EECE5644 Fall 2019 – Exam 2

Submit before Thursday, 2019-November-28 09:00ET

Please submit your solutions on Blackboard in a PDF file that includes all math and numerical results. In this PDF file, also include links to external repositories in which your code that generated the solutions can be found. Only the contents of the PDF file will be graded, so do not link out to online sites where you have results, such as a Jupyter Notebook. Include everything in the PDF file. Make sure that you cite all resources you benefit from (books, papers, software packages). This is a graded assignment and the entirety of your submission must contain only your own work. You may benefit from literature including software, as long as these sources are properly acknowledged.

Question 1 (50%)

You will train a multilayer perceptron neural network model using maximum likelihood parameter estimation procedure to approximate the maximum a posteriori (MAP) classification rule, which is the therotically optimal solution for minimizing probablity of error.

Part 1: Select data distribution.

- Select a data distribution for $\mathbf{x} \in \mathbb{R}^3$ that is a mixture of 4 Gaussians. Each Gaussian component represents a class label.
- Select distinct (non-uniform) class priors (i.e. distinct Gaussian component weights in the convex linear combination).
- Select distinct mean vectors and covariance matrices for each Gaussian class-conditional probability distribution.
- Do not choose covariance matrices to be diagonal; and select each coariance matrix to have distinct eigenvalues (so that your Gaussians are tilted with respect to coordinate axes and have elongated ellipsoid equilevel contours).
- Choose the mean vectors and covariance matrices of the Gaussians to allow for a moderate level of overlap between class distributions so that the theoretical minimum probability of error is neither too small (e.g. $< 3\%$) nor too large (e.g. $> 25\%$). This will make the problem more interesting.
- Clearly specify your data distribution and demonstrate it visually with a scatter plot using an appropriate number of samples drawn in an iid fashion from your distribution (e.g., draw 1000 samples and plot them in a 3-dimensional scatter plot with class labels color coded).

Part 2: Determine and evaluate the theoretically optimal MAP classifier.

- For your selected data distribution, determine and specify the MAP-classification rule.
- Generate (at least) 10000 iid (test) samples (for which you know the class labels of course since you generate them); apply the MAP classifier to these samples to get MAP-decision labels.
- Counting the number of misclassified samples and dividing by the total number of samples, estimate the theoretical minimum probability of error achievable for your data distribution.
- Save this test dataset for use in the next step.
- Present appropriate math, and visual/numerical results to convince the reader that you have implemented the theoretical MAP classifier appropriately, and to approximate the theoretically achieveable smallest error probability.

Part 3: Train and evaluate a classifier based on neural network based approximations of the class label posteriors; for a given input data vector class label decision will be made by selecting the neural network output index that has the largest value.

- Generate three separate training datasets from your data distribution; these datasets will respectively have 100, 1000, and 10000 iid samples and their true labels.
- For each dataset, you will train a multilayer perceptron to approximate class posterior probabilities given the data vector, using the maximum likelihood parameter estimation principle. These neural networks will each have a single hidden layer (i.e. two layers of weight matrices) with sigmoid outputs (choose your favorite from a suitable list, including logistic, hyperbolic tangent, and softplus functions). For the output layer nonlinearity, use the normalized exponential function (in order to ensure that your vector-valued output approximating class posteriors for each class label are in the probability simplex). Determine and present the necessary mathematical expression of the optimization problem that needs to be solved to achieve this goal. Implement appropriately using your preferred software package. Describe your implementation and explain how it matches the mathematical description of the problem you specified.
- For each multilayer perceptron neural network you train on a given dataset, determine the most appropriate number of perceptrons (units/nodes) in the hidden layer, use 10-fold cross-validation using probability of correct decisions as your performance measure (since it is 1 minus probability or error, and is consistent with our overarching objective to design a classifier with the smallest possible error probability). Present appropriate math, descriptions, visual and numerical results to convince the reader that you have done model order selection appropriately using cross-validation.*
- Once you determine the best number of perceptrons that the training data justifies, train your final neural network with appropriate model order to maximize data likelihood using all training data in the given set.
- Apply your trained neural network classifiers to the test dataset that you generated and used in the previous item where theoretically optimal MAP classifier was analyzed numerically. Report the test dataset probability error estimates for your neural network classifiers trained with different training dataset sizes. Discuss the effect of training set size on test dataset performance for these neural network models.

* Note that if we had not commited to a minimum-probability-of-error classifier design upfront, we could use log-likelihood of validation data as an appropriate model order selection objective in the cross-validation process.

Question 2 (50%)

Generate two-dimensional $\mathbf{x} = [x_1, x_2]^T$ samples with the attached Matlab script (the data is generated through iid sampling from a mixture of three Gaussians). Specifically generate 1000 samples for training and 10000 samples for testing.

Train and test a single hidden layer MLP function approximator to estimate the value of x_2 from the value of x_1 by minimizing the mean-squared-error (MSE) on the training set.

Using 10-fold cross-validation to select between logistic (sigmoid) and softplus (SmoothReLu) nonlinearities for the perceptrons in the hidden layer, as well as the number of perceptrons. Leave

the output layer linear (no nonlinearity). Once the best model architecture is identified using cross-validation, train the selected model with the entire training set. Apply the trained MLP to the test dataset. Estimate the test performance.

Explain your work clearly to inform the readers of all relevant details for reproducibility of your results, and to convince them that you have done everything correctly/properly, including a report of the following: (1) visual and numerical demonstrations of the cross-validation process indicating how the model selection was carried out; (2) visual and numerical demonstration of the performance of the trained model on the test data set.

Hint: $logistic(z) = 1/(1+e^{-z})$ & $softplus(z) = ln(1+e^{z})$