School of Electronic Engineering and Computer Science
Queen Mary University of London

ECS7020P Principles of Machine Learning
Supervised learning: Classification I

Dr Jesús Requena Carrión

19 Oct 2021

Agenda

Intro

Formulating classification problems

Linear classifiers

Logistic model

Nearest neighbours

Summary

Flexibility, complexity and overfitting

In any data science project we assume that our data follow a **pattern** and any deviation from this pattern is considered to be **noise**.

Our models need to be flexible enough to **capture the complexity of the underlying pattern**, but no too flexible, as we might end up **memorising irrelevant noise** details in our data.

A **rigorous methodology** is crucial to avoid falling into common traps, such as overfitting.

So someone has collected our dataset? Great! Let's go ahead, put our methodology to work and build a fantastic model. **Great?**

The 1936 Literary Digest Poll

Alfred Landon
Republican Party

Franklin D. Roosevelt
Democratic Party

- The Literary Digest conducted one of the largest polls ever.
- Predicted Landon would get 57 % of the vote, Roosevelt 43 %. Landon ended up getting 38 % of the votes and Roosevelt 62 %.
- What happened? **Bad sampling**. Names were taken from telephone directories, club membership lists, magazine subscribers lists, etc. Samples were **not representative** of the population.

Know your data!

Agenda

Intro

Formulating classification problems

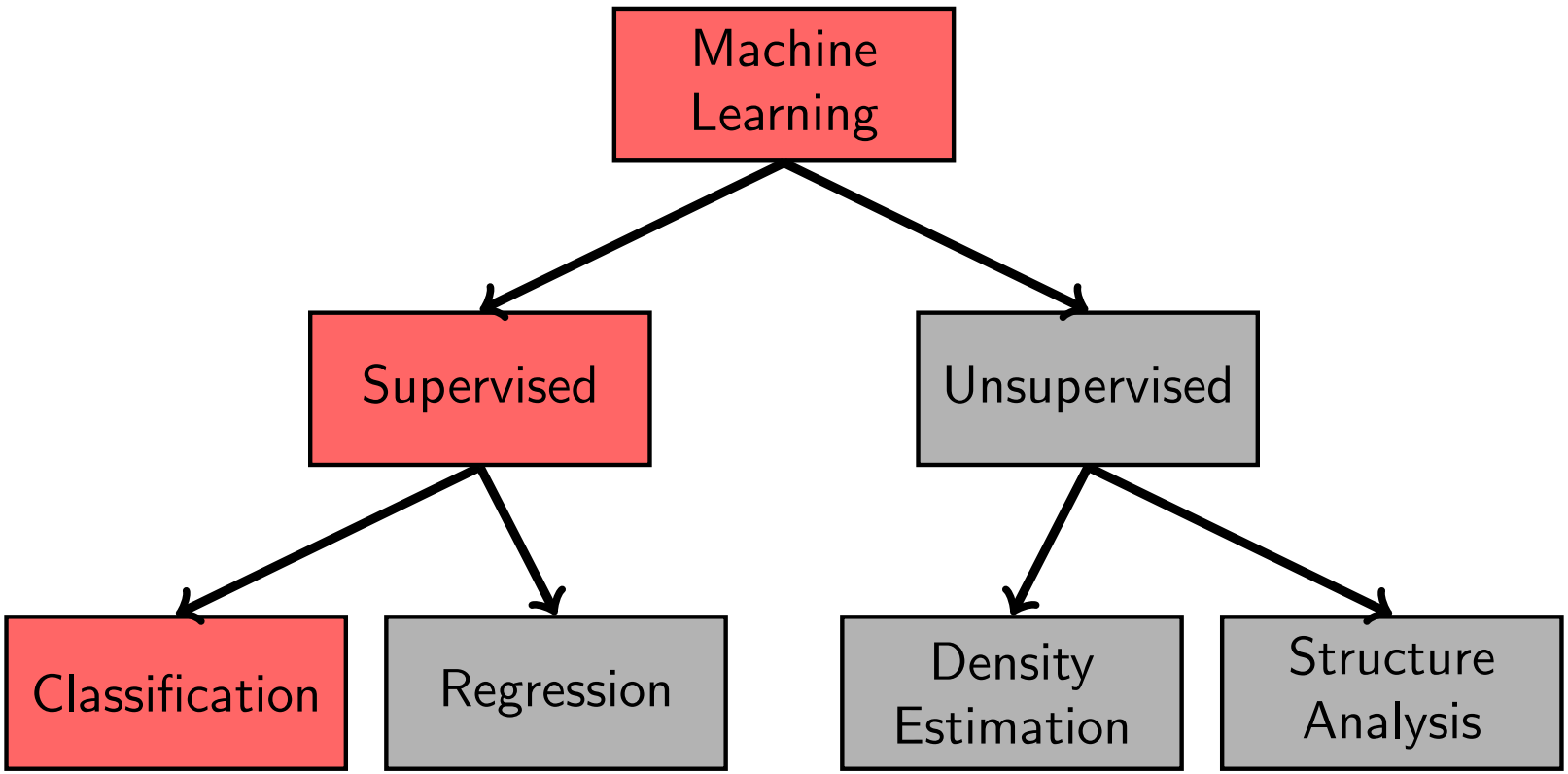
Linear classifiers

Logistic model

Nearest neighbours

Summary

Machine Learning taxonomy



Classification: Problem formulation

In **classification** (also known as **decision** or **detection**):

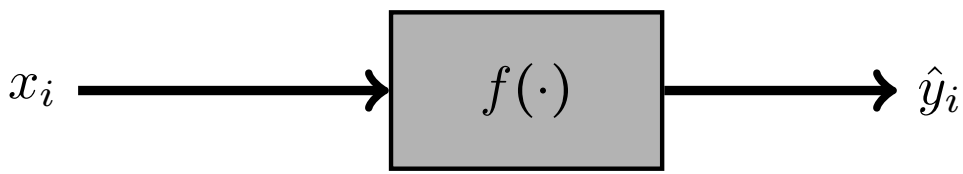
- We have a model that produces a **label** when shown a set of **predictors**.
- The label is **discrete** and its values are called **classes**.

This is not machine learning yet. Can you tell why?

Classification: Problem formulation

In a machine learning classification problem:

- We build a model $\hat{y} = f(x)$ **using a dataset** $\{(x_i, y_i) : 1 \leq i \leq N\}$.
- We have a notion of **model quality**.
- The pair (x_i, y_i) can be read as "sample i belongs to class y_i", or "the label of sample i is y_i".



A binary classification problem

Sentiment analysis allows to identify human opinions expressed in fragments of text. Multiple opinions can be considered, but in it's simplest form two are defined, namely positive and negative.

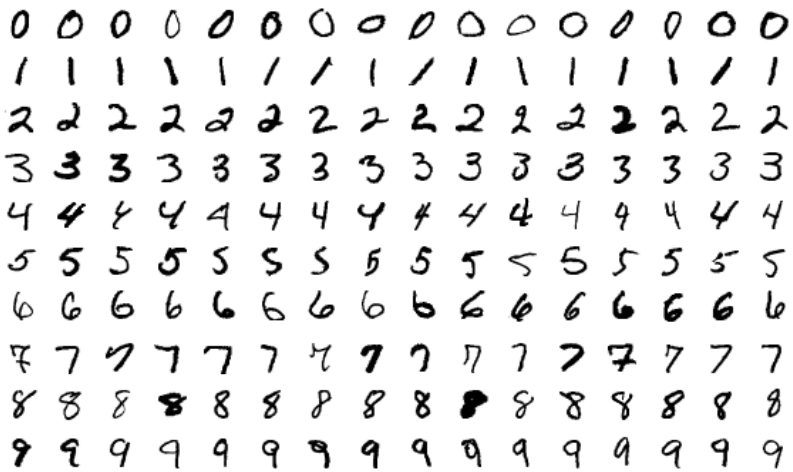
The *Large Movie Review Dataset* was created to build models that recognise polar sentiments in fragments of text:

- It contains 2500 samples for training and 2500 samples for testing
- Each instance consists of a **fragment of text** used as a **predictor** and a **binary label** (0 being negative opinion, 1 positive opinion).
- Downloadable from ai.stanford.edu/~amaas/data/sentiment/

A multiclass classification problem

Recognising digits in images containing handwritten representations is a classic multiclass classification problem. The predictor is an array of values (image) and there are 10 classes, namely $0, 1, 2, \ldots 9$.

In machine learning we use datasets of **labelled images**, i.e. pairs of **images** (predictors) and **numerical values** (label), to build such models.



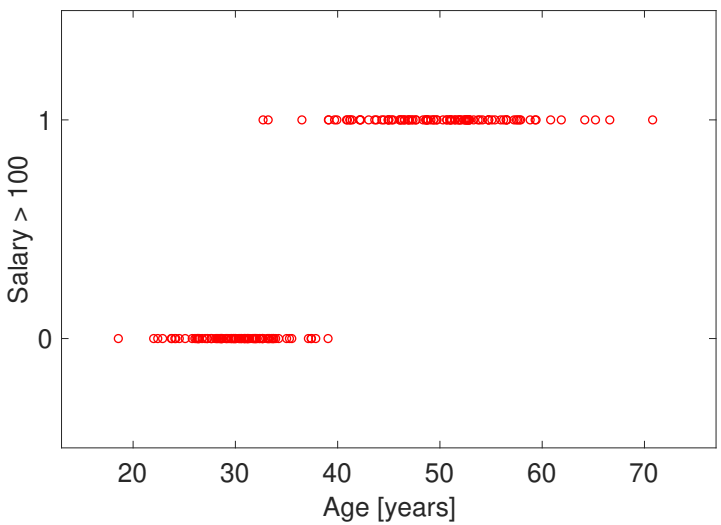
The MNIST dataset is a collection of handwritten digits:

- 60,000 images for training, 10,000 for testing
- Images are black and white, 28×28 pixels
- Downloadable from yann.lecun.com/exdb/mnist

The dataset in the attribute space

Labels can be represented by numerical values on a vertical axis. Be careful: the usual notions of **ordering and distance do not apply** to categorical variables.

One predictor, two classes

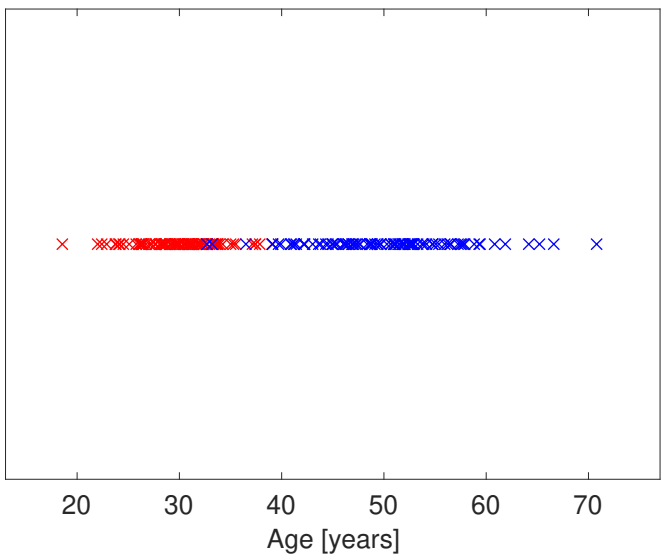


Two predictors, three classes

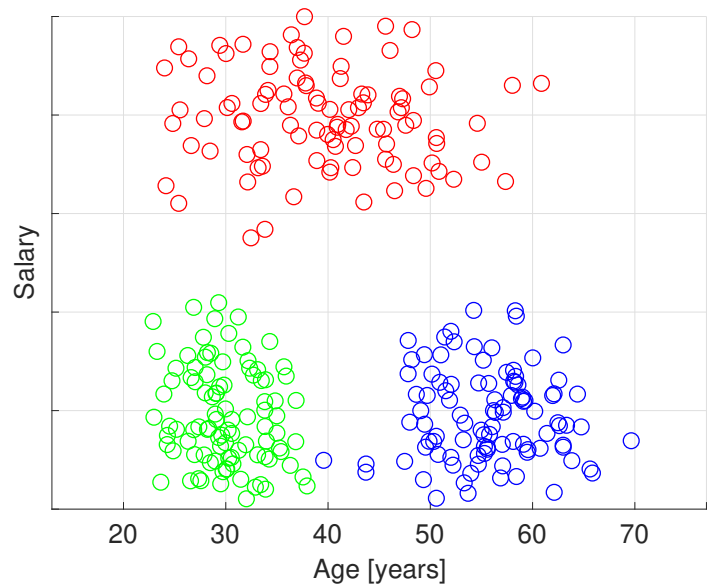
The dataset in the predictor space

A more convenient representation is to use **different symbols for each label** in the **predictor space**.

One predictor, two classes



Two predictors, three classes

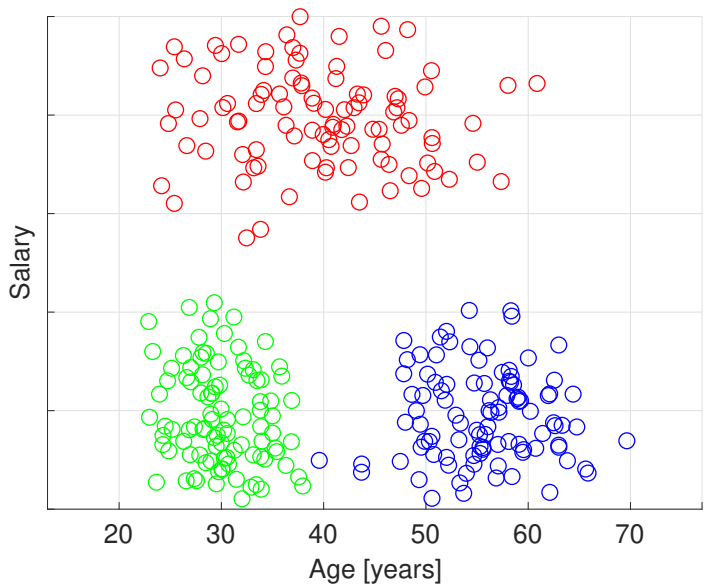


What does a classifier look like?

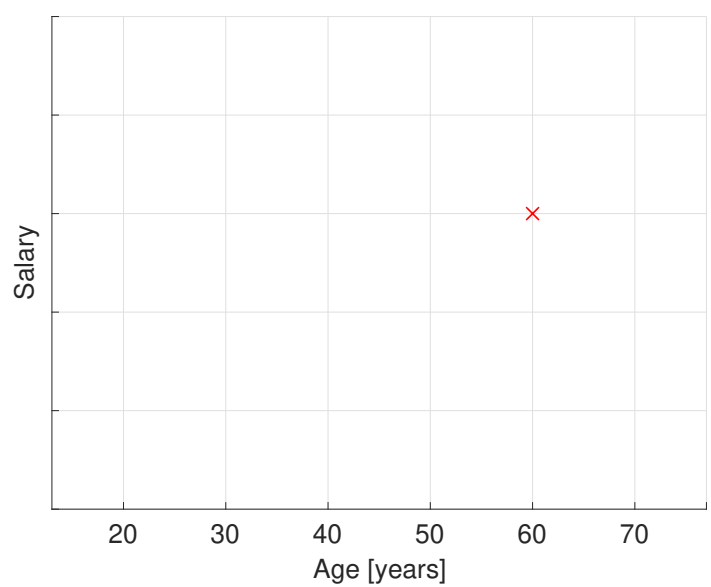
Regression models can be represented as curves/surfaces/hypersurfaces in the attribute space.

Now that we know how to represent our dataset in the predictor space, how can we represent a **classification model**?

Training data



New data point



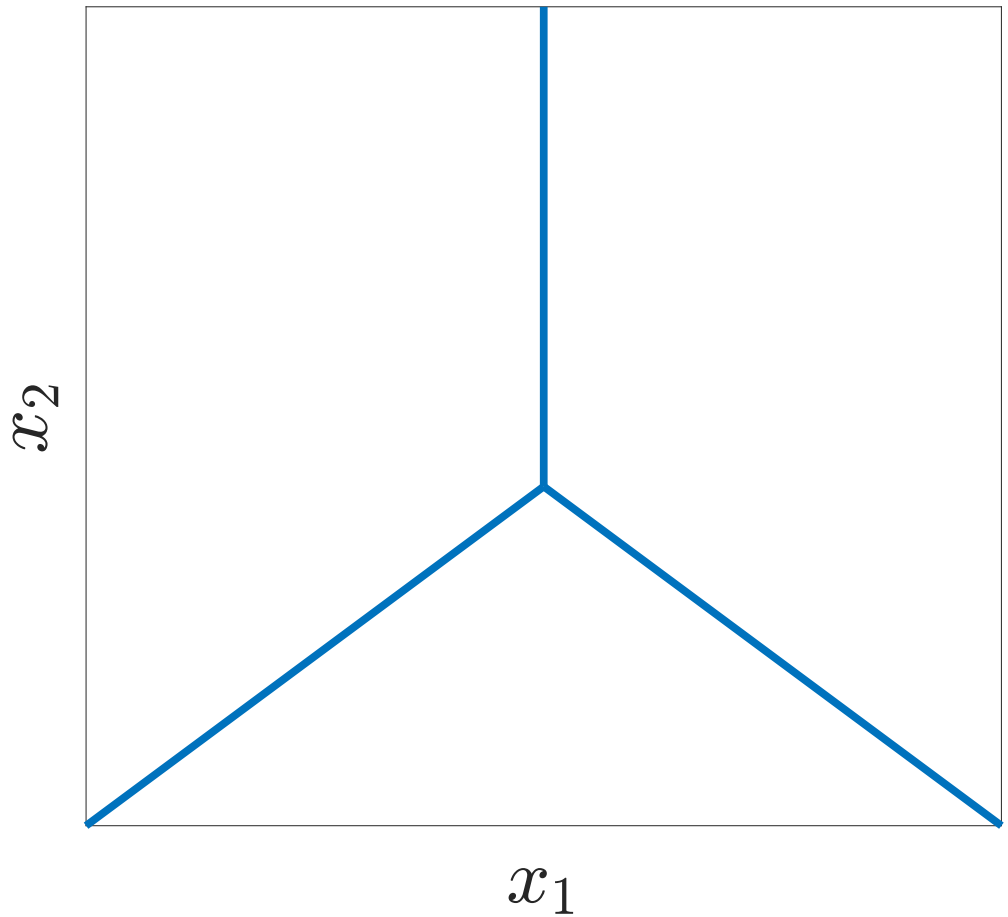
What does a classifier look like?

In classification problems we use the notion of **decision regions** in the predictor space.

- A decision region is made up of **points that are associated to the same label**.
- Regions can be defined by identifying their **boundaries**.
- A solution model in classification is a **partition of the predictor space** into decision regions separated by decision boundaries.

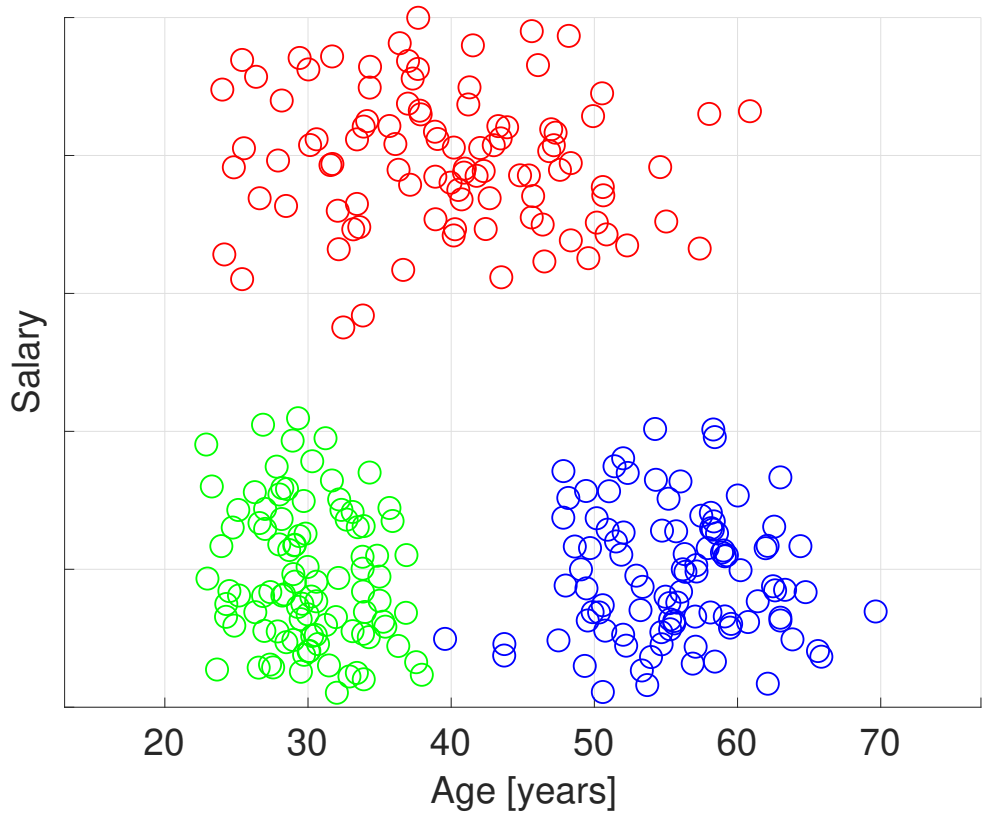
Remember: In machine learning we use datasets to build models, in classification to define the decision regions. The notions of training, test, validation, flexibility, complexity and overfitting also apply to classifiers!

What does a classifier look like?



Classifiers in machine learning

Given the training dataset below, suggest a classifier that labels new samples as o, o or o.



Agenda

Intro

Formulating classification problems

Linear classifiers

Logistic model

Nearest neighbours

Summary

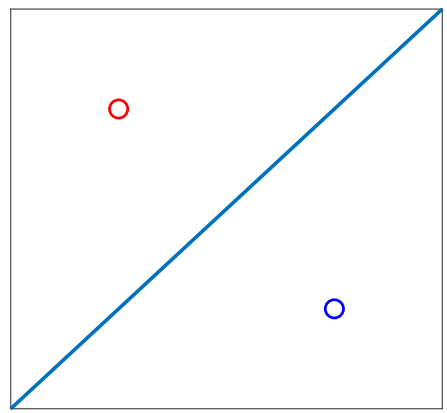
The simplest boundary

Let's consider a **binary** classification problem. The simplest boundary is:

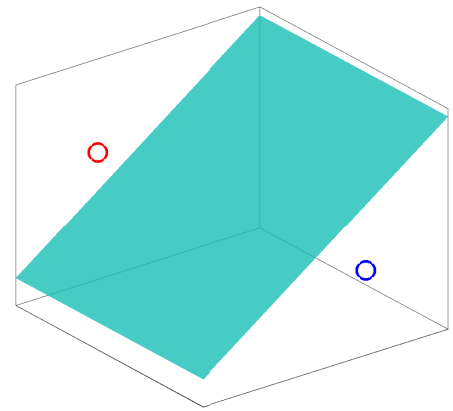
- A point (known as **threshold**) in 1D predictor spaces.
- A straight line in 2D predictor spaces.
- A plane surface in 3D predictor spaces.

All these boundaries are **linear**.

1 predictor



2 predictors



3 predictors

Definition of linear classifiers

Linear classifiers use straight lines/planes/hyperplanes as **boundaries** between decision regions:

- Linear boundaries are defined by the **linear equation** $\boldsymbol{w}^T\boldsymbol{x} = 0$. The extended vector $\boldsymbol{x} = [1, x_1, x_2 ...]^T$ contains the predictors and $\boldsymbol{w}$ is the coefficients vector.
- To classify a sample we simply identify the **side of the boundary** where it lies.

Definition of linear classifiers

If we know the coefficients vector $\boldsymbol{w}$ of a linear boundary, classifying a sample is simple:

- Build the extended vector $\boldsymbol{x}_i$ and compute $\boldsymbol{w}^T \boldsymbol{x}_i$.
  - If $\boldsymbol{w}^T \boldsymbol{x}_i > 0$, we are on one side of the boundary.
  - If $\boldsymbol{w}^T \boldsymbol{x}_i < 0$, we are on the other!
  - If $\boldsymbol{w}^T \boldsymbol{x}_i = 0$... where are we?

Our next step will be to find the **best linear classifier** for a given dataset. To answer this question, we need to define our **quality metric** first.

A basic quality metric

The only operation that we can perform with categorical variables is **comparison**, i.e. we can assess whether $y_i = \hat{y}_i$ is either true or false.

By comparing predictions and true labels, we can identify in a dataset:

- The correctly classified samples (**true predictions**) in each class.
- The incorrectly classified samples (**false predictions**) in each class.

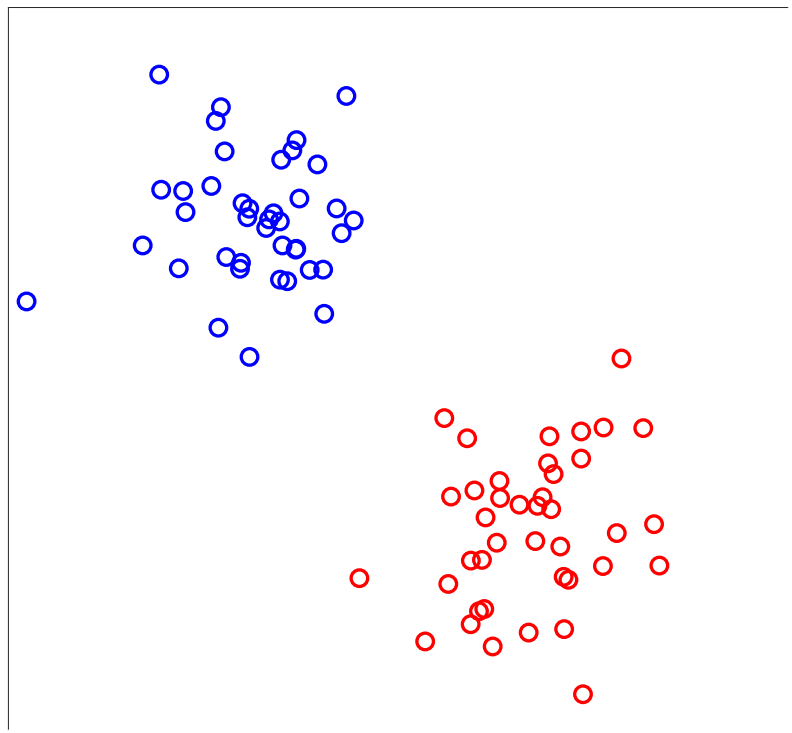
Two common and equivalent notions of quality are the **accuracy** A and the **error** (or misclassification) **rate** $E = 1 - A$, defined as:

$$A = \frac{\#\text{correctly classif. samples}}{\#\text{samples}} \quad , \quad E = \frac{\#\text{incorrectly classif. samples}}{\#\text{samples}}$$

Using these notions of quality, the best classifier can be defined as the one with the highest accuracy (or the lowest misclassification rate).

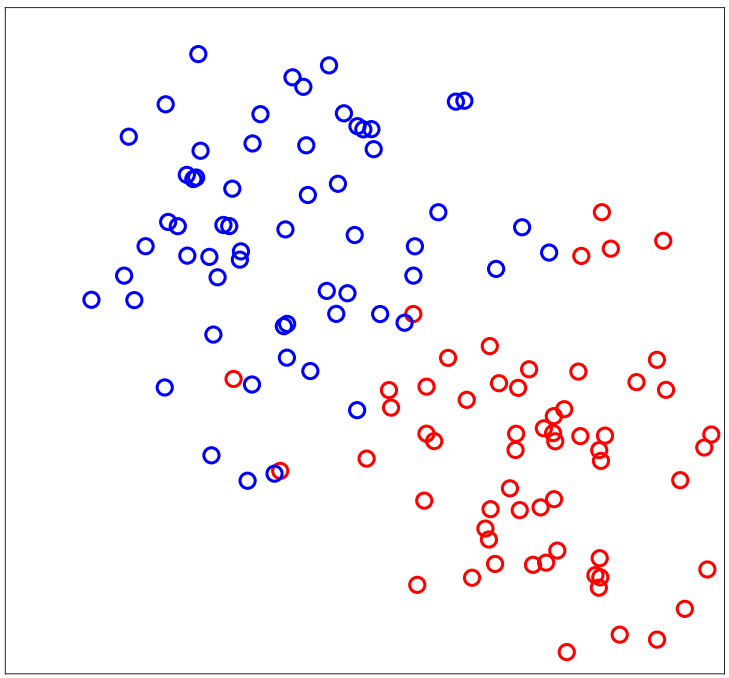
The best linear classifier: Separable case

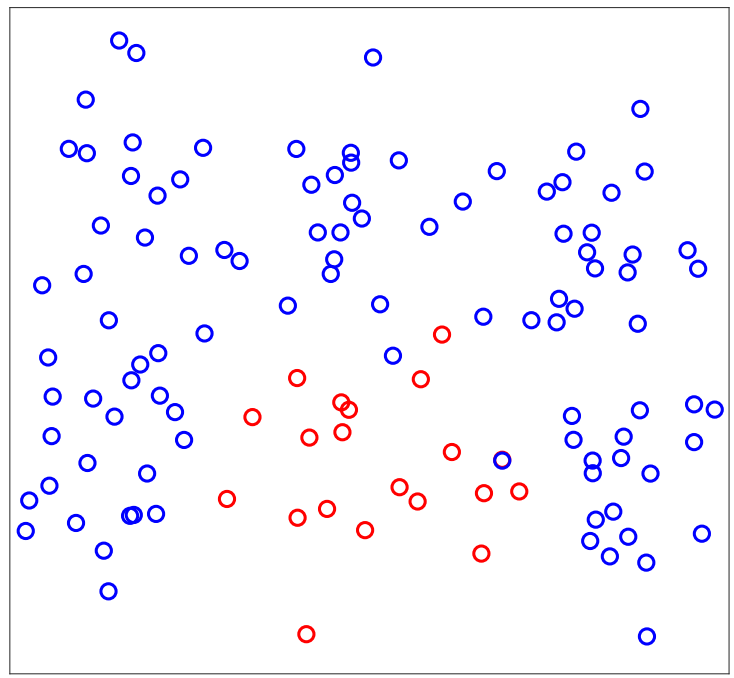
In linearly separable datasets, we can find a linear classifier that achieves the maximum accuracy ($A = 1$, $E = 0$).



The best linear classifier: Non separable case

In non linearly-separable datasets, the accuracy of a linear classifier is $A < 1$ ($E > 0$). The best one will achieve the highest accuracy.





Finding the best linear classifier

We have introduced the linear classifier and presented two quality metrics (accuracy and error rate). If we are given a linear classifier $\boldsymbol{w}$, we can use a dataset to calculate its quality.

The question we want to answer now is, how can we **use data to find** the best linear classifier?

Agenda

Intro

Formulating classification problems

Linear classifiers

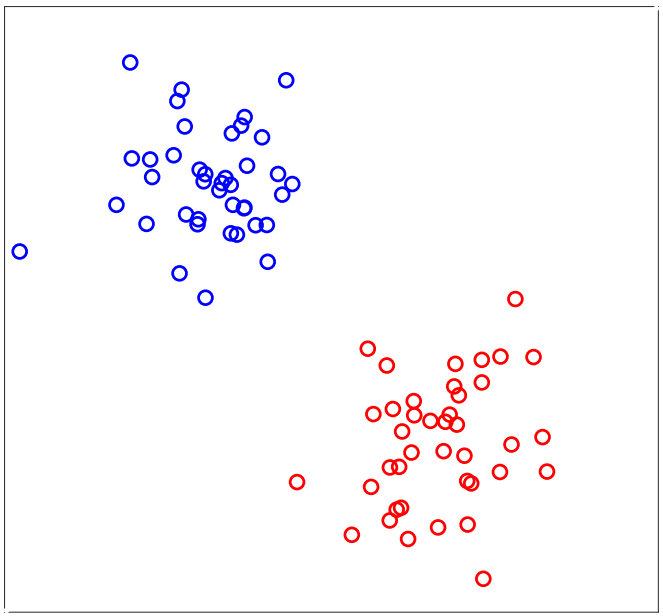
Logistic model

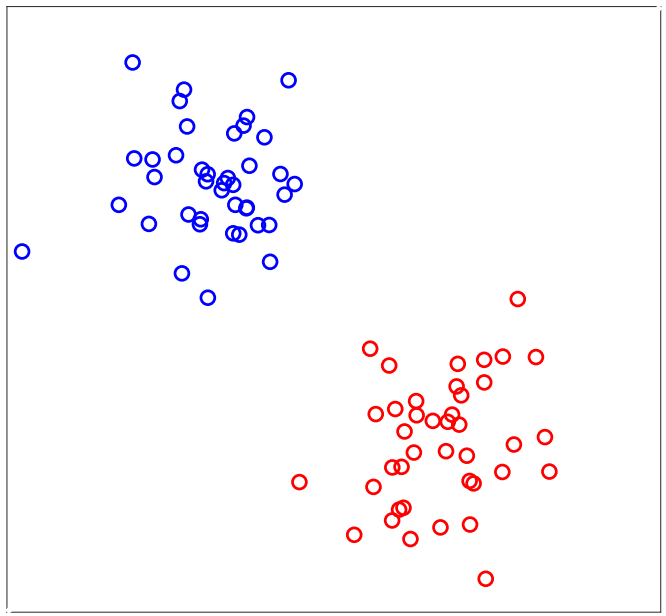
Nearest neighbours

Summary

Best, but risky, linear solutions

Draw two linear boundaries that achieve an accuracy $A = 1$. Which one would you choose? Why?

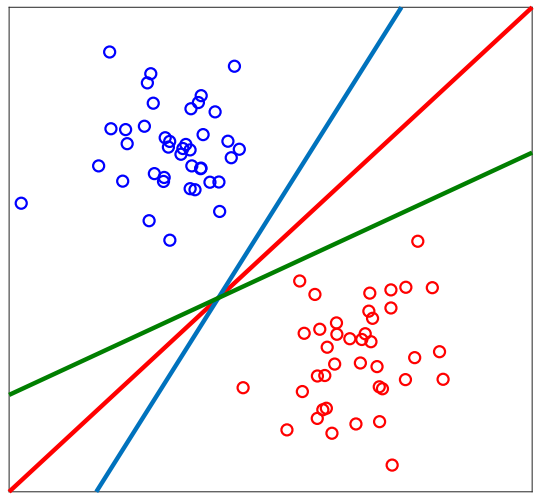




If you prefer one over the other, you might be inadvertently assessing their **generalisation ability** *and modelling the* **distribution of samples**. *Your unconscious ML mind is working faster than your conscious mind!*

Keep that boundary away from me!

As we get closer to the decision boundary, life gets harder for a classifier: it is **noise territory** and we should beware of **jumpy samples**.

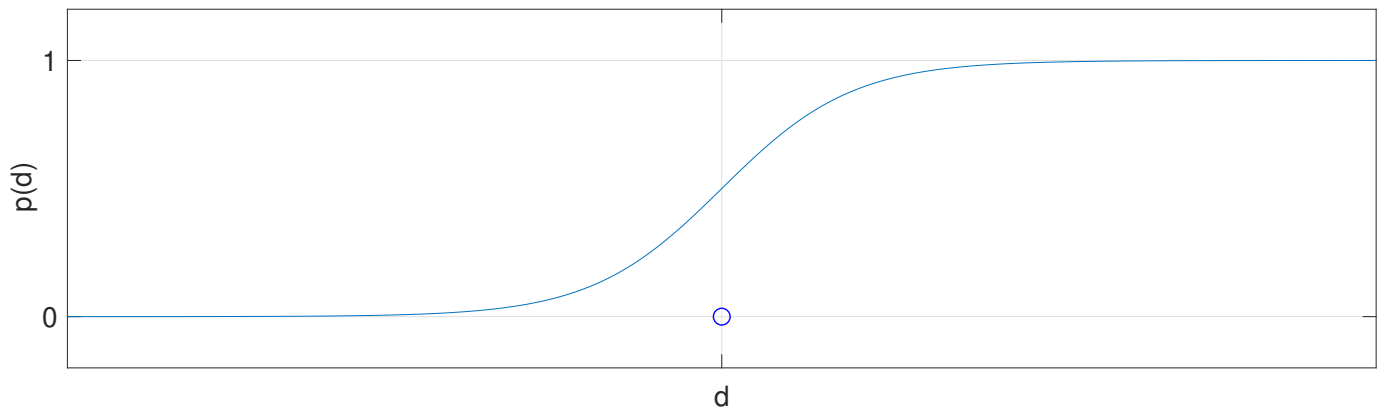


The **further** we are from the boundary, the **higher our certainty** that we are classifying samples correctly.

The logistic model

The logistic function $p(d)$ is defined as

$$p(d) = \frac{e^d}{1+e^d} = \frac{1}{1+e^{-d}}$$



Note that

- $p(0) = 0.5$.
- As $d \to \infty$, $p(d) \to 1$.
- As $d \to -\infty$, $p(d) \to 0$.

The logistic model

Given a linear boundary $\boldsymbol{w}$ and a predictor vector $\boldsymbol{x}_i$, the quantity $\boldsymbol{w}^T\boldsymbol{x}_i$ can be interpreted as the **distance** from the sample to the boundary.

If we set $d = \boldsymbol{w}^T\boldsymbol{x}_i$ in the logistic function, we get:

$$p(\boldsymbol{w}^T\boldsymbol{x}_i) = \frac{e^{\boldsymbol{w}^T\boldsymbol{x}_i}}{1 + e^{\boldsymbol{w}^T\boldsymbol{x}_i}}$$

For a fixed $\boldsymbol{w}$, we will simply denote it as $p(\boldsymbol{x}_i)$ to simplify the notation:

- *When* $\boldsymbol{w}^T\boldsymbol{x} \to \infty$, *the logistic function* $p(\boldsymbol{x}_i) \to 1$
- *When* $\boldsymbol{w}^T\boldsymbol{x} \to -\infty$, *the logistic function* $p(\boldsymbol{x}_i) \to 0$

We will use the logistic function to quantify the notion of **certainty** in classifiers. This certainty is a quantity between 0 and 1.

The logistic model

Consider a linear classifier $\boldsymbol{w}$ that labels samples such that $\boldsymbol{w}^T\boldsymbol{x}_i > 0$ as o and samples such that $\boldsymbol{w}^T\boldsymbol{x}_i < 0$ as o.

Notice that:

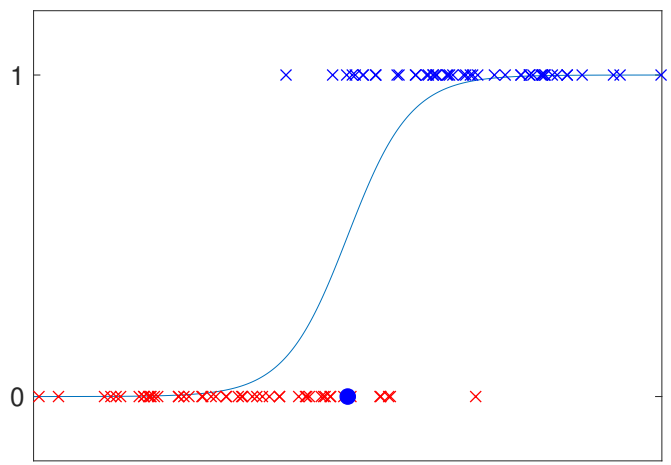
- If $\boldsymbol{w}^T\boldsymbol{x}_i = 0$ ($\boldsymbol{x}_i$ is on the boundary), $p(\boldsymbol{x}_i) = 0.5$.
- If $\boldsymbol{w}^T\boldsymbol{x}_i > 0$ ($\boldsymbol{x}_i$ is in the o region), $p(\boldsymbol{x}_i) \to 1$ as we move away from the boundary .
- If $\boldsymbol{w}^T\boldsymbol{x}_i < 0$ ($\boldsymbol{x}_i$ is in the o region), $p(\boldsymbol{x}_i) \to 0$ as we move away from the boundary.

Here is the crucial point, so use **all your neurons**:

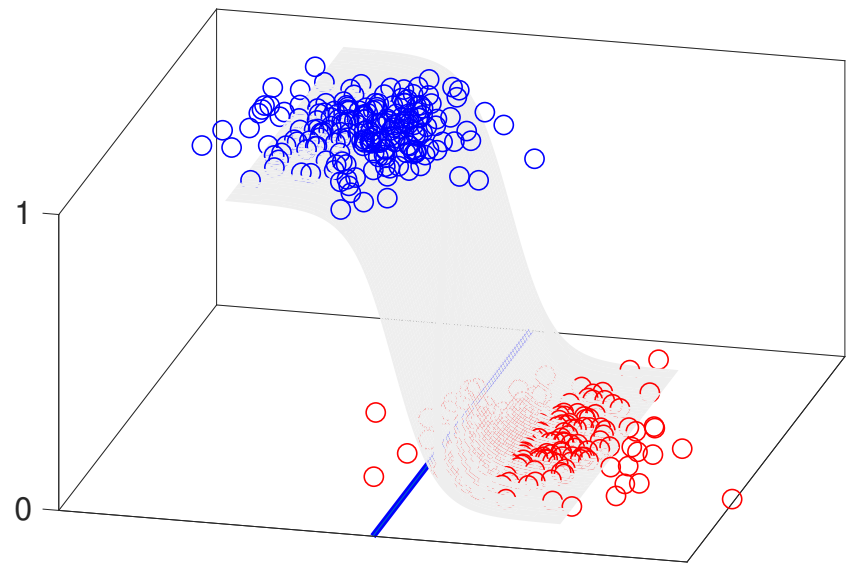
- $p(\boldsymbol{x}_i)$ is the **classifier's certainty** that $y_i =$ o true.
- $1 - p(\boldsymbol{x}_i)$ is the **classifier's certainty** that $y_i =$ o true.

Visualising logistic regression

1D predictor space



2D predictor space



The logistic classifier

We can obtain the classifier's certainty that $\boldsymbol{x}_i$ belongs to either o or o. Can we calculate the certainty for a **labelled dataset** $\{(\boldsymbol{x}_i, y_i)\}$?

The answer is yes, by **multiplying** the individual certainties:

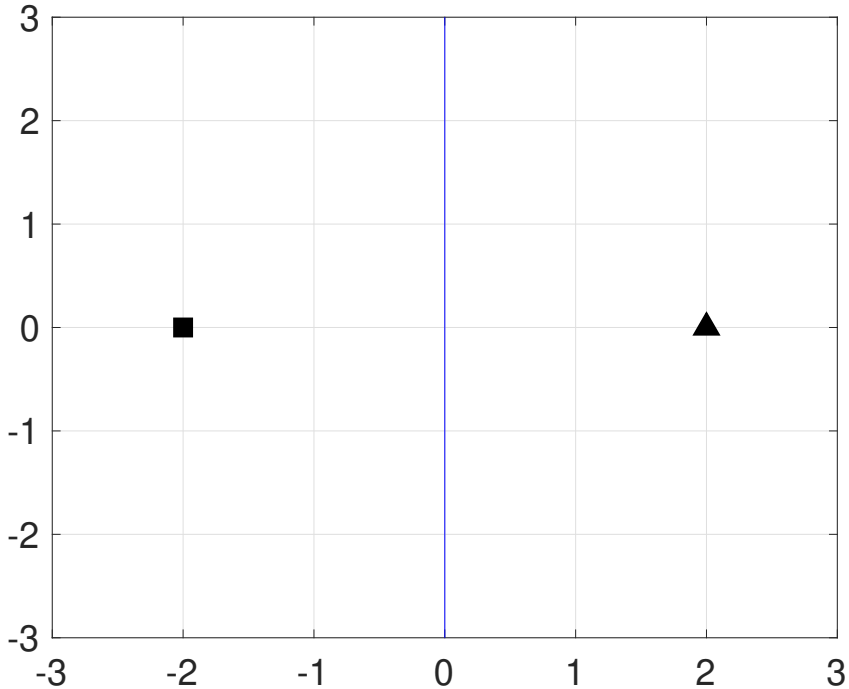
$$L = \prod_{y_i=\text{o}} (1 - p(\boldsymbol{x}_i)) \prod_{y_i=\text{o}} p(\boldsymbol{x}_i)$$

L is known as the **likelihood function** and defines a **quality metric**. Taking logarithms, we obtain the **log-likelihood**:

$$l = \sum_{y_i=\text{o}} \log\left[1 - p(\boldsymbol{x}_i)\right] + \sum_{y_i=\text{o}} \log\left[p(\boldsymbol{x}_i)\right]$$

The linear classifier that maximises L or l is known as the **Logistic Regression** classifier. It can be found using **gradient descent**.

Example I



- Let's define $d_i = \boldsymbol{w}^T \boldsymbol{x}_i$
- We can rewrite the logistic function as

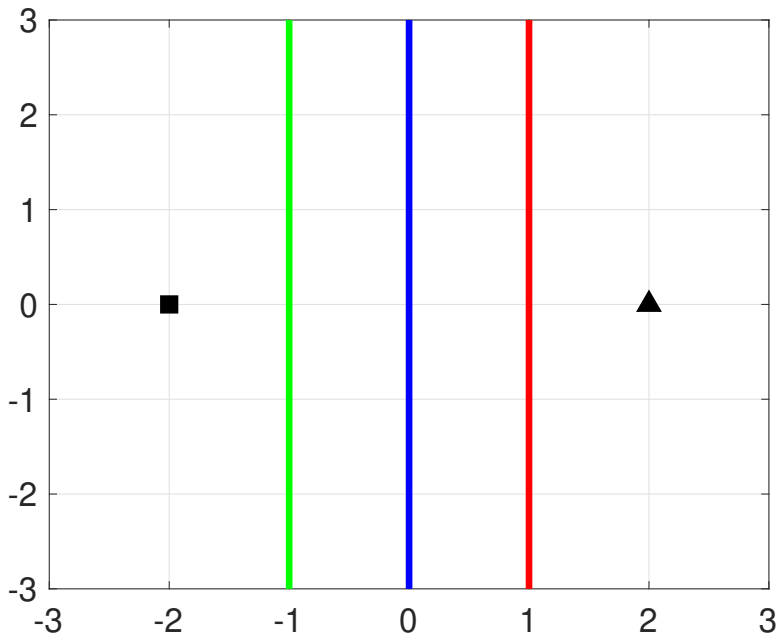
$$p(d_i) = \frac{e^{d_i}}{1 + e^{d_i}}$$

- For instance $p(0) = 0.5$, $p(1) \approx 0.73$, $p(2) \approx 0.88$, $p(-1) \approx 0.27$ and $p(-2) \approx 0.12$

Assume this linear classifier labels samples on the right half-plane as $\triangle$ and samples on the left half-plane as $\square$.

Then $p(\triangle) \approx 0.88$, $1 - p(\square) \approx 0.88$ and $L = p(\triangle)\,(1 - p(\square)) \approx 0.77$.

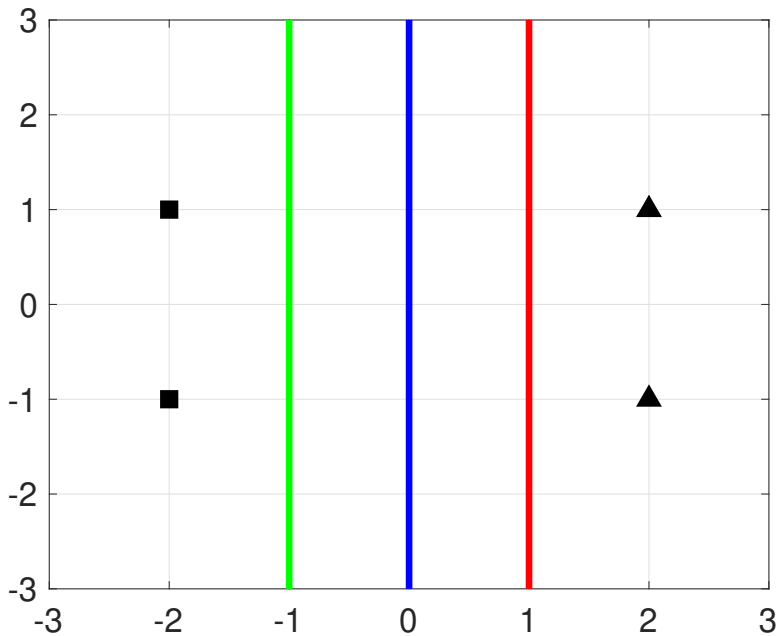
Example II



The global certainty of each classifier (i.e. boundary) is:

- $L = p(\triangle)\,(1 - p(\square)) \approx 0.70$
- $L = p(\triangle)\,(1 - p(\square)) \approx 0.77$
- $L = p(\triangle)\,(1 - p(\square)) \approx 0.70$

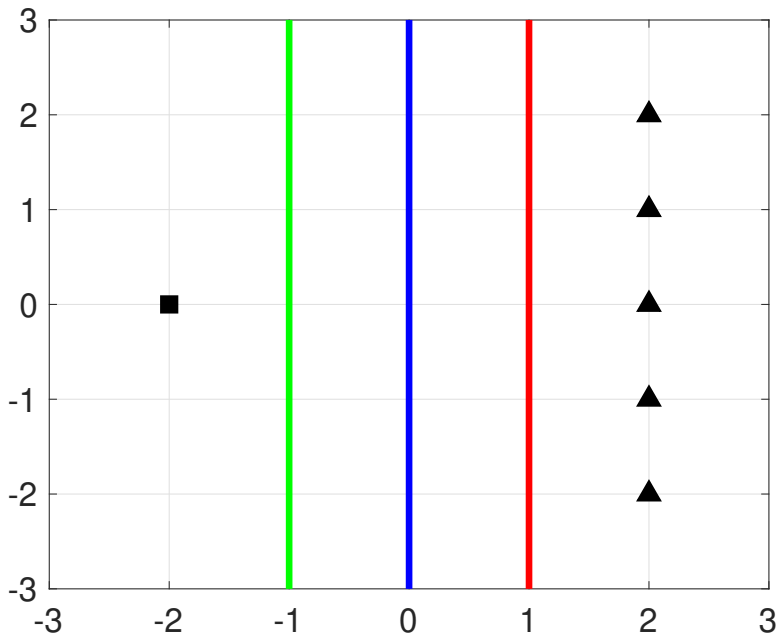
Example III



The global certainty of each classifier (i.e. boundary) is:

- $L \approx 0.49$
- $L \approx 0.60$
- $L \approx 0.49$

Example IV



The global certainty of each classifier (i.e. boundary) is:

- $L \approx 0.20$
- $L \approx 0.47$
- $L \approx 0.57$

Agenda

Intro

Formulating classification problems

Linear classifiers

Logistic model

Nearest neighbours

Summary

Parametric and non-parametric approaches

Linear classifiers belong to the family of **parametric** approaches: a shape is assumed (in this case linear) and our dataset is used to find the best boundary amongst all the boundaries with the preselected shape.

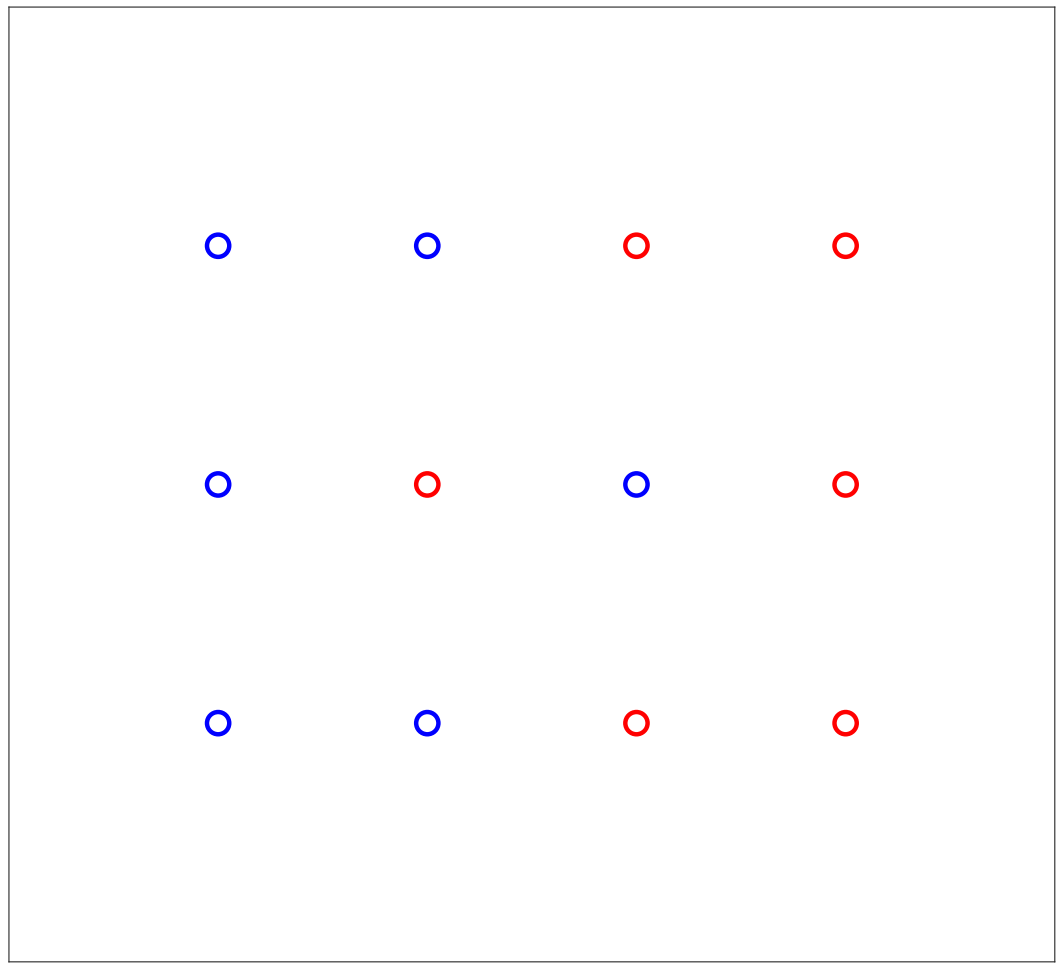
Non-parametric approaches offer a more flexible alternative, as they do not assume any type of boundary. In this section, we will study a popular non-parametric approach, namely **k Nearest Neighbours** (kNN).

Nearest Neighbours

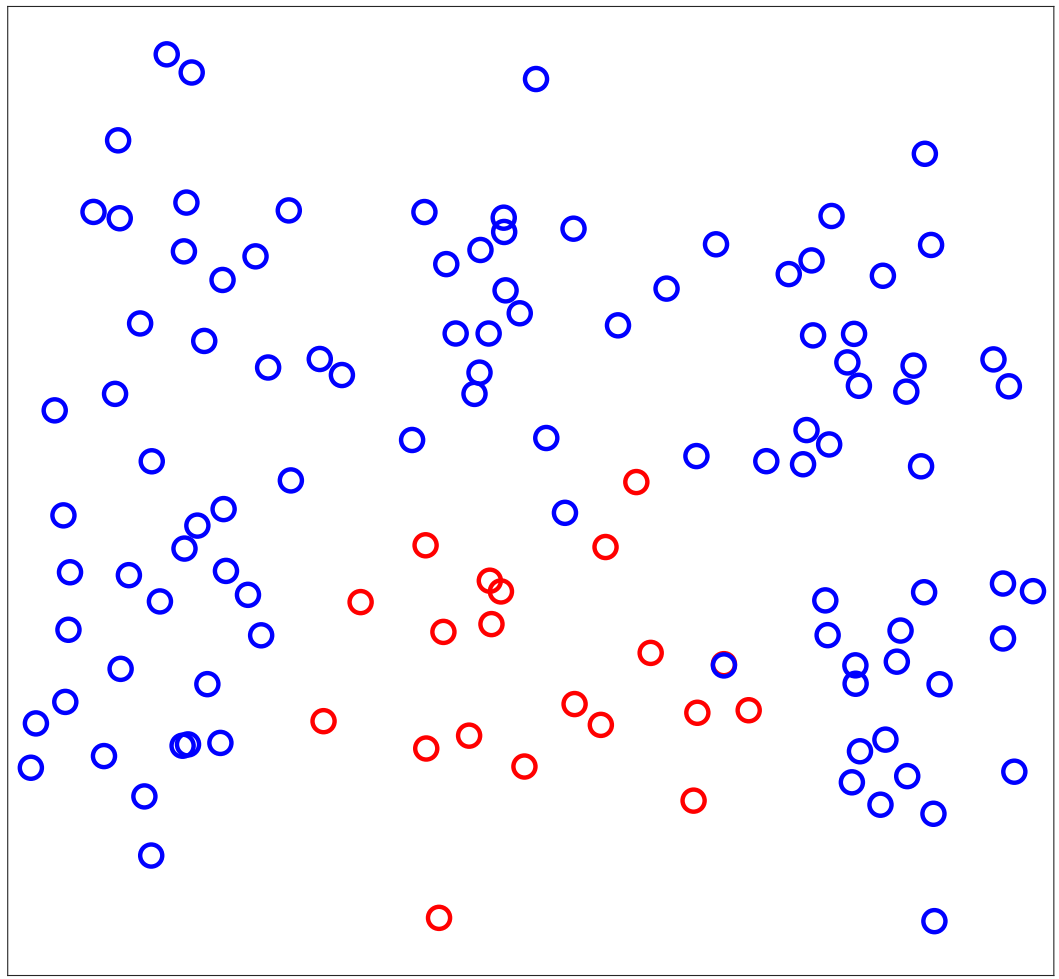
In nearest neighbours (NN), new samples are assigned the **label of the closest (*most similar*) training sample**. Therefore:

- Boundaries are not defined explicitly (although they exist and can be obtained).
- The whole training dataset needs to be **memorised**. That's why sometimes we say NN is an **instance-based method**.

Boundaries in Nearest Neighbours classifiers



Boundaries in Nearest Neighbours classifiers



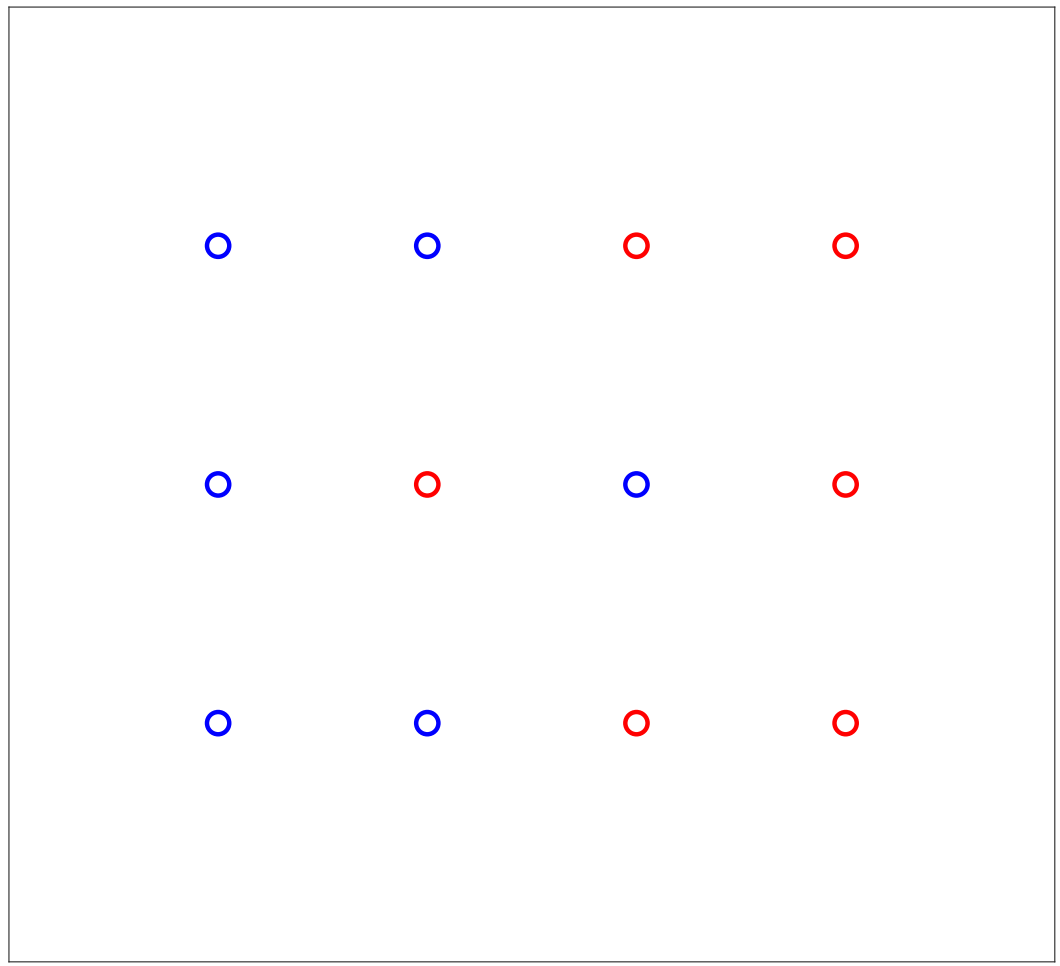
k Nearest Neighbours

Boundaries in nearest neighbours classifiers can be too complex and hard to interpret. Can we smooth them?

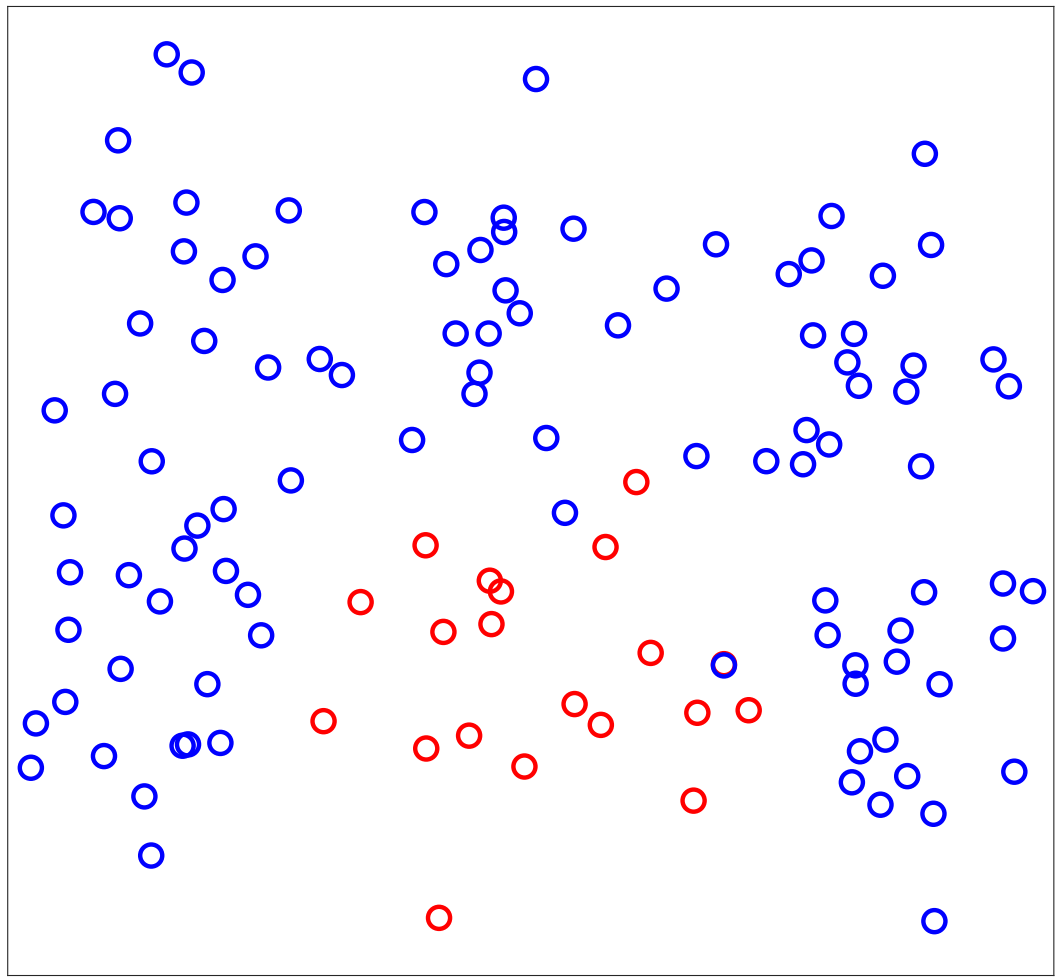
K nearest neighbours (kNN) is a simple extension of nearest neighbours that proceeds as follows. Given a new sample $\boldsymbol{x}$:

- We calculate the distance to all the training samples $\boldsymbol{x}_i$.
- Extract the K closest samples (neighbours).
- Obtain the number of neighbours that belong to each class.
- Assign the label of the most popular class among the neighbours.

Boundaries in kNN classifiers



Boundaries in kNN classifiers



k Nearest Neighbours

Note that:

- There is always an implicit boundary, although it is not used to classify new samples.
- As K increases, the boundary becomes less complex. We move away from **overfitting** (small K) to **underfitting** (large K) classifiers.
- In binary problems, the value of K is usually an odd number. The idea is to prevent situations where half of the nearest neighbours of a sample belong to each class.
- kNN can be easily implemented in multi-class scenarios.

Agenda

Intro

Formulating classification problems

Linear classifiers

Logistic model

Nearest neighbours

Summary

Machine learning classifiers

- Classifiers are partitions of the predictor space into **decision regions** separated by **boundaries**.
- Each decision region is associated with one label.
- In machine learning, classifiers are built using a **dataset** (otherwise it's not machine learning!).

Flexibility and complexity in classifiers

- The notions of flexibility, complexity, interpretability, overfitting and underfitting also apply to classifiers.
- Linear boundaries are simple and rigid; kNN produces boundaries whose complexity depends on the value of K.
- Logistic regression is a strategy to train linear classifiers. It's called *regression* because indirectly we solve a regression problem or the classifier's certainty.
- Weirdly, kNN does not involve training as it uses all the samples each time a new sample is to be classified.

Hey, hold on a second, what's going on?

- In machine learning we use a quality metric to define what we mean by the *best* model.
- We have presented two quality metrics: **accuracy** and **error rate**.
- However, **neither** the logistic regression nor kNN classifiers **use the notion of accuracy**.
- What's going on?