Columbia University – Fall 2013

# Machine Learning Homework #4

Joe Ellis - jge2105

November 21, 2013

## 1 Problem 1 – EM Derivation

In this problem we will derive the Expectation-Maximization algorithm for a mixture of multinomials. Suppose, that $x$ is represented as a vector such that $x(j) = 1$ if $x$ takes the $j^{th}$ value, and $\sum_j x(j) = 1$. The distribution of $x$ is described by a mixture of $K$ discrete multinomials such that:

$$p(x) = \sum_{k=1}^{K} \pi_k p(x|\mu_k) \tag{1}$$

where

$$p(x|\mu_k) = \prod_{j=1}^{M} \mu_k(j)^{x(j)}. \tag{2}$$

Where $\pi_k$ is the mixing coefficients, and $\mu_k$ specifies the parameters of the $k^t h$ component. We then must derive the Expectation and Maximization steps to maximize the log-likelihood of an observed data set $\{x_i\}$.

### 1.1 Expectation Step

First we want to calculate what is referred to in EM as the responsibilities of each data point given the mixtures. To do this we want to find the $p(z_i|x_i;\theta)$, where $z_i$ is the mixture that this data point belongs to. In this way, we can estimate the likelikhood that each data point was generated by one of the given mixtures, and then weight our estimates in the maximization step accordingly.

$$\begin{aligned} p(z_i|x_i|;\theta) &= \frac{p(x_i|z_i;\theta)p(z_i)}{p(xi)},\ byBayesRule && (3)\\ &= \frac{p(x_i|z_i;\theta)\pi_k}{p(xi)} && (4)\\ &= \frac{p(x_i|\mu_k)\pi_k}{\sum_{k=1}^K \pi_k p(x|\mu_k)} && (5)\\ &= \frac{\pi_k \mu_k^{x_i(j)}}{\sum_{k=1}^K \pi_k \mu_k^{x_i(j)})}. && (6)\\ & && (7) \end{aligned}$$

Therefore, we have the responsibilities of each data point, and have calculated how likely each data point was generated from each mixture.

### 1.2 Maximization Step

Now that we have the responsibilities we will in turn now calculate the parameters $\mu_k$ and $\pi_k$ that maximize the log-likelihood function of the seen data. Let's find the log-likelihood of the data given our hidden variables $\theta$, to do this, we must maximize the general form of the equation below.

$$\theta := \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{z_i} (Q_i(z_i) \frac{p(x_i, z_i; \theta)}{Q_i(z_i)}). \tag{8}$$

In the above equation $Q_i(z_i)$ is the responsibility for each mixture and data point that is calculated using the E-step from above. In the derivation below we will use the notation $\tau_{k,i}$, where $\tau_{k,i} = Q_i(z_k)$. Let's refer to the above equation as $l(\theta)$

$$
\begin{aligned}
l(\theta) &= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\frac{p(x_i,z_i;\theta)}{\tau_{k,i}} && (9)\\
&= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(p(x_i,z_i;\theta)) - \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\tau_{k,i}) && (10)\\
&= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(p(x_i,z_i;\theta)) - const && (11)\\
&= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\pi_k\prod_{j=1}^{M}\mu_k(j)^{x_i(j)}) && (12)\\
&= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\pi_k) + \sum_{j=1}^{M}log(\mu_k(j)^{x_i(j)}) && (13)\\
&= \sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\pi_k) + \tau_{k,i}\sum_{j=1}^{M}x_i(j)log(\mu_k(j)) && (14)\\
&. && (15)
\end{aligned}
$$

Now that we have derived the log likelihood we must maximize the above equation with respect to the parameters $\mu$ and $\pi$. However the maximization is not straight forward and we must use laplaclian variables to satisfy the constraints of, $\sum_j \mu_k(j) = 1$ and $\sum_k \pi_k = 1$. Therefore, let's do the derivation. We set the derivative with respect to $\mu_k(j)$ and the constraints with laplace variable $\lambda$ to zero and solve.

$$
\begin{aligned}
\frac{\partial l(\theta)}{\partial \mu_k(j)} &= \frac{\partial}{\partial \mu_k(j)}\sum_{i=1}^{N}\sum_{k=1}^{K}\tau_{k,i}log(\pi_k) + \tau_{k,i}\sum_{j=1}^{M}x_i(j)log(\mu_k(j)) - \sum_{k=1}^{K}\lambda_k(\sum_{j=1}^{M}\mu_k(j) - 1) && (16)\\
&= \sum_{i=1}^{N}\tau_{k,i}\frac{x_i(j)}{\mu_k(j)} - \lambda_k = 0. && (17)\\
& && (18)
\end{aligned}
$$

Thus,

$$\mu_k(j) = \frac{\sum_{i=1}^{N} x_i(j)\tau_{k,i}}{\lambda_k}. \tag{19}$$

Plugging $\mu_k(j)$ back into the equation, $\sum_{j=1}^{M} \mu_k(j) - 1 = 0$, gives us the equation for $\lambda_k$ below,

$$\lambda_k = \sum_{i=1}^{N} \sum_{m=1}^{M} \tau_{k,i} x_i(m). \tag{20}$$

So therefore we have the update for our $\mu_k(j)$ as,

$$\mu_k(j) = \frac{\sum_{i=1}^{N} \tau_{k,i} x_i(j)}{\sum_{i=1}^{N} \sum_{m=1}^{M} \tau_{k,i} x_i(m)} \tag{21}$$

Similarly, we can derive the value for the other parameter $\pi_k$, with laplacian constraints.

$$\begin{aligned} \frac{\partial l(\theta)}{\partial \pi_k} &= \frac{\partial}{\partial \pi_k} \sum_{i=1}^{N} \sum_{k=1}^{K} \tau_{k,i} log(\pi_k) + \tau_{k,i} \sum_{j=1}^{M} x_i(j) log(\mu_k(j)) - \lambda(\sum_{k=1}^{K} \pi_k - 1) && (22) \\ &= \sum_{i=1}^{N} \tau_{k,i} \frac{1}{\pi_k} - \lambda = 0. && (23) \\ & && (24) \end{aligned}$$

Thus,

$$\pi_k = \frac{\sum_{i=1}^{N} \tau_{k,i}}{\lambda}. \tag{25}$$

Plugging $\lambda$, back into the constraint equation, $\sum_{k=1}^{K} \pi_k - 1 = 0$, we solve for lambda,

$$\begin{aligned} \lambda &= \sum_{i=1}^{N} \sum_{k=1}^{K} \tau_{k,i} && (26) \\ &= N. && (27) \end{aligned}$$

So, therefore we have the update for our mixing coefficients ($\pi_k$) as,

$$\begin{aligned} \pi_k &= \frac{\sum_{i=1}^{N} \tau_{k,i}}{N} && (28) \\ & && (29) \end{aligned}$$

Therefore, we have derived the entire EM algorithm for a mixture of multinomials.

# 2 Problem 2

## 2.1 Mixtures of Gaussians

Using the code provided to us in the tutorial section of the class website we were able to perform EM on a mixture of gaussians on two datasets. The figures generated from the EM algorithm and the gaussian distributions found are superimposed over the data points.

Figures 1 and 2 show the results of the EM calculation.

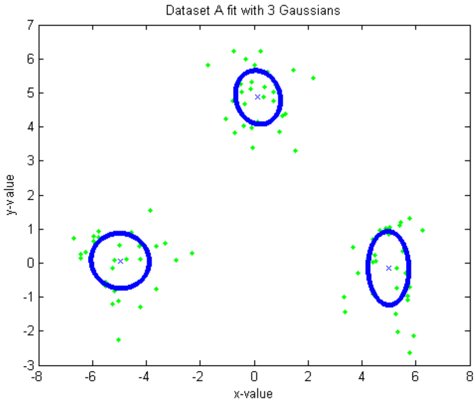

Figure 1: EM result on dataset A using 3 Gaussian distributions

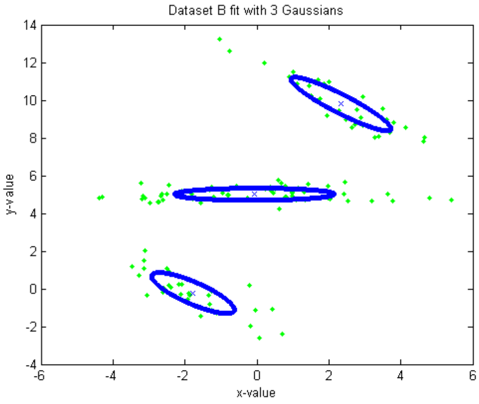

Figure 2: EM result on dataset B using 3 Gaussian distributions

## 2.2 Mixtures of Multinomials

In this section we explore using the derivation provided in section 1 to create a mixture of multinomials model to differentiate between plays written by Shakespeare and Middleton.

We have 9 plays written by Shakespeare and 9 plays written by Middleton, and we will try to classify how we well we can separate the two using multiple mixture models.

We began by implementing the EM algorithm with multinomial counts for documents clustering and classification. The results were very promising, and the responsibilities for each document after the EM algorithm were correctly classified into Shakespeare and Middleton. We can see below the mixture for each document moving to the correct responsibilities for some given random initialization with 2 assumed mixtures. To see how the responsibilities change throughout the EM algorithm, I have shown the responsibilities for each document after 1 iteration, 5 iterations, and convergence (36 iterations). These results can be seen in Figures 3, 4, and 5. As we can see from the graphs below the results converge to the proper responsibilities for each document, and we receive 100% accuracy.

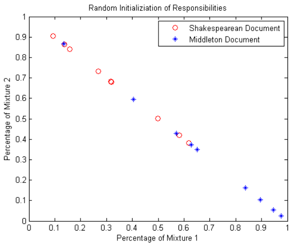

Figure 3: Document responsibilities after 1 iteration

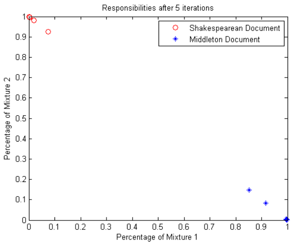

Figure 4: Document responsibilities after 5 iterations

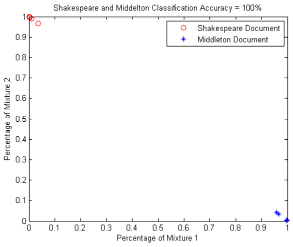

Figure 5: Final document responsibilities

Table 1: Log-Likelihood results for different numbers of multinomial mixtures

| Mixture Number ($K$) | ave train $l(\theta)$ | std train $l(\theta)$ | ave test $l(\theta)$ | std test $l(\theta)$ |
|---|---|---|---|---|
| K=1 | -3.0650e+06 | 2.7029e+04 | -8.7940e+05 | 2.7718e+04 |
| K=2 | -3.0927e+06 | 2.4745e+04 | -8.8021e+05 | 2.4963e+04 |
| K=3 | -3.0906e+06 | 1.9562e+04 | -8.9373e+05 | 1.9783e+04 |
| K=4 | -3.1112e+06 | 3.5735e+04 | -8.8161e+05 | 3.6037e+04 |

We can also see how the log-likelihood evolves during the EM algorithm. This is shown in Figure 6.

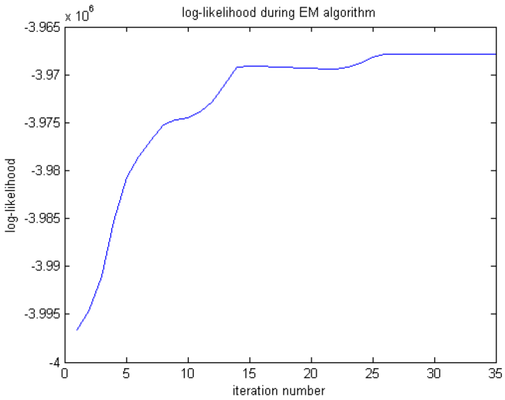

Figure 6: Log-likelihood during the EM algorithm

Finally, we want to decide the amount of mixtures that we should use on this type of data. To do this we removed 2 of the documents at random from each author in the training set, and then determined the log-likelihood of those models given our learned model on the rest of the data. The values for the average log-likelihood of the train and test set and standard deviation of the log-likelihood for train and test can be seen in Table 1. We can see that as the number of mixtures used $K$ rises we have over-fitting on the training set and rising log-likelihood. However, the log-likelihood on the test set lowers, because the model begins to over-fit to the training data. For most of the examples on the training sets we have larger K improves the log-likelihood which makes sense. It appears that using 2 or 1 mixture makes the most sense for this result, which makes some logical sense because there are 2 authors, but they may not have a large difference in writing style. 2 multinomial mixtures seems to be the proper number of multinomials for document classification or document clustering on this particular dataset.

The only issue that I had creating this program was that the responsibilities were converging on 1 and 0 to quickly, and causing problems. Therefore, to slow down the convergence and allow for the EM algorithm to work I divided the document probabilities for each mixture by 1000, and this solved the issue. Instead of the log probabilities for the documents being possibly 1000 values apart, they were then only 1 to 2 integers apart allowing for proper conversion. This fix can be seen in line 81 of “mix_mult_model.m”. The code to replicate these results can be seen in the files attached to this write up as, “mult_mix_model.m” and “RunProblem2.m”.

# 3 Problem 3

## 3.1 Part A

In this section we will prove that the arithmetic mean on non-negative numbers is at least their geometric mean. First let’s define the arithmetic mean and geometric mean for some set of numbers. Assume we have the set $X$, where $X = \{x_1, x_2, ..., x_N\}$ and $x_i \in \mathbb{P}^+$. The arithmetic mean can be found as follows,

$$mean_{arithmetic} = \frac{\sum_i x_i}{N}. \tag{30}$$

The geometric mean can be found as follows,

$$mean_{geometric} = \sqrt[N]{\prod_i x_i} \tag{31}$$

We will use Jensen’s Inequality with natural logarithm to prove that the arithmetic mean is always larger than the geometric mean.

$$\begin{align} log(mean_{arithmetic}) &= log(\frac{\sum_i xi}{N}) \tag{32}\\ &= log(\frac{x_1}{N} + \frac{x_2}{N} + ... + \frac{x_N}{N}) \tag{33}\\ &\geq \frac{1}{N}log(x_1) + \frac{1}{N}log(x_2) + ... + \frac{1}{N}log(x_N),\ by\ Jensen \tag{34}\\ &= log(\sqrt[N]{x_1}\sqrt[N]{x_2}...\sqrt[N]{x_N}) \tag{35}\\ &= log(\sqrt[N]{x_1 x_2 ... x_N}) \tag{36}\\ &= log(mean_{geometric}) \tag{37} \end{align}$$

Since we know that $log$ is a strictly increasing function, then we have proven that $mean_{arithmetic} > mean_{geometric}$.

## 3.2 Part B

Now we want to prove using Jensen’s inequality that,

$$\sum_i exp(\theta^T f_i) \geq exp(\theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i log(\alpha_i)), \tag{38}$$

$$where\ \alpha_i = \frac{exp(\theta^T f_i)}{\sum_j exp(\theta^T f_i)}. \tag{39}$$

First we notice that the set of constant variables $\sum_i \alpha_i = 1$, and therefore they are weights on each point. Using this knowledge, we can then use Jensen's inequality on this problem. Let's start with the right side of the equation, and simplify it. Thus,

$$\begin{aligned}
\theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i log(\alpha_i) &= \theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i log(\frac{exp(\theta^T f_i)}{\sum_j exp(\theta^T f_j)}) && (40) \\
&= \theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i (log(exp(\theta^T f_i)) - log(\sum_j exp(\theta^T f_j))) && (41) \\
&= \theta^T \sum_i \alpha_i f_i - (\sum_i \alpha_i \theta^T f_i - \sum_i \alpha_i log(\sum_j exp(\theta^T f_j))) && (42) \\
&= \theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i \theta^T f_i + \sum_i \alpha_i log(\sum_j exp(\theta^T f_j)) && (43) \\
&= \theta^T \sum_i \alpha_i f_i - \theta^T \sum_i \alpha_i f_i + \sum_i \alpha_i log(\sum_j exp(\theta^T f_j)) && (44) \\
&= \sum_i \alpha_i log(\sum_j exp(\theta^T f_j)) && (45) \\
&\leq log(\sum_i \alpha_i \sum_j exp(\theta^T f_j)) && (46) \\
&= log(\sum_i \frac{exp(\theta^T f_i)}{\sum_j exp(\theta^T f_j)} \sum_j exp(\theta^T f_j)) && (47) \\
&= log(\sum_i exp(\theta^T f_i)). && (48) \\
& && (49)
\end{aligned}$$

Now if we plug this upper bound, because *exp* is strictly increasing, into our original inequality we get,

$$\begin{aligned}
exp(\theta^T \sum_i \alpha_i f_i - \sum_i \alpha_i log(\alpha_i)) &\leq exp(log(\sum_i exp(\theta^T f_i))) && (50) \\
&= \sum_i exp(\theta^T f_i). && (51)
\end{aligned}$$

Therefore, we have shown what we wanted to prove.