CAP 6610 Machine Learning, Spring 2022

Homework 4 Solution

1. (15%) *Alternating optimization for PCA.* Recall that the principal component analysis (PCA) tries to find the projection matrix $\boldsymbol{\Theta}$ and embeddings $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ as the solution of

$$\underset{\boldsymbol{\Theta}, \boldsymbol{y}_i}{\text{minimize}} \quad \sum_{i=1}^{n} \|\boldsymbol{\phi}_i - \boldsymbol{\Theta}\boldsymbol{y}_i\|^2.$$

We can define matrices $\boldsymbol{\Phi} = [\ \boldsymbol{\phi}_1\ \boldsymbol{\phi}_2\ \cdots\ \boldsymbol{\phi}_n\ ]$ and $\boldsymbol{Y} = [\ \boldsymbol{y}_1\ \boldsymbol{y}_2\ \cdots\ \boldsymbol{y}_n\ ]$ and rewrite the problem as

$$\underset{\boldsymbol{\Theta}, \boldsymbol{Y}}{\text{minimize}} \quad \|\boldsymbol{\Phi} - \boldsymbol{\Theta}\boldsymbol{Y}\|^2. \tag{1}$$

Without imposing the constraint $\boldsymbol{\Theta}^\top\boldsymbol{\Theta} = \boldsymbol{I}$, derive an alternating optimization algorithm for (1).

Solution. Fixing $\boldsymbol{\Theta}$, minimizing $\|\boldsymbol{\phi}_i - \boldsymbol{\Theta}\boldsymbol{y}_i\|^2$ with respect to $\boldsymbol{y}_i$ is a least squares problem with closed-form solution $(\boldsymbol{\Theta}^\top\boldsymbol{\Theta})^{-1}\boldsymbol{\Theta}^\top\boldsymbol{\phi}_i$. Stacking them together for $i = 1, \ldots, n$ as columns, we get

$$\boldsymbol{Y} \leftarrow (\boldsymbol{\Theta}^\top\boldsymbol{\Theta})^{-1}\boldsymbol{\Theta}^\top\boldsymbol{\Phi}. \tag{2}$$

Now if we fix $\boldsymbol{Y}$, the problem is still a least squares problem if we transpose everything, thus

$$\boldsymbol{\Theta} \leftarrow \boldsymbol{\Phi}\boldsymbol{Y}^\top(\boldsymbol{Y}\boldsymbol{Y}^\top)^{-1}. \tag{3}$$

The resulting alternating optimization algorithm alternates between (2) and (3).

2. (15%) *Naive Bayes GMM.* Consider a Gaussian mixture model in which the marginal distribution of the latent variable $\boldsymbol{y}$ is $\Pr(\boldsymbol{y} = \boldsymbol{e}_c) = \pi_c, c = 1, \ldots, k$, and the conditional distribution of $\boldsymbol{x}$ given $\boldsymbol{y}$ is $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ where each of the covariance matrices $\boldsymbol{\Sigma}_c$ is diagonal. This means given the latent variable $\boldsymbol{y}$, elements of the ambient variable $\boldsymbol{x}$ are conditionally independent, resembling the naive Bayes model. Given a set of observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, derive the expectation-maximization algorithm for estimating the model parameters $\pi_c, \boldsymbol{\mu}_c$, and $\boldsymbol{\Sigma}_c$ for $c = 1, \ldots, k$.

Solution. Let's start by fixing the model parameters $\pi_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c, c = 1, \ldots, k$, and compute the conditional mean $\mathrm{E}[\boldsymbol{y}_i|\boldsymbol{x}_i]$

$$\mathrm{E}[y_{ic}|\boldsymbol{x}_i] = \frac{\pi_c \det(2\pi\boldsymbol{\Sigma}_c)^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_c)^\top\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_c)\right)}{\sum_j \pi_j \det(2\pi\boldsymbol{\Sigma}_j)^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_j)^\top\boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_j)\right)}. \tag{4}$$

Next we derive how to estimate the parameters by replacing $\boldsymbol{y}_i$ with $\boldsymbol{\psi}_i$ in $p(\boldsymbol{x}_i, \boldsymbol{y}_i; \boldsymbol{\theta})$:

$$\sum_{i=1}^{n} \mathrm{E}[-\log p(\boldsymbol{x}_i, \boldsymbol{y}_i)] = \sum_{i=1}^{n}\sum_{c=1}^{k} \psi_{ic}\left(\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^\top\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_c) + \frac{1}{2}\log\det(2\pi\boldsymbol{\Sigma}_c) - \log(\pi_c)\right).$$

By taking the gradient equal to zero, we get

$$\begin{aligned} \pi_c &= \frac{\sum_{i=1}^n \psi_{ic}}{n}, \qquad \boldsymbol{\mu}_c = \frac{1}{\sum_{i=1}^n \psi_{ic}} \sum_{i=1}^{n} \psi_{ic}\boldsymbol{x}_i, \\ \boldsymbol{\Sigma}_c &= \frac{1}{n}\operatorname{Diag}\left(\sum_{i=1}^{n} \psi_{ic}(\boldsymbol{x}_i - \boldsymbol{\mu}_c)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^\top\right), \qquad c = 1, \ldots, k. \end{aligned} \tag{5}$$

The resulting EM algorithm alternates between (4) and (5). We see that the only difference between this and the general EM algorithm for GMM is that each $\boldsymbol{\Sigma}_c$ matrix takes the diagonal of the empirical sample covariance.

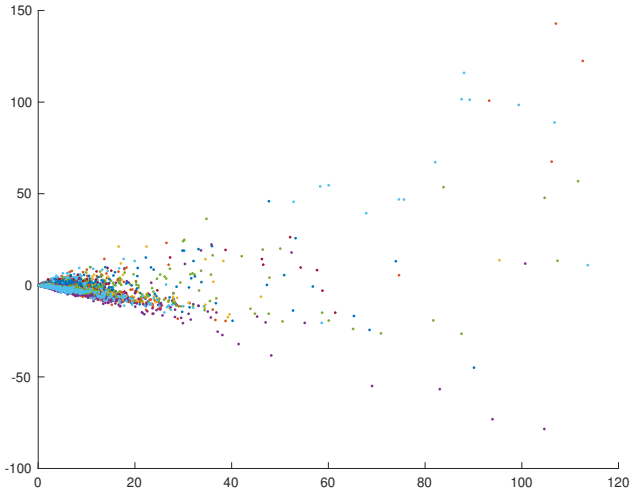
3. (40%) *20 Newsgroup revisited.* Let us revisit the 20 Newsgroup data set `<http://qwone.com/~jason/20Newsgroups/>`, and apply some of the unsupervised methods by ignoring their labels. We will only consider the training data. You are required to code the algorithms by yourselves in the language of your choice.

   (a) *LSI/PCA via orthogonal iteration.* Implement the orthogonal iteration algorithm that finds the PCA projection matrix $\boldsymbol{\Theta}$ of a data matrix $\boldsymbol{\Phi}$. You are allowed to use a pre-existing function of QR. Apply tf-idf to the term-document matrix to obtain $\boldsymbol{\Phi}$ and feed it into your orthogonal iteration algorithm. Remember to use sparse matrix operations to avoid unneccessary memory/computational complexities. Set $k = 2$ and let the algorithm run until $\boldsymbol{\Theta}$ doesn't change much. Then get $\boldsymbol{Y} = \boldsymbol{\Theta}^\top \boldsymbol{\Phi}$. Each column of $\boldsymbol{Y}$ is a two-dimensional vector that you can plot on a plain. Plot all the documents on a two-dimensional plain, and use a different color for each point that belong to different news groups.

   (b) *GMM via EM.* Implement the EM algorithm for the Gaussian mixture model (with different means and covariances for each Gaussian component). The data matrix $\boldsymbol{\Phi}$ is obtained from LSI with $k_{\text{LSI}} = 100$ using the previous orthogonal iteration algorithm. Run the EM algorithmn for GMM with $k_{\text{GMM}} = 20$ until convergence. For each Gaussian component $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, calculate $\boldsymbol{\Theta}\boldsymbol{\mu}_c$ where $\boldsymbol{\Theta}$ is the PCA projection; the vector $\boldsymbol{\Theta}\boldsymbol{\mu}_c$ should be element-wise nonnegative. For each cluster c, show the 10 terms that have the highest value in $\boldsymbol{\Theta}\boldsymbol{\mu}_c$. The index-term mapping can be found here `<http://qwone.com/~jason/20Newsgroups/vocabulary.txt>`. Does the result make sense?

Solution. A sample MATLAB code is given.

(a) The figure is given below. It looks like PCA embedding does not reveal much clustering structure from data.



(b) The top-10 keywords for all 20 clusters seem to be the same, even though a k-means++ initialization is applied. This shows the limitation of EM and the difficulty of estimating the parameters of GMM.

Here's the top-10 words appearing in all clusters. They are all commonly used words and thus not very informative:

- as
- they
- was
- you
- we
- he
- are
- by
- will
- not

4. (30%) *Mixture of multinomials.* Consider the latent variable model with the following probability distribution:

$$\Pr(y_i = c) = \frac{1}{k}, \quad \Pr(\boldsymbol{x}_i | y_i = c) \sim \text{Multi}(\boldsymbol{p}_c, L_i),$$

meaning that y_i is categorical, with k possible outcomes and equal probability; $(\boldsymbol{x}_i | y_i = c)$ follows a multinomial distribution by drawing from $\boldsymbol{p}_c$ L_i times, i.e.,

$$p(\boldsymbol{x}_i | y_i = c) = \frac{L_i!}{\prod_{j=1}^{d} x_{ij}!} \prod_{j=1}^{d} {p_{cj}}^{x_{ij}}.$$

Given data samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$:

(a) Write out the maximum likelihood formulation for estimating $\boldsymbol{p}_1, ..., \boldsymbol{p}_k$. Simplify the objective function as much as possible.

(b) Derive an expectation-maximization algorithm for approximately solving the aforementioned problem.

(c) Implement this algorithm and try it on the 20 News Group data set with $k = 20$ (on the raw word-count data, without tf-idf preprocessing). Show the top 10 words in each cluster.

Solution.

(a) The likelihood of $\boldsymbol{x}_i$ is

$$\begin{aligned} p(\boldsymbol{x}_i) &= \sum_{c=1}^{k} p(\boldsymbol{x}_i | y_i = c) \Pr(Y_i = c) \\ &= \sum_{c=1}^{k} \frac{L_i!}{\prod_{j=1}^{m} x_{ij}!} \prod_{j=1}^{m} {p_{cj}}^{x_{ij}} \times \frac{1}{k} \\ &= \frac{L_i!}{k \prod_{j=1}^{m} x_{ij}!} \sum_{c=1}^{k} \prod_{j=1}^{m} {p_{cj}}^{x_{ij}}. \end{aligned}$$

Its log-likelihood is

$$\log p(\boldsymbol{x}_i) = \log \sum_{c=1}^{k} \prod_{j=1}^{m} {p_{cj}}^{x_{ij}} + C_i,$$

where $C_i = \log \frac{L_i!}{k \prod_{j=1}^m x_{ij}!}$ is a constant that does not depend on the unknown parameter $\boldsymbol{p}_c$'s. Assuming the data samples are independent, the ML objective is

$$\begin{aligned} \underset{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_k}{\text{maximize}} \quad & \sum_{i=1}^n \log \sum_{c=1}^k \prod_{j=1}^m {p_{cj}}^{x_{ij}} \\ \text{subject to} \quad & \boldsymbol{p}_c \in \Delta^m, \quad c = 1, ..., k, \end{aligned} \tag{6}$$

where Δ^d is the probability simplex defined as

$$\Delta^m = \{\boldsymbol{x} \in \mathbb{R}^m : \boldsymbol{x} \geq 0, \boldsymbol{1}^\top \boldsymbol{x} = 1\}.$$

(b) First we fix the model parameters $\boldsymbol{p}_c, c = 1, \ldots, k$ and compute the conditional mean $\mathrm{E}[\boldsymbol{y}_i | \boldsymbol{x}_i]$

$$\begin{aligned} \psi_{ic} = \mathrm{E}[y_{ic} | \boldsymbol{x}_i] &= \frac{\frac{L_i!}{k \prod_{j=1}^d x_{ij}!} \prod_{j=1}^m p_{cj}^{x_{ij}}}{\frac{L_i!}{k \prod_{j=1}^m x_{ij}!} \sum_{s=1}^k \prod_{j=1}^m p_{sj}^{x_{ij}}} \\ &= \frac{\prod_{j=1}^m p_{cj}^{x_{ij}}}{\sum_{s=1}^k \prod_{j=1}^m p_{sj}^{x_{ij}}}. \end{aligned} \tag{7}$$

Next we maximize $\mathrm{E}[-\log p(\boldsymbol{y}, \boldsymbol{x})]$, which in this case is

$$\sum_{i=1}^n \sum_{c=1}^k \psi_{ic} \sum_{j=1}^m x_{ij} \log(p_{cj}).$$

To maximize it, we get

$$p_{cj} = \frac{\sum_{i=1}^n \psi_{ic} x_{ij}}{\sum_{i=1}^n \psi_{ic} L_i} \tag{8}$$

The EM algorithm for this model alternates between (7) and (8).

(c) An implementation is given. A practical issue is when calculating $\boldsymbol{\psi}_i$'s using (7), it takes the products of many small numbers, which may cause underflows.
To calculate the numerator of (7), we can use the formula

$$\nu_{ic} = \exp\left(\sum_{j=1}^m x_{ij} \log(p_{cj}) \right).$$

Then

$$\psi_{ic} = \frac{\nu_{ic}}{\sum_{s=1}^k \nu_{ic}}.$$

The result changes from different initializations, as expected when trying to minimize a nonconvex objective. Nevertheless, the solution seems to make more sense compared to the GMM model. To get more meaningful results, you might need to do some engineering first, for example by removing the 'stop words' (words that show up more often in all cases but do not carry much information, like 'that' or 'just').

topic 1	topic 2	topic 3	topic 4	topic 5	topic 6	topic 7	topic 8	topic 9	topic 10
more	am	hello	just	shipping	hello	subscribe	good	it	just
say	satan	testing	kidding	box	testing	quit	luck	only	kidding
need	just	just	good	manual	just	graphics	just	message	good
just	good	good	am	dennis	good	comp	am	test	am
good	luck	am	luck	included	am	just	satan	is	luck
am	kidding	luck	satan	just	luck	good	kidding	this	satan
luck	included	satan	included	good	satan	am	included	just	included
satan	shipping	kidding	shipping	am	kidding	luck	shipping	good	shipping
kidding	hello	included	hello	luck	included	satan	hello	am	hello
included	testing	shipping	testing	satan	shipping	kidding	testing	luck	testing

topic 11	topic 12	topic 13	topic 14	topic 15	topic 16	topic 17	topic 18	topic 19	topic 20
am	good	is	more	shipping	included	is	is	hello	included
satan	luck	this	say	box	shipping	this	this	testing	shipping
just	just	tesrt	need	manual	just	test	test	just	just
good	am	test	just	dennis	good	thanks	erme	good	good
luck	satan	thanks	good	included	am	erme	thanks	am	am
kidding	kidding	erme	am	just	luck	tesrt	tesrt	luck	luck
included	included	just	luck	good	satan	just	just	satan	satan
shipping	shipping	good	satan	am	kidding	good	good	kidding	kidding
hello	hello	am	kidding	luck	hello	am	am	included	hello
testing	testing	luck	included	satan	testing	luck	luck	shipping	testing