TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Machine Learning

Melissa Sanabria, TU Dresden
melissa.sanabria@tu-dresden.de
AG Poetsch

Machine Learning

Why Artificial intelligence is so difficult to grasp?

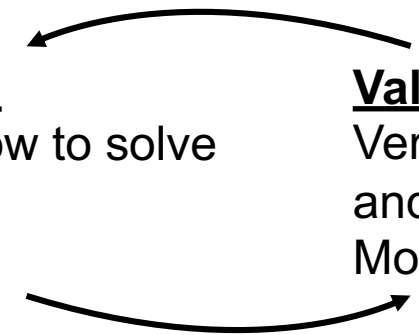
Frequently, when a technique reaches mainstream use, it is no longer considered as artificial intelligence; this phenomenon is described as the **AI effect**:
“AI is whatever hasn't been done yet.” (*Larry Tesler*)

e.g. GPS, Alpha Go, Face detection in our phones

AI is continuously evolving and so very difficult to grasp.

Machine Learning

Task
Is it a healthy sample?
Where are the cells in the image?
Is this gene expressed?
….

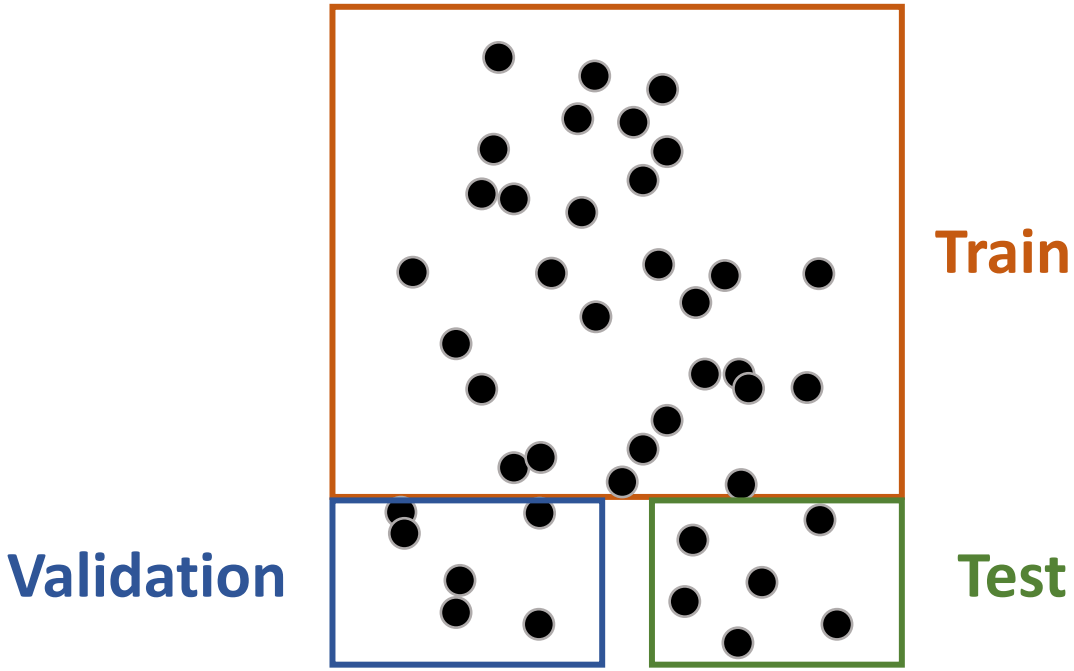


Training
Learn how to solve the task

Validation
Verify if you are actually **learning** and **not** just **remembering**.
Modify parameters

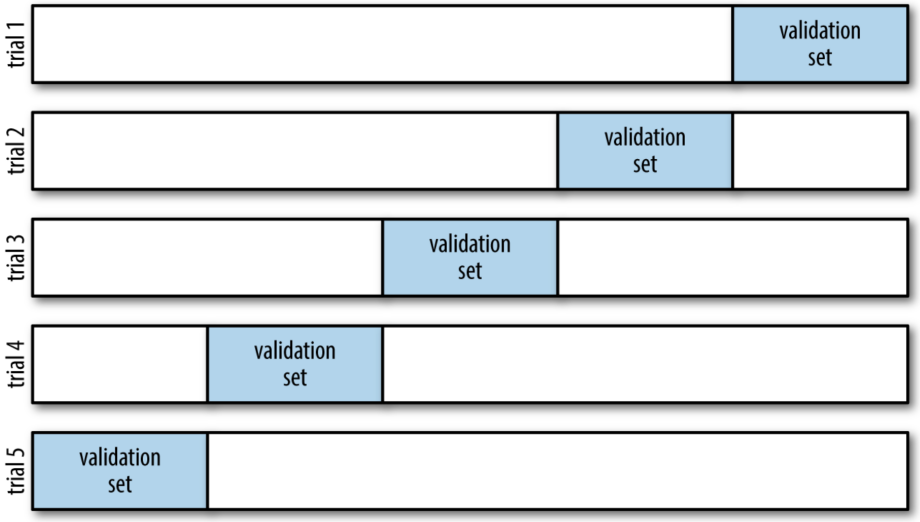
Test
Unseen data
Real-life score

Model Validation: Holdout sets



All the sets are independent of each other and **do not overlap**!

Model Validation: Cross validation



https://scikit-learn.org/stable/modules/cross_validation.html

Data Leakage

Is the scenario where the Machine Learning model is **already aware** of some part of test data during training.

Feature Leakage
A prediction target is inadvertently used in the training process

Training example Leakage
When you are not careful to distinguish training data from testing data.

Data Leakage

Feature Leakage

JOURNAL OF MEDICAL INTERNET RESEARCH Ye et al

Original Paper

Prediction of Incident Hypertension Within the Next Year: Prospective Study Using Statewide Electronic Health Records and Machine Learning

Of the six most important variables, five were: lisinopril, hydrochlorothiazide, enalapril maleate, amlodipine besylate, and losartan potassium. All of these are popular **antihypertensive drugs**.

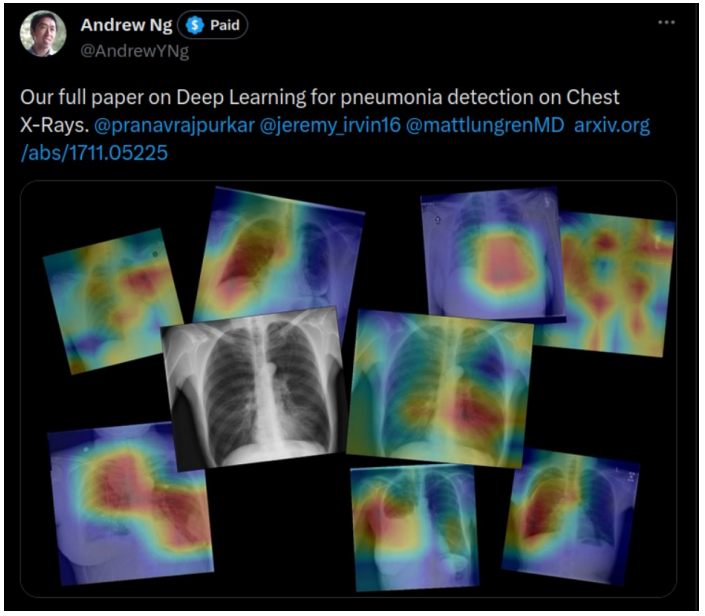
Just one variable (**the use of a hypertension drug**) is sufficient for physicians to infer the presence of hypertension.

Data Leakage

Training Example Leakage

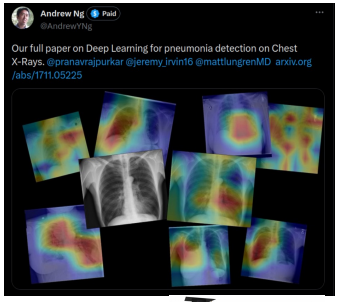
3.1. Training

We use the ChestX-ray14 dataset released by Wang et al. (2017) which contains 112,120 frontal-view X-ray images of 30,805 unique patients. Wang et al. (2017) annotate each image with up to 14 different thoracic pathology labels using automatic extraction methods on radiology reports. We label images that have pneumonia as one of the annotated pathologies as positive examples and label all other images as negative examples for the pneumonia detection task. We randomly split the entire dataset into 80% training, and 20% validation.



Data Leakage

Training Example Leakage



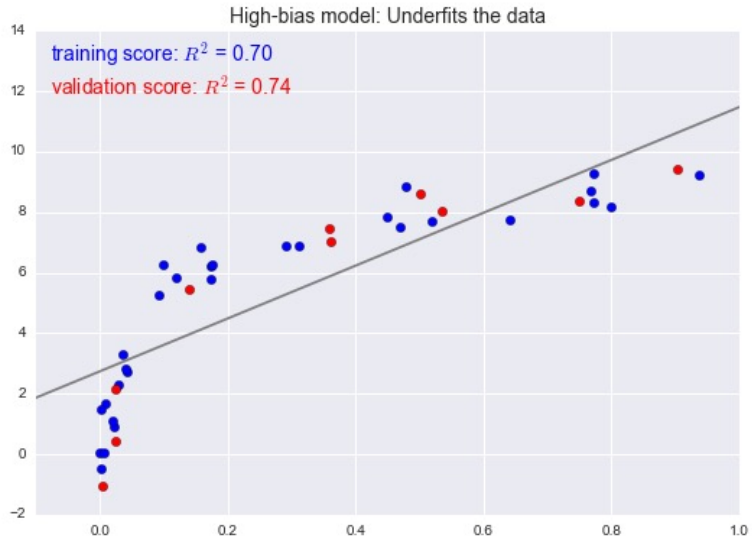
3.1. Training

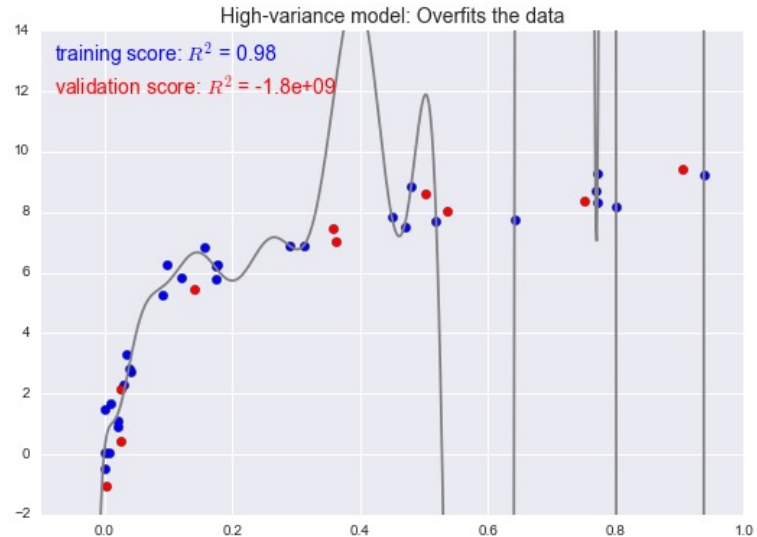
We use the ChestX-ray14 dataset released by Wang et al. (2017) which contains 112,120 frontal-view X-ray images of 30,805 unique patients. Wang et al. (2017) annotate each image with up to 14 different thoracic pathology labels using automatic extraction methods on radiology reports. We label images that have pneumonia as one of the annotated pathologies as positive examples and label all other images as negative examples for the pneumonia detection task. We randomly split the entire dataset into 80% training, and 20% validation.

3.1. Training

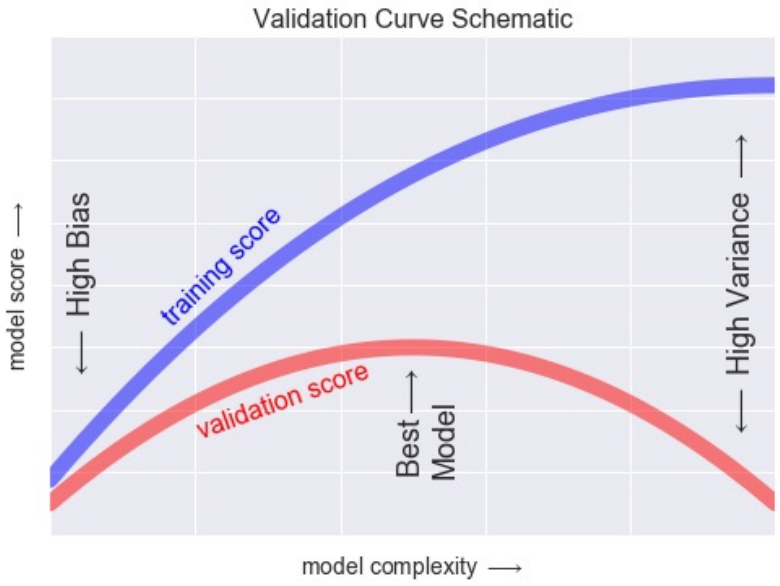
We use the ChestX-ray14 dataset released by Wang et al. (2017) which contains 112,120 frontal-view X-ray images of 30,805 unique patients. Wang et al. (2017) annotate each image with up to 14 different thoracic pathology labels using automatic extraction methods on radiology reports. We label images that have pneumonia as one of the annotated pathologies as positive examples and label all other images as negative examples. For the pneumonia detection task, we randomly split the dataset into training (28744 patients, 98637 images), validation (1672 patients, 6351 images), and test (389 patients, 420 images). There is no patient overlap between the sets.

Selecting the best model





Selecting the best model



- The training score is everywhere higher than the validation score.
- For very low model complexity, the model is **under-fit**: the model is a poor predictor both for the training data and for any previously unseen data.
- For very high model complexity, the model is **over-fit**: the model predicts the training data very well, but fails for any previously unseen data.
- For some intermediate value, the validation curve has a maximum. This level of complexity indicates a suitable trade-off between bias and variance.

Unsupervised vs. Supervised

Unsupervised learning

- **Does not** require labeled data.
- The algorithm must discover by itself hidden/underlying data structure.
- The number of classes and their nature **have not been** predetermined.
- Often used to:
  - Identify patterns and trends
  - Cluster similar data into a specific number of groups

Supervised learning

Require labels.

Requires human oversight.

Unsupervised Learning

K-means

It is an iterative algorithm that divides the unlabeled dataset into **k** different clusters in such a way that each sample belongs only to one group that has similar properties.

Initialization: set **k** centroids (randomly)

1) Assign each point to the cluster of the nearest centroid measured with a specific distance metric
2) Compute new centroid points (the centroid is the center, i.e., *mean point*, of the cluster)
3) Go back to Step 1), stop when no more new assignment (i.e., membership in each cluster no longer changes)

Unsupervised Learning

K-means

It is an iterative algorithm that divides the unlabeled dataset into **k** different clusters in such a way that each sample belongs only to one group that has similar properties.

Unsupervised Learning

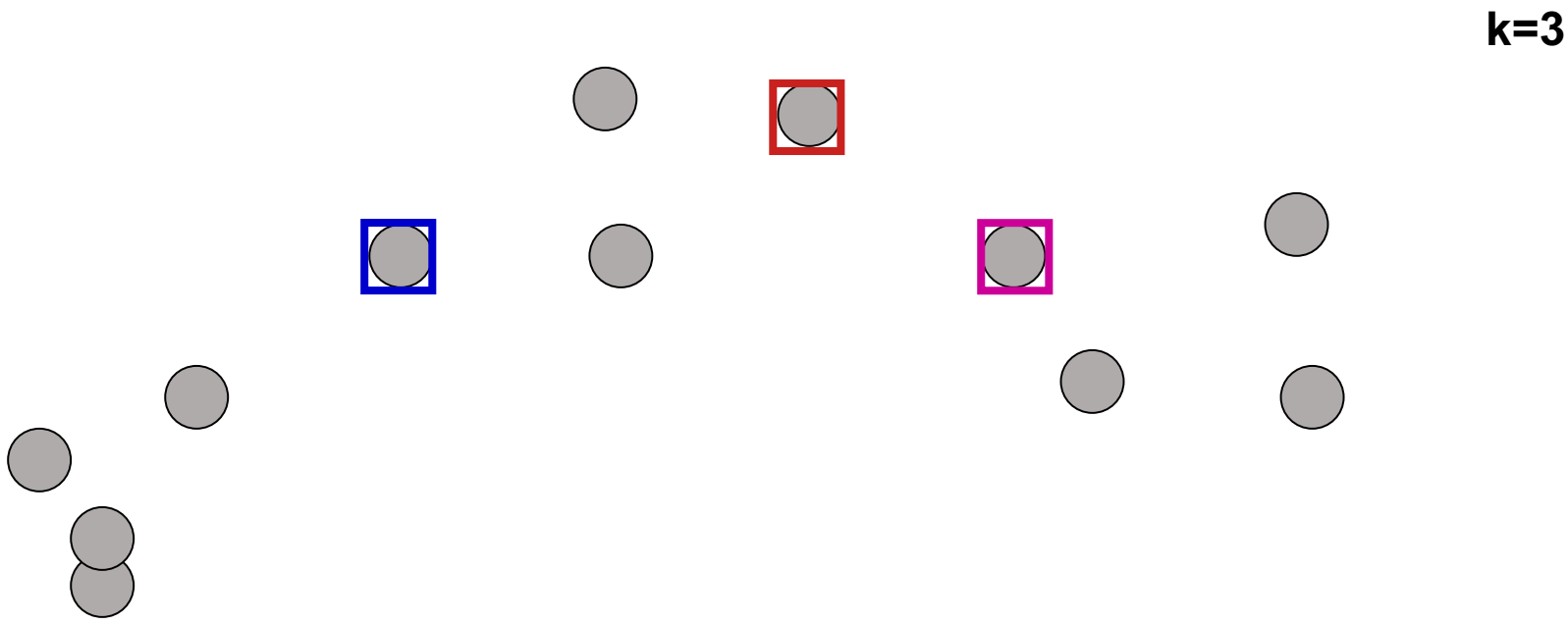
K-means

Initialization: set **k** centroids (randomly)

1) Assign each point to the cluster of the nearest centroid measured with a specific distance metric
2) Compute new centroid points (the centroid is the center, i.e., *mean point*, of the cluster)
3) Go back to Step 1), stop when no more new assignment (i.e., membership in each cluster no longer changes)

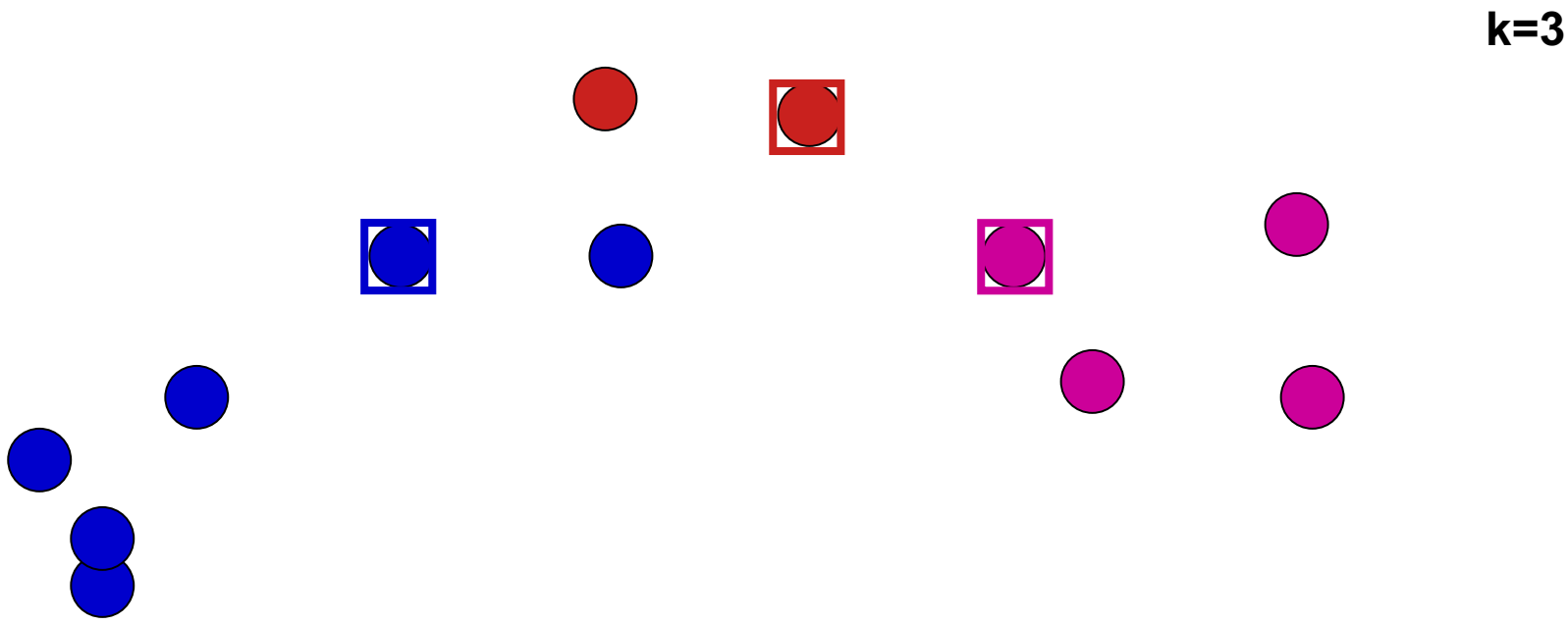
K-means: An example

Initialization: set **k** centroids (randomly)



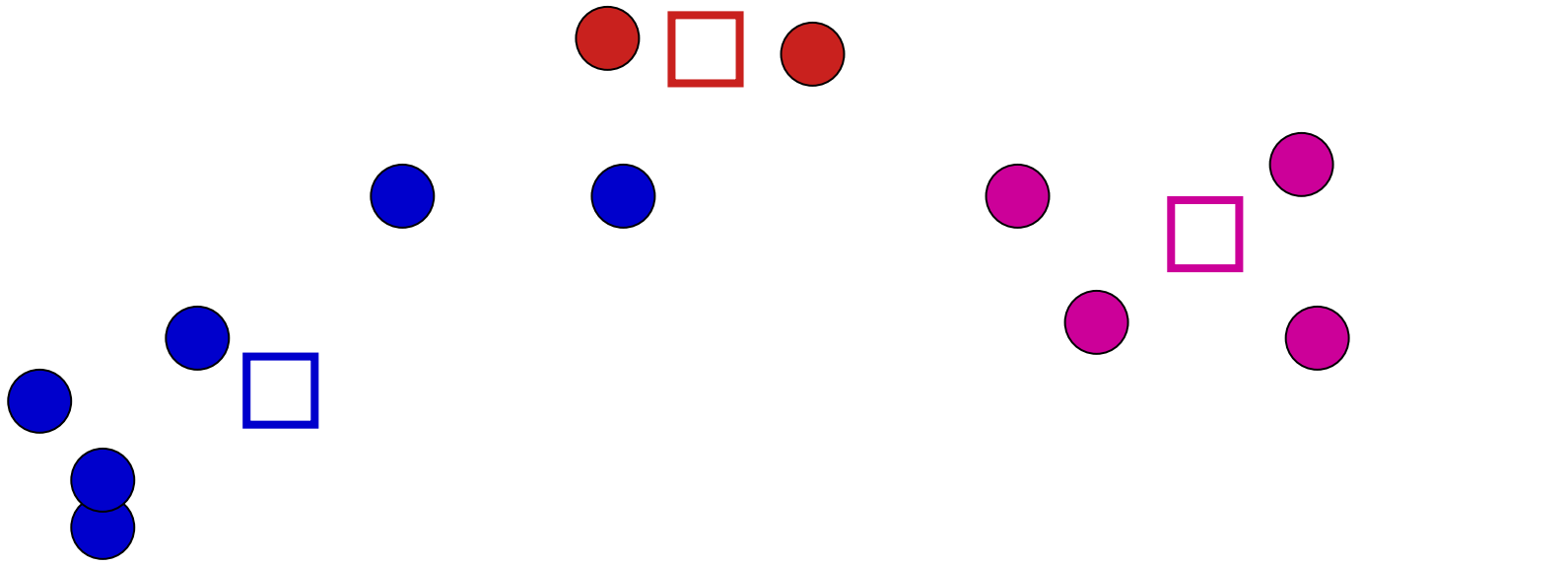
K-means: An example

Assign points to nearest centroid



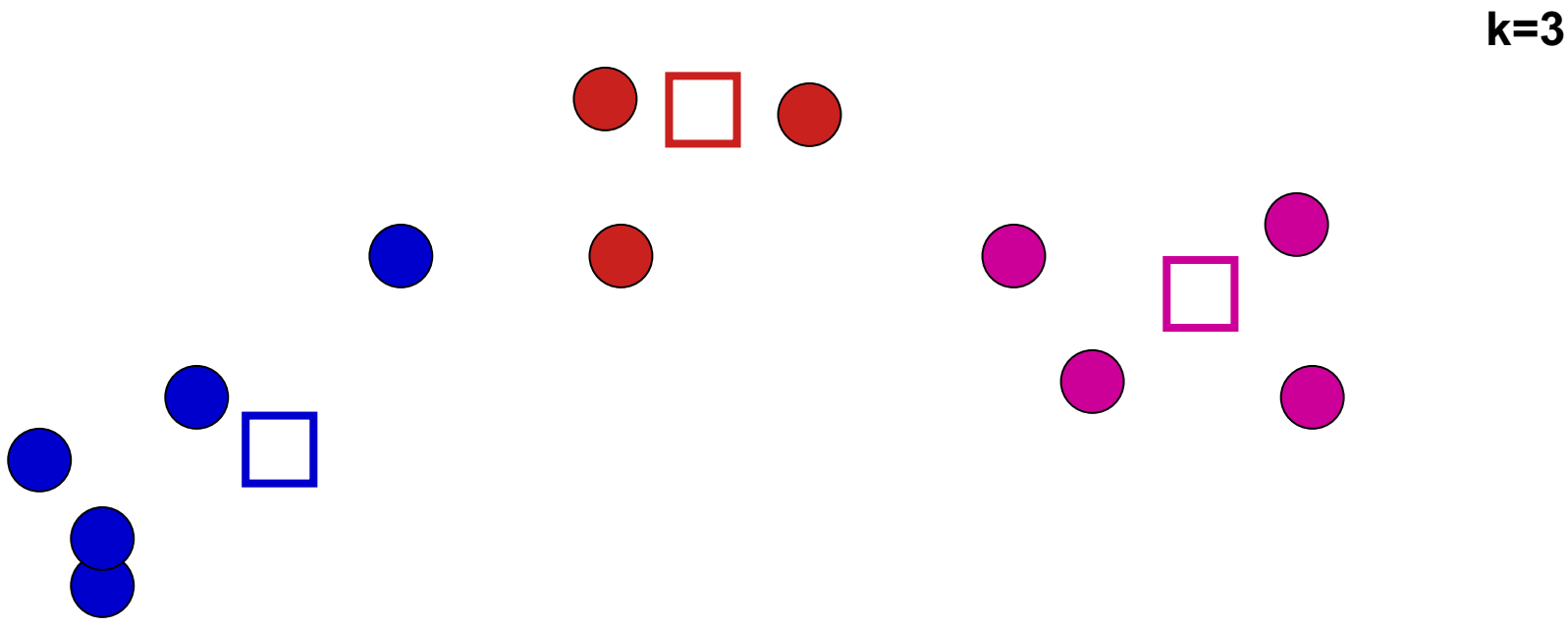
K-means: An example

Compute new centroid points



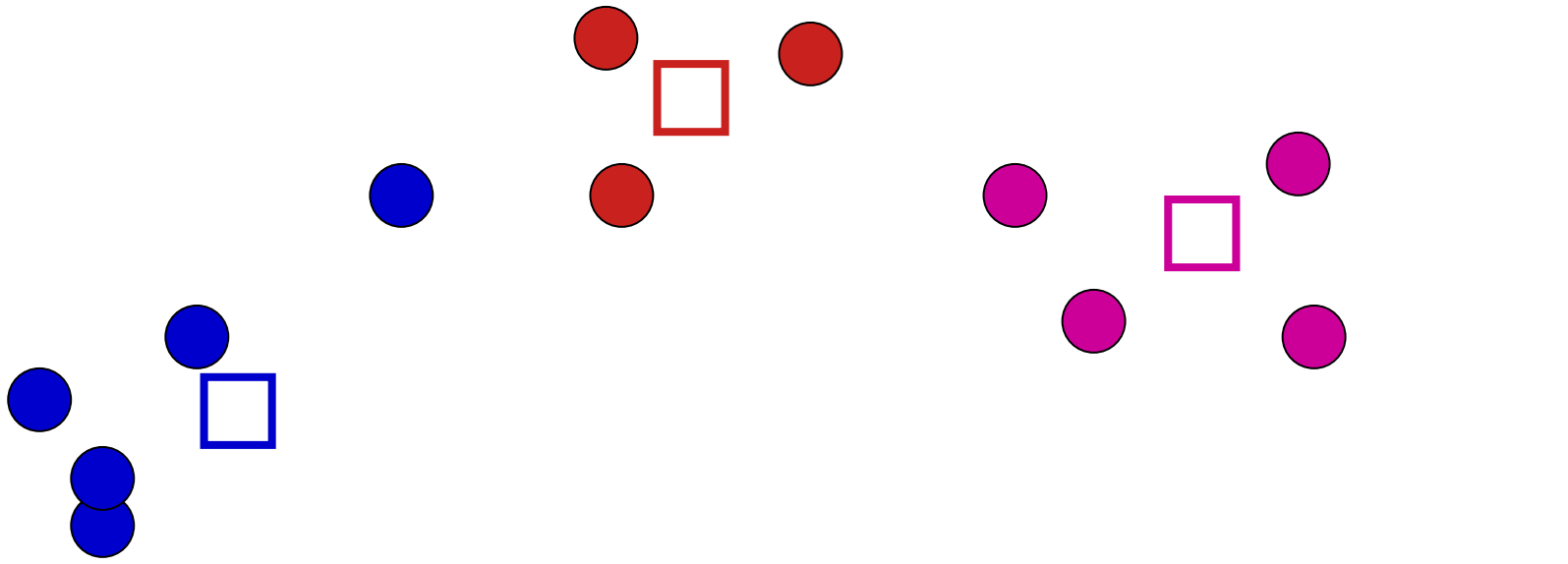
K-means: An example

Assign point to nearest centroid



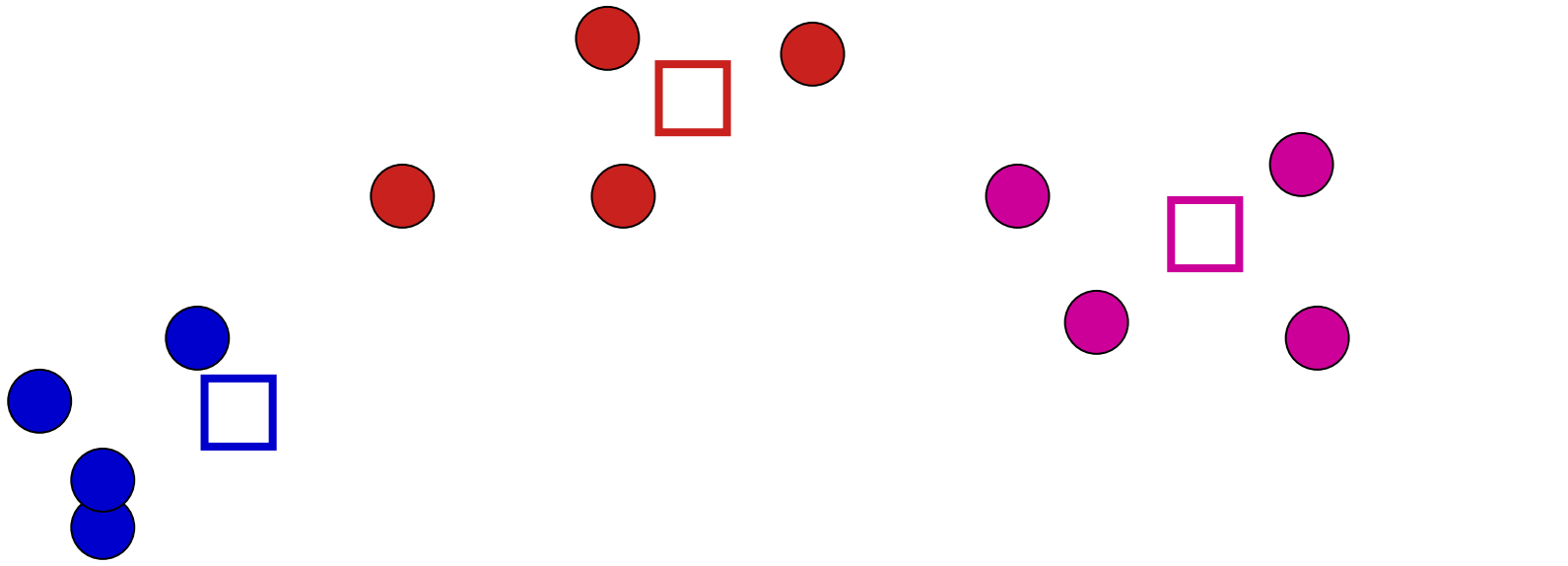
K-means: An example

Compute new centroid points



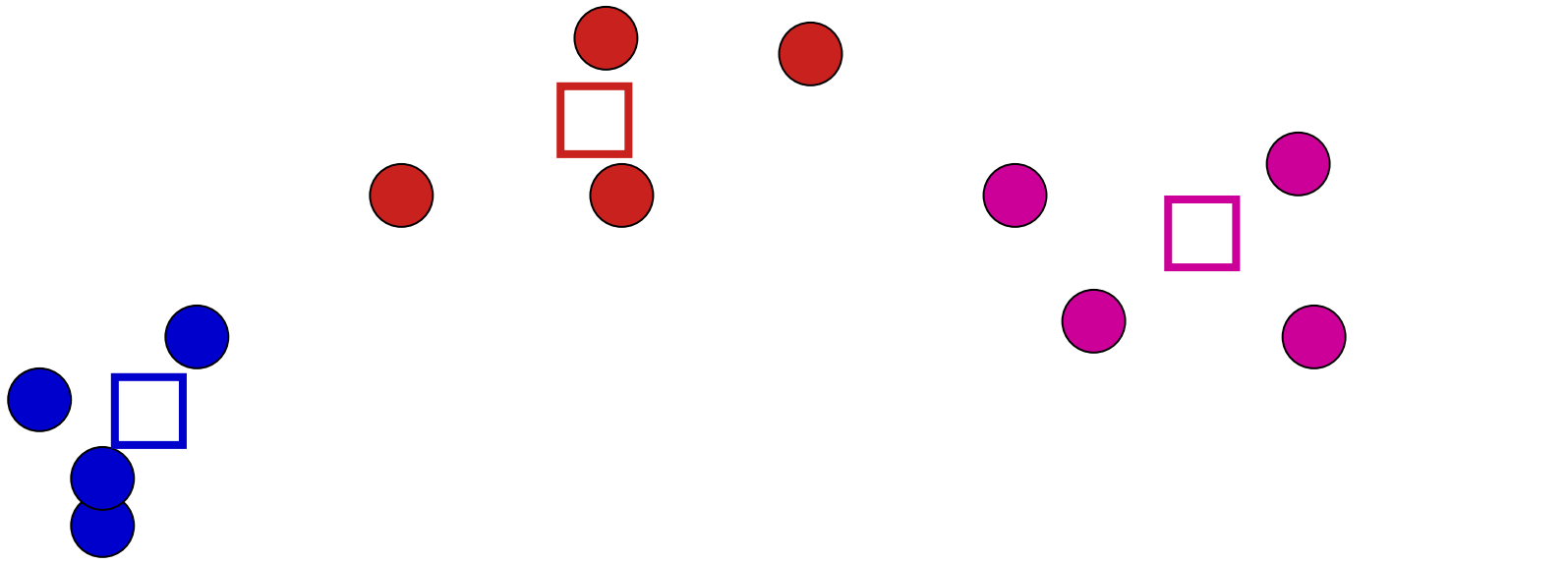
K-means: An example

Assign point to nearest centroid



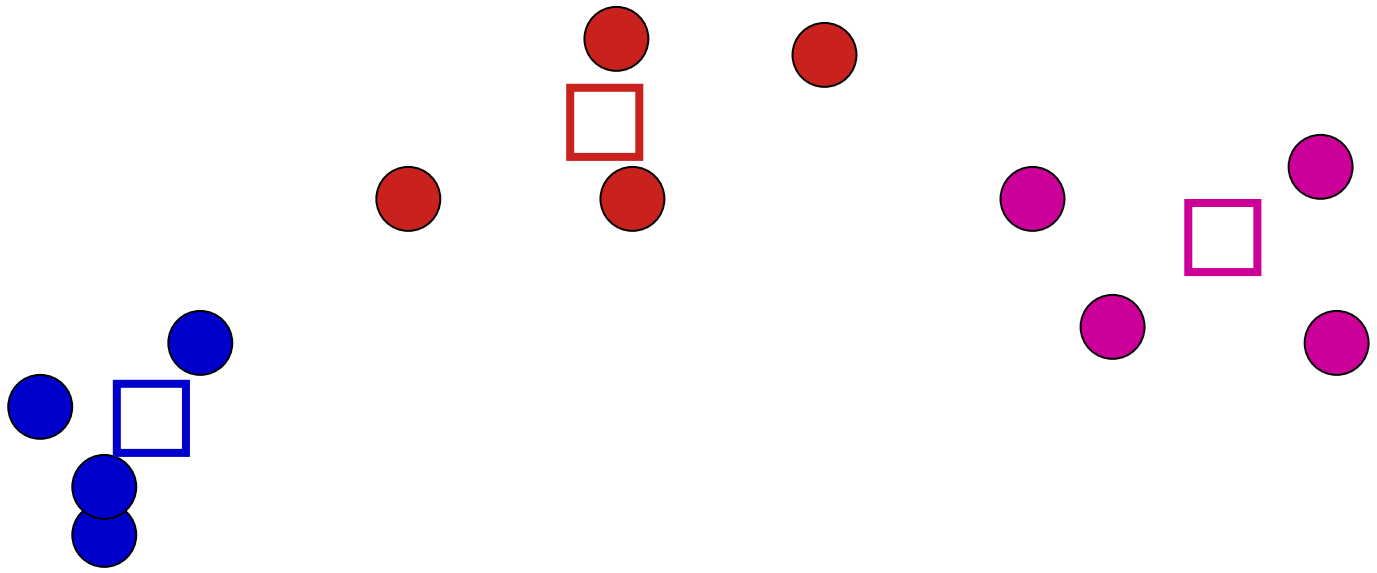
K-means: An example

Compute new centroid points



K-means: An example

Assign point to nearest centroid



No changes: **Done!**

Unsupervised Learning

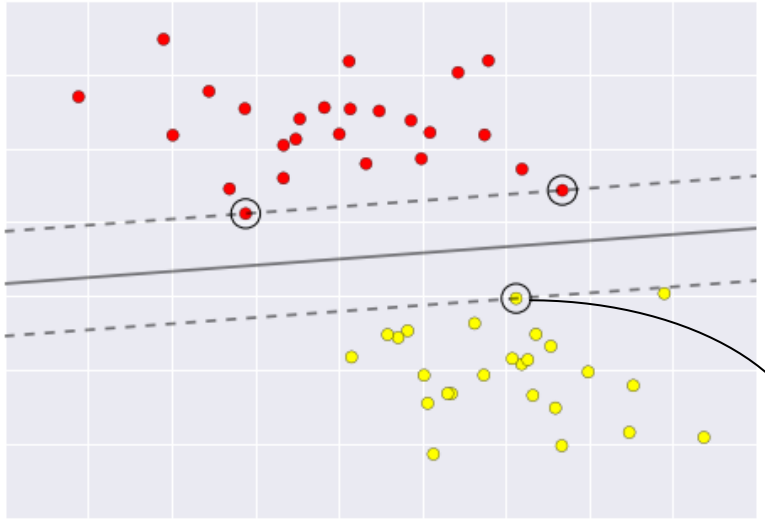
Other examples:

Autoencoders
GANs: **G**enerative **A**dversarial **N**etworks

https://samsunglabs.github.io/MegaPortraits/

Supervised Learning

Support Vector Machine (SVM)

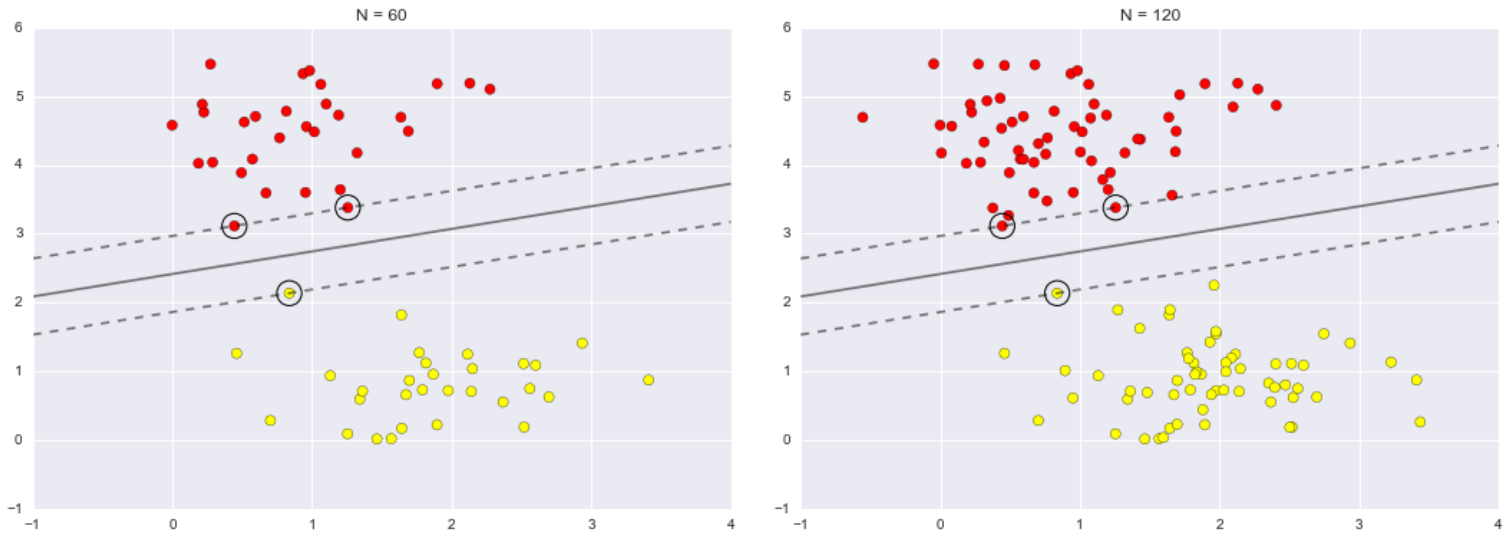


Support vectors

Finding the line that maximizes the margin between the two sets of points.

A good separation is achieved by the line that has the largest distance to the nearest training-data point of any class

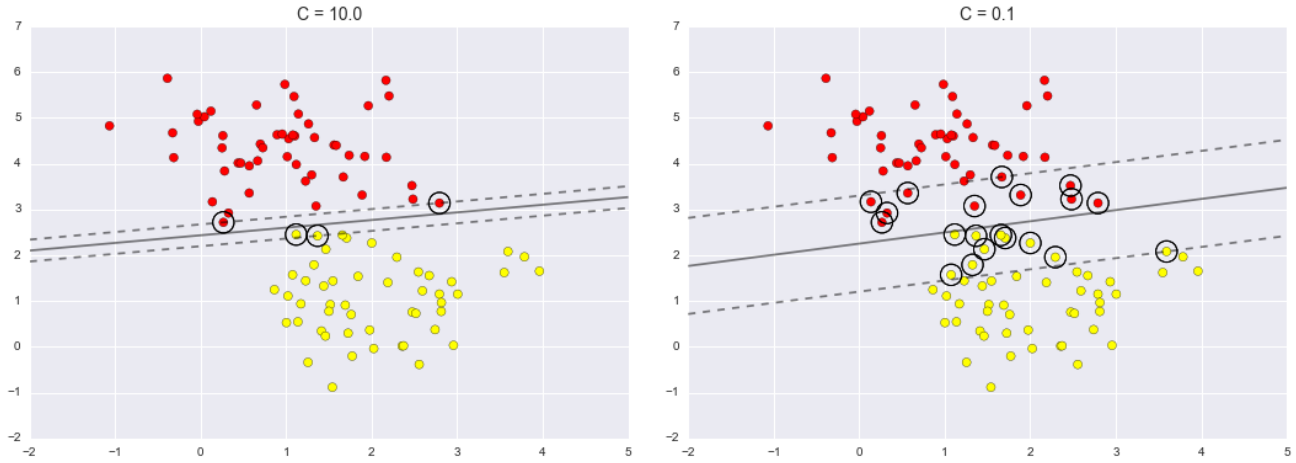
Support Vector Machine (SVM)



After finding the dividing line, only the position of the support vectors matter

Support Vector Machine (SVM)

Softening margins



For very large C, the margin is hard, and points cannot lie in it.
For smaller C, the margin is softer, and can grow to encompass some points.

Support Vector Machine (SVM)

Advantages

•Their dependence on relatively few support vectors means that they are very compact models, and take up very little memory.

•Once the model is trained, the prediction phase is very fast.

•Still effective in cases where number of dimensions is greater than the number of samples.

Disadvantages

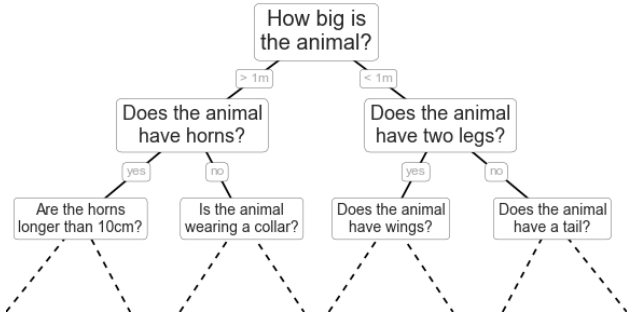
•It might be computational expensive for large numbers of training samples.

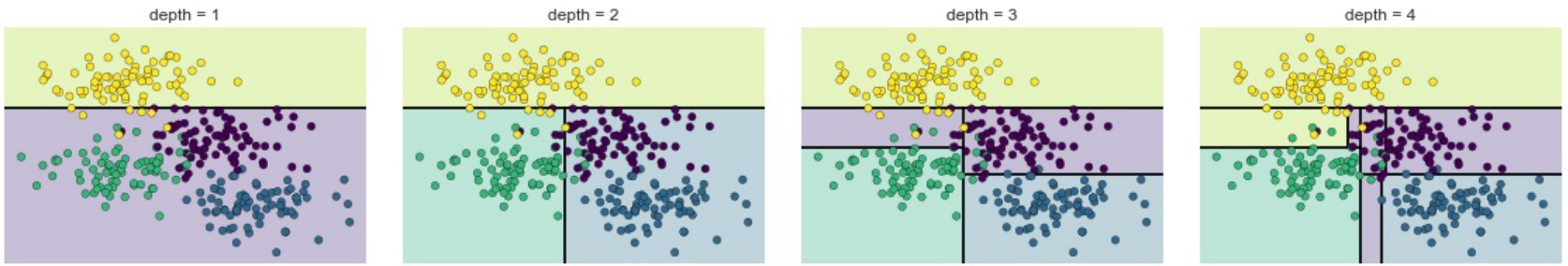
•The results are strongly dependent on a suitable choice for the softening parameter C. This must be carefully chosen via cross-validation, which can be expensive as datasets grow in size.

•The results do not have a direct probabilistic interpretation.

Supervised Learning

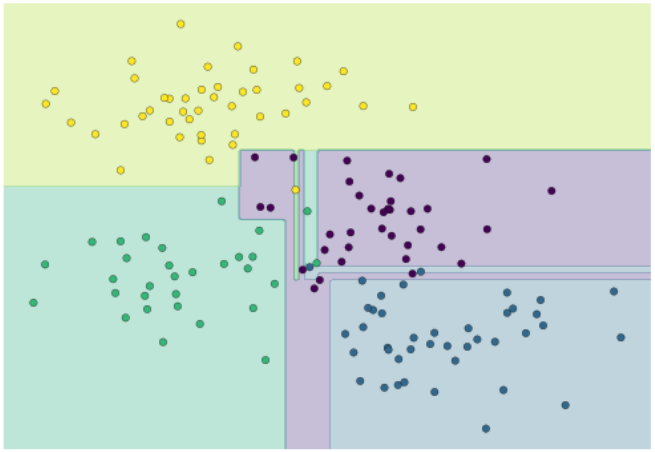
Decision Trees

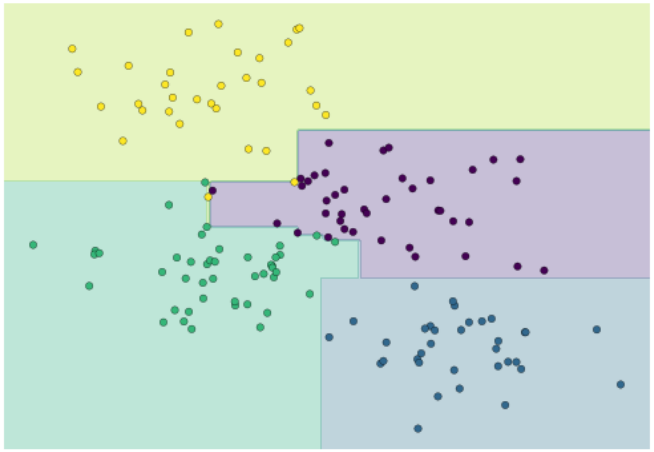




Decision Tree

Overfitting

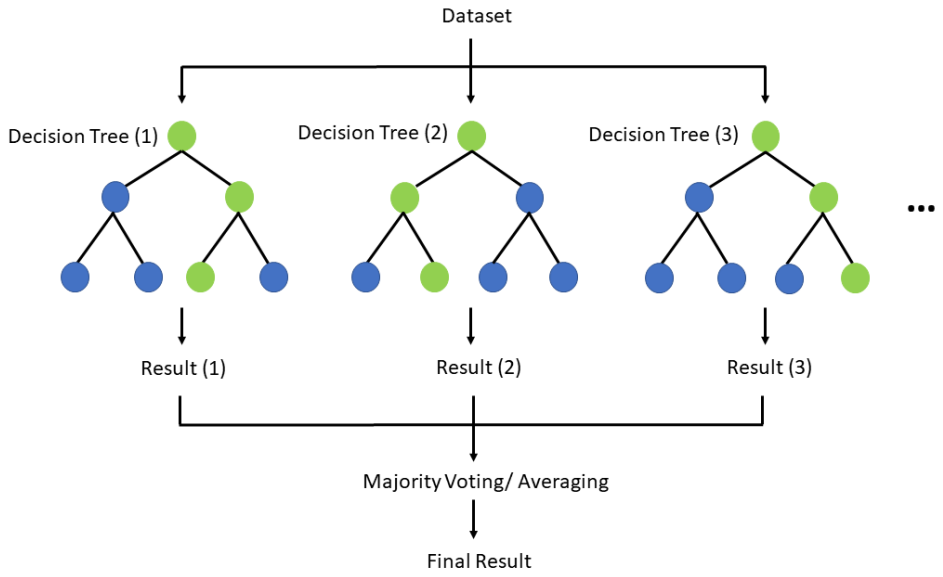




Over-fitting is a general property of decision trees: it is very easy to go too deep in the tree, and to fit details of the particular data rather than the overall properties of the distributions they are drawn from

Supervised Learning

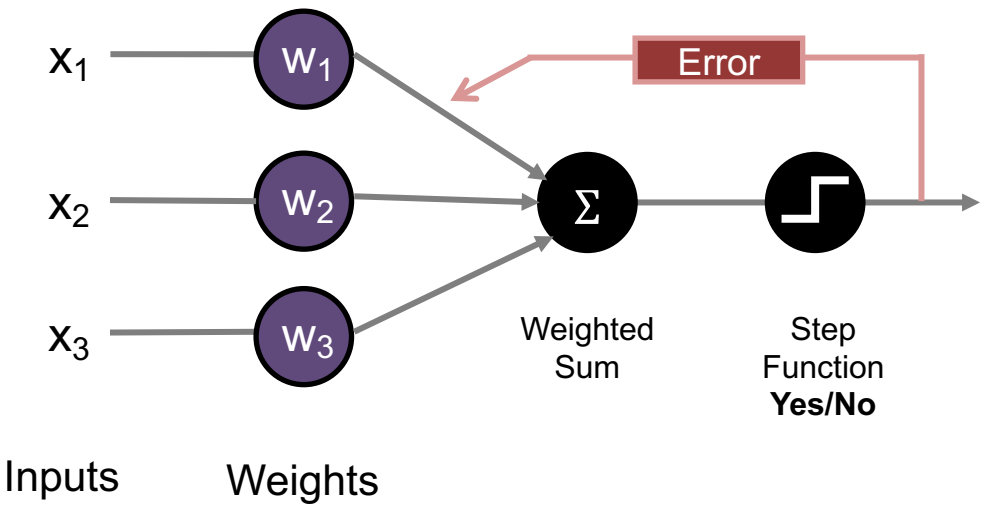
Random Forest



- Both training and prediction are very fast. Both tasks can be straightforwardly parallelized, because the individual trees are entirely independent entities.
- The multiple trees allow for a probabilistic classification: a majority vote among estimators gives an estimate of the probability (accessed in Scikit-Learn with the predict_proba() method).

Neural Networks

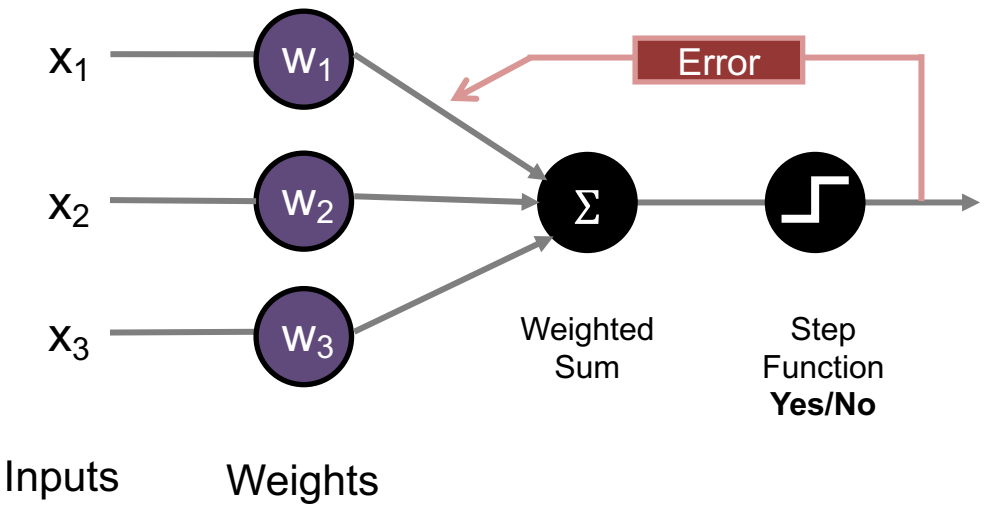
Perceptron



A **weight** is assigned to each input node of a perceptron, indicating the **significance** of that input to the output.
The perceptron's output is a **weighted sum** of the inputs that have been run through an activation function to decide whether or not the perceptron will fire.

Neural Networks

Perceptron



The step function compares this weighted sum to the threshold, which outputs 1 if the input is larger than a threshold value and 0 otherwise