Machine Learning Foundations
Data Preparation

February 18, 2026

1 Description

In this assignment, you will carry out data preparation and feature engineering tasks using the University of California, Irvine (UCI) Bank Marketing dataset. This dataset contains information about direct marketing campaigns conducted by a Portuguese banking institution. The objective of those campaigns was to determine whether a client would subscribe to a term deposit after being contacted by the bank.

Your task is not to build an accurate predictive model, but to carefully prepare the data so that such a model can be built correctly. You will gain hands-on experience with data exploration, handling implicit and explicit missing values, encoding categorical variables, scaling numerical features, addressing class imbalance, and reasoning about potential data leakage. These steps constitute the foundation of any reliable machine learning pipeline.

At the end of the assignment, you will train a simple Logistic Regression model as a sanity check. The emphasis, however, is on the quality, justification, and ordering of your data preparation decisions.

2 Objectives

This assignment gives you the opportunity to:

- explore and critically analyze the structure and characteristics of a real-world dataset;
- identify and properly handle explicit and implicit missing values;
- encode categorical variables and justify the encoding strategy adopted;
- select and justify an appropriate feature scaling strategy for numerical variables;
- split the data into training, validation, and test sets using an appropriate stratification strategy;
- address class imbalance and discuss its impact on model evaluation;
- perform principled feature selection while avoiding data leakage; and
- train a simple Logistic Regression model as a consistency check of the data preparation pipeline.

3 Dataset and Problem

The dataset can be downloaded from Kaggle. It is provided in `.csv` format and can be loaded into a Pandas DataFrame using `read_csv`. You should use the smaller version of the dataset (`bank-additional.csv`). The dataset contains information about direct marketing campaigns conducted by a Portuguese banking institution. Each row corresponds to a client who was contacted during a telemarketing campaign. The goal of the campaign was to determine whether the client would subscribe to a term deposit. The dataset includes demographic, financial, campaign-related, and macroeconomic variables, such as:

- **Demographic attributes**: age, job, marital status, education;
- **Financial attributes**: housing loan, personal loan, credit in default;
- **Campaign-related attributes**: contact type, month and day of contact, number of contacts performed during the campaign (`campaign`), number of days since last contact (`pdays`), number of previous contacts (`previous`), and call duration (`duration`);
- **Macroeconomic indicators**: employment variation rate (`emp.var.rate`), consumer price index (`cons.price.idx`), consumer confidence index (`cons.conf.idx`), Euribor 3-month rate (`euribor3m`), and number of employees (`nr.employed`).

Problem: For this assignment, we focus on data exploration and feature engineering for the following machine learning problem: **Given client and campaign information available at the time of contact, predict whether the client subscribes to a term deposit (i.e., a type of short-term investment).**

4 Instructions

Code the following tasks in Python using a Jupyter Notebook. Create a GitHub (GH) repository for this course and push your notebook to that repository. Name your notebook using the following template: `assignment_1_<name_surname[_surname]>.ipynb`. Do not wait until the end of the assignment to push your work. Instead, get used to pulling and pushing to your GH repository while you work. This practice will be beneficial once you start collaborating with your team on a shared codebase. See §7 for more information about using GH.

For each task, explain your approach, justify your choices, and explicitly reference the conceptual tools you used from the lectures. In particular, you should motivate your decisions about data cleaning, encoding, scaling, feature engineering, and dataset splitting. Your explanations should make it clear which information you treat as available at prediction time, and why.

You will encounter the following potential challenges:

- **Handling missingness:** identifying explicit and implicit missing values (e.g., `unknown` categories), and deciding when to drop, impute, or model missingness explicitly.
- **Feature engineering with constraints:** deciding which raw attributes should be transformed, removed, or combined, and predicting the impact of those choices on a linear model.
- **Avoiding data leakage:** ensuring that any transformation that learns parameters from data (e.g., imputation, scaling, resampling, feature selection) is fit on the training set only.
- **Addressing class imbalance:** understanding how imbalance affects both model behavior and evaluation, and applying balancing techniques without contaminating validation/test data.

IMPORTANT: A fundamental part of the assignment is to perform the tasks in the correct order. The tasks below are intentionally listed in *alphabetical order*. You must decide the order in which to execute them and argue why your chosen order avoids the conceptual and methodological issues discussed in class. Start from *Identifying the Prediction Target*. I give this one away for free ;-)

Task: Addressing Class Imbalance

Lecture material: Lecture 3 (Class Imbalance), Lecture 4 (Evaluation Metrics), Lecture 9 (Pipeline Discipline).

- Quantify the class distribution in the training set and explain why imbalance is or is not a concern for this prediction task.
- Propose and apply a resampling strategy (e.g., random oversampling, SMOTE, or ADASYN). Clearly justify at which stage of the pipeline the resampling step should occur.
- Justify your choice of resampling method in terms of its assumptions and expected effect on the learning algorithm.
- Explain what would happen if resampling were applied *before splitting the dataset into training, validation, and test sets*. Discuss the implications for model evaluation.
- Briefly discuss how class imbalance affects evaluation metrics such as accuracy, precision, and recall.

Note: Resampling is part of the training procedure and must be applied to the training set only. Validation and test sets must preserve the original class distribution.

Task: Data Loading and Exploration

Lecture material: Lecture 1 (Problem Formulation), Lecture 2 (Data Inspection and EDA).

- Load the dataset into a Pandas `DataFrame`.
- Inspect the structure of the dataset: number of observations, number of features, data types, and basic summary statistics.
- Identify which variables are numerical and which are categorical.

- Analyze the distribution of the target variable and comment on potential class imbalance.
- Detect explicit and implicit missing values (e.g., special categories such as `unknown`).
- Visualize the distribution of at least:
  - two numerical variables; and
  - two categorical variables.
- Identify at least one variable that may require special consideration before modeling (e.g., due to distributional properties, extreme skewness, or availability at prediction time), and briefly justify your reasoning.

Note: Exploratory analysis is not a checklist of plots. Each visualization or statistic should support a specific observation or hypothesis about the data.

Task: Data Splitting

Lecture material: Lecture 2 (Data Splitting and Leakage), Lecture 9 (ML Pipeline).

- Split the dataset into training, validation, and test sets.
- Justify your choice of proportions for each split.
- Perform *stratified* splitting with respect to the target variable and explain why stratification is necessary for this dataset.
- Clearly describe at which stage of your pipeline the split must occur, and explain what types of data leakage would arise if splitting were performed later.

Note: A recommended strategy is to first split the dataset into a training set and a temporary set, and then split the temporary set into validation and test sets. Use the `stratify` argument of `train_test_split` where appropriate.

Task: Encoding Categorical Variables

Lecture material: Lecture 4 (Categorical Encoding), Lecture 6 (Linear Models), Lecture 9 (Feature Engineering and Expressiveness).

- Identify all categorical variables in the dataset.
- Distinguish between *nominal* variables (categories without intrinsic order, e.g., job type) and *ordinal* variables (categories with a meaningful order, e.g., education level), and justify your classification.
- Select and apply an appropriate encoding strategy for each categorical variable.
- Clearly state which encoders must be fitted on the training set only, and explain why.
- Analyze how encoding changes:
  - the dimensionality of the dataset;
  - the interpretability of model coefficients;
  - the types of decision boundaries a linear model can represent.

Note: Encoding is not a purely mechanical transformation. Your justification should explicitly connect your encoding decisions to the assumptions and behavior of Logistic Regression.

Task: Feature Scaling

Lecture material: Lecture 5 (Feature Scaling), Lecture 6 (Logistic Regression and Optimization)..

- Identify the numerical variables that require scaling.
- Select and apply an appropriate scaling strategy (e.g., standardization or normalization) to those variables.
- Justify your choice of scaling method in the context of Logistic Regression.
- Clearly state which transformations must be fitted on the training set only, and explain why.
- Discuss how feature scaling affects:

  - gradient-based optimization;
  - the magnitude and comparability of model coefficients;
  - the interpretation of regularization penalties.

Note: Feature scaling is not a cosmetic transformation. Your justification should explicitly connect your scaling decision to the mathematical behavior of linear models.

Task: Feature Selection

Lecture material: Lecture 5 (Feature Selection), Lecture 6 (Linear Models), Lecture 9 (Pipeline Discipline).

- Identify and remove features with very low variance, if any. Justify the criterion used to define "low" variance.
- Identify highly correlated numerical features and decide whether any should be removed. Clearly state the threshold used and justify your decision.
- Discuss whether any features should be removed based on conceptual considerations (e.g., redundancy, availability at prediction time, or risk of data leakage).
- Explain why feature selection must be performed using the training set only.
- Discuss the consequences of performing feature selection on the entire dataset before splitting.

Note: Feature selection is not purely statistical. Your reasoning should explicitly connect your decisions to the assumptions and stability of Logistic Regression.

Task: Identifying the Prediction Target

Lecture material: Lecture 1 (Problem Formulation), Lecture 2 (Data Inspection).

- Inspect the dataset and identify which column should be treated as the target variable for this assignment.
- Justify why this column represents the appropriate prediction objective in the context of the marketing campaign.
- Identify at least two other variables that could superficially appear to be valid targets and explain why they should *not* be treated as the prediction objective.

Task: Managing Missing Values

Lecture material: Lecture 2 (Data Inspection), Lecture 5 (Preprocessing and Pipeline Discipline).

- Identify both *explicit* missing values (e.g., `NaN`) and *implicit* missing values (e.g., categories such as `unknown` or sentinel numerical values, i.e., values that may represent special codes rather than genuine measurements).
- Quantify the extent of missingness for each affected variable.
- Propose and justify a strategy for handling missing values in each case (e.g., removal, imputation, separate category, indicator variable).
- Clearly state which operations must be fitted using the training set only, and explain why.

Note: Your strategy should distinguish between "data cleaning" decisions (e.g., correcting inconsistent entries) and "modeling" decisions (e.g., whether missingness itself may carry predictive information).

Task: Task Ordering

Lecture material: Lecture 2 (Data Splitting and Leakage), Lecture 5 (Preprocessing), Lecture 9 (ML Pipeline).

- Determine the correct order in which the data preparation tasks in this assignment should be performed.
- Provide a structured justification for your chosen order.
- For each step in your proposed sequence, explain:
  - what information is allowed to be used at that stage;
  - what information must *not* be used;
  - what type of data leakage could occur if the order were changed.
- Discuss at least one example of an incorrect ordering and explain the consequences it would have on model evaluation.

Task: Training a Logistic Regression Model

Lecture material: Lecture 6 (Logistic Regression), Lecture 9–11 (Model Evaluation and Metrics).

- Train a Logistic Regression model to predict whether a client subscribes to a term deposit.
- Use the validation set to generate predictions.
- Report at least Accuracy, Precision, and Recall on the validation set.
- Compare the model's accuracy with the Zero Rule baseline and briefly interpret the result.

Note: The goal here is not to squeeze out the best possible performance. The goal is to verify that your data preparation pipeline is coherent and correctly implemented. If your preprocessing is principled, the model should behave sensibly. If it behaves strangely, that is a signal to revisit earlier decisions. Have fun finding a visually appealing way to display the predictions or the confusion matrix on the validation set. This is your chance to make the output readable and professional 8-)

5 Deliverables

Submit a single Jupyter Notebook containing:

- Well-structured and executable Python code implementing all required tasks;
- Clear Markdown explanations accompanying each major step;
- A critical discussion of your design choices, including alternative approaches you considered and why you rejected them.

Your explanations should be sufficiently detailed to stand alone as a short technical document. The notebook must run from start to finish without errors using `Restart Kernel and Run All`. Remember: The emphasis is not on producing the highest-performing model, but on demonstrating disciplined reasoning, correct pipeline design, and professional presentation.

6 Evaluation Criteria

- **Correct Implementation and Pipeline Discipline** (40%): All required steps (data loading, target identification, splitting, preprocessing, encoding, scaling, resampling, feature selection, and modeling) must be correctly implemented and executed in a logically coherent order that prevents data leakage.
- **Explanation and Rationale** (30%): Decisions must be justified. You should clearly explain why you selected specific methods, thresholds, or transformations, and how those choices relate to the assumptions of Logistic Regression and to the avoidance of data leakage.
- **Depth of Analysis and Critical Thinking** (20%): Your work should demonstrate understanding rather than mechanical execution. This includes identifying potential pitfalls (e.g., imbalance, implicit missing values, multicollinearity, leakage) and discussing their implications.
- **Code Organization and Professional Presentation** (10%): Code should be modular, readable, and well-documented. Explanatory Markdown should be structured and precise. GitHub should be used correctly and consistently.

WARNING: Code that fails to run from start to finish will receive 0, regardless of how trivial the error may seem. This might affect a single task but, often, might affect all the following parts of the notebook. Thus, be careful, **you must test the entire notebook before submission.** To perform a complete test, execute 'Restart Kernel and Run All' and ensure that every cell runs without errors. One last thing... Did I mention that you must test the entire notebook before submission?!

7 Minimal Everyday GitHub Guide

Follow this workflow to create and maintain your code in a course repository. The commands below assume you are using Git from a terminal, but the same logic applies when using an IDE with GitHub integration.

1. Log into GitHub and select `New Repository`.

2. Name the repository exactly: `ML-fundamentals-2026`. Do not change this name.
3. Check `Add a README file`.
4. Select `Python` under `Add .gitignore`.
5. Choose the `MIT License`.
6. Open the repository page and copy the SSH clone URL under `Code`. **Note:** this assumes you have already configured Git with your SSH key. If not, follow an appropriate tutorial.
7. In a terminal on your computer, clone the repository locally:
   - `git clone <repository-url>`
8. Move into the newly created directory:
   - `cd ML-fundamentals-2026`
9. Create your notebook file:
   - `assignment_1_<name_surname[_surname]>.ipynb`
10. Start implementing the assignment tasks.
11. Whenever you complete a meaningful step (e.g., one task), commit your changes:
    - `git add assignment_1_<name_surname[_surname]>.ipynb`
    - `git commit -m "Implement Task X: short meaningful description"`
    - `git pull`
    - `git push`
12. Before starting work on any day, always run:
    - `git pull`

    This establishes good professional habits and prevents conflicts when collaborating on a shared codebase.