Fundamentals / Foundation of Data Science

Data Science 9CFU

Computer Science 6CFU

Indro Spinelli

Sapienza University of Rome

IRIS Dataset

Introduced by statistician Fisher, frequently used toy example.

- Classify iris subspecies based on flower measurements
- 150 iris flowers: 50 versicolor, 50 virginica, 50 setosa
- Sepal length / width and petal length / width in [cm]

Source: https://rpubs.com/vidhividhi/irisdataeda

Data in Supervised Learning

Measurements on different aspects of n objects:

Target: output variable / goal of prediction

Features: properties that describe the object

We assume some relationship between features and target, and want to discover that predictive rule.

Features x				Target y
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
4.3	3.0	1.1	0.1	setosa
5.0	3.3	1.4	0.2	setosa
7.7	3.8	6.7	2.2	virginica
5.5	2.5	4.0	1.3	versicolor

Data Types

Features and target may be of different data types

- Numerical variables: → R
- Integer variables: → Z
- Categorical variables: → $\{C_1, \ldots, C_g\}$
- Binary variables: →{0, 1}

For the target variable, this results in different tasks of supervisedlearning: regression and classification.

Conversion

Most learning algorithms can only deal with numerical features, although there are some exceptions.

For other types, we usually have to pick or create an appropriate encoding, i.e., cast them to numerical values.

Encoding for Categoricals

A categorical variable with k mutually exclusive categories is transformed from a scalar value into a k-dimensional binary vector.

This vector has exactly one element set to 1 to indicate the category, and 0s elsewhere:

$$\mathbf{e}_i = (0, \ldots, 0, \underbrace{1}_{i\text{-th position}}, 0, \ldots, 0)^\top, \quad i \in \{1, \ldots, k\}$$

For features with a natural order in their categories we resort to encodings that reflect this ordinality, e.g., a sequence of integer values

Labels

We call the entries of the target column labels.

We distinguish two types of data:

- Labeled data: target is observed
- Unlabeled data: target is unknown

Dataset (mathy)

Let:

$$D = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\} \subset (X \times Y)^n$$

- X is the **input space**, typically $X \subseteq \mathbb{R}^p$
- Y is the **output/target space**

Each observation (row):

$$(x^{(i)}, y^{(i)}) \in X \times Y, \quad i = 1, \ldots, n$$

The j-th feature (column):

$$x_j = (x_j^{(1)}, \ldots, x_j^{(n)})^\top \in \mathbb{R}^n$$

Data Generation Process

We assume the observed data D is generated by an underlying process characterized by a probability distribution:

$$P_{XY} \text{ defined on X×Y}$$

Random variables following this distribution are denoted by lowercase x and y.

The **true distribution** P_{XY} is usually unknown.

Supervised Machine Learning aims to learn (part of) this distribution's structure to predict y from x.

Assumptions

We assume data to be drawn i.i.d. (independent and identically distributed) from P_{xy}.

This means: We assume that all samples are drawn from the same distribution and are mutually independent the i-th realization does not depend on the other n – 1.

This is a strong assumption, not always realistic. More complex scenarios (e.g., time series) exist, but we focus on the i.i.d. scenario here.

Data Points as Vectors

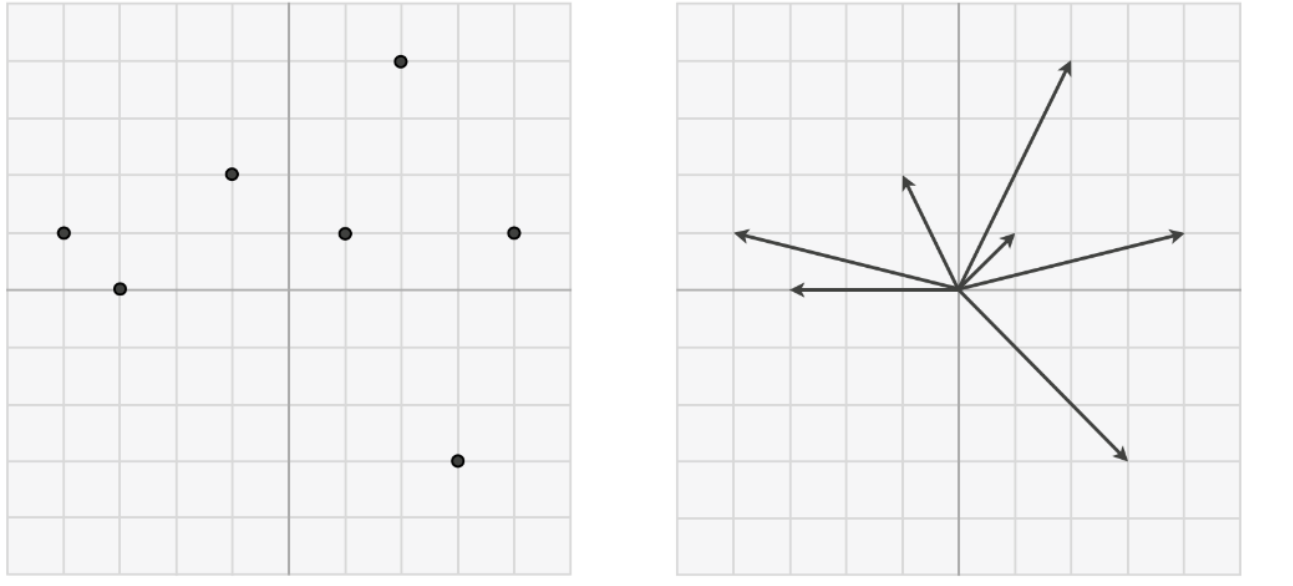
Lets focus only on X, we will assume that our dataset has been *mean-centered* - a simple and completely reversible operation that involves subtracting off the mean of the dataset along each input dimension - so that it straddles the origin.

Each observation in our dataset is a point in the input space.

In mathematics and physics, a **vector space** is a collection of objects, called **vectors**, that can be added together and multipied by numbers, known as **scalars**.

Visualizing vectors

When visualizing data points in a multi-dimensional vector space, we can represent them as either *dots* (left panel) or *arrows* (left panel).



Spanning set

A spanning set is a collection of vectors that can be combined (using addition and scalar multiplication) to produce **every vector** in a space.

Example: In 2D space, the two vectors

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

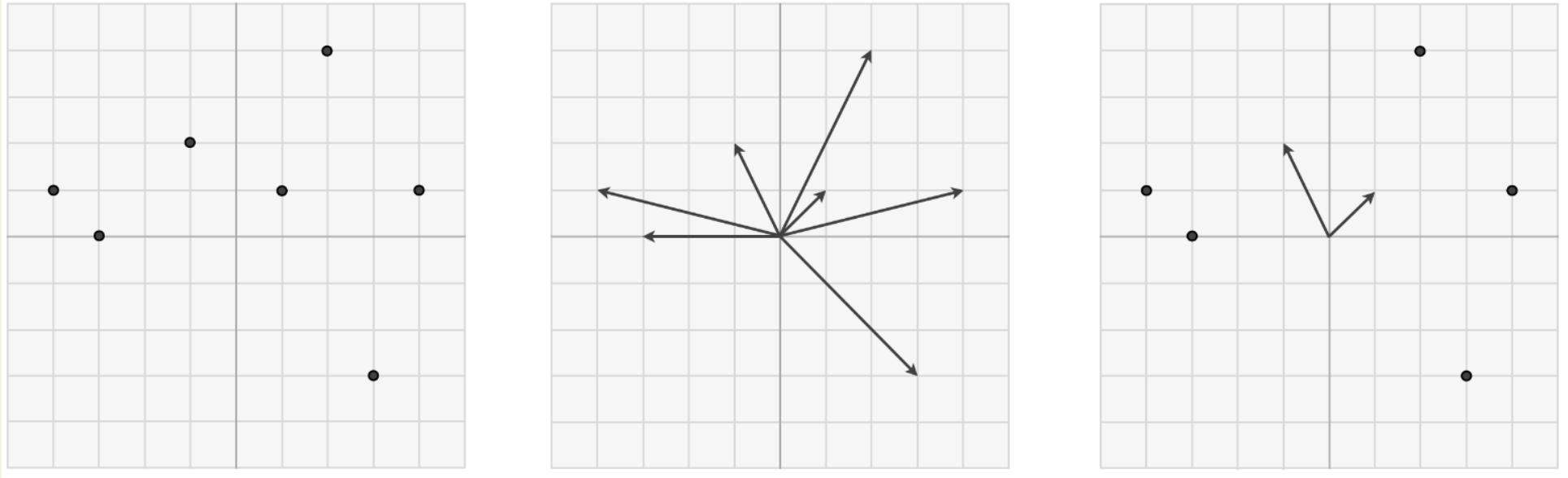
span the plane, because any point (x, y) can be written as

$$x\mathbf{e}_1 + y\mathbf{e}_2.$$

Basis

A *basis* is a **minimal spanning set**: the smallest collection of vectors that still spans the space, with the condition that no vector in the set is redundant (they must be *linearly independent*).

In 2D, the pair $\{\mathbf{e}_1, \mathbf{e}_2\}$ is a basis.



Basis

In order for our basis to be capable of perfectly representing all P of our points it too must live in the same N dimensional space.

For a candidate basis $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N$ to be capable of perfectly representing such generic N dimensional (input) data means that for each data point a set of weights exists so that, in particular linear combination, our basis set can match each data point as

$$\sum_{n=1}^{N} \mathbf{c}_n w_{n,p} = \mathbf{x}_p \qquad\qquad p = 1 \ldots P.$$

Standard Basis

Each standard basis vector consists of zeros everywhere except for a 1 in the k-th position: c_k=[0,0,...,1,...,0] where the 1 is in the k-th slot. Key properties:

$$\mathbf{c}_n^T \mathbf{c}_m = 0$$

$$\|\mathbf{c}_n\|_2^2 = 1 \quad \text{for } n = 1...N.$$

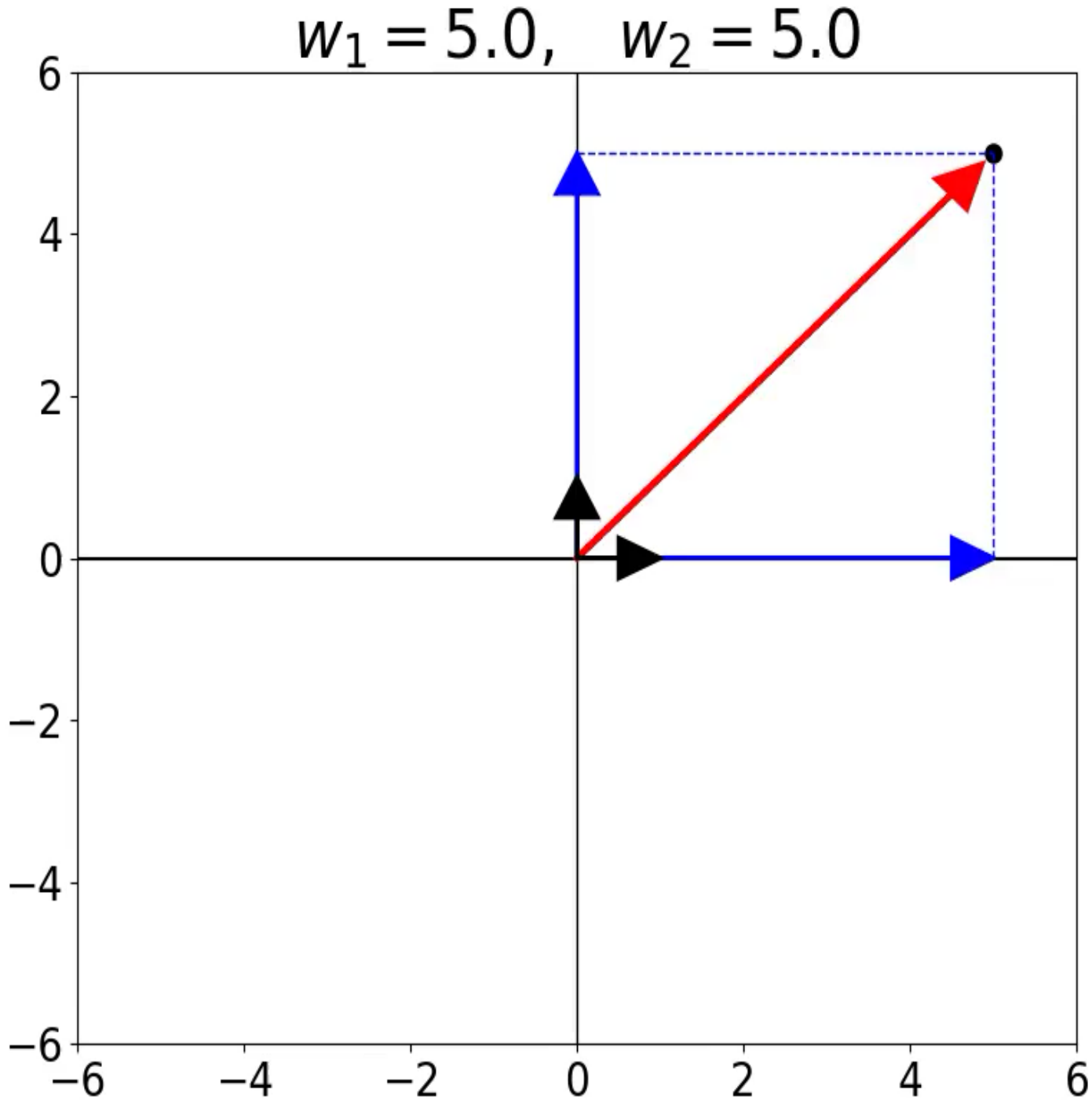
To represent a data point $\mathbf{x}_p$ over the standard basis is a trivial affair, and one can easily check that the perfect weights must be defined as $\mathbf{w}_p = \mathbf{x}_p$ i.e., each weight is simply equal to the value of the data point we aim to represent.

Standard Basis

Note that the same constraints expressed in terms of the concatenated basis matrix $\mathbf{C}$ can be written compactly as:

$$\mathbf{C}^T\mathbf{C} = \mathbf{I}_{N\times N}.$$

Often such a spanning sets are likewise referred to as *orthonormal*.


$w_1 = 5.0, \quad w_2 = 5.0$
6
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6

Standard Basis

Let’s express the particular weights over which a dataset of P points is represented over an orthonormal basis.

Thus the "system of equations" providing the ideal weight vector or *encoding* $\mathbf{w}_p$ or each data point $\mathbf{x}_p$ over an orthonormal basis of vectors is given simply by

$$\mathbf{w}_p = \mathbf{C}^T \mathbf{x}_p \qquad\qquad p = 1 \ldots P.$$

Standard Basis

In other words, if a basis is orthonormal there is no system of equations to solve for - we get the solution (the encoded version of each point) as a simple inner product of the spanning set against each data point!

Moreover, we can express our data representation over an orthonormal spanning set in equation very nicely in vector / matrix notation as:

$$\mathbf{C}\,\mathbf{C}^T\mathbf{x}_p = \mathbf{x}_p \qquad\qquad p = 1 \ldots P.$$

Non standard Basis

Otherwise $\mathbf{w}_p$ - typically referred to as the new ***encoding*** of $\mathbf{x}_p$ in the *transformed feature space* whose coordinate axes are defined by the spanning set - is a vector that differs from original data point.

$$\mathbf{w}_p = \begin{bmatrix} w_{1,\,p} \\ w_{2,\,p} \\ \vdots \\ w_{N,\,p} \end{bmatrix}$$

This weight set provides the new representation of $\mathbf{x}_p$ with respect to the spanning set.

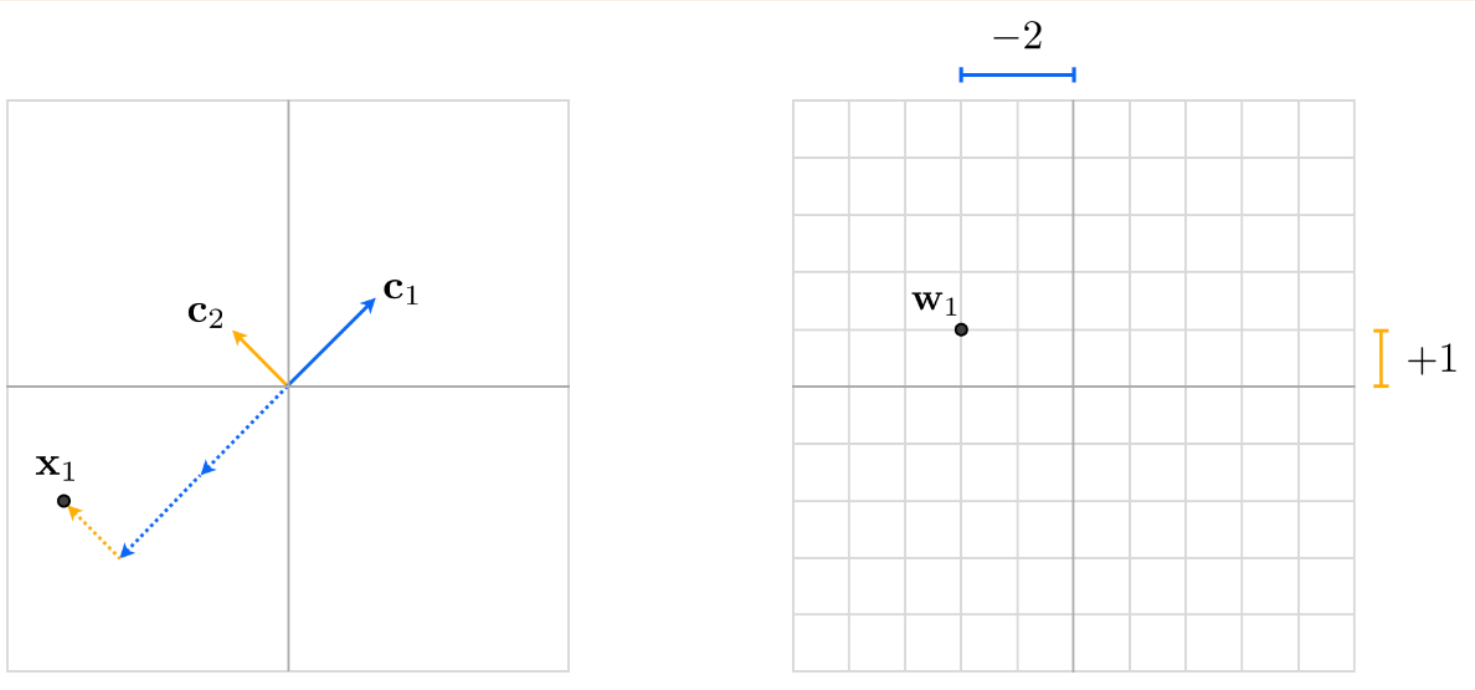
Non Standard Basis

Let's examine an example of a spanning set of vectors in N=2 dimensions.

$$\mathbf{c}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \qquad \mathbf{c}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

(left) A 2-d point, x_1 , shown in the space spanned by vectors c_1 and c_2 .

(right) Representation w_1 , also known as the *encoding*, of x_1 in the transformed feature space.



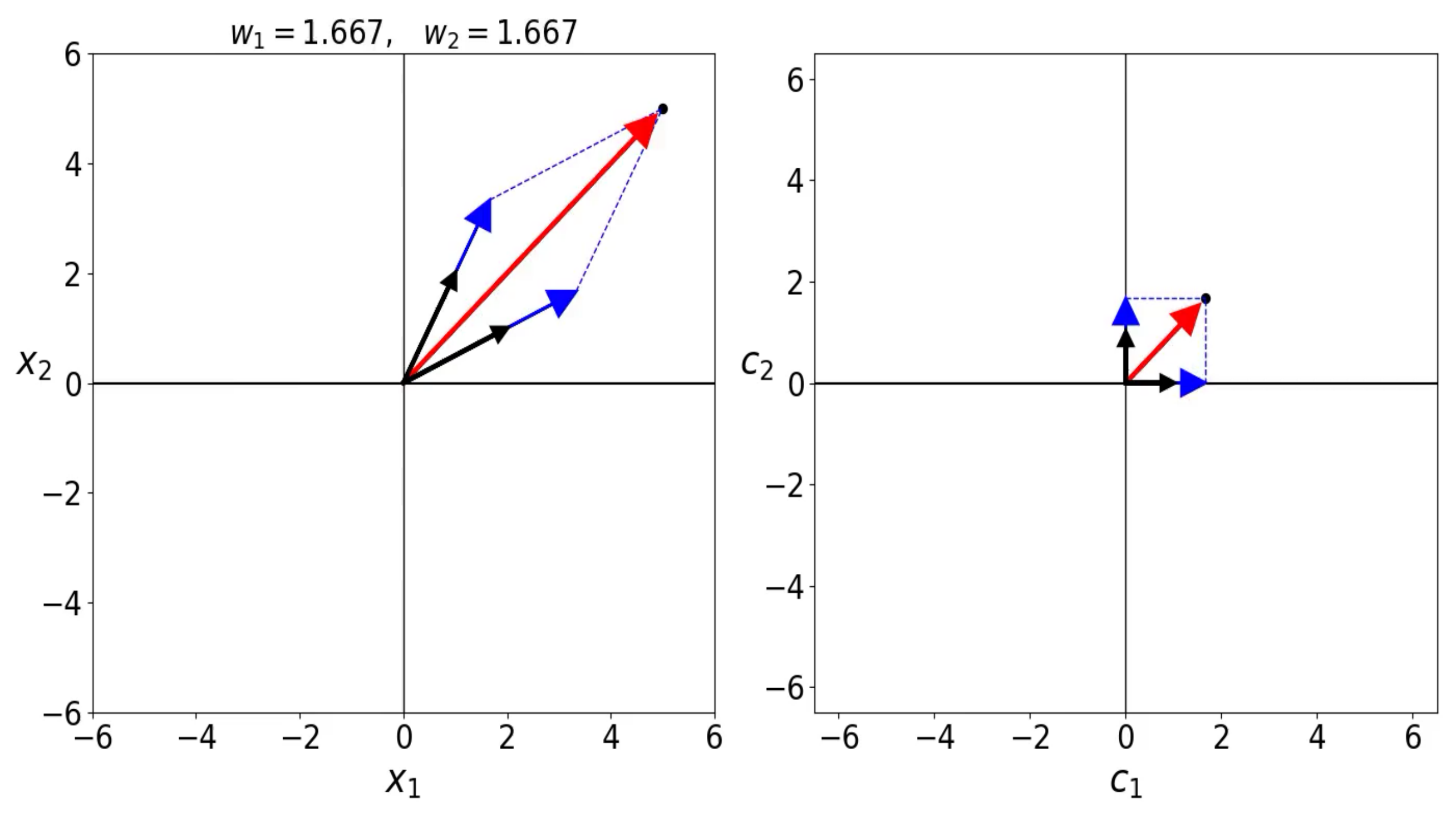
Non Standard Basis

Because these two vectors are *linearly independent*.

That is they do not overlap completely and point in different directions in the space, we can perfectly represent any point in the space using some linear combination of them as:

$$w_{1,p}\mathbf{c}_1 + w_{2,p}\mathbf{c}_2 = \mathbf{x}_p$$

Here the weights $w_{1,p}$ and $w_{2,p}$ are unique to each point $\mathbf{x}_p$.


$w_1 = 1.667, \quad w_2 = 1.667$
x_2
x_1
c_2
c_1
6
4
2
0
−2
−4
−6

Non Standard Basis

Once properly tuned the weight vector $\mathbf{w}_p = \begin{bmatrix} w_{1,\,p} \\ w_{2,\,p} \end{bmatrix}$ provides us with a new representation or encoding of $\mathbf{x}_p$ in the transformed feature space whose coordinates are precisely our spanning set.

In each panel the two spanning set vectors $\mathbf{c}_1$ and $\mathbf{c}_2$ are shown as solid black arrows, and points $\mathbf{x}_p$ (taken over a course range of the input space here) are shown as black dots.

In each space the appropriate linear combination of the two spanning vectors required to represent the point is shown as a red arrow, and each scaled spanning vector is shown as a blue arrow.

Finding W

Here we can in fact directly setup and solve the first order system of equations.

Our goal is tht for each point we minimize $h\left(\mathbf{w}_p\right) = \left\|\mathbf{C}\mathbf{w}_p - \mathbf{x}_p\right\|_2^2$.

Here the $N \times N$ matrix $\mathbf{C}$ is formed by stacking the spanning set vectors column-wise, and the $N \times 1$ vectors $\mathbf{w}_p$ as:

$$\mathbf{C} = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{c}_1 & \mathbf{c}_2 & & \mathbf{c}_N \\ | & | & \cdots & | \end{bmatrix} \qquad \mathbf{w}_p = \begin{bmatrix} w_{p,1} \\ w_{p,2} \\ \vdots \\ w_{p,N} \end{bmatrix}$$

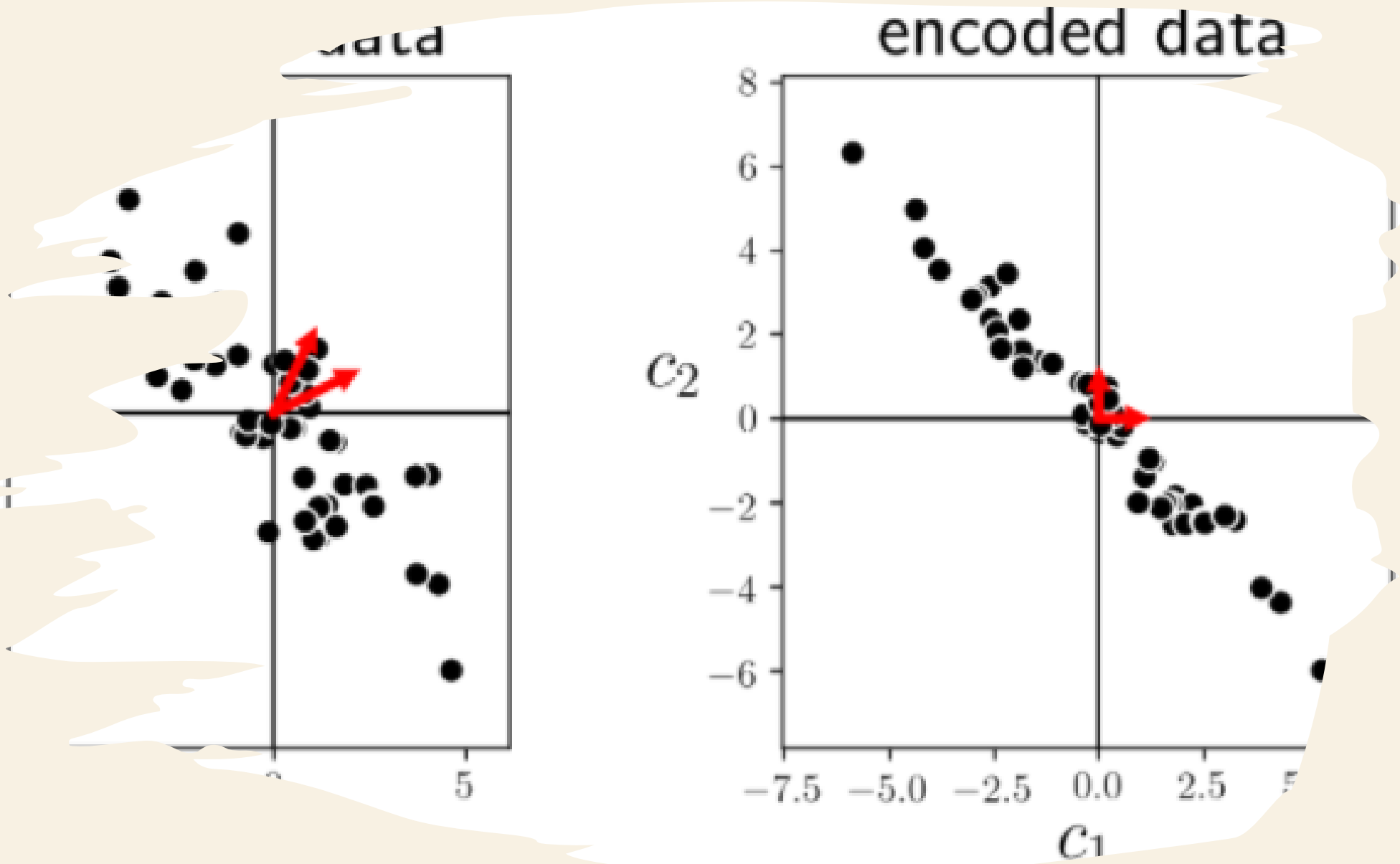
Finding W

Now setting the gradient of the cost function to zero and solving for gives the linear symmetric system of equations for each embedding.

$$\mathbf{C}^T\mathbf{C}\,\mathbf{w}_p = \mathbf{C}^T\mathbf{x}_p \qquad\qquad p = 1...P.$$

We can take the average of the individual single-point cost functions. Then, our optimization problem to recover all of the proper weight vectors can be expressed as:

$$\underset{\mathbf{w}_1,\mathbf{w}_2,...,\mathbf{w}_P}{\text{minimize}} \;\; \frac{1}{P}\sum_{p=1}^{P} \left\|\mathbf{C}\mathbf{w}_p - \mathbf{x}_p\right\|_2^2.$$


encoded data
8
6
4
2
0
−2
−4
−6
c_2
−7.5
−5.0
−2.5
0.0
2.5
c_1
5

Up to now

Above we reviewed two important facts that we require for a spanning set / basis in order for it to be capable of perfectly representing points in a generic N dimensional space:

The set of vectors are *linearly independent*, that is they point in different directions in the space

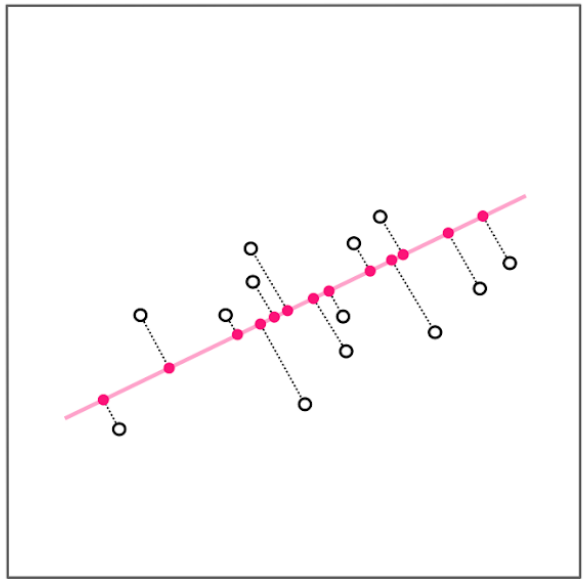
There are at least N spanning vectors

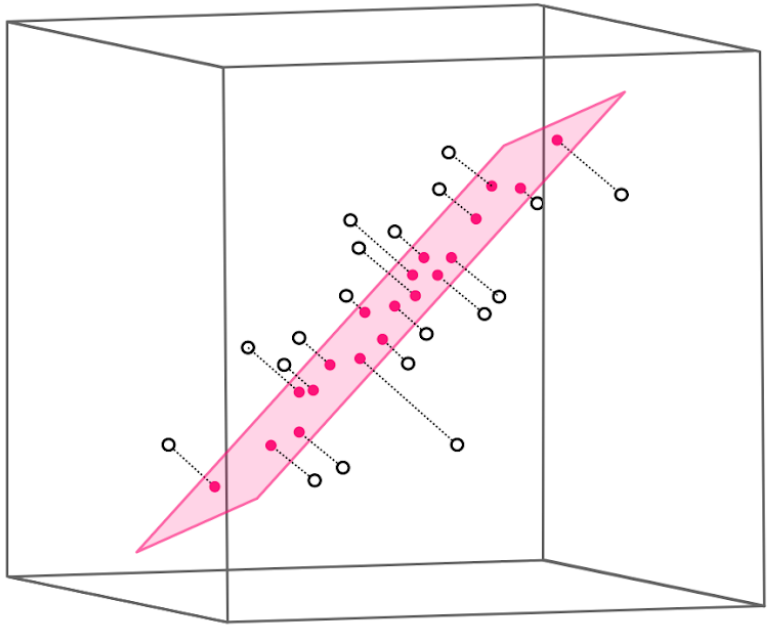
Subspace

What happens when we relax the second condition and suppose we more generally have $K \leq N$ spanning vectors / basis elements?

Unsurprisingly we can no longer perfectly represent a generic set of P points in the space.

In N dimensions this set of K vectors can - at best - span a K dimensional subspace.





If N=3 the best any set of K=2 spanning vectors can do is span a hyperplane.

A set consisting of a single K = 1 spanning vector can only span a line.

In both instances we clearly will not be able to perfectly represent all possible points in a space, since our reach in each case is restricted to a lower dimensional *subspace* of the full space.

Subspace

We have a similar looking *first order system* of symmetric equations providing our ideal weights or encoding vectors:

$$\mathbf{C}^T\mathbf{C}\,\mathbf{w}_p = \mathbf{C}^T\mathbf{x}_p \qquad\qquad p = 1 \ldots P.$$

Here there is of course only one caveat: since we have only K spanning set vectors the matrix $\mathbf{C}$ is $N \times K$, consisting of only our K spanning vectors lined up column-wise:

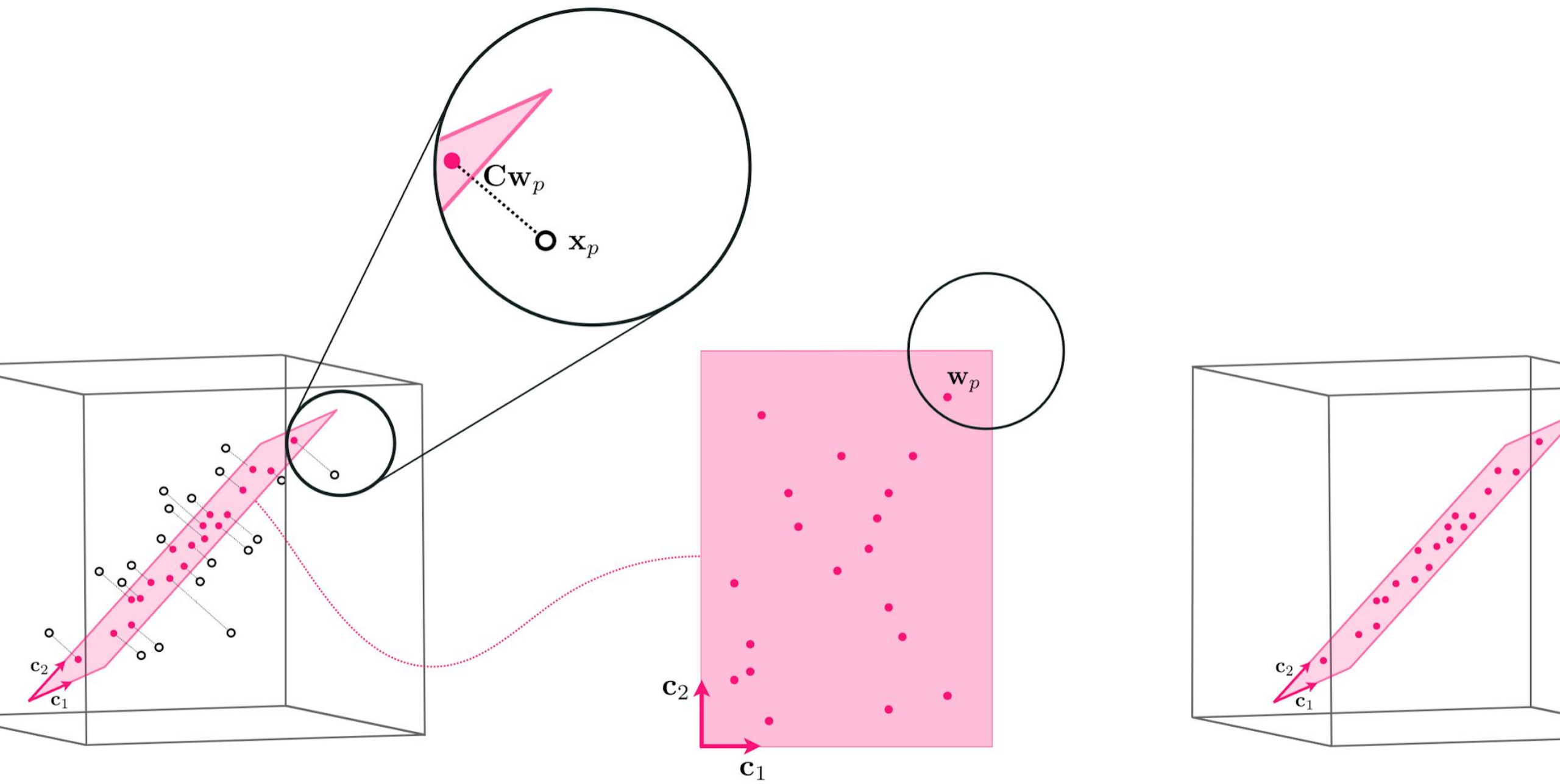
$$\mathbf{C} = \begin{bmatrix} | & | & \cdots & | \\ \mathbf{c}_1 & \mathbf{c}_2 & & \mathbf{c}_K \\ | & | & \cdots & | \end{bmatrix}$$

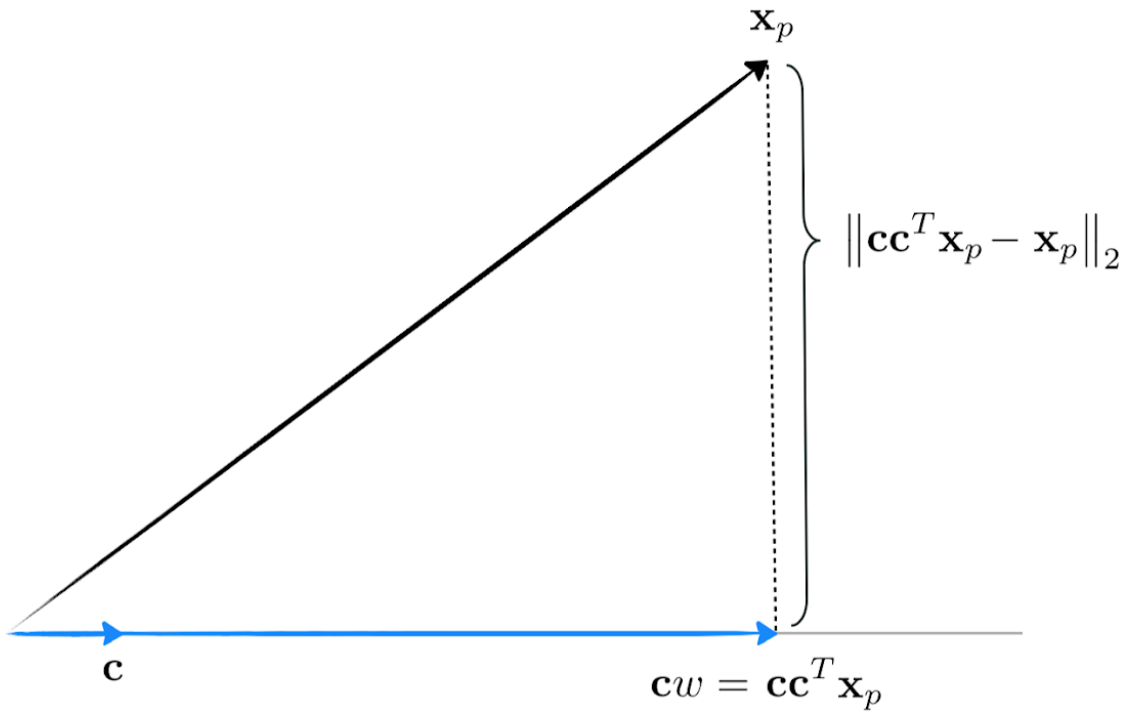
Subspace

This process is called a **projection**, because it is obtained by "dropping" a point perpendicularly onto the subspace spanned by the K basis elements.

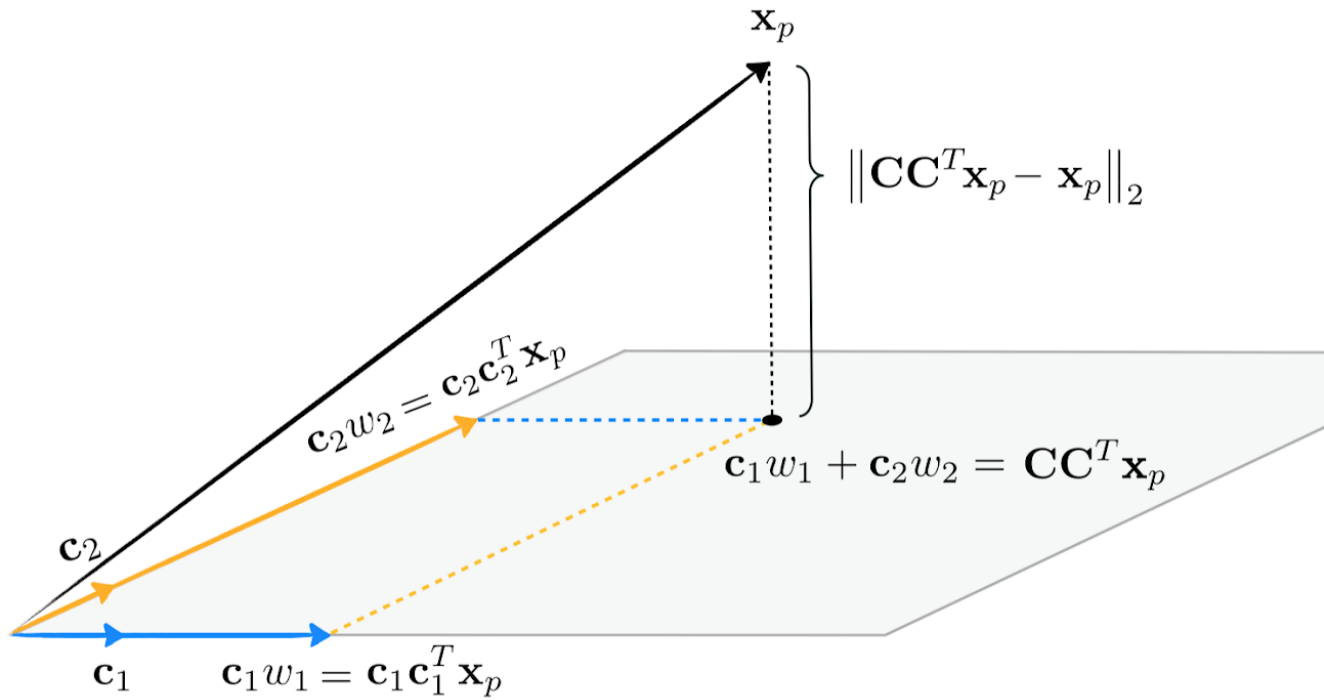
The resulting weight vector is the **encoding** of the point in this subspace.

The **decoding** step reconstructs the point by combining the encoding with the K basis vectors, giving the **projected point**, the closest approximation of the original data point within the subspace.


$\mathbf{C}\mathbf{w}_p$
$\mathbf{x}_p$
$\mathbf{w}_p$
$\mathbf{c}_2$
$\mathbf{c}_1$
$\mathbf{c}_2$
$\mathbf{c}_1$
$\mathbf{c}_2$
$\mathbf{c}_1$
original
encoded
decoded



$\|\mathbf{c}\|_2 = 1$



$\|\mathbf{c}_1\|_2 = \|\mathbf{c}_2\|_2 = 1$

$\mathbf{c}_1^T\mathbf{c}_2 = 0$

If our basis of K elements is orthonormal the above formula for each weight vector encoding reduces to $\mathbf{w}_p = \mathbf{C}^T\mathbf{x}_p \qquad p = 1...P$

$$\mathbf{C}\,\mathbf{C}^T\mathbf{x}_p \approx \mathbf{x}_p \qquad p = 1...P.$$

Slides based upon machine learning refined book chapter 8