Lab 1: Data-Centric ML Pipeline, Leakage Control, and Deployment Readiness

Projects in ML and AI — Spring 2026 | CSCI-4170/6170
Mini-project (2-3 weeks)

Release Week: Week 3
Lab Time: 90–110 minutes per week
Recommended Duration: 3 weeks
Team Policy: Individual or groups of 2–4

Learning objectives

- Create an end-to-end supervised ML pipeline that is reproducible and resistant to data leakage.
- Produce dataset documentation (datasheet) and a data-quality audit that supports responsible/ethical ML.
- Train and evaluate at least one non-tree, non-logistic baseline model (e.g., kNN, SVM, Naive Bayes, Ridge/ElasticNet for regression).
- Choose an operating threshold (classification) or decision rule (regression) using a simple, explicit cost/utility assumption and justify the decision policy.
- Complete a hands-on NVIDIA DLI lab and document your progress, results, and troubleshooting steps.

Constraints and allowed methods

Required

- Python + Jupyter Notebook/Google Colab
- pandas, NumPy, scikit-learn, matplotlib
- Git repository with clear commit history
- Keep your reading information gathered for this week's lab handy

Explicitly prohibited in Lab 1 (to avoid overlap with HW1/HW2)

- Do NOT implement Logistic Regression from scratch for the core model.
- Do NOT use Decision Trees, Random Forests, Gradient Boosting, AdaBoost, XGBoost, or related tree ensembles.
- Do NOT use AutoML or LLM APIs to generate code or results.

Problem setup: choose a real-world dataset and define the decision

Choose ONE task type only:

- Binary classification (recommended): predict an event/risk where false positives and false negatives have different costs.
- Regression (allowed): predict a continuous outcome and define a decision rule (e.g., flag top-k risks) for thresholding.

Dataset requirements:

- Public dataset with a stable download link and clear license/terms.
- At least 2,000 rows and at least 10 usable features after cleaning (exceptions require instructor approval). You may use the one you used for homework 1.
- Must contain at least one plausible source of leakage risk (IDs, timestamps, post-outcome fields, duplicates, or near-duplicates).

Deliverables (what to submit)

- A Git repo called Lab 1, containing notebooks + source code.
- Notebook 1: 01_datasheet_and_audit.ipynb (data documentation + quality + leakage audit).
- Notebook 2: 02_modeling_and_decision.ipynb (pipelines, models, calibration, thresholding).
- Notebook 3: 03_nvidia_dli_lab_progress.ipynb (selected NVIDIA DLI lab, completed exercises, results, and brief reflection).
- Model Card (1 page, markdown): intended use, metrics, limitations, risks.
- Experiment Log (CSV): one row per experiment run (model, params, seed, metrics, timestamp).

What you must build

1. Week 1 Checkpoints

Part A — Dataset datasheet + data-quality audit

1. Create a short dataset datasheet with: motivation, target definition, data source + license/terms, a brief feature dictionary (top features and types), and known limitations/risks.
2. Perform a quick data-quality audit: missingness summary, duplicate rows check, target distribution, and one bias/ethics consideration relevant to your dataset.
3. Create a leakage-risk note. Identify at least 1–2 plausible leakage vectors (e.g., IDs, timestamps, post-outcome fields, duplicates) and state how you will prevent them.

Part B — Leakage-safe preprocessing and baselines (Week 1)

4. Implement a leakage-safe split strategy using a single train/test split. If the dataset has time ordering, use a time-based split (e.g., last 20% as test) or briefly justify why not.
5. Build a single scikit-learn Pipeline (use ColumnTransformer only if you have mixed numeric/categorical features) so preprocessing is fit only on the training set.
6. Train one simple baseline model (choose one: Gaussian Naive Bayes, kNN, or Ridge/ElasticNet for regression). A trivial baseline is optional.
7. Report one primary metric on the test set (accuracy/F1 for classification; MAE/RMSE for regression) and include one small supporting artifact (confusion matrix for classification OR a residual summary for regression).

Week 2 Checkpoints

Part C — Core model: SVM (Week 2)

8. Train an SVM-family model (LinearSVC/SGDClassifier for linear; SVC for kernel) and do light tuning (2–3 settings). Use either a small validation split from the training set or 3-fold CV.
9. Compare the SVM to your Week 1 baseline using the same split, preprocessing, and metric definitions.
10. Log your runs in the Experiment Log (baseline + SVM variants). Keep seeds fixed for reproducibility.

Part D — Decision thresholding under a simple cost model (Week 2)

11. If classification: (recommended) use CalibratedClassifierCV with sigmoid (Platt scaling) if your model does not output calibrated probabilities. If you skip calibration, state that you are using uncalibrated scores.
12. Define a simple 2x2 cost matrix (or a constraint such as recall ≥ X). Choose a threshold using validation predictions to minimize expected cost or satisfy the constraint.
13. Report: metrics at the default threshold AND at your chosen threshold; include a confusion matrix and a 3–5 sentence decision justification.

Week 3 Checkpoints

Part E — Pick one of the NVIDIA Labs in the Lab 1 folder

During the lab session, complete as many exercises as possible. Document what you completed, what you learned, and any issues/troubleshooting. Target meaningful progress (roughly 50%+ of exercises) rather than perfection.

Required for Graduate (6170) (choose any ONE)

- Add confidence intervals for key performance metrics using bootstrap; report uncertainty.

- Add a lightweight interpretability section (permutation importance + partial dependence) and discuss failure modes.
- Add a constrained optimization: select threshold under a fairness constraint (e.g., equal opportunity gap ≤ δ) and explain trade-offs.

Weekly milestones and Thursday check-ins

Week	**Milestone (due by Thursday lab session)**	**Evidence to show in the 3–5 minute explanation**
Week 1	Mini-datasheet + quick audit + leakage note + baseline pipeline	Datasheet section completed, split + pipeline code, baseline metric, confusion matrix/residual summary, commits
Week 2	SVM + light tuning + threshold decision	Experiment log with ≥5 runs (teams) or ≥3 runs (individual), threshold choice + short justification, confusion matrix, commits
Week 3	Significant progress on one selected NVIDIA DLI lab	Show notebook outputs and notes; target meaningful progress (roughly 50%+ of exercises) with documented issues

End-of-lab explanation (required for full credit)

At the end of each Thursday lab session, your team must give a 3–5-minute explanation using this structure:

- What changed since last session? (show git commits or notebook diffs)
- What did you measure and what did you learn? (show one table/plot)
- One technical decision: what you tried, expected outcome, actual outcome, next step
- Instructor verification questions (any team member may answer)

Note: If the team cannot demonstrate evidence of progress and shared understanding, the oral-check component will be reduced for that week.

Grading rubric

Category	What we look for
Week 1 (Part A): Datasheet + data-quality audit	Clear target definition, brief feature dictionary, missingness/duplicate check, and one ethical consideration

Week 1 (Part B): Leakage-safe split + pipeline + baselines	Correct train/test split, pipeline prevents leakage, one baseline model, reproducibility (seed/versioning)
Week 2 (Part C): SVM + comparison	SVM with light tuning, disciplined comparison to baseline using same split/pipeline
Week 2 (Part D): Calibration + thresholding	Simple cost model/constraint, justified threshold choice, metrics at default vs chosen threshold (calibration optional)
Week 3 (Part E): NVIDIA DLI lab progress	Evidence of meaningful exercise completion (target ~50%+), with brief notes on results and troubleshooting
Week 3: Communication: report + model card + repo hygiene	Readable report, key figures, concise model card, clean structure
Thursday explanations (ongoing)	Evidence of progress, clarity, individual accountability

Submission checklist

- All notebooks run top-to-bottom without manual edits; all random seeds are fixed and reported.
- Repo folder contains an Experiment Log CSV with ≥5 runs (teams) or ≥3 runs (individual).
- Report includes setup, baseline, SVM results, threshold decision (and calibration if used), NVIDIA lab progress summary, and limitations.
- Model Card included as markdown in the repo.