ex11

September 10, 2024

TASK 1:

[351]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import classification_report, confusion_matrix
```

[352]:
```
data = pd.read_csv(r'datasets\nba_logreg.csv')

print(data.isnull().sum())
data.dropna(inplace=True)
```

```
Name            0
GP              0
MIN             0
PTS             0
FGM             0
FGA             0
FG%             0
3P Made         0
3PA             0
3P%            11
FTM             0
FTA             0
FT%             0
OREB            0
DREB            0
REB             0
AST             0
STL             0
BLK             0
TOV             0
TARGET_5Yrs     0
dtype: int64
```

[353]:
```
print(data.isnull().sum())
```

```
X = data.drop(['Name', 'TARGET_5Yrs'], axis=1)
y = data['TARGET_5Yrs']
```

```
Name           0
GP             0
MIN            0
PTS            0
FGM            0
FGA            0
FG%            0
3P Made        0
3PA            0
3P%            0
FTM            0
FTA            0
FT%            0
OREB           0
DREB           0
REB            0
AST            0
STL            0
BLK            0
TOV            0
TARGET_5Yrs    0
dtype: int64
```

[354]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[355]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[356]:
```
model = SVC(kernel='rbf')
model.fit(X_train, y_train)
```

[356]:
```
SVC()
```

[357]:
```
y_pred = model.predict(X_test)

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[ 55  35]
 [ 31 145]]
              precision    recall  f1-score   support

         0.0       0.64      0.61      0.63        90
```

```
           1.0       0.81      0.82      0.81       176

      accuracy                           0.75       266
     macro avg       0.72      0.72      0.72       266
  weighted avg       0.75      0.75      0.75       266
```

TASK 2:

[358]:
```
data2 = pd.read_csv(r'datasets\house_price_bd.csv')
```

[359]:
```
data2["Price_in_taka"] = data2["Price_in_taka"].str.replace(' ', '').str.
↪replace(',', '').astype("int64")
```

[360]:
```
import numpy as np

data2['Log_Price'] = np.log1p(data2['Price_in_taka'])
```

[361]:
```
data2.head()
```

[361]:

	Title	Bedrooms	Bathrooms	\
0	We Are Offering You A Very Spacious 1960 Sq Ft…	3.0	4.0	
1	Valuable 1705 Square Feet Apartment Is Ready T…	3.0	3.0	
2	1370 square feet apartment is ready to sale in…	3.0	3.0	
3	2125 Square Feet Apartment For Sale In Bashund…	3.0	3.0	
4	Buy This 2687 Square Feet Flat In The Nice Are…	3.0	3.0	

	Floor_no	Occupancy_status	Floor_area	City	Price_in_taka	\
0	3	vacant	1960.0	dhaka	39000000	
1	1	vacant	1705.0	dhaka	16900000	
2	6	vacant	1370.0	dhaka	12500000	
3	4	vacant	2125.0	dhaka	20000000	
4	4	vacant	2687.0	dhaka	47500000	

	Location	Log_Price
0	Gulshan 1, Gulshan	17.479072
1	Lake Circus Road, Kalabagan	16.642824
2	Shukrabad, Dhanmondi	16.341239
3	Block L, Bashundhara R-A	16.811243
4	Road No 25, Banani	17.676240

[362]:
```
data2 = data2.drop_duplicates(keep='last')
print(data2.isnull().sum())
```

```
Title                0
Bedrooms           831
Bathrooms          831
Floor_no           575
Occupancy_status    89
```

```
Floor_area           89
City                  0
Price_in_taka         0
Location              6
Log_Price             0
dtype: int64
```

[363]:
```
data2.dropna(inplace=True)
print(data2.isnull().sum())
```

```
Title              0
Bedrooms           0
Bathrooms          0
Floor_no           0
Occupancy_status   0
Floor_area         0
City               0
Price_in_taka      0
Location           0
Log_Price          0
dtype: int64
```

[364]:
```
data2.drop('Title', axis=1, inplace=True)
```

[365]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
data2['Location'] = le.fit_transform(data2['Location'])
data2['Occupancy_status'] = le.fit_transform(data2['Occupancy_status'])
data2['City'] = le.fit_transform(data2['City'])
data2['Floor_no'] = le.fit_transform(data2['Floor_no'])
```

Not all locations are unique so we can use the location feature for regression.

[366]:
```
data2['Location'].unique().shape
```

[366]: (419,)

[367]:
```
X = data2.drop(['Price_in_taka', 'Log_Price'], axis=1)
y = data2['Log_Price']
```

[368]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[369]:
```
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

[370]:
```
from sklearn.svm import SVR
```

```
model = SVR(C=10, epsilon=0.1, kernel='rbf')
model.fit(X_train, y_train)
```

[370]: SVR(C=10)

[371]:
```
from sklearn.metrics import mean_squared_error

y_pred = model.predict(X_test)

print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
print("R^2 Score:", model.score(X_test, y_test))
```

```
Mean Squared Error: 0.06290096367049423
R^2 Score: 0.7812516724673056
```