CSC411 Assignment 2

Yanzhi Zhang

November 2017

1 Class-Conditional Gaussians

1. Use Bayes' rule to derive an expression for $p(y = k|x, \mu, \sigma)$

Given that

$$p(y = k) = \alpha_k \tag{1}$$

$$p(x|y = k, \mu, \sigma) = (\prod_{i=1}^{D} 2\pi\sigma_i^2)^{-\frac{1}{2}} \times e^{(-\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - \sigma_{ki}^2))} \tag{2}$$

By the law of total probability,

$$p(x|\mu, \sigma) = \sum_{i \in (1,2,\dots,k)} (p(x|y = i, \mu, \sigma)p(y = i|\mu, \sigma)) \tag{3}$$

By Bayes' rule, we know that

$$p(y = k|x, \mu, \sigma) = \frac{p(x|y = k, \mu, \sigma)p(y = k, \mu, \sigma)}{p(x|\mu, \sigma)} \tag{4}$$

Plug (1),(2),(3) into (4)

$$p(y = k|x, \mu, \sigma) = \frac{((\prod_{i=1}^{D} 2\pi\sigma_i^2)^{-\frac{1}{2}} \times e^{(-\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - \sigma_{ki}^2))}) \times \alpha_k}{\sum_{j \in (1,2,\dots,k)} (p(x|y = j, \mu, \sigma)p(y = j|\mu, \sigma))} \tag{5}$$

$$p(y = k|x, \mu, \sigma) = \frac{((\prod_{i=1}^{D} 2\pi\sigma_i^2)^{-1\frac{1}{2}} \times e^{(-\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - \sigma_{ki}^2))}) \times \alpha_k}{\sum_{j \in (1,2,\dots,k)} ((\prod_{i=1}^{D} 2\pi\sigma_i^2)^{-\frac{1}{2}} \times e^{(-\sum_{j=1}^{D} \frac{1}{2\sigma_j^2}(x_j - \sigma_{k_j}^2))}) \times \alpha_j)} \tag{6}$$

2. Write down an expression for the negative likelihood function (NLL)

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

of a particular dataset $D = (y^{(1)}, x^{(1)}), (y^{(2)}, x^{(2)}), ..., (y^{(N)}, x^{(N)})$ with parameters $\theta = \{\alpha, \mu, \sigma\}$. (Assume that the data are iid.)

$$p((y^{(1)}, x^{(1)}), (y^{(2)}, x^{(2)}), ..., (y^{(N)}, x^{(N)}))|\theta) \tag{7}$$

$$\begin{aligned}(7) &= \prod_{i=1}^{n} p((y^{(i)}, x^{(i)}) \text{ , given that data are i.i.d.} \\ &= \prod_{i=1}^{n} p(x^{(i)}|y^{(i)}, \theta)p(y^{(i)}|\theta) \\ &= \prod_{i=1}^{n} p((\prod_{i=1}^{D} 2\pi\sigma_i^2)^{-\frac{1}{2}}) \times e^{(-\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - \sigma^2_{y^{(m)}i}))} \times \alpha_{y^{(m)}})\end{aligned}$$

$$NLL = -\log(l(\theta))$$

$$= \sum_{m=1}^{N} (\frac{1}{2}\log(\prod_{i=1}^{D} 2\pi\theta_i^2) + \sum_{i=1}^{D} \frac{1}{2\theta_i^2}(x_i^{(m)} - \mu_{y^{(m)}i})^2) - \log(\alpha_{y^{(i)}})) \tag{8}$$

3. Take partial derivatives of the likelihood with respect to each of the parameters μ_{ki} and with respect to the shared variances σ_i^2.

$$\frac{\partial(-\log l(\theta))}{\partial \mu_{ki}} =$$

$$= \frac{\partial \sum_{m=1}^{N} (\frac{1}{2}\log(\prod_{i=1}^{D} 2\pi\theta_i^2) + \sum_{i=1}^{D} \frac{1}{2\theta_i^2}(x_i^{(m)} - \mu_{y^{(m)}i})^2) - \log(\alpha_{y^{(i)}}))}{\partial \mu_{ki}} \tag{9}$$

$$= \sum_{m=1}^{N} \mathbb{1}(y^{(m)} = k) \sum_{n=1}^{D} \mathbb{1}(n = i) \frac{1}{\sigma_n^2}(\mu_{y^{(m)}n} - x_n^{(m)}) = \sum_{i=0}^{N} \mathbb{1}(y^{(i)} = k) \frac{1}{\sigma_n^2}(\mu_{y^{(m)}n} - x_n^{(m)})$$

$$\frac{\partial(-\log l(\theta))}{\partial \sigma_t^2}$$

$$= \frac{\partial \sum_{m=1}^{N} (\frac{1}{2}\log(\prod_{i=1}^{D} 2\pi\theta_i^2) + \sum_{i=1}^{D} \frac{1}{2\theta_i^2}(x_i^{(m)} - \mu_{y^{(m)}i})^2) - \log(\alpha_{y^{(i)}}))}{\partial \sigma_t^2} \tag{10}$$

$$= \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)\left[(\sigma_n^2)^{-1}-(\sigma_n^2)^{-2}(x_n^{(m)}-\mu_{y^{(m)}n})^2\right]$$

4. Find the maximum likelihood estimates for μ and σ.

let $\frac{\partial(-\log l(\theta))}{\partial \mu_{ki}} = 0$

$$\Rightarrow -\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)\sum_{n=1}^{D}(n=i)\frac{1}{\sigma_n^2}(\mu_{y^{(m)}n}-x_i^{(m)}) = 0$$

$$-\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)\sum_{n=1}^{D}(n=i)\frac{1}{\sigma_n^2}\times\mu_{y^{(m)}n} = -\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)\sum_{n=1}^{D}(n=i)\frac{1}{\sigma_n^2}x_i^{(m)}$$

$$\Rightarrow \mu_{ki} = \frac{\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)\sum_{n=1}^{D}\mathbb{1}(n=i)x_n^{(m)}}{\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)\sum_{n=1}^{D}\mathbb{1}(n=i)}$$

$$= \frac{\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)x_n^{(m)}}{\sum_{m=1}^{N}\mathbb{1}(y^{(m)}=k)}$$

let $\frac{\partial(-\log l(\theta))}{\partial(\sigma_i^2)} = 0$

$$\Rightarrow \frac{1}{2}\sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)\left[(\sigma_n^2)^{-1}-(\sigma_n^2)^{-2}(x_n^{(m)}-\mu_{y^{(m)}n})^2\right] = 0$$

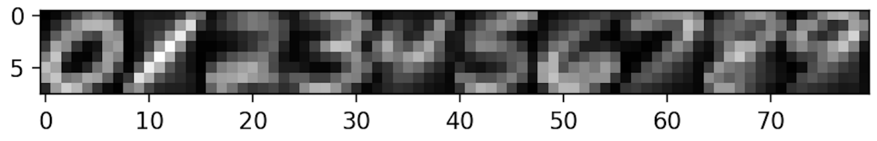
$$\Rightarrow \sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)\left[x_n^{(m)}-\mu_{y^{(m)}n})^2\right](\sigma_n^2)^{-2} = \sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)(\sigma_n^2)^{-1}$$

$$\Rightarrow \sigma_i^2 = \frac{\sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)\left[x_n^{(m)}-\mu_{y^{(m)}n}\right]^2}{\sum_{m=1}^{N}\sum_{n=1}^{D}\mathbb{1}(n=i)}$$

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

2 Handwritten Digit Classification

2.0 Load the data and plot the means for each of the digit classes in the training data



2.1 K-NN Classifier

1. Build a simple K-NN classifier using Euclidean distance on the raw pixel data and report the train and test classification accuracy.

For K = 1, the training set classification accuracy is 1.000000

For K = 1, the testing set classification gives 0.968750

For K = 15, the training set classification accuracy is 0.963714

For K = 15, the testing set classification accuracy is 0.960750

2. For K>1 K-NN might encounter ties that need to be broken in order to make a decision.

The method that I use here to is decrease the size of k when encountering a tie until the tie is broken.

Since K-NN is not a strictly mathematical method, the way that we break tie does not matter too much in term of accuracy. The intuition of K-NN is measuring the how close the target is related to the training model. Therefore, I think when we encounter a tie, it might be a good idea to decrease the value of k, which is counting the points that is much more closely related to the target data and intentionally ignore the data that is away from the target points. Additionally, we can guarantee to break tie in this way.

3. Use 10 fold cross validation to find the optimal K in the 1-15 range.

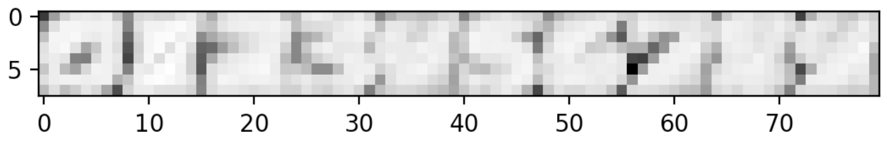
Using 1-NN classifier gives an accuracy of 0.964429
Using 2-NN classifier gives an accuracy of 0.964429
Using 3-NN classifier gives an accuracy of 0.965143
Using 4-NN classifier gives an accuracy of 0.965571
Using 5-NN classifier gives an accuracy of 0.963429

Using 6-NN classifier gives an accuracy of 0.964571
Using 7-NN classifier gives an accuracy of 0.960714
Using 8-NN classifier gives an accuracy of 0.961571
Using 9-NN classifier gives an accuracy of 0.958000
Using 10-NN classifier gives an accuracy of 0.956857
Using 11-NN classifier gives an accuracy of 0.955571
Using 12-NN classifier gives an accuracy of 0.955000
Using 13-NN classifier gives an accuracy of 0.953143
Using 14-NN classifier gives an accuracy of 0.954286

Among all the K-NN, the classifier performes the best when k is 4. When k=4, the training set classification accuracy is 0.986429, the testing set classification accuracy is 0.972750 and the average accuracy across folds is 0.965571

2.2 Conditional Gaussian Classifier

1. Plot an 8 by 8 image of the log of the diagonal elements of each covariance matrix $\sum k$. Plot all ten classes side by side using the same grayscale.



2. Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood,i.e. $\frac{1}{N}\sum_{i=1}^{N} log(p(y^{(i)}|x^{(i)},\theta))$ on both the train and test set

The derivation for average log is:

$$\frac{1}{N}\sum_{i=1}^{N}\log(p(y^{(i)}|x^{(i)},\theta)$$
$$=\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{p(x^{(i)}|y^{(i)},\theta)\cdot p(y^{(i)}|\theta)}{P(x^{(i)}|\theta)}\right)$$
$$=\frac{1}{N}\sum_{i=1}^{N}\left[\log P(x^{(i)}|y^{(i)},\theta)+\log(p^{(i)}|\theta)-\log(p(x^{(i)}|\theta)\right]$$
$$=\frac{1}{N}\sum_{i=1}^{N}\left[\log(p(x^{(i)}|y^{(i)},\theta)+\log(\tfrac{1}{10})-\log\left[\sum_{j=1}^{N}P(x^{(i)}|y^{(j)},\theta)P(y^{(j)}|\theta)\right]\right]$$

Average conditional likelihood over the true training class labels is -0.124624

Average conditional likelihood over the true testing class labels is -0.196673

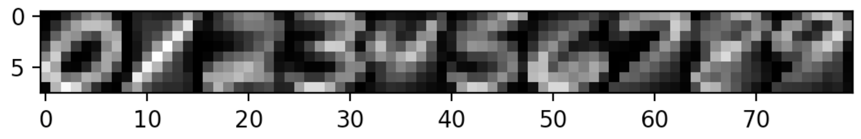
3. Select the most likely posterior class for each training and test data point as your prediction, and report your accuracy on the train and test set.

Conditional Gaussian classifier on testing set has an accuracy of 0.972750.

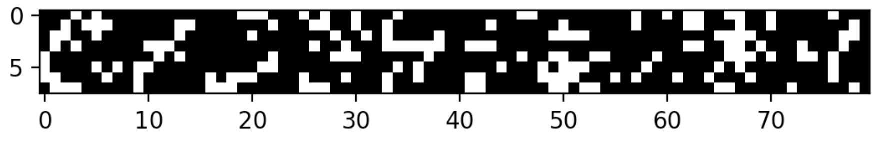
Conditional Gaussian classifier on training set has an accuracy of 0.981429.

2.3 Naive Bayes Classifier Training

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



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



5. Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. $\frac{1}{N}\sum_{i=1}^{N} log(p(y^{(i)}|x^{(i)},\theta))$ **on both the train and test set and report it**

Average conditional likelihood over the true training class labels is -0.943754

Average conditional likelihood over the true testing class labels is -0.987270

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

Naive Bayes classifier on testing set has an accuracy of 0.764250.

Naive Bayes classifier on training set has an accuracy of 0.774143.

2.4 Model Comparison

Conditional Gaussian performed the best.

The Naive Bayes cannot predict the digits very well because the relationship between each pixel is very dependent to the actual digits. Some pixels can be on in most of cases and some pixels can be off for most of the pixels. Therefore, the result that we are getting from Naive Bayes classifier is very vague.

Conditional Gaussian did the best job because we are having a very large data set so we can train a model that can generally fit the test set into the correct digit. Regardless of the dependence of each pixels, it takes into consideration that each pixel are not complete independent to each others, therefore, conditional Gaussian predicts the digits fairly well.

K-NN is very time-consuming because it is going through the whole training data set to find the point that matches the test data best. Since the class size is relatively small and each class is very diverse, we can conclude that the data set is not very noisy. In this case, K-NN will definitely have a good overall performance. However, K-NN might do worse if the dimension of feature becomes larger or the data set becomes more noisy.