# Bayesian Learning

Olive slides: Alpaydin

Black slides: Mitchell.

## Bayesian Learning

- Probabilistic approach to inference.
- Quantities of interest are governed by prob. dist. and optimal decisions can be made by reasoning about these prob.
- Learning algorithms that directly deal with probabilities.
- Analysis framework for non-probabilistic methods.

# Two Roles for Bayesian Methods

Provides practical learning algorithms:

- Naive Bayes learning
- Bayesian belief network learning
- Combine prior knowledge (prior probabilities) with observed data
- Requires prior probabilities

Provides useful conceptual framework

- Provides “gold standard” for evaluating other learning algorithms
- Additional insight into Occam’s razor

## Basic Probability Formulas

- *Product Rule*: probability $P(A \wedge B)$ of a conjunction of two events A and B:

$$P(A, B) = P(B, A) = P(A \wedge B) = P(A|B)P(B) = P(B|A)P(A)$$

- *Sum Rule*: probability of a disjunction of two events A and B:

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B)$$

- *Theorem of total probability*: if events $A_1, \ldots, A_n$ are mutually exclusive with $\sum_{i=1}^{n} P(A_i) = 1$, then

$$P(B) = \sum_{i=1}^{n} P(B|A_i)P(A_i)$$

## Bayes Theorem

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- $P(h)$ = prior probability that $h$ holds, before seeing the training data
- $P(D)$ = prior probability of observing training data $D$
- $P(D|h)$ = probability of observing $D$ in a world where $h$ holds
- $P(h|D)$ = probability of $h$ holding given observed data $D$
- Some useful tricks:
  - $P(h, D) = P(D, h)$
  - $P(h|D) = \frac{P(h,D)}{P(D)}$
  - $P(D, h) = P(D|h)P(h)$, from $P(D|h) = \frac{P(D,h)}{P(h)}$

## Bayes Theorem: Example

Does patient have cancer or not?

> A patient takes a lab test and the result comes back positive. The test returns a correct positive result in only $98\%$ of the cases in which the disease is actually present, and a correct negative result in only $97\%$ of the cases in which the disease is not present. Furthermore, $.001$ of the entire population have this cancer.

$$P(cancer) = \qquad P(\neg cancer) =$$

$$P(\oplus|cancer) = \qquad P(\ominus|cancer) =$$

$$P(\oplus|\neg cancer) = \qquad P(\ominus|\neg cancer) =$$

How does $P(cancer|\oplus)$ compare to $P(\neg cancer|\oplus)$?

## Bayes Theorem: Example

The test returns a correct positive result in only $98\%$ of the cases in which the disease is actually present, and a correct negative result in only $97\%$ of the cases in which the disease is not present. Furthermore, $.001$ of the entire population have this cancer.

$$
\begin{array}{rl}
P(cancer) = 0.001, given & P(\neg cancer) = 1 - P(cancer) = 1 - 0.001 = 0.999 \\
P(\oplus|cancer) = 0.98, given & P(\ominus|cancer) = 1 - P(\oplus|cancer) = 1 - 0.98 = 0.02 \\
P(\oplus|\neg cancer) = 1 - P(\ominus|\neg cancer) & P(\ominus|\neg cancer) = 0.97, given \\
= 1 - 0.97 = 0.03 &
\end{array}
$$

How does $P(cancer|\oplus)$ compare to $P(\neg cancer|\oplus)$?

$$
\begin{aligned}
P(cancer|\oplus) &= \frac{P(\oplus|cancer)P(cancer)}{P(\oplus)} \\
&= \frac{0.98 \times 0.001}{P(\oplus)} \\
&= \frac{0.00098}{P(\oplus, cancer) + P(\oplus, \neg cancer)} \\
&= \frac{0.00098}{P(\oplus|cancer)P(cancer) + P(\oplus|\neg cancer)P(\neg cancer)} \\
&= \frac{0.00098}{0.98 \times 0.001 + 0.03 \times 0.999} = 0.031664
\end{aligned} \tag{1}
$$

## Conditional Independence

**Definition:** $X$ is *conditionally independent* of $Y$ given $Z$ if the probability distribution governing $X$ is independent of the value of $Y$ given the value of $Z$; that is, if

$$(\forall x_i, y_j, z_k)\ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k)$$

more compactly, we write

$$P(X|Y,Z) = P(X|Z)$$

Example: $Thunder$ is conditionally independent of $Rain$, given $Lightning$

$$P(Thunder|Rain, Lightning) = P(Thunder|Lightning)$$

## Choosing Hypotheses

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Generally want the most probable hypothesis given the training data

*Maximum a posteriori* hypothesis $h_{MAP}$:

$$\begin{aligned} h_{MAP} &= \arg\max_{h \in H} P(h|D) \\ &= \arg\max_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \arg\max_{h \in H} P(D|h)P(h) \end{aligned}$$

# Choosing Hypotheses

- If all hypotheses are equally probable a priori:

$$P(h_i) = P(h_j), \forall h_i, h_j,$$

then, $h_{MAP}$ reduces to:

$$h_{ML} \equiv \underset{h \in H}{\operatorname{argmax}}\, P(D|h).$$

$\longrightarrow$ Maximum Likelihood hypothesis.

## Brute Force MAP Hypothesis Learner

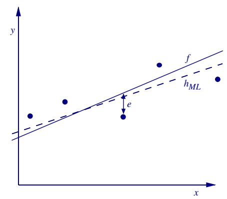
1. For each hypothesis $h$ in $H$, calculate the posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

2. Output the hypothesis $h_{MAP}$ with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{\operatorname{argmax}} P(h|D)$$

## Learning A Real Valued Function




Consider any real-valued target function $f$

Training examples $\langle x_i, d_i \rangle$, where $d_i$ is noisy training value

- $d_i = f(x_i) + e_i$
- $e_i$ is random variable (noise) drawn independently for each $x_i$ according to some Gaussian distribution with mean=0

Then the maximum likelihood hypothesis $h_{ML}$ is the one that minimizes the sum of squared errors:

$$h_{ML} = \arg\min_{h \in H} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

## Setting up the Stage

- Probability density function:

$$p(x_0) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon} P(x_0 \le x < x_0 + \epsilon)$$

- ML hypothesis

$$h_{ML} = \operatorname*{argmax}_{h \in H} p(D|h)$$

- Training instances $\langle x_1, ..., x_m \rangle$ and target values $\langle d_1, ..., d_m \rangle$, where $d_i = f(x_i) + e_i$.

- Assume training examples are mutually independent given $h$,

$$h_{ML} = \operatorname*{argmax}_{h \in H} \prod_{i=1}^{m} p(d_i|h)$$

Note: $p(a, b|c) = p(a|b, c) \cdot p(b|c) = p(a|c) \cdot p(b|c)$

## Derivation of ML for Func. Approx.

From $h_{ML} = \text{argmax}_{h \in H} \prod_{i=1}^{m} p(d_i|h)$:

- Since $d_i = f(x_i) + e_i$ and $e_i \sim \mathcal{N}(0, \sigma^2)$, it must be:

$$d_i \sim \mathcal{N}(f(x_i), \sigma^2).$$

  - $x \sim \mathcal{N}(\mu, \sigma^2)$ means random variable $x$ is normally distributed with mean $\mu$ and variance $\sigma^2$.

- Using pdf of $\mathcal{N}$:

$$h_{ML} = \underset{h \in H}{\text{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(d_i - \mu)^2}{2\sigma^2}}.$$

$$h_{ML} = \underset{h \in H}{\text{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(d_i - h(x_i))^2}{2\sigma^2}}.$$

## Derivation of ML

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(d_i - h(x_i))^2}{2\sigma^2}}.$$

- Get rid of constant factor $\frac{1}{\sqrt{2\pi\sigma^2}}$, and put on log:

$$\begin{aligned} h_{ML} &= \underset{h \in H}{\operatorname{argmax}} \ln \prod_{i=1}^{m} e^{-\frac{(d_i - h(x_i))^2}{2\sigma^2}} \\ &= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} \ln e^{-\frac{(d_i - h(x_i))^2}{2\sigma^2}} \\ &= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\frac{(d_i - h(x_i))^2}{2\sigma^2} \\ &= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} (d_i - h(x_i))^2 \qquad (2) \end{aligned}$$

# Least Square as ML

Assumptions

- Observed training values $d_i$ generated by adding random noise to true target value, where noise has a normal distribution with zero mean.
- All hypotheses are equally probable (uniform prior).
  - Note: it is possible that $MAP \neq ML$!

Limitations

- Possible noise in $x_i$ not accounted for.

## Minimum Description Length

Occam's razor: prefer the shortest hypothesis.

$$
\begin{aligned}
h_{MAP} &= \operatorname*{argmax}_{h \in H} P(D|h)P(h) \\
h_{MAP} &= \operatorname*{argmax}_{h \in H} \log_2 P(D|h) + \log_2 P(h) \\
h_{MAP} &= \operatorname*{argmin}_{h \in H} - \log_2 P(D|h) - \log_2 P(h)
\end{aligned}
$$

Surprisingly, the above can be interpreted as $h_{MAP}$ preferring shorter hypotheses, assuming a particular encoding scheme is used for the hypothesis and the data.

According to information theory, the shortest code length for a message occurring with probability $p_i$ is $-\log_2 p_i$ bits.

# MDL

$$h_{MAP} = \operatorname*{argmin}_{h \in H} -\log_2 P(D|h) - \log_2 P(h)$$

- $L_C(i)$: description length of message $i$ with respect to code $C$.
- $-\log_2 P(h)$: description length of $h$ under optimal coding $C_H$ for the hypothesis space $H$.

$$L_{C_H}(h) = -\log_2 P(h)$$

- $-\log_2 P(D|h)$: description length of training data $D$ given hypothesis $h$, under optimal encoding $C_{D|H}$.

$$L_{C_{D|H}}(D|h) = -\log_2 P(D|h)$$

- Finally, we get:

$$h_{MAP} = \operatorname*{argmin}_{h \in H} L_{C_{D|H}}(D|h) + L_{C_H}(h)$$

# MDL

- MAP:

$$h_{MAP} = \underset{h \in H}{\operatorname{argmin}} L_{C_{D|H}}(D|h) + L_{C_H}(h)$$

- MDL: Choose $h_{MDL}$ such that:

$$h_{MDL} = \underset{h \in H}{\operatorname{argmin}} L_{C_1}(h) + L_{C_2}(D|h)$$

  which is the hypothesis that minimizes the **combined length** of the hypotheis itself, and the data described by the hypothesis.

- $h_{MDL} = h_{MAP}$ if $C_1 = C_H$ and $C_2 = C_{D|H}$.

## Bayes Optimal Classifier

- What is the most probable hypothesis given the training data, **vs.** What is the most probable classification?
- Example:
  - $P(h_1|D) = 0.4$, $P(h_2|D) = 0.3$, $P(h_3|D) = 0.3$.
  - Given a new instance $x$, $h_1(x) = 1$, $h_2(x) = 0$, $h_3(x) = 0$.
  - In this case, probability of $x$ being positive is only 0.4.

## Bayes Optimal Classification

If a new instance can take classification $v_j \in V$, then the probability $P(v_j|D)$ of correct classification of new instance being $v_j$ is:

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i) P(h_i|D)$$

Thus, the optimal classification is

$$\operatorname*{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i) P(h_i|D).$$

## Bayes Optimal Classifier

What is the assumption for the following to work?

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Let's consider $H = \{h, \neg h\}$:

$$\begin{aligned}
P(v|D) &= P(v,h|D) + P(v,\neg h|D) \\
&= \frac{P(v,h,D)}{P(D)} + \frac{P(v,\neg h,D)}{P(D)} \\
&= \frac{P(v|h,D)P(h|D)P(D)}{P(D)} \\
&\quad + \frac{P(v|\neg h,D)P(\neg h|D)P(D)}{P(D)} \\
&\quad \{\text{if } P(v|h,D) = P(v|h), \text{ etc.}\} \\
&= P(v|h)P(h|D) + P(v|\neg h)P(\neg h|D)
\end{aligned}$$

## Bayes Optimal Classifier: Example

- $P(h_1|D) = 0.4$, $P(h_2|D) = 0.3$, $P(h_3|D) = 0.3$.
- Given a new instance $x$, $h_1(x) = 1$, $h_2(x) = 0$, $h_1(x) = 0$.
  - $P(\ominus|h_1) = 0$, $P(\oplus|h_1) = 1$, etc.
  - $P(\oplus|D) = 0.4 + 0 + 0$, $P(\ominus|D) = 0 + 0.3 + 0.3 = 0.6$
  - Thus, $\mathrm{argmax}_{v \in O\{\oplus,\ominus\}} P(v|D) = \ominus$.
- Bayes optimal classifiers maximize the probability that a new instance is correctly classified, given the available data, hypothesis space $H$, and prior probabilities over $H$.
- Some oddities: The resulting hypotheis can be outside of the hypothesis space.

## Gibbs Sampling

Finding $\mathrm{argmax}_{v \in V} P(v|D)$ by considering every hypothesis $h \in H$ can be infeasible. A less optimal, but error-bounded version is **Gibbs sampling**:

1. Randomly pick $h \in H$ with probability $P(h|D)$.

2. Use $h$ to classify the new instance $x$.

The result is that missclassification rate is at most $2\times$ that of BOC.

## Naive Bayes Classifier

Given attribute values $\langle a_1, a_2, ..., a_n \rangle$, give the classification $v \in V$:

$$v_{MAP} = \operatorname*{argmax}_{v_j \in V} P(v_j | a_1, a_2, ..., a_n)$$

$$\begin{aligned} v_{MAP} &= \operatorname*{argmax}_{v_j \in V} \frac{P(a_1, a_2, ..., a_n | v_j) P(v_j)}{P(a_1, a_2, ..., a_n)} \\ &= \operatorname*{argmax}_{v_j \in V} P(a_1, a_2, ..., a_n | v_j) P(v_j) \end{aligned}$$

- Want to estimate $P(a_1, a_2, ..., a_n | v_j)$ and $P(v_j)$ from training data.

## Naive Bayes

- $P(v_j)$ is easy to calculate: Just count the frequency.
- $P(a_1, a_2, ..., a_n|v_j)$ takes the number of posible instances $\times$ number of possible target values.
- $P(a_1, a_2, ..., a_n|v_j)$ can be approximated as

$$P(a_1, a_2, ..., a_n|v_j) = \prod_i P(a_i|v_j).$$

- From this naive Bayes classifier is defined as:

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)$$

- Naive Bayes only takes number of distinct attribute values $\times$ number of distinct target values.

Naive Bayes uses cond. indep. to justify

$$\begin{aligned} P(X, Y|Z) &= P(X|Y, Z)P(Y|Z) \\ &= P(X|Z)P(Y|Z) \end{aligned}$$

# Naive Bayes Algorithm

Naive_Bayes_Learn($examples$)

For each target value $v_j$

$\hat{P}(v_j) \leftarrow$ estimate $P(v_j)$

For each attribute value $a_i$ of each attribute $a$

$\hat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

Classify_New_Instance($x$)

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} \hat{P}(v_j) \prod_i \hat{P}(x_i|v_j)$$

# Naive Bayes: Example

Consider *PlayTennis* again, and new instance:

$$x = \langle Outlk = sun, Temp = cool, Humid = high, Wind = strong \rangle$$

$$V = \{Yes, No\}$$

Want to compute:

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_i P(x_i | v_j)$$

$$P(Y)\ P(sun|Y)\ P(cool|Y)\ P(high|Y)\ P(strong|Y) = .005$$

$$P(N)\ P(sun|N)\ P(cool|N)\ P(high|N)\ P(strong|N) = .021$$

Thus, $v_{NB} = No$

## Naive Bayes: Subtleties

1. Conditional independence assumption is often violated

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$

- ...but it works surprisingly well anyway. Note don't need estimated posteriors $\hat{P}(v_j|x)$ to be correct; need only that

$$\operatorname*{argmax}_{v_j \in V} \hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = \operatorname*{argmax}_{v_j \in V} P(v_j) P(a_1 \ldots, a_n | v_j)$$

- Naive Bayes posteriors often unrealistically close to 1 or 0.

## Naive Bayes: Subtleties

What if none of the training instances with target value $v_j$ have attribute value $a_i$? Then

$$\hat{P}(a_i|v_j) = 0, \text{ and...}$$

$$\hat{P}(v_j) \prod_i \hat{P}(a_i|v_j) = 0$$

Typical solution is Bayesian estimate for $\hat{P}(a_i|v_j)$

$$\hat{P}(a_i|v_j) \leftarrow \frac{n_c + mp}{n + m}$$

where

- $n$ is number of training examples for which $v = v_j$,
- $n_c$ number of examples for which $v = v_j$ and $a = a_i$
- $p$ is prior estimate for $\hat{P}(a_i|v_j)$
- $m$ is weight given to prior (i.e. number of "virtual" examples)

## Extra Slides: Will be covered, time permitting

# Expectation Maximization (EM)

When to use:

- Data is only partially observable
- Unsupervised clustering (target value unobservable)
- Supervised learning (some instance attributes unobservable)

Some uses:

- Train Bayesian Belief Networks
- Unsupervised clustering (AUTOCLASS)
- Learning Hidden Markov Models

# EM for Estimating $k$ Means

Given:

- Instances from $X$ generated by mixture of $k$ Gaussian distributions
- Unknown means $\langle \mu_1, \ldots, \mu_k \rangle$ of the $k$ Gaussians
- Don't know which instance $x_i$ was generated by which Gaussian

Determine:

- Maximum likelihood estimates of $\langle \mu_1, \ldots, \mu_k \rangle$

Think of full description of each instance as $y_i = \langle x_i, z_{i1}, z_{i2} \rangle$, where

- $z_{ij}$ is 1 if $x_i$ generated by $j$th Gaussian
- $x_i$ observable
- $z_{ij}$ unobservable

## EM for Estimating $k$ Means

EM Algorithm: Pick random initial $h = \langle \mu_1, \mu_2 \rangle$, then iterate

step: Calculate the expected value $E[z_{ij}]$ of each hidden variable $z_{ij}$, assuming the current hypothesis $h = \langle \mu_1, \mu_2 \rangle$ holds.

$$
\begin{aligned}
E[z_{ij}] &= \frac{p(x = x_i | \mu = \mu_j)}{\sum_{n=1}^{2} p(x = x_i | \mu = \mu_n)} \\
&= \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i - \mu_n)^2}}
\end{aligned}
$$

step: Calculate a new maximum likelihood hypothesis $h' = \langle \mu'_1, \mu'_2 \rangle$, assuming the value taken on by each hidden variable $z_{ij}$ is its expected value $E[z_{ij}]$ calculated above. Replace $h = \langle \mu_1, \mu_2 \rangle$ by $h' = \langle \mu'_1, \mu'_2 \rangle$.

$$
\mu_j \leftarrow \frac{\sum_{i=1}^{m} E[z_{ij}]\ x_i}{\sum_{i=1}^{m} E[z_{ij}]}
$$

# EM Algorithm

Converges to local maximum likelihood $h$

and provides estimates of hidden variables $z_{ij}$

In fact, local maximum in $E[\ln P(Y|h)]$

- $Y$ is complete (observable plus unobservable variables) data
- Expected value is taken over possible values of unobserved variables in $Y$

## General EM Problem

Given:

- Observed data $X = \{x_1, \ldots, x_m\}$
- Unobserved data $Z = \{z_1, \ldots, z_m\}$
- Parameterized probability distribution $P(Y|h)$, where
  - $Y = \{y_1, \ldots, y_m\}$ is the full data $y_i = x_i \cup z_i$
  - $h$ are the parameters

Determine:

- $h$ that (locally) maximizes $E[\ln P(Y|h)]$

## General EM Method

Define likelihood function $Q(h'|h)$ which calculates $Y = X \cup Z$ using observed $X$ and current parameters $h$ to estimate $Z$

$$Q(h'|h) \leftarrow E[\ln P(Y|h')|h, X]$$

EM Algorithm:

*Estimation (E) step:* Calculate $Q(h'|h)$ using the current hypothesis $h$ and the observed data $X$ to estimate the probability distribution over $Y$.

$$Q(h'|h) \leftarrow E[\ln P(Y|h')|h, X]$$

*Maximization (M) step:* Replace hypothesis $h$ by the hypothesis $h'$ that maximizes this $Q$ function.

$$h \leftarrow \underset{h'}{\operatorname{argmax}}\, Q(h'|h)$$

## Derivation of $k$-Means

- Hypothesis $h$ is parameterized by $\theta = \langle \mu_1 ... \mu_k \rangle$.
- Observed data $X = \{\langle x_i \rangle\}$
- Hidden variables $Z = \{\langle z_{i1}, ..., z_{ik} \rangle\}$:
  - $z_{ik} = 1$ if input $x_i$ is generated by th $k$-th normal dist.
  - For each input, $k$ entries.
- First, start with defining $\ln p(Y|h)$.

## **Deriving** $\ln P(Y|h)$

$$p(y_i|h') = p(x_i, z_{i1}, z_{i2}, \ldots, z_{ik}|h') = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\sum_{j=1}^{k} z_{ij}(x_i - \mu'_j)^2}$$

Note that the vector $\langle z_{i1}, \ldots, z_{ik} \rangle$ contains only a single 1 and all the rest are 0.

$$\begin{aligned}
\ln P(Y|h') &= \ln \prod_{i=1}^{m} p(y_i|h') \\
&= \sum_{i=1}^{m} \ln p(y_i|h') \\
&= \sum_{i=1}^{m} \left( \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{j=1}^{k} z_{ij}(x_i - \mu'_j)^2 \right)
\end{aligned}$$

# Deriving $E[\ln P(Y|h)]$

Since $P(Y|h')$ is a linear function of $z_{ij}$, and since $E[f(z)] = f(E[z])$,

$$\begin{aligned} E[\ln P(Y|h')] &= E\left[\sum_{i=1}^{m}\left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\sum_{j=1}^{k} z_{ij}(x_i - \mu'_j)^2\right)\right] \\ &= \sum_{i=1}^{m}\left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2\right) \end{aligned}$$

Thus,

$$\begin{aligned} Q(h'|h) &= Q(\langle \mu'_1, \ldots, \mu'_k \rangle | h) \\ &= \sum_{i=1}^{m}\left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}\sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2\right) \end{aligned}$$

## **Finding** $\operatorname{argmax}_{h'} Q(h'|h)$

With

$$E[z_{ij}] = \frac{e^{-\frac{1}{2\sigma^2}(x_i-\mu_j)^2}}{\sum_{n=1}^{2} e^{-\frac{1}{2\sigma^2}(x_i-\mu_n)^2}}$$

we want to find $h'$ such that

$$\begin{aligned}\operatorname*{argmax}_{h'} Q(h'|h) &= \operatorname*{argmax}_{h'} \sum_{i=1}^{m} \left( \ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2 \right) \\ &= \operatorname*{argmin}_{h'} \sum_{i=1}^{m} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2,\end{aligned}$$

which is minimized by

$$\mu_j \leftarrow \frac{\sum_{i=1}^{m} E[z_{ij}] x_i}{\sum_{i=1}^{m} E[z_{ij}]}.$$

# Deriving the Update Rule

Set the derivative of the quantity to be minimized to be zero:

$$
\begin{aligned}
& \frac{\partial}{\partial \mu'_j} \sum_{i=1}^{m} \sum_{j=1}^{k} E[z_{ij}](x_i - \mu'_j)^2 \\
= \quad & \frac{\partial}{\partial \mu'_j} \sum_{i=1}^{m} E[z_{ij}](x_i - \mu'_j)^2 \\
= \quad & 2 \sum_{i=1}^{m} E[z_{ij}](x_i - \mu'_j) = 0
\end{aligned}
$$

$$
\begin{aligned}
\sum_{i=1}^{m} E[z_{ij}]x_i - \sum_{i=1}^{m} E[z_{ij}]\mu'_j &= 0 \\
\sum_{i=1}^{m} E[z_{ij}]x_i &= \mu'_j \sum_{i=1}^{m} E[z_{ij}] \\
\mu'_j &= \frac{\sum_{i=1}^{m} E[z_{ij}]x_i}{\sum_{i=1}^{m} E[z_{ij}]}
\end{aligned}
$$

See Bishop (1995) *Neural Networks for Pattern Recognition*, Oxford U Press. pp. 63–64.

# Losses and Risks

- Actions: $\alpha_i$
- Loss of $\alpha_i$ when the state is $C_k$ : $\lambda_{ik}$
- Expected risk (Duda and Hart, 1973)

$$R(\alpha_i \mid \mathbf{x}) = \sum_{k=1}^{K} \lambda_{ik} P(C_k \mid \mathbf{x})$$

$$\text{choose } \alpha_i \text{ if } R(\alpha_i \mid \mathbf{x}) = \min_k R(\alpha_k \mid \mathbf{x})$$

# Losses and Risks: 0/1 Loss



$$\lambda_{ik} = \begin{cases} 0 \text{ if } i = k \\ 1 \text{ if } i \neq k \end{cases}$$

$$\begin{aligned} R(\alpha_i \mid \mathbf{x}) &= \sum_{k=1}^{K} \lambda_{ik} P(C_k \mid \mathbf{x}) \\ &= \sum_{k \neq i} P(C_k \mid \mathbf{x}) \\ &= 1 - P(C_i \mid \mathbf{x}) \end{aligned}$$

*For minimum risk, choose the most probable class*

# Losses and Risks: Reject

$$\lambda_{ik} = \begin{cases} 0 & \text{if } i = k \\ \lambda & \text{if } i = K+1, \quad 0 < \lambda < 1 \\ 1 & \text{otherwise} \end{cases}$$

$$R(\alpha_{K+1} \mid \mathbf{x}) = \sum_{k=1}^{K} \lambda P(C_k \mid \mathbf{x}) = \lambda$$

$$R(\alpha_i \mid \mathbf{x}) = \sum_{k \neq i} P(C_k \mid \mathbf{x}) = 1 - P(C_i \mid \mathbf{x})$$

choose $C_i$ if $P(C_i \mid \mathbf{x}) > P(C_k \mid \mathbf{x}) \ \forall k \neq i$ and $P(C_i \mid \mathbf{x}) > 1 - \lambda$

reject otherwise

# Discriminant Functions

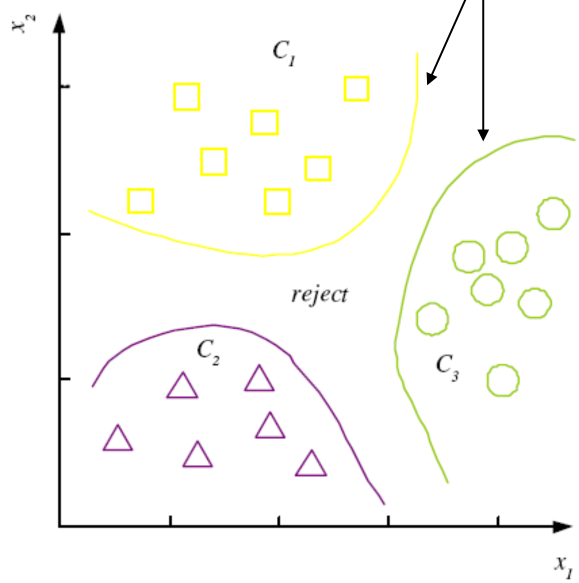
choose $C_i$ if $g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})$

$$g_i(\mathbf{x}) = \begin{cases} -R(\alpha_i \mid \mathbf{x}) \\ P(C_i \mid \mathbf{x}) \\ p(\mathbf{x} \mid C_i)P(C_i) \end{cases}$$

$K$ decision regions $\mathcal{R}_1,\ldots,\mathcal{R}_K$

$$\mathcal{R}_i = \{\mathbf{x} \mid g_i(\mathbf{x}) = \max_k g_k(\mathbf{x})\}$$

$g_i(\mathbf{x}), i = 1,\ldots,K$



# $K$=2 Classes

- Dichotomizer ($K$=2) vs Polychotomizer ($K$>2)
- $g(\boldsymbol{x}) = g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$

$$\text{choose} \begin{cases} C_1 \text{ if } g(\mathbf{x}) > 0 \\ C_2 \text{ otherwise} \end{cases}$$

- *Log odds:* $\log \frac{P(C_1 \mid \mathbf{x})}{P(C_2 \mid \mathbf{x})}$

# Utility Theory

- Prob of state *k* given exidence **x**: $P\ (S_k \mid \boldsymbol{x})$
- Utility of $\alpha_i$ when state is *k*: $U_{ik}$
- Expected utility:

$$EU(\alpha_i \mid \mathbf{x}) = \sum_k U_{ik} P(S_k \mid \mathbf{x})$$

$$\text{Choose } \alpha_i \text{ if } EU(\alpha_i \mid \mathbf{x}) = \max_j EU(\alpha_j \mid \mathbf{x})$$

# Association Rules

- Association rule: $X \to Y$
- *People who buy/click/visit/enjoy X are also likely to buy/click/visit/enjoy Y.*
- A rule implies association, not necessarily causation.

# Association measures



- Support ($X \rightarrow Y$):

$$P(X,Y) = \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers}\}}$$

- Confidence ($X \rightarrow Y$):

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)}$$

$$= \frac{\#\{\text{customers who bought } X \text{ and } Y\}}{\#\{\text{customers who bought } X\}}$$

- Lift ($X \rightarrow Y$):

$$= \frac{P(X,Y)}{P(X)P(Y)} = \frac{P(Y \mid X)}{P(Y)}$$

# References