First I imported all the libraries I will use to run this decision tree classification which includes:

|  |
| --- |
| import pandas as pd  import numpy as np  import seaborn as sns  import matplotlib.pyplot as plt  from sklearn.model\_selection import train\_test\_split  from sklearn.tree import DecisionTreeClassifier  from sklearn.metrics import confusion\_matrix  from sklearn.preprocessing import LabelEncoder  from sklearn.metrics import accuracy\_score |

Next, I import the Iris dataset I downloaded from the given link in the assignment using ‘read\_csv function’ and assign it to a variable named ‘df’, and then display the first five rows to confirm its importation.

|  |
| --- |
| # Importing the dataset  df = pd.read\_csv('Iris.csv')  print(df.head()) |

I checked for null values if there was any:

|  |
| --- |
| df.isnull().any() |

I then created a np-array of values for the independent variables by selecting the columns containing the independent variables which are index position 1 to 4, including all rows and assign it to ‘x’ variable; also, I created another np-array of values for the dependent variables by selecting the last column containing the dependent variable which is index position 5, including all rows and assign it to ‘y’ variable

|  |
| --- |
| x = df.iloc[:, 1:5].values  y = df.iloc[:, [5]].values |

Next, I encoded the target species to integers for easy calculation all through, assigning 'Iris-setosa', 'Iris-versicolor' and 'Iris-virginica' to 0, 1 and 2 respectively using a function and then called the function using a variable ‘y\_encoded’

|  |
| --- |
| def prepare\_targets(y\_enc):  le = LabelEncoder()  le.fit(y\_enc)  y\_enc = le.transform(y)  return y\_enc  y\_encoded = prepare\_targets(y)  print(y\_encoded) |

I plotted the graph to visualize the species given and saved it as ‘output.png’:

|  |
| --- |
| sns.pairplot(df, hue='Species')  plt.savefig('output.png',dpi=300) |

I then splitted the dataset into train set containing 70% (105 rows) and then the test set containing 30% (45 rows) of the given dataset

|  |
| --- |
| x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y\_encoded, test\_size=.3,random\_state=0) |

I called the ‘DecisionTreeClassifier()’ function and assign it to a variable ‘classifier’, and then fitted it to the training set

|  |
| --- |
| classifier = DecisionTreeClassifier(random\_state=0)  classifier.fit(x\_train, y\_train) |

Next, I applied the ‘predict function()’ to the test set and assign it to a variable ‘predictions’

|  |
| --- |
| predictions = classifier.predict(x\_test) |

I then applied confusion matrix to the dependent variable test values(y\_test) and the predicted values named ‘predictions’ to see the number of correctly predicted values and number of incorrect predicted values

|  |
| --- |
| cm = confusion\_matrix(y\_test,predictions)  print(cm) |

Next, I called the ‘accuracy\_score function()’ and passed ‘y\_test,predictions’ as arguments to check the accuracy score, and then displayed the accuracy result

|  |
| --- |
| print("Accuracy score: ", accuracy\_score(y\_test, predictions)\* 100, "%") |

The accuracy is 97.78%