This HW covers units 1 and 2. Each question is worth 20%.

Question 1: Consider the CIFAR-10 dataset. In this problem you will create a basic (not so powerful) handcrafted classifier that attempts to distinguish between ships and frogs based on aggregate information only (it does not consider the spatial location of pixels). You will optimize the single parameter on the training set and then evaluate the classifier's performance on the test set. The question contains a few guiding items.

1. Load the training dataset and separate it into positive images (frogs) and negative images (ships). Then consider the R, G, and B for each of the two classes and summarize the mean R, G, and B, pixel intensity for each class. That is, display six means, where each mean is taken over the $5000 \times 32 \times 32$ pixels of the respective color component and class.

2. Since the R component is quite similar for both classes your classifier will focus mostly on the G and B component. For this we will create a grey scale image where each grey pixel is determined via,

$$\text{Grey} = 0.1 \times \text{Red} + (1 - 0.1 - \beta) \times \text{Green} + \beta \times \text{Blue},$$

   and β is a parameter in the range $[0, 0.9]$ indicating the weight of the blue pixels.

   Momentarily fix $\beta = 0.5$ and create the grey scaled images for both the ships and the frogs. For each grey scaled image, calculate mean grey scaled pixel intensity (over 32×32 pixels) and the standard deviation grey scaled pixel intensity (again over 32×32 pixels). Now plot a scatter plot of mean vs. standard deviation where the points (mean and standard deviation pairs) matching the 5000 frog images are marked with one color and the points matching the 5000 ship images are marked with a different color.

3. Now consider a binary classifier with parameter γ.

$$\hat{f}(x) = \begin{cases} +1, & \text{mean}_x^2 + \text{std}_x^2 \leq \gamma^2, \\ -1, & \text{mean}_x^2 + \text{std}_x^2 > \gamma^2. \end{cases}$$

   This classifier determines "frog" (+1) if the mean and standard deviation of the image fall within the unit circle of radius γ. Otherwise, it determines "ship" (-1). Set $\gamma = 0.5$ and $\beta = 0.5$ and determine the performance of this classifier on the training set. Specifically what is the precision, recall, and F_1 score?

4. Now optimize with respect to the F_1 score by selecting the best β and γ parameters. This can be done via a grid search or any other means. Determine the precision, recall, and F_1 score (evaluated on the training set) and specify the β and γ parameters you found.

5. Now evaluate the performance of the classifier (with optimized parameters) on the test set.

Question 2: Continue with the CIFAR-10 dataset, but now use a linear classifier similarly to the linear (multi-class one-vs-rest) classifier used for MNIST in the lectures. Note that there are $32 \times 32 \times 3 + 1 = 3073$ features in this case. Use training data of lengths $1,000$, $10,000$, and $50,000$ progressively (starting with the smaller sizes and then increasing) to learn a linear classifier. In each case, evaluate the accuracy of the classifier on the test set (has 10000 images) and for the case of $50,000$ training images compute the confusion matrix as well.

Question 3: Consider scalar data-points $x_1, \ldots, x_n$ and labels $y_1, \ldots, y_n$. Assume we are considering a very basic regression model,

$$y = \theta x,$$

where θ is a scalar parameter (this is simple linear regression with no intercept term). Consider the loss function,

$$L(\theta) = \sum_{i=1}^{n} (y_i - \theta x_i)^2,$$

and assume we are minimizing the loss with respect to θ via gradient descent of the form,

$$\theta_{k+1} = \theta_k - \eta \nabla L(\theta_k), \tag{1}$$

where $\eta > 0$ is the learning rate and $\nabla L(z)$ is the gradient at the point z (in this case the derivative).

1. Compute $\nabla L(z)$ and represent it in terms of U_{xx} and U_{xy} where,

$$U_{xx} = \sum_{i=1}^{n} x_i^2, \qquad \text{and} \qquad U_{xy} = \sum_{i=1}^{n} x_i y_i.$$

2. Now represent the gradient descent equation (1) as,

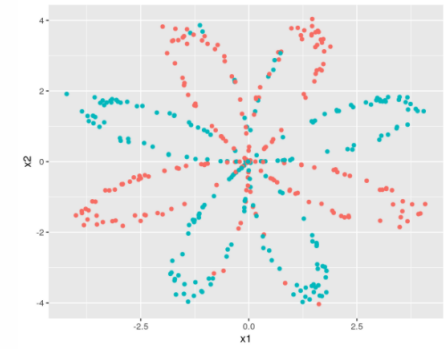
$$\theta_{k+1} = a\theta_k + b.$$

   What are a and b?

3. For what values of η does convergence occur?

4. Consider the dataset with x values $1, 2, 3, 4, 5$ and y values $0.9, 2.1, 3.1, 3.9, 5.1$. Write a short script that finds the best θ using gradient descent and a learning rate that is 90% of the maximally possible learning rate. Plot your model (line of best fit) and also plot the trajectory of θ_k during the learning process.

Question 4: Consider the "flower" from the following image.



The data from this image correspond to: two features x_1 and x_2 and one binary outcome y (0 for red dots and 1 for blue dots). An extract of the first 6 dots are presenting in the following table:

	x1	x2	y
	<dbl>	<dbl>	<dbl>
1	-1.71	2.78	0
2	-0.652	0.0109	1
3	-0.768	0.888	0
4	-2.75	-0.568	0
5	3.90	1.43	1
6	-0.0427	0.983	1

.

1. Load the dataset and split it into training and test sets. You should have 80% of your data in your train set and the remaining 20% in your test set. Carry out a statistical comparison of your choice between the distribution of the two classes in both the training set and the test set, aiming to show that the sets are randomly chosen.

2. Fit a logistic model to the training set using a generalized linear model (from any software) to create a binary classifier using the train data.

3. Evaluate the performance of your classifier on the test set. You should provide the confusion matrix as well as the F_1 score.

4. Now using first principles (not using any specific packages), build a shallow neural network (without any hidden layer) using the cross-entropy loss. Remember that this neural network is equivalent to the logistic model. Your model (algorithm) should follow these steps:

   (a) Initialize the model's parameters.

   (b) Loop:
   - Implement forward propagation.
   - Compute loss.
   - Implement backward propagation to get the gradients.
   - Update parameters (gradient descent).

5. Train your model using the training data. Provide a plot of the loss function during training to illustrate convergence of your model.

6. Evaluate the performance of your classifier on the test set. You should provide the confusion matrix and F_1 score and compare with the results of item 3 above.

Question 5: This question deals with convexity.

1. Consider m samples $\{(x_1, y_2), \ldots, (x_m, y_m)\}$ with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Show that the following function $f(w)$ is convex:

$$f(w) = \sum_{i=1}^{m} (y_i - w^T x_i)^2 + \gamma ||w||_2^2,$$

   where $w \in \mathbb{R}^d$ and $\gamma > 0$.

2. Solve this optimization problem,

$$\min_{w \in \mathbb{R}^d} f(w),$$

   by expressing the minimizer w in terms of the data matrix of x and the vector y.

3. Consider now that y_i is a discrete variable which can take one of the values $y \in \{1, \ldots, K\}$. We will consider the softmax regression (multinomial logistic regression) for this dataset. Specify the cost function, $J(w, b)$ (based on the cross-entropy loss) for this model.

4. Show that this cost function $J(w, b)$ is a convex function.

5. Show that in the case of $K = 2$ the softmax regression reduces to logistic regression.