Linear classifiers

Week 3 - Lecture 6

Dr Matthew Ellis - COM2004/3004 Data Driven Computing 23/24

Goals of today's session

By the end of today's session you should be able to:

1. Calculate the probability of an error for a probabilistic classifier.
2. Define risk and loss, and apply the likelihood ratio criteria.
3. Explain what is meant by a linear classifier and linear decision boundary.
4. Analyse Bayesian classifiers to determine their decision boundary.
5. Define a linear classifier and understand the weight vector.

Lecture 5 Recap

Distance Measures

Euclidean distance

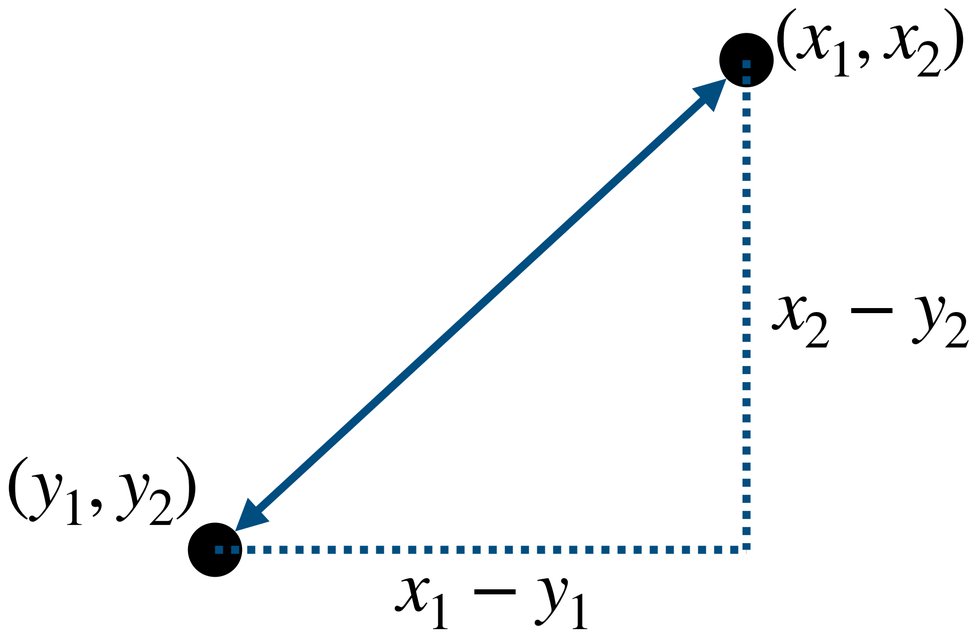
Consider two points on a 2-D plane.

From Pythagoras' theorem we can easily calculate the distance as:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

Or more compactly:

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{2} (x_i - y_i)^2 \right)^{\frac{1}{2}}$$



N-dimensional Euclidean distance

This distance can easily be generalised into a higher dimensional space. If we have two points $\mathbf{x} = \left(x_1, x_2, \cdots, x_N\right)^T$ and $\mathbf{y} = \left(y_1, y_2, \cdots, y_N\right)^T$

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{N} (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

The magnitude of the vector between the two points, forming the shortest ‘straight line’ distance between them.

l^p-norm distances

The Euclidean distance is a special case of a family called l^p-norm distances. It was the square root of the sum of squares:

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{N} (x_i - y_i)^2 \right)^{\frac{1}{2}}$$

To generalise we can change the $()^2$ and $()^{\frac{1}{2}}$ to $()^p$ and $()^{\frac{1}{p}}$, giving the l^p-norm

$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{N} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

But note that we are now taking the **absolute values** of each term i.e $|x_i - y_i|$

l^p-norm notation

The l^p-norm distance is usually written as

$$\|\mathbf{x}-\mathbf{y}\|_p = \left(\sum_{i=1}^{N} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

e.g the Euclidean norm would be

$$\|\mathbf{x}-\mathbf{y}\|_2 = \left(\sum_{i=1}^{N} |x_i - y_i|^2\right)^{\frac{1}{2}}$$

When $p = 2$ the subscript is sometimes dropped, e.g you see $\|\mathbf{x}-\mathbf{y}\|$ then this tends to mean the Euclidean distance.

Euclidean vs Manhattan distance

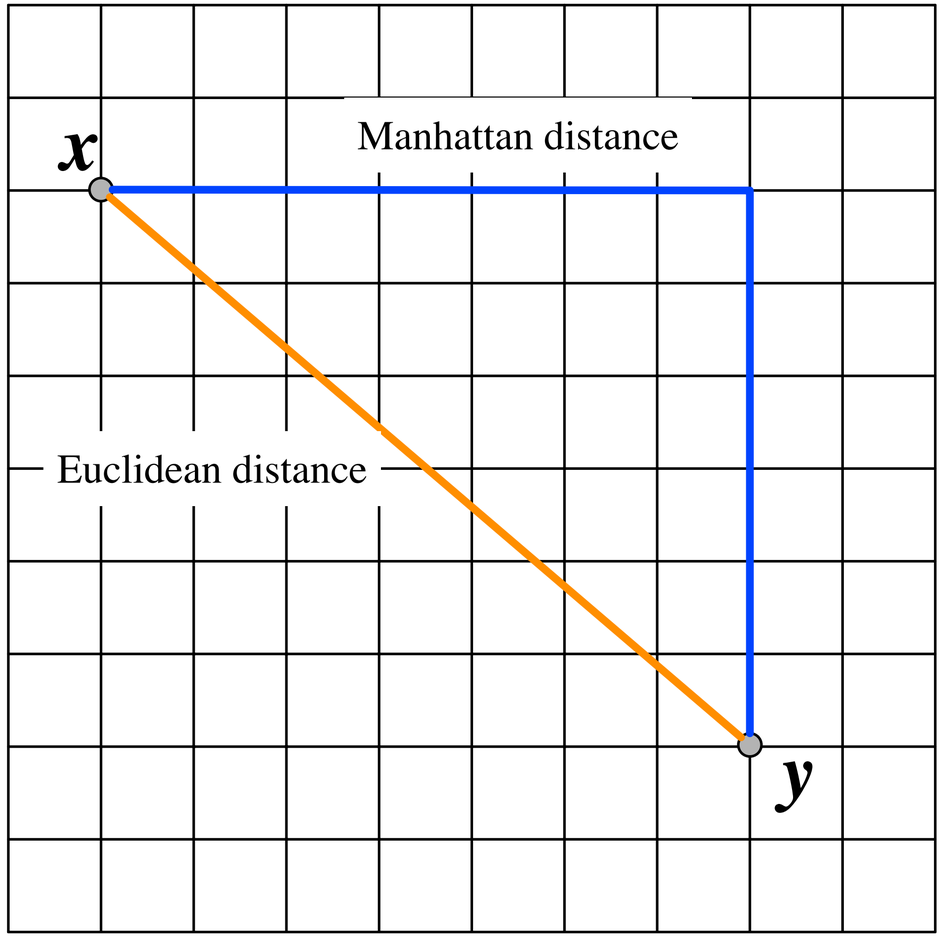
The l^1 and l^2 norms are common but quite different.

l^1-norm (Manhattan) distance:

$$\|\mathbf{x} - \mathbf{y}\|_1 = \sum_{i=1}^{N} |x_i - y_i|$$

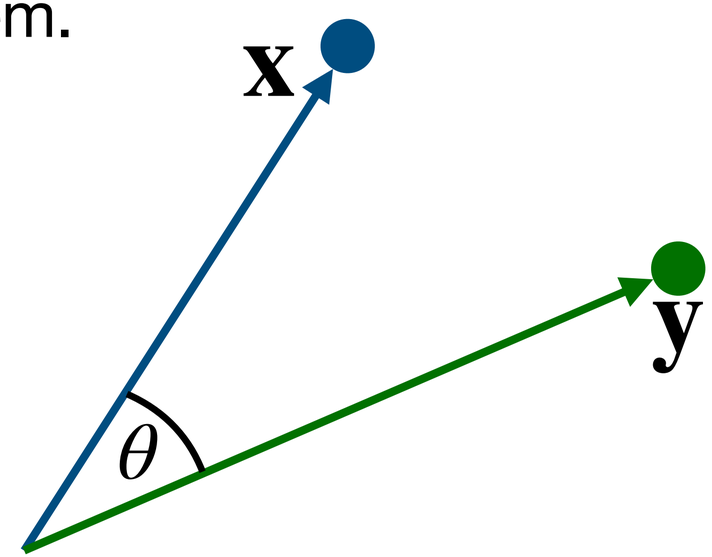
l^2-norm (Euclidean) distance:

$$\|\mathbf{x} - \mathbf{y}\|_2 = \left(\sum_{i=1}^{N} |x_i - y_i|^2 \right)^{\frac{1}{2}}$$



Cosine distance

A completely different way of measuring a distance between two points would be to consider the angle, θ, between them.



This can be calculated using

$$\cos(\theta) = \left(\frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|} \right)$$

Is this a good dissimilarity measure?

Instead we can define the cosine distance as

$$d_{\text{cosine}} = 1 - \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}||\mathbf{y}|}$$

Cosine vs Euclidean distance

Why might the cosine distance be useful?

It is not affected by scale (scale invariant):

$$d_{\text{cosine}}\left(a\mathbf{x}, b\mathbf{y}\right) = d_{\text{cosine}}\left(\mathbf{x}, \mathbf{y}\right) \text{ for all } a, b > 0$$

This scale invariance is often useful. E.g imagine comparing two text documents about football. We could extract the word frequencies of ‘Sheffield United’ and ‘Sheffield Wednesday’. The first document mentions both many times but the other only once each. Both are on similar topics but the Euclidean distance would be high compared the the cosine.

Summary

Distance measures take a pair of points (i.e., vectors) and return a scalar distance between them.

Distance measures are '**dissimilarity measures**' but we can also define '**similarity measures**'.

Some basic properties that need to be obeyed. No point can be closer than a point is to itself!

Commonly used examples include **Euclidean distance**, l^p**-norm distances**, **cosine distances**.

The appropriate distance to use will depend on the application. We will see examples throughout the module.

Minimising risk

Recap - Bayes classification rule

Our decision rules states $\mathbf{x}$ belongs to ω_1 if

$$P(\omega_1 \,|\, \mathbf{x}) > P(\omega_2 \,|\, \mathbf{x}).$$

Using Bayes' rule, this criterion can be expressed as

$$p(\mathbf{x} \,|\, \omega_1)P(\omega_1) > p(\mathbf{x} \,|\, \omega_2)P(\omega_2).$$

If the priors are equal $P(\omega_1) = P(\omega_2)$ this reduces to

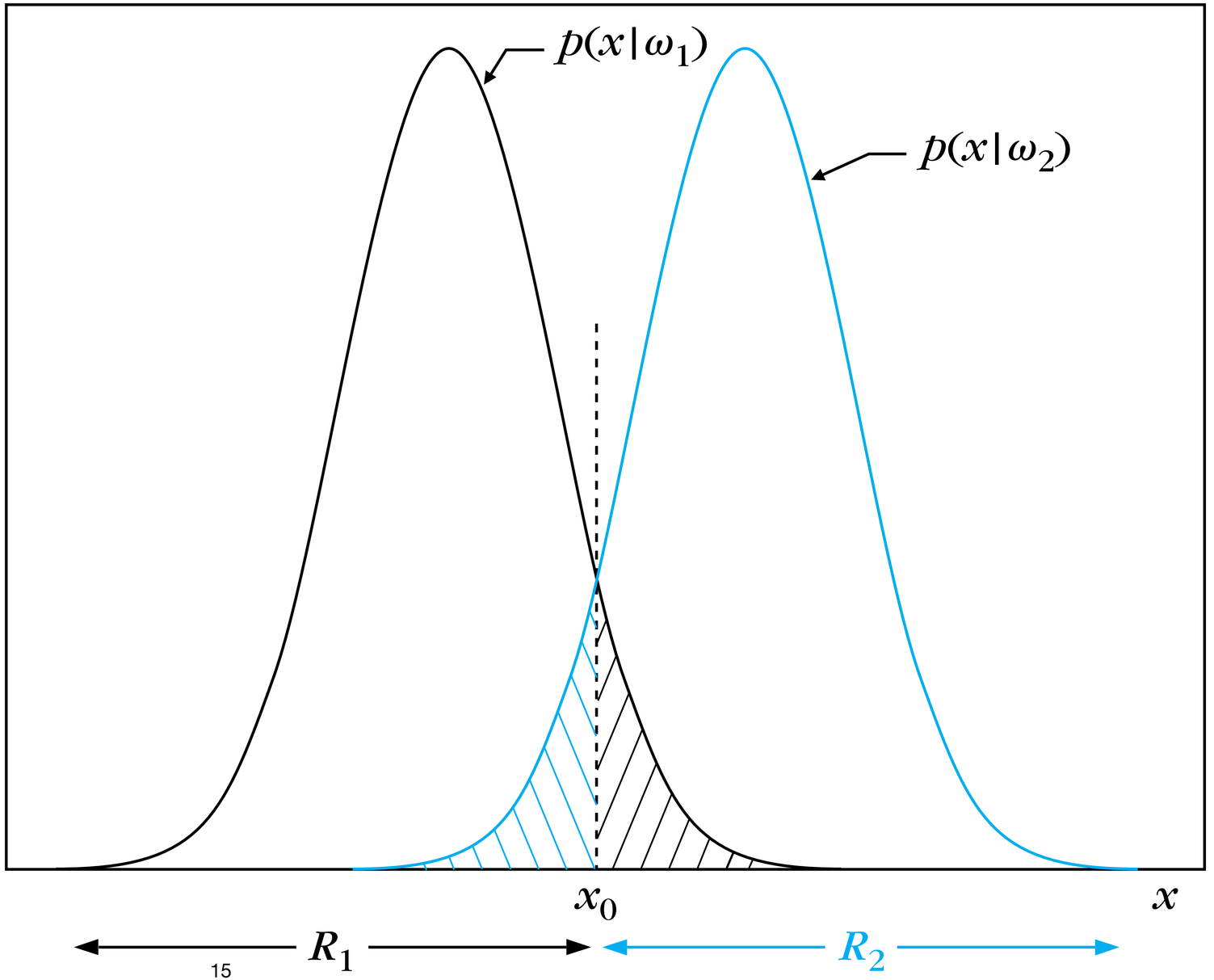
$$p(\mathbf{x} \,|\, \omega_1) > p(\mathbf{x} \,|\, \omega_2)$$

Gaussian pdfs

Case with equal priors.

$\mathbf{x}$ belongs to ω_1 if $\mathbf{x} \in R_1$

$\mathbf{x}$ belongs to ω_2 if $\mathbf{x} \in R_2$



Probability of an error

An error will occur where the pdfs overlap. We can define the probability of an error as:

$$P_e = \int_{-\infty}^{x_0} p(\mathbf{x}, \omega_2) d\mathbf{x} + \int_{x_0}^{\infty} p(\mathbf{x}, \omega_1) d\mathbf{x}$$

$$= \int_{-\infty}^{x_0} p(\mathbf{x} \mid \omega_2) P(\omega_2) d\mathbf{x} + \int_{x_0}^{\infty} p(\mathbf{x} \mid \omega_1) P(\omega_1) d\mathbf{x}$$

$\int_{-\infty}^{x_0} p(\mathbf{x}, \omega_2) d\mathbf{x}$ is the probability that $\mathbf{x}$ belongs to ω_2 but falls in region R_1.

Bayes classifier is optimal if we wish to minimise the number of classification errors.

Risk and loss

With two classes there are two types of errors, the cost of which may not be equal.

The loss, λ_{ij}, is the cost of misclassifying something from class i as belonging to class j.

Consider the losses λ_{12} and λ_{21} in the following scenarios:

- An airport scanning machine looking for guns.
- Computer vision system screening cells for cancer.
- Sexing day-old chicks at a poultry farm.

Risk

The **probability of an error** was

$$P_e = P(\mathbf{x} \in R_1, \omega_2) + P(\mathbf{x} \in R_2, \omega_1)$$

The **average risk** is defined as

$$r = \lambda_{21} P(\mathbf{x} \in R_1, \omega_2) + \lambda_{12} P(\mathbf{x} \in R_2, \omega_1)$$

Note: when the loss is equal, the risk is the same as the probability of an error. So in the ‘default’ case, minimising error also minimises the risk.

Risk

What about the cost of correct classifications?

The **average risk** - including correct classifications

$$r = \lambda_{11} P(\mathbf{x} \in R_1, \omega_1) + \lambda_{12} P(\mathbf{x} \in R_2, \omega_1)$$
$$+ \lambda_{21} P(\mathbf{x} \in R_1, \omega_2) + \lambda_{22} P(\mathbf{x} \in R_2, \omega_2)$$

Risk

Example: A surgeon for two classes:

Healthy people (ω_1) and tumour patients (ω_2). Failure to detect a tumour patient causes four times as much loss as incorrectly evaluating a healthy person as having the fatal disease. Note, even correct detection of a tumour patient causes some loss (i.e. cost).

Correct identification of a healthy person does not cause any loss.

$$\rightarrow \lambda_{11} = 0, \quad \lambda_{12} = \lambda, \quad \lambda_{12} = 4\lambda, \quad \lambda_{12} = 2\lambda$$

Risk

General formula for M classes:

Risk associated with class ω_k:

$$r_k = \sum_{i=1}^{M} \lambda_{ki} \int_{R_i} p(\mathbf{x} \mid \omega_k) d\mathbf{x}$$

Risk averaged over all classes:

$$r = \sum_{k=1}^{M} r_k P(\omega_k)$$

$$= \sum_{i=1}^{M} \int_{R_i} \underbrace{\left(\sum_{k=1}^{M} \lambda_{ki} p(\mathbf{x} \mid \omega_k) P(\omega_k) \right)}_{l_i} d\mathbf{x}$$

Risk

The average risk is minimised by selecting partitioning regions so that

$$\mathbf{x} \in R_i \quad \text{if} \quad l_i < l_j \quad \forall j \neq i$$

Two class case:

$$l_1 = \lambda_{11} p(\mathbf{x} \,|\, \omega_1) P(\omega_1) + \lambda_{12} p(\mathbf{x} \,|\, \omega_2) P(\omega_2)$$

$$l_2 = \lambda_{21} p(\mathbf{x} \,|\, \omega_1) P(\omega_1) + \lambda_{22} p(\mathbf{x} \,|\, \omega_2) P(\omega_2)$$

Assign $\mathbf{x}$ to class ω_1 is $l_1 < l_2$ I.e

$$(\lambda_{21} - \lambda_{22}) p(\mathbf{x} \,|\, \omega_2) P(\omega_2) < (\lambda_{12} - \lambda_{11}) p(\mathbf{x} \,|\, \omega_1) P(\omega_1)$$

Assuming $\lambda_{ij} > \lambda_{ii}$, we have the **likelihood ratio test**:

$$\frac{\lambda_{21} - \lambda_{22}}{\lambda_{12} - \lambda_{11}} \frac{P(\omega_2)}{P(\omega_2)} < \frac{p(\mathbf{x} \,|\, \omega_1)}{p(\mathbf{x} \,|\, \omega_2)}$$

Exercise

A single feature x with the pdfs:

$$p(x\,|\,\omega_1) = \frac{1}{\sqrt{\pi}} \exp\left(-x^2\right) \text{ and } p(x\,|\,\omega_2) = \frac{1}{\sqrt{\pi}} \exp\left(-(x-1)^2\right)$$

No prior information so $P(\omega_1) = P(\omega_2) = \frac{1}{2}$

Assume the losses are $\lambda_{11} = \lambda_{22} = 0$, $\lambda_{12} = \frac{1}{2}$ and $\lambda_{21} = 1$

1) What is the value x_0 for the classification boundary to minimise the classification error?

2) What is would x_0 be to minimise the risk?

Summary

Different types of classification error result in different **losses**:

e.g. classifying an ω_1 object as ω_2 may result in more loss than vice versa

Risk is the loss associated with error multiplied by the probability of the error. We generally want to **minimise risk.**

If all errors have equal cost then we minimise risk simply by minimising errors

In other cases the decision boundary will be shifted to reduce the number of the more costly errors at the expense of a greater number of cheaper errors

e.g. reducing the number of armed terrorists on planes at the expense of hand-searching many more bags of innocent airline passengers

For the two-class case the we can plug the losses (λ) into the **likelihood ratio test** to see how the decision boundary is changed.

Linear classifier

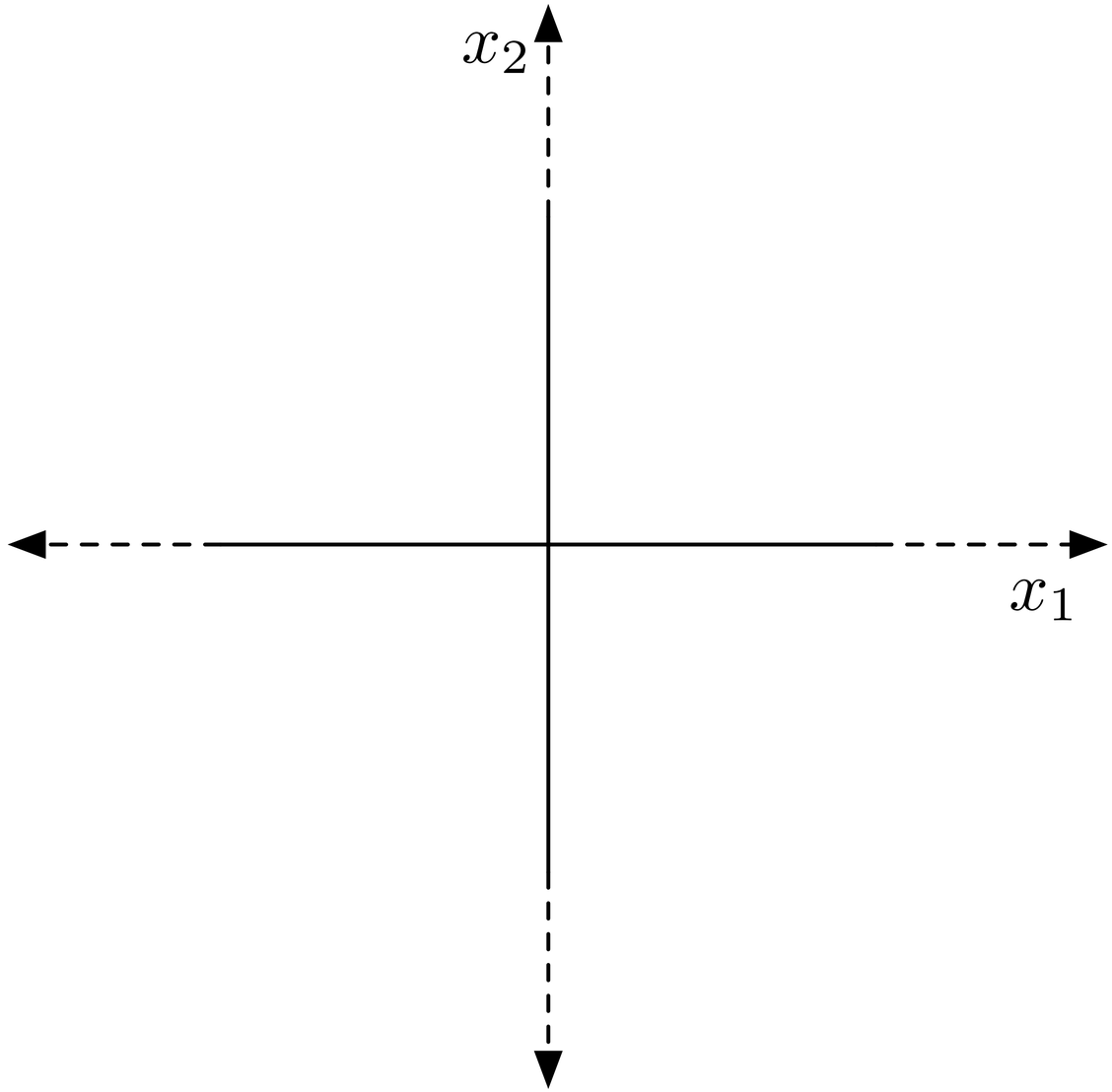
Feature space

Feature vectors can be considered as points lying in an N-dimensional space $\mathbf{x} \in \mathscr{R}^N$.

This is often referred to as **feature space**.

For ease of visualisation we can consider 2 features $\mathbf{x} = (x_1, x_2)$.

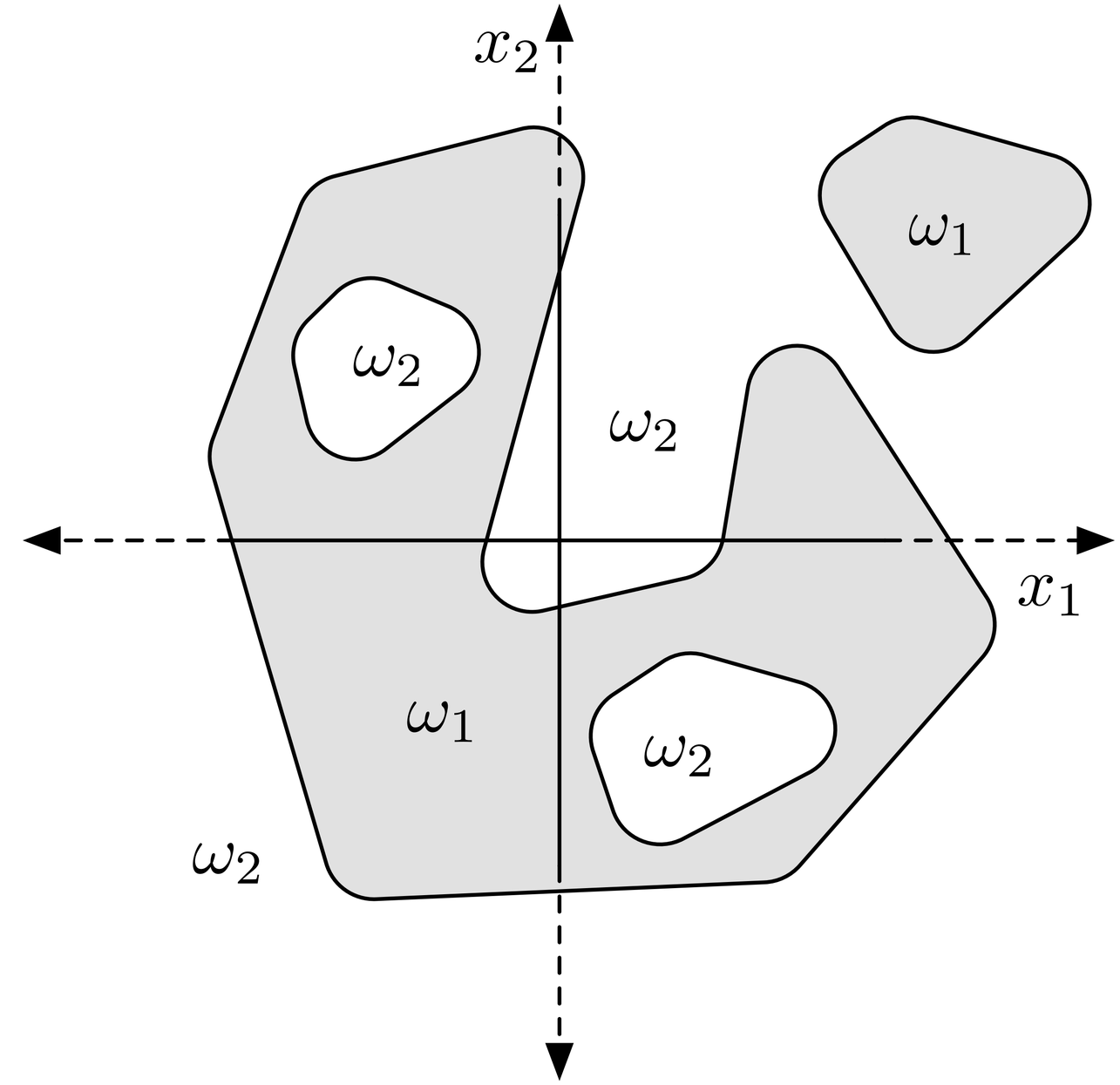
Feature space is a plane!



Feature space

Our classifier will produce a label for every point in space.

Here is an example for 2 classes.

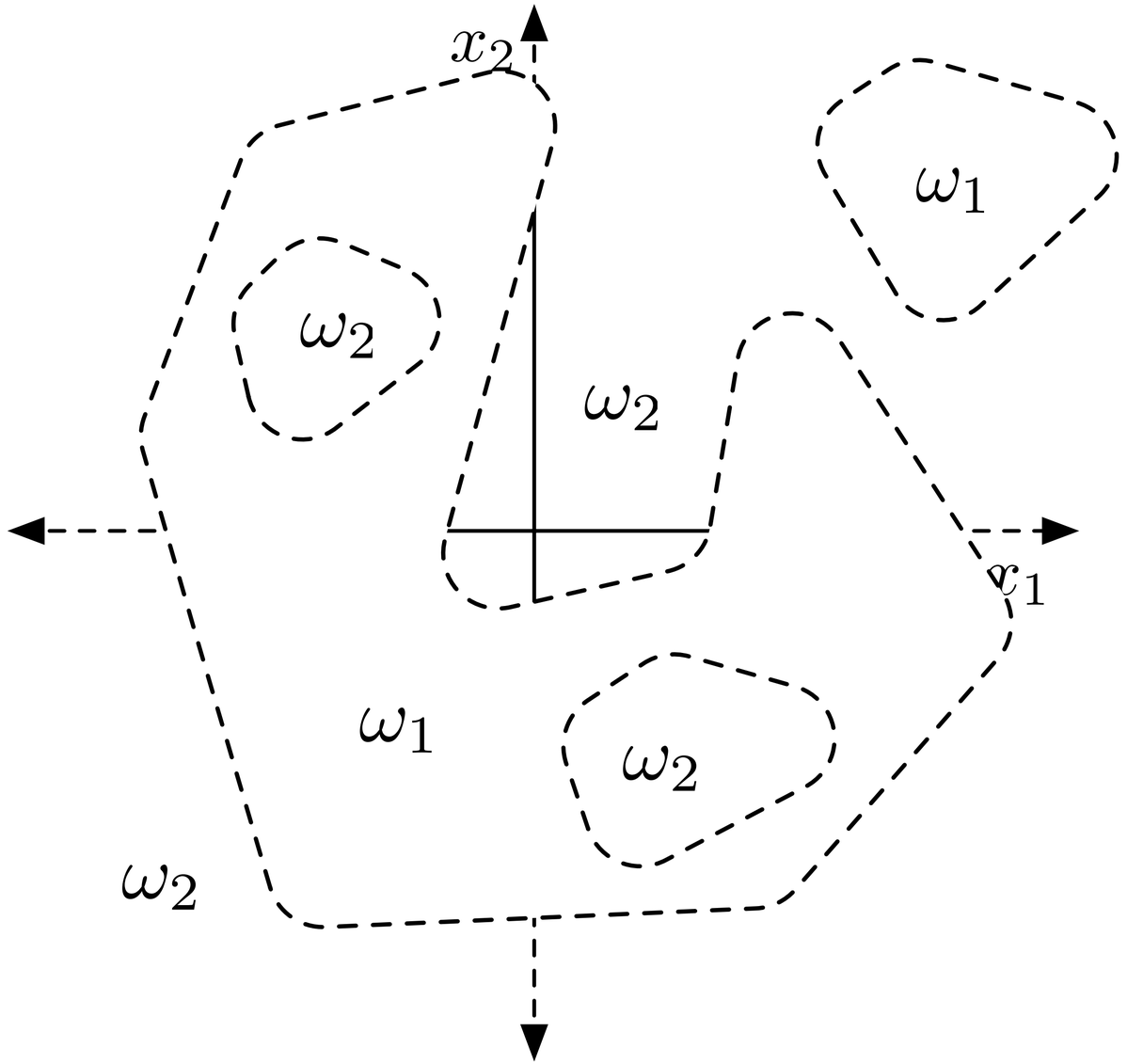


Decision Boundary

The decision boundary is the line (or lines) **separating classes**.

The boundary is **a property of the classifier**.

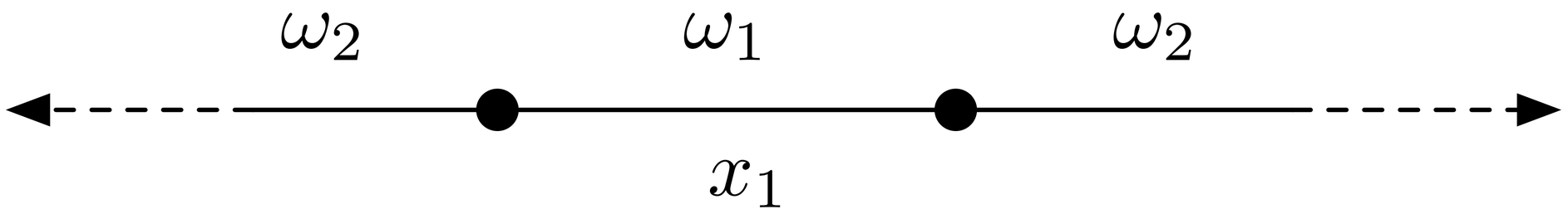
I.e different types of classifiers will have different boundaries even when trained on the same data.



Decision boundary

Generalising to N dimensional feature spaces

- In 2-D feature space the boundary is formed of 1-D lines
- In 3-D feature space the boundary is formed of 2-D surfaces
- In N-D feature space the boundary is formed of N – 1 dimensional ‘hypersurfaces‘.
- In 1-D feature space the boundary is formed by one or more points



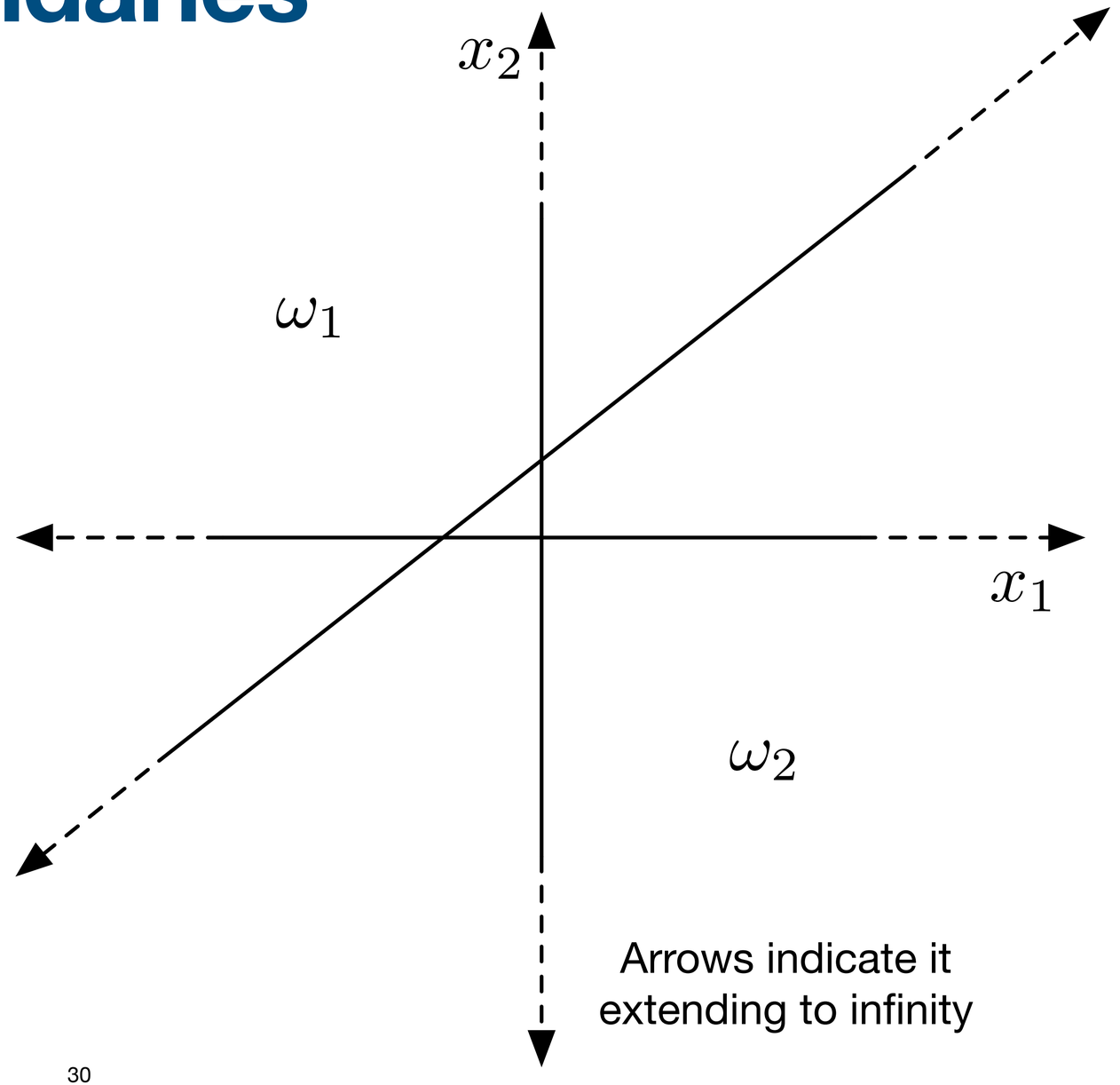
Linear decision boundaries

A **linear** decision boundary is simply one that is a **single straight line.**

In 3-D, it will be a plane.

In N-D, it will be a N-1 hyperplane.

In 1-D, it is a single point.



Linear classifiers

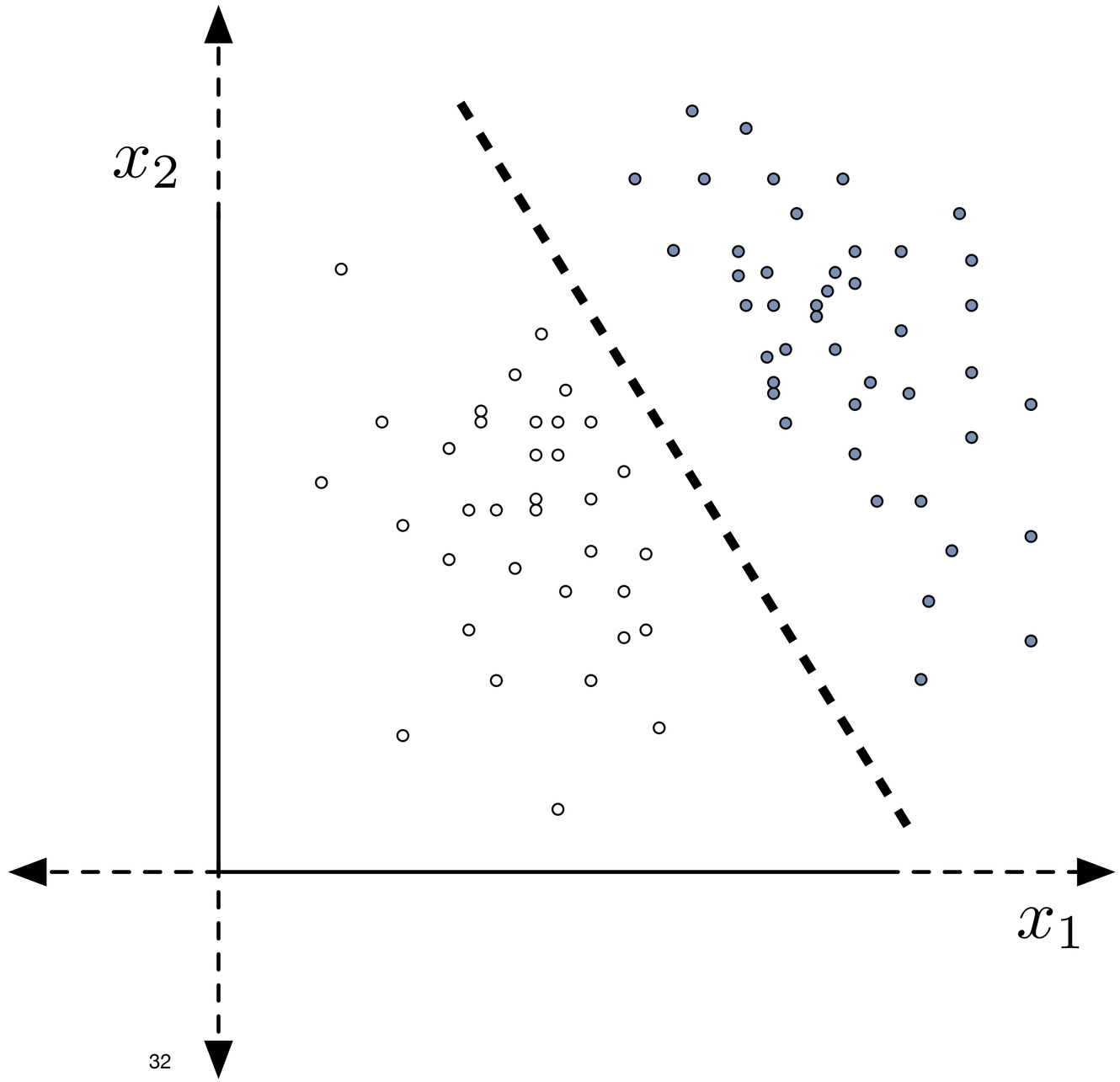
A **linear classifier** is simply a classifier that can only generate lineardecision boundaries:

- The simplest tool in our toolbox
- Easy to train
- Easy to analyse

Linear separability

Consider some data belonging to 2 classes.

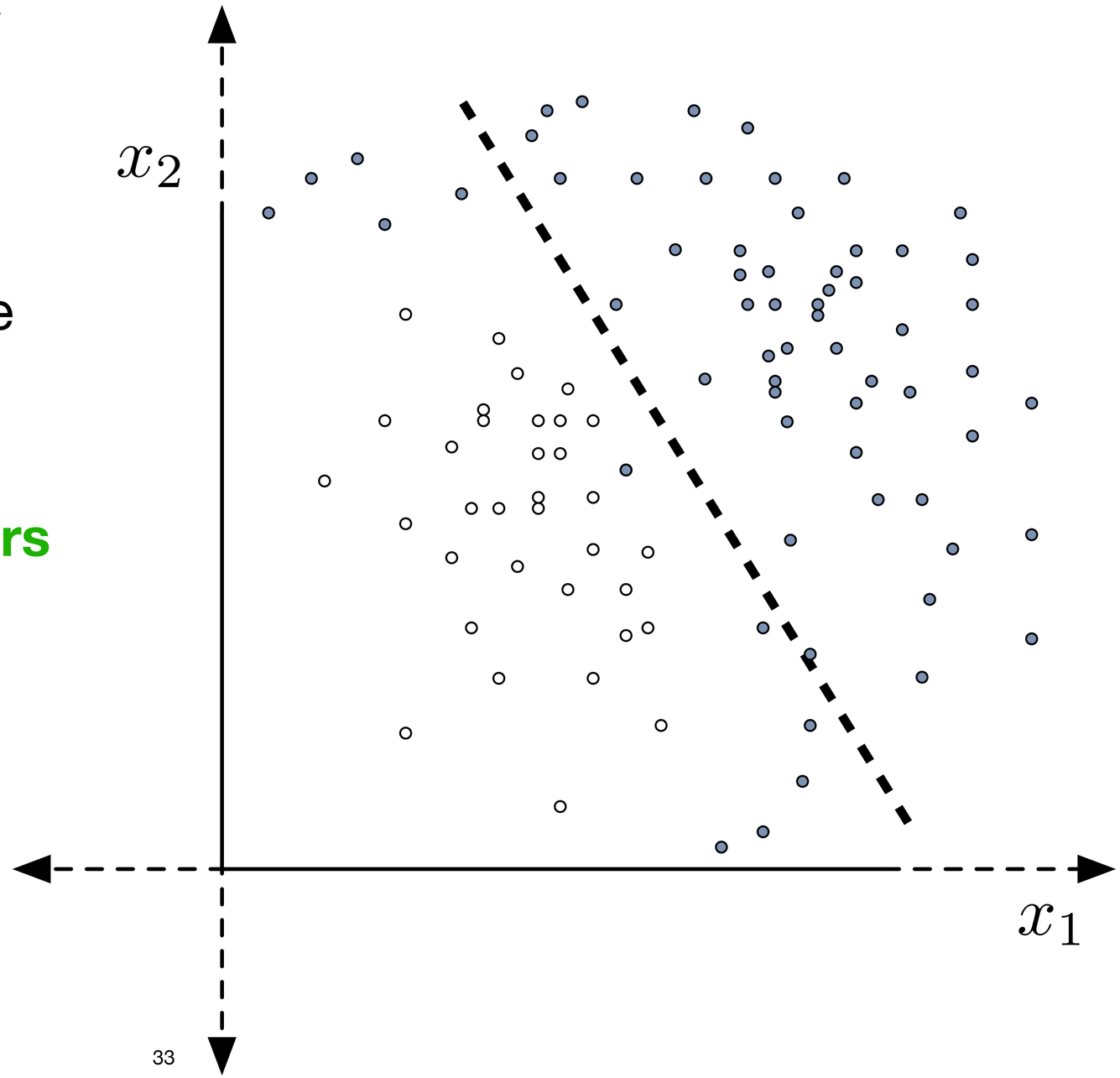
A linear decision boundary will **fully separate** the classes.



Linear separability

However, in general data won't be linearly separable, usually called **non-linear separability**.

We will need **non-linear classifiers** (e.g deep neural networks).



Summary

Feature space is the N-dimensional space in which we can plot our N-dimensional feature vectors, x.

A **classifier** will split this space into regions, with each region assigned to a single class.

The boundary between these regions is the classifier's **decision** boundary.

Linear classifiers are those which produce a hyper-plane decision boundary

A set of data is **linearly separable** if the classes can be separated by a linear decision boundary.

Analysing the Bayesian classifier

Decision boundaries

Consider the **Bayesian classifiers** with $p(\mathbf{x} \mid \omega)$modelled using Gaussian distributions that we discussed last week.

We'll be asking two questions

- Are they linear classifiers?
- If not, are there conditions under which they become linear?

Bayesian classifiers

Consider two classes, ω_1 and ω_2

The decision boundary is given by points x where:

$$P(\omega_1 \,|\, \mathbf{x}) = P(\omega_2 \,|\, \mathbf{x})$$

We can rewrite this as

$$P(\omega_1 \,|\, \mathbf{x}) - P(\omega_2 \,|\, \mathbf{x}) = 0$$

Note, on one side of the boundary that function is +ve and on the other side it is -ve.

Generalising to an M-class classification task:

$$P(\omega_i \,|\, \mathbf{x}) - P(\omega_j \,|\, \mathbf{x}) = 0$$

is the surface separating the regions R_i and R_j.

Bayesian classifiers

To make the maths easier, we define a **discriminant function**

$$g_i(\mathbf{x}) \equiv f\left(P(\omega_i \,|\, \mathbf{x})\right)$$

It must be monotonically increasing, i.e. $x_2 > x_1 \rightarrow f(x_2) > f(x_1)$

It then follows that from our decision rule

$$\text{If } P(\omega_i \,|\, \mathbf{x}) > P(\omega_j \,|\, \mathbf{x}) \text{ then } g_i(\mathbf{x}) > g_j(\mathbf{x})$$

Then our decision test can be: $\mathbf{x}$ belongs to ω_i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq i$

The decision boundary is defined by

$$g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0$$

Bayesian classifiers

For our Bayesian classifiers, define the natural log as our discriminant function:

$$g_i(\mathbf{x}) = \ln P(\omega_i \,|\, \mathbf{x})$$

Using this we can apply Bayes' rule and some rearrangement

$$g_i(\mathbf{x}) = \ln P(\omega_i \,|\, \mathbf{x}) \;\; = \ln \left(\frac{p(\mathbf{x} \,|\, \omega_i) P(\omega_i)}{p(\mathbf{x})} \right)$$

$$= \ln p(\mathbf{x} \,|\, \omega_i) + \ln P(\omega_i) - \ln p(\mathbf{x})$$

Gaussian Bayesian classifier

We will now focus on a classifier with a Gaussian pdf likelihood:

$$p(\mathbf{x}\,|\,\omega_i) = \frac{1}{\sqrt{(2\pi)^L\,|\,\boldsymbol{\Sigma}_i\,|}} \exp\left(-\frac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)\right)$$

With mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ that have been trained for each class i.

Adding this to discriminant function:

$$g_i(\mathbf{x}) = \ln p(\mathbf{x}\,|\,\omega_i) + \ln P(\omega_i) - \ln p(\mathbf{x})$$

$$= -\frac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_i\right) \underbrace{-\frac{1}{2}\ln(2\pi)^L\,|\,\boldsymbol{\Sigma}_i\,|}_{c_i} + \ln P(\omega_i) - \ln p(\mathbf{x})$$

Two classes with equal covariance

The decision boundary will be

$$g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0$$

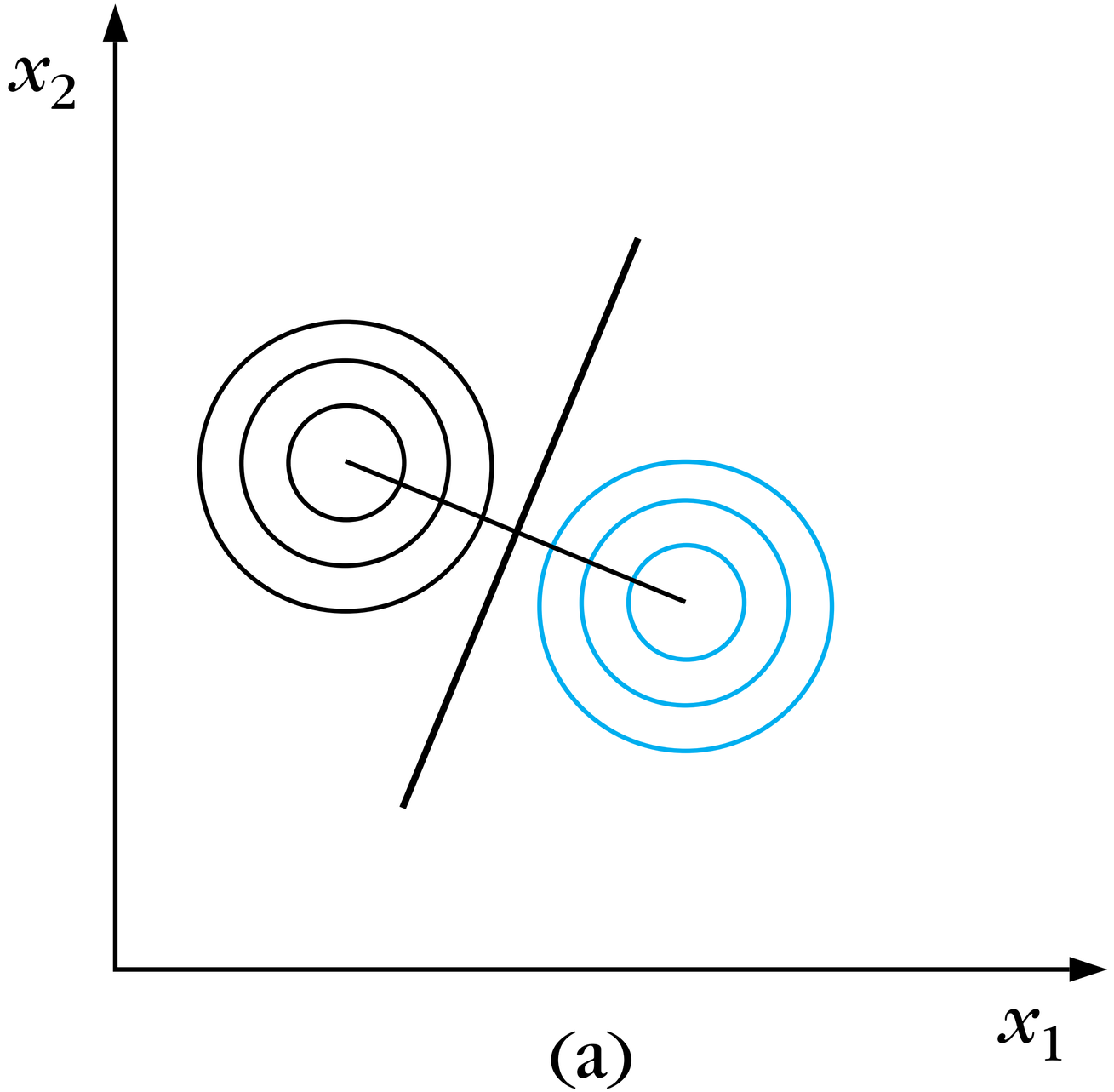
Since the covariances are equal, these terms will cancel out. Also the terms that only depend on $\mathbf{x}$ will cancel. Leaving something of the form

$$\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1} \mathbf{x} - \frac{1}{2} \boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_c + C = 0$$

This is actually the equation of a straight line in feature space

$$\mathbf{w}^T \mathbf{x} + w_0 = 0 = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_l x_L$$

When covariances are equal the decision boundary is linear.


x_2
x_1
(a)

Bayesian classifiers - minimum distance

For equiprobable classes with equal covariances, the discriminant function is

$$g_i(\mathbf{x}) = -\frac{1}{2}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)$$

- If $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$, maximum $g_i(\mathbf{x})$ implies the **minimum Euclidean distance**

$$d = \|\mathbf{x}-\boldsymbol{\mu}_i\|$$

- Non-diagonal $\boldsymbol{\Sigma}$, maximum $g_i(\mathbf{x})$ implies the **minimum Mahalanobis distance**:

$$d = \left[\left(\mathbf{x}-\boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}^{-1}\left(\mathbf{x}-\boldsymbol{\mu}_i\right)\right]^{\frac{1}{2}}$$

Summary

We can analyse a **decision boundary** by finding the $\mathbf{x}$ for which $P(\omega_i \,|\, \mathbf{x}) - P(\omega_j \,|\, \mathbf{x}) = 0$

We can often simplify this using a **discriminant function**,

$$\text{i.e. solve } g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0 \text{ where } g_i(\mathbf{x}) \equiv f\left(P(\omega_i \,|\, \mathbf{x})\right)$$

For **Gaussian classifiers** we can use $g_i(\mathbf{x}) = \ln P(\omega_i \,|\, \mathbf{x})$

It turns out that Bayesian classification with Gaussian distributions is, in general, non-linear.

The decision boundaries will be defined by **quadratic equations**.

- They can form ellipsoids, parabolas, hyperbolas

In the special case **where covariance matrices are equal, they become linear**, i.e. decision boundaries are hyperplanes.

Parameterising a linear classifier

A linear classifier

a linear classifier is a mapping which partitions feature space using a linear function (**a straight line**, or **a hyperplane**)

It is one of the simplest classifiers we can imagine.

In the 2-dimensional feature space the decision boundary, separating the two classes, is a **straight line**.

Equations for the decision boundary

e.g for a 2-D problem with $\mathbf{x} = (x_1, x_2)$

The decision boundary can be defined as

$$g(\mathbf{x}) = w_1 x_1 + w_2 x_2 + w_0 = 0$$

i.e a line given by $x_2 = \frac{w_1}{w_2} x_1 + \frac{w_0}{w_1}$

Equations for the decision boundary

Consider an L-dimensional feature space where $\mathbf{x} = (x_1, x_2, \cdots, x_L)$

The decision boundary hyperplane is defined as

$$g(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_L x_L + w_0 = 0$$

More compactly:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 = 0$$

Where $\mathbf{x}$ is the feature vector, $\mathbf{w}$ is the weight vector and w_0 is the threshold.

Applying the classifier

For an input $\mathbf{x}$, first compute

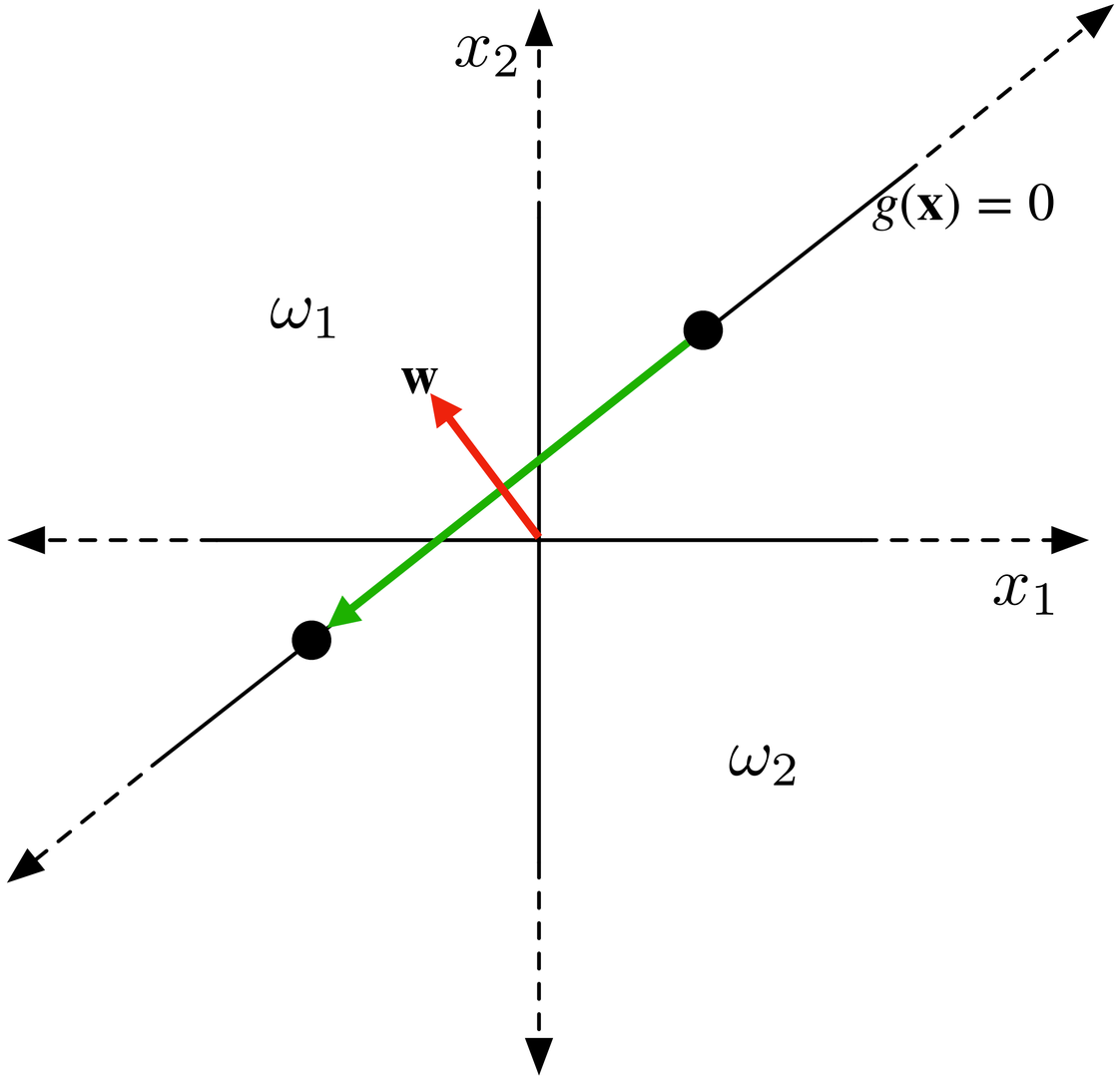
$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$$

Then

If $g(\mathbf{x}) > 0$ then $\mathbf{x}$ belongs to class 1

Else if $g(\mathbf{x}) < 0$ then $\mathbf{x}$ belongs to class 2.

Geometric interpretation of the weight vector



Suppose two points $\mathbf{x}_1$ and $\mathbf{x}_2$ both lie on the decision boundary so

$$\mathbf{w}^T\mathbf{x}_1 + w_0 = 0$$

$$\mathbf{w}^T\mathbf{x}_2 + w_0 = 0$$

So the difference will be

$$\mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

The vector $\mathbf{x}_1 - \mathbf{x}_2$ lies on the decision plane, so $\mathbf{w}$ must be perpendicular to it.

Counting parameters

How many parameters does a linear classifier have?

Compare this to the Gaussian Bayesian classifier:

In 2-D 2 class, each class has a mean vector, covariance matrix and prior:

$$\boldsymbol{\mu}_i = \begin{pmatrix} \cdot \\ \cdot \end{pmatrix}, \boldsymbol{\Sigma}_i = \begin{pmatrix} \cdot & \cdot \\ \cdot & \cdot \end{pmatrix}, P(\omega_i) \quad \text{so 14 across 2 classes}$$

but the priors sum to 1 and covariance is symmetric: (3 + 2)x2 + 1 = **11**

For L-D 2 class, $\left(\frac{1}{2}L(L+1)+L\right)\times 2+1 = O(L^2)$

Summary

Linear classifiers have **linear decision boundaries.**

The boundary can be expressed as an equation of the form

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0$$

The weight vector $\mathbf{w}$ will be **perpendicular to the decision boundary**, i.e., it specifies its orientation.

An L dimensional linear classifiers have **just L free parameters**. (c.f., Gaussians which has $O(L^2)$ parameters)

Required reading

Pattern Recognition by Sergios Theodoridis

- Chapter 3, Sections 3.1, 3.2, 3.3
- Chapter 4, Sections 4.1, 4.2, 4.3

Thanks for listening