CSC311 Assignment3

Benson Li

March 30, 2023

1 Back-propagation

1.1 (a)

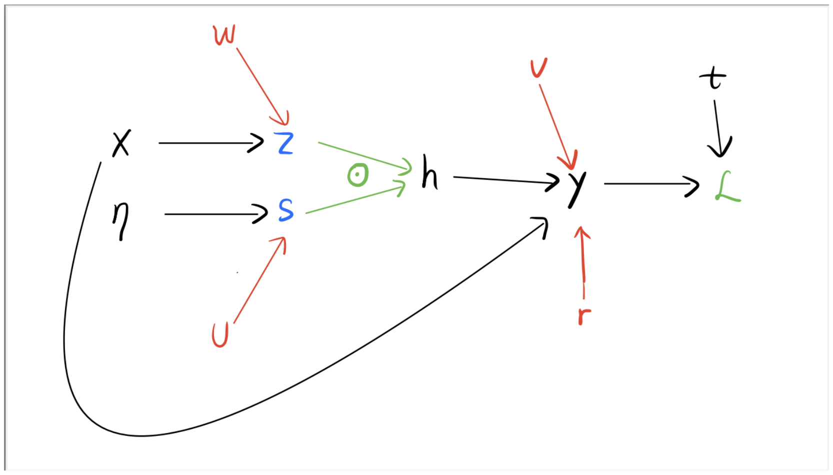


Figure 1: Computation graph

1.2 (b)

$$\begin{aligned}
\frac{d\sigma(x)}{dx} &= \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right) && (1)\\
&= \frac{-1}{(1+e^{-x})^2}\cdot\frac{d}{dx}(1+e^{-x}) && (2)\\
&= \frac{-1}{(1+e^{-x})^2}\cdot(-e^{-x}) && (3)\\
&= \frac{e^{-x}}{(1+e^{-x})^2} && (4)\\
&= \frac{1}{1+e^{-x}}\cdot\frac{e^{-x}}{1+e^{-x}} && (5)\\
&= \sigma(x)(1-\sigma(x)) && (6)
\end{aligned}$$

1.3 (c)

First, we compute the error signal for the output y:

$$\frac{\partial L}{\partial y} = \frac{t}{y} - \frac{1-t}{1-y}$$

Then, we compute the error signal for the hidden layer activations h:

$$\frac{\partial L}{\partial h} = \left(\frac{\partial L}{\partial y}\right) \cdot \left(\frac{\partial y}{\partial h}\right) = \left(\frac{t}{y} - \frac{1-t}{1-y}\right) \cdot \sigma(v^\top h + r^\top x)(1 - \sigma(v^\top h + r^\top x)) \cdot v^\top$$

Next, we compute the error signal for z and s, since $a \odot b$ can be viewed as $diag(a)diag(b)$, we then have:

$$\frac{\partial L}{\partial z} = \left(\frac{\partial L}{\partial h}\right) \cdot \left(\frac{\partial h}{\partial z}\right) = \left(\frac{\partial L}{\partial h}\right) \cdot diag(s)$$

$$\frac{\partial L}{\partial s} = \left(\frac{\partial L}{\partial h}\right) \cdot \left(\frac{\partial h}{\partial s}\right) = \left(\frac{\partial L}{\partial h}\right) \cdot diag(z)$$

Then, we compute the error signals for the weight matrices W and U:

$$\frac{\partial L}{\partial W} = \left(\frac{\partial L}{\partial z}\right) \cdot \left(\frac{\partial z}{\partial W}\right) = \left(\frac{\partial L}{\partial z}\right) \cdot x^\top$$

$$\frac{\partial L}{\partial U} = \left(\frac{\partial L}{\partial s}\right) \cdot \left(\frac{\partial s}{\partial U}\right) = \left(\frac{\partial L}{\partial s}\right) \cdot \eta^\top$$

Next, we compute the error signals for the bias vectors r and v:

$$\frac{\partial L}{\partial r} = \left(\frac{\partial L}{\partial y}\right) \cdot \left(\frac{\partial y}{\partial r}\right) = \left(\frac{\partial L}{\partial y}\right) \cdot \sigma(v^\top h + r^\top x)(1 - \sigma(v^\top h + r^\top x)) \cdot x^\top$$

$$\frac{\partial L}{\partial v} = \left(\frac{\partial L}{\partial y}\right) \cdot \left(\frac{\partial y}{\partial v}\right) = \left(\frac{\partial L}{\partial y}\right) \cdot \sigma(v^\top h + r^\top x)(1 - \sigma(v^\top h + r^\top x)) \cdot h^\top$$

Finally, we compute the error signals for the input vectors x and η:

$$\frac{\partial L}{\partial \eta} = \left(\frac{\partial L}{\partial s}\right) \cdot \left(\frac{\partial s}{\partial \eta}\right) = \left(\frac{\partial L}{\partial s}\right) \cdot U^\top$$

$$\begin{aligned}\frac{\partial L}{\partial x} &= \left(\frac{\partial L}{\partial y}\right) \cdot \left(\frac{\partial y}{\partial x}\right) + \left(\frac{\partial L}{\partial z}\right) \cdot \left(\frac{\partial z}{\partial x}\right) \\ &= \left(\frac{\partial L}{\partial y}\right) \cdot \sigma(v^\top h + r^\top x)(1 - \sigma(v^\top h + r^\top x)) \cdot r^\top + \left(\frac{\partial L}{\partial z}\right) \cdot W^\top\end{aligned}$$

2 Fitting a Naive Bayes Model

2.1 (a)

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

$$\ell(\theta,\pi|X) = \log(p(c|\pi)) + \sum_{j=1}^{784}\log(p(x_j|c,\theta_{jc}))$$

For π_{MLE}:

$$\log(p(c|\pi)) = \sum_{i=1}^{N}\left(\log(1-\sum_{j=0}^{8}\pi_j)\, t_9^{(i)} + \sum_{j=0}^{8}\log(\pi_j)\cdot t_j^{(i)}\right)$$

For $j \in [0,8]$

$$\frac{\partial \log(p(c|\pi))}{\partial \pi_j} = \sum_{i=1}^{N}\left(\frac{-t_9^{(i)}}{1-\sum_{j=1}^{8}\pi_j} + \frac{t_j^{(i)}}{\pi_j}\right) \overset{set}{=} 0$$

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

$$\frac{\hat{\pi}_j}{\hat{\pi}_9} = \frac{\sum_{i=1}^{N} t_j^{(i)}}{\sum_{i=1}^{N} t_9^{(i)}} = \frac{\#\text{ label } j}{\#\text{ label } 9}$$

Since $\sum_{i=0}^{9}\pi_i = 1$, we then have:

$$\frac{1}{\hat{\pi}_9} = \frac{\sum_{i=0}^{9}\hat{\pi}_j}{\hat{\pi}_9} = \frac{N}{\#\text{label } 9} \Rightarrow \hat{\pi}_j = \frac{\#\text{ of label } j}{N}$$

For θ_{MLE}

$$\sum_{j=1}^{784}\log(p(x_j|c,\theta_{jc})) = \sum_{i=1}^{N}\sum_{j=1}^{784} x_j^{(i)}\log(\theta_j) + (1-x_j^{(i)})\log(1-\theta_j)$$

$$= \sum_{i=1}^{N}\sum_{j=1}^{784}\sum_{c=0}^{9} t_c\left(x_j^{(i)}\log(\theta_{jc}) + (1-x_j^{(i)})\log(1-\theta_{jc})\right)$$

$$\frac{\partial \sum_{j=1}^{784}\log(p(x_j|c,\theta_{jc}))}{\partial\theta_{jc}} = \sum_{i=1}^{N} t_c^{(i)}\left(\frac{x_j^{(i)}}{\theta_{jc}} - \frac{1-x_j^{(i)}}{1-\theta_{jc}}\right)$$

$$= \sum_{j=1}^{N} t_c^{(i)}\frac{x_j^{(i)}-\theta_{jc}}{\theta_{jc}(1-\theta_{jc})} \overset{set}{=} 0$$

$$\sum_{i=1}^{N} t_c^{(i)} x_j^{(i)} = \theta_{jc}\sum_{i=1}^{N} t_c^{(i)} \Rightarrow$$

$$\hat{\theta}_{jc} = \frac{\sum_{i=1}^{N} t_c^{(i)} x_j^{(i)}}{\sum_{i=1}^{N} t_c^{(1)}} = \frac{\#\text{ of feature j with lable c}}{\#\text{ of label c}}$$

Figure 2: Derive the maximum likelihood estimator

2.2 (b)

$$\log(t \mid x, \theta, \pi) = \log\left(\frac{P(t \mid \theta, \pi)\, P(x \mid t, \theta, \pi)}{P(x \mid \theta, \pi)}\right)$$

$$= \log\left(\frac{P(t \mid \pi)\, P(x \mid t_c, \theta_c)}{P(x \mid \theta_c)}\right)$$

$$= \underbrace{\log(p(t \mid \theta, \pi))}_{(1)} + \underbrace{\log(p \mid t_c, \theta_c)}_{(2)} - \underbrace{\log(p(x \mid \theta_c)}_{(3)}$$

$$(1)\ \log(p(t \mid \theta, \pi)) = \log(\pi_c)$$

$$(2)\ \log(p(x \mid t_c, \theta_c)) = \log\left(\prod_{j=1}^{784} \theta_{jc}^{x_j} (1-\theta_j)^{1-x_j}\right)$$

$$= \sum_{j=1}^{784} x_j \log(\theta_{jc}) + (1-x_j)\log(1-\theta_{jc})$$

$$(3)\ \log(p(x \mid \theta_c)) = \log\left(\sum_{c=0}^{9} \pi_c \underbrace{\boxed{\prod_{j=1}^{784} \theta_{jc}^{x_j} (1-\theta_{jc})^{(1-x_j)}}}_{(4)}\right)$$

$$(4)\ \prod_{i=1}^{784} \theta_{jc}^{x_j} (1-\theta_{jc})^{(1-x_j)} = e^{\log\left(\prod_{i=1}^{784} \theta_{jc}^{x_j} (1-\theta_{jc})^{(1-x_j)}\right)}$$

$$= e^{\sum_{i=1}^{784} x_j \log(\theta_{jc}) + (1-x_j)\log(1-\theta_{jc})}$$

Figure 3: Derive the log-likelihood

2.3 (c)

The output is:

```
Average log-likelihood for MLE is  nan
```

Average log-likelihood for MLE is not defined.
Possible Reason: $\hat{\theta_{jc}}$ is zero, causes $log(\hat{\theta_{jc}})$ to be undefined.

2.4 (d)

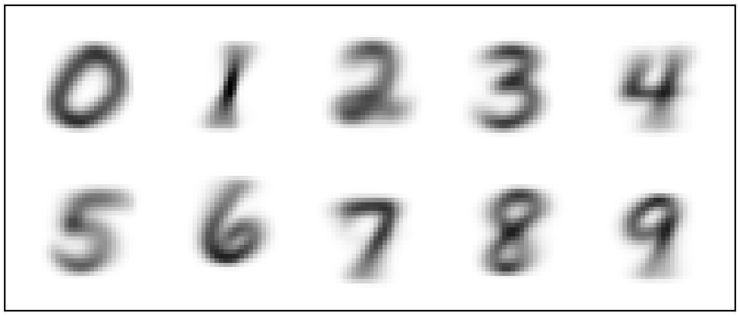
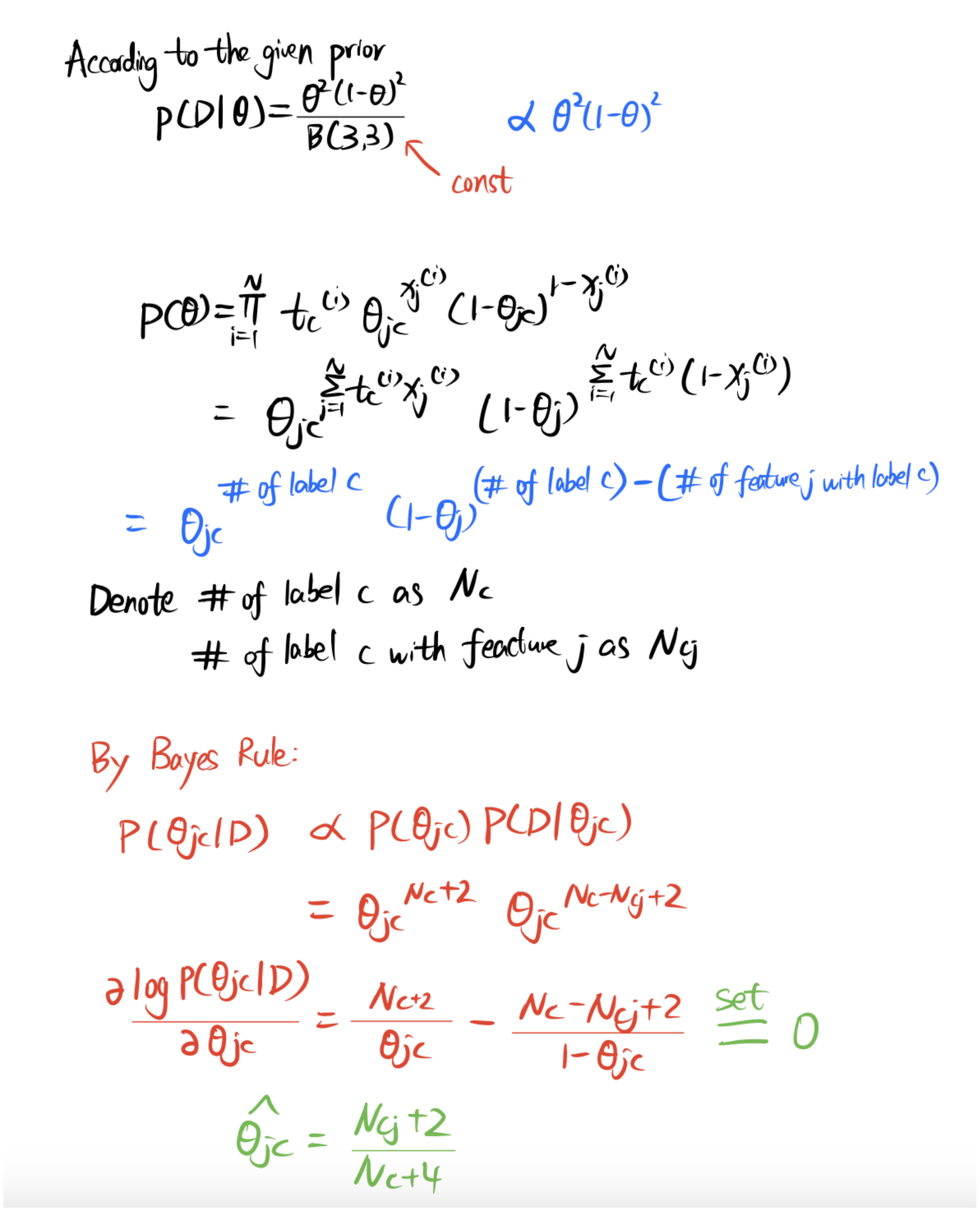


Figure 4: MLE images

2.5 (e)



According to the given prior

$$P(D|\theta) = \frac{\theta^2(1-\theta)^2}{\underbrace{B(3,3)}_{\text{const}}} \quad \propto \theta^2(1-\theta)^2$$

$$P(\theta) = \prod_{i=1}^{N} t_c^{(i)}\, \theta_{jc}^{x_j^{(i)}} (1-\theta_{jc})^{1-x_j^{(i)}}$$

$$= \theta_{jc}^{\sum_{i=1}^{N} t_c^{(i)} x_j^{(i)}} (1-\theta_j)^{\sum_{i=1}^{N} t_c^{(i)} (1-x_j^{(i)})}$$

$$= \theta_{jc}^{\#\text{ of label } c} (1-\theta_j)^{(\#\text{ of label } c) - (\#\text{ of feature } j \text{ with label } c)}$$

Denote # of label c as N_c

# of label c with feacture j as N_{cj}

By Bayes Rule:

$$P(\theta_{jc}|D) \propto P(\theta_{jc})\, P(D|\theta_{jc})$$

$$= \theta_{jc}^{N_c+2}\ \theta_{jc}^{N_c-N_{cj}+2}$$

$$\frac{\partial \log P(\theta_{jc}|D)}{\partial \theta_{jc}} = \frac{N_c+2}{\theta_{jc}} - \frac{N_c-N_{cj}+2}{1-\theta_{jc}} \overset{\text{set}}{=} 0$$

$$\hat{\theta}_{jc} = \frac{N_{cj}+2}{N_c+4}$$

Figure 5: Derive the MAP estimator

2.6 (f)

The output is:

```
Average log-likelihood for MAP is  -3.3570631378602873
Training accuracy for MAP is  0.9670433333333334
Test accuracy for MAP is  0.9632
```

2.7 (g)

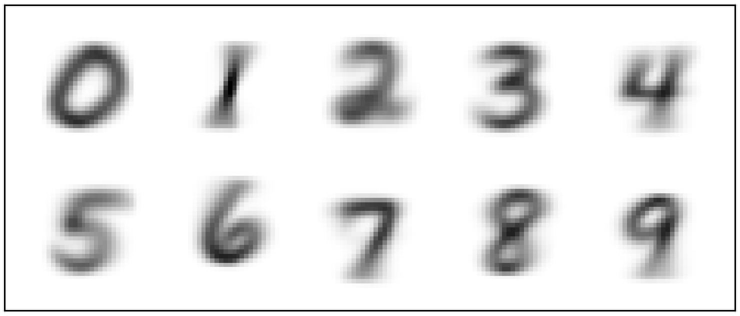


Figure 6: MAP images

2.8 (h)

Advantage of naive bayes for this problem:

Naive Bayes is a simple and easy to implement and understand. It's a fast algorithm that is computationally efficient for this task.

Disadvantage of naive bayes for this problem:

Naive Bayes assumes that all features are independent of each other, which is not always true for image data. Pixels in an image are often correlated with each other. If we move the main part of a image to its left by just one pixel, the result would be completely different. It would better to use convolutional neural network that can extract features from images.

3 Logistic Regression with Gaussian Prior.

3.1 (a)

The log-likelihood of the parameter vector θ for logistic regression is given by:

$$\begin{aligned}\ell(\theta) &= \sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)},\theta)\\ &= \sum_{i=1}^{N} y^{(i)} \log \frac{1}{1+\exp(-x^{(i)T}\theta)} + (1-y^{(i)}) \log \frac{\exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)}\\ &= \sum_{i=1}^{N} y^{(i)} x^{(i)T}\theta - \log(1+\exp(x^{(i)T}\theta)).\end{aligned}$$

To optimize the resulting log-likelihood, we can use gradient ascent algorithm. We can iteratively update the parameter vector until convergence. The gradient of the log-likelihood with respect to θ is given by:

$$\begin{aligned}\nabla_\theta \ell(\theta) &= \sum_{i=1}^{N} (y^{(i)} - p(y^{(i)}=1|x^{(i)},\theta))x^{(i)}\\ &= \sum_{i=1}^{N} (y^{(i)} - \frac{1}{1+\exp(-x^{(i)T}\theta)})x^{(i)}\end{aligned}$$

3.2 (b)

The prior distribution over the parameters θ is assumed to be a Gaussian distribution with zero mean and variance σ^2. Therefore, the prior density function $p(\theta)$ is given by:

$$p(\theta) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{1}{2\sigma^2}||\theta||^2\right)$$

The logarithm of this function is:

$$\begin{aligned}\log p(\theta) &= -\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}||\theta||^2\\ &= -\frac{1}{2\sigma^2}||\theta||^2 + \text{const}\end{aligned}$$

where const is a constant term that does not depend on θ.
Therefore, The MAP log-likelihood of the parameter vector θ for logistic regression with the Gaussian prior is given by:

$$\begin{aligned}\ell_{\text{MAP}}(\theta) &= \sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)},\theta) + \log p(\theta)\\ &= \sum_{i=1}^{N} y^{(i)} \log \frac{1}{1+\exp(-x^{(i)T}\theta)} + (1-y^{(i)}) \log \frac{\exp(-x^{(i)T}\theta)}{1+\exp(-x^{(i)T}\theta)} \quad - \frac{1}{2\sigma^2}||\theta||^2 + \text{const}\\ &= \sum_{i=1}^{N} y^{(i)} x^{(i)T}\theta - \log(1+\exp(x^{(i)T}\theta)) \quad - \frac{1}{2\sigma^2}||\theta||^2 + \text{const}.\end{aligned}$$

4 Gaussian Discriminant Analysis.

All of implementation can be refered to file q4.py.

4.1 (a)

Let X be a 64-dimensional feature vector, and y be the corresponding label. The probability density function (PDF) of the feature vector given the class k can be written as:

$$p(X|y = k; \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X - \mu_k)^T \Sigma_k^{-1}(X - \mu_k)\right)$$

where μ_k is the mean vector of class k, Σ_k is the covariance matrix of class k, and d=64 is the dimensionality of the feature vector.

The maximum likelihood estimation of the mean vector and covariance matrix for class k can be obtained by maximizing the log-likelihood function:

$$\ell(\mu_k, \Sigma_k) = \sum_{i=1}^{n} \log p(X^{(i)}|y^{(i)} = k; \mu_k, \Sigma_k)$$

where $X^{(i)}$ is the ith feature vector and $y^{(i)}$ is the corresponding label. Taking the derivative of $\ell(\mu_k, \Sigma_k)$ with respect to μ_k and Σ_k and setting them to zero, we get:

$$\mu_k = \frac{1}{n_k} \sum_{i=1}^{n} 1_{y^{(i)}=k} X^{(i)}$$

$$\Sigma_k = \frac{1}{n_k} \sum_{i=1}^{n} 1_{y^{(i)}=k} (X^{(i)} - \mu_k)(X^{(i)} - \mu_k)^T$$

where n_k is the number of training examples with label k.

We have added $0.01\mathbf{I}$ to each covariance matrix to ensure numerical stability.

After computing the MLE parameters $\mu_k, \Sigma_k{}_{k=0}^{9}$, we can compute the log-likelihood for a example X as:

$$\begin{aligned}\ell(\mu_k, \Sigma_k) &= \sum_{i=1}^{n} \log p(X^{(i)}|y^{(i)} = k; \mu_k, \Sigma_k) \\ &= \sum_{i=1}^{n} -\frac{1}{2}d \cdot log(2\pi) - \frac{1}{2}|\Sigma_k| - \frac{1}{2}(X - \mu_k)^T \Sigma_k^{-1}(X - \mu_k)\end{aligned}$$

Using Bayes rule, we can compute the posterior probabilities for each class:

$$p(y = k|X) = \frac{p(X|y = k)p(y = k)}{\sum_{j=0}^{9} p(X|y = j)p(y = j)}$$

Finally, we can make a prediction by choosing the class with the highest posterior probability:

$$\hat{y} = \arg \max_{k \in 0,1,\ldots,9} p(y = k|X)$$

To compute the average conditional log-likelihood, we can sum over the log-likelihoods of all examples and divide by the total number of examples:

$$\frac{1}{N} \sum_{i=1}^{N} \log p(y^{(i)}|X^{(i)}, \theta)$$

where θ represents the parameters $\mu_k, \Sigma_k{}_{k=0}^{9}$, and N is the number of examples.

4.2 (b)

I got the following output:

```
Average Train LL: -2.4272095296626754
Average Test LL: -2.4992582962493013
Train Accuracy:  0.9814285714285714
Test Accuracy:  0.97275
```

4.3 (c)

I got the following output under the assuption that the covariance matrix is diagonal:

```
Average Train LL: -3.5333505152668367
Average Test LL: -3.589845459749884
Train Accuracy:  0.85
Test Accuracy:  0.84
```

Analyze:
According to the output, the model without the assumption of diagonal covariance matrix performs better on both the train and test sets. Specifically, the average conditional log-likelihood is higher for the non-diagonal covariance matrix model, indicating that it is able to better capture the distribution of the data. The accuracy is also higher for the non-diagonal covariance matrix model on both train and test sets.
The reason is that, diagonal covariance matrix cannot model dependence between pixels. If we move the main part of a image to its left by just one pixel, the result would be completely different for the diagonal covariance matrix.