House Prices - Advanced Regression Techniques

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

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

```
#import the python libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

For the Training set

```
#import the training dataset using pandas
df = pd.read_csv('/content/train.csv', encoding = 'latin')
df
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	Mis
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	GdPrv	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	

1460 rows × 81 columns

```
df.replace([np.inf, -np.inf], np.nan, inplace=True)
```

Exploratory Data Analysis

```
#display the first five rows of the train set
df.head()
```

	ID	MSSUBCLASS	MSZONING	LOTFRONTAGE	LOTAREA	STREET	ALLEY	LOTSHAPE	LANDCONTOUR	UTILITIES	...	POOLAREA	POOLQC	FENCE	MISCFEAT
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	N
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	N
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	N
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	N
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	N

5 rows × 81 columns

```
#display the last five rows of the train set
df.tail()
```

	ID	MSSUBCLASS	MSZONING	LOTFRONTAGE	LOTAREA	STREET	ALLEY	LOTSHAPE	LANDCONTOUR	UTILITIES	...	POOLAREA	POOLQC	FENCE	MIS
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	GdPrv	
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	

5 rows × 81 columns

```
#display the shape of the set
df.shape
```

```
(1460, 81)
```

```
#display the columns of the train set
df.columns
```

```
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', 'SALEPRICE'],
      dtype='object')
```

```
#change the column names to upper case
df.columns = df.columns.str.upper()
df.columns
```

```
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', 'SALEPRICE'],
      dtype='object')
```

Separate the train set into numeric dtypes and categorical dtypes

```
df_num = df.select_dtypes(include = {int, float})
df_num.columns
```

```
Index(['ID', 'MSSUBCLASS', 'LOTFRONTAGE', 'LOTAREA', 'OVERALLQUAL',
       'OVERALLCOND', 'YEARBUILT', 'YEARREMODADD', 'MASVNRAREA', 'BSMTFINSF1',
       'BSMTFINSF2', 'BSMTUNFSF', 'TOTALBSMTSF', '1STFLRSF', '2NDFLRSF',
       'LOWQUALFINSF', 'GRLIVAREA', 'BSMTFULLBATH', 'BSMTHALFBATH', 'FULLBATH',
       'HALFBATH', 'BEDROOMABVGR', 'KITCHENABVGR', 'TOTRMSABVGRD',
       'FIREPLACES', 'GARAGEYRBLT', 'GARAGECARS', 'GARAGEAREA', 'WOODDECKSF',
       'OPENPORCHSF', 'ENCLOSEDPORCH', '3SSNPORCH', 'SCREENPORCH', 'POOLAREA',
       'MISCVAL', 'MOSOLD', 'YRSOLD', 'SALEPRICE'],
      dtype='object')
```

```
df_num.shape
```

```
(1460, 38)
```

```
#normalize the train set
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(df_num)
```

```
df_cat = df.select_dtypes(exclude = {int, float})
df_cat.columns
```

```
Index(['MSZONING', 'STREET', 'ALLEY', 'LOTSHAPE', 'LANDCONTOUR', 'UTILITIES',
       'LOTCONFIG', 'LANDSLOPE', 'NEIGHBORHOOD', 'CONDITION1', 'CONDITION2',
       'BLDGTYPE', 'HOUSESTYLE', 'ROOFSTYLE', 'ROOFMATL', 'EXTERIOR1ST',
       'EXTERIOR2ND', 'MASVNRTYPE', 'EXTERQUAL', 'EXTERCOND', 'FOUNDATION',
       'BSMTQUAL', 'BSMTCOND', 'BSMTEXPOSURE', 'BSMTFINTYPE1', 'BSMTFINTYPE2',
       'HEATING', 'HEATINGQC', 'CENTRALAIR', 'ELECTRICAL', 'KITCHENQUAL',
       'FUNCTIONAL', 'FIREPLACEQU', 'GARAGETYPE', 'GARAGEFINISH', 'GARAGEQUAL',
       'GARAGECOND', 'PAVEDDRIVE', 'POOLQC', 'FENCE', 'MISCFEATURE',
       'SALETYPE', 'SALECONDITION'],
      dtype='object')
```

```
df_cat.shape
```

```
(1460, 43)
```

```
#display the iformation about the numerical train set
df_num.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 38 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   ID             1460 non-null   int64  
 1   MSSUBCLASS     1460 non-null   int64  
 2   LOTFRONTAGE    1460 non-null   float64
 3   LOTAREA        1460 non-null   int64  
 4   OVERALLQUAL    1460 non-null   int64  
 5   OVERALLCOND    1460 non-null   int64  
 6   YEARBUILT      1460 non-null   int64  
 7   YEARREMODADD   1460 non-null   int64  
 8   MASVNRAREA     1460 non-null   float64
 9   BSMTFINSF1     1460 non-null   int64  
 10  BSMTFINSF2     1460 non-null   int64  
 11  BSMTUNFSF      1460 non-null   int64  
 12  TOTALBSMTSF    1460 non-null   int64  
 13  1STFLRSF       1460 non-null   int64  
 14  2NDFLRSF       1460 non-null   int64  
 15  LOWQUALFINSF   1460 non-null   int64  
 16  GRLIVAREA      1460 non-null   int64  
 17  BSMTFULLBATH   1460 non-null   int64  
 18  BSMTHALFBATH   1460 non-null   int64  
 19  FULLBATH       1460 non-null   int64  
 20  HALFBATH       1460 non-null   int64  
 21  BEDROOMABVGR   1460 non-null   int64  
 22  KITCHENABVGR   1460 non-null   int64  
 23  TOTRMSABVGRD   1460 non-null   int64  
 24  FIREPLACES     1460 non-null   int64  
 25  GARAGEYRBLT    1460 non-null   float64
 26  GARAGECARS     1460 non-null   int64  
 27  GARAGEAREA     1460 non-null   int64  
 28  WOODDECKSF     1460 non-null   int64  
 29  OPENPORCHSF    1460 non-null   int64  
 30  ENCLOSEDPORCH  1460 non-null   int64  
 31  3SSNPORCH      1460 non-null   int64  
 32  SCREENPORCH    1460 non-null   int64  
 33  POOLAREA       1460 non-null   int64  
 34  MISCVAL        1460 non-null   int64  
 35  MOSOLD         1460 non-null   int64  
 36  YRSOLD         1460 non-null   int64  
 37  SALEPRICE      1460 non-null   int64  
dtypes: float64(3), int64(35)
memory usage: 433.6 KB
```

```
#check for duplicate values
df.duplicated().any()
```

```
False
```

Filling missing values

```
df_num.isna().sum()
```

```
ID                0
MSSUBCLASS        0
LOTFRONTAGE     259
LOTAREA           0
OVERALLQUAL       0
OVERALLCOND       0
YEARBUILT         0
YEARREMODADD      0
MASVNRAREA        8
BSMTFINSF1        0
BSMTFINSF2        0
BSMTUNFSF         0
TOTALBSMTSF       0
1STFLRSF          0
2NDFLRSF          0
LOWQUALFINSF      0
GRLIVAREA         0
BSMTFULLBATH      0
BSMTHALFBATH      0
FULLBATH          0
HALFBATH          0
BEDROOMABVGR      0
KITCHENABVGR      0
TOTRMSABVGRD      0
FIREPLACES        0
GARAGEYRBLT      81
GARAGECARS        0
GARAGEAREA        0
WOODDECKSF        0
OPENPORCHSF       0
ENCLOSEDPORCH     0
3SSNPORCH         0
SCREENPORCH       0
POOLAREA          0
MISCVAL           0
MOSOLD            0
YRSOLD            0
SALEPRICE         0
dtype: int64
```

```
df_num['LOTFRONTAGE'].mean()
df_num['GARAGEYRBLT'].mean()
df_num['MASVNRAREA'].mean()
```

```
103.68526170798899
```

```
#fill the missing values with the average values
df_num['LOTFRONTAGE'].fillna(df_num['LOTFRONTAGE'].mean(), inplace = True)
df_num['GARAGEYRBLT'].fillna(df_num['GARAGEYRBLT'].mean(), inplace = True)
df_num['MASVNRAREA'].fillna(df_num['MASVNRAREA'].mean(), inplace = True)
```

```
df_num.isna().any().any()
```

```
False
```

```
df_num
```

	ID	MSSUBCLASS	LOTFRONTAGE	LOTAREA	OVERALLQUAL	OVERALLCOND	YEARBUILT	YEARREMODADD	MASVNRAREA	BSMTFINSF1	...	WOODDECKSF
0	1	60	65.0	8450	7	5	2003	2003	196.0	706	...	0
1	2	20	80.0	9600	6	8	1976	1976	0.0	978	...	298
2	3	60	68.0	11250	7	5	2001	2002	162.0	486	...	0
3	4	70	60.0	9550	7	5	1915	1970	0.0	216	...	0
4	5	60	84.0	14260	8	5	2000	2000	350.0	655	...	192
...	...	...	...	...	...	...	...	...	...	...	...	...

For the testing set

```
#import the test set using pandas
data = pd.read_csv('/content/test.csv', encoding = 'latin')
```

```
#display the first five rows of the test set
data.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN

5 rows × 80 columns

```
#display the last five rows of the test set
data.tail()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQ
1454	2915	160	RM	21.0	1936	Pave	NaN	Reg	Lvl	AllPub	...	0	0	Na
1455	2916	160	RM	21.0	1894	Pave	NaN	Reg	Lvl	AllPub	...	0	0	Na
1456	2917	20	RL	160.0	20000	Pave	NaN	Reg	Lvl	AllPub	...	0	0	Na
1457	2918	85	RL	62.0	10441	Pave	NaN	Reg	Lvl	AllPub	...	0	0	Na
1458	2919	60	RL	74.0	9627	Pave	NaN	Reg	Lvl	AllPub	...	0	0	Na

5 rows × 80 columns

```
data.replace([np.inf, -np.inf], np.nan, inplace=True)
```

```
#display the shape of the set
data.shape
```

```
(1459, 80)
```

```
#change the column names of the set to upper case
data.columns = data.columns.str.upper()
data.columns
```

```
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'],
      dtype='object')
```

Separate the test set into numeric dtypes and categorical dtypes

```
data_num = data.select_dtypes(include = {int, float})
data_num.columns
```

```
Index(['ID', 'MSSUBCLASS', 'LOTFRONTAGE', 'LOTAREA', 'OVERALLQUAL',
       'OVERALLCOND', 'YEARBUILT', 'YEARREMODADD', 'MASVNRAREA', 'BSMTFINSF1',
       'BSMTFINSF2', 'BSMTUNFSF', 'TOTALBSMTSF', '1STFLRSF', '2NDFLRSF',
       'LOWQUALFINSF', 'GRLIVAREA', 'BSMTFULLBATH', 'BSMTHALFBATH', 'FULLBATH',
       'HALFBATH', 'BEDROOMABVGR', 'KITCHENABVGR', 'TOTRMSABVGRD',
       'FIREPLACES', 'GARAGEYRBLT', 'GARAGECARS', 'GARAGEAREA', 'WOODDECKSF',
       'OPENPORCHSF', 'ENCLOSEDPORCH', '3SSNPORCH', 'SCREENPORCH', 'POOLAREA',
       'MISCVAL', 'MOSOLD', 'YRSOLD'],
      dtype='object')
```

```
X1_scaled = scaler.fit_transform(data_num)
```

```
#display the information of the test set
data_num.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1459 entries, 0 to 1458
Data columns (total 37 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   ID             1459 non-null   int64  
 1   MSSUBCLASS     1459 non-null   int64  
 2   LOTFRONTAGE    1232 non-null   float64
 3   LOTAREA        1459 non-null   int64  
 4   OVERALLQUAL    1459 non-null   int64  
 5   OVERALLCOND    1459 non-null   int64  
 6   YEARBUILT      1459 non-null   int64  
 7   YEARREMODADD   1459 non-null   int64  
 8   MASVNRAREA     1444 non-null   float64
 9   BSMTFINSF1     1458 non-null   float64
 10  BSMTFINSF2     1458 non-null   float64
 11  BSMTUNFSF      1458 non-null   float64
 12  TOTALBSMTSF    1458 non-null   float64
 13  1STFLRSF       1459 non-null   int64  
 14  2NDFLRSF       1459 non-null   int64  
 15  LOWQUALFINSF   1459 non-null   int64  
 16  GRLIVAREA      1459 non-null   int64  
 17  BSMTFULLBATH   1457 non-null   float64
 18  BSMTHALFBATH   1457 non-null   float64
 19  FULLBATH       1459 non-null   int64  
 20  HALFBATH       1459 non-null   int64  
 21  BEDROOMABVGR   1459 non-null   int64  
 22  KITCHENABVGR   1459 non-null   int64  
 23  TOTRMSABVGRD   1459 non-null   int64  
 24  FIREPLACES     1459 non-null   int64  
 25  GARAGEYRBLT    1381 non-null   float64
 26  GARAGECARS     1458 non-null   float64
 27  GARAGEAREA     1458 non-null   float64
 28  WOODDECKSF     1459 non-null   int64  
 29  OPENPORCHSF    1459 non-null   int64  
 30  ENCLOSEDPORCH  1459 non-null   int64  
 31  3SSNPORCH      1459 non-null   int64  
 32  SCREENPORCH    1459 non-null   int64  
 33  POOLAREA       1459 non-null   int64  
 34  MISCVAL        1459 non-null   int64  
 35  MOSOLD         1459 non-null   int64  
 36  YRSOLD         1459 non-null   int64  
dtypes: float64(11), int64(26)
memory usage: 421.9 KB
```

Filling missing values

```
data_num.isna().sum()
```

```
ID                0
MSSUBCLASS        0
```

```
LOTFRONTAGE      227
LOTAREA            0
OVERALLQUAL        0
OVERALLCOND        0
YEARBUILT          0
YEARREMODADD       0
MASVNRAREA        15
BSMTFINSF1         1
BSMTFINSF2         1
BSMTUNFSF          1
TOTALBSMTSF        1
1STFLRSF           0
2NDFLRSF           0
LOWQUALFINSF       0
GRLIVAREA          0
BSMTFULLBATH       2
BSMTHALFBATH       2
FULLBATH           0
HALFBATH           0
BEDROOMABVGR       0
KITCHENABVGR       0
TOTRMSABVGRD       0
FIREPLACES         0
GARAGEYRBLT       78
GARAGECARS         1
GARAGEAREA         1
WOODDECKSF         0
OPENPORCHSF        0
ENCLOSEDPORCH      0
3SSNPORCH          0
SCREENPORCH        0
POOLAREA           0
MISCVAL            0
MOSOLD             0
YRSOLD             0
dtype: int64
```

```
data_num.isna().any().any()
```

```
True
```

```
#fill the missing values with average values
data_num['LOTFRONTAGE'].fillna(data_num['LOTFRONTAGE'].mean(), inplace = True)
data_num['TOTALBSMTSF'].fillna(data_num['TOTALBSMTSF'].mean(), inplace = True)
data_num['BSMTUNFSF'].fillna(data_num['BSMTUNFSF'].mean(), inplace = True)
data_num['BSMTFINSF1'].fillna(data_num['BSMTFINSF1'].mean(), inplace = True)
data_num['BSMTFINSF2'].fillna(data_num['BSMTFINSF2'].mean(), inplace = True)
data_num['LOTFRONTAGE'].fillna(data_num['LOTFRONTAGE'].mean(), inplace = True)
data_num['GARAGEYRBLT'].fillna(data_num['GARAGEYRBLT'].mean(), inplace = True)
data_num['GARAGECARS'].fillna(data_num['GARAGECARS'].mean(), inplace = True)
data_num['GARAGEAREA'].fillna(data_num['GARAGEAREA'].mean(), inplace = True)
data_num['BSMTFULLBATH'].fillna(data_num['BSMTFULLBATH'].mean(), inplace = True)
data_num['BSMTHALFBATH'].fillna(data_num['BSMTHALFBATH'].mean(), inplace = True)
```

```
data_num['MASVNRAREA'].fillna(data_num['MASVNRAREA'].mean(), inplace = True)
```

```
data_num.isna().any().any()
```

```
False
```

Predictive Modelling in Machine Learning (Multiple Linear Regression)

```
#import the libraries for regression
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```
X_train = df_num[['ID', 'MSSUBCLASS', 'LOTFRONTAGE', 'LOTAREA', 'OVERALLQUAL',
       'OVERALLCOND', 'YEARBUILT', 'YEARREMODADD', 'MASVNRAREA', 'BSMTFINSF1',
       'BSMTFINSF2', 'BSMTUNFSF', 'TOTALBSMTSF', '1STFLRSF', '2NDFLRSF',
       'LOWQUALFINSF', 'GRLIVAREA', 'BSMTFULLBATH', 'BSMTHALFBATH', 'FULLBATH',
       'HALFBATH', 'BEDROOMABVGR', 'KITCHENABVGR', 'TOTRMSABVGRD',
       'FIREPLACES', 'GARAGEYRBLT', 'GARAGECARS', 'GARAGEAREA', 'WOODDECKSF',
```

```
        'OPENPORCHSF', 'ENCLOSEDPORCH', '3SSNPORCH', 'SCREENPORCH', 'POOLAREA',
        'MISCVAL', 'MOSOLD', 'YRSOLD',]]


X_test = data_num[['ID', 'MSSUBCLASS', 'LOTFRONTAGE', 'LOTAREA', 'OVERALLQUAL',
       'OVERALLCOND', 'YEARBUILT', 'YEARREMODADD', 'MASVNRAREA', 'BSMTFINSF1',
       'BSMTFINSF2', 'BSMTUNFSF', 'TOTALBSMTSF', '1STFLRSF', '2NDFLRSF',
       'LOWQUALFINSF', 'GRLIVAREA', 'BSMTFULLBATH', 'BSMTHALFBATH', 'FULLBATH',
       'HALFBATH', 'BEDROOMABVGR', 'KITCHENABVGR', 'TOTRMSABVGRD',
       'FIREPLACES', 'GARAGEYRBLT', 'GARAGECARS', 'GARAGEAREA', 'WOODDECKSF',
       'OPENPORCHSF', 'ENCLOSEDPORCH', '3SSNPORCH', 'SCREENPORCH', 'POOLAREA',
       'MISCVAL', 'MOSOLD', 'YRSOLD',]]


Y_train = df_num['SALEPRICE']


model = LinearRegression()


model.fit(X_train, Y_train)
```

```
LinearRegression
LinearRegression()
```

```
y_pred = model.predict(X_test)
y_pred
```

```
array([115926.59460149, 151030.34249914, 171983.97743377, ...,
       168773.52661752,  98928.72035158, 250497.41978714])
```

```
y_pred.shape
```

```
(1459,)
```