Logistic Regression

Introduction

Logistic Regression is a widely used algorithm for binary classification tasks. Unlike Linear Regression, which predicts continuous values, Logistic Regression estimates the probability that a given input belongs to a particular class. This probability is obtained by applying the sigmoid activation function to a weighted sum of the input features. Despite its name, Logistic Regression is a classification algorithm rather than a regression model.

Mathematical Foundation

1. Linear Combination of Features

Given an input feature vector $x = (x_1, x_2, \dots, x_n)$, weights $w = (w_1, w_2, \dots, w_n)$, and bias b, the model computes a weighted sum:

$$z = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b = w^T x + b \tag{1}$$

2. Sigmoid Activation Function

To transform the linear combination into a probability, we apply the sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2}$$

This function maps any real-valued number into the range $(0, 1)$, making it suitable for probability estimation.

3. Binary Classification Decision Rule

The probability output is converted into a class label using a threshold (typically 0.5):

$$\hat{y} = \begin{cases} 1 & \text{if } \sigma(z) \geq 0.5 \\ 0 & \text{if } \sigma(z) < 0.5 \end{cases} \tag{3}$$

4. Loss Function: Binary Cross-Entropy

To train the model, we minimize the Binary Cross-Entropy loss function:

$$J(w,b) = -\frac{1}{N}\sum_{i=1}^{N}\left[y^{(i)}\log(\hat{y}^{(i)}) + (1-y^{(i)})\log(1-\hat{y}^{(i)})\right] \tag{4}$$

Where:

- $y^{(i)}$ is the actual label (0 or 1).
- $\hat{y}^{(i)}$ is the predicted probability.
- N is the total number of training samples.

5. Optimization via Gradient Descent

To minimize the loss function, we update the model parameters using Gradient Descent:

$$w_j \leftarrow w_j - \alpha\frac{\partial J}{\partial w_j}, \quad b \leftarrow b - \alpha\frac{\partial J}{\partial b} \tag{5}$$

Where α is the learning rate. The gradients are computed as:

$$\frac{\partial J}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}^{(i)} - y^{(i)})x_j^{(i)}, \quad \frac{\partial J}{\partial b} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}^{(i)} - y^{(i)}) \tag{6}$$

Algorithm Steps

1. Initialize weights and bias to small random values or zeros.
2. Repeat until convergence:
   (a) Compute predictions using the current weights and bias.
   (b) Calculate the gradient of the loss function.
   (c) Update weights and bias using gradient descent.
3. Stop when parameter updates become sufficiently small or after a set number of iterations.

Key Characteristics

- Type: Binary classification model (extendable to multi-class via softmax).
- Activation Function: Sigmoid function for probability estimation.
- Loss Function: Binary Cross-Entropy for classification tasks.
- Optimization: Typically trained using Gradient Descent.

Strengths

- Simple, interpretable, and computationally efficient.
- Provides probabilistic outputs, making classification decisions more explainable.
- Works well when classes are linearly separable.

Weaknesses

- Assumes a linear relationship between input features and log odds, which may not always be valid.
- Sensitive to outliers, as extreme values can significantly affect the decision boundary.
- Can struggle in high-dimensional feature spaces without regularization.

Applications

Logistic Regression is widely used across industries due to its simplicity and interpretability:

- Medical Diagnosis: Predicting the likelihood of diseases based on patient data.
- Email Spam Detection: Classifying emails as spam or not spam.
- Credit Risk Assessment: Estimating the probability of loan defaults.
- Marketing: Predicting whether a customer will respond to an ad campaign.

Conclusion

Logistic Regression remains one of the most widely used classification algorithms due to its ease of implementation, interpretability, and firm performance on linearly separable data. While it has limitations in handling complex patterns, it provides a solid foundation for more advanced machine-learning models.