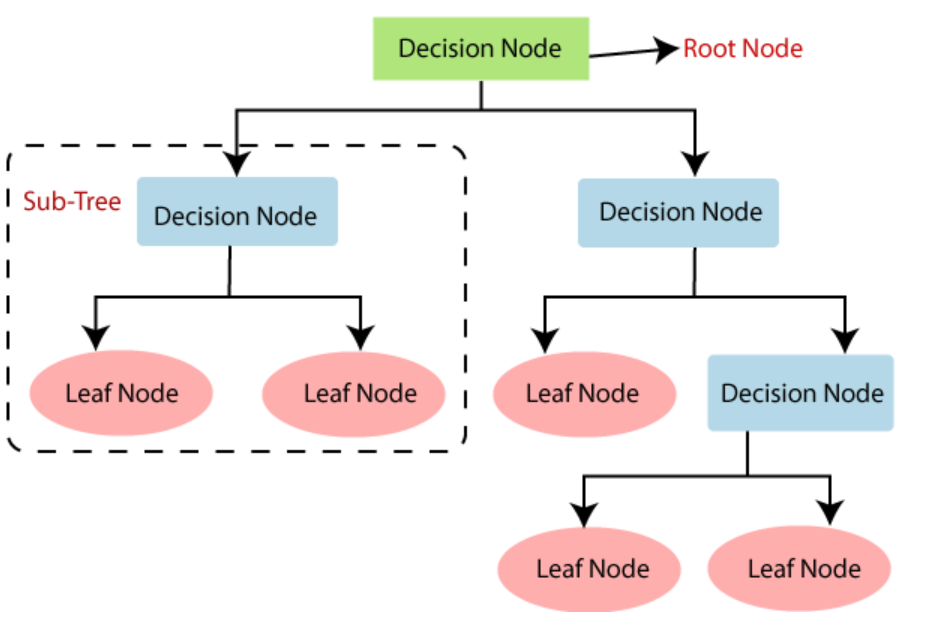
**Decision Trees:**



Decision trees are a popular and powerful tool used for classification and regression tasks in machine learning. They are simple to understand, interpret, and visualize, making them useful for various applications. Here's a detailed overview of decision trees:

**What is a Decision Tree?**

A decision tree is a flowchart-like structure in which each internal node represents a decision based on the value of a feature, each branch represents the outcome of the decision, and each leaf node represents a class label (for classification) or a continuous value (for regression).

**How Decision Trees Work**

1. **Splitting**: The process of dividing a node into two or more sub-nodes based on a certain criterion.
2. **Attribute Selection**: The choice of which feature to split on at each step, often using metrics like Information Gain, Gini Index, or Chi-square.
3. **Stopping Criteria**: Conditions that determine when to stop growing the tree, such as reaching a maximum depth, having a minimum number of samples per node, or achieving a minimum impurity decrease.
4. **Pruning**: The process of removing parts of the tree that do not provide additional power in predicting target values, which helps to prevent overfitting.

**Common Decision Tree Algorithms**

1. **ID3 (Iterative Dichotomiser 3)**:
   * **Description**: Uses a greedy approach to construct a tree by choosing the attribute that maximizes information gain at each node.
   * **Splitting Criterion**: Information Gain (based on entropy).
   * **Limitations**: Can overfit the data, does not handle continuous features, and is sensitive to noisy data.
2. **C4.5**:
   * **Description**: An extension of ID3 that can handle both categorical and continuous data, and deals with missing values.
   * **Splitting Criterion**: Information Gain Ratio.
   * **Features**: Includes pruning methods to avoid overfitting and handles incomplete data.
3. **CHAID (Chi-squared Automatic Interaction Detector)**:
   * **Description**: Uses Chi-square tests to determine splits and is often used in market research.
   * **Splitting Criterion**: Chi-square test for independence.
   * **Features**: Produces multi-way splits and merges categories of predictor variables.
4. **CART (Classification and Regression Trees)**:
   * **Description**: Can be used for both classification and regression tasks.
   * **Splitting Criterion**: Gini Index for classification and variance reduction for regression.
   * **Features**: Produces binary trees, handles both continuous and categorical data, and includes pruning techniques.
5. **MARS (Multivariate Adaptive Regression Splines)**:
   * **Description**: Extends the decision tree concept to handle regression problems with high-dimensional data.
   * **Splitting Criterion**: Uses basis functions to model relationships.
   * **Features**: Models complex interactions and nonlinearities, effective for regression.

**Advantages of Decision Trees**

* **Interpretability**: Easy to visualize and interpret, making them accessible to non-experts.
* **Non-parametric**: Do not assume any underlying distribution for the data.
* **Versatility**: Can handle both numerical and categorical data, and can be used for both classification and regression.
* **Feature Selection**: Implicitly performs feature selection by choosing splits.

**Disadvantages of Decision Trees**

* **Overfitting**: Can easily overfit the training data, especially if the tree is too deep.
* **Instability**: Small changes in the data can lead to significantly different trees.
* **Bias**: Can be biased if some classes dominate; requires balancing or weighting.

**Applications of Decision Trees**

* **Medical Diagnosis**: Classifying diseases based on patient symptoms and test results.
* **Finance**: Credit scoring and risk assessment.
* **Marketing**: Customer segmentation and targeted marketing.
* **Manufacturing**: Quality control and fault diagnosis.

**Pros of Decision Trees**

* **Easy to Understand**: Decision trees are intuitive and easy to understand, even for non-experts. The structure is straightforward and can be visualized, making it easy to interpret the decision-making process.
* **Visual Representation**: The tree-like model provides a clear and visual representation of decisions and their possible consequences.
* **Non-parametric and No Assumptions**: Decision trees do not assume any specific distribution of the data, making them versatile and suitable for a wide range of data types.
* **Handles Different Data Types**: They can handle both numerical and categorical data.
* **Classification and Regression**: Decision trees can be used for both classification and regression tasks.
* **Automatic Feature Selection**: During the tree-building process, decision trees automatically perform feature selection, choosing the most informative features to split the data.
* **Captures Non-linear Relationships**: Decision trees can capture non-linear relationships between features and the target variable.
* **Handles Irrelevant Features**: Decision trees can handle irrelevant features relatively well, as splits that don't improve the model's performance are unlikely to be chosen.
* **No problem** of Scaling, Normalizing, Standardizing, Calculations, mini batches

**Cons of Decision Trees**

* **Prone to Overfitting**: Decision trees can easily overfit the training data, especially if the tree is allowed to grow too deep. This results in poor generalization to unseen data.
* **Complex Trees**: Without pruning, decision trees can become very complex and capture noise in the data.
* **Sensitive to Data Variations**: Small changes in the training data can result in significantly different tree structures, making them unstable.
* **Variance**: High variance can be an issue, as the decision tree may vary greatly with small changes in the data.
* **Bias Towards Dominant Classes**: If some classes dominate the dataset, the decision tree might become biased towards these classes, leading to poor performance on minority classes.
* **Local Optima**: The greedy nature of decision tree algorithms (making the best choice at each node without considering future consequences) may not always lead to the globally optimal tree.
* **Myopic Decisions**: Decisions made at each node are based on immediate criteria and may not consider the overall best splits.
* **Discretization**: While decision trees can handle continuous variables, they do so by creating binary splits, which might not capture the complexity of the data as well as other algorithms.
* **Large Datasets**: For very large datasets, training a decision tree can be computationally expensive and time-consuming, especially if the dataset has a high dimensionality.

**Mitigating the Cons**

To address some of the limitations of decision trees, ensemble methods such as Random Forests and Gradient Boosting are commonly used. These methods combine multiple decision trees to improve stability, reduce overfitting, and enhance predictive performance.

* **Random Forests**: Combines multiple decision trees by averaging their predictions (for regression) or using majority vote (for classification). This reduces overfitting and increases robustness.
* **Gradient Boosting**: Builds trees sequentially, with each tree correcting the errors of the previous ones. This approach can produce highly accurate models.

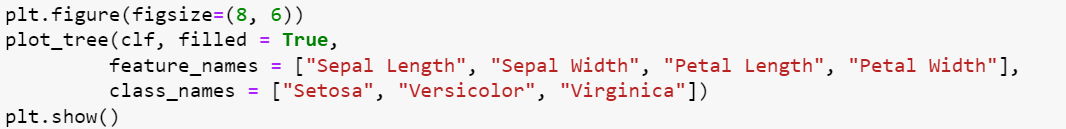
**Conclusion**

Decision trees are a powerful tool for both classification and regression tasks due to their simplicity and interpretability. However, their tendency to overfit and instability with small changes in the data can be significant drawbacks. Using ensemble methods like Random Forests or Gradient Boosting can help mitigate these issues, providing more robust and accurate models.

Example:

* 1. Create a Decision Tree based on the classes of flowers (Setosa, Versicolor, Virginica). Classify based on the dimensions of Sepal and Petal Dimensions
  2. Extract the dataset
  3. Create placeholder based on DecisionTreeClassifier()
  4. Train the mode by clf.fit()
  5. Visualize the tree with plot\_tree(), matplotlib() and with added parameters

Below tests are for Iris Dataset,



A diagram of a diagram

Description automatically generated with medium confidence

Samples considered = 150 rows (Data points)

How many Data points are there in each of the classes

4th Feature of the Input (X) i.e. Petal Width. How many Petal Width are < 0.8

**Gini Impurity**: This is a metric used to evaluate the purity of a dataset in decision tree algorithms. It measures the frequency at which any element of the dataset would be misclassified if it was randomly labeled according to the distribution of labels in the dataset.

Gini Impurity measures the changes in the accuracy when the data is randomly shuffled.

Bigger Misclassification 🡪 Bigger Gini Impurity

j

∑ Pi \* (1 - Pi)

i = 1

Example: Assume we have 2 classes (Red bin and Blue Bin) having 3 red balls and 7 blue balls

1. P(Red ball) = 3/10 = 0.3 P(~Red Ball or misclassification) = 7/10 = 0.7
2. P(Blue ball) = 7/10 = 0.7 P(~Blue Ball or misclassification) = 3/10 = 0.3

Gini = 3/10 \* 7/10 + 7/10 \* 3/10 = 0.42

Objective of the algorithm is to minimize the Gini impurity in the resulting sub-nodes after the split, thereby creating more homogeneous nodes.

Note:

* A Gini impurity of 0 indicates perfect purity, meaning all elements in the dataset belong to a single class.
* A higher Gini impurity indicates more diversity in the classes, and the maximum Gini impurity occurs when the elements are equally distributed among the classes.
* By default sklearn opts for Gini impurity for class splits

**Information Gain or Entropy**: This is used in ID3 or C4.5 or C5.0 algorithms. Information Gain (IG) is a metric used to evaluate the effectiveness of an attribute in classifying a dataset in decision tree algorithms. It measures the reduction in entropy (uncertainty or impurity) achieved by partitioning the dataset based on a given attribute. The attribute that provides the highest information gain is chosen for the split.

A white background with black text

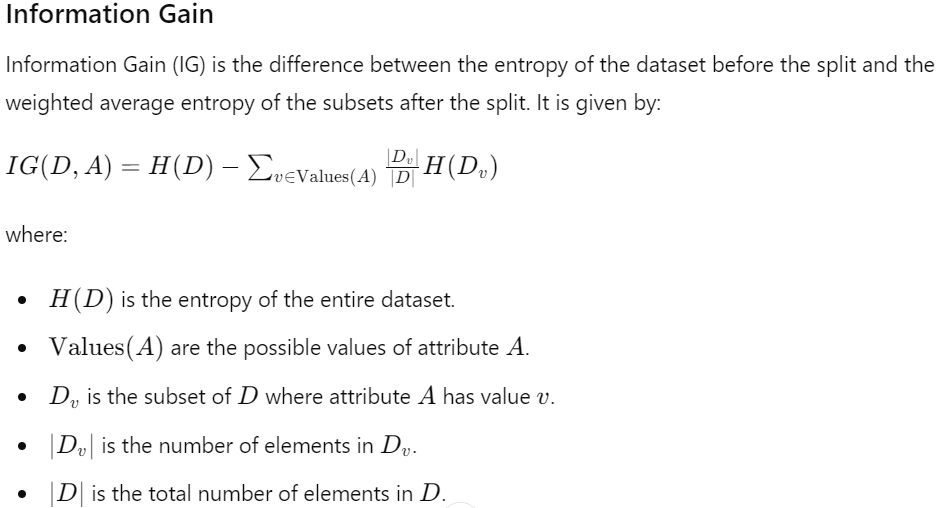
Description automatically generated

**Properties of Entropy**

1. **Non-negativity**: Entropy is always greater than or equal to 0.
2. **Maximum Entropy**: Entropy is maximized when all classes are equally likely. For nnn equally probable classes, H(D)=log⁡2(n)H(D) = \log\_2(n)H(D)=log2​(n).
3. **Minimum Entropy**: Entropy is 0 when the dataset is perfectly pure, meaning all instances belong to a single class.

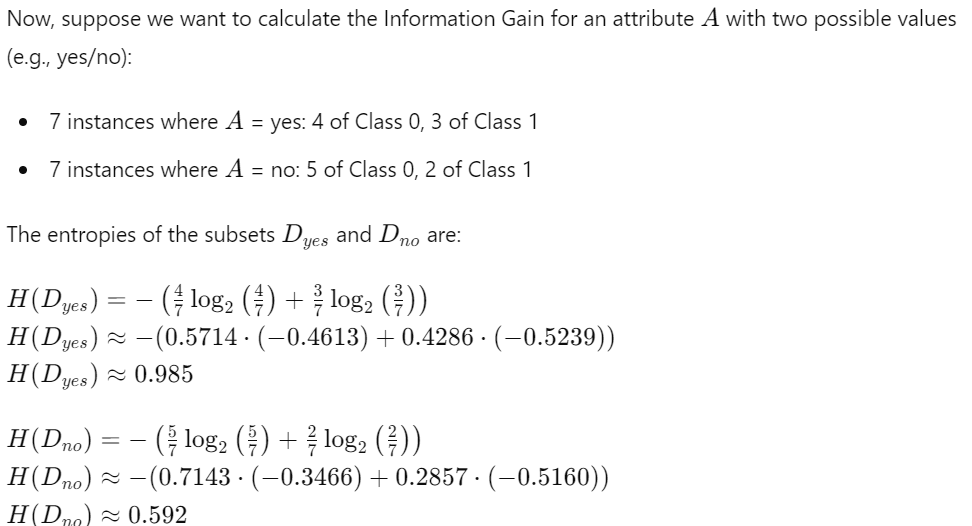
**Interpretation**

* **High Entropy**: Indicates high impurity or disorder. The classes are distributed more evenly.
* **Low Entropy**: Indicates low impurity or high order. The classes are more predictable and concentrated in fewer categories.



A math problem with numbers and equations

Description automatically generated



A math equations and numbers

Description automatically generated with medium confidence

**Interpretaation:**

* A higher Information Gain indicates a better attribute for splitting the dataset, leading to more homogeneous subsets.
* If an attribute has an Information Gain of 0, it means that splitting the dataset based on that attribute does not reduce the entropy, hence it is not a useful attribute for classification.

**Usage in Decision Trees**

In decision tree algorithms like ID3, C4.5, and others, the attribute with the highest Information Gain is selected as the splitting criterion at each node. This process is recursively repeated to build the entire decision tree, aiming to create branches that increase the purity of the subsets and lead to more accurate classifications. The goal is to reduce the entropy, creating subsets of the data that are more homogeneous in terms of class distribution. This reduction in entropy is quantified as Information Gain.

**Pruning**: Pruning is a technique used in decision tree algorithms to reduce the complexity of the tree and improve its generalization to unseen data. It involves removing sections of the tree that provide little to no additional predictive power. Pruning can help prevent overfitting, which occurs when a decision tree is too closely fitted to the training data and fails to perform well on new, unseen data.

There are two main types of pruning techniques: **pre-pruning (or early stopping)** and **post-pruning (or pruning after tree construction)**.

**Pre-pruning (Early Stopping)**

Pre-pruning involves halting the growth of the decision tree early based on certain conditions. Some common criteria for pre-pruning include:

* **Maximum depth**: Limiting the depth of the tree to a predefined maximum value.
* **Minimum samples per node**: Requiring a minimum number of samples in a node to allow for further splitting.
* **Minimum information gain**: Stopping the split if the information gain from the split is below a certain threshold.
* **Statistical tests**: Using statistical significance tests to decide whether to split a node further.

**Post-pruning (Pruning After Tree Construction)**

Post-pruning involves growing the full decision tree and then removing or collapsing nodes that contribute little to the predictive power of the tree. Common post-pruning methods include:

* **Reduced Error Pruning**: Removing nodes that do not reduce the overall error rate of the tree on a validation set. Nodes are pruned if their removal does not increase the error rate.
* **Cost Complexity Pruning (CCP)**: Also known as weakest link pruning, this method involves removing nodes that result in the smallest increase in a cost complexity criterion, which balances the tree's accuracy with its complexity. CCP is often used with decision trees like CART (Classification and Regression Trees).

**Reduced Error Pruning Example**

Consider a decision tree grown to classify whether a person will play tennis based on weather conditions. Suppose we have a fully grown tree and we use a validation set to evaluate the effect of pruning.

1. Evaluate the accuracy of the tree on the validation set.
2. Iteratively remove nodes (starting from the leaves) and evaluate the tree's accuracy after each removal.
3. If removing a node does not decrease the accuracy, keep the node pruned. If the accuracy decreases, retain the node.

**Practical Considerations**

* **Cross-validation**: It is often used to select the best pruning level by evaluating the performance of the tree on different subsets of the data.
* **Hyperparameter tuning**: The parameters controlling pre-pruning and post-pruning (such as maximum depth, minimum samples per node, and the complexity parameter α\alphaα) can be fine-tuned to achieve optimal performance.

**Benefits of Pruning**

* **Reduces overfitting**: Pruning helps the model generalize better to new data by removing overly specific patterns learned from the training set.
* **Simplifies the model**: A pruned tree is easier to interpret and understand.
* **Improves performance**: By reducing the size of the tree, pruning can also enhance computational efficiency and reduce prediction time.

**Conclusion**

Pruning is an essential step in decision tree algorithms that helps improve model performance and interpretability. By effectively managing the complexity of the tree, pruning techniques ensure that the decision tree remains robust and generalizes well to unseen data.

**RANDOM FOREST:**

Random Forest is an ensemble learning method used for classification, regression, and other tasks that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It was introduced by Leo Breiman and Adele Cutler and is known for its high accuracy, robustness, and ability to handle large datasets with higher dimensionality.

**Key Concepts**

1. **Ensemble Learning**:
   * Combines multiple models to improve performance over individual models.
2. **Decision Trees**:
   * A tree-like model used to make decisions based on feature values. Each internal node represents a "test" on a feature, each branch represents the outcome of the test, and each leaf node represents a class label or continuous value.
3. **Bootstrap Aggregating (Bagging)**:
   * A method used to reduce variance and improve model performance by training each tree on a different random subset of the data (with replacement). This is obtained by uniformly sampling the original dataset with replacement.
   * Without replacement: Here subset of the observations is selected randomly, and once an observation is selected it cannot be selected again.
   * With replacement: Here subset of observations is selected randomly, and an observation may be selected more than once.
4. **Random Feature Selection**:
   * At each split in the decision tree, a random subset of features is considered to introduce diversity among the trees.

**Steps in Random Forest Algorithm**

1. **Create Multiple Bootstrapped Datasets**:
   * Randomly select samples from the training data with replacement to create multiple subsets.
2. **Build Decision Trees**:
   * Train a decision tree on each bootstrapped dataset.
   * At each node in the tree, select the best split from a random subset of features.
3. **Aggregate Predictions**:
   * For classification, use majority voting from all trees.
   * For regression, use the average prediction from all trees.

**Advantages**

* **Accuracy**: Often provides better accuracy than a single decision tree by reducing overfitting.
* **Robustness**: Less sensitive to noisy data and outliers.
* **Versatility**: Can handle both classification and regression problems and works well with high-dimensional data.

**Disadvantages**

* **Complexity**: More complex and computationally intensive than a single decision tree.
* **Interpretability**: Individual trees are interpretable, but the ensemble is not easily interpretable.

**Applications**

* **Classification**: Medical diagnosis, spam detection, image and speech recognition.
* **Regression**: Predicting house prices, stock market trends.
* **Feature Selection**: Identifying important features in a dataset.

**NOTE**: Sometimes decision trees will clumsy due to overfitting. This can be avoided by using the pruning technique (ccp\_alpha = 0 to 1).