Intro to deep learning

Assignment 2

Jordan Diaz

Z23554771

Google Colab Link

↓

https://colab.research.google.com/drive/1eruRxcR9nSlYN3CAAC11AeLSVC_fKuZB?usp=sharing

Problem 1

a) Calculate and show the confusion matrix

		Predicted 1	Predicted 0
Actual	1	19	6
	0	6	9

TP = 19 TN = 9
FP = 6 FN = 6

b) Calculate and provide the accuracy of the model

$$\text{Accuracy} = \frac{TP + TN}{TP+TN+FP+FN} = \frac{19+9}{19+9+6+6} = \frac{28}{40} = \boxed{70.00\%}$$

c) Calculate the sensitivity of the model

$$\text{Sensitivity} = \frac{TP}{TP+FN} = \frac{19}{19+6} = \boxed{76.00\%}$$

d) Calculate the specificity of the model

$$\text{Specificity} = \frac{TN}{TN+FP} = \frac{9}{9+6} = \frac{9}{15} = \boxed{60.00\%}$$

e) Calculate the F1 measure of the model

$$F1 = \frac{2 * (\text{Sensitivity} * \text{Precision})}{\text{Sensitivity} + \text{Precision}} = \frac{2(0.76 \cdot 0.76)}{0.76+0.76} = \boxed{76.00\%}$$

f) Calculate the Precision

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{19}{25} = \boxed{76.00\%}$$

Problem 2

a) Calculate and show the confusion matrix

Actual \ Predicted	1	0
1	7	2
0	3	5

TP = 7 TN = 5
FP = 3 FN = 2

b) Calculate and report Accuracy, Sensitivity, Specificity

Accuracy = $\frac{7+5}{17}$ = $\boxed{70.59\%}$, Sensitivity = $\frac{7}{7+2}$ = $\boxed{77.78\%}$

Specificity = $\frac{5}{5+3}$ = $\boxed{62.50\%}$

c) calculate and report the perceptron classifier values of w_0, w_1 and, w_2

$V = x_0 w_0 + x_1 w_1 + x_2 w_2$, $V = 0$

Samples: $(1.5, 0)$, $(0, 3)$

$(1.5, 0)$ $w_0 + (1.5) w_1 + (0) w_2 = 0$

$w_0 + 1.5 w_1 = 0$

$w_0 = 1 \Rightarrow 1 + 1.5 w_1 = 0 \Rightarrow -0.67$

$(0, 3)$ $w_0 + (0) w_1 + 3 w_2 = 0$

$w_0 + 3 w_2 = 0$

$w_0 = 1 \Rightarrow 1 + 3 w_2 = 0 \Rightarrow -0.33$

$\boxed{w_0 = 1, w_1 = -0.67, w_2 = -0.33}$

Problem 2 continued

d) for the new data samples $(2, -1), (-0.5, 0.5)$, calculate and report the local field (v), perceptron output (y), and classification label.

$$v = w_0 x_0 + w_1 x_1 + w_2 x_2$$

$$= x_0 - 0.67 x_1 - 0.33 x_2$$

$$= x_0 - 0.67(2) - 0.33(-0.5)$$

$$= 1 - 0.67(2) - 0.33(-0.5)$$

$$v = -0.175$$

$$y = \begin{cases} 1 & \text{if } v \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$

$y =$ class 0

Problem 3: MNIST dataset - The MNIST dataset is divided into two sets - training and test. Each set comprises a series of images (28 x 28-pixel images of handwritten digits) and their respective labels (values from 0 - 9, representing which digit the image corresponds to).

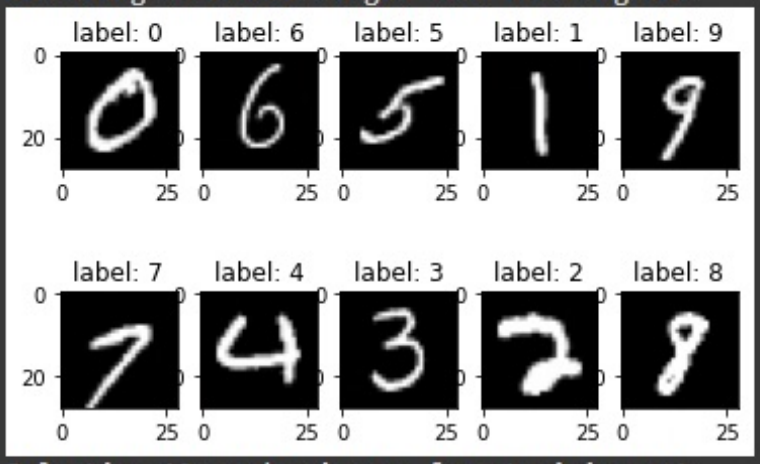
- a) Use mnist function in keras.datasets to split the MNIST dataset into the training and testing sets. Print the following: The number of images in each training and testing set, and the image width and height.
- b) Write a function (with images of ten digits and labels as the input) that plots a figure with 10 subplots for each 0-9 digits. Each subplot has the number of the handwritten digit in the title.
- c) Create a loop to call the plot function in (b) with images from each set to create three figures. Note: the code has to select the images randomly. Include all the 10 digits in each figure. Show the results of your code.
- d) In machine learning, we usually divide the training set into two sets of training and validation sets to adjust a machine learning model parameters. In your code, randomly select 20% of the training images and their corresponding labels and name them as x_valid and y_valid, respectively. Name the remaining training images and their labels as x_train and y_train, respectively. Print the number of images in each training and validation set. Note: that there are no overlaps between the two sets.

```
[ ] from keras.datasets import mnist
    import matplotlib.pyplot as plt
    import numpy as np

    # PART A

    # Split the MNIST dataset into training and testing sets
    (x_train, y_train), (x_test, y_test) = mnist.load_data()

    # Print out the training set
    print("There are", x_train.shape[0], "Images in the MNIST training set where each has a width of:", x_train.shape[1], "and a height of:", x_train.shape[2])

    # Print out the testing set
    print("There are", x_test.shape[0], "Images in the MNIST testing set where each has a width of:", x_test.shape[1],"and a height of:", x_test.shape[2])
```

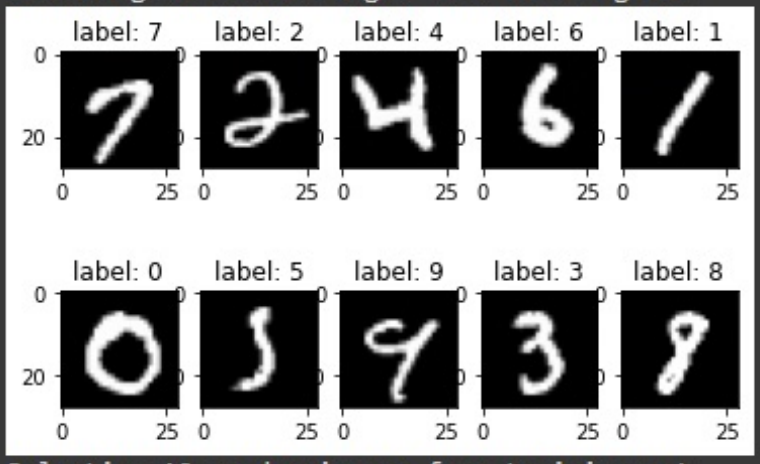
```
There are 60000 Images in the MNIST training set where each has a width of: 28 and a height of: 28
There are 10000 Images in the MNIST testing set where each has a width of: 28 and a height of: 28
```

```
[ ] # PART B


    def plot_figure(images, labels, digit_order):
        digits = digit_order

        #x_train_i = x_train_from_zero_to_nine[10,:,:]

        fig, axs = plt.subplots(nrows=2, ncols=5)

        axs = axs.ravel()

        for i in range(0, 10):
          x_train_from_zero_to_nine = x_train[y_train==int(digits[i]),:,:]
          axs[i].imshow(x_train_from_zero_to_nine[i], cmap='gray')
          axs[i].set_title('label: ' + str(digits[i]))

        plt.show()


    # PART C
    for i in range(0, 3):
      print("Selecting 10 random images from training set")
      digits = np.arange(0, 10)
      np.random.shuffle(digits)
      plot_figure(x_train, y_train, digits)
```

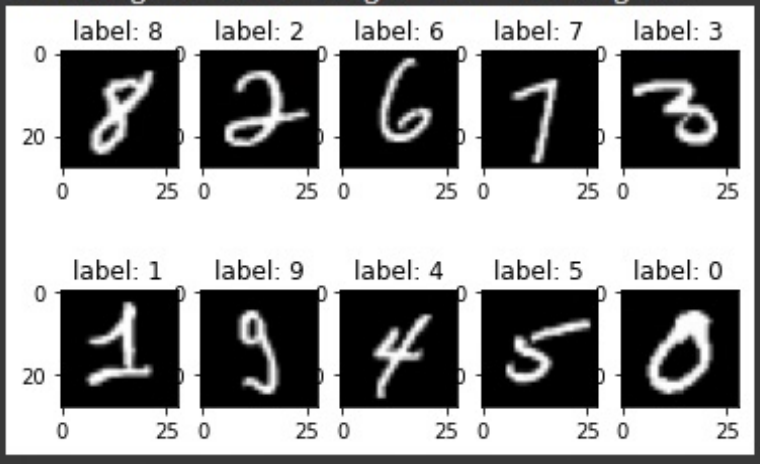
Selecting 10 random images from training set



Selecting 10 random images from training set



Selecting 10 random images from training set



```
[ ]  # PART D
     num_train_img = x_train.shape[0]
     train_ind=np.arange(0, num_train_img)
     train_ind_S = np.random.permutation(train_ind)

     x_train = x_train[train_ind_S,:,:]
     y_train = y_train[train_ind_S]


     # Selecting 20% of the training data
     x_valid = x_train[0:int(0.2*num_train_img),:,:]
     y_valid = y_train[0:int(0.2*num_train_img)]

     print("x_valid:", x_valid.shape[0])
     print("y_valid:", y_valid.shape[0])

     print()

     # The rest of the training set
     x_train = x_train[int(0.2*num_train_img):,:,:]
     y_train = y_train[int(0.2*num_train_img):]

     print("x_train:", x_train.shape[0])
     print("y_train:", y_train.shape[0])
```

```
x_valid: 12000
y_valid: 12000

x_train: 48000
y_train: 48000
```