ex2

February 1, 2024

[1]:
```
import numpy as np
import pandas as pd
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

[2]:
```
data = pd.read_csv("house_pred.csv")
test = pd.read_csv("test.csv")
```

[3]:
```
test.drop("Id",axis = 1,inplace = True)
data.drop("Id",axis = 1,inplace=True)
```

[4]:
```
data
```

[4]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	\
0	60	RL	65.0	8450	Pave	NaN	Reg	
1	20	RL	80.0	9600	Pave	NaN	Reg	
2	60	RL	68.0	11250	Pave	NaN	IR1	
3	70	RL	60.0	9550	Pave	NaN	IR1	
4	60	RL	84.0	14260	Pave	NaN	IR1	
…	…	…	…	…	…	…	…	
1455	60	RL	62.0	7917	Pave	NaN	Reg	
1456	20	RL	85.0	13175	Pave	NaN	Reg	
1457	70	RL	66.0	9042	Pave	NaN	Reg	
1458	20	RL	68.0	9717	Pave	NaN	Reg	
1459	20	RL	75.0	9937	Pave	NaN	Reg	

	LandContour	Utilities	LotConfig	…	PoolArea	PoolQC	Fence	MiscFeature	\
0	Lvl	AllPub	Inside	…	0	NaN	NaN	NaN	
1	Lvl	AllPub	FR2	…	0	NaN	NaN	NaN	
2	Lvl	AllPub	Inside	…	0	NaN	NaN	NaN	
3	Lvl	AllPub	Corner	…	0	NaN	NaN	NaN	
4	Lvl	AllPub	FR2	…	0	NaN	NaN	NaN	
…	…	…	…	…	…	…	…	…	
1455	Lvl	AllPub	Inside	…	0	NaN	NaN	NaN	
1456	Lvl	AllPub	Inside	…	0	NaN	MnPrv	NaN	
1457	Lvl	AllPub	Inside	…	0	NaN	GdPrv	Shed	
1458	Lvl	AllPub	Inside	…	0	NaN	NaN	NaN	
1459	Lvl	AllPub	Inside	…	0	NaN	NaN	NaN	

```
     MiscVal MoSold  YrSold  SaleType  SaleCondition  SalePrice
0          0      2    2008        WD         Normal     208500
1          0      5    2007        WD         Normal     181500
2          0      9    2008        WD         Normal     223500
3          0      2    2006        WD        Abnorml     140000
4          0     12    2008        WD         Normal     250000
…        …    …      …         …             …          …
1455       0      8    2007        WD         Normal     175000
1456       0      2    2010        WD         Normal     210000
1457    2500      5    2010        WD         Normal     266500
1458       0      4    2010        WD         Normal     142125
1459       0      6    2008        WD         Normal     147500

[1460 rows x 80 columns]
```

[5]:
```
x = data.iloc[:,:-1].values
y = data.iloc[:,-1].values
```

[6]:
```
datasets = [data, test]

cleaned_datasets = []

for dataset in datasets:
    null_threshold = len(dataset) * 0.75
    cleaned_dataset = dataset.dropna(axis=1, thresh=null_threshold)
    cleaned_datasets.append(cleaned_dataset)

data = cleaned_datasets[0]
test = cleaned_datasets[1]
```

[7]:
```
datasets = {'data': data, 'test': test}

for dataset_name, dataset in datasets.items():

    for column in dataset.columns:
        null_count = dataset[column].isnull().sum()
        print(f"Column '{column}' in {dataset_name}: {null_count} null values")

    columns_with_null_values = dataset.columns[dataset.isnull().any()].tolist()
    print(f"Columns with at least one null value in {dataset_name}:␣
 ↪{columns_with_null_values}")

    count = len(columns_with_null_values)
    print(f"Number of columns with at least one null value in {dataset_name}:␣
 ↪{count}")
```

```
    print("\n")
```

```
Column 'MSSubClass' in data: 0 null values
Column 'MSZoning' in data: 0 null values
Column 'LotFrontage' in data: 259 null values
Column 'LotArea' in data: 0 null values
Column 'Street' in data: 0 null values
Column 'LotShape' in data: 0 null values
Column 'LandContour' in data: 0 null values
Column 'Utilities' in data: 0 null values
Column 'LotConfig' in data: 0 null values
Column 'LandSlope' in data: 0 null values
Column 'Neighborhood' in data: 0 null values
Column 'Condition1' in data: 0 null values
Column 'Condition2' in data: 0 null values
Column 'BldgType' in data: 0 null values
Column 'HouseStyle' in data: 0 null values
Column 'OverallQual' in data: 0 null values
Column 'OverallCond' in data: 0 null values
Column 'YearBuilt' in data: 0 null values
Column 'YearRemodAdd' in data: 0 null values
Column 'RoofStyle' in data: 0 null values
Column 'RoofMatl' in data: 0 null values
Column 'Exterior1st' in data: 0 null values
Column 'Exterior2nd' in data: 0 null values
Column 'MasVnrArea' in data: 8 null values
Column 'ExterQual' in data: 0 null values
Column 'ExterCond' in data: 0 null values
Column 'Foundation' in data: 0 null values
Column 'BsmtQual' in data: 37 null values
Column 'BsmtCond' in data: 37 null values
Column 'BsmtExposure' in data: 38 null values
Column 'BsmtFinType1' in data: 37 null values
Column 'BsmtFinSF1' in data: 0 null values
Column 'BsmtFinType2' in data: 38 null values
Column 'BsmtFinSF2' in data: 0 null values
Column 'BsmtUnfSF' in data: 0 null values
Column 'TotalBsmtSF' in data: 0 null values
Column 'Heating' in data: 0 null values
Column 'HeatingQC' in data: 0 null values
Column 'CentralAir' in data: 0 null values
Column 'Electrical' in data: 1 null values
Column '1stFlrSF' in data: 0 null values
Column '2ndFlrSF' in data: 0 null values
Column 'LowQualFinSF' in data: 0 null values
Column 'GrLivArea' in data: 0 null values
Column 'BsmtFullBath' in data: 0 null values
```

```
Column 'BsmtHalfBath' in data: 0 null values
Column 'FullBath' in data: 0 null values
Column 'HalfBath' in data: 0 null values
Column 'BedroomAbvGr' in data: 0 null values
Column 'KitchenAbvGr' in data: 0 null values
Column 'KitchenQual' in data: 0 null values
Column 'TotRmsAbvGrd' in data: 0 null values
Column 'Functional' in data: 0 null values
Column 'Fireplaces' in data: 0 null values
Column 'GarageType' in data: 81 null values
Column 'GarageYrBlt' in data: 81 null values
Column 'GarageFinish' in data: 81 null values
Column 'GarageCars' in data: 0 null values
Column 'GarageArea' in data: 0 null values
Column 'GarageQual' in data: 81 null values
Column 'GarageCond' in data: 81 null values
Column 'PavedDrive' in data: 0 null values
Column 'WoodDeckSF' in data: 0 null values
Column 'OpenPorchSF' in data: 0 null values
Column 'EnclosedPorch' in data: 0 null values
Column '3SsnPorch' in data: 0 null values
Column 'ScreenPorch' in data: 0 null values
Column 'PoolArea' in data: 0 null values
Column 'MiscVal' in data: 0 null values
Column 'MoSold' in data: 0 null values
Column 'YrSold' in data: 0 null values
Column 'SaleType' in data: 0 null values
Column 'SaleCondition' in data: 0 null values
Column 'SalePrice' in data: 0 null values
Columns with at least one null value in data: ['LotFrontage', 'MasVnrArea',
'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
'Electrical', 'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageQual',
'GarageCond']
Number of columns with at least one null value in data: 13


Column 'MSSubClass' in test: 0 null values
Column 'MSZoning' in test: 0 null values
Column 'LotFrontage' in test: 49 null values
Column 'LotArea' in test: 0 null values
Column 'Street' in test: 0 null values
Column 'LotShape' in test: 0 null values
Column 'LandContour' in test: 0 null values
Column 'Utilities' in test: 0 null values
Column 'LotConfig' in test: 0 null values
Column 'LandSlope' in test: 0 null values
Column 'Neighborhood' in test: 0 null values
Column 'Condition1' in test: 0 null values
```

```
Column 'Condition2' in test: 0 null values
Column 'BldgType' in test: 0 null values
Column 'HouseStyle' in test: 0 null values
Column 'OverallQual' in test: 0 null values
Column 'OverallCond' in test: 0 null values
Column 'YearBuilt' in test: 0 null values
Column 'YearRemodAdd' in test: 0 null values
Column 'RoofStyle' in test: 0 null values
Column 'RoofMatl' in test: 0 null values
Column 'Exterior1st' in test: 0 null values
Column 'Exterior2nd' in test: 0 null values
Column 'MasVnrArea' in test: 2 null values
Column 'ExterQual' in test: 0 null values
Column 'ExterCond' in test: 0 null values
Column 'Foundation' in test: 0 null values
Column 'BsmtQual' in test: 5 null values
Column 'BsmtCond' in test: 5 null values
Column 'BsmtExposure' in test: 5 null values
Column 'BsmtFinType1' in test: 5 null values
Column 'BsmtFinSF1' in test: 0 null values
Column 'BsmtFinType2' in test: 5 null values
Column 'BsmtFinSF2' in test: 0 null values
Column 'BsmtUnfSF' in test: 0 null values
Column 'TotalBsmtSF' in test: 0 null values
Column 'Heating' in test: 0 null values
Column 'HeatingQC' in test: 0 null values
Column 'CentralAir' in test: 0 null values
Column 'Electrical' in test: 1 null values
Column '1stFlrSF' in test: 0 null values
Column '2ndFlrSF' in test: 0 null values
Column 'LowQualFinSF' in test: 0 null values
Column 'GrLivArea' in test: 0 null values
Column 'BsmtFullBath' in test: 0 null values
Column 'BsmtHalfBath' in test: 0 null values
Column 'FullBath' in test: 0 null values
Column 'HalfBath' in test: 0 null values
Column 'BedroomAbvGr' in test: 0 null values
Column 'KitchenAbvGr' in test: 0 null values
Column 'KitchenQual' in test: 0 null values
Column 'TotRmsAbvGrd' in test: 0 null values
Column 'Functional' in test: 0 null values
Column 'Fireplaces' in test: 0 null values
Column 'GarageType' in test: 14 null values
Column 'GarageYrBlt' in test: 14 null values
Column 'GarageFinish' in test: 14 null values
Column 'GarageCars' in test: 0 null values
Column 'GarageArea' in test: 0 null values
Column 'GarageQual' in test: 14 null values
```

```
Column 'GarageCond' in test: 14 null values
Column 'PavedDrive' in test: 0 null values
Column 'WoodDeckSF' in test: 0 null values
Column 'OpenPorchSF' in test: 0 null values
Column 'EnclosedPorch' in test: 0 null values
Column '3SsnPorch' in test: 0 null values
Column 'ScreenPorch' in test: 0 null values
Column 'PoolArea' in test: 0 null values
Column 'MiscVal' in test: 0 null values
Column 'MoSold' in test: 0 null values
Column 'YrSold' in test: 0 null values
Column 'SaleType' in test: 0 null values
Column 'SaleCondition' in test: 0 null values
Columns with at least one null value in test: ['LotFrontage', 'MasVnrArea',
'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2',
'Electrical', 'GarageType', 'GarageYrBlt', 'GarageFinish', 'GarageQual',
'GarageCond']
Number of columns with at least one null value in test: 13
```

[8]:
```
datasets = {'data': data, 'test': test}

for dataset_name, dataset in datasets.items():
    print(f"Processing dataset: {dataset_name}")

    for column in dataset.columns:
        mode_value = dataset[column].mode()[0]
        dataset.loc[:, column] = dataset.loc[:, column].fillna(mode_value).
 ↪copy()

    print(f"{dataset_name} has been processed.")

    print("\n")
```

```
Processing dataset: data
data has been processed.


Processing dataset: test
test has been processed.
```

[9]:
```
datasets = [data,test]

for dataset in datasets:
```

```
    for column in dataset.columns:
        null_count = dataset[column].isnull().sum()

    columns_with_null_values = dataset.columns[dataset.isnull().any()].tolist()
    print(f"Columns with at least one null value in dataset:␣
 ↪{columns_with_null_values}")

    count = len(columns_with_null_values)
    print(f"Number of columns with at least one null value in dataset: {count}")

    print("\n")
```

```
Columns with at least one null value in dataset: []
Number of columns with at least one null value in dataset: 0


Columns with at least one null value in dataset: []
Number of columns with at least one null value in dataset: 0
```

[10]:
```
datasets = {'data': data, 'test': test}

for dataset_name, dataset in datasets.items():
    print(f"Processing dataset: {dataset_name}")

    categorical_columns = dataset.select_dtypes(include=['object']).columns
    encoded_dataset = pd.get_dummies(dataset, columns=categorical_columns,␣
 ↪dtype=int)

    dataset.loc[:, categorical_columns] = None
    dataset.dropna(axis=1, how='all', inplace=True)
    datasets[dataset_name] = encoded_dataset

    print(f"{dataset_name} has been processed.")

    print("\n")
```

```
Processing dataset: data
data has been processed.


Processing dataset: test
test has been processed.
```

```
C:\Users\rsath\AppData\Local\Temp\ipykernel_20168\189478648.py:10:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  dataset.dropna(axis=1, how='all', inplace=True)
```

[11]:
```
data
```

[11]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	\
0	60	65.0	8450	7	5	2003	
1	20	80.0	9600	6	8	1976	
2	60	68.0	11250	7	5	2001	
3	70	60.0	9550	7	5	1915	
4	60	84.0	14260	8	5	2000	
…	…	…	…	…	…	…	
1455	60	62.0	7917	6	5	1999	
1456	20	85.0	13175	6	6	1978	
1457	70	66.0	9042	7	9	1941	
1458	20	68.0	9717	5	6	1950	
1459	20	75.0	9937	5	6	1965	

	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	…	WoodDeckSF	\
0	2003	196.0	706	0	…	0	
1	1976	0.0	978	0	…	298	
2	2002	162.0	486	0	…	0	
3	1970	0.0	216	0	…	0	
4	2000	350.0	655	0	…	192	
…	…	…	…	…	…	…	
1455	2000	0.0	0	0	…	0	
1456	1988	119.0	790	163	…	349	
1457	2006	0.0	275	0	…	0	
1458	1996	0.0	49	1029	…	366	
1459	1965	0.0	830	290	…	736	

	OpenPorchSF	EnclosedPorch	3SsnPorch	ScreenPorch	PoolArea	MiscVal	\
0	61	0	0	0	0	0	
1	0	0	0	0	0	0	
2	42	0	0	0	0	0	
3	35	272	0	0	0	0	
4	84	0	0	0	0	0	
…	…	…	…	…	…	…	
1455	40	0	0	0	0	0	
1456	0	0	0	0	0	0	
1457	60	0	0	0	0	2500	
1458	0	112	0	0	0	0	

```
1459           68              0          0             0         0        0

      MoSold  YrSold  SalePrice
0          2    2008     208500
1          5    2007     181500
2          9    2008     223500
3          2    2006     140000
4         12    2008     250000
…       …      …         …
1455       8    2007     175000
1456       2    2010     210000
1457       5    2010     266500
1458       4    2010     142125
1459       6    2008     147500

[1460 rows x 37 columns]
```

[12]:
```
test
```

[12]:
```
      MSSubClass  LotFrontage  LotArea  OverallQual  OverallCond  YearBuilt  \
0             20         78.0    10140            5            6       1975
1             20         90.0    14684            7            7       1990
2             20         60.0     8900            4            4       1966
3             20         70.0     9135            6            5       2003
4             20         70.0     7763            5            7       1962
..           …          …       …           …            …         …
251           60         62.0     7917            6            5       1999
252           20         85.0    13175            6            6       1978
253           70         66.0     9042            7            9       1941
254           20         68.0     9717            5            6       1950
255           20         75.0     9937            5            6       1965

      YearRemodAdd  MasVnrArea  BsmtFinSF1  BsmtFinSF2  …  GarageArea  \
0             1975         0.0         788           0  …         495
1             1991       234.0         485         177  …         701
2             1966         0.0        1056           0  …         384
3             2003       120.0         340           0  …         544
4             1980         0.0         504         108  …         506
..             …         …          …         …  …           …
251           2000         0.0           0           0  …         460
252           1988       119.0         790         163  …         500
253           2006         0.0         275           0  …         252
254           1996         0.0          49        1029  …         240
255           1965         0.0         830         290  …         276

      WoodDeckSF  OpenPorchSF  EnclosedPorch  3SsnPorch  ScreenPorch  PoolArea  \
0              0           88              0          0            0         0
```

```
1            84           70              0          0            0         0
2             0           42              0          0            0         0
3           192           23              0          0            0         0
4             0            0              0          0            0         0
..          ...          ...            ...        ...          ...       ...
251           0           40              0          0            0         0
252         349            0              0          0            0         0
253           0           60              0          0            0         0
254         366            0            112          0            0         0
255         736           68              0          0            0         0

     MiscVal  MoSold  YrSold
0          0       7    2006
1          0       6    2009
2          0      11    2006
3          0       5    2006
4          0      10    2008
..       ...     ...     ...
251        0       8    2007
252        0       2    2010
253     2500       5    2010
254        0       4    2010
255        0       6    2008

[256 rows x 36 columns]
```

[13]:
```python
import statsmodels.api as sm

dataset = data.copy()

vif_threshold = 3
vif_data = pd.DataFrame()
vif_data["Variable"] = dataset.columns

for column in dataset.columns:
    print(f"Calculating VIF for variable: {column}")

    predictors = dataset.drop(column, axis=1)

    predictors = sm.add_constant(predictors)

    try:
        model = sm.OLS(dataset[column], predictors).fit()
        rsquared = model.rsquared
        vif_value = 1 / (1 - rsquared) if (1 - rsquared) != 0 else vif_threshold
        vif_data.loc[vif_data["Variable"] == column, "VIF"] = vif_value
    except (ValueError, TypeError) as e:
```

```
        print(f"Error calculating VIF for variable {column}: {e}")
        vif_data.loc[vif_data["Variable"] == column, "VIF"] = vif_threshold
    except Exception as e:
        print(f"Unexpected error for variable {column}: {e}")


high_vif_columns = vif_data[vif_data['VIF'] > vif_threshold]['Variable'].
 ↪tolist()
print(f"Columns with VIF > {vif_threshold}: {high_vif_columns}")

for column in high_vif_columns:
    if column != 'SalePrice':
        dataset.drop(column, axis=1, inplace=True)
print(vif_data)
```

```
Calculating VIF for variable: MSSubClass
Calculating VIF for variable: LotFrontage
Calculating VIF for variable: LotArea
Calculating VIF for variable: OverallQual
Calculating VIF for variable: OverallCond
Calculating VIF for variable: YearBuilt
Calculating VIF for variable: YearRemodAdd
Calculating VIF for variable: MasVnrArea
Calculating VIF for variable: BsmtFinSF1
Calculating VIF for variable: BsmtFinSF2
Calculating VIF for variable: BsmtUnfSF
Calculating VIF for variable: TotalBsmtSF
Calculating VIF for variable: 1stFlrSF
Calculating VIF for variable: 2ndFlrSF
Calculating VIF for variable: LowQualFinSF
Calculating VIF for variable: GrLivArea
Calculating VIF for variable: BsmtFullBath
Calculating VIF for variable: BsmtHalfBath
Calculating VIF for variable: FullBath
Calculating VIF for variable: HalfBath
Calculating VIF for variable: BedroomAbvGr
Calculating VIF for variable: KitchenAbvGr
Calculating VIF for variable: TotRmsAbvGrd
Calculating VIF for variable: Fireplaces
Calculating VIF for variable: GarageYrBlt
Calculating VIF for variable: GarageCars
Calculating VIF for variable: GarageArea
Calculating VIF for variable: WoodDeckSF
Calculating VIF for variable: OpenPorchSF
Calculating VIF for variable: EnclosedPorch
Calculating VIF for variable: 3SsnPorch
Calculating VIF for variable: ScreenPorch
```

```
Calculating VIF for variable: PoolArea
Calculating VIF for variable: MiscVal
Calculating VIF for variable: MoSold
Calculating VIF for variable: YrSold
Calculating VIF for variable: SalePrice
Columns with VIF > 3: ['OverallQual', 'YearBuilt', 'TotRmsAbvGrd', 'GarageCars',
'GarageArea', 'SalePrice']
         Variable       VIF
0      MSSubClass  1.697311
1     LotFrontage  1.524854
2         LotArea  1.261927
3     OverallQual  3.747943
4     OverallCond  1.626945
5       YearBuilt  4.746104
6    YearRemodAdd  2.421054
7      MasVnrArea  1.416806
8      BsmtFinSF1  3.000000
9      BsmtFinSF2  3.000000
10      BsmtUnfSF  3.000000
11    TotalBsmtSF  3.000000
12       1stFlrSF  3.000000
13       2ndFlrSF  3.000000
14   LowQualFinSF  3.000000
15      GrLivArea  3.000000
16   BsmtFullBath  2.239407
17   BsmtHalfBath  1.152561
18       FullBath  2.956182
19       HalfBath  2.168561
20   BedroomAbvGr  2.382367
21   KitchenAbvGr  1.602560
22   TotRmsAbvGrd  4.942704
23     Fireplaces  1.589917
24    GarageYrBlt  2.534342
25     GarageCars  5.780940
26     GarageArea  5.339466
27     WoodDeckSF  1.225478
28    OpenPorchSF  1.223869
29  EnclosedPorch  1.283961
30      3SsnPorch  1.023384
31    ScreenPorch  1.118632
32       PoolArea  1.107230
33        MiscVal  1.024115
34         MoSold  1.050010
35         YrSold  1.052504
36      SalePrice  5.372581
```

[14]:
```
import statsmodels.api as sm

testset = test.copy()

vif_threshold = 3
vif_data = pd.DataFrame()
vif_data["Variable"] = testset.columns

for column in testset.columns:
    print(f"Calculating VIF for variable: {column}")

    predictors = testset.drop(column, axis=1)

    predictors = sm.add_constant(predictors)

    try:
        model = sm.OLS(testset[column], predictors).fit()
        rsquared = model.rsquared
        vif_value = 1 / (1 - rsquared) if (1 - rsquared) != 0 else vif_threshold
        vif_data.loc[vif_data["Variable"] == column, "VIF"] = vif_value
    except (ValueError, TypeError) as e:
        print(f"Error calculating VIF for variable {column}: {e}")
        vif_data.loc[vif_data["Variable"] == column, "VIF"] = vif_threshold
    except Exception as e:
        print(f"Unexpected error for variable {column}: {e}")


high_vif_columns = vif_data[vif_data['VIF'] > vif_threshold]['Variable'].
 ↪tolist()
print(f"Columns with VIF > {vif_threshold}: {high_vif_columns}")

for column in high_vif_columns:
    if column != 'SalePrice':
        testset.drop(column, axis=1, inplace=True)
print(vif_data)
```

```
Calculating VIF for variable: MSSubClass
Calculating VIF for variable: LotFrontage
Calculating VIF for variable: LotArea
Calculating VIF for variable: OverallQual
Calculating VIF for variable: OverallCond
Calculating VIF for variable: YearBuilt
Calculating VIF for variable: YearRemodAdd
Calculating VIF for variable: MasVnrArea
Calculating VIF for variable: BsmtFinSF1
Calculating VIF for variable: BsmtFinSF2
Calculating VIF for variable: BsmtUnfSF
```

```
Calculating VIF for variable: TotalBsmtSF
Calculating VIF for variable: 1stFlrSF
Calculating VIF for variable: 2ndFlrSF
Calculating VIF for variable: LowQualFinSF
Calculating VIF for variable: GrLivArea
Calculating VIF for variable: BsmtFullBath
Calculating VIF for variable: BsmtHalfBath
Calculating VIF for variable: FullBath
Calculating VIF for variable: HalfBath
Calculating VIF for variable: BedroomAbvGr
Calculating VIF for variable: KitchenAbvGr
Calculating VIF for variable: TotRmsAbvGrd
Calculating VIF for variable: Fireplaces
Calculating VIF for variable: GarageYrBlt
Calculating VIF for variable: GarageCars
Calculating VIF for variable: GarageArea
Calculating VIF for variable: WoodDeckSF
Calculating VIF for variable: OpenPorchSF
Calculating VIF for variable: EnclosedPorch
Calculating VIF for variable: 3SsnPorch
Calculating VIF for variable: ScreenPorch
Calculating VIF for variable: PoolArea
Calculating VIF for variable: MiscVal
Calculating VIF for variable: MoSold
Calculating VIF for variable: YrSold
Columns with VIF > 3: ['OverallQual', 'YearBuilt', 'TotRmsAbvGrd', 'GarageCars',
'GarageArea']
         Variable       VIF
0      MSSubClass  2.001839
1     LotFrontage  2.082745
2         LotArea  2.041054
3     OverallQual  3.542950
4     OverallCond  1.607287
5       YearBuilt  6.052856
6    YearRemodAdd  2.699262
7      MasVnrArea  1.578182
8      BsmtFinSF1  3.000000
9      BsmtFinSF2  3.000000
10      BsmtUnfSF  3.000000
11    TotalBsmtSF  3.000000
12       1stFlrSF  3.000000
13       2ndFlrSF  3.000000
14   LowQualFinSF  3.000000
15      GrLivArea  3.000000
16   BsmtFullBath  2.398345
17   BsmtHalfBath  1.157182
18       FullBath  2.995927
19       HalfBath  2.242014
```

```
20   BedroomAbvGr  2.834874
21   KitchenAbvGr  2.007450
22   TotRmsAbvGrd  5.271804
23     Fireplaces  1.864382
24    GarageYrBlt  2.637613
25     GarageCars  6.430574
26     GarageArea  6.841393
27     WoodDeckSF  1.285059
28    OpenPorchSF  1.420569
29  EnclosedPorch  1.526747
30      3SsnPorch  1.104809
31    ScreenPorch  1.317851
32       PoolArea  1.468288
33        MiscVal  1.334529
34         MoSold  1.145333
35         YrSold  1.214009
```

[15]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

[16]:
```
model = LinearRegression()
```

[17]:
```
def select_common_columns_from_b(a, b):
    common_columns = set(a.columns) & set(b.columns)

    result_dataset = b[list(common_columns)].copy()

    return result_dataset


final_data = select_common_columns_from_b(testset,dataset)
final_test = select_common_columns_from_b(dataset,testset)
x = final_data
y = y

model.fit(x,y)
```

[17]:
```
LinearRegression()
```

[18]:
```
y_test_pred = model.predict(final_test)
```

[19]:
```
y_test_pred.reshape(len(y_test_pred),1)
```

[19]:
```
array([[141932.51442639],
       [286658.39975241],
       [133089.27427961],
       [245161.01063897],
```

```
[150657.32719763],
[276666.17045317],
[222340.605833  ],
[188501.35184238],
[ 74606.07242712],
[147583.09844961],
[105645.79836657],
[ 88081.81440927],
[106732.96581104],
[207157.89946644],
[ 77649.82162329],
[120108.93635577],
[ 99027.37990588],
[119058.86592576],
[188587.78205966],
[208780.50725138],
[204351.8805116 ],
[106273.28978016],
[218772.57951378],
[173440.67975892],
[299085.53961512],
[132062.35404064],
[219147.6960519 ],
[128883.33393039],
[ 72630.34354574],
[134763.76646457],
[159010.80304801],
[117701.10186716],
[180360.07744388],
[214320.09833139],
[155388.18858061],
[229362.65890153],
[225232.88694387],
[226120.47780151],
[164453.29973378],
[313851.08116771],
[129577.80581995],
[241005.79256198],
[183022.37867912],
[130556.75243151],
[156197.7524974 ],
[ 88538.76757585],
[268587.10576677],
[206618.92444854],
[106767.37072922],
[261913.58972401],
[204358.21158427],
```

```
[135586.80709239],
[315270.63371574],
[107446.46994183],
[190349.41809057],
[143193.43414191],
[197861.22591189],
[112042.42737941],
[173604.743286  ],
[138898.71990703],
[191079.95961718],
[175680.4945668 ],
[ 90670.17940053],
[266621.02993719],
[382402.00579139],
[139939.3898391 ],
[226055.43239671],
[182107.09970356],
[121485.88875947],
[195003.67226258],
[165090.65506551],
[145664.30310563],
[148769.04632876],
[214500.36498548],
[246241.54771061],
[ 94456.04155043],
[223343.26659941],
[217207.32636295],
[170009.02400688],
[149088.1887089 ],
[235425.09827286],
[ 98536.081703  ],
[187392.80204725],
[210907.73110246],
[232509.55153584],
[279826.68429901],
[144114.52387227],
[135474.45928113],
[163868.06181904],
[182766.13200524],
[128187.10163915],
[150622.98405315],
[159322.61348746],
[148357.27127938],
[743025.21971355],
[155551.83977711],
[240457.84286148],
[143078.34403335],
```

```
[298551.80901866],
[230644.90209707],
[181195.0345639 ],
[294308.34437366],
[203323.70437302],
[137939.0424677 ],
[146890.12384654],
[190520.42982456],
[267549.88683743],
[205103.59278915],
[307498.56959507],
[292455.52369451],
[114385.25096934],
[210387.54587326],
[252355.11008161],
[189331.63648163],
[251565.45623404],
[ 90879.69278712],
[177542.57386486],
[ 53030.37195574],
[226607.97705979],
[ 87965.92466421],
[274090.36630303],
[ 85428.62357044],
[106065.77750238],
[136502.30074341],
[255970.31247257],
[197847.44487186],
[247865.40957079],
[ 97732.22538393],
[111998.60410901],
[144398.24636628],
[121707.46767281],
[202675.21551061],
[166036.40033759],
[ 86144.72075101],
[228742.6324387 ],
[105211.47384933],
[ 93046.19026794],
[160095.21386368],
[260299.12344718],
[149698.79363435],
[187338.47395494],
[103565.30028329],
[258266.33557095],
[259033.38112633],
[234453.62935461],
```

```
[193732.42137807],
[213765.7539408 ],
[165489.64129446],
[155078.86194831],
[376866.30681802],
[229089.70509622],
[204999.3179142 ],
[ 93415.12622785],
[161849.07051946],
[170683.06923189],
[292224.95025341],
[252908.53460924],
[238789.69602916],
[156479.50971986],
[179022.89062795],
[140308.80076853],
[196087.79375953],
[218496.379712  ],
[135184.36736537],
[149271.92016279],
[230767.43735095],
[121437.06399298],
[189597.86732836],
[261466.17235025],
[406504.09361305],
[231331.52004887],
[229793.51465329],
[ 63793.04594103],
[149943.84004108],
[107052.78150569],
[161279.69082688],
[ 79352.32854954],
[228028.29986236],
[148044.37035085],
[185899.72879359],
[ 93905.94604607],
[125185.83879448],
[295549.74893083],
[176891.9777154 ],
[275376.27776821],
[122364.32959639],
[237899.1137713 ],
[120873.48457616],
[112319.42804929],
[154082.65015017],
[227594.2866965 ],
[272591.20204581],
```

```
[169091.56080003],
[137152.35620216],
[174269.84836075],
[174058.72720411],
[ 94442.57028349],
[211522.9850838 ],
[207665.04843719],
[224955.47170888],
[104509.72646237],
[255433.585863  ],
[128122.06170072],
[118399.21517783],
[110821.23521968],
[237763.66129671],
[218355.92784212],
[168908.52965954],
[ 48350.29676843],
[295789.18439683],
[175917.16054315],
[233080.44603378],
[125947.61186632],
[313873.82464856],
[119971.23233157],
[248254.46282618],
[184096.51582416],
[145313.27773392],
[147805.2636603 ],
[165304.11635857],
[175031.72106123],
[136397.31724092],
[288897.06279475],
[154088.35763048],
[142838.58549057],
[208034.50155347],
[206415.5881524 ],
[122740.80389956],
[105490.56007294],
[238515.05914948],
[177080.68608447],
[219191.98641062],
[ 97660.57591277],
[311927.86493777],
[154219.89122836],
[184097.59547413],
[233526.18300273],
[159218.75955721],
[259920.36062813],
```

```
[ 91440.35697917],
[204131.21824647],
[ 91474.77235509],
[140109.27856717],
[276181.35989457],
[148403.50298108],
[ 96984.38074437],
[145830.08338361],
[228946.57626745],
[130140.91055358],
[154455.58296313],
[196157.70928318],
[199649.45820662],
[273129.52892126],
[235914.72814902],
[139445.36304418],
[165421.22736262]])
```