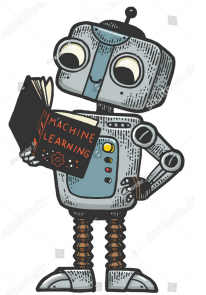


Introduction to Supervised Learning

MAIN educational 2021

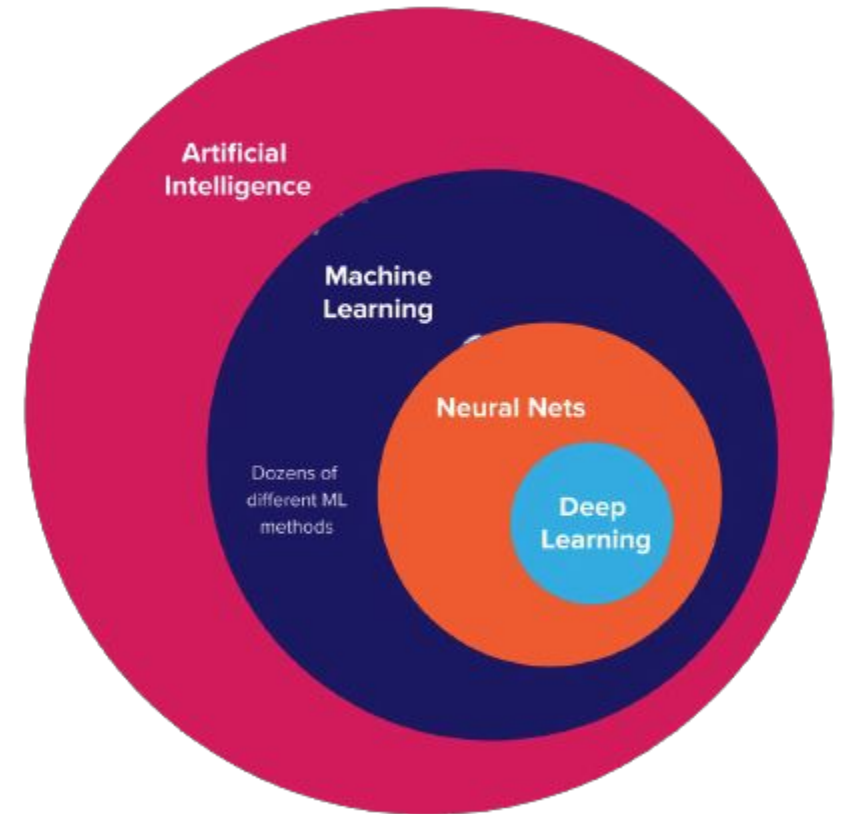
By
Nikhil Bhagwat & Jérôme Dockès

25 November 2021



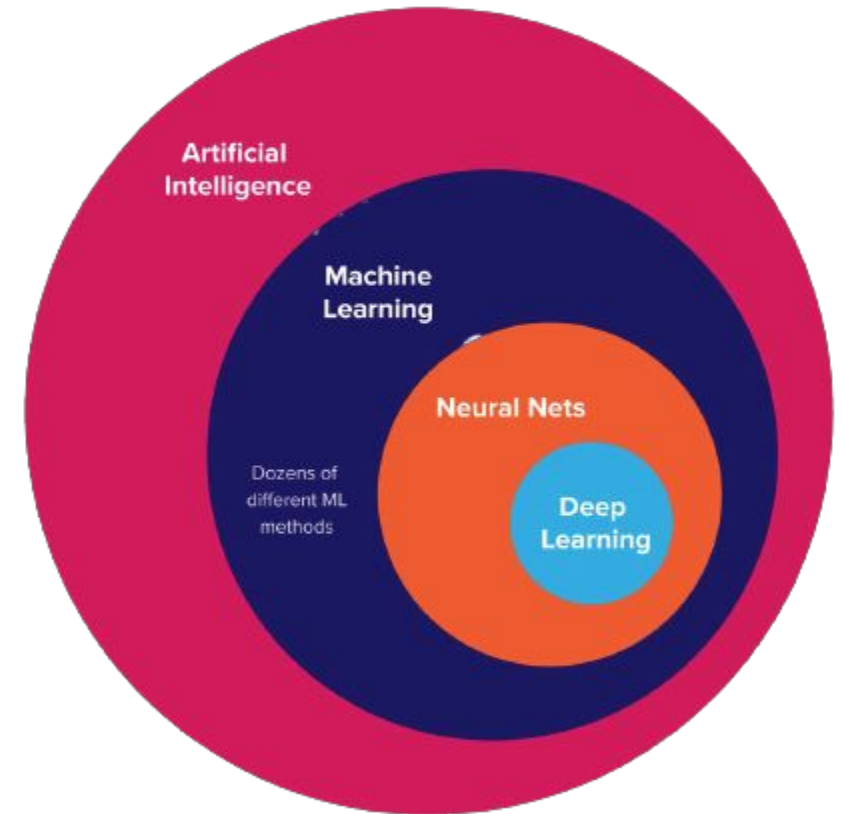
The Basic Questions

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



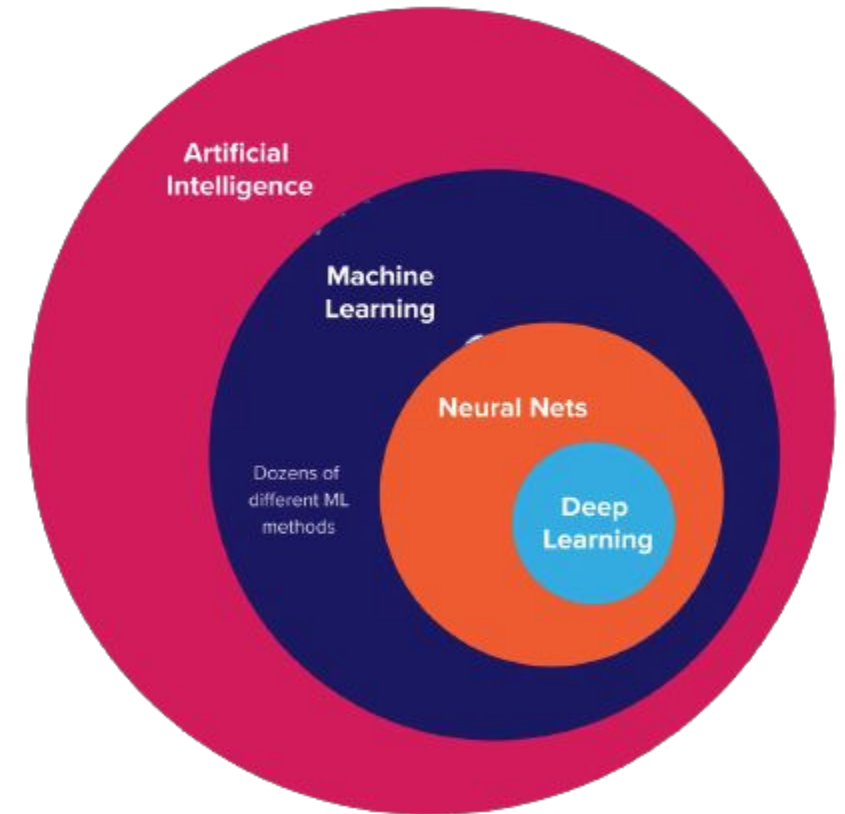
The Basic Questions

- 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 phenomena (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.

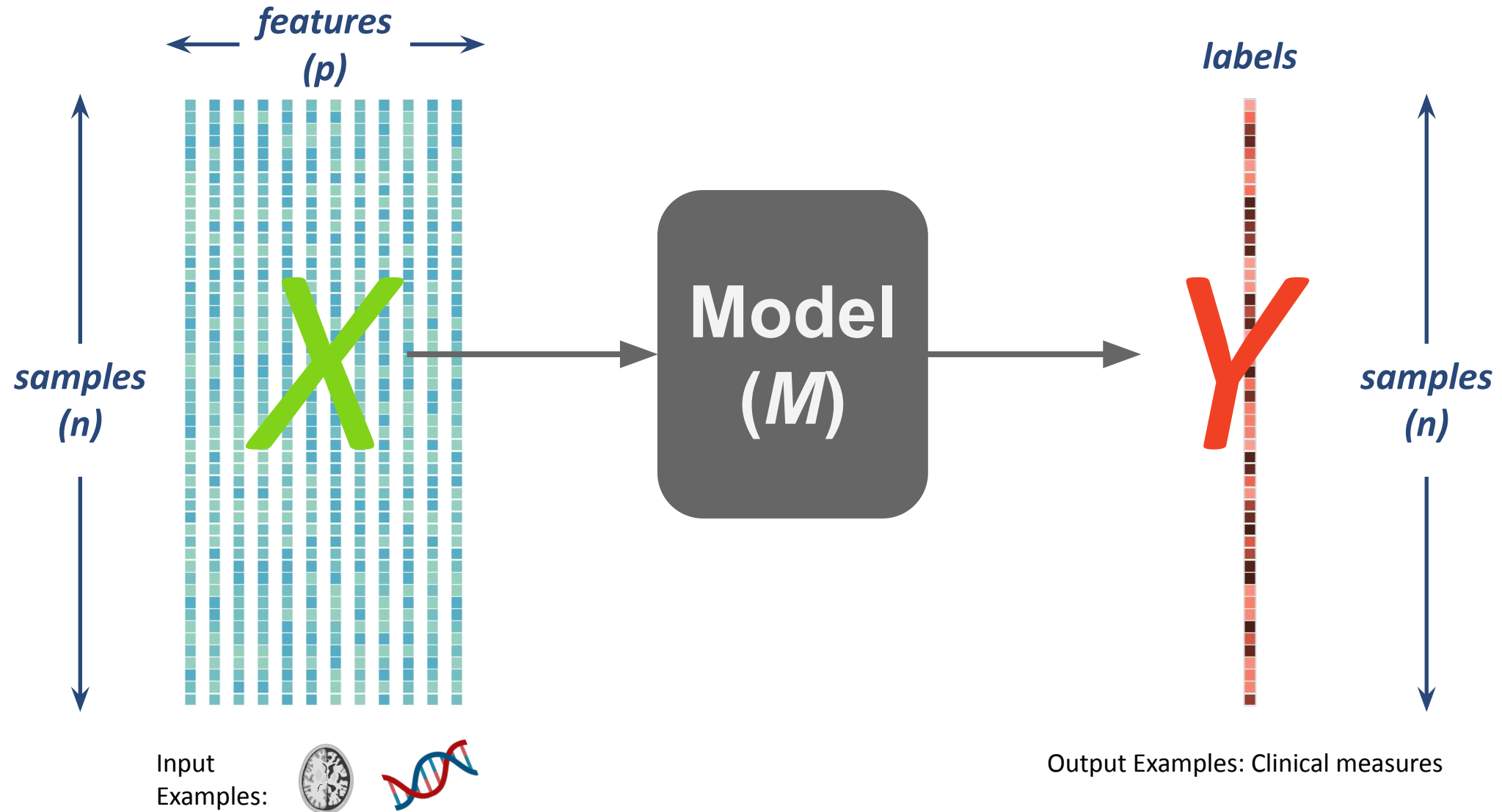


The Basic Questions

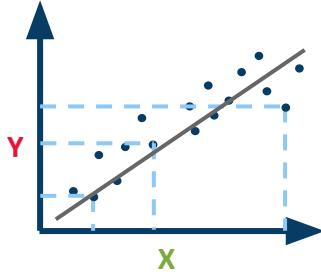
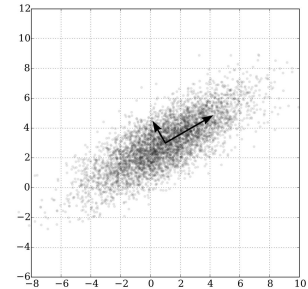
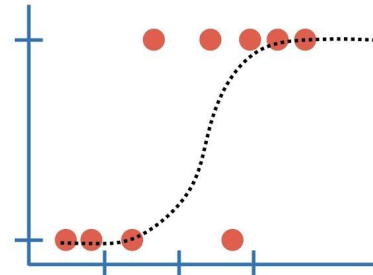
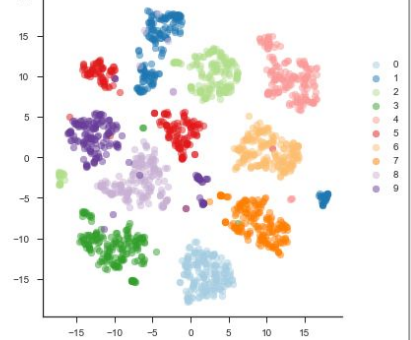
- What is Machine learning (ML)?
 - ML is the study of computer algorithms that improve automatically through experience and by the use of data.
- Why is it useful - especially in life sciences?
 - Biology, Medicine, Environmental sciences comprise phenomena (e.g. a disease) with large number of variables.
 - 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



Types of ML Algorithms

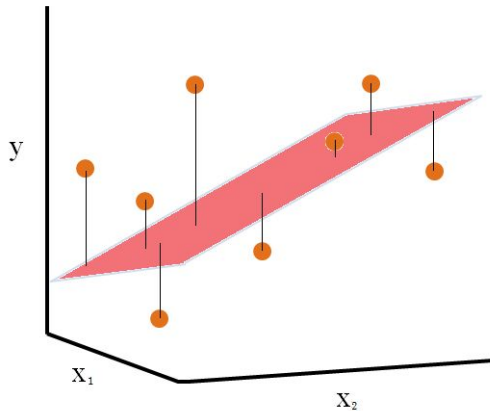
Outcome	Supervised Learning	Unsupervised Learning
Continuous	<p>Regression</p>  A scatter plot showing a positive linear correlation between variables X and Y. A solid black line represents the linear regression fit. The X-axis is labeled 'X' in green and the Y-axis is labeled 'Y' in red. A dashed blue box highlights a region of the data points.	<p>Dimensionality reduction</p>  A scatter plot showing a dense cloud of grey data points. Two black arrows originate from the center of the cloud, pointing in opposite directions, representing the principal components of the data distribution.
Categorical	<p>Classification</p>  A scatter plot showing two classes of data points (red circles) separated by a dashed black sigmoid curve. The X and Y axes are marked with tick marks but no labels.	<p>Clustering</p>  A scatter plot showing 10 distinct clusters of data points, each represented by a different color. A legend on the right side of the plot lists the cluster numbers from 0 to 9, corresponding to the colors: 0 (light blue), 1 (blue), 2 (light green), 3 (green), 4 (yellow-green), 5 (yellow), 6 (orange), 7 (dark orange), 8 (purple), and 9 (dark blue).

Supervised Learning

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

Supervised Learning $x \rightarrow M \rightarrow y$

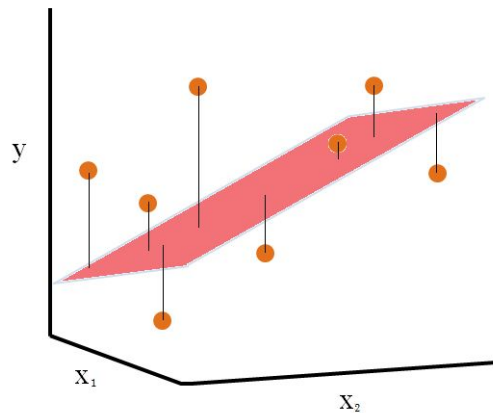
- Goal: *Learn* parameters (or weights) of a model (M) that maps x to y
- Example models:
 - Linear / Logistic regression



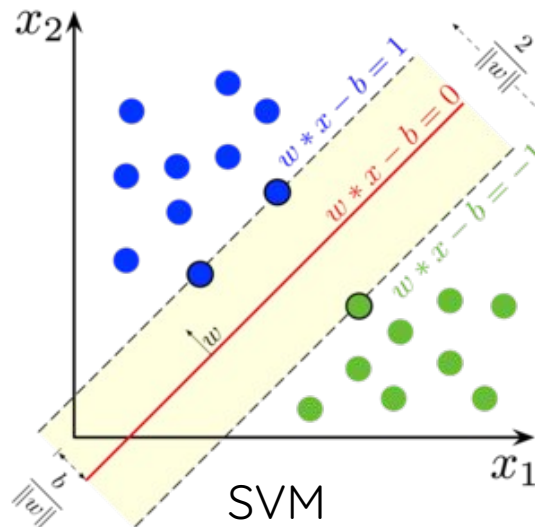
Linear Regression

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



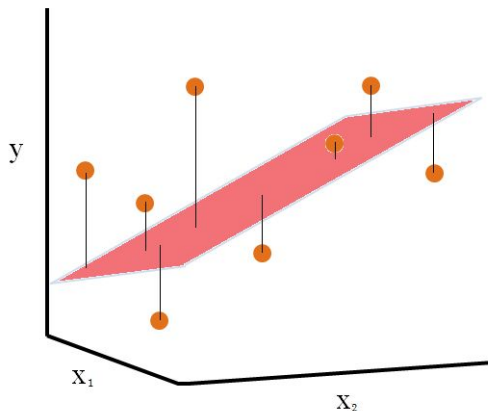
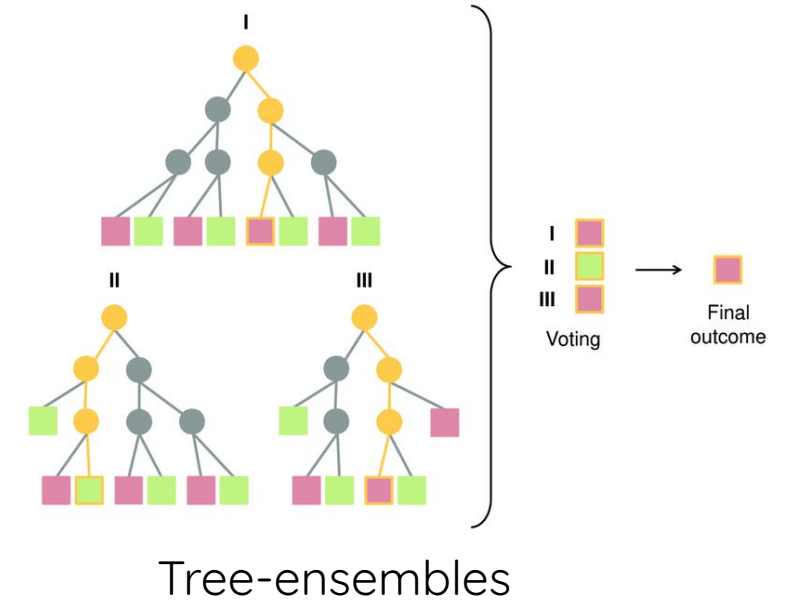
Linear Regression



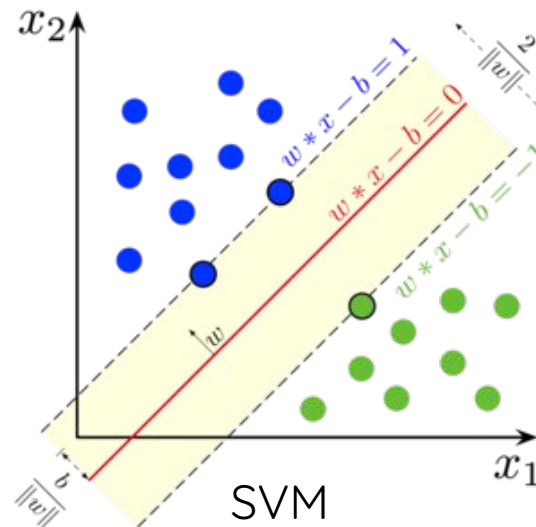
SVM

Supervised Learning $x \rightarrow M \rightarrow y$

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



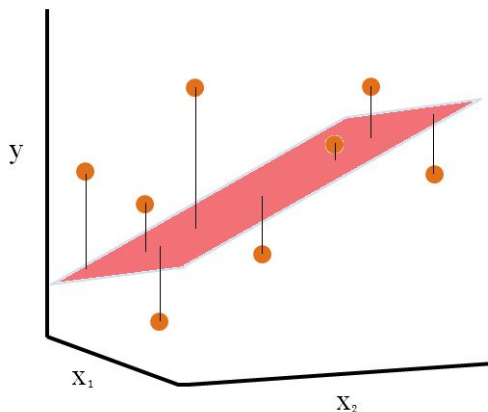
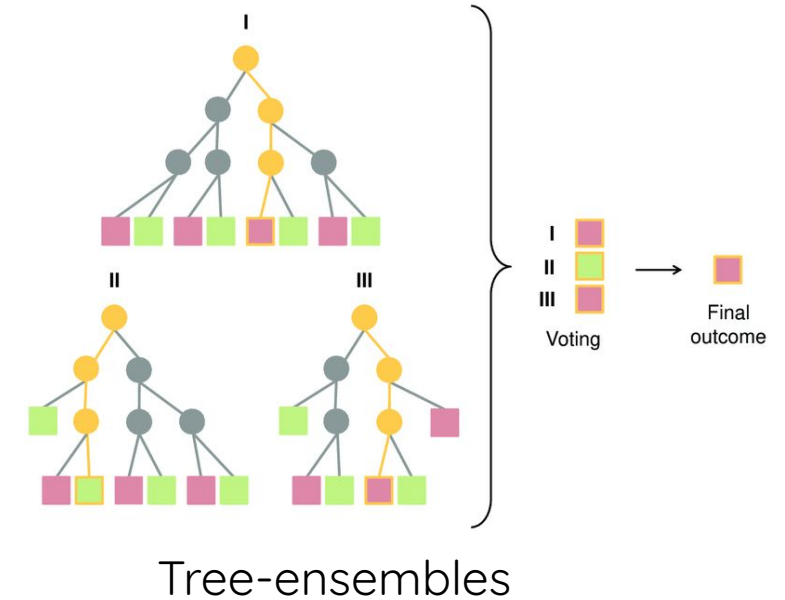
Linear Regression



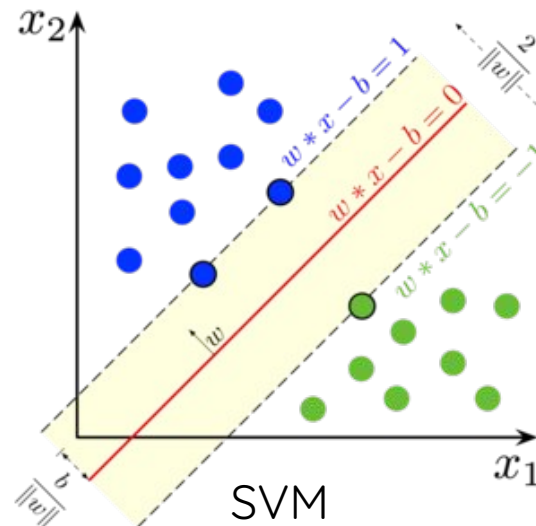
SVM

Supervised Learning $x \rightarrow M \rightarrow y$

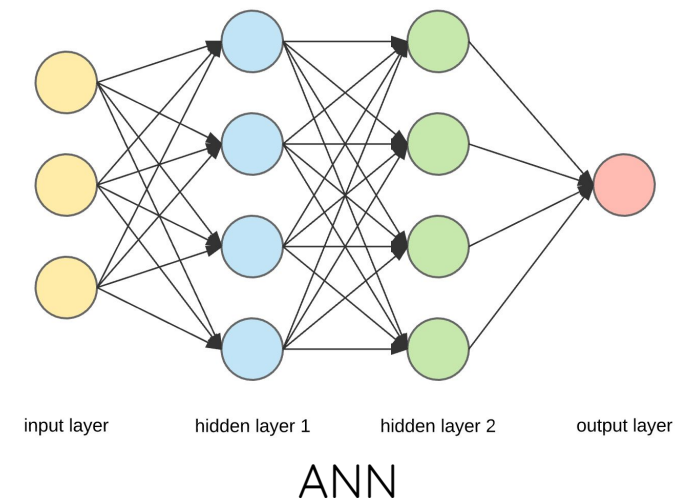
- Goal: *Learn* parameters (or weights) of a model (M) that maps x to y
- Example models:
 - Linear / Logistic regression
 - Support vector machines
 - Tree-ensembles: random forests, gradient boosting
 - Artificial Neural networks



Linear Regression

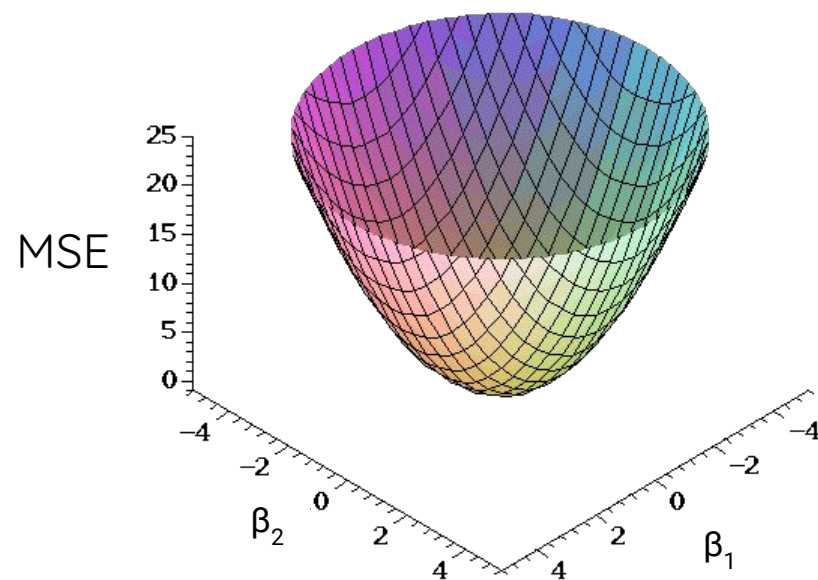
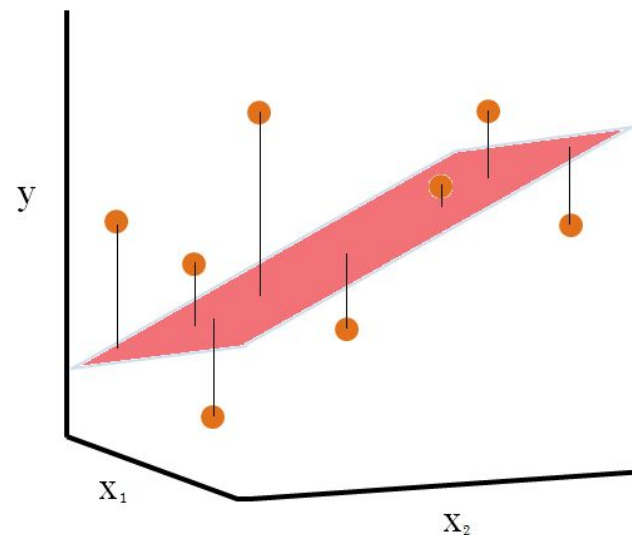


SVM



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

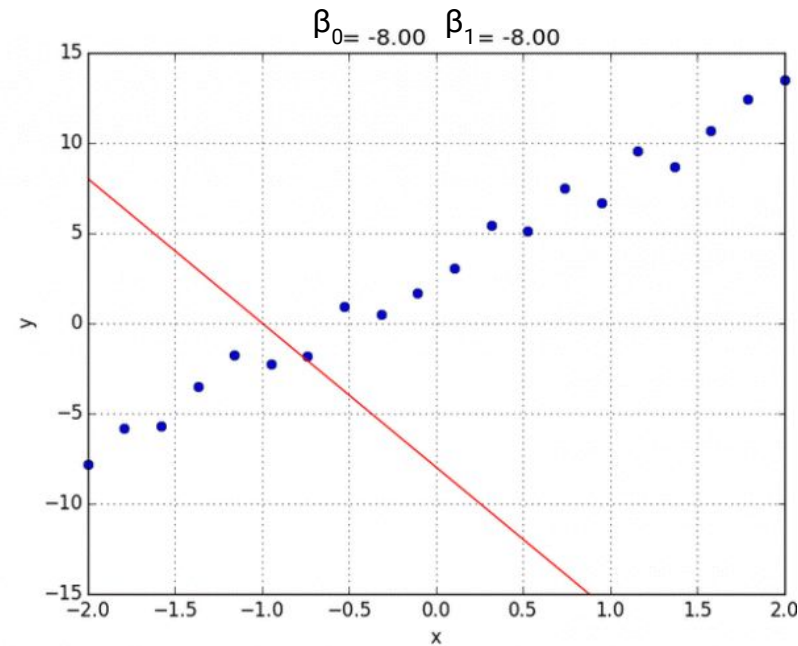
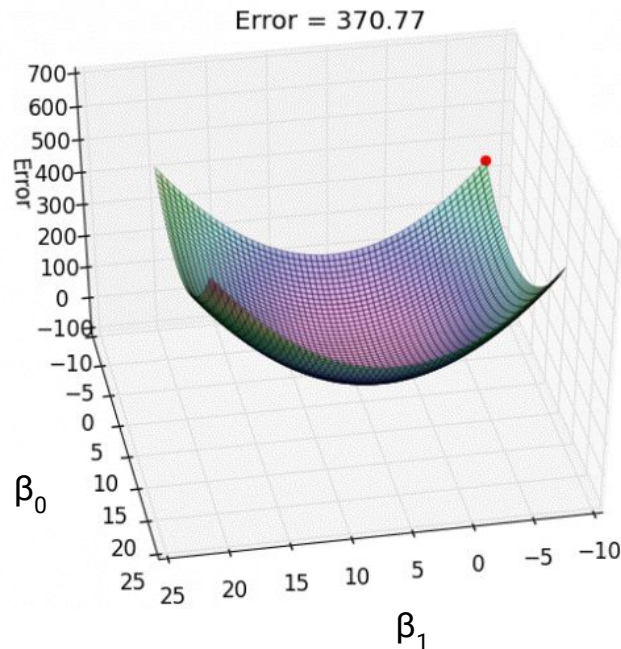


Model Fitting

- 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

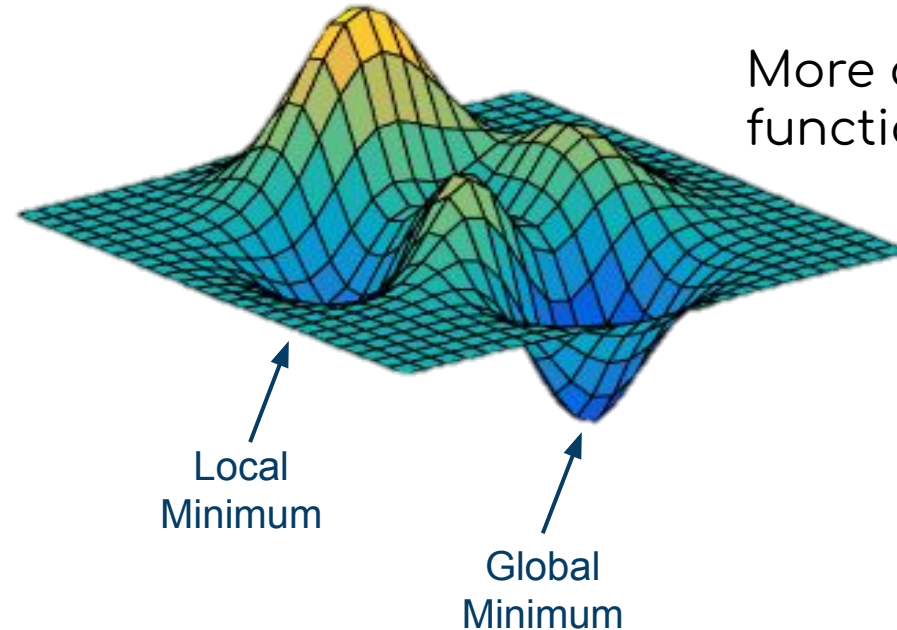
$$\hat{y}_i = \beta_0 + \beta_1 x_i$$

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



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 ($n_{\text{features}} \gg n_{\text{samples}}$)

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 ($n_features \gg n_samples$)
- How do we do it?
 - Modify the loss function
 - Constrain the learning process

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

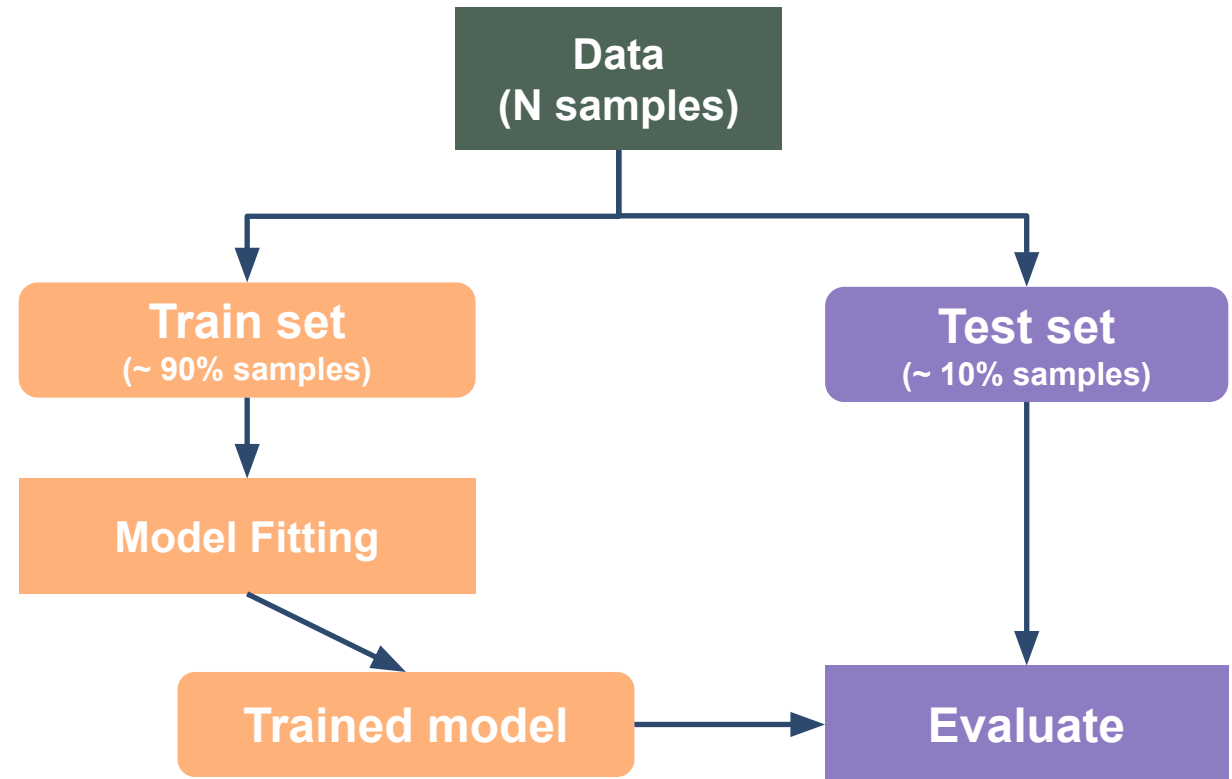
$$MSE = \sum_{i=1}^n (y_i - \underbrace{[\beta_0 + \sum_{j=1}^p x_{ij} \beta_j]}_{\hat{y}_i})^2 + \underbrace{\lambda \sum_{j=1}^p |\beta_j|}_{L_1}$$

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

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

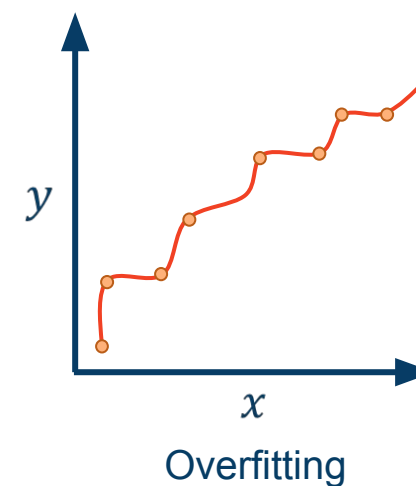
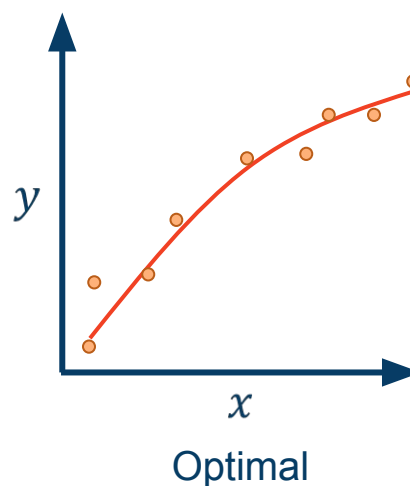
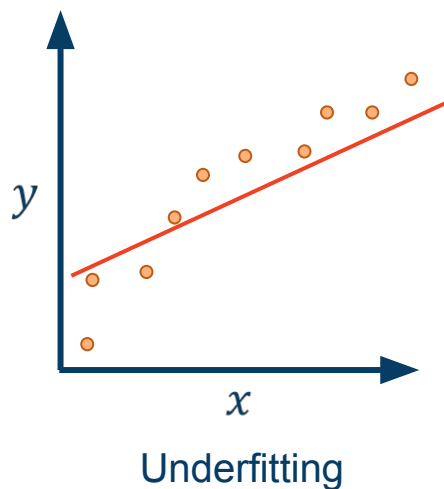
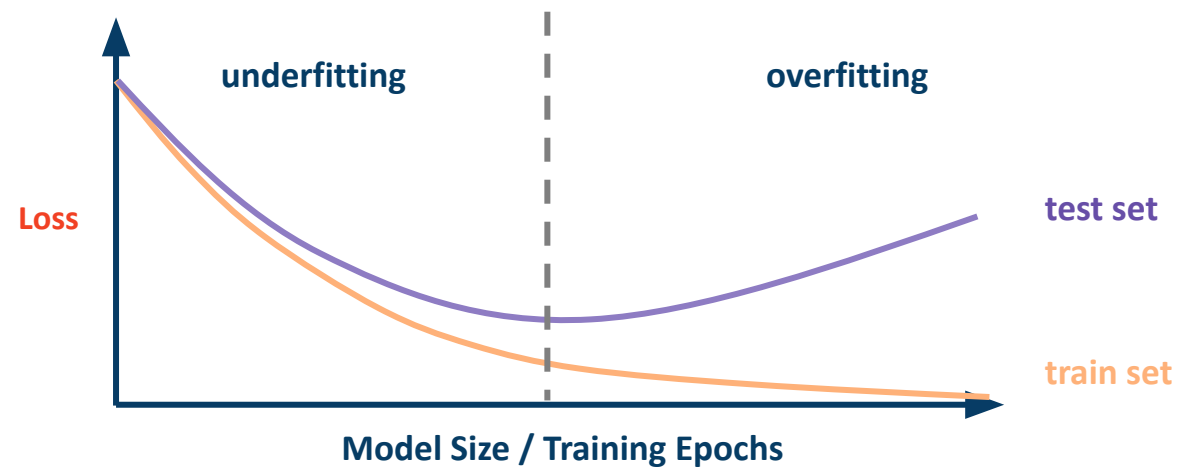
Model Evaluation

- What is model generalizability?
- Train performance \neq Test performance
- How do we sample train and test sets?
- How do we select a model?



Model Evaluation

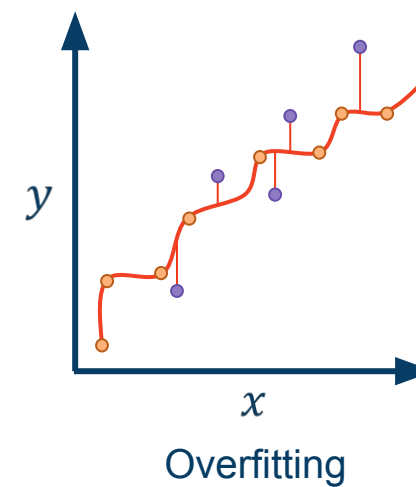
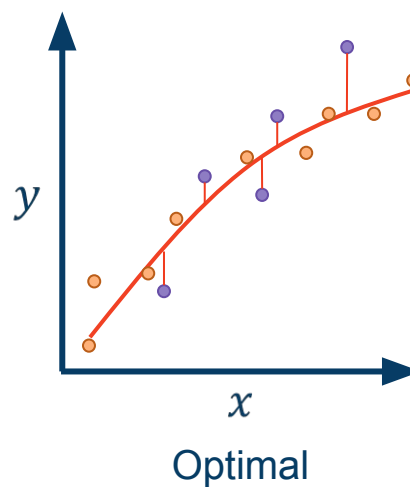
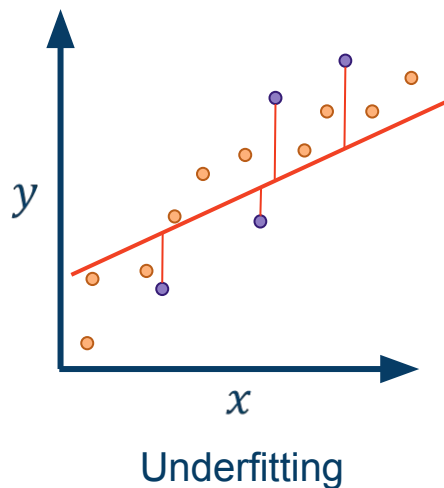
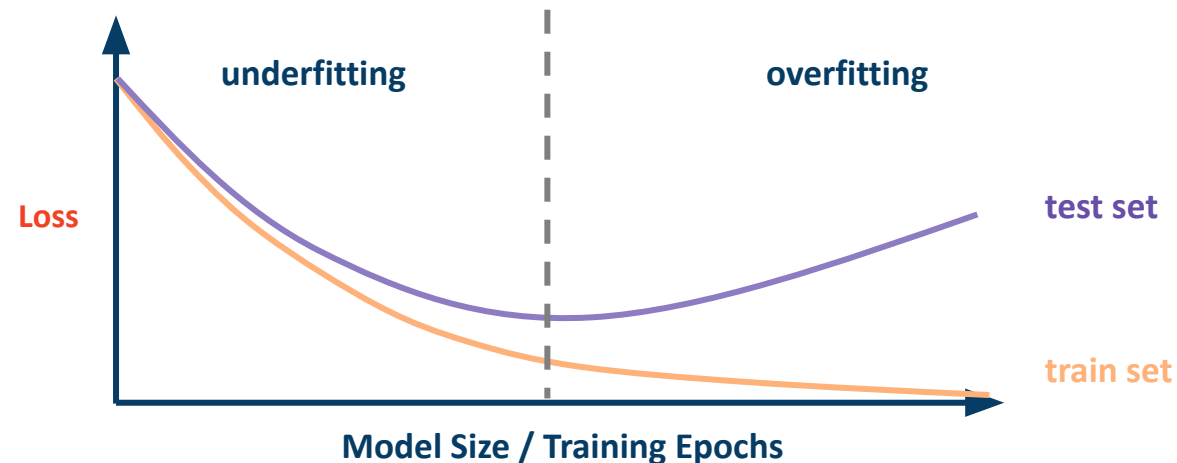
- Train performance \neq Test performance
 - Model: Overfitting vs underfitting
 - Errors: Bias - Variance tradeoff
 - Regression example



● Train set

Model Evaluation

- Train performance \neq Test performance
 - Model: Overfitting vs underfitting
 - Errors: Bias - Variance tradeoff
 - Regression example

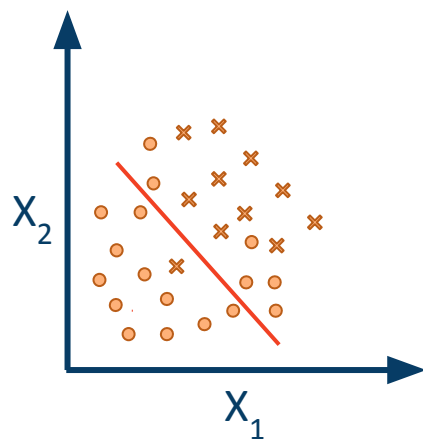
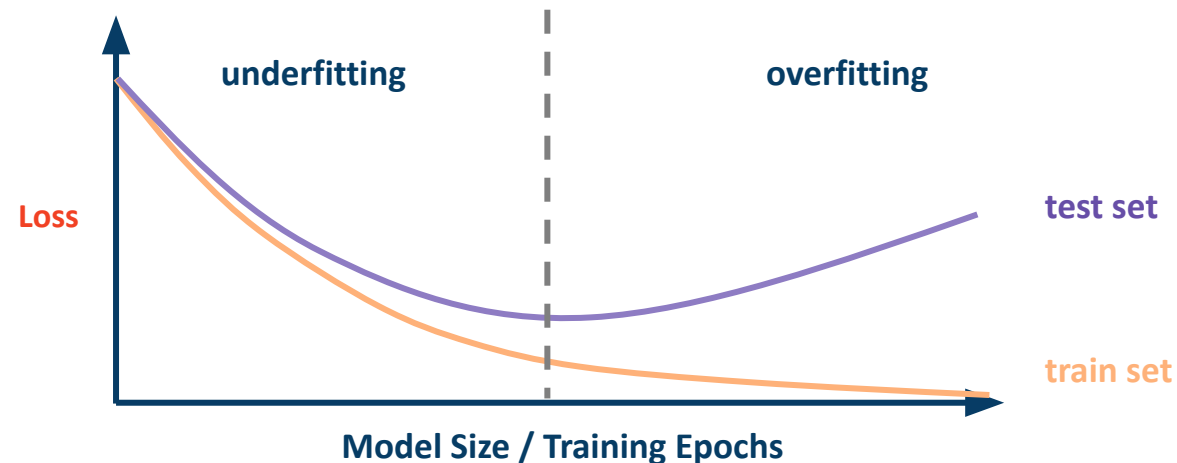


● Train set

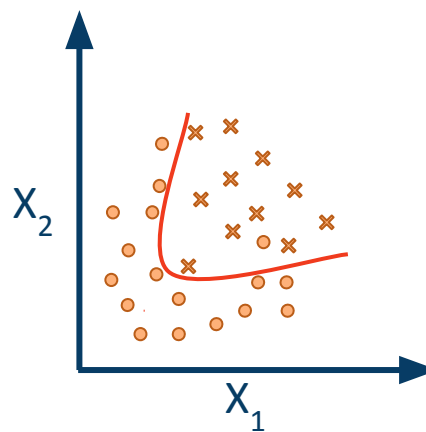
● Test set

Model Evaluation

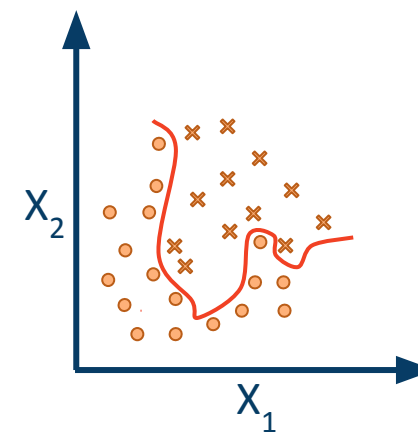
- Train performance \neq Test performance
 - Model: Overfitting vs underfitting
 - Errors: Bias - Variance tradeoff
 - Classification example



Underfitting



Optimal



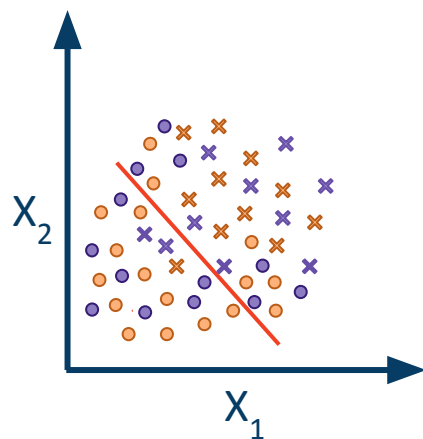
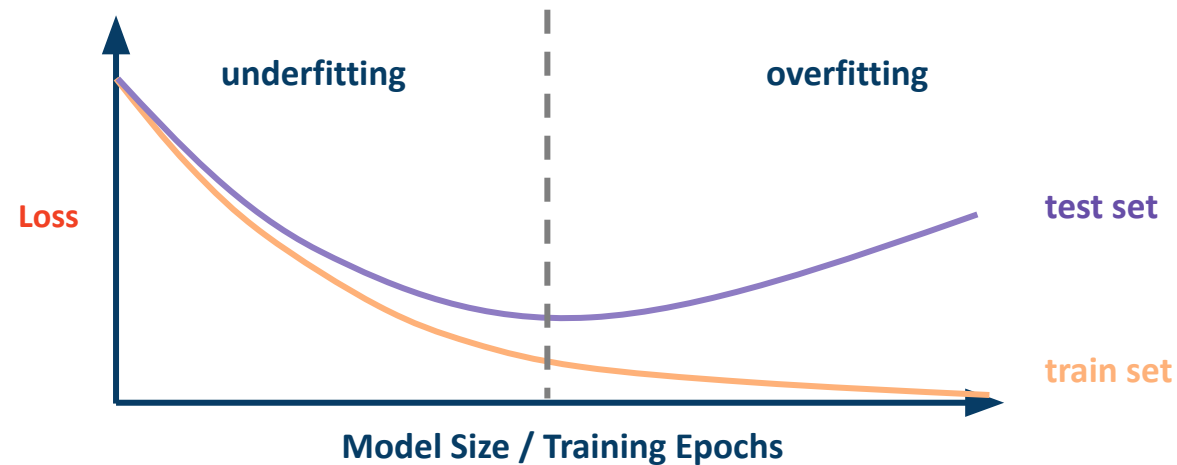
Overfitting

○ Train class_1

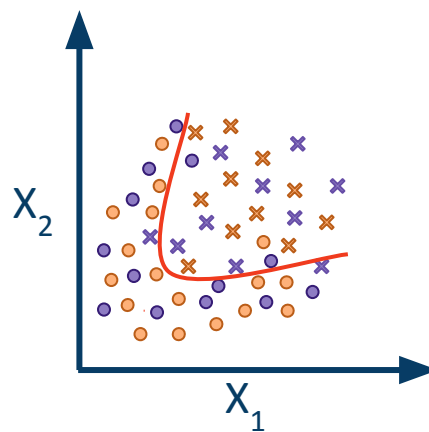
✕ Train class_2

Model Evaluation

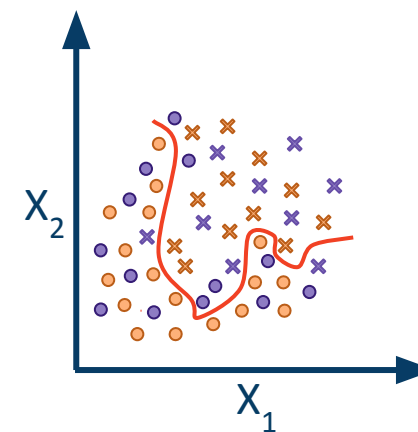
- Train performance \neq Test performance
 - Model: Overfitting vs underfitting
 - Errors: Bias - Variance tradeoff
 - Classification example



Underfitting



Optimal



Overfitting

● Train class_1

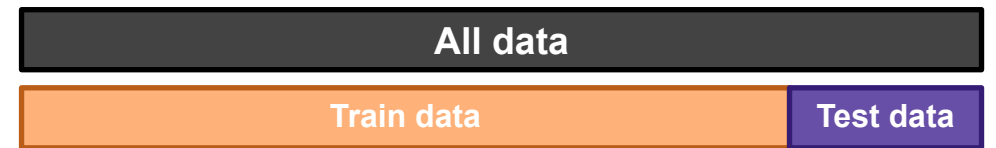
✕ Train class_2

● Test class_1

✕ Test class_2

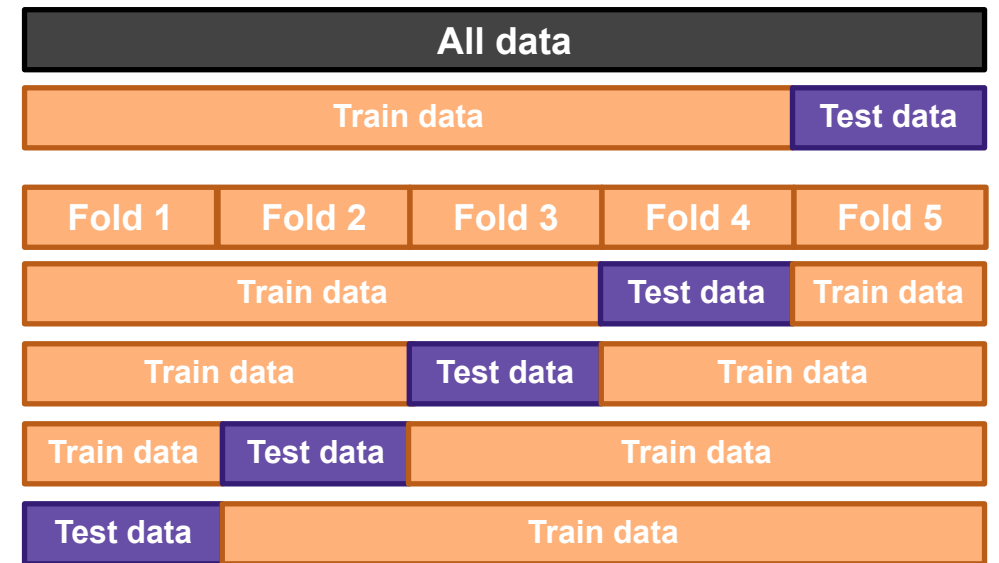
Model Cross-Validation

- 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

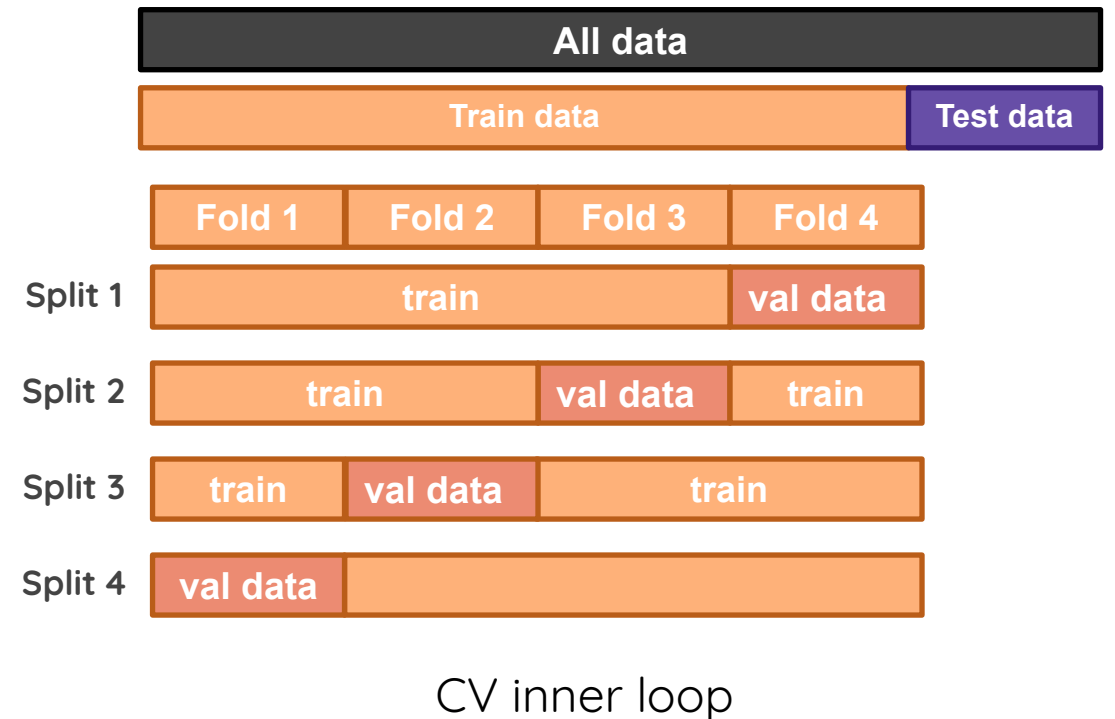
- How do we sample train and test sets?
 - Train set: learn model parameters
 - Test set (a.k.a held-out sample): Evaluate model performance
 - Repeat for different Train-Test splits
 - k-fold, shuffle-split
 - Report performance statistics over all test folds



CV outer loop

Model 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



Hyper-parameters

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

Hyper-parameters

- Hyper-parameter \neq 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 \neq 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 \rightarrow eg. cortical thickness and age have quadratic relationship.
 - Arbitrarily \rightarrow we gotta start with something!
 - Trial and error \rightarrow do a computationally feasible grid-search.

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!

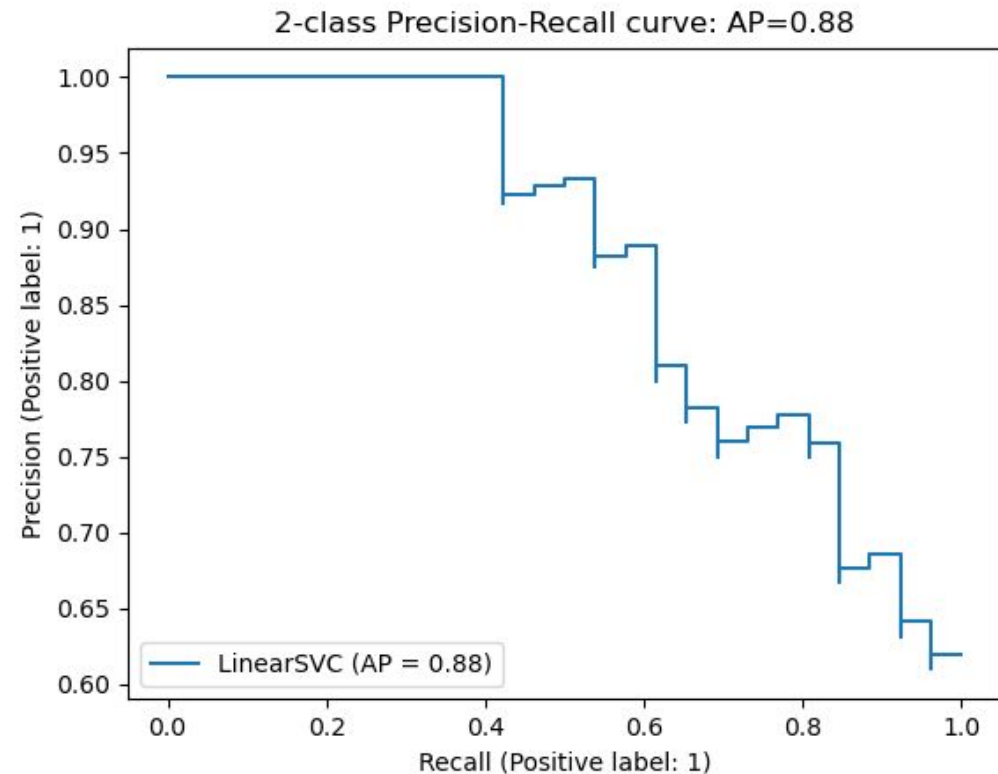
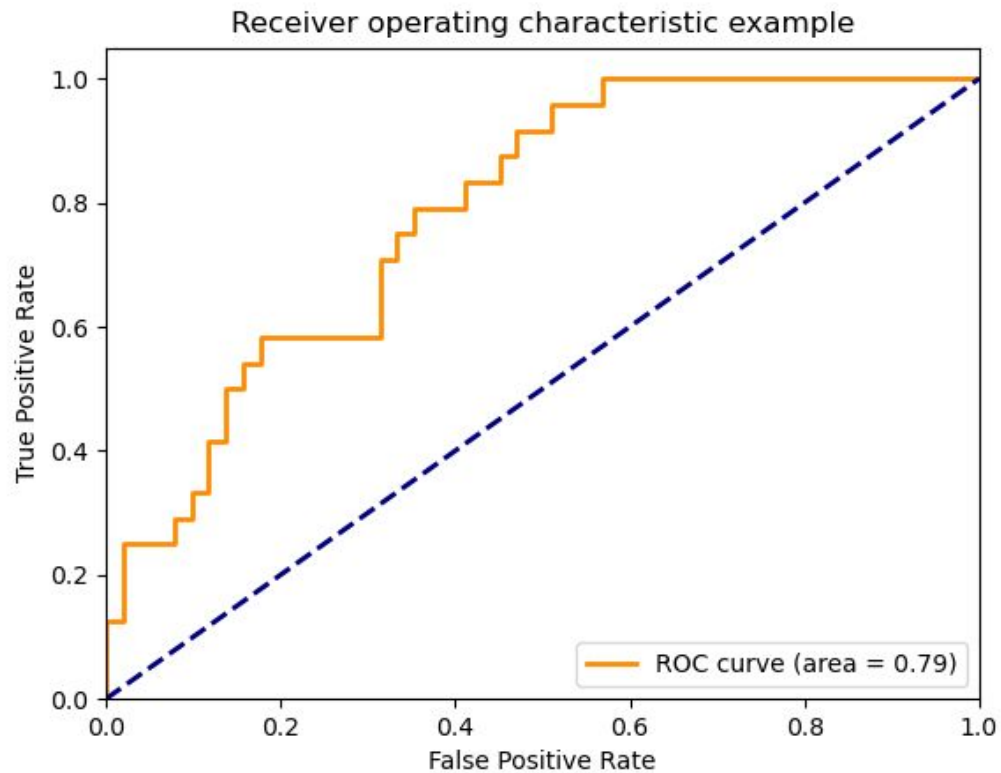
Performance Scores

- 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

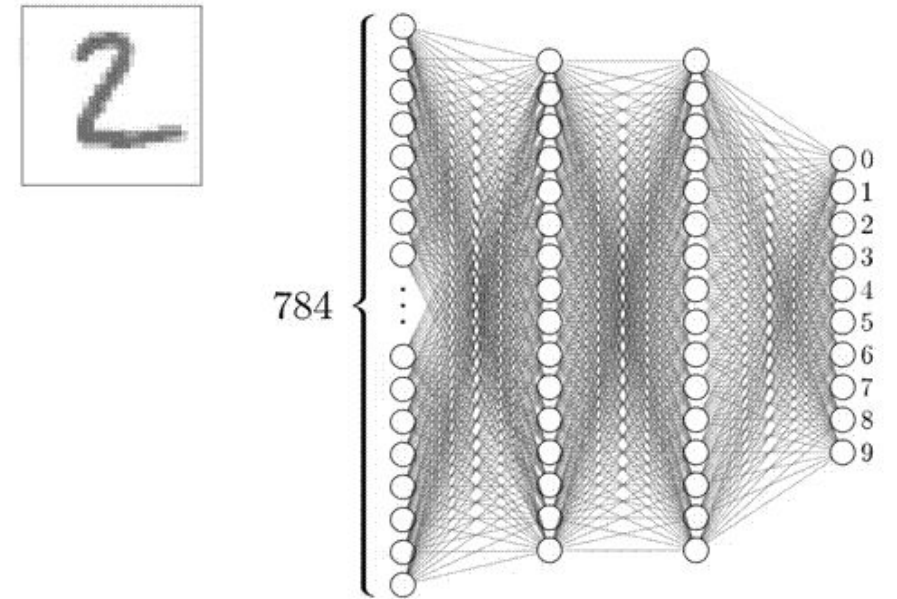
Performance Curves

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



Deep-learning

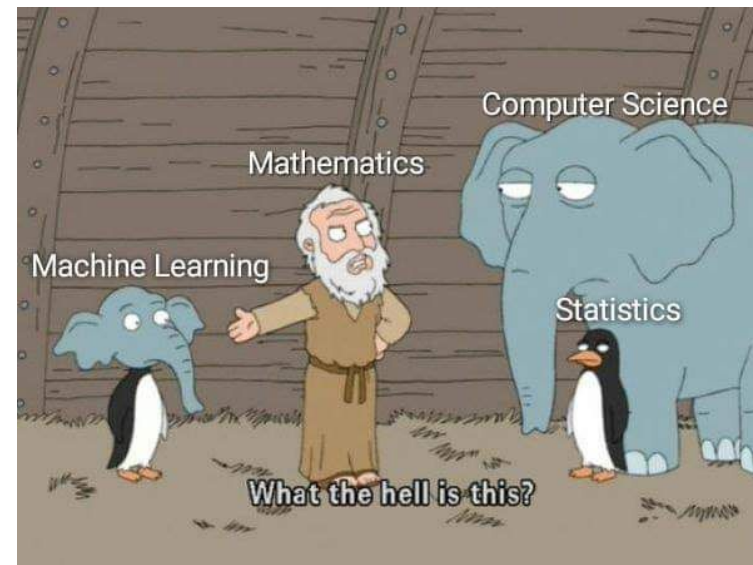
- Why the buzz?
 - Works amazing on structured input
 - Highly flexible → universal function approximator
- What are the challenges?
 - Large number of parameters → data hungry
 - Large number of hyper-parameters → difficult to train
- When do I use it?
 - If you have highly-structured input, eg. medical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



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

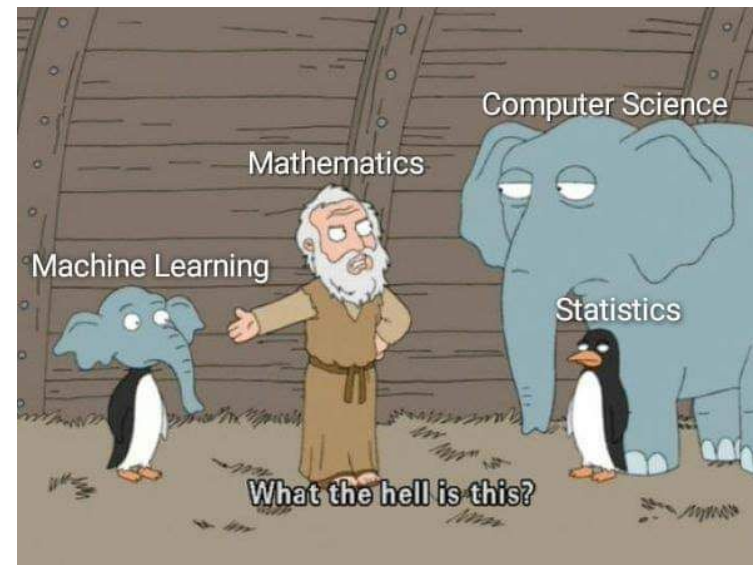
Pitfalls and Challenges

- 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

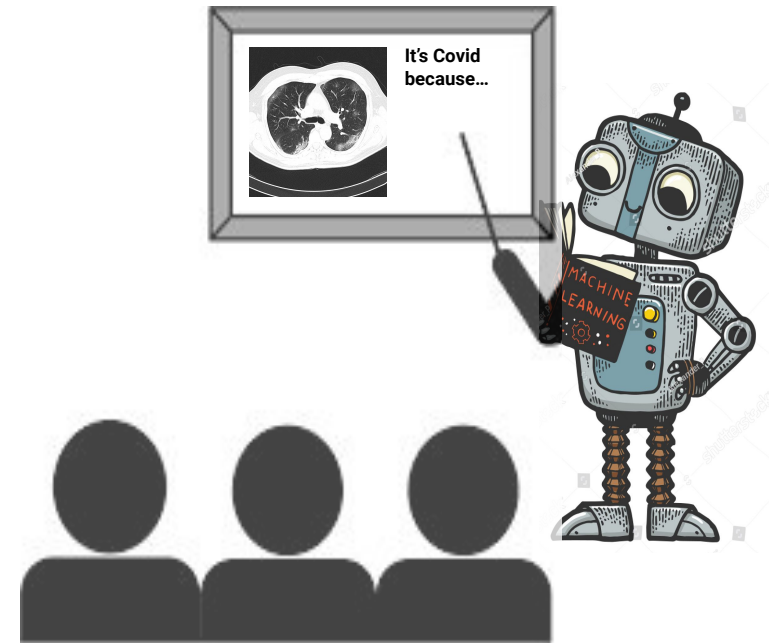
- What is my n_features and n_samples?
- Am I [encoding](#) categorical data correctly?
- Am I using information (e.g. mean) from test set to preprocess (eg. zscore) the data?

○ 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

End of Part 1