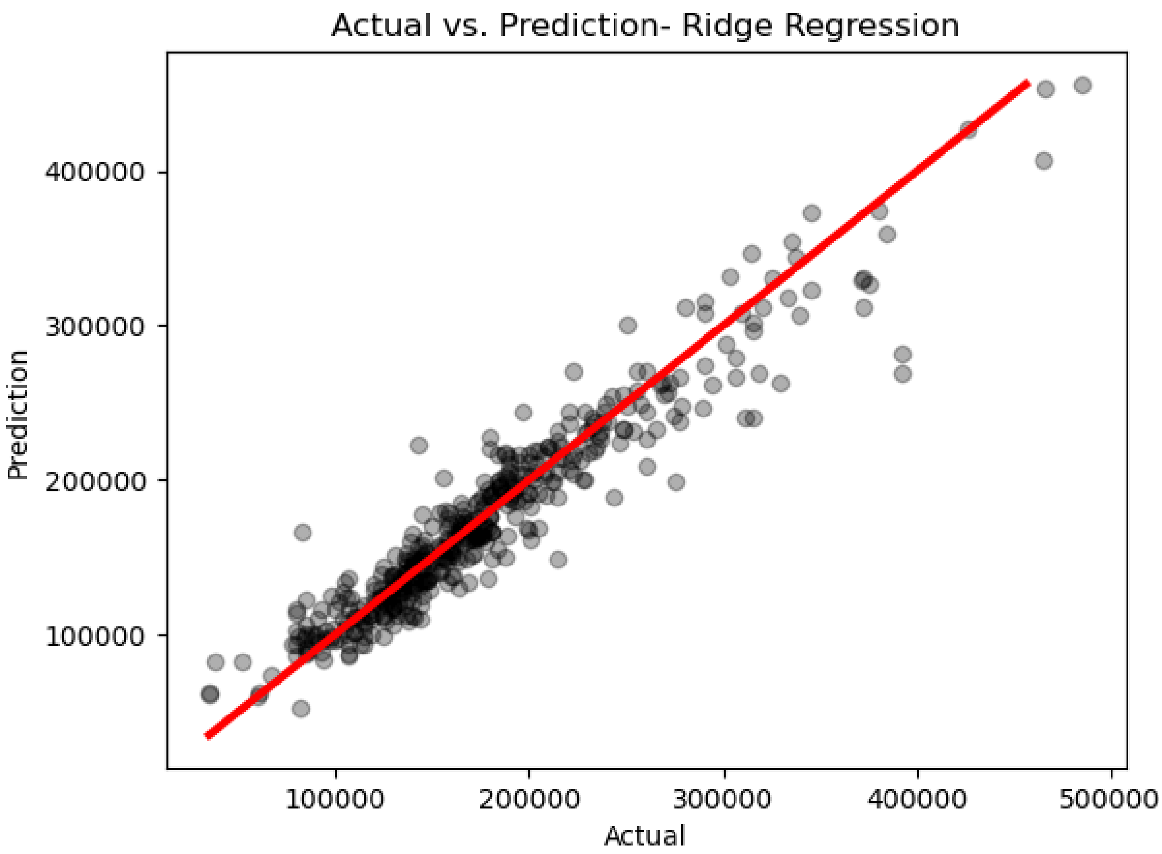
Out[1129]:
```
605     190934.096532
642     323658.838084
993     174622.099050
736      83348.271981
1239    232559.760090
            ...
805     200111.622486
112     359067.620016
348     179258.119853
205     166495.189843
622     127874.772532
Length: 438, dtype: float64
```

In [113…
```
# Absolute error
abs_err_ridge=abs(testv['SalePrice']-pred_ridge)


# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_ridge =mean_absolute_error(testv['SalePrice'], pred_ridge)
mse_ridge=mean_squared_error(testv['SalePrice'], pred_ridge)
rmse_ridge=np.sqrt(mse_bwd)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                         pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_ridge.mear
                            'MSE' :                         mse_ridge,
                            'RMSE' :                         rmse_ridge,
                            'Median of AbsErrors' :   abs_err_ridge.median(),
                            'SD of AbsErrors' :       abs_err_ridge.std(),
                            'IQR of AbsErrors':       iqr(abs_err_ridge),
                            'Min of AbsErrors':       abs_err_ridge.min(),
                            'Max of AbsErrors':       abs_err_ridge.max()},
                            index = ['LM_Ridge'])])
models_comp
```

Out[1130]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR o AbsError
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.08604
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.04312
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.34915
LM_Ridge	14919.438525	4.566752e+08	20794.637168	10644.006449	15317.349954	13898.22440

In [113…
```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_ridge,
            c='black',alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- Ridge Regression')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_ridge.max(),100)
plt.plot(xp,xp,c='red', linewidth=3)
plt.show()
```



LASSO Regression

In [113…

```
#Implement LASSO regression- model 5
from sklearn.linear_model import Lasso
lasso_reg=Lasso(alpha = 0.1) #lambda = 0.1 (regularization hyperparameter)
model_5=lasso_reg.fit(X_trainv_scaled, trans_y_trainv)
```

In [113…

```
model_5.coef_
```

Out[1133]:
```
array([ 0.        ,  0.        ,  0.        ,  0.30574096,  0.        ,
        0.01799305,  0.05325   ,  0.        ,  0.        ,  0.        ,
       -0.        ,  0.02711858,  0.02397547,  0.        , -0.        ,
        0.21689996,  0.        ,  0.        ,  0.        ,  0.        ,
        0.        , -0.        ,  0.        ,  0.01545009,  0.        ,
        0.09687159,  0.01267609,  0.        ,  0.        , -0.        ,
        0.        ,  0.        , -0.        ,  0.        ,  0.        ,
       -0.        , -0.        , -0.        ,  0.        ,  0.        ,
        0.03956782,  0.        ,  0.00184169,  0.04400497,  0.        ,
        0.00092407,  0.06388295,  0.        ,  0.04340723,  0.03603645,
        0.        ,  0.        ,  0.        ,  0.        , -0.        ,
       -0.        , -0.        , -0.        , -0.        , -0.00744165,
        0.        , -0.        , -0.        ,  0.        , -0.        ,
        0.        ,  0.        ,  0.        , -0.        , -0.        ,
        0.        , -0.        , -0.04601103, -0.        , -0.        ,
       -0.        , -0.        ,  0.        ,  0.        , -0.        ,
        0.        ,  0.        , -0.        , -0.        , -0.        ,
       -0.        , -0.        , -0.        ,  0.        ,  0.        ,
        0.        , -0.        ,  0.        , -0.        , -0.        ,
       -0.        , -0.        ,  0.        ,  0.        ,  0.        ,
       -0.        , -0.        , -0.        , -0.        ,  0.        ,
        0.        ,  0.        ,  0.        , -0.        , -0.        ,
        0.        , -0.        , -0.        ,  0.        ,  0.        ,
        0.        , -0.        , -0.        ,  0.        , -0.        ,
       -0.        , -0.        , -0.        , -0.        , -0.        ,
       -0.        , -0.        , -0.        ,  0.        , -0.        ,
        0.        , -0.        ,  0.        ,  0.        , -0.        ,
        0.        ,  0.        , -0.        , -0.        ,  0.        ,
       -0.        , -0.        , -0.        , -0.01625301,  0.        ,
        0.        ,  0.        , -0.        ,  0.        , -0.        ,
       -0.        ,  0.        , -0.        , -0.        , -0.        ,
        0.        , -0.        ,  0.        ,  0.        , -0.        ,
       -0.        , -0.        , -0.        ,  0.        , -0.        ,
        0.        , -0.        , -0.        , -0.        ,  0.        ,
       -0.        ,  0.        ,  0.        ,  0.        , -0.        ,
       -0.        , -0.        , -0.        ,  0.        ,  0.        ,
       -0.        , -0.        , -0.        , -0.        , -0.        ,
       -0.03309315, -0.        , -0.        , -0.        , -0.        ,
       -0.        ,  0.        , -0.        , -0.        , -0.        ,
        0.        , -0.        , -0.        ,  0.        , -0.        ,
        0.        ,  0.        , -0.        , -0.        ,  0.        ,
        0.        ,  0.        , -0.        , -0.        , -0.        ,
       -0.        ,  0.        ])
```

In [113…
```
# K-fold cross validation to choose the best model
cv_errors=np.zeros(shape=len(lambda_grid)) #to save cv results

for i in range(len(lambda_grid)):
    lasso_reg=Lasso(alpha = lambda_grid[i])
    scores= cross_val_score(estimator=lasso_reg,
                            X=X_trainv_scaled,
                            y=trans_y_trainv,
                            scoring='neg_root_mean_squared_error',
                            cv=5, n_jobs = -1)
    cv_errors[i]=scores.mean()

cv_errors
```

Out[1134]:
```
array([-0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99894425, -0.99894425, -0.99894425, -0.99894425,
       -0.99894425, -0.99126611, -0.91090268, -0.84186938, -0.78567644,
       -0.73453045, -0.68587402, -0.64329573, -0.60702493, -0.5761372 ,
       -0.54977637, -0.5271712 , -0.50634694, -0.48687507, -0.46964743,
       -0.45458504, -0.44151655, -0.4306598 , -0.42150252, -0.41289122,
       -0.4058153 , -0.39948476, -0.39374067, -0.3888594 , -0.3848302 ,
       -0.3818562 , -0.37966781, -0.37785305, -0.37623067, -0.37507591,
       -0.37446216, -0.37381802, -0.37339266, -0.37320096, -0.37301492,
       -0.37317602, -0.37354278, -0.37421027, -0.37498107, -0.37577563,
       -0.37665012, -0.37755507, -0.37859787, -0.3796678 , -0.38088082,
       -0.38208587, -0.3831096 , -0.38418516, -0.38525676, -0.38624816])
```

In [113…
```
# Best lambda
best_lamda=lambda_grid[np.argmax(cv_errors)]
best_lamda
```

Out[1135]:
```
0.008111308307896872
```

In [113…
```
# Best model coefs:
lasso_reg=Lasso(alpha=best_lamda)
model_5=lasso_reg.fit(X_trainv_scaled,trans_y_trainv)
model_5.coef_
```

```
Out[1136]: array([ 0.00000000e+00,  0.00000000e+00,  3.03229169e-02,  1.94443380e-01,
         8.11902056e-02,  6.86637144e-02,  5.28651507e-02,  1.26388962e-03,
         5.33789655e-02,  0.00000000e+00, -0.00000000e+00,  8.76771877e-02,
         6.37366791e-03,  0.00000000e+00,  0.00000000e+00,  2.65591604e-01,
         2.45746518e-02, -0.00000000e+00,  1.36528489e-02,  1.65796591e-02,
         0.00000000e+00, -1.98845212e-02,  3.71679339e-02,  2.63136661e-02,
         0.00000000e+00,  1.89673360e-02,  8.92502037e-02,  2.22810882e-02,
         1.16168627e-02, -0.00000000e+00,  1.89800710e-04,  3.26953053e-02,
        -0.00000000e+00, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        -1.18829845e-03, -0.00000000e+00,  1.04171260e-02, -0.00000000e+00,
         3.21210328e-02,  2.12839413e-03,  3.26214281e-02,  2.75428489e-02,
         0.00000000e+00,  3.79510472e-02,  3.54352260e-02,  4.48557798e-02,
         3.67247656e-02,  1.22319325e-02,  0.00000000e+00,  0.00000000e+00,
         9.36171749e-03, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        -1.08224861e-02, -0.00000000e+00, -0.00000000e+00, -5.57043185e-02,
         0.00000000e+00, -3.70291471e-03,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        -2.42241929e-03, -7.64651156e-02,  1.11025864e-02,  0.00000000e+00,
        -3.22457111e-02, -5.98963313e-03, -1.74113995e-02,  2.79800641e-04,
        -1.13703477e-02,  0.00000000e+00,  0.00000000e+00, -5.36237401e-03,
         0.00000000e+00,  1.52630398e-02, -0.00000000e+00, -0.00000000e+00,
        -1.35966926e-03,  0.00000000e+00, -5.32883415e-03,  0.00000000e+00,
         2.40182711e-02,  0.00000000e+00,  4.22249980e-02, -1.72027477e-02,
        -0.00000000e+00, -0.00000000e+00, -4.05315075e-02, -7.16235413e-03,
        -0.00000000e+00, -0.00000000e+00,  7.12440871e-03,  3.67871528e-02,
        -9.44757875e-03,  0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
         8.87676322e-03,  2.13624709e-02,  0.00000000e+00,  7.38953085e-03,
        -2.64156094e-02, -1.47874288e-02,  0.00000000e+00, -0.00000000e+00,
        -1.54052459e-02,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        -0.00000000e+00,  0.00000000e+00,  0.00000000e+00, -7.89089765e-02,
        -0.00000000e+00, -0.00000000e+00, -0.00000000e+00, -1.94855097e-04,
        -3.14107304e-02, -2.17081138e-02,  0.00000000e+00, -0.00000000e+00,
        -0.00000000e+00,  0.00000000e+00, -0.00000000e+00, -6.50722362e-03,
         0.00000000e+00,  0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
         4.27463050e-03, -0.00000000e+00, -0.00000000e+00,  5.92441366e-03,
        -0.00000000e+00, -0.00000000e+00,  1.11059018e-02, -1.69283271e-01,
         5.92060074e-03,  0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
         3.19592232e-03, -0.00000000e+00, -3.90037093e-02,  3.28693119e-02,
        -0.00000000e+00,  0.00000000e+00, -1.09121096e-02,  0.00000000e+00,
         2.57109666e-03, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        -4.37452752e-03, -0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        -0.00000000e+00, -1.19370363e-02, -0.00000000e+00,  0.00000000e+00,
        -4.17456911e-03,  0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        -1.04650003e-02, -3.10708512e-03,  0.00000000e+00,  0.00000000e+00,
        -2.31184240e-02, -1.79763530e-02, -0.00000000e+00,  5.31862760e-04,
        -3.97834446e-03, -4.24002444e-02,  0.00000000e+00, -2.03048689e-03,
         0.00000000e+00, -5.46620225e-03, -0.00000000e+00,  0.00000000e+00,
        -5.25464233e-03, -1.31965753e-02, -0.00000000e+00, -0.00000000e+00,
        -6.51398719e-03, -0.00000000e+00, -5.70078777e-03, -0.00000000e+00,
         4.92876159e-04,  2.56433412e-03,  1.78081628e-02, -5.85971901e-03,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00, -1.81954525e-02,
         0.00000000e+00, -5.50217053e-03, -6.00812077e-03,  2.29324063e-02])
```

Evaluation on the Validation Set (LASSO Regression)

```
In [113…
#Predict on test- model 5
pred_lasso=model_5.predict(X_testv_scaled)
pred_lasso=pd.Series(boxcox.inverse_transform(pred_lasso.reshape(-1, 1)).reshape(-1
pred_lasso
```

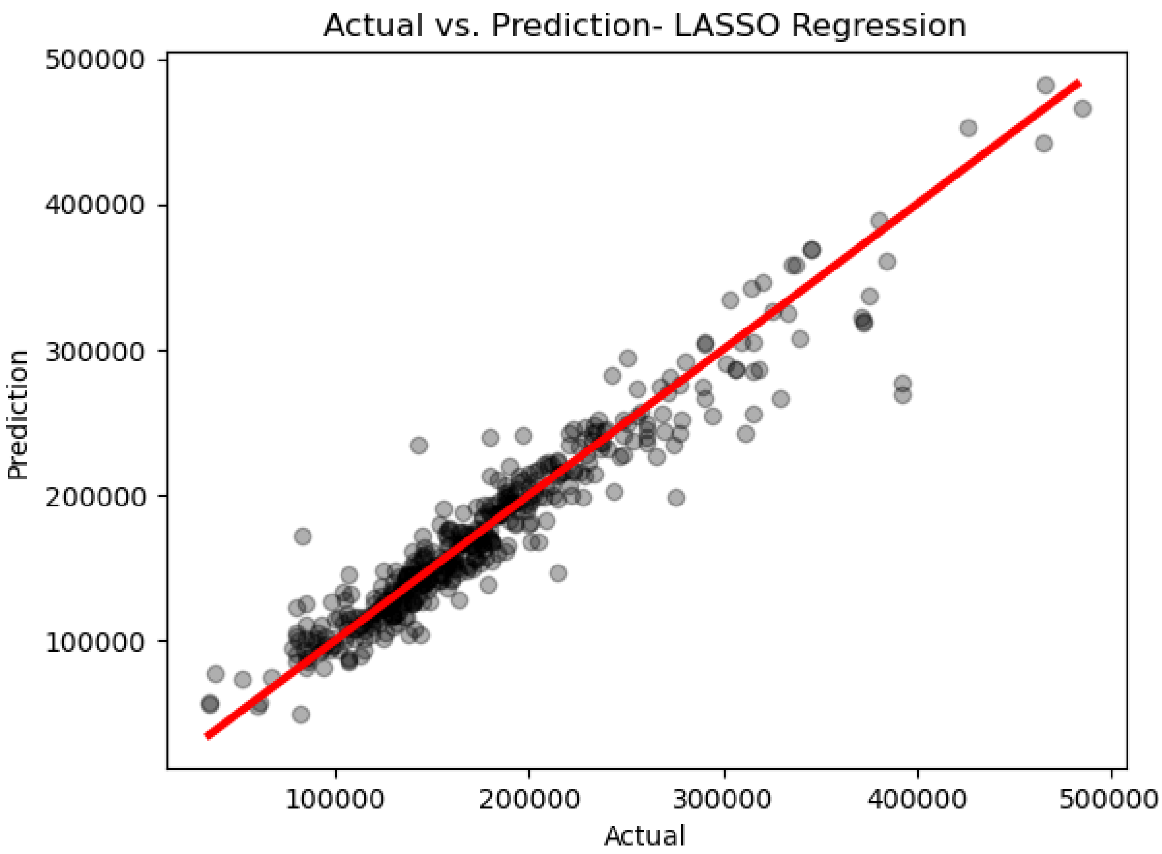
Out[1137]:
```
605     198809.663802
642     368352.393210
993     172618.090088
736      81324.737208
1239    226543.458543
             ...
805     198783.264307
112     361457.161753
348     179354.999264
205     167631.168521
622     130472.221470
Length: 438, dtype: float64
```

In [113…
```python
#Absolute error
abs_err_lasso=abs(testv['SalePrice']-pred_lasso)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_lasso=mean_absolute_error(testv['SalePrice'],pred_lasso)
mse_lasso=mean_squared_error(testv['SalePrice'],pred_lasso)
rmse_lasso=np.sqrt(mse_lasso)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                        pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_lasso.mear
                           'MSE' :                     mse_lasso,
                           'RMSE' :                     rmse_lasso,
                           'Median of AbsErrors' :     abs_err_lasso.median(),
                           'SD of AbsErrors' :         abs_err_lasso.std(),
                           'IQR of AbsErrors':         iqr(abs_err_lasso),
                           'Min of AbsErrors':         abs_err_lasso.min(),
                           'Max of AbsErrors':         abs_err_lasso.max()},
                           index = ['LM_LASSO'])])
models_comp
```

Out[1138]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR c AbsErro
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.08604
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.04312
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.34915
LM_Ridge	14919.438525	4.566752e+08	20794.637168	10644.006449	15317.349954	13898.22440
LM_LASSO	13721.483435	4.111831e+08	20277.650641	10065.606325	14947.042706	14111.47899

In [113…
```python
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_lasso,
            c='black', alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- LASSO Regression')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_lasso.max(),100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



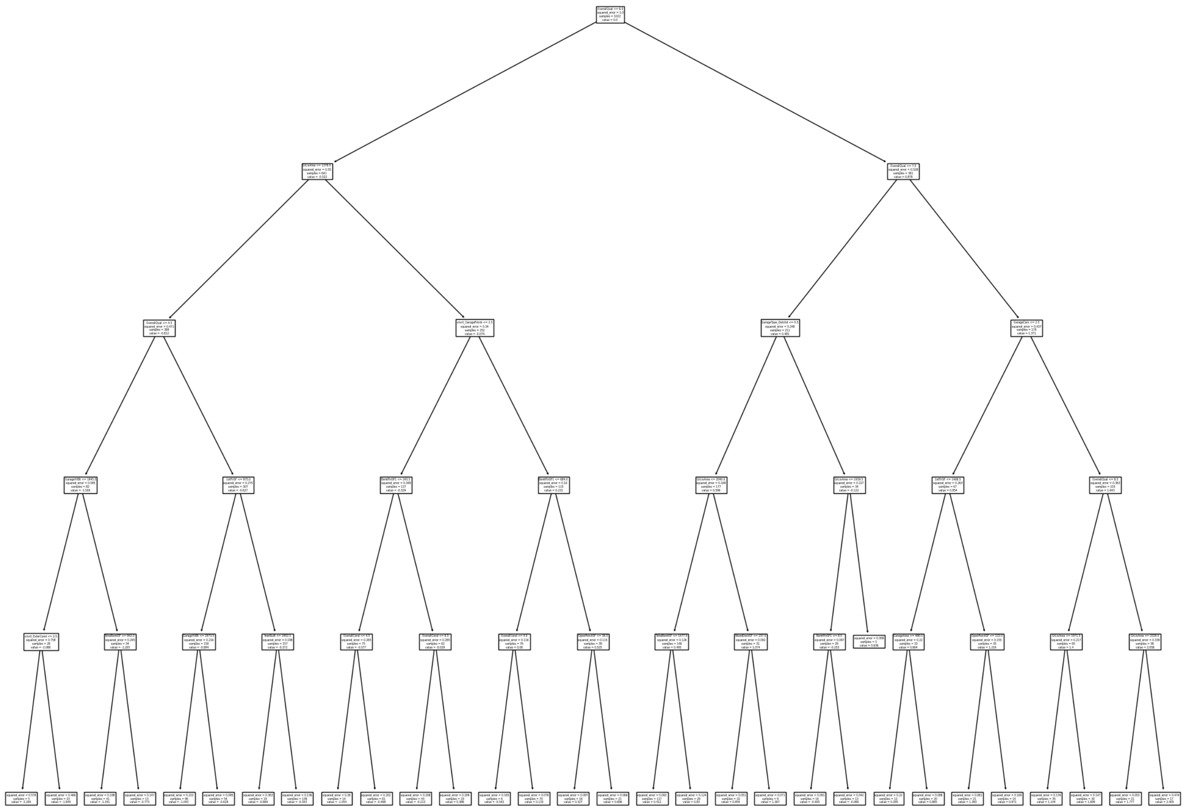
Decision Tree

In [114…

```
# Decision Tree- model 6
from sklearn.tree import DecisionTreeRegressor
#Post prunning approach
tree_reg=DecisionTreeRegressor(criterion='squared_error',
                                 max_depth=5,
                                 min_samples_leaf=5,
                                 ccp_alpha=0)
model_6=tree_reg.fit(X_trainv,trans_y_trainv)
```

In [114…

```
# Plot the Tree
from sklearn.tree import plot_tree

# Convert the feature names from Index to list
feature_names_list=X_trainv.columns.tolist()

# Plot the Tree
plt.figure(figsize=(25,20))
plot_tree(model_6,feature_names=feature_names_list)
plt.show()
```



In [114…

```python
# Post prunning
prunning_path=model_6.cost_complexity_pruning_path(X_trainv,trans_y_trainv)
prunning_path
```

Out[1142]:

```
{'ccp_alphas': array([0.        , 0.00080666, 0.00106658, 0.00129088, 0.00139674,
        0.00246204, 0.00256018, 0.00311564, 0.00329696, 0.00362981,
        0.00382836, 0.0039706 , 0.00464797, 0.00487952, 0.00500785,
        0.00503006, 0.00523436, 0.00581419, 0.00744315, 0.00839654,
        0.00999746, 0.01013965, 0.01407964, 0.01440621, 0.01886543,
        0.01917789, 0.02046571, 0.04894516, 0.07306761, 0.08167532,
        0.45820757]),
 'impurities': array([0.15709427, 0.15790093, 0.15896751, 0.16025839, 0.16165514,
        0.16411718, 0.16667736, 0.169793  , 0.17308996, 0.17671977,
        0.18054814, 0.18451874, 0.18916671, 0.19404623, 0.19905408,
        0.20408413, 0.20931849, 0.21513268, 0.22257582, 0.23097236,
        0.24096982, 0.25110946, 0.2651891 , 0.27959531, 0.29846074,
        0.31763863, 0.33810433, 0.38704949, 0.46011711, 0.54179243,
        1.        ])}
```

In [114…

```python
# Grid
alpha_grid=prunning_path['ccp_alphas']
alpha_grid
```

Out[1143]:

```
array([0.        , 0.00080666, 0.00106658, 0.00129088, 0.00139674,
       0.00246204, 0.00256018, 0.00311564, 0.00329696, 0.00362981,
       0.00382836, 0.0039706 , 0.00464797, 0.00487952, 0.00500785,
       0.00503006, 0.00523436, 0.00581419, 0.00744315, 0.00839654,
       0.00999746, 0.01013965, 0.01407964, 0.01440621, 0.01886543,
       0.01917789, 0.02046571, 0.04894516, 0.07306761, 0.08167532,
       0.45820757])
```

In [114…

```python
# K-fold cross validation to choose the best model
from sklearn.model_selection import cross_val_score
```

```python
cv_errors=np.zeros(shape =len(alpha_grid)) #to save cv results

import time #to measure the processing time
start_time=time.time()
for i in range(len(alpha_grid)):
    tree_reg=DecisionTreeRegressor(criterion='squared_error',
                                   max_depth=5,
                                   min_samples_leaf=5,
                                   ccp_alpha=alpha_grid[i])
    scores = cross_val_score(estimator=tree_reg,
                             X=X_trainv,
                             y=trans_y_trainv,
                             scoring='neg_root_mean_squared_error',
                             cv=10,n_jobs=-1)
    cv_errors[i]=scores.mean()

end_time=time.time()
print('The Processing time is: ',end_time-start_time, 'seconds')
cv_errors
```

```
The Processing time is:  10.102499008178711 seconds
```

Out[1144]:

```
array([-0.52047494, -0.51929931, -0.5192994 , -0.51761649, -0.51810206,
       -0.51593884, -0.51605044, -0.5186106 , -0.52056274, -0.52096726,
       -0.52410198, -0.52569109, -0.53813497, -0.54044781, -0.54314933,
       -0.54368472, -0.54276806, -0.54206891, -0.54372802, -0.54999315,
       -0.55461642, -0.55461642, -0.57442141, -0.5761312 , -0.59682646,
       -0.59586132, -0.60174861, -0.65098791, -0.6867531 , -0.70940313,
       -0.90257066])
```

In [114…

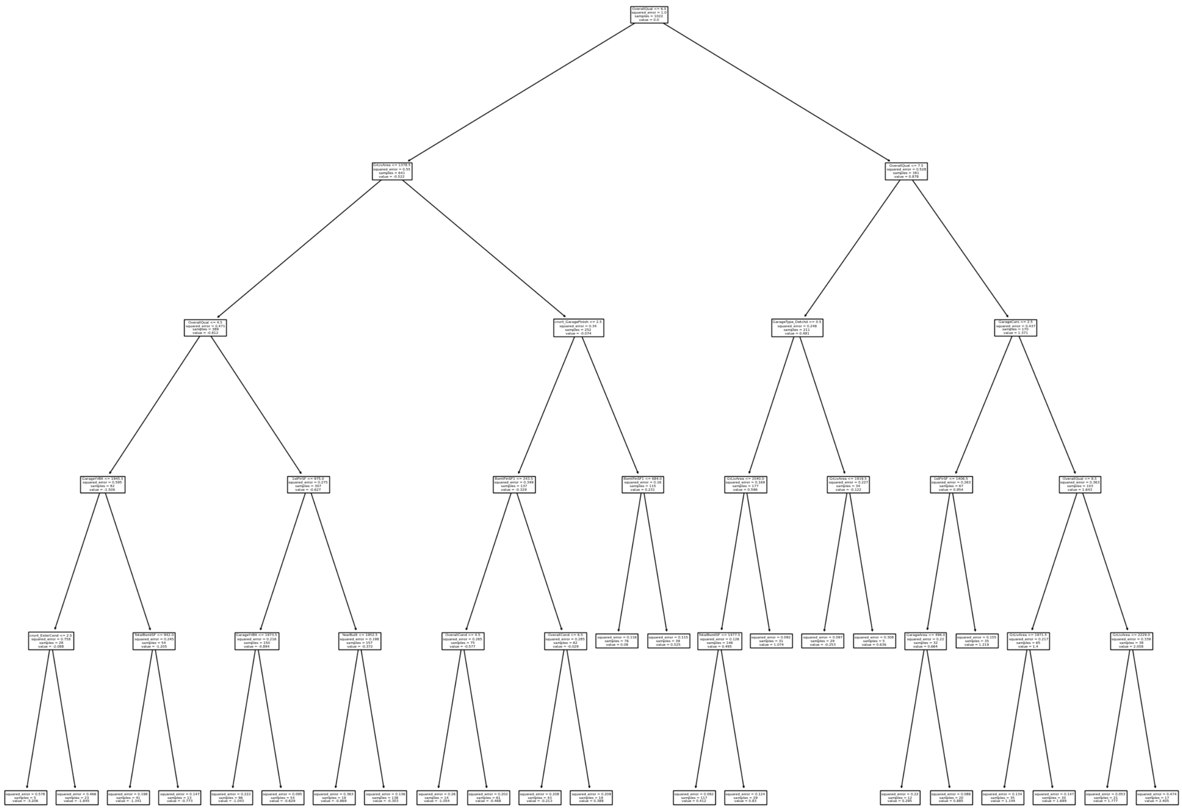
```python
# Best alpha
best_alpha=alpha_grid[np.argmax(cv_errors)]
best_alpha
```

Out[1145]:

```
0.0024620422784568117
```

In [114…

```python
#Best model
tree_reg=DecisionTreeRegressor(criterion='squared_error',
                               max_depth=5,
                               min_samples_leaf=5,
                               ccp_alpha=best_alpha)


model_6=tree_reg.fit(X_trainv,trans_y_trainv)

# Convert the feature names from Index to list
feature_names_list=X_trainv.columns.tolist()


#Plot the Tree
from sklearn.tree import plot_tree
plt.figure(figsize=(25, 20))
plot_tree(model_6, feature_names=feature_names_list)
plt.show()
```



In [114…

```
X_testv.shape
```

Out[1147]:

```
(438, 212)
```

In [114…

```
print(dummy_vars_tev.shape)
print(testv.shape)
```

```
(438, 157)
(438, 101)
```

Evaluation on the Validation Set (Decision Tree)

In [114…

```
 #Define feature matrix
# Train All columns except 'SalePrice'
X_X=testv.iloc[:,list(range(0,80))+list(range(81,100))]
# print(X_X.columns)
X_testv=pd.concat([X_X,dummy_vars_tev],axis = 1)
#add constant
# X_testv=sm.add_constant(X_testv)
X_testv = sm.add_constant(X_testv,has_constant='add')
X_testv.shape
print(X_testv.columns)
```

```
Index(['const', 'Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea',
       'Street', 'Alley', 'LotShape', 'LandContour',
       ...
       'SaleType_ConLD', 'SaleType_ConLI', 'SaleType_ConLw', 'SaleType_New',
       'SaleType_Oth', 'SaleCondition_Abnorml', 'SaleCondition_AdjLand',
       'SaleCondition_Alloca', 'SaleCondition_Family',
       'SaleCondition_Partial'],
      dtype='object', length=257)
```

In [115…

```
# Remove coloumn'Id'
# X_testv.drop(columns=['Id'],inplace=True)
X_testv.shape
```

Out[1150]:

```
(438, 257)
```

In [115…

```
X_testv=X_testv.drop(columns=X_testv.select_dtypes(include=['object']).columns)
# Check the data types of the cleaned DataFrame
print(X_testv.dtypes)
print(X_testv.shape)
```

```
const                      float64
Id                           int64
LotFrontage                  int64
LotArea                      int64
OverallQual                  int64
                            ...
SaleCondition_Abnorml        int32
SaleCondition_AdjLand        int32
SaleCondition_Alloca         int32
SaleCondition_Family         int32
SaleCondition_Partial        int32
Length: 213, dtype: object
(438, 213)
```

In [115…

```
# Convert float64 and int32 columns to int64
X_testv=X_testv.astype({col:'int64' for col in X_testv.select_dtypes(include=['floa

# Check the data types of the columns to verify the conversion
print(X_testv.dtypes)
```

```
const                      int64
Id                         int64
LotFrontage                int64
LotArea                    int64
OverallQual                int64
                           ...
SaleCondition_Abnorml      int64
SaleCondition_AdjLand      int64
SaleCondition_Alloca       int64
SaleCondition_Family       int64
SaleCondition_Partial      int64
Length: 213, dtype: object
```

In [115…

```
trans_y_trainv=PowerTransformer(method = 'box-cox').fit_transform(y_trainv.values.r
```

In [115…

```
X_trainv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1022 entries, 1017 to 815
Columns: 212 entries, const to SaleCondition_Partial
dtypes: int64(212)
memory usage: 1.7 MB
```

In [115…

```
X_trainv
```

Out[1155]:

	const	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnr
1017	1	0	5814	8	5	1984	1984	
405	1	0	9991	4	4	1976	1993	
6	1	75	10084	8	5	2004	2005	
388	1	93	9382	7	5	1999	2000	
501	1	75	9803	7	5	2005	2005	
...	...	...	...	...	...	...	...	
1228	1	65	8769	9	5	2008	2008	
1077	1	0	15870	5	5	1969	1969	
1318	1	0	14781	8	5	2001	2002	
723	1	60	8172	4	6	1954	1972	
815	1	48	12137	7	5	1998	1998	

1022 rows × 212 columns

In [115…
```
X_testv
```

Out[1156]:

	const	Id	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	M
605	1	606	85	13600	7	6	1965	1990	
642	1	643	75	13860	8	7	1972	1995	
993	1	994	68	8846	6	5	2005	2006	
736	1	737	60	8544	3	4	1950	1950	
1239	1	1240	64	9037	8	5	2006	2006	
...	...	...	...	...	...	...	...	...	
805	1	806	91	12274	7	5	2008	2008	
112	1	113	77	9965	7	5	2007	2007	
348	1	349	36	2448	7	5	2003	2004	
205	1	206	99	11851	7	5	1990	1990	
622	1	623	71	7064	5	6	1977	1977	

438 rows × 213 columns

In [115…
```
X_testv=X_testv.drop(columns=['Id'])
```

In [115…
```
X_testv
```

Out[1158]:

	const	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnr
605	1	85	13600	7	6	1965	1990	
642	1	75	13860	8	7	1972	1995	
993	1	68	8846	6	5	2005	2006	
736	1	60	8544	3	4	1950	1950	
1239	1	64	9037	8	5	2006	2006	
...	...	...	...	...	...	...	...	
805	1	91	12274	7	5	2008	2008	
112	1	77	9965	7	5	2007	2007	
348	1	36	2448	7	5	2003	2004	
205	1	99	11851	7	5	1990	1990	
622	1	71	7064	5	6	1977	1977	

438 rows × 212 columns

In [115…

```
#Prediction using model 6
pred_tree = model_6.predict(X_testv)
pred_tree = pd.Series(boxcox.inverse_transform(pred_tree.reshape(-1, 1)).reshape(-1
                      index = testv.index)
pred_tree
```

Out[1159]:

```
605     195524.030116
642     275335.089086
993     170783.032869
736      98981.547558
1239    275335.089086
            ...
805     195524.030116
112     258590.895989
348     195524.030116
205     195524.030116
622     146671.518695
Length: 438, dtype: float64
```

In [116…

```
#Absolute error
abs_err_tree=abs(testv['SalePrice']-pred_tree)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_tree=mean_absolute_error(testv['SalePrice'],pred_tree)
mse_tree=mean_squared_error(testv['SalePrice'],pred_tree)
rmse_tree=np.sqrt(mse_tree)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                       pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_tree.mean(
                          'MSE' :                      mse_tree,
                          'RMSE' :                      rmse_tree,
                          'Median of AbsErrors' :      abs_err_tree.median(),
                          'SD of AbsErrors' :          abs_err_tree.std(),
                          'IQR of AbsErrors':          iqr(abs_err_tree),
                          'Min of AbsErrors':          abs_err_tree.min(),
                          'Max of AbsErrors':          abs_err_tree.max()},
```

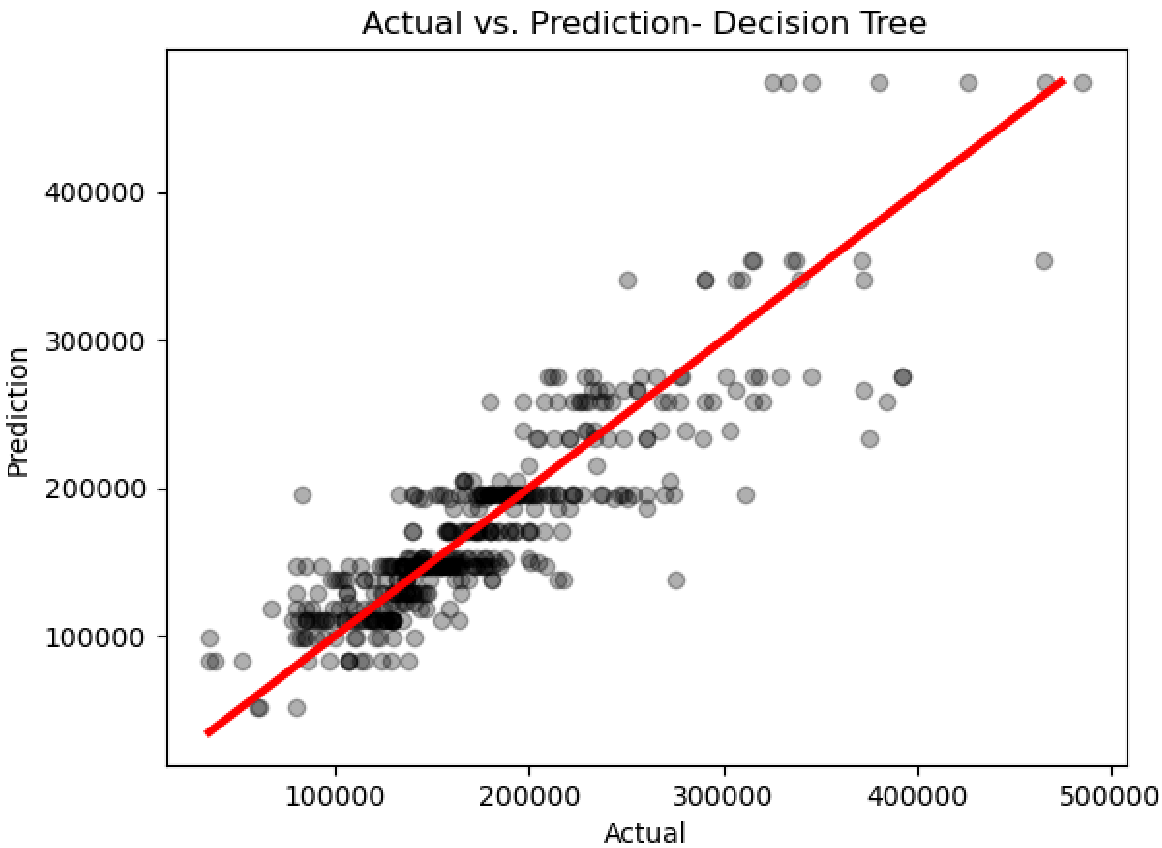
```
                                    index = ['Decision Tree'])])
models_comp
```

Out[1160]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR c AbsErroı
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.08604
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.04312
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.34915
LM_Ridge	14919.438525	4.566752e+08	20794.637168	10644.006449	15317.349954	13898.2244C
LM_LASSO	13721.483435	4.111831e+08	20277.650641	10065.606325	14947.042706	14111.4789S
Decision Tree	24744.257204	1.222742e+09	34967.727634	17104.607132	24735.817219	24528.4880E

In [116…

```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_tree,c='black',alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- Decision Tree')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_tree.max(),100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



Random Forest

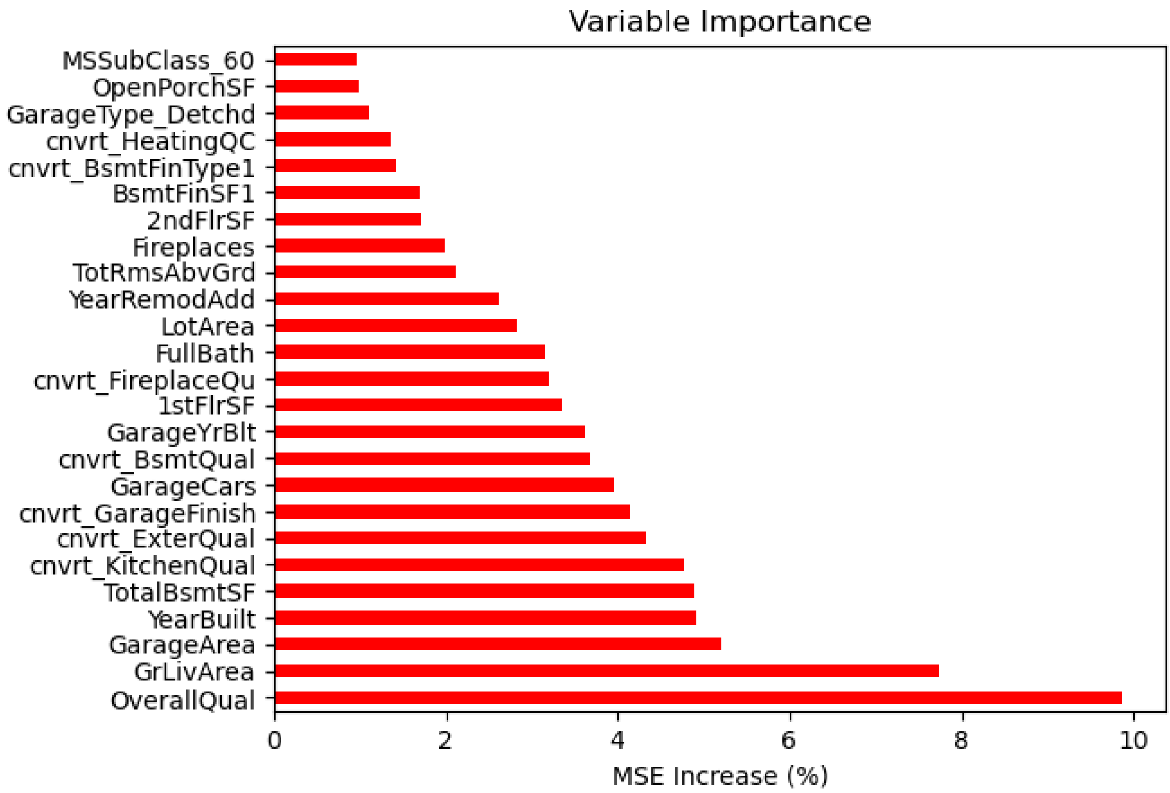
In [116…

```
#Sample Random Forest model (not tuned)
from sklearn.ensemble import RandomForestRegressor
rf_reg = RandomForestRegressor(n_estimators=300,
                               max_features=15,
                               criterion='squared_error',
                               max_depth=15,
                               min_samples_leaf=5,
                               ccp_alpha=0,
                               random_state =1234)
model_7=rf_reg.fit(X_trainv,trans_y_trainv.reshape(-1))
```

In [116…

```
#Calculate variable importance
importance=pd.DataFrame({'importance': model_7.feature_importances_ * 100},
                          index = X_trainv.columns)
 # importance.sort_values
importance_sorted = importance.sort_values(by='importance',ascending=False)
plt.figure(figsize=(30,25))
importance_sorted[:25].plot(kind='barh',color='r',legend=False)
plt.title('Variable Importance')
plt.xlabel('MSE Increase (%)')
plt.show()
```

```
<Figure size 3000x2500 with 0 Axes>
```



In [116…

```
#Create hyper-parameters grid
import itertools
n_estimators=[100,200,500,700]
max_features=['sqrt', 'log2']
max_depth=[2,5,10,20]
min_samples_leaf=[5,10,15]
params_grid=list(itertools.product(n_estimators,max_features,max_depth,min_samples_
params_grid=pd.DataFrame(data=params_grid,
                           index=range(1,97),
                           columns=['n_estimators',
                                    'max_features',
                                    'max_depth',
                                    'min_samples_leaf'])
params_grid
```

Out[1164]:

	n_estimators	max_features	max_depth	min_samples_leaf
1	100	sqrt	2	5
2	100	sqrt	2	10
3	100	sqrt	2	15
4	100	sqrt	5	5
5	100	sqrt	5	10
...	...	...	...	...
92	700	log2	10	10
93	700	log2	10	15
94	700	log2	20	5
95	700	log2	20	10
96	700	log2	20	15

96 rows × 4 columns

In [116…

```
#K-fold cross validation to choose the best model
from sklearn.model_selection import cross_val_score

import time #to measure the processing time
start_time = time.time()
cv_errors = np.zeros(shape = len(params_grid)) #to save cv results
for i in range(len(params_grid)):
    rf_reg=RandomForestRegressor(n_estimators=params_grid.iloc[i, 0],
                                   max_features=params_grid.iloc[i, 1],
                                   criterion='squared_error',
                                   max_depth=params_grid.iloc[i, 2],
                                   min_samples_leaf = params_grid.iloc[i, 3],
                                   ccp_alpha=0)
    scores = cross_val_score(estimator=rf_reg,
                             X=X_trainv,
                             y=trans_y_trainv,
                             scoring='neg_root_mean_squared_error',
                             cv=10, n_jobs=-1)
    cv_errors[i]=scores.mean()

end_time=time.time()
print('The Processing time is: ',end_time-start_time,'seconds')

cv_errors
```

```
The Processing time is:  249.83041548728943 seconds
```

Out[1165]:
```
array([-0.60381018, -0.60690393, -0.60527509, -0.44321644, -0.44930589,
       -0.46224956, -0.39462465, -0.42318739, -0.44478465, -0.39738872,
       -0.4230544 , -0.43830497, -0.67893773, -0.67577108, -0.68359319,
       -0.50176248, -0.51618475, -0.53435173, -0.44583828, -0.48365368,
       -0.51314453, -0.43841698, -0.48215898, -0.51010355, -0.6042227 ,
       -0.60538831, -0.60524574, -0.44377989, -0.45200596, -0.4628391 ,
       -0.39535818, -0.42256661, -0.44275874, -0.39343806, -0.42236925,
       -0.44306676, -0.67503845, -0.67637676, -0.67930367, -0.50012036,
       -0.51453952, -0.5330671 , -0.44257159, -0.48233494, -0.51067818,
       -0.44002402, -0.47804257, -0.51025026, -0.6055131 , -0.60340438,
       -0.60604228, -0.44173546, -0.45084015, -0.46001998, -0.39524779,
       -0.42221097, -0.44241641, -0.39224295, -0.42160915, -0.44193323,
       -0.67546337, -0.67787746, -0.67492205, -0.50133297, -0.51627132,
       -0.53312382, -0.44122402, -0.48197998, -0.5098095 , -0.43902599,
       -0.47848481, -0.51104146, -0.6047886 , -0.60582067, -0.60467717,
       -0.44235499, -0.45064633, -0.45904883, -0.39513627, -0.42139881,
       -0.44129663, -0.39344396, -0.42157587, -0.44200032, -0.67539977,
       -0.67310423, -0.67951514, -0.50081696, -0.51611935, -0.53094788,
       -0.4409903 , -0.48075506, -0.51000618, -0.437522  , -0.47923922,
       -0.51134618])
```

In [116…
```
#Best model
params_grid.iloc[np.argmax(cv_errors),:]
```

Out[1166]:
```
n_estimators         500
max_features        sqrt
max_depth             20
min_samples_leaf       5
Name: 58, dtype: object
```

In [116…
```
#Train model 7
rf_reg=RandomForestRegressor(n_estimators=params_grid.iloc[np.argmax(cv_errors),0],
                              max_features=params_grid.iloc[np.argmax(cv_errors),1
                              criterion='squared_error',
                              max_depth=params_grid.iloc[np.argmax(cv_errors),2],
                              min_samples_leaf=params_grid.iloc[np.argmax(cv_error
                              ccp_alpha=0,
                              random_state=1234)
model_7=rf_reg.fit(X_trainv,trans_y_trainv.reshape(-1))
```

Evaluation on the Validation Set (Random Forest)

In [116…
```
common_columns = X_trainv.columns.intersection(X_testv.columns)
X_trainv = X_trainv[common_columns]
X_testv = X_testv[common_columns]
```

In [116…
```
#Prediction using model 7
pred_rf=model_7.predict(X_testv)
pred_rf=pd.Series(boxcox.inverse_transform(pred_rf.reshape(-1, 1)).reshape(-1),
                    index=testv.index)
pred_rf
```

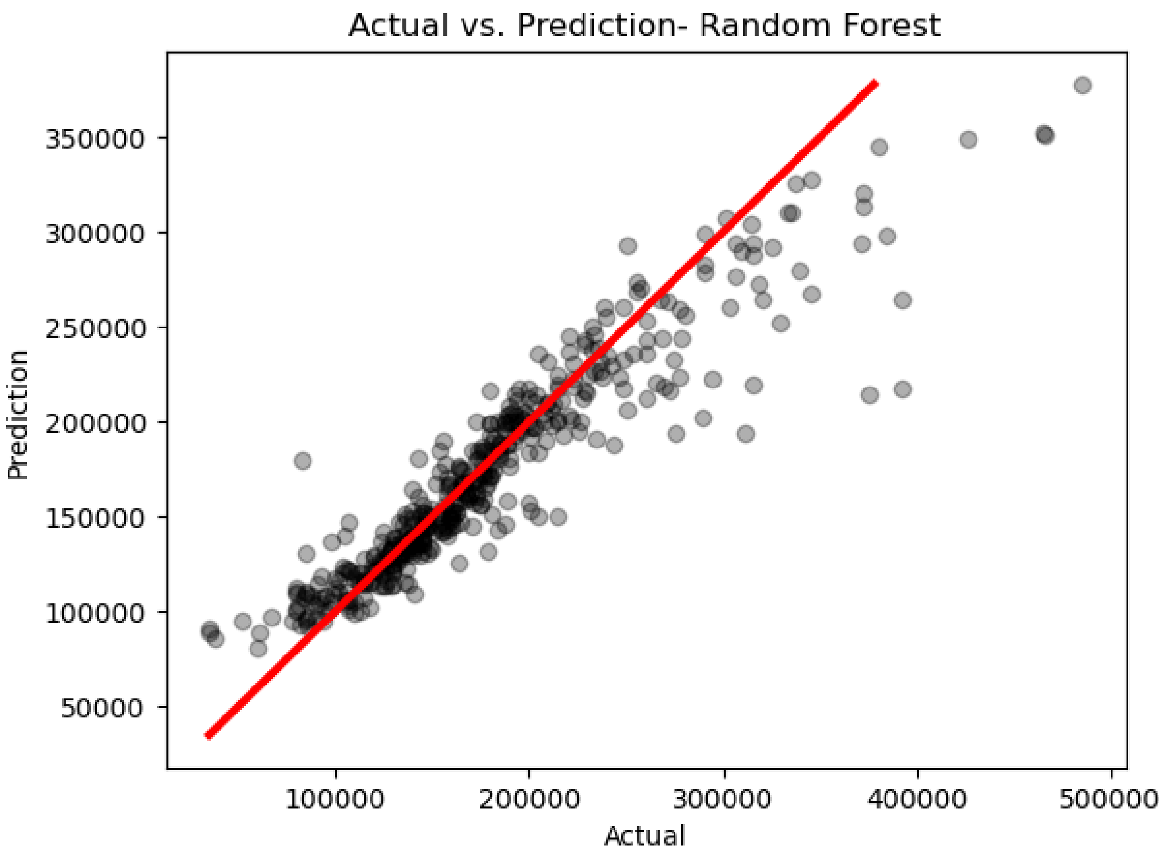
Out[1169]:
```
605     184044.820799
642     267396.627674
993     180236.913848
736      95530.959008
1239    220568.923598
            ...
805     212189.324233
112     298199.711119
348     184428.838293
205     178611.779143
622     135567.932106
Length: 438, dtype: float64
```

In [117…
```
#Absolute error
abs_err_rf=abs(testv['SalePrice']-pred_rf)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_rf=mean_absolute_error(testv['SalePrice'],pred_rf)
mse_rf=mean_squared_error(testv['SalePrice'],pred_rf)
rmse_rf=np.sqrt(mse_rf)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                        pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_rf.mean(),
                           'MSE' :                          mse_rf,
                           'RMSE' :                          rmse_rf,
                           'Median of AbsErrors' :   abs_err_rf.median(),
                           'SD of AbsErrors' :       abs_err_rf.std(),
                           'IQR of AbsErrors':       iqr(abs_err_rf),
                           'Min of AbsErrors':       abs_err_rf.min(),
                           'Max of AbsErrors':       abs_err_rf.max()},
                           index = ['Random Forest'])])
models_comp
```

Out[1170]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR c AbsErro
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.08604
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.04312
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.34915
LM_Ridge	14919.438525	4.566752e+08	20794.637168	10644.006449	15317.349954	13898.22440
LM_LASSO	13721.483435	4.111831e+08	20277.650641	10065.606325	14947.042706	14111.47899
Decision Tree	24744.257204	1.222742e+09	34967.727634	17104.607132	24735.817219	24528.48808
Random Forest	16830.078058	7.699673e+08	27748.284511	10214.919175	22086.863334	13755.28387

In [117…
```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_rf, c='black', alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- Random Forest')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_rf.max(),100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



Principal Component Regression (PCR)

In [117…

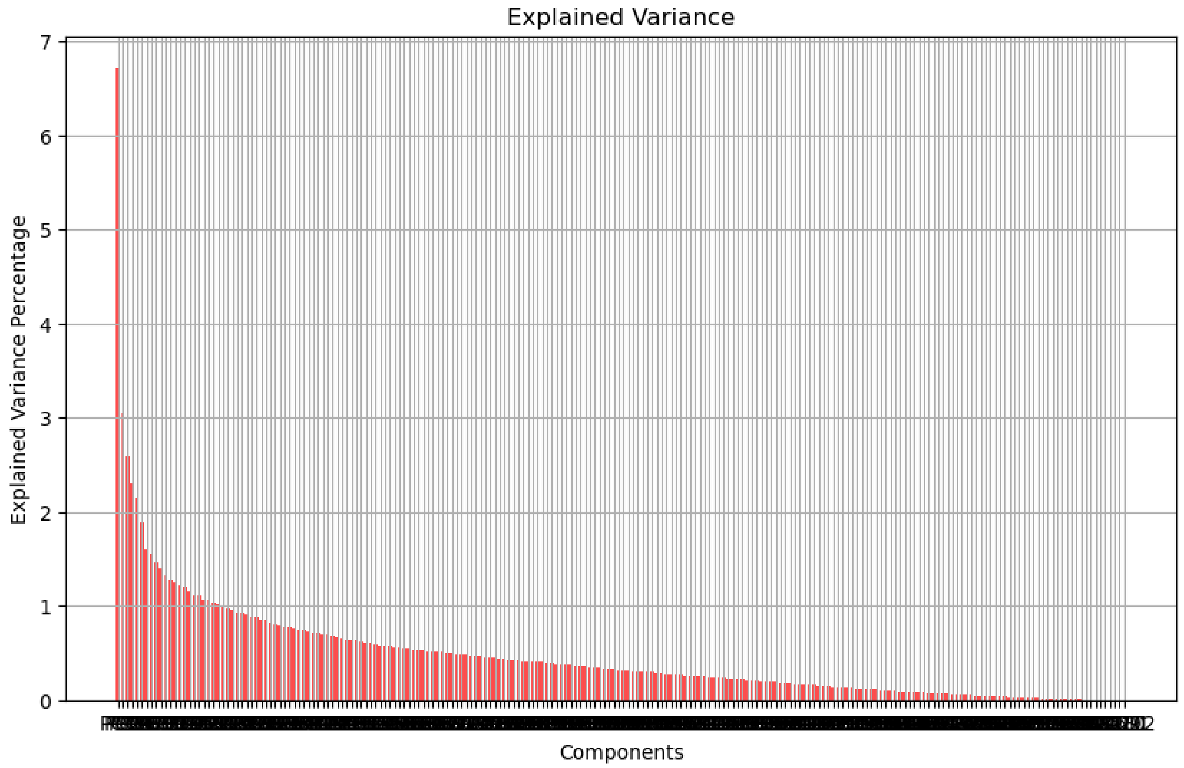
```
#Scale data
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
X_trainv_scaled=scaler.fit_transform(X_trainv)
```

In [117…

```
#PCA
from sklearn.decomposition import PCA
pca=PCA()
pca_model=pca.fit(X_trainv_scaled)
```

In [117…

```
#Explaind variance percentage
plt.figure(figsize = (10, 6))
plt.bar(x = range(1,213),
        height=pca_model.explained_variance_ratio_ * 100,
        color='red',
        alpha =0.7)
plt.title('Explained Variance')
plt.xlabel('Components')
plt.ylabel('Explained Variance Percentage')
plt.xticks(ticks=range(1,213), labels= ['PC' + str(_) for _ in range(1,213)])
plt.grid()
plt.show()
```



In [117…

```
#Cumulative explaind variance percentage
np.cumsum(np.round(pca_model.explained_variance_ratio_ * 100, 1))
```

Out[1175]:

```
array([ 6.7,  9.7, 12.3, 14.6, 16.7, 18.6, 20.2, 21.7, 23.2, 24.6, 25.9,
       27.2, 28.4, 29.6, 30.8, 32. , 33.1, 34.2, 35.3, 36.4, 37.4, 38.4,
       39.4, 40.4, 41.4, 42.3, 43.2, 44.1, 45. , 45.9, 46.8, 47.6, 48.4,
       49.2, 50. , 50.8, 51.6, 52.4, 53.1, 53.8, 54.5, 55.2, 55.9, 56.6,
       57.3, 58. , 58.7, 59.4, 60. , 60.6, 61.2, 61.8, 62.4, 63. , 63.6,
       64.2, 64.8, 65.4, 66. , 66.6, 67.1, 67.6, 68.1, 68.6, 69.1, 69.6,
       70.1, 70.6, 71.1, 71.6, 72.1, 72.6, 73.1, 73.6, 74.1, 74.6, 75.1,
       75.6, 76.1, 76.5, 76.9, 77.3, 77.7, 78.1, 78.5, 78.9, 79.3, 79.7,
       80.1, 80.5, 80.9, 81.3, 81.7, 82.1, 82.5, 82.9, 83.3, 83.7, 84.1,
       84.4, 84.7, 85. , 85.3, 85.6, 85.9, 86.2, 86.5, 86.8, 87.1, 87.4,
       87.7, 88. , 88.3, 88.6, 88.9, 89.2, 89.5, 89.8, 90.1, 90.4, 90.7,
       91. , 91.3, 91.5, 91.7, 91.9, 92.1, 92.3, 92.5, 92.7, 92.9, 93.1,
       93.3, 93.5, 93.7, 93.9, 94.1, 94.3, 94.5, 94.7, 94.9, 95.1, 95.3,
       95.5, 95.7, 95.9, 96.1, 96.3, 96.5, 96.6, 96.7, 96.8, 96.9, 97. ,
       97.1, 97.2, 97.3, 97.4, 97.5, 97.6, 97.7, 97.8, 97.9, 98. , 98.1,
       98.2, 98.3, 98.4, 98.5, 98.6, 98.7, 98.8, 98.9, 99. , 99.1, 99.2,
       99.3, 99.4, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5,
       99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5,
       99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5, 99.5,
       99.5, 99.5, 99.5])
```

In [117…

```
#Principal axes in feature space
print(pca_model.components_.shape)
print(pca_model.components_)
```

```
(212, 212)
[[ 1.15443546e-18  4.94498386e-02  6.00846956e-02 ... -1.23535551e-02
   2.31786067e-03  1.06864584e-01]
 [-5.83004331e-18  9.29742532e-02  9.23438187e-03 ...  9.44896264e-03
  -9.09412816e-03  6.69392176e-02]
 [-9.31616269e-19  6.54782920e-02  2.25395323e-01 ...  5.19661933e-02
   2.43262721e-02 -9.31866450e-02]
 ...
 [ 0.00000000e+00 -2.98372438e-16 -2.58624840e-18 ...  1.90819582e-16
   6.93889390e-18 -3.01841885e-16]
 [ 9.99998974e-01  2.79909185e-18  7.71864276e-18 ... -8.17217388e-18
  -4.92973175e-18  7.20994445e-18]
 [ 0.00000000e+00  1.80411242e-16 -5.49559333e-17 ...  1.49186219e-16
  -6.93889390e-18 -8.67361738e-18]]
```

In [117…

```python
#Transfrom data into new dimensions
X_trainv_pca=pca_model.transform(X_trainv_scaled)
X_trainv_pca.shape
```

Out[1177]:

```
(1022, 212)
```

In [117…

```python
#K-fold cross validation to choose the best model
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression

cv_errors=np.zeros(shape=X_trainv_pca.shape[1]) #to save cv results

import time #to measure the processing time
start_time=time.time()

for i in range(X_trainv_pca.shape[1]):
    linear_reg=LinearRegression()
    scores=cross_val_score(estimator=linear_reg,
                           X=X_trainv_pca[:, : i + 1],
                           y=trans_y_trainv,
                           scoring='neg_root_mean_squared_error',
                           cv=5,n_jobs = -1)
    cv_errors[i]=scores.mean()

end_time=time.time()
print('The Processing time is: ',end_time-start_time,'seconds')
cv_errors
```

```
The Processing time is:  14.828158140182495 seconds
```

Out[1178]:

```
array([-4.75555436e-01, -4.50726241e-01, -4.28165597e-01, -4.29811237e-01,
       -4.28184645e-01, -4.21903209e-01, -4.23433943e-01, -4.24734629e-01,
       -4.32797381e-01, -4.33007730e-01, -4.34284426e-01, -4.26557383e-01,
       -4.28637149e-01, -4.29400507e-01, -4.24347868e-01, -4.02252046e-01,
       -4.01704625e-01, -4.03268480e-01, -4.01794301e-01, -4.01228889e-01,
       -4.01518653e-01, -4.01858684e-01, -4.01737750e-01, -4.02988213e-01,
       -4.08942361e-01, -4.17700301e-01, -4.18346508e-01, -4.01139262e-01,
       -4.10020213e-01, -4.19456680e-01, -4.17237964e-01, -4.11653602e-01,
       -4.05944287e-01, -4.04801934e-01, -4.05794866e-01, -4.07542987e-01,
       -4.06382189e-01, -4.08624008e-01, -4.09258233e-01, -3.98455185e-01,
       -3.98650234e-01, -4.01077027e-01, -4.03044911e-01, -4.09752947e-01,
       -4.11892524e-01, -4.06566454e-01, -4.10533476e-01, -4.13499834e-01,
       -4.24263217e-01, -4.50455572e-01, -4.56684245e-01, -4.58752311e-01,
       -4.56789004e-01, -4.61110616e-01, -4.54498726e-01, -4.33085680e-01,
       -4.34686598e-01, -4.30376514e-01, -4.30771455e-01, -4.25646615e-01,
       -4.57497343e-01, -4.59135280e-01, -4.58012195e-01, -4.39999731e-01,
       -4.38488180e-01, -4.90017675e-01, -4.94825120e-01, -4.93508273e-01,
       -4.92923568e-01, -4.92422185e-01, -4.89913547e-01, -4.94633797e-01,
       -5.01926131e-01, -4.94667194e-01, -5.12751976e-01, -5.59091803e-01,
       -5.59258525e-01, -5.65254291e-01, -5.64959644e-01, -5.67047728e-01,
       -5.65622824e-01, -5.40770296e-01, -5.51822651e-01, -5.30975580e-01,
       -5.39413519e-01, -5.47460682e-01, -5.48391534e-01, -5.64537651e-01,
       -5.36867508e-01, -5.45034114e-01, -5.61725140e-01, -5.57413801e-01,
       -5.62006679e-01, -5.66456520e-01, -5.61162496e-01, -5.60690478e-01,
       -5.59614053e-01, -5.89324794e-01, -5.92287723e-01, -6.12056215e-01,
       -6.13128039e-01, -6.14544453e-01, -5.91091749e-01, -5.97068135e-01,
       -6.00290977e-01, -5.77071394e-01, -5.70903987e-01, -5.02328160e-01,
       -5.02537284e-01, -4.97859189e-01, -5.00288744e-01, -5.23852585e-01,
       -5.39831907e-01, -5.31963617e-01, -5.35575863e-01, -5.30336930e-01,
       -5.37656399e-01, -5.44147023e-01, -5.46195833e-01, -5.58238068e-01,
       -5.75737015e-01, -5.70483486e-01, -5.66266322e-01, -5.66905777e-01,
       -5.85880211e-01, -6.02149919e-01, -4.90750528e-01, -5.04943914e-01,
       -5.14884933e-01, -5.15815833e-01, -6.12557510e-01, -6.36618508e-01,
       -6.56533086e-01, -6.78697811e-01, -6.60769604e-01, -6.79372708e-01,
       -7.39914140e-01, -7.17062887e-01, -7.92432972e-01, -7.92211919e-01,
       -8.06604796e-01, -8.39528040e-01, -9.85763616e-01, -8.47522892e-01,
       -8.76134801e-01, -8.71974034e-01, -8.91930783e-01, -8.31805367e-01,
       -8.54294298e-01, -7.67218553e-01, -7.98509164e-01, -7.96294367e-01,
       -8.02452567e-01, -7.88226577e-01, -8.54730446e-01, -8.89951436e-01,
       -9.38571987e-01, -1.04575702e+00, -1.12393531e+00, -1.15113419e+00,
       -1.65695548e+00, -1.32160218e+00, -1.44043812e+00, -1.74893175e+00,
       -2.36779314e+00, -1.88474819e+00, -1.85046773e+00, -2.03589938e+00,
       -2.29461584e+00, -2.21862891e+00, -2.13239958e+00, -1.63764955e+00,
       -2.06605940e+00, -2.11644498e+00, -3.45369709e+00, -4.64149765e+00,
       -5.17457390e+00, -5.40743949e+00, -5.84254955e+00, -5.71634025e+00,
       -5.84034456e+00, -6.82390732e+00, -8.16708632e+00, -1.41171468e+01,
       -1.50943379e+01, -1.45259901e+01, -1.72501649e+01, -1.64638745e+01,
       -2.48073804e+01, -2.41969046e+01, -3.26153358e+01, -4.20586114e+01,
       -4.94382467e+01, -2.38542469e+01, -2.93656164e+01, -8.34457150e+01,
       -7.85292794e+01, -1.95102022e+03, -2.01963088e+10, -5.11865922e+11,
       -9.62345493e+11, -2.27735405e+12, -6.29770843e+12, -8.95587814e+12,
       -4.12206214e+12, -2.24500723e+12, -2.65063139e+12, -2.44332900e+12,
       -1.88016849e+12, -2.46677035e+12, -2.65796127e+12, -2.39730835e+12])
```

In [117…

```
#Best number of components
components_number=np.argmax(cv_errors)
components_number
```

Out[1179]:

```
39
```

In [118…

```
#Correlation btw components
np.round(pd.DataFrame(X_trainv_pca[:, : components_number]).corr(), 3)
```

Out[1180]:

	0	1	2	3	4	5	6	7	8	9	...	29	30	31	32	33	34	35
0	1.0	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	0.0	...	0.0	-0.0	0.0	0.0	0.0	-0.0	-0.0
1	-0.0	1.0	0.0	-0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	...	-0.0	0.0	0.0	0.0	-0.0	0.0	-0.0
2	-0.0	0.0	1.0	-0.0	0.0	-0.0	0.0	0.0	0.0	-0.0	...	-0.0	-0.0	0.0	-0.0	0.0	-0.0	0.0
3	-0.0	-0.0	-0.0	1.0	0.0	-0.0	0.0	0.0	0.0	-0.0	...	0.0	0.0	0.0	0.0	-0.0	0.0	-0.0
4	0.0	0.0	0.0	0.0	1.0	-0.0	-0.0	0.0	-0.0	0.0	...	-0.0	-0.0	-0.0	-0.0	-0.0	0.0	-0.0
5	-0.0	-0.0	-0.0	-0.0	-0.0	1.0	-0.0	0.0	-0.0	-0.0	...	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0
6	-0.0	-0.0	0.0	0.0	-0.0	-0.0	1.0	-0.0	-0.0	0.0	...	0.0	0.0	-0.0	-0.0	0.0	0.0	0.0
7	-0.0	0.0	0.0	0.0	0.0	0.0	-0.0	1.0	0.0	0.0	...	-0.0	0.0	0.0	-0.0	0.0	-0.0	0.0
8	-0.0	-0.0	0.0	0.0	-0.0	-0.0	-0.0	0.0	1.0	0.0	...	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
9	0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	0.0	0.0	1.0	...	-0.0	0.0	-0.0	-0.0	0.0	0.0	-0.0
10	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	0.0	0.0	...	-0.0	0.0	0.0	-0.0	-0.0	0.0	-0.0
11	-0.0	-0.0	0.0	0.0	-0.0	0.0	-0.0	-0.0	-0.0	0.0	...	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	0.0
12	0.0	0.0	-0.0	0.0	-0.0	-0.0	-0.0	0.0	0.0	0.0	...	-0.0	0.0	0.0	0.0	-0.0	-0.0	-0.0
13	-0.0	-0.0	0.0	0.0	0.0	-0.0	-0.0	-0.0	0.0	-0.0	...	0.0	-0.0	0.0	0.0	0.0	-0.0	-0.0
14	0.0	-0.0	0.0	-0.0	0.0	0.0	0.0	0.0	-0.0	0.0	...	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
15	-0.0	0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	0.0	-0.0	...	-0.0	-0.0	-0.0	-0.0	-0.0	0.0	0.0
16	0.0	-0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	0.0	0.0	...	0.0	-0.0	-0.0	0.0	-0.0	-0.0	0.0
17	0.0	0.0	-0.0	0.0	0.0	0.0	-0.0	0.0	-0.0	-0.0	...	-0.0	0.0	-0.0	-0.0	-0.0	0.0	-0.0
18	0.0	0.0	0.0	0.0	0.0	0.0	-0.0	0.0	-0.0	0.0	...	-0.0	0.0	-0.0	0.0	0.0	-0.0	-0.0
19	0.0	-0.0	-0.0	-0.0	-0.0	0.0	-0.0	0.0	-0.0	0.0	...	-0.0	-0.0	0.0	0.0	-0.0	0.0	0.0
20	0.0	0.0	-0.0	-0.0	0.0	0.0	-0.0	0.0	-0.0	-0.0	...	0.0	-0.0	-0.0	0.0	0.0	-0.0	0.0
21	-0.0	-0.0	0.0	-0.0	-0.0	0.0	0.0	-0.0	-0.0	0.0	...	-0.0	0.0	0.0	0.0	-0.0	-0.0	0.0
22	-0.0	-0.0	-0.0	-0.0	0.0	-0.0	0.0	0.0	-0.0	0.0	...	0.0	0.0	0.0	-0.0	0.0	0.0	-0.0
23	-0.0	-0.0	0.0	-0.0	-0.0	0.0	0.0	-0.0	-0.0	0.0	...	0.0	0.0	-0.0	0.0	0.0	-0.0	0.0
24	0.0	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	0.0	0.0	-0.0	...	-0.0	0.0	-0.0	-0.0	0.0	-0.0	-0.0
25	0.0	-0.0	-0.0	-0.0	0.0	0.0	-0.0	0.0	0.0	-0.0	...	-0.0	0.0	0.0	-0.0	-0.0	0.0	-0.0
26	0.0	-0.0	-0.0	0.0	-0.0	-0.0	0.0	0.0	0.0	-0.0	...	-0.0	-0.0	0.0	-0.0	-0.0	0.0	0.0
27	-0.0	0.0	0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	-0.0	...	-0.0	0.0	0.0	-0.0	0.0	0.0	-0.0
28	-0.0	0.0	-0.0	0.0	0.0	-0.0	0.0	-0.0	0.0	-0.0	...	0.0	0.0	-0.0	0.0	-0.0	-0.0	0.0
29	0.0	-0.0	-0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	-0.0	...	1.0	-0.0	-0.0	0.0	0.0	-0.0	0.0
30	-0.0	0.0	-0.0	0.0	-0.0	0.0	0.0	0.0	-0.0	0.0	...	-0.0	1.0	-0.0	-0.0	0.0	0.0	0.0
31	0.0	0.0	0.0	0.0	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	...	-0.0	-0.0	1.0	0.0	0.0	0.0	-0.0
32	0.0	0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0	...	0.0	-0.0	0.0	1.0	0.0	0.0	-0.0
33	0.0	-0.0	0.0	-0.0	-0.0	-0.0	0.0	0.0	-0.0	0.0	...	0.0	0.0	0.0	0.0	1.0	0.0	-0.0
34	-0.0	0.0	-0.0	0.0	0.0	-0.0	0.0	-0.0	-0.0	0.0	...	-0.0	0.0	0.0	0.0	0.0	1.0	-0.0
35	-0.0	-0.0	0.0	-0.0	-0.0	-0.0	0.0	0.0	-0.0	-0.0	...	0.0	0.0	-0.0	-0.0	-0.0	-0.0	1.0

	0	1	2	3	4	5	6	7	8	9	...	29	30	31	32	33	34	35
36	-0.0	0.0	0.0	0.0	0.0	-0.0	-0.0	-0.0	-0.0	-0.0	...	0.0	0.0	0.0	-0.0	-0.0	-0.0	-0.0
37	-0.0	0.0	-0.0	0.0	0.0	0.0	-0.0	-0.0	-0.0	0.0	...	-0.0	-0.0	-0.0	0.0	-0.0	-0.0	0.0
38	0.0	0.0	-0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	-0.0	-0.0	0.0	-0.0	-0.0

39 rows × 39 columns

In [118…

```
#Best model
from sklearn.linear_model import LinearRegression
linear_reg=LinearRegression()
model_8=linear_reg.fit(X_trainv_pca[:, : components_number],
                         trans_y_trainv)
```

In [118…

```
#Predict on test - model 8

#Scale data
X_testv_scaled=scaler.transform(X_testv)

#Transfrom data into new dimensions
X_testv_pca=pca_model.transform(X_testv_scaled)

#Prediction on test
pred_pcr=model_8.predict(X_testv_pca[:, : components_number])
pred_pcr=pd.Series(boxcox.inverse_transform(pred_pcr.reshape(-1, 1)).reshape(-1),
                    index = testv.index)
pred_pcr
```

Out[1182]:

```
605     186033.737058
642     308743.224699
993     169282.106628
736      91164.500356
1239    242909.085184
            ...
805     206845.634642
112     339181.386551
348     186033.660323
205     167994.133523
622     127149.357630
Length: 438, dtype: float64
```

In [118…

```
pred_pcr.shape
```

Out[1183]:

```
(438,)
```

Evaluation on the Validation Set (PCR)

In [118…

```
#Absolute error
abs_err_pcr=abs(testv['SalePrice']-pred_pcr)

# Calculate MAE, MSE, RMSE and Absolute error median, sd, IQR, min, max
mae_pcr=mean_absolute_error(testv['SalePrice'],pred_pcr)
mse_pcr=mean_squared_error(testv['SalePrice'],pred_pcr)
rmse_pcr=np.sqrt(mse_pcr)

#Absolute error mean, median, sd, IQR, max, min
models_comp=pd.concat([models_comp,
                         pd.DataFrame({'Mean of AbsErrors(MAE)': abs_err_pcr.mean()
                            'MSE' :                        mse_pcr,
```

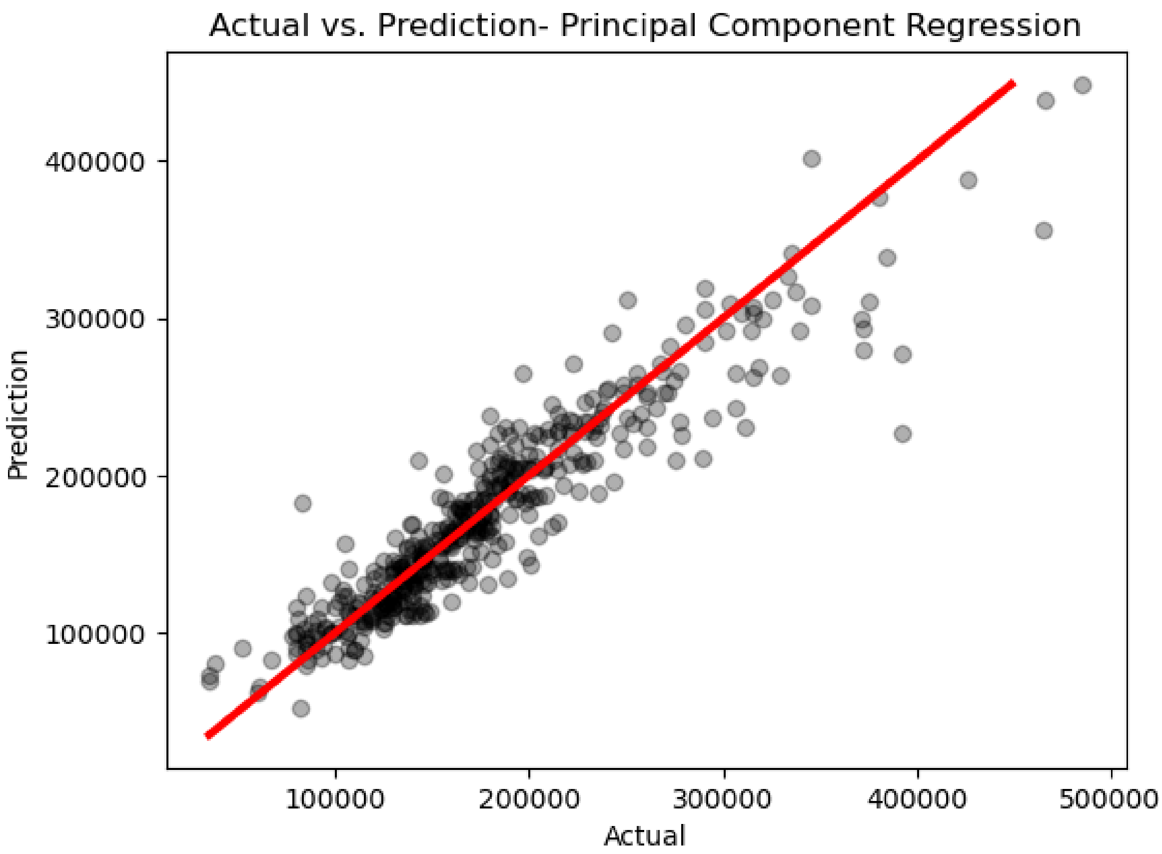
```
                            'RMSE' :                 rmse_pcr,
                            'Median of AbsErrors' :  abs_err_pcr.median(),
                            'SD of AbsErrors' :      abs_err_pcr.std(),
                            'IQR of AbsErrors':      iqr(abs_err_pcr),
                            'Min of AbsErrors':      abs_err_pcr.min(),
                            'Max of AbsErrors':      abs_err_pcr.max()},
                            index = ['PCR'])])
models_comp
```

Out[1184]:

	Mean of AbsErrors(MAE)	MSE	RMSE	Median of AbsErrors	SD of AbsErrors	IQR o AbsErro
LM_t-Test	14263.503558	4.293243e+08	20720.142797	10553.322434	15046.383761	13948.08604
LM_FWD	14448.051321	4.367638e+08	20898.894986	10006.922094	15117.519774	15270.04312
LM_BWD	14537.233266	4.324169e+08	20794.637168	10646.353527	14885.956503	15473.34915
LM_Ridge	14919.438525	4.566752e+08	20794.637168	10644.006449	15317.349954	13898.22440
LM_LASSO	13721.483435	4.111831e+08	20277.650641	10065.606325	14947.042706	14111.47899
Decision Tree	24744.257204	1.222742e+09	34967.727634	17104.607132	24735.817219	24528.48808
Random Forest	16830.078058	7.699673e+08	27748.284511	10214.919175	22086.863334	13755.28387
PCR	17693.876085	6.633898e+08	25756.354913	11838.665965	18738.148448	15608.83447

In [118…

```
# Actual vs. Prediction
plt.scatter(x=testv['SalePrice'],y=pred_pcr,c='black', alpha = 0.3)
plt.xlabel('Actual')
plt.ylabel('Prediction')
plt.title('Actual vs. Prediction- Principal Component Regression')

#Add 45 degree line
xp = np.linspace(testv['SalePrice'].min(),pred_pcr.max(),100)
plt.plot(xp,xp,c='red',linewidth = 3)
plt.show()
```



Model Evaluation

Test data understanding- Phase1

2: Test Data Set Description

In [118…

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
import seaborn as sns
```

In [118…

```
# Test data
#Read Test data from file
test=pd.read_csv('test.csv')
# test.info()
test.head(10)
```

Out[1187]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	U
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	
5	1466	60	RL	75.0	10000	Pave	NaN	IR1	Lvl	
6	1467	20	RL	NaN	7980	Pave	NaN	IR1	Lvl	
7	1468	60	RL	63.0	8402	Pave	NaN	IR1	Lvl	
8	1469	20	RL	85.0	10176	Pave	NaN	Reg	Lvl	
9	1470	20	RL	70.0	8400	Pave	NaN	Reg	Lvl	

10 rows × 80 columns

In [118…

```
test['Id'].nunique()
```

Out[1188]:

```
1459
```

In [118…

```
np.sum(test.duplicated())
```

Out[1189]:

```
0
```

3: Check for Missing Values

In [119…

```
# Step 1:Determine the type of MVs
# Know the cause
np.sum(test.isnull())
```

Out[1190]:

```
Id                 0
MSSubClass         0
MSZoning           4
LotFrontage      227
LotArea            0
                ... 
MiscVal            0
MoSold             0
YrSold             0
SaleType           1
SaleCondition      0
Length: 80, dtype: int64
```

In [119…

```
# Step 2: Determine the extent of MVs
#S ummary of MVs in each column
mvs_summary = pd.DataFrame({'freq':np.sum(test.isnull())})
mvs_summary['pct'] = round(mvs_summary['freq']/test.shape[0]* 100, 1)
mvs_summary.sort_values(by ='pct', ascending = False)
```

Out[1191]:

	freq	pct
PoolQC	1456	99.8
MiscFeature	1408	96.5
Alley	1352	92.7
Fence	1169	80.1
MasVnrType	894	61.3
...	...	...
Electrical	0	0.0
1stFlrSF	0	0.0
2ndFlrSF	0	0.0
LowQualFinSF	0	0.0
SaleCondition	0	0.0

80 rows × 2 columns

In [119…

```
#Summary of MVs for each case
test.loc[:,'mvs']=test.apply(lambda row: np.sum(row.isnull()), axis = 1)
test.sort_values(by ='mvs',ascending = False)
test.loc[:,'mvs_pct']=round(test.apply(lambda row: np.sum(row.isnull())/(test.shape
#(data.shape[1]-3), 3 refers to 3 extra columns (id, mvs, mvs-pct)
test.sort_values(by='mvs',ascending = False)
```

Out[1192]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
387	1848	20	RL	NaN	9000	Pave	NaN	Reg	Lv
729	2190	90	RL	65.0	6012	Pave	NaN	Reg	Lv
733	2194	50	RL	57.0	8050	Pave	NaN	Reg	Lv
1431	2892	30	C (all)	69.0	12366	Pave	NaN	Reg	Lv
660	2121	20	RM	99.0	5940	Pave	NaN	IR1	Lv
...	...	...	...	...	...	...	...	...	.
131	1592	30	RL	67.0	4853	Pave	NaN	Reg	Bnl
472	1933	60	RL	70.0	10457	Pave	NaN	IR1	Lv
305	1766	20	RL	92.0	10573	Pave	NaN	IR1	Lv
613	2074	20	RL	76.0	11355	Pave	NaN	IR1	Lv
1250	2711	80	RL	100.0	14330	Pave	NaN	IR1	Lov

1459 rows × 82 columns

Decision: Modify elements that are known as null but still contain information (Descriptive Statistics)

In [119…
```python
# Modifying 'MasVnrType'
test.loc[test['MasVnrArea']==0].shape[0]
```

Out[1193]:
```
877
```

In [119…
```python
test.loc[test['MasVnrArea'] == 0,'MasVnrType'] ='none'
np.sum(test['MasVnrType'].isnull())
```

Out[1194]:
```
18
```

In [119…
```python
np.sum(test['MasVnrArea'].isnull())
```

Out[1195]:
```
15
```

In [119…
```python
test.loc[test['MasVnrArea'].isnull()]
```

Out[1196]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
231	1692	60	RL	NaN	12891	Pave	NaN	IR1	Lv
246	1707	20	FV	90.0	7993	Pave	NaN	IR1	Lv
422	1883	60	RL	70.0	8749	Pave	NaN	Reg	Lv
532	1993	60	RL	NaN	7750	Pave	NaN	Reg	Lv
544	2005	20	RL	87.0	10037	Pave	NaN	Reg	Lv
581	2042	60	FV	NaN	7500	Pave	NaN	Reg	Lv
851	2312	60	RL	59.0	15810	Pave	NaN	IR1	Lv
865	2326	80	RL	NaN	11950	Pave	NaN	IR1	Lv
880	2341	20	RL	85.0	9965	Pave	NaN	Reg	Lv
889	2350	60	FV	112.0	12217	Pave	NaN	IR1	Lv
908	2369	120	FV	30.0	5330	Pave	Pave	IR2	Lv
1132	2593	20	RL	68.0	8298	Pave	NaN	IR1	HL
1197	2658	60	RL	103.0	12867	Pave	NaN	IR1	Lv
1226	2687	20	RL	49.0	15218	Pave	NaN	IR1	Lv
1402	2863	20	RL	75.0	8050	Pave	NaN	Reg	Lv

15 rows × 82 columns

In [119…
```python
test.loc[test['MasVnrType'].isnull()]
```

Out[1197]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
209	1670	20	RL	102.0	13514	Pave	NaN	IR1	Lv
231	1692	60	RL	NaN	12891	Pave	NaN	IR1	Lv
246	1707	20	FV	90.0	7993	Pave	NaN	IR1	Lv
422	1883	60	RL	70.0	8749	Pave	NaN	Reg	Lv
532	1993	60	RL	NaN	7750	Pave	NaN	Reg	Lv
544	2005	20	RL	87.0	10037	Pave	NaN	Reg	Lv
581	2042	60	FV	NaN	7500	Pave	NaN	Reg	Lv
851	2312	60	RL	59.0	15810	Pave	NaN	IR1	Lv
865	2326	80	RL	NaN	11950	Pave	NaN	IR1	Lv
880	2341	20	RL	85.0	9965	Pave	NaN	Reg	Lv
889	2350	60	FV	112.0	12217	Pave	NaN	IR1	Lv
908	2369	120	FV	30.0	5330	Pave	Pave	IR2	Lv
992	2453	20	RM	52.0	8626	Pave	NaN	Reg	Lv
1132	2593	20	RL	68.0	8298	Pave	NaN	IR1	HL
1150	2611	20	RL	124.0	27697	Pave	NaN	Reg	Lv
1197	2658	60	RL	103.0	12867	Pave	NaN	IR1	Lv
1226	2687	20	RL	49.0	15218	Pave	NaN	IR1	Lv
1402	2863	20	RL	75.0	8050	Pave	NaN	Reg	Lv

18 rows × 82 columns

In [119…
```
# test[['MasVnrArea','MasVnrType']]
```

In [119…
```
# Modifying'FireplaceQu'
test.loc[test['Fireplaces']==0,'FireplaceQu']='NA'
```

In [120…
```
np.sum(test['Fireplaces']==0)
```

Out[1200]:
```
730
```

In [120…
```
np.sum(test['FireplaceQu']=='NA')
```

Out[1201]:
```
730
```

In [120…
```
np.sum(test['FireplaceQu'].isnull())
```

Out[1202]:
```
0
```

In [120…
```
# Modifying'PoolQC'
test.loc[test['PoolArea']==0,'PoolQC']='NA'
```

In [120…
```
np.sum(test['PoolArea']==0)
```

Out[1204]:
```
1453
```

In [120…
```
np.sum(test['PoolQC']=='NA')
```
Out[1205]: 1453

In [120…
```
np.sum(test['PoolQC'].isnull())
```
Out[1206]: 3

In [120…
```
# test[['PoolArea','PoolQC']]
```
In [120…
```
test.loc[(test['PoolArea'] != 0)&(test['PoolQC']=='NA'),'PoolQC']=np.nan
```
In [120…
```
test.loc[test['PoolArea']!= 0]
```
Out[1209]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
514	1975	20	RL	106.0	12720	Pave	NaN	Reg	HLS
960	2421	20	RL	75.0	9532	Pave	NaN	Reg	Lv
1043	2504	50	RL	104.0	23920	Pave	NaN	Reg	Lv
1113	2574	20	RL	70.0	18044	Pave	NaN	IR1	HLS
1139	2600	20	RL	200.0	43500	Pave	NaN	Reg	Lv
1250	2711	80	RL	100.0	14330	Pave	NaN	IR1	Lov

6 rows × 82 columns

In [121…
```
# data.loc[1298, 'PoolQC']=np.nan
```
In [121…
```
np.sum(test['PoolQC'].isnull())
```
Out[1211]: 3

In [121…
```
test.loc[test['PoolQC'].isnull()]
```
Out[1212]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
960	2421	20	RL	75.0	9532	Pave	NaN	Reg	Lv
1043	2504	50	RL	104.0	23920	Pave	NaN	Reg	Lv
1139	2600	20	RL	200.0	43500	Pave	NaN	Reg	Lv

3 rows × 82 columns

In [121…
```
# Modifying'Fence'
np.sum(test['Fence'].isnull())
```
Out[1213]: 1169

In [121…
```
test.loc[test['Fence'].isnull(),'Fence']='NA'
```
In [121…
```
np.sum(test['Fence'].isnull())
```

Out[1215]: 0

In [121…
```
# test['Fence']
```

In [121…
```
# Modifying'MiscFeature'
np.sum(test['MiscFeature'].isnull())
```

Out[1217]: 1408

In [121…
```
test.loc[test['MiscFeature'].isnull(),'MiscFeature']='NA'
```

In [121…
```
np.sum(test['MiscFeature'].isnull())
```

Out[1219]: 0

In [122…
```
# test['MiscFeature']
```

In [122…
```
# Modifying 'GarageType','GarageCars','GarageYrBlt','GarageFinish','GarageQual'and'
np.sum(test['GarageArea']==0)
```

Out[1221]: 76

In [122…
```
test.loc[test['GarageArea']==0,['GarageType','GarageFinish','GarageQual','GarageCor
test.loc[test['GarageArea']==0,['GarageYrBlt','GarageCars']]=0
# test[['GarageArea','GarageType','GarageYrBlt','GarageFinish','GarageQual','Garage
```

In [122…
```
np.sum(test['GarageType'].isnull())
```

Out[1223]: 0

In [122…
```
np.sum(test['GarageYrBlt'].isnull())
```

Out[1224]: 2

In [122…
```
test.loc[test['GarageYrBlt'].isnull()]
```

Out[1225]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
666	2127	60	RM	57.0	8094	Pave	Grvl	Reg	Lv
1116	2577	70	RM	50.0	9060	Pave	NaN	Reg	Lv

2 rows × 82 columns

In [122…
```
np.sum(test['GarageFinish'].isnull())
```

Out[1226]: 2

In [122…
```
test.loc[test['GarageFinish'].isnull()]
```

Out[1227]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
666	2127	60	RM	57.0	8094	Pave	Grvl	Reg	Lv
1116	2577	70	RM	50.0	9060	Pave	NaN	Reg	Lv

2 rows × 82 columns

In [122…

```
np.sum(test['GarageQual'].isnull())
```

Out[1228]: 2

In [122…

```
test.loc[test['GarageQual'].isnull()]
```

Out[1229]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
666	2127	60	RM	57.0	8094	Pave	Grvl	Reg	Lv
1116	2577	70	RM	50.0	9060	Pave	NaN	Reg	Lv

2 rows × 82 columns

In [123…

```
np.sum(test['GarageCond'].isnull())
```

Out[1230]: 2

In [123…

```
test.loc[test['GarageCond'].isnull()]
```

Out[1231]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
666	2127	60	RM	57.0	8094	Pave	Grvl	Reg	Lv
1116	2577	70	RM	50.0	9060	Pave	NaN	Reg	Lv

2 rows × 82 columns

In [123…

```
np.sum(test['GarageCars'].isnull())
```

Out[1232]: 1

In [123…

```
test.loc[test['GarageCars'].isnull()]
```

Out[1233]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
1116	2577	70	RM	50.0	9060	Pave	NaN	Reg	Lv

1 rows × 82 columns

In [123…

```
# Check Missing Value in 'SaleType'
np.sum(test['SaleType'].isnull())
```

Out[1234]: 1

In [123…

```
test.loc[test['SaleType'].isnull()]
```

Out[1235]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
1029	2490	20	RL	85.0	13770	Pave	NaN	Reg	Lv

1 rows × 82 columns

In [123…

```
# Modifying 'LotFrontage'
np.sum(test['LotFrontage'].isnull())
```

Out[1236]:

```
227
```

In [123…

```
test.loc[test['LotFrontage'].isnull()]
```

Out[1237]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
6	1467	20	RL	NaN	7980	Pave	NaN	IR1	Lv
40	1501	160	FV	NaN	2980	Pave	NaN	Reg	Lv
41	1502	160	FV	NaN	2403	Pave	NaN	IR1	Lv
45	1506	20	RL	NaN	10456	Pave	NaN	IR1	Lv
47	1508	50	RL	NaN	18837	Pave	NaN	IR1	Lv
...	...	...	...	...	...	...	...	...	.
1387	2848	20	RL	NaN	11088	Pave	NaN	Reg	Lv
1390	2851	60	RL	NaN	21533	Pave	NaN	IR2	Lv
1440	2901	20	RL	NaN	50102	Pave	NaN	IR1	Lov
1441	2902	20	RL	NaN	8098	Pave	NaN	IR1	Lv
1448	2909	90	RL	NaN	11836	Pave	NaN	IR1	Lv

227 rows × 82 columns

In [123…

```
test.loc[test['LotFrontage'].isnull(),'LotFrontage']=0
# test['LotFrontage']
```

In [123…

```
np.sum(test['LotFrontage'].isnull())
```

Out[1239]:

```
0
```

In [124…

```
np.sum(test['LotFrontage']==0)
```

Out[1240]:

```
227
```

In [124…

```
# Modifying 'Alley'
np.sum(test['Alley'].isnull())
```

Out[1241]:

```
1352
```

In [124…
```
test.loc[test['Alley'].isnull(),'Alley']='NA'
# test['Alley']
```

In [124…
```
np.sum(test['Alley'].isnull())
```

Out[1243]: 0

In [124…
```
np.sum(test['Alley']=='NA')
```

Out[1244]: 1352

In [124…
```
# Modifying'BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1',
#'BsmtFinType2','BsmtFinSF2','BsmtUnfSF','TotalBsmtSF','BsmtFullBath','BsmtHalfBath
np.sum(test['TotalBsmtSF']==0)
```

Out[1245]: 41

In [124…
```
test.loc[test['TotalBsmtSF']==0,['BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1
# test[['TotalBsmtSF','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinT
```

In [124…
```
np.sum(test['BsmtCond'].isnull())
```

Out[1247]: 4

In [124…
```
np.sum(test['BsmtExposure'].isnull())
```

Out[1248]: 3

In [124…
```
test.loc[test['BsmtExposure'].isnull()]
```

Out[1249]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
27	1488	20	RL	73.0	8987	Pave	NA	Reg	Lvl
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl
888	2349	60	FV	81.0	10411	Pave	NA	Reg	Lvl

3 rows × 82 columns

In [125…
```
np.sum(test['BsmtQual'].isnull())
```

Out[1250]: 3

In [125…
```
np.sum(test['BsmtFinType1'].isnull())
```

Out[1251]: 1

In [125…
```
np.sum(test['BsmtFinType2'].isnull())
```

Out[1252]: 1

In [125…
```
test.loc[test['BsmtFinType2'].isnull()]
```

Out[1253]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [125…

```
np.sum(test['BsmtFinSF1'].isnull())
```

Out[1254]: 1

In [125…

```
test.loc[test['BsmtFinSF1'].isnull()]
```

Out[1255]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [125…

```
np.sum(test['BsmtFinSF2'].isnull())
```

Out[1256]: 1

In [125…

```
test.loc[test['BsmtFinSF2'].isnull()]
```

Out[1257]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [125…

```
np.sum(test['BsmtUnfSF'].isnull())
```

Out[1258]: 1

In [125…

```
test.loc[test['BsmtUnfSF'].isnull()]
```

Out[1259]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [126…

```
np.sum(test['TotalBsmtSF'].isnull())
```

Out[1260]: 1

In [126…

```
test.loc[test['TotalBsmtSF'].isnull()]
```

Out[1261]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [126…
```
np.sum(test['BsmtFullBath'].isnull())
```

Out[1262]:
```
1
```

In [126…
```
test.loc[test['BsmtHalfBath'].isnull()]
```

Out[1263]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [126…
```
np.sum(test['BsmtHalfBath'].isnull())
```

Out[1264]:
```
1
```

In [126…
```
test.loc[test['BsmtHalfBath'].isnull()]
```

Out[1265]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lvl

1 rows × 82 columns

In [126…
```
# Check Missing Value in 'Utilities'
test.loc[test['Utilities'].isnull()]
```

Out[1266]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
455	1916	30	NaN	109.0	21780	Grvl	NA	Reg	Lvl
485	1946	20	RL	0.0	31220	Pave	NA	IR1	Bnk

2 rows × 82 columns

In [126…
```
# Check Missing Value in 'MSZoning'
test.loc[test['MSZoning'].isnull()]
```

Out[1267]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
455	1916	30	NaN	109.0	21780	Grvl	NA	Reg	Lv
756	2217	20	NaN	80.0	14584	Pave	NA	Reg	Lov
790	2251	70	NaN	0.0	56600	Pave	NA	IR1	Lov
1444	2905	20	NaN	125.0	31250	Pave	NA	Reg	Lv

4 rows × 82 columns

In [126…

```
# Check Missing Value in 'Exterior1st'
test.loc[test['Exterior1st'].isnull()]
```

Out[1268]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
691	2152	30	RL	85.0	19550	Pave	NA	Reg	Lvl

1 rows × 82 columns

In [126…

```
# Check Missing Value in 'Exterior2nd'
test.loc[test['Exterior2nd'].isnull()]
```

Out[1269]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
691	2152	30	RL	85.0	19550	Pave	NA	Reg	Lvl

1 rows × 82 columns

In [127…

```
#Step 2: Determine the extent of MVs after Modifying
#Summary of MVs in each column after Modifying
mvs_summary = pd.DataFrame({'freq' : np.sum(test.isnull())})
mvs_summary['pct'] = round(mvs_summary['freq'] / test.shape[0] * 100, 1)
mvs_summary.sort_values(by = 'pct', ascending = False)
```

Out[1270]:

	freq	pct
MasVnrType	18	1.2
MasVnrArea	15	1.0
MSZoning	4	0.3
BsmtCond	4	0.3
PoolQC	3	0.2
...	...	...
Electrical	0	0.0
1stFlrSF	0	0.0
2ndFlrSF	0	0.0
LowQualFinSF	0	0.0
mvs_pct	0	0.0

82 rows × 2 columns

In [127…

```
#Summary of MVs for each case after Modifying
test.loc[:, 'mvs']=test.apply(lambda row: np.sum(row.isnull()), axis = 1)
test.sort_values(by='mvs', ascending = False)
test.loc[:,'mvs_pct']=round( test.apply(lambda row: np.sum(row.isnull())/(test.shap
#(data.shape[1]-3), 3 refers to 3 extra columns (id, mvs, mvs-pct)
test.sort_values(by='mvs', ascending = False)
```

Out[1271]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContou
660	2121	20	RM	99.0	5940	Pave	NA	IR1	Lv
1116	2577	70	RM	50.0	9060	Pave	NA	Reg	Lv
666	2127	60	RM	57.0	8094	Pave	Grvl	Reg	Lv
544	2005	20	RL	87.0	10037	Pave	NA	Reg	Lv
532	1993	60	RL	0.0	7750	Pave	NA	Reg	Lv
...	...	...	...	...	...	...	...	...	..
487	1948	20	RL	0.0	47280	Pave	NA	IR1	Lv
486	1947	120	RL	60.0	8118	Pave	NA	Reg	HLS
484	1945	20	RL	53.0	15401	Pave	NA	IR1	HLS
483	1944	60	RL	101.0	13543	Pave	NA	IR1	HLS
1458	2919	60	RL	74.0	9627	Pave	NA	Reg	Lv

1459 rows × 82 columns

In [127…

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 82 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             1459 non-null   int64  
 1   MSSubClass     1459 non-null   int64  
 2   MSZoning       1455 non-null   object 
 3   LotFrontage    1459 non-null   float64
 4   LotArea        1459 non-null   int64  
 5   Street         1459 non-null   object 
 6   Alley          1459 non-null   object 
 7   LotShape       1459 non-null   object 
 8   LandContour    1459 non-null   object 
 9   Utilities      1457 non-null   object 
 10  LotConfig      1459 non-null   object 
 11  LandSlope      1459 non-null   object 
 12  Neighborhood   1459 non-null   object 
 13  Condition1     1459 non-null   object 
 14  Condition2     1459 non-null   object 
 15  BldgType       1459 non-null   object 
 16  HouseStyle     1459 non-null   object 
 17  OverallQual    1459 non-null   int64  
 18  OverallCond    1459 non-null   int64  
 19  YearBuilt      1459 non-null   int64  
 20  YearRemodAdd   1459 non-null   int64  
 21  RoofStyle      1459 non-null   object 
 22  RoofMatl       1459 non-null   object 
 23  Exterior1st    1458 non-null   object 
 24  Exterior2nd    1458 non-null   object 
 25  MasVnrType     1441 non-null   object 
 26  MasVnrArea     1444 non-null   float64
 27  ExterQual      1459 non-null   object 
 28  ExterCond      1459 non-null   object 
 29  Foundation     1459 non-null   object 
 30  BsmtQual       1456 non-null   object 
 31  BsmtCond       1455 non-null   object 
 32  BsmtExposure   1456 non-null   object 
 33  BsmtFinType1   1458 non-null   object 
 34  BsmtFinSF1     1458 non-null   float64
 35  BsmtFinType2   1458 non-null   object 
 36  BsmtFinSF2     1458 non-null   object 
 37  BsmtUnfSF      1458 non-null   object 
 38  TotalBsmtSF    1458 non-null   object 
 39  Heating        1459 non-null   object 
 40  HeatingQC      1459 non-null   object 
 41  CentralAir     1459 non-null   object 
 42  Electrical     1459 non-null   object 
 43  1stFlrSF       1459 non-null   int64  
 44  2ndFlrSF       1459 non-null   int64  
 45  LowQualFinSF   1459 non-null   int64  
 46  GrLivArea      1459 non-null   int64  
 47  BsmtFullBath   1458 non-null   object 
 48  BsmtHalfBath   1458 non-null   object 
 49  FullBath       1459 non-null   int64  
 50  HalfBath       1459 non-null   int64  
 51  BedroomAbvGr   1459 non-null   int64  
 52  KitchenAbvGr   1459 non-null   int64  
 53  KitchenQual    1458 non-null   object 
 54  TotRmsAbvGrd   1459 non-null   int64  
 55  Functional     1457 non-null   object 
 56  Fireplaces     1459 non-null   int64  
 57  FireplaceQu    1459 non-null   object 
 58  GarageType     1459 non-null   object 
```

```
 59  GarageYrBlt    1457 non-null   float64
 60  GarageFinish   1457 non-null   object
 61  GarageCars     1458 non-null   float64
 62  GarageArea     1458 non-null   float64
 63  GarageQual     1457 non-null   object
 64  GarageCond     1457 non-null   object
 65  PavedDrive     1459 non-null   object
 66  WoodDeckSF     1459 non-null   int64
 67  OpenPorchSF    1459 non-null   int64
 68  EnclosedPorch  1459 non-null   int64
 69  3SsnPorch      1459 non-null   int64
 70  ScreenPorch    1459 non-null   int64
 71  PoolArea       1459 non-null   int64
 72  PoolQC         1456 non-null   object
 73  Fence          1459 non-null   object
 74  MiscFeature    1459 non-null   object
 75  MiscVal        1459 non-null   int64
 76  MoSold         1459 non-null   int64
 77  YrSold         1459 non-null   int64
 78  SaleType       1458 non-null   object
 79  SaleCondition  1459 non-null   object
 80  mvs            1459 non-null   int64
 81  mvs_pct        1459 non-null   float64
dtypes: float64(7), int64(27), object(48)
memory usage: 934.8+ KB
```

In [127…
```
test['mvs_pct'].max()
```

Out[1273]:
```
13.9
```

In [127…
```
#Step 3: Diagnose the randomness of the missing values processes
#Given the not too high percentage of missing values in columns and records (in a l
#it is acceptable to proceed with a simple imputation strategy without a detailed e
#This imputation will not significantly impact the overall results.
```

In [127…
```
# Step 4: Select the imputation method
np.sum(test.isnull())
```

Out[1275]:
```
Id               0
MSSubClass       0
MSZoning         4
LotFrontage      0
LotArea          0
                ..
YrSold           0
SaleType         1
SaleCondition    0
mvs              0
mvs_pct          0
Length: 82, dtype: int64
```

In [127…
```
print(test[['LotFrontage','MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'T
                     'BsmtFullBath', 'BsmtHalfBath', 'Fireplaces', 'GarageCars', 'G
```

```
LotFrontage     float64
MasVnrArea      float64
BsmtFinSF1      float64
BsmtFinSF2       object
BsmtUnfSF        object
TotalBsmtSF      object
BsmtFullBath     object
BsmtHalfBath     object
Fireplaces        int64
GarageCars      float64
GarageArea      float64
dtype: object
```

In [127…

```
# Create a dataframe to save mean of numeric variables for different imputation met
# Method : mean substitution
# Substiude missing values in numeric columns with the mean of each column.
float_column=['MasVnrArea', 'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF',
                     'BsmtFullBath', 'BsmtHalfBath','GarageCars', 'GarageArea','Gar

# Convert columns to numeric, forcing non-numeric values to NaN
for i in float_column:
    test[i]=pd.to_numeric(test[i], errors='coerce')

# Now, impute missing values with the mean of each column
for i in float_column:
    test[i]=test[i].fillna(test[i].mean())

np.sum(test.isnull())
```

Out[1277]:

```
Id                0
MSSubClass        0
MSZoning          4
LotFrontage       0
LotArea           0
                 ..
YrSold            0
SaleType          1
SaleCondition     0
mvs               0
mvs_pct           0
Length: 82, dtype: int64
```

In [127…

```
# Convert the float columns to int
test['LotFrontage'] = test['LotFrontage'].astype('int64')
test['MasVnrArea'] = test['MasVnrArea'].astype('int64')
test['BsmtFinSF1'] = test['BsmtFinSF1'].astype('int64')
test['BsmtFinSF2'] = test['BsmtFinSF2'].astype('int64')
test['BsmtUnfSF'] = test['BsmtUnfSF'].astype('int64')
test[ 'TotalBsmtSF'] = test['TotalBsmtSF'].astype('int64')
test['BsmtFullBath'] = test['BsmtFullBath'].astype('int64')
test['BsmtHalfBath'] = test['BsmtHalfBath'].astype('int64')
test['GarageCars'] = test['GarageCars'].astype('int64')
test['GarageArea'] = test['GarageArea'].astype('int64')
test['GarageYrBlt'] = test['GarageYrBlt'].astype('int64')
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 82 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1459 non-null   int64
 1   MSSubClass     1459 non-null   int64
 2   MSZoning       1455 non-null   object
 3   LotFrontage    1459 non-null   int64
 4   LotArea        1459 non-null   int64
 5   Street         1459 non-null   object
 6   Alley          1459 non-null   object
 7   LotShape       1459 non-null   object
 8   LandContour    1459 non-null   object
 9   Utilities      1457 non-null   object
 10  LotConfig      1459 non-null   object
 11  LandSlope      1459 non-null   object
 12  Neighborhood   1459 non-null   object
 13  Condition1     1459 non-null   object
 14  Condition2     1459 non-null   object
 15  BldgType       1459 non-null   object
 16  HouseStyle     1459 non-null   object
 17  OverallQual    1459 non-null   int64
 18  OverallCond    1459 non-null   int64
 19  YearBuilt      1459 non-null   int64
 20  YearRemodAdd   1459 non-null   int64
 21  RoofStyle      1459 non-null   object
 22  RoofMatl       1459 non-null   object
 23  Exterior1st    1458 non-null   object
 24  Exterior2nd    1458 non-null   object
 25  MasVnrType     1441 non-null   object
 26  MasVnrArea     1459 non-null   int64
 27  ExterQual      1459 non-null   object
 28  ExterCond      1459 non-null   object
 29  Foundation     1459 non-null   object
 30  BsmtQual       1456 non-null   object
 31  BsmtCond       1455 non-null   object
 32  BsmtExposure   1456 non-null   object
 33  BsmtFinType1   1458 non-null   object
 34  BsmtFinSF1     1459 non-null   int64
 35  BsmtFinType2   1458 non-null   object
 36  BsmtFinSF2     1459 non-null   int64
 37  BsmtUnfSF      1459 non-null   int64
 38  TotalBsmtSF    1459 non-null   int64
 39  Heating        1459 non-null   object
 40  HeatingQC      1459 non-null   object
 41  CentralAir     1459 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1459 non-null   int64
 44  2ndFlrSF       1459 non-null   int64
 45  LowQualFinSF   1459 non-null   int64
 46  GrLivArea      1459 non-null   int64
 47  BsmtFullBath   1459 non-null   int64
 48  BsmtHalfBath   1459 non-null   int64
 49  FullBath       1459 non-null   int64
 50  HalfBath       1459 non-null   int64
 51  BedroomAbvGr   1459 non-null   int64
 52  KitchenAbvGr   1459 non-null   int64
 53  KitchenQual    1458 non-null   object
 54  TotRmsAbvGrd   1459 non-null   int64
 55  Functional     1457 non-null   object
 56  Fireplaces     1459 non-null   int64
 57  FireplaceQu    1459 non-null   object
 58  GarageType     1459 non-null   object
```

```
 59  GarageYrBlt    1459 non-null   int64
 60  GarageFinish   1457 non-null   object
 61  GarageCars     1459 non-null   int64
 62  GarageArea     1459 non-null   int64
 63  GarageQual     1457 non-null   object
 64  GarageCond     1457 non-null   object
 65  PavedDrive     1459 non-null   object
 66  WoodDeckSF     1459 non-null   int64
 67  OpenPorchSF    1459 non-null   int64
 68  EnclosedPorch  1459 non-null   int64
 69  3SsnPorch      1459 non-null   int64
 70  ScreenPorch    1459 non-null   int64
 71  PoolArea       1459 non-null   int64
 72  PoolQC         1456 non-null   object
 73  Fence          1459 non-null   object
 74  MiscFeature    1459 non-null   object
 75  MiscVal        1459 non-null   int64
 76  MoSold         1459 non-null   int64
 77  YrSold         1459 non-null   int64
 78  SaleType       1458 non-null   object
 79  SaleCondition  1459 non-null   object
 80  mvs            1459 non-null   int64
 81  mvs_pct        1459 non-null   float64
dtypes: float64(1), int64(38), object(43)
memory usage: 934.8+ KB
```

In [127…

```
# Method : mode substitution
# Substiude missing values in categorical columns with the mode of each column.
categorical_columns=['MSZoning','Utilities','Exterior1st','Exterior2nd','MasVnrType
                       'BsmtExposure','BsmtFinType1','BsmtFinType2','GarageQual','F

for i in categorical_columns:
    mode_value=test[i].mode()[0]
    test[i]=test[i].fillna(mode_value)

print(test.shape)
np.sum(test.isnull())
```

```
(1459, 82)
```

Out[1279]:

```
Id               0
MSSubClass       0
MSZoning         0
LotFrontage      0
LotArea          0
                ..
YrSold           0
SaleType         0
SaleCondition    0
mvs              0
mvs_pct          0
Length: 82, dtype: int64
```

In [128…

```
# Remove added coloumns
test.drop(columns=['mvs', 'mvs_pct'], inplace=True)
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 80 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1459 non-null   int64
 1   MSSubClass     1459 non-null   int64
 2   MSZoning       1459 non-null   object
 3   LotFrontage    1459 non-null   int64
 4   LotArea        1459 non-null   int64
 5   Street         1459 non-null   object
 6   Alley          1459 non-null   object
 7   LotShape       1459 non-null   object
 8   LandContour    1459 non-null   object
 9   Utilities      1459 non-null   object
 10  LotConfig      1459 non-null   object
 11  LandSlope      1459 non-null   object
 12  Neighborhood   1459 non-null   object
 13  Condition1     1459 non-null   object
 14  Condition2     1459 non-null   object
 15  BldgType       1459 non-null   object
 16  HouseStyle     1459 non-null   object
 17  OverallQual    1459 non-null   int64
 18  OverallCond    1459 non-null   int64
 19  YearBuilt      1459 non-null   int64
 20  YearRemodAdd   1459 non-null   int64
 21  RoofStyle      1459 non-null   object
 22  RoofMatl       1459 non-null   object
 23  Exterior1st    1459 non-null   object
 24  Exterior2nd    1459 non-null   object
 25  MasVnrType     1459 non-null   object
 26  MasVnrArea     1459 non-null   int64
 27  ExterQual      1459 non-null   object
 28  ExterCond      1459 non-null   object
 29  Foundation     1459 non-null   object
 30  BsmtQual       1459 non-null   object
 31  BsmtCond       1459 non-null   object
 32  BsmtExposure   1459 non-null   object
 33  BsmtFinType1   1459 non-null   object
 34  BsmtFinSF1     1459 non-null   int64
 35  BsmtFinType2   1459 non-null   object
 36  BsmtFinSF2     1459 non-null   int64
 37  BsmtUnfSF      1459 non-null   int64
 38  TotalBsmtSF    1459 non-null   int64
 39  Heating        1459 non-null   object
 40  HeatingQC      1459 non-null   object
 41  CentralAir     1459 non-null   object
 42  Electrical     1459 non-null   object
 43  1stFlrSF       1459 non-null   int64
 44  2ndFlrSF       1459 non-null   int64
 45  LowQualFinSF   1459 non-null   int64
 46  GrLivArea      1459 non-null   int64
 47  BsmtFullBath   1459 non-null   int64
 48  BsmtHalfBath   1459 non-null   int64
 49  FullBath       1459 non-null   int64
 50  HalfBath       1459 non-null   int64
 51  BedroomAbvGr   1459 non-null   int64
 52  KitchenAbvGr   1459 non-null   int64
 53  KitchenQual    1459 non-null   object
 54  TotRmsAbvGrd   1459 non-null   int64
 55  Functional     1459 non-null   object
 56  Fireplaces     1459 non-null   int64
 57  FireplaceQu    1459 non-null   object
 58  GarageType     1459 non-null   object
```

```
 59  GarageYrBlt    1459 non-null   int64
 60  GarageFinish   1459 non-null   object
 61  GarageCars     1459 non-null   int64
 62  GarageArea     1459 non-null   int64
 63  GarageQual     1459 non-null   object
 64  GarageCond     1459 non-null   object
 65  PavedDrive     1459 non-null   object
 66  WoodDeckSF     1459 non-null   int64
 67  OpenPorchSF    1459 non-null   int64
 68  EnclosedPorch  1459 non-null   int64
 69  3SsnPorch      1459 non-null   int64
 70  ScreenPorch    1459 non-null   int64
 71  PoolArea       1459 non-null   int64
 72  PoolQC         1459 non-null   object
 73  Fence          1459 non-null   object
 74  MiscFeature    1459 non-null   object
 75  MiscVal        1459 non-null   int64
 76  MoSold         1459 non-null   int64
 77  YrSold         1459 non-null   int64
 78  SaleType       1459 non-null   object
 79  SaleCondition  1459 non-null   object
dtypes: int64(37), object(43)
memory usage: 912.0+ KB
```

Convert ordinal variables in to numeric (Label Encoding)

In [128…

```
# Convert 'LotShape' into numeric variable
test['cnvrt_LotShape']=test['LotShape'].replace(['Reg','IR1','IR2','IR3'],
                                                 list(range(4, 0,-1)), inplace = Fa
# Check the results
print(np.sum(test['cnvrt_LotShape'].isnull()))
test['cnvrt_LotShape'].describe()
# test['cnvrt_LotShape']
```

```
0
```

Out[1281]:

```
count    1459.000000
mean        3.607951
std         0.557864
min         1.000000
25%         3.000000
50%         4.000000
75%         4.000000
max         4.000000
Name: cnvrt_LotShape, dtype: float64
```

In [128…

```
# Convert 'LandSlope' into numeric variable
test['cnvrt_LandSlope']=test['LandSlope'].replace(['Gtl','Mod','Sev'],
                                                 list(range(3, 0,-1)), inplace = Fa
# Check the results
print(np.sum(test['cnvrt_LandSlope'].isnull()))
test['cnvrt_LandSlope'].describe()
# test['cnvrt_LandSlope']
```

```
0
```

Out[1282]:

```
count    1459.000000
mean        2.954764
std         0.217566
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         3.000000
Name: cnvrt_LandSlope, dtype: float64
```

In [128…

```
# Convert 'ExterQua' into numeric variable
test['cnvrt_ExterQual']=test['ExterQual'].replace(['Ex','Gd','TA','Fa','Po'],
                                                   list(range(5, 0,-1)), inplace = Fa
# Check the results
print(np.sum(test['cnvrt_ExterQual'].isnull()))
test['cnvrt_ExterQual'].describe()
# test['cnvrt_ExterQual']
```

Out[1283]:

```
0
count    1459.000000
mean        3.397533
std         0.586444
min         2.000000
25%         3.000000
50%         3.000000
75%         4.000000
max         5.000000
Name: cnvrt_ExterQual, dtype: float64
```

In [128…

```
# Convert 'ExterCond' into numeric variable
test['cnvrt_ExterCond']=test['ExterCond'].replace(['Ex','Gd','TA','Fa','Po'],
                                                   list(range(5, 0,-1)), inplace = Fa
#Check the results
print(np.sum(test['cnvrt_ExterCond'].isnull()))
test['cnvrt_ExterCond'].describe()
# test['cnvrt_ExterCond']
```

Out[1284]:

```
0
count    1459.000000
mean        3.087731
std         0.392637
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         5.000000
Name: cnvrt_ExterCond, dtype: float64
```

In [128…

```
# Convert 'BsmtQual' into numeric variable
test['cnvrt_BsmtQual']=test['BsmtQual'].replace(['Ex','Gd','TA','Fa','Po','NA'],
                                                 list(range(6, 0,-1)), inplace = Fa
#Check the results
print(np.sum(test['cnvrt_BsmtQual'].isnull()))
test['cnvrt_BsmtQual'].describe()
# test['cnvrt_BsmtQual']
```

Out[1285]:

```
0
count    1459.000000
mean        4.472241
std         0.920587
min         1.000000
25%         4.000000
50%         4.000000
75%         5.000000
max         6.000000
Name: cnvrt_BsmtQual, dtype: float64
```

In [128…

```
# Convert 'BsmtCond' into numeric variable
test['cnvrt_BsmtCond']=test['BsmtCond'].replace(['Ex','Gd','TA','Fa','Po','NA'],
                                                 list(range(6,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_BsmtCond'].isnull()))
test['cnvrt_BsmtCond'].describe()
# test['cnvrt_BsmtCond']
```

```
0
```

Out[1286]:

```
count    1459.000000
mean        3.910212
std         0.576897
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         5.000000
Name: cnvrt_BsmtCond, dtype: float64
```

In [128…

```
# Convert 'BsmtExposure' into numeric variable
test['cnvrt_BsmtExposure']=test['BsmtExposure'].replace(['Gd','Av','Mn','No','NA'],
                                                         list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_BsmtExposure'].isnull()))
test['cnvrt_BsmtExposure'].describe()
# test['cnvrt_BsmtExposure']
```

```
0
```

Out[1287]:

```
count    1459.000000
mean        2.619602
std         1.070838
min         1.000000
25%         2.000000
50%         2.000000
75%         3.000000
max         5.000000
Name: cnvrt_BsmtExposure, dtype: float64
```

In [128…

```
# Convert 'BsmtFinType1' into numeric variable
test['cnvrt_BsmtFinType1']=test['BsmtFinType1'].replace(['GLQ','ALQ','BLQ','Rec','L
                                                         list(range(7,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_BsmtFinType1'].isnull()))
test['cnvrt_BsmtFinType1'].describe()
# test['cnvrt_BsmtFinType1']
```

```
0
```

Out[1288]:

```
count    1459.000000
mean        4.541467
std         2.119580
min         1.000000
25%         2.000000
50%         5.000000
75%         7.000000
max         7.000000
Name: cnvrt_BsmtFinType1, dtype: float64
```

In [128…

```
# Convert 'BsmtFinType2' into numeric variable
test['cnvrt_BsmtFinType2']=test['BsmtFinType2'].replace(['GLQ','ALQ','BLQ','Rec','L
                                                         list(range(7,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_BsmtFinType2'].isnull()))
test['cnvrt_BsmtFinType2'].describe()
# test['cnvrt_BsmtFinType2']
```

```
0
```

Out[1289]:
```
count    1459.000000
mean        2.300891
std         1.013255
min         1.000000
25%         2.000000
50%         2.000000
75%         2.000000
max         7.000000
Name: cnvrt_BsmtFinType2, dtype: float64
```

In [129…
```
# Convert 'HeatingQC' into numeric variable
test['cnvrt_HeatingQC']=test['HeatingQC'].replace(['Ex','Gd','TA','Fa','Po'],
                                                   list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_HeatingQC'].isnull()))
test['cnvrt_HeatingQC'].describe()
# test['cnvrt_HeatingQC']
```

```
0
```

Out[1290]:
```
count    1459.000000
mean        4.158328
std         0.956684
min         1.000000
25%         3.000000
50%         5.000000
75%         5.000000
max         5.000000
Name: cnvrt_HeatingQC, dtype: float64
```

In [129…
```
# Convert 'KitchenQual' into numeric variable
test['cnvrt_KitchenQual']=test['KitchenQual'].replace(['Ex','Gd','TA','Fa','Po'],
                                                   list(range(5,0,-1)), inplace = Fal
#Check the results
print(np.sum(test['cnvrt_KitchenQual'].isnull()))
test['cnvrt_KitchenQual'].describe()
# data['cnvrt_KitchenQual']
```

```
0
```

Out[1291]:
```
count    1459.000000
mean        3.509938
std         0.660780
min         2.000000
25%         3.000000
50%         3.000000
75%         4.000000
max         5.000000
Name: cnvrt_KitchenQual, dtype: float64
```

In [129…
```
# Convert 'Functional' into numeric variable
test['cnvrt_Functional']=test['Functional'].replace(['Typ','Min1','Min2','Mod','Ma
                                                   list(range(8,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_Functional'].isnull()))
test['cnvrt_Functional'].describe()
# test['cnvrt_Functional']
```

```
0
```

Out[1292]:
```
count    1459.000000
mean        7.854695
std         0.610379
min         2.000000
25%         8.000000
50%         8.000000
75%         8.000000
max         8.000000
Name: cnvrt_Functional, dtype: float64
```

In [129…
```
# Convert 'FireplaceQu' into numeric variable
test['cnvrt_FireplaceQu']=test['FireplaceQu'].replace(['Ex','Gd','TA','Fa','Po','NA
                                                 list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_FireplaceQu'].isnull()))
test['cnvrt_FireplaceQu'].describe()
# test['cnvrt_FireplaceQu']
```

```
0
```

Out[1293]:
```
count    1459.000000
mean        2.710761
std         1.801147
min         1.000000
25%         1.000000
50%         1.000000
75%         5.000000
max         6.000000
Name: cnvrt_FireplaceQu, dtype: float64
```

In [129…
```
# Convert 'GarageFinish' into numeric variable
test['cnvrt_GarageFinish'] = test['GarageFinish'].replace(['Fin','RFn','Unf','NA'],
                                                 list(range(4,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_GarageFinish'].isnull()))
test['cnvrt_GarageFinish'].describe()
# test['cnvrt_GarageFinish']
```

```
0
```

Out[1294]:
```
count    1459.000000
mean        2.717615
std         0.900258
min         1.000000
25%         2.000000
50%         3.000000
75%         4.000000
max         4.000000
Name: cnvrt_GarageFinish, dtype: float64
```

In [129…
```
# Convert 'GararageQual' into numeric variable
test['cnvrt_GarageQual']=test['GarageQual'].replace(['Ex','Gd','TA','Fa','Po','NA']
                                                 list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_GarageQual'].isnull()))
test['cnvrt_GarageQual'].describe()
# test['cnvrt_GarageQual']
```

```
0
```

Out[1295]:
```
count    1459.000000
mean        3.795751
std         0.701328
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         5.000000
Name: cnvrt_GarageQual, dtype: float64
```

In [129…
```
# Convert 'GararageCond' into numeric variable
test['cnvrt_GarageCond']=test['GarageCond'].replace(['Ex','Gd','TA','Fa','Po','NA']
                                                    list(range(6,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_GarageCond'].isnull()))
test['cnvrt_GarageCond'].describe()
# test['cnvrt_GarageCond']
```

```
0
```

Out[1296]:
```
count    1459.000000
mean        3.812886
std         0.697791
min         1.000000
25%         4.000000
50%         4.000000
75%         4.000000
max         6.000000
Name: cnvrt_GarageCond, dtype: float64
```

In [129…
```
# Convert 'PavedDrive' into numeric variable
test['cnvrt_PavedDrive']=test['PavedDrive'].replace(['Y','P','N'],
                                                    list(range(3,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_PavedDrive'].isnull()))
test['cnvrt_PavedDrive'].describe()
# test['cnvrt_PavedDrive']
```

```
0
```

Out[1297]:
```
count    1459.000000
mean        2.805346
std         0.574204
min         1.000000
25%         3.000000
50%         3.000000
75%         3.000000
max         3.000000
Name: cnvrt_PavedDrive, dtype: float64
```

In [129…
```
# Convert 'PoolQC' into numeric variable
test['cnvrt_PoolQC']=test['PoolQC'].replace(['Ex','Gd','TA','Fa','NA'],
                                            list(range(5,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_PoolQC'].isnull()))
test['cnvrt_PoolQC'].describe()
# test['cnvrt_PoolQC']
```

```
0
```

Out[1298]:
```
count    1459.000000
mean        1.007539
std         0.167523
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         5.000000
Name: cnvrt_PoolQC, dtype: float64
```

In [129…
```
# Convert 'Fence' into numeric variable
test['cnvrt_Fence']=test['Fence'].replace(['GdPrv','MnPrv','GdWo','MnWw','NA'],
                                                  list(range(5,0,-1)), inplace = Fal

#Check the results
print(np.sum(test['cnvrt_Fence'].isnull()))
test['cnvrt_Fence'].describe()
# test['cnvrt_Fence']
```

```
0
```

Out[1299]:
```
count    1459.000000
mean        1.595613
std         1.230447
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         5.000000
Name: cnvrt_Fence, dtype: float64
```

In [130…
```
np.sum(test.isnull())
```

Out[1300]:
```
Id                   0
MSSubClass           0
MSZoning             0
LotFrontage          0
LotArea              0
                    ..
cnvrt_GarageQual     0
cnvrt_GarageCond     0
cnvrt_PavedDrive     0
cnvrt_PoolQC         0
cnvrt_Fence          0
Length: 99, dtype: int64
```

In [130…
```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 99 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Id                1459 non-null   int64
 1   MSSubClass        1459 non-null   int64
 2   MSZoning          1459 non-null   object
 3   LotFrontage       1459 non-null   int64
 4   LotArea           1459 non-null   int64
 5   Street            1459 non-null   object
 6   Alley             1459 non-null   object
 7   LotShape          1459 non-null   object
 8   LandContour       1459 non-null   object
 9   Utilities         1459 non-null   object
 10  LotConfig         1459 non-null   object
 11  LandSlope         1459 non-null   object
 12  Neighborhood      1459 non-null   object
 13  Condition1        1459 non-null   object
 14  Condition2        1459 non-null   object
 15  BldgType          1459 non-null   object
 16  HouseStyle        1459 non-null   object
 17  OverallQual       1459 non-null   int64
 18  OverallCond       1459 non-null   int64
 19  YearBuilt         1459 non-null   int64
 20  YearRemodAdd      1459 non-null   int64
 21  RoofStyle         1459 non-null   object
 22  RoofMatl          1459 non-null   object
 23  Exterior1st       1459 non-null   object
 24  Exterior2nd       1459 non-null   object
 25  MasVnrType        1459 non-null   object
 26  MasVnrArea        1459 non-null   int64
 27  ExterQual         1459 non-null   object
 28  ExterCond         1459 non-null   object
 29  Foundation        1459 non-null   object
 30  BsmtQual          1459 non-null   object
 31  BsmtCond          1459 non-null   object
 32  BsmtExposure      1459 non-null   object
 33  BsmtFinType1      1459 non-null   object
 34  BsmtFinSF1        1459 non-null   int64
 35  BsmtFinType2      1459 non-null   object
 36  BsmtFinSF2        1459 non-null   int64
 37  BsmtUnfSF         1459 non-null   int64
 38  TotalBsmtSF       1459 non-null   int64
 39  Heating           1459 non-null   object
 40  HeatingQC         1459 non-null   object
 41  CentralAir        1459 non-null   object
 42  Electrical        1459 non-null   object
 43  1stFlrSF          1459 non-null   int64
 44  2ndFlrSF          1459 non-null   int64
 45  LowQualFinSF      1459 non-null   int64
 46  GrLivArea         1459 non-null   int64
 47  BsmtFullBath      1459 non-null   int64
 48  BsmtHalfBath      1459 non-null   int64
 49  FullBath          1459 non-null   int64
 50  HalfBath          1459 non-null   int64
 51  BedroomAbvGr      1459 non-null   int64
 52  KitchenAbvGr      1459 non-null   int64
 53  KitchenQual       1459 non-null   object
 54  TotRmsAbvGrd      1459 non-null   int64
 55  Functional        1459 non-null   object
 56  Fireplaces        1459 non-null   int64
 57  FireplaceQu       1459 non-null   object
 58  GarageType        1459 non-null   object
```

```
 59  GarageYrBlt        1459 non-null   int64
 60  GarageFinish       1459 non-null   object
 61  GarageCars         1459 non-null   int64
 62  GarageArea         1459 non-null   int64
 63  GarageQual         1459 non-null   object
 64  GarageCond         1459 non-null   object
 65  PavedDrive         1459 non-null   object
 66  WoodDeckSF         1459 non-null   int64
 67  OpenPorchSF        1459 non-null   int64
 68  EnclosedPorch      1459 non-null   int64
 69  3SsnPorch          1459 non-null   int64
 70  ScreenPorch        1459 non-null   int64
 71  PoolArea           1459 non-null   int64
 72  PoolQC             1459 non-null   object
 73  Fence              1459 non-null   object
 74  MiscFeature        1459 non-null   object
 75  MiscVal            1459 non-null   int64
 76  MoSold             1459 non-null   int64
 77  YrSold             1459 non-null   int64
 78  SaleType           1459 non-null   object
 79  SaleCondition      1459 non-null   object
 80  cnvrt_LotShape     1459 non-null   int64
 81  cnvrt_LandSlope    1459 non-null   int64
 82  cnvrt_ExterQual    1459 non-null   int64
 83  cnvrt_ExterCond    1459 non-null   int64
 84  cnvrt_BsmtQual     1459 non-null   int64
 85  cnvrt_BsmtCond     1459 non-null   int64
 86  cnvrt_BsmtExposure 1459 non-null   int64
 87  cnvrt_BsmtFinType1 1459 non-null   int64
 88  cnvrt_BsmtFinType2 1459 non-null   int64
 89  cnvrt_HeatingQC    1459 non-null   int64
 90  cnvrt_KitchenQual  1459 non-null   int64
 91  cnvrt_Functional   1459 non-null   int64
 92  cnvrt_FireplaceQu  1459 non-null   int64
 93  cnvrt_GarageFinish 1459 non-null   int64
 94  cnvrt_GarageQual   1459 non-null   int64
 95  cnvrt_GarageCond   1459 non-null   int64
 96  cnvrt_PavedDrive   1459 non-null   int64
 97  cnvrt_PoolQC       1459 non-null   int64
 98  cnvrt_Fence        1459 non-null   int64
dtypes: int64(56), object(43)
memory usage: 1.1+ MB
```

In [130…

```
# Convert 'MSSubClass' to string type
test['MSSubClass']=test['MSSubClass'].astype(str)


#Create dummy variables for categorical variables
dummy_vars_te=pd.get_dummies(test[['MSSubClass','MSZoning','Street','Alley','LandCo
                                   'Utilities','LotConfig','Neighborhood','Conditio
                                   'BldgType','HouseStyle','RoofStyle','Heating','F
                                   'Exterior1st','Exterior2nd','MasVnrType','Founda
                                   'Electrical','GarageType','MiscFeature','SaleTyp

# dummy_vars containing only 0 and 1
dummy_vars_te=dummy_vars_te.astype(int)

print(dummy_vars_te.head(2))
dummy_vars_te.info()
```

```
   MSSubClass_120  MSSubClass_150  MSSubClass_160  MSSubClass_180  \
0               0               0               0               0
1               0               0               0               0

   MSSubClass_190  MSSubClass_20  MSSubClass_30  MSSubClass_40  MSSubClass_45  \
0               0              1              0              0              0
1               0              1              0              0              0

   MSSubClass_50  ...  SaleType_ConLw  SaleType_New  SaleType_Oth  \
0              0  ...               0             0             0
1              0  ...               0             0             0

   SaleType_WD  SaleCondition_Abnorml  SaleCondition_AdjLand  \
0            1                      0                      0
1            1                      0                      0

   SaleCondition_Alloca  SaleCondition_Family  SaleCondition_Normal  \
0                     0                     0                     1
1                     0                     0                     1

   SaleCondition_Partial
0                      0
1                      0

[2 rows x 171 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Columns: 171 entries, MSSubClass_120 to SaleCondition_Partial
dtypes: int32(171)
memory usage: 974.7 KB
```

In [130…

```
##Read data from file
# data=pd.read_csv('train.csv')
# data.head(5)
```

In [130…

```
## Train data
# train=data
# train.info()
# print(train.head(4))
```

In [130…

```
# ##Split Train data into test and train for validation (named trainv & testv)
# from sklearn.model_selection import train_test_split
# trainv, testv = train_test_split(data, test_size = 0.3, random_state = 1234)
print(trainv.shape)
# print(testv.shape)
```

```
(1022, 100)
```

In [130…

```
# #Create dummy variables for categorical variables
# dummy_vars_trv=pd.get_dummies(trainv[['MSSubClass','MSZoning','Street','Alley','L
#                                      'Utilities','LotConfig','Neighborhood','Condit
#                                      'BldgType','HouseStyle','RoofStyle','Heating',
#                                      'Exterior1st','Exterior2nd','MasVnrType','Four
#                                      'Electrical','GarageType','MiscFeature','SaleT

# # dummy_vars containing only 0 and 1
# dummy_vars_trv=dummy_vars_trv.astype(int)

# print(dummy_vars_trv.head(2))
# dummy_vars_trv.info()
```

In [130…

```
dummy_vars_trv.shape
```

Out[1307]: (1022, 157)

In [130…
```
dummy_vars_te.shape
```

Out[1308]: (1459, 171)

In [130…
```
test.shape
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 99 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Id                  1459 non-null   int64
 1   MSSubClass          1459 non-null   object
 2   MSZoning            1459 non-null   object
 3   LotFrontage         1459 non-null   int64
 4   LotArea             1459 non-null   int64
 5   Street              1459 non-null   object
 6   Alley               1459 non-null   object
 7   LotShape            1459 non-null   object
 8   LandContour         1459 non-null   object
 9   Utilities           1459 non-null   object
 10  LotConfig           1459 non-null   object
 11  LandSlope           1459 non-null   object
 12  Neighborhood        1459 non-null   object
 13  Condition1          1459 non-null   object
 14  Condition2          1459 non-null   object
 15  BldgType            1459 non-null   object
 16  HouseStyle          1459 non-null   object
 17  OverallQual         1459 non-null   int64
 18  OverallCond         1459 non-null   int64
 19  YearBuilt           1459 non-null   int64
 20  YearRemodAdd        1459 non-null   int64
 21  RoofStyle           1459 non-null   object
 22  RoofMatl            1459 non-null   object
 23  Exterior1st         1459 non-null   object
 24  Exterior2nd         1459 non-null   object
 25  MasVnrType          1459 non-null   object
 26  MasVnrArea          1459 non-null   int64
 27  ExterQual           1459 non-null   object
 28  ExterCond           1459 non-null   object
 29  Foundation          1459 non-null   object
 30  BsmtQual            1459 non-null   object
 31  BsmtCond            1459 non-null   object
 32  BsmtExposure        1459 non-null   object
 33  BsmtFinType1        1459 non-null   object
 34  BsmtFinSF1          1459 non-null   int64
 35  BsmtFinType2        1459 non-null   object
 36  BsmtFinSF2          1459 non-null   int64
 37  BsmtUnfSF           1459 non-null   int64
 38  TotalBsmtSF         1459 non-null   int64
 39  Heating             1459 non-null   object
 40  HeatingQC           1459 non-null   object
 41  CentralAir          1459 non-null   object
 42  Electrical          1459 non-null   object
 43  1stFlrSF            1459 non-null   int64
 44  2ndFlrSF            1459 non-null   int64
 45  LowQualFinSF        1459 non-null   int64
 46  GrLivArea           1459 non-null   int64
 47  BsmtFullBath        1459 non-null   int64
 48  BsmtHalfBath        1459 non-null   int64
 49  FullBath            1459 non-null   int64
 50  HalfBath            1459 non-null   int64
 51  BedroomAbvGr        1459 non-null   int64
 52  KitchenAbvGr        1459 non-null   int64
 53  KitchenQual         1459 non-null   object
 54  TotRmsAbvGrd        1459 non-null   int64
 55  Functional          1459 non-null   object
 56  Fireplaces          1459 non-null   int64
 57  FireplaceQu         1459 non-null   object
 58  GarageType          1459 non-null   object
```

```
 59  GarageYrBlt          1459 non-null   int64
 60  GarageFinish         1459 non-null   object
 61  GarageCars           1459 non-null   int64
 62  GarageArea           1459 non-null   int64
 63  GarageQual           1459 non-null   object
 64  GarageCond           1459 non-null   object
 65  PavedDrive           1459 non-null   object
 66  WoodDeckSF           1459 non-null   int64
 67  OpenPorchSF          1459 non-null   int64
 68  EnclosedPorch        1459 non-null   int64
 69  3SsnPorch            1459 non-null   int64
 70  ScreenPorch          1459 non-null   int64
 71  PoolArea             1459 non-null   int64
 72  PoolQC               1459 non-null   object
 73  Fence                1459 non-null   object
 74  MiscFeature          1459 non-null   object
 75  MiscVal              1459 non-null   int64
 76  MoSold               1459 non-null   int64
 77  YrSold               1459 non-null   int64
 78  SaleType             1459 non-null   object
 79  SaleCondition        1459 non-null   object
 80  cnvrt_LotShape       1459 non-null   int64
 81  cnvrt_LandSlope      1459 non-null   int64
 82  cnvrt_ExterQual      1459 non-null   int64
 83  cnvrt_ExterCond      1459 non-null   int64
 84  cnvrt_BsmtQual       1459 non-null   int64
 85  cnvrt_BsmtCond       1459 non-null   int64
 86  cnvrt_BsmtExposure   1459 non-null   int64
 87  cnvrt_BsmtFinType1   1459 non-null   int64
 88  cnvrt_BsmtFinType2   1459 non-null   int64
 89  cnvrt_HeatingQC      1459 non-null   int64
 90  cnvrt_KitchenQual    1459 non-null   int64
 91  cnvrt_Functional     1459 non-null   int64
 92  cnvrt_FireplaceQu    1459 non-null   int64
 93  cnvrt_GarageFinish   1459 non-null   int64
 94  cnvrt_GarageQual     1459 non-null   int64
 95  cnvrt_GarageCond     1459 non-null   int64
 96  cnvrt_PavedDrive     1459 non-null   int64
 97  cnvrt_PoolQC         1459 non-null   int64
 98  cnvrt_Fence          1459 non-null   int64
dtypes: int64(55), object(44)
memory usage: 1.1+ MB
```

In [131…

```
missing_in_train=set(dummy_vars_te.columns)-set(dummy_vars_trv.columns)
print(missing_in_train)
```

```
{'Utilities_AllPub', 'LotConfig_Inside', 'HouseStyle_1Story', 'RoofStyle_Gable',
'Alley_NA', 'Exterior1st_AsphShn', 'Foundation_PConc', 'Exterior2nd_VinylSd', 'Ele
ctrical_SBrkr', 'MSSubClass_150', 'Exterior1st_VinylSd', 'MSSubClass_20', 'MSZonin
g_RL', 'LandContour_Lvl', 'Neighborhood_NAmes', 'Street_Pave', 'Condition2_Norm',
'MasVnrType_none', 'SaleType_WD', 'MiscFeature_NA', 'BldgType_1Fam', 'GarageType_A
ttchd', 'Condition1_Norm', 'CentralAir_Y', 'Heating_GasA', 'RoofMatl_CompShg', 'Sa
leCondition_Normal'}
```

In [131…

```
dummy_vars_te=dummy_vars_te.reindex(columns=dummy_vars_trv.columns, fill_value=0)
```

In [131…

```
print(dummy_vars_trv.shape)
print(trainv.shape)


print(dummy_vars_te.shape)
print(test.shape)
```

```
(1022, 157)
(1022, 100)
(1459, 157)
(1459, 99)
```

In [131…

```
X_trainv.shape
```

Out[1313]:

```
(1022, 212)
```

In [131…

```
X_trainv
```

Out[1314]:

	const	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnr
1017	1	0	5814	8	5	1984	1984	
405	1	0	9991	4	4	1976	1993	
6	1	75	10084	8	5	2004	2005	
388	1	93	9382	7	5	1999	2000	
501	1	75	9803	7	5	2005	2005	
...	...	...	...	...	...	...	...	
1228	1	65	8769	9	5	2008	2008	
1077	1	0	15870	5	5	1969	1969	
1318	1	0	14781	8	5	2001	2002	
723	1	60	8172	4	6	1954	1972	
815	1	48	12137	7	5	1998	1998	

1022 rows × 212 columns

In [131…

```
test.shape
```

Out[1315]:

```
(1459, 99)
```

In [131…

```
#Define feature matrix
# Train All columns except 'SalePrice'
X_X=test.iloc[:,list(range(0,99))]
print(X_X.columns)
X_test=pd.concat([X_X,dummy_vars_te], axis = 1)
# X_trainv.info()
# #add constant
X_test=sm.add_constant(X_test)
X_test.head(2)
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
       'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
       'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
       'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
       'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
       'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
       'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
       'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
       'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
       'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
       'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
       'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
       'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
       'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
       'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'cnvrt_LotShape', 'cnvrt_LandSlope', 'cnvrt_ExterQual',
       'cnvrt_ExterCond', 'cnvrt_BsmtQual', 'cnvrt_BsmtCond',
       'cnvrt_BsmtExposure', 'cnvrt_BsmtFinType1', 'cnvrt_BsmtFinType2',
       'cnvrt_HeatingQC', 'cnvrt_KitchenQual', 'cnvrt_Functional',
       'cnvrt_FireplaceQu', 'cnvrt_GarageFinish', 'cnvrt_GarageQual',
       'cnvrt_GarageCond', 'cnvrt_PavedDrive', 'cnvrt_PoolQC', 'cnvrt_Fence'],
      dtype='object')
```

Out[1316]:

	const	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandCon
0	1.0	1461	20	RH	80	11622	Pave	NA	Reg	
1	1.0	1462	20	RL	81	14267	Pave	NA	IR1	

2 rows × 257 columns

In [131…

```
missing_in_train=set(dummy_vars_te.columns)-set(dummy_vars_trv.columns)
print(missing_in_train)
```

```
set()
```

In [131…

```
dummy_vars_te=dummy_vars_te.reindex(columns=dummy_vars_trv.columns, fill_value=0)
```

In [131…

```
# Remove coloumn'Id'
X_test.drop(columns=['Id'],inplace= True)
X_test.head()
```

Out[1319]:

	const	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour
0	1.0	20	RH	80	11622	Pave	NA	Reg	Lvl
1	1.0	20	RL	81	14267	Pave	NA	IR1	Lvl
2	1.0	60	RL	74	13830	Pave	NA	IR1	Lvl
3	1.0	60	RL	78	9978	Pave	NA	IR1	Lvl
4	1.0	120	RL	43	5005	Pave	NA	IR1	HLS

5 rows × 256 columns

In [132…

```
X_test=X_test.drop(columns=X_test.select_dtypes(include=['object']).columns)
# Check the data types of the cleaned DataFrame
print(X_test.dtypes)
```

```
const                        float64
LotFrontage                    int64
LotArea                        int64
OverallQual                    int64
OverallCond                    int64
                              ...
SaleCondition_Abnorml          int32
SaleCondition_AdjLand          int32
SaleCondition_Alloca           int32
SaleCondition_Family           int32
SaleCondition_Partial          int32
Length: 212, dtype: object
```

In [132…
```
X_trainv.shape
```

Out[1321]:
```
(1022, 212)
```

In [132…
```
X_test.shape
```

Out[1322]:
```
(1459, 212)
```

In [132…
```
# Convert float and int32 columns to int64
X_test=X_test.astype({col: 'int64' for col in X_test.select_dtypes(include=['float6

# Check the data types of the columns to verify the conversion
print(X_test.dtypes)
```

```
const                        int64
LotFrontage                  int64
LotArea                      int64
OverallQual                  int64
OverallCond                  int64
                             ...
SaleCondition_Abnorml        int64
SaleCondition_AdjLand        int64
SaleCondition_Alloca         int64
SaleCondition_Family         int64
SaleCondition_Partial        int64
Length: 212, dtype: object
```

Prediction of Test data on model 1- Linear Regression with Box-Cox Transformation and t_test

In [132…
```
# Ensure X_test only contains the significant columns
X_test_significant = X_test.reindex(columns=X_trainv_current.columns, fill_value=0)
# Predict using the final model
test_pred=model_final.predict(X_test_significant)

#Inverse transformation of predicted values
test['pred_lm']=pd.Series(boxcox.inverse_transform(test_pred.values.reshape(-1, 1))
                          index = test_pred.index)
test['pred_lm']
```

```
Out[1324]: 0       125907.002542
1       156994.092116
2       177170.343039
3       204098.780397
4       191286.525163
            ...
1454     82455.780862
1455     85137.010710
1456    173237.675760
1457    122852.962219
1458    223150.583344
Name: pred_lm, Length: 1459, dtype: float64
```

Prediction of Test data on Model 2- Forward Selection with Box-Cox Transformation

```
In [132…
#Predict on test- model 2
pred_fwd=fwd_models.loc[105, 'model'].predict(X_test[fwd_models.loc[105,'model'].mo
test['pred_fwd']=pd.Series(boxcox.inverse_transform(pred_fwd.values.reshape(-1, 1))
test['pred_fwd']
```

```
Out[1325]: 0       128098.015053
1       159394.570197
2       176230.517246
3       199737.267378
4       193131.062723
            ...
1454     85222.465036
1455     87928.559130
1456    164809.006336
1457    113780.846393
1458    211569.423539
Name: pred_fwd, Length: 1459, dtype: float64
```

Prediction of Test data on Model 3- Backward Elimination with Box-Cox Transformation

```
In [132…
#Predict on test- model 3
pred_bwd=bwd_models.loc[120,'model'].predict(X_test[bwd_models.loc[120,'model'].mod
test['pred_bwd']=pd.Series(boxcox.inverse_transform(pred_bwd.values.reshape(-1, 1))
test['pred_bwd']
```

```
Out[1326]: 0       128126.933033
1       158761.514493
2       178896.964660
3       203938.303461
4       195473.080202
            ...
1454     85664.611232
1455     87667.625749
1456    172751.968004
1457    117571.634622
1458    217195.653496
Name: pred_bwd, Length: 1459, dtype: float64
```

Prediction of Test data on Model 4- Ridge Regression with Box-Cox Transformation

```
In [132…
# Sacle test data set
X_test_scaled=scaler.transform(X_test)
pred_ridge=model_4.predict(X_test_scaled)
```

```
test['pred_ridge']=pd.Series(boxcox.inverse_transform(pred_ridge).reshape(-1),index
test['pred_ridge']
```

Out[1327]:
```
0       125156.159690
1       157355.521049
2       177114.480000
3       198803.437005
4       188809.429645
            ...
1454     87031.431834
1455     89466.758219
1456    164923.243579
1457    110478.969582
1458    211756.842354
Name: pred_ridge, Length: 1459, dtype: float64
```

Prediction of Test data on Model 5- Lasso Regression with Box-Cox Transformation

In [132…
```
#Predict on test- model 5
pred_lasso=model_5.predict(X_test_scaled)
test['pred_lasso']=pd.Series(boxcox.inverse_transform(pred_lasso.reshape(-1, 1)).re
test['pred_lasso']
```

Out[1328]:
```
0       126874.110951
1       153531.570360
2       175155.284747
3       196837.999800
4       192854.270200
            ...
1454     85483.160537
1455     86548.135649
1456    165461.506352
1457    116090.022513
1458    220529.754715
Name: pred_lasso, Length: 1459, dtype: float64
```

Prediction of Test data on Model 6- Decision Tree with Box-Cox Transformation

In [132…
```
#Prediction using model 6
pred_tree = model_6.predict(X_test)
test['pred_tree'] = pd.Series(boxcox.inverse_transform(pred_tree.reshape(-1, 1)).re
                      index = test.index)
test['pred_tree']
```

Out[1329]:
```
0       110511.666492
1       146671.518695
2       204918.670222
3       170783.032869
4       238455.141923
            ...
1454     82544.031066
1455     98981.547558
1456    146671.518695
1457    110511.666492
1458    195524.030116
Name: pred_tree, Length: 1459, dtype: float64
```

Prediction of Test data on Model 7- Random Forest with Box-Cox Transformation

In [133…

```
# common_columns = X_train.columns.intersection(X_test.columns)
# X_train = X_train[common_columns]
# X_test= X_test[common_columns]
```

In [133…

```
#Prediction using model 7
pred_rf=model_7.predict(X_test)
test['pred_rf']=pd.Series(boxcox.inverse_transform(pred_rf.reshape(-1, 1)).reshape(
                    index=test.index)
test['pred_rf']
```

Out[1331]:

```
0       125366.919935
1       149510.071184
2       183874.125950
3       187568.644255
4       188000.772033
            ...
1454     92018.382799
1455    100728.303890
1456    162411.293609
1457    121925.094123
1458    221216.178736
Name: pred_rf, Length: 1459, dtype: float64
```

Prediction of Test data on Model 8- PCR with Box-Cox Transformation

In [133…

```
#Predict on test - model 8
#Scale data
X_test_scaled=scaler.transform(X_test)

#Transfrom data into new dimensions
X_test_pca=pca_model.transform(X_test_scaled)

#Prediction on test
pred_pcr=model_8.predict(X_test_pca[:, : components_number])
test['pred_pcr']=pd.Series(boxcox.inverse_transform(pred_pcr.reshape(-1, 1)).reshap
                    index = test.index)
test['pred_pcr']
```

Out[1332]:

```
0       116643.969056
1       145816.919659
2       183742.806387
3       207067.614224
4       181154.240425
            ...
1454     84515.828518
1455     87631.157356
1456    166143.413497
1457    122765.099699
1458    224655.217156
Name: pred_pcr, Length: 1459, dtype: float64
```