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

ECS766 Data Mining
Week 3: Classification I

Dr Jesús Requena Carrión

9 Oct 2019

Agenda

Recap (with some extras)

Formulation of classification problems

Linear classifiers

Logistic regression

Nearest neighbours

Flexibility, complexity and overfitting

The starting point of data science projects is to assume that 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**, however we need to be careful not to learn irrelevant noise features, i.e. not to **overfit** our data.

By **testing** our model on a dataset different from the dataset used for **training**, we can assess how well we have been able to capture the pattern and discard the noise. **Validation** approaches use separate datasets as well.

Ventris' ultimate check

"... and a decisive check, preferably with the aid of **virgin material**, to ensure that the apparent results are not due to **fantasy**, **coincidence** or **circular reasoning**"

Don't fool yourself!

Agenda

Recap (with some extras)

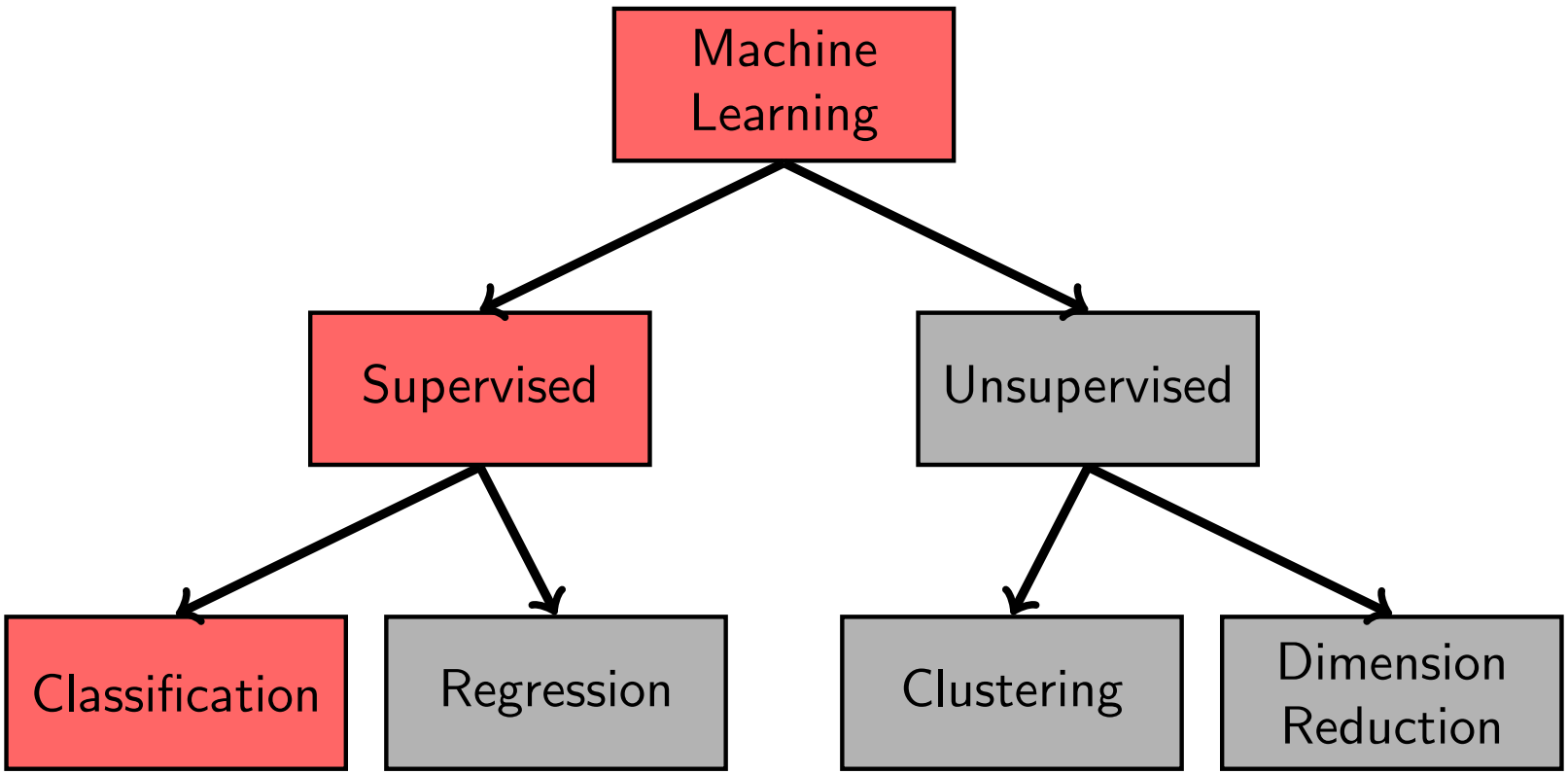
Formulation of classification problems

Linear classifiers

Logistic regression

Nearest neighbours

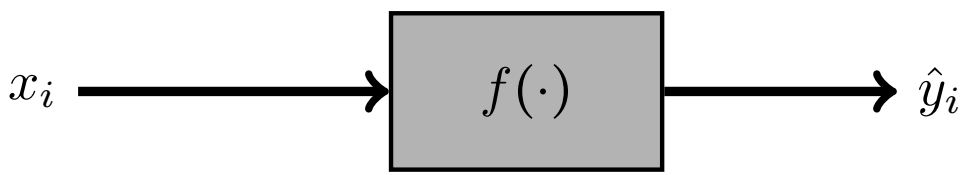
Data science taxonomy



Problem formulation

We will follow the same notation that we used for regression problems. In a classification problem:

- We assume the value of one of the attributes can be predicted based on the value of the remaining attributes (it is a **supervised** problem).
- The label y is a **discrete** (categorical/qualitative) **variable**. Its values are called **classes** and we say sample x belongs to class y.
- As usual, our job is then to **find the** *best* **model** $f(\cdot)$ that relates the response attribute to the predictors, $\hat{y} = f(x)$ and we will use a **dataset** $\{(x_i, y_i) : 1 \leq i \leq N\}$ to infer the relationship between response and predictors.



A binary classification problem

There are many examples of classifiers that aim at predicting whether **individuals have or lack a specific characteristic or behavior** based on some data about them (for instance whether they have a certain disease based on medical data).

A popular dataset for binary classification in machine learning is the *Adult Data Set*, built from anonymised US census data:

- It contains nearly 50,000 samples (i.e. individuals)
- A total of **14 categorical and numerical attributes** (including age, workclass, education and gender) can be used as **predictors**
- A **binary label** denoting whether the instance's salary is greater or less than \$50K
- Downloadable from archive.ics.uci.edu/ml/datasets/Adult

Another 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 create 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** (i.e. a vector of categorical variables) used as a **predictor** together with 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 $0, 1, 2, \ldots 9$.

Machine learning approaches use datasets of **labelled images**, i.e. **images (predictors)** with the corresponding **digit (label)**, to build a model.

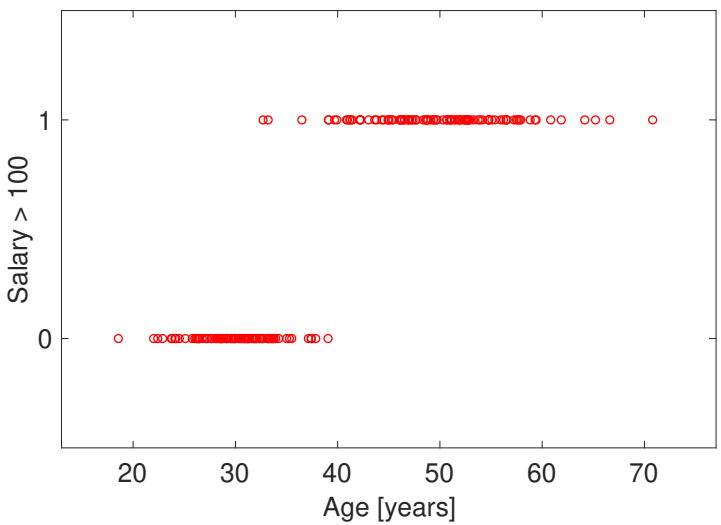
The MNIST dataset is a collection of handwritten digits:

- 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 as a point cloud

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

One predictor, two classes

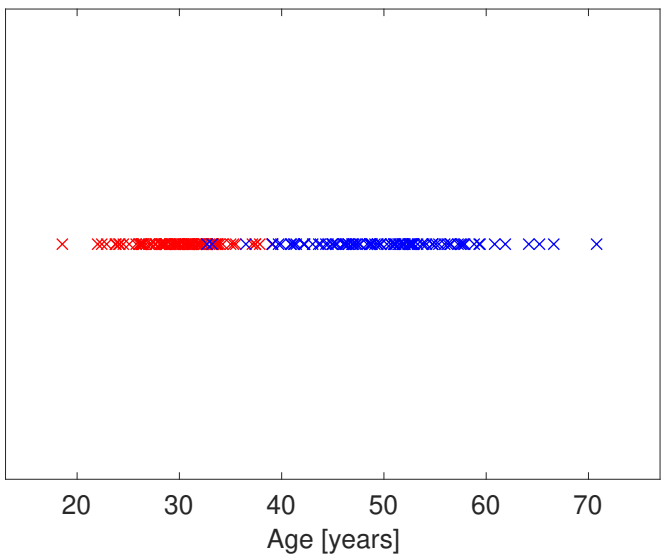


Two predictors, three classes

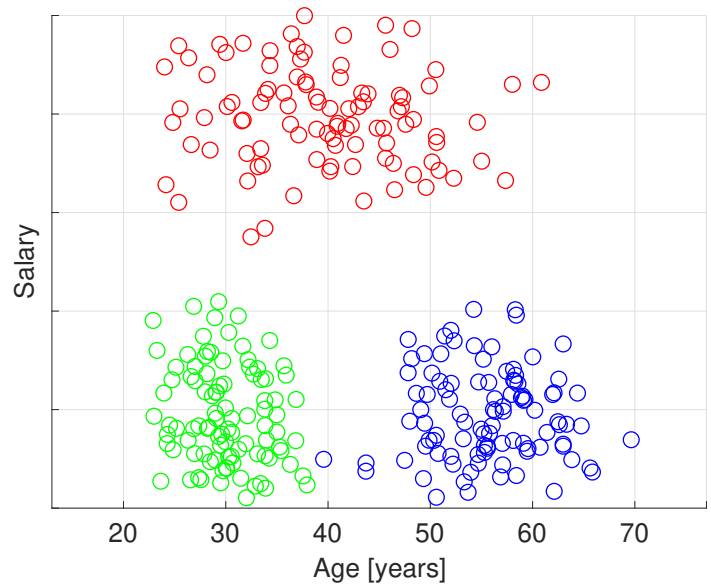
The dataset as a point cloud

Datasets can also be represented as collections of points in the **predictor space** that use **different symbols for each label**.

One predictor, two classes



Two predictors, three classes

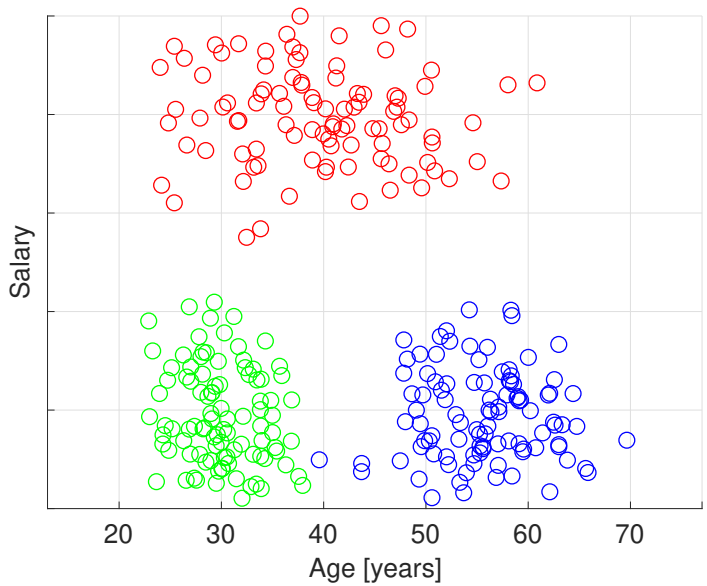


What does a solution look like?

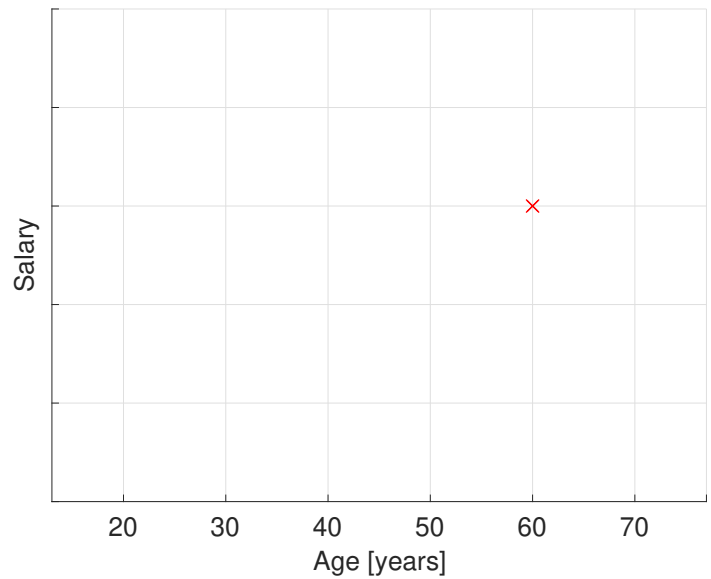
Regression models can be represented as curves/surfaces/hypersurfaces, such that for every predictor value a single response is returned.

How can we represent a classification model?

Training data



New data point



What does a solution look like?

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

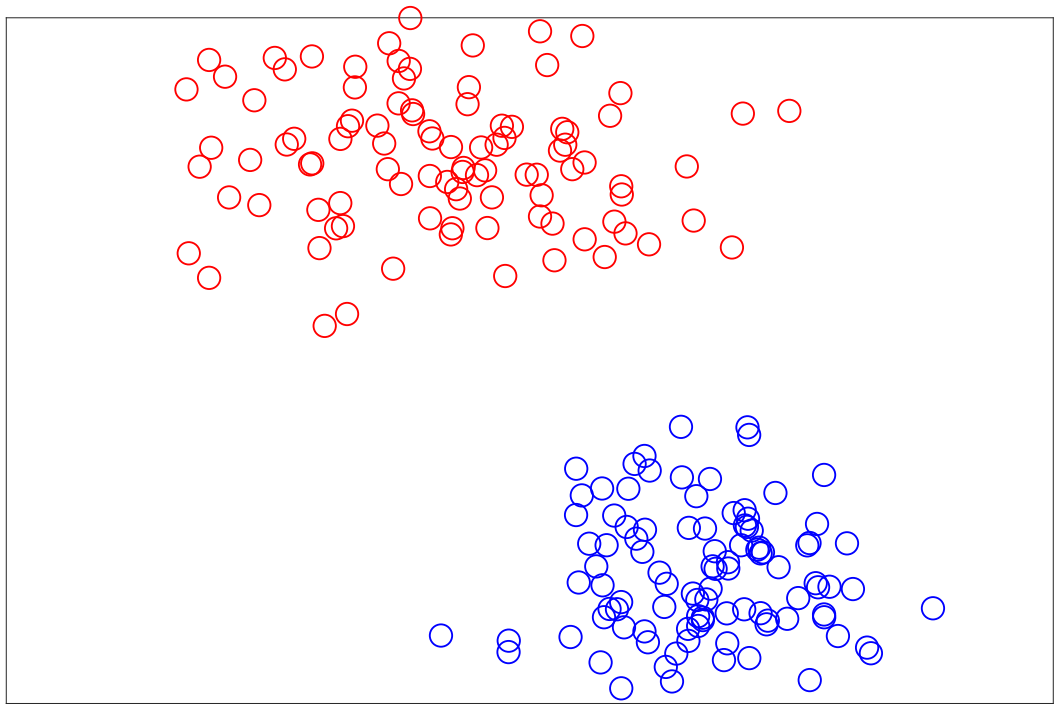
- A decision region in the predictor space corresponds to all the **points that are associated to the same label**.
- Regions can be defined by identifying their **boundaries**.
- Therefore, a solution model in classification is **defined by a set of regions or boundaries between regions**.

Now that we know what a solution looks like, we need to answer two questions:

- How do we define a sensible **goodness of fit** to identify the *best* model?
- How can we express our solution as a **computation**?

Overfitting in classifiers

The notions of flexibility, complexity and overfitting can also be applied to classifiers and are best understood by looking at the boundaries defining the decision regions.



Model quality

In regression, we defined quality metrics that used the **prediction error** $e_i = y_i - \hat{y}_i$ (**distance** between desired and predicted labels).

The MNIST dataset is a collection of images containing handwritten digits. Data scientists use the MNIST dataset to build models that can recognise the digits $0, 1, 2, \ldots 9$ from handwritten digits.

To build a model that recognises digits from their handwritten versions, the notion of distance between a desired response y_i and the predicted one $\hat{y}_i$:

(a) Can be defined as the subtraction $y_i - \hat{y}_i$

(b) Can be defined as the normalised subtraction $(y_i - \hat{y}_i)/10$

(c) Cannot be defined

Agenda

Recap (with some extras)

Formulation of classification problems

Linear classifiers

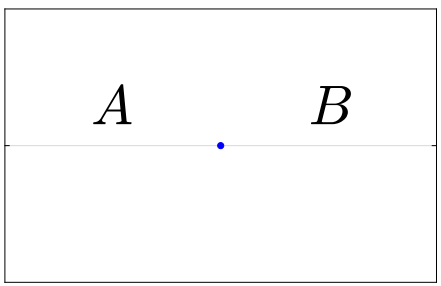
Logistic regression

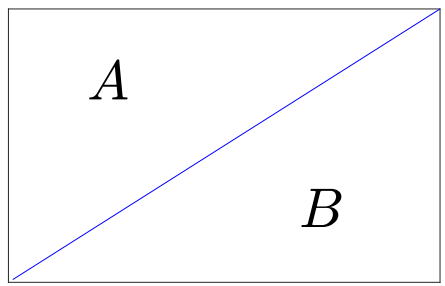
Nearest neighbours

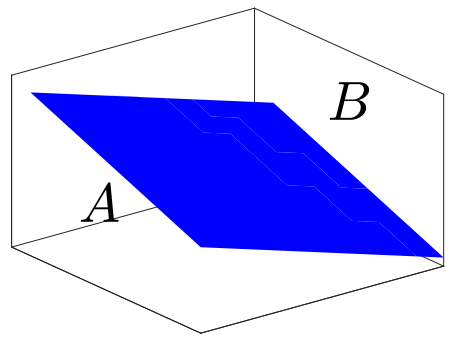
The simplest boundary

Let's consider a **binary** classification problem (i.e. two labels, for instance A and B). The simplest boundary between the two decision regions is:

- A point for one predictor.
- A straight line for 2 predictors.
- A plane surface for 3 predictors.







Definition of linear classifiers

Linear classifiers define straight lines, planes and hyperplanes as boundaries between decision regions:

- Linear boundaries are defined by the linear equation $\boldsymbol{w}^T\boldsymbol{x} = 0$. The extended vector $\boldsymbol{x} = [1, x_\alpha, x_\beta ...]^T$ contains the predictor variables
- The **idea** is to classify a sample based on whether it's on one side of the boundary or the other
- The **implementation** is simple. Given sample i's predictors, build the extended vector $\boldsymbol{x}_i$. Then carry out the operation $\boldsymbol{w}^T\boldsymbol{x}_i$. If the results is positive, you are on one side of the boundary. If it is negative, you are on the other
- The question we still need to answer is: What is the **best linear classifier** for a given dataset? In other words, what is the best $\boldsymbol{w}$?

Goodness of fit

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

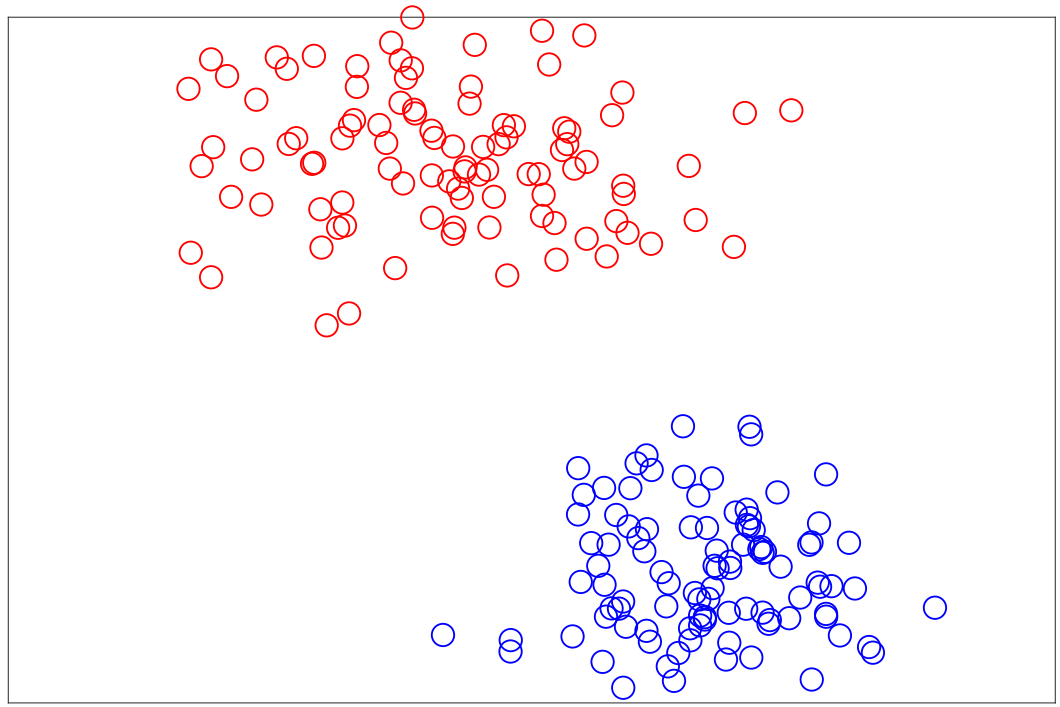
In a dataset with N_A instances labeled A and N_B instances labelled B:

- We can calculate the number of **true predictions** for a class (e.g. samples with label A that have been classified as A)
- We can calculate the number of **false predictions** for a class (e.g. samples with label B that have been classified as A)
- We can calculate the overall number of true and false predictions

A useful notion of goodness is the **accuracy** A, which is defined as the number correctly classified samples over the total number of samples. The **best linear classifier can be defined as the one with the highest accuracy**.

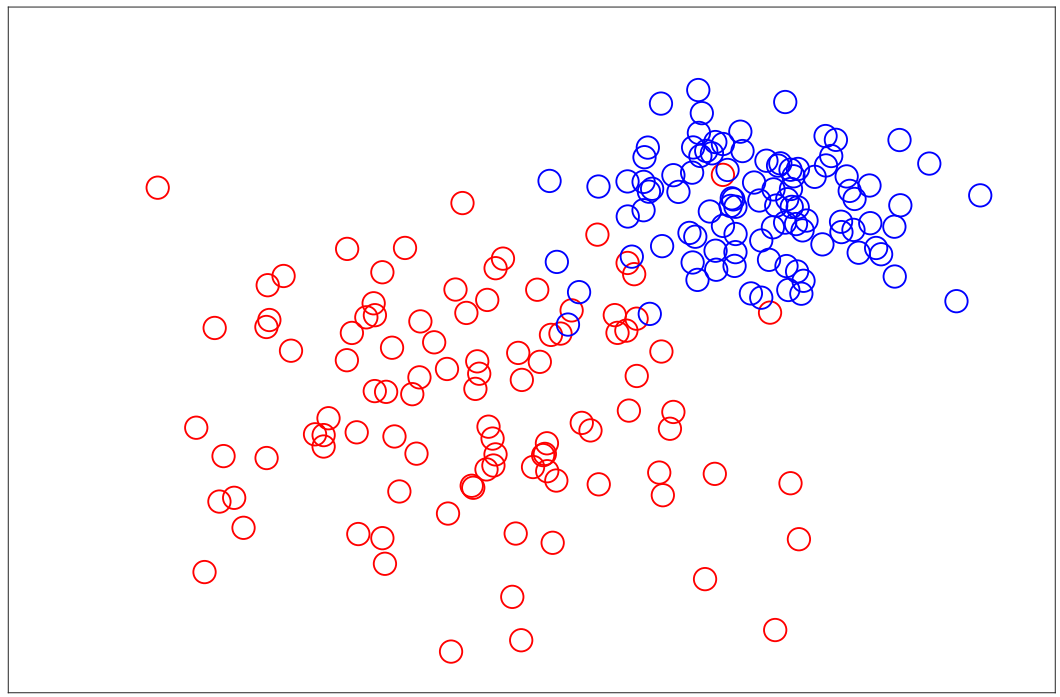
The best linear classifier: Separable case

When dealing with linearly separable datasets, we can achieve the maximum accuracy $A = 1$. Notice there can be multiple *best* solutions.



The best linear classifier: Non separable case

In the case of non linearly-separable datasets, the accuracy is always $A < 1$. The best solution will be the one(s) achieving the highest accuracy.



Agenda

Recap (with some extras)

Formulation of classification problems

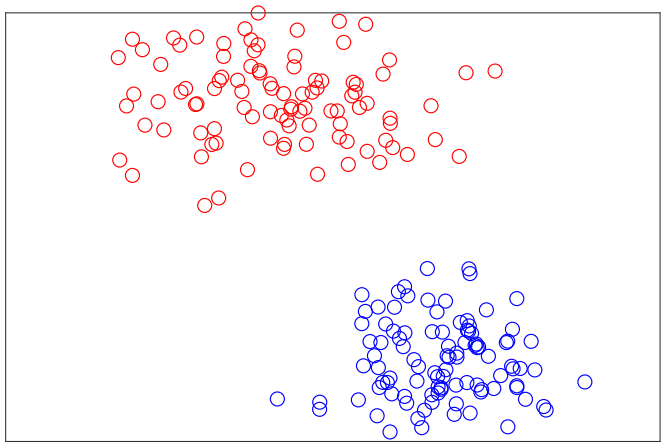
Linear classifiers

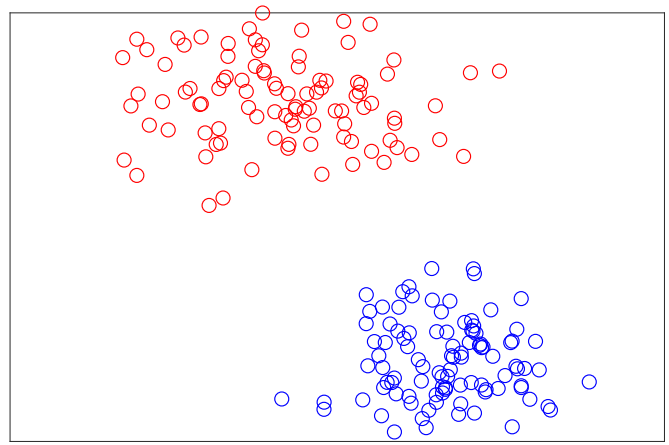
Logistic regression

Nearest neighbours

Best, but risky, linear solutions

Draw two linear classifiers (i.e. two linear boundaries) for the same dataset that achieve an accuracy $A = 1$. Which one would you choose? Why?

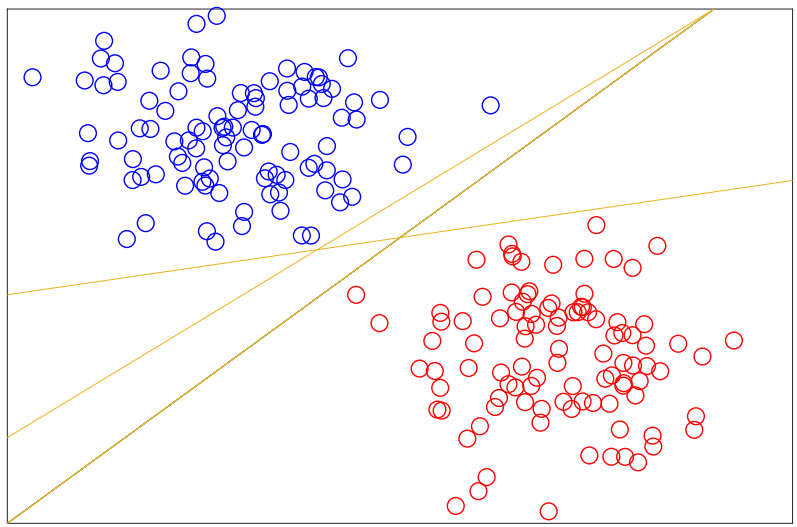




If you prefer one over the other, you might be inadvertently assessing their generalisation ability.

Keep that boundary away from me!

As we get closer to the boundary between two decision regions, life gets harder for a classifier: it is **noise territory**. Our uncertainty about the true identity of a sample there would be greater than far from the boundary.



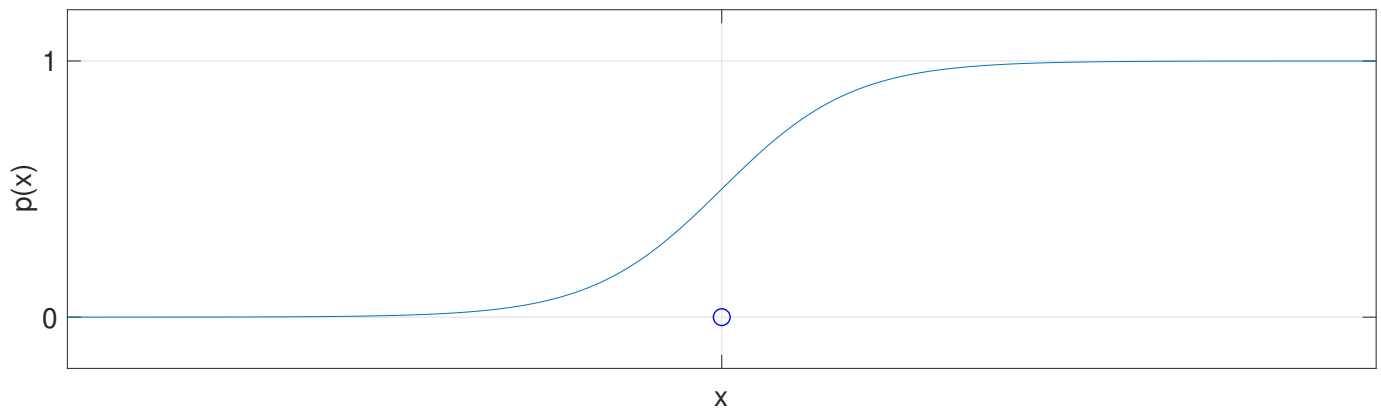
In other words, **the further we are from the boundary, the higher the chances we will be classifying our samples correctly**.

The logistic model

Given a linear boundary defined by $\boldsymbol{w}^T\boldsymbol{x} = 0$ and a sample's predictor vector $\boldsymbol{x}_i$, the logistic function is defined as

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

In the case of one single predictor variable and a boundary at 0, the logistic function looks like this



The logistic model

Remember that $\boldsymbol{w}^T\boldsymbol{x}_i$ can be interpreted as a distance from a point in the predictor space to the boundary. This distance is negative if $\boldsymbol{x}_i$ is in one region (say A), positive if it's in the other region (say B).

Notice that:

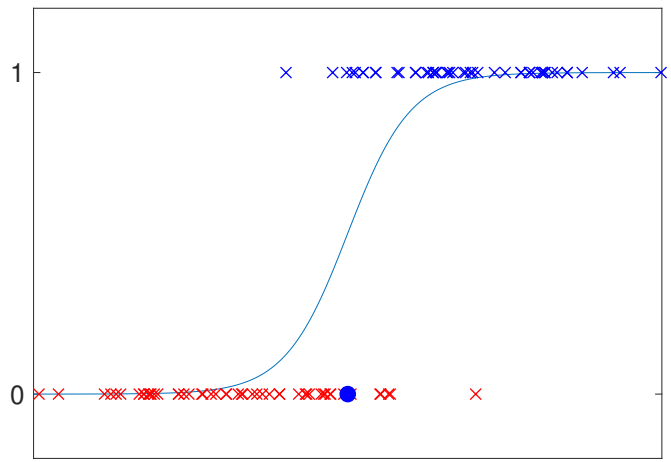
- If $\boldsymbol{x}_i$ is on the boundary, $p(\boldsymbol{x}_i) = 0.5$.
- If $\boldsymbol{w}^T\boldsymbol{x}_i > 0$ (we are in region B), as we move away from the boundary $p(\boldsymbol{x}_i) \to 1$.
- If $\boldsymbol{w}^T\boldsymbol{x}_i < 0$ (we are in region A)as we move away from the boundary,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

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

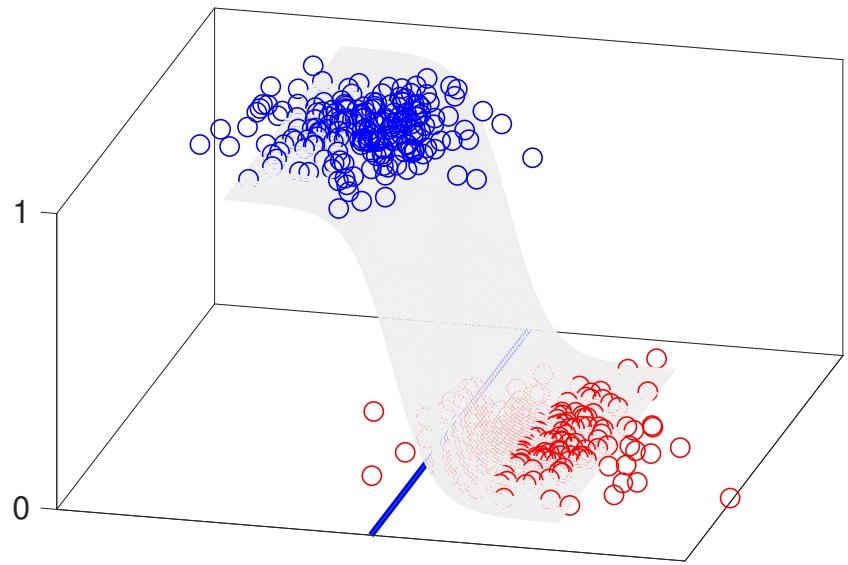
- If $y_i = B$, then $p(\boldsymbol{x}_i)$ is the **classifier's certainty** that $y_i = B$.
- If $y_i = A$, then $1 - p(\boldsymbol{x}_i)$ is the **classifier's certainty** that $y_i = A$.

Visualising logistic regression

One predictor



Two predictors



The logistic model

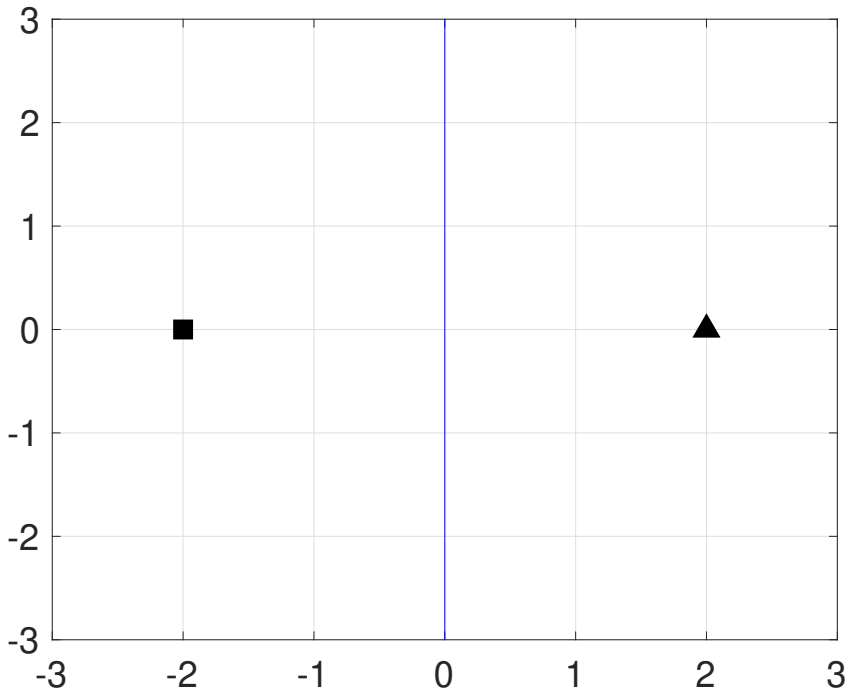
Given a linear boundary and a sample, we know how to calculate the classifier's certainty that the sample belongs to either class. Can we calculate the classifier's certainty for a whole dataset $\{(\boldsymbol{x}_i, y_i)\}$?

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

$$L(\boldsymbol{w}) = \prod_{y_i=A} (1 - p(\boldsymbol{x}_i)) \prod_{y_i=B} p(\boldsymbol{x}_i)$$

The classifier's certainty $L(\boldsymbol{w})$ is known as the **likelihood function** and the classifier that maximises it is the **Maximum Likelihood**, **MaxEnt** or simply **Logistic Regression** classifier. (Did I say regression?)

Example I



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

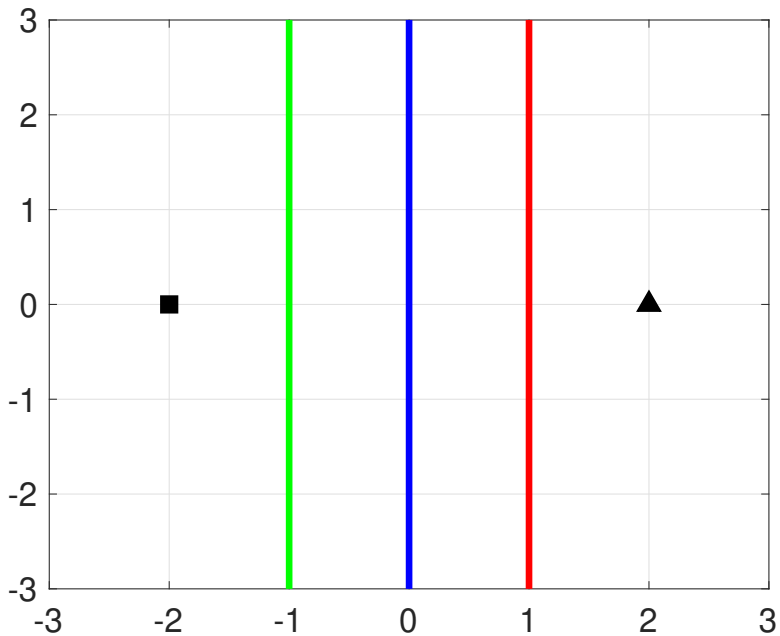
$$p(d_i) = \frac{e^{d_i}}{1 + e^{d_i}}$$

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

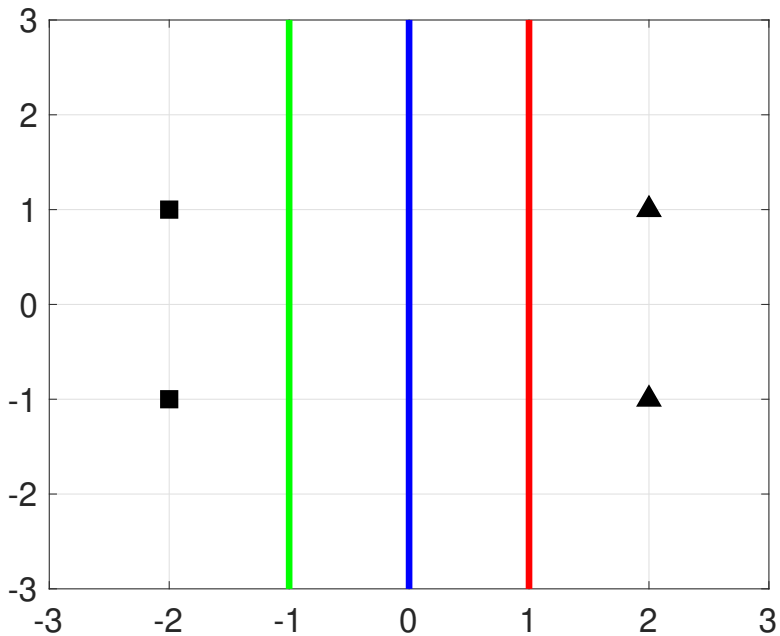
Example II



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

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

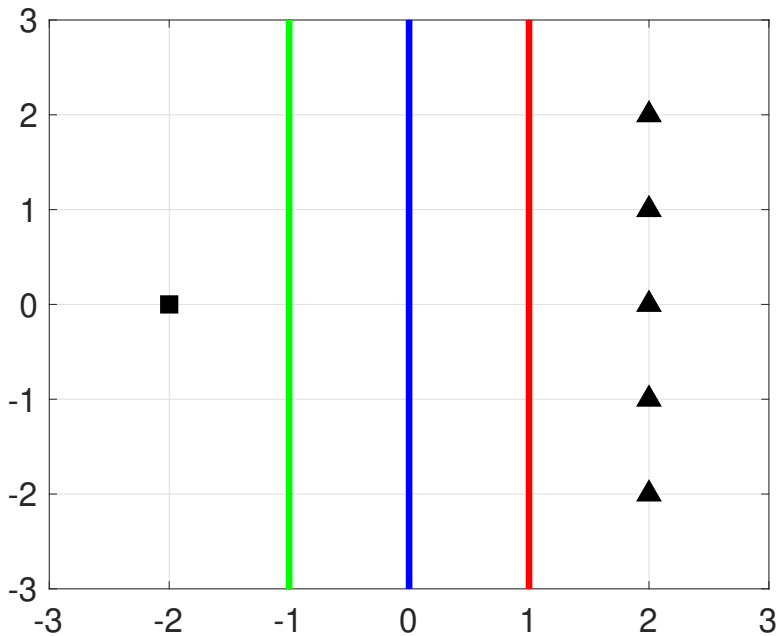
Example III



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:

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

Final thoughts about linear logistic regression

- How do we find the Maximum Likelihood classifier?
- What should we do if our classes are not linearly separable?
- How do we generalise to multiclass classification?
- What is the effect of outliers?
- What is the meaning of the weights $\boldsymbol{w}$?
- What if one of our predictors is a categorical variable?

Agenda

Recap (with some extras)

Formulation of classification problems

Linear classifiers

Logistic regression

Nearest neighbours

Parametric and non-parametric approaches

Linear classifiers belong to the family of **parametric** approaches : we assume a type of boundary (in this case linear) and use our dataset to identify the best boundary (defined by the vector $\boldsymbol{w}$) within the assumed type of boundaries.

Non-parametric approaches offer a more flexible alternative, as they do not assume any type of boundary. In this section, we will study two popular approaches:

- Nearest Neighbours (NN)
- K Nearest Neighbours (kNN)

Nearest Neighbours

A classifier is essentially a partition of the predictor space into decision regions separated by boundaries:

- Given a new sample x_i, its label y_i is determined by identifying the decision region where it is.

In nearest neighbours, boundaries in binary classifiers are defined as points that are half-way between two samples from different classes:

- To classify a new sample, we just need use the **label of the closest (*most similar*) training sample**.
- We need to **memorise the whole training dataset** (linear classifiers only needed to memorise the vector $\boldsymbol{w}$). That's why sometimes we call them **instance-based methods**.

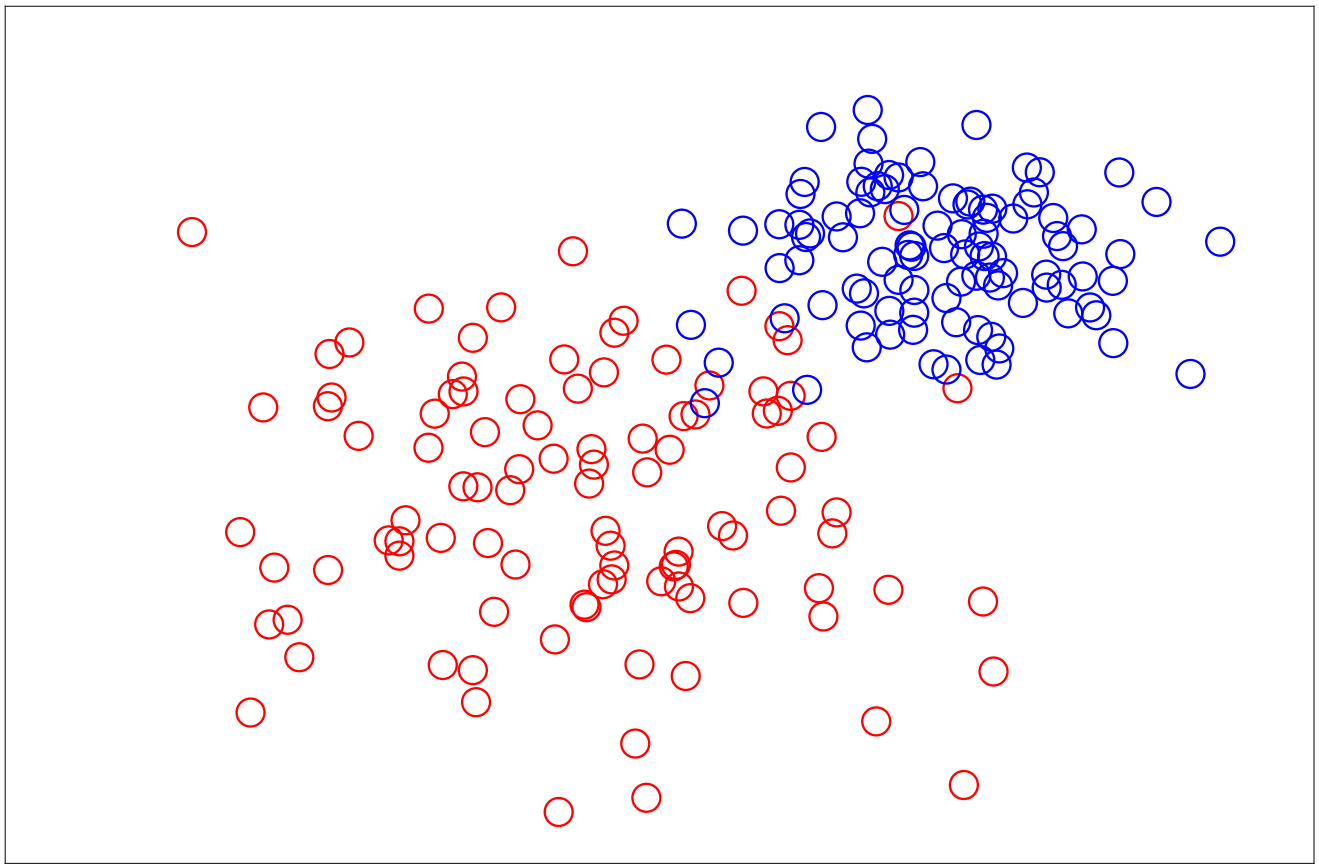
The notion of distance

Mathematically, we can use different *distances*:

- Euclidean
$$D_E(\boldsymbol{a}, \boldsymbol{b}) = \sum_k (a_k - b_k)^2$$
- Manhattan
$$D_M(\boldsymbol{a}, \boldsymbol{b}) = |a_k - b_k|$$
- Hamming
$$D_H(\boldsymbol{a}, \boldsymbol{b}) = \sum_k (a_k \neq b_k)$$

The shape of the final boundary depends on the definition of distance.
When is each distance appropriate?

Boundaries in Nearest Neighbours classifiers



k Nearest Neighbours

Boundaries in nearest neighbours classifiers can be too complicated, noisy and hard to interpret. How could we smooth them? K Nearest neighbours proceeds as follows. Given a new sample x:

- We calculate the distance to all the training samples.
- Choose the K closest points
- Note the class of each of the K points
- Assign a label based on the most popular class among those k points.

k Nearest neighbours: Example

Final thoughts about Nearest Neighbors

- How do we choose K?
- A predictor could have different units (for instance cm, m, km). Which unit should we use?
- Different predictors have different units. What is the meaning of a distance in the predictor space?