Brief Introduction of Machine Learning Techniques for Content Analysis

Wei-Ta Chu

2010/11/4
Announcement

- Midterm will be held in 8:45~11:30, Nov. 18, 2010
- Range: Starting from video syntax analysis to machine learning techniques
Outline

- Overview
- Gaussian Mixture Model (GMM)
- Hidden Markov Model (HMM)
- Support Vector Machine (SVM)
Overview

- Any computer program that can improve its performance at some task through experience (or training) can be called a learning program.

- During early days, computer scientists developed learning algorithms based on heuristics and insights into human reasoning mechanisms.
  - Decision tree, …

- Neuro-scientists attempted to devise learning methods by imitating the structure of human brains.


Multimedia Content Analysis, CSIE, CCU
Many learning tasks can be formulated as one of the following two problems.

Regression: $X$: input variable, $Y$: output variable. Infer a function $f(X)$ so that given a value of $x$ of the input variable $X$, $y = f(x)$ is a good predication of the true value $y$ of the output variable $Y$. 
**Basic Statistical Learning Problems**

- **Classification**: Assume that a random variable $X$ can belong to one of a finite set of classes $C = \{1, 2, \ldots, K\}$. Given the value $x$ of variable $X$, infer its class label $l = g(x)$, where $l \in C$.

- It is also of great interest to estimate the probability $P(k|x)$ that $X$ belongs to class $k$, $k \in C$.

- In fact both the regression and classification problems can be formulated using the same framework.
Categorizations of Machine Learning Techniques

- **Unsupervised vs. Supervised**
  - For inferring the functions $f(x)$ and $g(x)$, if pairs of training data $(x_i, y_i)$ or $(x_i, l_i)$, $i = 1, \ldots, N$ are available, then the inference process is called *supervised learning*.
  - Most regression methods are supervised learning.
  - Unsupervised methods strive to automatically partition a given data set into the predefined number of clusters – also called *clustering*.
Categorizations of Machine Learning Techniques

- **Generative Models vs. Discriminative Models**
  - Discriminative models strive to learn $P(k|x)$ directly from the training set without the attempt to modeling the observation $x$.
  - Generative models compute $P(k|x)$ by first modeling the class-conditional probabilities $P(x|k)$ as well as the class probabilities $P(k)$

$$P(k|x) \propto P(x|k)P(k)$$

Posterior prob. likelihood Priori prob.

Multimedia Content Analysis, CSIE, CCU
Categorizations of Machine Learning Techniques

- Generative models:
  - Naïve Bayes, Bayesian Networks, Gaussian Mixture Models (GMM), Hidden Markov Models (HMM), …

- Discriminative models:
  - Neural Networks, Support Vector Machines (SVM), Maximum Entropy Models (MEM), Conditional Random Fields (CRF), …
Categorizations of Machine Learning Techniques

- **Models for Simple Data vs. Models for Complex Data**

  - Complex data: consist of sub-entities that are strongly related one to another
    - E.g. a beach scene usually composed of a blue sky on top, an ocean in the middle, and a sand beach at the bottom
  
  - For simple: Naïve Bayes, GMM, NN, SVM
  
  - For complex: BN, HMM, MEM, CRF, M$^3$-net
Categorizations of Machine Learning Techniques

- Model Identification vs. Model Prediction
- Model identification: to discover an existing Law of Nature
- The model identification paradigm is an ill-posed problems, and is annoyed by the curse of dimensionality.
- The goal of model predication is to predict events well, but not necessarily through the identification of the model of events.
Gaussian Mixture Model

Wei-Ta Chu

2010/11/4
By using a sufficient number of Gaussians, and by adjusting their means and covariances as well as the coefficients in the linear combinations, almost any continuous density can be approximated to arbitrary accuracy.

C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006
Consider a superposition of $K$ Gaussian densities

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x | \mu_k, \Sigma_k)$$

Each Gaussian density $\mathcal{N}(x | \mu_k, \Sigma_k)$ is called a component of the mixture and has its own mean $\mu_k$ and covariance $\Sigma_k$. 
Introduction

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x | \mu_k, \Sigma_k)$$

- From the sum and product rules, the marginal density is given by
  $$p(x) = \sum_{k=1}^{K} p(k) p(x | k)$$

- We can view $\pi_k = p(k)$ as the prior probability of picking the $k$th component, and the density as the probability of $x$ conditioned on $k$:
  $$\mathcal{N}(x | \mu_k, \Sigma_k) = p(x | k)$$
From Baye’s theorem, the posterior probability $p(k|x)$ is given by

$$p(k|x) = \frac{p(k)p(x|k)}{\sum_l p(l)p(x|l)} = \frac{\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)}{\sum_l \pi_l \mathcal{N}(x|\mu_l, \Sigma_l)}$$

Gaussian mixture distribution is governed by parameters $\pi$, $\mu$, and $\Sigma$. One way to set these parameters is to use maximum likelihood.

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\}$$

Assume that different mixtures are independent and identically distributed.
In case of a single variable \( x \), the Gaussian distribution is in the form

\[
\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2}(x - \mu)^2 \right\}
\]

For a \( D \)-dimensional vector \( x \), the multivariate Gaussian distribution takes the form

\[
\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2}(x - \mu)^T\Sigma^{-1}(x - \mu) \right\}
\]
Maximizing Likelihood

\[ \ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n|\mu_k, \Sigma_k) \right\} \]

- Setting the derivative of \( \ln p(X|\pi, \mu, \Sigma) \) with respect to the means \( \mu \) of the Gaussian components to zero

\[ 0 = -\sum_{n=1}^{N} \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j} \pi_j N(x_n|\mu_j, \Sigma_j)} \sum_{k} (x_n - \mu_k) \gamma(z_{nk}) \]

\[ \mu_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \quad N_k = \sum_{n=1}^{N} \gamma(z_{nk}) \]

- The mean \( \mu_k \) for the \( k \)th Gaussian component is obtained by taking a weighted mean of all of the points in the data set, in which the weighting factor for data point \( x_n \) is given by the posterior probability \( \gamma(z_{nk}) \)
Maximizing Likelihood

- Setting the derivative of $\ln p(X | \pi, \mu, \Sigma)$ with respect to the covariance $\Sigma_k$ of the Gaussian components to zero

$$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k)(x_n - \mu_k)^T$$

- Each data point weighted by the corresponding posterior probability
Maximizing Likelihood

- Maximize $\ln p(X|\pi, \mu, \Sigma)$ with respect to the mixing coefficients $\pi_k$
- Constraint: the sum of mixing coefficients is one
- Using Lagrange multiplier and maximizing the following quantity

$$\ln p(X|\pi, \mu, \Sigma) + \lambda(\sum_{k=1}^{K} \pi_k - 1) \Rightarrow 0 = \sum_{n=1}^{N} \frac{N(x_n|\mu_k, \Sigma_k)}{\sum_j \pi_j N(x_n|\mu_j, \Sigma_j)} + \lambda$$

- If we multiply both sides by $\pi_k$ and sum over $k$ making use of the constraint $\sum_{k=1}^{K} \pi_k = 1$, we find $\lambda = -N$. Using this to eliminate $\lambda$ and rearranging we obtain $\pi_k = \frac{N_k}{N}$

- Mixing coefficient of the $k$th component is given by the average responsibility which that component takes for explaining the data points
We first choose some initial values for the means, covariances, and mixing coefficients.

Expectation step (E step)
- Use the current parameters to evaluate the posterior probabilities

Maximization step (M step)
- Re-estimate the means, covariances, and mixing coefficients

Each update to the parameters resulting from an E step followed by an M step is guaranteed to increase the log likelihood function.
Example
EM for Gaussian Mixtures

Given a Gaussian mixture model, the goal is to maximize the likelihood function with respect to the parameters (comprising the means and covariances of the components and the mixing coefficients).

1. Initialize the means $\mu_k$, covariances $\Sigma_k$ and mixing coefficients $\pi_k$, and evaluate the initial value of the log likelihood.

2. E step. Evaluate the responsibilities using the current parameter values

$$
\gamma(z_{nk}) = \frac{\pi_k N(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n | \mu_j, \Sigma_j)}.
$$

3. M step. Re-estimate the parameters using the current responsibilities

$$
\mu_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n
$$

$$
\Sigma_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) (x_n - \mu_k^{\text{new}})(x_n - \mu_k^{\text{new}})^T
$$

$$
\pi_k^{\text{new}} = \frac{N_k}{N}
$$

where

$$
N_k = \sum_{n=1}^{N} \gamma(z_{nk}).
$$

4. Evaluate the log likelihood

$$
\ln p(X | \mu, \Sigma, \pi) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k) \right\}
$$

and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.
Case Study

Figure 2: Flowchart of segmentation Algorithm


Multimedia Content Analysis, CSIE, CCU
Case Study

- The condition density of a pixel $\xi$ belongs to the playfield region $\Phi$ is modeled with $M$ Gaussian densities: $p(\xi|\Phi) = \sum_{i=1}^{M} w_i b_i(\xi)$

Figure 4: Segmentation results of some frames
Related Resources

- **GMMBAYES - Bayesian Classifier and Gaussian Mixture Model ToolBox**

- **Netlab**
  - [http://www.ncrg.aston.ac.uk/netlab/index.php](http://www.ncrg.aston.ac.uk/netlab/index.php)

- **Matlab toolboxes collection**
  - [http://stommel.tamu.edu/~baum/toolboxes.html](http://stommel.tamu.edu/~baum/toolboxes.html)
Hidden Markov Model

Wei-Ta Chu

2010/11/4
Assume that each of the condition distributions is independent of all previous observations except the most recent, we obtain the *first-order Markov chain*.

\[
p(x_1, \ldots, x_N) = \prod_{n=1}^{N} p(x_n|x_1, \ldots, x_{n-1})
\]

\[
p(x_1, \ldots, x_N) = p(x_1) \prod_{n=2}^{N} p(x_n|x_{n-1})
\]

\[
p(x_n|x_1, \ldots, x_{n-1}) = p(x_n|x_{n-1})
\]

**First-order Markov chain**

**Second-order Markov chain**

What’s the probability that the weather for eight consecutive days is “sun-sun-sun-rain-rain-sun-cloudy-sun”?

\[
P(O|Model) = P[3, 3, 3, 1, 1, 3, 2, 3|Model]
\]

\[
\]

\[
= \pi_3(a_{33})^2a_{31}a_{11}a_{13}a_{32}a_{23}
\]

\[
= (1.0)(0.8)^2(0.1)(0.4)(0.3)(0.1)(0.2)
\]

\[
= 1.536 \times 10^{-4}
\]


Coin-Toss Model

- You are in a room with a curtain through which you cannot see that is happening. On the other side of the curtain is another person who is performing a coin tossing experiment (using one or more coins). The person will not tell you which coin he selects at any time; he will only tell you the result of each coin flip.

- A typical observation sequence would be

$$O = (o_1 o_2 o_3 \cdots o_T) = (HHTTTHHTTH \cdots H)$$

- The question is: how do we build an model to explain the observed sequence of head and tails?
Coin-Toss Model

1-coin model
(Observable Markov Model)

\[ O = (HHTTTTHHHTTH) \]
\[ S = (1122221221) \]

2-coins model
(Hidden Markov Model)

\[ O = (HHTTTTHHHTTH) \]
\[ S = (2112221221) \]
Coin-Toss Model

3-coins model
(Hidden Markov Model)

\[ O = (HHTTTTHHTTH) \]
\[ S = (3123311231) \]

State 1
- P(H) = \( P_1 \)
- P(T) = 1 - \( P_1 \)

State 2
- P(H) = \( P_2 \)
- P(T) = 1 - \( P_2 \)

State 3
- P(H) = \( P_3 \)
- P(T) = 1 - \( P_3 \)
Elements of HMM

- $N$: the number of states in the model
- $M$: the number of distinct observation symbols per state
- The state-transition probability $A = \{a_{ij}\}$
  \[ a_{ij} = P[q_{t+1} = j | q_t = i] \quad 1 \leq i, j \leq N \]
- The observation symbol probability distribution $B = \{b_j(k)\}$
  \[ b_j(k) = P[o_t = v_k | q_t = j] \quad 1 \leq k \leq M \]
- The initial state distribution
  \[ \pi_i = P[q_1 = i] \quad 1 < i < N \]
- To describe an HMM, we usually use the compact notation
  \[ \lambda = (A, B, \pi) \]
Three Basic Problems of HMM

Problem 1: Probability Evaluation

Given the observation sequence \( O = (o_1 o_2 \cdots o_T) \), and a model \( \lambda = (A, B, \pi) \), how do we efficiently compute \( P(O|\lambda) \), the probability of the observation sequence, given the model?

How do we compute the probability that the observed sequence was produced by the model?

Scoring how well a given model matches a given observation sequence.
Three Basic Problems of HMM

- **Problem 2: Optimal State Sequence**

  Given the observation sequence \( O = (o_1 o_2 \cdots o_T) \), and the model \( \lambda \), how do we choose a corresponding state sequence \( q = (q_1 q_2 \cdots q_T) \) that is optimal in some sense (i.e., best explains the observations).

  Attempt to uncover the hidden part of the model – that is, to find the “correct” state sequence.

  For practical situations, we usually use an optimality criterion to solve this problem as best as possible.
Three Basic Problems of HMM

Problem 3: Parameter Estimation

How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$

Attempt to optimize the model parameters to best describe how a given observation sequence comes about.

The observation sequence used to adjust the model parameters is called a training sequence because it is used to “train” the HMM.
Solution to Problem 1

Given the observation sequence \( O = (o_1 o_2 \cdots o_T) \), and a model \( \lambda = (A, B, \pi) \), how do we efficiently compute \( P(O|\lambda) \), the probability of the observation sequence, given the model?

- There are \( N^T \) possible state sequences.
- Consider one fixed-state sequence \( q = (q_1 q_2 \cdots q_T) \).
- The prob. of the observation sequence given the state sequence \( q \)
  \[
P(O|q, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda)
\]
- Where we have assumed statistical independence of observations, thus we get
  \[
P(O|q, \lambda) = b_{q_1}(o_1) \cdot b_{q_2}(o_2) \cdots b_{q_T}(o_T)
\]
Solution to Problem 1

- The prob. of such state sequence can be written as
  \[ P(q|\lambda) = \pi_{q_1} a_{q_1q_2} a_{q_2q_3} \cdots a_{q_{T-1}q_T} \]

- The joint prob. of \(O\) and \(q\), i.e., the prob. that \(O\) and \(q\) occur simultaneously, is simply the product of the above terms
  \[ P(O, q|\lambda) = P(O|q, \lambda) P(q|\lambda) \]

- The prob. of \(O\) is obtained by summing this joint prob. over all possible state sequences \(q\), giving
  \[ P(O|\lambda) = \sum_{allq} P(O|q, \lambda) P(q|\lambda) \]
  \[ = \sum_{q_1, q_2, \ldots, q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1q_2} b_{q_2}(o_2) \cdots a_{q_{T-1}q_T} b_{q_T}(o_T) \]
Solution to Problem 1

- **The Forward Procedure**
  - The prob. of the partial observation sequence \( o_1, o_2, \ldots, o_t \) (until time \( t \)) and state \( i \) at time \( t \), given the model \( \lambda \)
    
    \[
    \alpha_t(i) = P(o_1 o_2 \cdots o_t, q_t = i | \lambda)
    \]
  - We solve for it inductively
  - 1. Initialization \( \alpha_1(i) = \pi_i b_i(o_1) \quad 1 \leq i \leq N \)
  - 2. Induction
    
    \[
    \alpha_{t+1}(j) = \left[ \sum_{i=1}^{N} \alpha_t(i) \alpha_{ij} \right] b_j(o_{t+1}) \quad 1 \leq t < T - 1 \quad 1 \leq j \leq N
    \]
  - 3. Termination
    
    \[
    P(O | \lambda) = \sum_{i=1}^{N} \alpha_T(i)
    \]
Solution to Problem 1

- **The Forward Procedure**

- Require on the order of $N^2T$ calculations, rather than $2TN^T$ as required by the direction calculation.
The Backward Procedure

The prob. of partial observation sequence from $t+1$ to the end, given state $i$ at time $t$ and the model $\lambda$

$$\beta_t(i) = P(o_{t+1}o_{t+2}\cdots o_T|q_t = i, \lambda)$$

- 1. Initialization $\beta_T(i) = 1 \quad 1 \leq i \leq N$
- 2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}b_j(o_{t+1})\beta_{t+1}(j)$$

$t = T - 1, T - 2, \ldots, 1$

$1 \leq i \leq N$
Solution to Problem 2

Given the observation sequence $O = (o_1 o_2 \cdots o_T)$, and the model $\lambda$, how do we choose a corresponding state sequence $q = (q_1 q_2 \cdots q_T)$ that is optimal in some sense (i.e., best explains the observations)?

- We define the quantity

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P[q_1 q_2 \cdots q_{t-1}, q_t = i, o_{t+1} o_{t+2} \cdots o_t | \lambda]$$

- Which is the best score (highest probability) along a single path, at time $t$, which accounts for the first $t$ observations and ends in state $i$. By induction we have

$$\delta_{t+1}(i) = [\max_j \delta_t(i) a_{i,j}] \cdot b_j(o_{t+1})$$
The Viterbi Algorithm

1. Initialization
\[ \delta_1(i) = \pi_i b_i(o_1) \quad \psi_1(i) = 0 \]

2. Recursion
\[ \delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t) \quad 1 \leq j \leq N \]
\[ \psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \quad 2 \leq t \leq T \]

3. Termination
\[ P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \]
\[ q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \]

4. Path (state sequence) backtracking
\[ q_t^* = \psi_{t+1}(q_{t+1}^*) \quad t = T - 1, T - 2, \ldots, 1 \]

The major difference between Viterbi and the forward procedure is the maximization over previous states.
Solution to Problem 3

How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$

- Choose $\lambda = (A, B, \pi)$ such that its likelihood, $P(O|\lambda)$, is locally maximized using an iterative procedure such as the Baum-Welch algorithm (also known as EM algorithm or forward-backward algorithm)

- Define the prob. of being in state $i$ at time $t$, and state $j$ at time $t+1$, given the model and the observation sequence.

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | O, \lambda)$$
Solution to Problem 3

\[ \xi_t(i, j) = P(q_t = i, q_{t+1} = j | O, \lambda) \]

\[ \xi_t(i, j) = \frac{P(q_t = i, q_{t+1} = j, O | \lambda)}{P(O | \lambda)} = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O | \lambda)} \]

\[ = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)} \]

\[ \gamma_t(i) = P(q_t = i | O, \lambda) \]

The prob. of being in state \( i \) at time \( t \), given the observation sequence \( O \) and the model

\[ \gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j) \]
\[
\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from state } i \text{ in } O
\]

\[
\sum_{t=1}^{T-1} \xi_t(i) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } O
\]

\[
\bar{\pi}_i = \text{expected frequency (number of times) in state } i \text{ at time } (t = 1) = \gamma_1(i)
\]

\[
\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}
\]

\[
\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j} = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(i)}
\]

Multimedia Content Analysis, CSIE, CCU
Types of HMMs

- Ergodic
- Left-right
- Parallel path left-right

Fig. 7. Illustration of 3 distinct types of HMMs. (a) A 4-state ergodic model. (b) A 4-state left-right model. (c) A 6-state parallel path left-right model.
Case Study

- **Features**
  - Field descriptor
  - Edge descriptor
  - Grass and sand
  - Player height

![Diagram](image)

**Fig. 2.** (a) HMM model for nice hits (b) HMM model for nice catches (c) HMM model for home runs (d) HMM model for plays within the diamond

Related Resources

- Hidden Markov Model (HMM) Toolbox for Matlab

- The General Hidden Markov Model library (GHMM)
  - http://ghmm.sourceforge.net/

- HTK Speech Recognition Toolkit
  - http://htk.eng.cam.ac.uk
Support Vector Machine

Wei-Ta Chu

2010/11/4
Consider the set of training data that consist of two classes $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$, $y_i \in \{-1, 1\}$, $\mathbf{x}_i \in \mathbb{R}^d$. A linear classifier able to separate the positive from the negative examples will be a hyperline in $\mathbb{R}^d$ characterized by a normal $\mathbf{w}$ and an offset $b$:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

For a linearly separable data set $S$, there exists a hyperplane that satisfies all the points in $S$

$$\mathbf{w}^T \mathbf{x}_i + b \geq 0 \text{ for } y_i = +1$$
$$\mathbf{w}^T \mathbf{x}_i + b < 0 \text{ for } y_i = -1$$
$$y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 0 \forall i$$

Rescale $(w, b)$ so that the closest points to the hyperplane satisfy $|w^T x_i + b| = 1$. This normalization leads to the canonical form for SVM:

$$y_i(w^T x_i + b) \geq 1 \quad \forall i$$
Properties

1. For any two points $x_1$ and $x_2$ lying on the hyperplane, $w^T(x_1 - x_2) = 0$. Therefore, $w$ is the vector normal to the surface of the hyperplane.

2. For any point $x_h$ on the hyperplane, $w^T x_h = -b$.

3. The signed distance from a point $x$ to the hyperplane is given by

$$d = \frac{w}{||w||}(x - x_0)$$

$$= \frac{1}{||w||}(w^T x - w^T x_0)$$

$$= \frac{1}{||w||}(w^T x + b) \quad (10.11)$$

where $x_0$ is the intersection point between the normal vector $w$ and the hyperplane. Since $x_0$ lies on the hyperplane, it satisfies the equality...
Properties

4. The perpendicular distance from the hyperplane to the origin equals $\|b\|/\|w\|$ (see Problem 10.2 at the end of the chapter).

5. The points for which the equality in (10.10) holds are those points that lie on the hyperplanes $w^T \mathbf{x} + b = \pm 1$ (denoted as $H_1$, $H_2$, respectively), and have the perpendicular distance $1/\|w\|$ to the separating hyperplane $w^T \mathbf{x} + b = 0$.

$$d^+ + d^- = \frac{1}{\|w\|} + \frac{1}{\|w\|} = \frac{2}{\|w\|}$$
The goal of SVM is to find the pair of hyperplanes $H_1$, $H_2$ that maximize the margin, subject to the constraints

$$y_i(w^T x_i + b) \geq 1 \quad \forall i$$

This can be formulated as the constrained optimization problem:

$$\min_{w,b} \frac{1}{2} w^T w,$$

subject to $$y_i(w^T x_i + b) \geq 1 \quad \forall i$$
When the SVM derived above is applied to non-separable data sets, some data points could be at a distance $\xi_i/\|w\|$ on the wrong side.

We transform the constraint to

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i$$

The SVM for non-separable case can be casted as the optimization problem:

$$\min_{w,b} \frac{1}{2} w^T w + \frac{C}{n} \sum_{i=1}^{n} \xi_i,$$

subject to $y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i$
General Idea

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

\[ \Phi: x \mapsto \varphi(x) \]
Optimization Problem

- Use Lagrange multiplier method to find the optimal solution – transform the original problem to a dual problem
- Kernel trick
  - We don’t really find the mapping function, but to use a “kernel function” to find the inner product of data points in the higher-dimensional space.
Related Resources

- LIBSVM
References


- Statistical Data Mining Tutorials, Tutorial Slides by Andrew Moore: http://www.autonlab.org/tutorials/index.html


References

- Hidden Markov Model (HMM) Toolbox for Matlab
- The General Hidden Markov Model library (GHMM)
  - http://ghmm.sourceforge.net/
- HTK Speech Recognition Toolkit
  - http://htk.eng.cam.ac.uk