Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación


# IIC2613 – Inteligencia Artificial

## Árboles de decisión

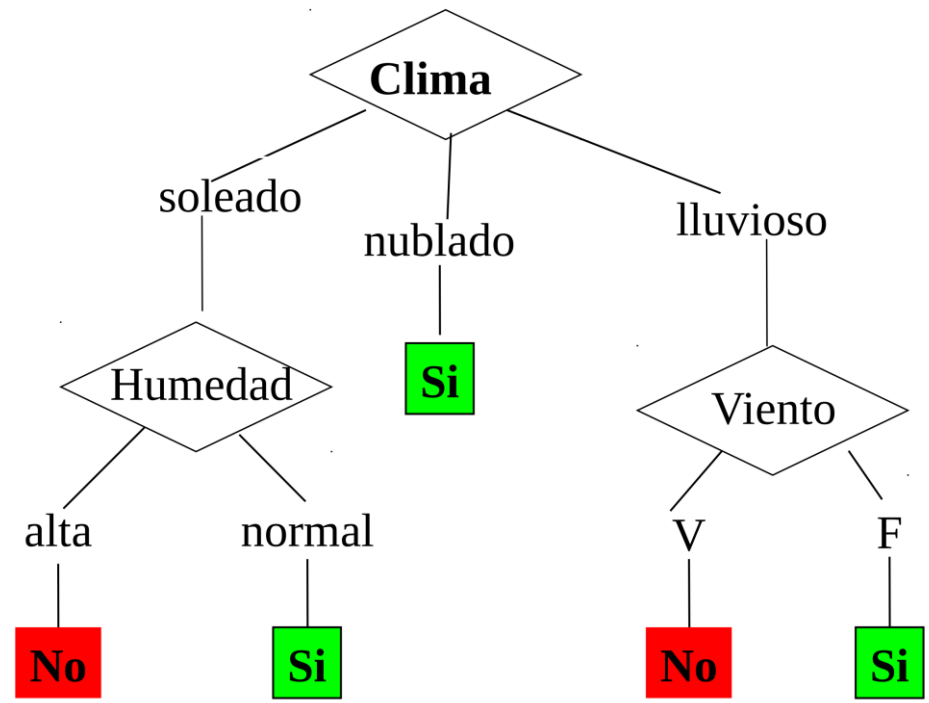
**Profesor:** Hans Löbel

¿Cómo solucionamos el siguiente problema de clasificación?

| Clima | Temperatura | Humedad | Viento | Jugar? |
|---|---|---|---|---|
| soleado | alta | alta | F | No |
| soleado | alta | alta | V | No |
| nublado | alta | alta | F | Si |
| lluvioso | Agradable | alta | F | Si |
| lluvioso | frio | normal | F | Si |
| lluvioso | frio | normal | V | No |
| nublado | frio | normal | V | Si |
| soleado | Agradable | alta | F | No |
| soleado | frio | normal | F | Si |
| lluvioso | Agradable | normal | F | Si |
| soleado | Agradable | normal | V | Si |
| nublado | Agradable | alta | V | Si |
| nublado | alta | normal | F | Si |
| lluvioso | Agradable | alta | V | No |

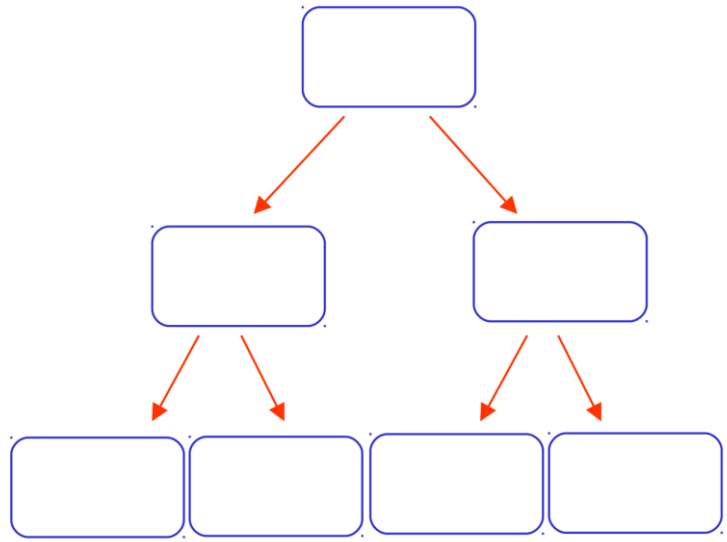
# ¿Cómo solucionamos el siguiente problema de clasificación?

| Clima | Temperatura | Humedad | Viento | Jugar? |
|---|---|---|---|---|
| soleado | alta | alta | F | No |
| soleado | alta | alta | V | No |
| nublado | alta | alta | F | Si |
| lluvioso | Agradable | alta | F | Si |
| lluvioso | frio | normal | F | Si |
| lluvioso | frio | normal | V | No |
| nublado | frio | normal | V | Si |
| soleado | Agradable | alta | F | No |
| soleado | frio | normal | F | Si |
| lluvioso | Agradable | normal | F | Si |
| soleado | Agradable | normal | V | Si |
| nublado | Agradable | alta | V | Si |
| nublado | alta | normal | F | Si |
| lluvioso | Agradable | alta | V | No |



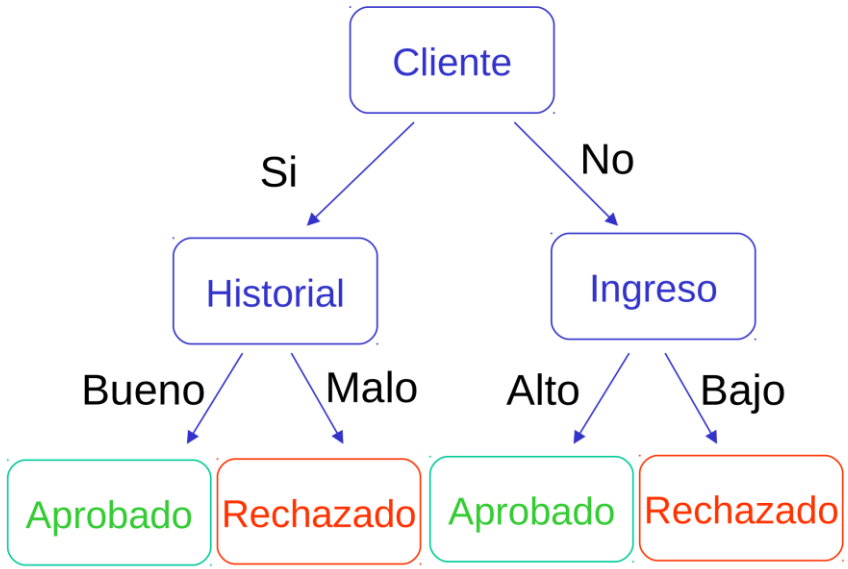
# Árboles de decisión pueden solucionar el caso anterior

- Técnica de aprendizaje supervisado.
- Pueden realizar clasificación y regresión.
- Pueden usarse sobre distintos tipos de variables (binaria, categórica, numérica).

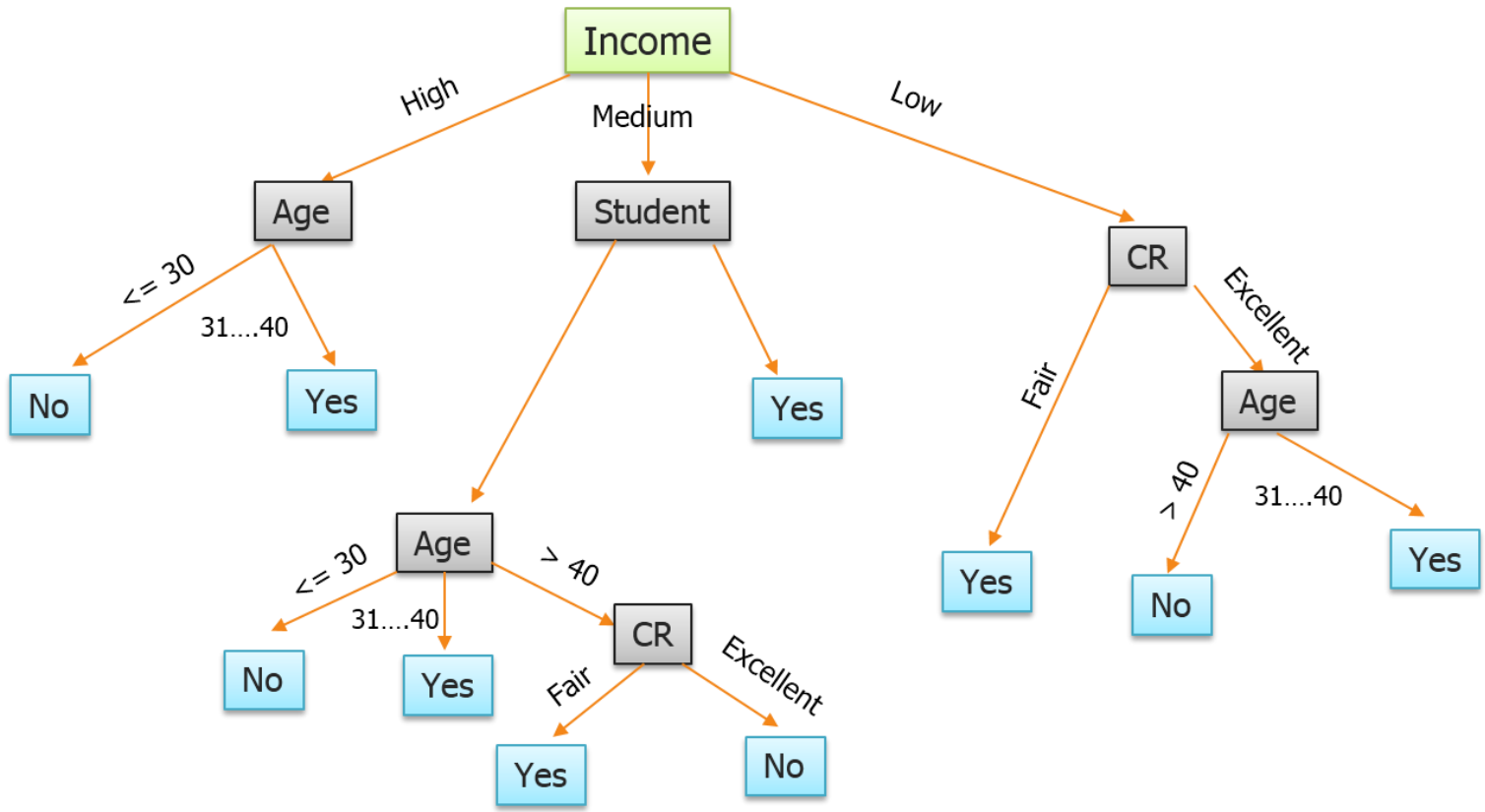


# Árboles de decisión son ampliamente utilizados en la práctica

- Cada nodo interno representa un atributo y cada nodo hoja representa una categoría.
- En cada nodo interno, se realiza un test en base a los valores del atributo.
- Aristas representan el resultado del test.
- Para clasificar un registro, se debe pasar desde la raíz hasta alguna hoja.

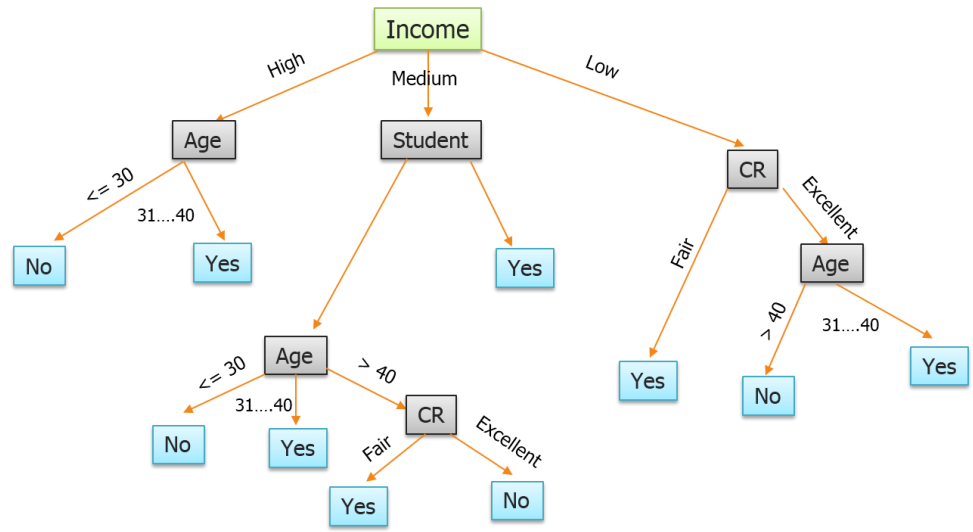


# ¿Cómo construimos un árbol de decisión?



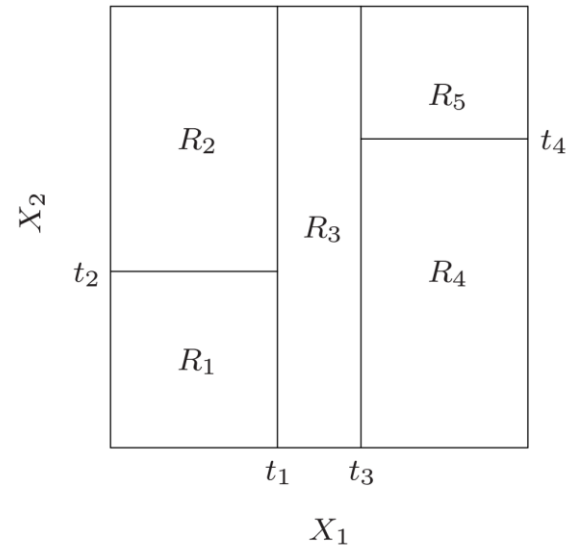
# Podemos destilar esto en dos preguntas más específicas

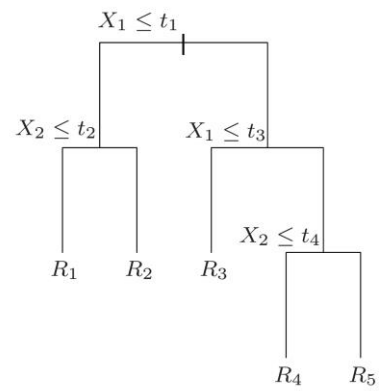
1. ¿En qué orden realizo los tests?
2. ¿En que parte del dominio pongo el umbral de decisión? (atributos numéricos)

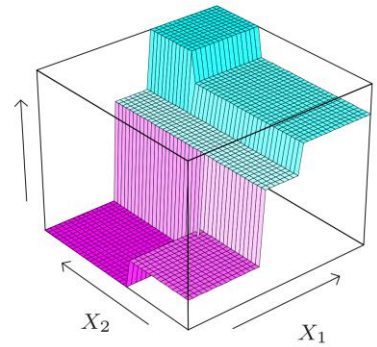


# Todo depende de cómo definamos lo que es mejor

1. ¿Pertenecen todos los registros a la misma clase?
   - Retornar marcando el nodo hoja con la clase respectiva.
2. ¿Tienen todos los registros el mismo valor para todos los atributos que determinan su clase?
   - Retornar marcando nodo hoja con la clase más común.
3. De lo contrario:
   i. Seleccionar el atributo que mejor separa los registros de las distintas clases.
   ii. Usar ese atributo como nodo raíz.
   iii. Dividir el set de entrenamiento de acuerdo a este atributo y para cada rama resultante continuar la construcción del árbol en forma recursiva.

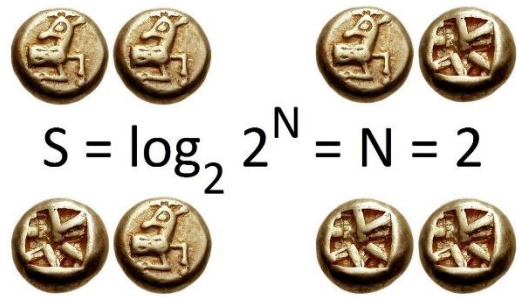


# Todo depende de cómo definamos lo que es mejor

- Si objetivo es clasificar, es razonable que el mejor atributo separe mejor de acuerdo a las clases.
- Cuán homogéneo o impuro es un atributo, en función de las categorías.
- Dos maneras típicas de medir esto son:
  - Gini Index: desigualdad (inequidad) sobre distintas categorías.
  - Information Entropy: bits necesarios para codificar información.

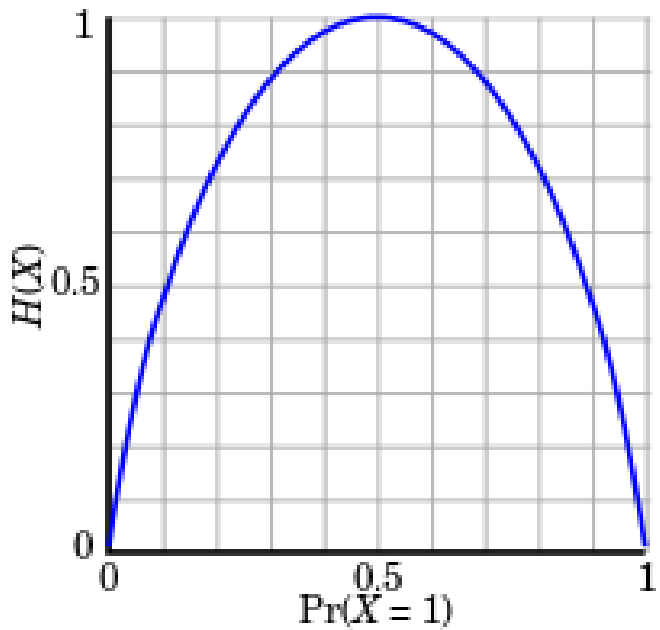


# Entropía permite capturar de manera eficiente cuán homogénea es la distribución

- Intuitivamente, puede verse como un promedio ponderado de probabilidades de ocurrencia:

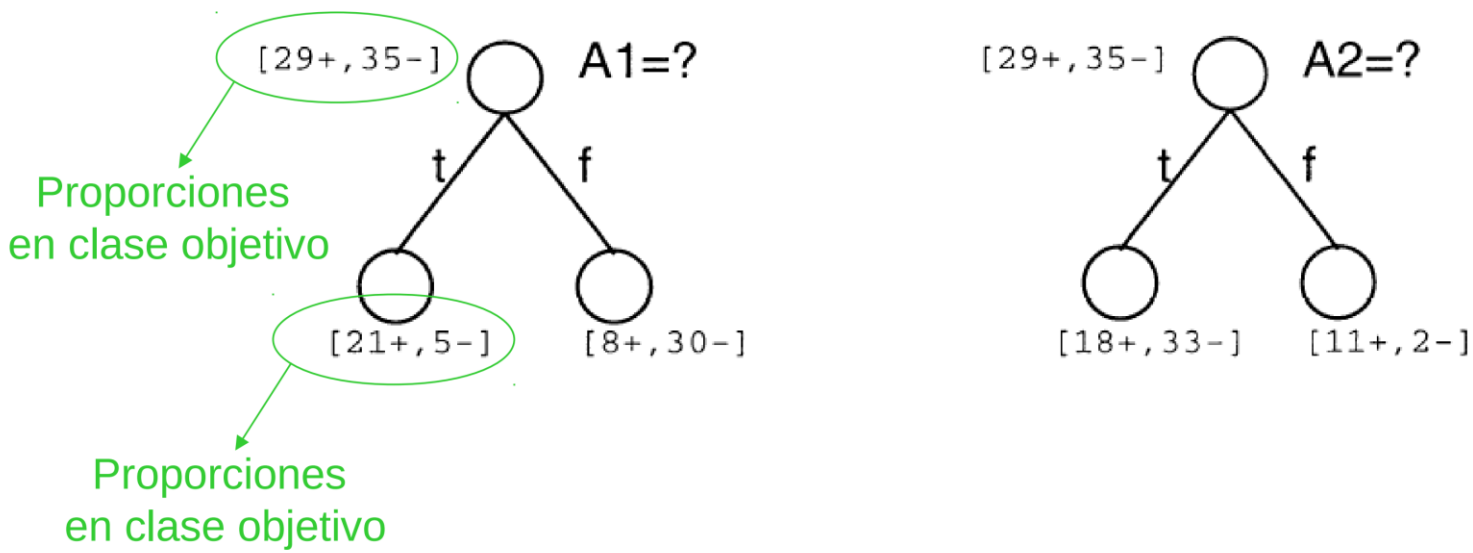
$$H(S) = -\sum_{c_i} p_i \log_2 p_i$$

- Por ejemplo:
  - 4 clases (A,B,C,D): 10 registros clase A, 20 clase B, 30 clase C, 40 clase D.
  - Entropía= -[(.1 log .1) + (.2 log .2) + (.3 log .3) + (.4 log .4)] = 1.85.



Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

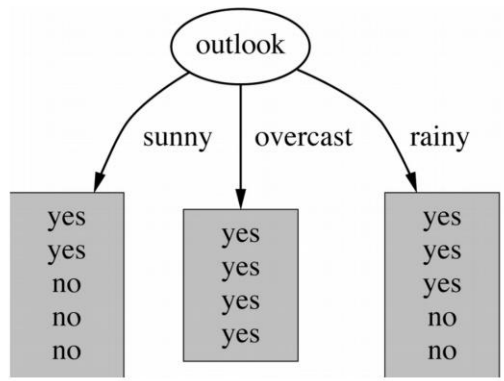


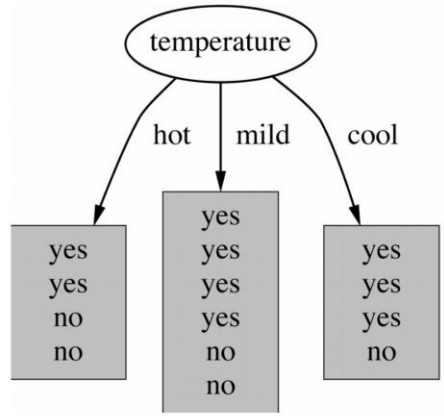
Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)

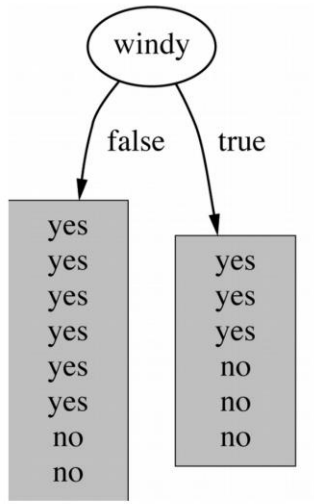
| Day | Outlook | Temperature | Humidity | Wind | PlayTenn |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6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

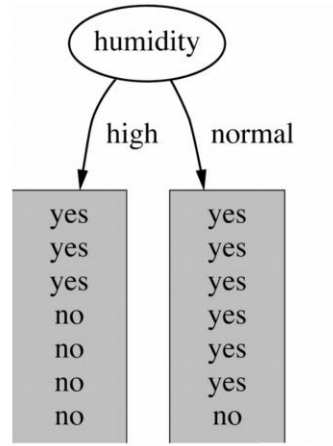
# Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)

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

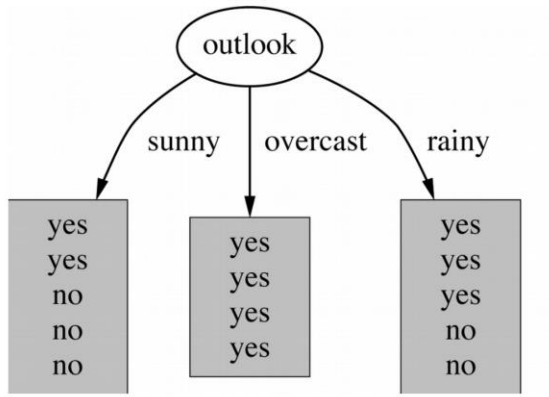


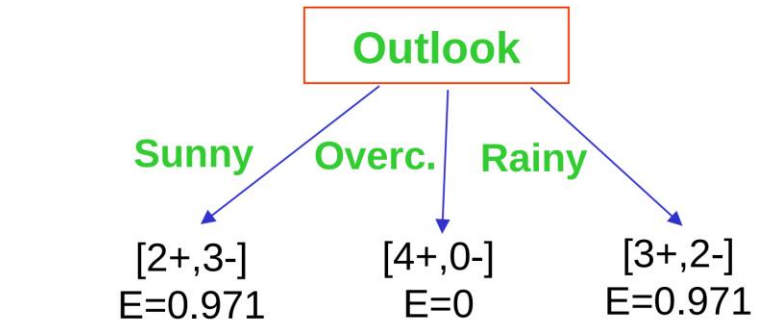



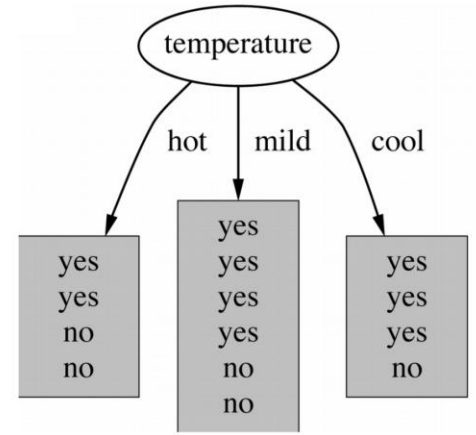
# Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)



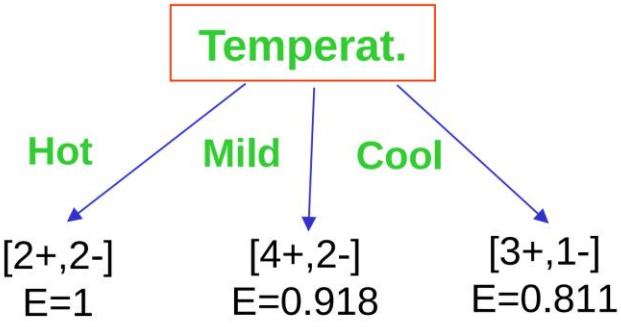

S:[9+,5-]
E=0.940




Gain(S, Outlook)=0.940-(5/14)0.971-0-(5/14)0.971
=0.266

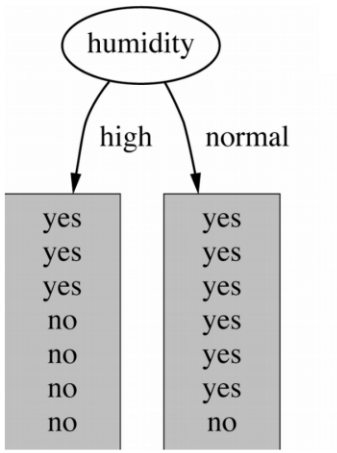



S:[9+,5-]
E=0.940




Gain(S, Temp.)=0.940-(4/14)-(6/14)0.918 -
(4/14)0.811
=0.029

# Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)




S:[9+,5-]
E=0.940

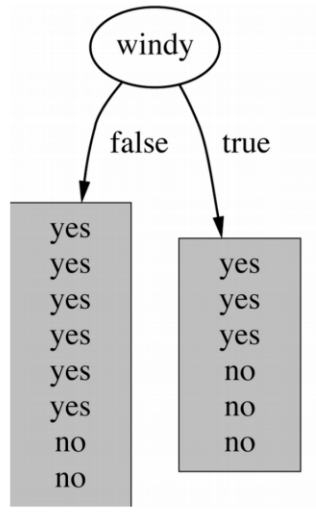
**Humidity**

**High** **Normal**

[3+,4-]
E=0.985

[6+,1-]
E=0.592

Gain(S, Humidity)=0.940-(7/14)0.985-(7/14)0.592
=0.151




S:[9+,5-]
E=0.940

**Windy**

**False** **True**

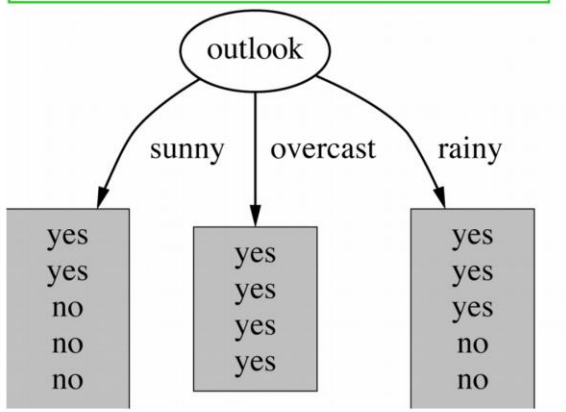
[6+,2-]
E=0.811

[3+,3-]
E=1

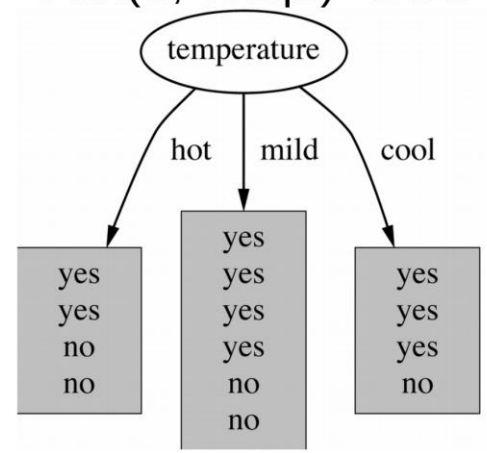
Gain(S, Windy)=0.940-(8/14)0.985-(6/14)
=0.048

Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)

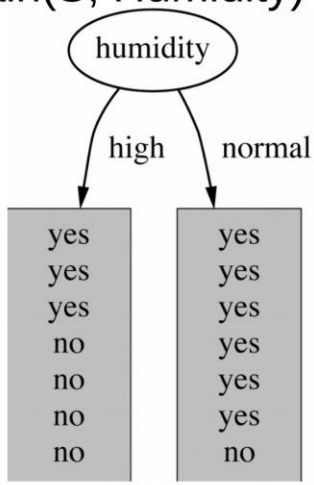
**Gain(S, Outlook)=0.266**



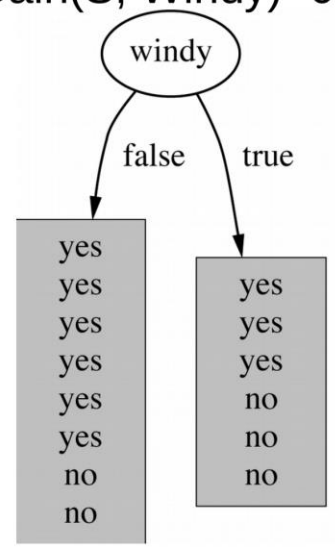

Gain(S, Temp.)=0.029




Gain(S, Humidity)=0.151

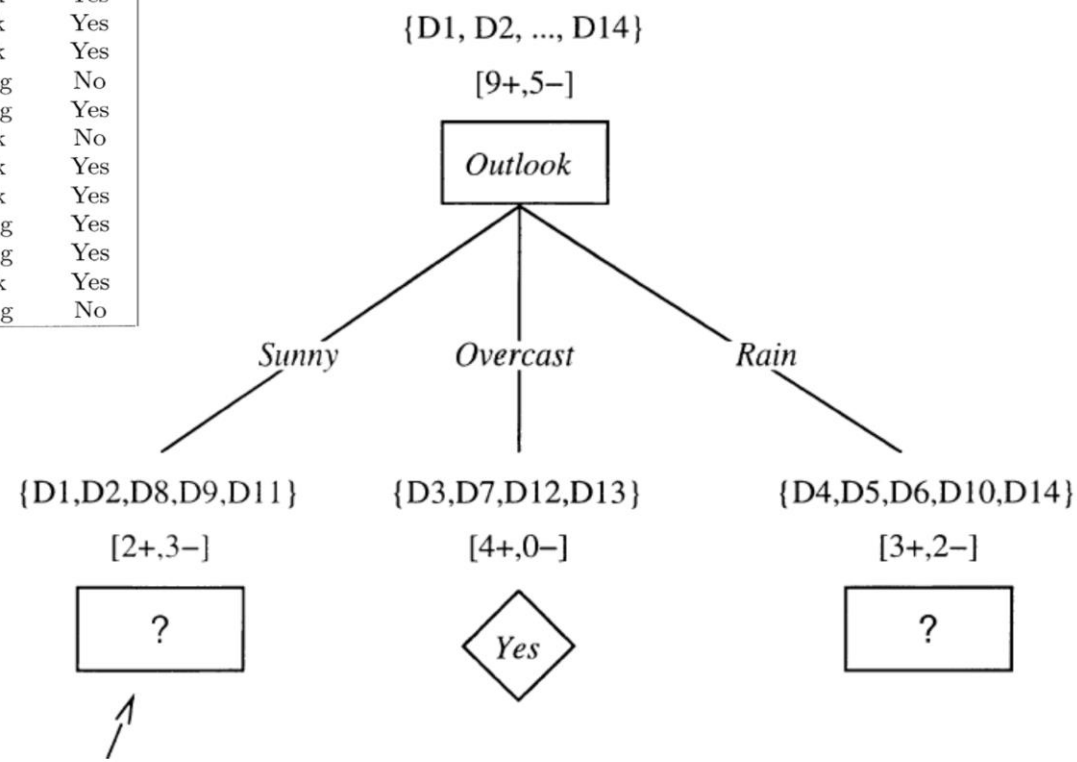



Gain(S, Windy)=0.048



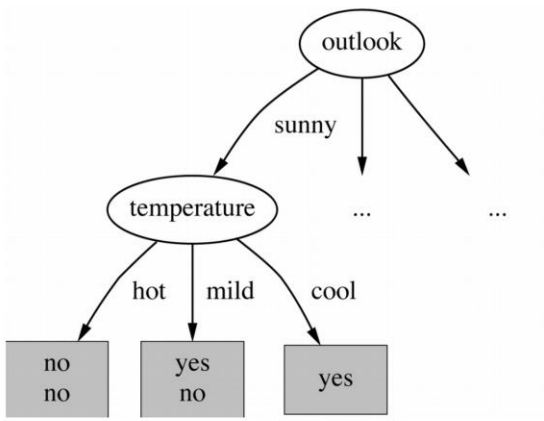
# Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)

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

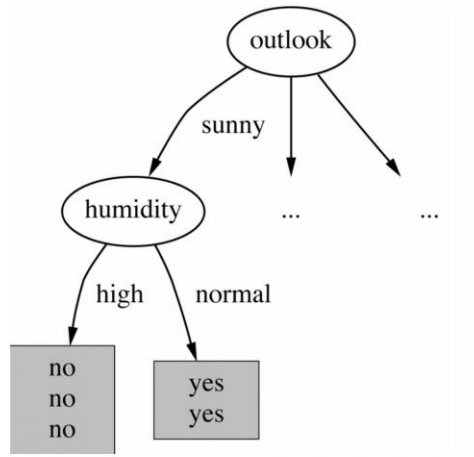



¿Cuál atributo?

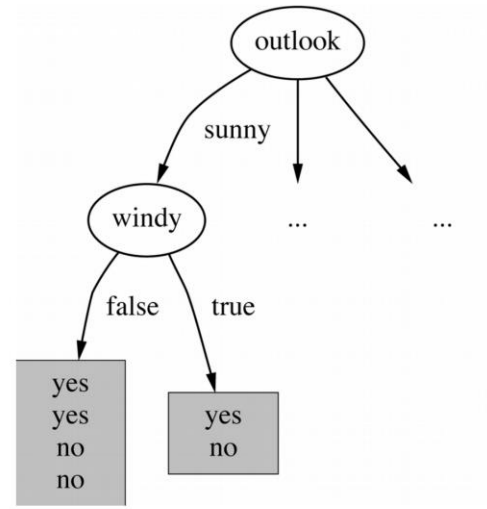
# Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)




Gain(S,Temp.)=?

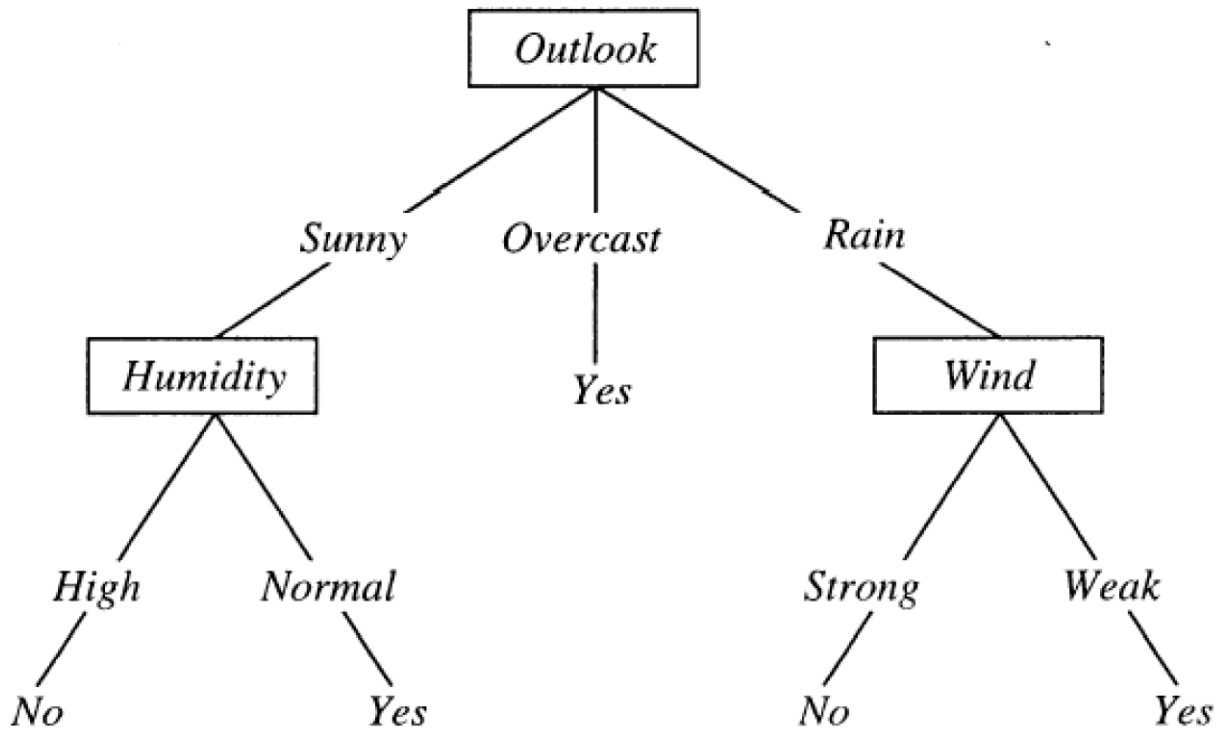



Gain(humid.,Temp.)=?




Gain(S,Windy)=?

Elegimos el atributo que entrega la mayor ganancia de información (mayor reducción de entropía)



¿Qué pasa con IG si hay muchos posibles valores para los atributos?

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

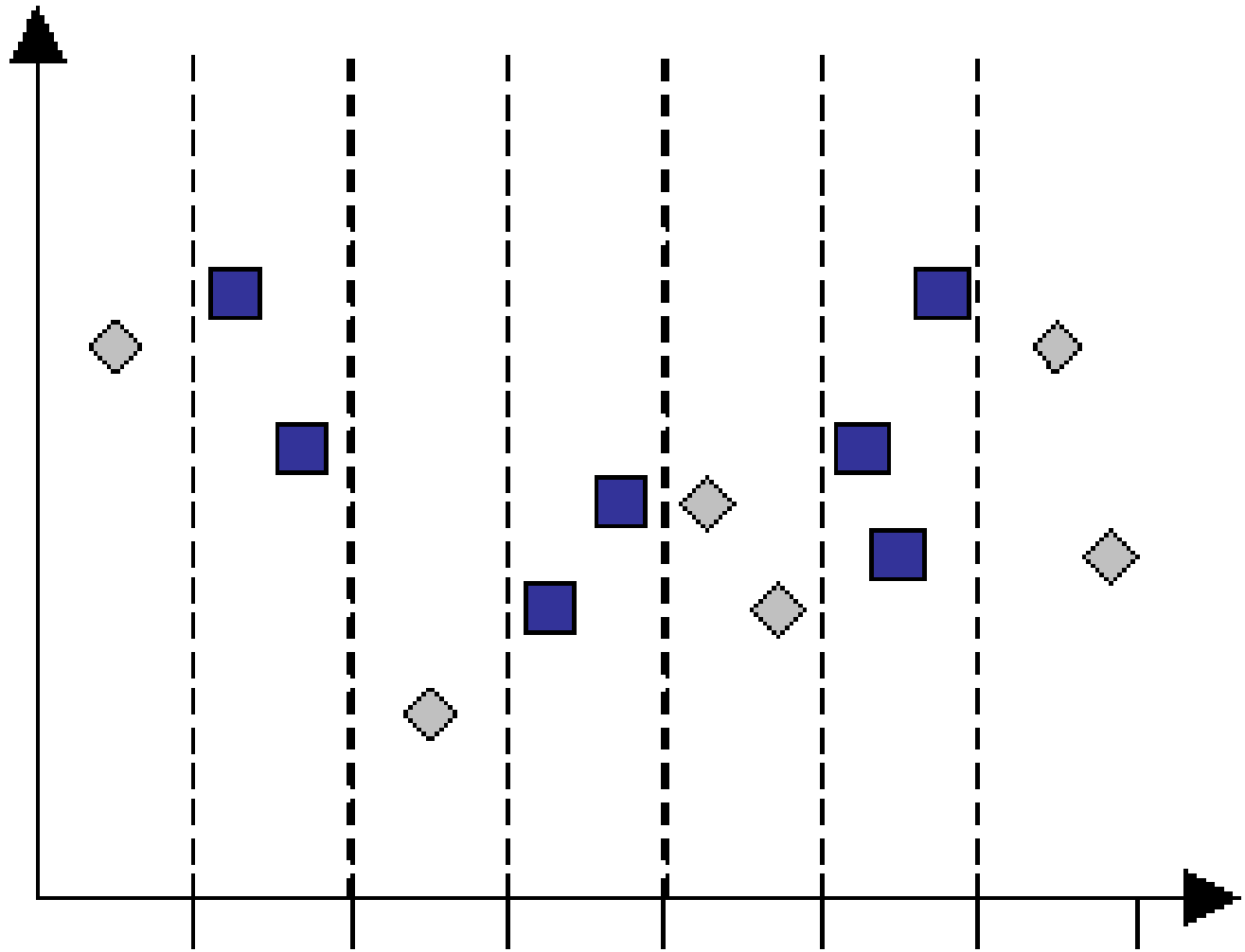
Si IG anda mal, se puede usar Gain Ratio

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

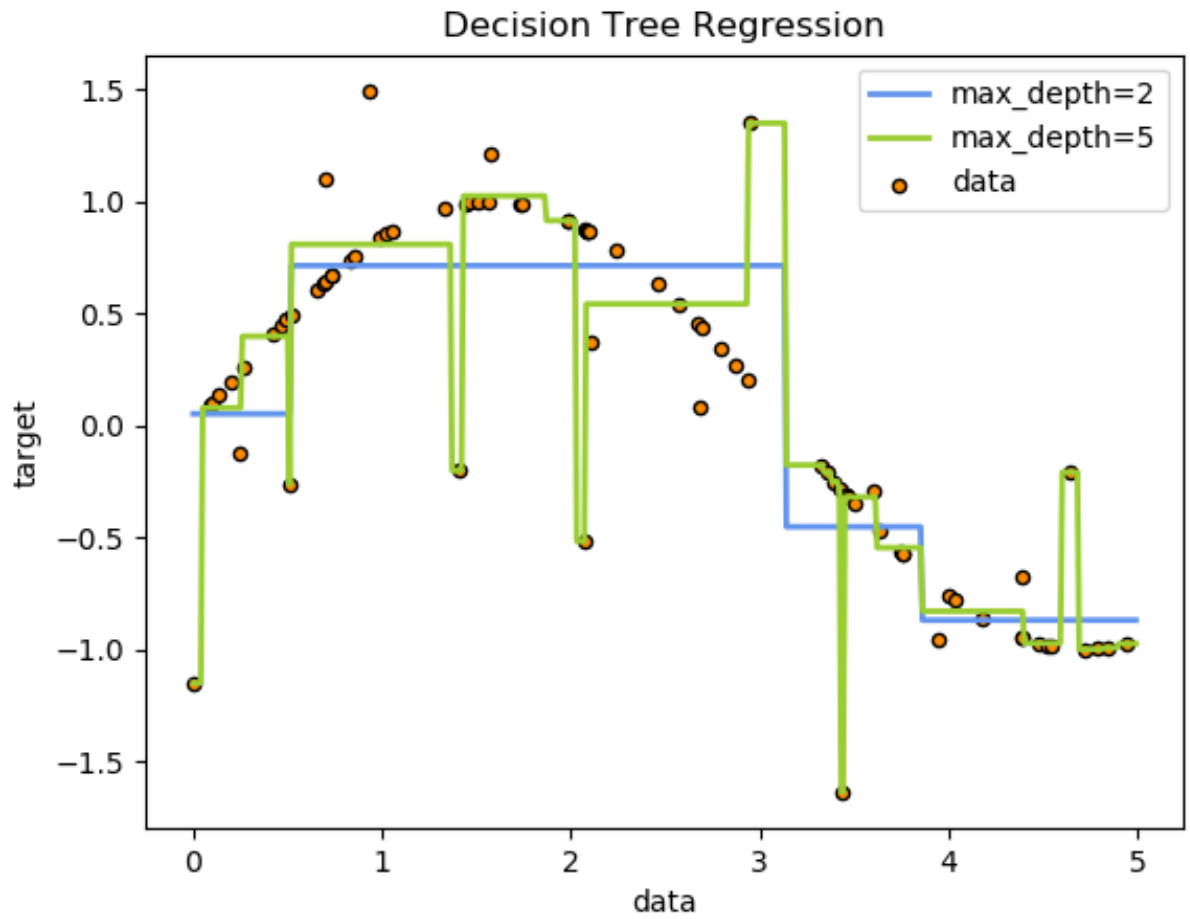
# Quedan aún dos preguntas relevantes

- ¿Cómo funciona este algoritmo si tenemos variables numéricas?
    1. Fuerza bruta
    2. Ordenar por dimensión, y evaluar split en cada cambio de categoría.
- ¿Cómo se puede hacer regresión usando árboles de este tipo?



# Quedan aún dos preguntas relevantes

1. ¿Cómo funciona este algoritmo si tenemos variables numéricas?
    - Fuerza bruta
    - Ordenar por dimensión, y evaluar split en cada cambio de categoría.
2. ¿Cómo se puede hacer regresión usando árboles de este tipo?
    - En vez de medir IG, se calcula la desviación cuadrática con respecto a la media.
    - Construcción recursiva sigue la misma idea que para clasificación.


Decision Tree Regression
max_depth=2
max_depth=5
data
target
data
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
1
2
3
4
5

# Volvamos un poco a overfitting y complejidad

¿Qué tipo de árboles prefiere construir el algoritmo que recién analizamos?

(sesgo inductivo)

Árboles pocos profundos (mientras más arriba aumenta la información, mejor)

¿Tiene esto algo que ver con el sobreajuste?

# Occam's Razor[1] (o la navaja de Occam, claramente una traducción poco afortunada) be my guide

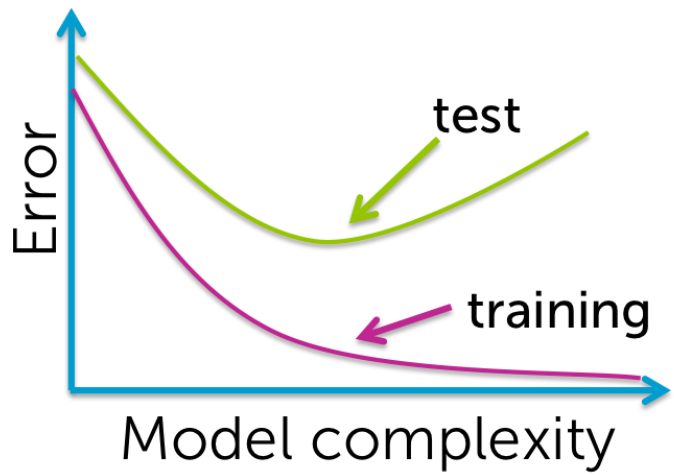
*En igualdad de condiciones, la explicación más sencilla suele ser la más probable [2]*

[1] Supuestamente fue enunciado mientras se afeitaba.
[2] Esto es un principio filosófico/metodológico, no una ley de la naturaleza.

# Overfitting es un problema importante para los árboles de decisión

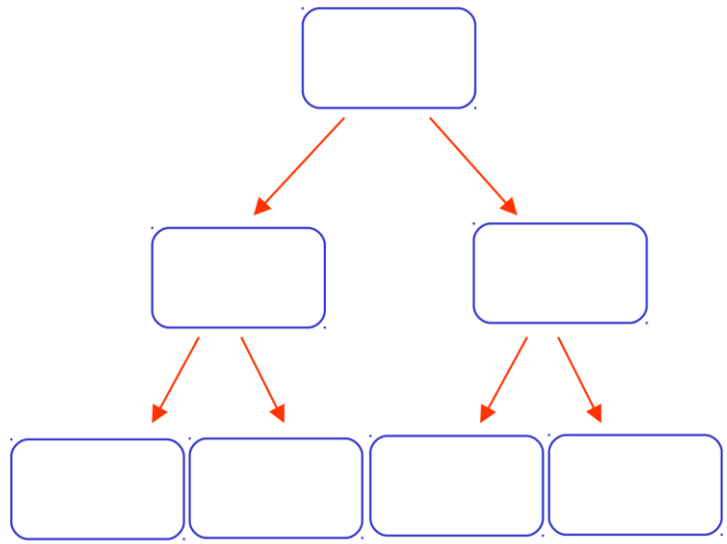
Existen varias técnicas que pueden ayudar reduciendo el overfitting.

- Detener construcción del árbol en base a un set de validación.
- Detener construcción cuando registros restantes no son estadísticamente significativos.
- Construir un árbol completo y luego podar ramas completas.
- Penalizar complejidad en métrica de selección del siguiente atributo.



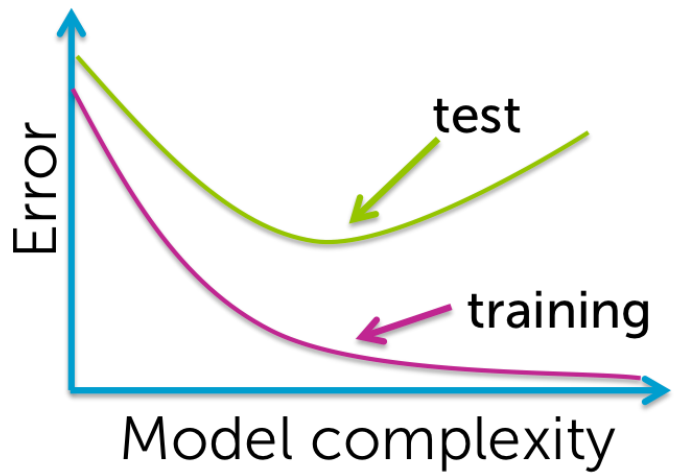
# Árboles de decisión son ampliamente utilizados en la práctica

- Gran ventaja radica en la simplicidad y facilidad de interpretación.
- Pueden sufrir de serios problemas de sobreajuste.



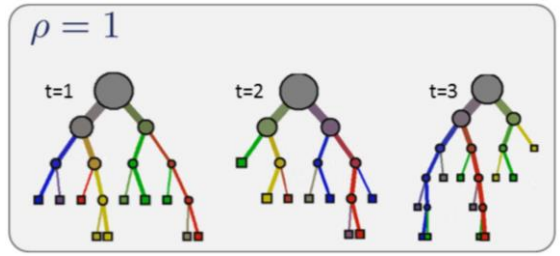
# No podemos quedarnos con el último punto de la diapositiva anterior

- ¿Existe alguna manera más "fundamental" de evitar el overfitting?
- ¿De donde proviene este problema?
  - Muestra es progresivamente reducida al bajar en el árbol.
  - Si hay muchos atributos, existe alta probabilidad de elegir alguno "inútil".



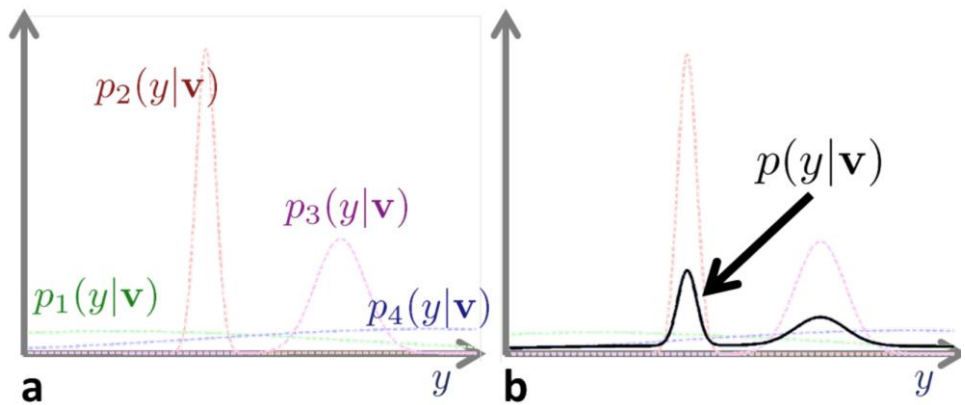
# Ensambles de modelos

- Si queremos estar seguros de algo, le preguntamos a más de un modelo.
- Si le preguntamos a suficientes, es probable que el promedio sea cercano a la verdad.
- ¿Qué condición deben cumplir estos modelos para que esto funcione?



# Ensambles de modelos
## no correlacionados

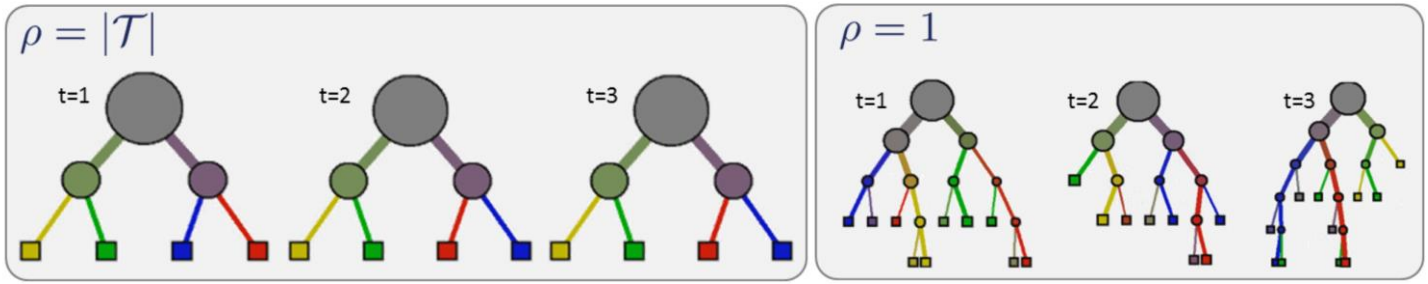
- Si el patrón de error (acierto) es distinto para todos, es altamente probable que en promedio, la respuesta del ensamble sea correcta.




¿Cómo logramos esta especialización
o diferenciación?

# La solución está en las muestras aleatorias

- Si cada modelo ve sólo una parte de los datos (elegida aleatoriamente), la correlación entre ellos disminuye.



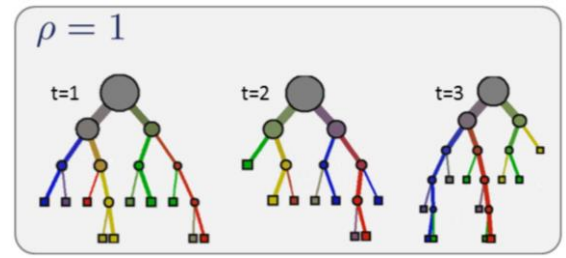

**a)** Low randomness, high tree correlation

**b)** High randomness, low tree correlation

¿Qué pasa si hay una variable muy buena?

# La solución está nuevamente en las muestras aleatorias, pero ahora de atributos

- Cada vez que se hace un *split*, se elige un subconjunto aleatorio de los atributos.
- Luego, para cada uno de ellos, se calcula la ganancia de información.
- Esto trae además el beneficio de reducir la complejidad.
- Por lo general, se utiliza una muestra proporcional a la raíz del número de atributos.



# La introducción de todas estas estrategias "aleatorizantes", transforma a los árboles en *random forests*

**Algorithm 15.1** *Random Forest for Regression or Classification.*

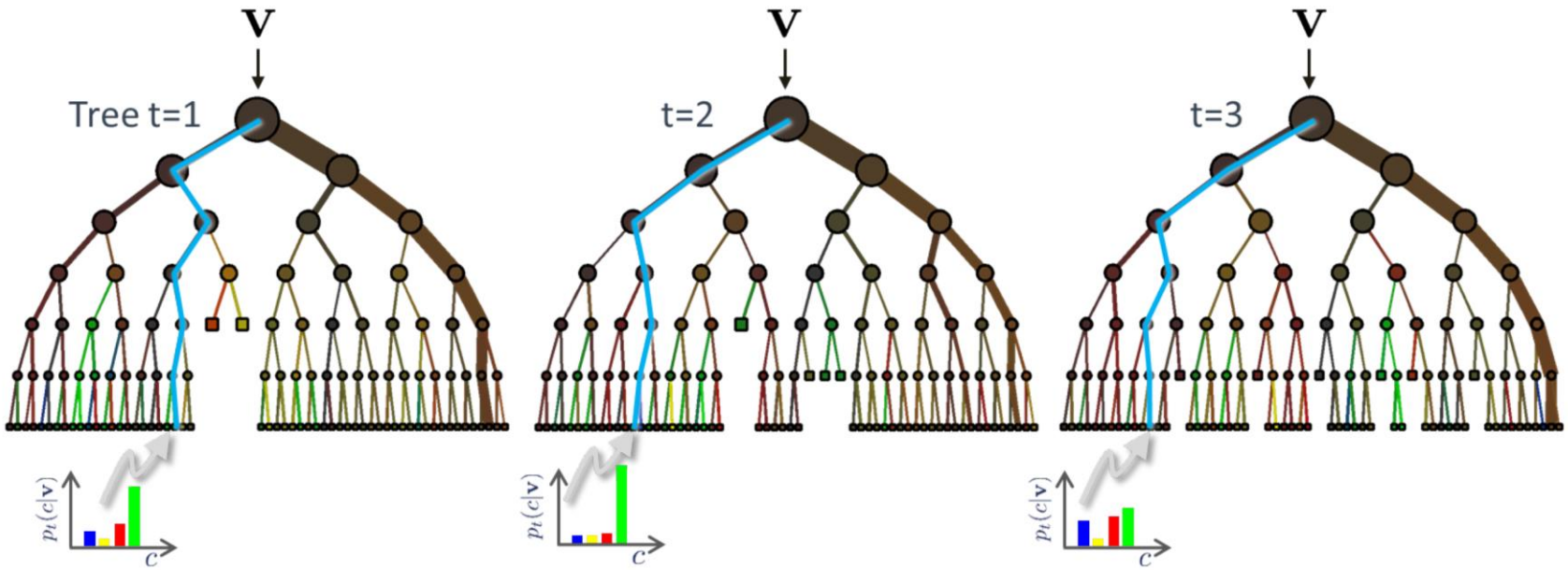
1. For $b = 1$ to $B$:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size $N$ from the training data.
   (b) Grow a random-forest tree $T_b$ to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $n_{min}$ is reached.
      i. Select $m$ variables at random from the $p$ variables.
      ii. Pick the best variable/split-point among the $m$.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point $x$:

*Regression:* $\hat{f}_{\text{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

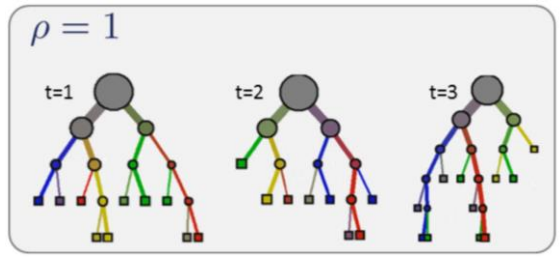
*Classification:* Let $\hat{C}_b(x)$ be the class prediction of the $b$th random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

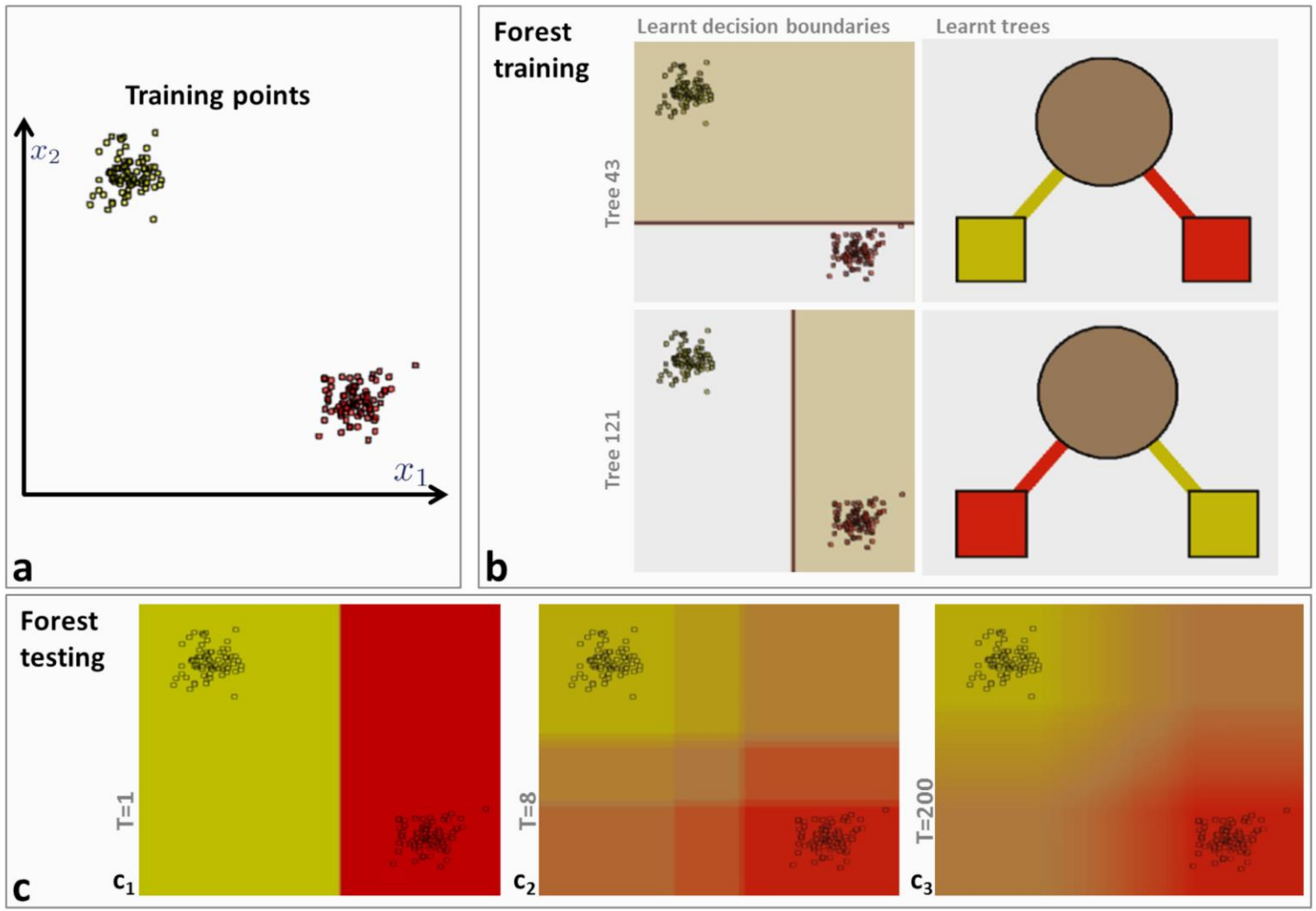
La introducción de todas estas estrategias “aleatorizantes”, transforma a los árboles en *random forests*

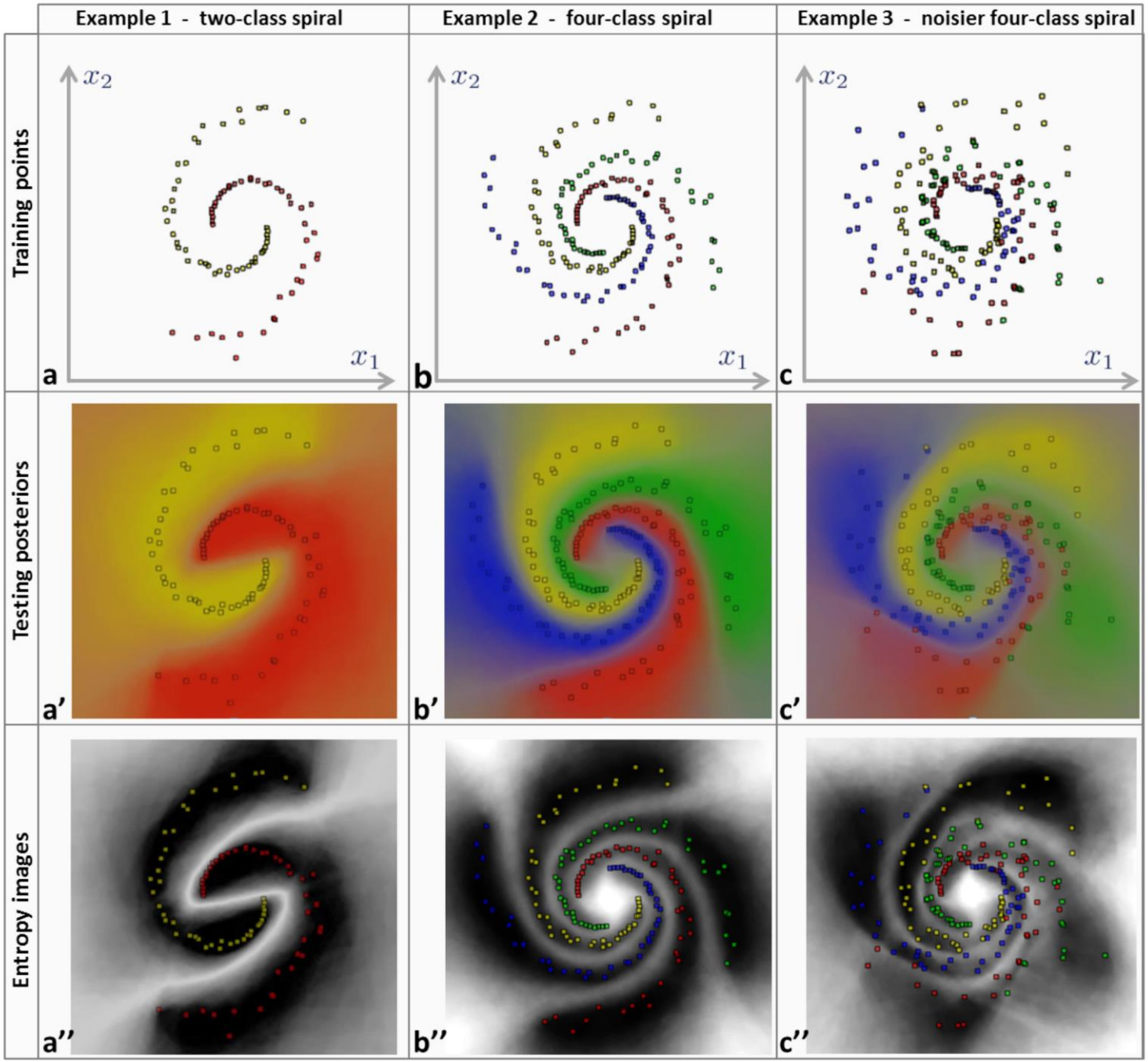


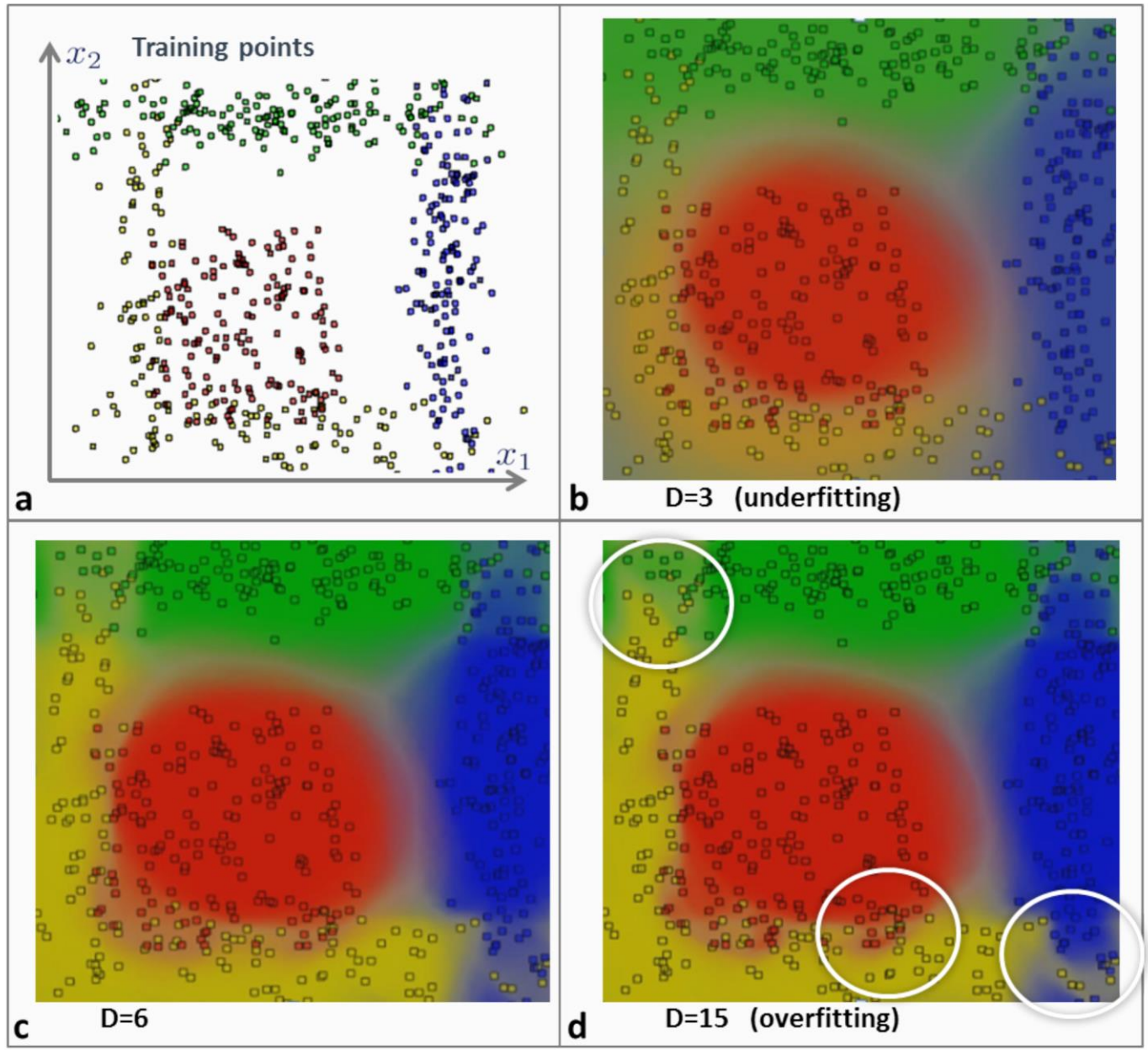
# Algunos consideraciones relacionadas con la evaluación

- Es posible medir el rendimiento usando ejemplos no vistos por lo árboles (*out-of-bag error*)
- Para medir la importancia de una variable, basta con permutar su valor entre los ejemplos (¿por qué?).



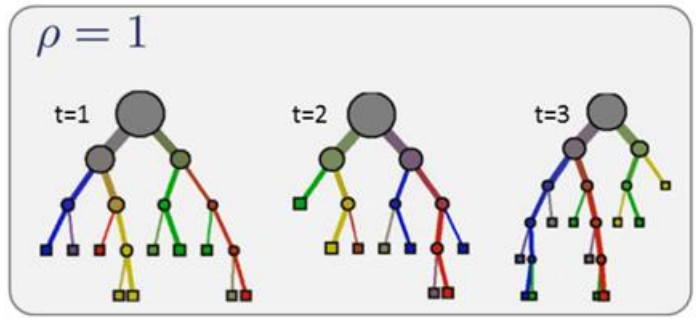

Training points
$x_2$
$x_1$
a
Forest training
Learnt decision boundaries
Learnt trees
Tree 43
Tree 121
b
Forest testing
T=1
T=8
T=200
$c_1$
$c_2$
$c_3$
c


Example 1 - two-class spiral
Example 2 - four-class spiral
Example 3 - noisier four-class spiral
Training points
Testing posteriors
Entropy images
$x_2$
$x_1$
a
b
c
a'
b'
c'
a''
b''
c''


Training points
$x_2$
$x_1$
a
b
D=3 (underfitting)
c
D=6
d
D=15 (overfitting)

# *Random forests* son ampliamente utilizados en la práctica

- Al igual que los árboles de decisión, son modelos fácilmente interpretables.
- Rendimiento es altamente competitivo en problemas con datos *estructurados*.
- Gracias a aleatorización en su construcción, son altamente resistentes al *overfitting*.



Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación


# IIC2613 – Inteligencia Artificial

## Árboles de decisión

**Profesor:** Hans Löbel