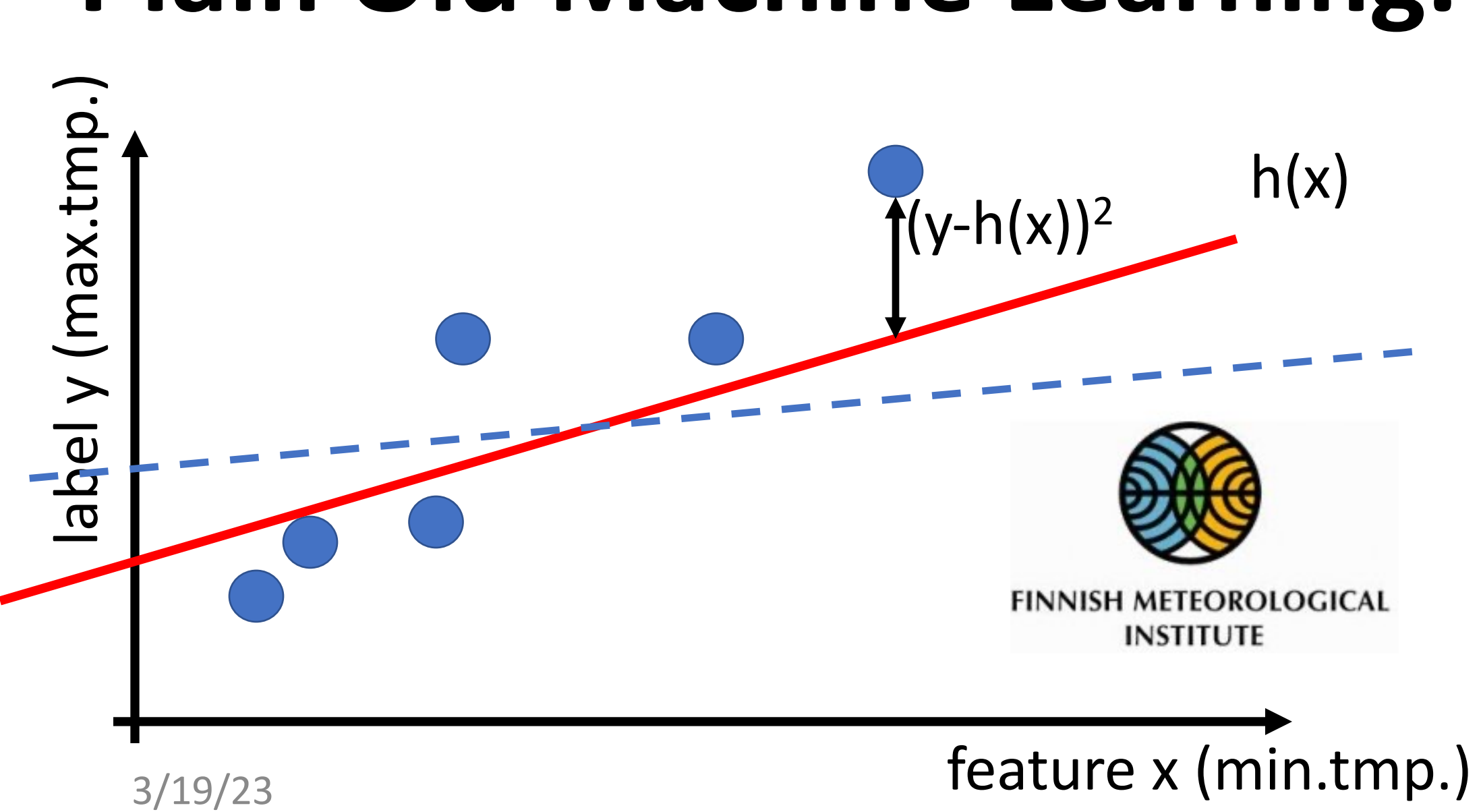
CS-E4740 Federated Learning

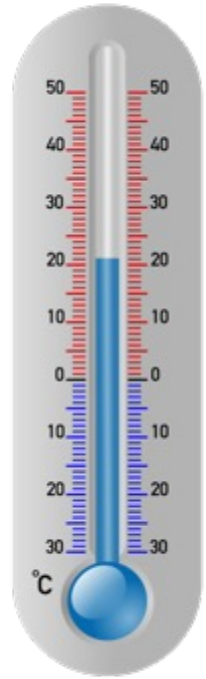
"Network Models"

Dipl.-Ing. Dr. techn. Alexander Helmut Jung

What are main components of ML and how are they combined?

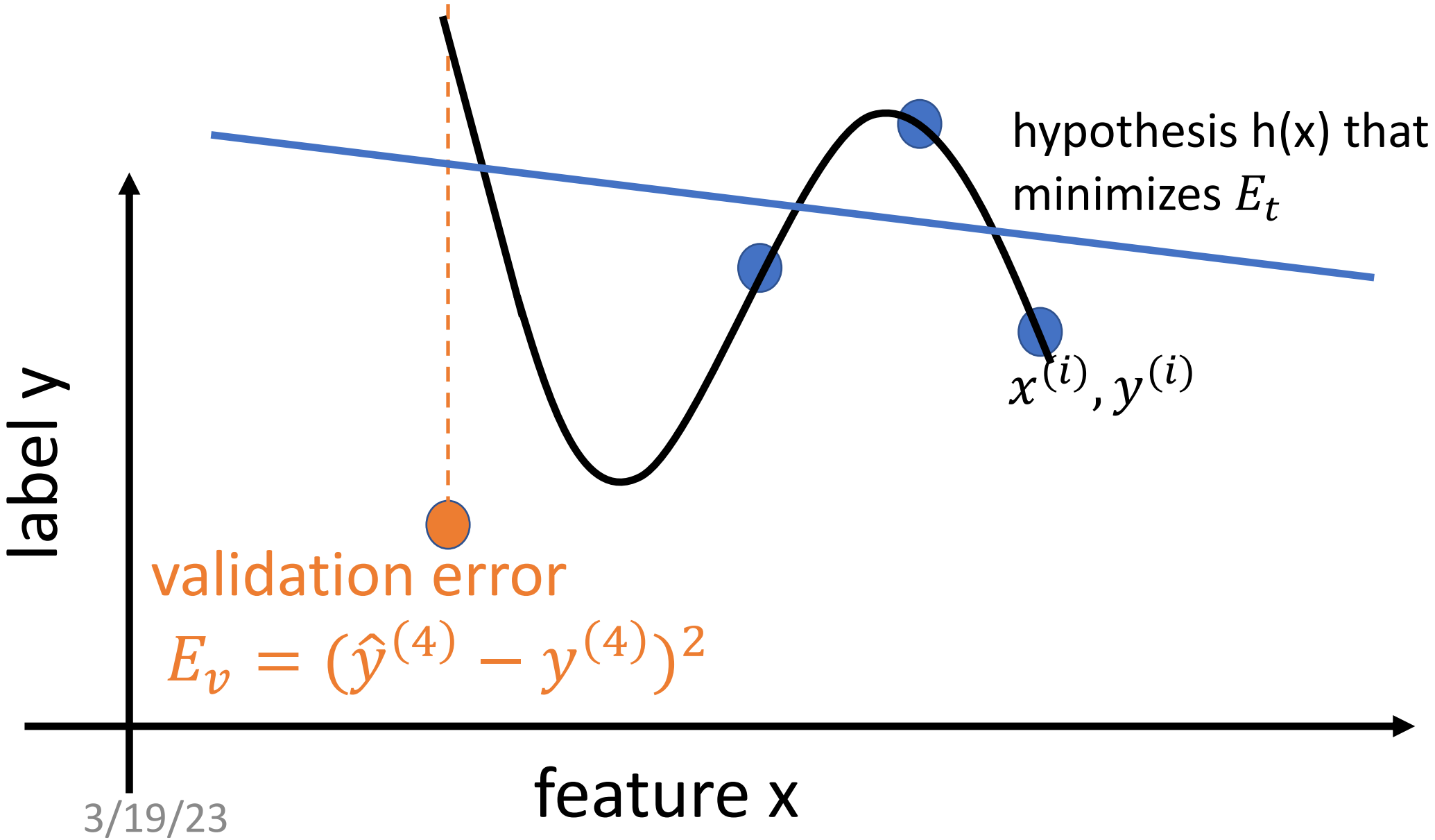
Plain Old Machine Learning.





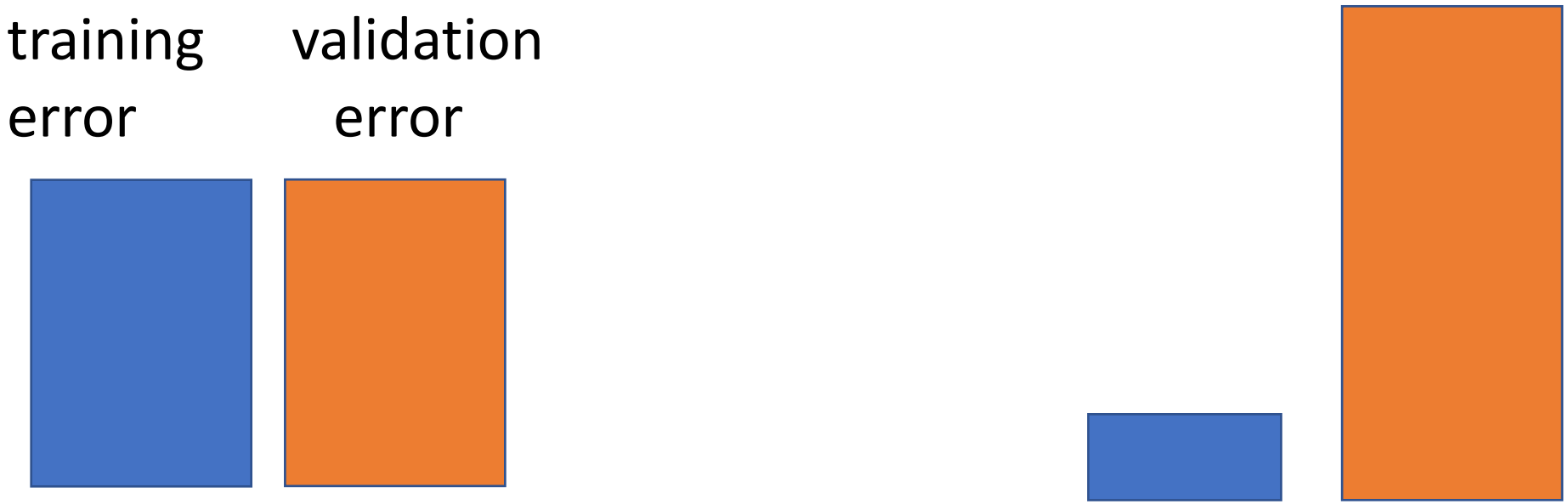
FINNISH METEOROLOGICAL INSTITUTE

Train and Validate.



Basic Idea of Model Selection

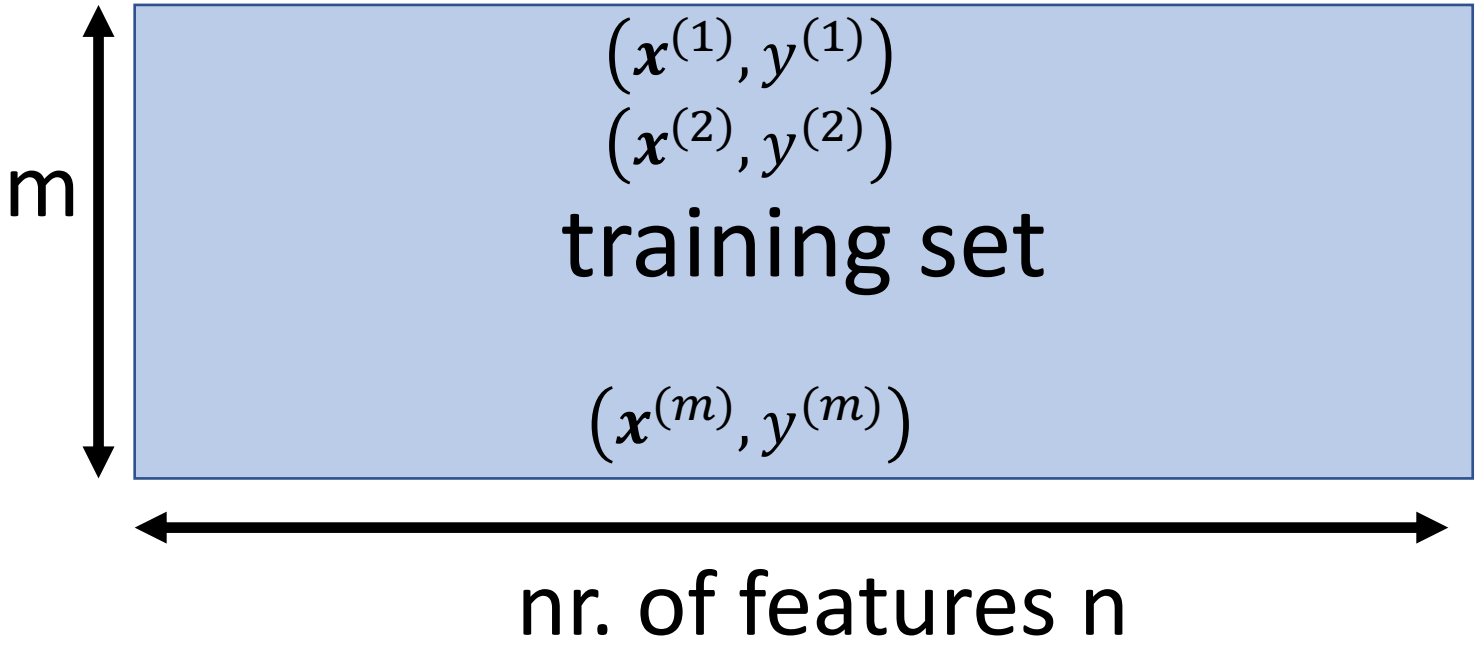
- choose model with smallest validation error!

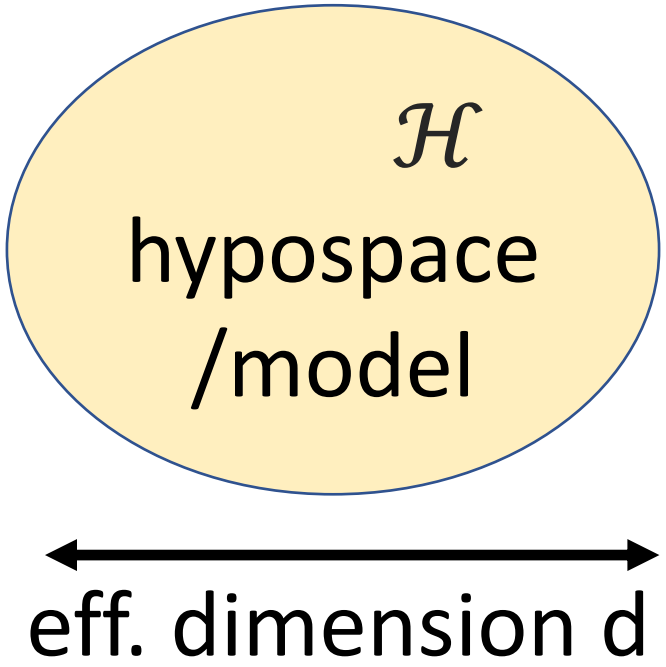


model 1
degree 1 polyn.

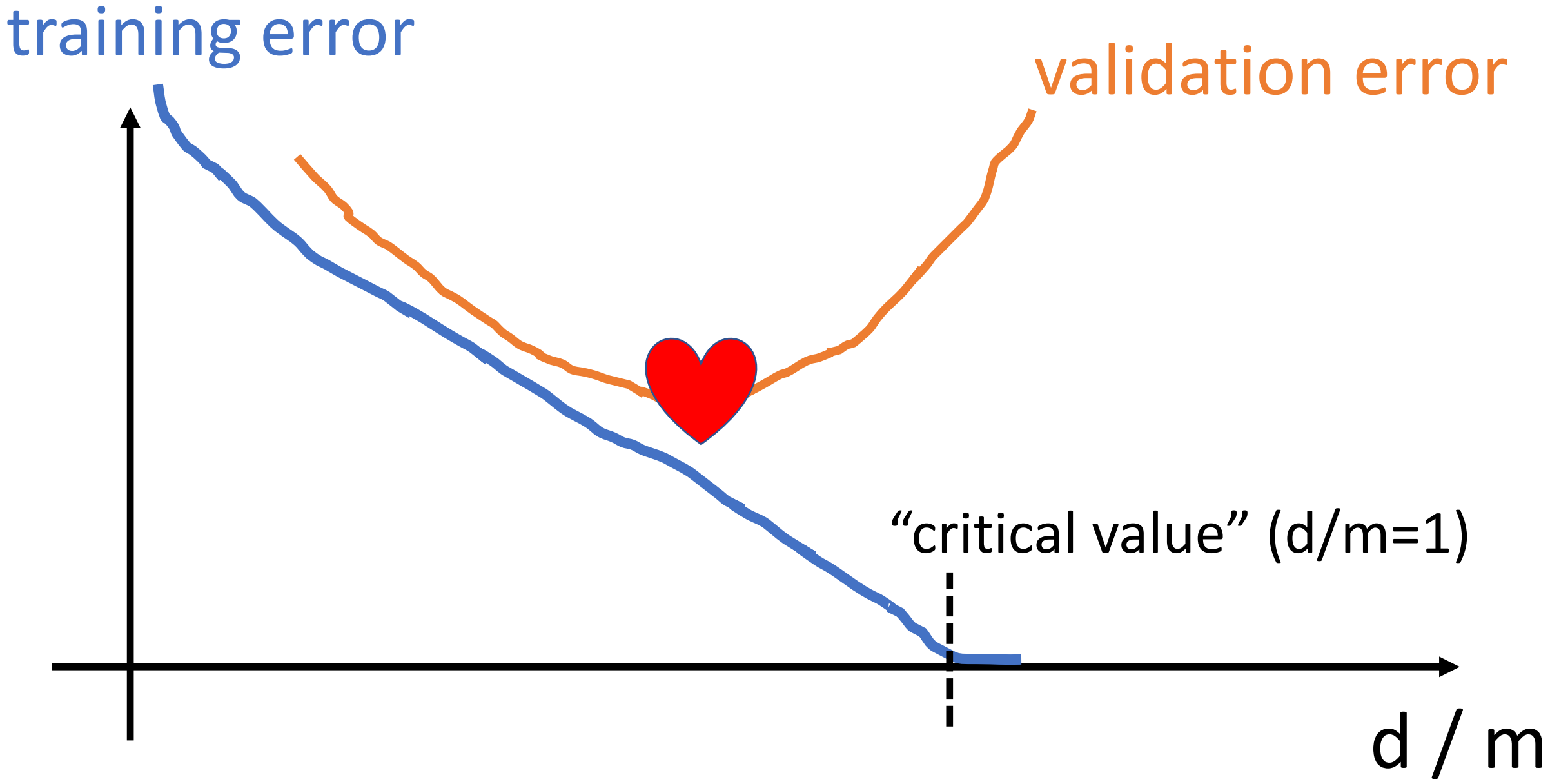
model 2:
degree 3 polyn.

Data and Model Size



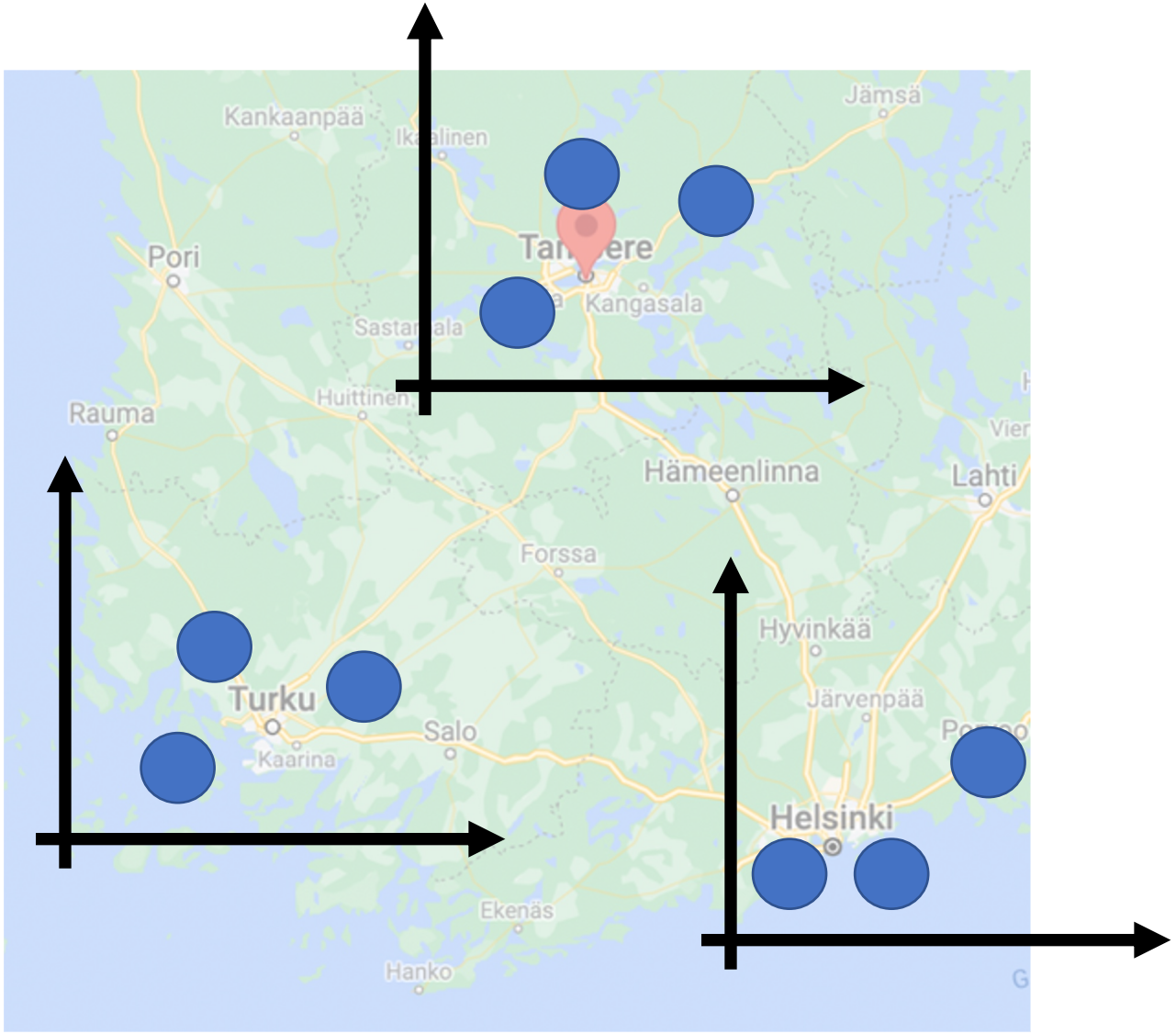


crucial parameter is the ratio d/m


training error
validation error
“critical value” (d/m=1)
d / m

Networked Data

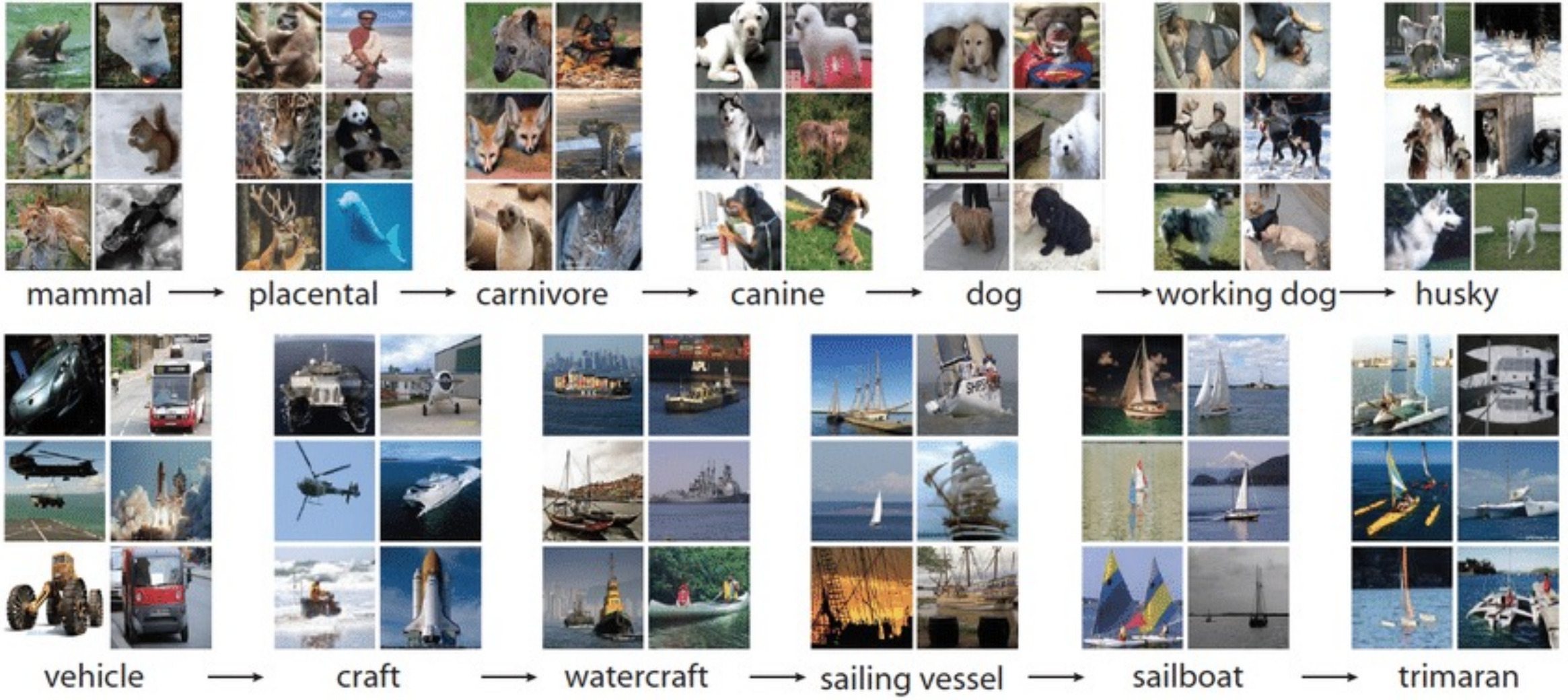
Networked Weather Data


Kankaanpää
Jämsä
Pori
Kangasala
Rauma
Hämeenlinna
Lahti
Forssa
Turku
Salo
Kaarina
Hyvinkää
Järvenpää
Helsinki
Ekenäs
Hanko

ImageNet.

"…ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images…"

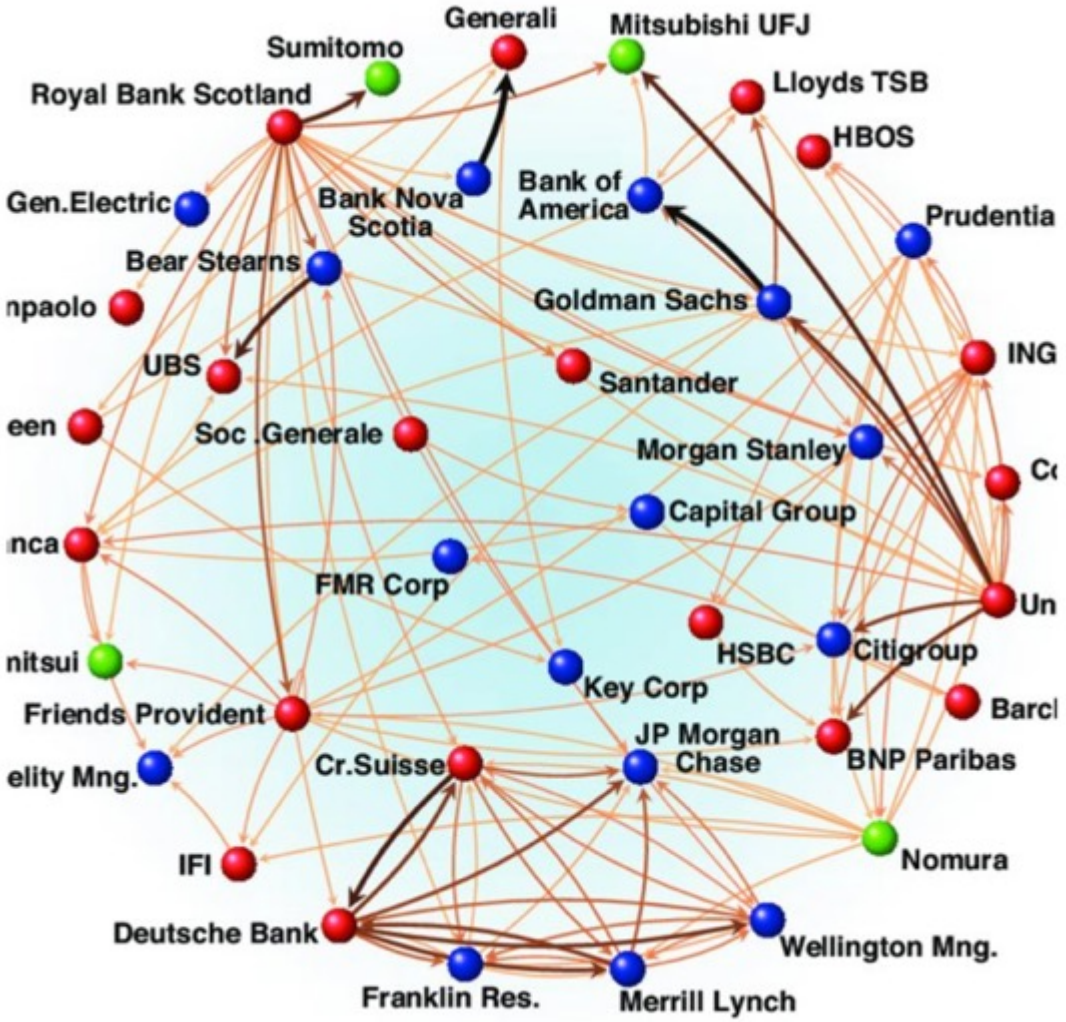
https://image-net.org/



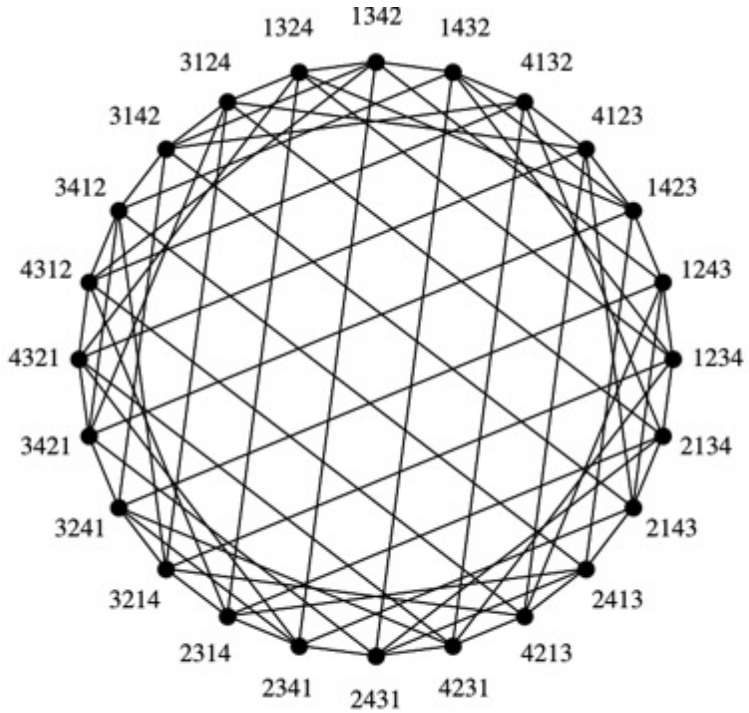
Devopedia. 2021. "ImageNet." Version 16, April 7. Accessed 2023-02-11. https://devopedia.org/imagenet

Noble Networks.

Finance Networks.



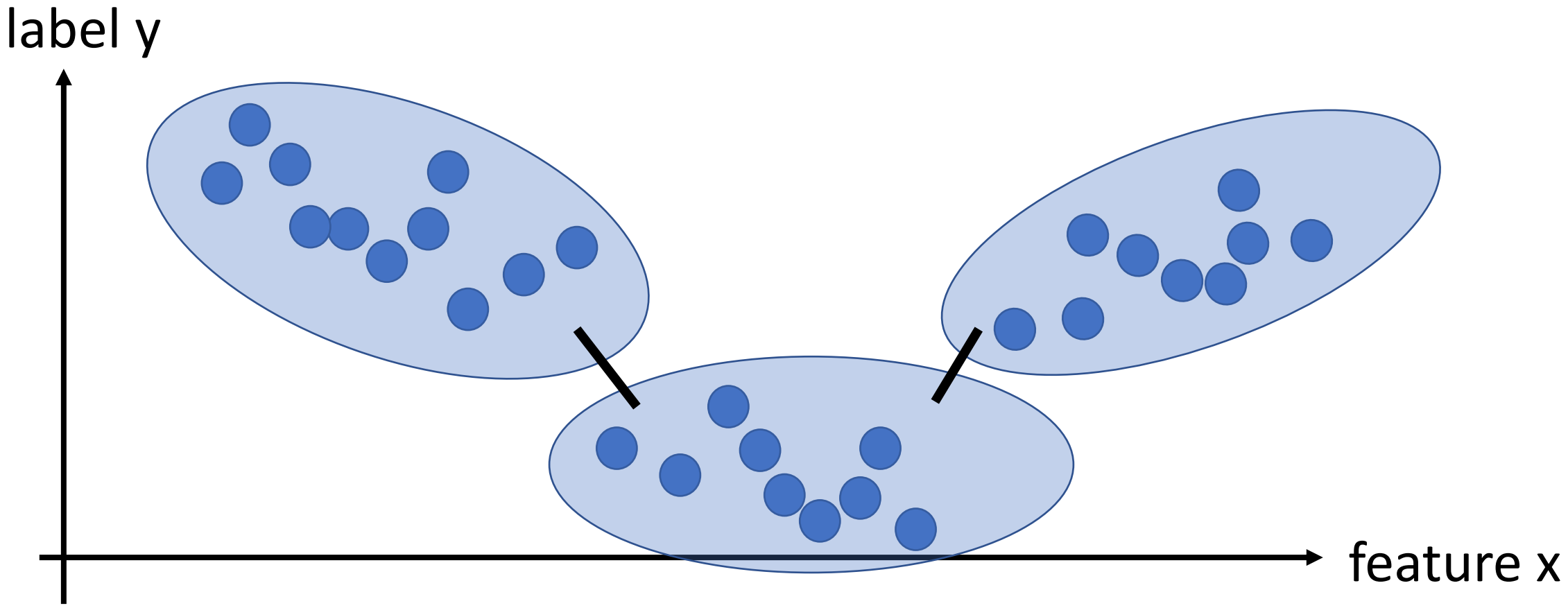
Algebraic Networks



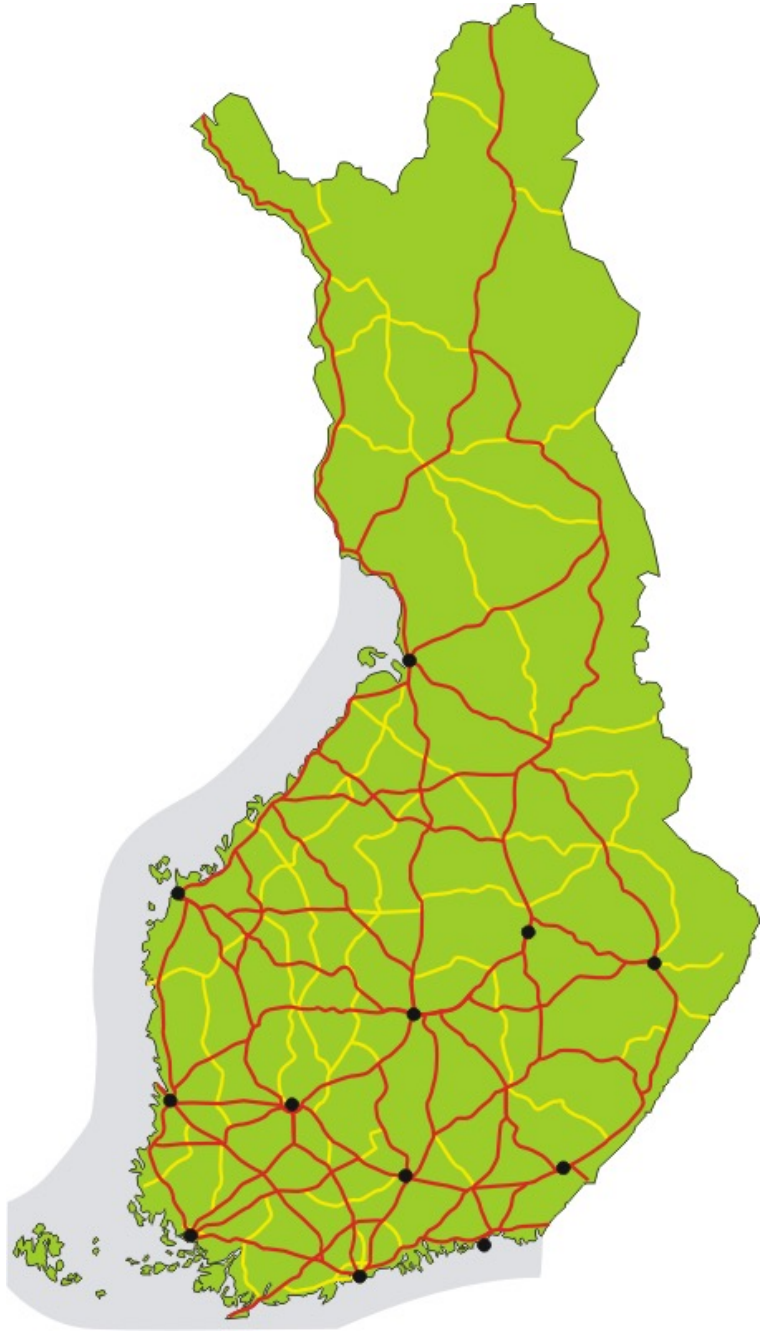
Example 4.3. Let $G = D_n$, the dihedral group of order $2n$, and take $S = \{r, r^{n-1}, s\}$.

Petteri Kaski, Eigenvectors and spectra of Cayley graphs, 2002

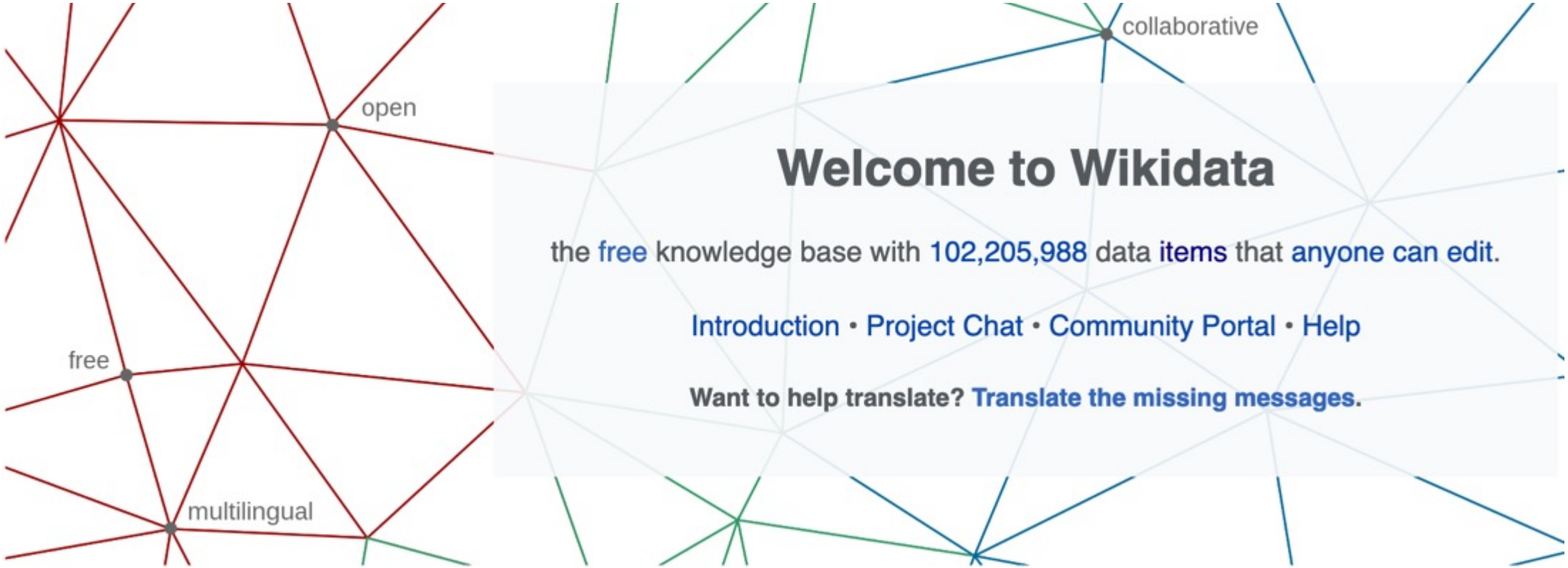
Point Cloud Networks.



Road Network

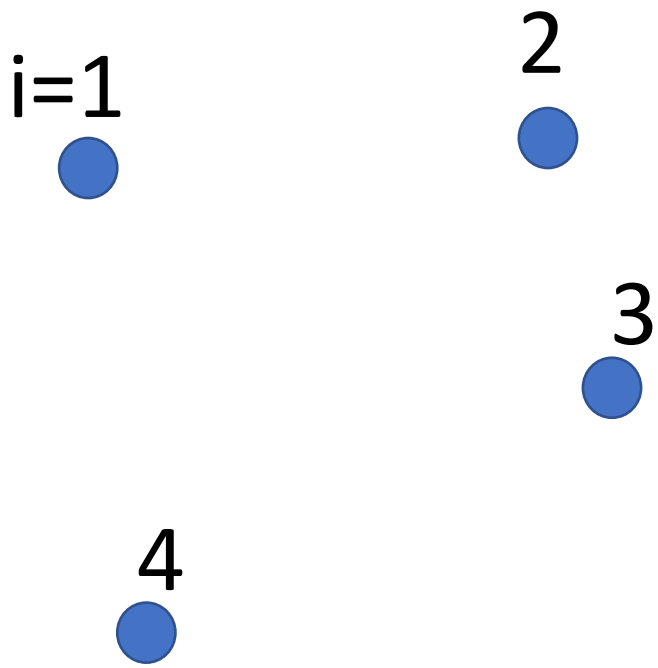


Business Process



https://www.wikidata.org/wiki/Wikidata:Main_Page

Empirical Graph - Nodes



each node carries:

- local dataset

```
# the node attribute "X" stores
G.nodes[iter_node]["X"] = X

# the node attribute "y" stores
G.nodes[iter_node]["y"] = y
```

```
G_FMI = nx.Graph()
G_FMI.add_nodes_from(range(0,num_stations))
```

Attach Local Datasets to Nodes

```
for i,station in enumerate(data.station.unique())

    # first filter out rows in dataframe data wit
    # current value of station

    df = data[data.station==station]

    # store the name of the current station in th
    G_FMI.nodes[i]['name'] = station
    # store the lat and lon of the current statio
    G_FMI.nodes[i]['coord'] = np.array([df.latitu
    # store the min. daytime temp. in the node at
    G_FMI.nodes[i]['X'] = df.min_temp.to_numpy().
    # store the max. daytime temp. in the node at
    G_FMI.nodes[i]['y'] = df.max_temp.to_numpy().
```

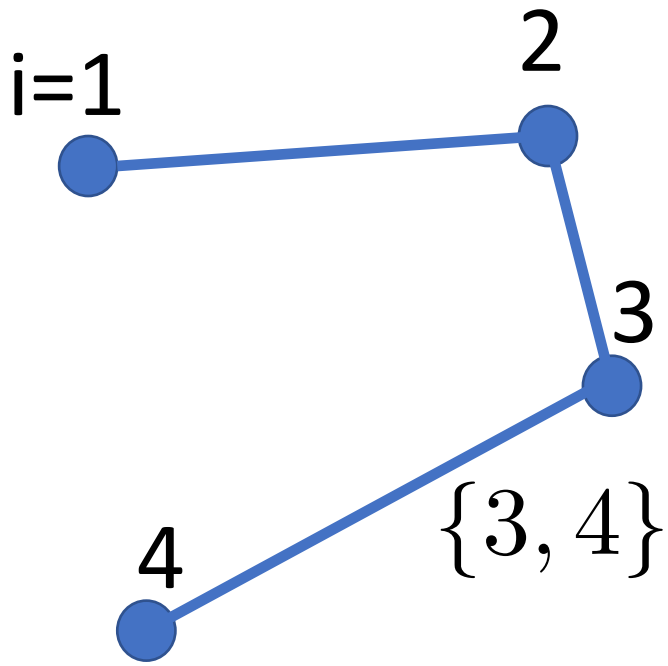
name of FMI station

latitude, longitude of FMI station

use min. daytime tmp. as feature of datapoint (=some day)

use max. daytime tmp. as labels of datapoint (= some day)

Empirical Graph - Edges



undirected edge indicates "similarity"

$\{3,4\} \in \mathcal{E}$

edge weight $A_{3,4}$ = level of similarity

How To Measure Statistical Sim.?

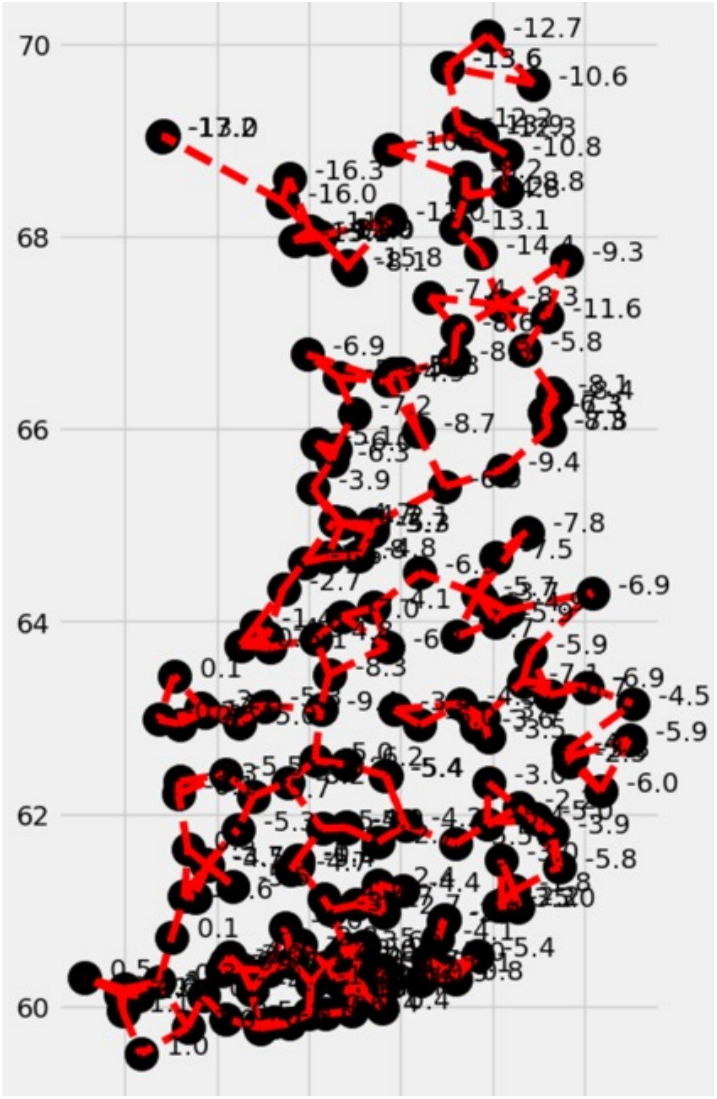
```
>>> from scipy.stats import ks_2samp
>>> import numpy as np
>>>
>>> np.random.seed(12345678)
>>> x = np.random.normal(0, 1, 1000)
>>> y = np.random.normal(0, 1, 1000)
>>> z = np.random.normal(1.1, 0.9, 1000)
>>>
>>> ks_2samp(x, y)
Ks_2sampResult(statistic=0.022999999999999909, pvalue=0.9518901680484964 7)
>>> ks_2samp(x, z)
Ks_2sampResult(statistic=0.41800000000000004, pvalue=3.7081494119242173e-77)
```

https://stackoverflow.com/questions/10884668/two-sample-kolmogorov-smirnov-test-in-python-scipy

https://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test

Spatio-Temporal Similarity

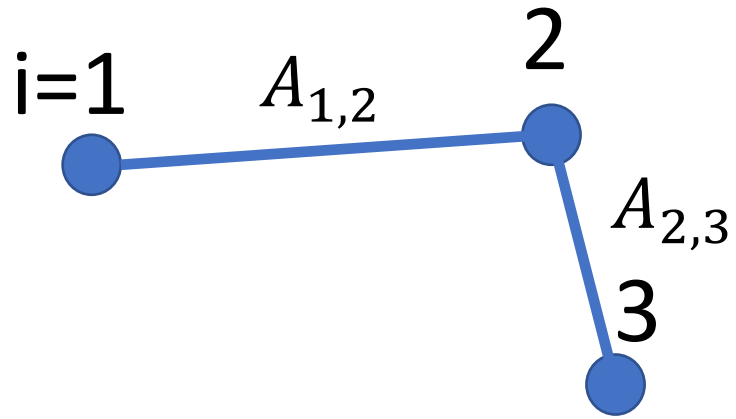
connect FMI stations with its k nearest neighbours (use geodesic distance)



How to choose Empirical Graph?

- empirical graph is design choice
- more edges means more comp./comm.
- sufficiently many edges between similar datasets

Graph Laplacian Matrix

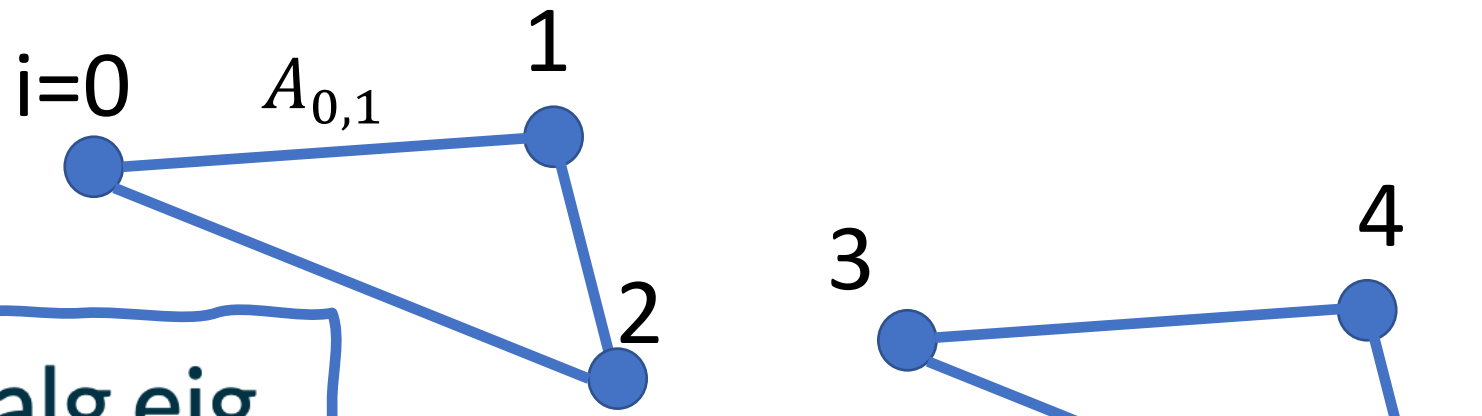


$$\mathbf{L} = \begin{pmatrix} A_{1,2} & -A_{1,2} & 0 \\ -A_{1,2} & A_{1,2} + A_{2,3} & -A_{2,3} \\ 0 & -A_{2,3} & A_{2,3} \end{pmatrix}$$

L is psd with eig.vals $\lambda_1 = 0 \leq \lambda_2 \ldots \leq \lambda_{nr\ nodes}$

Spectral Graph Clustering

eig.vecs of **L** corresponding to smallest pos. eig.vals $\lambda_1 = 0 \leq \lambda_2$ are piece-wise constant over clusters

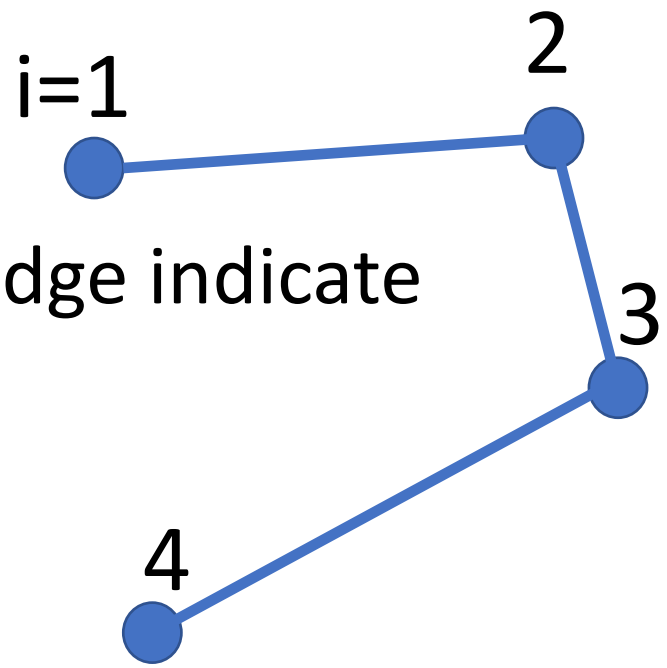


numpy.linalg.eig

```
linalg.eig(a)
```

What is Component 2 of ML ?

Local Model for Local Data



undirected edge indicate "similarity"

each node carries:

- local dataset

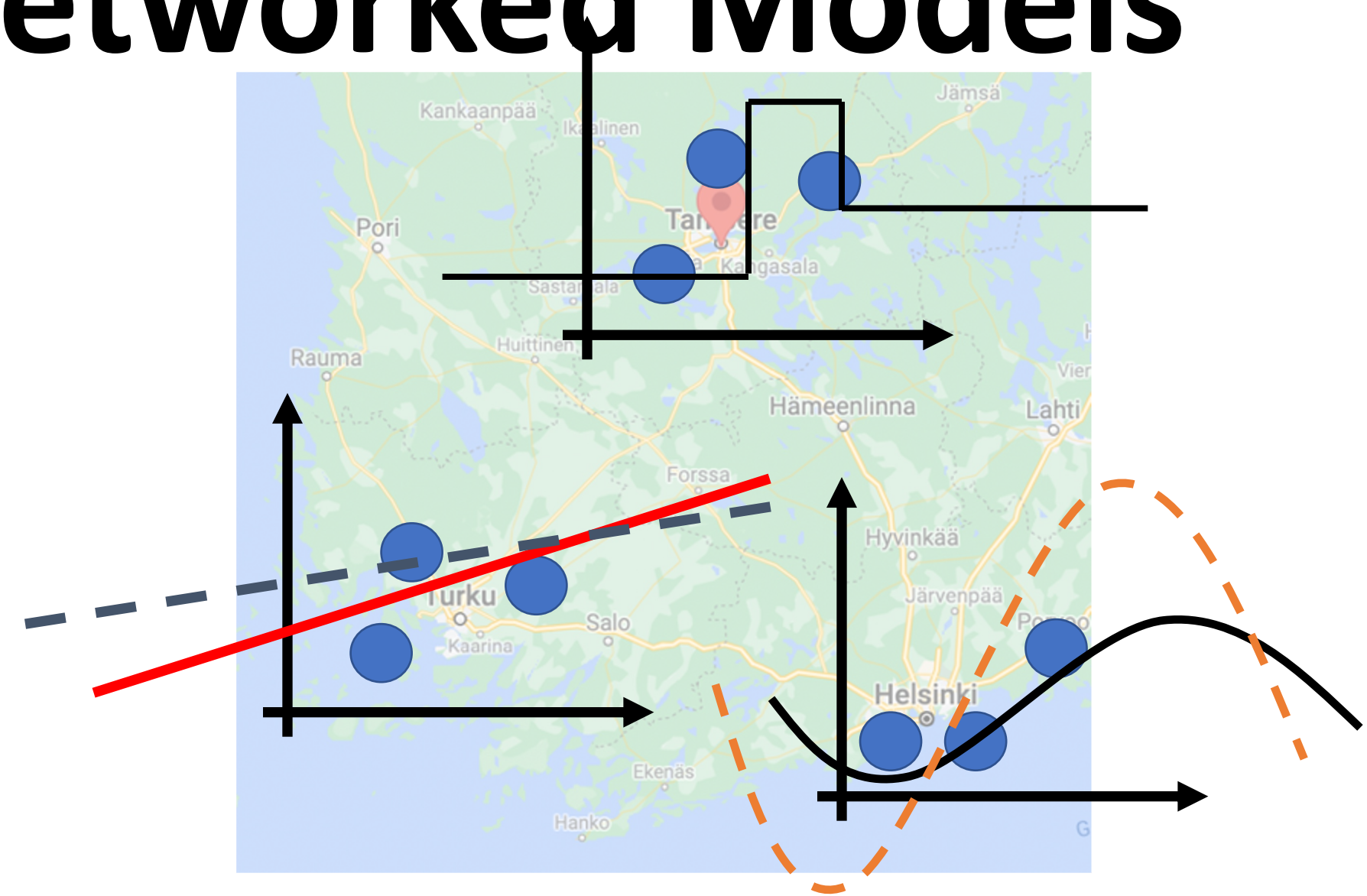
```
# the node attribute "X" stores
G.nodes[iter_node]["X"] = X

# the node attribute "y" stores
G.nodes[iter_node]["y"] = y
```

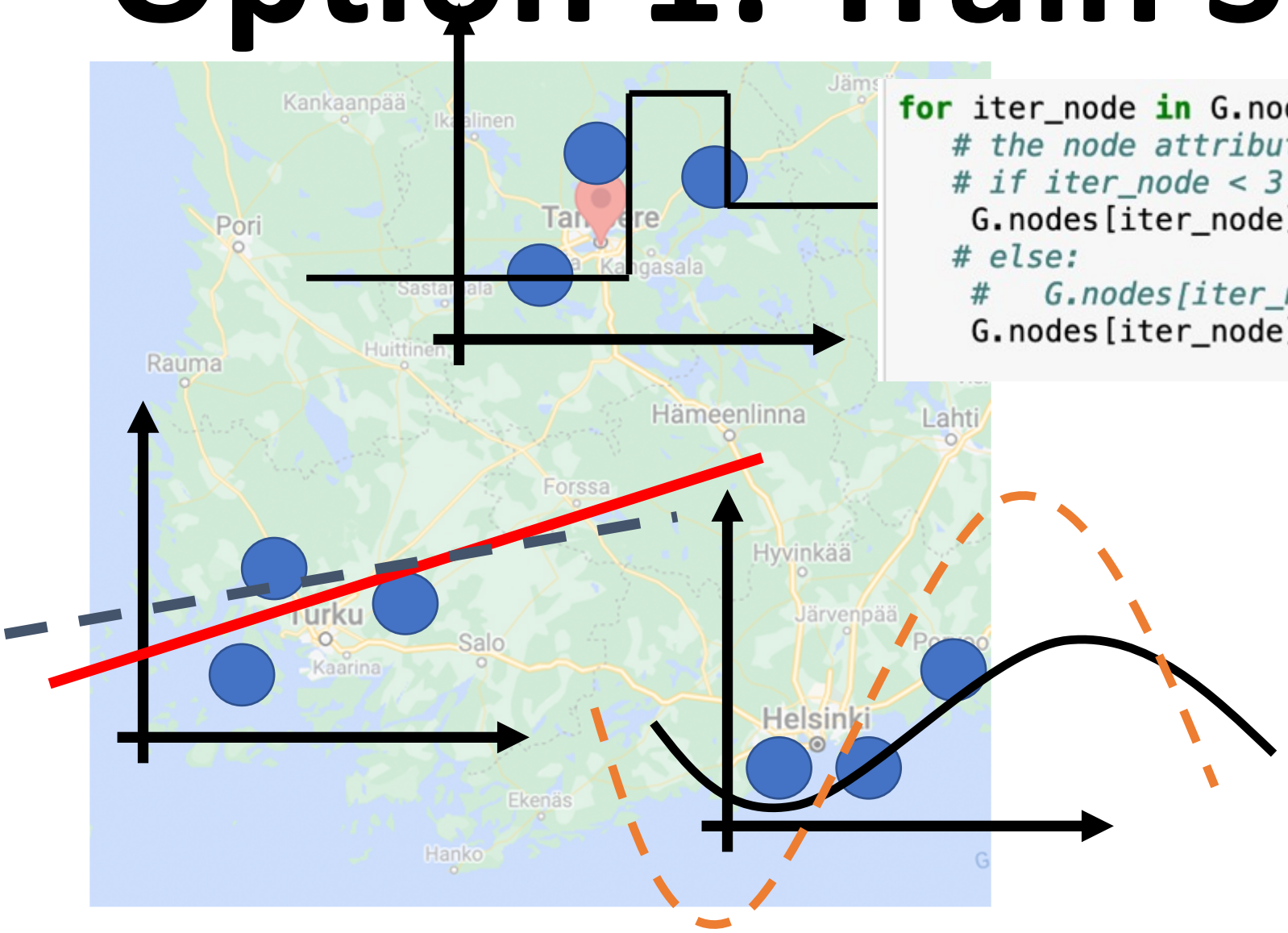
- local model

```
# the node attribute "model" stores a ML model of
G.nodes[iter_node]["model"] = LinearRegression()
G.nodes[iter_node]["model"].fit(G.nodes[iter_node]
```

Networked Models

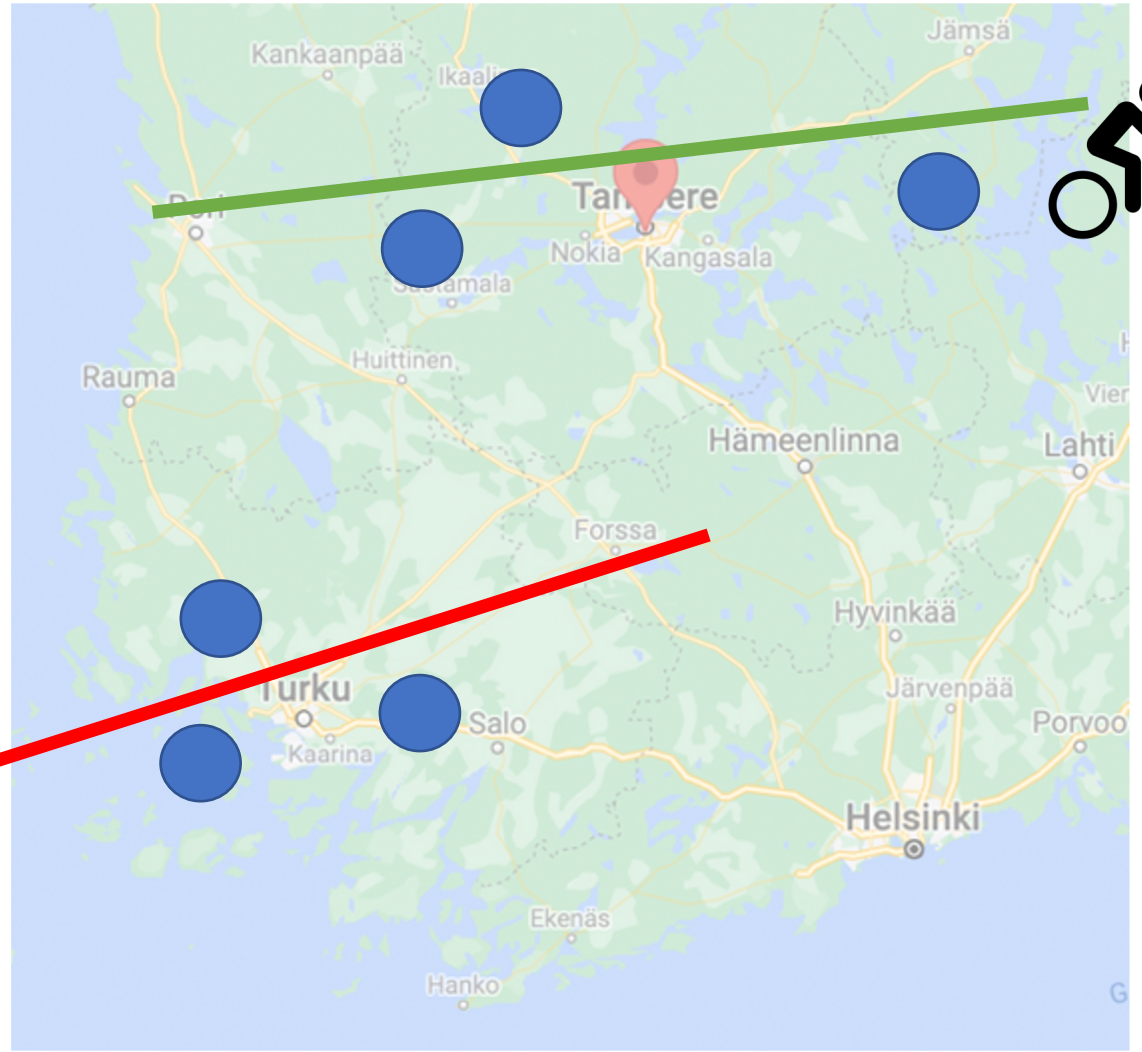


Option 1: Train Separately

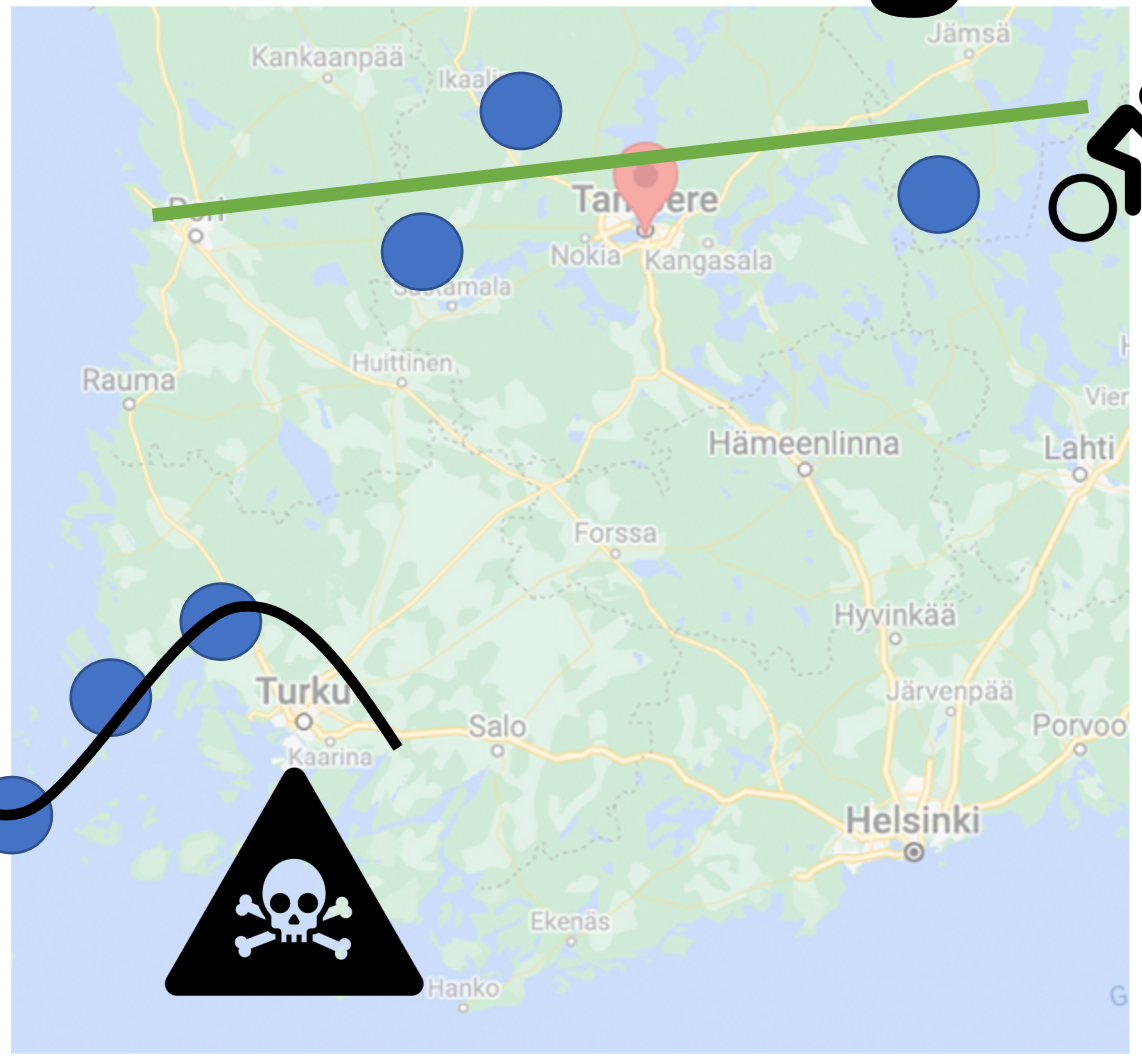



```
for iter_node in G.nodes:
    # the node attribute "model" stores a ML model of your choic
    # if iter_node < 3 :
     G.nodes[iter_node]["model"] = LinearRegression()  # initial
    # else:
     #   G.nodes[iter_node]["model"] = DecisionTreeRegressor(max
     G.nodes[iter_node]["model"].fit(G.nodes[iter_node]["X"],G.r
```

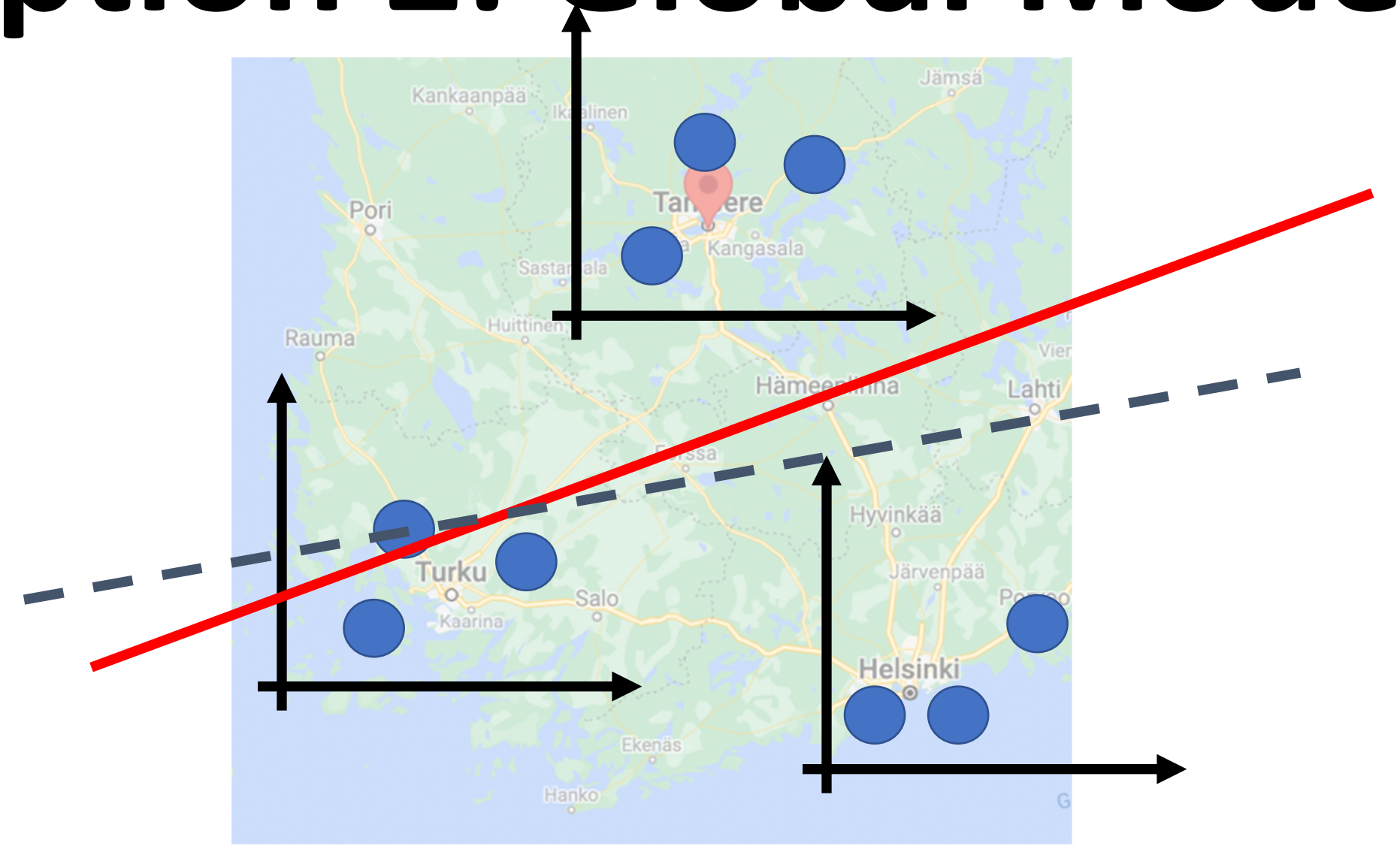
Pro: Personalization



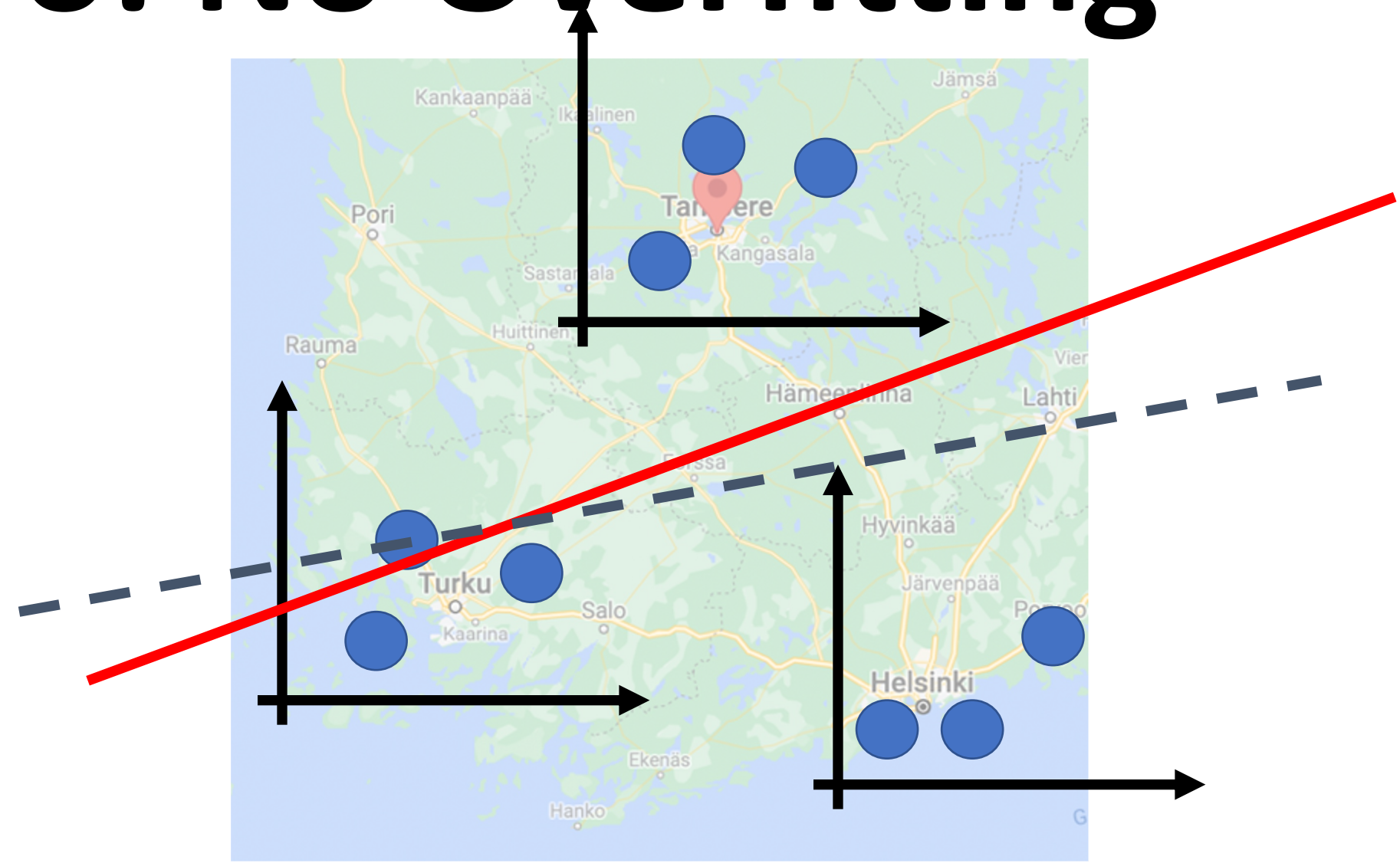
Con: Overfitting



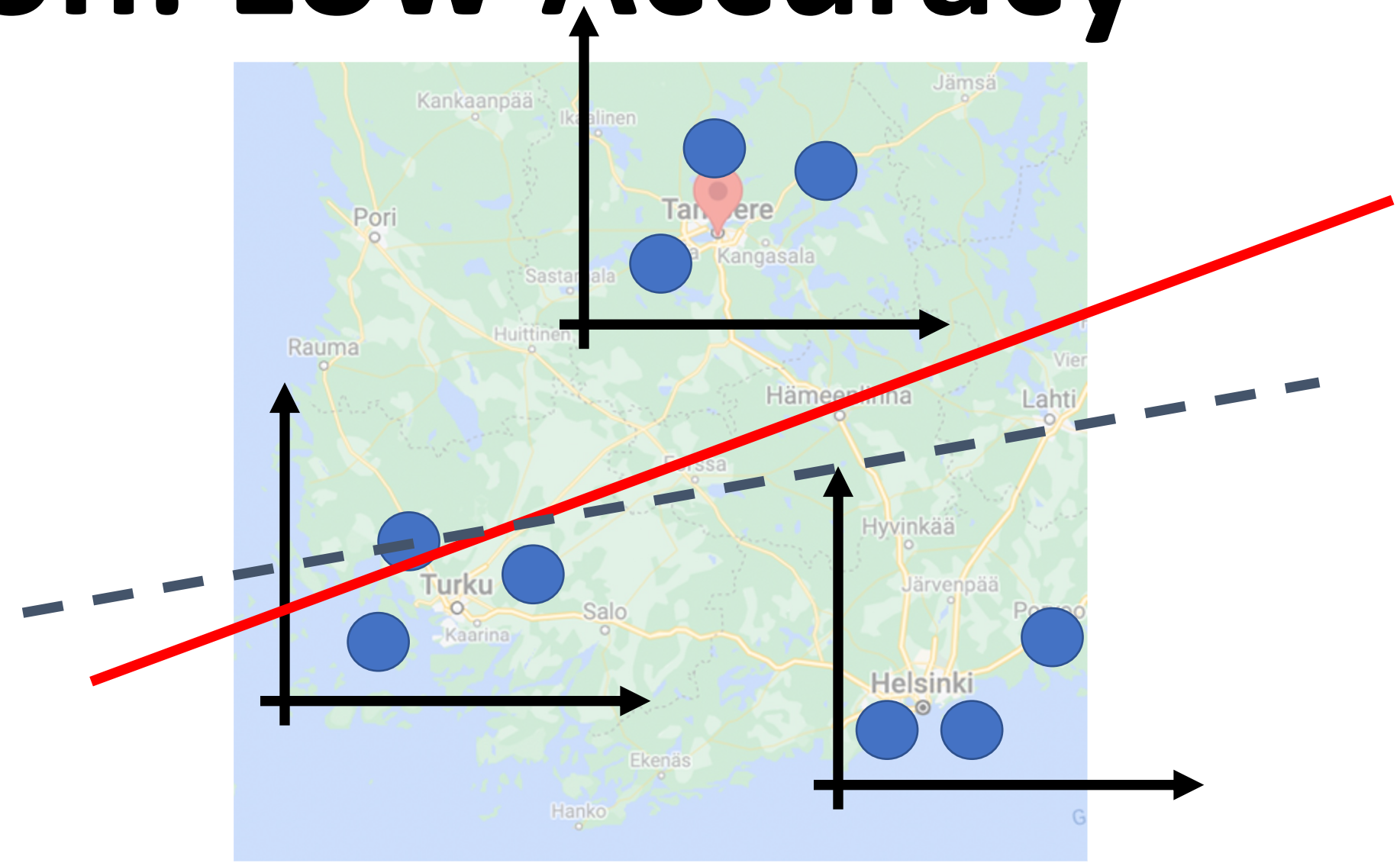
Option 2: Global Model



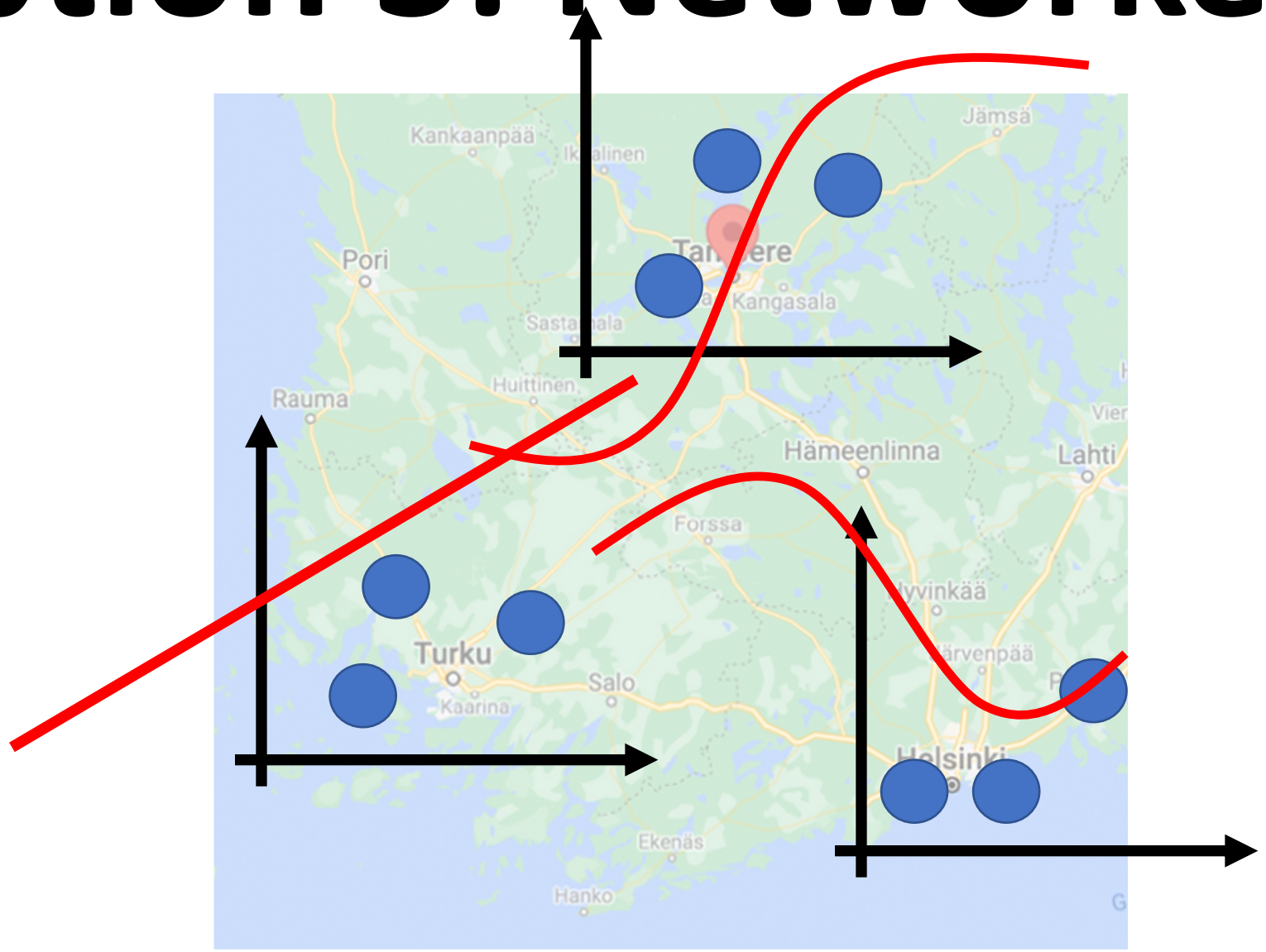
Pro: No Overfitting



Con: Low Accuracy



Option 3: Networked FL

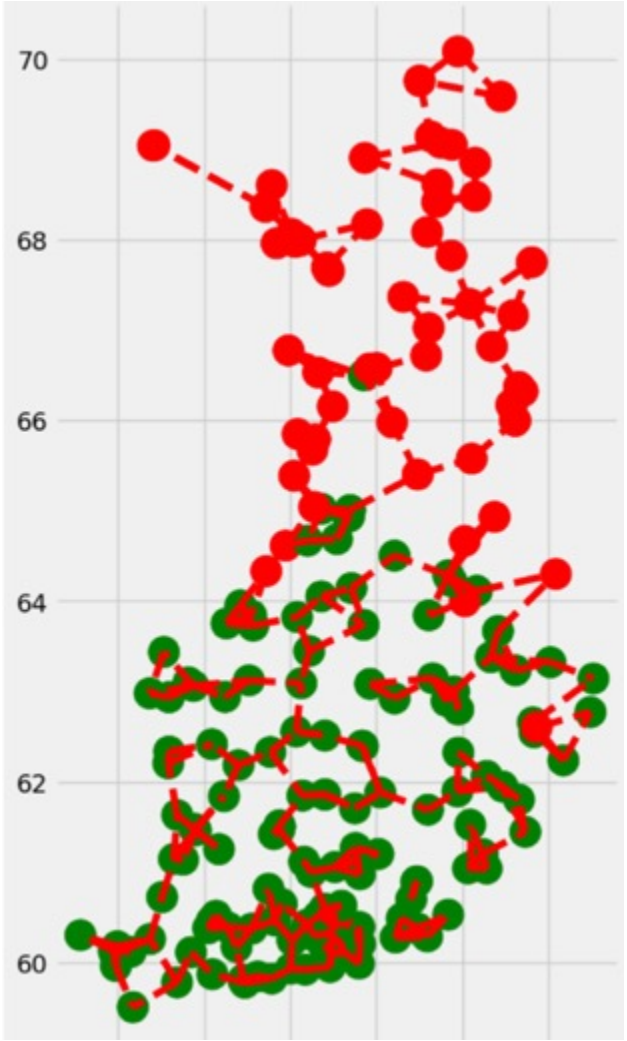


Networked FL

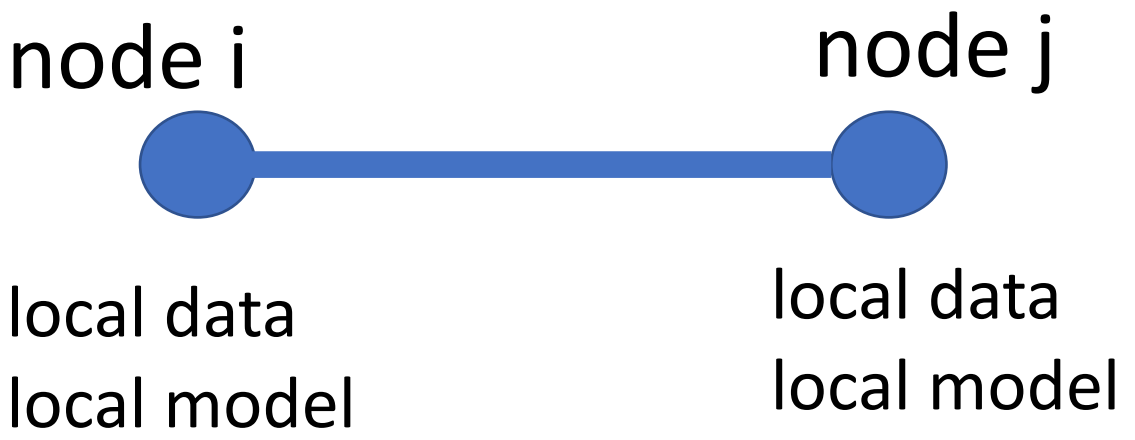
- maintain local models for each node (local dataset)
- local models are trained on local datasets
- couple the training of local models
- coupling via clustering (global model per cluster)
- different clustering methods

Coupling of Local Models

- coupling based on edges
- local models at connected nodes should deliver similar predictions



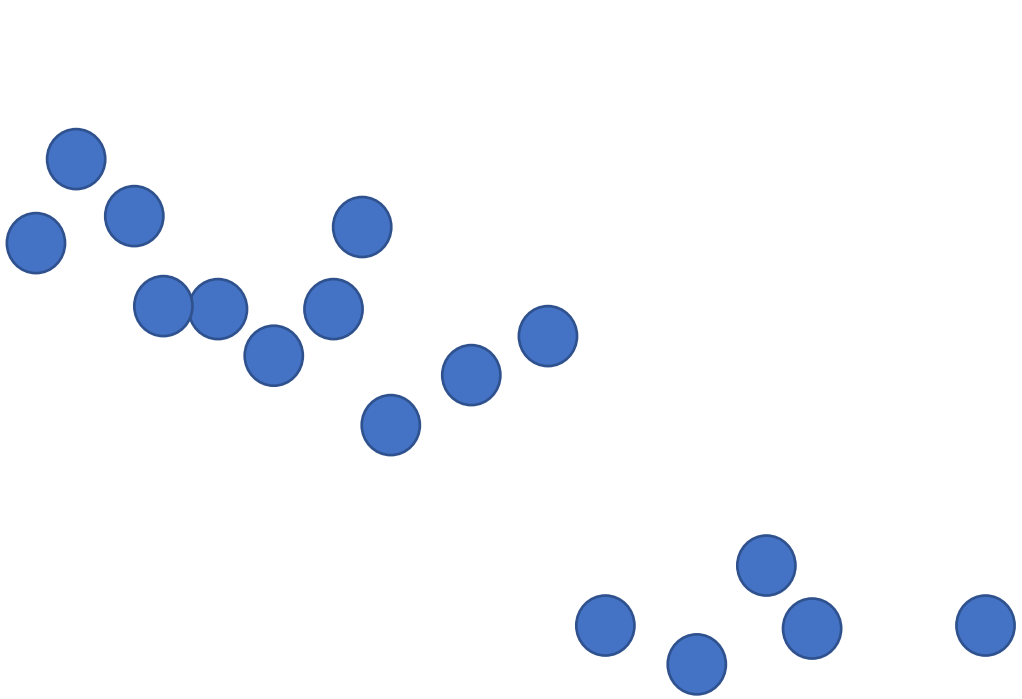
Coupling via Regularization




```
for node_i in G.nodes(data=False):
    tmp_y = G.nodes[node_i]["y"].reshape(-1,1)
    tmp_X = G.nodes[node_i]["X"]
    sample_weight = np.ones((samplesize,1))
    # loop over all neighbours of current node "node_i"
    for node_j in G[node_i]:
        tmp_X = np.vstack((tmp_X,G.nodes[node_i]["X"]))
        tmp_y = np.vstack((tmp_y,G.nodes[node_j]["model"].predict(G.nodes[node_i]["X"]).reshape(-1,1)))
        sample_weight = np.vstack((sample_weight,100*np.ones((samplesize,1))))
   # print(tmp_X.shape)

    G.nodes[node_i]["model"].fit(tmp_X,tmp_y,sample_weight.reshape(-1))
```

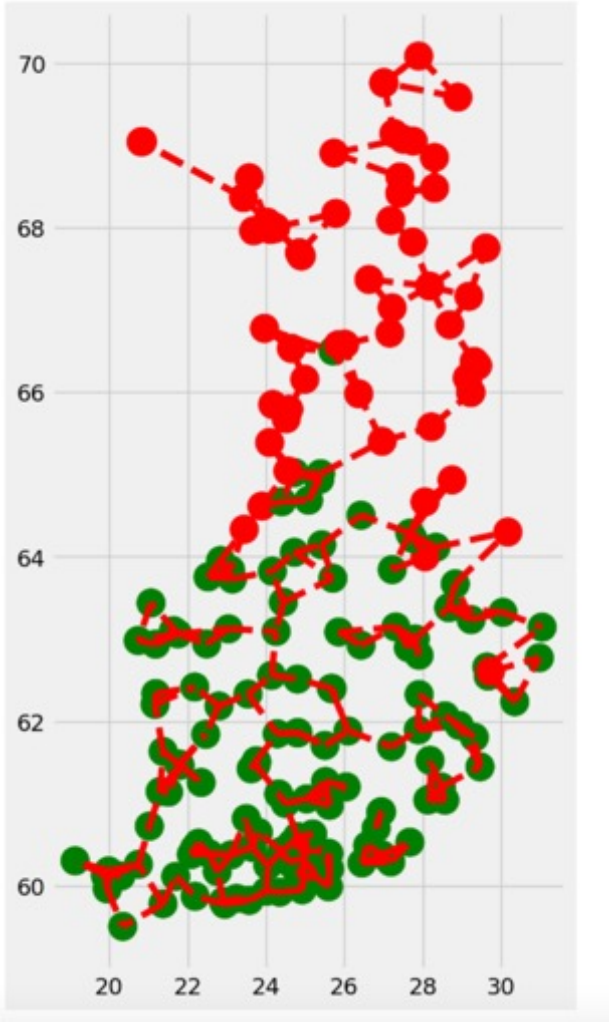
Clustering Assumption



local datasets often too small

let’s pool or cluster them but how ?

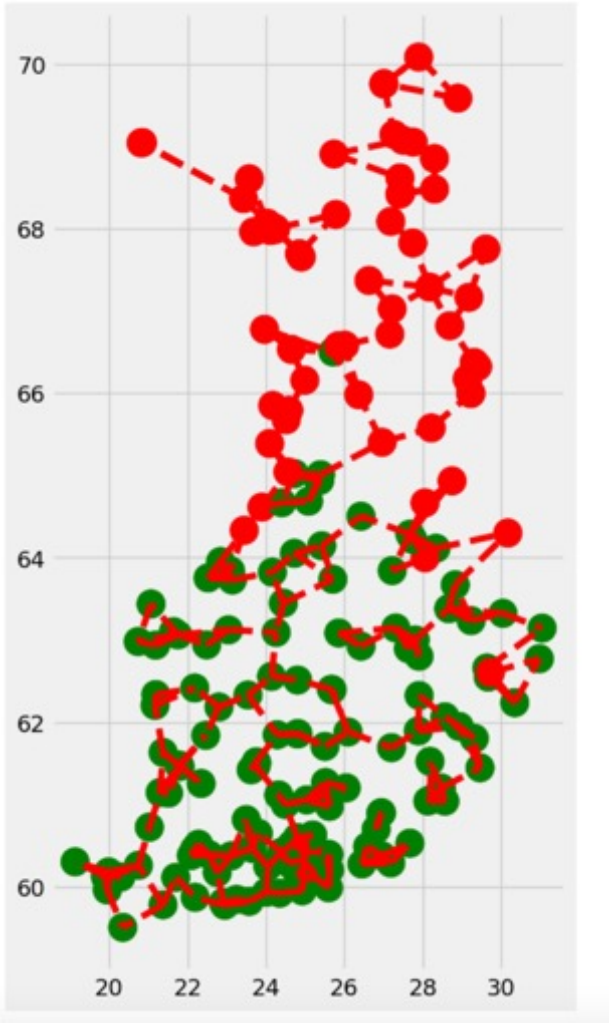
Clustering Assumption



the local datasets form clusters.

local datasets in same cluster c contain data points that can be approximated as realizations of i.i.d. RVs with prob. dist p(x,y;c)

Cluster-Wise Training



pool local datasets of nodes in same cluster to train ML model

trained model is used for all nodes in same cluster

pooling via regularization

Wrap-Up.

- emp. graph with nodes carrying local datasets and models
- undirected weighted edges represent similarities
- different sources for similarity
- emp. graph is a design choice

Any Questions ?