1. **10 Points**

Create a model to optimize prediction of handwritten digits (MNIST dataset). Evaluate the results and tune the parameters to achieve better predictions.

Hyperbolic Tangent Activation Function

The hyperbolic tangent activation function is a differentiable and monotonic function that ranges from -1 to 1. It is used for classification between two classes. This can be used as the first layer of a model followed by another layer for classification of more than two classes such as Softmax.

$$Tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{1}$$

Softmax Activation Function

The Softmax activation function is a normalized exponential that generalizes the sigmoid function to multiple dimensions. It is used for classification of more than two classes with a probability distribution over the output classes.

$$\sigma(\mathbf{x})_i = \frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}} \quad \mathbf{x} = (x_1, x_2, ...x_N) \tag{2}$$

a. Load the MNIST dataset and split the data into a training set and testing set.

b. Display the first three images of the test data.

c. Display the first three images of the training data.

d. Create a model with two dense layers using the tanh and softmax activation functions. Compile the model using the Stochastic Gradient Descent optimizer using the Mean Squared Error loss function.

e. Calculate the confusion matrix of this data after 1 epoch.

f. Do this again, but with 3 epochs and splitting the data into 10 segments.

g. Repeat the process, but this time use the Sparse Categorical Crossentropy as the loss function.

2. **10 Points**

The following random set of numbers $\mathbf{X}$ are $\mathbf{x}_i$ for i=1,2,...10 where $\mathbf{x}_k = (x_1,\ x_2)$ and the mean of $\mathbf{X}$ is $\boldsymbol{\mu} = (\mu_1, \mu_2)$:

$\mathbf{x}_1$ = (-4, -4)
$\mathbf{x}_2$ = (1, 0)
$\mathbf{x}_3$ = (2, 3)
$\mathbf{x}_4$ = (-1, 2)
$\mathbf{x}_5$ = (-2, -2)
$\mathbf{x}_6$ = (-4, -1)
$\mathbf{x}_7$ = (-2, -1)
$\mathbf{x}_8$ = (-5, -5)
$\mathbf{x}_9$ = (-2, -3)
$\mathbf{x}_{10}$ = (4, 4)

a. Calculate for k=1, 2, 3:

$$d_i = ||\mathbf{x}_k - \mathbf{x}_i||^2 \qquad \forall \mathbf{x}_i = (1, 2, ..., 10) \tag{3}$$

b. Based on the distance of each $\mathbf{x}_i$ from each $\mathbf{x}_k$, assign them into the cluster with $\mathbf{x}_k$ as the center.

c. Calculate the standard deviation or squared error of each point in the clusters relative to their mean, where n is the total number of points in each cluster.

$$\sigma_k^2 = \sum_{j=1}^{n} ||\mathbf{x}_j - \boldsymbol{\mu}_k||^2 \tag{4}$$

d. Recalculate the center of each cluster and repeat the process for another iteration.

e. Will another iteration make a difference? How do you know?

f. Use a random number generator to generate 20 new sets of numbers and repeat the process.

3. **10 Points**

Generate three clusters with 200 points each with a standard normal distribution but with the following variance (σ) and means (μ):

σ_1 = (1.2,0.8)	μ_1 = (-2,-2)
σ_2 = (0.9, 0.7)	μ_2 = (0,0)
σ_3 = (0.8, 0.5)	μ_3 = (3,4)

a. Plot the three clusters with different colors for each to show the desired response.

b. Calculate the K-by-K cross-correlation function for the three clusters. The initial cluster centers can be any three points in the dataset selected at random.

K-by-K Correlation Function

The K-by-K correlation function of the hidden layer output is defined by $\mathbf{R}(n)$ *where the distance metric used to calculate the output layer is the Standard Euclidean distance* $\phi(x_i, \mu_k)$ *from the cluster center:*

$$\mathbf{R}(n) = \sum_{i=1}^{n} \mathbf{\Phi}(\mathbf{x}_i)\mathbf{\Phi}^T(\mathbf{x}_i) \qquad \textit{where} \tag{5}$$

$$\mathbf{\Phi}(\mathbf{x}_i) = [\phi(x_i, \mu_1), ...\phi(x_i, \mu_K)] \qquad \textit{and} \tag{6}$$

$$\phi(x_i, \mu_k) = exp\left(-\frac{||x_i - \mu_k||^2}{2\sigma_k^2}\right) \quad k = 1, 2, ...K \quad (Number \quad of \quad clusters) \tag{7}$$

c. Plot the cluster assignments along with the centroids of the three clusters for this initial iteration.

d. Repeat for one more iteration and calculate the prior estimation error.

Prior Estimation Error

The prior estimation error $\alpha(n)$ *is based on the old estimate of the weight vector and its distance from the desired response at iteration* i*:*

$$\hat{\mathbf{w}}(n) = \hat{\mathbf{w}}(n-1) + \mathbf{R}^{-1}(n)\mathbf{\Phi}(n)\alpha(n) \tag{8}$$

$$\alpha(n) = d(n) - \mathbf{w}(n-1)\mathbf{\Phi}^T(n) \tag{9}$$

4. **10 Points**

Generate a sample of data for N = 50 points and variance $\sigma = 50$.

a. Plot the points.

b. Calculate the gradient descent for a line through the sample data points with a learning rate $\eta = 0.00001$ for 10,000 iterations.

c. Plot the points and the best fit line that was calculated using Gradient Descent.

d. Repeat the process for N = 100, 200, and 500 and for $\sigma = 100, 50, 300$.

e. What happens when you increase the learning rate from $\eta = 0.00001$ to $\eta = 0.0001$, then again to $\eta = 0.001$?

f. Plot the results at the 10th, 100th, and 1000th iterations.

5. **10 Points**

Generate a surface with the equation:

$$Z = 0.1X^3 + Y^2$$

For values of $X, Y = [-2, 2]$.

a. Plot the surface.

b. Implement the Steepest Descent algorithm to fine the minimum on this surface with an initial starting point of $X_0 = 1.5$ and $Y_0 = 1.8$.

c. What are the values of (X_1, Y_1) and X_2, Y_2.

d. How many iterations does it take for the values to converge such that $\epsilon < 0.0001$ where ϵ is the change in the value between X_n, Y_n and X_{n-1}, Y_{n-1}.

6. **10 Points**

Load the MNIST dataset again, or use the previously loaded training and testing data for MNIST. Select N = 15,000 sample points from either the training or testing data or a combination of both.

a. Calculate the Eigenvalue-Eigenvector pairs of the variance-covariance matrix of the sample data $\mathbf{X}$.

Variance-Covariance Matrix

The Variance-Covariance Matrix of a dataset $\mathbf{X}$ *is*$\nprec$.

$$\mathbf{\Sigma} = Var(\mathbf{X}) = \mathbb{E}[(\mathbf{X} - \mu)(\mathbf{X} - \mu)^T] \tag{10}$$

Eigenvalue-Eigenvector Pair

The Eigenvalue-Eigenvector Pair of $\mathbf{\Sigma}$ *are the eigenvectors normalized by their eigenvalues.*

$$(\boldsymbol{\lambda}, \mathbf{e}) = ([\lambda_1, \lambda_2, ...\lambda_p], [[\mathbf{e_1}, \mathbf{e_2}, ..., \mathbf{e_p}]) \tag{11}$$

They are ordered so that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p \geq 0$.

b. What is the shape of the variance-covariance matrix?

c. Express the MNIST data using the top two principal components and show a plot of the 1st PC against the 2nd PC.

d. Is this sufficient to represent the MNIST data?

7. **10 Points**

Load the MNIST dataset again, or use the previously loaded training and testing data for MNIST. Select N = 15,000 sample points from either the training or testing data or a combination of both.

a. Use the Logistic Hyperbolic Cosine function as the linear transformation function **W** to calculate the independent components of the selected MNIST data. Use a tolerance of 0.00001.

Independent Component Analysis

Independent Component Analysis finds the linear transform $\mathbf{W} = (w_1, w_2, ..., w_p)$ *for a dataset* $\mathbf{X}$ *so that* $\mathbf{S} = \mathbf{WX}$*. The independent components are* $\mathbf{A} = (a_1, a_2, ..., a_p)$ *and the dataset* $\mathbf{X}$ *is expressed as:*

$$X_i = a_{i,1}S_1 = a_{i,2}S_2 + ... + a_{i,p}S_p \tag{12}$$

b. Express the MNIST data using the top two independent components and show a plot of the 1st IC against the 2nd IC.

c. How long does it take the ICA algorithm to run, compared to PCA?

d. Is there better separation for the MNIST data when using ICA?

e. Is two components sufficient to represent the MNIST dataset when using ICA?

8. **20 Points**

Use the separated MNIST training and testing data with the model created in Problem 1.

a. Calculate the PCA for the training and testing data and run the model from Problem 1 using the Sparse Categorical Crossentropy as the loss function.

b. Calculate the confusion matrix of this data after 3 epochs.

c. Repeat part a. but use the ICA dimension reduction method.

d. Calculate the confusion matrix for this data after 3 epochs.

e. Are there parameters you can tune to improve results for the data after using PCA or ICA?

f. Use the k-Means clustering algorithm implemented in Problem 3 to cluster the MNIST data into 10 clusters.

g. Display the data assigned to each cluster.

h. Repeat the process with the data after PCA.

i. Repeat the process with the data after ICA.

j. What are some of the parameters you can tune in each step to get better results.