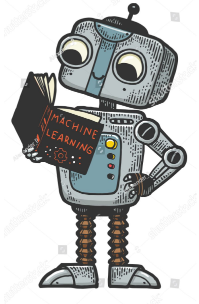
Introduction to Supervised Learning

MAIN educational 2021

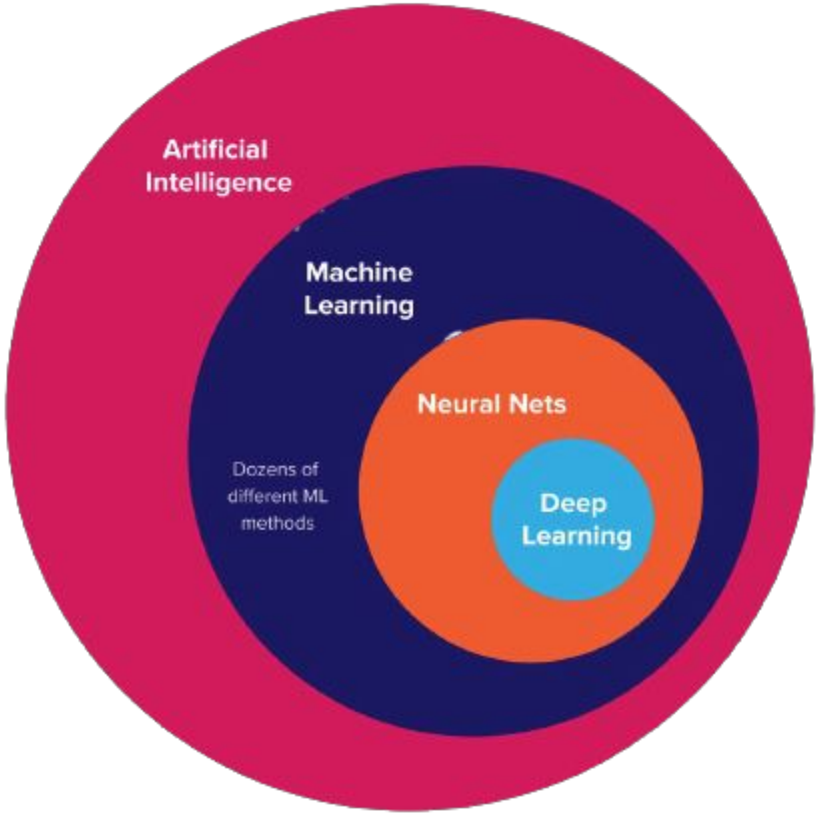
By
Nikhil Bhagwat & Jérôme Dockès

25 November 2021



The Basic Questions

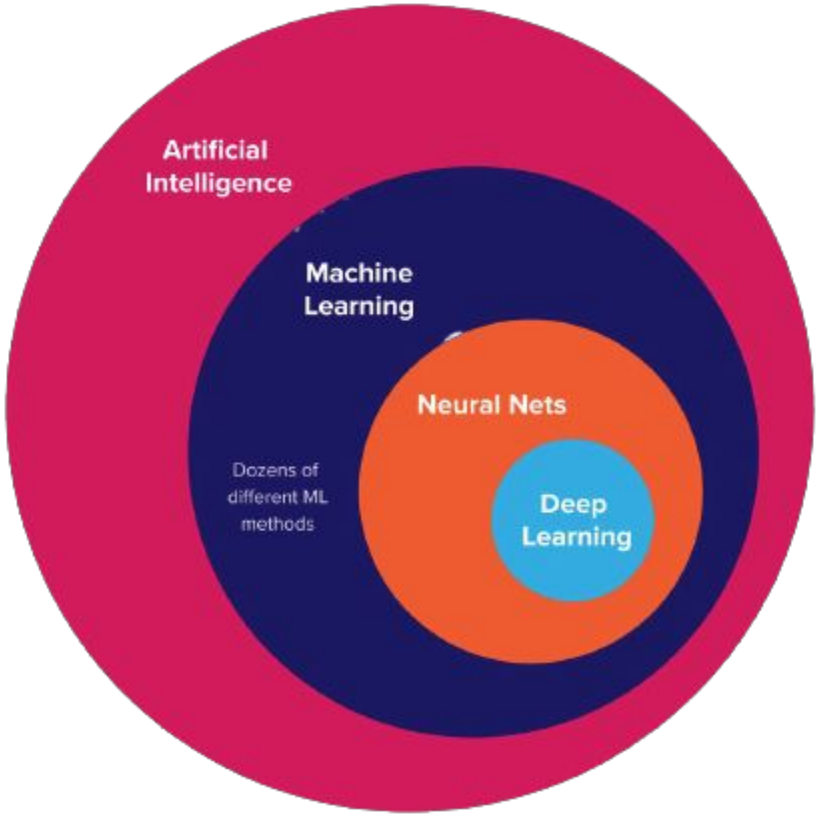
- What is Machine learning (ML)?
    - ML is the study of computer algorithms that improve automatically through experience and by the use of data.



The Basic Questions

- What is Machine learning (ML)?
  - ML is the study of computer algorithms that improve automatically through experience and by the use of data.

- Why is it useful - especially in life sciences?
  - Biology, Medicine, Environmental sciences comprise phenomenons (e.g. a disease) with large number of variables.
  - We want to model complex relationships within these variables.
  - ML can help in these cases and provide accurate predictions.

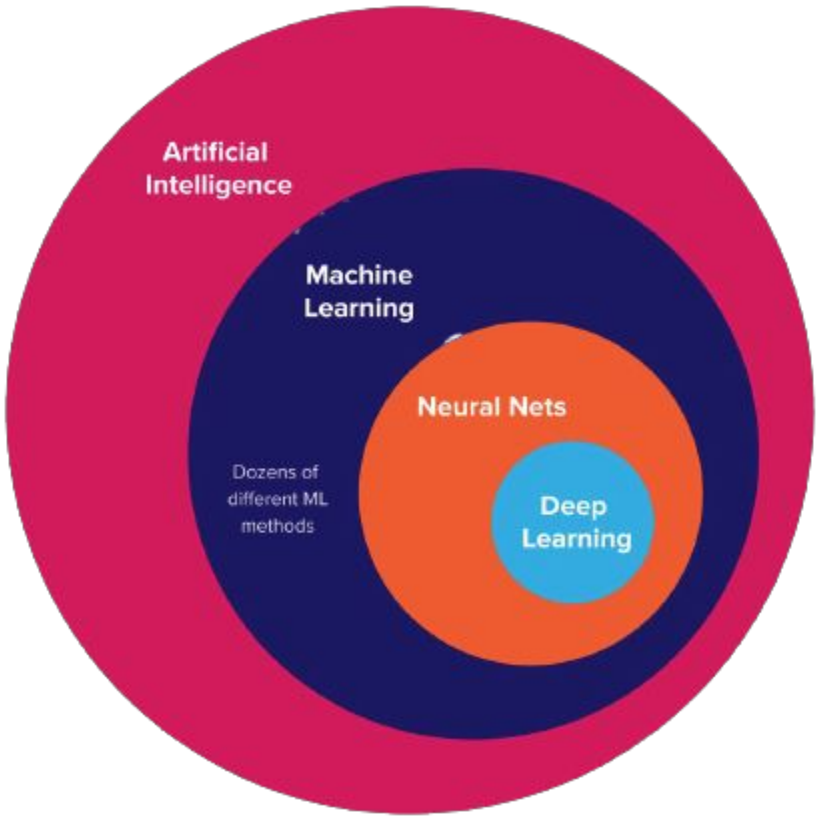


The Basic Questions

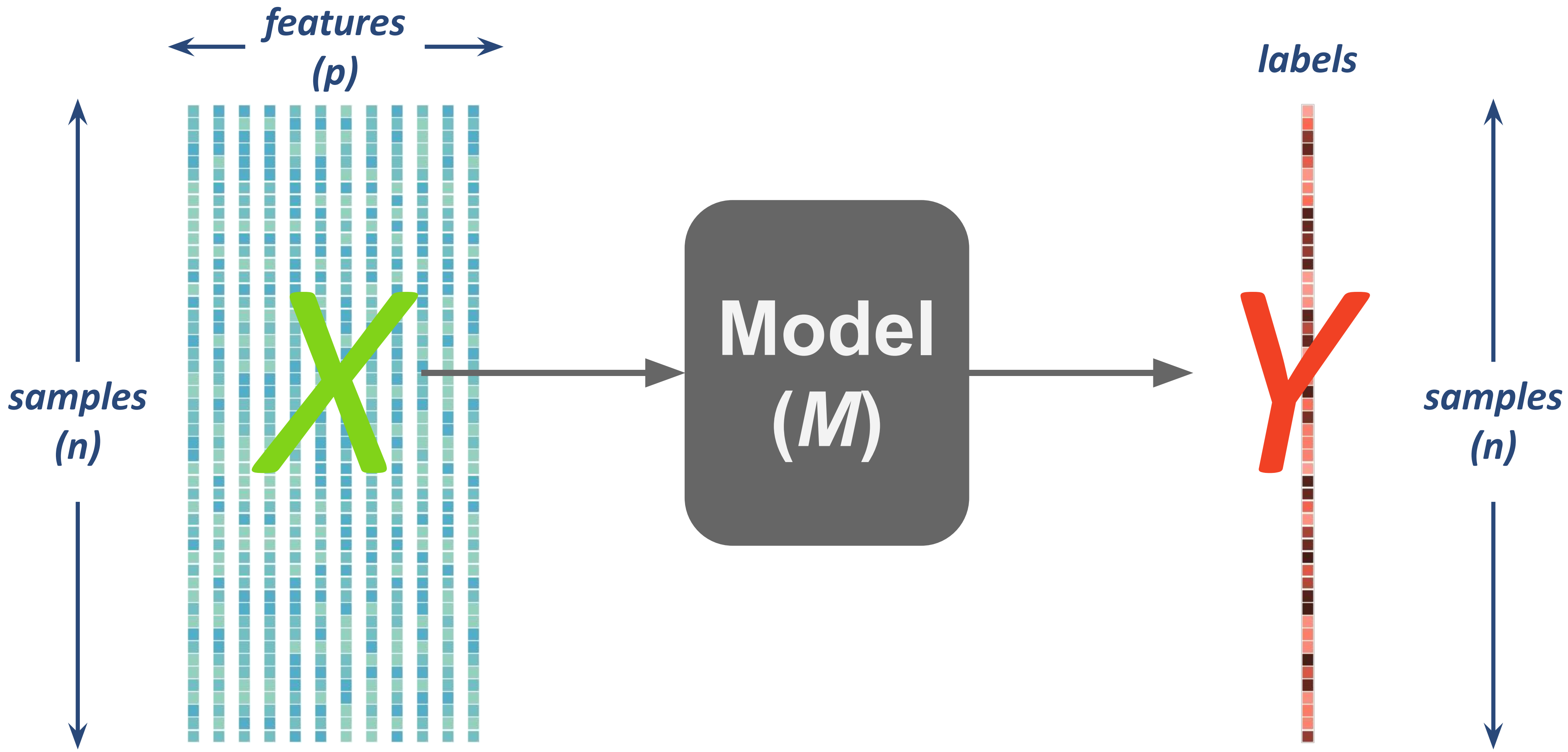
- What is Machine learning (ML)?
  - ML is the study of computer algorithms that improve automatically through experience and by the use of data.

- Why is it useful - especially in life sciences?
  - Biology, Medicine, Environmental sciences comprise phenomenons (e.g. a disease) with large number of variables.
  - We want to model complex relationships within these variables.
  - ML can help in these cases and provide accurate predictions.

- When do I use it?
  - You are interested in 1) prediction tasks or 2) low-dimensional representation.
  - You have sufficient data.



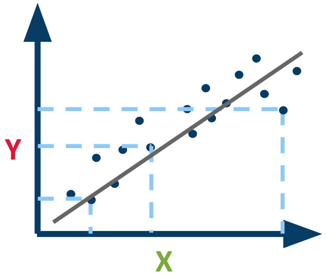
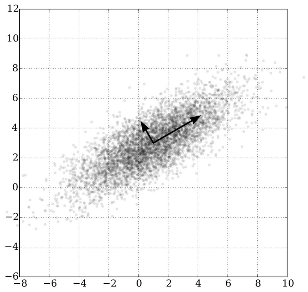
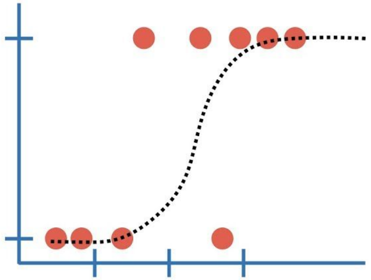
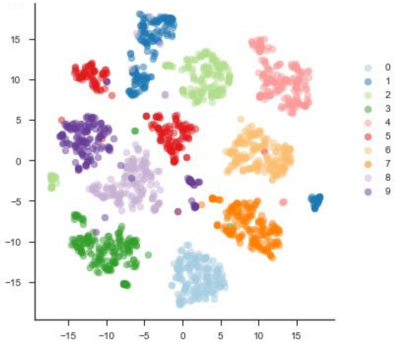
Terminology



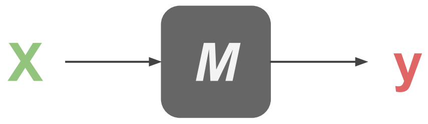
Input
Examples:  

Output Examples: Clinical measures

Types of ML Algorithms

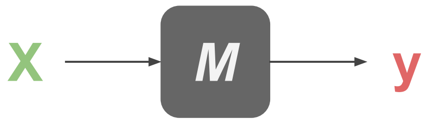
<table>
<tr><th>Outcome</th><th>Supervised Learning</th><th>Unsupervised Learning</th></tr>
<tr><td>Continuous</td><td>Regression
</td><td>Dimensionality reduction
</td></tr>
<tr><td>Categorical</td><td>Classification
</td><td>Clustering
</td></tr>
</table>

Supervised Learning

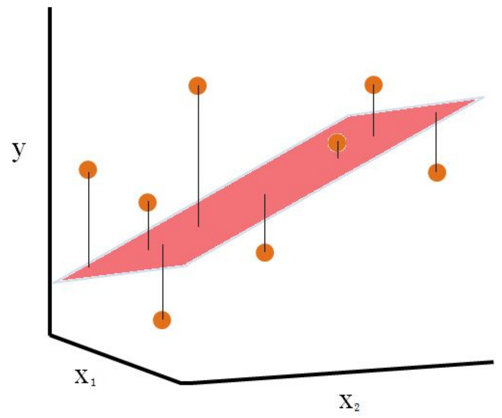


- Goal: *Learn* parameters (or weights) of a model (M) that maps X to y

Supervised Learning



- Goal: *Learn* parameters (or weights) of a model (M) that maps X to y
- Example models:
  - Linear / Logistic regression

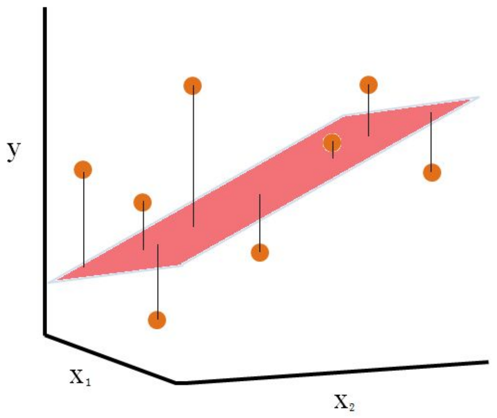


Linear Regression

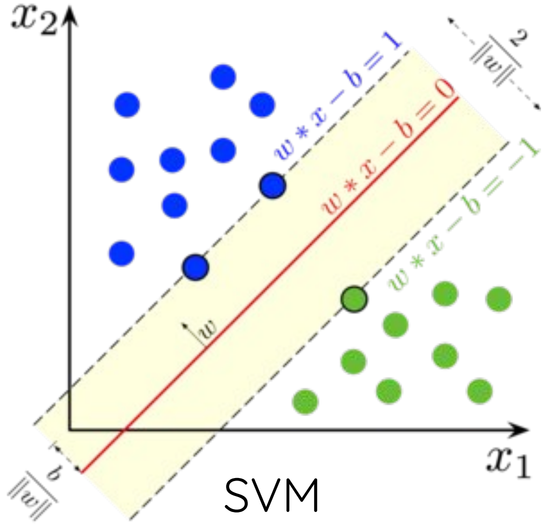
Supervised Learning

$x \rightarrow M \rightarrow y$

- Goal: *Learn* parameters (or weights) of a model (M) that maps X to y
- Example models:
  - Linear / Logistic regression
  - Support vector machines

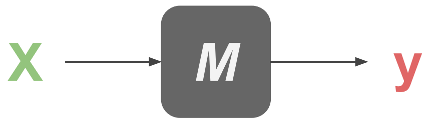


Linear Regression

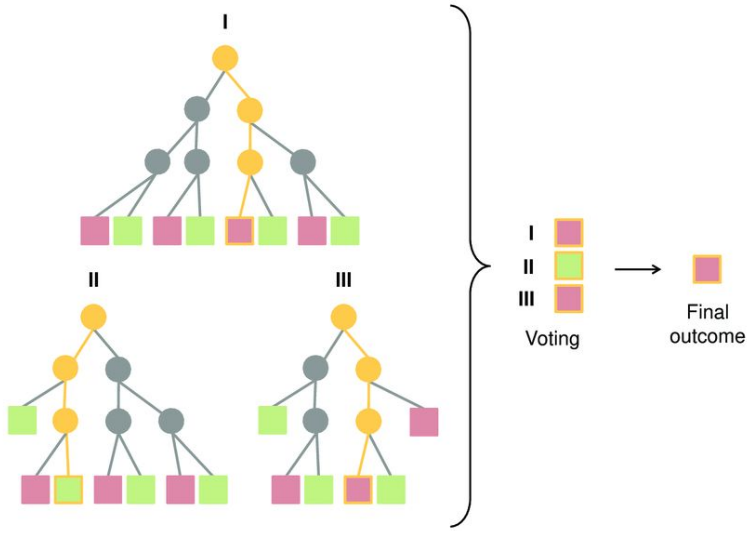


SVM

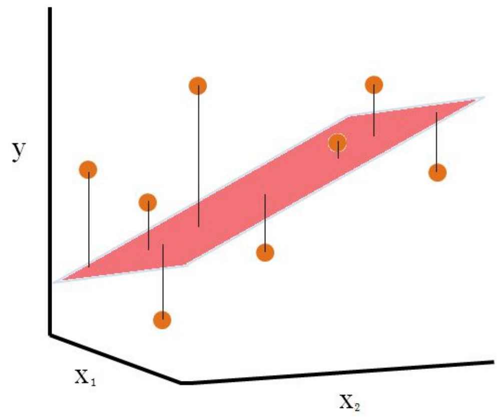
Supervised Learning



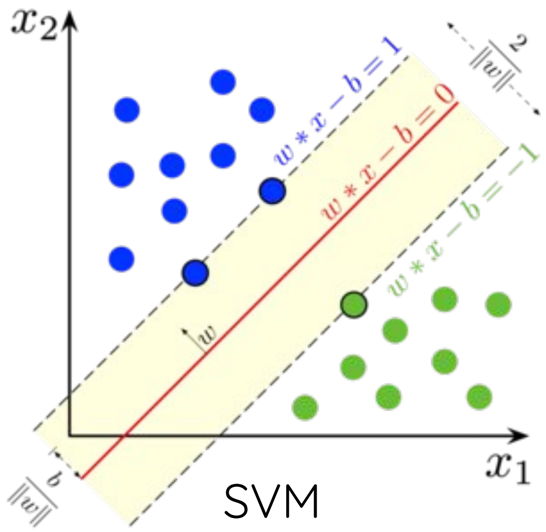
- Goal: *Learn* parameters (or weights) of a model (M) that maps X to y
- Example models:
  - Linear / Logistic regression
  - Support vector machines
  - Tree-ensembles: random forests, gradient boosting



Tree-ensembles



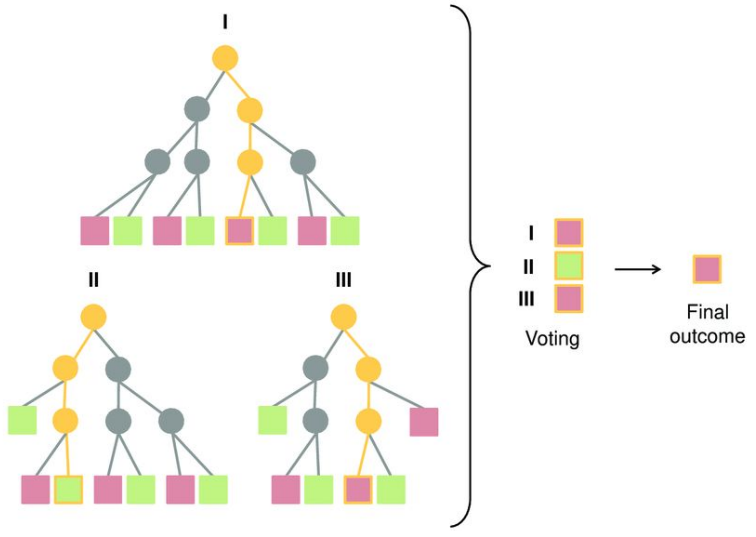
Linear Regression



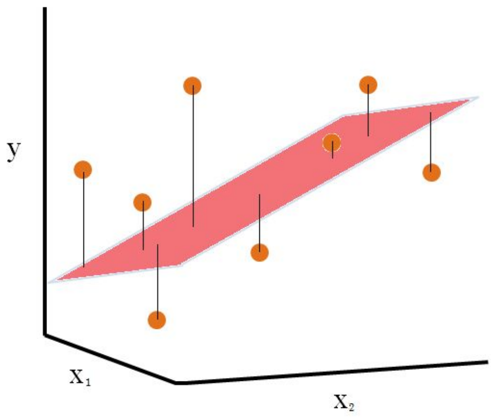
SVM

Supervised Learning $x \rightarrow \boxed{M} \rightarrow y$

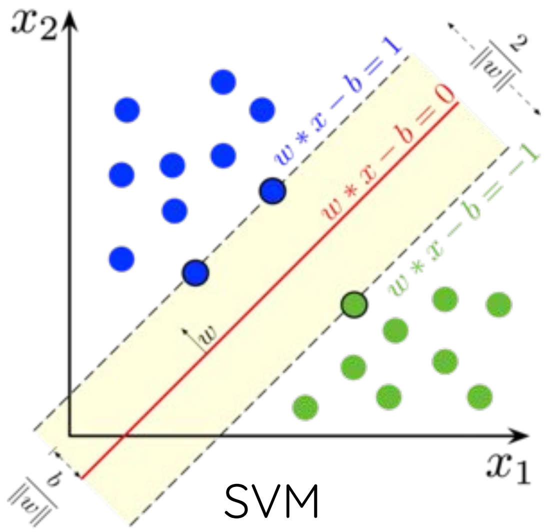
- Goal: *Learn* parameters (or weights) of a model (M) that maps X to y
- Example models:
  - Linear / Logistic regression
  - Support vector machines
  - Tree-ensembles: random forests, gradient boosting
  - Artificial Neural networks



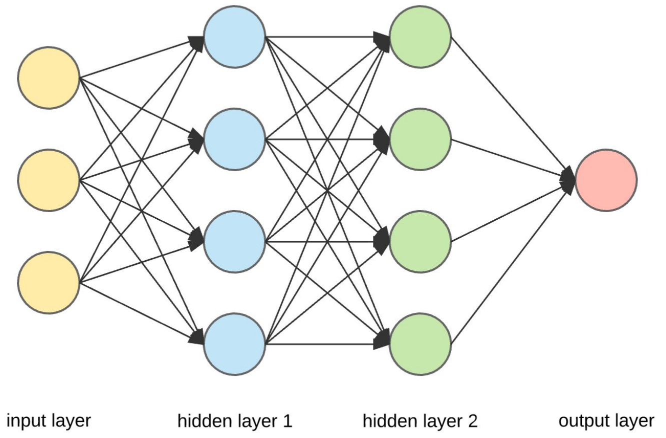
Tree-ensembles



Linear Regression



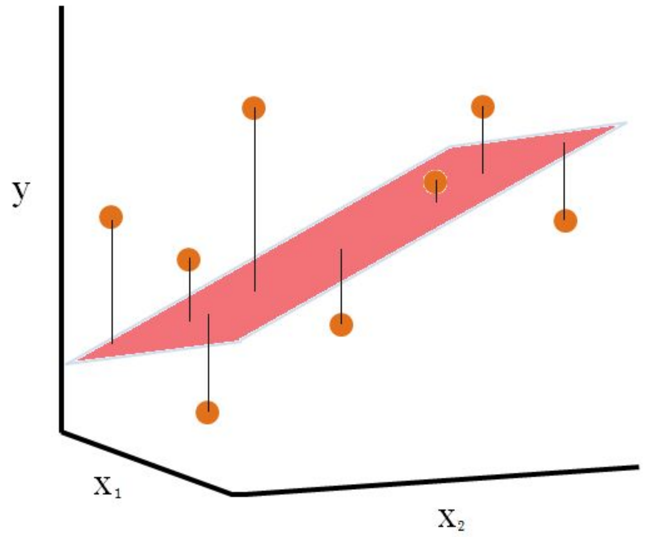
SVM

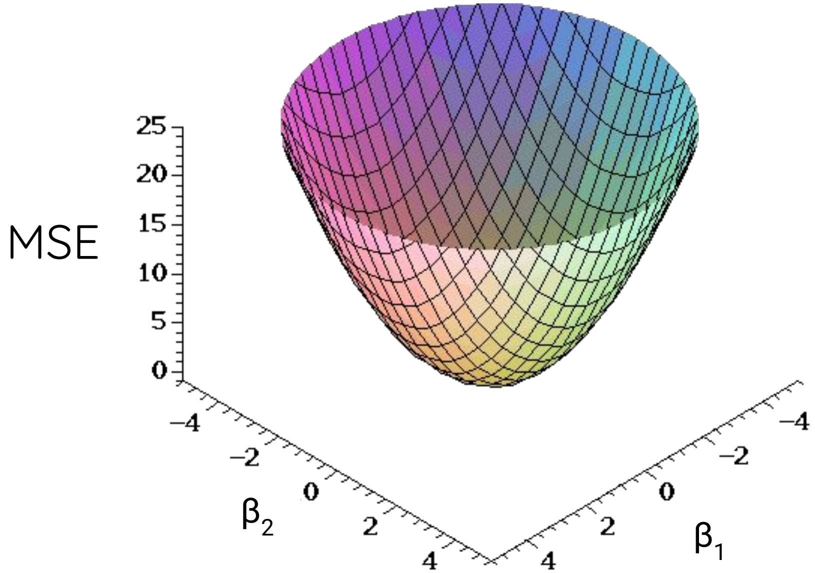


ANN

Model Fitting

- How do we learn the model weights?
  - Example: Linear regression
  - Model: $y = \beta_0 + \beta_1 x_1^2 + \beta_2 x_2^2$
  - Loss function: $MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
  - Optimization: Gradient descent



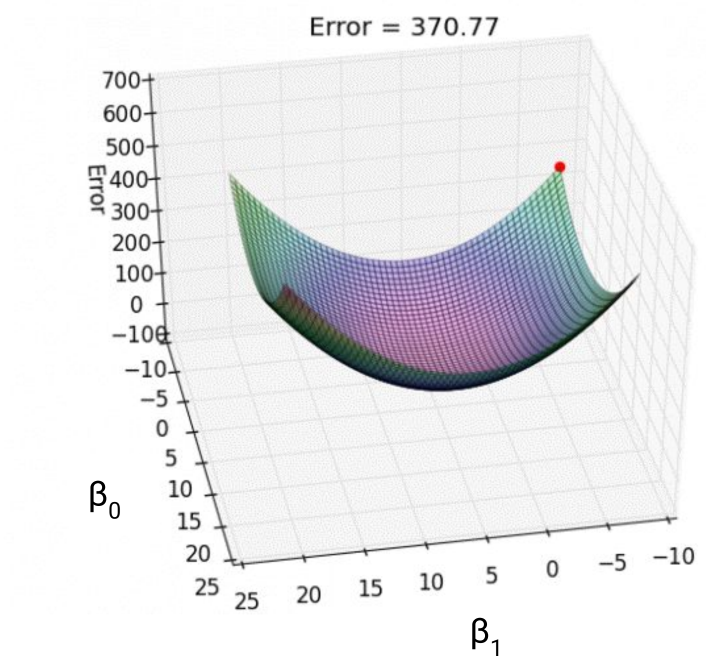


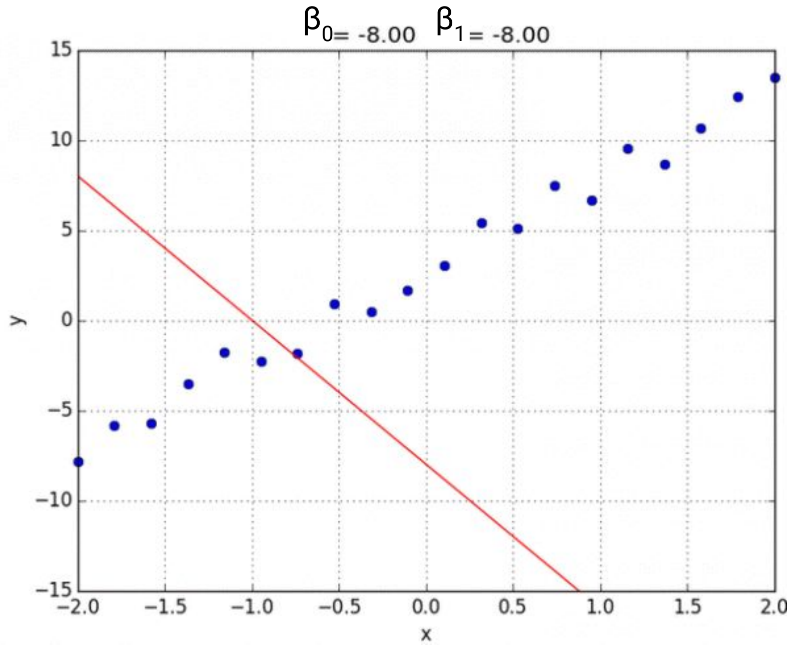
Model Fitting

- Gradient descent with a **single** input variable and **n** samples
  - Start with random weights (β_0 and β_1)
  - Compute loss (i.e. MSE)
  - Update weights based on the gradient

$$\hat{y}_i = \beta_0 + \beta_1 x_i$$

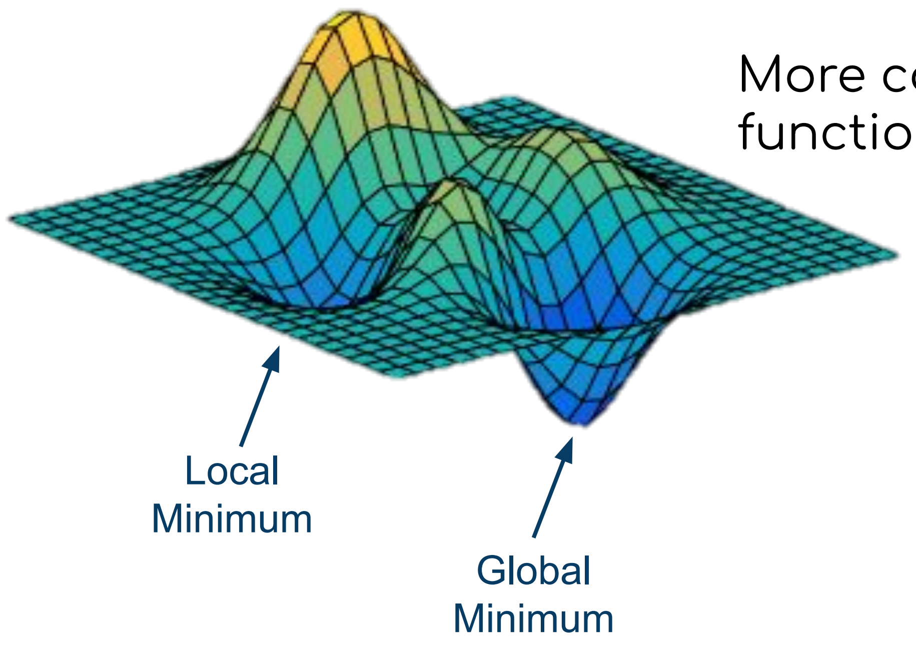
$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$





Model Fitting

- Gradient descent for complex models with non-convex loss functions
  - Start with random weights (β_0 and β_1)
  - Compute loss
  - Update weights based on the gradient



More complex models / loss functions (e.g. ANNs)

Model Regularization

- Why do we need to do it?
  - We have strong prior beliefs about what is a **plausible** model
    - e.g. I believe this symptom can be predicted with handful of genes.
  - Practical reasons
    - Prevent overfitting (n_features >> n_samples)

Model Regularization

- Why do we need to do it?
  - We have strong prior beliefs about what is a **plausible** model
    - e.g. I believe this symptom can be predicted with handful of genes.
  - Practical reasons
    - Prevent overfitting (n_features >> n_samples)

- How do we do it?
  - Modify the loss function
  - Constrain the learning process

1) L1/Lasso: constrains parameters to be ***sparse***

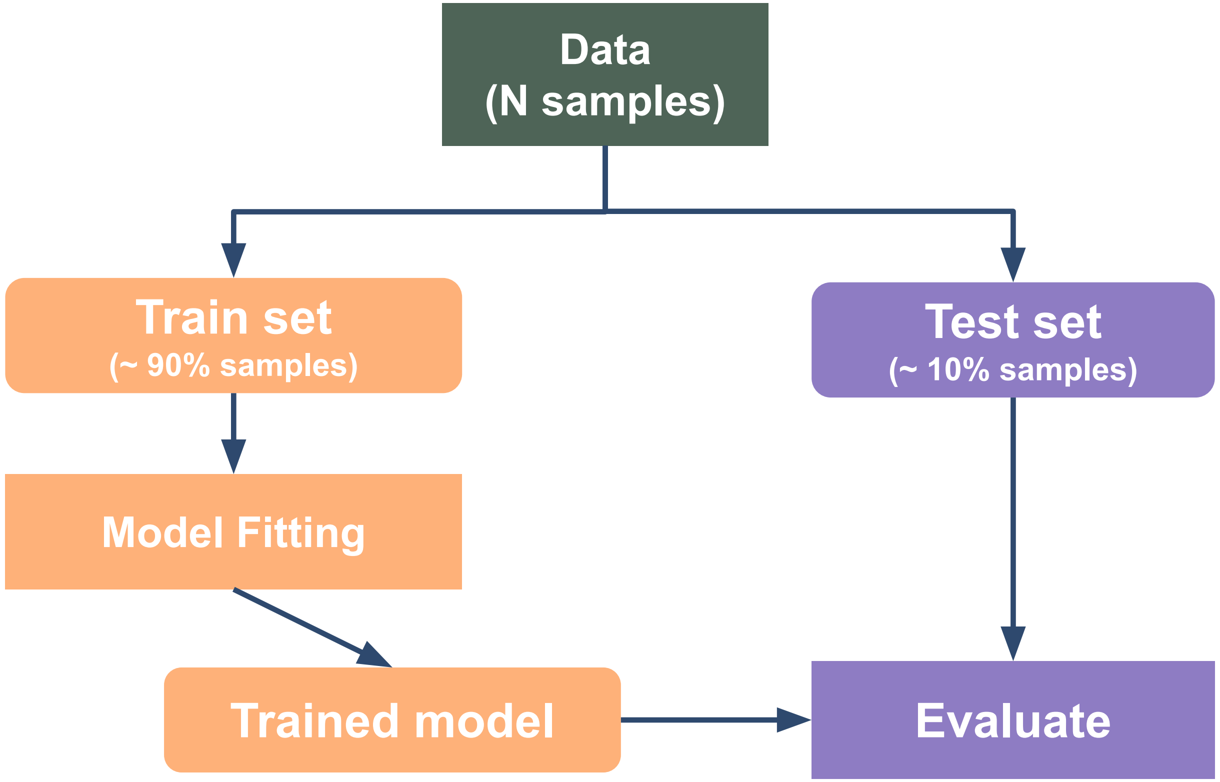
$$MSE = \sum_{i=1}^{n} (y_i - \underbrace{[\beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j]}_{\hat{y}_i})^2 + \underbrace{\lambda \sum_{j=1}^{p} |\beta_j|}_{L_1}$$

2) L2/Ridge: constrains parameters to be ***small***

$$MSE = \sum_{i=1}^{n} (y_i - \underbrace{[\beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j]}_{\hat{y}_i})^2 + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{L_2}$$

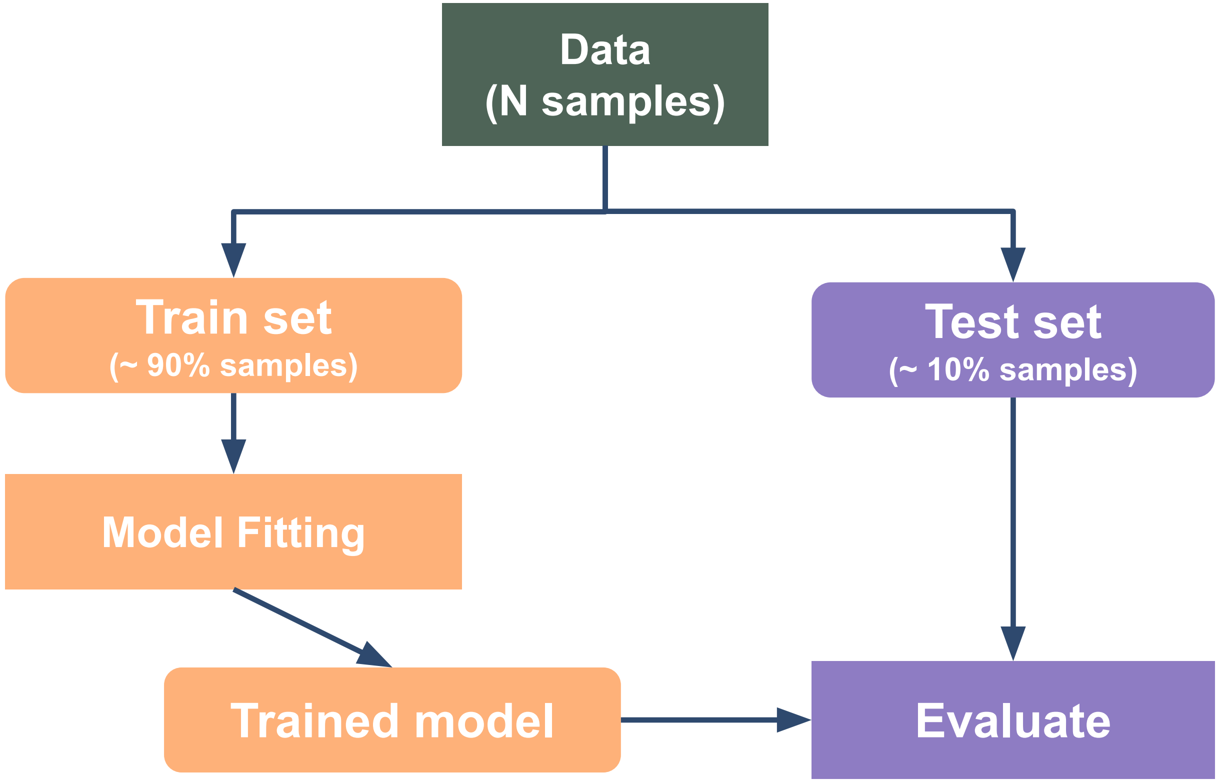
Model Evaluation

- What is model generalizability?
- How do we sample train and test sets?
- How do we select a model?



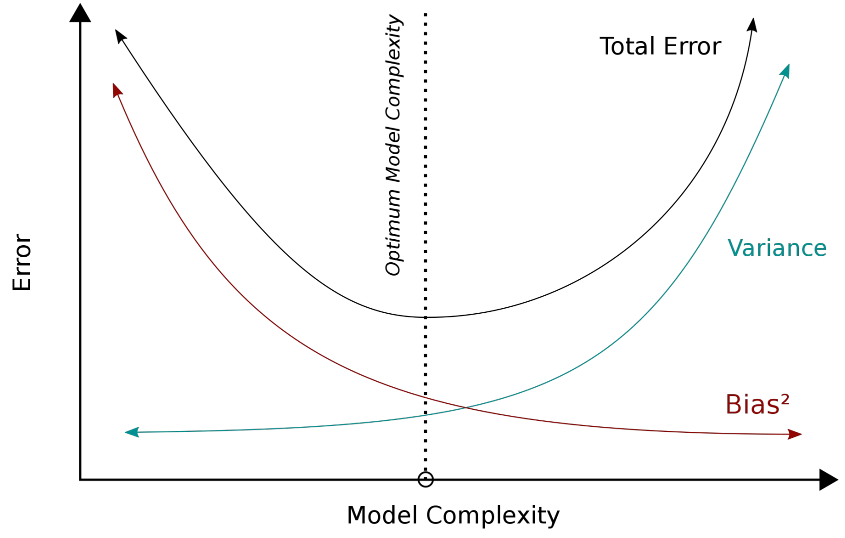
Model Evaluation

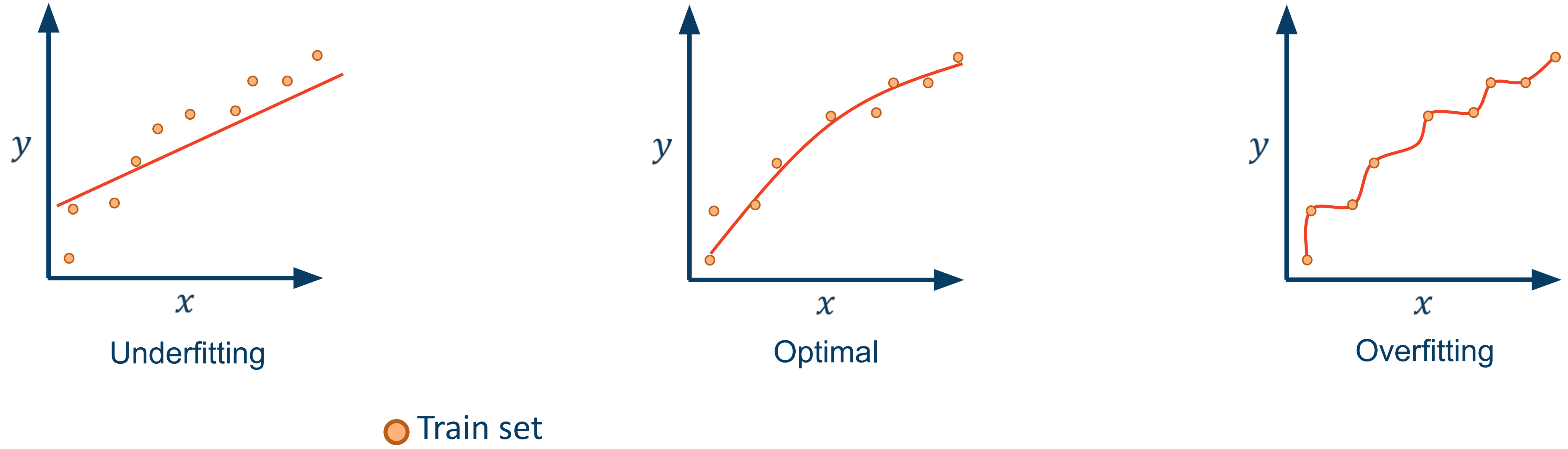
- **What is model generalizability?**
- How do we sample train and test sets?
- How do we select a model?



Model Evaluation

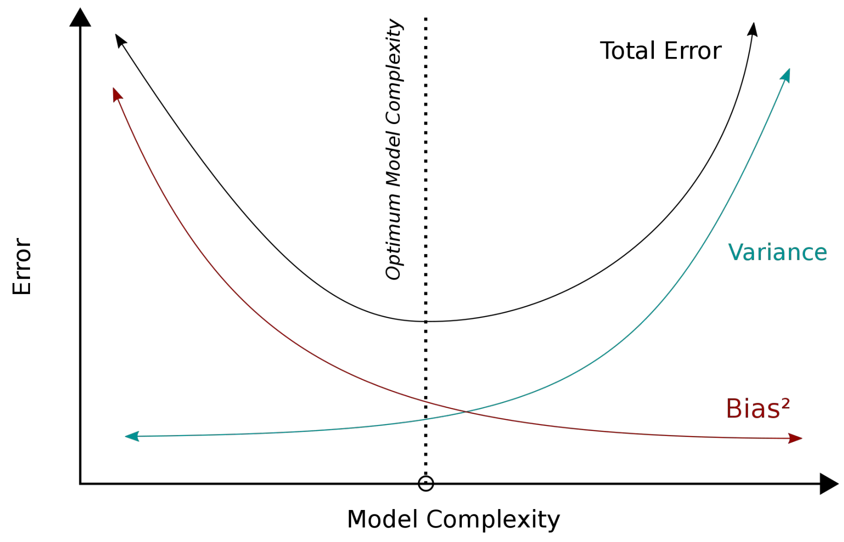
- Train performance ≠ Test performance
  - Model: Underfitting vs Overfitting
  - Errors: Bias - Variance tradeoff
  - Regression example

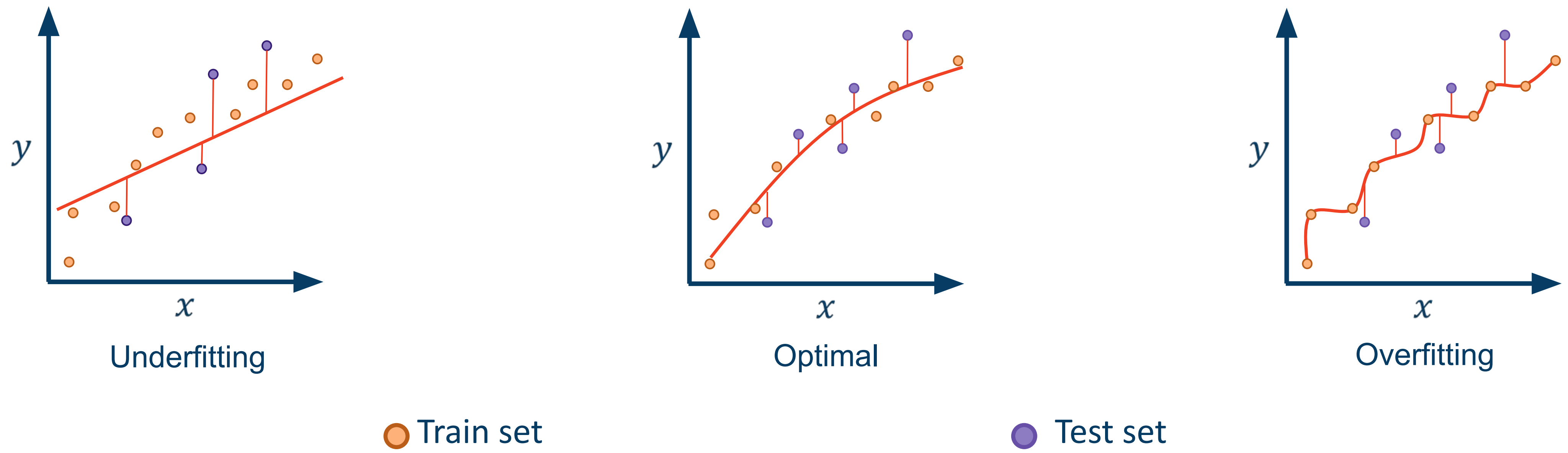




Model Evaluation

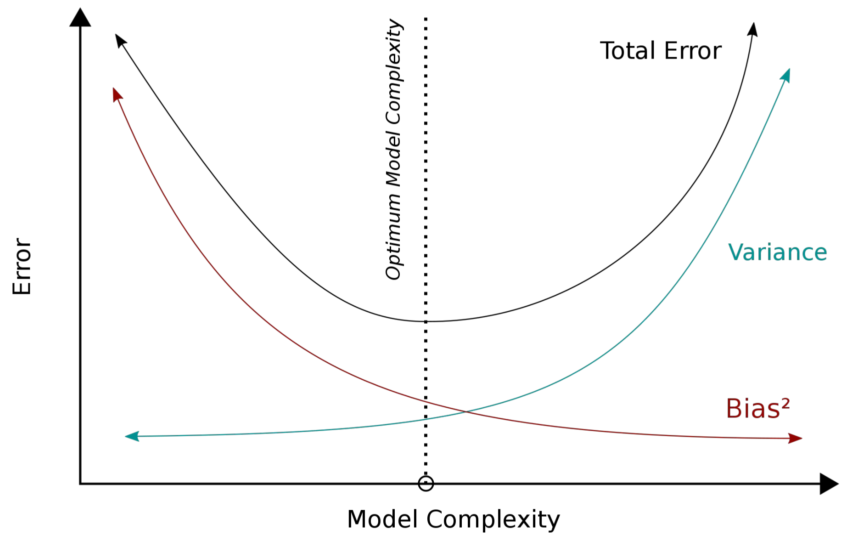
- Train performance ≠ Test performance
  - Model: Underfitting vs Overfitting
  - Errors: Bias - Variance tradeoff
  - Regression example

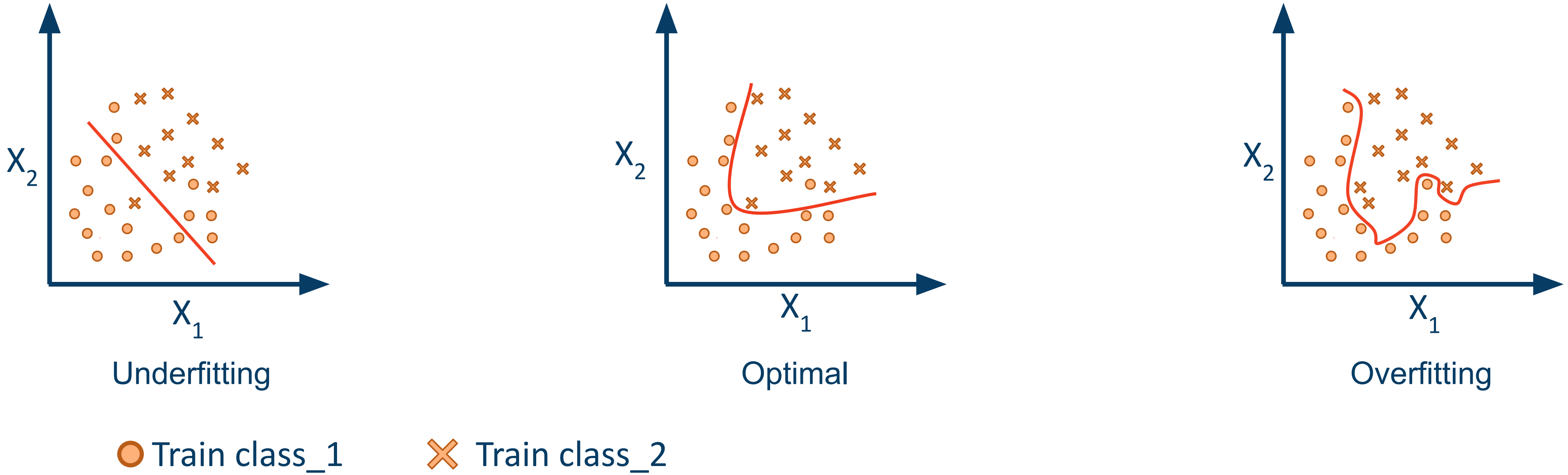




Model Evaluation

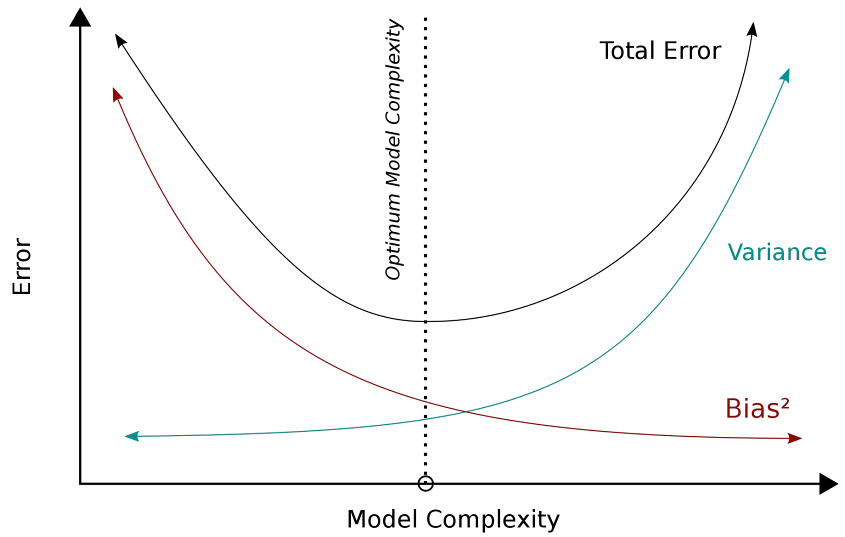
- Train performance ≠ Test performance
  - Model: Underfitting vs Overfitting
  - Errors: Bias - Variance tradeoff
  - Classification example

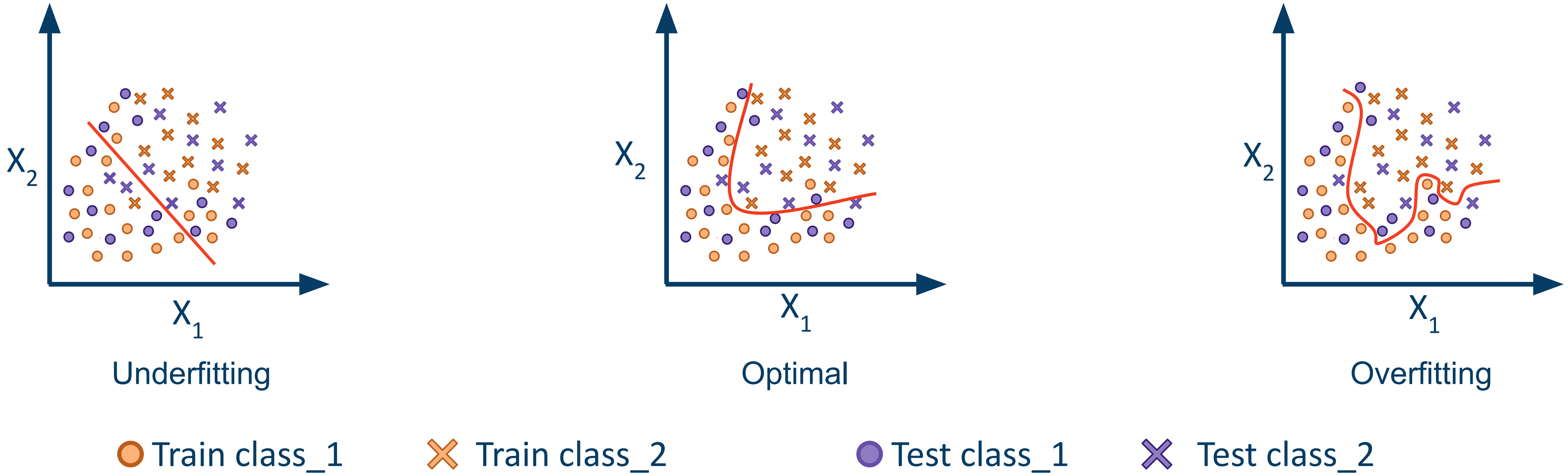




Model Evaluation

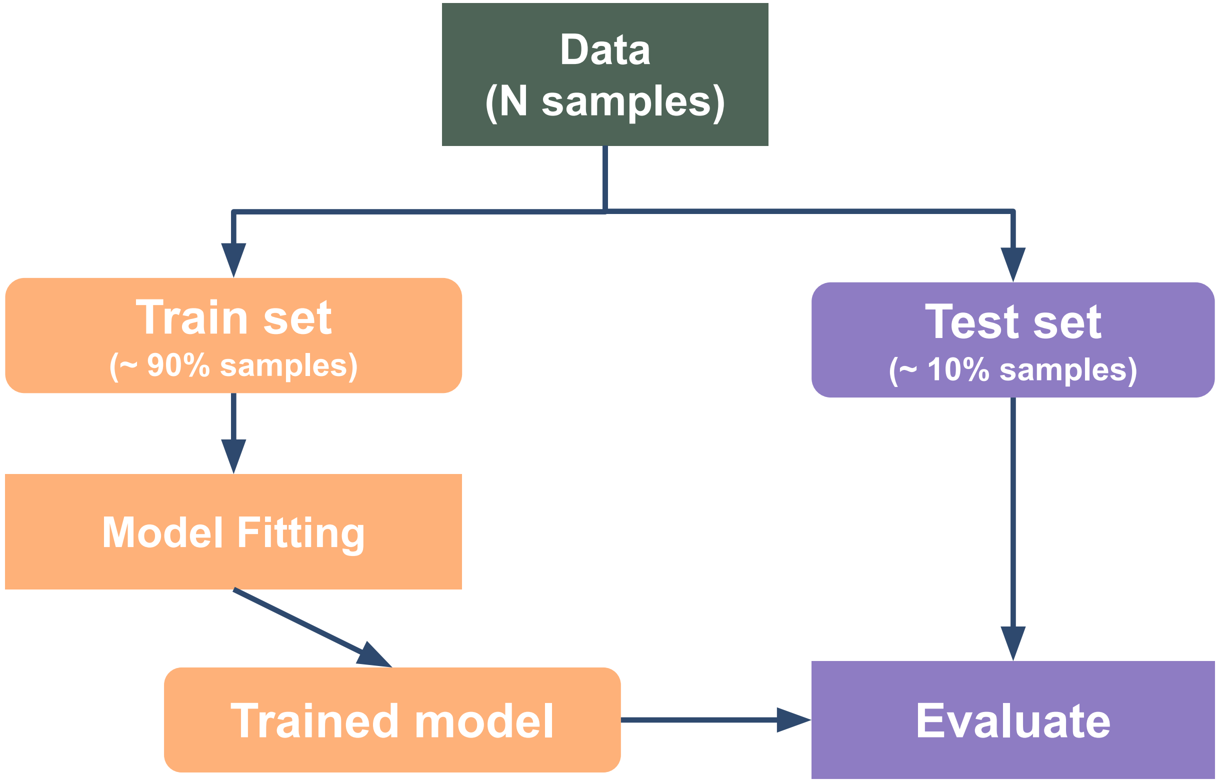
- Train performance ≠ Test performance
  - Model: Underfitting vs Overfitting
  - Errors: Bias - Variance tradeoff
  - Classification example





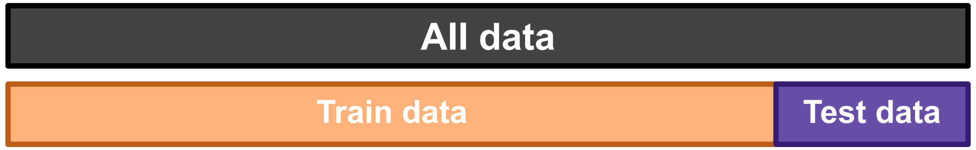
Model Evaluation

- What is model generalizability?
- **How do we sample train and test sets?**
- How do we select a model?



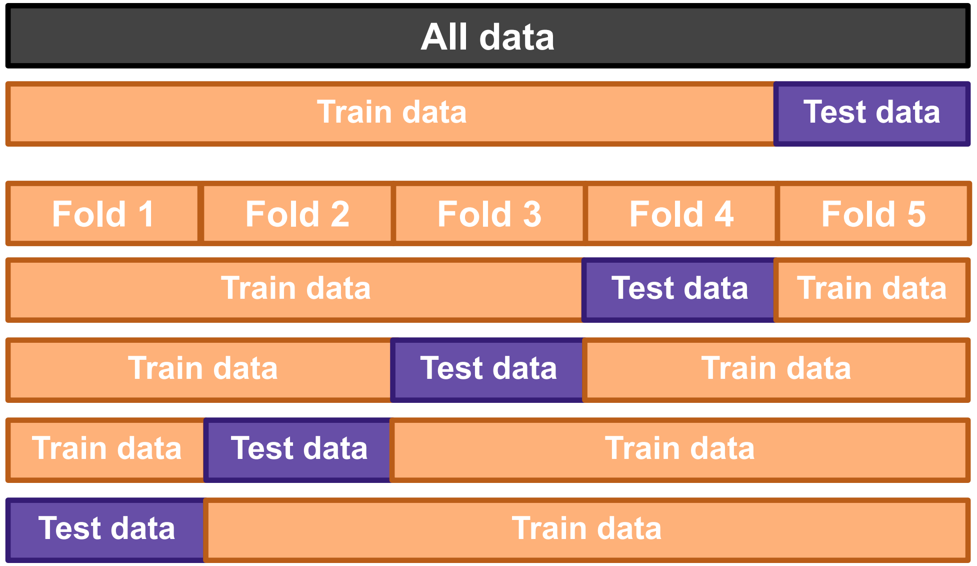
Model Cross-Validation

- How do we sample train and test sets?
  - Train set: learn model parameters
  - Test set (a.k.a held-out sample): Evaluate model performance



Model Cross-Validation

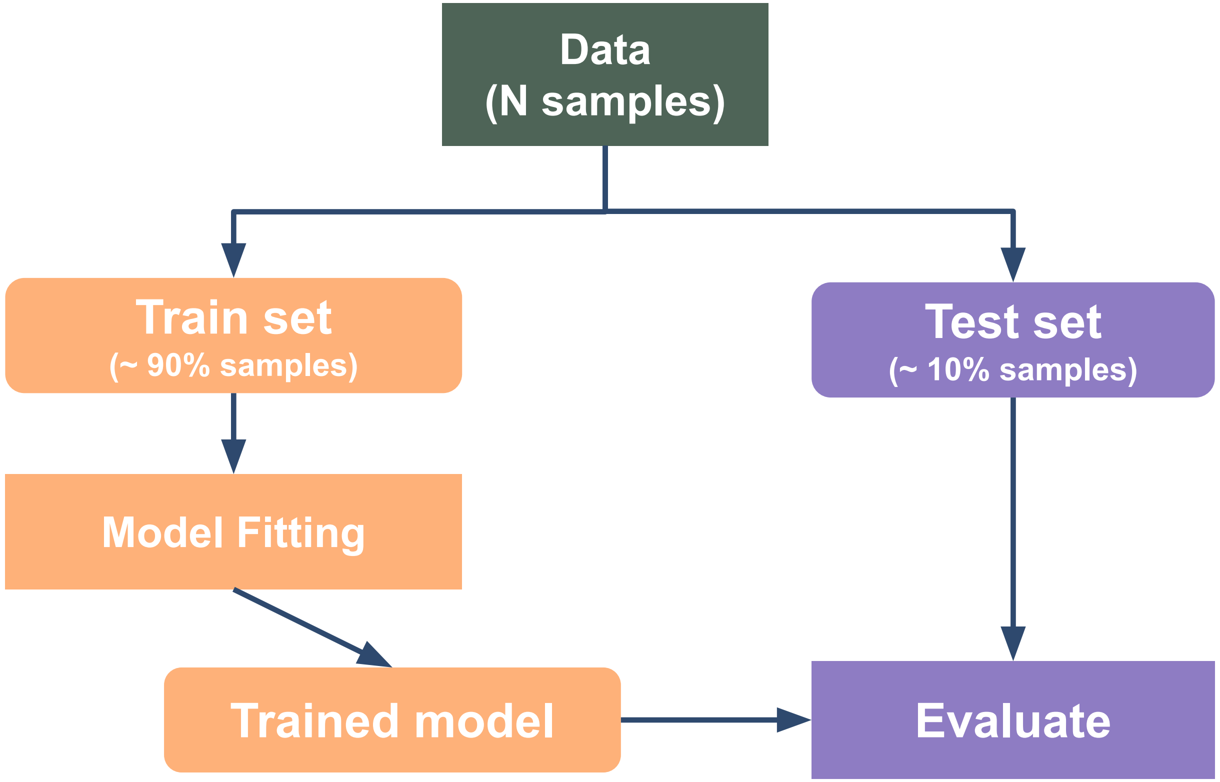
- How do we sample train and test sets?
  - Train set: learn model parameters
  - Test set (a.k.a held-out sample): Evaluate model performance
  - Repeat for different Train-Test splits
    - k-fold, shuffle-split
  - Report performance statistics over all test folds



CV outer loop

Model Evaluation

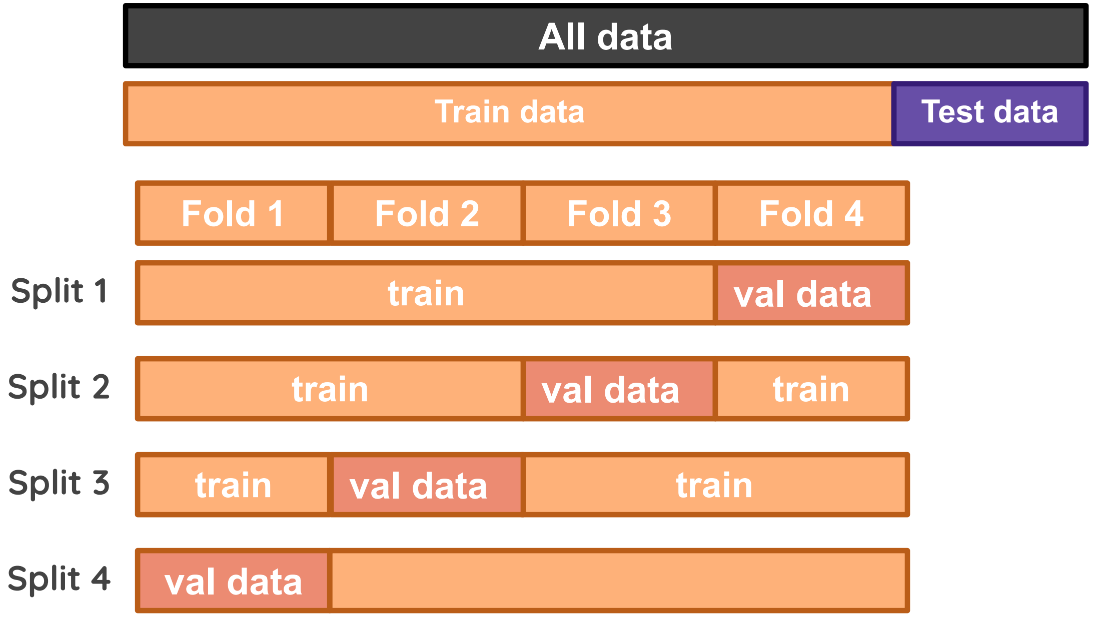
- What is model generalizability?
- How do we sample train and test sets?
- **How do we select a model?**



Model Cross-Validation

- How do we select a model?
  - Tune hyper-parameters of a model
  - Compare several different model architectures
  - Select / transform raw features

- This repeats for all train-test splits in the outer loop



CV inner loop

Hyper-parameters

- Hyper-parameter ≠ parameter (or weights)
  - Parameters are **learned**; hyper-parameters are **chosen**!

Hyper-parameters

- Hyper-parameter ≠ parameter (or weights)
  - Parameters are **learned**; hyper-parameters are **chosen**!

- Examples:
  - Degree of model (eg. linear vs quadratic)
  - Kernels
  - Number of trees
  - Number of layers, filters, batch-size, learning-rate in ANNs

Hyper-parameters

- Hyper-parameter ≠ parameter (or weights)
  - Parameters are **learned**; hyper-parameters are **chosen**!

- Examples:
  - Degree of model (eg. linear vs quadratic)
  - Kernels
  - Number of trees
  - Number of layers, filters, batch-size, learning-rate in ANNs

- How do we choose them?
  - Prior beliefs → eg. cortical thickness and age have quadratic relationship.
  - Arbitrarily → we gotta start with something!
  - Trial and error → do a computationally feasible grid-search.

Hands-on Exercise 1

https://github.com/neurodatascience/main-2021-ml-parts-1-2

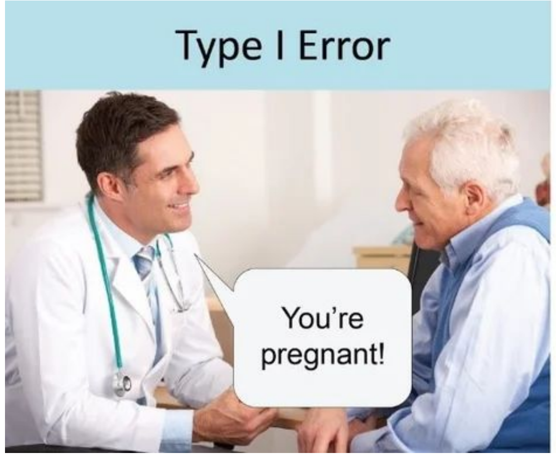
Performance Scores

- Loss functions → computationally well-suited metrics
  - May / need not completely capture performance metrics of interest

- Scores → practically useful metrics
  - Binary classification

Confusion Matrix		Ground Truth	
		POSITIVE	NEGATIVE
Prediction	POSITIVE	TP	FP
	NEGATIVE	FN	TN

False Positive



False Negative

Performance Scores

- ML model that detects Covid from chest CTs. Current Covid prevalence ~ 1 in 1000.
  - FP: model predicts *Covid* when person is *healthy*
  - FN: model predicts *healthy* when person has *Covid*
- What happens if we build model that predicts everyone as healthy?
  - i.e. zero FPs!

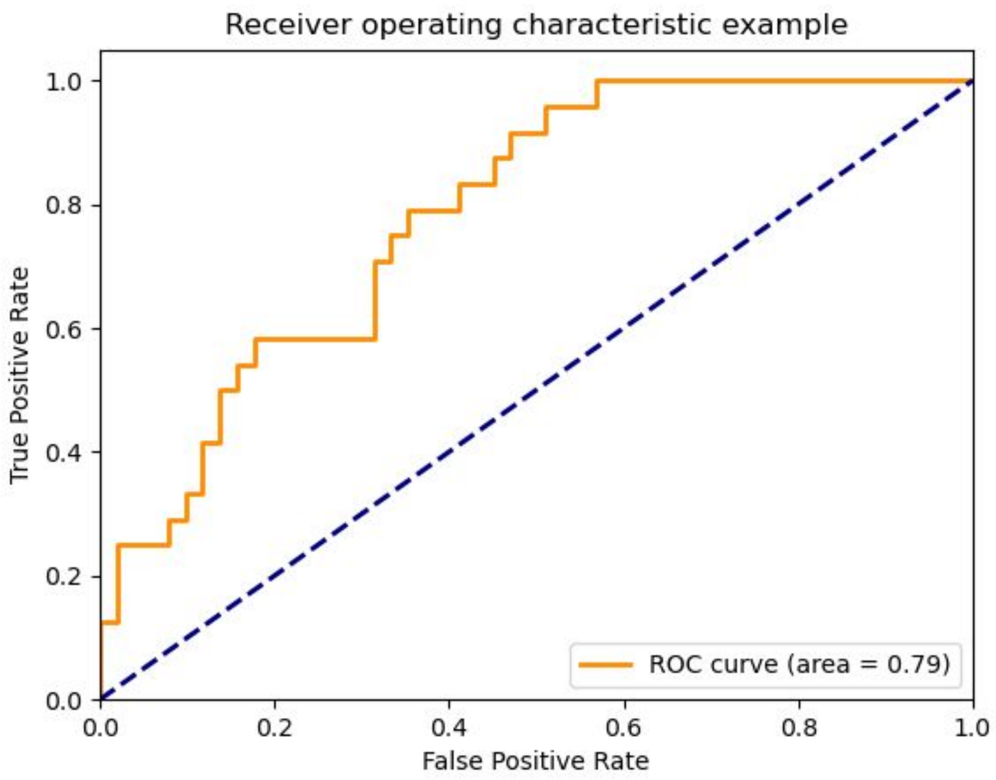
Performance Scores

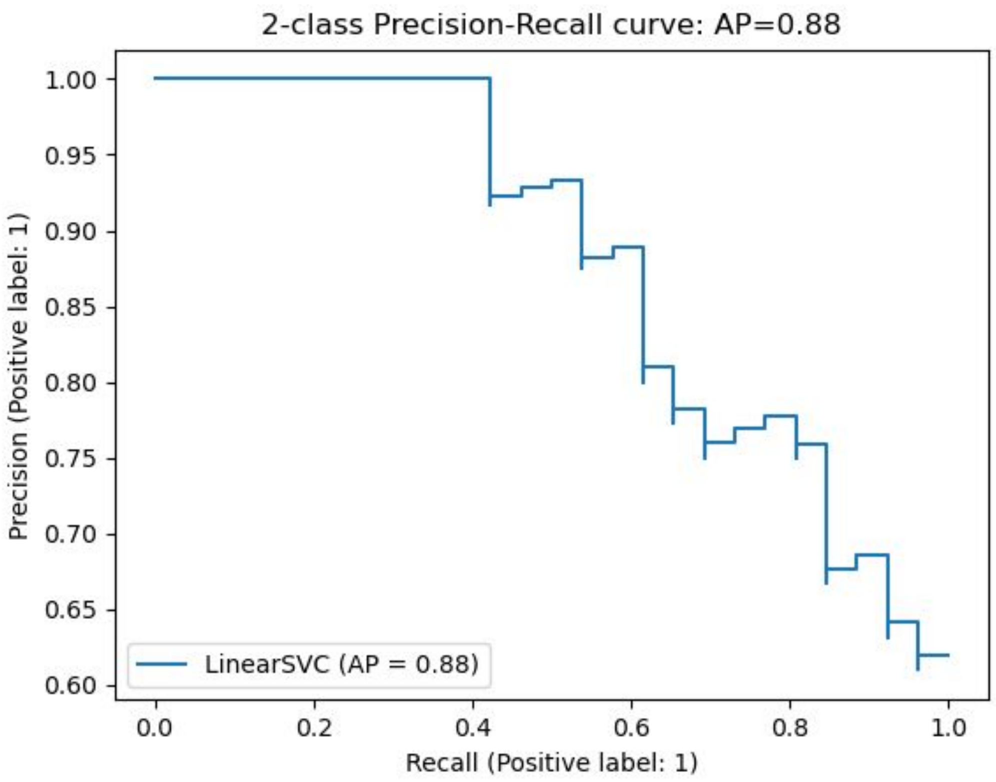
- ML model that detects Covid from chest CTs. Current Covid prevalence ~ 1 in 1000.
  - FP: model predicts *Covid* when person is *healthy*
  - FN: model predicts *healthy* when person has *Covid*
- What happens if we build model that predicts everyone as healthy?

Score	Formula	Null	What does it tell us?	When do I use it?
Accuracy	(TP+TN) / (TP+FP+FN+TN)	0.999	How many people did we correctly predict out of all the people scanned?	FNs & FPs have similar costs
Precision	TP/(TP+FP)	NaN	How many of those who we predicted as "covid" do actually have "covid"?	If you want to be more confident of your TPs
Recall (aka Sensitivity)	TP/(TP+FN)	0	Of all the people who have covid, how many of those did we correctly predict?	If you prefer FPs over FNs.
Specificity	TN/(TN+FP)	0.999	Of all the people who are healthy, how many of those did we correctly predict?	If you prefer FNs over FPs.
F1	2*(Recall * Precision) / (Recall + Precision)	NaN	Harmonic mean(average) of the precision and recall.	When you have an uneven class distribution

Performance Curves

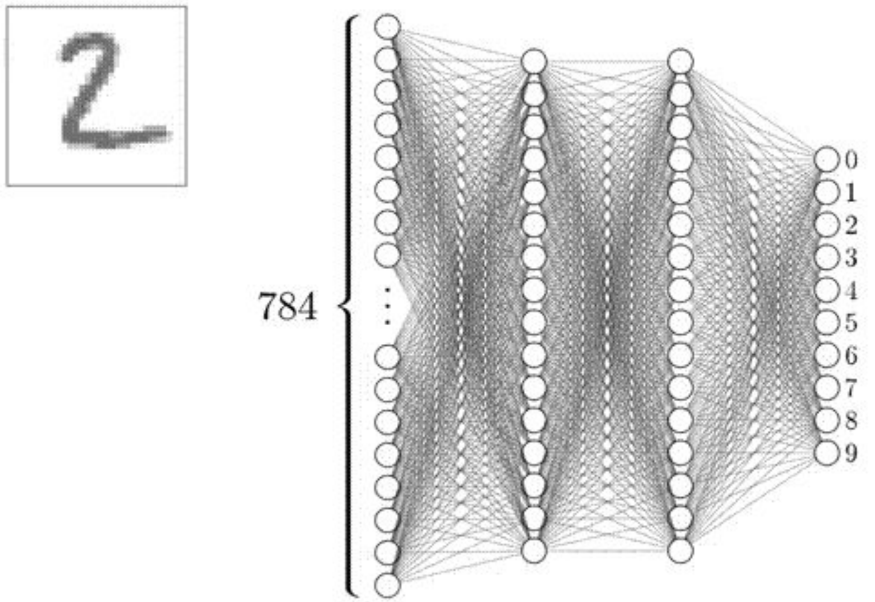
- Receiver Operating Characteristic (ROC) → Want high area-under-the-curve (AUC)
- Precision-Recall → Want high AUC or high Average precision (AP)





Deep-learning

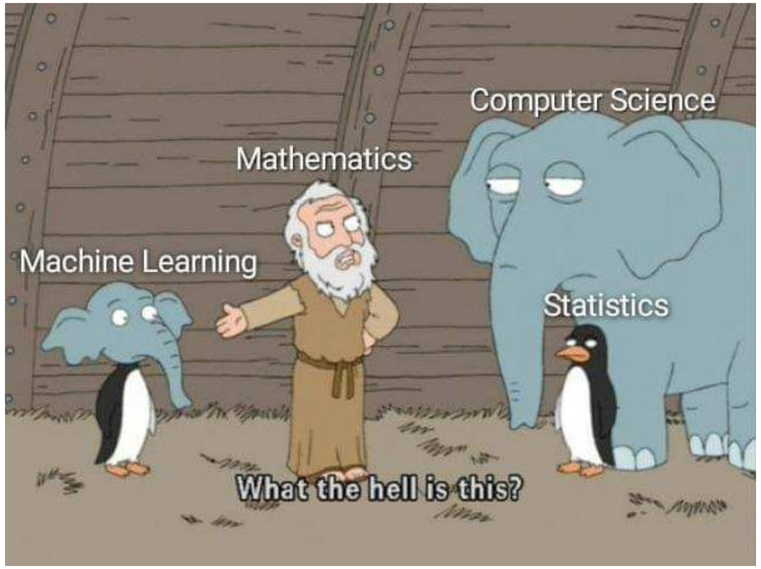
- Why the buzz?
  - Works amazing on structured input
  - Highly flexible → universal function approximator
- What are the challenges?
  - Large number of parameters → data hungry
  - Large number of hyper-parameters → difficult to train
- When do I use it?
  - If you have highly-structured input, eg. medical images.
  - You have a lot of data and computational resources.



ANN for handwritten-digit images
(gif source: 3b1b)

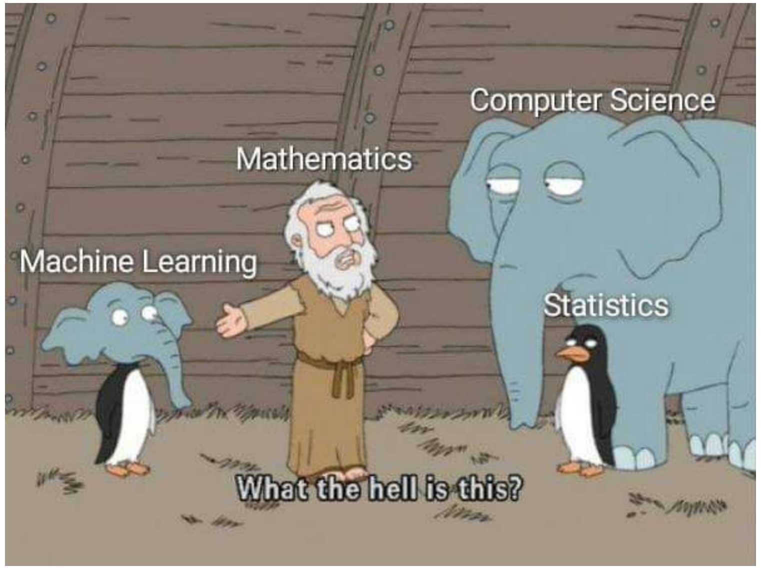
Pitfalls and Challenges

- Models do not generalize even after good CV performance
  - Implicit double-dipping
  - Dataset biases (eg. North-American demographics)
  - Noisy labels (eg. diagnosis definitions)
  - Data distribution shifts (eg. assay, scanner upgrades)



Pitfalls and Challenges

- Models do not generalize even after good CV performance
  - Implicit double-dipping
  - Dataset biases (eg. North-American demographics)
  - Noisy labels (eg. diagnosis definitions)
  - Data distribution shifts (eg. assay, scanner upgrades)



- Unnecessary complexity
  - Do I really need a giant deep-net or a simple linear model would do?

ML Novice Checklist

- Data
  - What is my n_features and n_samples?
  - Am I encoding categorical data correctly?
  - Am I using information (e.g. mean) from test set to preprocess (eg. zscore) the data?

ML Novice Checklist

- Data
  - What is my n_features and n_samples?
  - Am I encoding categorical data correctly?
  - Am I using information (e.g. mean) from test set to preprocess (eg. zscore) the data?

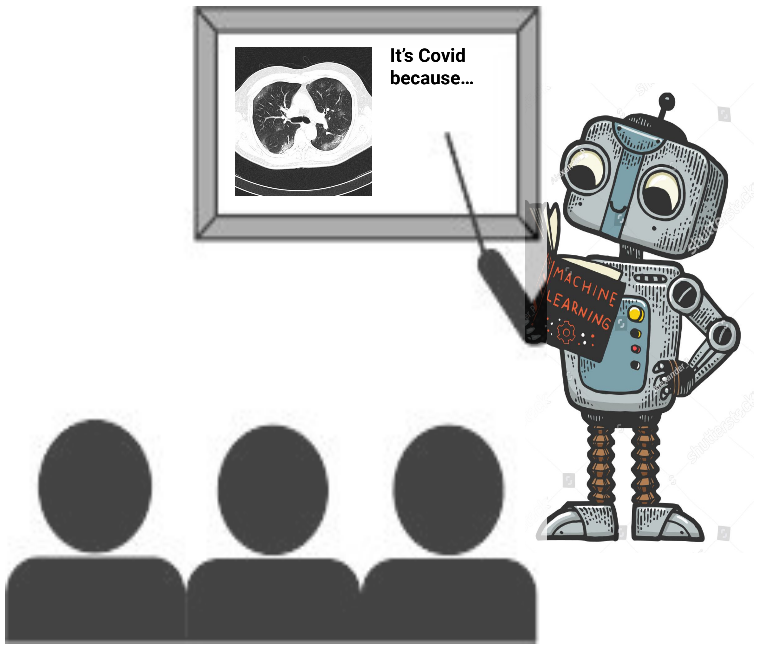
- Model
  - Do my performance metrics capture the practical use-case of interest?
  - What is the null / dummy model performance?
    - Classification: Predict majority class all the time
    - Regression: Predict the median value all the time
  - Am I interpreting model coefficients correctly?

Takeaways

- Supervised models are useful for **predictions**
  - eg. image segmentation, prognosis, drug development

- Our job is to ensure **generalizability** of these models
  - Multitude of validations
  - Understanding model biases and limitations

- Food for thought: ***engineering tools*** vs *scientific discovery*
  - Interpretability and explainability
  - Causality, reliability, fairness



Explainable AI

Hands-on Exercise 2

https://github.com/neurodatascience/main-2021-ml-parts-1-2