Exercise 1: Basic Perceptron Update (No Regularization)

You have a perceptron with initial weights $w = [0, 0]$ and bias $b = 0$. The dataset is:

x_1	x_2	y
3	2	1
−1	−3	−1
2	−2	−1
−3	3	1

Task:

1. Use the Perceptron update rule with **learning rate** $\eta = 1$ for **two epochs**.
2. Update w and b accordingly.
3. Show the final weights and bias.

Exercise 2: Perceptron with L2 Regularization

Now, apply **L2 Regularization** (weight decay) to the perceptron update rule.

Dataset:

x_1	x_2	y
2	1	1
−2	−3	−1
1	−2	−1
−1	2	1

Regularized Update Rule:

$$w \leftarrow (1 - \lambda)w + \eta y x$$

$$b \leftarrow b + \eta y$$

Task:

1. Perform two epochs of perceptron updates using $\lambda = 0.1$ and $\eta = 1$.
2. Record the changes in w and b at each step.
3. What effect does L2 regularization have on weight updates?

Exercise 3: Perceptron with L1 Regularization (Sparse Weights)

L1 regularization (Lasso) encourages **sparse** weight updates by penalizing large weights.

Dataset:

x_1	x_2	y
1	3	1
-2	-1	-1
2	-2	-1
-3	2	1

Regularized Update Rule:

$$w_j \leftarrow w_j + \eta \left(y x_j - \lambda \text{sign}(w_j) \right)$$

$$b \leftarrow b + \eta y$$

Task:

1. Perform perceptron updates with $\lambda = 0.2$ and $\eta = 1$ for **two epochs**.
2. Track the evolution of w and b.
3. Compare the results with L2 regularization. What do you notice?

Exercise 4: Perceptron with Margin-based Regularization

Instead of just misclassified points, we only update the weights when the prediction is **within a margin** $y(w^T x + b) < 1$.

Dataset:

x_1	x_2	y
1	2	1
-1	-1	-1
2	-3	-1
-2	1	1

Regularized Update Rule (Hinge loss-inspired):
Only update **if** $y(w^T x + b) < 1$, otherwise do nothing:

$$w \leftarrow w + \eta y x - \lambda w$$

$$b \leftarrow b + \eta y$$

Task:

1. Apply the perceptron learning rule with $\lambda = 0.1$ and $\eta = 1$ for **two epochs**.
2. How does this margin-based update differ from standard perceptron learning?

Solution 1: Basic Perceptron Update (No Regularization)

Dataset:

x_1	x_2	y
3	2	1
-1	-3	-1
2	-2	-1
-3	3	1

Initial Values:

- Weights: $w = [0, 0]$
- Bias: $b = 0$
- Learning rate: $\eta = 1$

Update Rule (Standard Perceptron)

$$w \leftarrow w + \eta y x$$

$$b \leftarrow b + \eta y$$

Epoch 1

Sample (x_1, x_2, y)	Prediction $\hat{y} = \text{sign}(w^T x + b)$	Update Applied?	New w	New b
(3,2,1)	$\text{sign}(0) = 0$ (incorrect)	Yes	(3,2)	1
(-1,-3,-1)	$\text{sign}(3(-1) + 2(-3) + 1) = -1$ (correct)	No	(3,2)	1
(2,-2,-1)	$\text{sign}(3(2) + 2(-2) + 1) = 1$ (incorrect)	Yes	(1,4)	0
(-3,3,1)	$\text{sign}(1(-3) + 4(3) + 0) = 9$ (correct)	No	(1,4)	0

Epoch 2

Sample (x_1, x_2, y)	Prediction $\hat{y}$	Update Applied?	New w	New b
(3,2,1)	$\text{sign}(1(3)+4(2)+0) = 11$ (correct)	No	(1,4)	0
(-1,-3,-1)	$\text{sign}(1(-1)+4(-3)+0) = -13$ (correct)	No	(1,4)	0
(2,-2,-1)	$\text{sign}(1(2)+4(-2)+0) = -6$ (correct)	No	(1,4)	0
(-3,3,1)	$\text{sign}(1(-3)+4(3)+0) = 9$ (correct)	No	(1,4)	0

Final Weights: $w = (1, 4)$, **Final Bias:** $b = 0$

Solution 2: Perceptron with L2 Regularization (Weight Decay)

We apply the Perceptron update rule with **L2 regularization** using $\lambda = 0.1$ for two epochs.

Dataset:

x_1	x_2	y
2	1	1
-2	-3	-1
1	-2	-1
-1	2	1

Initialization:

- **Weights:** $w = [0, 0]$
- **Bias:** $b = 0$
- **Learning rate:** $\eta = 1$
- **Regularization parameter:** $\lambda = 0.1$

Update Rule (L2 Regularization)

The standard perceptron update is:

$$w \leftarrow w + \eta y x$$

$$b \leftarrow b + \eta y$$

With **L2 Regularization (Weight Decay)**, we modify the weight update as follows:

$$w \leftarrow (1 - \lambda) w + \eta y x$$

$$b \leftarrow b + \eta y$$

Where:

- $(1 - \lambda) w$ shrinks the weights to prevent large values.

Epoch 1 (Pass through the dataset)

Step 1: First Sample (2,1), y = 1

- Prediction:

  $$\hat{y} = \text{sign}(0) = 0$$

  (Incorrect, so update)

- Weight Update:

  $$w \leftarrow (1 - 0.1)(0, 0) + 1(1)(2, 1) = (2, 1)$$

- Bias Update:

  $$b \leftarrow 0 + 1(1) = 1$$

Step 2: Second Sample (-2,-3), y = -1

- Prediction:

  $$\hat{y} = \text{sign}(2(-2) + 1(-3) + 1) = \text{sign}(-6) = -1$$

  (Correct, no update)

Step 3: Third Sample (1,-2), y = -1

- Prediction:

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

(Incorrect, update)

- Weight Update:

$$w \leftarrow (1 - 0.1)(2, 1) + 1(-1)(1, -2) = (0.8, 2.1)$$

- Bias Update:

$$b \leftarrow 1 - 1 = 0$$

Step 4: Fourth Sample (-1,2), y = 1

- Prediction:

$$\hat{y} = \text{sign}(0.8(-1) + 2.1(2) + 0) = \text{sign}(3.4) = 1$$

(Correct, no update)

Epoch 2 (Second Pass Through the Dataset)

Step 1: First Sample (2,1), y = 1

- Prediction:

$$\hat{y} = \text{sign}(0.8(2) + 2.1(1) + 0) = \text{sign}(3.7) = 1$$

(Correct, no update)

Step 2: Second Sample (-2,-3), y = -1

- Prediction:

$$\hat{y} = \text{sign}(0.8(-2) + 2.1(-3) + 0) = \text{sign}(-7.9) = -1$$

(Correct, no update)

Step 3: Third Sample (1,-2), y = -1

- Prediction:

$$\hat{y} = \text{sign}(0.8(1) + 2.1(-2) + 0) = \text{sign}(-3.4) = -1$$

(Correct, no update)

Step 4: Fourth Sample (-1,2), y = 1

- Prediction:

$$\hat{y} = \text{sign}(0.8(-1) + 2.1(2) + 0) = \text{sign}(3.4) = 1$$

(Correct, no update)

Final Weights and Bias After 2 Epochs:

- $w = (0.8, 2.1)$
- $b = 0$

- **L2 regularization prevents large weight updates** by multiplying by $(1 - \lambda)$ at each step.
- The final weights are **smaller compared to the standard perceptron**, preventing overfitting.
- The updates became stable after **one epoch**.