CSC411H1

Assignment 2

Jahyun Shin (998994888)

Submitted On:

November 12th, 2017

1 Class-Conditional Gaussians

1. $$P(x|\mu,\sigma) = \sum_{k=1}^{K}\left[P(x|y=k,\mu,\sigma)P(y=k)\right]$$
$$= \sum_{k=1}^{K}\left[\left(\prod_{i=1}^{D}2\pi\sigma_i^2\right)^{-\frac{1}{2}}\exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i-\mu_{ki})^2\right\}(\alpha_k)\right]$$

$$\therefore P(y=k|x,\mu,\sigma) = \frac{P(x|y=k,\mu,\sigma)P(y=k)}{P(x|\mu,\sigma)}$$
$$= \frac{\left(\prod_{i=1}^{D}2\pi\sigma_i^2\right)^{-\frac{1}{2}}\exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i-\mu_{ki})^2\right\}(\alpha_k)}{\sum_{k=1}^{K}\left[\left(\prod_{i=1}^{D}2\pi\sigma_i^2\right)^{-\frac{1}{2}}\exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i-\mu_{ki})^2\right\}(\alpha_k)\right]}$$

2. $$P(y,x|\alpha,\mu,\sigma) = \prod_{j=1}^{N}\left[P(x^{(j)}|y^{(j)}=k,\mu,\sigma)P(y^{(j)}=k)\right]$$
$$= \prod_{j=1}^{N}\left[\left(\prod_{i=1}^{D}2\pi\sigma_i^2\right)^{-\frac{1}{2}}\exp\left\{-\sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i^{(j)}-\mu_{y^{(j)}i})^2\right\}(\alpha_{y^{(j)}})\right] = L$$

$$\therefore -\log L = -\sum_{j=1}^{N}\left[-\frac{1}{2}\sum_{i=1}^{D}\log(2\pi\sigma_i^2) - \sum_{i=1}^{D}\frac{1}{2\sigma_i^2}(x_i^{(j)}-\mu_{y^{(j)}i})^2 + \log(\alpha_{y^{(j)}})\right]$$

3. $$\frac{\partial \log L}{\partial \mu_{ki}} = \sum_{j=1}^{N}\mathbb{1}(y^{(j)}=k)\frac{1}{\sigma_i^2}(x_i^{(j)}-\mu_{ki})$$

$$\frac{\partial \log L}{\partial \sigma_i^2} = \sum_{j=1}^{N}\left[-\frac{1}{2}\cdot\frac{1}{2\pi\sigma_i^2}\cdot 2\pi - \left(-\frac{1}{2(\sigma_i^2)^2}(x_i^{(j)}-\mu_{y^{(j)}i})^2\right)\right]$$
$$= \sum_{j=1}^{N}\left[-\frac{1}{2\sigma_i^2} + \frac{1}{2\sigma_i^4}(x_i^{(j)}-\mu_{y^{(j)}i})^2\right]$$

4. $\frac{\partial \log L}{\partial \mu_{ki}} = \sum_{j=1}^{N} \mathbb{1}(y^{(j)}=k) \frac{1}{\sigma_i^2} (x_i^{(j)} - \mu_{ki}) = 0$

$$\sum_{j=1}^{N} \mathbb{1}(y^{(j)}=k)\mu_{ki} = \sum_{j=1}^{N} \mathbb{1}(y^{(j)}=k) x_i^{(j)}$$

$$\therefore \mu_{ki} = \frac{\sum_{j=1}^{N} \mathbb{1}(y^{(j)}=k) x_i^{(j)}}{\sum_{j=1}^{N} \mathbb{1}(y^{(j)}=k)}$$

$\rightarrow \mu = \{\mu_{ki}\} \quad \forall k = 1, 2, 3, \cdots, K$
and $\forall i = 1, 2, 3, \cdots, D$

$$\frac{\partial \log L}{\partial \sigma_i} = \sum_{j=1}^{N} \left[-\frac{1}{2} \cdot \frac{1}{2\pi\sigma_i^2} \cdot 4\pi\sigma_i - (-\sigma_i^{-3} (x_i^{(j)} - \mu_{y^{(j)}i})^2) \right]$$

$$= \sum_{j=1}^{N} \left[-\frac{1}{\sigma_i} + \frac{1}{\sigma_i^3} (x_i^{(j)} - \mu_{y^{(j)}i})^2 \right] = 0$$

$$\sum_{j=1}^{N} \left[-\sigma_i^2 + (x_i^{(j)} - \mu_{y^{(j)}i})^2 \right] = 0$$

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

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

$$\therefore \sigma_i = \sqrt{\frac{\sum_{j=1}^{N} (x_i^{(j)} - \mu_{y^{(j)}i})^2}{N}} \rightarrow \sigma = \{\sigma_i\} \quad \forall i = 1, 2, 3, \cdots, D$$

* IF the question was asking for variance vector, σ^2 instead of σ:

$$\frac{\partial \log L}{\partial \sigma_i^2} = \sum_{j=1}^{N} \left[-\frac{1}{2\sigma_i^2} + \frac{1}{2\sigma_i^4} (x_i - \mu_{y^{(j)}i})^2 \right] = 0$$

$$\sum_{j=1}^{N} \left[-\sigma_i^2 + (x_i - \mu_{y^{(j)}i})^2 \right] = 0$$

↳ same result as above (with σ)

$$\therefore \sigma_i^2 = \frac{\sum_{j=1}^{N} (x_i^{(j)} - \mu_{y^{(j)}i})^2}{N} \rightarrow \sigma^2 = \{\sigma_i^2\} \quad \forall i = 1, 2, 3, \cdots, D$$

5. Let $f(\alpha_k) = L$ & $g(\alpha_k) = 1 - \sum_k \alpha_k$

such that Max L subject to $1 - \sum_k \alpha_k$

$\rightarrow$ Max $G = L - \lambda(1 - \sum_k \alpha_k) = L + \lambda(\sum_k \alpha_k - 1)$

$$\frac{\partial G}{\partial \alpha_k} = \frac{\partial L}{\partial \alpha_k} + \lambda \frac{\partial(\sum_k \alpha_k)}{\partial \alpha_k} = 0$$

- $\frac{\partial L}{\partial \alpha_k} = \sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k) \frac{1}{\alpha_k}$ (from the expression of L in Question 2)
- $\frac{\partial(\sum_k \alpha_k)}{\partial \alpha_k} = 1$

$$\therefore \sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k) \frac{1}{\alpha_k} + \lambda = 0 \rightarrow \alpha_k = \frac{\sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k)}{-\lambda} \quad \cdots\cdots (1)$$

* $\sum_k \alpha_k = 1 \rightarrow -\frac{1}{\lambda} \sum_k \sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k) = 1$, where $\sum_k \sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k) = N$

$$\therefore -\frac{N}{\lambda} = 1 \quad \therefore \lambda = -N \quad \cdots (2)$$

Plugging (2) in (1):

$$\alpha_k = \frac{\sum_{j=1}^{N} \mathbb{1}(y^{(j)} = k)}{N}$$

2 Handwritten Digit Classification

2.0 Plotting the Means for Each Digit Class

The means for each of the 10 digit classes were plotted side by side as 8 x8 2D array as following:

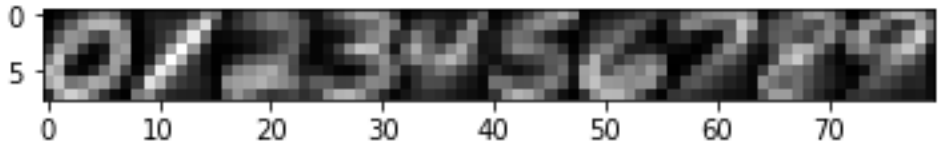

Figure 1. Plot of the Means for Each of the 10 Digit Classes as 8 by 8 Grayscale Images Side by Side

2.1 K-NN Classifier

1. In order to compute classification accuracy, an accuracy matrix is constructed and manipulated within the python code. In classification_accuracy function, “accuracy_matrix” of size (10 x 10) is organized as the following matrix:

True Digit (i) \ Predicted Digit (j)	0	1	2	3	4	5	6	7	8	9
0	**TP***	**False Negatives (FN)**								
1	**False Positives (FP)**	**TP**								
2			**TP**							
3				**TP**						
4					**TP**					
5						**TP**				
6							**TP**			
7								**TP**		
8									**TP**	
9										**TP**

*TP denotes True Positive

Each cell contains the number of the matching cases (i.e. (i,j) = (0,0) cell contains the number of data points with a true label of 0 and was correctly predicted as 0, while (3,8) cell contains the number of data points with a true label of 3 but was incorrectly predicted as 8). The diagonal values of this matrix represent True Positive cases where the true digit and predicted digit coincide. Each column excluding the True Positive element represents False Positive cases

for each prediction. Each row excluding the True Positive element represents False Negative cases for each true digit.

After constructing the accuracy matrix, average precision, average recall, and F1 score values are computed as following:

- Average Precision = $\frac{1}{10}\sum_{j=0}^{9}\frac{(TP)j}{(TP)j+\sum_{i=0}^{9}(FP)i}$

- Average Recall = $\frac{1}{10}\sum_{i=0}^{9}\frac{(TP)i}{(TP)i+\sum_{j=0}^{9}(FP)j}$

- F1 Score = $\frac{2*\text{Average Precision}*\text{Average Recall}}{\text{Average Precision}+\text{Average Recall}}$

The following table summarizes F1 Score values for train and test classification for 1-NN and 15-NN classifier algorithms:

K	**Train** Classification Accuracy (F1 Score)	**Test** Classification Accuracy (F1 Score)
1	**1.0***	**0.968859866927**
15	**0.825421052549**	**0.796846408018**

Table 1. Train and Test Classification Accuracies for K=1 & K=15 K-NN Classifier

**Note*: *This perfect 100% training accuracy of 1-NN Classifier is only due to the fact that each training point has 0 distance with itself.*

2. For K > 1 K-NN Classifier, there may be ties between the number of nearest labels. The method chosen to break such ties is as follows:

*"Out of all labels that appear the **most** out of k labels, the label that appears the **earliest** in the k-nearest-neighbour array ("k_nearest_labels" in my code) will be most likely to be the correct label because the first neighbour with the most frequent label has the shortest distance with the test point."*

3. **<u>10-Fold Cross Validation to find the Optimal K in 1 – 15 Range:</u>**

The result of 10-Fold Cross Validation on the training set to find the optimal K in 1-15 range is shown in the following plot:

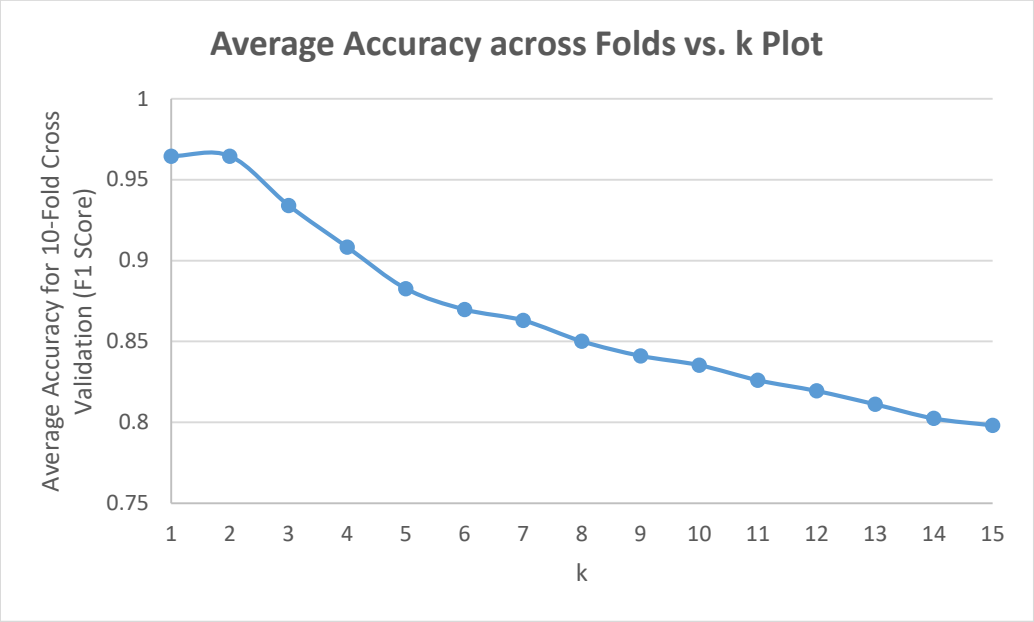

Figure 2. Average Accuracy across Folds vs. k Plot

As seen in the plot above, a declining pattern in the average accuracy across folds was observed as k increased from 1 to 15. Cases where k = 1 and k = 2 showed the equal highest accuracy of 0.964443314986. The following table summarizes the train classification accuracy, average accuracy across folds, and test accuracy for the classification performed with the optimal K of 1 & 2 (the identical accuracies were observed for classifications with k = 1 and k = 2) :

Optimal K	**Train** Classification Accuracy with Optimal K (F1 Score)	**Average** Accuracy **across Folds** with Optimal K (F1 Score)	**Test** Classification Accuracy with Optimal K (F1 Score)
1 & 2*	**1.0**	**0.964443314986**	**0.968859866927**

Table 2. Optimal K, Train and Test Classification Accuracy, and Average Accuracy across Folds for 10-Fold Cross Validation to find the Optimal K in 1 – 15 Range

*Note: The F1 score for k = 1 & k = 2 resulted in an equal value

2.2 Conditional Gaussian Classifier Training

1. The log of the diagonal elements of each of the 10 covariance matrices Σ_k for k = 0, 1, ... , 9 were plotted side by side as 8 x8 2D array as following:

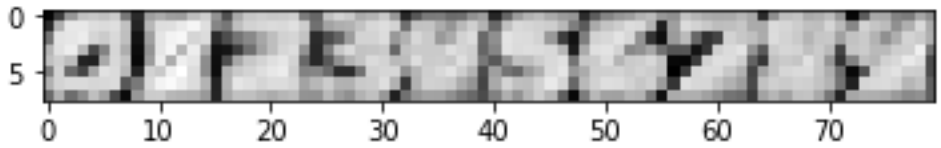

Figure 3. Plot of the Log of the Diagonal Elements of Each Covariance matrix Σ_k for k = 0, 1, ... , 9 as 8 by 8 Grayscale Images Side by Side

2. The following table summarizes the average conditional log likelihood for training and test set computed using conditional Gaussian classifier training:

Average Conditional Log Likelihood for **Training** Set	Average Conditional Log Likelihood for **Test** Set
75.2443963035	**73.7748404459**

Table 3. Average Conditional Log Likelihood for Training and Test Set using Conditional Gaussian Classifier Training

3. The following table summarizes train and test classification accuracies in terms of F1 score computed using conditional Gaussian classifier for digit classification:

Train Classification Accuracy (F1 Score)	**Test** Classification Accuracy (F1 Score)
0.981465014348	**0.972870663193**

Table 4. Train and Test Classification Accuracies for Conditional Gaussian Classifier

4. The leading eigenvectors of each of the 10 covariance matrices Σ_k for k = 0, 1, ... , 9 were plotted side by side as 8 x8 2D array as following:

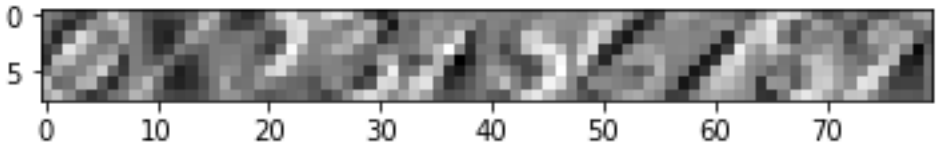

Figure 4. Plot of Leading Eigenvectors for Each Covariance matrix Σ_k for k = 0, 1, ... , 9 as 8 by 8 Grayscale Images Side by Side

2.3 Naïve Bayes Classifier Training

2. In the code, regularization method was used instead of the beta prior estimation.

3. Each of the 10 η_k vectors for k = 0, 1, ... , 9 were plotted side by side as 8 x8 2D array as following:

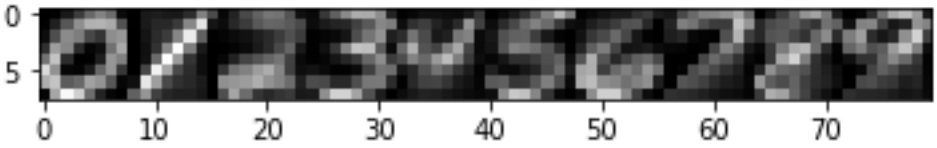

Figure 5. Plot of Each of η_k Vectors for k = 0, 1, ... , 9 as 8 by 8 Grayscale Images Side by Side

4. A newly sampled data for each of the 10 digits classes with η_k parameters for k = 0, 1, ... , 9 were plotted side by side as 8 x8 2D array as following:

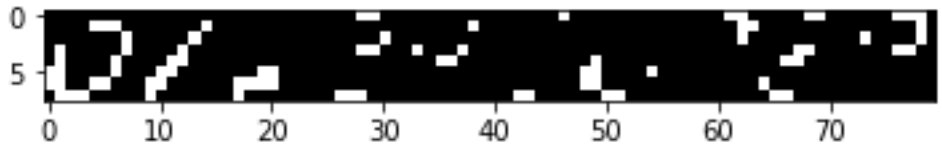

Figure 6. Plot of Newly Sampled Data Points as 8 by 8 Grayscale Images Side by Side

5. The following table summarizes the average conditional log likelihood for training and test set computed using Naïve Bayes classifier training:

Average Conditional Log Likelihood for **Training** Set	Average Conditional Log Likelihood for **Test** Set
- 0.9437538618	**- 0.987270433725**

Table 5. Average Conditional Log Likelihood for Training and Test Set using Naïve Bayes Classifier Training

6. The following table summarizes train and test classification accuracies in terms of F1 score computed using Naïve Bayes classifier for digit classification:

Train Classification Accuracy (F1 Score)	**Test** Classification Accuracy (F1 Score)
0.776652641714	**0.766824458597**

Table 6. Train and Test Classification Accuracies for Naïve Bayes Classifier

2.4 Model Comparison

The model that performed best in terms of the highest train and test classification accuracy was Conditional Gaussian Classifier Training. This result was expected because modelling of the conditional probability of each data point as multivariate Gaussian distribution is a much more accurate representation of real-life data compared to the conditional independence assumed in Naïve Bayes or simple distance-based K-NN algorithm. It must be noted that the perfect 100% training accuracy of 1-NN Classifier is only due to the fact that each training point has 0 distance with itself. 1-NN Classifier has lower test accuracy than that of Conditional Gaussian Classifier.

The model that performed worst and resulted in the lowest train and test classification accuracy was Naïve Bayes Training. Such outcome was expected because the overly simplified Naïve Bayes assumption that all data points are independent and identically distributed is highly unlikely in real data.