EE 559 **Homework 8** Posted: Mon., 4/10/2023
Chugg/Jenkins Due: Mon., 4/17/02023, 11:59 PM PDT

1. **Universal Function Approximation**

In this problem, we consider approximation of the function $f(x) = e^{-2x}\cos(4\pi x)$ defined on the interval $x \in [0,1]$ Specifically, consider an ANN with one input node, one hidden layer (using ReLu activations), and one output node (linear activation). We consider choosing the weights in this ANN approximation, as developed in lecture, to provide a piecewise linear approximation of $f(x)$. Specifically on the grid defined by $\mathcal{G}_{\mathcal{M}} = \{x_i = i/(M-1)\}_{i=0}^{M-1}$ this approximation is exact and it is a linear function in between these grid points.

a) Let each weight that is determined by the grid points be a parameter. How many parameters are in the hidden layer? How many total parameters in the ANN? Specify how each of the weights and biases in both layers should be chosen to obtain the approximation.

b) Let $\hat{f}_M(x)$ denote the approximation using the size $M$ grid $\mathcal{G}_M$. Produce a plot of $f(x)$ and $\hat{f}_M(x)$ for $M \in \{2,4,8,16\}$.

c) Consider the mean squared error of this approximation:

$$MSE(M) = \int_0^1 \left(f(x) - \hat{f}_M(x)\right)^2 dx.$$

Note that this is not the average sum of squared error on $\mathcal{G}_M$, which is zero. MSE can be computed by using a finer grid of points than $\mathcal{G}_{\mathcal{M}}$; use $G = 10000$ points. The normalized or relative MSE is

$$NMSE(M) = MSE(M)/E \text{ where } E \text{ is the energy in } f(x): E = \int_0^1 f^2(x)dx.$$

Produce a plot of $NMSE(M)$ $vs.$ $M$; more specifically, plot the $NMSE$ in dB – i.e., $NMSE_{dB}(M) = 10\log_{10}\left(NMSE(M)\right)$ $vs.$ $M$. Plot this for each integer $M \geq 2$ up to large enough $M$ so that the NMSE goes below -40 dB.

d) Interpretation of the NMSE results:

i. Are some values of $NMSE_{dB}(M)$ on your plot greater than 0? If so, explain what this means. **Hint:** What is the $NMSE_{dB}$ when the approximation is taken to be the zero function?

ii. Check if the $NMSE$ monotonically decreases. Produce plots of the form from part (b) for several consecutive values of $M$ on a range where the $NMSE$ is not monotonically decreasing (i.e., where it increases and then decreases). Explain why this occurs. In terms of a functional fit, what is the fit criterion function we have used?

iii. What is the minimum value of $M$ required so that the $NMSE_{dB}(M)$ is less than $-40\ dB$? How many parameters are in the network for this value of $M$?

e) – f) **[Extra credit]** Does the derivative of f-hat approximate the derivative of y? To find out, repeat parts b and c above, except use f-hat' = derivative of f-hat (for a given M), and compare with y' = derivative of y. For f-hat', start from the f-hat for a given M, and calculate algebraically it's derivative w.r.t. x (for each linear segment). For y-hat, calculate its

derivative y'(x) algebraically. Then both can be plotted by computer, and NMSE between f-hat' and y' can be calculated by computer.

g) **[Extra credit]** Does the second derivative of f-hat approximate the derivative of y? No need to derive this or to run it; but give an answer to the question and justify your answer.

2. **Whitening and Simulation of Gaussian Random Vectors**

In this problem, you will be generating scatter plots. For each of these plots, use the same range for the x and y axes. Also, use a square size for the plot – e.g., `plt.figure(figsize=(6,6))`. This will allow you to see the directional preference of the data.

(a) Consider a two-dimensional Gaussian random vector with mean vector and covariance matrix given by

$$\underline{m}_x = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\underline{\underline{\sum}}_x = \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}$$

Using `np.random.normal(0,1,.)` to generate i.i.d. standard normal realization, produce a scatter plot with 5000 realizations of the random vector $\vec{x}$. Explain your method.

(b) In this problem, we generate a white Gaussian vector in (i), then color it in (ii), and then whiten in 2 steps ((iii)-(iv)). Then we analyze the results in (v).

(i) Generate 5000 realizations of a zero-mean, white, Gaussian random vector $\underline{w}$. Plot a scatter plot of these realizations. A white random vector is one with identity covariance matrix and zero mean.

(ii) Let $y = \underline{\underline{A}}\underline{w}$ where $\underline{\underline{A}} = \begin{bmatrix} -1 & -1 \\ 2 & 4 \end{bmatrix}$. What are the mean and variances of the two components of $\underline{y}$? What is the correlation coefficient $r_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$ between these two components? Are the positively or negatively correlated? Produce a scatter plot for the values of $\underline{y}$ obtained from the 5000 realizations of $\underline{w}$. Compute the sample covariance matrix using these realizations and compare this to the covariance matrix of $\underline{y}$ as derived.

(iii) Consider decorrelating, then whitening $\underline{y}$. First, define $\underline{v} = \underline{\underline{E}}^T \underline{y}$, where $\underline{\underline{E}}$ is the 2 x 2 matrix of orthonormal eigen-vectors of the covariance matrix of $\underline{y}$. What is the covariance matrix of $\underline{v}$? Produce a scatter plot of the corresponding 5000 realizations of $\underline{v}$, produced by $\underline{v} = \underline{\underline{E}}^T \underline{y}$ using the realization of $\underline{y}$ from the previous part.

(iv) Let $\underline{z} = \underline{\underline{\Lambda}}^{-1/2} \underline{v}$ What is the covariance matrix of $\underline{z}$? Produce a scatter plot of the corresponding 5000 realizations of $\underline{z}$, produced by $\underline{z} = \underline{\underline{\Lambda}}^{-1/2} \underline{v}$ . using the realization of $\underline{v}$ from the previous part.

(v) Discuss the relationship between $\underline{w}$ and $\underline{z}$. Specifically, answer the following questions. What is the pdf of $\underline{w}$ and the pdf of $\underline{z}$? Is $\underline{z} = \underline{w}$ -- i.e., since we colored $\underline{w}$ to get $\underline{y}$, then we whitened $\underline{y}$ to get $\underline{z}$? If not explain why.

3. **Comparison of PCA and MDA on wine dataset.**

*Comment*: you will probably find that solving this problem is shorter than it looks at first.

In this problem you will use the UCI Wine dataset [1], which gives as features results of a chemical analysis of wines derived from 3 different cultivars (varieties) of grape. The goal is to predict the cultivar from the chemical analysis of the wine. It has 3 classes and 13 features.

You can use the csv data file posted on piazza with this homework assignment (it is the same as the UCI dataset except re-formatted to csv). You may use sklearn, NumPy, and matplotlib as you like for this problem.

For your cross-validation runs below, use the entire provided dataset to divide into training and validation sets for each cross-validation loop. We will not need a separate test set in this problem.

For the parts below, you will use the original (unnormalized) dataset for some runs, and use a standardized version of the dataset for other runs. For this problem, you can standardize the whole dataset using parameters calculated from the whole dataset; then store it for future use.

Normally, it's best practice to standardize the dataset within the cross validation loop, so that the dataset can be standardized using parameters calculated from only the training data. In this problem each cross-validation fold has only 5% of data points in the validation set, so they will on average have only a small effect on the normalization; and we are only using the results to compare across different methods, not to get a best estimate of performance on unknowns. So we are taking the simpler approach of standardizing the dataset once at the beginning.

[1] https://archive.ics.uci.edu/ml/datasets/Wine

(a) Baseline for comparison.

First standardize the dataset.

(i) We will pick 2 pairs of features to look at the data as follows: plot the data projected into $x_1, x_2$ space, and also plot the data instead projected into $x_1, x_6$ space. For both plots, use 3 different symbols to denote data points belonging to class 1, 2, and 3.

(ii) Run a multiclass perceptron classifier on the 2D data using only features $x_1, x_2$. For each run, first shuffle the data, then use 20-fold cross validation, and compute the mean classification error rate over the 20 folds. Also store the weight values from the result of the first fold.

Do a total of 5 runs. Report the mean classification error rate from each cross-val run, and also report the average and standard deviation of the mean classification error over the 5 runs.

Also report 2 plots: for the run with the lowest classification error rate, plot the results in a scatter plot (labelled data points of the entire dataset, decision boundaries and regions based on the stored weights resulting from the first fold); then produce another similar plot for the run with the highest classification error rate.

(iii) Repeat part (ii) for $x_1, x_6$.

(b) PCA based on unnormalized dataset.

For this part, use the original dataset with no standardization.

(i) Run PCA, reducing to 2 dimensions, on the entire dataset. Plot the data projected into the 2D space, using the same symbols as in part (a) to denote the class label of each data point. Compared with the baseline of (a), do you expect a better classification result with PCA?

(ii) Repeat (a)(ii) except for the 2 new features resulting from PCA (plot everything in the new $(\tilde{x}_1, \tilde{x}_2)$ space).

(iii) How does it compare with the baselines in (a)(ii)?

(c) PCA based on standardized dataset.

Repeat (b)(i)-(ii) except first standardize the data.

(iii) How does PCA with standardized data compare with PCA using unnormalized data? Why?

(d) MDA (using LDA as an approximation to MDA).

For this part, first standardize the data.

(i) Because this is a 3-class problem, MDA (rather than FLD) is appropriate to use. We will implement this using instead linear discriminant analysis (LDA), which is similar to MDA (more detail on the similarities and differences will be given in a future lecture).

You can use
`sklearn.discriminant_analysis.LinearDiscriminantAnalysis`.

on the entire dataset to reduce the dimensionality to 2. Plot the data projected into the 2D space, using the same symbols as in part (a) to denote the class label of each data point. How do you expect a linear classifier to do on this data compared with (b) and (a) above?

(ii) Repeat (a)(ii) except for the 2 new features resulting from MDA/LDA (plot everything in the new $(\tilde{x}_1, \tilde{x}_2)$ space).

(iii) How does it compare with the baselines in (a)(ii) and PCA in (b)(ii)?

4. **Mahalanobis distance and Bayes classification for minimum error**

In a 2-class classification problem, the class conditional densities are:

$$\mathrm{p}\left(\underline{x}\,\middle|\,S_i\right) = \mathrm{N}\left(\underline{x}\,\middle|\,\underline{m}_i, \underline{\underline{\Sigma}}_i\right),\ \ i = 1{,}2$$

(a) Given the following class means and covariance matrices for a 2D problem:

$$\underline{m}_1 = \begin{pmatrix} 2 \\ 1 \end{pmatrix},\ \ \underline{m}_2 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

$$\underline{\underline{\Sigma}}_1 = \begin{pmatrix} 1 & -1 \\ -1 & 4 \end{pmatrix},\ \ \underline{\underline{\Sigma}}_2 = \begin{pmatrix} 4 & 0 \\ 0 & 1 \end{pmatrix}$$

On a single 2D plot, plot the 2 class means, and make a filled-contour plot of constant Mahalanobis distances from the mean for each class, as follows: $d_M^2\left(\underline{x}, \underline{m}_1\right) = B^2$ and $d_M^2\left(\underline{x}, \underline{m}_2\right) = B^2$, for values $B = 0.5, 1.0, 1.5, 2.0$. (End result should be 4 curves for $d_M^2\left(\underline{x}, \underline{m}_1\right) = B^2$ and 4 curves for $d_M^2\left(\underline{x}, \underline{m}_2\right) = B^2$, with colored fills between the curves.)

**Tips:** use `matplotlib.pyplot.contourf` for the filled-contour plot; use the `levels` parameter to input the values of $B^2$; be mindful of $B$ vs. $B^2$.

See https://www.tutorialspoint.com/matplotlib/matplotlib_contour_plot.htm and https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.contourf.html for more info.

(b) Suppose we implement a Bayes classifier for minimum error, with decision rule:

$$\ln\left[p\left(\underline{x}\,\middle|\,S_1\right)P(S_1)\right] > \ln\left[p\left(\underline{x}\,\middle|\,S_2\right)P(S_2)\right] \ \Rightarrow\ \underline{x} \in \Gamma_1$$

$$\ln\left[p\left(\underline{x}\,\middle|\,S_1\right)P(S_1)\right] < \ln\left[p\left(\underline{x}\,\middle|\,S_2\right)P(S_2)\right] \ \Rightarrow\ \underline{x} \in \Gamma_2$$

Give an expression for the decision boundary for the normal densities given at the beginning of this problem, by plugging in for $N\left(\underline{x}\,\middle|\,\underline{m}_i, \underline{\underline{\Sigma}}_i\right),\ i = 1{,}2$. Give your answer in terms of $d_M\left(\underline{x}, \underline{m}_i\right)$, $\underline{\underline{\Sigma}}_i$, $P(S_i),\ i = 1{,}2$.

(c) For the means and covariance matrices given in (a), use your answer to (b) to plot the decision regions and boundaries of the Bayes classifier for minimum error. Also show the class means on the plot. Do this for 3 cases:

(i) $P(S_1) = P(S_2) = 0.5$

(ii) $\mathrm{P}(\mathrm{S}_1) = 0.3,\ \mathrm{P}(\mathrm{S}_2) = 0.7$

(iii) $P(S_1) = 0.1,\ P(S_2) = 0.9$

**Tip:** you can use `contourf` for this as well, or another routine of your choice.