Lecture 17: EM and HMM
Winter 2018

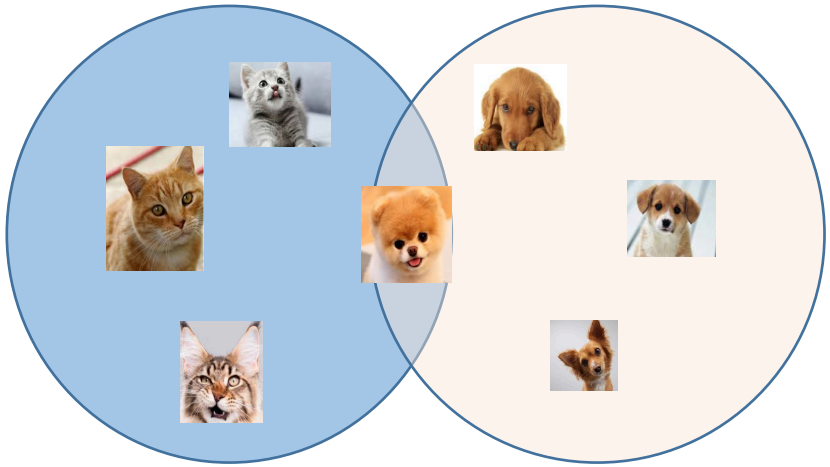
Kai-Wei Chang

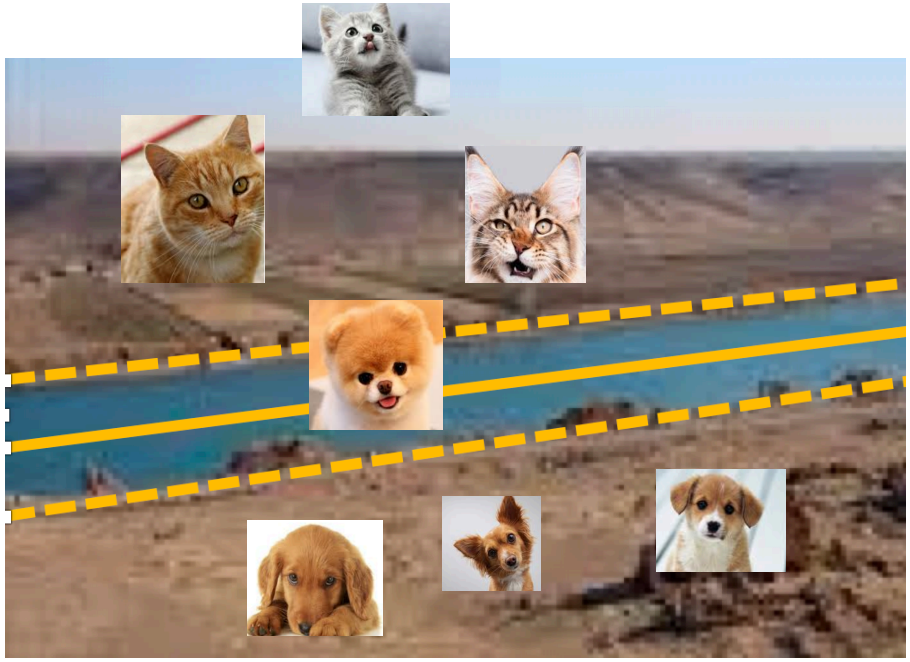
CS @ UCLA

kw+cm146@kwchang.net

The instructor gratefully acknowledges Dan Roth, Vivek Srikuar, Sriram Sankararaman, Fei Sha, Ameet Talwalkar, Eric Eaton, and Jessica Wu whose slides are heavily used, and the many others who made their course material freely available online.

Generative Model v.s. Discriminative model

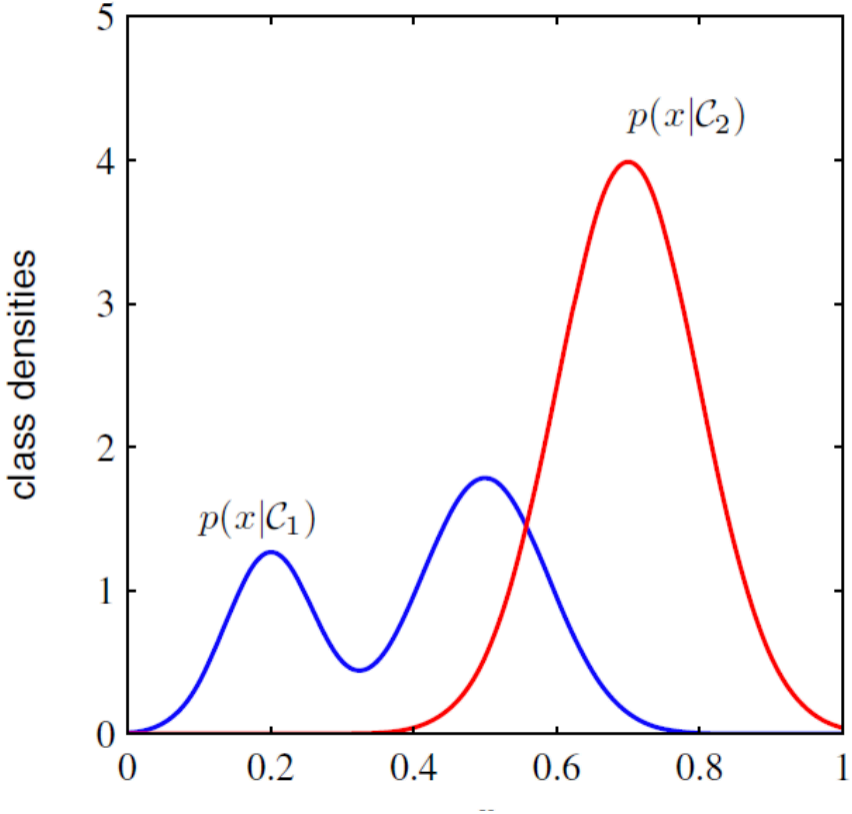




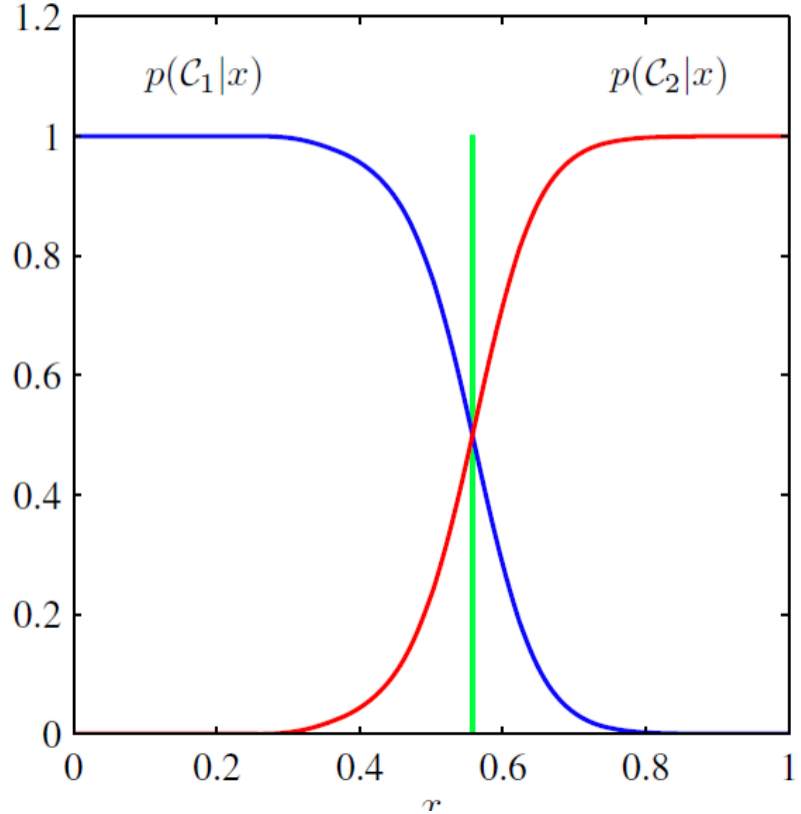
Learn $P(X, Y \mid \Theta)$ or
Learn $P(X \mid Y, \Theta)$ and $P(Y \mid \Theta)$

Learn $P(Y \mid X, \Theta)$

Generative Model's view

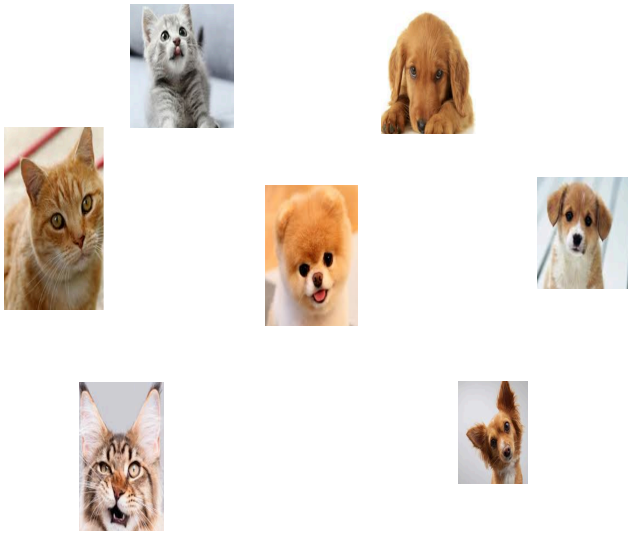


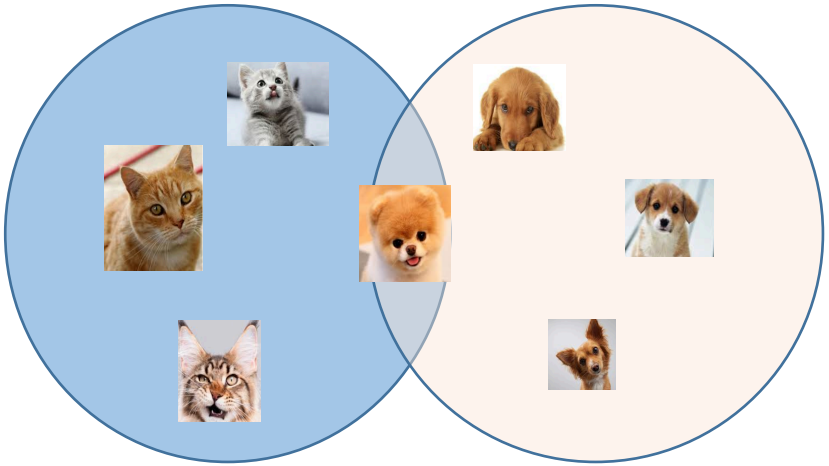
Discriminative Model's view



How about unsupervised learning

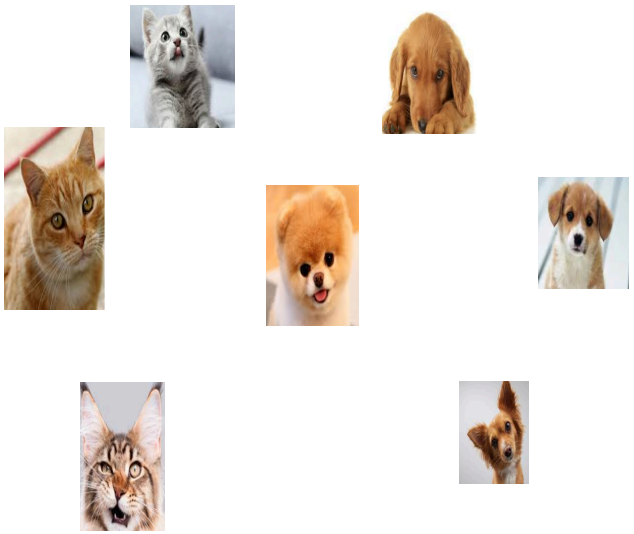
- In unsupervised learning, we only observed input distribution $\tilde{P}(X)$

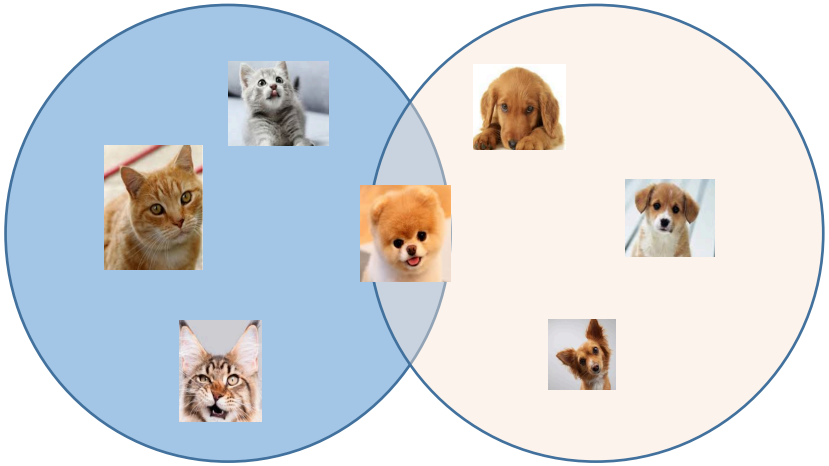




MLE in unsupervised learning

- We only have observation of $\tilde{P}(X)$
- In generative model, we have $P(X, Y \mid \Theta)$
- In discriminative model, we have $P(Y \mid \Theta, X)$
- Which model is more suitable for unsupervised learning?





MLE in unsupervised learning

- We only have observation of $\tilde{P}(X)$
- In generative model, we have $P(X, Y \mid \Theta)$
- We know $P(X \mid \Theta) = \sum_Y P(X, Y \mid \Theta)$
- Therefore, MLE is
  $\text{argmax}_\Theta P(X \mid \Theta) =$
  $\text{argmax}_\Theta \sum_Y P(X, Y \mid \Theta)$

EM algorithm

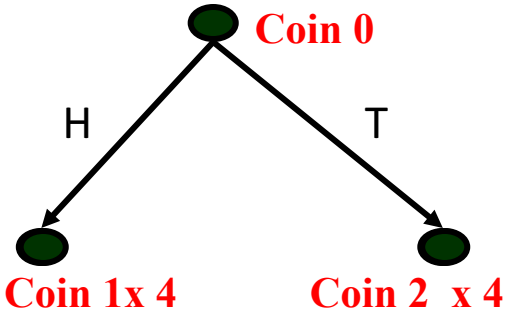
- EM algorithm solves $\text{argmax}_\Theta \sum_Y P(X, Y \mid \Theta)$ by iteratively updating Θ
- In general, known to converge to a local maximum of the maximum likelihood function

Three Coins Example

- We observe a series of coin tosses generated in the following way:
- A person has three coins.
  - Coin 0: probability of Head is α
  - Coin 1: probability of Head p
  - Coin 2: probability of Head q

- Consider the following coin-tossing scenarios:

Scenario I



- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

Observing the sequence
HHHHT, THTHT, HHHHT, HHTTH

produced by Coin 0 , Coin1 and Coin2

Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Scenario I

Supervised Learning

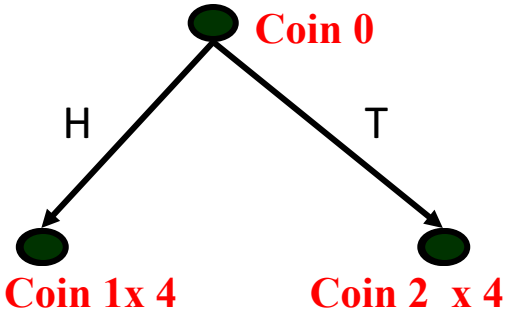
- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

  Observing the sequence
  HHHHT, THTHT, HHHHT, HHTTH

  produced by Coin 0 , Coin1 and Coin2

  Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Scenario II



- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

Observing the sequence
HHHT, HTHT, HHHT, HTTH

produced by ~~Coin 0~~, Coin1 and/or Coin2

Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Intuition of EM algorithm

- Use an iterative approach for estimating the parameters:
  - Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.
  - Now, compute the most likely value of the parameters. [recall the scenario I]
  - Compute the likelihood of the data given this model.
  - Re-estimate the initial parameter setting: set them to maximize the likelihood of the data.

Step 1: initialization

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.

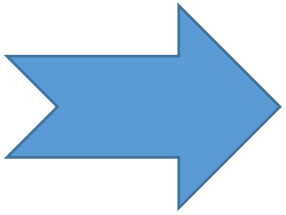
	coin 1=H	coin 1=T
HHHT	100%	0 %
HTHT	100%	0%
HHHT	100%	0%
HTTH	0%	100%

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

	coin 1=H	coin 1=T
HHHT	100%	0 %
HTHT	100%	0%
HHHT	100%	0%
HTTH	0%	100%



HHHHT
HHTHT
HHHHT
THTTH

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

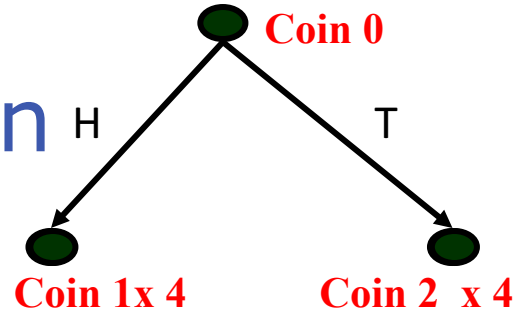
HHHHT
HHTHT
HHHHT
THTTH

$$\alpha_1 = \frac{3}{3+1} = \frac{3}{4}$$

$$p_1 = \frac{8}{8+4} = \frac{2}{3}$$

$$q_1 = \frac{2}{2+2} = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ $\quad$ $(1-\alpha_1)q_1^{\#H}(1-q_1)^{\#T}$

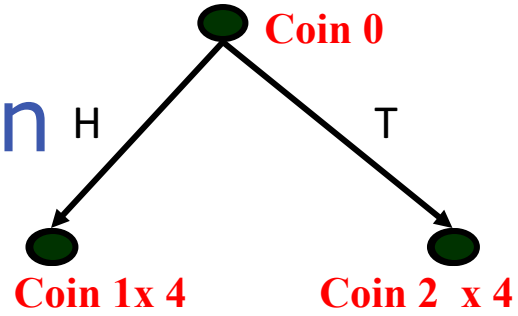
	coin 1=H	coin 1=T
HHHT	$\frac{3}{4}\left(\frac{2}{3}\right)^3\frac{1}{3}$	$\frac{1}{4}\left(\frac{1}{2}\right)^3\frac{1}{2}$
HTHT	$\frac{3}{4}\left(\frac{2}{3}\right)^2\left(\frac{1}{3}\right)^2$	$\frac{1}{4}\left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)^2$
HHHT	$\frac{3}{4}\left(\frac{2}{3}\right)^3\frac{1}{3}$	$\frac{1}{4}\left(\frac{1}{2}\right)^3\frac{1}{2}$
HTTH	$\frac{3}{4}\left(\frac{2}{3}\right)^2\left(\frac{1}{3}\right)^2$	$\frac{1}{4}\left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)^2$

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

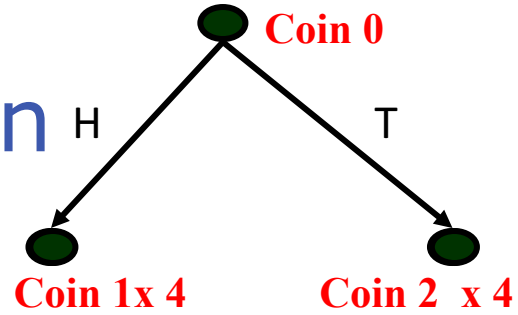
	$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ coin 1=H	$(1-\alpha_1)q_1^{\#H}(1-q_1)^{\#T}$ coin 1=T
HHHT	0.074	0.0156
HTHT	0.037	0.0156
HHHT	0.074	0.0156
HTTH	0.037	0.0156

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ $(1-\alpha_1) q_1^{\#H}(1-q_1)^{\#T}$

	coin 1=H	coin 1=T
HHHT	82.6%	17.4%
HTHT		29.7%
HHHT	82.6%	17.4%
HTTH	*70.3%*	29.7%

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

e.g., $\frac{0.074}{0.074+0.0156} = 82.6\%$

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

	coin 1=H	coin 1=T
HHHT	82.6%	17.4%
HTHT		29.7%
HHHT	82.6%	17.4%
HTTH	*70.3%*	29.7%

HHHHT 82.6%
THHHT 17.4%
HHTHT 70.3%
THTHT 29.7%
HHHHT 82.6%
THHHT 17.4%
HHTTH 70.3%
THTTH 29.7%

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

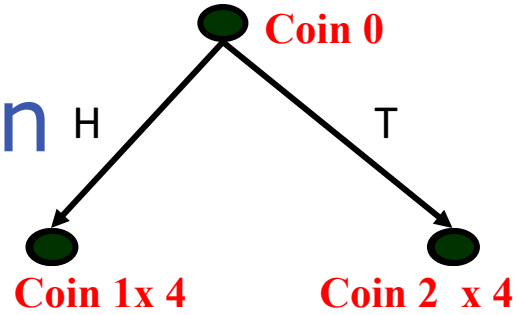
HHHHT 82.6%
HHTHT 70.3%
HHHHT 82.6%
HHTTH 70.3%
THHHT 17.4%
THTHT 29.7%
THHHT 17.4%
THTTH 29.7%

$$\alpha_2 = \frac{82.6\times 2 + 70.3\times 2}{400} = 76.5\%$$

$$p_2 = \frac{82.6\times 6 + 70.3\times 4}{82.6\times 8 + 70.3\times 8} = 63.5\%$$

$$q_2 = \frac{17.4\times 6 + 29.7\times 4}{17.4\times 8 + 29.7\times 8} = 59.2\%$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_2 p_2^{\#H}(1-p_2)^{\#T}$ coin 1=H

$(1-\alpha_2)q_2^{\#H}(1-q_2)^{\#T}$ coin 1=T

HHHT
HTHT
HHHT
HTTH

$$\alpha_2 = 76.5\%$$

$$p_2 = 63.5\%$$

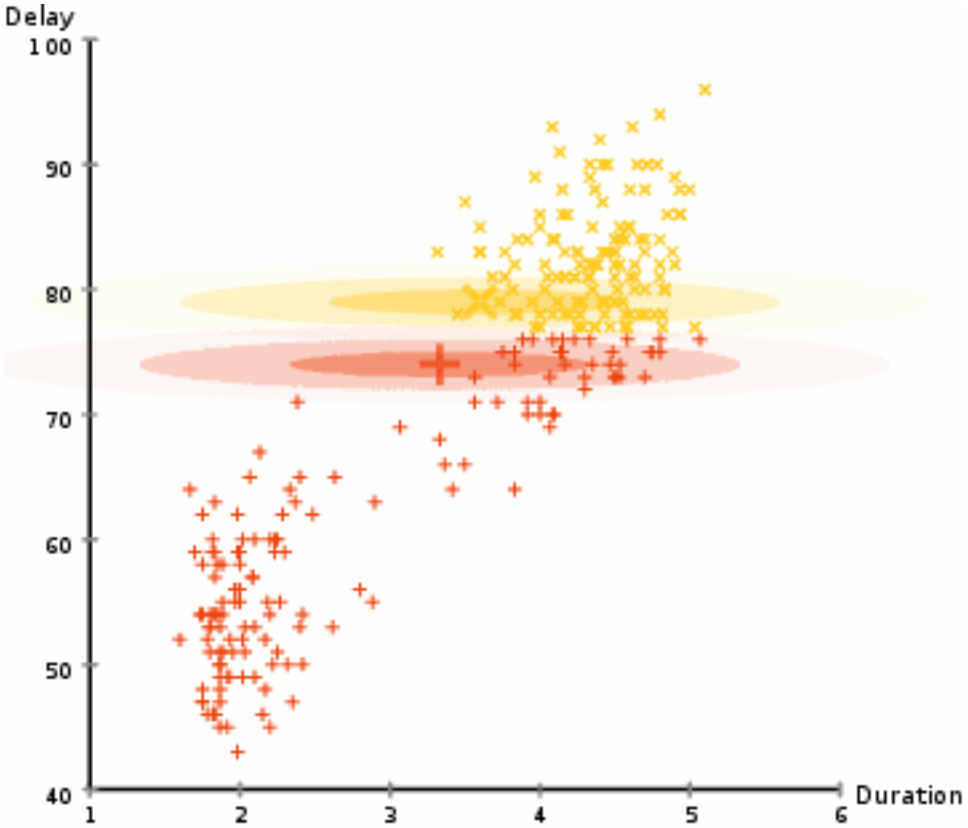
$$q_2 = 59.2\%$$

Intuition of EM algorithm

- Use an iterative approach for estimating the parameters:
  - Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.
  - Now, compute the most likely value of the parameters. [recall the scenario I]
  - Compute the likelihood of the data given this model.
  - Re-estimate the initial parameter setting: set them to maximize the likelihood of the data.

Real world Example

GMM clustering of Old Faithful eruption data



GMM as the marginal distribution P(x) of a joint distribution P(x, z)

- Remember, in GMM, we model the marginal probability as

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$\omega_k = p(z = k)$$

Iterative procedure

- LetType equation here. θ represent all parameters $\{\omega_k, \mu_k, \Sigma_k\}$

Step 0: initialize $\boldsymbol{\theta}$ with some values (random or otherwise)

Step 1: compute γ_{nk} using the current $\boldsymbol{\theta}$

Step 2: update $\boldsymbol{\theta}$ using the just computed γ_{nk}

Step 3: go back to Step 1

γ_{nk}: given a data point x_n how likely it belongs to k^{th} cluster

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Since γ_{nk} is binary, the previous solution is nothing but

- For ω_k: count the number of data points whose z_n is k and divide by the total number of data points (note that $\sum_k \sum_n \gamma_{nk} = N$)
- For $\boldsymbol{\mu}_k$: get all the data points whose z_n is k, compute their mean
- For $\boldsymbol{\Sigma}_k$: get all the data points whose z_n is k, compute their covariance matrix

Estimate γ_{nk}

- γ_{nk} the assignment of instance n to cluster k, can be defined as $\gamma_{nk} = P(z_n = k|x_n)$
- Can be computed vis the posterior probability

$$p(z_n = k|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{p(\boldsymbol{x}_n)} = \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{\sum_{k'=1}^{K} p(\boldsymbol{x}_n|z_n = k')p(z_n = k')}$$

$N(x|\mu_k, \Sigma_k)$ ω_k

Parameter estimation for GMMs

- If cluster assignments are observed $\{z_n\}$ are given
  - We know the cluster of each point
  - Let $\gamma_{nk} = 1$ if instance n belongs to cluster k, otherwise $\gamma_{nk} = 0$
- Then the maximum likelihood estimation is

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Hidden Markov Model

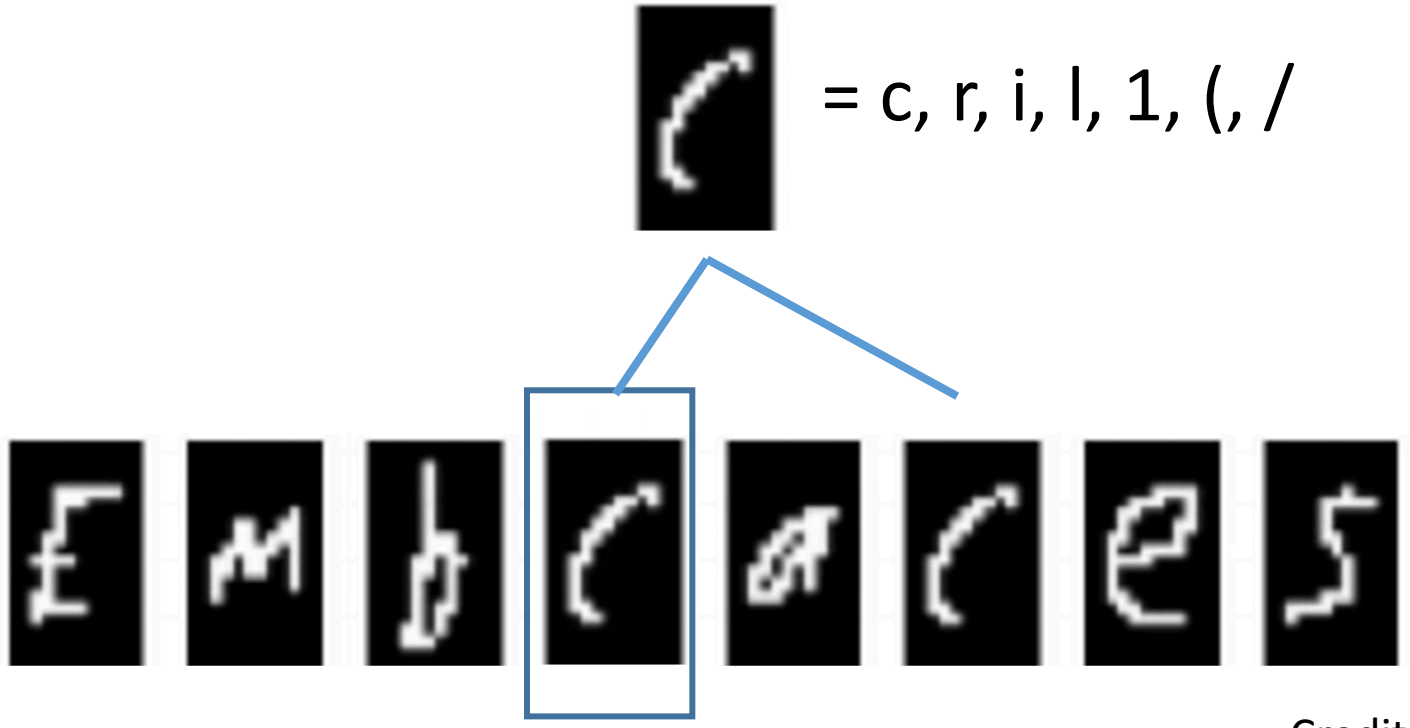
-- Go beyond binary/multiclass classification

Previous Lectures

- Binary linear classification
  - Perceptron, SVMs, Logistic regression, Naïve Bayes
  - Output: $y \in \{1, -1\}$
- Multi-class classification
  - Multiclass Perceptron, Multiclass SVM…
  - Output: $y \in \{1,2,3, \ldots K\}$

Why is structure important?
Hand written recognition example

- What is this English letter?



Credit: Ben Taskar

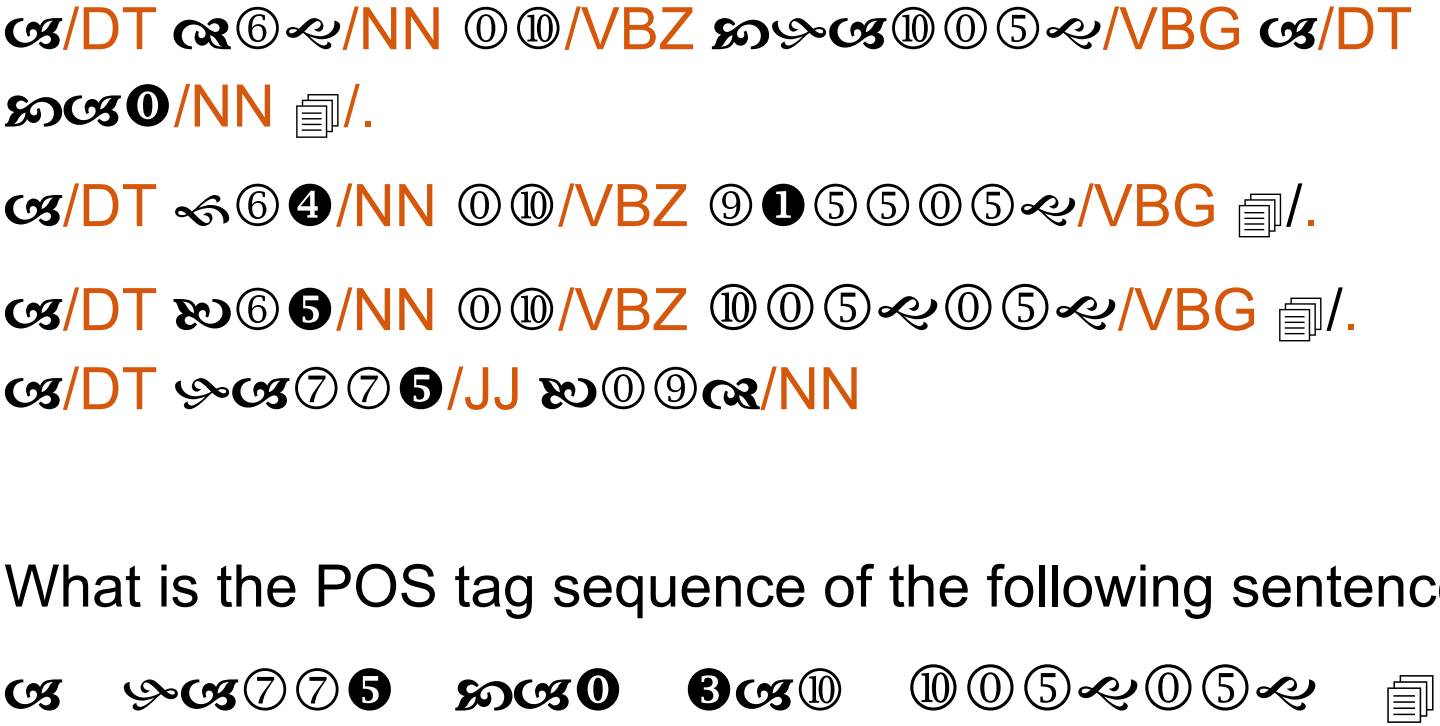
Sequential tagging

- The process of assigning a part-of-speech to each word in a collection (sentence).

WORD	tag
the	**DET**
koala	**N**
put	**V**
the	**DET**
keys	**N**
on	**P**
the	**DET**
table	**N**

Let's try

Don't worry! There is no problem with your eyes or computer.



What is the POS tag sequence of the following sentence?

Let’s try

- 𝒢/DT 𝒸⑥𝓈/NN ⓪⑩/VBZ 𝓃𝒽𝒢⑩⓪⑤𝓈/VBG 𝒢/DT 𝓃𝒽𝒢❶/NN 🗐/.
  a/DT dog/NN is/VBZ chasing/VBG a/DT cat/NN ./.
- 𝒢/DT 𝒻⑥❹/NN ⓪⑩/VBZ ⑨❶⑤⑤⓪⑤𝓈/VBG 🗐/.
  a/DT fox/NN is/VBZ running/VBG ./.
- 𝒢/DT 𝓫⑥❺/NN ⓪⑩/VBZ ⑩⓪⑤𝓈⓪⑤𝓈/VBG 🗐/.
  a/DT boy/NN is/VBZ singing/VBG ./.
- 𝒢/DT 𝒽𝒢⑦⑦❺/JJ 𝓫⓪⑨𝒸/NN
  a/DT happy/JJ bird/NN
- 𝒢 𝒽𝒢⑦⑦❺ 𝓃𝒢❶ ❸𝒢⑩ ⑩⓪⑤𝓈⓪⑤𝓈 🗐
  a happy cat was singing .

How you predict the tags?

- Two types of information are useful
  - Relations between words and tags
  - Relations between tags and tags
    - DT NN, DT JJ NN…
    - Fed in "The Fed" is a Noun because it follows a Determiner
- One possible model
  - Each output label is dependent on its neighbors in addition to the input

Outline

- *Sequence models*
- Hidden Markov models
  - Inference with HMM
  - (Supervised) Learning for HMM

Sequences

- Sequences of states
  - Text is a sequence of words or even letters
  - A video is a sequence of frames
- Our goal (for now):
  Define probability distributions over sequences
- If $x_1, x_2, \cdots, x_n$ is a sequence that has n tokens, we want to be able to define

$$P(x_1, x_2, \cdots, x_n)$$

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \cdots, x_{i-1})$$

- Each token is dependent on all the tokens that came before it
  - Simple conditioning
  - Each $P(x_i \mid \ldots)$ is a multinomial probability distribution over the tokens

Example: A Language model

It was a bright cold day in April.

$$P(\text{It was a bright cold day in April}) =$$
$$P(\text{It}) \times \quad \leftarrow \text{Probability of a word starting a sentence}$$
$$P(\text{was}|\text{It}) \times \quad \leftarrow \text{Probability of a word following “It”}$$
$$P(\text{a}|\text{It was}) \times \quad \leftarrow \text{Probability of a word following “It was”}$$
$$P(\text{bright}|\text{It was a}) \times \quad \leftarrow \text{Probability of a word following “It was a”}$$
$$P(\text{cold}|\text{It was a bright}) \times$$
$$P(\text{day}|\text{It was a bright cold}) \times \cdots$$

A history-based model

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | x_1, x_2, \cdots, x_{i-1})$$

- Each token is dependent on all the tokens that came before it
  - Simple conditioning
  - Each $P(x_i \mid \ldots)$ is a multinomial probability distribution over the tokens
- What is the problem here?
  - How many parameters do we have?
    - Grows with the size of the sequence!

Solution: Lose the history

Discrete Markov Process

- A system can be in one of K states at a time
- State at time t is x_t
- First-order Markov assumption

  The state of the system at any time is ***independent*** of the full sequence history given the previous state

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i | x_{i-1})$$

  - Defined by two sets of probabilities:
    - Initial state distribution: $P(x_1 = S_j)$
    - State transition probabilities: $P(x_i = S_j \mid x_{i-1} = S_k)$

Example: Another language model

It was a bright cold day in April

$$P(\text{It was a bright cold day in April}) =$$
$$P(\text{It}) \times$$
$$P(\text{was}|\text{It}) \times$$
$$P(\text{a}|\text{was}) \times$$
$$P(\text{bright}|\text{a}) \times$$
$$P(\text{cold}|\text{bright}) \times$$
$$P(\text{day}|\text{cold}) \times \cdots$$

Probability of a word starting a sentence

Probability of a word following "It"

Probability of a word following "was"

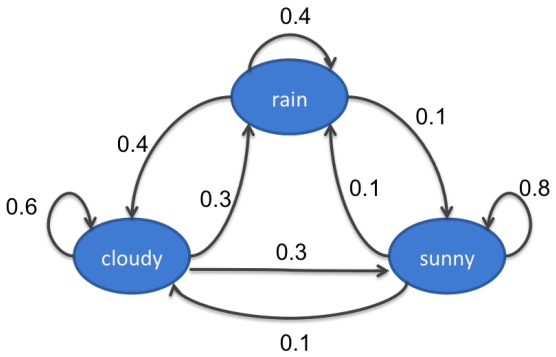
Probability of a word following "a"

If there are K tokens/states, how many parameters do we need? $O(K^2)$

Example: The weather

- Three states: rain, cloudy, sunny

State transitions:



- Observations are Markov chains:

Eg: *cloudy sunny sunny rain*

Probability of the sequence =

P(cloudy) P(sunny|cloudy) P(sunny | sunny) P(rain | sunny)

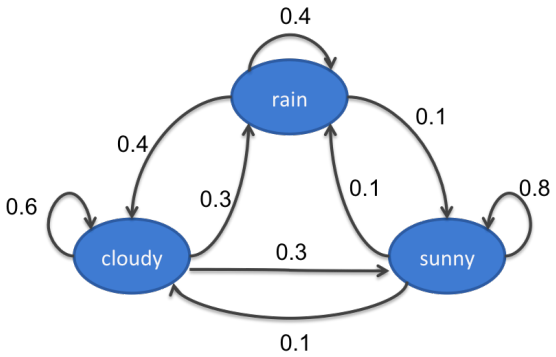
Initial probability

Transition probabilities

Example: The weather

- Three states: rain, cloudy, sunny

State transitions:



- Obse

These probabilities define the model; can find P(any sequence)

Eg: *cl*

Probability of the sequence =

P(cloudy) P(sunny|cloudy) P(sunny | sunny) P(rain | sunny)

Initial probability

Transition probabilities

m^{th} order Markov Model

- A generalization of the first order Markov Model
  - Each state is only dependent on m previous states
  - How many parameters do you need?

Outline

- *Sequence models*
- Hidden Markov models
  - Inference with HMM
  - Supervised Learning for HMM

Hidden Markov Model

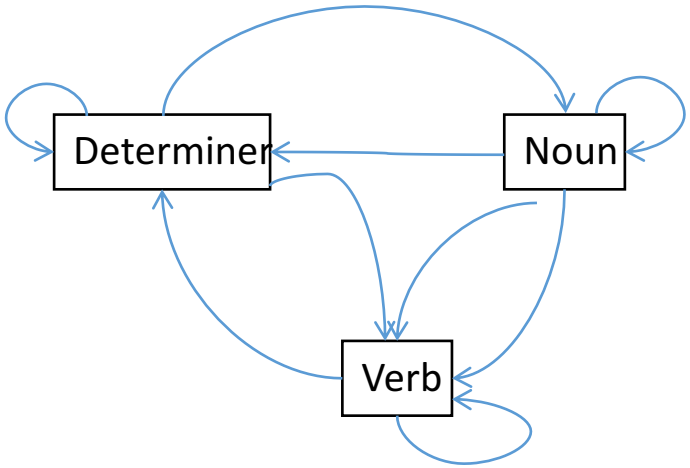
- Discrete Markov Model:
  - States follow a Markov chain
  - *Each state is an observation*

- Hidden Markov Model:
  - States follow a Markov chain
  - States are not observed
  - Each state stochastically emits an observation

Toy part-of-speech example

The Fed raises interest rates

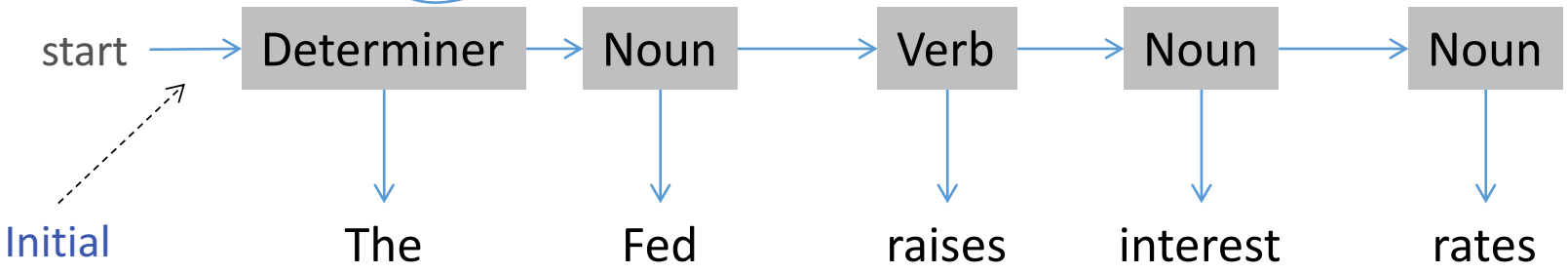
Transitions



Emissions

P(The | Determiner) = 0.5
P(A | Determiner) = 0.3
P(An | Determiner) = 0.1
P(Fed | Determiner) = 0
…

P(Fed| Noun) = 0.001
P(raises| Noun) = 0.04
P(interest| Noun) = 0.07
P(The| Noun) = 0
…



Joint model over states and observations

- Notation
  - Number of states = K, Number of observations = M
  - π: Initial probability over states (K dimensional vector)
  - A: Transition probabilities (K×K matrix)
  - B: Emission probabilities (K×M matrix)

- Probability of states and observations
  - Denote states by $y_1, y_2, \cdots$ and observations by $x_1, x_2, \cdots$

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$= \pi_{y_1} \prod_{i=1}^{n-1} A_{y_i, y_{i+1}} \prod_{i=1}^{n} B_{y_i, x_i}$$

Jason and his Ice Creams

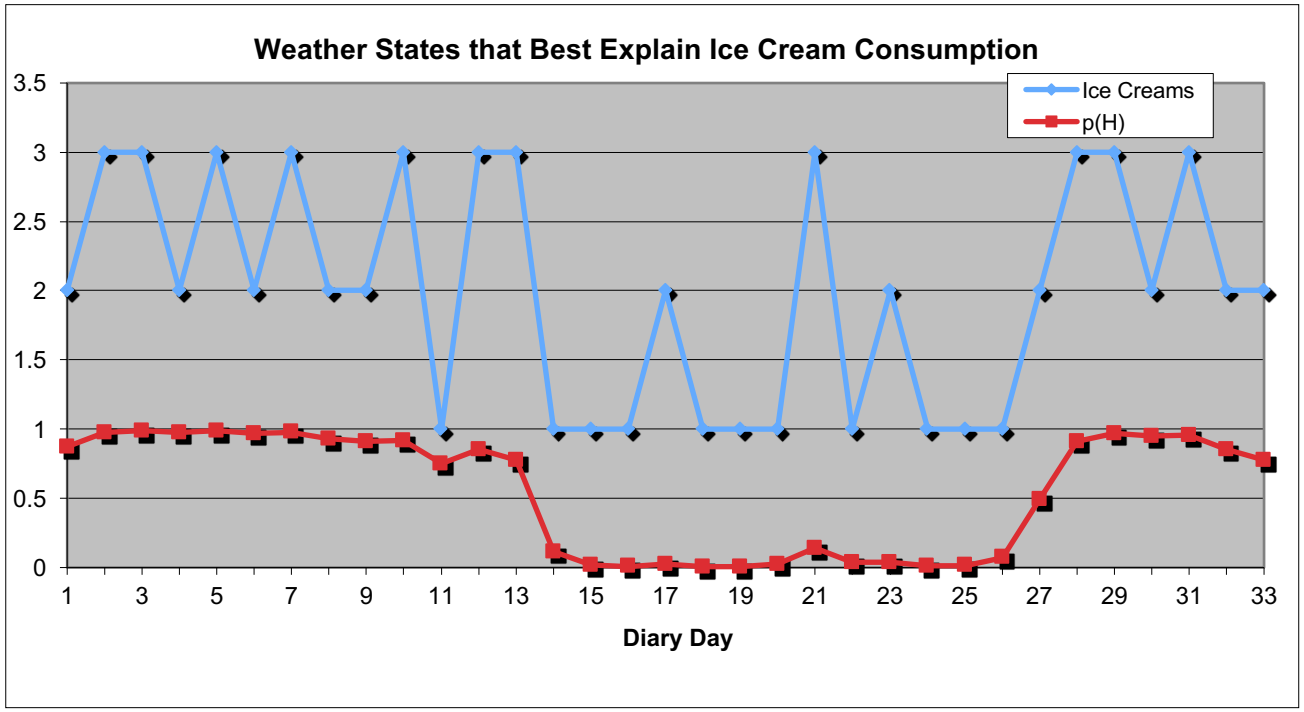
- You are a climatologist in the year 2799
- Studying global warming
- You can't find any records of the weather in Baltimore, MA for summer of 2007
- But you find Jason Eisner's diary
- Which lists how many ice-creams Jason ate every date that summer
- Our job: figure out how hot it was

http://videolectures.net/hltss2010_eisner_plm/
http://www.cs.jhu.edu/~jason/papers/eisner.hmm.xls

(C)old day v.s. (H)ot day

#cones

	p(…|C)	p(…|H)	p(…|START)
(1|…)	0.7	0.1	
(2|…)	0.2	0.2	
(3|…)	0.1	0.7	
(C|…)	0.8	0.1	0.5
(H|…)	0.1	0.8	0.5
…)	0.1	0.1	0



Other applications

- Speech recognition
  - Input: Speech signal
  - Output: Sequence of words
- NLP applications
  - POS Tagging
- Computational biology
  - Aligning protein sequences
  - Labeling nucleotides in a sequence as exons, introns, etc.

Questions?

Three questions for HMMs

[Rabiner 1999]

1. Given an observation sequence, $x_1, x_2, \cdots x_n$ and a model (π, A, B), how to efficiently calculate the probability of the observation?
2. Given an observation sequence, $x_1, x_2, \cdots, x_n$ and a model (π, A, B), how to efficiently calculate the most probable state sequence?

Inference

3. How to calculate (π, A, B) from observations?

Learning

Outline

- *Sequence models*
- Hidden Markov models
  - Inference with HMM
  - Supervised Learning for HMM

Most likely state sequence

- Input:
  - A hidden Markov model (π, A, B)
  - An observation sequence $\mathbf{x} = (x_1, x_2, \cdots, x_n)$

- Output: A state sequence $\mathbf{y} = (y_1, y_2, \cdots, y_n)$ that corresponds to
  - Maximum *a posteriori* inference (MAP inference)

$$\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}, \pi, A, B)$$

- Computationally: combinatorial optimization

MAP inference

- We want $\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}, \pi, A, B)$
- We have defined

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

- But $P(\mathbf{y}|\mathbf{x}, \pi, A, B) \propto P(\mathbf{x}, \mathbf{y}|\pi, A, B)$
  - And we don't care about P(**x**) we are maximizing over **y**
- So, $\arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}, \pi, A, B) = \arg\max_{\mathbf{y}} P(\mathbf{y}, \mathbf{x}|\pi, A, B)$

How many possible sequences?

The	Fed	raises	interest	rates
Determiner	Verb Noun	Verb Noun	Verb Noun	Verb Noun
1	2	2	2	2

List of allowed tags for each word

In this simple case, 16 sequences (1×2×2×2×2)

Naïve approaches

1. Try out every sequence
   - Score the sequence **y** as $P(\mathbf{y}|\mathbf{x}, \pi, A, B)$
   - Return the highest scoring one
   - What is the problem?
     - Correct, but slow, $O(K^n)$
2. Greedy search
   - Construct the output left to right
   - For each i, elect the best y_i using y_{i-1} and x_i
   - What is the problem?
     - Incorrect but fast, $O(nK)$

Solution: Use the independence assumptions

Recall: The first order Markov assumption

The state at token i is only influenced by the previous state, the next state and the token itself

Given the adjacent labels, the others do not matter

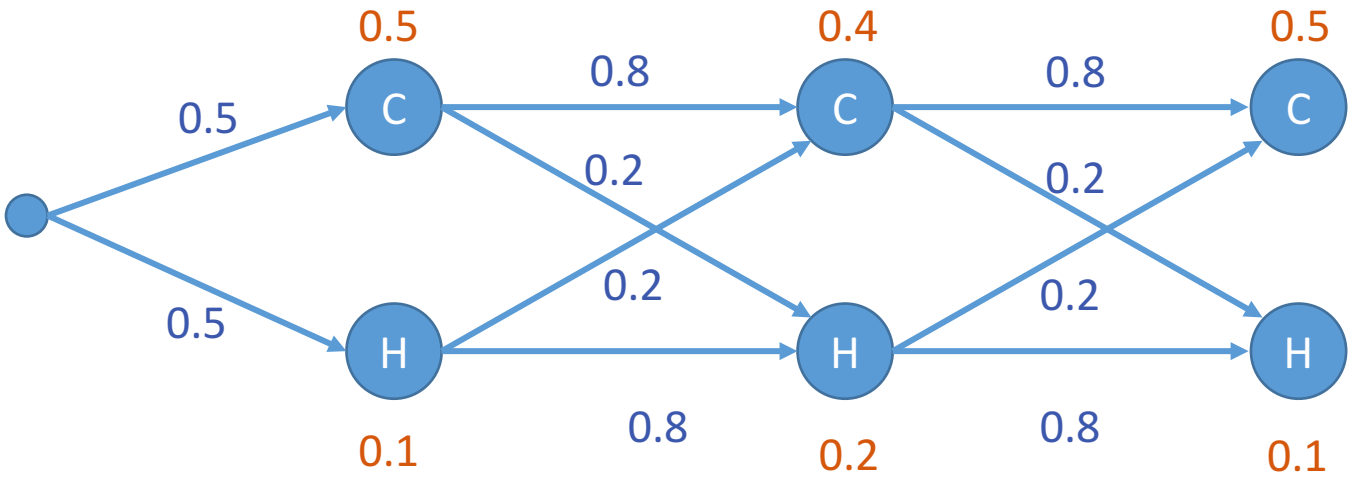
Suggests a recursive algorithm

Jason's ice cream

#cones

	p(…|C)	p(…|H)	p(…|START)
(1|…)	**0.5**	**0.1**	
(2|…)	**0.4**	**0.2**	
(3|…)	**0.1**	**0.7**	
(C|…)	**0.8**	**0.2**	**0.5**
(H|…)	**0.2**	**0.8**	**0.5**

- Best tag sequence for P("1,2,1")?



Viterbi algorithm

π: Initial probabilities
A: Transitions
B: Emissions

Max-product algorithm for first order sequences

1. Initial: For each state s, calculate

$$\text{score}_1(s) = P(s)P(x_1|s) = \pi_s B_{x_1,s}$$

1. Recurrence: For i = 2 to n, for every state s, calculate

$$\begin{aligned}\text{score}_i(s) &= \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\text{score}_{i-1}(y_{i-1}) \\ &= \max_{y_{i-1}} A_{y_{i-1},s}B_{s,x_i}\text{score}_{i-1}(y_{i-1})\end{aligned}$$

1. Final state: calculate

$$\max_{\mathbf{y}} P(\mathbf{y}, \mathbf{x}|\pi, A, B) = \max_s \text{score}_n(s)$$

This only calculates the max. To get final answer (*argmax*),

- keep track of which state corresponds to the max at each step
- build the answer using these back pointers

Questions?

General idea

- Dynamic programming
  - The best solution for the full problem relies on best solution to sub-problems
  - Memoize partial computation

- Examples
  - Viterbi algorithm
  - Dijkstra's shortest path algorithm
  - …

Complexity of inference

- Complexity parameters
  - Input sequence length: n
  - Number of states: K
- Memory
  - Storing the table: nK (scores for all states at each position)
- Runtime
  - At each step, go over pairs of states
  - $O(nK^2)$

Outline

- *Sequence models*
- Hidden Markov models
  - Inference with HMM
  - Supervised Learning for HMM

Learning HMM parameters

- Assume we know the number of states in the HMM
- Two possible scenarios

1. We are given a data set $D = \{<\mathbf{x}_i, \mathbf{y}_i>\}$ of sequences labeled with states

   And we have to learn the parameters of the HMM (π, A, B)

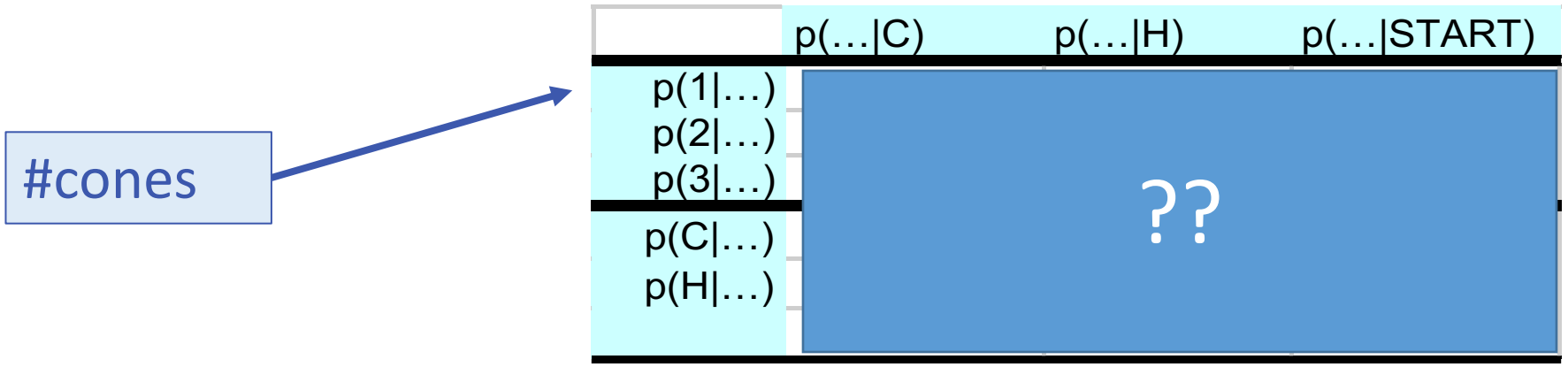
   Supervised learning with complete data

2. We are given only a collection of sequences $D = \{\mathbf{x}_i\}$

   And we have to learn the parameters of the HMM (π, A, B)

   Unsupervised learning, with incomplete data

Jason's ice cream



- Can we figure out (π, A, B) from history records?

#cones	1	2	1	1	2	3	3	3	3	2	3	1	2
Hot/Cold	C	C	C	C	H	H	H	H	C	C	H	H	H

#cones	3	2	2	2	2	3	1	1	1	2	3	2	2
Hot/Cold	H	H	H	C	C	H	H	C	C	H	H	H	C

Supervised learning of HMM

- We are given a dataset D = {<$\mathbf{x}_i$, $\mathbf{y}_i$>}
  - Each $\mathbf{x}_i$ is a sequence of observations and $\mathbf{y}_i$ is a sequence of states that correspond to $\mathbf{x}_i$

  Goal: Learn initial, transition, emission distributions (π, A, B)

- How do we learn the parameters of the probability distribution?
  - The maximum likelihood principle

Where have we seen this before?

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi,A,B} P(D|\pi, A, B) = \max_{\pi,A,B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

And we know how to write this in terms of the parameters of the HMM

Supervised learning details

$$(\hat{\pi}, \hat{A}, \hat{B}) = \max_{\pi,A,B} P(D|\pi, A, B) = \max_{\pi,A,B} \prod_i P(\mathbf{x}_i, \mathbf{y}_i|\pi, A, B)$$

π, A, B can be estimated separately just by counting

- Makes learning simple and fast

[Exercise: Derive the following using derivatives of the log likelihood. Requires Lagrangian multipliers.]

Number of instances where the first state is s

$$\pi_s = \frac{\text{count}(\text{start} \rightarrow s)}{n}$$

Number of examples

Initial probabilities

$$A_{s',s} = \frac{\text{count}\,(s \rightarrow s')}{\text{count}\,(s)}$$

Transition probabilities

$$B_{s,x} = \frac{\text{count}\begin{pmatrix} s \\ \downarrow \\ x \end{pmatrix}}{\text{count}\,(s)}$$

Emission probabilities

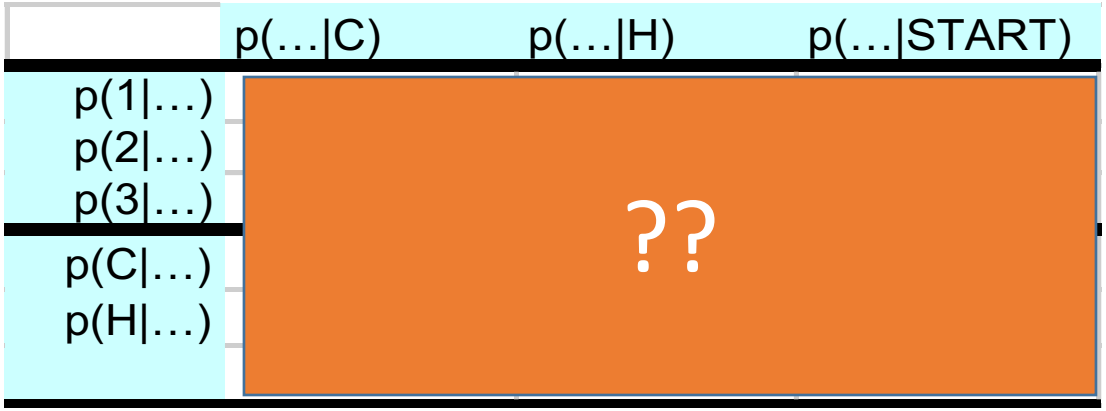
Priors and smoothing

- Maximum likelihood estimation works best with lots of annotated data
  - Never the case
- Priors inject information about the probability distributions
- Effectively additive smoothing
  - Add small constants to the counts

Question I haven't answer

- Can we figure out (π, A, B) from just observation

#cones



#cones	1	2	1	1	2	3	3	3	3	2	3	1	2
Hot/Cold	??												

#cones	3	2	2	2	2	3	1	1	1	2	3	2	2
Hot/Cold	??												

Hidden Markov Models summary

- Predicting sequences
  - As many output states as observations
- Markov assumption helps decompose the score
- Several algorithmic questions
  - Most likely state (decoding)
  - Learning parameters
    - Supervised, Unsupervised