CSC321 Assignment 4

Zikun Chen

April 4, 2017

Part 1

1) - For π_k:

The relevant terms are

$$\log(\boldsymbol{\pi}) = \log(\prod_{k=1}^{K} \pi_k^{a_k - 1}) = \sum_{k=1}^{K} (a_k - 1)\log(\pi_k)$$

$$\log \Pr(z^{(i)} = k) = \log(\pi_k)$$

We also have the constraint

$$\sum_{k=1}^{K} \pi_k = 1$$

Lagrangian:

$$\mathcal{L} = \sum_{k=1}^{K} \sum_{i=1}^{N} r_k^{(i)} \log(\pi_k) + \log(\boldsymbol{\pi}) + \lambda(1 - \sum_{k=1}^{K} \pi_k)$$

$$\frac{\partial \mathcal{L}}{\partial \pi_k} = \frac{\sum_{i=1}^{N} r_k^{(i)}}{\pi_k} + \frac{a_k - 1}{\pi_k} - \lambda = 0$$

Let $N_k = \sum_{i=1}^{N} r_k^{(i)}$ be the count for outcome k

$$\pi_k = \frac{N_k + a_k - 1}{\lambda}$$

plug into the constraint:

$$\frac{\sum_{k=1}^{K} (N_k + a_k - 1)}{\lambda} = 1$$

$$\lambda = \sum_{k=1}^{K}(N_k + a_k - 1)$$

Since a_k's are all equal in this case, let $a_{\text{mix}} = a_1 = \cdots = a_K$

$$\pi_k \leftarrow \frac{N_k + a_k - 1}{\sum_{k'=1}^{K}(N_{k'} + a_{k'} - 1)} = \frac{N_k + a_{\text{mix}} - 1}{\sum_{k'=1}^{K} N_{k'} + Ka_{\text{mix}} - K}$$

- For $\theta_{k,j}$:

The relevant terms are

$$\log(\boldsymbol{\Theta}) = \log(\prod_{k=1}^{K}\prod_{j=1}^{D}\theta_{k,j}^{a-1}(1-\theta_{k,j})^{b-1}) = \sum_{k=1}^{K}\sum_{j=1}^{D}[(a-1)\log(\theta_{k,j})+(b-1)\log(1-\theta_{k,j})]$$

$$\log\, p(\boldsymbol{x}^{(i)}|z^{(i)} = k) = \log(\prod_{j=1}^{D}\theta_{k,j}^{x_j^{(i)}}(1-\theta_{k,j})^{1-x_j^{(i)}}) = \sum_{j=1}^{D}[x_j^{(i)}\log(\theta_{k,j})+(1-x_j^{(i)})\log(1-\theta_{k,j})]$$

$$\frac{\partial}{\partial\theta_{k,j}}\sum_{i=1}^{N}\sum_{k'=1}^{K} r_{k'}^{(i)}\log\, p(\boldsymbol{x}^{(i)}|z^{(i)} = k') + \log(\boldsymbol{\Theta})$$

$$= \sum_{i=1}^{N} r_k^{(i)}(\frac{x_j^{(i)}}{\theta_{k,j}} - \frac{1-x_j^{(i)}}{1-\theta_{k,j}}) + \frac{a-1}{\theta_{k,j}} - \frac{b-1}{1-\theta_{k,j}}$$

$$= \frac{\sum_{i=1}^{N} r_k^{(i)}(x_j^{(i)} - \theta_{k,j})}{\theta_{k,j}(1-\theta_{k,j})} + \frac{a-1+(2-a-b)\theta_{k,j}}{\theta_{k,j}(1-\theta_{k,j})} = 0$$

Then

$$\sum_{i=1}^{N} r_k^{(i)}x_j^{(i)} - \theta_{k,j}\sum_{i=1}^{N} r_k^{(i)} + a - 1 + (2-a-b)\theta_{k,j} = 0$$

Therefore, rearrange we get

$$\theta_{k,j} \leftarrow \frac{\sum_{i=1}^{N} r_k^{(i)}x_j^{(i)} + a - 1}{a + b - 2 + \sum_{i=1}^{N} r_k^{(i)}}$$

2) Output:

```
>>> print_part_1_values()
pi[0] 0.085
pi[1] 0.13
theta[0, 239] 0.642710622711
theta[3, 298] 0.465736124958
```

Part 2

1) Add in the variable $m_j^{(i)}$ representing partial observation to the Bernoulli Distribution:

$$p(\mathbf{x}_{obs}^{(i)}|z=k) = \prod_{j=1}^{D} (\theta_{k,j}^{x_j^{(i)}} (1-\theta_{k,j})^{1-x_j^{(i)}})^{m_j^{(i)}}$$

By Bayes' Rule:

$$\Pr(z=k|\mathbf{x}_{obs}) = \frac{\Pr(\mathbf{x}_{obs}|z=k)\Pr(z=k)}{\sum_{k'=1}^{K} \Pr(\mathbf{x}_{obs}|z=k')\Pr(z=k')}$$

$$= \frac{\pi_k \prod_{i=1}^{N} \prod_{j=1}^{D} (\theta_{k,j}^{x_j^{(i)}} (1-\theta_{k,j})^{1-x_j^{(i)}})^{m_j^{(i)}}}{\sum_{k'=1}^{K} \pi_{k'} \prod_{i=1}^{N} \prod_{j=1}^{D} (\theta_{k',j}^{x_j^{(i)}} (1-\theta_{k',j})^{1-x_j^{(i)}})^{m_j^{(i)}}}$$

2)

$$p(z, \mathbf{x}) = \frac{\Pr(z=k|\mathbf{x}_{obs})}{\Pr(\mathbf{x}_{obs})} \propto \Pr(z=k|\mathbf{x}_{obs})$$

$$\log p(z, \mathbf{x}) \propto \log \Pr(z=k|\mathbf{x}_{obs}) = \log \Pr(\mathbf{x}_{obs}|z=k) + \log \Pr(z=k)$$

$$= \log(\pi_k) + \sum_{i=1}^{N} \sum_{j=1}^{D} [m_j^{(i)} x_j^{(i)} \log(\theta_{k,j}) + m_j^{(i)} (1-x_j^{(i)}) \log(1-\theta_{k,j})]$$

3) Output:

```
>>> print_part_2_values()
R[0, 2] 0.174889514921
R[1, 0] 0.688537676109
P[0, 183] 0.651615199813
P[2, 628] 0.474080172491
```

Part 3

1) If a pixel is always 0 in the training set and we set $a = 1$, i.e. $x_j^{(i)}$ for all $n = 1, \cdots, N$, then from the formula in Part 1, during training, the denominator of the $\theta_{k,j}$ update is $\sum_{i=1}^{N} r_k^{(i)} x_j^{(i)} + a - 1 = 0$.

After training, this particular pixel will be assigned the probability $\theta_{k,j} = 0$ of being 1. If the pixel occurs in the test set as 1, then the MAP learning algorithm does not give a good theta estimation for this pixel. Therefore, we should avoid choosing an uniform prior distribution ($a = 1$).

2) Although the model in part 1 is given the additional label information, the number of components is restricted at 10 different digit classes, which is insufficient to capture the different styles people write digits in.

   On the other hand, in the second model, it is capable of learning more components (initially we chose a large number 100, and some components die out during training), this makes sense because there are different ways to write the same digit, for example, 7 can have a bar in the middle. It would maximize the log probability of the pixels a bit better if it learns about the different ways people write digits. Since the actual task here is image completion instead of getting the correct label specified by humans, it turns out that the components learned by the model itself (more than ten), are better at completing the unobserved part of the image by observing what kind of style the digit is written in.

3) It is true that there are more human-labeled 1's than 8's in both the training set and the test set, which can influence the average log likelihood computation. But this is not a necessary condition for the difference in the average log probabilities between 1 and 8.

   However, for each image, it might assign high probability for more than one digit classes because the model gives soft probability assignments for all 10 digits. For example, 7 or 2 with certain styles can look very much like 1, and the model would give a high log probability to 1 as well. On the other hand, 8 has a very distinctive shape and the model will rarely assign a higher probability of being 8 to other digits. So when we sum up the log probabilities for each image at the end to get the average log probabilities, other than human-labeled 1's, we will get a big contribution from other digits as well (like 7 and 2), which causes the average log probability of 1 to be higher than 8.