CPSC 540: Machine Learning

Expectation Maximization

Mark Schmidt

University of British Columbia

Winter 2020

Last Time: Learning with MAR Values

- We discussed learning with "missing at random" values in data:

$$X = \begin{bmatrix} 1.33 & 0.45 & -0.05 & -1.08 & ? \\ 1.49 & 2.36 & -1.29 & -0.80 & ? \\ -0.35 & -1.38 & -2.89 & -0.10 & ? \\ 0.10 & -1.29 & 0.64 & -0.46 & ? \\ 0.79 & 0.25 & -0.47 & -0.18 & ? \\ 2.93 & -1.56 & -1.11 & -0.81 & ? \\ -1.15 & 0.22 & -0.11 & -0.25 & ? \end{bmatrix}$$

- Imputation approach:
  - Guess the most likely value of each ?, fit model with these values (and repeat).

- K-means clustering algorithm is a special case:
  - Gaussian mixture ($\pi_c = 1/k$, $\Sigma_c = I$) and ? being the cluster (? $\in \{1, 2, \cdots, k\}$).

Parameters, Hyper-Parameters, and Nuisance Parameters

- Are the ? values "parameters" or "hyper-parameters"?
- Parameters:
  - Variables in our model that we optimize based on the training set.
- Hyper-Parameters
  - Variables that control model complexity, typically set using validation set.
  - Often become degenerate if we set these based on training data.
  - We sometimes add optimization parameters in here like step-size.
- Nuisance Parameters
  - Not part of the model and not really controlling complexity.
  - An alternative to optimizing ("imputation") is to consider all values.
    - Based on marginalization rule for probabilities.
    - Consider all possible imputations, and weight them by their probability.

Expectation Maximization Notation

- Expectation maximization (EM) is an optimization algorithm for MAR values:
  - Applies to problems that are easy to solve with "complete" data (i.e., you knew ?).
  - Allows probabilistic or "soft" assignments to MAR (or other nuisance) variables.
    - Imputation approach is sometimes called "hard" EM.

- EM is among the most cited paper in statistics.

- EM notation: we use O as observed data and H as hidden (?) data.
  - Semi-supervised learning: observe $O = \{X, y, \bar{X}\}$ but don't observe $H = \{\bar{y}\}$.
  - Mixture models: observe data $O = \{X\}$ but don't observe clusters $H = \{z^i\}_{i=1}^n$.

- We use Θ as parameters we want to optimize.
  - In Gaussian mixtures this will be the π_c, μ_c, and Σ_c variables.

The Two Likelihoods: "Complete" and "Marginal"

- "Complete" likelihood: likelihood with known hidden values, $p(O, H \mid \Theta)$.
  - We assume that this is "nice". Maybe it has a closed-form MLE or is convex.
- "Marginal" likelihood: likelihood with unknown hidden values, $p(O \mid \Theta)$.
  - This is our usual likelihood, the thing we actually want to optimize.
- The "complete" and "marginal" likelihoods are related by the marginalization rule:

$$\underbrace{p(O \mid \Theta)}_{\text{"marginal"}} = \sum_{H_1} \sum_{H_2} \cdots \sum_{H_m} p(O, H \mid \Theta) = \sum_{H} \underbrace{p(O, H \mid \Theta)}_{\text{"complete likelihood"}}.$$

where we sum over all possible $H \equiv \{H_1, H_2, \dots, H_m\}$.
  - For mixture models, this sums over all possible clusterings (k^n values).
  - Replace the sums by integrals for continuous hidden values.

Expectation Maximization Bound

- The negative log-likelihood (that we want to optimize) thus has the form

$$-\log p(O \mid \Theta) = -\log\left(\sum_H p(O, H \mid \Theta)\right),$$

- which has a sum inside the log.
  - This does not preserve convexity: minimizing it is usually NP-hard.

- Both EM and imputation are based on the approximation:

$$-\log\left(\sum_H p(O, H \mid \Theta)\right) \approx -\sum_H \alpha_H \log p(O, H \mid \Theta)$$

where α_H is some probability for the assignment H to the hidden variables.
  - An expectation over "complete" log-likelihood.
  - This is useful when the approximation is easier to minimize.

Expectation Maximization Bound

- Each iteration of EM and imputation optimize the approximation:

$$\Theta^{t+1} \in \operatorname*{argmin}_{\Theta} -\sum_{H} \alpha_H^t \log p(O, H \mid \Theta).$$

where the probabilities α_H^t are updated after each iteration t.

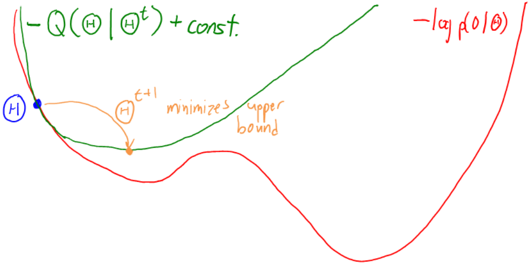
- Imputation sets $\alpha_H^t = 1$ for the most likely H given Θ^t (all other $\alpha_H^t = 0$).
  - It assumes that the imputations are correct, then optimizes with the guess
- In EM we set $\alpha_H^t = p(H \mid O, \Theta^t)$, weighting H by probability given Θ^t.
  - It weighs different imputations by their probability, then optimizes.

Expectation Maximization as Bound Optimization

- We'll show that the EM approximation minimizes an upper bound,

$$-\underbrace{\log p(O \mid \Theta)}_{\text{what we want}} \leq \underbrace{-\sum_H p(H \mid O, \Theta^t) \log p(O, H \mid \Theta)}_{Q(\Theta \mid \Theta^t):\ \text{what we optimize}} + \text{const.},$$

- Geometry of expectation maximization as "bound optimization":
  - At each iteration t we optimize a bound on the function.



Expectation Maximization (EM)

- So EM starts with Θ^0 and sets Θ^{t+1} to maximize $Q(\Theta \mid \Theta^t)$.
- This is typically written as two steps:
  1. E-step: Define expectation of complete log-likelihood given last parameters Θ^t,
  $$Q(\Theta \mid \Theta^t) = \sum_H \underbrace{p(H \mid O, \Theta^t)}_{\text{fixed weights } \alpha_H^t} \underbrace{\log p(O, H \mid \Theta)}_{\text{nice term}}$$
  $$= \mathbb{E}_{H \mid O, \Theta^t}[\log p(O, H \mid \Theta)],$$
  which is a weighted version of the "nice" $\log p(O, H)$ values.
  2. M-step: Maximize this expectation to generate new parameters Θ^{t+1},
  $$\Theta^{t+1} = \underset{\Theta}{\text{argmax}}\, Q(\Theta \mid \Theta^t).$$

Expectation Maximization for Mixture Models

- In the case of a mixture model with extra "cluster" variables z^i EM uses

$$
\begin{aligned}
Q(\Theta \mid \Theta^t) &= \mathbb{E}_{z \mid X,\Theta}[\log p(X, z \mid \Theta)] \\
&= \sum_{z^1=1}^{k} \sum_{z^2=1}^{k} \cdots \sum_{z^n=1}^{k} \underbrace{p(z \mid X, \Theta^t)}_{\alpha_z} \underbrace{\log p(X, z \mid \Theta)}_{\text{“nice”}} \\
&= \sum_{z^1=1}^{k} \sum_{z^2=1}^{k} \cdots \sum_{z^n=1}^{k} \left(\prod_{i=1}^{n} p(z^i \mid x^i, \Theta^t)\right) \left(\sum_{i=1}^{n} \log p(x^i, z^i \mid \Theta)\right) \\
&= \text{(see EM notes, tedious use of distributive law and independences)} \\
&= \sum_{i=1}^{n} \sum_{z^i=1}^{k} p(z^i \mid x^i, \Theta^t) \log p(x^i, z^i \mid \Theta).
\end{aligned}
$$

- Sum over k^n clusterings turns into sum over nk 1-example assignments.
  - Same simplification happens for semi-supervised learning, we'll discuss why later.

Expectation Maximization for Mixture Models

- In the case of a mixture model with extra “cluster” variables z^i EM uses

$$Q(\Theta \mid \Theta^t) = \sum_{i=1}^{n} \sum_{z^i=1}^{k} \underbrace{p(z^i \mid x^i, \Theta^t)}_{r_c^i} \log p(x^i, z^i \mid \Theta).$$

- This is just a weighted version of the usual likelihood.
  - We just need to do MLE in weighted Gaussian, weighted Bernoulli, etc.

- We typically write update in terms of responsibilitites (easy to calculate),

$$r_c^i \triangleq p(z^i = c \mid x^i, \Theta^t) = \frac{p(x^i \mid z^i = c, \Theta^t) p(z^i = c \mid \Theta^t)}{\sum_{c'=1}^{k} p(x^i \mid z^i = c', \Theta^t) p(z^i = c' \mid \Theta^t)} \quad \text{(Bayes rule)},$$

the probability that cluster c generated x^i.
  - In k-means $r_c^i = 1$ for most likely cluster and 0 otherwise.
  - You may get underflow when computing r_c^i (see bonus for log-domain tricks).

Expectation Maximization for Mixture of Gaussians

- For mixture of Gaussians, E-step computes all r_c^i and M-step minimizes the weighted NLL:

$$\pi_c^{t+1} = \frac{1}{n}\sum_{i=1}^{n} r_c^i \qquad \text{(proportion of examples soft-assigned to cluster } c\text{)}$$

$$\mu_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i}\sum_{i=1}^{n} r_c^i x^i \qquad \text{(mean of examples soft-assigned to cluster } c\text{)}$$

$$\Sigma_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i}\sum_{i=1}^{n} r_c^i (x^i - \mu_c^{t+1})(x^i - \mu_c^{t+1})^\top \qquad \text{(covariance of examples soft-assigned to } c\text{).}$$

- Now you would compute new responsibilities and repeat.
  - Notice that there is no step-size.

- EM for fitting mixture of Gaussians in action:
  `https://www.youtube.com/watch?v=B36fzChfyGU`

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn't reach global optimum.
  - Classic solution: restart the algorithm from different initializations.
  - Lots of work in CS theory on getting better initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
  - Use MAP estimates or remove these clusters.
- EM does not fix "propagation of errors" from imputation approach.
  - But it reduces problem by incorporating a "confidence" over different imputations.
- Can you make it robust?
  - Use mixture of Laplace of student t distributions.
  - Don't have closed-form EM steps: compute responsibilities then need to optimize.

Outline

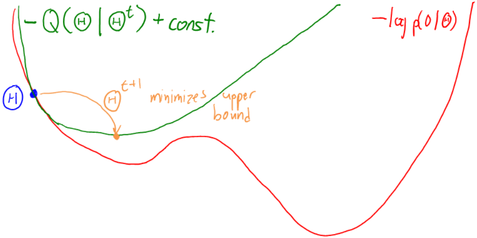
Monotonicity of EM

- Classic result is that EM iterations are monotonic:

$$\log p(O \mid \Theta^{t+1}) \geq \log p(O \mid \Theta^{t}),$$

- We don't need a step-size and this is useful for debugging.

- We can show this by proving that the below picture is "correct":



- The Q function leads to a global bound on the original function.
- At Θ^t the bound matches original function.
  - So if you improve on the Q function, you improve on the original function.

Monotonicity of EM

- Let's show that the Q function gives a global upper bound on NLL:

$$
\begin{aligned}
-\log p(O \mid \Theta) &= -\log\left(\sum_H p(O, H \mid \Theta)\right) && \text{(marginalization rule)} \\
&= -\log\left(\sum_H \alpha_H \frac{p(O, H \mid \Theta)}{\alpha_H}\right) && \text{(for } \alpha_H \neq 0\text{)} \\
&\leq -\sum_H \alpha_H \log\left(\frac{p(O, H \mid \Theta)}{\alpha_H}\right),
\end{aligned}
$$

because $-\log(z)$ is convex and the α_H are a convex combination.

Monotonicity of EM

- Using that log turns multiplication into addition we get

$$
\begin{aligned}
-\log p(O \mid \Theta) &\leq -\sum_H \alpha_H \log\left(\frac{p(O, H \mid \Theta)}{\alpha_H}\right) \\
&= \underbrace{-\sum_H \alpha_H \log p(O, H \mid \Theta)}_{Q(\Theta \mid \Theta^t)} + \underbrace{\sum_H \alpha_H \log \alpha_H}_{\text{negative entropy}} \\
&= -Q(\Theta \mid \Theta^t) - \text{entropy}(\alpha),
\end{aligned}
$$

  so we have the first part of the picture, $-\log p(O \mid \Theta^{t+1}) \leq -Q(\Theta|\Theta^t) + \text{const}$.
  - Entropy is a measure of how "random" the α_H values are.
  - Q behaves more like true objective for H that are more "predictable".

- Now we need to show that this holds with equality at Θ^t.

Bound on Progress of Expectation Maximization

- To show equality at Θ^t we use definition of conditional probability,

$$p(H \mid O, \Theta^t) = \frac{p(O, H \mid \Theta^t)}{p(O \mid \Theta^t)} \quad \text{or} \quad \log p(O \mid \Theta^t) = \log p(O, H \mid \Theta^t) - \log p(H \mid O, \Theta^t)$$

- Multiply by α_H and summing over H values,

$$\sum_H \alpha_H \log p(O \mid \Theta^t) = \underbrace{\sum_H \alpha_H \log p(O, H \mid \Theta^t}_{Q(\Theta^t \mid \Theta^t)} - \sum_H \alpha_H \log \underbrace{p(H \mid O, \Theta^t)}_{\alpha_H}.$$

- Which gives the result we want:

$$\log p(O \mid \Theta^t) \underbrace{\sum_H \alpha_H}_{=1} = Q(\Theta^t \mid \Theta^t) + \text{entropy}(\alpha),$$

Bound on Progress of Expectation Maximization

- Thus we have the two bounds

$$\log p(O \mid \Theta) \geq Q(\Theta \mid \Theta^t) + \text{entropy}(\alpha)$$
$$\log p(O \mid \Theta^t) = Q(\Theta^t \mid \Theta^t) + \text{entropy}(\alpha).$$

- Subtracting these and using $\Theta = \Theta^{t+1}$ gives a stronger result,

$$\log p(O \mid \Theta^{t+1}) - \log p(O \mid \Theta^t) \geq Q(\Theta^{t+1} \mid \Theta^t) - Q(\Theta^t \mid \Theta^t),$$

  that we improve objective by at least the decrease in Q.

- Inequality holds for any choice of Θ^{t+1}.
  - Approximate M-steps are ok: we just need to decrease Q to improve likelihood.

- For imputation, we instead improve "complete" log-likelihood, $\log p(O, H \mid \Theta^t)$.
  - Which isn't quite what we want, treats hidden data as a "parameter".

Summary

- Expectation maximization:
  - Optimization with MAR variables, when knowing MAR variables make problem easy.
  - Instead of imputation, works with "soft" assignments to nuisance variables.
  - Maximizes log-likelihood, weighted by all imputations of hidden variables.
- Monotonicity of EM: EM is guaranteed not to decrease likelihood.
- Next time: generalizing histograms?

EM Alternatives

- Are there alternatives to EM?
  - Could use gradient descent, SGD, and so on.
  - Many variations on EM to speed up its convergence (for example, "adaptive" bound optimization).
  - Spectral and other recent methods have some global guarantees.

Avoiding Underflow when Computing Responsibilities

- Computing responsibility may underflow for high-dimensional x^i, due to $p(x^i \mid z^i = c, \Theta^t)$.
- Usual ML solution: do all but last step in log-domain.

$$\log r_c^i = \log p(x^i \mid z^i = c, \Theta^t) + \log p(z^i = c \mid \Theta^t) \\ - \log\left(\sum_{c'=1}^{k} p(x^i \mid z^i = c', \Theta^t) p(z^i = c' \mid \Theta^t)\right).$$

- To compute last term, use "log-sum-exp" trick.

Log-Sum-Exp Trick

- To compute $\log(\sum_i \exp(v_i))$, set $\beta = \max_i\{v_i\}$ and use:

$$\begin{aligned}\log(\sum_c \exp(v_i)) &= \log(\sum_i \exp(v_i - \beta + \beta)) \\ &= \log(\sum_i \exp(v_i - \beta)\exp(\beta)) \\ &= \log(\exp(\beta))\sum_i \exp(v_i - \beta)) \\ &= \log(\exp(\beta)) + \log(\sum_i \exp(v_i - \beta)) \\ &= \beta + \log(\sum_i \underbrace{\exp(v_i - \beta)}_{\leq 1}).\end{aligned}$$

- Avoids overflows due to computing $\exp$ operator.

Alternate View of EM as BCD

- We showed that given α the M-step minimizes in Θ the function

$$F(\Theta, \alpha) = -\mathbb{E}_\alpha[\log p(O, H \mid \Theta)] - \text{entropy}(\alpha).$$

- The E-step minimizes this function in terms of α given Θ.
  - Setting $\alpha_H = p(H \mid O, \Theta)$ minimizes it.

- Note that F is not the NLL, but F and the NLL have same stationary points.

- From this perspective, we can view EM as a block coordinate descent method.

- This perspective is also useful if you want to do approximate E-steps.

Alternate View of EM as KL-Proximal

- Using definitions of expectation and entropy and α in the last slide gives

$$
\begin{aligned}
F(\Theta, \alpha) &= -\sum_H p(H \mid O, \theta^t) \log p(O, H \mid \Theta) + \sum_H p(H \mid O, \theta^t) \log p(H \mid O, \theta^t) \\
&= -\sum_H p(H \mid O, \theta^t) \log \frac{p(O, H \mid \theta)}{p(H \mid O, \theta^t)} \\
&= -\sum_H p(H \mid O, \theta^t) \log \frac{p(H \mid O, \theta) p(O \mid \theta)}{p(H \mid O, \theta^t)} \\
&= -\sum_H \log p(O \mid \Theta) - \sum_H p(H \mid O, \theta^t) \log \frac{p(H \mid O, \theta)}{p(H \mid O, \theta^t)} \\
&= NLL(\Theta) + \mathsf{KL}(p(H \mid O, \theta^t) \mid\mid p(H \mid O, \theta)).
\end{aligned}
$$

- From this perspective, we can view EM as a "proximal point" method.
  - Classical proximal point method uses $\frac{1}{2}\|\theta^t - \theta\|^2$, EM uses KL divergence.
- From this view we can see that EM doesn't depend on parameterization of Θ.
- If we linearize NLL and we multiply KL term by $1/\alpha_k$ (step-size), we get the natural gradient method.