DATABASE SYSTEMS GROUP

Ludwig-Maximilians-Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

LMU

Lecture notes

Knowledge Discovery in Databases

Summer Semester 2012

Lecture 5: Classification II

Lecture: Dr. Eirini Ntoutsi
Tutorials: Erich Schubert

http://www.dbs.ifi.lmu.de/cms/Knowledge_Discovery_in_Databases_I_(KDD_I)

Sources

- Previous KDD I lectures on LMU (Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Jörg Sander, Matthias Schubert, Arthur Zimek)
- Jiawei Han, Micheline Kamber and Jian Pei, *Data Mining: Concepts and Techniques, 3rd ed.,* Morgan Kaufmann, 2011.
- Margaret Dunham, Data Mining, *Introductory and Advanced Topics*, Prentice Hall, 2002.
- Tan P.-N., Steinbach M., Kumar V., *Introduction to Data Mining*, Addison-Wesley, 2006
- D. Jurafsky and C. Manning, Natural Language Processing course, https://www.coursera.org/course/nlp

Outline

- Introduction
- Bayesian classifiers
- Lazy vs Eager learners
- k-Nearest Neighbors (or learning from your neighbors)
- Artificial neural networks
- Things you should know
- Homework/tutorial

Bayesian classifiers

- A probabilistic framework for solving classification problems
- Predict class membership probabilities for an instance
- The class of an instance is the most likely class for the instance (Maximum Likelihood classification)
- Based on Bayes' rule
- Bayesian classifiers
  - Naïve Bayes classifiers
    - Assume class-conditional independence among attributes
  - Bayesian Belief networks
    - Graphical models
    - Model dependencies among attributes
- Lately used a lot for: Text classification, Sentiment analysis

Bayes' theorem

- The probability of an event C given an observation A:

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

a posteriori probability

likelihood

prior probability

- e.g., given:
  - A doctor knows that meningitis causes stiff neck 50% of the time
  - Prior probability of any patient having meningitis is 1/50,000
  - Prior probability of any patient having stiff neck is P(S)=1/20

- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Bayesian classifiers I

- Let $C=\{c_1, c_2, \ldots, c_k\}$ be the class attribute.
- Let $X=(A_1, A_2, A_3, \ldots A_n)$ be a n-dimensional instance.
- Classification problem: What is the probability of a class value c in C given an instance observation X?
  - The event C to be predicted is the class value of the instance
  - The observation is the instance values X

  - $P(c_1|X)$
  - $P(c_2|X)$
  - …
  - $P(c_k|X)$
- The class of the instance is the class value with the higher probability: $\text{argmax}_c(P(c|X))$

Bayesian classifiers II

- Consider each attribute and class label as random variables
- Given an instance X with attributes ($A_1A_2…A_n$)
  - Goal is to predict class label c in C
  - Specifically, we want to find the value c of C that maximizes P(c|X)

$$c = \arg\max_{c \in C} P(c \mid X)$$

Bayes' rule

$$c = \arg\max_{c \in C} \frac{P(X \mid c)P(c)}{P(X)}$$

$$c = \arg\max_{c \in C} P(X \mid c)P(c)$$

max a posteriori = the most likely class

likelihood

prior

Bayesian classifiers III

- How can we estimate: $c = \arg\max_{c \in C} P(X \mid c)P(c)$?
- Class prior P(c):
  - How often c occurs?
  - Just count the relative frequencies in the training set
- Instance likelihood P(X|c):
  - What is the probability of an instance X given the class c?
  - but X=($A_1A_2…A_n$), so, P(X|c)=P($A_1A_2…A_n$ |c)
  - i.e., the probability of an instance given the class is equal to the probability of a set of features given the class
- So:

$$c = \arg\max_{c \in C} P(A_1 A_2 \ldots A_n \mid c)P(c)$$

Naïve Bayes classifier

How to estimate $P(A_1A_2…A_n \mid c)$?

- Assume independence among attributes A_i when class is given:
  - $P(A_1A_2…A_n \mid C_j) = \prod P(A_i \mid c) = P(A_1 \mid c)P(A_2 \mid c)… P(A_n \mid c)$

Strong conditional independence assumption!!!

  - Can estimate $P(A_i \mid c)$ for all A_i and c in C based on training set
  - New point is classified to:

$$c = \arg\max_{c \in C} P(c) \prod P(A_i \mid c)$$

How to estimate probabilities from data I

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- How to estimate class prior P(c)?
  - $P(c) = N_c/N$

    e.g., P(No) = 7/10, P(Yes) = 3/10

- How to estimate $P(A_i | c)$?
  - For discrete attributes:

    → $P(A_i \mid c) = |A_{ic}| / N_c$

    $|A_{ic}|$: # instances having attribute A_i and belonging to class c

    e.g.:

    P(Status=Married|No) = 4/7

    P(Refund=Yes|Yes)=0

How to estimate probabilities from data II

- How to estimate $P(A_i | c)$? For continuous attributes
  - Discretize the range into bins
    - one ordinal attribute per bin
  - Two-way split: $(A < v)$ or $(A > v)$
    - choose only one of the two splits as new attribute
  - Probability density estimation:
    - Assume attribute follows a normal distribution
    - Use data to estimate parameters of distribution (e.g., mean and standard deviation)
    - Once probability distribution is known, can use it to estimate the conditional probability $P(A_i | c)$
    - e.g. assume Gaussian (normal) distribution:

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

How to estimate probabilities from data III

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(A_i \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(A_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

- e.g., for attribute *income* and class *no*:
  - Sample mean = 110
  - Sample variance s^2=2975

Sample variance $s^2 = \frac{\sum_{i=1}^{n}(X_i - X_{avg})^2}{n-1}$

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Population variance $\sigma^2 = \frac{\sum_{i=1}^{n}(X_i - X_{avg})^2}{n}$

Naive Bayes classifier: Example I

Training set

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

Test instance X

Outlook	Temperature	Humidity	Wind	Play
Sunny	Cool	High	Strong	?

Observations

$$P(\text{yes} \mid X) = \frac{P(X \mid \text{yes})P(\text{yes})}{P(X)} = \frac{P(O = \text{"sunny"} \mid \text{yes})P(T = \text{"}cool\text{"} \mid \text{yes})P(H = \text{"high"} \mid \text{yes})P(W = \text{"strong"} \mid \text{yes})P(\text{yes})}{P(X)}$$

$$P(O = \text{"sunny"} \mid \text{yes}) = \frac{2}{9} \qquad P(T = \text{"}cool\text{"} \mid \text{yes}) = \frac{3}{9} \qquad P(H = \text{"}high\text{"} \mid \text{yes}) = \frac{3}{9} \qquad P(W = \text{"}strong\text{"} \mid \text{yes}) = \frac{3}{9}$$

$$P(yes) = \frac{9}{14}$$

$$P(\text{no} \mid X) = \frac{P(X \mid \text{no})P(\text{no})}{P(X)} = \frac{P(O = \text{"sunny"} \mid \text{no})P(T = \text{"}cool\text{"} \mid \text{no})P(H = \text{"high"} \mid \text{no})P(W = \text{"strong"} \mid \text{no})P(\text{no})}{P(X)}$$

Naive Bayes classifier: Example II

Training set

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Test instance X

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

$$P(X \mid M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(X \mid N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(X \mid M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(X \mid N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

The problem of 0-probabilities

- Naïve Bayesian prediction requires each conditional probability $P(A_i|c)$ be non-zero. Otherwise, the predicted probability will be zero

$$c = \arg\max_{c \in C} P(c) \prod P(A_i \mid c)$$

- e.g., suppose a dataset with 1000 tuples: income=low (0); income= medium (990); income = high (10)

- **Probability estimation:**

$$\text{Original}: P(A_i \mid c) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace}: P(A_i \mid c) = \frac{N_{ic} + 1}{N_c + k}$$

$$\text{m-estimate}: P(A_i \mid c) = \frac{N_{ic} + mp}{N_c + m}$$

k: number of classes

p: prior probability

m: parameter

The problem of 0-probabilities: example

- in our example: Suppose a dataset with 1000 tuples:
  - income=low (0)
  - income= medium (990)
  - income = high (10)

- Use Laplacian correction (or Laplacian estimator): add 1 to each class value
  - Prob(income = low) = 1/1003
  - Prob(income = medium) = 991/1003
  - Prob(income = high) = 11/1003

- Result
  - The probabilities are never 0
  - The “corrected” prob. estimates are close to their “uncorrected” counterparts

Naïve Bayes classifiers: overview

(+) Easy to implement

(+) It works surprisingly good in practice, although the independence assumption is to strong .

- It does not require precise estimations of the probabilities
- It is enough if the max probability belongs to the correct class

(+) Robust to irrelevant attributes

(+) Handle missing values by ignoring the instance during probability estimate calculations

(+) Robust to noise

(+) Incremental

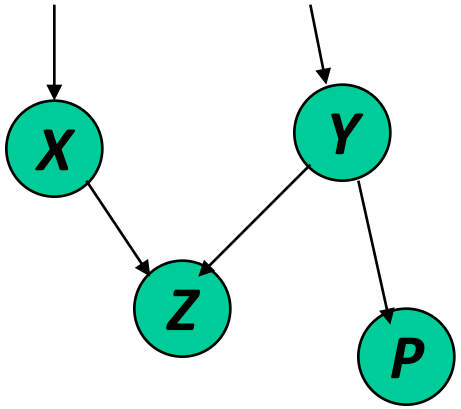
(-) Strong independence assumption

(-) Practically, dependencies exist among variables

- Dependencies among these cannot be modeled by Naïve Bayesian Classifiers
- Use other techniques such as Bayesian Belief Networks (BBN)

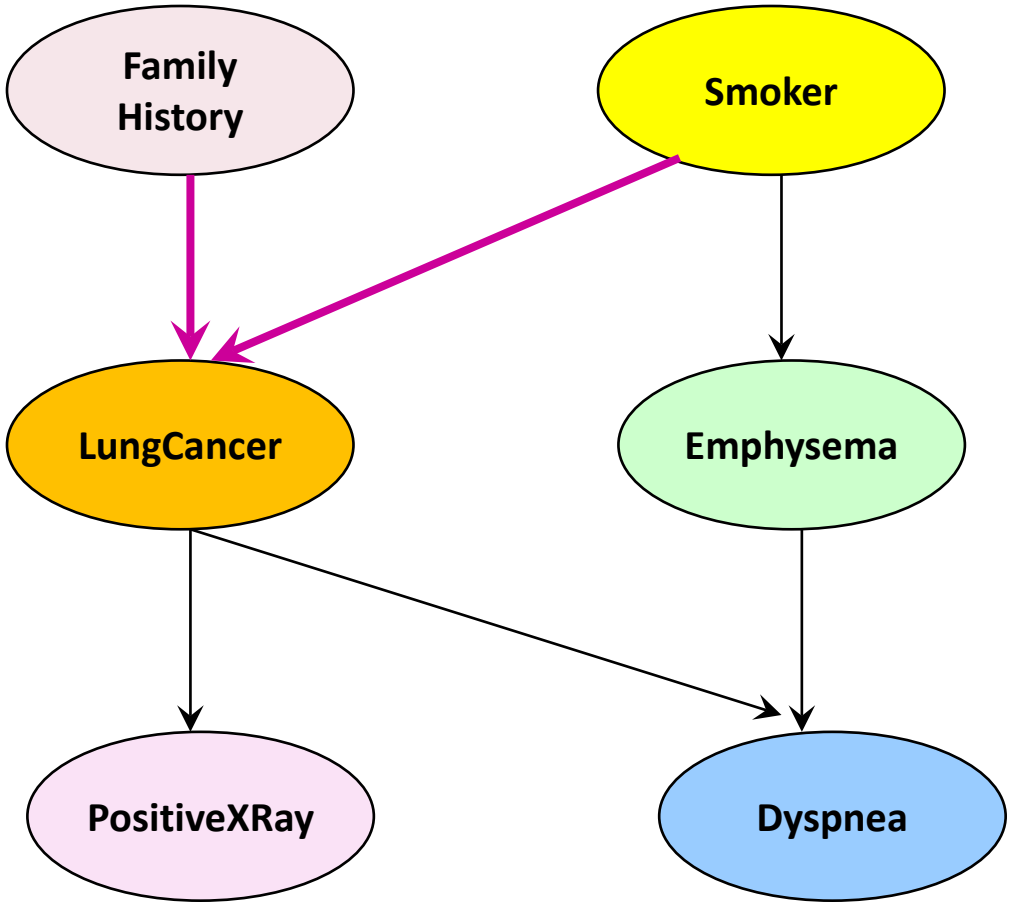
Bayesian Belief Networks

- Bayesian belief networks allow class conditional independence to be defined between subsets of variables.
- A graphical model of causal relationships
- A belief network is defined by two components:
  - A directed acyclic graph of nodes representing variables and arcs representing dependence relations among the variables.
  - A set of conditional probability tables (CPT)



- Nodes: random variables
- Links: dependency between variables
- X, Y are the parents of Z; Y is the parent of P
- No dependency between Z and P

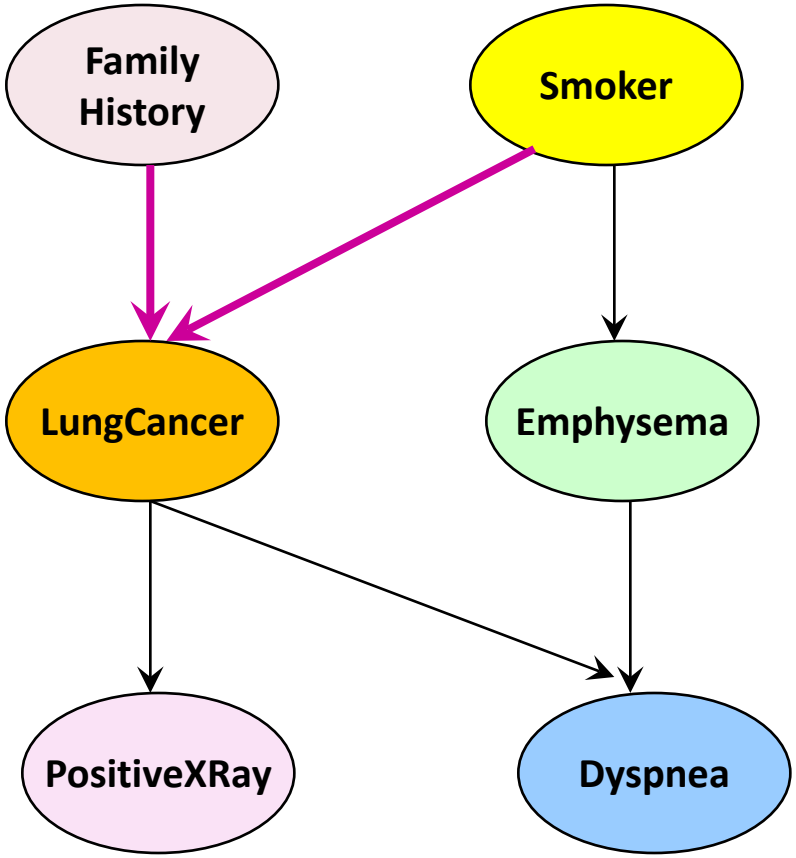
An example



- E.g., having lung cancer is influenced by a person's family history and on whether or not the person is a smoker
- PositiveXRay is independent of "family history" and "smoker" attributes once we know that the person has a PositiveXRay

Bayesian Belief Networks

A Bayesian Belief Network has a conditional probability table (CPT) for each variable Y
• CPT of Y specifies the conditional distribution P(Y|Parents(Y))



- The conditional probability table (CPT) for variable LungCancer:

	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

- Let $X = (x_1, x_2, \ldots, x_n)$ be an instance described by the variables of attributes $A_1, A_2, \ldots A_n$, respectively.
- The probability of X is given by:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(Y_i))$$

Outline

- Introduction
- Bayesian classifiers
- Lazy vs Eager learners
- k-Nearest Neighbors (or learning from your neighbors)
- Artificial neural networks
- Things you should know
- Homework/tutorial

Lazy vs Eager learners

- Eager learners
  - Construct a classification model (based on a training set)
  - Learned models are ready and eager to classify previously unseen instances
  - e.g., decision trees
- Lazy learners
  - Simply store training data and wait until a previously unknown instance arrives
  - No model is constructed.
  - known also as instance based learners, because they store the training set
  - e.g., k-NN classifier

Eager learners

• Do lot of work on training data
• Do less work on classifying new instances

Lazy learners

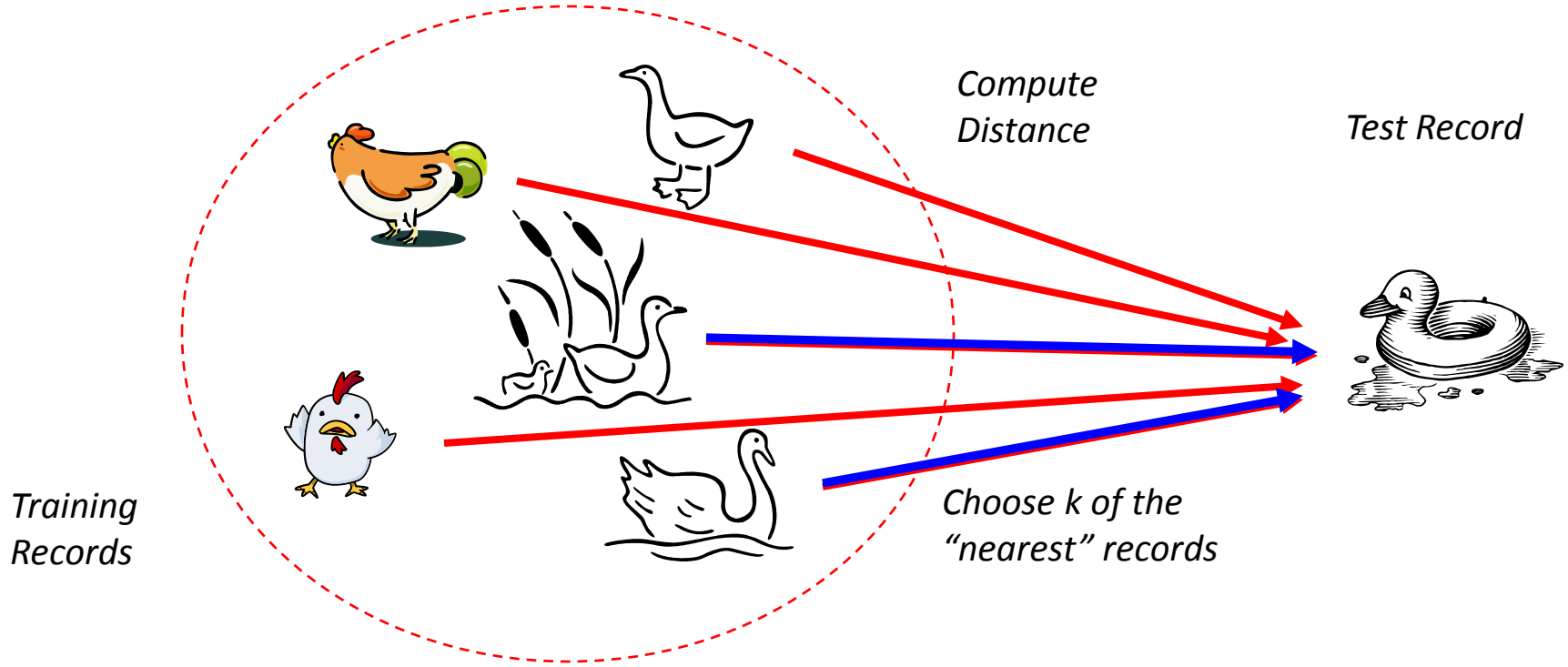
• Do less work on training data
• Do more work on classifying new instances

Outline

- Introduction
- Bayesian classifiers
- Lazy vs Eager learners
- k-Nearest Neighbors (or learning from your neighbors)
- Artificial neural networks
- Things you should know
- Homework/tutorial

Lazy learners/ Instance-based learners: k-Nearest Neighbor classifier

- Nearest-neighbor classifiers compare a given unknown instance with training tuples that are similar to it
- Basic idea: *If it walks like a duck, quacks like a duck, then it's probably a duck*



k-Nearest Neighbor classifiers

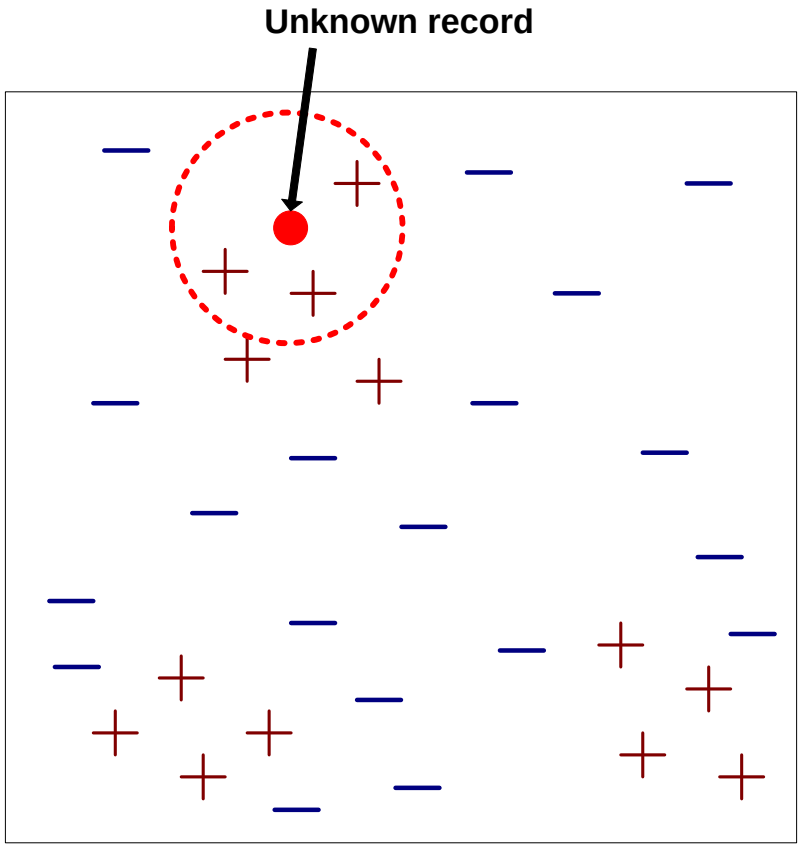
Input:

- A training set D (with known class labels)
- A distance metric to compute the distance between two instances
- The number of neighbors k

Method: Given a new unknown instance X

- Compute distance to other training records
- Identify k nearest neighbors
- Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

It requires O(|D|) for each new instance



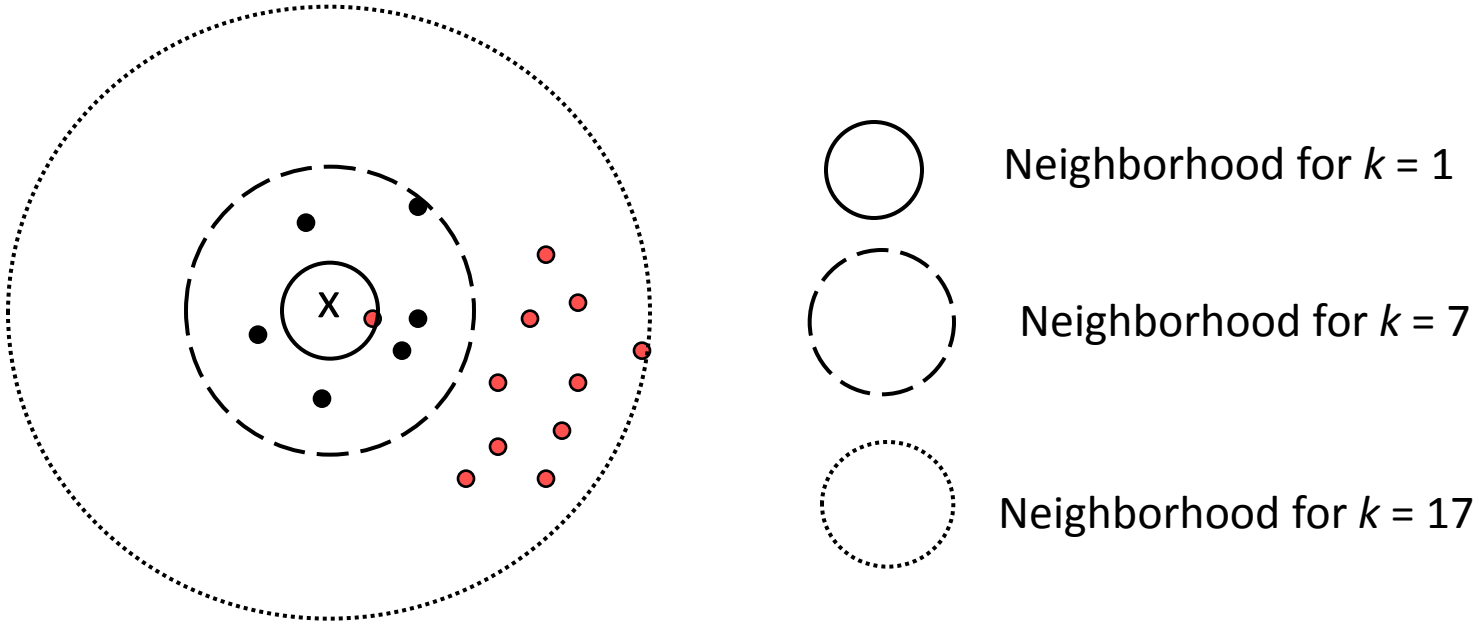
kNN algorithm

Pseudocode:

```
Input:
   T                 //training data
   K                 //Number of neighbors
   t                 //Input tuple to classify
Output:
   c                 //Class to which t is assigned
KNN algorithm:  //Algorithm to classify tuple using KNN
begin
   N = ∅;
   //Find set of neighbors, N, for t
   for each d ∈ T do
         if |N| ≤ K
         then N = N ∪ {d};
         else if ∃ u ∈ N such that
                  sim(t,u) ≤ sim(t,d) AND sim(t,u) ≤ sim(t,u') ∀ u' ∈ N
         then N = N – {u}; N = N ∪ {d};
   //Find class for classification
   c = class to which the most u ∈ N are classified
end
```

Definition of k nearest neighbors

- too small k: high sensitivity to outliers
- too large k: many objects from other classes in the resulting neighborhood
- average k: highest classification accuracy, usually 1 << k < 10



x: unknown instance

Nearest neighbor classification

- "Closeness" is defined in terms of a distance metric
  - e.g. Euclidean distance

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

- The k-nearest neighbors are selected among the training set
- The class of the unknown instance X is determined from the neighbor list
  - If k=1, the class is that of the closest instance
  - Majority voting: take the majority vote of class labels among the neighbors
    - Each neighbor has the same impact on the classification
    - The algorithm is sensitive to the choice of k
  - Weighted voting: Weigh the vote of each neighbor according to its distance from the unknown instance
    - weight factor, $w = 1/d^2$

Nearest neighbor classification: example

Name	Gender	Height	Output1	
Kristina	F	1.6m	Short	1
Jim	M	2m	Tall	
Maggie	F	1.9m	Medium	
Martha	F	1.88m	Medium	
Stephanie	F	1.7m	Short	3
Bob	M	1.85m	Medium	
Kathy	F	1.6m	Short	2
Dave	M	1.7m	Short	4
Worth	M	2.2m	Tall	
Steven	M	2.1m	Tall	
Debbie	F	1.8m	Medium	
Todd	M	1.95m	Medium	
Kim	F	1.9m	Medium	
Amy	F	1.8m	Medium	
Wynette	F	1.75m	Medium	5
Pat	F	1.6m	?	Short

Nearest neighbor classification issues I

- Different attributes have different ranges
  - e.g., height in [1.5m-1.8m]; income in [$10K -$1M]
  - Distance measures might be dominated by one of the attributes
  - Solution: normalization

- k-NN classifiers are lazy learners
  - No model is built explicitly, like in eager learners such as decision trees
  - Classifying unknown records are relatively expensive
  - Possible solutions:
    - Use index structures to speed up the nearest neighbors computation
    - Partial distance computation based on a subset of attributes

Nearest neighbor classification issues II

- The "curse of dimensionality"
  - Ratio of ($D_{max\_d} - D_{min\_d}$) to D_{min_d} converges to zero with increasing dimensionality *d*
    - D_{max_d}: distance to the nearest neighbor in the d-dimensional space
    - D_{min_d}: distance to the farthest neighbor in the d-dimensional space
  - This implies that:
    - all points tend to be ~ equidistant from each other in high dimensional spaces
    - the distances between points cannot be used to differentiate them
  - Possible solutions:
    - Dimensionality reduction (e.g. PCA)
    - Work on a subset of dimensions

k-NN classifiers: overview

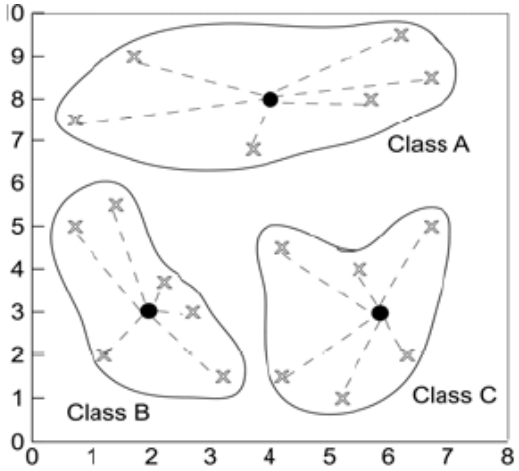
(+-) Lazy learners: Do not require model building , but testing is more expensive

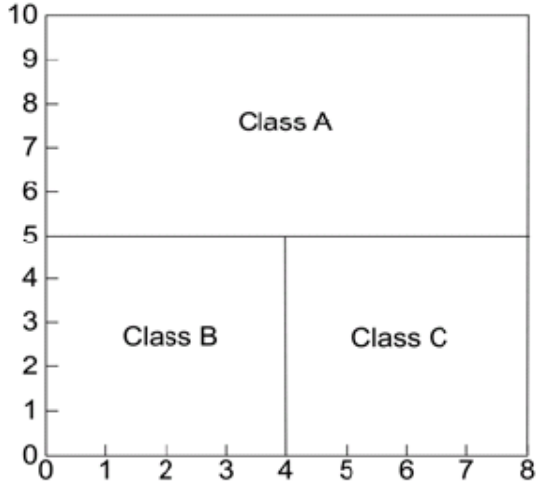
(-) Classification is based on local information in contrast to e.g. DTs that try to find a global model that fits the entire input space: Susceptible to noise

(+) Incremental classifiers

(-) The choice of distance function and k is important

(+) Nearest-neighbor classifiers can produce arbitrarily shaped decision boundaries, in contrary to e.g. decision trees that result in axis parallel hyper rectangles



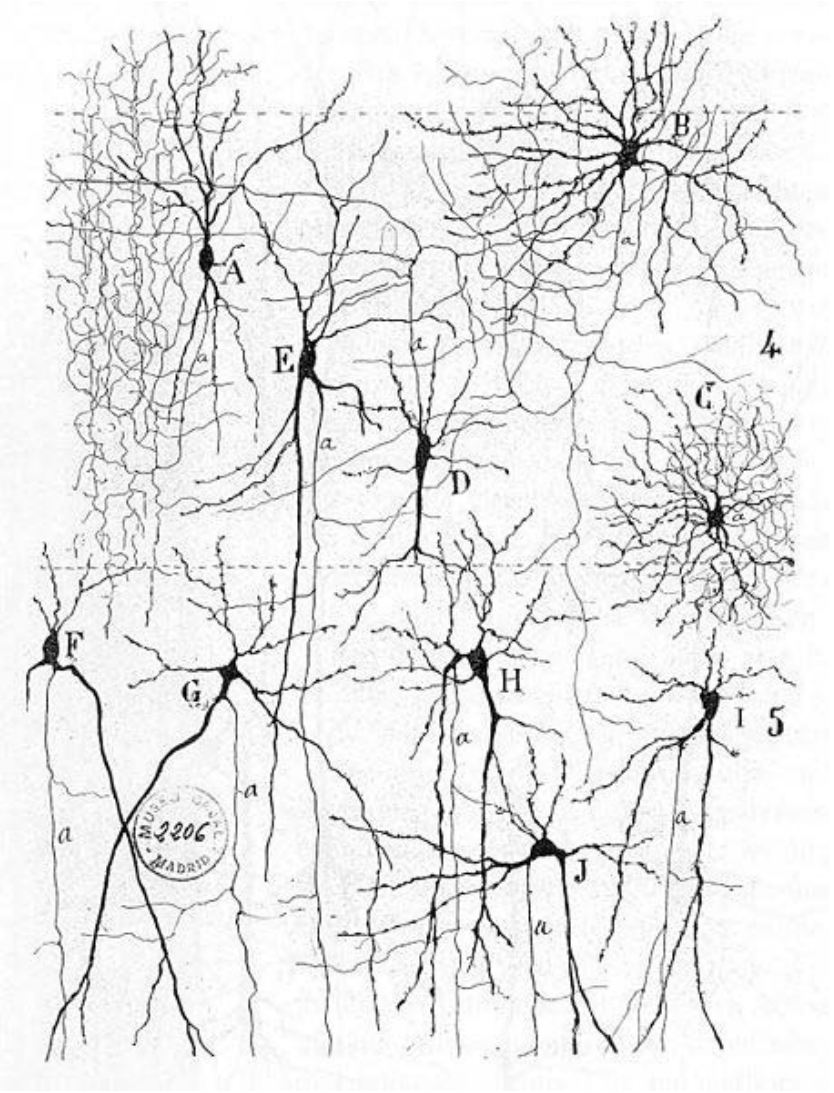


Outline

- Introduction
- Bayesian classifiers
- Lazy vs Eager learners
- k-Nearest Neighbors (or learning from your neighbors)
- Artificial neural networks
- Things you should know
- Homework/tutorial

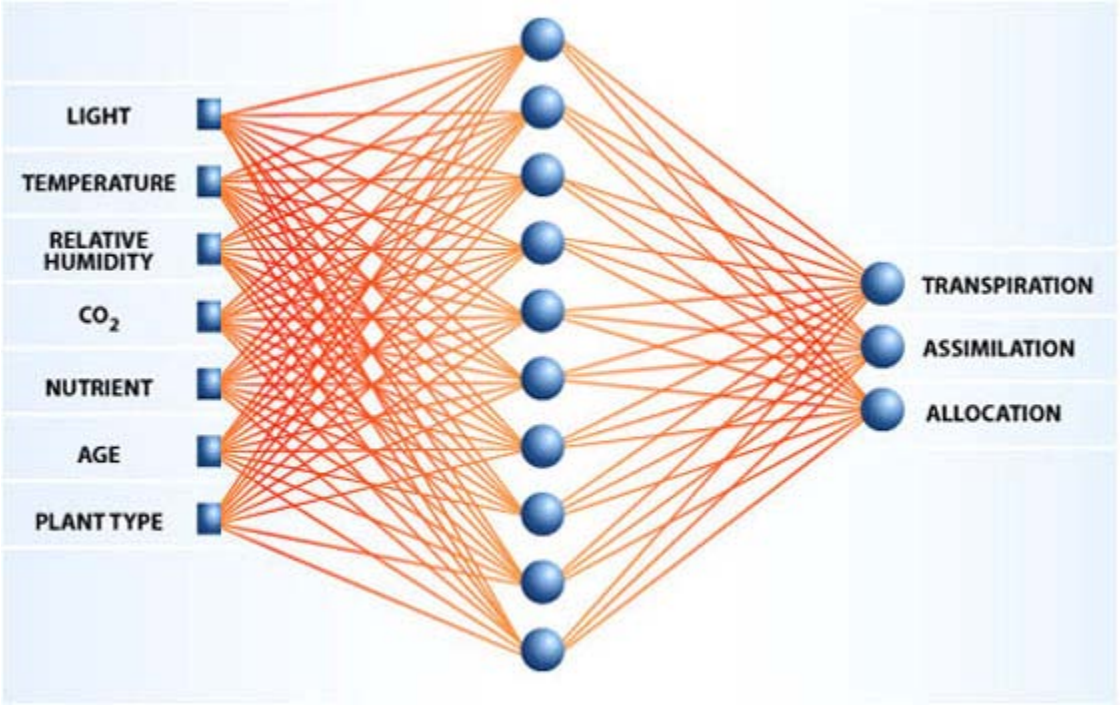
Artificial Neural Networks (ANN): motivation

- Inspired by attempts to simulate biological neural systems
- Human brain consists primarily of nerve cells (neurons), linked together with other neurons via strands of fiber (axons)
  - Axons are used to transmit nerve impulses from one neuron to another whenever the neurons are stimulated
- A neuron is connected to the axons of other neurons by dendrites
- The contact point between a dendrite and an axon is called a synapse
- Neurologists have discovered that the human brain learns by changing the strength of the synaptic connection between neurons upon repeated stimulation by the same impulse



Artificial Neural Networks (ANN) I

- Analogous to human brain function, an ANN consists of an interconnected assembly of nodes and directed links.

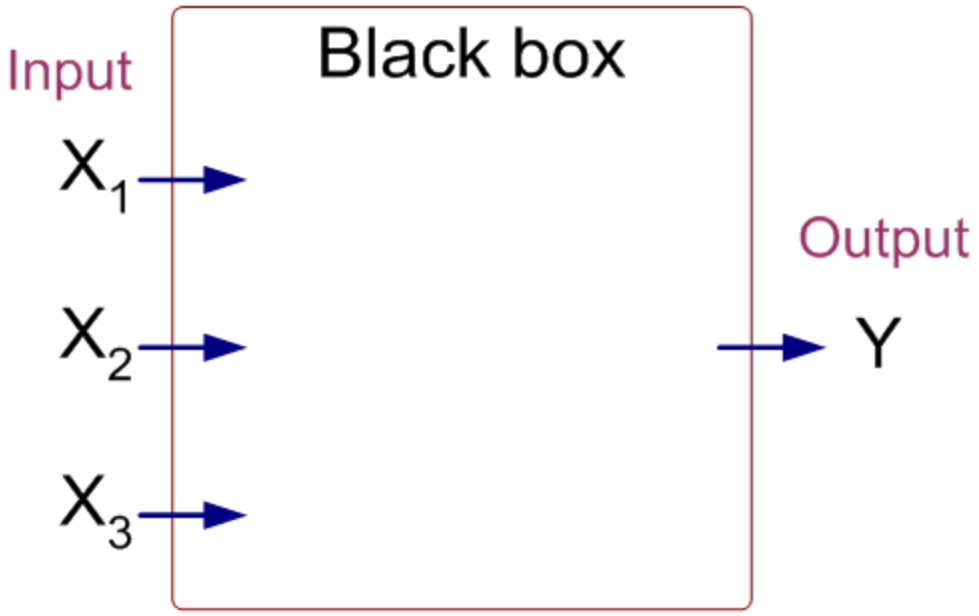


NASA: A Prediction of Plant Growth in Space

http://aemc.jpl.nasa.gov/activities/bio_regen.cfm

Artificial Neural Networks (ANN) II

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

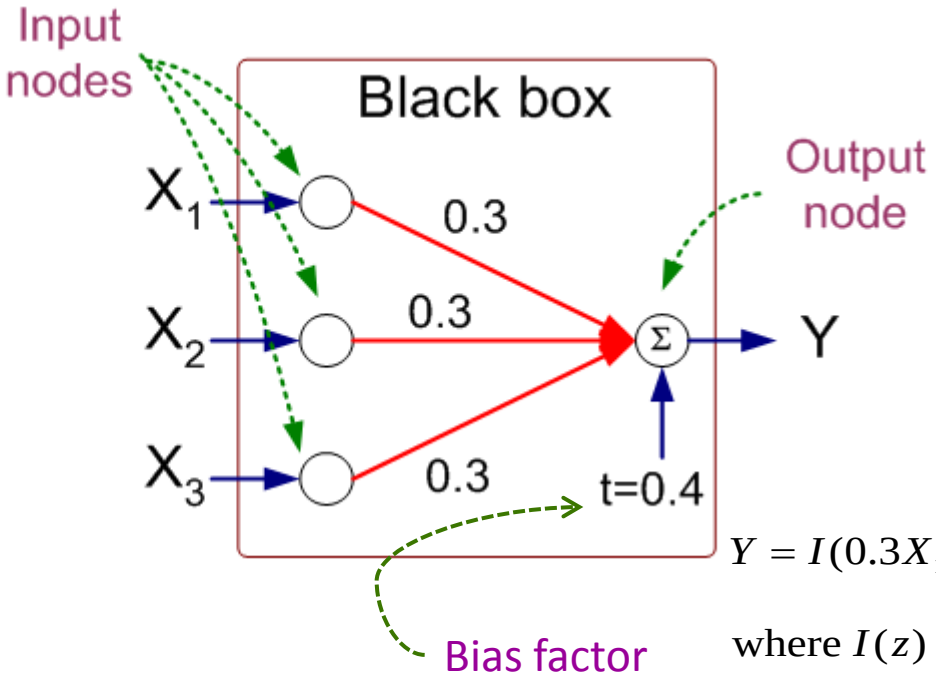


Output Y is 1 if at least two of the three inputs are equal to 1.

Artificial Neural Networks (ANN) III

- The simplest ANN model is called perceptron and consists of two types of nodes (also called neurons or units):
  - input nodes: represent the input variables
  - output nodes: represent model output
- Each input node is connected via a weighted link to an output node

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

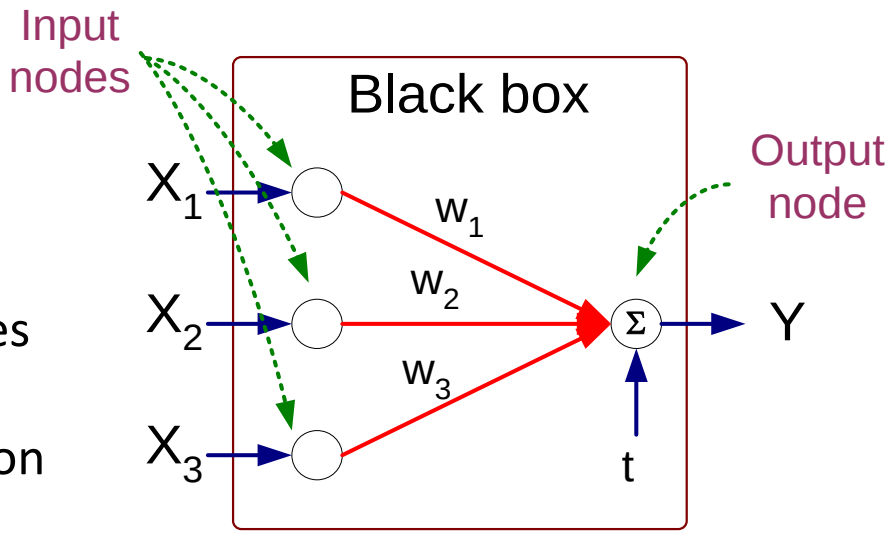


$$Y = I(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4 > 0)$$

$$\text{where } I(z) = \begin{cases} 1 & \text{if } z \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Artificial Neural Networks (ANN) IV

- Model is an assembly of interconnected nodes and weighted links
- Input nodes simply transmit the values they receive to their outgoing nodes without performing any transformation
- Output node sums up each of its input value according to the weights of its links
- Compare output node against some threshold t



Perceptron Model

$$Y = I(\sum_i w_i X_i - t) \quad or$$

$$Y = sign(\sum_i w_i X_i - t)$$

Learning perceptron model

- During the training phase of a perceptron model, the weight parameters w are adjusted until the outputs of the perceptron become consistent with the true outputs of the training data

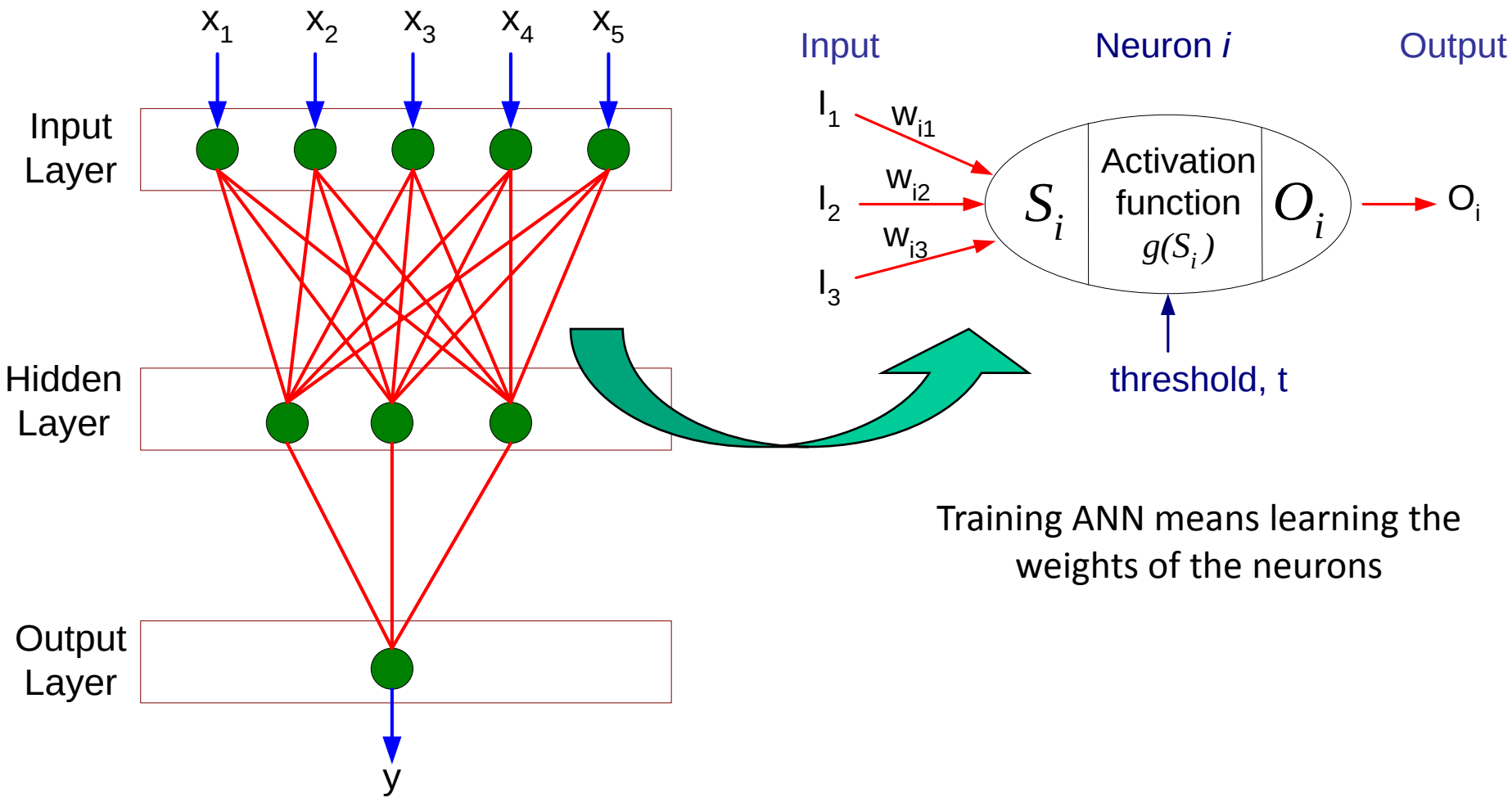
Perceptron learning algorithm

1. Let $D=\{(x_i,y_i)|i=1,2,\ldots,n\}$ be the training set
2. Initialize the weigh vector with random values $w^{(0)}$
3. repeat
4. for each training example (x_i,y_i) in D do
5. compute the predicted output $y'_i{}^{(k)}$
6. for each weight w_j do
7. Update the weight $w_j^{(k+1)}= w_j^{(k)} + \lambda(y_i - y'_i{}^{(k)})x_{ij}$
8. end for
9. until stopping condition is met

Weight update formula

Learning rate

General structure: Multilayer ANN



Training ANN means learning the weights of the neurons

Outline

- Introduction
- Bayesian classifiers
- Lazy vs Eager learners
- k-Nearest Neighbors (or learning from your neighbors)
- Artificial neural networks
- Things you should know
- Homework/tutorial

Things you should know

- Bayesian Classifiers: Bayes rule, Maximum Likelihood classification
- Naïve Bayes classifiers
  - Independence assumption
- Bayesian Belief Networks : general idea
- Eager learners – Lazy learners
- k-NN classifiers
  - k/ Distance function
  - Voting schema
- Neural networks: general idea

Homework/ Tutorial

Tutorial: this Thursday tutorial on

- Distance functions/ Evaluation of classifiers /Decision trees
- No lecture next Tuesday! Tutorial yes next Thursday

Homework:

- Implement a Naïve Bayes classifier for classifying text posts into 20 predefined categories.
- 20 newsgroup dataset: http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.data.html

Suggested reading:

- Han J., Kamber M., Pei J. *Data Mining: Concepts and Techniques 3rd ed., Morgan Kaufmann, 2011* (Chapters 8, 9)
- Tan P.-N., Steinbach M., Kumar V., *Introduction to Data Mining*, Addison-Wesley, 2006 (Chapters 4, 5).
  - Chapter 4 is available online at: http://www-users.cs.umn.edu/~kumar/dmbook/ch4.pdf

Dataset categories

1. alt.atheism
2. comp.graphics
3. comp.os.ms-windows.misc
4. comp.sys.ibm.pc.hardware
5. comp.sys.mac.hardware
6. comp.windows.x
7. misc.forsale
8. rec.autos
9. rec.motorcycles
10. rec.sport.baseball
11. rec.sport.hockey
12. sci.crypt
13. sci.electronics
14. sci.med
15. sci.space
16. soc.religion.christian
17. talk.politics.guns
18. talk.politics.mideast
19. talk.politics.misc
20. talk.religion.misc