INFO9023 - Project description

Spring 2025

Introduction

During the class Machine Learning Systems Design you will get to implement one large **group project**.

The goal of this course is to give you the skills required to build real world ML applications. The group project's main goal is to teach you new tools and best practices of MLOps.

You are in control of your project and will be able to make choices in terms of design and tooling.

Most importantly, find a project that excites you, maybe that even serves a real world use case. Pick a cool name and work in teams 👫🚀

Practicals

- Form teams of **2 - 4 students**
- There will be **3 milestone meetings** where you can present your results
- The topics covered in lectures and in labs are split in 6 sprints. The components to be implemented during the group project **follow the same 6 sprints.** We recommend following the pace of the sprints (so implement the components in the same sprint), but as long as you have them implemented for the **milestone meetings** it's fine.
- Each sprint has several work packages.
- Note that all work packages are not mandatory. Some components are just *optional*. You can achieve a perfect score without them if you show great quality in the required work packages. But the optional ones can help you increase your score if not.
- There will be some time after each class to sit down with your team and make progress on your projects. The teaching staff will be in the same room and can provide support.

Handovers

You will get to show your work in two ways:

1. Milestone presentations

There will be 3 milestone presentations where you will get to present your work.

- The practicals for the project are **updated in the Github repo**. Please use that as the central place for organization information concerning the project.

MS	Topics	Sprints
1	Present your general *use case*, the *data preparation* and the result from your model *experimentation*.	Sprint 1 & 2
2	Present your architecture for *model serving, model deployment* and (if possible) *model pipeline*.	Sprint 3 (& 4)
3	Present your overall project work. You can present a demo of your model/use case and any other topic you think is relevant.	All sprints

2. Code submissions

The way to share your implementation will be through **code reviews**.

- Make sure to share your github repo with the teaching staff.
- Follow **Gitflow** practices
    - Develop individual features on feature branches based on develop and merge them as you go
    - At significant milestones, create a Pull Request (PR) from the develop branch to the master branch and include the teaching staff as reviewers.
        - (Careful to follow naming conventions)
- The **README** and other markdown files will be used as the main documentation points. All important aspects of your project should be explained there.
    - We recommend using the main README to explain the project structure and have different markdown files per building blocks (e.g. EXPERIMENTATION.md, DEPLOYMENT.md, …).
    - The PR description should include main new changes and what the teaching staff should actually have a look at
- The teaching staff won’t run the codes, so make sure to heavily document all important points in markdown files.
- For this project, also include non-code material in the repo such as presentation decks
- Once ready, send a link to the pull request to the teaching staff **by email**. Tagging the teaching staff in Github is not enough.

Building blocks

This section covers the components you should implement for your project. The structure follows the overall course sprints.

Sprint 1: Project organization

This sprint will focus on overall project setup and organization. It mostly covers functional aspects of a project as well as collaboration and communication tools.

Note that for each sprint some concepts are explained in the 2nd week. The W*X* next to a task indicates in which week we'll cover that topic in lecture/lab.

#	Week	Work package	Requirement
1.1	W01	Pick a **team** ● Try to mix skills and experience ● If you didn't find one let one of the teachers know and we'll allocate you to one	Required
1.2	W02	Select a **use case** Source options ● Previous course ● Kaggle Datasets ● … Make sure to pick a use case where **data is available**. Note that the ML modeling itself won't be a big part of the course. If you can pick data from one of your previous course projects that's perfect. Ideally pick something with interesting data and a real world application.	Required
1.3	W02	**Define** your use case ● Fill in a ML Canvas template page ● You can skip the *Inference* part as we will tackle that in a later sprint.	Required
1.4	W02	Find a **cool name** for your project ✨	Required
1.5	W02	Setup **communication channel** ● Discord ● Trello board ● …	Required
1.6	W02	Setup a **code** versioning **repository** ● We recommend Github as we will cover Github Actions during this course	Required
1.7	W02	Submit your project by sending a filled in project card to the teaching staff with basic information about your project. We might give you some feedback and ask for parts to be changed.	Required

Sprint 2: Cloud & model development

The goal of this sprint is to prepare your data and train and optimize your ML model.
Note that you will not be graded on the performance of your model, only on your development methodology.
So do **not spend much time optimizing your data or model**.

#	Week	Work package	Requirement
2.1	W03	Prepare your data and run an Exploratory Data Analysis.	Required
2.2	W03	Prepare your Cloud environment. That means creating a Cloud project, granting correct access rights to all members of your group and setting up a billing account. **Attention**: You can have free credits for the Cloud, as explained during the course.	Required
2.3	W04	Train your ML model	Required
2.4	W04	Evaluate your ML model	Required
2.5	W03 & W04	Document your data analysis and model performance	Required

Sprint 3: API implementation

This sprint focuses on building a **model serving API**, a **Docker container** hosting the model serving and **deploying** your model serving API.

On top of that, *if you are interested*, you can experiment with managed services such as Sagemaker Predict or Vertex Predictions.

#	Week	Work package	Requirement
3.1	W05	Build an **API** to **serve your model** and any extra logic that is needed to serve it (e.g. using Flask). You should be able to run the API locally.	Required
3.2	W05	Package your model serving API in a **Docker container**. This too should be run locally.	Required
3.3	W06	**Deploy** your model serving API in the Cloud. You should be able to call your model to generate new predictions from another machine. **Attention**: This can incur Cloud **costs**. Make sure to use a platform where you have credits and not burn through them. You can ask for support from the teaching staff in that regard.	Required

Sprint 4: Model pipeline

In this sprint, you will implement a **model pipeline** to automatically run different steps of your model training and deployment.

#	Week	Work package	Requirement
4.1	W08	Build a **pipeline** to automatically run different sequential components such as training your model and deploying your model. For it you can use orchestrated pipeline tools such as Kubeflow Pipelines, AWS Sagemaker or GCP Vertex. **Attention**: If you run this pipeline in the Cloud it can incur Cloud **costs**. Make sure to use a platform where you have credits and not burn through them. You can ask for support from the teaching staff in that regard.	Optional

Sprint 5: Optimisation & monitoring

You can run your model training in the Cloud. This allows you to **automate** your model training and to use **stronger compute**.
If you decide to do so, you have multiple options:

- Containerise your training script and run it on a VM in the Cloud
- Use a managed service such as Vertex Training or Sagemaker Training

You will also build a simple user interface to let users interact with the results of your model.

#	Week	Work package	Requirement
5.1	W09	Run your model training as a job in the Cloud. You can implement this in different ways: ● Containerise your training script and run it on a VM in the Cloud (e.g. on EC2 or on Cloud Run, example,) ● Use a managed service such as Vertex Training or Sagemaker Training **Attention**: This can incur Cloud **costs**. Make sure to use a platform where you have credits and not burn through them. You can ask for support from the teaching staff in that regard.	Optional
5.2	W10	Build a simple user interface or dashboard to show your results and deploy it on the Cloud.	Optional

Sprint 6: CICD

In this last sprint, you will implement a CICD pipeline to safely and efficiently redeploy your application once changes are made.

#	Week	Work package	Requirement
6.1	W10	Build a CICD pipeline using Github Actions (or other tool) to automatically run some of the following steps. Include at least one step. The rest is optional. It's up to you to decide what is relevant.	Required
6.2	W10	CICD step: Automatically deploy building blocks of your application such as model serving.	Optional
6.3	W10	CICD step: Pylint	Optional
6.4	W10	CICD step: Pytest for any unit test you think is relevant	Optional