A. Component analysis:

1. We see the following 5 data points:

\[
\begin{align*}
\mathbf{x}^1 &= \begin{bmatrix} 0.1 & 2 & -2.3 \end{bmatrix} \\
\mathbf{x}^2 &= \begin{bmatrix} -1.1 & 3.5 & -2.6 \end{bmatrix} \\
\mathbf{x}^3 &= \begin{bmatrix} 1.5 & -4 & 12.5 \end{bmatrix} \\
\mathbf{x}^4 &= \begin{bmatrix} 0.8 & -5 & 3.6 \end{bmatrix} \\
\mathbf{x}^5 &= \begin{bmatrix} -0.9 & -7.3 & 13.8 \end{bmatrix}
\end{align*}
\]

Which of the following would you expect to be the first principal component?

(a) \[
\begin{bmatrix} 1 \\ 0.15 \\ 0 \\ 0 \end{bmatrix}
\]

(b) \[
\begin{bmatrix} 0 \\ 0.4 \\ 0.8 \\ 0.4 \end{bmatrix}
\]

(c) \[
\begin{bmatrix} 0 \\ 0 \\ -0.5 \\ 0.9 \end{bmatrix}
\]

(d) \[
\begin{bmatrix} -0.15 \\ 0.9 \\ -0.3 \\ -0.25 \end{bmatrix}
\]

2. For each of the following sets of three components, label whether you expect them to come from PCA, ICA, or NMF?

(a) \[
\begin{bmatrix} 0.5 \\ 0 \\ 0.3 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.2 \\ 0.8 \\ 0 \\ 0.6 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0.8 \end{bmatrix}
\]

(b) \[
\begin{bmatrix} 0.7 \\ 0.3 \\ 0.2 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0.6 \\ -0.6 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.6 \\ -1 \\ 0 \end{bmatrix}
\]

(c) \[
\begin{bmatrix} -0.7 \\ 0.5 \\ 0.8 \\ 0.5 \end{bmatrix}, \begin{bmatrix} 0.6 \\ 0 \\ -0.8 \\ 0 \end{bmatrix}, \begin{bmatrix} -0.4 \\ 0 \\ 0.4 \end{bmatrix}
\]
3. Presume the following are principal components:

\[
\mathbf{u}^1 = \begin{bmatrix} 0 \\ 0.67 \\ 0.67 \end{bmatrix}, \quad \mathbf{u}^2 = \begin{bmatrix} -0.64 \\ -0.51 \\ 0.26 \end{bmatrix}, \quad \mathbf{u}^3 = \begin{bmatrix} 0.77 \\ -0.43 \\ 0.21 \end{bmatrix}
\]

a. What are the corresponding three PCA weights \( z_{qi} \) for each \( x^i \) below?

\[
\begin{align*}
\mathbf{x}^1 &= \begin{bmatrix} -3 \\ 0 \\ 5 \end{bmatrix} \\
\mathbf{x}^2 &= \begin{bmatrix} 2.5 \\ 1.2 \\ -2 \end{bmatrix} \\
\mathbf{x}^3 &= \begin{bmatrix} 4.5 \\ 6 \\ -2 \end{bmatrix}
\end{align*}
\]

b. Use the first two principal components to estimate \( x^3 \) and \( x^2 \).

4. Presume the following are independent components:

\[
\mathbf{u}^1 = \begin{bmatrix} 0.55 \\ 0 \\ -0.83 \end{bmatrix}, \quad \mathbf{u}^2 = \begin{bmatrix} 0.25 \\ -0.97 \\ 0 \end{bmatrix}, \quad \mathbf{u}^3 = \begin{bmatrix} 0.45 \\ 0.45 \\ 0.77 \end{bmatrix}
\]

a. Identify the one independent component that best describes each data point \( x^i \) below. (By “best describes,” I mean we can create a low-error reconstruction of \( x^i \) using just \( u^q \) and the corresponding \( z_{qi} \).)

\[
\begin{align*}
\mathbf{x}^1 &= \begin{bmatrix} -5.2 \\ 1 \\ 5 \end{bmatrix} \\
\mathbf{x}^2 &= \begin{bmatrix} -1 \\ 3 \\ 6 \end{bmatrix}
\end{align*}
\]
B. Neural networks

1. Let us assume we have a neural network with three layers. Layer 1 has 6 units, layer 2 has 4 units, and layer 3 has a single unit. There are 9 features fed into the units in layer 1.

a. Assuming we also have a unit-specific constant $b_k$ offset for each unit, how many parameters must we learn for the network as described.

We establish a measurement of “likelihood” derived from the error $(r_{1,1}^3 - y)^2$. Using this likelihood measure on a training data set and using additional variables, we observe the AIC for our current 3-layer neural network model is: -30.5.

b. If we add 2 more units to layer 2 and the likelihood remains unchanged, how will the AIC be affected?

c. How does the likelihood have to change to preserve the original AIC=-30.5 measure? Be as specific as possible, e.g. “the likelihood has to go up by 10” or “the likelihood has to decrease by 50%” or “no change to the likelihood is needed.”

2. Presume the following Neural Network, where each unit performs the standard dot product (weighted sum) and sigmoid transformation.

![Neural Network Diagram](image)

The following feature values are input: $x_1=0.5$, $x_2=0.9$.

The original weights are: $w_{1,1}^1=3$, $w_{1,2}^1=-4$, $w_{2,1}^2=-2$, $w_{2,2}^1=1$, $w_{3,1}^1=4$, $w_{3,2}^1=-1$, $w_{1,1}^2=-3$, $w_{1,2}^2=5$, $w_{1,3}^2=0$, $w_{1,1}^3=0.5$, $w_{2,1}^3=-1$.

Assume all $b$’s are 0.

The resulting unit outputs are: $r_1^1=0.1$, $r_2^1=0.5$, $r_3^1=0.8$, $r_1^2=0.9$, $r_3^2=?$, $r_2^3=?$.
a. Compute $r_1^3$ and $r_2^3$

The desired neural network output at the top layer are: top unit 1 = 0; top unit 2 = 1

b. Compute the updates to the following weights, assuming $\epsilon = 0.1$:

\[
\begin{align*}
    w_{1,1}^3 &= \quad w_{2,1}^3 &= \quad w_{1,1}^2 &= \quad w_{1,2}^2 &= \\
\end{align*}
\]

Let us now replace the sum-and-sigmoid units in the current network with linear-rectifier units using the function to the right. We will keep the initial weights provided at the start of question 2.

The following feature values are input:

\[
\begin{align*}
    x_1 &= 0.3 \quad & x_2 &= 0.5 \\
\end{align*}
\]

Compute the outputs:

\[
\begin{align*}
    r_1^1 &= \quad r_2^1 &= \quad r_1^2 &= \quad r_1^3 &= \\
\end{align*}
\]

C. Programming

In this question, you will work with two data sets. You will implement and test two methods: CNNs and feature selection using Naïve Bayes classification.

To submit Part C, create the directory HW3 inside your private/CIS5800 directory. Leave all relevant pieces of code in your private/CIS5800/HW3 directory.

First, you will implement the 3-layer neural network shown to the right. Each unit computes a weighted sum and then applies the sigmoid function. You will represent the weights for each layer in a separate matrix – layer1W, layer2W, layer3W. Each column will correspond to the weights for a single unit. So, layer1W will have dimensions 5x2 – 2 units each taking in inputs from 4 features plus a constant b offset.

1. Write a function `feedforward` that takes in the features $\mathbf{x}$ and the network weights, and outputs the responses from layer 3.

Specifically: you will be able to call the function as
layer3Out = feedforward(x, layer1W, layer2W, layer3W);

x is a vector containing 4 input features, the layer?W matrices are as described above. layer3Out will be a vector of two numbers for the outputs from layer 3 r^3_1 and r^3_2.

Use of matrix mathematics and/or the dot command may help you in this function. You also may use a sigmoid function you wrote for a previous homework, or that is already available for Matlab.

2. Write a function backpropTop that takes in the features x, the current network weights, and the set of desired outputs. It will output the updated weights for only layer 3. Specifically: you will be able to call the function as

newLayer3W = backpropTop(x, y, layer1W, layer2W, layer3W);

x is a vector containing 4 input features, y is a vector containing the desired outputs for the 2 units in layer 3 [r^3_1 r^3_2], and the layer?W matrices are as described above. newLayer3W will be a matrix of the updated weights for the two units in layer 3.

For the updates, assume = 0.1.

Accessing our data
For questions 3-5, the file letter-recognition.mat is available on our website (and on erdos using cp ~dleeds/MLpublic/letter-recognition.mat). Load each file into your Matlab session to get access to the trainData and testData matrices. For each matrix, each row is one example data point. The first column represents the class.


We will classify letters A and B based on the 16 features in letter-recognition.mat.

Assume each of the 16 features is independent and each feature comes from a Gaussian distribution N(μ, σ) with class-specific mean and standard deviation. For example, consider feature 10 for the letter ‘A’. The probability of feature 10 for one data point x can be described by:

P(x_{10} | y = A) = N(μ_{10}^A, σ_{10}^A)

3. From the training data, compute the mean and standard deviation for each feature for each of the 2 classes. Report your answer in a comment inside nbClassify.m, which you will define further for the next question. (You may find these answers with relatively simple mean and std commands on Matlab.)
4. Implement a Naïve Bayes classifier to classify letters A and B based on the 16 features in letter-recognition.mat. **Assume letters A and B have equal prior probabilities.** You will need to use the mean and standard deviation values computed in Question C3 as well.

Specifically: you will be able to call the function as  
\[
\text{classList} = \text{nbClassify}(xFeatures, A\text{means}, A\text{stds}, B\text{means}, B\text{stds})
\]

`xFeatures` is a vector of the 16 feature values for a single data point, and the remaining inputs are the feature means and standard deviations for classes A and B. `A\text{means}`, `A\text{stds}`, `B\text{means}`, and `B\text{stds}` each will be a vector containing 16 values. `classList` will be either 'A' or 'B', determined to be the most probable letter based on the input.

5. Implement a function to perform feature selection, starting by selecting the 1 best feature and adding features in the manner covered in class until testing error begins to increase. **Note, I ask you to use testing error as your termination criterion in this assignment.**

Specifically: you will be able to call the function as  
\[
\text{featsOut} = \text{featSelect}(\text{testData}, A\text{means}, A\text{stds}, B\text{means}, B\text{stds})
\]

`testData` is a matrix holding the labels and all 16 feature values for all the test data in the format provided in letter-recognition.mat. The remaining inputs are the features mean and standard deviations for classes A and B, as specified in question C4. `featsOut` is the output, which is the vector of features chosen once `featSelect` has reached its termination condition. For example, if features 5 and 10 were selected, `featSelect` will return [5 10].

Report your selected features in a comment inside `featSelect.m`.

**Note 1:** Keeping track of selected and not-yet-selected features can be tricky task! Start early.

**Note 2:**
If you have a vector \( A = [2 \ 5 \ 10 \ 34 \ 12] \) and then run in Matlab the command \( A(3)=[] \), the third element of A will be deleted, leaving you \( A = [2 \ 5 \ 34 \ 12] \).