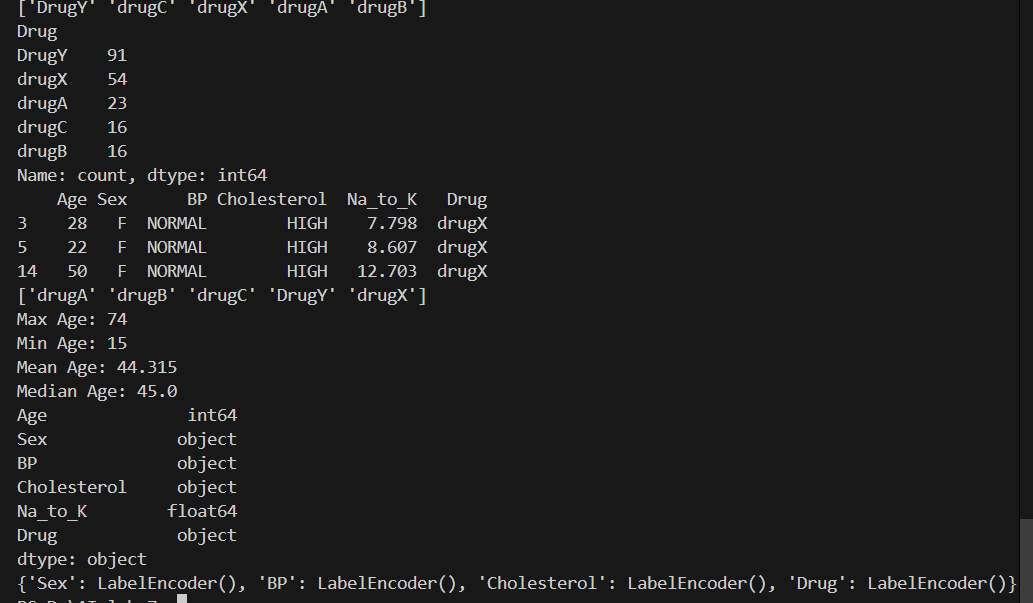
Name : Abdul Rehman Section :B

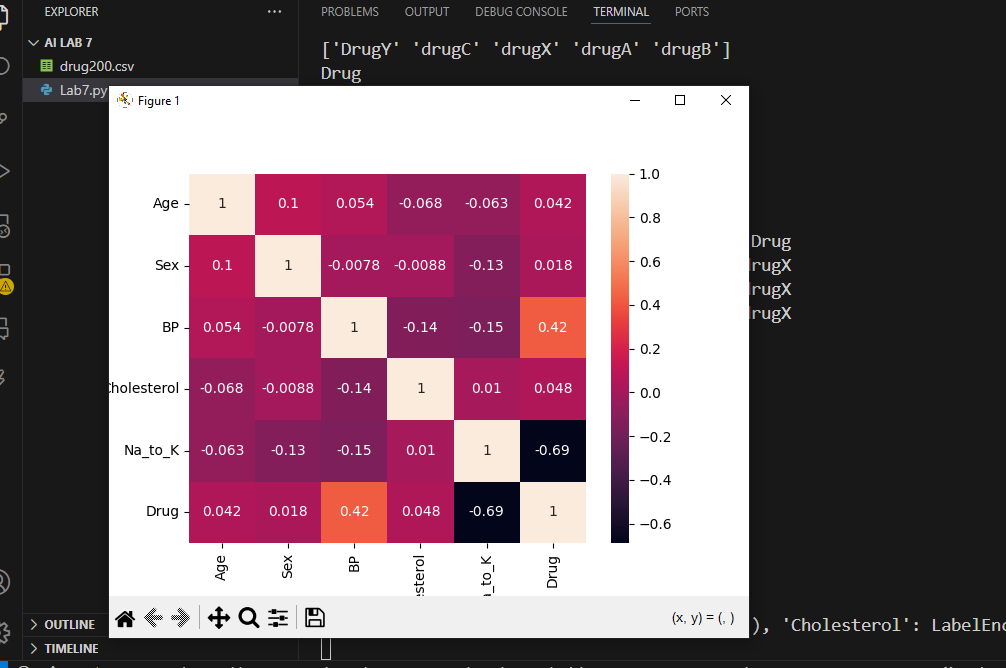
Course : AI&ES Roll No : CT-22052

=====================================================================

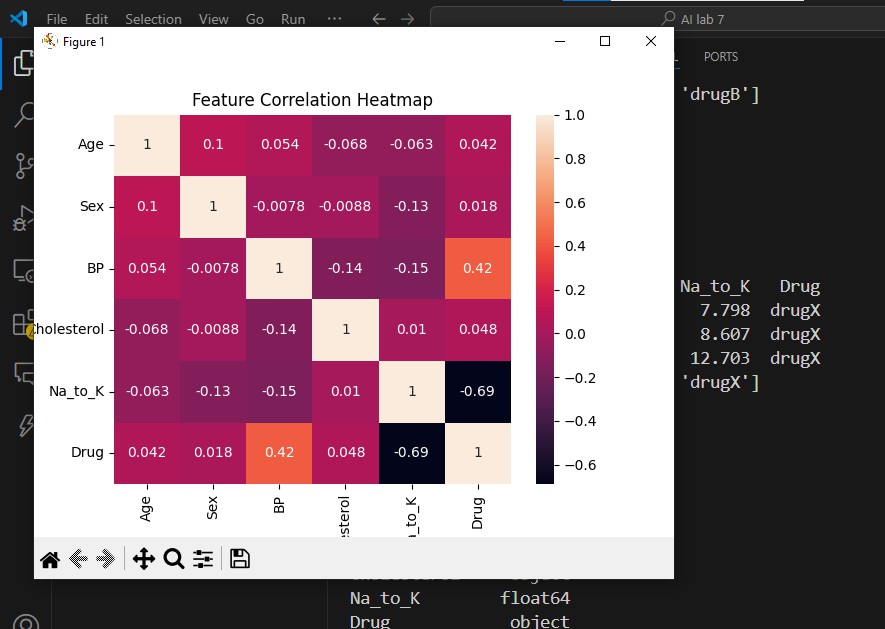
**Example 1**

****

**Example 2**

****

**Example 3**

****

* **What is the difference between hard margin and soft margin SVM?**

Hard Margin: Assumes data is perfectly linearly separable; no misclassifications allowed. Soft Margin: Allows some misclassification to handle noise and improve generalization.

* **How does SVM handle imbalanced datasets?**

SVM can use class weights (e.g., class\_weight='balanced') to give more importance to minority classes during training.

* **What is the kernel trick in SVM?**

The kernel trick allows SVM to operate in a high-dimensional space without explicitly transforming the data, enabling it to solve non-linear problems efficiently.

* **Write a SVM code to predict whether a person has diabetes or not.**

from sklearn.datasets import load\_diabetes

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.preprocessing import StandardScaler

X, y = load\_diabetes(return\_X\_y=True)

y = (y > y.mean()).astype(int) # Binary target

X = StandardScaler().fit\_transform(X)

xtrain, xtest, ytrain, ytest = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = SVC() model.fit(xtrain, ytrain)

print("Accuracy:", model.score(xtest, ytest))

****

* **Make a Movie Recommendation System Using SVM**

|  |  |  |
| --- | --- | --- |
| import pandas as pd  import numpy as np | | |
| from from from | sklearn.model\_selection import train\_test\_split sklearn.svm import SVC  sklearn.preprocessing import StandardScaler |  |
| data | = pd.DataFrame({  'Genre': [1, 0, 1, 0, 1, 0, 1, 0],  'Year': [2012, 2015, 2010, 2020, 2011, 2018, 2019, | 2013], |
| }) | 'Actor\_Score': [8.1, 6.5, 9.0, 5.5, 8.5, 6.0, 9.2,  'Liked': [1, 0, 1, 0, 1, 0, 1, 0] | 4.8], |
| X = data.drop('Liked', axis=1) y = data['Liked']  scaler = StandardScaler() X\_scaled = scaler.fit\_transform(X)  xtrain, xtest, ytrain, ytest = train\_test\_split(  X\_scaled, y, test\_size=0.25, random\_state=42, stratify=y  )  model = SVC(kernel='linear') model.fit(xtrain, ytrain)  accuracy = model.score(xtest, ytest) print(f"Model Accuracy: {accuracy \* 100:.2f}%") | | |

new\_movie\_df = pd.DataFrame([[1, 2021, 7.8]], columns=['Genre', 'Year', 'Actor\_Score'])

new\_movie\_scaled = scaler.transform(new\_movie\_df) prediction = model.predict(new\_movie\_scaled)

print("Recommendation:", "👍 Like" if prediction[0] == 1 else "👎 Dislike")

**.**



* **Use SVM to classify handwritten digits from the MNIST dataset.**

****

from sklearn.datasets import load\_digits

from sklearn.model\_selection import train\_test\_split from sklearn.svm import SVC

digits = load\_digits()

X, y = digits.data, digits.target

xtrain, xtest, ytrain, ytest = train\_test\_split(X, y, test\_size=0.2, random\_state=0)

model = SVC() model.fit(xtrain, ytrain)

print("Accuracy:", model.score(xtest, ytest))