EE 6882
Overview of Statistical Models for Video Indexing

Prof. Shih-Fu Chang

Columbia University

TA:
Eric Zavesky

Fall 2007, **Lecture 4**
Course web site: http://www.ee.columbia.edu/~sfchang/course/svia

dvmm

DIGITAL VIDEO MULTIMEDIA LAB

Statistical Paradigm

- Many problems can be posed as pattern recognition
  - Image classification: indoor vs. outdoor? Face?
  - shot boundary detection, story segmentation
    - Is the current point a boundary?
- Statistical models to handle uncertainty and provide flexibility
- Image processing tools available
  - E.g., homework #1
- Rich tools for learning and prediction
  - See course web site
- Increasing data available
  - NIST TREC Video: 300+ hours
  - Consumer and youtube videos

dvmm

DIGITAL VIDEO MULTIMEDIA LAB

A Very High-Level Stat. Pattern Recog. Architecture

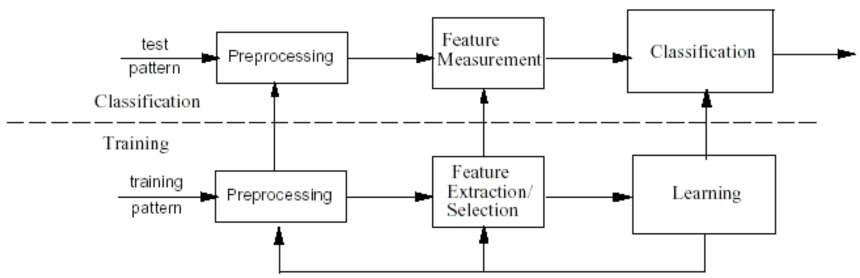


Figure 1: Model for statistical pattern recognition.

(From Jain, Duin, and Mao, SPR Review, ’99)

Important issues (1)

- Image/video processing
  - What’s the adequate quality, resolution, etc?
- Feature extraction
  - Color, texture, motion, region, shape, interest points, audio, speech, text, etc
- Feature representation
  - Histogram, bag, graph etc
  - Invariance to scale, rotation, translation, view, illumination, …
  - How to reduce dimensions?

Important issues (2)

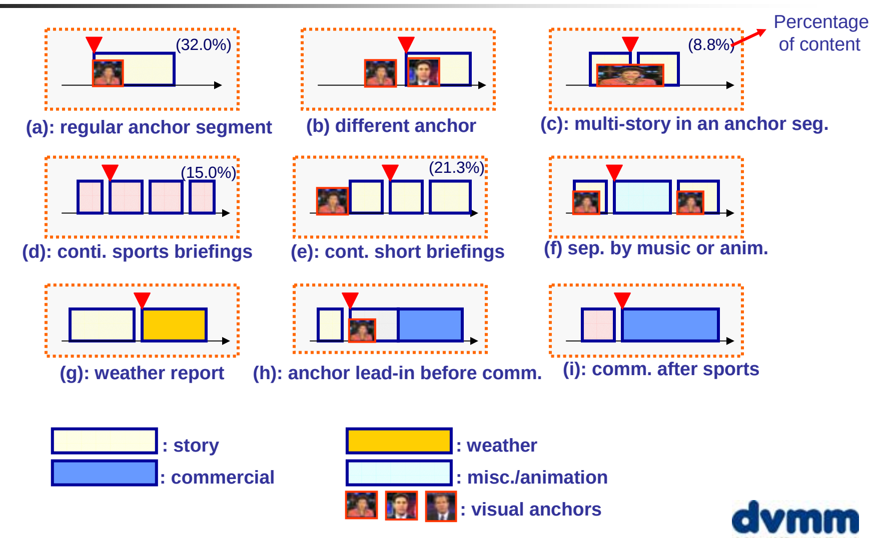
- Distance measurement
  - How to measure similarity between images/videos?
  - L1, L2, Mahalanobis, Earth Mover's distance, vector/graph matching
- Classification models
  - Generative vs. discriminative
  - Multi-modal fusion, early fusion vs. late fusion
    - E.g., how to use joint audio-visual features to detect events (dancing, wedding…)
- Efficiency issues
  - how to speed up training and testing processes?
  - How to rapidly build a model for new domains
- Validation and evaluation
  - How to measure performance?
  - Are models generalizable to new domains?

Three related problems

- Retrieval, Ranking
  - Given a query image, find relevant ones
  - May apply rank threshold to decide relevance
- Classification, categorization, detection
  - Given an image x, predict class label y
- Clustering, grouping
  - Group images/videos into clusters of distinct attributes

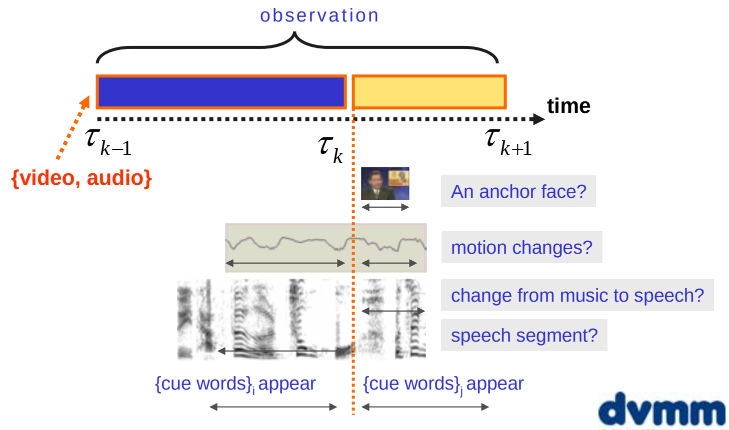
- An example
  - News story segmentation using multi-modal, multi-scale features

First Understand Data Types and Explore Unique Characteristics



News Story Segmentation

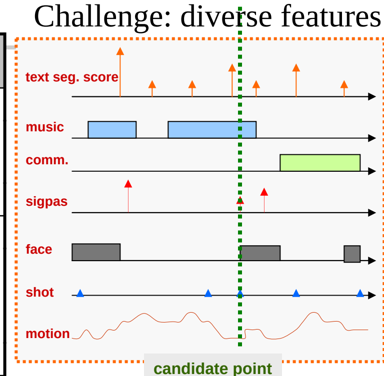
- Objective: a story boundary at time τ_k?
  - τ_k= { shot boundaries or significant pauses}



Need to decide how to formulate features

Modality	Raw Features	Data Type	Value
Video	motion	segment	continuous
	shot boundary	point	binary
	face	segment	continuous
	commercial	segment	binary
Audio	pause	point	continuous
	pitch jump	point	continuous
	significant pause	point	continuous
	musc./spch. disc.	segment	binary
	spch seg./rapidity	segment	continuous
Text	ASR cue terms	point	binary
	V-OCR cue terms	point	binary
	text seg. score	point	continuous
Misc.	combinatorial	point	binary
	sports	segment	binary

Challenge: diverse features



- One way is to use binary predicate:

if x > threshold, then predict segment boundary (b=1)

Example Predicates

no	raw feature set	Predicates
1	Anchor Face	An anchor face segment just starts after the candidate point
2	Significant pause & non-commercial	A significant pause within the non-commercial section appears in the surrounding observation window.
3	Pause	An audio pause with the duration larger than 2.0 second appears after the boundary point.
4	Significant pause	The surrounding observation window has a significant pause with the pitch jump intensity larger than the normalized pitch threshold 1.0 and the pause duration larger than 0.5 second.
5	Speech segment	A speech segment before the candidate point
6	Speech segment	A speech segment starts in the surrounding observation window
7	Commercial	A commercial starts in 15 to 20 seconds after the candidate point.
8	Speech segment	A speech segment ends after the candidate point
9	Anchor face	An anchor face segment occupies at least 10% of next window
10	Pause	The surrounding observation window has a pause with the duration larger than 0.25 second.

Collect Features from Training Samples

Each row represents one predicate f_i

- Face
- Motion
- Significant Pause
- Speech segment
- Commercial
- Text segmentation score
- ASR cue terms

One training sample

b	0	0	1	0	0	0	0	0	1	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0
1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
2	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0
3	0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	0	1	0
4	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0
5	0	0	1	0	1	0	1	0	0	0	0	0	1	1	0	1	0	0	0
6	0	0	1	0	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0
7	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	0
8	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0
9	0	0	1	0	1	0	1	0	1	0	0	0	1	0	1	0	1	0	0
...	0	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	0	0

Choose Model

Maximum entropy model

$$q_\lambda(b \mid x) = \frac{1}{Z_\lambda(x)} e^{\sum_i \lambda_i \cdot f_i(x,b)}$$

$$where \;\; f_i(x,b), b \in \{0,1\}$$

For example

predicate $f_1 = 'anchor\ face'$ $\; f_2 = 'significant\ pause'$

if current observation: $face =$ YES $\; pause =$ NO

$q(b = YES \mid x) = e^{\lambda_1} / (e^{\lambda_1} + e^{\lambda_2})$

$q(b = NO \mid x) = e^{\lambda_2} / (e^{\lambda_1} + e^{\lambda_2})$

Classification: if *q(b=YES|x)* > 0.5, then predict YES.

dvmm DIGITAL VIDEO MULTIMEDIA LAB

Background: Entropy

- Entropy (bits) $\quad H = -\sum_{i=1}^{m} p_i \log_2 p_i$
- Kullback-Leibler (K-L) Distance
  - A measure of 'distance' between 2 distributions

$$D_{KL}\left(p(x), q(x)\right) = \sum_x q(x) \log \frac{q(x)}{p(x)}$$

$$or = \int_{-\infty}^{\infty} q(x) \log \frac{q(x)}{p(x)} dx$$

  - $D_{KL} \geq 0, \; and = 0 \; iff \;\; p(\cdot) = q(\cdot)$
  - Not necessarily symmetric, may not satisfy triangular inequality

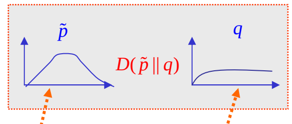
dvmm DIGITAL VIDEO MULTIMEDIA LAB

How to Determine the Weights in the Model?

- Estimate $q_\lambda(b \mid x)$ from training data $T = \{(x_k, b_k)\}$ by minimizing Kullback-Leibler divergence, defined as

$$D(\tilde{p} \parallel q_\lambda) = \sum_x \sum_b \tilde{p}(b,x) \log \frac{\tilde{p}(b \mid x)}{q_\lambda(b \mid x)}$$

$$= -\sum_x \sum_b \tilde{p}(x,b) \log q_\lambda(b \mid x) + \text{constant}(\tilde{p})$$



empirical distribution from data

estimated model

- Find λ_i to maximize the 'entropy'

$$L_{\tilde{p}}(q_\lambda) \equiv \sum_x \sum_b \tilde{p}(x,b) \log q_\lambda(b \mid x)$$

- Iteratively find λ_i

$$\lambda_i' = \lambda_i + \Delta\lambda_i \qquad \Delta\lambda_i = \frac{1}{M} \log\left(\frac{\sum_{x,b} \tilde{p}(x,b) f_i(x,b)}{\sum_{x,b} \tilde{p}(x) q_\lambda(b \mid x) f_i(x,b)} \right)$$

• The objective function is convex.
So the iterative process can reach the optimum.

The same model used to select features

- **Input: collection of candidate features, training samples, and the desired model size**
- **Output: optimal subset of features and their corresponding exponential weights**
- **Current model q augmented with feature h with weight α,**

$$q_{\alpha,h}(b \mid x) = \frac{e^{\alpha h(x,b)} q(b \mid x)}{Z_\alpha(x)}$$

- **Select the candidate which improves current model q the most, in each iteration;**

$$h^* = \arg\max_{h \in C} \left\{ \sup_\alpha \left\{ D(\tilde{p} \parallel q) - D(\tilde{p} \parallel q_{\alpha,h}) \right\} \right\}$$

Reduction of divergence

$$= \arg\max_{h \in C} \left\{ \sup_\alpha \left\{ L_{\tilde{p}}(q_{\alpha,h}) - L_{\tilde{p}}(q) \right\} \right\}$$

Increase of log-likelihood

Optimal Features (from CNN news video)

*** The first 10 "A+V" features automatically discovered for the CNN channel**

no	raw feature set	gain	λ	interpretation
1	Anchor Face	0.3879	0.4771	An anchor face segment just starts after the candidate point
2	Significant pause & non-commercial	0.0160	0.7471	A significant pause within the non-commercial section appears in the surrounding observation window.
3	Pause	0.0058	0.2434	An audio pause with the duration larger than 2.0 second appears after the boundary point.
4	Significant pause	0.0024	0.7947	The surrounding observation window has a significant pause with the pitch jump intensity larger than the normalized pitch threshold 1.0 and the pause duration larger than 0.5 second.
5	Speech segment	0.0019	-0.3566	A speech segment before the candidate point
6	Speech segment	0.0015	0.3734	A speech segment starts in the surrounding observation window
7	Commercial	0.0015	1.0782	A commercial starts in 15 to 20 seconds after the candidate point.
8	Speech segment	0.0022	-0.4127	A speech segment ends after the candidate point
9	Anchor face	0.0016	0.7251	An anchor face segment occupies at least 10% of next window
10	Pause	0.0008	0.0939	The surrounding observation window has a pause with the duration larger than 0.25 second.

every modality helps : anchor face, prosody, and speech segment

Issues of this model (Discussion)

- binary predicate:
if x > threshold, then predict segment boundary (b=1)

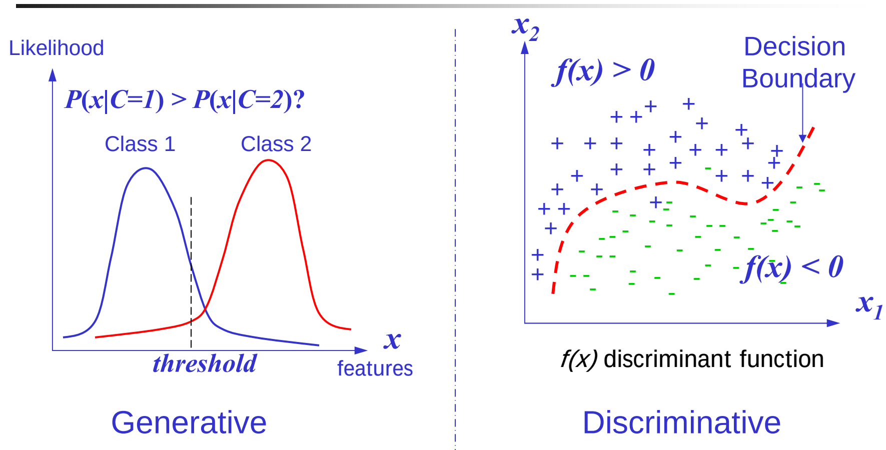
- Features
  - Binary predicates reasonable?
  - Does it capture the unique characteristics?

$$q_\lambda(b\,|\,x) = \frac{1}{Z_\lambda(x)} e^{\sum_i \lambda_i \cdot f_i(x,b)}$$

- Models
  - Exponential models with linear weights adequate?
  - How about the learning algorithm?
    - Enough data to learn the probability models?
    - Speed and complexity

dvmm
DIGITAL VIDEO MULTIMEDIA LAB

A Broader Perspective: Classification Paradigms



Which one does the previous model fall into?

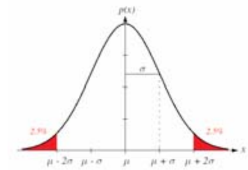
- Generative Models
  - Gaussian Mixture Model

One common issue is to learn probability models

- Gaussian distribution

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)}$$

- Given the same mean and variance, Gaussian has the max entropy
- Sum of a large number of small, independent random variables approaches Gaussian



$\Pr[|x-\mu| \le \sigma] \cong 0.68$

$\Pr[|x-\mu| \le 2\sigma] \cong 0.95$

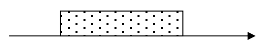
$\Pr[|x-\mu| \le 3\sigma] \cong 0.997$

Mahalanobis distance from x to μ

$r = |x-\mu| / \sigma$

$\boxed{\begin{array}{l} \textit{Entropy of Gaussian}: \\ H_{gau} = 0.5 + \log_2(\sqrt{2\pi}\sigma) \end{array}}$

Comparison w. uniform



Multi-variate Gaussian

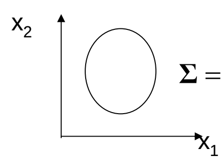
- Multivariate Gaussian, $N(\mu, \Sigma)$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}\sqrt{|\mathbf{\Sigma}|}} e^{\left(\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}$$

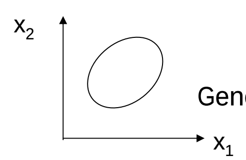
where $\mathbf{x}, \boldsymbol{\mu}$ are D-dimensional vectors

$\mathbf{\Sigma}: D \times D$ matrix

$|\mathbf{\Sigma}|$ is the determinant of $\mathbf{\Sigma}$



$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$



General Σ

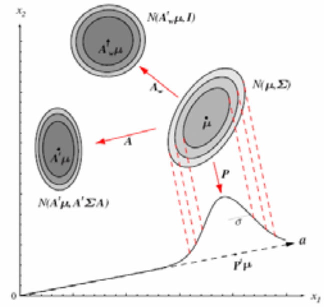
$$(\sigma_{ij})^2 = \Sigma(i,j) = \mathrm{cov}(x(i), x(j))$$
$$= E[(x(i) - \mu(i))(x(j) - \mu(j))]$$

Effect of Linear Transformation

- Linear transformation of Gaussian

$$y = A^t x \qquad y : k \times 1,\ A : d \times k,\ x : d \times 1$$

$$y \sim N(A^t \mu, A^t \Sigma A)$$



- Whitening transform

$$\Sigma = \Phi \Lambda \Phi^t \quad \text{(SVD, Eigenvectors)}$$

$\Phi : [\phi_1 \mid \phi_2 \mid ... \mid \phi_d]$ columns are orthogonal ev $\qquad \Lambda$: diag. matrix of eigenvalues

$$A^t \Sigma A = A^t \Phi \Lambda \Phi^t A = I$$

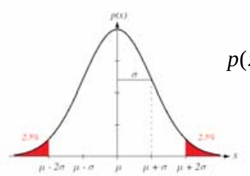
Whitening Trans. $A_w = \Phi \Lambda^{-1/2}$

also PCA Transform

$$y = A_w{}^t x \ \sim \ N(A_w{}^t \mu,\ I)$$

Mahalanobis Distance

- Mahalanobis dist in 1-D



$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\left(-\frac{1}{2\sigma^2}(x-\mu)^2\right)}$$

Mahalanobis distance from x to μ

$$r = |x-\mu| / \sigma$$

$$\Pr[|x-\mu| \le \sigma] = \Pr[r \le 1] \cong 0.68$$

$$\Pr[|x-\mu| \le 2\sigma] = \Pr[r \le 2] \cong 0.95$$

$$\Pr[|x-\mu| \le 3\sigma] = \Pr[r \le 3] \cong 0.997$$

- Multi-Dimensional case

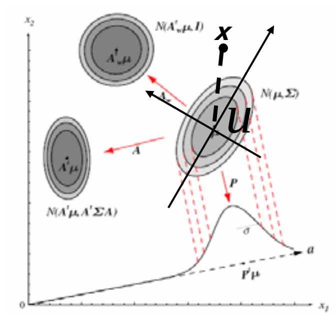
$$p(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu})^t\, \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)$$

$$= \frac{1}{(2\pi)^{D/2}\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu})^t\, \Phi\Lambda^{-1}\Phi^t (\mathbf{x}-\boldsymbol{\mu})\right)$$

$$= \frac{1}{(2\pi)^{D/2}\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(\frac{-1}{2}\underbrace{(A_w{}^t(\mathbf{x}-\boldsymbol{\mu}))^t (A_w{}^t(\mathbf{x}-\boldsymbol{\mu}))}_{r^2}\right)$$

r is the Mahalanobis distance
is also the Euclidean dist in the PCA space

- Malalanobis distance from point x to the mean of a Gaussian



Gaussian Used In Classification

$x \rightarrow C_j, \;\; if \;\; \mathrm{p}\left(C_{\mathrm{j}}|x\right) \geq p\left(C_{\mathrm{i}}|x\right), \;\; when \; \mathrm{i} \neq \mathrm{j}$

MAP classifier

$$\mathrm{p}\left(C_{\mathrm{j}}|x\right) = \frac{\overbrace{p(x\,|\,C_j)}^{likelihood}\;\overbrace{p(C_j)}^{prior}}{\underbrace{p(x)}_{evidence}}$$

ML classification:

$$if \;\; \text{uniform } \mathrm{p}\left(C_{\mathrm{j}}\right) \Rightarrow C_j = \arg\max_{C} p(x\,|\,C)$$

$p(x\,|\,C)$ can be modeled by Gaussians

How to Estimate Gaussian Model Parameters?

- θ = (μ, σ²) $l = \ln P(x_k \mid \theta) = -\frac{1}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}(x_k - \mu)^2$

$$\nabla_\theta l = \begin{pmatrix} \frac{\partial}{\partial \mu}(\ln P(x_k \mid \theta)) \\ \frac{\partial}{\partial \theta_2}(\ln P(x_k \mid \theta)) \end{pmatrix} = \begin{bmatrix} \frac{1}{\theta_2}(x_k - \mu) \\ -\frac{1}{2\sigma^2} - \frac{(x_k-\mu)^2}{2(\sigma^2)^2} \end{bmatrix}$$

$$\sum_k \nabla_\theta \ln P(x_k \mid \theta) = 0 \quad \Rightarrow \quad \hat{\mu} = \frac{\sum_k x_k}{n} \qquad \hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n}(x_k - \hat{\mu})^2$$

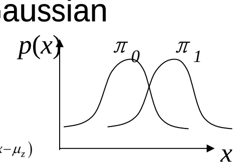
- Multi-Dimensional θ= (μ, Σ)

$$\Rightarrow \quad \hat{\mu} = (1/n)\sum_k \vec{x}_k \qquad \hat{\Sigma} = (1/n)\sum_k (\vec{x}_k - \mu)(\vec{x}_k - \mu)^t$$

ML estimator: mean -> sample mean, variance ->biased sample variance

Mixture Of Gaussians

- Real distributions seldom follow a single Gaussian
  → mixture of Gaussians



$$p(x) = \sum_z p(x,z) \quad = \sum_z p(z)p(x \mid z)$$

$$= \sum_z \pi_z N\left(x \middle| \mu_z, \Sigma_z\right) \quad = \sum_{z=1}^{Z} \pi_z \frac{1}{(2\pi)^{D/2}\sqrt{|\Sigma_z|}} e^{-\frac{1}{2}(x-\mu_z)^T \Sigma_z^{-1}(x-\mu_z)}$$

- Given data $x_1, \ldots, x_N$, define log-likelihood:

$$l = \log(\prod_n p(x_n)) = \sum_{n=1}^{N} \log\left(\pi_0 N(x_n|\mu_0, \Sigma_0) + \pi_1 N(x_n|\mu_1, \Sigma_1)\right)$$

- Posterior probability of x being generated by a specific component

$$\underset{\left(\text{responsiblity of component } i\right)}{posteriers} = \tau^i = p\left(z = i \middle| x, \theta\right), \quad \theta = \{\mu_0, \Sigma_0, \mu_1, \Sigma_1\}$$

E-M Optimization Method

log likelihood $\quad l(\theta) = \sum_{n=1}^{N} \log \sum_z p(x_n, z \mid \theta)$

- Maximization of $l(\theta)$ directly is hard due to log_of_sum
- Instead, look at

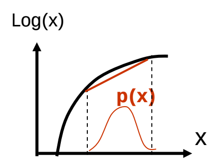
$$\Delta l(\theta) = l(\theta) - l(\theta_t), \quad \theta_t : \textit{current} \text{ estimation of } \theta$$

- Jensen's Inequality

$$\textit{If } \text{f is concave, } \mathrm{f}(\mathrm{E}\{\mathrm{x}\}) \geq \mathrm{E}(\mathrm{f}\{\mathrm{x}\})$$

$$\mathrm{f}(\mathrm{E}\{\mathrm{g}(\mathrm{x})\}) \geq \mathrm{E}(\mathrm{f}\{\mathrm{g}(\mathrm{x})\})$$

$$\textit{e.g.}, \mathrm{f}(\mathrm{x}) = \log(x)$$



$$\log\left(\sum_i p_i x_i\right) \geq \sum_i p_i \log(x_i), \textit{ where } \sum_i p_i = 1$$

$$\textit{If } \text{f is convex, } \mathrm{f}(\mathrm{E}\{\mathrm{x}\}) \leq \mathrm{E}(\mathrm{f}\{\mathrm{x}\})$$

Auxiliary Function in E-M

$$\Delta l(\theta) = l(\theta) - l(\theta_t) = \sum_{n=1}^{N} \log p(x_n \mid \theta) - \sum_{n=1}^{N} \log p(x_n \mid \theta_t)$$

$$= \sum_{n=1}^{N} \log \frac{p(x_n \mid \theta)}{p(x_n \mid \theta_t)} = \sum_{n=1}^{N} \log \sum_z \frac{p(x_n, z \mid \theta)}{p(x_n \mid \theta_t)}$$

marginalization

$$= \sum_{n=1}^{N} \log \sum_z \frac{p(x_n, z \mid \theta)}{p(x_n \mid \theta_t)} \frac{p(x_n, z \mid \theta_t)}{p(x_n, z \mid \theta_t)}$$

$$= \sum_{n=1}^{N} \log \sum_z p(z \mid x_n, \theta_t) \frac{p(x_n, z \mid \theta)}{p(x_n, z \mid \theta_t)}$$

$$\geq \sum_{n=1}^{N} \sum_z p(z \mid x_n, \theta_t) \log \frac{p(x_n, z \mid \theta)}{p(x_n, z \mid \theta_t)}$$

Jensen's inequality

$$= Q(\theta \mid \theta_t)$$

- Note there is no log_of_sum. So taking derivative is easier

E-M improves likelihood

- Auxiliary function derived based on Jensen's Inequality,

$$Q(\theta \mid \theta_t) = \sum_{n=1}^{N} \sum_z \underbrace{p(z \mid x_n, \theta_t)}_{\substack{\text{expectation over} \\ \text{z with current } \theta_t}} \underbrace{\log p(x_n, z \mid \theta)}_{\substack{\text{joint likelihood of} \\ \text{observed \& hidden}}} + const$$

- Now estimate θ_{t+1} by maximizing Q

$$\theta_{t+1} = \arg\max_{\theta} Q(\theta \mid \theta_t)$$

- So in the expectation step, compute τ_n^z , the 'responsibility ' of component z for sample x_n

- In the maximization step, take derivative of Q over θ, and find the new estimate for θ *(Note only sum_of_log is involved)*

EM Always Improves Likelihood

- Why does EM always improve $\boldsymbol{l(\theta)}$ **?**

$$\Delta l(\theta_{t+1}) = l(\theta_{t+1}) - l(\theta_t) \geq Q(\theta_{t+1} \mid \theta_t)$$

$$Q(\theta_{t+1} \mid \theta_t) = \max_{\theta} Q(\theta \mid \theta_t) \geq Q(\theta_t \mid \theta_t) = 0 \quad \therefore \quad \Delta l(\theta_{t+1}) \geq 0$$

$$\boxed{Q(\theta \mid \theta_t) = \sum_{n=1}^{N} \sum_z \underbrace{p(z \mid x_n, \theta_t)}_{\substack{\text{expectation over} \\ \text{z with current } \theta_t}} \underbrace{\log p(x_n, z \mid \theta)}_{\substack{\text{joint likelihood of} \\ \text{observed \& hidden}}} + const}$$

- General steps of EM:
  - Define likelihood model with parameters θ
  - Identify hidden variables z
  - Derive the auxiliary function and the E and M equations
  - In each iteration, estimate the posteriors of hidden variables
  - Re-estimate the model parameters. Repeat until stop

Expectation-Maximization (E-M) Solution of GMM

$$Q(\theta \mid \theta_t) = \sum_{n=1}^{N} \sum_z \underbrace{p(z \mid x_n, \theta_t)}_{\substack{\text{expectation over} \\ \text{z with current } \theta_t}} \underbrace{\log p(x_n, z \mid \theta)}_{\substack{\text{joint likelihood of} \\ \text{observed \& hidden}}} + const$$

- EM for estimating θ and τ_i.
  - Follow 'divide and conquer' principle. In iteration step t:

$$Expectation: \ \tau_n^{i(t)} = \frac{\pi_i^{(t)} N\left(x_n \mid \mu_i^{(t)}, \Sigma_i^{(t)}\right)}{\sum_j \pi_i^{(t)} N\left(x_n \mid \mu_j^{(t)}, \Sigma_j^{(t)}\right)}$$

Weight from component i

$$Maximation \ : \ \mu_i^{(t+1)} = \frac{\sum_n \tau_n^{i(t)} x_n}{\sum_n \tau_n^{i(t)}} \qquad \Sigma_i^{(t+1)} = \frac{\sum_n \tau_n^{i(t)} \left(x_n - \mu_i^{(t)}\right)\left(x_n - \mu_i^{(t)}\right)^T}{\sum_n \tau_n^{i(t)}}$$

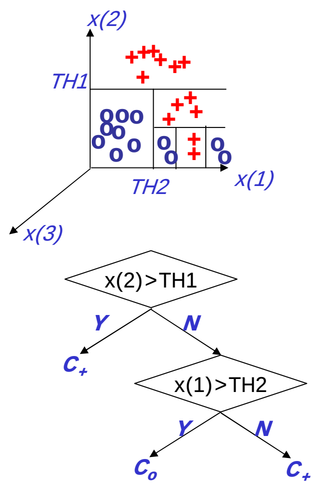
$$\pi_i^{(t+1)} = \frac{\sum_n \tau_n^{i(t)}}{N}$$

Divide data to each group,
Compute mean and variance
from each group

- Discriminative Models

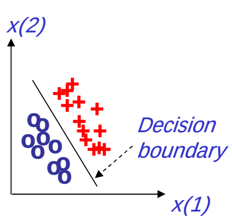
Simple Discriminative Classifier

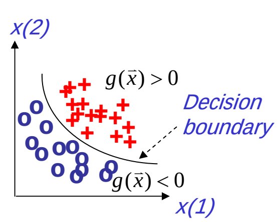
- Find most opportunistic dimension in each step
- Selection criterion
  - Entropy
  - Variance before / after
- Stop criterion
  - Avoid overfitting



Parametric Discriminant Analysis

- Example of discriminant function
  - Linear and quadratic discriminant

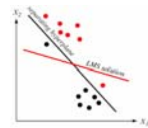




$$g(\vec{x}) = ax_1 + bx_2$$

$$g(\vec{x}) = ax_1^2 + bx_2^2 + cx_1x_2$$

Linear Discriminant Classifiers

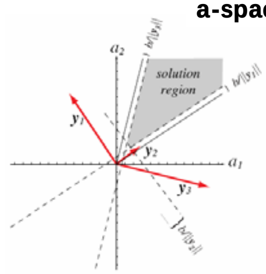


$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 \quad \Rightarrow$ find weight $\mathbf{w}$ and bias w_o

- Augmented Vector $\mathbf{y} = \begin{bmatrix}1\\ \mathbf{x}\end{bmatrix} = \begin{bmatrix}1\\ x_1\\ \vdots\\ x_d\end{bmatrix} \quad \mathbf{a} = \begin{bmatrix}w_0\\ \mathbf{w}\end{bmatrix} = \begin{bmatrix}w_0\\ w_1\\ \vdots\\ w_d\end{bmatrix} \quad \Rightarrow g(\mathbf{x}) = g(\mathbf{y}) = \mathbf{a}^t\mathbf{y}$

$\boxed{\text{map } \mathbf{y} \text{ to class } \omega_1 \text{ if } g(\mathbf{y})>0, \text{ otherwise class } \omega_2}$

a-space

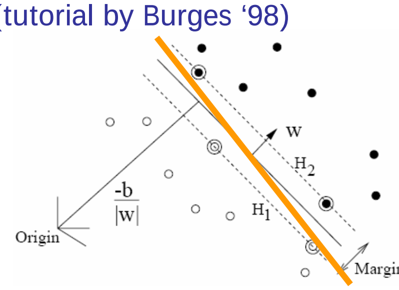


- Design Objective

$$\mathbf{a}^t\mathbf{y}_i > \text{b}, \ \text{ if class } \omega_1,$$
$$\mathbf{a}^t\mathbf{y}_i < \text{-b}, \ \text{ if class } \omega_2, \forall \mathbf{y}_i \quad, \text{b}>0$$

- Each y_i defines a half plane in the weight space (a).
- Note we search weight solutions in the a-space.

Support Vector Machine (tutorial by Burges '98)



- **Look for separation plane with the highest margin**

  Decision boundary

  $H_0: \ \mathbf{w}^t\mathbf{x} + b = 0$

- **Linearly separable**

$\mathbf{w}^t\mathbf{x}_i + b \geq +1 \quad \forall \mathbf{x}_i$ in class ω_1 i.e. $y_i = +1$

$\mathbf{w}^t\mathbf{x}_i + b \leq -1 \quad \forall \mathbf{x}_i$ in class ω_2 i.e. $y_i = -1$

Inequality constraints : $\quad (label_i)\cdot(\mathbf{w}^t\mathbf{x}_i + b) - 1 \geq 0 \ , \ \forall \ i$

- **Two parallel hyperplanes defining the margin**

  hyperplane $H_1(H_+): \mathbf{w}^t\mathbf{x}_i + b = +1$

  hyperplane $H_2(H\_): \mathbf{w}^t\mathbf{x}_i + b = -1$

- **Margin: sum of distances of the closest points to the separation plane**

$\boxed{\text{margin} = 2/\|\mathbf{w}\|}$

- **Best plane defined by w and b**

Finding the maximal margin

minimize $\frac{1}{2}\|\mathbf{w}\|^2$ subject to inequality constraints

$y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1 \geq 0 \quad i = 1, \cdots, l \quad ; y_i$ is label

- **Use the Lagrange multiplier technique for the constrained opt. problem**

minimize L_p *w.r.t.* $\mathbf{w}$ and b

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l}\alpha_i(y_i(\mathbf{w}^t\mathbf{x}_i + b) - 1)$$

$$\alpha_i \geq 0$$

$$\frac{dL_p}{d\mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{l}\alpha_i y_i \mathbf{x}_i$$

$$\frac{dL_p}{db} = 0 \quad \Rightarrow \quad \sum_{i=1}^{l}\alpha_i y_i = 0$$

maximize L_D *w.r.t.* $\mathbf{w}$ and b

$$L_D = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

with conditions :

$$\sum_{i=1}^{l}\alpha_i y_i = 0$$

$$\alpha_i \geq 0$$

Quadratic Programming

Primal Problem $\longleftrightarrow$ **Dual Problem**

- **Prime and Dual have the same solutions of w and b**

KKT conditions

$$\frac{\partial}{\partial w_\nu}L_P = w_\nu - \sum_i \alpha_i y_i x_{i\nu} = 0 \qquad \nu = 1, \cdots, d \quad \longrightarrow \quad \mathbf{w}^* = \sum_{i=1}^{l}\alpha_i y_i \mathbf{x}_i$$

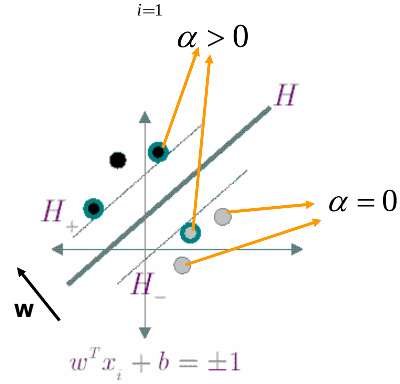
$$\frac{\partial}{\partial b}L_P = -\sum_i \alpha_i y_i = 0$$

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0 \qquad i = 1, \cdots, l$$

$$\alpha_i \geq 0 \qquad \forall i$$

$$\alpha_i(y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1) = 0 \qquad \forall i$$

- **Weight sum from positive class = Weight sum from negative class**
- **Direction of w: roughly from negative support vectors to positive ones**



if $\alpha_i > 0$, $\mathbf{x}_i$ is on H_+ or H_- and is a support vector

Non-separable: not every sample can be correctly classified

- **Add slack variables** ξ_i

$$\begin{aligned} \mathbf{x}_i \cdot \mathbf{w} + b &\geq +1 - \xi_i \quad \text{for } y_i = +1 \\ \mathbf{x}_i \cdot \mathbf{w} + b &\leq -1 + \xi_i \quad \text{for } y_i = -1 \\ \xi_i &\geq 0 \ \forall i. \end{aligned}$$

if $\xi_i > 1$, then $\mathbf{x}_i$ is misclassified (i.e. training error)

Lagrange multiplier: minimize

$$L_P = \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i}_{\text{New objective function}} - \sum_i \alpha_i\{y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i$$

Ensure positivity

KKT Conditions for non-separable Solutions

$$\frac{\partial L_P}{\partial w_\nu} = w_\nu - \sum_i \alpha_i y_i x_{i\nu} = 0$$

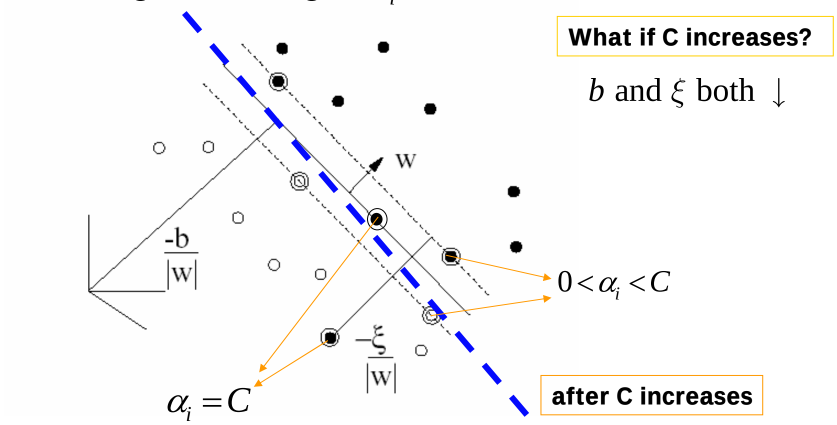
$$\frac{\partial L_P}{\partial b} = -\sum_i \alpha_i y_i = 0$$

$$\begin{aligned} \frac{\partial L_P}{\partial \xi_i} = C - \alpha_i - \mu_i &= 0 \\ y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i &\geq 0 \\ \xi_i &\geq 0 \\ \alpha_i &\geq 0 \\ \mu_i &\geq 0 \\ \alpha_i\{y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i\} &= 0 \\ \mu_i \xi_i &= 0 \end{aligned}$$

If $0 < \alpha_i < C$, then $\xi_i = 0$: $\mathbf{x}_i$ is on H_1 or H_2

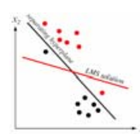
If $\alpha_i = C$,
then $\xi_i > 0$: $\mathbf{x}_i$ is inside the margin or on the wrong side
or $\xi_i = 0$: $\mathbf{x}_i$ is on H_1 or H_2

- **All the points located in the margin gap or the wrong side will get $\alpha_i = C$**



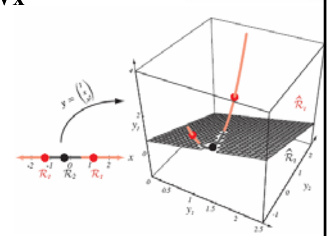
- **When C increases, incorrect samples get more weights**
  - **try to minimize incorrect samples**
  - **better training accuracy, but smaller margin**
  - **less generalization performance**

Generalized Linear Discriminant Functions



- **Include more than just the linear terms**

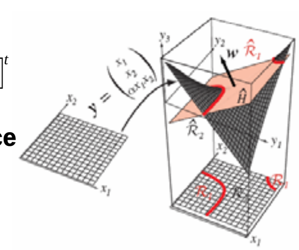
$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=1}^{d} w_{ij} x_i x_j = w_0 + \mathbf{w}^t\mathbf{x} + \mathbf{x}^t\mathbf{W}\mathbf{x}$$



- **Shape of decision boundary**
  - **ellipsoid, hyperhyperboloid, lines etc**
- **In general** $g(\mathbf{x}) = \sum_{i=1}^{\hat{d}} a_i \phi_i(\mathbf{x}) = \mathbf{a}^t \Phi$
- **Example**

$$g(x) = a_1 + a_2 x + a_3 x^2 = [a_1 \quad a_2 \quad a_3][1 \quad x \quad x^2]^t$$

$$g(x) = a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 = [a_1 \quad a_2 \quad a_3][1 \quad x_1 \quad x_1 x_2]^t$$



- **Data become separable in higher-dimensional space**
  - **learning parameters in high dimension is hard (curse of dimensionality)**
  - **instead, try to maximize margins → SVM**

Non-Linear Space

$\Phi : \mathbf{R}^d \mapsto \mathcal{H}.$ Map to a high dimensional space, to make the data separable

$$\Phi(\mathbf{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}\, x_1 x_2 \\ x_2^2 \end{pmatrix}$$

- **Find the SVM in the high-dim space**

$$g(\mathbf{x}) = \underbrace{\sum_{i=1}^{N_s} \alpha_i y_i \Phi(\mathbf{s}_i)}_{\mathbf{w}} \cdot \Phi(\mathbf{x}) + b$$

- **Luckily, we don't have to find** $\Phi(\mathbf{s}_i)$ nor $\sum_{i=1}^{l} \alpha_i y_i \Phi(\mathbf{s}_i)$
- **Instead, we define kernel** $K(\mathbf{s}_i, \mathbf{x}) = \Phi(\mathbf{s}_i) \cdot \Phi(\mathbf{x})$

$$\Rightarrow g(\mathbf{x}) = \sum_{i=1}^{N_s} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b$$

- **We can use the same method to maximize L_D to find** α_i

$$L_D = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$

$$= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Some popular kernels

$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^p$ **polynomial**

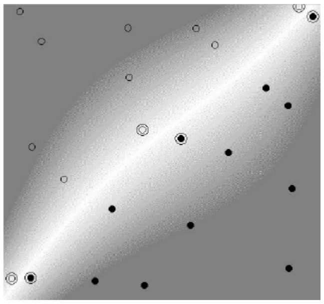
$K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$ **Gaussian Radial Basis Function (RBF)**

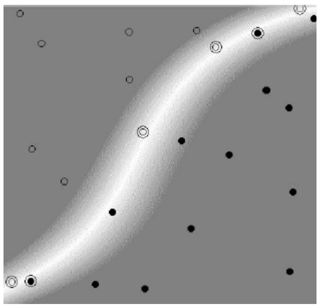
$K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} - \delta)$ **sigmoidal neural network**

separable

Cubic polynomial

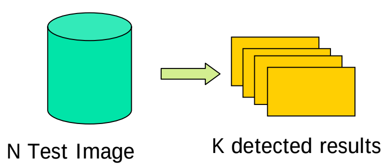
non-separable





- See the SVM demos (Eric)

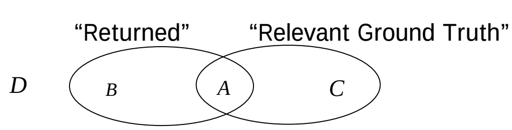
Evaluation



$$V_n = \begin{cases} 1 & \text{"Relevant"} \\ 0 & \text{"Irrelevant"} \end{cases} \quad n = 0 \cdots N-1$$

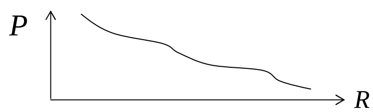
- Detection $A = \sum_{n=0}^{K-1} V_n$
- False Alarms $B = \sum_{n=0}^{K-1} (1 - V_n)$
- Misses $C = (\sum_{n=0}^{N-1} V_n) - A$
- Correct Dismissals $D = (\sum_{n=0}^{N-1} (1 - V_n)) - B$

- Recall $R = A/(A+C)$
- Precision $P = A/(A+B)$
- Fallout $F = B/(B+D)$
- Combined $F_1 = \frac{P \cdot R}{(P+R)/2}$



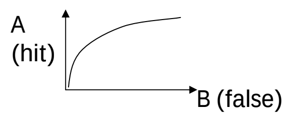
Evaluation Measures

Precision Recall Curve



2. Receiver Operating Characteristic (ROC Curve)

A vs *B*



dvmm

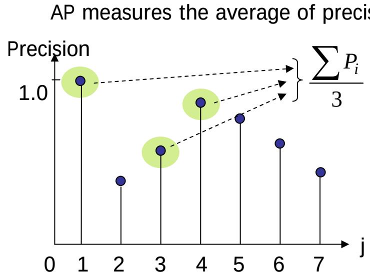
Evaluation Metric: Average Precision

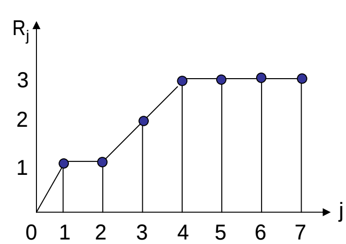
S → Ranked list of data in response to a query

	D_{15}	D_8	D_{63}	D_{21}	...	...	D_s
Ground truth	*1*	*0*	*1*	*1*	*0*	*0*	*0*
Precision	*1/1*	*1/2*	*2/3*	*3/4*	*3/5*	*3/6*	*3/7*

Average precision: $AP = \frac{1}{R}\sum_{j=1}^{s}\frac{R_j}{j}I_j,\quad R: total \text{ number of relevant data}$

AP measures the average of precision values at R relevant data points





Evaluation Metric: Average Precision (2)

- AP depends on the rankings of relevant data and the size of the relevant data set. E.g., R=10

Case I: + + + + + + + + + + - - - - - - -

Pre: 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 AP=1

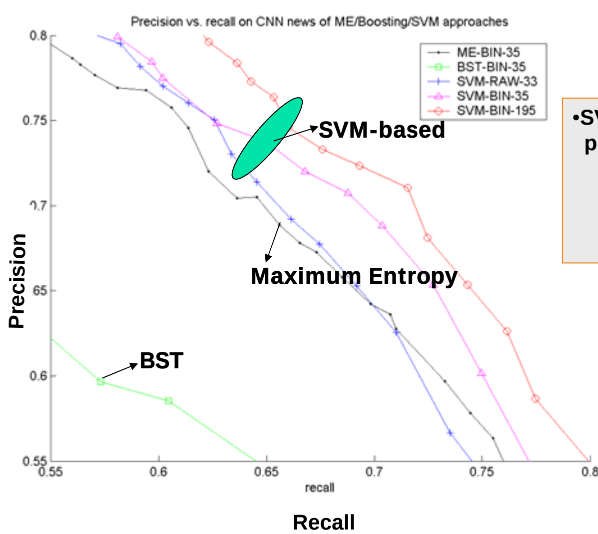
Case II: - + - + - + - + - + - + - + - + - + - +

Pre: 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 1/2 AP=1/2

Case III: - - - - - - - - - - + + + + + + + + + +

Pre: 1/11 2/12 … … 10/20 AP~0.3

Example: SVM for News Story Segmentation

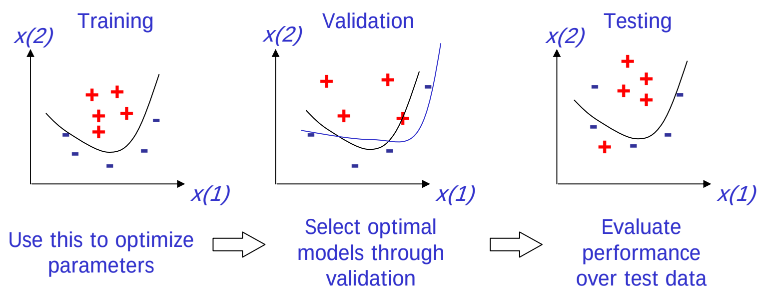


•SVM with 195 binary features performs the best

→SVM has excellent feature fusion capability

→Predicate binarization shields noise in the feature

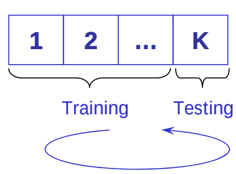
Training / Validation / Testing



- Appropriate if the same distributions are followed over different sets

Training / Validation / Testing (cont.)

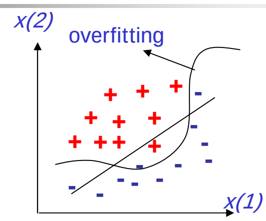
- Cross validation, leave-one-out

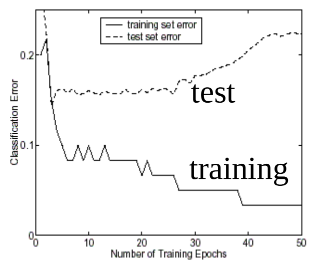


Rotate the choice of the test set and average the performance over runs

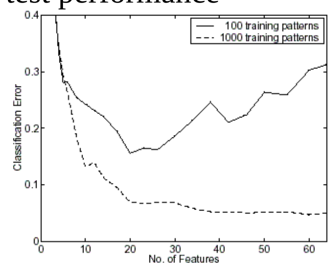
Curse of Dimensionality and Overtraining

A case of overfitting





test performance



Very rough rule of thumb –

$$\frac{\text{\# of training samples per class}}{\text{feature dimension}} > 10$$