Classification and Prediction

——Classification by Neural Networks——

徐华
清华大学 计算机系 智能技术与系统国家重点实验室
xuhua@tsinghua.edu.cn

1

Classification and Prediction

- Basic Concepts
- Issues Regarding Classification and Prediction
- Decision Tree
- Bayesian Classification
- Neural Networks
- Support Vector Machine
- K-Nearest Neighbor
- Associative classification
- Classification Accuracy

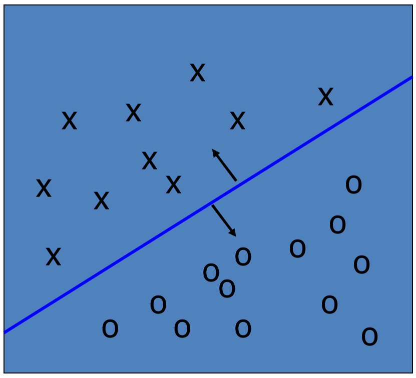
2

Classification: A Mathematical Mapping

- **Classification:**
  - predicts categorical class labels
- E.g., Personal homepage classification
  - $x_i = (x_1, x_2, x_3, \ldots)$, $y_i = +1$ or -1
  - x_1 : the number of a word "homepage"
  - x_2 : the number of a word "welcome"
- Mathematically
  - $x \in X = \Re^n$, $y \in Y = \{+1,-1\}$
  - We want a function f: X -> Y

3

Linear Classification



- Binary Classification problem
- The data above the blue line belongs to class 'x'
- The data below blue line belongs to class 'o'
- Examples:

  SVM, Perceptron,

4

Discriminative(判别式的) Classifiers

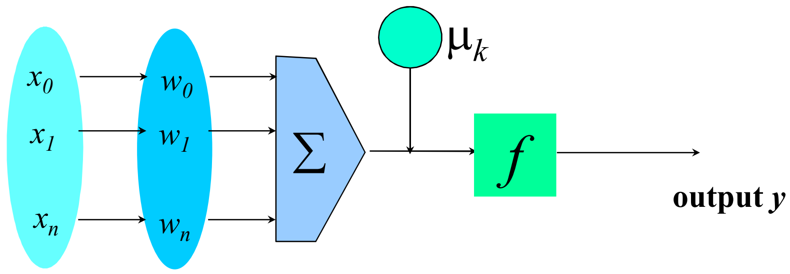
- Advantages
  - prediction accuracy is generally high
    - (as compared to Bayesian methods – in general)
  - robust, works when training examples contain errors
  - fast evaluation of the learned target function
    - (Bayesian networks are normally slow)
- Criticism
  - long training time
  - difficult to understand the learned function (weights)
    - (Bayesian networks can be used easily for pattern discovery)
  - not easy to incorporate domain knowledge
    - (easy in the form of priors on the data or distributions)

Neural Networks

- Analogy to Biological Systems (Indeed a great example of a good learning system)
- Massive Parallelism allowing for computational efficiency
- The first learning algorithm came in 1959 (Rosenblatt) who suggested that if a target output value is provided for a single neuron with fixed inputs, one can incrementally change weights to learn to produce these outputs using the perceptron learning rule

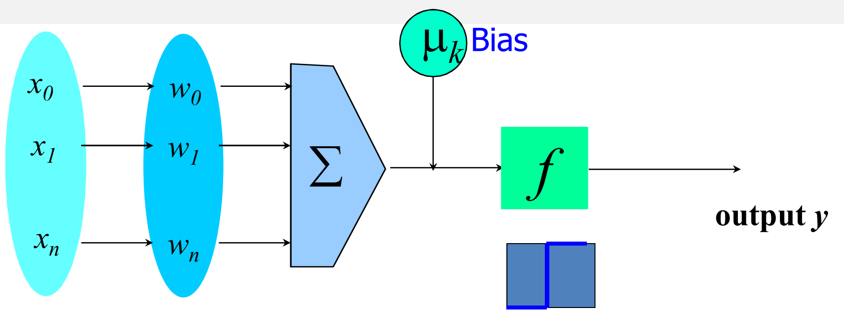
A Neuron (= a perceptron)

- The n-dimensional input vector x is mapped into variable y by means of the scalar product(内积) and a nonlinear function mapping



Input vector x | weight vector w | weighted sum | Activation Function(激活函数)

A Neuron (= a perceptron)



Input vector x | weight vector w | weighted sum | Activation function

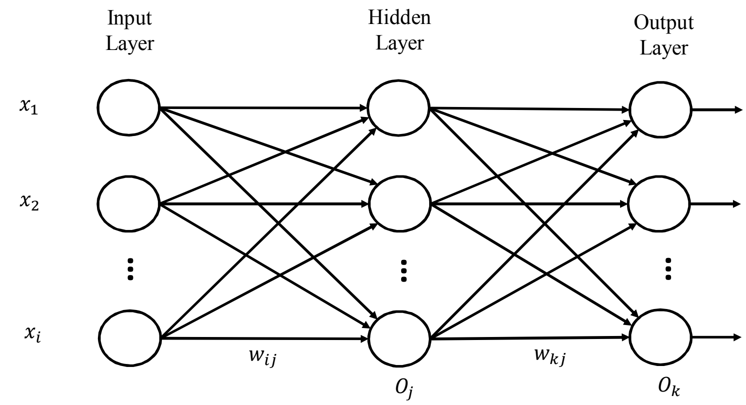
Logistic or sigmoid

For Example

$$y = \mathrm{sign}\left(\sum_{i=0}^{n} w_i x_i + \mu_k\right)$$

$$O = \frac{1}{1+e^{-I}}$$

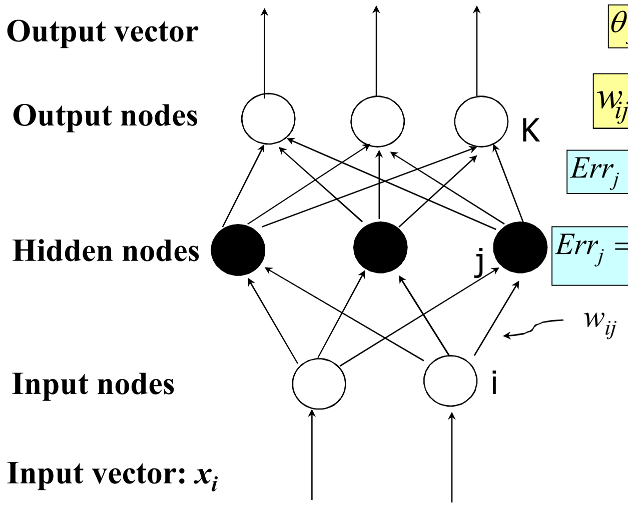
Multi layer Neural Network



Network Training

- The ultimate objective of training
  - obtain a set of weights that makes almost all the tuples in the training data classified correctly
- Steps
  - Initialize weights with random values
  - Feed the input tuples into the network one by one
  - For each unit
    - Compute the net input to the unit as a linear combination of all the inputs to the unit
    - Compute the output value using the activation function
    - Compute the error
    - Update the weights and the bias

Back Propagation



$$\theta_j = \theta_j + (l)Err_j$$

$$w_{ij} = w_{ij} + (l)Err_j O_i$$

$$Err_j = O_j(1-O_j)(T_j - O_j)$$

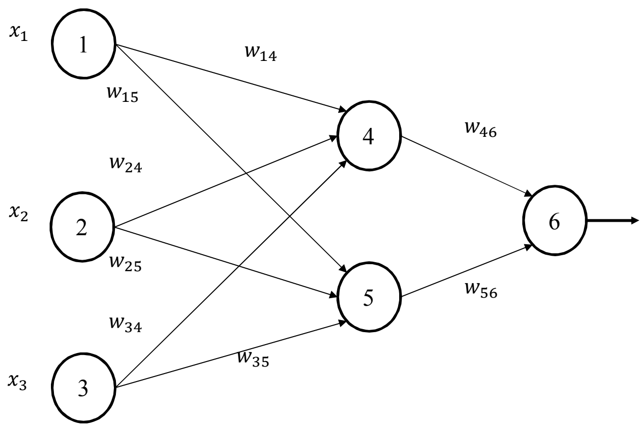
$$Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$$

$$O_j = \frac{1}{1+e^{-I_j}}$$

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

11

Back Propagation



12

One example (1)

- **Input:** X={1,0,1}, output: 1
  - x1=1, x2=0, x3=1; w14=0.2, w15=-0.3;w24=0.4, w25=0.1; w34=-0.5, w35=0.2; w46=-0.3; w56=-0.2,
- **Bias:** node 4:-0.4, node 5: 0.2, node 6: 0.1
- **Learning rate l=0.9**
- **Node 4:**
  - input :w14*x1+w24*x2+w34*x3+bias of node 4=1*0.2+0.4*0-0.5*1-0.4=-0.7
  - output: O4=0.332 $O_j = \frac{1}{1+e^{-I_j}}$
- **The same:** node 5: input: 0.1, output:0.525
- **Node 6:**
  - input :w46*o4+w56*o5+bias of node 6=-0.3*0.332-0.2*0.525+0.1=-0.105
  - output:0.474

One example (2)

- **Node 6:** $Err_j = O_j(1-O_j)(T_j - O_j)$

  0.474*(1-0.474)*(1-0.474)=0.1311
- **Node 5:** $Err_j = O_j(1-O_j)\sum_k Err_k w_{jk}$

  0.525*(1-0.525)*0.1311*(-0.2)=-0.0065
- **Node 4:-0.0087**
  - **W_{46}:** $w_{ij} = w_{ij} + (l) Err_j O_i$

    -0.3+(0.9)(0.1311)(0.332)=-0.261
- **Other W_{ij} is the same with w_{46}**
- **Bias of node 6:** $\theta_j = \theta_j + (l) Err_j$
  - **0.1+(0.9)*(0.1311)=0.218**
- **Other bias is the same with node 6**

Network Pruning and Rule Extraction

- **Network pruning**
  - Fully connected network will be hard to articulate
  - *N* input nodes, *h* hidden nodes and *m* output nodes lead to *h(m+N)* weights
  - Pruning: Remove some of the links without affecting classification accuracy of the network
- **Extracting rules : replace individual activation value by the cluster average trained network**
  - Discretize activation values maintaining the network accuracy
  - Enumerate the output from the discretized activation values to find rules between activation value and output
  - Find the relationship between the input and activation value
  - Combine the above two to have rules relating the output to input

15

Thanks !

16