



MACHINE LEARNING

MEI/1

University of Beira Interior,
Department of Informatics

Hugo Pedro Proença,
hugomcp@di.ubi.pt, 2026/2027



Machine Learning

Week 05

Syllabus

- Feature Representation
- Dimensionality Reduction
- PCA
 - Eigenvectors and Eigenvalues
- PCA vs LDA

Feature Representation

- Raw input data (e.g., high-dimensional images, or other kinds of signals) often contains redundant or irrelevant information.
- Feeding high-dimensional raw data directly to the model increases:
 - The number of parameters **exponentially**.
 - The risk of overfitting.
 - The computational cost of training.
 - The effects of the **Curse of Dimensionality** → sparse data in high-dimensional spaces.
- **Feature Extraction** transforms “Raw Data” into a meaningful, “**Lower-dimensional Representation**”, making models more efficient, accurate, and robust.

Features:

“Woman”

1,62m

“Blue Eyes”

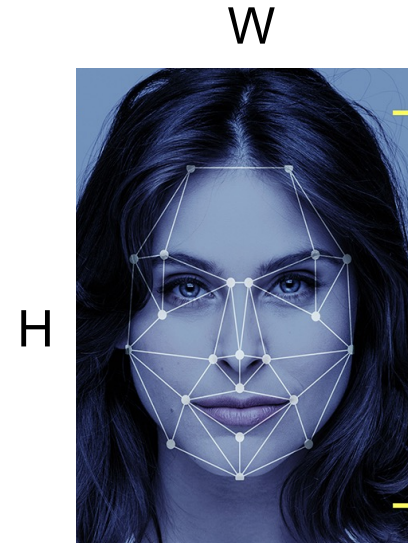
“Long Hair”

“Young”

“Black Hair”

50kg

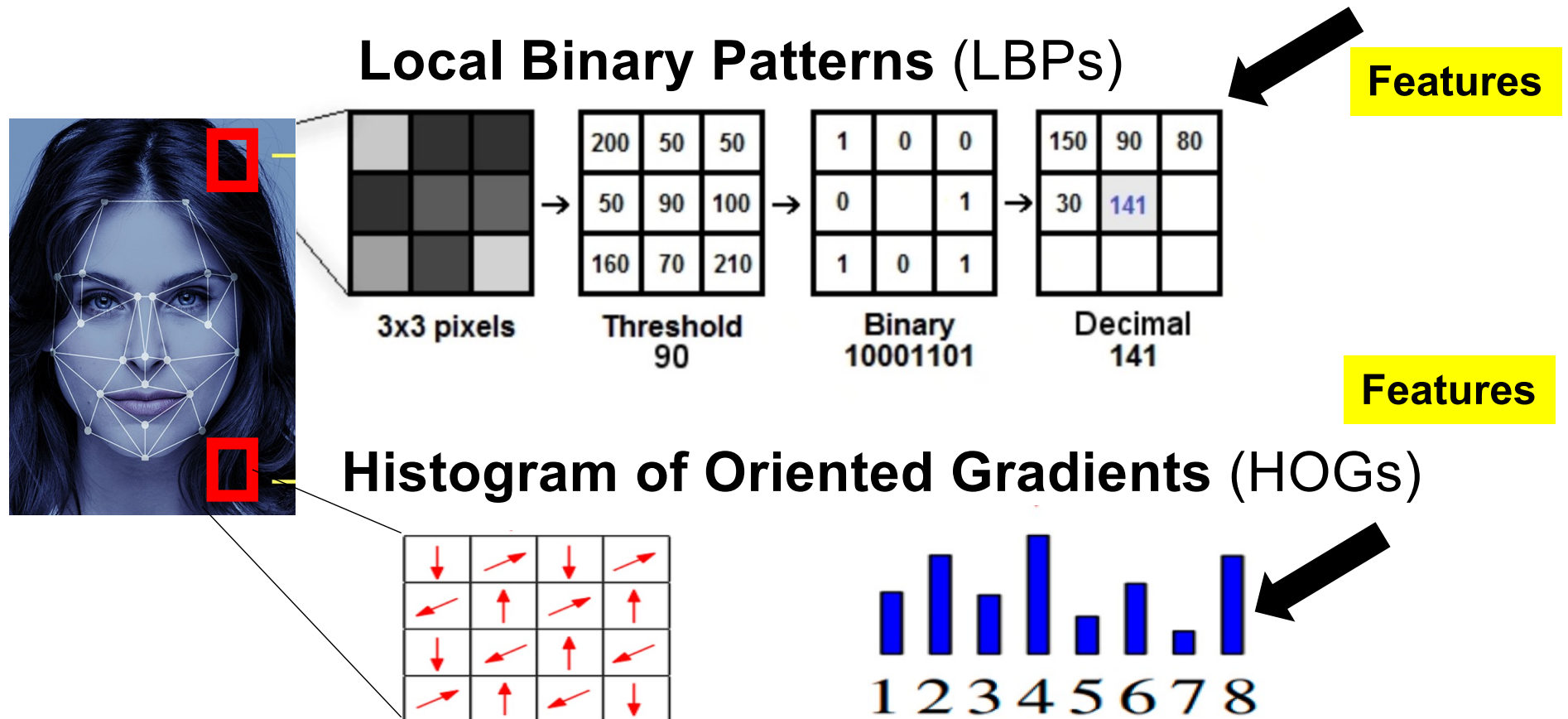
$\mu_{\text{intensity}} = 154$



H x W might lead to an “**excessive high hyperspace**”



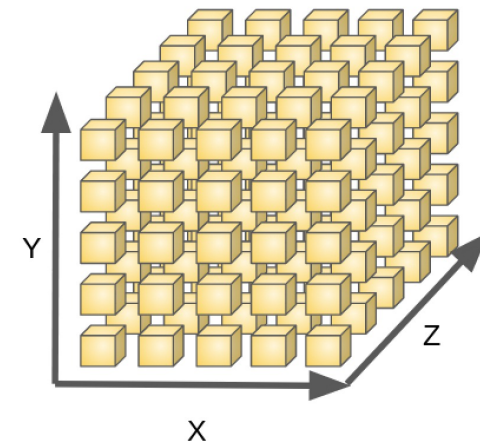
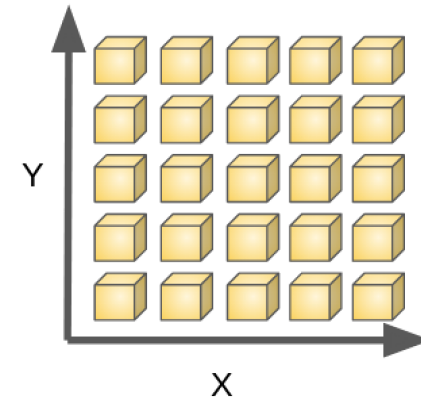
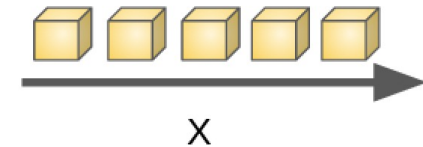
Feature Representation



Hu Moments, Chain codes, Shape Context, SIFT, SURF, Statistical (e.g., mean, std, kurtosis,...),....

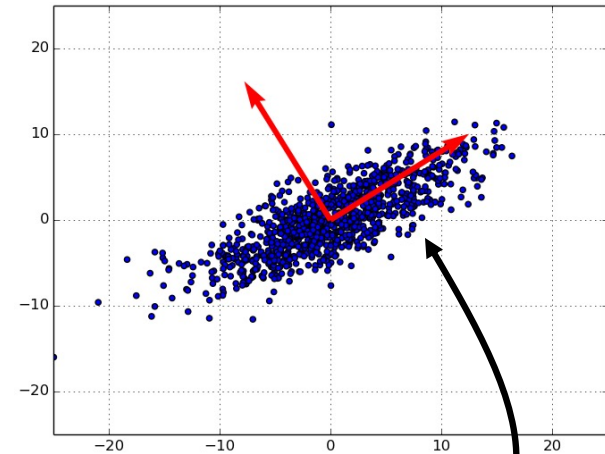
Dimensionality Reduction

- The **curse of dimensionality** is one of the most classical phenomena in the development of Machine Learning systems.
 - In short, when the **dimensionality increases**, the volume of the space increases so fast that the data become **sparse**.
 - **Sparsity is problematic** for any method that requires statistical significance, i.e., densely populated spaces.
- For example, consider 100 evenly spaced sample points (instances) inside a unit interval.
 - On average, points will be separated around $10^{-2}=0.01$
- An equivalent sampling that will yield similar density in a 10-dimensional unit hypercube would require $10^{20}[(10^2)^{10}]$ sample points



Dimensionality Reduction

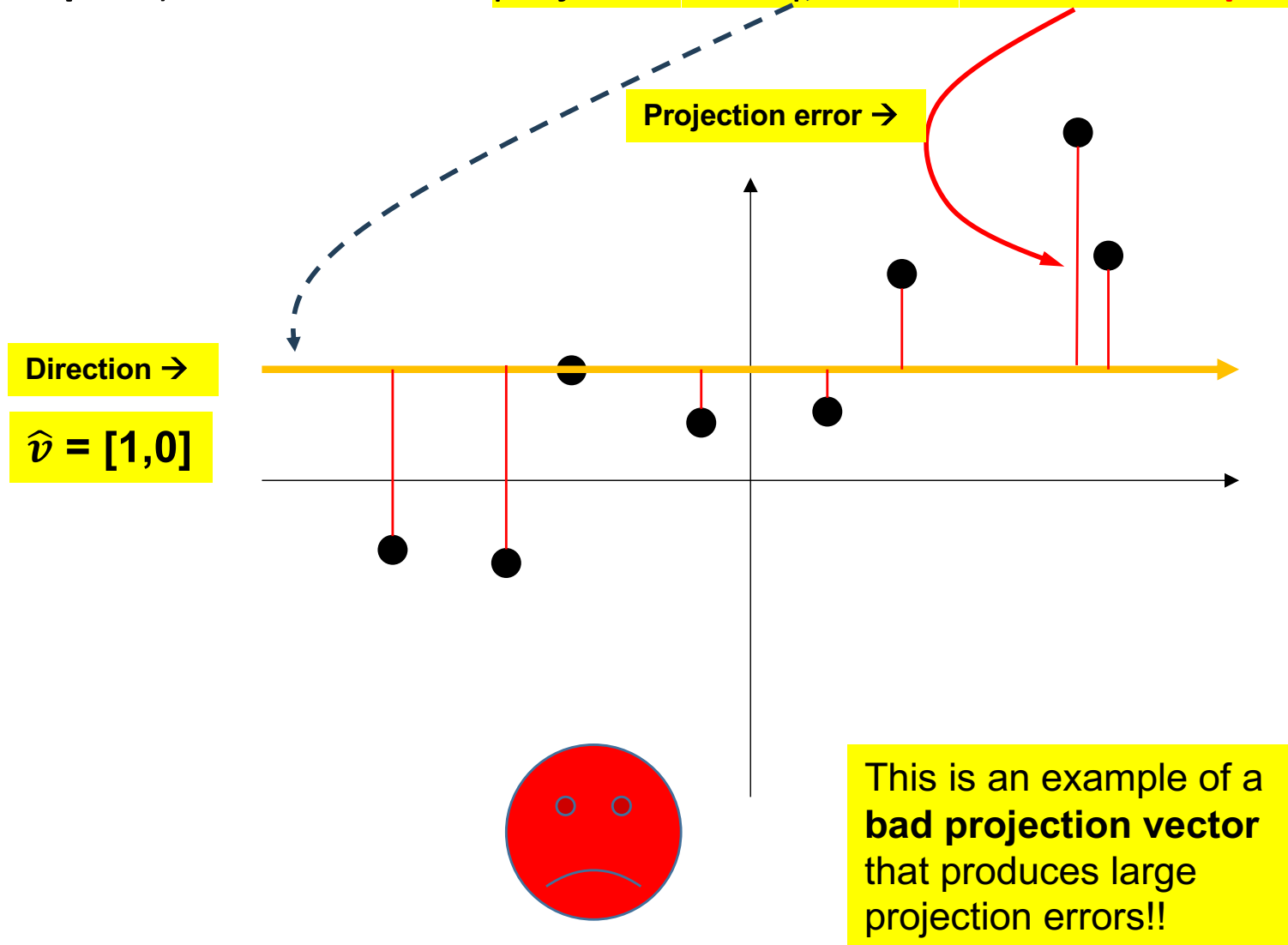
- In statistics, machine learning, and information theory, dimensionality reduction is the process of **reducing the number of random variables** under consideration by obtaining a set of principal variables.
- In general, there are two families of methods to reduce the dimensionality of a data set:
 - **Feature Selection**. The idea is to find a subset of the original features that better represent the problem, i.e., that minimally decrease the amount of available information, when compared to the original dataset.
 - Most approaches are based in filters (based in information gain), wrappers (based in accuracy) and embedded (features iteratively selected/removed according to prediction errors)
 - **Feature Extraction**. It is often also designated as “**Feature Projection**” and the idea is to transform the original feature space into a space of fewer dimensions, while keeping as much of the original information as we can.
 - **Principal Component Analysis (PCA)** is the main technique in this family.



The key idea is to find the direction(s) (vector(s)) onto which data maximally **span**

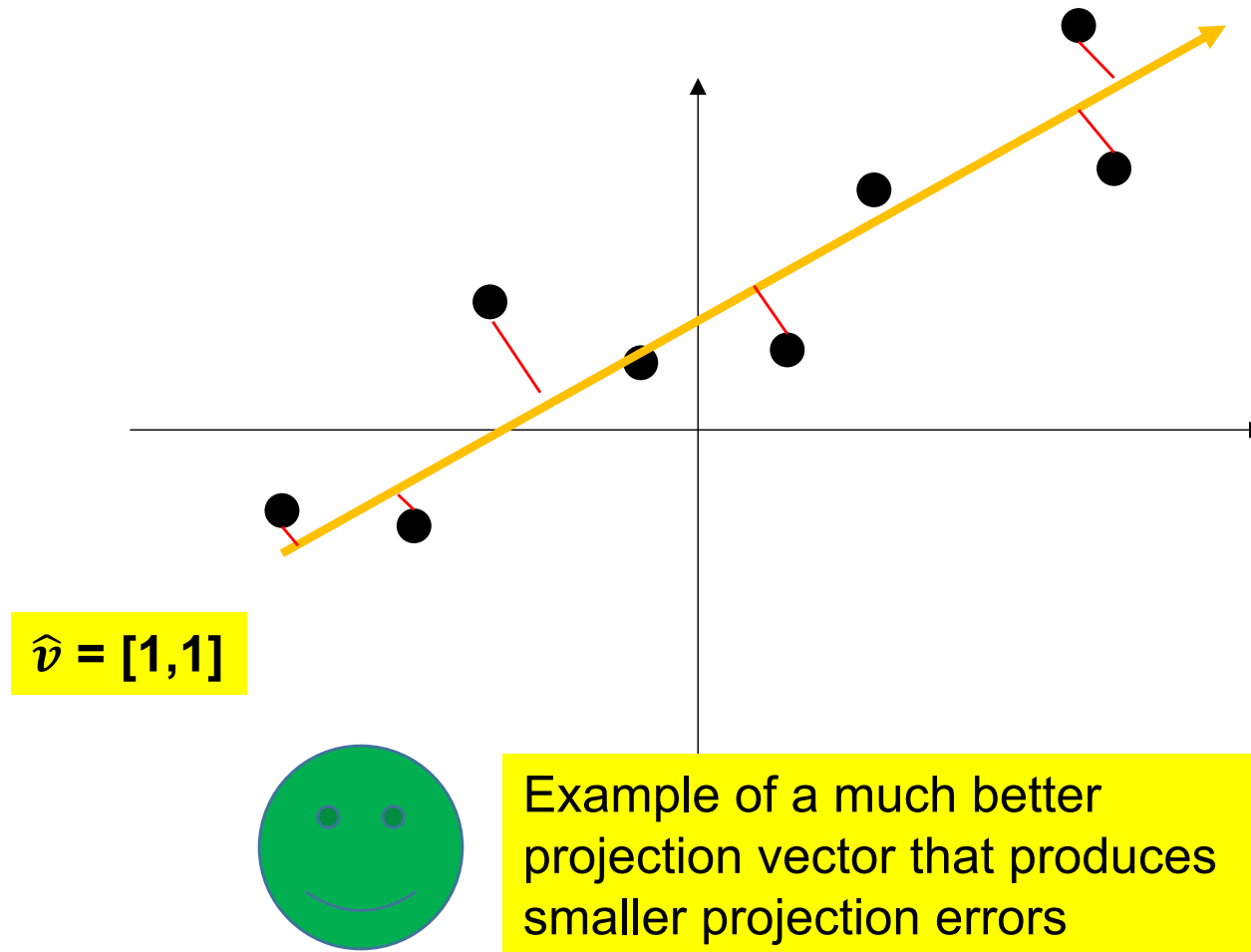
PCA

- Graphically, we are interested in finding the **direction (vector in the original space)** onto which the projected data provides the **minimal projection error**:



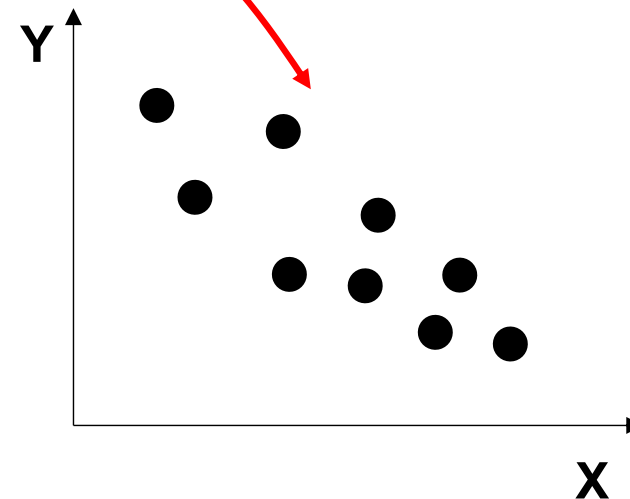
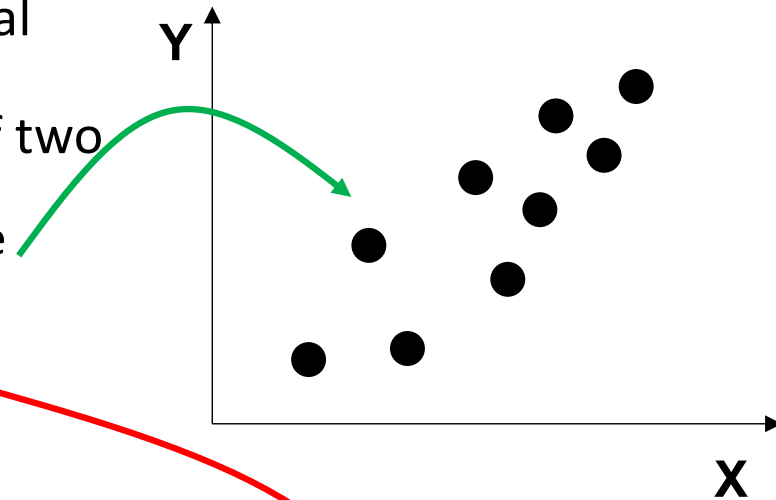
PCA

- Graphically, we are interested in finding the **direction (vector in the original space)** onto which the **projected data have minimal projection error**:



PCA: Covariance

- The **covariance** can be obtained for any two dimensions (features) of a n-dimensional feature space
- It is a measure of the **joint variability** of two features
 - If both variables **vary** in a **direct** way, the covariance is **positive**
 - On the contrary, if both variables **vary inversely**, the variance values will be **negative**.
- The sign of the covariance shows the tendency in the **linear relationship** between the variables.
- The magnitude of the covariance is not easy to interpret because it is not normalized and hence depends on the magnitudes of the variables.
- The normalized version of the covariance, the correlation coefficient, however, shows by its magnitude the strength of the linear relation.



PCA: Covariance

- The distance between sample points and their mean is multiplied. Then, the result is divided by the number of data points minus 1:

$$cov(X, Y) = \frac{\sum_{i=1}^n (X_i - X^*)(Y_i - Y^*)}{n - 1}$$

where X_i , Y_i are the i th data points, X^* , Y^* are the sample means and “ n ” is the number of data points.

- The results is meaningful essentially by analysing it's sign:
 - Positive: Both dimensions **vary directly**.
 - Negative: Both dimensions **vary inversely**.
 - Zero: Dimensions are **independent**.

PCA: Covariance Matrix

- The **Covariance Matrix C** contains all covariance pair values between every possible dimensions of a feature space :

$$C = [c_{ij} \mid c_{ij} = cov(X_i, X_j)]$$

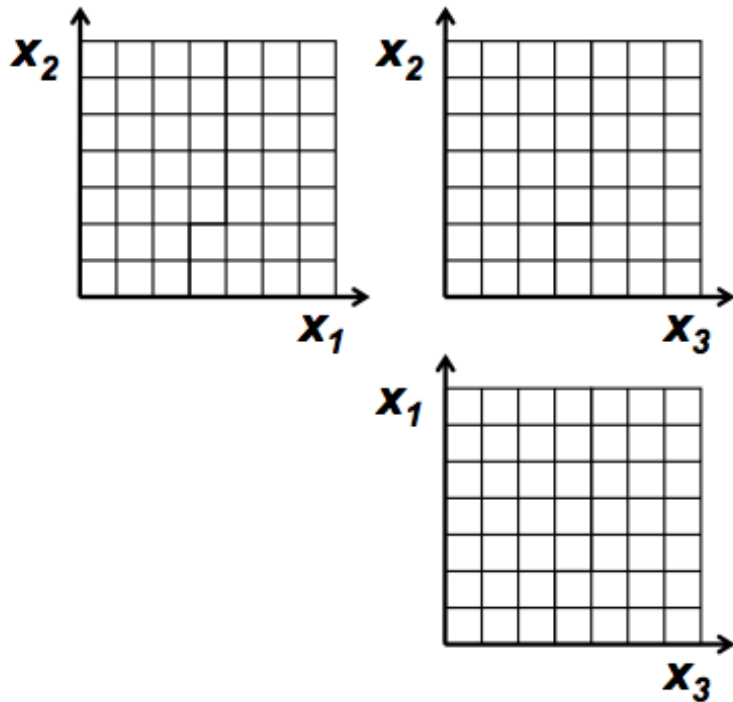
- For exemple, considering a three dimensional space $\{X, Y, Z\}$, the covariance matrix will correspond to:

$$\begin{bmatrix} cov(X, X) & cov(X, Y) & cov(X, Z) \\ cov(Y, X) & cov(Y, Y) & cov(Y, Z) \\ cov(Z, X) & cov(Z, Y) & cov(Z, Z) \end{bmatrix}$$

- Values along the **main diagonal** describe the **variance** of the corresponding dimension.
- Based on its definition, it is obvious that $cov(X, Y) = cov(Y, X)$, i.e., the covariance matrix is symetric with respect to its main diagonal.

PCA: Covariance Matrix

- **Exercise.** Obtain the covariance matrix for the given data set:



Obs.	X1	X2	X3
1	2	2	4
2	3	4	6
3	5	4	2
4	6	6	4

[illegible]

Eigenvectors and eigenvalues

- Consider the multiplication of a **matrix** by a **vector**:


$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 3 \end{pmatrix} = \begin{pmatrix} 11 \\ 5 \end{pmatrix}$$

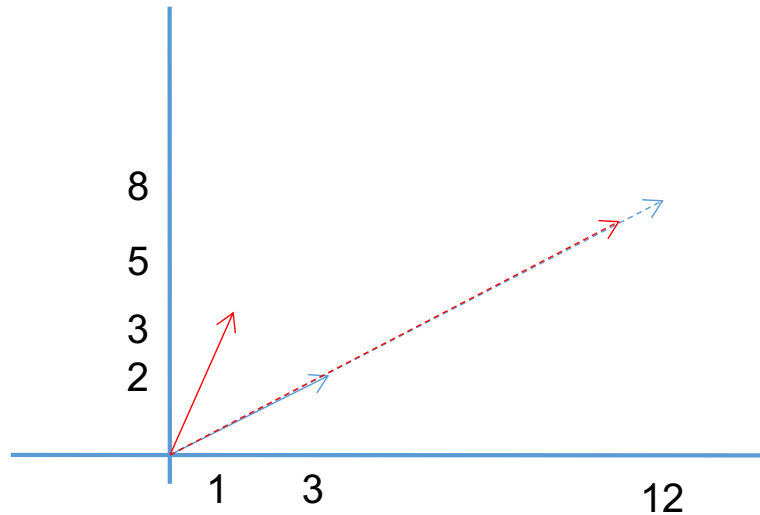
- However, there are some **particularly interesting vectors**:

$$\begin{pmatrix} 2 & 3 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 2 \end{pmatrix} = \begin{pmatrix} 12 \\ 8 \end{pmatrix} = 4 \begin{pmatrix} 3 \\ 2 \end{pmatrix}$$

- In the first case, the resulting vector is not a multiple of the original vector.
- Oppositely, in the second case, the resulting vector (12,8) is a multiple of the multiplier.
- As such, the latter is an **eigenvector**.
 - The corresponding **eigenvalue** is “4”

Eigenvectors and eigenvalues

- By analysing the direction of the original and resultant vectors:



- Considering the matrix as a **transformation**, it can be concluded that in the second case, the direction was not changed. **This is the key insight the notion of eigenvector.**
 - The given matrix does not change the direction of its eigenvectors.

Eigenvectors and eigenvalues

- As we've seen, the notion of **eigenvalue** is strongly related to the **eigenvector**.
 - It is the value that should be multiplied by the eigenvector to obtain the original vector.
 - In the above example, 4 was the eigenvalue that corresponds to the given eigenvector.
- As such, eigenvalues and eigenvectors come in pairs and are always inter-related.

Eigenvectors and eigenvalues

- As a summary, the eigenvectors of a matrix correspond to the directions that are not changed by the (transformation) matrix.
- Not all matrices have eigenvectors.
- Matrices have to be square.
- A ($n \times n$) matrix has – at most – “ n ” eigenvectors.
- The set of eigenvectors of a matrix (image) is widely used to describe the spatial content of that image (feature).
- In MATLAB, this eigenanalysis is made by the “`eig()`” function:
 - $[V, D] = \text{eig}(A)$
 - Returns the eigenvectors (D) and corresponding eigenvalues (V) of matrix A .
- In Python, this can simply be done by:
 - $V, D = \text{LA.eig}(A)$

Eigenvectors and eigenvalues

- There is an important property to be stressed: **the eigenvectors of a matrix are orthogonal**. This is to say that they form an **orthogonal basis** of the matrix.
 - We are able to express every point of a data set by linear combinations of its basis-vectors.
 - **This is specially useful for the analysis of principal components (PCA).**
 - It is usual to determine the eigenvectors/eigenvalues in their normalized version, i.e., with length normalized to 1.
 - As previously seen, the length of a vector does not affect its property of being (or not) an eigenvector.
 - Hence, having an eigenvector $(x_1, \dots, x_n)$ it is usual to divide each component by the norm of this vector, in order to obtain length “1”:
 - $|| (x_1, \dots, x_n) || = \sqrt{x_1^2 + \dots + x_n^2}$

Eigenvectors and eigenvalues

- **Exercise**

- Determine, from the following vectors, which are eigenvectors of the matrix given below and, if positive, determine the corresponding eigenvalue.

- Matrix:

$$\begin{bmatrix} 3 & 0 & 1 \\ -4 & 1 & 2 \\ -6 & 0 & -2 \end{bmatrix}$$

- Vectors:

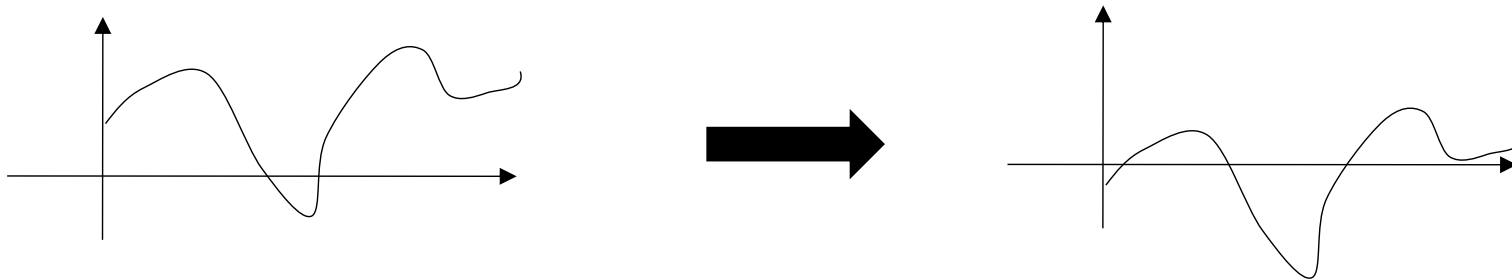
$$\begin{array}{ccccc} \begin{bmatrix} 2 \\ 2 \\ -1 \end{bmatrix} & \begin{bmatrix} -1 \\ 0 \\ 2 \end{bmatrix} & \begin{bmatrix} -1 \\ 1 \\ 3 \end{bmatrix} & \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} & \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \end{array}$$

PCA: Principal Component Analysis

- **The Principal Component Analysis (PCA)** it's a well known way to detect patterns on data, by expressing it on a way that enhances similarities or differences.
- Detecting patterns on high dimensional data is a hard task, either for humans or machines.
 - Requires huge amounts of data. An empirical rule says that at the minimum, d^2 instances are required to analyze a d -dimensional data set.
- PCA is typically used to compress data (reduce dimensionality), without losing substantial information.

PCA: Principal Component Analysis

- **Step 1.** The analysis of principal components requires a data set (with dimension n) and cardinality (k).
- **Step 2.** Removal of energy. For each dimension, the corresponding mean is subtracted to each component. As such, all dimensions of the data set have zero energy.



Principal Component Analysis

- **Step 3.** Calculus of the covariance matrix. Here, the relationships between independent components are detected, together with an assessment of the data dispersion in each dimension (by analysing the main diagonal components).
- **Step 4.** As the covariance matrix is square, it is possible to obtain the set of eigenvectors and corresponding eigenvalues.
- **Step 4.1.** Eigenvectors normalization. All eigenvectors are normalized to have norm equal to 1.

Principal Component Analysis

- **Step 5.** Selection of components. The set of eigenvectors is sorted by decreasing order, considering the corresponding eigenvalues. From this set, the “ k_1 ” principal components are selected.
 - This is the step that performs the reduction of dimensionality.
- **Step 6.** A transformation matrix is built, by concatenating the eigenvectors selected in the previous step.
 - This matrix will be used to represent all points in the reduced dimensionality feature space.
MAT=[vect1, vect2, ... Vect k_1]

Principal Component Analysis

- **Step 7. Data Transformation.** As the transformation matrix has “d” lines (corresponding to the dimension of the original feature space and k_1 columns (corresponding to the dimension of the new feature space), when multiplying each original data point by the transformation matrix, we obtain a vector of k_1 components. These are the new representation of the data points, in the principal components space.

$$[1 \times d] \times [d \times k_1] = [1 \times k_1]$$

Principal Component Analysis

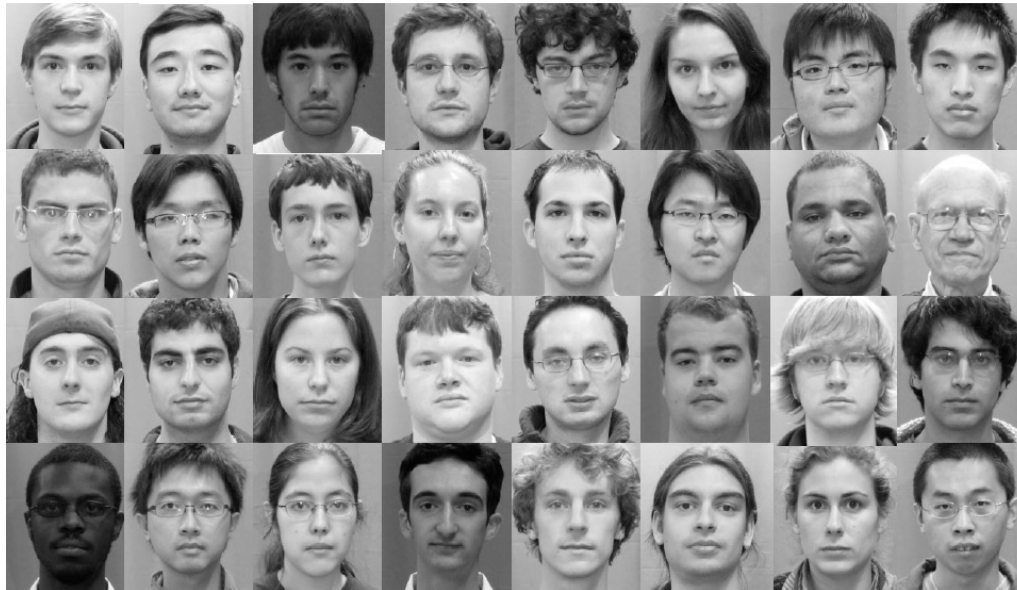
- **How to choose the value of “k”?**

- The previously described process does not give any information about a strategy to select the dimensionality of the principal components feature space.
- There is no formal rule. However, some heuristics about what is generally better exist.
- Usually, the variation in magnitude of consecutive eigenvalues (after sorting) is measured. When changes in magnitude are higher than a threshold, the selection process is stopped.
- But most frequently, the proportion of the data variability that is kept by the selected components is considered as the main criterium.
 - Typically, we are interested in keeping around 90, 95, 99% of the original data variability.
 - The analysis can be done by measuring the proportion of the sum of eigenvalues:

- **Variability:** $\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^d \lambda_i}$, “k”: number of selected vectors, and “d”: dimensionality

PCA: Example

- Consider a set of 128 face grayscale images (with dimensions 64 x 64).
 - Each image is represented by a 64×64 matrix = (4096), where each position represents the intensity at a point (0: black pixel, ... 255: White pixel)
- Each face can be regarded as a point represented in a 4096 dimensional feature space



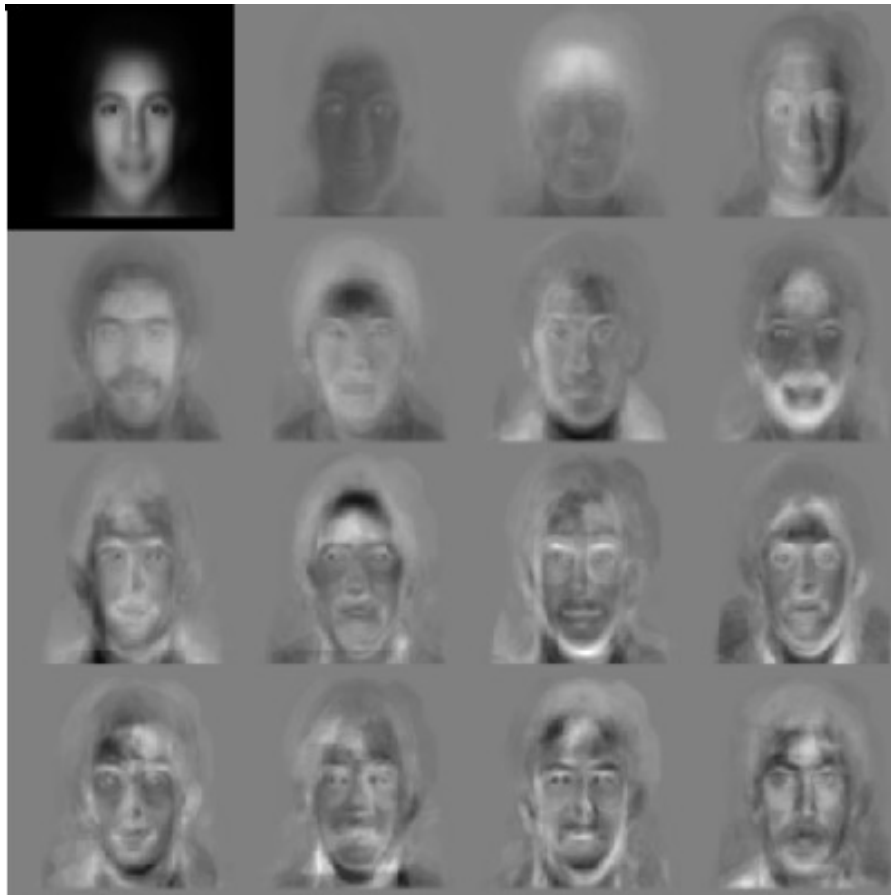
PCA: Example

- We can use the PCA to select the principal components in this space (i.e., the directions in which the elements mostly span (vary)) .
 - In practice, the eigenvectors (each one with dimension 4096) with the largest corresponding eigenvalues will be selected.
- Next, each original face can be represented as a weighted combination of the top-k eigenvalues.
 - In such case, each face will actually be represented by weights α : $\alpha_1, \dots, \alpha_k$
 - The PCA can be also regarded as a way to represent a face, with much less information than the originally used, while keeping the most important information.
- Further, the facial recognition process can be done in the new feature space of (much more) reduced dimension, i.e., typically $k \ll d$ (original space).

PCA: Example

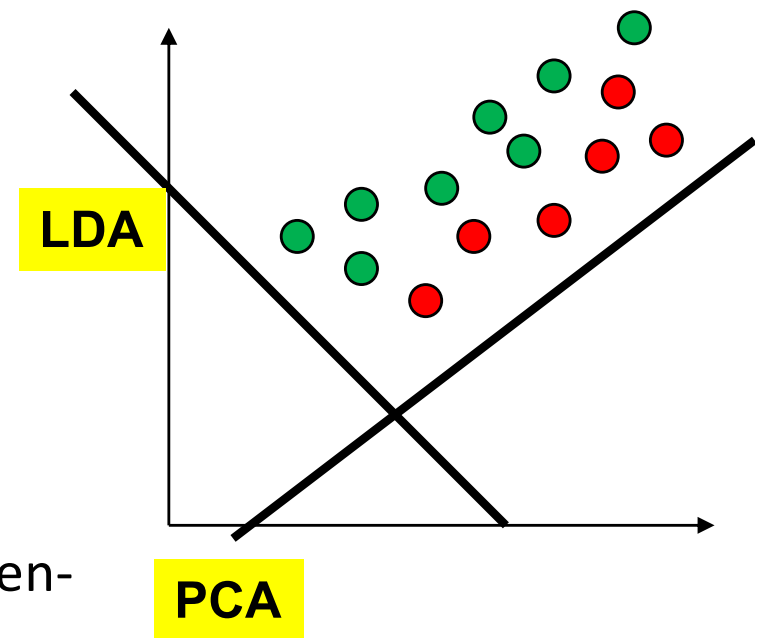
- Example of the 16 principal components (eigenvectors with the largest eigenvalues) from the above data set:

What do the
brightest regions
in each vector
represent?



PCA vs. LDA

- PCA (*Principal Component Analysis*)
 - Unsupervised (ignores class labels).
 - Maximizes variance in the data.
 - Produces components that best capture overall data spread.
- LDA (*Linear Discriminant Analysis*):
 - Supervised (uses class labels).
 - Maximizes class separability (ratio of between-class variance to within-class variance).
 - Produces components that best discriminate between classes.



LDA

1. Compute class statistics:

1. Mean vector for each class.
2. Overall mean of the dataset.

$$\mu_i = \frac{1}{n_i} \sum_{x \in C_i} x$$

$$\mu = \frac{1}{N} \sum_{i=1}^k \sum_{x \in C_i} x$$

2. Compute scatter matrices:

1. **Within-class scatter matrix** (S_W): how samples spread within each class.
2. **Between-class scatter matrix** (S_B): how class means spread relative to the overall mean.

$$S_W = \sum_{i=1}^k \sum_{x \in C_i} (x - \mu_i)(x - \mu_i)^T$$

$$S_B = \sum_{i=1}^k n_i (\mu_i - \mu)(\mu_i - \mu)^T$$

3. Solve the eigenvalue problem:

1. Find eigenvectors w_i of $S_W^{-1} S_B$
2. **Select top eigenvectors** (that give the most discriminant directions.)

$$W = [w_1, w_2, \dots, w_m]$$

4. Project data:

1. **Transform original data** into the new lower-dimensional space using the selected eigenvectors.

$$y = W^T x$$