Decision trees

Machine Learning Techniques

Dr. Ashish Tendulkar

IIT Madras

Decision trees are non-parametric supervised learning methods that can be used for modelling classification as well as regression problems.

We will first study binary decision trees in classification setting.

Binary decision tree

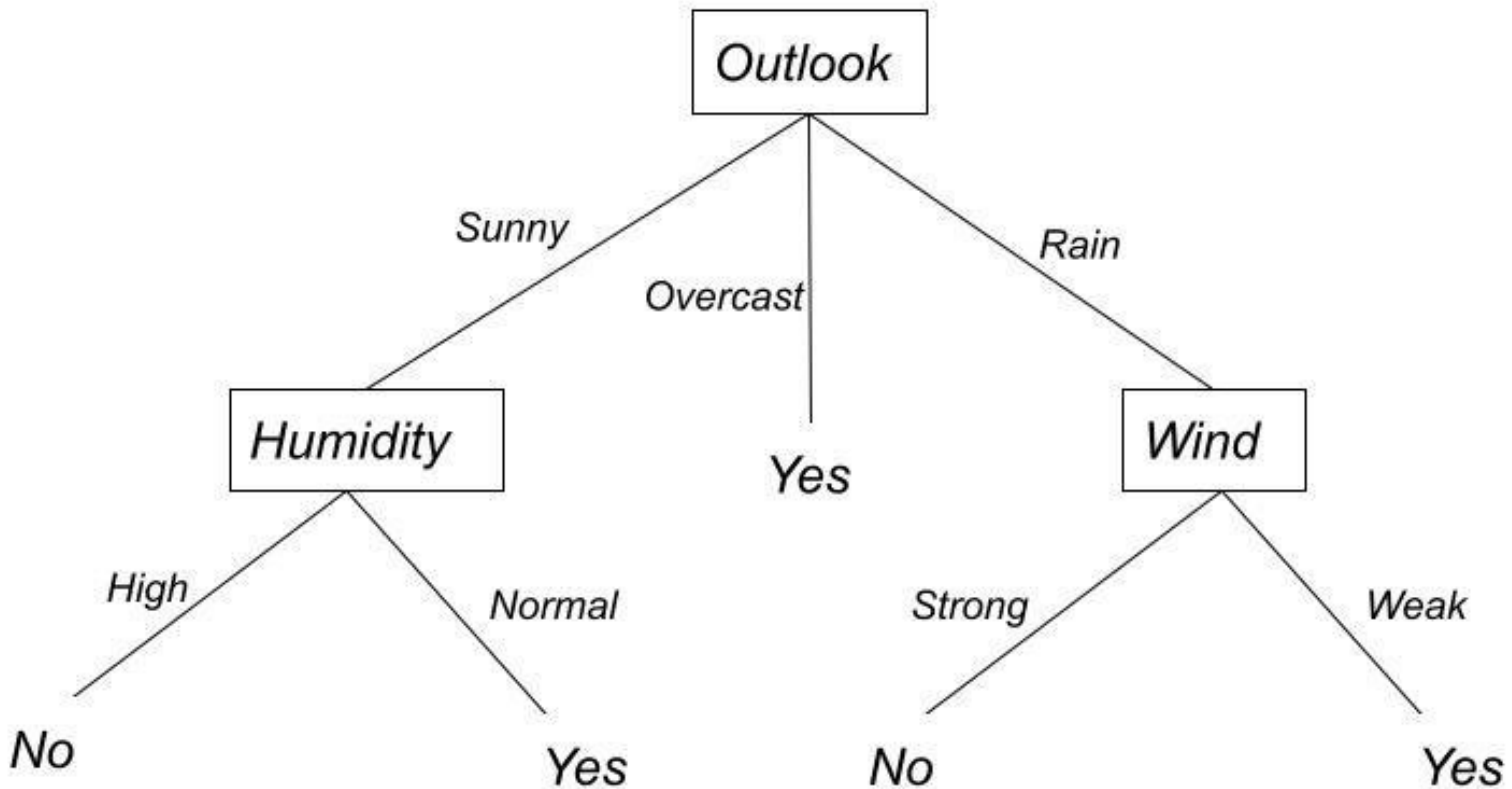
Tree-based methods partition the feature space into a set of rectangles (or cuboids) and then fit a simple model (like a constant) in each one. They are conceptually simple yet powerful.

Binary tree splits into two branches at each node.

Consider an example of binary classification problem with four features.

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Decision tree for the same dataset may look like



- Which attribute to choose for splitting?
- What should be the splitting criterion?
- What tree size will give the optimal solution?

ID3 Algorithm for Decision Trees

Data

A set of n ordered pairs of a feature vector, $\mathbf{x}$ and a label y representing examples. We denote it by D:

$$D = \{(\mathbf{X}, \mathbf{y})\} = \left\{(\mathbf{x}^{(i)}, y^{(i)})\right\}_{i=1}^{n}$$

The training data consists of input feature vectors $\mathbf{x^{(1)}}, \mathbf{x^{(2)}}, \mathbf{...}, \mathbf{x^{(n)}}$ along with the corresponding labels $y^{(1)}, y^{(2)}, ..., y^{(n)}$.

For classification problem, $y^{(i)}$ is an element from discrete set.

For regression problem, $y^{(i)}$ is a real number.

Model

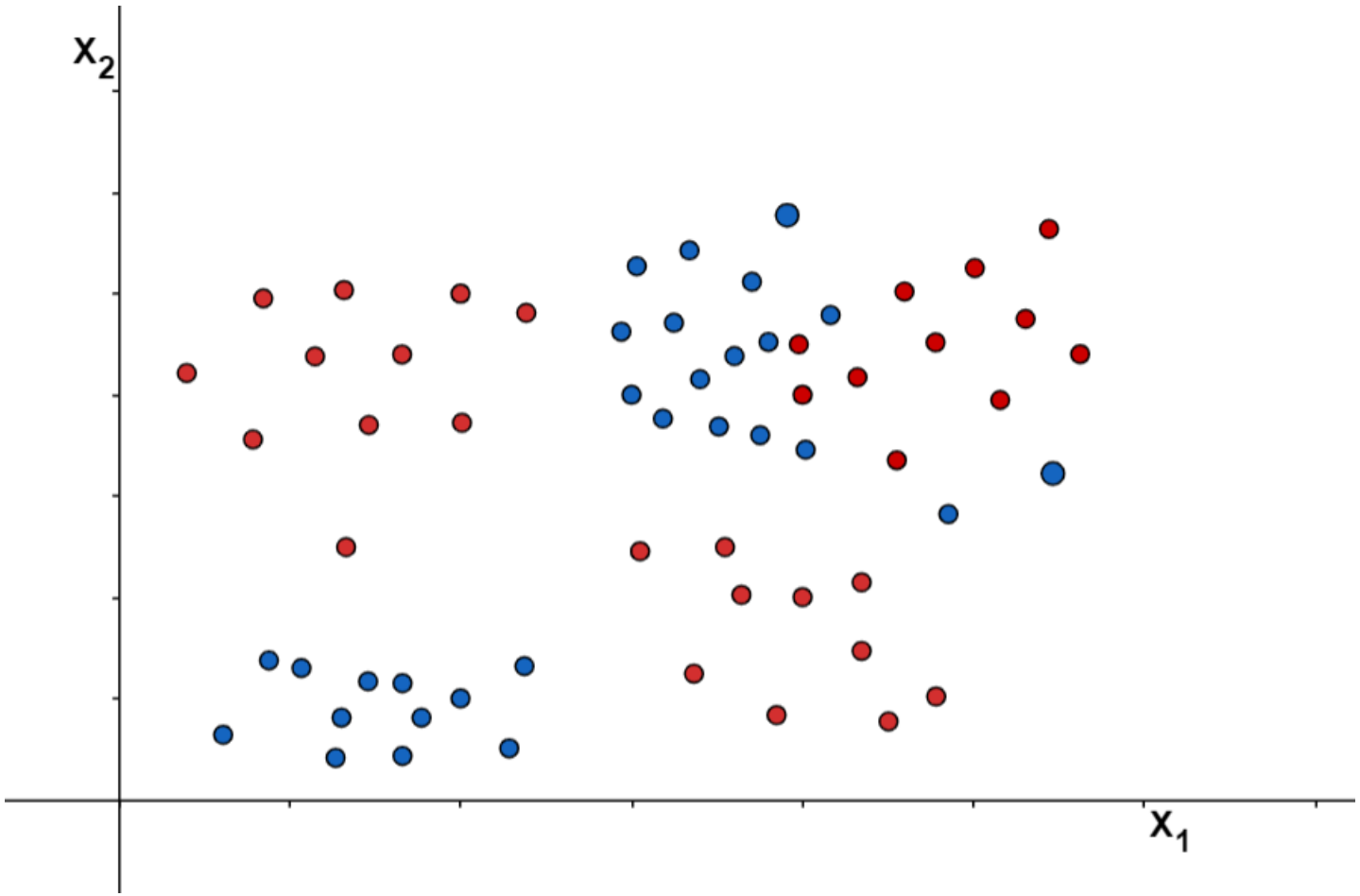
A variety of algorithms have been developed for learning decision trees.
Most of these algorithms are variations of a core algorithm known as the ID3 algorithm (Iterative Dichotomizer 3).

- It employs a top-down, greedy search through the space of possible decision trees.

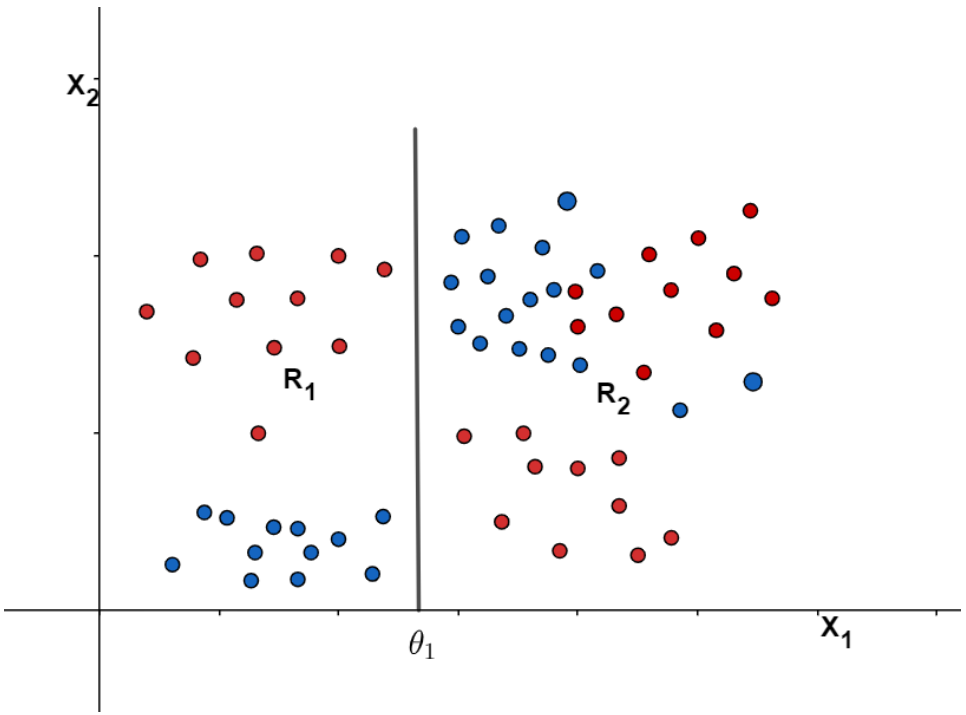
ID3 learns decision trees by beginning with the question "which attribute should be tested at the root of the tree?"

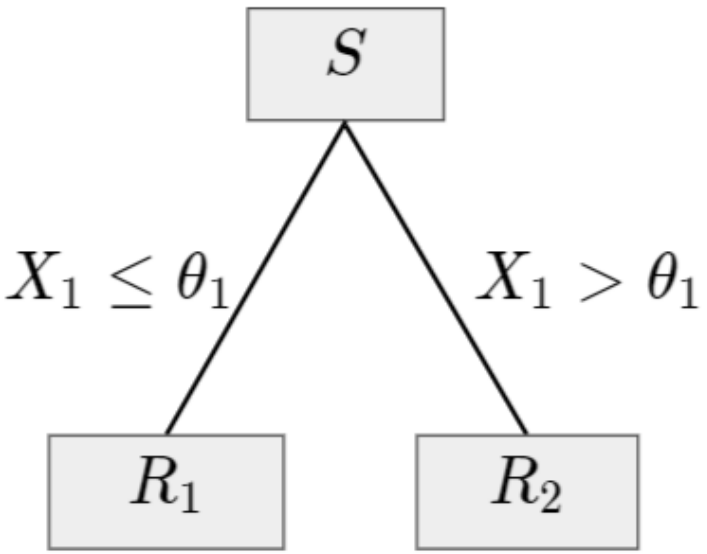
To answer this question, each attribute is evaluated using some measure to determine how well it alone classifies the training examples.

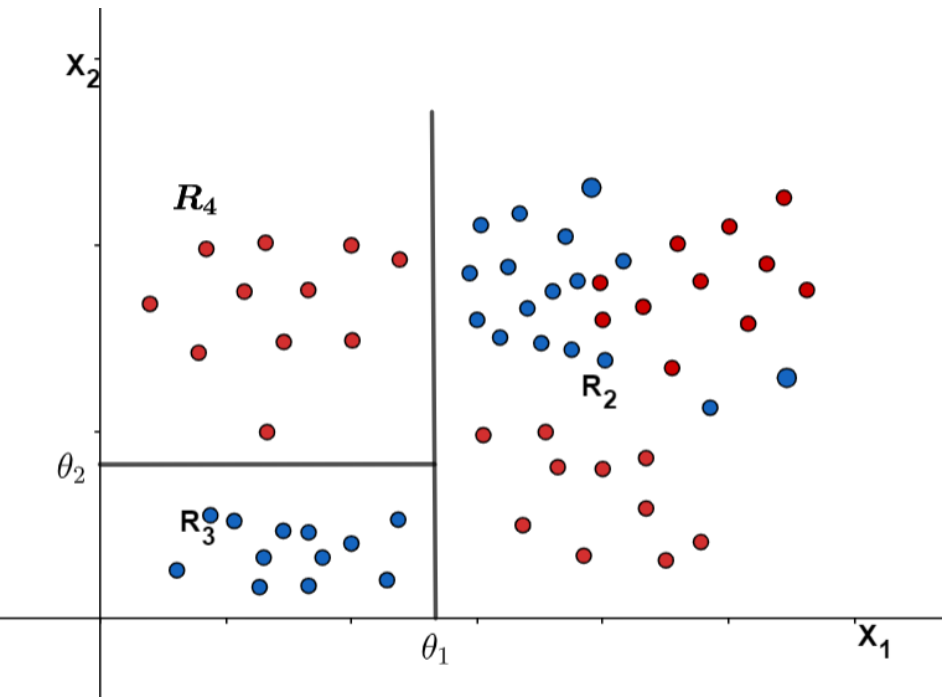
Consider a Classification problem with label y and features X_1 and X_2.

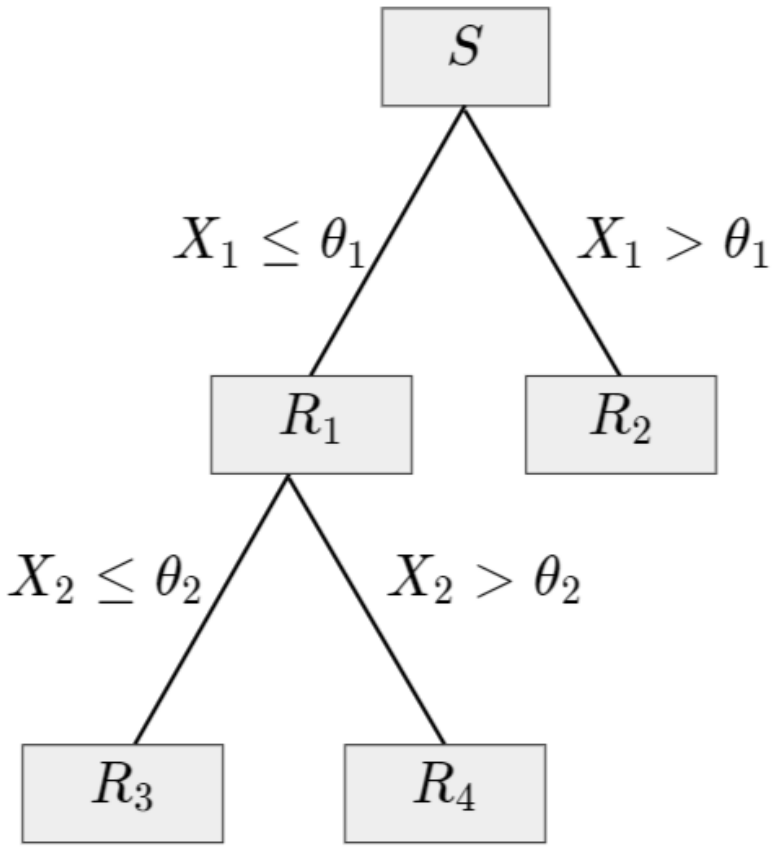


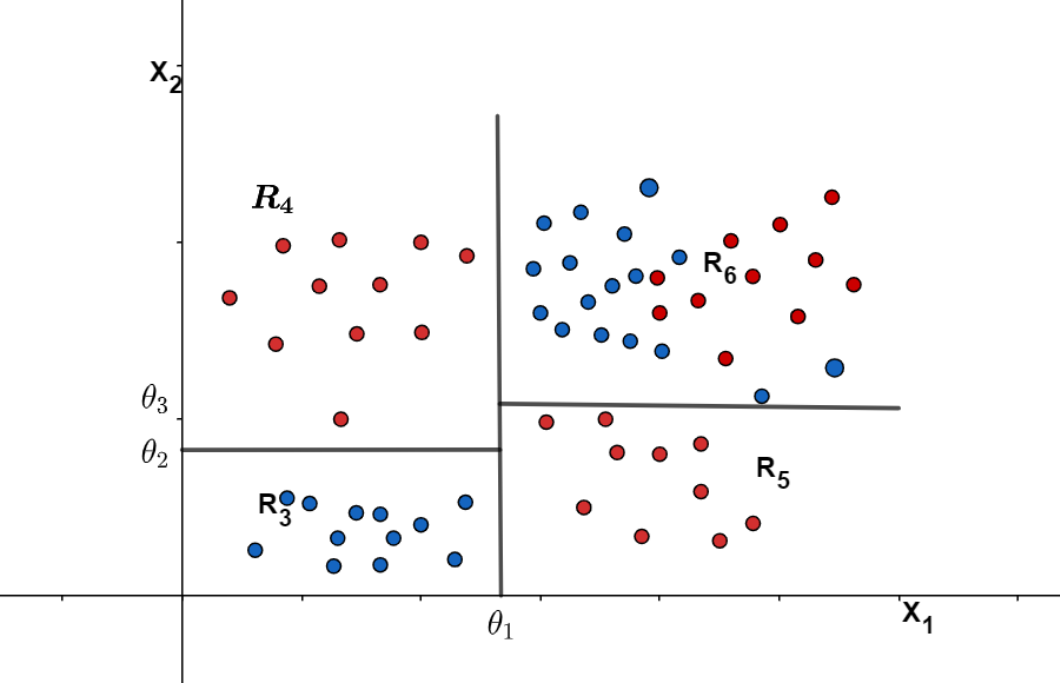
If we split the dataset according to the feature X_1, then

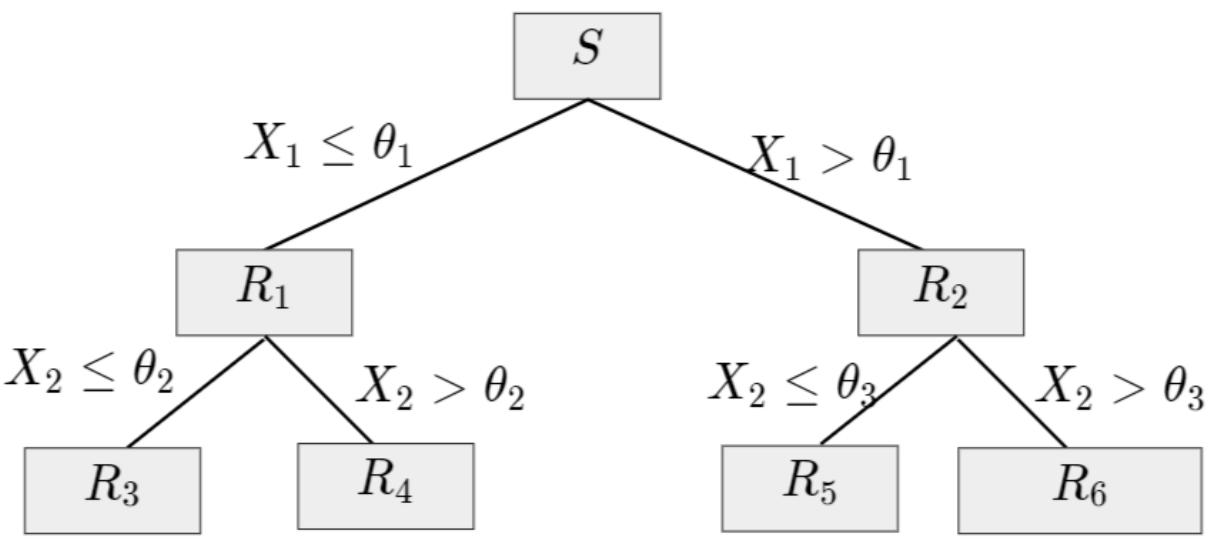


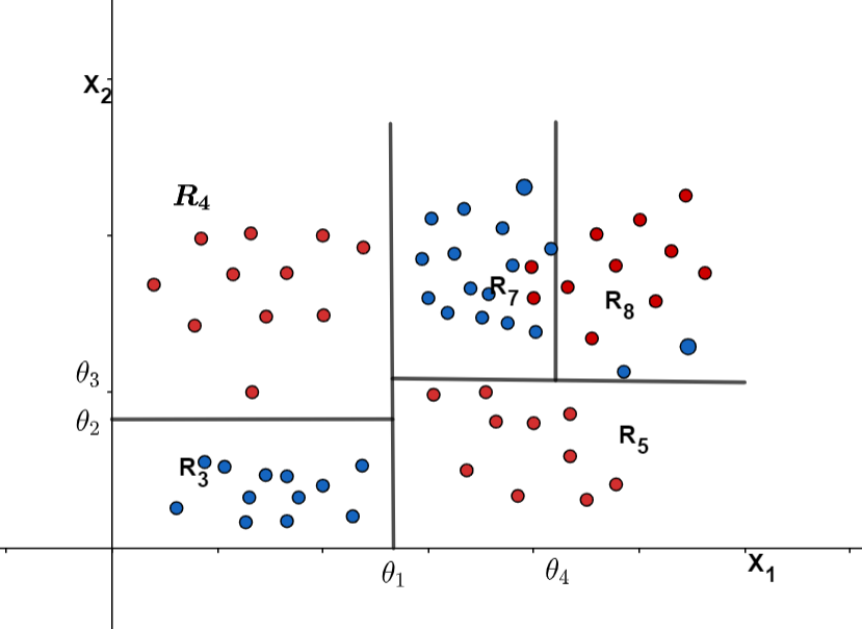


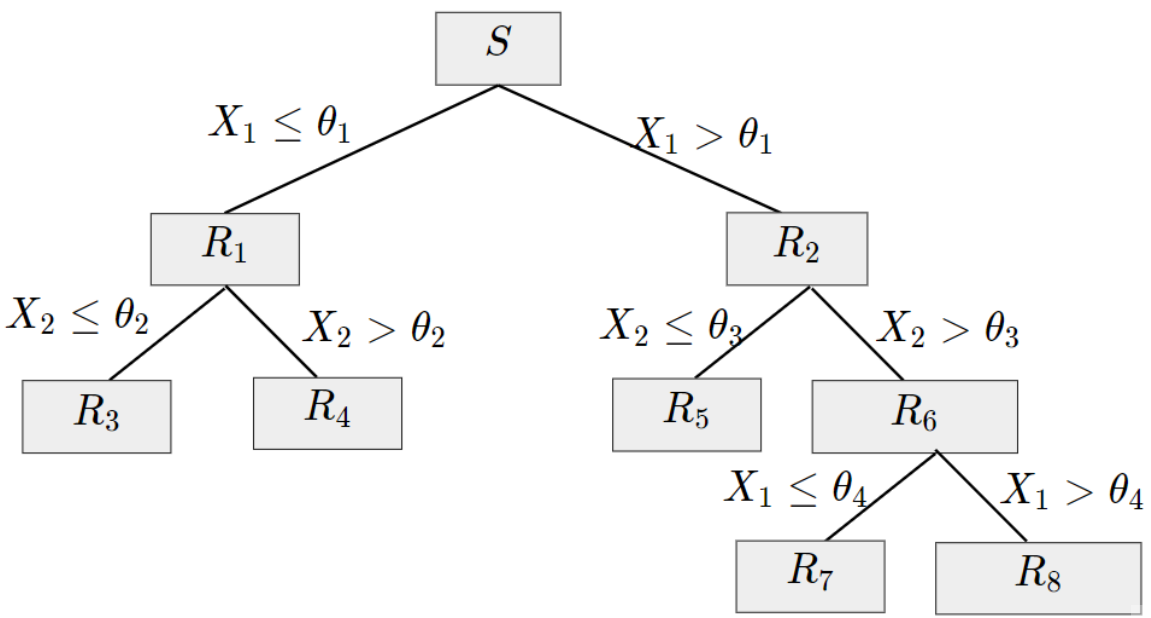

X2
R4
R2
θ2
R3
θ1
X1


S
$X_1 \leq \theta_1$
$X_1 > \theta_1$
R_1
R_2
$X_2 \leq \theta_2$
$X_2 > \theta_2$
R_3
R_4


X2
R4
R6
θ3
θ2
R3
R5
θ1
X1


S
X1 ≤ θ1
X1 > θ1
R1
R2
X2 ≤ θ2
X2 > θ2
X2 ≤ θ3
X2 > θ3
R3
R4
R5
R6


X2
R4
R7
R8
θ3
θ2
R5
R3
θ1
θ4
X1


S
X1 ≤ θ1
X1 > θ1
R1
R2
X2 ≤ θ2
X2 > θ2
X2 ≤ θ3
X2 > θ3
R3
R4
R5
R6
X1 ≤ θ4
X1 > θ4
R7
R8

Training Process

The best attribute is selected and used for splitting at the root node of the tree.

A descendant of the root node is then created for each possible value of this attribute and the training examples are assigned to the appropriate descendant.

The entire process is then repeated using the training examples associated with each descendant node to select the best attribute for splitting at that point in the tree.

- This forms a greedy search for an acceptable decision tree, in which the algorithm never backtracks to reconsider earlier choices.

Impurity measures

If we define $p_{i,k}$ to be the proportion of data points in node i (R_i) assigned to class k, where $k = 1, ..., K$ then

$$p_{i,k} = \frac{1}{N_i} \sum_{x^{(i)} \in R_i} 1(y^{(i)} = k)$$

number of samples in region R_i.

Misclassification error

It is the proportion of misclassified examples in node i.

$$Q_i(T) = 1 - p_{i,k(i)} = \frac{1}{N_i} \sum_{x^{(i)} \in R_i} 1(y^{(i)} \neq k(i))$$

where $k(i) = \arg\max_k p_{i,k}$ = prediction of class in node i

Gini index

$$G_i = \sum_{k=1}^{K} p_{i,k}(1 - p_{i,k})$$

- Rather than classify observations to the majority class in the node, we could classify them to class k with probability $p_{i,k}$, then the training error rate of this rule in the node is the Gini index.
- Similarly, if we code each observation as 1 for class k and zero otherwise, the variance over the node of this 0-1 response is $p_{i,k}(1-p_{i,k})$. Summing over classes again gives the Gini index.

Entropy

Entropy of node i is given by

$$H_i = -\sum_{k=1}^{n} p_{i,k} \log_2 p_{i,k}$$

In ID3, entropy is calculated for each remaining attribute. The attribute with the smallest entropy is used to split the dataset on this iteration.

Information Gain

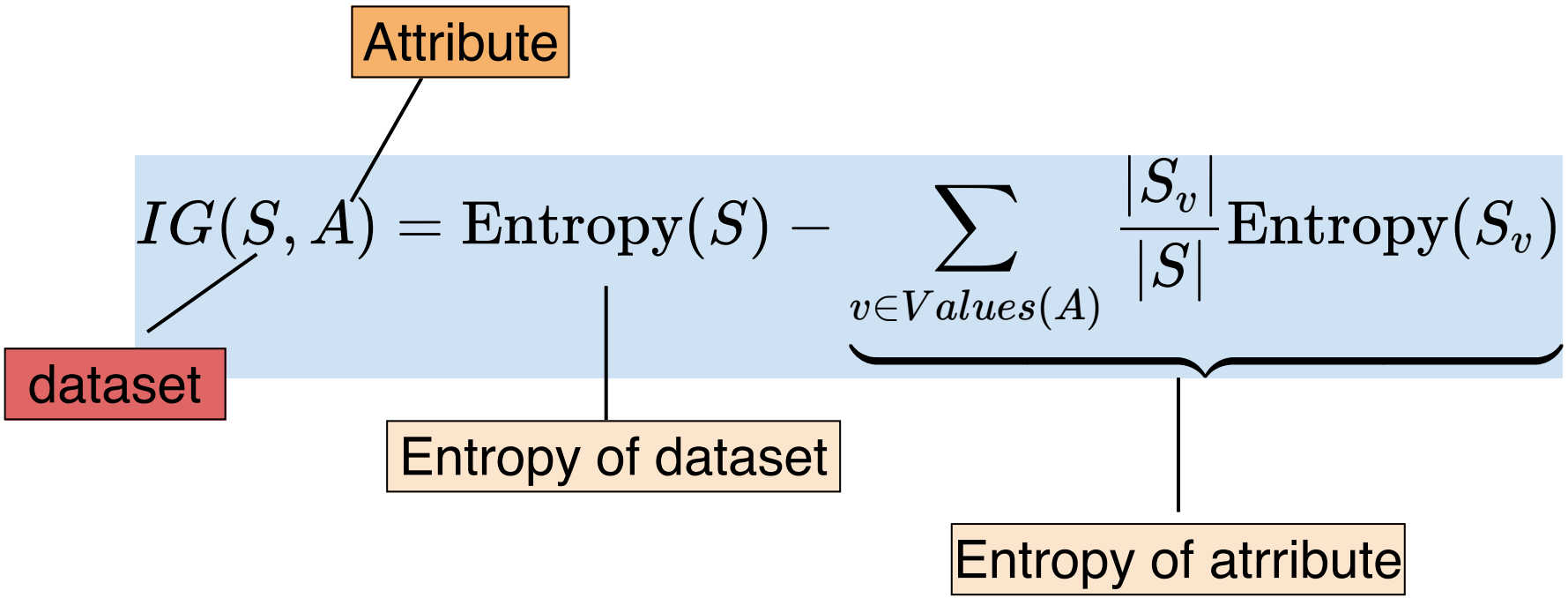
Information gain (IG) is a measure of the effectiveness of an attribute in classifying the training data.

It is simply the expected reduction in the entropy caused by partitioning the examples according to this attribute.

Intuitively,

IG(attr) = entropy of dataset - entropy of attribute

Formally, the information gain, IG(S, A) of an attribute A, relative to a collection of examples S, is defined as



In ID3, information gain can be calculated (instead of entropy) for each remaining attribute. The attribute with the **largest** information gain is used to split the dataset on this iteration.

Stopping criterion

Recursion on a subset may stop in one of these cases:

- Every element in the subset belongs to the same class.
  - In which case the node is turned into a leaf node and labelled with the class of the examples.
- There are no more attributes to be selected, but the examples still do not belong to the same class.
  - In this case, the node is made a leaf node and labelled with the most common classes of the examples in the subset.
- There are no examples in the subset.
  - which happens when no example in the parent set was found to match a specific value of the selected attribute.

Let's try to model a tree on a dataset mentioned earlier:

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

For determining the feature for splitting the data, we calculate the entropy.

Play
No
No
Yes
Yes
Yes
No
Yes
No
Yes
Yes
Yes
Yes
Yes
No

In the above dataset, there are 14 examples, where 9 are positive examples (with a target value of 'Yes') and 5 are negative examples (with a target value of 'No').

The entropy of dataset S will be

$$H(S) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.94$$

Wind	Play
Weak	No
Strong	No
Weak	Yes
Weak	Yes
Weak	Yes
Strong	No
Strong	Yes
Weak	No
Weak	Yes
Weak	Yes
Strong	Yes
Strong	Yes
Weak	Yes
Strong	No

Dividing samples based on Wind, which is either Strong or Weak, then

- S_{weak} has 6 'Yes' and 2 'No'

$$\mathrm{H}(S_{\mathrm{weak}}) = -\left(\frac{6}{8}\log_2\frac{6}{8}\right) - \left(\frac{2}{8}\log_2\frac{2}{8}\right) = 0.811$$

- S_{strong} has 3 'Yes' and 3 'No'

$$\mathrm{H}(S_{strong}) = -\left(\frac{3}{6}\log_2\frac{3}{6}\right) - \left(\frac{3}{6}\log_2\frac{3}{6}\right) = 1$$

$$\mathrm{H}(S_{\text{weak}}) = -\left(\frac{6}{8}\log_2\frac{6}{8}\right) - \left(\frac{2}{8}\log_2\frac{2}{8}\right) = 0.811$$

$$\mathrm{H}(S_{strong}) = -\left(\frac{3}{6}\log_2\frac{3}{6}\right) - \left(\frac{3}{6}\log_2\frac{3}{6}\right) = 1$$

$$IG(S, Wind) = \mathrm{H}(S) - \sum_{v \in \{Weak, Strong\}} \frac{|S_v|}{|S|}\mathrm{H}(S_v)$$

$$IG(S, Wind) = \mathrm{H}(S) - \frac{8}{14}\mathrm{H}(S_{weak}) - \frac{6}{14}\mathrm{H}(S_{strong})$$

$$IG(S, Wind) = 0.940 - \frac{8}{14}(0.811) - \frac{6}{14}(1) = 0.048$$

This means that if we divide the dataset by keeping Wind attribute as the root, we get a reduction of 0.048 in the entropy value.

Outlook	Play
Sunny	No
Sunny	No
Overcast	Yes
Rain	Yes
Rain	Yes
Rain	No
Overcast	Yes
Sunny	No
Sunny	Yes
Rain	Yes
Sunny	Yes
Overcast	Yes
Overcast	Yes
Rain	No

Dividing samples based on Outlook, which one of sunny, overcast, rain, then

- S_{sunny} has 2 'Yes' and 3 'No'

$$\mathrm{H}(S_{\mathrm{sunny}}) = -\left(\frac{2}{5}\log_2\frac{2}{5}\right) - \left(\frac{3}{5}\log_2\frac{3}{5}\right) = 0.971$$

- $S_{overcast}$ has 4 'Yes' and 0 'No'

$$\mathrm{H}(S_{overcast}) = -\left(\frac{4}{4}\log_2\frac{4}{4}\right) - \left(\frac{0}{4}\log_2\frac{0}{4}\right) = 0$$

- S_{rain} has 3 'Yes' and 2 'No'

$$\mathrm{H}(S_{rain}) = -\left(\frac{3}{5}\log_2\frac{3}{5}\right) - \left(\frac{2}{5}\log_2\frac{2}{5}\right) = 0.971$$

$$\mathrm{H}(S_{\text{sunny}}) = -\left(\frac{2}{5}\log_2\frac{2}{5}\right) - \left(\frac{3}{5}\log_2\frac{3}{5}\right) = 0.971$$

$$\mathrm{H}(S_{overcast}) = -\left(\frac{4}{4}\log_2\frac{4}{4}\right) - \left(\frac{0}{4}\log_2\frac{0}{4}\right) = 0$$

$$\mathrm{H}(S_{rain}) = -\left(\frac{3}{5}\log_2\frac{3}{5}\right) - \left(\frac{2}{5}\log_2\frac{2}{5}\right) = 0.971$$

$$IG(S, Outlook) = \mathrm{H}(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|}\mathrm{H}(S_v)$$

$$IG(S, Outlook) = \mathrm{H}(S) - \frac{5}{14}\mathrm{H}(S_{sunny}) - \frac{4}{14}\mathrm{H}(S_{overcast}) - \frac{5}{14}\mathrm{H}(S_{rain})$$

$$IG(S, Outlook) = 0.940 - \frac{5}{14}(0.971) - \frac{4}{14}(0) - \frac{5}{14}(0.971) = 0.246$$

This means that if we divide the dataset by keeping 'Outlook' attribute as the root, we get a reduction of 0.246 in the entropy value.

Temperature	Play
Hot	No
Hot	No
Hot	Yes
Mild	Yes
Cool	Yes
Cool	No
Cool	Yes
Mild	No
Cool	Yes
Mild	Yes
Mild	Yes
Mild	Yes
Hot	Yes
Mild	No

Dividing the samples based on Temperature, which is one of Cold or Mild or Hot, then

- S_{cold} has 3 'Yes' and 1 'No'

$$\mathrm{H}(S_{\mathrm{cold}}) = -\left(\frac{3}{4}\log_2\frac{3}{4}\right) - \left(\frac{1}{4}\log_2\frac{1}{4}\right) = 0.811$$

- S_{mild} has 4 'Yes' and 2 'No'

$$\mathrm{H}(S_{mild}) = -\left(\frac{4}{6}\log_2\frac{4}{6}\right) - \left(\frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$$

- S_{hot} has 2 'Yes' and 2 'No'

$$\mathrm{H}(S_{rain}) = -\left(\frac{2}{4}\log_2\frac{2}{4}\right) - \left(\frac{2}{4}\log_2\frac{2}{4}\right) = 1$$

$$\mathrm{H}(S_{\text{cold}}) = -\left(\frac{3}{4}\log_2\frac{3}{4}\right) - \left(\frac{1}{4}\log_2\frac{1}{4}\right) = 0.811$$

$$\mathrm{H}(S_{mild}) = -\left(\frac{4}{6}\log_2\frac{4}{6}\right) - \left(\frac{2}{6}\log_2\frac{2}{6}\right) = 0.918$$

$$\mathrm{H}(S_{rain}) = -\left(\frac{2}{4}\log_2\frac{2}{4}\right) - \left(\frac{2}{4}\log_2\frac{2}{4}\right) = 1$$

$$IG(S, Temperature) = \mathrm{H}(S) - \sum_{v \in \{Cold, Mild, Hot\}} \frac{|S_v|}{|S|}\mathrm{H}(S_v)$$

$$IG(S, Temperature) = \mathrm{H}(S) - \frac{4}{14}\mathrm{H}(S_{cold}) - \frac{6}{14}\mathrm{H}(S_{mild}) - \frac{4}{14}\mathrm{H}(S_{hot})$$

$$IG(S, Temperature) = 0.940 - \frac{4}{14}(0.811) - \frac{6}{14}(0.918) - \frac{4}{14}(1) = 0.029$$

This means that if we divide the dataset by keeping 'Temperature' attribute as the root, we get a reduction of 0.029 in the entropy value.

Humidity	Play
High	No
High	No
High	Yes
High	Yes
Normal	Yes
Normal	No
Normal	Yes
High	No
Normal	Yes
Normal	Yes
Normal	Yes
High	Yes
Normal	Yes
High	No

Dividing samples based on Humidity which is

- High
- Normal

S_{high} has 3 'Yes' and 4 'No'

$$\mathrm{H}(S_{\mathrm{high}}) = -\left(\frac{3}{7}\log_2\frac{3}{7}\right) - \left(\frac{4}{7}\log_2\frac{4}{7}\right) = 0.985$$

S_{normal} has 6 'Yes' and 1 'No'

$$\mathrm{H}(S_{normal}) = -\left(\frac{6}{7}\log_2\frac{6}{7}\right) - \left(\frac{1}{7}\log_2\frac{1}{7}\right) = 0.593$$

$$\mathrm{H}(S_{\text{high}}) = -\left(\frac{3}{7}\log_2\frac{3}{7}\right) - \left(\frac{4}{7}\log_2\frac{4}{7}\right) = 0.985$$

$$\mathrm{H}(S_{normal}) = -\left(\frac{6}{7}\log_2\frac{6}{7}\right) - \left(\frac{1}{7}\log_2\frac{1}{7}\right) = 0.593$$

$$IG(S, Humidity) = \mathrm{H}(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|}\mathrm{H}(S_v)$$

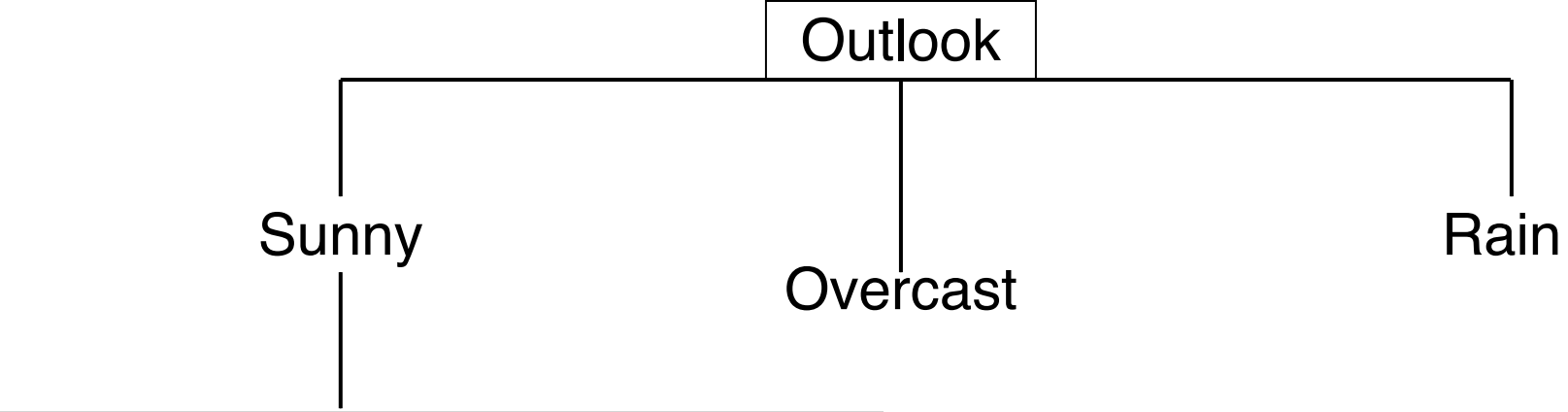
$$IG(S, Humidity) = \mathrm{H}(S) - \frac{7}{14}\mathrm{H}(S_{high}) - \frac{7}{14}\mathrm{H}(S_{normal})$$

$$IG(S, Humidity) = 0.940 - \frac{7}{14}(0.985) - \frac{7}{14}(0.593) = 0.151$$

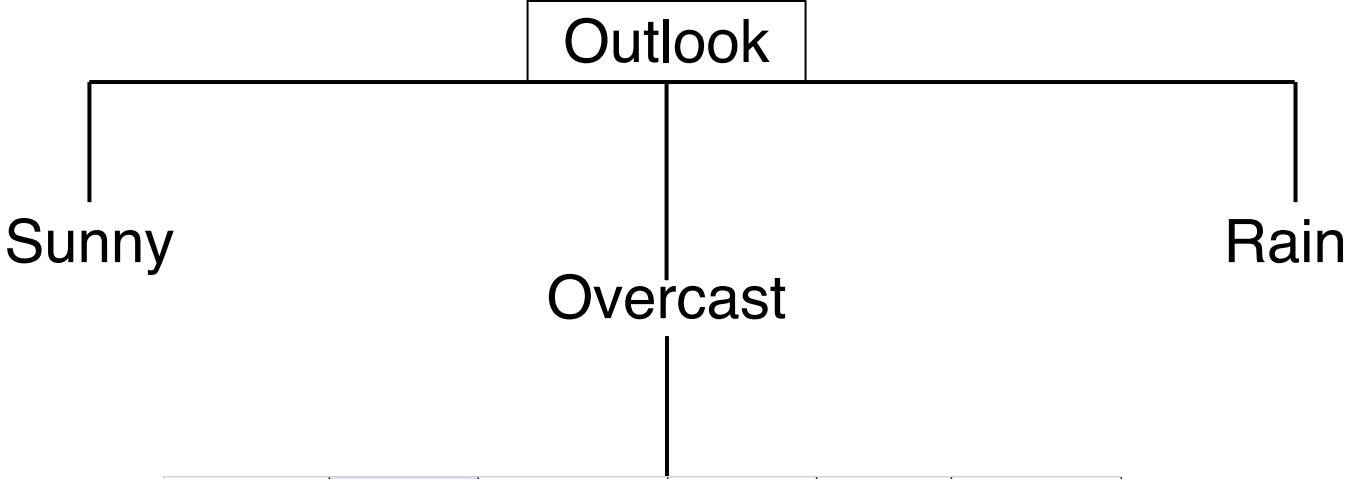
This means that if we divide the dataset by keeping 'Humidity' attribute as the root, we get a reduction of 0.151 in the entropy value.

Feature	**IG**
Wind	0.048
Outlook	0.246
Temperature	0.029
Humidity	0.151

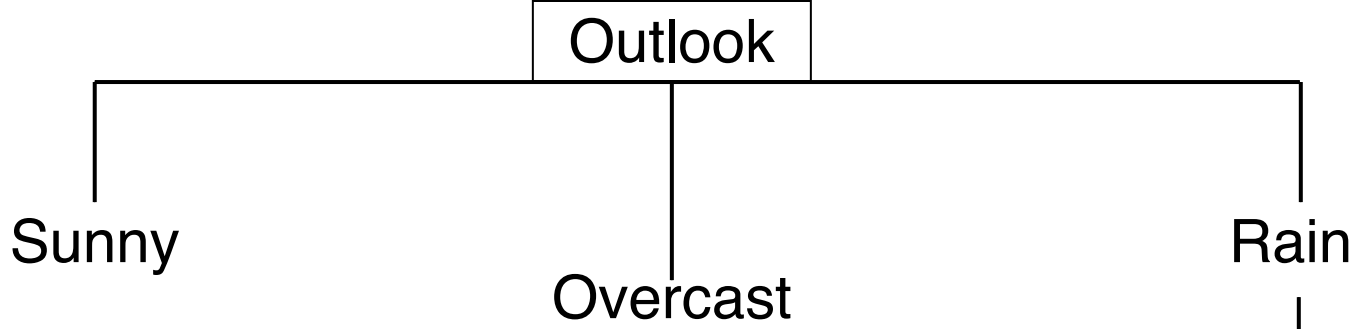
Since the attribute 'Outlook' provides the maximum reduction in entropy, we choose it as the root node.



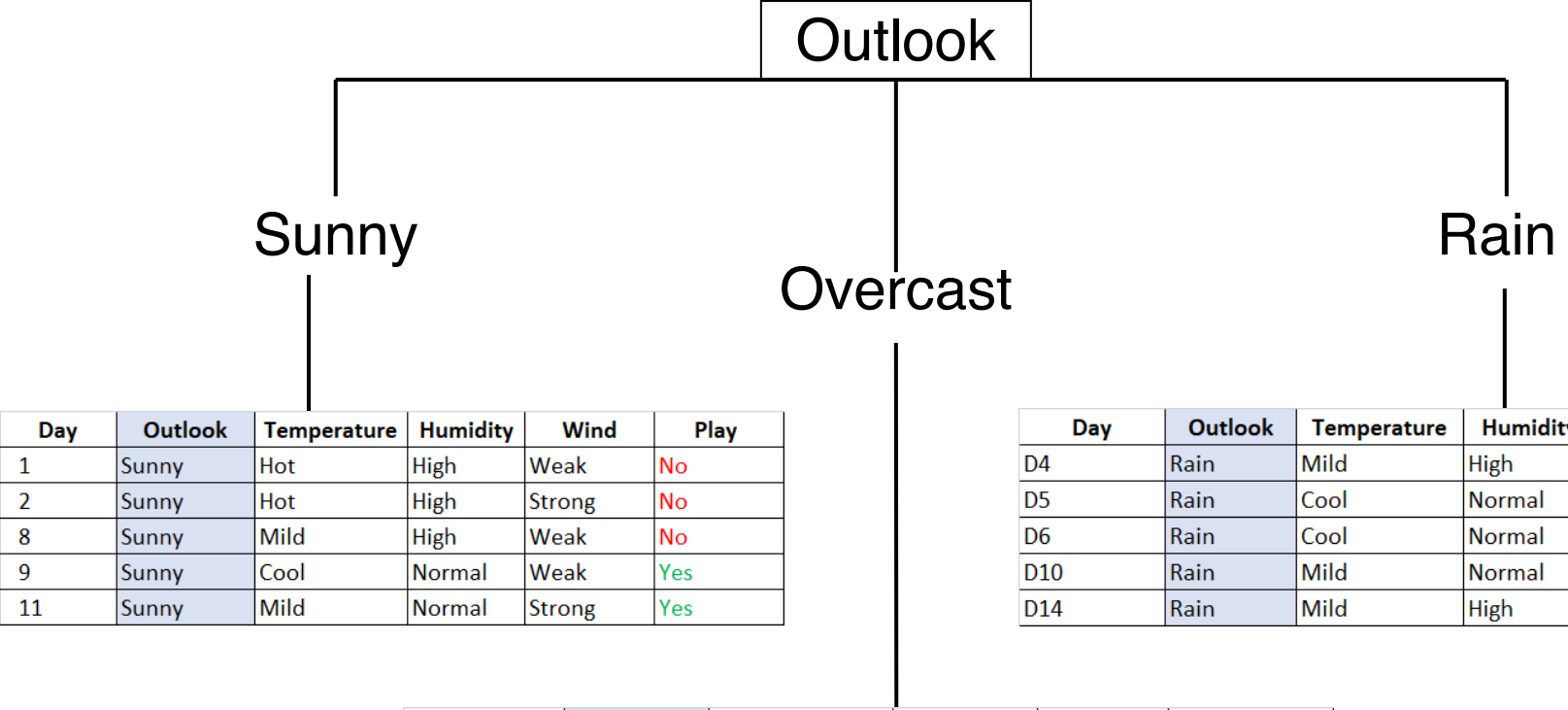
Day	Outlook	Temperature	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes



Day	Outlook	Temperature	Humidity	Wind	Play
D3	Overcast	Hot	High	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes



Day	Outlook	Temperature	Humidity	Wind	Play
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

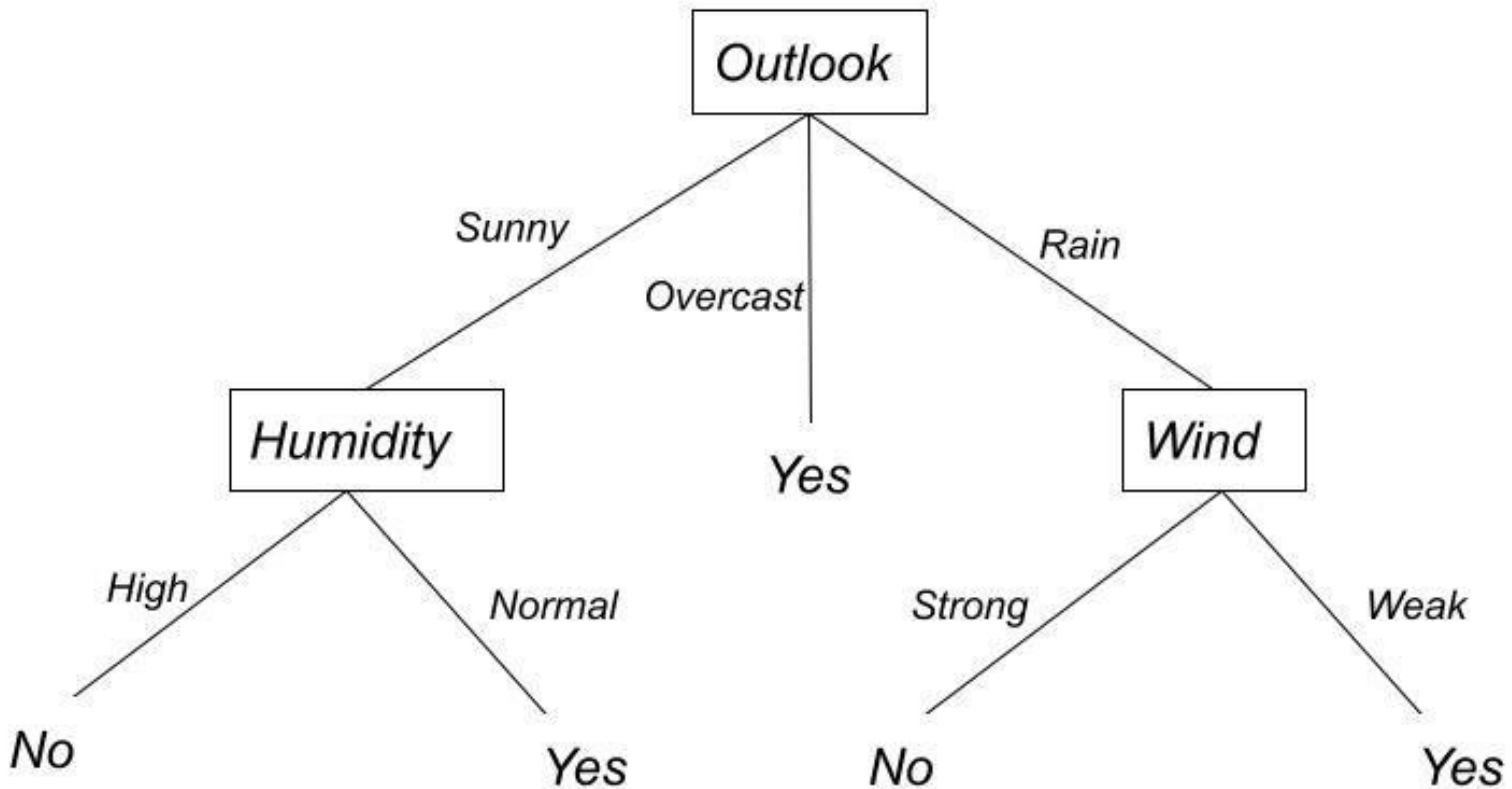


Day	Outlook	Temperature	Humidity	Wind	Play
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Day	Outlook	Temperature	Humidity	Wind	Play
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D10	Rain	Mild	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Day	Outlook	Temperature	Humidity	Wind	Play
D3	Overcast	Hot	High	Weak	Yes
D7	Overcast	Cool	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes

We can split the data in a similar manner. The final tree will look like



Classification And Regression Trees (CART)

CART can model both regression and classification problems.

CART learns decision trees for classification very similar to ID3.

Model

Suppose first that we have a partition into M regions $R_1, R_2, ..., R_M$ and we model the response as a constant c_i in each region:

$$\hat{y} = \sum_{i=1}^{M} c_i 1(x \in R_i)$$

where 1 is an indicator function.

In order to learn such a model from a training set, we have to determine the structure of the tree.

- Which input variable is chosen at each node to form the split criterion?
- What will be the value of the threshold parameter θ_i for the split?
- What will be the values of the predictive variables c_i within each region?

Loss Function and optimization

If the partitioning of the input space is given, and we minimize the sum-of-squares error function.

$$J = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2$$

Now finding the best binary partition in terms of minimum sum of squares is generally computationally infeasible.

Hence, we proceed with a greedy algorithm.

We start with all of the data, consider a splitting variable j and split point s, and define the pair of half-planes.

$$R_1 = \{x^{(i)} | x_j^{(i)} \leq s\} \text{ and } R_2 = \{x^{(i)} | x_j^{(i)} > s\}$$

We try to find the splitting variable j and split point s that solves

$$\min_{j,s}\left[\min_{c_1}\sum_{i;\mathbf{x}^{(i)}\in R_1}(\mathbf{y}^{(i)}-c_1)^2+\min_{c_2}\sum_{i;\mathbf{x}^{(i)}\in R_2}(\mathbf{y}^{(i)}-c_2)^2\right]$$

Notice that for different choices of j and s, different samples may belongs to child nodes.

For any choice of j and s, inner minimization is solved as

$$\begin{aligned}
\hat{c}_1 &= \min_{c_1} \sum_{i; x^{(i)} \in R_1} (y^{(i)} - \hat{y}^{(i)})^2 \\
&= \min_{c_1} \sum_{i; x^{(i)} \in R_1} (y^{(i)} - c_1)^2 \\
&= \frac{1}{N_1} \sum_{i; x^{(i)} \in R_1} y^{(i)}
\end{aligned}$$

where N_1 is the number of data points in region R_1

$$\hat{c}_2 = \min_{c_2} \sum_{i; x^{(i)} \in R_2} (y^{(i)} - \hat{y}^{(i)})^2$$

$$\hat{c}_2 = \min_{c_2} \sum_{i; x^{(i)} \in R_2} (y^{(i)} - c_2)^2$$

$$= \frac{1}{N_2} \sum_{i; x^{(i)} \in R_2} y^{(i)}$$

where N_2 is the number of data points in region R_2

For each splitting variable, determination of the best pair (j, s) is feasible.

The determination of the split point s can be done by scanning through all of the inputs,

Having found the best split, we partition the data into the two resulting regions

Repeat the splitting process on each of the two regions.

This process is repeated on all of the resulting regions.

Note: Variance reduction can also be use in place of sum of squared error in regression trees.

Classification tree

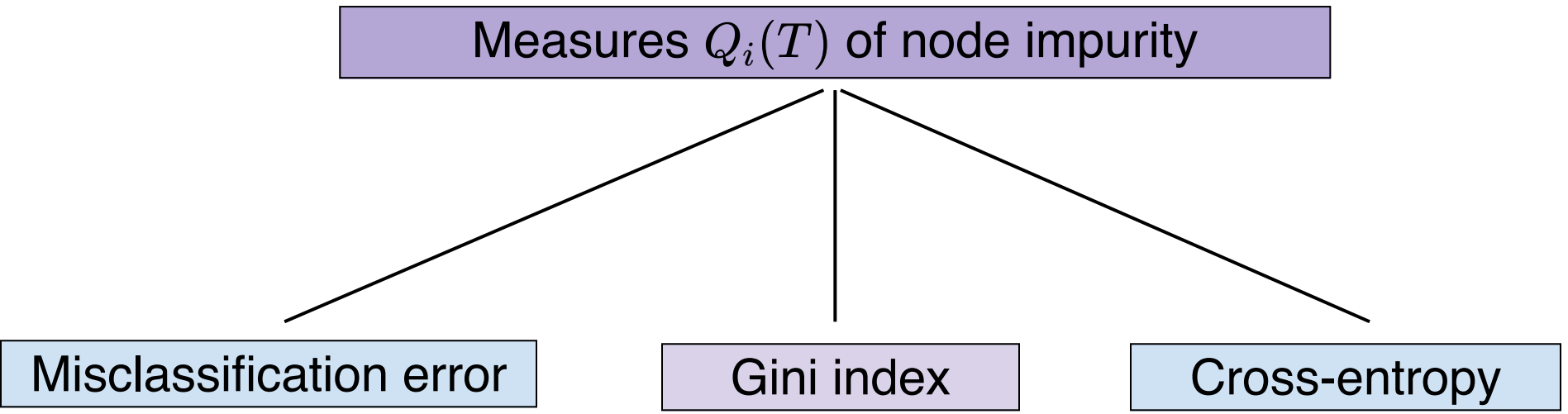
For classification problems, the process of growing and pruning the tree is similar, except that the sum-of-squares error is replaced by a more appropriate measure of performance.

If we define $p_{i,k}$ to be the proportion of data points in region R_i assigned to class k, where $k = 1, ..., K$ then

$$p_{i,k} = \frac{1}{N_i} \sum_{x^{(i)} \in R_i} 1(y^{(i)} = k)$$

We classify the observations in node i to class k if

$$p_{i,k} > p_{i,j} \quad \forall j = 1, 2, ...k-1, k+1, ...K$$



- Cross-entropy and the Gini index are differentiable, and hence more amenable to numerical optimization.
- The cross-entropy and the Gini index are better measures than the misclassification rate for growing the tree because they are more sensitive to the node probabilities.
- For subsequent pruning of the tree, the misclassification rate is generally used.

Size of the tree

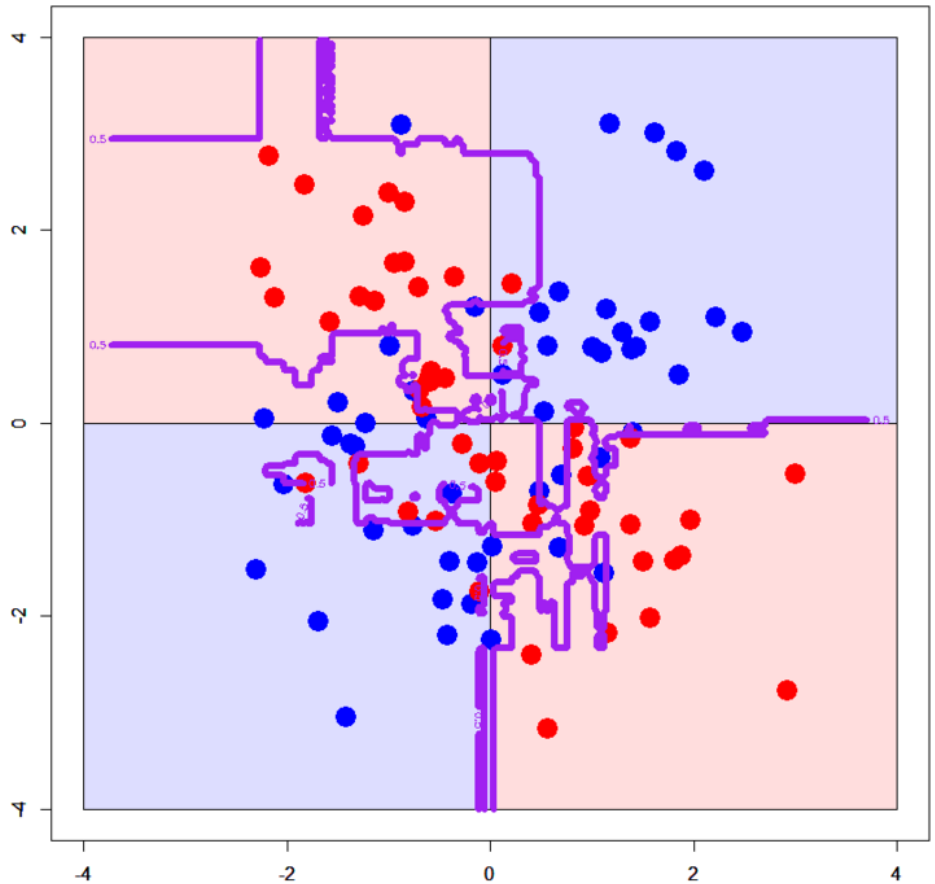
The size of the tree may result in the overfitting or underfitting of the model.

Length of the longest path from the tree root to a leaf is known as the maximum depth of the tree.

- The root node is considered to have a depth of 0.

Overfitting

- In decision trees, over-fitting occurs when the tree is designed so as to perfectly fit all samples in the training dataset.
- Thus it ends up with branches with strict rules and sparse data.
- This affects the accuracy when predicting on test dataset.
- Clearly, a very large tree might overfit the data.
- A small tree might not capture the important structure.
- Therefore, tree size is a tuning parameter.



If we do not control the depth, decision tree will also fit the noise, as shown in above image.

image source: https://tjmachinelearning.com/lectures/1819/rf/

How to avoid overfitting?

Pre-puning

One approach to avoid overfitting is to stop the algorithm before it becomes a fully-grown tree.

Stopping criterion:

- Stop if the number of samples is less than some user specified threshold.
- Stop if expanding the current node does not improve impurity.

This strategy is too short-sighted, however, since a seemingly worthless split might lead to a very good split below it.

The preferred strategy is to grow a large tree T_0, stopping the splitting process only when some minimum node size is reached. Then this large tree is pruned using cost-complexity pruning.

Post-pruning

Grow a large tree (call it T_0), using a stopping criterion based on the number of data points associated with the leaf nodes.

Then prune back the resulting tree.

The pruning is based on a criterion that balances residual error against a measure of model complexity.

We define $T \subset T_0$ to be a subtree of T_0 if it can be obtained by pruning nodes from T_0.

The prediction for region R_i is then given by

$$c_i = \frac{1}{N_i} \sum_{x^{(i)} \in R_m} y^{(i)}$$

where N_i is the number of data points in region R_i.

And the corresponding contribution to the residual sum-of-squares is given by

$$Q_i(T) = \sum_{x^{(i)} \in R_i} (y^{(i)} - c_i)^2$$

The pruning criterion is then given by

$$C(T) = \sum_{i=1}^{|T|} Q_i(T) + \alpha|T|$$

The regularization parameter α determines the trade-off between the overall residual sum-of-squares error and the complexity of the model as measured by the number $|T|$ of leaf nodes, and its value is chosen by cross-validation.

Large values of α result in smaller trees T, and conversely for smaller values of α.

Advantages and disadvantages of decision trees

Advantages of decision trees

- Versatile, can perform classification, regression and multi-output tasks.
- Simple to understand and interpret. Trees can be visualized.
- Needs little data preprocessing.
- Cost of using tree is logarithmic in the number of training data samples.

Disdvantages of decision trees

- Can create overly complex trees that do not generalize to unseen data. Pruning is needed to address this issue.
- Decision trees suffer from the problem of high variance, which can be mitigated by ensembles.
- Trees learn piecewise linear approximation and hence are not good at extrapolation.