

Artificial Intelligence

— — kNN and NB



Yanghui Rao

Assistant Prof., Ph.D

School of Data and Computer Science,

Sun Yat-sen University

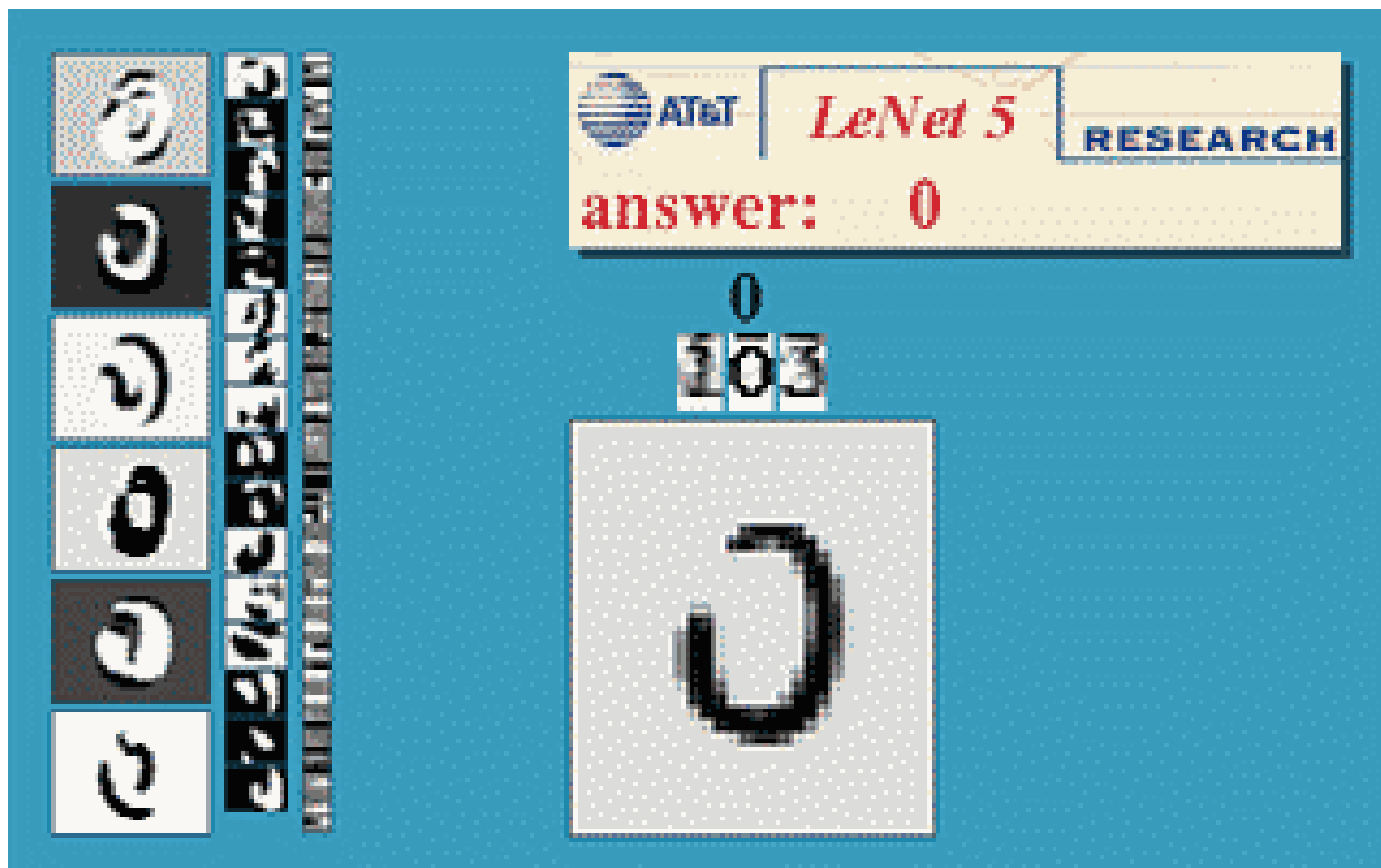
raoyangh@mail.sysu.edu.cn

Regression vs Classification

DocumentID	Words (split by space)	joy
train1	sheva delight us	0.6
train2	goal delight for sheva	0.7
test1	sheva goal	?

DocumentID	Words (split by space)	emotion
train1	sheva know us	not joy
train2	goal delight for sheva	joy
test1	sheva goal	?

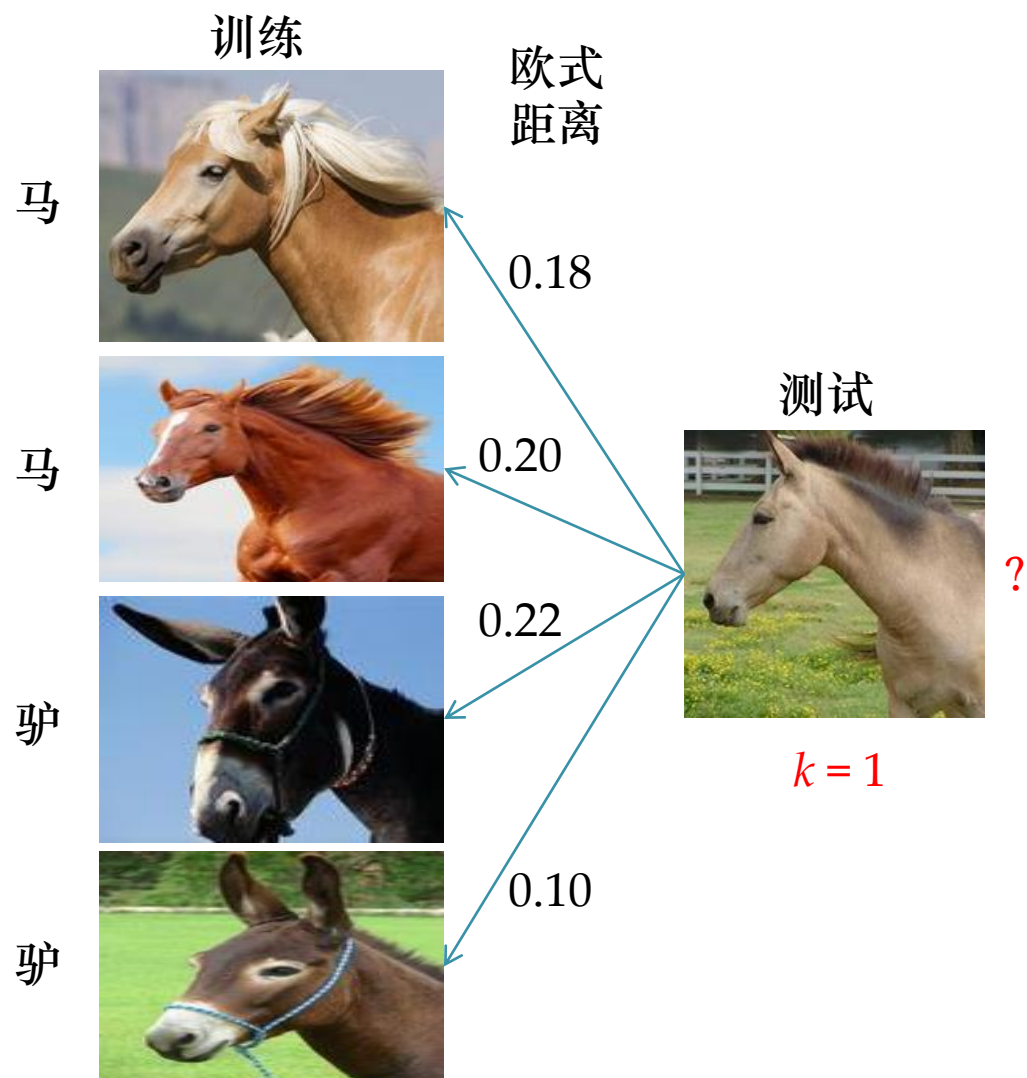
Regression vs Classification



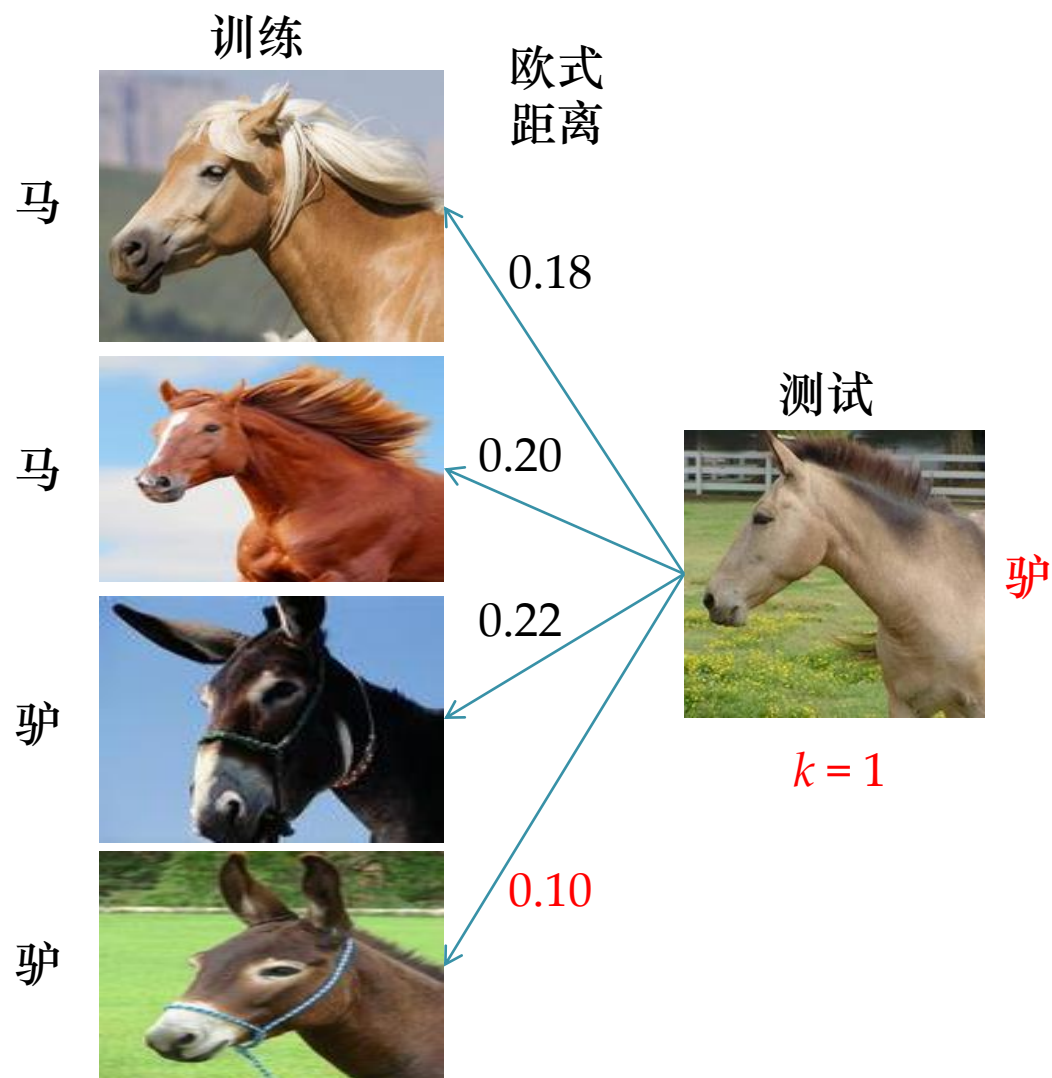
k -Nearest Neighbor

- All data/objects correspond to vectors in the n -D space (n 维空间的向量)
- The nearest neighbor could be defined in terms of Euclidean distance, etc.
- Target (目标) vector could be discrete- or real- valued
- For discrete-valued, k -NN returns the **most common** value (众数) among the k training examples nearest to X (测试)

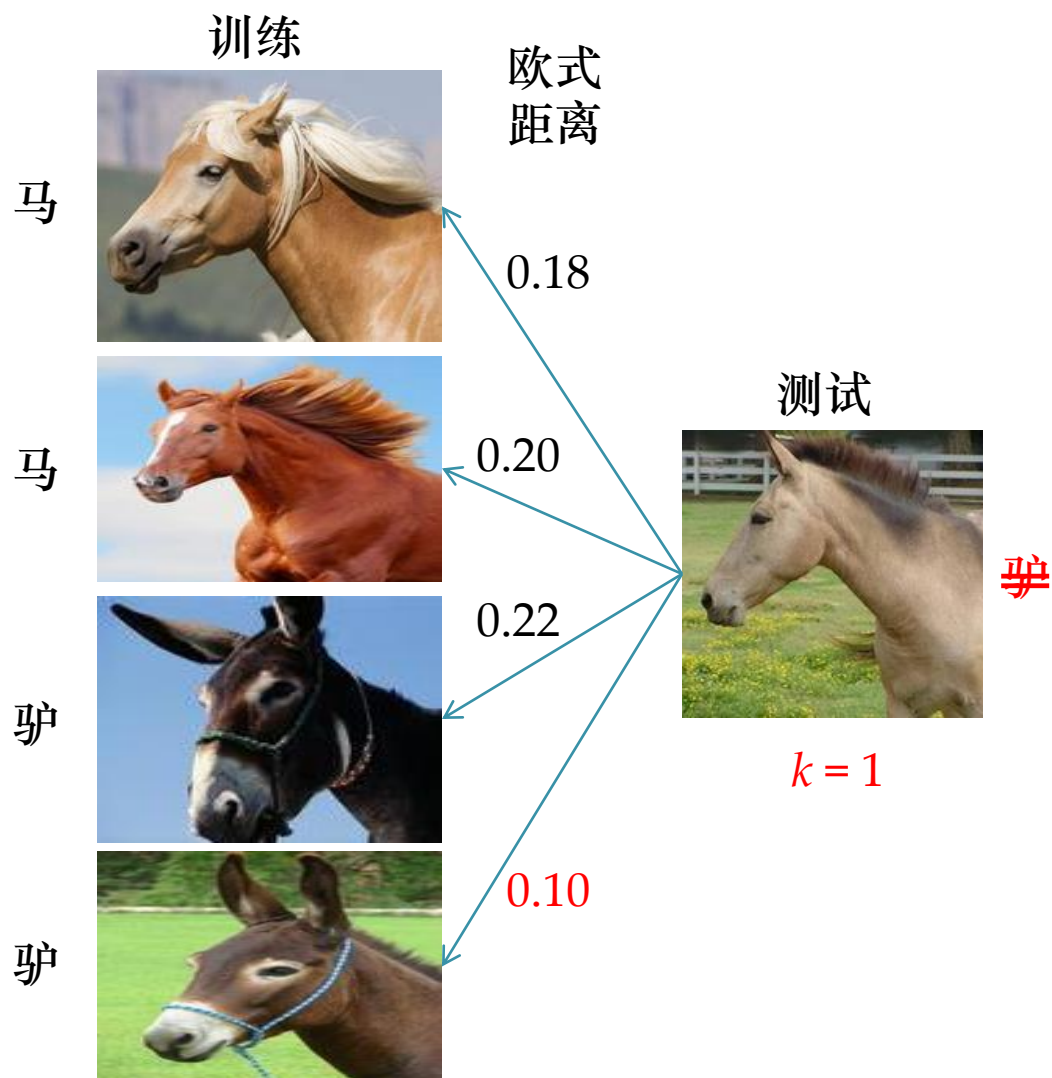
k -Nearest Neighbor



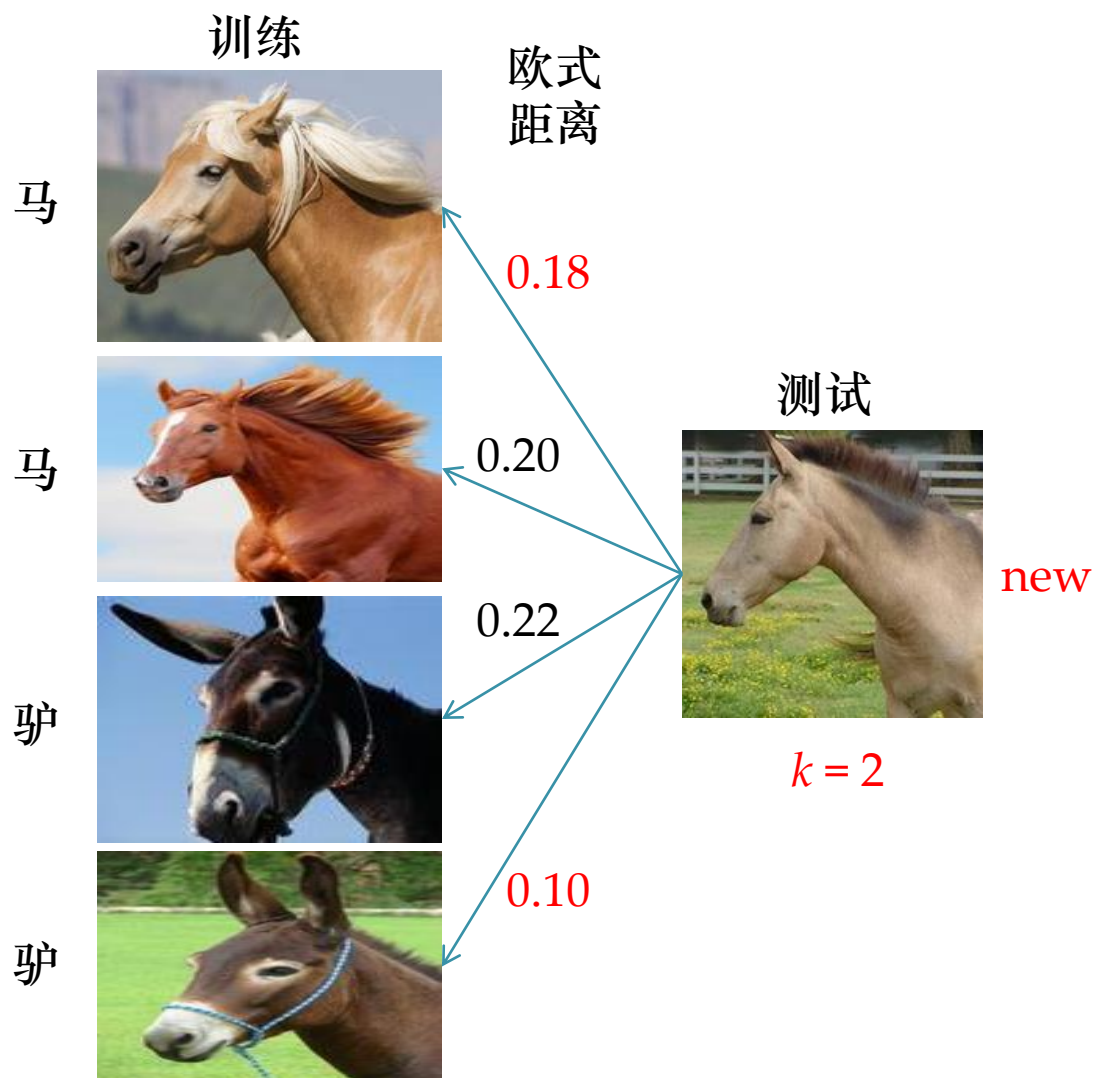
k -Nearest Neighbor



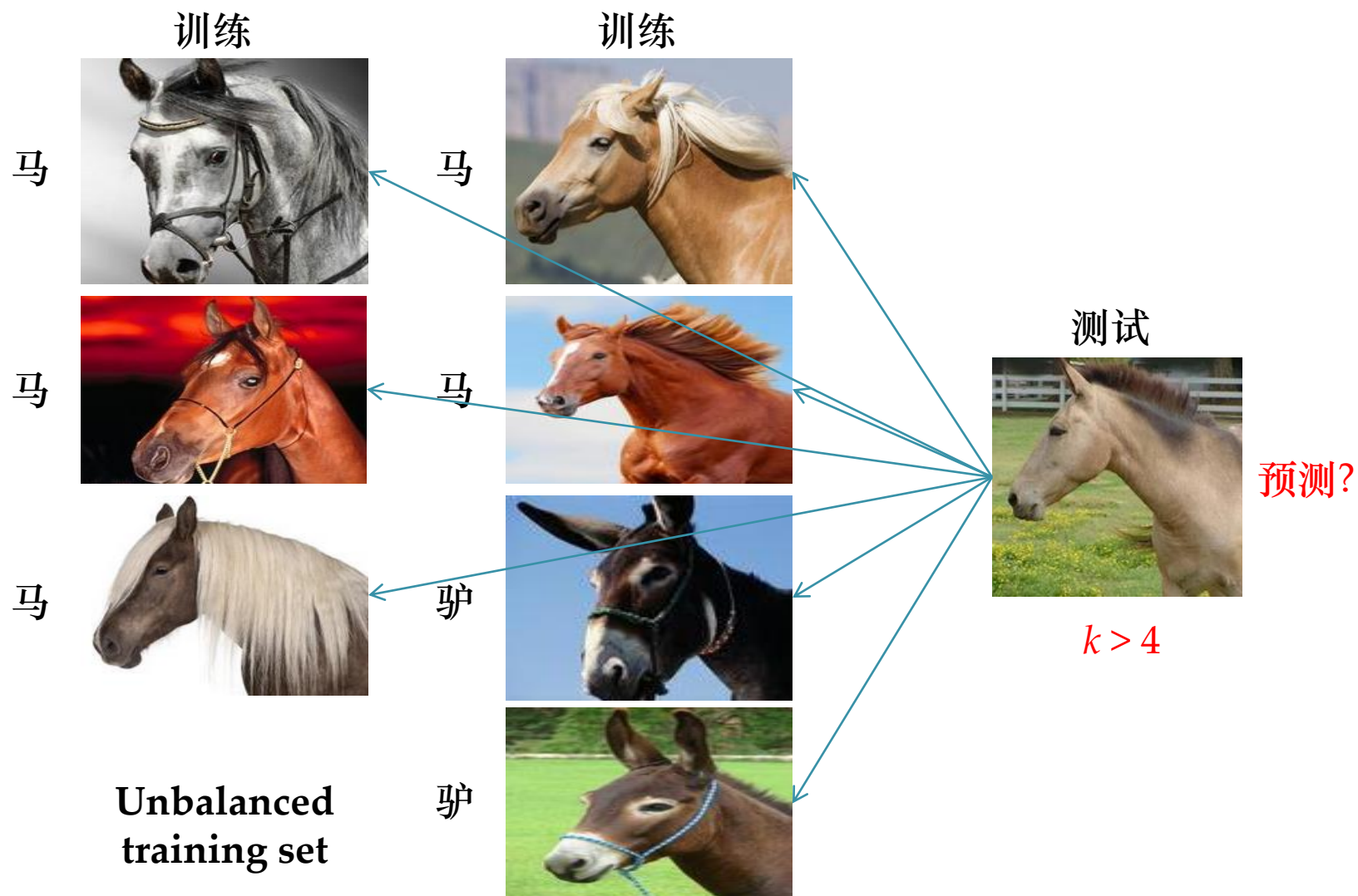
k -Nearest Neighbor



k -Nearest Neighbor



k -Nearest Neighbor

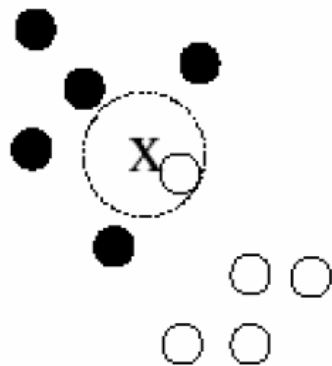


k -Nearest Neighbor

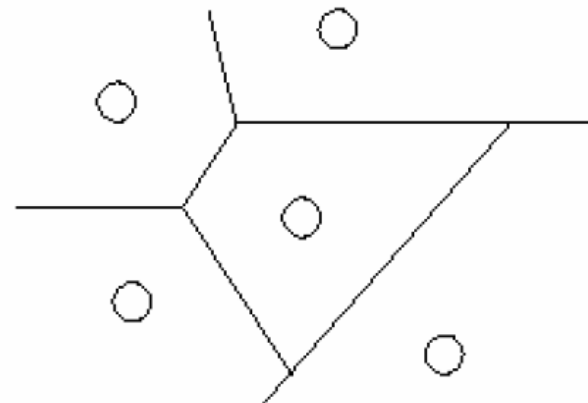
- k -NN for real-valued prediction for a given unknown X (测试)
 - Returns the **mean / median** gold values of the k nearest neighbors
- Instance-based learning/Lazy-learning
 - initially by Fix and Hodges (1951)
 - theoretical error bound analysis by Duda & Hart (1957)
 - store all the training samples
 - high computational cost for each new object if using the original k -NN algorithm

k -Nearest Neighbor

1-NN: assign "x" (new point) to the class of its nearest neighbor



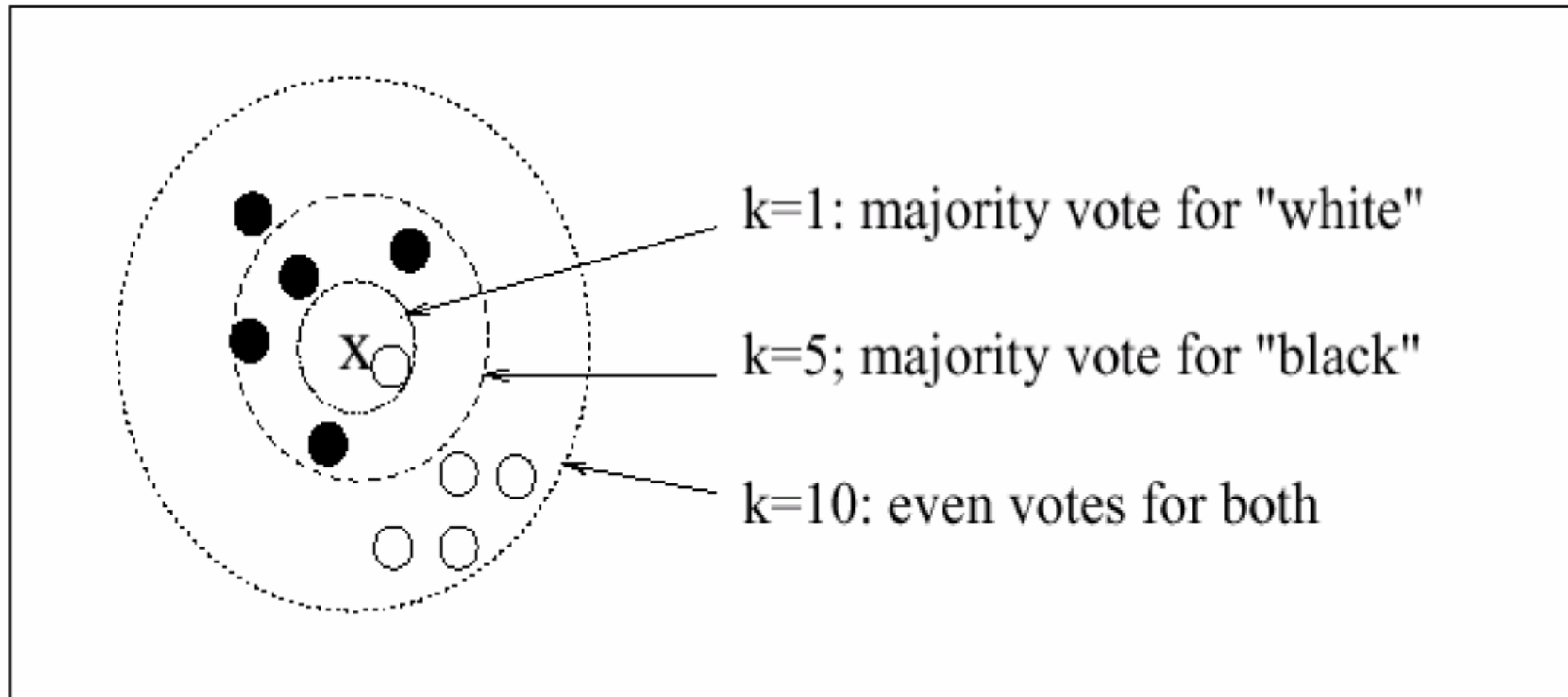
assign "x" to "white"



decision surface divided by points
("Voronoi diagram")

k -Nearest Neighbor

k -NN using a majority voting scheme



k -Nearest Neighbor

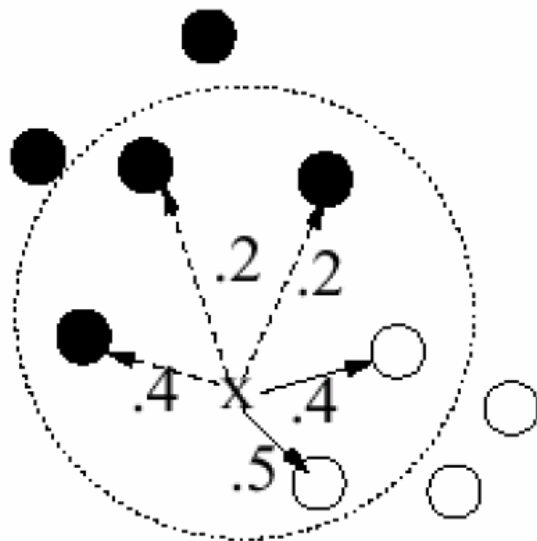
- Key aspects (影响因素) of k -NN

k -Nearest Neighbor

- Key aspects (影响因素) of k -NN
 - Similar item: We need a functional definition of similarity if we want to apply this automatically.
 - How many neighbors? (the value of k)
 - Does each neighbor get the same weight?
 - Count all classes for all neighbors? Or, use the frequently-occurred classes to make decisions?

k -Nearest Neighbor

k -NN using a weighted-sum voting scheme



kNN ($k = 5$)

Assign "white" to x because the weighted sum of "whites" is larger than the sum of "blacks".

Each neighbor is given a weight according to its nearness.

Naïve Bayesian

- A statistical model
 - Use Bayes' (贝叶斯) Theorem to perform probabilistic prediction, e.g., predict class membership probabilities
- Assumption
 - The effect of an attribute on a given class is independent of other attributes
- Performance
 - Comparable with decision trees (决策树) and selected neural network classifiers

Naïve Bayesian Classifier

- Given a training set of attributes and their associated class labels, and each object is represented by a n -D vector (n 维向量) $\mathbf{X} = (x_1, x_2, \dots, x_n)$
- Suppose there are m classes, *i.e.*, C_1, C_2, \dots, C_m .
- Naïve Bayesian Classifier is to derive the maximum posteriori (后验概率), *i.e.*, the maximal $P(C_i | \mathbf{X})$

Naïve Bayesian Classifier

- This can be derived from Bayes' theorem

$$P(C_i | \mathbf{X}) = \frac{P(\mathbf{X} | C_i)P(C_i)}{P(\mathbf{X})}$$

- Since $P(\mathbf{X})$ is constant for all classes, only

$$P(C_i | \mathbf{X}) \propto P(\mathbf{X} | C_i)P(C_i)$$

needs to be maximized

- $P(C_i)$ can be obtained from training set s_i/s

Derivation

- **Assumption:** attributes are conditionally independent (i.e., no dependence relation between attributes):
$$P(\mathbf{X} | C_i) = \prod_{k=1}^n P(x_k | C_i)$$
- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k | C_i) = s_{ik}/s_i$, count the distribution
- If A_k is continuous-valued, $P(x_k | C_i)$ can be computed based on Gaussian distribution

Example

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no
<=30	medium	yes	fair	?

Example

- $P(C_i)$: $P(\text{buys_computer} = \text{"yes"}) = 9/14 = 0.643$
 $P(\text{buys_computer} = \text{"no"}) = 5/14 = 0.357$

- Compute $P(X|C_i)$ for each class

$$P(\text{age} = \leq 30 \mid \text{buys_computer} = \text{"yes"}) = 2/9 = 0.222$$

$$P(\text{age} = \leq 30 \mid \text{buys_computer} = \text{"no"}) = 3/5 = 0.6$$

$$P(\text{income} = \text{"medium"} \mid \text{buys_computer} = \text{"yes"}) = 4/9 = 0.444$$

$$P(\text{income} = \text{"medium"} \mid \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$$

$$P(\text{student} = \text{"yes"} \mid \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$$

$$P(\text{student} = \text{"yes"} \mid \text{buys_computer} = \text{"no"}) = 1/5 = 0.2$$

$$P(\text{credit_rating} = \text{"fair"} \mid \text{buys_computer} = \text{"yes"}) = 6/9 = 0.667$$

$$P(\text{credit_rating} = \text{"fair"} \mid \text{buys_computer} = \text{"no"}) = 2/5 = 0.4$$

- $X = (\text{age} \leq 30, \text{income} = \text{medium}, \text{student} = \text{yes}, \text{credit_rating} = \text{fair})$

$$P(X|C_i) : P(X|\text{buys_computer} = \text{"yes"}) = 0.222 \times 0.444 \times 0.667 \times 0.667 = 0.044$$

$$P(X|\text{buys_computer} = \text{"no"}) = 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$$

$$P(X|C_i) \cdot P(C_i) : P(X|\text{buys_computer} = \text{"yes"}) \cdot P(\text{buys_computer} = \text{"yes"}) = 0.028$$

$$P(X|\text{buys_computer} = \text{"no"}) \cdot P(\text{buys_computer} = \text{"no"}) = 0.007$$

Comments

- **Advantages**

- Easy to implement
- Good results obtained in most of the cases

- **Disadvantages**

- Assumption: **class conditional independence** (给定每个类别, 条件独立), therefore loss of accuracy
 - Practically, dependencies do exist among variables, e.g., Symptoms: fever, cough, etc.
 - Dependencies among these cannot be modeled by Naïve Bayesian Classifier
- **0 probability value**: for example, new words in the testing document. Laplace smoothing factor?