

Machine Learning

Decision Trees

Random Forest

Francesco Pugliese, PhD
neural1977@gmail.com

We will discuss about...

- ✓ Decision Trees
- ✓ Random Forest

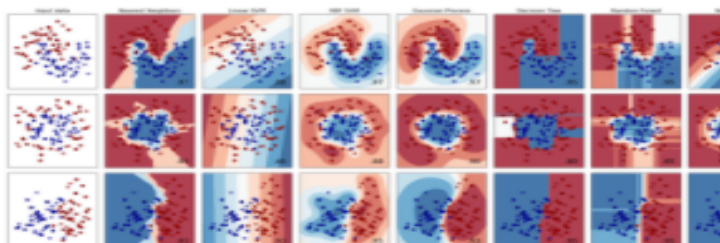
Machine Learning and Scikit-Learn Universe

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: SVM, nearest neighbors, random forest, and more...

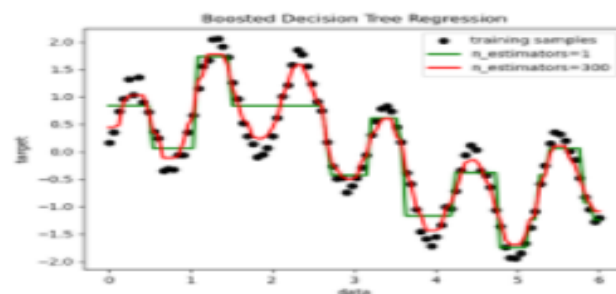


Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, nearest neighbors, random forest, and more...



Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift, and more...

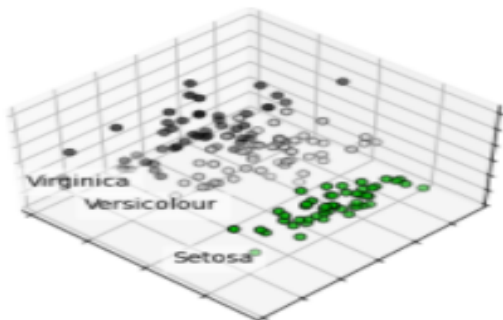


Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: k-Means, feature selection, non-negative matrix factorization, and more...

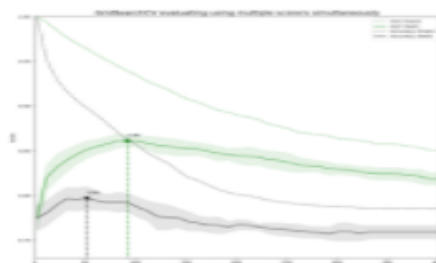


Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter tuning

Algorithms: grid search, cross validation, metrics, and more...

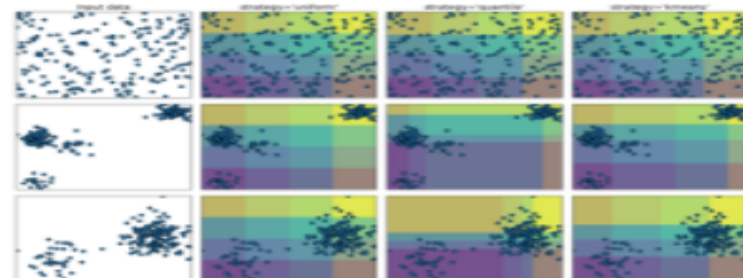


Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.

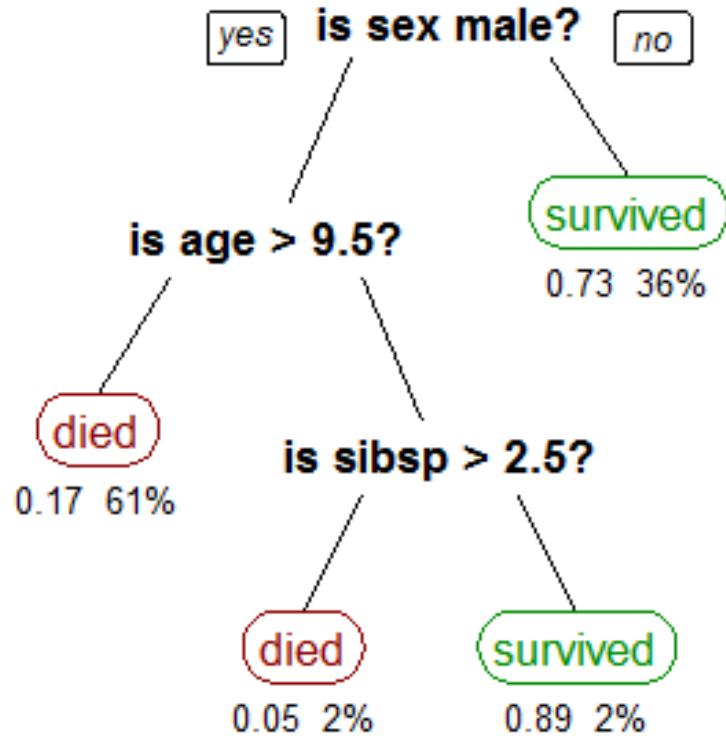
Algorithms: preprocessing, feature extraction, and more...



Decision trees

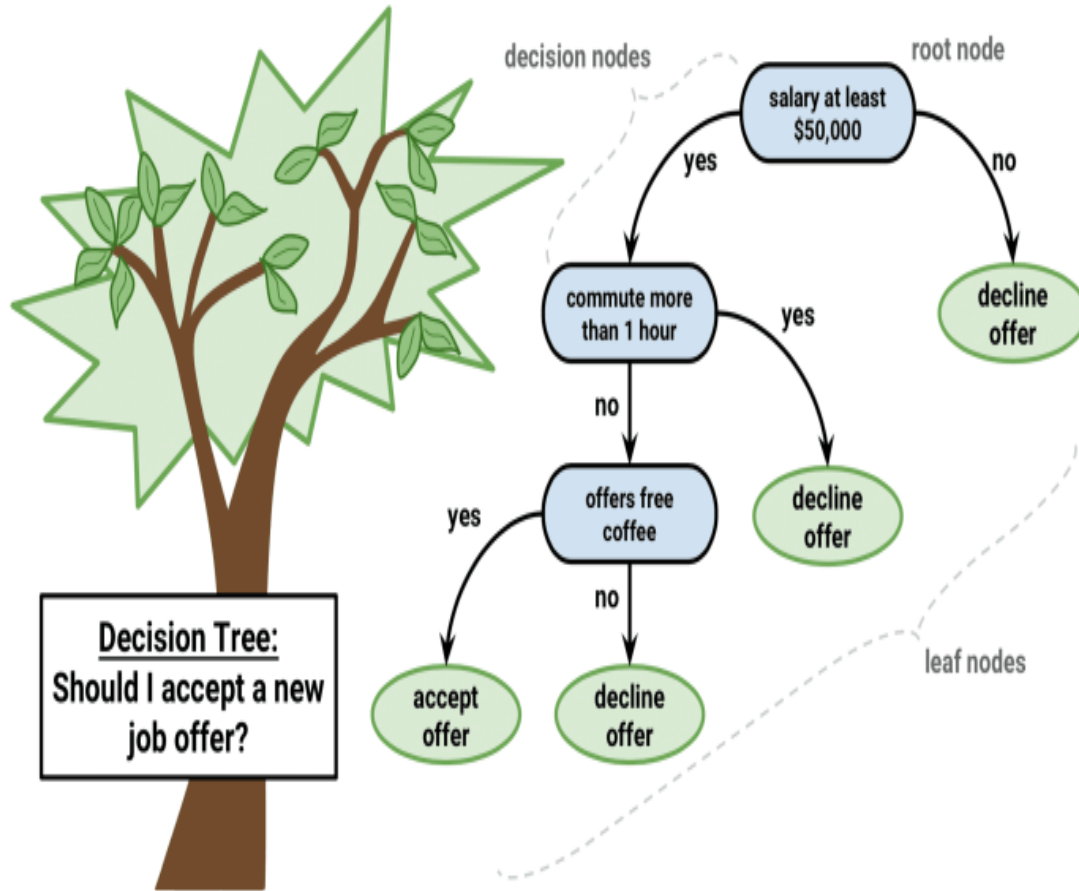
- ✓ A decision tree is a **decision support tool** that uses a **tree-like** model of decision-making process and the possible **consequences**.
- ✓ It covers event outcomes, resource costs, and utility of decisions.
- ✓ Resemble an algorithm or a **flowchart** that contains only conditional control statements.
- ✓ Each decision tree has 3 key parts: a **root** node, **leaf** nodes, **branches**.
- ✓ In a decision tree, each internal node represents a test or an event. Say, a heads or a tail in a coin flip.
- ✓ Each **branch** represents the **outcome** of the test and each **leaf** node represents a **class** label — a **decision taken after computing all attributes**.
- ✓ The **paths** from root to leaf nodes represent the **classification rules**.

Decision trees



- In **Decision Trees**, for predicting a class label for a record we start from the **root** of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.
- **Decision Trees** as the name suggests works on a set of decisions derived from the data and its behavior.
- It does not use a linear classifier or regressor, so its performance is independent of the linear nature of the data

Some Concepts related to Decision trees



1.Root Node: It represents the entire population or sample and this further gets divided into two or more homogeneous sets.

2.Splitting: It is a process of dividing a node into two or more sub-nodes.

3.Decision Node: When a sub-node splits into further sub-nodes, then it is called the decision node.

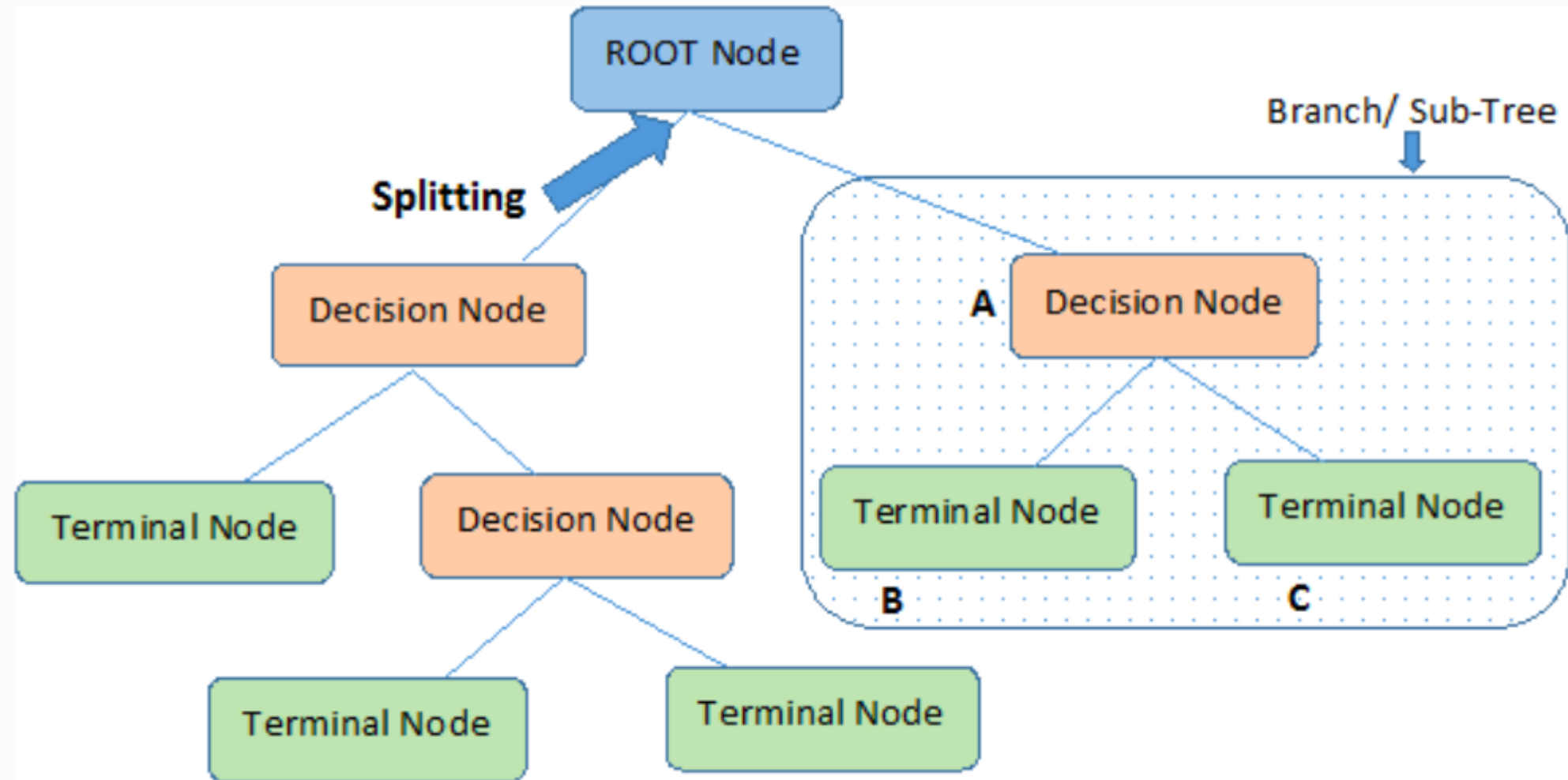
4.Leaf / Terminal Node: Nodes do not split is called Leaf or Terminal node.

5.Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.

6.Branch / Sub-Tree: A subsection of the entire tree is called branch or sub-tree.

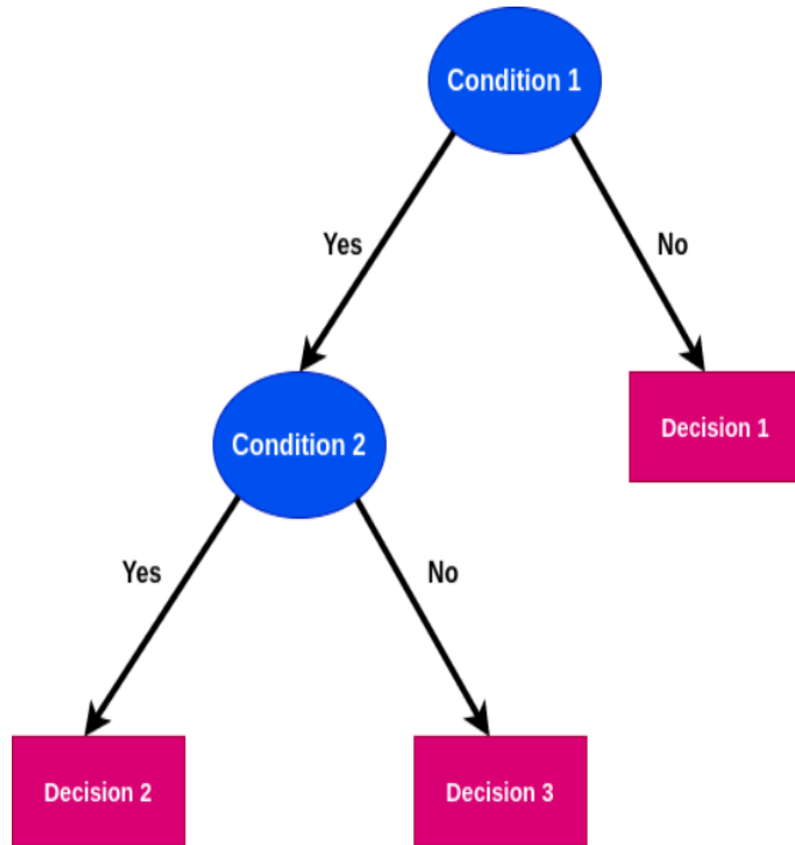
7.Parent and Child Node: A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.

Some Concepts related to Decision trees



Note:- A is parent node of B and C.

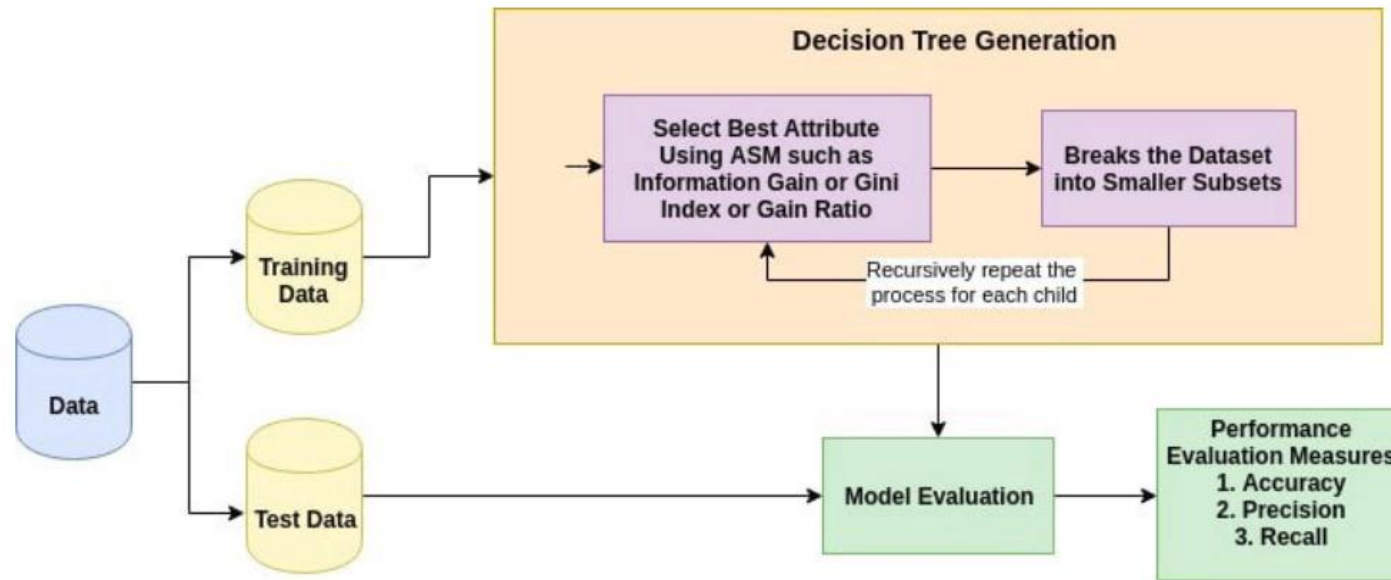
Decision trees Training



- The **decision trees** use the **CART** algorithm (Classification and Regression Trees). In both cases, decisions are based on conditions on any of the features.
- The internal nodes represent the conditions and the leaf nodes represent the decision based on the conditions.
- A **decision tree** is a graphical representation of all possible solutions to a decision based on certain conditions.
- On each step or node of a **decision tree**, used for classification, we try to form a condition on the features to separate all the labels or classes contained in the dataset to the fullest **purity**.

Decision trees Training

- The creation of sub-nodes increases the **homogeneity** of resultant sub-nodes. In other words, we can say that the **purity** of the node increases with respect to the target variable.
- The decision tree splits the nodes on all available variables and then selects the split which results in most **homogeneous** sub-nodes.



Decision trees Training



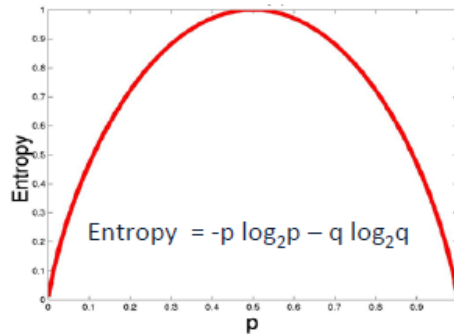
The algorithm selection is based on the type of target variables.

Some algorithms used in Decision Trees:

- **ID3** → (extension of D3)
- **C4.5** → (successor of ID3)
- **CART** → (Classification And Regression Tree)
- **CHAID** → (Chi-square automatic interaction detection Performs multi-level splits when computing classification trees)
- **MARS** → (multivariate adaptive regression splines)

Decision trees Training

Entropy



a) Entropy of the target (Entropy using the frequency table of one attribute)

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

b) Entropy of each attribute (Entropy using the frequency table of two attributes)

$$E(T, X) = \sum_{c \in X} P(c) E(c)$$

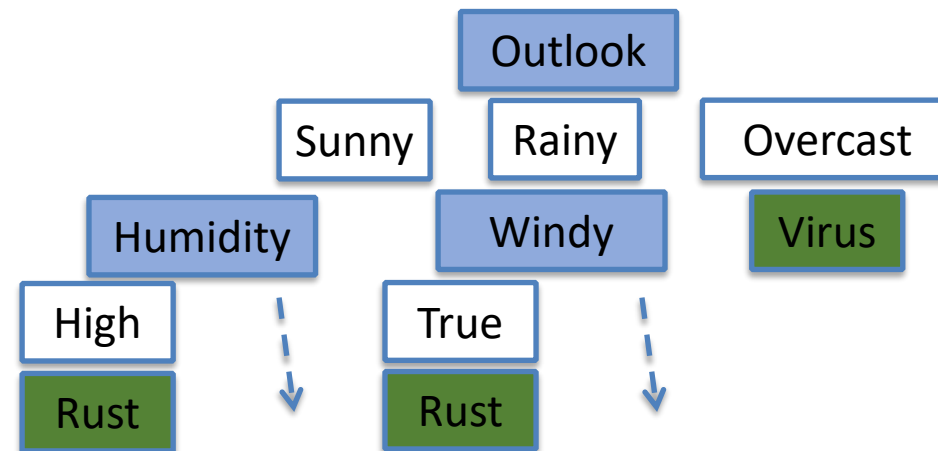
Information Gain

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

ID3 Learning Algorithm (Quinlan, 1986)

Steps:

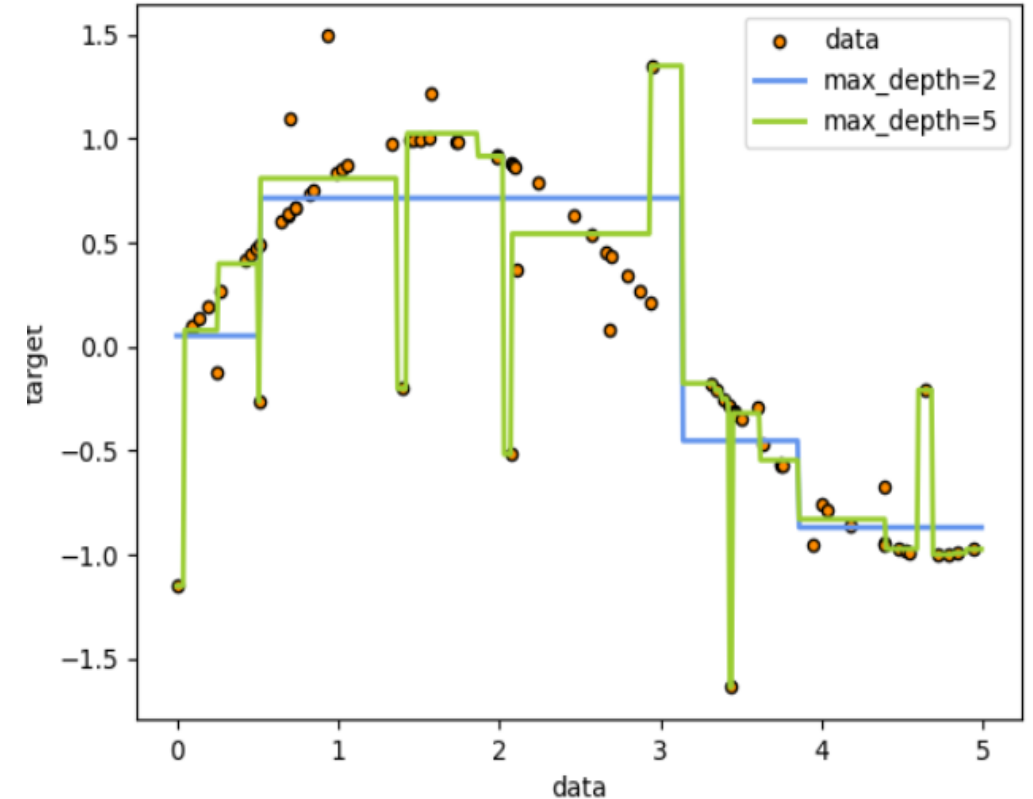
1. Calculate entropy of the target.
2. Calculate the Information Gain of each attribute.
3. Choose the attribute with the largest Information Gain as Node of the tree.
4. Recursively run the algorithm on the non-leaf branches.



Decision trees Training ID3

Steps in ID3 (Iterative Dichotomiser 3) algorithm used to generate a decision tree from a dataset. ID3 is the precursor to the C4.5 algorithm:

1. It begins with the original set S as the root node.
2. On each iteration of the algorithm, it iterates through the very unused attribute of the set S and calculates **Entropy(H)** and **Information gain(IG)** of this attribute.
3. It then selects the attribute which has the smallest Entropy or Largest Information gain.
4. The set S is then split by the selected attribute to produce a subset of the data.
5. The algorithm continues to recur on each subset, considering only attributes never selected before.



Decision trees Training ID3

Steps in ID3 (Iterative Dichotomiser 3) algorithm used to generate a decision tree from a dataset. ID3 is the precursor to the C4.5 algorithm:

Namely:

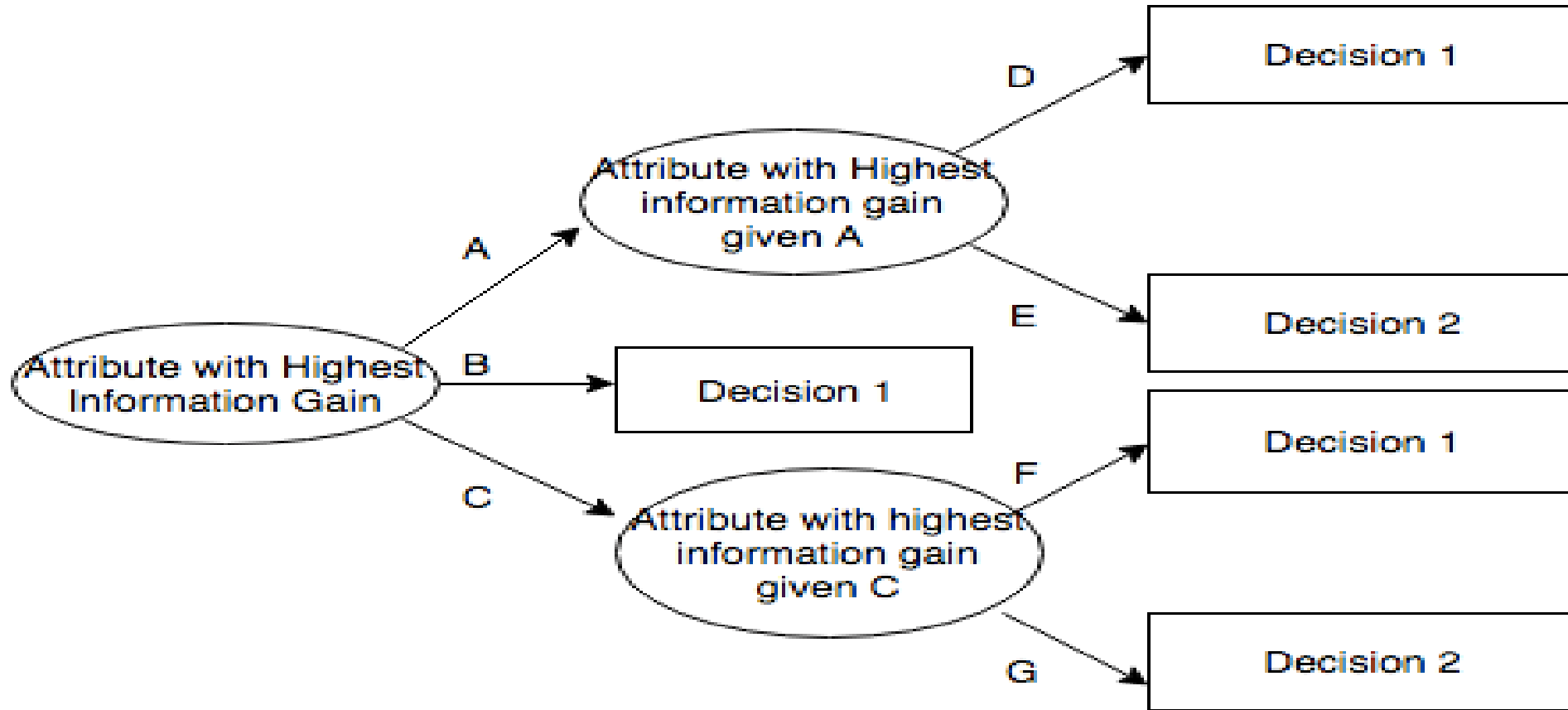
The ID3 algorithm begins with the original set S as the root node. On each iteration of the algorithm, it iterates through every unused attribute of the set S and calculates the entropy $H(S)$ or the information gain $IG(S)$ of that attribute. It then selects the attribute which has the smallest entropy (or largest information gain) value. The set S is then split or partitioned by the selected attribute to produce subsets of the data.

For example, a node can be split into child nodes based upon the subsets of the population whose ages are less than 50, between 50 and 100, and greater than 100. The algorithm continues to recurse on each subset, considering only attributes never selected before.

Recursion on a subset stops when:

- 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 class 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

Decision trees Training ID3

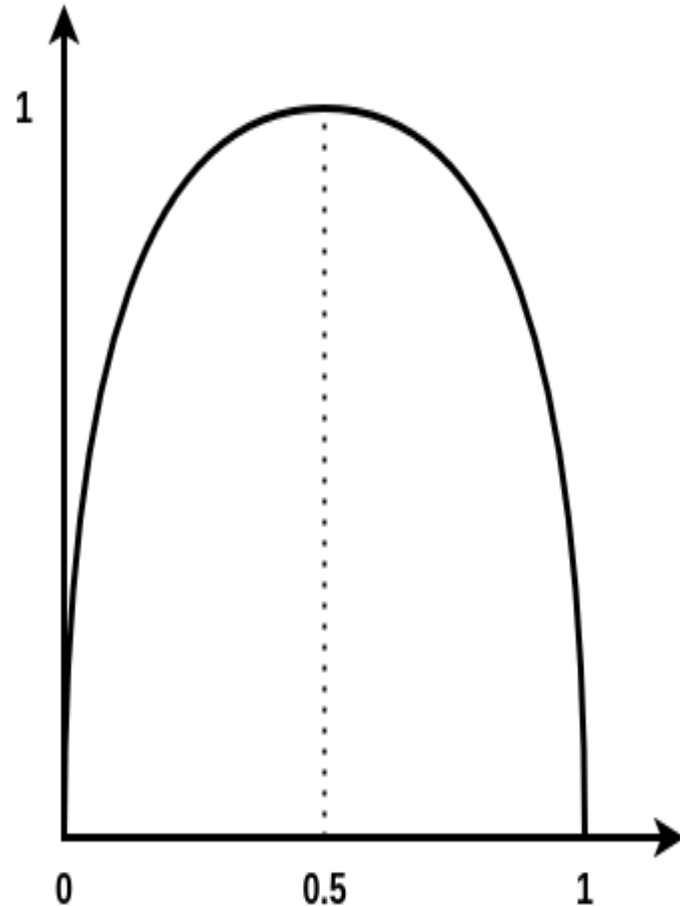


Decision trees Training: Entropy

- **Entropy:** It gives the measure of impurity or randomness in the data. It is given by:

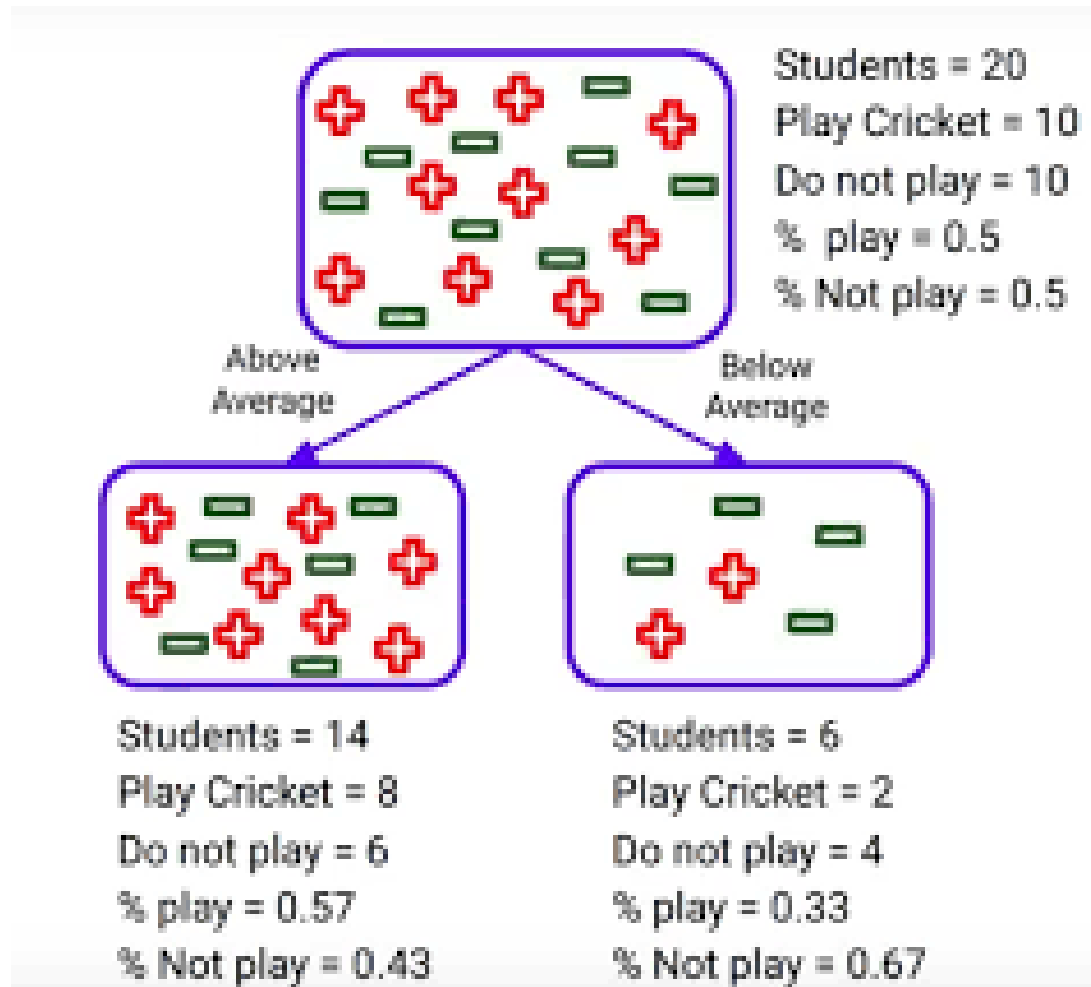
$$\text{Entropy} = -P(\text{class 1}) \times \log(P(\text{class 1})) - P(\text{class 2}) \times \log(P(\text{class 2}))$$

where P denotes the probability.



- If there are two classes, class 1 and class 2, of equal numbers, i.e, the number of entries of class 1 is equal to the number of entries of class 2, and we randomly select an entry, it will belong to any class 1 or 2 with a 50% probability each. In such cases, the entropy will be high.
- If a certain dataset has all data belonging to either class 1 or class 2, the entropy obtained is 0, as in that case $P(\text{class1})$ or $P(\text{class2})$ will be equal to 1. If $P(\text{class1})=0$ then $P(\text{class2})$ should be equal to 1. So, it is evident that the entropy will be high if we have impure or mixed class labels in a dataset.
- The beside diagram shows the variation of the probability of labels with entropy. We can see if the probability of a label is 0.5, the entropy is the maximum.

Decision trees Training: Information Gain




In other words **Information gain** is used to decide which feature to split on at each step in building the tree. Simplicity is best, so we want to keep our tree small. To do so, at each step we should choose the split that results in the purest daughter nodes. A commonly used measure of purity is called information. For each node of the tree, the information value measures how much information a feature gives us about the class. The split with the highest information gain will be taken as the first split and the process will continue until all children nodes are pure, or until the information gain is 0.


Decision trees Training: Information Gain

Feature 1	Feature 2	Label
1	1	lab_1
1	1	lab_1
2	1	lab_2
3	2	lab_2
3	3	lab_2
3	3	lab_1
2	3	lab_2
1	2	lab_1
1	3	lab_2
2	2	lab_2

- **Information Gain:** The information gain is the decrease in the entropy after the dataset is split on the basis of an attribute. Constructing a decision tree depends on finding the attribute that returns the highest information gain. It helps in choosing which feature or attribute will be used to create the deciding internal node at a particular point.
- Information gain = Entropy(s) — [(Weighted average) x (Entropy of each feature)]



Feature 1==1		Feature 1==2		Feature 1==3
labels		labels		labels
lab_1		lab_2		lab_2
lab_1		lab_2		lab_2
lab_1		lab_2		lab_1
lab_2				



Feature 2==1		Feature 2==2		Feature 2==3
labels		labels		labels
lab_1		lab_2		lab_2
lab_1		lab_1		lab_1
lab_2		lab_2		lab_2
				lab_2

Information Gain Ft1= 0.313

Information Gain Ft2 = 0.1

Decision trees Training: GINI Impurity

- **Pure**

Pure means, in a selected sample of dataset all data belongs to same class (PURE).

- **Impure**

Impure means, data is mixture of different classes.

- Definition of Gini Impurity

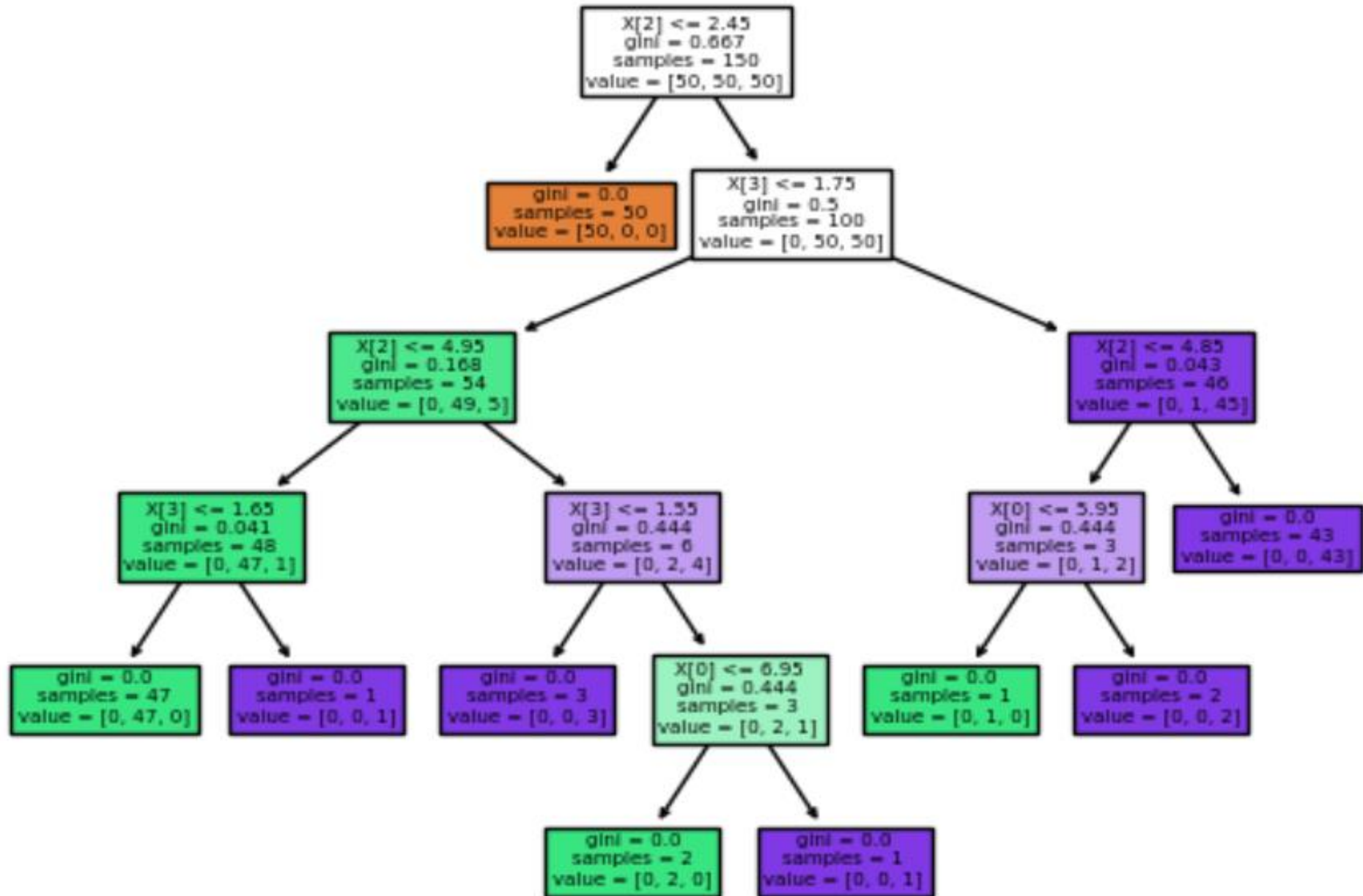
Gini Impurity is a measurement of the likelihood of an incorrect classification of a new instance of a random variable, if that new instance were randomly classified according to the distribution of class labels from the data set.

If our dataset is Pure then likelihood of incorrect classification is 0. If our sample is mixture of different classes then likelihood of incorrect classification will be high.

$$\text{Gini} = 1 - \sum_{i=1}^n (p_i)^2$$

When you use the Gini index as the criterion for the algorithm to select the feature for the root node.,The feature with the least Gini index is selected.

Decision trees Training



Decision trees

- ✓ Can be a powerful machine learning algorithm for **classification** and **regression**.
- ✓ **Classification** tree works on the target to classify if it was a **heads** or a **tail**.
- ✓ **Regression** trees are represented in a similar manner, but they predict **continuous** values like **house prices** in a neighbourhood.
- ✓ The best part about decision trees:
 - ✓ Handle both **numerical** and **categorical** data
 - ✓ Handle **multi-output** problems
 - ✓ Decision trees require relatively **less effort in data preparation**
 - ✓ **Nonlinear** relationships between parameters do not affect tree performance

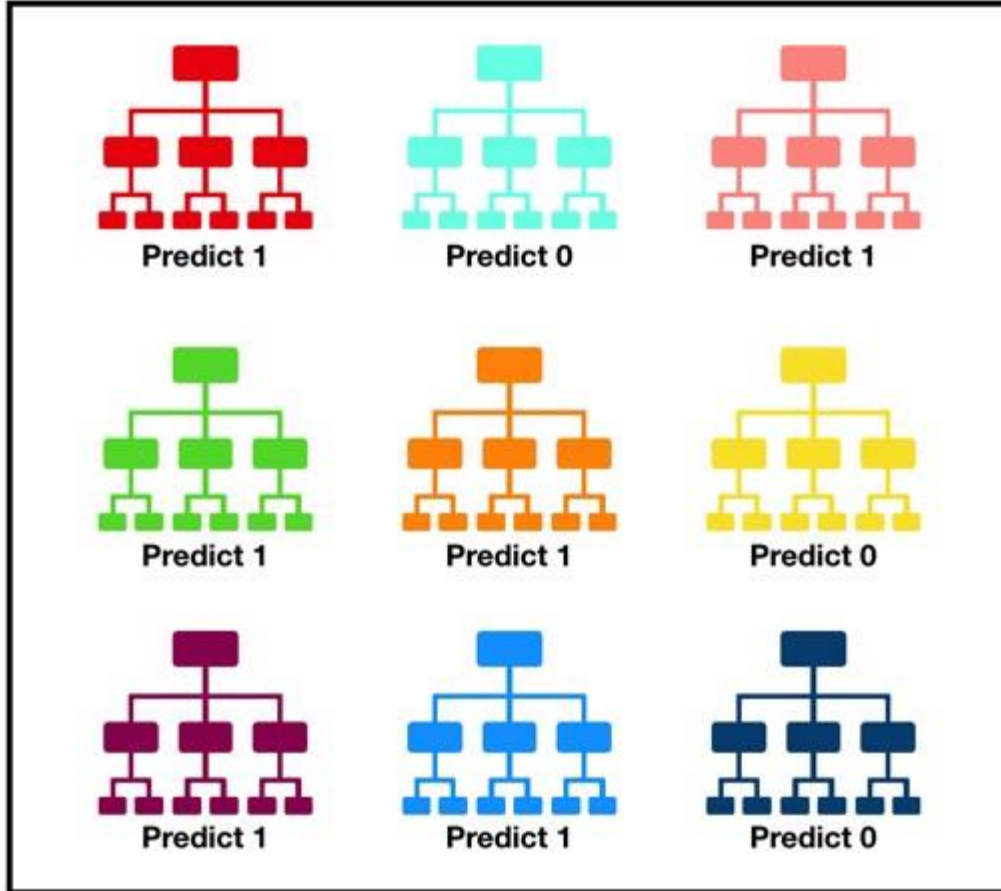
Applications of Decision trees

- ✓ Selecting a flight to travel
- ✓ Predicting high occupancy dates for hotels
- ✓ Number of drug stores nearby was particularly effective for a client X
- ✓ Cancer vs non-cancerous cell classification where cancerous cells are rare say 1%
- ✓ Suggest a customer what car to buy

Random Forest

- ✓ Is an ensemble learning technique about classification, regression and other operations that depend on a multitude of decision trees at the training time.
- ✓ They are fast, flexible, represent a robust approach to mining high-dimensional data and are an extension of classification and regression decision trees we talked about above.
- ✓ Ensemble learning, in general, can be defined as a model that makes predictions by combining individual models.
- ✓ The ensemble model tends to be more flexible with less bias and less variance.
- ✓ Ensemble Learning has two popular methods as:
 - ✓ **Bagging**: Each individual tree to randomly sample from the dataset and trained by s random subset of data, resulting in different trees
 - ✓ **Boosting**: Each individual tree /model learns from mistakes made by the previous model and improves

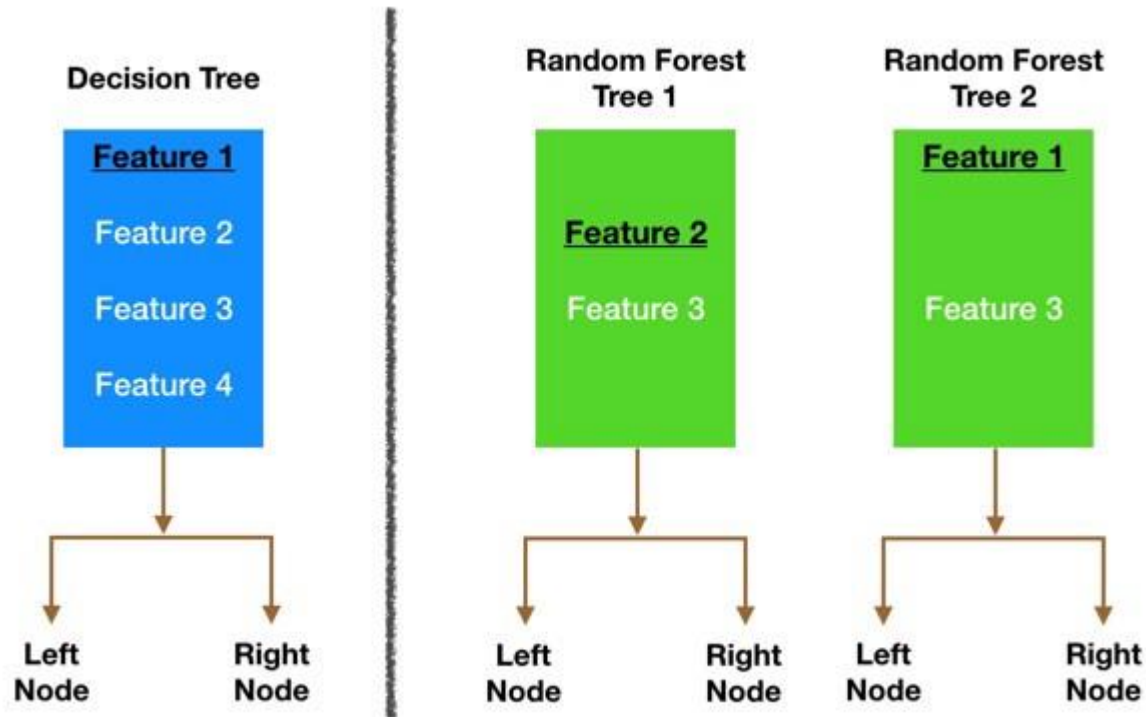
Random Forest



Tally: Six 1s and Three 0s
Prediction: 1

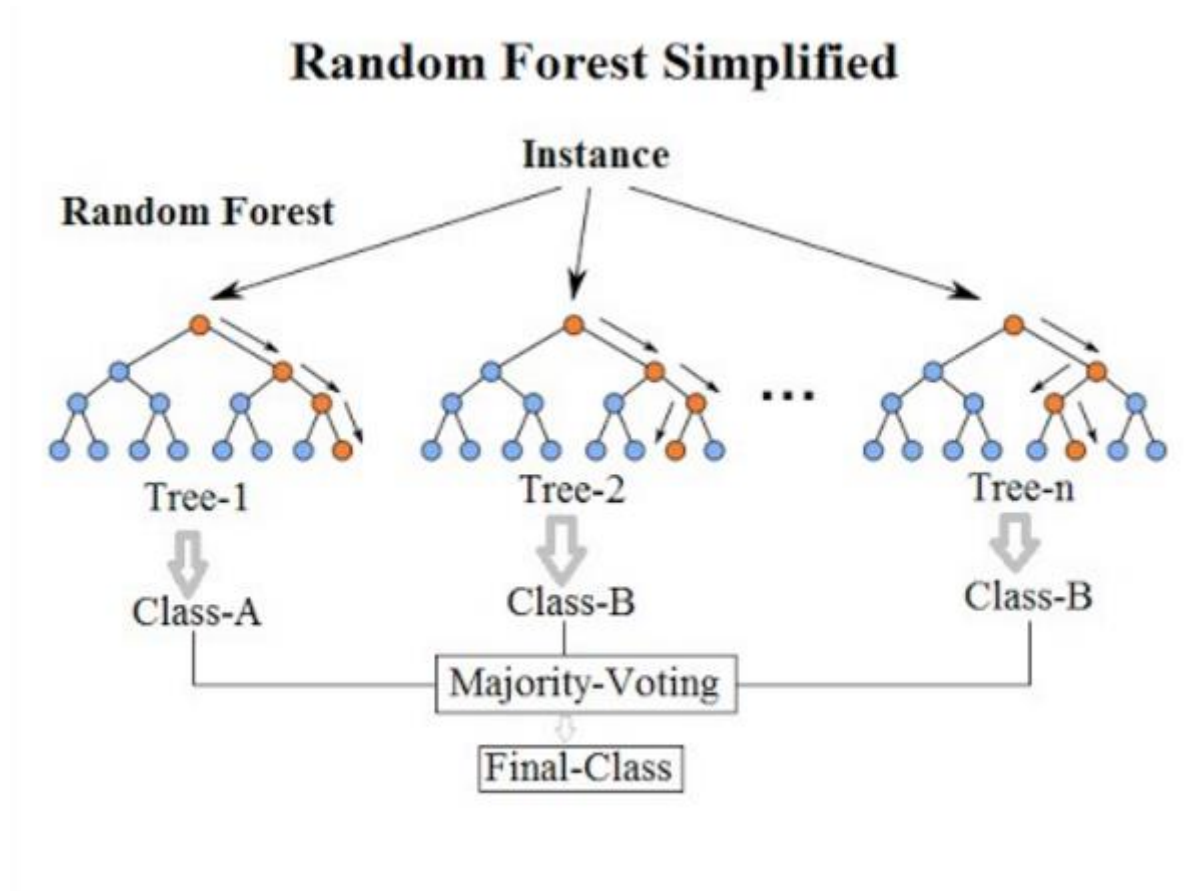
- **Random forest**, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction
- A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.
- **Bagging (Bootstrap Aggregation)** — Decisions trees are very sensitive to the data they are trained on — small changes to the training set can result in significantly different tree structures. Random forest takes advantage of this by allowing each individual tree to randomly sample from the dataset with replacement, resulting in different trees. This process is known as bagging.

Random Forest



- We are not subsetting the training data into smaller chunks and training each tree on a different chunk. Rather, if we have a sample of size N , we are still feeding each tree a training set of size N (unless specified otherwise). But instead of the original training data, we take a random sample of size N with replacement. For example, if our training data was $[1, 2, 3, 4, 5, 6]$ then we might give one of our trees the following list $[1, 2, 2, 3, 6, 6]$. Notice that both lists are of length six and that "2" and "6" are both repeated in the randomly selected training data we give to our tree (because we sample with replacement).

Random Forest



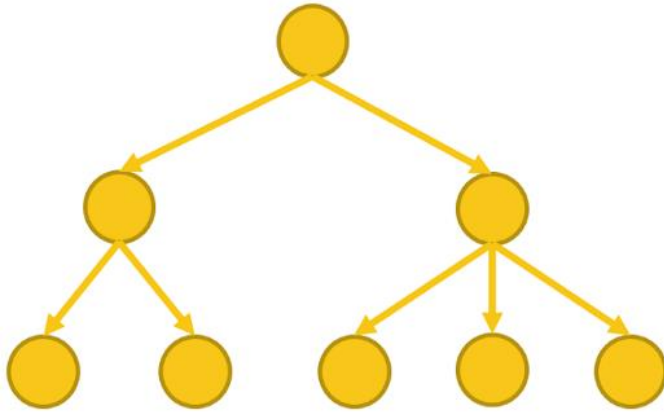
Feature Randomness — In a normal decision tree, when it is time to split a node, we consider every possible feature and pick the one that produces the most separation between the observations in the left node vs. those in the right node. In contrast, each tree in a random forest can pick only from a random subset of features. This forces even more variation amongst the trees in the model and ultimately results in lower correlation across trees and more diversification.

Random Forest

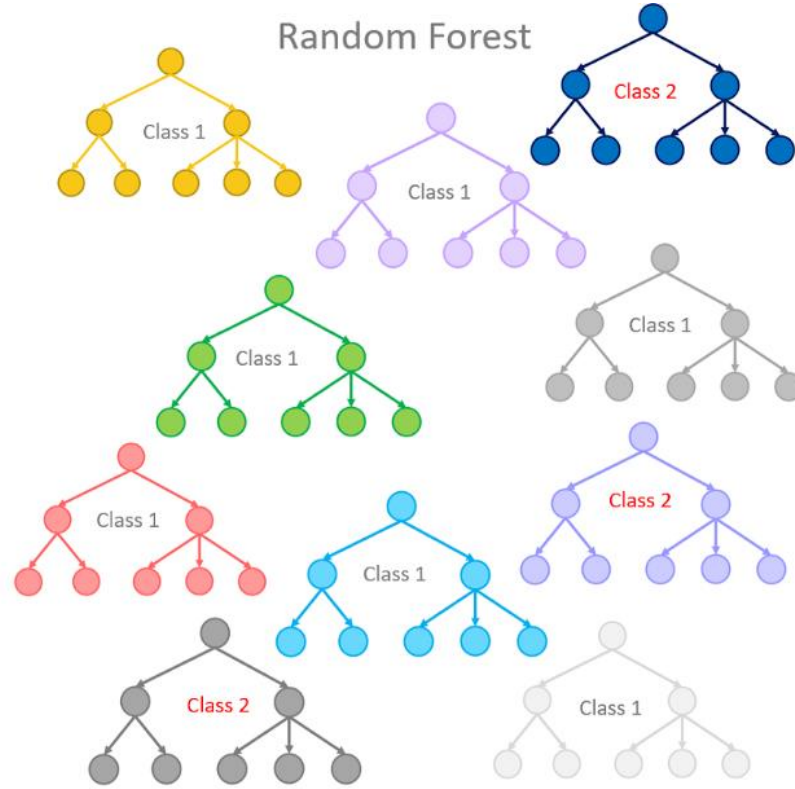
- ✓ Random forest run times are quite fast.
- ✓ They are pretty efficient in dealing with missing and incorrect data.
- ✓ On the negatives, they cannot predict **beyond** the defined range in the training data, and that they may **over-fit** data sets that are particularly noisy.
- ✓ A random forest should have a number of trees between **64–128 trees**.
- ✓ **Random Forest vs Decision Tree:**
 - ✓ Random Forest is essentially a collection of Decision Trees.
 - ✓ A decision tree is built on an entire dataset, using all the features/variables of interest, whereas a random forest **randomly** selects **observations/rows** and **specific features/variables** to build multiple decision trees from and then **averages** the results.

Random Forest

Single Decision Tree



Random Forest



Applications of Random Forest

- ✓ Fraud detection for bank accounts, credit card
- ✓ Detect and predict the drug sensitivity of a medicine
- ✓ Identify a patient's disease by analysing their medical records
- ✓ Predict estimated loss or profit while purchasing a particular stock

References

Kingsford, C., & Salzberg, S. L. (2008). What are decision trees?. *Nature biotechnology*, 26(9), 1011-1013.

Cheng, J., Fayyad, U. M., Irani, K. B., & Qian, Z. (1988). Improved decision trees: a generalized version of id3. In *Machine Learning Proceedings 1988* (pp. 100-106). Morgan Kaufmann.

Oshiro, T. M., Perez, P. S., & Baranauskas, J. A. (2012, July). How many trees in a random forest?. In *International workshop on machine learning and data mining in pattern recognition* (pp. 154-168). Springer, Berlin, Heidelberg.

Francesco Pugliese

