Machine Learning for Artificial Intelligence Engineering, Winter Term 2025
Homework 2
Course Project

Maximilian Baronig

Tutor:	Nicolas Pfob, nicolas.pfob@student.tugraz.at
Points to achieve:	35 Points
Deadline:	23.01.2026 23:59 (strict, no late submissions allowed)
Hand-in procedure:	Submit **a report (PDF)**, the predictions `submission_leaderboard_GROUPNAME.csv`, `submission_final_GROUPNAME.csv`, and your project code/assets as `Code.zip`
Plagiarism:	If detected, 0 points on the entire assignment sheet for all parties involved. If this happens twice, we will grade the group with "Ungültig aufgrund von Täuschung"
Course info:	TeachCenter, https://tc.tugraz.at/main/course/view.php?id=6273

Contents

General remarks

Your submission will be graded based on:

- Correctness (Is your code doing what it should be doing? Is your derivation correct?)
- The depth of your interpretations (Usually, only a couple of lines are needed.)
- The quality of your plots (Is everything clearly readable/interpretable? Are axes labeled? …)
- Your submission must run with Python 3.11.5. In your submitted `Code.zip`, there should be a `requirements.txt` with all packages one needs to install to run your code (with their versions). You are free to use **any** third-party packages in this project.

The usage of LLMs and LLM-based coding assistants is **explicitly allowed** for this assignment. Feel free to use them both for coding, and for coming up with ideas/strategies to tackle the issues you face.

1 Problem Description

You work in a company that produces sensors that measure several physical quantities at the production sites of your customers. Your company has developed a sensor that can detect if a certain discrete event $y \in \{0, 1, 2, 3\}$ occurs at a production facility of your customer. Moreover, the sensor records auxiliary information about the environment (scalar values, collected in $\mathbf{x} \in \mathbb{R}^d$, $d = 12$) when it reports the event y. However, the parts of the sensor that "directly" measure y are extremely expensive, whereas the parts that measure the auxiliary information $\mathbf{x}$ are *much* cheaper. The company also assumes that there is some relationship between $\mathbf{x}$ and y, which you are tasked to find. If there was a model f_θ that predicts y from $\mathbf{x}$ well, your company could omit the expensive parts of the sensor, just record $\mathbf{x}$ and predict y using f_θ.

Data Collection. To collect data, the company intern is tasked to mount one sensor at each of the three different production sites at your customer's facility (i.e., three sensors in total). Since there is no time dependency in any of the measurements, we can assume that the recorded pairs $(\mathbf{x}, y)$ are i.i.d. samples of some true, unknown distribution $p^*(\mathbf{x}, y)$. Unfortunately, the intern forgot to note down which recordings came from which sensor. Thus, you are given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ where $\mathbf{x}^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \{0, 1, 2, 3\}$ for each $i \in \{1, \ldots, N\}$, with $d = 12$ and $N = 9000$.

Outliers. Sometimes, the mounted sensors are known to malfunction randomly: In these cases, the recorded $(\mathbf{x}, y)$ pairs are highly atypical, i.e., $\mathbf{x}$ is not a sample from $p^*(\mathbf{x})$, and y is not a sample from $p^*(y \mid \mathbf{x})$. Roughly 20% of the recordings are known to be outliers. In these cases, your model should not just predict $\hat{y} = f_\theta(\mathbf{x})$, but should check if $\mathbf{x}$ looks like an outlier (if so, this could be displayed to the user instead of blindly showing a prediction $\hat{y}$). To this end, the company also supplies you with a (small) set of auxiliary recordings $\mathcal{D}_{\text{out}} = \{\mathbf{x}^{(i)}\}_{i=1}^{O}$, with $O = 57$, which are known to be outliers (manual inspection).

Leaderboard. In this assignment, you will try to solve these problems by applying various machine learning techniques. Similar to popular online machine learning challenges, there will be a public leaderboard on TeachCenter which will evaluate the performance of your models on a hidden test set which you will have no access to. The structure is based on the workflow of real-world machine learning projects and encourages you to be creative about your approaches. We will focus on experimentation, evaluation and interpretability of different machine learning methods.

Supplied Data Files. Regarding the data, you will receive the following files:

- `D.csv`, containing the data in $\mathcal{D}$
- `D_out.csv`, containing the data in $\mathcal{D}_{\text{out}}$
- `D_test_leaderboard.csv`, containing test points $\{\mathbf{x}^{(i)}\}_{i=1}^{2943}$ you will need to classify to get a *public* leaderboard score prior to the deadline
- `D_test_final.csv`, containing test points $\{\mathbf{x}^{(i)}\}_{i=1}^{3000}$ you will need to classify to get a *final* score on the leaderboard *after* the deadline (see Section 6)

2 Exploratory Data Analysis [6 points]

After loading the data, your task is to perform (basic) *Exploratory Data Analysis* (EDA), e.g. distributions or correlations in the dataset. Provide **at least five** meaningful plots and include them in the report. Meaningful in this context means, that patterns, relationships, or anomalies in the data are revealed that can inform and justify modeling decisions. You can use any plotting library you like, but make sure the axes, legends and titles of the plots are informative and meaningful. Some ideas: Dimensionality reduction, clustering, plotting distribution of class labels, correlation between features, etc.

Include in your report:

- Five to ten meaningful and readable plots, as outlined above. [4 points]
- A single paragraph, providing a brief interpretation on all included plots and how they might influence decision making during the experiments. [2 points]

3 Baseline Model [3 points]

You do not want to just evaluate the performance of your model using the public leaderboard. Instead, carefully think about how you want to evaluate your model locally. Pick a validation strategy that you think best represents the actual performance of your models. Be sure to avoid common pitfalls, like label leakage. You will use this validation strategy throughout this project.
Next, train a simple `KNeighborsClassifier` from the `sklearn` library to get your first results. This baseline model will be the starting point of your experiments. Using your validation strategy, pick a good value for K. Think about if you want to preprocess the data. For each of the four classes, compute *Precision*, *Recall*, the F_1 score, and the *Accuracy* of your baseline model. Moreover, compute and report the unweighted average of the per-class F_1 scores (macro[1] average).

Include in your report:

- Briefly describe your validation strategy. [1 points]
- Report your choice of K. Using the baseline model, report *Precision*, *Recall*, F_1, *Accuracy* (for all classes), and the macro-averaged F_1 score. [2 points]

4 Model Experimentation & Validation [19 points]

After obtaining initial results from our baseline model and deciding on a validation strategy, your next task is to iteratively improve your approaches. Train models with **at least three different architectures** and show that you performed **at least ten total experiments** with different hyperparameter configurations, and/or feature/data engineering approaches.
Choices in models, optimizers, regularization techniques, hyperparameters, feature engineering approaches, etc. are completely up to you. You can leverage models we have discussed in this course, but feel free to experiment with models we have not covered in class: Exploration and experimentation are strongly encouraged! If you do not know where to start, we recommend exploring the following models: Logistic Regression, Support Vector Machines, Decision Trees, Random Forests, Gradient Boosted Trees (e.g. with `xgboost`, `LightGBM`), MLPs, or Gaussian Processes. Even though MLPs (neural networks) are not covered in this lecture, you are allowed to use them as one of the three architectures.

Data Preprocessing. For each model type you choose, think critically about whether data preprocessing is necessary or beneficial. Do you need to preprocess the features for your model (e.g. standardization)? Are there any additional features that you could add or can removing features improve performance?

[1] See e.g. `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html`

Validation. Evaluation is crucial to compare the different experimental settings and models. Stick to the validation strategy you already chose carefully in Task 3 and for all experiments, report per-class F_1, Precision, Recall, Accuracy, and macro-weighted F_1. Be sure to avoid common pitfalls, like label leakage, different class proportions or wrong preprocessing on a test set. Hint: If your validation scores are close to your leaderboard scores, you are on the right path. **Important: Insufficient local validation will lead to point deductions!**

Include in your report:

- For each experiment, report per-class F_1, Precision, Recall, Accuracy, and macro-weighted F_1. A minimum of ten total experiments is required with at least three different architectures. [10 points]
- Explain briefly what decisions you have made during the experiments and why (e.g. 'we increased the weight decay because our model showed signs of overfitting'). [3.5 points]
- What changes led to performance improvements and what changes did not? Your report should give a good overview of the methods you have tried, even if they have not led to a performance improvement. [3.5 points]
- Using the model you consider best, plot a confusion matrix with data not used for training. [1 points]
- If you were to start again from scratch what would you do differently? [1 points]

5 Outlier Detection [7 points]

So far, we have focused on the supervised learning task of predicting y from $\mathbf{x}$. We will now work on detecting whether a given $\mathbf{x}$ is an *outlier*, as described in Section 1. There are many ways to perform this task, and you are free to explore any of them.
One way to tackle this problem is to learn a probabilistic model $p_\theta(\mathbf{x})$ such that it fits the data distribution well, i.e., $p_\theta(\mathbf{x}) \approx p^*(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^d$. Since we assume that outlier $\mathbf{x}$ are "unlikely" under p^*, we may expect that p_θ also assigns low density to this outlier. For example, you could fit a *Gaussian Mixture Model* (GMM) $p_\theta(\mathbf{x})$ using the features in our training data $\mathcal{D}$, and proceed to use this GMM to detect outliers. A simple way to get started would be to figure out a threshold $\tau > 0$ such that typically, $p(\mathbf{x}_{\text{in}}) > \tau$, and $p(\mathbf{x}_{\text{out}}) \leq \tau$, where $\mathbf{x}_{\text{in}}$ is an inlier, and $\mathbf{x}_{\text{out}}$ is an outlier. You may use $\mathcal{D}_{\text{out}}$ to determine such a threshold.

Using the outlier detection approach of your choice, filter the original dataset $\mathcal{D}$ to obtain a new dataset that excludes outliers ($\mathcal{D}_{\text{in}}$). Re-train the model you considered best on $\mathcal{D}_{\text{in}}$. Does it improve performance? Include in your report:

- Clearly explain your strategy for outlier detection. Which model did you train and how did you use it to label instances $\mathbf{x}$ as inliers and outliers? If you have tried multiple approaches, briefly describe all of them. [5.5 points]
- Report how the performance of your best model changed when training on inliers only. [1.5 points]

6 Leaderboard Predictions [Bonus Points & Prizes]

Select the model you think works best. For each test data point $\mathbf{x}$, use this model to predict (1) the class label $\hat{y} \in \{0, 1, 2, 3\}$, and `is_outlier` $\in \{0, 1\}$. Save your predictions in the correct format (see `code_skeleton.py`) on the provided test sets as the following `csv` file :

- Save predictions on `D_test_leaderboard.csv` as `submission_leaderboard_GROUPNAME.csv`, where `GROUPNAME` should be your group name that will be displayed on the leaderboard.
- Save predictions on `D_test_final.csv` as `submission_final_GROUPNAME.csv`, which will be used for the final leaderboard score (after the deadline).

Feel free to pick `GROUPNAME` to be different from your Group ID (artistic freedom).

Leaderboard Phases. There will be two leaderboad "phases":

1. **12.12. 23:59 – 23.01. 23:59**: The leaderboard will score the uploaded `submission_leaderboard_GROUPNAME.csv` file and display the results (discussed below) on the leaderboard on TeachCenter. The leaderboard will update everyday at the following times: $\{06{:}00, 09{:}00, 12{:}00, 15{:}00, 18{:}00, 21{:}00, 24{:}00\}$ and will only consider the currently uploaded submission with the correct name.
2. **24.01. 00:01**: On Saturday after the deadline, we will score the predictions in `submission_final_GROUPNAME.csv` (which was uploaded before the deadline). The score we compute here is the **final leaderboard score** and will determine the **winners** of this competition. The top 10 places will receive **bonus points**:
   - 1 bonus point for rank 10
   - 2 bonus points for rank 9
   - ...
   - 8 bonus points for rank 3
   - 9 bonus points for rank 2
   - 10 bonus points for rank 1

Leaderboard Scoring. Note that the test data for the public leaderboard again contains *inlier* and *outlier* points (the same is true for the hidden final test dataset). However, we only consider predictions of your model for *inlier* data points: We compute the *unweighted* (macro) average of the per-class F_1 scores, denoted as F_1^{macro}, since we care equally much about predictions for each class. In the leaderboard, we display this as $\texttt{ClassScore} = 100 \cdot F_1^{\text{macro}}$. Furthermore, we compute the F_1 score of the binary predictions `is_outlier` on *all* the test data, shown as $\texttt{OutlierScore} = 100 \cdot F_1$ in the leaderboard. Finally, the leaderboard score `Score` is given by

$$\texttt{Score} := 0.8 \cdot \texttt{ClassScore} + 0.2 \cdot \texttt{OutlierScore}$$

We will provide a tutors baseline and a random baseline as an orientation. **Important: Stick to the format of the sample submission, any deviations will lead to an error during scoring!**