Machine Learning

Unsupervised Learning

Topics

- **Types of unsupervised learning**
- **Challenges in unsupervised learning**
- **Cluster Analysis**
- **K-means**
- **Hierarchical Aggregation**
- **Normalization**
- **Dimensionality Reduction: Principal Component Analysis**

Introduction

- **In Machine Learning** there may be many cases in which we do **not have labeled data** and need to find **the hidden patterns** from the given dataset. So, to solve such types of cases in machine learning, we need **unsupervised learning techniques**.
- In these learning techniques do not need to **supervise the model.** Instead, it allows the model to **work on its own to discover patterns** and **information that was previously undetected.**

Introduction

- It allows users to perform more complex processing tasks compared to supervised learning, because of its ability to group and interpret information based on similarities, patterns, and differences.

- Its ability to **discover similarities and differences** in information make it the ideal solution for **exploratory data analysis, cross-selling strategies, customer segmentation and image recognition.**

Introduction

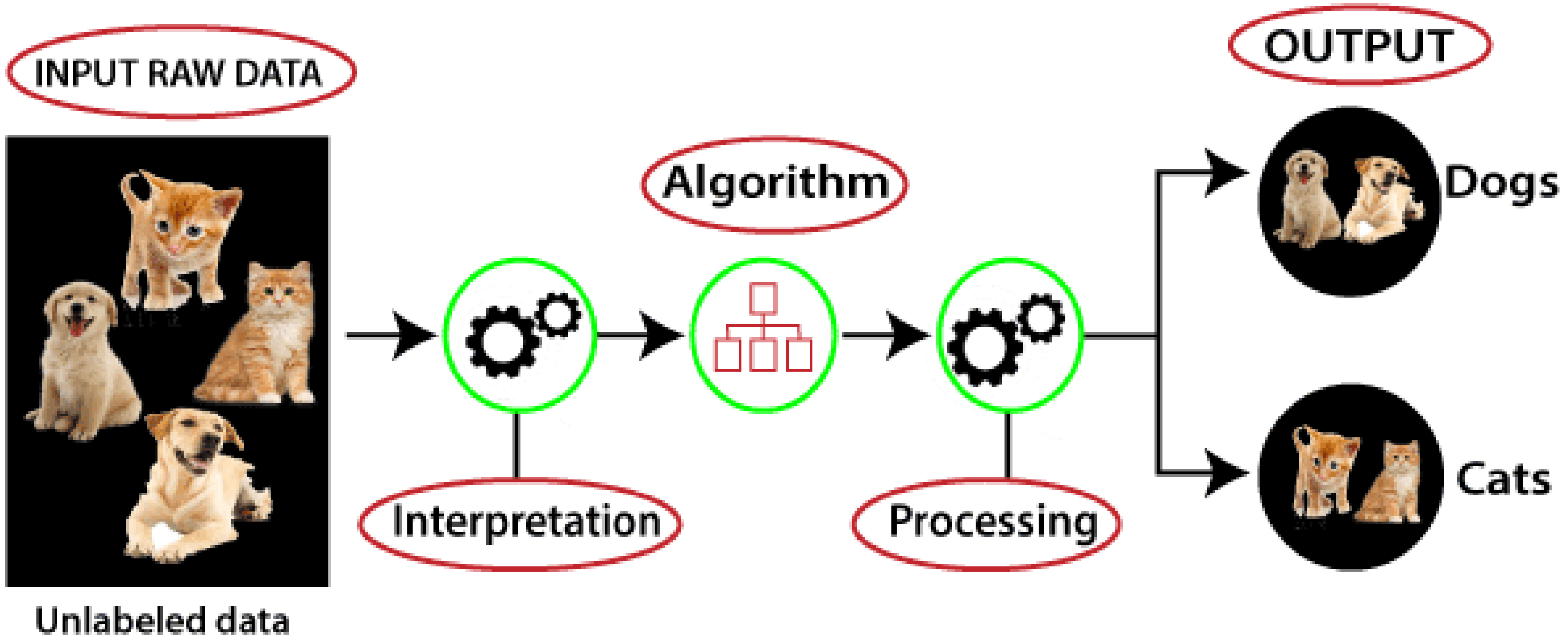
- Unsupervised learning cannot be directly applied to a **regression or classification problem** because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying **structure of dataset, group that data according to similarities and represent that dataset in a compressed format.**
- **Unsupervised learning is a powerful approach that has the potential to unlock valuable insights from unlabeled data.**

Introduction

Example-1 : Suppose we are given an input dataset containing images of **different types of cats and dogs**. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own.

Introduction

Example-1 : Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.



Introduction

Example-2 :

For example, if we were releasing a new product, we can use unsupervised learning methods to identify who the target market for the new product will be: this is because there is no historical information about who the target customer is and their demographics.

Why Unsupervised Learning?

Prime reasons for using Unsupervised Learning:

- Unsupervised machine learning finds all kind of unknown patterns in data.
- Unsupervised methods help you to find features which can be useful for categorization.
- Unsupervised learning is much similar as a human learns to think by their own experiences, which makes it closer to the real AI.
- It is easier to get unlabeled data from a computer than labeled data, which needs manual intervention.

Types of Unsupervised Learning

Since no labels are present, unsupervised learning methods are typically applied to build a concise representation of the data so we can derive **imaginative content** from it.

- **Unsupervised learning can be of three main types:**
  1. **Clustering**
  2. **Association rules**
  3. **Dimensionality reduction**

Types of Unsupervised Learning - Clustering

- Clustering methods involve grouping untagged data based on their similarities and differences. When two instances appear in different groups, we can infer they **have dissimilar properties**.
- Cluster analysis finds the commonalities between the data objects and categorizes them as per the presence and absence of those commonalities.

Types of Unsupervised Learning - Clustering

- You can even break it down further into different types of clustering
    - **Exclusive clustering:** Data is grouped such that a single data point exclusively belongs to one cluster.
    - **Overlapping clustering:** A soft cluster in which a single data point may belong to multiple clusters with varying degrees of membership.
    - **Hierarchical clustering:** A type of clustering in which groups are created such that similar instances are within the same group and different objects are in other groups.
    - **Probalistic clustering:** Clusters are created using probability distribution

Types of Unsupervised Learning - Association

- Association a **rule-based approach** to discovering interesting relationships between features in a given dataset.
- It works by using a **measure of interest** to identify strong rules found within a dataset.
- We typically see association rule mining used for market **basket analysis:** this is a **data mining technique** retailers use to gain a better **understanding of customer purchasing patterns** based on the **relationships between various products**.

Types of Unsupervised Learning - Association

- **Association rule** makes marketing strategy more effective. Such as people who **buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item**.
- The most widely used algorithm for association rule learning is the **Apriori algorithm.**
- Other algorithms are the **Eclat and FP-growth algorithms**

Types of Unsupervised Learning - Dimensionality Reduction

- These algorithms seek to transform data from high-dimensional spaces to low-dimensional spaces without compromising meaningful properties in the original data.
- These techniques are typically deployed during **exploratory data analysis (EDA)** or **data processing** to prepare the data for modeling.
- Popular algorithms used for dimensionality reduction include **principal component analysis (PCA)** and **Singular Value Decomposition (SVD).**
-

Types of Unsupervised Learning - Dimensionality Reduction

- It’s helpful to reduce the dimensionality of a dataset during **EDA** to help **visualize data**: this is because visualizing data in more than three dimensions is difficult.
- From **a data processing perspective**, reducing the dimensionality of the data simplifies the modeling problem.

Unsupervised Learning Applications

- **Natural language processing (NLP).**
- **Image and video analysis.**
- **Anomaly detection.**
- **Customer segmentation.**
- **Recommendation Engines.**

Challenges of Unsupervised Learning

- **Lack of Ground Truth**
- **Curse of Dimensionality**
- **Clustering Ambiguity**
- **Scalability and Efficiency**
- **Interpretability**

Challenges of Unsupervised Learning - Lack of Ground Truth

- Unsupervised learning operates in an environment where the data is unlabelled, making it challenging to evaluate the performance of algorithms objectively
- Without ground truth labels, it is difficult to quantify how well an unsupervised learning model has learned the underlying patterns or structures in the data.
- Researchers often resort to **heuristic methods**, such as **clustering quality metrics or visual inspection**, to assess the quality of unsupervised models.

Challenges of Unsupervised Learning - Curse of Dimensionality

- Unsupervised learning often deals with high-dimensional data.
- As the **number of features or dimensions in the data increases**, the amount of data required to effectively learn the underlying structure **grows exponentially.**
- This means that unsupervised learning algorithms may struggle when confronted with **high-dimensional data** due to **data sparsity** and **increased computational complexity**.
- To mitigate the curse of dimensionality, techniques like **dimensionality reduction**, **feature selection**, and **feature engineering** are commonly employed.

Challenges of Unsupervised Learning - Clustering Ambiguity

- One of the challenges in clustering is the inherent ambiguity in defining **what constitutes a “good” cluster.**
- Different clustering algorithms can produce varying results for the same dataset, and **there may not be a single correct clustering solution.**
- To address clustering ambiguity, researchers have developed various **clustering evaluation metrics**, such as **silhouette scores** and **Davies-Bouldin index**, to quantify the quality of clustering results.

Challenges of Unsupervised Learning - Scalability and Efficiency

- Many unsupervised learning algorithms are computationally intensive and may not scale well to **large datasets or high-dimensional data.**
- Training such models on massive datasets can be **time-consuming** and require substantial **computational resources.**
- The **scalability issues** makes them impractical for real-time or production use.

Challenges of Unsupervised Learning - Scalability and Efficiency

- Efforts are ongoing to **develop more efficient unsupervised** learning algorithms and **parallel computing techniques** that can handle large-scale data efficiently.

Challenges of Unsupervised Learning - Interpretability

- Interpreting the results of unsupervised learning models can be challenging, especially when dealing with **complex, high-dimensional data.**
- While it detects patterns, understanding what these patterns represent in real-world terms is not always straightforward.

Challenges of Unsupervised Learning - Interpretability

- Improving the interpretability of unsupervised learning models is an active area of research.
- Techniques such as **visualization tools, dimensionality reduction, and feature importance analysis** are being explored to make the results more accessible and interpretable to end-users.

Challenges of Unsupervised Learning - Interpretability

- Improving the interpretability of unsupervised learning models is an active area of research.
- Techniques such as **visualization tools, dimensionality reduction, and feature importance analysis** are being explored to make the results more accessible and interpretable to end-users.

Cluster Analysis

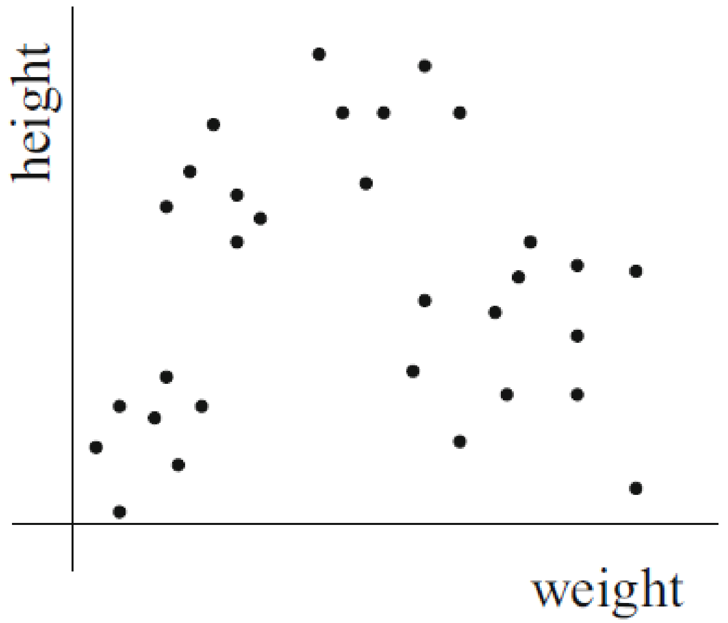
Cluster Analysis

- The fundamental task in unsupervised learning is **cluster analysis.**
- Here, the input is a set of examples, each described by **a vector of attribute values**—but no class labels.
- The output is a set of **two or more clusters of examples**

Cluster Analysis

- Figure 14.1 shows a simple domain with a few examples described by two attributes: **weight and height.** An observer can easily see that the examples form three or four groups,

Fig. 14.1 A two-dimensional domain with clusters of examples



- Visual identification of such groups in a two-dimensional space is easy, but in **four or more dimensions**, humans can neither visualize the data nor see the clusters.

- These can only be detected by **cluster-analysis algorithms.**

Cluster Analysis - Representing Clusters by Centroids

- If all attributes are numeric, the centroid is identified with the averages of the individual attributes.
- For instance, suppose a two-dimensional cluster consists of the following examples: **(2, 5), (1, 4), (3, 6).**
- In this case, the centroid is described by vector **(2,5)** because the first attribute's average is:

$$\frac{2+1+3}{3} = 2 \qquad \frac{5+4+6}{3} = 5.$$

Cluster Analysis - Representing Clusters by Centroids

- The averages can be calculated even when the attributes are discrete if we know how to turn them into numeric ones.
- **Clusters should not overlap each other**: each example must belong to one and only one cluster.
- Within the same cluster, the examples should be relatively close to each other, certainly much closer than to the examples from the other clusters.
- An important question will ask **how many clusters the data contain**. **Machine learning** software is expected to determine the number automatically.

Cluster Analysis - Representing Clusters by Centroids

Problems with Measuring Distances

- Cluster analysis usually need a mechanism to evaluate the distance between an example and a cluster.
- If the clusters are described by their **centroids**, the Euclidean distance between the two centroids will be the cluster distance.
- Euclidean distance may be inconvenient in the case of **discrete attributes**, but we already know how to deal with them.

Cluster Analysis - Representing Clusters by Centroids

Problems with Measuring Distances

- Also the problem of **scaling plays** its role in this case. Most of the time, engineers get around the difficulties by normalizing all attribute values into the unit interval, $x_{i=}[0,1]$
- If $x = (2,1.5, summer)$ and $y = (1, 0.5, winter)$; then Equation gives the following distance between the two:

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt{(2-1)^2 + (1.5-0.5)^2 + 1} = \sqrt{3}$$

For example , the difference between two sizes say **size1 = 1 and size2= 12, which means that d(size1, size2) = 121,** which can totally dominate the **difference between two seasons.**

Cluster Analysis - Representing Clusters by Centroids

A More General Formula for Distances

- If the examples are described by a mixture of numeric and discrete attributes, we can rely on the sum of the squared distances along corresponding attributes. More specifically, the following expression is recommended

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt{\Sigma_{i=1}^{n} d(x_i, y_i)}$$

For instance, we can use $d(x_i, y_i) = (x_i \; - y_i \;)^2$ for continuous attributes, whereas for discrete attributes, we put $d(x_i, y_i) = 0$ if $x_i = y_i$ and $d(x_i, y_i) = 1$ if $x_i \neq y_i$.

Cluster Analysis - Representing Clusters by Centroids

Which Cluster Should an Example Belong To?

- For instance, suppose that we use the Euclidean distance, and that there are three clusters.
- If the centroids are $c_1 = (3,4),\ c_2 = (4,6)\ and\ \ c_3 = (5,7)$
- If the example is $x = (4,4)$, then the Euclidean distances are

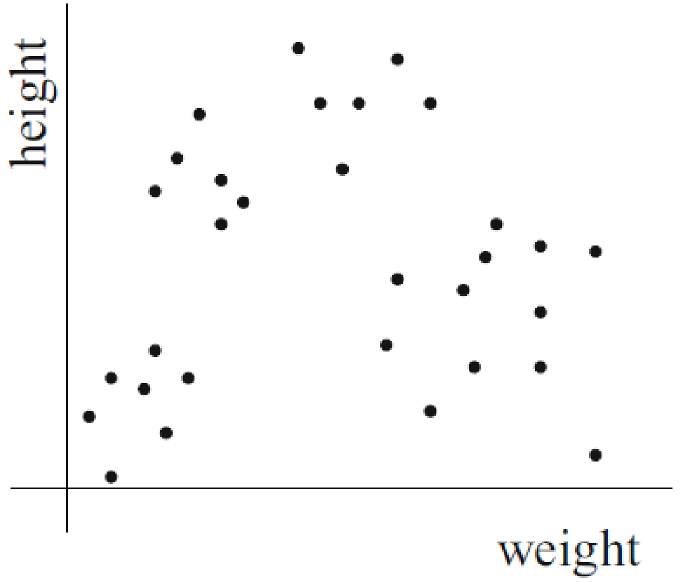
$$\boldsymbol{d(x, c_1) = 1,\ \ d(x, c_2) = 2}, \textbf{ and } \boldsymbol{d(x,\ c_3) = \sqrt{10}}$$

- Since $\boldsymbol{d(x, c_1) = 1}$ is the smallest of the three values, we conclude that x should belong to $\boldsymbol{c_1}$.

Cluster Analysis - Benefits

1. Benefit 1: Estimating Missing Values

- The knowledge of the clusters can help us estimate missing attribute values.
-
  - For example, **if height is missing** and if weight is low, the example is bound to belong to the bottom-left cluster.
  - In this case, also height is likely to be low because it is low in all examples found in this cluster.



Cluster Analysis - Benefits

1. **Benefit 1: Estimating Missing Values**

- Unknown/missing can value be estimated as the average or the most frequent value encountered in the training set
- An estimate of the missing value as the average or the most frequent value of the given cluster is sounder because it uses more information about the nature of the domain.

Cluster Analysis - Benefits

2. **Benefit 2: Reducing the Size of RBF Networks and Bayesian Classifiers**

- Cluster analysis can assist such techniques as **Bayesian learners and radial-basis-function networks**.
- The reader will recall that these paradigms operate with centers and the centers are identified with the attribute vectors of the individual examples.
- In domains with millions of examples, however, this would lead to **impractically big classifiers.**
- **The engineer then prefers to divide the training set into N clusters, and to identify the gaussian centers with the centroids of the clusters.**

Cluster Analysis - Benefits

2. **Benefit 2: Reducing the Size of RBF Networks and Bayesian Classifiers**
3. **Benefit 3: A Simple Classifier**
   - Finally, the knowledge of data clusters may be useful in supervised learning.
   - It is quite common that all (or almost all) examples in a cluster belong to the same class.
   - In that case, the developer of a supervised learning software may decide first to identify the clusters, and then label each cluster with its dominant class.

Cluster Analysis

A Simple Algorithm: k-Means

k-Means - Algorithm

- The **“k”** in the name denotes the **requested number of clusters**—a parameter whose value is supplied by the user.

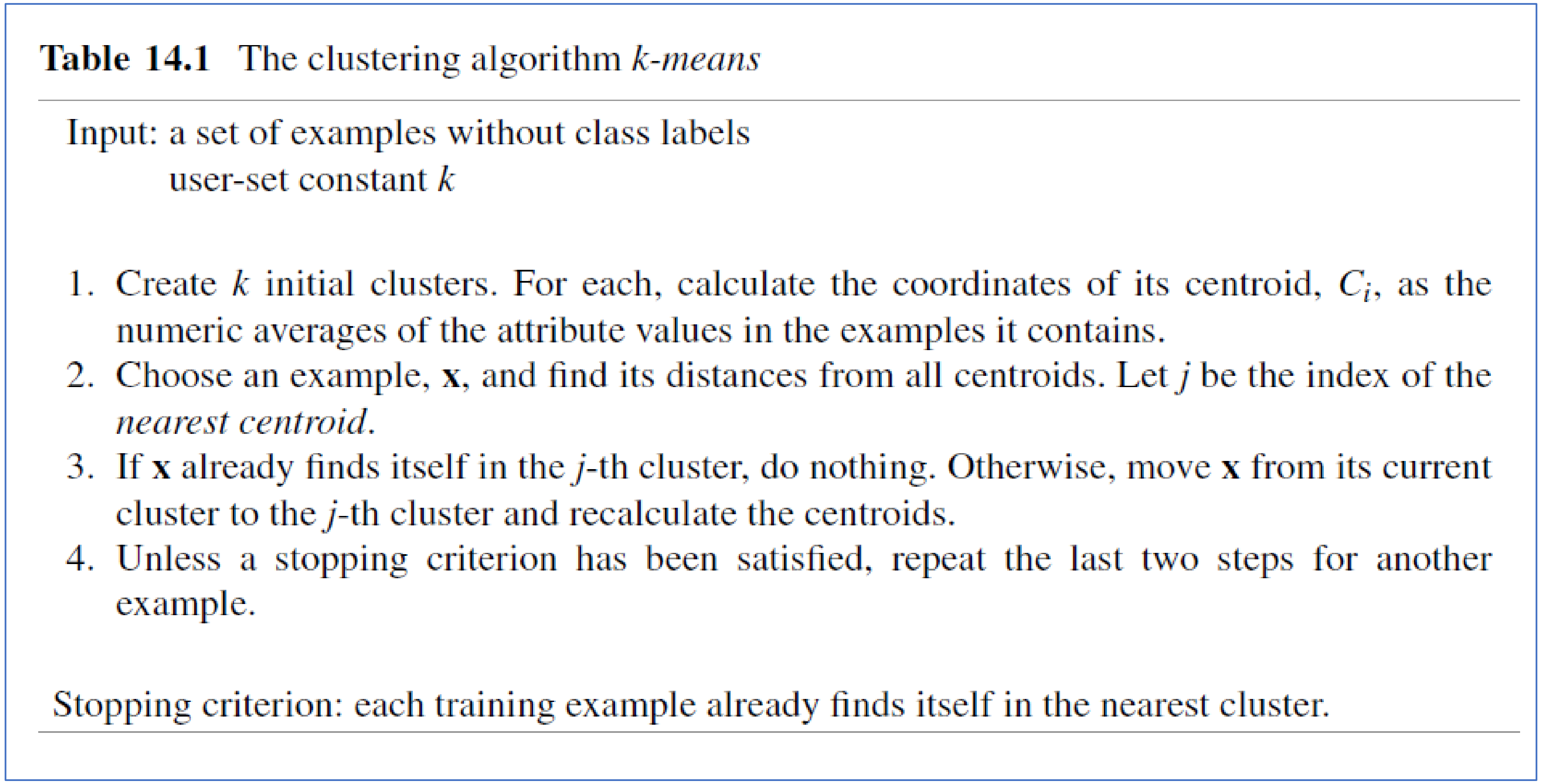


Table 14.1 The clustering algorithm *k-means*

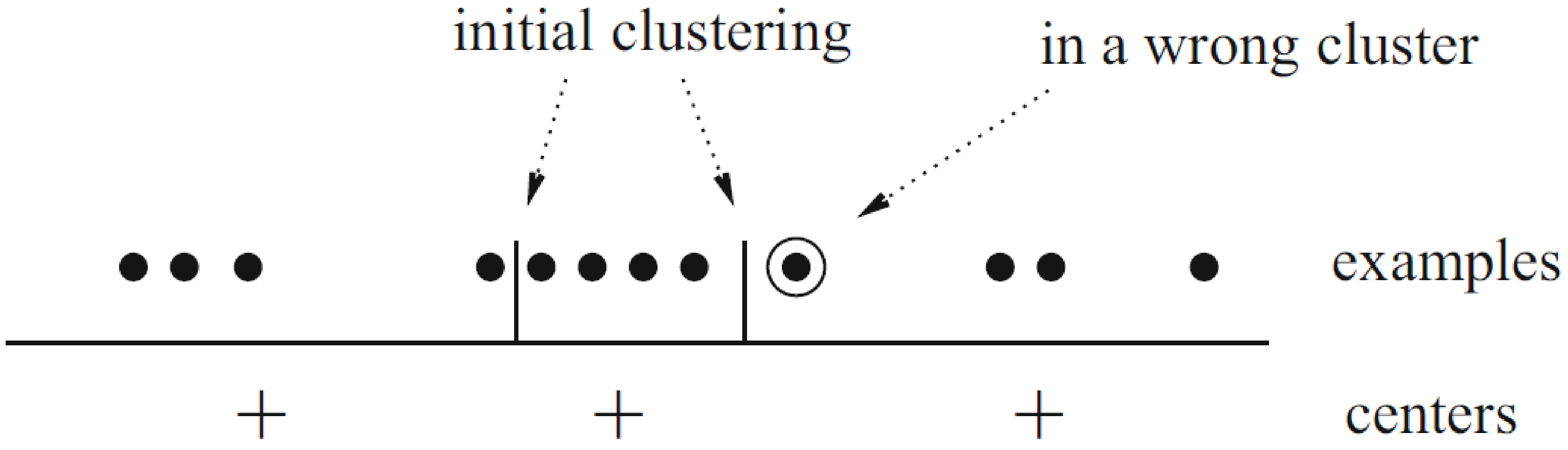
Input: a set of examples without class labels
user-set constant k

1. Create k initial clusters. For each, calculate the coordinates of its centroid, C_i, as the numeric averages of the attribute values in the examples it contains.
2. Choose an example, $\mathbf{x}$, and find its distances from all centroids. Let j be the index of the *nearest centroid*.
3. If $\mathbf{x}$ already finds itself in the j-th cluster, do nothing. Otherwise, move $\mathbf{x}$ from its current cluster to the j-th cluster and recalculate the centroids.
4. Unless a stopping criterion has been satisfied, repeat the last two steps for another example.

Stopping criterion: each training example already finds itself in the nearest cluster.

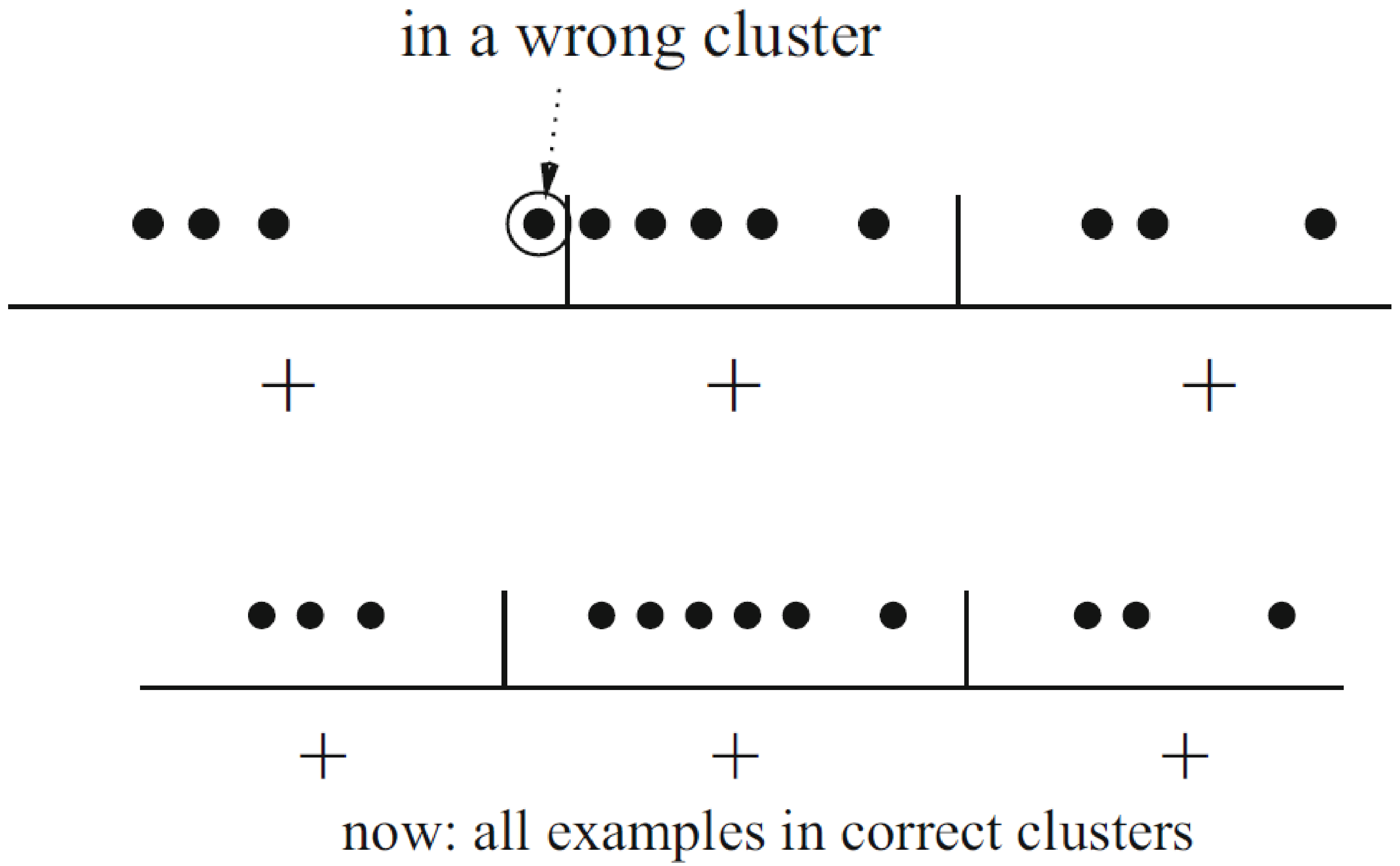
k-Means – Illustration

- Note that after the relocation of a data point, the centroids of the **two affected clusters** (the one that lost the example, and the one that gained it) have to be recalculated
- Graphical illustration by the single-attribute domain



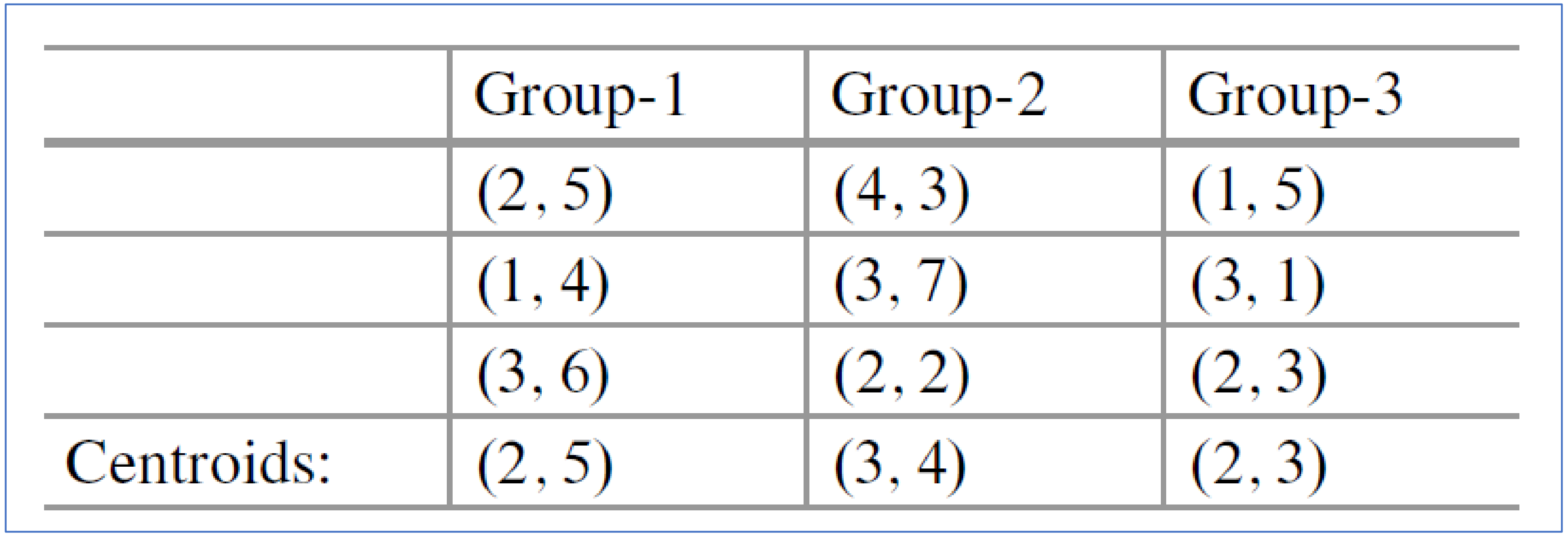
k-Means - Illustration

- **Graphical illustration by the single-attribute domain**



k-Means: Numeric Example

- The table below contains three initial groups of vectors. The task is to find "ideal" clusters using the k-means (**k = 3** assuming the user specified this value).



	Group-1	Group-2	Group-3
	(2, 5)	(4, 3)	(1, 5)
	(1, 4)	(3, 7)	(3, 1)
	(3, 6)	(2, 2)	(2, 3)
Centroids:	(2, 5)	(3, 4)	(2, 3)

k-Means: Numeric Example

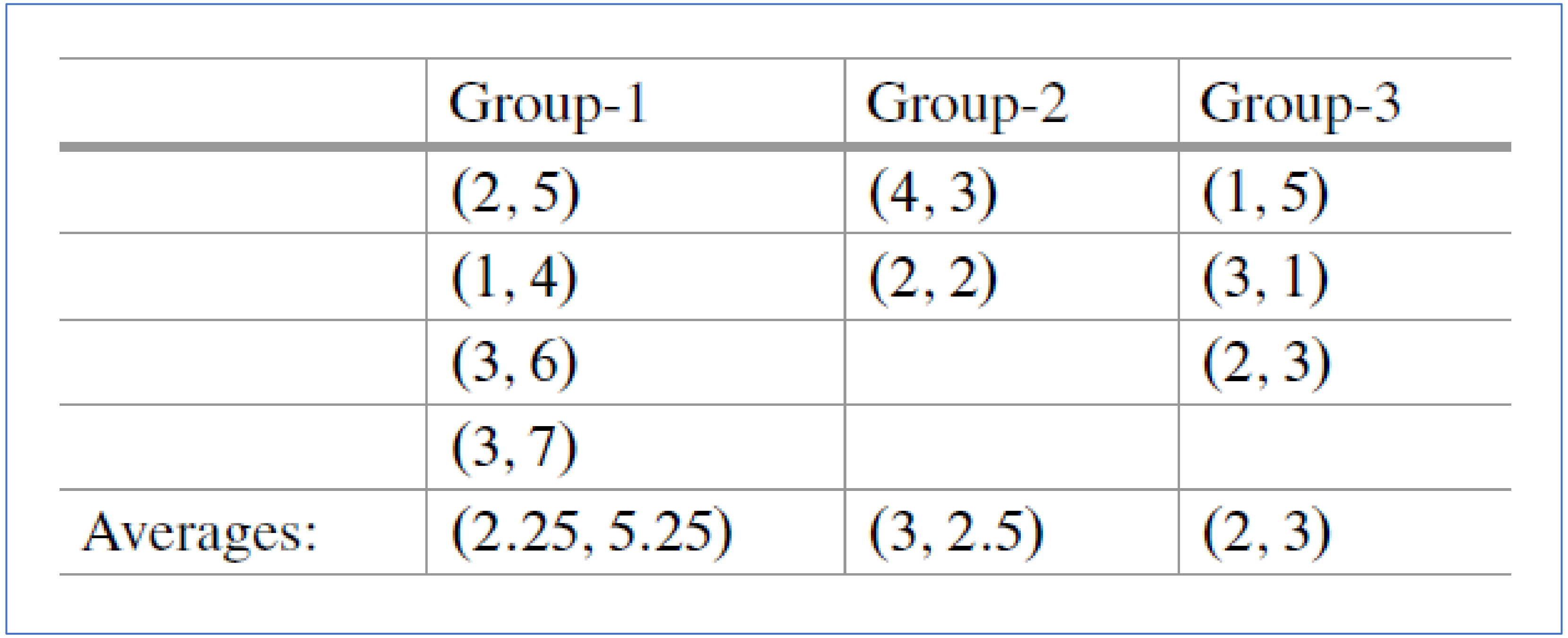
- **For each example, its distance from each centroid is calculated.**
- Let us pick $(4,3)$ in group-2. The Euclidean distances between this example, and the centroids of the three groups are : $\boldsymbol{\sqrt{8}}$**,** $\boldsymbol{\sqrt{2}}$, $\boldsymbol{\sqrt{4}}$ respectively.
- $(4,3)$ finds itself in the right cluster(group), k-means does not do anything

k-Means: Numeric Example

- Let us pick (3,7) from group-2. The Euclidean distances between this example, and the centroids are : $\sqrt{\mathbf{5}}$, $\sqrt{\mathbf{9}}$, $\sqrt{\mathbf{17}}$ respectively.
- (3,7) finds itself in the wrong cluster(group), because it is closest to group 1.
- So this example is moved from **group-2** to **group-1**. After this, the averages of the two affected groups are recalculated.

k-Means: Numeric Example

- **Here are the new clusters:**



	Group-1	Group-2	Group-3
	(2, 5)	(4, 3)	(1, 5)
	(1, 4)	(2, 2)	(3, 1)
	(3, 6)		(2, 3)
	(3, 7)		
Averages:	(2.25, 5.25)	(3, 2.5)	(2, 3)

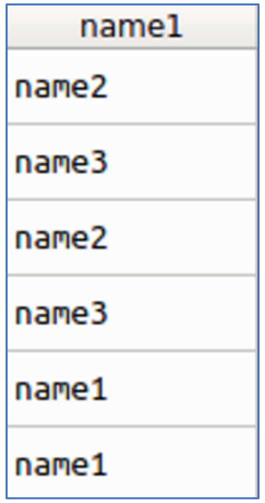
k-Means: The Need for Normalization

- It is always a good idea to **normalize the vectors** so that all **numeric attributes** have values from the same range, say, from 0 to 1.
- The simplest way of doing so is to determine for the given attribute its maximum (MAX) and minimum (MIN) value in the training set. Then, each value of this attribute is re-calculated using the following formula:

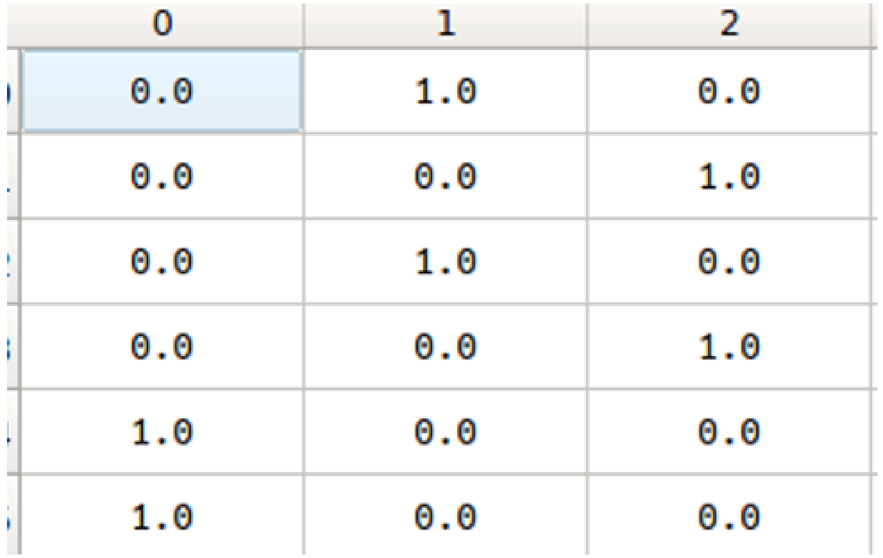
$$x = \frac{x - MIN}{MAX - MIN}$$

k-Means: The Need for Normalization

- As for **Boolean attributes,** their values can simply be replaced with 1 and 0 for true and false, respectively.
- Finally, an attribute that acquires **n discrete values** (such as season, which has four different values) can be replaced with **n Boolean attributes**, one for each value—and, again, for the values of these Boolean attributes, **1 or 0 are used.**
- **Example**



name1
name2
name3
name2
name3
name1
name1



0	1	2
0.0	1.0	0.0
0.0	0.0	1.0
0.0	1.0	0.0
0.0	0.0	1.0
1.0	0.0	0.0
1.0	0.0	0.0

k-Means: Computational Aspects of Initialization

- To reach its goal, the k-means needs to go through a certain number of transfers of examples from wrong clusters to the right clusters.
- How many such transfers are needed depends on the contents of the initial clusters.
- If initial clusters are already perfect, and not a single example needs to be moved
- Practically this is not always possible. Initialization matters in the sense that a **better initial clustering ensures that the solution is found sooner**.

k-Means: How to Initialize

- In some domains, we can take advantage of some background knowledge about the problem at hand.
- For instance, seeking to create initial clusters in a database of a company’s employees, the data analyst may speculate that it makes sense to group them by **their age, salary, or some other intuitive criterion**, and that the groups thus obtained will be good initial clusters.

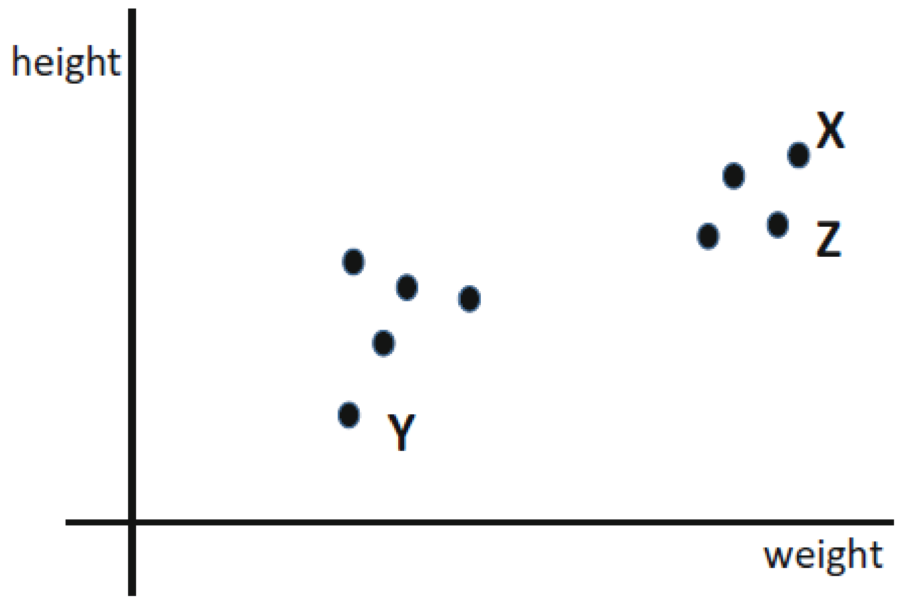
k-Means: How to Initialize

- In other applications, however, no such guidelines exist.
- The simplest procedure then picks k random training examples and regards them as **code vectors** to define initial centroids.

k-Means: A More Serious Problem with Initialization

- Note that the composition of the resulting clusters (once the k-means has completed its work) may depend on initialization.
- Choose a different set of initial code vectors, and the technique may generate a different set of clusters.

Fig. 14.3 Suppose $k = 2$. If the code vectors are [**x**,**y**], the initial clusters for *k*-means will be different than when the code vectors are [**x**,**z**]



k-Means: Good and Bad Things

- The good thing about k-means is that it is easy to explain and easy to implement.
- The technique is sensitive to initialization; the user is expected to provide the number of clusters.
- Some clusters can never be identified

k-Means: A complete Numerical Example

k-Means: A complete numerical example

- Apply k-means for the following dataset.

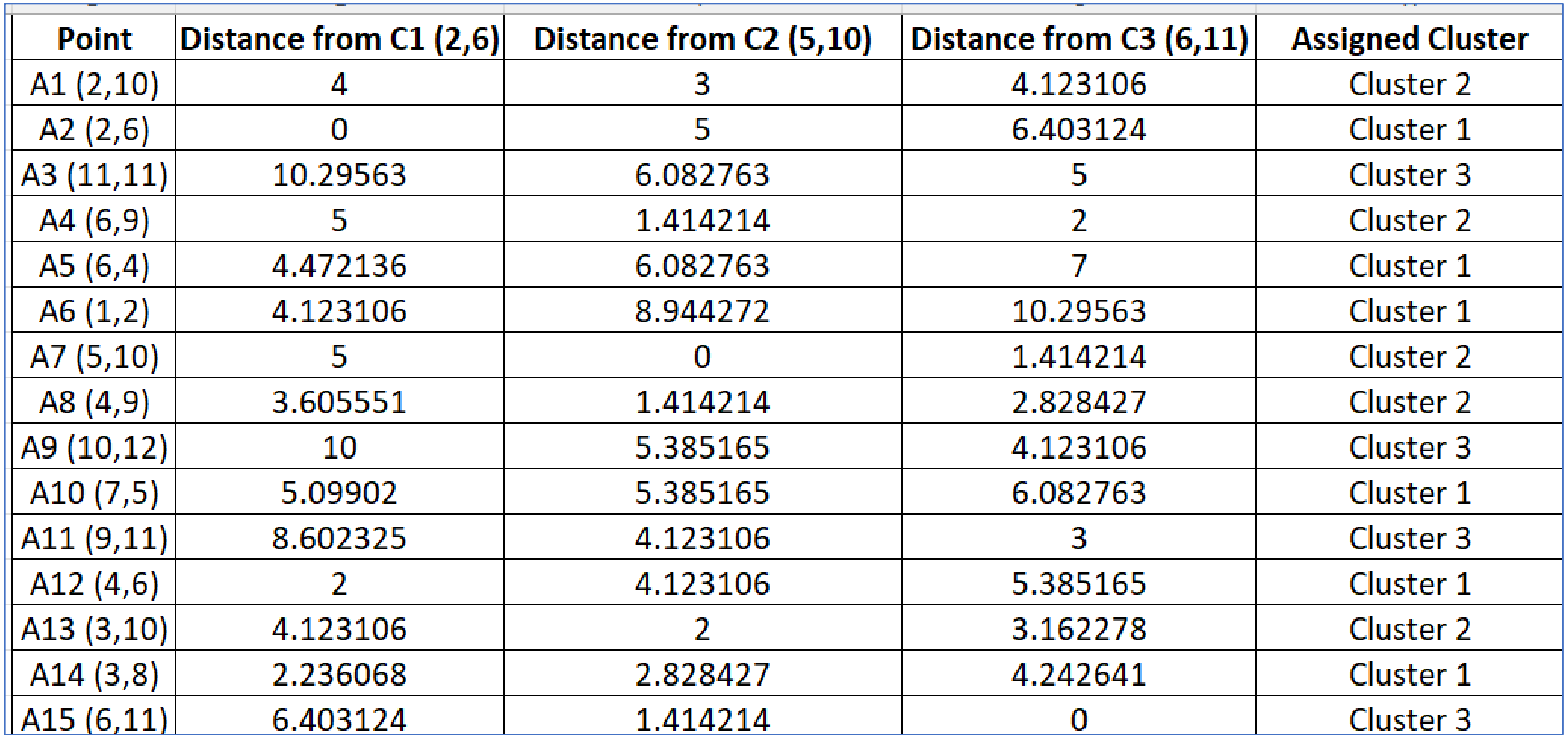
Point	Coordinates
A1	(2,10)
A2	(2,6)
A3	(11,11)
A4	(6,9)
A5	(6,4)
A6	(1,2)
A7	(5,10)
A8	(4,9)
A9	(10,12)
A10	(7,5)
A11	(9,11)
A12	(4,6)
A13	(3,10)
A14	(3,8)
A15	(6,11)

Initial centroids

- Centroid 1 C1=(2,6) is associated with cluster 1.
- Centroid 2 C2=(5,10) is associated with cluster 2.
- Centroid 3 C3=(6,11) is associated with cluster 3.

k-Means: A Complete numerical example

- Assign cluster for each data point



Point	Distance from C1 (2,6)	Distance from C2 (5,10)	Distance from C3 (6,11)	Assigned Cluster
A1 (2,10)	4	3	4.123106	Cluster 2
A2 (2,6)	0	5	6.403124	Cluster 1
A3 (11,11)	10.29563	6.082763	5	Cluster 3
A4 (6,9)	5	1.414214	2	Cluster 2
A5 (6,4)	4.472136	6.082763	7	Cluster 1
A6 (1,2)	4.123106	8.944272	10.29563	Cluster 1
A7 (5,10)	5	0	1.414214	Cluster 2
A8 (4,9)	3.605551	1.414214	2.828427	Cluster 2
A9 (10,12)	10	5.385165	4.123106	Cluster 3
A10 (7,5)	5.09902	5.385165	6.082763	Cluster 1
A11 (9,11)	8.602325	4.123106	3	Cluster 3
A12 (4,6)	2	4.123106	5.385165	Cluster 1
A13 (3,10)	4.123106	2	3.162278	Cluster 2
A14 (3,8)	2.236068	2.828427	4.242641	Cluster 1
A15 (6,11)	6.403124	1.414214	0	Cluster 3

k-Means: A Complete numerical example

Now, we will calculate the new centroid for each cluster:

- In cluster 1, we have 6 points i.e. A2 (2,6), A5 (6,4), A6 (1,2), A10 (7,5), A12 (4,6), A14 (3,8). Compute new centroid by taking average. The new centroid for cluster 1 is **(3.833, 5.167).**
- In cluster 2, we have 5 points i.e. A1 (2,10), A4 (6,9), A7 (5,10) , A8 (4,9), and A13 (3,10). Hence, the new centroid for cluster 2 is

  (4, 9.6)
- In cluster 3, we have 4 points i.e. A3 (11,11), A9 (10,12), A11 (9,11), and A15 (6,11). Hence, the new centroid for cluster 3 is

  (9, 11.25)

k-Means: A Complete numerical example

Reassign the datapoints

Point	Distance from C1 (3.833, 5.167)	Distance from C2 (4, 9.6)	Distance from C3 (9, 11.25)	Assigned Cluster
1 (2,10)	5.169	2.04	7.111	Cluster 2
A2 (2,6)	2.013	4.118	8.75	Cluster 1
A3 (11,11)	9.241	7.139	2.016	Cluster 3
A4 (6,9)	4.403	2.088	3.75	Cluster 2
A5 (6,4)	2.461	5.946	7.846	Cluster 1
A6 (1,2)	4.249	8.171	12.23	Cluster 1
A7 (5,10)	4.972	1.077	4.191	Cluster 2
A8 (4,9)	3.837	0.6	5.483	Cluster 2
A9 (10,12)	9.204	6.462	1.25	Cluster 3
A10 (7,5)	3.171	5.492	6.562	Cluster 1
A11 (9,11)	7.792	5.192	0.25	Cluster 3
A12 (4,6)	0.85	3.6	7.25	Cluster 1
A13 (3,10)	4.904	1.077	6.129	Cluster 2
A14 (3,8)	2.953	1.887	6.824	Cluster 2
A15 (6,11)	6.223	2.441	3.01	Cluster 2

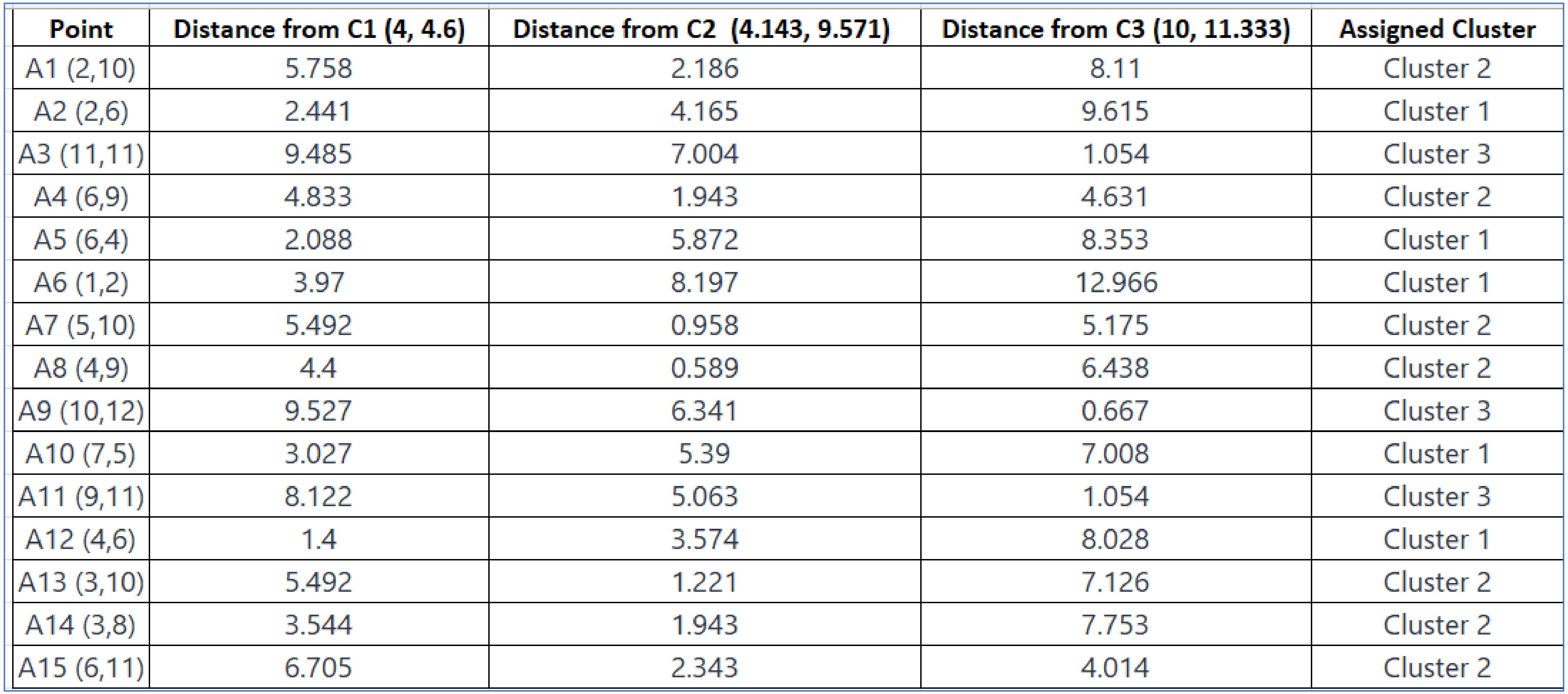
k-Means: A Complete numerical example

Now, we will calculate the new centroid for each cluster:

- In cluster 1, we have 5 points i.e. A2 (2,6), A5 (6,4), A6 (1,2), A10 (7,5), and A12 (4,6). The new centroid for cluster 1 is **(4, 4.6)**
- In cluster 2, we have 7 points i.e. A1 (2,10), A4 (6,9), A7 (5,10) , A8 (4,9), A13 (3,10), A14 (3,8), and A15 (6,11). Hence, the new centroid for cluster 2 is **(4.143, 9.571)**
- In cluster 3, we have 3 points i.e. A3 (11,11), A9 (10,12), and A11 (9,11). Hence, the new centroid for cluster 3 is **(10, 11.333).**

k-Means: A Complete numerical example

Reassign the datapoints



Point	Distance from C1 (4, 4.6)	Distance from C2 (4.143, 9.571)	Distance from C3 (10, 11.333)	Assigned Cluster
A1 (2,10)	5.758	2.186	8.11	Cluster 2
A2 (2,6)	2.441	4.165	9.615	Cluster 1
A3 (11,11)	9.485	7.004	1.054	Cluster 3
A4 (6,9)	4.833	1.943	4.631	Cluster 2
A5 (6,4)	2.088	5.872	8.353	Cluster 1
A6 (1,2)	3.97	8.197	12.966	Cluster 1
A7 (5,10)	5.492	0.958	5.175	Cluster 2
A8 (4,9)	4.4	0.589	6.438	Cluster 2
A9 (10,12)	9.527	6.341	0.667	Cluster 3
A10 (7,5)	3.027	5.39	7.008	Cluster 1
A11 (9,11)	8.122	5.063	1.054	Cluster 3
A12 (4,6)	1.4	3.574	8.028	Cluster 1
A13 (3,10)	5.492	1.221	7.126	Cluster 2
A14 (3,8)	3.544	1.943	7.753	Cluster 2
A15 (6,11)	6.705	2.343	4.014	Cluster 2

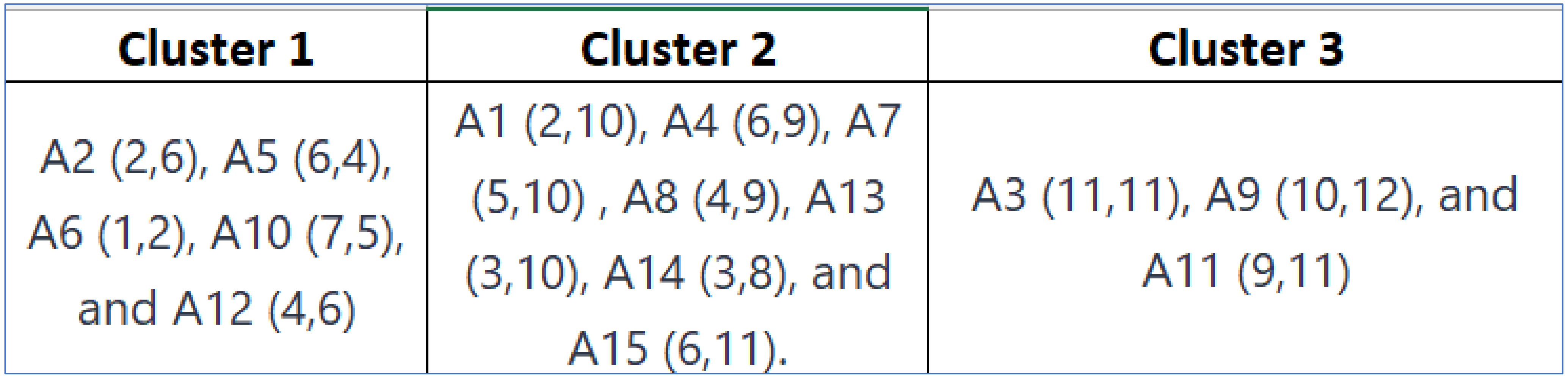
k-Means: A Complete numerical example

Now, we will calculate the new centroid for each cluster:

- In cluster 1, we have 5 points i.e. A2 (2,6), A5 (6,4), A6 (1,2), A10 (7,5), and A12 (4,6). The new centroid for cluster 1 is **(4, 4.6).**
- In cluster 2, we have 7 points i.e. A1 (2,10), A4 (6,9), A7 (5,10) , A8 (4,9), A13 (3,10), A14 (3,8), and A15 (6,11). Hence, the new centroid for cluster 2 is **(4.143, 9.571)**
- In cluster 3, we have 3 points i.e. A3 (11,11), A9 (10,12), and A11 (9,11). Hence, the new centroid for cluster 3 is **(10, 11.333).**

k-Means: A Complete numerical example

Here, you can observe that no point has changed its cluster compared to the previous iteration. Due to this, the centroid also remains constant. Therefore, we will say that the clusters have been stabilized. Final 3 clusters are given below.



Cluster 1	Cluster 2	Cluster 3
A2 (2,6), A5 (6,4), A6 (1,2), A10 (7,5), and A12 (4,6)	A1 (2,10), A4 (6,9), A7 (5,10) , A8 (4,9), A13 (3,10), A14 (3,8), and A15 (6,11).	A3 (11,11), A9 (10,12), and A11 (9,11)

centroid C1 **(4, 4.6).** centroid C2 **(4.143, 9.571)** centroid C3 **(10, 11.333)**

Cluster Analysis
Hierarchical Aggregation

Hierarchical Aggregation

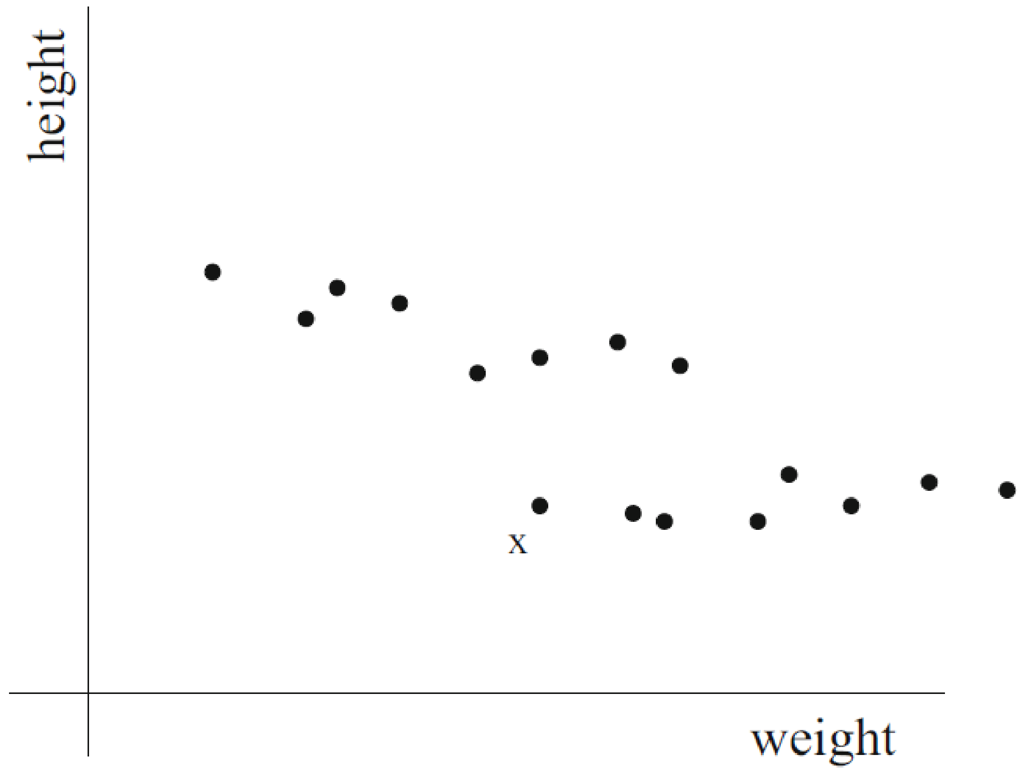
A Serious Limitation of k-Means

- The k-Means algorithm that we discussed previously associates a data point to the nearest centroid. Thus, clusters produces by the algorithm is known as **Convex Clusters.**
- Such clusters can be useful in the context of **Bayesian classifiers and the RBF networks** where they reduce, sometimes very significantly, the number of **employed gaussian centers.**

Hierarchical Aggregation

A Serious Limitation of k-Means

- This approach, however, will do a poor job if the clusters are of a different ("non-convex") nature.



Here, the leftmost example, x, in the bottom cluster is closer to the centroid of the upper cluster, and k-means would therefore relocate it accordingly—and yet we feel that this would not do justice to the nature of the two groups.

To deal with data of this kind, we need another technique, one capable of identifying clusters

Hierarchical Aggregation

A Serious Limitation of k-Means:

- Instead of associating a data point to its nearest cluster, we can use another method where we will measure the distances between all pairs of examples, [x,y], such that x comes from the first cluster and y from the second.
- The **smallest value** found among all these example-to-example distances then defines the **distance between the two clusters**.

Hierarchical Aggregation

A Serious Limitation of k-Means:

- **Numeric Example**

A
$\mathbf{x}_1 = (1, 0)$
$\mathbf{x}_2 = (2, 2)$

B
$\mathbf{y}_1 = (3, 3)$
$\mathbf{y}_2 = (4, 4)$

Using the Euclidean formula, we calculate the individual example-to-example distances as follows

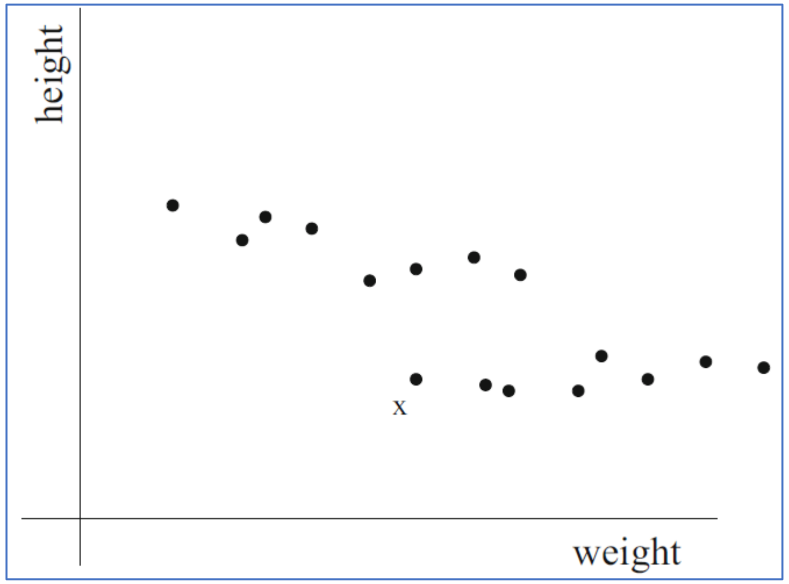
$$
\begin{aligned}
d(\mathbf{x}_1, \mathbf{y}_1) &= \sqrt{13}, \\
d(\mathbf{x}_1, \mathbf{y}_2) &= \sqrt{25}, \\
d(\mathbf{x}_2, \mathbf{y}_1) &= \sqrt{2}, \\
d(\mathbf{x}_2, \mathbf{y}_2) &= \sqrt{8}.
\end{aligned}
$$

We conclude that the distance between the two clusters is $d(A, B) = \sqrt{2}$

Hierarchical Aggregation

A Serious Limitation of k-Means:

- Using this distance calculation method example x is closer to the bottom cluster than to the upper clusters.
- This means that the limitation mentioned in the previous paragraph has been in this particular case eliminated.



Hierarchical Aggregation

A Serious Limitation of k-Means:

- The price for this improvement is increased computational costs: if N_A is the number of examples in the first cluster, and N_B the number of examples in the second, then $\boldsymbol{N_A} \times \boldsymbol{N_B}$ example-to-example distances have to be evaluated.
- Most of the time, however, the clusters are not going to so big as to make this an issue.

Hierarchical Aggregation

- For domains such as the one we discussed the clustering technique known as **hierarchical aggregation** is recommended.
- The principle is summarized by the pseudocode in Table 14.3.

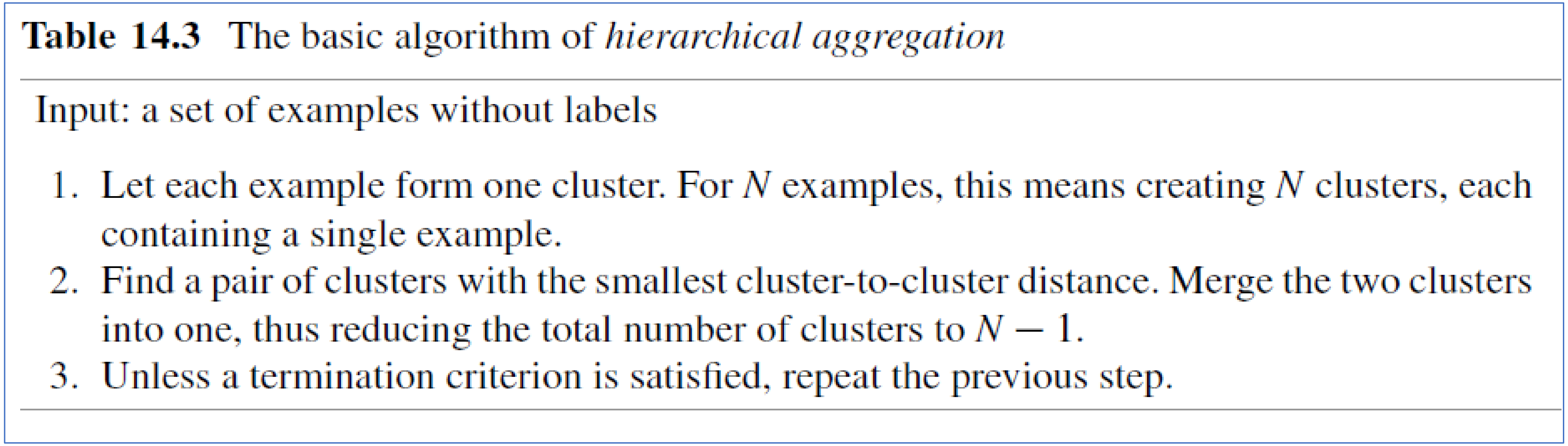
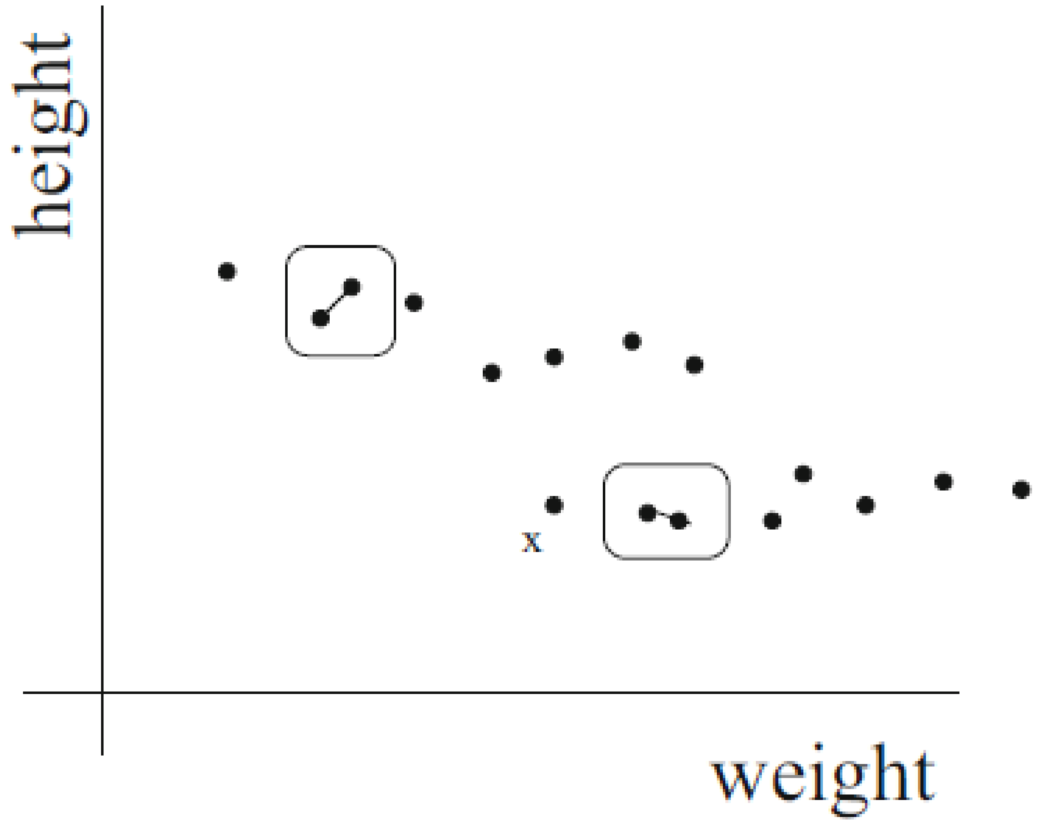


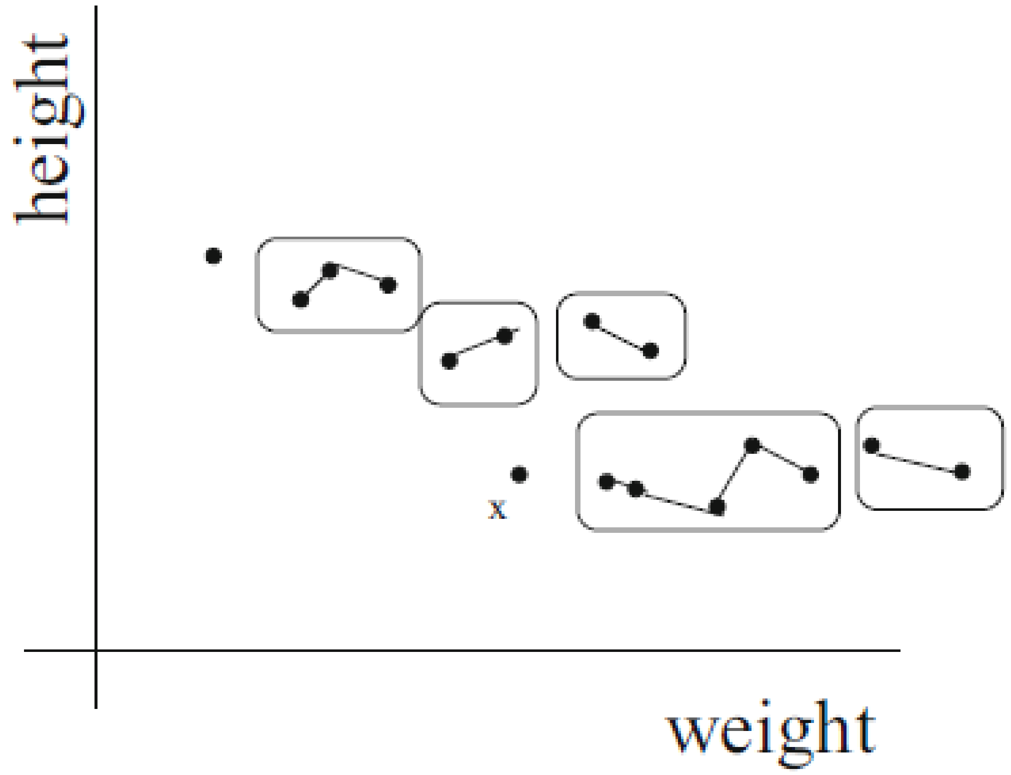
Table 14.3 The basic algorithm of *hierarchical aggregation*

```
Input: a set of examples without labels

  1. Let each example form one cluster. For $N$ examples, this means creating $N$ clusters, each
     containing a single example.
  2. Find a pair of clusters with the smallest cluster-to-cluster distance. Merge the two clusters
     into one, thus reducing the total number of clusters to $N - 1$.
  3. Unless a termination criterion is satisfied, repeat the previous step.
```

Hierarchical Aggregation





Hierarchical aggregation after first two steps (*left*)

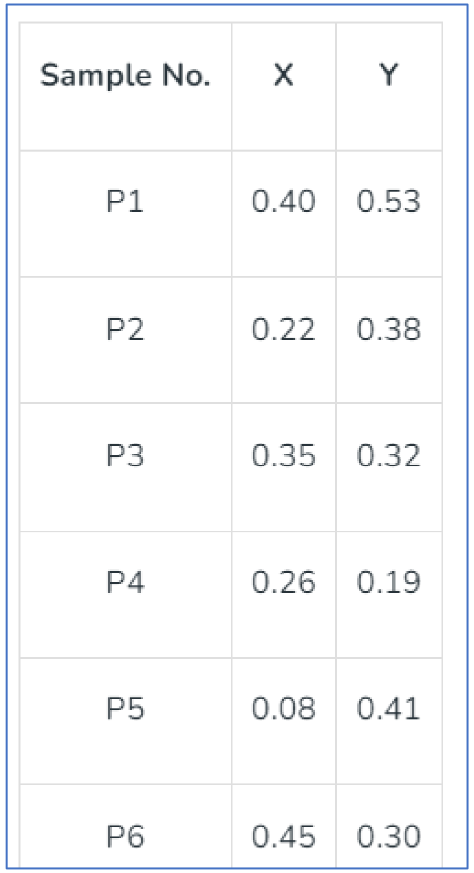
After first nine steps (*right*).

Hierarchical Aggregation

- Hierarchical aggregation always identifies a pair of clusters that have the smallest mutual distance along the distance metric from the previous paragraphs. These clusters are then merged.
- **The process continues until an appropriate termination criterion is satisfied.**

An example for Hierarchical Aggregation

- Find the clusters using a single link technique. Use Euclidean distance and draw the dendrogram.



Sample No.	X	Y
P1	0.40	0.53
P2	0.22	0.38
P3	0.35	0.32
P4	0.26	0.19
P5	0.08	0.41
P6	0.45	0.30

Note that $d(P_i, P_i) = 0 \; and$
$d(P_i, P_j) = d(P_j, P_i)$ thus matrix is symmetric

$$\begin{pmatrix} & P1 & P2 & P3 & P4 & P5 & P6 \\ P1 & 0 & & & & & \\ P2 & & 0 & & & & \\ P3 & & & 0 & & & \\ P4 & & & & 0 & & \\ P5 & & & & & 0 & \\ P6 & & & & & & 0 \end{pmatrix}$$

An example for Hierarchical Aggregation

- **Compute initial distances:**

$$d(P_2, P_1) = \sqrt{(0.4 - 0.22)^2 + (0.53 - 0.38)^2} = 0.23$$

$$d(P_3, P_1) = \sqrt{(0.35 - 0.40)^2 + (0.32 - 0.53)^2} = 0.22$$

$$d(P_3, P_2) = \sqrt{(0.35 - 0.22)^2 + (0.32 - 0.38)^2} = 0.14$$

$$d(P_4, P_1) = \sqrt{(0.26 - 0.4)^2 + (0.19 - 0.53)^2} = 0.37$$

$$d(P_4, P_2) = \sqrt{(0.26 - 0.22)^2 + (0.38 - 0.19)^2} = 0.19$$

$$d(P_4, P_3) = \sqrt{(0.26 - 0.35)^2 + (0.19 - 0.32)^2} = 0.13$$

An example for Hierarchical Aggregation

- **Compute initial distances:**

$$\boldsymbol{d(P_5,P_1) = \sqrt{(0.08-0.4)^2+(0.41-0.53)^2} = 0.34}$$
$$\boldsymbol{d(P_5,P_2) = \sqrt{(0.08-0.22)^2+(0.41-0.38)^2} = 0.14}$$
$$\boldsymbol{d(P_5,P_3) = \sqrt{(0.08-0.35)^2+(0.41-0.32)^2} = 0.28}$$
$$\boldsymbol{d(P_5,P_4) = \sqrt{(0.08-0.26)^2+(0.41-0.19)^2} = 0.23}$$
$$\boldsymbol{d(P_6,P_1) = \sqrt{(0.45-0.4)^2+(0.30-0.53)^2} = 0.24}$$
$$\boldsymbol{d(P_6,P_2) = \sqrt{(0.45-0.22)^2+(0.30-0.38)^2} = 0.24}$$
$$\boldsymbol{d(P_6,P_3) = \sqrt{(0.45-0.35)^2+(0.30-0.32)^2} = 0.10}$$
$$\boldsymbol{d(P_6,P_4) = \sqrt{(0.45-0.26)^2+(0.30-0.19)^2} = 0.22}$$
$$\boldsymbol{d(P_6,P_5) = \sqrt{(0.45-0.08)^2+(0.30-0.41)^2} = 0.39}$$

An example for Hierarchical Aggregation

- **Compute initial distance matrix is :**

$$\begin{pmatrix} & P1 & P2 & P3 & P4 & P5 & P6 \\ P1 & 0 & & & & & \\ P2 & 0.23 & 0 & & & & \\ P3 & 0.22 & 0.14 & 0 & & & \\ P4 & 0.37 & 0.19 & 0.13 & 0 & & \\ P5 & 0.34 & 0.14 & 0.28 & 0.23 & 0 & \\ P6 & 0.24 & 0.24 & 0.10 & 0.22 & 0.39 & 0 \end{pmatrix}$$

An example for Hierarchical Aggregation

- Here the **minimum value is 0.10** and hence we combine P3 and P6. Now, form clusters of elements corresponding to the minimum value and update the distance matrix.

min ((P3,P6), P1) = min ((P3,P1), (P6,P1)) = min (0.22,0.24) = 0.22

min ((P3,P6), P2) = min ((P3,P2), (P6,P2)) = min (0.14,0.24) = 0.14

min ((P3,P6), P4) = min ((P3,P4), (P6,P4)) = min (0.13,0.22) = 0.13

min ((P3,P6), P5) = min ((P3,P5), (P6,P5)) = min (0.28,0.39) = 0.28

An example for Hierarchical Aggregation

- Now we will update the Distance Matrix:

- $$\begin{pmatrix}
 & P1 & P2 & P3, P6 & P4 & P5 \\
P1 & 0 & & & & \\
P2 & 0.23 & 0 & & & \\
P3, P6 & 0.22 & 0.14 & 0 & & \\
P4 & 0.37 & 0.19 & 0.13 & 0 & \\
P5 & 0.34 & 0.14 & 0.28 & 0.23 & 0
\end{pmatrix}$$

An example for Hierarchical Aggregation

- The minimum value is **0.13** and hence we combine **P3, P6 and P4**. Now, form the clusters of elements corresponding to the minimum values and update the Distance matrix.

min (((P3,P6) P4), P1) = min (((P3,P6), P1), (P4,P1)) = min (0.22,0.37) = 0.22

min (((P3,P6), P4), P2) = min (((P3,P6), P2), (P4,P2)) = min (0.14,0.19) = 0.14

min (((P3,P6), P4), P5) = min (((P3,P6), P5), (P4,P5)) = min (0.28,0.23) = 0.23

An example for Hierarchical Aggregation

- Now we will update the Distance Matrix:

$$\begin{pmatrix} & P1 & P2 & P3,P6,P4 & P5 \\ P1 & 0 & & & \\ P2 & 0.23 & 0 & & \\ P3,P6,P4 & 0.22 & 0.14 & 0 & \\ P5 & 0.34 & 0.14 & 0.23 & 0 \end{pmatrix}$$

An example for Hierarchical Aggregation

- The minimum value is **0.14** and hence we combine P2 and P5. Now, form cluster of elements corresponding to minimum value and update the distance matrix.

min ((P2,P5), P1) = min ((P2,P1), (P5,P1)) = min (0.23, 0.34) = 0.23

min ((P2,P5), (P3,P6,P4)) = min ((P3,P6,P4), (P3,P6,P4)) = min (0.14. 0.23) = 0.14

Update Distance Matrix will be:

$$\begin{pmatrix} & P1 & P2, P5 & P3, P6, P4 \\ P1 & 0 & & \\ P2, P5 & 0.23 & 0 & \\ P3, P6, P4 & 0.22 & 0.14 & 0 \end{pmatrix}$$

An example for Hierarchical Aggregation

- The minimum value is 0.14 and hence we combine P2,P5 and P3,P6,P4. Now, form cluster of elements corresponding to minimum value and update the distance matrix.

  min ((P2,P5,P3,P6,P4), P1) = min ((P2,P5), P1), ((P3,P6,P4), P1)) = min (0.23, 0.22) = 0.22

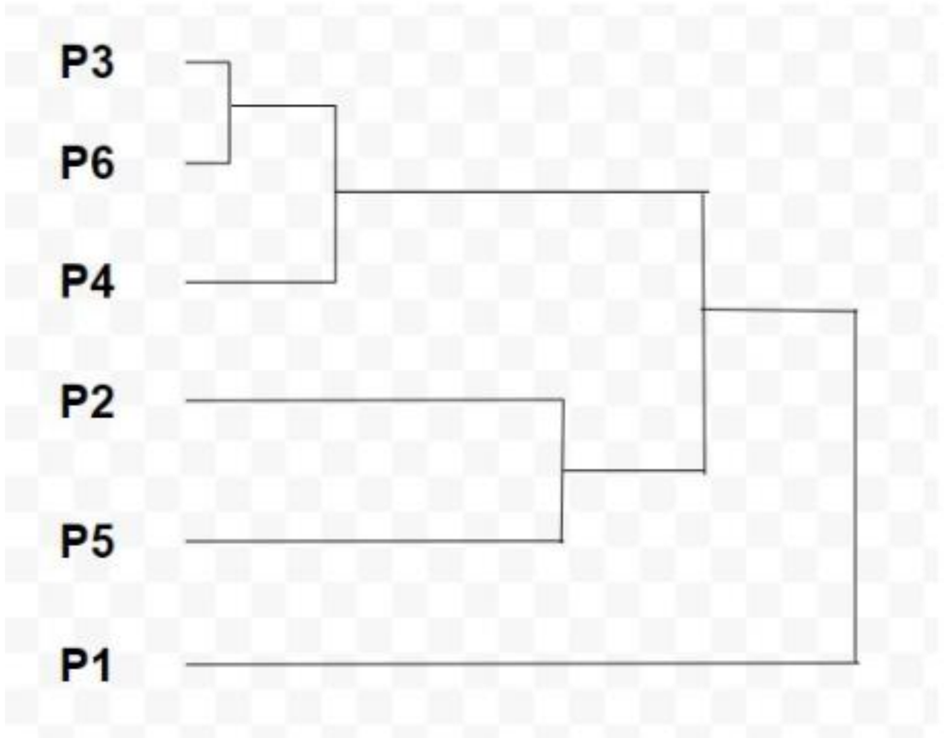
-

Updated Distance Matrix will be:

$$\begin{pmatrix} & P1 & P2, P5, P3, P6, P4 \\ P1 & 0 & \\ P2, P5, P3, P6, P4 & 0.22 & 0 \end{pmatrix}$$

An example for Hierarchical Aggregation

- The dendrogram of the hierarchical clustering is :



Dendogram of the cluster formed

Advantages & Disadvantages

- Hierarchical aggregation always identifies a pair of clusters that have the smallest mutual distance along the distance metric from the previous paragraphs. These clusters are then merged.
- **The process continues until an appropriate termination criterion is satisfied.**

Principal Component Analysis (PCA)

What Is Principal Component Analysis?

- **Principal component analysis**, or **PCA**, is a **dimensionality reduction method** that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one **that still contains most of the information in the large set.**

What Is Principal Component Analysis?

- **Basic Idea :** PCA uses an **orthogonal transformation** that converts a set of **correlated variables** to a set of **uncorrelated variables**. These new transformed features are called the **Principal Components.**
- It works on the condition that while the data in **a higher dimensional** space is mapped **to data in a lower dimension space**, the **variance of the data in the lower dimensional space should be maximum**.

What Is Principal Component Analysis?

- Principal Component Analysis(PCA) technique was introduced by the mathematician **Karl Pearson in 1901.**
- PCA is the most widely used tool in **exploratory data analysis** and in machine learning for **predictive models**

What Is Principal Component Analysis?

- Reducing the number of variables of a data set naturally **comes at the expense of accuracy**, but the trick in dimensionality reduction is to **trade a little accuracy for simplicity.**

- Because **smaller data sets are easier to explore and visualize**, and thus **make analyzing data points much easier and faster** for machine learning algorithms without extraneous variables to process.

What Is Principal Component Analysis?

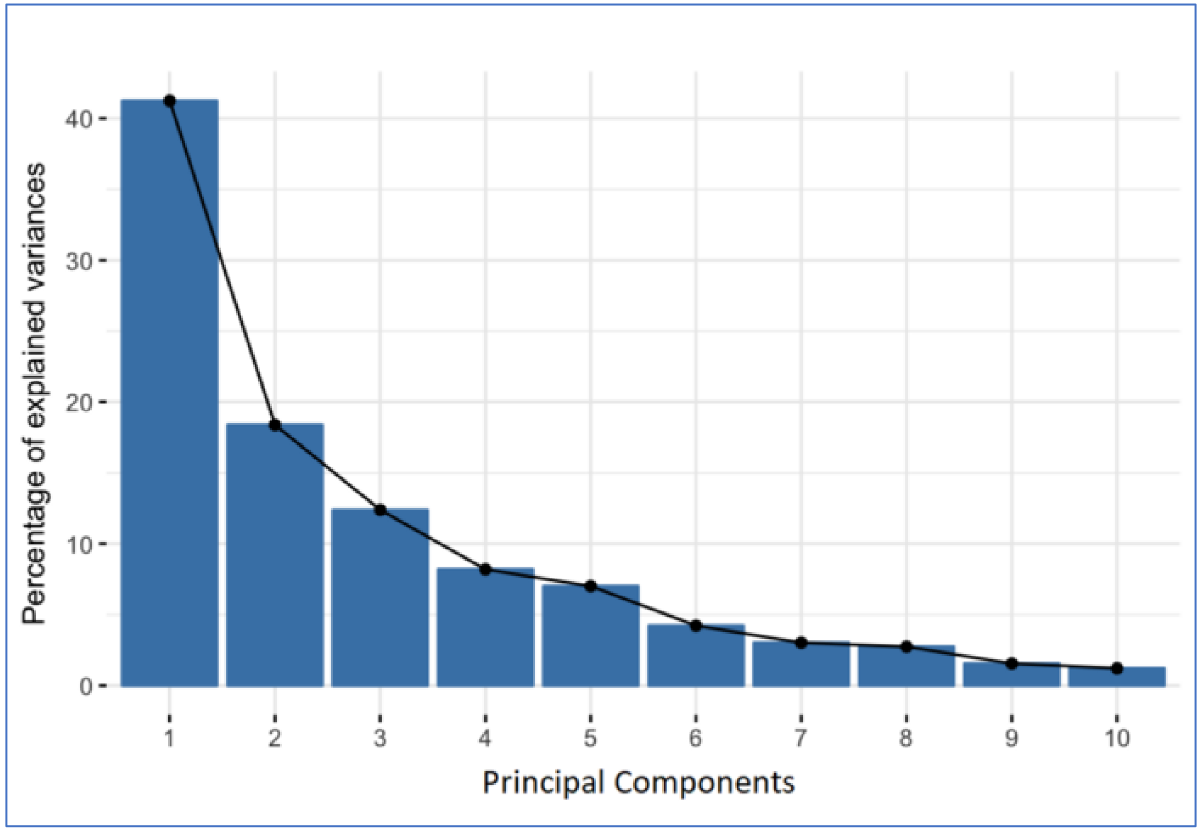
- The main goal of PCA is to **reduce the dimensionality of a dataset** while **preserving the most important patterns** or relationships between the variables **without any prior knowledge of the target variables (unsupervised learning)**.

What Are Principal Components?

- **Principal components are new variables** that are constructed as linear combinations or mixtures of the initial variables.
- These combinations are done in such a way that the new variables (i.e., principal components) **are uncorrelated** and **most of the information within the initial variables** is squeezed or **compressed into the first components**.

What Are Principal Components?

- So, the idea is **10-dimensional data gives you 10 principal components,** but PCA tries to put **maximum possible information in the first component**, then **maximum remaining information in the second and so on**, until having something like shown in the scree plot below.



- **Discard the components with low information** and consider the **remaining components as your new variables.**

What Are Principal Components?

- In other words, the **first principal component captures the most variation in the data**, but **the second principal component captures the maximum variance that is orthogonal to the first principal component**, and so on.
- The total variance captured by all the principal components is equal to the total variance in the original dataset.

PCA Example

- Let’s say we have a data set of dimension **300 (n) × 50 (p). n represents the number of observations**, and **p represents the number of predictors.**

- For 2D plot, there are $C_2^p = \frac{p(p-1)}{2} > 1000$ scatter plots. That is i.e., more than 1000 plots possible to analyze the variable relationship.

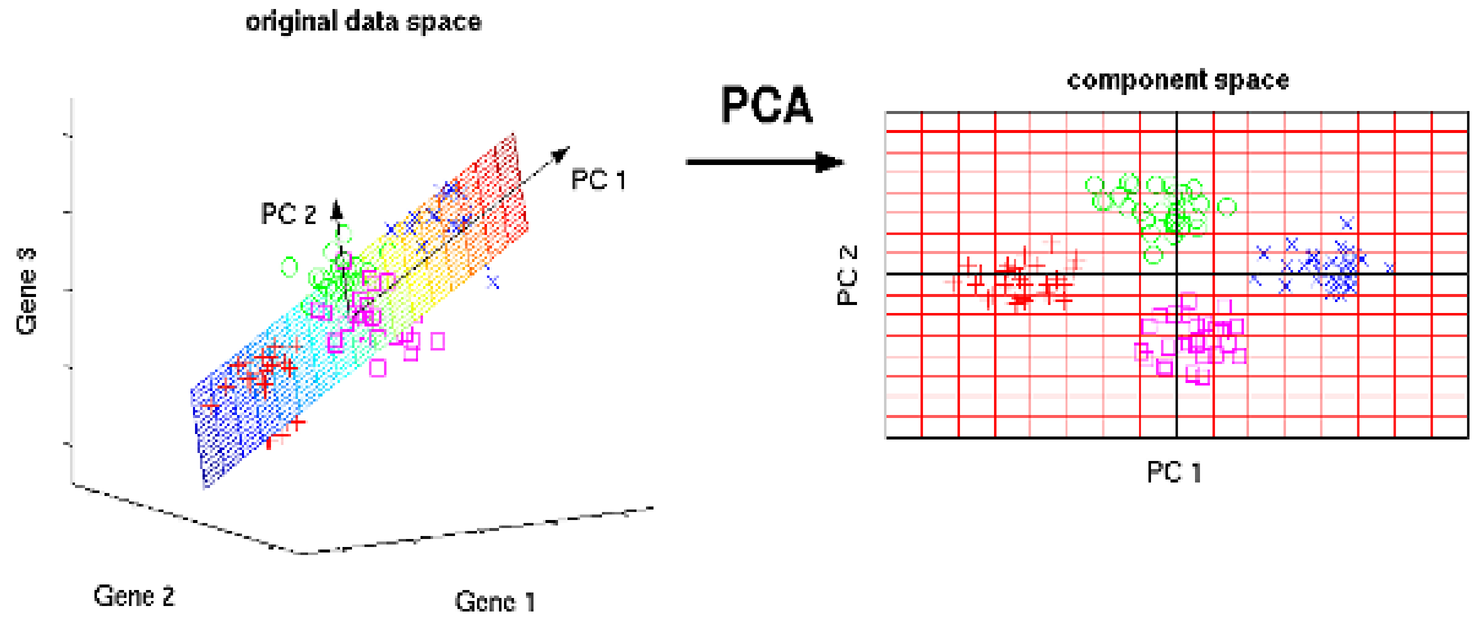
- Thus, this would be a tedious job to perform **exploratory analysis on this data**

PCA Example

- In this case, it would be a lucid approach to select a subset of p (p << 50) predictor **which captures so much information**, followed by plotting the observation in the resultant **low-dimensional space.**
-

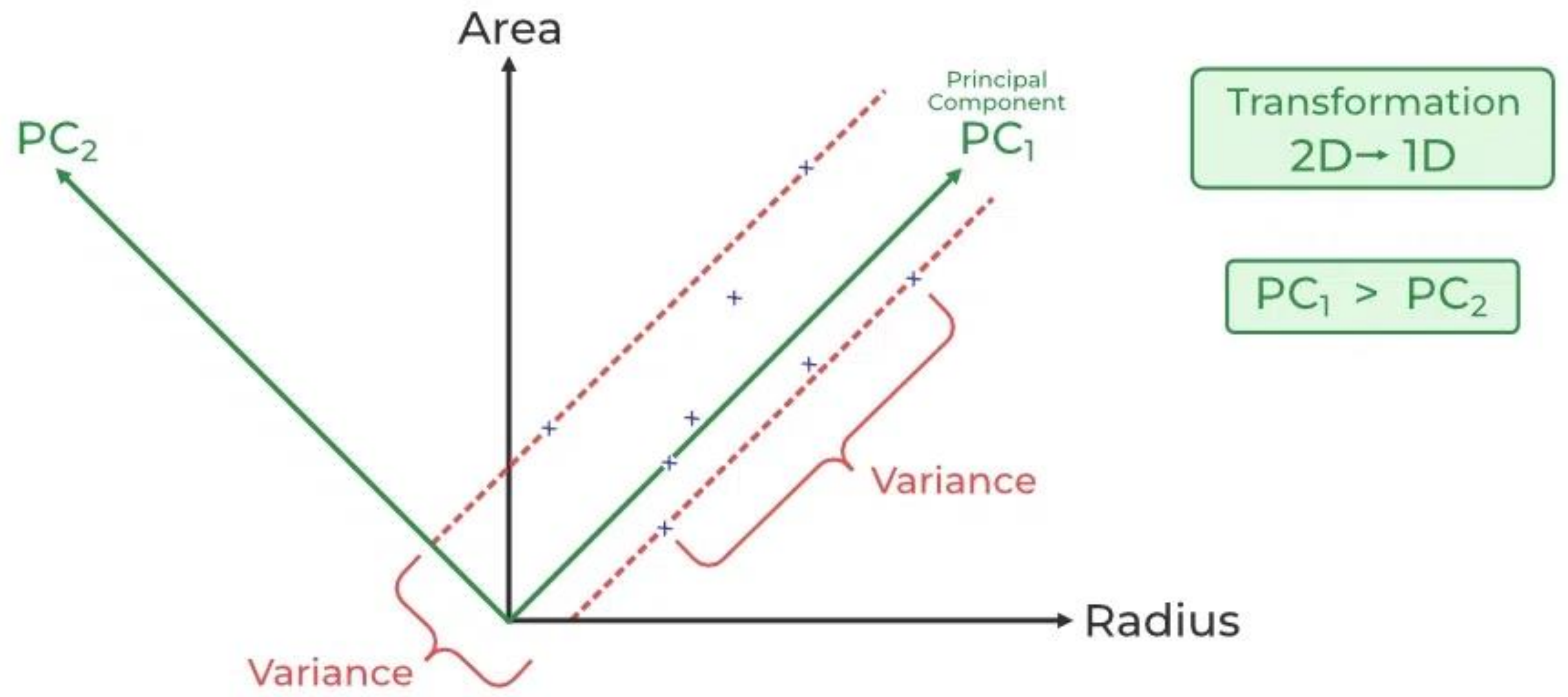
The transformation of high-dimensional data (3 dimension) to low-dimensional data (2 dimension) using PCA.

Not to forget, each resultant dimension is a linear combination of *p* features



PCA Example

- 2D to 1D



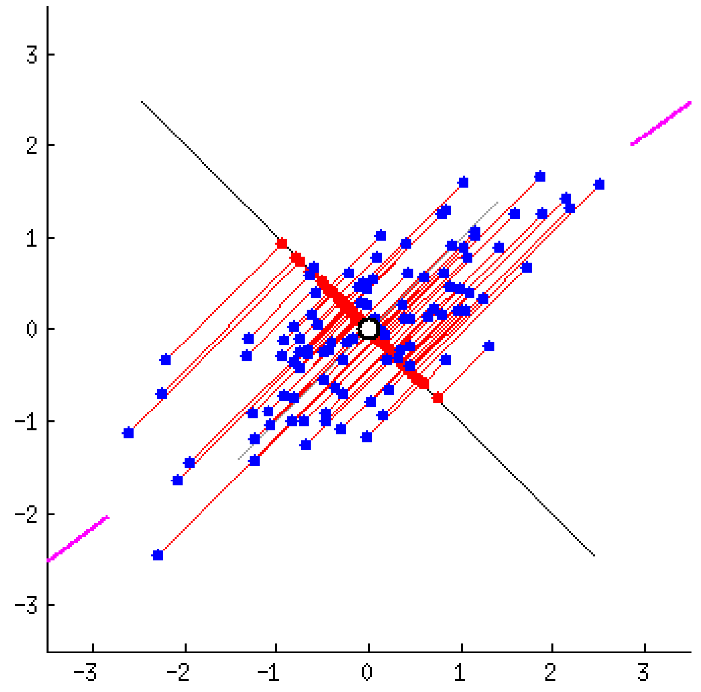
PCA Example

- **Geometrically speaking**, principal components represent **the directions of the data that explain a maximal amount of variance**, that is to say, the **lines that capture most information of the data**.
- The relationship between **variance and information** here, is that, the **larger the variance carried by a line, the larger the dispersion of the data points along it,** and **the larger the dispersion along a line, the more information it has. (the higher the variation in a feature, the more information that features carries.)**

How PCA Constructs the Principal Components

- **Principal components are constructed in such a manner that the first principal component accounts for the largest possible variance in the data set.**

Here first PC is the line that matches the purple marks because it's the line in which the projection of the points (red dots) is the most spread out.



It's the line that maximizes the variance (**the average of the squared distances from the projected points (red dots) to the origin**).

How PCA Constructs the Principal Components

- The second principal component is calculated in the same way, with the condition that it is uncorrelated with (i.e., perpendicular to) the first principal component and that it accounts for the next highest variance.
- This continues until a total of p principal components have been calculated, equal to the original number of variables.

Steps in PCA

1. **Standardization**
2. **Covariance Matrix Computation**
3. **Compute Eigenvalues and Eigenvectors of Covariance Matrix to Identify Principal Components**
4. **Create a Future Vector**
5. **Recast the Data along the Principal Components**

Steps in PCA - STANDARDIZATION

- If there are large differences between the ranges of initial variables, those variables with larger ranges will dominate over those with small ranges.
- For example, a variable that ranges between **0 and 100 will dominate over a variable that ranges between 0 and 1**, which will lead to biased results. So, transforming the data to comparable scales can prevent this problem. This can be done using the equation

$$z = \frac{value - mean}{standard\ deviation}$$

$$Z = \frac{X - \mu}{\sigma}$$

Once the standardization is done, all the variables will be transformed to the same scale.

Steps in PCA - COVARIANCE MATRIX COMPUTATION

- This step checks if there is any relationship between variables.
- Sometimes, variables are highly correlated in such a way that they contain redundant information. So, in order to identify these correlations, we compute the covariance matrix.
- The covariance matrix is a **p × p symmetric matrix** (where p is the number of dimensions) that has as entries the covariances associated with all possible pairs of the initial variables.

Covariance Matrix for 3-Dimensional Data

$$\begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix}$$

Steps in PCA - COVARIANCE MATRIX COMPUTATION

Note

- The covariance of a variable with itself is its variance **(Cov(a,a)=Var(a)), so** in the main diagonal we actually have the **variances of each initial variable.**
- The covariance is commutative (Cov(a,b)=Cov(b,a)), the entries of the covariance matrix are symmetric.
- If covariance is positive, then the two variables increase or decrease together (**correlated**).
- If negative, then one increases when the other decreases (**Inversely correlated**)
- **Zeros:** No direct relation

Steps in PCA - COVARIANCE MATRIX COMPUTATION

Covariance Formula :

$$cov_{x,y} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{N - 1}$$

$$Cov_{x,x} = \frac{\sum (x_i - \bar{x})^2}{N-1} = Var(x)$$

x_i	= data value of x
y_i	= data value of y
$\bar{x}$	= mean of x
$\bar{y}$	= mean of y
N	= number of data values

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- Next step is to compute **Eigenvectors and Eigenvalues** compute from the covariance matrix in order to determine the principal components of the data.
- **Eigenvectors and Eigenvalues** always come in pairs, so that **every eigenvector has an eigenvalue.**
- Also, **their number is equal to the number of dimensions of the data.** For example, for a 3-dimensional data set, there are 3 variables, therefore there are 3 eigenvectors with 3 corresponding eigenvalues.

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- The eigenvectors of the Covariance matrix are actually **the directions of the axes** where there is the **most variance (most information)** and that we call **Principal Components**.
- Eigenvalues are simply the **coefficients attached to eigenvectors**, which give the amount of variance carried in each Principal Component.

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- The eigenvectors of the Covariance matrix are actually **the directions of the axes** where there is the **most variance (most information)** and that we call **Principal Components**.
- Eigenvalues are simply the **coefficients attached to eigenvectors**, which give the amount of variance carried in each Principal Component.
- By **ranking your eigenvectors in order of their eigenvalues**, highest to lowest, you get the principal components in order of significance.

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- Let A be a square $N \times N$ matrix and X be a non-zero vector for which

  $AX = \lambda X$ for some scalar value λ

- Here λ is known as the **eigenvalue** of matrix A and X is known as the **eigenvector** of matrix A for the corresponding eigenvalue.

- **It can also be written as :**

  $\boldsymbol{(A - \lambda)X = (A - \lambda I)X = 0}$ **where I is the identity matrix of size** $N \times N$

- The above condition will be true if

  $\boldsymbol{|A - \lambda I| = 0}$ (if determinant is zero (singular matrix))

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- Note that from the above equation we can find we can find the eigenvalues λ and therefore corresponding eigenvector can be found using the equation:

$$AX = \lambda X$$

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- Let's suppose that our data set is 2-dimensional with 2 variables x, y and that the eigenvectors and eigenvalues of the covariance matrix are as follows:

$$v1 = \begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix} \qquad \lambda_1 = 1.284028$$

$$v2 = \begin{bmatrix} -0.7351785 \\ 0.6778736 \end{bmatrix} \qquad \lambda_2 = 0.04908323$$

- Since $\lambda1>\lambda2$, which means that the **principal component (PC1)** $\boldsymbol{will\ have}$ **more information than** the **principal component (PC2)**

Steps in PCA - COMPUTE THE EIGENVECTORS AND EIGENVALUES OF THE COVARIANCE MATRIX TO IDENTIFY THE PRINCIPAL COMPONENTS

- To compute the percentage of variance (information) accounted for by each component, we divide the eigenvalue of each component by the sum of eigenvalues.
- If we apply this on the example above, we find that PC1 and PC2 carry respectively **96 percent and 4 percent** of the variance of the data.

Steps in PCA - CREATE A FEATURE VECTOR

- In this step, we choose whether to keep all these components or discard those of lesser significance (of **low eigenvalues**), and **form with the remaining ones a matrix of vectors that we call Feature vector**.
- So, the feature vector is simply a matrix that has as **columns the eigenvectors of the components** that we decide to keep.
- If we choose to keep only **p eigenvectors (components) out of n**, the final data set will have only **p dimensions.**

Steps in PCA - CREATE A FEATURE VECTOR

- Continuing with the example from the previous step, we can either form a feature vector with both of the eigenvectors v_1 and v_2:

$$\begin{bmatrix} 0.6778736 & -0.7351785 \\ 0.7351785 & 0.6778736 \end{bmatrix}$$

- Or discard the eigenvector v_2, which is the one of lesser significance, and form a feature vector with v1 only:

$$\begin{bmatrix} 0.6778736 \\ 0.7351785 \end{bmatrix}$$

Steps in PCA - CREATE A FEATURE VECTOR

- Given that v_2 was carrying only **4 percent of the information**, the loss will be therefore not important and we will still have **96 percent of the information** that is carried by v_1 .

Steps in PCA - RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES

- Given that we selected the principal components and form the feature vector in the previous step, but the input data set remains always in terms **of the original axes** (i.e, in **terms of the initial variables**).
- In this step, we use the feature vector formed using the eigenvectors of the covariance matrix, **to reorient the data from the original axes to the ones represented by the principal components** (hence the name Principal Components Analysis).

Steps in PCA - RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES

- This can be done by multiplying the transpose of the original data set by the transpose of the feature vector.

$$FinalDataSet = FeatureVector^T * StandardizedOriginalDataSet^T$$

PCA - Example

PCA - Example

- Let us start by considering the following two three-sample attribute traces

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \text{ and } Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}.$$

- These two attribute vectors can be combined into the matrix S as

$$S = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

PCA - Example

- We then compute the following covariance matrix C of the matrix S using the covariance formula:

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

$$C = \begin{bmatrix} \sigma_{XX} & \sigma_{XY} \\ \sigma_{YX} & \sigma_{YY} \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$$

- **In this case** , the covariance matrix C can also be computed in the following way:

$$C = S^T S = \begin{bmatrix} 1 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}.$$

PCA - Example

- We now want to compute the eigenvalues and eigenvectors of C. The eigenvalue equation is given by:

$$\boldsymbol{Cu} = \boldsymbol{\lambda u}$$

where λ is an eigenvalue (a scalar) and $\mathrm{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ is an eigenvector. Note that for this example there are two eigenvectors (since there are two variables) and each eigenvector has an associated eigenvalue.

- Note that a **covariance matrix is always a square matrix**.
- Also note that **all eigenvalues of the variance covariance matrix** are real and non-negative.

PCA - Example

- First we compute the eigenvalues and eigenvectors of C. We know that

$$\boldsymbol{Cu} = \boldsymbol{\lambda u}$$

$$(\boldsymbol{C} - \boldsymbol{\lambda})\boldsymbol{u} = (\boldsymbol{C} - \boldsymbol{\lambda I})\boldsymbol{u} = \boldsymbol{0}$$

provided $|\boldsymbol{C} - \boldsymbol{\lambda I}| = \boldsymbol{0}$ where $\boldsymbol{I}$ is the identity matrix. Thus,

- $$\begin{vmatrix} 2-\lambda & -1 \\ -1 & 2-\lambda \end{vmatrix} = 0 \Rightarrow \lambda^2 - 4\lambda + 3 = 0 \Rightarrow \lambda_1 = 3, \text{and } \lambda_2 = 1.$$

- Here $\boldsymbol{\lambda}_1 = 3$ *and* $\boldsymbol{\lambda}_1 = 1$ *are the eigenvalues.*

PCA - Example

- **To find** eigenvectors we substitute eigenvalues in the following equation

$$(\boldsymbol{C} - \boldsymbol{\lambda I})\boldsymbol{u} = \boldsymbol{0}$$

$$\begin{bmatrix} \boldsymbol{2 - \lambda} & \boldsymbol{-1} \\ \boldsymbol{-1} & \boldsymbol{2 - \lambda} \end{bmatrix} \begin{bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{0} \end{bmatrix}$$

For $\lambda_1 = 3$ we get $\boldsymbol{u_1 = -u_2}$. As we can see, there are infinite values that satisfy this, But we always consider the simplest eigenvector. Thus $\begin{bmatrix} \boldsymbol{u_1} \\ \boldsymbol{u_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{-1} \end{bmatrix}$

For $\lambda_2 = 1$ we get $\boldsymbol{v_1 = v_2}$. As we can see, there are infinite values that satisfy this, But we always consider the simplest eigenvector. Thus , $\begin{bmatrix} \boldsymbol{v_1} \\ \boldsymbol{v_2} \end{bmatrix} = \begin{bmatrix} \boldsymbol{1} \\ \boldsymbol{1} \end{bmatrix}$

PCA - Example

- **It is also general practice to find the simplest eigenvector in each case by normalizing it so that the sum of the squares of its components equals 1.**

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \text{ and } \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

- To normalize a vector, we divide the vector(each element) by its magnitude.
- If $X = (x_1, x_2, x_3)$ is a vector, then the normalized vector is

$$\boldsymbol{\hat{X} = \frac{X}{|X|}\ where\ |X| = \sqrt{x_1{}^2 + x_2{}^2 + x_3{}^2}\ (magnitude)}$$

PCA - Example

- These two eigenvectors, which represent principal components can be put into matrix form as

$$U = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}.$$

- The eigenvector matrix is orthonormal, which means that when it is multiplied by its transpose, we get the identity matrix,

$$UU^T = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

- Also, the transpose and inverse of U are identical. That is:

$$U^T = U^{-1} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}.$$

PCA - Example

- Now we can recast the data along the principal components. This can be done by multiplying the components of each eigenvector by the attribute vectors and summing the result.

$$P_1 = u_1 X + u_2 Y, \text{ and}$$
$$P_2 = v_1 X + v_2 Y.$$

- Substituting the values for the eigenvectors, :

$$P_1 = \frac{1}{\sqrt{2}}(X - Y), \text{ and } P_2 = \frac{1}{\sqrt{2}}(X + Y).$$

PCA - Example

- Now

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \text{ and } Y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}.$$

- Thus, we get,

$$P_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 2 \\ -1 \\ -1 \end{bmatrix} \text{ and } P_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}.$$

PCA - Example

- In the matrix form we write :

$$P = \frac{1}{\sqrt{2}}\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}.$$

-

- Note that if P is multiplied by its transpose, we can recover eigenvalue

- $$P^T P = \frac{1}{2}\begin{bmatrix} 2 & -1 & -1 \\ 0 & 1 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ -1 & -1 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}.$$

PCA - Example

- As a final point, note that we can **also recover the attributes from P by a linear sum**. In other words, we can recover the attributes as follows:

$$PU^T = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

PCA - Example

- **Principal component analysis using geometry**
- Recall that we initially defined the matrix a in terms of two column vectors, which we called attributes, and which were written as

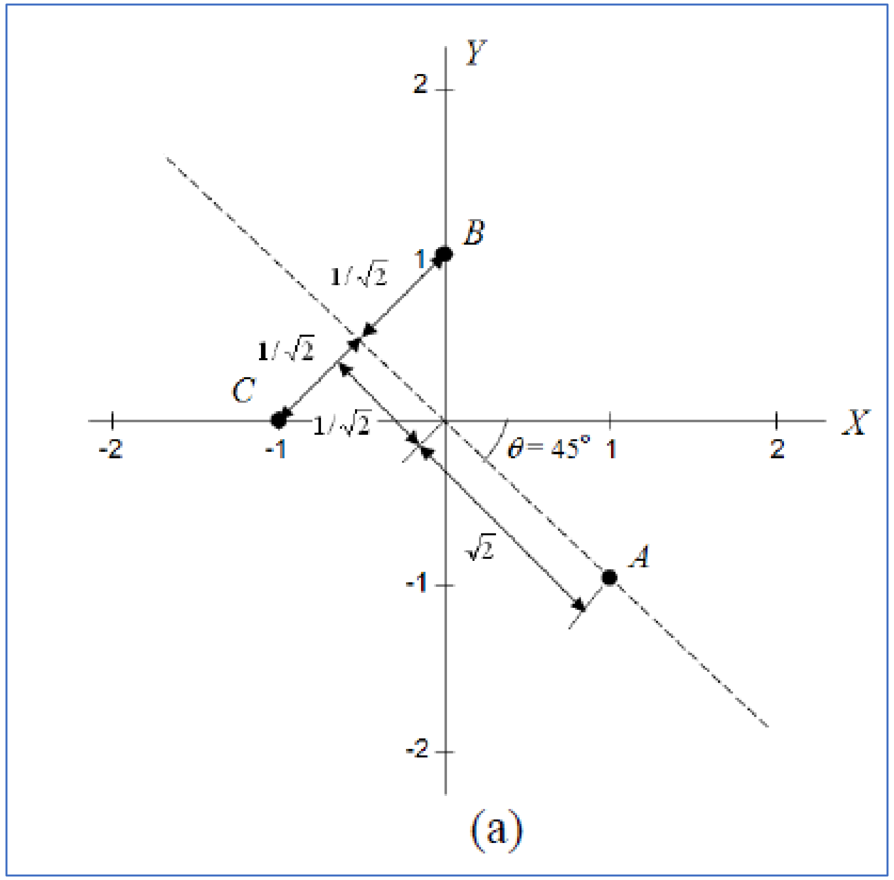
$$S = [X \quad Y] = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ -1 & 0 \end{bmatrix}$$

- An alternate interpretation of the matrix S is as three row vectors, which are two-dimensional points in attribute space.

$$A = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \text{and } C = \begin{bmatrix} -1 \\ 0 \end{bmatrix}.$$

PCA - Example

- **Principal component analysis using geometry**
- These three points are plotted in Figure Below. Also shown in this figure is a dashed line drawn at -45^o to the vertical axis, which goes through point 1 and bisects the other two points.
-



(a) shows the points in our example before PCA

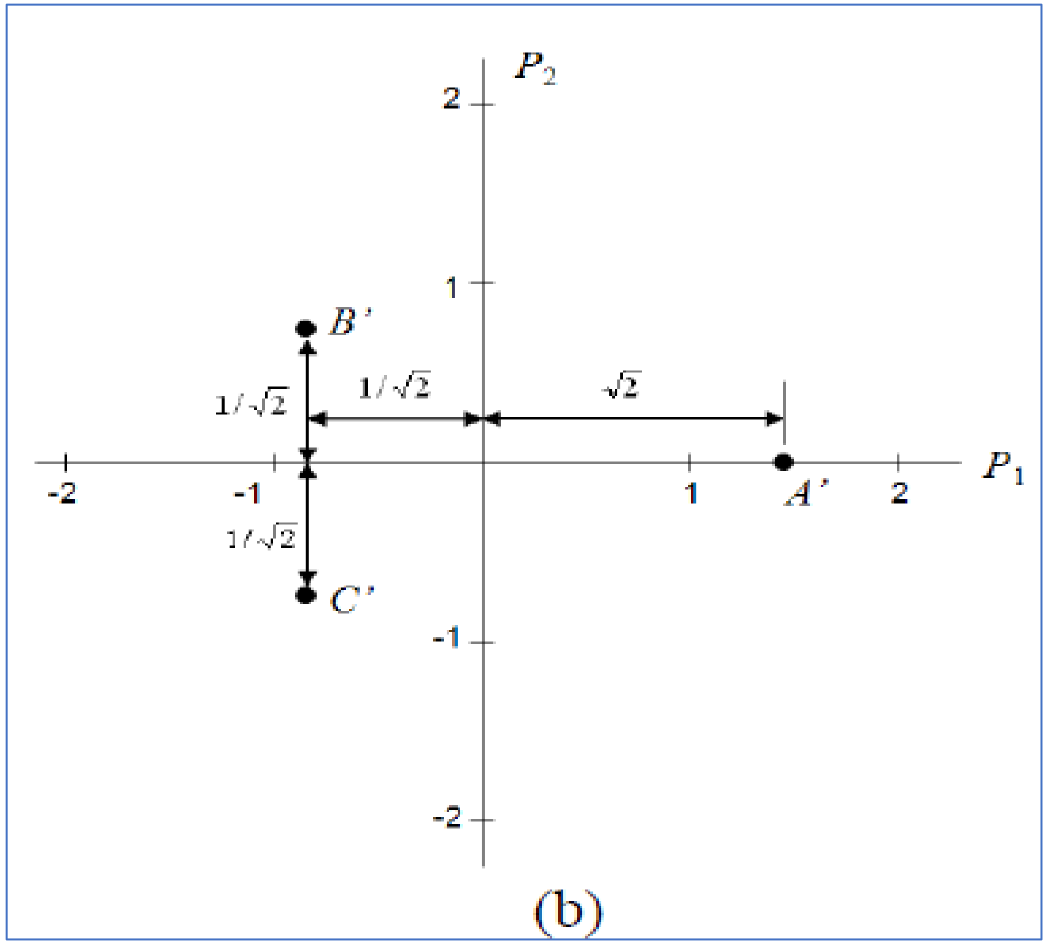
PCA - Example

- **Principal component analysis using geometry**
- Similarly, we can interpret the matrix P is as three row vectors, which are two-dimensional points in attribute space.

- $$P = \frac{1}{\sqrt{2}}\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ -1 & -1 \end{bmatrix}.$$

where $A' = \begin{bmatrix} \sqrt{2} \\ 0 \end{bmatrix}$, $B' = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 \\ 1 \end{bmatrix}$, and $C' = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 \\ -1 \end{bmatrix}$.

(b) shows the points after PCA.



PCA - Example

Note

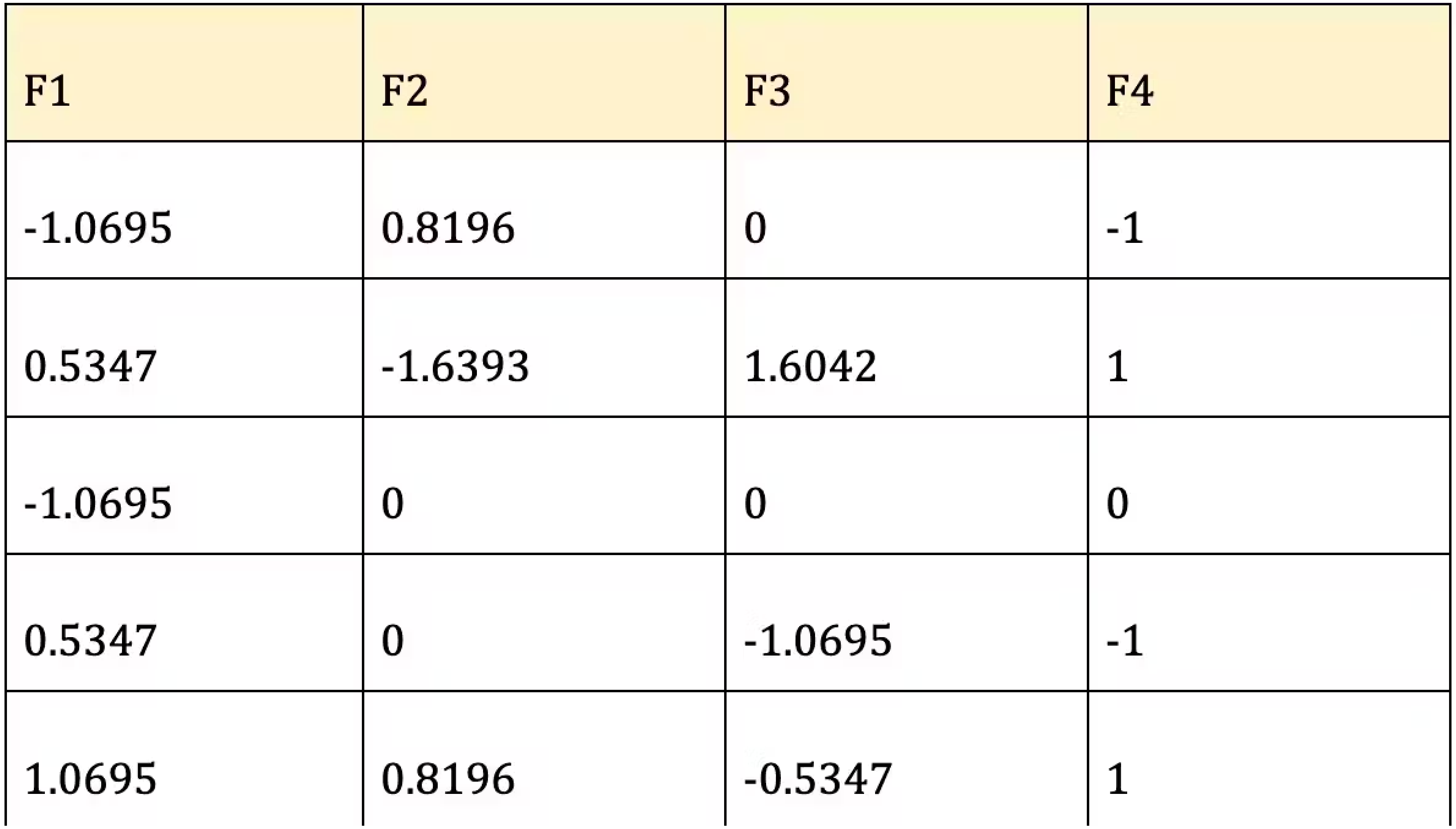
- Note that the transform involves a rotation of the axes(-45^o)
- Note that this transform has preserved the same distance relationships

PCA - Example2

LARGE SIZE APPLES	ROTTEN APPLES	DAMAGED APPLES	SMALL APPLES
F1	F2	F3	F4
1	5	3	1
4	2	6	3
1	4	3	2
4	4	1	1
5	5	2	3

PCA - Example2

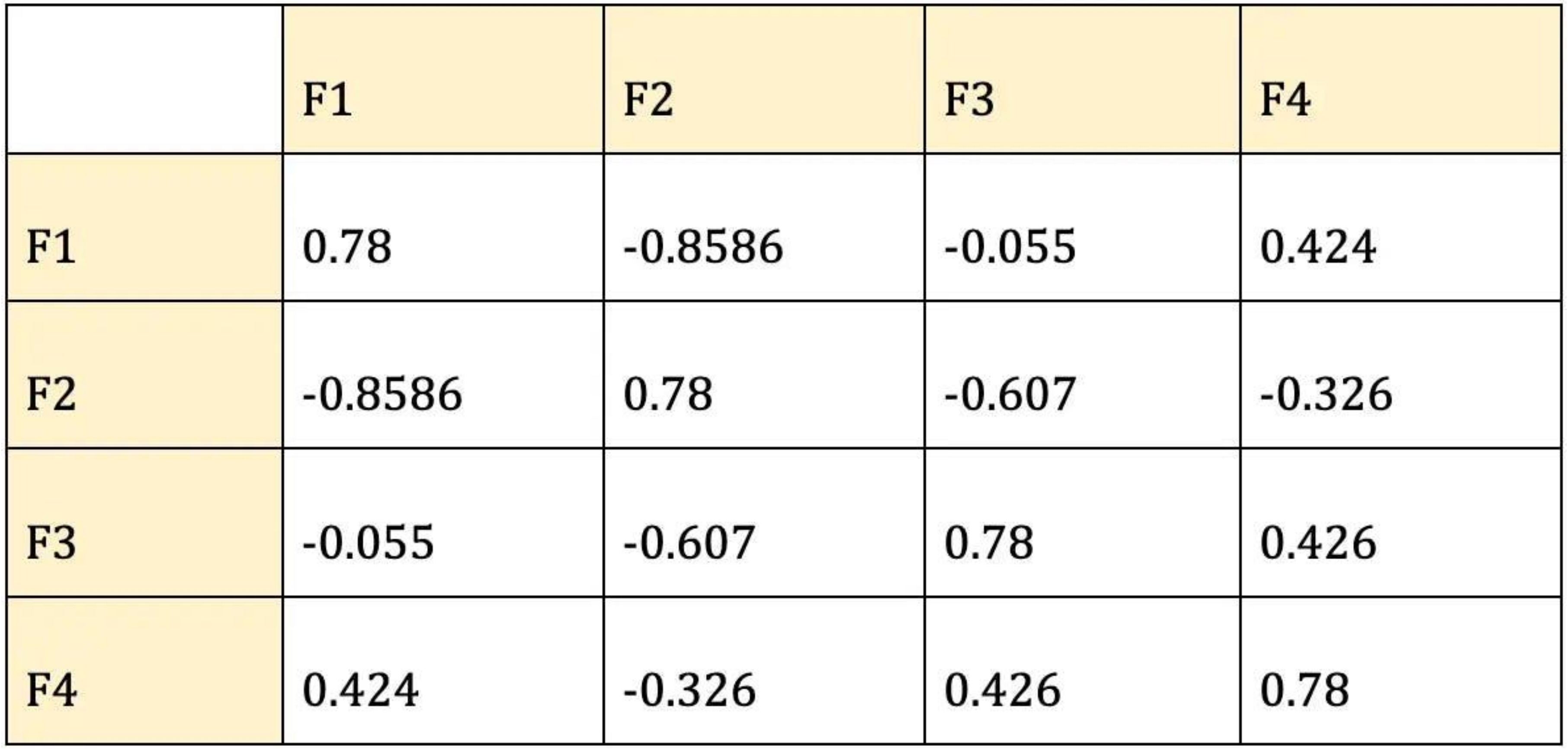
Then, after the Standardization of each variable, the results are tabulated below.



F1	F2	F3	F4
-1.0695	0.8196	0	-1
0.5347	-1.6393	1.6042	1
-1.0695	0	0	0
0.5347	0	-1.0695	-1
1.0695	0.8196	-0.5347	1

PCA - Example2

COVARIANCE MATRIX COMPUTATION



	F1	F2	F3	F4
F1	0.78	-0.8586	-0.055	0.424
F2	-0.8586	0.78	-0.607	-0.326
F3	-0.055	-0.607	0.78	0.426
F4	0.424	-0.326	0.426	0.78

PCA - Example2

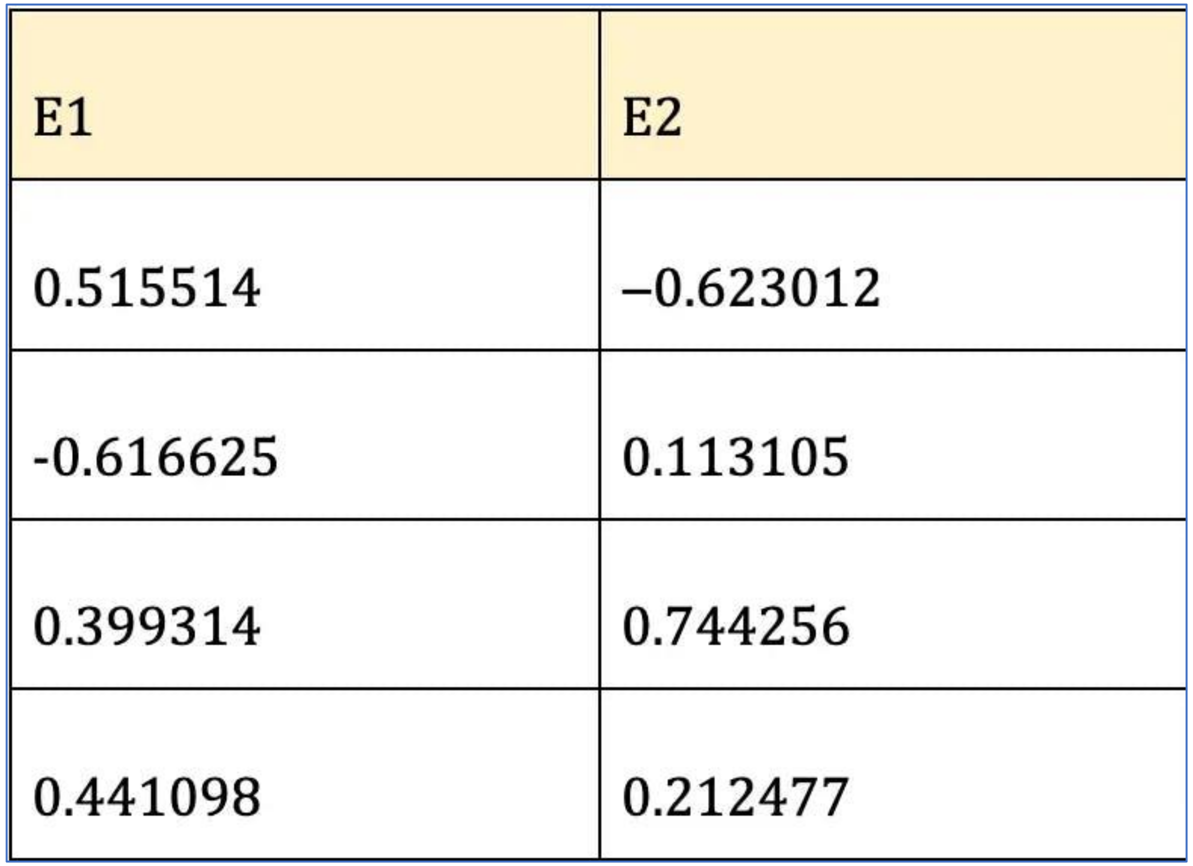
Find eigen value and eigen vectors

λ = 2.11691 , 0.855413 , 0.481689 , 0.334007

E1	E2	E3	E4
0.515514	–0.623012	0.0349815	–0.587262
-0.616625	0.113105	0.452326	–0.634336
0.399314	0.744256	–0.280906	–0.455767
0.441098	0.212477	0.845736	0.212173

PCA - Example2

Select feature vectors



E1	E2
0.515514	–0.623012
-0.616625	0.113105
0.399314	0.744256
0.441098	0.212477

PCA - Example2

RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES

Standardized Original Data Set

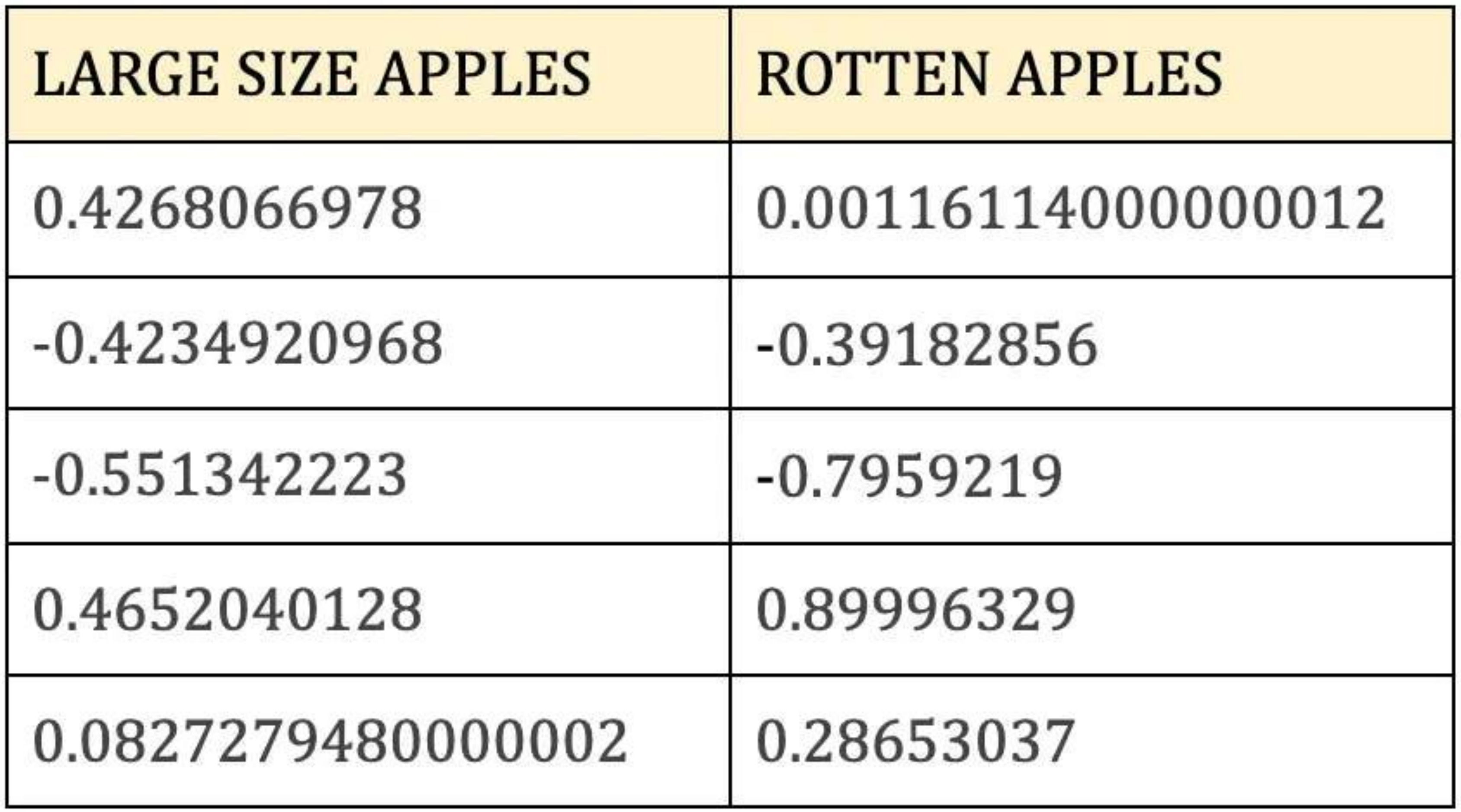
F1	F2	F3	F4
-1.0695	0.8196	0	-1
0.5347	-1.6393	1.6042	1
-1.0695	0	0	0
0.5347	0	-1.0695	-1
1.0695	0.8196	-0.5347	1

Feature Vector

E1	E2
0.515514	0.744256
0.441098	0.212477
0.399314	0.113105
-0.616625	–0.623012

PCA - Example2

RECAST THE DATA ALONG THE PRINCIPAL COMPONENTS AXES



LARGE SIZE APPLES	ROTTEN APPLES
0.4268066978	0.0011611400000012
-0.4234920968	-0.39182856
-0.551342223	-0.7959219
0.4652040128	0.89996329
0.0827279480000002	0.28653037

Applications of PCA Analysis

- PCA in machine learning is used to visualize multidimensional data.
- In healthcare data to explore the factors that are assumed to be very important in increasing the risk of any chronic disease.
- PCA helps to resize an image.
- PCA is used to analyze stock data and forecasting data.
- You can also use Principal Component Analysis to analyze patterns when you are dealing with high-dimensional data sets.

Advantages of PCA Analysis

1. By reducing the data to two dimensions, you can easily visualize it.
2. PCA removes multicollinearity. Multicollinearity arises when two features are correlated. PCA produces a set of new orthogonal axes to represent the data, which, as the name suggests, are uncorrelated.
3. PCA removes noise. By reducing the number of dimensions in the data, PCA can help remove noisy and irrelevant features.
4. PCA reduces model parameters: PCA can help reduce the number of parameters in machine learning models.
5. PCA reduces model training time. By reducing the number of dimensions, PCA simplifies the calculations involved in a model, leading to faster training times.

Disdvantages of PCA Analysis

1. The run-time of PCA is cubic in relation to the number of dimensions of the data. This can be computationally expensive at times for large datasets.
2. PCA transforms the original input variables into new principal components (or dimensions). The new dimensions offer no interpretability.
3. While PCA simplifies the data and removes noise, it always leads to some loss of information when we reduce dimensions.
4. PCA is a linear dimensionality reduction technique, but not all real-world datasets may be linear.
5. PCA gets affected by outliers. This can distort the principal components and affect the accuracy of the results.