CSCI-B 559, Project_Part1 of 2

1. (20) Feature Ranking

Given a dataset as below (you can also select your own dataset if you want), write code to calculate the information gain for each feature, and rank all the features. The code should include the steps to calculate entropy, etc., without using the libraries directly. Give analysis and figures also to explain the mathematics you have used in your code, the simulation results, and what you can conclude from your results.

$\langle \boldsymbol{x}_i, y_i \rangle$

Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

2. (40) Efficient Classifier Building based on Information Gain

Use the above dataset (you can also select your own dataset if you want), built a decision tree with the information gain-based method. You can reuse the function you have built in task 1 to calculate information gain. But notice the difference here: you need to build a tree. Starting from the root node, you should be able to generate the branches.

3. (40) Efficient Neural Network Building and Feature Contribution Analysis

Build a neural network for the plant classification, with the 'Iris plants dataset' (https://scikit-learn.org/1.5/datasets/toy_dataset.html) available in the python 'scikit-learn' library, which has 4 features for each of 150 instances (50 instances in each of three classes). Split the dataset to be training and testing with a ratio like 80%:20%. Train and testing the neural network, to report (1) training curve, (2) testing accuracy, (3) using the 'Backward Search'-based wrapper method for feature selection (i.e., dropping one feature in each iteration till only one feature remaining) and getting the performance trend when dropping features. Note: no validation dataset is needed in order to better focus on the 'Backward Search' algorithm.

You can also use other datasets, as long as requested results as required in (1), (2), and (3) are given in your report.

Notes:

- Due date Nov 09, 11:59pm
- Group work is encouraged, but separate report needs to be submitted (with different parameters). Also add your team members' name on the report
- Electronic submission to canvas
- Submission: a project report (doc, docx, or PDF), all executable code files, a readme file to introduce the code
- Python is suggested. Other languages also acceptable.
- All codes need to have detailed comments in the file, especially close to your parameters, and subfunctions
- A readme file to introduce the codes files are required, explaining the flow and dependency