UNIVERSITÀ DEGLI STUDI DI TRIESTE

Class 6: Decision-Making and Interfaces for ML Models

Master Course:

Data-driven Systems Engineering (ML Operations)

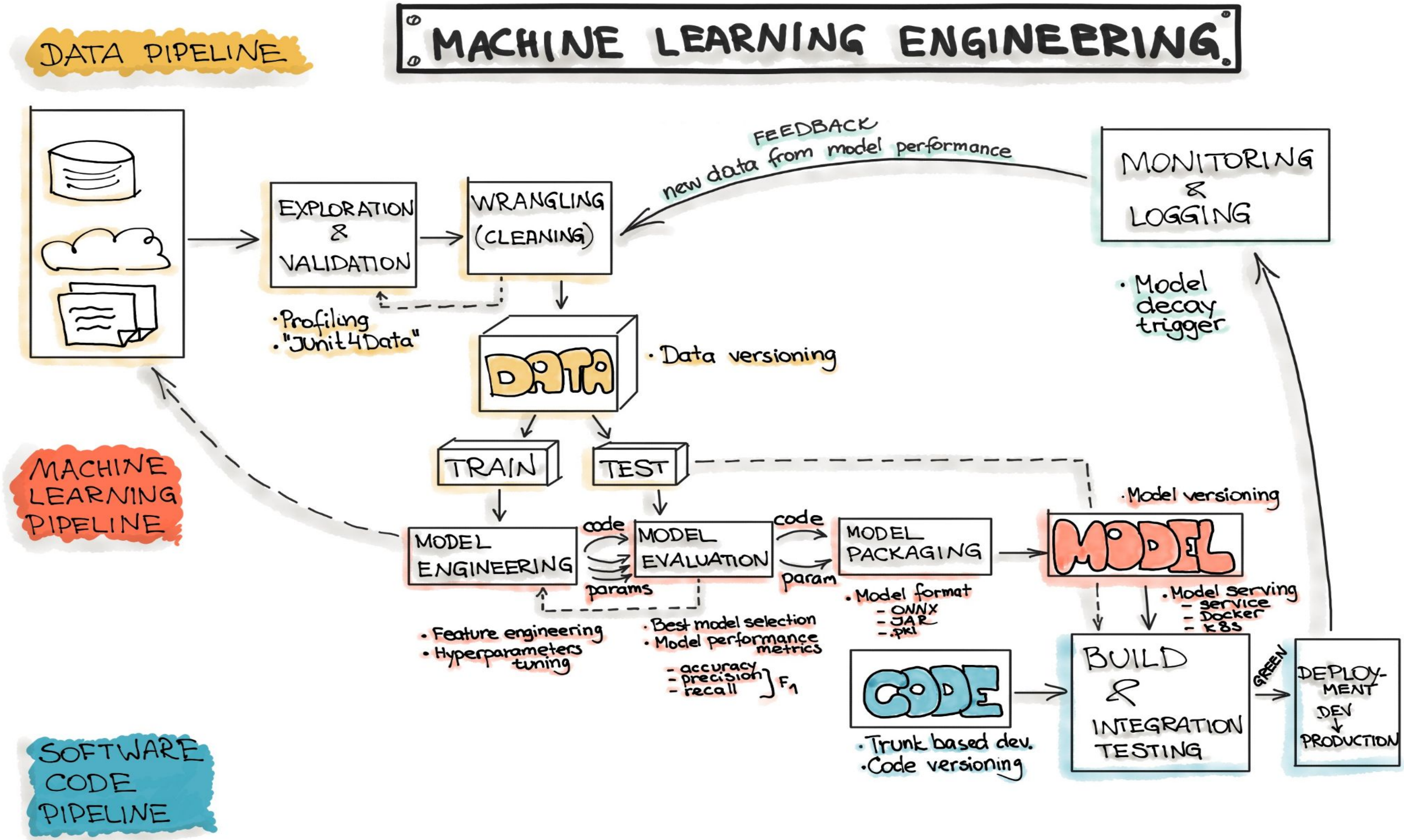
440MI and 305SM

Agenda

- Learning Goals
- From Models to Decisions
- Visualizing Model Predictions
- Interfaces for Machine Learning
- Introduction to Streamlit
- Live Demo: Building a Streamlit
- Discussion and Best Practices

Learning Goals

- Understand how predictions from ML models can support decision-making.
- Visualize model outputs interactively.
- Build simple web interfaces for ML using Streamlit.
- Translate technical predictions into actionable insights.


DATA PIPELINE
MACHINE LEARNING ENGINEERING
EXPLORATION & VALIDATION
WRANGLING (CLEANING)
FEEDBACK
new data from model performance
MONITORING & LOGGING
·Profiling
·"JUnit4Data"
DATA
·Data versioning
·Model decay trigger
TRAIN
TEST
MACHINE LEARNING PIPELINE
·Model versioning
MODEL ENGINEERING
code
params
MODEL EVALUATION
code
param
MODEL PACKAGING
MODEL
·Feature engineering
·Hyperparameters tuning
·Best model selection
·Model performance metrics
- accuracy
- precision
- recall
F1
·Model format
- ONNX
- JAR
- .pkl
·Model serving
- service
- Docker
- K8s
CODE
BUILD & INTEGRATION TESTING
GREEN
DEPLOYMENT
DEV
PRODUCTION
·Trunk based dev.
·Code versioning
SOFTWARE CODE PIPELINE

From Models to Decisions

A machine learning model is only useful when its output is **interpreted and acted upon**.
For example:

- A **fraud detection model** triggers a *review process*.
- A **credit risk model** informs *loan approval*.
- A **classification model** in healthcare supports *diagnosis decisions*.

Open points:

- Predictions are not final decisions — they are **decision-support tools**.
- Importance of **thresholds**, **confidence**, and **cost of false decisions**.
- Role of **human-in-the-loop** systems.

Visualizing Model Predictions

- Prediction Distribution
  a. Show predicted probabilities (e.g., fraud probability, disease likelihood).
  b. Use histograms, calibration plots, or scatter plots.
- Confusion Matrix and Thresholds
  a. Demonstrate how different thresholds affect outcomes.
  b. Visualization idea: a slider that adjusts decision threshold dynamically.
- Feature Importance
  a. Visualize feature_importances_ (tree models) or SHAP values.
  b. Help users interpret why a prediction was made.

Interfaces for Machine Learning

Objective: Enable interaction between users and models.

Common interface types:

Type	Description	Example Tools
Web Apps	Interactive apps for model exploration	Streamlit, Gradio, Dash
Dashboards	Static or semi-interactive results	Power BI, Tableau
APIs	Backend-only services	Flask, FastAPI
Embedded Interfaces	Integration in other software	ERP systems, Chatbots

- Improves accessibility of ML results.
- Encourages collaboration between data scientists and stakeholders.
- Provides real-time feedback on model performance.

Introduction to Streamlit

What is Streamlit?

A Python-based open-source framework for creating interactive data apps and dashboards quickly.

Core Features:

- Build interfaces with pure Python (no HTML/JS).
- Supports interactive widgets (sliders, text boxes, buttons).
- Ideal for model demos, dashboards, and educational apps.

Installation:

```
pip install streamlit
```

Run an app:

```
streamlit run app.py
```

Streamlit

Live Demo: ML Decision Interface

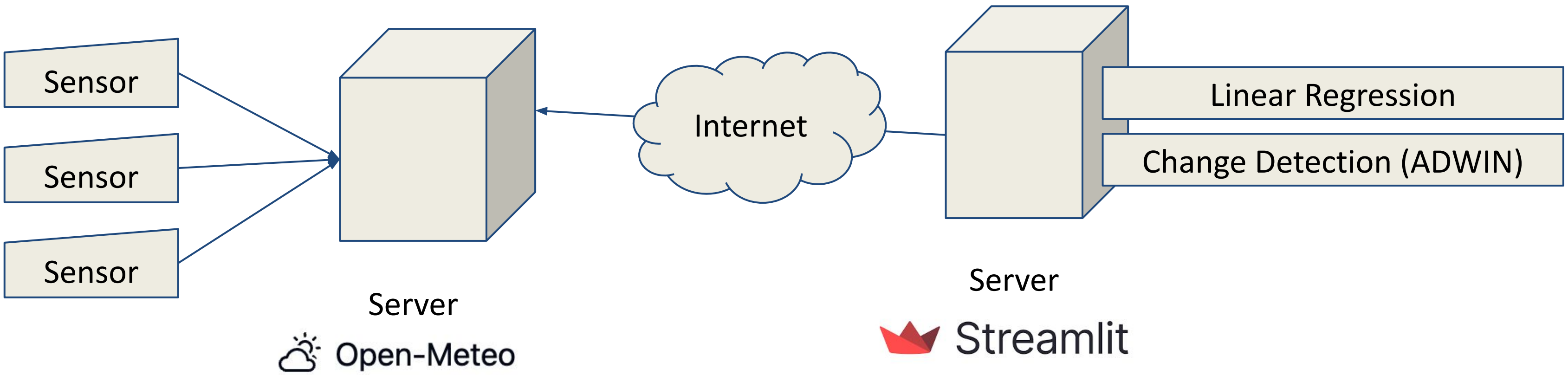
Streamlit Example Code

```
import streamlit as st
import pickle
import numpy as np

# Load model
model = pickle.load(open('model.pkl', 'rb'))

st.title("Credit Card Fraud Detection App")

st.sidebar.header("Transaction Input Features")
amount = st.sidebar.number_input("Transaction Amount (€)", min_value=0.0)
v1 = st.sidebar.slider("Behavior Score (V1)", -5.0, 5.0, 0.0)
v2 = st.sidebar.slider("Risk Factor (V2)", -5.0, 5.0, 0.0)

input_data = np.array([[v1, v2, amount]])
prediction = model.predict(input_data)[0]
prob = model.predict_proba(input_data)[0][1]

st.subheader("Prediction Result")
if prediction == 1:
    st.error(f"⚠️ Fraud detected (probability: {prob:.2f})")
else:
    st.success(f"✅ Legitimate transaction (probability: {prob:.2f})")
```

Project:

Online Weather Prediction with Streamlit and River

Demonstrating Real-Time Machine Learning and Concept Drift Detection



Project: **Online Weather Prediction with Streamlit and River**

- Traditional ML models are *trained offline* — they can't adapt quickly to new data.
- In real environments (weather, finance, IoT), **data arrives continuously**.
- Models must **learn incrementally** and detect **changes in patterns** (*concept drift*).

Goal:
Predict temperature in real time using **wind speed** data, continuously updating the model as new data arrives.

Project: **Online Weather Prediction with Streamlit and River**

Component	Description
Data Source	Open-Meteo API provides live weather data (temperature, wind speed).
Model	Online Linear Regression from River (`LinearRegression + StandardScaler`).
Metrics	MAE (Mean Absolute Error) for performance tracking.
Drift Detection	ADWIN algorithm detects changes in error distribution.
Interface	Streamlit app for real-time visualization and control.

References:

- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
- Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. Springer.
- Mitchell, T. M. (1997). Machine Learning. McGraw-Hill Education.
- Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. Communications of the ACM, 55(10), 78–87.
- Jordan, M. I., & Mitchell, T. M. (2015). Machine Learning: Trends, Perspectives, and Prospects. Science, 349(6245), 255–260.