# Assignment A1 — From ML to FL

CS-E4740 Federated Learning

**Read Carefully**

**There is no submission.** Your work is assessed via a multiple-choice quiz.

The quiz will test the following quantities, which you must compute and print:

1. **System statistics:**
   - number of nodes, i.e., the number of Finnish Meteorological Institute (FMI) stations with observations in the CSV.
2. **Local model performance:**
   - average validation error across all nodes.
3. **Global model results:**
   - global training error (using average squared error loss),
   - global validation error.

## Python Environment

To ensure reproducibility, the following environment is recommended:

- Python $\geq$ 3.11
- `numpy` == 2.3.4
- `scikit-learn` == 1.7.2 (for `LinearRegression`)
- `networkx` == 3.5 (for representing the node set)
- `matplotlib` == 3.10.7 (optional; plotting not required)

Some useful documentation:

- `scikit-learn` documentation for implementing linear regression
- `NetworkX` documentation for storing graphs
- `scikit-learn` documentation of computing mean squared error (MSE)

You can verify installed package versions via:

```
python -m pip show numpy scikit-learn networkx matplotlib
```

## 1 Purpose

This assignment revolves around the transition from basic machine learning (ML) to collaborative federated learning (FL). In particular, using weather observations collected at FMI stations, you will compare:

- a **single global model** trained on pooled data, and
- the **average performance of local models** trained independently.

## 2 Data

You must use the following CSV file:

https://github.com/alexjungaalto/FederatedLearning/blob/main/Edition2026/assignments/fmidata.csv

Each row corresponds to **one day at one FMI weather station**. The CSV file contains the following columns:

| | |
|---|---|
| `station` | station name (string) |
| `lat` | latitude (float) |
| `lon` | longitude (float) |
| `day` | date in `YYYY-MM-DD` format |
| `tmax` | daily maximum temperature (float); will be used as label |
| `tmin` | daily minimum temperature (float); will be used as feature |

All temperatures are measured in degrees Celsius (°C). **You must not modify or replace the dataset**.

## 3 Federated learning network (FL network)

You must construct an FL network, with graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose:

- nodes $\mathcal{V}$ represent the FMI weather stations,
- edge set is **empty**, $\mathcal{E} = \emptyset$, i.e., there is no collaboration between the nodes.

The code snippet below demonstrates how to implement $\mathcal{G}$ using a `networkx.Graph`:

```
import networkx as nx

G = nx.Graph()    # empty undirected graph
```

The following snippet illustrates the structure of the CSV file:

```
station,lat,lon,day,tmax,tmin
Alajärvi Möksy,63.08898,24.26084,2025-12-31,-15.3,-22.0
Alajärvi Möksy,63.08898,24.26084,2026-01-01,-16.5,-22.3
Porvoo Kilpilahti satama,60.30373,25.54916,2026-01-01,-3.2,-7.8
```

The FL network can be implemented as an undirected `networkx.Graph` object whose node set consists of the station identifiers. No edges are added to the graph.

This FL network represents an federated learning system (FL system) without collaboration, i.e., local models are trained independently and no information is exchanged between nodes.

## 4 Local Models

For each node (FMI station), train a separate local linear model

$$t_{\max} \approx w_0 + w_1 t_{\min}.$$

Each local model must be implemented using the `LinearRegression` class from the `scikit-learn` library, trained on the local training set of the corresponding FMI station.

**Important:** Make sure that each local model includes an intercept (bias) term, corresponding to $w_0$ in the expression above.

## 5 Constructing training set and validation set

For each station:

- training set: all but the *the latest available calendar day* for that station.
- validation set: the *the latest available calendar day* for that station.

## 6 Global model (Centralized Baseline)

Construct a global baseline model by:

- pooling all local training sets across FMI stations,
- training a single linear model (you must use again an intercept/bias term!)
- pooling all local validation sets across FMI stations,
- computing training error and validation error (using average squared error loss)