Machine Learning Project Notebook – Modeling & Evaluation Guideline (ML Part)

Iman Mossavat, Lecturer Fontys, October 2025, Version 2

> **Note:** Depending on the problem, some of these steps may differ, merge, or be skipped. This document is **not a form to be filled out**, but a **guideline** to help check whether the essential reasoning and reflection steps of a complete ML process have been covered.

Scope Clarification

> The sections **Data Understanding** and **Data Cleaning** are evaluated by the **DAIA teacher**.
> Evaluation focus starts **from Feature Engineering onward**, covering **representation issues, modeling, and evaluation** — the *core ML process*.
> While representation is important (encoding, grouping, bias), the main emphasis is **modeling quality**, **comparative analysis**, and **interpretation of results**.
> Include **at least one imbalanced classification problem** to meaningfully discuss **precision, recall**, and related metrics.

Themes in ML Work

Theme	Core Expectation	Typical Student Gap
1. Structure & Planning	Organize the notebook logically; include a plan, track progress, state what was done and learned.	Students dive into code with no structure or reflection.
2. Depth of Understanding	Explain each major decision (encoding, model choice, hyperparameters); show comprehension of what code means.	Students run GPT-generated code without understanding.
3. Modeling & Analysis	Build proper ML models, compare alternatives, discuss bias, over/underfitting, and metrics.	Students stop at "working code" or rule-based logic.
4. Experimentation & Evaluation	Perform parameter sweeps, interpret feature importance, test different methods, compare results.	Students run one model once, no systematic comparison.

Theme	Core Expectation	Typical Student Gap
5. Reflection & Communication	Summarize findings, discuss limitations, evaluate personal learning and relevance.	Students deliver outputs, not insights.

Machine Learning Project Notebook Template

0 Project Overview & Plan

Purpose: Define the modeling goal and keep track of progress.
Learning goal: Understand where you are in the process and what each step contributes to the bigger picture.

Step	What I Learned .
Data exploration	
Feature engineering	
Modeling	
Evaluation	

Step	Status	Feedback From last time
One-page synthesis (objective + subjective) present		
Target + features clearly stated		
Baseline shown and compared		
≥ 2 models compared (train & test/CV metrics)		
Hyperparameter sweep with interpretation		
Over/underfitting check with interpretation		
Feature importance + bias discussion (quantified)		
Final decision + prioritized next steps		
Reproducibility notes & citations of external code		
Narrative present under major outputs (1–3 sentences each)		

1 One-Page Synthesis — Fill This First (Required)

Purpose: Summarize findings and learning before showing code.
Learning goal: Distinguish between what the *data* says (objective) and what *you* learned (subjective).

A. Objective Insights

- Target & task (one line), e.g., "Target = `liked_episode` — binary classification."
- List of features concisely. Include feature importance findings if available.
- Key findings of hyperparameter optimization.
- One-line performance baseline on train and test (e.g., majority class accuracy = X; chosen model F1 = Y).
- Two to three useful findings from data/models (features, imbalance, major signal).
- Note on model reliability (overfit? stable CV? train vs. test gap = Z).
- Decision based on findings (e.g., choose model X, change metric to recall).

B. Subjective Reflections

- What surprised you or what remains unclear.
- Which design choices could introduce bias and why.
- If additional time were available, what would you try next.
- How this work helps the stakeholder.
- One insight you would explain differently to a teammate.

> This section must be completed before any model code runs.

---

2 Data & Model Visualization

Purpose: Create interpretable plots that reveal trends, outliers, and model behavior.
Learning goal: Understand data distributions, feature effects, and model predictions clearly.

2.1 General Principles

- Always check for outliers. If a few extreme points dominate, consider:
  - Removing or isolating them for separate analysis.
  - Applying transformations (log, square root) to compress skewed ranges.
  - Using percentile-based clipping to focus on the bulk of the data.

- Compare results visually:
  - Keep axes consistent across plots for direct comparison.
  - Use zoomed subplots to reveal details when the full range is large.
  - Apply log scales when variables span multiple orders of magnitude.

2.2 Scatter & Density Plots

- Avoid messy scatter plots when points overlap heavily.
- Use density estimation to reveal data structure:
  - **Seaborn 2D KDE:** Easy, reliable, interpretable.
  - **Scikit-learn KernelDensity:** Flexible for custom grids and larger datasets.
  - **2D histograms** (`plt.hist2d`) are acceptable for quick checks but less smooth.

2.3 Feature / Residual Visualization

- Residual plots: make axes symmetric around zero; highlight largest residuals.
- Compare histograms/distributions **before and after transformations** (e.g., scaling, encoding).
- Overlay multiple results (train vs. test predictions) to identify discrepancies.

2.4 Visualization Best Practices Box

- Use consistent color maps, axis labels, and scales across plots.
- Document any transformations or clipping applied.
- Always explain in one to two sentences what the plot reveals about data or model.

---

3 Feature Engineering (Representation Focus)

Purpose: Transform raw data into model-ready features.
Learning goal: Understand how features affect bias, interpretability, and model quality.

- Avoid eliminating features too early unless justified (e.g., negligible impact, perfect correlation).
- Describe the final features and rationale for each transformation.
- Explain why representation is appropriate (ordinal vs. categorical).
- Justify grouping or encoding choices; note potential bias.
- Do not perform manual feature selection based solely on correlation:
  - After hyperparameters are optimized, compare against an extensive model or use algorithmic methods (Lasso, ElasticNet, tree importance, permutation importance,

SHAP).

- Manual dropping without comparison risks representation bias.

---

4 Intermediate Checks Between Steps

Purpose: Verify the impact of preprocessing transformations.
Learning goal: Ensure preprocessing improves model readiness without unintended distortions.

- Visualize feature distributions before and after each major transformation (scaling, encoding, imputation) using histograms, boxplots, or KDEs.
- Avoid jumping straight into pipelines; examine intermediate results.
- Comment on distribution or feature changes and why they matter for stability.
- Check for outliers/extremes at each stage.
- Document decisions to clip, transform, or remove values for reproducibility.

---

5. Modeling (Algorithms & Training)

Purpose: Connect theory to application.
Learning goal: Understand algorithm behavior and interpret model performance.

- List candidate algorithms and rationale.
- Define train/test split or CV strategy.
- Train at least two models and report train/test metrics.
- Discuss metric choice (precision, recall, F1, RMSE) and reasoning.

5.1 Hyperparameter Exploration

Purpose: Understand model sensitivity and tuning.
Learning goal: Learn how hyperparameters influence generalization, overfitting, and performance.

- **READ THE DOCS!** Identify key hyperparameters (e.g., `max_depth`, `C`, `learning_rate`).
- Optimize hyperparameters before comparing algorithms (basic sweep or cross-validated search).
- Use **parameter sweeps** to visualize sensitivity; only compare models after tuning. Plot both train and test performance.

- Co-optimize a few hyperparameters via `GridSearchCV` or, if multiple parameters are involved, more efficient alternatives: `RandomizedSearchCV`, `HalvingGridSearchCV`, or Bayesian optimization (e.g., Optuna).
- Use parameter sweeps to define meaningful ranges, then apply fine optimization.
- If a hyperparameter best value is **NaN**, explain its effect using documentation.
- For AdaBoost: tune `n_estimators`, `learning_rate`, and `max_depth` carefully.

Reporting Hygiene

- Clearly report tuned parameters and remaining defaults.
- Interpret observed effects briefly (≤ 3 sentences).
- Explain overfitting/underfitting in curves and subsequent tuning decisions.
- Use tables to summarize results when multiple plots are involved.

---

6. Evaluation & Over/Underfitting Checks

Purpose: Assess whether the model truly learns intended patterns.
Learning goal: Distinguish apparent vs. genuine model performance.

- Consider both most performant and most interpretable models.
- Show train vs. test performance; include learning curves if possible.
- Discuss stability across folds or resamples.
- Interpret precision, recall, confusion matrix in context.
- For regression, include residual plots with symmetric axes around zero.
- Explore effects of sample size by subsampling.

6.1 Error Analysis & Outliers

Purpose: Understand where the model fails and why.
Learning goal: Identify error patterns and outliers to guide feature engineering, model choice, or data collection.

- **Identify the largest errors:**
  - Regression: points with the largest residuals.
  - Classification: misclassified examples (false positives and false negatives).
- **Inspect challenging cases:**
  - Check whether extreme cases (e.g., very expensive cars) dominate errors.
  - Identify outliers, mislabeled data, or underrepresented patterns.
- **Optional outlier experiments:**

    - Remove extreme points or outliers and rerun models to see performance changes.
    - Document changes in train/test metrics and interpret how the model reacts.
- **Analyze hyperparameter effects on residuals:**
    - Visualize residuals with boxplots for train and test sets.
    - Compare residual spread for different settings (e.g., `max_depth` in decision trees or boosting models).
    - Check whether increasing model complexity helps reduce errors for difficult cases or leads to overfitting.
- **Reflect and summarize:**
    - Describe patterns observed in errors and potential causes.
    - Suggest improvements: more data, alternative features, robust models, or model simplifications.
    - Document findings and next steps clearly in 1–3 sentences per key insight.

---

7. Feature Importance & Bias Checks

Purpose: Interpret model decisions and detect potential unfairness.
Learning goal: Understand which features matter and possible bias.

- Quantify importance using tree importance, permutation importance, or SHAP.
- Explain conclusions from plots/tables; move exploratory plots to appendix.
- Compare simplified vs. full feature sets; report metric changes.
- Never claim importance without quantitative evidence.
- Reflect on fairness implications.
- Test simple counterfactuals by modifying a feature to observe prediction changes.

---

8. Decision & Next Steps

Purpose: Consolidate learning and communicate takeaways.
Learning goal: Translate results into actions or future experiments.

- State final model choice and rationale.
- Prioritize next experiments (2–3 items).
- Mention limitations or open questions.

---

9. Reproducibility & Transparency

Purpose: Ensure work is traceable and credible.
Learning goal: Build habits of scientific integrity.

- Cite external code (URL or “GPT generated”) and verify correctness.
- Explain what the code does in your own words.
- Save random seeds and document package versions for reproducibility.