Big data science Day 2

F. Legger - INFN Torino
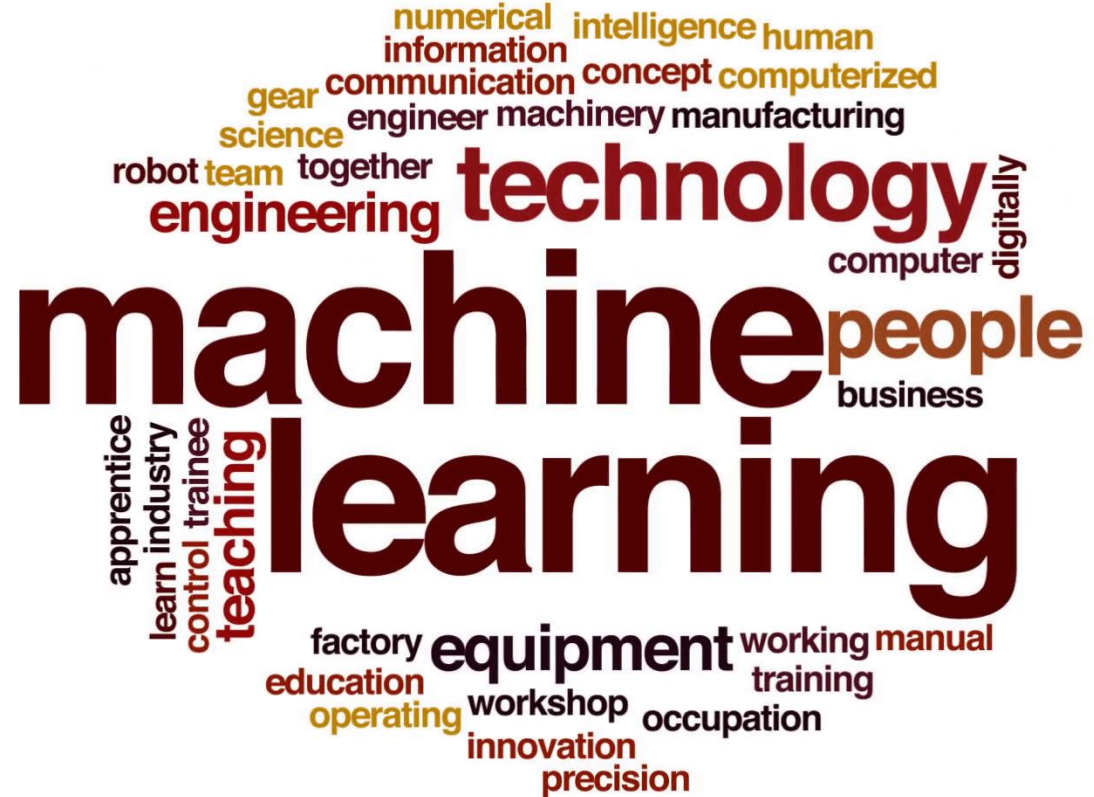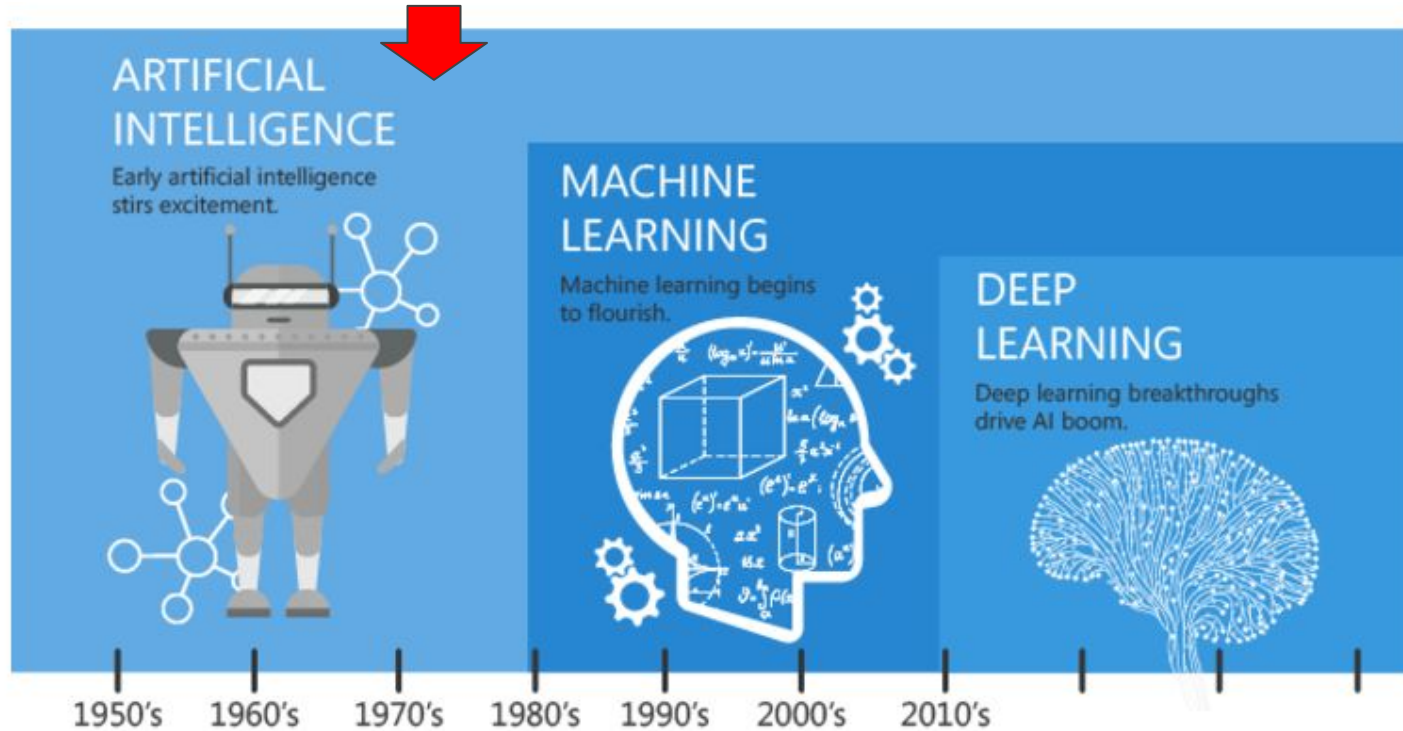https://github.com/leggerf/MLCourse-INFN-2021

Yesterday

- Big data
- Analytics

Today

- **Machine learning**

Our ultimate objective is to make programs that learn from their experience as effectively as humans do [John McCarthy, 1958]

ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.

MACHINE LEARNING
Machine learning begins to flourish.

DEEP LEARNING
Deep learning breakthroughs drive AI boom.

1950's   1960's   1970's   1980's   1990's   2000's   2010's

Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Machine Learning

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead   [Wikipedia]

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at task in T, as measured by P, improves with experience E   [Tom Mitchell, 1997]
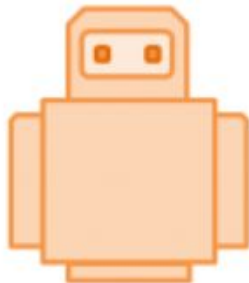
Machine Learning is the science of getting computers to act without being explicitly programmed   [Andrew Ng]

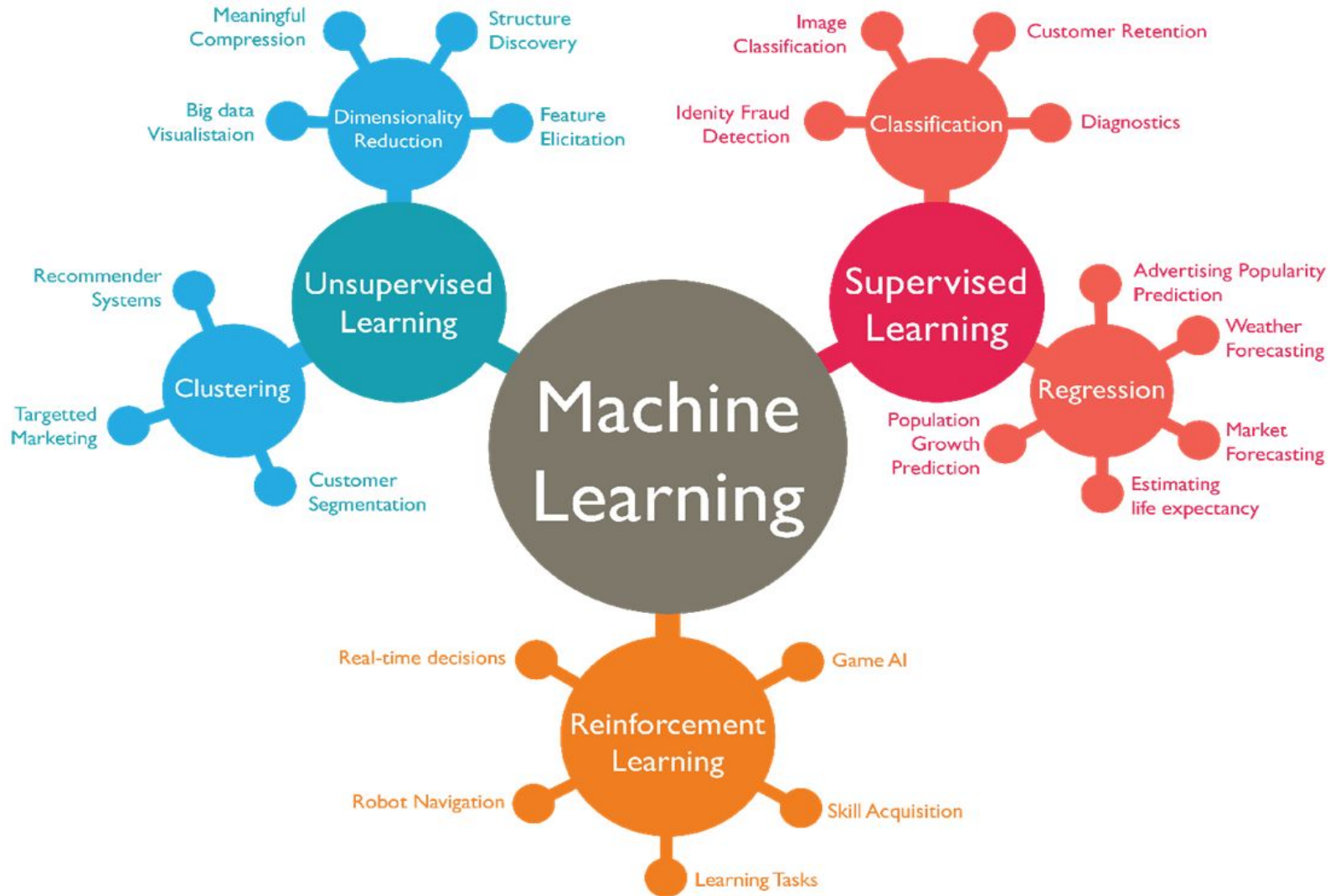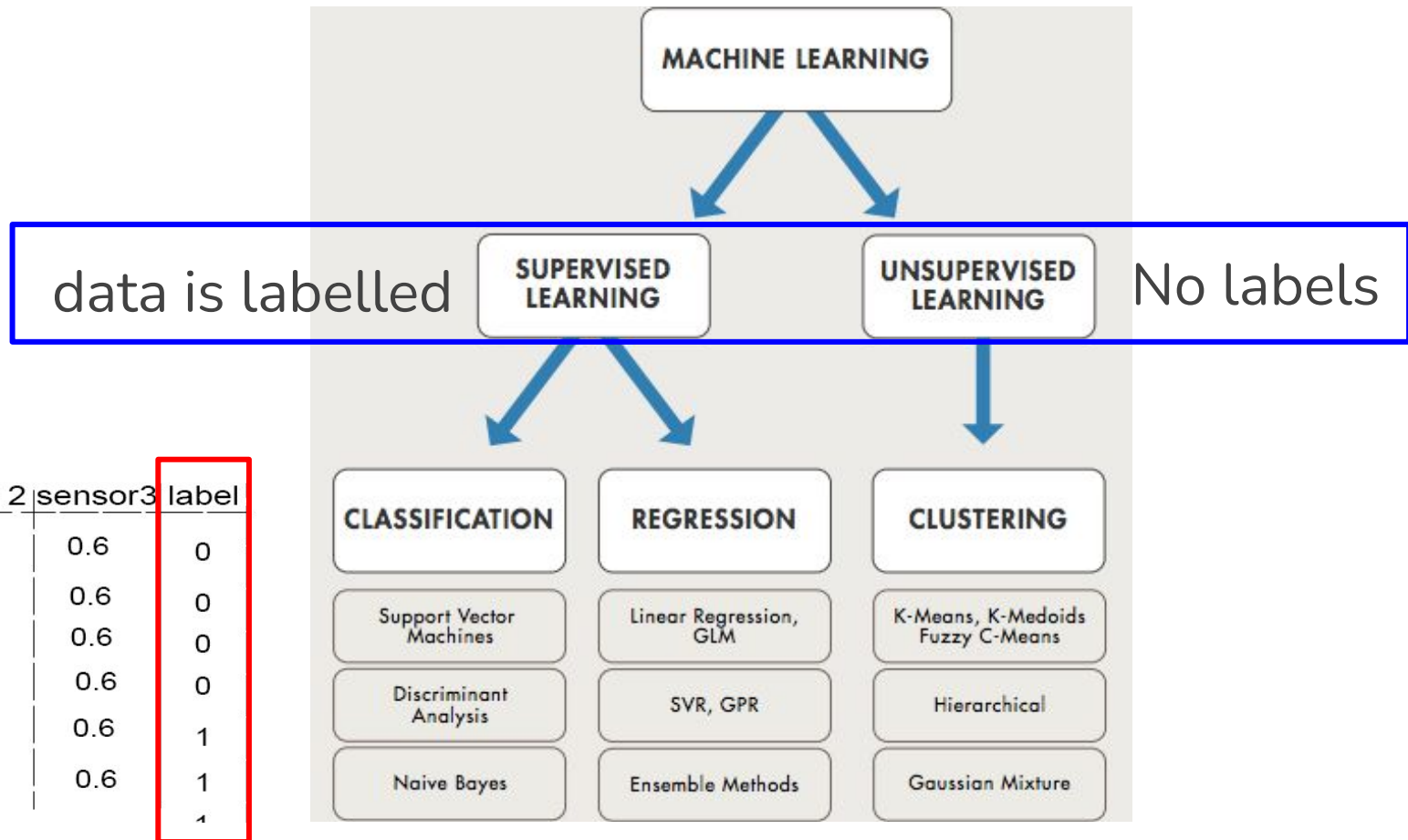# Machine Learning

Input Data

Information (+ Answers)

Output

Optimum Model

- Relationships
- Patterns
- Dependencies
- Hidden structures

Algorithms + Techniques

Machine Learning

Unsupervised Learning

Dimensionality Reduction
- Meaningful Compression
- Structure Discovery
- Big data Visualistaion
- Feature Elicitation

Clustering
- Recommender Systems
- Targetted Marketing
- Customer Segmentation

Supervised Learning

Classification
- Image Classification
- Customer Retention
- Idenity Fraud Detection
- Diagnostics

Regression
- Advertising Popularity Prediction
- Weather Forecasting
- Population Growth Prediction
- Market Forecasting
- Estimating life expectancy

Reinforcement Learning
- Real-time decisions
- Game AI
- Robot Navigation
- Skill Acquisition
- Learning Tasks

MACHINE LEARNING

data is labelled → SUPERVISED LEARNING   UNSUPERVISED LEARNING ← No labels

sensor1	sensor 2	sensor3	label
0.3	0.2	0.6	0
0.3	0.2	0.6	0
0.3	0.2	0.6	0
0.3	0.2	0.6	0
0.3	0.2	0.6	1
0.3	0.2	0.6	1
			1

CLASSIFICATION
- Support Vector Machines
- Discriminant Analysis
- Naive Bayes

REGRESSION
- Linear Regression, GLM
- SVR, GPR
- Ensemble Methods

CLUSTERING
- K-Means, K-Medoids Fuzzy C-Means
- Hierarchical
- Gaussian Mixture

Discrete labels   Continuous labels

7

Reinforcement learning

- getting an agent to act in the world so as to maximize its rewards
- sparse and time delayed labels (**rewards**)

Agent State S_t^a

Observation O_t

Reward R_t

Action A_t

Environmental State S_t^e
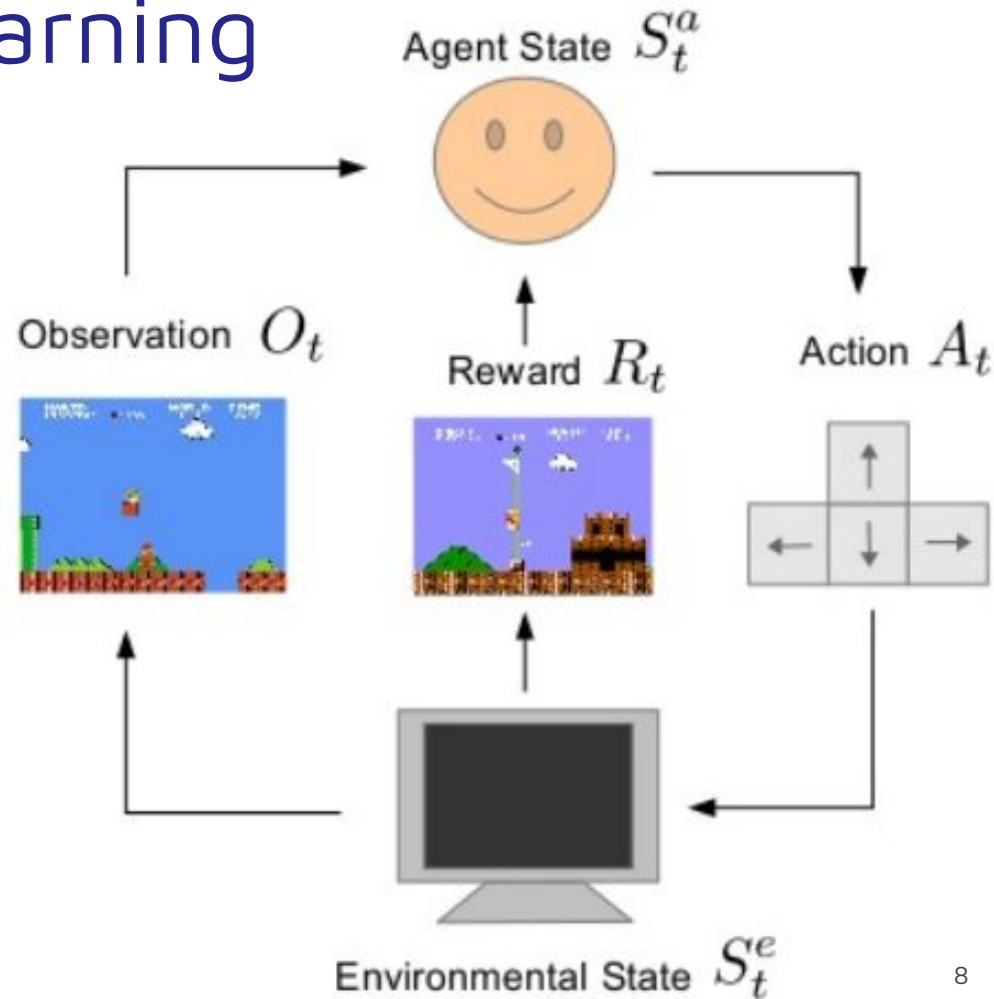
Classification

- Businesses who target customers: good vs bad, stay or leave
- **Signal vs background**
- ....

hands-on

Supervised, discrete labels
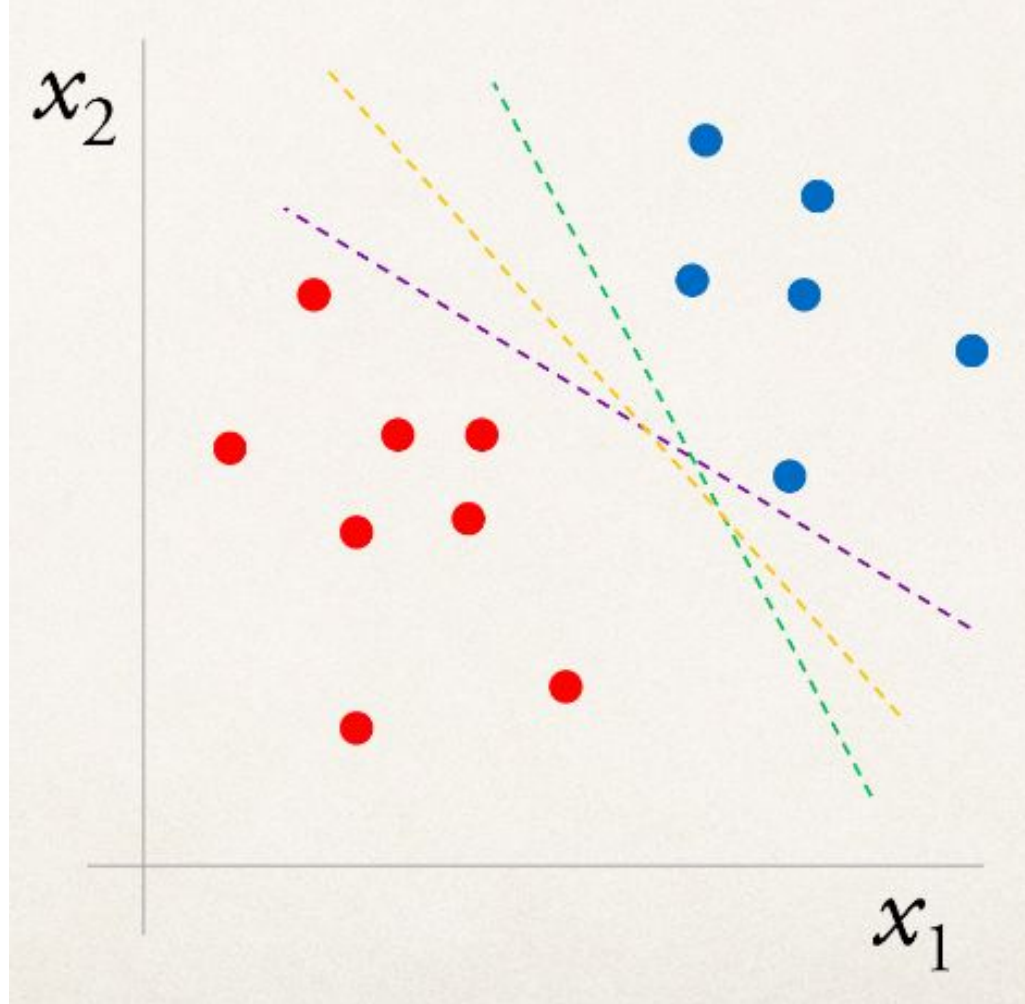
Regression

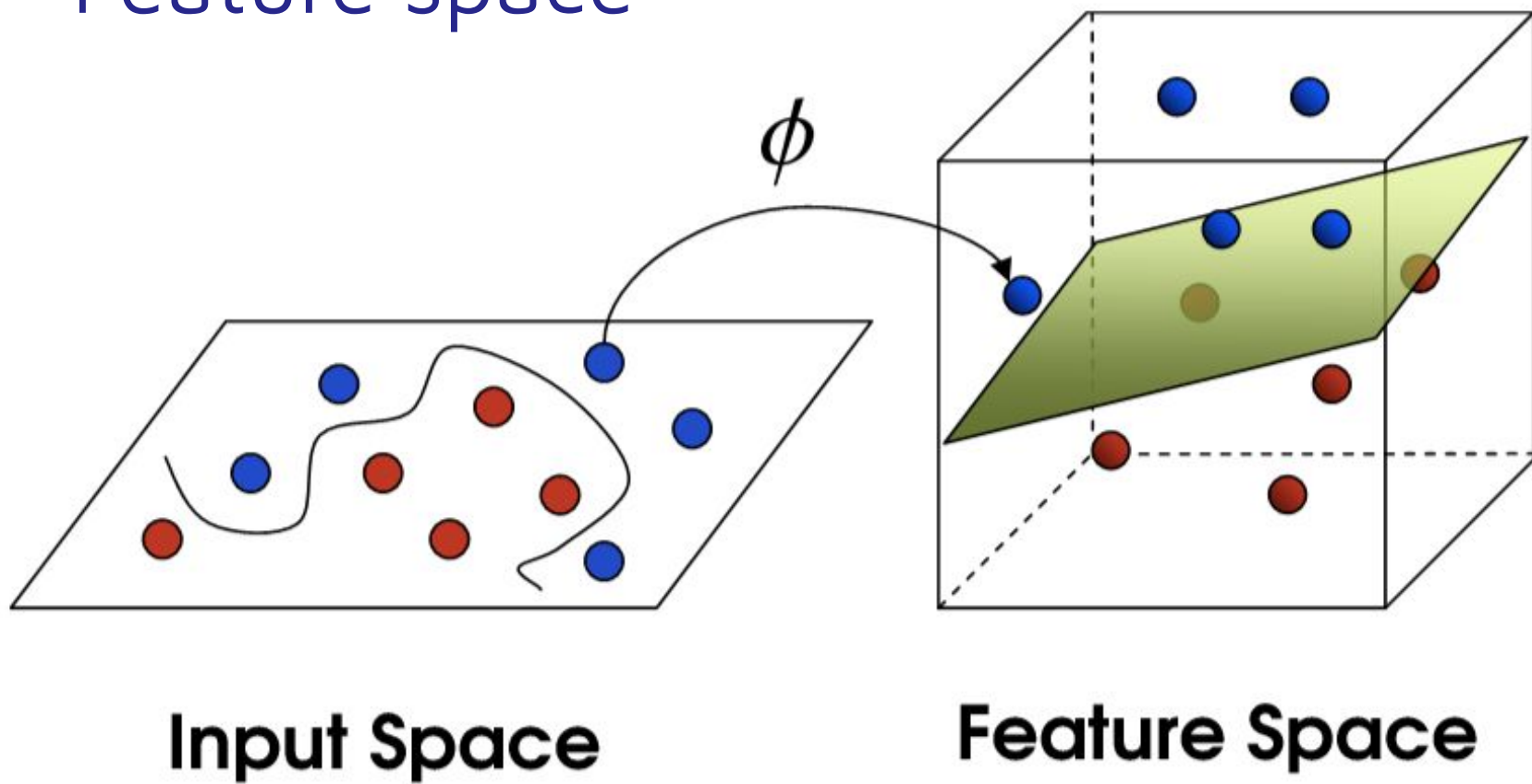- Businesses who predict customer behavior: e.g. house prices, ...
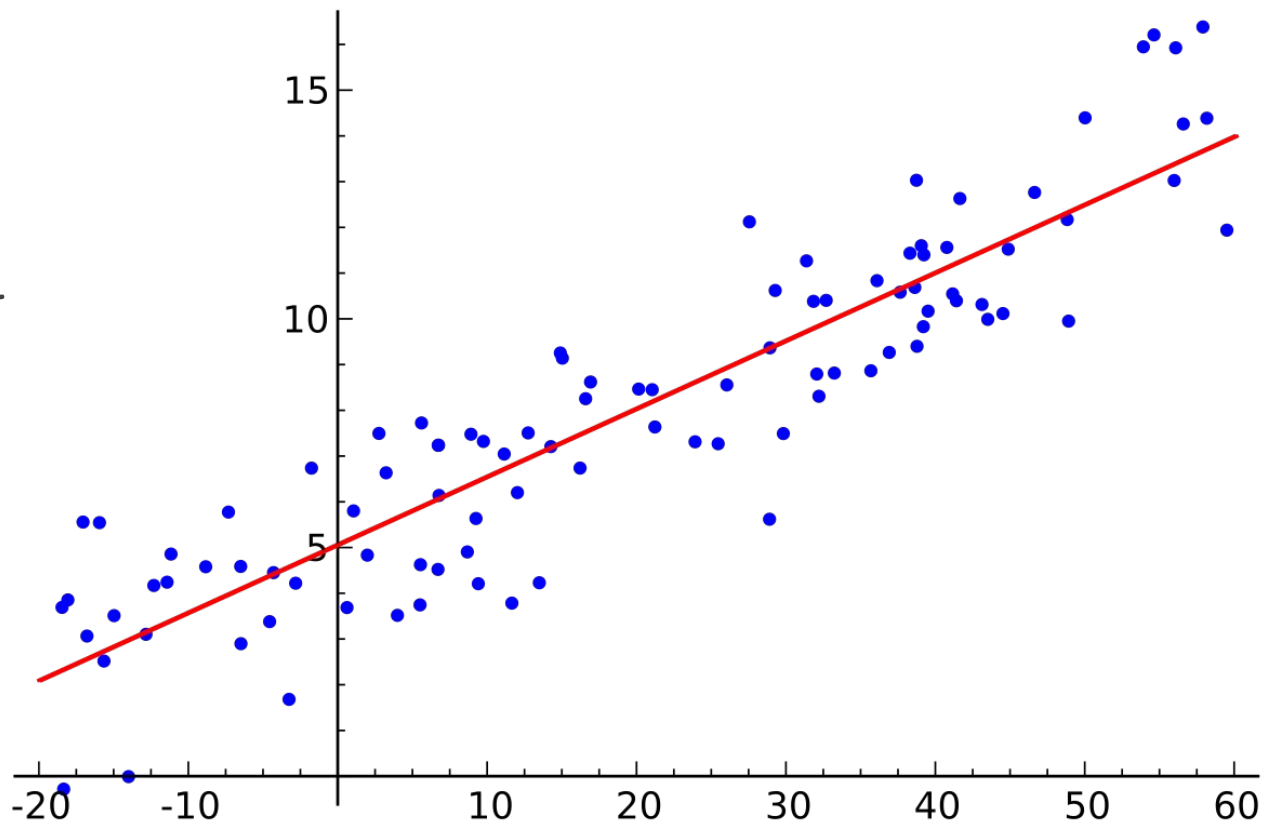
Supervised, continuous labels

Clustering

- Businesses who identify customer categories
- Light vs heavy flavour jets
- ....

Unsupervised

Feature space

ϕ

Input Space

Feature Space

Feature engineering

Training and test set
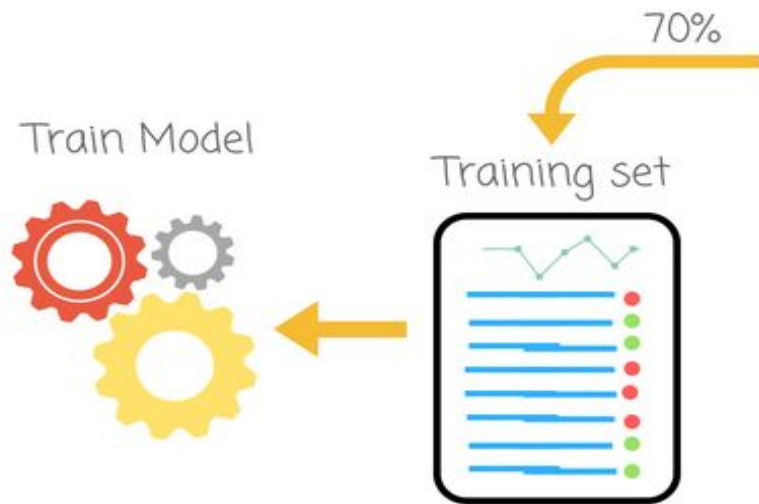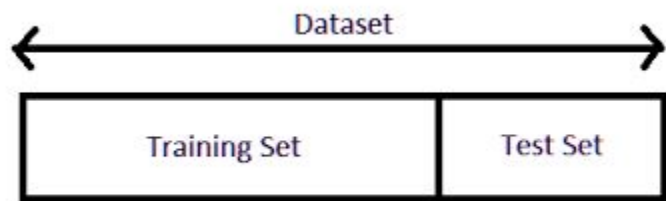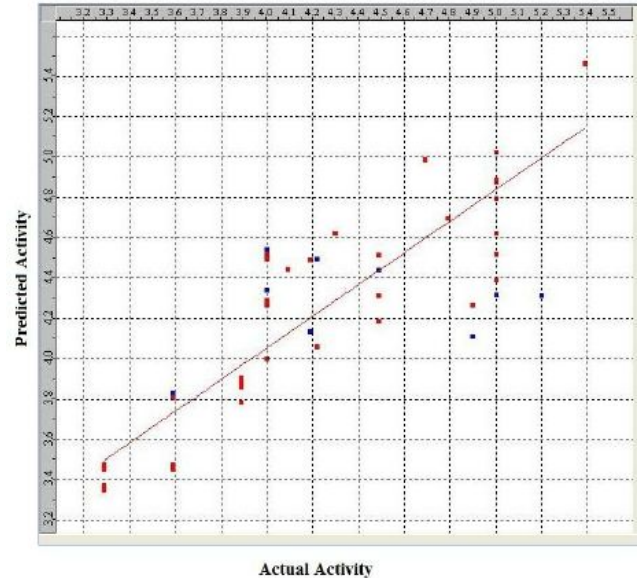
Example:supervised classification

Ingredients

- **Inputs: X**, e.g. timestamp, price, color, size, etc.
- **Features: *X***, transformed inputs
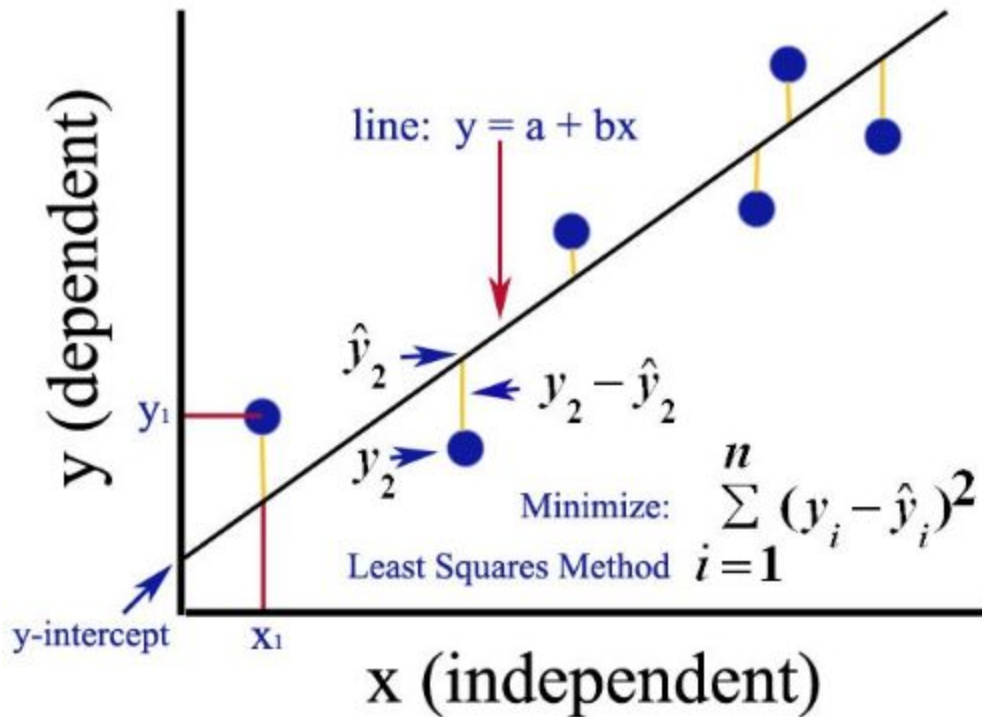- **Labels: y**
- **Training** and **test** datasets

Recipe

- **Weights: W** (matrix) parameters to be found by the model
- **Activation function: φ** (step function, e.g. sigmoid)
- **Predictions: z = φ(W^T *X*)** yields (0,1)
- **Cost function == loss function == prediction error: J(W)**, e.g. $\sum (y_i - z_i)^2 / 2$

- Aim: find weights W that <u>minimize cost function</u> & give best separation

Another example, linear regression

- Inputs (features): x_i
- Labels: y_i
- Model: $y = a + bx$
- Weight+bias (parameters to be found): a, b
- Cost function: **Mean Square Error (MSE)**
- No **activation function:** problem is linear

line: $y = a + bx$

$\hat{y}_2$

$y_2 - \hat{y}_2$

y_1

y_2

y (dependent)

y-intercept

x_1

x (independent)

Minimize: $\sum\limits_{i=1}^{n}(y_i - \hat{y}_i)^2$

Least Squares Method

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

16

Neural networks: supervised classification

- Basic unit: **Neuron**
- A neuron takes inputs, does some math with them, and produces **one** output
- **Feedforward**: process of passing inputs forward to get an output

Hands-on today!
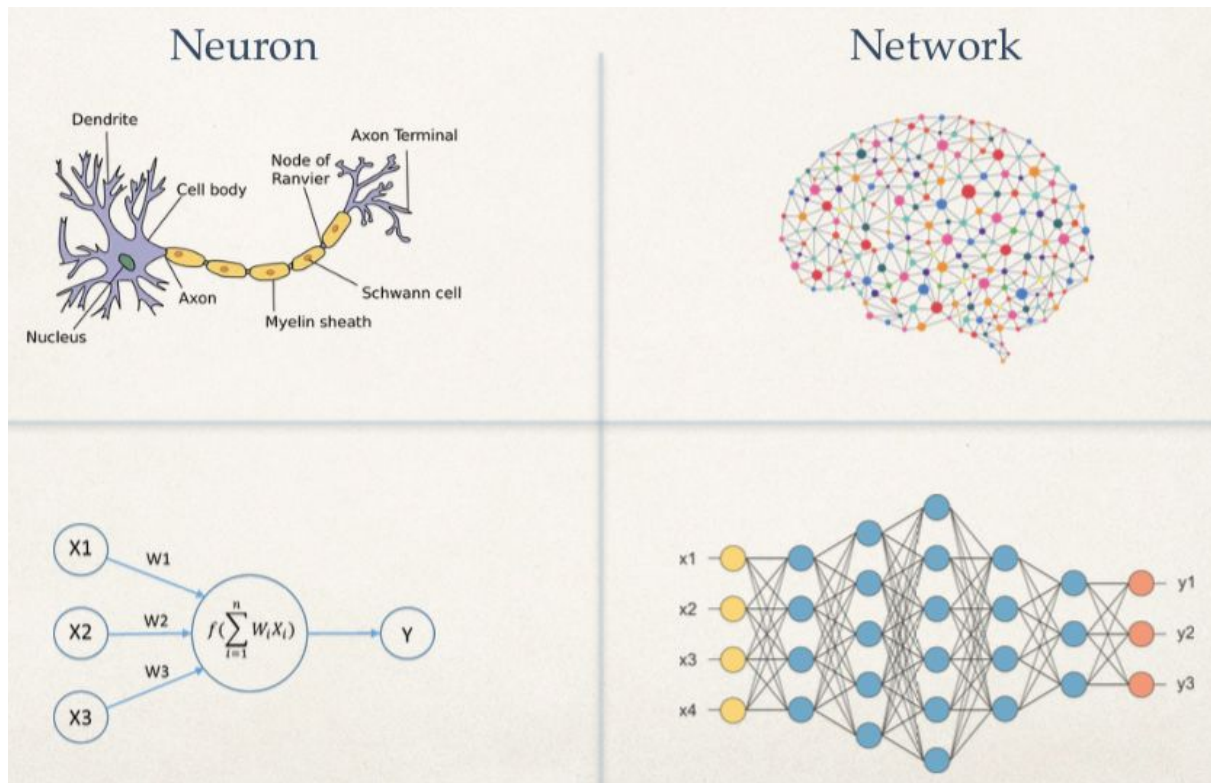
Neuron
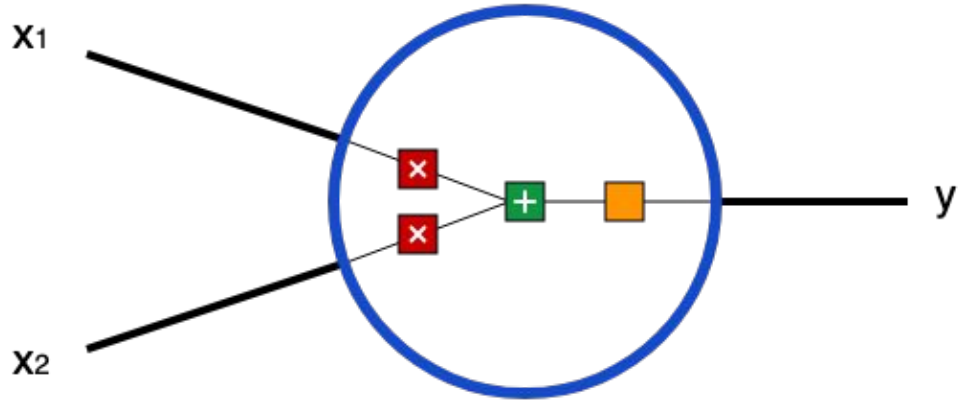
- each **input x** is multiplied by a **weight w**

$$x_1 \rightarrow x_1 * w_1$$

$$x_2 \rightarrow x_2 * w_2$$

- all the weighted inputs are added together with a **bias b**

$$(x_1 * w_1) + (x_2 * w_2) + b$$

- the sum is passed through an **activation function f**

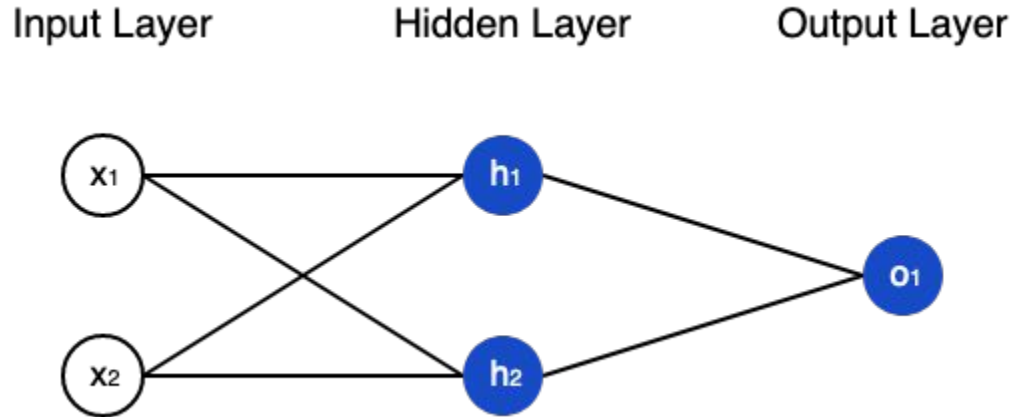$$y = f(x_1 * w_1 + x_2 * w_2 + b)$$

Activation function

- Turns unbounded output into a known range/shape
- For example, **sigmoid** function only outputs numbers in the range (0, 1)
  - big negative numbers become ~0
  - big positive numbers become ~1.

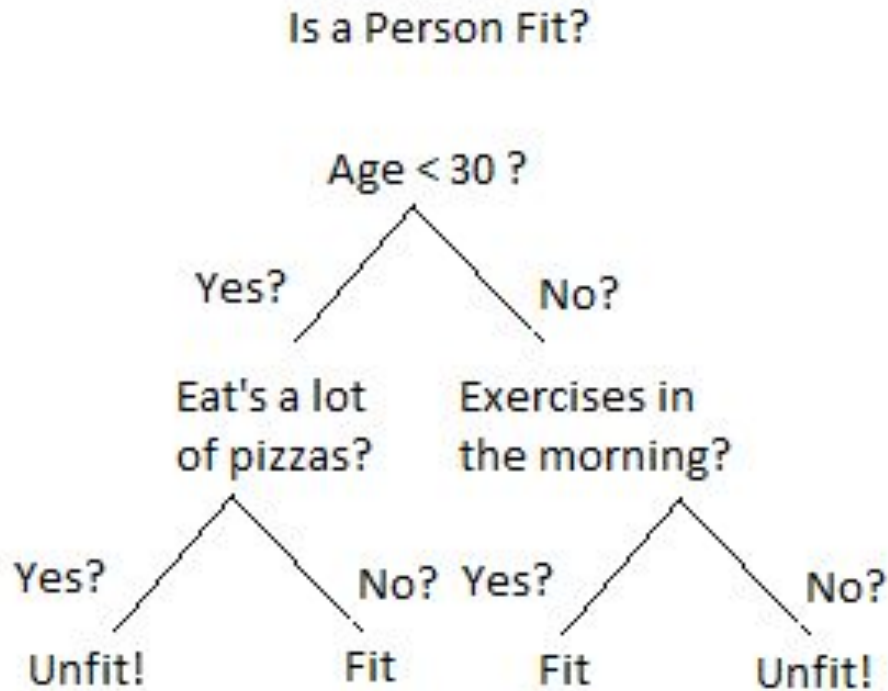$$S(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}.$$
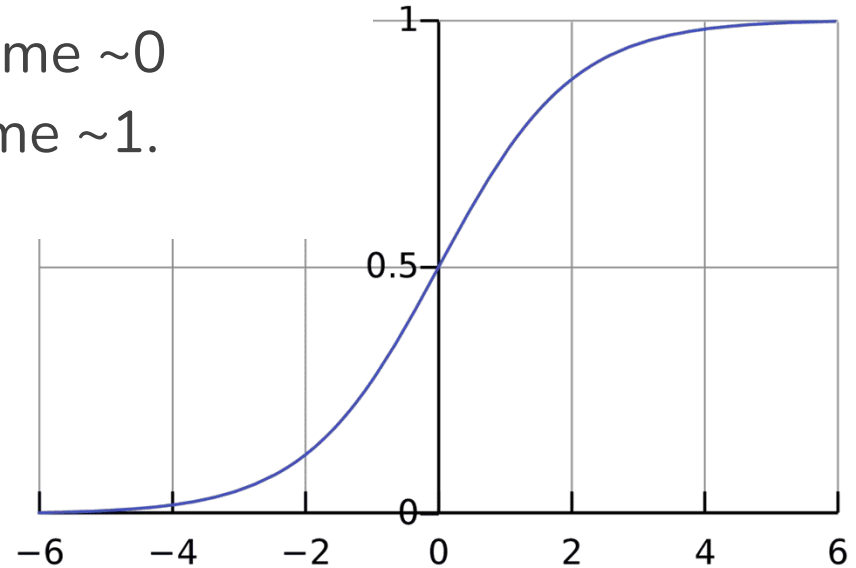
Neural network

- Combining more neurons
- A **hidden layer** is any layer between the input (first) layer and output (last) layer
  - There can be multiple hidden layers
- This network has:
  - one **input** layer with 2 inputs
  - one **hidden** layer with 2 neurons
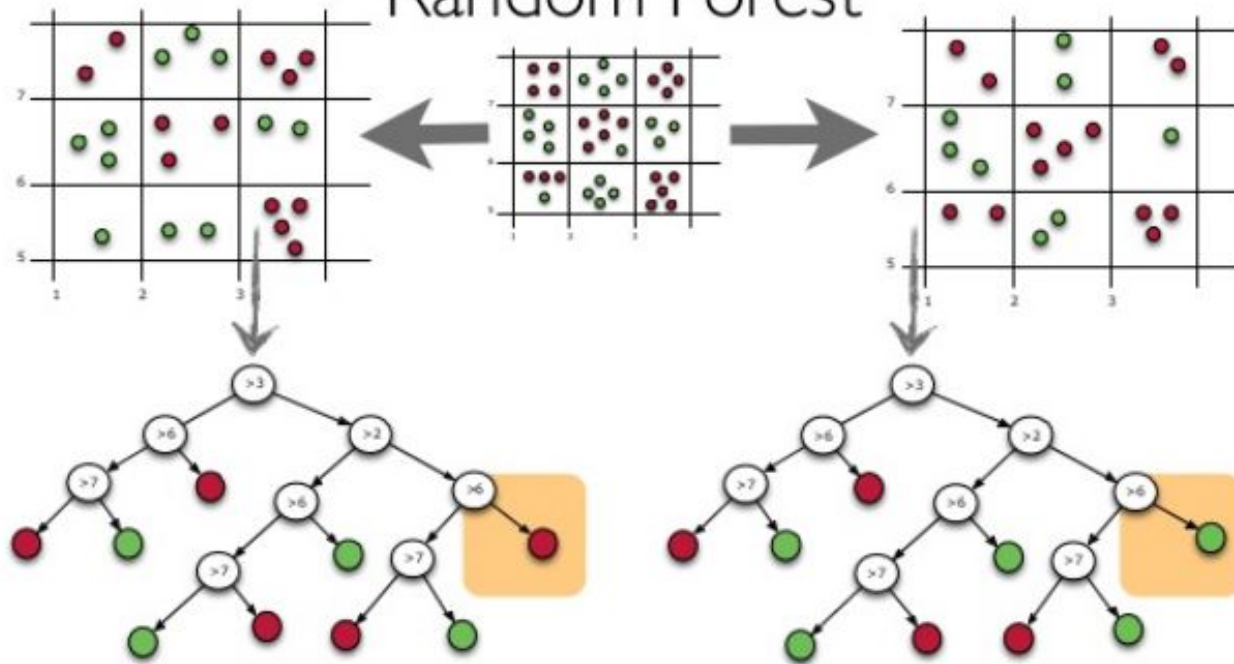 - one **output** layer with 1 neuron

Input Layer   Hidden Layer   Output Layer

Decision trees

Typically used in combinations (Random forest, Gradient Tree Boosting)

Is a Person Fit?

Age < 30 ?

Yes? / No?

Eat's a lot of pizzas? / Exercises in the morning?
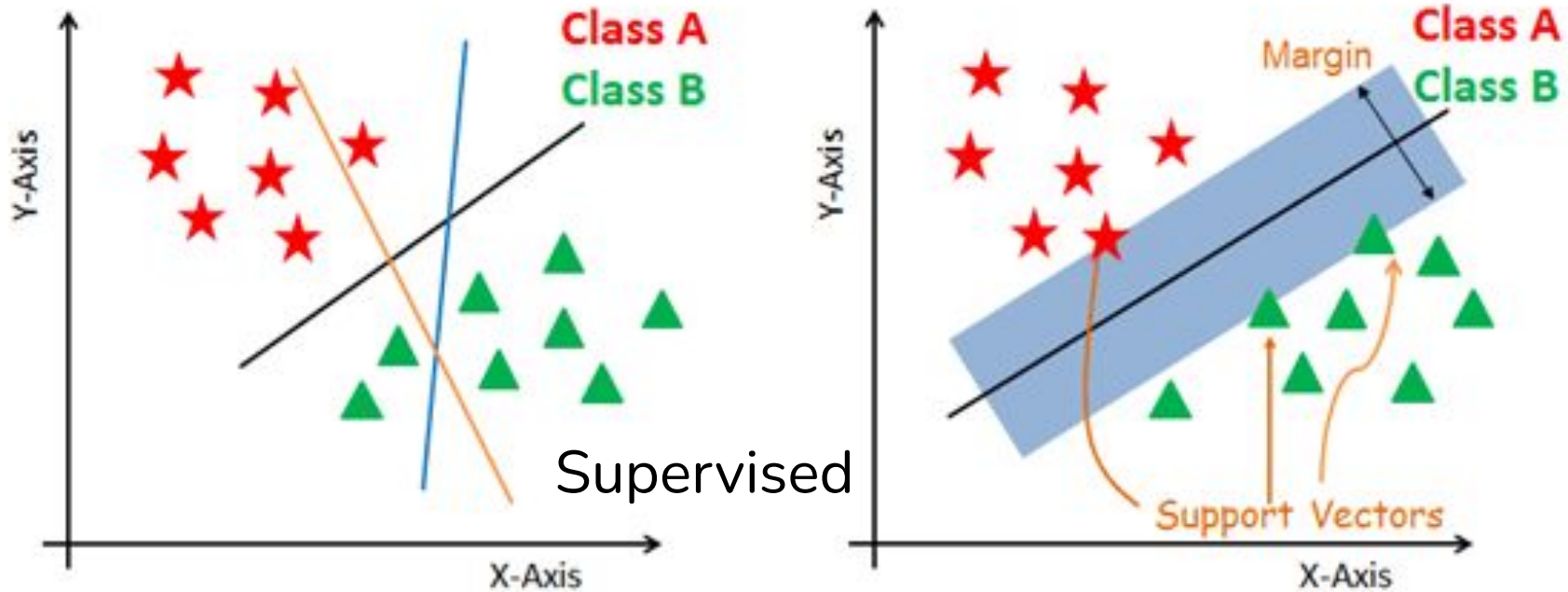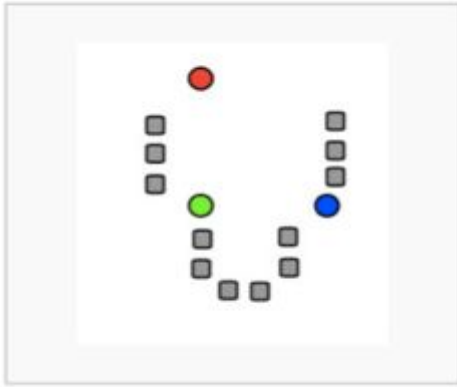
Yes? / No? / Yes? / No?

Unfit! / Fit / Fit / Unfit!

Random Forest

• Each tree sees part of the training sets and captures part of the information it contains
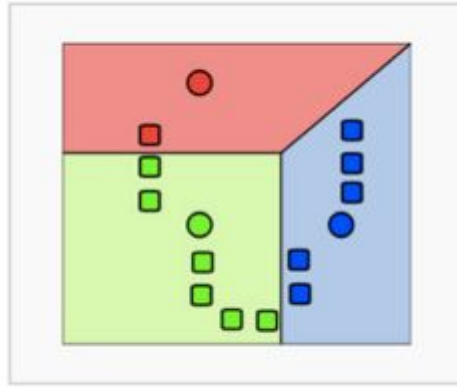
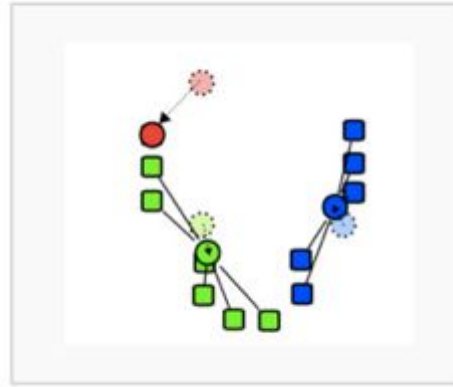# Support vector machines (SVG)

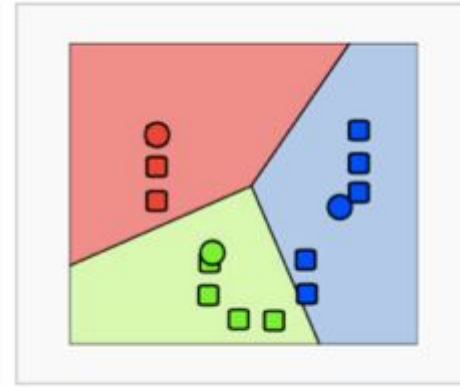Supervised

K-means clustering

1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).

2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

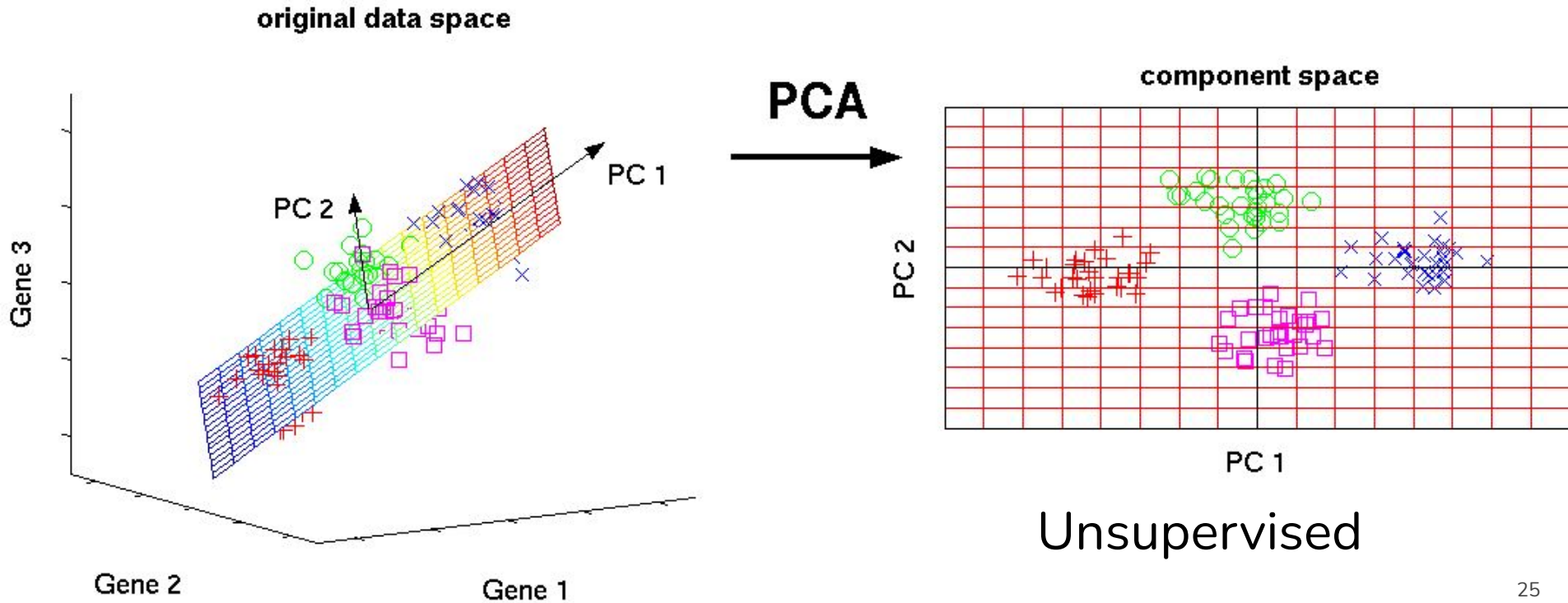3. The centroid of each of the *k* clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.
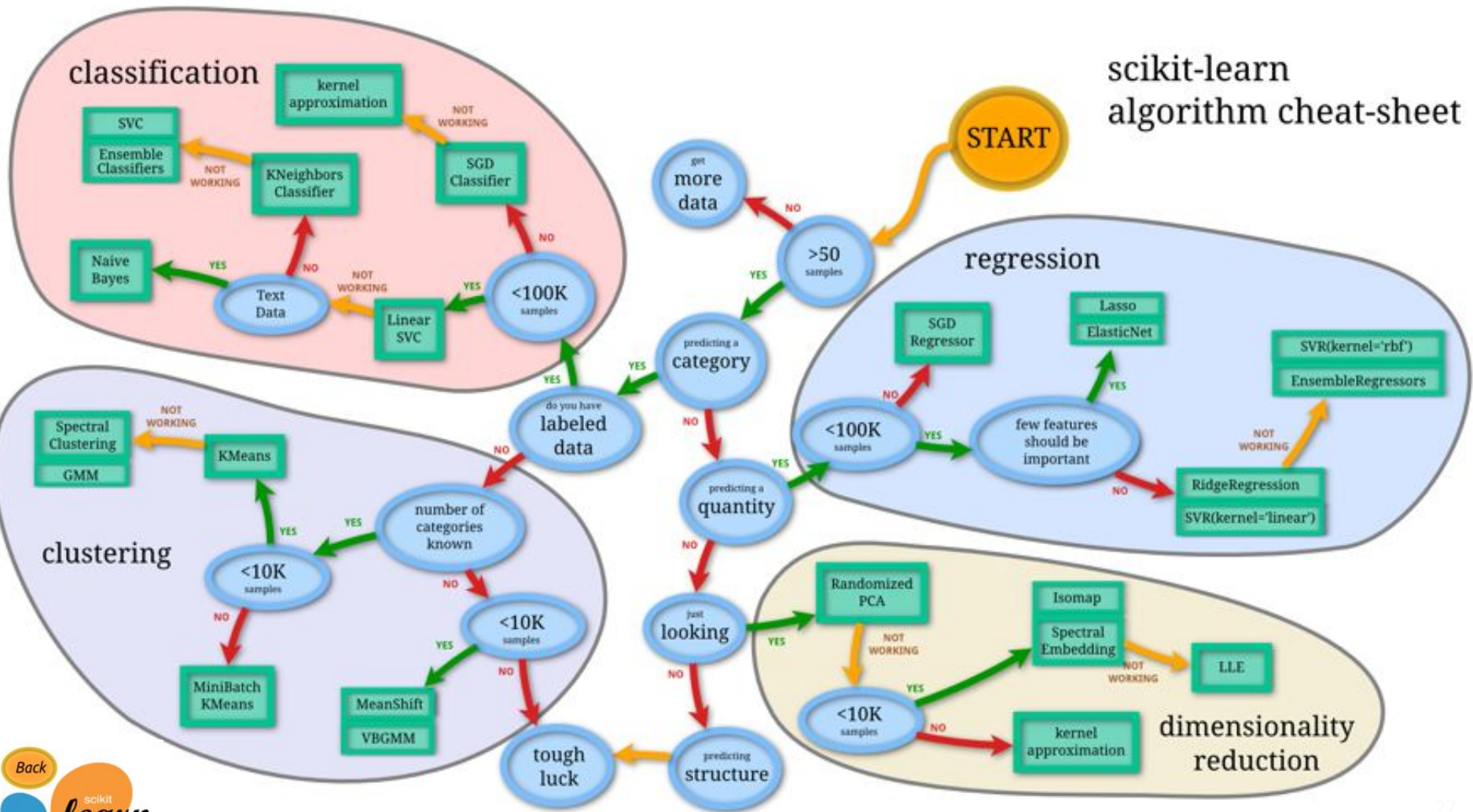
Unsupervised

https://www.saedsayad.com/clustering_kmeans.htm

Principal Component Analysis (PCA)

● Used for dimensionality reduction

original data space → **PCA** → component space

Unsupervised

Ensembles

- **Bagging**
  - building multiple models (typically of the *same* type) from different subsamples of the training dataset
- **Boosting**
  - building multiple models (typically of the *same* type) each of which learns to fix the predictions errors of a prior model in the chain
- **Stacking**
  - building multiple models (typically of *different* types) and supervisor model that learns how to best combine the predictions of the primary model
- **Weighting|Blending**
  - combine multiple models into single prediction using different weight functions

scikit-learn algorithm cheat-sheet

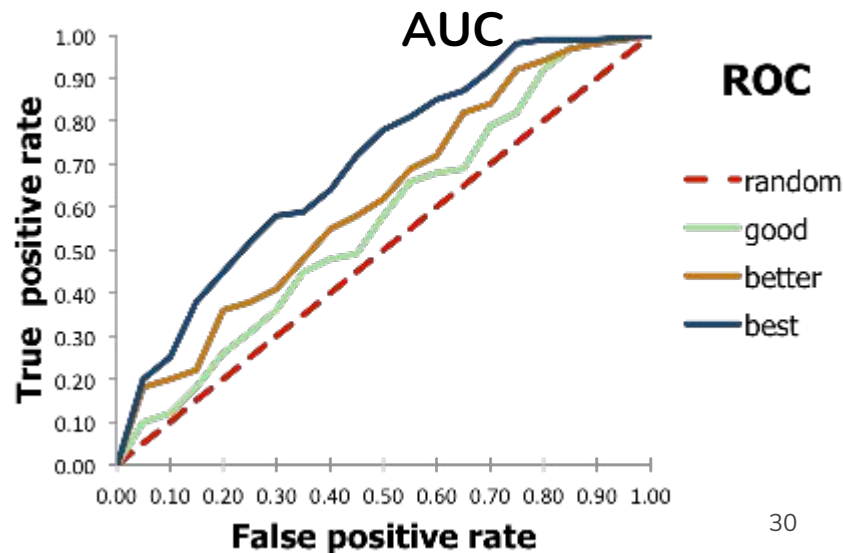| TYPE	NAME	DESCRIPTION	ADVANTAGES	DISADVANTAGES
Linear	Linear regression	The "best fit" line through all data points. Predictions are numerical.	Easy to understand — you clearly see what the biggest drivers of the model are.	X Sometimes too simple to capture complex relationships between variables. X Tendency for the model to "overfit".
Linear	Logistic regression	The adaptation of **linear regression** to problems of classification (e.g., yes/no questions, groups, etc.)	Also easy to understand.	X Sometimes too simple to capture complex relationships between variables. X Tendency for the model to "overfit".
Tree-based	Decision tree	A graph that uses a **branching method** to match all possible outcomes of a decision.	Easy to understand and implement.	X Not often used on its own for prediction because it's also often too simple and not powerful enough for complex data.
Tree-based	Random Forest	Takes the average of many decision trees, each of which is made with a sample of the data. Each tree is weaker than a full decision tree, but **by combining them we get better overall performance.**	A sort of "wisdom of the crowd". Tends to result in very high quality models. Fast to train.	X Can be slow to output predictions relative to other algorithms. X Not easy to understand predictions.
Tree-based	Gradient Boosting	Uses even weaker decision trees, that are increasingly **focused on "hard" examples.**	High-performing.	X A small change in the feature set or training set can create radical changes in the model. X Not easy to understand predictions.
Neural networks	Neural networks	Mimics the behavior of the brain. Neural networks are interconnected neurons that pass messages to each other. Deep learning uses several **layers of neural networks put one after the other.**	Can handle extremely complex tasks - no other algorithm comes close in image recognition.	X Very, very slow to train, because they have so many layers. Require a lot of power. X Almost impossible to understand predictions.

All models are wrong, but some are useful (George Box)
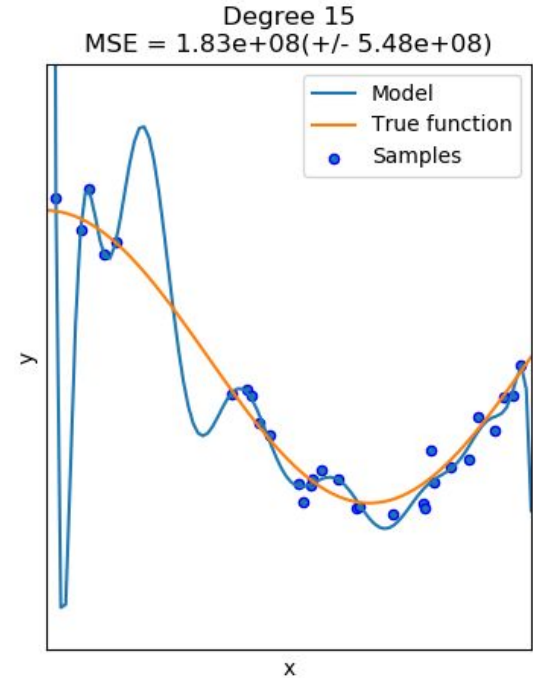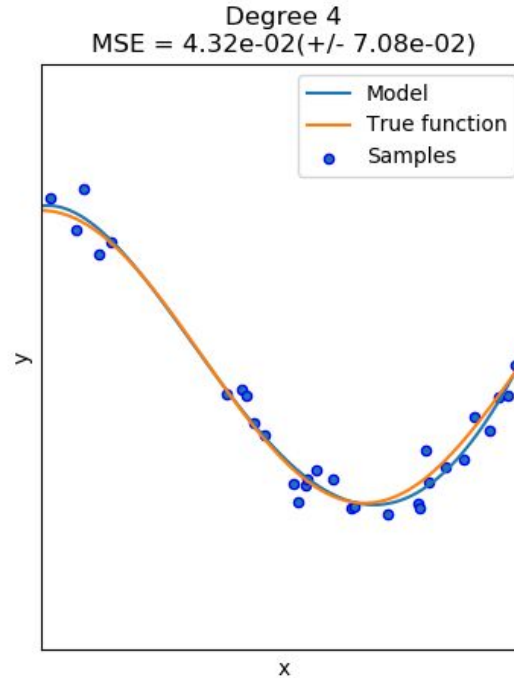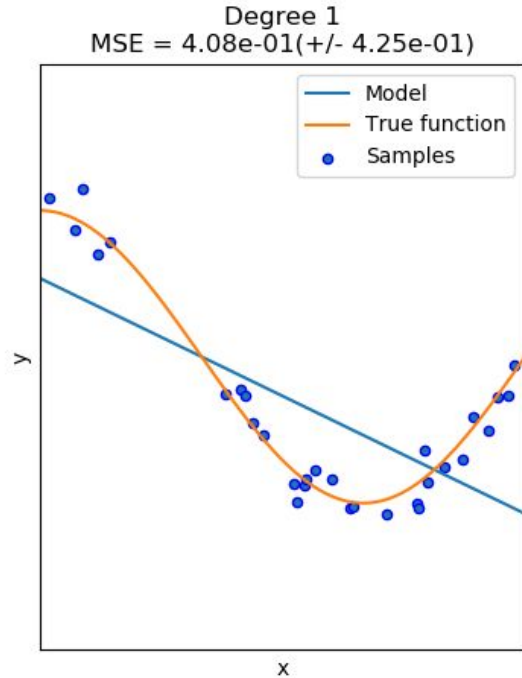
Classification metrics

- **ROC**: Receiver Operating Characteristics
- **AUC**: Area under the curve
- **TPR**: True positive rate
- **FPR**: False positive rate
- **TNR/FNR**: True/False negative rate

Confusion matrix

	True class	
	p	**n**
Y	True Positives	False Positives
N	False Negatives	True Negatives
Column totals:	P	N

Hypothesized class

$$\text{fp rate} = \frac{FP}{N} \qquad \text{tp rate} = \frac{TP}{P}$$

$$\text{precision} = \frac{TP}{TP+FP} \qquad \text{recall} = \frac{TP}{P}$$

$$\text{accuracy} = \frac{TP+TN}{P+N}$$

$$\text{F-measure} = \frac{2}{1/\text{precision}+1/\text{recall}}$$

AUC

ROC

- - - random
- good
- better
- best

True positive rate / False positive rate

Overfitting / underfitting

Degree 1
MSE = 4.08e-01(+/- 4.25e-01)

Degree 4
MSE = 4.32e-02(+/- 7.08e-02)

Degree 15
MSE = 1.83e+08(+/- 5.48e+08)

Underfitting
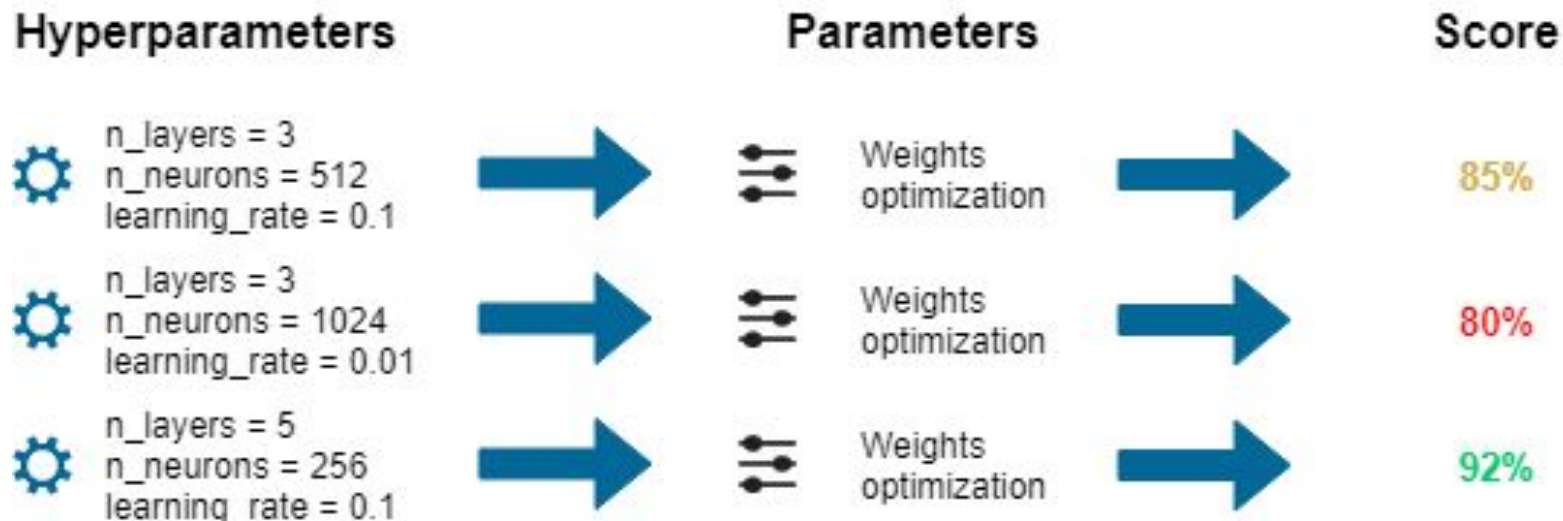Model doesn't have enough parameters to describe data

Overfitting
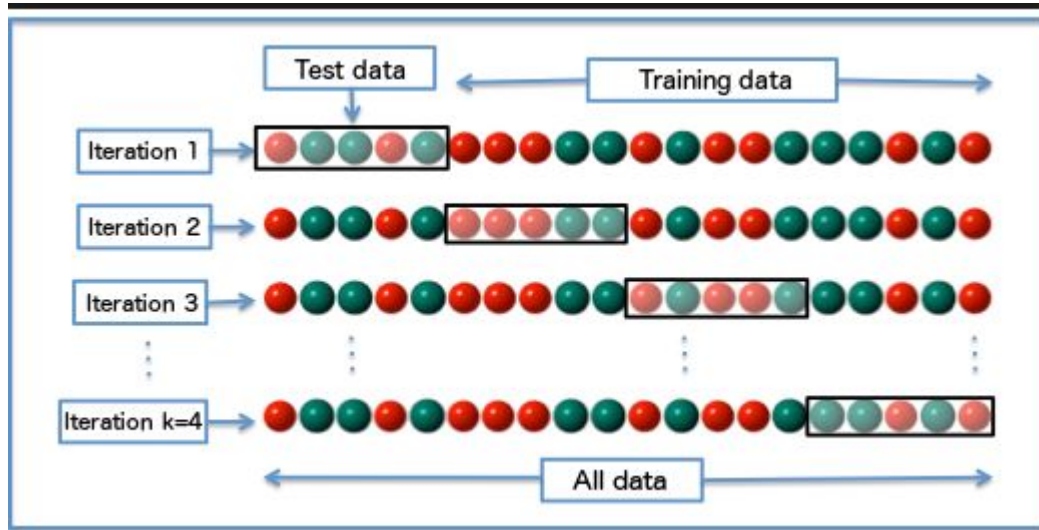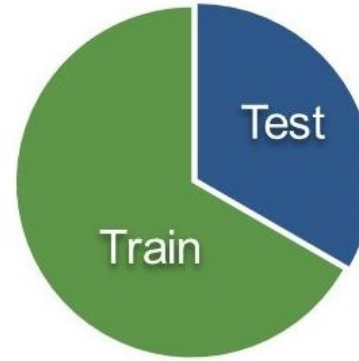Model has too many parameters

Hyperparameters vs parameters

- Model **parameters** are learned during training when we optimize a loss function
- **Hyperparameters** are not model parameters and they cannot be directly trained from the data

Hyperparameters		Parameters		Score
n_layers = 3 n_neurons = 512 learning_rate = 0.1	→	Weights optimization	→	85%
n_layers = 3 n_neurons = 1024 learning_rate = 0.01	→	Weights optimization	→	80%
n_layers = 5 n_neurons = 256 learning rate = 0.1	→	Weights optimization	→	92%

Cross-validation

- Train/test split
- K-folds cross validation

Modeling Algorithm

Tune

hyperparameters (tuning options)

- Polynomial order, penalty parameter, ...
- Network configuration, solver options, ...
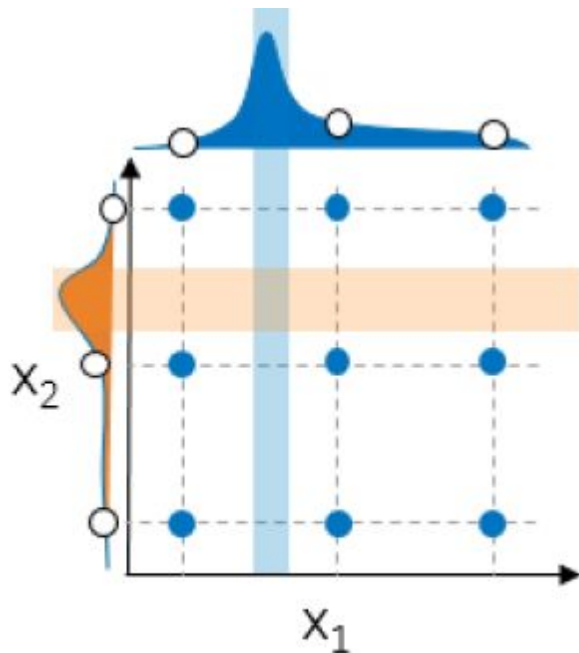- Max tree depth, splitting criterion, ...

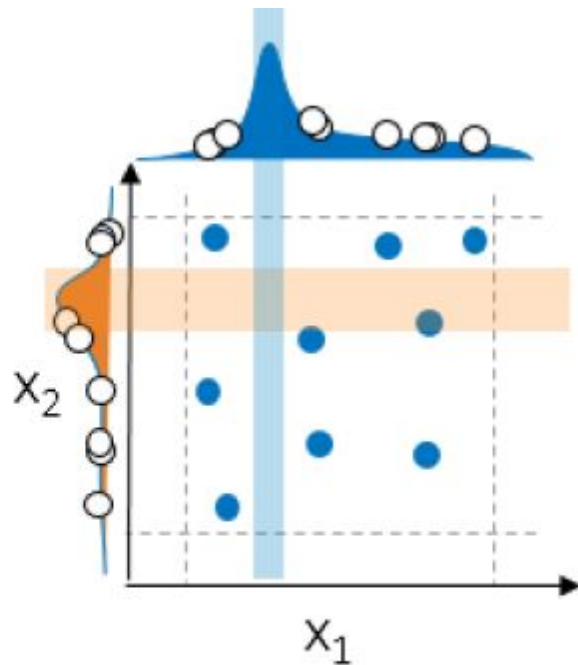## Model

Train

model parameters

- Regression coefficients
- Neural net weights
- Tree splitting rules

...

Hyperparameter tuning

(a) Standard Grid Search

(b) Random Search