Let us start by importing some important basic libraries.

**import numpy as np**

**import pandas as pd**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

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

**from sklearn.preprocessing import StandardScaler**

**from sklearn.linear\_model import LogisticRegression**

**from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix**

**import warnings**

**warnings.simplefilter("ignore")**

matplotlib and seaborn are used for visualizations and warnings; we can ignore all the warnings we encounter.

Import the dataset from your local desktop. Use pandas for it. Enter the path to the dataset file in the read\_csv method. It will import the iris dataset.

**#Import iris dataset**

**df=pd.read\_csv(r"C:\Users\HP\VCS\Iris.csv")**

to view the dataset

**df**

to get the information of the data frame

**df.info()**

If there are any missing values, then modify them before using the dataset. For modifying you can use the fillna() method. It will fill null values.

#checking for null values

**df.isnull().sum()**

to view the columns present in the data frame

**df.columns**

to Drop unwanted columns

**df=df.drop(columns="Id")**

# %%

**df**

View the count plot of species feature using seaborn.

**df['Species'].value\_counts()**

to plot an pair plot

**sns.pairplot(df)**

to plot a histogram plot

**sns.histplot(df['Species']);**

to locate the exact cell

**x=df.iloc[:,:4]**

**y=df.iloc[:,4]**

#Import train\_test\_split to split the data into train and test datasets.

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

**x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,random\_state=2)**

#View their shapes. Use the shape method to view.

**x\_train.shape**

**(108, 4)**

**x\_test.shape**

**(36, 4)**

**y\_train.shape**

**(108,)**

**y\_test.shape**

**(36,)**

# Support vector machine algorithm

**from sklearn.svm import SVC**

**model\_svc = SVC()**

**model\_svc.fit(x\_train, y\_train)**

**prediction1 = model\_svc.predict (x\_test)**

# Calculate the accuracy

**from sklearn.metrics import accuracy\_score**

**print (accuracy\_score (y\_test, prediction1))**

#LOGISTIC REGRESSION

**from sklearn.linear\_model import LogisticRegression**

**model\_LR=LogisticRegression()**

**model\_LR. fit (x\_train,y\_train)**

# For creating the model, import LogisticRegression from the sci-kit learn library.

**from sklearn.linear\_model import LogisticRegression**

**model=LogisticRegression()**

#Now train the model using the fit method. In the fit method, pass training datasets in it. x\_train and y\_train are the training datasets.

**model.fit(x\_train,y\_train)**

**LogisticRegression()**

#Now predict the results using predict method.

**y\_pred=model.predict(x\_test)**

#View the results now,

**y\_pred**

# Decision TreeClassifier

**from sklearn.tree import DecisionTreeClassifier**

**model\_DTC=DecisionTreeClassifier()**

**model\_DTC.fit (x\_train,y\_train)**

**prediction3 = model\_svc.predict(x\_test)**

# Calculate the accuracy

**from sklearn.metrics import accuracy\_score**

**print(accuracy\_score(y\_test, prediction3))**

# A detailed classification report

**from sklearn.metrics import classification\_report**

**print(classification\_report(y\_test, prediction1))**

confusion matrix testing

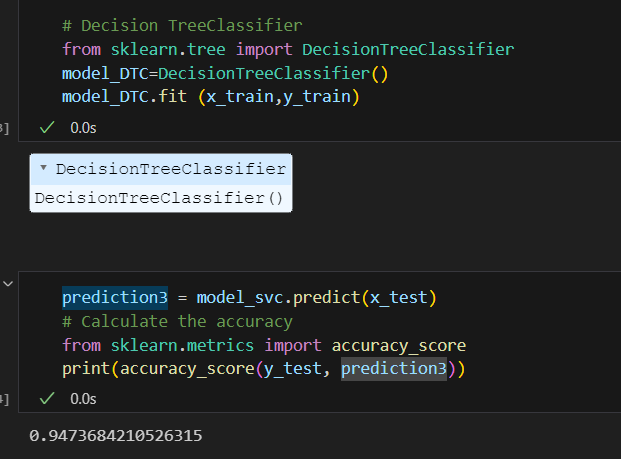
**from sklearn.metrics import accuracy\_score,confusion\_matrix**

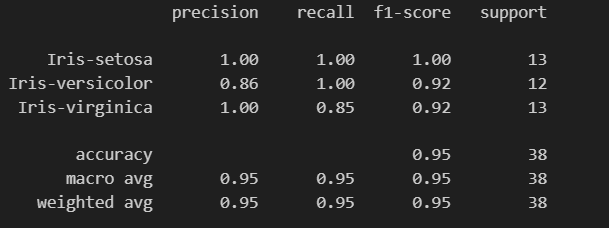
**confusion\_matrix(y\_test,y\_pred)**

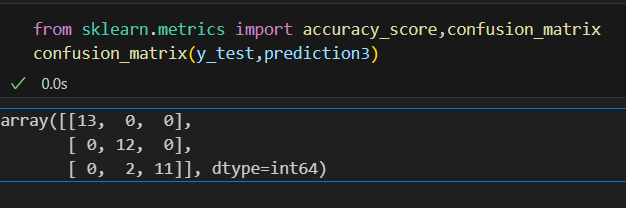
accuracy testing of the model

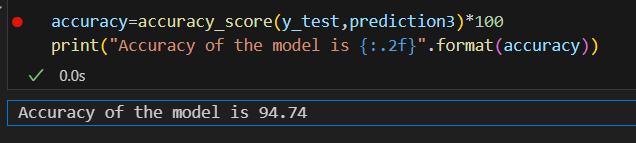
**accuracy=accuracy\_score(y\_test,y\_pred)\*100**

**print("Accuracy of the model is {:.2f}".format(accuracy))**

****

****

****

****