

Assignment: Build a Single Layer Perceptron with 3 Inputs (From Scratch)

Objective

Build a Single Layer Perceptron using only Python basics to predict whether a person is **Fit (1)** or **Unfit (0)** based on lifestyle indicators designed by you.

This assignment focuses on:

- understanding features
 - designing rules
 - implementing perceptron learning manually
-

Problem Statement

Create a system that predicts whether a person is **Fit (1)** or **Unfit (0)** based on three inputs:

1. Hours of physical activity per week
2. Average sleep hours per day
3. Daily water intake (liters)

 You must design your own dataset and labeling rule.

Input Format

Each data point must follow:

[activity_hours, sleep_hours, water_intake]

Example (format only, do not copy):

[3, 6, 1.5]

Output

1 → Fit

0 → Unfit

Students must clearly define:

- what conditions qualify as “Fit”

- what conditions qualify as “Unfit”

This rule must be explained in comments.

☐ Mandatory Rules

☒ Not Allowed

- numpy
- sklearn
- tensorflow / keras / pytorch
- Copying code from internet or classmates

☒ Must Use

- for loops
 - if-else conditions
 - lists
 - functions
 - user input (input())
 - clear printed output
-

 Task Breakdown

Task **1**: Dataset Creation

- Create 12–18 data points
- Store features in a list of lists
- Store labels in a separate list
- Clearly explain your labeling rule in comments

Students should try to create a balanced dataset.

Task **2**: Perceptron Initialization

Manually define:

- Three weights
- One bias
- Learning rate

Students must explain:

- why the learning rate was chosen
 - whether weights were random or fixed
-

Task 3: Activation Function

Write a function that:

1. Calculates weighted sum
2. Applies a step function
3. Returns 0 or 1

Students must choose and explain the threshold value.

Task 4: Training Loop

Train for multiple epochs:

For each data point:

- Predict output
- Compute error
- Update weights and bias

✦ Students must write and explain the update rule in comments.

Task 5: Monitoring Learning

After each epoch print:

- Epoch number
- Total error
- Current weights

This helps verify learning.

Task 6: User Input Testing

After training:

- Take user input for:
 - activity hours
 - sleep hours
 - water intake
- Predict Fit or Unfit
- Print a meaningful interpretation

Example:

“This person is likely to be Fit based on the given lifestyle pattern.”

Short Report (Required)

Students must submit a short explanation:

1. How the dataset was designed
2. What rule was used for labeling
3. How they checked that learning was happening
4. What happened to weights during training

⚠ Identical datasets or explanations will be treated as plagiarism.

Bonus (Optional)

Students may attempt:

1. Calculate accuracy
 2. Print error per epoch
 3. Show weight changes in a table
 4. Try different learning rates and compare results
-

★ Challenge Question (Bonus Marks)

If one feature is removed (for example water intake), does the model perform worse?
Explain why.
