Logistic Regression: An Intuitive Guide

Methods and Algorithms – MSc Data Science

L02 Introductory Reading

Every day, banks decide who gets a loan and who doesn't. Doctors decide whether a patient needs further testing. Email systems decide what is spam. Behind all of these decisions is the same question: *"What is the chance that something is true?"* This guide explains one of the most widely used tools for answering that question — **logistic regression**. No background in statistics or mathematics is assumed; every idea is built from scratch using plain language and concrete examples.

Yes-or-No Questions Need Probabilities

Imagine Maria applies for a credit card. The bank needs to decide: will she pay back what she borrows, or won't she? A flat "yes" or "no" is not very useful. The bank really wants to know *how likely* she is to pay back. In other words, it wants a number that captures Maria's risk.

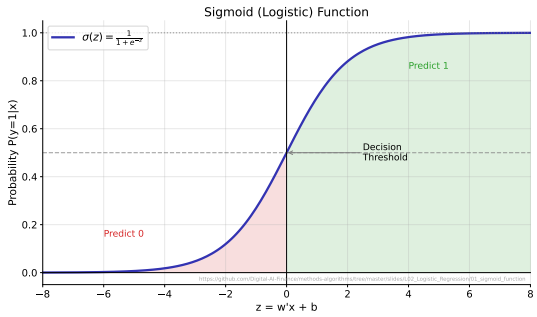
The ideal answer is a probability — a number between 0 and 1. A probability of 0 means "she will definitely not repay." A probability of 1 means "she will definitely repay." Anything in between expresses the bank's level of confidence.

The problem. Suppose we gather information about Maria — her income, how long she has been employed, how much debt she already has — and assign points for each piece of information. We might add up all the points and get a score like 2.5 or −0.3. These numbers are perfectly fine as scores, but they are *not* probabilities because they can be larger than 1 or smaller than 0.

The solution: the S-shaped curve.

We need a mathematical "translator" that takes *any* score and squishes it into the range from 0 to 1. That translator is the **sigmoid function**, and its shape is a gentle S-curve (see the figure to the right). The formula is:

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



In plain English: take your score z, flip its sign, raise the number e (approximately 2.718) to that power, add 1, then divide 1 by the result. No matter what score you start with, the output is always between 0 and 1.

A quick numerical check. If the score is exactly 0, the sigmoid gives 0.5 — a 50/50 chance, right on the fence. If the score is a large positive number like 3, the sigmoid gives about 0.95 — very likely. If the score is a large negative number like −3, the sigmoid gives about 0.05 — very unlikely. The curve smoothly connects these extremes.

How Does the Model Make Predictions?

Features and Weights

The pieces of information we know about a person — income, age, years of employment, existing debts — are called **features**. Think of features as the columns in a spreadsheet where each row is one applicant.

The model assigns a **weight** to each feature. A weight is just a number that says how important that feature is and in which direction it pushes the prediction:

- A *positive* weight means this feature increases the chance of repayment.
- A *negative* weight means this feature decreases the chance of repayment.
- A weight near zero means the feature has little influence.

Think of it like a scorecard: each feature earns you points (positive or negative), and the total determines your final score.

A Step-by-Step Example

Let us walk through Maria's application. The bank's model has already learned the following weights from thousands of past applicants:

Feature	Maria's Value	Weight
Starting value (intercept)	–	-2.10
Monthly income ($, in thousands)	4.5	$+0.42$ per thousand
Debt-to-income ratio	0.35	-1.15
Years employed	6	$+0.35$ per year
Years of credit history	8	$+0.18$ per year
Past missed payments	1	-0.90 per event

Step 1 — Compute the score. Multiply each of Maria's values by the corresponding weight and add everything up:

$$\begin{aligned}\text{Score} &= -2.10 + (0.42 \times 4.5) + (-1.15 \times 0.35) \\ &\quad + (0.35 \times 6) + (0.18 \times 8) + (-0.90 \times 1) \\ &= -2.10 + 1.89 - 0.40 + 2.10 + 1.44 - 0.90 = 2.03.\end{aligned}$$

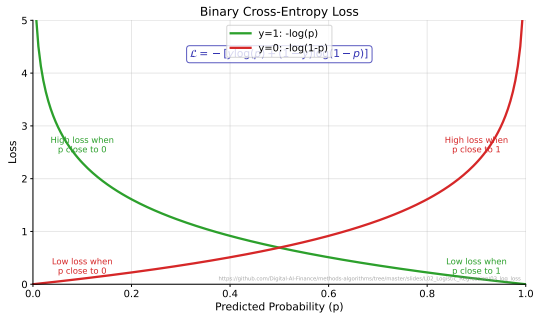
Step 2 — Pass through the S-curve. Feed the score of 2.03 into the sigmoid: $\sigma(2.03) = 1/(1+e^{-2.03}) \approx 0.883$. Maria's estimated probability of repayment is about **88.3%**.

How Does the Model Learn the Weights?

Think of adjusting the dials on a radio to get the clearest signal. The model starts with random weights and then keeps adjusting them based on past data.

The tool that measures how wrong the model is at any moment is called the **loss function**. The chart to the right shows how it works:

- When the model says "90% chance of repayment" and the person *did* repay, the loss is small (the model was right — good!).
- When the model says "90% chance of repayment" but the person *did not* repay, the loss is very large (the model was confidently wrong — bad!).



The model's goal during *training* is to adjust the weights until the total loss across all past applicants is as small as possible. In practice the computer repeats a simple cycle thousands of times: compute the loss, nudge each weight a tiny amount in the direction that reduces the loss, and repeat.

Check your understanding. If a model predicts a 70% chance of repayment and the person does repay, is the loss large or small? *(Answer: small — the model was mostly right, though not perfectly confident.)*

How Do We Know If the Model Is Any Good?

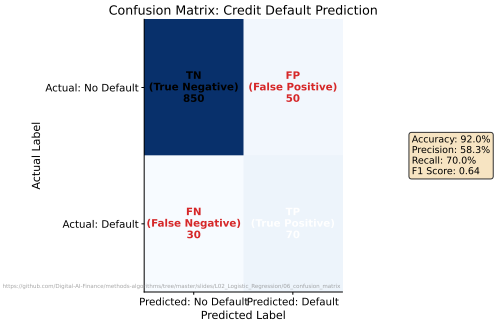
The Confusion Matrix

After training, we test the model on data it has never seen before. For each applicant in the test set, the model predicts "repay" or "default," and we compare with what actually happened. The results are

summarised in a simple 2×2 table called the **confusion matrix**.

Each cell has a name:

- **True Positive (TP):** model said "will repay" and they *did* repay. Correct!
- **False Positive (FP):** model said "will repay" but they did *not*. The bank loses money.
- **True Negative (TN):** model said "won't repay" and they indeed did not. Correctly rejected.
- **False Negative (FN):** model said "won't repay" but they *would* have repaid. The bank lost a good customer.



Precision and Recall

From these four counts we can compute two particularly useful numbers:

Precision answers: "Of everyone we *approved*, what fraction actually repaid?" The formula is Precision $= \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$.

Recall answers: "Of everyone who *would* have repaid, what fraction did we approve?" The formula is Recall $= \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$.

The trade-off. If you approve almost everyone, recall is very high (you miss few good customers) but precision drops (you approve many bad ones too). If you are extremely strict, precision is high but you turn away many good customers. There is no free lunch — improving one usually hurts the other.

Check your understanding. A model approves 100 applicants. Of those, 85 repay and 15 do not. What is the precision? *(Answer:* $85/100 = 85\%$*.)*

The ROC Curve and Choosing a Threshold

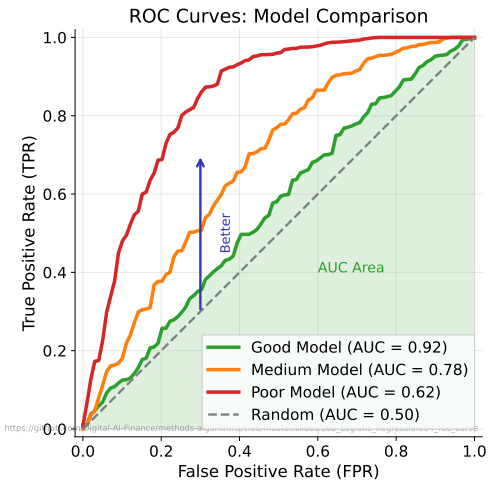
One Picture That Tells the Whole Story

Remember that the model gives each applicant a probability score. To make an actual decision we need a **threshold**: if the probability of repayment is above the threshold, approve; if below, reject.

Different thresholds lead to different trade-offs:

- **Low threshold** (approve almost everyone): we catch nearly all good borrowers but also approve many risky ones.
- **High threshold** (approve almost no one): we reject most risky borrowers but also turn away many good ones.

The **ROC curve** draws *all* possible trade-offs in a single picture. Each point on the curve corresponds to one threshold. A perfect model hugs the top-left corner of the chart; a model that guesses randomly follows the diagonal line.



AUC and Gini — Summarising Quality in One Number

The **AUC** (Area Under the Curve) measures the total area under the ROC curve. It is a single number between 0.5 and 1.0 that summarises how well the model separates good borrowers from bad ones:

- AUC $= 1.0$ means the model is perfect.

- AUC = 0.5 means the model is no better than flipping a coin.

In banking, the same information is often reported as the **Gini coefficient**, which is simply Gini $= 2 \times$ AUC $- 1$. Industry rules of thumb: a Gini above 0.40 is acceptable; above 0.60 is good.

Common Pitfalls

The accuracy trap. Suppose only 1% of borrowers default. A lazy model that always says "will repay" is correct 99% of the time — impressive accuracy! But it is completely useless because it never identifies anyone who might default. *Always* look at precision, recall, and AUC rather than accuracy alone.

The threshold is a business decision. Choosing the threshold is not a purely mathematical exercise. A bank might accept more risk on small personal loans (lower threshold) and demand much more confidence for large mortgages (higher threshold). The model provides the probabilities; the business decides where to draw the line.

Check your understanding. A model has an AUC of 0.80. What is its Gini coefficient? *(Answer:* $2 \times 0.80 - 1 = 0.60$ *— this would be considered a good model in banking.)*

Putting It All Together

Maria's Loan Application — The Full Pipeline

Let us recap every step of Maria's journey through the model:

1. **Collect features**: income \$4,500/month, debt ratio 0.35, employed 6 years, credit history 8 years, 1 past missed payment.
2. **Compute the score**: multiply each feature by its weight and add up. Maria's score is 2.03.
3. **Apply the S-curve**: $\sigma(2.03) \approx 0.883$. Probability of repayment = 88.3%.
4. **Probability of default**: $1 - 0.883 = 0.117$, or 11.7%.
5. **Apply the bank's threshold**: the bank's cutoff is a 15% default rate. Maria's 11.7% is below the cutoff → **APPROVED**.

What if Maria had more missed payments? Suppose she had 3 past missed payments instead of 1. Each one carries a weight of -0.90, so the score drops by $0.90 \times 2 = 1.80$ (two extra missed payments). The new score is $2.03 - 1.80 = 0.23$, and the sigmoid gives $\sigma(0.23) \approx 0.557$. The probability of default is now 44.3% — far above the 15% cutoff → **DENIED**. This illustrates how the model weighs *all* features together, not just one in isolation.

When Should You Use Logistic Regression?

Use logistic regression when:

- The outcome is yes/no (repay or default, spam or not spam, disease or healthy).
- You need to *explain* the decision — each weight tells you exactly how important a feature is and in which direction it pushes the prediction.
- You need a *probability*, not just a label.

Consider alternatives when:

- The patterns in the data are very complex and non-linear (tree-based models or neural networks may perform better).
- The input is unstructured — images, free text, audio.
- Explainability is not required.

In banking, logistic regression is the standard tool because regulators *require* that every lending decision can be explained to the applicant.

Five Things to Remember

1. Logistic regression predicts the **probability** of a yes-or-no outcome.

2. It works by assigning weights to features and passing the total through an S-shaped curve (the sigmoid).
3. The model learns by minimising how often it is confidently wrong (the loss function).
4. We measure quality using the confusion matrix, precision, recall, and the ROC curve (AUC / Gini).
5. It is the standard tool in banking and healthcare because every prediction can be traced back to specific features and their weights.

Glossary of Key Terms

Term	Plain-English Meaning
Feature	A single piece of information about an applicant (e.g. income, age).
Weight	A number that tells the model how important a feature is and in which direction it pushes the prediction.
Sigmoid	The S-shaped curve that converts any score into a probability between 0 and 1.
Loss function	A measure of how wrong the model's predictions are; the model tries to make this as small as possible.
Training	The process of adjusting weights so that the model's predictions match the real outcomes as closely as possible.
Confusion matrix	A 2×2 table that counts the model's correct and incorrect predictions.
Precision	The fraction of approved applicants who actually repaid.
Recall	The fraction of all good applicants who were correctly approved.
Threshold	The cutoff probability above which the model approves an applicant.
ROC curve	A chart that shows all possible trade-offs between catching good borrowers and accidentally approving bad ones.
AUC	Area Under the ROC Curve — a single number (0.5 to 1.0) summarising model quality.
Gini	$2 \times \text{AUC} - 1$; the banking industry's preferred way of reporting AUC.

What to Read Next

This guide has given you the core intuition behind logistic regression. When you are ready to go further, the L02 Self-Study Document covers the same topic at a more technical level, including the mathematical derivation of the learning algorithm, regularization techniques, and formal statistical tests for evaluating whether individual features matter. The references below are also excellent starting points.

References

1. James, G., Witten, D., Hastie, T., Tibshirani, R. (2021). *An Introduction to Statistical Learning*, 2nd ed. Springer. Chapter 4.
2. Molnar, C. (2022). *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/
3. scikit-learn User Guide: Logistic Regression. https://scikit-learn.org/stable/modules/linear_model.html
4. Khan Academy: Introduction to Probability. https://www.khanacademy.org/math/statistics-probability/probability-library