Large-Scale Data Engineering for AI

Project 1

Advanced Databases (BDA-GIA)

Context

Data Science Projects

Data science projects require creating systems that deploy data pipelines spanning three different areas:

- Business understanding (domain)
  - What do we want to analyse?
  - What is the added value of an analytical question for the organisation?
- Data management
  - Data discovery
  - Data modeling
  - Data storage
  - Data processing
  - Data querying
- Data analysis
  - Data preparation
  - Modeling
  - Validation
  - Explainability
  - Visualization

Data lifecycle at a real Polish company

An energy company collects measurements on thermal energy consumption of their customers and use it to predict energy consumption

1. Data is generated by several independent devices. Data is stored locally and periodically sent as a text file to the company server. Data is then integrated with weather data at the server-side (join by date)
2. From all this data, they generate a monthly dataset containing measurements (per day) and weather data. For years 2016-2020 it meant 160 CSV files.
3. Over these files, they conducted outlier detection and standarization scripts (i.e., data preparation) written in Python and executed at the server side.
4. The final CSV files are stored in a local directory, which are then read by Python scripts that prepare the data and train a model using *scikit-learn* (https://scikit-learn.org/stable/).
5. Validation and its interpretation is conducted using some data not used for the training, using 10-cross fold validation. The model is then visualized to facilitate its interpretation using *matplotlib* (https://matplotlib.org/).
6. Data scientists iterated throughout this pipe several times until obtaining a satisfactory model persisted as a *scikit-learn* model using *pickel* (https://docs.python.org/3/library/pickle.html).

https://github.com/xLaszlo/datascience-fails

Example of bad practices

- Tasks (1)–(3) were implemented as Python scripts and run in a Jupyter notebook on a local workstation. As a consequence, a private cleaned dataset was created each time the script was run.
- Task (4) was implemented as CSV files.
- Task (5) was not implemented at all, as data were stored locally in OS files. As a consequence, data were not shared and access to data was non-optimal.
- Tasks (6)–(8) were executed in a Jupyter notebook on a local workstation. For this reason, pre-processing, intermediate results, and model building were executed from scratch each time, without the capability to share the results.

Operationalizing Data Science Pipelines

The Baseline Pipeline

Pros

- Dynamic and integrated environment that includes text, code and visualisations
- Enables ad-hoc (not systematic) sharing of analytical workflows
- Enables trial / error (the data science loop)
- Acces to tons of analytical libraries

Cons

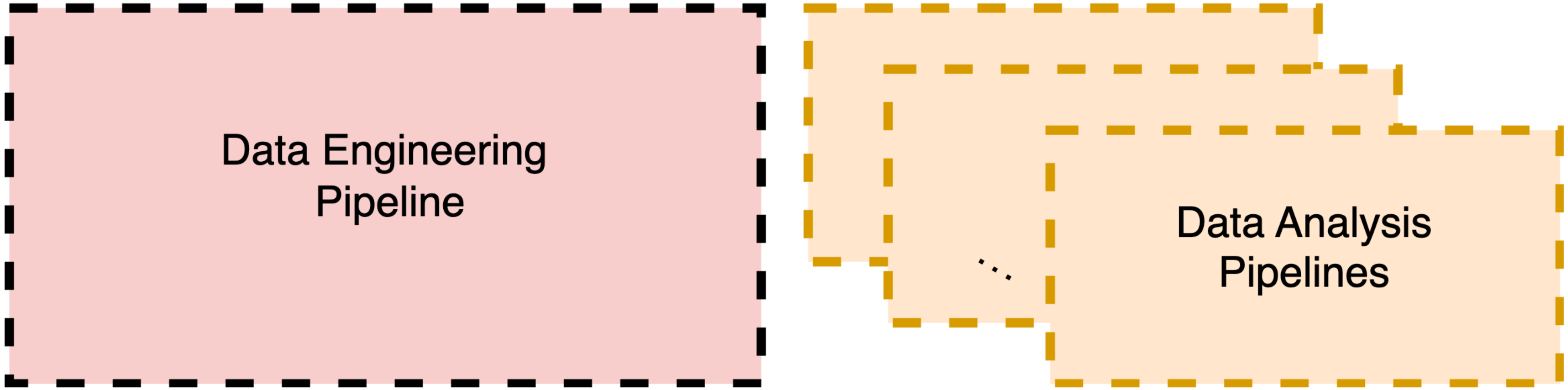
- Most data science libraries process data locally (e.g, in CSV). This approach compromises:
  - Governance
  - Persistence
  - Efficiency in data access
  - Concurrency
  - Reliability
  - Security
  - Performance

NO COMMON BACKBONE

- No code / data sharing / reusage
- No single source of truth (data / code)
- No global optimizations

A common Data Engineering pipeline (DataOps)

- The baseline pipeline does not scale
  - In terms of users (data scientists)
  - In terms of data (volume)
- A systematic approach is required to prepare data for data scientists



DataOps in a Nutshell

The data engineering pipeline (common for the whole organisation)

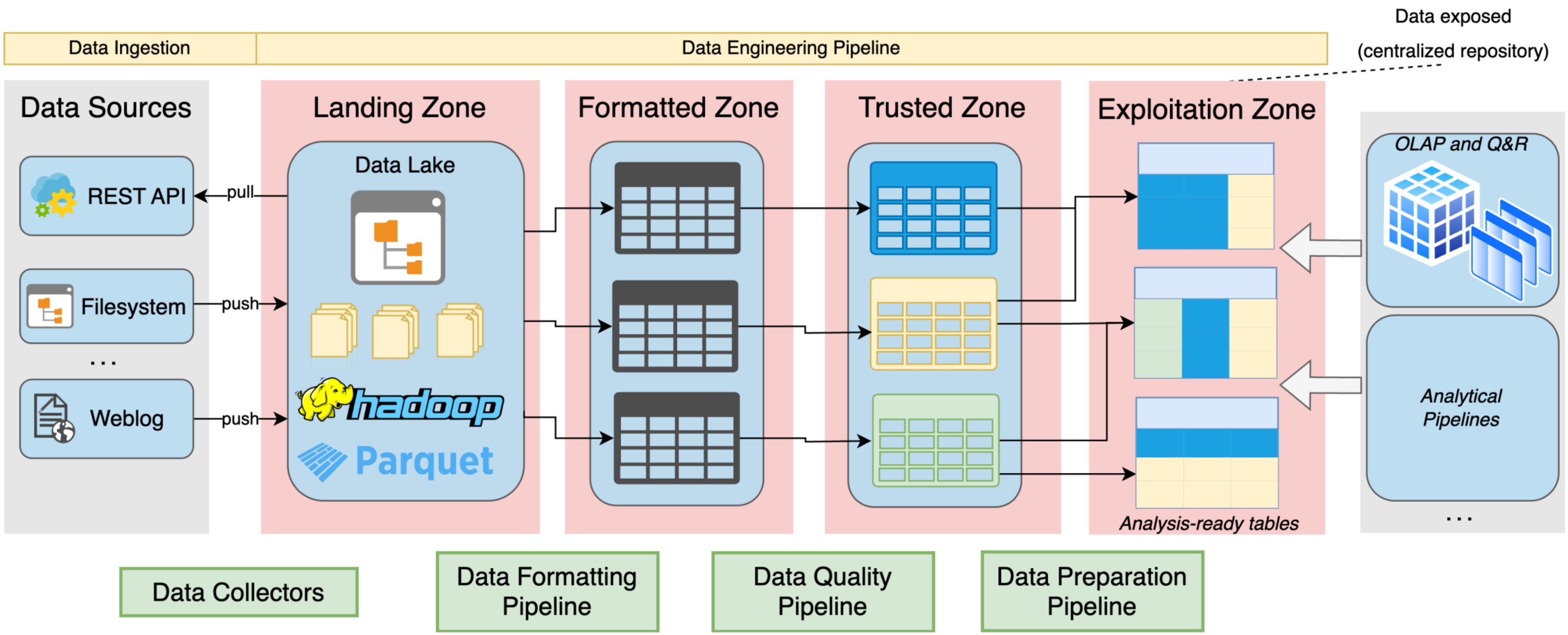
- Ingests and stores external data into the System
  - Data collection
  - Data Integration (standardization and data crossing)
    - Syntactic homogenization of data
    - Semantic homogenization of data
  - Data quality
    - Clean data / eliminate duplicates
- Expose a cleaned and centralized repository

The data analysis pipelines (one for every analytical need)

- Extract a data view from the centralized repository
- Feature engineering
- Specific pre-processing for the given analytical task at hand
  - Labeling
  - Data preparation specific for the algorithm chosen
- Create test and validation datasets
- Learn models (either descriptive stastitical analysis or advanced predictive models)
- Validate and interprete the model

The Data Engineering Pipeline

The Data Engineering Pipeline

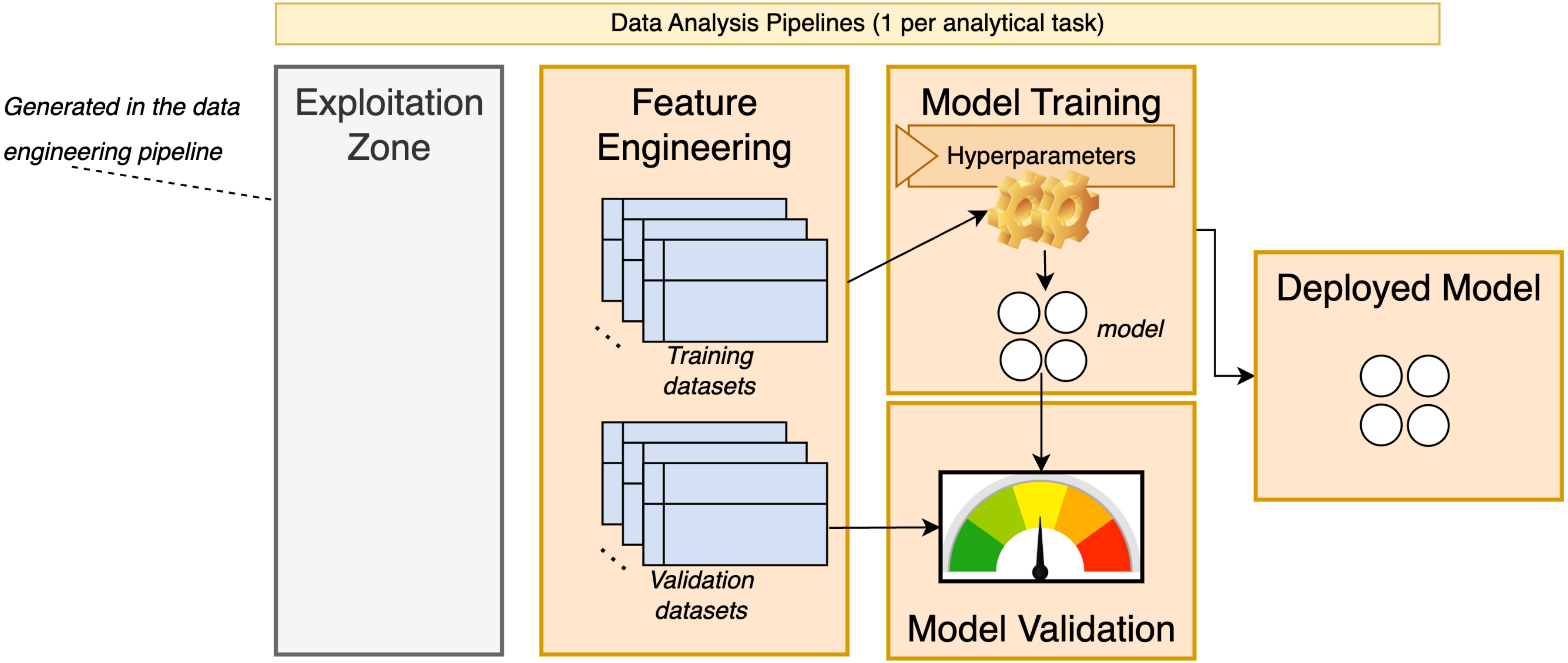


The Data Engineering Pipeline

- Data is ingested in the Landing Zone as it is produced (raw data)
  - The Data Lake stores files in a suitable format (SequenceFile, Avro, Parquet, …)
  - No data transformations are applied here
- Data is then homogenized, according to a canonical data model in the Formatted Zone (syntactic homogenization)
- Then, data quality is assessed and data cleaning processes are applied, storing the data in the Trusted Zone
- The Exploitation Zone exposes data ready to be consumed / analysed either by advanced analytical pipelines or external tools. This zone involves semantic homogenization:
  - Data integration: new data views are generated by combining the instances from the Formatted Zone. A view may serve one or several data analysis tasks. Relevantly, data integration spans data discovery, entity resolution, ad-hoc transformations and data loading into a target schema in a potentially different data model

The Data Analysis Pipelines

The Data Analysis Pipelines



The Data Analysis Backbone

Unlike the data management backbone, there is an analytical pipeline per analysis need. Thus, a project may define several analytical pipelines

- During feature generation, features are generated from the Exploitation Zone. The following tasks take place:
  - Data preparation rules, specific for the algorithm and kind of analysis, are applied
  - Labeling, if required, is also conducted here
  - As result, two corpus of datasets are generated: the training and validation datasets
- Model training requires choosing an algorithm and specifying the required algorithm hyperparameters, and this outputs a model
- Then, the generated model is validated according to some quality criteria (e.g., accuracy, recall, etc.)
- Out of the models generated, one is chosen to be deployed

Objectives of the project

Implementation of an end-to-end Data Science pipeline

Objectives

- The objective of the project is to implement and end-to-end data science pipeline
  - A Data Engineering pipeline processing at least **three** data sources
  - At least **two** Data Analysis pipelines
- You choose the data sources and the analytical tasks
  - Be ambitious!
- Some candidate data sources
  - Open Data Portals: OpenDataBCN, Dades Obertes L'Hospitalet, …
  - Third party APIs: Google Maps, TripAdvisor, Twitter, …
  - Publicly available: Kaggle, Data.World, AWS Data Marketplace, …

The Landing Zone

- Implement a Data Collector for each data source
  - Download a dataset
  - Convert it to a suitable format and give it a meaningful name
    - Importantly: the Data Collector should allow for **periodic executions**
- Decide on the organization of your Data Lake
  - No need to use HDFS, you can use Google Drive or your Local File System
- Several datasets might be ingested from one data source

The Formatted Zone

- Data in the Data Lake must be homogeneized into a common data format
  - The Relational Data Model
- You choose the technology and the design of the database
  - PostgreSQL
  - DuckDB
  - Delta Lake
  - …
- The Data Formatting Pipeline must be implemented using Spark / SparkSQL (either the SQL or the DataFrame API)
- As a rule of thumb, there should be one table per dataset

The Trusted Zone

- Three main tasks are conducted here
  - Identification of Data Quality rules on your datasets
  - An assessment of the Quality of the Data
  - Application of Data Cleaning processes (individually per dataset)
- Data quality rules can be expressed as Denial Constraints (explained in class)
- The Trusted Zone will store the same tables as the Formatted Zone where their quality has been improved
- The Data Quality Pipeline must be implemented using Spark / SparkSQL

The Exploitation Zone

- Here we will merge and integrate the different datasets in the Trusted Zone so they are ready to be consumed by the Data Analysis Pipelines
  - It might be possible that Data Reconciliation must be performed (an example will be provided)

The Data Analysis pipelines

- Implement at least two Data Analysis pipelines
- It can be any task and technology of your choice
  - Train a classification / regression model
  - Data visualization
  - Natural Language Processing
- Technologies
  - Scikit Learn
  - PyTorch
  - LLMs
  - Spark's MLlib
  - …

Organization

Teams

- Groups of 3 students
  - Create teams in LearnSQL

Timeline

- February 28th
  - Project statement
- March 14th
  - Follow-up: Data Collectors and Landing Zone
- March 21nd
  - Follow-up: Formatted and Trusted Zones
- March 28th
  - Follow-up: Exploitation Zone and (at least) one Data Analysis Pipeline
- April 10th (Thursday before the next lab class).
  - Final submission
- This dates are indicative of the expected progress. Follow-up sessions are not mandatory no attend

What to submit?

- All the code required to deploy your Data Science project
  - A single notebook
  - A set of notebooks
  - A set of Python scripts
- All design choices and non-obvious decisions **must be explained and documented**
  - In the notebooks themselves
  - As a companion document (max 5 pages)

Closing