CPSC 440: Advanced Machine Learning

Expectation Maximization

Mark Schmidt

University of British Columbia

Winter 2021

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 most likely value of each ?, fit model with these values (and maybe repeat).

- Semi-supervised learning: supervised learning with labeled and unlabeled data.
  - Imputation method is a form of "self-taught" learning.

Back to Mixture Models

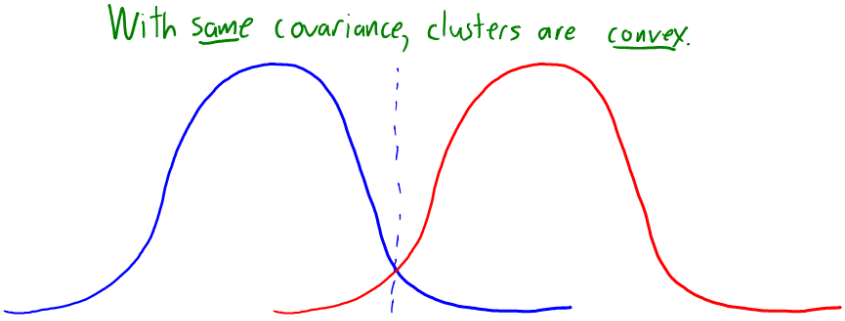
- To fit mixture models we often make n MAR new variables z^i.
- Why???
- Consider mixture of Gaussians, and let z^i be the cluster number of example i:
  - So $z^i \in \{1, 2, \cdots, k\}$ tells you which Gaussian generated example i.
  - Given the z^i it's easy to optimize the parameters of the mixture model.
    - Solve for $\{\pi_c, \mu_c, \Sigma_c\}$ maximizing $p(x^i, z^i)$ (learning step in GDA).
  - Given $\{\pi_c, \mu_c, \Sigma_c\}$ it's easy to optimize the clusters z^i:
    - Find the cluster c maximizing $p(x^i, z_i = c)$ (prediction step in GDA).

Imputation Approach for Mixtures of Gaussians

- Consider mixture of Gaussians with the choice $\pi_c = 1/k$ and $\Sigma_c = I$ for all c.
- Here is the imputation approach for fitting a mixtures of Gaussian:
  - Randomly pick some initial means μ_c.
  - Assigns x^i to the closest mean (imputation of missing z^i values).
    - This is how you maximize $p(x^i, z^i)$ in terms of z^i.
  - Set μ_c to the mean of the points assigned to cluster c (parameter update).
    - This is how you maximize $p(x^i, z^i)$ in terms of μ_c given the z^i.
- This is exactly k-means clustering.

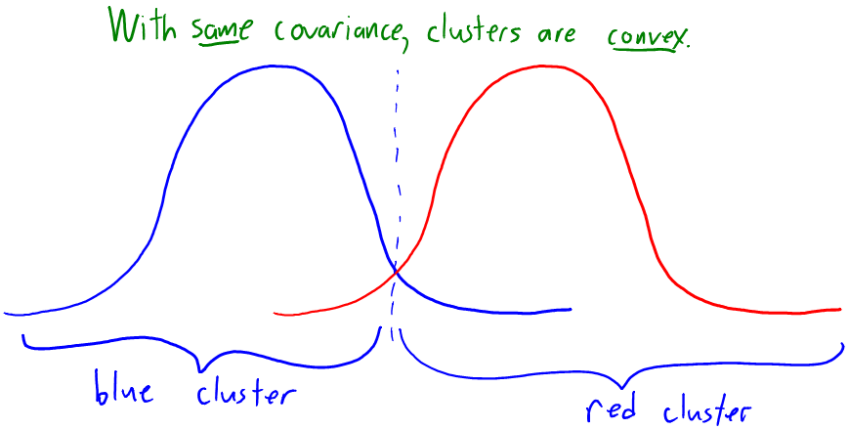
K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



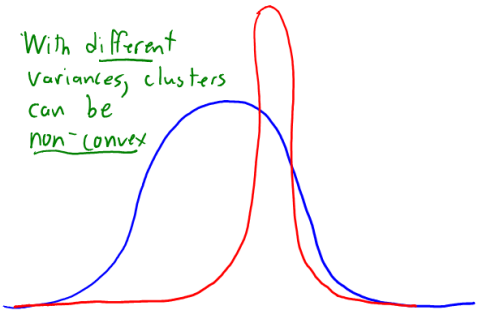
K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



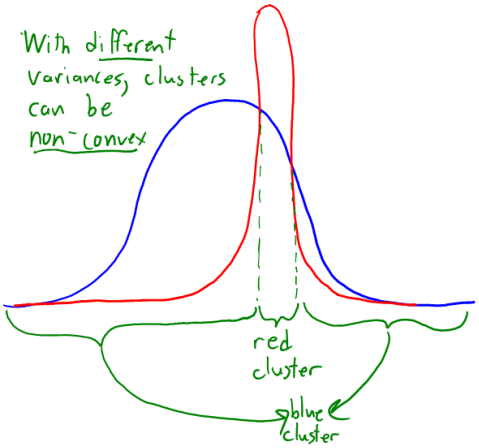
K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



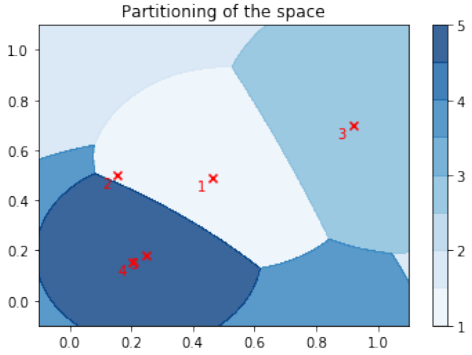
K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



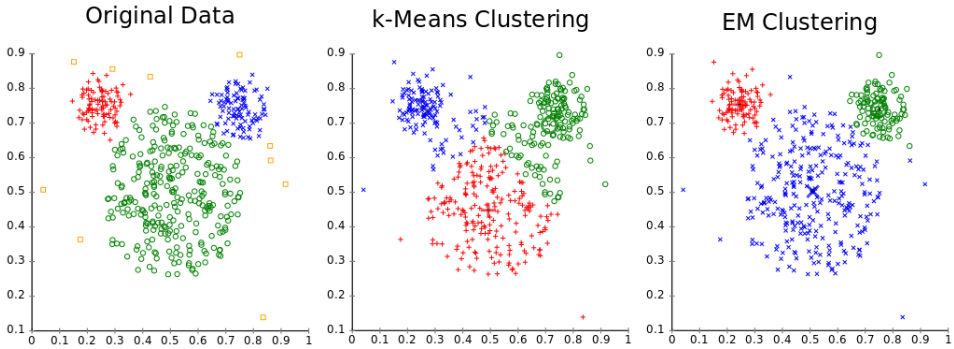
K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



K-Means vs. Mixture of Gaussians

- K-means can be viewed as fitting mixture of Gaussians (common Σ_c).
  - But variable Σ_c in mixture of Gaussians allow non-convex clusters.



https://en.wikipedia.org/wiki/K-means_clustering

Outline

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.
    - Because "optimizing worse" can decrease overfitting.

- 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.

Drawbacks of Imputation Approach

- Imputation approach to MAR variables treats the ? as parameters.
  - Use density estimator to find the "best" way to "fill in" the missing values.
  - Now fit the "complete data" using a standard method.

- But "hard" assignments of missing values leads to propagation of errors.
  - What if cluster is ambiguous in k-means clustering?
  - What if label is ambiguous in "self-taught" learning?

- Expectation maximization (EM) treats the ? as nuissance parameters:
  - Use probabilities of different assignments ("soft" assignments) instead of imputation.
  - If the MAR values are obvious, this will act like the imputation approach.
  - For ambiguous MAR values, takes into account our uncertainty.

Example: Expectation Maximization for Mixture of Gaussians

- EM for mixtures needs "probability that example i comes from mixture c".
  - We call this the responsibility r_c^i of cluster c for example i.
    - EM computes these given parameters Θ^t at iteration t.

- You can compute the responsibilities with Bayes rule:

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

  which depends on the mixture proportions $p(z^i = c \mid \Theta^t)$ and the likelihood.
  - Though you may get underflow when computing r_c^i (see bonus for log-domain tricks).

- Can think of imputation/k-means as using $r_c^i = 1$ for most likely cluster.
  - And $r_c^i = 0$ for all other clusters.
- EM insteads "soft-assign" each training example to each cluster.
  - Example i could be 40% in cluster 1, 35% in cluster 2, and 25% in cluster 3.

Example: Expectation Maximization for Mixture of Gaussians

- The EM algorithm for training mixtures of Gaussans repeats the following steps:
  1. Computer probability that example i is in cluster c based on parameters Θ^t.
  $$r_c^i = p(z^i = c \mid x^i, \Theta^t).$$
  2. Update cluster probabilities based on number of examples soft-assigned to clusters.
  $$\pi_c^{t+1} = \frac{1}{n} \sum_{i=1}^{n} r_c^i.$$
  3. Update means based on means of examples soft-assigned to each cluster.
  $$\mu_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i} \sum_{i=1}^{n} r_c^i x^i.$$
  4. Update covariances based on covariances of examples soft-assigned to each cluster.
  $$\Sigma_c^{t+1} = \frac{1}{\sum_{i=1}^{n} r_c^i} \sum_{i=1}^{n} r_c^i (x^i - \mu_c^{t+1})(x^i - \mu_c^{t+1})^\top.$$
- Video: `https://www.youtube.com/watch?v=B36fzChfyGU`

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.
  - Special cases independently discovered in 50s-70s, general case in 1977.

- 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 imputed 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 $H \equiv \{H_1, H_2, \ldots, H_m\}$.

- 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.
    - The specific α_H chosen by EM makes the approximation tight at Θ^t.

Summary

- Introducing MAR variables to fit mixture models.
  - Create new variables that represent "which cluster this example came from".
  - K-means is a special case of imputation approach to MAR.
- 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.
  - Simple and intuitive updates for fitting standard mixtures models.
- Next time: properties of EM (I'm not going to pretend this is exciting).

EM Alternatives

- Are there alternatives to for optimizing marginal likelihood EM?
  - Could use gradient descent, SGD, and so on.
    - We now know that EM converges faster than gradient descent for standard mixture models.
  - 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.