In [200]:
```
#Name: Paul Galvez
#Date: 4/27/23
#Course: DSC 550
#Week 7 Part 1
```

In [201]:
```
#importing libraries

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [202]:
```
#importing the housing CSV data

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

In [203]:
```
df.head()
```

Out[203]:

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

5 rows × 81 columns

In [204]:
```
df.describe()
```

Out[204]:

	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrAre
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1452.00000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.267808	1984.865753	103.68526
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.202904	20.645407	181.06620
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.00000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.000000	1967.000000	0.00000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.000000	1994.000000	0.00000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.000000	2004.000000	166.00000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	1600.00000

8 rows × 38 columns

In [205]:
```
#Drop the "Id" column and any features that are missing more than 40% of their values.

df.drop(['Id'], axis=1, inplace=True)
df.dropna(axis=1, thresh=len(df)*.4, inplace=True)
```

In [206]:

```
#the updated dataframe without the ID and the updated features that are missing
#40% of their values column shown below

df
```

Out[206]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	Enclosec
0	60	RL	65.0	8450	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1	20	RL	80.0	9600	Pave	Reg	Lvl	AllPub	FR2	Gtl	...	
2	60	RL	68.0	11250	Pave	IR1	Lvl	AllPub	Inside	Gtl	...	
3	70	RL	60.0	9550	Pave	IR1	Lvl	AllPub	Corner	Gtl	...	
4	60	RL	84.0	14260	Pave	IR1	Lvl	AllPub	FR2	Gtl	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1455	60	RL	62.0	7917	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1456	20	RL	85.0	13175	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1457	70	RL	66.0	9042	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1458	20	RL	68.0	9717	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1459	20	RL	75.0	9937	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	

1460 rows × 76 columns

In [207]:

```
#For numerical columns, fill in any missing data with the median value.
#For categorical columns, fill in any missing data with the most common value (mode).

def fill_data(df):
    for col in df.columns:
        if df[col].dtype == 'object':
            df[col].fillna(df[col].mode()[0], inplace=True)
        else:
            df[col].fillna(df[col].median(), inplace=True)
    return df

df = fill_data(df)
```

In [208]:

```
#the dataframe below with the missing data for the median value and mode

df
```

Out[208]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	...	Enclosec
0	60	RL	65.0	8450	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1	20	RL	80.0	9600	Pave	Reg	Lvl	AllPub	FR2	Gtl	...	
2	60	RL	68.0	11250	Pave	IR1	Lvl	AllPub	Inside	Gtl	...	
3	70	RL	60.0	9550	Pave	IR1	Lvl	AllPub	Corner	Gtl	...	
4	60	RL	84.0	14260	Pave	IR1	Lvl	AllPub	FR2	Gtl	...	
...	...	...	...	...	...	...	...	...	...	...	...	
1455	60	RL	62.0	7917	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1456	20	RL	85.0	13175	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1457	70	RL	66.0	9042	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1458	20	RL	68.0	9717	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	
1459	20	RL	75.0	9937	Pave	Reg	Lvl	AllPub	Inside	Gtl	...	

1460 rows × 76 columns

In [209]:
```
df.isna().sum()
```

Out[209]:
```
MSSubClass       0
MSZoning         0
LotFrontage      0
LotArea          0
Street           0
                ..
MoSold           0
YrSold           0
SaleType         0
SaleCondition    0
SalePrice        0
Length: 76, dtype: int64
```

In [210]:
```
# Convert the categorical columns to dummy variables.

df = pd.get_dummies(df, drop_first=True)
```

In [211]:

```
df.head(10)
```

Out[211]:

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2
0	60	65.0	8450	7	5	2003	2003	196.0	706	(
1	20	80.0	9600	6	8	1976	1976	0.0	978	(
2	60	68.0	11250	7	5	2001	2002	162.0	486	(
3	70	60.0	9550	7	5	1915	1970	0.0	216	(
4	60	84.0	14260	8	5	2000	2000	350.0	655	(
5	50	85.0	14115	5	5	1993	1995	0.0	732	(
6	20	75.0	10084	8	5	2004	2005	186.0	1369	(
7	60	69.0	10382	7	6	1973	1973	240.0	859	32
8	50	51.0	6120	7	5	1931	1950	0.0	0	(
9	190	50.0	7420	5	6	1939	1950	0.0	851	(

10 rows × 237 columns

In [212]:

```
df.shape
```

Out[212]: (1460, 237)

In [213]:
```
df.describe
```

```
Out[213]: <bound method NDFrame.describe of       MSSubClass  LotFrontage  LotArea  OverallQual  OverallCond  Year
Built  \
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  ...  SaleType_ConLI  \
0             2003       196.0         706           0  ...               0
1             1976         0.0         978           0  ...               0
2             2002       162.0         486           0  ...               0
3             1970         0.0         216           0  ...               0
4             2000       350.0         655           0  ...               0
...            ...         ...         ...         ...  ...             ...
1455          2000         0.0           0           0  ...               0
1456          1988       119.0         790         163  ...               0
1457          2006         0.0         275           0  ...               0
1458          1996         0.0          49        1029  ...               0
1459          1965         0.0         830         290  ...               0

      SaleType_ConLw  SaleType_New  SaleType_Oth  SaleType_WD  \
0                  0             0             0            1
1                  0             0             0            1
2                  0             0             0            1
3                  0             0             0            1
4                  0             0             0            1
...              ...           ...           ...          ...
1455               0             0             0            1
1456               0             0             0            1
1457               0             0             0            1
1458               0             0             0            1
1459               0             0             0            1

      SaleCondition_AdjLand  SaleCondition_Alloca  SaleCondition_Family  \
0                         0                     0                     0
1                         0                     0                     0
```

```
2                     0                     0                     0
3                     0                     0                     0
4                     0                     0                     0
...                 ...                   ...                   ...
1455                  0                     0                     0
1456                  0                     0                     0
1457                  0                     0                     0
1458                  0                     0                     0
1459                  0                     0                     0

      SaleCondition_Normal  SaleCondition_Partial
0                        1                      0
1                        1                      0
2                        1                      0
3                        0                      0
4                        1                      0
...                    ...                    ...
1455                     1                      0
1456                     1                      0
1457                     1                      0
1458                     1                      0
1459                     1                      0

[1460 rows x 237 columns]>
```

In [214]:

```
#Split the data into a training and test set, where the SalePrice column is the target.

X = df.drop(['SalePrice'], axis=1)
y = df['SalePrice']
```

In [215]:

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

In [216]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

In [217]:
```
#Run a linear regression and report the R2-value and RMSE on the test set.

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

In [218]:
```
model_linear_regression = LinearRegression()
model_linear_regression.fit(X_train, y_train)
y_pred = model_linear_regression.predict(X_test)
```

In [219]:
```
#Calculate the RMSE and R2-value.

rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
print(f'The RMSE value is: {rmse}')
print(f'The R2 value is: {r2}')
```

```
The RMSE value is: 47734.04133858645
The R2 value is: 0.6643774377493321
```

In [220]:
```
#Fit and transform the training features with a PCA so that 90% of the variance is retained

from sklearn.decomposition import PCA
my_pca = PCA(.9)
my_pca.fit(X_train)
X_train_pca = pca.transform(X_train)
X_test_pca = pca.transform(X_test)
```

In [221]:
```
#How many features are in the PCA-transformed matrix?

print('The total number of features in the PCA matrix is: {X_train_pca.shape[1]}')
```

```
The total number of features in the PCA matrix is: {X_train_pca.shape[1]}
```

In [222]:
```
#Repeat step 7 with your PCA transformed data.

model_linear_regression = LinearRegression()
model_linear_regression.fit(X_train_pca, y_train)
y_pred = model_linear_regression.predict(X_test_pca)
```

In [223]:
```
mean_squared_error_pca = np.sqrt(mean_squared_error(y_test, y_pred))
r2_pca = r2_score(y_test, y_pred)
print(f'The RMSE value is:: {mean_squared_error_pca}')
print(f'The R2 value is:: {r2_pca}')
```

```
The RMSE value is:: 79269.9564392479
The R2 value is:: 0.07442422802806481
```

In [224]:
```
#Take your original training features (from step 6) and apply a min-max scaler to them.
#Find the min-max scaled features in your training set that have a variance above 0.1
```

In [225]:
```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(X_train)
X_train_scaled = scaler.transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

In [226]:
```
# Transform but DO NOT fit the test features with the same steps applied in steps 11 and 12.
# Repeat step 7 with the high variance data.

model_linear_regression = LinearRegression()
model_linear_regression.fit(X_train_scaled, y_train)
y_pred = model_linear_regression.predict(X_test_scaled)
```

In [227]:
```
mean_squared_error_scaled = np.sqrt(mean_squared_error(y_test, y_pred))
r2_scaled = r2_score(y_test, y_pred)
print(f'The RMSE value is: {mean_squared_error_scaled}')
print(f'R2 value is: {r2_scaled}')
```

```
The RMSE value is: 956664900243605.1
R2 value is: -1.3480761307724928e+20
```