**In-Lab**

**In-Lab Task 1**

|  |
| --- |
| **Code:**  #Importing required libraries  import pandas as pd  from sklearn.tree import DecisionTreeClassifier  from sklearn.model\_selection import train\_test\_split  from sklearn import metrics  print("Libraries Imported Ssuccessfully!"))  **Output:** |

**In-Lab Task 2**

|  |
| --- |
| **Code:**  #Loading Dataset  colNames = ['pregnant', 'glucose', 'bp', 'skin', 'insulin', 'bmi', 'pedigree', 'age', 'label']  pima = pd.read\_csv("diabetes.csv", header = None, names = colNames)  pima\_df = pima[1::]  pima\_df.head(5)  print("Dataset loaded successfully!")  **Output:** |

**In-Lab Task 3**

|  |
| --- |
| **Code:**  #Feature Selection  featureCols = ['pregnant', 'glucose', 'bp', 'insulin', 'bmi', 'pedigree', 'age']  X = pima\_df[featureCols]  y = pima\_df.glucose  print(“Features Selection Successful!”)  **Output:** |

**In-Lab Task 4**

|  |
| --- |
| **Code:**  #Splitting the dataset  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.3, random\_state = 1)  print("Dataset Splitting Done!")  **Output:** |

**In-Lab Task 5**

|  |
| --- |
| **Code:**  #Building the decision tree model  model = DecisionTreeClassifier()  model = model.fit(X\_train, y\_train)  y\_pred = model.predict(X\_test)  print("Decision Tree Model Built and Fit successfully!")  **Output:** |

**In-Lab Task 6**

|  |
| --- |
| **Code:**  #Evaluating the model  print("Accuracy: ", round(metrics.accuracy\_score(y\_test, y\_pred),4)\*100,"%")  **Output:** |

**In-Lab Task 7**

|  |
| --- |
| **Code:**  #Visualizing the Decision Trees  from sklearn.tree import export\_graphviz  import graphviz  dotData = export\_graphviz(model, out\_file = None,  feature\_names = X\_train.columns,  class\_names = [str(x) for x in model.classes\_],  filled = True, rounded = True, special\_characters = True)  graph = graphviz.Source(dotData)  graph.render("decision\_tree")  graph.view("decision\_tree")  **Output:**  ***{The output is in the PDF file format}*** |

**In-Lab Task 8**

|  |
| --- |
| **Code:**  #Optimizing the Decision Tree Performance  model = DecisionTreeClassifier(criterion = "entropy", max\_depth = 3)  model = model.fit(X\_train,y\_train)  y\_pred = model.predict(X\_test)  print("Accuracy: ", round(metrics.accuracy\_score(y\_test, y\_pred),3)\*100,"%")  **Output:**    ***{According to the Lab Manual, the output should have become better whereas after the optimization, the accuracy score is in shambles}*** |