# CSCI 561
# Foundation for Artificial Intelligence

# 23: Bayesian Learning, Unsupervised Learning, and EM Algorithms

Professor Wei-Min Shen
University of Southern California

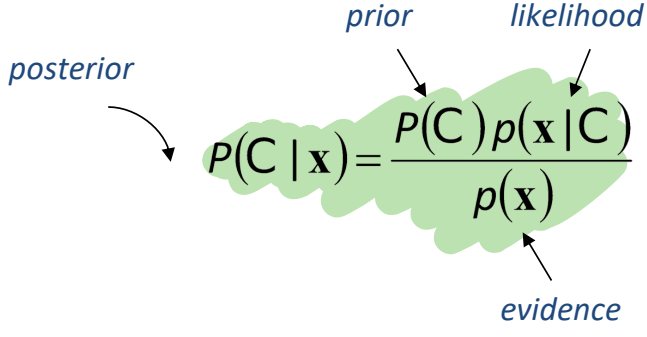
# Outline

- Bayesian Learning
- Unsupervised Learning (clustering)
  - Naïve Bayes Models (e.g., Autoclasses)
  - K-Means Clustering Algorithm
- The EM algorithm
  - The Crown-Jewel of Machine Learning !!!
  - I am so proud that you are learning this ☺️

# Notations Used (for your preparation)

- Data
  - $D=\{D_1,D_2, \ldots\}$ or sometimes $X=\{x_1,x_2, \ldots\}$
- Concepts, Theories, or Hypotheses
  - $C=\{C_1,C_2, \ldots\}$ or sometimes $X=\{X_1,X_2, \ldots\}$
- Background knowledge/information at time t = 0,1,2,…
  - $\{X_{t=0}, X_{t=1}, X_{t=2}, \ldots\}$ or sometimes $\{C_{t=0}, C_{t=1}, C_{t=2}, \ldots\}$

# Bayes' Rule




// $\boldsymbol{x}$ is data $D_t$
// $C$ is the concept or theory

$$P(C=0)+P(C=1)=1$$

$$p(\mathbf{x})=p(\mathbf{x}\mid C=1)P(C=1)+p(\mathbf{x}\mid C=0)P(C=0)$$

$$p(C=0\mid \mathbf{x})+P(C=1\mid \mathbf{x})=1$$

# Bayesian Learning

// $D_t$ is data at time $t$
// $C_j$ is the concept $j$ or theory $j$
// $X_t$ is the background knowledge at time $t$

- Objective
  - Given a set of new data $D$, and background knowledge $X$
  - Predict a concept $C_j$ where P($C_j$|$DX$) is the most probable
- Incremental Learning (*always learning*)
  - When training data come one at a time, t = 0, 1, …, where $X_{t+1}$ is the (background) knowledge learned at time $t$

$$P(C_j \mid X_{t+1}) \leftarrow P(C_j \mid D_t X_t) = P(C_j \mid X_t)\frac{P(D_t \mid C_j X_t)}{P(D_t \mid X_t)}$$

- Batch Learning (*learn once for all*)
  - When all the training example data $D$ are known at the outset

$$P(C_j \mid D) = P(C_j)\frac{P(D \mid C_j)}{P(D)}$$

# Bayes Rule for Incremental Learning

// $D_t$ is data
// $C_j$ is the concept or theory
// $X_t$ is the background knowledge at time t

- Bayesian is powerful for incremental learning
  - Let $C$ be "your hypothesis theory"; $D_t$ the "new data" at time t; and $X_t$ the "background information" at time $t$

Your new theory "$C$" after learning from data "$D_t$"

Your previous theory "$C$" before this learning step

How well is your previous theory "$C$" explaining the new data "$D_t$"

$$P(C_j \mid X_{t+1}) \leftarrow P(C_j \mid D_t X_t) = P(C_j \mid X_t) \frac{P(D_t \mid C_j X_t)}{P(D_t \mid X_t)}$$

The result at time $t$ becomes the prior background knowledge for time $t$+1

The likelihood of seeing the new data "$D_t$" regardless of your theory

# An Example: Estimate a Coin Incrementally

- Problem Statement
  - Randomly select ***X*** from two biased coins: {***A***(0.25), ***B***(0.75)}
  - Repeat this step at time ($t$ = 1, 2, 3, …)
    - Flip ***X***, observe the result $\boldsymbol{D_t}$ = {H(ead), T(ail)}
    - Estimate which was the selected coin? That is: ***X=A?*** or ***X=B?***
  - Assume the following
    - Your background knowledge $C_{t=0}$ at time=0 is that you did not know which X is
    - Your experiment results are ***D*** = {T, T, H, …. }

// $D_t$ is data at time $t$
// $X_j$ is the concept or theory
// $C_t$ is the background knowledge at time t

- Method: use the Bayesian Formula (see later slice)

# An Example: Estimate a Coin Incrementally

- Problem Statement
  - Randomly select ***X*** from two biased coins:
    {***A***(0.25), ***B***(0.75)}
  - Repeat this at each time step ($t$ = 1, 2, 3, …)
    - Flip ***X*** and observe the result $\boldsymbol{D_t}$ = {H(ead), T(ail)}
    - Estimate ***X=A?*** or ***X=B?***

- Two hypotheses: $A$, $B$;
- Background Knowledge at time $t$: $C_t$
- Incremental Bayesian Learning
  - $P(A|D_{t+1}C_t) = P(A|C_t)P(D_{t+1}|AC_t)/P(D_{t+1}|C_t)$
  - $P(B|D_{t+1}C_t) = P(B|C_t)P(D_{t+1}|BC_t)/P(D_{t+1}|C_t)$

What if there were 3 or $n$ hypotheses?

Assume the results of your experiments are ***D*** = {H, H, T, H, …. }

$i$ = 0:

$P(A|C_0) = ½$, $P(B|C_0) = ½$ // $C_0$ is "don't know which coin was selected"

$i$ = 1:

$D_1$ = H, $P(A|C_0)=½$, $P(B|C_0)=½$

$P(A|D_1C_0) = P(A|C_0)P(D_1|AC_0)/P(D_1|C_0) = ½ ¼ / [(½ ¼)+(½ ¾)} = ¼$

$P(B|D_1C_0) = P(B|C_0)P(D_1|BC_0)/P(D_1|C_0) = ½ ¾ / [(½ ¼)+(½ ¾)} = ¾$

$i$ = 2:

$D_2$ = H, $P(A|C_1)=¼$ , $P(B|C_1)=¾$ // $C_1$ is "after learned from $D_1$"

$P(A|D_2C_1) = P(A|C_1)P(D_2|AC_1)/P(D_2|C_1) = ¼¼ / [(¼¼)+(¾¾)} = 1/10$

$P(B|D_2C_1) = P(B|C_1)P(D_2|BC_1)/P(D_2|C_1) = ¾¾ / [(¼¼)+(¾¾)} = 9/10$

$i$ =3:

$D_3$ = T, $P(A|C_2) = 1/10$ , $P(B|C_2) = 9/10$ // $C_2$ is "after learned from $D_2$"

$P(A|D_3C_2) = ?$

$P(A|D_3C_2) = ?$

$i$ =4:

$D_4$ = H, $P(A|C_3) = ?$, $P(B|C_3) = ?$

……

……

eg 2:

# An Example: Concept Learning

### 4.12.1 Concept Learning

Let us go through a very simple example that illustrates the application of Bayesian probability theory to concept learning. Suppose we are given a learning task $X$ in which the instance space has three instances, $\{1, 2, 3\}$, and we consider the complete hypothesis space, which in this case has eight hypotheses:

| | | | |
|---|---|---|---|
| $C_0 \equiv \emptyset$ | $C_2 \equiv \{2\}$ | $C_4 \equiv \{1,2\}$ | $C_6 \equiv \{2,3\}$ |
| $C_1 \equiv \{1\}$ | $C_3 \equiv \{3\}$ | $C_5 \equiv \{1,3\}$ | $C_7 \equiv \{1,2,3\}$ |

Eight hypotheses

This is a very simple task: 3 instances and $2^3$=8 hypotheses

For details, please read section 4.12 in the ALFE book

// $D_t$ is data at time $t$
// $C_j$ is the concept or theory
// $X_t$ is the background knowledge at time t

# Incremental Bayesian Learning

When the new data D is given to us in a stream (one at a time):

Let the proposition $A_i$ stands for "$C_i$ is the unknown concept," and $D_j$ for "the instance $j$ is in the unknown concept." Note that given $X$, all possible data the learner might see is $D_1, \overline{D_1}, D_2, \overline{D_2}, D_3$, and $\overline{D_3}$. We assume there is a probability of $10^{-6}$ that a training example is in error. Since the learner has no knowledge to favor any particular concept at the offset, we set the initial probabilities $p(A_i|X) = \frac{1}{8}$, where $0 \leq i \leq 7$. From this simple learning task, we wish to see how each training example $D_j$ is used to compute the probabilities $p(A_i|D_jX)$. The concept with the highest probability will then be the best guess for the target concept.

Initially, t=0. and $p(A_i \mid X_{t=0}) = 1/8$, for all *i*

// $D_t$ is data
// $A_i$ or $C_j$ is the concept or theory
// $X_t$ is the background knowledge at time t

# Incremental Learning from Example $D_1$

Now suppose the first training example is $D_1$ "instance 1 is in the unknown concept." By the Bayesian theorem, we can calculate $p(A_i|D_1X)$ as follows:

$$p(A_i|D_1X) = p(A_i|X)\frac{p(D_1|A_iX)}{p(D_1|X)}$$

The terms on the right-hand side are easy to compute. To compute $p(D_1|A_iX)$, notice the fact that $C_0, C_2, C_3$, and $C_6$ do not cover instance 1 unless the training example is incorrect. If $A_0$ were true (i.e., the concept $C_0$ was the target concept), then the probability of seeing $D_1, D_2$, or $D_3$ is $10^{-6}$, and $p(D_1|A_0X) = \frac{1}{3}(10^{-6})$. This reasoning can also be applied to the cases of assuming $A_2$, $A_3$, and $A_6$. So we have

$$p(D_1|A_0X) = p(D_1|A_2X) = p(D_1|A_3X) = p(D_1|A_6X) = \frac{1}{3}(10^{-6})$$

# Incremental Learning from Example $D_1$

If the target concept were one of $C_1, C_4, C_5$, and $C_7$, then $D_1$ is a correct training example. Since the probability of seeing a correct training example in these cases is $(1 - 10^{-6})$ and $D_1$ is one of the three possibly correct training examples, the probability of seeing $D_1$ is $\frac{1}{3}(1 - 10^{-6})$. So we have

$$p(D_1|A_1X) = p(D_1|A_4X) = p(D_1|A_5X) = p(D_1|A_7X) = \frac{1}{3}(1 - 10^{-6})$$

Finally, since given $X$ there are six possible training examples, we have $p(D_1|X) = \frac{1}{6}$. (Another way to compute $p(D_1|X)$ is as follows. Since $D_1 = D_1A_0 + D_1A_1 + \ldots + D_1A_7$, we have $p(D_1|X) = p(D_1A_0|X) + p(D_1A_1|X) + \ldots + p(D_1A_7|X)$. The reader can verify that this is also $\frac{1}{6}$.) Putting them all together, we have

$$\begin{array}{ll} p(A_0|D_1X) = \frac{1}{4}(10^{-6}) & p(A_4|D_1X) = \frac{1}{4}(1 - 10^{-6}) \\ p(A_1|D_1X) = \frac{1}{4}(1 - 10^{-6}) & p(A_5|D_1X) = \frac{1}{4}(1 - 10^{-6}) \\ p(A_2|D_1X) = \frac{1}{4}(10^{-6}) & p(A_6|D_1X) = \frac{1}{4}(10^{-6}) \\ p(A_3|D_1X) = \frac{1}{4}(10^{-6}) & p(A_7|D_1X) = \frac{1}{4}(1 - 10^{-6}) \end{array}$$

Notice that $\sum_{i=0}^{7} p(A_i|D_1X) = 1$.

Now, which $A_i$ is more likely?

Now t=1, and $p(A_i \mid X_{t=1})$ as shown

# Incremental Learning from Example $D_2$

To go further into the learning process, now suppose the second training example is $\overline{D_2}$: "Instance 2 is not in the unknown concept." To process this training example, the Bayesian theorem is used again:

$$p(A_i|\overline{D_2}X') = p(A_i|X')\frac{p(\overline{D_2}|A_iX')}{p(\overline{D_2}|X')}$$

Notice that at this point the background information is not $X$ but $X' = XD_1$, for $D_1$ is now a part of the learner's experience. As before, the terms on the right-hand side are ready to be computed: the values for $p(A_i|X')$ were just computed above. The value of $p(\overline{D_2}|X') = p(\overline{D_2}|X) = \frac{1}{6}$ because the training examples are given to be independently chosen. The values of $p(\overline{D_2}|A_iX')$, which is equal to $p(\overline{D_2}|A_iX)$ because of the independence of $D_1$ and $\overline{D_2}$, can be computed as

$$p(\overline{D_2}|A_0X') = p(\overline{D_2}|A_1X') = p(\overline{D_2}|A_3X') = p(\overline{D_2}|A_5X') = \frac{1}{3}(1 - 10^{-6})$$

because $\overline{D_2}$ is a correct training example for $C_0, C_1, C_3$, and $C_5$, and

$$p(\overline{D_2}|A_2X') = p(\overline{D_2}|A_4X') = p(\overline{D_2}|A_6X') = p(\overline{D_2}|A_7X') = \frac{1}{3}(10^{-6})$$

because $\overline{D_2}$ is an incorrect training example for $C_2, C_4, C_6$, and $C_7$.

# Incremental Learning from Example $D_2$

Putting them all together, we have

$$
\begin{aligned}
p(A_0|\overline{D_2}X') &= \frac{1}{4}10^{-6}\cdot 6\cdot\frac{1}{3}(1-10^{-6}) \\
p(A_1|\overline{D_2}X') &= \frac{1}{4}(1-10^{-6})\cdot 6\cdot\frac{1}{3}(1-10^{-6}) \\
p(A_2|\overline{D_2}X') &= \frac{1}{4}10^{-6}\cdot 6\cdot\frac{1}{3}10^{-6} \\
p(A_3|\overline{D_2}X') &= \frac{1}{4}10^{-6}\cdot 6\cdot\frac{1}{3}(1-10^{-6}) \\
p(A_4|\overline{D_2}X') &= \frac{1}{4}(1-10^{-6})\cdot 6\cdot\frac{1}{3}10^{-6} \\
p(A_5|\overline{D_2}X') &= \frac{1}{4}(1-10^{-6})\cdot 6\cdot\frac{1}{3}(1-10^{-6}) \\
p(A_6|\overline{D_2}X') &= \frac{1}{4}10^{-6}\cdot 6\cdot\frac{1}{3}10^{-6} \\
p(A_7|\overline{D_2}X') &= \frac{1}{4}(1-10^{-6})\cdot 6\cdot\frac{1}{3}10^{-6}
\end{aligned}
$$

Now, which *Ai* is more likely?

Notice again that $\sum_{i=0}^{7} p(A_i|\overline{D_2}X') = 1$.

Now t=2, and $p(A_i \mid X_{t=2})$ as shown

# Incremental Bayesian Learning

// $D_t$ is data
// $C_j$ is the concept or theory
// $X_t$ is the background knowledge at time t

at time 0:
    let the concepts to be learned be: C1, C2, …, Ci, …, Cn
    initialize P(Ci | X1) to be equal to 1/n

at time 1:
    get the first example D1
    Use Bayesian rule to compute P(Ci | D1, X1) = P(Ci | X1) P(D1 | Ci, X1) / P(D1 | X1)

at the time 2:
    set the value P(Ci | X2) to be equal to P(Ci | D1, X1)
    get the second example D2
    Use Bayesian rule to compute P(Ci | D2, X2) = P(Ci | X2) P(D2 | Ci, X2) / P(D2 | X2)

……
at the time t:
    set the value P(Ci | X_t) to be equal to P(Ci | D_t-1, X_t-1)
    get the t_th example D_t
    Use Bayesian rule to compute P(Ci | D_t, X_t) = P(Ci | X_t) P(D_t | Ci, X_t) / P(D_t | X_t)

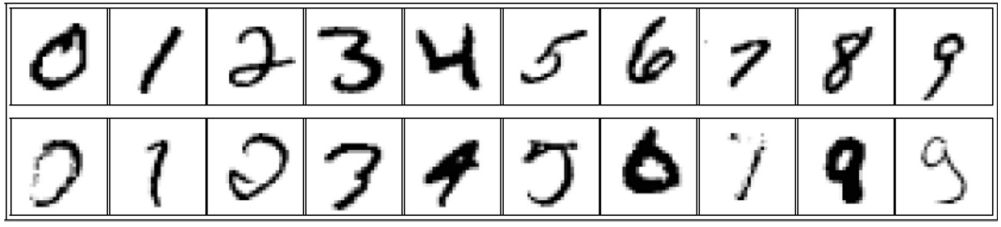
at the time t+1:
    set the value P(Ci | X_t+1) to be equal to P(Ci | D_t, X_t)
    get the t+1_th example D_t+1
    Use Bayesian rule to compute P(Ci | D_t+1, X_t+1) = P(Ci | X_t+1) P(D_t+1 | Ci, X_t+1) / P(D_t+1 | X_t+1)

continue

# 2. Batch Learning: When all data are given upfront

- Let the hypotheses be $C_1$, $C_2$, …, $C_m$
  - To start, all $C_i$ are equally likely, $P(C_i) = 1/m$
- Given all training data $\mathbf{D}=\{D_1, D_2, \ldots, D_n\}$
  - Where, each $D_i$ is an example
- Build a Bayesian classifier from **D**
  - $P(C_i|\mathbf{D}) = P(C_i)\, P(\mathbf{D}|C_i) / P(\mathbf{D})$
- Use this Bayesian classifier to evaluate any new data in the future (no more learning)

# Example: Recognize Written Digits



Each instance *Di* above has *k* pixels, each pixel has value *d*={0,1}.
The class *Cj* is the "Digit *j*," from "0" through "9"

$$P(C_j \mid D) = P(C_j)\frac{P(D \mid C_j)}{P(D)}$$

// $D_t$ is data
// $C_j$ is the concept or hypothesis

We learn the terms on the right-hand-side of the Bayesian Rule, then
We predict $C_j$ for any *new image* $D_x$ where $P(C_j|D_x)$ is the highest

# Bayesian Learning from a Batch

Formally, let the symbol $X$ denote the fact that we are given a $K$-dimensional data space $\mathcal{D}$, where each point $D_i$ in this space is an object with $K$ attributes: $D_i = (d_{i1}, \ldots, d_{ik}, \ldots, d_{iK})$ (the value of an attribute may be continuous or discrete). Let $D \equiv D_1, D_2, \ldots, D_I$, be a set of $I$ data points that have been observed. Our task is to divide the space $\mathcal{D}$ into a set of regions that best reflect the information contained in $D$. Let $H_J$ be the proposition that "the data space $\mathcal{D}$ is covered by $J$ classes: $\mathcal{D} = C_1 + \cdots + C_j + \cdots + C_J$, where each $C_j$ is defined as a set of $K$ probability distribution functions (pdfs): $f_{j1}, \ldots, f_{jK}$, one for each attribute." Then our task can be formalized as finding a $C_j$ such that the probability of $C_j$ given $D$ and $X$ is the highest.

$$P(C_j \mid D) = P(C_j)\frac{P(D \mid C_j)}{P(D)}$$

// $D_t$ is data
// $C_j$ is the concept or hypothesis

So our task is simply computing the terms on the right-hand side of this equation for every $C_j$ based on the given data **D**

$P(C_j)$ = (# of $C_j$ in **D**) / J  // at time t=0
$P(\mathbf{D} \mid C_j)$ = Product of all $P(d_{ik} \mid f_{ik})$ for all *i* and *k*
$P(\mathbf{D})$ = Sum of all $P(\mathbf{D}C_j)$ = Sum of all $P(C_j)P(\mathbf{D} \mid C_j)$ for *j*

# The Classification Problem

An example of Classification:

Given: price $X_1$ and power $X_2$

Learn: What is a "family car" C

// $X_t$ is data
// $C_j$ is concept/hypothesis

Given Data **X** = [$X_1$, $X_2$, C] where C=1 if it is a family car, C=0 otherwise.

So if we do know P(C | $X_1$, $X_2$) when a new car arrives, then we can predict:

$$choose \begin{cases} C = 1 \text{ if } P(C = 1|x_1, x_2) > 0.5 \\ C = 0 \text{ otherwise} \end{cases}$$

or: $$choose \begin{cases} C = 1 \text{ if } P(C = 1|x_1, x_2) > P(C = 0|x_1, x_2) \\ C = 0 \text{ otherwise} \end{cases}$$

---

or: $$choose \begin{cases} C = 1 \text{ if } P(C = 1|x_1, x_2) > P(C = 0|x_1, x_2) \\ C = 0 \text{ otherwise} \end{cases}$$

- In this example the error is: $1 - max(P(C = 1|x_1, x_2), P(C = 0|x_1, x_2))$
- The problem, however, in most cases is to actually calculate: P(C | **x**)

# A Classification Example

- Credit scoring: Inputs are income and savings.
  Output is low-risk vs high-risk
- Input: $\boldsymbol{x} = [x_1, x_2]^T$ ,Output: C = {0,1}
- Prediction:

// $X_t$ is data
// $C_j$ is concept/hypothesis

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C = 1 \mid x_1, x_2) > 0.5 \\ C = 0 \text{ otherwise} \end{cases}$$

or

$$\text{choose} \begin{cases} C = 1 \text{ if } P(C = 1 \mid x_1, x_2) > P(C = 0 \mid x_1, x_2) \\ C = 0 \text{ otherwise} \end{cases}$$

# Bayesian Rule for *Multiple* Classes

$$P(C_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x} \mid C_i)P(C_i)}{\sum_{k=1}^{K} p(\mathbf{x} \mid C_k)P(C_k)}$$

// $x_t$ is data
// $C_j$ is concept/hypothesis

$$P(C_i) \geq 0 \text{ and } \sum_{i=1}^{K} P(C_i) = 1$$

$$\text{choose } C_i \text{ if } P(C_i \mid \mathbf{x}) = \max_k P(C_k \mid \mathbf{x})$$

# The Classification Definition

- In the general case we have K mutually exclusive classes $C_i$, i = 1, …, K
- Prior probabilities satisfy:

$$P(C_i) \geq 0 \text{ and } \sum_{i=1}^{K} P(C_i) = 1$$

- Posterior probability of class $C_i$ is calculated as: when data X is given

$$P(C_i|\mathbf{x}) = \frac{P(C_i)p(\mathbf{x}|C_i)}{p(\mathbf{x})} = \frac{P(C_i)p(\mathbf{x}|C_i)}{\sum_{k=1}^{K} P(C_k)p(\mathbf{x}|C_k)}$$

- Bayes' classifier:

$$\text{choose } C_i \text{ if } P(C_i|\mathbf{x}) = max_k P(C_k|\mathbf{x})$$

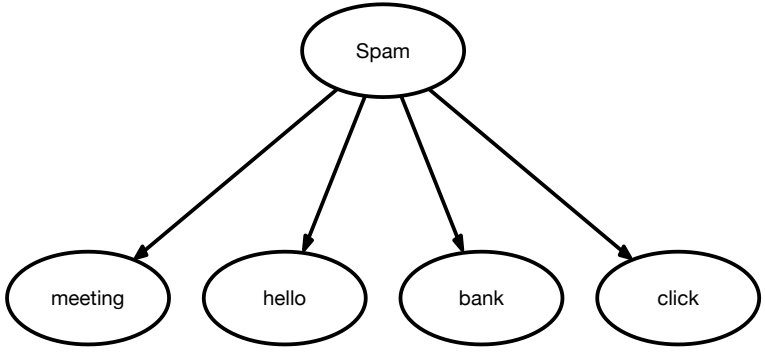
// $x_t$ is data
// $C_j$ is concept/hypothesis

# Example: Text Classification

- Learn a function (from training data) that maps input into (discrete) classes
- e.g.: Spam Detection
  - Classes: Spam vs. Not-Spam (binary classification)
  - Features: "Bag-of-words" represented as a vector (all the words in the email message, *without accounting for oder*)
  - Training Data: collection of email messages marked as Spam or not-Spam
- Where does the training data come from?
  - Expert annotation, crowd-sourcing, found data, records from the past, user data, etc.

# Naive Bayes for Text Classification

- Naive Bayes classification for text categorization
  - Very common baseline (can do surprisingly well)
  - Classify news stories, ..
  - Classify spam vs. not-spam
  - Bag-of-words features
  - P(Spam | words)



# Naive Bayes for Text Classification

Suppose I want to know if a news article is about sports, politics, entertainment

Classes: sports, politics, entertainment

Probability that a document *d* belongs to class *c:* $P(d|c)$

Probability of class *c* given document *d:* $P(c \mid d)$

// $d_t$ is data
// $c_j$ is the concept or hypothesis

Compute for every class, and choose the max to predict which $c^*$

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

$$\begin{aligned} c^* &= \arg\max_c P(c|d) \\ &= \arg\max_c \frac{P(c)P(d|c)}{P(d)} \\ &= \arg\max_c P(c)P(d|c) \end{aligned}$$

# Naive Bayes for Text Classification

How do we estimate P(c)?

How do we estimate P(d | c)?

Naive Bayes assumption: words are independent

$$P(d|c) = P(w_1|c)P(w_2|c)\ldots P(w_L|c)$$

Please see the discussion slides for some detailed examples.

# Spam Filtering

| | |
|---|---|
| SPAM | click for pharmacy |
| OK | free time today |
| SPAM | online pharmacy link |
| OK | no free time |
| OK | free good pharmacy |
| SPAM | pharmacy free link |
| OK | for time today |
| OK | time is money |

Vocabulary size: 12

| | |
|---|---|
| click | online |
| for | link |
| pharmacy | no |
| free | good |
| time | is |
| today | money |

$P(spam) =$ Maximum likelihood estimate

Given the data, we have $P(spam) = \frac{3}{8} \quad P(\neg spam) = \frac{5}{8}$

# Spam Filtering with Naive Bayes

| | |
|---|---|
| SPAM | click for pharmacy |
| OK | free time today |
| SPAM | online pharmacy link |
| OK | no free time |
| OK | free good pharmacy |
| SPAM | pharmacy free link |
| OK | for time today |
| OK | time is money |

Msg = “Pharmacy for pharmacy”

$$P(spam) = \frac{3}{8} \qquad P(\neg spam) = \frac{5}{8}$$

$$P(\text{"pharmacy"}|spam) = 1/3$$
$$P(\text{"pharmacy"}|\neg spam) = 1/15$$
$$P(\text{“for”}|\ spam) = 1/9$$
$$P(\text{”for"}|\neg spam) = 1/15$$

$$P(spam|Msg) = \frac{P(spam)P(Msg|spam)}{P(spam)P(Msg|spam) + P(\neg spam)P(Msg|\neg spam)}$$

$$P(Msg|spam) = P(w_1|spam)P(w_2|spam)P(w_3|spam)$$

# Spam Filtering with Naive Bayes

$$P(spam|Msg) = \frac{3/8 \cdot 1/3 \cdot 1/9 \cdot 1/3}{P(spam)P(Msg|spam) + P(\neg spam)P(Msg|\neg spam)}$$

$$P(spam|Msg) = \frac{1/216}{1/216 + P(\neg spam)P(Msg|\neg spam)}$$

$$P(spam|Msg) = \frac{1/216}{1/216 + 5/8 \cdot 1/15 \cdot 1/15 \cdot 1/15} \quad = 0.96$$

# Spam Filtering with Naive Bayes

| | |
|---|---|
| SPAM | click for pharmacy |
| OK | free time today |
| SPAM | online pharmacy link |
| OK | no free time |
| OK | free good pharmacy |
| SPAM | pharmacy free link |
| OK | for time today |
| OK | time is money |

$$P(spam) = \frac{3}{8} \qquad P(\neg spam) = \frac{5}{8}$$

$$P(\text{“}time\text{”} \mid spam) = 0$$
$$P(\text{“}time\text{”} \mid \neg spam) = 4/15$$

Msg2 = “Time for pharmacy”

$P$ (spam | Msg2) = 0.0
$P$ (¬spam | Msg2) > 0.0

 Is this classification good?

# Unsupervised Learning (Clustering)

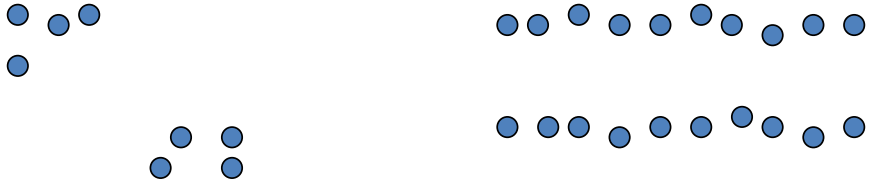
- Type I: Clustering the data
  - Automatically group the data into clusters
  - Today's lecture
- Type II: Parameter Learning
  - States are known
  - Learn transitions, sensor models, & current state
  - Today's lecture
- Type III: Structural Learning (beyond the scope of CS561)
  - States are not known (only observations and actions are known)
  - Surprise-Based Learning (see www.isi.edu/robots for references)
    - POMDP where states are NOT known and must be learned from experiences
    - States are not just "symbols" but may have internal structures

# Clustering

- Clustering systems:
  - Unsupervised learning
  - Detect patterns in unlabeled data
    - E.g. group emails or search results
    - E.g. find categories of customers
    - E.g. detect anomalous executions
  - Useful when don't know what you're looking for
  - Requires data, but no labels
  - Often get gibberish but may have surprisingly good results

# Clustering

- Basic idea: group together similar instances
- Example: 2D point patterns



- What could “similar” mean?
  - One option: small (squared) Euclidean distance

$$\text{dist}(x, y) = (x - y)^{\top}(x - y) = \sum_{i}(x_i - y_i)^2$$

# An Example of "Clustering"


- Given the four data points D above
  - Cluster them in one, two, three, or four clusters?
  - Let the hypothesis be $H_1$, $H_2$, $H_3$, $H_4$
  - Choose the $H_J$ such that $P(H_J|DX)$ is the highest
    - Where X is the background knowledge

Bayes Model

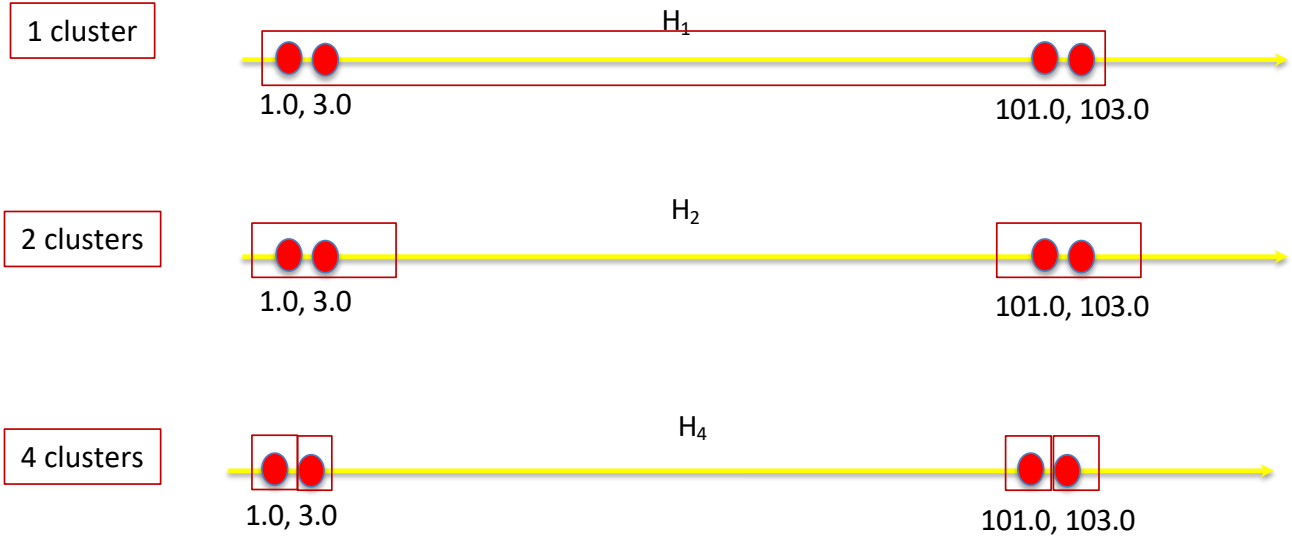
$$P(H_J|DX) = P(H_J|X)\frac{P(D|H_JX)}{P(D|X)}$$

How well hypothesis explain data?

How likely is the data?

How likely is the hypothesis?

# Different Hypotheses for Clustering (Assume Gaussian Distributions)



# 2. A Naïve Bayes Model

Formally, let the symbol $X$ denote the fact that we are given a $K$-dimensional data space $\mathcal{D}$, where each point $D_i$ in this space is an object with $K$ attributes: $D_i = (d_{i1}, \ldots, d_{ik}, \ldots, d_{iK})$ (the value of an attribute may be continuous or discrete). Let $D \equiv D_1, D_2, \ldots, D_I$, be a set of $I$ data points that have been observed. Our task is to divide the space $\mathcal{D}$ into a set of regions that best reflect the information contained in $D$. Let $H_J$ be the proposition that "the data space $\mathcal{D}$ is covered by $J$ classes: $\mathcal{D} = C_1 + \cdots + C_j + \cdots + C_J$, where each $C_j$ is defined as a set of $K$ probability distribution functions (pdfs): $f_{j1}, \ldots, f_{jk}, \ldots, f_{jK}$, one for each attribute." Then our task can be formalized as finding an $H_J$ such that the probability of $H_J$ given $D$ and $X$ is the highest.

Using the Bayesian theorem, the probability of $H_J$ given $D$ and $X$ is

$$P(H_J|DX) = P(H_J|X)\frac{P(D|H_JX)}{P(D|X)} \qquad (4.5)$$

So our task is simply computing the terms on the right-hand side of this equation for every possible $H_J$.

// $D_t$ is data
// $H_j$ is the concept or theory
// $X_t$ is the background knowledge at time t

Example 1: data D = freshly caught fishes, cluster J = types of fishes
Example 2: data D = star observations, cluster J = type of stars

# A Naïve Bayes Model

Let us first consider $P(D|H_J X)$. Assuming that the data points $D_i$ are independent, we have

Naïve

$$P(D|H_J X) = \prod_{i=1}^{I} P(D_i|H_J X) \tag{4.6}$$

Furthermore, since each data point $D_i$ must belong to some class, we have

$$D_i = D_i(C_1 + \cdots + C_j + \cdots + C_J)$$

Background knowledge

and, by the sum and product axioms,

$$P(D_i|H_J X) = \sum_{j=1}^{J} P(D_i C_j|H_J X) = \sum_{j=1}^{J} P(C_j|H_J X) P(D_i|C_j H_J X) \tag{4.7}$$

By the definition of classes, we also have

$$P(D_i|C_j H_J X) = P(d_{i1}, \ldots, d_{ik}, \ldots, d_{iK}|f_{j1}, \ldots, f_{jk}, \ldots, f_{jK}, H_J, X) \tag{4.8}$$

If attributes can be assumed independent, then we have

Naïve

$$P(D_i|C_j H_J X) = \prod_{k=1}^{K} P(d_{ik}|f_{jk}, H_J, X) \tag{4.9}$$

The Autoclass Clustering Algorithm

# AUTOCLASS Algorithm

For an attribute $k$ that is continuous, we assume that the corresponding pdf $f_{jk}$ is a Gaussian function $\mathcal{N}(\mu_{jk}, \sigma_{jk}^2)$, where $\mu_{jk}$ and $\sigma_{jk}^2$ are the mean and variance of $f_{jk}$, respectively. In this case, the corresponding term in Equation 4.9 is computed as:

$$P(d_{ik}|f_{jk}, H_J, X) = \mathcal{N}(d_{ik}, \sigma_{jk}^2) \cdot \Delta d_{ik}$$

where $\Delta$ is the error in measuring $d_{ik}$.

For an attribute $k$ that has a set of discrete values $v_1, \ldots, v_m$, the corresponding pdf $f_{jk}$ is assumed to be a set of probabilities $p_1, \ldots, p_m$, where $p_m$ is the probability of the attribute having the value $v_m$. In this case, the corresponding term in Equation 4.9 is computed as:

$$P(d_{ik}|f_{jk}, H_J, X) = d_{ik} p_i$$

# AUTOCLASS Algorithm

To compute the term $P(H_J|X)$ in Equation 4.5, we assume that we have no prior knowledge about the number of classes, and so $J$ can be any index from 1 to $I$:

$$P(H_J|X) = \frac{1}{I}$$

To compute the denominator of Equation 4.5, $P(D|X)$, we notice that the number of classes cannot be more than the number of data points $I$, and $D = D(H_1 + \cdots + H_i + \cdots + H_I)$, thus,

$$P(D|X) = \sum_{i=1}^{I} P(DH_i|X) = \sum_{i=1}^{I} P(H_i|X)P(D|H_iX)$$

where both $P(H_i|X)$ and $P(D|H_iX)$ on the right-hand side are as computed above.

Putting all this information together, we can now compute the probability $P(H_J|DX)$ as follows, which is derived from the right-hand side of Equation 4.5:

$$\frac{P(H_J|X) \cdot \prod_{i=1}^{I} \sum_{j=1}^{J} P(C_j|H_JX) \prod_{k=1}^{K} P(d_{ik}|f_{jk}, H_J, X)}{\sum_{h=1}^{I} P(H_h|X) \cdot \prod_{i=1}^{I} \sum_{j=1}^{h} P(C_j|H_hX) \prod_{k=1}^{K} P(d_{ik}|f_{jk}, H_h, X)} \quad (4.10)$$

# Back to the example

1 cluster

$$P(D|H_1X) = (\frac{1}{\sqrt{2\pi}})^4[e^{\frac{-(1-52)^2}{2}}][e^{\frac{-(3-52)^2}{2}}][e^{\frac{-(101-52)^2}{2}}][e^{\frac{-(103-52)^2}{2}}] = \frac{e^{-[51^2+49^2]}}{(2\pi)^2}$$

2 clusters

When $J = 2$, $\mu_1 = 2$, and $\mu_2 = 102$, we have

$$P(D|H_2X) = (\frac{1}{\sqrt{2\pi}})^4[e^{-\frac{(1-2)^2}{2}} + e^{\frac{(1-102)^2}{2}}][e^{-\frac{(3-2)^2}{2}} + e^{\frac{(3-102)^2}{2}}]$$
$$[e^{-\frac{(101-2)^2}{2}} + e^{\frac{(101-102)^2}{2}}][e^{-\frac{(103-2)^2}{2}} + e^{\frac{(103-102)^2}{2}}] = \frac{e^{-2}}{(8\pi)^2}$$

$P(D|H_3X)$ = .....

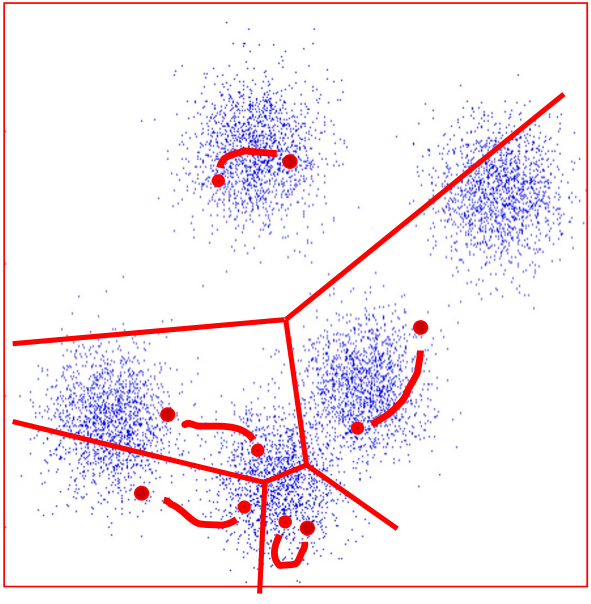
4 clusters

$$P(D|H_4X) = \frac{(1+e^{-2})^4}{(32\pi)^2}$$

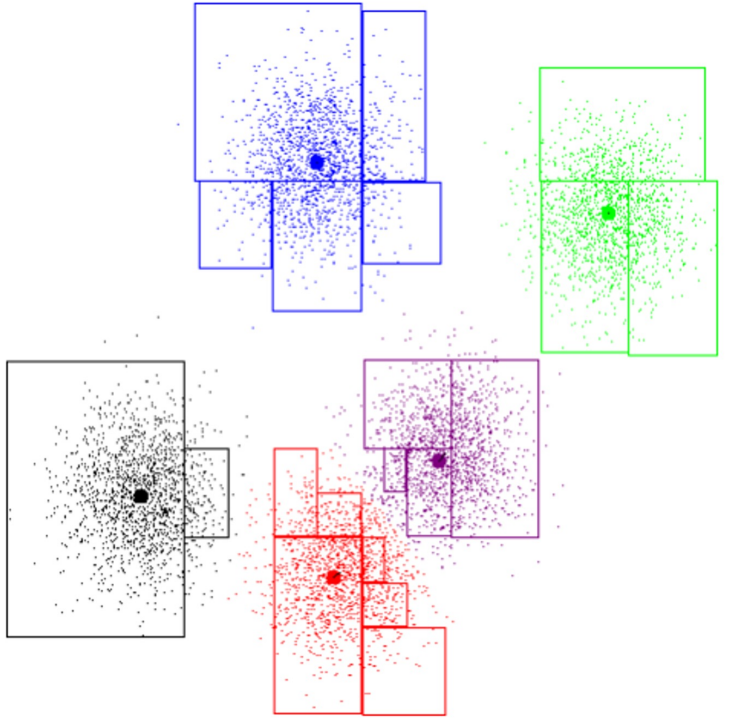
Since $P(X|H_2X) > P(X|H_3X) > P(X|H_4X) > P(X|H_1X)$, the Bayesian theorem tells that $H_2$ is the best hypothesis, which matches to our intuition!

# K-Means Clustering

- An iterative clustering algorithm
  - Pick K random points as cluster centers (means)
  - Alternate:
    - Assign data instances to closest mean
    - Assign each mean to the average of its assigned points
  - Stop when no points' assignments change



# K-Means Example



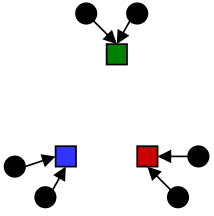
# K-Means as Optimization

- Consider the total distance to the means:

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \text{dist}(x_i, c_{a_i})$$

points assignments means

- Each iteration reduces ϕ



- Two stages each iteration:
  - Update assignments: fix means c, change assignments a
  - Update means: fix assignments a, change means c

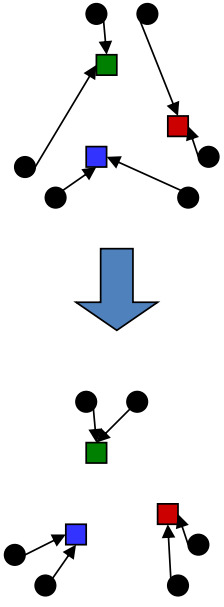
# Phase I: Update Assignments

- For each point, reassign to the closest mean:

$$a_i = \underset{k}{\operatorname{argmin}}\, \mathsf{dist}(x_i, c_k)$$

- Can only decrease total distance ϕ !

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \mathsf{dist}(x_i, c_{a_i})$$

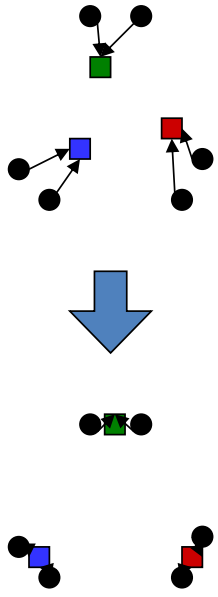


# Phase II: Update Means

- Move each mean to the average of its assigned points:

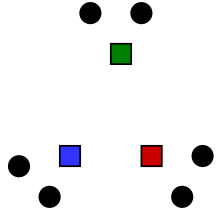
$$c_k = \frac{1}{|\{i : a_i = k\}|} \sum_{i:a_i=k} x_i$$

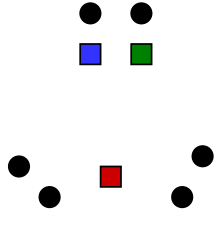
- Also can only decrease total distance…



# Initialization

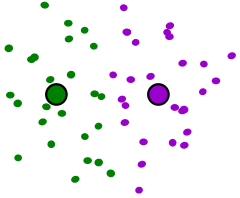
- K-means is non-deterministic
  - Requires initial means
  - It does matter what you pick!
  - What can go wrong?
  - Various schemes for preventing this kind of thing: variance-based split / merge, initialization heuristics

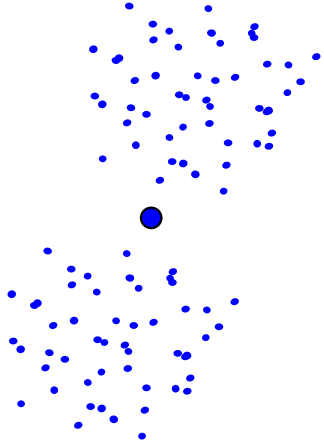




# K-Means Getting Stuck

- A local optimum:





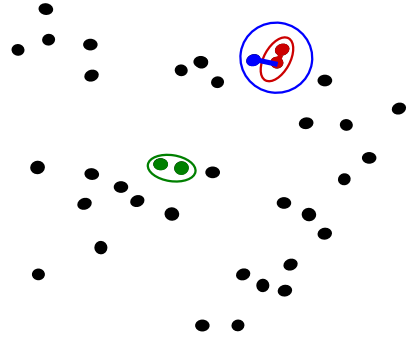
*Why doesn't this work out like the earlier example, with the purple taking over half the blue?*

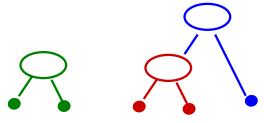
# K-Means Questions

- Will K-means converge?
  - To a global optimum?
- Will it always find the true patterns in the data?
  - If the patterns are very very clear?
- Will it find something interesting?
- Do people ever use it?
- How many clusters to pick?

# Agglomerative Clustering

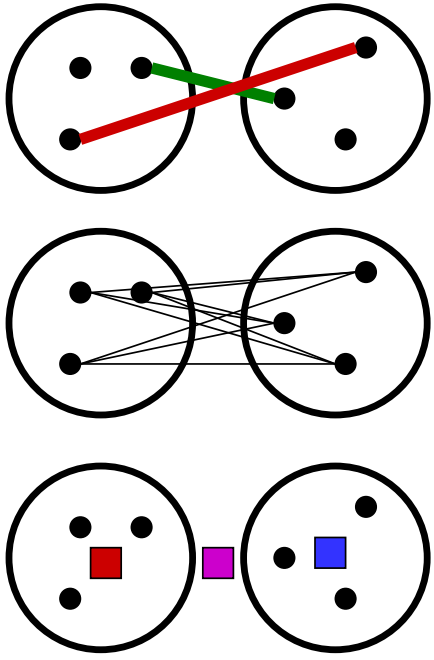
- Agglomerative clustering:
  - First merge very similar instances
  - Incrementally build larger clusters out of smaller clusters
- Algorithm:
  - Maintain a set of clusters
  - Initially, each instance in its own cluster
  - Repeat:
    - Pick the two closest clusters
    - Merge them into a new cluster
    - Stop when there's only one cluster left
- Produces not one clustering, but a family of clusters represented by a dendrogram



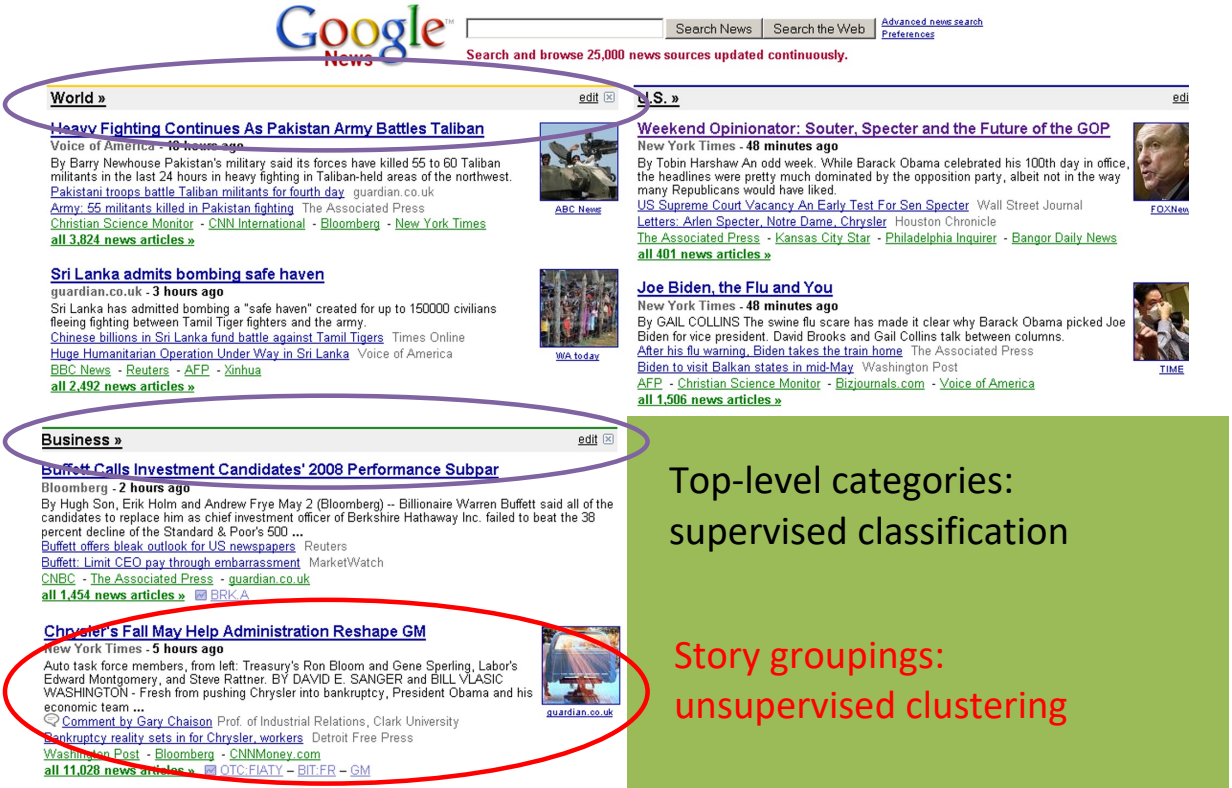


# Agglomerative Clustering

- How should we define “closest” for clusters with multiple elements?
- Many options
  - Closest pair (single-link clustering)
  - Farthest pair (complete-link clustering)
  - Average of all pairs
  - Ward’s method (min variance, like k-means)
- Different choices create different clustering behaviors



# Clustering Application




Top-level categories:
supervised classification

Story groupings:
unsupervised clustering

# Clustering Algorithms (Review)

- Naïve Bayes Model (AUTOCLASS)
  - Select $M_i$ that has the best $P(D|M_i)$,
- K-Means
  - Loop until no improvement
    - Assign data to the nearest cluster
    - Adjust clusters to fit the assignments
- Agglomerative
  - Always merge the pair of “closest” clusters

# The EM Algorithm

- It is not an “algorithm”, it is a **Framework**!
- A loop of two phases
  - Estimation (using Maximum likelihood)
  - Modification (using Expectation maximization)
- For example, when we do clustering
  - Phase 1: update assignment (data to cluster)
  - Phase 2: update means (adjust clusters)

# The EM Algorithm

- A very general framework
- Many forms & applications
  - Clustering with Gaussians
  - Bayesian net with hidden variables
  - Hidden Markov models (HMM)
  - Partially Observable Markov Decision Process (POMDP)
    - See e.g., ALFE 5.10
  - Others
- We describe it by form/applications
  - Clustering
  - Learning POMDP 
    when states are known

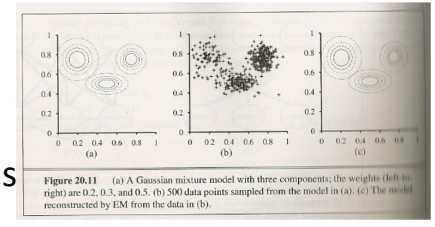


Figure 20.11 (a) A Gaussian mixture model with three components; the weights (left to right) are 0.2, 0.3, and 0.5. (b) 500 data points sampled from the model in (a). (c) The model reconstructed by EM from the data in (b).

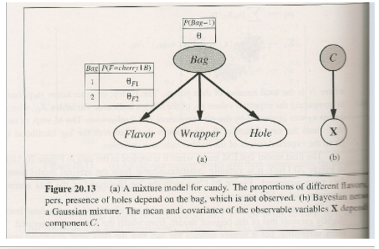



Figure 20.13 (a) A mixture model for candy. The proportions of different flavors, presence of holes depend on the bag, which is not observed. (b) Bayesian network a Gaussian mixture. The mean and covariance of the observable variables X depend component C.

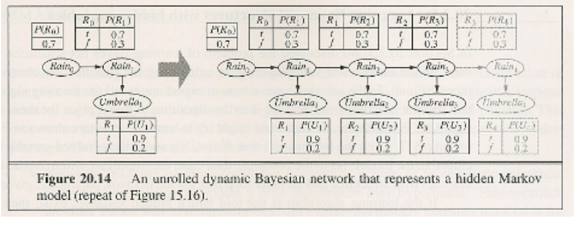


Figure 20.14 An unrolled dynamic Bayesian network that represents a hidden Markov model (repeat of Figure 15.16).

# The Big Ideas

- Training (aka, "parameter estimation" or "maximum likelihood estimation")
  - Given data $X=\{x_1, x_2, \ldots, x_N\}$,
  - Find a model with *parameters* $\vartheta$ that $p(X|\vartheta)$ is maximum
    - Define $p(X|\vartheta)$, and then solve $\partial P(X|\theta)/\partial\theta = 0$ to obtain $\vartheta$
- Testing (aka, "classification", "prediction", "expectation maximization")
  - Given a test data $x$,
  - Predict a class $C_i$ where $P(x|C_i) = P(x|\vartheta_i)$ is maximum
- Note
  - A data point $x$ can have a single feature or *multivariate*

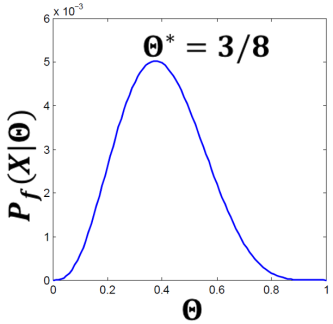
# 2. Maximum Likelihood Estimation

- **Maximum Likelihood Principle (MLE):**
  - **Given:** Data $\boldsymbol{X}$
  - **Assumption:** Data is generated by some model $\boldsymbol{f(\Theta)}$
    - $\boldsymbol{f}$ **… model**
    - $\boldsymbol{\Theta}$ **… model parameters**
  - Want to estimate $\boldsymbol{P_f(X|\Theta)}$: how well does the model support data
    - The probability that our model $\boldsymbol{f}$ (with parameters $\boldsymbol{\Theta}$) generated the data
  - **Now let's find the most likely model (including the parameters) that could have generated the data:** $\arg\max_{\Theta} \boldsymbol{P_f(X|\Theta)}$

eg1:

# Example: MLE (1 coin)

- **Imagine we are given a set of coin flips**
- **Task:** Figure out the bias of a coin!
  - **Data:** Sequence of coin flips: $\boldsymbol{X = [1, 0, 0, 0, 1, 0, 0, 1]}$
  - **Model:** $\boldsymbol{f(\Theta)}$ = return 1 with prob. $\Theta$, else return 0
  - **What is $\boldsymbol{P_f(X|\Theta)}$?** Assuming coin flips are independent
    - **So,** $\boldsymbol{P_f(X|\Theta) = P_f(1|\Theta) * P_f(0|\Theta) * P_f(0|\Theta) \ldots * P_f(1|\Theta)}$
      - What is $\boldsymbol{P_f(1|\Theta)}$? **Simple,** $\boldsymbol{P_f(1|\Theta) = \Theta}$
    - **Then,** $\boldsymbol{P_f(X|\Theta) = \Theta^3(1-\Theta)^5}$
    - **For example:**
      - $\boldsymbol{P_f(X|\Theta = 0.5) = 0.003906}$
      - $\boldsymbol{P_f\left(X\middle|\Theta = \frac{3}{8}\right) = 0.005029}$
  - **What did we learn?** Our data was most likely generated by coin with bias $\boldsymbol{\Theta = 3/8}$



# Example: MLE

- Find the best $\vartheta$ for $P(X \mid \vartheta) = \vartheta^3 (1-\vartheta)^5$
  - Set its partial derivative to 0: $\partial P(X \mid \theta) / \partial \theta = 0$
    - $3\vartheta^2(1-\vartheta)^5 - 5\vartheta^3(1-\vartheta)^4 = 0$
  - Solve this equation
    - $3\vartheta^2(1-\vartheta)^5 = 5\vartheta^3(1-\vartheta)^4$
    - $3(1-\vartheta) = 5\vartheta$
    - $3 - 3\vartheta = 5\vartheta$
    - $\vartheta = 3/8$

# MLE Example (2 coins)

$$\hat{\theta}_A = \frac{\text{\# of heads using coin A}}{\text{total \# of flips using coin A}}$$

and

$$\hat{\theta}_B = \frac{\text{\# of heads using coin B}}{\text{total \# of flips using coin B}}$$

**a** Maximum likelihood


| Coin A | Coin B |
|---|---|
| | 5 H, 5 T |
| 9 H, 1 T | |
| 8 H, 2 T | |
| | 4 H, 6 T |
| 7 H, 3 T | |
| 24 H, 6 T | 9 H, 11 T |

$$\hat{\theta}_A = \frac{24}{24+6} = 0.80$$

$$\hat{\theta}_B = \frac{9}{9+11} = 0.45$$

# Maximum Likelihood Estimation

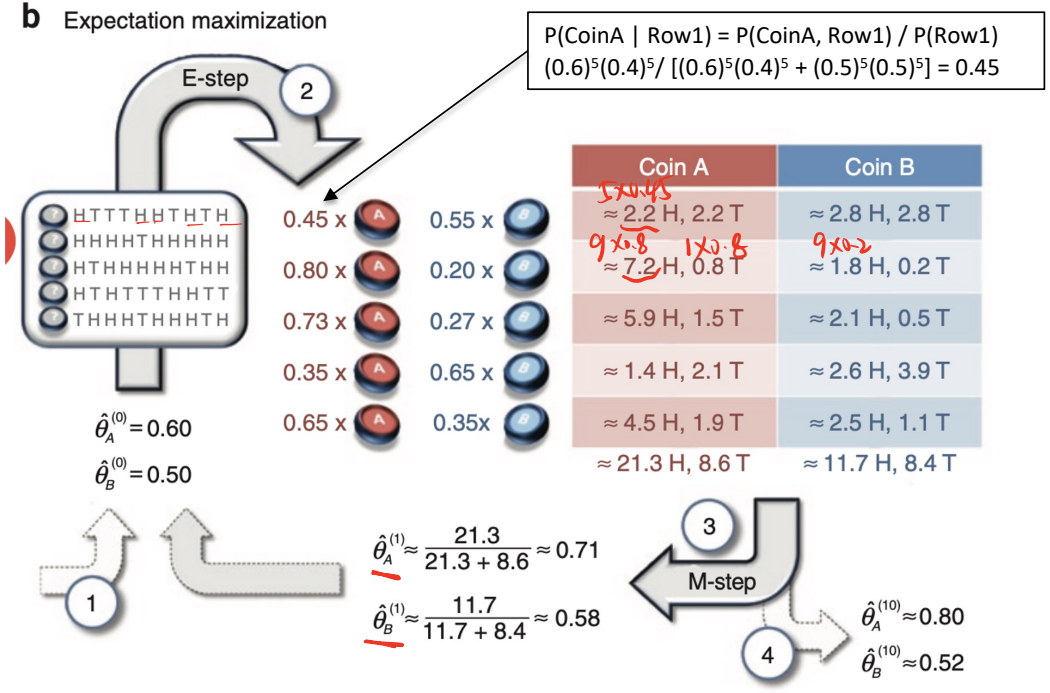
- Given $X = \{x^t\}$, a sample of data
  - X is independent and identically distributed
  - X is drawn from a known density P($x$|θ)
- Compute P(θ | X)
  - Using Bayesian Rule P(θ|X)=P(θ)P(X|θ)/P(X) ~ P(X|θ)
  - Using the independence of X

$$l(\theta|X) = p(X|\theta) = \prod_{t=1}^{N} p(x^t|\theta)$$

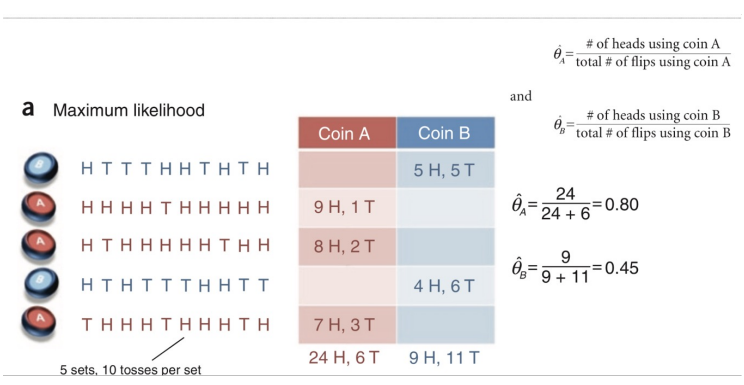
$$\mathcal{L}(\theta|X) = \log l(\theta|X) = \sum_{t=1}^{N} \log p(x^t|\theta)$$

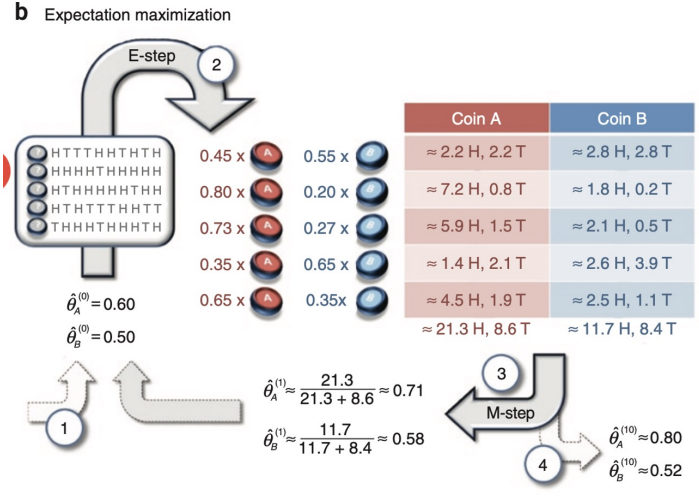
  - Choose a model θ that makes X most likely

# Expectation Maximization



# EM Example (complete)



# Another EM Example

- Given an experience to an agent in an environment
  - E.g., the little prince's experience on his planet
- How does the agent build a better model for its environment?
  - $\mathbf{P}(M_{t=T} | \mathbf{E}_{1:T})$
  - Baum-Welch Learning Procedure
    - For learning HMM or POMDP when the states are known

# HMM/POMDP (A Review)

$$M = (B, Z, S, P, \theta, \pi)$$

hidden Markov model $M$ is a stochastic process $(B, Z, S, P, \theta, \pi)$ with the components defined as follows:

- $B$ is a set of basic actions. As before, the actions can be applied to both the environment and the model.
- $Z$ is a set of percepts and represents the output symbols of the environment that can be observed by the learner.
- $S$ is a finite set of (internal) model states. We assume that at any single instant $t$, the current environmental state $q_t$ corresponds to exactly one model state $s_t$, and the identity of $s_t$ is sufficient to stochastically determine the effects of action in the environment. This is the *Markov assumption*.
- $P = \{P_{ij}[b]\}$ is a set of probabilities concerning model state transitions. For each basic action $b \in B$ and a pair of model states $s_i$ and $s_j \in S$, the quantity $P_{ij}[b]$ specifies the probability that executing action $b$ when the current model state is $s_i$ will move the environment to an environmental state that corresponds to the model state $s_j$.
- $\theta = \{\theta_i(k)\}$, where $\theta_i(k) = p(z_k|s_i)$, $z_k \in Z$, and $s_i \in S$, is a set of probability distributions of observation symbols in each model state. (This corresponds to the appearance function for the deterministic models defined at the beginning of this chapter.) For each observation symbol $z_k$, the quantity $\theta_i(k)$ specifies the probability of observing $z_k$ if the current model state is $s_i$.
- $\pi(t) = \{\pi_i(t)\}$ is the probability distribution of the current model state at time $t$. That is, $\pi_i(t) = p(i_t = s_i)$, where $i_t$ denotes the current model state and $s_i \in S$ specifies the probability of $s_i$ being the current model state at time $t$.

1. Actions
2. Percepts (observations)
3. States
4. Appearance: states ⍰ observations
5. Transitions: (states, actions) ⍰ states
6. Current State

Three key components:
- Sensor model θ=p(z|s)
- Action model P(s|s,a)
- Current state $\pi_t$(s) (localization)

# Little Prince Example

$A \equiv$ {forward, backward, turn-around}

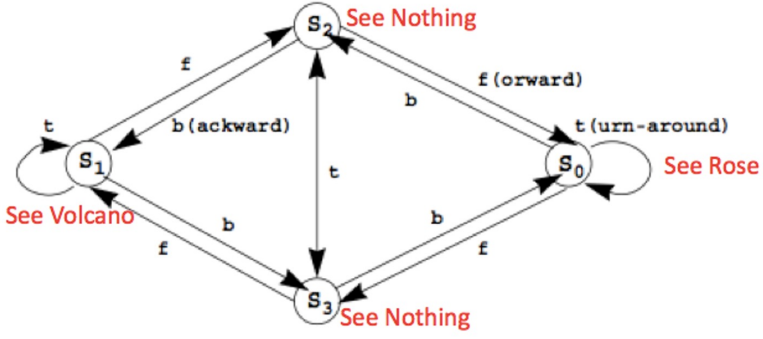
$Z \equiv$ {rose, volcano}

$S \equiv \{s_1, s_2, s_3, s_4\}$

$\phi \equiv$ {P(s3|s0,f)=.51, P(s2|s1,b)=.32, P(s4|s3,t)=.89, …}

$\theta \equiv$ {P(rose|s0)=.76, P(volcano|s1)=.83, P(nothing|s3)=.42, …}

$$\pi_1(0) = 0.25, \pi_2(0) = 0.25, \pi_3(0) = 0.25, \pi_4(0) = 0.25$$



# Learning HMM/POMDP

- M = (B, Z, S, P, θ, π)
- Task:
  - Given B, Z, S, and an experience $E \equiv \{z_1, b_1, z_2, b_2, \ldots, z_{T-1}, b_{T-1}, z_T\}$
  - Improve P, θ, π.
    - "Improve" means better match the experience
- How do we do that?
  - EM: Bayesian Again!
    - P(E|M): Use M to explain E
    - Use the explanation to improve P(M|E)
    - C is the background knowledge

$$p(M|EC) = p(M|C)\left[\frac{p(E|MC)}{p(E|C)}\right]$$

# Equations for "Improving based on explanation"

Since the experience $E$ is a combination of $A$ and $O$, that is, $E = OA$, equation 5.7 can be written in other forms to reflect the fact that $A$ can be chosen *independently* of $M$. In that case, we have

$$p(M|EC) = p(M|OAC) = p(M|C)\left[\frac{p(O|AMC)}{p(O|AC)}\right] \qquad (5.8)$$

The proof of equation 5.8 is as follows:

$$\begin{aligned} p(M|OAC) &= p(M|C)\left[\frac{p(OA|MC)}{p(OA|C)}\right] \\ &= p(M|C)\left[\frac{p(A|MC)p(O|AMC)}{p(A|C)p(O|AC)}\right] \\ &= p(M|C)\left[\frac{p(O|AMC)}{p(O|AC)}\right] \end{aligned}$$

Assume A and M are independent: P(A|MC)=P(A|C)

improving

explanation

# The EM Algorithm

- E-Step: Estimate P(E|M) the likelihood of the experience E given the model M
  - Using the model M to explain the experience E
- M-Step: Maximizing the parameters of the model M using the knowledge learned from the experience
  - Using the explanation to improve the model
    - E.g., Baum-Welch Learning Procedure

# Baum-Welch Learning Procedure

Using the explanation of the experience to change the model:

$\overline{\pi}_i(1) \Longleftarrow$ the probability of being in state $s_i$ at time $t = 1$ given $M$ and $E$

$\overline{P}_{ij}[b] \Longleftarrow$ the ratio of the expected number of transitions from state $s_i$ to $s_j$ under action $b$ divided by the expected number of transitions out of state $s_i$ under action $b$, given $M$ and $E$

$\overline{\theta}_i(k) \Longleftarrow$ the ratio of the expected number of times of being in state $s_i$ and observing symbol $z_k$ divided by the expected number of times of being in state $s_i$, given $M$ and $E$

# Update P, θ, π using α, β, γ, ξ

$$\overline{\pi}_i(1) = \gamma_1(i) \qquad \text{for all } s_i \in S$$

(Smooth) Use the whole experience to determine the beginning

$$\overline{P}_{ij}[b] = \frac{\sum_{t=1,b_t=b}^{T} \xi_t(i,j)}{\sum_{t=1,b_t=b}^{T} \gamma_t(i)}$$

From all states $s_i$, how many transit to $s_j$

$$\overline{\theta}_i(k) = \frac{\sum_{t=1,z_t=z_k}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

From all $s_i$, how many look like $z_k$

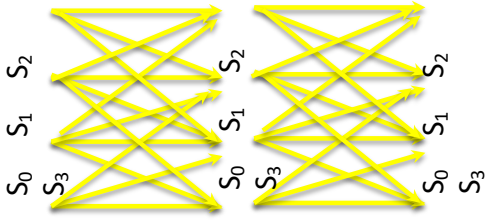
$$\pi_j(t+1) = \frac{\theta_j(z_{t+1}) \sum_i \pi_i(t) \theta_i(z_t) P_{ij}[b_t]}{p(z_t, z_{t+1} | M, C, b_t)}$$

(Smooth) Use the whole model to determine the next state

# Little Prince Example

- Computing α, β, γ, ξ using the experience

E = {rose}, forward, {nothing}, forward, {volcano}




States: $s_0$, $s_1$, $s_2$, $s_3$

Init state distribution: 0.25, 0.25, 0.25, 0.25

Transitions: P( $S_j$ | $S_i$, forward):

$$P(S|s_0) =< 0.3, 0.1, 0.2, 0.4 >,\ P(S|s_1) =< 0.3, 0.3, 0.3, 0.1 >,$$
$$P(S|s_2) =< 0.5, 0.3, 0.1, 0.1 >,\ P(S|s_3) =< 0.1, 0.3, 0.5, 0.1 >$$

Observations Z=<rose, volcano, nothing>

Sensor model:

$$P(Z|s_0) =< 0.4, 0.5, 0.1 >,\ P(\check{Z}|s_1) =< 0.2, 0.6, 0.2 >,$$
$$P(Z|s_2) =< 0.5, 0.3, 0.2 >,\ P(Z|s_3) =< 0.8, 0.1, 0.1 >$$

# Forward Procedure

Compute $\pi_{i_1}(1)\theta_{i_1}(z_1)P_{i_1 i_2}[b_1]\cdots\theta_{i_{T-1}}(z_{T-1})P_{i_{T-1}i_T}[b_{T-1}]\theta_{i_T}(z_T)$ step by step:

$\alpha_t(i)$: the probability at the state $s_i$ at time $t$ given $E$

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

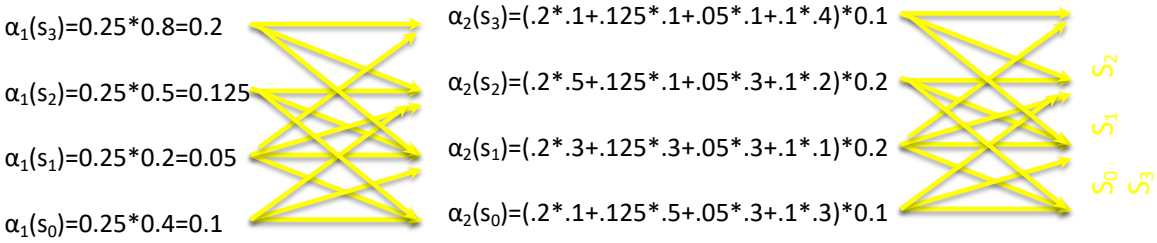
$$\alpha_1(i) = \pi_i\theta_i(z_1)$$

$$\begin{aligned}
\alpha_1(i) &= p(z_1, i_1 = s_i|AMC) \\
\alpha_2(i) &= p(z_1, z_2, i_2 = s_i|AMC) \\
&\vdots \\
\alpha_t(i) &= p(z_1, z_2, \ldots, z_t, i_t = s_i|AMC) \\
&\vdots \\
\alpha_{T-1}(i) &= p(z_1, z_2, \ldots, z_{T-1}, i_{T-1} = s_i|AMC) \\
\alpha_T(i) &= p(z_1, z_2, \ldots, z_{T-1}, z_T, i_T = s_i|AMC)
\end{aligned}$$

$$\alpha_{t+1}(j) = \sum_{i\in S}\alpha_t(i)P_{ij}[b_t]\theta_j(z_{t+1}) \qquad (5.10)$$

# Compute α values

$$\alpha_1(i) = \pi_i \theta_i(z_1) \quad \Rightarrow \quad \alpha_{t+1}(j) = \sum_{i \in S} \alpha_t(i) P_{ij} [b_t] \theta_j(z_{t+1}) \quad \Rightarrow$$

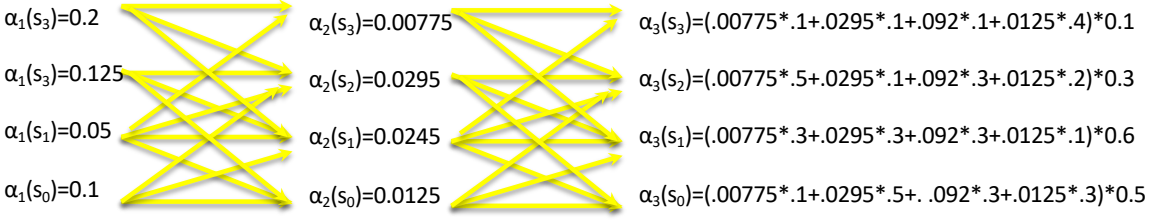



$$P(S|s_0) =< 0.3, 0.1, 0.2, 0.4 >,\ P(S|s_1) =< 0.3, 0.3, 0.3, 0.1 >,$$
$$P(S|s_2) =< 0.5, 0.3, 0.1, 0.1 >,\ P(S|s_3) =< 0.1, 0.3, 0.5, 0.1 >$$

$$P(Z|s_0) =< 0.4, 0.5, 0.1 >,\ P(\check{Z}|s_1) =< 0.2, 0.6, 0.2 >,$$
$$P(Z|s_2) =< 0.5, 0.3, 0.2 >,\ P(Z|s_3) =< 0.8, 0.1, 0.1 >$$

# Compute α values

$$\alpha_{t+1}(j) = \sum_{i \in S} \alpha_t(i) P_{ij}[b_t] \theta_j(z_{t+1})$$




$$P(S|s_0) =< 0.3, 0.1, 0.2, 0.4 >,\ P(S|s_1) =< 0.3, 0.3, 0.3, 0.1 >,$$
$$P(S|s_2) =< 0.5, 0.3, 0.1, 0.1 >,\ P(S|s_3) =< 0.1, 0.3, 0.5, 0.1 >$$
$$P(Z|s_0) =< 0.4, 0.5, 0.1 >,\ P(\check{Z}|s_1) =< 0.2, 0.6, 0.2 >,$$
$$P(Z|s_2) =< 0.5, 0.3, 0.2 >,\ P(Z|s_3) =< 0.8, 0.1, 0.1 >$$

# Compute $\beta_t(i)$ by Backward Procedure

$\beta_t(i)$: given E, the probability of being at the state $s_i$ at time $t$

$$\beta_T(i) = 1 \text{ for all states } s_i$$

$$\begin{aligned}
\beta_{T-1}(i) &= p(z_T|AMC, i_{T-1} = s_i) \\
\beta_{T-2}(i) &= p(z_{T-1}, z_T|AMC, i_{T-2} = s_i) \\
&\vdots \\
\beta_t(i) &= p(z_{t+1}, \ldots, z_{T-1}, z_T|AMC, i_t = s_i) \\
&\vdots \\
\beta_2(i) &= p(z_3, \cdots, z_{T-1}, z_T|AMC, i_2 = s_i) \\
\beta_1(i) &= p(z_2, z_3, \cdots, z_{T-1}, z_T|AMC, i_1 = s_i)
\end{aligned}$$

Each quantity $\beta_t(i)$ specifies the probability of seeing the sequence $z_{t+1}, \ldots, z_T$ if the state at time $t$ is $s_i$.

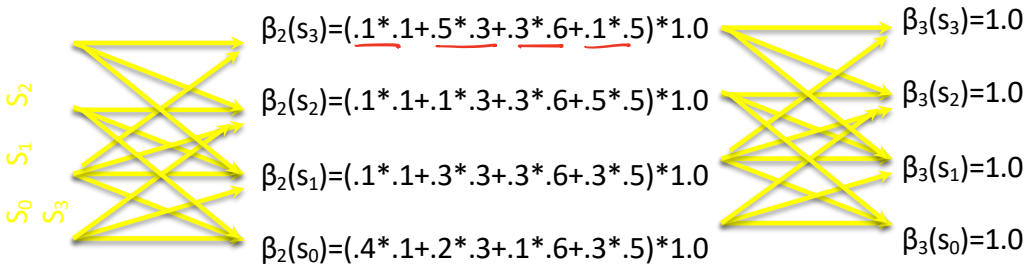
$$\beta_{t-1}(i) = \sum_{j \in S} P_{ij}[b_{t-1}]\theta_j(z_t)\beta_t(j) \tag{5.13}$$

Finally, when all $\beta_1(i)$ are known, $p(O|AMC)$ can be computed by summing up $\beta_1(i)$ on all model states. That is,

$$p(O|AMC) = \sum_{s_i \in S} \pi_i(1)\theta_i(z_1)\beta_1(i) \tag{5.14}$$

# Compute β values

$$\beta_{t-1}(i) = \sum_{j \in S} P_{ij}[b_{t-1}]\theta_j(z_t)\beta_t(j) \qquad \beta_T(i) = 1 \text{ for all states } s_i$$




$$P(S|s_0) =< 0.3, 0.1, 0.2, 0.4 >,\ P(S|s_1) =< 0.3, 0.3, 0.3, 0.1 >,$$
$$P(S|s_2) =< 0.5, 0.3, 0.1, 0.1 >,\ P(S|s_3) =< 0.1, 0.3, 0.5, 0.1 >$$

$$P(Z|s_0) =< 0.4, 0.5, 0.1 >,\ P(Z|s_1) =< 0.2, 0.6, 0.2 >,$$
$$P(Z|s_2) =< 0.5, 0.3, 0.2 >,\ P(Z|s_3) =< 0.8, 0.1, 0.1 >$$

# Compute β values

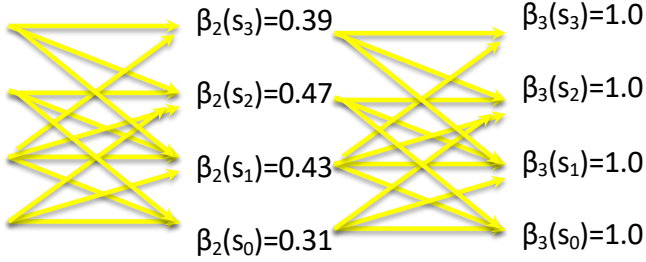
$$\beta_{t-1}(i) = \sum_{j \in S} P_{ij}[b_{t-1}]\theta_j(z_t)\beta_t(j)$$

$\beta_1(s_3)$ = fill in here in the class

$\beta_1(s_2)$=

$\beta_1(s_1)$=

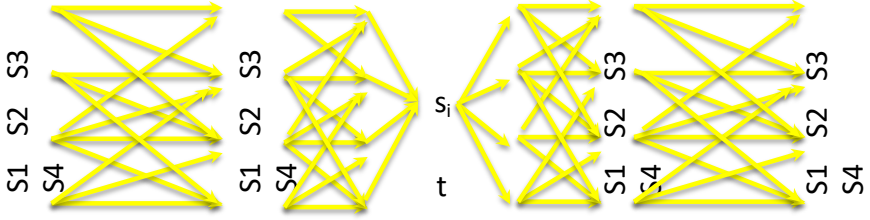
$\beta_1(s_0)$=




$$P(S|s_0) =< 0.3, 0.1, 0.2, 0.4 >,\ P(S|s_1) =< 0.3, 0.3, 0.3, 0.1 >,$$
$$P(S|s_2) =< 0.5, 0.3, 0.1, 0.1 >,\ P(S|s_3) =< 0.1, 0.3, 0.5, 0.1 >$$
$$P(Z|s_0) =< 0.4, 0.5, 0.1 >,\ P(Z|s_1) =< 0.2, 0.6, 0.2 >,$$
$$P(Z|s_2) =< 0.5, 0.3, 0.2 >,\ P(Z|s_3) =< 0.8, 0.1, 0.1 >$$

# $\Upsilon_t(i)$ Value: Putting $\alpha$ and $\beta$ together

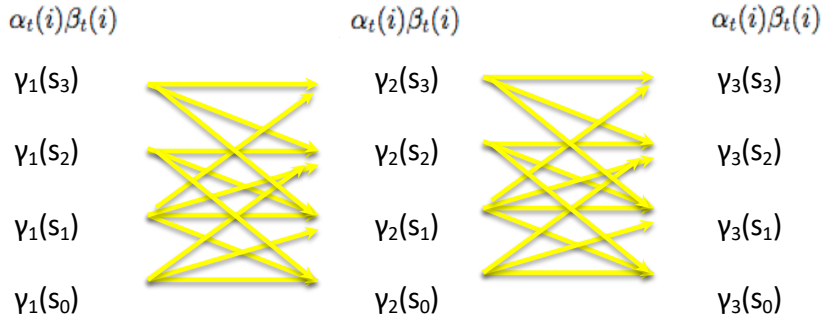
- $\Upsilon_t(i)$ is the probability of being at state $s_i$ at time $t$ given the entire experience $E_{1:T}$



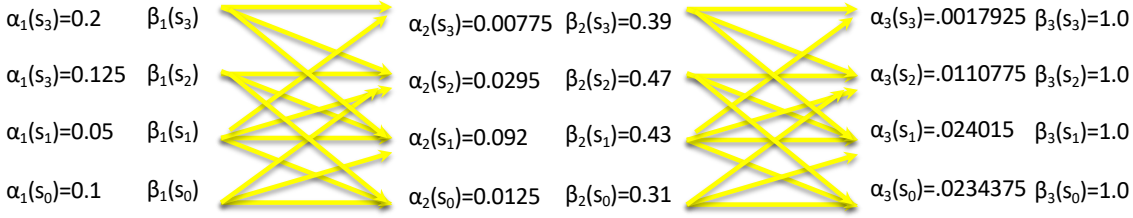

$$
\begin{aligned}
\gamma_t(i) &= p(i_t = s_i | EMC) \\
&= \frac{p(i_t = s_i | AMC) p(O | i_t = s_i, AMC)}{p(O|AMC)} \\
&= \frac{p(O, i_t = s_i | AMC)}{p(O|AMC)} \\
&= \frac{\alpha_t(i)\beta_t(i)}{p(O|AMC)}
\end{aligned}
$$

where $\alpha_t(i)$ accounts for having seen $\{z_1, \ldots, z_t\}$ ending in $s_i$ at time $t$, and $\beta_t(i)$ accounts for seeing $\{z_{t+1}, \ldots, z_T\}$ after time $t$ given state $s_i$ at time $t$.

# Compute γ = α*β

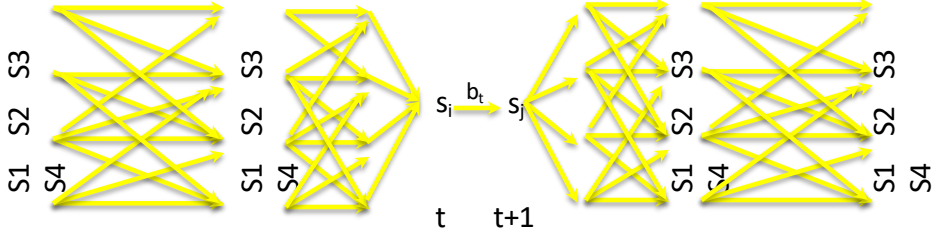

$\alpha_t(i)\beta_t(i)$
$\gamma_1(s_3)$
$\gamma_1(s_2)$
$\gamma_1(s_1)$
$\gamma_1(s_0)$
$\alpha_t(i)\beta_t(i)$
$\gamma_2(s_3)$
$\gamma_2(s_2)$
$\gamma_2(s_1)$
$\gamma_2(s_0)$
$\alpha_t(i)\beta_t(i)$
$\gamma_3(s_3)$
$\gamma_3(s_2)$
$\gamma_3(s_1)$
$\gamma_3(s_0)$

# Putting α and β together



# $\xi_t(i,j)$ Value

- $\xi_t(i,j)$ is the probability of making transition from $s_i$ to $s_j$ at time $t$ in the experience $E_{1:T}$




$$
\begin{aligned}
\xi_t(i,j) &= p(i_t = s_i, i_{t+1} = s_j | EMC) \\
&= \frac{p(i_t = s_i, i_{t+1} = s_j | AMC) p(O | AMC, i_t = s_i, i_{t+1} = s_j)}{p(O|AMC)} \\
&= \frac{p(O, i_t = s_i, i_{t+1} = s_j | AMC)}{p(O|AMC)} \\
&= \frac{\alpha_t(i) \cdot P_{ij}[b_t] \cdot \theta_j(z_{t+1}) \cdot \beta_{t+1}(j)}{p(E|MC)}
\end{aligned}
$$

where $\alpha_t(i)$ accounts for having seen $\{z_1, b_1, \ldots, z_t\}$ ending in state $s_i$ at time $t$, $P_{ij}[b_t] \cdot \theta_{t+1}(z_{t+1})$ accounts for the transition from $s_i$ to $s_j$ under action $b_t$ and then seeing $z_{t+1}$ at time $t+1$, and $\beta_{t+1}(j)$ accounts for seeing $\{z_{t+2}, \ldots, z_T\}$ after time $t+1$ given state $s_j$ at time $t+1$.

# Improve the model by explanation

## Update P, θ, π using α, β, γ, ξ

$$\overline{\pi}_i(1) = \gamma_1(i) \qquad \text{for all } s_i \in S$$

Use the whole experience to determine the beginning

$$\overline{P}_{ij}[b] = \frac{\sum_{t=1,b_t=b}^{T} \xi_t(i,j)}{\sum_{t=1,b_t=b}^{T} \gamma_t(i)}$$

From all $s_i$ in E, how many go to $s_j$

$$\overline{\theta}_i(k) = \frac{\sum_{t=1,z_t=z_k}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

From all $s_i$ in E, how many appear as $z_k$

$$\pi_j(t+1) = \frac{\theta_j(z_{t+1}) \sum_i \pi_i(t) \theta_i(z_t) P_{ij}[b_t]}{p(z_t, z_{t+1}|M, C, b_t)}$$

Use the whole experience to determine the distribution for the next state

# The General EM Algorithm

- E-Step: Estimate P(E|M) the likelihood of the experience E given the model M
  - E.g., computing $\alpha$, $\beta$, $\gamma$, $\xi$ using the experience
  - K-means: assigning data to the (closest) clusters
- M-Step: Maximize the parameters of the model M using the knowledge (e.g., explanations) learned from the experience
  - E.g., update P, $\theta$, $\pi$ using $\alpha$, $\beta$, $\gamma$, $\xi$
  - K-means: move the clusters based on the assignments

# Comments on EM

- The most general and powerful learning method
  - Many existing algorithms are special cases of EM
- Tremendous application potentials
  - Robot navigation, localization, mapping, SLAM, manipulation, planning, etc.
  - Natural language processes (IBM's Watson)
  - Data Mining
  - Gaming that can improve themselves
  - Discovering patterns from genetic and health data