# Trees based methods

## Decision Trees, Random Forests and Gradient Boosting

Hicham Zmarrou, PhD

2018-11-16

# Aims of this lesson

- Understand what are decision trees (DT), random forests (RF) and gradient boosting (GB), how they works, and how to evaluate a DT a RF or a GB model.

# Aims of this lesson

- Understand what are decision trees (DT), random forests (RF) and gradient boosting (GB), how they works, and how to evaluate a DT a RF or a GB model.
- Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) mostly used in classification problems.

## Aims of this lesson

- Understand what are decision trees (DT), random forests (RF) and gradient boosting (GB), how they works, and how to evaluate a DT a RF or a GB model.
- Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) mostly used in classification problems.
- It works for both categorical and continuous input and output variables.

# Aims of this lesson

- Understand what are decision trees (DT), random forests (RF) and gradient boosting (GB), how they works, and how to evaluate a DT a RF or a GB model.
- Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) mostly used in classification problems.
- It works for both categorical and continuous input and output variables.
- In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) based on most significant splitter/differentiator in input variables.

# Decision trees

# A decision tree example

Training examples: **9 yes / 5 no**

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

New data:

| Day | Outlook | Humidity | Wind | Play |
|---|---|---|---|---|
| D15 | Rain | High | Weak | ? |

Figure 1: Playing tennis?

# A decision tree example

Training examples: **9 yes / 5 no**

| Day | Outlook | Humidity | Wind | Play |
|---|---|---|---|---|
| D1 | Sunny | High | Weak | No |
| D2 | Sunny | High | Strong | No |
| D3 | Overcast | High | Weak | Yes |
| D4 | Rain | High | Weak | Yes |
| D5 | Rain | Normal | Weak | Yes |
| D6 | Rain | Normal | Strong | No |
| D7 | Overcast | Normal | Strong | Yes |
| D8 | Sunny | High | Weak | No |
| D9 | Sunny | Normal | Weak | Yes |
| D10 | Rain | Normal | Weak | Yes |
| D11 | Sunny | Normal | Strong | Yes |
| D12 | Overcast | High | Strong | Yes |
| D13 | Overcast | Normal | Weak | Yes |
| D14 | Rain | High | Strong | No |
| New data: | | | | |
| D15 | Rain | High | Weak | ? |

Figure 2: Playing tennis?

# A decision tree example

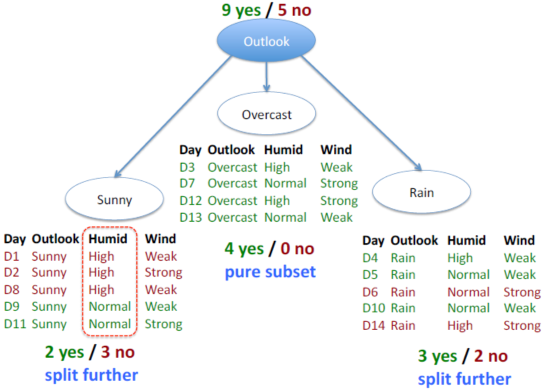



Figure 3: Playing tennis?

# A decision tree example

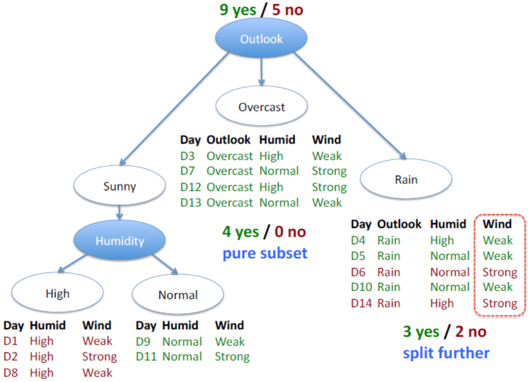



Figure 4: Playing tennis?

# A decision tree example

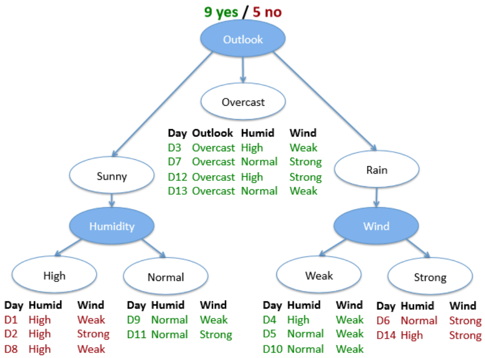



Figure 5: Playing tennis?

# A decision tree example

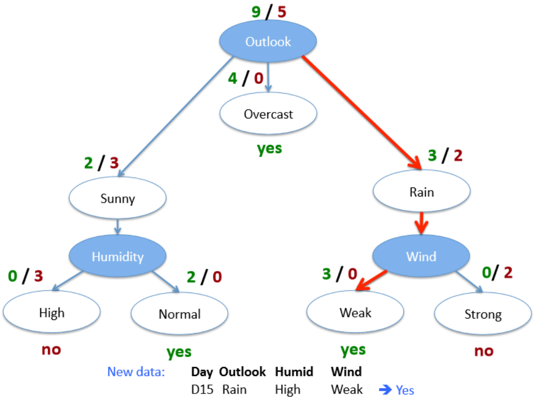



Figure 6: Playing tennis?

# Types of decision trees

- **Classification decision tree:** Decision trees which have categorical target variable

# Types of decision trees

- **Classification decision tree:** Decision trees which have categorical target variable
  - Models suitable for answering questions: Which category(ies)

# Types of decision trees

- **Classification decision tree:** Decision trees which have categorical target variable
  - Models suitable for answering questions: Which category(ies)
- **Regression trees:** decision trees that have continuous target variable

# Types of decision trees

- **Classification decision tree:** Decision trees which have categorical target variable
  - Models suitable for answering questions: Which category(ies)
- **Regression trees:** decision trees that have continuous target variable
  - Models suitable for answering questions: How mach, how many

# Terminology related to decision trees

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

# Terminology related to decision trees

1. **Root Node:** It represents 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.

# Terminology related to decision trees

1. **Root Node:** It represents 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 decision node.

# Terminology related to decision trees

1. **Root Node:** It represents 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 decision node.
4. **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.

# Terminology related to decision trees

1. **Root Node:** It represents 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 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 opposite process of splitting.

# Terminology related to decision trees

1. **Root Node:** It represents 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 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 opposite process of splitting.
6. **Branch / Sub-Tree:** A sub section of entire tree is called branch or sub-tree.

# Terminology related to decision trees

1. **Root Node:** It represents 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 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 opposite process of splitting.
6. **Branch / Sub-Tree:** A sub section of entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

# Terminology related to decision trees

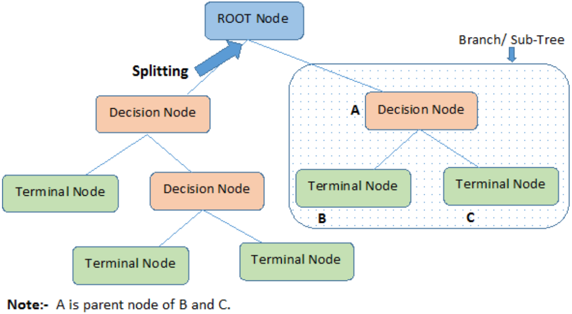



Figure 7: Trees Terminology

# Advantages & disadvantages

## Advantages

- Easy to Understand:

## Disadvantages

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:

## Disadvantages

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:
- Less data cleaning required.

## Disadvantages

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:
- Less data cleaning required.
- Data type is not a constraint.

## Disadvantages

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:
- Less data cleaning required.
- Data type is not a constraint.
- Non parametric method.

## Disadvantages

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:
- Less data cleaning required.
- Data type is not a constraint.
- Non parametric method.

## Disadvantages

- Over fitting

# Advantages & disadvantages

## Advantages

- Easy to Understand:
- Useful in data exploration:
- Less data cleaning required.
- Data type is not a constraint.
- Non parametric method.

## Disadvantages

- Over fitting
- Not fit for continuous variables

# How does a tree decide where to split?

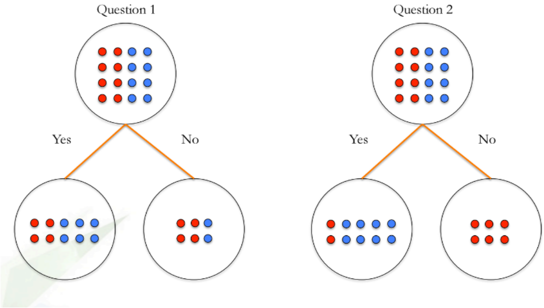
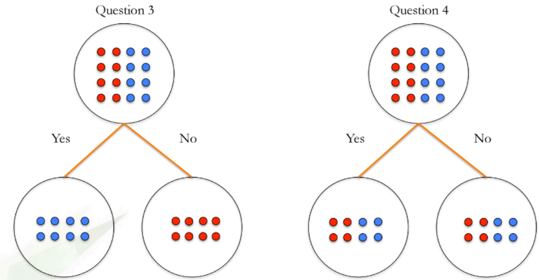



Figure 8: Tree spliting

# How does a tree decide where to split?




$$E(S) = -p_+ log(p_+) - p_- log(p_-)$$

Figure 9: Compute the entropy

# How does a tree decide where to split?

$$-\left(\frac{8}{8}\right) log_2 \left(\frac{8}{8}\right) - \left(\frac{0}{8}\right) log_2 \left(\frac{0}{8}\right) = 0$$

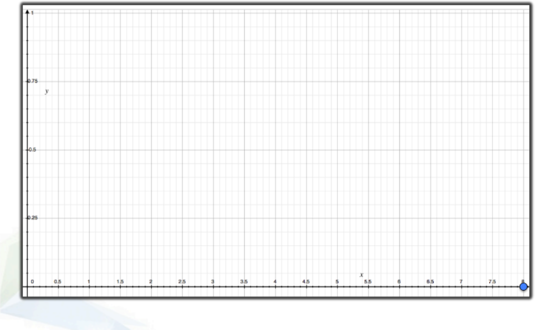



Figure 10: Compute the entropy

# How does a tree decide where to split?

$$-\left(\frac{7}{8}\right) log_2 \left(\frac{7}{8}\right) - \left(\frac{1}{8}\right) log_2 \left(\frac{1}{8}\right) = 0.54$$

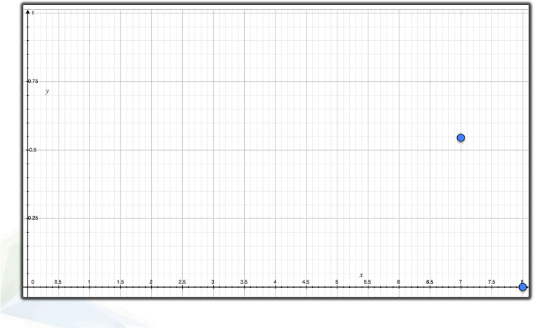


Figure 11: Compute the entropy

# How does a tree decide where to split?

$$-\left(\frac{6}{8}\right) log_2 \left(\frac{6}{8}\right) - \left(\frac{2}{8}\right) log_2 \left(\frac{2}{8}\right) = 0.81$$

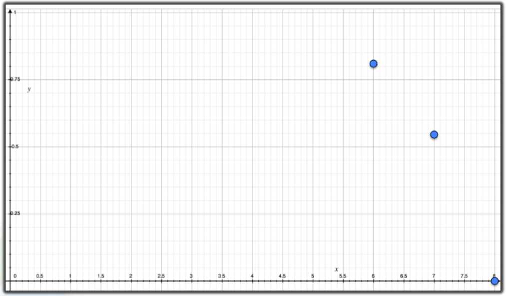



Figure 12: Compute the entropy

# How does a tree decide where to split?

$$-\left(\frac{5}{8}\right) log_2 \left(\frac{5}{8}\right) - \left(\frac{3}{8}\right) log_2 \left(\frac{3}{8}\right) = 0.95$$

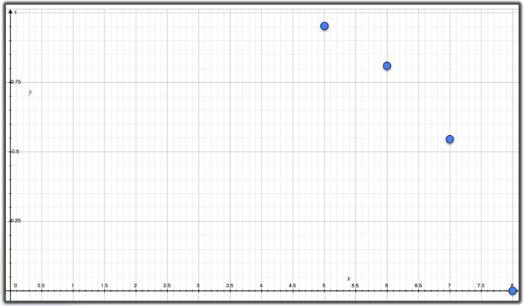



Figure 13: Compute the entropy

# How does a tree decide where to split?

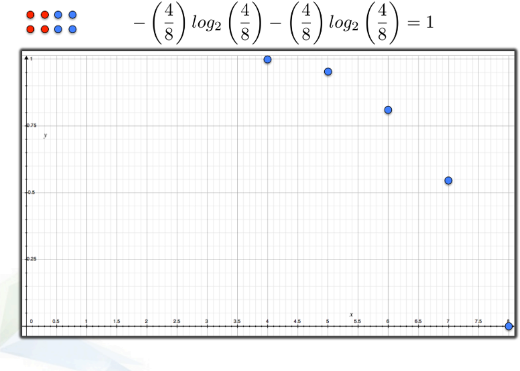



Figure 14: Compute the entropy

# How does a tree decide where to split?

$$-\left(\frac{3}{8}\right) log_2 \left(\frac{3}{8}\right) - \left(\frac{5}{8}\right) log_2 \left(\frac{5}{8}\right) = 0.95$$

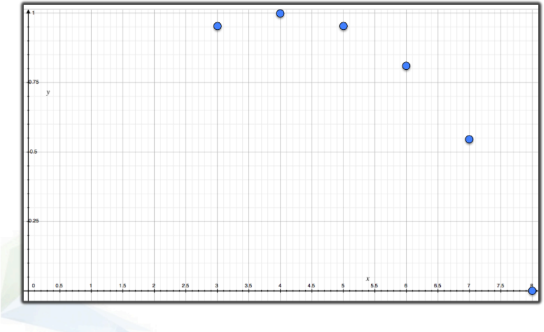



Figure 15: Compute the entropy

# How does a tree decide where to split?

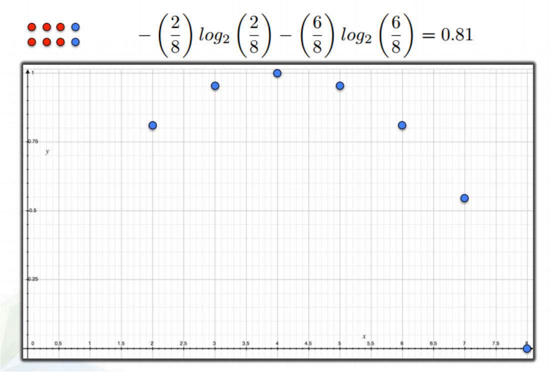



Figure 16: Compute the entropy

# How does a tree decide where to split?

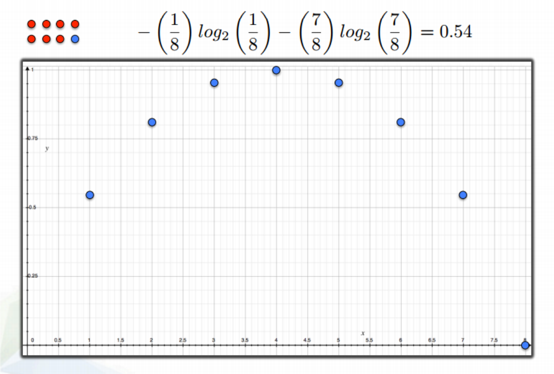



Figure 17: Compute the entropy

# How does a tree decide where to split?

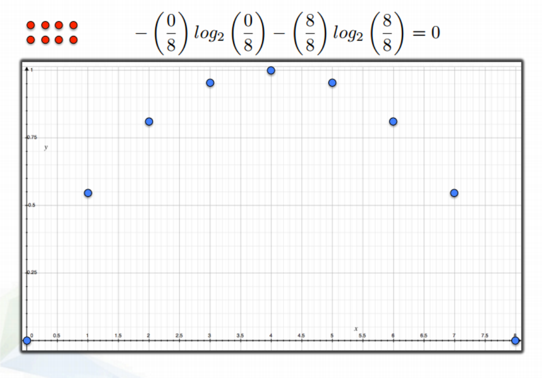



Figure 18: Compute the entropy

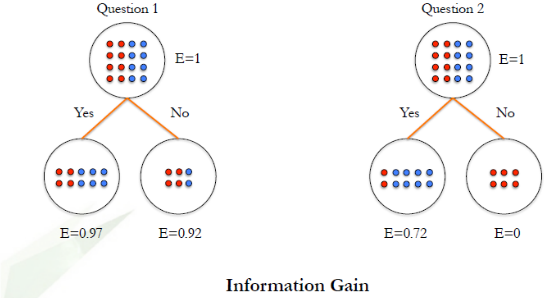
# How does a tree decide where to split?

$$y = -\sum_{i=1}^{k} p_i log_k(p_i)$$

$$y = -\underbrace{\left[\left(\frac{1}{10}\right) log_4 \left(\frac{1}{10}\right)\right]}_{\text{Red}} - \underbrace{\left[\left(\frac{3}{10}\right) log_4 \left(\frac{3}{10}\right)\right]}_{\text{Green}} - \underbrace{\left[\left(\frac{2}{10}\right) log_4 \left(\frac{2}{10}\right)\right]}_{\text{Blue}} - \underbrace{\left[\left(\frac{4}{10}\right) log_4 \left(\frac{4}{10}\right)\right]}_{\text{Yellow}}$$

Figure 19: Compute the entropy

# How does a tree decide where to split?




$$G(S,Q) = E(S) - \sum_{i=1}^{k} p_i E(S,Q_i)$$

Figure 20: Compute information gain

# How does a tree decide where to split?

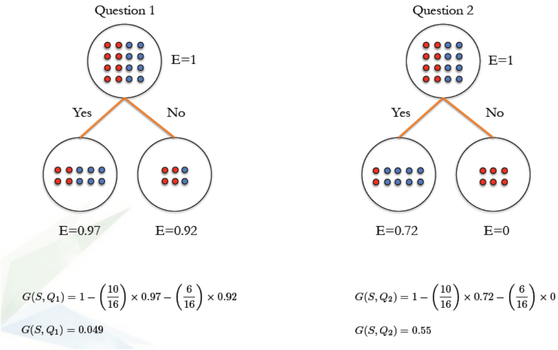



Figure 21: Compute information gain

# How does a tree decide where to split?

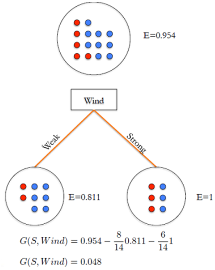



Figure 22: Compute information gain

# How does a tree decide where to split?

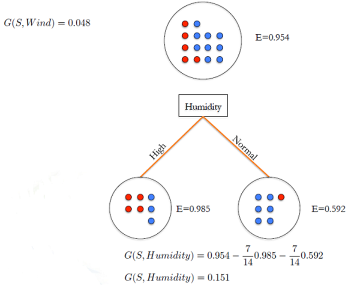



Figure 23: Compute information gain

# How does a tree decide where to split?

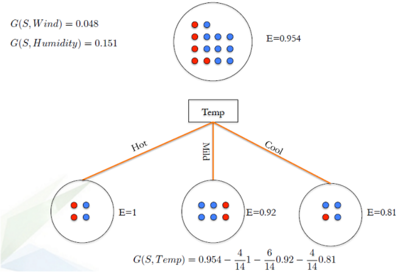



Figure 24: Compute information gain

# How does a tree decide where to split?

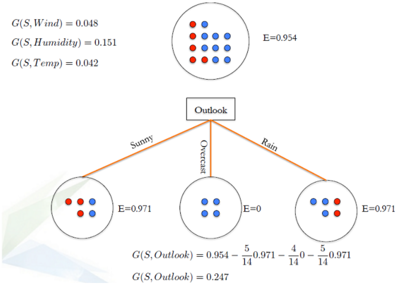



Figure 25: Compute information gain

## key parameters of tree modeling

- Overfitting is one of the key challenges faced while modeling decision trees.

# key parameters of tree modeling

- Overfitting is one of the key challenges faced while modeling decision trees.
- If no limit set, tree give you 100% accuracy on training set

# key parameters of tree modeling

- Overfitting is one of the key challenges faced while modeling decision trees.
- If no limit set, tree give you 100% accuracy on training set
- Preventing overfitting is essential in fitting a decision tree and it can be done in 2 ways:

# key parameters of tree modeling

- Overfitting is one of the key challenges faced while modeling decision trees.
- If no limit set, tree give you 100% accuracy on training set
- Preventing overfitting is essential in fitting a decision tree and it can be done in 2 ways:
  - Setting constraints on tree size

# key parameters of tree modeling

- Overfitting is one of the key challenges faced while modeling decision trees.
- If no limit set, tree give you 100% accuracy on training set
- Preventing overfitting is essential in fitting a decision tree and it can be done in 2 ways:
  - Setting constraints on tree size
  - Tree pruning

# Setting constraints on tree size

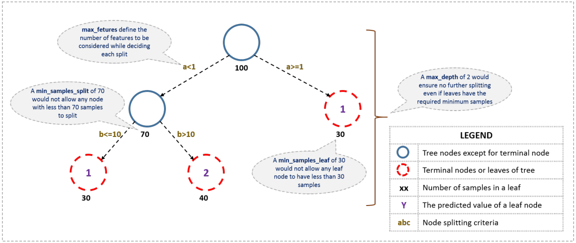



Figure 26: constraints on tree size

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
   - Control over-fitting similar to min_samples_split.

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
   - Control over-fitting similar to min_samples_split.
3. Maximum depth of tree (vertical depth, `max_depth`)

## Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
   - Control over-fitting similar to min_samples_split.
3. Maximum depth of tree (vertical depth, `max_depth`)
   - Control over-fitting Should be tuned using CV

## Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
   - Control over-fitting similar to min_samples_split.
3. Maximum depth of tree (vertical depth, `max_depth`)
   - Control over-fitting Should be tuned using CV
4. Maximum number of terminal nodes

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
    - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
    - Control over-fitting similar to min_samples_split.
3. Maximum depth of tree (vertical depth, `max_depth`)
    - Control over-fitting Should be tuned using CV
4. Maximum number of terminal nodes
    - Can be defined in place of `max_depth`. In a binary tree, a depth of 'n' would produce a maximum of $2^{n+1} - 1$ leaves.

# Setting constraints on tree size

1. Minimum samples for a node split (`min_samples_split`)
   - Control over-fitting. Should be tuned using CV.
2. Minimum samples for a terminal node (leaf)
   - Control over-fitting similar to min_samples_split.
3. Maximum depth of tree (vertical depth, `max_depth`)
   - Control over-fitting Should be tuned using CV
4. Maximum number of terminal nodes
   - Can be defined in place of `max_depth`. In a binary tree, a depth of 'n' would produce a maximum of $2^{n+1} - 1$ leaves.
5. Maximum features to consider for split

# Tree pruning

1. Make the decision tree to a large depth.

Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we will see that the overall gain is $+10$ and keep both leaves.

# Tree pruning

1. Make the decision tree to a large depth.
2. Start at the bottom and start removing leaves which are giving us negative IG when compared from the top.

Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we will see that the overall gain is $+10$ and keep both leaves.

# Are tree based models better than logistic models?

- If the relationship between feature and label is well approximated by a linear model, linear regression will outperform tree based model.

# Are tree based models better than logistic models?

- If the relationship between feature and label is well approximated by a linear model, linear regression will outperform tree based model.
- If there is a high non-linearity andcomplex relationship between feature and label tree model will outperform a classical regression method.

# Are tree based models better than logistic models?

- If the relationship between feature and label is well approximated by a linear model, linear regression will outperform tree based model.
- If there is a high non-linearity andcomplex relationship between feature and label tree model will outperform a classical regression method.
- If you need to build a model which is easy to explain to people, a decision tree model will always do better than a linear model. Decision tree models are even simpler to interpret than linear regression!

# Working with decision trees in R

**Go to the notebook**