

## Question 2: Feature Maps

- (a) This data is not separable by a linear classifier. We know that a linear classifier separating this data must partition  $\mathbb{R}^2$  into half-spaces  $H_1$  (positive) and  $H_0$  (negative) based on a threshold  $t$ .

In this dataset,  $p_1 = (-2, -1)$  and  $p_2 = (2, 3)$  have label 1, and so are in  $H_1$ . We know that half-spaces are convex, and so any convex combination of  $p_1$  and  $p_2$  must lie in  $H_1$ . If we take  $\lambda = 3/4$ , then

$$p_3 = p_1 + \lambda(p_2 - p_1) = (-2, -1) + \frac{3}{4}(2 - (-2), 3 - (-1)) = (-2, -1) + \frac{3}{4}(4, 4) = (1, 2)$$

must lie in  $H_1$  since it is convex combination of  $p_1$  and  $p_2$ . However, we know that  $p_3$  has label 0, and so must lie in  $H_0$ . This is a contradiction, and so no linear classifier can separate this data.

- (b) We will assume that a threshold  $t = 0$  is used. The constraints are as follows:

$$-2w_1 + w_2 \geq 0 \quad (1)$$

$$w_1 + 4w_2 < 0 \quad (2)$$

$$2w_1 + 9w_2 \geq 0 \quad (3)$$

- (c) The feasible area is given by the intersection of the blue, red, and green half-spaces below, which is the shaded region to the left of the red line, above the green line, and below the blue line. The feasible area is shown in Figure 1.

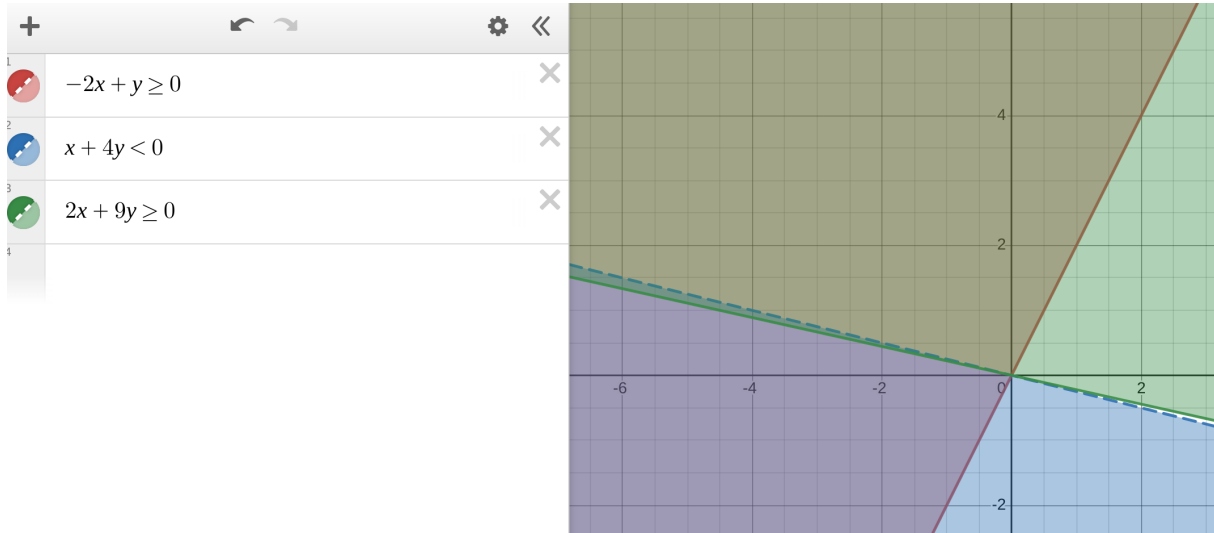


Figure 1: Feasible area for  $w_1, w_2$

## Question 3: kNN vs Logistic Regression

3.1 (a) The plot of classification vs validation accuracy is given below in Figure 2.

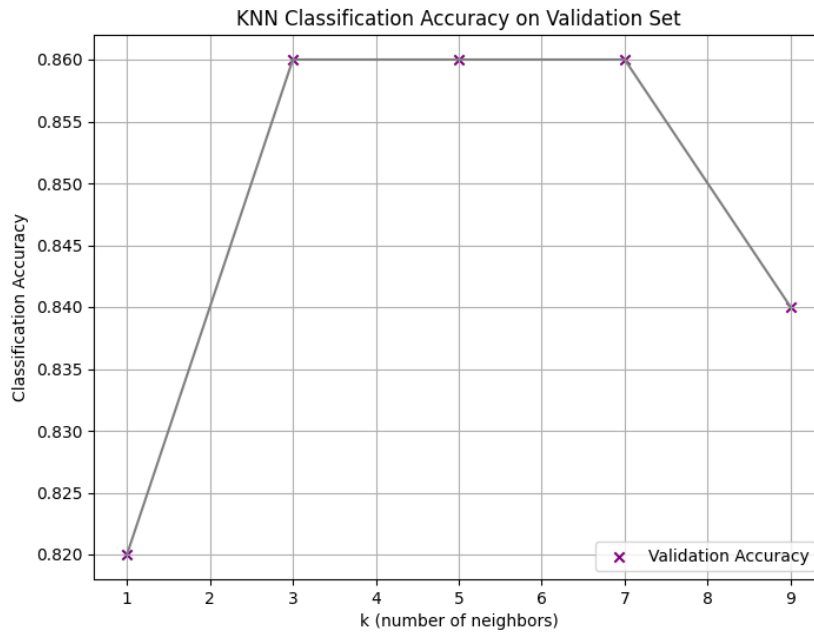


Figure 2: Classification vs Validation Accuracy

- (b) The value of  $k^*$  chosen is  $k^* = 3$ . This is because 3 is the smallest value of the hyperparameter  $k$  which achieves the maximum accuracy seen across all values of  $k$  (0.860). I have chosen this value of  $k$  in adherence with the principle of Occam's Razor, which states that the simplest solution is often the best one. In this case, the simplest solution is the least complex model (lowest  $k$ ), as the model complexity increases with  $k$ . Here is a report of the validation and test accuracies for  $k^*, k^* + 2, k^* - 2$ :

```
Classification accuracies for k_star=3
  Validation accuracy: 0.86
  Test accuracy: 0.92
Classification accuracies for (k_star + 2)=5
  Validation accuracy: 0.86
  Test accuracy: 0.94
Classification accuracies for (k_star - 2)=1
  Validation accuracy: 0.82
  Test accuracy: 0.88
```

We see that the maximum validation accuracy is reached when  $k^* = 3$ , increasing from 0.82 when  $k = 1$  and staying at the same value of 0.86 when  $k = 5$ .

However, the test accuracy increases from 0.88 to 0.92 when going from  $k^* - 2$  to  $k^*$ , and continues to increase with  $k = k^* + 2 = 5$ . This illustrates that the simplest model that performs best on the validation set may not always be the one that performs best on the test set.

3.2 (a) Done!

- (b) I've completed the missing parts in `run_logistic_regression`, and run the code on both `mnist_train` and `mnist_train_small`. The value returned by `run_check_grad` is small, and is given below:

```
diff = 2.246931370781212e-08
```

I used an initial value of  $w = \vec{0}$ , and a learning rate of  $\eta = 0.2$ . I chose this value as it was the largest value of  $\eta$  that did not cause the loss to fluctuate in the early training iterations on both the training and validation sets.

I used 500 iterations, as it appears that both the cross entropy and classification accuracies converge on the training and validation sets after 500 iterations.

The cross entropy and classification accuracies for the training and validation sets for  $\eta = 0.2$  and 500 iterations are given below:

```
Classification accuracy on training set: 1.0
Classification accuracy on validation set: 0.88
Classification accuracy on test set: 0.92
Cross entropy on training set: 0.02149375612870086
Cross entropy on validation set: 0.19186085743772488
Cross entropy on test set: 0.2142534589131518
```

- (c) Here are figures showing how cross-entropy changes as training progresses, one for the `mnist_train` and one for the `mnist_train_small` datasets.

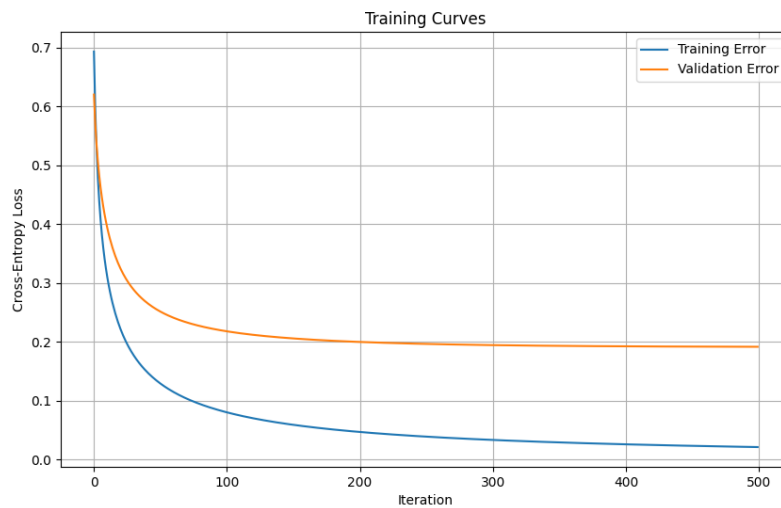


Figure 3: Cross entropy vs Training Iterations for `mnist_train`

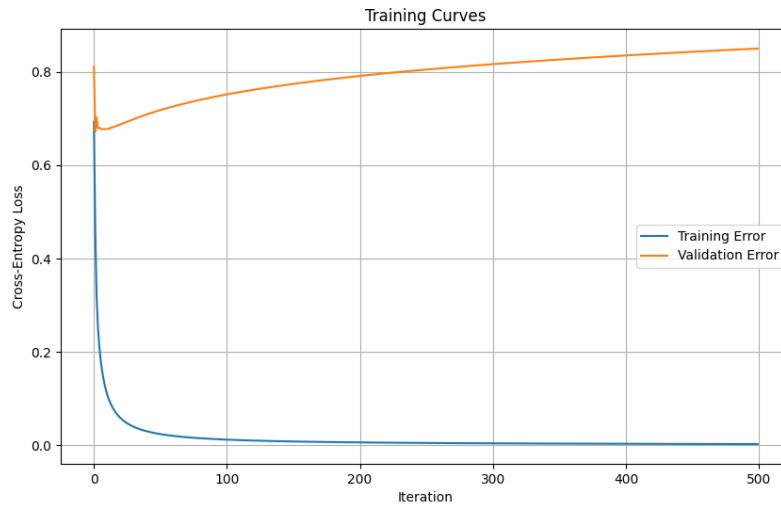


Figure 4: Cross entropy vs Training Iterations for `mnist_train_small`

We see that the validation error increases as the model trains on `mnist_train_small`, indicating that overfitting is taken place. No such phenomena is observed for `mnist_train`, indicating that the model is not overfitting on this dataset.

The results change with varying hyperparameters. Some ways to choose the best parame