Decision Tree

Reading Material

PW SKILLS

Topics Covered

1. What is Decision Tree Classifier
   - Definition and Significance
   - How Decision Tree Classification Works
   - Applications of Decision Tree Classifier

2. In-Depth Mathematical Intuition
   - Entropy and Information Gain
   - Gini Impurity
   - Splitting Criteria
   - Tree Pruning

3. Confusion Matrix
   - Definition and Significance
   - Components of Confusion Matrix (True Positives, False Positives, True Negatives, False Negatives)
   - Interpretation and Use

4. Best Metric Selection
   - Metrics for Classification (Accuracy, Precision, Recall, F1 Score)
   - Choosing the Best Metric Based on the Problem
   - Trade-offs between Metrics

5. Decision Tree Regressor
   - Definition and Significance
   - How Decision Tree Regression Works
   - Applications of Decision Tree Regressor

6. In-Depth Mathematical Intuition for Decision Tree Regressor
   - Variance Reduction
   - Mean Squared Error (MSE) and Mean Absolute Error (MAE)
   - Tree Pruning in Regression

1. What is Decision Tree Classifier

Definition and Significance

A Decision Tree Classifier is a type of supervised learning method utilized for categorization purposes within the fields of statistics and machine learning. It shows choices and their potential results in a diagram that looks like a tree, with each inner node representing a characteristic, each decision rule shown as a branch, and each end node indicating a result or classification. Decision trees are preferred because of their simplicity and clarity, which makes them suitable for decision-making tasks.

How Decision Tree Classification Works

Constructing a decision tree requires going through various important stages.

Choosing the Optimal Attribute: The algorithm assesses different attributes by utilizing metrics such as Gini impurity or entropy in order to identify the top feature for dividing the data.

Dividing the Dataset: The dataset is split into subsets according to the chosen attribute, resulting in the formation of new nodes.

Iterative division: The splitting operation is carried out iteratively on every subset until a certain stopping condition is satisfied, like when all data points in a node are from the same category or the tree reaches its maximum depth.

The resultant framework enables easy categorization depending on the routes from the central point to the end nodes.

Application of Decision Tree Classifier

Decision Tree Classifiers are commonly employed in a variety of fields, such as:

- **Financial analysis:** Used for evaluating creditworthiness and assessing potential risks.
- **Healthcare:** Supporting in the identification of illnesses and deciding on treatment strategies.
- **Market analysis:** To divide customers into groups and forecast how they will behave.

Managing operations involves making strategic decisions and allocating resources effectively.
Their popularity for numerous classification tasks comes from their capacity to manage both categorical and numerical data effectively, as well as their easily interpreted visual displays.

2. In-Depth Mathematical Intuition

Calculating the likelihood of classes

A Decision Tree has the capability to predict the likelihood of an instance being in a specific class k.
It moves through the tree to the leaf node that matches the instance and gives back the ratio of training instances of class k in that node.

If a leaf node contains 54 samples, with 49 belonging to "Iris versicolor" and 5 belonging to "Iris virginica", the predicted probabilities would be as follows:

0% of the Iris setosa specimens (0 out of 54) did not have any observed result.

90.7% representing Iris versicolor (49 out of 54)

9.3% of Iris virginica samples were found in the study (5 out of 54).

The predicted class is the one with the highest probability.

Scikit-Learn utilizes the CART algorithm for training Decision Trees, which are also known as "growing" trees. The method operates by initially dividing the training dataset into two parts using one characteristic k and a specific value t (such as "petal length ≤ 2.45 cm"). What is the process for selecting k and t? It looks for the combination (k, t) that generates the most unadulterated groups (taking into account their size). The cost function in the equation above is what the algorithm aims to reduce. After the CART algorithm splits the training set into two parts, it continues to split each subset using the same method, and repeats this process recursively. The recursion ends when it hits the specified maximum depth or when it is unable to find a split that decreases impurity. Several more hyperparameters (to be described shortly) manage other stopping conditions such as min_samples_split, min_samples_leaf, min_weight_fraction_leaf, and max_leaf_nodes.

Entropy measures the amount of uncertainty or disorder in a dataset. It quantifies how mixed the class labels are. The formula for entropy H(D)H(D)H(D) for a dataset DDD is:

$H(D)=-\Sigma pi \log2(pi)$

where:
- c is the number of classes,
- pi is the proportion of examples in class iii.

Information Gain (IG) is the reduction in entropy after splitting a dataset based on a particular feature. It helps us determine which feature provides the most information about the class labels. The formula for Information

Gain is:
$IG(D,A)=H(D)-\Sigma|Dv|/|D|H(Dv)$

where:
- D is the dataset,
- A is the attribute on which the split is being made,
- Dv is the subset of D where the attribute A takes value v,
- H(D) is the entropy of the dataset D,
- H(Dv) is the entropy of the subset Dv.

The goal is to maximize Information Gain when deciding which feature to split on, as it indicates a greater reduction in uncertainty.

Gini Impurity

Gini Impurity is another metric to evaluate the quality of a split. It measures the likelihood of incorrect classification of a randomly chosen element if it was labeled according to the distribution of labels in the dataset. The formula for Gini Impurity G(D) is:

$G(D)=1-\Sigma pi^2$

where:
- Pi is the proportion of examples in class i.

A lower Gini Impurity indicates a purer node. When building a decision tree, the aim is to select splits that minimize the Gini Impurity.

Splitting Criteria
The splitting criteria in decision trees determine how the dataset is divided at each node. Frequently utilized criteria consist of:

Information Gain: As mentioned earlier, it identifies the feature that results in the greatest decrease in entropy.

Gini Impurity: This method selects the feature that leads to the least Gini Impurity following the division.

Chi-Square: This statistical analysis measures how closely observed data matches the expected results, specifically for variables that are categorical.
Every criteria has advantages. For example, Gini impurity is frequently less computationally demanding than entropy, making it a preferred option for bigger datasets.

Tree Pruning

Tree Pruning involves removing sections of the tree with low predictive power to prevent overfitting in decision trees. There are two primary categories of pruning:

Early stopping, also known as pre-pruning, involves stopping the tree-building process before it achieves perfect classification of the training set. Early stopping criteria can involve imposing a limit on depth, stipulating a minimum number of samples for each leaf, or halting when information gain drops beneath a specific threshold.

Post-Pruning involves growing the tree to its full depth first, then removing insignificant branches. This technique strives to streamline the model while retaining its ability to predict. Methods such as cost complexity pruning are frequently utilized, in which nodes are eliminated according to their influence on the tree's overall accuracy.

3. Confusion Matrix

Definition and Significance

A confusion matrix is a method utilized for evaluating the effectiveness of a classification model. It outlines the outcomes from forecasting by contrasting the real target values with the estimated values. Understanding how effectively the model differentiates between various classes is crucial, especially when dealing with imbalanced datasets or varying costs of misclassifications.

Components of the Confusion Matrix

True Positives (TP): Instances where the model correctly predicts the positive class.
Example: If the actual class is "spam" and the model predicts "spam," this counts as a true positive.
Significance: A high TP count indicates strong performance in identifying the positive class.

False Positives (FP): Instances where the model incorrectly predicts the positive class.
Example: If the actual class is "not spam" but the model predicts "spam," this is a false positive
Significance: High FP values can lead to unnecessary actions, such as mislabeling important emails as spam.

True Negatives (TN): Instances where the model correctly predicts the negative class.
Example: If the actual class is "not spam" and the model predicts "not spam," this is a true negative.
Significance: A high TN count indicates effective identification of the negative class.

False Negatives (FN): Definition: Instances where the model incorrectly predicts the negative class.
Example: If the actual class is "spam" but the model predicts "not spam," this is a false negative.
Significance: High FN values can result in missing critical detections, such as failing to identify actual spam.

Interpretation and Use

The confusion matrix can be used to calculate several important metrics:

- **Accuracy:** Formula =(TP+TN)/(TP+TN+FP+FN)

Interpretation: Represents the proportion of correctly classified instances.Although useful, accuracy can be misleading if the classes are imbalanced.

- **Precision:** Formula=TP/(TP+FP)

Interpretation: Measures how many instances predicted as positive are actually positive. High precision indicates a low false positive rate.

- **Recall (Sensitivity or True Positive Rate):** Formula=TP/(TP+FN)

Interpretation: Measures how many actual positives are correctly identified by the model. High recall indicates a low false negative rate.

- **F1-Score:** Formula=(2×Precision×Recall)/(Precision+Recall)

Interpretation: The F1-Score is the harmonic mean of precision and recall, providing a balanced measure when dealing with imbalanced datasets.

- **Specificity (True Negative Rate):** Formula=TN/(TN+FP)

Interpretation: Measures how effectively the model identifies the negative class.

4. Best Metric Selection

Metrics for Classification

Accuracy
Accuracy measures the ratio of correctly predicted instances (both true positives and true negatives) to the total number of instances. It is a straightforward metric that provides a quick overview of model performance.
When to Use: Accuracy is suitable for balanced datasets where the costs of false positives and false negatives are similar.

Limitation: In imbalanced datasets, accuracy can be misleading. For instance, if 95% of the data belongs to one class, a model that predicts all instances as that class may achieve high accuracy but fail to identify the minority class effectively.

Precision
Precision quantifies the proportion of true positives among all instances predicted as positive. It indicates how many of the predicted positive cases are actually correct.

When to Use: Precision is critical when the cost of false positives is high. For example, in spam detection, minimizing the number of legitimate emails marked as spam is essential.
Limitation: Precision does not account for false negatives, so it may not be sufficient in situations where missing positive cases is crucial.

Recall (Sensitivity or True Positive Rate)
Recall measures the proportion of true positives out of all actual positive instances. It reflects the model's ability to identify positive cases.

When to Use: Recall is vital when the cost of false negatives is significant. For example, in medical diagnoses, it is crucial to minimize missed disease cases.
Limitation: High recall can sometimes lead to lower precision, resulting in more false positives.

F1 Score

The F1 Score is the harmonic mean of precision and recall, providing a balanced measure of both metrics.

When to Use: The F1 Score is beneficial when you need a balance between precision and recall, particularly in imbalanced datasets.

Limitation: The F1 Score may not capture the complete picture if the importance of precision and recall varies depending on the specific problem.

Choosing the Best Metric Based on the Problem

- **Balanced Datasets:** If your dataset is balanced and the costs of false positives and false negatives are similar, accuracy is a suitable metric. For example, predicting whether an email is read or not in a balanced dataset.
- **Imbalanced Datasets:** In cases of imbalanced datasets, precision, recall, or F1 Score may be more appropriate. Accuracy can be misleading in these scenarios, such as in fraud detection, where positive cases (fraud) are much fewer than negative cases (non-fraud).
- **High Cost of False Positives:** When false positives are more costly, precision should be prioritized. For instance, in email spam detection, marking a legitimate email as spam can have negative consequences.
- **High Cost of False Negatives:** When false negatives are more costly, recall becomes the key metric. For example, in cancer detection, failing to identify a positive case can have serious implications.
- **Balancing Precision and Recall:** If both false positives and false negatives are important, the F1 Score is the best choice. For example, in information retrieval systems, it is crucial to balance retrieving relevant documents (high precision) while not missing relevant documents (high recall).

Trade-offs Between Metrics

- **Precision vs. Recall:** Increasing precision often decreases recall and vice versa. You need to decide which error (false positive or false negative) is more acceptable for your problem. For instance, in an anti-fraud system, tightening rules to increase precision might reduce recall, leading to fewer actual fraud cases being caught.
- **Accuracy vs. F1 Score:** Accuracy might be high in imbalanced datasets, but the F1 Score provides a clearer understanding of the model's performance in those scenarios. For example, in a dataset where 95% of instances belong to one class, high accuracy may only reflect the model's ability to predict the majority class rather than its effectiveness in distinguishing between classes.

5. Decision Tree Regressor

Definition and Significance

A Decision Tree Regressor is an algorithm in machine learning that is utilized for regression purposes. The main objective is to forecast a continuous value by analyzing input features. This model functions by dividing the data into subsets according to feature values, forming a tree structure with each node indicating a decision point. Traversing the tree from the root to a leaf node determines the final output, with the predicted value usually being the average of the target values at that leaf.

The importance of Decision Tree Regression lies in its ability to be easily understood and its adaptability. It is capable of processing numerical and categorical data, making it appropriate for various purposes. Moreover, it simplifies the modeling process by not necessitating thorough data preprocessing like normalization or scaling.

How Decision Tree Regression Works

The Decision Tree Regressor operates through a series of steps:

1. **Splitting the Data:** The algorithm starts by choosing the top feature and threshold to divide the dataset into two subsets. The objective is to reduce the variability (or increase the homogeneity) in each subset. This procedure is iteratively done for every subset, forming a tree-like arrangement.
2. **Choosing the Best Split:** The algorithm evaluates potential splits using metrics like mean squared error (MSE) to determine which feature and threshold combination results in the most significant reduction in variance.
3. **Stopping Criteria:** The recursion continues until a stopping criterion is met, such as reaching a predefined maximum depth, having a minimum number of samples in a leaf node, or achieving a variance reduction below a certain threshold.
4. **Making Predictions:** Once the tree is constructed, predictions for new instances are made by following the path from the root to a leaf node, where the predicted value is the average of the target values of the training instances in that leaf.

Applications of Decision Tree Regressor

Decision Tree Regressors are widely used across various fields due to their versatility:

- **Finance:** For predicting stock prices, assessing loan risks, or estimating property values based on various features.
- **Healthcare:** To forecast patient outcomes based on medical history and other relevant factors.
- **Marketing:** For predicting customer spending behavior or estimating the effectiveness of marketing campaigns.
- **Environmental Science:** To model and predict environmental phenomena, such as pollution levels or climate changes based on various indicators.

6. In-Depth Mathematical Intuition for Decision Tree Regressor

Variance Reduction

In a Decision Tree Regressor, the goal is to predict a continuous target variable. The splitting criteria for regression tasks focus on reducing the variance in the target variable within the resulting nodes after each split. Variance measures the spread of data points around the mean. When a split is made, the goal is to reduce the variance within the subgroups, making the target values within each node more homogeneous.
For a dataset D with n data points and target values y1,y2,...,yn, the variance Var(D is calculated as:

Var(D)=1/nΣi=1n(yi-y¯)^2

where:
- y¯ is the mean of the target values in D,
- yi is the target value for the i-th data point.

When a split is made based on a feature, the variance of the target values in each resulting node is calculated. The goal is to find the split that leads to the greatest reduction in variance. The **Variance Reduction** for a split on attribute A is given by:

Variance Reduction(D,A)=Var(D)-(|D1|/|D|Var(D1)+|D2|/|D|Var(D2))

where:
- D1 and D2 are the subsets of D created by the split on attribute A,
- Var(D1) and Var(D1) are the variances of the target values in these subsets.

The split that maximizes the Variance Reduction is chosen, as it leads to more accurate predictions by minimizing the error within each node.

Mean Squared Error (MSE) and Mean Absolute Error (MAE)

Two common metrics used in decision tree regression to measure the quality of the splits and the overall performance of the model are **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE).**

1. **Mean Squared Error (MSE):** MSE is the average of the squared differences between the predicted values and the actual target values. It penalizes larger errors more heavily due to the squaring of differences. The formula for MSE is:
   MSE=1/nΣ(yi-y^i)^2

where:
- n is the number of data points,
- yi is the actual target value,
- y^i is the predicted value.

2. MSE is often used in regression tasks because it is differentiable, which makes it easier to optimize. However, it can be sensitive to outliers due to the squaring of errors.

3. **Mean Absolute Error (MAE):** MAE is the average of the absolute differences between the predicted and actual target values. Unlike MSE, it treats all errors equally, making it less sensitive to outliers. The formula for MAE is:
   MAE=1/nΣ|yi-y^i|
   MAE provides a more robust measure of error when the dataset contains outliers, as it does not exaggerate their impact.

Both MSE and MAE can be used to evaluate the performance of a decision tree regressor. While MSE is commonly used in practice due to its mathematical properties, MAE can be a better choice when dealing with datasets that have outliers.

Tree Pruning in Regression

Tree trimming in regression tasks shares the same principles as classification tasks, but with an emphasis on reducing prediction errors like MSE or MAE.

Pruning is necessary in order to avoid overfitting, which happens when the decision tree becomes excessively intricate and begins to capture noise present in the training data. Pruning in regression trees is the process of decreasing the tree's depth by eliminating branches that have a minimal impact on reducing error when predicting unseen data.

There exist two primary forms of pruning:

1. Preventing overfitting by stopping the growth of the decision tree before it becomes too complex:Restricts the tree's expansion by setting limitations, like a maximum depth or minimum samples needed for a split.Preventing the tree from becoming too large and fitting the training data too closely can be achieved through pre-pruning, but it necessitates precise adjustment of the stopping criteria.

2. Pruning after training:Requires growing the entire tree and then trimming the nodes that do not greatly reduce error on a validation set.This method may be more efficient by enabling the tree to analyze intricate splits and later streamlining the model through the elimination of redundant branches.