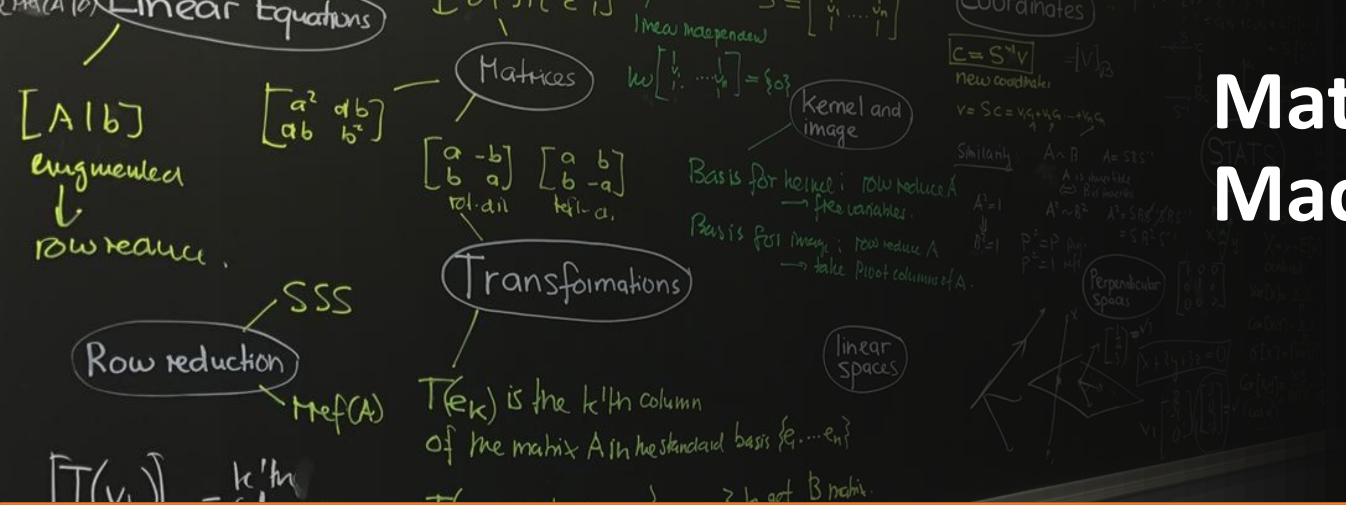


Mathematical Foundations for Machine Learning and Data Science

Supervised Learning

Dr. Zubair Khalid

Department of Electrical Engineering
School of Science and Engineering
Lahore University of Management Sciences

https://www.zubairkhalid.org/ee212_2021.html

LUMS
A Not-for-Profit University

Outline

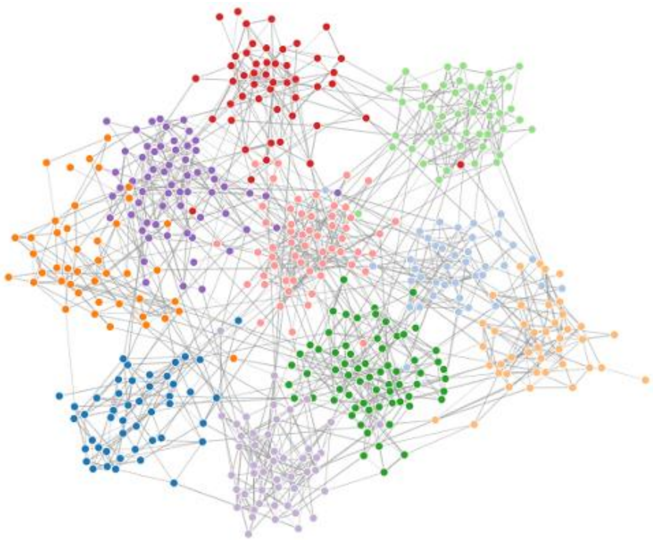
- *Supervised Learning Overview*
- *Formulation*
- *Train-Test Split*

Machine Learning: Overview

What is Machine Learning?

Given examples (training data), make a machine learn system behavior or discover patterns

Data



Algorithm

Model

$f(\mathbf{x})$

Given to us

We need to design it

Final output which enables us to make predictions

Machine Learning: Overview

Algorithms vs Model

- Linear regression algorithm produces a model, that is, a vector of values of the coefficients of the model.

- Decision tree algorithm produces a model comprised of a tree of if-then statements with specific values.

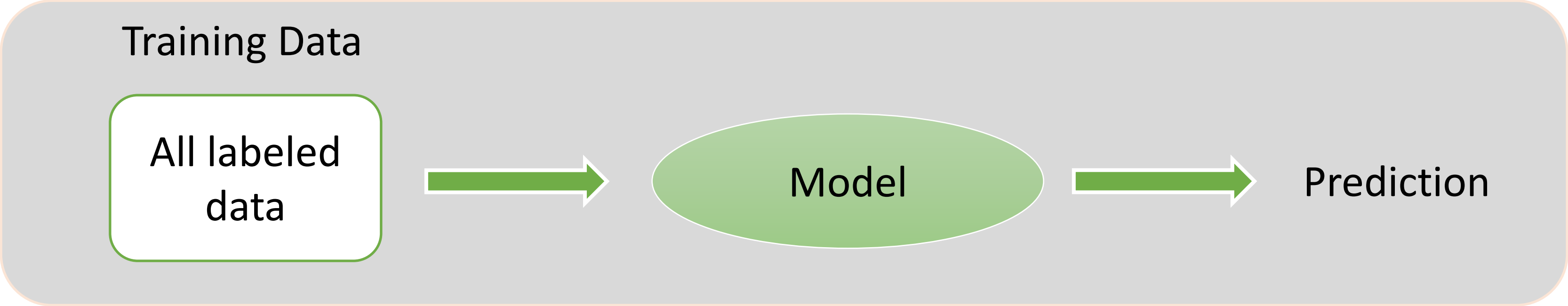
- Neural network along with backpropagation + gradient descent: produces a model comprised of a trained (weights assigned) neural network.

Machine Learning: Overview

Nature of ML Problems

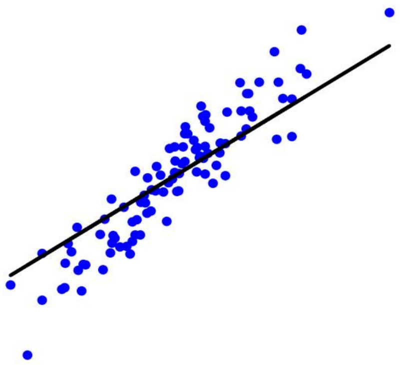
1. Supervised Learning

The learning algorithm would receive a set of inputs along with the corresponding correct outputs to train a model



Supervised Learning

Regression



Regression: Quantitative Prediction on a continuous scale

Examples: Prediction of

- Age of a person from his/her photo
- Price of 10 Marla, 5-bedroom house in 2050
- USD/PKR exchange rate after one week
- Efficacy of Pfizer Covid vaccine
- Average temperature/Rainfall during monsoon
- Cumulative score in this course
- Probability of decrease in the electricity prices in Pakistan
- No. of steps per day

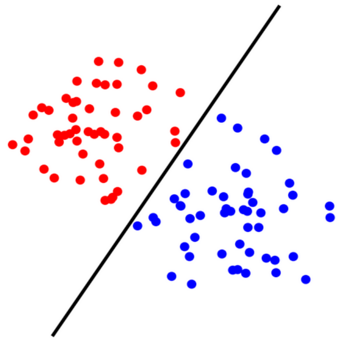
What do all these problems have in common?

Continuous outputs

Predicting continuous outputs is called regression

Supervised Learning

Classification



Classification: Given a data sample, predict its class (**discrete**)

Examples: Prediction of

- Gender of a person using his/her photo or hand-writing style
- Spam filtering
- Object or face detection in a photo
- We will be Covid-free in 2025
- Temperature/Rainfall normal or abnormal during monsoon
- Letter grade in ML course
- Decrease expected in electricity prices in Pakistan next year
- More than 10000 Steps taken today

What do all these problems have in common?

Discrete outputs: Categorical

Yes/No (Binary Classification)

Multi-class classification: multiple classes

Predicting a categorical output is called classification

Supervised Learning Setup

Nomenclature

In these regression or classification problems, we have

- Inputs – referred to as Features
- Output – referred to as Label
- Training data – (input, output) for which the output is known and is used for training a model by ML algorithm
- A Loss, an objective or a cost function – determines how well a trained model approximates the training data
- Test data – (input, output) for which the output is known and is used for the evaluation of the performance of the trained model

Supervised Learning Setup

Nomenclature - Example

Predict Stock Index Price

- *Features (Input)*
- *Labels (Output)*
- *Training data*

Interest_Rate	Unemployment_Rate	Stock_Index_Price
2.75	5.3	1464
2.5	5.3	1394
2.5	5.3	1357
2.5	5.3	1293
2.5	5.4	1256
2.5	5.6	1254
2.5	5.5	1234
2.25	5.5	1195
2.25	5.5	1159
2.25	5.6	1167
2	5.7	1130
2	5.9	1075
2	6	1047
1.75	5.9	965
1.75	5.8	943
1.75	6.1	958
1.75	6.2	971
1.75	6.1	949
1.75	6.1	884
1.75	6.1	866
1.75	5.9	876
1.75	6.2	?
1.75	6.2	?
1.75	6.1	?

Supervised Learning Setup

Formulation

We assume that we have d columns (features) of the input. In this example, we have two features; interest rate and unemployment rate, that is, $d = 2$.

In general, we use $\mathbf{x_i}$ to refer to featues of the i-th sample, that is,

$$\mathbf{x_i} = [x_{i,1}, x_{i,2}, x_{i,3}, \ldots x_{i,d}]$$

If y_i is the label associated with the i-th sample $\mathbf{x}_i$, we formulate training data in pairs as

$$(\mathbf{x_i}, y_i), \quad i = 1, 2, \ldots, n$$

Here, n denotes the number of samples in the training data. In this example, we have $n = 21$

Interest_Rate	Unemployment_Rate	Stock_Index_Price
2.75	5.3	1464
2.5	5.3	1394
2.5	5.3	1357
2.5	5.3	1293
2.5	5.4	1256
2.5	5.6	1254
2.5	5.5	1234
2.25	5.5	1195
2.25	5.5	1159
2.25	5.6	1167
2	5.7	1130
2	5.9	1075
2	6	1047
1.75	5.9	965
1.75	5.8	943
1.75	6.1	958
1.75	6.2	971
1.75	6.1	949
1.75	6.1	884
1.75	6.1	866
1.75	5.9	876
1.75	6.2	?
1.75	6.2	?
1.75	6.1	?

Supervised Learning Setup

Formulation

Using the adopted notation, we can formalize the supervised machine learning setup. We represent the entire training data as

$$D = \{(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)\} \subseteq \mathcal{X}^d \times \mathcal{Y}$$

Here $\mathcal{X}^d$ - d dimensional feature space and $\mathcal{Y}$ is the label space.

Regression: $\mathcal{Y} = \mathbf{R}$ (prediction on continuous scale)

Classification: $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, 1\}$ or $\mathcal{Y} = \{1, 2\}$ (Binary classification)

$\mathcal{Y} = \{1, 2, \ldots, M\}$ (M-class classification)

Supervised Learning Setup

Example

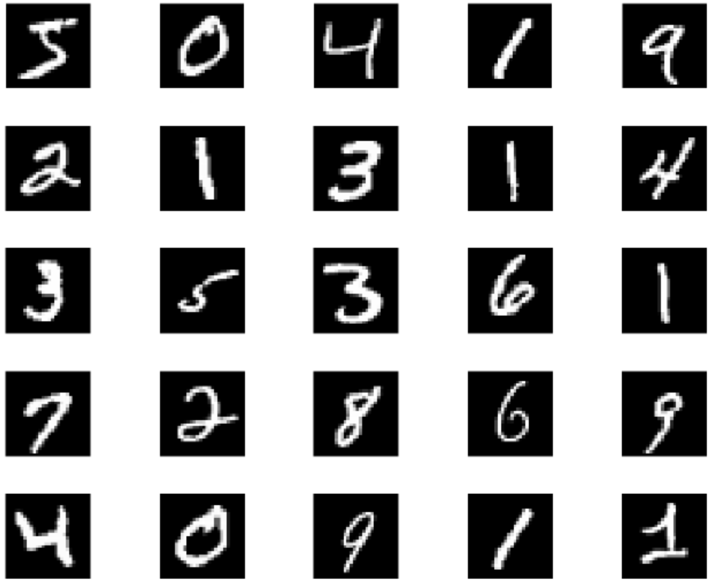
Data of 200 Patients:

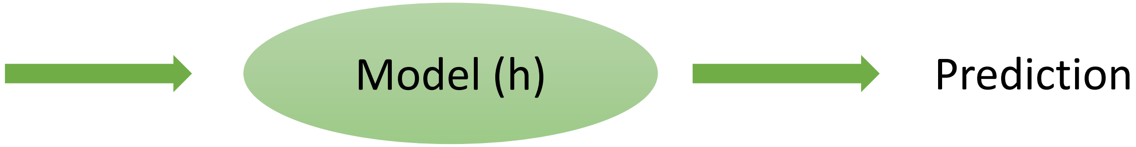
- Age of the patient
- Cholesterol levels
- Glucose levels
- BMI
- Height
- Heart Rate
- Calories intake
- No. of steps taken

→ Model (h) → Prediction of Oxygen Saturation

Supervised Learning Setup

Example





MNIST Data:

- Each sample 28x28 pixel image
- 60,000 training data
- 10,000 testing data

Supervised Learning Setup

Learning

Recall a problem in hand. We want to develop a model that can predict the label for the input for which label is unknown.

We assume that the data points $(\mathbf{x_i}, y_i)$ are drawn from some $\boxed{\text{(unknown)}}$ distribution $P(X, Y)$.

Our goal is to learn the machine (model, function or hypothesis) h such that for a new pair $(\mathbf{x}, y)$ P, we can use h to obtin

$$h(\mathbf{x}) = y$$

with high probability or

$$h(\mathbf{x}) \approx y$$

in some optimal sense.

Supervised Learning Setup

Hypothesis Class

We call the set of possible functions or candidate models (linear model, neural network, decision tree, etc.) "the hypothesis class".

Denoted by $\mathcal{H}$

For a given problem, we wish to select hypothesis (machine) $h \in \mathcal{H}$.

Q: How?

A: Define hypothesis class $\mathcal{H}$ for a given learning problem.

Evaluate the performance of each candidate function and choose the best one.

Supervised Learning Setup

Q: How do we evaluate the performance?

A: Define a loss function to quantify the accuracy of the prediction.

Loss Function

Loss function should quantify the error in predicting y using hypothesis function h and input $\mathbf{x}$.

Denoted by $\mathcal{L}$.

Supervised Learning Setup

0/1 Loss Function:

Zero-one loss is defined as

$$\mathcal{L}_{0/1}(h) = \frac{1}{n}\sum_{i=1}^{n} 1 - \delta_{h(\mathbf{x_i})-y_i}$$

Here $\delta_{h(\mathbf{x_i})-y_i}$ is the delta function defined as

$$\delta_k = \begin{cases} 1, & k = 0 \\ 0 & \text{otherwise} \end{cases}$$

Interpretation:

- *Note normalization by the number of samples. This makes it the loss per sample.*
- *Loss function counts the number of mistakes made by hypothesis function D.*
- ***Not used frequently due to non-differentiability and non-continuity.***

Supervised Learning Setup

Squared Loss Function:

Squared loss is defined as (also referred to as **mean-square error, MSE**)

$$\mathcal{L}_{\text{sq}}(h) = \frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x_i}) - y_i\right)^2$$

Interpretation:

- Again note normalization by the number of samples.
- Loss is always nonnegative.
- Loss grows quadratically with the absolute error amount in each sample.

Root Mean Squared Error (RMSE):

RMSE is just square root of squared loss function:

$$\mathcal{L}_{\text{rms}}(h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x_i}) - y_i\right)^2}$$

Supervised Learning Setup

Absolute Loss Function:

Absolute loss is defined as

$$\mathcal{L}_{\text{abs}}(h) = \frac{1}{n}\sum_{i=1}^{n} |h(\mathbf{x_i}) - y_i|$$

Interpretation:

- Loss grows linearly with the absolute of the error in each prediction.
- Used in regression and suited for noisy data.

* All of the losses are non-negative

Supervised Learning Setup

Learning

We wish to select hypothesis (machine) $h \in \mathcal{H}$ such that

$$h^* = \min_{h \in \mathcal{H}} \mathcal{L}(h)$$

(Optimization problem)

Recall We assume that the data points $(\mathbf{x_i}, y_i)$ are drawn from some $\boxed{\text{(unknown)}}$ distribution $P(X, Y)$.

We can come up with a function h after solving this minimization problem that gives low loss on our data.

Q: How can we ensure that hypothesis *h* will give low loss on the input not in *D*?

Supervised Learning Setup

To illustrate this, let us consider a model h trained on every input in D, that is, giving zero loss. Such function is referred to as memorizer and can be formulated as follows

$$h(\mathbf{x}) = \begin{cases} y_i, & \exists\ (\mathbf{x}_i, y_i) \in D, \quad \mathbf{x}_i = \mathbf{x}, \\ 0, & \text{otherwise} \end{cases}$$

Interpretation:

- *0% loss error on the training data (Model is fit to every data point in D).*
- *Large error for some input not in D*
- *First glimpse of **overfitting.***

Revisit:

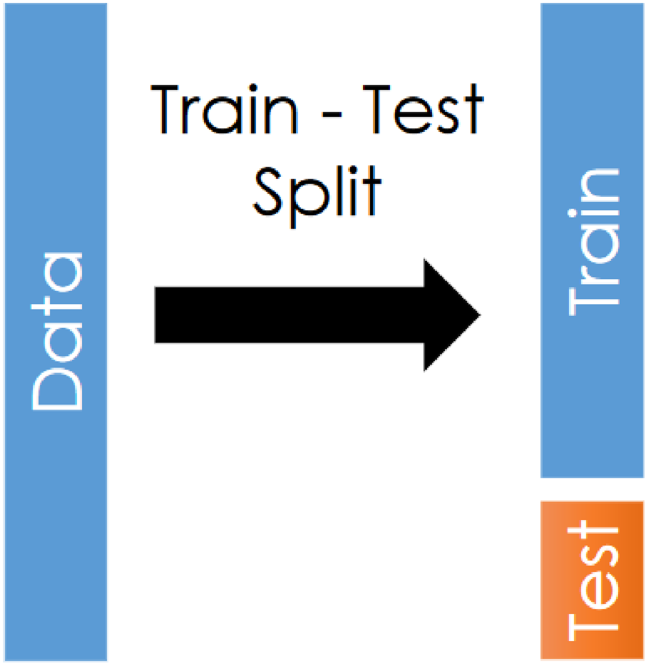
Q: How can we ensure that hypothesis *h* will give low loss on the input not in *D*?

A: Train/Test Split

Supervised Learning Setup

Generalization: The Train-Test Split

To resolve the overfitting issue, we usually split D into train and test subsets:



- D_{TR} as the training data, (70, 80 or 90%)
- D_{TE} as the test data, (30, 20, or 10%)

How to carry out splitting?

- Split should be capturing the variations in the distribution.
- Usually, we carry out splitting using i.i.d. sampling and time series with respect to time

You can only use the test dataset once after deciding on the model using training dataset

Supervised Learning Setup

Learning (Revisit after train-test split)

We had the following optimiztion problem as

$$h^* = \min_{h \in \mathcal{H}} \mathcal{L}(h)$$

We generalize it as

$$h^* = \min_{h \in \mathcal{H}} \frac{1}{|D_{\mathrm{TR}}|} \sum_{(\mathbf{x},y) \in D_{\mathrm{TR}}} \mathcal{L}(\mathbf{x}, y)|h)$$

Evaluation

Loss on the testing data is given by

$$\epsilon_{\mathrm{TE}} = \frac{1}{|D_{\mathrm{TE}}|} \sum_{(\mathbf{x},y) \in D_{\mathrm{TE}}} \mathcal{L}(\mathbf{x}, y)|h*)$$

Supervised Learning Setup

Generalization loss

We define the generalized loss on the distribution P from which the D is drawn as the expected value (average value, probability weighted average to be precise) of the loss for a given h^* s

$$\epsilon = E\big[\mathcal{L}(\mathbf{x}, y|h^*)\big]$$

The expectation here is over the distribution P of $(\mathbf{x}, y)$.

Under the assumption that data D is i.i.d (independent and identically distributed) drawn from P, ϵ_{TE} serves as an unbiased estimator of the generalized loss ϵ. This simply means ϵ_{TE} converges to ϵ with the increase in the data size, that is,

$$\lim_{n \to \infty} \epsilon_{\mathrm{TE}} = \epsilon.$$

Supervised Learning Setup

Generalization: The Train-Test Split

At times, we usually split D into three subsets, that is, the training data is further divided into traaining and validation datasets:

- D_{TR} as the training data, (80%)
- D_{VA} as the validation data, (10%)
- D_{TE} as the test data, (10%)

Q: Idea:

Validation data is used to evaluate the loss for a function h that is determined using the learning on the training data-set. If the loss on validation data is high for a given h, the hypothesis or model needs to be changed.

Supervised Learning Setup

Generalization: The Train-Test Split

More explanation* to better understand the difference between validation and test data:

- **Training set:** A set of examples used for learning, that is to fit the parameters of the hypothesis (model).

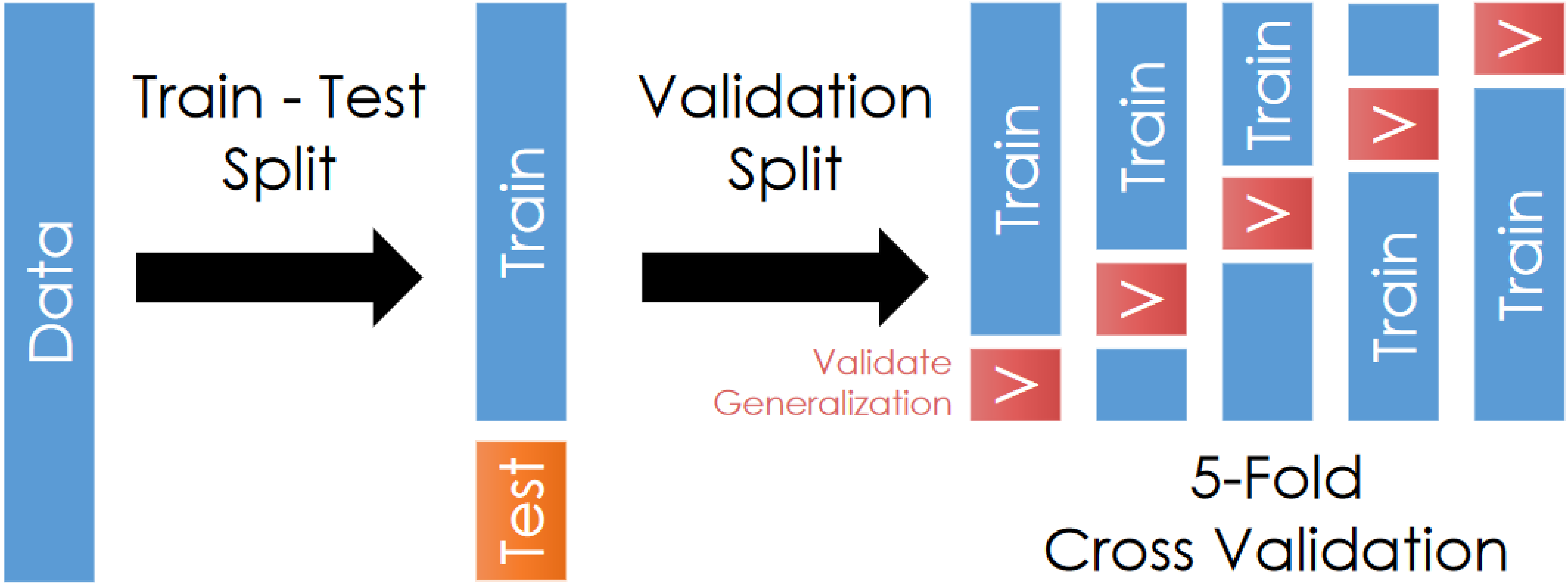
- **Validation set:** A set of examples used to tune the hyper-parameters of the hypothesis function, for example to choose the number of hidden units in a neural network OR the order of polynomial approximating the data.

- **Test set:** A set of examples **used** only to assess the performance of a fully-specified model or hypothesis.

Adapted from *Brian Ripley, Pattern Recognition and Neural Networks, 1996

Supervised Learning Setup

Generalization: The Train-Test Split (Example)



Cross validation simulates multiple train-test splits on the training data