Day 7 Training Report

1 July 2025

Introduction to Supervised Learning + Linear & Logistic Regression

On **Day 7**, the focus was on understanding the **core concepts of supervised learning**, including **linear regression** and **logistic regression**, which are two of the most fundamental algorithms in machine learning. This day introduced students to how labeled data is used to train models that can make predictions on new, unseen data.

---

1. Introduction to Supervised Learning

Supervised learning is a **machine learning paradigm** where the model is trained using **labeled data**, meaning each training example consists of an input and its corresponding correct output.

- **Input (X):** Features or predictors
- **Output (Y):** Target labels or values

The goal is for the model to **learn a mapping function** f:X→Yf: X \rightarrow Yf:X→Y so that it can predict YYY for new inputs XXX.

Types of Supervised Learning:

- □ **Regression** → Predicting continuous values (e.g., house price, temperature).
- □ **Classification** → Predicting categorical values (e.g., spam/ham, disease/no disease).

Examples:

- Predicting stock prices (regression)
- Classifying emails as spam or not spam (classification)
- Predicting student grades based on study hours (regression)

---

2. Linear Regression

Linear regression is used when the **target variable is continuous**. It fits a **straight line** through the data that minimizes the error between predicted and actual values.

Equation:

y=mx+cy = mx + cy=mx+c

Where:

- yyy = predicted value
- mmm = slope (coefficient)
- xxx = input feature
- ccc = intercept

Implementation Example:

```
import pandas as pd
from sklearn.linear_model import LinearRegression

# Sample data
data = {'Hours': [1, 2, 3, 4, 5],
        'Marks': [50, 55, 65, 70, 75]}
df = pd.DataFrame(data)

X = df[['Hours']]
y = df['Marks']

# Model training
model = LinearRegression()
model.fit(X, y)

# Prediction
predicted = model.predict([[6]])
print("Predicted Marks for 6 hours:", predicted[0])
```

☑ **Key Points:**

- Best for numeric prediction tasks.
- Assumes a **linear relationship** between input and output.
- Easy to interpret coefficients.

---

3. Logistic Regression

Despite its name, **logistic regression is used for classification problems**, not regression. It predicts **probabilities** using the **sigmoid (logistic) function** and classifies outputs based on a threshold (commonly 0.5).

Sigmoid Function:

P(y=1|x)=11+e−(mx+c)P(y=1|x) = \frac{1}{1 + e^{-(mx+c)}}P(y=1|x)=1+e−(mx+c)1

Implementation Example:

```
import pandas as pd
from sklearn.linear_model import LogisticRegression

# Sample dataset
```

```
data = {'Hours_Studied': [1, 2, 3, 4, 5, 6],
        'Passed': [0, 0, 0, 1, 1, 1]}
df = pd.DataFrame(data)

X = df[['Hours_Studied']]
y = df['Passed']

# Model training
log_model = LogisticRegression()
log_model.fit(X, y)

# Prediction
print("Probability of passing for 3 hours:", log_model.predict_proba([[3]]))
print("Prediction (0=Fail, 1=Pass):", log_model.predict([[3]]))
```

✅ **Key Points:**

- Suitable for **binary classification** (e.g., yes/no, 0/1).
- Outputs probabilities and class predictions.
- Can be extended to **multiclass classification** using one-vs-rest strategy.

---

4. Comparison: Linear vs Logistic Regression

Feature	**Linear Regression**	**Logistic Regression**
Output Type	Continuous numeric value	Probability (0–1), then class (0/1)
Use Case	Regression problems	Classification problems
Linearity Assumption	Yes	Logistic transformation used
Example	Predict house price	Predict pass/fail

---

5. Hands-on Implementation & Exercises

Students implemented both regression types on small datasets to understand their behavior. Exercises included:

- Predicting **student marks** from study hours using linear regression.
- Classifying **emails as spam or not** using logistic regression.
- Visualizing the **regression line** and **decision boundary** to see how models work internally.

Visualization Example (Regression Line):

```
import matplotlib.pyplot as plt

plt.scatter(X, y, color='red')
plt.plot(X, model.predict(X), color='blue')
plt.title('Linear Regression - Hours vs Marks')
plt.xlabel('Hours')
plt.ylabel('Marks')
plt.show()
```