We want to predict the price of houses by creating a model that will help us predict the prices

Let's start by importing the libraries needed for the project

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
sns.set_style("whitegrid")
import warnings
warnings.filterwarnings("ignore")
```

Let's load the datasets

```
"""df1 = pd.read_csv("test.csv")"""

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

```
df.shape
```

```
    (1460, 81)
```

EDA

```
df.head()
```

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

5 rows × 81 columns

Missing Values

```
df.isnull().sum()
```

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

Taking care of the missing values

```
df.dropna(axis=1, inplace = True)
```

```
df.shape
```

```
(1460, 62)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 62 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   LotArea        1460 non-null   int64
 4   Street         1460 non-null   object
 5   LotShape       1460 non-null   object
 6   LandContour    1460 non-null   object
 7   Utilities      1460 non-null   object
 8   LotConfig      1460 non-null   object
 9   LandSlope      1460 non-null   object
 10  Neighborhood   1460 non-null   object
 11  Condition1     1460 non-null   object
 12  Condition2     1460 non-null   object
 13  BldgType       1460 non-null   object
 14  HouseStyle     1460 non-null   object
 15  OverallQual    1460 non-null   int64
 16  OverallCond    1460 non-null   int64
 17  YearBuilt      1460 non-null   int64
 18  YearRemodAdd   1460 non-null   int64
 19  RoofStyle      1460 non-null   object
 20  RoofMatl       1460 non-null   object
 21  Exterior1st    1460 non-null   object
 22  Exterior2nd    1460 non-null   object
 23  ExterQual      1460 non-null   object
 24  ExterCond      1460 non-null   object
 25  Foundation     1460 non-null   object
 26  BsmtFinSF1     1460 non-null   int64
 27  BsmtFinSF2     1460 non-null   int64
 28  BsmtUnfSF      1460 non-null   int64
 29  TotalBsmtSF    1460 non-null   int64
 30  Heating        1460 non-null   object
 31  HeatingQC      1460 non-null   object
 32  CentralAir     1460 non-null   object
 33  1stFlrSF       1460 non-null   int64
 34  2ndFlrSF       1460 non-null   int64
 35  LowQualFinSF   1460 non-null   int64
 36  GrLivArea      1460 non-null   int64
 37  BsmtFullBath   1460 non-null   int64
 38  BsmtHalfBath   1460 non-null   int64
 39  FullBath       1460 non-null   int64
 40  HalfBath       1460 non-null   int64
 41  BedroomAbvGr   1460 non-null   int64
 42  KitchenAbvGr   1460 non-null   int64
 43  KitchenQual    1460 non-null   object
 44  TotRmsAbvGrd   1460 non-null   int64
 45  Functional     1460 non-null   object
 46  Fireplaces     1460 non-null   int64
 47  GarageCars     1460 non-null   int64
 48  GarageArea     1460 non-null   int64
 49  PavedDrive     1460 non-null   object
 50  WoodDeckSF     1460 non-null   int64
 51  OpenPorchSF    1460 non-null   int64
 52  EnclosedPorch  1460 non-null   int64
 53  3SsnPorch      1460 non-null   int64
```

```
df.describe()
```

	Id	MSSubClass	LotArea	OverallQual	OverallCond	YearBuilt
count	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000

```
df.columns
```

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotArea', 'Street', 'LotShape',
       'LandContour', 'Utilities', 'LotConfig', 'LandSlope', 'Neighborhood',
       'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'OverallQual',
       'OverallCond', 'YearBuilt', 'YearRemodAdd', 'RoofStyle', 'RoofMatl',
       'Exterior1st', 'Exterior2nd', 'ExterQual', 'ExterCond', 'Foundation',
       'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
       'HeatingQC', 'CentralAir', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF',
       'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath',
       'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual', 'TotRmsAbvGrd',
       'Functional', 'Fireplaces', 'GarageCars', 'GarageArea', 'PavedDrive',
       'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3SsnPorch',
       'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
       'SaleCondition', 'SalePrice'],
      dtype='object')
```

EDA

```
df.nunique()
```

```
Id               1460
MSSubClass         15
MSZoning            5
LotArea          1073
Street              2
                 ... 
MoSold             12
YrSold              5
SaleType            9
SaleCondition       6
SalePrice         663
Length: 62, dtype: int64
```

```
df.Street.value_counts()
```

```
Pave    1454
Grvl       6
Name: Street, dtype: int64
```

```
df.MSZoning.value_counts()
```

```
RL         1151
RM          218
FV           65
RH           16
C (all)      10
Name: MSZoning, dtype: int64
```

```
df.YrSold.value_counts()
```

```
     2009    338
     2007    329
     2006    314
     2008    304
     2010    175
     Name: YrSold, dtype: int64
```

```
df.MoSold.value_counts()
```

```
     6     253
     7     234
     5     204
     4     141
     8     122
     3     106
     10     89
     11     79
     9      63
     12     59
     1      58
     2      52
     Name: MoSold, dtype: int64
```

```
df = df.select_dtypes(exclude=['object'])
df
```

```
            Id  MSSubClass  LotArea  OverallQual  OverallCond  YearBuilt  YearRemodA
```

```
X = df.drop('SalePrice',axis = 1).values
Y = df.SalePrice.values
Y = np.reshape (Y, (-1,1))
```

```
print(X.shape)
print(Y.shape)
```

```
(1460, 34)
(1460, 1)
```

Lets split the dataset into Training et Testing set

```
# We will split the data by using train_test_split from scikit
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,Y,test_size=0.25,random_state=0)
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(1095, 34)
(365, 34)
(1095, 1)
(365, 1)
```

Feature Engineering

```
X = np.append(arr = np.ones((1460,1)), values = X, axis=1)
```

```
import statsmodels.api as sm
X_opt = X[:,[0,1,2,3,4,5]]
regressor = sm.OLS(endog = Y, exog = X_opt).fit()
regressor.summary()
```

```
                        OLS Regression Results
   Dep. Variable:   y                         R-squared:  0.666
          Model:    OLS                   Adj. R-squared: 0.665
         Method:    Least Squares            F-statistic: 579.5
           Date:    Thu, 26 May 2022 Prob (F-statistic):  0.00
           Time:    21:47:03              Log-Likelihood: -17744.
No. Observations:   1460                             AIC: 3.550e+04
    Df Residuals:   1454                             BIC: 3.553e+04
        Df Model:   5
 Covariance Type:   nonrobust
           coef       std err      t       P>|t|   [0.025      0.975]
const  -9.22e+04   9029.698  -10.211  0.000  -1.1e+05    -7.45e+04
  x1    1.2840        2.859     0.449  0.653  -4.325       6.893
  x2   -162.5297     28.833    -5.637  0.000  -219.089    -105.970
  x3    1.3523        0.123    11.024  0.000  1.112        1.593
  x4    4.452e+04   880.507    50.561  0.000  4.28e+04     4.62e+04
  x5   -775.8548   1088.662    -0.713  0.476  -2911.371    1359.661
     Omnibus:      593.097  Durbin-Watson:     1.968
Prob(Omnibus):     0.000   Jarque-Bera (JB):   7698.739
        Skew:      1.525          Prob(JB):    0.00
    Kurtosis:      13.829         Cond. No.    1.09e+05
```

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.09e+05. This might indicate that there are
strong multicollinearity or other numerical problems.