CODE DOCUMENTATION

Code structure:

We structured our code in the following way:

- First we ask the user to choose whether he uses a default training dataset or enters his own one.
- In case he chooses to enter his own training dataset :
    - We provide the number of parameters, the sequence and the choice of the string to choose from.
    - The user can write stop whenever he wants to stop entering his data
- In case he chooses the default dataset, he will move directly to make a prediction
- In between the input of dataset and asking the user to make a prediction, we:
    - We encode the training dataset using the LabelEncoder imported from sklearn
    - We fit the data into a tree using the Decision Tree Classifier which will user entropy as a criterion of prediction
- We then prompt the user to input 10 parameters that will be used to make a prediction based on the dataset chosen at the beginning
- He will get an answer corresponding to yes or no, meaning that based on the decision tree that was built along with the input data the customer will wait or will not wait.
- The user can make as many predictions as he wants until he chooses not to.
- Upon termination, a visual representation of the decision tree is generated as a pdf, as well as a classification report and confusion matrix to see the accuracy of the prediction based on the dataset used previously.

Code functionality:

The functionality of our code is as follows:

- The user is prompted to choose to enter his own dataset or a default one, any other input except for “custom” of “default” will not be accepted and will let the user know and ask him again
- If he chooses custom:
    - He will access a menu showing the nature of the input or stop if he has no more datasets to add
    - If the data set is empty, it will end the program
    - If the dataset entered has less then or equal to 3 entries, it will ask the user to enter more
    - For each input, it will check for the length of it and the content to see if it matches the expected pattern, otherwise it will display an error message and ask the user to re-enter an appropriate one.

- If he chooses a default one, he will get a dataset of 12 entries.
- Whichever he chooses, these will be added to an array using numpy array.
- We will then import tree and preprocessing from sklearn
- We will split the dataset into 2 and assign them x and y.
- X will hold the first 10 columns and y the last column
- Using the class LabelEncoder, we will encode the data into integers ranging from 0 to 3 and transform the strings to integers. Later we will inverse_transform when making prediction
- Using the class tree, we will build a Decision tree classifier using “entropy” as criterion, it will put as the root tree the attribute with the less entropy based on this formula

  $$H(Q_m) = -\sum_k p_{mk} \log(p_{mk})$$

  and then the second node as the one having the second less entropy and so on, in our example using the default data set it will put patrons as the root tree. For more information you can visit the official documentation at https://scikit-learn.org/stable/modules/tree.html#tree-mathematical-formulation.

- Fit() method is called on the decision tree classifier object ‘dtc’ by giving it our input variable ‘x’ and ‘y’ as an output variable. This function will build the decision tree model according to the data provided
- plot_tree() function is later used to generate a graphical representation of the decision tree by passing the ‘dtc’ object to it as an argument
- A while loop is used to repeatedly prompt the user for input until the user chooses to stop by entering “no”
  - The variable ‘another_predict’ is used to fix the wording in the case of asking the user if he would want to enter a another prediction
  - The user’s input is checked if it’s valid and in the case of an invalid response, an error message is displayed, and the prompt is repeated
  - The loop will break if the user didn’t want to make a prediction anymore
  - The user is after prompted a list of parameters to choose from, and that is done by choosing the corresponding numerical value to it
  - The user is then asked to input the numerical values for the parameters separated by spaces and they are saved in the ‘user_input’ variable
  - This ‘user_input’ variable is after splitted into a list of parameters

  - This list of parameters saved in the ‘parameters’ variable is then checked to see if it’s valid. This is done by checking the length of the list is checked to see if it’s equal to 10 and by checking the constituents of the parameters to make sure they are a part of the valid values saved in the ‘valid_values’ variable
  - An error message is printed and loop is repeated in the of any errors in the input

    - predict() method is used on the decision tree classifier object ‘dtc’ by giving it our ‘parameters’ variable in order to return the predicted label of what the output will be in reference to the constructed decision tree.
    - This predicted label is later inverse transformed back to the original wording representation and printed, and that is done by the use of the inverse_transform() method which is part of the LabelEncoder class

- At the end, an extra feature was added that prompts the users to input ‘1’ if they would want an improved version of the decision tree as well as a classification report
- If proceeded, the following steps will occur:
    - Imports the modules ‘graphviz’ and ‘classification_report’ function from the ‘sklearn.metrics’ module which are necessary for visualizing the tree and generating the report
    - Uses ‘export_graphviz’ function to generate a representation of the tree as well as ‘feature_names’ and ‘class_names’ parameters to specify the class and feature names
    - The ‘render’ method of the ‘graphviz.Source’ object is used to render the tree as a PDF file
    - Target variables are predicted using the ‘predict’ method and storing it in a variable called ‘y_p’
    - Prints a classification report by taking y and y_p and comparing them
    - Displays a thank you message
- If the user did not input ‘1’, the code skips the generation step and displays a thank you message