

Applied Machine Learning

Course number: W207

Prof. Alexander I. Iliev, Ph.D.

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Evaluation: the key to success

- How predictive is the model we have learned?
- Error on the training data is *not* a good indicator of performance on future data
- Simple solution that can be used if a large amount of (labeled) data is available:
 - Split data into training and test set
- However: (labeled) data is usually limited
 - More sophisticated techniques need to be used

Training and testing I

- Natural performance measure **for classification problems:**
error rate
 - *Success*: instance's class is predicted correctly
 - *Error*: instance's class is predicted incorrectly
 - Error rate: proportion of errors made over the whole set of instances
- *Resubstitution error*: error rate obtained by evaluating model on training data
- Resubstitution error is (hopelessly) **optimistic!**

Training and testing II

- *Test set*: independent instances that have played no part in formation of classifier
 - Assumption: both training data and test data are representative samples of the underlying problem
- Test and training data may differ in nature
 - Example: classifiers built using customer data from two different towns A and B
 - To estimate performance of classifier from town A in completely new town, test it on data from B

Inferring rudimentary rules

- ZeroR rule learner: **simplest classifier of them all**
- It **focuses only on the target** (class) and ignores all other attributes
- So it predicts the **majority** category
- Hence its **predictability** strength **is none**
- Used for **benchmarking / baseline** for other classifiers (can't go lower)
- Hence **huge error** in results
- It is dependent on constructing a *frequency table* for the target and chooses its **most frequent value**

Dealing with numeric attributes

- Simple Examples:

- The Weather Problem: *Weather Data with Some **Nominal** Attributes*

Outlook	Temperature	Humidity	Windy	Play
Sunny	hot	high	false	no
Sunny	hot	high	true	no
Overcast	Play		false	yes
Rainy	Yes	No	false	yes
Rainy			false	yes
Rainy	9	5	true	no
Overcast	cool	normal	true	yes
Sunny	mild	high	false	no
Sunny	cool	normal	false	yes
Rainy	mild	normal	false	yes
Sunny	mild	normal	true	yes
Overcast	mild	high	true	yes
Overcast	hot	normal	false	yes
Rainy	mild	high	true	no

Dealing with numeric attributes

- Simple Examples:

- The Weather Problem:

*Weather Data with Some **Numeric** Attributes*

The diagram shows a confusion matrix for a classification task. The matrix is divided into three main sections: 'Confusion matrix', 'Play', and 'Outcome'. The 'Play' section is further divided into 'Yes' and 'No'. The 'Outcome' section is divided into 'Positive predictive value' and 'Negative predictive value'. The 'ZeroR' row shows the predicted values for 'Yes' and 'No'. The 'Sensitivity' and 'Specificity' are calculated for the 'Yes' and 'No' classes respectively. The 'Accuracy' is calculated as the sum of true positives and true negatives divided by the total number of instances.

Confusion matrix		Play		Outcome	
		Yes	No		
ZeroR	Yes	9	5	Positive predictive value	0.64
	No	0	0	Negative predictive value	0.00
		Sensitivity 1.00	Specificity 0.00	Accuracy = 0.64	

Annotations:

- actual values (points to the 'Play' section)
- true positives (points to the cell with value 9)
- false positives (points to the cell with value 5)
- predicted values (points to the 'ZeroR' row)
- false negatives (points to the cell with value 0 in the 'Yes' row)
- true negatives (points to the cell with value 0 in the 'No' row)

Inferring rudimentary rules

- OneR rule learner: **learns a 1-level decision tree**
 - A set of **rules** that **all test one particular attribute** that has been identified as the one that **yields the lowest classification error**
- Basic version for finding the rule set from a given training set (assumes nominal attributes):
 - For each attribute
 - **Make one branch for each value** of the attribute
(... or one rule for each predictor)
 - To each branch, **assign the most frequent class** value of the instances pertaining to that branch
 - **Error rate**: proportion of instances that do not belong to the majority class of their corresponding branch
 - **Choose** attribute with **lowest error rate**

Pseudo-code for 1R

- 1R method at work

```
For each attribute,  
  For each value of the attribute, make a rule as follows:  
    count how often each class appears  
    find the most frequent class  
    make the rule assign that class to this attribute-value  
  Calculate the error rate of the rules  
Choose the rules with the smallest error rate
```

- 1R's handling of missing values:
a missing value is treated as a separate attribute value
- If the value is numerical it must be converted to categorical before we proceed

Evaluating the weather attributes

Outlook	Temp	Humidity	Windy	Play				
Sunny	Hot	High	False	No	Attribute	Rules	Errors	Total errors
Sunny	Hot	High	True	No	Outlook	Sunny → No	2/5	4/14
Overcast	Hot	High	False	Yes		Overcast → Yes	0/4	
Rainy	Mild	High	False	Yes		Rainy → Yes	2/5	
Rainy	Cool	Normal	False	Yes	Temp	Hot → No	2/4	5/14
Rainy	Cool	Normal	True	No		Mild → Yes	2/6	
Overcast	Cool	Normal	True	Yes		Cool → Yes	1/4	
Sunny	Mild	High	False	No	Humidity	High → No	3/7	4/14
Sunny	Cool	Normal	False	Yes		Normal → Yes	1/7	
Rainy	Mild	Normal	False	Yes		Windy	False → Yes	
Sunny	Mild	Normal	True	Yes	True → No		3/6	
Overcast	Mild	High	True	Yes				
Overcast	Hot	Normal	False	Yes				
Rainy	Mild	High	True	No				

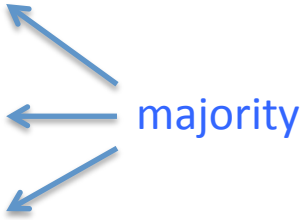
Results with overfitting avoidance

- Let's focus on *Outlook* for a moment:
 - Lets create a *frequency table* for Outlook
 - Then lets construct a confusion matrix for it

Frequency table		Play	
		yes	no
Outlook	Sunny	2	3
	Overcast	4	0
	Rainy	3	2

Rules:

if Outlook = Sunny THEN Play = No
if Outlook = Overcast THEN Play = Yes
if Outlook = Rainy THEN Play = Yes



The word "majority" is written in blue text to the right of the rules. Three blue arrows point from the word "majority" to the "Yes" outcomes in the three rules above: the first arrow points to the "Yes" in the Overcast rule, the second to the "Yes" in the Rainy rule, and the third to the "No" in the Sunny rule.

Results with overfitting avoidance

- Let's focus on *Outlook* for a moment:
 - Lets create a frequency table for Outlook
 - Then lets construct a *confusion matrix* for it

The diagram shows a confusion matrix for the 'Outlook' variable. The matrix is a 2x2 grid with 'Play' as the header and 'Yes' and 'No' as the columns. The rows are 'Yes' and 'No' under the 'OneR' model. The counts are: 7 true positives (Yes/Yes), 2 false positives (No/Yes), 2 false negatives (Yes/No), and 3 true negatives (No/No). Annotations with arrows point to these counts: 'true positives' to 7, 'false positives' to 2 (No/Yes), 'predicted values false negatives' to 2 (Yes/No), and 'true negatives' to 3. A red box highlights the 'Play' header and the 'Yes'/'No' columns. Another red box highlights the 'Yes'/'No' rows. The bottom row shows 'Sensitivity 0.78' and 'Specificity 0.60'. The bottom right cell shows 'Accuracy = 0.71'.

Confusion matrix		Play		Outcome	
		Yes	No		
OneR	Yes	7	2	Positive predictive value	0.78
	No	2	3	Negative predictive value	0.60
		Sensitivity 0.78	Specificity 0.60	Accuracy = 0.71	

Discussion of 1R

- 1R was described in a paper by Holte (1993):

Very Simple Classification Rules Perform Well on Most Commonly Used Datasets,

Robert C. Holte, Computer Science Department, University of Ottawa

- Contains an experimental evaluation on 16 datasets (using *cross-validation* to estimate *classification accuracy* on fresh data)
- Required minimum number of instances in majority class was set to 6 after some experimentation
- *1R's* simple rules *performed not much worse* than much more complex decision trees
- **Lesson:** *simplicity first can pay off* on practical datasets
- Note that 1R does not perform as well on more recent, more sophisticated benchmark datasets

Simple probabilistic modeling

- “Opposite” of 1R: **use all the attributes**
- Two assumptions: Attributes are
 - *equally important*
 - *statistically independent* (given the class value)
- **Independence assumption is almost never correct!**
- But ... this scheme often works surprisingly well in practice
- The scheme is easy to implement in a program and very fast
- It is known as *naïve Bayes*

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Probabilities for weather data

Outlook			Temperature			Humidity			Windy			Play	
Yes		No	Yes		No	Yes		No	Yes		No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								
						Outlook	Temp	Humidity	Windy	Play			
						Sunny	Hot	High	False	No			
						Sunny	Hot	High	True	No			
						Overcast	Hot	High	False	Yes			
						Rainy	Mild	High	False	Yes			
						Rainy	Cool	Normal	False	Yes			
						Rainy	Cool	Normal	True	No			
						Overcast	Cool	Normal	True	Yes			
						Sunny	Mild	High	False	No			
						Sunny	Cool	Normal	False	Yes			
						Rainy	Mild	Normal	False	Yes			
						Sunny	Mild	Normal	True	Yes			
						Overcast	Mild	High	True	Yes			
						Overcast	Hot	Normal	False	Yes			
						Rainy	Mild	High	True	No			

Probabilities for weather data

Outlook			Temperature			Humidity			Windy			Play	
	Yes	No		Yes	No		Yes	No		Yes	No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

- A new day:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

Likelihood of the two classes

$$\text{For "yes"} = 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0053$$

$$\text{For "no"} = 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0206$$

Conversion into a probability by normalization:

$$P(\text{"yes"}) = 0.0053 / (0.0053 + 0.0206) = 0.205$$

$$P(\text{"no"}) = 0.0206 / (0.0053 + 0.0206) = 0.795$$

Can combine probabilities using Bayes's rule

- Famous rule from probability theory due to

Thomas Bayes

Born: 1702 in London, England

Died: 1761 in Tunbridge Wells, Kent, England

- Probability of an **event** H (*class*) given observed **evidence** E (*data, predictor, attribute*):

$$P(H | E) = P(E | H)P(H) / P(E)$$

- A posteriori probability** of the class H given the data E : $P(H | E)$
 - Probability of event after evidence is seen
- A priori probability** of H (*class prior probability*): $P(H)$
 - Probability of event before evidence is seen
- Marginal likelihood** E (*predictor prior probability*): $P(E)$
 - Probability of the attribute before class is seen
- Probability** of the data E given the class H (*likelihood*): $P(E | H)$

Naïve Bayes for classification

- It is a **supervised classification** learning:
 - what is the probability of the class given an instance?
 - Evidence E = instance's non-class attribute values
 - Event H = class value of instance
- **Naïve assumption**: evidence splits into parts (i.e., attributes) that are **conditionally independent**
- This means, given n attributes, we can write Bayes' rule using a product of **per-attribute probabilities**:

$$P(H | E) = P(E_1 | H)P(E_2 | H) \dots P(E_n | H)P(H) / P(E)$$

Weather data example

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

← **Evidence E**

Probability of class “yes” →

$$\begin{aligned} P(\text{yes} \mid E) &= P(\text{Outlook} = \text{Sunny} \mid \text{yes}) \\ &\quad P(\text{Temperature} = \text{Cool} \mid \text{yes}) \\ &\quad P(\text{Humidity} = \text{High} \mid \text{yes}) \\ &\quad P(\text{Windy} = \text{True} \mid \text{yes}) \\ &\quad P(\text{yes}) / P(E) \\ &= \frac{2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14}{P(E)} \end{aligned}$$

Probabilities for weather data

Outlook			Temperature			Humidity			Windy			Play	
	Yes	No		Yes	No		Yes	No		Yes	No	Yes	No
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

$$P(E | H) = P(\text{Sunny} | \text{Yes}) = 2 / 9 = 0.22$$

Likelihood matrix		Play		Outcome
		Yes	No	
Outlook	Sunny	2/9	3/5	5/14
	Overcast	4/9	0/5	4/14
	Rainy	3/9	2/5	5/14
		9/14	5/14	

$$P(E) = P(\text{Sunny}) = 5 / 14 = 0.36$$

$$P(H) = P(\text{Yes}) = 9 / 14 = 0.64$$

Posterior probability:

$$P(H | E) = P(\text{Yes} | \text{Sunny})$$

$$P(H | E) = P(\text{Sunny} | \text{Yes}) * P(\text{Yes}) / P(\text{Sunny}) = 0.22 * 0.64 / 0.36 = 0.39$$

The “zero-frequency problem”

- What if an attribute value does not occur with every class value? (e.g., “Humidity = high” for class “yes”)
 - **Probability will be zero:** $P(\text{Humidity} = \text{High} \mid \text{yes}) = 0$
 - *A posteriori* probability will also be zero: $P(\text{yes} \mid E) = 0$
(Regardless of how likely the other values are!)
- **Remedy:** **add 1** to the count for every attribute value-class combination (*Laplace estimator*)
- **Result:** **probabilities will never be zero**
- Additional advantage: stabilizes probability estimates computed from small samples of data

Modified probability estimates

- In some cases **adding a constant** different from 1 might be more appropriate
- Example: attribute *outlook* for class *yes*

$$\frac{2 + \mu/3}{9 + \mu}$$

Sunny

$$\frac{4 + \mu/3}{9 + \mu}$$

Overcast

$$\frac{3 + \mu/3}{9 + \mu}$$

Rainy

- Weights don't need to be equal (but they must sum to 1)

$$\frac{2 + \mu p_1}{9 + \mu}$$

$$\frac{4 + \mu p_2}{9 + \mu}$$

$$\frac{3 + \mu p_3}{9 + \mu}$$

Missing values

- Training: **instance is not included** in frequency count for attribute value-class combination
- Classification: **attribute will be omitted** from calculation
- Example:

Outlook	Temp.	Humidity	Windy	Play
?	Cool	High	True	?

Likelihood of "yes" = $3/9 \times 3/9 \times 3/9 \times 9/14 = 0.0238$

Likelihood of "no" = $1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0343$

$P(\text{"yes"}) = 0.0238 / (0.0238 + 0.0343) = 41\%$

$P(\text{"no"}) = 0.0343 / (0.0238 + 0.0343) = 59\%$

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Applied Machine Learning

Weather Data				
Day	Outlook	Humidity	Wind	Play
1	Sunny	High	Weak	No
2	Sunny	High	Strong	No
3	Overcast	High	Weak	Yes
4	Rain	High	Weak	Yes
5	Rain	Normal	Weak	Yes
6	Rain	Normal	Strong	No
7	Overcast	Normal	Strong	Yes
8	Sunny	High	Weak	No
9	Sunny	Normal	Weak	Yes
10	Rain	Normal	Weak	Yes
11	Sunny	Normal	Strong	Yes
12	Overcast	High	Strong	Yes
13	Overcast	Normal	Weak	Yes
14	Rain	High	Strong	No
New Day				
15	Rain	High	Weak	?

Applied Machine Learning

Lecture 4 ...

- *OneR, ZeroR, Frequency table, Confusion matrix*
- *Laplace smoothing*
- *Bias, Variance*
- *Decision trees*
- *Entropy, Information gain*

Criterion for attribute selection

- Which is the best attribute?
 - Want to get the **smallest tree**
 - **Heuristic**: choose the attribute that produces the “purest” nodes
- Popular selection criteria: **information gain**
 - Information gain **increases with** the average **purity** of the subsets
- Strategy: amongst attributes available **for splitting, choose attribute that gives greatest information gain**
- Information gain requires measure of **impurity**
- **Impurity** measure that it **uses** is the **entropy** of the class distribution, which is a measure from information theory

What is Entropy?

- *Various definitions same meaning:*
 - *Entropy*: lack of order or predictability; gradual decline into disorder
 - *Entropy*: (in information theory) a logarithmic measure of the rate of transfer of information in a particular message or language
 - *Entropy* may be understood as a measure of disorder within a macroscopic system
 - *Entropy* is the measure of the level of disorder in a closed but changing system, a system in which energy can only be transferred in one direction from an ordered state to a disordered state
 - The higher the entropy, the higher the disorder and the system's energy to do useful work is lower

Computing information

- We have a probability distribution: the class distribution in a subset of instances
- The **expected information** required to determine an outcome (i.e., class value), **is the distribution's *entropy***
- Formula for computing the entropy:
$$\text{Entropy}(p_1, p_2, \dots, p_n) = -p_1 \log p_1 - p_2 \log p_2 \dots - p_n \log p_n$$
- Using **base-2 logarithms**, entropy gives the information required in expected ***bits***
- ***Entropy is maximal when all classes are equally likely and minimal when one of the classes has probability 1***

Example: attribute *Outlook*

- *Outlook = Sunny* :

$$\text{Info}([2, 3]) = 0.971 \text{ bits}$$

$$\text{Entropy}(S_{\text{sunny}}) = -\frac{2}{5} \log_2 \left(\frac{2}{5} \right) - \frac{3}{5} \log_2 \left(\frac{3}{5} \right)$$

- *Outlook = Overcast* :

$$\text{Info}([4, 0]) = 0.0 \text{ bits}$$

$$\text{Entropy}(S_{\text{overcast}}) = -\frac{4}{4} \log_2 \left(\frac{4}{4} \right) - 0 \log_2 (0)$$

- *Outlook = Rainy* :

$$\text{Info}([3, 2]) = 0.971 \text{ bits}$$

$$\text{Entropy}(S_{\text{rain}}) = -\frac{3}{5} \log_2 \left(\frac{3}{5} \right) - \frac{2}{5} \log_2 \left(\frac{2}{5} \right)$$

- Expected information for attribute:

$$\begin{aligned} \text{Info}([2, 3], [4, 0], [3, 2]) &= (5/14) \times 0.971 + (4/14) \times 0 + (5/14) \times 0.971 \\ &= 0.693 \text{ bits} \end{aligned}$$

Computing information gain

- Information gain: information before splitting – information after splitting

$$\begin{aligned}\text{Gain (Outlook)} &= \text{Info}([9,5]) - \text{info}([2,3],[4,0],[3,2]) \\ &= 0.940 - 0.693 \\ &= 0.247 \text{ bits}\end{aligned}$$

- Information gain for attributes from weather data:

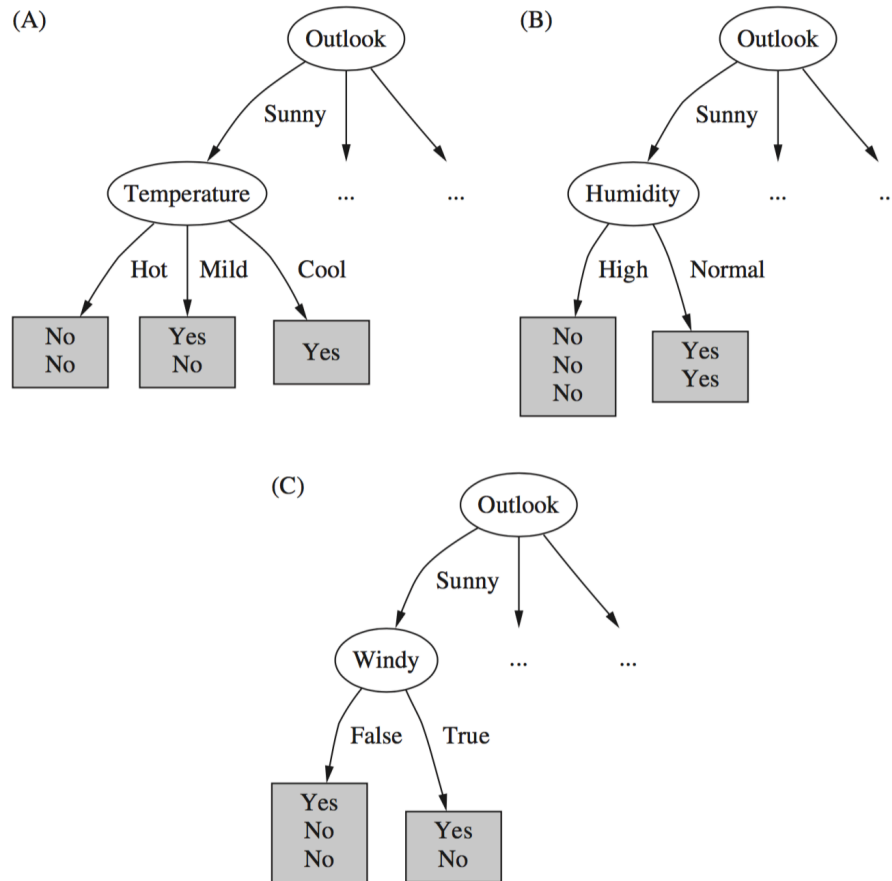
$$\text{Gain (Outlook)} = 0.247 \text{ bits}$$

$$\text{Gain (Temperature)} = 0.029 \text{ bits}$$

$$\text{Gain (Humidity)} = 0.152 \text{ bits}$$

$$\text{Gain (Windy)} = 0.048 \text{ bits}$$

Continuing to split

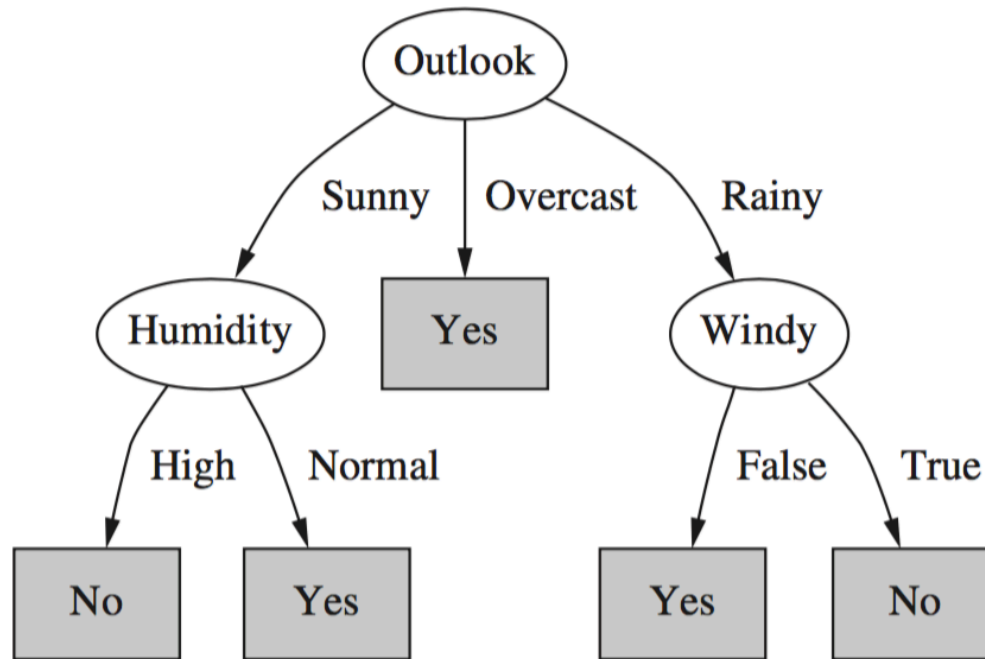


Gain (*Temperature*) = 0.571 bits

Gain (*Humidity*) = 0.971 bits

Gain (*Windy*) = 0.020 bits

Final decision tree



- Note: not all leaves need to be pure; sometimes identical instances have different classes
 - \Rightarrow Splitting stops when data cannot be split any further

Discussion

- Top-down induction of **decision trees**: **ID3**, algorithm developed by Ross Quinlan
 - Gain ratio just one modification of this basic algorithm
 - **C4.5** tree learner deals with numeric attributes, missing values, noisy data
- Similar approach: **CART** tree learner
 - Uses Gini index rather than entropy to measure impurity
- There are many other attribute selection criteria!
(But little difference in accuracy of result)