Project: Learning Across Domains on High-Dimensional Tabular Data

DDL: 30 December 2025

1 Introduction

This project focuses on machine learning with **high-dimensional, low-sample tabular data**, a setting that frequently appears in areas such as medical diagnosis, and other scientific applications. Students should design models that must (1) fit a small labelled dataset with thousands of features and (2) generalize to a dataset with another domain. Students work in teams of up to 4 people .

1.1 Problem setting

You are given a **binary classification** dataset with:

- **High feature dimension** (on the order of several thousand input variables),
- **Small sample size** (dozens to a few hundred examples),

Concretely, think of each instance as a high-dimensional vector

$$\mathbf{x} \in \mathbb{R}^d, \quad d \gg n,$$

with a binary label $y \in \{0, 1\}$ (e.g., disease vs. healthy, cancer subtype A vs. subtype B). n is the size of data.

Your goal is to design learning pipelines that:

1. Handle the *"small n, large p"* regime (few samples, many features),
2. Can **transfer** to a related but distributionally different test set (another domain).

1.2 Data overview

You will receive three files:

- `train.csv` – **labelled training data** from Domain A
  - Columns: `f1, f2, ...f12700, y`.
  - You may use this file (and only this labelled file) to train, validate, and select your models.
- `test_in_domain.csv` – **unlabelled in-domain test data** from Domain A
  - Same feature columns as `train.csv`, but **without** the label y.
  - Used for **Task 1** evaluation.
- `test_cross_domain.csv` – **unlabelled cross-domain test data** from Domain B
  - Feature columns are compatible with the training data (same dimension and ordering).
  - Used for **Task 2** evaluation.

The true labels for both test files are hidden and will be used by the instructor for grading.

2 Tasks

2.1 Task 1: In-domain generalization (same domain)

Using only the labelled training data from Domain A, you must:

1. Build classification models that predict the label y on Domain A.
2. Carefully design validation strategies (e.g., cross-validation, stratified splits) to cope with high dimensionality and limited samples.
3. Produce predictions for **every sample** in `test_in_domain.csv`.

You will submit a prediction file:

- `pred_in_domain.csv`, containing a single column `y_pred` with one prediction per row.
- The i-th entry of `y_pred` must correspond to the i-th sample in `test_in_domain.csv`; that is, the predictions must strictly follow the row order of `test_in_domain.csv`, and each value should be your predicted class label (e.g. 0/1).

2.2 Task 2: Cross-domain generalization (different domain)

In Task 2, you must apply the same model that you trained on the labelled data from Domain A (and used to generate your Task 1 predictions) to make predictions on Domain B, represented by `test_cross_domain.csv`.

Key points:

- You do not receive any labels from Domain B.
- The feature representation is compatible, but the data distribution is shifted (another domain).
- You may use any **unsupervised** information from Domain B (e.g., clustering, distribution matching) but you cannot access true labels.

You will submit:

- `pred_cross_domain.csv`, containing a single column `y_pred` with one prediction per row.
- The i-th value of `y_pred` must be the predicted class label (e.g. 0/1) for the i-th sample in `test_cross_domain.csv`.

3 Evaluation Guidelines

The project has a total of 100 marks. Your score depends on the quality of your report and the performance of your predictions on the two tasks.

4 Grading Scheme

Each team receives a final project score based on two components:

$$\text{FinalScore} = \min\big(100,\ \alpha \cdot (\text{ReportScore} + \text{AlgoPerformanceScore})\big),$$

where

- $\text{ReportScore} \in [0, 30]$,
- $\text{AlgoPerformanceScore} \in [0, 70]$,
- α is a scaling factor that depends on the team size.

4.1 Team-size-dependent scaling factor α

Let m denote the number of students in the team. The scaling factor α is defined as

$$\alpha = \begin{cases} 1.1 & \text{if } m = 2, \\ 1.0 & \text{if } m = 3, \\ 0.9 & \text{if } m = 4. \end{cases}$$

The final score is capped at 100. That is, if $\alpha \cdot (\text{ReportScore} + \text{AlgoPerformanceScore}) > 100$, the final score is recorded as 100.

4.2 Report component (up to 30 points)

It should be clearly structured and written in understandable academic English. In particular, it should briefly situate your work in the context of related methods, explain how your algorithm is designed and implemented for Task 1 and Task 2 (including the main modeling and preprocessing choices), and describe the division of work among team members.

4.3 Algorithm performance component (up to 70 points)

The algorithm performance score AlgoPerformanceScore is based on your predictive performance on both Task 1 (in-domain test set) and Task 2 (cross-domain test set). For each task, teams will be ranked according to an evaluation metric chosen by the instructor (e.g., accuracy or AUC). These rankings across all teams and both tasks will then be combined to assign a performance-based score in the range $[0, 70]$: teams with better overall ranking across Task 1 and Task 2 receive higher AlgoPerformanceScore.

5 Submissions

Each team should make **one joint submission**; all team members share the same set of files and final scores. You must submit the following files:

1. **Report**
   - `TeamID_Name_report.pdf`
   - Example: `12_SanZhang_report.pdf`, where `Name` is the group leader's name.
2. **Predictions and model**
   - `TeamID_Name_pred_in_domain.csv`
   - `TeamID_Name_pred_cross_domain.csv`
   - `TeamID_Name_model.xxx` (trained model file, e.g. `.pkl` or `.pt`)
3. **Code**
   - `TeamID_Name_code.zip` containing your source code and configuration files.

All files should be submitted before the deadline announced on bb.

6 Prohibition and Academic Integrity

You will receive **0 marks** for this project if any of the following occurs:

- Late submission beyond the grace period specified by the instructor.
- Plagiarism in the report or code (copying from other students, online repositories).
- Label leakage or cheating, including but not limited to:
  - Directly searching for the hidden labels of the test sets.
  - Using external labelled data that are too similar to the test sets without explicit permission.
  - Manually tuning predictions after seeing any part of the hidden labels.

You may reuse known algorithms and open-source implementations (e.g., from scikit-learn), but you must cite your sources and clearly explain how you used or modified them.