STAT5003

Week 4: High-dimensional visualisation and analytics in R

Jaslene Lin
The University of Sydney

Readings and R functions covered

Important

- **Introduction to Statistical Learning**
- PCA Dimension reduction, see Section 10.2 - Clustering, see Section 10.3
- How to use t—SNE effectively
- **The Elements of Statistical Learning**
    - MDS, see Section 14.8
- **R** functions
- `hclust` (hierarchical clustering)
- `kmeans` (K-means clustering)
- `dist` (distance matrix)
- `Rtsne::rtsne` (t-SNE)
- `cmdscale` (Multi-Dimensional Scaling)

Unsupervised Learning

Supervised Learning: Given a dataset with features/predictors $(X_{i1}, X_{i2}, ...X_{ip})$ and a target outcome variable (Y_i), the goal is to learn a **model** $f(X_{i1}..X_{ip})$ to explain/predict the target.

- example: Is there a relationship between `BuildingArea` and `Price` of houses in St Kilda?

Unsupervised learning: Given a dataset with features $(X_{i1}, X_{i2}, ...X_{ip})$, the goal is to visualise, find patterns, subgroups/clusters within the data.

- example: Can we identify different types of houses in St Kilda based on their features?

Clustering

Clustering Overview

- Groups similar observations based on **a similarity measure**

Clustering Goals

- **Compact Clusters**: Observations within a cluster should be **close together**.
- **Well-Separated Clusters**: Observations from different clusters should be **far apart**.
- Example algorithms:
  - **Hierarchical clustering**
  - K**-means clustering**
  - Gaussian mixture model

Typical methods

- Partitioning
  - Pre-specified number K of mutually exclusive and exhaustive groups
  - Iterate until criteria is met
- Hierarchical methods:
  - Agglomerative: Bottom up, more popular
  - Divisive: Top down, less popular
  - Display results with dendrogram

K-Means Clustering

Measuring Cluster Compactness

- For cluster C_k, we can define within-group sum of squares as:

$$\text{WSS}_k = \frac{1}{|C_k|} \sum_{i,j \in C_k} ||x_i - x_j||^2, \quad k = 1, \ldots, K$$

- This is the sum of all the pairwise squared Euclidean distances between observations in the k^{th} cluster, divided by total number of observations in the k^{th} cluster
- $\sum_{i,j \in C_k} ||x_i - x_j||^2$ is the sum of all the pairwise squared Euclidean distances between observations in the k^{th} cluster.
- $|C_k|$ is the total number of observations in the k^{th} cluster

K-Means Clustering

Clustering Objective

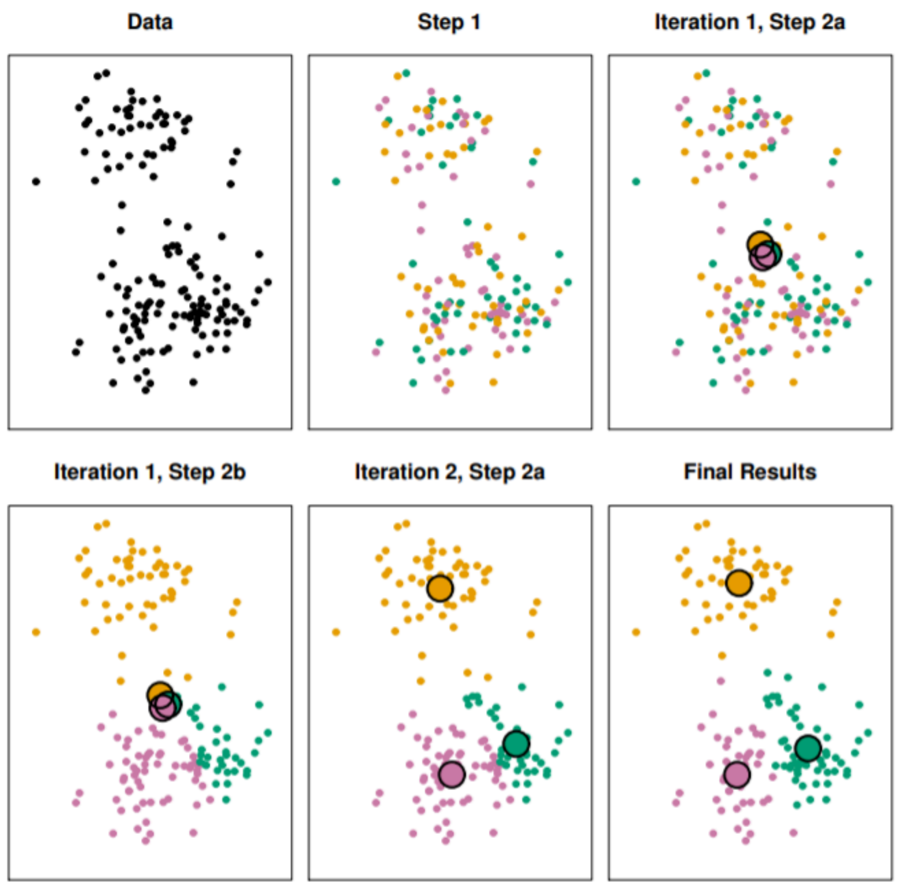
- To **minimize** the total within-group sum of squares criterion

$$\mathrm{WSS}_{\mathrm{Total}} = \sum_{k=1}^{K} \mathrm{WSS}_k$$

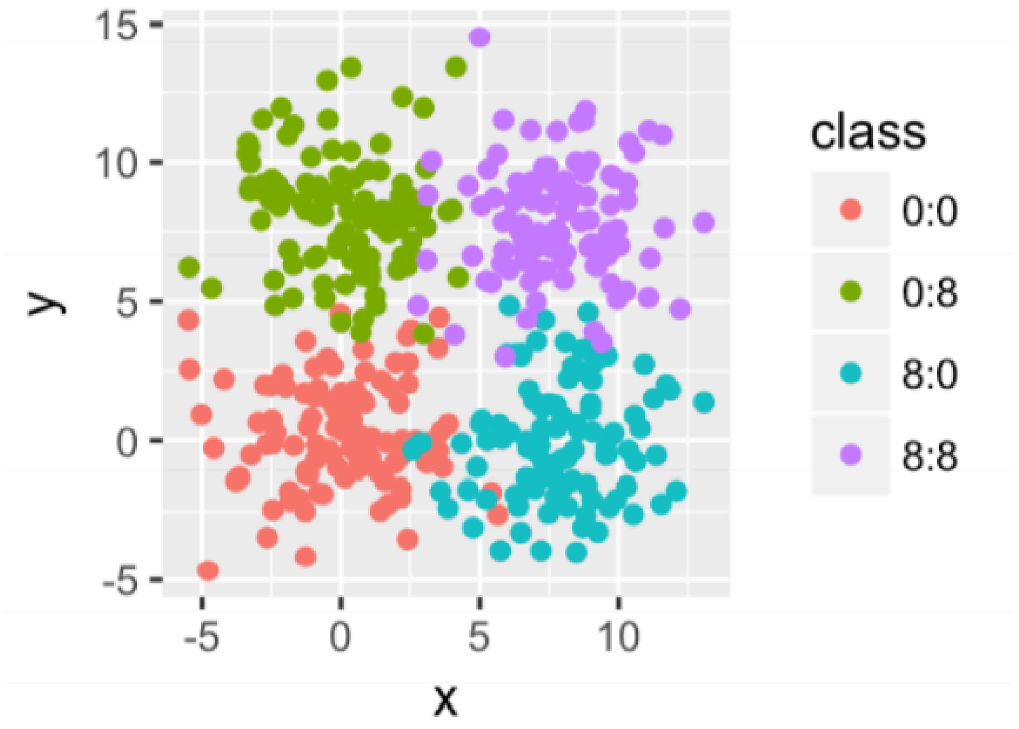
- **The total within sum of square criterion will decrease as** K increases
- Rule of thumb: Look for the elbow

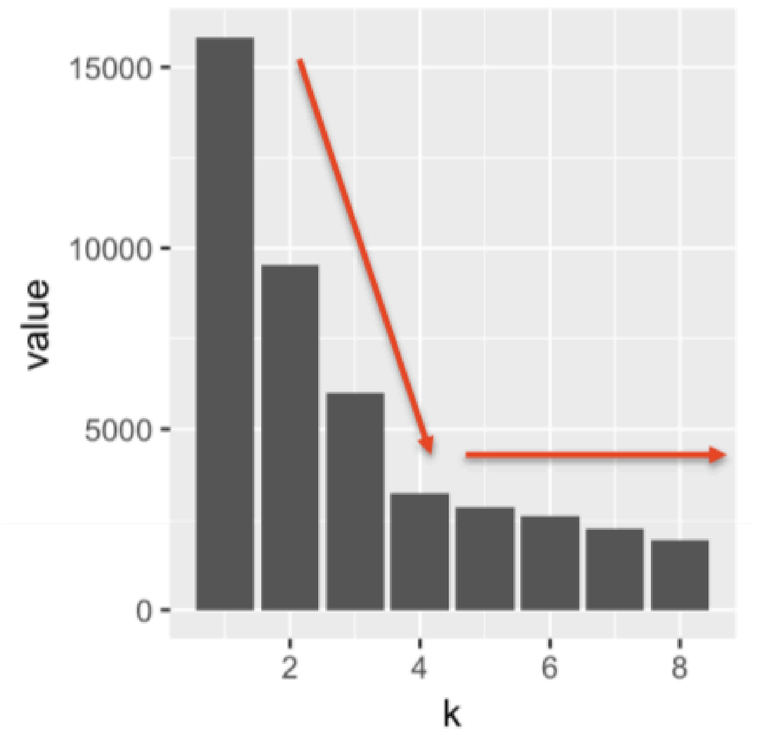
K-Means Clustering

- Initialise each observation at random to a cluster.
- Iterate the following until the assignments stop changing:
    1. For each of the K clusters, compute the cluster centroid.
    2. Assign each observation to the cluster whose centroid is closest (where the *closest* is defined using the Euclidean distance)



Choosing K via Elbow plot



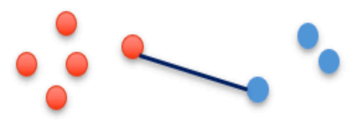


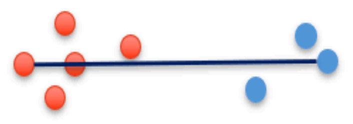
K-Means properties

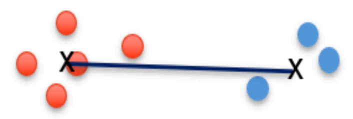
- The number of clusters K needs to be specified.
- Local solution and not necessarily global solution.
- Depends on starting values (the random starting values)-run the algorithms multiple times.
- Best for *compact*, *spherical* clusters.
- Does not work well *when cluster sizes are different*.

Hierarchical clustering

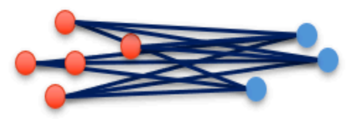
- Begin with every observation representing a single cluster
- At each iteration, merge the two closest clusters into one cluster
  - ➟ Needs a measure of **similarity/dissimilarity between two clusters**
  - ➟ These measures are called ***linkages***.
- Linkages: Measure of dissimilarity between two sets of objects that determine how two set of objects are merged.
  - ➟ Single linkage
  - ➟ Complete linkage
  - ➟ Average Linkage


Single (minimum)

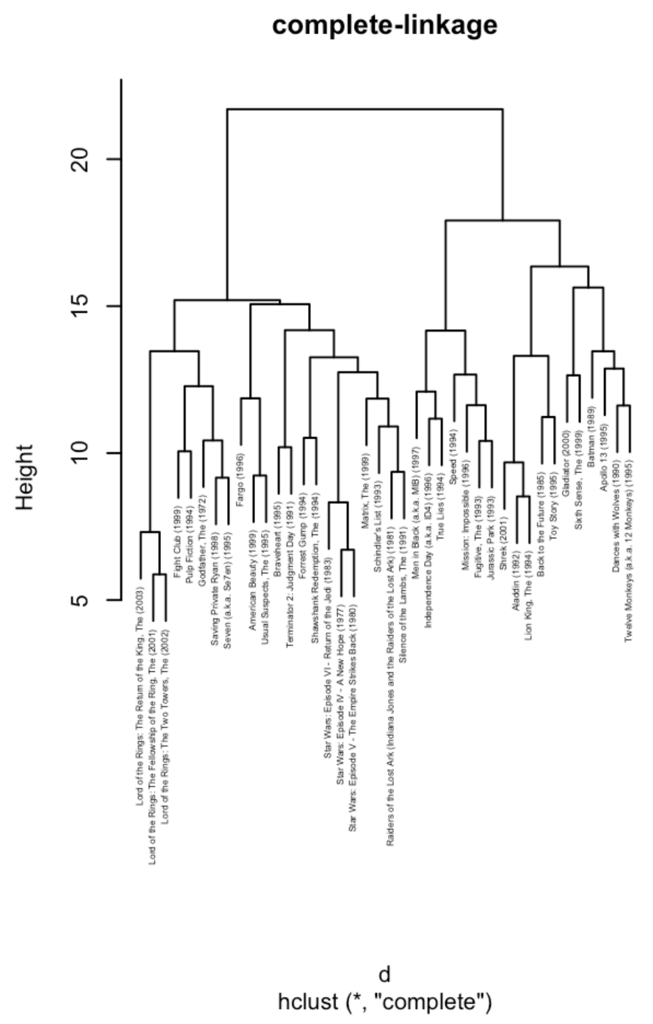

Complete (maximum)

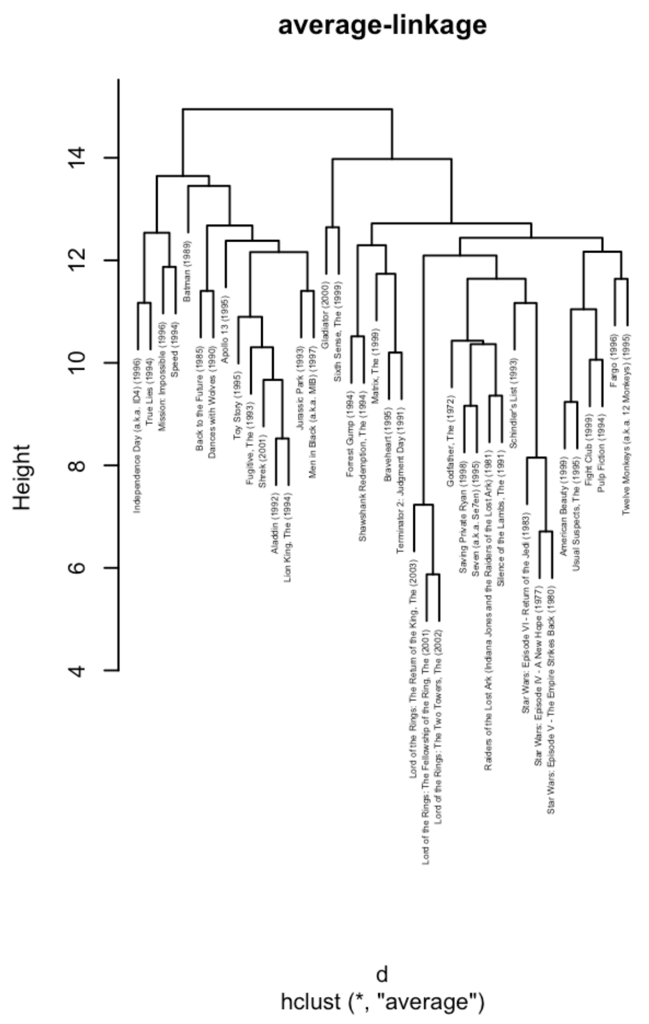


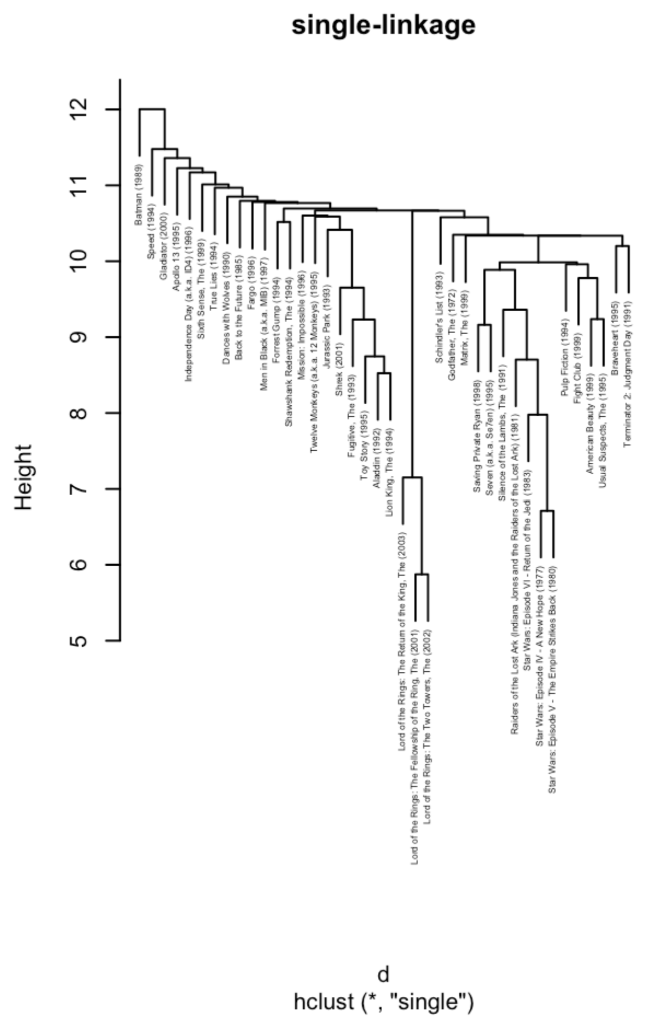
Distance between centroids


Average (mean) linkage

Hierarchical clustering

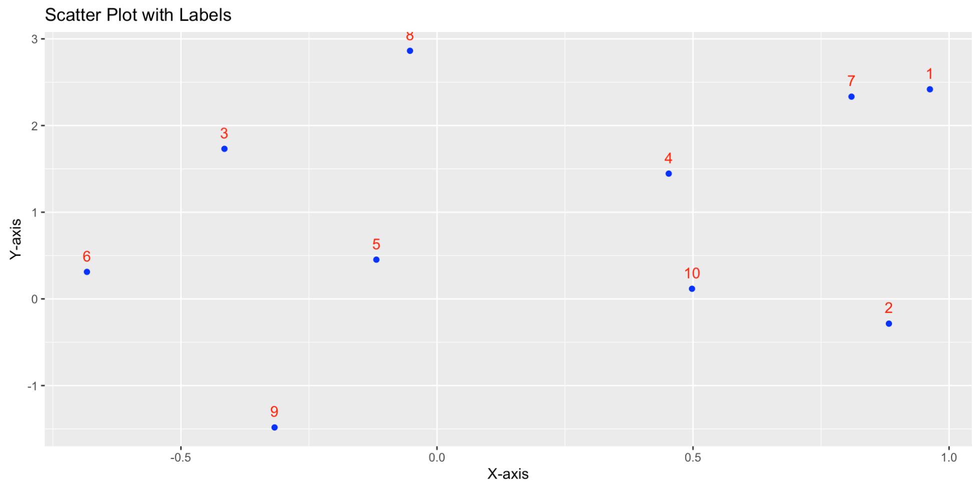


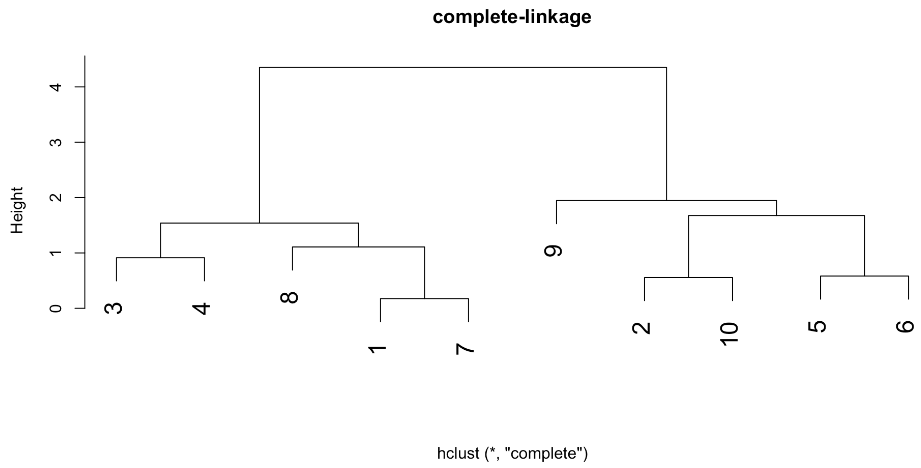




Hierarchical Clustering

Dendrogram





Dimension reduction: Principal Components Analysis (PCA)

High dimensional data

- High-dimensional data refers to data set with **more features** p than observations n
    - ➟ Examples: in genetic data, we can easily measure 500k individual DNA mutations (human genome have ~3 billion base pairs of DNA), but experiments generally have < 1000 people, e.g., $p \sim 500\mathrm{k}$, $n \sim 1000$
- It is very hard to visualize high-dimensional data
    - ➟ Only have 2 (sometimes 3 or 4) dimensional canvas to create plots
- Many algorithms and methods have been designed for low dimensional data and would not work well for high-dimensional data
- To build a linear regression model data with $500\mathrm{k}$ features will result in $500\mathrm{k}$ parameters. This problem is underdetermined if we only have 1000 observations

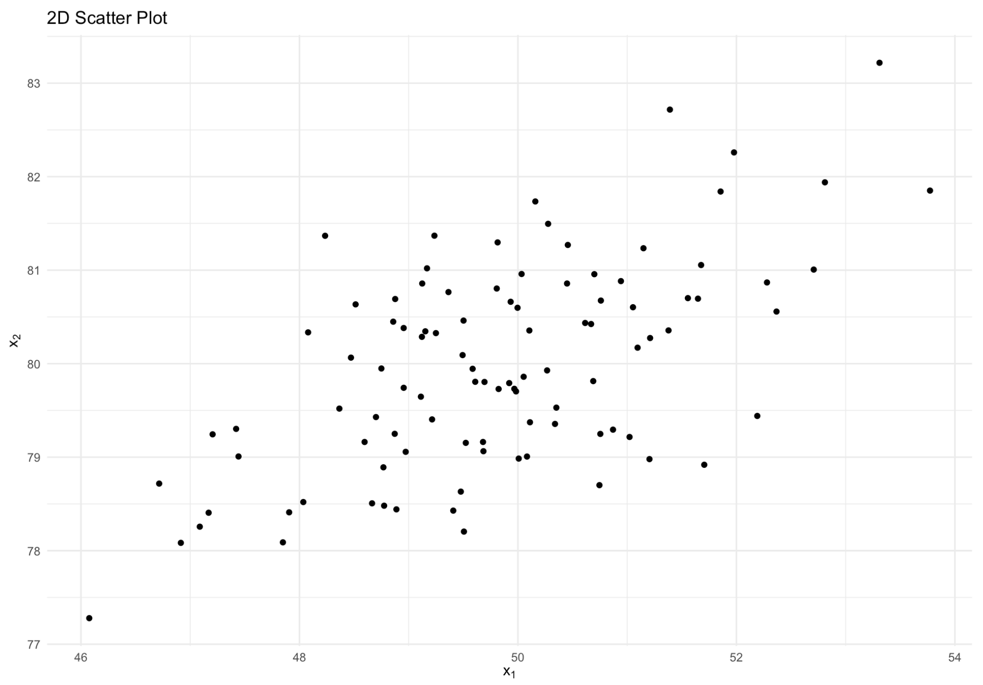
Dimension reduction strategies

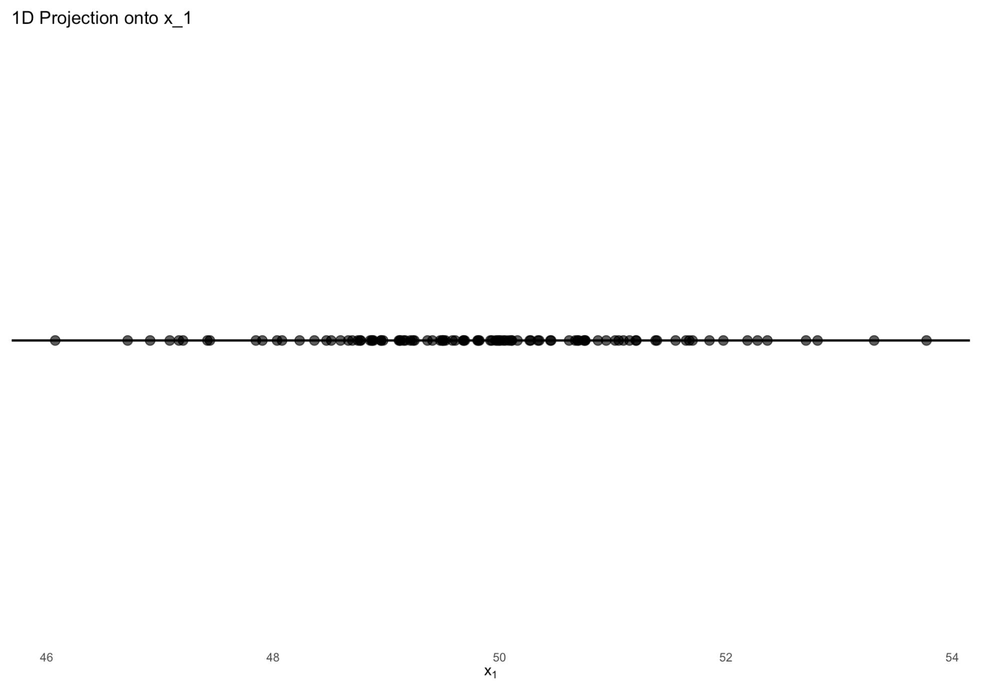
- Eliminate or remove features
  - ➡ Need to decide which features to be eliminated? Keep ones with high variance?
- Select features
  - ➡ Example: Lasso and ridge regression (coming soon in later module)
- Build or construct new features from existing ones
  - ➡ Replace many existing features with a single one
  - ➡ PCA and t-SNE

PCA

- Suppose we have a data matrix X with n observations and p features
  - ➟ Can we plot the data in a 2-dimensional plot?
- Naively, we can do all pairwise combinations, i.e., 1 vs 2, 1 vs 3, $\ldots, (p$ vs $(p-1))$
  - ➟ $\binom{p}{2} = \frac{p(p-1)}{2} = \mathcal{O}(p^2)$ different plots!
- Principal Component Analysis (PCA) helps us reorganise/transform the data into a new coordinate system.
  - ➟ The data still has p dimensions, but in a new set of axes.
  - ➟ The goal is to capture most of the variation using just the first few dimensions, making the data easier to analyse.

Best way to represent 2d in 1d?





- Could use a single variable? x_1 say?
- Or could remap x_1 and x_2 to a single variable
  - ➡ $z = \phi_1 x_1 + \phi_2 x_2$
- Can generalize this to many dimensions
  - ➡ $z = \sum_{j=1}^{p} \phi_j x_j$
- Pick the transformation that **maximises the variance!**

Principal components

Start with a data matrix $\boldsymbol{X}$, and assume it has **mean zero**

$$\boldsymbol{X} = (X_1, X_2, \ldots, X_p)$$

The first principal component is the **normalised linear combination of the features** that **maximises the variance** in the new component

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \cdots + \phi_{p1}X_p = \sum_{j=1}^{p} \phi_{j1}X_j = \boldsymbol{X}\boldsymbol{\phi}_1, \quad \boldsymbol{\phi}_1 = (\phi_{11}, \ldots, \phi_{p1})^T$$

The elements ϕ_{j1} are known as the loadings of the first principal component

- By normalised, we mean the squared loadings have to sum to 1, i.e. $\sum_{j=1}^{p} \phi_{j1}^2 = 1 \Leftrightarrow \boldsymbol{\phi}_1^T \boldsymbol{\phi}_1 = 1$
- It is desired to maximise the sample variance of $Z_{i1}, \ldots, Z_{n1}$, where Z_{i1} is the ith component of Z_1:

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

Principal component scores

- Given the principal component loadings, we can project our data matrix $\boldsymbol{X}$ onto the principal component space
  - ➠ The projection is a linear combination of the sample feature values:

$$z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \cdots + \phi_{p1}x_{ip}$$

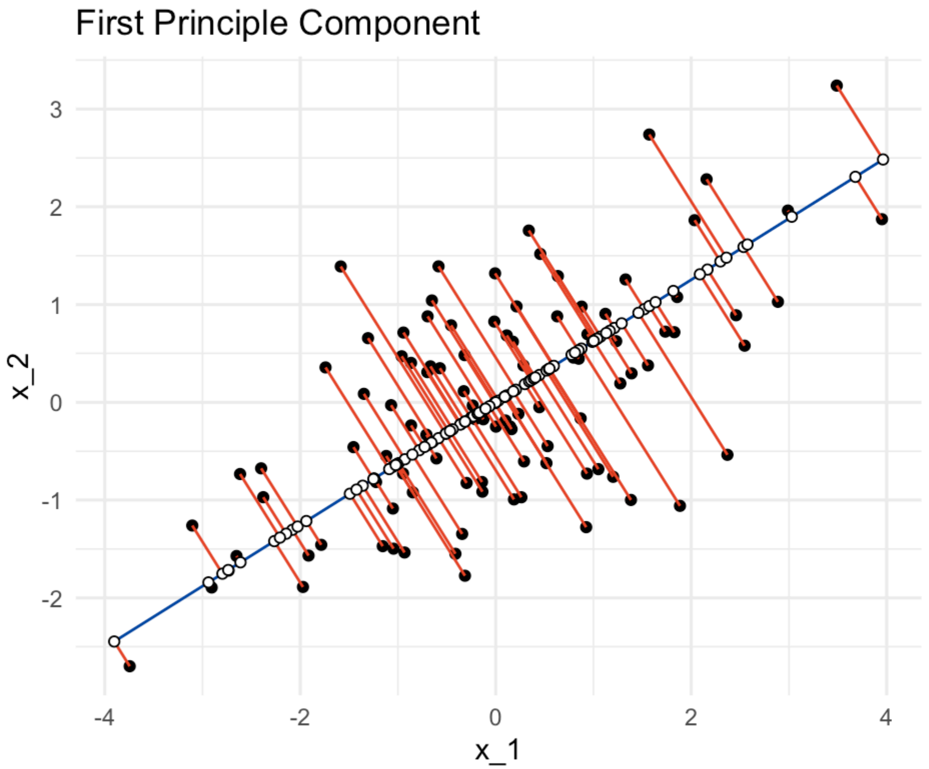
    This is known as the principal component score

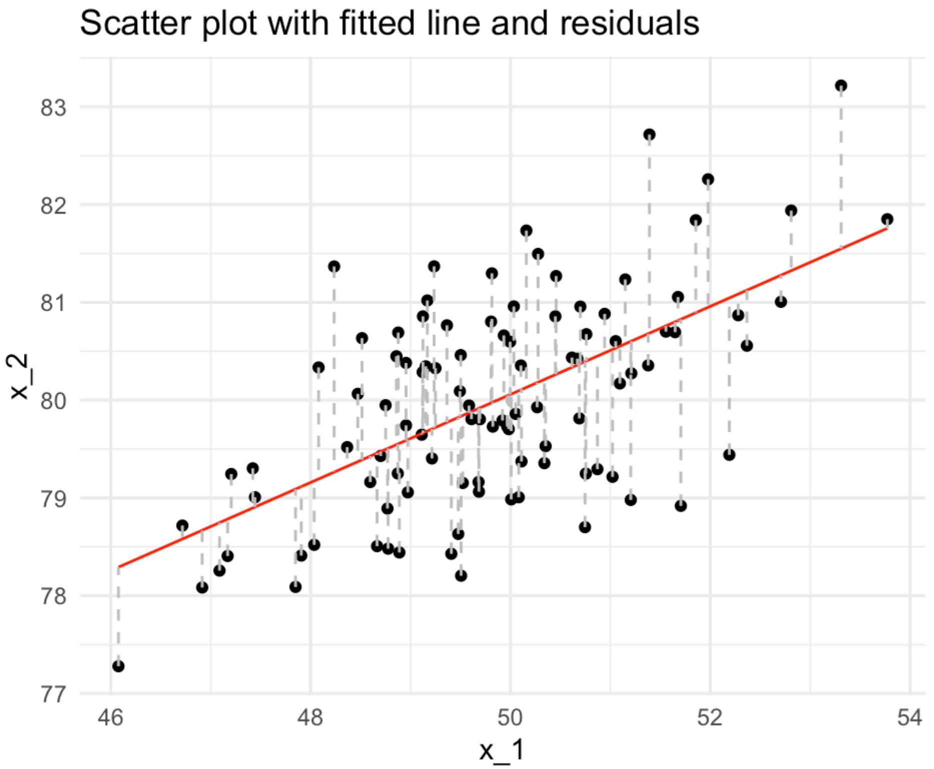
- The first principal component score vector is

$$\boldsymbol{Z}_1 = (z_{11}, z_{21}, \ldots, z_{n1})$$

  The principal component score vectors are uncorrelated (orthogonal)

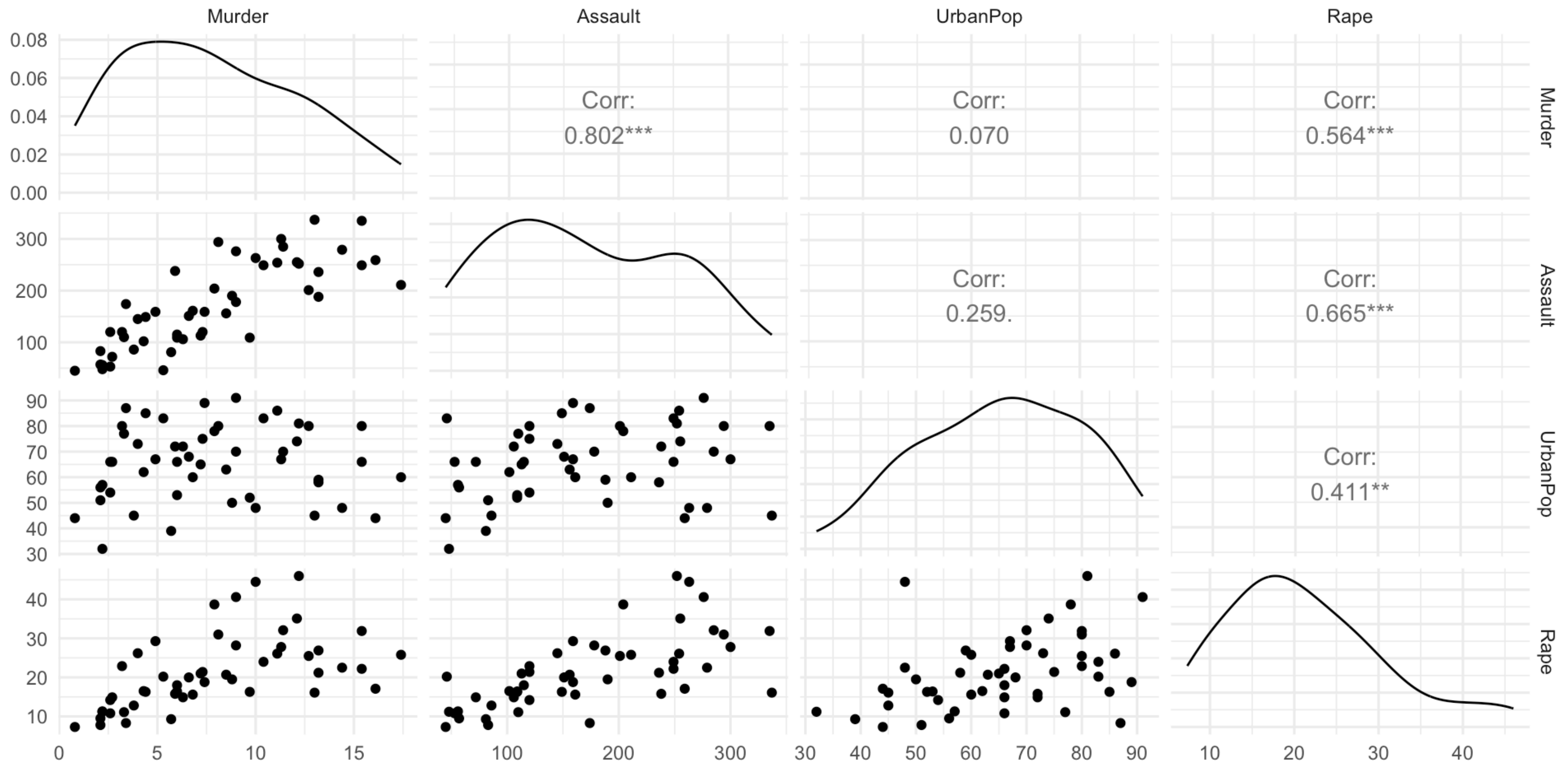
Geometric interpretation


First Principle Component
x_2
x_1
3
2
1
0
-1
-2
-4
-2
0
2
4


Scatter plot with fitted line and residuals
x_2
x_1
83
82
81
80
79
78
77
46
48
50
52
54

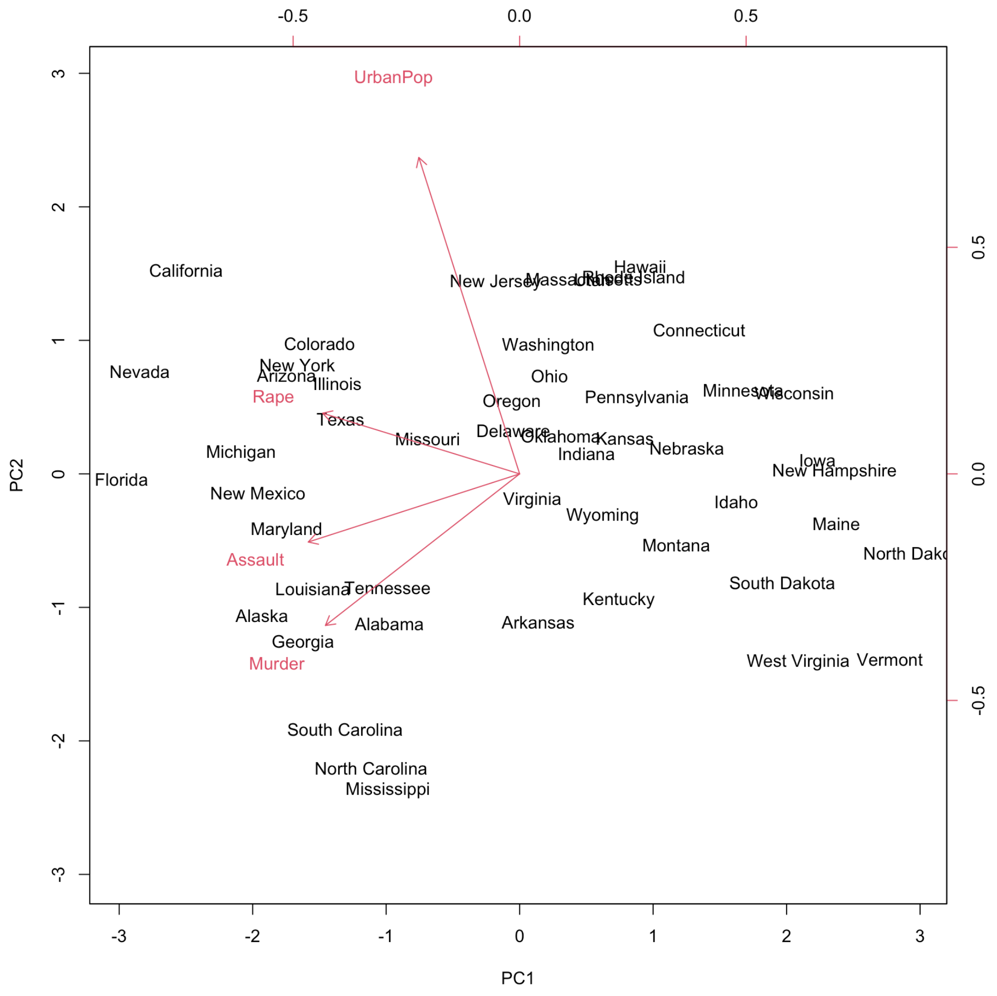
USArrests Example

```r
library(GGally)
ggpairs(USArrests) + theme_minimal()
```



Biplot of the USArrests

- ▶ Code



Preprocessing for PCA

- In **PCA**, it's common to **center** variables by subtracting the mean.
- You can also **standardise** them so all variables have a **standard deviation of 1**.

Why Standardisation Matters

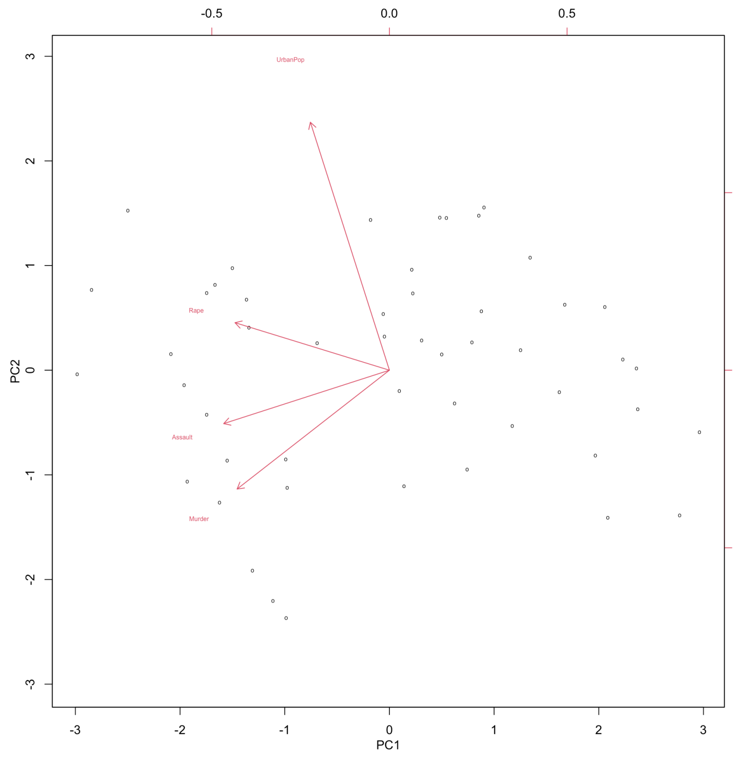
- Variables may have **different units**
- This leads to **different variances**, affecting PCA results.

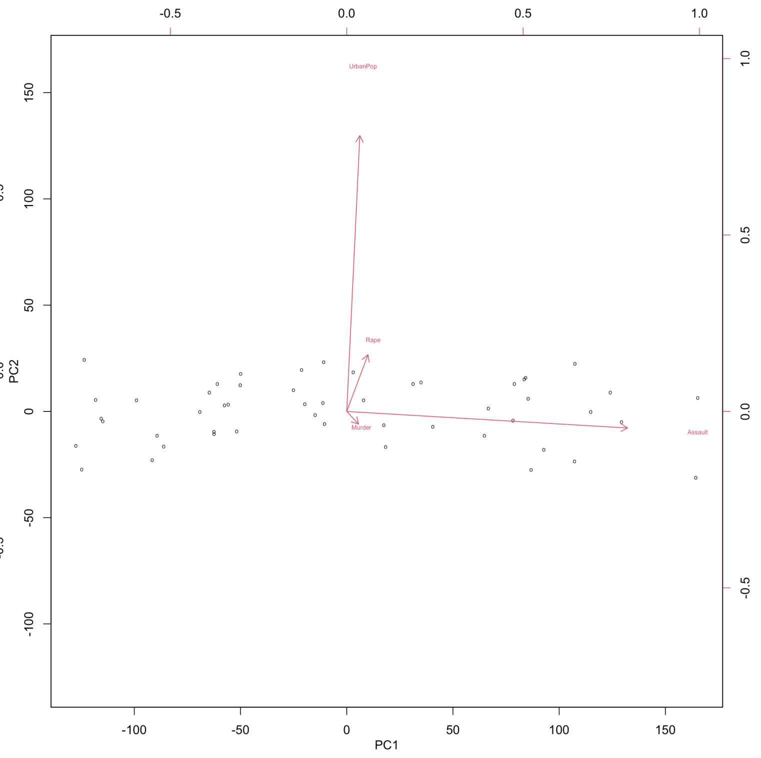
Impact on Loadings

- **PCA loadings** give more weight to variables with **higher variance**.
- This can **bias** the analysis if some variables dominate.

Solution: Standardisation

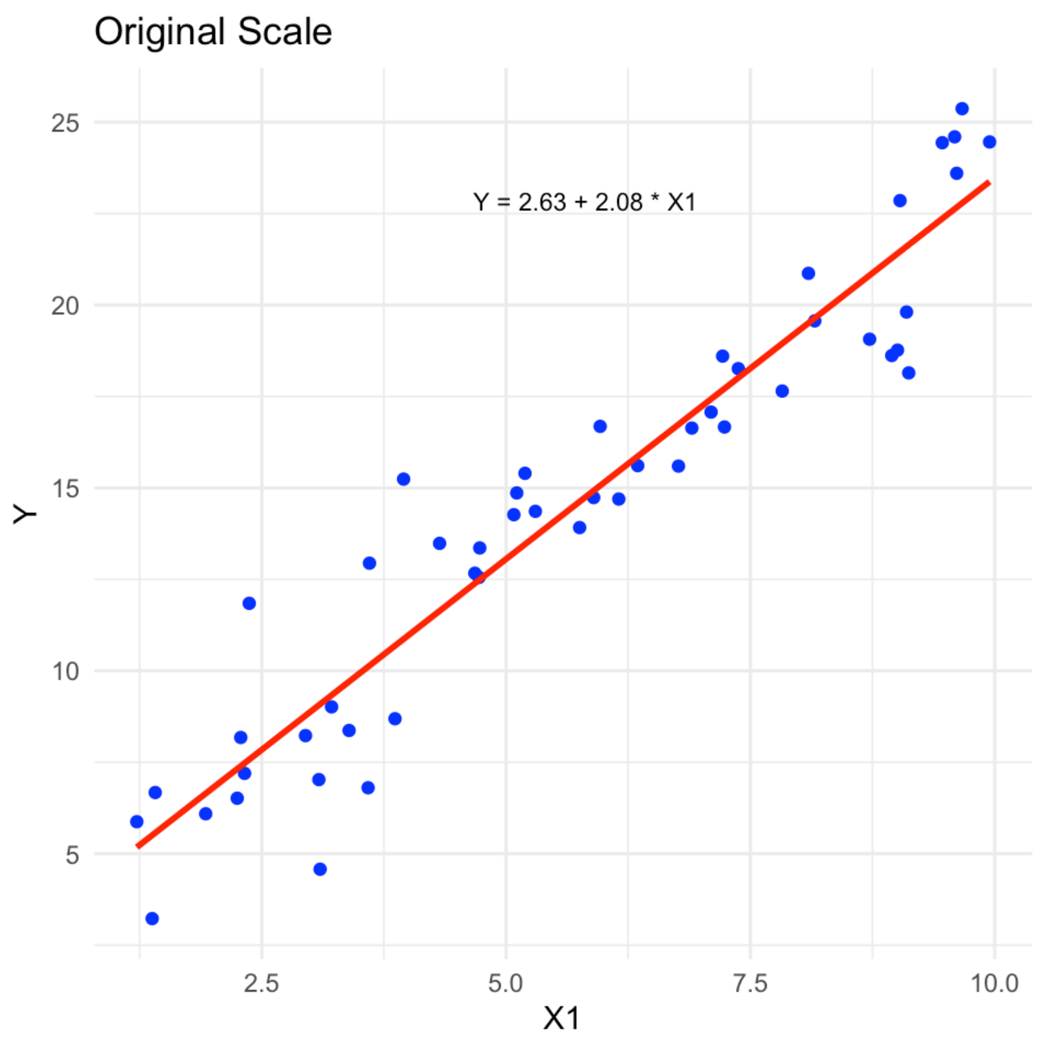
Effect of scaling (left) vs unscaled (right) in PCA

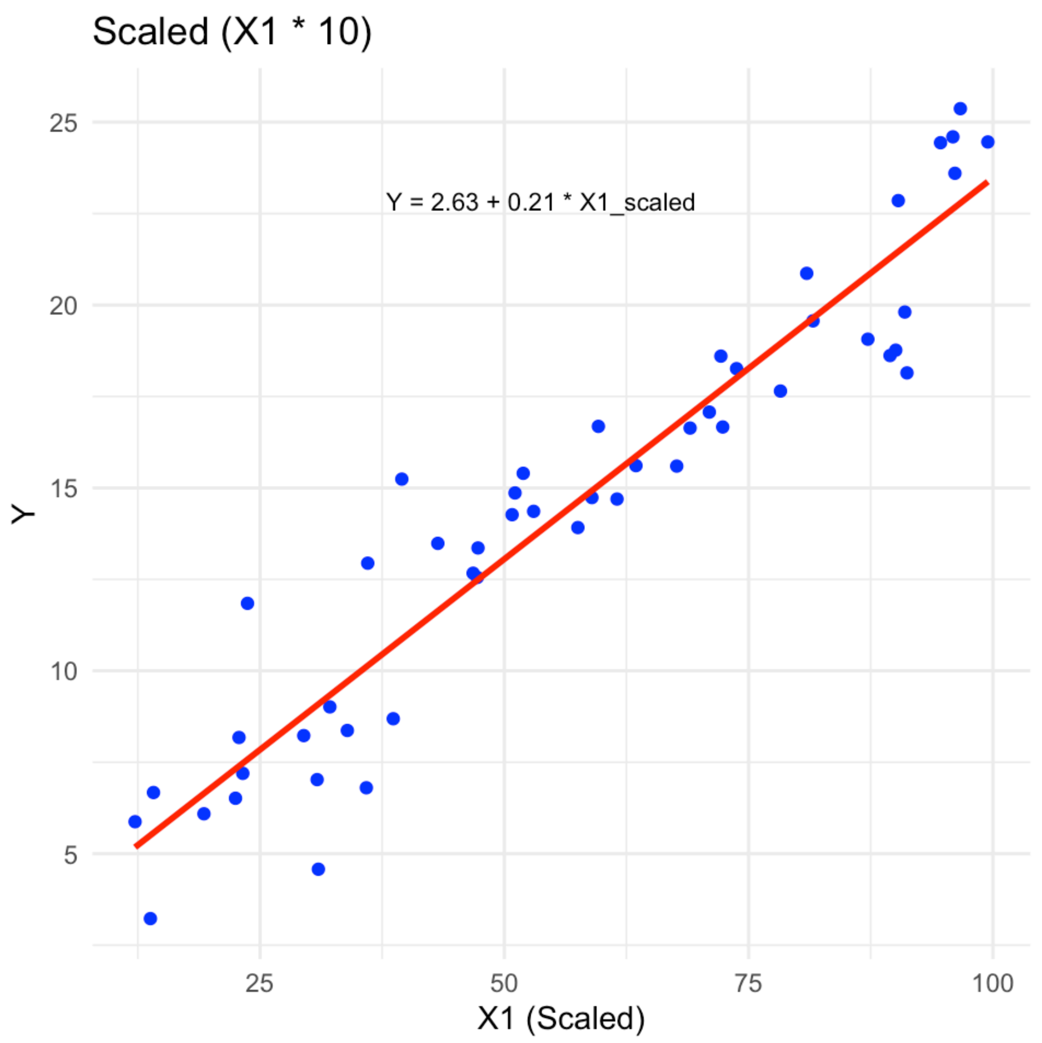




- Unscaled
  - `Murder`, `Rape` and `assault` are reported as the number cases per 100,000 people while `UrbanPop` is measured in % term.
  - Variances associated with four features are 18.97, 87.73, 6945.16 and 209.5.

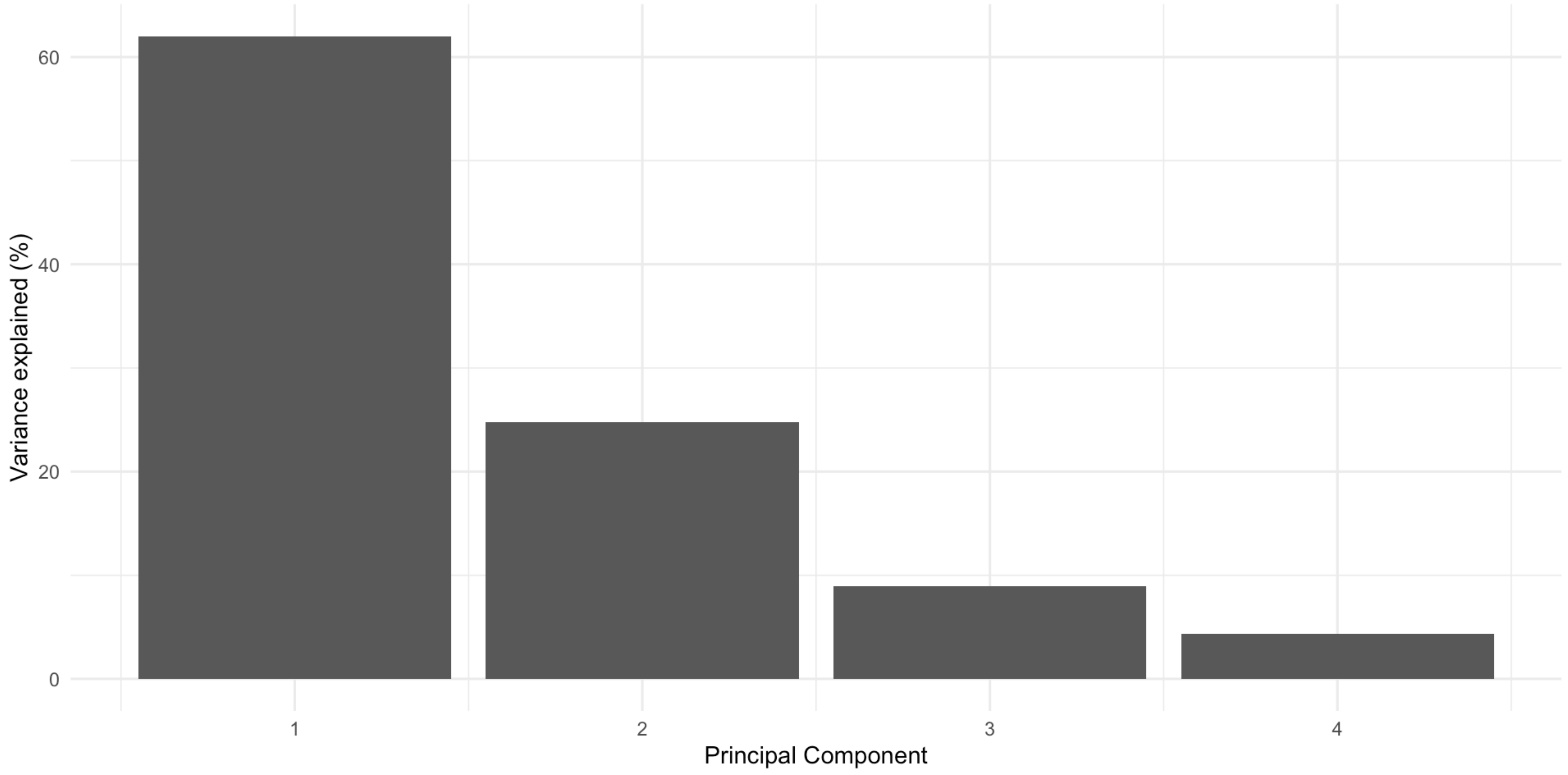
Effect of scaling (left) vs unscaled (right) in linear regression





Scree plot

▶ Code



PCA Properties

- **Unique** and **Global** solution!
- Ordered components
- Best linear dimension reduction possible
- Is not the best for non-linear relationships

PCA with K-means

- Very common approach to deal with high dimensional data
- Use the first M principal component scores as inputs into the K-means algorithm, where $M \ll p$
- Can help improve the clustering model if the signal in the data can be captured in a few principal components

PCA with regression

- Use the first M principal component scores as the predictors in a linear regression model
- We are assuming that a small number of principal components can explain most of the variability in the data as well as the response
- PCA is useful when variables in the data are highly correlated (i.e., collinear)

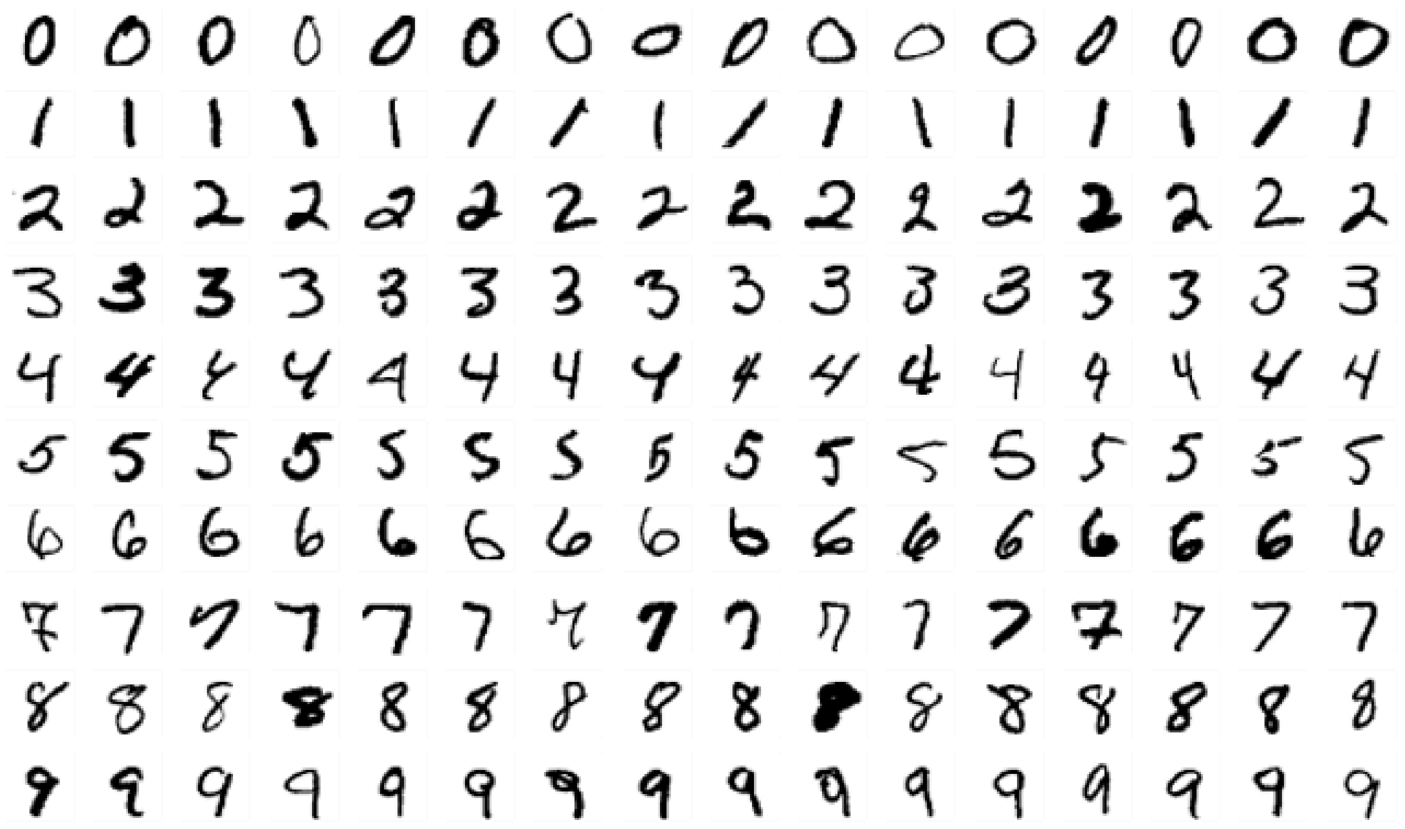
Dimension reduction t-SNE

t-SNE

- The name comes from t-distributed **s**tochastic **n**eighbour **e**mbedding
- **Non-linear technique** developed for visualizing high dimensional datasets
- Use local structure in the data to find a low dimensional representation
- Applications include computer security research, music analysis, cancer research, bioinformatics, and biomedical signal processing

MNIST Example

- 28 by 28 pixel images of handwritten digits
- 784 features



Three steps in t-SNE

t-SNE helps visualize high-dimensional data by preserving local structures in a lower-dimensional space.

Step 1: Construct High-Dimensional Probability Distribution (Not assessable)

- Define similarities between data points in the high-dimensional space.
- Similar points have a **higher probability** of being neighbors.

Mathematical definition:

$$p_{j|i}(\sigma_i^2) = \frac{\phi(x_j; x_i, \sigma_i^2)}{\sum_{k \neq i} \phi(x_k; x_i, \sigma_i^2)}$$

- $\phi(x; \mu, \sigma)$ denotes the Gaussian density
- Think of $p_{j|i}(\sigma_i^2)$ as a conditional probability that $\boldsymbol{x}_i$ would pick $\boldsymbol{x}_j$ as its neighbour if neighbours were picked in proportion to their probability density under a Gaussian centered at $\boldsymbol{x}_i$ and with variance $\boldsymbol{\sigma}_i^2$
- σ_i is determined by a parameter called ***perplexity***
- The conditional probability can be made symmetric with

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

Step 2: Define Low-Dimensional Probability Distribution (Not assessable)

- Map the data into a lower-dimensional space (e.g., **2D or 3D**).
- Define a new probability distribution to reflect the relationships between data points.
- Use a **Student's t-distribution** (with one degree of freedom) instead of a Gaussian to avoid "crowding" effects.

Mathematical definition:

$$q_{ij} = \frac{(1 + ||y_j - y_i||_2^2)^{-1}}{\sum_{k \neq i}(1 + ||y_j - y_i||_2^2)^{-1}}$$

- The above formula comes from the probability density function of t-distributions with one degree of freedom (i.e., Cauchy distribution)

Step 3: Optimizing with Kullback-Leibler (KL) Divergence (Not assessable)

- Adjust the positions of points in the low-dimensional space to **minimize** the difference between high-dimensional and low-dimensional distributions.
- KL divergence is a non-symmetric measure of the difference between two probability distributions
- KL divergence is defined as:

$$\mathrm{KL}(p||q) = \sum_{i \neq j} p_{ij} \log \left(\frac{p_{ij}}{q_{ij}} \right)$$

Think of KL divergence as a measure of how many bits of information is lost when we use $\boldsymbol{q}$ to approximate $\boldsymbol{p}$

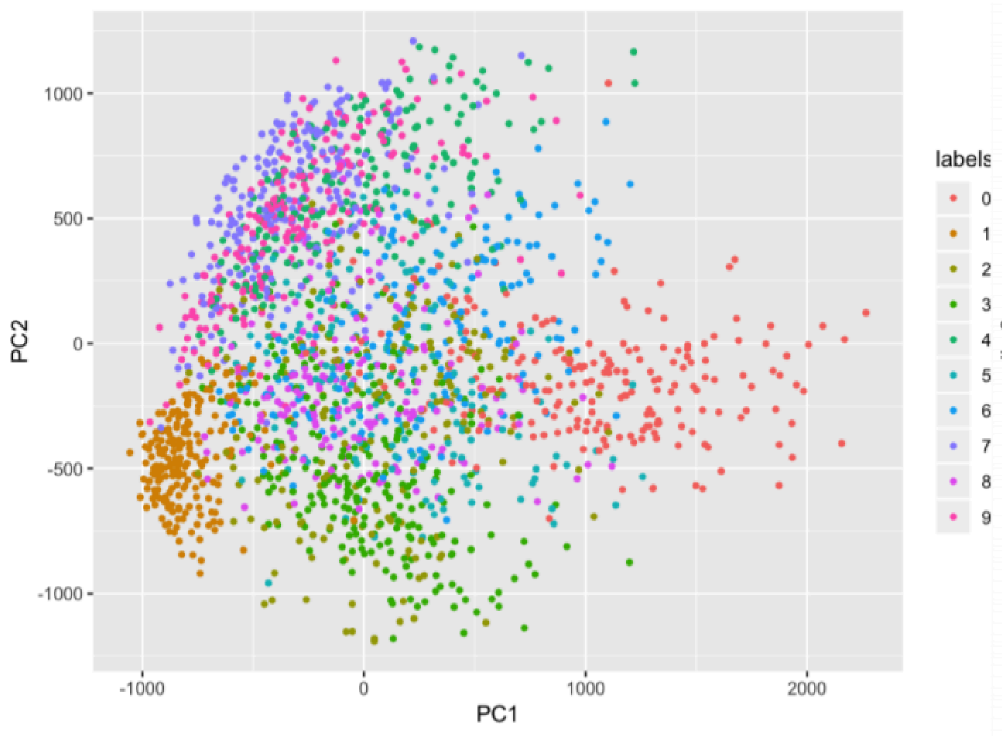
where:

- A lower KL divergence means the lower-dimensional representation **better matches** the original high-dimensional structure.

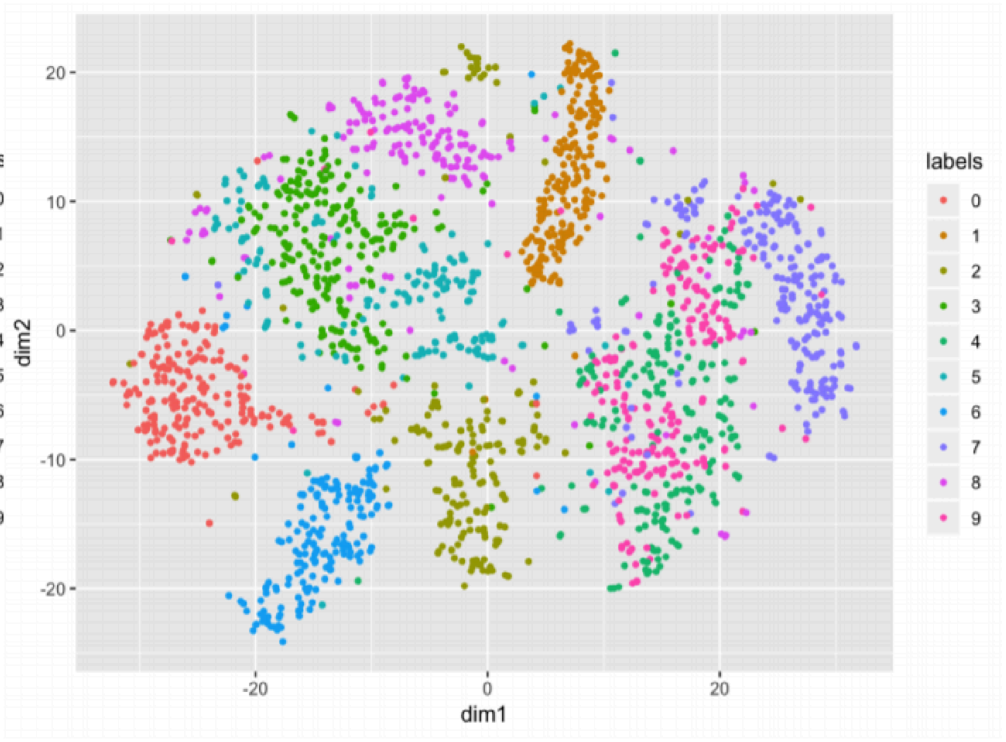
t-SNE vs PCA

MNIST Example

PCA



tSNE



- PCA is a linear method and can only capture linear relationships.
- PCA is defined by a mathematical formula.
- The principal components (PCs) from PCA can be interpreted and used for inference.
- PCA is less computationally intensive

- t-SNE is a non-linear technique and is able to find complicated non-linear relationships.
- t-SNE is a probabilistic method – it will give you a different representation every time you run it
- t-SNE is mostly a visualization method. The visualization from t-SNE cannot be used for inference.
- t-SNE is more computationally intensive

Dimension reduction: Multidimensional Scaling (MDS)

Multidimensional Scaling (MDS)

- Visually represent proximities (similarities or distances) between objects in a lower dimensional space (usually 2 or 3d space)
- The objective of MDS is to take a Matrix of similarities or dissimilarities, $\boldsymbol{D}$, and find projections $z_1, \ldots, z_k$ where $\boldsymbol{k}$ is the desired lower dimension
- The distances are near preserved by optimizing a stress function
- Full data not required

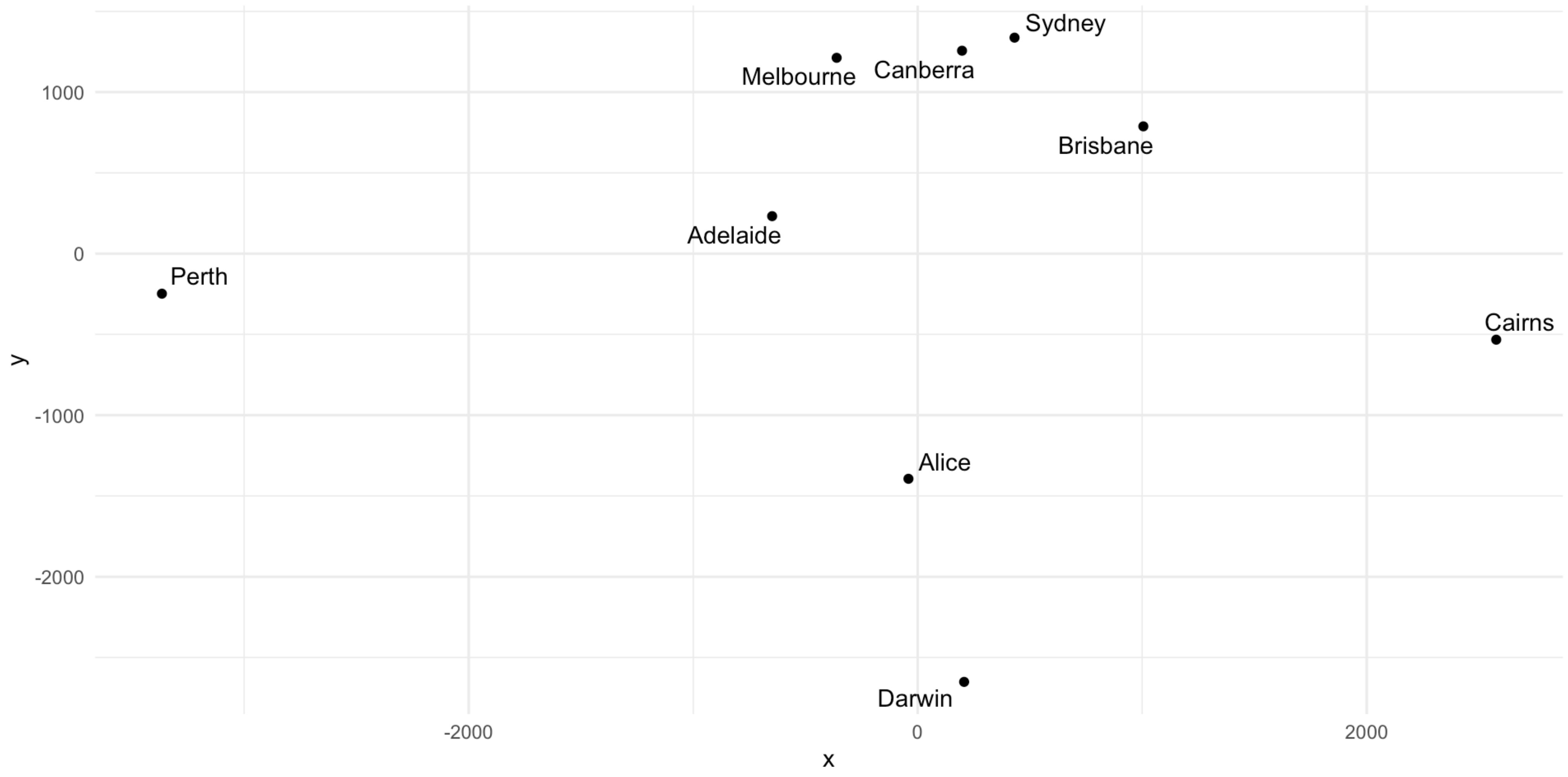
Multidimensional Scaling (MDS) Example

```r
cities <- c("Adelaide", "Alice", "Brisbane", "Cairns", "Canberra", "Darwin", "Melbourne", "Perth", "Sydney")
city.dist <- matrix(c(0, 1533, 2044, 3143, 1204, 3042, 728, 2725, 1427,
                      1533, 0, 3100, 2500, 2680, 1489, 2270, 3630, 2850,
                      2044, 3100, 0, 1718, 1268, 3415, 1669, 4384, 1010,
                      3143, 2500, 1718, 0, 2922, 3100, 3387, 5954, 2730,
                      1204, 2680, 1268, 2922, 0, 3917, 647, 3911, 288,
                      3042, 1489, 3415, 3100, 3917, 0, 4045, 4250, 3991,
                      728, 2270, 1669, 3387, 647, 4045, 0, 3430, 963,
                      2725, 3630, 4384, 5954, 3911, 4250, 3430, 0, 4110,
                      1427, 2850, 1010, 2730, 288, 3991, 963, 4110, 0),
                    nrow = length(cities), ncol = length(cities))
colnames(city.dist) <- rownames(city.dist) <- cities
city.dist %>% kbl %>% kable_paper("hover", full_width = TRUE)
```

	Adelaide	Alice	Brisbane	Cairns	Canberra	Darwin	Melbourne	Perth	Sydney
Adelaide	0	1533	2044	3143	1204	3042	728	2725	1427
Alice	1533	0	3100	2500	2680	1489	2270	3630	2850
Brisbane	2044	3100	0	1718	1268	3415	1669	4384	1010
Cairns	3143	2500	1718	0	2922	3100	3387	5954	2730
Canberra	1204	2680	1268	2922	0	3917	647	3911	288
Darwin	3042	1489	3415	3100	3917	0	4045	4250	3991
Melbourne	728	2270	1669	3387	647	4045	0	3430	963
Perth	2725	3630	4384	5954	3911	4250	3430	0	4110

Multidimensional Scaling (MDS) Example

```r
mds <- cmdscale(city.dist, k = 2); colnames(mds) <- c("x", "y")
mds <- data.frame(mds, City = colnames(city.dist))
ggplot(mds, aes(x = x, y = y, label = City)) + geom_point() + ggrepel::geom_text_repel() + theme_minimal()
```



MDS Stress functions

- These functions attempt to force the lower dimensional projections to preserve the distances in the original data
- Common stress functions
  - ➟ Least squares $S_{\mathrm{LS}}(z_1, z_2) = \sqrt{\sum_{i \neq j} (d_{ij} - ||z_i - z_j||)^2}$

Comments on MDS

Benefits & Drawbacks of MDS

- Requires only a distance or dissimilarity matrix, not the full dataset.
- Must choose the number of dimensions KK (similar to the elbow method in a Scree plot).
- Useful for visualizing non-linear data in a lower-dimensional space.

How to Interpret MDS Maps

- Rotation does not matter – axes and orientation are arbitrary.
- Focus on relative positions – only distances between points are meaningful.
- Objects closer together on the MDS map are more similar based on the input distance matrix.