# Tuning Regularization in Logistic Regression: From Overfitting to Simplicity

**Student Name: Polepelly Nithish Reddy**
**Student ID: 23105587**
**GitHub:**https://github.com/polepellynithishreddy/Tuning-Regularization-in-Logistic-Regression-From-Overfitting-to-Simplicity/tree/main

---

## 1. Introduction

Logistic regression is a classic method for binary classification that remains widely used because it is simple, efficient, and relatively interpretable. In practice, however, an untuned logistic regression model can either underfit (too simple) or overfit (too complex), leading to unreliable predictions on new data. Regularization is the main tool used to control this trade-off.
This tutorial investigates how L2 regularization affects a logistic regression model on a simple two-dimensional synthetic dataset generated with make_classification from scikit-learn. The focus is on the regularization strength parameter C, and how changing C influences classification accuracy, decision boundaries, and coefficient magnitudes. The goal is to give readers practical intuition for selecting C and recognising under- and overfitting.

By the end of the tutorial, a reader should be able to explain what logistic regression does for binary classification, describe how L2 regularization works, interpret accuracy–versus–C curves in terms of the bias–variance trade-off, and visually compare decision boundaries for weak versus strong regularization.

---

## 2. Logistic regression and L2 regularization

Logistic regression models the probability that a binary label is 1 given a feature vector by applying the sigmoid function to a weighted sum of the inputs. The model computes a score $wTx + bw\hat{}T\,x + bwTx + b$ for each example, converts it to a probability between 0 and 1 using a squashing function, and learns the parameters www and bbb by minimising a loss based on how well predicted probabilities match observed labels.

L2 regularization augments this loss with a penalty proportional to the sum of squared weights, discouraging large coefficients and favouring simpler models that generalise better. In scikit-learn's LogisticRegression, this penalty is controlled indirectly through the hyperparameter C, defined as the inverse of the regularization strength: small C means strong regularization and large C means weak regularization. Exploring a range of C values therefore reveals how model complexity and performance are linked.

---

# 3. Synthetic dataset and baseline model

To isolate the effect of regularization from data-cleaning issues, the tutorial uses a synthetic binary classification dataset created with sklearn.datasets.make_classification. The dataset contains 500 samples with 2 informative features, no redundant features, one cluster per class, and a small amount of label noise (5%), so the classes are mostly but not perfectly separable. The classes are almost balanced (244 vs 256), making accuracy a meaningful and easy-to-interpret metric.

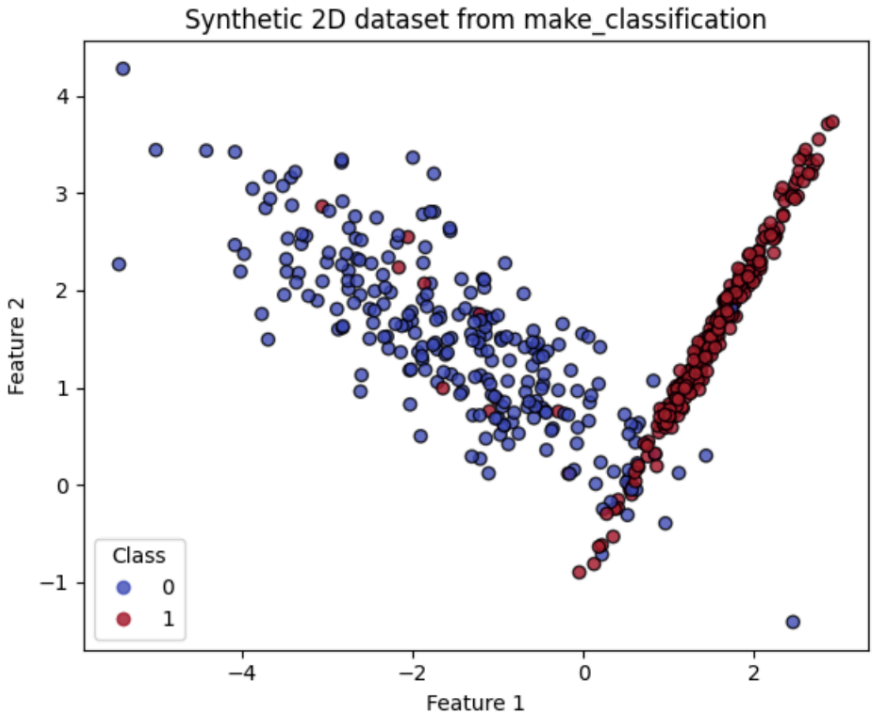


**Figure 1 – Synthetic dataset scatter plot**

The dataset is split into 60% training, 20% validation, and 20% test sets using stratified sampling to preserve class balance. A baseline logistic regression model is implemented as a scikit-learn Pipeline with StandardScaler followed by LogisticRegression with L2 penalty and C = 1.0. On this split, the baseline achieves 0.910 accuracy on both the training and validation sets, and 0.960 on the held-out test set, suggesting that at C = 1.0 the model already strikes a reasonable balance between under- and overfitting.

# 4. Accuracy as a function of C

To study how regularization strength influences performance, models are trained for 13 values of C on a logarithmic grid from 0.001 (very strong regularization) to 1000 (very weak regularization). For each C, training, validation, and test accuracies are computed on the fixed splits.

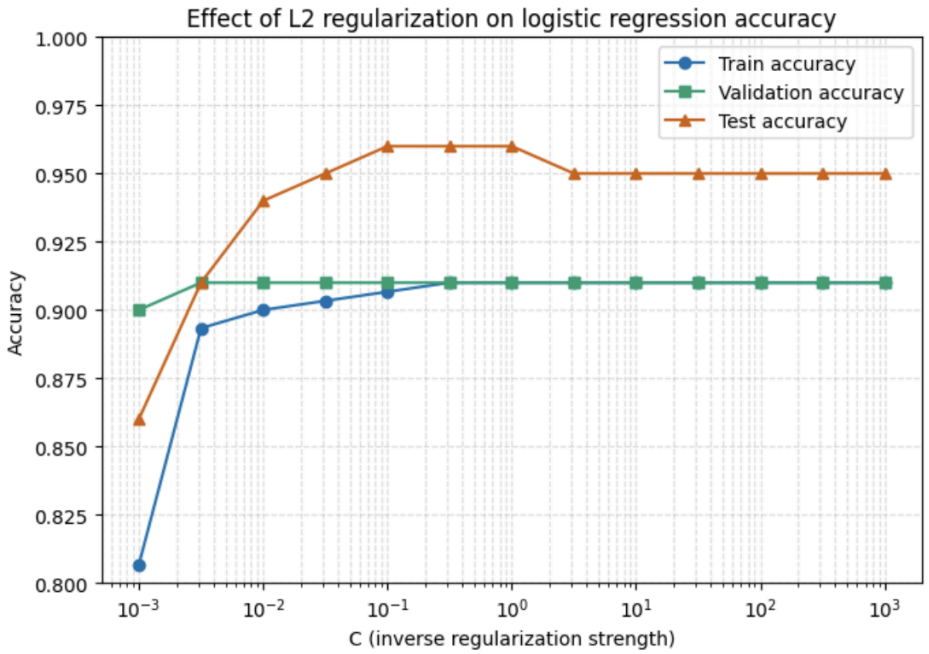


**Figure 2 – Accuracy vs C**

The plot shows that when C is extremely small (around 0.001), the model underfits: training accuracy drops to about 0.81 and test accuracy to about 0.86, because the coefficients are heavily shrunk and the decision boundary is overly simple. As C increases into the range 0.01–1, training accuracy rises to roughly 0.91 and test accuracy peaks near 0.96, while validation accuracy remains stable around 0.91. This region corresponds to a good bias–variance trade-off, where the model is complex enough to fit the main pattern but still robust to noise.

For C values larger than 1, weakening regularization further does not yield meaningful gains: the three accuracy curves remain almost flat, indicating that the model has already captured the structure of this relatively simple dataset. On more complex or higher-dimensional problems, one might expect a clearer overfitting regime (high train, lower validation/test), but on this 2D toy example the plateau simply shows that increasing C beyond a moderate value is unnecessary.

# 5. Decision boundaries under different regularization strengths

Numerical accuracy alone can be abstract, so the tutorial also visualises decision boundaries for selected C values. Three models are trained with C = 0.01 (strong regularization), C = 1.0 (moderate), and C = 1000 (weak), and their decision regions are plotted over the training data.

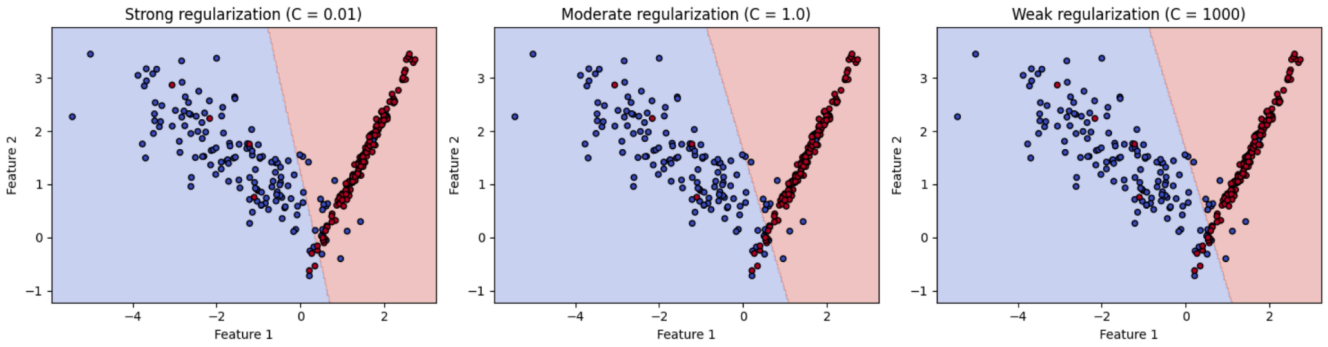


**Figure 3 – Decision boundaries for C = 0.01, 1.0, and 1000**

With strong regularization (C = 0.01), the separating line is nearly vertical and ignores many of the finer details of the data, misclassifying several points near the overlapping region because the weights are tightly constrained. At the moderate setting (C = 1.0), the boundary tilts towards the diagonal structure of the data, correctly classifying most points in both classes while remaining relatively simple. With very weak regularization (C = 1000), the boundary rotates slightly further to accommodate additional training points, making the model more sensitive to noise; however, the change in accuracy is small on this dataset.

These visuals turn the abstract idea of "model complexity" into a concrete geometric picture: strong regularization favours simple, conservative boundaries, while weak regularization allows the boundary to move closer to individual data points.

## 5.1 Understanding the bias–variance trade-off in this experiment

The behaviour observed in this tutorial is a concrete instance of the wider bias–variance trade-off that appears across many machine-learning methods. When the regularization strength is very high (small `C`), the logistic regression model is heavily constrained: it uses coefficients that are close to zero and produces a nearly straight, conservative decision boundary. Such a model has **high bias**, because it cannot capture all the true structure in the data, and **low variance**, because small changes in the training set would not change the fitted line very much. The low training and test accuracies at `C = 0.001` illustrate this underfitting regime clearly.

As `C` increases, the penalty weakens, the coefficients grow, and the model becomes more flexible. In the middle range of `C`, the logistic regression is flexible enough to align with the diagonal separation between the two classes, but regularization is still strong enough to prevent it from chasing every noisy point, so both bias and variance remain moderate. This is where the validation and test accuracies peak. For very large `C`, the model is free to move the boundary in response to individual data points, so variance increases, even if this specific toy dataset does not show a dramatic drop in test accuracy. The combination of accuracy curves, boundary plots, and coefficient magnitudes provides a visual and quantitative way to understand this trade-off, making the abstract bias–variance concept more intuitive for practitioners.

---

# 6. Coefficient magnitudes and model complexity

Because logistic regression is a linear model, regularization directly affects the size of its coefficients. To make this explicit, the absolute values of the two feature coefficients are recorded for the same grid of C values and plotted as a function of C.

For very strong regularization (C = 0.001), both coefficients are close to zero (around 0.11 for the first feature and 0.01 for the second), meaning that the model effectively ignores the input features and produces an almost flat decision boundary. As C increases, the regularization penalty weakens and the coefficients grow in magnitude, reflecting a more decisive use of the features; they eventually plateau near 4.06 and 0.76 for the two features respectively, indicating that beyond a certain point further increases in C change the solution very little.

This experiment illustrates the central role of L2 regularization: by shrinking coefficients towards zero, small C values enforce simpler models, while larger C values allow the weights to expand and fit the data more closely. The earlier accuracy plot shows that on this dataset, moderate coefficient sizes already achieve the best generalisation.

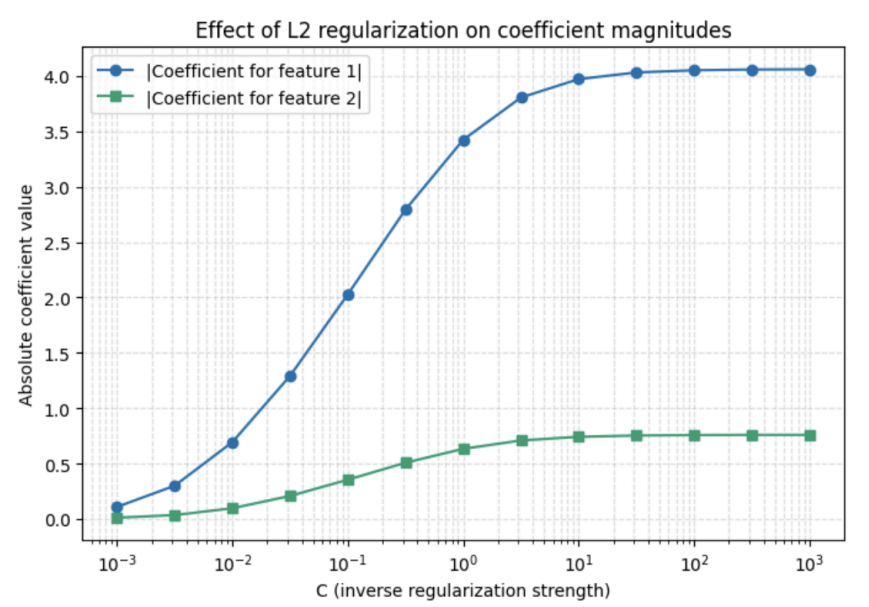


**Figure 4 – Coefficient magnitude vs C**

---

# 7. Ethical and accessibility considerations

Although this study uses a synthetic low-stakes dataset, regularised logistic regression is often deployed in domains such as medical diagnosis, hiring, and credit scoring where each prediction can have significant consequences. An overfitted model may appear accurate on historical data yet generalise poorly, leading to unfair or unsafe decisions, especially when its probabilistic outputs are interpreted as trustworthy risk scores.

Carefully tuning C on a separate validation set reduces overfitting but does not guarantee fairness or eliminate bias: if the training data reflect historical discrimination or sampling gaps, a regularised model can still reproduce these patterns. Responsible use therefore requires combining technical tools such as regularization with critical data checks, domain expertise, and appropriate oversight frameworks.

From an accessibility perspective, the tutorial uses a colour-blind-friendly Matplotlib style and combines colour with marker shapes and clear legends, so plots remain interpretable under common colour-vision deficiencies. Axis labels and titles explicitly describe what each figure shows (for example "Accuracy vs C" or "Effect of L2 regularization on coefficient magnitudes"), and the written report provides concise alt text for each figure so that screen-reader users can follow the narrative even without seeing the images.

---

# 8. Practical guidelines and conclusion

The experiments in this tutorial suggest several simple guidelines for tuning L2-regularised logistic regression:

- Always scale features before fitting the model, especially when using gradient-based solvers.
- Use a validation set (or cross-validation) to explore a range of C values on a logarithmic scale, rather than guessing a single value.
- Watch both performance and coefficient magnitudes: very small C produces tiny weights and underfitting, whereas extremely large C rarely improves accuracy and may increase sensitivity to noise.
- For simple, low-dimensional problems like the 2D dataset used here, a moderate C around 0.1–1 is often sufficient; more complex tasks may require more careful tuning.

Overall, the tutorial shows how regularization controls the balance between simplicity and flexibility in logistic regression, and how the single hyperparameter C can be used to navigate the spectrum from underfitting to potential overfitting. By combining accuracy curves, decision-boundary plots, and coefficient analysis, readers can build an intuitive understanding of L2 regularization that transfers to real-world applications where reliable generalisation is critical.

# 9. References


- Scikit-learn developers. "LogisticRegression — scikit-learn documentation." Accessed 2025. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

- Scikit-learn developers. "make_classification — scikit-learn documentation." Accessed 2025. https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html

- DigitalOcean. "Mastering Logistic Regression with Scikit-Learn: A Complete Guide." 2025. https://www.digitalocean.com/community/tutorials/logistic-regression-with-scikit-learn

- CompGenomR book. "Logistic regression and regularization." 2020.

https://compgenomr.github.io/book/logistic-regression-and-regularization.html

- KNIME. “Understanding regularization for logistic regression: L1, L2, Gauss or Laplace.” 2018. https://www.knime.com/blog/regularization-for-logistic-regression-l1-l2-gauss-or-laplace

- Mostafa-Samir. “Regularization and the Bias–Variance Trade-off.” 2011. https://mostafa-samir.github.io/ml-theory-pt3/

- Sebastian Raschka. “Does regularization in logistic regression always result in better fit?” 2025. https://sebastianraschka.com/faq/docs/regularized-logistic-regression-performance.html

- NIH article, 2024. Health Equity and Ethical Considerations in Using Artificial Intelligence in Public Health and Medicine - PMC

- Infosys BPM. “AI Credit Scoring: Ethical Frameworks.” 2025. https://www.infosysbpm.com/blogs/financial-services/ai-credit-scoring-ethical-framework.html

- IJRPR. “AI-Powered Credit Scoring Models: Ethical Considerations.” https://ijrpr.com/uploads/V6ISSUE3/IJRPR40581.pdf

- Matplotlib documentation. “Choosing Colormaps in Matplotlib.” 2025. Choosing Colormaps in Matplotlib

- Rockborne. “Data Accessibility Tips: Making Visualisations Colour Blind Friendly.” 2023. https://rockborne.com/graduates/blog/data-accessibility-colour-blindness/

- University of Reading. “Accessibility tips: Use of colours.” 2025. Accessibility tips: Use of colours