Machine Learning

Pedro M. Santos

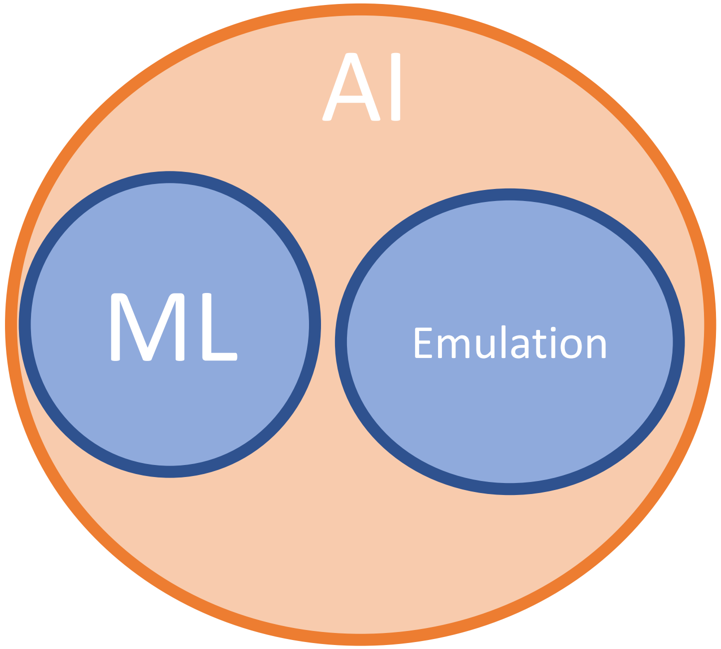
Assistant Researcher, CISTER Research Center, Instituto Politécnico do Porto

Invited Assistant Lecturer, Faculdade de Engenharia, Universidade do Porto

Profoundly based on the slides of Jaime S. Cardoso (check out his page!)

Historical Perspective on Machine Learning & AI

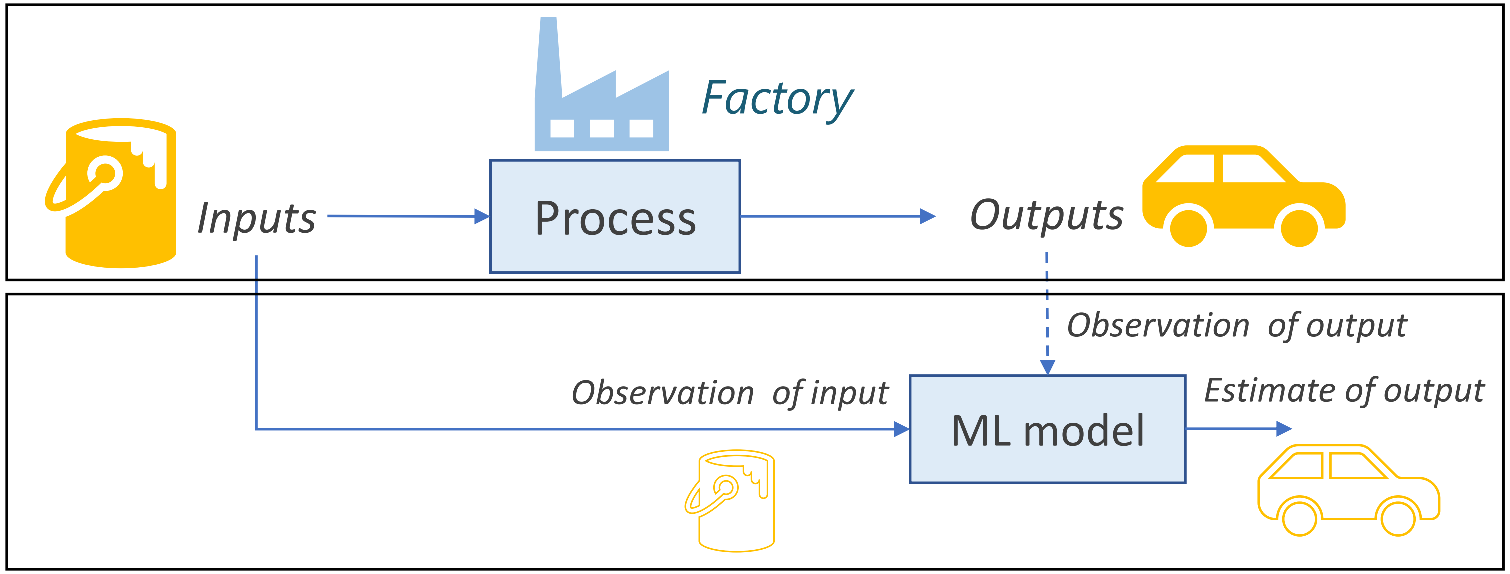
- The first waves of AI (around 60’s-70’s) development looked into **the emulation of human-like processes** (decision-making, problem-solving, learning). Resulting AI systems were efficient only in very narrow applications.
- In the 80’s, AI development took a turn to data-based approaches and still continues, leading **Machine Learning** to be the main driving force behind current AI efforts.



Machine Learning

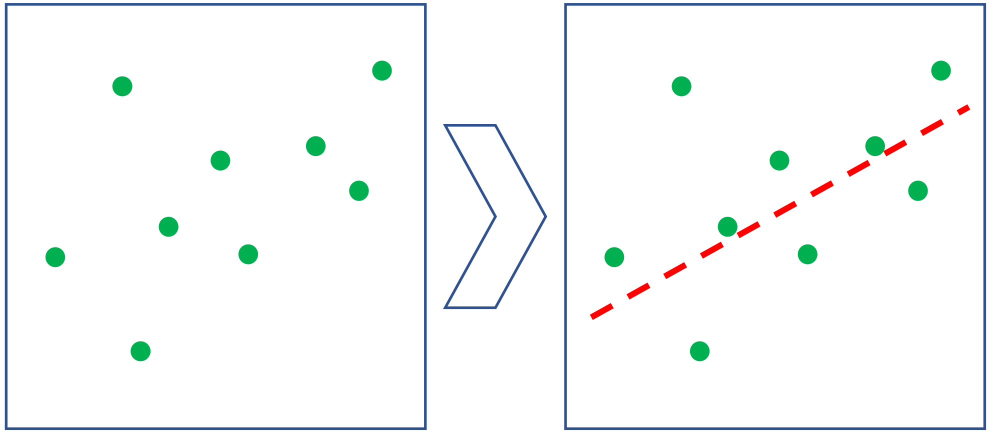
A pragmatic definition: Collection of algorithms and statistical models (methods) for machines to carry out automated tasks ***based on the observation of inputs and/or outputs of a process***

- The goal of Machine Learning is to produce an estimate or a classification given a set of input values.
- We often distinguish:
  - ML method: the mechanism to train a model (NN, SVM, DT, etc.)
  - ML model: a instance of the method trained to replicate the behaviour of the target process

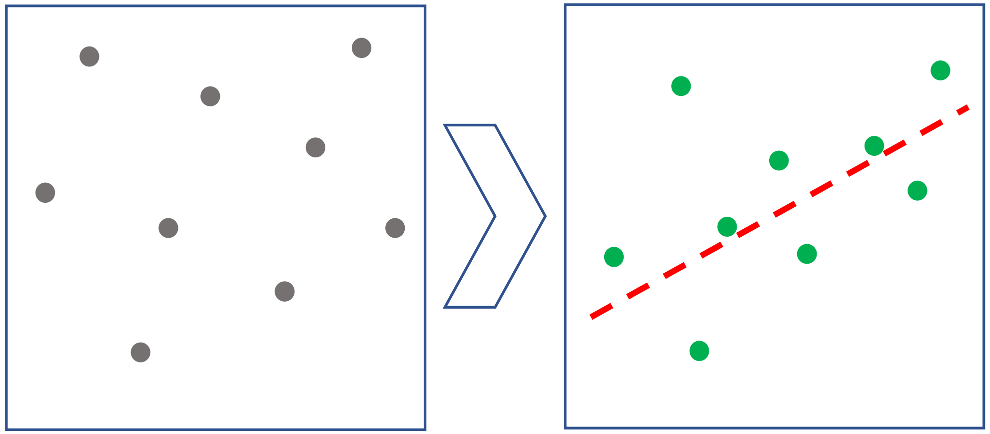


Types of Tasks and Features

- Regression: estimates values between known data points
- Classification: assign a class (from 2 or more classes) to data points



Regression (interpolate values)



Classification (assign classes to points)

- Features & Dimensionality
  - Features: relevant characteristics or processed metrics (average, std.dev., etc.) of the raw dataset that are feed to ML model
  - The number of input features typically defines the dimensionality of problem the ML mechanism needs to solve

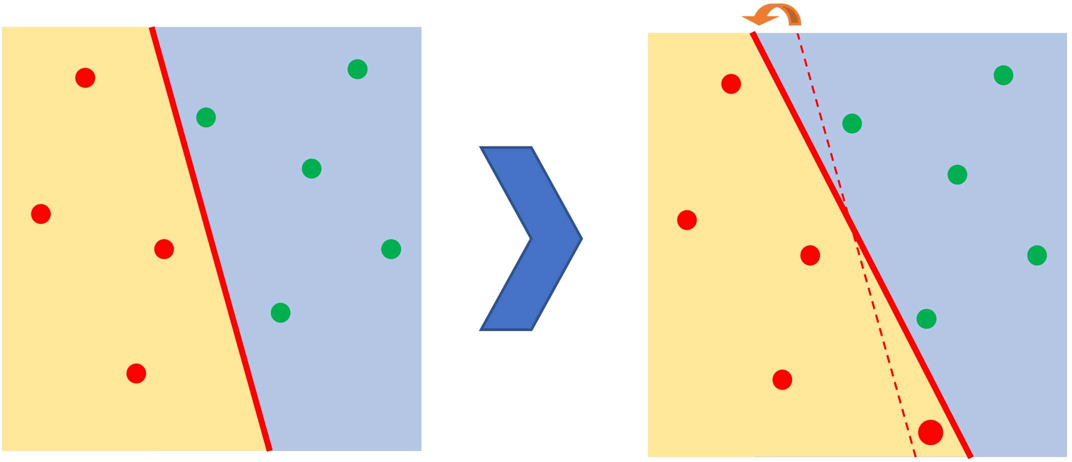
Input Data	
Dimension/Feature 1	**Dimension/Feature 2**
A	1
B	1
C	2

Types of Learning

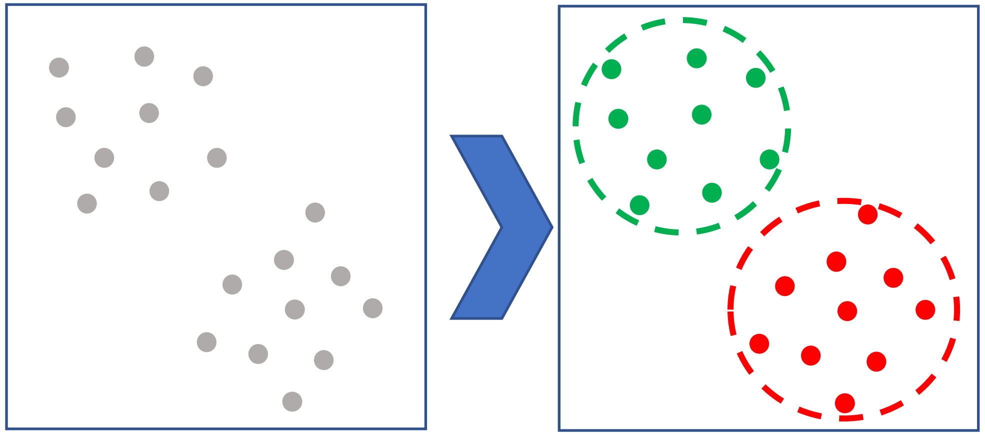
Ground Truth
necessary for training

Historical Dataset		
Feature 1	**Feature 2**	**Class**
A	1	X
B	1	X
C	2	Y

- Supervised: model is trained with a dataset of the target process
  - When training for a classification task, the historical dataset should contain the **Ground Truth** – the actual class of a given sample
- Unsupervised: classification or regression does not depend on prior knowledge



Supervised (classification depends on historical inputs)

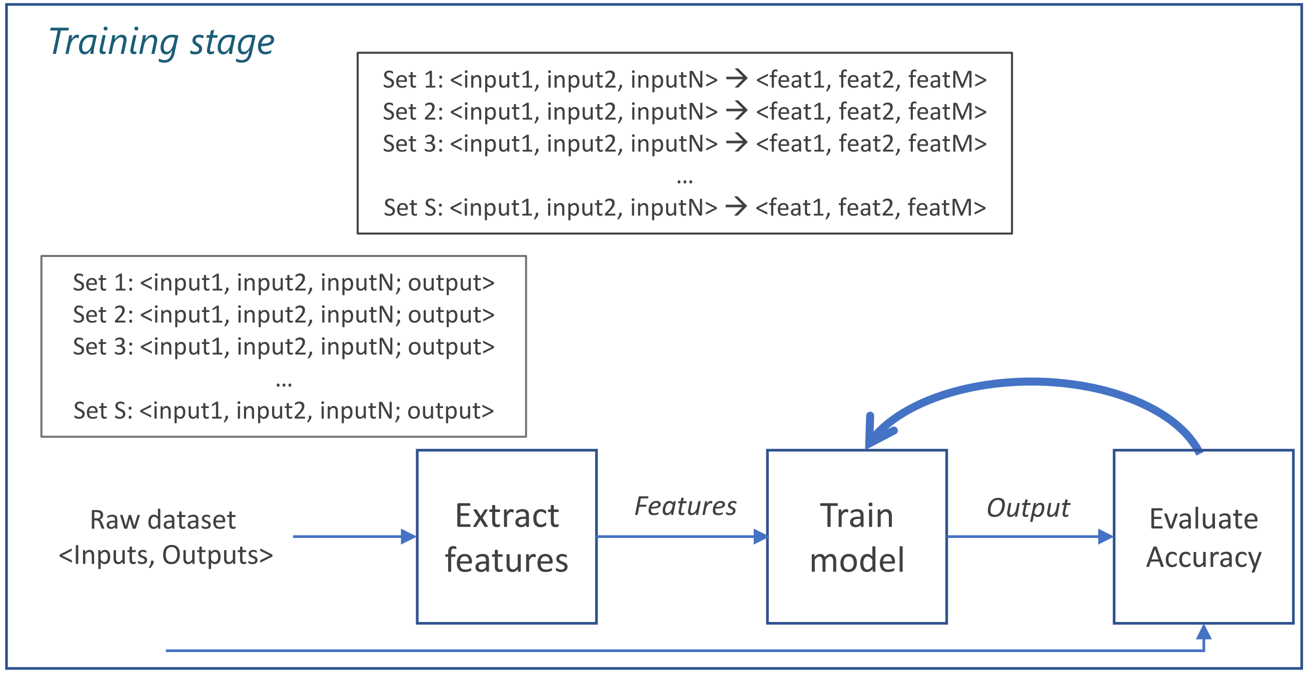


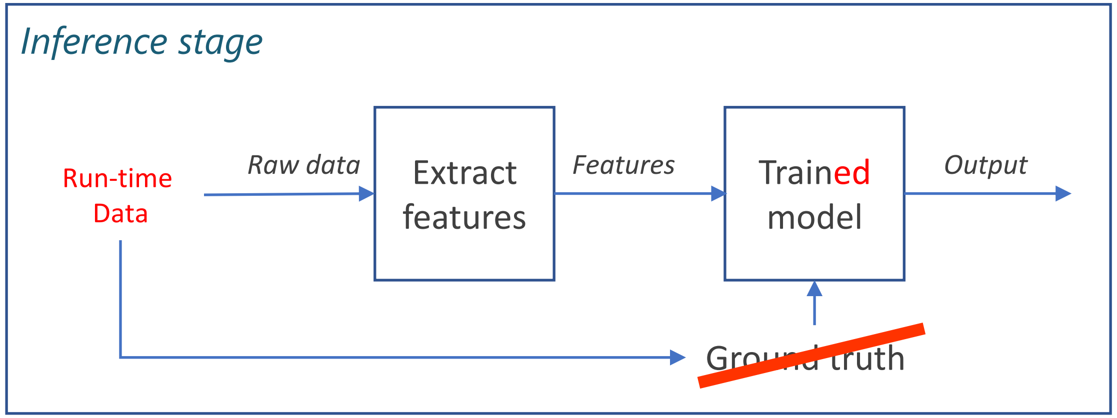
Unsupervised (classes are created by, e.g., finding clusters of similar data points)

Stages of supervised training

- Training stage
  1. A dataset with input data and corresponding output is presumed to exist. The input data is pre-processed to identify and/or extract relevant features. Pairs of <input feature set, actual output> are produced.
  2. The feature data is input to the ML method, typically one feature set. For each input set, the method produces an estimate likely to have an error.
  3. The method compares the estimate with the actual process output (the Ground Truth), and updates the model's internal processes to improve the accuracy of the estimates.
  4. The process is repeated until performance of the method is within acceptable bounds. We can now say that we have a **ML model** of the target process.

- Inference
  - The trained model is deployed in its target setting. Given inputs, it can produce estimates of the process output.
  - However, the method not longer has access to the ground truth, and it thus enable of further learning.





A Review of the Main Families

- Examples of Supervised Learning methods
  - Regression
  - Neural Networks
  - Support Vector Machines
  - Decision Trees
  - Bayes
  - Markov Chains
- Examples of Unsupervised Learning methods
  - K-means
  - Self-Organizing Map

When and which to use?

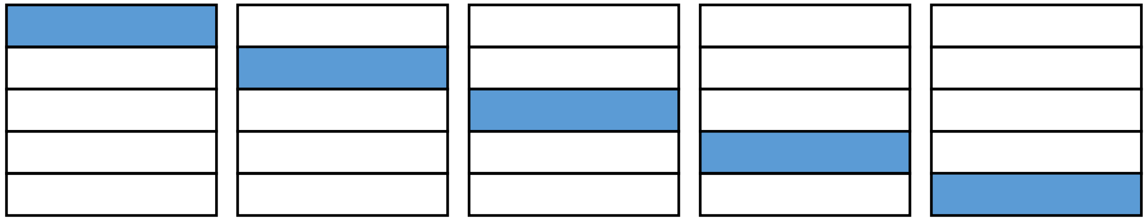
- Rule-of-thumb: When you have data ☺ But there are some more filters to go through. Here's a checklist:
- Does my problem really need such a complex ML-based solution?
  - ML is often over-used as a tool for solving/modelling for relatively simple processes
  - ML methods are appropriate when the **process being learned is complex** and **cannot be easily modelled by other tools** (e.g., closed analytical formulations, heuristics, probabilistic/statistical characterization, or even a rule-based mechanisms)

Opinion; take it at your discretion

- Supervised learning: Do I have the ground truth?
  - In order to train your method, you need annotated data (i.e., the data points must have an associated classification).
  - Such datasets often are difficult to obtain (e.g., may require manual classification).
  - Make sure you have them if planning to use Supervised Learning.
- Unsupervised learning: Can my data be separated into meaningful or "distinguishable" classes?
- Which one to use?
  - Really depends on the problem; having a good overview of existing methods is the best way of knowing that. At least it is (usually) fairly simple to identify if the approach should be supervised or unsupervised.

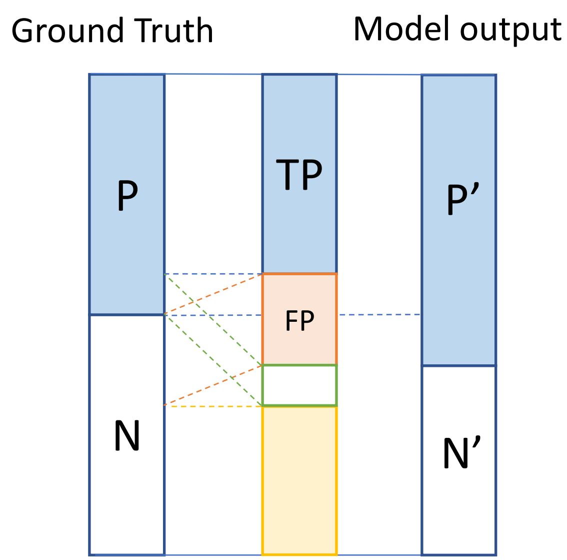
Data Preparation

- Data Cleaning
  - If using a real-world dataset, there is a lot of data cleaning to do, i.e., identify strongly biasing data sources (unbalanced dataset), removing outliers, etc.
  - If data is not properly cleaned, it may lead to over-fitting
- Features and Dimensionality Reduction
  - It may be necessary or useful to refine raw data into relevant characteristics (e.g., mean, median, std. dev.)
  - Sometimes a blind set of features is produced, and then only the most relevant are selected (e.g., decision trees)
- Separating Training dataset and Test dataset (Hold-out)
  - Training dataset is used to train the model; test dataset is used to evaluate the performance of the trained model
  - As a rule of thumb, from the initial dataset, you take a larger chunk (between 60% and 90%) for the training dataset, and remainder for the test set
- Cross-validation
  - Running the training multiple times, using different subsets of the dataset as test dataset



Metrics in Classification

Example with 2 classes



- Given classes Positive and Negative:
- Direct metrics:
  - True Positives (TP): samples correctly classified being of class Positive
  - True Negatives (TN): samples correctly classified being of class Negative
  - False Positives (FP): samples incorrectly classified as being of class Positive
  - False Negatives (FN): samples incorrectly classified as being of class Negative
- Derived metrics:
  - Accuracy
  - Precision / Positive Predictive Value
  - True positive rate (TPR) / Recall / Sensitivity
  - False positive rate (TPR) / False alarm rate
  - F1-measure

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$

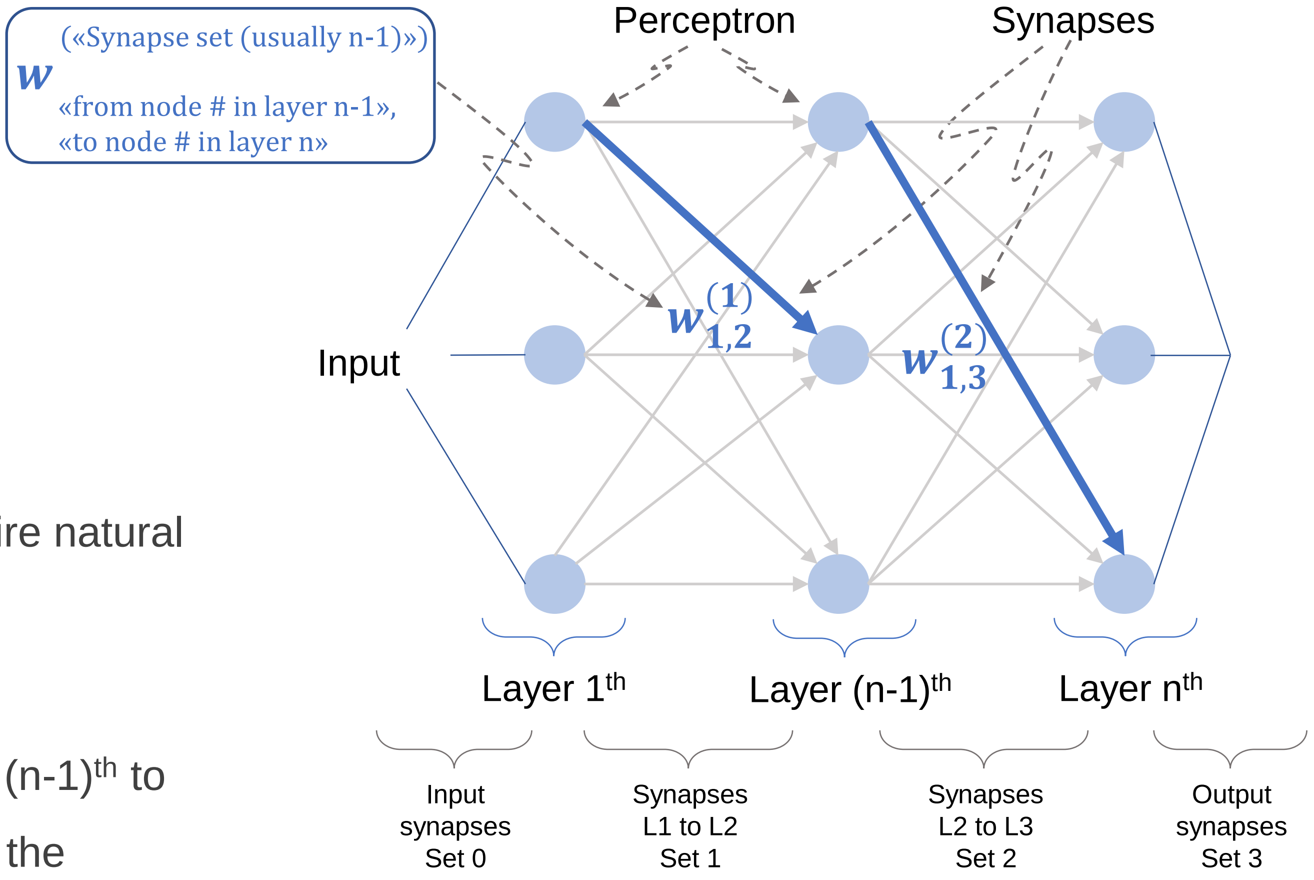
$$\text{Recall} = \frac{TP}{TP + FN} \qquad \text{False Alarm Rate} = \frac{FP}{TN + FP}$$

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{F1-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Supervised Learning

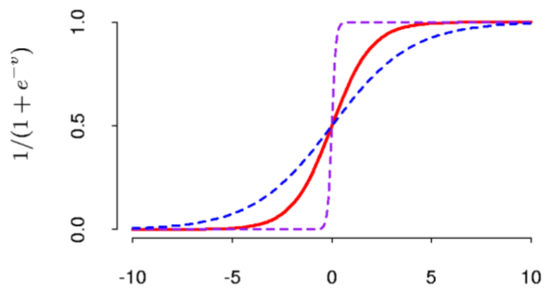
Neural Networks

- One of the most successful methods of ML
- Building blocks: **Perceptron** and **Synapse**
- **Perceptron:** typically a function that maps the entire natural range into a bounded interval ([0,1] or [-1,1])
  - Example: Logistic sigmoid function
- **Synapses:** connections from perceptrons of layer (n-1)th to perceptrons of layer nth, each applying a weight to the transmitted value
- Training Neural Networks is mostly about finding the weights of those synapses,



Logistic sigmoid function

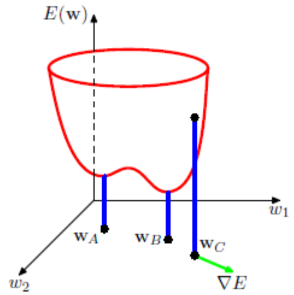
$$\sigma = \frac{1}{1+e^{-v}}$$



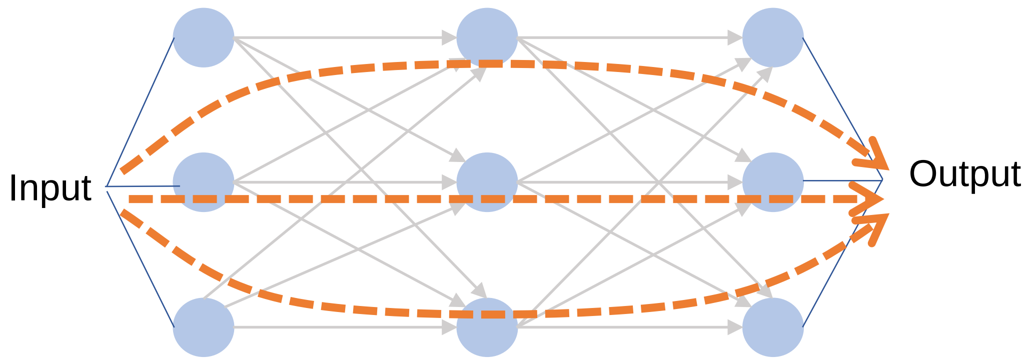
Training

- The weights of the synapses should be such that they minimize a given error function, in a process related/similar to gradient descent optimization
- **Inference:** also referred to as feedforward operation
- **Training:** The mechanism of weight updating is called **error backpropagation**
  1. New data point is input; feedforward is performed
  2. The output of feedforward operation is compared with the true (ground truth) value, and an error is computed.
  3. Output synapses are updated to mitigate error
  4. Update in weights of output synapses is **back-propagated** to the (n-1)th layer
  5. Procedure repeats until weights of input synapses are also updated

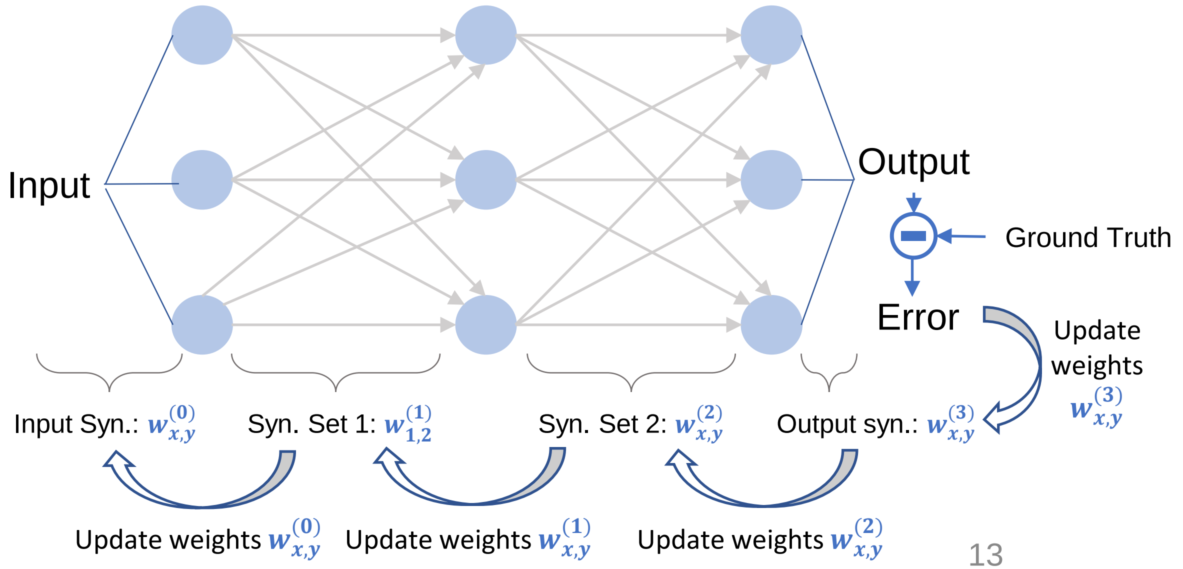
$y = f(x) + \varepsilon$



Inference: feed-forward operation



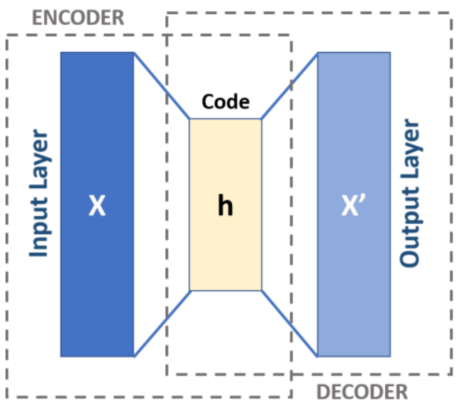
Training: error backpropagation



Some variants

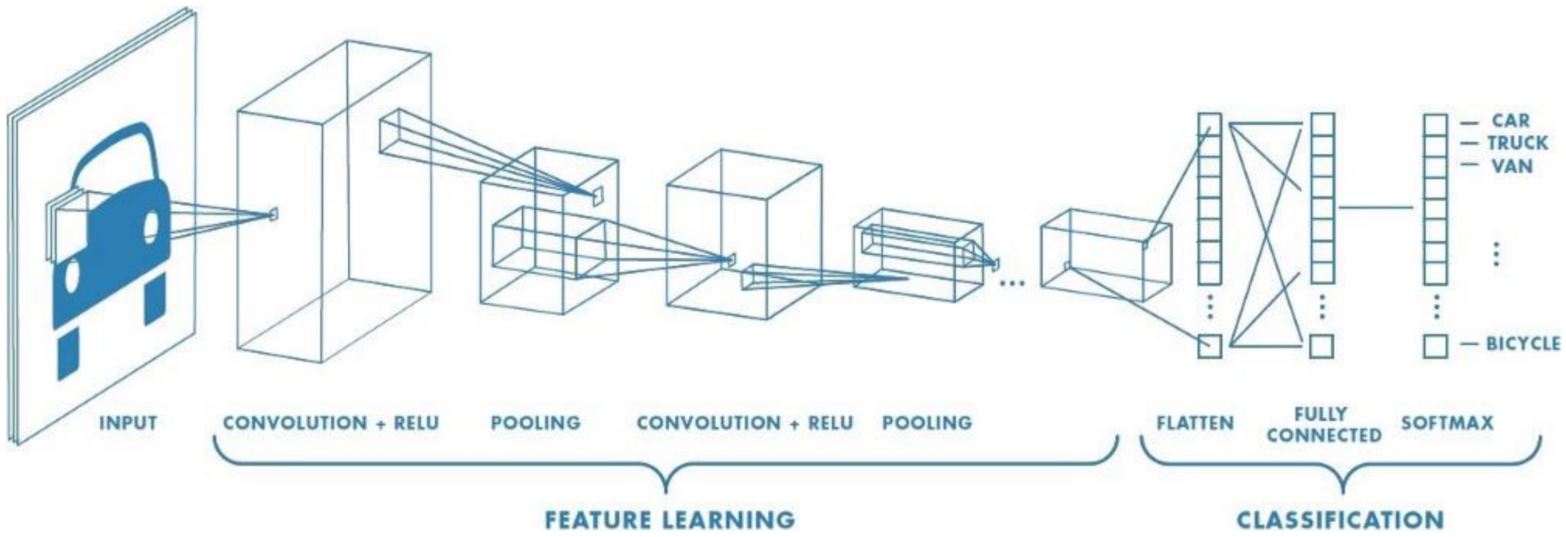
Feedforward	• Nodes of layer (n-1)th connect to perceptron of layer nth • **Multi-layer perceptron** (MLP)
Recurrent (RNN)	• Include cyclic connections, i.e., outputs of nodes of layer nth can be inputs to nodes of layers nth, (n-1)th and others • **Fully-recurrent:** output of all neurons are inputs to all neurons • **Long Short-Term Memory (LSTM):** a variant of RNN capable of storing information for either long or short periods of time.
Auto-encoder	• An unsupervised neural network with the target output the same as the input. • Inner synapses hold a "codified" version of the data.
Convolutional (CNN)	• Very successful in image processing.

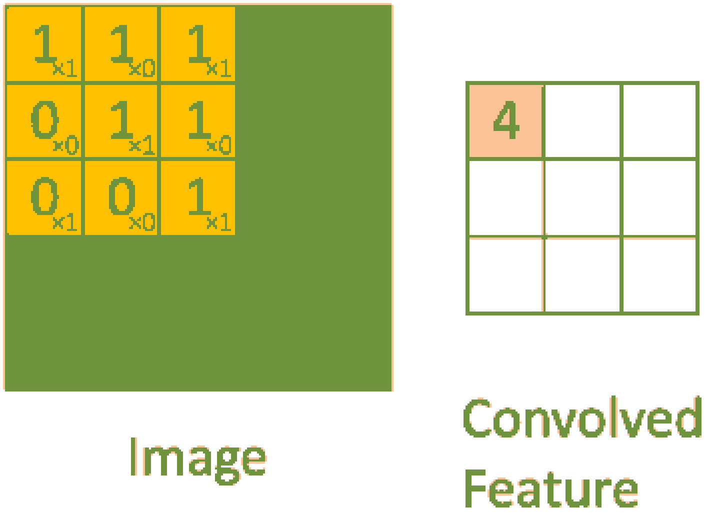
Auto-Encoder



By Michela Massi - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=80177333

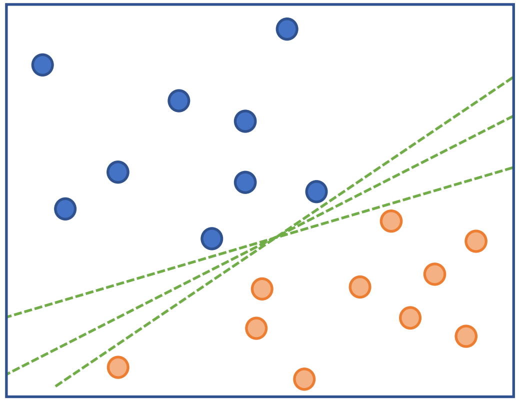
Convolutional Neural Networks

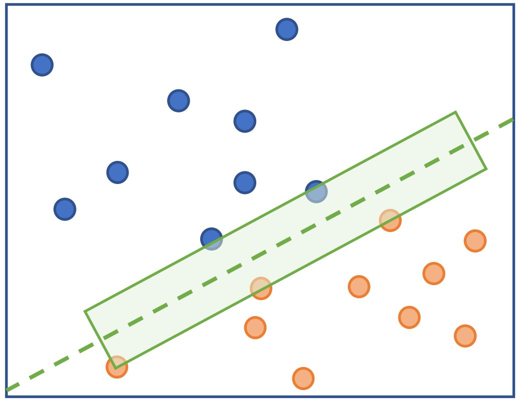




Support Vector Machines

- Different linear classifiers can produce a multitude of boundaries to discriminate two classes
- SVMs try to select a decision boundary for which the margin between data points of different classes is maximized





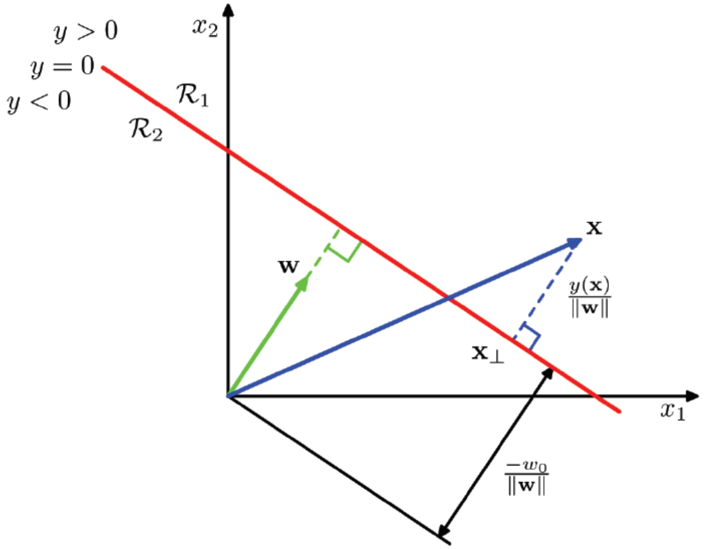
Background

- Assume a plane defined by $y(\mathrm{x}) = \mathbf{w}^{\mathrm{T}}\mathrm{x} + \mathrm{b}$
- Distance of a point x_n to the plane: finding the minimum distance of point x_n to a plane, $\|(x - x_n)\|$, is equivalent to finding the minimum of $\|(x - x_n)\|^2 = (\mathrm{x} - x_n)^{\mathrm{T}}\,(\mathrm{x} - x_n)$: $\frac{|y(x_n)|}{||w||}$
- The distance of a training point x_n (correctly classified) to the decision surface is given by

$$\frac{y_n y(\mathbf{x}_n)}{||\mathbf{w}||} = \frac{y_n(\mathbf{w}^T\mathbf{x}+b)}{||\mathbf{w}||}$$

- Thus, we want to maximize the (minimum) distance of all data points to the decision boundary:

$$\mathrm{argmax}_{\mathbf{w},b}\left\{\frac{1}{||\mathbf{w}||}\min(y_n(\mathbf{w}^T\mathbf{x}_n + b))\right\}$$



Lagrange Multipliers

- Lagrange Multipliers
  - **A strategy for finding the local maxima and minima of a function subject to equality constraints.**
  - Consider the optimization problem $\begin{cases} \text{Maximize (or minimize): } f(x, y) \\ \text{subject to: } g(x, y) = c \end{cases}$
  - To find the maximum (or minimum) of $f(x, y)$, form the Lagrangian function $L = f(x) - \lambda\, g(x)$ and find the stationary points of L considered as a function of x and the Lagrange multiplier λ.
  - If $f(x_0, y_0)$ is a maximum of $f(x, y)$ for the original constrained problem and $\nabla g(x_0, y_0) \neq 0$, then there exists λ_0 such that $(x_0,\, y_0,\, \lambda_0)$ is a stationary point for the Lagrange function (points where the first partial derivatives of L are zero).

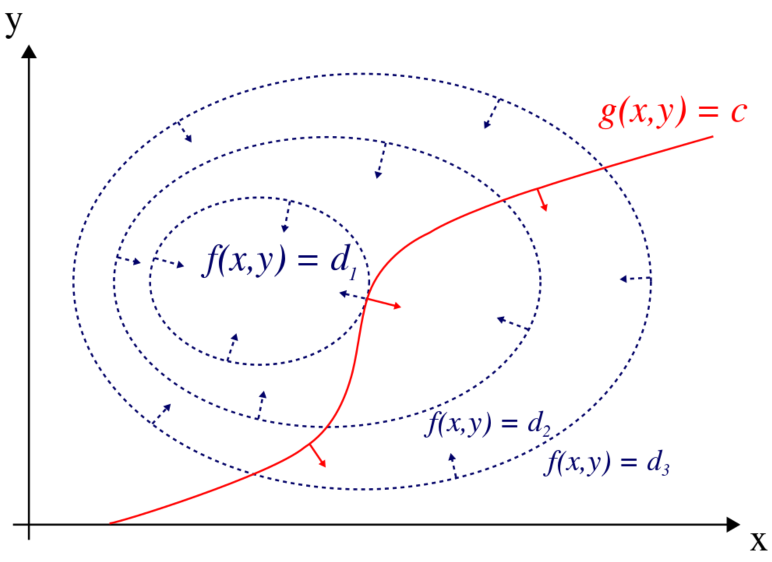


Figure 1: The red curve shows the constraint *g*(*x*, *y*) = *c*. The blue curves are contours of *f*(*x*, *y*). The point where the red constraint tangentially touches a blue contour is the maximum of *f*(*x*, *y*) along the constraint, since $d_1 > d_2$.

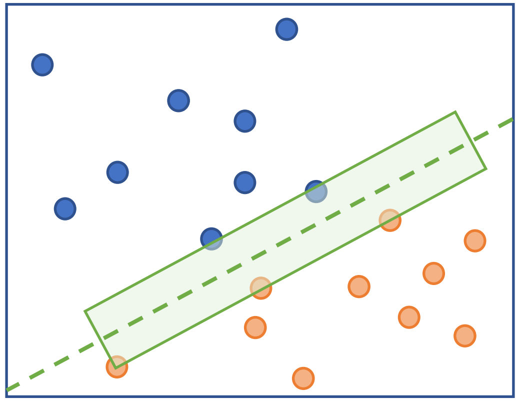
- Explanation
  1. Suppose we walk along the contour line with $g = c$. We are interested in finding points where f almost does not change as we walk, since these points might be maxima.
     We could touch a contour line of f, since by definition f does not change as we walk along its contour lines.
     This would mean that the tangents to the contour lines of f and g are parallel here.
  2. Thus we want points (x, y) where $g(x, y) = c$ and $\nabla_{x,y} f = \lambda \nabla_{x,y} g$ for some λ, with $\nabla_{x,y} f = \left(\frac{\partial f}{\partial f}, \frac{\partial f}{\partial y}\right)$ and $\nabla_{x,y} f = \left(\frac{\partial f}{\partial f}, \frac{\partial f}{\partial y}\right)$
  3. We introduce an auxiliary function: $L(x, y, \lambda) = f(x, y) - \lambda\, g(x, y)$ and solve $\nabla_{x,y,\lambda} L(x, y, \lambda) = 0$. Note that this amounts to solving three equations in three unknowns.

Support Vector Machines

- Recall that:
  1. We want to maximize the distance of data points to the decision boundary:

$$\text{argmax}_{\mathbf{w},b}\left\{\frac{1}{||\mathbf{w}||}\min(y_n(\mathbf{w}^T\mathbf{x}_n+b))\right\}$$

  2. Finding the minimum distance of point x_n to a plane, $\|(x-x_n)\|$, is equivalent to finding the minimum of $\|(x-x_n)\|^2 = (\mathrm{x}-x_n)^{\mathrm{T}}\,(\mathrm{x}-x_n)$



- In turn, the problem of finding the minimum of $(\mathrm{x}-x_n)^{\mathrm{T}}\,(\mathrm{x}-x_n)$ subject to $w^{\mathrm{T}}x+\mathrm{b}=0$ is equivalent to the problem of finding the minimum of the Lagrange multipliers $L=(\mathrm{x}-x_n)^{\mathrm{T}}\,(\mathrm{x}-x_n)+\lambda(w^Tx+\mathrm{b})$ for the variables x and λ

- After some mathematical manipulation, we arrive to a formulation of the problem based on Lagrange multipliers:

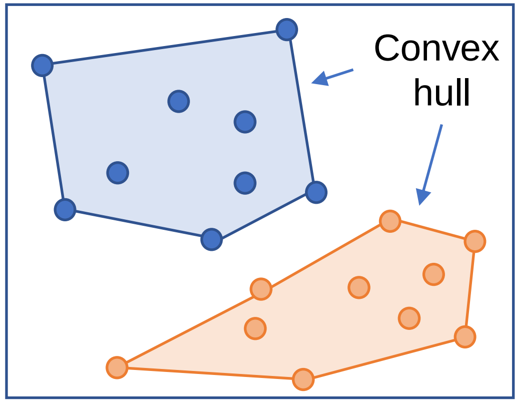
$$\begin{array}{ll}\text{argmax}_\lambda & \sum_{n=1}^{N}\lambda_n-\frac{1}{2}\sum_{i,j}\lambda_i\lambda_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j\\ \text{subject to:} & \sum_n \lambda_n y_n = 0\\ & \lambda_i\geq 0, i=1,\cdots,N\end{array}$$

An Alternative Explanation

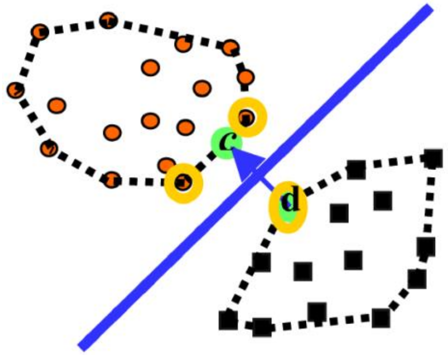
- Convex hull:
  - Given a set of points $\{x_i\}$, we define the convex hull as the set of all points such that

$$\mathbf{x} = \sum_i \alpha_i \mathbf{x}_i$$

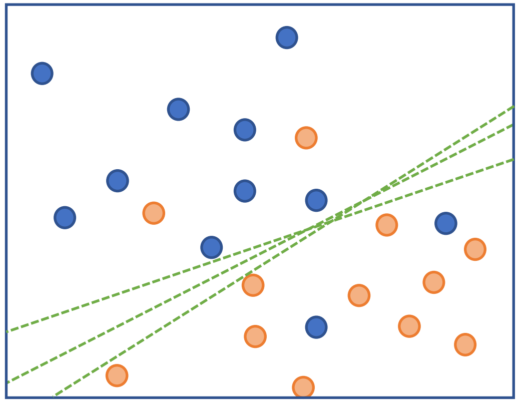
$$\text{subject to: } \begin{cases} \alpha_i \geq 0 \\ \sum_i \alpha_i = 1 \end{cases}$$



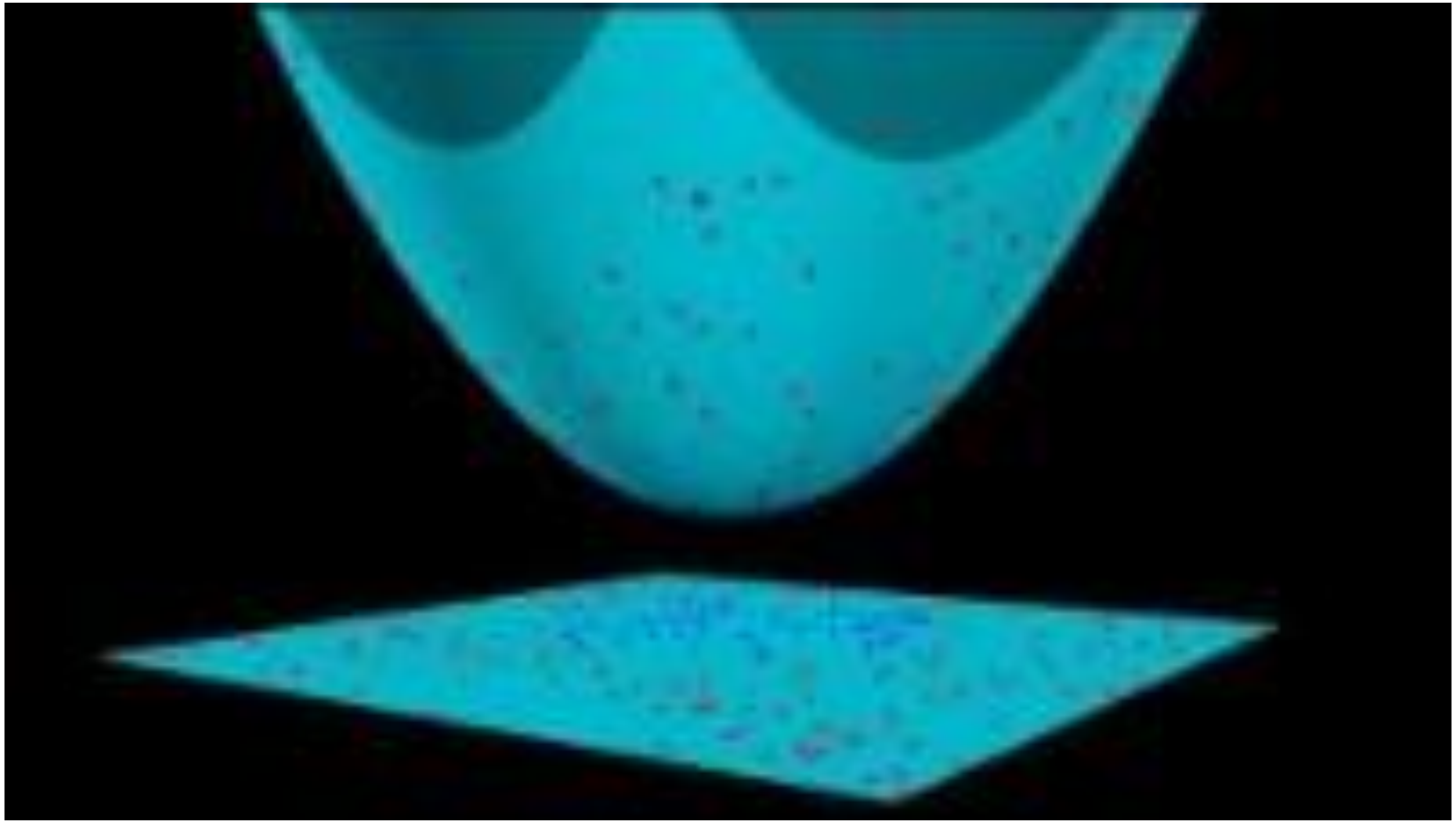
- A geometric interpretation
  - find the convex hull of both sets of points
  - find the two closest points in the two convex hulls
    (the convex hull is the smallest convex set containing the points)
  - construct the plane that bisects these two points



Non-linearly Separable Classes

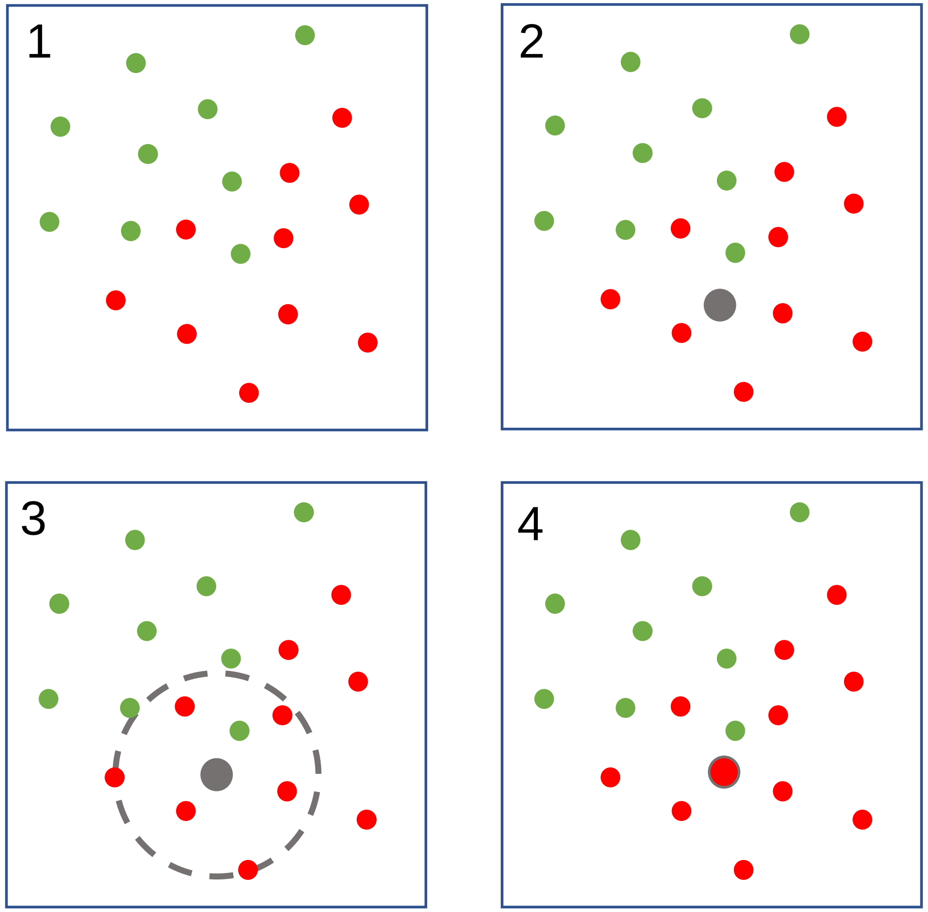


- What happens when classes are not **linearly separable**?
- We use **kernels**, which map the data points to a higher-dimensionality space where points can be linearly separated



K-Nearest Neighbours (KNN)

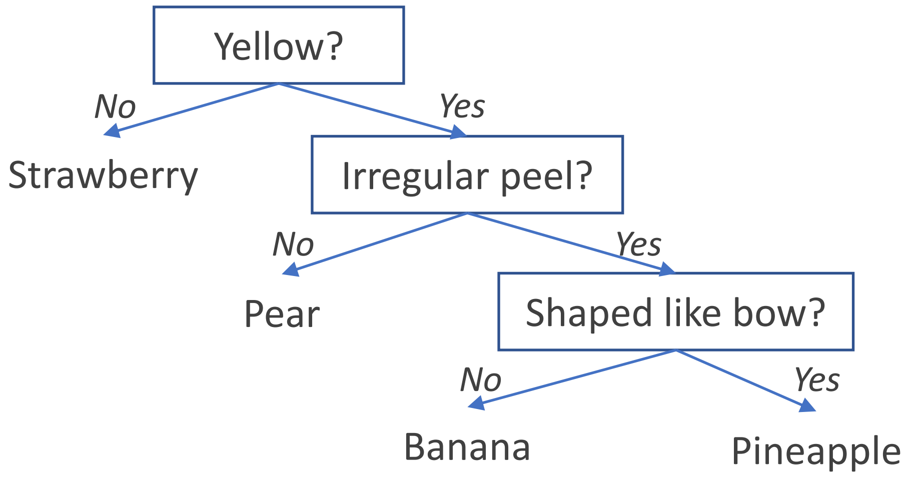
- Density Estimation Technique, Non-parametric
- Assigns classes to new data points by inspecting "tendency" of neighbours around
- Operation:
  1. Pre-existing set of data points already classified
  2. Define radius and number of neighbours to make decision
  3. For each new data point, number of neighbours within radius is computed



Decision Trees

- Decisions trees are a sequence of threshold-based deciders, where decision is made around a feature value.
- The main idea of the model training is to:
  - Ordering features by how discriminative they are (thus resulting in a sequence of threshold rules)
  - Identifying feature value to use as threshold

Universe: Banana, Pineapple, Pear, Strawberry



Feature: Color					
Fruit	Color	Similarity			
		Straw.	*Pear*	*Pine*	*Banana*
Straw.	Red	-	0	0	0
Pear	Yellow	0	-	1	1
Pine.	Yellow	0	1	-	1
Banana	Yellow	0	1	1	-

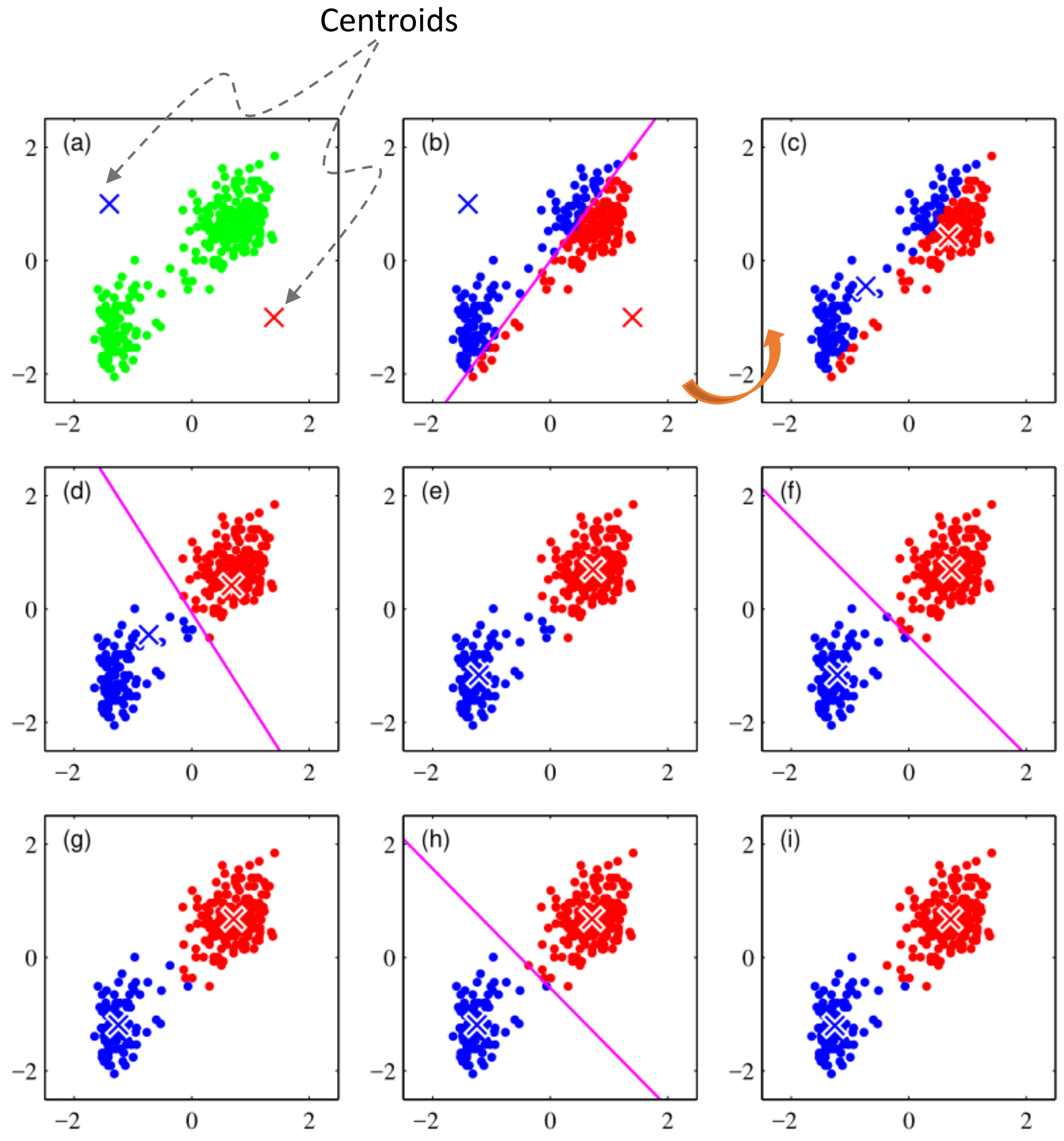
Feature: Peel Texture					
Fruit	Texture	Similarity			
		Straw.	*Pear*	*Pine*	*Banana*
Straw.	Punctured	-	-	-	-
Pear	Smooth	.6	-	-	-
Pine.	Thorny	.3	0	-	-
Banana	Smooth	.6	1	0	-

Feature: Shape					
Fruit	Shape	Similarity			
		Straw.	*Pear*	*Pine*	*Banana*
Straw.	Cone	-	-	-	-
Pear	Pear	0.2	-	-	-
Pine.	Ellipsoid	0.2	0.2	-	-
Banana	Bow	0	0	0	-

Unsupervised Learning

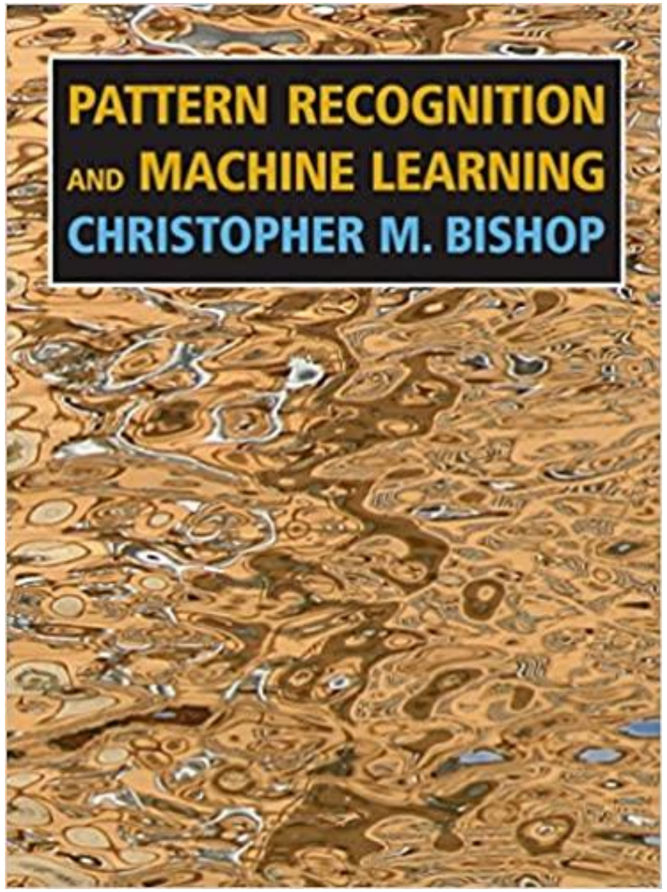
K-means

- Centroid: non-data point that indicates center of cluster as identified by K-means
- Operation:
  1. Deploy N centroids randomly (N proportional to number of expected classes)
  2. Assign randomly data points to classes
  3. Repeat iteratively
     1. Compute center of gravity of each class;
     2. Centroid is repositioned in that center of gravity
     3. Update boundary
  4. Stop when updates become negligible



Tools & Bibliography for Machine Learning

- Tools:
    - R
    - Python
        - Scikit
        - PyTorch
        - …
    - Online: RapidMiner
    - Specific: Image Processing: OpenCV
- Books:
    - Christopher Bishop, Pattern Recognition and Machine Learning.



Assignment

- Take the Titanic survivor data
  - https://raw.githubusercontent.com/guru99-edu/R-Programming/master/titanic_data.csv
- Instructions for pre-processing and Decision Trees
  - https://www.guru99.com/r-decision-trees.html
- Apply classification with the following:
  - Neural networks
    - R: http://www.di.fc.ul.pt/~jpn/r/neuralnets/neuralnets.html
  - SVM
    - R: http://www.di.fc.ul.pt/~jpn/r/svm/svm.html
  - K-nearest neighbours
    - R: http://www.di.fc.ul.pt/~jpn/r/clustering/clustering.html#k-nearest-neighbour
  - K-means
    - R: http://www.di.fc.ul.pt/~jpn/r/clustering/clustering.html#k-means

Thank you