Logistic Regression: A Self-Study Guide

Methods and Algorithms – MSc Data Science

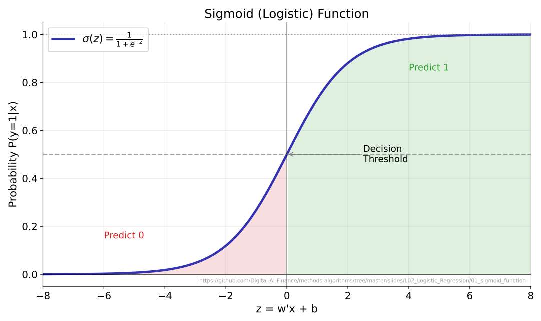
L02 Self-Study Document

Logistic regression is the workhorse of binary classification in finance, healthcare, and marketing. This document covers the mathematical foundations, estimation, inference, and a credit scoring application—everything you need for independent study.

The Logistic Model

The classification problem. In binary classification the response $Y \in \{0, 1\}$ is categorical, so we need a model that maps features $\mathbf{x}$ to a probability $P(Y{=}1 \mid \mathbf{x}) \in [0, 1]$. A naïve approach would be to use the linear function $\mathbf{w}^\top \mathbf{x} + b$ directly, but this can produce values outside $[0, 1]$. Instead, we pass the linear predictor through the **sigmoid function**:

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



Key properties of the sigmoid:

- Range: $\sigma(z) \in (0, 1)$ for all $z \in \mathbb{R}$.
- Centre: $\sigma(0) = 0.5$.
- Symmetry: $\sigma(-z) = 1 - \sigma(z)$.
- Derivative: $\sigma'(z) = \sigma(z)\big(1 - \sigma(z)\big)$.

The derivative property is remarkably elegant: the gradient at any point is determined entirely by the function value itself. This makes back-propagation efficient and is one reason logistic units appear throughout neural networks.

Odds and log-odds. Define the *odds* of the positive class as odds $= p/(1 - p)$. Odds of 3 mean the event is three times more likely to occur than not. The *logit* (log-odds) links the probability to a linear predictor:

$$\text{logit}(p) = \ln \frac{p}{1 - p} = \mathbf{w}^\top \mathbf{x} + b.$$

This equation is the core of logistic regression: the log-odds are a *linear function* of the features. Inverting the logit via the sigmoid recovers the probability.

Coefficient interpretation. Each coefficient acts multiplicatively on the odds: e^{w_j} is the factor by which the odds change when x_j increases by one unit, holding all other features constant. For example, $w_j = 0.42$ implies $e^{0.42} \approx 1.52$, i.e. the odds increase by 52%. If $w_j < 0$, the odds *decrease*; if $w_j = 0$, feature x_j has no effect.

Decision boundary. The model predicts $\hat{Y} = 1$ when $P(Y{=}1 \mid \mathbf{x}) > 0.5$, which occurs when $\mathbf{w}^\top \mathbf{x} + b > 0$. In two dimensions, this boundary is a straight line; in higher dimensions, a hyperplane. The threshold 0.5 can be adjusted depending on the cost of false positives vs. false negatives.

Parameter Estimation

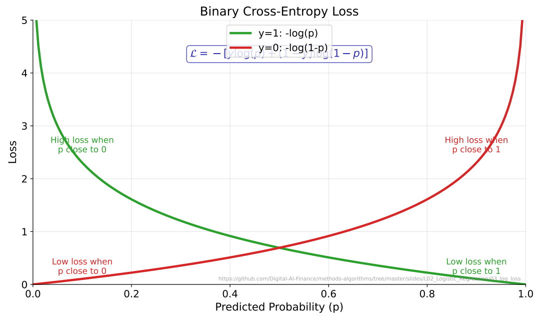
Unlike linear regression, there is no closed-form solution for the logistic regression parameters. We estimate $\mathbf{w}$ via **maximum likelihood estimation** (MLE).

Likelihood function. Let $p_i = \sigma(\mathbf{w}^\top \mathbf{x}_i)$. Assuming observations are independent, the likelihood of the observed labels is:

$$L(\mathbf{w}) = \prod_{i=1}^{n} p_i^{y_i} (1 - p_i)^{1-y_i}.$$

Log-likelihood. Taking logarithms converts the product to a sum, which is easier to optimise:

$$\ell(\mathbf{w}) = \sum_{i=1}^{n} \big[y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \big].$$



Binary cross-entropy. The binary cross-entropy loss is $\text{BCE} = -\frac{1}{n}\,\ell(\mathbf{w})$, so minimising BCE is equivalent to maximising the log-likelihood. The figure to the right shows how the loss penalises confident wrong predictions much more heavily than slightly uncertain correct ones.

Gradient derivation. Applying the chain rule to a single sample yields a remarkably clean expression: $\partial \ell / \partial w_j = (y_i - p_i)\, x_{ij}$. The gradient is proportional to the *residual* $(y_i - p_i)$, just like in linear regression. In matrix form:

$$\nabla_{\mathbf{w}}\, \ell = \mathbf{X}^\top (\mathbf{y} - \mathbf{p}).$$

Gradient descent update: $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta \cdot \frac{1}{n}\, \mathbf{X}^\top (\mathbf{y} - \mathbf{p})$, where η is the learning rate. Convergence is linear (first-order).

Newton–Raphson / IRLS. For faster convergence, we use second-order information. The Hessian of the log-likelihood is:

$$\mathbf{H} = -\mathbf{X}^\top \mathbf{S} \mathbf{X}, \quad \text{where } \mathbf{S} = \text{diag}\big(p_i(1 - p_i)\big).$$

The Newton update is $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathbf{H}^{-1} \nabla \ell$. Because $\mathbf{H}$ is negative semi-definite, the log-likelihood is *concave* and Newton–Raphson converges *quadratically*—typically in 5–10 iterations, versus hundreds for gradient descent. This algorithm is also known as *Iteratively Reweighted Least Squares* (IRLS) because each Newton step solves a weighted least-squares problem with weights $p_i(1 - p_i)$.

Inference and Model Selection

A key advantage of logistic regression over black-box classifiers is its rich statistical inference framework. After fitting, we can test hypotheses about individual coefficients, compare nested models, and construct confidence intervals.

Standard errors. The inverse of the observed information matrix gives approximate variances: $\text{SE}(\hat{w}_j) = \sqrt{[\mathbf{H}^{-1}]_{jj}}$. These standard errors are *asymptotic*—they rely on large-sample theory.

Wald test. To test $H_0\colon w_j = 0$ (i.e., feature x_j is irrelevant), compute:

$$z_j = \frac{\hat{w}_j}{\text{SE}(\hat{w}_j)}.$$

Reject at the 5% level if $|z_j| > 1.96$. Confidence intervals: $\hat{w}_j \pm 1.96 \cdot \text{SE}(\hat{w}_j)$.

Likelihood Ratio Test (LRT). For testing q restrictions jointly (e.g., removing a group of features):

$$\Lambda = -2\big[\ell(\text{reduced}) - \ell(\text{full})\big] \;\sim\; \chi^2_q.$$

The LRT is generally more powerful than the Wald test for small samples and is preferred when comparing nested models.

Model selection criteria.

- AIC $= -2\ell + 2k$, where k is the number of parameters.
- BIC $= -2\ell + k \ln n$, which penalises complexity more heavily.

BIC is preferred for regulatory models in banking because it favours parsimony, leading to more interpretable and stable scorecards.

Regularization

When the number of features is large relative to the sample size, or when features are correlated, the MLE can overfit. Regularization adds a penalty to the negative log-likelihood:

- **L2 (Ridge):** $\min_{\mathbf{w}} -\ell(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2$ — shrinks all coefficients towards zero but retains all features.
- **L1 (Lasso):** $\min_{\mathbf{w}} -\ell(\mathbf{w}) + \lambda\|\mathbf{w}\|_1$ — induces sparsity, effectively performing automatic feature selection.
- **Elastic Net:** $\min_{\mathbf{w}} -\ell(\mathbf{w}) + \lambda\big[\alpha\|\mathbf{w}\|_1 + (1-\alpha)\|\mathbf{w}\|_2^2\big]$ — combines both penalties; useful when features are grouped.

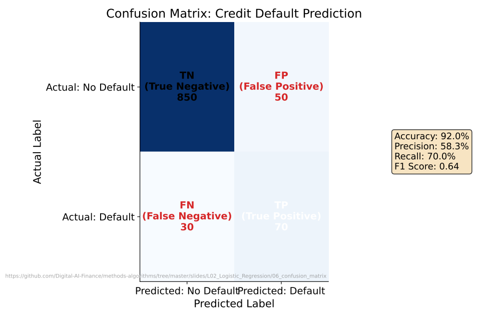
The hyperparameter λ (or $C = 1/\lambda$ in scikit-learn) is chosen via k-fold cross-validation. Larger λ means stronger regularization. A practical rule: use the "one-SE rule" to select the most parsimonious model within one standard error of the minimum CV loss. L1 regularization is particularly popular in credit scoring for producing sparse, interpretable models.

Evaluation Metrics

Confusion matrix. A 2×2 table of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) at a given classification threshold.

Derived metrics:

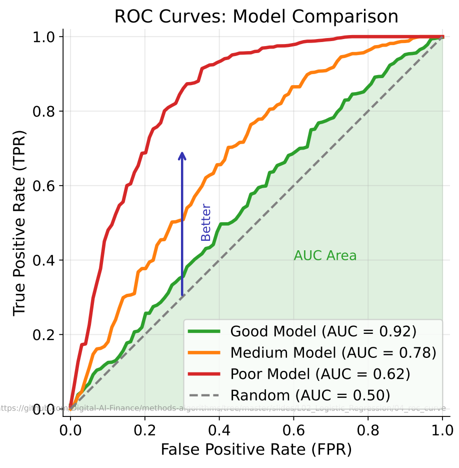
- Precision $= \text{TP}/(\text{TP}+\text{FP})$ — of predicted positives, how many correct?
- Recall $= \text{TP}/(\text{TP}+\text{FN})$ — of actual positives, how many found?
- $F_1 = 2 \cdot \text{Prec} \cdot \text{Rec} \,/\, (\text{Prec} + \text{Rec})$ — harmonic mean balancing both.
- Accuracy $= (\text{TP} + \text{TN})/n$ — overall fraction correct.



ROC curve and AUC. The Receiver Operating Characteristic (ROC) curve plots the True Positive Rate (recall) against the False Positive Rate at varying classification thresholds. A perfect classifier hugs the top-left corner; a random classifier follows the diagonal. The **AUC** (area under the ROC curve) summarises discriminatory power in a single number. It equals the probability that a randomly chosen positive example ranks higher than a randomly chosen negative example.

Gini coefficient. In banking, the Gini coefficient is defined as Gini $= 2 \cdot \text{AUC} - 1$. Industry benchmarks: Gini > 0.40 is acceptable; Gini > 0.60 is good for retail credit.

Warning about imbalanced data. With severely imbalanced data (e.g., a 1% default rate), raw accuracy is meaningless—a model predicting "no default" for everyone



achieves 99% accuracy but is useless. Always evaluate using AUC, Gini, or precision–recall metrics in such settings.

Application: Credit Scoring

Consider a bank building a scorecard to predict credit card default. The model is trained on historical data with five features. We frame the model as $P(\text{repayment} \mid \mathbf{x})$, so that positive coefficients indicate *better* creditworthiness.

Feature	**Variable**	**Example**	$\hat{w}_j$	**Odds Ratio** $e^{\hat{w}_j}$
Intercept	–	–	-2.10	–
Monthly Income (k$)	x_1	4.5	0.42	1.52
Debt-to-Income Ratio	x_2	0.35	-1.15	0.32
Employment Years	x_3	6	0.35	1.42
Credit History (yrs)	x_4	8	0.18	1.20
Past Delinquencies	x_5	1	-0.90	0.41

Worked example. For the sample applicant above, we compute the linear predictor:

$$\begin{aligned} z &= -2.10 + 0.42(4.5) + (-1.15)(0.35) + 0.35(6) \\ &\quad + 0.18(8) + (-0.90)(1) \\ &= -2.10 + 1.89 - 0.4025 + 2.10 + 1.44 - 0.90 \\ &= 2.0275. \end{aligned}$$

The predicted probability of repayment is:

$$P(\text{repay}) = \sigma(2.0275) = \frac{1}{1 + e^{-2.0275}} \approx 0.883.$$

Therefore, the **probability of default** is $\text{PD} = 1 - 0.883 = 0.117$ (11.7%).

Odds ratio interpretation. Each additional year of employment multiplies the odds of repayment by $e^{0.35} = 1.42$, i.e. a 42% increase in odds, all else equal. Conversely, each past delinquency multiplies the odds by $e^{-0.90} = 0.41$, roughly halving them. The debt-to-income ratio has the strongest negative effect: a 0.1 increase in DTI multiplies the odds by $e^{-1.15\times0.1} = e^{-0.115} \approx 0.89$, an 11% reduction.

Basel regulatory context. Under the Basel framework, the *expected loss* for a credit exposure is:

$$\text{EL} = \text{PD} \times \text{LGD} \times \text{EAD}.$$

For our applicant with LGD = 0.45 and EAD = \$25,000: EL = $0.117 \times 0.45 \times 25{,}000$ = \$1,316. This expected loss feeds into the bank's loan-loss provisions and regulatory capital calculations.

Why logistic regression for scorecards? Logistic regression remains the *regulatory gold standard* for credit scoring because: (i) coefficients are directly interpretable as log-odds ratios; (ii) the model produces well-calibrated probabilities; (iii) regulators require that banks can explain every decision; (iv) the model is stable and easy to monitor over time.

Gini benchmarks for retail credit: Gini > 0.40 is acceptable, Gini > 0.60 is good. More complex models (gradient boosting, neural networks) may achieve higher Gini, but they face significant regulatory hurdles around explainability.

Summary

- Logistic regression maps features to calibrated probabilities via the sigmoid function.
- MLE with cross-entropy loss yields a convex problem—guaranteed global optimum.
- Newton–Raphson converges quadratically; gradient descent is simpler but slower.

- The Wald test and LRT assess individual and joint significance of features.
- Regularization (L1/L2/Elastic Net) controls overfitting and performs feature selection.
- In credit scoring, logistic regression remains the regulatory gold standard due to its interpretability and well-understood statistical properties.

References

1. James, G., Witten, D., Hastie, T., Tibshirani, R. (2021). *An Introduction to Statistical Learning*, 2nd ed. Springer. Chapter 4.
2. Hastie, T., Tibshirani, R., Friedman, J. (2009). *The Elements of Statistical Learning*, 2nd ed. Springer. Chapter 4.
3. Molnar, C. (2022). *Interpretable Machine Learning*. https://christophm.github.io/interpretable-ml-book/
4. scikit-learn User Guide: Logistic Regression. https://scikit-learn.org/stable/modules/linear_model.html