In [2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import time
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVC
from sklearn import svm
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import accuracy_score

%matplotlib inline
```

In [4]:
```
df=pd.read_csv("C:\\Users\\Anjali Kumari\\Downloads\\House price\\data.csv")
```

In [5]:
```
df.head()
```

Out[5]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	conditio
0	2014-05-02 00:00:00	313000.0	3.0	1.50	1340	7912	1.5	0	0	
1	2014-05-02 00:00:00	2384000.0	5.0	2.50	3650	9050	2.0	0	4	
2	2014-05-02 00:00:00	342000.0	3.0	2.00	1930	11947	1.0	0	0	
3	2014-05-02 00:00:00	420000.0	3.0	2.25	2000	8030	1.0	0	0	
4	2014-05-02 00:00:00	550000.0	4.0	2.50	1940	10500	1.0	0	0	

In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   date           4600 non-null   object 
 1   price          4600 non-null   float64
 2   bedrooms       4600 non-null   float64
 3   bathrooms      4600 non-null   float64
 4   sqft_living    4600 non-null   int64  
 5   sqft_lot       4600 non-null   int64  
 6   floors         4600 non-null   float64
 7   waterfront     4600 non-null   int64  
 8   view           4600 non-null   int64  
 9   condition      4600 non-null   int64  
 10  sqft_above     4600 non-null   int64  
 11  sqft_basement  4600 non-null   int64  
 12  yr_built       4600 non-null   int64  
 13  yr_renovated   4600 non-null   int64  
 14  street         4600 non-null   object 
 15  city           4600 non-null   object 
 16  statezip       4600 non-null   object 
 17  country        4600 non-null   object 
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
```

In [7]:
```
df.values
```

Out[7]:
```
array([['2014-05-02 00:00:00', 313000.0, 3.0, ..., 'Shoreline',
        'WA 98133', 'USA'],
       ['2014-05-02 00:00:00', 2384000.0, 5.0, ..., 'Seattle',
        'WA 98119', 'USA'],
       ['2014-05-02 00:00:00', 342000.0, 3.0, ..., 'Kent', 'WA 98042',
        'USA'],
       ...,
       ['2014-07-09 00:00:00', 416904.166667, 3.0, ..., 'Renton',
        'WA 98059', 'USA'],
       ['2014-07-10 00:00:00', 203400.0, 4.0, ..., 'Seattle', 'WA 98178',
        'USA'],
       ['2014-07-10 00:00:00', 220600.0, 3.0, ..., 'Covington',
        'WA 98042', 'USA']], dtype=object)
```

In [8]:
```
df.columns
```

Out[8]:
```
Index(['date', 'price', 'bedrooms', 'bathrooms', 'sqft_living', 'sqft_lot',
       'floors', 'waterfront', 'view', 'condition', 'sqft_above',
       'sqft_basement', 'yr_built', 'yr_renovated', 'street', 'city',
       'statezip', 'country'],
      dtype='object')
```

In [9]:
```
df.describe
```

Out[9]:
```
<bound method NDFrame.describe of                  date         price  bedrooms
bathrooms  sqft_living  \
0     2014-05-02 00:00:00  3.130000e+05       3.0       1.50         1340
1     2014-05-02 00:00:00  2.384000e+06       5.0       2.50         3650
2     2014-05-02 00:00:00  3.420000e+05       3.0       2.00         1930
3     2014-05-02 00:00:00  4.200000e+05       3.0       2.25         2000
4     2014-05-02 00:00:00  5.500000e+05       4.0       2.50         1940
...                   ...           ...       ...        ...          ...
4595  2014-07-09 00:00:00  3.081667e+05       3.0       1.75         1510
4596  2014-07-09 00:00:00  5.343333e+05       3.0       2.50         1460
4597  2014-07-09 00:00:00  4.169042e+05       3.0       2.50         3010
4598  2014-07-10 00:00:00  2.034000e+05       4.0       2.00         2090
4599  2014-07-10 00:00:00  2.206000e+05       3.0       2.50         1490

      sqft_lot  floors  waterfront  view  condition  sqft_above  \
0         7912     1.5           0     0          3        1340
1         9050     2.0           0     4          5        3370
2        11947     1.0           0     0          4        1930
3         8030     1.0           0     0          4        1000
4        10500     1.0           0     0          4        1140
...        ...     ...         ...   ...        ...         ...
4595      6360     1.0           0     0          4        1510
4596      7573     2.0           0     0          3        1460
4597      7014     2.0           0     0          3        3010
4598      6630     1.0           0     0          3        1070
4599      8102     2.0           0     0          4        1490

      sqft_basement  yr_built  yr_renovated                    street  \
0                 0      1955          2005      18810 Densmore Ave N
1               280      1921             0           709 W Blaine St
2                 0      1966             0  26206-26214 143rd Ave SE
3              1000      1963             0           857 170th Pl NE
4               800      1976          1992         9105 170th Ave NE
...             ...       ...           ...                       ...
4595              0      1954          1979            501 N 143rd St
4596              0      1983          2009          14855 SE 10th Pl
4597              0      2009             0          759 Ilwaco Pl NE
4598           1020      1974             0         5148 S Creston St
4599              0      1990             0         18717 SE 258th St

          city  statezip country
0     Shoreline  WA 98133     USA
1       Seattle  WA 98119     USA
2          Kent  WA 98042     USA
3      Bellevue  WA 98008     USA
4       Redmond  WA 98052     USA
...         ...       ...     ...
4595    Seattle  WA 98133     USA
4596   Bellevue  WA 98007     USA
4597     Renton  WA 98059     USA
4598    Seattle  WA 98178     USA
4599  Covington  WA 98042     USA

[4600 rows x 18 columns]>
```

In [10]:
```
df["country"].values
```

Out[10]:
```
array(['USA', 'USA', 'USA', ..., 'USA', 'USA', 'USA'], dtype=object)
```

In [11]:
```
df["city"].drop_duplicates().values
```

Out[11]:
```
array(['Shoreline', 'Seattle', 'Kent', 'Bellevue', 'Redmond',
       'Maple Valley', 'North Bend', 'Lake Forest Park', 'Sammamish',
       'Auburn', 'Des Moines', 'Bothell', 'Federal Way', 'Kirkland',
       'Issaquah', 'Woodinville', 'Normandy Park', 'Fall City', 'Renton',
       'Carnation', 'Snoqualmie', 'Duvall', 'Burien', 'Covington',
       'Inglewood-Finn Hill', 'Kenmore', 'Newcastle', 'Mercer Island',
       'Black Diamond', 'Ravensdale', 'Clyde Hill', 'Algona', 'Skykomish',
       'Tukwila', 'Vashon', 'Yarrow Point', 'SeaTac', 'Medina',
       'Enumclaw', 'Snoqualmie Pass', 'Pacific', 'Beaux Arts Village',
       'Preston', 'Milton'], dtype=object)
```

In [13]:
```
df["country"].drop_duplicates().values
```

Out[13]:
```
array(['USA'], dtype=object)
```

In [14]:
```
df["street"].drop_duplicates().values
```

Out[14]:
```
array(['18810 Densmore Ave N', '709 W Blaine St',
       '26206-26214 143rd Ave SE', ..., '759 Ilwaco Pl NE',
       '5148 S Creston St', '18717 SE 258th St'], dtype=object)
```

In [15]:
```
df[['yr_built']].drop_duplicates().values.reshape(1,-1)
```

Out[15]:
```
array([[1955, 1921, 1966, 1963, 1976, 1938, 1989, 1985, 1945, 1948, 1909,
        1980, 1939, 1965, 1956, 1997, 1987, 1983, 1923, 1954, 2005, 1991,
        1959, 1920, 1979, 1951, 1967, 2004, 2014, 1995, 1944, 1992, 1957,
        1978, 1974, 1950, 1990, 1949, 1968, 1977, 1996, 2001, 2000, 1929,
        2006, 2008, 2009, 2012, 2013, 2007, 1986, 1973, 1988, 1900, 1928,
        1982, 1960, 2003, 1998, 1942, 1908, 1958, 1971, 1975, 1961, 1924,
        1902, 1962, 1953, 1999, 1994, 1930, 1984, 2010, 1912, 1947, 1916,
        1940, 1970, 1952, 1964, 1926, 1905, 1969, 1903, 1943, 1993, 1946,
        1910, 1904, 1981, 1906, 1922, 2011, 2002, 1917, 1913, 1914, 1937,
        1925, 1932, 1918, 1972, 1941, 1919, 1936, 1911, 1927, 1931, 1901,
        1907, 1915, 1935, 1933, 1934]], dtype=int64)
```

In [16]:
```
df['yr_renovated'].drop_duplicates().values.reshape(1,-1)
```

Out[16]:
```
array([[2005,    0, 1992, 1994, 2010, 1988, 2009, 1969, 2000, 1979, 1989,
        2014, 1999, 2003, 1983, 1997, 1912, 1923, 1954, 2011, 2001, 2013,
        2006, 1972, 1985, 1998, 2004, 1958, 2008, 1970, 1982, 1986, 1996,
        2002, 1971, 1990, 1956, 1945, 1984, 2012, 1993, 2007, 1981, 1974,
        1963, 1968, 1995, 1934, 1953, 1966, 1955, 1987, 1960, 1978, 1980,
        1948, 1991, 1913, 1977, 1975]], dtype=int64)
```

In [17]:
```
df['condition'].value_counts(normalize=True)
```

Out[17]:
```
3    0.625000
4    0.272174
5    0.094565
2    0.006957
1    0.001304
Name: condition, dtype: float64
```

In [18]:
```
df['bathrooms'].value_counts(normalize=True)
```

Out[18]:
```
2.50    0.258478
1.00    0.161522
1.75    0.136739
2.00    0.092826
2.25    0.091087
1.50    0.063261
2.75    0.060000
3.00    0.036304
3.50    0.035217
3.25    0.029565
3.75    0.008043
4.50    0.006304
4.25    0.005000
4.00    0.005000
0.75    0.003696
4.75    0.001522
5.00    0.001304
5.25    0.000870
5.50    0.000870
1.25    0.000652
6.25    0.000435
0.00    0.000435
8.00    0.000217
5.75    0.000217
6.50    0.000217
6.75    0.000217
Name: bathrooms, dtype: float64
```

In [20]:
```
df['bedrooms'].value_counts(normalize=True)
```

Out[20]:
```
3.0    0.441739
4.0    0.332826
2.0    0.123043
5.0    0.076739
6.0    0.013261
1.0    0.008261
7.0    0.003043
8.0    0.000435
0.0    0.000435
9.0    0.000217
Name: bedrooms, dtype: float64
```

In [21]:
```
df['floors'].value_counts(normalize=True)
```

Out[21]:
```
1.0    0.472609
2.0    0.393696
1.5    0.096522
3.0    0.027826
2.5    0.008913
3.5    0.000435
Name: floors, dtype: float64
```

In [22]:
```
df['waterfront'].value_counts(normalize=True)
```

Out[22]:
```
0    0.992826
1    0.007174
Name: waterfront, dtype: float64
```

In [23]:
```
df['view'].value_counts(normalize=True)
```

Out[23]:
```
0    0.900000
2    0.044565
3    0.025217
4    0.015217
1    0.015000
Name: view, dtype: float64
```

In [24]:
```
df['sqft_basement'].value_counts(normalize=True)
```

Out[24]:
```
0       0.596739
500     0.011522
600     0.009783
800     0.009348
900     0.008913
          ...
2300    0.000217
265     0.000217
1610    0.000217
862     0.000217
1640    0.000217
Name: sqft_basement, Length: 207, dtype: float64
```

In [25]:
```
df[['yr_built','yr_renovated']].min()
```

Out[25]:
```
yr_built        1900
yr_renovated       0
dtype: int64
```

In [45]:
```
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Load the California Housing dataset
california = fetch_california_housing()
X = california.data
y = california.target

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=

# Train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("Mean Squared Error:", mse)
print("R-squared:", r2)

# Optionally, print coefficients and intercept
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

```
Mean Squared Error: 0.5558915986952422
R-squared: 0.5757877060324524
Coefficients: [ 4.48674910e-01  9.72425752e-03 -1.23323343e-01  7.83144907e-01
 -2.02962058e-06 -3.52631849e-03 -4.19792487e-01 -4.33708065e-01]
Intercept: -37.02327770606369
```

In [26]:
```
df[(dataset['yr_built']>2000) & (dataset['view']>0)].shape
```

Out[26]:
```
(61, 18)
```

In [27]:
```
df[(dataset['yr_built']<2000) & (dataset['view']>1)].shape
```

Out[27]:
```
(328, 18)
```

In [28]:
```
df[(dataset['yr_built']<2000) & (dataset['view']>1)].shape
```

Out[28]:
```
(328, 18)
```

In [29]:
```
df[(dataset['yr_built']>2000) & (dataset['floors']>1) & (dataset['waterfront']>0) & (d
```

Out[29]:

	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_a

In [30]:
```
df[(dataset['yr_built']<1999) & (dataset['floors']>1) & (dataset['waterfront']>0) & (d
```

Out[30]:
```
(9, 18)
```

In [31]:
```
X=dataset[['bedrooms','bathrooms','sqft_living','sqft_lot','floors','waterfront','view
           'condition','sqft_above','sqft_basement','yr_built','yr_renovated']]
Y=dataset[['price']]
```

In [32]:
```
X_train,X_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=1)
```

In [33]:
```
model=LinearRegression()
```

In [34]:
```
model.fit(X_train,y_train)
```

Out[34]:
```
LinearRegression()
```

In [37]:
```
pip install scikit-learn
```

```
Requirement already satisfied: scikit-learn in c:\users\anjali kumari\anaconda3\jupyt
er\lib\site-packages (1.0.2)
Requirement already satisfied: scipy>=1.1.0 in c:\users\anjali kumari\anaconda3\jupyt
er\lib\site-packages (from scikit-learn) (1.9.1)
Requirement already satisfied: joblib>=0.11 in c:\users\anjali kumari\anaconda3\jupyt
er\lib\site-packages (from scikit-learn) (1.1.0)
Requirement already satisfied: numpy>=1.14.6 in c:\users\anjali kumari\anaconda3\jupy
ter\lib\site-packages (from scikit-learn) (1.24.4)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\anjali kumari\anacond
a3\jupyter\lib\site-packages (from scikit-learn) (2.2.0)
Note: you may need to restart the kernel to use updated packages.
```

In [38]:
```
from sklearn.metrics import mean_squared_error
```

```
# Assuming y_pred and y_test are already defined
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

```
Mean Squared Error: 65832396764.362465
```

In [39]:
```
model.coef_
```

Out[39]:
```
array([[-5.35063642e+04,  5.93439260e+04,  1.61776161e+02,
        -6.56937457e-01,  3.20511164e+04,  5.88167665e+05,
         3.54197553e+04,  3.57541024e+04,  9.74014157e+01,
         6.43747457e+01, -2.19652274e+03,  8.22436344e+00]])
```

In [40]:
```
model.intercept_
```

Out[40]:
```
array([4213559.49398007])
```

In [41]:
```
model.score(X_train,y_train)
```

Out[41]:
```
0.19597756216049567
```

In [42]:
```
model.score(X_test,y_test)
```

Out[42]:
```
0.4601795305842167
```

In [43]:
```
dataset2=dataset.drop(['date','city','street','statezip','country','price'],axis=1)
dataset2.corr()
```

Out[43]:

	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	conditi
bedrooms	1.000000	0.545920	0.594884	0.068819	0.177895	-0.003483	0.111028	0.0250
bathrooms	0.545920	1.000000	0.761154	0.107837	0.486428	0.076232	0.211960	-0.1199
sqft_living	0.594884	0.761154	1.000000	0.210538	0.344850	0.117616	0.311009	-0.0628
sqft_lot	0.068819	0.107837	0.210538	1.000000	0.003750	0.017241	0.073907	0.0005
floors	0.177895	0.486428	0.344850	0.003750	1.000000	0.022024	0.031211	-0.2750
waterfront	-0.003483	0.076232	0.117616	0.017241	0.022024	1.000000	0.360935	0.0003
view	0.111028	0.211960	0.311009	0.073907	0.031211	0.360935	1.000000	0.0630
condition	0.025080	-0.119994	-0.062826	0.000558	-0.275013	0.000352	0.063077	1.0000
sqft_above	0.484705	0.689918	0.876443	0.216455	0.522814	0.078911	0.174327	-0.1781
sqft_basement	0.334165	0.298020	0.447206	0.034842	-0.255510	0.097501	0.321602	0.2006
yr_built	0.142461	0.463498	0.287775	0.050706	0.467481	-0.023563	-0.064465	-0.3996
yr_renovated	-0.061082	-0.215886	-0.122817	-0.022730	-0.233996	0.008625	0.022967	-0.1868

In [ ]:

In [ ]: