Project in Data Intensive Systems

4DV652

Welf Löwe

1

Content

- The course is a project course that gives you realistic problems and settings integrating Machine Learning (ML) and software development.
- You are placed in the role of a small development team within an agile development process that should develop data-driven products.
- You are expected to work using agile processes in teams of ca 5 developers, and you are expected to perform all roles except product owner.
- The startup environment requires fast releases and effective use of available resources focus is put on Lean agile as well as applied ML and data processing.

2

Goals – You should be able to
(from Syllabus)

- ML
  - characterize the role of ML in a software system and how it can be integrated in the structure of such systems,
  - enumerate which properties an ML tool or software library should have to be applied to a given problem,
  - name and explain the most common problems when attempting to use unprocessed data in ML,
  - independently learn different tools, methods, and software libraries used within ML,
  - elicit requirements from a customer and determine which data is needed and which ML techniques is most suitable,
  - from a set of customer requirements, define metrics that can be used to evaluate how well an ML model performs,
  - implement and evaluate a system with an ML component,
- Software development
  - prioritize functionality and continuously release new functionality to customers,
  - ensure that the system is operational,
  - critically reflect on the result of a project and how well it fulfills the customer requirements with respect to, e.g., technology used, architecture (software and hardware), data, metrics, etc.
  - critically reflect on how "agile" and "lean" were used throughout the project with respect to, e.g., the work environment.

3

You will get practical skills in
(from Syllabus)

- ML
  - How ML projects are structured and implemented.
  - Tools, services, and libraries that are used for data analysis and ML, e.g., Tensorflow.
  - The configuration of pipelines for ML systems and exploit available software and hardware, e.g., accelerators.
  - The practice of working with real data with respect to, e.g., collection, processing, and analysis.
  - Evaluation of performance based on customer requirements.
- Software development
  - Experiment-driven development with short design, training, and evaluation cycles.
  - How Lean-Agile combines the ideas from Lean with agile processes?
  - What is waste in software development and how can it be reduced?
  - How can just-in-time-production be applied to software development?
  - How can a team learn, e.g., from reflection after a sprint and how can the process highlight the development team (and their competences).
  - Advanced skills in writing reflection reports.

4

Practical Q&A

- Teacher/TA: Welf Löwe, Alisa Lincke (Q1), Sebastian Hönel (Q2)
- Exam:
  - Grades: A, B, C, D, E, Fx, F.
  - To pass the course, grade E or higher is required for all parts.
  - The final grade is decided (according to Syllabus):
    - Vision and planning documents (20%),
    - Project work (incl. deliverables) (50%),
    - Reflection report - Choices and outcomes (10%),
    - Reflection report - Lean (10%), and
    - Design, implementation, and result (10%).
- Course and course material
  - Available in MyMoodle: https://mymoodle.lnu.se/course/view.php?id=57155
  - Aligned with Machine learning (4DV660) and Deep Machine Learning (4DV661)

5

Software

- ML
  - For implementing/training/testing the approaches: Python3 and ML libraries: https://www.geeksforgeeks.org/best-python-libraries-for-machine-learning/
- Agile
  - Team communication: Your choice, e.g., Slack
  - Code repository:
    - LNU's Gitlab, https://gitlab.lnu.se, students can create their own projects on our server
    - GitHub Teams, https://github.com/team
  - Issue tracking:
    - Youtrack, https://www.jetbrains.com/youtrack/download/get_youtrack.html#section=incloud
  - DevOps:
    - MLflow: https://mlflow.org/
- Endpoint (backend) using the trained ML model and frontend calling it
  - Your choice ☺
- Reporting
  - Jupyter notebook

6

Course Agenda

1. Course introduction (today, week 3)
2. ~~Introduction to Lean-Agile development (week 4)~~
3. Introduction to ML and Python (week 5)
4. Project kick-off (week 6)
5. Five to six sprints; retrospectives and planning on Wednesdays 10am
6. Project deadline, Part 1 (week 12, 24/3)

7

Before you start an ML project

- Define the problem and choose/assemble a dataset
- Choose a measure of success
- Decide on an evaluation protocol
- Set and assess baseline
- Integrate the ML project into your production
  - How do the business requirements trigger ML requirements
  - How to integrate ML results into the product
  - How to integrate ML sprints into the development process

8

7 Steps in ML Projects

Iterations over

1. Collect the data
2. Prepare the data
3. Choose an ML approach
4. Train the model
5. Evaluate the model
6. Tune the hyper-parameters
7. Integrate the model and use it

9

1 – Collect the Data

- The quantity and the quality of your data dictate how accurate our model gets
- The outcome of this step is generally a representation of data that we will use for training/testing
- Be inclusive rather than restrictive (more is usually better) but
- Be realistic and mind the collection costs

10

2 – Prepare the Data

- Wrangle data and prepare it for training/testing
- Clean if necessary (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.)
- Randomize data, which erases the effects of a particular order in which we collected and/or otherwise prepared our data
- Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis
- Select and combine features, which can be automated
- If applicable/necessary, augment the dataset (generate additional data)
- Split into training and testing sets

11

3 – Choose an Approach

- Different approaches/models for different tasks
  - Don't forget classic algorithmic, e.g., optimization, solutions if applicable
- Discussed in the ML courses 4DV660 and 4DV661
- Choose the right one

12

4 – Train the Model

- Derive a parameterization of the model (if parametric model)
- The foremost goal of training is to get a model to make a predictions, inference questions are secondary but still relevant to gain trust
- Training is an optimization problem: minimize error, maximize accuracy, minimize/maximize distance to centroids/separation plane etc
- Example:
  - Simple linear regression $y = f(x) = ax+b$: learns parameters a and b for given pairs x, y minimizing the error
  - Naïve Bayesian classification: learns the probabilities $p(c|x_1, \ldots, x_n)$ for given feature values x_i and classes $c \in C$, predicts the class with maximum conditional probability

13

5 – Evaluate the Model

- Uses some metric or combination of metrics to "measure" objective performance of model
- Test the model against previously unseen data
- This test data is meant to be a fair sample of the data population and, hence, representative for the model performance in the real world,
- Select an appropriate train/test split or cross-validation, depending on domain, data availability, dataset particularities, etc.

14

6 – Tune the Hyper-Parameters

- Model hyper-parameters are parameters of the ML approach used to derive the model parameters
- Model hyper-parameters include, e.g., number of training steps, learning rate, initialization values, etc.
- Tune/adjust model hyper-parameters for improved performance
- Yet another optimization problem: it can be automated using (adaptive) grid or randomized approaches

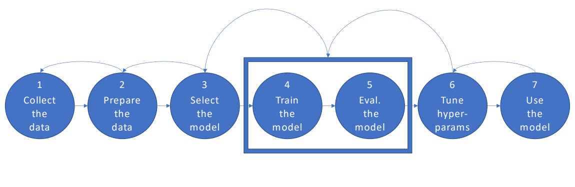
15

7 – Use the Model

- Using further test data that has, until this point, been withheld from the model and for which the ground truth is known (e.g., output value or class), to test the model
- This fraction of the data can also be used for end-to-end testing your integrated solution
- Gives a better approximation of how the model will perform in the real world
- Then integrate and use the model

16

Process of 7 Steps in ML Projects



17

Agile software development principles

The Manifesto for Agile Software Development is based on twelve principles

1. Customer satisfaction by early and continuous delivery of valuable software.
2. Welcome changing requirements, even in late development.
3. Deliver working software frequently (weeks rather than months)
4. Close, daily cooperation between businesspeople and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

18

Agile team roles

- **Team lead** ("Scrum Master") is responsible for facilitating the team, obtaining resources for it, and protecting it from problems. This role encompasses the soft skills of project management but not the technical ones such as planning and scheduling, activities which are better left to the team as a whole.
- **Team member** (developer) is responsible for the creation and delivery of a system. This includes modeling, programming, testing, and release activities etc.
- Optional
  - Technical experts (Development, ML/AI, MLOps)
  - Domain experts
  - Independent tester
- Product owner and Stakeholder are outside your teams, the teacher/TA take these roles

19

Lab assignment 1: Setup the project

- Software development
  - Set up your agile teams and management infrastructure
  - Ca 5 team members including one team lead
  - Define a first sprint with the following tasks
    - Set up Slack, Reporting ,and a Git repository, Install Python 3
    - Write a hello world Web frontend/Python backend system (Client-Server),
    - Don't forget to test its, to check it in to your Git and
    - Build a CI/CD pipeline
- Reporting
  - Install Jupyter notebook
  - Report in a first notebook: team setup (roles, sprint organization, etc.), software and service installation and setup (Slack, Git, Python, CI/CD, Jupyter notebook, etc.)
- Deadline: 2023-24-01

20