Introdução à Aprendizagem Automática (IAA)

SUSANA BRÁS

SUSANA.BRAS@UA.PT

IAA – L2

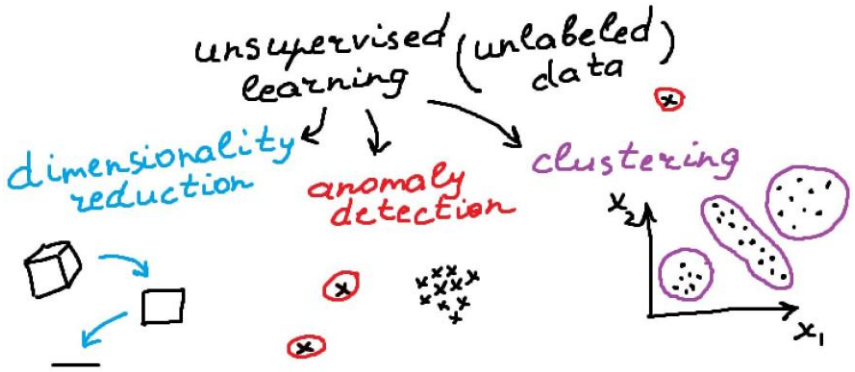
- Unsupervised Learning
  - K-means
    - Principle
    - K selection
  - Density based methods
  - Hierarchical methods
  - Dimensionality reduction
    - PCA

Data Matrix

matrix X (mxn)	feature x_1	feature x_2	.....	feature x_n
Example 1	$x_1^{(1)}$	$x_2^{(1)}$		$x_n^{(1)}$
Example 2	$x_1^{(2)}$	$x_2^{(2)}$		$x_n^{(2)}$
…				
Example i	$x_1^{(i)}$	$x_2^{(i)}$		$x_n^{(i)}$
…				
…				
Example m	$x_1^{(m)}$	$x_2^{(m)}$		$x_n^{(m)}$

Unsupervised Learning

Unsupervised Learning



Unsupervised learning = finding patterns without labels

Algorithms can learn hidden structure from unlabeled data, focusing on patterns, clusters, groups, and representations.

Examples: customer segmentation, anomaly detection, image compression

Key Goals:

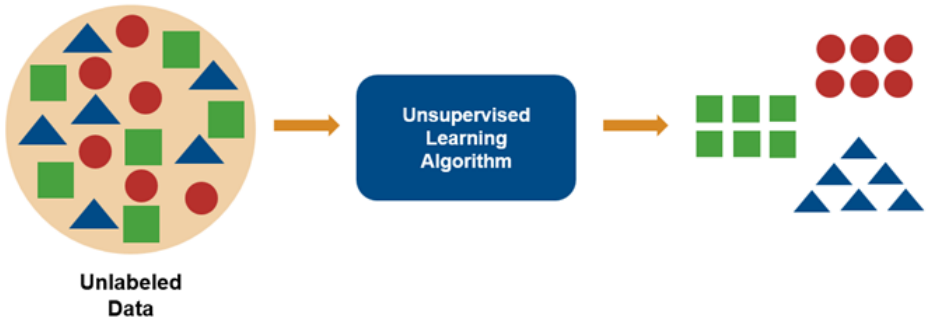
Grouping similar data points (Clustering)
Reducing dimensionality for visualization & efficiency
Discovering latent structures

Unsupervised learning helps us explore the unknown in data

Unsupervised Learning - Clustering

Intuition: find natural groups in the data

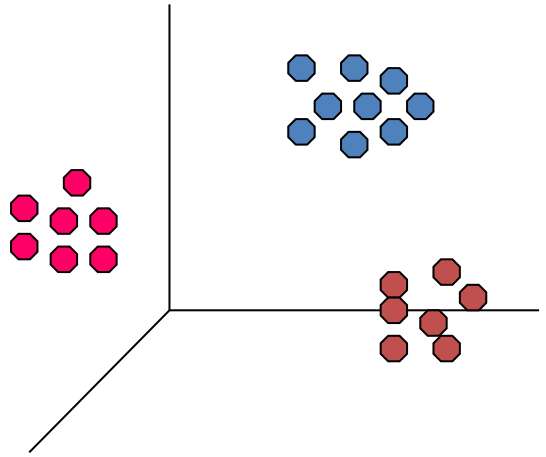
Applications: customer segmentation, document grouping



Evaluation Metrics:
Internal measures: silhouette score, cohesion/separation
External measures (when ground truth exists): Adjusted Rand Index, mutual information

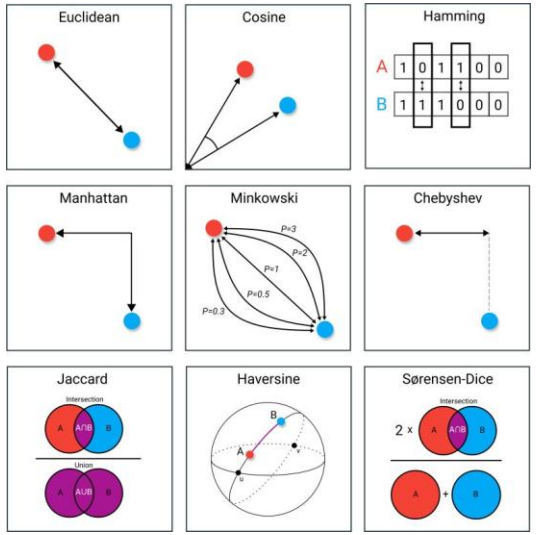
Challenges:
No ground truth – hard to validate
Sensitivity to parameters (e.g. number of clusters)
Interpretability of latent features

Unsupervised Learning - Clustering

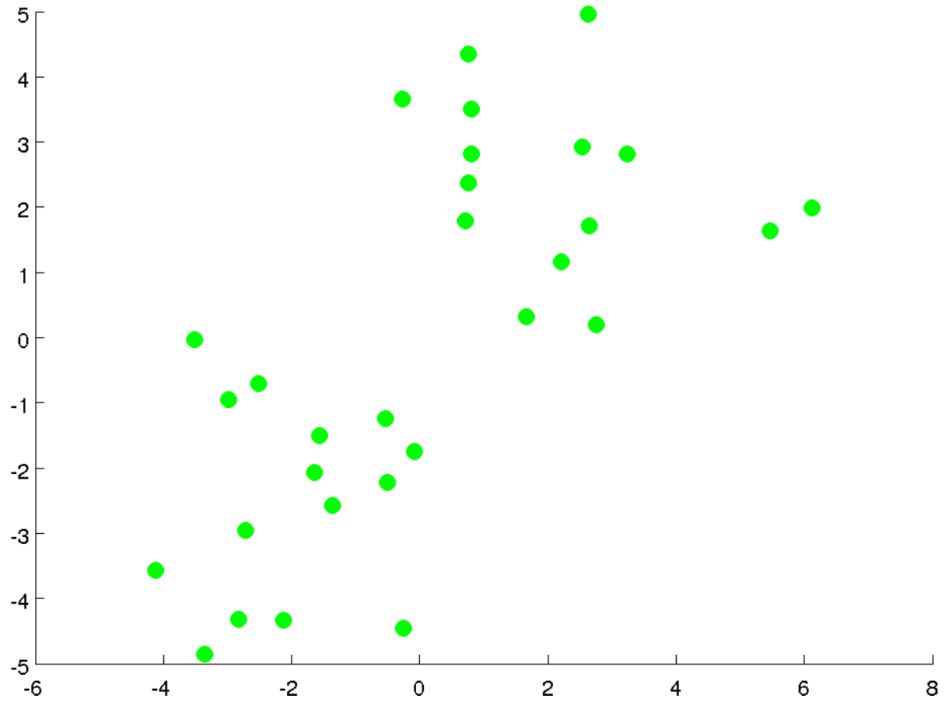


- Given a set of not labeled examples
- Find a relevant grouping of the examples into clusters such that:
  - Examples in the same cluster have high similarity
  - Examples from different clusters have high dissimilarity

Similarity measures – Euclidian distance; Chebyshev distance; Manhattan distance



Unsupervised Learning – K-means

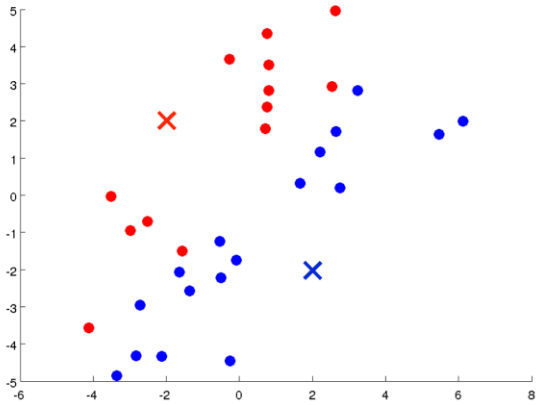


K=? (How many clusters?)
Distance=?

(Where did you define the margin of the clusters?)

Unsupervised Learning – K-means

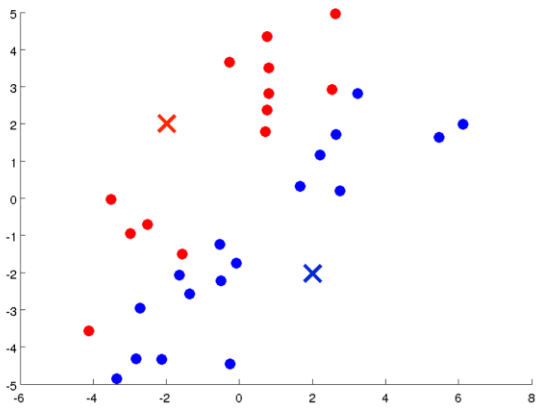
K=2 (Randomly initialized)
Distance = Euclidean



Iteration 1

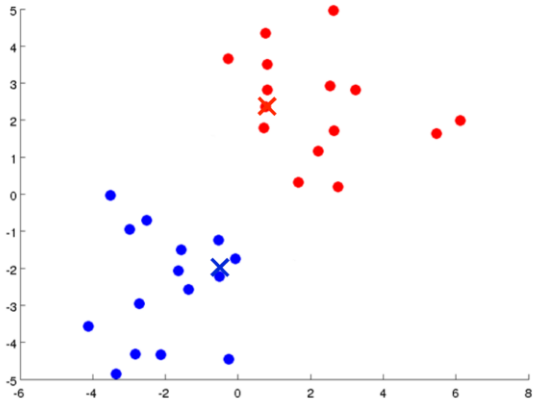
Unsupervised Learning – K-means

K=2 (Randomly initialized)
Distance = Euclidean



Iteration 1

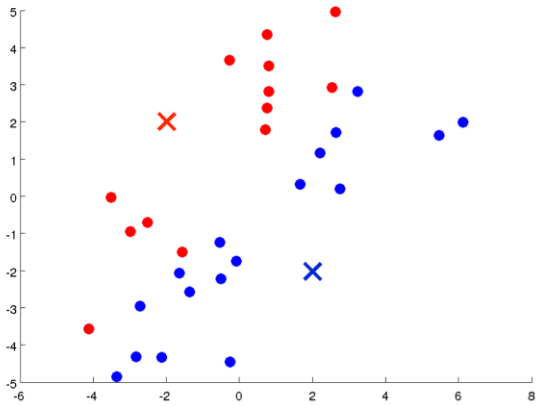
Compute new centroids = mean of the points assigned to that cluster.
Assign data points to the new closest centroid.



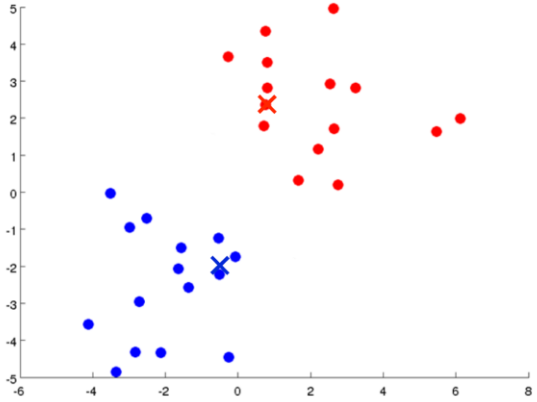
Iteration 2

Unsupervised Learning – K-means

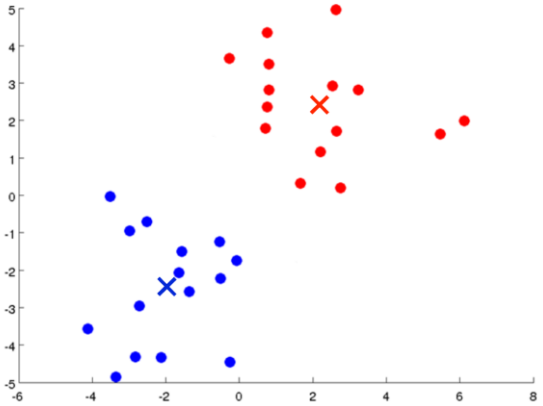
K=2 (Randomly initialized)
Distance = Euclidean



Iteration 1



Iteration 2



Iteration 3

Repeat until convergence…

Unsupervised Learning – K-means

Input:
- K (number of clusters)
- Training set (no labels)

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$
Repeat {
    for i = 1 to m
        $c^{(i)}$:= index (from 1 to K) of cluster centroid closest to $x^{(i)}$
    for k = 1 to K
        μ_k := average (mean) of points assigned to cluster k

}

Cluster assignment => step

Move centroid => step

K-means optimization

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

$$\min_{\substack{c^{(1)},\ldots,c^{(m)},\\ \mu_1,\ldots,\mu_K}} J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$$

Stop K-means learning (different criteria):

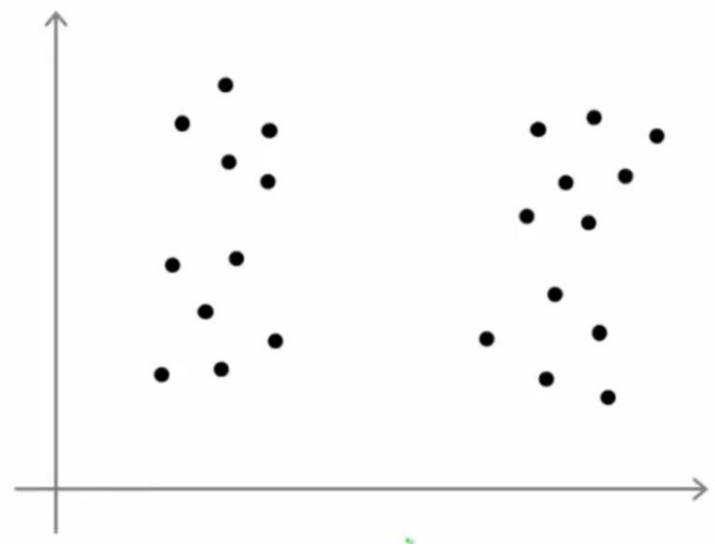
- Achieved Max number of iterations
- J < some threshold
- No improvement of *J* between subsequent iterations

For i = 1 to 100 {

Randomly initialize K-means.
Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.
Compute cost function (distortion)
$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$
}

Pick clustering that gave lowest cost $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

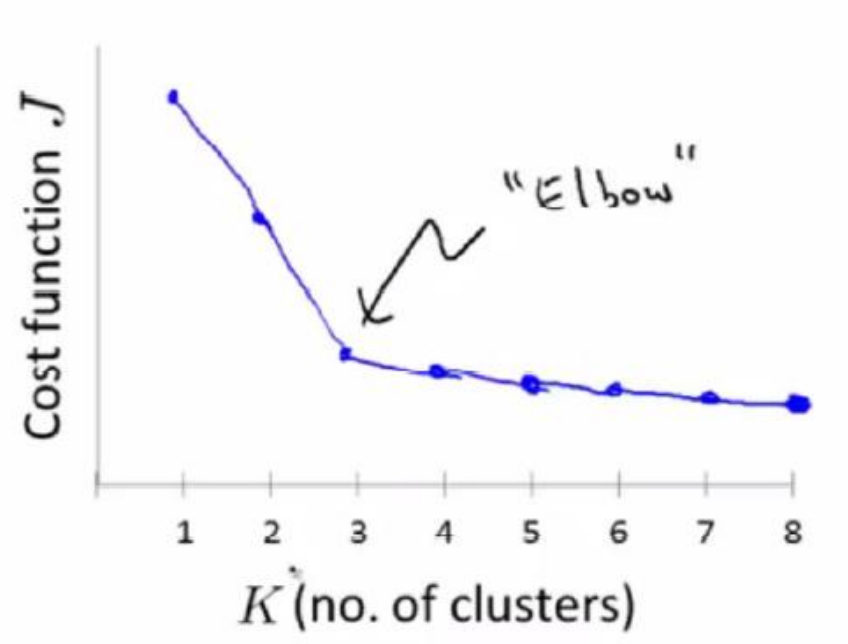
Unsupervised Learning – K-means

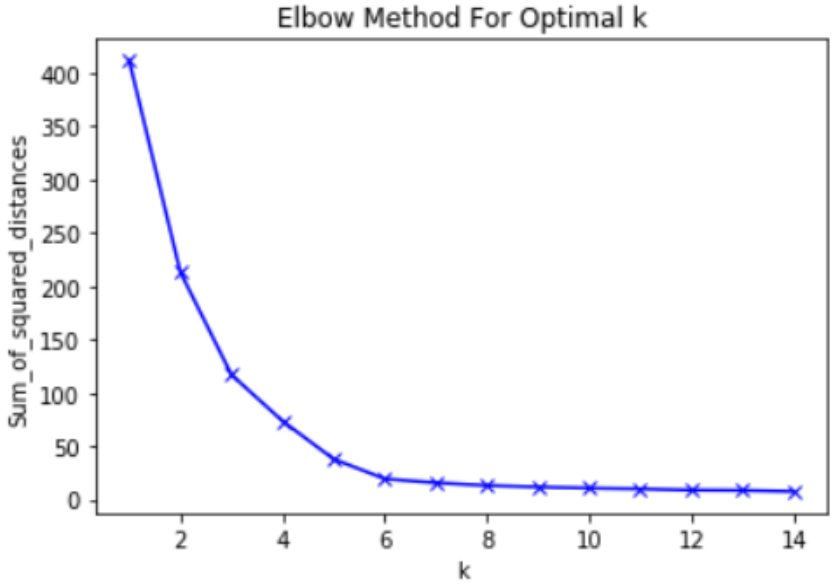


How to choose the K number of clusters?

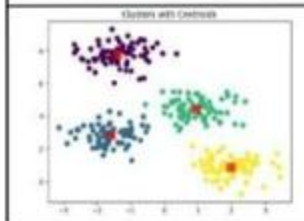
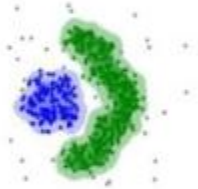
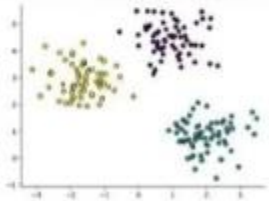
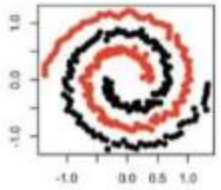
- Choose K by data visualisation (if possible)
- Ask domain experts (highly recommendable) , e.g. anomaly detection (experts should know how many types of anomalies are expected)
- Choose K automatically (e.g. Elbow method)

Unsupervised Learning – K-means



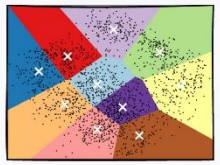
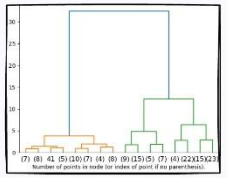
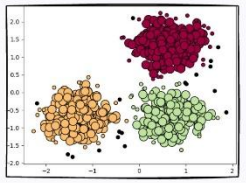
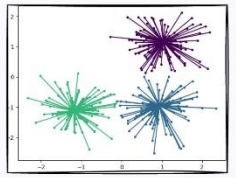
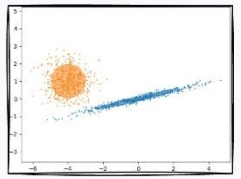
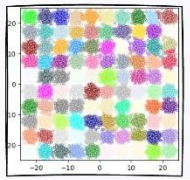


Other Clustering Techniques

Representation	Algorithm Name	Hyperparameter
	K-Means Clustering	Partitions data into K clusters by minimizing variance within each cluster.
	Hierarchical Clustering	Builds a hierarchy of clusters by iteratively merging or splitting existing groups.
	DBSCAN	Forms clusters based on density; groups densely packed points and marks outliers.
	Mean Shift	Finds clusters by locating and adapting to the centroids of data points.
	Spectral Clustering	Uses eigenvalues of similarity matrix to reduce dimensions before clustering.

6 Types of Clustering Algorithms in Machine Learning

blog.DailyDoseofDS.com

Clustering Algorithm Type		Clustering Methodology	Algorithm(s)
	Centroid-based	Cluster points based on proximity to centroid	KMeans KMeans++ KMedoids
	Connectivity-based	Cluster points based on proximity between clusters	Hierarchical Clustering (Agglomerative and Divisive)
	Density-based	Cluster points based on their density instead of proximity	DBSCAN OPTICS HDBSCAN
	Graph-based	Cluster points based on graph distance	Affinitiy Propagation Spectral Clustering
	Distribution-based	Cluster points based on their likelihood of belonging to the same distribution.	Gaussian Mixture Models (GMMs)
	Compression-based	Transform data to a lower dimensional space and then perform clustering	BIRCH

K-means - Summary

- The most popular clustering method.
- Need to know K.
- May converge to a Local Minimum .
- High number of computations.

Unsupervised Learning - Summary

Advantages of Unsupervised Learning

- **No Manual Labeling Required:** It works directly on raw data, saving time and cost associated with annotating large datasets.
- **Discovery of Hidden Patterns:** It identifies latent structures, trends, or segments (e.g., market segmentation, customer behavior) that human analysis might miss.
- **Anomaly Detection:** It is highly effective at finding outliers, fraud, or faulty equipment by learning "normal" patterns.
- **Handling Unstructured Data:** Capable of handling massive, unstructured, or complex datasets.
- **Flexibility:** It can be applied to a wide range of data types (e.g., text, images, sensor data).

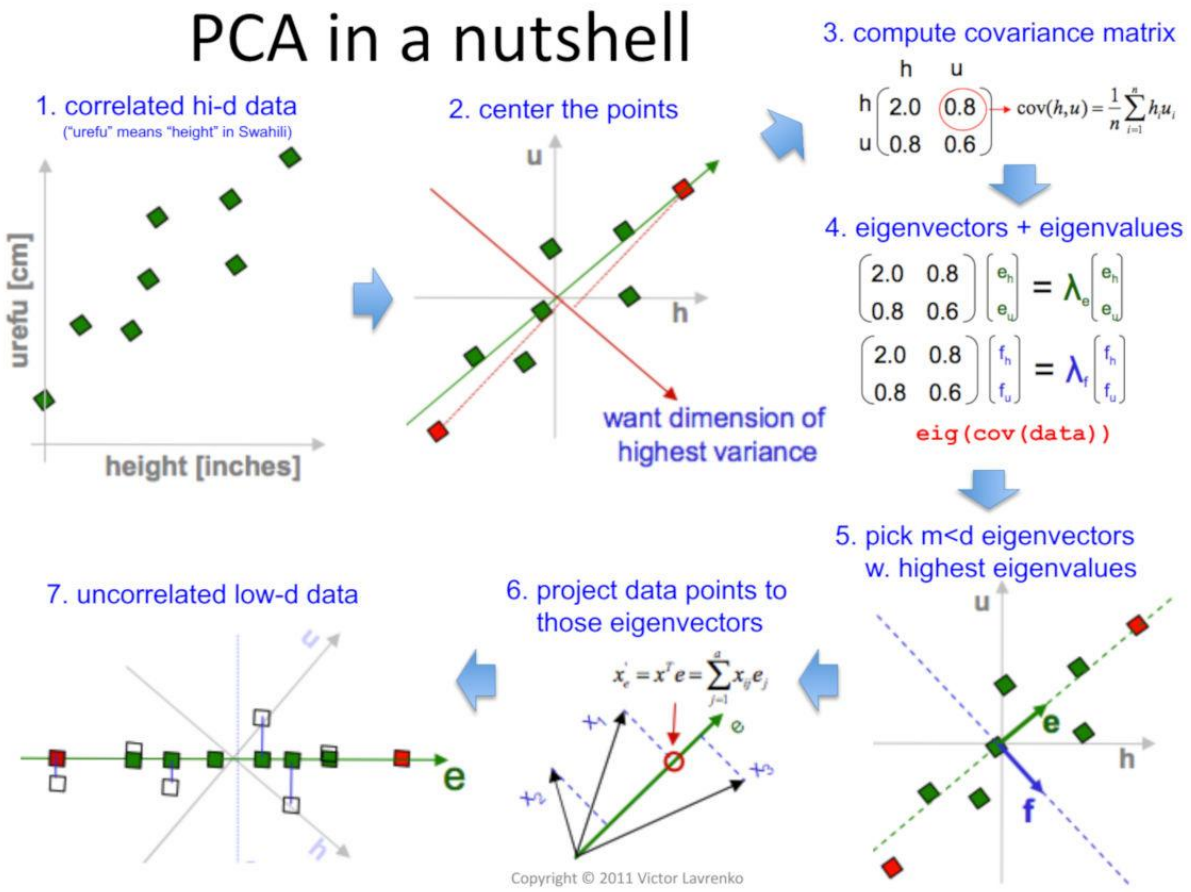
Limitations of Unsupervised Learning

- **Lower Accuracy/Interpretability:** Because the model does not have a "correct" answer to learn from, results can be less accurate or difficult to interpret.
- **Sensitivity to Noise/Scale:** Algorithms are often sensitive to outliers and require significant data preprocessing (e.g., scaling features) to avoid skewed results.
- **Evaluation Challenges:** It is difficult to measure the accuracy or effectiveness of the output (e.g., finding the "right" number of clusters).
- **Computational Complexity:** Some algorithms require significant computational resources and longer training times when dealing with high-dimensional data.
- **Ambiguous Results:** The clusters or patterns discovered might not correspond to meaningful, real-world categories.

Dimensionality Reduction

Dimensionality Reduction

- **Principal component analysis (PCA):**



Dimensionality Reduction

Dimensionality reduction with axis rotation

- Project the data in a new data space reducing the correlation between predictors.

- **Principal component analysis (PCA):** The goal is to rotate the data into an axis-system where the greatest amount of variance is captured in a small number of dimensions.
  - Usually the method should be applied after **mean centering** of the data (subtracting the mean to each data point).

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$

Preprocessing (feature scaling/mean normalization):

$$\mu_j = \frac{1}{m} \sum_{i=1}^{m} x_j^{(i)}$$

Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

Thus, all features have zero mean !

Dimensionality Reduction

Dimensionality reduction with axis rotation

- Project the data in a new data space reducing the correlation between predictors.

- **Principal component analysis (PCA):** The goal is to rotate the data into an axis-system where the greatest amount of variance is captured in a small number of dimensions.
  - Usually the method should be applied after **mean centering** of the data (subtracting the mean to each data point).
  - PCA is **sensitive to the scale** of the original data.

If the features have significantly different range of values, normalize them., e.g. in the interval [0,1] or [-1,1] or mean=0 & std=1

MinMaxScaler - Transforms features by scaling each feature to a given range.

StandardScaler - Standardize features by removing the mean and scaling to unit variance.

Dimensionality Reduction

Dimensionality reduction with axis rotation

- Project the data in a new data space reducing the correlation between predictors.

- **Principal component analysis (PCA):** The goal is to rotate the data into an axis-system where the greatest amount of variance is captured in a small number of dimensions.
  - Usually the method should be applied after **mean centering** of the data (subtracting the mean to each data point).
  - PCA is **sensitive to the scale** of the original data.
  - It is an **orthogonal transformation** of the data to convert a possibly correlated observations in a set of variables linearly non correlated called principal components.
  - The number of **principal components is always smaller that the number of original** variables.
  - The **first PC** is defined as the linear combination of the predictors that **captures the most variability** of all possible linear combinations. Then, subsequent PCs are derived such that these linear combinations capture the most remaining variability while also being uncorrelated with all previous PCs.
  - PCA can be seen a trade-off between faster computation and less memory consumption versus information loss.

Dimensionality Reduction

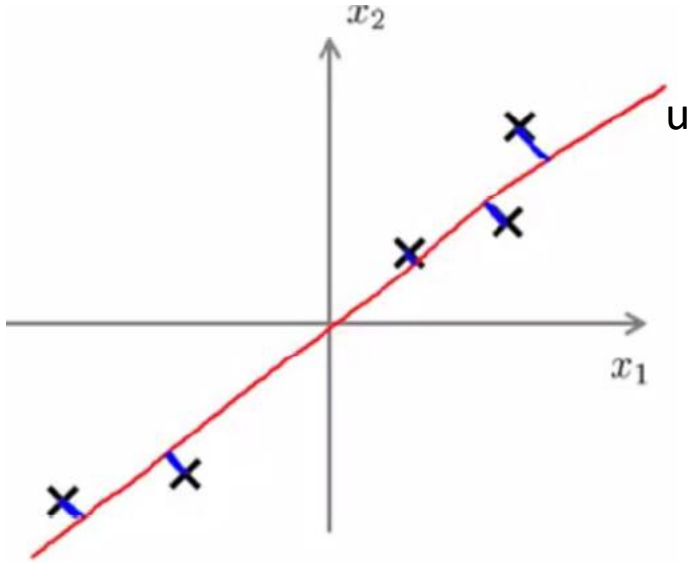
Principal component analysis (PCA):

- Drawbacks:
  - **PCA works only if the observed variables are linearly correlated**. If there's no correlation, PCA will fail to capture adequate variance with fewer components.
  - **PCA is lossy**. Information is lost when we discard insignificant components.
  - **Scaling of variables can yield different results.** Hence, scaling that you use should be documented. Scaling should not be adjusted to match prior knowledge of data.
  - Since each principal components is a **linear combination** of the original features, **visualizations** are **not easy to interpret** or relate to original features.

PCA – in practice

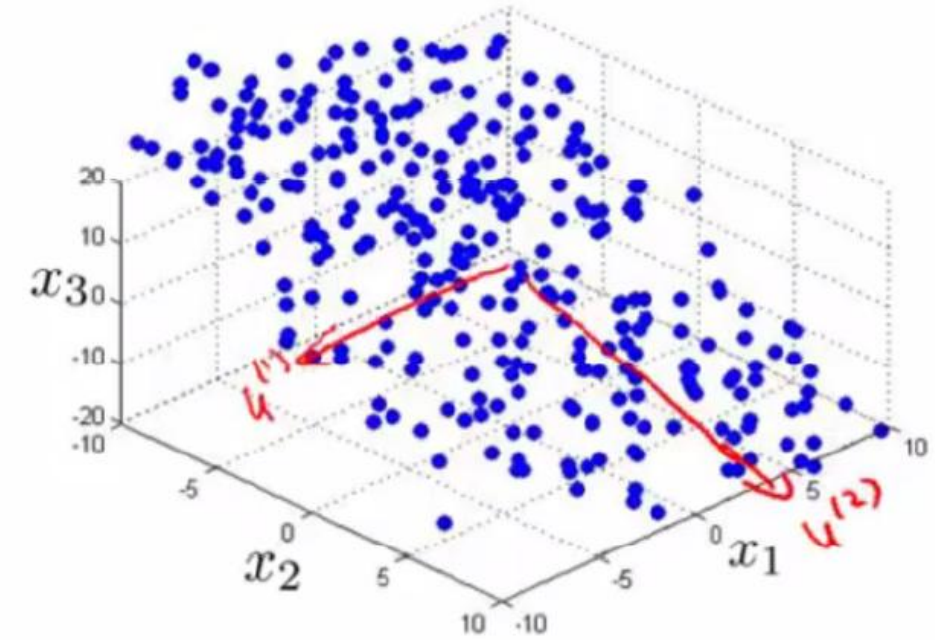
Reduce from 2D to 1D:

find the best direction (vector u) onto which to project data such that to minimize the projection error



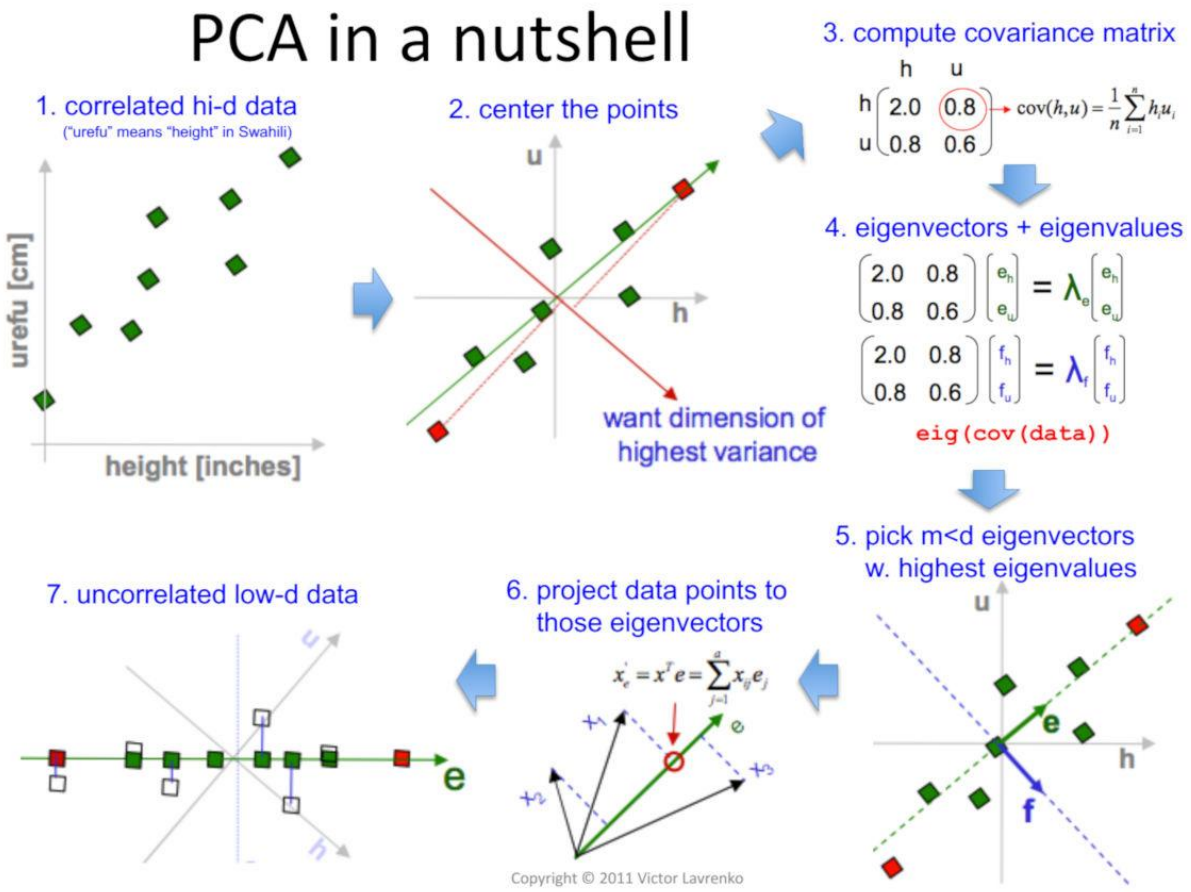
Reduce from 3D to 2D:

find the orientation of the best plane (vectors u1, u2) onto which to project data such that to minimize the projection error



Dimensionality Reduction

- **Principal component analysis (PCA):**



PCA – Singular Value Decomposition

- Compute Covariance matrix of the mean normalized data matrix *X* (dimension *mxn* - m examples, n features):

$Cov=X^T*X\ /m$

- Compute Singular Value Decomposition(SVD) of Covariance matrix:

Cov=U*S*V

U (*nxn*) - matrix of eigenvectors:

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \cdots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

S (*nxn*)– diagonal matrix of singular values in decreasing order:

$$S_{nxn} = \begin{bmatrix} S_{11} & 0 & \cdots & 0 \\ 0 & S_{22} & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \cdots & 0 & S_{nn} \end{bmatrix}$$

SVD is equivalent to eigen-values/eigen-vector decomposition.

PCA - projection

The projection vectors are the first *k* columns of *U* (k<n): -*Task 1*

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \dots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n} \quad \Longrightarrow \quad Ureduce_{nxk} = U(:, 1:k)$$

Step 3: Compute the new (projected) data matrix *Z* (*m* examples, *k* features): - *Task 2*

$$Z_{mxk} = X_{mxn} * Ureduce_{nxk}$$

Step 4: Reconstruct data matrix *X* from the projected *Z* matrix :

$$X_{approx(mxn)} = Z_{mxk} * Ureduce^{T}_{kxn}$$

PCA – Choosing K

Average squared approximation error: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2$

Total data variation: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2$

Typically, choose k to be smallest value so that

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2} \leq 0.01 \qquad (1\%)$$

"99% of variance is retained"

(typically the desired retained variance is between 90-99%)

PCA - Summary

PCA is used to reduce the dimensionality of the original feature space and, at the same time, to maximise the orthogonality between the features in the transformed feature space.

The new set of features obtained through the PCA process are the principal components, which are computed by applying a linear transform to the original features.

Such principal components correspond to largest eigenvalues of the co-variance matrix of features. Then a reduced set of principal components can be used to reconstruct most of the original data with maximum variance, thus keeping most of its information.

The orthogonality between components ensures decorrelation in the transformed feature space.

Best Practices & Takeaways

- Always explore your data before analysis
- Preprocessing is often the most time-consuming step
- Document decisions (e.g., how you handled missing values)
- No “one-size-fits-all” solution