This document contains the problem statements and instructions related to the **final exam**.

Overview

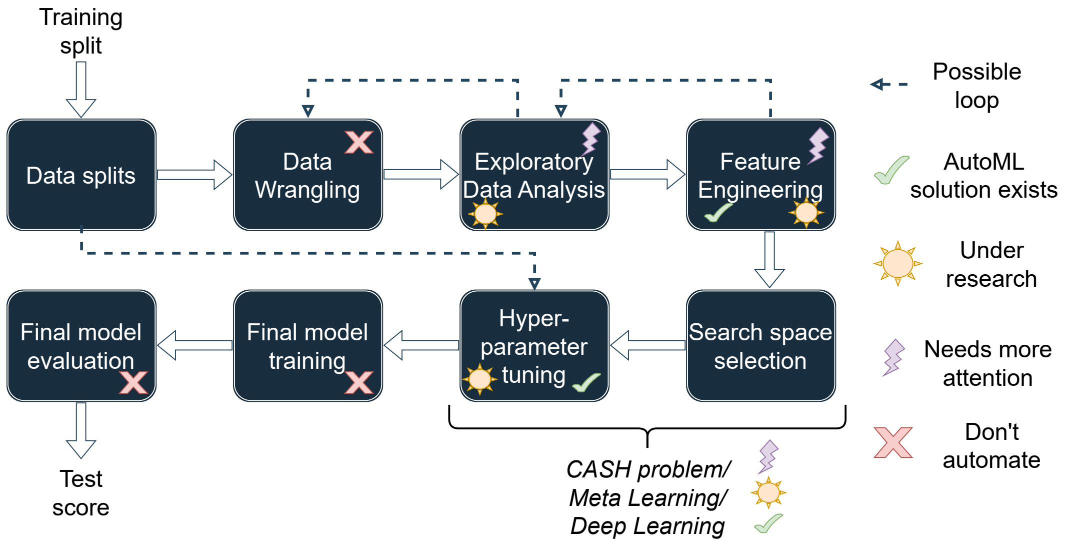
The exam primarily constitutes finding the best tuned and trained model or pipeline for a given data modality, as given by its performance on a specific test set. Each group must choose (at least) one modality to demonstrate their AutoML solution on. The designed solution must include components from the lecture weeks and/or bonus material provided (includes the set of literature provided). Teams have complete freedom in using code from the exercises from this course and available open-sourced code too.

Modality I: Tabular data

Given: The (input, target) of the training and test splits of 3 regression datasets, meant to be used for developing your solution.
`GitHub Code Template`: Contains code to access data and a reference structure, only as an example.
For final code and poster submissions, Github Classroom links will be made available later along with the final test data.
`Poster template`.

Goal: Find the best obtainable R^2 score[1] on the *final-exam-dataset*, where the (input, target) of training and (input, —) of the test split will be released.

Scope: An entire tabular data solution flow is given below, representing what a typical Data Science pipeline looks like. The final target is a *tuned, trained model* that can be *evaluated on the test set.* Any number of sub-component from the pipeline below can be designed and automated to obtain a trained model. We mark the following in the flow: i) Components that have been automated, with public solutions available; ii) Components that are still *under research* but with public solutions available; iii) Components that have not received enough attention; iv) Components that we believe either do not require automation or are not worth pursuing for automation in the context of a university exam.



[1] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#r2-score

Summary: Construct an AutoML pipeline of your choice, that given a training split of a dataset, can yield a model that when evaluated on a hidden test set, gives strong performance. The designed solution should:

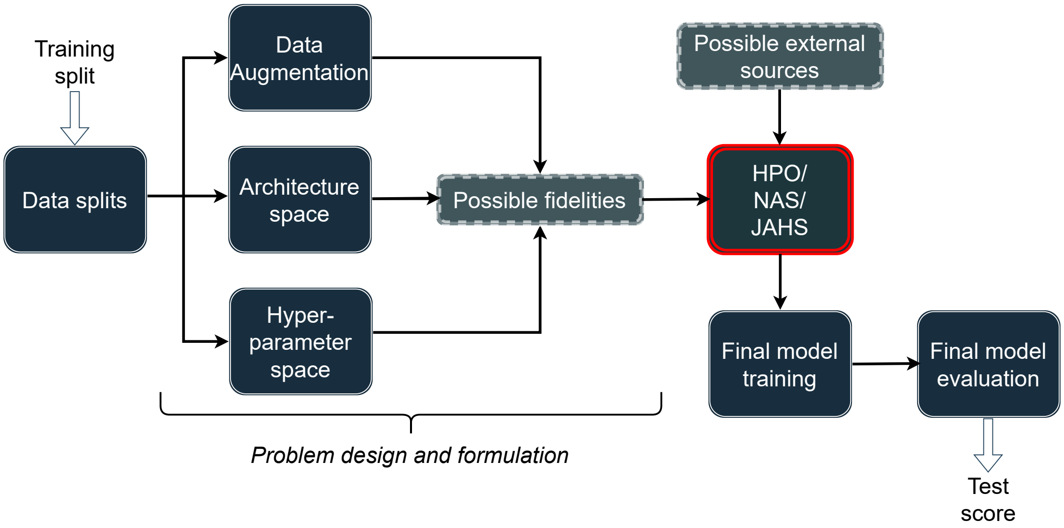
- Be a one-click solution that yields a trained model that can be used for inference on a test set.
- Optionally offers a window of user interaction: at the beginning (e.g. expert prior), once in the middle (staging, active learning, etc.), or at the end (e.g. selection from Pareto front). Multiple modes of interaction can be combined with suitable justification on how the solution is still *Auto*ML.
- The total cost of obtaining a trained model, given a new dataset, should not exceed `24 hours`.
- Any meta-learned component that does not use any of the data provided can be excluded from the above budget.

Reference baseline: When releasing the *final-exam-dataset*, we will release the test score we obtain through a naive, off-the-shelf AutoML solution on the this dataset, for an undisclosed budget. This is to serve as a vague reference and NOT a baseline to necessarily beat. A final performance worse than this number can be compensated by a reasonably innovative and scientific approach.

Modality II: Image data

Given: The (input, target) of the training and test splits of 3 classification datasets, meant to be used for developing your solution.
`GitHub Code Template`: Contains code to access data and a reference structure, only as an example.
For final code and poster submissions, Github Classroom links will be made available later along with the final test data..
`Poster template`.

Goal: Find the best obtainable *top-1 accuracy*[2] on the *final-exam-dataset*, where the (input, target) of training and (input, —) of the test split will be released.



[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.top_k_accuracy_score.html

Scope: Complete freedom to choose and design the components required to achieve the primary goal of the best hyperparameter setting to train the final model. Given image data, the design space could be represented as:

- Choice of architecture(s)
  - Could be part of search space or fixed design
  - Clear motivation and explanation required for the choices made here
  - Can use publicly available pretrained vision models for finetuning
- Choice of hyperparameter space
  - Could be part of search space or fixed design
  - Clear motivation and explanation required for the choices made here, including the bounds and possible defaults
- (Optional) Space of possible augmentations
  - Could be part of search space or be fixed as part of the pipeline
- (Optional) Choice and design of fidelity dimension(s)
  - Any variable that can define a suitable proxy or approximation of the final target problem (epochs, dataset, image resolution, model size, etc.)
  - Each such choice brings its own nuance and should be considered and explained
- (Optional) External source(s)
  - Can be any well-motivated, publicly available, or any other setup that can be trained offline with data not part of this exam
  - Such examples could be, and not limited to, expert prior inputs, meta-trained surrogates, etc.

Summary: Construct an AutoML pipeline of your choice, that given a training split of a dataset, can yield a model that when evaluated on a hidden test set, gives strong performance. The designed solution should:

- Be a one-click solution that yields a trained model.
- Optionally, offers a window of user interaction: at the beginning (e.g. expert prior), once in the middle (staging, active learning, etc.), or at the end (e.g. selection from Pareto front). Multiple modes of interaction can be combined with suitable justification on how the solution is still *Auto*ML.
- The total cost of obtaining the best configuration, given a new dataset, should not exceed `24 hours`.
- The time or cost for the `Final model training` and `Final model evaluation` is **excluded** from the above budget.

Reference baseline: When releasing the *final-exam-dataset*, we will release the test score we obtain through a naive, off-the-shelf AutoML solution on the this dataset, for an undisclosed budget. This is to serve as a vague reference and NOT a baseline to necessarily beat. A final performance worse than this number can be compensated by a reasonably innovative and scientific approach.

Grading guidelines

- Surpassing the reference baseline score is a good benchmark but not the primary goal. Focus on developing a creative approach that:
  - Applies concepts from the lecture
  - Follows good scientific practices for experiments and reporting

  These aspects will be given higher weightage in the evaluation.
- Document the cost of AutoML, including:
  - The time taken to search for the best model configuration
  - Ideally, report the best performance observed over time
  - Alternatively, consider framing the cost as a multi-objective problem
  - If the test performance is relatively lower, explain how it could still be part of the optimal Pareto front when considering the cost
- Submit the complete code along with:
  - A list of dependencies
  - A command to run the method on a new dataset
  - A command to retrain and evaluate a configuration

For poster:

- Present with good practices for empirical science followed
- Denote on poster the weeks from which concepts were used for the designed solution
- Report compute or resources used overall

NOTE: The maximum weightage of grades come from the actual poster, its presentation and question-answers after that, during the poster session.

Use of LLMs

Complete freedom as long as you credit it overall in the poster or presentation for significant input. Copilot integration in IDEs or using LLMs for understanding concepts or finding papers are in fact recommended. *Optional*: Upload a file containing relevant prompts or prompt templates[3] whose results were used.

Submission guide

- The state of code submitted as part of the Github classroom deadline (like the exercises) is part of the exam submission. Please follow the *README.md* in the Github Classroom repository for exact details. It is expected that the submitted code can be run through one command, given dependencies.
- A file containing test predictions with the name, location and format as specified in the Github Classroom *README.md* file.
- Upload poster that will be finally presented. It should have the method description and the analysis on the 3 practice datasets. It should report the test score obtained on the *final-exam-dataset*. The poster[4] should also contain time and resources used and other experimental protocols necessary for sound scientific reporting.

[3] please mask out or anonymize such prompts before uploading

[4] `https://docs.google.com/presentation/d/1lyE-iLGXIKi31CLFwueGhjfcsR_8r7_L/edit?usp=sharing&ouid=118357408080604124767&rtpof=true&sd=true`

- Deadline for code and poster submission will be 06.08.2024 (23:59). In case you want to iterate on poster after this date or missed the deadline, please let us know and bring your own printed poster in the poster session. If code is incomplete until 06.08.2024 (23:59), that will affect grade.

Compute suggestions

There is no restriction on the nature of compute used. The budgets allowed for the project is compute independent. The final solution and performances of each team will not be compared and evaluated in isolation and hence the loose compute constraints.

Moreoever, if under a compute constraint, teams are encouraged to develop more innovative solutions that account for such low compute and demonstrate that AutoML can still improve upon manual tuning of the chosen pipeline!

Some free options we recommend for compute:

- `Google Colaboratory`
- State provided clusters for all students
  - `NEMO`: can request up to $\leq$2k CPUs
  - `Helix`: can request CPUs and GPUs

  This might require a few days to get access and we recommend at least one member from each team explore the protocol for access.
- Check the `TF cluster pool` for possible access to a GPU and increase of allowed disk storage

Any team using private compute resources that include their own hardware or paid cloud services are completely free to do so. However, every resource used MUST be reported in the final presentation.

NOTE: Please DO NOT wait till the last week to find or resolve compute resources!

Contact and TA support

The usual exercise sessions on Thursdays, post-lecture time and Discord will continue to be possible medium for students to ask their questions.

For more specifics:

- *Modality I: Tabular data*: Contact `Eddie`, `Neeratyoy`
- *Modality II: Image data*: Contact `Johannes`, `Neeratyoy`
- Github related issues: Contact `Martin`, `Neeratyoy`
- Organization and admin related queries: mail at automl-lecture-orga23@cs.uni-freiburg.de

---

This assignment is due on 06.08.2024 (23:59). Submit your solution by pushing your codes and the poster to your group's repository. Teams of at most 3 students are allowed.