Homework 3

Deadline: Tuesday, April 4, 2023, 5:00PM ET.

Submission: You will need to submit three files:

- Your answers to all of the questions, as a PDF file titled `hw3_writeup.pdf`. You can produce the file however you like (e.g. LaTeX, Microsoft Word, scanner), as long as it is readable. If you need to split your writeup into multiple files, that's OK, as long as we can figure out what you did. You are expected to use vectorized notation when applicable.
- The completed Python files `naive_bayes.py` and `q4.py`.

Neatness Point: One point will be given for neatness. You will receive this point as long as we don't have a hard time reading your solutions or understanding the structure of your code.

Late Submission: Everyone will receive 3 grace days for the course, which can be used at any point during the semester on the three assignments. No credit will be given for assignments submitted after 3 days.

Homeworks are individual work. See the Course Information handout[1] for detailed policies.

[1] `https://www.cs.toronto.edu/michael/teaching/csc311w23/index.html`

1. **[5pts] Backprop**

   In this question, you will derive the backprop updates for a particular neural net architecture. The network is similar to the multilayer perceptron architecture from lecture, and has one linear hidden layer. However, there are two architectural differences:

   - In addition to the usual vector-valued input $\mathbf{x}$, there is a vector-valued "context" input $\boldsymbol{\eta}$. (The particular meaning of $\boldsymbol{\eta}$ isn't important for your derivation, but think of it as containing additional task information, such as whether to focus on the left or the right half of the image.) The hidden layer activations are *modulated* based on $\boldsymbol{\eta}$; this means they are multiplied by a value which depends on $\boldsymbol{\eta}$.
   - The network has a *skip connection* which sends information directly from the input to the output of the network.

   The loss function is the binary cross-entropy loss. The forward pass equations and network architecture are as follows. The symbol $\odot$ represents elementwise multiplication, and σ denotes the logistic/sigmoid function ($\sigma(x) = \dfrac{1}{1+e^{-x}}$).

   $$\begin{aligned}
   \mathbf{z} &= \mathbf{W}\mathbf{x} \\
   \mathbf{s} &= \mathbf{U}\boldsymbol{\eta} \\
   \mathbf{h} &= \mathbf{z} \odot \mathbf{s} \\
   y &= \sigma(\mathbf{v}^\top \mathbf{h} + \mathbf{r}^\top \mathbf{x}) \\
   \mathcal{L} &= t \log y + (1-t)\log(1-y)
   \end{aligned}$$

   The model parameters are matrices $\mathbf{W}$ and $\mathbf{U}$, and vectors $\mathbf{v}$ and $\mathbf{r}$. Note that there is only one output unit, i.e. y and t are scalars.

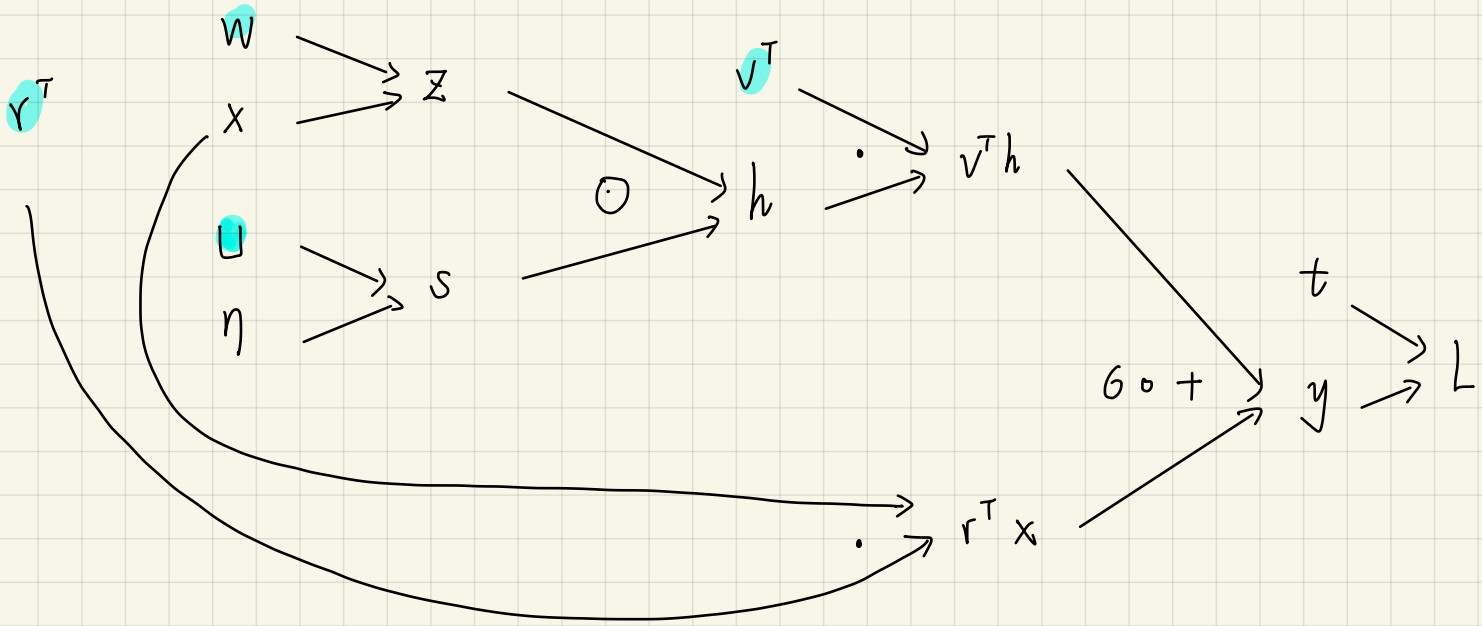
   (a) **[1pt]** Draw the computation graph relating $\mathbf{x}$, $\mathbf{z}$, $\boldsymbol{\eta}$, $\mathbf{s}$, $\mathbf{h}$, $\mathcal{L}$, and the model parameters.

   (b) **[1pt]** Recall that $\sigma(x)$ is the logistic function. Show that

   $$\frac{d\sigma(x)}{dx} = \sigma(x)(1-\sigma(x)).$$

   (c) **[3pts]** Derive the backprop formulas to compute the error signals for all of the model parameters, as well as $\overline{\mathbf{x}}$ and $\overline{\boldsymbol{\eta}}$ (recall from lecture that these are the derivatives of the cost function with respect to the variables in question). Also include the backprop formulas for all intermediate quantities needed as part of the computation.

(a)



(b) $$\frac{d\sigma(x)}{dx} = \frac{d\ \frac{1}{1+e^{-x}}}{dx}$$

$$= \frac{-1}{(1+e^{-x})^2} \cdot (-1) \cdot e^{-x}$$

$$= \frac{e^{-x}}{(1+e^{-x})^2}$$

$$= \sigma^2(x) \cdot e^{-x} \quad , \text{since } \sigma(x) = \frac{1}{1+e^{-x}}$$

$$= \sigma^2(x) \cdot \frac{1-\sigma(x)}{\sigma(x)}$$

$$= \sigma(x) \cdot (1-\sigma(x)) \quad , \text{since } \sigma(x) = \frac{1}{1+e^{-x}} \Rightarrow e^{-x} = \frac{1-\sigma(x)}{\sigma(x)}$$

(c)

$$\bar{L} = 1$$

$$\bar{y} = \bar{L} \cdot \frac{dL}{dy}$$

$$= 1 \cdot \left(\frac{t}{y} + \frac{t-1}{1-y} \right)$$

$$= \frac{t(1-y) + (t-1) \cdot y}{y(1-y)}$$

$$= \frac{t-y}{y(1-y)}$$

$$\overline{v^T h} = \bar{y} \cdot \frac{dy}{d v^T h}$$

$$= \bar{y} \cdot \frac{dy}{d(v^T h + r^T x)} \cdot \frac{d(v^T h + r^T x)}{d(v^T h)}$$

$$= \bar{y} \cdot y \cdot (1-y) \cdot 1$$

$$= \bar{y} \cdot y \cdot (1-y)$$

$$\overline{v^T} = \overline{v^T h} \cdot \frac{d(v^T h)}{d v^T}$$

$$= \bar{y} \cdot \sigma(y)(1-\sigma(y)) \cdot h^T$$

$$\overline{h} = \overline{v^T h} \cdot \frac{d(v^T h)}{d(h)}$$

$$= \bar{y} \cdot \sigma(y)(1-\sigma(y)) \cdot v^T$$

$$\overline{r^T x} = \bar{y} \cdot \frac{dy}{d r^T x}$$

$$= \bar{y} \cdot \frac{dy}{d(v^T h + r^T x)} \cdot \frac{d(v^T h + r^T x)}{d(r^T x)}$$

$$= \bar{y} \cdot y(1-y) \cdot 1$$

$$= \bar{y} \cdot y \cdot (1-y)$$

$$\overline{r^T} = \overline{r^T x} \cdot \frac{d\, r^T x}{d r^T}$$

$$= \overline{r^T x} \cdot x^T$$

$$\bar{z} = \bar{h} \cdot \frac{dh}{dz}$$

$$= \bar{h} \cdot s$$

$$\bar{w} = \bar{z} \cdot \frac{dz}{dw}$$

$$= \bar{z} \cdot x^T$$

$$\bar{x} = \bar{z} \cdot \frac{dz}{dx}$$

$$= \bar{z} \cdot w$$

$$\bar{s} = \bar{h} \frac{dh}{ds}$$

$$= \bar{h} \cdot z$$

$$\bar{u} = \bar{s} \cdot \frac{ds}{du}$$

$$= \bar{s} \cdot \eta^T$$

$$\bar{\eta} = \bar{s} \cdot u$$

2. **[13pts] Fitting a Naïve Bayes Model**

In this question, we'll fit a Naïve Bayes model to the MNIST digits using maximum likelihood. In addition to the mathematical derivations, you will complete the implementation in `naive_bayes.py`.

The starter code will download the dataset and parse it for you: Each training sample $(\mathbf{t}^{(i)}, \mathbf{x}^{(i)})$ is composed of a vectorized binary image $\mathbf{x}^{(i)} \in \{0,1\}^{784}$, and 1-of-10 encoded class label $\mathbf{t}^{(i)}$. i.e. $t_c^{(i)} = 1$ means image i belongs to class c.

Given parameters $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$, Naïve Bayes defines the joint probability of each data point $\mathbf{x}$ and its class label c as follows:

$$p(\mathbf{x}, c \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) = p(c \,|\, \boldsymbol{\theta}, \boldsymbol{\pi}) p(\mathbf{x} \,|\, c, \boldsymbol{\theta}, \boldsymbol{\pi}) = p(c \,|\, \boldsymbol{\pi}) \prod_{j=1}^{784} p(x_j \,|\, c, \theta_{jc}).$$

where $p(c \,|\, \boldsymbol{\pi}) = \pi_c$ and $p(x_j = 1 \,|\, c, \boldsymbol{\theta}) = \theta_{jc}$. Here, $\boldsymbol{\theta}$ is a matrix of probabilities for each pixel and each class, so its dimensions are 784×10, and $\boldsymbol{\pi}$ is a vector with one entry for each class. (Note that in the lecture, we simplified notation and didn't write the probabilities conditioned on the parameters, i.e. $p(c|\boldsymbol{\pi})$ is written as $p(c)$ in lecture slides).

For binary data ($x_j \in \{0,1\}$), we can write the Bernoulli likelihood as

$$p(x_j \,|\, c, \theta_{jc}) = \theta_{jc}^{x_j} (1 - \theta_{jc})^{(1-x_j)}, \tag{1}$$

which is just a way of expressing $p(x_j = 1|c, \theta_{jc}) = \theta_{jc}$ and $p(x_j = 0|c, \theta_{jc}) = 1 - \theta_{jc}$ in a compact form.

For the prior $p(\mathbf{t} \,|\, \boldsymbol{\pi})$, we use a categorical distribution (generalization of Bernoulli distribution to multi-class case),

$$p(t_c = 1 \,|\, \boldsymbol{\pi}) = p(c \,|\, \boldsymbol{\pi}) = \pi_c \;\text{ or equivalently }\; p(\mathbf{t} \,|\, \boldsymbol{\pi}) = \Pi_{j=0}^{9} \pi_j^{t_j} \quad \text{where} \quad \sum_{i=0}^{9} \pi_i = 1,$$

where $p(c \,|\, \boldsymbol{\pi})$ and $p(\mathbf{t} \,|\, \boldsymbol{\pi})$ can be used interchangeably. You will fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using MLE and MAP techniques. In both cases, your fitting procedure can be written as a few simple matrix multiplication operations. For this question, you should write down three significant digits for numerical answers.

(a) **[3pts]** First, derive the *maximum likelihood estimator* (MLE) for the class-conditional pixel probabilities $\boldsymbol{\theta}$ and the prior $\boldsymbol{\pi}$. Derivations should be rigorous.

Hint 1: We saw in lecture that MLE can be thought of as 'ratio of counts' for the data, so what should $\hat{\theta}_{jc}$ be counting?

Hint 2: Similar to the binary case, when calculating the MLE for π_j for $j = 0, 1, ..., 8$, write $p(\boldsymbol{t}^{(i)} \,|\, \boldsymbol{\pi}) = \Pi_{j=0}^{9} \pi_j^{t_j^{(i)}}$ and in the log-likelihood replace $\pi_9 = 1 - \Sigma_{j=0}^{8} \pi_j$, and then take derivatives w.r.t. π_j. This will give you the ratio $\hat{\pi}_j / \hat{\pi}_9$ for $j = 0, 1, .., 8$. You know that $\hat{\pi}_j$'s sum up to 1.

(b) **[2pts]** Derive the log-likelihood $\log p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\pi})$ for a single training image.

(c) **[1pt]** Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MLE, and try to report the average log-likelihood per data point $\frac{1}{N} \Sigma_{i=1}^{N} \log p(\mathbf{t}^{(i)}|\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$, using Equation (1). What goes wrong? (it's okay if you can't compute the average log-likelihood here).

(d) **[1pt]** Plot the MLE estimator $\hat{\boldsymbol{\theta}}$ as 10 separate greyscale images, one for each class.

(e) **[3pts]** Derive the *Maximum A posteriori Probability* (MAP) estimator for the class-conditional pixel probabilities $\boldsymbol{\theta}$, using a Beta(α, β) prior on each θ_{jc}. Evaluate the estimator for $\alpha = 3, \beta = 3$; how does it compare to the MLE estimator?

(f) **[1pt]** Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MAP estimators from the previous part, and report both the average log-likelihood per data point, $\frac{1}{N}\Sigma_{i=1}^{N} \log p(\mathbf{t}^{(i)}|\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$, and the accuracy on both the training and test set. The accuracy is defined as the fraction of examples where the true class is correctly predicted using $\hat{c} = \text{argmax}_c \log p(t_c = 1|\mathbf{x}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$.

(g) **[1pt]** Plot the MAP estimator $\hat{\boldsymbol{\theta}}$ as 10 separate greyscale images, one for each class.

(h) **[1pt]** List one advantage of the Naïve Bayes approach and one reason why it may not be reasonable for this problem.

(a) **[3pts]** First, derive the *maximum likelihood estimator* (MLE) for the class-conditional pixel probabilities $\boldsymbol{\theta}$ and the prior $\boldsymbol{\pi}$. Derivations should be rigorous.

Hint 1: We saw in lecture that MLE can be thought of as 'ratio of counts' for the data, so what should $\hat{\theta}_{jc}$ be counting?

Hint 2: Similar to the binary case, when calculating the MLE for π_j for $j = 0, 1, ..., 8$, write $p(\boldsymbol{t}^{(i)} \mid \boldsymbol{\pi}) = \Pi_{j=0}^{9} \pi_j^{t_j^{(i)}}$ and in the log-likelihood replace $\pi_9 = 1 - \Sigma_{j=0}^{8} \pi_j$, and then take derivatives w.r.t. π_j. This will give you the ratio $\hat{\pi}_j / \hat{\pi}_9$ for $j = 0, 1, .., 8$. You know that $\hat{\pi}_j$'s sum up to 1.

$$p(x, c \mid \theta, \pi) = p(c \mid \pi) \prod_{j=1}^{784} p(x_j \mid c, \theta_{jc})$$

$$\Rightarrow p(\vec{x}, c \mid \theta, \pi) = \prod_{i=1}^{N} p(c^{(i)} \mid \pi) \prod_{j=1}^{784} p(x_j^{(i)} \mid c^{(i)}, \theta_{jc^{(i)}})$$

$$\Rightarrow \log p(\vec{x}, c \mid \theta, \pi) = \sum_{i=1}^{N} \log p(c^{(i)} \mid \pi) \cdot \prod_{j=1}^{784} p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}})$$

$$= \sum_{i=1}^{N} \left[\log p(c^{(i)} \mid \pi) + \sum_{j=1}^{784} \log p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}}) \right]$$

$$= \sum_{i=1}^{N} \left[\log p(\vec{t}^{(i)} \mid \pi) + \sum_{j=1}^{784} \log p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}}) \right]$$

$$= \sum_{i=1}^{N} \left[\log \prod_{j=0}^{9} \pi_j^{t_j^{(i)}} + \sum_{j=1}^{784} \log p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}}) \right]$$

$$= \sum_{i=1}^{N} \left[\sum_{j=0}^{9} t_j^{(i)} \log \pi_j + \sum_{j=1}^{784} \log p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}}) \right]$$

$$= \sum_{i=1}^{N} \sum_{j=0}^{9} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} \sum_{j=1}^{784} \log p(x_j^{(i)} \mid c^{(i)}, \theta_{j,c^{(i)}})$$

$$= \sum_{i=1}^{N} \sum_{j=0}^{9} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} \sum_{j=1}^{784} \log \theta_{j,c^{(i)}}^{x_j^{(i)}} (1 - \theta_{j,c^{(i)}})^{1 - x_j^{(i)}}$$

$$= \sum_{i=1}^{N} \sum_{j=0}^{9} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} \sum_{j=1}^{784} x_j^{(i)} \log \theta_{j,c^{(i)}} + (1 - x_j^{(i)}) \cdot \log(1 - \theta_{j,c^{(i)}})$$

$$\Rightarrow \arg\max_{\theta, \pi} \log p(\vec{x}, c \mid \theta, \pi) = \arg\max_{\theta, \pi} \sum_{i=1}^{N} \sum_{j=0}^{9} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} \sum_{j=1}^{784} x_j^{(i)} \log \theta_{j,c^{(i)}} + (1 - x_j^{(i)}) \cdot \log(1 - \theta_{j,c^{(i)}})$$

$$= \arg\max_{\pi} \underbrace{\sum_{i=1}^{N} \sum_{j=0}^{9} t_j^{(i)} \log \pi_j}_{(1)}$$

$$+$$

$$\arg\max_{\theta} \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{784} x_j^{(i)} \log \theta_{j,c^{(i)}} + (1-x_j^{(i)}) \cdot \log(1-\theta_{j,c^{(i)}})}_{②}$$

① $$\arg\max_{\pi} \sum_{i=1}^{N}\sum_{j=0}^{9} t_j^{(i)} \log \pi_j$$

$$= \arg\max_{\pi} \sum_{i=1}^{N}\left[\sum_{j=0}^{8} t_j^{(i)} \log \pi_j + t_9^{(i)} \log \pi_9\right]$$

$$= \arg\max_{\pi} \sum_{i=1}^{N}\left[\sum_{j=0}^{8} t_j^{(i)} \log \pi_j + t_9^{(i)} \log\left(1-\sum_{j=0}^{8}\pi_j\right)\right]$$

$$\Rightarrow \frac{\partial \sum_{i=1}^{N}\sum_{j=0}^{9} t_j^{(i)} \log \pi_j}{\partial \pi_j} = \sum_{i=1}^{N}\left[\frac{t_j^{(i)}}{\pi_j} - \frac{t_9^{(i)}}{1-\sum_{j=0}^{8}\pi_j}\right], \quad j = 0, 1, \cdots, 8$$

$$\Rightarrow \frac{\partial \sum_{i=1}^{N}\sum_{j=0}^{9} t_j^{(i)} \log \pi_j}{\partial \pi_j} = 0 \iff \sum_{i=1}^{N}\left[\frac{t_j^{(i)}}{\pi_j} - \frac{t_9^{(i)}}{1-\sum_{j=0}^{8}\pi_j}\right] = 0$$

$$\iff \sum_{i=1}^{N}\left[\frac{t_j^{(i)}}{\pi_j} - \frac{t_9^{(i)}}{\pi_9}\right] = 0$$

$$\iff \frac{1}{\pi_j}\sum_{i=1}^{N} t_j^{(i)} - \frac{1}{\pi_9}\sum_{i=1}^{N} t_9^{(i)} = 0$$

$$\iff \frac{1}{\pi_j} = \frac{1}{\pi_9} \cdot \frac{\sum_{i=1}^{N} t_9^{(i)}}{\sum_{i=1}^{N} t_j^{(i)}}$$

$$\iff \frac{\pi_9}{\pi_j} = \frac{\sum_{i=1}^{N} t_9^{(i)}}{\sum_{i=1}^{N} t_j^{(i)}}$$

$$\iff \frac{\pi_j}{\pi_9} = \frac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_9^{(i)}}$$

Thus, $$\frac{\hat{\pi}_j}{\hat{\pi}_9} = \frac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_9^{(i)}}, \quad j = 0, 1, \cdots, 8$$

Also, $\hat{\pi}_9$ is free and can be set as $\frac{\sum_{i=1}^{N} t_9^{(i)}}{N}$

Thus, $\hat{\pi}_j = \frac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_9^{(i)}} \cdot \frac{\sum_{i=1}^{N} t_9^{(i)}}{N} = \frac{\sum_{i=1}^{N} t_j^{(i)}}{N}$

② : $\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{j=1}^{784} x_j^{(i)} \log \theta_{j,c^{(i)}} + (1-x_j^{(i)}) \cdot \log(1-\theta_{j,c^{(i)}})$

$\Rightarrow \hat{\theta}_{j,c} = \frac{\sum_{i=1}^{N} \mathbb{1}[x_j^{(i)}=1 \ \& \ c^{(i)}=c]}{\sum_{i=1}^{N} \mathbb{1}[c^{(i)}=c]} = \frac{\#\, x_j=1 \text{ appears in class } c}{\#\text{class } c \text{ in dataset}}$, by the lecture 7

where $j = 1, 2, \cdots, 784$

(b) **[2pts]** Derive the log-likelihood $\log p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\pi})$ for a single training image.

$$p(\vec{t}|\vec{x},\theta,\pi) = \frac{p(\vec{t},\vec{x}\,|\,\theta,\pi)}{p(\vec{x})}$$

$$\Rightarrow \log p(\vec{t}|\vec{x},\theta,\pi) = \log p(\vec{t},\vec{x}|\theta,\pi) - \log p(\vec{x})$$

$$= \log\left[p(\vec{x}|\vec{t},\theta,\pi)\cdot p(\vec{t}|\theta,\pi)\right] - \log p(\vec{x})$$

$$= \log p(\vec{x}|\vec{t},\theta,\pi) + \log p(\vec{t}|\theta,\pi) - \log p(\vec{x})$$

$$= \log \prod_{j=1}^{784} p(x_j|\vec{t},\theta,\pi) + \log p(\vec{t}|\theta,\pi) - \log p(\vec{x})$$

$$= \sum_{j=1}^{784} \log p(x_j|\vec{t},\theta,\pi) + \log p(\vec{t}|\theta,\pi) - \log p(\vec{x})$$

$$= \sum_{j=1}^{784}\left[x_j \log(\theta_{jc}) + (1-x_j)\log(1-\theta_{jc})\right] + \log p(\vec{t}|\theta,\pi) - \log p(\vec{x})$$

$$= \sum_{j=1}^{784}\left[x_j \log(\theta_{jc}) + (1-x_j)\log(1-\theta_{jc})\right] + \log \prod_{j=0}^{9} \pi_j^{t_j} - \log p(\vec{x})$$

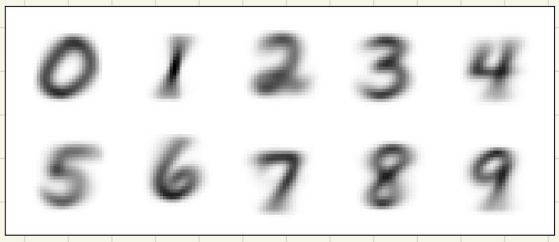
$$= \sum_{j=1}^{784}\left[x_j \log(\theta_{jc}) + (1-x_j)\log(1-\theta_{jc})\right] + \sum_{j=0}^{9} t_j \log \pi_j - \log p(\vec{x})$$

(c) **[1pt]** Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MLE, and try to report the average log-likelihood per data point $\frac{1}{N}\Sigma_{i=1}^{N} \log p(\mathbf{t}^{(i)}|\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$, using Equation (1). What goes wrong? (it's okay if you can't compute the average log-likelihood here).

Avg log-likelihood for MLE is -24.986.

The problem is: the probability is $e^{-24.986}$ which is very small

(d) **[1pt]** Plot the MLE estimator $\hat{\boldsymbol{\theta}}$ as 10 separate greyscale images, one for each class.



(e) **[3pts]** Derive the *Maximum A posteriori Probability* (MAP) estimator for the class-conditional pixel probabilities $\boldsymbol{\theta}$, using a Beta(α, β) prior on each θ_{jc}. Evaluate the estimator for $\alpha = 3, \beta = 3$; how does it compare to the MLE estimator?

$$\hat{\theta}_{MAP} = \arg\max_{\theta, c} p(\theta, c \mid D)$$

$$= \arg\max_{\theta, c} p(\theta, c) \cdot p(D \mid \theta, c)$$

$$= \arg\max_{\theta, c} \prod_{j=1}^{784} \left[\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_{jc}^{\alpha-1} (1-\theta_{jc})^{\beta-1} \right] \cdot \prod_{i=1}^{N} \prod_{j=1}^{784} \theta_{jc}^{x_j^{(i)}} (1-\theta_{jc})^{(1-x_j^{(i)})}$$

$$= \arg\max_{\theta, c} \prod_{j=1}^{784} \left[\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_{jc}^{\alpha-1} (1-\theta_{jc})^{\beta-1} \right] \cdot \prod_{j=1}^{784} \theta_{jc}^{\sum_{i=1}^{N} (x_j^{(i)} \cdot \| x^{(i)} \in c)} (1-\theta_{jc})^{\sum_{i=1}^{N} \| x^{(i)} \in c - \sum_{i=1}^{N} (\| x^{(i)} \in c \cdot x_j^{(i)})}$$

$$= \arg\max_{\theta, c} \prod_{j=1}^{784} \left[\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_{jc}^{\alpha-1} (1-\theta_{jc})^{\beta} \theta_{jc}^{\sum_{i=1}^{N} (x_j^{(i)} \cdot \| x^{(i)} \in c)} (1-\theta_{jc})^{\sum_{i=1}^{N} \| x^{(i)} \in c - \sum_{i=1}^{N} (\| x^{(i)} \in c \cdot x_j^{(i)})} \right]$$

$$= \arg\max_{\theta, c} \prod_{j=1}^{784} \left[\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta_{jc}^{\alpha-1 + \sum_{i=1}^{N} (x_j^{(i)} \cdot \| x^{(i)} \in c)} (1-\theta_{jc})^{\beta-1 + \sum_{i=1}^{N} \| x^{(i)} \in c - \sum_{i=1}^{N} (\| x^{(i)} \in c \cdot x_j^{(i)})} \right]$$

$$= \arg\max_{\theta, c} \sum_{j=1}^{784} \left(\alpha - 1 + \sum_{i=1}^{N} (x_j^{(i)} \cdot \| x^{(i)} \in c) \right) \log \theta_{jc} + \left(\beta - 1 + \sum_{i=1}^{N} \| x^{(i)} \in c - \sum_{i=1}^{N} (\| x^{(i)} \in c \cdot x_j^{(i)}) \right) \cdot \log(1-\theta_{jc}) \Big)$$

$$\Rightarrow \hat{\theta}_{jc\,(MAP)} = \frac{\alpha - 1 + \sum_{i=1}^{N} x_j^{(i)} \cdot \| x^{(i)} \in c}{\alpha + \beta - 2 + \sum_{i=1}^{N} x^{(i)} \in c}$$

When $\alpha = \beta = 3$,

$$\hat{\theta}_{jc\,(MAP)} = \frac{2 + \sum_{i=1}^{N} \left[x_j^{(i)} \cdot \mathbb{1}\{x^{(i)} \in c\} \right]}{4 + \sum_{i=1}^{N} \mathbb{1}\{x^{(i)} \in c\}}$$

Recall that $\hat{\theta}_{jc\,(MLE)} = \frac{\sum_{i=1}^{N} \mathbb{1}[x_j^{(i)} = 1 \ \& \ c^{(i)} = c]}{\sum_{i=1}^{N} \mathbb{1}[c^{(i)} = c]}$. Since the numerator of $\hat{\theta}_{jc\,(MAP)}$ is 2 larger than that of $\hat{\theta}_{jc\,(MLE)}$, but the denominator of $\hat{\theta}_{jc\,(MAP)}$ is 4 larger than that of $\hat{\theta}_{jc\,(MLE)}$, so

$$\hat{\theta}_{jc\,(MAP)} < \hat{\theta}_{jc\,(MLE)}$$

Similarly, $\pi_c = \sum_{j=1}^{784} \theta_{j,c}$

$$\Rightarrow \hat{\pi}_{c\,(MAP)} = \sum_{j=1}^{784} \hat{\theta}_{jc\,(MAP)}$$

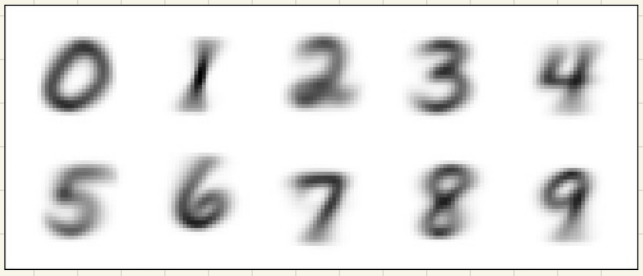
(f) **[1pt]** Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MAP estimators from the previous part, and report both the average log-likelihood per data point, $\frac{1}{N}\Sigma_{i=1}^{N} \log p(\mathbf{t}^{(i)}|\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$, and the accuracy on both the training and test set. The accuracy is defined as the fraction of examples where the true class is correctly predicted using $\hat{c} = \text{argmax}_c \log p(t_c = 1|\mathbf{x}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$.

Average log-likelihood for MAP is -5.1750

Training accuracy for MAP is 0.7936

Test accuracy for MAP is 0.759

(g) **[1pt]** Plot the MAP estimator $\hat{\boldsymbol{\theta}}$ as 10 separate greyscale images, one for each class.



(h) **[1pt]** List one advantage of the Naïve Bayes approach and one reason why it may not be reasonable for this problem.

It is simple to implement because conditional probabilities are easy to evaluate. However, Naive Bayes approach is based on the assumption that $p(x_i \mid c)$ is independent of $p(x_j \mid c)$, which doesn't make sense in our example. It is because every digit has a pattern and if x_i appears in the digit affects if x_j appears.

3. **[4pts] Logistic Regression with Gaussian Prior.**

Consider a binary classification problem where the output $y \in \{0, 1\}$ and the input $\mathbf{x} \in \mathbb{R}^p$. We model the probability of $y = 1$ given $\mathbf{x}$ using logistic regression:

$$p(y = 1|\mathbf{x}, \boldsymbol{\theta}) = \frac{1}{1 + \exp(-\mathbf{x}^T\boldsymbol{\theta})}$$

where $\boldsymbol{\theta} \in \mathbb{R}^p$ is the parameter vector.

(a)**[2pts] Maximum Likelihood Estimation (MLE).** Given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$, write down the expression for the log-likelihood of the parameter vector $\boldsymbol{\theta}$ for logistic regression. How would you optimize the resulting log-likelihood?

(b)**[2pts] Maximum A Posteriori (MAP) Estimation with Gaussian Prior.** Now consider a Gaussian prior on the parameter vector $\boldsymbol{\theta}$ with zero mean and covariance matrix $\sigma^2 I$:

$$p(\boldsymbol{\theta}) \sim \mathcal{N}(0, \sigma^2 I)$$

Derive the MAP log-likelihood of the parameter vector $\boldsymbol{\theta}$ for logistic regression with the Gaussian prior.

(a) $L(\theta) = \prod_{i=1}^{N} \left[\frac{1}{1+\exp(-x^{(i)T}\theta)}\right]^{y^{(i)}} \cdot \left[1 - \frac{1}{1+\exp(-x^{(i)T}\theta)}\right]^{1-y^{(i)}}$

$\Rightarrow \log L(\theta) = \sum_{i=1}^{N} \log\left(\frac{1}{1+\exp(-x^{(i)T}\theta)}\right)^{y^{(i)}} \cdot \left(1 - \frac{1}{1+\exp(-x^{(i)T}\theta)}\right)^{1-y^{(i)}}$

$= \sum_{i=1}^{N} y^{(i)} \cdot \log\left(\frac{1}{1+\exp(-x^{(i)T}\theta)}\right) + (1-y^{(i)}) \cdot \log\left(1 - \frac{1}{1+\exp(-x^{(i)T}\theta)}\right)$

$= \sum_{i=1}^{N} -y^{(i)} \cdot \log(1+\exp(-x^{(i)T}\theta)) + (1-y^{(i)}) \cdot \log \frac{\exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)}$

$= \sum_{i=1}^{N} (1-y^{(i)}) \cdot \log(1+\exp(-x^{(i)T}\theta))$

$+ (1-y^{(i)}) \cdot \left(-x^{(i)T}\theta - \log(1+\exp(-x^{(i)T}\theta))\right)$

$= \sum_{i=1}^{N} y^{(i)} x^{(i)T}\theta - x^{(i)T}\theta - \log(1+\exp(-x^{(i)T}\theta))$

$= \theta \cdot \sum_{i=1}^{N} \left(y^{(i)} x^{(i)T} - x^{(i)T}\right) - \sum_{i=1}^{N} \log(1+\exp(-x^{(i)T}\theta))$

$\Rightarrow \frac{d\,l(\theta)}{d\theta} = \sum_{i=1}^{N} y^{(i)} x^{(i)T} - x^{(i)T} + \sum_{i=1}^{N} \frac{x^{(i)T} \cdot \exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)}$

$= 0$

$\Leftrightarrow \sum_{i=1}^{N} y^{(i)} x^{(i)T} - x^{(i)T} = -\sum_{i=1}^{N} \frac{x^{(i)T} \cdot \exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)}$

(ii) $\sum_{i=1}^{N} \frac{x^{(i)T} \cdot \exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)} = \sum_{i=1}^{N} (1-y^{(i)}) \cdot x^{(i)T}$

(iii) $\sum_{i=1}^{N} \frac{x^{(i)T} \cdot \exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)} + (y^{(i)}-1) \cdot x^{(i)T} = 0$

(b) $\hat{\theta}_{MAP} = \arg\max_{\theta} p(\theta|D)$

$= \arg\max_{\theta} p(\theta)\, p(D|\theta)$

$= \arg\max_{\theta} \log p(\theta) + \log p(D|\theta)$

$= \arg\max_{\theta} -\frac{1}{2\sigma^2}\|\theta\|_2^2 + \theta \cdot \sum_{i=1}^{N}\left(y^{(i)} x^{(i)T} - x^{(i)T}\right) - \sum_{i=1}^{N} \log(1+\exp(-x^{(i)T}\theta))$, since

① $\log p(\theta) = \log\left[\frac{1}{(2\pi)^{D/2}\cdot|\sigma^2 I|^{1/2}} \cdot \exp\left[-\frac{1}{2}\theta^T(\sigma^2 I)^{-1}\theta\right)\right]$

$= -\log(2\pi)^{D/2}\cdot|\sigma|^D - \frac{1}{2}\theta^T(\sigma^2 I)^{-1}\theta$

$= -\frac{1}{2}\theta^T(\sigma^2 I)^{-1}\theta + \text{const}$

$= \frac{1}{\sigma^2}\cdot -\frac{1}{2}\theta^T\theta + \text{const}$

$= -\frac{1}{2\sigma^2}\cdot\|\theta\|_2^2 + \text{const}$

② (a)

4. **[5pts] Gaussian Discriminant Analysis.** For this question you will build classifiers to label images of handwritten digits. Each image is 8 by 8 pixels and is represented as a vector of dimension 64 by listing all the pixel values in raster scan order. The images are grayscale and the pixel values are between 0 and 1. The labels y are $0, 1, 2, \ldots, 9$ corresponding to which character was written in the image. There are 700 training cases and 400 test cases for each digit; they can be found in the `data` directory in the starter code.

   A skeleton (`q4.py`) is is provided for each question that you should use to structure your code. Starter code to help you load the data is provided (`data.py`). Note: the `get_digits_by_label` function in `data.py` returns the subset of digits that belong to a given class.

   Using maximum likelihood, fit a set of 10 class-conditional Gaussians with a separate, full covariance matrix for each class. Remember that the conditional multivariate Gaussian probability density is given by,

$$p(\mathbf{x} \mid y = k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_\mathbf{k}) = (2\pi)^{-d/2} |\boldsymbol{\Sigma}_\mathbf{k}|^{-1/2} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_\mathbf{k}^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\} \tag{2}$$

   You should take $p(y = k) = \frac{1}{10}$. You will compute parameters μ_{kj} and Σ_k for $k \in (0...9), j \in (1...64)$. You should implement the covariance computation yourself (i.e. without the aid of `np.cov`). For this question, you should write down three significant digits for numerical answers. *Hint: To ensure numerical stability you may have to add a small multiple of the identity to each covariance matrix. For this assignment you should add* $0.01\mathbf{I}$ *to each matrix.*

   (a) **[2pts]** Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. $\frac{1}{N} \sum_{i=1}^{N} \log(p(y^{(i)} \mid \mathbf{x}^{(i)}, \theta))$ on both the train and test set and report it. *Hint: for numerical stability, you will want to use* ***`np.logaddexp`****, as discussed in Lecture 4.*

   (b) **[1pt]** Select the most likely posterior class for each training and test data point as your prediction, and report your accuracy on the train and test set.

   (c) **[2pt]** Redo 4.a and 4.b by assuming that the covariance matrix is diagonal. Compare your answers and provide qualitative explanations.

   The performance is worse compared with full-covariance matrix (lower likelihood and accuracy). Diagonal covariance matrix cannot model dependence between pixels.

Q4

(a)

	average conditional log-likelihood
Train	0.0078
Test	0.0185

(b)

	accuracy
Train	0.8814
Test	0.9727

(c)

	accuracy
Train	0.833
Test	0.825

The accuracy declines if we just use diagonal covariance.
It is because diagonal matrix cannot model the dependence between pixel is thus infeior to the full-covariance matrix.