Logistic Regression

Logistic Regression is a **classification algorithm** used to predict **two categories (binary classification)** such as:

- Spam vs Not Spam
- Disease vs No Disease
- Customer Buy vs Not Buy
- Pass vs Fail

It predicts a **probability** between **0 and 1**, and then applies a **threshold (usually 0.5)** to decide the final output.

Linear Regression outputs any value (−∞ to +∞), but classification needs probability (0 to 1). So Logistic Regression uses a **Sigmoid Function** to convert output to probability.

Mathematical Formula

Sigmoid Function (Activation)

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

For Logistic Regression:

$$z = w_1 x_1 + w_2 x_2 + \ldots + b$$

Final prediction:

$$\hat{y} = \sigma(z)$$

Real-Life Example

Predict whether a student will pass an exam

Input features:

- **Study hours**
- **Attendance**

Output:

- **1 → Pass**
- **0 → Fail**

If the probability (output of sigmoid) is 0.78 → student will Pass.
If probability is 0.32 → student will Fail.

Pseudocode for Logistic Regression

Training Phase (Gradient Descent)

```
Initialize weights w and bias b

Set learning_rate = α


Repeat until convergence:

    z = w * x + b

    y_pred = sigmoid(z)


    Compute loss = -[y*log(y_pred) + (1-y)*log(1-y_pred)]
```

```
    dw = (1/m) * Σ ((y_pred - y) * x)

    db = (1/m) * Σ (y_pred - y)


    w = w - learning_rate * dw

    b = b - learning_rate * db
```

Prediction Phase

```
z = w * x + b

y_pred = sigmoid(z)


if y_pred >= 0.5:

    return 1

else:

    return 0
```

📌 7. Pseudocode Example with Sample Data

Suppose dataset:

Hours	Pass (y)

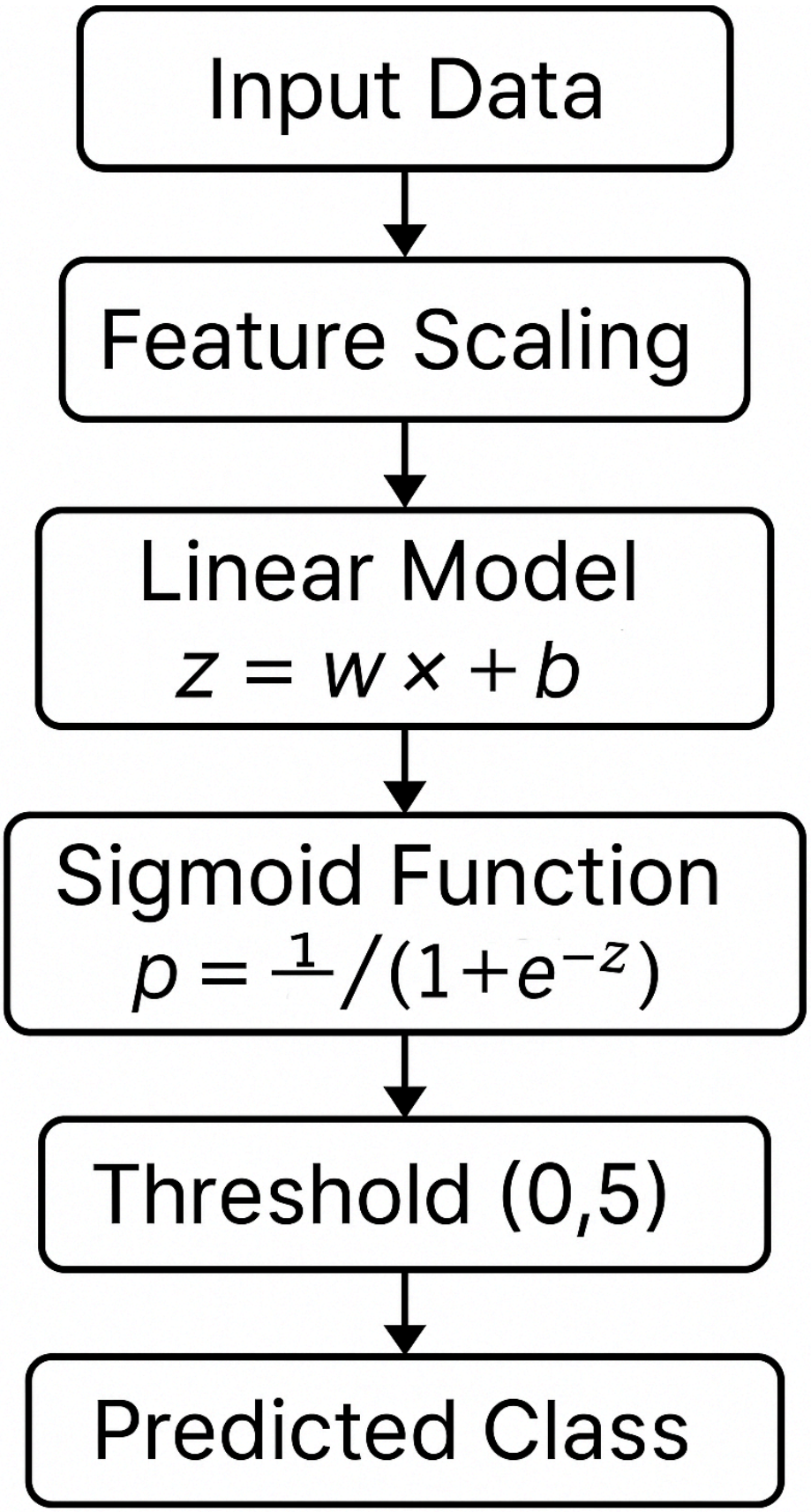
2	0
4	0
6	1
7	1

Let w = 0.1, b = 0.2

One training iteration:

```
For each sample:
    z = w*hours + b
Apply sigmoid:
    y_pred = 1/(1 + e^-z)
Compute error:
    error = y_pred - y
Update:
    dw = error * hours
db = error
After loop:
w = w - α * dw
    b = b - α * db
```

This continues until loss becomes small.


Input Data
Feature Scaling
Linear Model
$z = w x + b$
Sigmoid Function
$p = 1/(1+e^{-z})$
Threshold (0,5)
Predicted Class

Project: Bank Loan Eligibility Prediction

- Input: Income, credit score, employment
- Output: Loan approved or not.

🏦 Bank Loan Eligibility Prediction

A complete Machine Learning project that predicts whether a bank should approve a loan for a customer based on financial and demographic features.

📌 Features
- ML Model: Logistic Regression
- Dataset: 1000 synthetic loan applicant records
- Web App: Streamlit app for real-time predictions
- Fully interactive input form
- Probability-based output

📁 Project Structure

```
loan_eligibility_project/
│
├── data/
│   └── loan_data.csv
│
├── model/
│   ├── train_model.py
│   ├── loan_model.pkl
│   └── scaler.pkl
│
├── app/
│   ├── app.py
│   └── requirements.txt
│
└── README.md
```

yaml
Copy code

🚀 How to Run the Project

###1️⃣ Install dependencies
pip install -r app/requirements.txt

bash
Copy code

###2️⃣ Train the Model
cd model
python train_model.py

markdown
Copy code

This generates:
- `loan_model.pkl`
- `scaler.pkl`

###3️⃣ Run the Streamlit App
cd ../app
streamlit run app.py

yaml
Copy code

Your app will open in the browser.

🔮 Future Enhancements
- Add loan purpose
- Add customer salary slips / credit history PDF upload
- Deploy app on cloud (Render / Hugging Face / Streamlit Cloud)

👨‍💻 Author
Akshay – AI/ML Developer