Supervised Learning: Classification I

Foundations, Core Algorithms, and Evaluation

This lecture introduces the formal framework of classification tasks. We will delineate the core concepts of binary and multiclass classification, the critical methodology of data splitting and cross-validation, and survey foundational algorithms including k-NN, Logistic Regression, Naive Bayes, and Decision Trees. The session concludes with a rigorous treatment of evaluation metrics derived from the confusion matrix and diagnostic tools for model assessment.

Keywords: Classification, Generalization, Inductive Bias, Model Evaluation, Bias-Variance Tradeoff.

Core Concepts: The Classification Task

Definition & Problem Formulation

Supervised Classification

A predictive modeling task where the goal is to learn a mapping function $f: R^n \rightarrow C$ from an input feature space to a discrete output space of class labels.

Binary Classification

The simplest case where : $C = \{0, 1\}$ *or* $C = \{-1, +1\}$.

- Examples: Spam detection (Spam/Ham), Medical diagnosis (Diseased/Healthy).

Multiclass Classification

The general case where `|C| > 2`.

- Examples: Digit recognition (0-9), Object categorization (Cat, Dog, Car).

Objective: Learn a hypothesis `h` from a labeled training set $D = (x_i, y_i)$ that generalises well to unseen data.

Core Concepts: Ensuring Generalization

Data Splitting and Resampling for Robustness

The Problem of Overfitting: A model that memorises the training data fails to perform on new, unseen data, indicating poor generalisation.

Training Set

(~60-80%): Used exclusively to fit the model parameters (weights and biases).

Validation Set

(~10-20%): Used for hyperparameter tuning and model selection. Essential for providing an unbiased evaluation of a model fit during training.

Test Set

(~10-30%): Used only once for a final, unbiased assessment of the model's generalisation error.

k-Fold Cross-Validation

A robust resampling technique to mitigate variability from a single, arbitrary data split. The data is partitioned into *k* folds; the model is trained *k* times, using a different fold for validation each time. The final performance is the average over all *k* iterations. Optimal for small datasets.

Algorithm Survey I: Instance-Based & Probabilistic

k-Nearest Neighbors (k-NN)

Intuition: An instance-based (lazy) learner. It classifies a query point based on the majority vote of its *k* closest training examples in the feature space (typically using Euclidean distance).

Key Hyperparameter: `k` (number of neighbors). A small `k` leads to high variance; a large `k` leads to high bias.

- Non-parametric and simple to implement.
- Computationally expensive at inference time due to distance calculations.

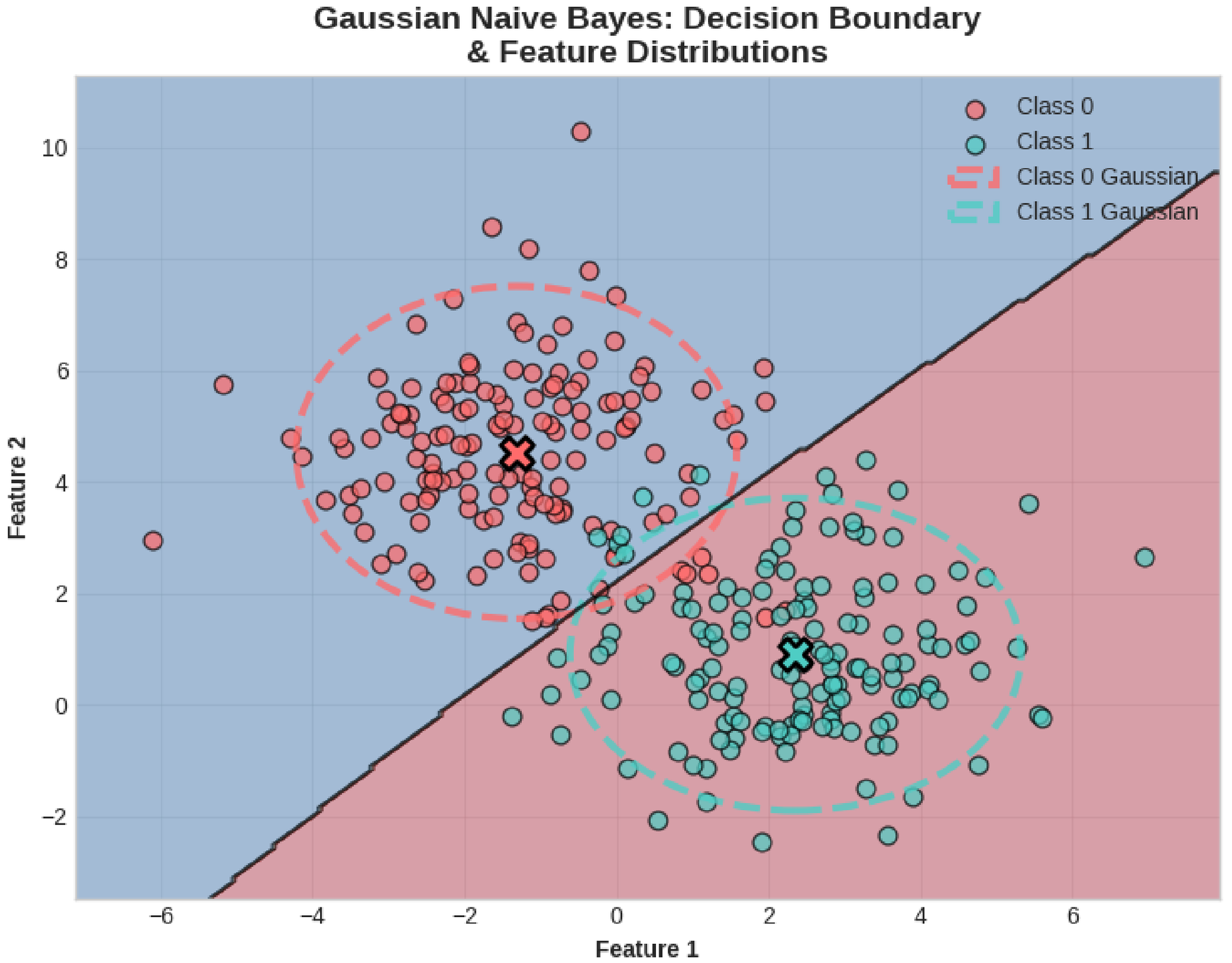
Naive Bayes

Intuition: A powerful probabilistic classifier rooted in Bayes' Theorem. It makes the "naive" assumption of conditional independence between features given the class label.

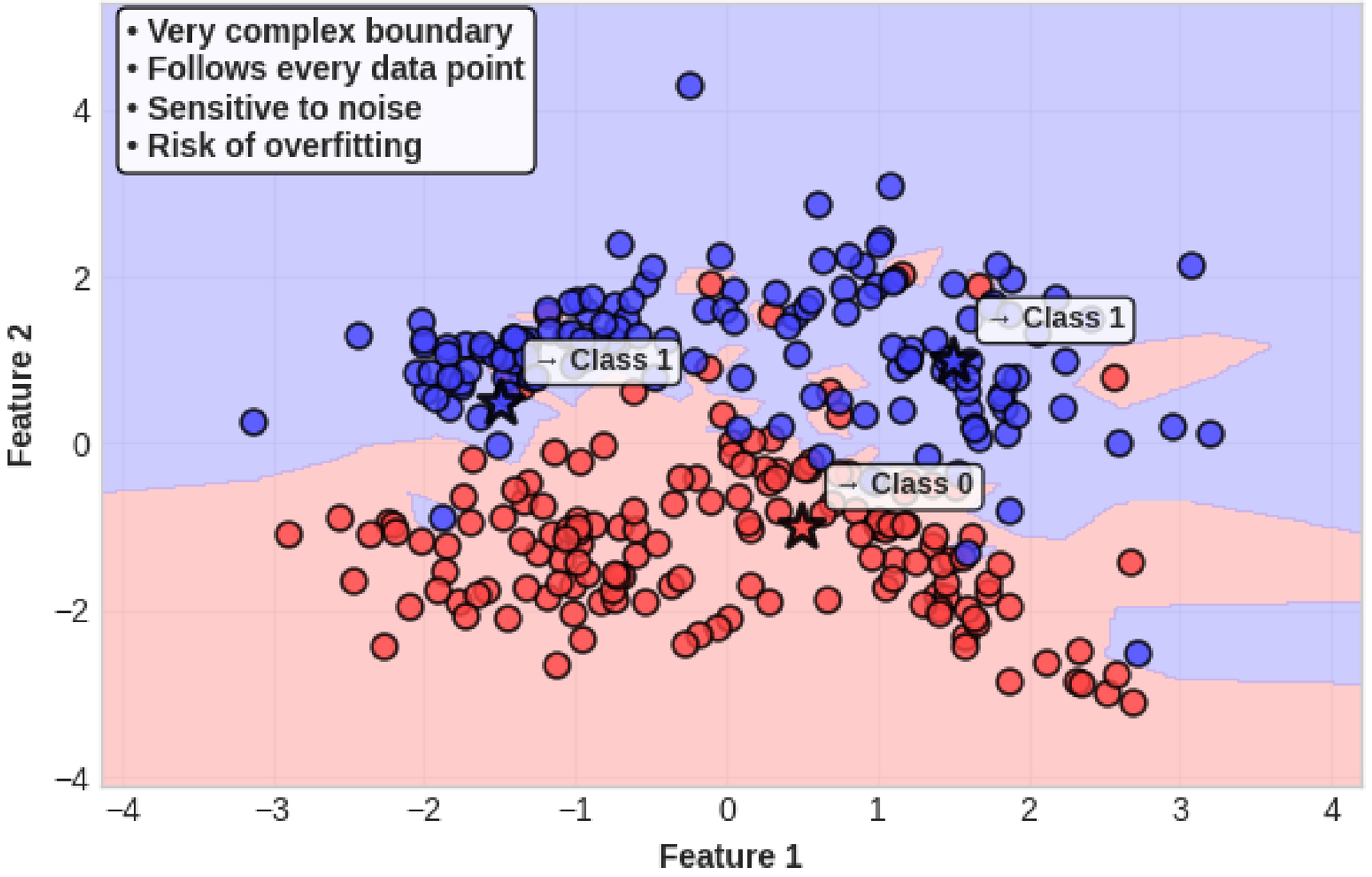
Model Forms:

- **Gaussian:** Assumes continuous features follow a normal distribution.
- **Multinomial:** Suitable for discrete count data, such as word frequencies in text classification.

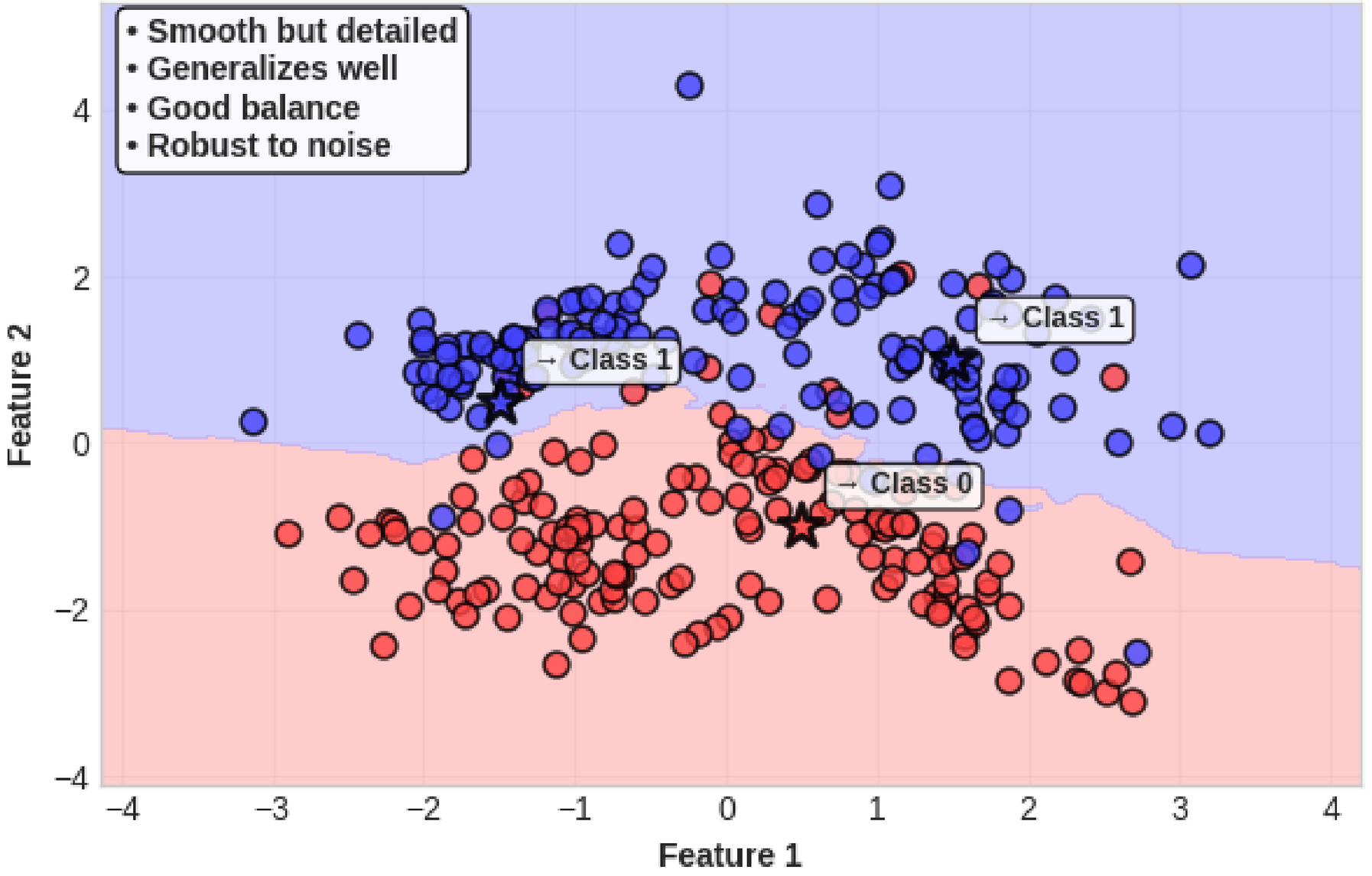
Highly scalable, efficient, and often serves as an excellent, fast baseline model.


Gaussian Naive Bayes: Decision Boundary
& Feature Distributions
Class 0
Class 1
Class 0 Gaussian
Class 1 Gaussian
Feature 2
Feature 1
10
8
6
4
2
0
−2
−6
−4
−2
0
2
4
6

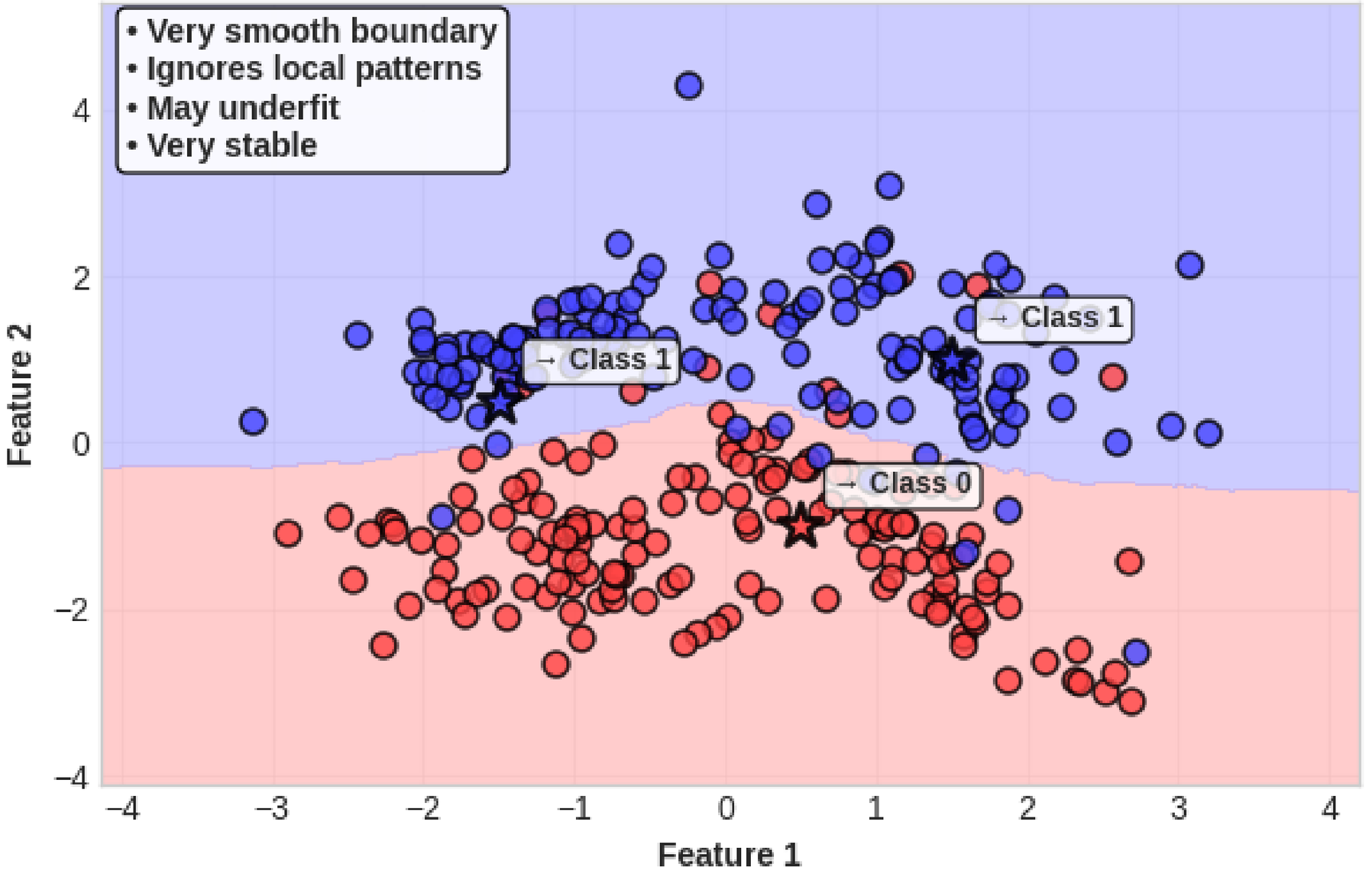
k = 1
High Complexity
(Low Bias, High Variance)



k = 5
Balanced
(Medium Bias/Variance)



k = 50
Low Complexity
(High Bias, Low Variance)



Algorithm Survey II: Linear & Tree-Based

Logistic Regression

Intuition: A linear classifier that uses the logistic sigmoid function to model the probability of an input belonging to a particular class:

$$P(Y = 1 \mid X) = \frac{1}{1 + e^{-(w^T x + b)}}$$

Learning: Parameters (weights `w` and bias `b`) are typically learned through Maximum Likelihood Estimation (MLE).

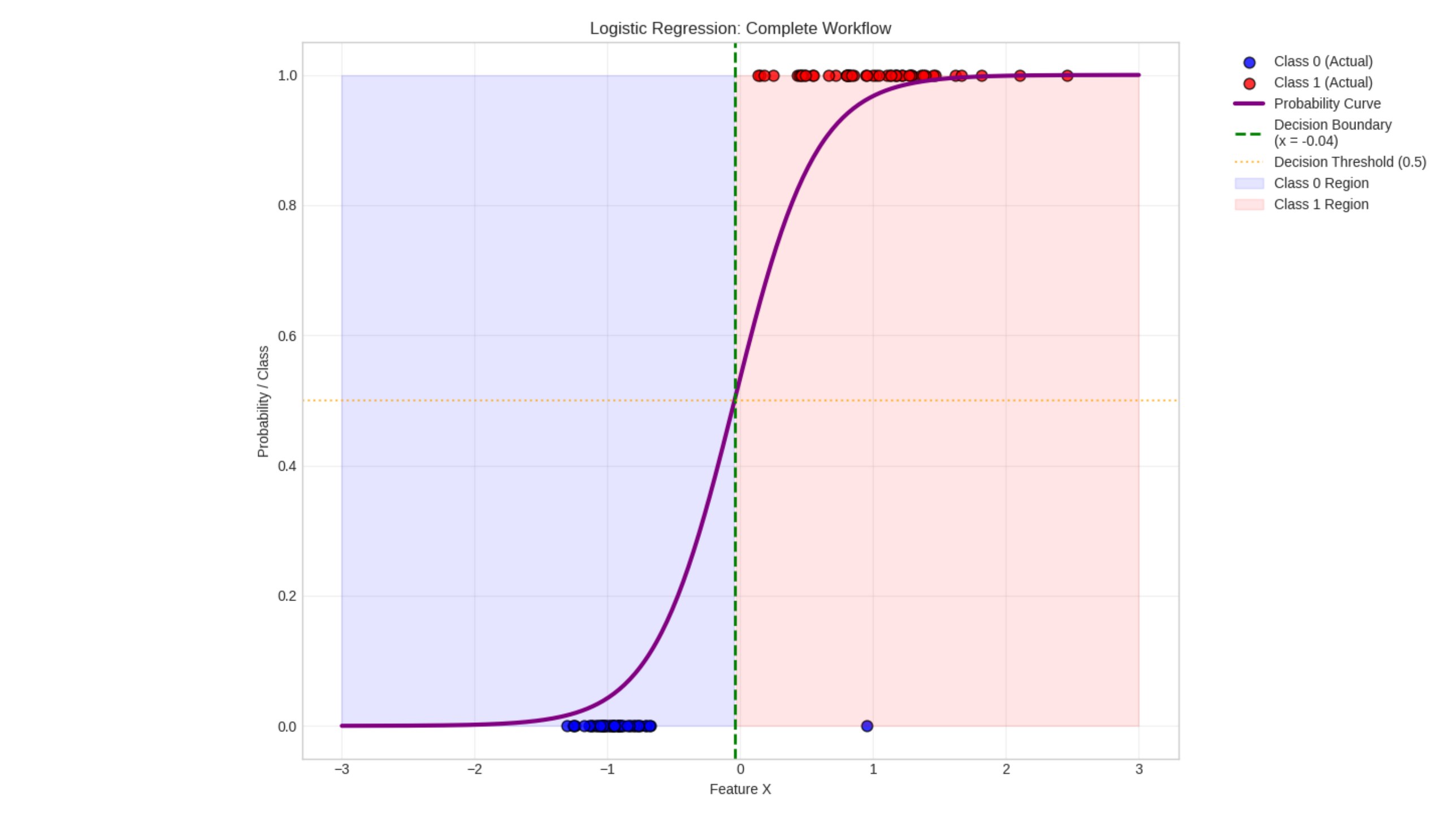
- Produces well-calibrated probabilities, not just binary class predictions.
- Highly interpretable due to its linear nature.

Decision Trees (CART)

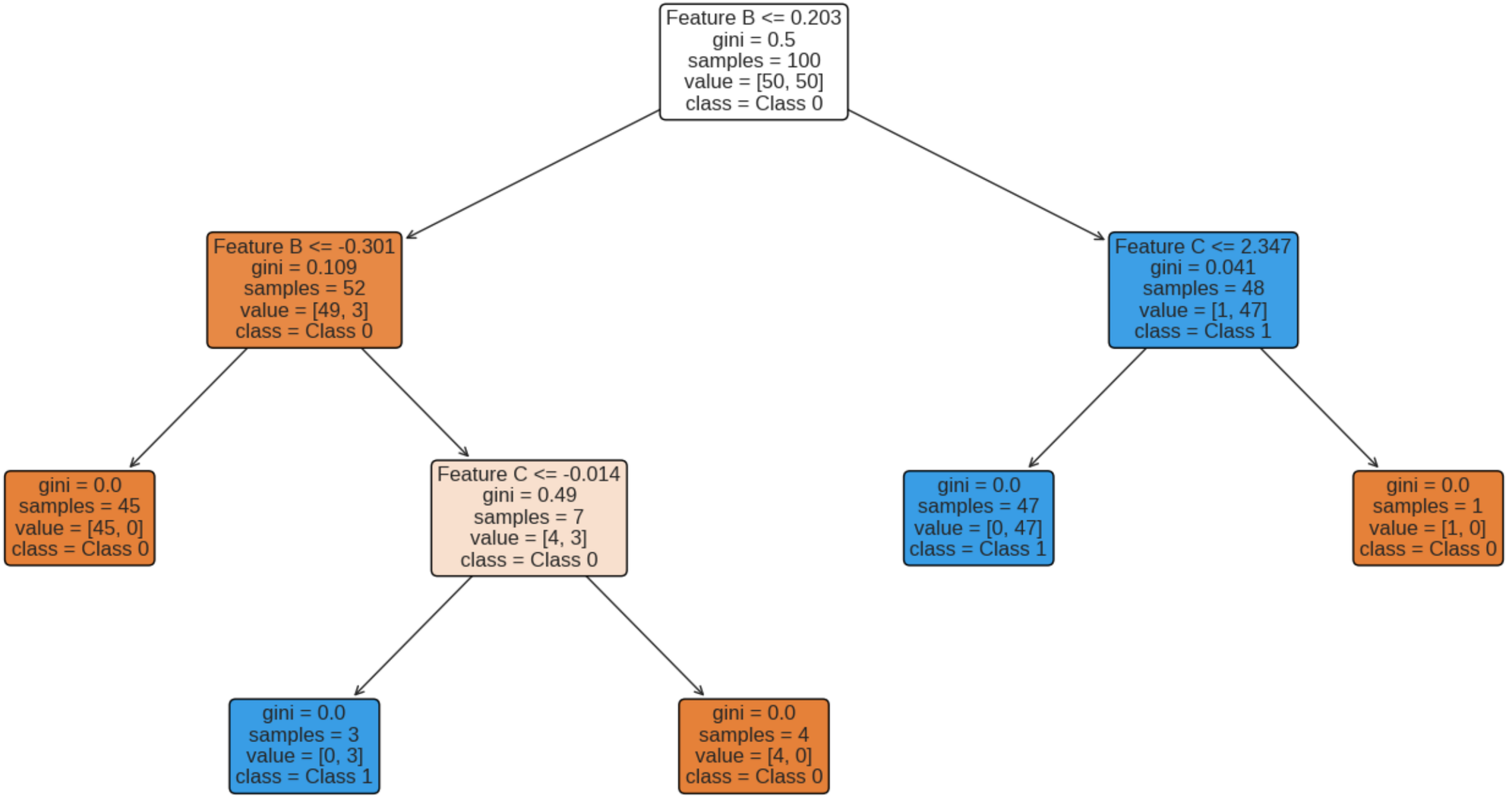
Intuition: A hierarchical, non-linear model that recursively partitions the feature space into simple, axis-parallel regions.

Splitting Criterion: Nodes are split using metrics like Gini Impurity or Information Gain (Entropy) to maximise the separability of classes.

- Highly interpretable and easy to visualise.
- Prone to overfitting if not constrained (e.g., by pruning or limiting maximum depth).


Logistic Regression: Complete Workflow
Probability / Class
Feature X
1.0
0.8
0.6
0.4
0.2
0.0
−3
−2
−1
0
1
2
3
Class 0 (Actual)
Class 1 (Actual)
Probability Curve
Decision Boundary
(x = -0.04)
Decision Threshold (0.5)
Class 0 Region
Class 1 Region

Detailed Decision Tree Visualization



Model Evaluation: The Confusion Matrix

The Foundation for Performance Metrics in Binary Classification

Actual Negative (0)	True Negative (TN)	False Positive (FP)
Actual Positive (1)	False Negative (FN)	True Positive (TP)

True Components (Correct)

- TP: Correctly predicted positive class.
- TN: Correctly predicted negative class.

False Components (Errors)

- FP: **Type I Error**. Negative instance incorrectly predicted as positive.
- FN: **Type II Error**. Positive instance incorrectly predicted as negative.

Extension to Multiclass: An `n x n` matrix where the diagonal elements represent all correct classifications across all classes.

Metrics I: Accuracy, Precision, and Recall

Moving Beyond Simple Correctness in Model Assessment

Accuracy

The overall fraction of correct predictions:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Pitfall: Highly misleading under severe **class imbalance** (e.g., a 99% baseline accuracy is often trivial).

Precision

Measures the quality of positive predictions: "Of all items predicted positive, how many were truly positive?"

$$\text{Precision} = \frac{TP}{TP + FP}$$

Focus: Minimising False Positives. Critical when the cost of FP is high (e.g., criminal conviction or spam filtering).

Recall (Sensitivity)

Measures the model's ability to find all positive samples: "Of all true positive items, how many did we catch?"

$$\text{Recall} = \frac{TP}{TP + FN}$$

Focus: Minimising False Negatives. Critical when the cost of FN is high (e.g., medical diagnosis for a serious disease).

Metrics II: The F1 Score and Multiclass

The F1 Score: Harmonizing Precision and Recall

The **harmonic mean** of Precision and Recall. It provides a single metric that simultaneously balances both concerns (the trade-off between FP and FN). The F1 score is generally more informative than accuracy, especially for evaluating models on imbalanced datasets.

$$\text{F1 Score} = 2 \frac{\text{Precision} \quad \text{Recall}}{\text{Precision} + \text{Recall}}$$

Metrics for Multiclass Classification

When evaluating a multiclass model, performance metrics are typically aggregated using two primary strategies:

- **Macro-Averaging:** Calculates the metric (e.g., Precision, Recall, F1) independently for each class, then takes the unweighted average. This treats all classes equally, regardless of sample size.
- **Micro-Averaging:** Aggregates the total counts of TPs, FPs, and FNs across all classes to compute the metric globally. This approach is heavily influenced by the majority class and is equivalent to overall accuracy.

Diagnostics: Learning Curves

A Visual Tool for Diagnosing Bias and Variance

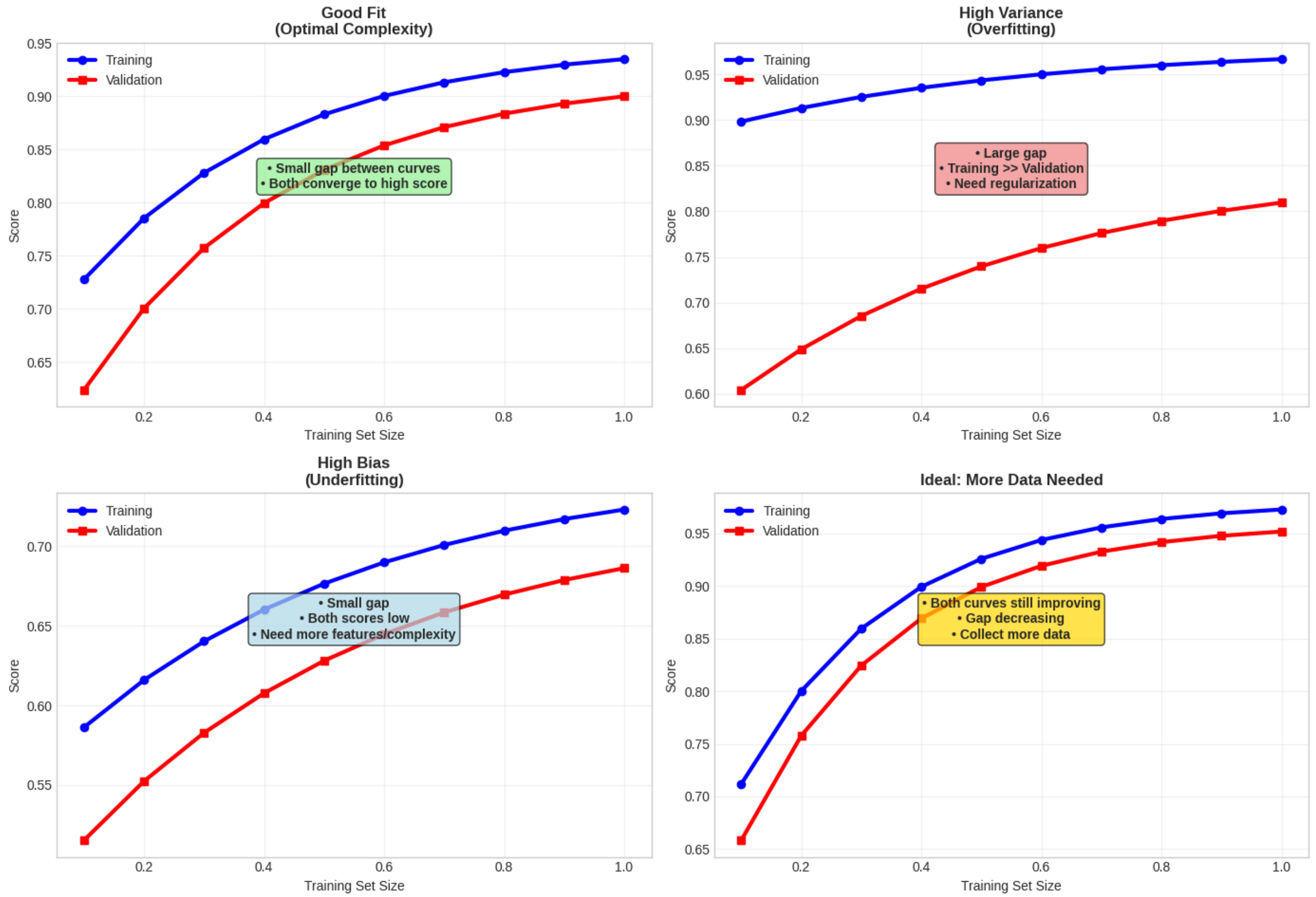
High Bias (Underfitting)

Both training and validation error curves converge to a high, unsatisfactory plateau. The model is too simple and has not learned the underlying relationships.
Remedy: Use a more complex model or add more features.

High Variance (Overfitting)

There is a significant gap between the low training error and the high validation error. The model has fit the noise in the training data too closely.
Remedy: Add more training data, use regularisation, or simplify the model.

Validation Curves: A complementary diagnostic plot that shows model performance as a function of a specific hyperparameter (e.g., `k` in k-NN) to help identify the optimal setting before the final test phase.


Learning Curve Diagnostics: Common Scenarios
Good Fit
(Optimal Complexity)
Training
Validation
• Small gap between curves
• Both converge to high score
Score
Training Set Size
High Variance
(Overfitting)
Training
Validation
• Large gap
• Training >> Validation
• Need regularization
Score
Training Set Size
High Bias
(Underfitting)
Training
Validation
• Small gap
• Both scores low
• Need more features/complexity
Score
Training Set Size
Ideal: More Data Needed
Training
Validation
• Both curves still improving
• Gap decreasing
• Collect more data
Score
Training Set Size