Actuaries Institute.

Fellowship Program

Data Science Applications

Chapter 6: Unsupervised Learning

Table of Contents

6. Unsupervised Learning

This chapter covers the following learning objectives:

Item	Learning Objectives
2.	**Develop data science solutions to business problems using a variety of tools and techniques**
2.2.	**Develop solutions to problems using unsupervised learning**
2.2.1.	Explain the purpose of dimension reduction
2.2.2.	Implement principal component analysis
2.2.3.	Recognise alternative dimension reduction techniques
2.2.4.	Explain how clustering techniques can be used to gain business insight
2.2.5.	Contrast k-means and hierarchical clustering techniques
2.2.6.	Perform k-means and hierarchical clustering
2.2.7.	Discuss alternative clustering techniques
2.2.8.	Evaluate a clustering algorithm using internal, external, and manual validation

6.1. Introduction

Unsupervised learning is a data science technique that looks for patterns in data. Unlike supervised learning, for which responses are available in the training data and used in training a model, unsupervised learning does not rely on the existence of responses to supervise the learning.

Supervised and unsupervised learning tasks attempt to answer fundamentally different questions. Supervised learning is about making predictions of an outcome using data features. More formally, supervised learning involves estimating the properties of $P(Y|X)$, where Y represents a dataset's responses (the outcome variable) and X represents the data's features. In contrast, unsupervised learning is about understanding the properties of the distribution of data features, $P(X)$. Therefore, estimation of the joint probability density $P(X,Y)$, which can be re-expressed as $P(X,Y) = P(Y|X) \times P(X)$, involves both supervised learning, which gives information about $P(Y|X)$, and unsupervised learning, which gives information about $P(X)$.

For example, a supervised learning task might involve using the characteristics (features) of a supermarket's customers, together with past observations of how much each customer spent (response) to predict how much future customers might spend (prediction). In contrast, an unsupervised learning task might not have access to information about how much customers spent in the past. The unsupervised learning task might, instead, use the features in the data to group customers into homogenous 'clusters', with the aim of providing insights for the marketing team.

Therefore, although unsupervised learning does not rely on responses, the training data still provides features that unsupervised learning can draw upon. As another example, imagine that you walk into a greengrocer and notice a fruit you have not seen before. While you do not know the name of the fruit (the 'response'), you can use information ('features') such as its size, shape, smell, colour and other fruits it has been placed near to guess which fruits it is likely to be similar to.

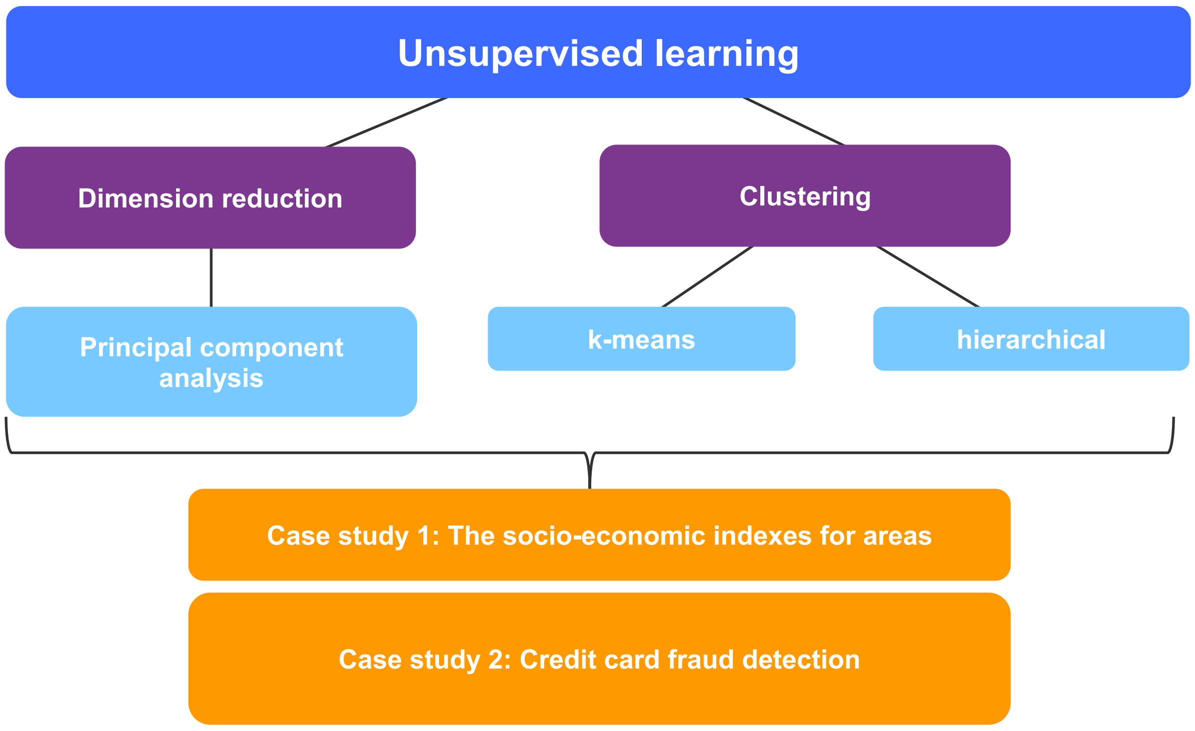
Further examples of unsupervised learning include:

- converting a high-dimensional dataset into one with a smaller number of features, without losing important information;
- identifying groups of car models with similar technical attributes to efficiently estimate risk premiums for individual car models;[1] and
- finding unusual credit card transactions to identify potential cases of fraud.

[1] Campbell, M. (1986). An integrated system for estimating the risk premium of individual car models in motor insurance. *ASTIN Bulletin, 16(2)*, 165–183.

This chapter covers two branches of unsupervised learning—dimension reduction and clustering—as shown in Figure 6.1.

Figure 6.1 – Chapter 6 mind map


Unsupervised learning
Dimension reduction
Clustering
Principal component analysis
k-means
hierarchical
Case study 1: The socio-economic indexes for areas
Case study 2: Credit card fraud detection

The first example of unsupervised learning provided in the bullet point above Figure 6.1 is an example of dimension reduction. The second and third bullet points are examples of clustering.

This chapter will start with a recap of the dimension reduction technique known as principal component analysis (PCA). PCA was introduced in the Foundation program in Actuarial Statistics (CS1). It is also taught in the Actuary program subject Data Science Principles. Alternative dimension reduction techniques will also be outlined.

The chapter will then introduce the k-means and hierarchical clustering techniques, along with several alternative clustering algorithms.

The chapter will conclude with two case studies to demonstrate how unsupervised learning can be applied in different domains.

6.1.1. Example Dataset

The following dataset is used in examples provided throughout this chapter.

COVID dataset

The COVID dataset is sourced from Our World in Data.[2] It contains country-by-country data on confirmed coronavirus disease (COVID-19) cases as at 31 March 2021. The data contains 31 COVID-19-related features for over 100 countries. These features include:

- total cases per million people;
- total new cases per million people;
- total deaths per million people;
- new deaths per million people;
- reproduction rate of the disease;
- positive testing rate;
- total tests per thousand people;
- ICU patients per million people; and
- hospital patients per million people.

[2] Ritchie, H., Ortiz-Ospina, E., Beltekian, D., Mathieu, E., Hasell, J., Macdonald, B., Giattino, C., and Roser, M. (2020). *Our World in Data COVID-19 dataset* [Dataset]. https://ourworldindata.org/covid-cases, sourced on 31 March 2021.

6.2. Dimension Reduction

Dimension reduction refers to unsupervised learning techniques that aim to represent a high-dimensional dataset with a reduced number of dimensions. A high-dimensional dataset refers to one with a large number of features. Examples of high-dimensional datasets include:

- health data recording the health status of individuals, which might include many different features such as recorded parameters from blood analysis, immune system status, genetic background, nutrition, alcohol, tobacco and drug consumption, past operations, treatments, and diagnosed diseases;
- image data, which might capture the amount of red, green and blue in each pixel of an image, where the pixel count might be in the 1,000s; and
- earth observation data, which might capture features such as longitude, latitude, temperature, pressure, rainfall, humidity, altitude, time, season, and soil type.

While high-dimensional datasets can provide a rich source of information, they suffer from the following drawbacks:

- it is difficult to visualise and, therefore, interpret high-dimensional data since the human eye can only really cope with seeing data plotted in three dimensions;
- many of the dataset’s features might be unimportant for the problem to be solved and these redundant features might make the dataset more ‘noisy’ and difficult to interpret;
- a large number of observations is needed to successfully apply most data science techniques as typically the number of observations needs to be well in excess of the number of features; and
- large datasets can slow down computing time.

These drawbacks are referred to as the ‘**curse of dimensionality**’.

Thus, it can be useful to reduce the dimension of the data in many applications to facilitate its visualisation, analysis and interpretation. Dimension reduction techniques help to reduce the dimensionality of a dataset without the loss of too much important information along the way. In other words, dimension reduction techniques find features of the data that explain most of its variability.

Dimension reduction can be as straightforward as removing features that:

- you know are unrelated to the variable of interest (the response) based on your domain knowledge, or based on advice from a domain expert;

- are not significant predictors of the response variable, based on statistical analysis;
- have a high ratio of missing values, although care needs to be taken not to remove features that you know are important, despite the high ratio of missing values;
- are known to be unreliable; or
- are highly correlated with other features that are also in the data, therefore making some features redundant.

The dimension reduction techniques listed above are known as 'feature selection' methods. These techniques are used to filter irrelevant or redundant features from a dataset and involve keeping a subset of the original features of the dataset.

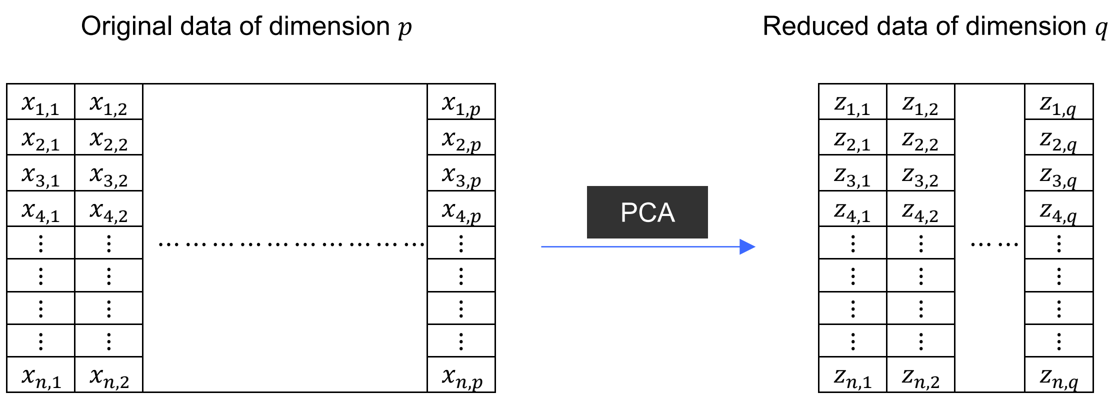
An alternative approach to feature selection is 'feature extraction'. Under feature extraction, rather than keeping a subset of the original features, a new set of features is created. You have already seen an example of feature extraction in Chapter 5 (Classification). Neural networks automatically perform feature extraction within each hidden layer of the network, where new features (neurons) are created from the neurons in the previous layer of the network.

A popular feature extraction method is principal component analysis (PCA). This technique is described in Section 6.2.1 and is contrasted against several other feature extraction methods in Section 6.2.2.

6.2.1. Principal Component Analysis

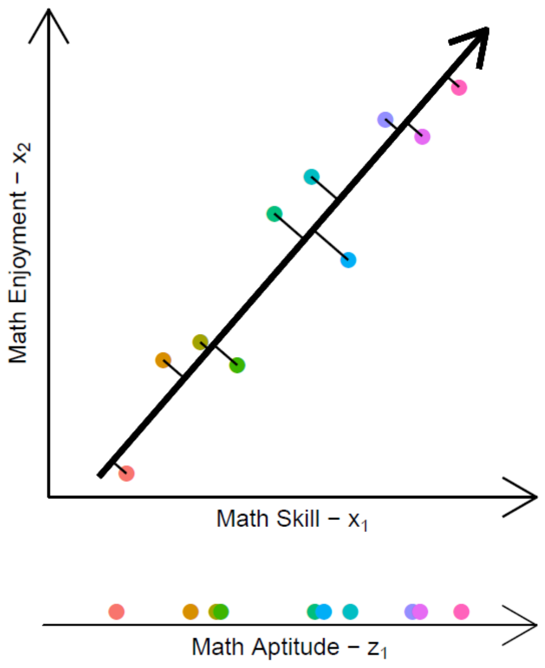
Principal component analysis (PCA) is perhaps the most common dimension reduction technique. PCA takes a dataset with p features—which is said to be p-dimensional—and produces a smaller dataset of dimension $q < p$. The q features or 'components' of the new dataset explain most of the variability in the data and each of the new features is uncorrelated with all other $q - 1$ features. This is illustrated in Figure 6.2.

Figure 6.2 – Dimension reduction


Original data of dimension p
Reduced data of dimension q
$x_{1,1}$ $x_{1,2}$ $x_{1,p}$
$x_{2,1}$ $x_{2,2}$ $x_{2,p}$
$x_{3,1}$ $x_{3,2}$ $x_{3,p}$
$x_{4,1}$ $x_{4,2}$ $x_{4,p}$
$x_{n,1}$ $x_{n,2}$ $x_{n,p}$
PCA
$z_{1,1}$ $z_{1,2}$ $z_{1,q}$
$z_{2,1}$ $z_{2,2}$ $z_{2,q}$
$z_{3,1}$ $z_{3,2}$ $z_{3,q}$
$z_{4,1}$ $z_{4,2}$ $z_{4,q}$
$z_{n,1}$ $z_{n,2}$ $z_{n,q}$

To gain some intuition behind the idea of PCA, assume that you are interviewing potential candidates for a graduate actuarial job. For each candidate, there is a feature x_{i1} that represents their maths skill and a feature x_{i2} that represents their maths enjoyment. You could expect someone who enjoys maths to most likely also be more skilful at maths than someone who does not enjoy it. This is depicted in Figure 6.3.

Figure 6.3 – Illustration of projection of data from 2 dimensions into 1 dimension



Given the high correlation between maths skill and maths enjoyment, it could be useful to summarise these maths-related features into a single feature, $Z_{.1}$, that represents each candidate's 'maths aptitude'. This can be achieved by projecting $X_{.1}$ and $X_{.2}$ onto the line that best explains the variability of the data through the following equation:

Equation 6.1

$$\boldsymbol{Z_{.1} = \phi_1 X_{.1} + \phi_2 X_{.2}}$$

where $Z_{.1}$ is the first principal component and ϕ_1 and ϕ_2 are the 'principal component loadings' that represent how much maths skill ($X_{.1}$) and maths enjoyment ($X_{.2}$) contribute to maths aptitude.

Computing the First Principal Component

Following the notation introduced in Chapter 5 (Classification), consider a dataset with n distinct data points, each with p continuous features. Note that PCA relies on being able to calculate Euclidean distances between features (see Section 6.3.1) so it is unsuited to dealing with categorical features. Section 6.2.2 briefly discusses other dimension reduction techniques that can be used when some of the features are categorical variables.

Let x_{ij} indicate the jth feature for the ith observation, with $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, p$. Let X be the $n \times p$ matrix, where x_{ij} denotes element (i, j), rows represent observations, and columns are features. $X_{.j}$ represents a feature vector (column).

The first principal component, $Z_{.1}$, of a set of standardised features $X_{.1}, X_{.2}, \dots, X_{.p}$ is the linear combination of the standardised features that has the largest variance. As explained in Chapter 5, standardisation is the process of converting all features to have a mean of 0 and a standard deviation of 1. It is important to use standardised features when performing PCA to avoid more emphasis being given to those variables with a higher variance than to those variables with very low variances when identifying each principal component.

The first principal component, $Z_{.1}$, can be expressed mathematically as:

Equation 6.2

$$\boldsymbol{Z_{.1} = \phi_{11} X_{.1} + \phi_{21} X_{.2} + \cdots + \phi_{p1} X_{.p}}$$

and the principal component loadings $\phi_{11}, \phi_{21}, \dots, \phi_{p1}$ of the first principal component can be found by solving the following optimisation problem:

Equation 6.3

$$\boldsymbol{(\phi_{11}, \phi_{21}, \dots, \phi_{p1}) = argmax\, Var(\phi_{11} X_{.1} + \phi_{21} X_{.2} + \cdots + \phi_{p1} X_{.p})}$$

$$\boldsymbol{= argmax \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\}}$$

subject to $\sum_{j=1}^{p} \phi_{j1}^2 = 1$.

The last constraint ($\sum_{j=1}^{p} \phi_{j1}^2 = 1$) is needed, as setting the principal component loadings to an arbitrarily large absolute value would lead to an arbitrarily large variance.

At first glance, Equation 6.2 may appear to be similar to linear regression. However, PCA is attempting to solve a different problem to that of linear regression. When using PCA, there is no response variable, and the aim is to transform the original features into a new set of uncorrelated features (principal components). In contrast, linear regression is a supervised learning technique which aims to find the best-fitting linear relationship between the features and the response variable.

Computing Other Principal Components

Once the first principal component has been found, additional principal components can be calculated. This can be done by finding other linear combinations of the inputs that maximise the variance explained but are also uncorrelated to the prior principal components. For example, the second principal component, $Z_{.2}$, is the linear combination of the standardised features that has the largest variance, subject to it being uncorrelated with $Z_{.1}$. Again, this can be expressed mathematically as:

Equation 6.4 $$\boldsymbol{Z_{.2} = \phi_{12}X_{.1} + \phi_{22}X_{.2} + \cdots + \phi_{p2}X_{.p}}$$

and the principal component loadings $\phi_{12}, \phi_{22}, \ldots, \phi_{p2}$ of the second principal component can be found by solving the following optimisation problem:

Equation 6.5 $$\boldsymbol{(\phi_{12}, \phi_{22}, \ldots, \phi_{p2}) = argmax \left\{\frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{p} \phi_{j2}x_{ij}\right)^2\right\}}$$

subject to $\sum_{j=1}^{p} \phi_{j2}^2 = 1$ and $\sum_{j=1}^{p} \phi_{j1}\phi_{j2} = 0$.

Video 6.1 provides helpful graphical representations to explain the logic behind the calculations of the principal components of a dataset outlined above.

Video 6.1 – Principal component analysis

https://www.youtube.com/watch?v=FgakZw6K1QQ (21 mins)

Record your video notes here

Variance Explained by the Principal Components

For a dataset of dimension p, PCA will return p principal components. As explained above, these p principal components are each linear combinations of the original features, and each principal component is uncorrelated with all other principal components.

Since the aim of PCA is to *reduce* the dimension of the dataset, an important step in using PCA is to select the number of principal components to be used ($k < p$), such that the k features provide a good representation of the original p dimensional data.

To help select an appropriate value of k, the variance of the entire dataset that is explained by each of the principal components can be calculated, as outlined below.

If the original features have been centred to have a mean value of 0, the variance of the data is given by:

Equation 6.6

$$\sum_{j=1}^{p} \boldsymbol{Var}(\boldsymbol{X}_{.j}) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_{ij}^{2}$$

and the variance of the mth principal component is given by:

Equation 6.7

$$\boldsymbol{Var}(\boldsymbol{Z}_{.m}) = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{z}_{im}^{2} = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \boldsymbol{\phi}_{jm} \boldsymbol{x}_{ij} \right)^{2}$$

The variance of the total dataset that is explained by the mth principal component is, therefore, given by:

Equation 6.8

$$\frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{p}\phi_{jm}x_{ij}\right)^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$$

For example, Figure 6.4 shows the variance of the total dataset that is explained by each of the principal components when PCA is applied to the scaled COVID dataset outlined in Section 6.1.1.

Figure 6.4 – Variance explained by principal components (COVID data)

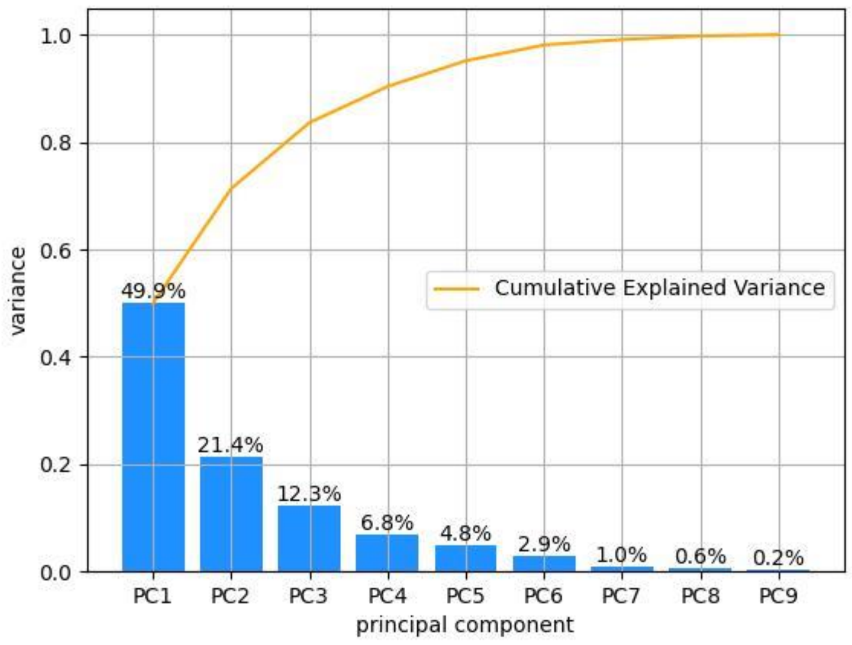


Figure 6.4, which is referred to as a ‘scree plot’, shows that:

- principal component 1 (PC1) explains 49.9% of the dataset’s variance;
- PC2 explains 21.4% of the dataset’s variance;
- PC3 explains 12.3% of the dataset’s variance; and
- PC4 explains 6.8% of the dataset’s variance.

The intrinsic dimension of a dataset is the minimum number of features required to describe the data. In this case, it could be argued that the COVID dataset has an intrinsic dimension of four, as the first four principal components explain a large proportion (approximately 90%) of the variance in the COVID dataset. Hence, a selected value of $k = 4$ seems reasonable.

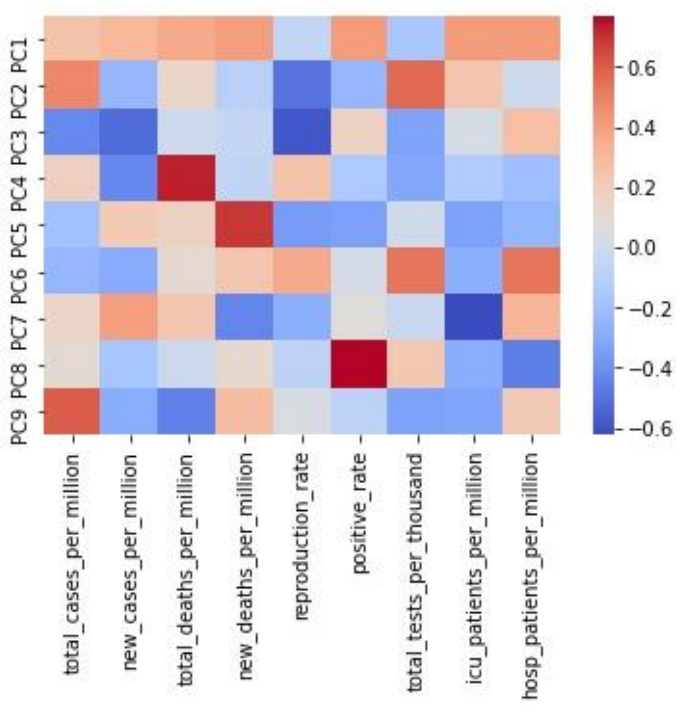
In general, an optimal value for k can be selected by visually inspecting the plot of cumulative variance explained (the orange line in Figure 6.4), and choosing a value for k at an 'elbow' in the curve. This is because additional principal components beyond the 'elbow' only return a marginal or less significant increase in cumulative variance explained, compared to the principal components before the 'elbow'. This elbow method is discussed further in Section 6.3.2.

Investigating the Principal Components

PCA produces new features for a dataset that are not immediately intuitive for someone analysing the data. For example, when PCA was applied to the COVID dataset in the previous section, the dataset changed from having interpretable features such as 'total cases per million people' and 'total new cases per million people' to having features called 'PC1', 'PC2', 'PC3', and so on. It is not obvious what meaning these principal components have or how they should be interpreted.

To infer some meaning from the principal components, it is helpful to consider how much each of the original features of the dataset contributes to the calculation of each of the principal components. This can be visualised using a heat map, such as the one shown in Figure 6.5 for the COVID dataset.

Figure 6.5 – Relationship between COVID dataset features and principal components



The heat map shown in Figure 6.5 contains the loadings $(\phi_{1m}, \phi_{2m}, \ldots, \phi_{pm})$ for each principal component. By plotting these loadings in a heat map, the level of influence of each original feature in the dataset on each of the principal components can be seen.

In Figure 6.5, the red ('hot') boxes indicate a positive contribution of an original feature to a principal component, and the blue ('cold') boxes indicate a negative contribution of a feature to a principal component. The darker the colour in the boxes, the stronger the contribution of the original feature to the principal component. For example, Figure 6.5 suggests the following:

- the first principal component (PC1) is positively influenced by many of the original features from the COVID dataset;
- PC1 is also negatively influenced by 'reproduction_rate' and 'total_tests_per_thousand';
- PC2 is most strongly positively influenced by the features 'total_tests_per_thousand' and 'total_cases_per_million' and is most strongly negatively influenced by 'reproduction rate'; and
- PC3 is most (negatively) influenced by 'reproduction rate' and 'new_cases_per_million'.

Even with the analysis described above, you should always consider whether it is appropriate to apply PCA as a dimension reduction technique in each business context. For example, if a regulator asks you to explain the features in your model, you need to be confident that you could explain exactly what the PCA features are doing and be able to consider issues such as any potential discrimination that might arise from the use of these features.

Visualising the Data

As outlined in the introduction to Section 6.2, PCA can help to visualise high-dimensional data in a lower-dimensional space. For example, it is not possible to plot the COVID data with its nine original features. However, it can be visualised using a scatterplot of the first 2 principal components, as shown in Figure 6.6.

Figure 6.6 – Scatterplot of COVID data on first two principal components

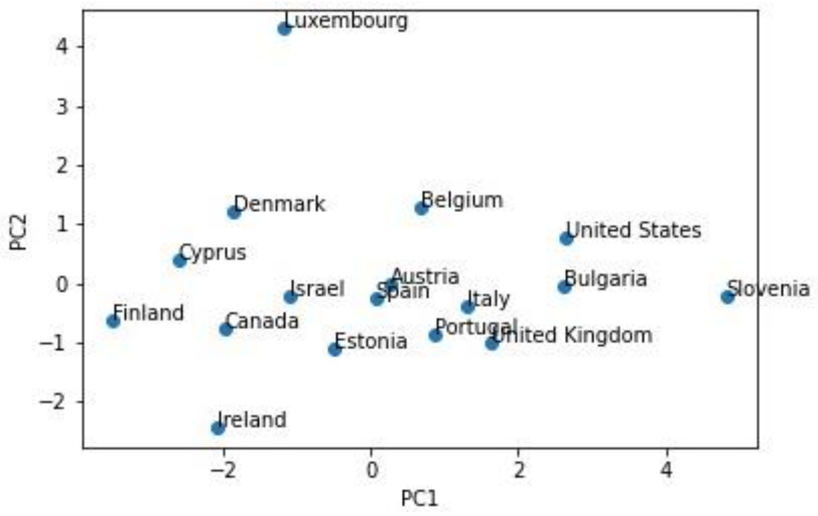


Figure 6.4 shows that PC1 and PC2 explained approximately 70% of the variance in the COVID data. Therefore, although Figure 6.6 does not reflect all of the information available in the COVID dataset, it can help us to quickly draw some observations about the data. For example, Figure 6.6 indicates that Luxembourg is an outlier in this dataset. It would be much more difficult to identify this based on a visual inspection of the original data across each of its nine features.

Exercise 6.1

The notebook 'DSA_C06_Ex1.ipynb' contains the PCA performed on the COVID dataset that is described in the sections above.

Try rerunning this PCA exercise without first scaling the COVID data. You can do this by updating the 'Perform PCA' section to replace:

- pipeline.fit(covid_data) with pca_model.fit(covid_data); and
- pca_features = pipeline.transform(covid_data) with pca_features = pca_model.transform(covid_data).

Explain what happens to the amount of variance explained by each principal component when you remove the scaling step in the analysis.

6.2.2. Other Dimension Reduction Techniques

PCA is relatively quick and easy to implement. However, it is difficult to interpret the derived principal components and the method is best suited to continuous variables. There are other feature extraction methods for dimension reduction that each have their own strengths and weaknesses. These include:

- factor analysis;
- independent component analysis (ICA);
- t-distributed Stochastic Neighbour Embedding (t-SNE); and
- uniform manifold approximation and projection (UMAP).

You do not need to understand the dimension reduction techniques listed above in any detail. However, you should be aware that alternative approaches to PCA exist if PCA is not suitable for the problem you are attempting to solve.[3]

[3] Interested students can find out more about the PCA alternatives listed in this section in the following article: Sharma, P. (2018). The Ultimate Guide to 12 Dimensionality Reduction Techniques (with Python codes). *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/?utm_campaign=News&utm_medium=Community&utm_source=DataCamp.com

6.3. Clustering

Clustering is another unsupervised learning technique. The goal of clustering is to separate observations into homogenous groups or clusters based on their features such that the observations in each cluster are more like each other than observations in other clusters.

Clustering can help to achieve specific outcomes, such as detecting fraud or identifying similar groups of customers. Clustering can also be used to summarise a complex situation or to produce easy-to-understand narratives, which can be used by managers within a business. Clustering algorithms are often run iteratively until there is a clear story to support the development of a business strategy.

Video 6.2 provides a brief introduction to clustering and introduces some examples of how clustering can be used to solve problems.

Video 6.2 – Clustering

https://www.youtube.com/watch?v=wk2yIl1qgU0 (7 mins)

Record your video notes here

There are many different clustering algorithms. These broadly fit into one of the following two categories:[4]

- partitional; or
- hierarchical.

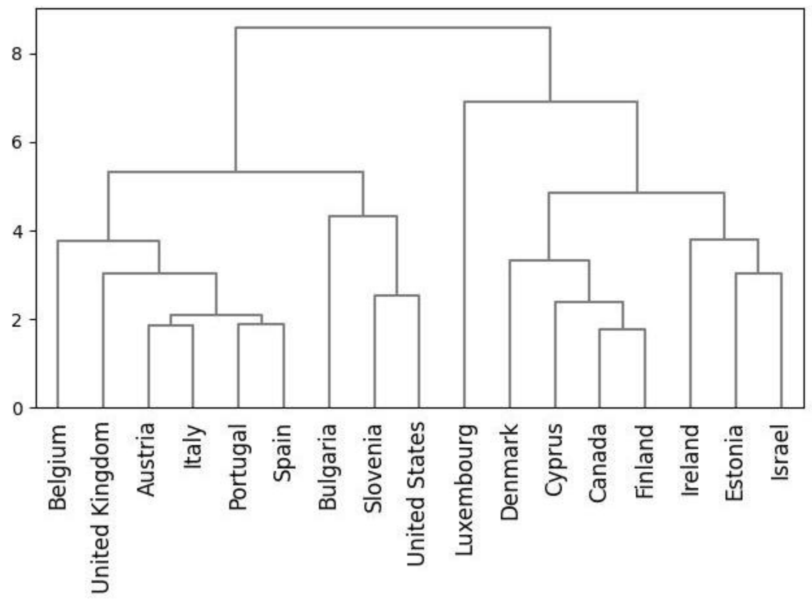
[4] Gan, G., and Valdez, A. E. (2020). Data Clustering with Actuarial Applications. *North American Actuarial Journal*. Campbell, M. (1986). An integrated system for estimating the risk premium of individual car models in motor insurance. *ASTIN Bulletin, 24(2)*, 168–186. This reference was used extensively for the writing of Section 6.3.

Partitional clustering, such as the k-means approach that is discussed in Section 6.3.2, divides a set of observations into K 'partitions' of the data, with each partition representing a cluster. Partitional algorithms require the number of clusters, K, to be specified before training a model.

Partitional algorithms can be further divided into 'hard' and 'soft' clustering. Hard clustering requires each observation to be assigned to only one cluster, while soft clustering allows observations to belong to multiple clusters. Under soft clustering, weights are used to determine the degree to which each observation belongs to each cluster.

Hierarchical clustering, unlike partitional clustering, does not require the number of clusters to be pre-specified. Instead, hierarchical clustering builds a hierarchy of clusters, displayed in the form of a dendrogram. An example dendrogram, based on the COVID data introduced in Section 6.1.1, is shown in Figure 6.7.

Figure 6.7 – Example dendrogram based on COVID data



In the dendrogram in Figure 6.7, each country belongs to a hierarchy of clusters. For example, Italy belongs to the following clusters:

- a cluster at the bottom of the dendrogram that contains only Italy and Austria;
- a cluster higher in the dendrogram that contains Italy, Austria, Portugal, and Spain;
- a cluster higher still in the dendrogram that contains Italy, Austria, Portugal, Spain, and the United Kingdom;
- a cluster slightly higher still in the dendrogram that contains Italy, Austria, Portugal, Spain, the United Kingdom, and Belgium;

- a cluster slightly higher still that contains Italy, Austria, Portugal, Spain, the United Kingdom, Belgium, Bulgaria, Slovenia, and the United States; and
- a cluster at the top of the dendrogram that contains all countries included in the clustering exercise.

Section 6.3.3 provides more information about how to interpret the clusters within a dendrogram.

Hierarchical clustering algorithms can be either agglomerative (bottom-up) or divisive (top-down). The agglomerative approach starts with all observations as individual clusters at the bottom of the dendrogram. These individual clusters are then gradually joined into larger clusters until all observations belong to a single cluster at the top of the dendrogram. The divisive approach works in the opposite direction—all observations start as a single large cluster at the top of the dendrogram. This cluster is then repeatedly broken into smaller clusters until all observations form a unique cluster at the bottom of the dendrogram. This chapter focuses on agglomerative hierarchical clustering, which is discussed in Section 6.3.3.

The justification for the clustering algorithm choices can include:

- Use K-means when:
  - You know the number of clusters in advance.
  - You have large datasets with relatively well-separated, spherical clusters.
  - You need an efficient algorithm for large-scale applications.
- Use Hierarchical Clustering when:
  - You are working with smaller datasets and want a hierarchical understanding of your data.
  - The number of clusters is unknown, and you want flexibility to explore different levels of cluster granularity.
  - You expect clusters of varying sizes and shapes but can afford the computational expense.
- Use DBSCAN when:
  - Your data has noise, or you want to detect outliers.
  - Clusters may be non-spherical or have irregular shapes.
  - You need an algorithm that can determine the number of clusters automatically based on density, particularly for spatial or density-related data.

6.3.1. Distance Measures

To group observations into clusters, there needs to be a way to define what is meant by observations being 'similar' to each other. This is usually defined using a measure of distance between two observations. The following distance measures are discussed (below):

- Euclidean distance;
- correlation-based distance;
- simple matching distance; and
- Gower's distance.

Euclidean distance is a popular measure for clustering continuous data. Euclidean distance (D_{euc}) between two p-dimensional continuous observations, $X_{i.}$ and $X_{i'.}$, is defined as follows:

Equation 6.9
$$\boldsymbol{D_{euc}(X_{i.},X_{i'.}) = \sqrt{\sum_{j=1}^{p}\left(x_{ij} - x_{i'j}\right)^2}}$$

That is, Euclidean distance is the length of a line segment between the two observations in p-dimensional Euclidean space.[5]

Correlation-based distance is another option for clustering continuous data. Correlation measures the degree of association between two vectors. In many statistical contexts, correlation measures the degree of association between two *features*, $X_{.j}$ and $X_{.j'}$. Instead, in the context of clustering, correlation-based distance measures the degree of association between two *observations*, $X_{i.}$ and $X_{i'.}$

Two observations might be highly correlated with one another even if the Euclidean distance between them is large. For example, two shoppers might have very similar mixes of products in their shopping baskets, even though one of these shoppers typically spends 10 times as much as the other. In this example, the contents (features) of these shoppers' baskets will likely have low correlation-based distance but high Euclidean distance.

Correlation-based distance uses correlation measures such as Pearson, Spearman or Kendall correlation, as follows:

[5] Originally, Euclidean space was the three-dimensional geometric space. In modern mathematics, there are Euclidean spaces of any nonnegative integer dimension, including the one-dimensional line, the two-dimensional plane and the three- (or more) dimensional space.

Equation 6.10
$$\boldsymbol{D_{cor}(X_{i.}, X_{i'.}) = 1 - \rho}$$

where ρ is the correlation coefficient between observations $X_{i.}$ and $X_{i'.}$.

If, for example, Pearson correlation is used in Equation 6.10, then ρ is calculated as follows:

Equation 6.11
$$\boldsymbol{\rho = \sum_{j=1}^{p}(x_{ij} - \bar{X}_{i.})(x_{i'j} - \bar{X}_{i'.}) \Big/ \sqrt{\sum_{j=1}^{p}(x_{ij} - \bar{X}_{i.})^2 \sum_{j=1}^{p}(x_{i'j} - \bar{X}_{i'.})^2}}$$

where $\bar{X}_{i.}$ is the average value of the j features for observation i.

Simple matching distance is a commonly used measure for discrete data. Simple matching distance (D_{sim}) between two p-dimensional discrete observations, $X_{i.}$ and $X_{i'.}$, is defined as follows:

Equation 6.12
$$\boldsymbol{D_{sim}(X_{i.}, X_{i'.}) = \sum_{j=1}^{p}\delta(x_{ij}, x_{i'j})}$$

where $\delta(x_{ij}, x_{i'j}) = \begin{cases} 0, if\ x_{ij} = x_{i'j}, \\ 1, if\ x_{ij} \neq x_{i'j}. \end{cases}$

There are also distance measures that are suitable for datasets that contain a mixture of continuous and discrete features. One such measure is Gower's distance,[6] which is defined as follows:

Equation 6.13
$$\boldsymbol{D_{gow}(X_{i.}, X_{i'.}) = \sqrt{\frac{1}{p}\sum_{j=1}^{p} d^2(x_{ij}, x_{i'j})}}$$

where $d(x_{ij}, x_{i'j})$ is a distance measure for the j^{th} feature and is defined differently for ordinal, continuous and nominal features. For ordinal and continuous features, the distance measure is defined as:

Equation 6.14
$$\boldsymbol{d(x_{ij}, x_{i'j}) = \frac{|x_{ij} - x_{i'j}|}{R_j}}$$

where R_j is the range of the j^{th} feature.

For nominal features, the distance measure is defined as:

Equation 6.15
$$\boldsymbol{d(x_{ij}, x_{i'j}) = \delta(x_{ij}, x_{i'j})}$$

where $\delta(x_{ij}, x_{i'j})$ is as defined in Equation 6.12.

Gower's distance measure, as set out in Equation 6.13, applies equal weight to all features in the data. It is also possible to apply different weights to different features. This allows domain knowledge to be used to place greater weight on features that are considered to be more

[6] Gower, J. (1971). A general coefficient of similarity and some of its properties. *Biometrics 27(4),* 857-874.

important for measuring distance between observations in any given context. The ‘gower’ package in Python[7] allows different weights to be applied to each feature when calculating Gower’s distance.

The choice of distance measure can have a strong influence on the outcome of a clustering exercise. For example, imagine a project that aims to create clusters of bank customers based on the profile of the services they have purchased. In this case, a correlation-based distance measure might be more suitable, as it will group together customers with a similar overall mix of services. This contrasts to the use of Euclidean distance, which might, instead, cluster together customers with a similar magnitude of expenditure on services. This is another example of the business context being an important factor when deciding on which option to choose in your modelling.

The choice of distance measure might also inform whether you need to apply feature scaling before calculating the distances between observations. For example, if you use correlation-based or simple matching distance measures, feature scaling should not be performed prior to calculating distances. In the case of correlation-based distance, feature scaling is already performed within the distance calculation, with the mean and standard deviation of each feature being explicitly used in the calculation. The simple matching distance measure is only concerned with whether observations are the same or not, hence the scale of each feature does not impact the calculation.- However, the use of Euclidean distance will be impacted by the scale of each feature, and feature scaling should be used to avoid very large features dominating in the calculation of distances between observations.

Here are key considerations when selecting a distance measure:

- Data Type: Choose Euclidean or Manhattan distance for purely numeric data. Choose Jaccard or Gower’s distance for categorical or mixed-type data.
- Data Distribution and Scale: Choose Euclidean distance for normalized data without significant outliers. Choose Manhattan distance if the data contains outliers or does not need to be squared.
- Cluster Shape: Choose Euclidean distance if you expect spherical clusters. Choose Manhattan or Cosine distance if you expect elongated or irregular clusters.

[7] Yan, M. (2019). Introducing Python Package --- gower. *Think Data Science.* https://www.thinkdatascience.com/post/2019-12-16-introducing-python-package-gower/

- Dimensionality of the Data: Choose Cosine similarity for high-dimensional data, especially when direction (and not magnitude) is important (e.g., text or document clustering). Choose Euclidean or Manhattan distance for low-dimensional data or when magnitude matters.
- Nature of the Clustering Algorithm: Choose Euclidean distance for algorithms like K-means. Explore different measures for algorithms like hierarchical clustering to find the best fit for the data structure.
- Interpretability: Choose Euclidean or Manhattan if interpretability is important. Use Cosine similarity for tasks where similarity is the focus rather than distance per se (e.g., document clustering).
- Data Sparsity: Choose Cosine similarity for sparse datasets. Choose Euclidean or Manhattan distance for dense datasets.
- Correlation between Features: Choose Correlation-based distances if features are highly correlated and you're interested in patterns or trends. Choose Euclidean or Manhattan distance for independent features where absolute differences matter.
- Type of Similarity Focus: Choose Euclidean/Manhattan for magnitude differences. Choose Cosine similarity for directional or orientation-based comparisons.
- Computational Complexity: Choose Manhattan distance if computational efficiency is a priority. Choose Cosine similarity for efficiency in sparse, high-dimensional data.

Exercise 6.2

The three observations shown in the table below are from the data tables introduced in Chapter 4 (Data architecture).

Observation	Name	Height (m)	Date of birth	Gender
A	Robert	0.91	1 March 1988	Male
B	Suzanne	0.95	11 August 1986	Female
C	Robert	1.10	1 March 1988	Male

Identify which distance measure(s) could be used to calculate the distance between each of these three observations.

Determine which two of the observations are most similar.

6.3.2. K-Means Clustering

K-means clustering is a simple and popular partitional clustering algorithm. Its aim is to partition a dataset into K clusters by minimising the amount by which observations in a cluster differ from each other ('within-cluster variation').

A value of K needs to be specified by the user before the clustering algorithm can be performed.

Let $C_1, \dots C_K$ denote K clusters that contain, in total, each observation in an $n \times p$ dataset X, where x_i denotes observation i. Each observation in X belongs to exactly one of the K clusters. Thus, $C_1 \cup C_2 \cup \dots \cup C_K = X = \{X_{1,}, X_{2,}, \dots, X_{n.})$ and $C_i \cap C_{i'} = \emptyset \; if \; i \neq i'$.

The within-cluster variation for cluster C_k, denoted by $W(C_k)$, is defined using distance measures such as those presented in Section 6.3.1. For example, if the distance measure used is Euclidean distance, then:

Equation 6.16 $$\boldsymbol{W(C_k) = \frac{1}{|C_k|} \sum_{X_{i,}, X_{i'.} \in C_k} D_{euc}(X_{i,}, X_{i'.})}$$

where $|C_k|$ is the number of observations in C_k. Other distance measures, such as those set out in Section 6.3.1, can be substituted for $D_{euc}(X_{i,}, X_{i'.})$ in Equation 6.16.

The loss function to be minimised across all K clusters can then be defined as:

Equation 6.17 $$\boldsymbol{J(C_1, \dots C_K) = \sum_{k=1}^{K} W(C_k) = \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{X_{i,}, X_{i'.} \in C_k} D_{euc}(X_{i,}, X_{i.'})}$$

Values for $C_1, \dots C_K$ that minimise $J(C_1, \dots C_K)$ can be found using the following algorithm:

Algorithm 6.1

1. Randomly assign a number, from 1 to K, to each observation, $X_{i,}$, in X.
2. Repeat the following steps until the cluster to which each observation is assigned remains constant:
   2.1. for each of the K clusters, compute the cluster 'centroid', Z_k, as the vector of the means of each of the p features across all observations in the kth cluster; and then
   2.2. assign each observation, $X_{i,}$, to the cluster whose centroid is closest (where closest is defined using the chosen distance measure).

Note that in k-means clustering, the cluster centroid, Z_k, can only be calculated if the features are continuous. If the features are nominal or ordinal, the k-modes clustering algorithm can, instead, be used. K-modes is similar to k-means in principle but uses modes instead of means and, thus, its loss function is different than that shown for the k-means algorithm in Equation 6.17.

As Algorithm 6.1 is run, the clustering obtained will keep reducing the loss function until the cluster to which each observation is assigned stops changing. This signifies that a local optimum has been reached.

As was discussed in Chapter 5 (Classification and neural networks), the reaching of a local optimum does not guarantee that a global minimum has been found. It is, therefore, important to try different initial random assignments of the observations (Step 1 of Algorithm 6.1) before deciding on the best clustering result to use.

Video 6.3 provides an example of the k-means algorithm being applied to a simple dataset. The notation used in Video 6.3 is slightly different to that outlined above, but the underlying algorithm used is the same.

Video 6.3 – K-means example

https://www.youtube.com/watch?v=_aWzGGNrcic&t=17s (7 mins)

Record your video notes here

Video 6.3 touched on the issue of choosing a suitable value for K. This might be determined based on the purpose of the clustering exercise. For example, if the goal of clustering is to group customers into four different segments, then the value of K is pre-determined as being 4.

Often, however, the goal of a clustering exercise will be less tightly defined than this. Instead, the goal might be, for example, to find some homogenous customer segments to help tailor the marketing strategy for each segment to increase sales volumes. In this case, you might run the k-means algorithm for different values of K before deciding on the best clustering model to use.

In the above example, you might use an ‘elbow curve’ to select the most suitable value of K. This method involves plotting a validation measure for the clustering outcomes (see Section 6.3.5) against different values of K and using the 'elbow' of the curve to select the number of clusters to use. The elbow is the point on the curve at which adding additional clusters does not provide much better clustering outcomes. The elbow curve method is explained in Video 6.4.

Video 6.4 – The elbow curve method for choosing K

https://www.youtube.com/watch?v=FqIGui0rwh4 (6 mins)

Record your video notes here

Exercise 6.3

Plot the following points on a Cartesian plane:

X = {(0,1), (7,7), (10,7), (2,4), (1,0), (8,6), (1,1), (9,6), (2,3), (7,9)}

Apply k-means clustering to partition the points into two clusters.

Exercise 6.4

The notebook 'DSA_C06_Ex4.ipynb' contains Python code to apply k-means clustering to the COVID dataset described in Section 6.1.1.

Try different values of K.

Examine the relationship between different countries' COVID experience by running the algorithm with different numbers of clusters.

6.3.3. Hierarchical Clustering

This section explains agglomerative hierarchical clustering. This is a bottom-up approach to hierarchical clustering. It starts with every observation as a separate cluster at the bottom of the tree (dendrogram) and repeatedly merges clusters together until there is only one cluster at the top of the dendrogram that contains all observations in the dataset.

As for the k-means algorithm, the steps for the agglomerative hierarchical clustering algorithm are relatively simple, as follows:

Algorithm 6.2

1. Assign each observation, $X_{i,}$, in X to its own cluster, denoted C_k for $k = 1, 2, \dots, n$.
2. Calculate the distance between C_k and $C_{k'}$ for all $1 \leq k \leq n$ and $1 \leq k' \leq n$.
3. Repeat the following steps until only one cluster is left:
   a. merge the two clusters that have the smallest distance between them to form a new cluster; and then
   b. calculate the distances between the new cluster and the remaining clusters.

While the hierarchical clustering algorithm appears to be straightforward, a complicating factor is how to calculate the distance between the clusters that have more than one observation in them.

Calculating Distance Between Clusters

When calculating the distance between clusters, the following two choices need to be made:

- the distance measure to use to calculate the distance between individual observations; and
- the method of 'linkage' to use to calculate the distance between clusters, which may contain more than one observation.

The various methods available for calculating the distance between individual observations were discussed in Section 6.3.1.

Once the distance measure for individual observations has been determined, these can be used by 'linkage' methods to calculate the distance between two clusters. Four common linkage methods are:

- complete linkage;
- single linkage;
- average linkage; and
- centroid linkage.

Complete linkage between two clusters ($C_k$and $C_{k'}$) measures the *largest* pairwise distance between observations in C_k and observations $C_{k'}$.

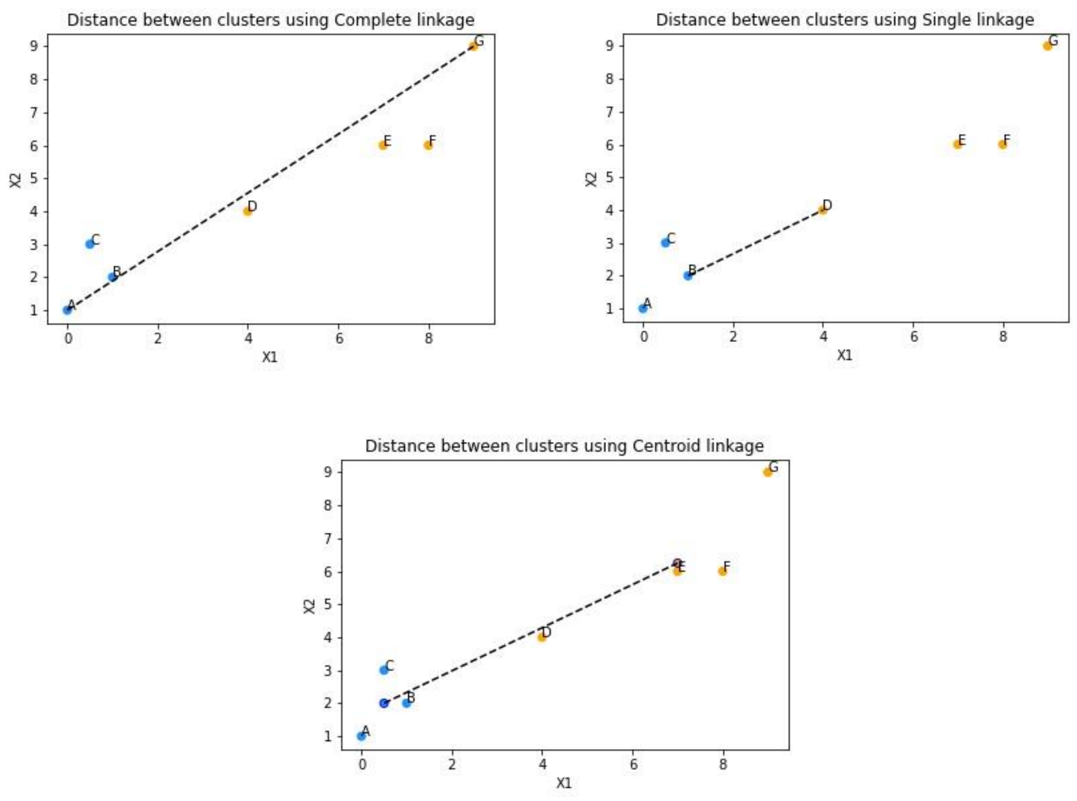
Single linkage between two clusters ($C_k$and $C_{k'}$) measures the *smallest* pairwise distance between observations in C_k and observations $C_{k'}$.

Average linkage between two clusters ($C_k$and $C_{k'}$) measures the *average* pairwise distance between observations in C_k and observations $C_{k'}$.

Centroid linkage between two clusters ($C_k$and $C_{k'}$) measures the distance between the centroid of C_k and the centroid of $C_{k'}$, where 'centroid' has the same definition as it does for the k-means clustering algorithm (see Section 6.3.2).

To help illustrate the differences between each of the above linkage methods, consider the seven observations that have been assigned to two clusters (blue and orange) in the plots in Figure 6.8. Note that the method that has been used to assign these observations to the two clusters is not important for this example.

Figure 6.8 – Calculating the distance between clusters



Assume that Euclidean distance is the distance measure used to calculate the distance between individual observations.

The *largest* distance between an observation in the blue cluster and an observation in the orange cluster is the straight line between observation A and observation G, represented by the dotted line in the top left-hand plot in Figure 6.8. Under complete linkage, the distance between the blue and orange clusters is, therefore, the distance between observation A and observation G.

The *smallest* distance between observations in the two clusters is the straight line between observation B and observation D, represented by the dotted line in the top right-hand plot in Figure 6.8. Under single linkage, the distance between the blue and orange clusters is, therefore, the distance between observation B and observation D.

Under average linkage, all of the pairwise distances between observations in the blue cluster (A, B and C) and those in the orange cluster (D, E, F and G) are calculated. This results in 12 calculated distances—that is, the distance between A and each of D, E, F and G, the distance between B and each of D, E, F and G, and the distance between C and each of D, E, F and G. The average of these 12 distances is then used to determine the overall distance between the blue and orange clusters.

Finally, under centroid linkage, the distance between the blue and orange clusters is the distance between the centroid of the blue cluster and the centroid of the orange cluster. This is shown as a dotted line in the bottom plot in Figure 6.8.

The choice of linkage method can have a large impact on the resulting dendrogram. Average and complete linkage methods tend to produce more balanced or 'ball-shaped' clusters in which clusters of multiple observations are commonly fused together. In contrast, single linkage can result in extended, string-like clusters in which single observations are fused into other clusters one at a time.

Exercise 6.5

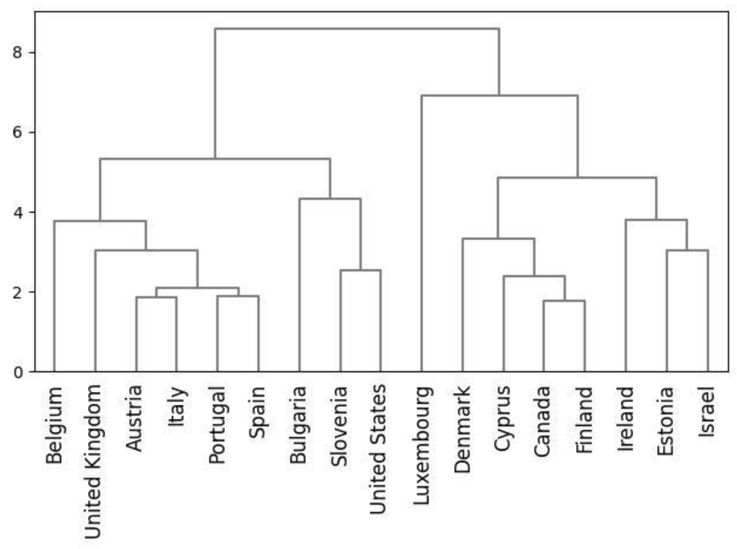
Explain why single linkage can result in 'extended, string-like clusters'.

It can be helpful to experiment with different linkage methods and to compare the outcomes under each before deciding which is the most appropriate to use.

Creating a Dendrogram

The output of a hierarchical clustering algorithm is often displayed as a dendrogram that can be 'cut' at different heights to produce different sets of clusters. Figure 6.9 repeats the dendrogram from Figure 6.7, which is based on the COVID data introduced in Section 6.1.1.

Figure 6.9 – Dendrogram example



The bottom of the dendrogram in Figure 6.9 shows each of the individual observations (countries, in this case). Higher up the tree:

- individual observations fuse with other observations (e.g. Austria fuses with Italy);
- clusters of observations fuse with other individual observations (e.g. Slovenia and the United States fuse with Bulgaria); and
- clusters of observations fuse with other clusters (e.g. Australia and Italy fuse with Portugal and Spain).

The height at which each of these fusions is recorded in the dendrogram is the measured distance between the fused clusters. Therefore, the higher up the dendrogram that a fusing takes place, the greater the distance (or dissimilarity) between the two clusters that are fused together. For example, Austria and Italy fuse at a low point on the dendrogram, indicating these two countries are more like each other than are Estonia and Israel, which fuse together at a higher point.

In addition, while the United States is next to Luxembourg on the horizontal axis of Figure 6.9, they do not fuse together into the same cluster until at the very top of the dendrogram. This suggests that the United States is quite dissimilar to Luxenbourg based on their COVID experiences.

Exercise 6.6

The notebook 'DSA_C06_Ex6.ipynb' applies agglomerative hierarchical clustering to the COVID dataset described in Section 6.1.1. In the code supplied, Euclidean distance is used by default and the complete linkage method has been selected.

Describe how the dendrogram changes when a 'correlation' distance measure is used rather than Euclidean distance.

Describe how the dendrogram changes when single, average, or centroid linkage methods are used.

Obtaining Clusters

Once a dendrogram has been created, clusters of observations can be obtained by 'cutting' the dendrogram at a particular height, depending on the number of clusters that are desired.

For example, if the dendrogram in Figure 6.9 is cut at a height of 5 then three clusters (and one 'outlier') are returned, as shown in Figure 6.10.

Figure 6.10 – Cutting a dendrogram to obtain clusters

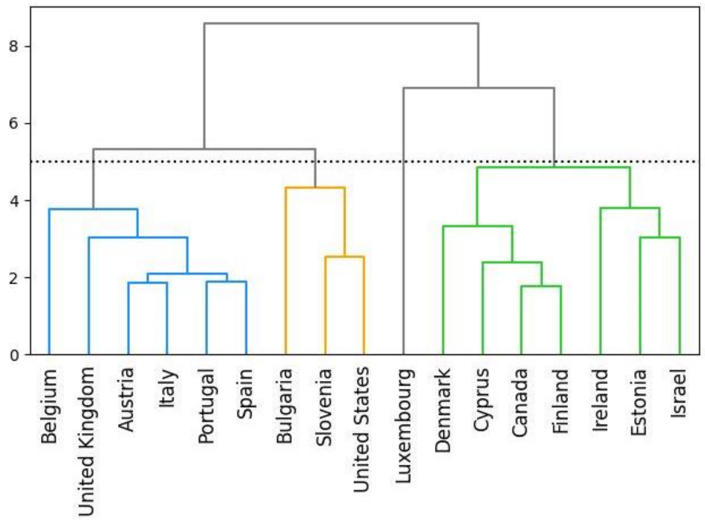


Figure 6.10 suggests that the dataset contains the following three distinct clusters:

- Cluster 1 (blue): Belgium, the United Kingdom, Austria, Italy, Portugal, and Spain;
- Cluster 2 (yellow): Bulgaria, Slovenia, and the United States; and
- Cluster 3 (green): Denmark, Cyprus, Canada, Finland, Ireland, Estonia, and Israel.

It also suggests that the COVID experience for Luxembourg has been quite different to that of the other countries included in this analysis. Luxembourg is effectively treated as its 'cluster' in Figure 6.10.

The height that a dendrogram is cut will determine the number of clusters that are obtained. For example, if Figure 6.10 had been cut at 7 rather than at 5, the following two clusters would be obtained:

- Cluster 1 (blue and yellow): Belgium, the United Kingdom, Austria, Italy, Portugal, Spain, Bulgaria, Slovenia, and the United States; and
- Cluster 2 (Luxembourg and green): Luxembourg, Denmark, Cyprus, Canada, Finland, Ireland, Estonia, and Israel.

This demonstrates one advantage of hierarchical clustering over k-means clustering—rather than specifying the number of clusters before the algorithm is run, the number of clusters can be decided after analysing the results of hierarchical clustering.

6.3.4. Other Clustering Techniques

While k-means and hierarchical clustering are currently two popular clustering algorithms, there are many others that also exist. This section provides a brief outline of several alternative clustering algorithms.[8]

Fuzzy C-Means

The introduction to Section 6.3 noted that k-means is a partitional clustering algorithm that divides a set of observations into K partitions. Further, k-means is a 'hard' partitional clustering algorithm that requires each observation to be assigned to only one cluster. In contrast, fuzzy c-means is an example of a 'soft' partitional clustering algorithm that allows each observation to belong to more than one cluster.

Fuzzy c-means employs an iterative process to find the K clusters similar to the algorithm used in k-means clustering. The user must also assign a value to a parameter $\propto$, which is called the 'fuzzifier'. When $\propto \rightarrow \infty$, each observation will have equal membership to each of the K clusters.

[8] For further information on these and other alternative clustering techniques, see Gan, G., and Valdex, E.A. (2020). Data Clustering with Actuarial Applications. *North American Actuarial Journal*. 24:2. 168–186. https://www.tandfonline.com/doi/full/10.1080/10920277.2019.1575242

When $\propto \rightarrow 1$, each observation is assigned to only one cluster and the k-means algorithm is recovered.

The outcome from a fuzzy c-means algorithm is a set of values ($u_{ik} \in [0,1]$) that specify the degree to which observation i belongs to cluster k, where $\sum_{k=1}^{K} u_{ik}$ = 1 for $i = 1, 2, \dots, n$ and $\sum_{i=1}^{n} u_{ik}$ > 0 for $k = 1, 2, \dots, K$.

Divisive Hierarchical Clustering

Divisive hierarchical clustering was introduced in Section 6.3. Unlike agglomerative hierarchical clustering (see Section 6.3.3), which uses a bottom-up approach, divisive hierarchical clustering uses a top-down approach. The top-down approach begins with all observations belonging to a single cluster. At each step of the algorithm, a cluster is split into two, and the number of clusters is increased by one.

Divisive hierarchical clustering can be very time-consuming, as there are many ways that a single cluster can be split into two. For this reason, agglomerative hierarchical clustering tends to be a more popular approach.

Density-Based Clustering[9]

One pitfall in hierarchical and k-means clustering is that every observation is assumed to be a part of one of the derived clusters. Density-based clustering avoids this pitfall by only clustering observations that are tightly packed and assuming that other observations are noise.

To define what is meant by 'tightly packed', a user must specify two parameters:

- a radius, ϵ; and
- a neighbourhood density, θ.

For every observation, a count is made of the number of other observations that are close to (at most ϵ away from) the observation. Each observation is then classified as:

- a 'core' observation if it has more than θ close observations;
- a 'border' observation if it has fewer than θ close observations but is itself a close observation of a core observation; or

[9] Information about this clustering algorithm was sourced from the following article: Open Data Science. (2018). Three Popular Clustering Methods and When to Use Each. *Medium*. https://medium.com/predict/three-popular-clustering-methods-and-when-to-use-each-4227c80ba2b6

- noise.

The number of clusters does not need to be specified for this algorithm. Instead, it will automatically discover some number of clusters based on the specified values of ϵ and θ, where the parameter ϵ tells the algorithm how far away an observation must be from a certain measurement point (such as a centroid cluster) before it is considered to be noise.

Scalable Clustering Algorithms

As the size of datasets grows, the number of clusters that a user wishes to obtain can also grow. These two effects can present computational challenges for running clustering algorithms efficiently.

Some clustering algorithms that are efficient in dividing a large dataset into many clusters are the:

- truncated fuzzy c-means; and
- hierarchical k-means.

The truncated fuzzy c-means (TFCM) is an extension of the fuzzy c-means algorithm. TFCM reduces the number of distance calculations required during the iterative process of the fuzzy c-means algorithm.

Hierarchical k-means, as the name suggests, uses a combination of hierarchical and k-means clustering to help speed up the clustering process. This algorithm uses a divisive hierarchical clustering approach, with k-means used repeatedly to split each cluster into a small number of K clusters.

6.3.5. Validating Clustering Outcomes

In supervised learning, response variables are available to both train a model and to validate its predictions. In unsupervised learning, response variables are often not available and not even relevant for the problem being solved. Instead, validation of clustering outcomes involves assessing whether the characterisation of the data or simplification of the features is reasonable. This means alternative ways from those used to validate supervised learning outcomes are needed to determine the reasonableness of a clustering model's outcomes.

The following items represent three options for validating clustering outcomes:

- internal validation;
- external validation; and
- manual validation.

Internal validation evaluates the clustering results based only on quantities and features from the underlying dataset. This method of validation usually assigns a score to the clustering algorithm based on the *level of similarity* between observations *within* a cluster and the *level of dissimilarity between* clusters.

Examples of internal validation measures that can be used are:

- within-cluster sum of squares;
- Davies-Bouldin index; and
- Dunn index.

Within-cluster sum of squares, $WCSS$, measures how far each observation is from its cluster centroid. It is calculated as the sum of squares of the distances of each observation in all clusters to their respective cluster centroids as follows:

Equation 6.18
$$\boldsymbol{WCSS = \sum_{k=1}^{K} \sum_{X_{i.} \in C_k} d(X_{i.},\ Z_k)^2}$$

where K is the number of clusters, C_k is cluster k, Z_k is the centroid of cluster k and $d(X_{i.},\ Z_k)$ is the distance, using an appropriate measure such as Euclidean, between observation i, $X_{i.}$and the cluster centroid, Z_k, to which observation i belongs. A lower value of $WCSS$ is desirable, as this represents less variation of observations within each cluster.

Unlike the $WCSS$ measure, the Davies-Bouldin index, DB, allows for both *intra*-cluster variation (variation *within* a cluster), which should be as small as possible, and *inter*-cluster variation (variation *between* clusters), which should be as large as possible. It is calculated as follows:

Equation 6.19
$$\boldsymbol{DB = \frac{1}{K} \sum_{k=1}^{K} \max_{k \neq k'} \left(\frac{\sigma_k + \sigma_{k'}}{d(Z_k, Z_{k'})} \right)}$$

where K is the number of clusters, σ_k is the average distance (using an appropriate measure such as Euclidean) of all observations in cluster k to centroid Z_k and $d(Z_k, Z_{k'})$ is the distance between centroids Z_k and $Z_{k'}$, again using a distance measure such as Euclidean distance.

A low DB index is desirable, as this arises from having low variability within clusters (i.e. the σ_k terms are small) and high variability between clusters (i.e. the distances between cluster centroids, $d(Z_k, Z_{k'})$, are large; hence, $\frac{1}{d(Z_k, Z_{k'})}$ terms are small).

The Dunn index (D) also tries to optimise both *intra*- and *inter*-cluster variability. It is calculated as follows:

Equation 6.20
$$\boldsymbol{D} = \min_{1\le k<k'\le K} \boldsymbol{d}(\boldsymbol{k},\boldsymbol{k}') \Big/ \max_{1\le k\le K} \boldsymbol{d}^*(\boldsymbol{k})$$

where K is the number of clusters, $d(k,k')$ is the distance between clusters k and k' and $d^*(k)$ is the intra-cluster distance of cluster k.

The distance measure $d(k,k')$ can be measured in a variety of ways such as by using complete, single, average, or centroid linkage (see Section 6.3.3). $d^*(k)$ can also be measured in a variety of ways, such as by calculating the minimum, maximum or average distance between any pairs of observations in cluster k.

A high Dunn index is desirable as this will arise from having large distances between clusters (i.e. high values of $d(k,k')$) and low variation within clusters (i.e. low values of $d^*(k)$).

One drawback of using internal validation is that the validation measure will be biased towards clustering algorithms that use the same measure of similarity. For example, assume that the internal validation measure uses Euclidean distance to measure the similarity between observations within a cluster. This validation measure will likely give a higher score to a clustering algorithm that also uses Euclidean distance as opposed to a clustering algorithm that uses a different distance measure. However, this does not necessarily mean that the clusters produced by the first algorithm are more useful to the end-user than those produced by the second algorithm.

External validation involves evaluating clustering outcomes based on data that was *not* used for the clustering. For example, clustering outcomes might be compared with responses that were not used in the clustering algorithm (if available) or with external benchmarks that are often created by (expert) humans.

External validation measures how close the clustering outcomes are to the predetermined benchmark classes, which is similar to the task of validating classifiers discussed in Chapter 5 (Classification and neural networks).

External validation includes the use of measures such as:

- the Rand index ($RI = \frac{TP+TN}{TP+FP+FN+TN}$); and
- the F-score ($F = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$).

Here, TP = true positive, TN = true negative, FP = false positive and FN = false negative, as described in Chapter 5.

Under external validation, the higher the Rand index or F-score of a model, the better the model is considered to be.

One issue with external validation is that it requires prior knowledge of the clusters that exist in the dataset being analysed. This often runs counter to the aim of a clustering algorithm, which is to *discover* the underlying structure of a dataset that was previously unknown.

Manual validation involves validation by a human expert, who reviews the clustering outcomes and makes a subjective evaluation of whether they are intuitive, given the expert's understanding of the underlying data. For example, manual validation of the clustering exercises undertaken in Exercise 6.4 and Exercise 6.6 might involve asking the following experts to judge whether the resulting clusters make sense, given their experience and knowledge:

- an epidemiologist who has studied the spread of COVID-19 in different countries; and
- a political scientist who has studied government responses to COVID-19 in different parts of the world.

As with most validation activities, it is likely that a combination of the above validation methods will be the most effective in determining the best model to adopt for the given purpose.

The **optimal number of clusters** will be typically determined by reference to multiple validation methods. The following issues should be considered when determining the optimal number of clusters:

- Purpose of Clustering Exercise: We want to understand the nature and outcomes of the calls for insights purposes. Therefore, the optimal cluster count needs to be limited to a size humans can intuitively grasp in their working memory i.e. no more than 7 or 8 clusters.
- Internal Validation: Quantify the quality of clustering.
  - Silhouette Score: The Silhouette Score can be used to determine the optimal number of clusters by calculating the average silhouette coefficient for different numbers of clusters. The highest Silhouette Score indicates the best clustering configuration, where a value closer to 1 suggests dense and well-separated clusters. Therefore, the number of clusters corresponding to the peak Silhouette Score is considered optimal.
  - Within-Cluster Sum of Squares (WCSS): WCSS measures the distance of each observation from its cluster centroid. It can be used to determine the optimal number of clusters by looking for the "elbow point" in the WCSS plot, where the rate of decrease slows down.

  - Davies-Bouldin Index: The Davies-Bouldin Index considers both intra-cluster and inter-cluster variations. A lower index value indicates better clustering, helping to identify the optimal number of clusters.
  - Dunn Index: The Dunn Index evaluates the compactness and separation of clusters. A higher Dunn Index is desirable, indicating well-separated clusters, which can guide the determination of the optimal number of clusters.
  - 2-Dimensional Plotting: 2-dimensional plotting can be used to determine the optimal number of clusters by visualising the data points and observing how they naturally group together. Assign a different colour to each cluster. The two plot axes can be the first 2 PCA or t-SNE dimensions.

- Manual Validation: Manual validation involves human judgment and domain knowledge to assess clustering quality. It includes visual inspection, comparing with known labels, and interpreting the real-world intuitive meaning of cluster differences. The optimal number of clusters will separate the data into meaningful and relevant groups. There will be enough clusters to separate important groups, but not so many that similar (specific to the business context) groups are not clustered together.

6.4. Case Studies

This section presents two case studies that illustrate how PCA and clustering can be used to solve real-world problems.

6.4.1. Case study 1—The Socio-Economic Indexes for Areas

Define the Problem

This case study demonstrates the use of PCA in the construction of an index to rank the socio-economic status of different geographical areas within a country. Such an index can be useful to identify areas that require funding or services, or as input for research into the relationship between socio-economic conditions and other outcomes in different areas.

In Australia, the Australian Bureau of Statistics (ABS) produces such indexes, referred to as the Socio-Economic Indexes for Areas (SEIFA).[10] These indexes rank areas in Australia based on their relative socio-economic advantage and disadvantage.

Exercise 6.7

Read pages 15 to 20 of the SEIFA technical document: https://www.abs.gov.au/ausstats/subscriber.nsf/log?openagent&SEIFA%202016%20Technical%20Paper.pdf&2033.0.55.001&Publication&756EE3DBEFA869EFCA258259000BA746&&2016&27.03.2018&Latest.

This case study aims to replicate, as closely as possible, the methodology adopted by the ABS in constructing one of these indexes, namely the Index of Relative Socio-Economic Disadvantage (IRSD).

Design a Solution

In Australian Government data, there are three different levels that are used to group geographical areas together:

[10] Australian Bureau of Statistics. (2011). Socio-Economic Indexes for Areas. https://www.abs.gov.au/websitedbs/censushome.nsf/home/seifa.

- Statistical Area Level 1 (SA1);
- Statistical Area Level 2 (SA2); and
- Statistical Area Level 3 (SA3).

SA1 is the most granular level, with each SA1 having an average population of approximately 400 people. Most of the data generated from the ABS Census is available at SA1. SA1s are referred to by number only. Each **SA2** is similar in size to suburbs, and these are referred to by suburb names. Each **SA3** is a region of Australia.

For this PCA exercise, each observation is an SA1. The features for each SA1 that are used in the index are derived variables based on the Australian Census data for 2016. These features are summarised in Table 6.1.

Table 6.1 – Features in the IRSD PCA analysis[11]

Feature name	Description
inc_low	% people with stated annual household equivalised income between $1 and $25,999
childjobless	% families with children under 15 years of age who live with jobless parents
nonet	% occupied private dwellings with no Internet connection
noyr12higher	% people aged 15 years and over whose highest level of education is Year 11 or lower
unemployed	% people (in the labour force) unemployed
lowrent	% occupied private dwellings paying rent less than $215 per week (excluding $0 per week)
oneparent	% one-parent families with dependent offspring only
disabilityu70	% people aged under 70 who have a long-term health condition or disability and need assistance with core activities
sepdivorced	% people aged 15 and over who are separated or divorced
nocar	% occupied private dwellings with no cars
noedu	% people aged 15 years and over who have no educational attainment

[11] These features correspond to those shown in Table 4.2 of the SEIFA technical document referenced in Exercise 6.6.

Feature name	Description
englishpoor	% people who do not speak English well
labour	% employed people classified as ‘labourers’
drivers	% employed people classified as Machinery Operators and Drivers
service	% employed people classified as Low Skill Community and Personal Service workers

The data for the 15 variables for 53,956 of the SA1 is available in the file ‘DSA_C06_CS1_data1.csv’. The IRSD is essentially a scaled version of the first principal component of this dataset.

The notebook ‘DSA_C06_CS1.ipynb’ contains the steps used to replicate the ABS’s calculation of the IRSD. You should review this code now.

Monitor the Results

The loadings for the principal components of the data are presented in Table 6.2.

Table 6.2 – Loadings for principal components for the IRSD dataset

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12	PC13	PC14	PC15
inc_low	-0.36	0.03	-0.31	0.06	-0.08	0.08	-0.17	0.00	0.07	-0.03	0.03	-0.06	0.75	-0.39	0.01
childjobless	-0.08	0.19	0.12	-0.57	0.51	0.42	-0.08	-0.26	0.28	-0.05	0.13	0.00	-0.01	-0.03	-0.01
nonet	-0.39	-0.04	-0.19	0.13	-0.02	0.11	0.13	-0.06	0.10	-0.07	0.16	-0.45	-0.03	0.55	0.47
noyr12higher	-0.31	-0.35	0.26	0.06	0.03	-0.04	0.15	-0.14	0.01	0.01	0.06	-0.46	-0.04	-0.03	-0.67
unemployed	-0.29	0.24	0.01	-0.29	0.12	0.03	0.10	0.19	-0.60	0.21	-0.53	-0.11	0.02	0.08	-0.01
lowrent	-0.29	-0.16	-0.22	0.25	0.00	0.36	0.38	-0.19	0.21	-0.07	-0.40	0.47	-0.16	0.00	-0.10
oneparent	0.00	-0.01	0.21	-0.38	-0.80	0.41	0.01	0.01	0.00	-0.03	0.03	-0.01	-0.01	0.01	0.01
disabilityu70	-0.31	-0.08	0.10	0.02	-0.08	-0.19	-0.41	-0.60	-0.40	-0.20	0.12	0.29	-0.11	0.06	0.07
sepdivorced	-0.28	-0.19	-0.11	-0.14	-0.04	-0.14	-0.63	0.34	0.40	0.14	-0.26	0.05	-0.25	0.10	-0.07
nocar	-0.19	0.35	-0.49	-0.03	-0.06	0.05	0.04	0.24	-0.16	-0.26	0.39	-0.01	-0.42	-0.21	-0.26
noedu	-0.15	0.49	0.28	0.24	-0.12	-0.12	0.00	-0.23	0.26	0.05	-0.25	-0.29	-0.26	-0.44	0.21
englishpoor	-0.08	0.59	0.18	0.19	-0.09	-0.06	-0.06	0.02	0.17	0.04	0.05	0.22	0.26	0.52	-0.39
labour	-0.34	-0.05	0.25	0.10	0.04	0.06	0.12	0.18	-0.05	0.66	0.45	0.28	-0.07	-0.14	0.15
drivers	-0.24	-0.07	0.51	0.08	0.14	0.02	0.05	0.47	-0.03	-0.61	0.03	0.14	0.05	-0.07	0.16
service	-0.19	-0.01	-0.07	-0.48	-0.11	-0.65	0.43	-0.07	0.24	-0.06	0.02	0.16	0.06	0.00	0.06
% variance explained	32%	14%	9%	7%	7%	6%	5%	4%	3%	3%	3%	2%	2%	1%	1%

Table 6.2 shows that the first principal component explains 32% of the variance. It also shows that the feature that contributes the most to the IRSD is nonet_perc (% occupied private dwellings with no Internet connection), meaning that the lack of an Internet connection at home is a strong indicator of socio-economic disadvantage.

Using the loadings for PC1 from Table 6.2, the raw IRSD score can be calculated for each SA1 area i as:

Equation 6.21
$$\boldsymbol{IRSD_i^{raw} = \sum_{j=1}^{15} \phi_{j1} x_{ij}}$$

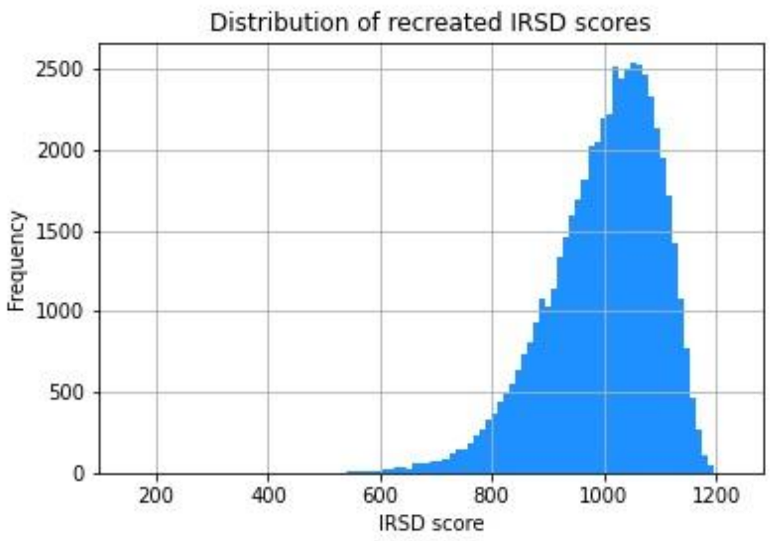
where ϕ_{j1} is the first principal component loading for the j^{th} feature and x_{ij} is the standardised value of feature j for SA1 i.

In computing the final IRSD scores for each SA1, for convenience, the ABS scales the raw scores so that the final IRSD values have a mean of 1,000 and a standard deviation of 100. Thus, the final IRSD values are given by:

Equation 6.22
$$\boldsymbol{IRSD_i = 1,000 + \frac{100}{stdev(IRSD^{raw})} IRSD_i^{raw}}$$

Figure 6.11 shows a histogram of scores for the 53,956 SA1s in the dataset, which is very similar to Figure 4.11 in the SEIFA technical report.

Figure 6.11 – Histogram of recreated IRSD scores



Finally, the replicated IRSD scores can be compared with the ABS's IRSD scores to see how well this case study has replicated the ABS's index. Figure 6.12 plots the ranking of each SA1 using the replicated IRSD scores from this case study against the ranking using the real IRSD scores reported by the ABS.

Figure 6.12 – Comparison of replication and ABS IRSD scores

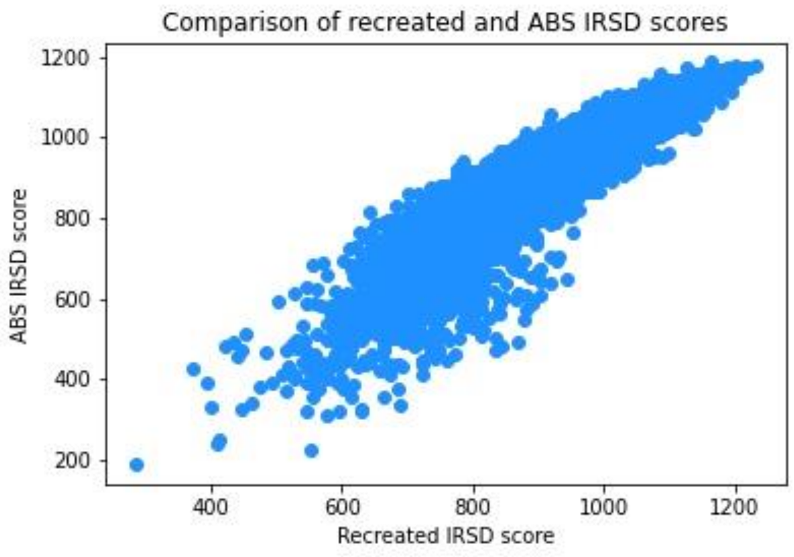


Figure 6.12 suggests this case study performed reasonably well in replicating the results from the ABS. The calculated correlation between the rankings is 0.96.

6.4.2. Case Study 2—Credit Card Fraud Detection

Define the Problem

Fraud involves deceptive actions that are taken with the intention of illegally or unethically gaining something at the expense of others. Some examples of fraud include:

- stealing another person's credit card details and using these to purchase goods;
- providing false information to an insurance company to obtain an insurance benefit that you are not entitled to; and
- using another person's personal information, without their permission, to commit a crime or to deceive or defraud that other person.

In July 2019, the Centre for Counter Fraud Studies at the University of Portsmouth estimated that fraud costs the global economy £3.89 trillion, with losses from fraud rising by 56% in the past decade.[12] Fraudulent behaviour can occur in any industry, including in many of the industries that actuaries typically work in, such as banking and finance.

[12] Gee, J., and Button, M. (2019). *The Financial Cost of Fraud 2019: The latest data from around the world.* https://www.crowe.com/uk/croweuk/insights/financial-cost-of-fraud-2019

Design a Solution

Fraud detection is a set of activities that are undertaken to identify fraud. Fraud detection can help to rectify the outcomes of any fraudulent behaviour and/or to prevent fraud from occurring in the future.

Video 6.5 discusses four different types of fraud detection solutions.

Video 6.5 – Fraud detection[13]

https://www.youtube.com/watch?v=pIfDlQt6HPo&feature=share (14 mins)

Record your video notes here

The fraud detection solutions that are discussed in Video 6.5 are:

- reputation list;
- rules engine;
- supervised machine learning; and
- unsupervised machine learning.

A **reputation list,** or ‘blacklist’, is a relatively static list of known fraudulent identities. For example, a bank’s ‘blacklist’ might include a list of individuals who have previously been convicted of credit card fraud. A drawback of this method is that it is difficult to keep the reputation list up-to-date and it often requires identities to have been found committing fraudulent behaviour in the past before they can be added to the list.

[13] The first 14 minutes of this video are the most relevant to this chapter. You can skip over the last 7 minutes of the video.

A **rules engine** approach to fraud detection involves running a set of rules over an activity. If the activity meets one of the rules, it might be further investigated manually. For example, a retailer might use a rules engine to flag any potentially fraudulent online purchases. They might, for example, flag for further investigation any transactions that are over a certain volume or that are being requested for delivery to a foreign country. While rules engines have the benefit of being easy to understand, they can also be hard to keep up-to-date with recent fraudulent activities.

A **supervised learning** approach to fraud detection involves training a computer to recognise future fraud by providing it with examples of past fraud. For example, a social media site might have a historical dataset that contains a list of new accounts and their attributes (‘features’) as well as a label to indicate whether each new account was opened by a legitimate customer (‘not fraud’) or by someone pretending to be a legitimate customer (‘fraud’). The classification techniques discussed in Chapter 5 could be used for this supervised learning approach to fraud detection.

Supervised learning, therefore, approximates a rules engine and might be used to create a rules engine. A drawback of this method is that it requires a large dataset of past examples of fraud, which can be very ‘expensive’ to obtain, as the exercise of validating historical instances of fraud can be very time-consuming. In addition, past examples of fraud may not be a good indicator of the types of fraud an organisation could be targeted by in the future. In fact, there should be some anticipation that fraudsters will change their strategies over time when they see that their existing methods of fraud are thwarted or investigated.

Unsupervised learning is described in Video 6.5 as being the cutting-edge solution to fraud detection. As discussed in this chapter, one branch of unsupervised learning, clustering, involves finding subsets of a population that are like each other and different from other subsets. Clustering can be used in fraud detection in one of two ways:

- by identifying outliers that are dissimilar to other observations and do not align closely with any of the clusters found in the dataset—these outliers are potential cases of fraud; or, conversely,
- by identifying a cluster of observations, when all other observations appear to be more random and not tightly bunched together—this method is described in Video 6.5.

You should now review the code contained in ‘DSA_C06_CS2.ipynb’. This code demonstrates the use of clustering techniques in fraud detection.

Monitor the Results

The output of fraud detection is a 'short-list' of possible outliers. Incorrectly accusing someone of fraudulent behaviour can have very detrimental financial and reputational impacts on an organisation, as well as likely causing the accused individual significant stress. For this reason, the results of a fraud detection activity usually need to be reviewed manually by a human to determine whether fraud has occurred or is likely to occur in each case. Further monitoring and/or investigation might be required before deciding whether fraud has occurred. Again, when conducting fraud investigations, it is important to consider the potential harm that can be caused to someone undergoing investigation.[14]

At the same time, letting fraud go undetected and uncorrected can also be very costly for an organisation. Therefore, it is important to track how well any fraud detection algorithm performs over time. As the nature of fraud changes over time, it is important to assess whether the algorithm is keeping pace with the changing environment.

[14] See, for example, the significant harm caused to an insurance customer who argued that the surveillance used against her was ‘deeply inappropriate’: Williams, R. (2018). ‘OMG…I want results’: TAL hired investigator to dig dirt on nurse. *Sydney Morning Herald.* https://www.smh.com.au/business/banking-and-finance/omg-i-want-results-tal-hired-investigator-to-dig-dirt-on-nurse-20180913-p503jf.html

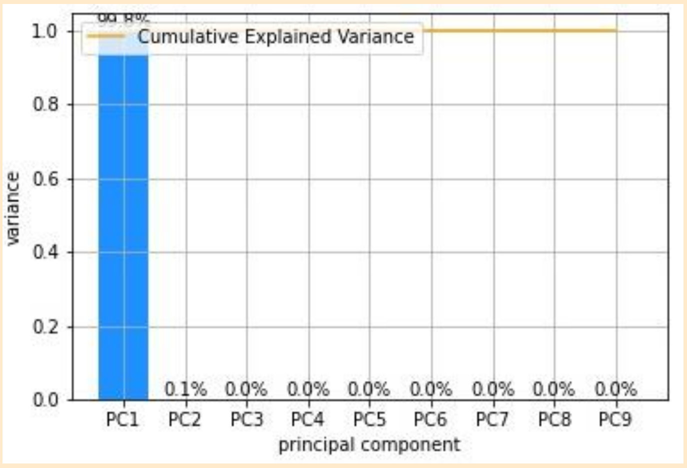
6.5. Key Learning Points

- Unsupervised learning involves understanding the properties of the distribution of data features, $P(X)$.
- Unsupervised learning, such as dimension reduction and clustering, does not rely on having response variables in the dataset.
- High-dimensional datasets are hard to visualise, often contain unimportant features, and can be difficult to work with.
- Dimension reduction techniques include feature selection and feature extraction.
- Principal component analysis (PCA) represents a high-dimensional dataset with a lower-dimensional dataset that explains most of the variability in the original data.
- Other feature extraction techniques exist that might be more suitable for the problem being solved.
- Clustering aims to separate observations into homogenous groups or clusters.
- Partitional clustering divides a set of observations in K 'partitions' of the data, where K is set by the user before modelling commences.
- K-means clustering aims to partition a dataset into K clusters by minimising the amount by which observations in a cluster differ from each other.
- Hierarchical clustering does not require the number of clusters to be pre-specified and, instead, builds a hierarchy of clusters, displayed in the form of a dendrogram.
- In both dimension reduction and clustering, it is often necessary to calculate the distance between two observations. Distance measures include Euclidean, correlation-based, simple matching, and Gower's distance.
- In hierarchical clustering, it is also necessary to calculate the distance between clusters. Cluster distance options include complete, single, average, and centroid linkage.
- The choice of distance or linkage measures can have a large impact on the outcome of a clustering exercise.
- Alternatives to k-means and agglomerative (bottom-up) hierarchical clustering include fuzzy c-means, divisive hierarchical clustering, density-based clustering, and scalable clustering algorithms.
- Three methods for validating clustering outcomes are internal, external, and manual validation.

6.6. Answers To Exercises

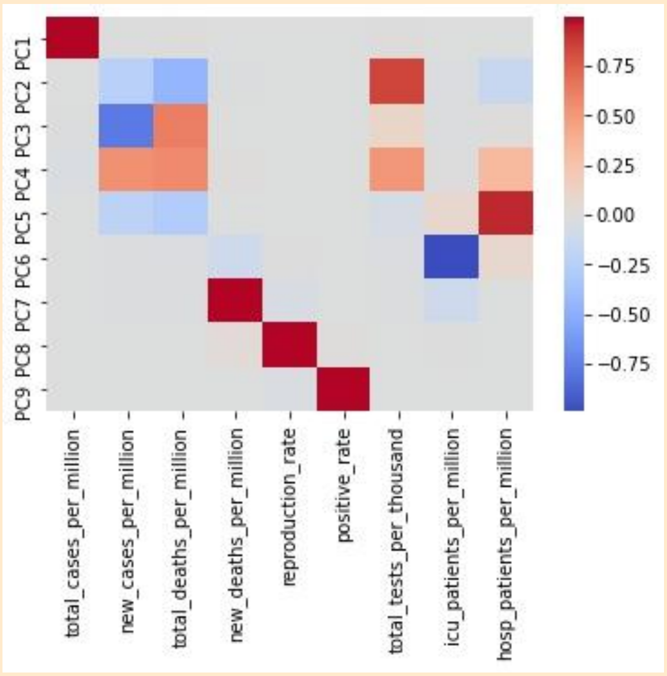
Answer to Exercise 6.1

When the original COVID data features are not scaled before performing PCA, the following scree plot is obtained:



This shows that almost 100% of the variance is explained by principal component 1 (PC1).

An inspection of the resulting heat map (below) can help to explain why this occurs:



Answer to Exercise 6.1

The new heat map shows that PC1 relies solely on the original feature 'total_cases_per_million'. This is because the scale of this feature is significantly higher than the scale of any of the other features. The average value for 'total_cases_per_million' is approximately 37,000. The next highest average value of a feature is 769 for 'total_tests_per_thousand'. Therefore, if scaling is not applied to the features before performing PCA, 'total_cases_per_million' drowns out all of the other features and all of the variance of the dataset is explained by the 'total_cases_per_million' feature.

However, the scale of features in this dataset does not dictate how important they are to understanding similarities and differences between the COVID experience of different countries. This helps to illustrate the importance of scaling data before you perform PCA and many other data science techniques.

Answer to Exercise 6.2

This dataset contains a mix of nominal features ('Name' and 'Gender') and continuous features ('Height' and 'Date of birth'). Gower's distance can be used for such a dataset, as it allows the distance formula to vary depending on the type of feature.

Distance between observations—Name

Equation 6.15 ($d(x_{ij}, x_{i'j}) = \delta(x_{ij}, x_{i'j}) = \begin{cases} 0, if\ x_{ij} = x_{i'j}, \\ 1, if\ x_{ij} \neq x_{i'j}. \end{cases}$) is used to calculate the distance between each pair of observations for the nominal feature 'Name'.

Distance for 'Name'	A	B	C
A		1	0
B	1		1
C	0	1	

The 'Name' of observation B ('Suzanne') is different to that of observation A ('Robert') and C ('Robert'). The distance measure between B and A and between B and C is, therefore, 1 for this field. As the 'Name' of observation A (Robert) is the same as that of observation C, the distance measure between these two observations is 0.

Answer to Exercise 6.2

Distance between observations—Gender

Equation 6.15 is also used to calculate the following distances between each pair of observations for the nominal feature ‘Gender’.

Distance for ‘Gender’	A	B	C
A		1	0
B	1		1
C	0	1	

Distance between observations—Height

Equation 6.14 ($d(x_{1j}, x_{2j}) = \frac{|x_{1j} - x_{2j}|}{R_j}$) is used to calculate the distance between each pair of observations for the continuous feature ‘Height’. For this feature, R = 0.19 (i.e. the difference between the minimum and maximum heights for this dataset).

Distance for ‘Height’	A	B	C
A		0.21	1.00
B	0.21		0.79
C	1.00	0.79	

Observations A and B are the most similar for the ‘Height’ feature as they have the smallest distance measure of 0.21.

Actuaries Institute.

Answer to Exercise 6.2

Distance between observations—Date of birth

Equation 6.14 is also used to calculate the distance between each pair of observations for the continuous feature 'Date of birth'. For this feature, R = 568 (i.e. the difference between the minimum and maximum dates of birth for this dataset, calculated in days).

Distance for 'Date of birth'	A	B	C
A		1.00	0.00
B	1.00		1.00
C	0.00	1.00	

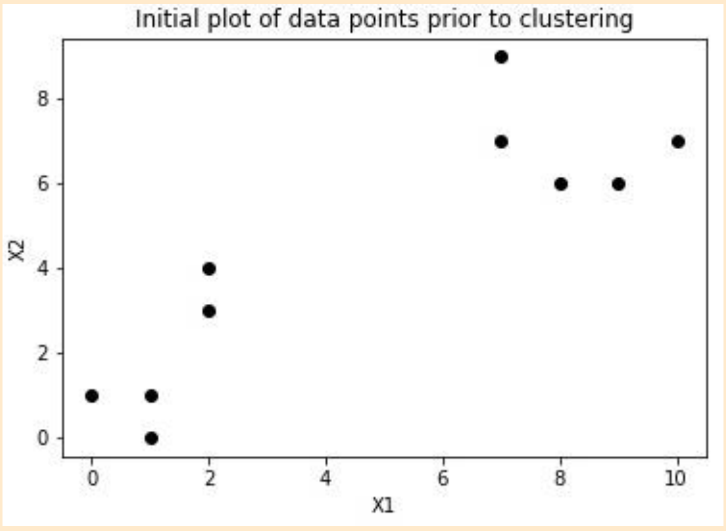
The total Gower's distance between each pair of observations can then be calculated using Equation 6.13 ($D_{gow}(X_{i.}, X_{i'.}) = \sqrt{\frac{1}{p}\sum_{j=1}^{p} d^2(x_{ij}, x_{i'j})}$). This formula combines the individual distances calculated for each feature, with the following results:

Gower's distance across all features	A	B	C
A		0.87	0.50
B	0.87		0.95
C	0.50	0.95	

Gower's distance suggests that observations A and C are the most similar, with a distance measure of 0.50. Observations B and C are the least similar, with a distance measure of 0.95.

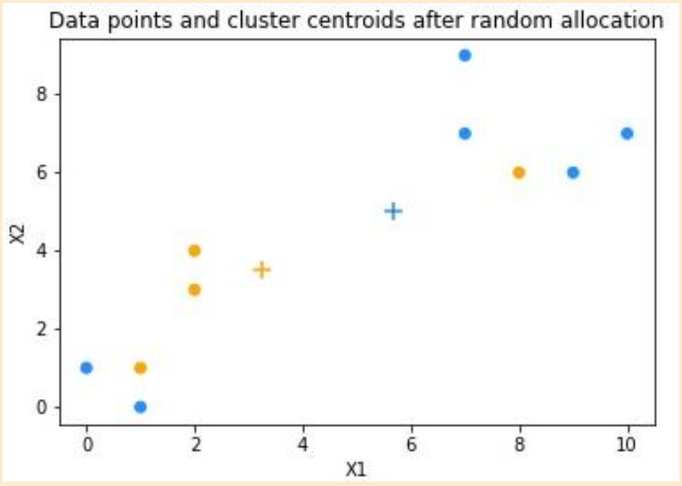
Answer to Exercise 6.3

The following is a scatterplot of the 10 data points prior to any clustering being performed:



It is clear from the scatterplot above that the data points fall into one of two distinct clusters.

To use k-means clustering to recover those two clusters, each data point is initially randomly assigned to cluster 1 (C_1) or cluster 2 (C_2). One possible random allocation is shown in the following scatterplot, with data points assigned to C_1 shown in blue and those in C_2 shown in orange.

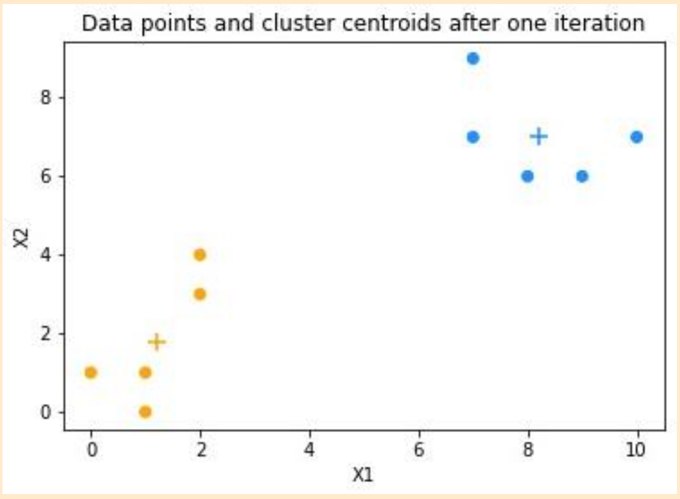


Answer to Exercise 6.3

The cluster centroids, Z_1 and Z_2, for this initial random allocation are shown in the above diagram as crosses. These were calculated as follows:

- Z_1 = (average $X_{1.}$ value for data points in C_1 , average $X_{2.}$ value for data points in C_1); and
- Z_2 = (average $X_{1.}$ value for data points in C_2 , average $X_{2.}$ value for data points in C_2).

The next step is to re-assign each data point in X to the cluster with the nearest centroid. This is shown in the following diagram:



The cluster centroids, Z_1 and Z_2, have been recalculated based on the new allocation of data points to each cluster. These updated centroids are again shown in the diagram as crosses.

For this clustering exercise, there is no need to complete more than one iteration, as each data point is now already assigned to the closest updated centroid.

The Python code used to complete this exercise can be found in 'DSA_C06_Ex3.ipynb'. With this simple dataset, a spreadsheet could also easily be used to perform the clustering.

Answer to Exercise 6.4

The number of clusters can be specified in the code provided by updating the value for k (it is initially set to 3).

The k-means algorithm produces the following clusters for different values of k:

Cluster	$k = 2$	$k = 3$	$k = 4$	$k = 5$
1	Austria Belgium Bulgaria Italy Portugal Slovenia Spain UK USA	Austria Belgium Bulgaria Italy Portugal Slovenia Spain UK USA	Austria Belgium Bulgaria Italy Portugal Slovenia Spain UK USA	Austria Belgium Israel Italy Portugal Spain UK
2	Canada Cyprus Denmark Estonia Finland Ireland Israel Luxembourg	Canada Cyprus Denmark Estonia Finland Ireland Israel	Canada Cyprus Denmark Finland	Canada Cyprus Denmark Estonia Finland Ireland
3	n/a	Luxembourg	Estonia Ireland Israel	USA Slovenia
4	n/a	n/a	Luxembourg	Bulgaria
5	n/a	n/a	n/a	Luxembourg

Answer to Exercise 6.4

Based on the data used for this analysis, the table above suggests that the following countries have had similar COVID-19 experiences as they have been clustered together under each of the values of k that were explored in this exercise:

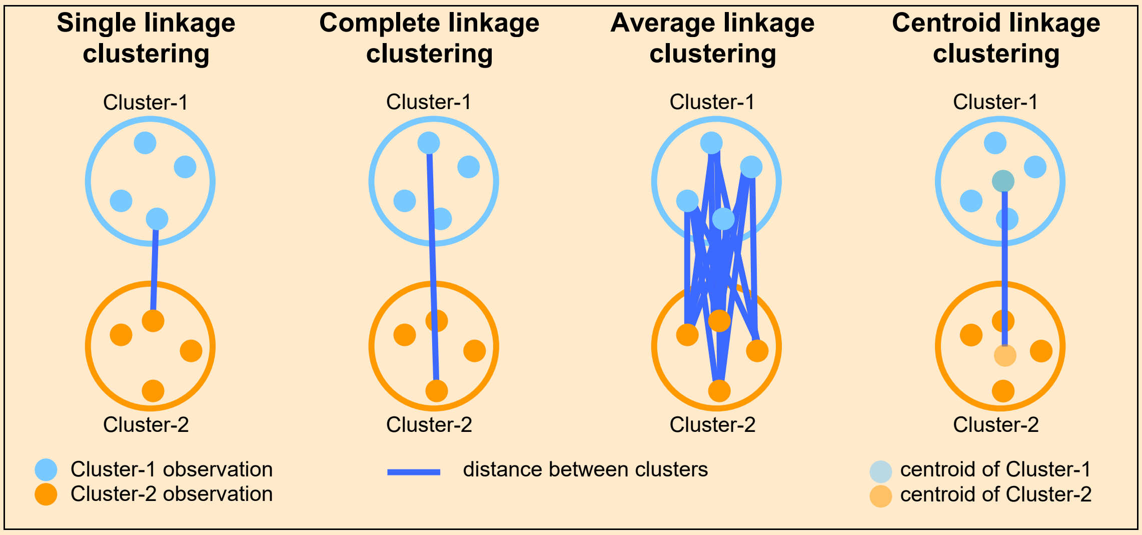
- Austria, Belgium, Italy, Portugal, Spain, and the UK;
- Canada, Cyprus, Denmark, and Finland; and
- the USA and Slovenia.

Luxembourg appears to be an outlier, clustering on its own when $k \geq 2$.

The observations above are reasonably consistent with those seen in Figure 6.6—that is, the scatterplot of the data on its first two principal components.

Answer to Exercise 6.5

The image below[15] is helpful in understanding the outcomes under single linkage compared to those under complete, average, or centroid linkage methods.


Single linkage clustering
Complete linkage clustering
Average linkage clustering
Centroid linkage clustering
Cluster-1
Cluster-2
Cluster-1 observation
Cluster-2 observation
distance between clusters
centroid of Cluster-1
centroid of Cluster-2

[15] Adapted from Babu, M.M. (2015). An Introduction to Microarray Data Analysis. p. 242.

Answer to Exercise 6.5

The complete linkage method (second image from the left) focuses on the MAXIMUM pairwise distances between observations in two clusters. This makes the method very sensitive to outliers within a cluster and the overall structure of each cluster. This method has the impact of merging, at each step, the two clusters that result in the smallest diameter of the new merged cluster. This results in tight clusters that have small diameters (referred to as 'balanced or ball-shaped' in Section 6.3.3).

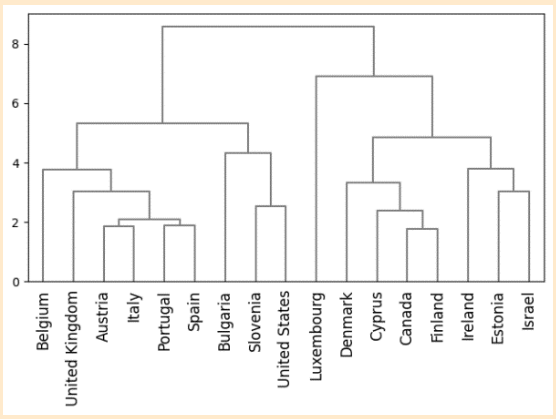
Average linkage (third image from left) and centroid linkage (image on right) are also impacted in this way by outliers in a cluster and the spread of observations in a cluster, but to a lesser extent than under the complete linkage method since they focus (each in slightly different ways) on average rather than maximum distances between observations in different clusters.

In contrast, the single linkage method (image on left) focuses on the MINIMUM pairwise distances between observations in two clusters. This makes this method insensitive to outliers within a cluster and insensitive to the overall structure of each cluster. You can see from the left most image above that the single linkage method will calculate the same distance between Cluster-1 and Cluster-2, even if all observations except the two with the shortest pairwise distance are spread further out. This method can therefore result in long, straggly clusters with large diameters (referred to as 'extended, string-like clusters' in Section 6.3.3).

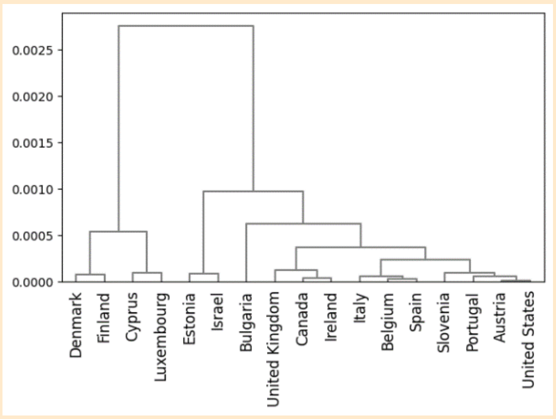
Answer to Exercise 6.6

The following two dendrograms both use complete linkage to measure distance between clusters but use different approaches (Euclidean and correlation-based) to measure distance between individual observations:

Euclidean distance



Correlation-based distance



The clusters obtained using a correlation-based distance measure are very different to those obtained using Euclidean distance. For example, under Euclidean distance, Denmark and Canada join into the same cluster at a much earlier point than under correlation-based distance. As another example, Luxembourg is a clear outlier using Euclidean distance, but not when using correlation-based distance.

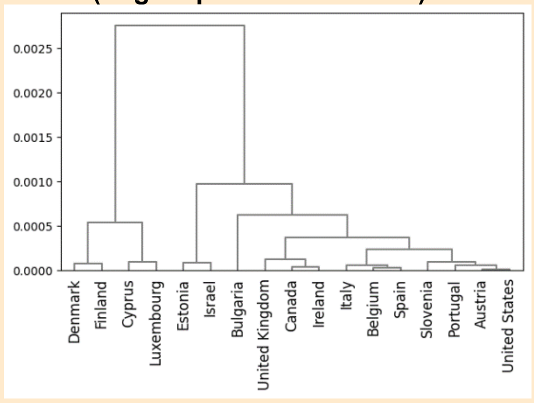
Remember that correlation-based distance considers two observations to be similar if their features are highly correlated, even if the Euclidean distance between the two observations is large. It is, therefore, important to consider the context of the clustering when deciding which distance measure to choose.

In this example, the features used are all rates (e.g. total cases per million people or reproduction rate per infected person). As we are most interested in the absolute value of each of these rates, rather than their size relative to each of the other rates, Euclidean distance is more appropriate.

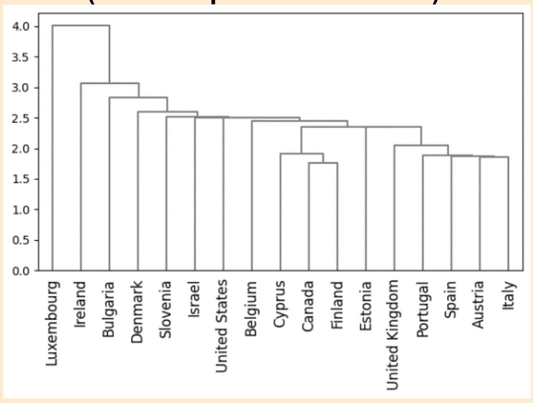
Answer to Exercise 6.6

The following four dendrograms both use Euclidean distance to measure the distance between observations but use different approaches (complete, single, average, and centroid linkage) to measure the distance between clusters:

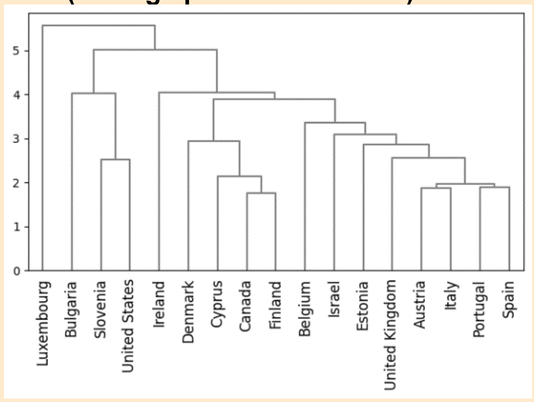
Complete linkage
(largest pairwise distance)



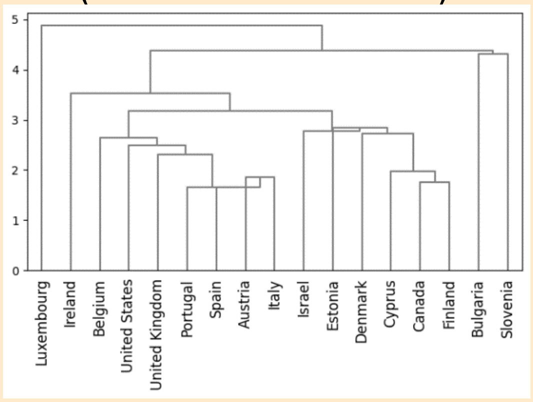
Single linkage
(smallest pairwise distance)



Average linkage
(average pairwise distance)



Centroid linkage
(distance between centroids)



The linkage method used also had a significant impact on the clusters obtained for this sample dataset. In particular, the complete linkage method gives quite different clusters to those under the other three linkage methods. For example, Luxembourg is a clear outlier under single, average, and centroid linkage methods, but not under the complete linkage method. As another example, Austria and Italy join together into a cluster early under the single, average, and centroid methods, but they join together further up the dendrogram under complete linkage.

The figure for the single linkage method (top-right plot) provides a nice illustration of the 'string-like' clusters that can arise under this linkage method, as discussed in Exercise 6.5.

Answer to Exercise 6.7

n/a

Actuaries Institute.

About the Actuaries Institute

The Actuaries Institute is the sole professional body for actuaries in Australia. The Institute provides expert comment on public policy issues where there is uncertainty of future financial outcomes. Actuaries have a reputation for a high level of technical financial skills and integrity. They apply their risk management expertise to allocate capital efficiently, identify and mitigate emerging risks and to help maintain system integrity across multiple segments of the financial and other sectors. This expertise enables the profession to comment on a wide range of issues including life insurance, health insurance, general insurance, climate change, retirement income policy, enterprise risk and prudential regulation, finance and investment and health financing.

Published June 2024

Institute of Actuaries of Australia

ABN 69 000 423 656
Level 2, 50 Carrington Street,
Sydney NSW 2000, Australia
t +61 (0) 2 9239 6100
f +61 (0) 2 9239 6170
actuaries@actuaries.asn.au
www.actuaries.asn.au