CS-E4740 - Federated Learning

# Trustworthy FL

Assoc. Prof. Alexander Jung

Spring 2025

**Playlist**

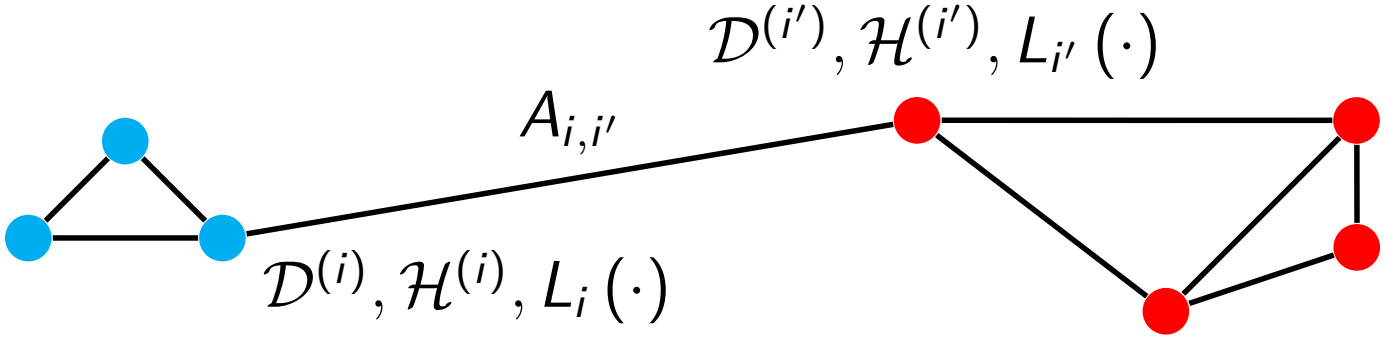
**Glossary**

**Course Site**

# Outline

# Table of Contents

# FL Network as Mathematical Model for FL




- An FL network consists of devices $i = 1, \ldots, n$.
- Some $i, i'$ connected by edge with the weight $A_{i,i'} > 0$.
- Device $i$ **generates data** $\mathcal{D}^{(i)}$ and **trains model** $\mathcal{H}^{(i)}$.
- Data $\mathcal{D}^{(i)}$ used to construct loss func. $L_i(\cdot)$.

# GTV Minimization (for Parametric Models)

We train local models in a collaborative fashion by solving

$$\min_{\mathbf{w}^{(1)},\ldots,\mathbf{w}^{(n)}} \sum_{i=1}^{n} L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{\{i,i'\}\in\mathcal{E}} A_{i,i'} \left\|\mathbf{w}^{(i)} - \mathbf{w}^{(i')}\right\|_2^2 \quad \text{(GTVMin)}.$$

- Solution consists of learnt model params. $\widehat{\mathbf{w}}^{(i)}$.
- Tuning parameter $\alpha \geq 0$ controls clustering of $\widehat{\mathbf{w}}^{(i)}$.
- For $\alpha = 0$, GTVMin reduces to separate ERM for each $i$.
- Increasing $\alpha$ makes $\widehat{\mathbf{w}}^{(i)}$ more similar across nodes $i$.

# FL Algorithms

A core computational step of FL algorithms is

$$\mathbf{w}^{(i,k+1)} = \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right].$$

- This is precisely the update of FedRelax.
- FedGD replaces $L_i\left(\mathbf{w}^{(i)}\right)$ with

$$\eta \left( \nabla L_i\left(\mathbf{w}^{(i',k)}\right) \right)^T \left(\mathbf{w}^{(i)} - \mathbf{w}^{(i,k)}\right) + \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i,k)} \right\|_2^2.$$

- FedAvg uses star graph with centre (server) $L_i(\cdot) = 0$.

# FL via Regularization

Note that

$$\mathbf{w}^{(i,k+1)} = \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right] \quad \text{(A)}.$$

is nothing but regularized ERM with specific choice for penalty.

The penalty $\alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2$ uses the neighbours model params. $\mathbf{w}^{(i',k)}$, for $i' \in \mathcal{N}^{(i)}$.

# Basic ML (Regularized ERM)

```
X, y = read_data()
model = LinearRegression()
model.fit(X, y)
```

$$\widehat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} \left[ (1/|\mathcal{D}|) \sum_{(\mathbf{x},y)\in\mathcal{D}} L\big((\mathbf{x},y), h^{(\mathbf{w})}\big) + \alpha \mathcal{R}\{\mathbf{w}\} \right]$$

For some $L(\cdot), \mathcal{R}\{\ \}$, this is equivalent to data augmentation[1]

$$\widehat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^d} (1/|\mathcal{D}'|) \sum_{(\mathbf{x},y)\in\mathcal{D}'} L\big((\mathbf{x},y), h^{(\mathbf{w})}\big)$$

with augmented dataset $\mathcal{D}' = \mathcal{D} \cup \{\ldots\}$.

[1] see Sec. 7.3 of AJ, "Machine Learning: The Basics," Springer, 2022. preprint: https://mlbook.cs.aalto.fi

# From ML to FL via Regularization

node $i = 1$, IP: 192.168.0.1

```
X, y = read_data()
model = LinearRegression()
model.fit(X, y)
```

$$\mathbf{w}^{(1,k+1)} = \underset{\mathbf{w}^{(1)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_1\left(\mathbf{w}^{(1)}\right) + \alpha A_{1,2} \left\| \mathbf{w}^{(1)} - \mathbf{w}^{(2,k)} \right\|_2^2 \right].$$

node $i = 2$, IP: 192.168.0.3

```
X, y = read_data()
model=LogisticRegression()
model.fit(X, y)
```

$$\mathbf{w}^{(2,k+1)} = \underset{\mathbf{w}^{(2)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_2\left(\mathbf{w}^{(2)}\right) + \alpha A_{1,2} \left\| \mathbf{w}^{(2)} - \mathbf{w}^{(1,k)} \right\|_2^2 \right].$$

# Learning Goals

After completing this module, you know

- key requirements for trustworthy AI,
- how to increase robustness of GTVMin,
- how to implement privacy protection,
- a mathematical model for the explainability of FL.

# Table of Contents

# FL and AI

- FL systems can be used to implement AI services.
- Users of AI service consume predictions.
- Predictions are delivered by personalized (local) model.
- Thus, regulations for AI are also regulations for FL.

# Key Requirements for the EU

European Union defined seven key requirements for trustworthy AI[2]

- KR1 - Human Agency and Oversight
- KR2 - Technical Robustness and Safety
- KR3 - Privacy and Data Governance
- KR4 - Transparency
- KR5 - Diversity, Non-discrimination and Fairness
- KR6 - Societal and Environmental Well-Being
- KR7 - Accountability

[2] High-Level Expert Group on Artificial Intelligence, "Ethics guidelines for trustworthy AI," Tech. Rep., European Commission, April 2019.

# Key Requirements for AI beyond EU

- Australia's 8 Artificial Intelligence (AI) Ethics Principles.[3]
- AI ethical principles of US DoD.[4]
- Recommendation on the Ethics of AI by UNESCO.[5]
- AI principles of the OECD.[6]

---

[3] https://www.industry.gov.au/publications/australias-artificial-intelligence-ethics-principles/australias-ai-ethics-principles

[4] D. Oniani, et al. Adopting and expanding ethical principles for generative artificial intelligence from military to healthcare. npj Digit. Med. 6, 225 (2023). https://doi.org/10.1038/s41746-023-00965-x

[5] https://www.unesco.org/en/articles/recommendation-ethics-artificial-intelligence

[6] https://www.oecd.org/en/topics/sub-issues/ai-principles.html

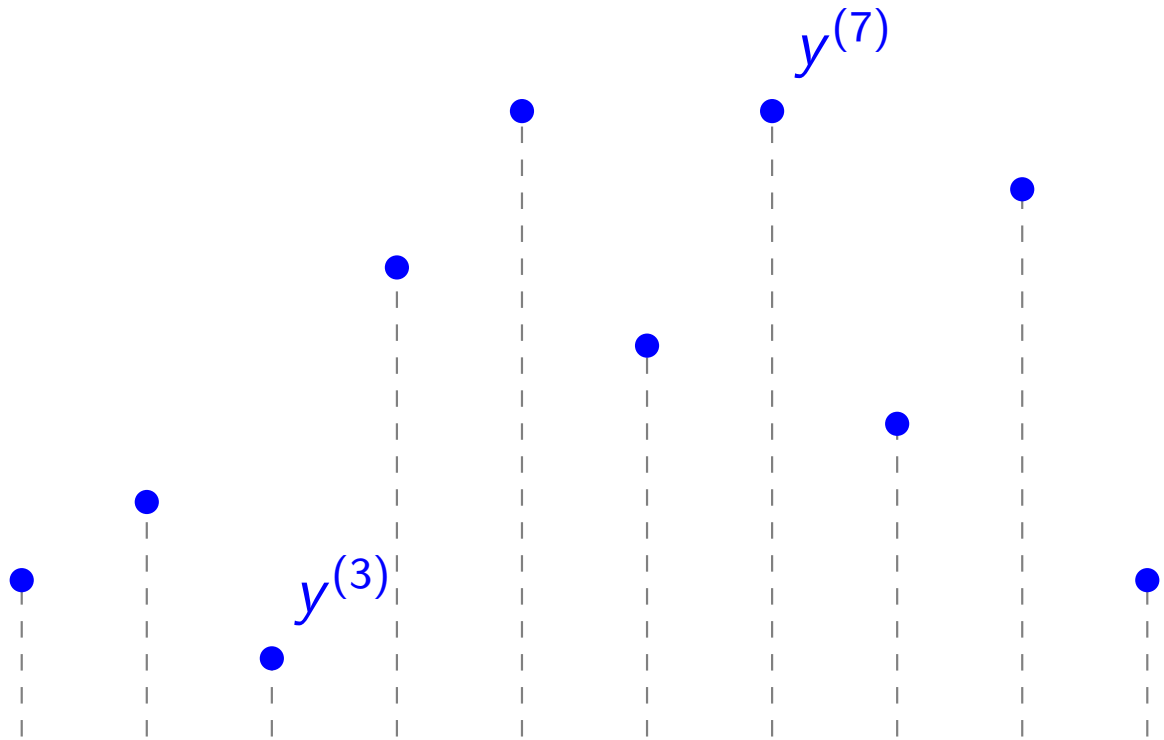
# Table of Contents

# A Very Simple FL Network

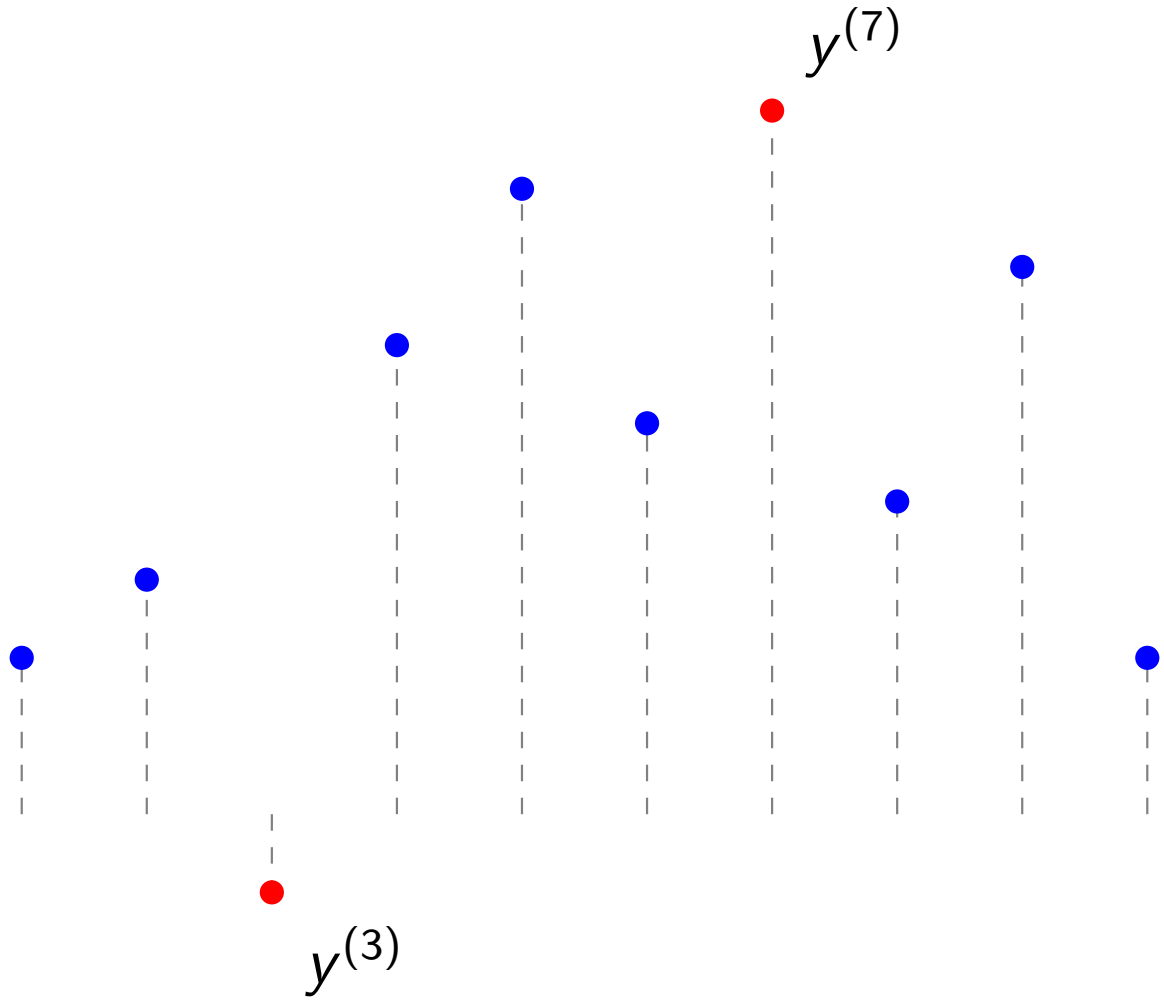
- Consider a FL network with single node $i = 1$, which
- carries the local dataset $\mathcal{D}^{(1)} := \left\{ y^{(1)}, \ldots, y^{(m_1)} \right\}$ and
- trains local model with parameter $w^{(1)}$ by
- minimizing average squared error loss

$$L_1\left(w^{(1)}\right) := (1/m_1) \sum_{r=1}^{m_1} \left(w^{(1)} - y^{(r)}\right)^2.$$

# Is the Local Dataset Reliable?



# Data Poisoning




Adversary changes (poisons) $\eta m_1$ data points of $\mathcal{D}^{(1)}$.

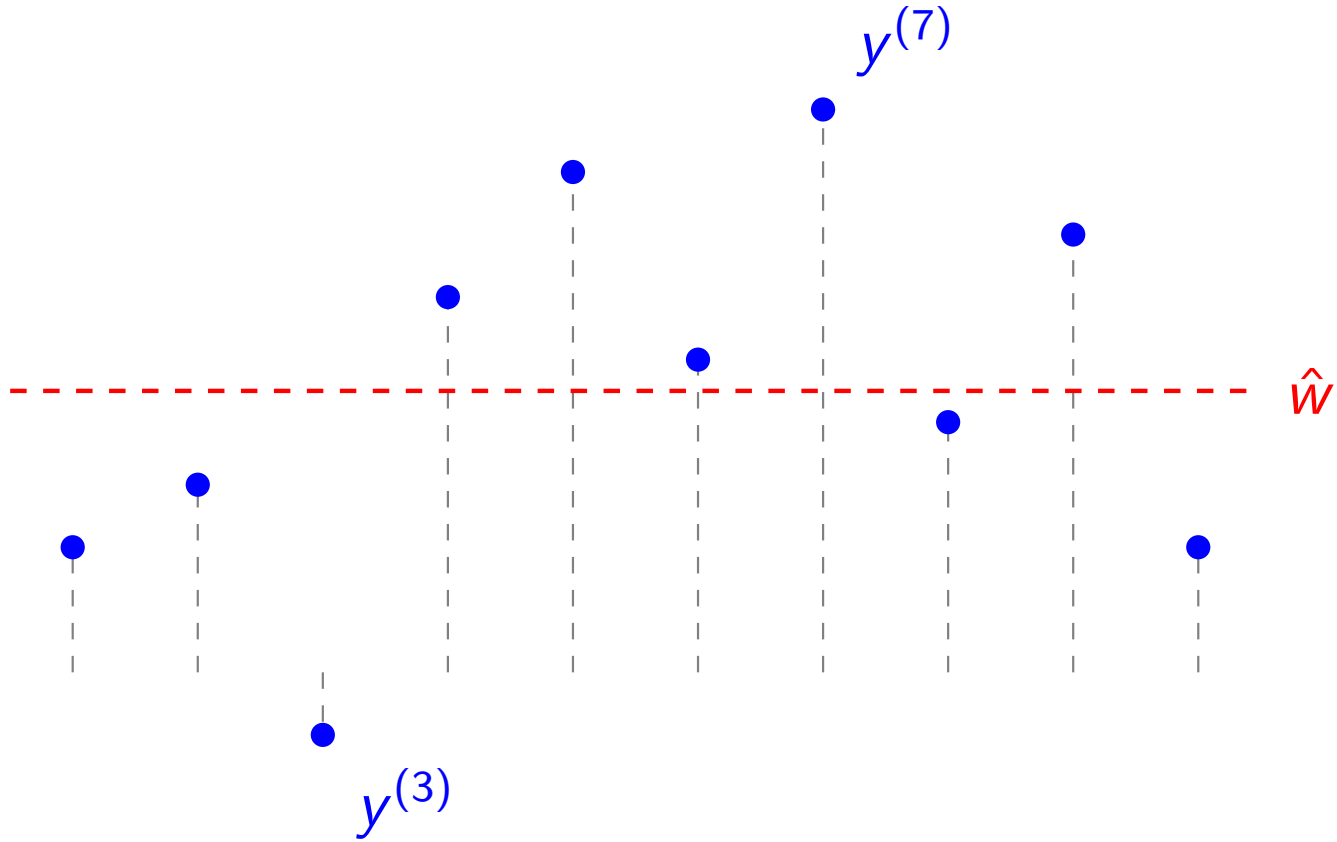
# Robust Learning

- Assume at most $\eta m_1$ data points are poisoned.
- How to reliably train a model despite data poisoning?
- Optimal approach depends on poisoning strategy:
  - **Oblivious poisoning:** strategy fixed before-hand.[7]
  - **Adaptive poisoning:** based on stat. of clean dataset.[8]
- Adaptive poisoning is more harmful.

[7] M. Chen et al., A general decision theory for Huber's $\varepsilon$-contamination model, Electron. J. Stat., 2016.

[8] G. Lugosi et al., Robust multivariate mean estimation: The optimality of trimmed mean, Ann. Statist., 2021.

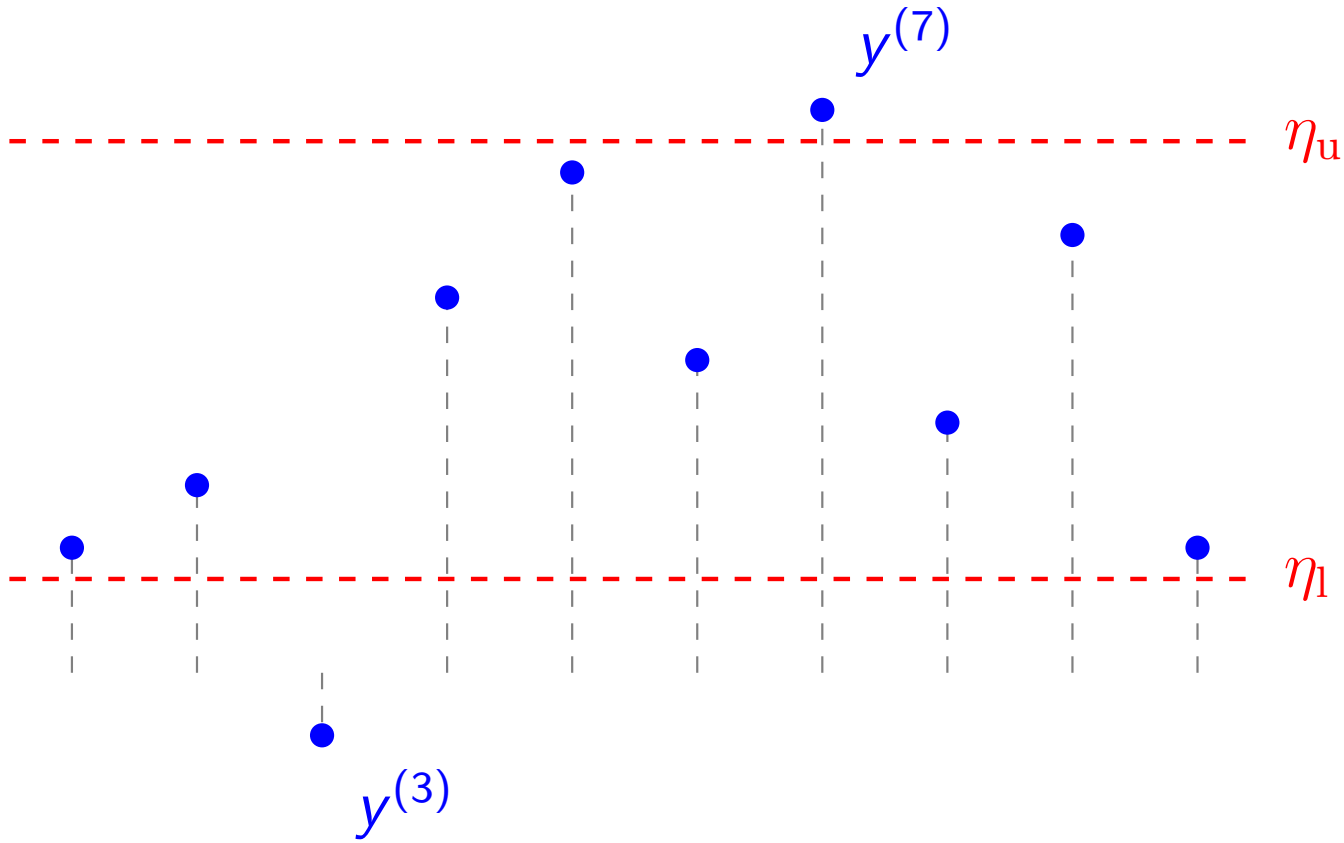
# Median




Median $\hat{w} \in \underset{w}{\operatorname{argmin}} \sum_{r=1}^{m_1} \left| y^{(r)} - w \right|$ is a special case of Huber regression.[9]

[9] P.J. Huber. "Robust Estimation of a Location Parameter." Ann. Math. Statist. 35 (1) 73 - 101, March, 1964. https://doi.org/10.1214/aoms/1177703732

# Trimmed Mean




$$\hat{w} = \sum_{r=1}^{m_1} \phi\left(y^{(r)}\right) \text{ with } \phi(y) := \begin{cases} \eta_{\mathrm{u}} & \text{for } y \geq \eta_{\mathrm{u}} \\ y & \text{for } \eta_{\mathrm{l}} \leq y \leq \eta_{\mathrm{u}} \\ \eta_{\mathrm{l}} & \text{for } y \leq \eta_{\mathrm{l}}. \end{cases}^{10}$$

[10] G. Lugosi et al., Robust multivariate mean estimation: The optimality of trimmed mean, Ann. Statist., 2021.

# Robust FL

A core computational step of FL algorithms is

$$\mathbf{w}^{(i,k+1)} = \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right].$$

To make it more robust, we can

- use $\left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2$ instead of $\left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2$,[11]
- detect anomalous $\mathbf{w}^{(i',k)}$ (similar to trimmed mean).[12]

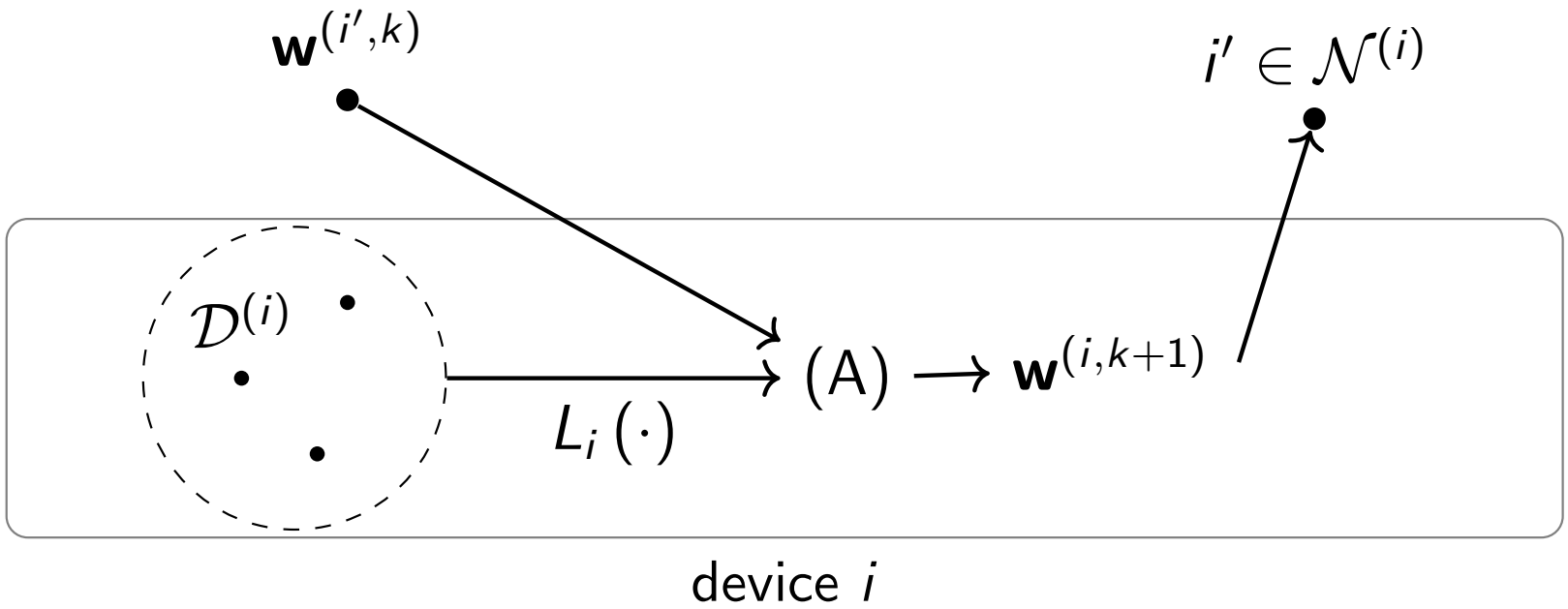
[11] K. Pillutla,et.al., "Robust Aggregation for Federated Learning," in IEEE Trans. on Sig. Proc., 2022.

[12] P. Blanchard, et.al., Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent, NeurIPS 2017

# Table of Contents

# FL from the Perspective of Node $i$

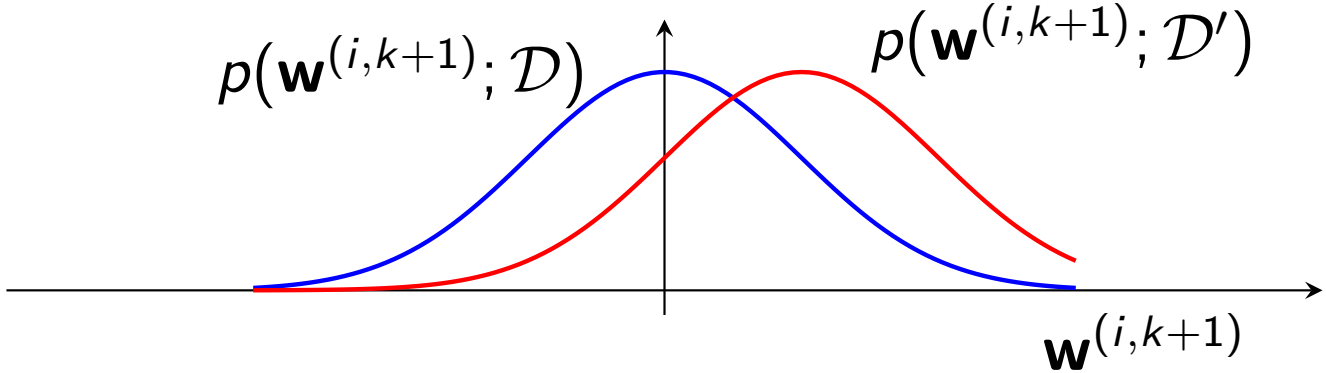
$$\mathbf{w}^{(i,k+1)} = \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right] \quad \text{(A)}.$$




How much sensitive info. does $\mathbf{w}^{(i,k+1)}$ reveal about $\mathcal{D}^{(i)}$?

# Measuring Privacy Leakage

- Consider node $i$ with local dataset $\mathcal{D}^{(i)}$.
- FL algo. computes $\mathbf{w}^{(i,k+1)}$ and shares it with others.
- Interpret $\mathbf{w}^{(i,k+1)}$ as RV with pdf $p(\mathbf{w}^{(i,k+1)}; \mathcal{D}^{(i)})$.
- Can one distinguish two instances $\mathcal{D}, \mathcal{D}'$ of $\mathcal{D}^{(i)}$?

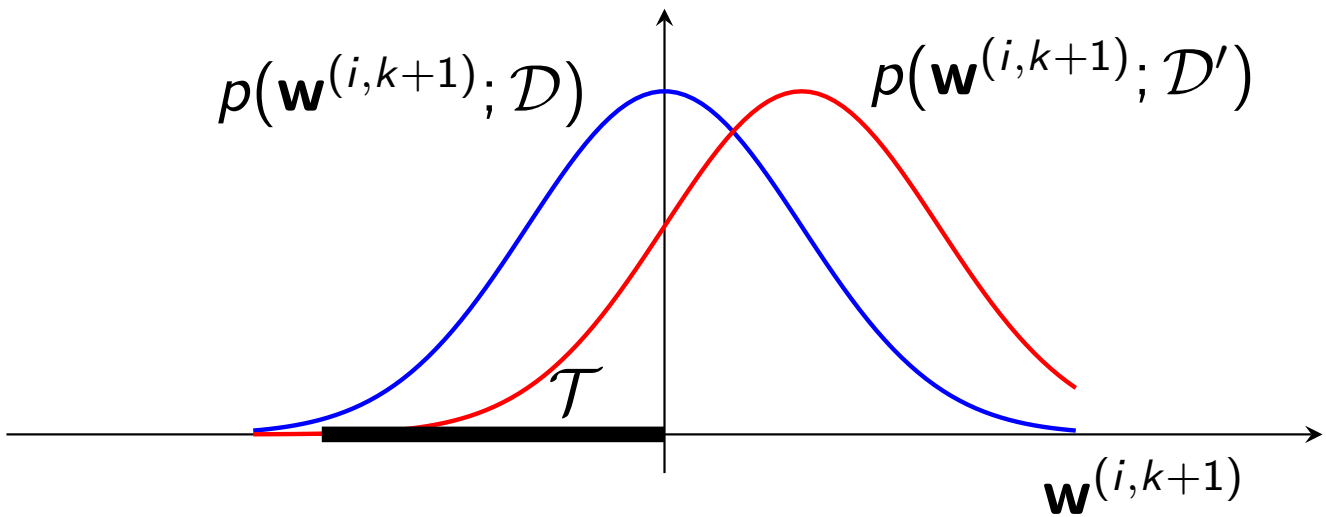



Neighbouring instances $\mathcal{D} \sim \mathcal{D}'$ differ in a single sensitive attribute.

# Differential Privacy (DP)

The update at node $i$ is $(\varepsilon, \delta)$-differentially private if for any measurable set $\mathcal{T}$ and any two neighbouring datasets $\mathcal{D}, \mathcal{D}'$

$$\text{Prob}\{\mathbf{w}^{(i,k+1)} \in \mathcal{T}\} \leq \exp(\varepsilon)\text{Prob}\{\mathbf{w}^{(i,k+1)} \in \mathcal{T}\} + \delta.^{13}$$




Smaller values of $\varepsilon$ and $\delta$ in DP imply lower privacy leakage.

[13] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," 2014.

# Ensuring DP

- Consider the update at node $i$ by

$$\mathcal{A}(\mathcal{D}) := \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right].$$

- Local loss $L_i\left(\mathbf{w}^{(i)}\right)$ is constructed from local dataset $\mathcal{D}$.
- Ensure DP by adding zero-mean noise, $\mathcal{A}(\mathcal{D}) + \text{noise}$.
- Noise strength chosen inversely to sensitivity

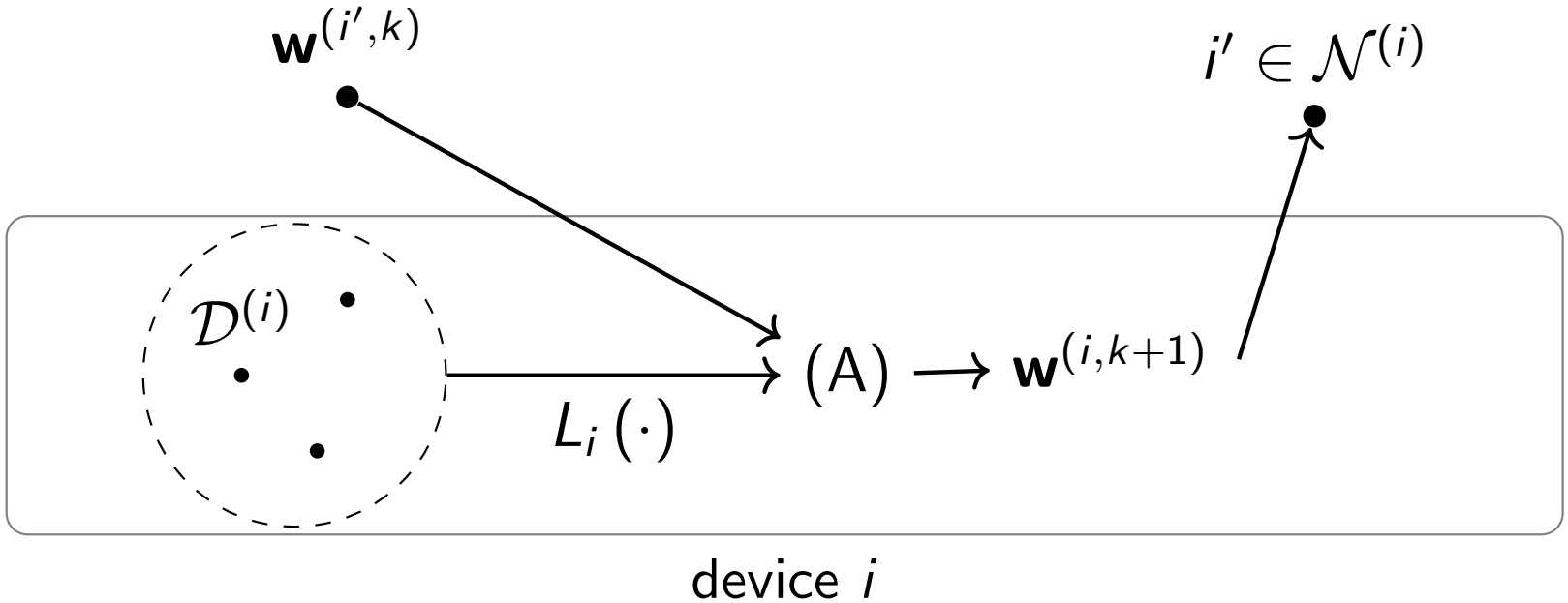
$$\max_{\mathcal{D} \sim \mathcal{D}'} \left\| \mathcal{A}(\mathcal{D}) - \mathcal{A}(\mathcal{D}') \right\|_2 .$$

- Sensitivity depends on shape of $L_i(\cdot)$, $\mathcal{N}^{(i)}$ and $\alpha$.

# Table of Contents

# FL from the Perspective of Node $i$

$$\mathbf{w}^{(i,k+1)} = \underset{\mathbf{w}^{(i)} \in \mathbb{R}^d}{\operatorname{argmin}} \left[ L_i\left(\mathbf{w}^{(i)}\right) + \alpha \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i',k)} \right\|_2^2 \right] \quad \text{(A)}.$$




How explainable is the model with params. $\mathbf{w}^{(i,k+1)}$ to the user of device $i$?

# Measuring Subjective Explainability

- Consider a trained model (or learnt hypothesis) $\hat{h}$.
- User provides labels $u(\mathbf{x})$ for data points in some $\mathcal{D}^{(\mathrm{u})}$.
- Compare user labels with predictions $\hat{h}(\mathbf{x})$ by computing

$$\mathcal{E} := \left(1/\left|\mathcal{D}^{(\mathrm{u})}\right|\right) \sum_{\mathbf{x} \in \mathcal{D}^{(\mathrm{u})}} \left(\hat{h}(\mathbf{x}) - u(\mathbf{x})\right)^2.$$

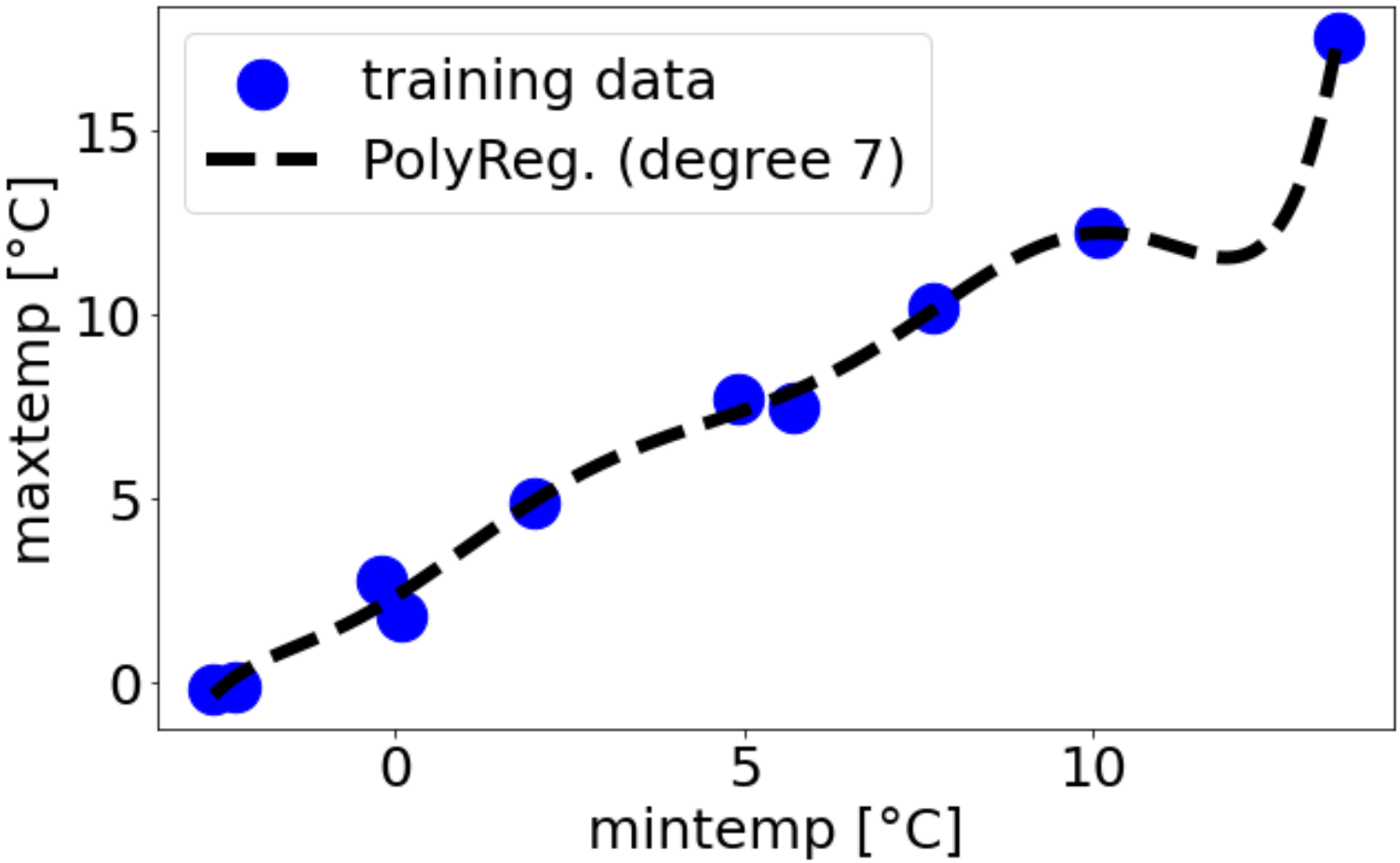
- Use $\mathcal{E}$ to measure subjective explainability of $\hat{h}$.
- Ensure subjective explainability by using $\mathcal{E}$ as regularizer.

# Example: Explainable Weather Prediction

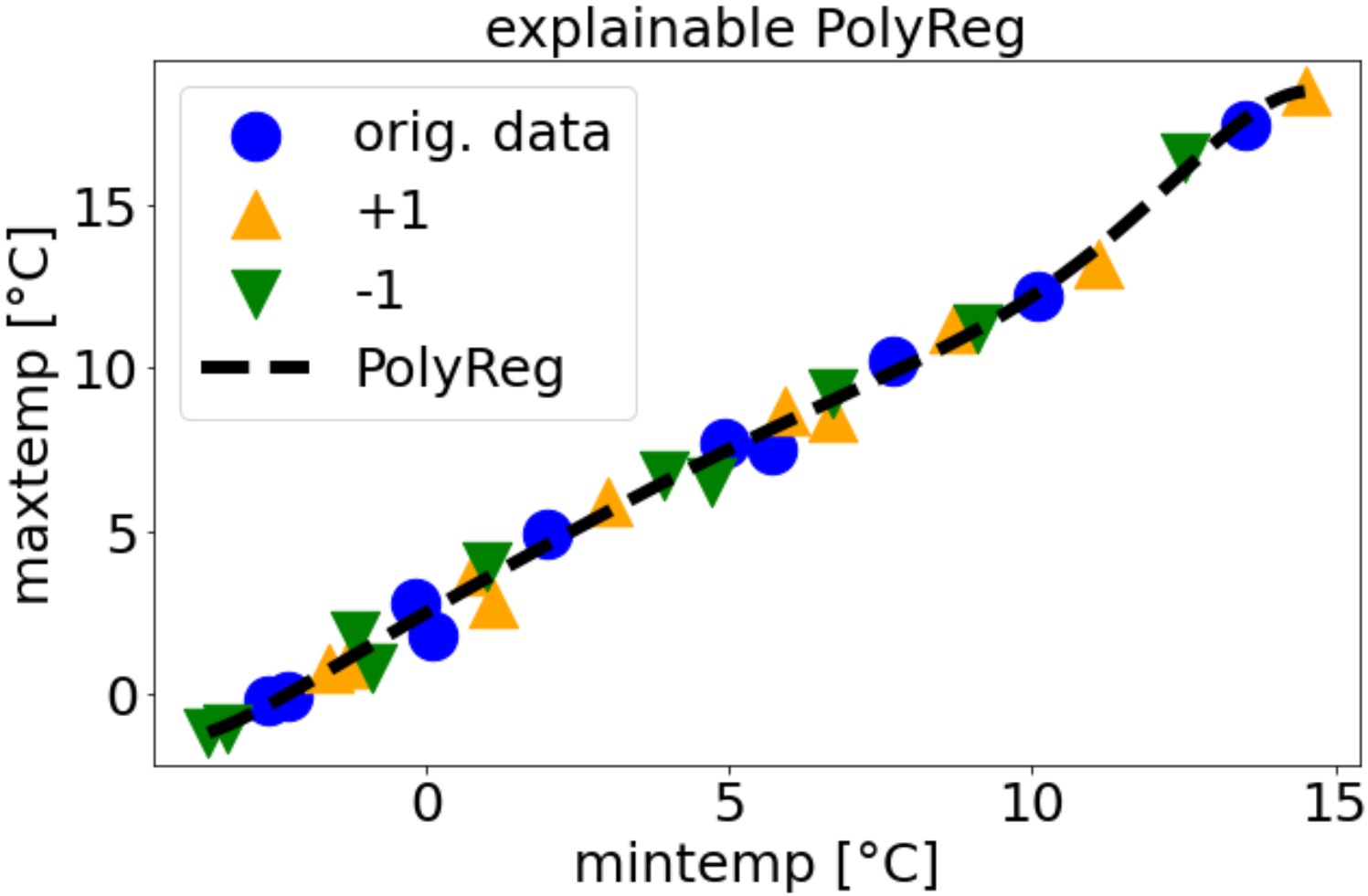
- Consider a day with min. tmp. $x$ and max. tmp. $y$.
- Train a polynomial regression model to predict $y$ from $x$.
- Intuition: *If x changes by +/- 1 degree, so does y!*
- Create $\mathcal{D}^{(\mathrm{u})}$ by adding $\delta \in \{-1, 1\}$ to $x$ in train. set.
- User label $u(x + \delta) = y + \delta$.
- Train polyn. regression model by using $\mathcal{E}$ as regularizer.

Python script for the example:

# Example: Polynomial Regression w/o Explainability



# Example: Polynomial Regression w Explainability



# What's Next?

This was the final of our six core modules.

You will next implement concepts taught during the modules in your FL project.

Aalto students submit project report via MyCourses. Others submit via EasyChair.

# Further Resources

- **YouTube:** @alexjung111
- **LinkedIn:** Alexander Jung
- **GitHub:** alexjungaalto