#1.(a)

(a) **[5pts]** The goal of this question is to show that the AdaBoost algorithm changes the weights in order to force the weak learner to focus on difficult data points. Here we consider the case that the target labels are from the set $\{-1, +1\}$ and the weak learner also returns a classifier whose outputs belongs to $\{-1, +1\}$ (instead of $\{0, 1\}$). Consider the t-th iteration of AdaBoost, where the weak learner is

$$h_t \leftarrow \operatorname*{argmin}_{h \in \mathcal{H}} \sum_{i=1}^{N} w_i \mathbb{I}\{h(\mathbf{x}^{(i)}) \neq t^{(i)}\},$$

the w-weighted classification error is

$$\mathrm{err}_t = \frac{\sum_{i=1}^{N} w_i \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}}{\sum_{i=1}^{N} w_i},$$

and the classifier coefficient is $\alpha_t = \frac{1}{2} \log \frac{1-\mathrm{err}_t}{\mathrm{err}_t}$. (Here, log denotes the natural logarithm.) AdaBoost changes the weights of each sample depending on whether the weak learner h_t classifies it correctly or incorrectly. The updated weights for sample i is denoted by w_i' and is

$$w_i' \leftarrow w_i \exp\left(-\alpha_t t^{(i)} h_t(\mathbf{x}^{(i)})\right).$$

Show that the error w.r.t. $(w_1', \ldots, w_N')$ is exactly $\frac{1}{2}$. That is, show that

$$\mathrm{err}_t' = \frac{\sum_{i=1}^{N} w_i' \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}}{\sum_{i=1}^{N} w_i'} = \frac{1}{2}.$$

Note that here we use the weak learner of iteration t and evaluate it according to the new weights, which will be used to learn the $t+1$-st weak learner. What is the interpretation of this result?

Hints:

i. Start from err_t' and divide the summation to two sets of $E = \{i : h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}$ and its complement $E^c = \{i : h_t(\mathbf{x}^{(i)}) = t^{(i)}\}$.

ii. Note that

$$\frac{\sum_{i \in E} w_i}{\sum_{i=1}^{N} w_i} = \mathrm{err}_t.$$

$$\mathrm{err}_t' = \frac{\sum_{i=1}^{N} w_i' \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\}}{\sum_{i=1}^{N} w_i'}$$

$$a_t = \frac{1}{2} \log \frac{1-\mathrm{err}_t}{\mathrm{err}_t}$$

By #1(b) the update data weights is equivalent to

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp\left(2 a_t \cdot \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\}\right) \cdot c \quad , \text{ where } c \text{ is a constant}$$

if misclassifies $x^{(i)}$, i.e. $h_t(x^{(i)}) \neq t^{(i)}$

then $\mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\} = 1$

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp\left(2 a_t \cdot \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\}\right) \cdot c$$

$$= w_i^{(t)} \cdot c e^{2 a_t}$$

$$= w_i^{(t)} \cdot c \cdot e^{\log \frac{1-\mathrm{err}_t}{\mathrm{err}_t}}$$

$$= w_i^{(t)} \cdot c \cdot \frac{1-\mathrm{err}_t}{\mathrm{err}_t}$$

if classifies $x^{(i)}$ correctly, i.e. $h_t(x^{(i)}) = t^{(i)}$

then $\mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\} = 0$

$$w_i^{(t+1)} = w_i^{(t)} \cdot \exp\left(2 a_t \cdot \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\}\right) \cdot c$$

$$= w_i^{(t)} \cdot c$$

$$w_i^{(t+1)} = \begin{cases} w_i^{(t)} \cdot c \cdot \frac{1-\mathrm{err}_t}{\mathrm{err}_t} & (E) \text{ if misclassifies } x^{(i)}, \text{ i.e. } h_t(x^{(i)}) \neq t^{(i)} \\ w_i^{(t)} \cdot c & (E^c) \text{ if classifies } x^{(i)} \text{ correctly, i.e. } h_t(x^{(i)}) = t^{(i)} \end{cases}$$

$$err'_t = err_{t+1} = \frac{\sum_{i \in E} w_i^{(t+1)}}{\sum_{i=1}^{N} w_i^{(t+1)}} = \frac{\sum_{i \in E} c w_i^{(t)} \cdot \frac{1-err_t}{err_t}}{\sum_{i \in E} c w_i^{(t)} \cdot \frac{1-err_t}{err_t} + \sum_{i \notin E} c w_i^{(t)}}$$

the $(t+1)^{th}$ w-weighted classification error is determined by the t^{th} w-weighted classification error

$$\vdots$$

$$err_2 = \frac{\sum_{i \in E} w_i^{(2)}}{\sum_{i=1}^{N} w_i^{(2)}} = \frac{\sum_{i \in E} c w_i^{(1)} \cdot \frac{1-err_1}{err_1}}{\sum_{i \in E} c w_i^{(1)} \cdot \frac{1-err_1}{err_1} + \sum_{i \notin E} c w_i^{(1)}}$$

the 2^{nd} w-weighted classification error is determined by the 1^{st} w-weighted classification error

Since we initialize sample weights with $w^{(i)} = \frac{1}{N}$ for $i = 1, \dots, N$

statistically, we have 50% probability to classify the data correctly, i.e. $err_1 = \frac{1}{2}$, then $\frac{1-err_1}{err_1} = \frac{1/2}{1/2} = 1$

and we can then get $err_2 = 1/2$. Along with this thread of thinking, $err_t = \frac{1}{2}$ and so $err_{t+1} = \frac{1}{2}$

Hence, we showed that the error w.r.t. $(w'_1, \dots, w'_N)$ is exactly $\frac{1}{2}$

#1. (b)

(b) **[2pts]** Recall from lecture that we can rewrite the 0-1 loss as: $\mathbb{I}[h(x^{(n)}) \neq t^{(n)}] = \frac{1}{2}(1 - h(x^{(n)}) \cdot t^{(n)})$. Use this identity to prove that the weight update from part (a):

$$w_i' \leftarrow w_i \exp\Big(-\alpha_t t^{(i)} h_t(\mathbf{x}^{(i)})\Big).$$

is proportional, up to a constant factor, to weight update:

$$w_i' \leftarrow w_i \exp\Big(2\alpha_t \mathbb{I}\{h_t(\mathbf{x}^{(i)}) \neq t^{(i)}\}\Big).$$

What is the constant factor? Since we normalize the weights when computing err_t, these two updates are equivalent.

We want to show

$$\exp(-\alpha_t t^{(i)} h_t(x^{(i)})) \quad \propto \quad \exp(2\alpha_t \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\})$$

By given identity, we can get

$$2\,\mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\} \;=\; 2 \cdot \tfrac{1}{2}(1 - h_t(x^{(i)}) \cdot t^{(i)}) \;=\; 1 - h_t(x^{(i)}) \cdot t^{(i)}$$

Then, we can get

$$e^{\left[2\alpha_t \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\}\right]} = \left(e^{\left[1 - h_t(x^{(i)}) \cdot t^{(i)}\right]}\right)^{\alpha_t}$$

$$= \left(e \cdot e^{\left[-h_t(x^{(i)}) \cdot t^{(i)}\right]}\right)^{\alpha_t}$$

$$= \underbrace{e^{\alpha_t}}_{\text{constant factor}} \cdot e^{\left[-\alpha_t h_t(x^{(i)}) \cdot t^{(i)}\right]}$$

Hence, we proved $\exp(-\alpha_t t^{(i)} h_t(x^{(i)}))$ is proportional to $\exp(2\alpha_t \mathbb{I}\{h_t(x^{(i)}) \neq t^{(i)}\})$

The constant factor is e^{α_t}

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

In this question, we'll fit a Naïve Bayes model to the MNIST digits using maximum likelihood. 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 the 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 $p(x_j = 1 \,|\, c, \boldsymbol{\theta}, \boldsymbol{\pi}) = \theta_{jc}$. 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

$$p(x_j \,|\, c, \theta_{jc}) = \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}, \tag{0.1}$$

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),

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

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.

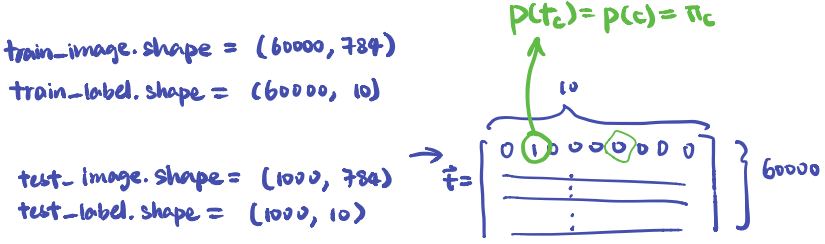
(a) **[3pts]** First, derive the *maximum likelihood estimator* (MLE) for the class-conditional pixel probabilities $\boldsymbol{\theta}$ and the prior $\boldsymbol{\pi}$. Derivations should be rigorous.

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 $p(\mathbf{t}^{(i)} \,|\, \boldsymbol{\pi}) = \Pi_{j=0}^{9} \pi_j^{t_j^{(i)}}$ 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.

train_image.shape = (60000, 784)
train_label.shape = (60000, 10)
test_image.shape = (10000, 784)
test_label.shape = (10000, 10)

$p(t_c) = p(c) = \pi_c$

$\vec{t} =$ 

$t = [t_0 \; t_1 \; \cdots \; t_9]$

x data points

$c \equiv t$ class label

784 pixels → j, 10 classes → c ⇒ every pixel has a binary label c.

θ (matrix, 784×10) prob for each pixel and each class

π (vector, 10×1) with one entry for each class

let D = num of features (here, $D = 784$)

let N = num of data ($N = 60000$ for training data)

The likelihood func is

$$L(\pi, \theta) = \sum_{i=1}^{N} p(c, \vec{x})$$

The log-likelihood is

$$\begin{aligned}
\ell(\pi, \theta) &= \log\left(\sum_{i=1}^{N} p(\vec{t}^{(i)}, \vec{x}^{(i)})\right) \\
&= \sum_{i=1}^{N} \log p(\vec{t}^{(i)}, \vec{x}^{(i)}) \\
&= \sum_{i=1}^{N} \log \left\{ p(\vec{x}^{(i)} \,|\, \vec{t}^{(i)}) \, p(\vec{t}^{(i)}) \right\} \\
&= \sum_{i=1}^{N} \log \left\{ p(\vec{t}^{(i)}) \prod_{j=1}^{D} p(x_j^{(i)} \,|\, \vec{t}^{(i)}) \right\} \\
&= \sum_{i=1}^{N} \left[\log p(\vec{t}^{(i)}) + \sum_{j=1}^{D} \log p(x_j^{(i)} \,|\, \vec{t}^{(i)}) \right] \\
&= \underbrace{\sum_{i=1}^{N} \log p(\vec{t}^{(i)})}_{\text{Bernoulli log-likelihood of labels}} + \underbrace{\sum_{i=1}^{N} \sum_{j=1}^{D} \log p(x_j^{(i)} \,|\, \vec{t}^{(i)})}_{\text{Bernoulli log-likelihood for feature } x_j}
\end{aligned}$$

($\vec{t}^{(i)}$ → $[\,\cdots\,]_{j=1}^{D}$)

For the prior, $p(\vec{t}^{(i)})$, we maximize $\sum_{i=1}^{N} \log p(\vec{t}^{(i)})$

since $\pi_j = \dfrac{\sum_{i=1}^{N} \mathbb{I}(t_j^{(i)} = 1)}{N} = \dfrac{\text{num of data of class } j}{\text{total num of data}}$

then $p(t_j^{(i)} = 1) = \pi_j$

$$\begin{aligned}
p(\vec{t}^{(i)}) &= \prod_{j=0}^{9} p(t_j^{(i)} = 1) \\
&= \prod_{j=0}^{9} \pi_j^{t_j^{(i)}} \\
&= \prod_{j=0}^{8} \pi_j^{t_j^{(i)}} \cdot (\pi_9^{t_9^{(i)}})
\end{aligned}$$

$t_j^{(i)}$: image i belongs to class j

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

$$p(t_j^{(i)} = 1 \,|\, \pi) = \prod_{j=0}^{8} \pi_j^{t_j^{(i)}} \cdot (\pi_9^{t_9^{(i)}})$$

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

$p(x, c) = p(c)\, p(x|c) = p(c) \prod_{j=1}^{} p(x_j|c)$ Pg. 11

$p(c \,|\, \boldsymbol{\pi}) = \pi_c$ and $p(x_j = 1 \,|\, c, \boldsymbol{\theta}, \boldsymbol{\pi}) = \theta_{jc}$.

$p(c) = \pi_c$

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

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

$p(x_j \,|\, c, \theta_{jc}) = \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}$, (0

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

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

$p(x_j = 1 \,|\, c) = \theta_{jc}$, $p(x_j = 0 \,|\, c) = 1 - \theta_{jc}$

$$= \prod_{j=0}^{8} \pi_j^{t_j^{(i)}} \cdot \left(1 - \sum_{j=0}^{8} \pi_j\right)^{t_9^{(i)}}$$

$$\sum_{i=1}^{N} \log p(\vec{t}^{(i)}) = \sum_{i=1}^{N} \log\left[\prod_{j=0}^{8} \pi_j^{t_j^{(i)}} \cdot \left(\pi_9^{t_9^{(i)}}\right)\right]$$

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

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

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

$$= \sum_{i=1}^{N}\sum_{j=0}^{8} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} t_9^{(i)} \log(\pi_9) \quad \text{since} \quad \pi_9 = 1 - \sum_{j=0}^{8} \pi_j$$

$$= \sum_{i=1}^{N}\sum_{j=0}^{8} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N} t_9^{(i)} \log\left(1 - \sum_{j=0}^{8} \pi_j\right) \quad t_9^{(i)} = 1 - \sum_{j=0}^{8} t_j^{(i)}$$

$$= \sum_{i=1}^{N}\sum_{j=0}^{8} t_j^{(i)} \log \pi_j + \sum_{i=1}^{N}\left(1 - \sum_{j=0}^{8} t_j^{(i)}\right) \log\left(1 - \sum_{j=0}^{8} \pi_j\right)$$

$$\frac{\partial}{\partial \pi_0}\left(\sum_{i=1}^{N} \log p(\vec{t}^{(i)})\right) = \frac{\partial}{\partial \pi_0}\left(\sum_{j=0}^{8} \log \pi_j \sum_{i=1}^{N} t_j^{(i)} + \log\left(1 - \sum_{j=0}^{8} \pi_j\right) \sum_{i=1}^{N} t_9^{(i)}\right)$$

$$= \frac{\partial}{\partial \pi_0} \sum_{j=0}^{8} \log \pi_j \sum_{i=1}^{N} t_j^{(i)} + \frac{\partial}{\partial \pi_0} \log\left(1 - \sum_{j=0}^{8} \pi_j\right) \sum_{i=1}^{N} t_9^{(i)}$$

$$= \frac{\partial}{\partial \pi_0}\left(\log \pi_0 \sum_{i=1}^{N} t_0^{(i)} + \sum_{j=1}^{8} \log \pi_j \sum_{i=1}^{N} t_j^{(i)}\right) + \sum_{i=1}^{N} t_9^{(i)} \frac{\partial}{\partial \pi_0} \log\left(1 - \sum_{j=0}^{8} \pi_j\right)$$

$$= \frac{1}{\pi_0} \sum_{i=1}^{N} t_0^{(i)} + \frac{1}{1 - \sum_{j=0}^{8} \pi_j} \cdot (-1) \sum_{i=1}^{N} t_9^{(i)}$$

$$= \frac{1}{\pi_0} \sum_{i=1}^{N} t_0^{(i)} - \frac{1}{1 - \sum_{j=0}^{8} \pi_j} \sum_{i=1}^{N} t_9^{(i)}$$

$$= \frac{1}{\pi_0} \sum_{i=1}^{N} t_0^{(i)} - \frac{1}{\pi_9} \sum_{i=1}^{N} t_9^{(i)}$$

let $f(\pi_0) = 1 - \sum_{j=0}^{8} \pi_j$

$$\frac{\partial \log(f(\pi_0))}{\partial \pi_0} = \frac{1}{f(\pi_0)} \cdot \frac{\partial}{\partial \pi_0}\left(1 - \sum_{j=0}^{8} \pi_j\right)$$

$$= \frac{1}{1 - \sum_{j=0}^{8} \pi_j} \cdot \frac{\partial}{\partial \pi_0}(1 - \pi_0 - \pi_1 - \cdots - \pi_8)$$

$$= \frac{1}{1 - \sum_{j=0}^{8} \pi_j} \cdot (-1)$$

Similarly, the derivative for π_0 also applies to $\pi_1, \pi_2, \ldots, \pi_8$

so, for $0 \le j \le 8$,

$$\frac{\partial}{\partial \pi_j}\left(\sum_{i=1}^{N} \log p(\vec{t}^{(i)})\right) = \frac{1}{\pi_j} \sum_{i=1}^{N} t_j^{(i)} - \frac{1}{1 - \sum_{j=0}^{8} \pi_j}$$

In summary,

$$\frac{\partial}{\partial \pi_j}\left(\sum_{i=1}^{N} \log p(\vec{t}^{(i)})\right) = \begin{cases} \frac{1}{\pi_j} \sum_{i=1}^{N} t_j^{(i)} - \frac{1}{\pi_9} \sum_{i=1}^{N} t_9^{(i)} & \text{for } 0 \le j \le 8 \\ \frac{1}{\pi_j} \sum_{i=1}^{N} t_9^{(i)} & \text{for } j = 9 \end{cases}$$

let $\frac{d}{d\pi_j}\left(\sum_{i=1}^{N} \log p(\vec{t}^{(i)})\right) = 0$

that is, for $0 \le j \le 8$, let $\frac{1}{\pi_j} \sum_{i=1}^{N} t_j^{(i)} = \frac{1}{\pi_9} \sum_{i=1}^{N} t_9^{(i)} \Rightarrow \frac{\pi_j}{\pi_9} = \frac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_9^{(i)}}$

$$\begin{cases} \dfrac{\hat{\pi}_j}{\hat{\pi}_q} = \dfrac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} \Rightarrow \hat{\pi}_j = \hat{\pi}_q \dfrac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} & \text{for } 0 \le j \le 8 \\ \sum_{j=0}^{q} \hat{\pi}_j = 1 \Rightarrow \hat{\pi}_q = 1 - \sum_{j=0}^{8} \hat{\pi}_j \end{cases}$$

$$\sum_{j=0}^{q} \hat{\pi}_j = 1 \Rightarrow \sum_{j=0}^{8} \hat{\pi}_j + \hat{\pi}_q = 1$$

$$\Rightarrow \hat{\pi}_q \frac{\sum_{i=1}^{N} t_0^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} + \hat{\pi}_q \frac{\sum_{i=1}^{N} t_1^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} + \cdots + \hat{\pi}_q \frac{\sum_{i=1}^{N} t_8^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} + \hat{\pi}_q = 1$$

$$\Rightarrow \hat{\pi}_q \left(\frac{\sum_{i=1}^{N} t_0^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} + \frac{\sum_{i=1}^{N} t_1^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} + \cdots + \frac{\sum_{i=1}^{N} t_8^{(i)}}{\sum_{i=1}^{N} t_q^{(i)}} \right) = 1$$

$$\Rightarrow \hat{\pi}_q = \frac{1}{\frac{\sum_{j=0}^{q} \left(\sum_{i=1}^{N} t_j^{(i)} \right)}{\sum_{i=1}^{N} t_q^{(i)}}} = \frac{\sum_{i=1}^{N} t_q^{(i)}}{\sum_{j=0}^{q} \left(\sum_{i=1}^{N} t_j^{(i)} \right)}$$

$$\vec{\pi} = \begin{bmatrix} \pi_0 \\ \pi_1 \\ \vdots \\ \pi_8 \\ \pi_q \end{bmatrix}$$

$$\Rightarrow \hat{\vec{\pi}} = \begin{bmatrix} \hat{\pi}_0 \\ \hat{\pi}_1 \\ \vdots \\ \hat{\pi}_8 \\ \hat{\pi}_q \end{bmatrix} = \frac{\hat{\pi}_q}{\sum_{i=1}^{N} t_q^{(i)}} \begin{bmatrix} \hat{\pi}_0 \sum_{i=1}^{N} t_0^{(i)} / \hat{\pi}_q \\ \hat{\pi}_1 \sum_{i=1}^{N} t_1^{(i)} / \hat{\pi}_q \\ \vdots \\ \hat{\pi}_8 \sum_{i=1}^{N} t_8^{(i)} / \hat{\pi}_q \\ \hat{\pi}_q \sum_{i=1}^{N} t_q^{(i)} / \hat{\pi}_q \end{bmatrix} = \frac{1}{\sum_{j=0}^{q} \left(\sum_{i=1}^{N} t_j^{(i)} \right)} \begin{bmatrix} \sum_{i=1}^{N} t_0^{(i)} \\ \sum_{i=1}^{N} t_1^{(i)} \\ \vdots \\ \sum_{i=1}^{N} t_8^{(i)} \\ \sum_{i=1}^{N} t_q^{(i)} \end{bmatrix} = \frac{1}{\sum_{j=0}^{q} \left(\sum_{i=1}^{N} t_j^{(i)} \right)} \begin{bmatrix} \sum_{i=1}^{N} \mathbb{1}(t_0^{(i)} = 1) \\ \sum_{i=1}^{N} \mathbb{1}(t_1^{(i)} = 1) \\ \vdots \\ \sum_{i=1}^{N} \mathbb{1}(t_8^{(i)} = 1) \\ \sum_{i=1}^{N} \mathbb{1}(t_q^{(i)} = 1) \end{bmatrix}$$

$$= \boxed{\frac{1}{N} \begin{bmatrix} \sum_{i=1}^{N} \mathbb{1}(t_0^{(i)} = 1) \\ \sum_{i=1}^{N} \mathbb{1}(t_1^{(i)} = 1) \\ \vdots \\ \sum_{i=1}^{N} \mathbb{1}(t_8^{(i)} = 1) \\ \sum_{i=1}^{N} \mathbb{1}(t_q^{(i)} = 1) \end{bmatrix}}$$

To maximize $\sum_{i=1}^{N} \log p(x_j^{(i)} \mid \vec{t}^{(i)})$

$\theta_{jc} = p(x_j^{(i)} = 1 \mid c)$

$p(x_j^{(i)} \mid c) = \theta_{jc}^{x_j^{(i)}} (1 - \theta_{jc})^{1 - x_j^{(i)}}$

θ_{jc} : num of class-c pixel in a pic / num of class-c pic in dataset

$$\sum_{i=1}^{N} \log p(x_j^{(i)} \mid c^{(i)})$$

$$= \sum_{i=1}^{N} \sum_{k=0}^{9} c^{(i)} \left\{ \log \theta_{jk}^{x_j^{(i)}} + \log (1 - \theta_{jk})^{1 - x_j^{(i)}} \right\}$$

$$= \sum_{i=1}^{N} \sum_{k=0}^{9} c^{(i)} \left\{ x_j^{(i)} \log \theta_{jk} + (1 - x_j^{(i)}) \log (1 - \theta_{jk}) \right\}$$

$$\frac{\partial}{\partial \theta_{jc}} \left(\sum_{i=1}^{N} \log p(x_j^{(i)} \mid c^{(i)}) \right)$$

$$= \frac{\partial}{\partial \theta_{jc}} \left(\sum_{i=1}^{N} \sum_{k=0}^{9} c^{(i)} \left\{ x_j^{(i)} \log \theta_{jk} + (1 - x_j^{(i)}) \log (1 - \theta_{jk}) \right\} \right)$$

$$= \sum_{i=1}^{N} \sum_{k=0}^{9} c^{(i)} \left\{ x_j^{(i)} \frac{\partial}{\partial \theta_{jc}} \log \theta_{jk} + (1 - x_j^{(i)}) \frac{\partial}{\partial \theta_{jc}} \log (1 - \theta_{jk}) \right\}$$

$$\Rightarrow \boxed{\hat{\theta}_{jc} = \frac{\sum_{i=1}^{N} \mathbb{I}[x_j^{(i)} = 1 \ \& \ c^{(i)} = c]}{\sum_{i=1}^{N} \mathbb{I}[c^{(i)} = c]}} \quad \text{where } c \in [0, 9]$$

(b) [1pt] Derive the log-likelihood $\log p(\mathbf{t}|\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\pi})$ for a single training image.

$$p(\vec{t}|\vec{x}) = \frac{p(\vec{t},\vec{x})}{p(\vec{x})} = \frac{p(\vec{t})\,p(\vec{x}|\vec{t})}{\sum_{i=1}^{q} p(\vec{t}^{(i)})\,p(\vec{x}|\vec{t}^{(i)})} = \frac{p(\vec{t})\prod_{j=1}^{D} p(x_j|\vec{t})}{\sum_{i=1}^{q} p(\vec{t}^{(i)})\prod_{j=1}^{D} p(x_j|\vec{t}^{(i)})} \qquad D=784$$

$$p(c|\vec{x}) = \frac{p(c,\vec{x})}{p(\vec{x})} = \frac{p(c)\,p(\vec{x}|c)}{\sum_{c'=0}^{q} p(c')\,p(\vec{x}|c')} = \frac{p(c)\prod_{j=1}^{D} p(x_j|c)}{\sum_{c'=0}^{q} p(c')\prod_{j=1}^{D} p(x_j|c')} = \frac{\pi_c \prod_{j=1}^{D} \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}}{\sum_{c'=0}^{q}\pi_{c'}\prod_{j=1}^{D} p(x_j|c')}$$

$$\log p(c|\vec{x}) = \log \frac{\pi_c \prod_{j=1}^{D} \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)}}{\sum_{c'=0}^{q}\pi_{c'}\prod_{j=1}^{D} p(x_j|c')}$$

$$= \log \pi_c + \log \prod_{j=1}^{D} \theta_{jc}^{x_j}(1-\theta_{jc})^{(1-x_j)} - \log \sum_{c'=0}^{q}\pi_{c'}\prod_{j=1}^{D} p(x_j|c')$$

$$= \log \pi_c + \sum_{j=1}^{D}\left[\log \theta_{jc}^{x_j} + \log(1-\theta_{jc})^{(1-x_j)}\right] - \sum_{c'=0}^{q}\left[\log \pi_{c'} + \log \sum_{j=1}^{D} p(x_j|c')\right]$$

$$= \log \pi_c + \sum_{j=1}^{D}\left[x_j \log \theta_{jc} + (1-x_j)\log(1-\theta_{jc})\right] - \sum_{c'=0}^{q}\log \pi_{c'} - \sum_{c'=0}^{q}\sum_{j=1}^{D}\log p(x_j|c')$$

$$= \log \pi_c + \sum_{j=1}^{784}\left[x_j \log \theta_{jc} + (1-x_j)\log(1-\theta_{jc})\right] - \sum_{c'=0}^{q}\sum_{j=1}^{784}\log p(x_j|c')$$

1×1 1×10 1×1 1×10 1×10

$$\sum_{c'=0}^{q}\log \pi_{c'} = \log \sum_{c'=0}^{q}\pi_{c'} = \log 1 = 0$$

since $\hat{\pi}_c = \frac{\sum_{i=1}^{N}\mathbb{I}(t_c^{(i)}=1)}{N}$

$$\hat{\theta}_{jc} = \frac{\sum_{i=1}^{N}\mathbb{I}[x_j^{(i)}=1 \;\&\; c^{(i)}=c]}{\sum_{i=1}^{N}\mathbb{I}[c^{(i)}=c]} \quad \text{where } c\in[0,9]$$

$$\log p(c|\vec{x}) = \log \hat{\pi}_c + \sum_{j=1}^{784}\left[x_j \log \hat{\theta}_{jc} + (1-x_j)\log(1-\hat{\theta}_{jc})\right] - \sum_{c'=0}^{q}\sum_{j=1}^{784}\log p(x_j|c')$$

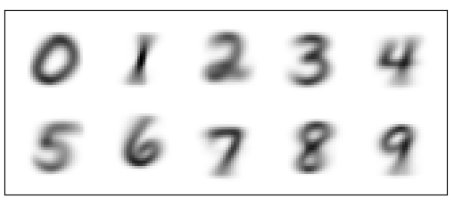
$$= \log \frac{\sum_{i=1}^{N}\mathbb{I}(t_c^{(i)}=1)}{N} + \sum_{j=1}^{784}\left[x_j \log \frac{\sum_{i=1}^{N}\mathbb{I}[x_j^{(i)}=1 \;\&\; c^{(i)}=c]}{\sum_{i=1}^{N}\mathbb{I}[c^{(i)}=c]} + (1-x_j)\log\left(1-\frac{\sum_{i=1}^{N}\mathbb{I}[x_j^{(i)}=1 \;\&\; c^{(i)}=c]}{\sum_{i=1}^{N}\mathbb{I}[c^{(i)}=c]}\right)\right] - \sum_{c'=0}^{q}\sum_{j=1}^{784}\log p(x_j|c')$$

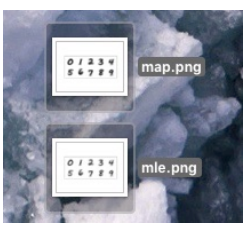
(c) [**3pt**] Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MLE, and try to report the average log-likelihood per data point $\frac{1}{N}\Sigma_{i=1}^{N} \log p(\mathbf{t}^{(i)}|\mathbf{x}^{(i)}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}})$, using Equation (0.1). What goes wrong? (it's okay if you can't compute the average log-likelihood here).

```
In[2]: runfile('/Users/ChangyanXu/Desktop/naive_bayes.py', wdir='/Users/ChangyanXu/Desktop')
Backend MacOSX is interactive backend. Turning interactive mode on.
/Users/ChangyanXu/Desktop/naive_bayes.py:178: RuntimeWarning: divide by zero encountered in log
  x_dot_log_theta = image.dot(np.log(theta))
```

log(0) will result in an negative infinity.

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





(e) **[2pt]** Derive the *Maximum A posteriori Probability* (MAP) estimator for the class-conditional pixel probabilities $\boldsymbol{\theta}$, using a Beta(3, 3) prior on each θ_{jc}. Hint: it has a simple final form, and you can ignore the Beta normalizing constant.

$$\begin{aligned}\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta} \mid \mathcal{D}) \\ &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta}, \mathcal{D}) \\ &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta})\, p(\mathcal{D} \mid \boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \; \log p(\boldsymbol{\theta}) + \log p(\mathcal{D} \mid \boldsymbol{\theta})\end{aligned}$$

$\beta(a=3, b=3)$

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

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

$$= \arg\max_{\theta} \; p(\theta)\; p(D \mid \theta)$$

$$= \arg\max_{\theta} \; \log\left[p(\theta)\; p(D \mid \theta) \right]$$

$$\hat{\theta}_{jc,\,MAP} = \arg\max_{\theta_{jc}} \prod_{i=1}^{N} \pi_c \prod_{j=1}^{784} \theta_{jc}^{x_j^{(i)}} (1-\theta_{jc})^{1-x_j^{(i)}} \; \theta_{jc}^{a-1} (1-\theta_{jc})^{b-1}$$

$$= \arg\max_{\theta_{jc}} \left(\theta_{jc}^{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} x_j^{(i)} + (a-1)} \; (1-\theta_{jc})^{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} (1-x_j^{(i)}) + (b-1)} \right)$$

let $A = \sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} x_j^{(i)} + (a-1)$, $B = \sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} (1-x_j^{(i)}) + (b-1)$

$$\frac{d}{d\theta_{jc}} \log\left[\theta_{jc}^{A} (1-\theta_{jc})^{B} \right] = \frac{d}{d\theta_{jc}} \left[A \log\theta_{jc} + B \log(1-\theta_{jc}) \right]$$

$$= \frac{A}{\theta_{jc}} - \frac{B}{1-\theta_{jc}} = 0$$

$$\hat{\theta}_{jc} = \frac{A}{A+B} = \frac{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} x_j^{(i)} + (a-1)}{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} + (a+b-2)}$$

Since $\beta(a=3, b=3)$

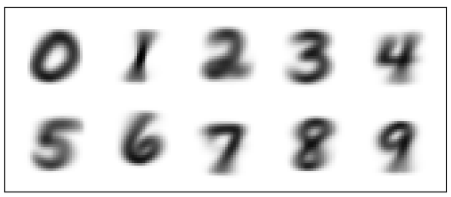
$$\hat{\theta}_{jc,\,MAP} = \frac{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} x_j^{(i)} + (a-1)}{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} + (a+b-2)}$$

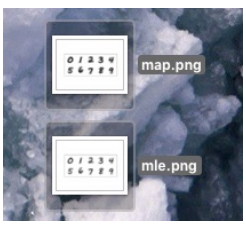
$$= \frac{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} x_j^{(i)} + 2}{\sum_{i=1}^{N} \mathbb{I}\{c^{(i)}=c\} + 4}$$

(f) [**2pt**] Fit the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ using the training set with MAP estimators from 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 MLE is  nan
Average log-likelihood for MAP is  0.03364712014983577
Training accuracy for MAP is  0.0
Test accuracy for MAP is  0.0
```

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





3. **[4pts] Generating from a Naïve Bayes Model**

Defining a joint probability distribution over the data lets us generate new data, and also lets us answer all sorts of queries about the data. This is why these models are called *Generative Models*. We will use the Naïve Bayes model trained in previous question to generate data.

(a) **[1pt]** True or false: Given this model's assumptions, any two pixels x_i and x_j where $i \neq j$ are independent given c.

if x_i and x_j are independent,

$p(x_i)\, p(x_j) = p(x_i, x_j)$

False. Should be "conditional independent", rather than "independent"

(b) **[1pt]** True or false: Given this model's assumptions, any two pixels x_i and x_j where $i \neq j$ are independent after marginalizing over c.

True

$$p(x_i, c) = p(c \mid x_i) \cdot p(x_i)$$

$$p(x_j, c) = p(c \mid x_j) \cdot p(x_j)$$

$$p(x_i \mid c) = \frac{p(x_i, c)}{p(c)} \qquad p(x_j \mid c) = \frac{p(x_j, c)}{p(c)}$$

$$p(x_i \mid c)\, p(x_j \mid c) = \frac{p(x_i, c)\; p(x_j, c)}{p(c)\; p(c)} = \frac{p(c \mid x_i) \cdot p(x_i)\;\; p(c \mid x_j) \cdot p(x_j)}{p(c)\; p(c)}$$

(c) **[2pts]** Using the parameters fit using MAP in Question 1, produce random image samples from the model. That is, randomly sample and plot 10 binary images from the marginal distribution $p(\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\pi})$. Hint: To sample from $p(\mathbf{x} \mid \boldsymbol{\theta}, \boldsymbol{\pi})$, first sample random variable c from $p(c \mid \boldsymbol{\pi})$ using `np.random.choice`, then depending on the value of c, sample x_j from $p(x_j \mid c, \hat{\theta}_{jc})$ for $j = 1, ..., 784$ using `np.random.binomial(1,..)`. These functions can take matrix probabilities as input, so your solution to this part should be a few lines of code.