Bayesian Learning
- based on Tom Mitchell's slides -

December 1, 2020

- Bayesian belief networks
- Expectation Maximization algorithm

Naive Bayes Algorithm

Naive_Bayes_Learn($examples$)

For each target value v_j

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

For each attribute value a_i of each attribute a
$\widehat{P}(a_i|v_j) \leftarrow$ estimate $P(a_i|v_j)$

Classify_New_Instance(x)

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} \widehat{P}(v_j) \prod_{a_i \in x} \widehat{P}(a_i|v_j)$$

Naive Bayes: Example

Consider *PlayTennis* again, and new instance

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

Want to compute:

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

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

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

$$\rightarrow v_{NB} = n$$

Bayesian Belief Networks (also called Bayes Nets)

Interesting because:

- Naive Bayes assumption of conditional independence too restrictive
- But it's intractable without some such assumptions...
- Bayesian Belief networks describe conditional independence among *subsets* of variables

→ allows combining prior knowledge about (in)dependencies among variables with observed training data

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

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

or, more compactly, we write

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

Example 1 *Thunder is conditionally independent of Rain, given Lightning*

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

Naive Bayes uses conditional independence to justify the following equation which very useful computationally:

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

The network represents a set of conditional independence assertions:

- Each node is asserted to be conditionally independent of its non-descendants, given its immediate predecessors.
- Directed acyclic graph

Represents joint probability distribution over all variables

- e.g., $P(Storm, BusTourGroup, \ldots, ForestFire)$

- In general,
$$P(y_1, \ldots, y_n) = \prod_{i=1}^{n} P(y_i | Parents(Y_i))$$
where $Parents(Y_i)$ denotes immediate predecessors of Y_i in graph
- Therefore, the joint distribution is fully defined by the graph, plus the $P(y_i|Parents(Y_i))$

Inference in Bayesian Networks

How can one infer the (probabilities of) values of one or more network variables, given observed values of others?

- Bayes net contains all information needed for this inference
- If only one variable with unknown value, easy to infer it
- In general case, problem is NP hard

In practice, can succeed in many cases

- Exact inference methods work well for some network structures
- Monte Carlo methods "simulate" the network randomly to calculate approximate solutions

Learning of Bayesian Networks

Several variants of this learning task

- Network structure might be *known* or *unknown*
- Training examples might provide values of *all* network variables, or just *some*

If structure known and observe all variables then it's as easy as training a Naive Bayes classifier.

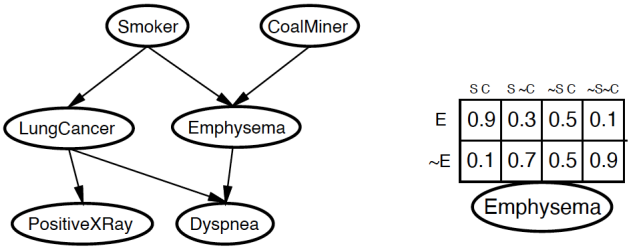
Suppose structure known, variables partially observable

e.g., observe *ForestFire, Storm, BusTourGroup, Thunder*, but not *Lightning, Campfire...*

- Similar to training neural network with hidden units (not covered yet)
- In fact, can learn network conditional probability tables using gradient ascent (not covered yet)!
- Converge to network h that (locally) maximizes $P(D|h)$

Gradient Ascent for Bayes Nets

This section is based on the 1997 paper *Adaptive Probabilistic Networks with Hidden Variables*, by John Binder, D. Koller, S. Russell, and K. Kanazawa, published in Machine Learning, 29, 213-244. It poses (and provides one answer) to the question *How can a probabilistic network be learned from data?*. In the following we assume a known network structure with hidden variables and that what we need to learn are the CP Tables.



(a) **(b)**

Figure 1: (a) A simple probabilistic network showing a proposed causal model. (b) A node with associated conditional probability table. The table gives the conditional probability of each possible value of the variable *Emphysema*, given each possible combination of values of the parent nodes *Smoker* and *CoalMiner*.

According to the table we have $P(Emphysema|Smoker, CoalMiner) = 0.9$, etc.

There are several reasons why the hidden variables are assumed:

1. Any particular variable may not be hidden in all the observed cases
2. The hidden variable might be one that we are interested in querying (several papers written in the '80s discuss this)
3. Networks with hidden variables can be *more compact* than the corresponding fully observable network (see Figure **??**).
4. With hidden variables it is possible to take advantage of local structure (if known)

   $\Longrightarrow$ more concise representation for the joint distribution on the observable variables
   $\Longrightarrow$ possible to learn from fewer examples.

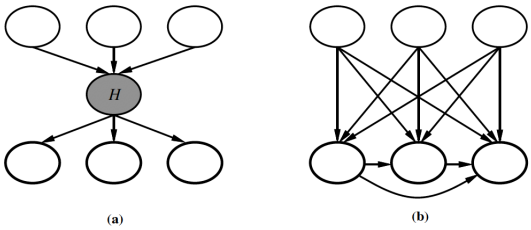


Figure 2: (a) A probabilistic network with a 2-valued hidden variable, labelled H; all other variables are 3-valued. Thus the network requires 45 independent parameters. (b) The corresponding fully observable network, following arc reversal and node removal. The network now requires 708 parameters.

Task

GIVEN: 1. A network structure and initial (possibly randomly generated) values for the CPTs.

2. A set of cases $\mathbf{D} = \{D_1, \ldots, D_m\}$.
   - Assume that the cases are generated independently from some underlying distribution.
3. Each data case provides the values of some subset of the variables;
4. this subset may differ from case to case.

OBJECTIVE: Find the CPT parameters w that *best model the data*, where w_{ijk} denotes a specific CPT entry: the probability that variable X_i takes on its jth possible value assignment given that its parents u_i take on their kth possible value assignment.

That is,

$$w_{ijk} = P(X_i = y_{ij} | Parents(Y_i) = \text{the list } u_{ik} \text{ of values})$$

e.g., if $X_i = Dyspnea$, then u_{ik} might be $\langle Emphysema = T, LungCancer = F \rangle$

To operationalize the phrase *best model the data*, assume that each possible setting of w is equally likely a priori, so that the maximum likelihood (ML) model is appropriate.

This means that the aim is to maximize $P_{\mathbf{w}}(\mathbf{D})$, that is, the probability assigned by the network to the observed data when the CPT parameters are set to $\mathbf{w}$.

- **Idea in Gradient Ascent**: Need to maximize a quantity. Use gradient - the vector of its partial derivatives with respect to its variables and adjust these in the direction of the gradient.

The Gradient Ascent Training for Bayesian networks

View the probability $P_{\mathbf{w}}(\mathbf{D})$ as a function of the CPT entries $\mathbf{w}$.

$\Longrightarrow$ Reduces the problem to one of finding the maximum of a multivariate nonlinear function.

It is easier to maximize the log-likelihood function $\ln P_{\mathbf{w}}(\mathbf{D})$, since the two functions are monotonically related and hence maximizing one is equivalent to maximizing the other.

The simplest approach is *gradient ascent* (also known as "hill climbing").

- At each point $\mathbf{w}$, it computes $\nabla\mathbf{w}$, the gradient vector of partial derivatives with respect to the CPT entries.
- The algorithm then takes a small step in the direction of the gradient to the point $\mathbf{w} + \alpha\nabla\mathbf{w}$, where α is the step-size parameter. This simple algorithm converges to a local optimum for small enough α.

For our problem we need to modify the algorithm to account for the constraints on $\mathbf{w}$:

- Since $\mathbf{w}$ are probabilities, $w_{ijk} \in [0, 1]$ and $\sum_j w_{ijk} = 1$, which is done by a renormalization of these values.
- The algorithm terminates when a local maximum is reached.

Derivation of the gradient formula

By independence of the data cases, we have

$$P_{\mathbf{w}}(\mathbf{D}) = \prod_{l=1}^{m} P_{\mathbf{w}}(D_l)$$

Hence

$$\ln P_{\mathbf{w}}(\mathbf{D}) = \sum_{l=1}^{m} \ln P_{\mathbf{w}}(D_l)$$

Therefore,

$$\begin{aligned} \frac{\partial \ln P_{\mathbf{w}}(\mathbf{D})}{\partial w_{ijk}} &= \sum_{l=1}^{m} \frac{\partial \ln P_{\mathbf{w}}(D_l)}{\partial w_{ijk}} \\ &= \sum_{l=1}^{m} \frac{1}{P_{\mathbf{w}}(D_l)} \times \frac{\partial P_{\mathbf{w}}(D_l)}{\partial w_{ijk}}. \end{aligned}$$

In order to get an expression in terms of information local to the parameter w_{ijk}, introduce X_i and U_i by averaging over their possible values:

$\frac{1}{P_{\mathbf{w}}(D_l)} \times \frac{\partial P_{\mathbf{w}}(D_l)}{\partial w_{ijk}} =$

$= \frac{1}{P_{\mathbf{w}}(D_l)} \times \frac{\partial}{\partial w_{ijk}} \left(\sum_{j',k'} P_{\mathbf{w}}(D_l | x_{ij'}, u_{ik'}) P_{\mathbf{w}}(x_{ij'}, u_{ik'}) \right)$

using the chain rule for probabilities

$= \frac{1}{P_{\mathbf{w}}(D_l)} \times \frac{\partial}{\partial w_{ijk}} \left(\sum_{j',k'} P_{\mathbf{w}}(D_l | x_{ij'}, u_{ik'}) P_{\mathbf{w}}(x_{ij'} | u_{ik'}) P_{\mathbf{w}}(u_{ik'}) \right)$

only one term in the summation will produce a result different from 0: $j' = j$ and $k = k'$

$= \frac{1}{P_{\mathbf{w}}(D_l)} \times P_{\mathbf{w}}(D_l | x_{ij}, u_{ik}) P_{\mathbf{w}}(x_{ij} | u_{ik}) P_{\mathbf{w}}(u_{ik})$

using Bayes Theorem, we obtain

$= \frac{1}{P_{\mathbf{w}}(D_l) P_{\mathbf{w}}(x_{ij}, u_{ik})} \times P_{\mathbf{w}}(x_{ij}, u_{ik} | D_l) P_{\mathbf{w}}(D_l) P_{\mathbf{w}}(u_{ik})$

$= \frac{P_{\mathbf{w}}(x_{ij}, u_{ik} | D_l)}{P_{\mathbf{w}}(x_{ij} | u_{ik})} = \frac{P_{\mathbf{w}}(x_{ij}, u_{ik} | D_l)}{w_{ijk}}$

The resulting algorithm is then as follows:

Input: Given a (Bayesian) probabilistic network N with CPT entries $\mathbf{w}$, and $\mathbf{D}$ a set of data cases,

Repeat until $\Delta \mathbf{w} \approx \mathbf{0}$

$\Delta \mathbf{w} \leftarrow \mathbf{0}$

for each $D_l \in \mathbf{D}$

Set the evidence in N from D_l

For each variable i, value j, and conditioning case k

$$\Delta w_{ijk} \leftarrow \Delta w_{ijk} + \frac{P_{\mathbf{w}}(x_{ij}, u_{ik} | D_l)}{w_{ijk}}$$

$$\mathbf{w} \leftarrow \mathbf{w} + \eta \Delta \mathbf{w}$$

Renormalize the w_{ijk} to assure

$\sum_j w_{ijk} = 1$

$0 \leq w_{ijk} \leq 1$

Learning the Structure of the Bayesian Network

Very difficult (some research is being done):

- Greedy search among network structures on a cost function that trades off structure complexity for accuracy;
- Constraint-based approaches.

More on Learning Bayes Nets: The EM algorithm can also be used

Repeatedly

1. Calculate probabilities of unobserved variables, assuming h
2. Calculate new w_{ijk} to maximize $E[\ln P(D|h)]$ where D now includes both observed and (calculated probabilities of) unobserved variables

When the structure is unknown...

- Algorithms use greedy search to add/substract edges and nodes
- Active research topic

Summary: Bayesian Belief Networks

- Combine prior knowledge with observed data
- Impact of prior knowledge (when correct!) is to lower the sample complexity
- Active research area
  - Extend from boolean to real-valued variables
  - Parameterized distributions instead of tables
  - Extend to first-order instead of propositional systems
  - More effective inference methods
  - ...

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

Generating Data from Mixture of k Gaussians

Each instance x generated by

1. Choosing one of the k Gaussians with uniform probability
2. Generating an instance at random according to that Gaussian

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 jth 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

E 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.

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

M step: Calculate a new maximum likelihood hypothesis $h' = \langle \mu'_1, \mu'_2 \rangle$. In this case the ML hypothesis is given by

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

The above equation is similar to the equation for the sample mean

$$\mu_{ML} = \frac{1}{m}\sum_{i=1}^{m} x_i,$$

which is the solution of

$$\mu_{ML} = \arg\min_{\mu} \sum_{i=1}^{m} (x_i - \mu)^2,$$

which is used to estimate μ for a single Normal (Gaussian) distribution.

So, the new expression is just the weighted sample mean for μ_j with each instance weighted by the expectation $E[z_{ij}]$ that it was generated by the jth Normal distribution.

Thus **Essence of the EM approach:**

- The current hypothesis is used to estimate the unobserved variables, and the expected values of these variables are then used to calculate an improved hypothesis.
- The algorithm thus converges to a local maximum likelihood hypothesis for $\langle \mu_1, \mu_2 \rangle$.

General Statement of EM Algorithm

the EM algorithm can be applied in many settings where we wish to estimate some set of parameters θ that describe an underlying probability distribution, given only the observed portion of the full data produced by this distribution.

In the two-means example the parameters of interest were $\theta = \langle \mu_1, \mu_2 \rangle$, and the full data were the triples $\langle x_i, z_{i1}, z_{i2} \rangle$ of which only the x_i were observed.

In general

$$X = \{x_1, ..., x_m\}$$

denotes the observed data in a set of m independently drawn instances

$$Z = \{z_1, ..., z_m\}$$

denotes the unobserved data in these same instances,

$$Y = X \cup Z$$

denotes the full data.

Treat Z as a random variable whose probability distribution depends on the unknown parameters θ and on the observed data X.

Similarly, Y is a random variable because it is defined in terms of the random variable Z.

Use h to denote the current hypothesized values of the parameters θ, and h' to denote the revised hypothesis that is estimated on each iteration of the EM algorithm.

The EM algorithm searches for the *maximum likelihood hypothesis* h' such that

$$h = \underset{h'}{\operatorname{argmax}} E[\ln P(Y|h')]$$

This expected value is taken over the probability distribution of Y , which is determined by the unknown parameters θ.

Meaning of $E[\ln P(Y|h')]$

- $P(Y|h')$ is the likelihood of the full data Y given hypothesis h'.
- We wish to find a h'that maximizes some function of this quantity.
- Maximizing of $InP(Y|h')$ also maximizes $P(Y|h')$,
- Introduce $E[lnP(Y|h')]$ because the full data Y is itself a random variable.
- Given that the full data Y is a combination of the observed data X and unobserved data Z, we must average over the possible values of the unobserved Z, weighting each according to its probability.

- We take the expected value $E[\ln P(Y|h')]$ over the probability distribution governing the random variable Y
- Y distribution: completely known values for X + distribution governing Z.
- QUESTION: What is the probability distribution governing Y ?
- Quick answer: In general, not known: it is determined by the parameters θ that we are trying to estimate.
- Use the current hypothesis h instead of θ to estimate the distribution for Y.

Define

$$Q(h|h') = E[lnP(Y|h')|h, X]$$

The notation $Q(h|h')$ indicates that Q is defined by the assumption that the current hypothesis, h, is equal to θ.

Estimation Calculate $Q(h'lh)$ using the current hypothesis h and the observed data X to estimate the probability distribution over Y .

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

Maximization: Replace hypothesis h by the hypothesis h' that maximizes this Q function.

$$h = \underset{h'}{\operatorname{argmax}} E[lnP(Y|h')]$$

- When the function Q is continuous, the EM algorithm converges to a stationary point of the likelihood function $P(Y|h')$.
- When $P(Y|h')$ has a single maximum, EM will converge to this global maximum likelihood estimate for h'.
- Otherwise, it is guaranteed only to converge to a local maximum. (Similar limitations as other optimization methods such as gradient descent)

The k Means Algorithm

Problem: estimate the means of a mixture of k Normal distributions.

That is, the k-means problem is to estimate the parameters $\theta = \langle \mu_1, \ldots, \mu_k \rangle$ that define the means of the k Normal distributions.

We are given the observed data $X = \{x_1, \ldots x_m\}$;

The hidden variables $Z = \{\langle z_{i1}, \ldots, z_{ik} \rangle\}$ indicate which of the k Normal distributions was used to generate x_i.

Must derive an expression for $Q(h|h')$ that applies to our k-means problem.

Derive an expression for $lnp(Y|h'))$; Note:

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

Only one of the z_{ij} can have the value 1, and all others must be 0.

Therefore, this expression gives the probability distribution for x_i generated by the selected Normal distribution.

Given this probability for a single instance $p(y_i|h')$ the logarithm of the probability $\ln P(Y|h')$ for all m instances in the data is

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

Taking the expectation we obtain

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

Thus

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

and

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

Derivation of the k-means Algorithm

Estimate the means of a mixture of k Normal Distributions: estimate the parameter $\theta = \langle \mu_1, \ldots, \mu_k \rangle$

We are given the observations $X = \{\angle x_i\rangle\}$. The hidden variables are $Z = \{\langle z_{i1}, \ldots, z_{ik}\rangle$, indicating which of the k Normal distributions have generated data x_i.

Derive first an expression for $\ln p(Y|h')$. For the single instance $y_i = \langle x_i, z_{i1}, \ldots, z_{ik}\rangle$, we have

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

Then, for Y, we obtain:

$$P(Y|h') = \prod_{i=1}^{m} p(y_i|h')$$

and therefore,

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

Note that $\ln P(Y|h')$ is a linear function in z_{ij}. Therefore,

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

Therefore,

$$Q(h|h') = \sum_{i=1}^{m} \left(\ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{\sigma^2} \sum E[z_{ij}](x_i - \mu'_j)^2 \right)$$

where $h' = \langle \mu'_1, \ldots, \mu'_k \rangle$ and $E[z_{ij}]$ is calculated as

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

Thus we have the following two steps:

1. E Step: Compute $E[z_{ij}]$ according to the above equation
2. M Step: Find h', which maximizes $Q(h'|h)$, that is solve

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

which is

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