CSCI 5922 problem set 1

Jake Krol

January 2025

1 Q1

1.1 Q1a

a1) Deep learning (DL) is different from machine learning (ML) since ML usually requires hand-crafted, task-specific input features, while DL is capable of learning features from statistical patterns of less-curated input data. Also, DL is a subset of ML. In DL, multi-layered neural networks are used to produce output by composing many functions over the input. ML does not always use multi-layered neural networks.

a2) Artificial intelligence (AI) and DL differ in their approaches to automate complex tasks. AI is the broad concept of machines performing complex tasks which includes deterministic programs with no explicit learning component. Meanwhile, DL is a subset of AI programs which are designed to learn how to perform a complex task when provided with example data.

1.2 Q1b

1.2.1 Q1bi

bi) An example AI algorithm which is not an ML algorithm is a deterministic program which computes the position of a stellar object using fixed instructions and parameters. In this example of computing the position of a stellar object, the program maps input to output with no updating to model parameters or program rules.

1.2.2 Q1bii

bii) Logistic regression is an ML algorithm, not a DL algorithm, which learns to classify inputs when provided labeled training data. Logistic regression is not a DL algorithm because the model is not a neural network with multiple hidden layers.

2 Q2

2.1 Q2a

Deep learning datasets are split into train and test to evaluate how well the trained model generalizes to new data, unseen during training. Often, DL models need to effectively perform the objective on input data which is similar, yet not explicitly present in the training dataset.

2.2 Q2b

No, there should be no overlap between train and test datasets. If data overlaps between train and test, then model performance on test data is less impactful. The lowered impact is because the evaluation does not quantify how the DL model can generalize to new data.

2.3 Q2c

Classification and regression differ in the range of their output values. Classification outputs discrete categorical values while regression outputs a continuous value.

3 Q3

3.1 Q3a

The perceptron model parameter update equation is

$$w'_j = w_j + \Delta w$$

where

$$\Delta w = \eta(y_i - \hat{y}_i)x_{ij}$$

. The bias, b, is similar updated similarly, except x_{ij} is fixed to 1.

$$b' = b + \eta(y_i - \hat{y}_i) \cdot 1$$

The variables are

- w_j := weight of jth feature
- w'_j := updated weight for jth feature
- Δw := change of weight for jth feature, based on learning algorithm
- η := learning rate
- y_i := target/label for ith training example
- $\hat{y}_i$:= output of perceptron for ith training example
- x_{ij} := value of jth feature for ith training example

Given the training data and the previously defined parameter update equation, the initial weights $\mathbf{w}^T = [1, 1]$ and bias $b = -1$ can be updated after each training example.

Training sample	**Updated** w_1	**Updated** w_2	**Updated** b
1	0.5	0.5	-1.5
2	0.0	1.0	-1.0
3	-0.5	0.5	-0.5
4	-0.5	0.5	-0.5

Table 1: **Weight and bias updates after each training sample**

Output and weight updates are computed by

$$\begin{aligned} \hat{y} &= sign(w_1 x_1 + w_2 x_2 + b) \\ \Delta w &= \eta(y_i - \hat{y}_i) x_{ij} \\ w_j' &= w_j + \Delta w \end{aligned}$$

Initial parameters are $\mathbf{w}^T = [1, 1]$ and $b = -1$.

- Training sample 1

$$\begin{aligned} \hat{y} &= sign((1 \cdot 1) + (1 \cdot 1) - 1)) = 1 \\ w_1' &= 1 + \frac{1}{4}(-1 - 1) \cdot 1 = 1 - \frac{1}{2} = \frac{1}{2} \\ w_2' &= 1 + \frac{1}{4}(-1 - 1) \cdot 1 = 1 - \frac{1}{2} = \frac{1}{2} \\ b &= -1 + \frac{1}{4}(-1 - 1) \cdot 1 = -1 - \frac{1}{2} = -\frac{3}{2} \\ &\rightarrow \mathbf{w}^T = [\frac{1}{2}, \frac{1}{2}], b = -\frac{3}{2} \end{aligned}$$

- Training sample 2

$$\begin{aligned} \hat{y} &= sign((\frac{1}{2} \cdot -1) + (\frac{1}{2} \cdot 1) - \frac{3}{2}) = -1 \\ w_1' &= \frac{1}{2} + \frac{1}{4}(1 - (-1)) \cdot -1 = \frac{1}{2} - \frac{1}{2} = 0 \\ w_2' &= \frac{1}{2} + \frac{1}{4}(1 - (-1)) \cdot 1 = \frac{1}{2} + \frac{1}{2} = 1 \\ b' &= -\frac{3}{2} + \frac{1}{4}(1 - (-1)) \cdot 1 = -\frac{3}{2} + \frac{1}{2} = -1 \\ &\rightarrow \mathbf{w}^T = [0, 1], b = -1 \end{aligned}$$

- Training sample 3

$$\hat{y} = sign((0 \cdot -1) + (1 \cdot -1) - 1) = -1$$
$$w_1' = 0 + \frac{1}{4}(1 - (-1)) \cdot -1 = 0 + -\frac{1}{2} = -\frac{1}{2}$$
$$w_2' = 1 + \frac{1}{4}(1 - (-1)) \cdot -1 = 1 - \frac{1}{2} = \frac{1}{2}$$
$$b' = -1 + \frac{1}{4}(1 - (-1)) \cdot 1 = -1 + \frac{1}{2} = -\frac{1}{2}$$
$$\rightarrow \mathbf{w}^T = [-\frac{1}{2}, \frac{1}{2}], b = -\frac{1}{2}$$

- Training sample 4

$$\hat{y} = sign((-\frac{1}{2} \cdot 1) + (\frac{1}{2} \cdot -1) - \frac{1}{2}) = -1$$

No parameter updates since $target - output = -1 - (-1) = 0$.
Final parameters are $\mathbf{w}^T = [-\frac{1}{2}, \frac{1}{2}], b = -\frac{1}{2}$.

3.2 Q3b

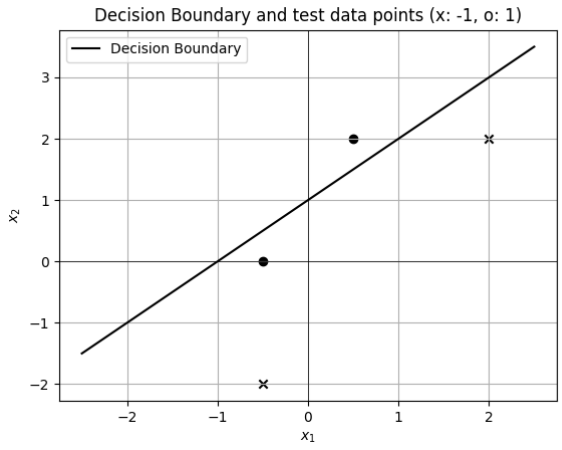

Figure 1: **Test data and trained model decision boundary**. "x"s are test samples with $target = -1$, and "•"s are test samples with $target = 1$. The black, solid line is the decision boundary after training.

Model predictions on the test data are

- Test sample 1

$$\hat{y}_1 = sign((\frac{1}{2} \cdot -\frac{1}{2}) + (\frac{1}{2} \cdot 2) - \frac{1}{2}) = 1$$

- Test sample 2

$$\hat{y}_2 = sign((-\frac{1}{2} \cdot 2) + (\frac{1}{2} \cdot 2) - \frac{1}{2}) = -1$$

- Test sample 3

$$\hat{y}_3 = sign((-\frac{1}{2} \cdot -\frac{1}{2}) + (\frac{1}{2} \cdot 0) - \frac{1}{2}) = -1$$

- Test sample 4

$$\hat{y}_4 = sign((-\frac{1}{2} \cdot -\frac{1}{2}) + (\frac{1}{2} \cdot -2) - \frac{1}{2}) = -1 \quad (1)$$

3.3 Q3c

	$y = 1$	$y = -1$
$\hat{y} = 1$	1	0
$\hat{y} = -1$	1	2

Table 2: **Confusion matrix of results on test data.**

- $Accuracy_{test} = \frac{3}{4}$
- $Precision_{test} = \frac{TP}{TP+FP} = \frac{1}{1+0} = 1$
- $Recall_{test} = \frac{TP}{TP+FN} = \frac{1}{1+1} = \frac{1}{2}$

3.4 Q3d

- The **confusion matrix** shows the counts of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). Overall, the model performed decently: $TP = 1, FP = 0, FN = 1, TN = 2$.
- The model correctly predicted $\frac{3}{4}$ test samples for an **accuracy** of 0.75. Since there are an equal amount of positive and negative test samples, a 0.75 accuracy is better than a random classifier, which would have 0.5 accuracy.
- The $\mathbf{precision_{test}} = 1$ shows that whenever the model predicted a sample was positive ($\hat{y} = 1$) it was always correct.
- The $\mathbf{recall_{test}} = \frac{1}{2}$ shows that the model did not correctly predict all test samples with a ground truth positive class.

3.5 Q3e

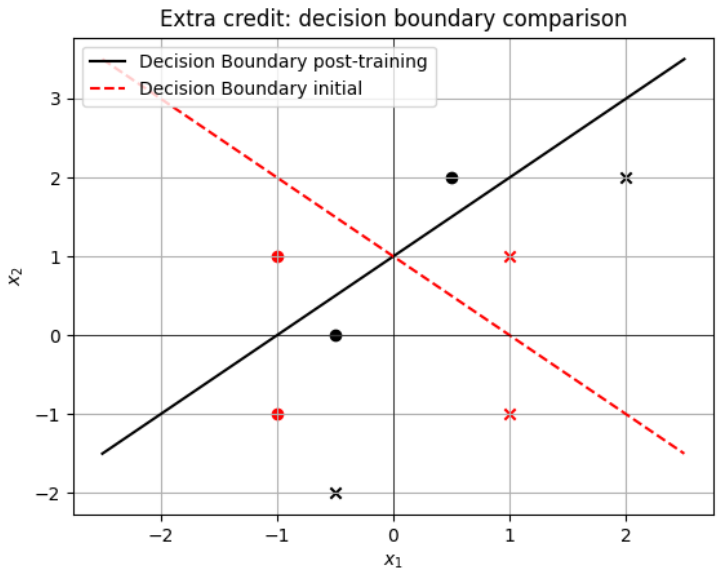

Figure 2: **Decision boundary comparison.** All train data points (red) and test data points (black) are plotted in the 2-d feature space. Decision boundaries are shown for the model before training (red, dashed) and after training (black, solid). "x"s are samples with $target = -1$, and "•"s are samples with $target = 1$.

An optimal decision boundary that separates all -1 and 1 labeled points is when $w_1 = 2$,$w_2 = 1$, and $b = 0$. These parameters would create a decision boundary line which crosses the origin and has a positive slope (with respect to x_1) of 2.

Yes, training moved the decision boundary closer to the optimal solution.