Data Visualisation

CMP020L013A

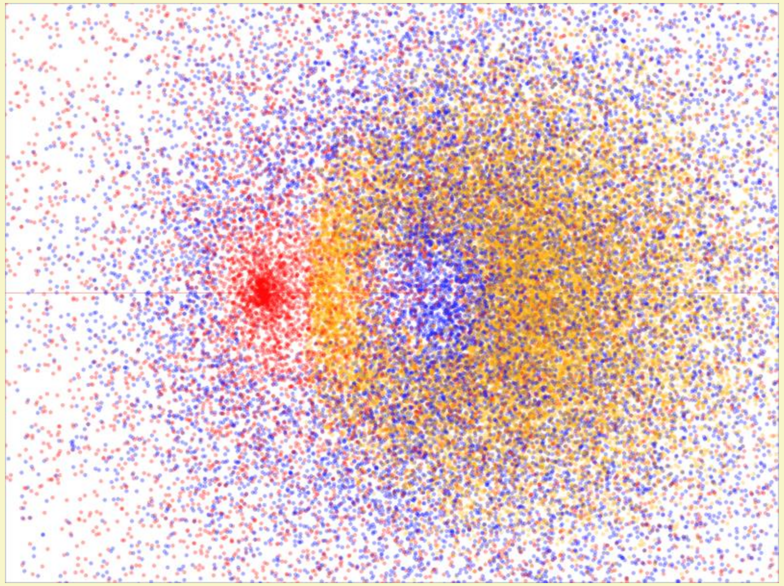
Week 10: Visualisation of High-Dimensional Data

Mohammad Javaheri

(Dr Mohammad Ali Javaheri Javid)

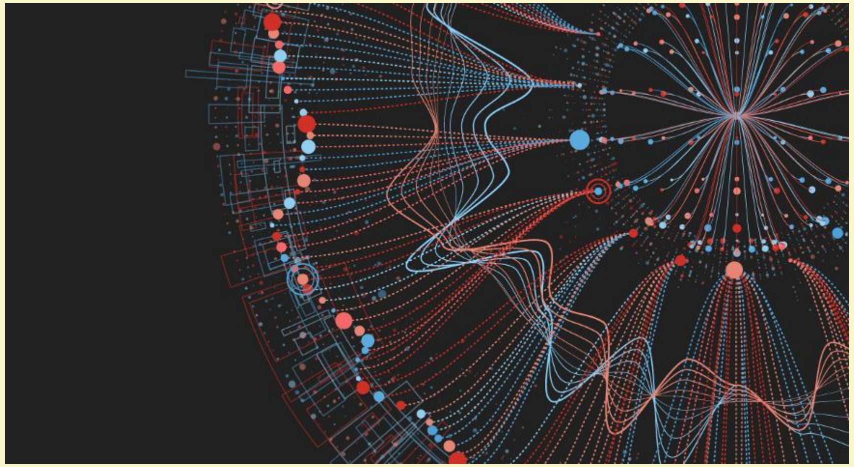
Agenda

- Visualizing high-dimensional data
- Data matrix preparation
- Clustering
- Dimensionality reduction with projections
- Parallel coordinates / parallel sets
- Table lens and radar charts (spider chart)



Visualizing High Dimensional Data

- Imagine you get a dataset with hundreds of features (variables) and have little understanding of the domain the data belongs to.
- You are expected to identify hidden patterns in the data, explore and analyse the dataset.
- How to explore a multidimensional dataset?
- Tackling high dimensional data is a common hurdle faced today while handling massive real-world datasets

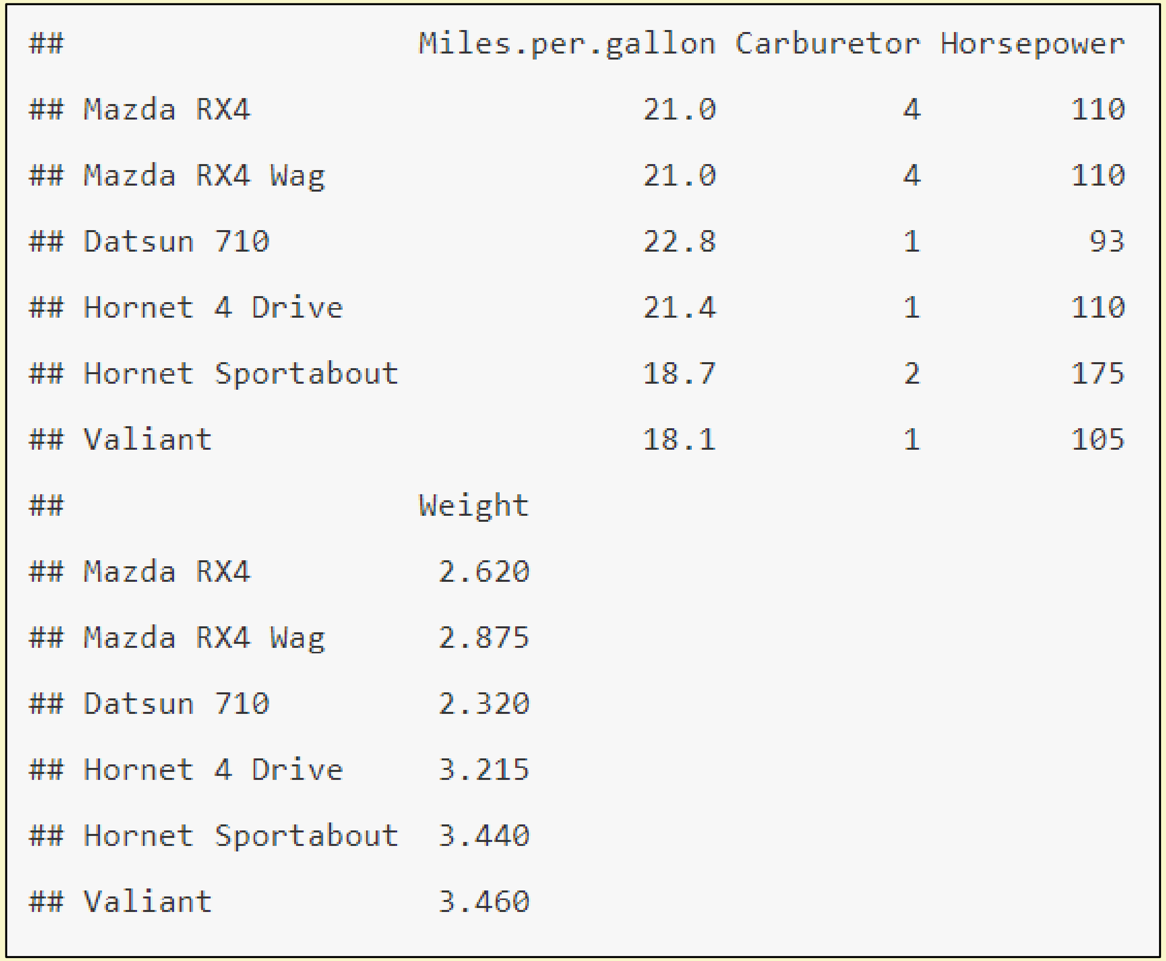


Visualizing High Dimensional Data

- ►First, we use heatmaps to discover interesting patterns
- ►Second, we reorder data matrices via clustering methods
- ►Third, we use dimensionality reduction methods to scale up visualisations to very high-dimensional data
- ►Finally, we introduce other practical visualisation methods for high-dimensional data

Dataset Preparation

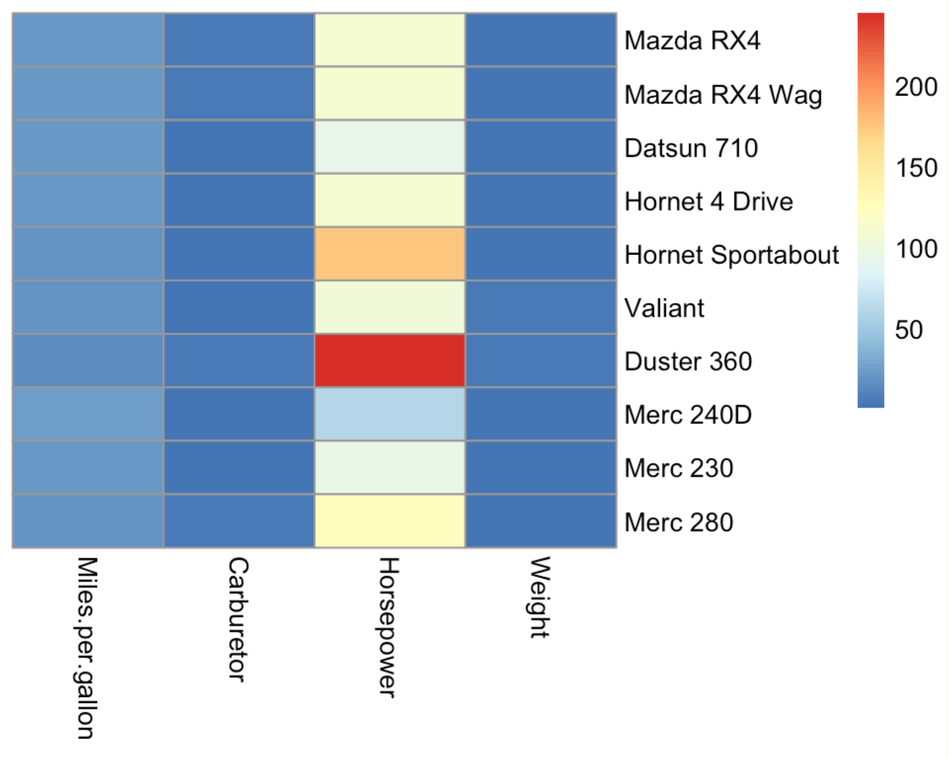
- [Cars dataset](#) is used here to analyse data, detect patterns, and visualise
- A subset of the base dataset consists of 10 rows (cars) and 4 selected variables



```
##                   Miles.per.gallon Carburetor Horsepower
## Mazda RX4                     21.0          4        110
## Mazda RX4 Wag                 21.0          4        110
## Datsun 710                    22.8          1         93
## Hornet 4 Drive                21.4          1        110
## Hornet Sportabout             18.7          2        175
## Valiant                       18.1          1        105
##                   Weight
## Mazda RX4          2.620
## Mazda RX4 Wag      2.875
## Datsun 710         2.320
## Hornet 4 Drive     3.215
## Hornet Sportabout  3.440
## Valiant            3.460
```

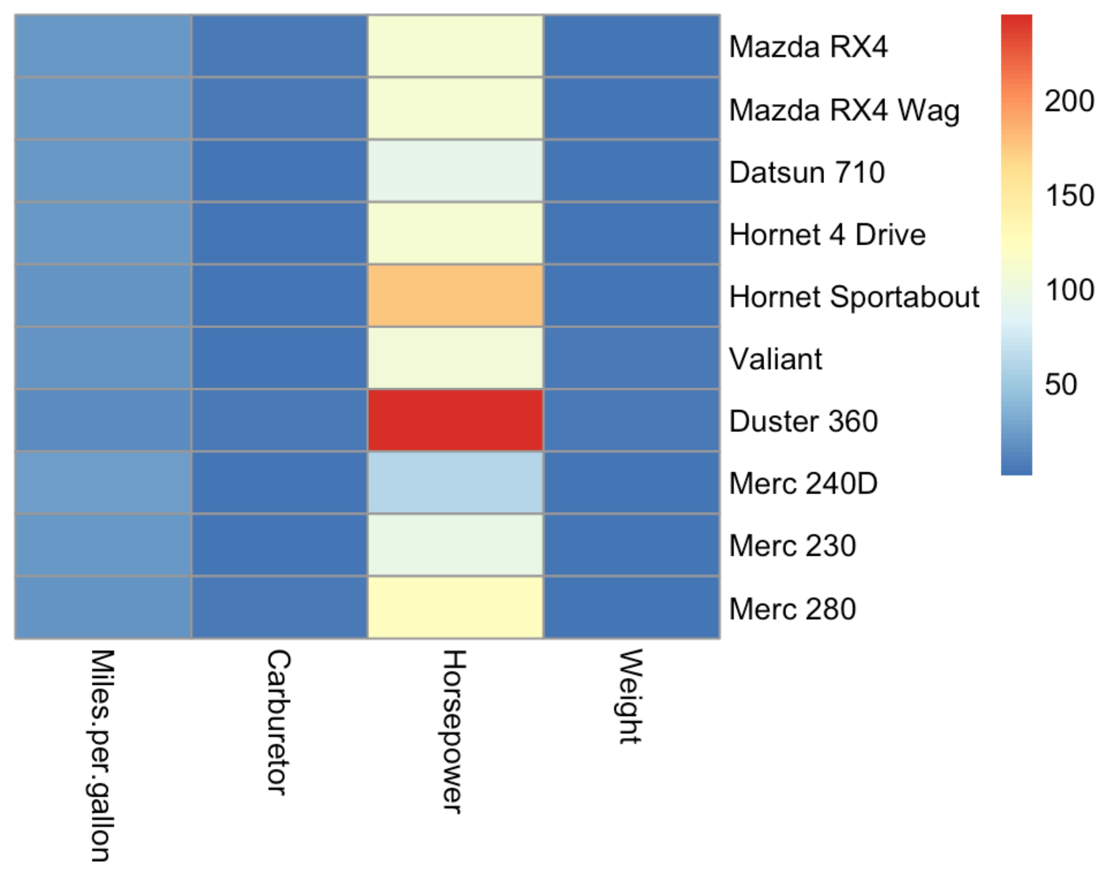
Heatmaps

- Heatmaps simply display data matrices as an image by color-coding its entries
- Heatmaps allow visualisation of data matrices of up to 1,000 rows and columns (order of magnitude), i.e., the pixel resolution of your screen



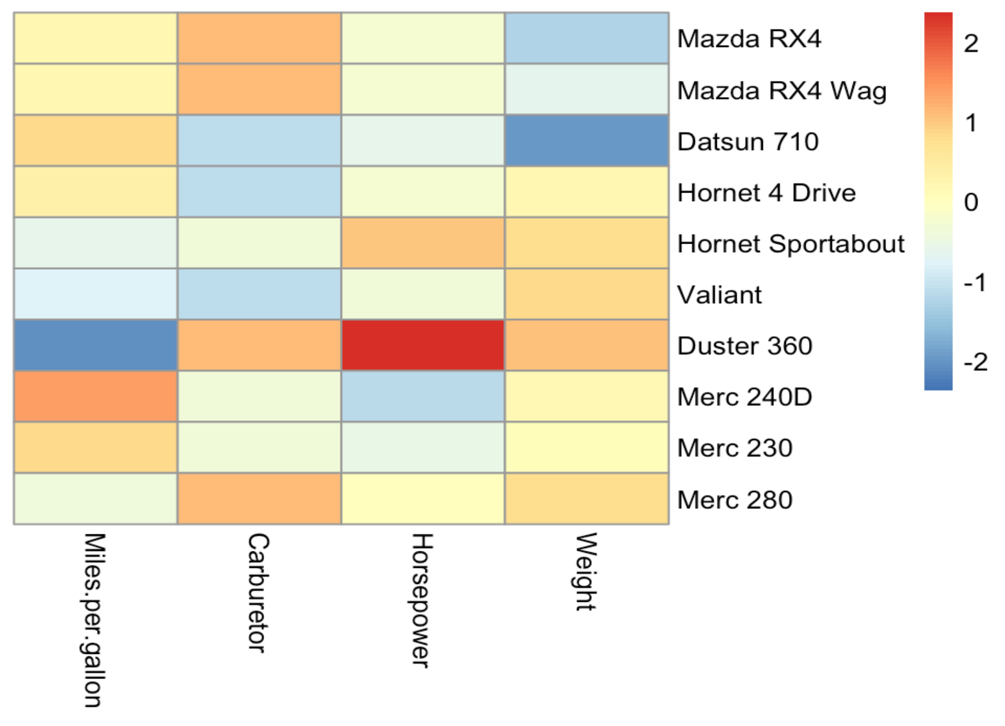
Heatmaps

- ►**Centering and scaling variables**
- ►Bringing variables to a common scale is useful for visualization but also for computational and numerical reasons
- ►It also makes analysis independent of the units chosen



Heatmaps

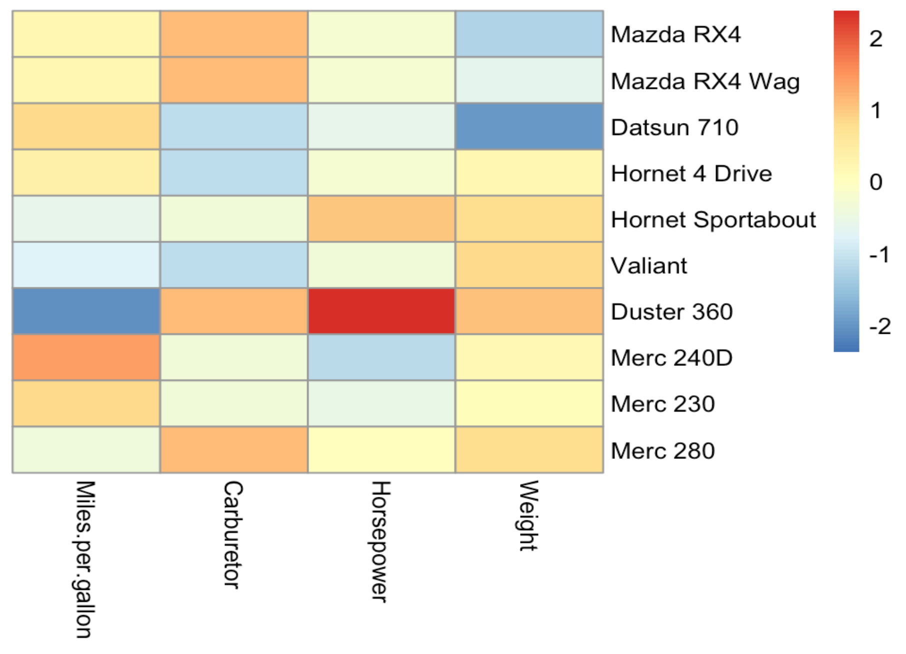
- **Centering and scaling variables**
- The widely used operations to bring variables to the same scale are:
  1. centring: subtracting the mean
  2. standard scaling or Z-score normalisation: centering then dividing by the standard deviation



For instance, the Cars dataset provide car weights in 1,000 pounds and gas consumption in miles per gallon. We want the analysis to be the same if these variables were expressed with the metric system

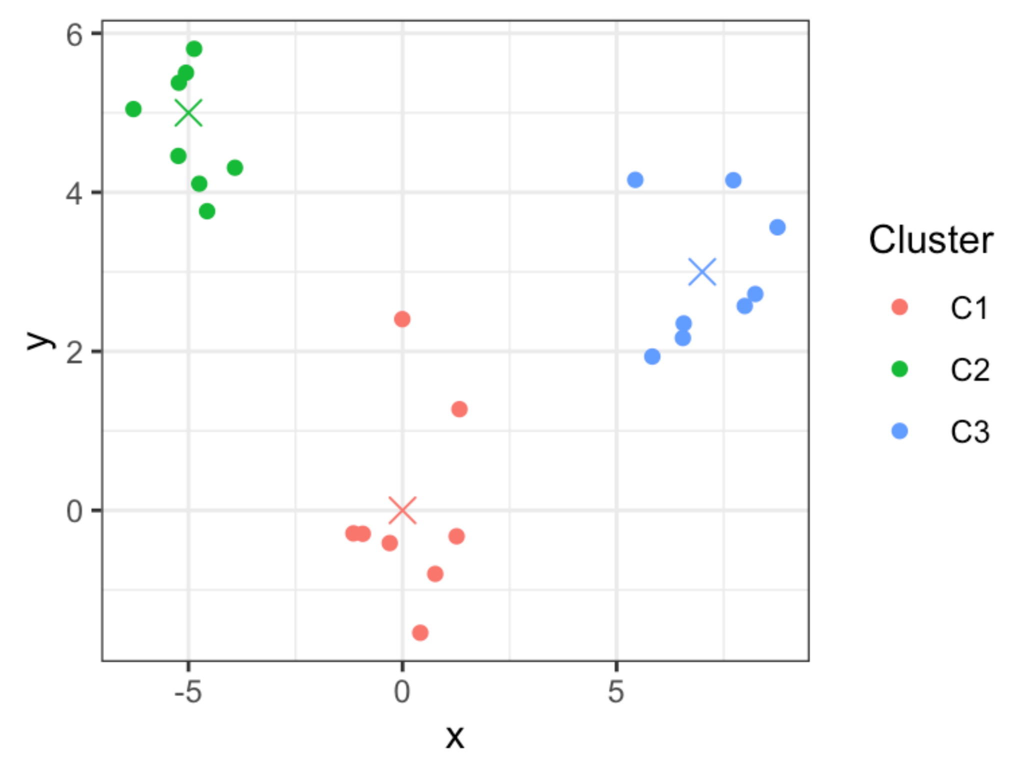
Clustering

- It is hard to see a pattern emerging in this heatmap figure
  - Which cars are similar to each other?
  - Which variables are similar to each other?
- Clustering is the task of grouping observations by similarities
- Clustering helps to find patterns in data matrices
- Clustering can also be applied to variables, e.g., applied on the transpose of the data matrix



K-Means Clustering

- ►K-Means clustering aims to partition the observations into K non-overlapping clusters
- ►The number of clusters K is predefined.
- ►The clusters C1,...CK define a partition of the observations, i.e., every observation belongs to one and only one cluster
- ►To this end, one makes use of so-called cluster centroids, denoted µ1,...,µK, and associates each observation to its closest centroid

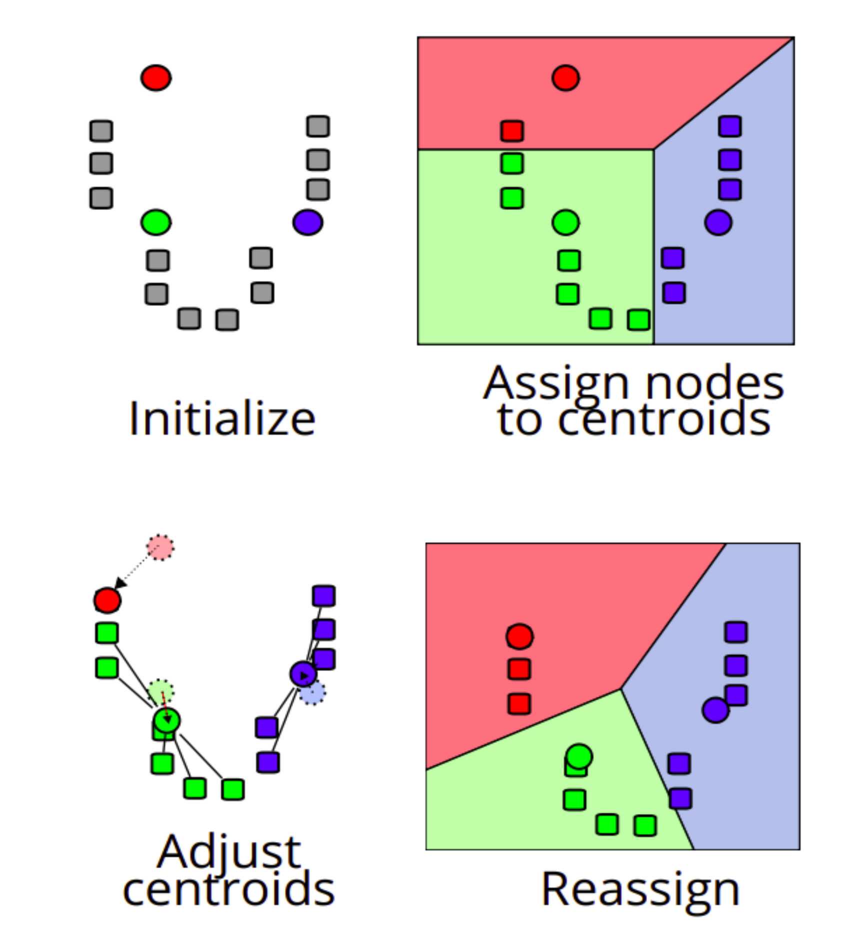


clusters K = 3

K-Means Clustering

▶ K-Means algorithm

1. Choose the K initial centroids (one for each cluster). Different methods such as sampling random observations are available for this task
2. Assign each observation xi to its nearest centroid by computing the Euclidean distance between each observation to each centroid
3. Update the centroids µk by taking the mean value of all of the observations assigned to each previous centroid
4. Repeat steps 2 and 3 until the difference between new and former centroids is less than a previously defined threshold

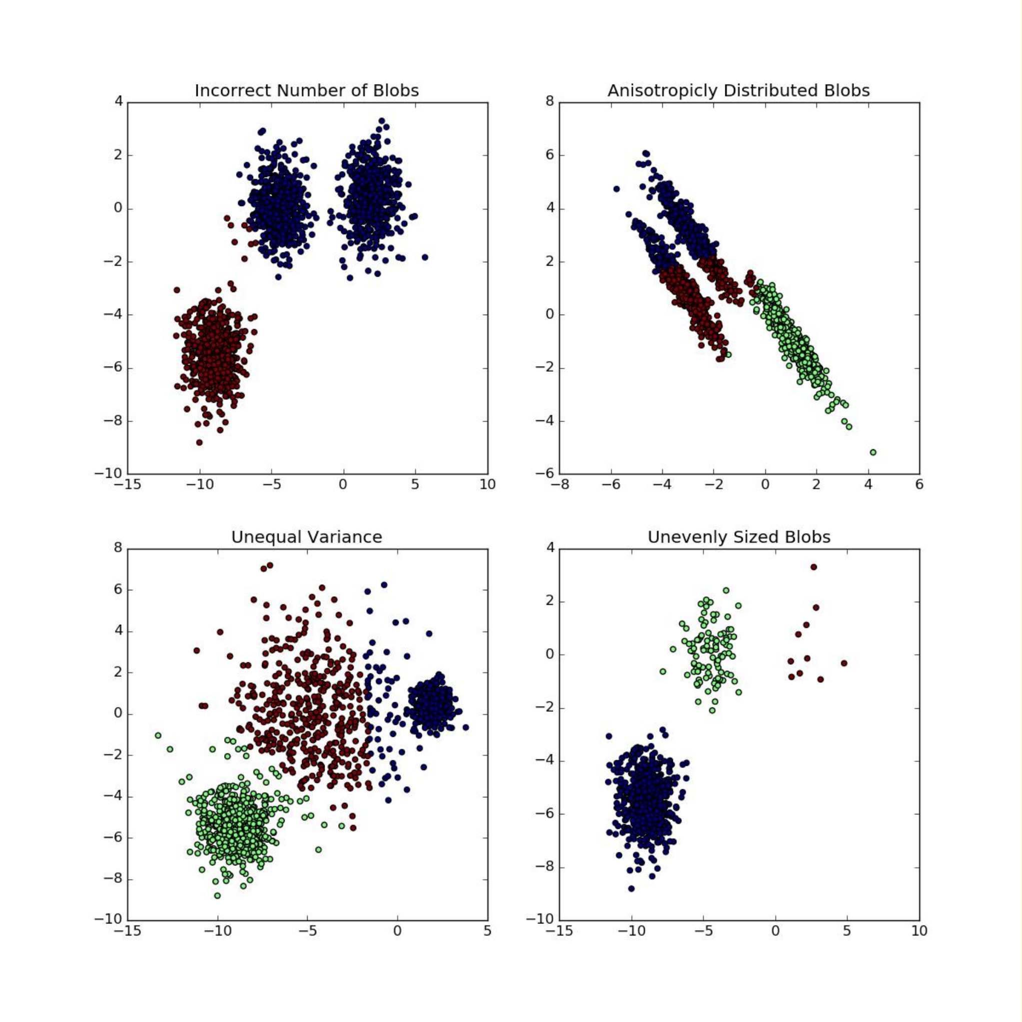


Source: https://en.wikipedia.org/wiki/K-means_clustering

K-Means Clustering

- K-Means considerations

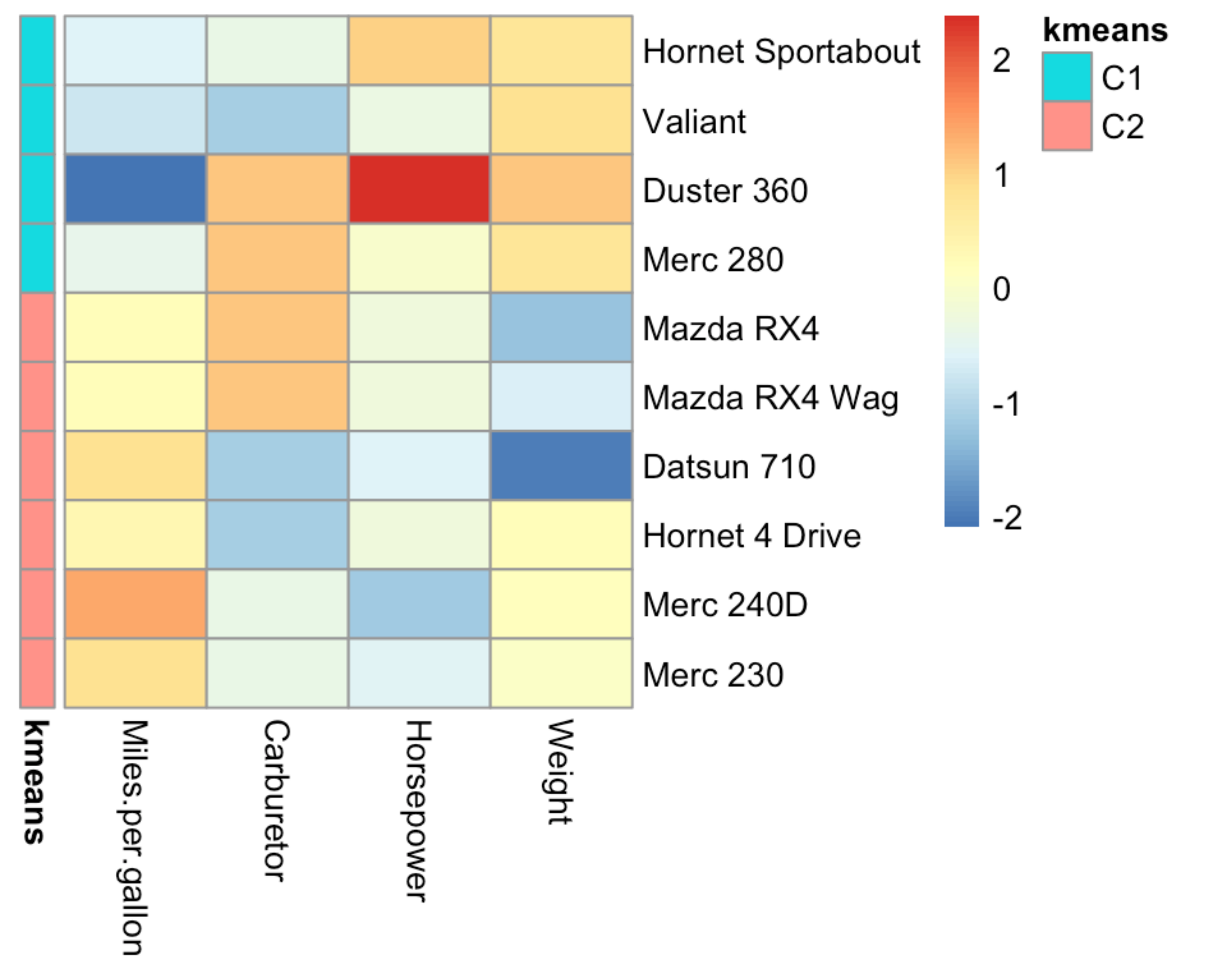
1. We have to make sure that the following assumptions are met when performing k-Means clustering
2. The number of clusters KK is properly selected
3. The clusters are isotopically distributed, i.e., in each cluster the variables are not correlated and have equal variance
4. The clusters have equal (or similar) variance
5. The clusters are of similar size



Source: Source: scikit-learn

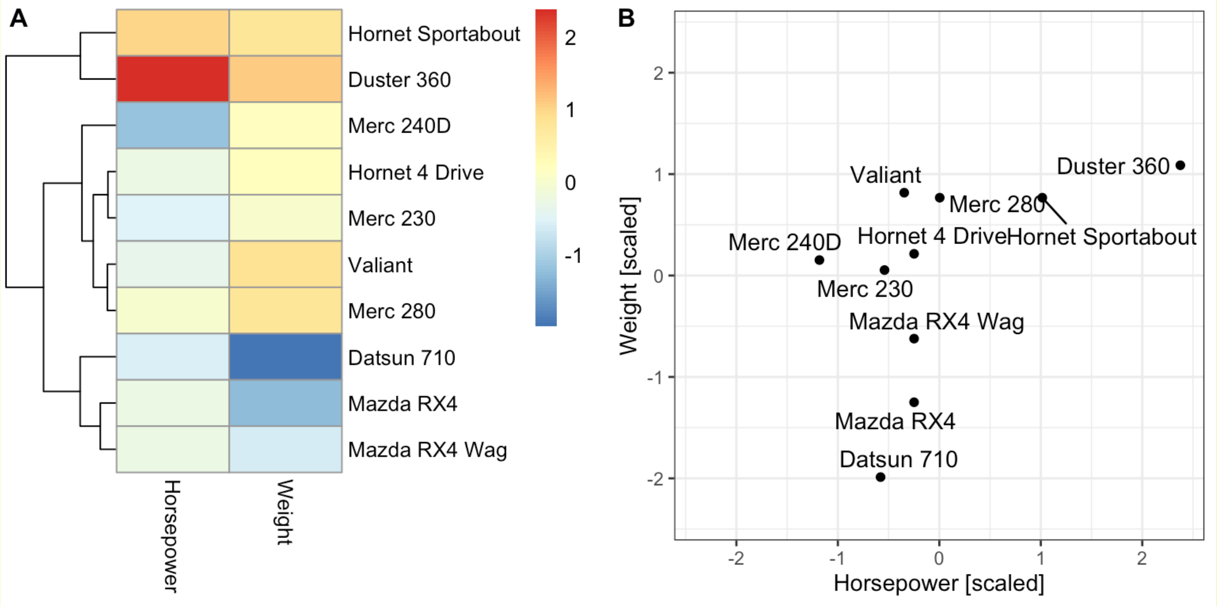
K-Means Clustering

- ►Pretty heatmap with K-mean cluster annotation
- ►Cluster C1 appears to group the heavy, powerful and gas-consuming cars and cluster C2 the light, less powerful and more economic cars



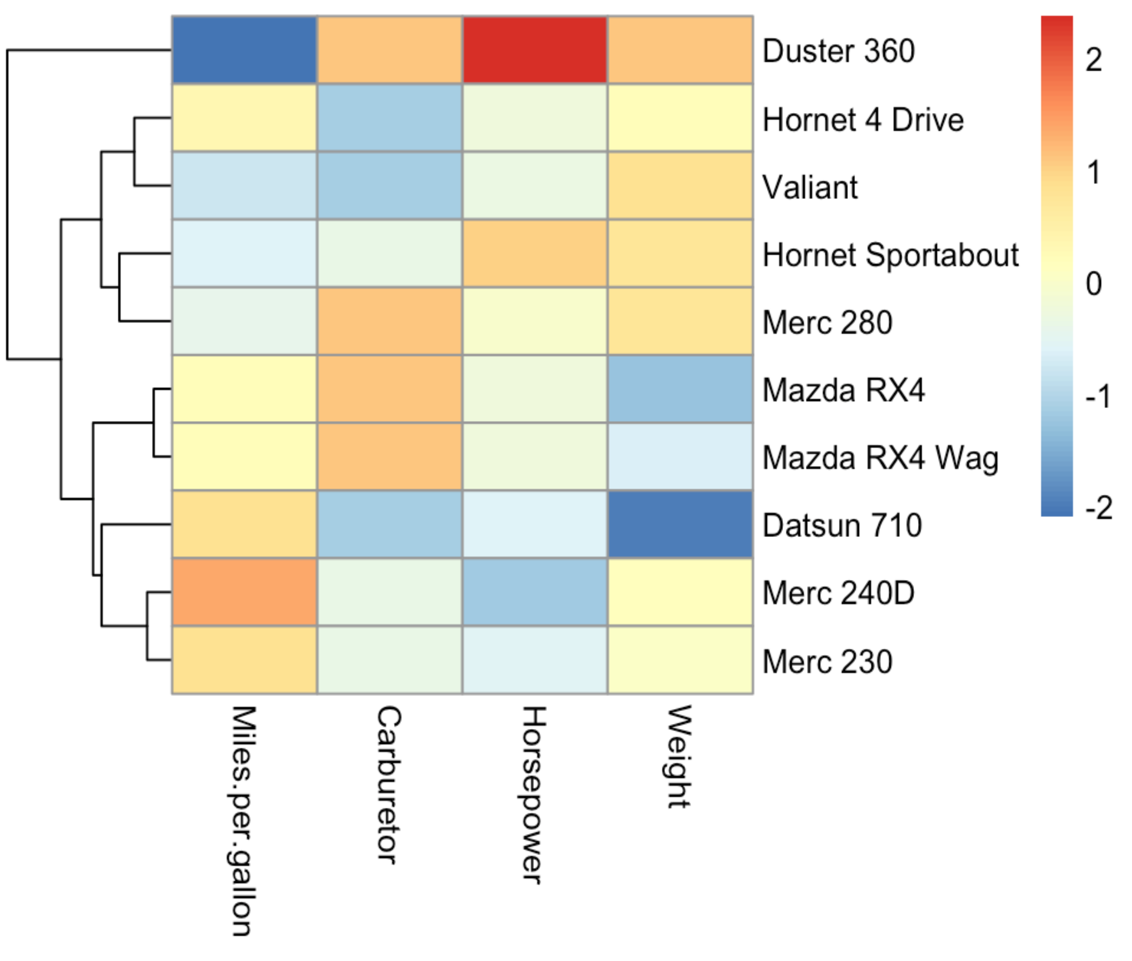
Hierarchical Clustering

- A major limitation of the K-means algorithm is that it relies on a predefined number of clusters.
- What if the interesting number of clusters is larger or smaller?
- Hierarchical clustering allows exploring multiple levels of clustering granularity at once by computing nested clusters.
- It results in a tree-based representation of the observations, called a dendrogram.



Hierarchical Clustering

- A major limitation of the K-means algorithm is that it relies on a predefined number of clusters
- Compared to K-means, the hierarchical clustering shows useful different degrees of granularity of the clusters.
- We see the most similar cars grouping together (the Mercedes 230 and the Mercedes 240D, as well as the Mazda RX4 and the Mazda RX4 Wag).
- At the high level, the Duster 360 stands out as an outlier.



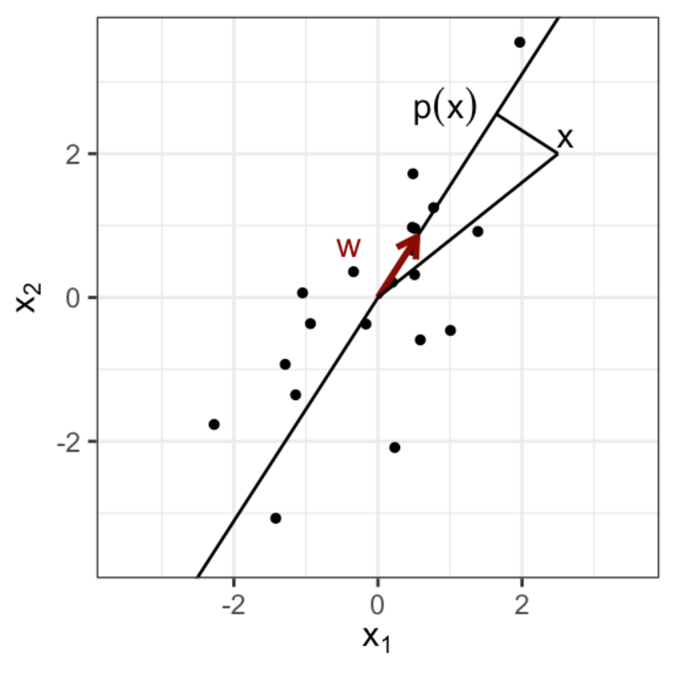
Dimensionality Reduction

- Beyond dimensions exceeding the thousands of variables, dimension reduction techniques are needed.
- The idea of dimension reduction is simple:
  - “if the dimension of our data *p* is too large, let us consider instead a representation of lower dimension *q* which retains much of the information of the database”
- Beyond dimensions exceeding the thousands of variables, dimension reduction techniques are needed
- We introduce here two common approaches to reduce and visualize high-dimensional data:
  1. Principal Component Analysis (PCA)
  2. t-distributed Stochastic Neighbor Embedding (t-SNE)

Principal Component Analysis (PCA)

- PCA transforms the correlated features in the data into linearly independent (orthogonal) components so that all the important information from the data is captured while reducing its dimensionality

1. **Definition of the first principal component (PC1)**
   - A *direction vector* of the line of length 1
2. **PC1 maximises the variance of the projected data**
   - The *proportion of variance* captured by PC1 is defined as the ratio of the variance of the projected data over the total variance of the data

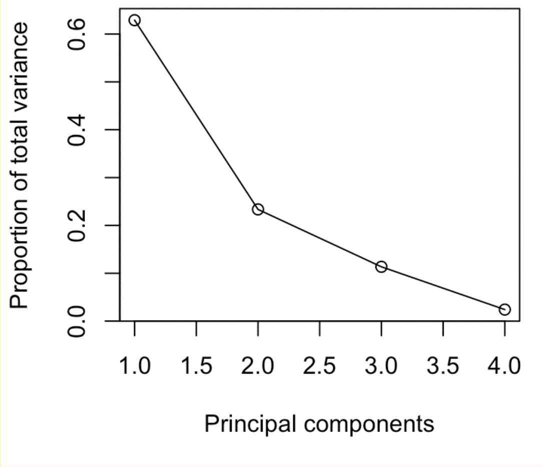


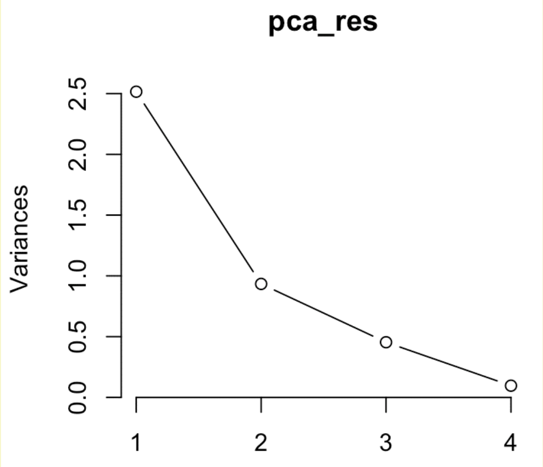
Principal Component Analysis (PCA)

- In the general case, with *p* variables and *n* observations, one searches for the *q*-dimensional plane that is closest to the data in terms of sums of squared Euclidean distances
- This is also the *q*-dimensional plane that maximises the variance of the projected data
- An important property relates principal components to the eigen-decomposition of the covariance matrix
- The covariance matrix is $\frac{1}{n}\mathbf{X}^\top\mathbf{X}$
- It is a symmetric positive matrix. We denote w1,...,wj,...w1,...,wj,... its eigenvectors ordered by decreasing eigenvalues λ1>...>λj>...
- **Result:** The variance explained by the PCA q-dimensional plane equals to the sum of the q-first eigenvalues of the covariance matrix.

Principal Component Analysis (PCA)

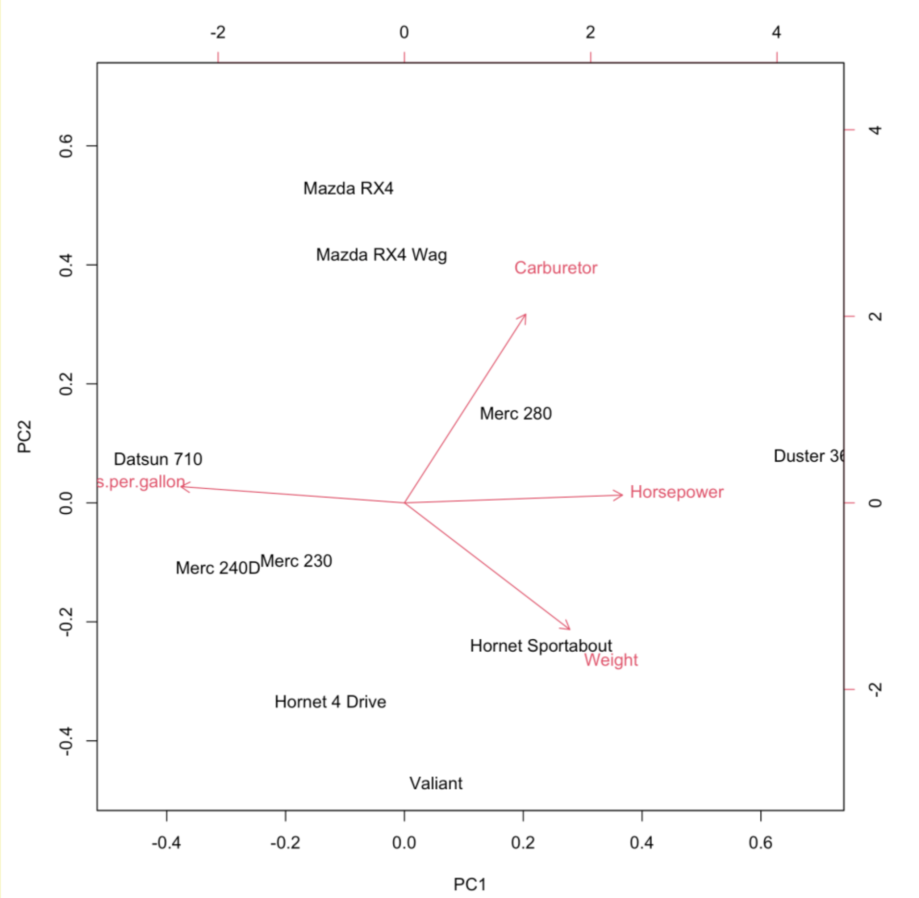
- ►Plotting PCA results
- ►The scree plot is a good first step for visualising the PCA output
- ►The scree plot shows the variance in each projected direction. The y-axis contains the eigenvalues, which essentially stand for the amount of variation.
- ►If the scree plot has an 'elbow' shape, it can be used to decide how many principal components to use for further analysis.





Principal Component Analysis (PCA)

- ►Plotting PCA results
- ►The biplot shows the projection of the data on the first two principal components
- ►It includes both the position of each sample in terms of PC1 and PC2
- ►It shows how the initial variables map onto this
- ►The correlation between variables can be derived from the angle between the vectors. Here, a small angle is related to a high correlation.



Principal Component Analysis (PCA)

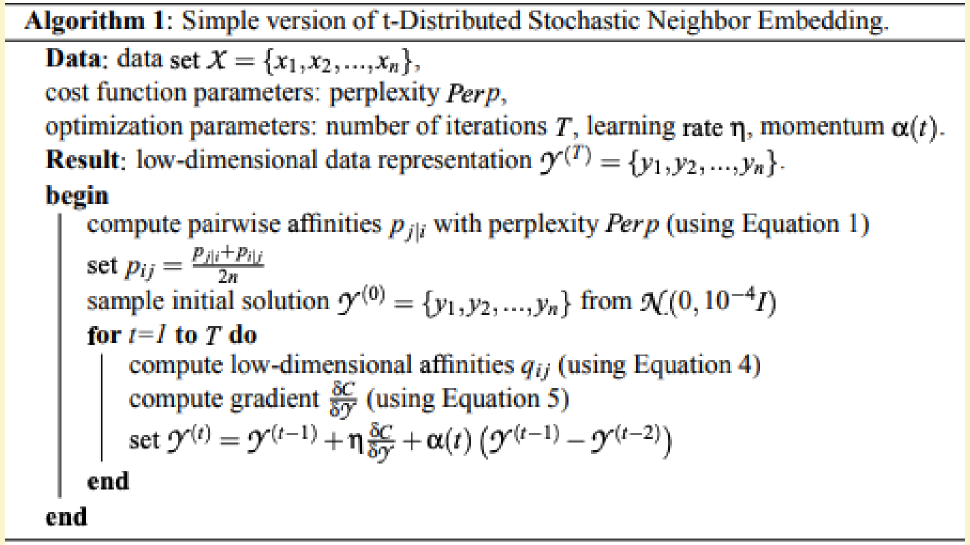
- One limitation of PCA is that it is restricted to linear transformation of the data
- What if the data lies closer to a parabola rather than a straight line?
- There are many non-linear alternatives to PCA including:
  - Independent Component Analysis (ICA)
  - kernel PCA, and
  - t-SNE

t-distributed Stochastic Neighbor Embedding (t-SNE)

- There was a lack of techniques able to preserve the local structure of high-Dimensional data and visualize the data as well.
- In 2008, Laurens van der Maaten and Geoffrey Hinton presented a new technique, ‘t-SNE’
- t-SNE can capture much of the local structure of the high-dimensional data, while also visualizing it in 2D
- Stochastic Neighbor Embedding (SNE) was presented by Hinton and Roweis in 2002, which constructed reasonably good visualizations
  - its results were obstructed because its cost function was difficult to optimize and due to the occurrence of the ‘crowding problem’
  - Crowding Problem occurs when the area in the 2D space is not able to accommodate all the points in the high-dimensional data.

t-distributed Stochastic Neighbor Embedding (t-SNE)

- To find a way around these problems, t-SNE uses the symmetric version of SNE cost function
- It further uses the Student-t distribution rather than a Gaussian to compute the similarity between two points in the low-dimensional space
- t-SNE uses the concept of similarity of data points to identify observed clusters by finding patterns with multiple features in the data.



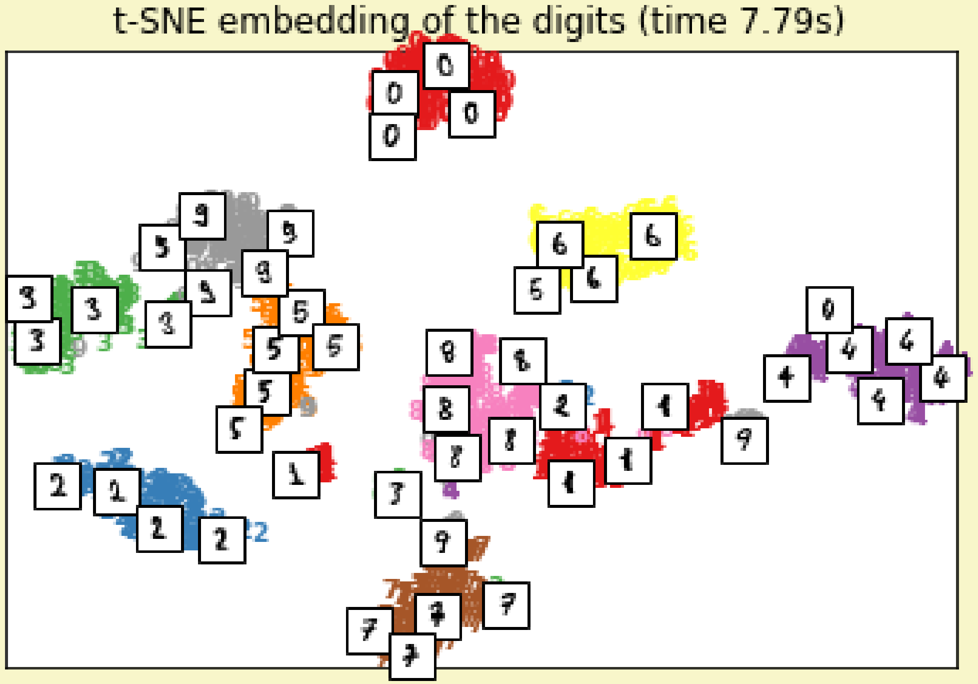
Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

```
Data: data set $\mathcal{X} = \{x_1, x_2, ..., x_n\}$,
cost function parameters: perplexity $Perp$,
optimization parameters: number of iterations $T$, learning rate $\eta$, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, ..., y_n\}$.
begin
    compute pairwise affinities $p_{j|i}$ with perplexity $Perp$ (using Equation 1)
    set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
    sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, ..., y_n\}$ from $\mathcal{N}(0, 10^{-4} I)$
    for $t$=1 to $T$ do
        compute low-dimensional affinities $q_{ij}$ (using Equation 4)
        compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
        set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t) \left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
    end
end
```

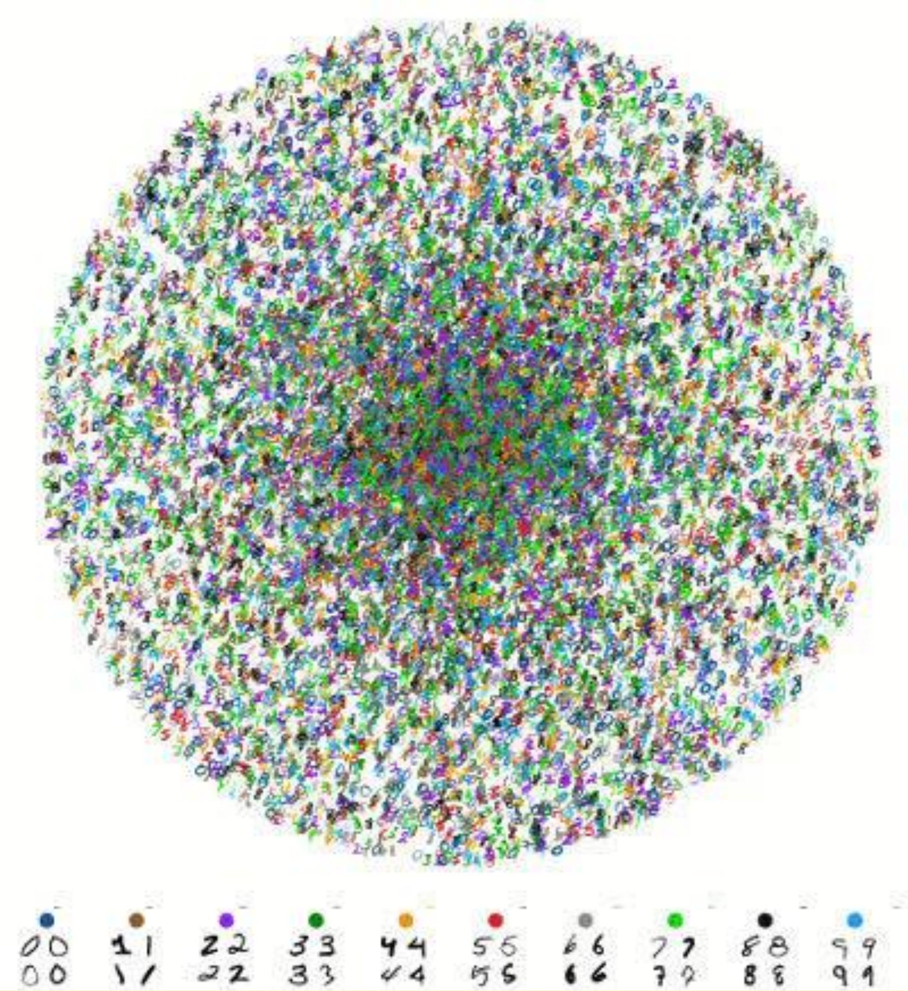
t-distributed Stochastic Neighbor Embedding (t-SNE)

- ► Example of t-SNE visualization using scikit-learn

- ► The MNIST handwritten digits dataset was used to test how t-SNE visualises the output

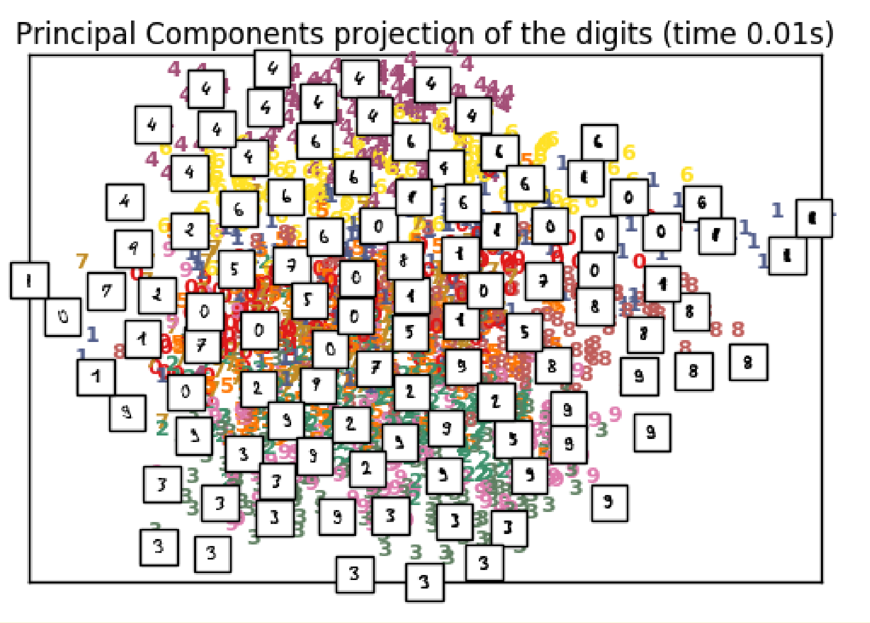


t-distributed Stochastic Neighbor Embedding (t-SNE)

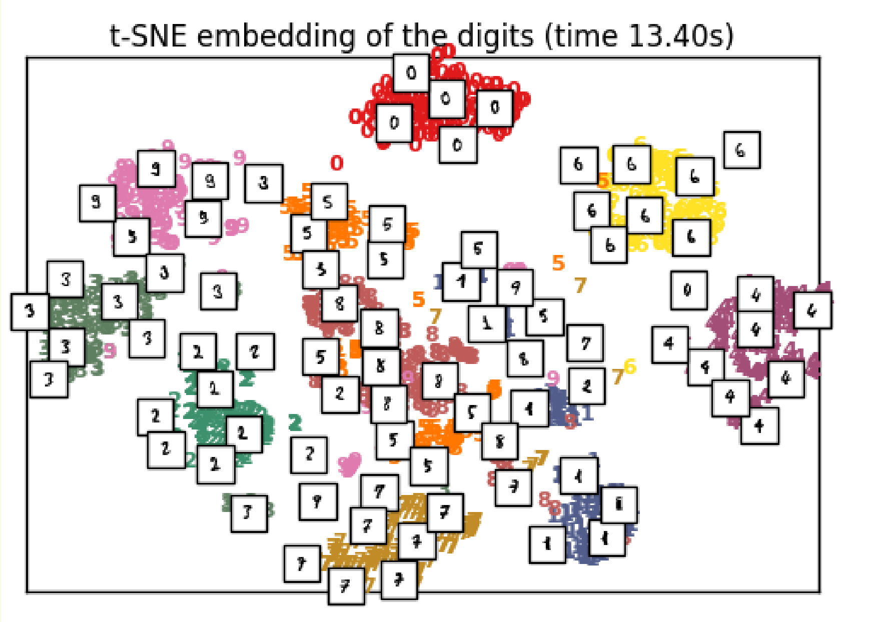


t-distributed Stochastic Neighbor Embedding (t-SNE)

- ►PCA and t-SNE visualization (the same dataset)
- ►Here it is evident that t-SNE has done a good job as compared to PCA in visualizing the digits



Principal Components projection of the digits (time 0.01s)



t-SNE embedding of the digits (time 13.40s)

Parallel Coordinates / Parallel Sets

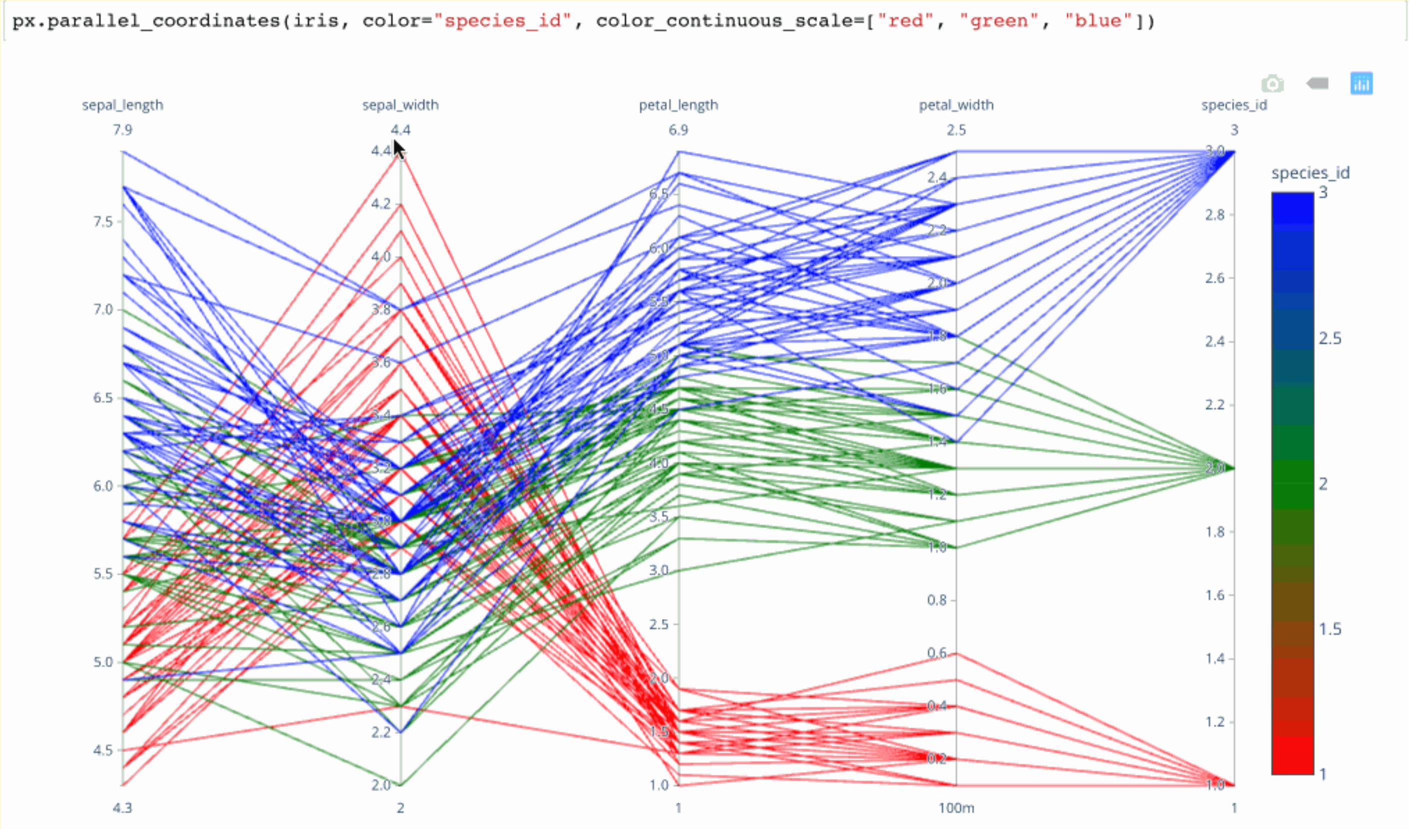
- ►Parallel coordinates are for dimensions that are numeric
- ►Parallel sets are for relationships among categorical dimensions
- ►Quantitative dimensions could be incorporate into parallel sets by binning them (like a histogram)
- ►Brushes (the ability to filter along dimensions by dragging a cursor) are critical to the usability of these charts
- ►https://observablehq.com/@d3/parallel-coordinates
- ► or a non-brushable parallel coordinate

Parallel Coordinates / Parallel Sets

- The ability to reorder the dimensions is also critical
- Given the susceptibility of these graphics to end up in a “hairball” state as network graphs often do
- Filtering is necessary to enable the user to untangle the lines into a useful state
- This blog is a Tableau research explains how to interpret the parallel coordinates:
- https://eagereyes.org/techniques/parallel-coordinates

Parallel Coordinates / Parallel Sets

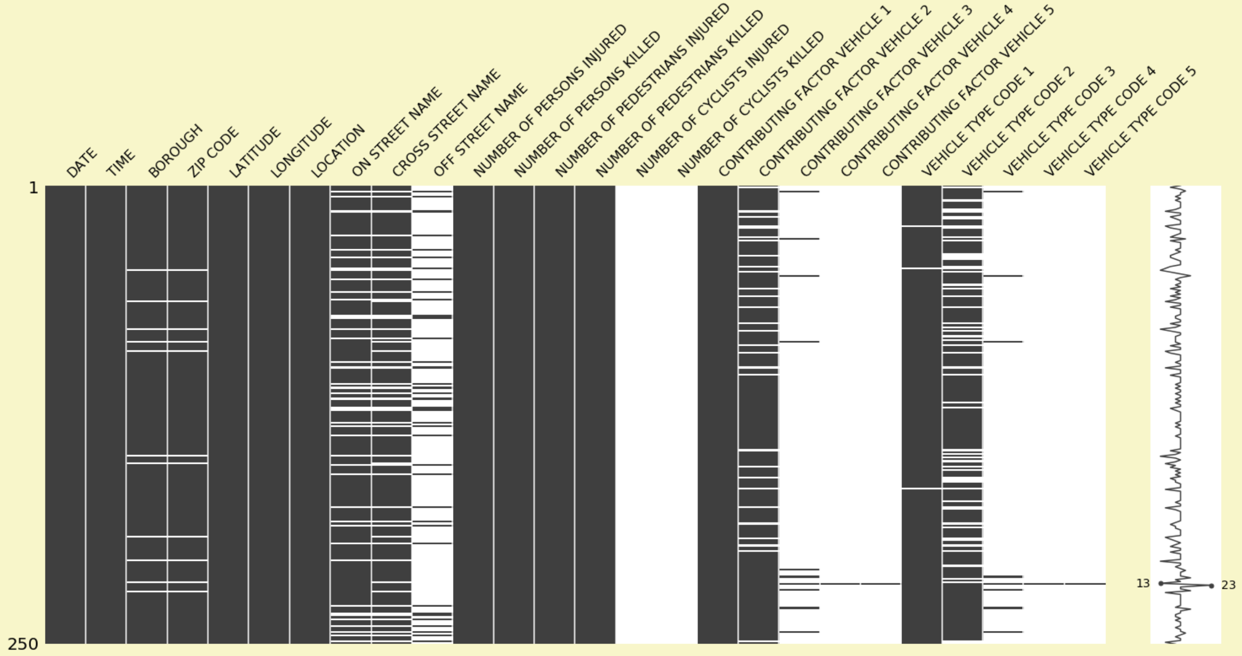
- ►plotly.express module produces interactive parallel coordinates in 1 line of Python
- ►It’s a fast way to produce interactive brushable and reorderable parallel dimension visualisations



```
px.parallel_coordinates(iris, color="species_id", color_continuous_scale=["red", "green", "blue"])
```

Other Techniques for Handling High Dimensions

- Table Lens
- This tool was developed by Ramana Rao in 1994.
- A video presentation introducing the technique is here:



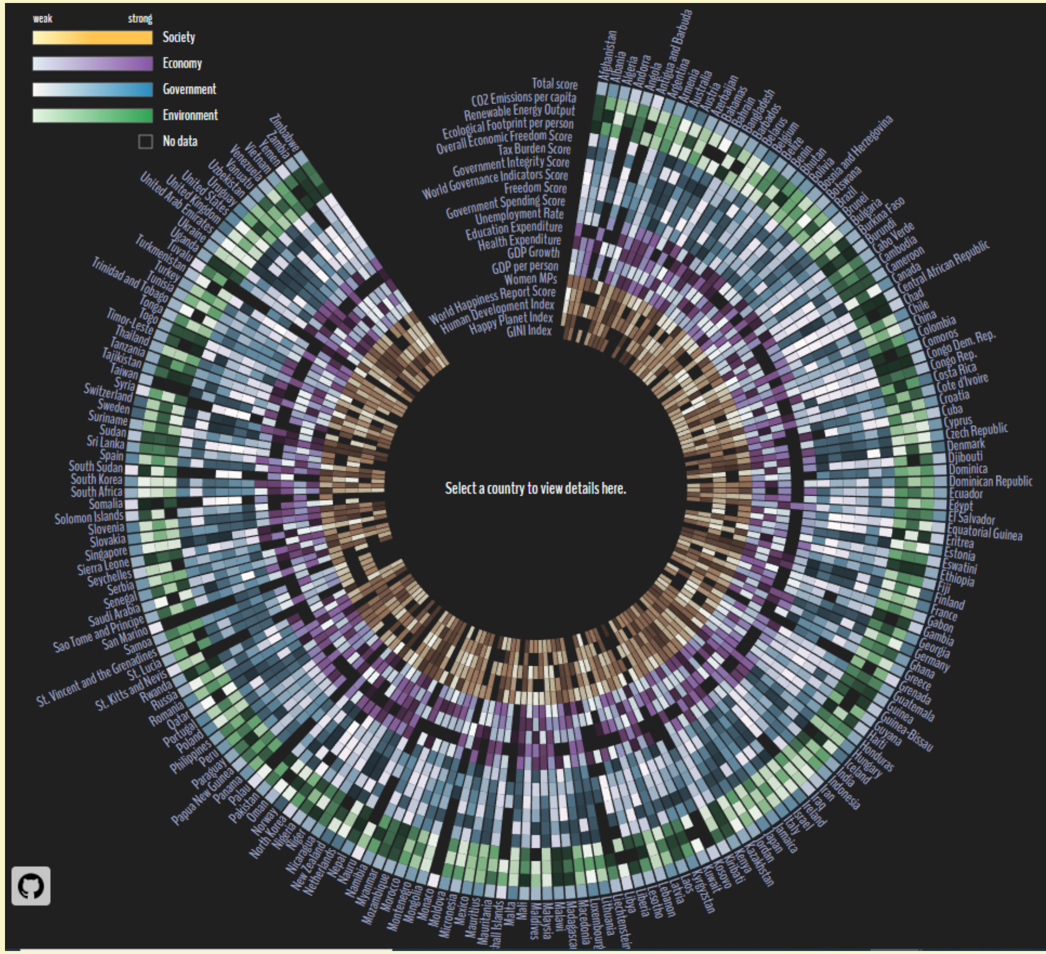
https://www.youtube.com/watch?v=ZDY9YCYv7z8

Other Techniques for Handling High Dimensions

- Radar/Spider Chart
- The usability of radar charts has been hotly debated within the data visualisation community for many years
- Examples where radar charts were found appropriate:
  - www.Collegeswimming.com uses them to provide a fingerprint for a swimmer’s overall performance across the 5 main event types

Other Techniques for Handling High Dimensions

- ▶Radar/Spider Chart
- ▶For the World Government Data Vis Competition, Karol Stopyra used radar charts to visualise 20 variables from each country at once, accompanied by a color-coded radial heatmap (see next slide)
- ▶https://stopyransky.github.io/wdvp-eye/



Further reading…

- https://gagneurlab.github.io/dataviz/high-dimensional-visualizations.html
- Plotly.express by Nicholas Kruchten https://medium.com/plotly/introducing-plotly-express-808df010143d
- Parallel Sets, A visualisation technique for multidimensional categorical data
  - https://www.jasondavies.com/parallel-sets/
- Demonstration of k-means assumptions:
  - https://scikitlearn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html
- Tabula Muris: https://tabula-muris.ds.czbiohub.org/