決策樹 (Decision Tree)

Dr. Tun-Wen Pai

1)使用決策樹的優缺點
2)決策樹架構
3)從建構決策樹所學到的經驗
4)決策樹演算法 (ID3)
5) Entropy and Information Gain

March 15, 2023

Definition of Decision Tree

- A **decision tree** is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

- **Decision trees** are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

https://en.wikipedia.org/wiki/Decision_tree

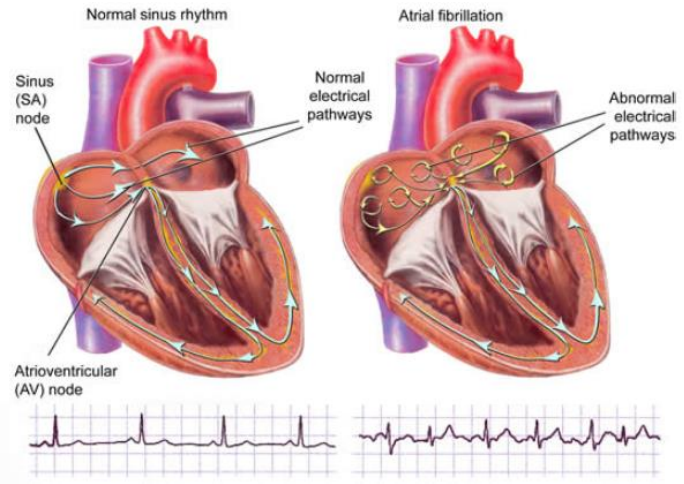
How to realize the decision tree?

- Imagine a conversation between a doctor and a patient.
  - Doctors ask questions to determine the cause of the ailment. The doctor would continue to ask questions, till she/he is able to arrive at a reasonable decision.
  - If nothing seems plausible, the doctor might recommend some tests to generate more data and options.

- This is how experts in any field solve problems.
  - Use decision trees or decision rules.
  - For every question they ask, the potential answers create separate branches for further questioning.
  - For each branch, the expert would know how to proceed ahead.
  - The process continues until the end of the tree is reached, which means a leaf node is reached.

Questions vs. Decision Tree

- The decision tree showed that if Blood Pressure was higher than (140/90mmHg), the chance of another heart attack was very high (70%).
- If the patient's Blood Pressure was ok, the next question to ask was the patient's age.
- If the age was low (<=62), then the patient's survival was almost guaranteed (98%).
- If the age was higher, then the next question to ask was about sinus node problems.
- If their sinus was ok, the chances of survival were 89%. Otherwise, the chance of survival dropped to 50%.
- This decision tree predicts 86.5% of the cases correctly.



(Source: Salford Systems).

Another Example

Iris Versicolor

Iris Setosa

Iris Virginica

- A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a **class label** (decision taken after computing all attributes). The paths from root to leaf represent **classification rules**.

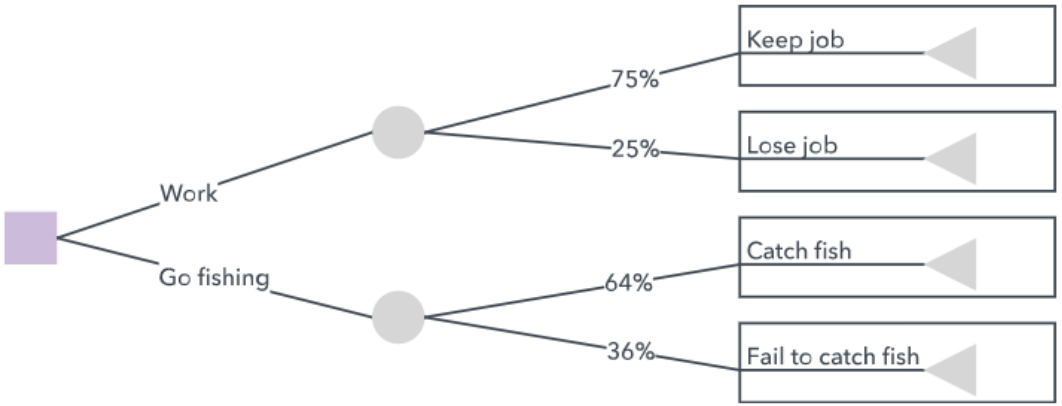
- In decision analysis, a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated.

- A decision tree diagram consists of three types of nodes:

Decision nodes – typically represented by squares(a decision to be made)

Chance nodes – typically represented by circles (probabilities of certain results)

End nodes – typically represented by triangles (final outcome of a decision path)



使用決策樹的優點

• Decision tree can be visualized and simple to understand and to interpret.

• Requires **little data preparation**. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module **does not** support missing values.

•The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.

•Able to handle both numerical and categorical data. Other techniques are usually specialised in analysing datasets that have only one type of variable.

•Able to handle multi-output problems.

•Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.

•Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.

•Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

使用決策樹的缺點

•Decision-tree learners can create **over-complex** trees that do not generalize the data well. This is called **overfitting**. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.

•Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an **ensemble**(random forest).

•The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.

•There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

•Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

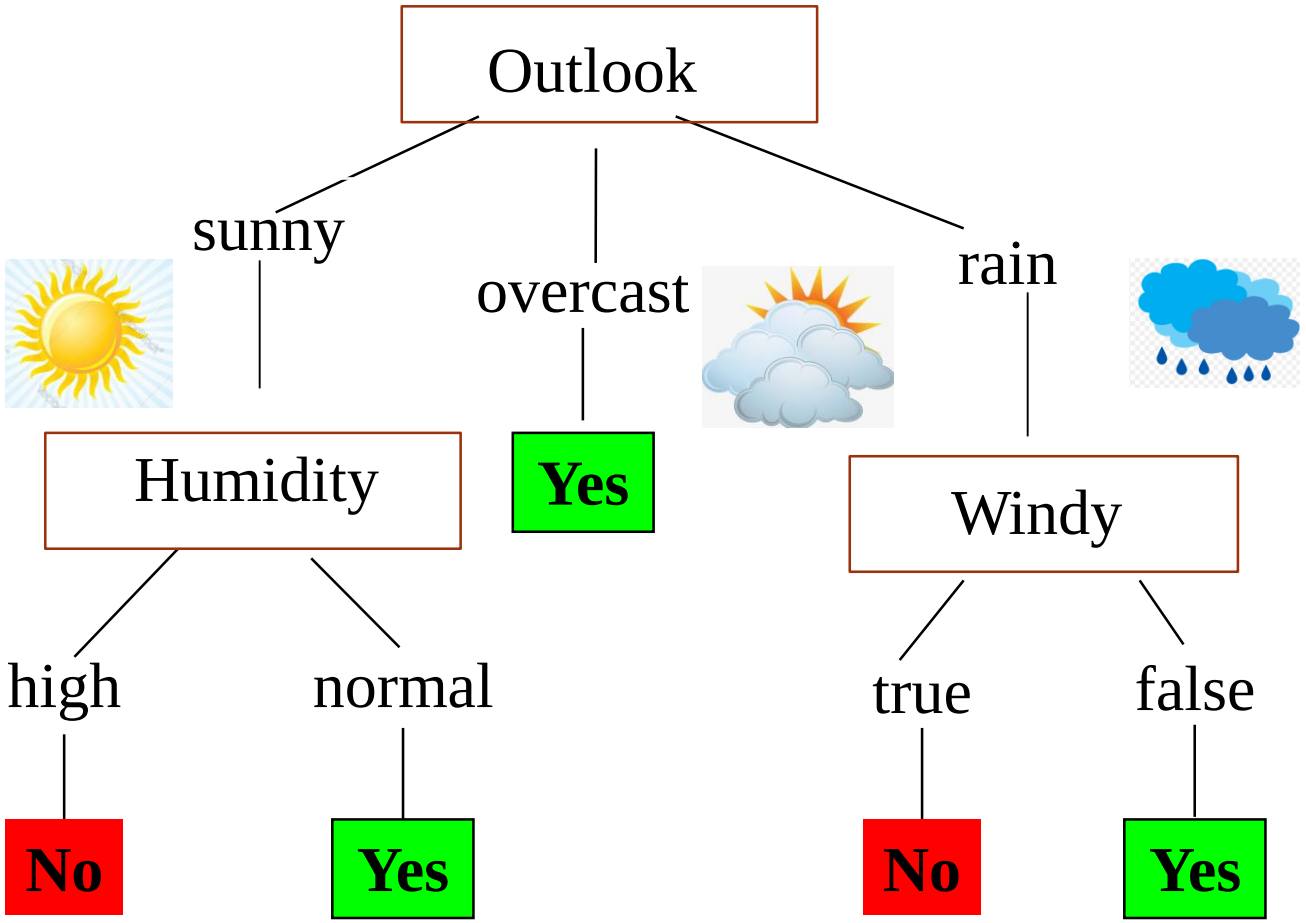
決策樹的問題

如何立決策樹?

舉例: 關於是否核可開放戶外比賽(遊戲)的決策?

外觀	氣溫	溼度	刮風	開放遊戲
晴朗	炎熱	正常	真	？？

決策樹範例


Outlook
sunny
overcast
rain
Humidity
Yes
Windy
high
normal
true
false
No
Yes
No
Yes

外觀	氣溫	溼度	刮風	開放遊戲
晴朗	炎熱	高	偽	否
晴朗	炎熱	高	真	否
陰天	炎熱	高	偽	是
雨天	溫和	高	偽	是
雨天	涼爽	正常	偽	是
雨天	涼爽	正常	真	否
陰天	涼爽	正常	真	是
晴朗	溫和	高	偽	否
晴朗	涼爽	正常	偽	是
雨天	溫和	正常	偽	是
晴朗	溫和	正常	真	是
陰天	溫和	高	真	是
陰天	炎熱	正常	偽	是
雨天	溫和	高	真	否

P6-4

決策樹架構

決策樹是一種層級式分支結構。

屬性	規則	錯誤	錯誤合計
外觀	晴朗→否	2/5	

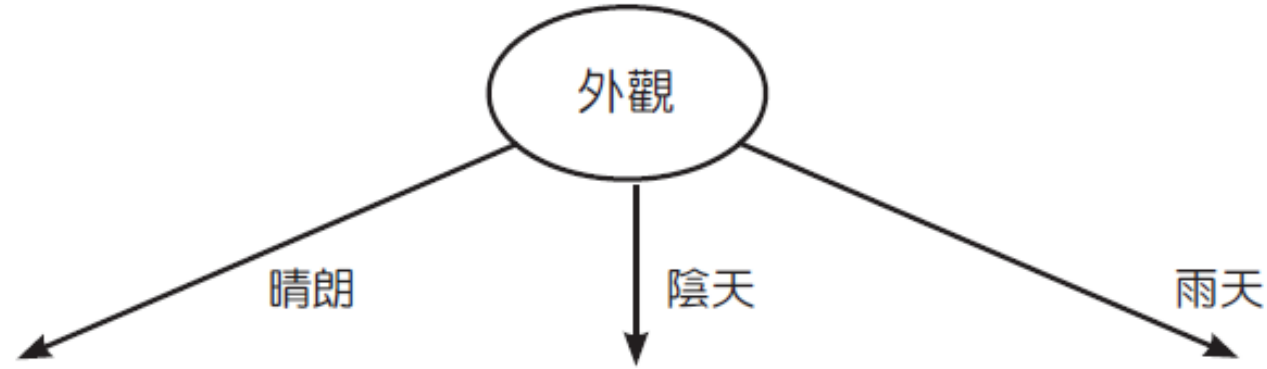
屬性	規則	錯誤	錯誤合計
外觀	晴朗→否	2/5	
	陰天→是	0/4	

屬性	規則	錯誤	錯誤合計
外觀	晴朗→否	2/5	4/14
	陰天→是	0/4	
	雨天→是	2/5	

屬性	規則	錯誤	錯誤合計
外觀	晴朗→否	2/5	4/14
	陰天→是	0/4	
	雨天→是	2/5	
氣溫	炎熱→否	2/4	5/14
	溫和→是	2/6	
	涼爽→是	1/4	
溼度	高→否	3/7	4/14
	正常→是	1/7	
刮風	偽→是	2/8	5/14
	真→否	3/6	

分割決策樹

第一層的分割後，決策樹看起來像這樣。



氣溫	溼度	刮風	開放
炎熱	高	偽	否
炎熱	高	真	否
溫和	高	偽	否
涼爽	正常	偽	是
溫和	正常	真	是

氣溫	溼度	刮風	開放
溫和	高	偽	是
涼爽	正常	偽	是
涼爽	正常	真	否
溫和	正常	偽	是
溫和	高	真	否

外觀晴朗

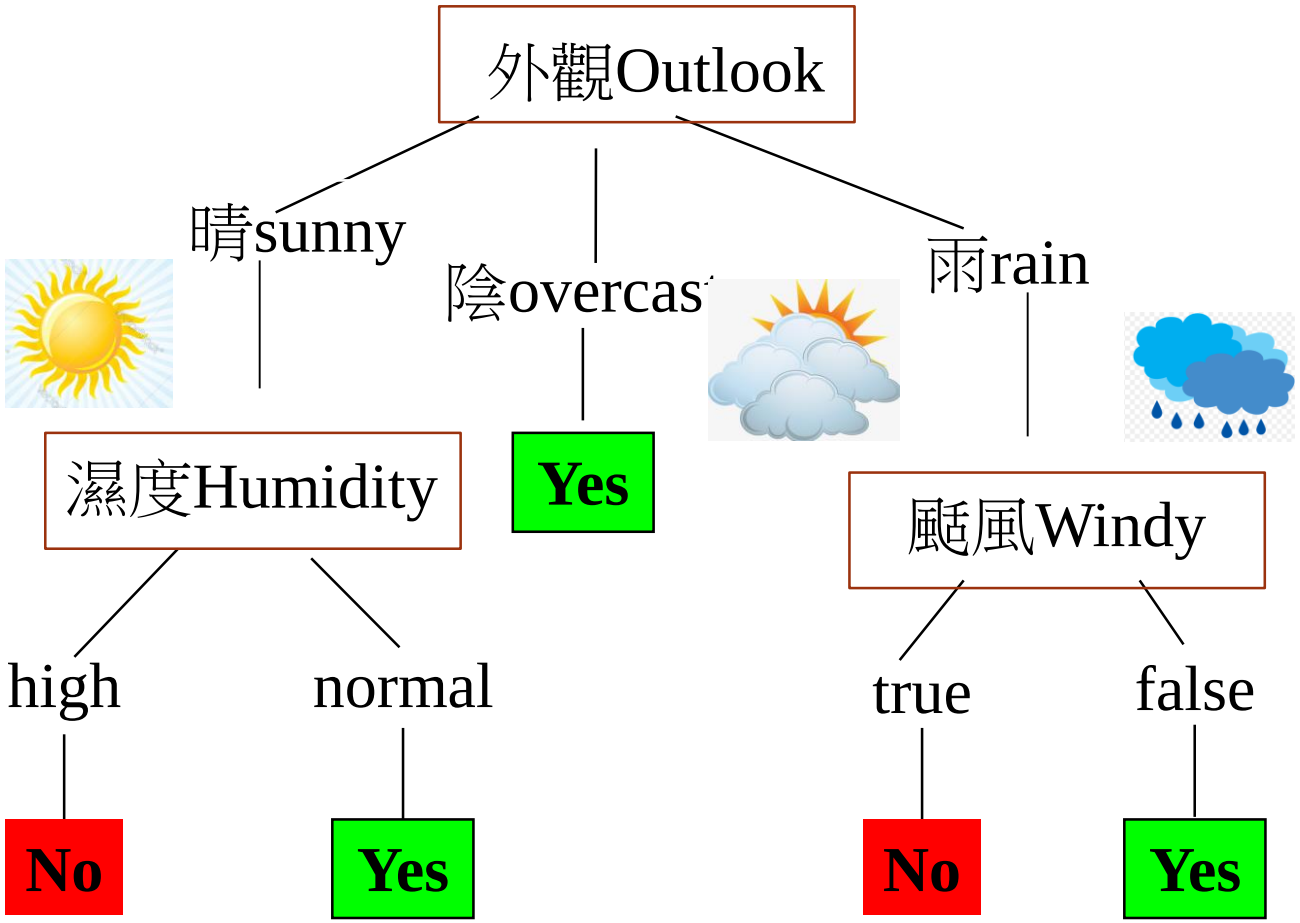
屬性	規則	錯誤	錯誤合計
氣溫	炎熱→否	0/2	1/5
	溫和→否	1/2	
	涼爽→是	0/1	
溼度	高→否	0/3	0/5
	正常→是	0/2	
刮風	偽→否	1/3	2/5
	真→是	1/2	

P6-8 參照書籍

外觀雨天

屬性	規則	錯誤	錯誤合計
氣溫	溫和→是	1/3	2/5
	涼爽→是	1/2	
溼度	高→否	1/2	2/5
	正常→是	1/3	
刮風	偽→是	0/3	0/5
	真→否	0/2	

決策樹範例


外觀Outlook
晴sunny
陰overcas
雨rain
濕度Humidity
Yes
颳風Windy
high
normal
true
false
No
Yes
No
Yes

決策樹可用來解決目前的問題。以下再次列出問題。

外觀	氣溫	溼度	刮風	開放遊戲
晴朗	炎熱	正常	真	？？

根據決策樹，第一個詢問的問題是外觀。在這個問題裡，外觀是晴朗。因此，決策問題便移至決策樹的「晴朗」分支。該子樹的節點為溼度。在問題裡，溼度為正常。該分支會導引至「是」的回答。因此，回覆是否開放問題的答案便是「是」。

外觀	氣溫	溼度	刮風	開放遊戲
晴朗	炎熱	正常	真	是

P6-9~10

從建構決策樹所學到的經驗

▼表 6.1：比較決策樹與表格查找

	決策樹	表格查找
準確度	不同準確度層級	100% 準確
通用性	一般。適用所有狀況。	只適用早先曾發生類似狀況時。
精簡度	只需要 3 種變數	需要所有 4 種變數
簡單	只需要 1 種，最多 2 種變數	4 種變數值皆需要
容易	邏輯化，並容易了解	查找起來可能很麻煩；不必了解決策背後的邏輯

決策樹演算法

製作決策樹的虛擬碼：

1. 建立一個根節點，並指派所有訓練資料至其中。
2. 根據特定條件，選擇最佳的分枝屬性。
3. 在根節點為每項分枝值加入分支。
4. 沿著特定分枝路線，將資料分枝至互斥的子集。
5. 為每個葉節點重複步驟2 與3，直到達到停止條件為止。

決策樹演算法基於三項主要元素而有所不同：

1. 分枝準則（Splitting criteria）
2. 停止條件 (Stopping)
3. 修剪 (pruning)

決策樹	C4.5	CART	CHAID
全名	Iterative Dichotomiser（ID3）	分類與迴歸樹	卡方自動交互偵測
基礎演算法	Hunt's 演算法	Hunt's 演算法	調整的顯著性檢定
開發者	Ross Quinlan	Bremman	Gordon Kass
何時開發	1986	1984	1980
決策樹類型	分類	分類與迴歸樹	分類與迴歸
連續實作	樹生長與樹修剪	樹生長與樹修剪	樹生長與樹修剪
資料類型	離散與連續；不完整資料	離散與連續	亦接受非正常資料
分枝類型	多元分枝	只限二元分枝；聰明替代分枝以降低樹深度	預設為多元分枝
分枝準則	資訊增益	吉尼係數與其他	卡方測定
修剪條件	聰明由下而上技術避免過度配適	首先移除最弱連結	決策樹可能變得十分龐大
實作	公開提供	在大部分套件中公開提供	常用於市場研究，作為區隔用

▲ 圖 6.2：常見決策樹演算法比較

ID3 決策樹演算法

- ID3 is a mathematical algorithm for building the decision tree invented by J. Ross Quinlan in 1979.
- Uses Information Theory invented by Shannon in 1948.
- Builds the tree from the top down, with no backtracking.
- Information Gain is used to select the most useful attribute for classification.

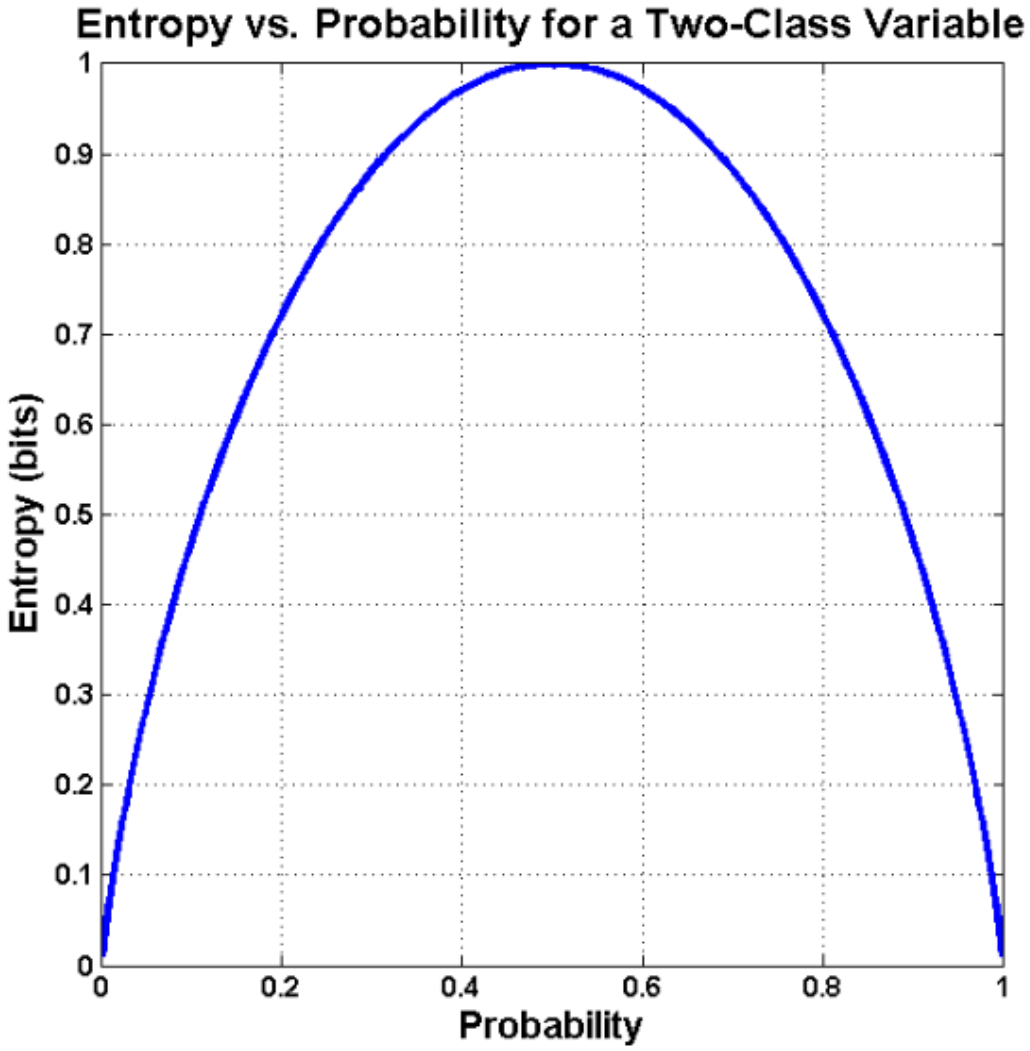
Entropy

- A formula to calculate the homogeneity of a sample.
- A completely homogeneous sample has entropy of 0.
- An equally divided sample has entropy of 1.
- Entropy(S) = - p*$\log_2$ (p) –q*$\log_2$ (q) for a sample of negative (p) and positive (q) elements.
- Define: $\log_2$ (0) = $\log_2$ (1) = 0

The formula for entropy is:

Entropy= - $\sum_i$ p_i (X) $log_2\, p_i(X)$=- (9/14) $\log_2$ (9/14) - (5/14) $\log_2$ (5/14) = 0.940

Uncertainty



** 資訊熵是接收的每個訊息(來自分布或數據流中的事件、樣本或特徵)所包含資訊的平均量

Information Gain (IG)

- The **information gain** is based on the **decrease** in entropy after a dataset is split on an attribute.
- Which attribute creates the most homogeneous branches?
- First the entropy of the total dataset is calculated.
- The dataset is then split according to different attributes.
- The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split.
- The resulting entropy is subtracted from the entropy before the split.

Information Gain

- The result is the **Information Gain(IG)**, or decrease in entropy.
- The attribute that yields **the largest IG is chosen** for the decision node.
- A branch set with entropy of 0 is a leaf node.
- Otherwise, the branch needs further splitting to classify its dataset.
- The ID3 algorithm is run recursively on the non-leaf branches, until all data is classified.

  - Gain(S,A): expected reduction in entropy due to sorting S on attribute A

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

Advantages of using ID3

- Understandable prediction rules are created from the training data.
- Builds the fastest tree.
- Builds a short tree.
- Only need to test enough attributes until all data is classified.
- Finding leaf nodes enables test data to be pruned, reducing number of tests.
- Whole dataset is searched to create tree.

Disadvantages of using ID3

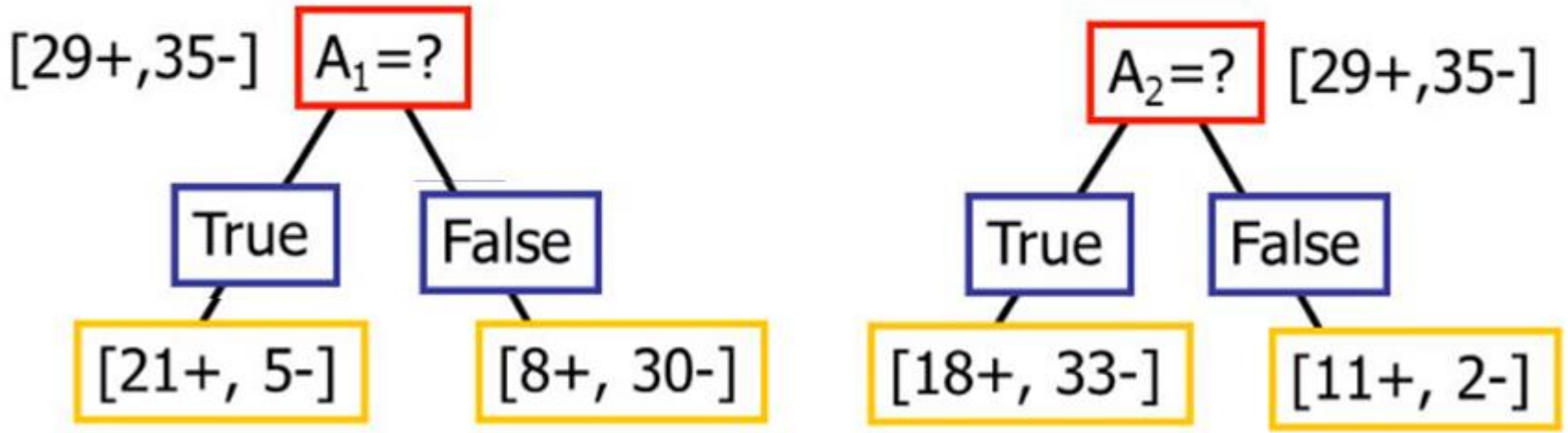
- Data may be over-fitted or over-classified, if a small sample is tested.
- Only one attribute at a time is tested for making a decision.
- Classifying continuous data may be computationally expensive, as many trees must be generated to see where to break the continuum.

Example of IG calculation

- Gain(S,A): expected reduction in entropy due to sorting S on attribute A

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

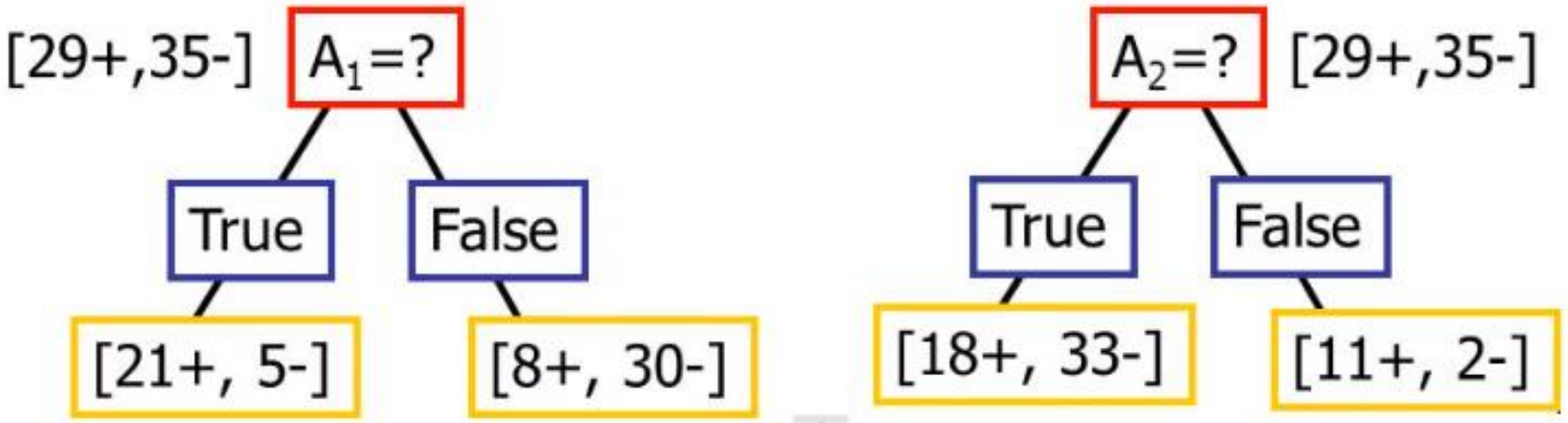
Entropy([29+,35-]) = -29/64 $\log_2$ 29/64 – 35/64 $\log_2$ 35/64
= 0.99



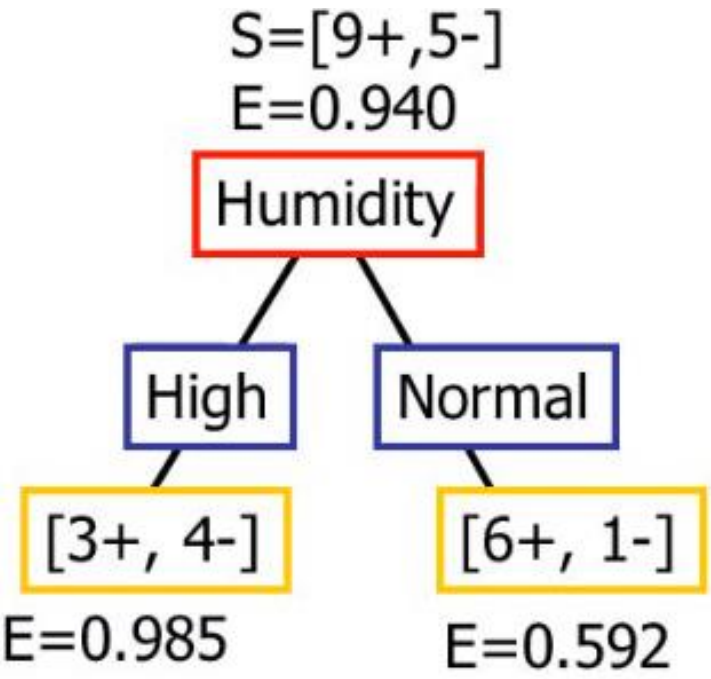
Example of IG calculation

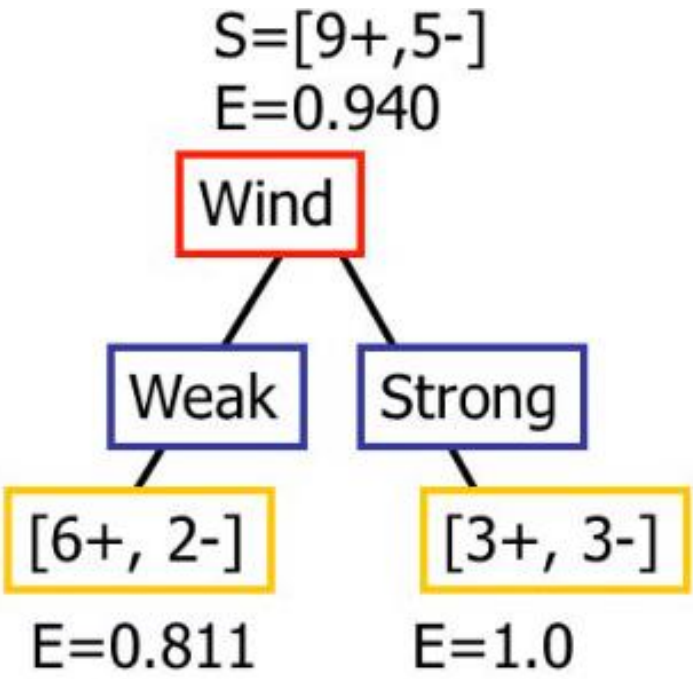
Entropy([21+,5-]) = 0.71
Entropy([8+,30-]) = 0.74
Gain(S,A_1)=Entropy(S)
-26/64*Entropy([21+,5-])
-38/64*Entropy([8+,30-])
=0.27

Entropy([18+,33-]) = 0.94
Entropy([11+,2-]) = 0.62
Gain(S,A2)=Entropy(S)
-51/64*Entropy([18+,33-])
-13/64*Entropy([11+,2-])
=0.12



Day	Outlook	Temp.	Humidity	Wind	Play Tennis
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	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	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

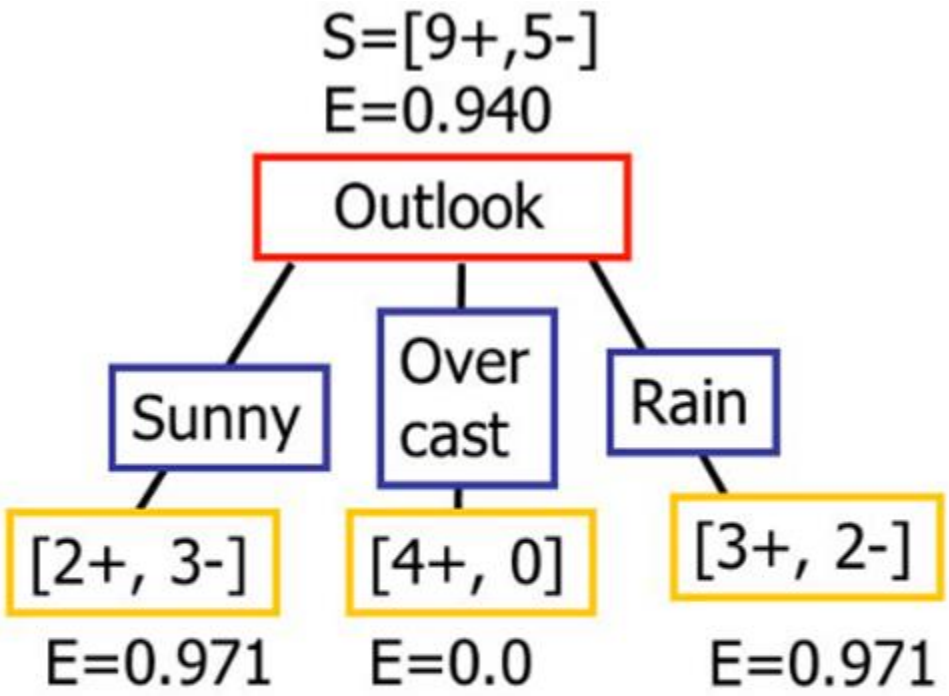




Gain(S,Humidity)
=0.940-(7/14)*0.985
– (7/14)*0.592
=0.151

Gain(S,Wind)
=0.940-(8/14)*0.811
– (6/14)*1.0
=0.048

Humidity provides greater info. gain than Wind, w.r.t target classification.

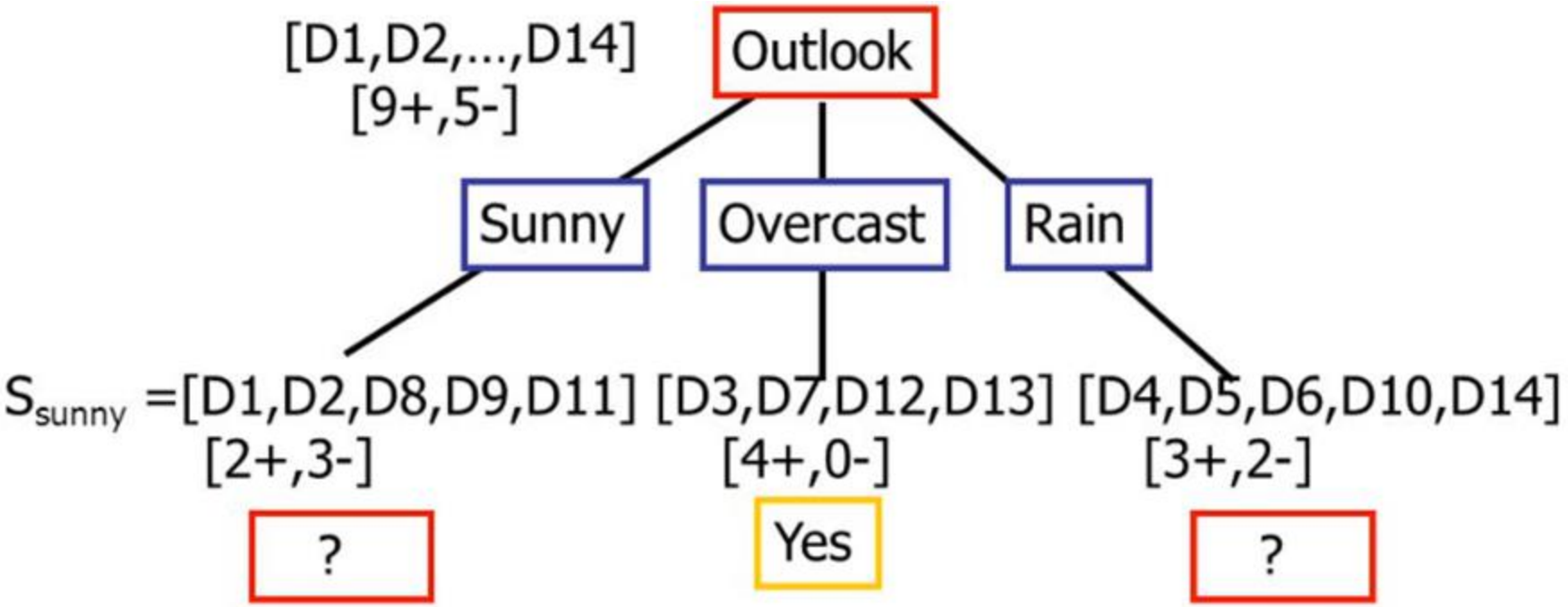


Gain(S,Outlook)
=0.940-(5/14)*0.971
-(4/14)*0.0 – (5/14)*0.971
=0.247

The information gain values for the 4 attributes are:

- Gain(S,Outlook) =0.247
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

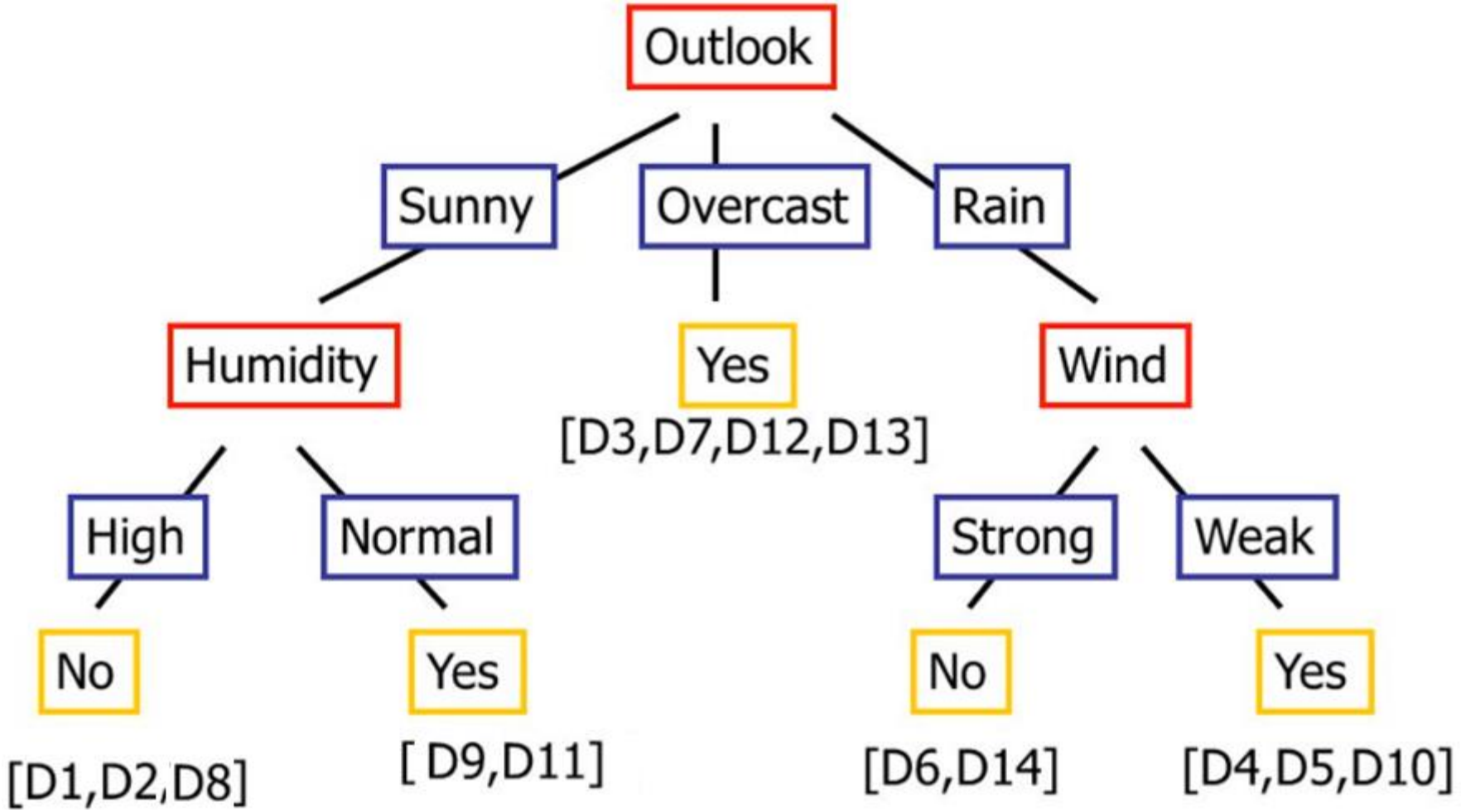
where S denotes the collection of training examples



Gain(S_{sunny}, Humidity)=0.971-(3/5)0.0 – 2/5(0.0) = 0.971

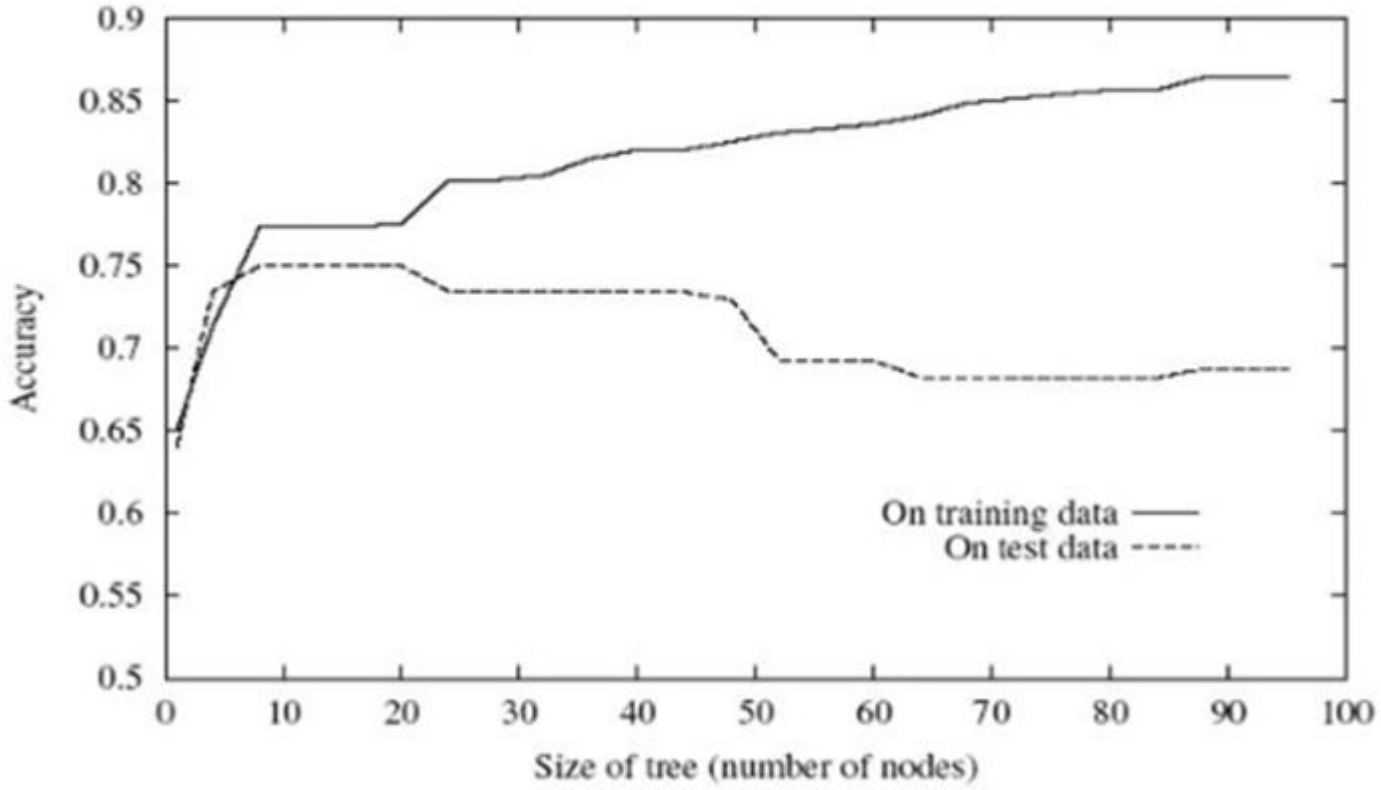
Gain(S_{sunny}, Temp.)=0.971-(2/5)0.0 –2/5(1.0)-(1/5)0.0 = 0.570

Gain(S_{sunny}, Wind)=0.971= -(2/5)1.0 – 3/5(0.918) = 0.019


Outlook
Sunny
Overcast
Rain
Humidity
Yes
[D3,D7,D12,D13]
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes
[D1,D2,D8]
[D9,D11]
[D6,D14]
[D4,D5,D10]

Overfitting

- One of the biggest problems with decision trees is **Overfitting**



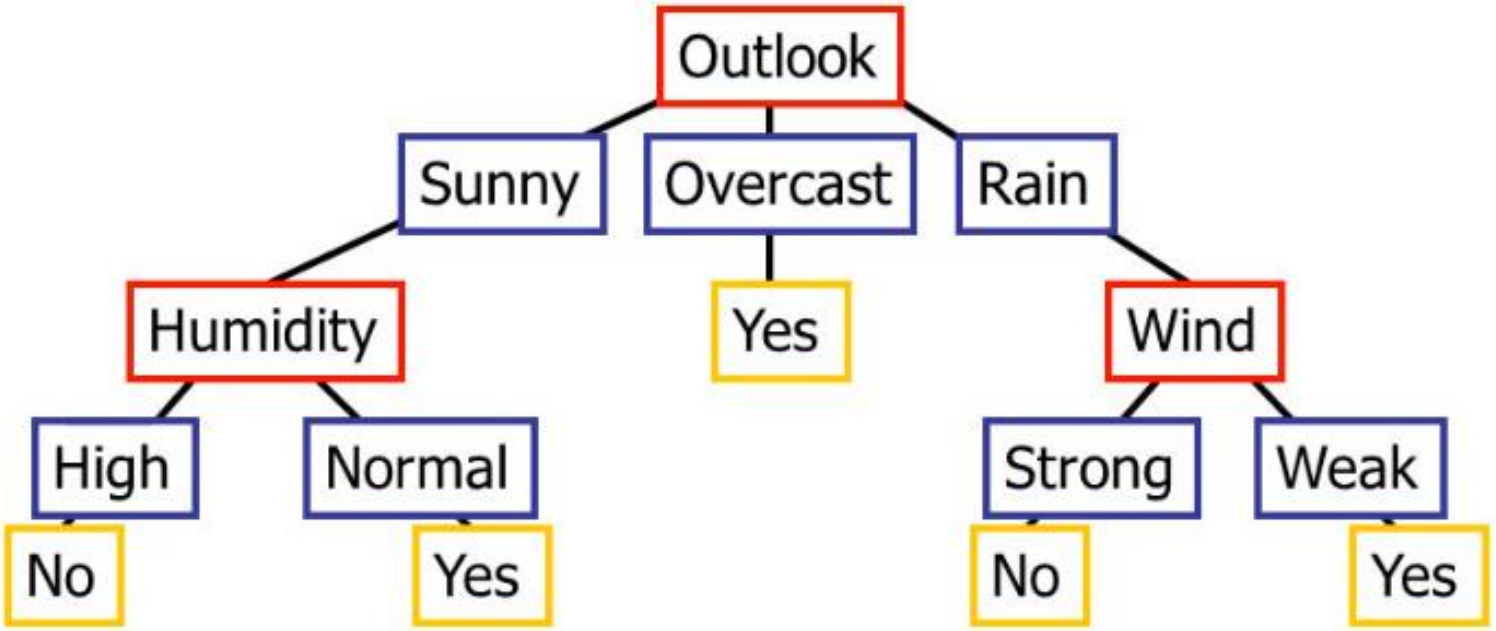
Avoid overfitting

- stop growing when split not statistically significant
- grow full tree, then post-prune

Select "best" tree:

- measure performance over training data
- measure performance over separate validation data set

Converting a tree to Rules



R_1: If (Outlook=Sunny) $\wedge$ (Humidity=High) Then PlayTennis=No
R_2: If (Outlook=Sunny) $\wedge$ (Humidity=Normal) Then PlayTennis=Yes
R_3: If (Outlook=Overcast) Then PlayTennis=Yes
R_4: If (Outlook=Rain) $\wedge$ (Wind=Strong) Then PlayTennis=No
R_5: If (Outlook=Rain) $\wedge$ (Wind=Weak) Then PlayTennis=Yes

Continuous valued attributes

Create a discrete attribute to test continuous

- Temperature = 24.5^{0}C
- (Temperature > 20.0^{0}C) = {true, false}

Where to set the threshold?

Temperature	15^{0}C	18^{0}C	19^{0}C	22^{0}C	24^{0}C	27^{0}C
PlayTennis	No	No	Yes	Yes	Yes	No

Solution: considering all possible T ={ 16.5, 18.5, 20.5, 23, 25.5} (middle)
Use these possible thresholds to calculated Entropy(A feature, Thresholds, Sample sets). The largest Entropy would be provided by the best threshold.

Unknown attribute values

What if some examples have missing values of A?

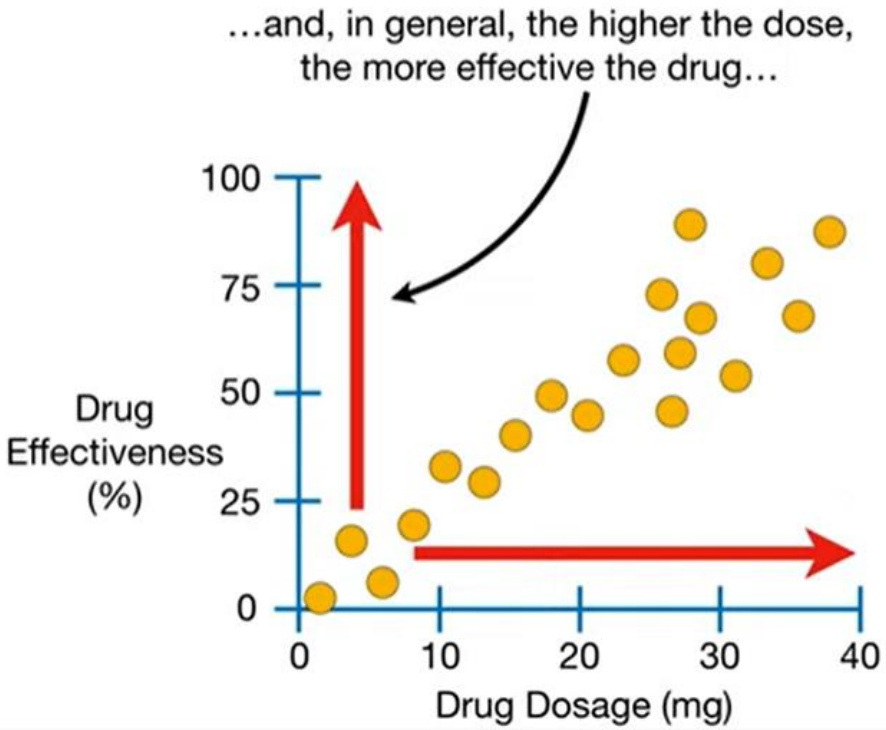
Use training example anyway sort through tree

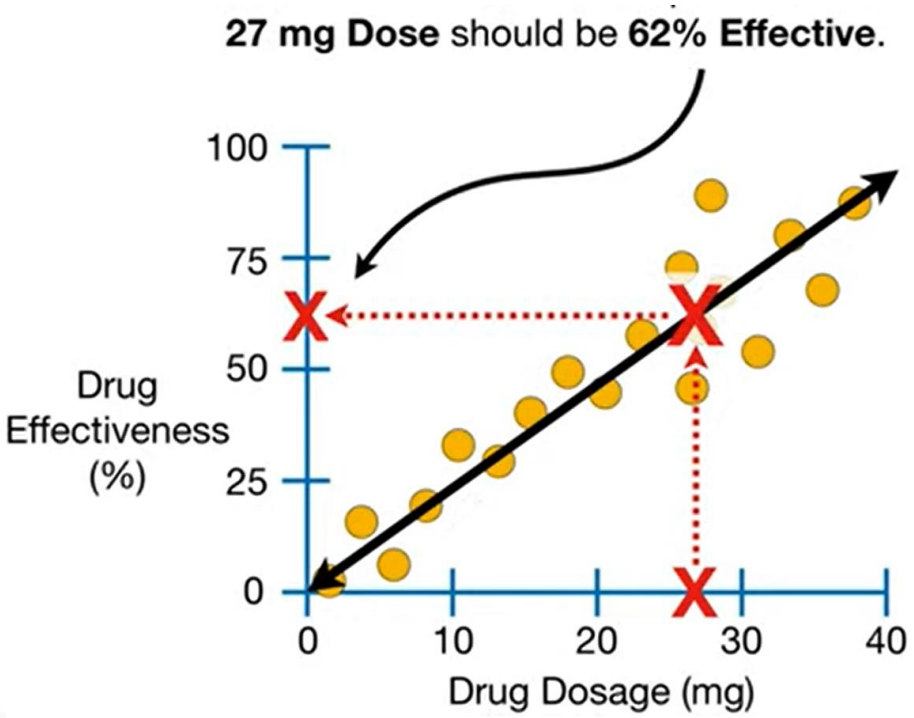
- If node *n* tests A, assign most common value of A among other examples sorted to node *n*.
- Assign most common value of A among other examples with same target value
- Assign probability p_i to each possible value v_i of A
  - Assign fraction p_i of example to each descendant in tree

Classify new examples in the same fashion

Example of Regression Tree
Please link to the web for details:

https://www.youtube.com/watch?v=g9c66TUylZ4





https://www.youtube.com/watch?v=g9c66TUylZ4