CS6784
Review, Notation and Terminology

Spring 2014

Thorsten Joachims

Cornell University
Department of Computer Science

Project

- Do you have a project idea?
  - Yes, then prepare “pitch” to present in-class on Tuesday 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) and name the file <YourLastName>.<pdf|ppt>
  - Email Josh (jlmo@cs.cornell.edu) the slide by 10:00am on Tuesday 2/4.
  - Give a 2-minute explanation why the project is
    - interesting,
    - significant,
    - relevant to the CS6784, and
    - feasible.
- Form project groups
  - Matchmaking in class or online via piazza
  - Declare group on CMT by 2/9
  - Jointly prepare project proposal by 2/16 and submit via CMT

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: Friday, Jan 31, 11:59pm
  - Bidding online via CMT
- First two 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.

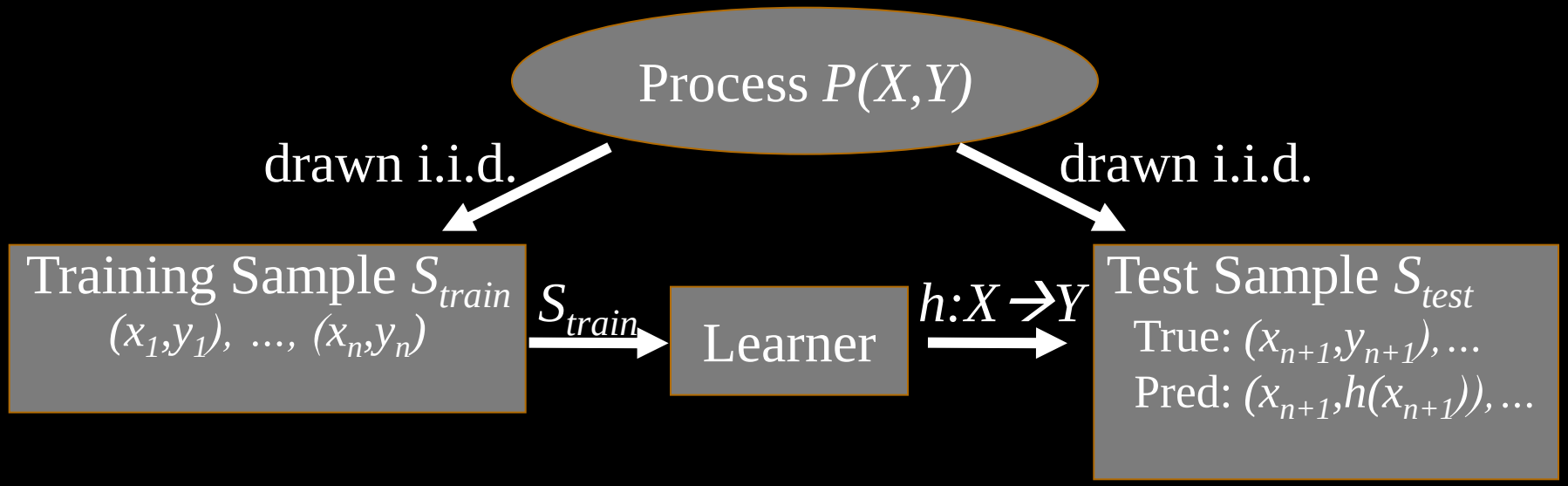
Machine Learning Tasks that are Relevant for CS6784

- 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)

Machine Learning Tasks that are Relevant for CS6784

- 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
- Online Learning
  - Data: (x,a,f), not necessarily iid
  - x: context; a: action; f: full or partial feedback
  - Learn: Policy π: x → A that minimized regret

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 $\rightarrow$ Y with low prediction error $Err_P(h)$

Sample Error and Generalization Error

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

Definition: $\Delta(a, b)$ *is the 0/1-loss function*

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

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

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

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

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

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

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

Discriminative Model

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

Generative Model

- Model *P(X,Y)* with distributions *P(Y,X|w)*
- Find w 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

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

- Equivalent Reformulations: For 0/1-Loss,

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

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

$$\hat{w} = \underset{w \in W}{\text{argmax}} \prod_{i=1}^{n} P(Y = y_i, X = x_i | w)$$

Multivariate Naïve Bayes' Classifier

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

$$\mathrm{P}(\mathrm{Y} = +1), \mathrm{P}(\mathrm{Y} = -1)$$

  - Class conditional model (one for each class)

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

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

- Classification rule:

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

Estimating the Parameters of Naïve Bayes

- 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: $\hat{w} = \underset{w \in W}{\operatorname{argmax}} \prod_{i=1}^{n} \left[P(Y_i = y_i) \prod_{j=1}^{N} P(X_i^{(j)} = x_i^{(j)} | Y_i = y_i) \right]$
  - P(Y): Maximum Likelihood Estimate

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

  - P(X|Y): Maximum Likelihood Estimate

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

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

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

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,w)*
  - Find *w* e.g. via MLE
  - Examples: Log. Reg., CRF
- Model discriminant functions
  - Find $h_w \in H$ with low train loss (e.g. Emp. Risk Min.)
  - Examples: SVM, Dec. Tree

Generative Model

- Model *P(X,Y)* with distributions *P(Y,X|w)*
- Find w 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: $h(x) = \underset{y \in Y}{\operatorname{argmin}} \left[\sum_{y' \in Y} \Delta(y', y) P(Y = y' | X = x) \right]$
  - Assume 0/1 Loss Δ(y,y') = 1, if y ≠ y', 0 else

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

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

$$\hat{w} = \underset{w \in W}{\operatorname{argmax}} \prod_{i=1}^{n} P(Y = y_i | X = x_i, w)$$

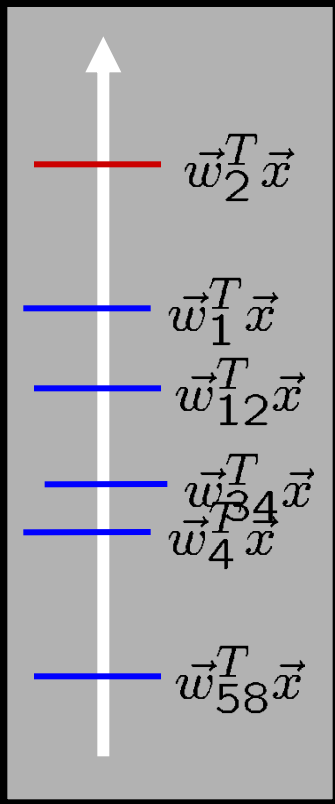
Logistic Regression

- Assume: $P(Y = y|X = x, w = (w_1, \dots, w_k)) = \frac{e^{w_y^T x}}{\sum_{y' \in Y} e^{w_y^T x}}$

  →Learn one weight vector w_y for each class $y \in Y$ = [1..k] (linear discriminant):

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

$$= \underset{y \in \text{Y}}{\text{argmax}} [w_y^T x]$$



- Maximum Likelihood training:

$$\hat{w} = \max_{w \in W} \left[\prod_{i=1}^{N} P(Y = y_i|X = x_i, w) \right]$$

$$= \max_{w \in W} \left[\sum_{i=1}^{n} w_y^T x_i - \log \left(\sum_{y' \in Y} e^{w_{y'}^T x_i} \right) \right]$$

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,w)*
  - Find *w* e.g. via MLE
  - Examples: Log. Reg., CRF
- Model discriminant functions
  - Find $h_w \in H$ with low train loss (e.g. Emp. Risk Min.)
  - Examples: SVM, Dec. Tree

Generative Model

- Model *P(X,Y)* with distributions *P(Y,X|w)*
- Find w 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: $X \times Y \rightarrow \Re$

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

$$= \underset{y \in Y}{\operatorname{argmin}} [h(x, y)] \quad (e.g., h(x, y) = [w_y^T x])$$

- 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_W and "large" S: Rule $h' \in H_W$ with best $Err_S(h')$ has $Err_P(h')$ close to $min_{h \in H} \{Err_P(h)\}$
- Learning: Empirical Risk Minimization (ERM)
  - Assume class H_W of discriminant functions h_w: $X \rightarrow Y$
  - Find

$$\hat{h}_w = \min_{h_w \in \mathrm{H}_W} \left[\sum_{i=1}^{n} \Delta(y_i, h_w(x_i)) \right]$$

Support Vector Machine

- Training Examples: $(x_1, y_1), \ldots, (x_n, y_n), x_i \in R^N, y_i \in \{1, -1\}$
- Hypothesis Space: $H_W = \{h(x) = sign[w^T x + b]: ||w|| \leq W\}$
- Training Loss: $Err_{train}(h) = \sum_{i=1}^{n} \Delta(y_i, h(x_i)) \leq \sum_{i=1}^{n} \xi_i$

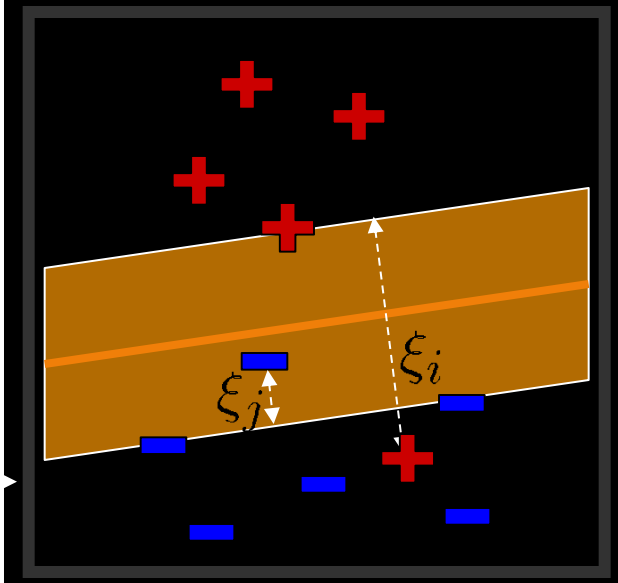
Optimization Problem:

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

$$s.t. \quad y_1(\vec{w}^T\vec{x}_1 + b) \geq 1 - \xi_1$$

$$\ldots$$

$$y_n(\vec{w}^T\vec{x}_n + b) \geq 1 - \xi_n$$



[Vapnik et al.]

[Crammer & Singer 02]

- Tra $k]$
- Hy

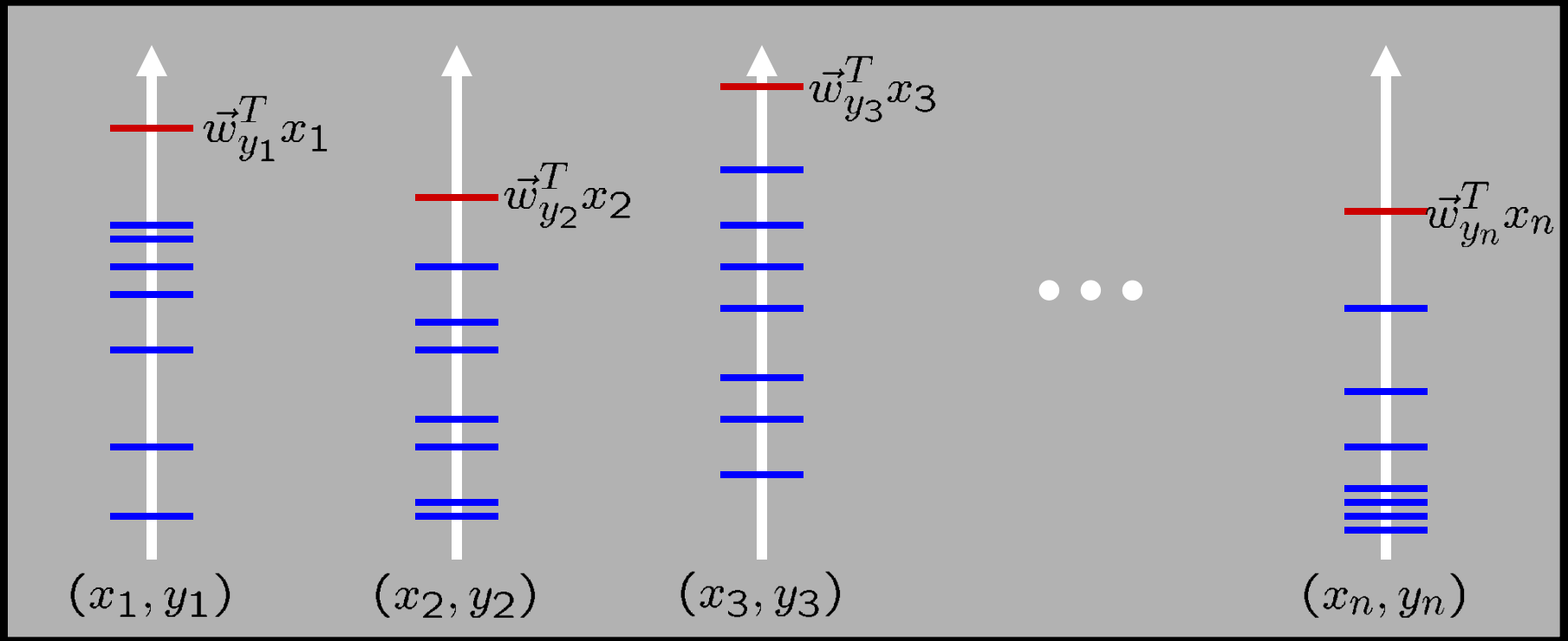
Training: Find $<\vec{w}_1, ..., \vec{w}_k>$ that solve

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

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

$$\ldots$$

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

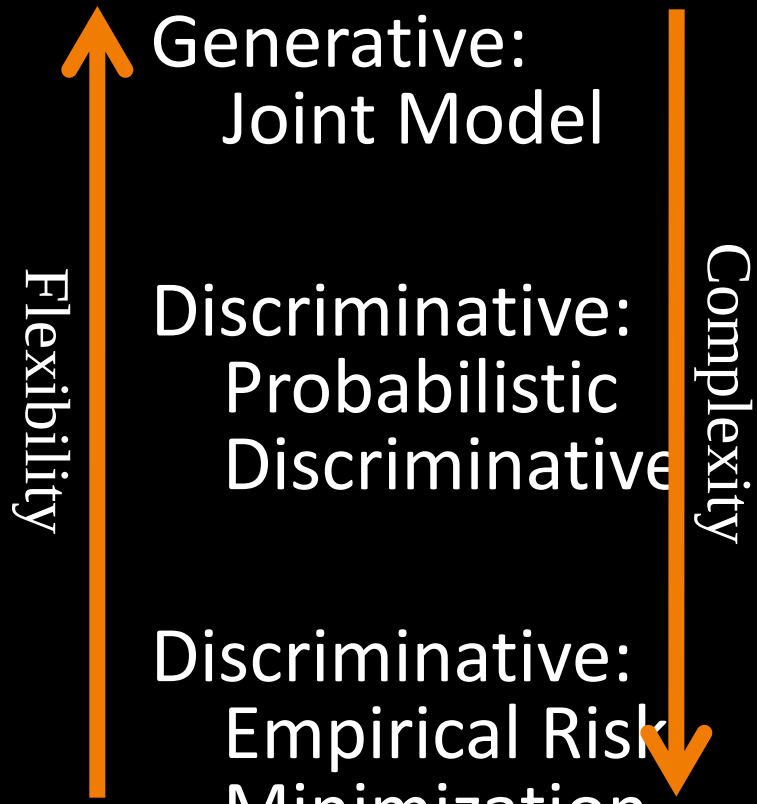


Types of Learning Methods

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

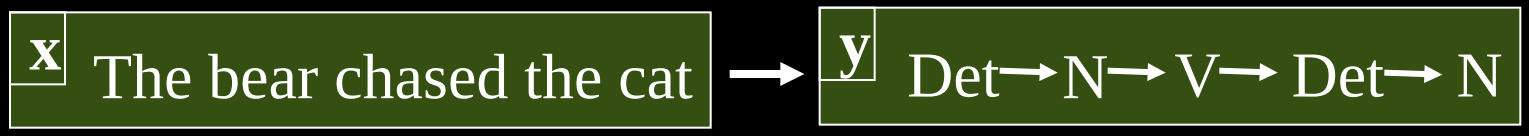
$$= \underset{y \in Y}{\operatorname{argmax}}[P(Y = y | X = x)]$$

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



So what about Structured Outputs?

- Approach: view as multi-class classification task
  - Every complex output y $\in$ Y is one class



- Problem: Exponentially many classes!
  - Generative Model: P(X,Y|w)
  - Discriminative Model: P(Y|X,w)
  - Discriminant Functions: h_w: $X \times Y \rightarrow \Re$
- Challenges
  - How to compactly represent model?
  - How to do efficient inference with model (i.e. argmax over Y)?
  - How to effectively estimate model from data? (e.g. solve ERM problem)