BMC PGP AI/ML - CLASSIFICATION
ASSIGNMENT 3
Understanding Decision Tree

Objective: The objective of this assignment is to understand the Decision Tree Classifier and explore the impact of hyperparameters on model performance.

Dataset: Dataset consists of several details about billionaires of the world. For this assignment, we will use the provided features to predict whether the billionaire is self-made or not [target column is selfMade]

Tasks:

1. Preprocess and clean the dataset: Deal with missing values, correlated features [2 mark]
2. Model Building: Implement a Decision Tree Classifier using scikit-learn.
   Split the dataset into training and testing sets (e.g., 80% training, 20% testing).
3. Baseline Model:
   a. Train the Decision Tree model with default hyperparameters on the training dataset [1 mark]
   b. Evaluate the model on the test dataset using accuracy, precision, recall, and F1 score [0.5 mark]
   c. Visualize the decision tree structure using the plot_tree function in scikit-learn [0.5 mark]
4. Hyperparameter Experimentation: We will experiment with two hyperparameters: max depth, min samples split
   a. Train and evaluate the model with different values for each hyperparameter
      i. Max_depth - [3,5,9,15]
      ii. Min_samples_split - [2,5,9,15] [2 marks]
   b. Compare the performance metrics (calculated for basic model in part A) for each set of hyperparameters [1 mark]
   c. Plot F1 score on y-axis against max_depth and min_samples_split values tested in part a on x-axis [1 mark]
5. Parameter Tuning Analysis: Discuss the impact of changing each hyperparameter on the model's performance. [1 mark]

Submission Guidelines:

Submit a Jupyter notebook documenting each step of the process
Include visualizations, code snippets, and explanations for better understanding

Resources:

Refer to scikit-learn documentation for Decision Tree Classifier
Trees: Utilize online resources and tutorials for understanding hyperparameter tuning

Notes: (Additional tips and concepts) - these will not be accounted for making the assignment.

1. Explore GridSearch for hyperparameter tuning - read and understand how and why it could be very useful when many hyperparameters need to be searched.
2. Use this assignment to get in the habit of writing more modular code, make functions for retrieving classification metrics which take in true and predicted values - for a cleaner interface for the experiments.
3. When a matrix of results has to be presented, like classification results for all different hyperparameter experiments in part 4 above, try making small dataframes and populate them with the results - print those instead of simple print statements. This will also enable you to do explorations on these results themselves, if need be.