Reference Material

Deng Cai (蔡登)

College of Computer Science
Zhejiang University

dengcai@gmail.com

Three Topics

- Bayesian Decision Theory
- Artificial Neural Network & Deep Learning
- k Nearest Neighbor Classifier

- Experimental materials download link:

http://summer2019.in.zjulearning.org/

Bayesian Decision Theory

Deng Cai (蔡登)

College of Computer Science
Zhejiang University

dengcai@gmail.com

Goal

- To finish the problems *bayes_decision_rule* and *text_classification*, you need to know:
  - What is Bayes' Theorem.
  - How to calculate the prior, likelihood, and posterior.
  - What are maximum likelihood decision rule, optimal bayes decision rule (maximum posterior), and minimum bayes risk rule.

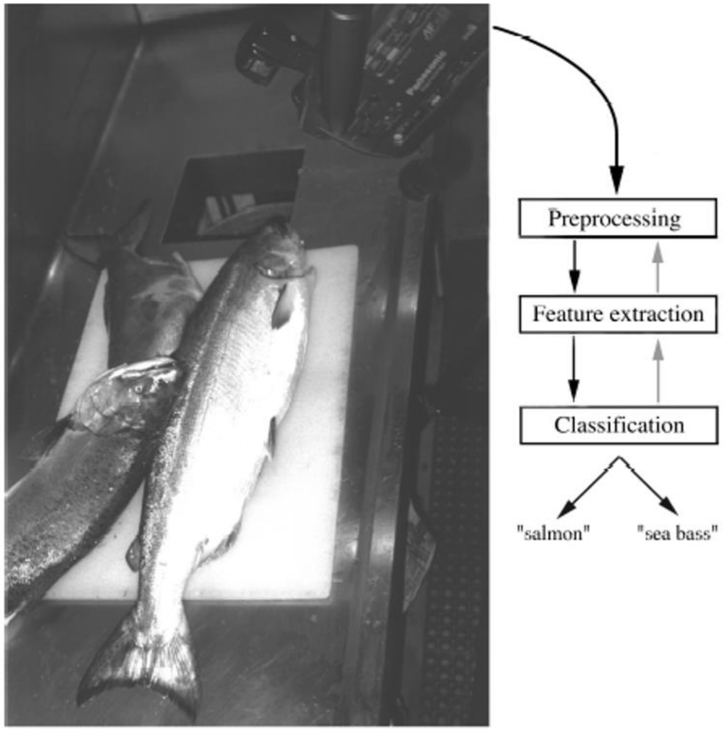
  How to design a classifier according to the above rules.

Bayesian Decision Theory

- Decision problem posed in probabilistic terms

- x: sample
- ω: state of the nature
- $P(\omega|x)$: given x , what is the probability of the state of the nature.

- Sea bass / Salmon Example



Basics of Probability

- An experiment is a well-defined process with observable outcomes.
- The set or collection of all outcomes of an experiment is called the sample space, S.
- An event E is any subset of outcomes from S.
- Probability of an event, P(E) is P(E) = number of outcomes in E / number of outcomes in S.

Bayes' Theorem

- Conditional probability: $P(A|B) = \frac{P(A,B)}{P(B)}$.
  - Test of Independence: A and B are said to be independent if and only if P(A, B) = P(A) P(B).

- Bayes' Theorem:

likelihood prior

$$\underbrace{P(A|B)}_{\text{posterior}} = \frac{\overbrace{P(B|A)}^{\text{likelihood}}\overbrace{P(A)}^{\text{prior}}}{P(B)}$$

Prior

- A priori (prior) probability of the state of nature
  - Random variable (State of nature is unpredictable)
  - Reflects our prior knowledge about how likely we are to observe a sea bass or salmon
  - The catch of salmon and sea bass is equiprobable
    - $P(\omega_1) = P(\omega_2)$ (uniform priors)
    - $P(\omega_1) + P(\omega_2) = 1$ (exclusivity and exhaustivity)

- Decision rule with only the prior information
  - Decide ω_1 if $P(\omega_1) > P(\omega_2)$, otherwise decide ω_2

Likelihood

- Suppose now we have a measurement or feature on the state of nature - say the fish lightness value
- $P(x|\omega_1)$ and $P(x|\omega_2)$ describe the difference in lightness feature between populations of sea bass and salmon
- $P(x|\omega_j)$ is called the **likelihood** of *ω_j with respect to x; the category ω_j for which $P(x \mid \omega_j)$* is large is more likely to be the true category
- **Maximum likelihood decision**
  - Assign input pattern x to class ω_1 if

$$P(x \mid \omega_1) > P(x \mid \omega_2), \text{ otherwise } \omega_2$$

Posterior

- Bayes formula

$$P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{P(x)}$$

$$P(x) = \sum_{i=1}^{k} P(x|\omega_i)P(\omega_i)$$

- **Posterior** = (**Likelihood × Prior**) / Evidence
  - Evidence $P(\boldsymbol{x})$ can be viewed as a scale factor that guarantees that the posterior probabilities sum to 1

Posterior ∝ Likelihood × Prior

Optimal Bayes Decision Rule

- $P(\omega_1 \mid x)$ is the probability of the state of nature being ω_1 given that feature value x has been observed
- Decision given the posterior probabilities, Optimal Bayes Decision rule

X is an observation for which:

if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$ ➔ True state of nature = ω_1

if $P(\omega_1 \mid x) < P(\omega_2 \mid x)$ ➔ True state of nature = ω_2

Bayes decision rule minimizes the probability of error, that is the term Optimal comes from. But why? Can you prove it?

Optimal Bayes Decision Rule

Based on Bayes decision rule, whenever we observe a particular x, the probability of error is:

$$P(error \mid x) = P(\omega_1 \mid x) \text{ if we decide } \omega_2$$

$$P(error \mid x) = P(\omega_2 \mid x) \text{ if we decide } \omega_1$$

Bayes decision rule:

Decide ω_1 if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$; otherwise decide ω_2

Therefore:

$$P(error \mid x) = min\ [P(\omega_1 \mid x), P(\omega_2 \mid x)]$$

- The unconditional error, *P(error), obtained by integration over all x w.r.t. p(x)*

Optimal Bayes Decision Rule

- Decide ω_1 *if* $P(\omega_1 \mid x) > P(\omega_2 \mid x)$; otherwise decide ω_2

- Special cases:

(i) $P(\omega_1) = P(\omega_2)$; *Decide* ω_1 if

$$P(x \mid \omega_1) > P(x \mid \omega_2), \text{ otherwise } \omega_2$$

Maximum likelihood decision

(ii) $P(x \mid \omega_1) = P(x \mid \omega_2)$; *Decide* ω_1 if

$$P(\omega_1) > P(\omega_2), \text{ otherwise } \omega_2$$

Bayes Risk

- Conditional risk

$$R(\alpha_i|\boldsymbol{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)P(\omega_j|\boldsymbol{x})$$

- Select the action for which the conditional risk $R(\alpha_i|\boldsymbol{x})$ *is minimum*

$$R = \int R(\alpha_i|\boldsymbol{x})\,p(\boldsymbol{x})d\boldsymbol{x}$$

- Risk R is minimum and R in this case is called the
  - Bayes risk = best performance that can be achieved!

Example 1: Two-category classification

$$\alpha_1 : deciding\ \omega_1$$

$$\alpha_2 : deciding\ \omega_2$$

$$\lambda_{ij} = \lambda(\alpha_i \mid \omega_j)$$

Conditional risk:

$$R(\alpha_1 \mid x) = \lambda_{11} P(\omega_1 \mid x) + \lambda_{12} P(\omega_2 \mid x)$$

$$R(\alpha_2 \mid x) = \lambda_{21} P(\omega_1 \mid x) + \lambda_{22} P(\omega_2 \mid x)$$

How to achieve Bayes risk?

Example 1: Two-category classification

Bayes rule is the following:

$$\text{if } R(\alpha_1 \mid x) < R(\alpha_2 \mid x)$$

action α_1: "decide ω_1" is taken

This results in the equivalent rule:

decide ω_1 if:

$$(\lambda_{21} - \lambda_{11}) P(x \mid \omega_1) P(\omega_1) > (\lambda_{12} - \lambda_{22}) P(x \mid \omega_2) P(\omega_2)$$

and decide ω_2 otherwise

Example 1: Two-category classification

- The preceding rule is equivalent to the following rule:

- If $\frac{P(\boldsymbol{x}|\omega_1)}{P(\boldsymbol{x}|\omega_2)} > \frac{\lambda_{12}-\lambda_{22}}{\lambda_{21}-\lambda_{11}} \times \frac{P(\omega_2)}{P(\omega_1)}$

Then take action α_1 (decide ω_1)

Otherwise take action α_2 (decide ω_2)

- "If the likelihood ratio exceeds a threshold value that is independent of the input pattern x, we can take optimal actions"

Naïve Bayes Classifier

- Given $\boldsymbol{x} = (x_1, \cdots x_p)^T$
  - Goal is to predict class ω
  - Specifically, we want to find the value of ω that maximizes $P(\omega|\boldsymbol{x}) = P(\omega|x_1, \cdots x_p)$

$$P(\omega|x_1, \cdots x_p) \propto P(x_1, \cdots x_p|\omega)P(\omega)$$

- Independence assumption among features

$$P(x_1, \cdots x_p|\omega) = P(x_1|\omega) \cdots P(x_p|\omega)$$

How to Estimate Probabilities from Data?

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

- Class: $P(\omega_k) = \frac{N_{\omega_k}}{N}$
  - e.g., P(No) = 7/10, P(Yes) = 3/10
- For discrete attributes:

  $$P(x_i|\omega_k) = \frac{|x_{ik}|}{N_{\omega_k}}$$

  - where $|x_{ik}|$ is number of instances having attribute x_i and belongs to class ω_k
  - Examples:

    P(Status=Married|No) = 4/7
    P(Refund=Yes|Yes)=0

How to Estimate Probabilities from Data?

- For continuous attributes:
  - Discretize the range into bins
    - one ordinal attribute per bin
    - violates independence assumption
  - Two-way split: $(x < v)$ or $(x > v)$
    - 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 $P(x_1|\omega)$

How to Estimate Probabilities from Data?

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:

$$P(x_i \mid \omega_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} \exp\left(-\frac{(x_i - \mu_{ij})^2}{2\sigma_{ij}^2}\right)$$

  - One for each (x_i, ω_i) pair

- For (Income, Class=No):
  - If Class=No
    - sample mean = 110
    - sample variance = 2975

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

Example of Naïve Bayes Classifier

Given a Test Record:

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120\text{K})$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
P(Marital Status=Single|No) = 2/7
P(Marital Status=Divorced|No)=1/7
P(Marital Status=Married|No) = 4/7
P(Marital Status=Single|Yes) = 2/7
P(Marital Status=Divorced|Yes)=1/7
P(Marital Status=Married|Yes) = 0

For taxable income:

If class=No:	sample mean=110
	sample variance=2975
If class=Yes:	sample mean=90
	sample variance=25

- P(X|Class=No) = P(Refund=No|Class=No) × P(Married| Class=No) × P(Income=120K| Class=No)
  = 4/7 × 4/7 × 0.0072 = 0.0024
- P(X|Class=Yes) = P(Refund=No| Class=Yes) × P(Married| Class=Yes) × P(Income=120K| Class=Yes)
  = 1 × 0 × 1.2 × 10^{-9} = 0

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)

=> Class = No

Example of Naïve Bayes Classifier

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

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

A: attributes

M: mammals

N: non-mammals

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

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

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

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

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

=> Mammals

Naïve Bayes (Summary)

- Advantages
  - Robust to isolated noise points
  - Handle missing values by ignoring the instance during probability estimate calculations
  - Robust to irrelevant attributes

- Disadvantages
  - Independence assumption may not hold for some attributes
  - Smoothing

$$P(x_i|\omega_k) = \frac{|x_{ik}| + 1}{N_{\omega_k} + K}$$

Artificial Neural Network & Deep Learning

Deng Cai (蔡登)

College of Computer Science
Zhejiang University

dengcai@gmail.com

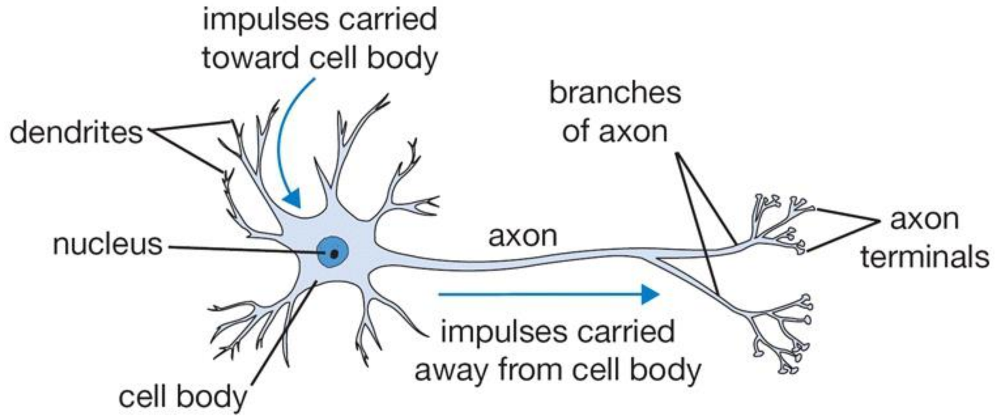
Goal

- To finish the problem *neural_networks*, you need to know:
  - What is artificial neural network.
  - The forward and backpropagation processes of the neural network.

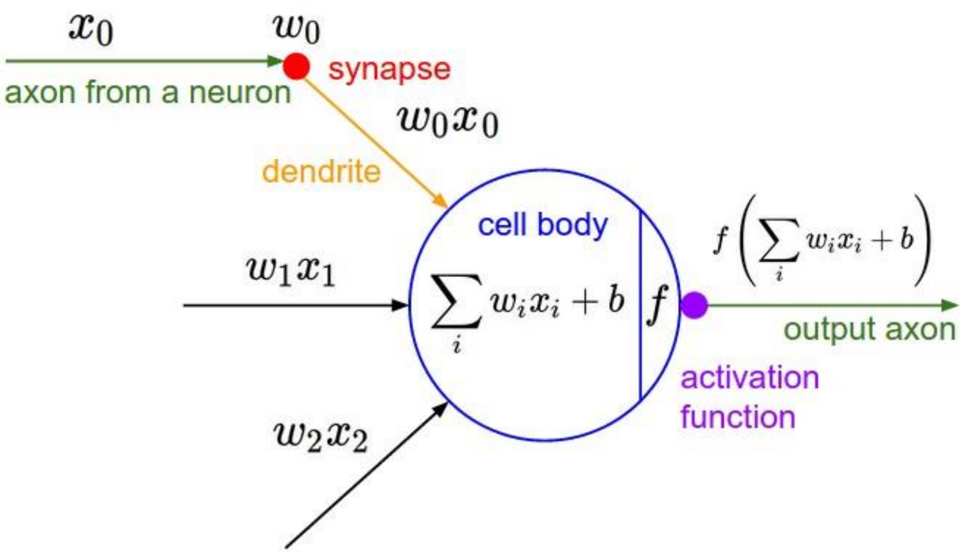
Natural Neural Net Models

- Human brain consists of very large number of neurons (between 10^{10} to 10^{12})
- No. of interconnections per neuron is between 1K to 10K
- Total number of interconnections is about 10^{14}
- Damage to a few neurons or synapse (links) does not appear to impair overall performance significantly (robustness)

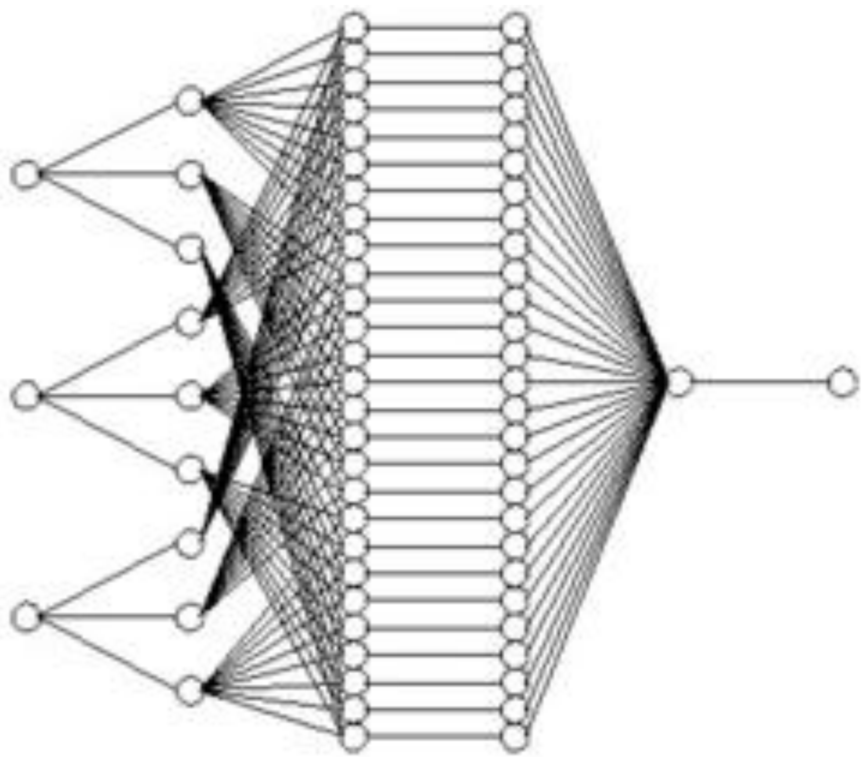
The Artificial Neural Network



- A cartoon drawing of a biological neuron



The Artificial Neural Network



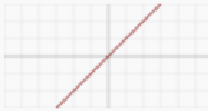
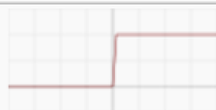
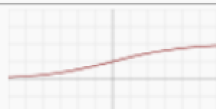
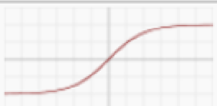
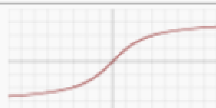
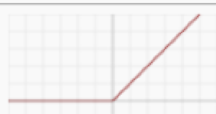
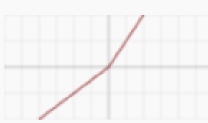
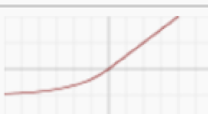
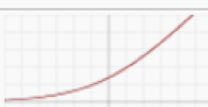
- This is the fully-connected neural network. There are many varieties of neural network, *e.g.*, convolutional neural network (CNN) and recurrent neural network (RNN).

Activation Function

- Activation Function *f*
  - Must be non-linear (otherwise, 3-layer network is just a linear discriminant) and saturate (have max and min value) to keep weights and activation functions bounded
  - Activation function and its derivative must be continuous and smooth; optionally monotonic
  - Choice may depend on the problem. Eg. Gaussian activation if the data comes from a mixture of Gaussians
  - Eg: sigmoid (most popular), polynomial, tanh

- Parameters of activation function (e.g. Sigmoid)
  - Centered at 0, odd function f(-net) = -f(net) (anti-symmetric); leads to faster learning
  - Depend on the range of the input values

Activation Function

Name	Plot	Equation	Derivative
Identity		$f(x)=x$	$f'(x)=1$
Binary step		$f(x)=\begin{cases}0 & \text{for } x<0\\ 1 & \text{for } x\geq 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for } x\neq 0\\ ? & \text{for } x=0\end{cases}$
Logistic (a.k.a Soft step)		$f(x)=\frac{1}{1+e^{-x}}$	$f'(x)=f(x)(1-f(x))$
TanH		$f(x)=\tanh(x)=\frac{2}{1+e^{-2x}}-1$	$f'(x)=1-f(x)^2$
ArcTan		$f(x)=\tan^{-1}(x)$	$f'(x)=\frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x)=\begin{cases}0 & \text{for } x<0\\ x & \text{for } x\geq 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for } x<0\\ 1 & \text{for } x\geq 0\end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x)=\begin{cases}\alpha x & \text{for } x<0\\ x & \text{for } x\geq 0\end{cases}$	$f'(x)=\begin{cases}\alpha & \text{for } x<0\\ 1 & \text{for } x\geq 0\end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x)=\begin{cases}\alpha(e^x-1) & \text{for } x<0\\ x & \text{for } x\geq 0\end{cases}$	$f'(x)=\begin{cases}f(x)+\alpha & \text{for } x<0\\ 1 & \text{for } x\geq 0\end{cases}$
SoftPlus		$f(x)=\log_e(1+e^x)$	$f'(x)=\frac{1}{1+e^{-x}}$

......

General Feedforward Operation

- Case of c output units

$$g_k(\boldsymbol{x}) \equiv z_k = f\left(\sum_{j=1}^{n_H} w_{kj}\, f\left(\sum_{i=1}^{d} w_{ji}x_i + w_{j0}\right) + w_{k0}\right)$$

$$k = 1, \cdots, c$$

- Hidden units enable us to express more complicated nonlinear functions and extend classification capability
- Assume for now that all activation functions are identical
- Question: Can every decision boundary be implemented by a three-layer network described by the above equation?

Expressive Power of Multi-layer Networks

- Answer: Yes (due to A. Kolmogorov)
  - Any continuous function from input to output can be implemented in a three-layer net, given sufficient number of hidden units n_H, proper nonlinearities, and weights.

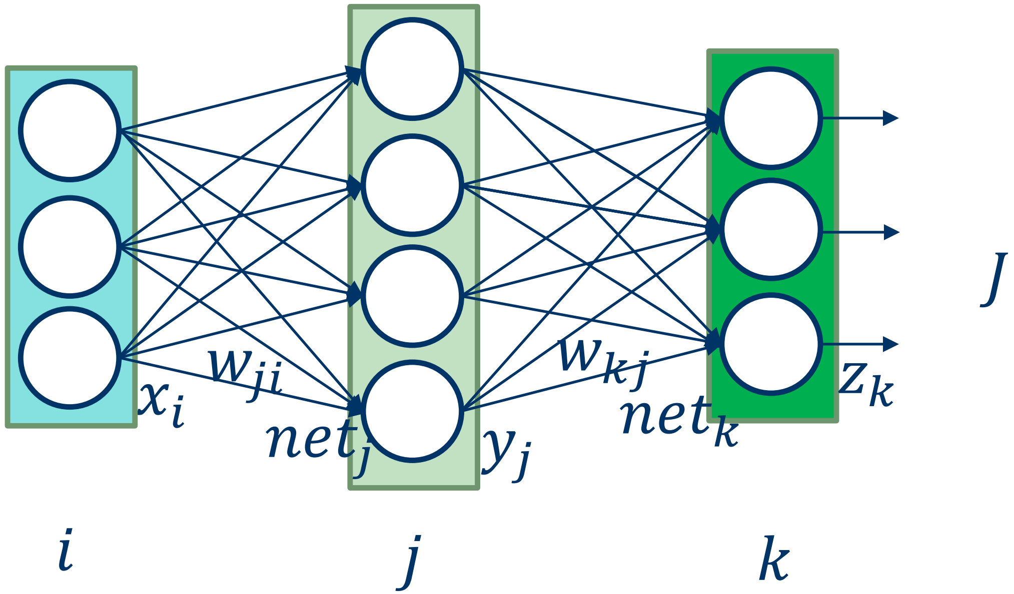
- Any continuous function $g(\boldsymbol{x})$ defined on the unit hypercube $I^n (I = [0,1]$ and $n \geq 2)$ can be represented in the following form:

$$g(\boldsymbol{x}) = \sum_{j=1}^{2n+1} \Xi_j \left(\sum_{i=1}^{d} \Phi_{ij}(x_i) \right)$$

for properly chosen functions Ξ_j and Φ_{ij}

Gradient Descent

Backpropagation Algorithm



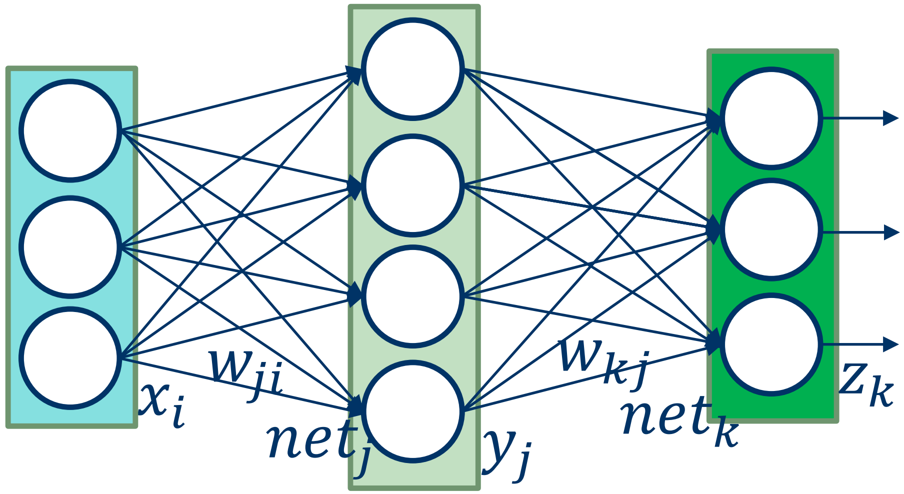
$$\Delta \boldsymbol{w} = -\eta \nabla J = -\eta \frac{\partial J}{\partial \boldsymbol{w}} \qquad \Delta w_{mn} = -\eta \frac{\partial J}{\partial w_{mn}}$$

- Where η is the learning rate which indicates the relative size of the change in weights

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \Delta \boldsymbol{w}^{(t)}$$

- where t indexes the particular pattern presentation

Backpropagation Algorithm



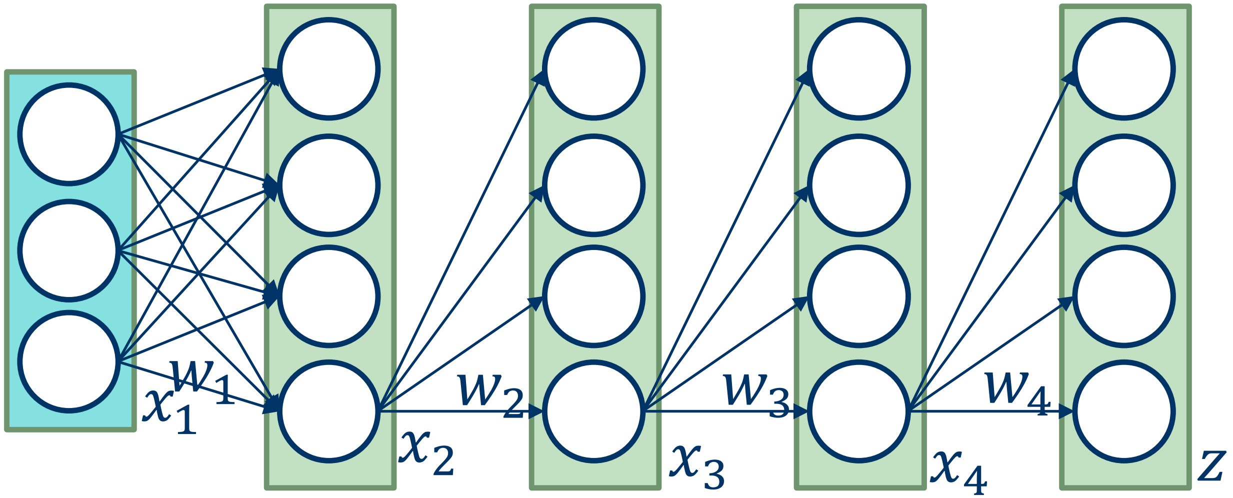
$$J = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2$$

$$\frac{\partial J}{\partial w_{kj}} = \frac{\partial J}{\partial z_k} \cdot \frac{\partial z_k}{\partial net_k} \cdot \frac{\partial net_k}{\partial w_{kj}} = (z_k - t_k) \cdot f'(net_k) \cdot y_j$$

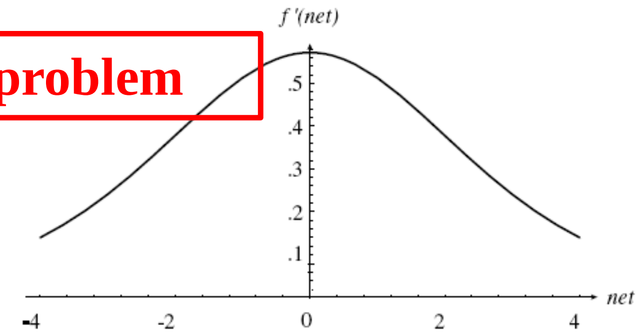
$$\frac{\partial J}{\partial w_{ji}} = \frac{\partial J}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}}$$

$$= \left(\sum_{k=1}^{c}(z_k - t_k) \cdot f'(net_k) \cdot w_{kj}\right) \cdot f'(net_j) \cdot x_i$$

Backpropagation Algorithm



vanishing gradient problem



First order derivative

$$\frac{\partial J}{\partial w_4} = \frac{\partial J}{\partial z} \cdot f'(net_4) \cdot x_4$$

$$\frac{\partial J}{\partial w_3} = \frac{\partial J}{\partial z} \cdot f'(net_4) \cdot w_4 \cdot f'(net_3) \cdot x_3$$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial z} \cdot f'(net_4) \cdot w_4 \cdot f'(net_3) \cdot w_3 f'(net_2) \cdot x_2$$

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial z} \cdot \overset{<1}{f'(net_4) \cdot w_4} \cdot \overset{<1}{f'(net_3) \cdot w_3} \overset{<1}{f'(net_2) \cdot w_2} \overset{<1}{f'(net_1)} \cdot x_1$$

k Nearest Neighbor Classifier

Deng Cai (蔡登)

College of Computer Science
Zhejiang University

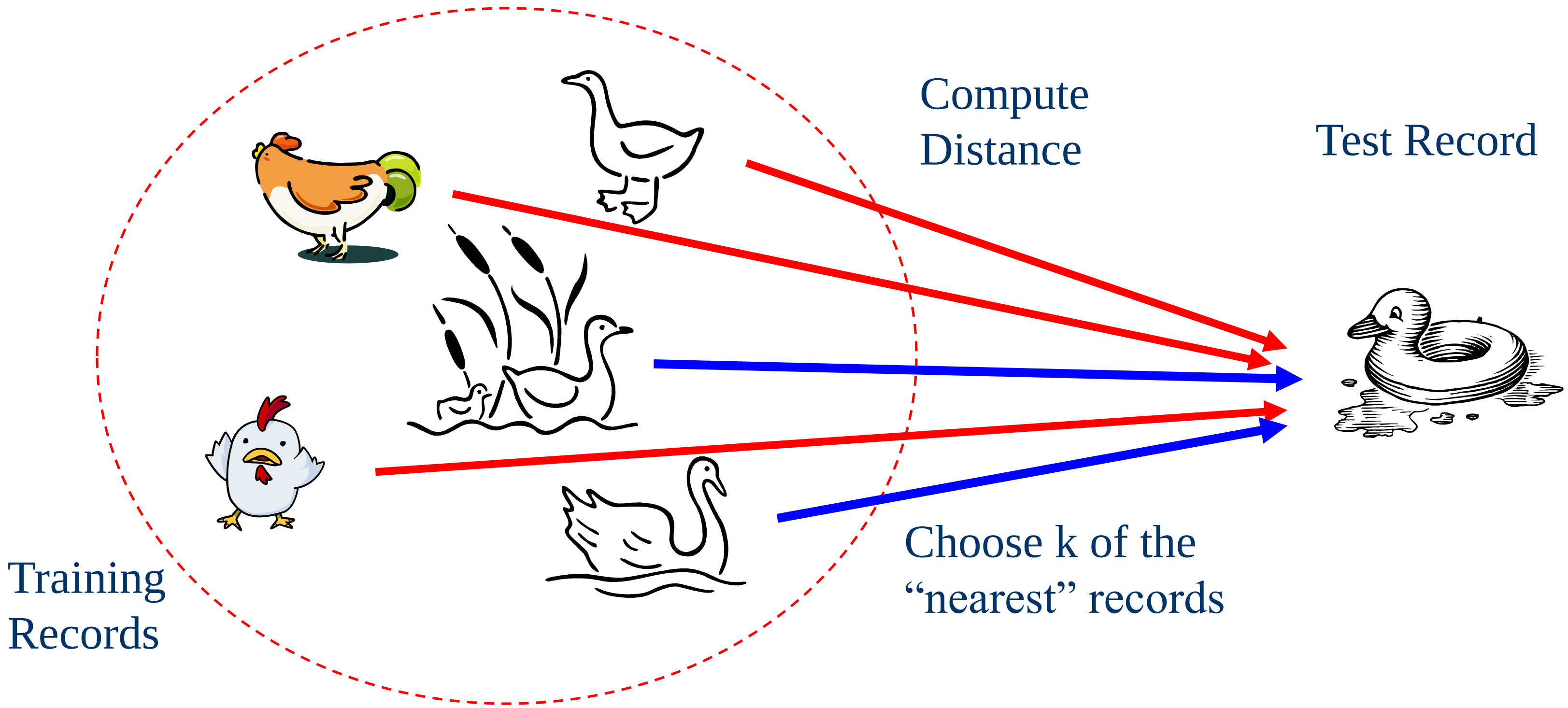
dengcai@gmail.com

Goal

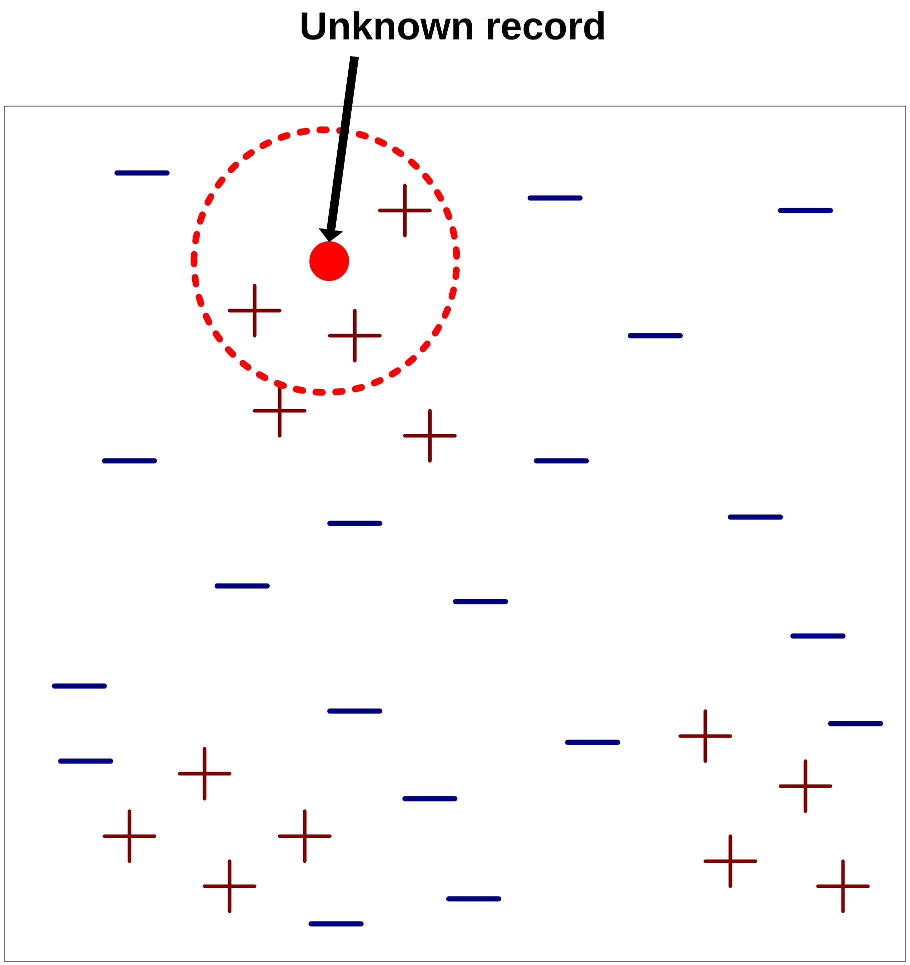
- To finish the problem *knn*, you need to know:
  - What is the main idea of knn.
  - How to apply knn in a practical problem.

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



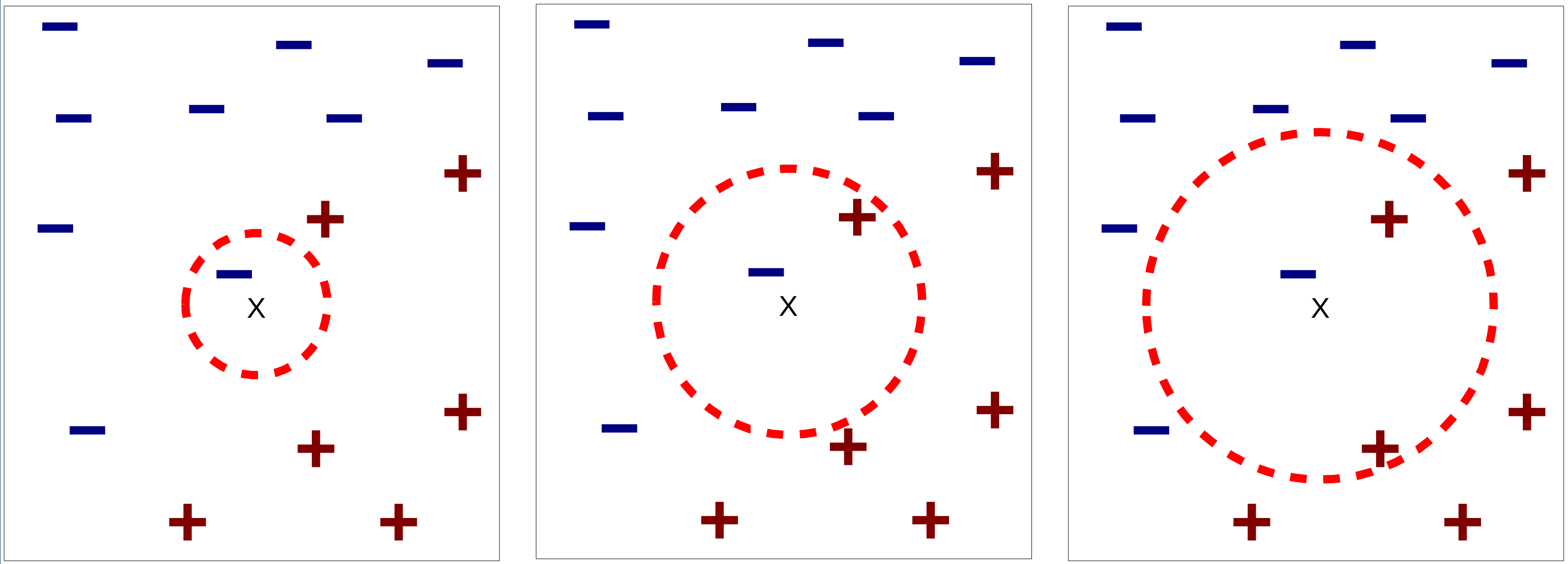
Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - 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)

Definition of Nearest Neighbor



(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***

How many parameters in kNN?

- A Linear Classifier

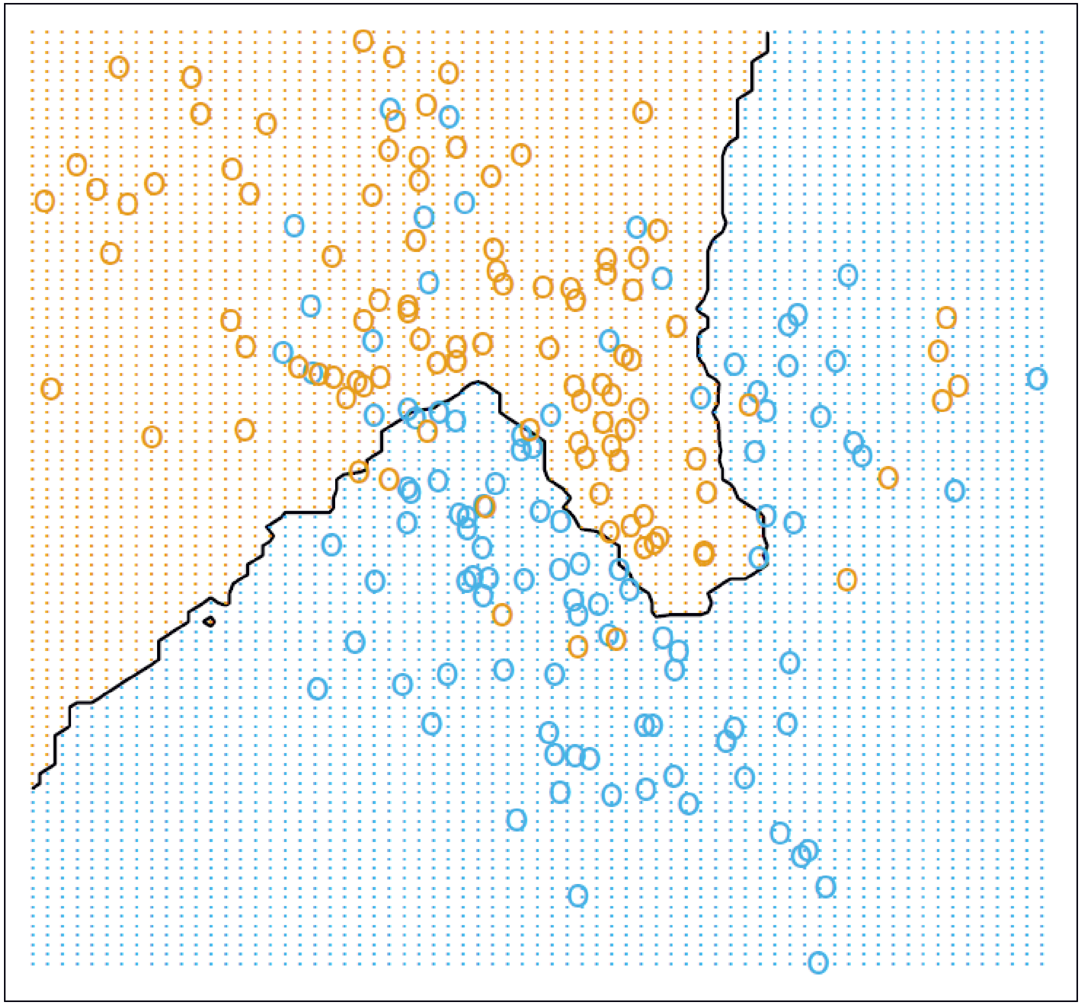
$$f(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x}$$

  - The number of parameters?

- kNN Classifier
  - Effective number of parameters?

$$\frac{N}{k}$$

15-Nearest Neighbor Classifier



1-Nearest Neighbor Classifier