Reproducing and Extending Wine Quality Prediction from Physicochemical Tests

Archaeology of Intelligent Machines — Project Description

First Author
Maeschi Stefan
Second Author
Timplarescu Alexandru
Third Author
Suhan Nicoleta

Abstract

We reproduce and extend the classic study of Cortez et al. (2009) on predicting vinho verde wine quality from physicochemical tests. The original work frames wine quality as a regression problem and evaluates multiple regression, neural networks, and support vector machines (SVM), reporting that SVM performs best across practical error tolerances and mean absolute deviation (MAD). Building on this, we implement comparable models in Python and additionally test a random forest baseline. We provide open-source code and document the full pipeline: preprocessing, training, hyperparameter tuning, and evaluation using MAD and tolerance-based accuracy.

1 Introduction

Problem. Wine quality scoring is typically obtained via sensory evaluation, but physicochemical lab tests provide objective measurements that may help predict quality scores. Our goal is to predict the vinho verde quality score (0–10) from 11 physicochemical attributes and compare several machine learning approaches, following Cortez et al. (2009).

Contributions (per member).

- **Nico :** Implemented and documented the linear regression baseline (data loading, preprocessing, evaluation). Ensured reproducibility and project structure.
- **Stefan :** Implemented the neural network approach and performed hyperparameter tuning; produced result tables/plots. Implemented a random forest baseline.
- **Alex :** Implemented SVR (RBF kernel) and a random added tolerance-based accuracy evaluation and comparisons to the paper. Added tolerance-based accuracy evaluation and comparisons to the paper.

Approach summary. We treat the task as regression and evaluate models using MAD and tolerance-based accuracy at $T = 0.5$ and $T = 1.0$ (a prediction is correct if $|y - \hat{y}| \leq T$), consistent with the evaluation perspective emphasized in Cortez et al. (2009).

Why this project. This paper is a widely used benchmark example of practical regression with meaningful evaluation beyond a single metric (e.g., tolerance-based accuracy). Reproducing it is a good exercise in experimental rigor and reproducible ML.

Related work. Cortez et al. (2009) compare multiple regression (MR), neural networks (NN), and SVM for the red and white vinho verde datasets, and report that SVM achieves the best overall performance under several evaluation criteria.

What we did not fully reproduce / unclear parts. The original paper uses an integrated variable and model selection strategy (including sensitivity analysis and a parsimony search). We reproduce the main modeling/evaluation setup, but our hyperparameter search protocol and implementation details may differ; we document these differences in the repository.

2 Approach

Repository. https://github.com/sm1267/Wine-Quality/tree/main

2.1 Tools and software

Python, Jupyter notebooks, NumPy/Pandas, scikit-learn (plus any additional libraries used in the repository).

2.2 Dataset

We use the vinho verde wine quality dataset described by Cortez et al. (2009). Each sample includes 11 physicochemical input variables and a sensory quality score (0–10). The paper reports

two subsets: red (1599 samples) and white (4898 samples) (Cortez et al., 2009).

2.3 Preprocessing

We standardize numeric features (zero mean, unit variance) for scale-sensitive models (SVR, MLP). We split data into training/testing sets (or cross-validation folds) and keep preprocessing consistent across models to ensure fair comparison.

2.4 Data loading and variants

All our implementations follow the same pattern:

1. Read CSV files with `sep=";"`: `winequality-red.csv` and `winequality-white.csv`.
2. Choose a `wine_variant` in {`red`, `white`, `both`}.
3. If `both` is selected, concatenate red and white and add a binary `wine_type` feature (0 = red, 1 = white).
4. Define X as all columns except `quality`, and y as the `quality` column.

2.5 Train/test split

For linear regression, MLP, and random forest notebooks we use an 80/20 holdout split. For the SVR script we use a 67/33 split (`test_size`=0.33, `random_state`=42), matching the 2/3–1/3 holdout ratio discussed in Cortez et al. (2009).

2.6 Models

Linear regression (baseline). We use scikit-learn's `LinearRegression` on the selected dataset variant. This baseline does not apply feature scaling.

Neural network regressor (MLP). We implement a multilayer perceptron regressor using scikit-learn's `MLPRegressor` in a pipeline with `StandardScaler`:

- `hidden_layer_sizes = (10,)`
- `activation = logistic`
- `solver = adam`
- `max_iter = 2000`
- `random_state = 42`

Scaling is used because MLP optimization is sensitive to feature magnitudes.

Support Vector Regression (SVR). We implement SVR with an RBF kernel:

$$K(x, x') = \exp(-\gamma \|x - x'\|^2),$$

following the SVR formulation discussed in the paper. We standardize inputs, fix $C = 3$, set γ based on the selected wine type (e.g., $\gamma = 0.7$ for white and $\gamma = 0.15$ for red in our script), and compute ϵ using a 3-nearest neighbor heuristic similar in spirit to **?**:

$$\epsilon = \hat{\sigma}/\sqrt{N}, \quad \hat{\sigma} = \frac{1.5}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i^{(3-NN)})^2.$$

Random forest classifier (alternative formulation). In addition to regression, we try to predict the discrete quality label with a `RandomForestClassifier`. We add one engineered feature:

- **Acidity level:** `fixed acidity` is discretized into four bins (`Low`, `Medium`, `High`, `Very High`) and label-encoded as an additional feature.

We train a random forest with `n_estimators=100` and `random_state=42`.

2.7 Evaluation

Regression metrics. We report:

- **MAD** (mean absolute deviation): $\frac{1}{N} \sum_i |y_i - \hat{y}_i|$.
- **Tolerance-based accuracy** at $T \in \{0.25, 0.5, 1.0\}$:

$$Acc_T = \frac{1}{N} \sum_i 1[|y_i - \hat{y}_i| \leq T].$$

Classification metrics. For the random forest classifier we report:

- Standard **accuracy**, **precision/recall/F1** (macro and weighted), and the **confusion matrix**.
- An **ordinal tolerance accuracy** where a prediction is counted as correct if it is within one quality point of the true label.
- One-vs-rest ROC curves and macro-AUC over classes (when probabilities are available).

Setting	MAD	$\text{Acc}_{T=0.5}$	$\text{Acc}_{T=1.0}$
Paper (Red, SVM)	0.46	62.4%	89.0%
Paper (White, SVM)	0.45	64.6%	86.8%
Ours (Red, SVM)	*0.46*	*61.3%*	*88.4%*
Ours (White, SVM)	*0.447*	*63.1%*	*87.2%*
Ours (RF, best)	*0.35*	*67.7%*	*96.6%*

Table 1: Reference results from Cortez et al. (2009) and our reproduction (fill in).

2.8 Results

Table 1 shows the paper SVR reference numbers and placeholders for our runs (to be filled with the metrics printed by our scripts/notebooks).

Because the random forest is a classifier (not a regressor), it is evaluated with classification metrics (accuracy/F1/confusion matrix) and an optional within-one-grade tolerance. These outputs and plots (ROC/AUC and feature importance) are provided in our notebook.

3 Limitations

- **Protocol differences.** Exact reproduction depends on splits, CV folds, and selection procedure; deviations can change results.
- **Rare extreme classes.** Very low/high quality scores are rare; tolerance metrics can hide poor tail performance.
- **Feature scope.** Only 11 lab measurements are used; sensory perception may depend on unobserved factors.

4 Conclusions and Future Work

We implemented a reproducible pipeline for wine quality prediction inspired by Cortez et al. (2009). The original study reports SVR/SVM as the best-performing approach; our reproduction tests whether the same conclusion holds under our training protocol and compares it to a random forest baseline.

What we would do differently. Tighter experimental control (fixed seeds, identical folds across models) and automated experiment tracking.

Future work.

- Implement the paper's variable selection and sensitivity analysis more faithfully.
- Add modern baselines (e.g., gradient boosting) and uncertainty estimates (prediction intervals).
- Add interpretability (permutation importance, partial dependence) for key features such as alcohol and sulphates.

References

Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. 2009. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, 47(4):547–553.