Name :- Vijay Gurung

ID Number : MST03 – 0070

*** Topic : Decision_Tree ***

Introduction

A Decision Tree is a popular machine learning algorithm used for classification and regression tasks. It splits data into subsets based on the value of input features, making decisions at each node to reach a final prediction. Decision Trees are intuitive, easy to interpret, and can handle both numerical and categorical data.

How Decision Trees Work :

1. **Root Node:** Represents the entire dataset and is the starting point of the tree.
2. **Decision Nodes:** Points where the dataset is split based on feature values.
3. **Leaf Nodes:** Terminal nodes that represent the final output or decision.

The tree is built by recursively splitting the data based on the feature that provides the best separation according to a certain criterion (e.g., Gini impurity, Information Gain).

Key Concepts :

1. **Entropy and Information Gain:**
    - **Entropy:** Measures the impurity or uncertainty in a dataset. Entropy(S)=−∑i=1cpilog⁡2(pi)\text{Entropy}(S) = - \sum_{i=1}^{c} p_i \log_2(p_i)Entropy(S)=−i=1∑cpilog2(pi) where pip_ipi is the probability of class iii in dataset SSS.
    - **Information Gain:** Reduction in entropy after a dataset is split on a feature. Information Gain(S,A)=Entropy(S)−∑v∈Values(A)|Sv||S|Entropy(Sv)\text{Information Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)Information Gain(S,A)=Entropy(S)−v∈Values(A)∑|S||Sv|Entropy(Sv) where SvS_vSv is the subset of SSS for which feature AAA has value vvv.
2. **Gini Impurity:** Measures the impurity of a dataset.

Gini(S)=1−∑i=1cpi2\text{Gini}(S) = 1 - \sum_{i=1}^{c} p_i^2Gini(S)=1−i=1∑cpi2

where pip_ipi is the probability of class iii in dataset SSS.

Example: Classifying Iris Species :

Let's consider the famous Iris dataset, which contains 150 samples of iris flowers with four features (sepal length, sepal width, petal length, petal width) and three species (setosa, versicolor, virginica)

Step-by-Step Process:

1. Load the Dataset:

```
from sklearn.datasets import load_iris
import pandas as pd

iris = load_iris()
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
df['species'] = iris.target
```

2.Split the Data:

from sklearn.model_selection import train_test_split

X = df.drop('species', axis=1)

y = df['species']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

3.Train the Decision Tree Model:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

clf = DecisionTreeClassifier(criterion='entropy', random_state=42)
clf.fit(X_train, y_train)
```

4.Visualize the Decision Tree:

import matplotlib.pyplot as plt

plt.figure(figsize=(20,10))

tree.plot_tree(clf, filled=True, feature_names=iris.feature_names, class_names=iris.target_names)

plt.show()

5.Evaluate the Model:

```
from sklearn.metrics import accuracy_score

y_pred = clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy * 100:.2f}%')
```

Example Output and Interpretation :

After running the above code, We should see a visual representation of the decision tree. Each node in the tree shows the feature and threshold used for the split, the Gini impurity, the number of samples at the node and the class distribution.

Interpretation:

- **Root Node:** The root node represents the entire dataset.
- **Decision Nodes:** Each internal node represents a decision based on a feature value.
- **Leaf Nodes:** Terminal nodes represent the final prediction (iris species in this case).

For example, the tree might show a decision based on "petal length (cm) <= 2.45". If true, it leads to one branch, otherwise to another. This decision-making continues until the tree reaches a leaf node.

Advantages and Disadvantages :

1.Advantages:

- Easy to understand and interpret.
- Can handle both numerical and categorical data.
- Requires little data preprocessing.
- Non-parametric: No assumptions about data distribution.

2.Disadvantages:

- Prone to overfitting, especially with deep trees.
- Can be unstable: small changes in data can lead to different trees.
- Can be biased towards features with more levels.

Conclusion :

Decision Trees are powerful and intuitive models for classification and regression tasks. By understanding their structure and how they split data, We can build and interpret models effectively. However, it's essential to manage their complexity to prevent overfitting, often using techniques like pruning or ensemble methods (e.g., Random Forests).