Decision Trees and Random Forests

Dit-Yan Yeung

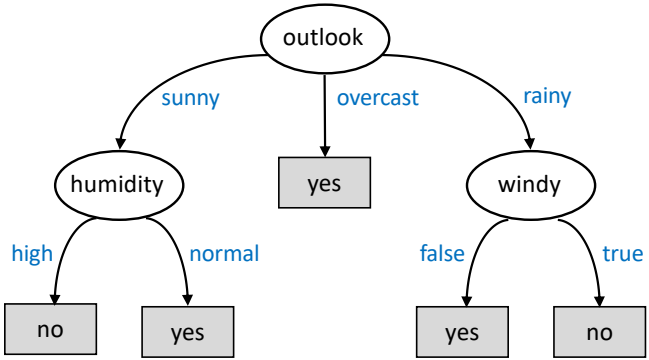
Department of Computer Science and Engineering
Hong Kong University of Science and Technology

COMP 4211: Machine Learning (Fall 2022)

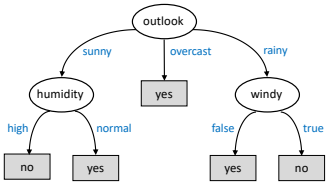
A Sample Dataset

Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

A Decision Tree Example



A Decision Tree Example (2)



- Binary classification: decide whether to play tennis by considering the weather conditions (outlook, humidity, windy).
- Decision rule corresponding to the decision tree classifier:

  if ('outlook is sunny' AND 'humidity is normal') OR
  'outlook is overcast' OR
  ('outlook is rainy' AND 'it is not windy') **then**
  play tennis
  else
  do not play tennis
  end if

Decision Trees

- Decision trees can be used for both classification and regression applications, but we will only consider their use for classification here.
- Two types of nodes in a decision tree:
  - (Internal) decision node: a test function with discrete values for labeling the branches.
  - (Terminal) leaf node: the value associated with it constitutes the output (i.e., a class label for classification).
- Decision tree learning is a recursive tree construction procedure following the divide-and-conquer approach.
- Advantages of decision trees:
  - Fast localization of the region covering the input as a result of hierarchical placement of the decision nodes.
  - High interpretability: can be converted easily into decision rules that can be interpreted easily.

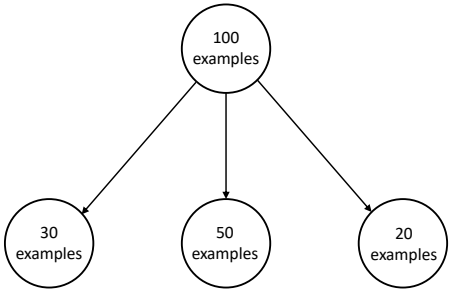
Decision Tree Learning Algorithm

function dtree(*examples*, *attributes*, *default*) **returns** a decision tree
if *examples* is empty **then**
  return *default*
else if all *examples* have the same class label **then**
  return the class label
else if *attributes* is empty **then**
  return majority_class(*examples*)
else
  best $\leftarrow$ choose_attribute(*attributes*, *examples*)
  tree $\leftarrow$ a new decision tree with root test *best*
  for each value v_i of *best* **do**
    $examples_i \leftarrow$ {elements of *examples* with $best = v_i$}
    $m \leftarrow$ majority_class($examples_i$)
    $subtree \leftarrow$ dtree($examples_i$, $attributes - best$, m)
    add a branch to *tree* with label v_i and subtree *subtree*
  end for
  return *tree*
end if

How to Choose the Best Attribute?

- Let there be K classes.
  - A node is said to be most impure if the examples associated with the node are distributed uniformly among the K classes.
  - A node is said to be least impure (or purest) if all the examples associated with the node belong to the same class.
- Intuitively, a good attribute is one which expands an impure node to give purer nodes. The one which leads to the maximum reduction in impurity is the best attribute.

An Illustrative Example



- Let $\mathcal{I}, \mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3$ denote the values of some impurity measure for the four nodes above.
- Expected reduction in impurity (a.k.a. information gain):

$$\mathcal{I} - \left(\frac{30}{100} \mathcal{I}_1 + \frac{50}{100} \mathcal{I}_2 + \frac{20}{100} \mathcal{I}_3 \right).$$

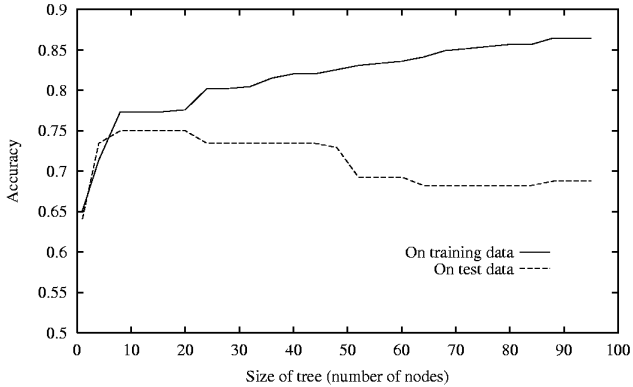
Entropy

- A common randomness or impurity measure is entropy, borrowed from physics and information theory.
- In a node, let p_i denote the fraction of examples which belong to class i, for $1 \leq i \leq K$. We assume that $\sum_{i=1}^{K} p_i = 1$.
- Entropy of a node:

$$-\sum_{i=1}^{K} p_i \log_2 p_i.$$

  In case $p_j = 0$ for some j, we set the term $p_j \log_2 p_j = 0$ in the summation.

Overfitting



- As a decision tree gets deeper, fewer examples are associated with the deeper nodes, often leading to overfitting with large variance.

Decision Tree Ensembles

- A decision tree ensemble combines multiple decision trees to produce better predictive performance than utilizing a single decision tree.
- The final prediction is made by averaging the predictions of the individual decision trees.
- As an ensemble method, it is based on the principle that a group of weak learners come together to form a strong learner with lower variance.
- An ensemble is more effective if the decision trees involved are more independent of each other.
- Each decision tree in an ensemble is trained using a variant of the training set. Two common approaches:
  - Bagging
  - Random Forests

Bagging

- Bagging is the short form for bootstrap aggregating, which uses the bootstrapping process to generate multiple bootstrap samples for training the decision trees in an ensemble.
- From the original training set of N examples, bootstrapping generates multiple bootstrap samples, each of which is also of size N, by drawing examples randomly from the original set with replacement.
- In general, there exist duplicate examples in each bootstrap sample.
- Also, the bootstrap samples are quite dependent. Consequently, some attributes may be chosen for the decision nodes by most if not all of the decision trees, making the decision trees also dependent.

Random Forests

- Random forests alleviate the problem of bagging by making the decision trees in an ensemble less dependent.
- The decision trees in a random forest are like ordinary decision trees with only one exception in the learning algorithm: each decision node is created by selecting from a random subset of the attributes (instead of all the attributes), called attribute bagging.
- Attribute bagging prevents a few effective attributes from being used by many decision trees, making them less dependent.
- The number of randomly selected attributes that can be searched at each decision node is a hyperparameter of the random forest learning algorithm. For example, it may be set to the square root of the total number of available attributes.
- Like bagging, different decision trees are trained using different bootstrap samples.

To Learn More...

- Gradient boosting trees