

```
#Importing essential libraries
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

```
file_path = '/content/drive/MyDrive/test.csv'
r=pd.read_csv(file_path)
print(r.head())
```

```
<bound method NDFrame.head of
0      1461      20      RH      80.0      11622      Pave      NaN      Reg
1      1462      20      RL      81.0      14267      Pave      NaN      IR1
2      1463      60      RL      74.0      13830      Pave      NaN      IR1
3      1464      60      RL      78.0      9978      Pave      NaN      IR1
4      1465      120     RL      43.0      5005      Pave      NaN      IR1
...      ...      ...      ...      ...      ...      ...      ...
1454    2915      160     RM      21.0      1936      Pave      NaN      Reg
1455    2916      160     RM      21.0      1894      Pave      NaN      Reg
1456    2917      20      RL      160.0     20000      Pave      NaN      Reg
1457    2918      85      RL      62.0     10441      Pave      NaN      Reg
1458    2919      60      RL      74.0      9627      Pave      NaN      Reg

      LandContour Utilities ... ScreenPorch PoolArea PoolQC Fence \
0      Lvl1 AllPub ...      120      0      NaN MnPrv
1      Lvl1 AllPub ...      0      0      NaN NaN
2      Lvl1 AllPub ...      0      0      NaN MnPrv
3      Lvl1 AllPub ...      0      0      NaN NaN
4      HLS AllPub ...      144      0      NaN NaN
...      ...      ...      ...      ...      ...      ...
1454    Lvl1 AllPub ...      0      0      NaN NaN
1455    Lvl1 AllPub ...      0      0      NaN NaN
1456    Lvl1 AllPub ...      0      0      NaN NaN
1457    Lvl1 AllPub ...      0      0      NaN MnPrv
1458    Lvl1 AllPub ...      0      0      NaN NaN

      MiscFeature MiscVal MoSold YrSold SaleType SaleCondition
0      NaN      0      6      2010      WD      Normal
1      Gar2     12500      6      2010      WD      Normal
2      NaN      0      3      2010      WD      Normal
3      NaN      0      6      2010      WD      Normal
4      NaN      0      1      2010      WD      Normal
...      ...      ...      ...      ...      ...      ...
1454    NaN      0      6      2006      WD      Normal
1455    NaN      0      4      2006      WD      Abnorml
1456    NaN      0      9      2006      WD      Abnorml
1457    Shed     700      7      2006      WD      Normal
1458    NaN      0     11      2006      WD      Normal
```

[1459 rows x 80 columns]>

```
file_path='/content/drive/MyDrive/train.csv'
p=pd.read_csv(file_path)
print(p.head())
```

```
<bound method NDFrame.head of
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
...      ...      ...      ...      ...      ...      ...
1455    1456      60      RL      62.0      7917      Pave      NaN      Reg
1456    1457      20      RL      85.0     13175      Pave      NaN      Reg
1457    1458      70      RL      66.0      9042      Pave      NaN      Reg
1458    1459      20      RL      68.0      9717      Pave      NaN      Reg
1459    1460      20      RL      75.0      9937      Pave      NaN      Reg

      LandContour Utilities ... PoolArea PoolQC Fence MiscFeature MiscVal \
0      Lvl1 AllPub ...      0      NaN      NaN      NaN      0
1      Lvl1 AllPub ...      0      NaN      NaN      NaN      0
2      Lvl1 AllPub ...      0      NaN      NaN      NaN      0
3      Lvl1 AllPub ...      0      NaN      NaN      NaN      0
4      Lvl1 AllPub ...      0      NaN      NaN      NaN      0
...      ...      ...      ...      ...      ...      ...
1455    Lvl1 AllPub ...      0      NaN      NaN      NaN      0
1456    Lvl1 AllPub ...      0      NaN      MnPrv      NaN      0
1457    Lvl1 AllPub ...      0      NaN      GdPrv      Shed     2500
1458    Lvl1 AllPub ...      0      NaN      NaN      NaN      0
1459    Lvl1 AllPub ...      0      NaN      NaN      NaN      0

      MoSold YrSold SaleType SaleCondition SalePrice
0      2      2008      WD      Normal      208500
1      5      2007      WD      Normal      181500
2      9      2008      WD      Normal      223500
```

3	2	2006	WD	Abnormal	140000
4	12	2008	WD	Normal	250000
...	...	...	...	...	...
1455	8	2007	WD	Normal	175000
1456	2	2010	WD	Normal	210000
1457	5	2010	WD	Normal	266500
1458	4	2010	WD	Normal	142125
1459	6	2008	WD	Normal	147500

[1460 rows x 81 columns]>

```
print("Columns in the training dataset:", r.columns.tolist())
print("Columns in the testing dataset:", p.columns.tolist())
```

```
'ScreenPorch', 'PoolArea', 'PoolQC', 'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType', 'SaleCondition']
ScreenPorch', 'PoolArea', 'PoolQC', 'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType', 'SaleCondition', 'SalePrice']
```

```
# Extract relevant features and target from training data
X_train = r[['GrLivArea', 'BedroomAbvGr', 'FullBath', 'HalfBath']]
y_train = p['SalePrice']
```

```
# Extract relevant features from testing data
X_test = r[['GrLivArea', 'BedroomAbvGr', 'FullBath', 'HalfBath']]
```

```
# Handle missing values in training data
train_df = p.dropna(subset=['GrLivArea', 'BedroomAbvGr', 'FullBath', 'HalfBath', 'SalePrice'])
```

```
# Extract relevant features and target from training data
X_train = train_df[['GrLivArea', 'BedroomAbvGr', 'FullBath', 'HalfBath']]
y_train = train_df['SalePrice']
```

```
# Ensure consistent lengths
print(f"X_train shape: {X_train.shape}")
print(f"y_train shape: {y_train.shape}")
```

```
X_train shape: (1460, 4)
y_train shape: (1460,)
```

```
#Initialize and train the linear regression model
model=LinearRegression()
model.fit(X_train,y_train)
```

```
LinearRegression
LinearRegression()
```

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

```
print(y_pred.shape)
```

```
(1459,)
```

```
# Evaluate the model if 'SalePrice' exists in the test data (for validation purposes)
if 'SalePrice' in r.columns:
    y_test = r['SalePrice']
    mse = mean_squared_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)
    print("Model Evaluation:")
    print(f"Mean Squared Error (MSE): {mse}")
    print(f"R-squared (R2): {r2}")
```

```
# Display coefficients
print("\nModel Coefficients:")
print(f"Intercept: {model.intercept_}")
print(f"Coefficients: {model.coef_}")
```

```
Model Coefficients:
Intercept: 47997.69971509653
```


Coefficients: [ 108.22377873 -27911.62493864 30380.78357956 3586.62008062]

```
from google.colab import drive
```

```
# Mount Google Drive
drive.mount('/content/drive')
```

```
# Save predictions to a CSV file in Google Drive
output_path = '/content/drive/My Drive/test_with_predictions.csv'
p.to_csv(output_path, index=False)
```

```
print(f"Predictions saved to Google Drive at: {output_path}")
```

 Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).  
Predictions saved to Google Drive at: /content/drive/My Drive/test\_with\_predictions.csv