Classification II: Advanced Algorithms and In-Depth Evaluation

Audience: Graduate students and practitioners in machine learning.

This lecture progresses beyond foundational classifiers to explore high-performance ensemble methods and provide a rigorous treatment of evaluation metrics for nuanced model assessment. We will delineate the theoretical underpinnings of Support Vector Machines, Random Forests, and Gradient Boosting Machines.

Key Concepts

Max-Margin • Ensemble Learning • Bagging • Boosting • ROC-AUC • PR-AUC • Calibration

Support Vector Machines (SVM): The Max-Margin Principle

Theoretical Foundation and the Kernel Trick

Core Objective: Max-Margin Hyperplane

Find the hyperplane that maximises the margin—the distance from the decision boundary to the nearest data points, known as **Support Vectors**.

Hard-Margin SVM (Linearly Separable)

Assumes data is perfectly separable. Optimisation seeks to minimise weight vector magnitude $\left(\frac{1}{2}||\mathbf{w}||^2\right)$ subject to correct classification of all points:

Soft-Margin SVM (Non-Separable Data)

Introduces slack variables (ξ_i) and a penalty parameter (C) to allow for classification errors, balancing margin size against misclassification cost.

The Kernel Trick

Maps input data to a higher-dimensional feature space where it becomes linearly separable, avoiding expensive, explicit calculation of the high-dimensional coordinates.

Linear Kernel

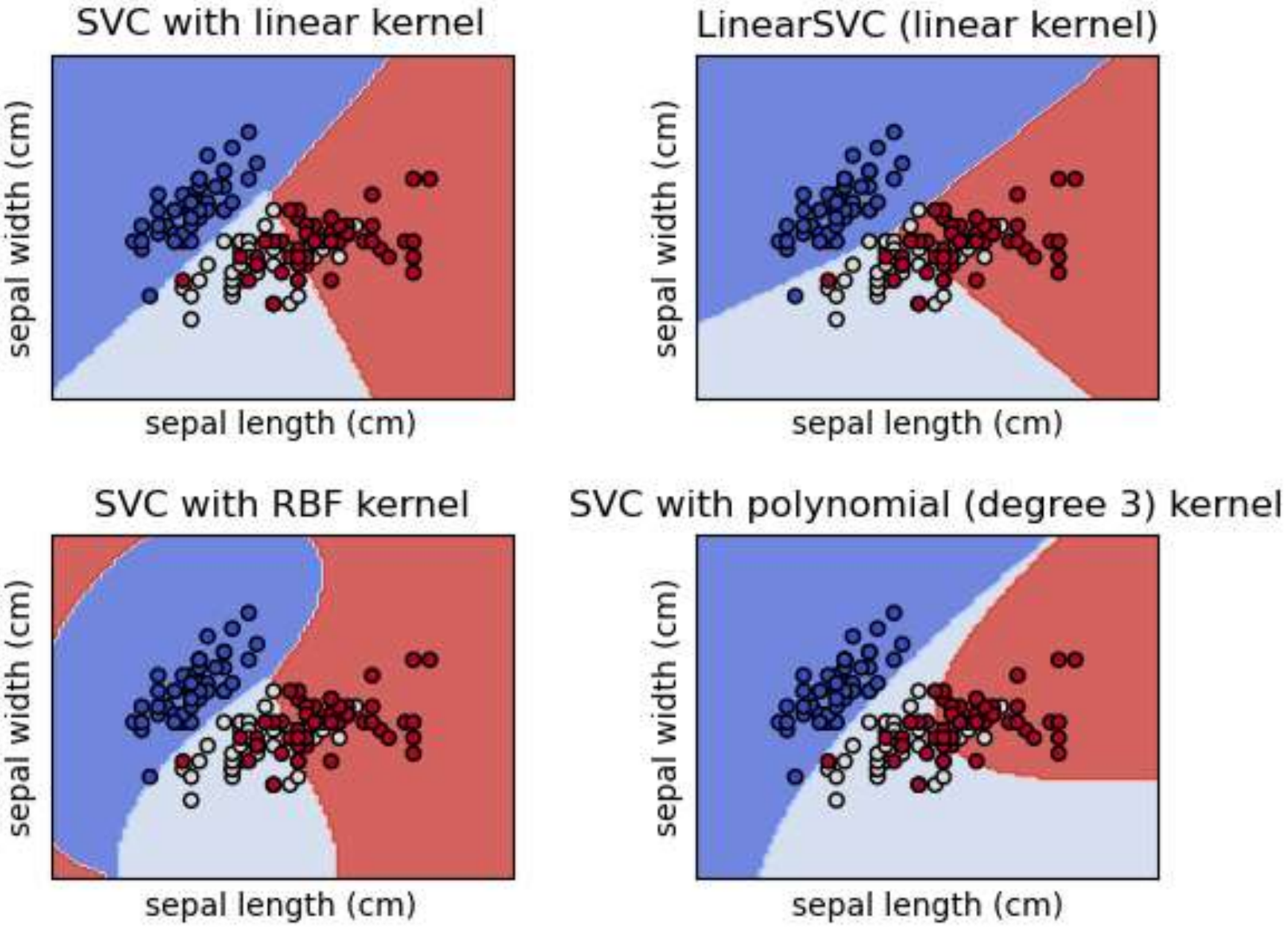
$$K(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{x_i}^{\top}\mathbf{x_j}$$

Polynomial Kernel

$$K(\mathbf{x_i}, \mathbf{x_j}) = (\gamma \mathbf{x_i}^{\top}\mathbf{x_j} + r)$$

RBF (Gaussian) Kernel

$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma ||\mathbf{x_i} - \mathbf{x_j}||)$$


SVC with linear kernel
LinearSVC (linear kernel)
SVC with RBF kernel
SVC with polynomial (degree 3) kernel
sepal width (cm)
sepal length (cm)

Ensemble Methods I: Random Forests (Bagging)

Harnessing the Wisdom of Many to Reduce Variance

1

Bootstrap Aggregating (Bagging)

Create multiple subsets of the training data by sampling **with replacement**. Train an independent base estimator (e.g., Decision Tree) on each bootstrap sample.

2

Prediction Aggregation

The final output is determined by averaging the predictions (for regression) or applying a majority vote (for classification). **Primary Effect: Reduction in Variance.**

1

Random Forests Extension

Random Forests combine bagging with an additional layer of randomness to decorrelate the trees further.

2

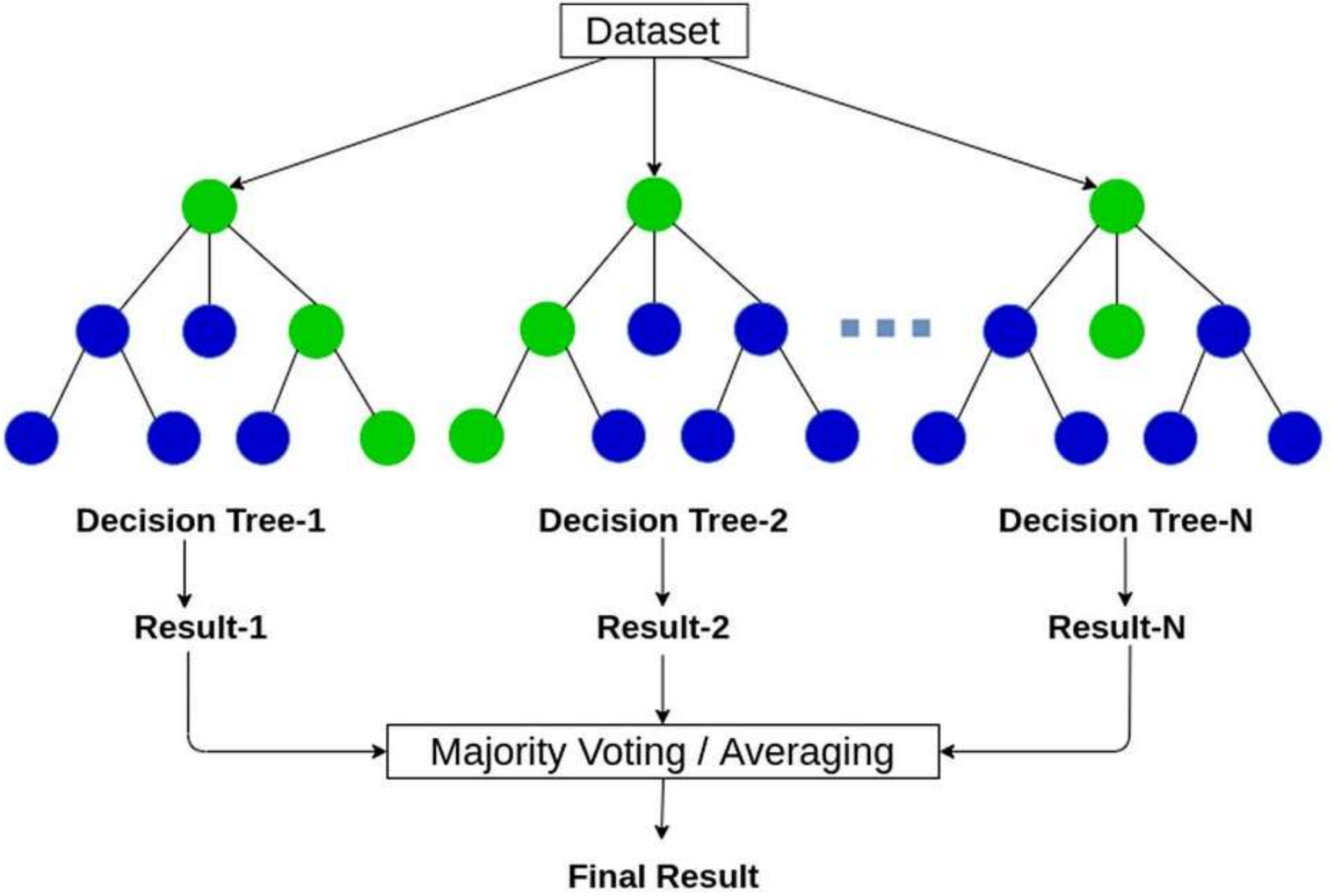
Feature Subsampling

When splitting a node in any individual tree, only a random subset of features is considered (e.g., $\sqrt{p}$ features for classification).

3

Robustness and Importance

The resulting ensemble is highly robust and less sensitive to noise. It naturally provides **Feature Importance** based on the mean decrease in impurity across all trees.


Dataset
Decision Tree-1
Decision Tree-2
Decision Tree-N
Result-1
Result-2
Result-N
Majority Voting / Averaging
Final Result

Ensemble Methods II: Gradient Boosting (Boosting)

Sequential Error Correction to Reduce Bias

01

Initial Model & Residuals

Start with a simple model (e.g., predicting the mean). Compute the negative gradient of the loss function, which are the **residuals** (errors) for each data instance.

02

Sequential Weak Learner Training

Train a new, shallow decision tree (a "weak learner") specifically to predict and correct these residuals.

03

Model Update and Learning Rate

Add the new learner's scaled predictions (controlled by a learning rate *eta*) to the existing ensemble, gradually refining the overall prediction.

04

Iteration for Bias Reduction

Repeat the process for a fixed number of iterations. **Primary Effect: Significant Reduction in Bias**, leading to high predictive accuracy.

Modern Implementations (State-of-the-Art)

XGBoost

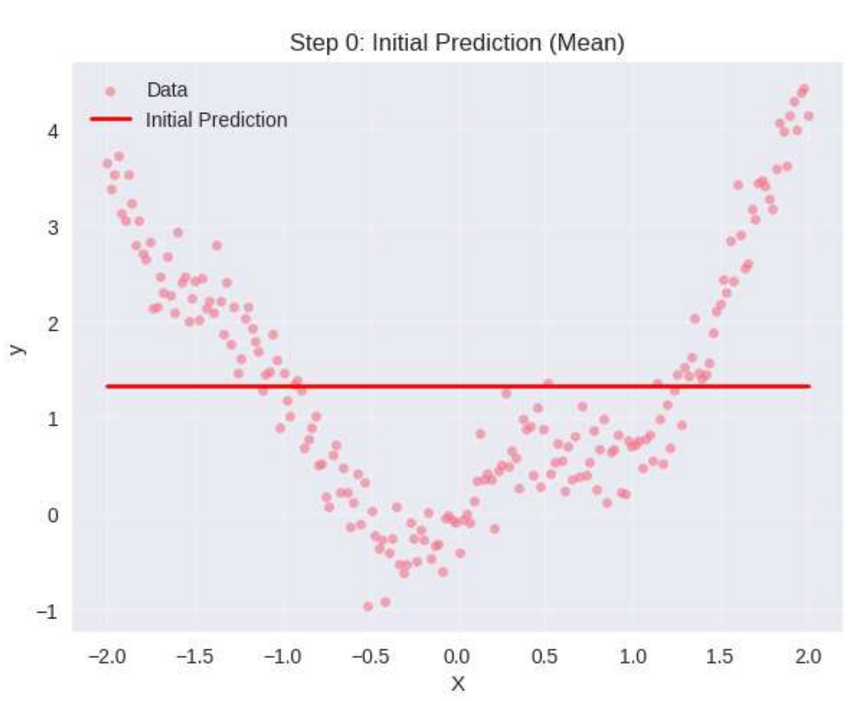
Optimised for performance; incorporates L1/L2 regularization to prevent overfitting.

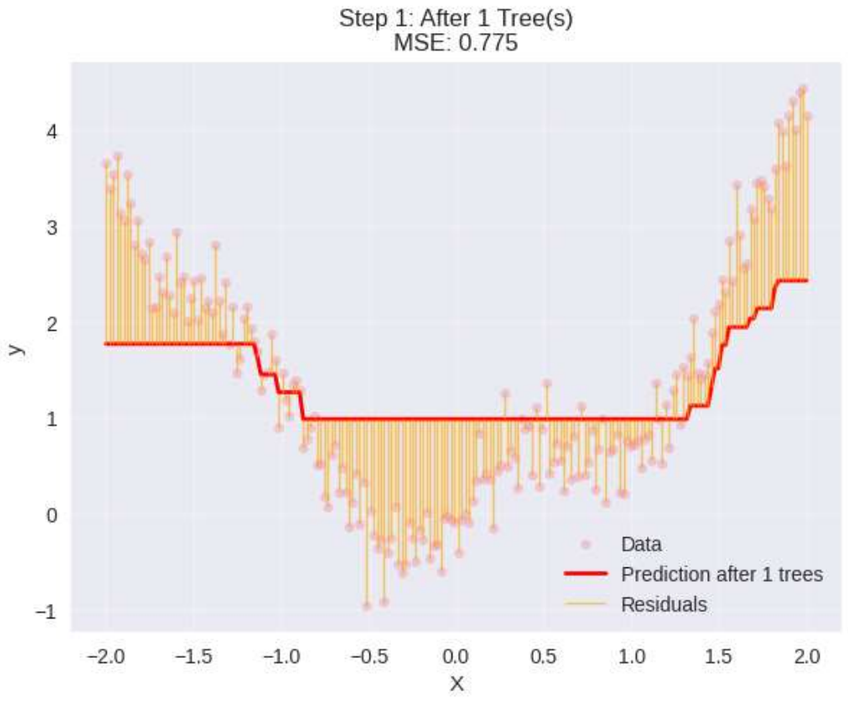
LightGBM

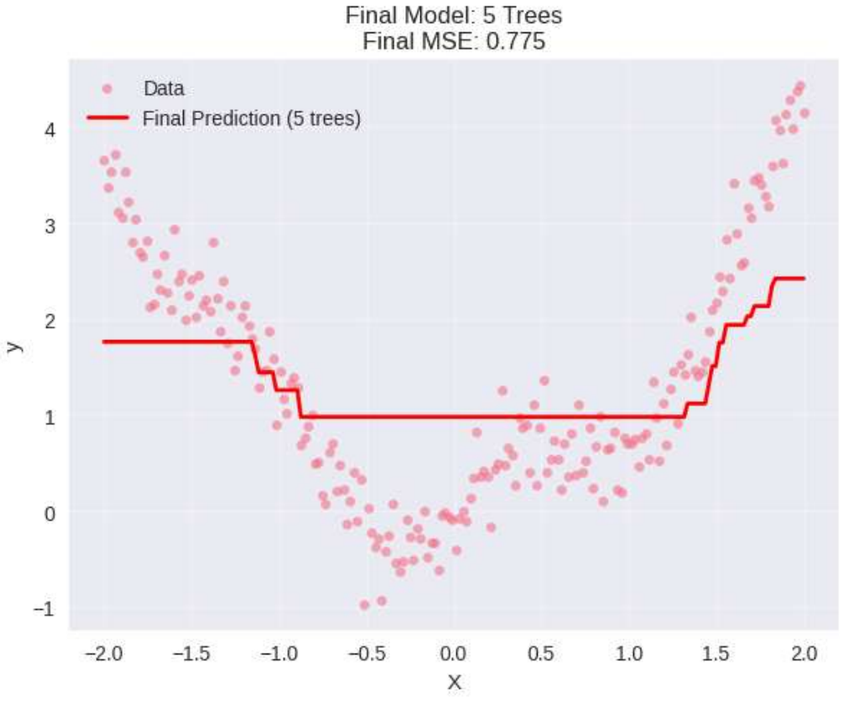
Uses histogram-based algorithms, resulting in faster training times and lower memory consumption.

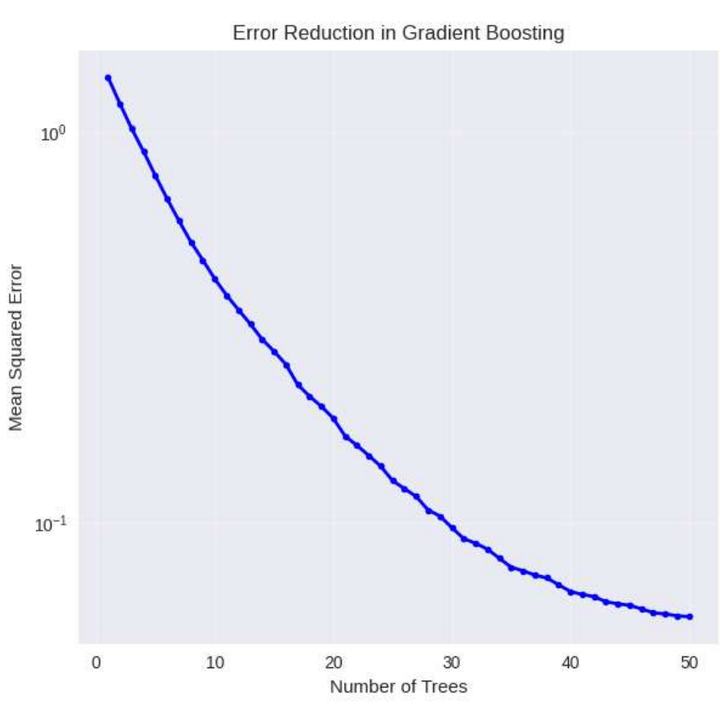
CatBoost

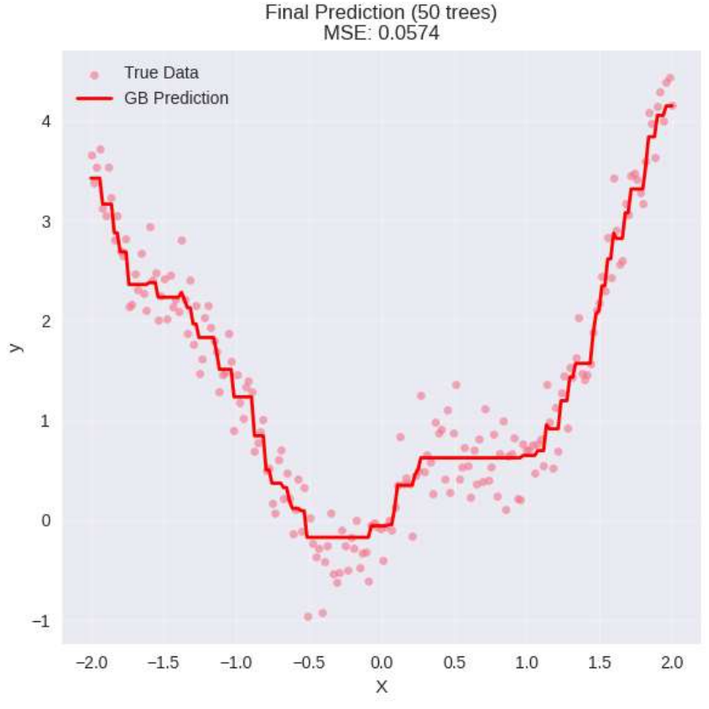
Designed to handle categorical features natively using a permutation-aware method; reduces target leakage.


Step 0: Initial Prediction (Mean)
Data
Initial Prediction
X
y


Step 1: After 1 Tree(s)
MSE: 0.775
Data
Prediction after 1 trees
Residuals
X
y


Final Model: 5 Trees
Final MSE: 0.775
Data
Final Prediction (5 trees)
X
y


Error Reduction in Gradient Boosting
Mean Squared Error
Number of Trees


Final Prediction (50 trees)
MSE: 0.0574
True Data
GB Prediction
X
y

Algorithmic Summary & Comparative Analysis

A high-level comparison of the core mechanisms, effects, and typical use-cases for the advanced classification algorithms discussed.

SVM	Max-margin optimisation with kernels	Low Variance, High-Dimensional Capability	`C, kernel, gamma`	Strong theoretical foundation; highly effective in high-dimensional feature spaces.
Random Forest	Bagging + random feature subsetting	Variance Reduction	`n_estimators, max_features`	Robust to noise, parallelisable for efficiency, provides inherent feature importance scores.
Gradient Boosting	Sequential fitting of residuals (negative gradient)	Bias Reduction	`n_estimators, learning_rate, max_depth`	Achieves state-of-the-art predictive accuracy across diverse datasets; highly flexible.

Evaluation II: Precision-Recall Curve (PR-AUC)

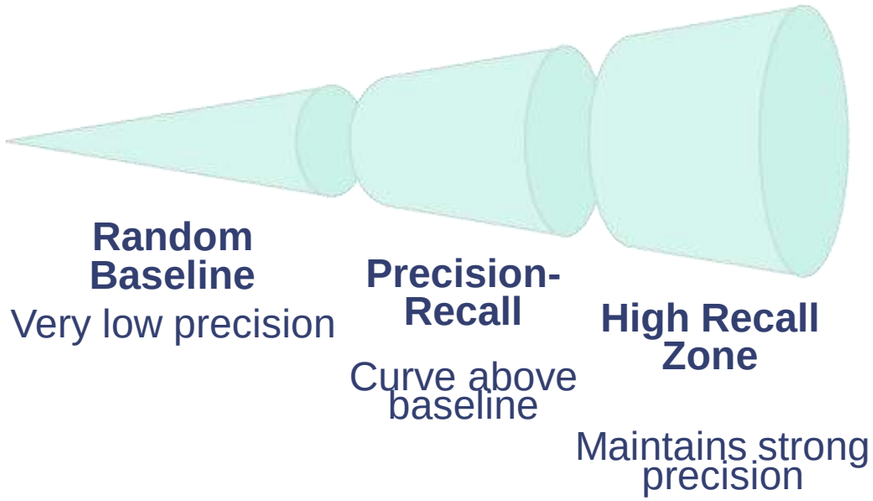
The Focus on the Positive Class in Imbalanced Data

Precision-Recall Curve

Plots **Precision** (positive predictive value) against **Recall** (True Positive Rate/Sensitivity) across all possible thresholds.

- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$

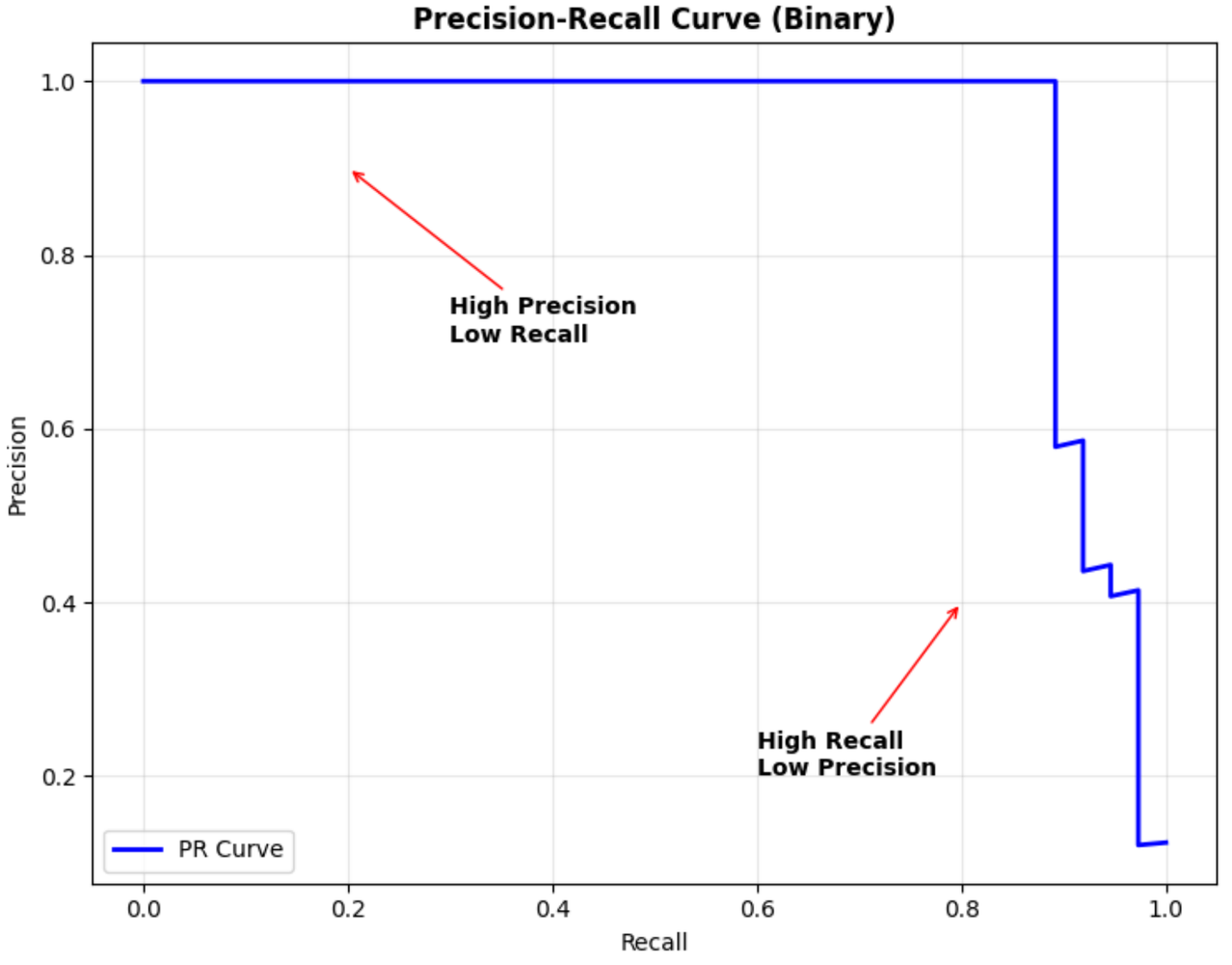
AUC-PR provides a performance summary focused on the ability to correctly identify the minority positive class without generating too many false alarms.

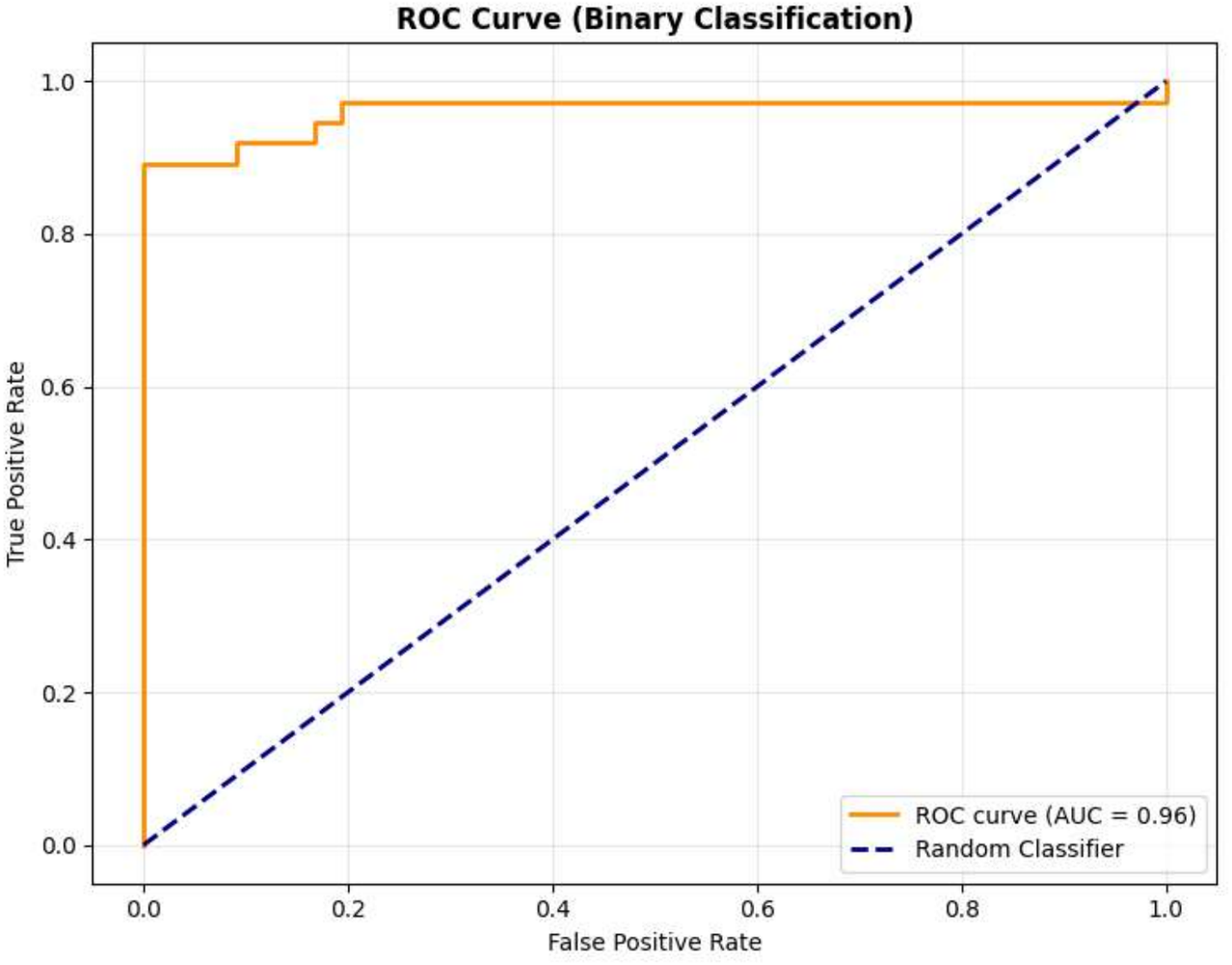


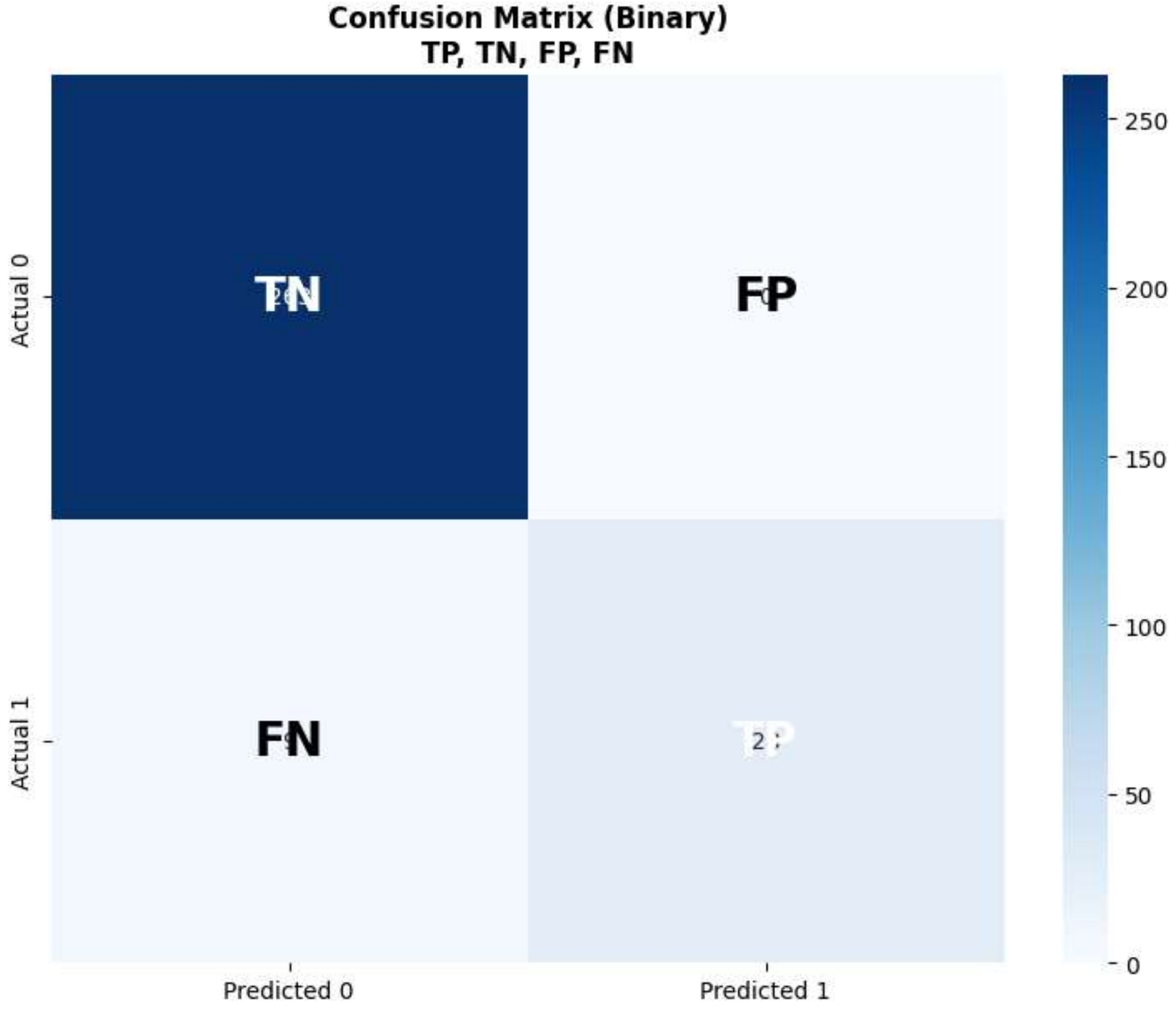
Why PR is Superior to ROC for Imbalance

> On severely imbalanced datasets, the ROC curve can give an overly optimistic view of performance. This is because the large number of **True Negatives** (TN) makes the False Positive Rate (FPR) artificially low, even if the model performs poorly on the minority positive class.
>
> The PR curve, which does not use True Negatives in its calculation, is therefore a more realistic measure of performance when the positive class is rare.

Primary Use Case: The PR curve is the primary choice for imbalanced datasets and whenever the cost of False Positives is high (e.g., unnecessary medical intervention).


Precision-Recall Curve (Binary)
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0
High Precision
Low Recall
High Recall
Low Precision
PR Curve


ROC Curve (Binary Classification)
True Positive Rate
False Positive Rate
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
ROC curve (AUC = 0.96)
Random Classifier


Confusion Matrix (Binary)
TP, TN, FP, FN
TN
FP
FN
TP
Actual 0
Actual 1
Predicted 0
Predicted 1
250
200
150
100
50
0

Evaluation III: Metrics for Imbalanced & Multi-Class Problems

Matthew's Correlation Coefficient (MCC)

A single balanced measure for binary classification that utilises all four confusion matrix quadrants (TP, TN, FP, FN).

Value range: +1 (perfect), 0 (random), -1 (total disagreement). **Superior to F1** as it accounts for disparity in class sizes.

Cohen's Kappa (κ)

Measures the agreement between the model's predictions and the true labels, correcting for the agreement expected purely by chance.

Highly useful for **multi-class assessment**. Formula: $\kappa = \frac{p_o - p_e}{1 - p_e}$, where p_o is observed agreement (accuracy) and p_e is expected agreement.

Multi-Class Averaging Strategies

- **Macro-Average**

  Calculates the metric (e.g., Precision) for each class independently and then takes the unweighted average. **Treats all classes equally**.

- **Micro-Average**

  Aggregates the total TP, TN, FP, and FN across all classes and then computes the metric. **Favours larger classes** (equivalent to overall accuracy).

- **Weighted-Average**

  Averages the per-class metrics but weighted by the number of instances (support) in each class. Provides a balance between the two above.

Evaluation IV: Probabilistic Assessment - Log Loss

Penalizing Overconfidence in Wrong Predictions

Log Loss (Cross-Entropy Loss)

Log Loss measures the quality of a model's **probabilistic predictions** rather than just the final hard-class assignments.

It is a measure of the "unexpectedness" of the true labels given the model's predicted probabilities.

For binary classification (p_i is predicted probability, y_i is true label):

$$\text{Log Loss} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)]$$

1

High Penalty

Heavily penalises confident but incorrect predictions (e.g., predicting **P=0.99** when the true label is **Y=0**).

2

Reward Calibration

Rewards models that produce **well-calibrated probabilities** (i.e., when the model predicts 0.8, the instance is correct 80% of the time).

3

Objective

A lower Log Loss is better, with **0** representing a perfect probabilistic model.

> Log Loss is essential for evaluating models like Logistic Regression or Neural Networks where probability outputs are crucial. It is frequently used as the primary training objective function for many advanced algorithms, including Gradient Boosting.