THE UNIVERSITY *of* EDINBURGH
informatics

Applied Machine Learning (AML)

Decision Trees

Oisin Mac Aodha • Siddharth N.

Decision Trees

Nonlinear Data

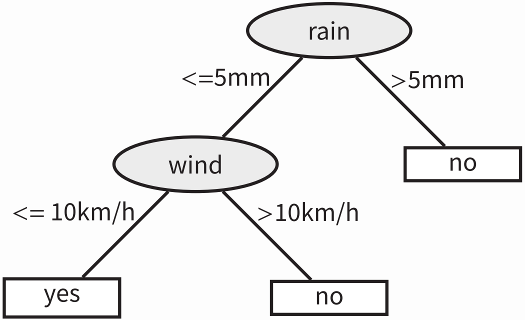
- Linear classifiers are not capable of separating *nonlinear* data
- Many real world problems of interest may not necessarily have linearly separable data

Nonlinear Data

- Linear classifiers are not capable of separating *nonlinear* data
- Many real world problems of interest may not necessarily have linearly separable data
- Decision trees are a popular approach for nonlinear **classification** and **regression**
- They operate by recursively partitioning the input feature space and then defining local models in each of the resulting regions

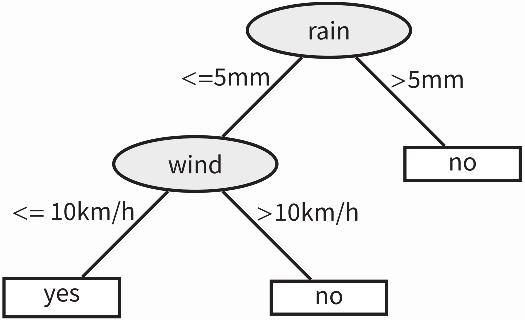
Decision Tree Example

Should I go for a walk?



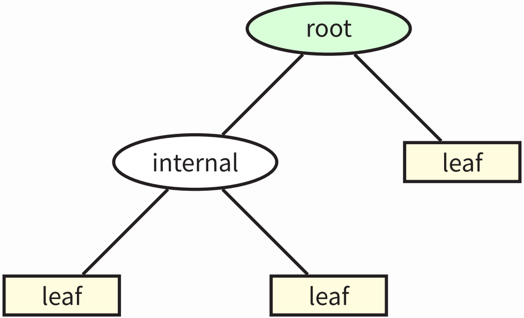
Advantages of Decision Trees

- Intuitive
- Efficient
- Nonlinear
- General -
  - Classification
  - Regression
- Can handle mixed data types



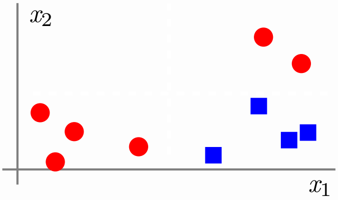
Tree Terminology

- There are three main types of nodes in a tree: root, internal, and leaves
- Each non-leaf node is a parent, and has a left and right child



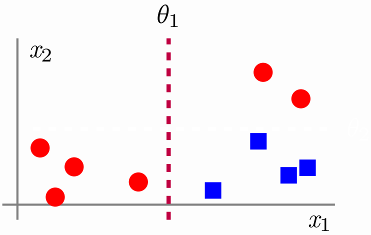
2D Example

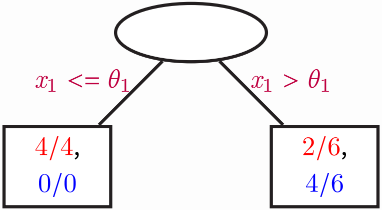
- In this example, we have 10 2D datapoints, i.e. $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{10}\}$, where $\boldsymbol{x} \in \mathbb{R}^2$
- We have six red (y=1) and four blue (y=2) datapoints



2D Example

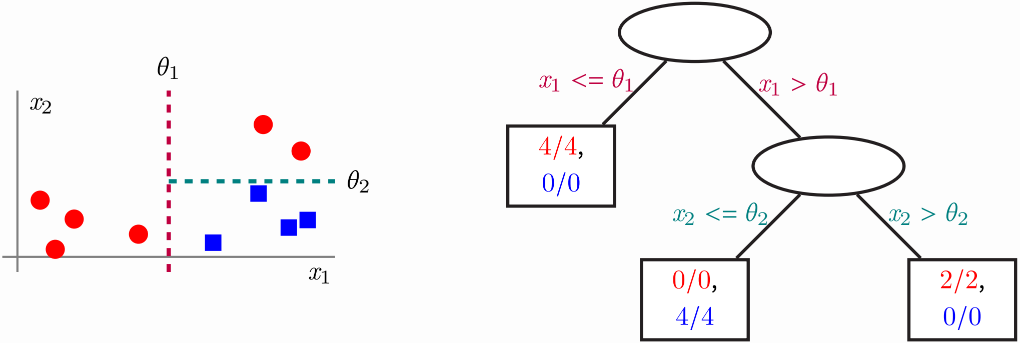
- At each node, we split the data based on a feature dimension and threshold, here θ_1
- Then we store the percentage of examples from each class (p_c) at the leaves





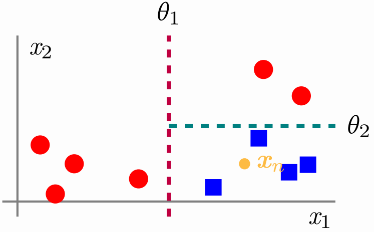
2D Example

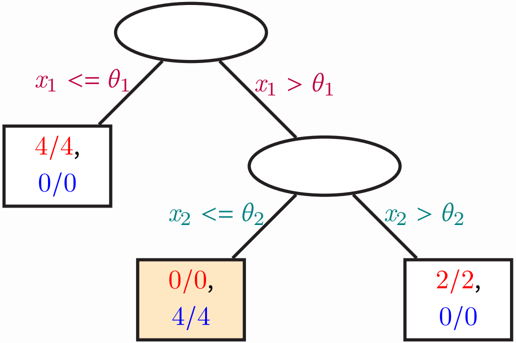
- We can keep splitting the tree until we reach some predefined stopping criteria
- Note, we split a feature dimension multiple times



2D Example - Evaluating a Test Datapoint

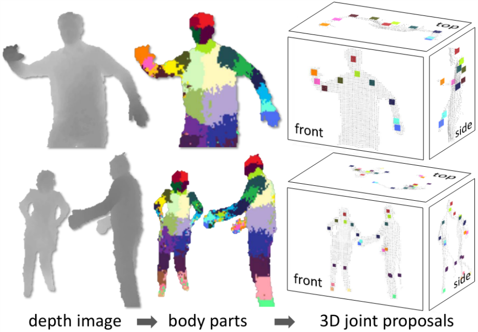
- How can we predict the class label of a new example x_n?
- We simply evaluate each relevant node to find the leaf that contains it





Applications of Decision Trees

- Due to their speed and performance, decision trees have been applied to many different tasks



Human body pose estimation using decision trees from Shotton et al. *CVPR 2011*.

Fitting Decision Trees

Decision Tree Learning

- Start with all the data at the root node of the tree
- Grow the tree by recursively splitting the data at each node
- Keeping growing until you reach a specified condition, e.g. the tree reaches a predefined maximum depth or it is not possible to split the data any further
- Different methods have been proposed over the years, e.g. CART, ID3, ...

Measuring the Quality of a Split

- How do we determine what threshold and feature dimension to use at each node in the tree?
- We should favour splits that result in child nodes that have high ‘purity’, i.e. low ‘impurity’

Measuring the Quality of a Split

- How do we determine what threshold and feature dimension to use at each node in the tree?
- We should favour splits that result in child nodes that have high 'purity', i.e. low 'impurity'
- One common approach for classification is to measure the entropy at each node
  - The entropy of a random variable is the average level of 'information', 'surprise', or 'uncertainty' inherent to the variable's possible outcomes

$$I_E(S) = -\sum_{c=1}^{C} p_c \log_2 p_c$$

Evaluating Entropy

Entropy can be computed using the distribution of datapoints at a given node.

$$I_E(S) = -\sum_{c=1}^{C} p_c \log_2 p_c$$

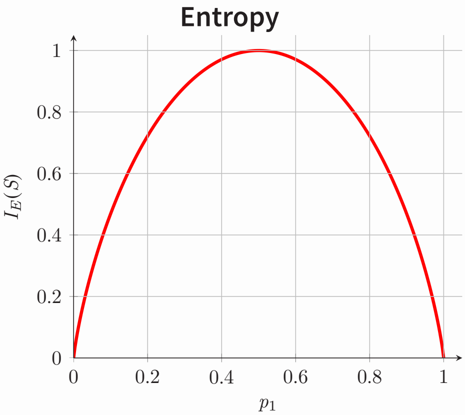
- C is the number of classes in the dataset, i.e. $y \in \{1, ..., C\}$
- S is the subset of datapoints that have arrived at the node, where $S \subseteq \{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$
- p_c is the proportion of examples from class c that are present at the node, where $p_c \in [0, 1]$

Entropy

We have low entropy when most, if not all, the datapoints at a node are from the same class.

$$I_E(S) = -\sum_{c=1}^{C} p_c \log_2 p_c$$

Note, that the expression for entropy is often also notated as $H(S)$.



Binary case:

$$I_E(S) = -p_1 \log_2 p_1 - p_2 \log_2 p_2$$

Alternative Splitting Criteria

There are alternative splitting criteria, e.g. Gini Impurity.

$$I_G(S) = 1 - \sum_{c=1}^{C} p_c^2$$

Binary case:

$$I_G(S) = 1 - p_1^2 - p_2^2$$

Information Gain

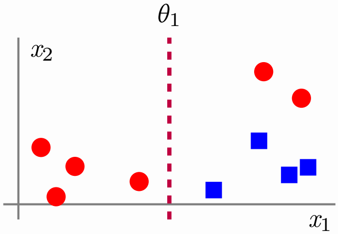
- Now that we can measure the purity at each node in a tree, we can use this to determine the quality of different splits
- We do this measuring the Information Gain of a split

$$Gain(S, \theta, d) = I(S) - \left(\frac{|S_l|}{|S|} I(S_l) + \frac{|S_r|}{|S|} I(S_r) \right)$$

Here $|S| = |S_l| + |S_r|$

Information Gain

- Now that we can measure the purity at each node in a tree, we can use this to determine the quality of different splits
- We do this measuring the Information Gain of a split



$|S| = 10$

$|S_l| = 4$

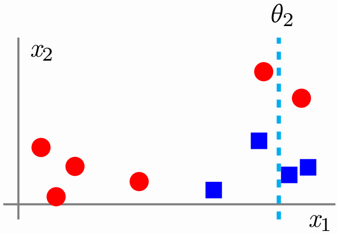
$|S_r| = 6$

$$Gain(S, \theta, d) = I(S) - \left(\frac{|S_l|}{|S|} I(S_l) + \frac{|S_r|}{|S|} I(S_r) \right)$$

Here $|S| = |S_l| + |S_r|$

Information Gain

- Now that we can measure the purity at each node in a tree, we can use this to determine the quality of different splits
- We do this measuring the Information Gain of a split



$|S| = 10$

$|S_l| = 7$

$|S_r| = 3$

$$Gain(S, \theta, d) = I(S) - \left(\frac{|S_l|}{|S|} I(S_l) + \frac{|S_r|}{|S|} I(S_r) \right)$$

Here $|S| = |S_l| + |S_r|$

Different splits will result in different Information Gain

Choosing the Best Split

- Evaluate the Information Gain for each feature dimension and threshold pair at a given node
- Choose the pair with the largest gain

Choosing the Best Split

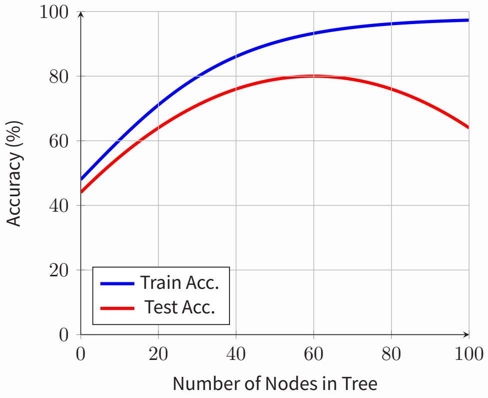
- Evaluate the Information Gain for each feature dimension and threshold pair at a given node
- Choose the pair with the largest gain
- If trying all combinations is impractical, one can choose the best pair from a random subset

Stopping Criteria

- A tree can always classify training examples perfectly, i.e.
  - Keep splitting each node until there is only one example at each leaf
  - These ‘singleton’ nodes will be pure
- This will result in *overfitting* to the training data, i.e. the model will not generalise well to new data

Avoiding Overfitting

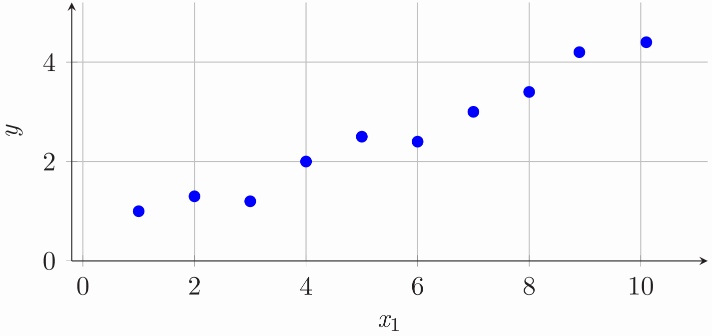
- Introduce an additional hyperparameter
  - Maximum tree depth
  - Minimum number of datapoints per node
  - Minimum information gain
- Grow the tree to full depth, and then ‘prune’ it



Additional Topics

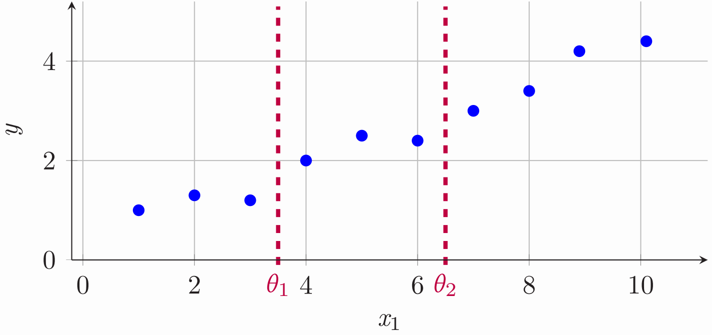
Regression Trees

- We can also model continuous targets using regression trees, i.e. $y \in \mathbb{R}$
- The tree models data locally as a piece-wise constant function, where it stores a different mean value $\bar{y}_i$ at each leaf node



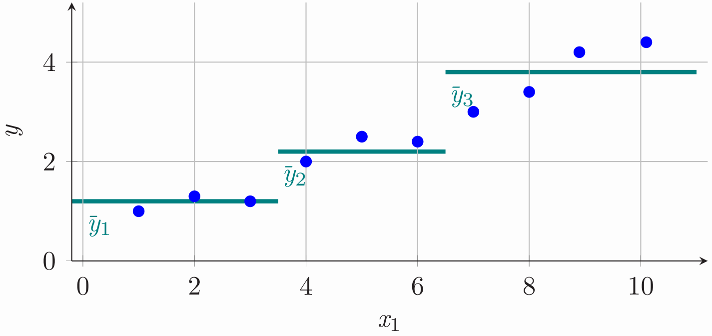
Regression Trees

- We can also model continuous targets using regression trees, i.e. $y \in \mathbb{R}$
- The tree models data locally as a piece-wise constant function, where it stores a different mean value $\bar{y}_i$ at each leaf node



Regression Trees

- We can also model continuous targets using regression trees, i.e. $y \in \mathbb{R}$
- The tree models data locally as a piece-wise constant function, where it stores a different mean value $\bar{y}_i$ at each leaf node



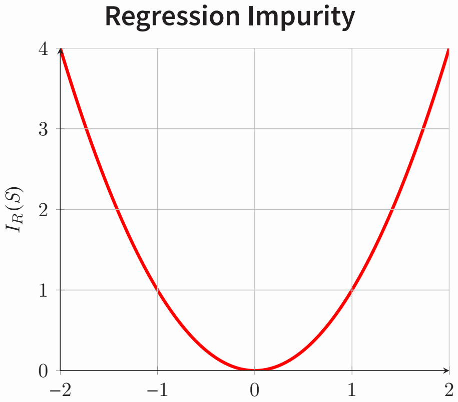
Regression Criteria

- In the case of regression, our ground truth targets are continuous values
- As a result, we require a different definition of node purity

$$I_R(S) = \frac{1}{|S|} \sum_{y \in S} (y - \bar{y})^2$$

- At each leaf we store the mean of all the datapoints that arrived at the node

$$\bar{y} = \frac{1}{|S|} \sum_{y \in S} y$$



Discrete Features

- Decision trees can handle both continuous or discrete (i.e. categorical) features
- In practice, popular implementations may not support natively
- For non-ordinal categorical variables it is possible to transform them using a *one-hot* encoding

Trees are Interpretable

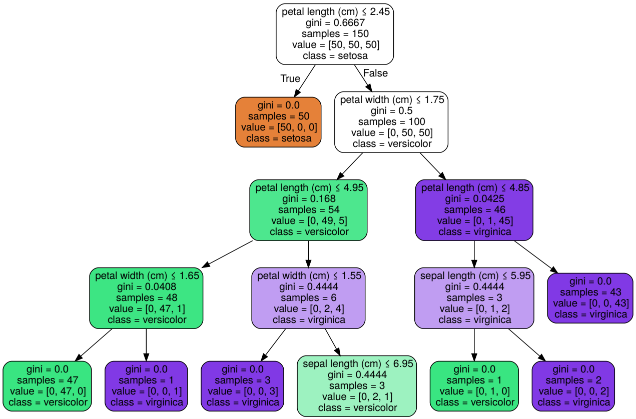


Image credit: https://scikit-learn.org/stable/modules/tree.html

Ensembles of Trees

- Grow an ensemble of K different decision trees:
  - Pick a random subset of the data
  - Train a decision tree on this data
    - When splitting, choose a random subset of features
  - Repeat this K different times

Ensembles of Trees

- Grow an ensemble of K different decision trees:
  - Pick a random subset of the data
  - Train a decision tree on this data
    - When splitting, choose a random subset of features
  - Repeat this K different times
- Given a new datapoint x at test time:
  - Classify x separately using each tree
  - Combine the predictions from each individual tree for the final output, e.g. using the majority vote
- Simple, but can be very effective