```
import numpy as np

import pandas as pd

df = pd.read_csv('House_Pricing.csv')

df
```

| | ID | Date House was Sold | Sale Price | No of Bedrooms | No of Bathrooms | Flat Area (in Sqft) | Lot Area (in Sqft) | No of Floors | Waterfront View | No of Times Visited | ... | Overall Grade | Area of the House from Basement (in Sqft) | Basement Area (in Sqft) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 7129300520 | 14 October 2017 | 221900.0 | 3 | 1.00 | 1180.0 | 5650.0 | 1.0 | No | NaN | ... | 7 | 1180.0 | 0 |
| **1** | 6414100192 | 14 December 2017 | 538000.0 | 3 | 2.25 | 2570.0 | 7242.0 | 2.0 | No | NaN | ... | 7 | 2170.0 | 400 |
| **2** | 5631500400 | 15 February 2016 | 180000.0 | 2 | 1.00 | 770.0 | 10000.0 | 1.0 | No | NaN | ... | 6 | 770.0 | 0 |
| **3** | 2487200875 | 14 December 2017 | 604000.0 | 4 | 3.00 | 1960.0 | 5000.0 | 1.0 | No | NaN | ... | 7 | 1050.0 | 910 |
| **4** | 1954400510 | 15 February 2016 | 510000.0 | 3 | 2.00 | 1680.0 | 8080.0 | 1.0 | No | NaN | ... | 8 | 1680.0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **21608** | 263000018 | 14 May 2017 | 360000.0 | 3 | 2.50 | 1530.0 | 1131.0 | 3.0 | No | NaN | ... | 8 | 1530.0 | 0 |
| **21609** | 6600060120 | 15 February 2016 | 400000.0 | 4 | 2.50 | 2310.0 | 5813.0 | 2.0 | No | NaN | ... | 8 | 2310.0 | 0 |
| **21610** | 1523300141 | 14 June 2017 | 402101.0 | 2 | 0.75 | 1020.0 | 1350.0 | 2.0 | No | NaN | ... | 7 | 1020.0 | 0 |
| **21611** | 291310100 | 15 January 2016 | 400000.0 | 3 | 2.50 | 1600.0 | 2388.0 | 2.0 | No | NaN | ... | 8 | 1600.0 | 0 |
| **21612** | 1523300157 | 14 October 2017 | 325000.0 | 2 | 0.75 | 1020.0 | 1076.0 | 2.0 | No | NaN | ... | 7 | 1020.0 | 0 |

21613 rows × 21 columns

```
duplicate_rows = df[df.duplicated()]       #Check for duplicate rows in the dataset, if any, and remove them
print("Duplicate Rows:")
print(duplicate_rows)
df_cleaned = df.drop_duplicates()
```

```
Duplicate Rows:
Empty DataFrame
Columns: [ID, Date House was Sold, Sale Price, No of Bedrooms, No of Bathrooms, Flat Area (in Sqft), Lot Area (in Sqft), No of Floors, W
Index: []

[0 rows x 21 columns]
```

```
df = df.T.drop_duplicates().T          #Identify and drop duplicate columns, if any, based on their values

import pandas as pd
import numpy as np
```

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

```
df
```

| | ID | Date House was Sold | Sale Price | No of Bedrooms | No of Bathrooms | Flat Area (in Sqft) | Lot Area (in Sqft) | No of Floors | Waterfront View | No of Times Visited | ... | Overall Grade | Area of the House from Basement (in Sqft) | Basement Area (in Sqft) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 7129300520 | 14 October 2017 | 221900.0 | 3 | 1.00 | 1180.0 | 5650.0 | 1.0 | No | NaN | ... | 7 | 1180.0 | 0 |
| **1** | 6414100192 | 14 December 2017 | 538000.0 | 3 | 2.25 | 2570.0 | 7242.0 | 2.0 | No | NaN | ... | 7 | 2170.0 | 400 |
| **2** | 5631500400 | 15 February 2016 | 180000.0 | 2 | 1.00 | 770.0 | 10000.0 | 1.0 | No | NaN | ... | 6 | 770.0 | 0 |
| **3** | 2487200875 | 14 December 2017 | 604000.0 | 4 | 3.00 | 1960.0 | 5000.0 | 1.0 | No | NaN | ... | 7 | 1050.0 | 910 |
| **4** | 1954400510 | 15 February 2016 | 510000.0 | 3 | 2.00 | 1680.0 | 8080.0 | 1.0 | No | NaN | ... | 8 | 1680.0 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **21608** | 263000018 | 14 May 2017 | 360000.0 | 3 | 2.50 | 1530.0 | 1131.0 | 3.0 | No | NaN | ... | 8 | 1530.0 | 0 |
| **21609** | 6600060120 | 15 February 2016 | 400000.0 | 4 | 2.50 | 2310.0 | 5813.0 | 2.0 | No | NaN | ... | 8 | 2310.0 | 0 |
| **21610** | 1523300141 | 14 June 2017 | 402101.0 | 2 | 0.75 | 1020.0 | 1350.0 | 2.0 | No | NaN | ... | 7 | 1020.0 | 0 |
| **21611** | 291310100 | 15 January 2016 | 400000.0 | 3 | 2.50 | 1600.0 | 2388.0 | 2.0 | No | NaN | ... | 8 | 1600.0 | 0 |
| **21612** | 1523300157 | 14 October 2017 | 325000.0 | 2 | 0.75 | 1020.0 | 1076.0 | 2.0 | No | NaN | ... | 7 | 1020.0 | 0 |

21613 rows × 21 columns

```
for column in df.columns:                                   #Handling Missing Values
    if df[column].isnull().sum() > 0:
        if df[column].dtype in ['float64', 'int64']:
            df[column] = df[column].fillna(df[column].mean())
        else:
            df[column] = df[column].fillna(df[column].mode()[0])
```

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler            #Scaling Numerical Variables
numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns
features_to_scale = numerical_cols.drop('Sale Price')
scaler = MinMaxScaler()
df[features_to_scale] = scaler.fit_transform(df[features_to_scale])
df.head()
```

| | ID | Date House was Sold | Sale Price | No of Bedrooms | No of Bathrooms | Flat Area (in Sqft) | Lot Area (in Sqft) | No of Floors | Waterfront View | No of Times Visited | ... | Overall Grade | Area of the House from Basement (in Sqft) | Basement Area (in Sqft) | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.720103 | 14 October 2017 | 221900.0 | 0.090909 | 0.12500 | 0.067170 | 0.003108 | 0.0 | No | Twice | ... | 0.666667 | 0.097588 | 0.000000 | 0 |
| **1** | 0.647853 | 14 December 2017 | 538000.0 | 0.090909 | 0.28125 | 0.172075 | 0.004072 | 0.4 | No | Twice | ... | 0.666667 | 0.206140 | 0.082988 | 0 |
| **2** | 0.568795 | 15 February 2016 | 180000.0 | 0.060606 | 0.12500 | 0.036226 | 0.005743 | 0.0 | No | Twice | ... | 0.555556 | 0.052632 | 0.000000 | 0 |
| **3** | 0.251157 | 14 December 2017 | 604000.0 | 0.121212 | 0.37500 | 0.126038 | 0.002714 | 0.0 | No | Twice | ... | 0.666667 | 0.083333 | 0.188797 | 0 |
| **4** | 0.197333 | 15 February 2016 | 510000.0 | 0.090909 | 0.25000 | 0.104906 | 0.004579 | 0.0 | No | Twice | ... | 0.777778 | 0.152412 | 0.000000 | 0 |

5 rows × 21 columns

```
categorical_cols = df.select_dtypes(include='object').columns            #Encoding Categorical Variables
df = pd.get_dummies(df, columns=categorical_cols, drop_first=True)
df.head()
```

| | ID | Sale Price | No of Bedrooms | No of Bathrooms | Flat Area (in Sqft) | Lot Area (in Sqft) | No of Floors | Overall Grade | Area of the House from Basement (in Sqft) | Basement Area (in Sqft) | ... | Date House was Sold_15 March 2016 | Date House was Sold_15 May 2016 | Waterfront View_Yes | No Vis |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.720103 | 221900.0 | 0.090909 | 0.12500 | 0.067170 | 0.003108 | 0.0 | 0.666667 | 0.097588 | 0.000000 | ... | False | False | False | |
| **1** | 0.647853 | 538000.0 | 0.090909 | 0.28125 | 0.172075 | 0.004072 | 0.4 | 0.666667 | 0.206140 | 0.082988 | ... | False | False | False | |
| **2** | 0.568795 | 180000.0 | 0.060606 | 0.12500 | 0.036226 | 0.005743 | 0.0 | 0.555556 | 0.052632 | 0.000000 | ... | False | False | False | |
| **3** | 0.251157 | 604000.0 | 0.121212 | 0.37500 | 0.126038 | 0.002714 | 0.0 | 0.666667 | 0.083333 | 0.188797 | ... | False | False | False | |
| **4** | 0.197333 | 510000.0 | 0.090909 | 0.25000 | 0.104906 | 0.004579 | 0.0 | 0.777778 | 0.152412 | 0.000000 | ... | False | False | False | |

5 rows × 37 columns

```
numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns   #outlier removal
features_to_check = numerical_cols.drop('Sale Price')
for col in features_to_check:
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    df = df[(df[col] >= lower_bound) & (df[col] <= upper_bound)]
print("Shape after outlier removal:", df.shape)
df.head()
```

```
Shape after outlier removal: (15103, 37)
```

| | ID | Sale Price | No of Bedrooms | No of Bathrooms | Flat Area (in Sqft) | Lot Area (in Sqft) | No of Floors | Overall Grade | Area of the House from Basement (in Sqft) | Basement Area (in Sqft) | ... | Date House was Sold_15 March 2016 | Date House was Sold_15 May 2016 | Waterfront View_Yes | No Vis |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0.720103 | 221900.0 | 0.090909 | 0.12500 | 0.067170 | 0.003108 | 0.0 | 0.666667 | 0.097588 | 0.000000 | ... | False | False | False | |
| **2** | 0.568795 | 180000.0 | 0.060606 | 0.12500 | 0.036226 | 0.005743 | 0.0 | 0.555556 | 0.052632 | 0.000000 | ... | False | False | False | |
| **3** | 0.251157 | 604000.0 | 0.121212 | 0.37500 | 0.126038 | 0.002714 | 0.0 | 0.666667 | 0.083333 | 0.188797 | ... | False | False | False | |
| **4** | 0.197333 | 510000.0 | 0.090909 | 0.25000 | 0.104906 | 0.004579 | 0.0 | 0.777778 | 0.152412 | 0.000000 | ... | False | False | False | |
| **6** | 0.133387 | 257500.0 | 0.090909 | 0.28125 | 0.107547 | 0.003816 | 0.4 | 0.666667 | 0.156250 | 0.000000 | ... | False | False | False | |

5 rows × 37 columns

```
from sklearn.model_selection import train_test_split      #train test split
X = df.drop('Sale Price', axis=1)
y = df['Sale Price']
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
X_train shape: (12082, 36)
X_test shape: (3021, 36)
y_train shape: (12082,)
y_test shape: (3021,)
```