CS6784 - Spring 2010

Review, Notation and Terminology

Thorsten Joachims

Cornell University
Department of Computer Science

Project

- **Do you have a project idea?**
  - Yes, then prepare “pitch” to present in-class on Thursday 2/4.
  - No, then join one of the pitched ideas.
- **How to prepare your pitch?**
  - Make one slide following the template on the next slide (either PDF or Powerpoint)
  - Email me the slide by 10:00am on Thursday 2/4.
  - Give a 2-minute explanation why the project is
    - interesting,
    - significant,
    - relevant to the CS6784, and
    - feasible.
- **Form project groups**
  - Before 2/11, matchmaking in class or online
  - Jointly prepare project proposal

Project Title

Proposer: your name, your email

Here you can say anything that helps you explain why your project is interesting, significant, relevant to CS6784, and feasible.

Do not use more than one slide!

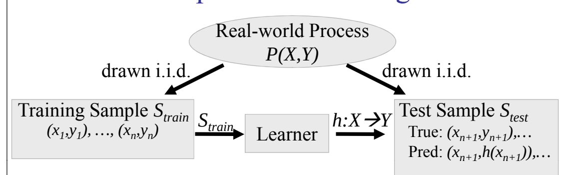
Paper Assignments

- **Papers are on course homepage**
- **Bid on papers:**
  - Deadline: Monday, Feb 1, 11:59pm
  - Bidding online
- **First three papers:**
  - Feb 11
    - Ben Taskar, Carlos Guestrin and Daphne Koller. Max-Margin Markov Networks. NIPS, 2004.
    - D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng. Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data. CVPR, 2005.
  - Feb 16
    - J. Weston, O. Chapelle, A. Elisseeff, B. Schoelkopf and V. Vapnik, Kernel Dependency Estimation, NIPS, 2002.

Machine Learning Tasks Relevant for CS6784

- **Supervised Learning**
  - Data: (x,y) ~ iid P(X,Y)
  - x: Input
  - y: Label / output
  - Learn: h: X → Y
- **Unsupervised Learning**
  - Data: (x) ~ iid P(X)
  - x: Observation
  - Learn: structure of P(X)
- **Reinforcement Learning**
  - Data: Markov decision Process P(S|A,S’), P(R|S)
  - (s,a,r)*: Sequence of state/action/reward triples
  - Learn: policy π: S → A that maximizes reward

Supervised Learning



- **Learning Task: *$P(X,Y) = P(X)\,P(Y|X)$***
  - Input Space: X (e.g. feature vectors, word sequence, etc.)
  - Output Space: Y (e.g. class label $1..k$)
  - Training Data: $S_{train}=((x_1,y_1), \ldots, (x_n,y_n)) \sim_{iid} P(X,Y)$
- **Goal: Find h: X → Y with low prediction error *$Err_P(h)$***

Generalization Error and Sample Error

Definition: *The* **prediction error/generalization error/true error/expected loss/risk** $Err_P(h)$ *of a hypothesis* h *for a learning task* $P(X, Y)$ *is*

$$Err_P(h) = \sum_{\vec{x} \in X, y \in Y} \Delta(h(\vec{x}), y) P(X = \vec{x}, Y = y).$$

Definition: $\Delta(a, b)$ *is a loss function that measures the cost of making a wrong prediction. A commonly used loss function is the the 0/1-loss*

$$\Delta(a, b) = \begin{cases} 0 & if (a == b) \\ 1 & else \end{cases}$$

Definition: *The* **error on sample** S $Err_S(h)$ *of a hypothesis* h *is* $Err_S(h) = \frac{1}{n} \sum_{i=1}^{n} \Delta(h(\vec{x}_i), y_i)$.

Classifying Examples

- **Bayes' Decision Rule:** Optimal decision is

$$h(x) = \underset{y \in Y}{\operatorname{argmin}} \left[\sum_{y' \in Y} \Delta(y', y) P(Y = y' | X = x) \right]$$

- **Equivalent Reformulation:** For 0/1-Loss

$$\Delta(y, y') = \begin{cases} 1 & if (y \neq y') \\ 0 & if (y == y') \end{cases}$$

$$h(x) = \underset{y \in Y}{\operatorname{argmax}} [P(Y = y | X = x)]$$

Generative vs. Discriminative Models

Learning Task:
- Generator: Generate descriptions according to distribution *P(X)*.
- Teacher: Assigns a value to each description based on *P(Y|X)*.

Training Examples $(x_1, y_1), \ldots, (x_n, y_n) \sim P(X, Y)$

Discriminative Model
- Model *P(Y|X)* with *P(Y|X,ω)*
  - Find *ω* e.g. via MLE
  - Examples: Log. Reg., CRF
- Model discriminant functions
  - Find $h_\omega \in H$ with low train loss (e.g. Emp. Risk Min.)
  - Examples: SVM, Dec. Tree

Generative Model
- Model *P(X,Y)* with distributions *P(Y,X|ω)*
- Find ω that best matches *P(X,Y)* on training data (e.g. MLE)
- Examples: naive Bayes, HMM

Generative Model: Model P(X,Y)

- **Bayes' Decision Rule:** For 0/1-Loss, optimal decision is

$$h(x) = \underset{y \in Y}{\operatorname{argmax}} [P(Y = y | X = x)]$$

- **Equivalent Reformulations:** For 0/1-Loss,

$$\begin{aligned} h(x) &= \underset{y \in Y}{\operatorname{argmax}} [P(Y = y | X = x)] \\ &= \underset{y \in Y}{\operatorname{argmax}} \left[\frac{P(X = x | Y = y) P(Y = y)}{P(X = x)} \right] \\ &= \underset{y \in Y}{\operatorname{argmax}} [P(X = x | Y = y) P(Y = y)] \\ &= \underset{y \in Y}{\operatorname{argmax}} [P(X = x, Y = y)] \end{aligned}$$

- **Learning: maximum likelihood (or MAP, or Bayesian)**
  - Assume model class P(X,Y|ω) with parameters ω ∈ Ω
  - Find $\omega' = \underset{\omega \in \Omega}{\operatorname{argmax}} \prod_{i=1}^{n} [P(Y = y_i, X = x_i | \omega)]$

Naïve Bayes' Classifier (Multivariate)

fever (3)	cough (2)	pukes (2)	flu?
high	yes	no	1
high	no	yes	1
low	yes	no	-1
low	yes	yes	1
high	no	yes	???

- **Input Space X: Feature Vector**
- **Output Space Y: {1,-1}**
- **Model:**
  - Prior class probabilities

    $P(Y = +1)$ $\quad P(Y = -1)$
  - Class conditional model (one for each class)

$$P(X = \vec{x} | Y = +1) = \prod_{j=1}^{N} P(X^{(j)} = x^{(j)} | Y = +1)$$

$$P(X = \vec{x} | Y = -1) = \prod_{j=1}^{N} P(X^{(j)} = x^{(j)} | Y = -1)$$

- **Classification rule:**

$$h_{naive}(\vec{x}) = argmax_{y \in \{+1, -1\}} \left\{ P(Y = y) \prod_{j=1}^{N} P(X^{(j)} = x^{(j)} | Y = y) \right\}$$

Estimating the Parameters of Naïve Bayes

fever (3)	cough (2)	pukes (2)	flu?
high	yes	no	1
high	no	yes	1
low	yes	no	-1
low	yes	yes	1
high	no	yes	???

- **Count frequencies in training data**
  - n: number of training examples
  - n_+ / n_- : number of pos/neg examples
  - $\#(X^{(j)} = x^{(j)}, y)$: number of times feature $X^{(j)}$ takes value $x^{(j)}$ for examples in class y
  - $| X^{(j)} |$: number of values attribute of $X^{(j)}$
- **Estimating:** $\omega' = \underset{\omega \in \Omega}{\operatorname{argmax}} \prod_{i=1}^{n} \left[P(Y = y_i) \prod_{j=1}^{N} \left[P(X_i^{(j)} = x_i^{(j)} | Y = y_i) \right] \right]$
  - P(Y): Maximum Likelihood Estimate

$$\hat{P}(Y = 1) = \frac{n_+}{n} \quad \hat{P}(Y = -1) = \frac{n_-}{n}$$

  - P(X|Y): Maximum Likelihood Estimate

$$\hat{P}(X^{(j)} = x^{(j)} | Y = y) = \frac{\#(X^{(j)} = x^{(j)}, y)}{n_y}$$

  - P(X|Y): Smoothing with Laplace estimate

$$\hat{P}(X^{(j)} = x^{(j)} | Y = y) = \frac{\#(X^{(j)} = x^{(j)}, y) + 1}{n_y + |X^{(j)}|}$$

Generative vs. Discriminative Models

Learning Task:
- Generator: Generate descriptions according to distribution $P(X)$.
- Teacher: Assigns a value to each description based on $P(Y|X)$.

Training Examples $(x_1, y_1), ..., (x_n, y_n) \sim P(X, Y)$

Discriminative Model
- Model $P(Y|X)$ with $P(Y|X,\omega)$
  - Find ω e.g. via MLE
  - Examples: Log. Reg., CRF
- Model discriminant functions
  - Find $h_\omega \in H$ with low train loss (e.g. Emp. Risk Min.)
  - Examples: SVM, Dec. Tree

Generative Model
- Model $P(X,Y)$ with distributions $P(Y,X|\omega)$
- Find ω that best matches $P(X,Y)$ on training data (e.g. MLE)
- Examples: naive Bayes, HMM

Discriminative Model: Model P(Y|X)

- **Bayes' Decision Rule:**
  - General: $h(x) = \underset{y \in Y}{\text{argmin}} \left[\sum_{y' \in Y} \Delta(y', y) P(Y = y' | X = x) \right]$
  - Assume 0/1 Loss $\Delta(y,y') = 1$, if $y \neq y'$, 0 else

$$h(x) = \underset{y \in Y}{\text{argmax}} \, [P(Y = y | X = x)]$$

- **Learning: maximum likelihood (or MAP, or Bayesian)**
  - Assume model class $P(Y|X,\omega)$ with parameters $\omega \in \Omega$
  - Find

$$\omega' = \underset{\omega \in \Omega}{\text{argmax}} \prod_{i=1}^{n} [P(Y = y_i | X = x_i, \omega)]$$

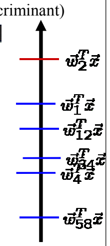
Logistic Regression

- **Assume:**

$$P(Y = y | X = x, \omega = (w_1, ..., w_k)) = \frac{e^{w_y^T x}}{\sum_{y' \in Y} e^{w_{y'}^T x}}$$

→Learn one weight vector w_y for each class $y \in Y$ (linear discriminant)

$$h(x) = argmax_{y \in Y} [P(Y = y | X = x, \omega)]$$
$$= argmax_{y \in Y} \left[w_y^T x \right]$$



- **Maximum Likelihood training:**

$$\omega' = \underset{\omega \in \Omega}{\text{argmax}} \left[\prod_{i=1}^{n} P(Y = y_i | X = x_i, \omega) \right]$$
$$= \underset{\omega \in \Omega}{\text{argmax}} \left[\sum_{i=1}^{n} \log(P(Y = y_i | X = x_i, \omega)) \right]$$
$$= \underset{(w_1, ..., w_k)}{\text{argmax}} \left[\sum_{i=1}^{n} w_{y_i}^T x_i - \log \left(\sum_{y' \in Y} e^{w_{y'}^T x_i} \right) \right]$$

Generative vs. Discriminative Models

Learning Task:
- Generator: Generate descriptions according to distribution $P(X)$.
- Teacher: Assigns a value to each description based on $P(Y|X)$.

Training Examples $(x_1, y_1), ..., (x_n, y_n) \sim P(X, Y)$

Discriminative Model
- Model $P(Y|X)$ with $P(Y|X,\omega)$
  - Find ω e.g. via MLE
  - Examples: Log. Reg., CRF
- Model discriminant functions
  - Find $h_\omega \in H$ with low train loss (e.g. Emp. Risk Min.)
  - Examples: SVM, Dec. Tree

Generative Model
- Model $P(X,Y)$ with distributions $P(Y,X|\omega)$
- Find ω that best matches $P(X,Y)$ on training data (e.g. MLE)
- Examples: naive Bayes, HMM

Discriminative Model: Model Discriminant Function *h* Directly

- **Discriminant Function: h_ω: $X \times Y \rightarrow \Re$**

$$h(x) = \underset{y \in Y}{\text{argmin}} \left[\sum_{y' \in Y} \Delta(y', y) P(Y = y' | X = x) \right]$$
$$= \underset{y \in Y}{\text{argmax}} \, [h(x, y)] \quad (\text{e.g. } h(x, y) = \left[w_y^T x \right])$$

- **Consistency of Empirical Risk:**
  - Training Error (i.e. Empirical Risk): $Err_S(h) = \sum_{i=1}^{n} \Delta(y_i, h(x_i))$
  - For sufficiently "small" H_Ω and "large" S: Rule $h' \in H_\Omega$ with best $Err_S(h')$ has $Err_P(h')$ close to $min_{h \in H\Omega}\{Err_P(h)\}$
- **Learning: Empirical Risk Minimization (ERM)**
  - Assume class H_Ω of discriminant functions $h_\omega: X \rightarrow Y$
  - Find

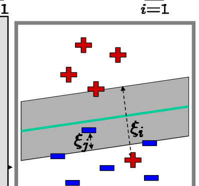
$$h'_\omega = \underset{h_\omega \in H_\Omega}{\text{argmin}} \left[\sum_{i=1}^{n} \Delta(y_i, h_\omega(x_i)) \right]$$

Support Vector Machine

- **Training Examples:** $(\vec{x}_1, y_1), ..., (\vec{x}_n, y_n) \;\; \vec{x} \in \Re^N \;\; y \in \{+1, -1\}$
- **Hypothesis Space:** $H_\Omega = \left\{ h(\vec{x}) = sgn\left[\vec{w}^T \vec{x} + b \right] : ||\vec{w}|| < \Omega \right\}$
- **Training Loss:** $Err_{S_{train}}(h) = \sum_{i=1}^{n} \Delta(y_i, h(\vec{x}_i)) \leq \sum_{i=1}^{n} \xi_i$

Optimization Problem:

$$\min_{\vec{w}, \vec{\xi}, b} \quad \frac{1}{2} \vec{w}^T \vec{w} + C \sum_{i=1}^{n} \xi_i$$
$$s.t. \quad y_1(\vec{w}^T \vec{x}_1 + b) \geq 1 - \xi_1$$
$$...$$
$$y_n(\vec{w}^T \vec{x}_n + b) \geq 1 - \xi_n$$



[Vapnik et al.]

[Crammer & Singer 02]

Training: Find $<\vec{w}_1, ..., \vec{w}_k>$ that solve

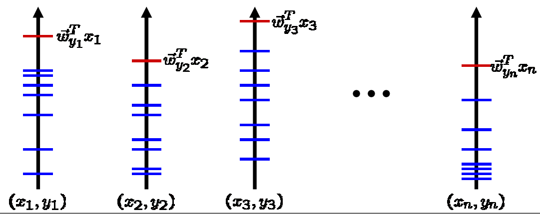
$$\min_{\vec{w}_1,...,\vec{w}_n,\vec{\xi}} \sum_{i=1}^{k} \vec{w}_i^2 + C \sum_{i=1}^{n} \xi_i$$

$$s.t. \quad \forall j \neq y_1 : \vec{w}_{y_1}^T \vec{x}_1 \geq \vec{w}_j^T \vec{x}_1 + 1 - \xi_1$$

$$\cdots$$

$$\forall j \neq y_n : \vec{w}_{y_n}^T \vec{x}_n \geq \vec{w}_j^T \vec{x}_n + 1 - \xi_n$$

..., $k\}$



Types of Learning Methods

$$h(x) = \operatorname*{argmax}_{y \in Y} [P(Y = y, X = x)]$$

Generative: **Joint Model**

$$= \operatorname*{argmax}_{y \in Y} [P(Y = y | X = x)]$$

Discriminative: **Probabilistic Discriminative**

$$= \operatorname*{argmax}_{y \in Y} [h(x, y)]$$

Discriminative: **Empirical Risk Minimization**

Flexibility

Complexity

So what about Structured Outputs?

- **Approach: view as multi-class classification task**
  - Every complex output $y \in Y$ is one class
  - x The bear chased the cat → y Det→N→V→Det→N
- **Problem: Exponentially many classes!**
  - Generative Model: $P(X,Y|\omega)$
  - Discriminative Model: $P(Y|X,\omega)$
  - Discriminant Functions: $h_\omega: X \times Y \to \Re$
- **Challenges**
  - How to compactly represent model?
  - How to do efficient inference with model (i.e. $\operatorname*{argmax}_{y \in Y} [h(x, y)]$)?
  - How to effectively estimate model from data?
    (e.g. compute $h_\omega' = \operatorname*{argmin}_{h_\omega \in H_\Omega} \sum_{i=1}^{n} \Delta(y_i, h_\omega(x_i))$)