Wine Quality Classification: Logistic Regression and SVM with Kernel Extensions

October 17, 2025

Abstract

I predicted wine quality (binary: good if quality ≥ 6) on the combined red+white UCI Wine Quality data. All models are implemented **from scratch** (no scikit-learn estimators) and evaluated under a leakage-safe protocol: deduplication, Z-Score diagnostics with IQR winsorization (fit on train only), training-only standardization, stratified train/test split, and 5-fold cross-validation for hyperparameter tuning. On the holdout test set, **Kernel Logistic Regression (RBF)** achieves the best F1 (**0.807**) with Accuracy 0.748, Precision 0.783, Recall 0.833, closely followed by a strong **Logistic Regression** baseline (F1 = 0.800). The **SVM-RBF** model attains very high recall (0.954) with lower precision, representing a different operating point that may be preferable if recall is prioritized.

1 Introduction and Objectives

The goal is to compare **Logistic Regression (LR)** and **Support Vector Machines (SVM)**—both linear and kernelized—for predicting wine quality. I emphasized a **sound methodology**: (i) prevent test-set leakage, (ii) use cross-validation for hyperparameter tuning, (iii) handle outliers robustly without discarding data, and (iv) **exclude the `type` (red/white) variable from modeling** to avoid shortcut learning (kept for EDA only).

2 Dataset and Target Definition

I merged the red and white wine datasets into a single frame of **6,497 rows** and **14 columns**: `[fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol, quality, type, target]`. The binary target is **1** for quality ≥ 6 and **0** otherwise. There were **no missing values** and **1,177 exact duplicates**, which I dropped prior to splitting. Class balance and wine-type counts are shown in Fig. 1 and Fig. 2.

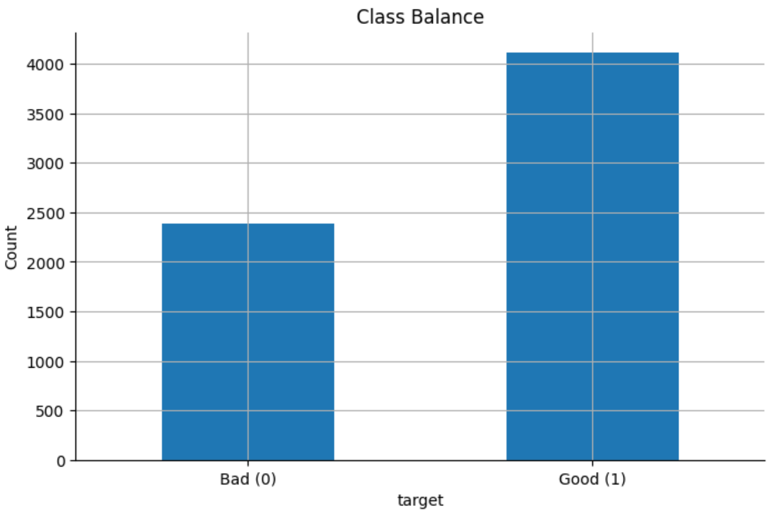


Figure 1: Class balance on the combined dataset (Bad vs Good).

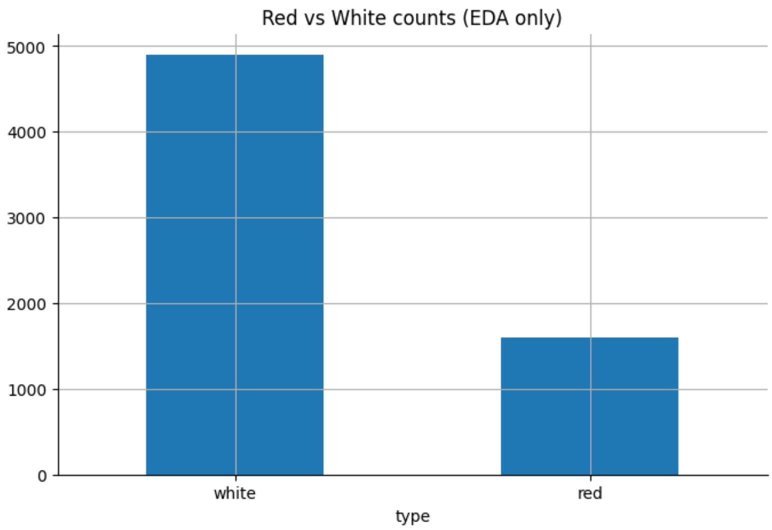


Figure 2: Counts of white vs. red wines (EDA only; `type` is excluded from modeling).

To understand marginal distributions and dependencies, Fig. 3 shows feature histograms and Fig. 4 shows the correlation heatmap. Note: alcohol positively correlates with `quality`; volatile acidity correlates negatively—consistent with oenological expectations.

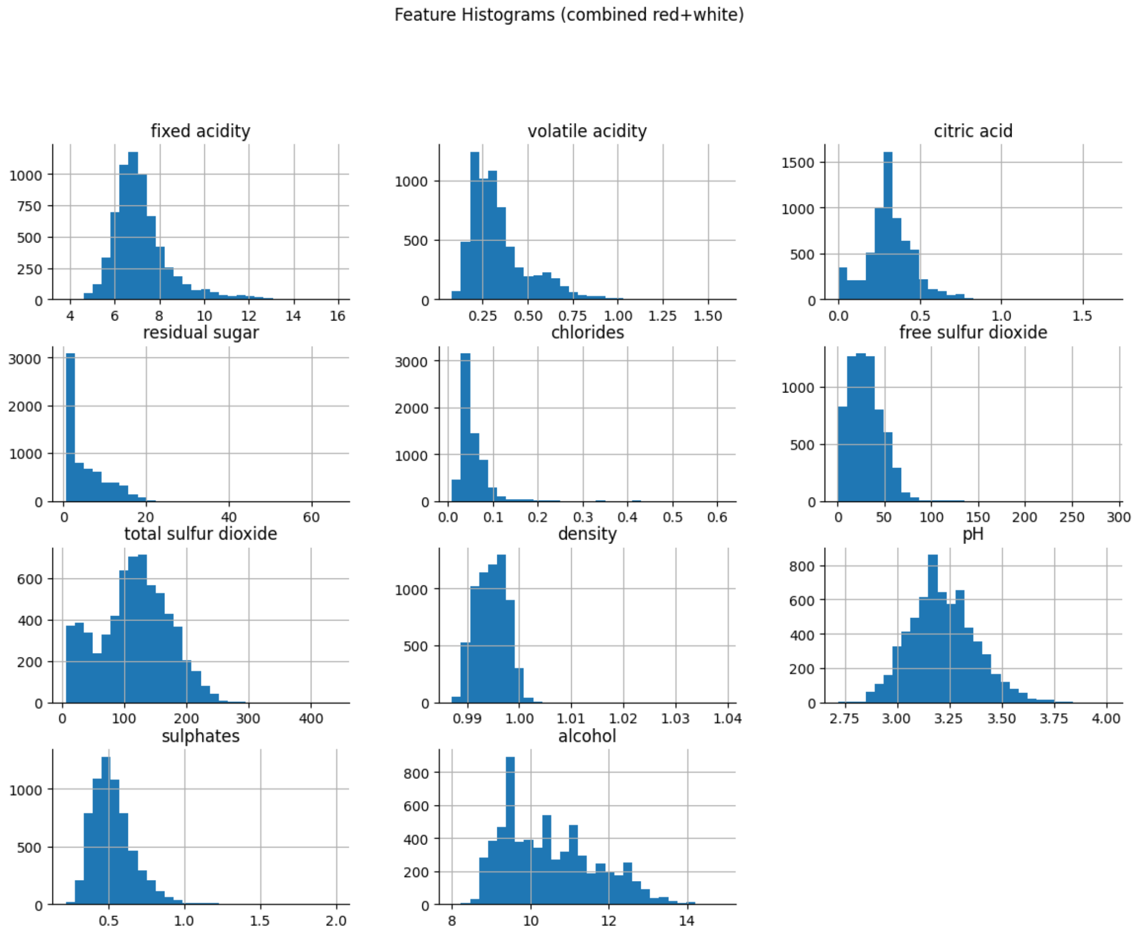


Figure 3: Feature histograms (combined red+white). Heavy tails in several variables motivate winsorization.

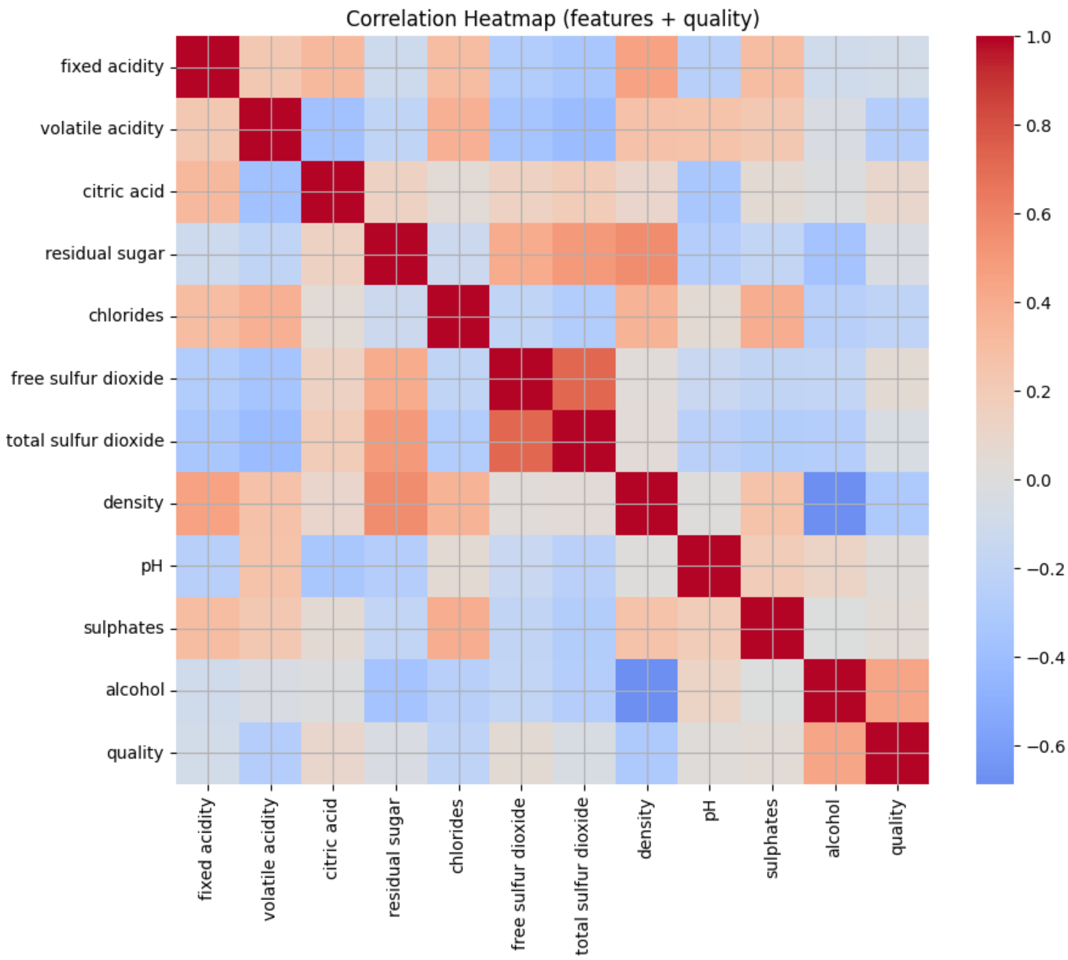


Figure 4: Correlation heatmap (features + quality). Alcohol correlates positively with quality; volatile acidity is negatively associated.

For completeness, Fig. 5 visualizes feature distributions by `type`. This is *EDA only*; `type` is not used in any model.

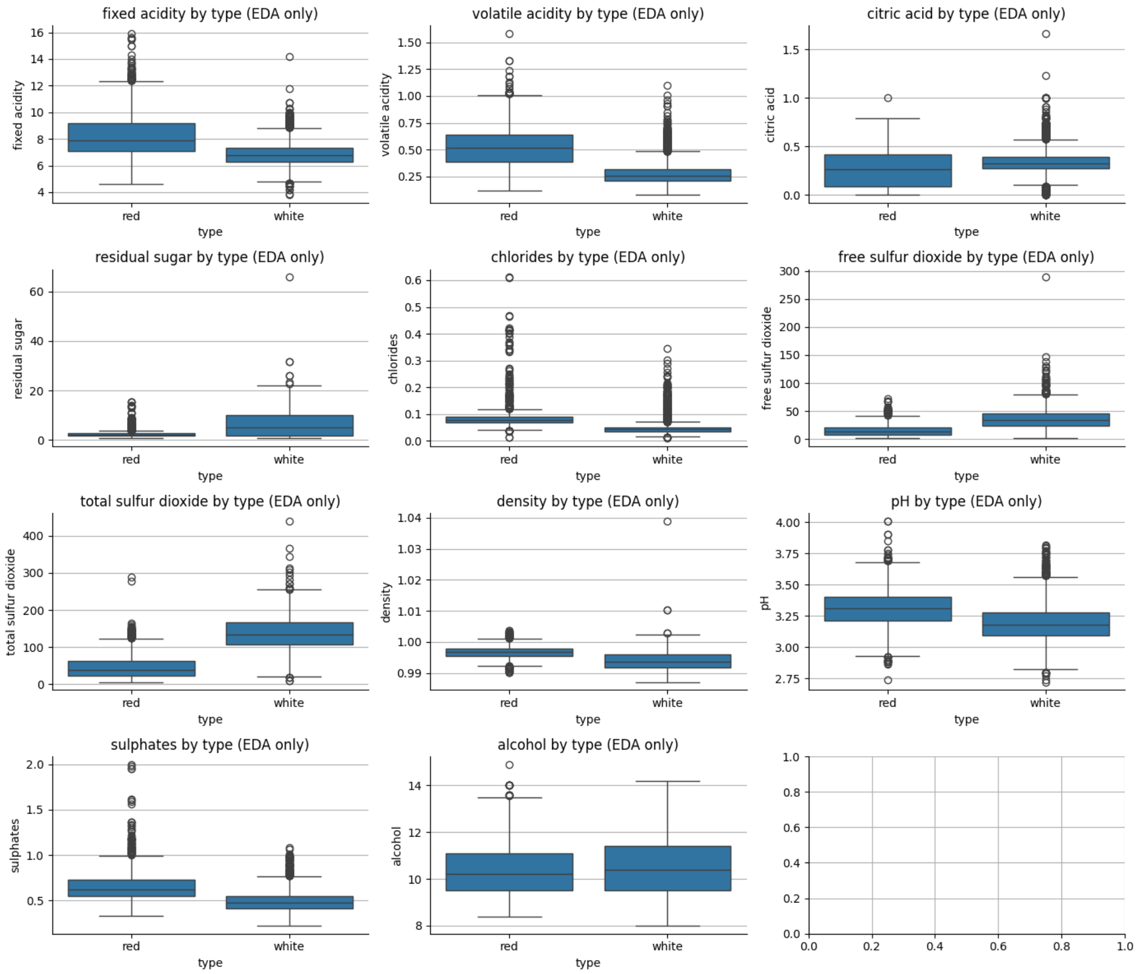


Figure 5: Feature distributions by `type` (EDA only).

3 Preprocessing and Leakage Control

3.1 Why Winsorization (and not deletion)?

Z-Score screening shows non-negligible tails (e.g., fixed acidity $\sim$1.97% with $|Z| > 3$). Deleting outliers would reduce data and distort distributions. I instead perform **IQR winsorization**, capping values outside $[Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR]$. Caps are **learned on the training set only** and then applied to validation/test, preventing leakage. Fig. 6 and Fig. 7 show the effect.

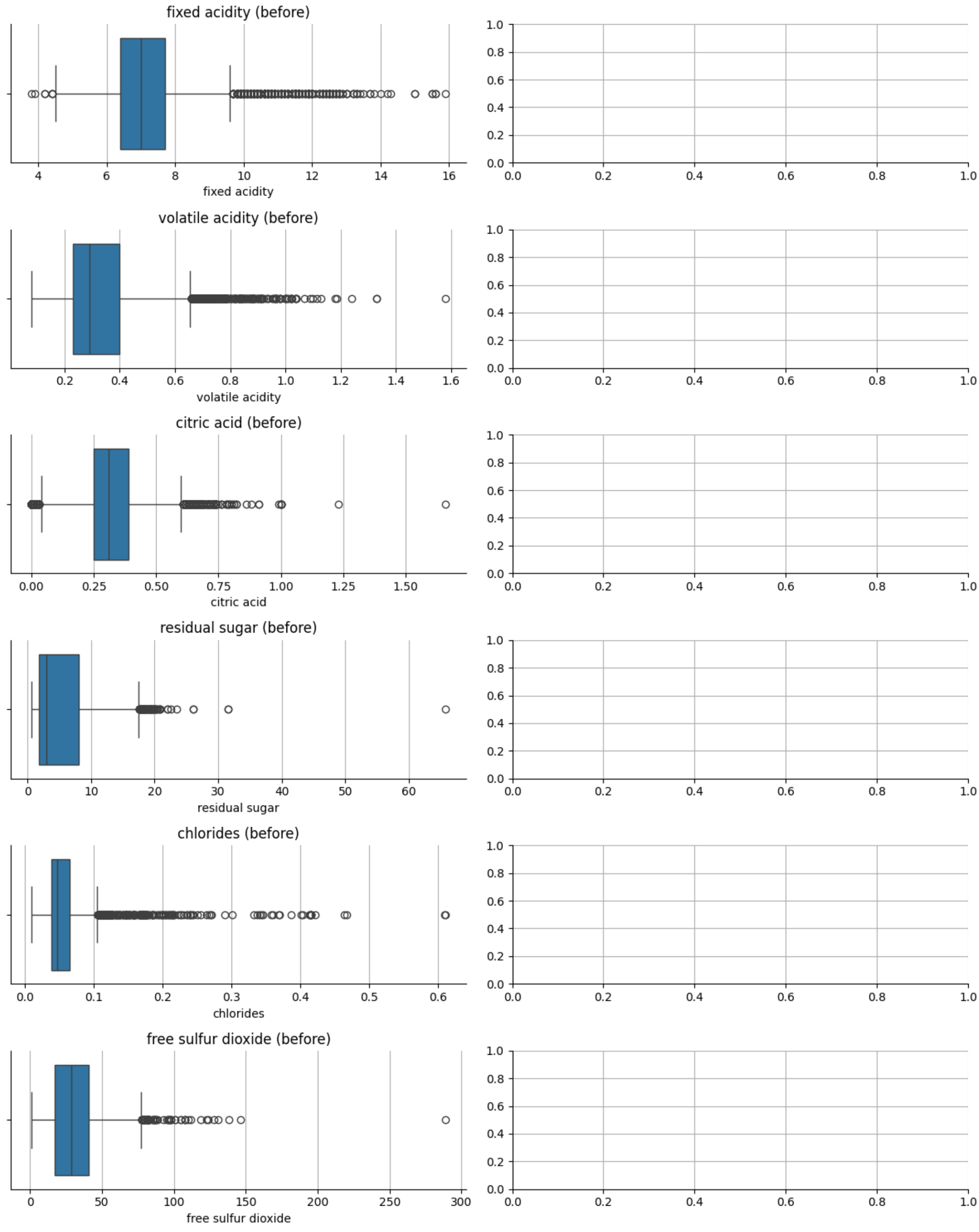


Figure 6: Diagnostic boxplots *before* IQR winsorization. Multiple features exhibit long right tails.

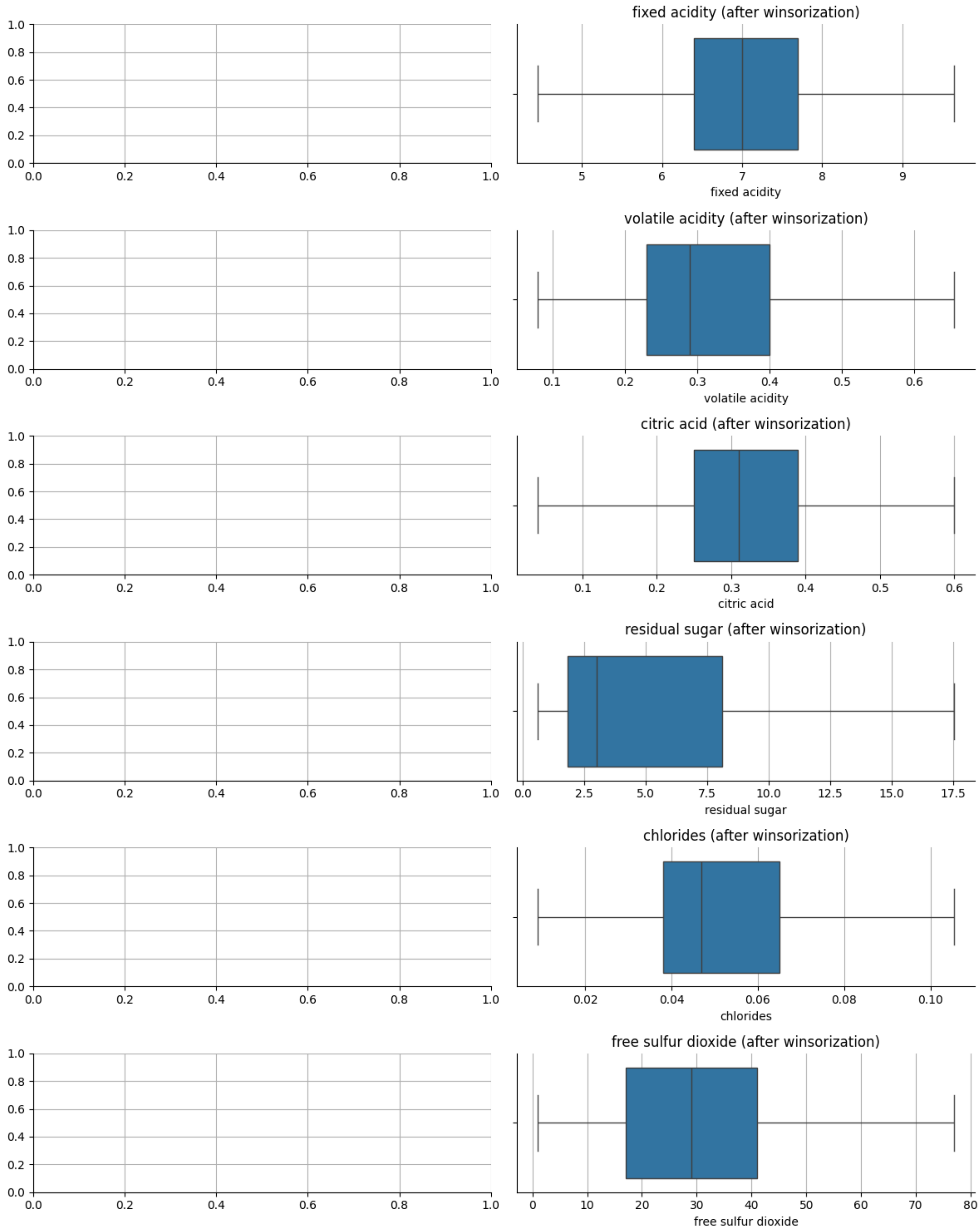


Figure 7: Boxplots *after* winsorization. Caps mitigate leverage of extreme values while preserving samples.

3.2 Standardization and Split Strategy

I **standardized** each feature using mean/SD from the **training data** only; the same transform is applied to CV folds and test data. I performed a **stratified** train/test split to preserve the class ratio. All cross-validation is done within the training split.

4 Models Implemented from Scratch

4.1 Logistic Regression (Linear)

I minimize the L2-regularized logistic loss in the *primal*, trained via (stochastic) gradient descent with a step-decay schedule. Prediction: $\hat{p}(x) = \sigma(w^\top x + b)$; by default used a 0.5 threshold.

4.2 Kernel Logistic Regression (KLR)

I use a **dual** representation with coefficients α and kernel matrix $K(X, X)$: $\hat{p}(x) = \sigma(\sum_i \alpha_i K(x, X_i))$, with L2 regularization on α. I considered two kernels: (i) **RBF** $K(x, z) = \exp(-\gamma\|x - z\|^2)$ capturing smooth local structure; (ii) **Polynomial** (degree 2) $K(x, z) = (\alpha\, x^\top z + c)^2$ capturing pairwise interactions. I center/normalize K for numerical stability in the polynomial case.

4.3 Support Vector Machines

Linear SVM is trained via **Pegasos** (stochastic subgradient on hinge loss with L2 penalty λ). Kernel SVM applies the kernel trick in dual form with the same kernels as KLR. Predictions are $\mathrm{sign}\big(\sum_i \beta_i K(x, X_i) + b\big)$.

5 Model Selection and Metrics

5.1 Cross-Validation and Grids

I use **5-fold stratified CV** on the training split and select hyperparameters by **F1**, more informative under moderate imbalance. Illustrative grids:

- **LR:** $lr \in \{0.005, 0.01, 0.1\}$; $n\_epochs \in \{400, 600, 800\}$; $reg \in \{10^{-3}, 10^{-2}, 10^{-1}\}$.
- **KLR-Poly:** degree 2; $\alpha \approx 0.05$; $c = 1.0$; centered/normalized; $reg \in \{0.001, 0.01, 0.1\}$; $lr \in \{0.005, 0.01\}$; epochs $\in \{400, 800\}$.
- **KLR-RBF:** γ tuned; $reg/lr/epochs$ as above.
- **SVM-Linear:** $\lambda \in [10^{-4}, \dots]$; epochs/mini-batch sizes tuned.
- **SVM-Kernels:** analogous to KLR for γ (RBF) or (α, c) (Poly); λ tuned.

5.2 Why These Metrics?

Accuracy can be misleading with imbalance; **Precision** (purity of predicted positives) and **Recall** (coverage of true positives) provide complementary views; their harmonic mean **F1** summarizes the trade-off. **ROC AUC** measures ranking quality independent of threshold. I reported all four on the holdout test set; ROC curves appear in Fig. 8.

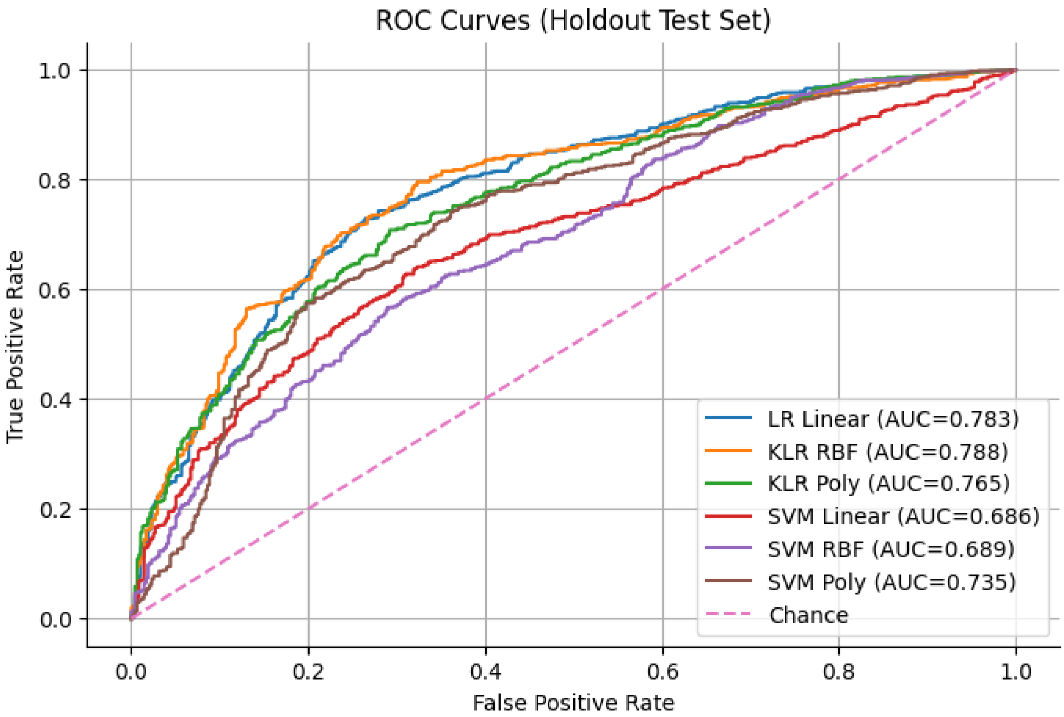
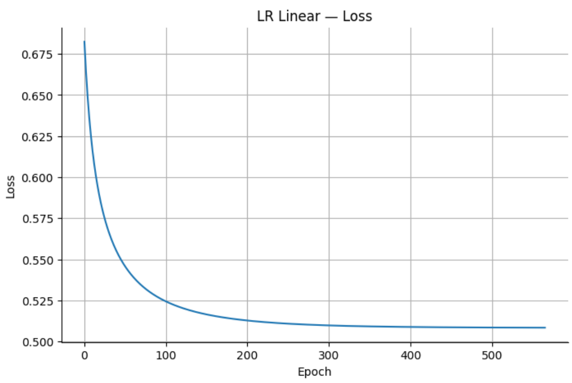


Figure 8: ROC curves on the holdout test set. KLR-RBF and LR are the top two by AUC.

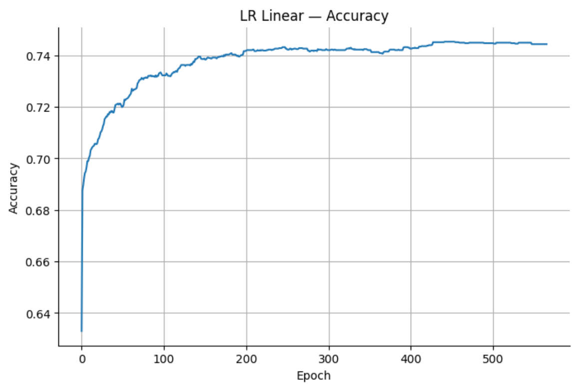
6 Results

6.1 Training Dynamics

Fig. 9 shows LR loss/accuracy converging smoothly, indicating stable optimization; Fig. 10 shows linear SVM margin/accuracy across epochs under Pegasos.

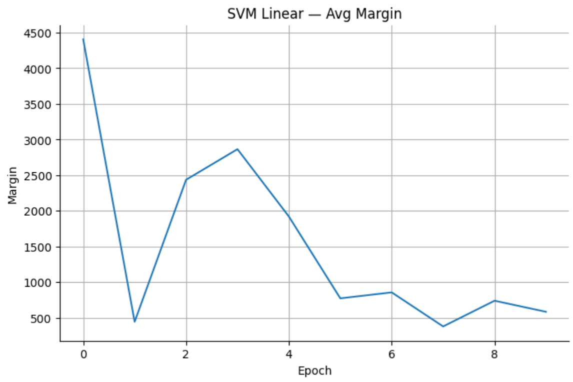


(a) LR loss vs. epochs

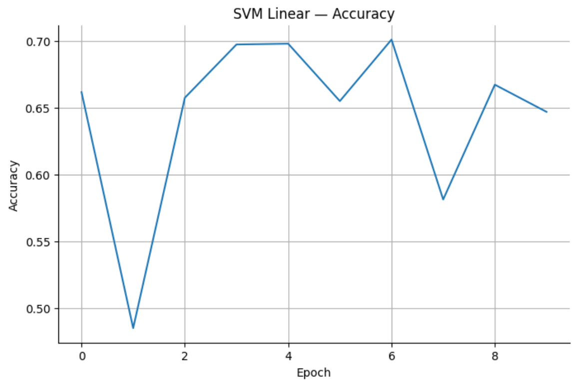


(b) LR accuracy vs. epochs

Figure 9: Convergence of Logistic Regression.



(a) Avg margin vs. epochs



(b) Accuracy vs. epochs

Figure 10: Dynamics of Linear SVM (Pegasos).

Learning Dynamics (per plot)

LR Loss vs Epochs.

- A steady, monotonic decrease indicates consistent progress minimizing the logistic objective.
- Typical shape: steep early drop (large gradients), then a shallow tail as the model approaches a minimum.
- Small oscillations with stochastic updates are fine as long as the overall trend decreases.
- If the curve plateaus high, learning rate may be too small or L2 too strong; divergence suggests learning rate too large.
- Continued loss decrease after accuracy plateaus often reflects improved probability calibration rather than better classification.

LR Accuracy vs Epochs.

- Accuracy should rise rapidly then saturate as the decision boundary stabilizes.
- If accuracy peaks then declines while training loss still falls, that's overfitting—increase regularization or use early stopping.
- High jitter suggests tiny batches or an aggressive step size; smooth via larger batches or decays.
- A flat curve from the start implies underfitting (too much regularization, too few epochs, or missing features).

Linear SVM (Pegasos) — Average Margin vs Epochs.

- Pegasos seeks a max-margin separator with step size $1/(\lambda t)$; an increasing then stabilizing margin indicates convergence.
- Early fluctuations are expected as support vectors settle; persistent oscillations may mean step size too large or λ too small.
- A late falling margin can indicate an over-aggressive step or ill-scaled features—standardization mitigates this.

- Larger margins tend to generalize better, but margin alone does not guarantee optimal Precision/Recall at a fixed threshold.

Linear SVM (Pegasos) — Accuracy vs Epochs.

- Training accuracy improves with the margin, then plateaus as the margin stabilizes.
- Spikes/dips occur when batches contain many (near-)support vectors; decaying steps reduce this with time.
- If accuracy degrades after a peak, try increasing λ or reducing the step size.
- A low, flat curve points to underfitting; try more epochs, a smaller λ, or a kernel if the boundary is non-linear.

6.2 Holdout Metrics

Table 1: Holdout test performance. Best F1 in **bold**.

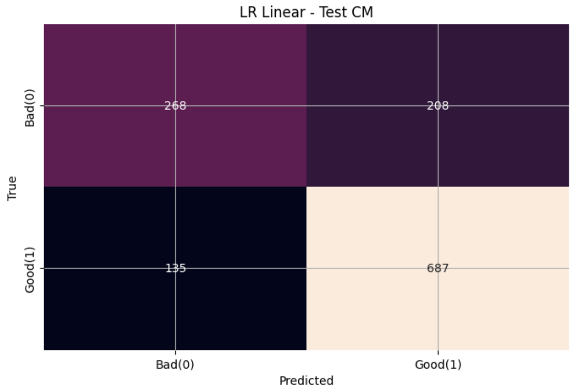
Model	Accuracy	Precision	Recall	F1
KLR RBF	0.748	0.783	0.833	**0.807**
LR Linear	0.736	0.768	**0.836**	0.800
SVM RBF	0.693	0.685	0.954	0.798
SVM Poly	0.697	0.746	0.791	0.768
KLR Poly	0.686	0.814	0.655	0.726
SVM Linear	0.652	0.754	0.668	0.708

Table 2: ROC AUC on the test set.

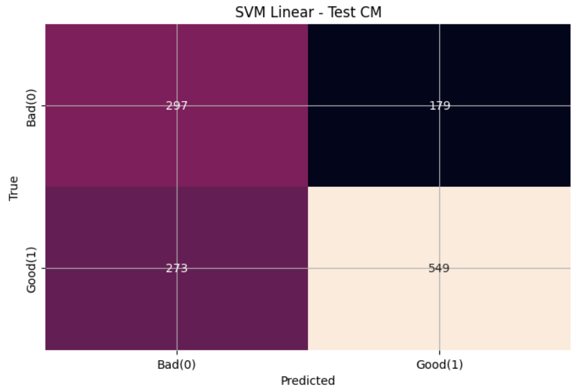
Model	AUC
KLR RBF	0.788
LR Linear	0.783
KLR Poly	0.765
SVM Poly	0.735
SVM RBF	0.689
SVM Linear	0.686

6.3 Confusion Matrices & Operating Points

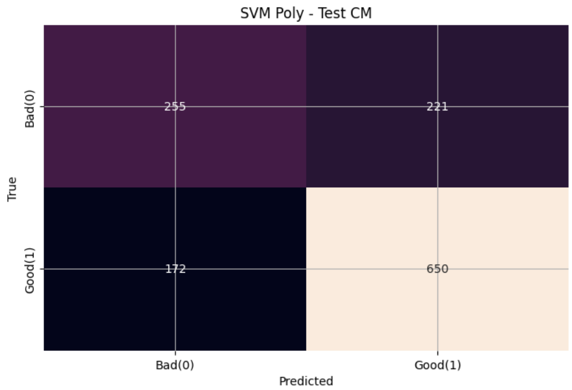
Fig. 11–12 show confusion matrices. Notably, **SVM-RBF** achieves very high **recall** (few false negatives) at the expense of more false positives, while **KLR-RBF** and **LR** strike a more balanced trade-off.



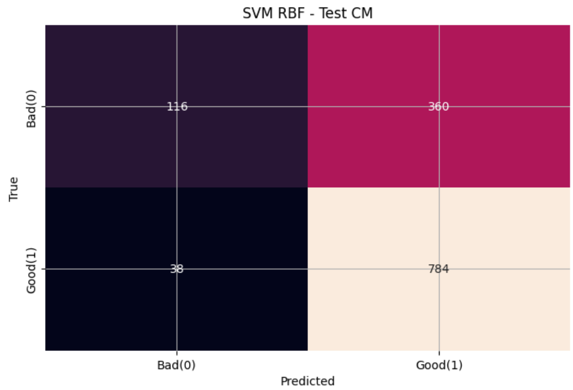
(a) LR (Linear)



(b) SVM Linear

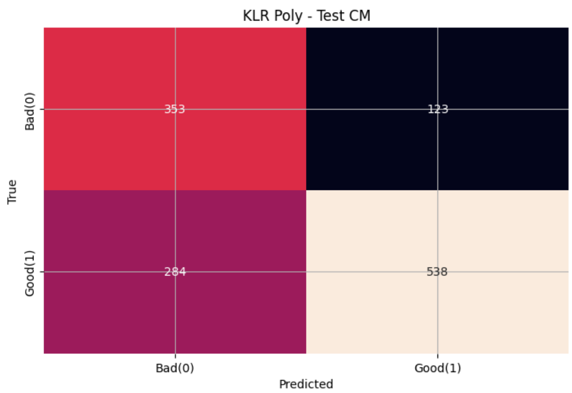


(c) SVM Poly

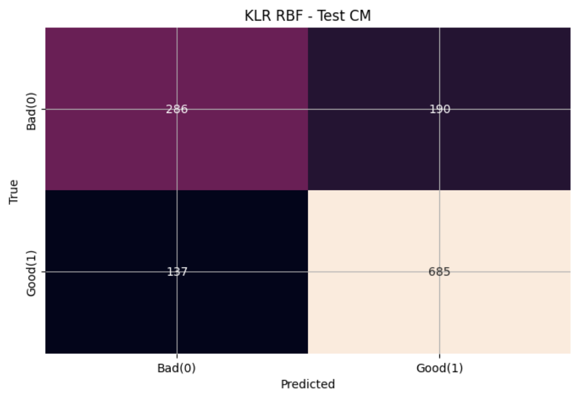


(d) SVM RBF

Figure 11: Confusion matrices (part 1).



(a) KLR Poly



(b) KLR RBF

Figure 12: Confusion matrices (part 2).

Confusion Matrices (per model)

How to read these matrices. Rows are the *true* class (`Bad`=0, `Good`=1) and columns are the *predicted* class. Top-left = TN, top-right = FP, bottom-left = FN, bottom-right = TP. Precision = $TP/(TP + FP)$ quantifies purity of predicted `Good`; Recall = $TP/(TP + FN)$ quantifies coverage of true `Good`.

Logistic Regression (Linear). Balanced trade-off with moderately low FP and FN, matching its test Precision 0.768 and Recall 0.836. Misclassifications cluster near the decision boundary. To tilt toward higher precision: raise the classification threshold above 0.5 or increase L2; to raise recall: lower the threshold or reduce L2 slightly.

SVM Linear. Lower Recall (0.668) implies *more FN* than LR—a conservative boundary typical for a linear margin on mildly non-linear data. Actions: reduce λ (weaker regularization), run more epochs, or switch to a kernel; optionally shift the decision threshold on the distance-to-hyperplane score.

SVM Polynomial (degree 2). Improves Recall (0.791) over SVM-Linear but introduces more FP (Precision 0.746), reflecting quadratic interactions. Tuning: adjust (α, c) to modulate feature scale/bias; increase λ to curb FP, or reduce λ to chase recall.

SVM RBF. Extremely high Recall (0.954) with lower Precision (0.685) means *many FP but very few FN*. Interpretation: small λ or large γ pushes the boundary outward. To rebalance: increase λ or decrease γ; or move the threshold away from 0 on the signed margin.

Kernel Logistic Regression (Polynomial, degree 2). High Precision (0.814) and lower Recall (0.655) indicate a *conservative positive rule* (many FN, few FP). This kernel captures some interactions but misses local structure; to boost Recall: reduce regularization, increase degree to 3, or alter (α, c) while keeping kernel centering/normalization.

Kernel Logistic Regression (RBF). Best overall F1: Precision 0.783, Recall 0.833. Matrix shows a balanced FP/FN profile; probabilistic training with a smooth kernel yields calibrated scores. To target a specific operating point, tune the probability threshold on validation (e.g., maximize F1 or business metric).

7 Interpretation and Insights

7.1 What the Metrics Tell Us

F1 balances Precision and Recall. Here, **KLR-RBF** is best by F1 and AUC, indicating strong overall ranking and a good operating point. **LR** remains competitive—evidence that much of the signal is approximately linear after preprocessing. **SVM-RBF**'s very high recall suggests a low effective margin or larger γ, aggressively labeling positives.

7.2 Feature-Level Intuition (from Linear Models)

Table 3 reports LR coefficients (standardized feature space). Alcohol and sulphates are positive; volatile acidity and density are negative—consistent with domain knowledge.

Table 3: Logistic Regression coefficients (standardized features). Signs align with oenological expectations.

Feature	Coefficient
alcohol	1.034
sulphates	0.328
free sulfur dioxide	0.268
residual sugar	0.431
fixed acidity	0.125
pH	0.111
citric acid	-0.091
chlorides	-0.026
density	-0.228
total sulfur dioxide	-0.375
volatile acidity	-0.707

7.3 Misclassification Analysis

The top "confidently wrong" KLR-RBF cases have borderline chemistry: alcohol $\approx$ 11.4–12.0, moderate sulphates, low residual sugar/chlorides; true class = 0 but predicted = 1 with confidence $\sim$ 0.38–0.45. These sit near the decision boundary or may reflect latent sensory attributes not captured by lab measures. This also explains why a simple degree-2 polynomial doesn't suffice, whereas RBF improves boundaries locally.

7.4 Over/Underfitting

CV and test scores are aligned; LR training curves are smooth (Fig. 9). SVM-RBF's pattern reflects a different operating point rather than classical overfit; adjusting λ or γ, or tuning the decision threshold, can rebalance precision/recall.

8 Design Choices Justified

- **Z-Score + IQR winsorization**: handles heavy tails without discarding data; crucial for stable optimization in from-scratch learners.
- **Standardization**: necessary for gradient descent and kernel scale interpretability.
- **Exclude `type` in modeling**: prevents shortcut learning and keeps the task focused on chemistry $\rightarrow$ quality.
- **RBF vs. Poly**: RBF captures smooth local structure; Poly-2 adds global pairwise interactions. Results show local smoothness matters more here.
- **F1-centric selection**: appropriate under moderate imbalance; complemented by Accuracy/Precision/Recall and AUC.

9 Recommendations and Next Steps

Operating point: If recall is paramount (screening), SVM-RBF is attractive; otherwise use KLR-RBF/LR and tune the threshold for desired Precision/Recall. **KLR-Poly improvements**: try degree 3 with stronger reg (0.03–0.3), larger c and smaller α, keep centering/normalization, and consider $\eta_t = \eta_0/\sqrt{t}$. **Robustness**: repeated CV (e.g., 3×5-fold) and sensitivity to winsor caps (e.g., 2.5×IQR vs 1.5×IQR). **Explainability**: report bootstrapped CIs for LR coefficients; consider classwise reliability diagrams for KLR probabilities.

10 Reproducibility Checklist

- Random seeds fixed for splits and initializations.
- All transforms fit on training data only (and within each CV fold).
- Hyperparameters selected by 5-fold CV; test used once for final reporting.
- Notebook: `Wine_Classification.ipynb`.

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

Antonella Convertini