



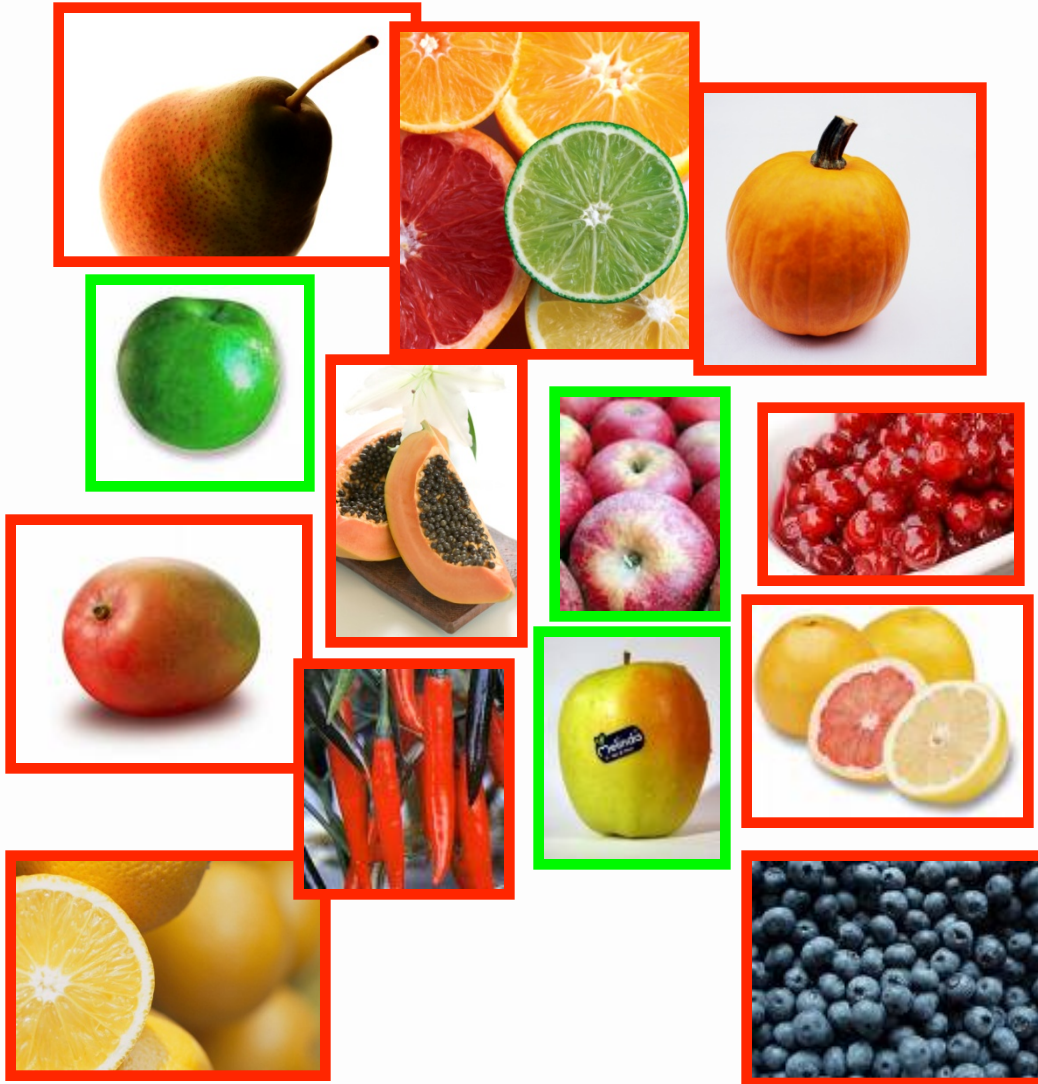
Classification: Other Approaches

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

- Instance-Based Learning
- Naïve Bayes Classifiers
- Logistic Regression
- Bayesian Belief Networks

What is Instance-Based Learning?

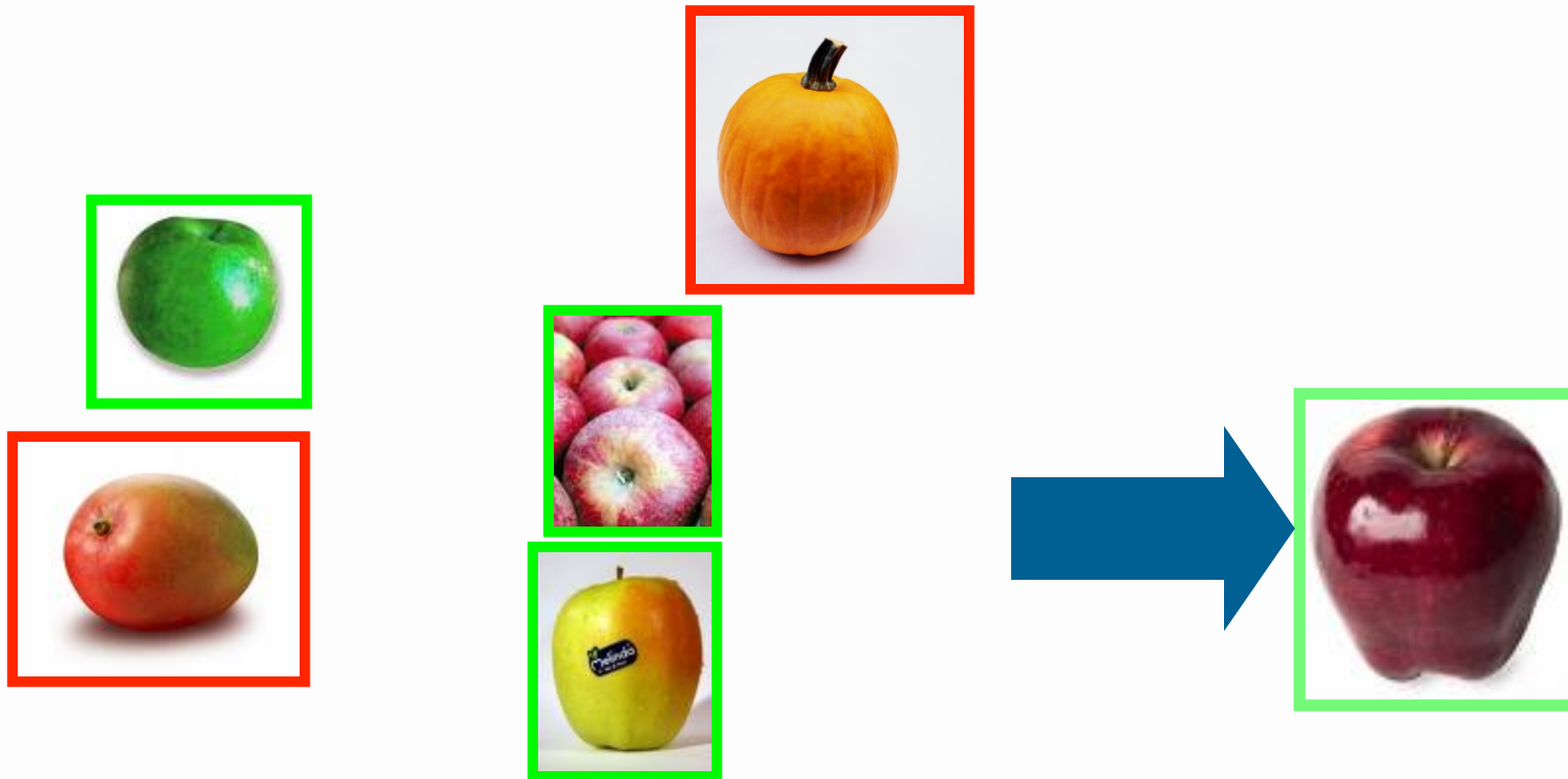
At the Core of Classification Problems! Apples Again!



is this an apple?

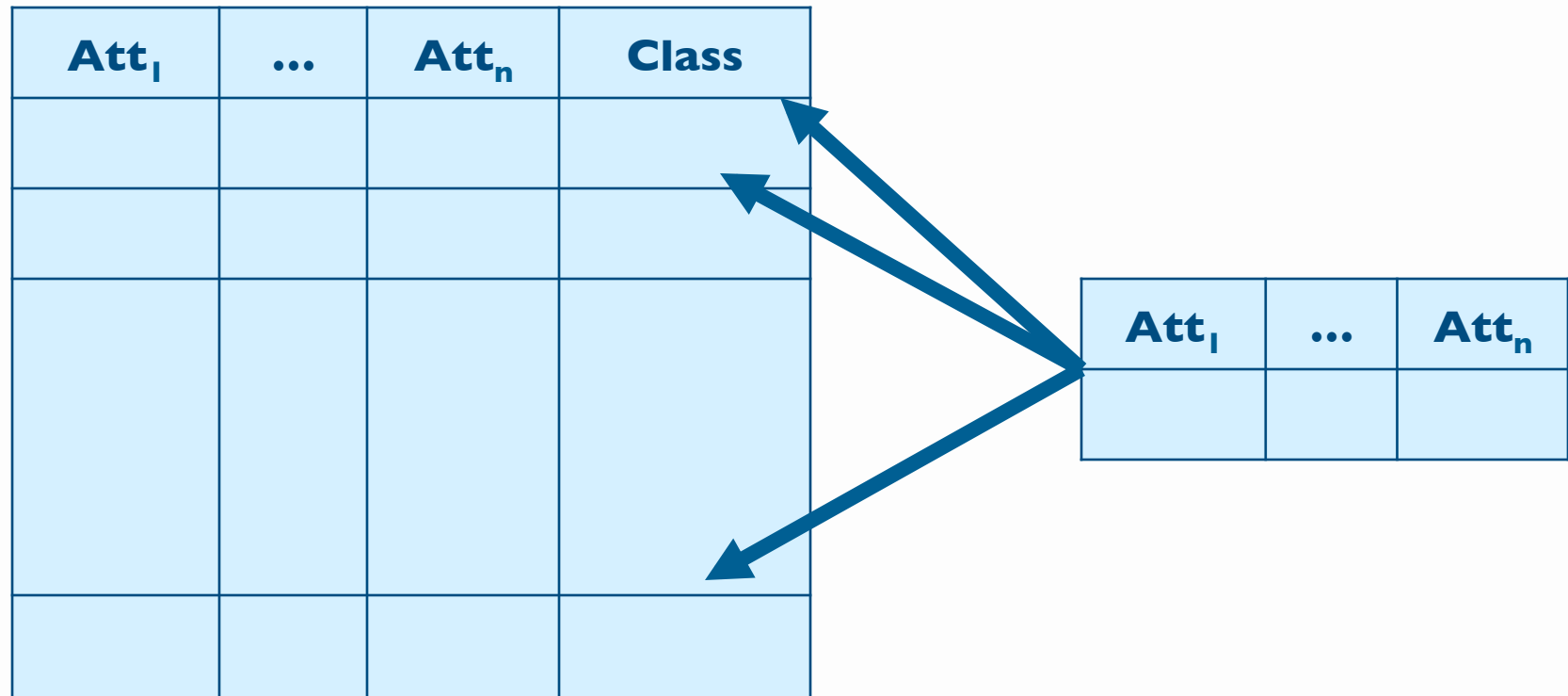


- To decide the label for an unseen example, consider the **k examples from the training data** that are more similar to the unknown one
- Classify the unknown example using the most frequent class

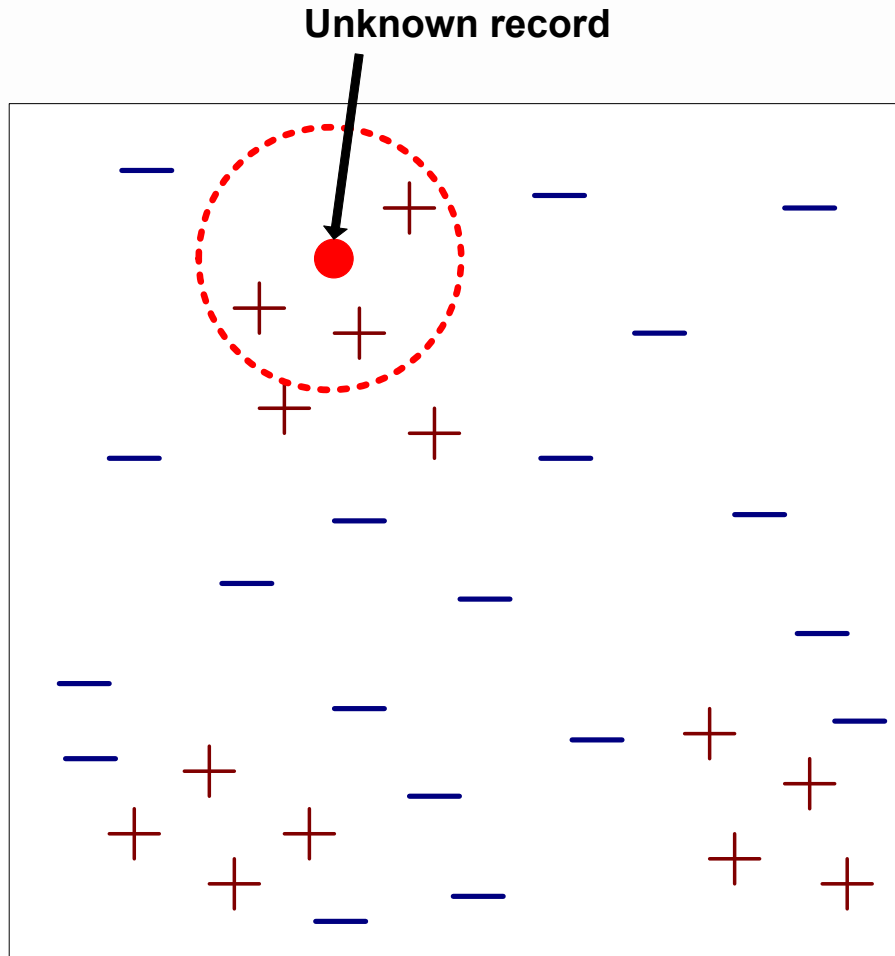


- If $k=5$, these 5 fruits are the most similar ones to the unclassified example
- Since, the majority of the fruits are apples, we decide that the unknown fruit is an apple

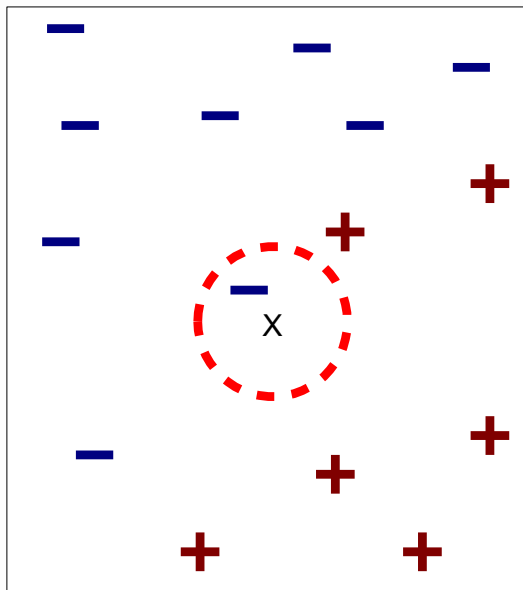
- Store the training records only, no model is computed
- Use the training records to predict an unknown class label



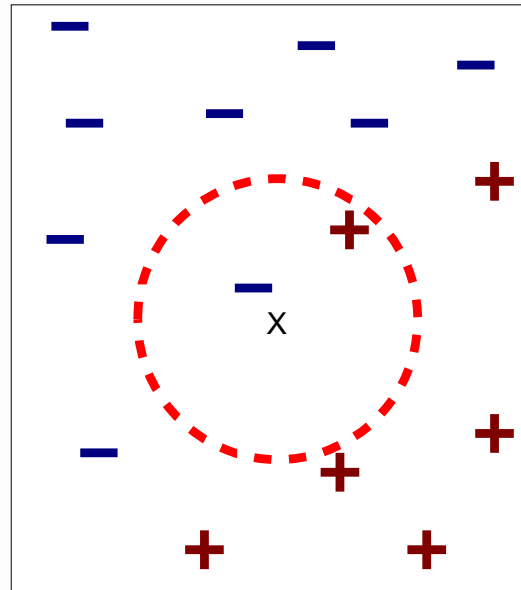
- They are the simplest form of learning
- The training dataset is searched for the instances that are more similar to the unlabeled instance
- The training dataset is the model itself, it is the knowledge
- The similarity function defines what's "learned"
- They implement a "lazy evaluation" (or lazy learning) scheme, nothing happens until a new unlabeled instance must be classified
- Known methods include "Rote Learning", "Case Base Reasoning", "k-Nearest Neighbor"



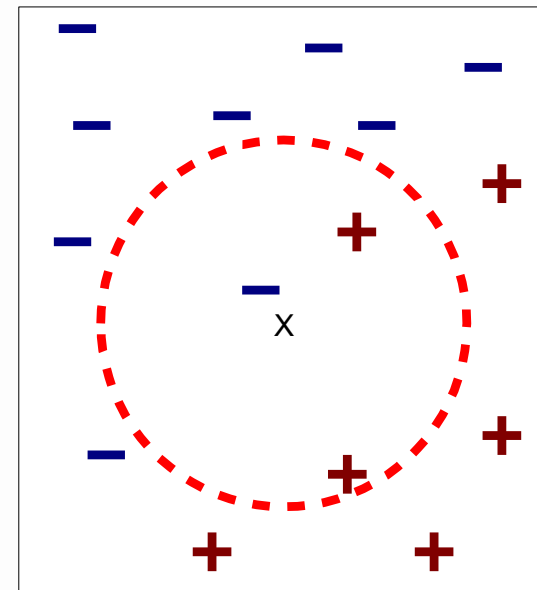
- Three elements
 - The training dataset
 - Similarity function (or distance metric)
 - The value of k , the number of nearest neighbors to retrieve
- Classification
 - Compute distance to other training records
 - Identify the k nearest neighbors
 - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)



(a) 1-nearest neighbor



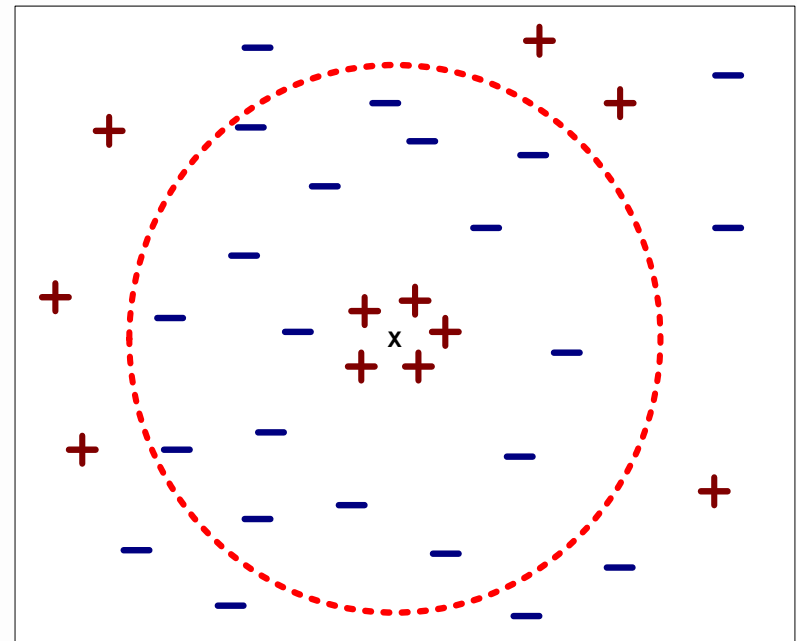
(b) 2-nearest neighbor



(c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

- If k is too small, sensitive to noise points
- If k is too large, neighborhood may include examples from other classes



- Euclidian distance is the typical function used to compute the similarity between two examples

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- To determine the class from nearest neighbor list
 - Take the majority vote of class labels among the k neighbors
 - Or weight the vote according to distance (e.g., $w = 1/d^2$)
- Another popular metric is city-block (Manhattan) metric, distance is the sum of absolute differences

- Different attributes are measured on different scales
- Attributes might need to be normalized:

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

$$a_i = \frac{v_i - Avg(v_i)}{StDev(v_i)}$$

v_i is the actual value of attribute i

a_i is the normalized value

- For nominal attributes, the distance either 0 or 1
- Missing values are usually assumed to be maximally distant (given normalized attributes)

- K-Nearest Neighbor is often very accurate but slow, since simple version scans entire training data to derive a prediction
- Assumes all attributes are equally important, thus may need attribute selection or weights
- Statisticians have used k-NN since early 1950s, If $n \rightarrow \infty$ and $k/n \rightarrow 0$, error approaches minimum

Naïve Bayes

- Conditional Probability:

$$P(C|A) = \frac{P(A \wedge C)}{P(A)}$$

$$P(A|C) = \frac{P(A \wedge C)}{P(C)}$$

- Bayes theorem,

$$P(C|A) = \frac{P(A|C)P(C)}{P(A)}$$

- A priori probability of C, $P(C)$, is the probability of event before evidence is seen
- A posteriori probability of C, $P(C|A)$, is the probability of event after evidence is seen

- What's the probability of the class given an instance?
- Evidence E = instance, represented as a tuple of attributes $\langle e_1, \dots, e_n \rangle$
- Event H = class value for instance
- We are looking for the class value with the highest probability for E

$$class = \arg \max_{h \in H} P(H|E)$$

- I.e., we are looking for the hypothesis that has the highest probability to explain the evidence E

- Given the hypothesis H and the example E described by n attributes, Bayes Theorem says that

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

- Naïve assumption: attributes are statistically independent
- Evidence splits into parts that are independent

$$P(H|E) = \frac{P(e_1|H) \cdots P(e_n|H)P(H)}{P(E)}$$

- Training: the class probability $P(H)$ and the conditional probability $P(e_i|H)$ for each attribute value e_i and each class value H
- Testing: given an instance E , the class is computed as

$$\begin{aligned} \text{class} &= \arg \max_{h \in H} P(H|E) \\ &= \arg \max_{h \in H} \frac{P(e_1|H) \cdots P(e_n|H)P(H)}{P(E)} \\ &= \arg \max_{h \in H} P(e_1|H) \cdots P(e_n|H)P(H) \end{aligned}$$

- It is the “opposite” of OneRule as it uses all the attributes
- Two assumptions
 - Attributes are equally important
 - Attribute are statistically independent
- Statistically independent means that knowing the value of one attribute says nothing about the value of another (if the class is known)
- Independence assumption is almost never correct! But the scheme works well in practice

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

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	Cool	High	True	?

What is the value of “play”?

- Conversion into a probability by normalization:

Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

$$\begin{aligned}
 P(\text{yes}|E) &= P(\text{Sunny}|\text{yes})P(\text{Cool}|\text{yes})P(\text{High}|\text{yes})P(\text{True}|\text{yes}) \\
 &= \frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14} \\
 &= 0.0053
 \end{aligned}$$

$$\begin{aligned}
 P(\text{no}|E) &= P(\text{Sunny}|\text{no})P(\text{Cool}|\text{no})P(\text{High}|\text{no})P(\text{True}|\text{no}) \\
 &= \frac{3}{5} \times \frac{1}{5} \times \frac{4}{5} \times \frac{3}{5} \times \frac{5}{14} \\
 &= 0.0206
 \end{aligned}$$

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

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

- What if an attribute value does not occur with every class value? (for instance, “Humidity = high” for class “yes”)
- The corresponding probability will be zero,

$$P(\textit{Humidity} = \textit{high}|\textit{yes}) = 0$$

- A posteriori probability will also be zero!
(No matter how likely the other values are!)

$$P(\textit{yes}|\langle \dots, \textit{Humidity} = \textit{high}, \dots \rangle) = 0$$

- The typical remedy is to add 1 to the count for every attribute value-class combination (Laplace estimator)
- The probabilities will never be zero! (also: stabilizes probability estimates)

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

Outlook	Temperature	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” = $1/5 \times 4/5 \times 3/5 \times 5/14 = 0.0343$

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

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

- We assume that the attributes have a normal or Gaussian probability distribution (given the class)
- The probability density function for the normal distribution is defined by two parameters, the mean and the standard deviation
- Sample mean,

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i$$

- Standard deviation,

$$\sigma = \frac{1}{n-1} \sum_{i=1}^n (x_i - \mu)^2$$

- Then the density function $f(x)$ is

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Statistics for Weather data with numeric temperature

	Outlook		Temperature		Humidity			Windy		Play	
	Yes	No	Yes	No	Yes	No		Yes	No	Yes	No
Sunny	2	3	64, 68,	65, 71,	65, 70,	70, 85,	False	6	2	9	5
Overcast	4	0	69, 70,	72, 80,	70, 75,	90, 91,	True	3	3		
Rainy	3	2	72, ...	85, ...	80, ...	95, ...					
Sunny	2/9	3/5	$\mu = 73$	$\mu = 75$	$\mu = 79$	$\mu = 86$	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	$\sigma = 6.2$	$\sigma = 7.9$	$\sigma = 10.2$	$\sigma = 9.7$	True	3/9	3/5		
Rainy	3/9	2/5									

$$f(\text{temperature} = 66 | \text{yes}) = \frac{1}{\sqrt{2\pi}6.2} e^{-\frac{(66-73)^2}{2 \times 6.2^2}} = 0.0340$$

- A new day,

Outlook	Temperature	Humidity	Windy	Play
Sunny	66	90	true	?

- Missing values during training are not included in calculation of mean and standard deviation

$$\text{Likelihood of "yes"} = 2/9 \times 0.0340 \times 0.0221 \times 3/9 \times 9/14 = 0.000036$$

$$\text{Likelihood of "no"} = 3/5 \times 0.0291 \times 0.0380 \times 3/5 \times 5/14 = 0.000136$$

$$P(\text{"yes"}) = 0.000036 / (0.000036 + 0.000136) = 20.9\%$$

$$P(\text{"no"}) = 0.000136 / (0.000036 + 0.000136) = 79.1\%$$

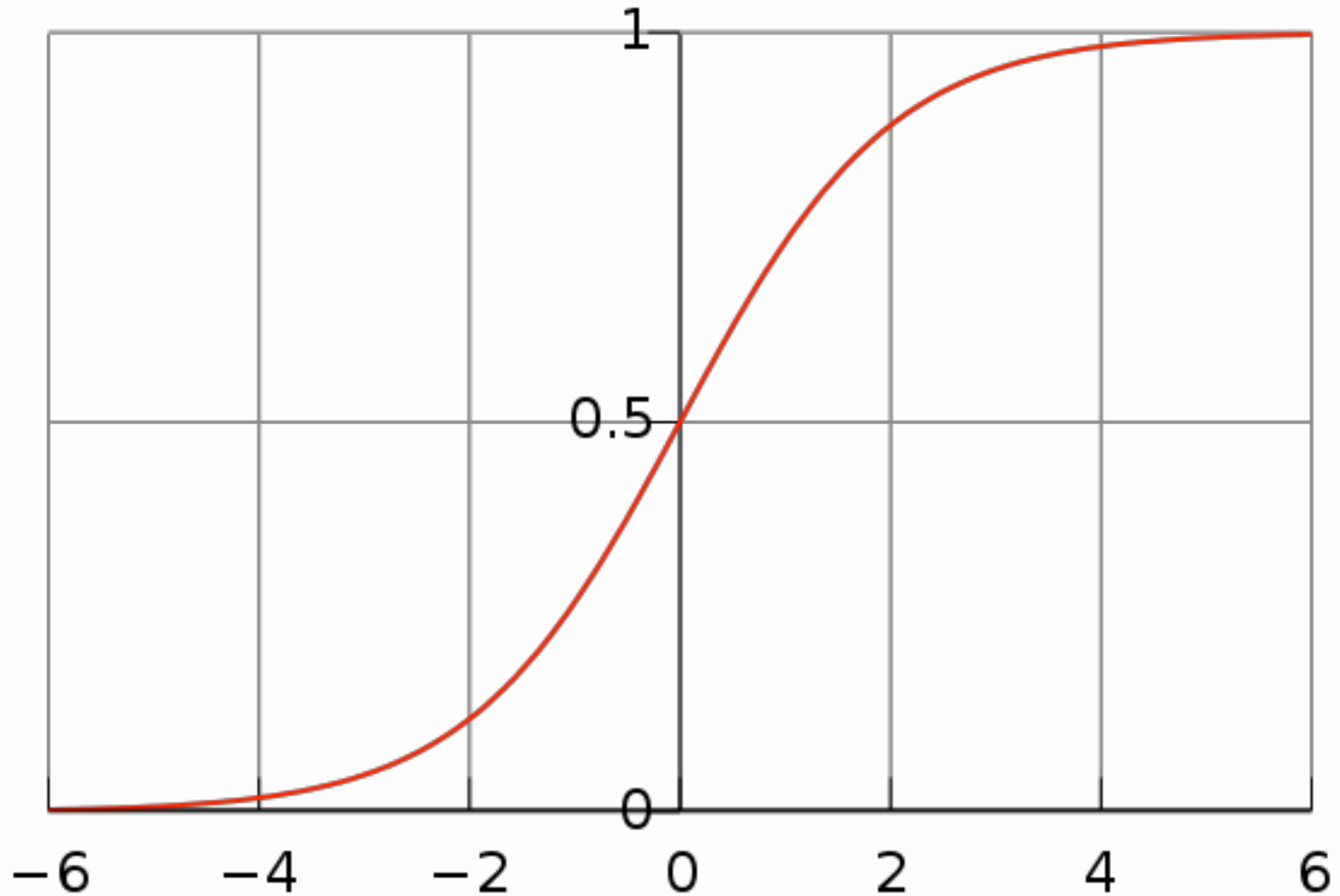
- Naïve Bayes works surprisingly well, even if independence assumption is clearly violated
- Why? Because classification doesn't require accurate probability estimates as long as maximum probability is assigned to correct class
- However, adding too many redundant attributes will cause problems (e.g. identical attributes)
- Also, many numeric attributes are not normally distributed

Logistic Regression

- Well-known and widely used statistical classification method
- Typically applied to two classes problems, but it can be easily extended to the case of multiple classes
- Instead of applying regression to evaluate $P(H|E)$, apply it to evaluate the logit function computed as

$$P(H|E) = \frac{1}{1 + \exp -w_0 - w_1 e_1 \cdots - w_n e_n}$$

where w_i are the weights and e_i are the attribute value for example E



- Logistic regression assumes the following parametric model

$$P(Y = 1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

$$P(Y = 0|X) = \frac{\exp(w_0 + \sum_{i=1}^n w_i X_i)}{1 + \exp(w_0 + \sum_{i=1}^n w_i X_i)}$$

- To classify an example X , we select the class Y that maximizes the probability $P(Y=y_k | X)$ or check

$$1 < \frac{P(Y = 0|X)}{P(Y = 1|X)} \quad \text{or} \quad 1 < \exp(w_0 + \sum_{i=1}^n w_i X_i)$$

- By taking the natural logarithm of both sides, we obtain a linear classification rule that assign the label $Y=0$ if

$$0 < w_0 + \sum_{i=1}^n w_i X_i$$

- $Y=1$ otherwise

- Linear regression usually search for the weights which minimize the squared error on the training data
- Logistic regression search for the weights that maximize the log-likelihood or,

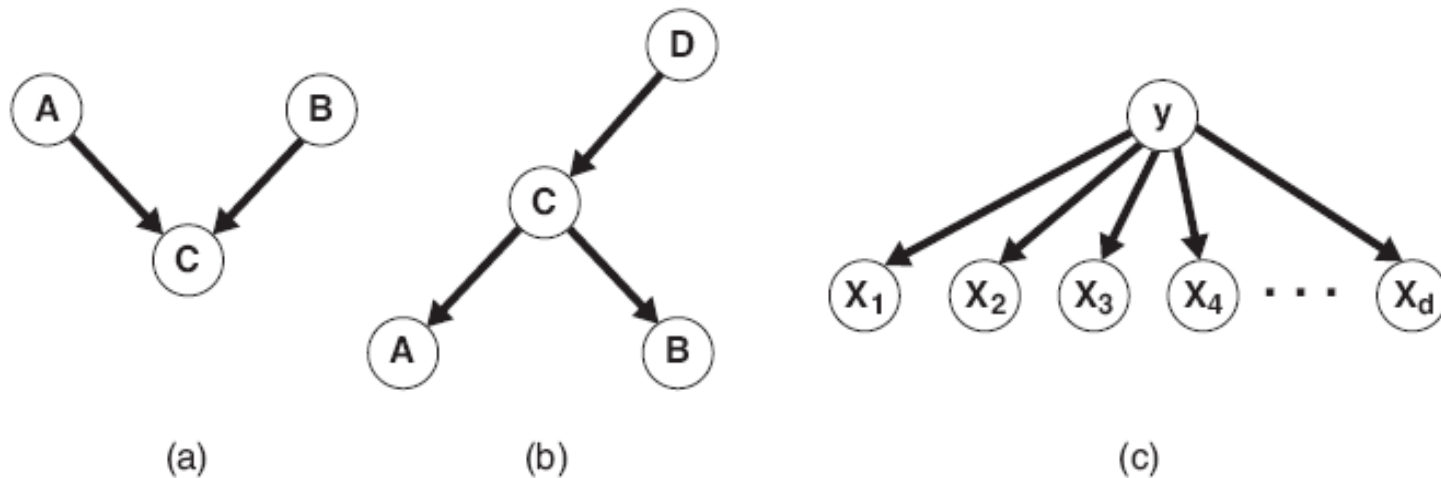
$$\sum_{i=1}^k [(1 - h^{(i)}) \log(1 - P(1|e_1^{(i)}, \dots, e_n^{(i)})) + h^{(i)} \log P(1|e_1^{(i)}, \dots, e_n^{(i)})]$$

- Weights can be simply computed by solving a sequence of weighted least square regression problems until the log-likelihood converges to the max

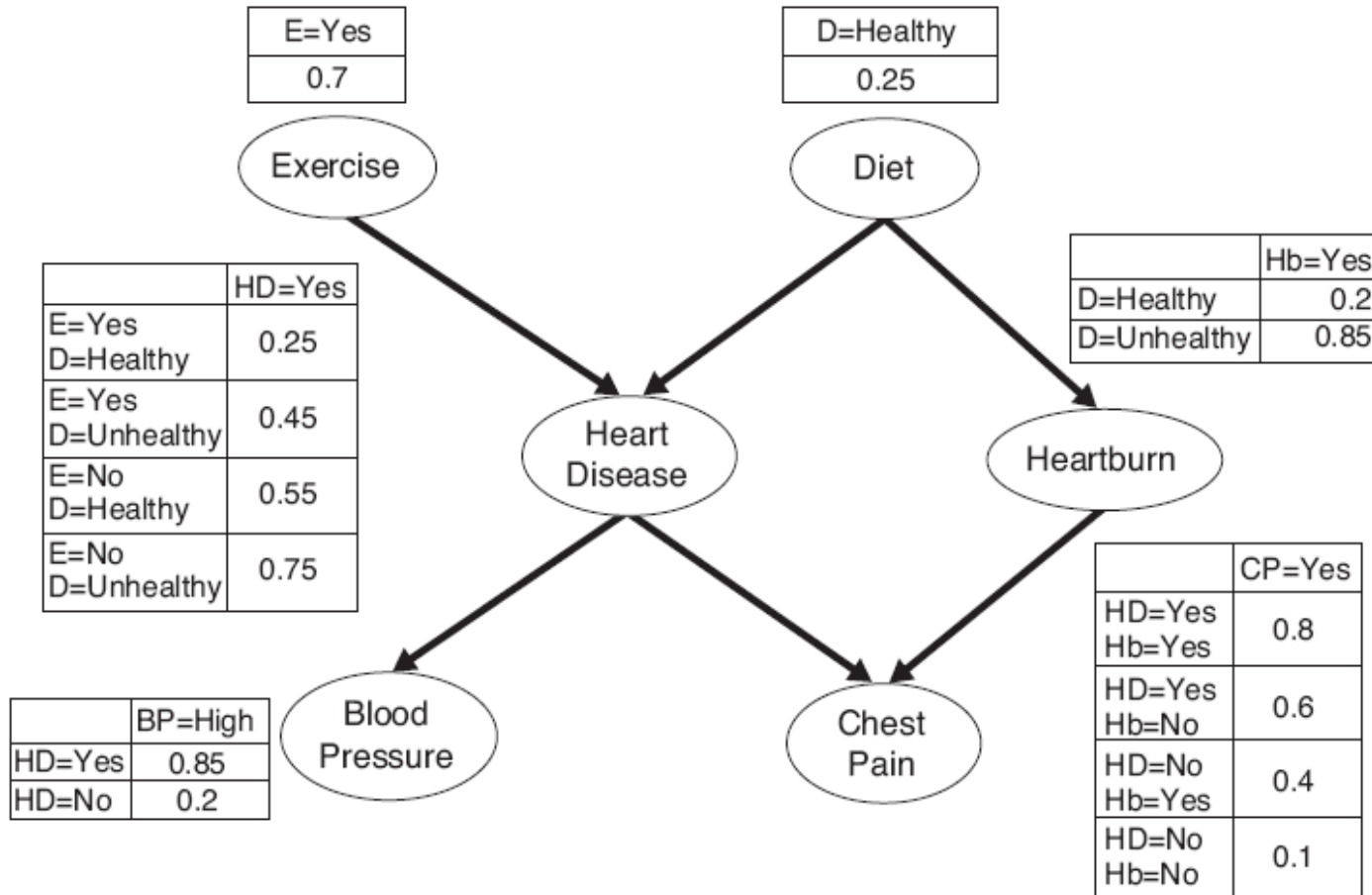
Bayesian Belief Networks

- The conditional independence assumption,
 - makes computation possible
 - yields optimal classifiers when satisfied
 - but is seldom satisfied in practice, as attributes (variables) are often correlated
- Bayesian Belief Networks (BBN) allows us to specify which pair of attributes are conditionally independent
- They provide a graphical representation of probabilistic relationships among a set of random variables

- Describe the probability distribution governing a set of variables by specifying
 - Conditional independence assumptions that apply on subsets of the variables
 - A set of conditional probabilities
- Two key elements
 - A direct acyclic graph, encoding the dependence relationships among variables
 - A probability table associating each node to its immediate parents node



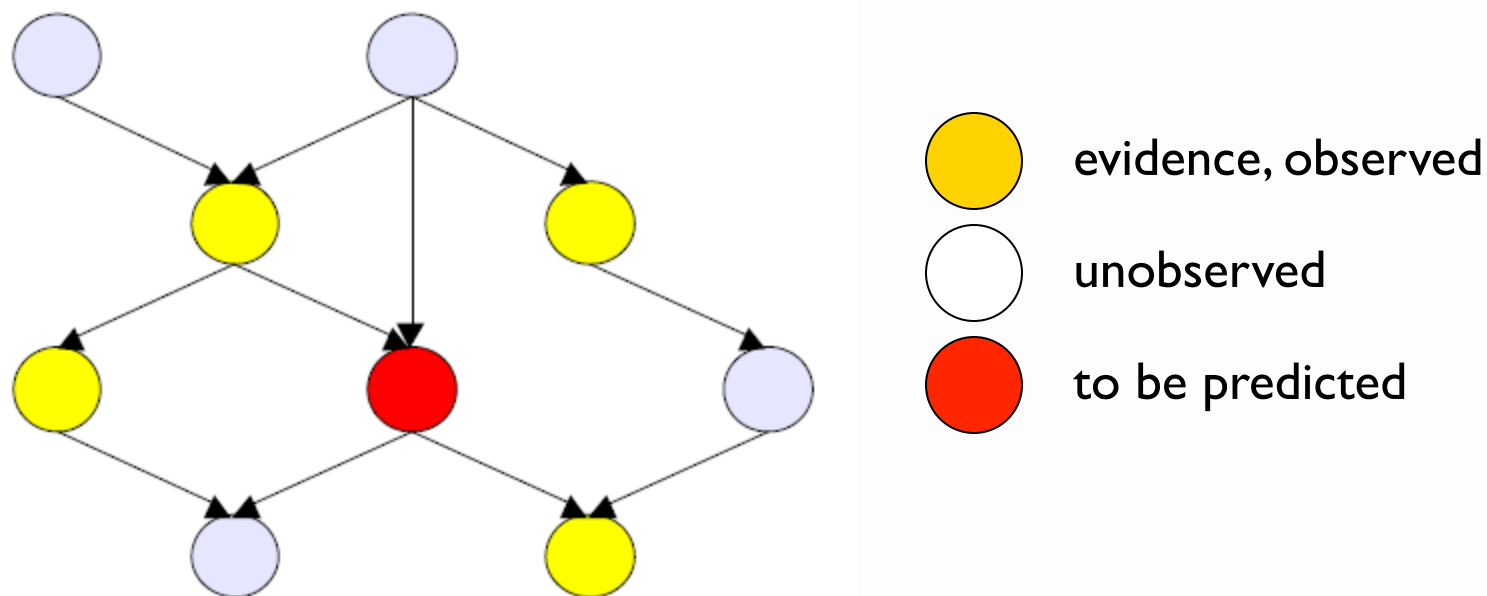
- Variable A and B are independent, but each one of them has an influence on the variable A
- A is conditionally independent of both B and D, given C
- The configuration of the typical naïve Bayes classifier



- The network topology imposes conditions regarding the variable conditional independence
- Each node is associated with a probability table
 - If a node X does not have any parents, then the table contains only the prior probability $P(X)$
 - If a node X has only one any parent Y , then the table contains only the conditional probability $P(X|Y)$
 - If a node X has multiple parents, Y_1, \dots, Y_k the the table contains the conditional probability $P(X|Y_1 \dots Y_k)$

Algorithm 5.3 Algorithm for generating the topology of a Bayesian network.

- 1: Let $T = (X_1, X_2, \dots, X_d)$ denote a total order of the variables.
 - 2: **for** $j = 1$ to d **do**
 - 3: Let $X_{T(j)}$ denote the j^{th} highest order variable in T .
 - 4: Let $\pi(X_{T(j)}) = \{X_{T(1)}, X_{T(2)}, \dots, X_{T(j-1)}\}$ denote the set of variables preceding $X_{T(j)}$.
 - 5: Remove the variables from $\pi(X_{T(j)})$ that do not affect X_j (using prior knowledge).
 - 6: Create an arc between $X_{T(j)}$ and the remaining variables in $\pi(X_{T(j)})$.
 - 7: **end for**
-



- In general the inference is NP-complete but there are approximating methods, e.g. Monte-Carlo

- Bayesian belief network allows a subset of the variables conditionally independent
- A graphical model of causal relationships
- Several cases of learning Bayesian belief networks
 - Given both network structure and all the variables is easy
 - Given network structure but only some variables
 - When the network structure is not known in advance

- BBN provides an approach for capturing prior knowledge of a particular domain using a graphical model
- Building the network is time consuming, but adding variables to a network is straightforward
- They can encode causal dependencies among variables
- They are well suited to dealing with incomplete data
- They are robust to overfitting