

✓ Roll Number:- 22102B2006

Name:- Khushil Girish Bhimani

Github Link:- <https://github.com/KhushilBhimani2004/Machine-Learning>

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from sklearn.model_selection import train_test_split
6 from sklearn.linear_model import LinearRegression
7 from sklearn.metrics import mean_squared_error as mse, r2_score
8 from sklearn.metrics import accuracy_score
9 import plotly.express as px
```

```
1 df = pd.read_csv("/content/housing.csv")
```

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

| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population | households | median_income | median_house_value |
|---|-----------|----------|--------------------|-------------|----------------|------------|------------|---------------|--------------------|
| 0 | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 | 126.0 | 8.3252 | 452600.0 |
| 1 | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 | 1138.0 | 8.3014 | 358500.0 |
| 2 | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 | 177.0 | 7.2574 | 352100.0 |
| 3 | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 | 219.0 | 5.6431 | 341300.0 |
| 4 | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 | 259.0 | 3.8462 | 342200.0 |
| 5 | -122.25 | 37.85 | 52.0 | 919.0 | 213.0 | 413.0 | 193.0 | 4.0368 | 269700.0 |
| 6 | -122.25 | 37.84 | 52.0 | 2535.0 | 489.0 | 1094.0 | 514.0 | 3.6591 | 299200.0 |
| 7 | -122.25 | 37.84 | 52.0 | 3104.0 | 687.0 | 1157.0 | 647.0 | 3.1200 | 241400.0 |
| 8 | -122.26 | 37.84 | 42.0 | 2555.0 | 665.0 | 1206.0 | 595.0 | 2.0804 | 226700.0 |
| 9 | -122.25 | 37.84 | 52.0 | 3549.0 | 707.0 | 1551.0 | 714.0 | 3.6912 | 261100.0 |

Next steps: [Generate code with df](#) [View recommended plots](#)

```
1 # from google.colab import drive
2 # drive.mount('/content/drive')
```

```
1 df.shape
```

```
(20640, 10)
```

```
1 df.columns
```


```
Index(['longitude', 'latitude', 'housing_median_age', 'total_rooms',
      'total_bedrooms', 'population', 'households', 'median_income',
      'median_house_value', 'ocean_proximity'],
      dtype='object')
```

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  ---
0   longitude              20640 non-null  float64
1   latitude               20640 non-null  float64
2   housing_median_age     20640 non-null  float64
3   total_rooms            20640 non-null  float64
4   total_bedrooms         20433 non-null  float64
5   population             20640 non-null  float64
6   households              20640 non-null  float64
7   median_income          20640 non-null  float64
8   median_house_value     20640 non-null  float64
9   ocean_proximity        20640 non-null  object
dtypes: float64(9), object(1)
```


memory usage: 1.6+ MB

1 df.describe()



| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | populat: |
|--------------|--------------|--------------|--------------------|--------------|----------------|--------------|
| count | 20640.000000 | 20640.000000 | 20640.000000 | 20640.000000 | 20433.000000 | 20640.000000 |
| mean | -119.569704 | 35.631861 | 28.639486 | 2635.763081 | 537.870553 | 1193.202208 |
| std | 2.003532 | 2.135952 | 12.585558 | 2181.615252 | 421.385070 | 1496.931006 |
| min | -124.350000 | 32.540000 | 1.000000 | 2.000000 | 1.000000 | 1.000000 |
| 25% | -121.800000 | 33.930000 | 18.000000 | 1447.750000 | 296.000000 | 1193.202208 |
| 50% | -118.490000 | 34.260000 | 29.000000 | 2127.000000 | 435.000000 | 1193.202208 |
| 75% | -118.010000 | 37.710000 | 37.000000 | 3148.000000 | 647.000000 | 1193.202208 |
| max | -114.310000 | 41.950000 | 52.000000 | 39320.000000 | 6445.000000 | 1193.202208 |

1 df.isnull().sum()



```

longitude      0
latitude       0
housing_median_age  0
total_rooms    0
total_bedrooms 207
population     0
households     0
median_income  0
median_house_value  0
ocean_proximity  0
dtype: int64

```

1 df.duplicated().sum()

 0


```

1 # med_value = df['total_bedrooms'].median()
2 # med_value

```


1 df =df.dropna(axis=0, how='any')

1 df.sample(10)



| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | populat: |
|--------------|-----------|----------|--------------------|-------------|----------------|----------|
| 13868 | -117.31 | 34.35 | 9.0 | 2404.0 | 390.0 | 107 |
| 19338 | -122.82 | 38.53 | 27.0 | 1823.0 | 360.0 | 90 |
| 5620 | -118.25 | 33.78 | 32.0 | 296.0 | 139.0 | 51 |
| 4694 | -118.37 | 34.07 | 52.0 | 2195.0 | 435.0 | 88 |
| 2126 | -119.71 | 36.77 | 11.0 | 5112.0 | 1384.0 | 248 |
| 15686 | -122.42 | 37.79 | 48.0 | 4506.0 | 1342.0 | 198 |
| 17892 | -121.91 | 37.36 | 42.0 | 3224.0 | 708.0 | 194 |
| 7441 | -118.20 | 33.94 | 45.0 | 1570.0 | 328.0 | 132 |
| 6629 | -118.15 | 34.16 | 18.0 | 1711.0 | 383.0 | 147 |
| 13279 | -117.65 | 34.10 | 44.0 | 1526.0 | 337.0 | 83 |

1 df.isnull().sum()



```

longitude      0
latitude       0
housing_median_age  0
total_rooms    0
total_bedrooms  0
population     0
households     0
median_income  0
median_house_value  0
ocean_proximity  0
dtype: int64

```

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20433 entries, 0 to 20639
Data columns (total 10 columns):
#   Column                Non-Null Count  Dtype
---  -
0   longitude              20433 non-null  float64
1   latitude               20433 non-null  float64
2   housing_median_age     20433 non-null  float64
3   total_rooms            20433 non-null  float64
4   total_bedrooms        20433 non-null  float64
5   population             20433 non-null  float64
6   households            20433 non-null  float64
7   median_income          20433 non-null  float64
8   median_house_value     20433 non-null  float64
9   ocean_proximity        20433 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.7+ MB
```

```
1 unique_val = df['ocean_proximity'].unique()
2 unique_val
```

```
array(['NEAR BAY', '<1H OCEAN', 'INLAND', 'NEAR OCEAN', 'ISLAND'],
      dtype=object)
```

```
1 df['ocean_proximity'] = df['ocean_proximity'].map({'NEAR BAY': 1, '<1H OCEAN': 2, 'INLAND': 3, 'NEAR OCEAN': 4, 'ISLAND': 5})
2 #
3 #
```

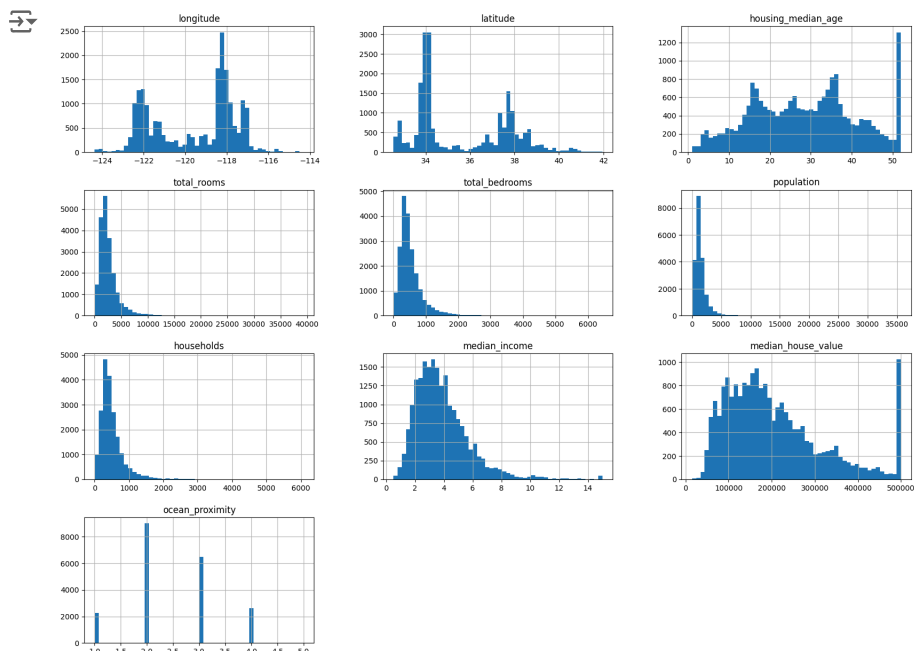
```
1 df.head(10)
```

```
<class 'pandas.core.frame.DataFrame'>
longititude  latitude  housing_median_age  total_rooms  total_bedrooms  population
0   -122.23      37.88              41.0          880.0           129.0         322.0
1   -122.22      37.86              21.0         7099.0          1106.0        2401.0
2   -122.24      37.85              52.0         1467.0           190.0         496.0
3   -122.25      37.85              52.0         1274.0           235.0         558.0
4   -122.25      37.85              52.0         1627.0           280.0         565.0
5   -122.25      37.85              52.0           919.0           213.0         413.0
6   -122.25      37.84              52.0         2535.0           489.0        1094.0
7   -122.25      37.84              52.0         3104.0           687.0        1157.0
8   -122.26      37.84              42.0         2555.0           665.0        1206.0
9   -122.25      37.84              52.0         3549.0           707.0        1551.0
```


Next steps:

[Generate code with df](#)
[View recommended plots](#)

```
1 df.hist(bins=50, figsize=(20,15))
2 plt.show()
```




```
1 df.corr()
```



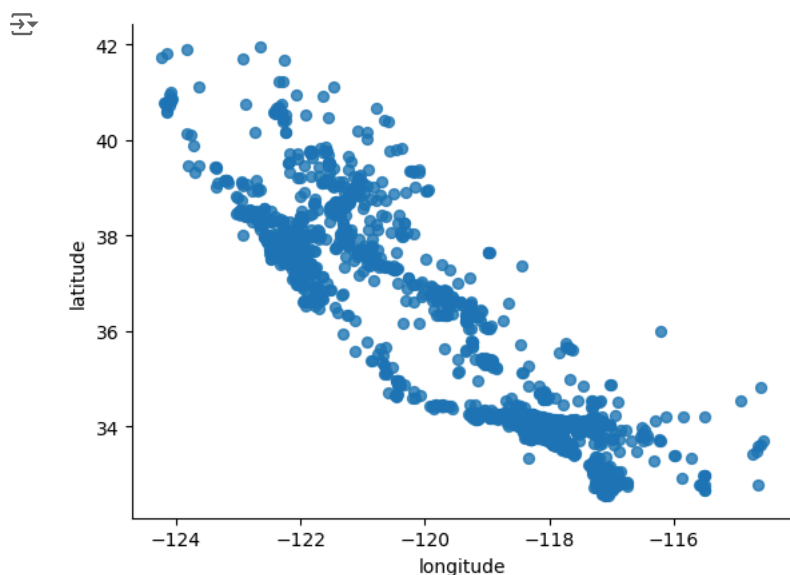
| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms |
|--------------------|-----------|-----------|--------------------|-------------|----------------|
| longitude | 1.000000 | -0.924616 | -0.109357 | 0.045480 | 0.061821 |
| latitude | -0.924616 | 1.000000 | 0.011899 | -0.036667 | -0.062549 |
| housing_median_age | -0.109357 | 0.011899 | 1.000000 | -0.360628 | -0.321772 |
| total_rooms | 0.045480 | -0.036667 | -0.360628 | 1.000000 | 0.939549 |
| total_bedrooms | 0.069608 | -0.066983 | -0.320451 | 0.930380 | 1.000000 |
| population | 0.100270 | -0.108997 | -0.295787 | 0.857281 | 0.871406 |
| households | 0.056513 | -0.071774 | -0.302768 | 0.918992 | 0.971912 |
| median_income | -0.015550 | -0.079626 | -0.118278 | 0.197882 | -0.000931 |
| median_house_value | -0.045398 | -0.144638 | 0.106432 | 0.133294 | 0.045794 |
| ocean_proximity | 0.181198 | -0.067980 | -0.206178 | 0.015917 | 0.000941 |

```
1 corrln = df.corr()['median_house_value']
2 round(corrln,2)
```

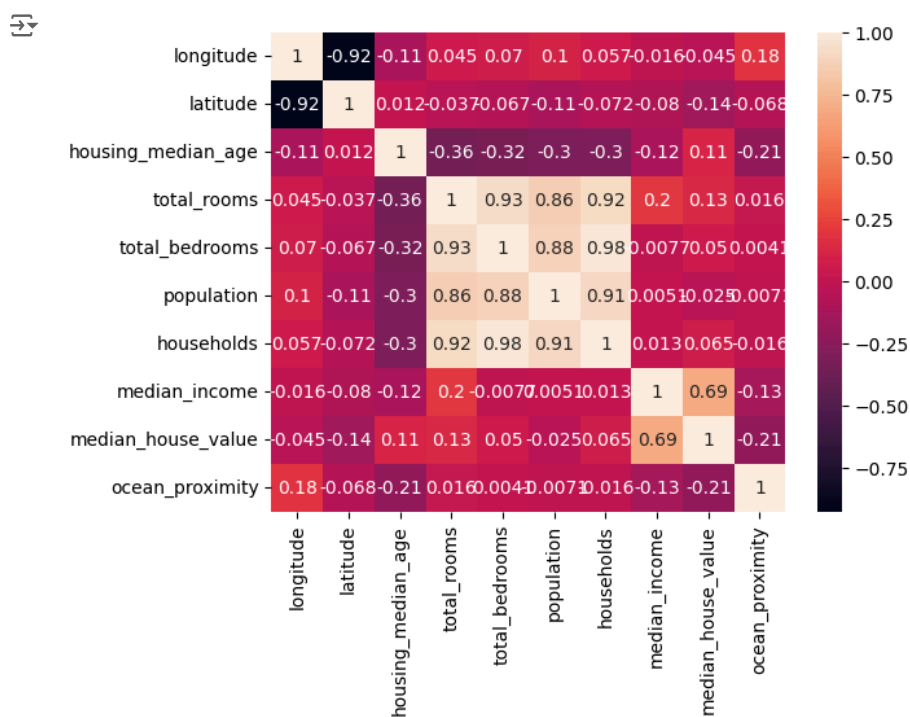


```
longitude      -0.05
latitude       -0.14
housing_median_age  0.11
total_rooms    0.13
total_bedrooms  0.05
population     -0.03
households     0.06
median_income  0.69
median_house_value  1.00
ocean_proximity -0.21
Name: median_house_value, dtype: float64
```

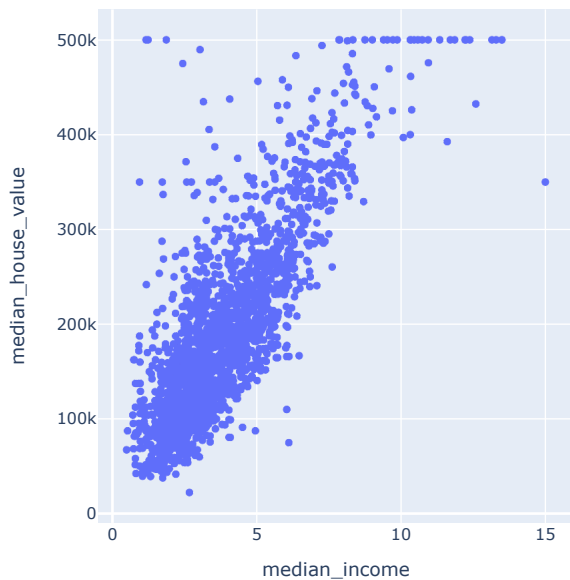
```
1 df.sample(3000).plot(kind='scatter', x='longitude', y='latitude', s=32, alpha=.8)
2 plt.gca().spines[['top', 'right']].set_visible(False)
```



```
1 ax = sns.heatmap(df.corr(), annot = True)
```



```
1 data = df.head(2000)
2 px.scatter(data, x='median_income', y='median_house_value')
```



```
1 df.columns
2 housing_df = df[['longitude', 'latitude', 'housing_median_age', 'total_rooms',
3                 'total_bedrooms', 'population', 'households', 'median_income',
4                 'ocean_proximity', 'median_house_value']]
```

```
1 housing_df.head()
```



| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population |
|---|-----------|----------|--------------------|-------------|----------------|------------|
| 0 | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 |
| 1 | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 |
| 2 | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 |
| 3 | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 |
| 4 | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 |

Next steps:

[Generate code with housing_df](#)
[View recommended plots](#)

```
1 train_pd, test_pd, val_pd = housing_df[:18000], housing_df[18000:19217], housing_df[19215:]
2 len(train_pd), len(test_pd), len(val_pd)
```



```
(18000, 1217, 1218)
```

```
1 X_train, y_train = train_pd.drop('median_house_value', axis=1), train_pd.to_numpy()[:, -1]
2 X_train.head(10)
```



| | longitude | latitude | housing_median_age | total_rooms | total_bedrooms | population |
|---|-----------|----------|--------------------|-------------|----------------|------------|
| 0 | -122.23 | 37.88 | 41.0 | 880.0 | 129.0 | 322.0 |
| 1 | -122.22 | 37.86 | 21.0 | 7099.0 | 1106.0 | 2401.0 |
| 2 | -122.24 | 37.85 | 52.0 | 1467.0 | 190.0 | 496.0 |
| 3 | -122.25 | 37.85 | 52.0 | 1274.0 | 235.0 | 558.0 |
| 4 | -122.25 | 37.85 | 52.0 | 1627.0 | 280.0 | 565.0 |
| 5 | -122.25 | 37.85 | 52.0 | 919.0 | 213.0 | 413.0 |
| 6 | -122.25 | 37.84 | 52.0 | 2535.0 | 489.0 | 1094.0 |
| 7 | -122.25 | 37.84 | 52.0 | 3104.0 | 687.0 | 1157.0 |
| 8 | -122.26 | 37.84 | 42.0 | 2555.0 | 665.0 | 1206.0 |
| 9 | -122.25 | 37.84 | 52.0 | 3549.0 | 707.0 | 1551.0 |

Next steps:

[Generate code with X_train](#)
[View recommended plots](#)

```
1 X_val, y_val = val_pd.to_numpy()[::-1], val_pd.to_numpy()[::-1]
2 X_test, y_test = test_pd.to_numpy()[::-1], test_pd.to_numpy()[::-1]
```

```
1 X_train.shape, y_train.shape, X_test.shape, y_test.shape, X_val.shape, y_val.shape
```

```
((18000, 9), (18000,), (1217, 9), (1217,), (1218, 9), (1218,))
```

```
1 from sklearn.preprocessing import StandardScaler
2 scaler = StandardScaler()
3 X_train = scaler.fit_transform(X_train)
4 X_test = scaler.transform(X_test)
5 X_val = scaler.transform(X_val)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning:
```

X does not have valid feature names, but StandardScaler was fitted with feature names

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning:
```

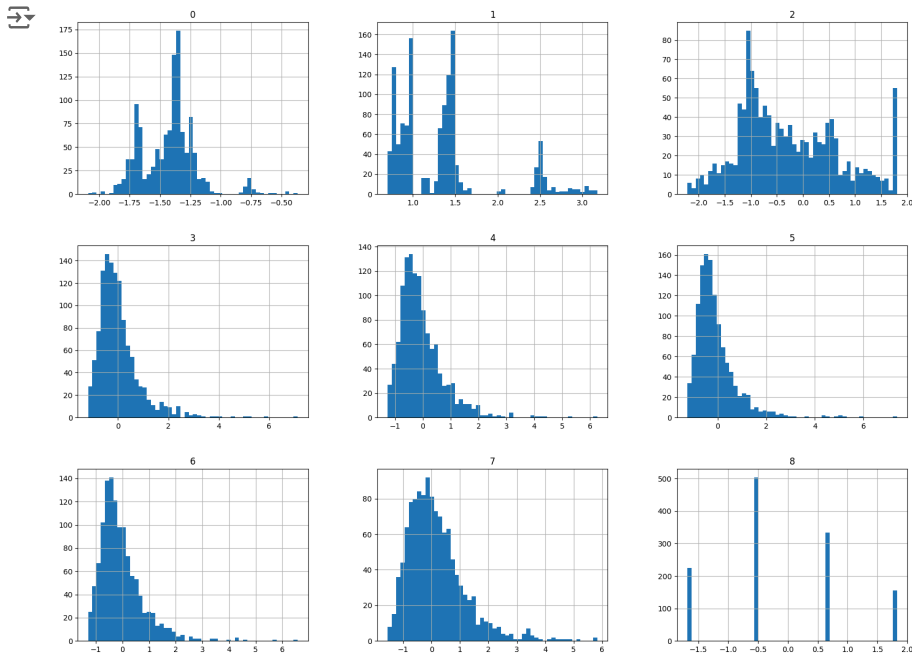
X does not have valid feature names, but StandardScaler was fitted with feature names

```
1 pd.DataFrame(X_train)
```

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | |
|-------|-----------|----------|-----------|-----------|-----------|-----------|-----------|----------|
| 0 | -1.453822 | 1.204250 | 0.935040 | -0.795703 | -0.964371 | -0.970876 | -0.972988 | 2.32936 |
| 1 | -1.448767 | 1.194570 | -0.642170 | 2.018961 | 1.315597 | 0.843015 | 1.637958 | 2.31688 |
| 2 | -1.458876 | 1.189729 | 1.802506 | -0.530032 | -0.822019 | -0.819064 | -0.841409 | 1.76947 |
| 3 | -1.463931 | 1.189729 | 1.802506 | -0.617382 | -0.717005 | -0.764970 | -0.733049 | 0.92303 |
| 4 | -1.463931 | 1.189729 | 1.802506 | -0.457617 | -0.611991 | -0.758863 | -0.629850 | -0.01915 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 17995 | -1.352724 | 0.947718 | -0.799891 | 0.530841 | 1.597968 | 0.432076 | 1.676658 | -0.19260 |
| 17996 | -1.352724 | 0.957398 | -1.036472 | 0.345732 | 1.518624 | 0.038586 | 1.392859 | -0.15553 |
| 17997 | -1.347669 | 0.952558 | -0.642170 | -0.075631 | 0.314465 | 0.044694 | 0.479544 | 0.01104 |
| 17998 | -1.347669 | 0.952558 | -0.326728 | -0.255309 | -0.362291 | -0.141145 | -0.309932 | 0.46045 |
| 17999 | -1.342614 | 0.952558 | -1.036472 | -0.693870 | -0.439302 | -0.816447 | -0.462151 | 0.16955 |

18000 rows x 9 columns

```
1 pd.DataFrame(X_test).hist(bins=50, figsize=(20,15))
2 plt.show()
```



```
1 X_train.shape, X_test.shape, X_val.shape,
```

```
((18000, 9), (1217, 9), (1218, 9))
```

```
1 ##Linear Regression Model
```

```
1 # Preprocessing - scaling the data
```

```
2 scaler = StandardScaler()
```

```
3 X_train_scaled = scaler.fit_transform(X_train)
```

```
4 X_val_scaled = scaler.transform(X_val)
```

```
5
```

```
6 # Train the model
```

```
7 lm = LinearRegression().fit(X_train_scaled, y_train)
```

```
1 y_train_pred = lm.predict(X_train_scaled)
```

```
2 y_val_pred = lm.predict(X_val_scaled)
```

```
1 mse_train = mse(y_train, y_train_pred)
```

```
2 rmse_train = mse(y_train, y_train_pred, squared=False)
```

```
3
```

```
4 # Calculate MSE and RMSE for validation set
```

```
5 mse_val = mse(y_val, y_val_pred)
```

```
6 rmse_val = mse(y_val, y_val_pred, squared=False)
```

```
7
```

```
8 # Calculate R2 score for training set
```

```
9 r2_train = r2_score(y_train, y_train_pred)
```

```
10
```

```
11 # Calculate R2 score for validation set
```

```
12 r2_val = r2_score(y_val, y_val_pred)
```



```
1 print(f'Training MSE: {mse_train}')
2 print(f'Training RMSE: {rmse_train}')
3 print(f'Training R²: {r2_train}')
```

Training MSE: 4985623211.241477

```
1 print(f'Validation MSE: {mse_val}')
2 print(f'Validation RMSE: {rmse_val}')
3 print(f'Validation R²: {r2_val}')
```

Validation MSE: 3021634923.4105325
Validation RMSE: 54969.399882212034
Validation R²: 0.6626316715336671

Double-click (or enter) to edit

```
1 print('Some predictions on the validation set:', y_val_pred[:5])
```

Some predictions on the validation set: [104479.44966118 192574.74579031 146838.49904684 131771.43881713 109040.18165488]

```
1 plt.figure(figsize=(10, 5))
2
3 # Plot actual values
4 plt.scatter(range(len(y_val)), y_val, color='blue', alpha=0.5, label='Actual')
5
6 # Plot predicted values
7 plt.scatter(range(len(y_val)), y_val_pred, color='red', alpha=0.5, label='Predicted')
8
9 plt.xlabel('House Price')
10 plt.ylabel('Value')
11 plt.title('Actual vs Predicted Values')
12 plt.legend()
13 plt.show()
14
```

