**Question 1 : What is Information Gain, and how is it used in Decision Trees?**

Answer:

Information Gain (IG) is a metric used in decision trees to measure how well a given feature splits a dataset into target classes. It is based on the concept of entropy, which measures the impurity or randomness in the dataset.

Entropy Formula:

**Where:**

* is the probability of class 𝑖 in dataset .
* 𝑐 is the total number of classes
* Information Gain Formula:

**Where:**

* 𝐴 Is a feature
* 𝑆𝑣 is the subset of 𝑆 where feature 𝐴 has value 𝑣

**Use in Decision Trees:**

* At each node, the algorithm calculates IG for all features.
* The feature with the highest Information Gain is selected to split the dataset.
* This process continues recursively to build the tree.

**Question 2: What is the difference between Gini Impurity and Entropy? Hint: Directly compares the two main impurity measures, highlighting strengths, weaknesses, and appropriate use cases**.

Answer:

**Definitions/formulas**

* Gini Impurity (G):

It measures the probability of misclassifying a randomly chosen sample if it were labeled according to the class distribution in the node.

* Entropy (H):

It measures the expected information (in bits) required to identify the class of a randomly chosen sample.

**Range**

* Gini: 0 (for binary classes 0≤).
* Entropy:  (for binary classes ).

**Behavioral comparison**

* Both measure impurity; both are 0 when the node is pure.
* Gini is a quadratic function of class probabilities (computationally cheaper — no logs).
* Entropy grows more slowly near pure distributions but penalizes mid-range uncertainty more subtly.

**When they differ practically**

* For many splits they produce similar rankings of candidate splits (i.e., they often choose the same feature), but small differences can appear in edge cases.
* Gini tends to prefer larger, more balanced partitions slightly (because of quadratic form), while entropy is a bit more sensitive to distribution differences.

**Computation speed**

* Gini is faster (no logarithms). This is why CART (Classification And Regression Trees) uses Gini by default.

**Use cases / algorithm defaults**

**Gini**

* Gini impurity — used by CART (and hence scikit-learn’s DecisionTreeClassifier default). while using the (Classification and Regression Trees) algorithm.
* while working with large datasets where speed matters more than slight gains in accuracy.

**Entropy**

* Entropy — used by ID3/C4.5 family and sometimes chosen when interpreting splits in information-theory terms is desirable.
* while handling datasets where class distribution is highly imbalanced or where accuracy is more important than speed

**Strengths and weaknesses**

* Gini:
  + Faster to compute.
  + Often produces shallower trees (practical advantage).
  + − Slight bias towards features with more categories if not controlled.
* Entropy:
  + Theoretically grounded in information theory; good for interpreting information gain.
  + − Slightly slower to compute; may produce different splits in some datasets.

**Recommendation:**

Use Gini for performance and typically similar results. Use Entropy when you want information-theoretic interpretability or when experimenting shows entropy gives better validation performance for your dataset.

**Question 3:What is Pre-Pruning in Decision Trees?**

Answer:

Pre-pruning (also called early stopping) is the practice of halting the growth of a decision tree before it perfectly fits (or grows deep on) the training data, using predefined constraints. The goal is to prevent overfitting by limiting complexity during training.

**Common pre-pruning hyperparameters (scikit-learn names included):**

* max\_depth — maximum depth of the tree.
* min\_samples\_split — minimum number of samples required to split an internal node.
* min\_samples\_leaf — minimum number of samples required to be at a leaf node.
* max\_leaf\_nodes — maximum number of leaf nodes.
* min\_impurity\_decrease — a split will be made only if it decreases impurity by at least this threshold.
* ccp\_alpha — complexity parameter used for Minimal Cost-Complexity Pruning (but note: ccp\_alpha is applied in a post-pruning style in scikit-learn via pruning after full growth).

**How pre-pruning works in practice**

* The tree stops splitting further if:
  + Maximum depth is reached (max\_depth)
  + Minimum number of samples to split is not met (min\_samples\_split)
  + Minimum samples in a leaf node is small (min\_samples\_leaf)
  + Information Gain or Gini decrease is below a threshold.
  + During tree expansion, if a potential split would violate any pre-pruning condition (e.g., would create leaves with fewer than min\_samples\_leaf), the split is not performed and the node becomes a leaf.
  + This keeps the tree simpler and reduces variance at the cost of possibly increasing bias.

**Advantages**

* + Faster training time
  + Produces simpler and more interpretable trees
  + Controls overfitting by reducing tree complexity.
  + Less computational time and memory than growing a large tree and post-pruning.
  + Direct control over model complexity.

**Disadvantages**

* + Choosing hyperparameters incorrectly can lead to underfitting.
  + It might stop growth too early, preventing the discovery of useful, fine-grained patterns.

**Question 4:Write a Python program to train a Decision Tree Classifier using Gini**

Impurity as the criterion and print the feature importances (practical). Hint: Use criterion='gini' in DecisionTreeClassifier and access .feature\_importances\_. (Include your Python code and output in the code box below.)