EECE5644 Summer 1 2022 – Assignment 1

Submit: Tuesday, 2022-May-31 before 11:59 EDT (upload PDF to Canvas)

Please submit your solutions at the classroom assignments page in Canvas. Please upload a single PDF file that includes all math, numerical and visual results, and code (as an appendix or as a link to your online code repository for this assignment). If you point to an online repository, do NOT edit the contents after the deadline, because TAs may interpret a last-modified timestamp past the deadline as a late submission of the assignment. Only the contents of the PDF will be graded. Please do NOT link from the PDF to external documents online where results may be presented (e.g. online Notebooks of any kind).

This is a graded assignment, and the entirety of your submission must contain only your own work. You may benefit from publicly available literature including software (not from classmates), as long as these sources are properly acknowledged in your submission. Copying math or code from each other is not allowed and will be considered as academic dishonesty. While there cannot be any written material exchange between classmates, verbal discussions to help each other are acceptable. Discussing with the instructor, the teaching assistant, and classmates at open office periods to get clarification or to eliminate doubts are acceptable.

By submitting a PDF file in response to this take-home assignment, you are declaring that the contents of your submission, and the associated code is your own work, except as noted in your citations of external resources.

Question 1 (30%)

The probability density function (pdf) for a 3-dimensional real-valued random vector $\mathbf{X}$ is as follows: $p(\mathbf{x}) = p(\mathbf{x}|L=0)P(L=0) + p(\mathbf{x}|L=1)P(L=1)$. Here L is the true class label that indicates which class-label-conditioned pdf generates the data.

The class priors are $P(L=0) = 0.65$ and $P(L=1) = 0.35$. The class class-conditional pdfs are $p(\mathbf{x}|L=0) = g(\mathbf{x}|\mathbf{m}_0, \mathbf{C}_0)$ and $p(\mathbf{x}|L=1) = g(\mathbf{x}|\mathbf{m}_1, \mathbf{C}_1)$, where $g(\mathbf{x}|\mathbf{m}, \mathbf{C})$ is a multivariate Gaussian probability density function with mean vector $\mathbf{m}$ and covariance matrix $\mathbf{C}$. The parameters of the class-conditional Gaussian pdfs are:

$$\mathbf{m}_0 = \begin{bmatrix} -1/2 \\ -1/2 \\ -1/2 \end{bmatrix} \quad \mathbf{C}_0 = \begin{bmatrix} 1 & -0.5 & 0.3 \\ -0.5 & 1 & -0.5 \\ 0.3 & -0.5 & 1 \end{bmatrix} \quad \mathbf{m}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \quad \mathbf{C}_1 = \begin{bmatrix} 1 & 0.3 & -0.2 \\ 0.3 & 1 & 0.3 \\ -0.2 & 0.3 & 1 \end{bmatrix}$$

For numerical results requested below, generate 10000 samples according to this data distribution, keep track of the true class labels for each sample. Save the data and use the same data set in all cases.

Part A: ERM classification using the knowledge of true data pdf:

1. Specify the minimum expected risk classification rule in the form of a likelihood-ratio test: $\frac{p(\mathbf{x}|L=1)}{p(\mathbf{x}|L=0)} \overset{?}{>} \gamma$, where the threshold γ is a function of class priors and fixed (non-negative) loss values for each of the four cases $D = i|L = j$ where D is the decision label that is either 0 or 1, like L.

2. Implement this classifier and apply it on the 10K samples you generated. Vary the threshold γ gradually from 0 to ∞, and for each value of the threshold compute the true positive (detection) probability $P(D=1|L=1;\gamma)$ and the false positive (false alarm) probability $P(D=1|L=0;\gamma)$. Using these paired values, trace/plot an approximation of the ROC curve of the minimum expected risk classifier. Note that at $\gamma = 0$ The ROC curve should be at $\binom{1}{1}$, and as *gamma* increases it should traverse towards $\binom{0}{0}$. Due to the finite number of samples used to estimate probabilities, your ROC curve approximation should reach this destination value for a finite threshold value. Keep track of $(D=0|L=1;\gamma)$ and $P(D=1|L=0;\gamma)$ values for each *gamma* value for use in the next section.

3. Determine the threshold value that achieves minimum probability of error, and on the ROC curve, superimpose clearly (using a different color/shape marker) the true positive and false positive values attained by this minimum-P(error) classifier. Calculate and report an estimate of the minimum probability of error that is achievable for this data distribution. Note that $P(error;\gamma) = P(D=1|L=0;\gamma)P(L=0) + P(D=0|L=1;\gamma)P(L=1)$. How does your empirically selected γ value that minimizes P(error) compare with the theoretically optimal threshold you compute from priors and loss values?

Part B: ERM classification attempt using incorrect knowledge of data distribution (Naive Bayesian Classifier, which assumes features are independent given each class label)... For this part, assume that you know the true class prior probabilities, but for some reason you think that

the class conditional pdfs are both Gaussian with the true means, but (incorrectly) with covariance matrices that are both equal to the identity matrix (with diagonal entries equal to true variances, off-diagonal entries equal to zeros, consistent with the independent feature assumption of Naive Bayes). Analyze the impact of this model mismatch in this Naive Bayesian (NB) approach to classifier design by repeating the same steps in Part A on the same 10K sample data set you generated earlier. Report the same results, answer the same questions. Did this model mismatch negatively impact your ROC curve and minimum achievable probability of error?

Part C: In the third part of this exercise, repeat the same steps as in the previous two cases, but this time using a Fisher Linear Discriminant Analysis (LDA) based classifier. Using the 10K available samples, estimate the class conditional pdf mean and covariance matrices using sample average estimators for mean and covariance. From these estimated mean vectors and covariance matrices, determine the Fisher LDA projection weight vector (via the generalized eigendecomposition of within and between class scatter matrices): $\mathbf{w}_{LDA}$. For the classification rule $\mathbf{w}_{LDA}^T\mathbf{x}$ compared to a threshold τ, which takes values from $-\infty$ to ∞, trace the ROC curve. Identify the threshold at which the probability of error (based on sample count estimates) is minimized, and clearly mark that operating point on the ROC curve estimate. Discuss how this LDA classifier performs relative to the previous two classifiers.

Note: When finding the Fisher LDA projection matrix, do not be concerned about the difference in the class priors. When determining the between-class and within-class scatter matrices, use equal weights for the class means and covariances, like we did in class.

Question 2 (30%)

A 2-dimensional random vector $\mathbf{X}$ takes values from a mixture of four Gaussians. Each Gaussian pdf is the class-conditional pdf for one of four class labels $L \in \{1,2,3,4\}$. For this problem, pick your own 4 distinct Gaussian class conditional pdfs $p(\mathbf{x}|L=j),\ j \in \{1,2,3,4\}$. Set class priors to $0.2, 0.25, 0.25, 0.3$. Select your Gaussian class conditional pdfs to have mean vectors approximately equally spaced out along a line, and the covariance matrices to be scaled versions of the identity matrix (with scale factors that lead to a significant amount of overlap between the data from these Gaussians). Label these classes in order according to the ordering of mean vectors along the line, so that classes have consecutive integer labels.

Part A: Minimum probability of error classification (0-1 loss, also referred to as Bayes Decision rule or MAP classifier).

1. Generate 10000 samples from this data distribution and keep track of the true labels of each sample.
2. Specify the decision rule that achieves minimum probability of error (i.e., use 0-1 loss), implement this classifier with the true data distribution knowledge, classify the 10K samples and count the samples corresponding to each decision-label pair to empirically estimate the confusion matrix whose entries are $P(D=i|L=j)$ for $i, j \in \{1,2,3,4\}$. Present the results.
3. Provide a visualization of the data (scatter-plot in 2-dimensional space), and for each sample indicate the true class label with a different marker shape (dot, circle, triangle, square) and whether it was correctly (green) or incorrectly (red) classified with a different marker color as indicated in parentheses.

Part B: Repeat the exercise for the ERM classification rule with the following loss values (errors between Gaussian pairs that have higher separation in their means will be penalized more):

$$\Lambda = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} \tag{1}$$

Note that, the $(i,j)^{th}$ entry of the loss matrix indicates the loss incurred by deciding on class i when the true label is j. For this part, using the 10K samples, estimate the minimum expected risk that this optimal ERM classification rule will achieve. Present your results with visual and numerical representations. Briefly discuss interesting insights, if any.

Hint: For each sample, determine the loss matrix entry corresponding to the decision-label pair that this sample falls into, and add this loss to an estimate of cumulative loss. Divide cumulative loss by the number of samples to get average loss as an estimate for expected loss.

Question 3 (40%)

Download the following datasets...

- Wine Quality dataset located at `https://archive.ics.uci.edu/ml/datasets/Wine+Quality`, specifically the **white** wine dataset, which consists of 11 features, and class labels from 0 to 10 indicating wine quality scores. There are 4898 samples.
- Human Activity Recognition dataset located at `https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones`, which consists of 561 features, and 6 activity labels. There are 10299 samples.

Implement minimum-probability-of-error classifiers for these problems, assuming that the class conditional pdf of features for each class you encounter is a Gaussian. Using all available samples from a class, estimate mean vectors and covariance matrices with sample averages. Using sample counts, also estimate class priors. In case your sample estimates of covariance matrices are ill-conditioned[1], consider adding a regularization term to your covariance estimate as in: $\mathbf{C}_{Regularized} = \mathbf{C}_{SampleAverage} + \lambda \mathbf{I}$ where $\lambda > 0$ is a small regularization parameter that ensures the regularized covariance matrix $\mathbf{C}_{Regularized}$ has all eigenvalues larger than this parameter. Using regularization in this context will allow you to solve an otherwise ill-posed problem.

With these estimated (trained) Gaussian class conditional pdfs and class priors, apply the minimum-P(error) classification rule on all (training) samples, count the errors, and report the error probability estimate you obtain for each problem, as well as the confusion matrices for this classification rule.

Visualize the datasets in 2- or 3-dimensional projections of subsets of features and then do the same 2- or 3-dimensional plots using principal component analysis (PCA). Discuss the following:

- If Gaussian class conditional models are appropriate for these datasets, commenting on the differences in how feature-subsets or PCA helped you draw your conclusions
- How your model choice might have influenced the confusion matrix and probability of error values you obtained
- Any modeling assumptions, *e.g.* how you estimated/selected necessary parameters for your model and classification rule

Describe your analyses in mathematical terms supplemented by numerical and visual results, conveying your understanding of what you have accomplished and demonstrated.

Hint: Later in the course, we will talk about how to select regularization/hyper-parameters. For now, you may consider using a value on the order of arithmetic average of sample covariance matrix estimate non-zero eigenvalues $\lambda = \alpha \times trace(\mathbf{C}_{SampleAverage})/rank(\mathbf{C}_{SampleAverage})$ *or geometric average of sample covariance matrix estimate non-zero eigenvalues, where* $0 < \alpha < 1$ *is a small real number. This makes your regularization term proportional to the eigenvalues observed in the sample covariance estimate.*

[1] Read here about ill-conditioned matrices: `https://deepai.org/machine-learning-glossary-and-terms/ill-conditioned-matrix`