COMP809 Data Mining and Machine Learning

Patricio Maturana-Russel
p.maturana.russel@aut.ac.nz

Department of Mathematical Sciences and Computer Science and Software Engineering Departments, Auckland University of Technology, Auckland, New Zealand

Semester 1, 2024

Contents

- Unsupervised and supervised machine learning techniques.
- Clustering analysis.
- K-means.
- Case study.

Machine learning

Machine learning techniques are usually divided in 2 categories:

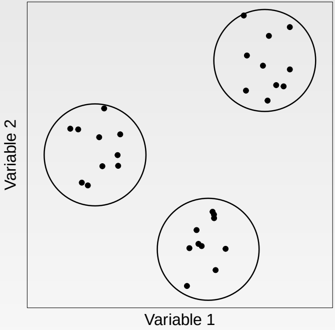
- **Unsupervised**: no labels are given to the learning algorithm.
- **Supervised**: labels are given to the learning algorithm.

Cluster analysis

The idea of clustering analysis is to find groups of objects such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups.

For this:

- intra-cluster distances are minimized.
- inter-cluster distances are maximized.



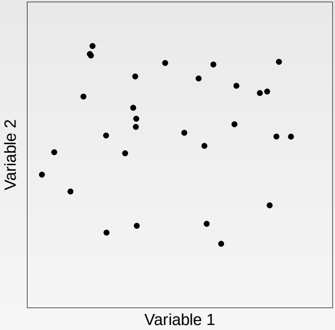
Cluster analysis

The idea of clustering analysis is to find groups of objects such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups.

For this:

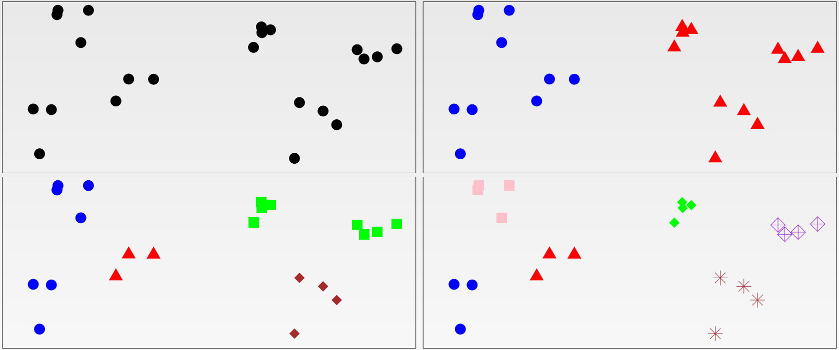
- intra-cluster distances are minimized.
- inter-cluster distances are maximized.

However, reality can be a bit more complex.



Cluster analysis

The notion of cluster can be ambiguous:



How many cluster would you consider?

Clustering considerations

- What does it mean for objects to be similar?
- What algorithm and approach do we take?
  - Top-down: k-means
  - Bottom-up: hierarchical agglomerative clustering
- Do we need a hierarchical arrangement of clusters?
- How many clusters?
- Can we label or name the clusters?
- How do we make it efficient and scalable?

Clustering considerations

What makes docs "related"?

- Ideal: semantic similarity.
- Practical: statistical similarity.
  - Treat documents as vectors.
  - For many algorithms, easier to think in terms of a distance (rather than similarity) between docs.
  - Think of either cosine similarity or Euclidean distance.

Clustering algorithms

Partitional algorithms

- Usually start with a random (partial) partitioning
- Refine it iteratively
  - K means clustering
  - Model based clustering

Hierarchical algorithms

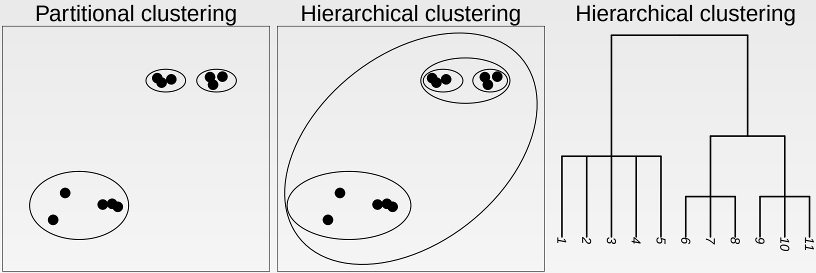
- Bottom-up, agglomerative
- Top-down, divisive

Clustering algorithms

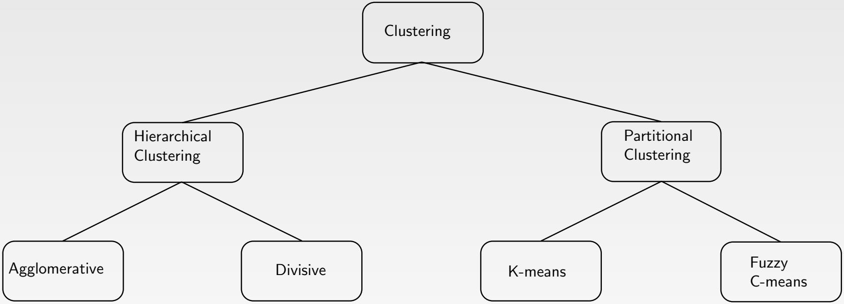
Differences:

- **Partitional clustering**: A division data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset.
- **Hierarchical clustering**: A set of nested clusters organized as a hierarchical tree.

Clustering algorithms



Types of clustering



K-means

It is a partitional clustering approach.

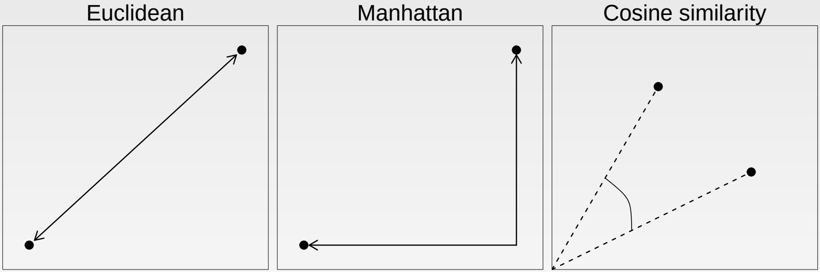
- Number of clusters, K, must be specified.
- Each cluster is associated with a centroid (center point).
- Each point is assigned to the cluster with the closest centroid.
- The basic algorithm is very simple.

Algorithm:

1. Select K points as the initial centroids.
2. **Repeat**
3.   Form K clusters by assigning all points to the closest centroid.
4.   Recompute the centroid of each cluster.
5. **Until** the centroids do not change.

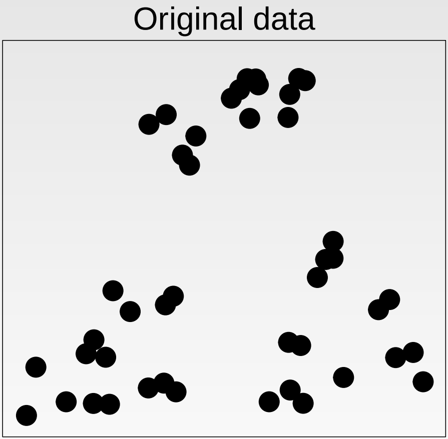
K-means

Distances:



Distance measure determines the similarity between the observations and influence the shape of the clusters.

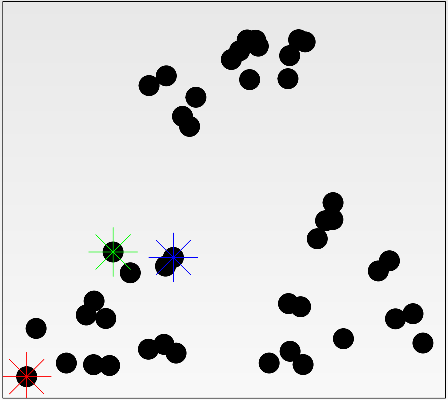
K-means



Number of clusters: K=3; distance: Euclidean.

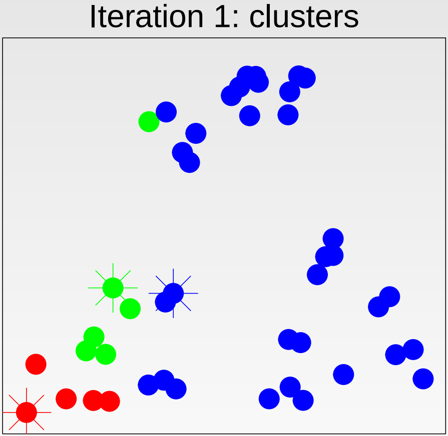
K-means

Iteration 1: Random centres



Number of clusters: K=3; distance: Euclidean.

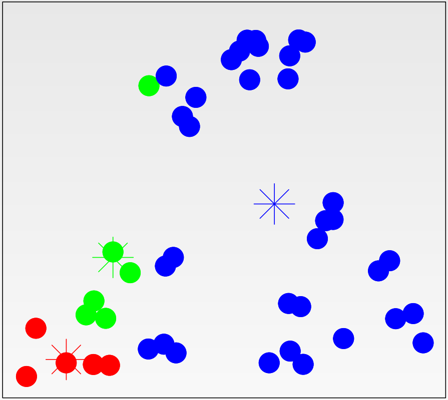
K-means



Number of clusters: K=3; distance: Euclidean.

K-means

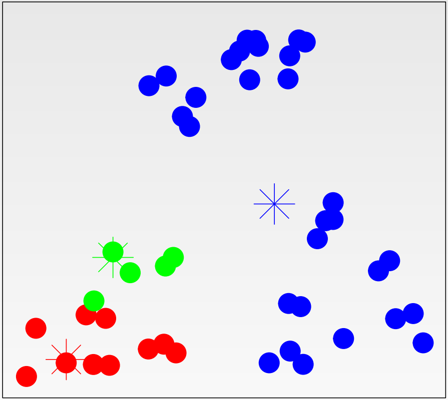
Iteration 2: Readjusting centres



Number of clusters: K=3; distance: Euclidean.

K-means

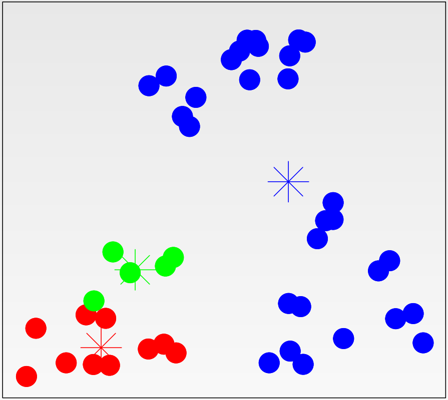
Iteration 2: clusters



Number of clusters: K=3; distance: Euclidean.

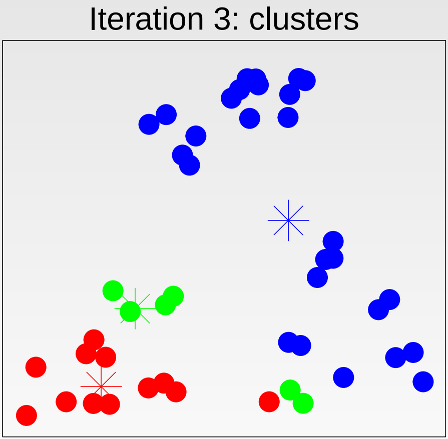
K-means

Iteration 3: Readjusting centres



Number of clusters: K=3; distance: Euclidean.

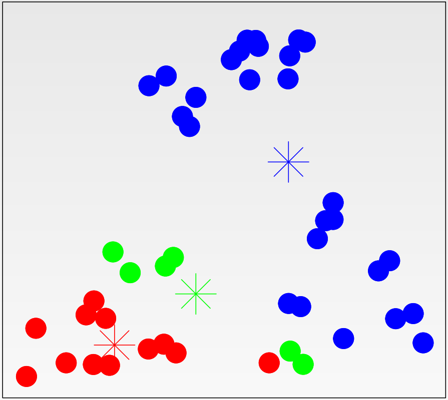
K-means



Number of clusters: K=3; distance: Euclidean.

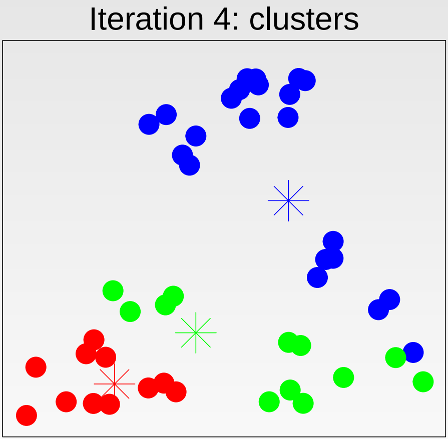
K-means

Iteration 4: Readjusting centres



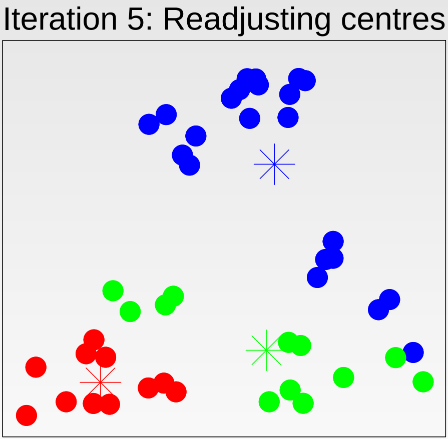
Number of clusters: K=3; distance: Euclidean.

K-means



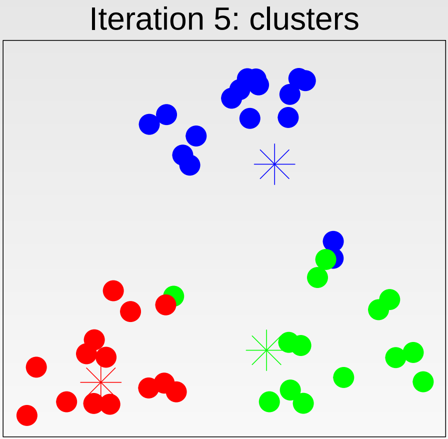
Number of clusters: K=3; distance: Euclidean.

K-means



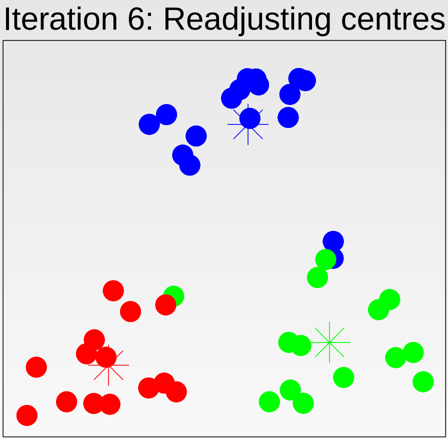
Number of clusters: K=3; distance: Euclidean.

K-means



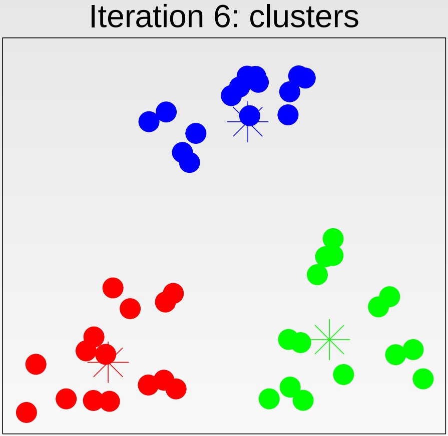
Number of clusters: K=3; distance: Euclidean.

K-means



Number of clusters: K=3; distance: Euclidean.

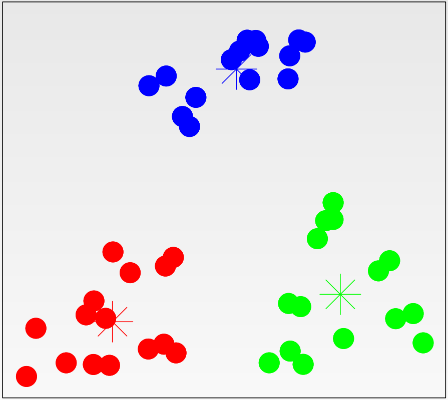
K-means



Number of clusters: K=3; distance: Euclidean.

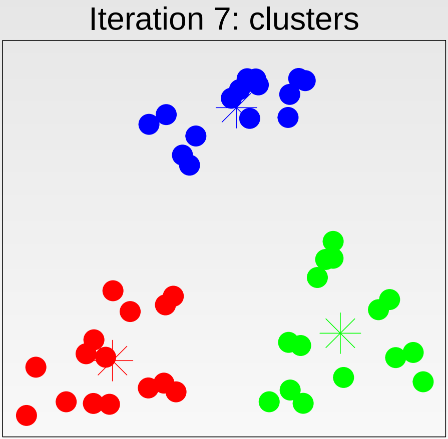
K-means

Iteration 7: Readjusting centres



Number of clusters: K=3; distance: Euclidean.

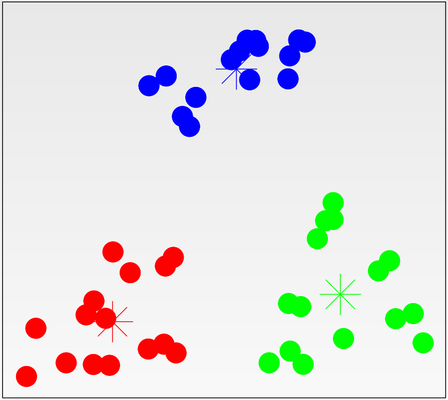
K-means



Number of clusters: K=3; distance: Euclidean.

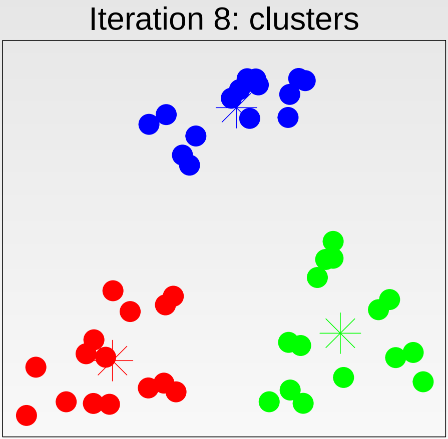
K-means

Iteration 8: Readjusting centres



Number of clusters: K=3; distance: Euclidean.

K-means



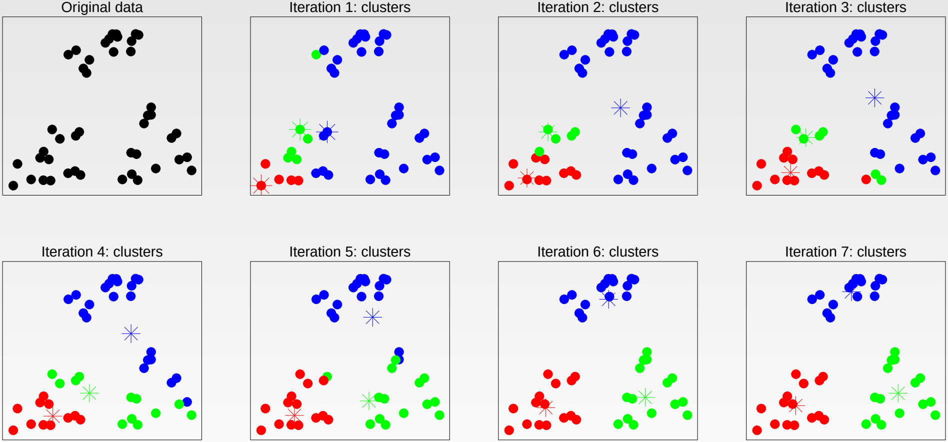
Number of clusters: K=3; distance: Euclidean.

K-means

Remarks:

- Initial centroids are often chosen randomly.
    - Clusters produced might vary from one run to another.
- The centroid is (typically) the mean of the points in the cluster.
- *Closeness* can be measured by Euclidean distance, cosine similarity, correlation, etc.
- Most of the convergence happens in the first few iterations.
    - Often the stopping condition is changed to "Until relatively few points change clusters".
- Most of the variants of the k-means which differ in
    - Selection of the initial k-means.
    - Dissimilarity calculations.
    - Strategies to calculate cluster means.
- Initial centroids have a crucial importance in the performance of the method (see next slide).

K-means


Original data
Iteration 1: clusters
Iteration 2: clusters
Iteration 3: clusters
Iteration 4: clusters
Iteration 5: clusters
Iteration 6: clusters
Iteration 7: clusters

K-means

Advantages:

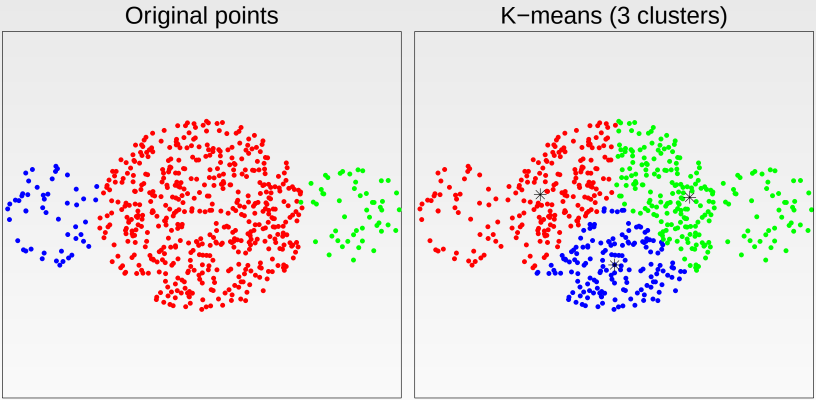
- Very fast: it is linear both in terms of number of samples as well as number of data dimensions.

Disadvantages:

- K must be supplied. However, K can be tuned, e.g., using GridSearchCV.
- Inability to deal with non-spherical clusters.
- It has problems when data contains outliers.

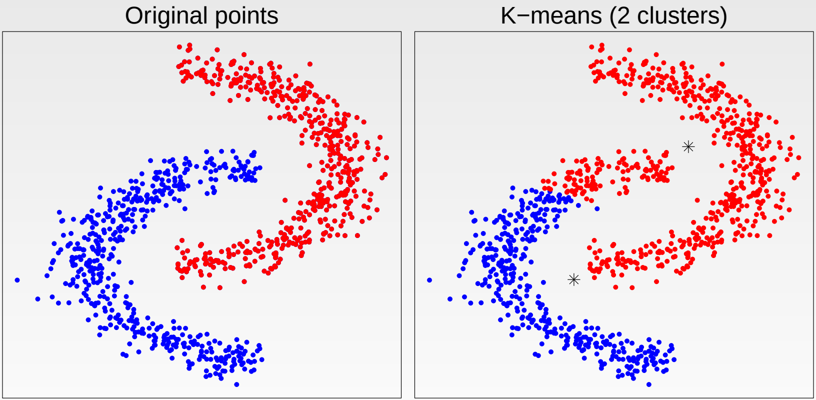
K-means

Different sizes.



K-means

Non-globular shapes.



Determining K

There are multiple methods to determine the number of clusters for the K-means method. Here, we will discuss two methods, which are based on:

- the sum of squares error (Elbow method), and
- the silhouette score.

Determining K

Elbow method

We calculate WCSS (within-Cluster Sum of Square) for a series of K values. WCSS is the sum of squared distance between each point and the centroid in a cluster. As the number of clusters increases, the WCSS value will start to decrease.

The optimal value is in the so called "elbow" point, where for higher values the WCSS does not change much.

Determining K

Silhouette score

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette ranges from -1 to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters. Source: Wikipedia

Case study

The dataset has been compiled from the United Nations Demographic Yearbook 1990 (United Nations publications) and has the following variables: *birth rate*, *death rate*, *infant death rate*, and *country*.

Can these variables be used to categorize these countries?

```
>>> import pandas as pd
>>> data = pd.read_csv("poverty.csv");
>>> print(data);
    Birth  Death  InfantDeath         Country
0    24.7    5.7         30.8         Albania
1    13.4   11.7         11.3  Czechoslovakia
2    11.6   13.4         14.8         Hungary
3    13.6   10.7         26.9         Romania
4    17.7   10.0         23.0            USSR
..    ...    ...          ...             ...
92   50.1   20.2        132.0         Somalia
93   44.6   15.8        108.0           Sudan
94   31.1    7.3         52.0         Tunisia
95   50.5   14.0        106.0        Tanzania
96   51.1   13.7         80.0          Zambia
```

Case study

```
>>> data[["Birth", "Death", "InfantDeath"]].describe();
            Birth       Death  InfantDeath
count   97.000000   97.000000    97.000000
mean    29.229897   10.836082    54.901031
std     13.546695    4.647495    45.992584
min      9.700000    2.200000     4.500000
25%     14.500000    7.800000    13.100000
50%     29.000000    9.500000    43.000000
75%     42.200000   12.500000    83.000000
max     52.200000   25.000000   181.600000
```

The SDs are quite different. The data will be standardized.

```
>>> from sklearn.preprocessing import StandardScaler
>>> X      = data.iloc[:,[0,1,2]];
>>> scaler = StandardScaler(); # creating object
>>> fitted = scaler.fit(X);
>>> X_std  = pd.DataFrame(fitted.transform(X));
```

Case study

Elbow method.

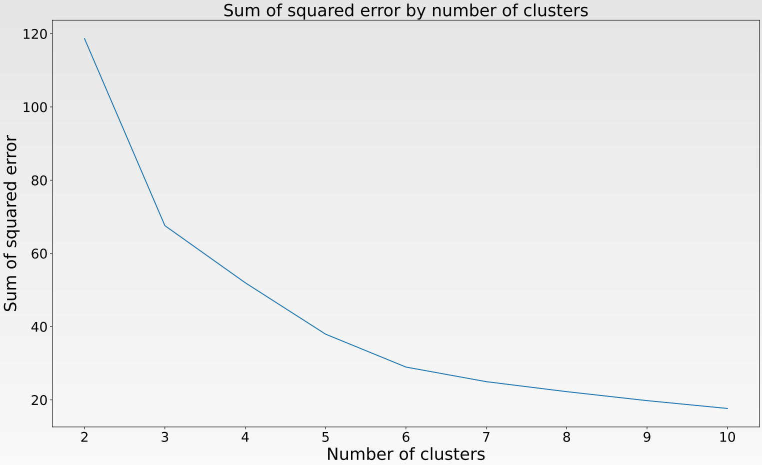
```
>>> from sklearn.cluster import KMeans
>>> def wcss(x, kmax):
        wcss_s = []
        for k in range(2, kmax + 1):
            kmeans = KMeans(n_clusters = k);
            kmeans.fit(x);
            wcss_s.append(kmeans.inertia_);# sample distances to closest cluster center
        return wcss_s

# Plot
>>> from matplotlib import pyplot as plt
>>> fig  = plt.figure(figsize = (19,11));
>>> ax   = fig.add_subplot(1,1,1);
>>> kmax = 10; # maximum number of clusters
>>> ax.plot(range(2, kmax + 1), wcss(X_std, kmax));
>>> ax.tick_params(axis="both", which="major", labelsize=20);
>>> ax.set_xlabel("Number of clusters", fontsize = 25);
>>> ax.set_ylabel("Sum of squared error", fontsize = 25);
>>> ax.set_title("Sum of squared error by number of clusters", fontsize = 25);
>>> plt.show();
```

Case study

Silhouette score.

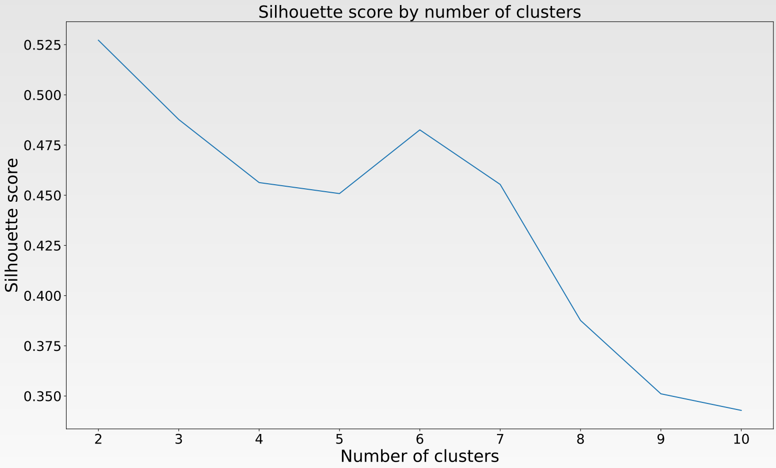
```
>>> from sklearn.metrics import silhouette_score
>>> def Silhouette(x, kmax):
        sil = []
        for k in range(2, kmax+1):
            kmeans = KMeans(n_clusters = k).fit(x)
            sil.append(silhouette_score(x, kmeans.labels_, metric = "euclidean"))
        return sil

# Plot
>>> fig = plt.figure(figsize = (19,11));
>>> ax = fig.add_subplot(1,1,1);
>>> ax.plot(range(2,kmax+1), Silhouette(X_std,kmax));
>>> ax.tick_params(axis="both", which="major", labelsize=20);

>>> ax.set_xlabel("Number of clusters", fontsize = 25);
>>> ax.set_ylabel("Silhouette score", fontsize = 25);
>>> ax.set_title("Silhouette score by number of clusters", fontsize = 25);
>>> plt.show();
```

Case study



The elbow point is determined visually. Here, it could be at K= 3 or 5.

Case study



The maximum value is reached at K=2, followed by 3 and 6.

Case study

The silhouette score favors K=2 or 3 (or 6). However, K=2 has the highest sum of squared error. Therefore, we will explore K=3.

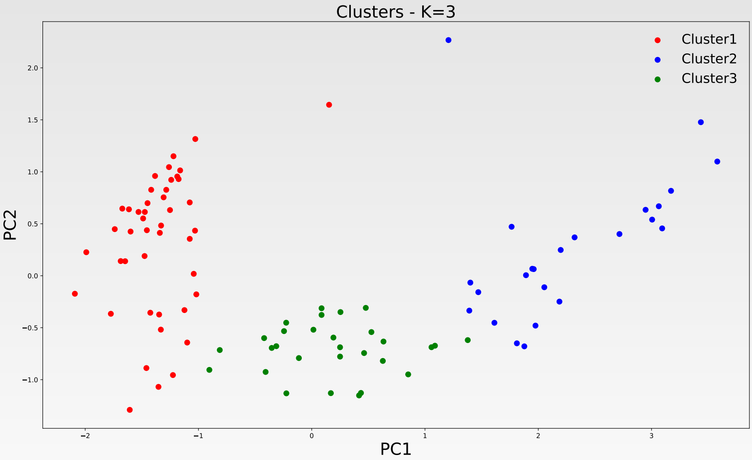
Visual inspection in the multivariate case is difficult or impossible. So, we will reduce the dimensionality of the data via the principal component method.

Case study

```
>>> from sklearn.decomposition import PCA
>>> pca = PCA(n_components=2);
>>> principalComponents = pca.fit_transform(X_std);
>>> np.sum(pca.explained_variance_ratio_);
    0.9620015356918279
>>> PCs = pd.DataFrame(data = principalComponents, columns = ["PC1", "PC2"]);

>>> kmeans   = KMeans(n_clusters = 3, init = "k-means++", random_state = 42);
>>> y_kmeans = kmeans.fit_predict(X_std);

# Plotting PCs
>>> fig = plt.figure(figsize = (19,11));
>>> ax  = fig.add_subplot(1,1,1);
>>> plt.scatter(PCs.iloc[y_kmeans == 0, 0], PCs.iloc[y_kmeans == 0, 1], s=60,
                c="red", label = "Cluster1");
>>> plt.scatter(PCs.iloc[y_kmeans == 1, 0], PCs.iloc[y_kmeans == 1, 1], s=60,
                c="blue", label = "Cluster2");
>>> plt.scatter(PCs.iloc[y_kmeans == 2, 0], PCs.iloc[y_kmeans == 2, 1], s=60,
                c="green", label = "Cluster3");
>>> plt.xlabel("PC1", fontsize = 25);
>>> plt.ylabel("PC2", fontsize = 25);
>>> ax.set_title("Clusters - K=3", fontsize = 25);
>>> plt.legend(fontsize = 20);
>>> plt.show();
```

Note that the first 2 PCs explain 96% of the variability of the variables.

Case study



There is a clear separation between the clusters, they do not overlap. It seems that K=3 represents quite well the cluster structure of the data.

Prediction

Given a new observation, we can estimate to which cluster belong to. The new observation is associated to its closest cluster center.

Let classify the following countries:

	Birth	Death	InfantDeath
Country A	10	3	5
Country B	29	11	55
Country C	52	25	180

Prediction

```
>>> new_data = pd.DataFrame([[10,3,5], [29, 11, 55], [52, 25, 180]],
                            columns=["Birth", "Death", "InfantDeath"]);
>>> new_data_std = pd.DataFrame(fitted.transform(new_data));
>>> print(kmeans.predict(new_data_std) + 1); # clusters 0, 1, 2 (+1 correction)
    [1 3 2]
```

Countries A, B and C are classified into clusters 1, 3, and 2, respectively.

Note that A has values close to the minimum values, B close to the mean values, and C close to the maximum values. This can potentially help to understand the clusters in this particular case.

Prediction

```
>>> data_clusters = pd.concat([data["Country"],
                               pd.DataFrame(y_kmeans, columns = ["Cluster"])], axis=1);

>>> print("Cluster 1:\n", list(data_clusters["Country"][(data_clusters['Cluster']==0)]));
    ['Albania', 'Czechoslovakia', 'Hungary', 'Romania', 'USSR', 'Ukrainian_SSR', 'Chile',
     'Uruguay', 'Finland', 'France', 'Greece', 'Italy', 'Norway', 'Spain', 'Switzerland',
     'Austria', 'Canada', 'Israel', 'Kuwait', 'China', 'Korea', 'Singapore', 'Thailand',
     'Bulgaria', 'Former_E._Germany', 'Poland', 'Yugoslavia', 'Byelorussia_SSR',
     'Argentina', 'Venezuela', 'Belgium', 'Denmark', 'Germany', 'Ireland', 'Netherlands',
     'Hong_Kong', 'Sri_Lanka']

>>> print("Cluster 2:\n", list(data_clusters["Country"][(data_clusters['Cluster']==1)]));
    ['Bolivia', 'Mexico', 'Afghanistan', 'Bangladesh', 'Gabon', 'Ghana', 'Namibia',
     'Sierra_Leone', 'Swaziland', 'Uganda', 'Zaire', 'Cambodia', 'Nepal', 'Angola',
     'Congo', 'Ethiopia', 'Gambia', 'Malawi', 'Mozambique', 'Nigeria', 'Somalia', 'Sudan',
     'Tanzania', 'Zambia']

>>> print("Cluster 3:\n", list(data_clusters["Country"][(data_clusters['Cluster']==2)]));
    ['Ecuador', 'Paraguay', 'Iran', 'Oman', 'Turkey', 'India', 'Mongolia', 'Pakistan',
     'Algeria', 'Botswana', 'Egypt', 'Libya', 'Morocco', 'South_Africa', 'Zimbabwe',
     'Brazil', 'Columbia', 'Guyana', 'Peru', 'Iraq', 'Jordan', 'Lebanon', 'Saudi_Arabia',
     'Indonesia', 'Malaysia', 'Philippines', 'Vietnam', 'Kenya', 'Tunisia']
```

End