Machine learning - Logistic regression

Botond Koroknai

2025.08.24

1 Introduction

Logistic regression is a supervised machine learning algorithm primarily used for binary classification, which predicts the probability of an outcome belonging to one of two categories (e.g., yes/no, true/false, 0/1). The name can be misleading, since its a classification model, not a regression one.

The core concept of the model is that it uses a sigmoid function to convert the output of a linear equation into a probability value between 0 and 1, which later can be used for classification purposes.

1.1 Why is logistic regression considered fundamental in the world of machine learning?

- **Simplicity:** Logistic regression is mathematically straightforward and computationally inexpensive. It is fast to train and deploy, even on very large datasets.
- **Interpretability:** The model's coefficients provide clear, direct insights into the relationship between each input feature and the likelihood of the outcome. This transparency is crucial in applications like healthcare, finance, and social sciences, where understanding the why behind a prediction is as important as the prediction itself.
- **Probabilistic Output:** Unlike some classifiers that only provide a hard "yes" or "no" answer, logistic regression outputs a probability score (between 0 and 1). This probabilistic output gives users more control and flexibility, allowing them to set different thresholds for classification based on the specific needs of their application and the acceptable level of risk for making a wrong prediction.

1.2 Assumptions

- The outcome must be categorical: The model is designed for a target variable with a limited, distinct number of outcomes, such as "yes" or "no." It can't predict a continuous value like a person's height or a stock price.
- Independent Observations: We must think of each data point as a unique, self-contained event.

- Sufficiently large sample size: Since the model uses maximum likelihood estimation, it requires enough data to produce reliable and stable results.

source: https://www.youtube.com/watch?v=nmB1XG18Xys

2 Mathematical background

2.1 Step 1 - Basics

We start with a dataset of m samples:

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{m}$$

where:

- $\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}]^T \in \mathbb{R}^n$ represents **input features** for sample i,
- $y^{(i)} \in \{0, 1\}$ is the **binary output**,
- The goal is to predict the **probability that** $y = 1$ given $\mathbf{x}$.

By using logistic regression, we first compute a linear combination of input features:

$$z^{(i)} = \mathbf{w}^T \mathbf{x}^{(i)} + b$$

where:

- $z^{(i)}$ is called the linear predictor and it represents an underlying "score" that determines the probability of the outcome,
- $\mathbf{w} \in \mathbb{R}^n$ is the weight which controls how much each feature contributes,
- $b \in \mathbb{R}$ is the bias term.

While $z^{(i)}$ is linear in the features (like in linear regression), it cannot be used directly as a probability because it can take any real value. To overcome this problem we use a function called sigmoid to map it into the valid probability range of $[0, 1]$.

2.2 Step 2 - Sigmoid Function and Probability Mapping

The sigmoid (also called logistic) function is defined as:

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

This function has several important properties:

- Range: $\sigma(z) \in (0, 1)$ for all $z \in \mathbb{R}$, making it suitable for modeling probabilities.
- Monotonicity: larger z implies larger $\sigma(z)$, which preserves the ordering of the linear predictor.
- Symmetry: $\sigma(-z) = 1 - \sigma(z)$, meaning positive and negative scores are mirror images.

We use the sigmoid to map the linear predictor $z^{(i)}$ to a probability:

$$\hat{y}^{(i)} = P(y^{(i)} = 1 \mid \mathbf{x}^{(i)}) = \sigma(z^{(i)}) = \frac{1}{1 + e^{-(\mathbf{w}^T \mathbf{x}^{(i)} + b)}}$$

Thus:

$$P(y^{(i)} = 0 \mid \mathbf{x}^{(i)}) = 1 - \hat{y}^{(i)}$$

If we express the probabilities in terms of *odds*:

$$\text{odds} = \frac{P(y = 1 \mid \mathbf{x})}{P(y = 0 \mid \mathbf{x})} = \frac{\hat{y}}{1 - \hat{y}} = \frac{\sigma(z)}{1 - \sigma(z)} = \frac{\frac{1}{1+e^{-z}}}{1 - \frac{1}{1+e^{-z}}} = \frac{\frac{1}{1+e^{-z}}}{\frac{e^{-z}}{1+e^{-z}}} = \frac{1}{e^{-z}} = e^z$$

and by taking its logarithm, we get the *log-odds*, also called the *logit*:

$$\log \frac{\hat{y}}{1 - \hat{y}} = z = \mathbf{w}^T \mathbf{x} + b$$

This shows that logistic regression is linear in the log-odds of the probability, even though it is nonlinear in the probability itself.

Besides mapping z into a valid probability, the sigmoid is also computationally convenient. Its derivative has a particularly simple closed form:

$$\begin{aligned} \frac{d\sigma(z)}{dz} &= -1 \cdot (1 + e^{-z})^{-2} \cdot \frac{d}{dz}(1 + e^{-z}) && (1) \\ &= -(1 + e^{-z})^{-2} \cdot (-e^{-z}) && (2) \\ &= \frac{e^{-z}}{(1 + e^{-z})^2}. && (3) \end{aligned}$$

Now express this in terms of $\sigma(z)$:

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

Multiply these two expressions:

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

Therefore,

$$\boxed{\frac{d\sigma(z)}{dz} = \sigma(z)\big(1 - \sigma(z)\big).}$$

This property highlights an important aspect:

- The gradient depends only on the function value itself, not on z directly.

2.3 Step 3 - Maximum Likelihood Estimation (MLE)

Having defined the predicted probability through the sigmoid function, the next fundamental task in logistic regression is to determine the optimal parameters, $\mathbf{w}$ (weights) and b (bias), that allow the model to make accurate predictions on unseen data. Intuitively, these parameters control how much influence each feature has on the final probability prediction.

Logistic regression approaches this parameter selection using the framework of maximum likelihood estimation (MLE). The key idea behind MLE is rooted in probability theory: given a set of observed outcomes (our dataset $\mathcal{D}$), we want to find the parameter values that make the observed data most probable under the model. In other words, if we treat the observed labels as outcomes of a probabilistic process modeled by the sigmoid function, MLE identifies the parameters that maximize the likelihood that the model would generate exactly those outcomes.

Formally, for each individual observation $(\mathbf{x}^{(i)}, y^{(i)})$, the model assigns a probability to the observed outcome, $P(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}, b)$. If $y^{(i)} = 1$, this is simply the predicted probability $\hat{y}^{(i)}$, and if $y^{(i)} = 0$, it is $1 - \hat{y}^{(i)}$. Since we generally assume that all samples are independent, the probability of observing the entire dataset is the product of the individual probabilities. This combined probability is called the likelihood function, denoted as $\mathcal{L}(\mathbf{w}, b)$. Maximizing this function ensures that the parameters chosen make the observed labels as likely as possible according to the model.

To **go into details** a little bit, for a single data point $(\mathbf{x}^{(i)}, y^{(i)})$, the probability of observing the actual label $y^{(i)}$ given the model parameters is:

$$P(y^{(i)} = 1 \mid \mathbf{x}^{(i)}, \mathbf{w}, b) = \hat{y}^{(i)}, \quad P(y^{(i)} = 0 \mid \mathbf{x}^{(i)}, \mathbf{w}, b) = 1 - \hat{y}^{(i)}$$

where $\hat{y}^{(i)} = \sigma(\mathbf{w}^T \mathbf{x}^{(i)} + b)$ is the predicted probability from the sigmoid function. We can write these two cases in a single formula by raising the probability to the power of the observed outcome $y^{(i)}$:

$$P(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}, b) = (\hat{y}^{(i)})^{y^{(i)}} \left(1 - \hat{y}^{(i)}\right)^{1 - y^{(i)}}.$$

Explanation:

- If $y^{(i)} = 1$, the formula becomes $(\hat{y}^{(i)})^1 (1 - \hat{y}^{(i)})^0 = \hat{y}^{(i)}$, which is exactly the probability of observing a positive label.
- If $y^{(i)} = 0$, the formula becomes $(\hat{y}^{(i)})^0 (1 - \hat{y}^{(i)})^1 = 1 - \hat{y}^{(i)}$, which is the probability of observing a negative label.
- This compact form is convenient because it unifies both cases into a single expression suitable for further derivations, like the likelihood function.

Having defined the probability of a single observation, we now extend this idea to the entire dataset. Assuming that each sample $(\mathbf{x}^{(i)}, y^{(i)})$ is independent of the others (a standard assumption in logistic regression), the probability of observing all m samples is the product of the individual probabilities:

$$\mathcal{L}(\mathbf{w}, b) = \prod_{i=1}^{m} P(y^{(i)} \mid \mathbf{x}^{(i)}, \mathbf{w}, b) = \prod_{i=1}^{m} (\hat{y}^{(i)})^{y^{(i)}} \left(1 - \hat{y}^{(i)}\right)^{1-y^{(i)}}.$$

Explanation:

- $\mathcal{L}(\mathbf{w}, b)$ is called the **likelihood function**. It is a function of the parameters $\mathbf{w}$ and b, and it tells us how "likely" the observed data is for a particular choice of parameters.
- The maximum likelihood principle states that the best parameters are those that maximize this likelihood. Intuitively, we are trying to find $\mathbf{w}$ and b that make the model assign the highest probability to the labels we actually observed in our dataset.

However, computing the product of many numbers under 1 can lead to numerical instability. To address this, we take the natural logarithm of the likelihood, yielding the **log-likelihood function**:

$$\ell(\mathbf{w}, b) = \log \mathcal{L}(\mathbf{w}, b) = \sum_{i=1}^{m} \Big[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \Big].$$

- The logarithm converts the product of probabilities into a sum because of the fundamental property:
$$\log \prod_{i=1}^{m} a_i = \sum_{i=1}^{m} \log a_i.$$
This property follows directly from the definition of the logarithm as the inverse of exponentiation. Specifically, for any positive numbers a and b:
$$\log(ab) = \log a + \log b.$$
Extending this property repeatedly to m terms gives the sum over all indices i.
- This transformation is particularly useful in maximum likelihood estimation. By converting a product of probabilities into a sum of logarithms:
  - Differentiation becomes much simpler, since the derivative of a sum is just the sum of derivatives.
  - Numerical stability is improved, because multiplying many small probabilities can result in very tiny numbers, which may underflow in computer calculations. Adding their logarithms avoids this problem.

This usage of this form is also intuitive:

- If a sample has $y^{(i)} = 1$, the term $y^{(i)} \log \hat{y}^{(i)}$ pushes $\hat{y}^{(i)}$ higher, encouraging the model to assign a greater probability to the positive class.
- If a sample has $y^{(i)} = 0$, the term $(1-y^{(i)}) \log(1-\hat{y}^{(i)})$ pushes $\hat{y}^{(i)}$ lower, encouraging the model to assign a greater probability to the negative class.

- Summing over all samples balances these adjustments, leading to parameter values that best fit the overall dataset.

To maximize the log-likelihood, we compute its gradient with respect to the parameters:

$$\frac{\partial \ell}{\partial w_j} = \sum_{i=1}^{m} \big(y^{(i)} - \hat{y}^{(i)}\big) x_j^{(i)}, \quad \frac{\partial \ell}{\partial b} = \sum_{i=1}^{m} \big(y^{(i)} - \hat{y}^{(i)}\big).$$

- Recall that the log-likelihood is defined as

$$\ell(\mathbf{w}, b) = \sum_{i=1}^{m} \Big[y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \Big],$$

  where $\hat{y}^{(i)} = \sigma(\mathbf{w}^T \mathbf{x}^{(i)} + b)$.

- To compute the gradient, we apply the chain rule. For w_j:

$$\frac{\partial \ell}{\partial w_j} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{y}^{(i)}} \cdot \frac{\partial \hat{y}^{(i)}}{\partial z^{(i)}} \cdot \frac{\partial z^{(i)}}{\partial w_j}.$$

  Each term is:

$$\begin{aligned} \frac{\partial \ell}{\partial \hat{y}^{(i)}} &= \frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}}, \\ \frac{\partial \hat{y}^{(i)}}{\partial z^{(i)}} &= \hat{y}^{(i)}(1 - \hat{y}^{(i)}), \\ \frac{\partial z^{(i)}}{\partial w_j} &= x_j^{(i)}. \end{aligned}$$

  Multiplying these together simplifies to $y^{(i)} - \hat{y}^{(i)}$, giving the gradient formula:

$$\frac{\partial \ell}{\partial w_j} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) x_j^{(i)}.$$

- Similarly, for the bias term b, since $\frac{\partial z^{(i)}}{\partial b} = 1$, the gradient is

$$\frac{\partial \ell}{\partial b} = \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}).$$

- These gradients have a clear interpretation: they measure the discrepancy between the observed labels $y^{(i)}$ and the predicted probabilities $\hat{y}^{(i)}$. Positive values indicate that the model underestimates the probability of $y = 1$, prompting an increase in the corresponding weight or bias; negative values indicate overestimation, prompting a decrease.

- Using these gradients, we can iteratively update the parameters via **gradient ascent**:

$$w_j \leftarrow w_j + \alpha \frac{\partial \ell}{\partial w_j}, \quad b \leftarrow b + \alpha \frac{\partial \ell}{\partial b},$$

  where α is the learning rate. This process continues until convergence (when the log-likelihood reaches a maximum or changes negligibly between iterations.)

3 Sources:

- https://en.wikipedia.org/wiki/Bernoulli_distribution
- https://www.geeksforgeeks.org/machine-learning/understanding-logistic-regression/
- https://en.wikipedia.org/wiki/Logistic_regression