AI INTERNSHIP REPORT

# **Task 5**

# Supervised & Unsupervised Learning Models

## Submitted by: Rija Mobin

## Submitted to: Miss Samia

## Submission Date: August 19, 2025

**Task Requirements**

- Study and implement Supervised & Unsupervised Learning Models

- Implement each model from scratch and using libraries

- Provide detailed documentation explaining each model’s theory, code, and real-life examples

- Demonstrate with visualizations

**Objectives**

- Understand the fundamental differences between Supervised and Unsupervised learning.

- Implement Linear Regression, Logistic Regression, and KMeans Clustering from scratch.

- Verify results using scikit-learn library implementations.

- Visualize results for better comprehension.

**Detailed Theory of Supervised & Unsupervised Learning**

🔹 **What is Machine Learning?**

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that enables computers to learn patterns from data and make predictions or decisions without explicit programming. ML can be used in applications like recommendation systems, medical diagnosis, and stock price prediction.

**Two main paradigms:**

**1.** Supervised Learning

**2.** Unsupervised Learning

**🔹 Supervised Learning**

**Definition:** Uses labeled data where each input X has a known output Y. The model learns a mapping function f(X) → Y.

**Key Characteristics:**

- Requires labeled training data.

- Performance measured using metrics like Mean Squared Error (MSE) for regression, Accuracy for classification.

- Common tasks: Regression (predict continuous values) and Classification (predict categorical values).

**Linear Regression (Regression)**

**- Purpose:** Predict continuous numerical values (e.g., house price, salary, temperature).

**- Formula:** Y = mX + b, where m = slope, b = intercept.

**- Goal:** Find m and b that minimize Mean Squared Error (MSE):

MSE = (1/n) \* Σ (y\_i - y\_pred\_i)^2

- **Real-life Example:** Predicting house price based on size.

**- Code Explanation (Scratch):**

1. Initialize m and b to 0.

2. Use Gradient Descent to update m and b iteratively.

3. Compute predicted values using current m and b.

4. Calculate gradients for slope and intercept.

5. Update m and b to minimize MSE.

**- Library Code:** LinearRegression from sklearn performs the same optimization internally.

**✅ Logistic Regression (Classification)**

**- Purpose:** Predict probability of a binary outcome (e.g., pass/fail, spam/not spam).

**- Sigmoid function:** σ(z) = 1 / (1 + e^-z), where z = wX + b.

**- Decision rule:** If σ(z) > 0.5 → Class 1; else → Class 0.

**- Loss Function:** Binary Cross-Entropy (Log Loss):

L = -(1/n) Σ [y\_i \* log(y\_pred\_i) + (1 - y\_i) \* log(1 - y\_pred\_i)]

**- Real-life Example:** Predicting whether a student passes based on study hours.

**- Code Explanation (Scratch):**

1. Initialize weights w and bias b.

2. Compute predictions using sigmoid function.

3. Calculate gradient of loss with respect to w and b.

4. Update w and b using gradient descent.

**- Library Code:** LogisticRegression from sklearn performs this optimization efficiently.

**🔹 Unsupervised Learning**

**Definition:** Works on unlabeled data, discovers hidden patterns, clusters, or structures.

**- Common tasks:** Clustering, Dimensionality Reduction.

**KMeans Clustering (Clustering)**

**- Purpose:** Partition data into K clusters based on distance from cluster centroids.

**- Algorithm Steps:**

**1.** Choose number of clusters k.

**2.** Initialize k centroids randomly.

**3.** Assign each data point to the nearest centroid.

**4.** Update centroid positions to the mean of assigned points.

**5.** Repeat steps 3-4 until centroids stabilize.

**- Objective:** Minimize Within-Cluster Sum of Squares (WCSS):

WCSS = Σ\_i=1^k Σ\_x∈C\_i ||x - μ\_i||^2

**- Real-life Example:** Grouping customers by purchasing habits.

**- Code Explanation (Scratch):**

**1.** Randomly initialize centroids.

**2.** Assign points based on Euclidean distance.

**3.** Recalculate centroids as mean of points in each cluster.

**4.** Repeat until convergence.

**- Library Code:** KMeans from sklearn handles initialization, assignment, and convergence automatically.

📌 **Key Features of Supervised Learning**

**- Data:** Labeled (input + output)

**- Goal:** Learn mapping X → Y

**- Examples:** Linear Regression, Logistic Regression

**- Output:** Predicted labels or continuous values

**📌 Key Features of Unsupervised Learning**

**- Data:** Unlabeled (only input)

**- Goal:** Discover hidden patterns, clusters, or structures

**- Examples:** KMeans, PCA

**- Output:** Clusters, groups, or patterns

**📌 Model Explanations & Code**

**✅ Linear Regression (Supervised)**

import numpy as np  
X = np.array([1,2,3,4,5])  
y = np.array([2,4,6,8,10])  
m, b = 0, 0  
learning\_rate = 0.01  
for \_ in range(1000):  
 y\_pred = m\*X + b  
 dm = -(2/len(X))\*sum(X\*(y - y\_pred))  
 db = -(2/len(X))\*sum(y - y\_pred)  
 m -= learning\_rate\*dm  
 b -= learning\_rate\*db  
  
from sklearn.linear\_model import LinearRegression  
model = LinearRegression()  
model.fit(X.reshape(-1,1), y)

**Visualization:** Scatter plot with regression line.

**✅ Logistic Regression (Supervised)**

def sigmoid(z):  
 return 1/(1+np.exp(-z))  
w, b = 0, 0  
for \_ in range(1000):  
 z = w\*X + b  
 y\_pred = sigmoid(z)  
 dw = (1/len(X))\*sum(X\*(y\_pred - y))  
 db = (1/len(X))\*sum(y\_pred - y)  
 w -= learning\_rate\*dw  
 b -= learning\_rate\*db  
  
from sklearn.linear\_model import LogisticRegression  
model = LogisticRegression()  
model.fit(X.reshape(-1,1), y)

**Visualization:** Scatter plot with decision boundary.

**✅ KMeans Clustering (Unsupervised)**

from sklearn.cluster import KMeans  
kmeans = KMeans(n\_clusters=3)  
kmeans.fit(X.reshape(-1,1))

**Visualization:** Scatter plot with clusters and centroids.

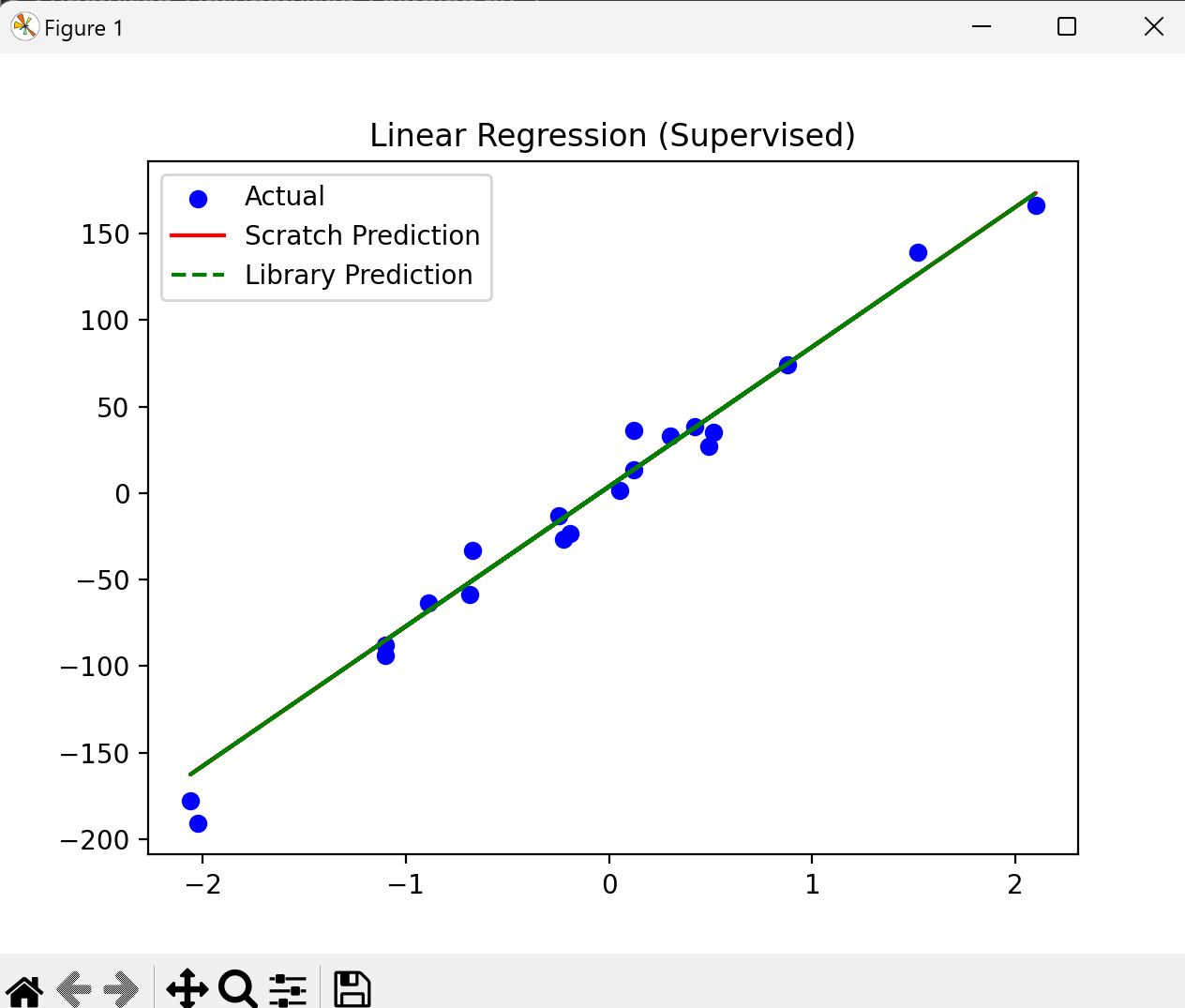
**📌 Summary of Models Implemented**

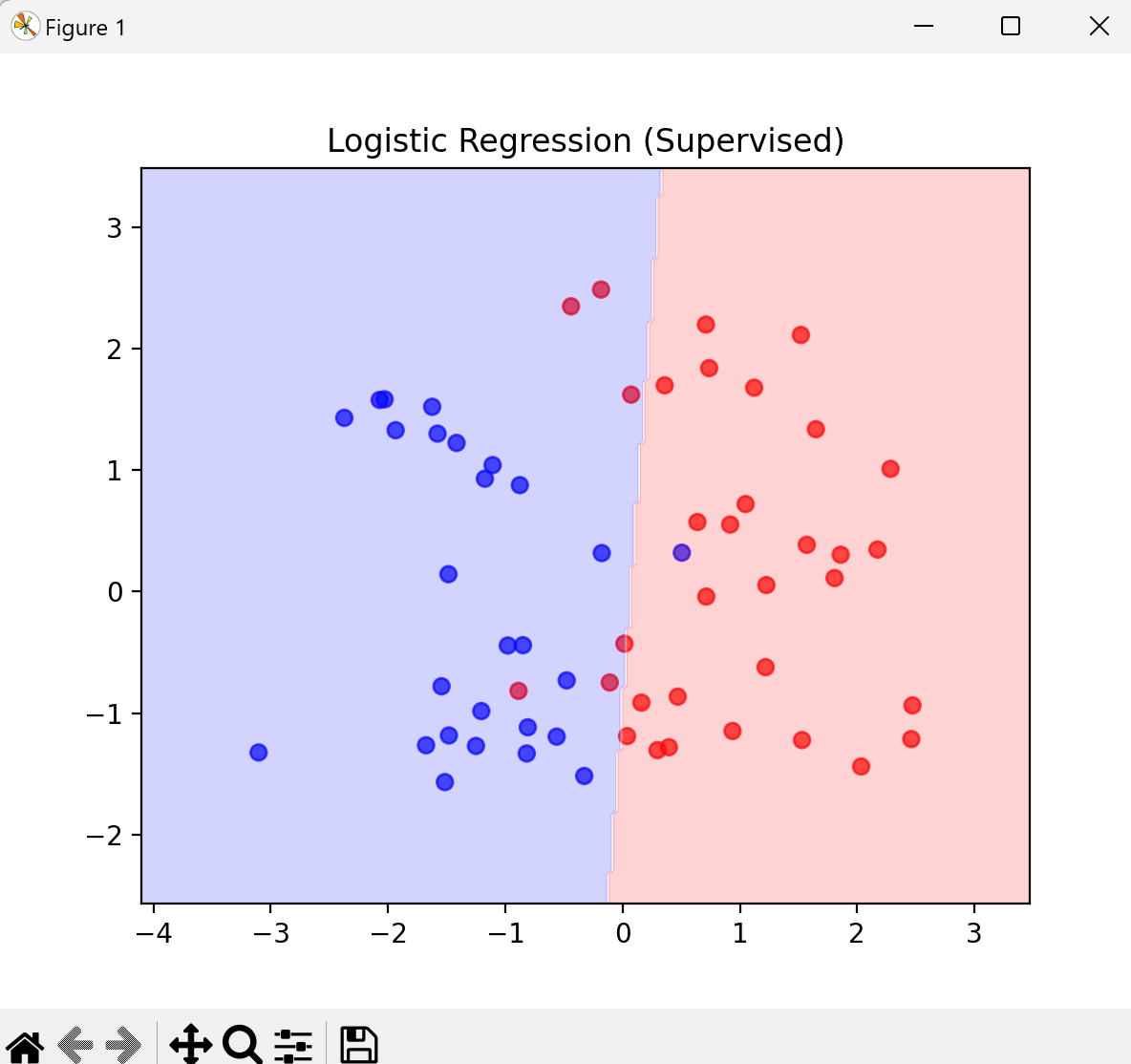
**- Linear Regression:** Supervised, Regression, predicts numbers, Scratch and Library implementations, Visualized with regression line.

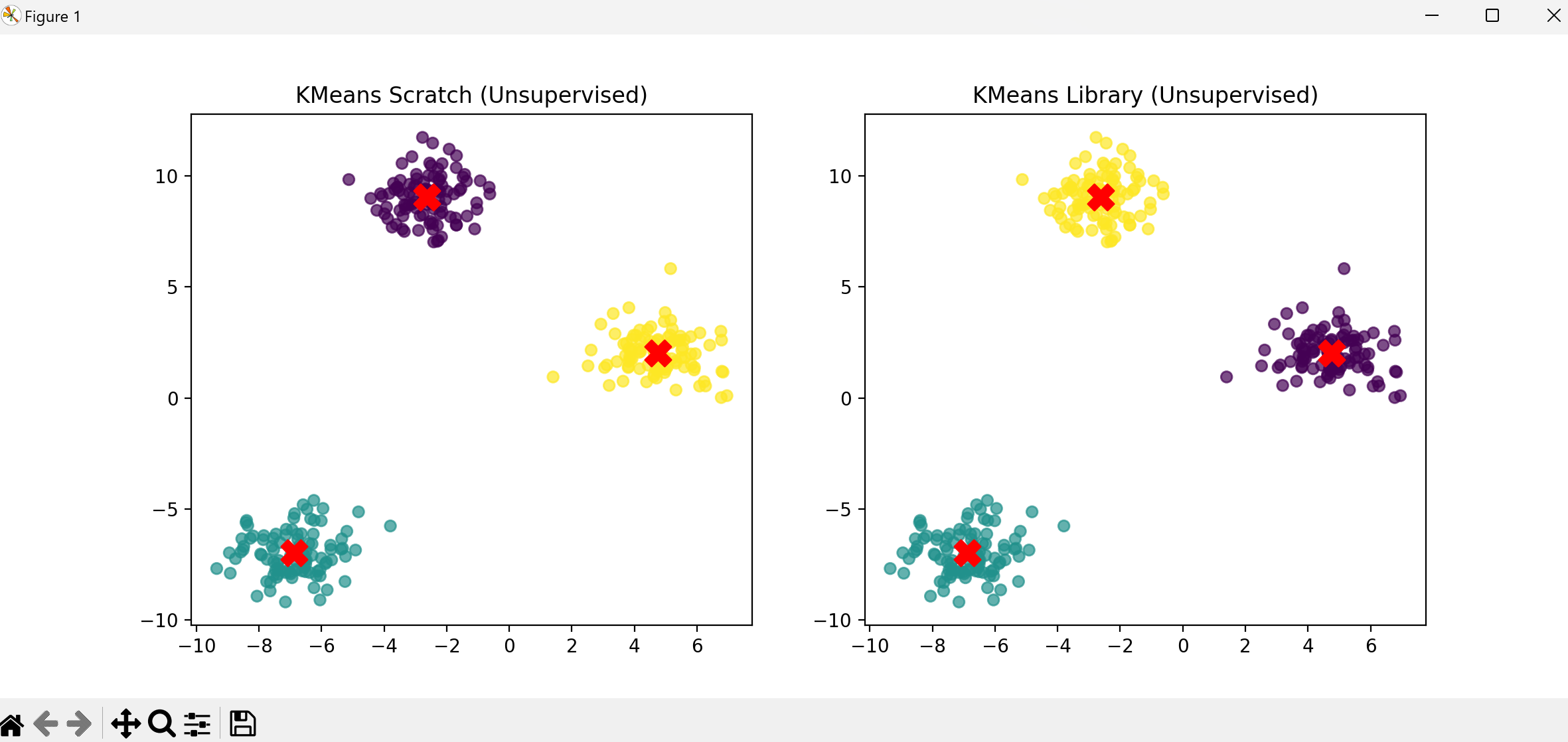
**- Logistic Regression:** Supervised, Classification, predicts binary classes, Scratch and Library implementations, Visualized with decision boundary.

**- KMeans Clustering:** Unsupervised, Clustering, groups data into clusters, Scratch and Library implementations, Visualized with clusters and centroids.

**Visualizations:**

****

****

****

**📌 Learning Outcomes**

- Clearly distinguished supervised vs unsupervised learning.

- Implemented ML models from scratch and understood underlying math.

- Practiced efficient implementation using scikit-learn.

- Learned to visualize data and model predictions.

📌 **Conclusion**

This task enhanced understanding of ML models mathematically and programmatically. Scratch implementations provided deep insight into gradients, error calculations, and optimization, while library implementations demonstrated practical application and efficiency.