PRINCIPAL COMPONENT ANALYSIS

→ Datasets are typically high dimensional (i.e. text, images)

- However, this high dimensionality is not actual; We only get high dimensions in images bc. we choose to make each pixel a separate attribute
- The data underneath doesn't uniformly fill that high dimensional space, but it lies on a lower dimensional subsurface within the space.

↳ True dimensionality is often much lower!

CURSE OF DIMENSIONALITY

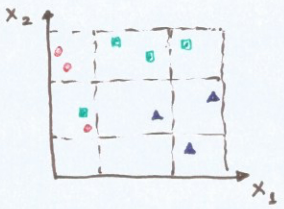
→ Why do you want to reduce dimensionality?

1) High-d data tends to be sparse

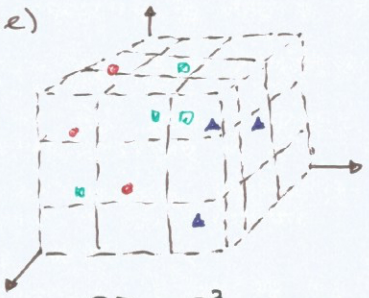
- As dimensionality grows, there will be fewer observations per region (bc. the size of our training data remains the same)

i.e.

1D – 3 regions



2D – 3^2 regions



3D – 3^3 regions

- Not good for many supervised methods
  ↳ ML methods are statistical by nature

↳ Most of the observations are going to be zeros!
We have lots of regions in space but we cannot make any inference at all.

2) Can make non-linearly separable problems into one that is linearly separable

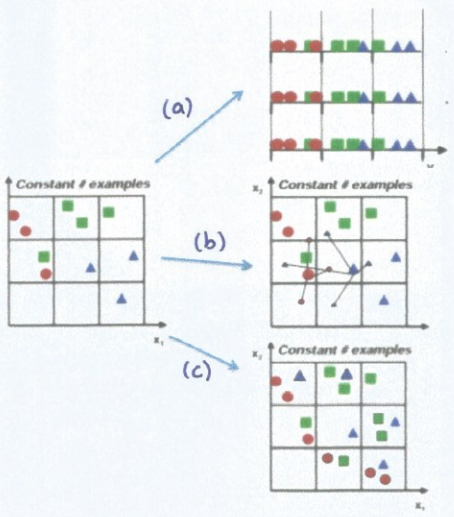
→ How to deal w/ this?

1) Make assumptions about the dimensions

a. INDEPENDENCE
   - Count along each dim. separately

b. SMOOTHNESS
   - Propagate class counts to neighboring regions

c. SYMMETRY
   - e.g. Invariance to order of dimensions
     $X_1 \leftrightarrow X_2$



2) Ignore dimensions which help little in the prediction task

3) Selecting a subset of features using information gain

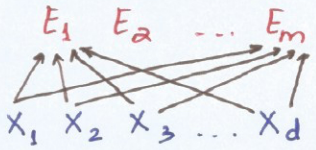
4) Using a set of feature detectors smaller in number than the original dim. of the data

5) Reduce the dimensionality of the data

→ GOAL – represent instances w/ fewer variables

→ We try to preserve as much struct. in the data as possible

→ 2 TYPES
- Feature Selection
- Feature Extraction → Construct a new set of dimensions that are linear combinations of original



PCA

→ defines a set of principal components:

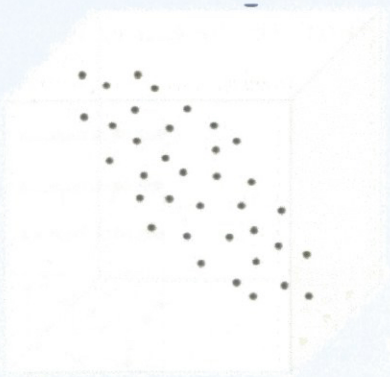
- 1st PC – direction of the greatest variance by any projection of the data
- 2nd PC – perpendicular to 1st PC

⋮ and so on until d (Original dimensionality)

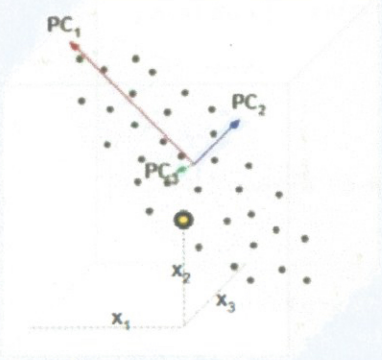
→ The first $m \ll d$ components become m new dimensions

↳ Change co-ordinates of every data pt. to these dimensions

i.e.

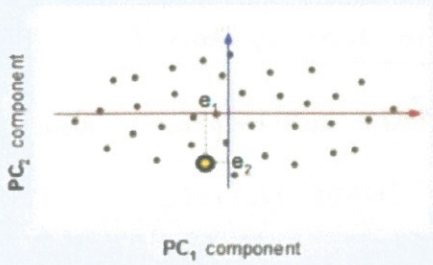


↑ 3D space but all the points lie on a 2D hyperplane within the space



↑ Along PC_1, the data is spread out as much as it can be.

Note that the variance of data along PC_3 is very small.



↑ so we pick PC_1 & PC_2 as our new co-ordinate vectors

STEPS TO PERFORM PCA

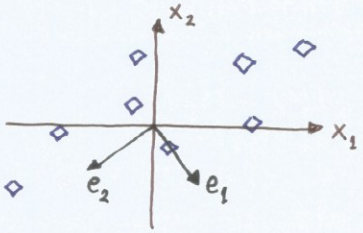
1) Standardisation

→ Starting w/ correlated data, we first transform the data to zero mean and unit variance; 'center' the data at zero.

→ Done by subtracting mean from each attribute

2) Compute the covariance matrix Σ

i.e. For 2D data:



$$\Sigma = \begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cc} x_1 & x_2 \end{array} \\ \begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix} \end{array}$$

$\text{Cov}(x_2, x_1) = \text{Cov}(x_1, x_2)$

$\text{Cov}(x_2, x_2) = \text{Var}(x_2) = \frac{1}{n}\sum_{i=1}^{n} x_{i,2}^2$

$\text{Cov}(x_1, x_2) = \frac{1}{n}\sum_{i=1}^{n} x_{i,1} \cdot x_{i,2}$

↳ The covariance of dimensions x_1 and x_2 tells us about the correlation between these two. (Look at the sign of the covariance)

3) Compute the eigenvectors & eigenvalues

→ Recall: **Eigenvector** of a matrix is a vector which when it is multiplied by the matrix, the result is a multiple of the same vector

Eigenvalues of a matrix are the lengths of the vectors that arise when the eigenvectors are multiplied by the matrix

$$A\vec{x} = \lambda\vec{x}$$

→ Eigenvalues are obtained by solving $\det(\Sigma - \lambda I) = 0$

$$\det\begin{pmatrix} 2.0-\lambda & 0.8 \\ 0.8 & 0.6-\lambda \end{pmatrix} = (2-\lambda)(0.6-\lambda) - (0.8)(0.8)$$

$$= \lambda^2 - 2.6\lambda + 0.56 = 0$$

$$\{\lambda_1, \lambda_2\} = \{2.36,\ 0.23\}$$

→ Find i^{th} eigenvector by solving $\Sigma e_i = \lambda_i e_i$

$$\begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix}\begin{pmatrix} e_{1,1} \\ e_{1,2} \end{pmatrix} = 2.36\begin{pmatrix} e_{1,1} \\ e_{1,2} \end{pmatrix} \rightarrow e_1 = \begin{bmatrix} 2.2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.91 \\ 0.41 \end{bmatrix}$$

Unit vector: $\frac{2.2}{\sqrt{2.2^2+1^2}}$, $\frac{1}{\sqrt{2.2^2+1^2}}$

$$e_2 = \begin{bmatrix} -0.41 \\ 0.91 \end{bmatrix}$$

∴ **1st PC** $= \begin{pmatrix} 0.91 \\ 0.41 \end{pmatrix}$ and **2nd PC** $= \begin{pmatrix} -0.41 \\ 0.91 \end{pmatrix}$

4) Pick the eigenvectors

5) Project the data onto the eigenvectors

→ Say we have instance $x = \{x_1, \ldots, x_d\}$ and eigenvectors $e_1, \ldots, e_m$ which are our new dimension vectors

→ We want to project our instance to the m-dimensional space and obtain new co-ordinates $x' = \{x'_1, \ldots, x'_m\}$

HOW?

1) Center the instance $x - \mu$

2) Project to each dimension $(x-\mu)^T e_j$ for $j = 1 \ldots m$

$$(\vec{x} - \vec{\mu}) = [(x_1 - \mu_1) \quad (x_2 - \mu_2) \quad \cdots \quad (x_d - \mu_d)]$$

$$\begin{bmatrix} x_1' \\ x_2' \\ \vdots \\ x_m' \end{bmatrix} = \begin{bmatrix} (\vec{x} - \vec{\mu})^T e_1 \\ (\vec{x} - \vec{\mu})^T e_2 \\ \vdots \\ (\vec{x} - \vec{\mu})^T e_m \end{bmatrix} = \begin{bmatrix} (x_1 - \mu_1) e_{1,1} + \cdots + (x_d - \mu_d) e_{1,d} \\ (x_1 - \mu_1) e_{2,1} + \cdots + (x_d - \mu_d) e_{2,d} \\ \vdots \\ (x_1 - \mu_1) e_{m,1} + \cdots + (x_d - \mu_d) e_{m,d} \end{bmatrix}$$

HOW MANY PRINCIPAL COMPONENTS TO USE ?

→ So we have eigenvectors $e_1, \ldots, e_d$ and we want $m \ll d$

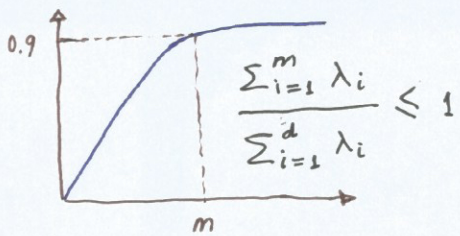
→ To find m, we pick eigenvectors that 'explain' the most variance:

APPROACH #1

1) Sort eigenvectors by eigenvalues such that $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$

2) Pick first m eigenvectors which account for over 90% of the total variance

typical threshold: 0.9 or 0.95



$$\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \leq 1$$

→ Alternatively, we can form a scree plot of the eigenvalues and pick the ones above the knee and use the associated eigenvectors as PCs

APPROACH #2