University of Sheffield | Department of Automatic Control and Systems Engineering

Industry Training Programme ACS6402, ACS6403 Laser Powder Bed Fusion - Process Monitoring & Defect Forecasting Seminar Two

Academic year 2023/24

Mohamed Atwya[1]

[1]Department of Automatic Control and Systems Engineering
University of Sheffield

March 11, 2024

Table of Contents

Table of Contents

Project Objectives

- Review the literature on additive manufacturing porosity detection and localisation
- Review the industry (AMRC) data and sensing system to understand the input and target values and perform any necessary data treatment (repeated data, data imbalance, missing data, etc.)
- Identify and add features to the input data to improve the data set for a machine learning problem
- **Propose a suitable method, and apply, validate, and test a supervised machine learning algorithm for localising the porosities (binary classification problem)**

Table of Contents

The Four Elements of Machine Learning

- **Assumption**
  - What we think the world is like (observations)
- **Model**
  - A mathematical way of expressing the assumption/thought
- **Inference paradigm**
  - A framework to match the model to the observation
- **Inference engine**
  - A means of matching the model to the observation

Table of Contents

Multi-layer Perceptron (MLP) Neural Network Architecture (1)

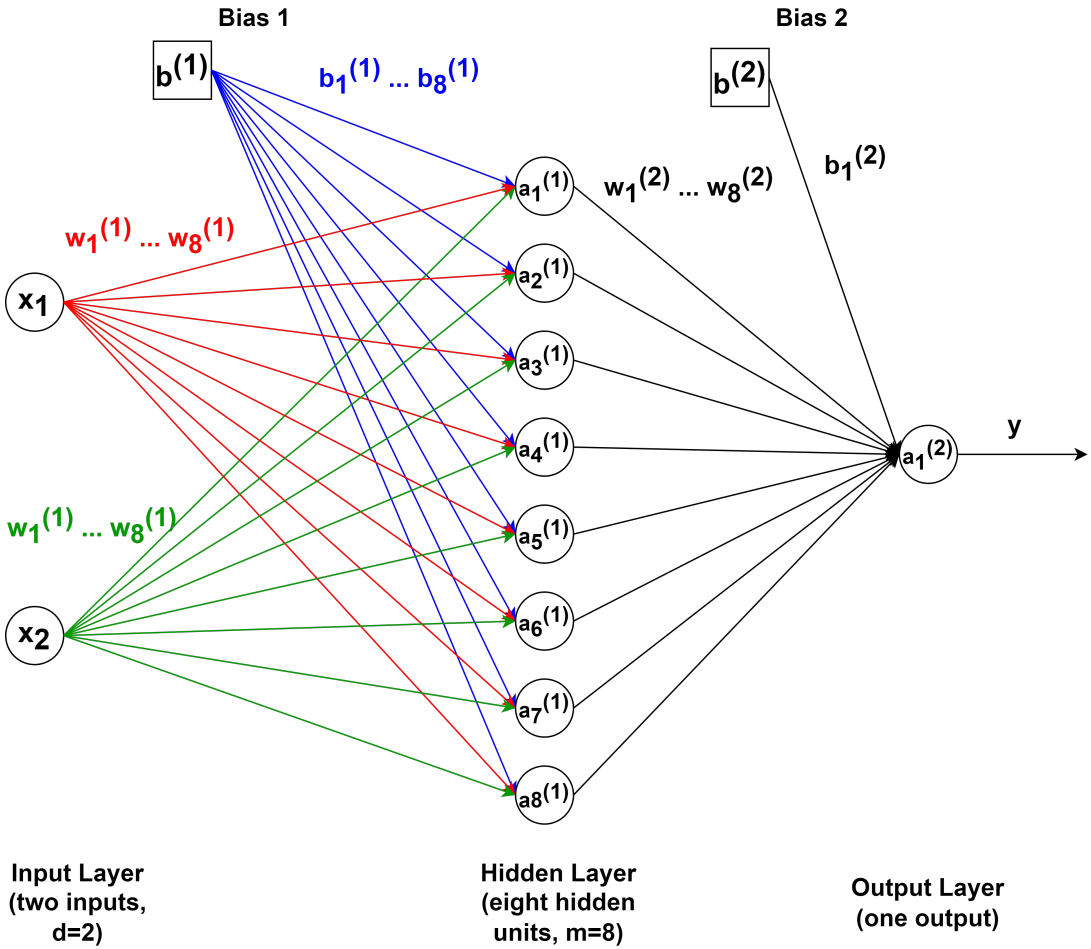


Figure: MLP architecture diagram.

MLP Layer one Equations (2)

$$\underline{x}_i, i \in \{1, 2, \cdots, d\}, \tag{1}$$

$$z_j^{(1)} = \sum_{i=1}^{d} \left(x_i w_{ji}^{(1)} + b_j^{(1)} \right), j \in \{1, 2, \cdots, m\}, \tag{2}$$

$$a_j^{(1)} = h\left(z_j^{(1)} \right), \tag{3}$$

Where d is the number of inputs, m is the number of hidden units, h is the hidden activation function, $\underline{x}_i$ are the input features, and a_j^1 is the hidden later activation output.

MLP Layer Two Equations (3)

$$z_k^{(2)} = b_k^{(2)} \sum_{j=1}^{m} \left(a_j^{(1)} w_{jk}^{(2)} \right), k \in \{1, 2, \cdots, n\}, \tag{4}$$

$$a_k^{(2)} = g\left(z_k^{(2)} \right), \tag{5}$$

$$y_k = a_k^{(2)}, \tag{6}$$

Where n is the number of output variables, g is the output layer activation function, and $a_k^{(2)}$ is the output layer activation output.

Table of Contents

Hidden Layer Activation Function (1)

$$Sigmoid : \frac{1}{1 + e^{-x}} \tag{7}$$

- Smooth gradient and output values bound between 0 and 1
- Can saturate and have 0 gradients: The output of sigmoid saturates for a large positive or large negative number. Thus, the gradient at these regions is almost zero.
- Not zero-centred: Sigmoid outputs are not zero-centred, which is undesirable because it can indirectly introduce undesirable zig-zagging dynamics in the gradient updates for the weights

Hidden Layer Activation Function (2)

$$ScaledSigmoid : \tan x \tag{8}$$

- Smooth gradient and output values bound between -1 and 1
- Zero-centred
- Can saturate and have a 0 gradients

$$RectifiedLinearUnit : \begin{pmatrix} x & for & x > 0 \\ 0 & for & x < 0 \end{pmatrix} \tag{9}$$

- Dying ReLU problem, where neurons with negative inputs become permanently inactive and stop contributing to the network's output
- Computationally efficient

Output Layer Activation Function

- Linear activation function (regression)

$$y = a_k^{(2)} \tag{10}$$

- Logistic activation function, outputs a value in the range $[0, 1]$ (classification)

$$\frac{1}{1 + e^{-x}} \tag{11}$$

Table of Contents

Model Weight Initialisation (1)

- Before training, the model bias values are typically set to 0
- However, the model weights are initialised with "small" random values
  - If all the weights are initialized with 0, the derivative with respect to loss function is the same for all the weights and all the weights would continue to have the same value during training
- Vanishing gradient: the gradient of the loss function with respect to the parameters of the network becomes very small, causing lower layers of the network to not receive meaningful updates during training
- Exploding gradient: the gradient of the loss function with respect to the parameters becomes very large, causing the optimizer to overshoot and diverge
- Each time a neural network is initialised with random weights, the final set of weights after training will be different and perform differently on the OOS data

Model Weight Initialisation (2)

- "xavier" initialization for the sigmoid and tanh activation functions

$$\mathcal{U}\left(-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right) \tag{12}$$

- "He" initialization for the ReLu activation function

$$\mathcal{N}\left(0, \sqrt{\frac{2}{d}}\right) \tag{13}$$

- The choice of the model weight initialisation strategy is more important the deeper the neural network is due to the vanishing gradient problem
- If the neural network is very deep, when the gradients are multiplied together in the backpropagation algorithm, they become exponentially smaller as they propagate through the network

Table of Contents

Complexity Control (1)

- Overfitting is training a model to "exactly" fit the training data, leading to poor generalisation (poor performance on unseen data, OOS)
  - In other words, the model is overly-complex has a high prediction performance variance
- Underfitting is when the model is too simple and fails to capture the non-linear input-output relation (poor performance on the training and OOS data)
- Typically, we set the number of hidden units to a sufficiently "large" value in the range [5, 100] and use a complexity control measure to avoid over-fitting
  - L1 regularisation (Lasso Regression)
  - L2 regularisation (Ridge Regression)
  - Elastic Nets

Complexity Control (2)

- Let $\underline{w}$ be a vector of the neural network weights with I elements
- L1 regularisation: The L1 norm is not differentiable at zero, thus the optimization algorithm will tend to push some of the weights to exactly zero

$$L_r = \|\underline{w}\|_1 = \sum_{i=1}^{I} |\underline{w}_i| \tag{14}$$

- L2 regularisation: Shrinks the magnitude of all the weights towards zero, resulting in a smoother solution

$$L_r = \|\underline{w}\|_2^2 = \underline{w}^T \underline{w} \tag{15}$$

- Elastic Nets

$$L_r = \|\underline{w}\|_1 + \|\underline{w}\|_2^2 \tag{16}$$

Table of Contents

The loss function & Training Performance Index

- let $\underline{y}$ and$\underline{\hat{y}}$ be the target and prediction vectors with I elements
- Half Summed Square Error (HSS) is one performance index that can be used for the empirical loss in regression problems 17
- Logistic loss function, also known as the binary cross-entropy cost is a performance index that can be used for the empirical loss in classification problems (Eq. 18)
- Combining the empirical and reguralsiation losses we get the neural network cost function in Eq. 19

$$L_e = \frac{1}{2}\sum_{i=1}^{I}\left(\left(\underline{y}_i - \underline{\hat{y}}_i\right)^2\right). \tag{17}$$

$$L_e = -\sum_{i=1}^{I}\left(\underline{y}_i log\left(\underline{\hat{y}}_i\right) + \left(1-\underline{y}_i\right) log\left(1-\underline{\hat{y}}_i\right)\right). \tag{18}$$

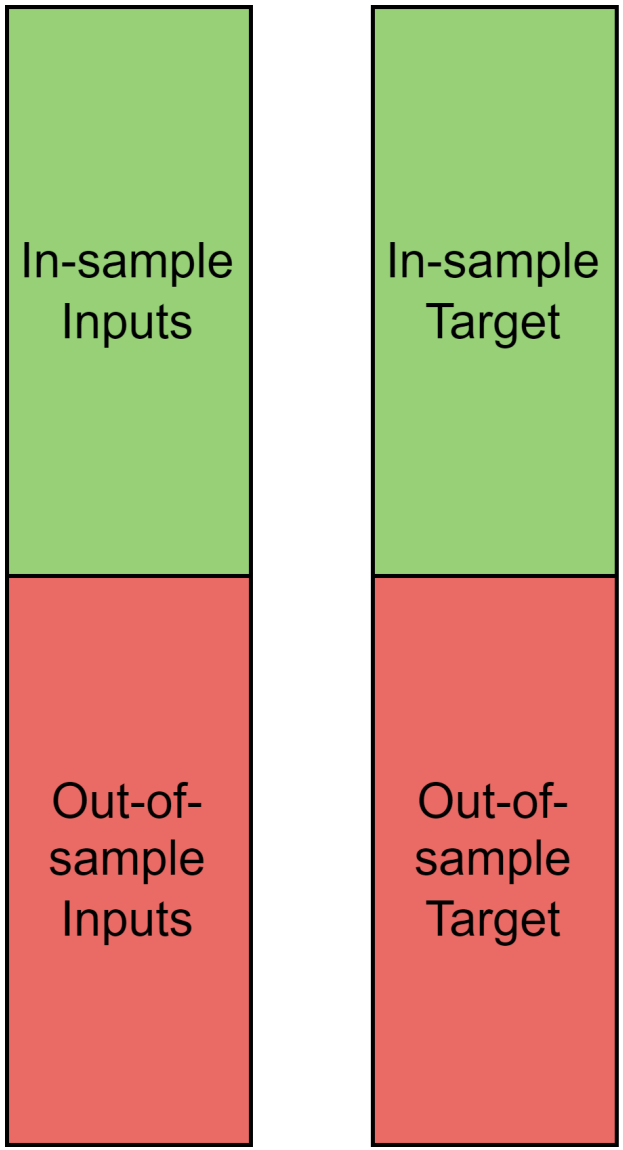
$$L_{NN} = L_e + \rho_1\|\underline{w}\|_1 + \rho_2\|\underline{w}\|_2^2, \text{where } \rho_1 \text{ and } \rho_2 \text{ are the loss weights.} \tag{19}$$

Table of Contents

Hyperparameters and Cross-validation

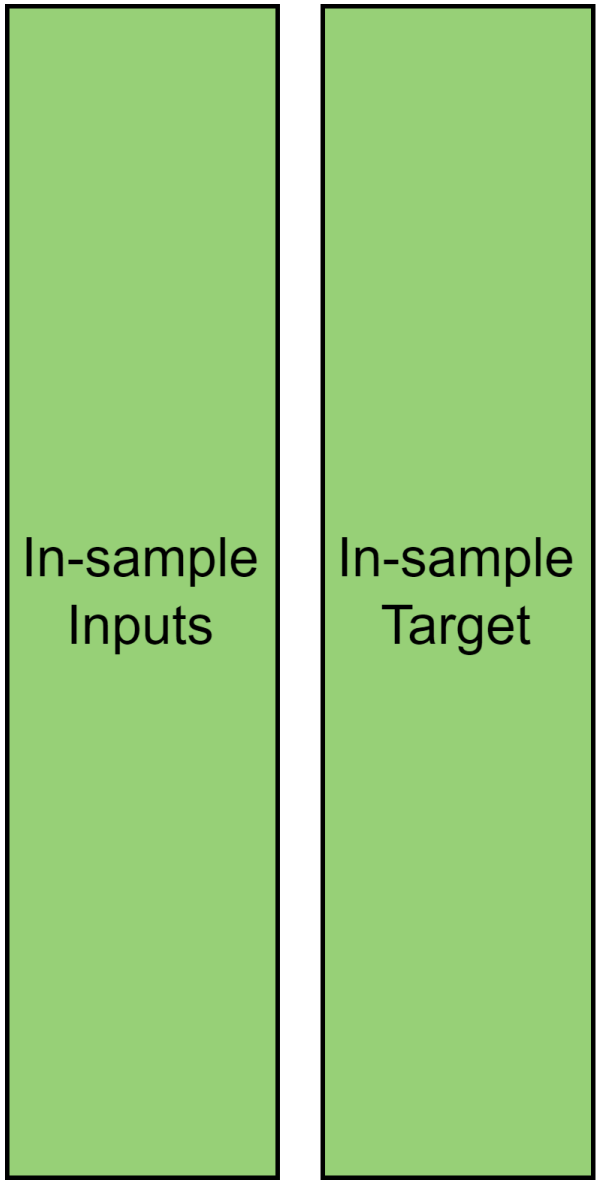
- Hyperparameters: number of hidden units, L1 and L2 regularisation weights
- We need a strategy to select and validate the hyperparameters, but we have limited data
- We can use the stratified k-fold cross-validation method
  - Divide the training data into k distinct subsets (fold)
    - Each fold should approximately be the same size
    - Each fold should have similar "statistics" e.g. means – stratification
  - Remove one subset and train on the rest
  - Apply the model on the removed subset and store the prediction $\hat{y}$
  - Replace Subset
  - Cycle through all k subsets
  - Compute the performance index for $\underline{y}$ and $\underline{\hat{y}}$

Cross-validation Example (1)

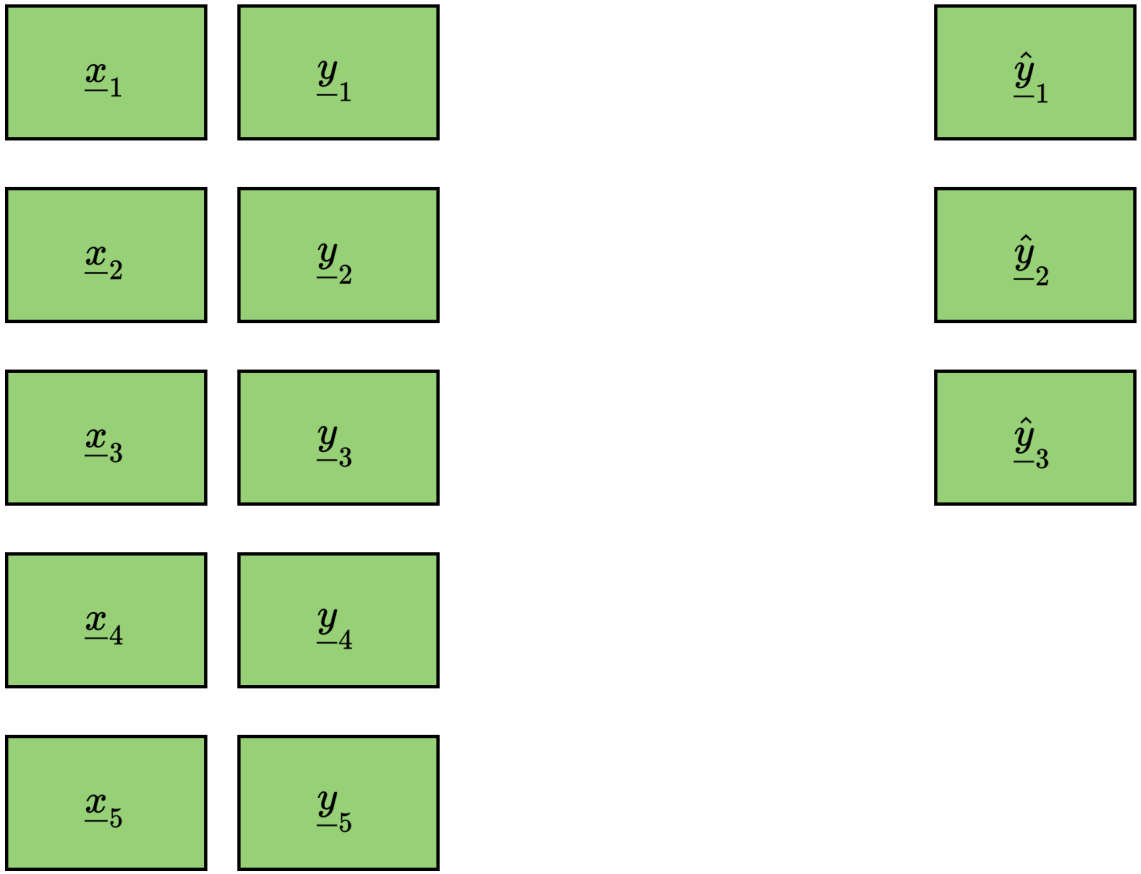


50% Split

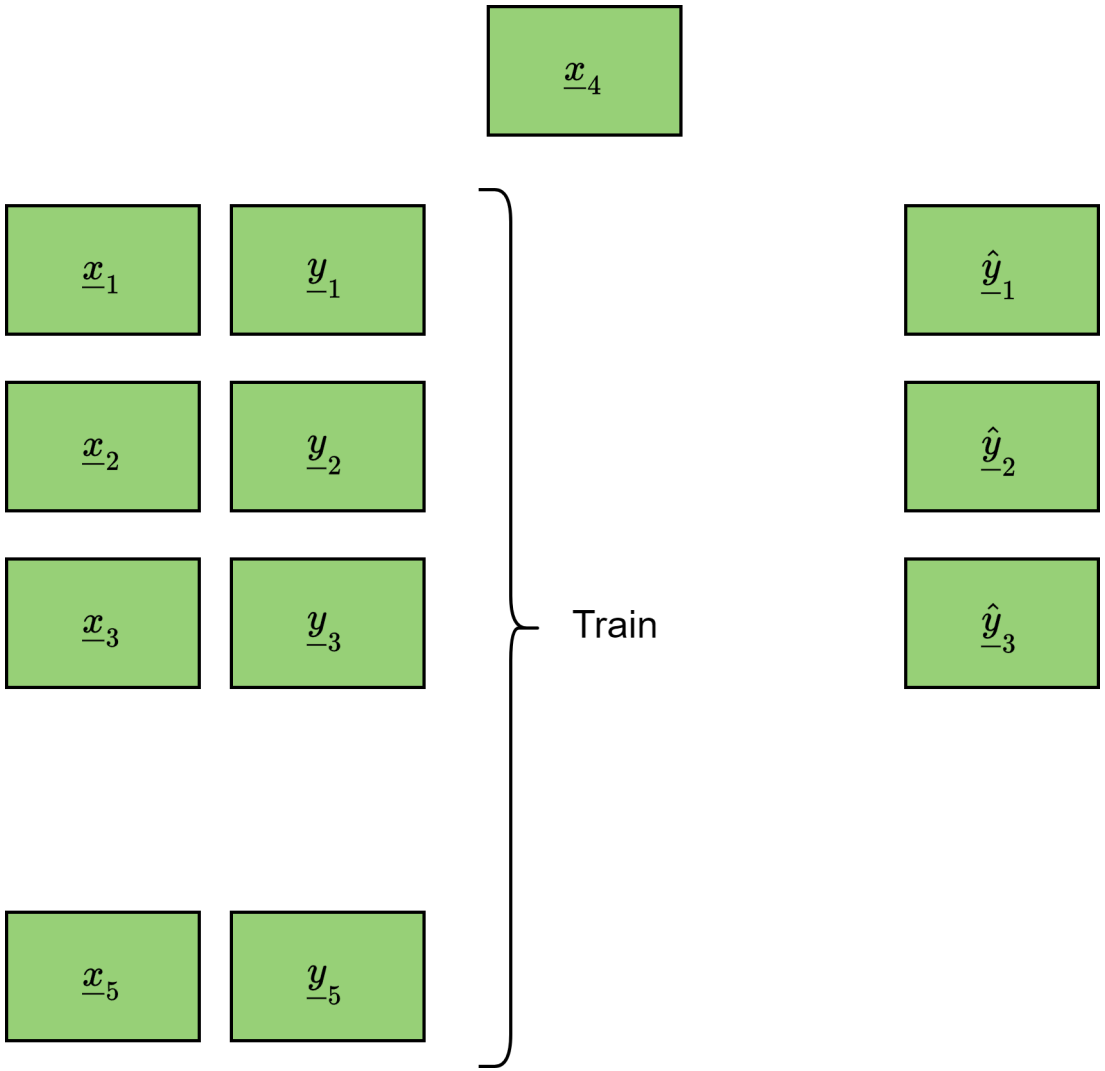
Cross-validation Example (2)



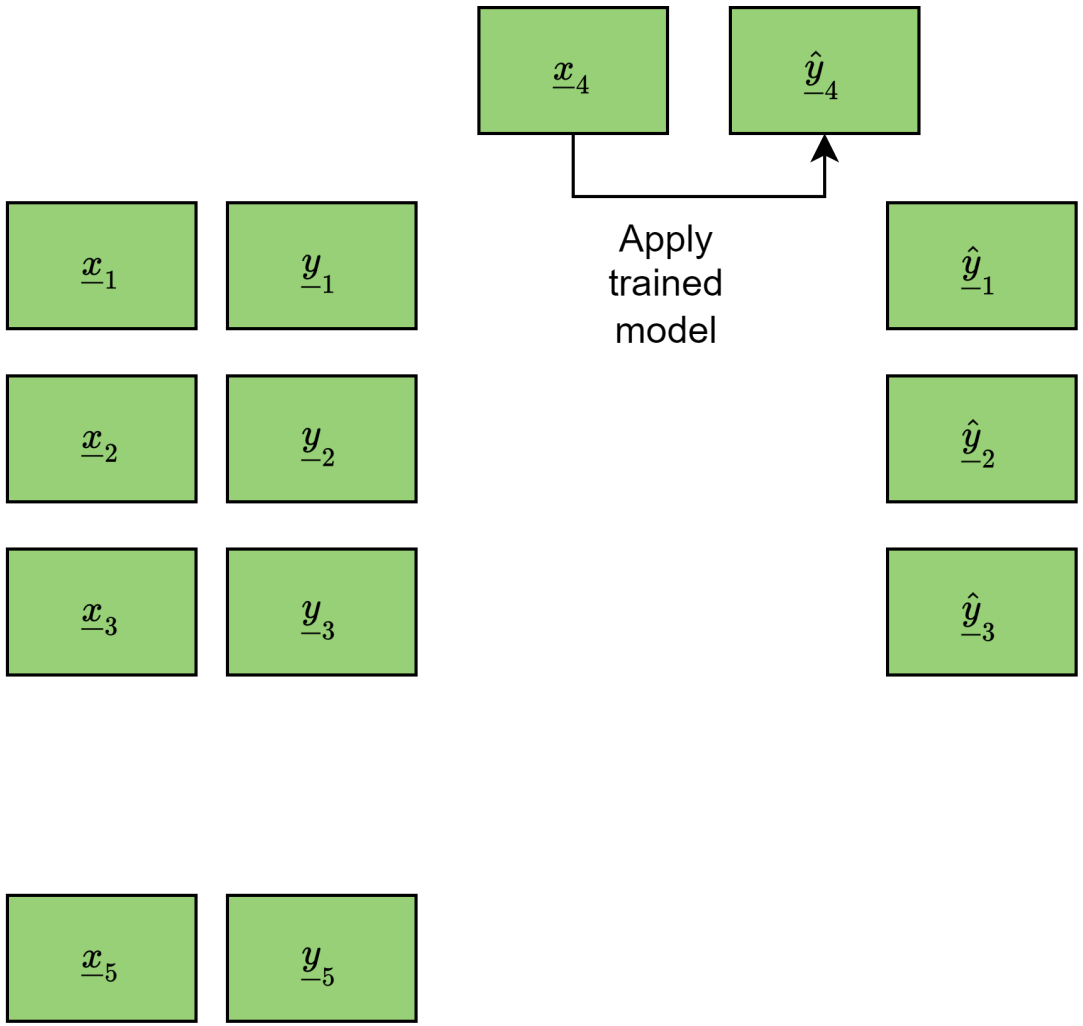
Cross-validation Example (3)



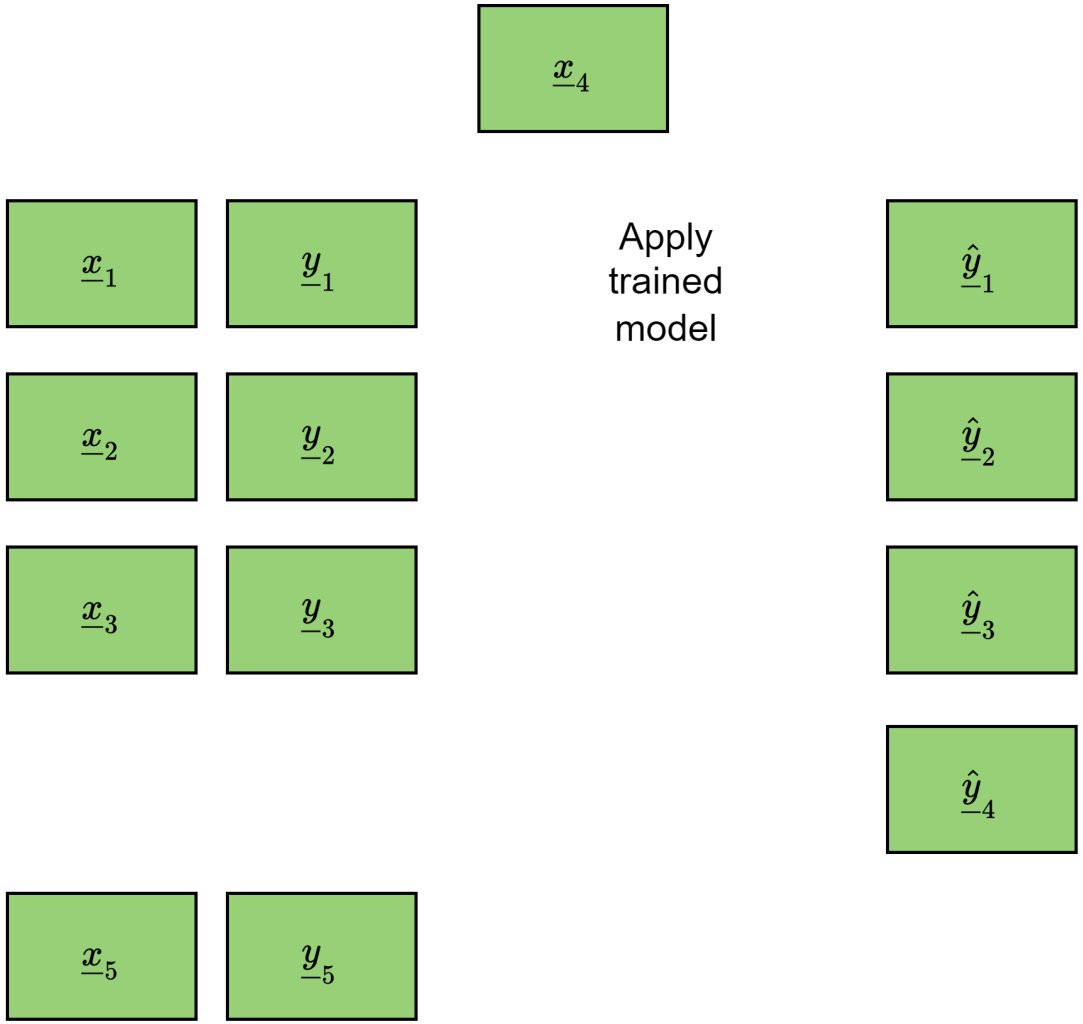
Cross-validation Example (4)



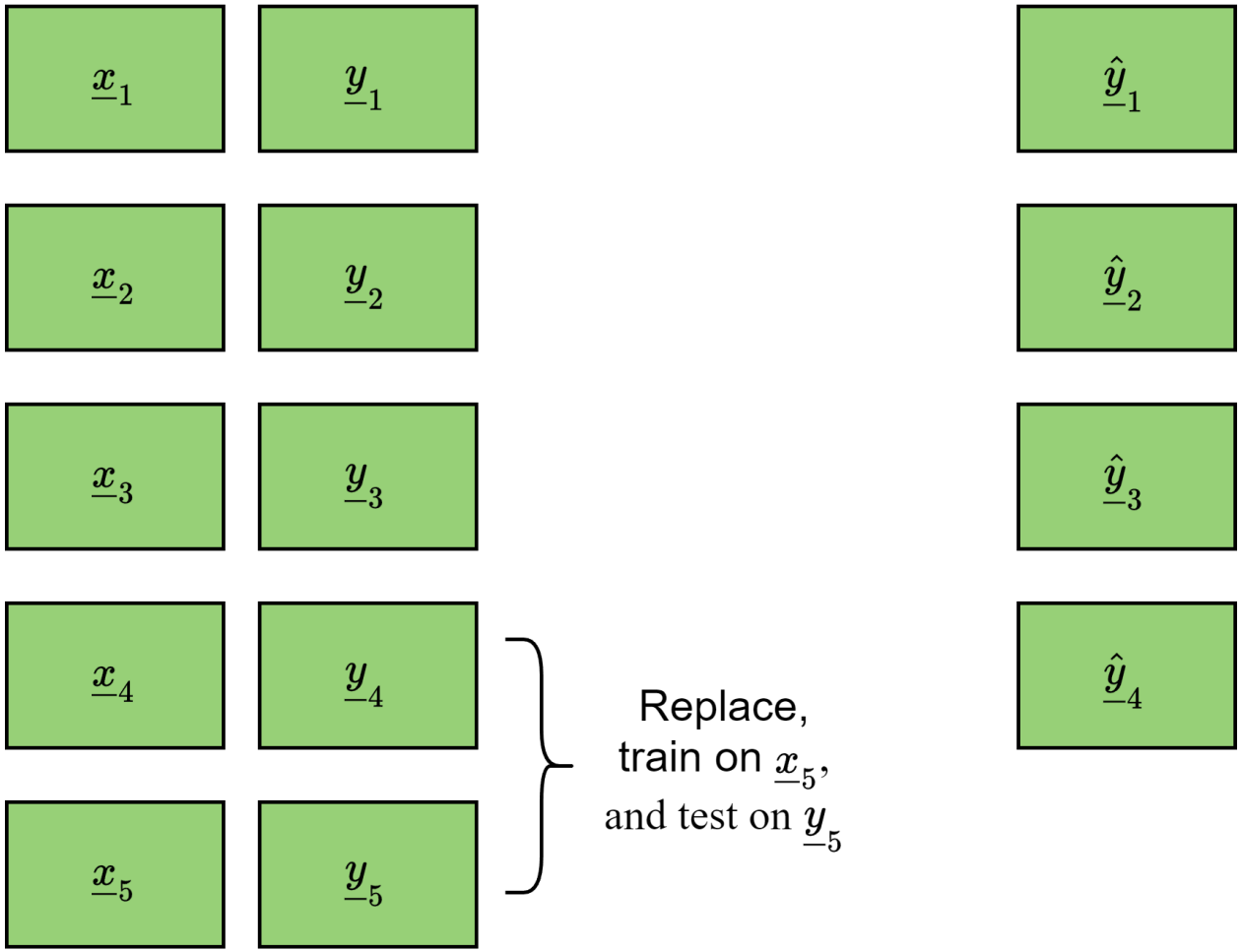
Cross-validation Example (5)



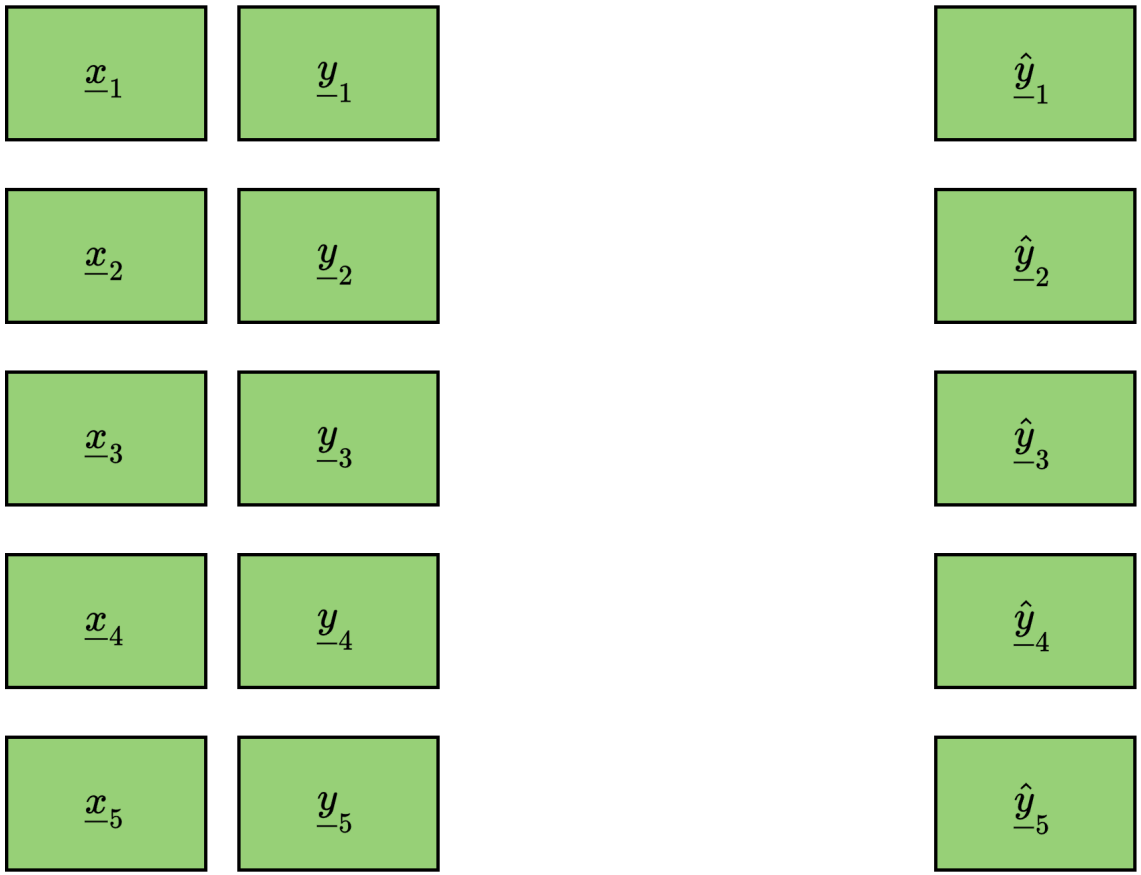
Cross-validation Example (6)



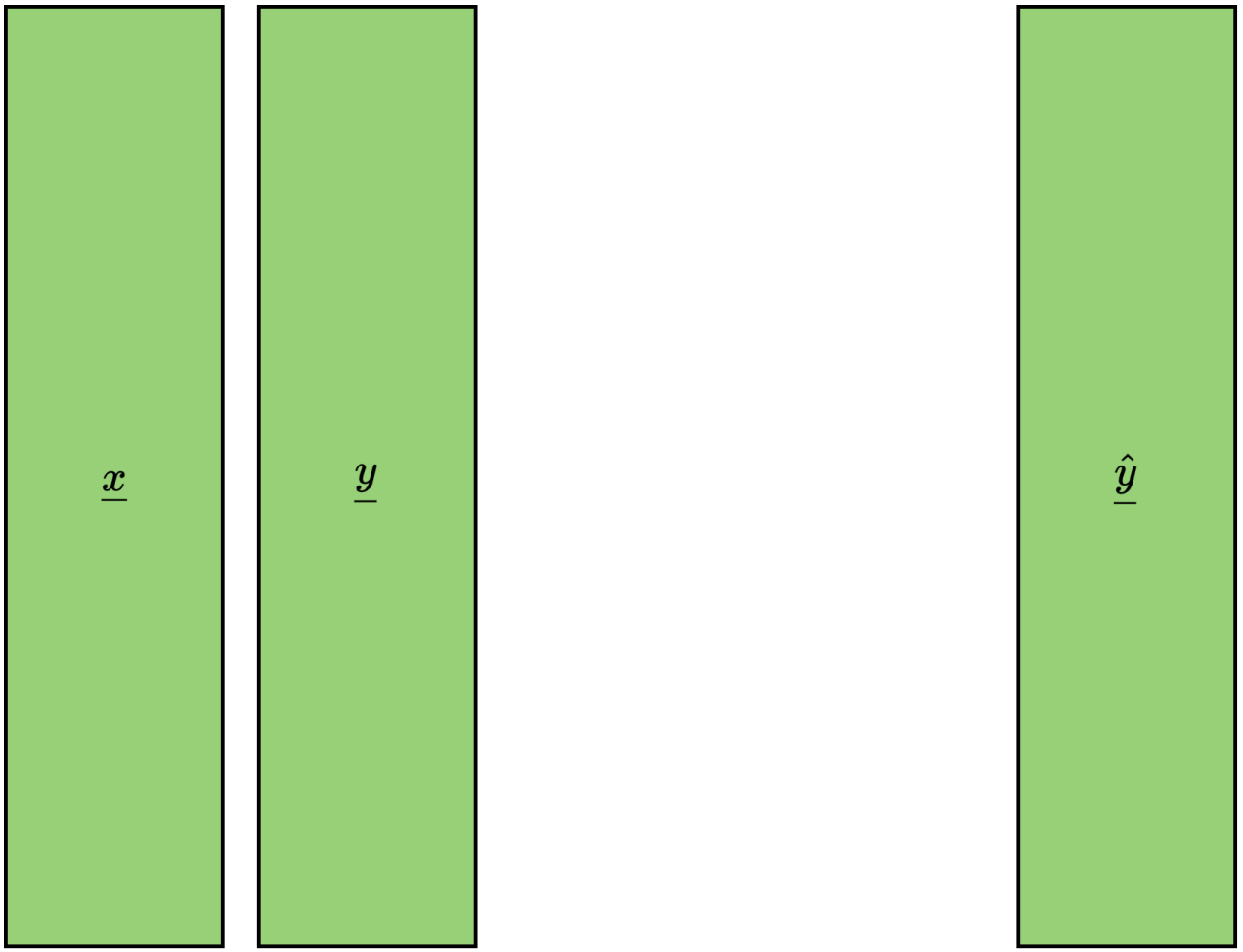
Cross-validation Example (7)



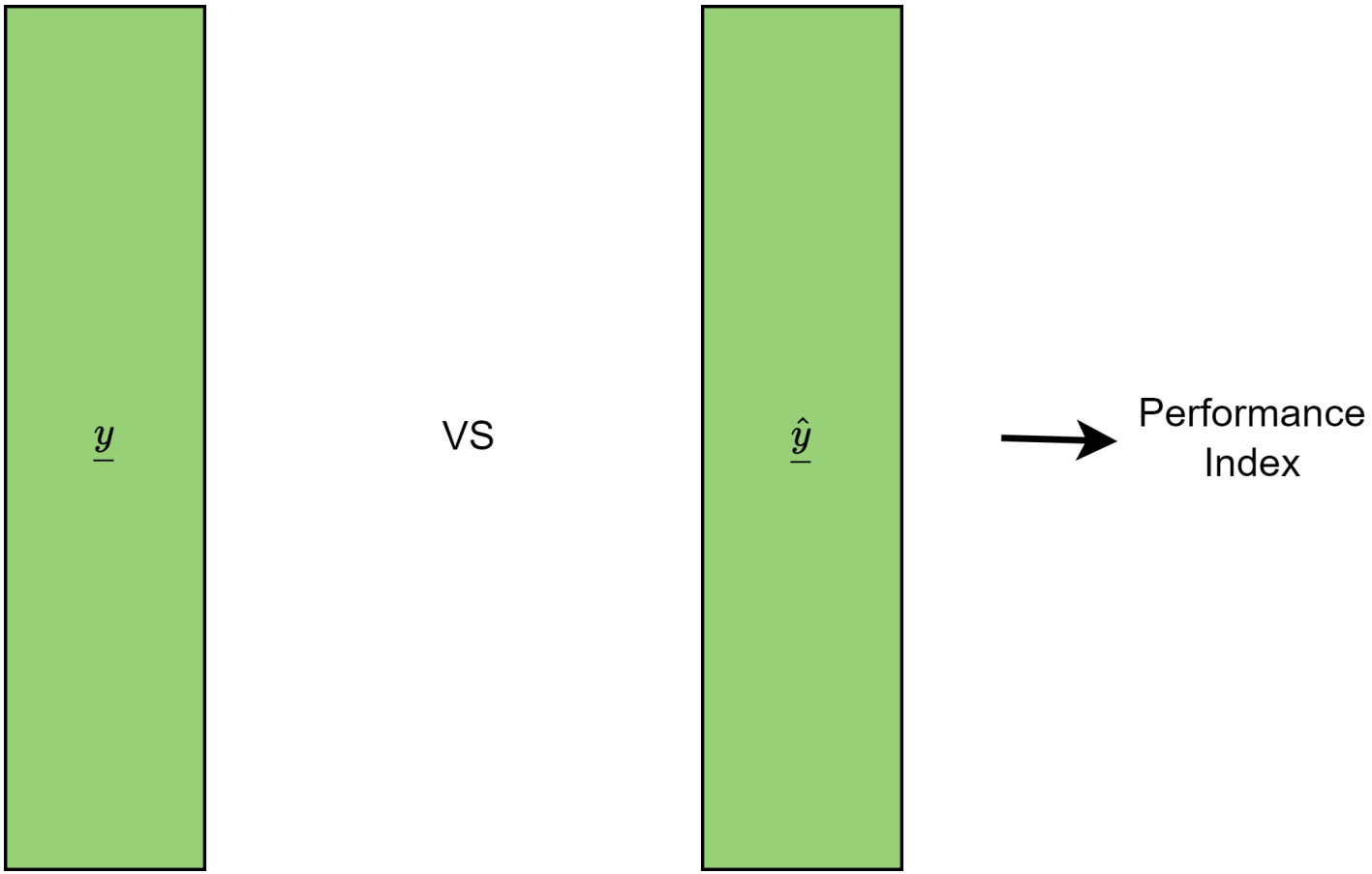
Cross-validation Example (8)



Cross-validation Example (9)



Cross-validation Example (10)



Cross-validation Example (11)

- The number of cross validation fold is a trade-off between bias and variance in the PI estimate
- An increase in the number of folds decreases bias but increases uncertainty
- k = 5 or 10 is standard, k = 3 for very large data sets
- We can repeat k-fold cross validation "repeated k-fold validation" by re-dividing the data in different subsets and doing an overall performance evaluation
- Repeating the k-fold cross validation at least three times is recommended
- K-fold cross validation tends to underestimate the "real-world" performance
- Once the repeated cross validation is completed and the hyperparameters are selected, we can train a model on the entire in-sample data set

Validation Performance Index

- Root Mean Square Error is one performance index that can be used for validation and testing in regression problems (Eq. 20)
- Cross-entropy cost is one performance index that can be used for validation and testing in classification problems (Eq. 21)
- Other metrics include the normalised sum of squared error, coefficient of determination, normalised sum of absolute error, and correlation.
- No universal "best"

$$E_e = \sqrt{\frac{\sum_{i=1}^{I}\left(\left(\underline{y}_i - \underline{\hat{y}}_i\right)^2\right)}{I}} \tag{20}$$

$$E_e = -\frac{1}{I}\sum_{i=1}^{I}\left(\underline{y}_i log\left(\underline{\hat{y}}_i\right) + \left(1 - \underline{y}_i\right) log\left(1 - \underline{\hat{y}}_i\right)\right) \tag{21}$$

Table of Contents

Design Strategy

- For 2 or 3 hyperparameters (low-dimension) optimisation we can perform a grid search
- Apply repeated k-fold cross-validation at each grid point (e.g. number of hidden points and L1 regularisation weight)
- Choose the number of hidden units and L1 regularisation weight with the best averaged PI
- Retrain on the entire in-sample training data set
- Re-initialise the neural network at least 10 times (ideally 50 for a statistical analysis) and report the average or median performance (depending on the metric) across the 10 neural networks

Table of Contents

Gradient-based Optimisatoin (1)

- The cost function is rarely convex
  - Multi-modal / non-smooth
  - No closed-form solution
- To optimise the neural network: iterative search
  - Gradient-based methods
  - Genetic algorithms, particle swarms
- A multi-modal cost function means the start point determines the end point
  - Local and global turning points
  - Convergence to a local minimum
  - Can not know if we have reached the global minimum

Gradient-based Optimisatoin (2)

- Gradient-based optimisation methods include: Gradient descent, (Scaled) Conjugate Gradients, Levenburg-Marquardt, Adam optimiser
- When selecting a gradient-based optimiser be aware of:
  - Convergence speed
  - Robustness to noise
  - Memory efficiency
- Be aware of ensuring the hidden layer activation function is compatible with the gradient-based optimisation method
  - For example, the leaky ReLu, Swish, and Scaled Exponential Linear Unit (Selu) activation functions allow a small non-zero gradient when the input is less than zero, which can cause issues when using the Scaled Conjugate Gradient (SCG) method

Table of Contents

Advanced Objectives

- If you complete the basic objectives, attempt the advanced objects to perform multi-class classification
- Note, in the advanced objectives we omit the porosity-free samples
- Once you covert the porosity volumes to a spherical equivalent diameter (SED), bin the SED values into three classes where class 0 represents $SED \leq 50.00\mu m$, class 1 represents $50.00 < SED \leq 100.00\mu m$, and class 2 represents $SED > 100.00\mu m$
- You will notice that there is a big data-imbalance
- Two novel questions to answer:
  - Why is it easy to correctly classify classes 0 and 2, but class 1 is usually miss-classified into class 2?
  - If we can not achieve a good average accuracy across all three classes, how can you re-frame the problem to successfully predict the porosity volume?

Table of Contents

Next Steps

- You should be meeting as a whole group 1-2 times per week
- You should have a a preliminary project plan and systems engineering plan
- You should now be working towards first full draft of requirements capture, and specifications
- Questions for industry/me, if any, should have been submitted last week, responses received this week
- Deadline: Week 7 Monday, 9am, Interim Report
- Today was the last seminar/direct help - you should be preparing now the preliminary technical solution/plan based on your literature review
- Next interaction is a 1:1 project management meeting