CSE 318 Assignment - 4: Decision Tree

In this assignment, you will implement a **decision tree learning algorithm** and apply it to the provided datasets. Your algorithm must support the following three attribute selection criteria:

- **Information Gain (IG)**
- **Information Gain Ratio (IGR)**
- A custom variant of IG, **Normalized Weighted Information Gain (NWIG)**

Definitions of the Criteria

Let:

- S: the dataset
- A: an attribute in S
- S_v: the subset of S where attribute A has value v
- k: the number of unique values of attribute A
- $|S|$: the size (number of examples) in dataset S

Information Gain (IG)

Information Gain is defined as the reduction in entropy after a dataset is split on an attribute A. It is calculated as:

$$IG(S, A) = Entropy(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \cdot Entropy(S_v)$$

Entropy is defined as:

$$Entropy(S) = -\sum_{c \in \text{Classes}} p(c) \log_2 p(c)$$

where $p(c)$ is the proportion of class c in dataset S.

Information Gain Ratio (IGR)

To correct IG's bias, the Gain Ratio is introduced. It is a normalized form of IG, defined as:

$$GainRatio(S, A) = \frac{IG(S, A)}{IV(A)}$$

where the **Intrinsic Value (IV)** of attribute A is given by:

$$IV(A) = -\sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \cdot \log_2\left(\frac{|S_v|}{|S|}\right)$$

This metric penalizes attributes that split the data into too many small subsets (like IDs or timestamps).

For more details on these criteria, please refer to the references at the end of this document.

Normalized Weighted Information Gain (NWIG)

The custom NWIG modifies IG by penalizing attributes with high cardinality and adjusting for dataset size. It is defined as:

$$NWIG(S, A) = \left(\frac{IG(S, A)}{\log_2(k+1)}\right) \cdot \left(1 - \frac{k-1}{|S|}\right)$$

This penalizes over-splitting by reducing the score of attributes with many distinct values, especially in small datasets.

Tasks

1. Implement a decision tree learning algorithm using either **C++ or Java**. The attribute selection criterion should be a command line parameter: either `IG`, `IGR`, or `NWIG`.
2. Your program must accept another **maximum depth** value as a command line parameter and stop growing the tree beyond that depth (i.e., perform depth-based pruning).

   - Command-line usage:

     ```
     ./decision_tree <criterion> <maxDepth>
     ```

   - Example:

     ```
     ./decision_tree IG 3
     ```

   Note, maxDepth value 0 means no pruning.

3. Randomly split the dataset into **80%** training and **20%** testing. Train your tree using the training data and evaluate its performance on the test set.Repeat the experiment **20 times**, each time with a new random 80/20 split. Record the **average accuracy** across these runs.

4. Compare the average accuracy between IG, IGR, and NWIG across different maximum depths. Also, compare the number of nodes and depth of the tree for these criteria without pruning.

Submission

Submit the following:

- **Source Code** (`.cpp` or `.java`)
- **Report** You must submit a brief report in PDF format along with the code. Your report should include:
  - **Graphs:** Plot average accuracy vs max tree depth for all three criteria for both datasets. Include plots of the number of nodes vs depth wherever applicable.
  - **Observations and Analysis** Note down your observations, such as which criterion performed better at which depths, or which reduced overfitting, how pruning affected accuracy, which one performs consistently better, etc. Include any unexpected patterns, trade-offs you noticed, too.

References

Check your textbooks first. The following sites may also help.

- Wikipedia: *Decision tree learning* – `https://en.wikipedia.org/wiki/Decision_tree_learning`
- A multipart blog from Medium.com `https://ompramod.medium.com/decision-trees-c989527a999` Check other relevant parts, too.
- `https://medium.com/data-science/decision-tree-classifier-explained-a-visual-guide-w`
- Victor Zhou, *Information Gain Explained* – `https://victorzhou.com/blog/information-gain/`