Minería de Datos

Árboles de Decisión para Clasificación

Máster en Ciencia de Datos

Con la colaboración de:

Rodrigo García Manzanas (rodrigo.manzanas@unican.es)
Departamento de Matemática Aplicada y Ciencias de la Computación
Universidad de Cantabria

Objectives

- Introduce the basic concepts, terminology and importance of decision trees in data mining.
- Present the division algorithms (ID3, C4.5 and CART), highlighting their differences.
- Explain how decision trees work in **classification** problems (regression problems will be studied in future sessions).
- Introduce the importance of pruning techniques to prevent overfitting.
- Provide practical experience in building and evaluating decision trees (for classification problems) using software tools.

In **classification** problems...

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

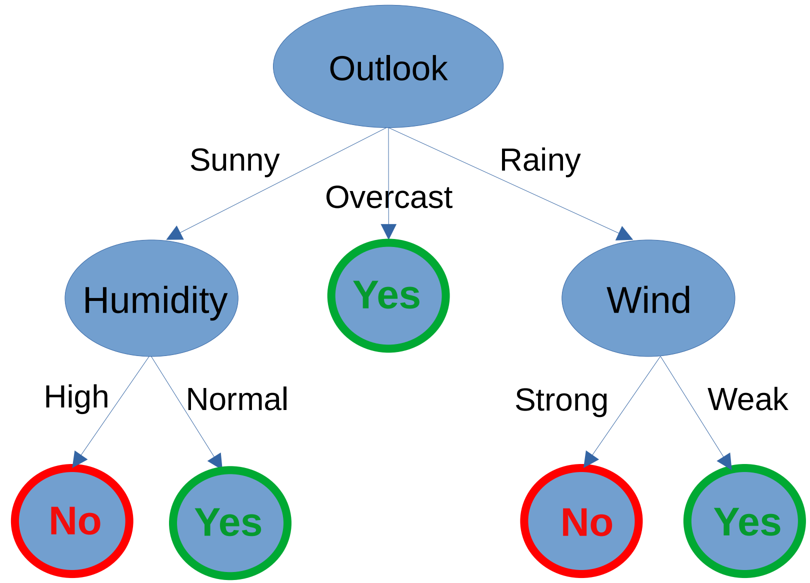
Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

How:
By iteratively partitioning the predictors' space, leading to a hierarchical (inverted) tree-like structure in which:

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



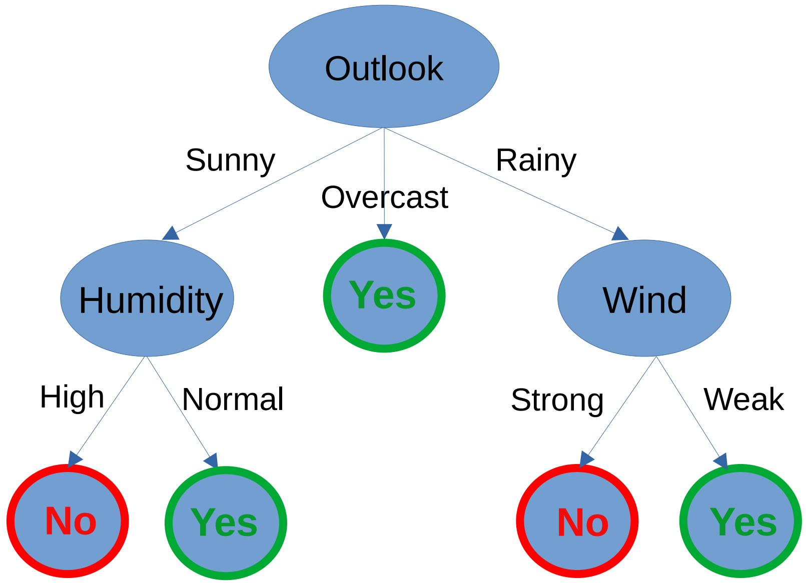
In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

How:
By iteratively partitioning the predictors' space, leading to a hierarchical (inverted) tree-like structure in which:

- Each **node** corresponds to a test on an **attribute** (i.e., a predictor)

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



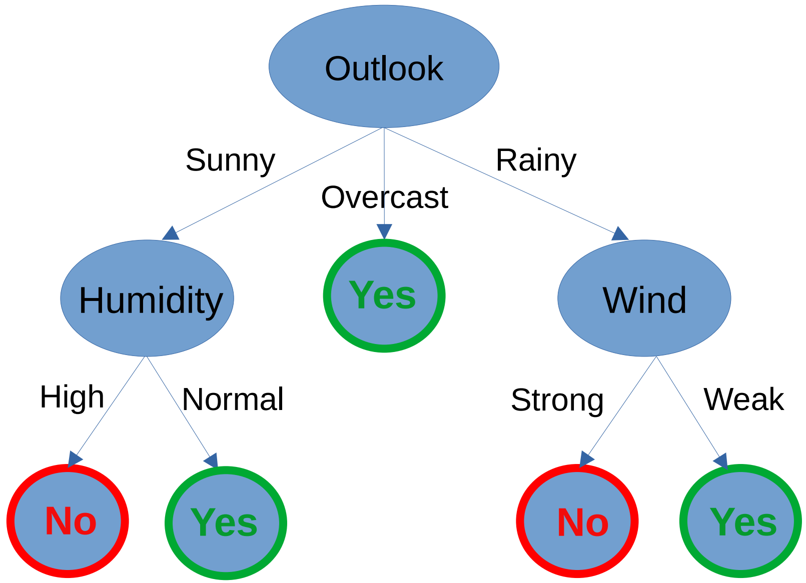
In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

How:
By iteratively partitioning the predictors' space, leading to a hierarchical (inverted) tree-like structure in which:

- Each **node** corresponds to a test on an **attribute** (i.e., a predictor)
- For a given node, each **branch** corresponds to a particular value of the tested attribute

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



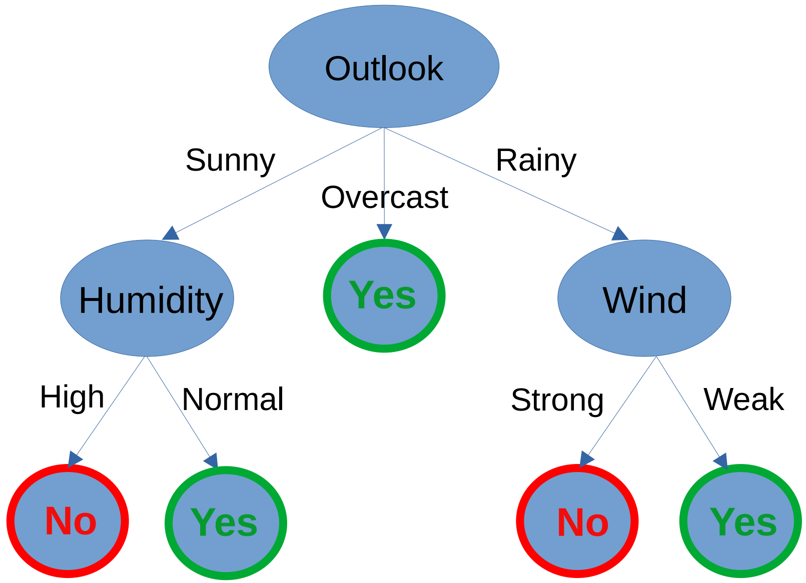
In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

How:
By iteratively partitioning the predictors' space, leading to a hierarchical (inverted) tree-like structure in which:

- Each **node** corresponds to a test on an **attribute** (i.e., a predictor)
- For a given node, each **branch** corresponds to a particular value of the tested attribute
- Each **leaf** (terminal node) represents a final class

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

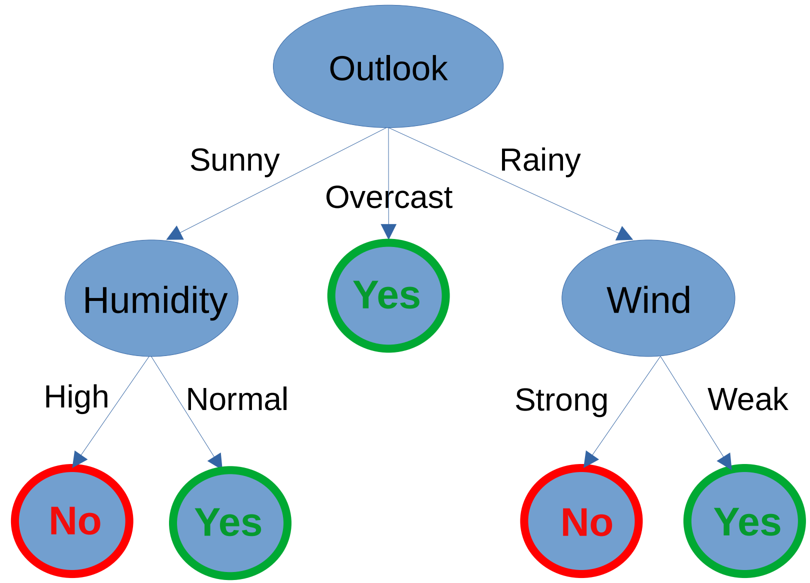
How:
By iteratively partitioning the predictors' space, leading to a hierarchical (inverted) tree-like structure in which:

- Each **node** corresponds to a test on an **attribute** (i.e., a predictor)
- For a given node, each **branch** corresponds to a particular value of the tested attribute
- Each **leaf** (terminal node) represents a final class

Pros:

- Intuitive: Can be represented graphically
- Are built quite fast
- Provide reasonably good results

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



In **classification** problems…

Aim:
To classify a **categorical** target variable based on a set of **categorical or continuous** predictors

How:
By iteratively partitioning the predictors’ space, leading to a hierarchical (inverted) tree-like structure in which:

- Each **node** corresponds to a test on an **attribute** (i.e., a predictor)
- For a given node, each **branch** corresponds to a particular value of the tested attribute
- Each **leaf** (terminal node) represents a final class

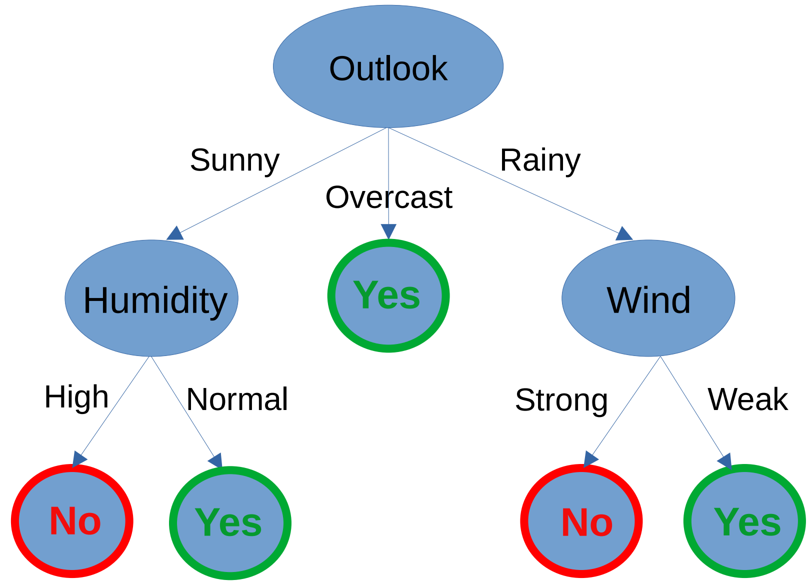
Pros:

- Intuitive: Can be represented graphically
- Are built quite fast
- Provide reasonably good results

Cons:

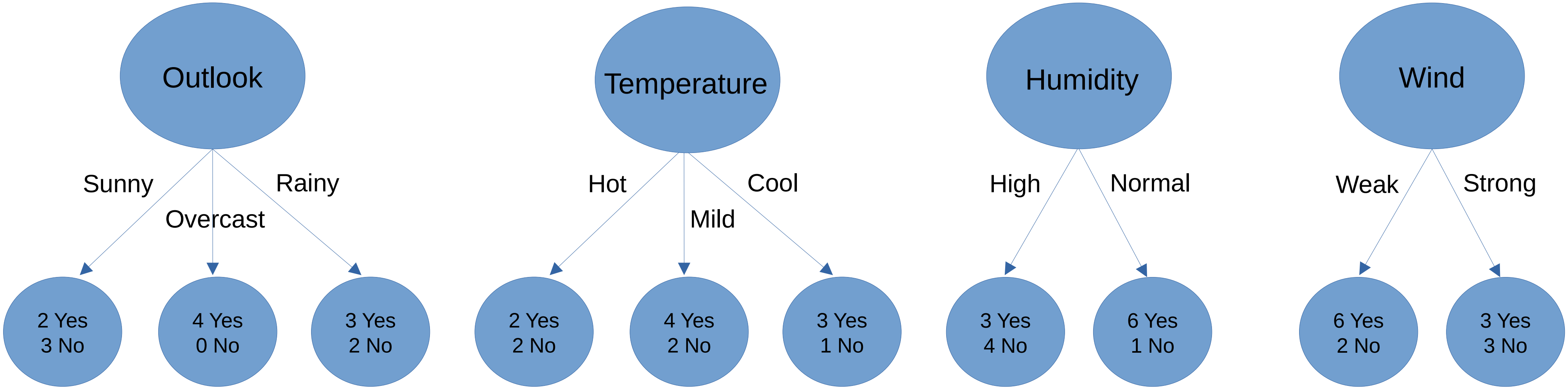
- Lack of robustness: Sensitive to small modifications in the training data

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

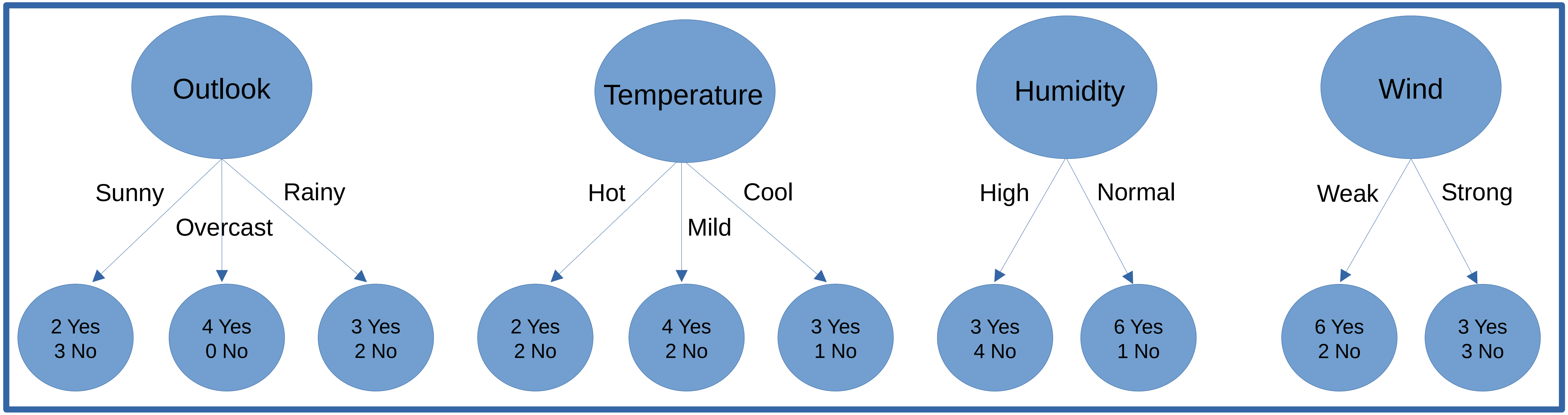


There are several algorithms to **grow the tree**. However, the idea of all of them is the same: **choose**, in each step of the process, **the attribute that best separates the target variable**

There are several algorithms to **grow the tree**. However, the idea of all of them is the same: **choose**, in each step of the process, **the attribute that best separates the target variable**

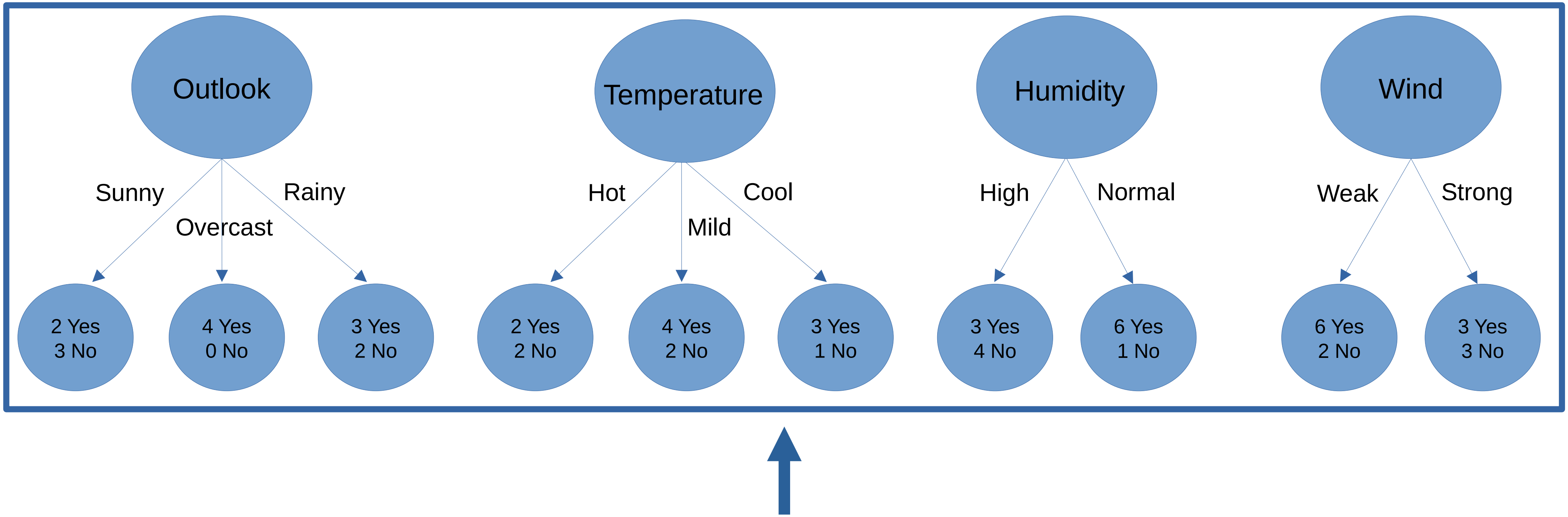


There are several algorithms to **grow the tree**. However, the idea of all of them is the same: **choose**, in each step of the process, **the attribute that best separates the target variable**



Which is the most discriminating attribute?

There are several algorithms to **grow the tree**. However, the idea of all of them is the same: **choose**, in each step of the process, **the attribute that best separates the target variable**



Which is the most discriminating attribute?

There are several algorithms to answer this question:

- **ID3** (**I**terative **D**ichotomizer), **C4.5** and **C5.0**
- **CART** (**C**lassification **A**nd **R**egression **T**rees)

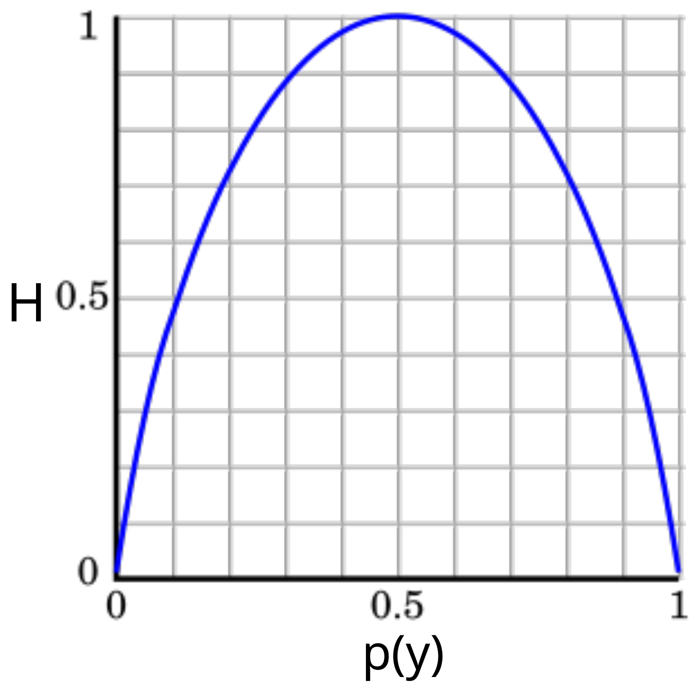
Their main difference is the criterion followed to perform the division of the node (splitting)

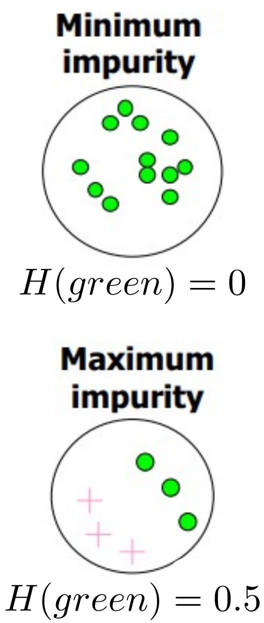
ID3 relies on the **information gain (IG)** to grow the tree. The goal is to **maximize, at each node, the IG each attribute leads to.** In other words, to select the predictor that leads to the highest reduction in entropy (H) for our target variable. H can be seen as a measure of **purity**.

$$IG(Y|X) = H(Y) - H(Y|X)$$

$$H(Y) = -\sum_{Y} p(y) log_2(p(y))$$

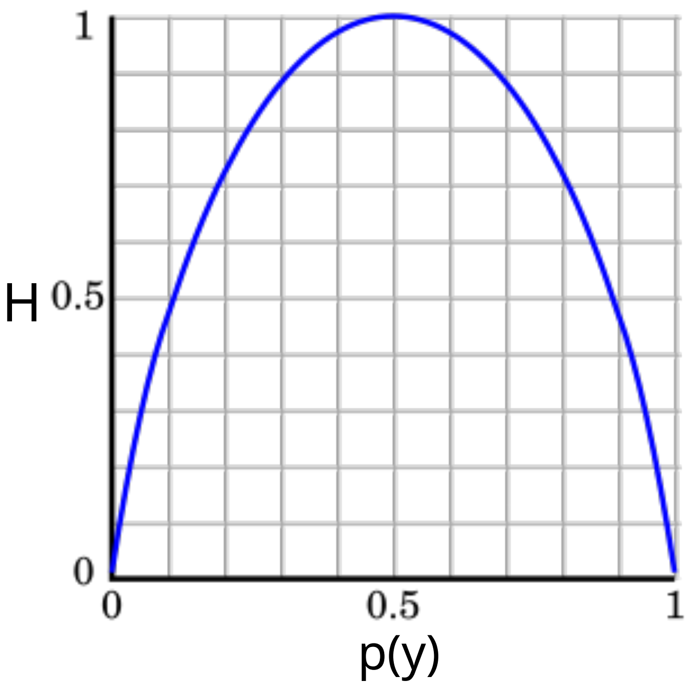
$$H(Y|X) = -\sum_{Y}\sum_{X} p(y,x) log_2(p(y|x))$$

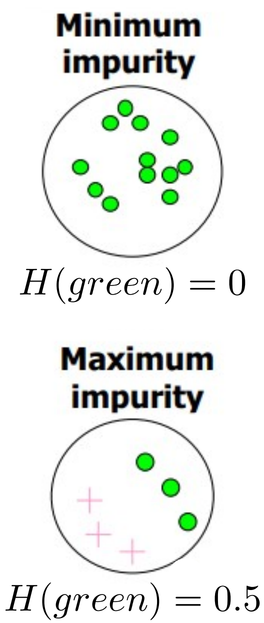




ID3 relies on the **information gain (IG)** to grow the tree. The goal is to **maximize, at each node, the IG each attribute leads to.** In other words, to select the predictor that leads to the highest reduction in entropy (H) for our target variable. H can be seen as a measure of **purity**.

$$IG(Y|X) = H(Y) - H(Y|X)$$
$$H(Y) = -\sum_{Y} p(y) log_2(p(y))$$
$$H(Y|X) = -\sum_{Y}\sum_{X} p(y,x) log_2(p(y|x))$$





So, we need to calculate:

$$IG(PT|Outlook)$$
$$IG(PT|Temperature)$$
$$IG(PT|Humidity)$$
$$IG(PT|Wind)$$

And see which is the largest one to identify the attribute with the best discriminating power.

For instance, for $IG(PT|Wind)$, we need $H(PT)$ and $H(PT|Wind)$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

$$H(PT) = -\frac{9}{14} log_2\left(\frac{9}{14}\right) - \frac{5}{14} log_2\left(\frac{5}{14}\right) = 0.940$$

For instance, for $IG(PT|Wind)$, we need $H(PT)$ and $H(PT|Wind)$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

$$H(PT|Wind) = -p(Yes, Strong)\, log_2\big(p(Yes|Strong)\big) - p(No, Strong)\, log_2\big(p(No|Strong)\big)$$
$$-p(Yes, Weak)\, log_2(p(Yes|Weak)) - p(No, Weak)\, log_2(p(No|Weak))$$

For instance, for $IG(PT|Wind)$, we need $H(PT)$ and $H(PT|Wind)$

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

$$H(PT|Wind) = -\frac{3}{14} log_2\left(\frac{3}{6}\right) - \frac{3}{14} log_2\left(\frac{3}{6}\right) - \frac{6}{14} log_2\left(\frac{6}{8}\right) - \frac{2}{14} log_2\left(\frac{2}{8}\right) = 0.892$$

$$H(PT) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.940$$

$$H(PT|Wind) = -\frac{3}{14}log_2\left(\frac{3}{6}\right) - \frac{3}{14}log_2\left(\frac{3}{14}\right) - \frac{6}{14}log_2\left(\frac{6}{8}\right) - \frac{2}{14}log_2\left(\frac{2}{14}\right) = 0.892$$

$$IG(PT|Wind) = 0.940 - 0.892 = 0.048$$

$$H(PT) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.940$$

$$H(PT|Wind) = -\frac{3}{14}log_2\left(\frac{3}{6}\right) - \frac{3}{14}log_2\left(\frac{3}{14}\right) - \frac{6}{14}log_2\left(\frac{6}{8}\right) - \frac{2}{14}log_2\left(\frac{2}{14}\right) = 0.892$$

$$IG(PT|Wind) = 0.940 - 0.892 = 0.048$$

$$IG(PT|Humidity) = 0.151$$

$$IG(PT|Outlook) = 0.246$$

$$IG(PT|Temperature) = 0.029$$

$$H(PT) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.940$$

$$H(PT|Wind) = -\frac{3}{14}log_2\left(\frac{3}{6}\right) - \frac{3}{14}log_2\left(\frac{3}{14}\right) - \frac{6}{14}log_2\left(\frac{6}{8}\right) - \frac{2}{14}log_2\left(\frac{2}{14}\right) = 0.892$$

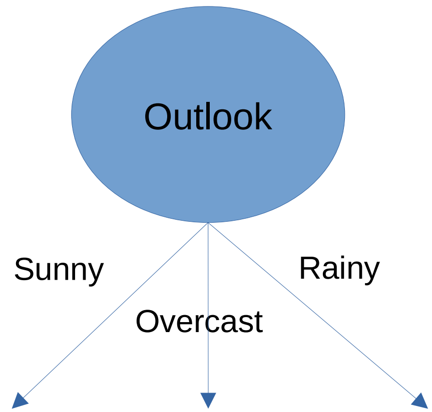
$$IG(PT|Wind) = 0.940 - 0.892 = 0.048$$

$$IG(PT|Humidity) = 0.151$$

$$IG(PT|Outlook) = 0.246$$

Root node

$$IG(PT|Temperature) = 0.029$$



$$H(PT) = -\frac{9}{14}log_2\left(\frac{9}{14}\right) - \frac{5}{14}log_2\left(\frac{5}{14}\right) = 0.940$$

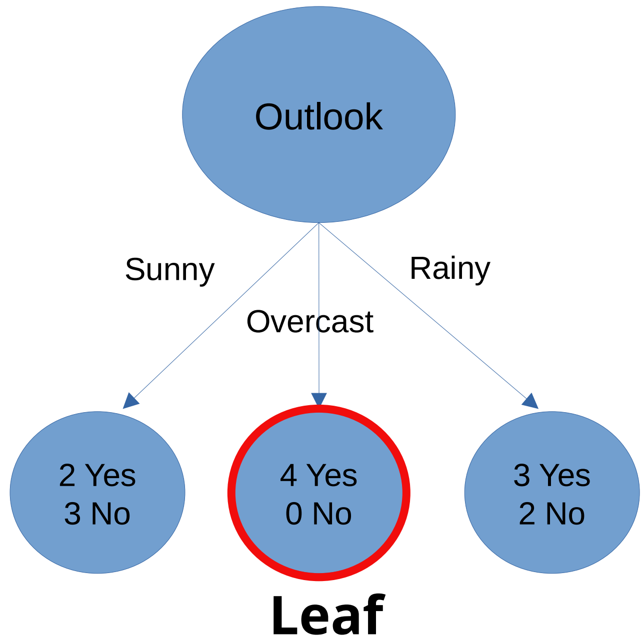
$$H(PT|Wind) = -\frac{3}{14}log_2\left(\frac{3}{6}\right) - \frac{3}{14}log_2\left(\frac{3}{14}\right) - \frac{6}{14}log_2\left(\frac{6}{8}\right) - \frac{2}{14}log_2\left(\frac{2}{14}\right) = 0.892$$

$$IG(PT|Wind) = 0.940 - 0.892 = 0.048$$

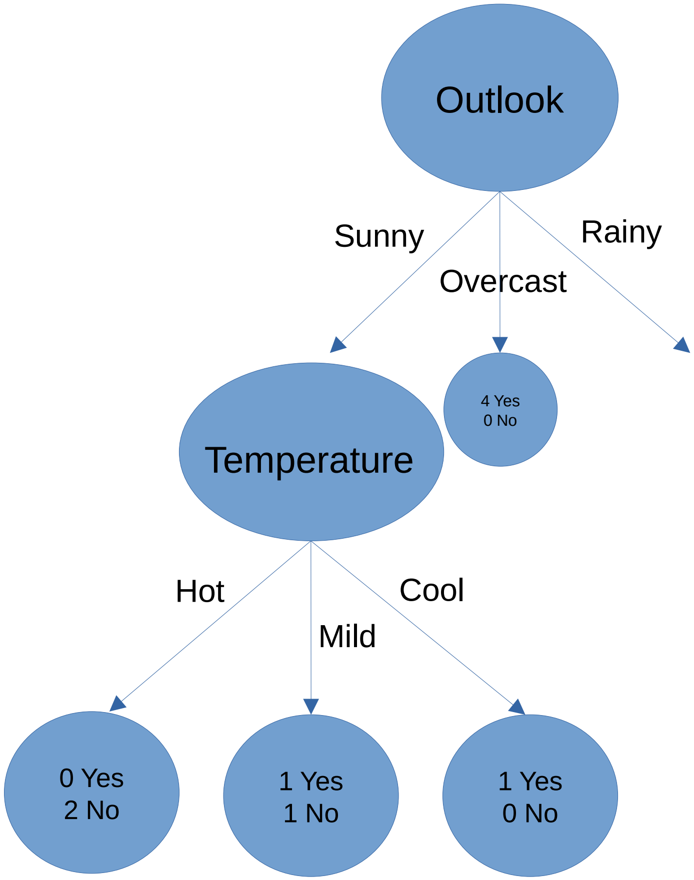
$$IG(PT|Humidity) = 0.151$$

$$IG(PT|Outlook) = 0.246$$ **Root node**

$$IG(PT|Temperature) = 0.029$$

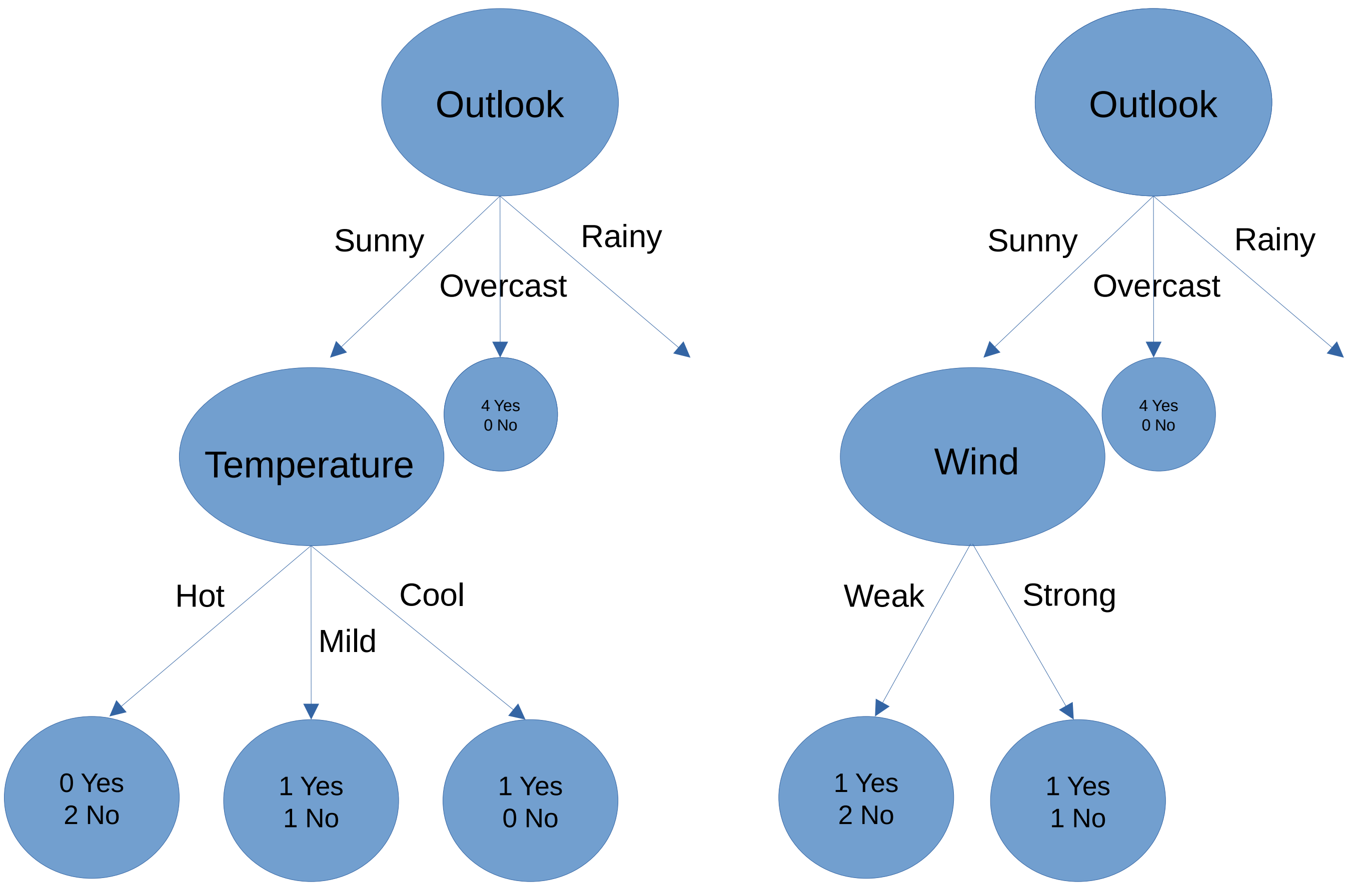


... continue to split ...



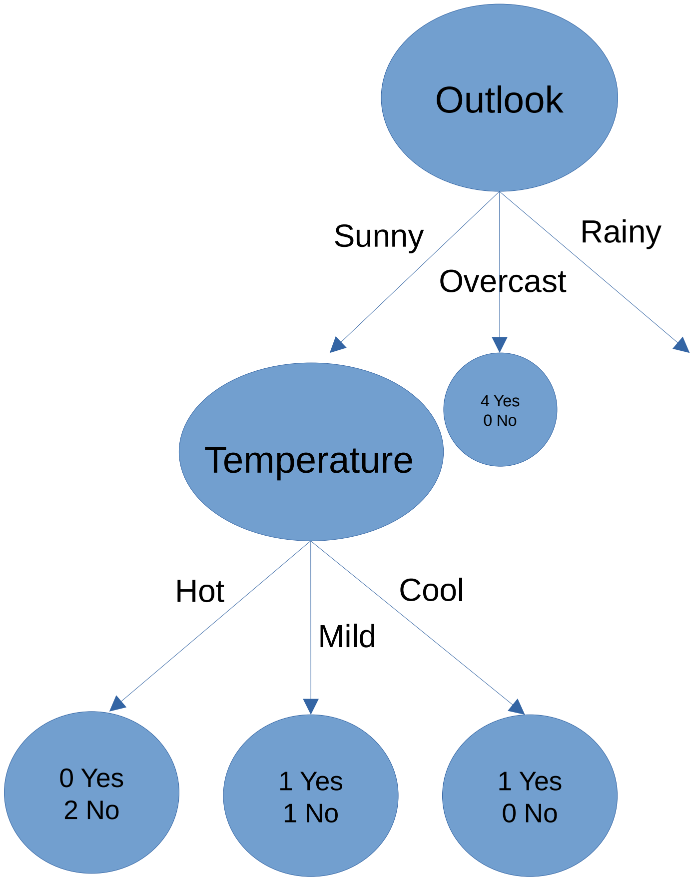
$$IG(PT|Temperature) = 0.570$$

... continue to split ...

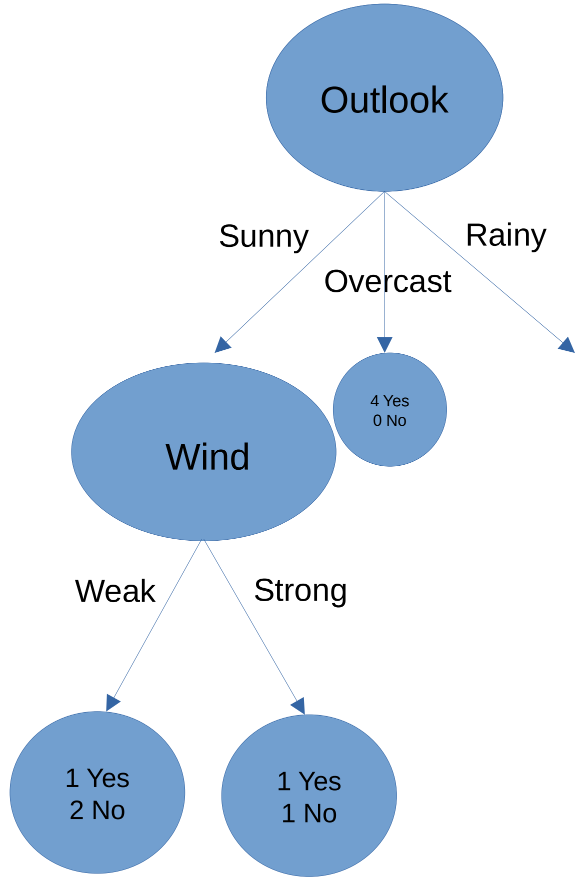


$IG(PT|Temperature) = 0.570$ $IG(PT|Wind) = 0.019$

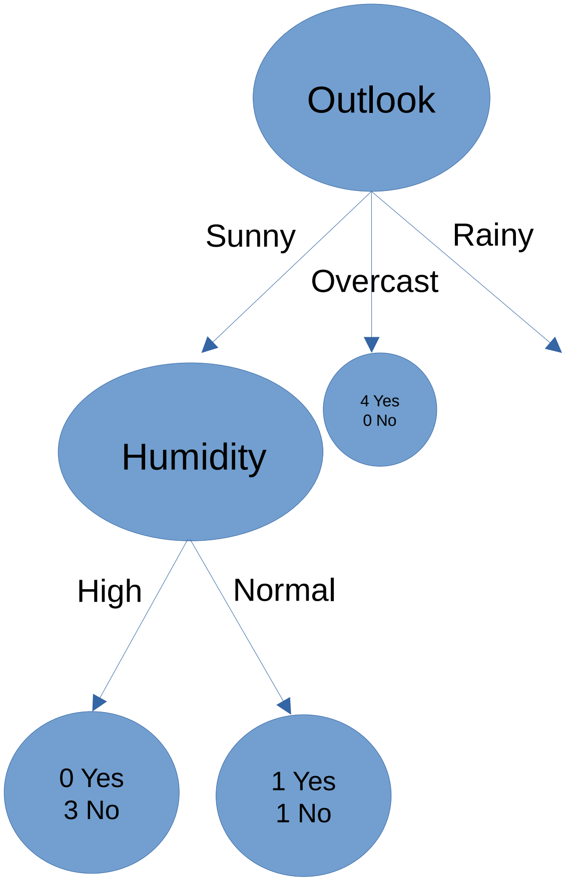
... continue to split ...



$$IG(PT|Temperature) = 0.570$$

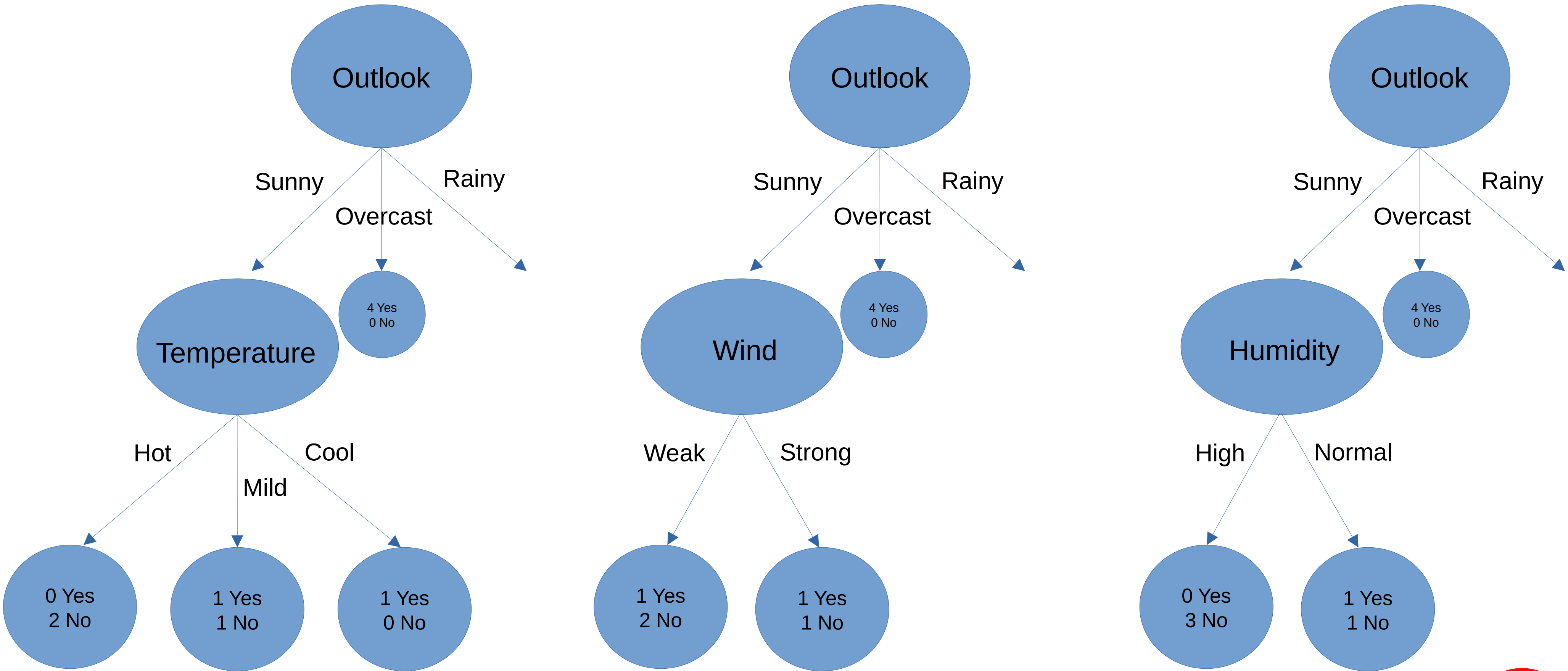


$$IG(PT|Wind) = 0.019$$



$$IG(PT|Humidity) = 0.970$$

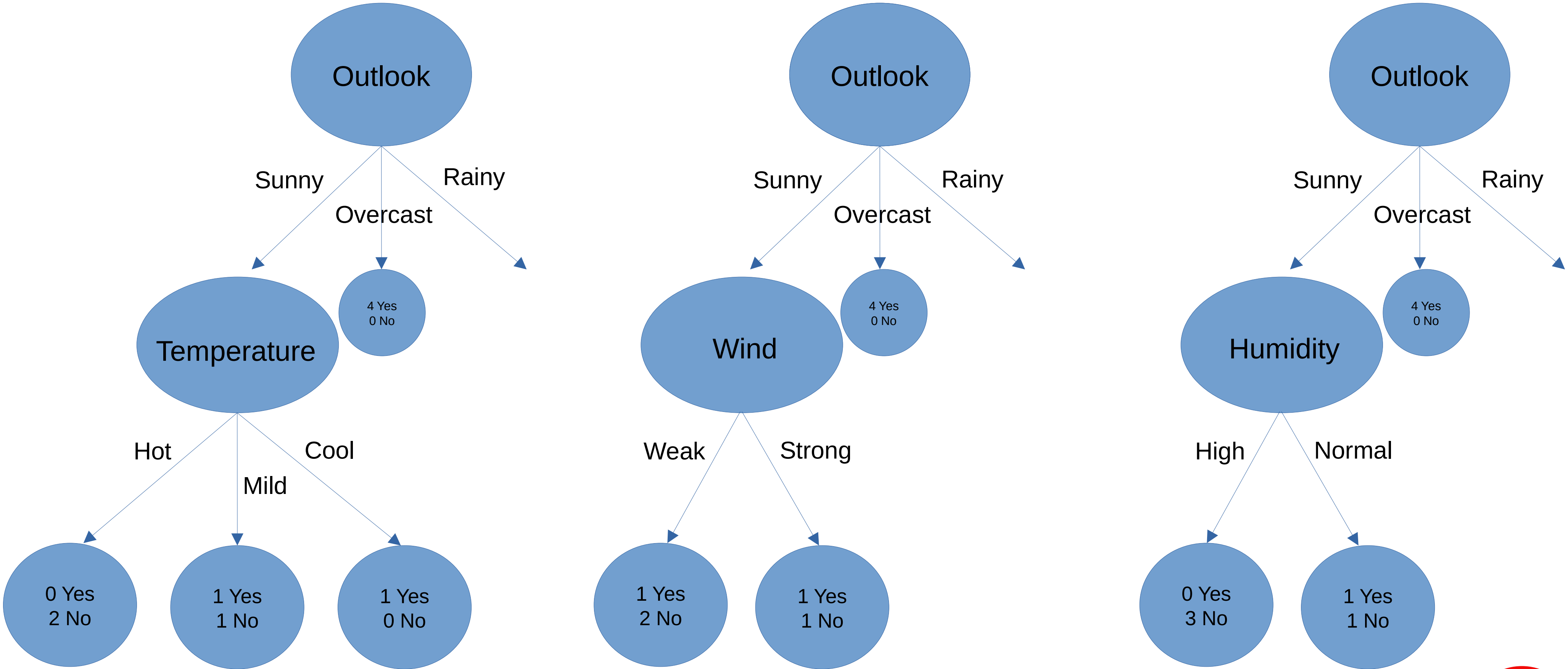
... continue to split ...



$IG(PT|Temperature) = 0.570$ $IG(PT|Wind) = 0.019$ $IG(PT|Humidity) = 0.970$

Next attribute chosen:
Humidity

... continue to split ...

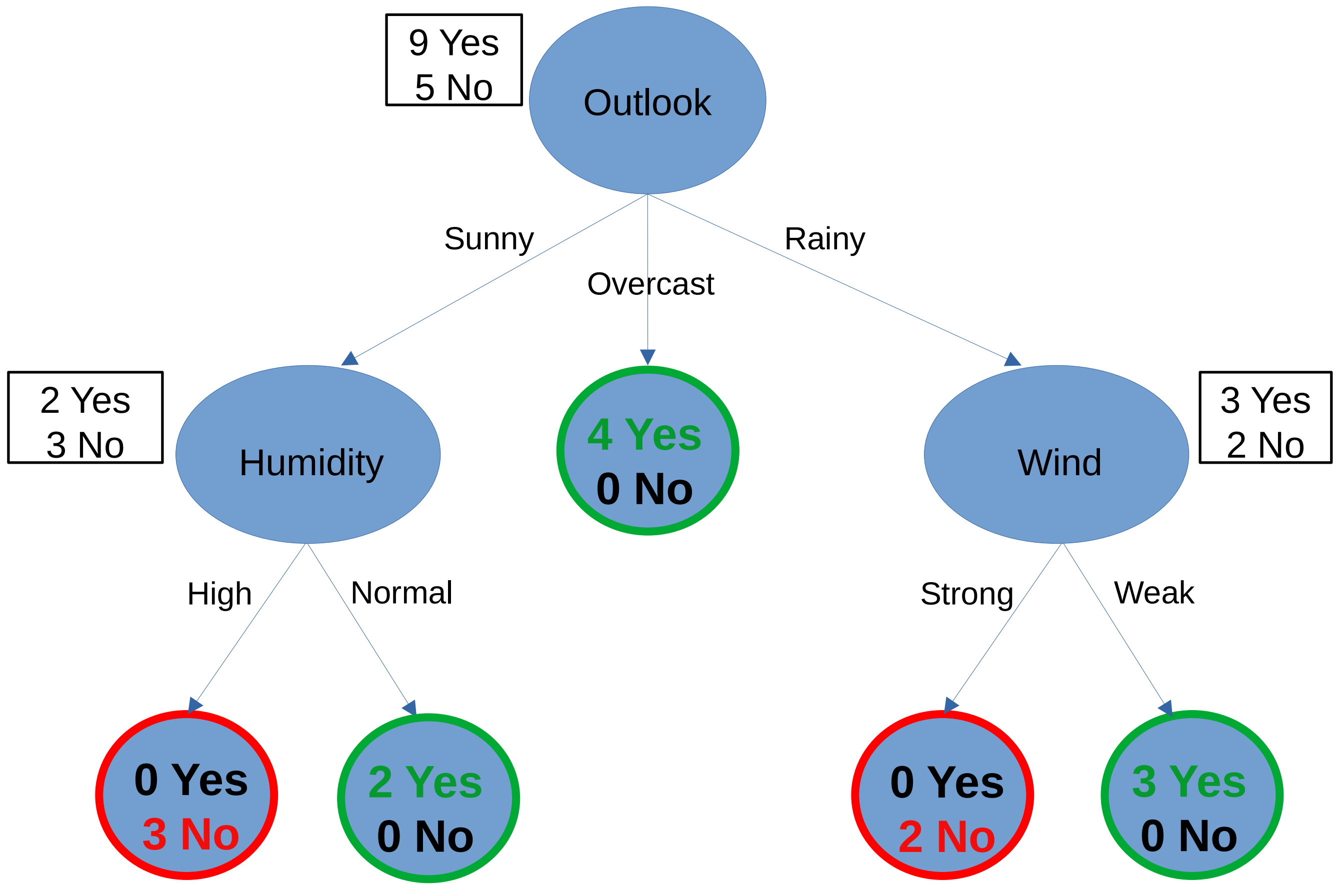


$$IG(PT|Temperature) = 0.570 \quad IG(PT|Wind) = 0.019 \quad IG(PT|Humidity) = 0.970$$

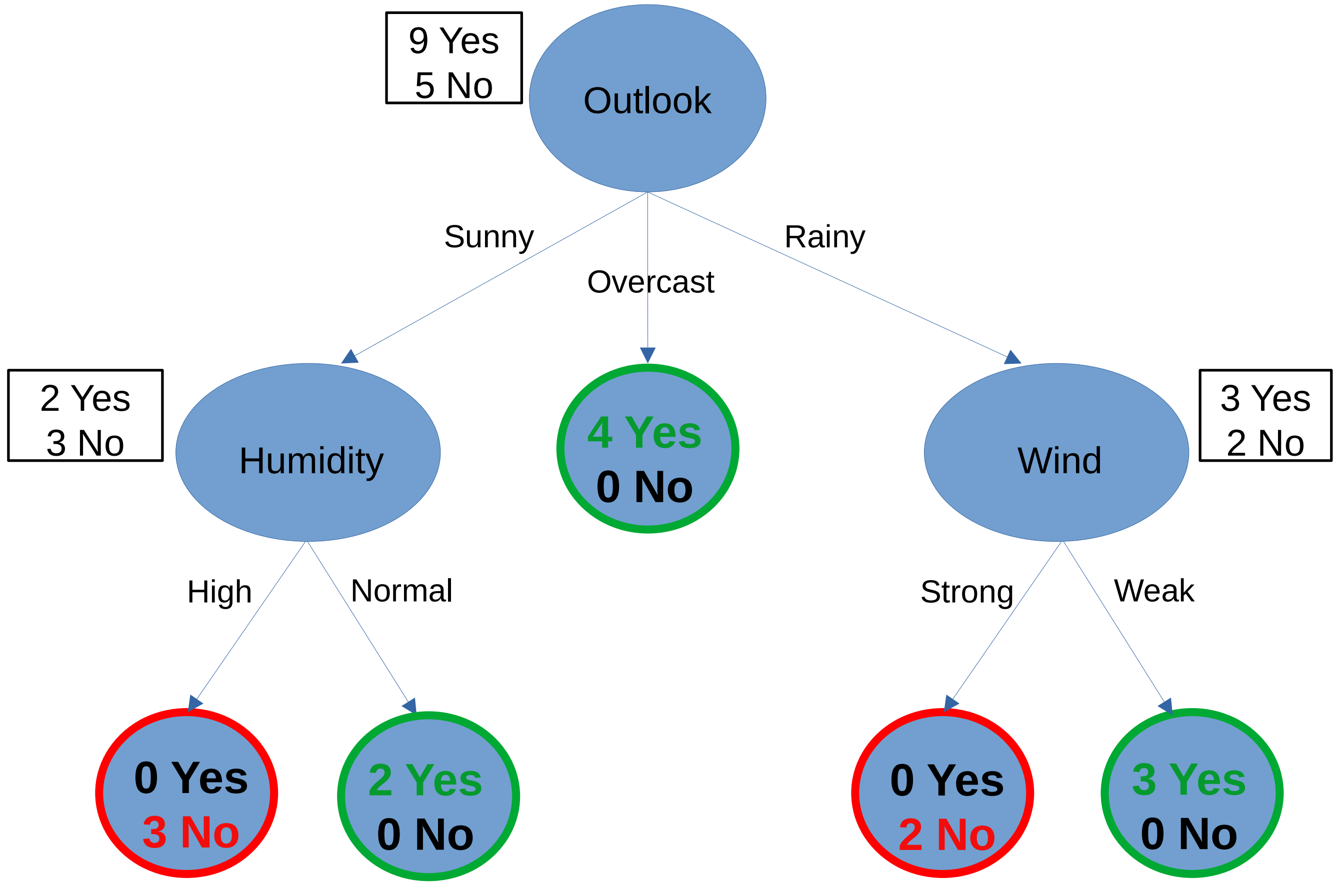
Next attribute chosen:
Humidity

... continue to split ...

... final tree!

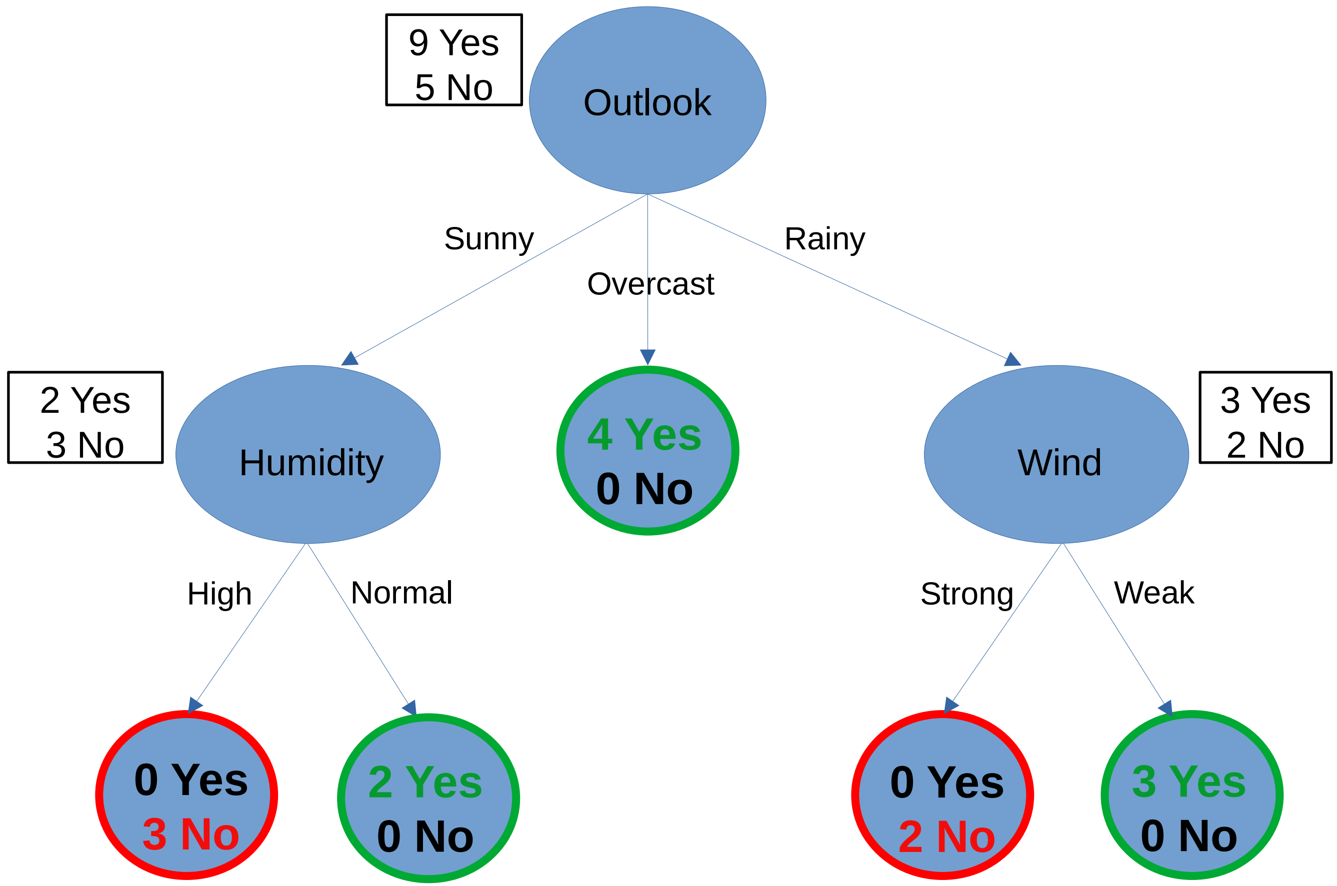

9 Yes
5 No
Outlook
Sunny
Overcast
Rainy
2 Yes
3 No
Humidity
4 Yes
0 No
Wind
3 Yes
2 No
High
Normal
Strong
Weak
0 Yes
3 No
2 Yes
0 No
0 Yes
2 No
3 Yes
0 No

... final tree!



- For a sufficiently complex (i.e. large) tree, all instances can be correctly classified (as in this case). However, this can lead to **overfitting** (we will see this later)

... final tree!



- For a sufficiently complex (i.e. large) tree, all instances can be correctly classified (as in this case). However, this can lead to **overfitting** (we will see this later)
- **If some attributes are not useful for classification, they will not be selected to grow the tree** (e.g., temperature in this case). For this reason, decision trees are often used as pre-processing tools for other learning algorithms which suffers from the presence of irrelevant information

The **information gain** is a measure that tends to prefer attributes with large number of possible values. To minimize this effect, the successor of ID3, **C4.5**, uses the **gain ratio** as partitioning criterion. In addition, this new algorithm was improved to handle with missing data and **continuous attributes** (which are splitted into **two categories** according to a **threshold value**).

Gain ratio (GR): Takes into account the **number of branches an attribute leads to**, penalizing those with many. It also penalizes attributes that lead to uniformly distributed data.
At each node, **the attribute chosen for splitting is the one that leads to the highest GR.**

$$GR(Y|X) = \frac{IG(Y|X)}{Info(X)}$$

$$\boxed{Info(X) = -\sum_{X} p(x) log_2(p(x))}$$

correction term

$$Info(Outlook) = -\frac{5}{14} log_2\left(\frac{5}{14}\right) - \frac{5}{14} log_2\left(\frac{5}{14}\right) - \frac{4}{14} log_2\left(\frac{4}{14}\right) = 1.577$$

$$GR(PT|Outlook) = -\frac{IG(PT|Outlook)}{Info(Outlook)} = \frac{0.246}{1.577} = 0.157$$

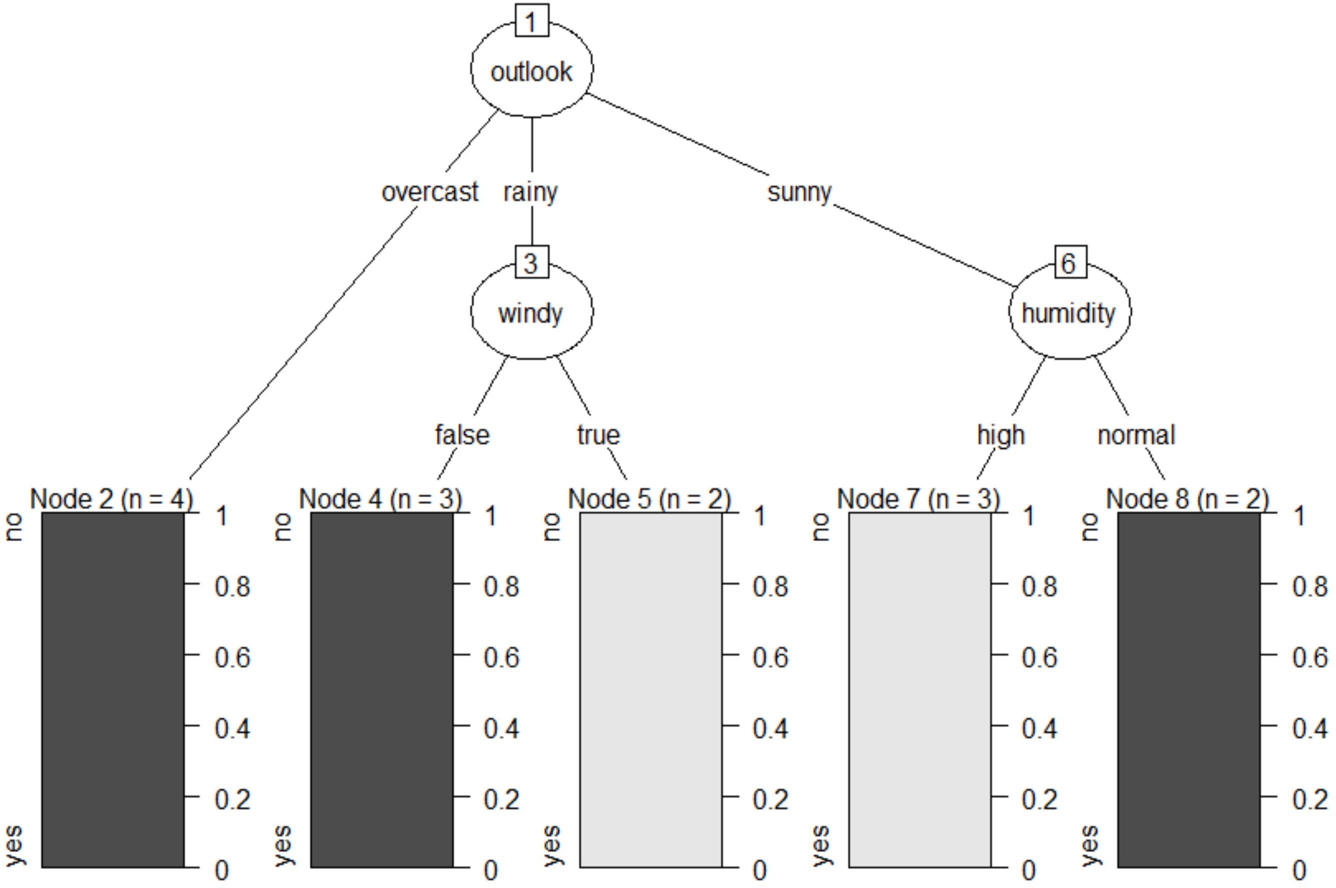
$$GR(PT|Humidity) = 0.152$$
$$GR(PT|Wind) = 0.049$$
$$GR(PT|Temperature) = 0.018$$

Attribute chosen: **Outlook**

C5.0 is just a more efficient implementation of C4.5 (faster computing times).

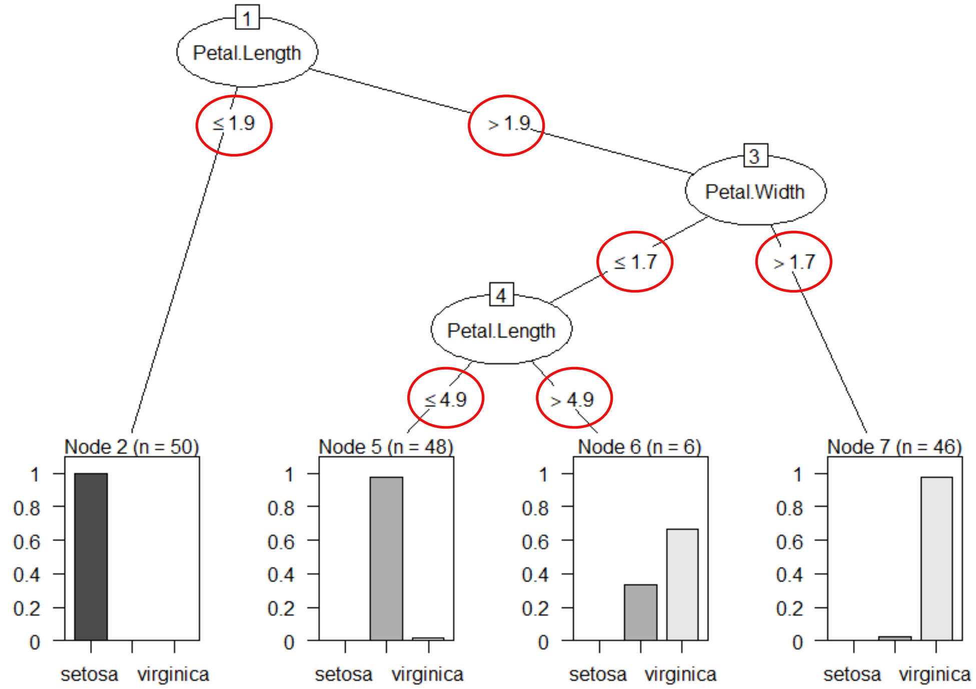
There are many packages in R to build classification tress: tree, rpart, C5.0, etc. Let's start by using ***C5.0****, which is based on the GR, for the* ***playTennis*** *dataset (****categorical attributes****)*

```
## read dataset
tennis = read.csv(“…/tennis.csv")
tennis =
as.data.frame(unclass(tennis),
stringsAsFactors = TRUE)

## grow the tree
library(C50)
t = C5.0(formula = play ~ .,
data = tennis)
## plot the tree
plot(t)
summary(t)
```

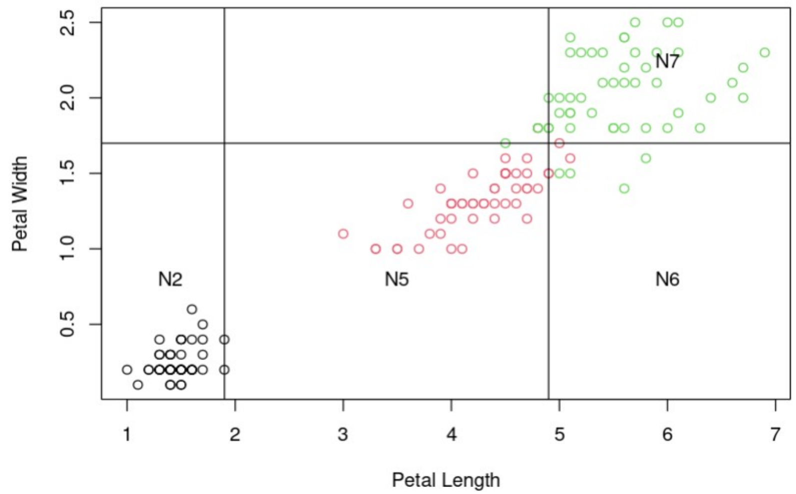


*Let's now move to the **iris** dataset (**continuous attributes**)*

```
## continuous attributes are splitted into categories based on thresholds
t = C5.0(formula = Species ~ ., data = iris)
plot(t)
summary(t)

## there are only two relevant predictors
t = C5.0(formula = Species ~ Petal.Length + Petal.Width, data = iris)
plot(t)
summary(t)
```




```
## the predictors' space is partitioned according to the thresholds determined by the tree
with(iris, plot(Petal.Length, Petal.Width, col = Species, xlab = "Petal Length", ylab = "Petal Width"))
legend("topright", levels(iris$Species), col = 1:length(levels(iris$Species)), pch = 1)
```



*Let's now move to the **iris** dataset (**continuous attributes**)*

continuous attributes are splitted into categories based on thresholds

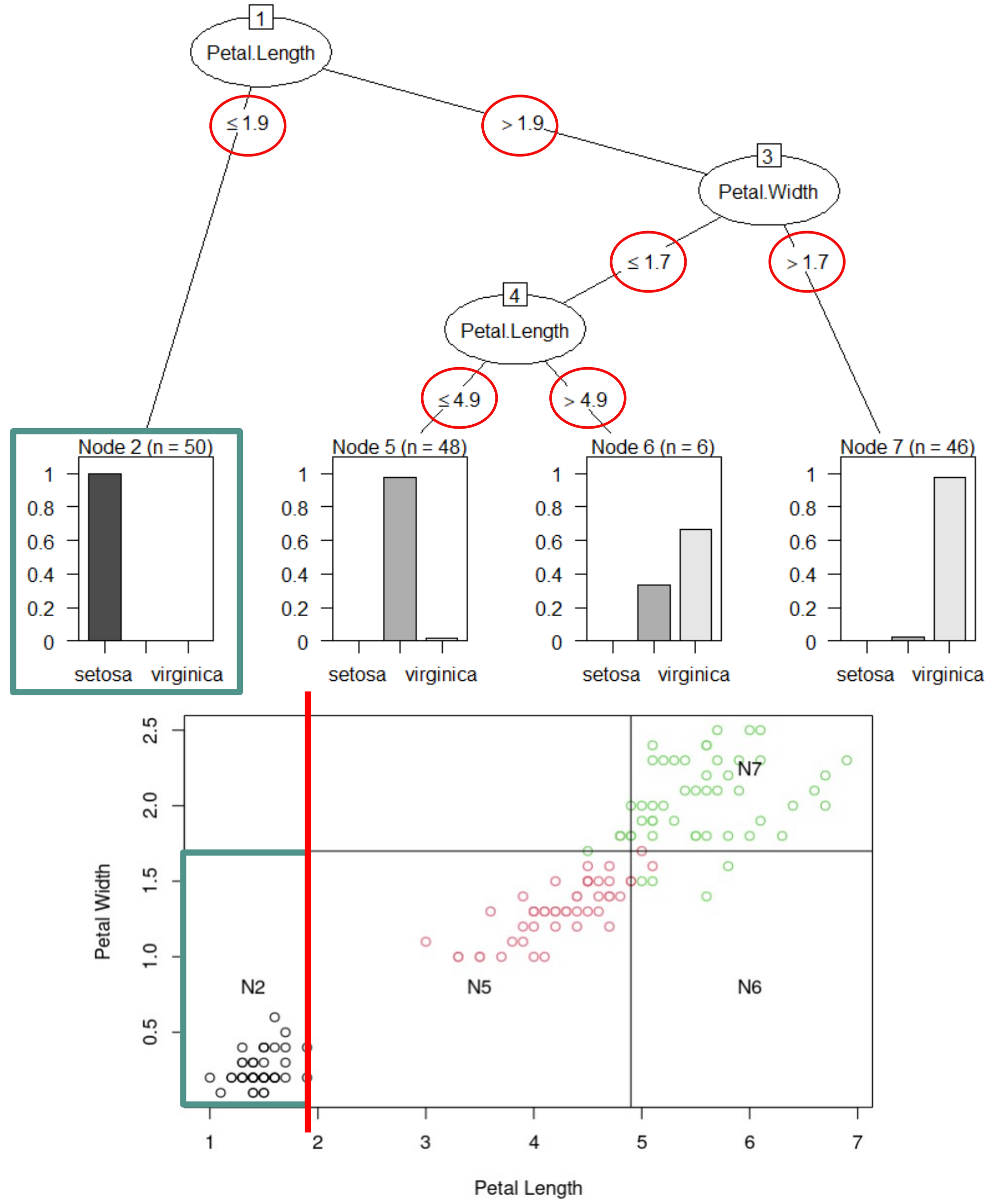
```
t = C5.0(formula = Species ~ ., data = iris)
plot(t)
summary(t)
```

there are only two relevant predictors

```
t = C5.0(formula = Species ~ Petal.Length + Petal.Width, data = iris)
plot(t)
summary(t)
```

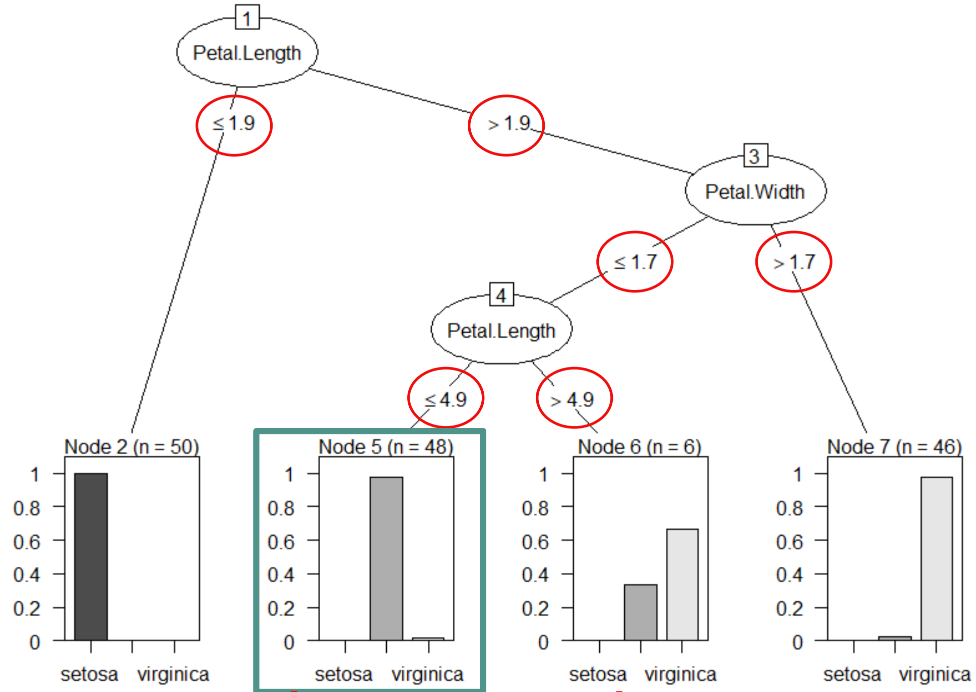
the predictors' space is partitioned according to the thresholds determined by the tree

```
with(iris, plot(Petal.Length, Petal.Width, col = Species, xlab = "Petal Length", ylab = "Petal Width"))
legend("topright", levels(iris$Species), col = 1:length(levels(iris$Species)), pch = 1)
```

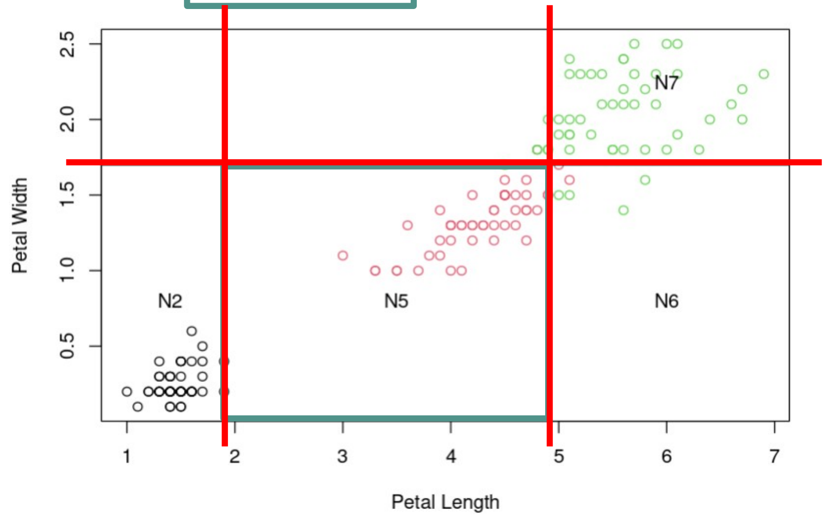


*Let's now move to the **iris** dataset (**continuous attributes**)*

```
## continuous attributes are splitted into categories based on thresholds
t = C5.0(formula = Species ~ ., data = iris)
plot(t)
summary(t)

## there are only two relevant predictors
t = C5.0(formula = Species ~ Petal.Length + Petal.Width, data = iris)
plot(t)
summary(t)
```



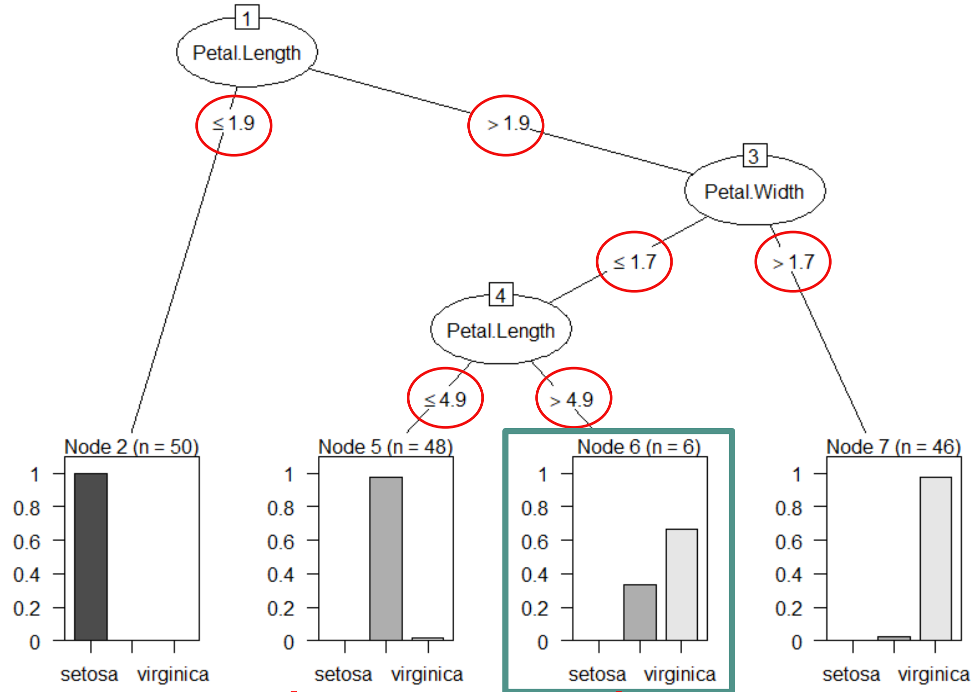

```
## the predictors' space is partitioned according to the thresholds determined by the tree
with(iris, plot(Petal.Length, Petal.Width, col = Species, xlab = "Petal Length", ylab = "Petal Width"))
legend("topright", levels(iris$Species), col = 1:length(levels(iris$Species)), pch = 1)
```



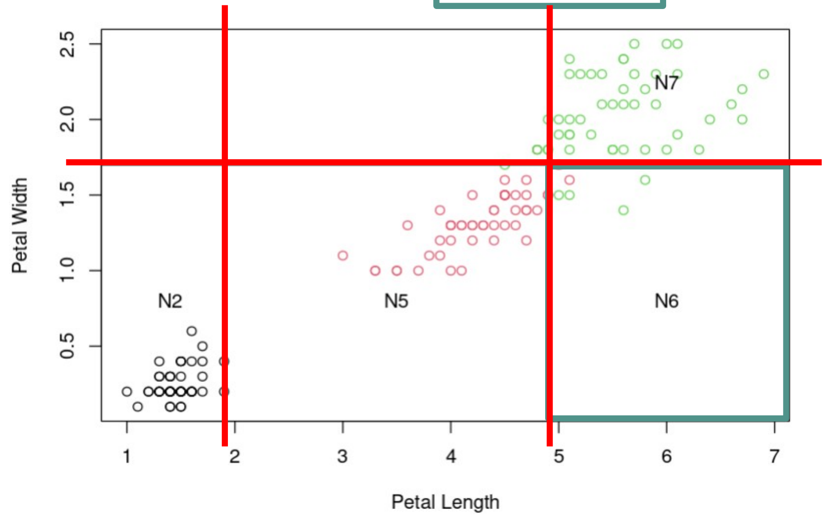
*Let's now move to the **iris** dataset (**continuous attributes**)*

continuous attributes are splitted into categories based on thresholds
t = C5.0(formula = Species ~ ., data = iris)
plot(t)
summary(t)

there are only two relevant predictors
t = C5.0(formula = Species ~ Petal.Length + Petal.Width, data = iris)
plot(t)
summary(t)



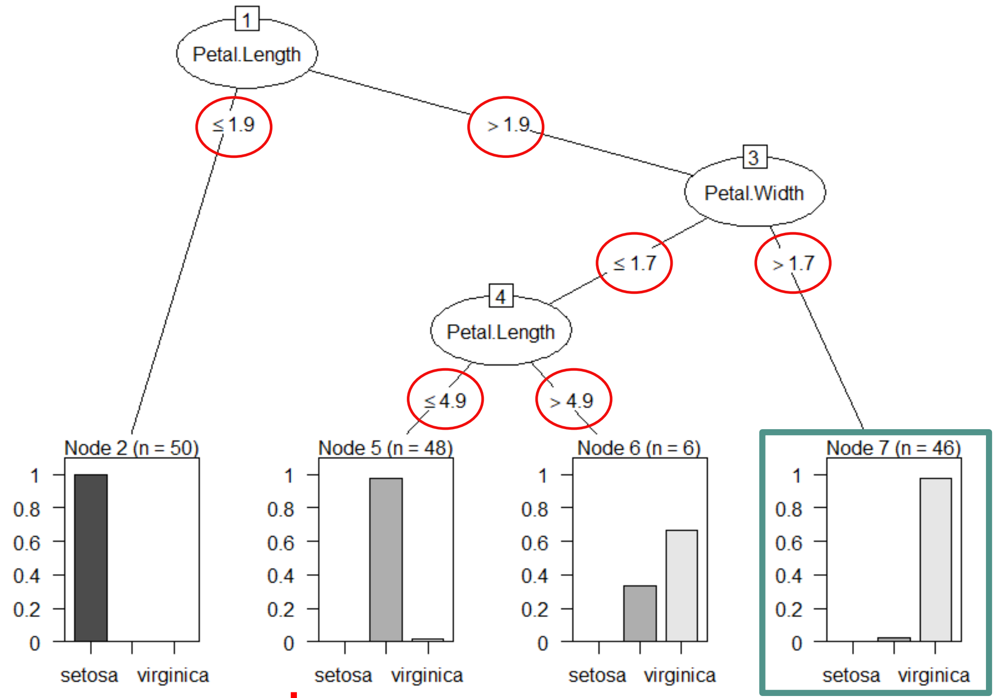
the predictors' space is partitioned according to the thresholds determined by the tree
with(iris, plot(Petal.Length, Petal.Width, col = Species, xlab = "Petal Length", ylab = "Petal Width"))
legend("topright", levels(iris$Species), col = 1:length(levels(iris$Species)), pch = 1)



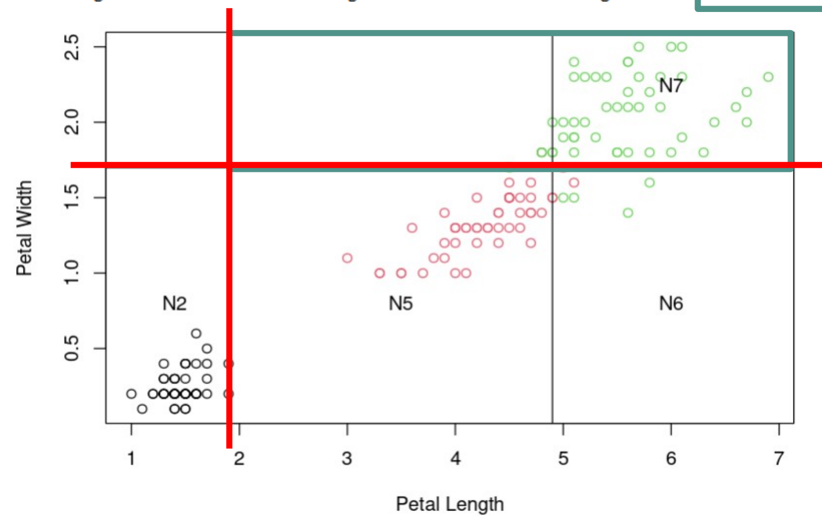
Examples in R

*Let's now move to the **iris** dataset (**continuous attributes**)*

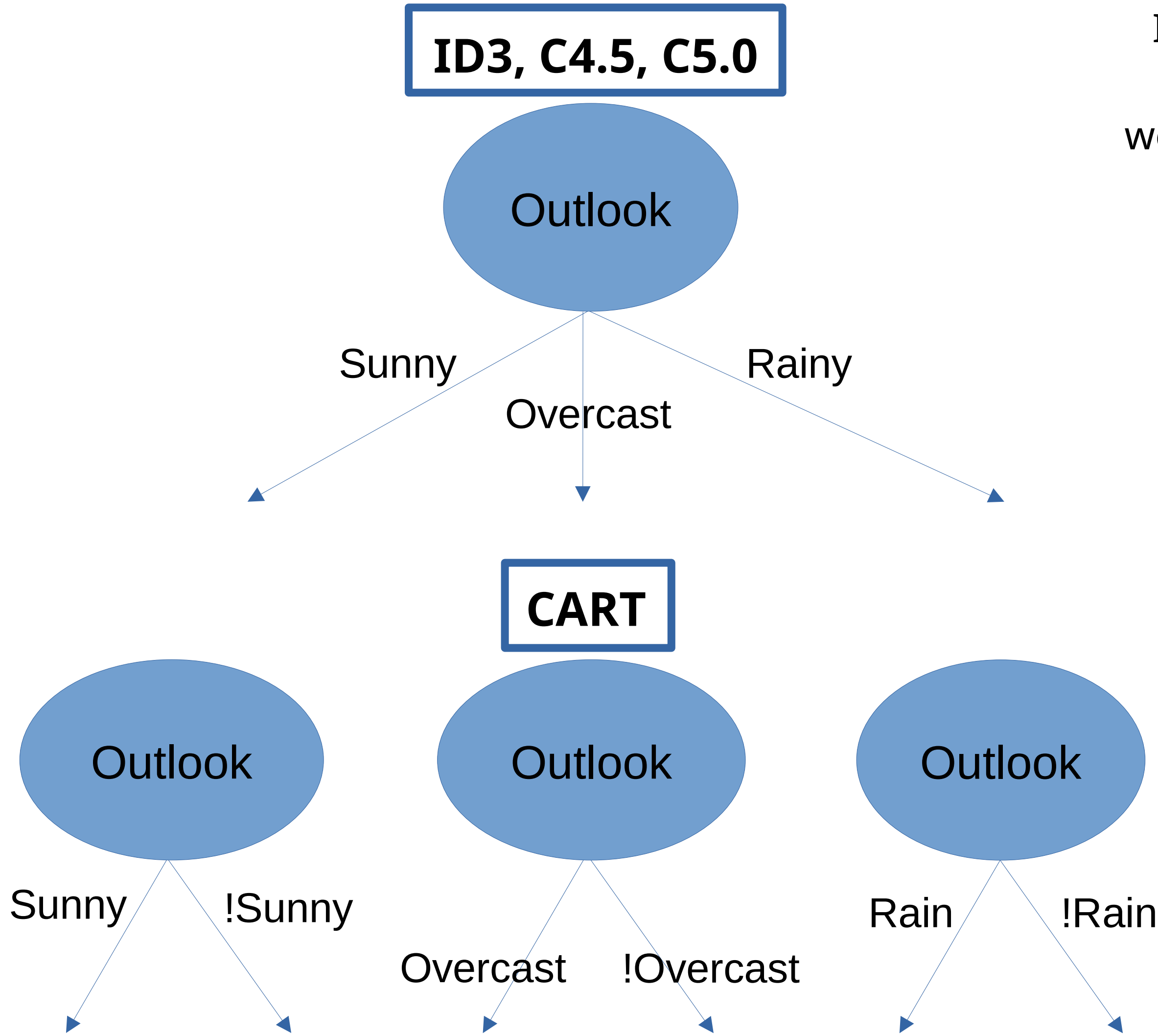
```
## continuous attributes are splitted into categories based on thresholds
t = C5.0(formula = Species ~ ., data = iris)
plot(t)
summary(t)

## there are only two relevant predictors
t = C5.0(formula = Species ~ Petal.Length + Petal.Width, data = iris)
plot(t)
summary(t)
```




```
## the predictors' space is partitioned according to the thresholds determined by the tree
with(iris, plot(Petal.Length, Petal.Width, col = Species, xlab = "Petal Length", ylab = "Petal Width"))
legend("topright", levels(iris$Species), col = 1:length(levels(iris$Species)), pch = 1)
```

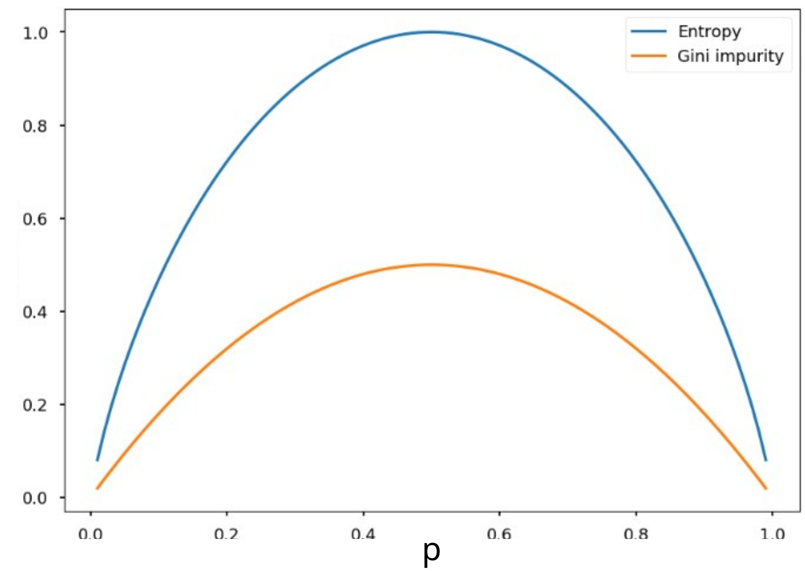


CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

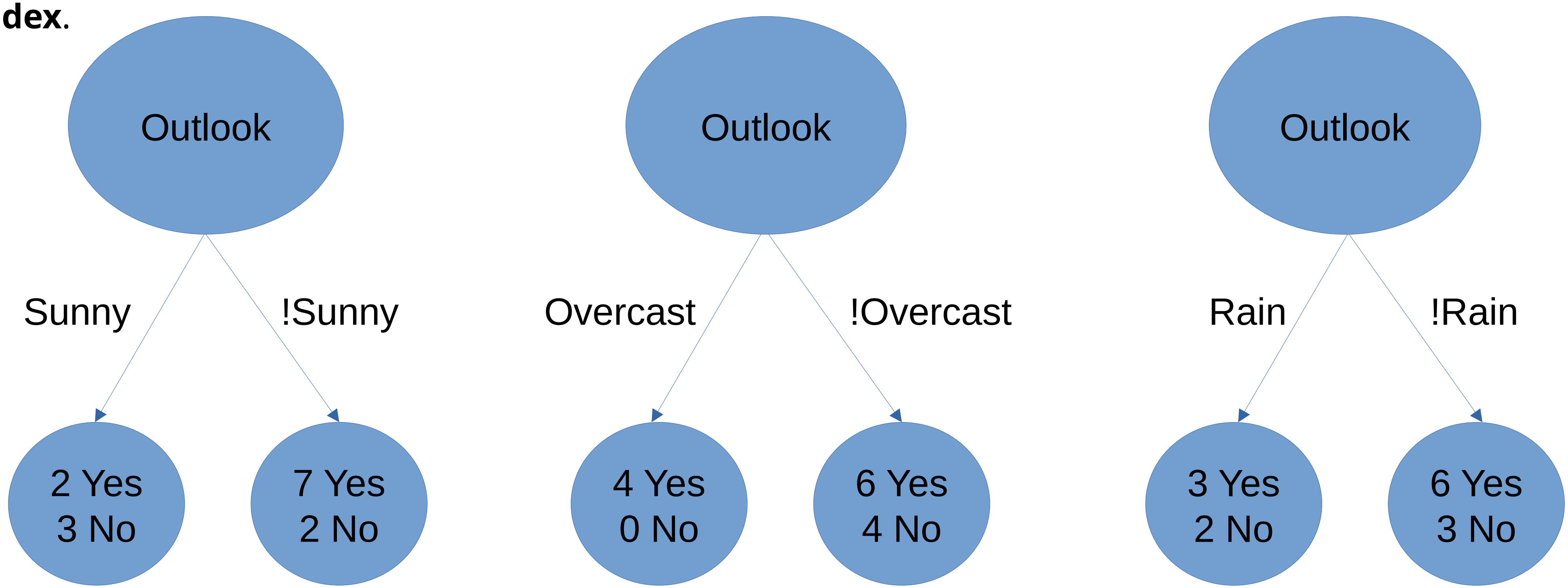


It is first independently calculated for each of the two child nodes. Then, an average value, weighted by the number of data that fall in each child node, is obtained for the parent node

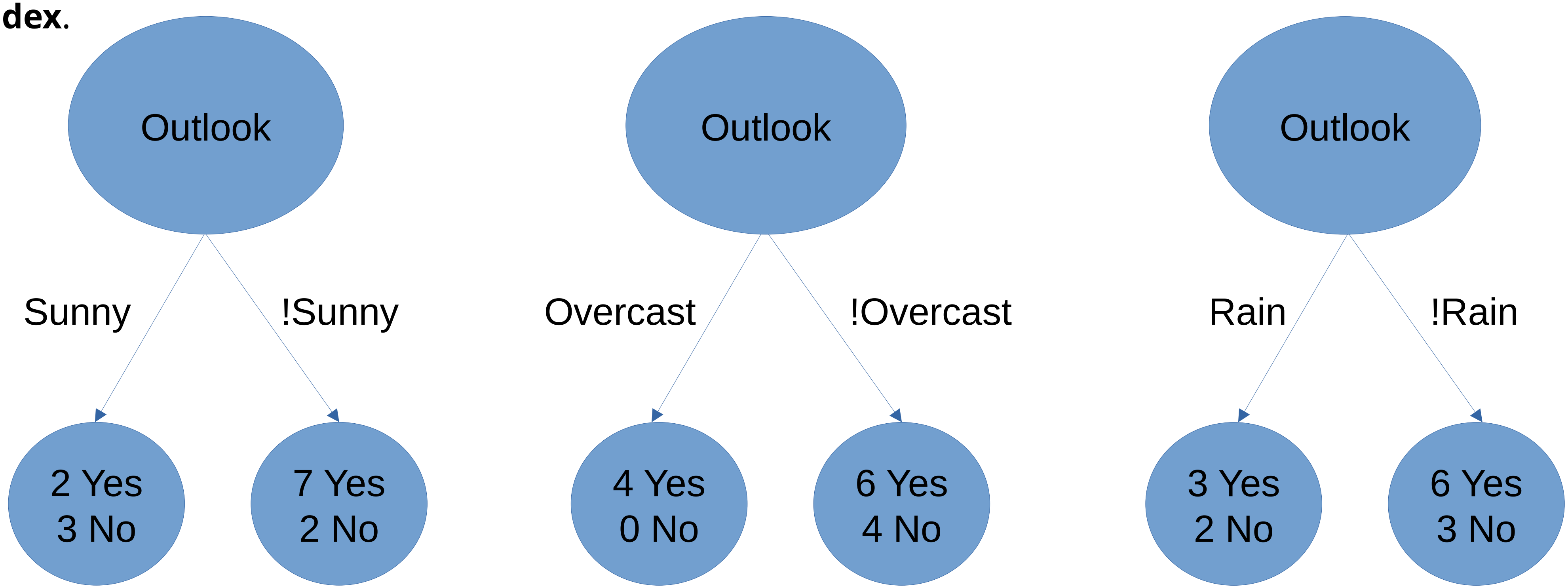
$$GINI = 1 - \sum_{i=1}^{n} p_i^2$$



CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



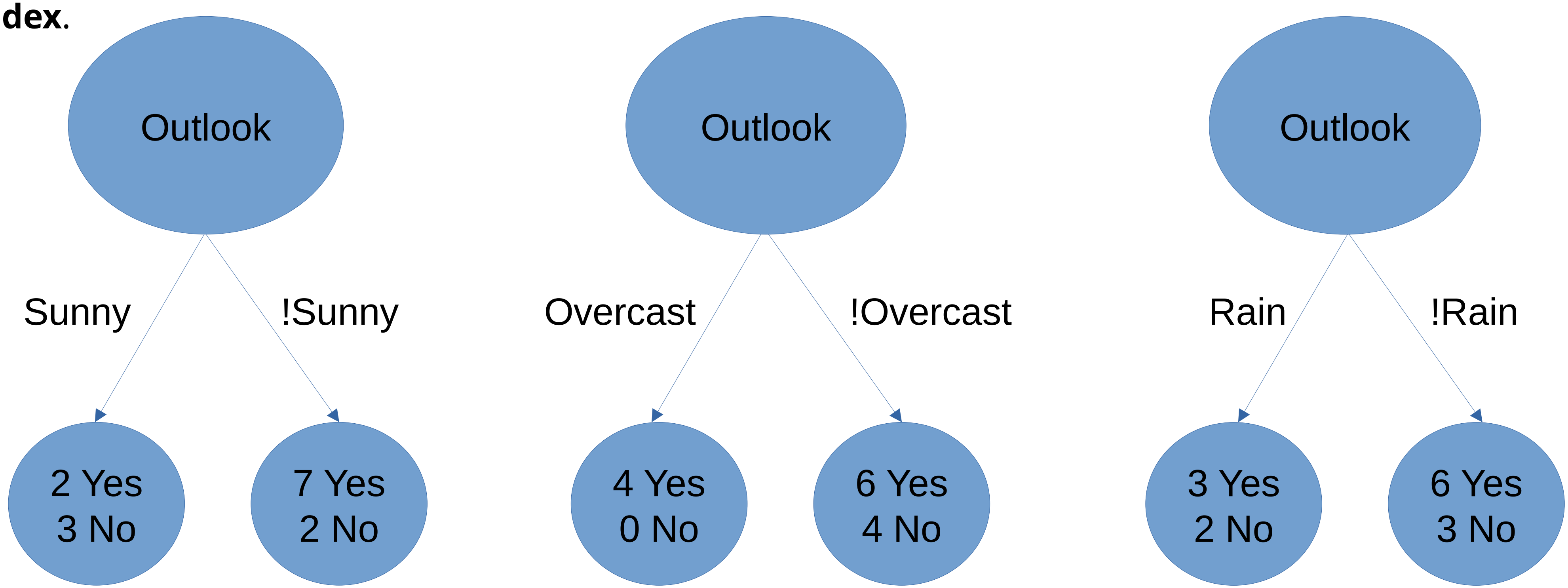
$$G_{Sunny} = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.480$$

$$G_{!Sunny} = 1 - \left(\frac{7}{9}\right)^2 - \left(\frac{2}{9}\right)^2 = 0.346$$

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

CART

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



$$G_{Sunny} = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.480$$

$$G_{!Sunny} = 1 - \left(\frac{7}{9}\right)^2 - \left(\frac{2}{9}\right)^2 = 0.346$$

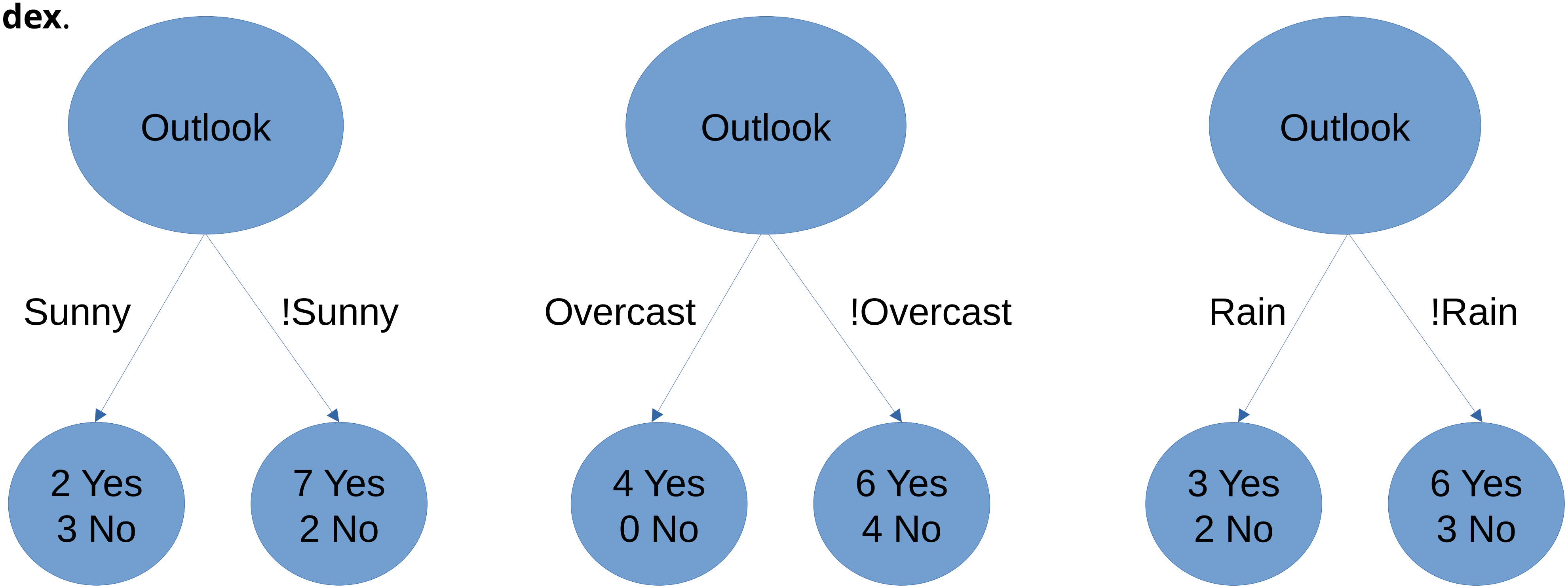
$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$G_{Overcast} = 1 - \left(\frac{4}{4}\right)^2 - \left(\frac{0}{4}\right)^2 = 0$$

$$G_{!Overcast} = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.480$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



$$G_{Sunny} = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2 = 0.480$$

$$G_{!Sunny} = 1 - \left(\frac{7}{9}\right)^2 - \left(\frac{2}{9}\right)^2 = 0.346$$

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$G_{Overcast} = 1 - \left(\frac{4}{4}\right)^2 - \left(\frac{0}{4}\right)^2 = 0$$

$$G_{!Overcast} = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.480$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

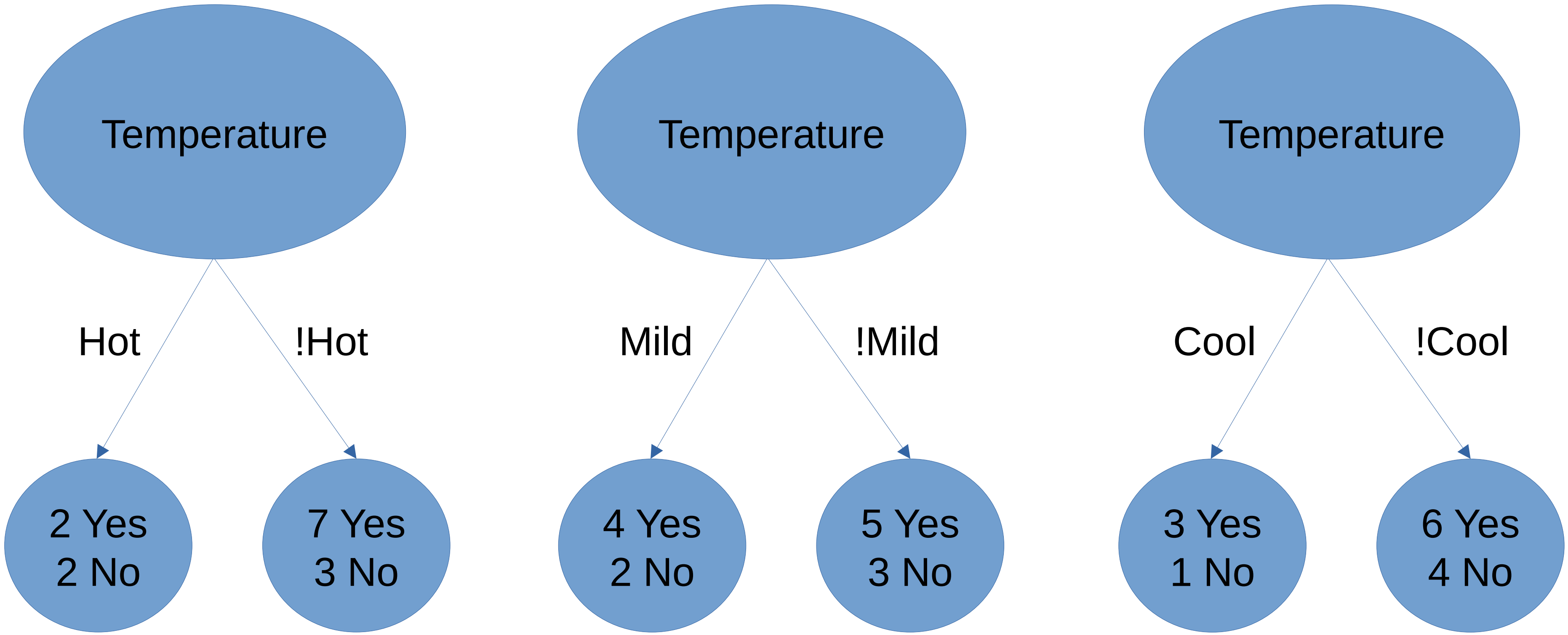
$$G_{Rain} = 1 - \left(\frac{3}{5}\right)^2 - \left(\frac{2}{5}\right)^2 = 0.480$$

$$G_{!Rain} = 1 - \left(\frac{6}{9}\right)^2 - \left(\frac{3}{9}\right)^2 = 0.444$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

CART

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



$$G_{Hot} = 1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2 = 0.500$$

$$G_{!Hot} = 1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2 = 0.420$$

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$G_{Mild} = 1 - \left(\frac{4}{6}\right)^2 - \left(\frac{2}{6}\right)^2 = 0.444$$

$$G_{!Mild} = 1 - \left(\frac{5}{8}\right)^2 - \left(\frac{3}{8}\right)^2 = 0.469$$

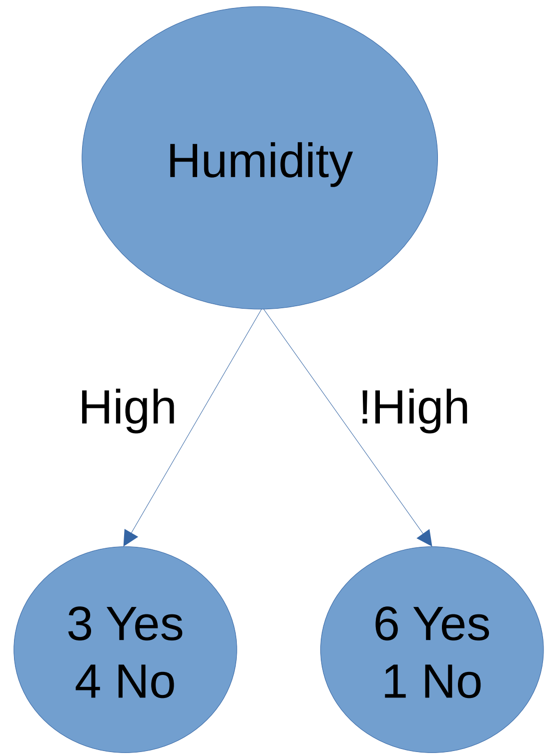
$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

$$G_{Cool} = 1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2 = 0.375$$

$$G_{!Cool} = 1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2 = 0.480$$

$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

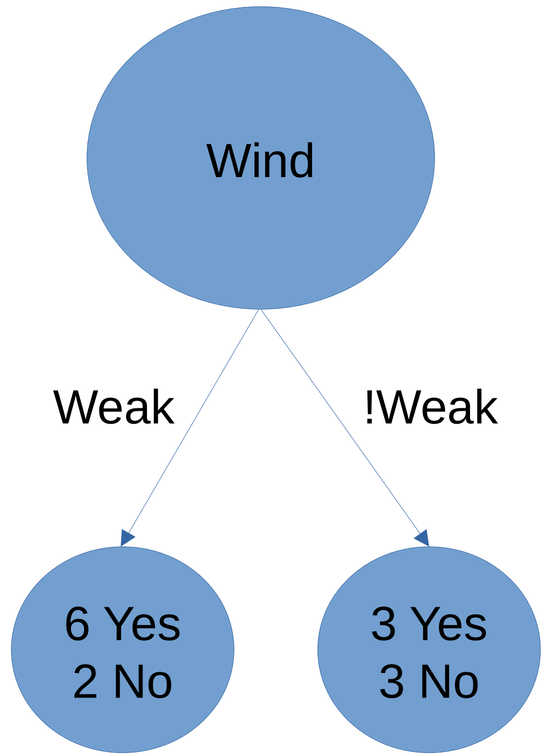


$$G_{High} = 1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2 = 0.490$$

$$G_{!High} = 1 - \left(\frac{6}{7}\right)^2 - \left(\frac{1}{7}\right)^2 = 0.245$$

$$\frac{7}{14} G_{High} + \frac{7}{14} G_{!High} = 0.367$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.



$$G_{Weak} = 1 - \left(\frac{6}{8}\right)^2 - \left(\frac{2}{8}\right)^2 = 0.375$$

$$G_{!Weak} = 1 - \left(\frac{3}{6}\right)^2 - \left(\frac{3}{6}\right)^2 = 0.500$$

$$\frac{8}{14} G_{Weak} + \frac{6}{14} G_{!Weak} = 0.429$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

Outlook

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

Temperature

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

Humidity

$$\frac{7}{14}G_{High} + \frac{7}{14}G_{!High} = 0.367$$

Wind

$$\frac{8}{14}G_{Weak} + \frac{6}{14}G_{!Weak} = 0.429$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

Outlook

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

Temperature

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

Humidity

$$\frac{7}{14}G_{High} + \frac{7}{14}G_{!High} = 0.367$$

Wind

$$\frac{8}{14}G_{Weak} + \frac{6}{14}G_{!Weak} = 0.429$$

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

Outlook

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

Temperature

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

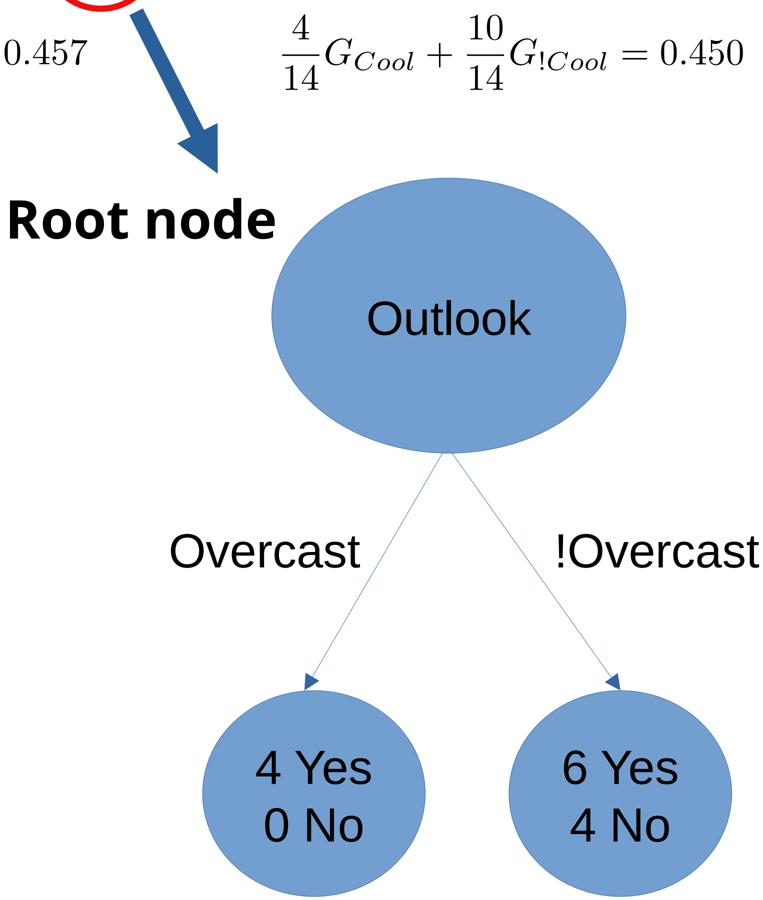
$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

Humidity

$$\frac{7}{14}G_{High} + \frac{7}{14}G_{!High} = 0.367$$

Wind

$$\frac{8}{14}G_{Weak} + \frac{6}{14}G_{!Weak} = 0.429$$



CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

Outlook

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

Temperature

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

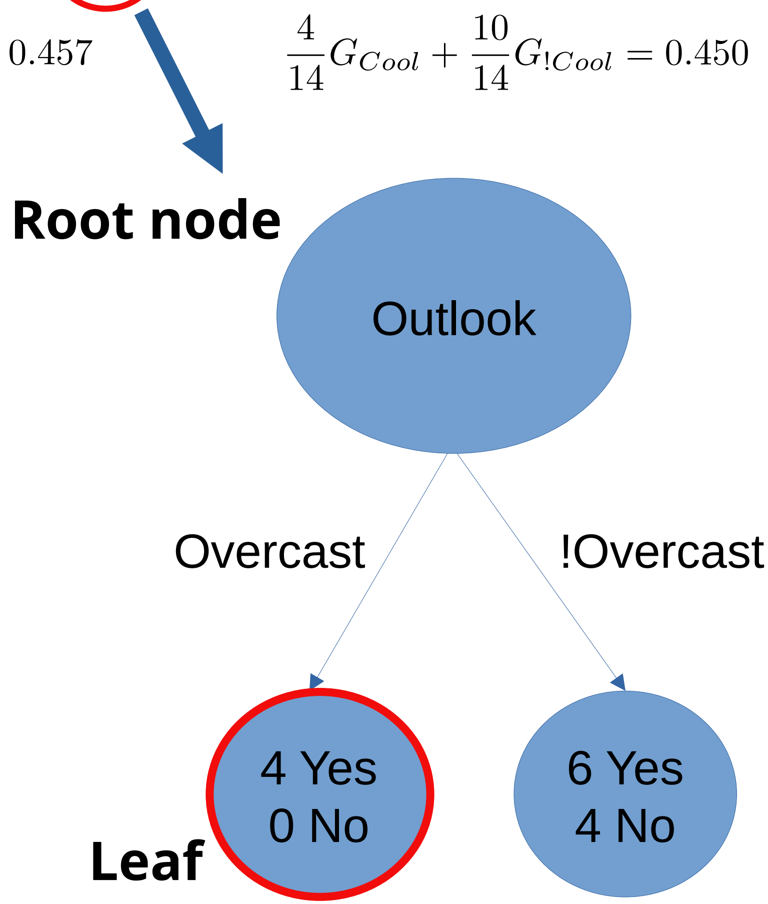
$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

Humidity

$$\frac{7}{14}G_{High} + \frac{7}{14}G_{!High} = 0.367$$

Wind

$$\frac{8}{14}G_{Weak} + \frac{6}{14}G_{!Weak} = 0.429$$



CART

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

Outlook

$$\frac{5}{14}G_{Sunny} + \frac{9}{14}G_{!Sunny} = 0.394$$

$$\frac{4}{14}G_{Overcast} + \frac{10}{14}G_{!Overcast} = 0.343$$

$$\frac{5}{14}G_{Rain} + \frac{9}{14}G_{!Rain} = 0.457$$

Temperature

$$\frac{4}{14}G_{Hot} + \frac{10}{14}G_{!Hot} = 0.443$$

$$\frac{6}{14}G_{Mild} + \frac{8}{14}G_{!Mild} = 0.456$$

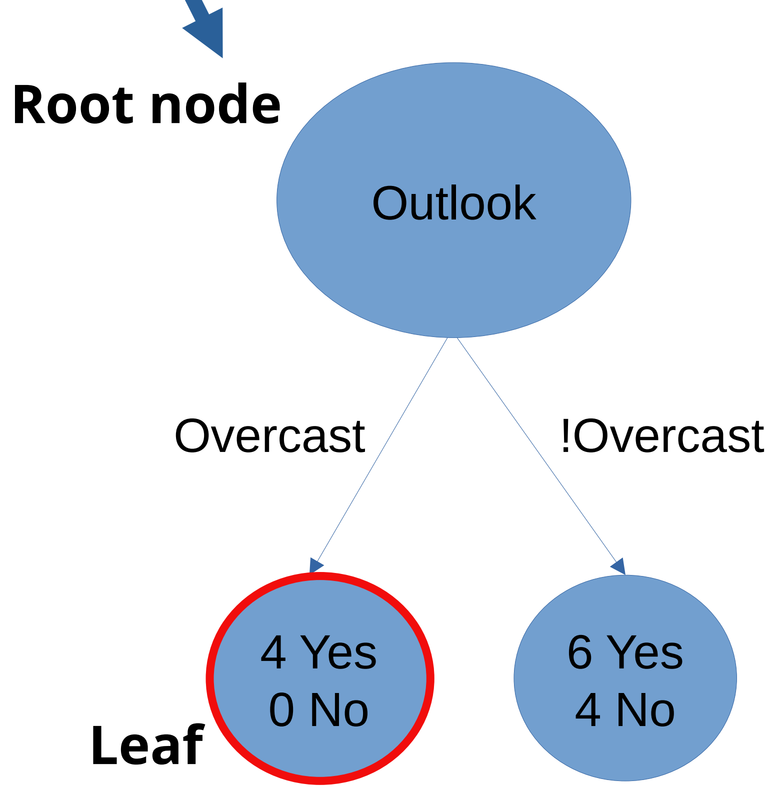
$$\frac{4}{14}G_{Cool} + \frac{10}{14}G_{!Cool} = 0.450$$

Humidity

$$\frac{7}{14}G_{High} + \frac{7}{14}G_{!High} = 0.367$$

Wind

$$\frac{8}{14}G_{Weak} + \frac{6}{14}G_{!Weak} = 0.429$$



... continue to split ...

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

→

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5

70.5

75.0

79.5

87.0

97.5

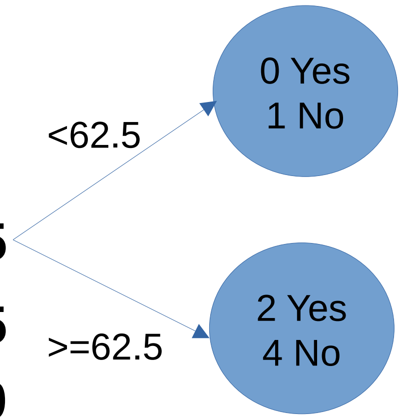
CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5
70.5
75.0
79.5
87.0
97.5



$$G_{<62.5} = 1 - \left(\frac{0}{1}\right)^2 - \left(\frac{1}{1}\right)^2$$

$$G_{\geq 62.5} = 1 - \left(\frac{2}{6}\right)^2 - \left(\frac{4}{6}\right)^2$$

$$\frac{1}{7}G_{<62.5} + \frac{6}{7}G_{\geq 62.5} = 0.381$$

$$G_{62.5} = 0.381$$

CART

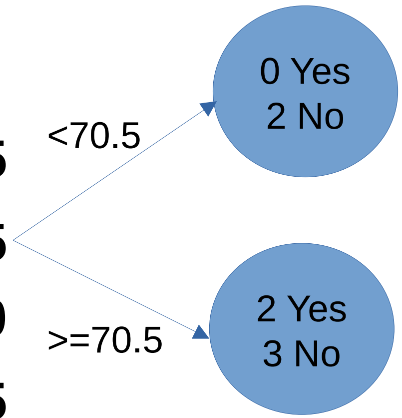
CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5
70.5
75.0
79.5
87.0
97.5



$$G_{<70.5} = 1 - \left(\frac{0}{2}\right)^2 - \left(\frac{2}{2}\right)^2$$

$$G_{\geq 70.5} = 1 - \left(\frac{2}{5}\right)^2 - \left(\frac{3}{5}\right)^2$$

$$\frac{2}{7} G_{<70.5} + \frac{5}{7} G_{\geq 70.5} = 0.343$$

$$G_{62.5} = 0.381 \quad G_{70.5} = 0.343$$

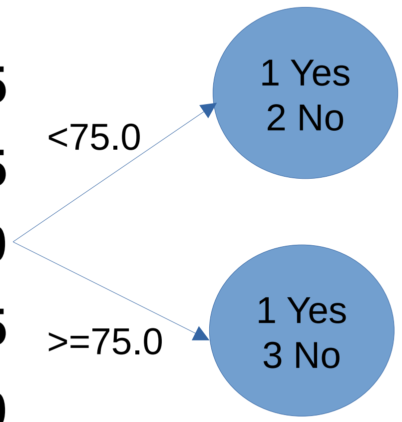
CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5
70.5
75.0
79.5
87.0
97.5



$$G_{<75.0} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2$$

$$G_{\geq 75.0} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2$$

$$\frac{3}{7}G_{<75.0} + \frac{4}{7}G_{\geq 75.0} = 0.405$$

$$G_{62.5} = 0.381 \quad G_{70.5} = 0.343 \quad G_{75.0} = 0.405$$

CART

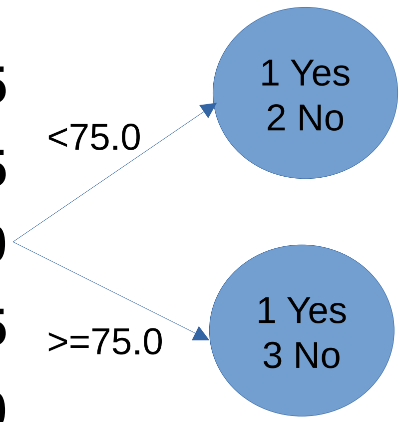
CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5
70.5
75.0
79.5
87.0
97.5



$$G_{<75.0} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2$$

$$G_{\geq 75.0} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2$$

$$\frac{3}{7} G_{<75.0} + \frac{4}{7} G_{\geq 75.0} = 0.405$$

... continue ...

$$G_{62.5} = 0.381 \quad G_{70.5} = 0.343 \quad G_{75.0} = 0.405$$

CART

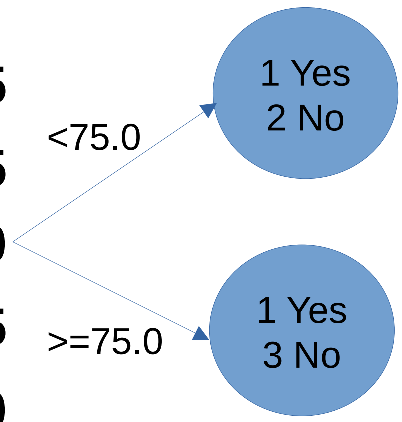
CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5
70.5
75.0
79.5
87.0
97.5



$$G_{<75.0} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2$$

$$G_{\geq 75.0} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2$$

$$\frac{3}{7}G_{<75.0} + \frac{4}{7}G_{\geq 75.0} = 0.405$$

... continue ...

$$G_{62.5} = 0.381 \;\; G_{70.5} = 0.343 \;\; G_{75.0} = 0.405 \;\; G_{79.5} = 0.405 \;\; G_{87.0} = 0.371 \;\; G_{97.5} = 0.238$$

CART

CART (**C**lassification **A**nd **R**egression **T**rees) split **binary** attributes. The attribute's value upon which the split is performed is chosen based on the **Gini index**. At each node, **the goal is to minimize the Gini index**.

In the case of **continuous attributes**, it is needed to find the threshold value that best separate the target variable.

Weight	Heart Attack
57	No
68	No
103	Yes
92	No
73	Yes
77	No
82	No

Weight	Heart Attack
57	No
68	No
73	Yes
77	No
82	No
92	No
103	Yes

62.5

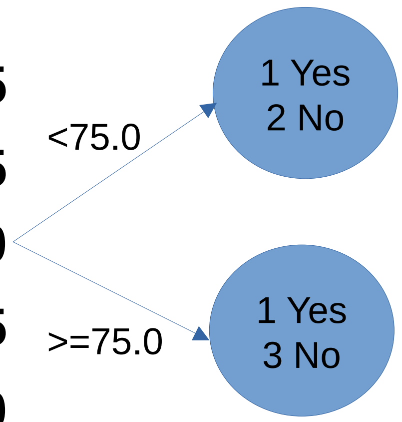
70.5

75.0

79.5

87.0

97.5



$$G_{<75.0} = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2$$

$$G_{\geq 75.0} = 1 - \left(\frac{1}{4}\right)^2 - \left(\frac{3}{4}\right)^2$$

$$\frac{3}{7} G_{<75.0} + \frac{4}{7} G_{\geq 75.0} = 0.405$$

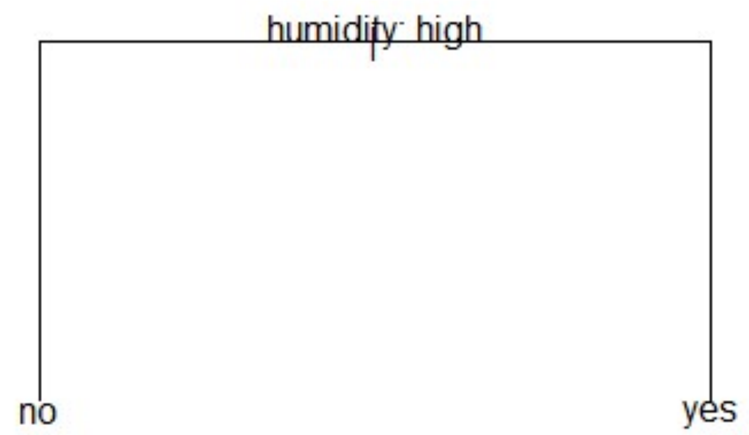
... continue ...

$$G_{62.5} = 0.381 \quad G_{70.5} = 0.343 \quad G_{75.0} = 0.405 \quad G_{79.5} = 0.405 \quad G_{87.0} = 0.371 \quad G_{97.5} = 0.238$$

Threshold chosen:
97.5

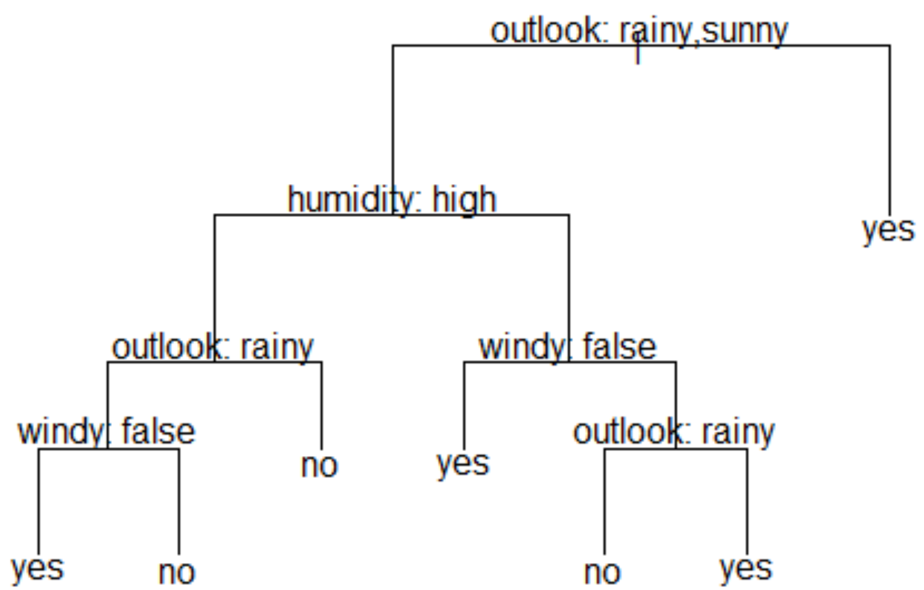
*We have already used the C5.0 package to grow classification trees based on GR. For CART (based on the Gini index), let's now use **tree** and **rpart** for the **playTennis** dataset (categorical attributes). Let's start with **tree**.*

```
## tree package
library(tree)
## default parameters
t = tree(formula = play ~ ., data = tennis)
plot(t)
text(t, pretty = F)
```



Documentation must be carefully read!!

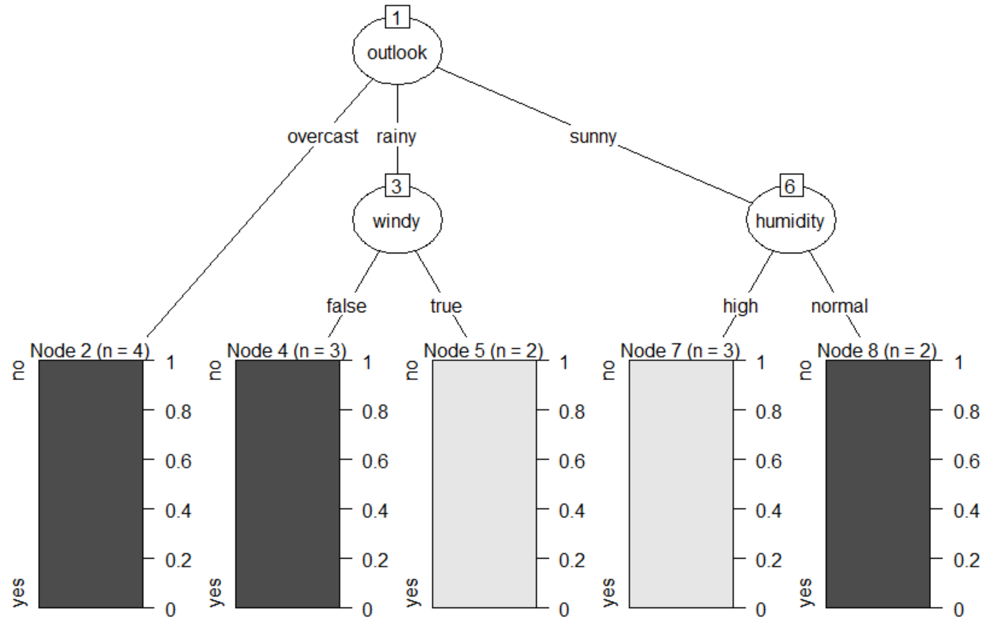
```
## user-defined parameters
t = tree(formula = play ~ ., data = tennis,
         minsize = 2)
plot(t)
text(t, pretty = F)
```

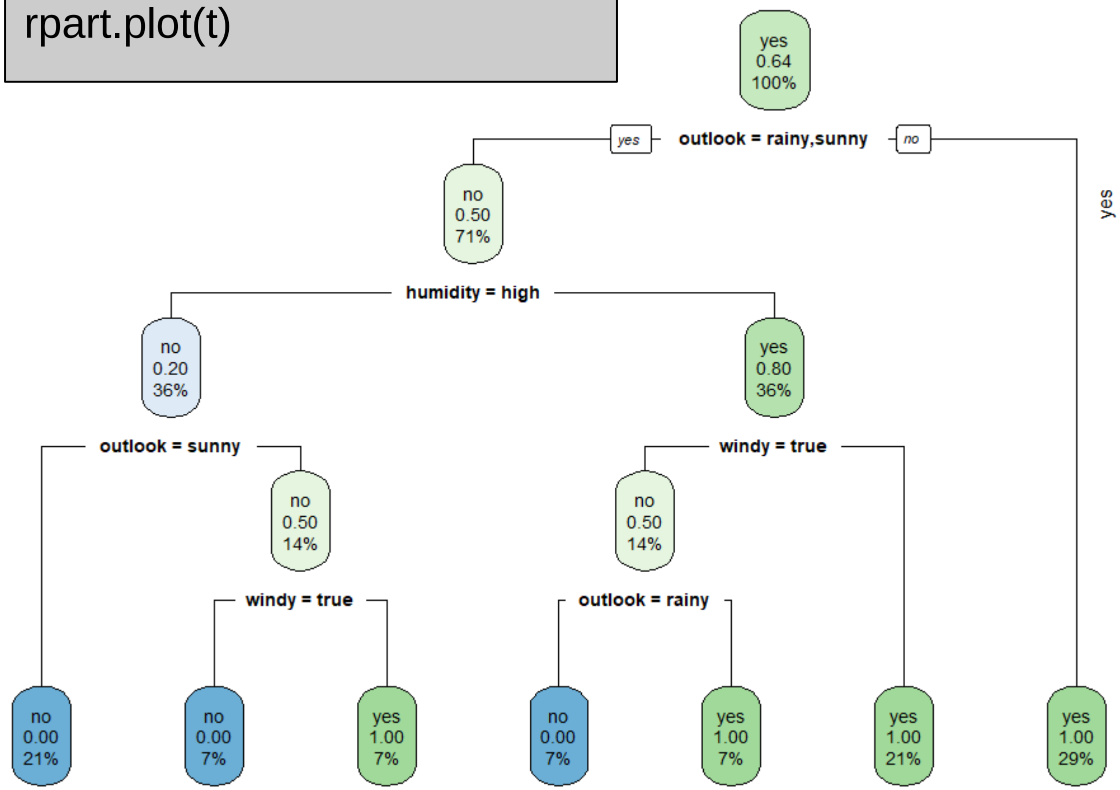


```
## rpart/rpart.plot packages
library(rpart)
## default parameters
t = rpart(formula = play ~ .,
data = tennis)
library(rpart.plot)
rpart.plot(t)
```

```
## user-defined parameters
t = rpart(formula = play ~ .,
data = tennis, minsplit = 2)
rpart.plot(t)
```

Now, let's move to ***rpart*** *(still for* ***playTennis****)*



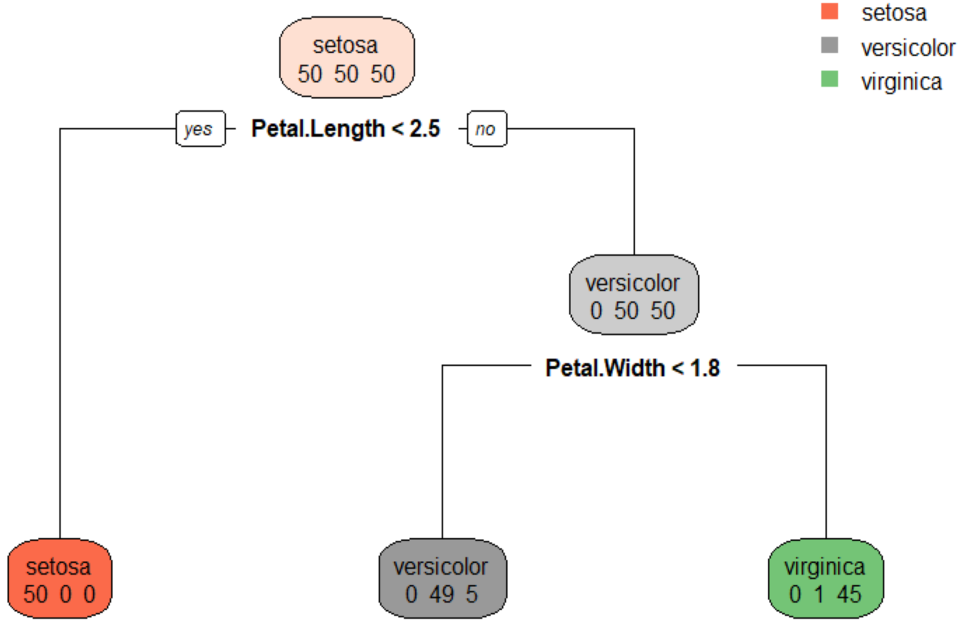


Compare with the result obtained with the *C5.0* package: **which are the differences?**

Documentation must be carefully read!!

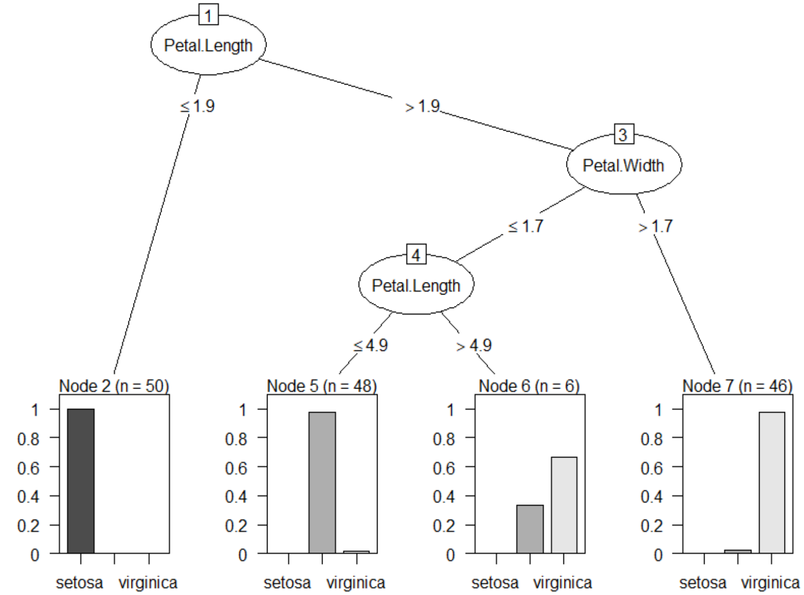
*Let's change to **iris** (keep on using **rpart**)*

```
## default parameters
t = rpart(formula = Species ~ ., data = iris)
rpart.plot(t, extra = 1)
```

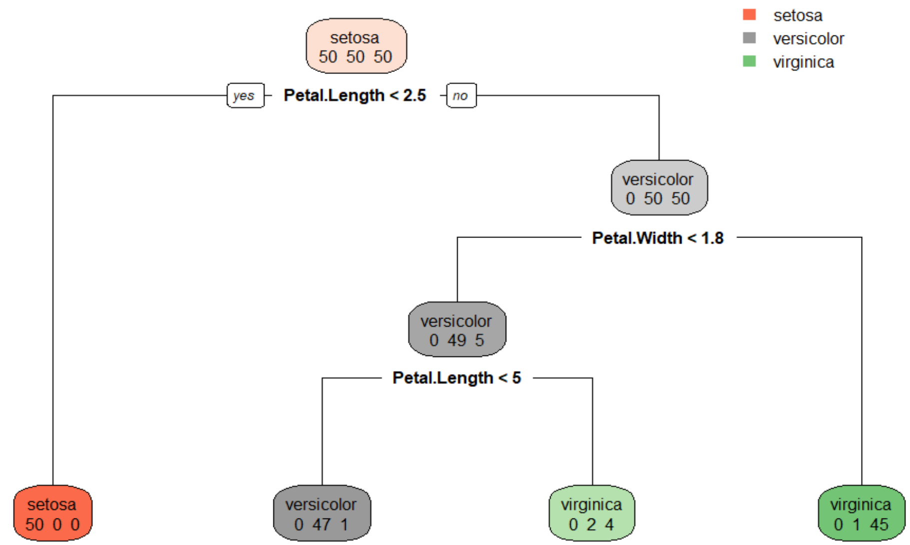


Documentation must be carefully read!!

```
## user-defined parameters
t = rpart(formula = Species ~ ., data = iris,
minsplit = 2)
rpart.plot(t, extra = 1)
```

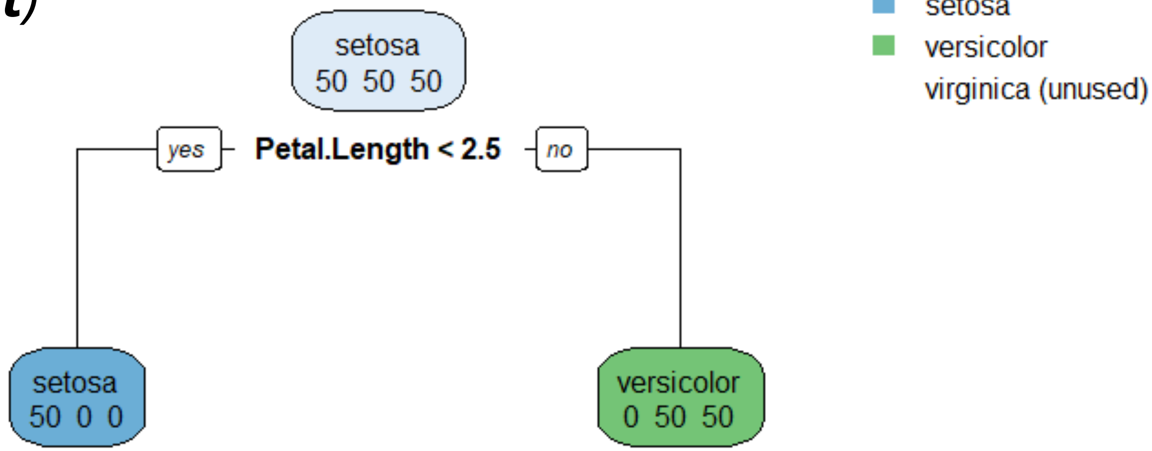


Compare with the result obtained with the *C5.0* package: **which are the differences?**

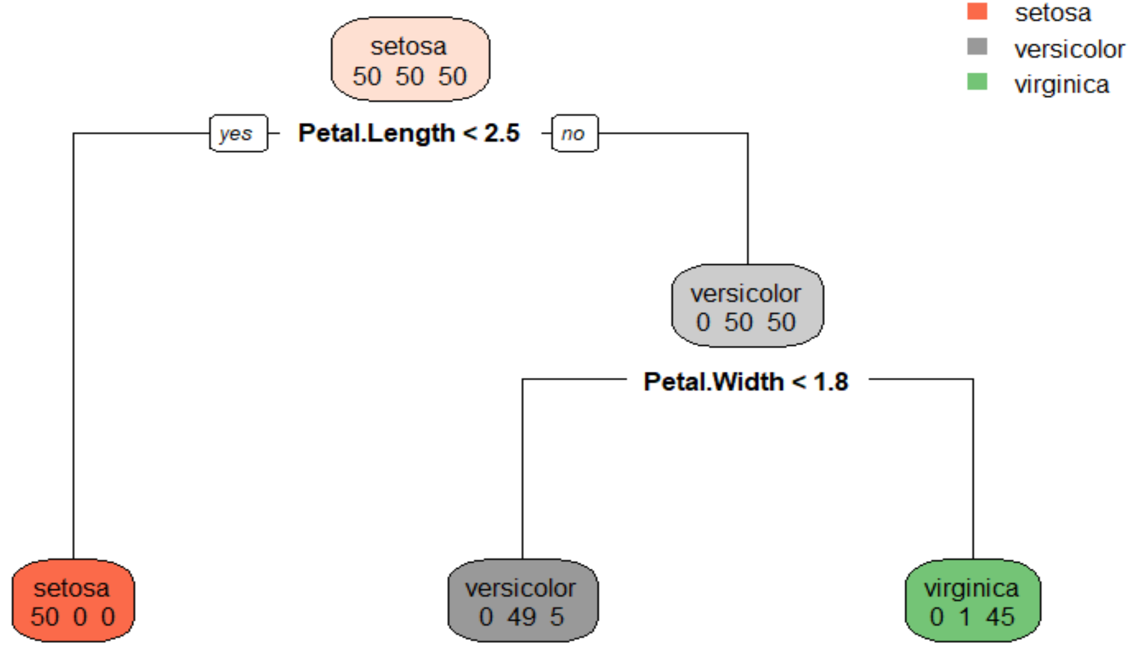


Let's change to ***iris*** *(keep on using* ***rpart****)*

```
## user-defined parameters
t = rpart(formula = Species ~ ., data = iris,
maxdepth = 1)
rpart.plot(t, extra = 1)
```




```
## user-defined parameters
t = rpart(formula = Species ~ ., data = iris,
maxdepth = 2)
rpart.plot(t, extra = 1)
```

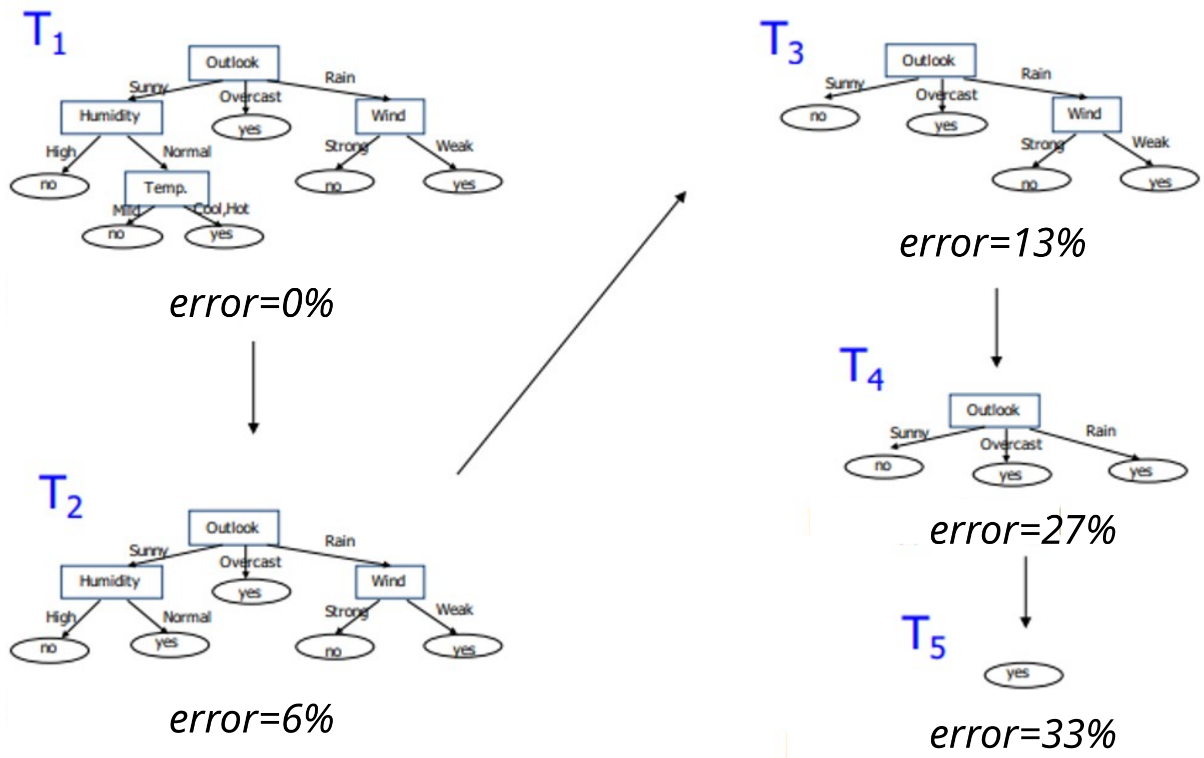


Pruning

Most likely, a large tree (i.e. with many terminal nodes) will be overfitted to the training data, leading to poor performance in the test set. Generally, we can improve this behavior by means of **pruning**.

<u>Pre-pruning</u>: Stop growing the tree before it reaches the point at which it perfectly classifies the learning sample. To do so, stop splitting a node if the **number of elements within is too small** or the **impurity is low enough**. Note that this approach can indeed lead to small trees but **can also miss relevant splits**.

<u>Post-pruning</u>: Allow the tree to fully grow (thus, incurring in overfitting) and then, **remove the less useful nodes**. To do so, compute a sequence of trees {T1, T2, ...} where T1 is the **complete** (i.e. **fully grown**) tree. T2 is obtained by removing from T1 the node that less increases the classification error, and so on. In general, this is **preferred option**.



The question is: **where to stop?** In practice, cross-validation is used to find the optimal size (i.e. number of leaves and/or depth levels) of the tree.

*Let's find the **optimum number of depth levels** that best classify the vehicles in the **cars** dataset as cheap or expensive. Consider a car to be cheap (expensive) when Price is equal or above (below) 22k $. Select the 75% of the data for cross-validation and the 25% for test. Apply a **10-fold cross-validation** framework using **caret**. Which is the accuracy of the optimum tree for the test set?*

```
## exploring the dataset
library(caret)
data("cars"); summary(cars)

## convert continuous variable "Price" to categorical
cars$Price = as.factor(ifelse(cars$Price >= 22000, "E", "C"))

## 75% of the dataset for cross-validation and the other 25% for test
indcv = createDataPartition(y = ***, p = ***, list = FALSE)
dataset.cv = ***
dataset.test = ***

## 10-fold cross-validation
trctrl = trainControl(method = ***, number = ***)
## caret automatically tries different values of the method's parameter (4 in this case, internally selected)
t = train(Price ~ ., data = ***,
          method = ***,
          trControl = trctrl,
          tuneLength = 4)
plot(t)

## prediction
pred = predict(***, newdata = ***)
## validation
sum(diag(table(pred, dataset.test$Price))) / dim(dataset.test)[1]
0.9353234
```