

Multivariate distributions and distance measures

Week 3 - Lecture 5

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. Explain what is meant by a joint probability distribution.
2. Recall the definition of the multivariate Gaussian distribution.
3. Define the properties of a similarity and dissimilarity measure.
4. Recall the Euclidean, l^p -norm and cosine distances.
5. Define risk and how it can be minimised e.g by likelihood ratio test.

Lecture 4 Recap

Maximum likelihood estimate (MLE)

Estimate the parameters θ that best maximise the likelihood based on the data:

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\theta} p(X; \theta)$$

This is a maximisation problem, how do we solve this?

We need to find the values of θ where

$$\frac{\partial p(X; \theta)}{\partial \theta} = 0$$

i.e the gradient of the likelihood is zero when $\theta = \hat{\theta}_{\text{MLE}}$

A cunning trick

We need to find the parameters θ that give the **largest value** of $p(\mathbf{X}; \theta)$ for some observed data \mathbf{X} .

We can substitute this for an easier problem that shares the same solution:

Find the parameters θ that maximise $\ln p(\mathbf{X}; \theta)$

I.e instead of maximising the **likelihood**, we maximise the **log likelihood**

Why is this equivalent?

Why is it easier?

The log likelihood

Since our samples are **independent** then the likelihood can be written as

$$p(\mathbf{X}; \theta) = \prod_{i=1}^N p(\mathbf{x}_i; \theta)$$

This means the log likelihood is then

$$L(\theta) = \ln p(\mathbf{X}; \theta) = \ln \left(\prod_{i=1}^N p(\mathbf{x}_i; \theta) \right) = \sum_{i=1}^N \ln p(\mathbf{x}_i; \theta)$$

Note: since the log is a monotonic function they maximise for the same parameters.

The log likelihood

$$L(\theta) = \ln p(\mathbf{X}; \theta) = \sum_{i=1}^N \ln p(\mathbf{x}_i; \theta)$$

Hence the gradient (and maximisation criteria) is:

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{i=1}^N \frac{\partial}{\partial \theta} \ln p(\mathbf{x}_i; \theta) = 0$$

MLE example - Gaussian pdf

Let's consider a simple 1d example with N data points x_1, x_2, \dots, x_N , distributed from a Gaussian pdf with unknown mean μ and unknown variance σ^2 :

$$p(x_i; \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i - \mu)^2}{2\sigma^2}\right)$$

Now the log likelihood is

$$\begin{aligned} L(\mu, \sigma^2) &= \sum_{i=1}^N \ln p(x_i; \mu, \sigma^2) \\ &= -\frac{N}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^N (x_i - \mu)^2 \end{aligned}$$

Exercise - find the MLE parameters

$$L(\mu, \sigma^2) = -\frac{N}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^N (x_i - \mu)^2$$

Exercise - find the derivatives then solve for where they equal zero.

$$\frac{\partial L(\mu, \sigma^2)}{\partial \mu} = 0$$

$$\frac{\partial L(\mu, \sigma^2)}{\partial \sigma^2} = 0$$

Solutions

After doing the derivation we find that the MLE values for the Gaussian distribution are the sample mean and variances. These can be used for learning the pdf's that are then used as part of the classifier.

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i$$

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

Summary

Parameter estimation is the task of **fitting a distribution to some observed data**.

Maximum likelihood estimate (MLE): find the parameters of the distribution that maximise the likelihood of the data.

This can be performed by differentiating likelihood with respect to the parameters and finding where the gradient is 0.

It is typically easier to solve the equivalent problem: find the parameters that **maximise the log likelihood**.

In some simple cases (exam questions!) we can find the maximum by inspection: i.e., plotting the likelihood as a function of the parameters and finding the peak.

Multivariate Gaussian

Multivariate probability distributions

We have seen how a normal distribution can model the distribution of a single feature, e.g. height, i.e. $p(x)$

However, in our classification problems we need to model $p(\mathbf{x})$ where \mathbf{x} is a vector containing many features.

We need to generalise the normal distribution from one dimension to multiple dimensions, i.e., to model the joint distribution of multiple features.

Joint distribution $P(X, Y)$ describes the joint statistics of two random variables X and Y .

Joint probability distributions

For discrete RVs we can write $P(X, Y) \equiv P(X \cap Y)$

Think of this a table, e.g for dice rolls:

$$P(X_1, X_2) =$$

	1	2	3	4	5	6
1	1/36	1/36	1/36	1/36	1/36	1/36
2	1/36	1/36	1/36	1/36	1/36	1/36
3	1/36	1/36	1/36	1/36	1/36	1/36
4	1/36	1/36	1/36	1/36	1/36	1/36
5	1/36	1/36	1/36	1/36	1/36	1/36
6	1/36	1/36	1/36	1/36	1/36	1/36

Both RVs are uniform
and independent!

Continuous joint distributions

Same as before, we can define a joint cumulative distribution function

$$F(x, y) = P(X \leq x, Y \leq y)$$

Which is used to define out joint probability density function:

$$F(x, y) = \int_{-\infty}^x \int_{-\infty}^y p(u, v) du dv$$

Again which is equivalent to

$$\frac{\partial^2 F(x, y)}{\partial x \partial y} = p(x, y)$$

Continuous joint distributions

More generally for L random variables $\mathbf{x} = (x_1, x_2, \dots, x_L)^T$

We have the cumulative density function

$$F(\mathbf{x}) = P(X_1 \leq x_1, X_2 \leq x_2, \dots, X_L \leq x_L)$$

Which defines the probability density function

$$F(\mathbf{x}) = \int_{-\infty}^{x_1} \dots \int_{-\infty}^{x_L} p(u_1, \dots, u_L) du_1 \dots du_L$$

Multivariate Gaussian distribution

What this all means is that we can now start to define a function to represent the probability density of L dimensional feature vector.

The L dimensional Gaussian pdf is

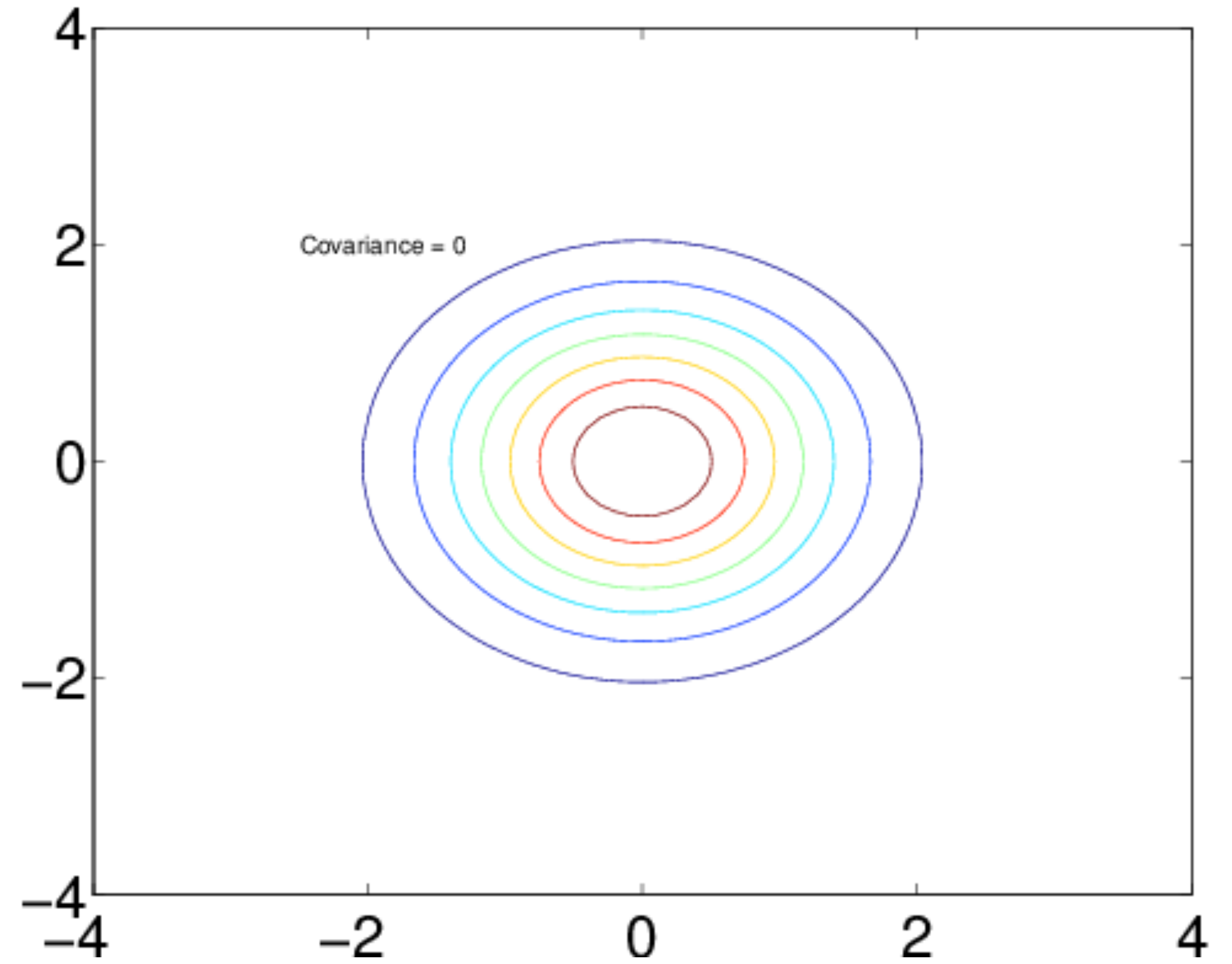
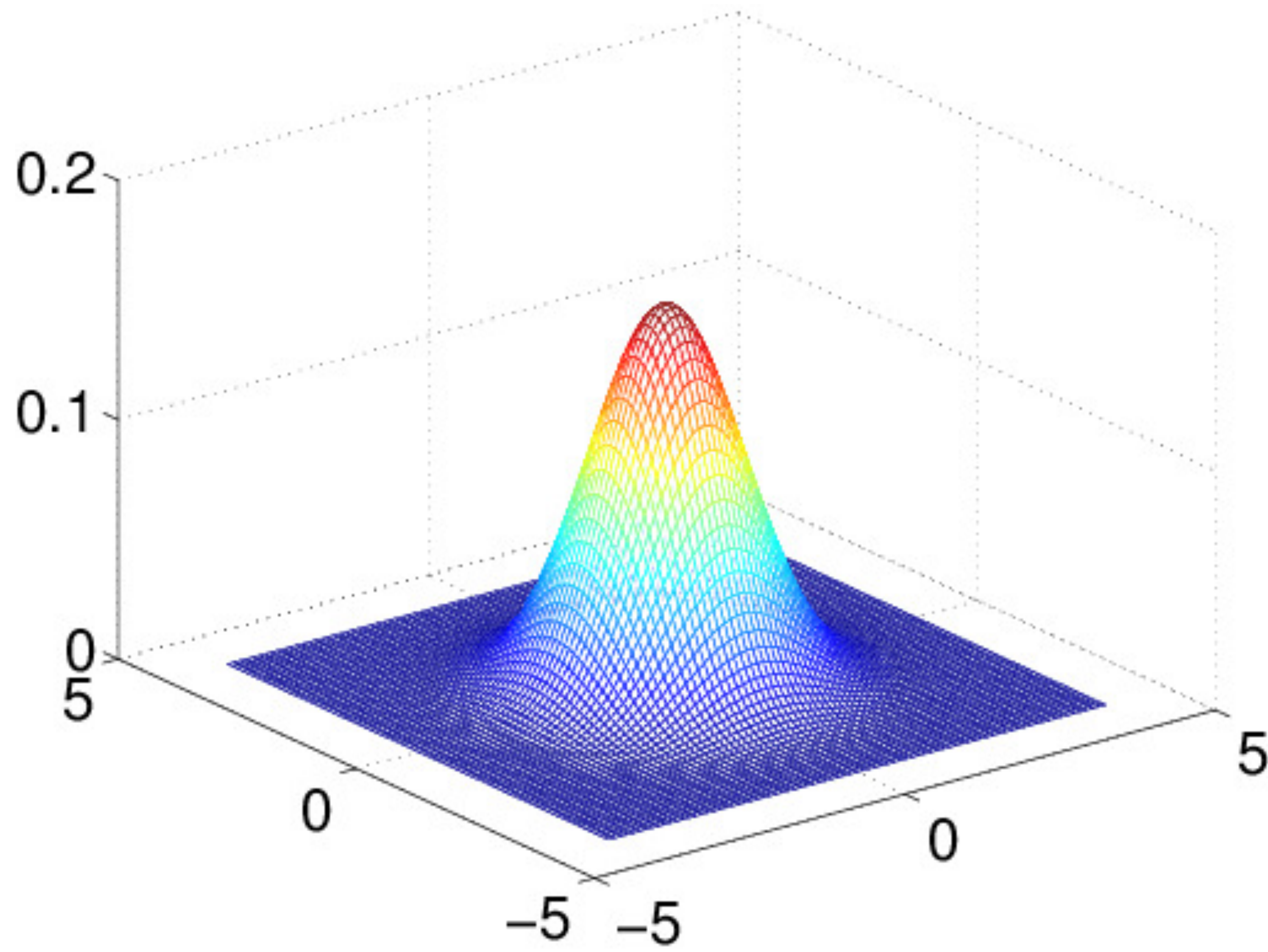
$$p(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^L |\boldsymbol{\Sigma}|}} \exp \left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right)$$

$\boldsymbol{\mu}$ is a L dimensional mean vector

$\boldsymbol{\Sigma}$ is a $L \times L$ dimensional covariance matrix

$|\boldsymbol{\Sigma}|$ is the matrix determinant and $\boldsymbol{\Sigma}^{-1}$ is the matrix inverse.

2d example



Maximum likelihood estimate for multivariate Gaussian

We can now also consider the MLE when we have L dimensional feature vectors.

$$\begin{aligned} L(\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{i=1}^N \ln p(\mathbf{x}_i; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= -\frac{N}{2} \ln \left((2\pi)^L |\boldsymbol{\Sigma}| \right) - \frac{1}{2} \sum_{i=1}^N (\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \end{aligned}$$

Which we can then maximise to estimate the parameters. When we have found them based on the training data we can classify using them.

Summary

Probability density functions can be generalised to represent the **joint distribution** of multiple variables.

The univariate Gaussian distribution becomes the **multivariate Gaussian distribution**.

The univariate distribution had two parameters: a mean and a variance.

The multivariate normal distribution is parameterised with a **mean vector** and a **covariance matrix**.

Now with L features, we need to estimate the L values of the mean vector and L^2 of the covariance for a Gaussian classifier.

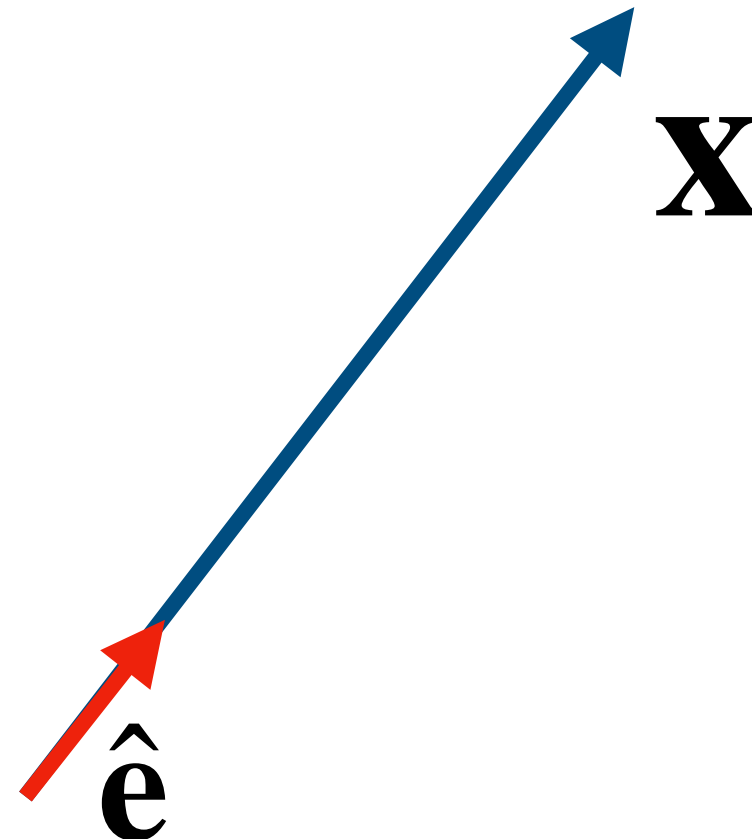
Vectors and matrices

Vectors and matrices

Both of these are quite common in machine learning so it is useful to remind ourselves of how they work.

I have already introduced that we will contain all our features into a feature vector. We can then view this vector as existing in a **L dimensional 'feature space'**. Each sample is then a point within this feature space.

Each vector is a direction and a magnitude:

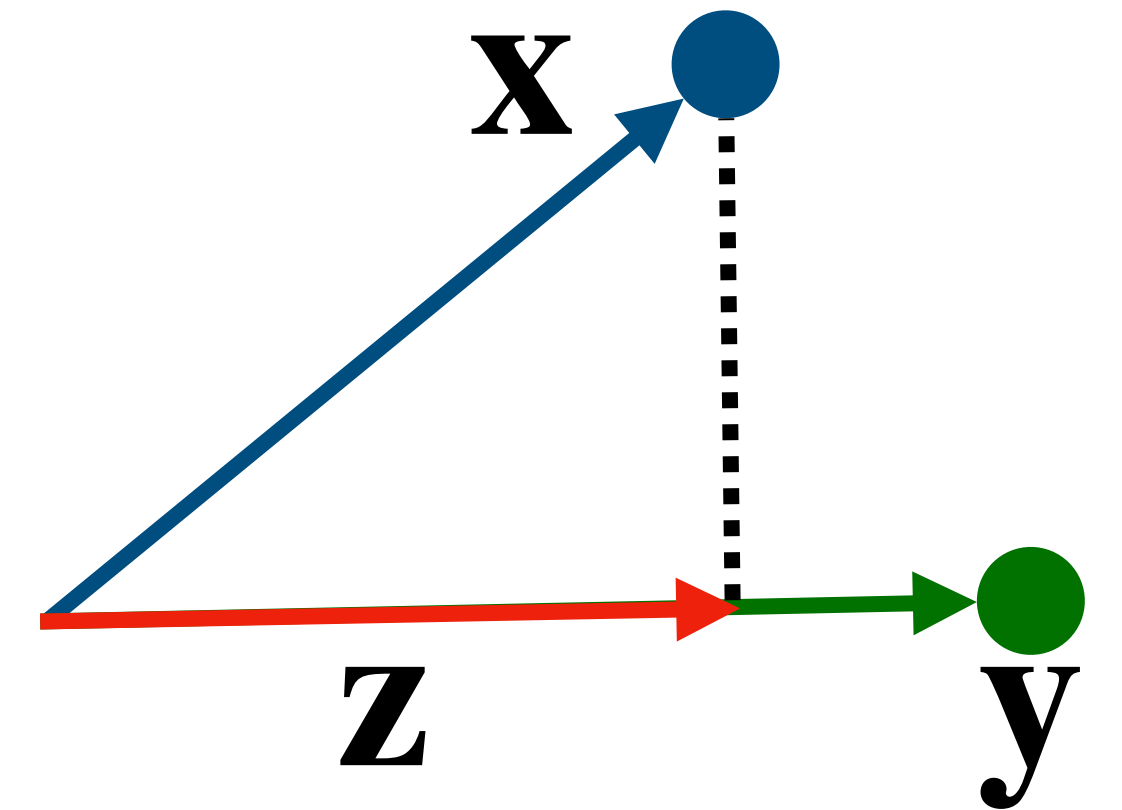
$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{pmatrix} = |\mathbf{x}| \hat{\mathbf{e}} \quad |\mathbf{x}| = \sqrt{\sum_{i=1}^L x_i^2}$$


The diagram illustrates the decomposition of a vector \mathbf{x} into its magnitude and direction. A blue arrow labeled \mathbf{x} points upwards and to the right. A red arrow labeled $\hat{\mathbf{e}}$ points in the same direction as \mathbf{x} but is shorter, representing the unit vector in the direction of \mathbf{x} .

Inner product and projections

The inner (or dot) product is an important operation:

$$\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^L x_i y_i = \mathbf{x}^T \mathbf{y} = (x_1 \ x_2 \ \cdots \ x_L) \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{pmatrix}$$



It measures the magnitude of the projection of one vector onto another. We can define the projection vector of **x** on to **y** as:

$$\mathbf{z} = (\mathbf{x} \cdot \mathbf{y}) \frac{\mathbf{y}}{(\mathbf{y} \cdot \mathbf{y})}$$

The outer product

The outer product is between two vectors and results in a matrix:

$$\mathbf{x} \otimes \mathbf{y} = \mathbf{xy}^T = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{pmatrix} (y_1 \quad y_2 \quad \vdots \quad y_l) = \begin{pmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_L \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_L \\ \vdots & \vdots & \ddots & \vdots \\ x_L y_1 & x_L y_2 & \cdots & x_L y_L \end{pmatrix}$$

Matrix products

Later in the course we will come across cases where we need to multiply a matrix by either a vector or a matrix. If \mathbf{A} is a M by L matrix then the matrix vector product is:

$$\mathbf{A}\mathbf{x} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1L} \\ a_{21} & a_{22} & \cdots & a_{2L} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{ML} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^L a_{1i}x_i \\ \sum_{i=1}^L a_{2i}x_i \\ \vdots \\ \sum_{i=1}^L a_{Mi}x_i \end{pmatrix}$$

A good thing to remember is that the product of a M by L matrix with a L by K will result in a M by K matrix. In maths and programming it is important to make sure that the shapes of your matrices or arrays match.

Distance Measures

Why are distance measures important?

Many techniques in 'Data Driven Computing' rely on having a way of measuring the distance between two objects.

Where 'distance' is a measure of how different they are.

Examples:

Classification: comparing an item to a labeled prototype.

Regression: measuring the error between observations and predictions.

Clustering: determine whether items are similar enough to be in the same cluster.

Measuring the distance between vectors

Remember, our objects are going to be represented by vectors.

So a distance measure is a function that takes a pair of vectors and returns a scalar.

There are many different ways that the distance between two vectors can be defined:

e.g Mahalanobis distance, Cosine difference

But there are some properties that they share.

Properties of distance measures

Two types of proximity measures:

Dissimilarity measures - larger value = further apart

Similarity measures - larger value = closer

Typically, by distance measure people are referring to a dissimilarity measure.

Dissimilarity (i.e distance) measures

Function that measure dissimilarity between a pair of points (larger score - greater dissimilarity).

A function is a **valid dissimilarity measure** if:

1. it returns the same value d_0 when measuring dissimilarity between a point and itself,

$$d(\mathbf{x}, \mathbf{x}) = d_0, \forall \mathbf{x} \in X$$

2. And dissimilarity between two points is never **less than** d_0 ,

$$d(\mathbf{x}, \mathbf{y}) \geq d_0, \forall \mathbf{x}, \mathbf{y} \in X$$

Metric dissimilarity measures

It can also be called a **metric** dissimilarity measure if it obeys the two following conditions:

1. Dissimilarity between **non-identical** points is always $> d_0$

$$d(\mathbf{x}, \mathbf{y}) = d_0, \text{ if and only if } \mathbf{x} = \mathbf{y}$$

2. It obeys the **triangle inequality**,

$$d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$$

Similarity measures

Similarity measures are essentially the opposite of dissimilarity ones, larger value = closer, greater similarity.

A function is a **valid similarity measure** if:

1. it returns the same value s_0 when measuring similarity between a point and itself,

$$s(\mathbf{x}, \mathbf{x}) = s_0, \forall \mathbf{x} \in X$$

2. And similarity between two points is never **greater than** s_0 ,

$$s(\mathbf{x}, \mathbf{y}) \geq s_0, \forall \mathbf{x}, \mathbf{y} \in X$$

should be " \leq "

Metric similarity measures

Again like dissimilarity measures, they can be said to be **metric similarity measures** if they obey the following:

1. Similarity between **non-identical** points is always $< s_0$

$$s(\mathbf{x}, \mathbf{y}) = s_0, \text{ if and only if } \mathbf{x} = \mathbf{y}$$

2. And

$$s(\mathbf{x}, \mathbf{y})s(\mathbf{y}, \mathbf{z}) \leq [s(\mathbf{x}, \mathbf{y}) + s(\mathbf{y}, \mathbf{z})]s(\mathbf{x}, \mathbf{z})$$

Some further notes

Similarity and dissimilarity measures do not have to be symmetric.

I.e it is ok for $s(\mathbf{x}, \mathbf{y}) \neq s(\mathbf{y}, \mathbf{x})$

Distance measures can have negative values, there just has to be a well-defined minimum value.

Euclidean distance

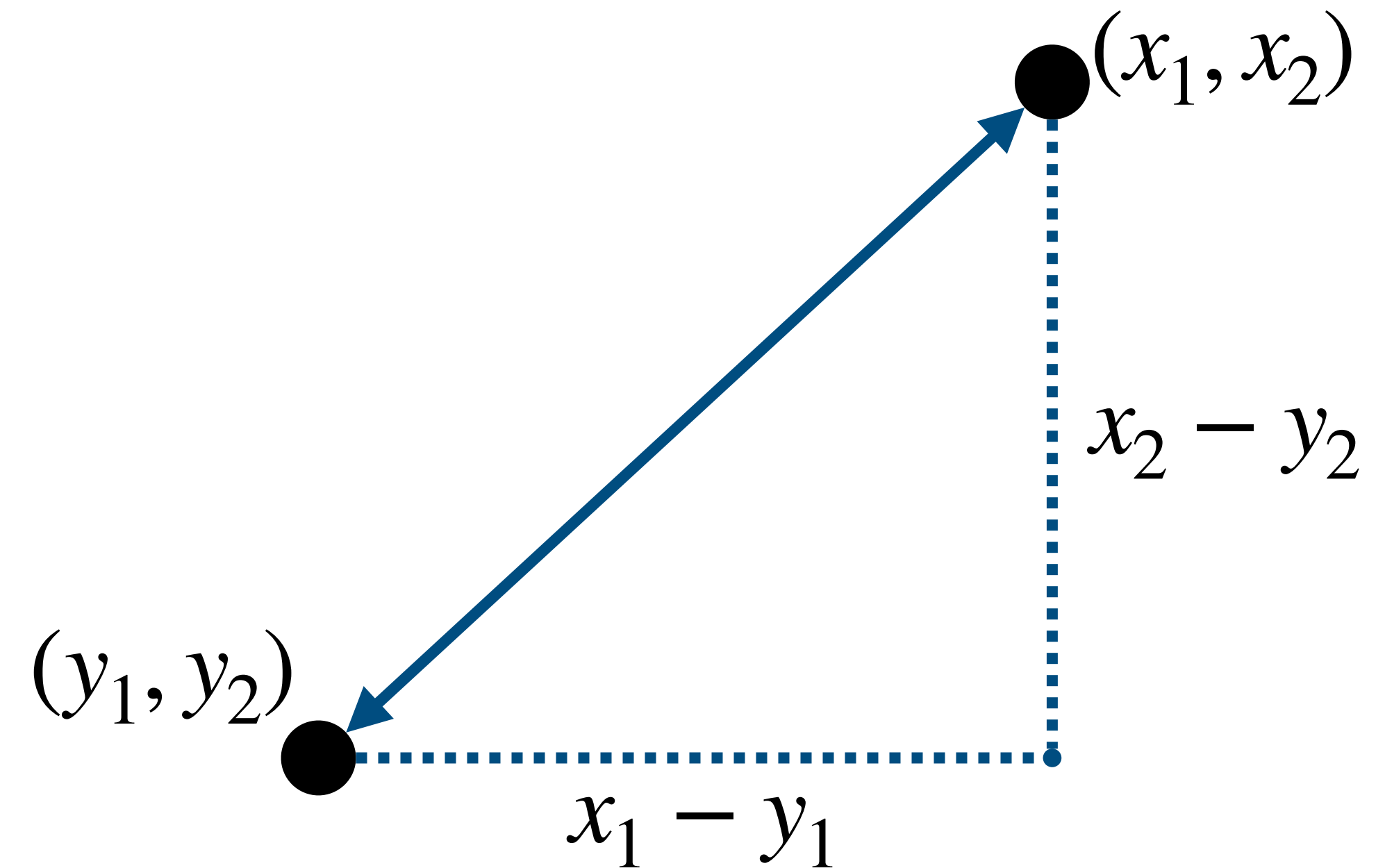
Consider two points on a 2-D plane.

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 $\mathbf{x} = (x_1, x_2, \dots, x_N)^T$ and $\mathbf{y} = (y_1, y_2, \dots, y_N)^T$

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

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

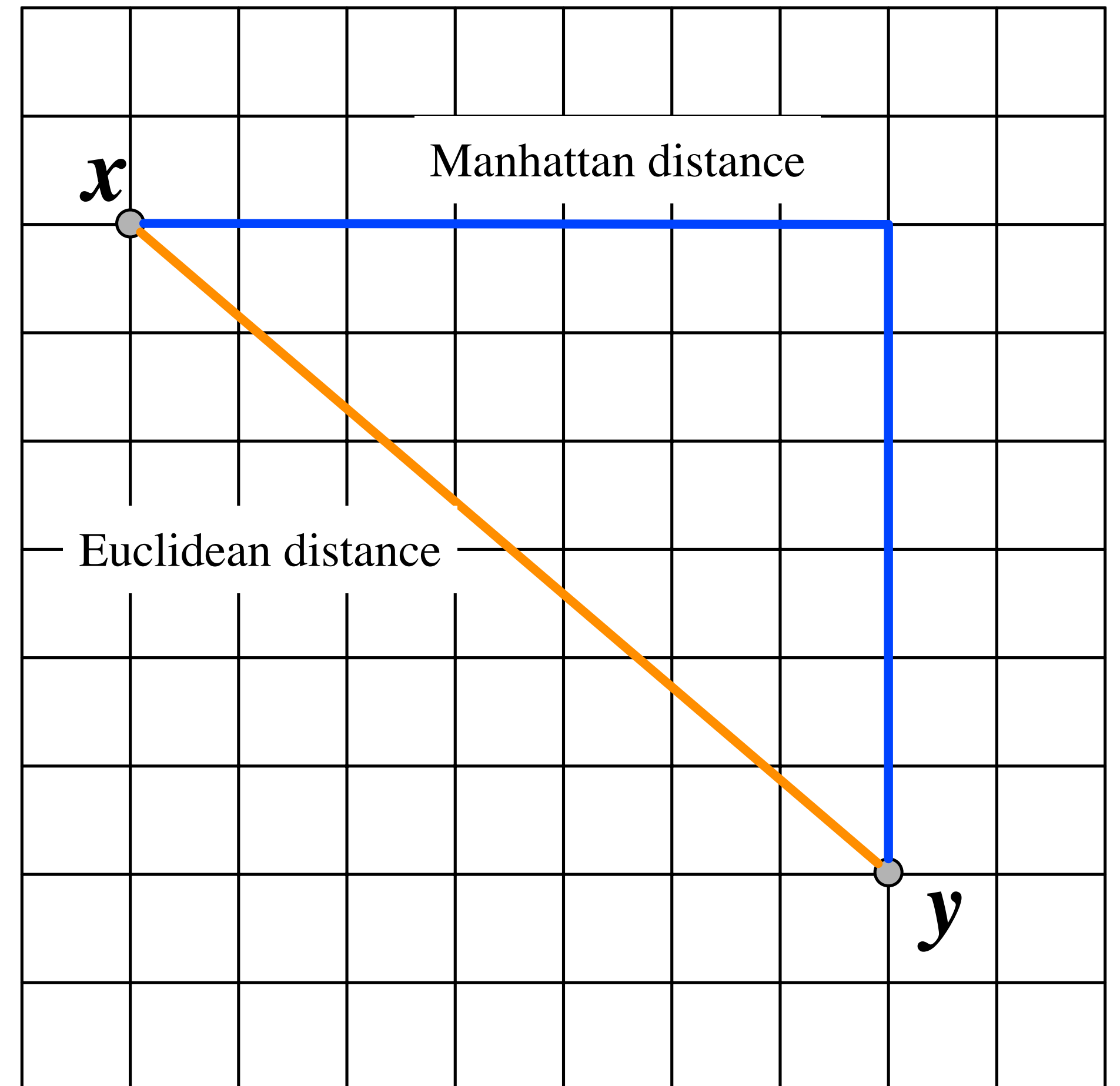
The l^1 and l^2 norms are common but quite different.

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.

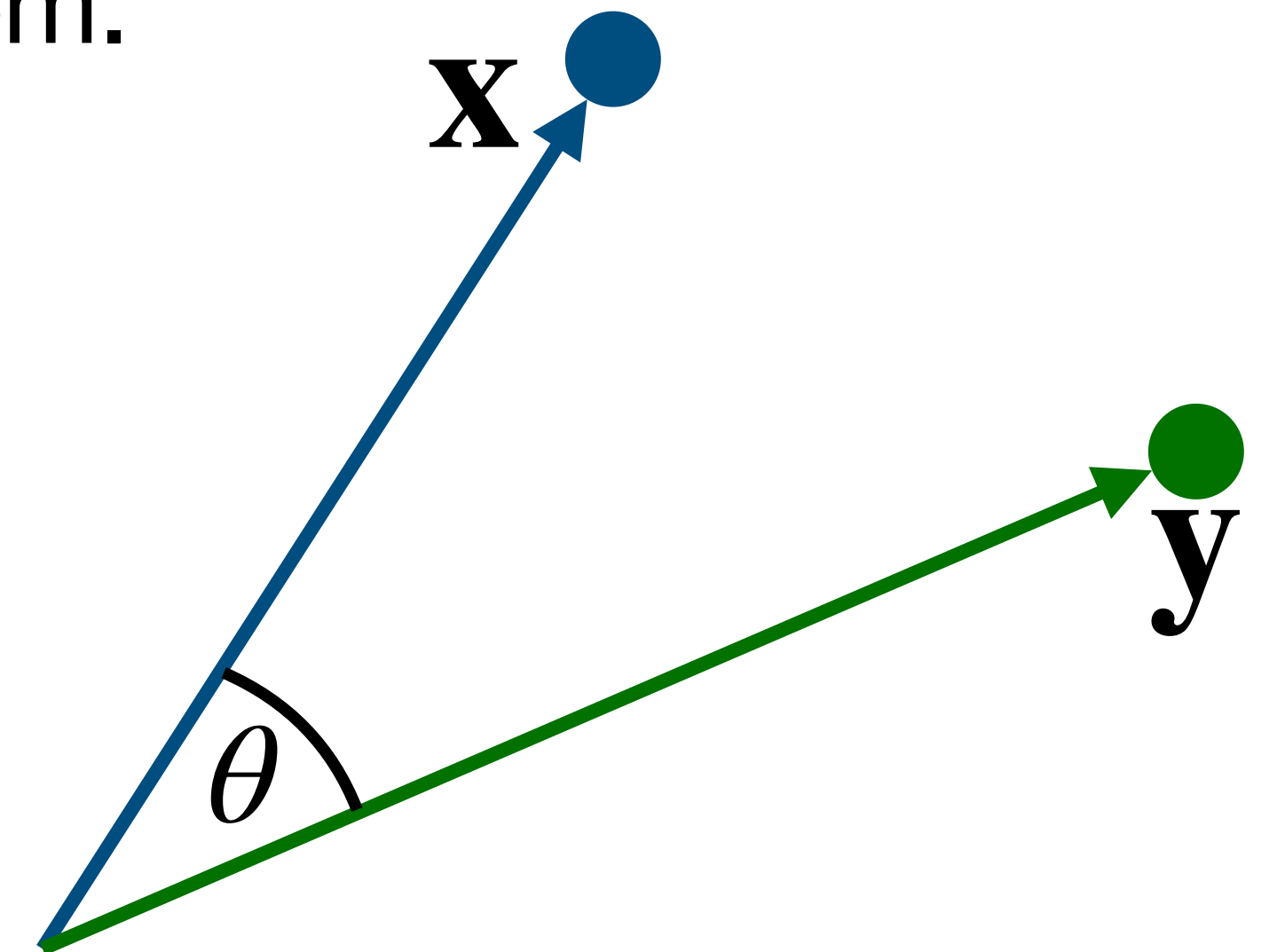
This can be calculated using

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

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):

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

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

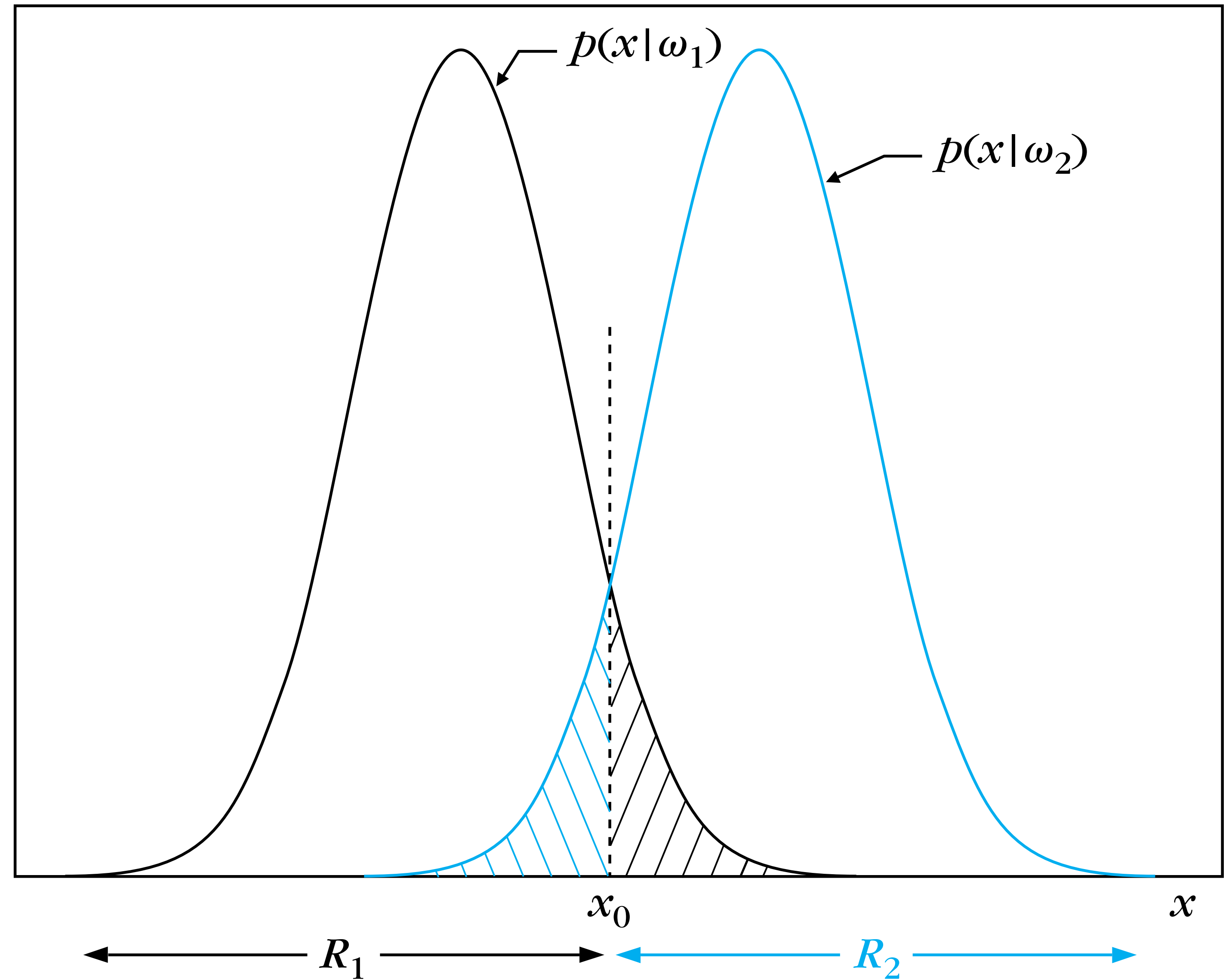
$$p(\mathbf{x} | \omega_1) > p(\mathbf{x} | \omega_2)$$

Gaussian pdfs $p(x|\omega)$

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:

$$\begin{aligned} P_e &= \int_{-\infty}^{x_0} p(\mathbf{x}, \omega_2) d\mathbf{x} + \int_{x_0}^{\infty} p(\mathbf{x}, \omega_1) d\mathbf{x} \\ &= \int_{-\infty}^{x_0} p(\mathbf{x} | \omega_2) P(\omega_2) d\mathbf{x} + \int_{x_0}^{\infty} p(\mathbf{x} | \omega_1) P(\omega_1) d\mathbf{x} \end{aligned}$$

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

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

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.

$$\rightarrow \lambda_{11} = 0, \quad \lambda_{12} = \lambda, \quad \lambda_{21} = 4\lambda, \quad \lambda_{22} = 2\lambda$$

Risk

General formula for M classes:

Risk associated with class ω_k :

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

Risk averaged over all classes:

$$\begin{aligned} r &= \sum_{k=1}^M r_k P(\omega_k) \\ &= \sum_{i=1}^M \int_{R_i} \underbrace{\left(\sum_{k=1}^M \lambda_{ki} p(\mathbf{x} | \omega_k) P(\omega_k) \right)}_{l_i} d\mathbf{x} \end{aligned}$$

Risk

The average risk is minimised by selecting partitioning regions so that

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

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**:

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

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

Thanks for listening