CSC 411 Assignment 2

1. 1. $P(y=k|x,\mu,\sigma) = \frac{P(x|y=k,\mu,\sigma)\,P(y=k)}{P(x|\mu,\sigma)}$ ... ①

Use the law of total probability we have

$P(x|\mu,\sigma) = \sum_{j=1}^{K} P(x|y=j,\mu,\sigma)\,P(y=j)$ $(j\in[1,k]\ \&\ i\in\mathbb{Z})$

$$① = \frac{\partial_k \left(\prod_{i=1}^{D} 2\pi\sigma_i^2\right)^{-\frac{1}{2}} \exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i-\mu_{ki})^2\right\}}{\sum_{j=1}^{K}\left[\partial_j \left(\prod_{i=1}^{D} 2\pi\sigma_i^2\right)^{-\frac{1}{2}} \exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i-\mu_{ji})^2\right\}\right]}$$

2. $l(\theta;D) = -\log p(y^{(1)}, x^{(1)}, y^{(2)}, x^{(2)}, \dots y^{(N)}, x^{(N)}|\theta)$

$$= -\left(\sum_{i=1}^{N}\log p(x^{(i)}|y=y^{(i)},\theta) + \sum_{i=1}^{N}\log p(y=y^{(i)})\right)$$

$$= -\frac{N}{2}\sum_{j=1}^{D}\log(2\pi\sigma_j^2) - \sum_{i=1}^{D}\frac{1}{2\sigma_j^2}\sum_{i=1}^{N}(x_j^{(i)}-\mu_{y^{(i)}j})^2 - \sum_{i=1}^{N}\log(a_{y^{(i)}})$$

3. $\frac{\partial}{\partial\mu_{ki}} l(\theta,D) = \frac{1}{\sigma_i^2}\sum_{n=1}^{N} 1(y^{(n)}=k)(x_i^{(n)}-\mu_{y^{(n)}i})$

$$\frac{\partial}{\partial\sigma_i^2} l(\theta,D) = -\frac{N}{2\sigma_i^2} + \frac{1}{2(\sigma_i^2)^2}\sum_{n=1}^{N} 1(x_i^{(n)}-\mu_{y^{(n)}i})^2$$

4. For μ_{ki}: make $\frac{\partial}{\partial\mu_{ki}} l(\theta,D) = 0$

$$\Rightarrow \frac{1}{\sigma_i^2}\sum_{n=1}^{N} 1(y^{(i)}=k)(x^{(i)}-\mu_{ki}) = 0$$

$$\Rightarrow \sum_{n=1}^{N} 1(y^{(i)}=k)x^{(i)} = \mu_{ki}\sum_{n=1}^{N} 1(y^{(i)}=k)$$

$$\Rightarrow \mu_{ki} = \frac{\sum_{i=1}^{N} 1(y^{(i)}=k)x^{(i)}}{\sum_{i=1}^{N} 1(y^{(i)}=k)}$$

For σ_i^2: make $\frac{\partial}{\partial\sigma_i^2} l(\theta,D) = 0$

$$\Rightarrow N - \frac{1}{\sigma_i^2}\sum_{n=1}^{N} 1(x_i^{(n)}-\mu_{y^{(n)}i})^2 = 0$$

$$\Rightarrow \frac{1}{\sigma_i^2} = \frac{N}{\sum_{n=1}^{N} 1(x_i^{(n)}-\mu_{y^{(n)}i})^2} = \frac{\sum_{n=1}^{N} 1(x_i^{(n)}-\mu_{y^{(n)}i})^2}{N}$$

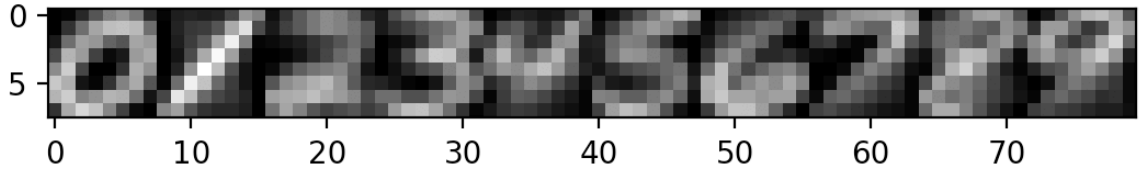
$$\Rightarrow \sigma_i^2 = \frac{\sum_{n=1}^{N} 1(x_i^{(n)}-\mu_{y^{(n)}i})^2}{N}$$

2.

0

Load the data and plot the means for each of the digit classes in the training data (include these in your report). Given that each image is a vector of size 64, the mean will be a vector of size 64 which needs to be reshaped as an 8 × 8 2D array to be rendered as an image. Plot all 10 means side by side using the same scale.

As shown by running q2_0.py



1.1

Build a simple K nearest neighbor classifier using Euclidean distance on the raw pixel data.

(a) For K = 1 report the train and test classification accuracy.

For K = 1,

The train classification accuracy is: 1.0

The test classification accuracy is: 0.96875

(b) For K = 15 report the train and test classification accuracy.

For K = 15,

The train classification accuracy is: 0.9637142857142857

The test classification accuracy is: 0.96075

1.2

For K > 1 K-NN might encounter ties that need to be broken in order to make a decision. Choose any (reasonable) method you prefer and explain it briefly in your report.

I choose the one which has the smallest mean of L2 distance to the test point. This means if many points circle around the test point and there is a tie, we always choose the class that is more close to the test point instead of the one that is further even though they are all covered under K.

1.3

Use 10 fold cross validation to find the optimal K in the 1-15 range. You may use the KFold implementation in sklearn or your existing code from Assignment 1. Report this value of K along with the train classification accuracy, the average accuracy across folds and the test accuracy.

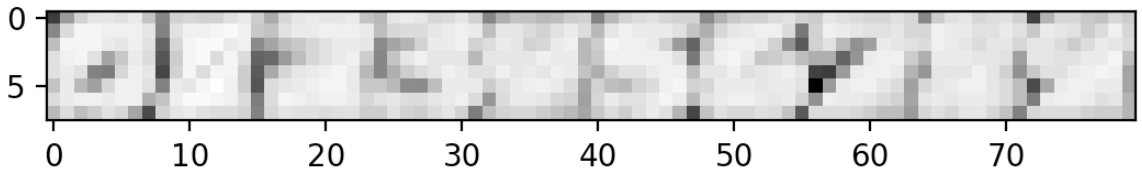
The optimal K is: 4

The train classification accuracy is: 0.9864285714285714

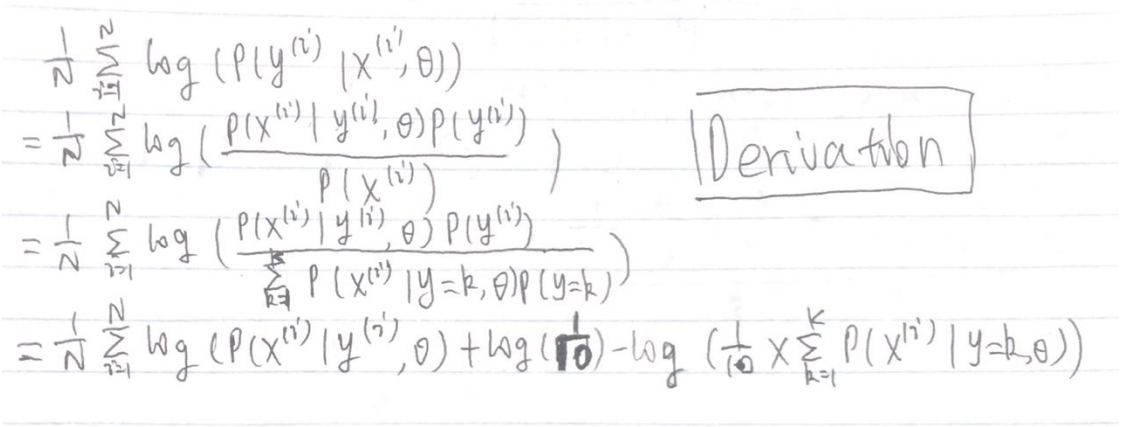
The average accuracy across folds is: 0.9657142857142859

The test classification accuracy is: 0.972

2.1
Plot an 8 by 8 image of the log of the diagonal elements of each covariance matrix Σk. Plot all ten classes side by side using the same grayscale.
As shown by running q2_1.py



2.2
Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. 1 N PN i=1 log(p(y (i) |x (i) , θ)) on both the train and test set and report it.
Average Conditional Likelihood for training set is: -0.124624436669
Average Conditional Likelihood for test set is: -0.196673203255



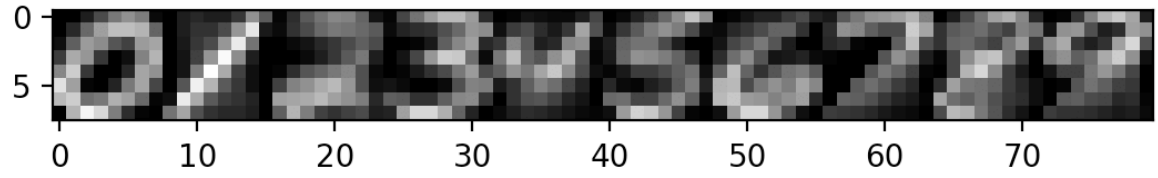
2.3
Select the most likely posterior class for each training and test data point as your prediction, and report your accuracy on the train and test set.

Accuracy for training set is: 0.9814285714285714
Accuracy for test set is: 0.97275

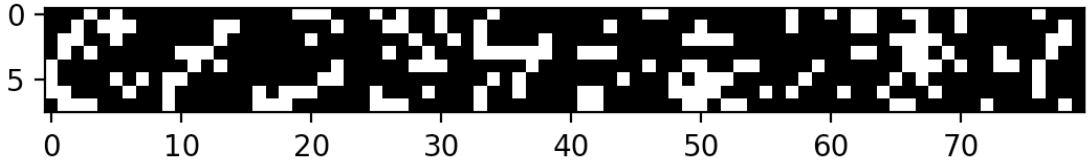
3.1
Convert the real-valued features x into binary features b using 0.5 as a threshold: bj = 1 if xj > 0.5 otherwise bj = 0.
As shown in q2_3.py

3.2
Using these new binary features b and the class labels, train a Bernoulli Naive Bayes classifier using MAP estimation with prior Beta(α, β) with α = β = 2. In particular, fit the model below on the training set.
As shown in q2_3.py

3.3
Plot each of your ηk vectors as an 8 by 8 grayscale image. These should be presented side by side and with the same scale



3.4
Given your parameters, sample one new data point using your generative model for each of the 10 digit classes. Plot these new data points as 8 by 8 grayscale images side by side.



3.5
Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. 1 N PN i=1 log(p(y (i) |x (i) , θ)) on both the train and test set and report it.

Average Conditional Likelihood for training set is: -0.9437538618
Average Conditional Likelihood for test set is: -0.987270433725

3.6
Select the most likely posterior class for each training and test data point, and report your accuracy on the train and test set.
Accuracy for training set is: 0.7741428571428571
Accuracy for test set is: 0.76425

4
Briefly (in a few sentences) summarize the performance of each model. Which performed best? Which performed worst? Did this match your expectations?

K-NN Classifier:
It uses all the training set data point to help classify the test point, it is slow but perform pretty good on the test set. The high performance is due to the handwritten digit is relatively easy to classify and does not share a lot similarities between digits. But notice if data features/dimensions increase by a huge amount or noise in the data set increase, this would lead to degradation of the performance.

Conditional Gaussian Classifier:
This classifier is based on the assumption that all the pixels are not completely independent. It takes covariance as its argument and this would improve the accuracy of the result. For example, digit '1' would properly have all pixels on in one vertical line. Certain pattern would apply to certain digits which mean this assumption is important for us to classify the data. And the data set is big which make the classification more precise because it will tune the parameter to better fit into the data set and gives us better test result.

Naive Bayes Classifier:
This classifies performs poorly as what has been pointed out in Conditional Gaussian Classifier, pixels' independence. And by converting x into binary features, a lot of the information would be lost and influence the performance.

Overall, based on the analysis and the statistics we get from running the code, we have that Conditional Gaussian Classifier performs best and Naive Bayes Classifier performed worst. This matches my expectation.