Data Mining and Machine Learning

Data Visualization

Dr. Edgar Acuna
Departamento de Ciencias Matematicas
Universidad de Puerto Rico Recinto de Mayaguez
Website: academic.uprm.edu/eacuna
Octubre 2021

Outline

- Visualization role
- Representing data in 1,2, and 3-D
- Representing data in 4+ dimensions
  - Scatterplot Matrix
  - Heatmaps
  - Parallel coordinates
  - Radviz, Starcoord

The visualization role

- Visualization is the process of transforming information into a visual form enabling the user to observe the information.
- Using successful visualizations for data mining and knowledge discovery tasks can reduce the time it takes to understand the underlying data, find relationships, and discover information.
- One of the goals of visualization is *explorative analysis.*

Visualization role (cont)

- In *explorative analysis*, visualization techniques are applied prior to the application of classification techniques to obtain insight into the characteristics of the dataset. The process involves a search for structures and the result is a visualization of the data which provides a hypothesis about the data.
- This use of visualization may improve the understanding that users have of their data, thereby, increasing the likelihood that new and useful information will be gained from the data.

Visualization Role

- Support interactive exploration
- Help in result presentation
- Disadvantages:
  - requires human eyes
  - Can be misleading

Tufte's Principles of Graphical Excellence

- Give the viewer
  - the greatest number of ideas
  - in the shortest time
  - with the least ink in the smallest space.

- Tell the truth about the data!

(**E.R. Tufte, "The Visual Display of Quantitative Information", 2nd edition**)

Visualization Methods

- Visualizing in 1-D, 2-D and 3-D
  - well-known visualization methods
- Visualizing in more dimensions
  - Scatterplot matrix
  - Heatmaps
  - Survey plots
  - Parallel Coordinates
  - Radviz
  - Star Coordinates

1-D (Univariate) data

Python:

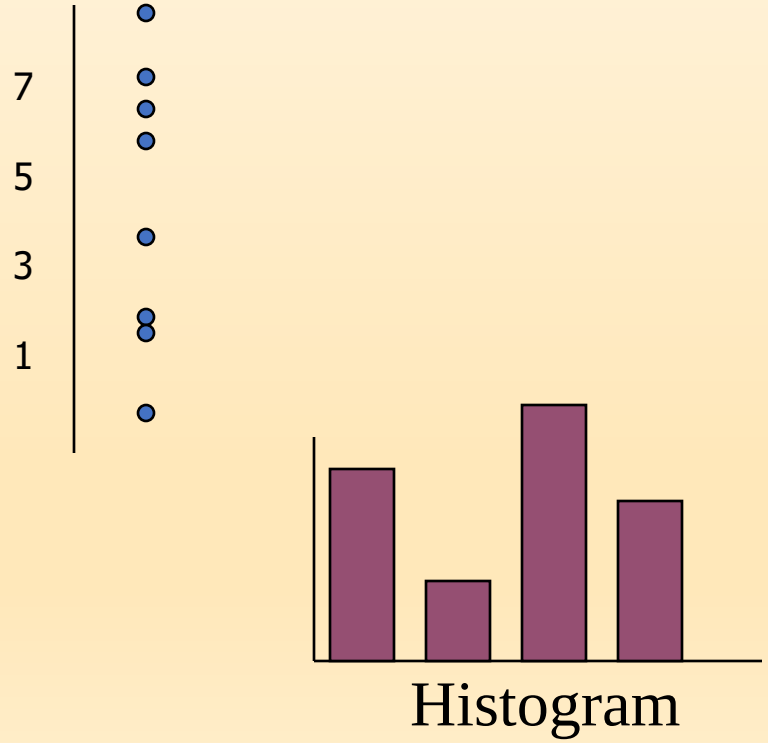
Dotplot is available in the plotnine library

Histogram is available in several library: plotnine, matplotlib, seaborn, plotly, bokeh

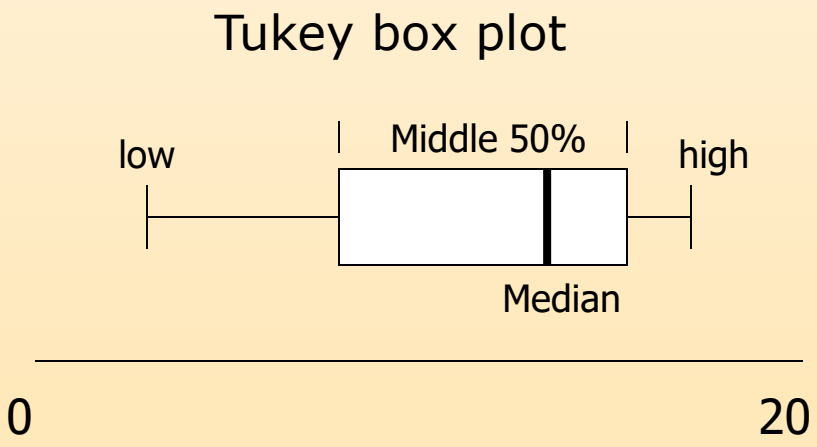
Boxplot si available in plotnine, matplotlib, seaborn, plotly y bokeh

1-D (Univariate) Data

- Representations

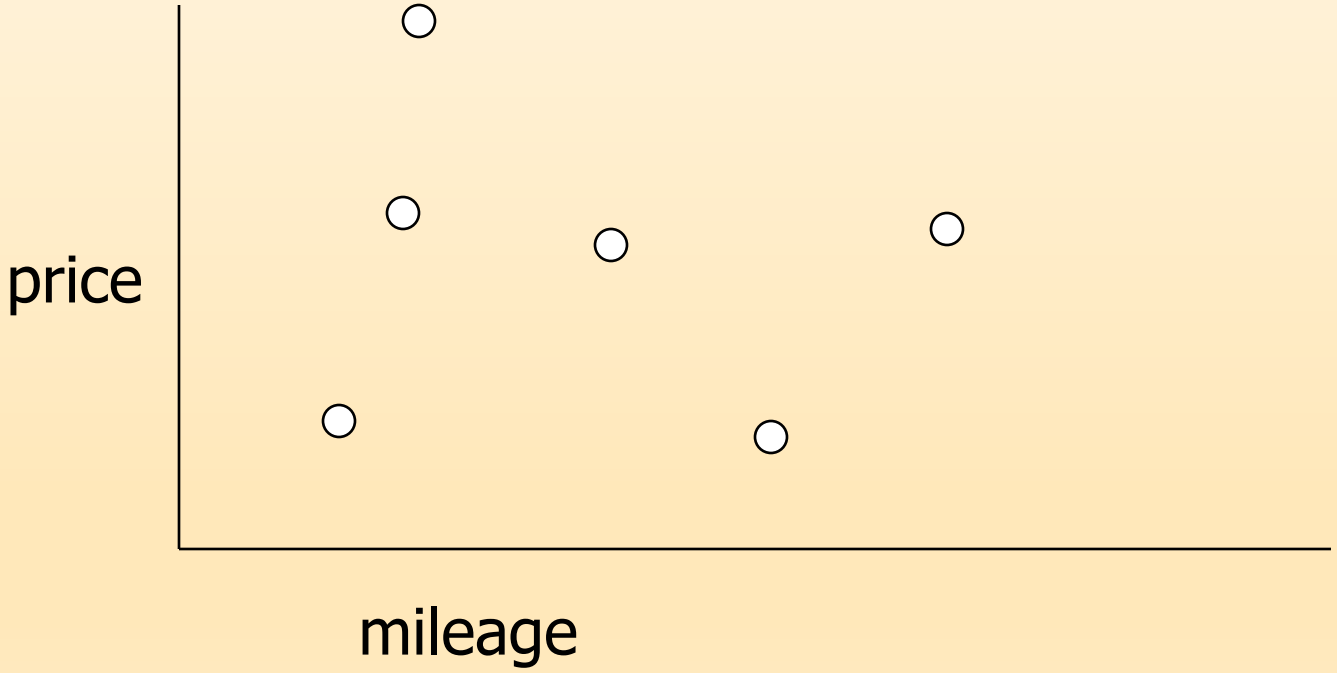


Histogram



2-D (Bivariate) Data

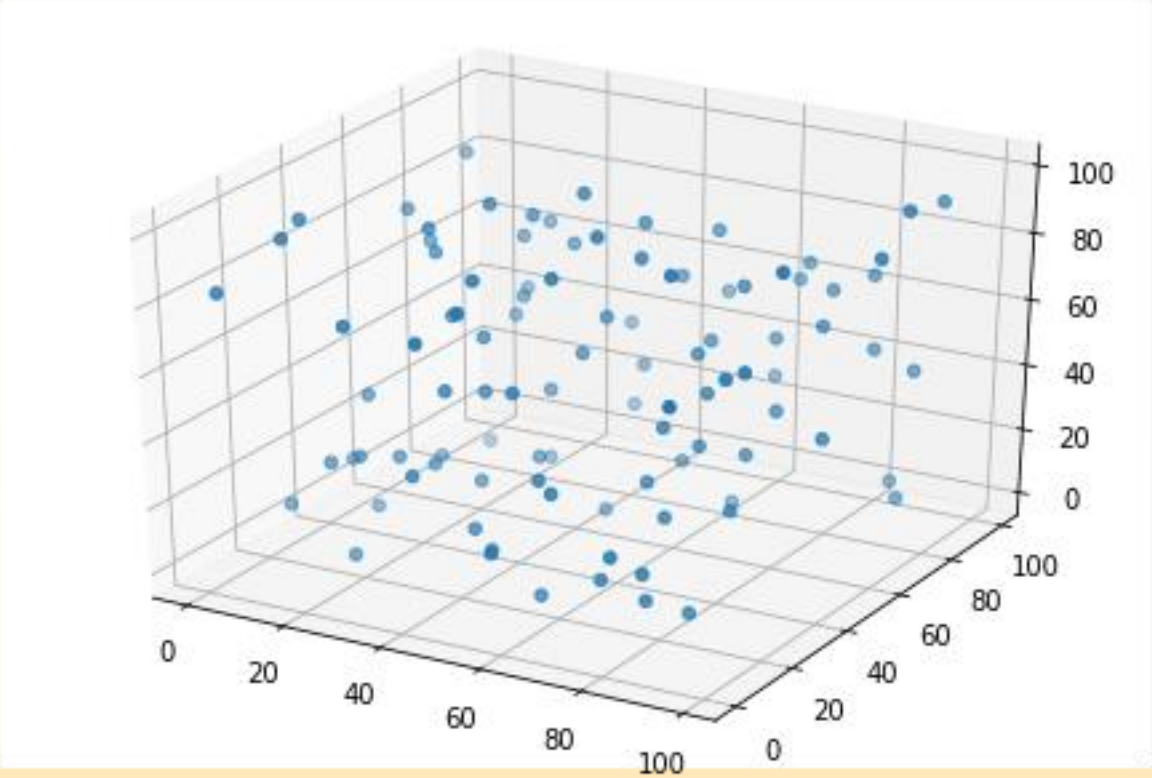
- Scatter plot, …



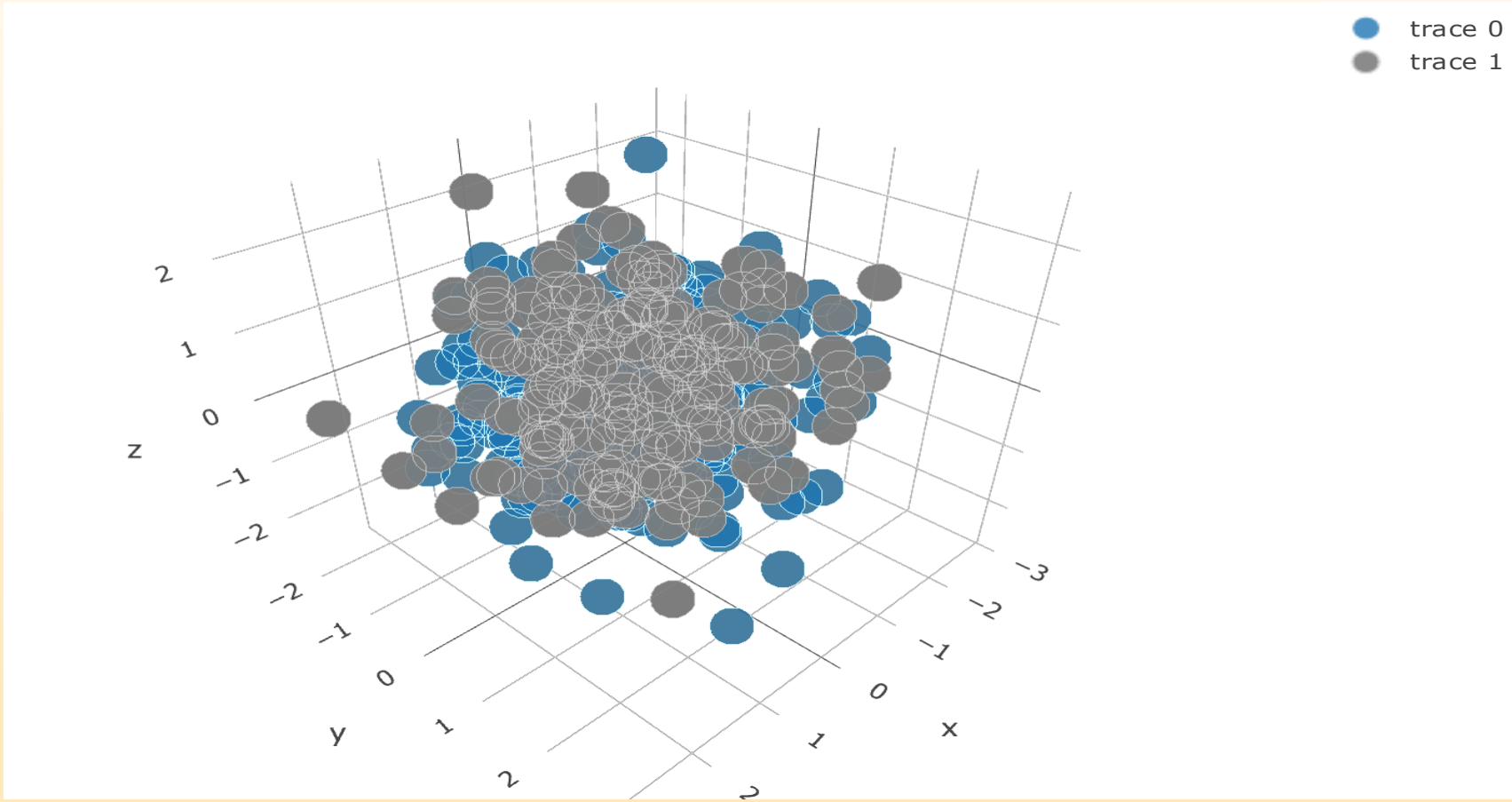
3-D Data

- In Python: Libraries matplotlib, bokeh, plotly have functions to perform scatterplot 3D

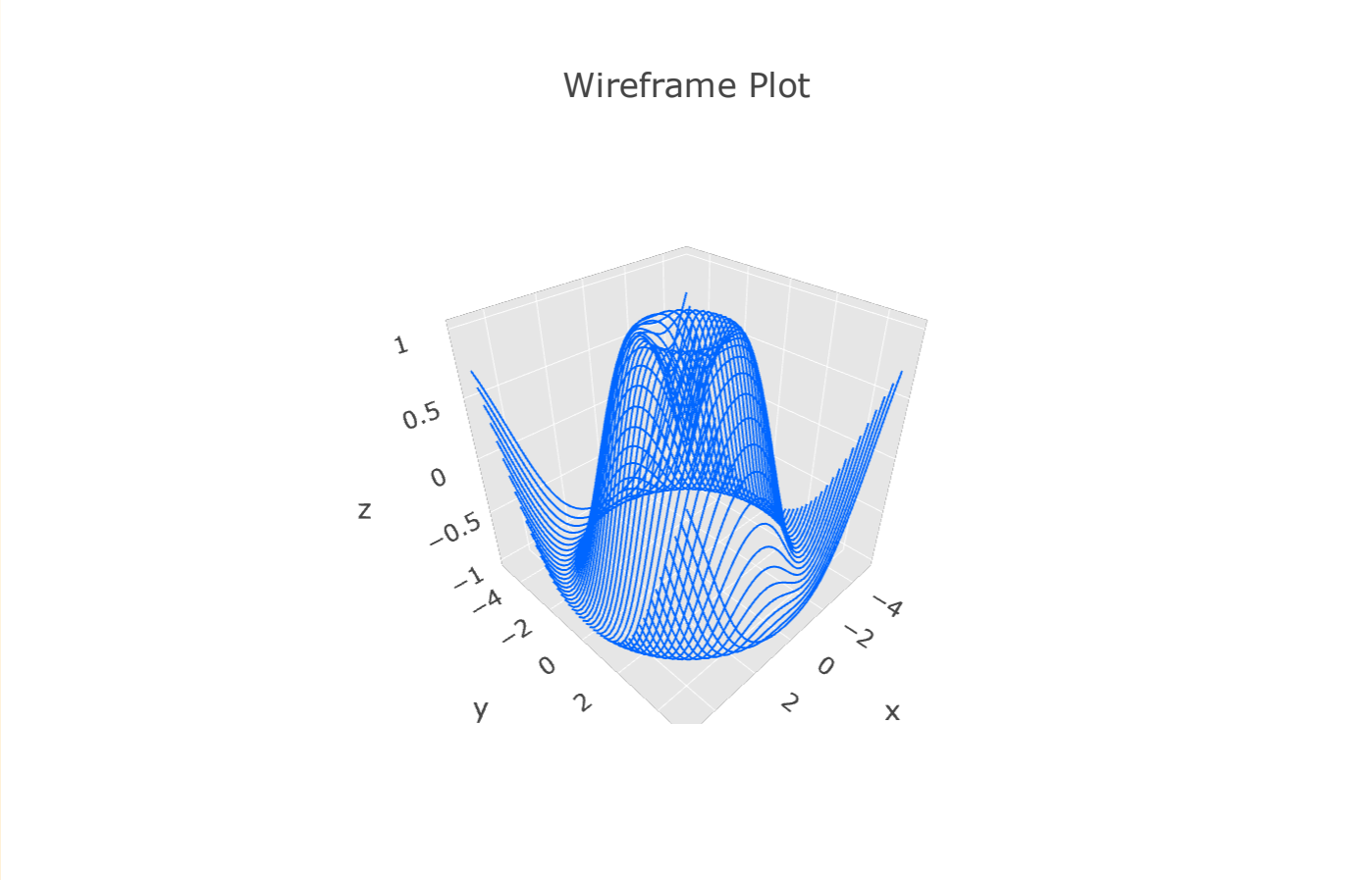
Scatter3d using matplotlib



Scatterplot 3d using plotly



3-D wireframe(plotly)



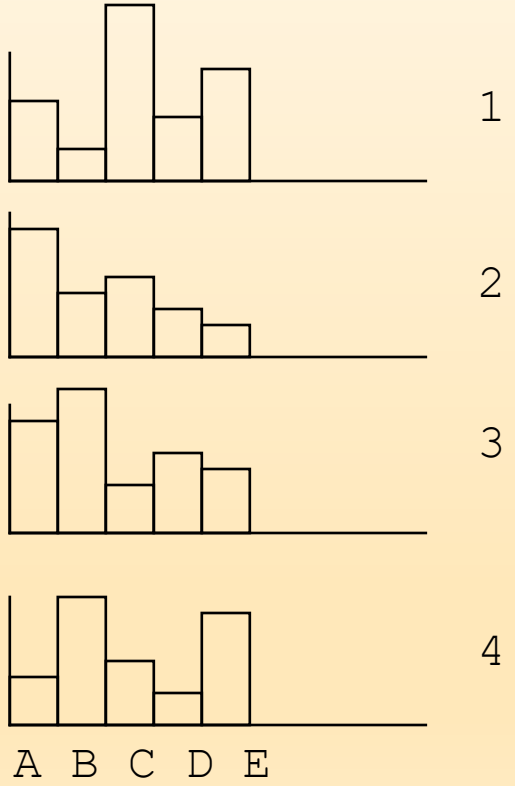
Visualizando en 4 o mas Dimensiones

- Scatterplot Matrix
- Heatmaps
- Parallel coordinate plot
- Radviz
- Star Coordinates

Multiple Views

Give each variable its own display

	A	B	C	D	E
1	4	1	8	3	5
2	6	3	4	2	1
3	5	7	2	4	3
4	2	6	3	1	5



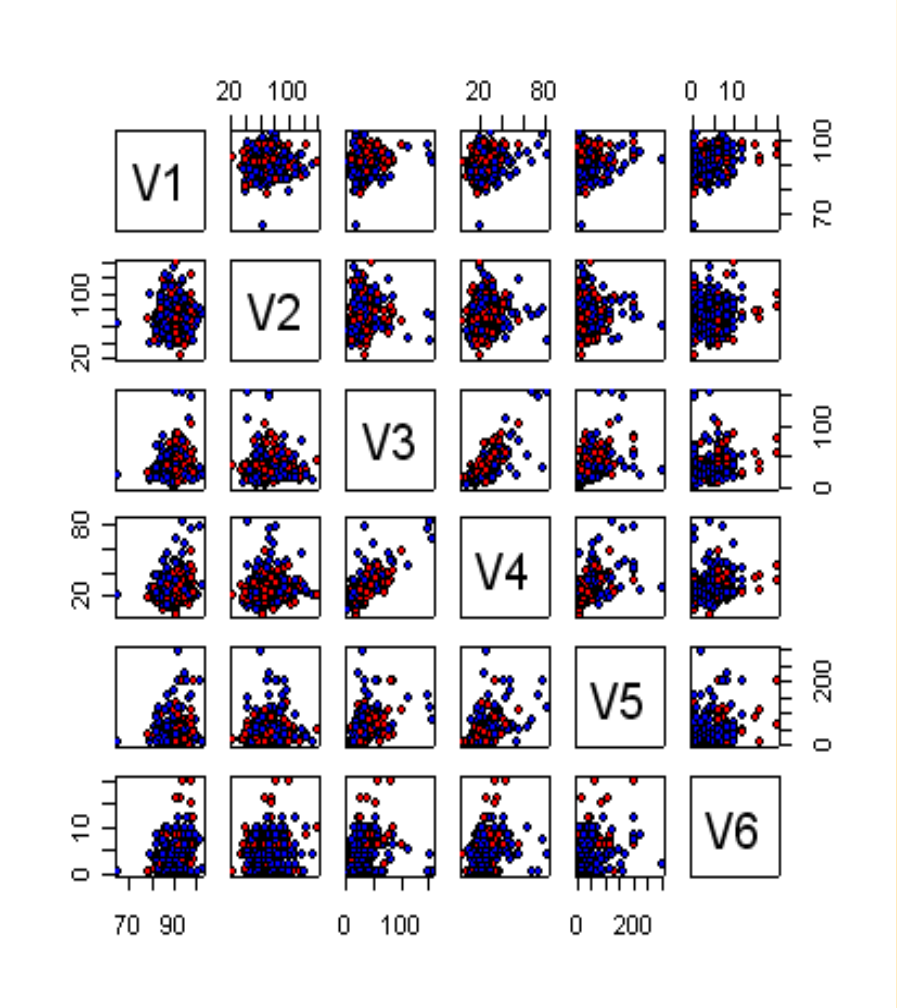
Problem: does not show correlations

Scatterplot Matrix

Represent each possible pair of variables in their own 2-D scatterplot (car data)

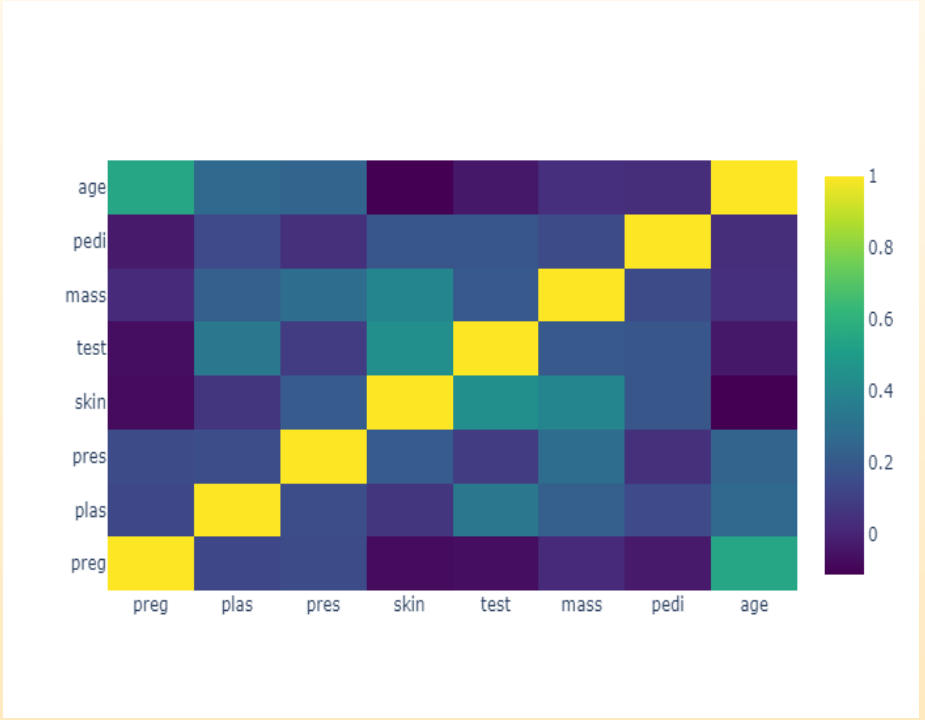
Useful for detecting
linear correlations (e.g. V3 & V4)

But misses
multivariate effects



Heatmaps

Heatmaps visualise data through variations in colouring. When applied to a tabular format, Heatmaps are useful for cross-examining multivariate data, through placing variables in the rows and columns and colouring the cells within the table. Heatmaps are good for showing variance across multiple variables, revealing any patterns, displaying whether any variables are similar to each other, and for detecting if any correlations exist in-between them.

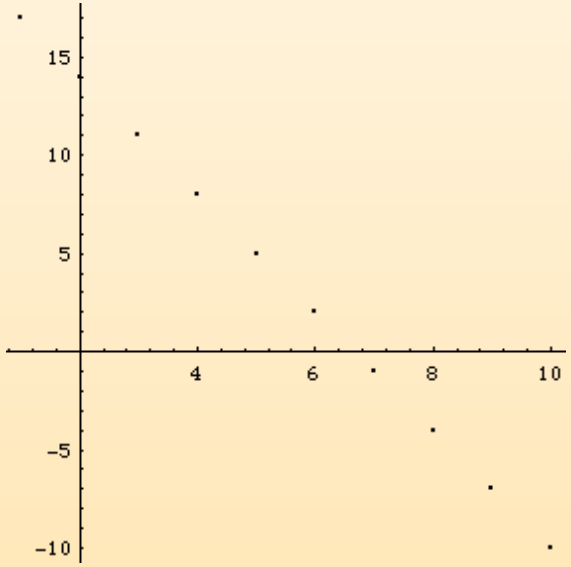


Heatmap to explore the correlations among the features of the diabetes dataset

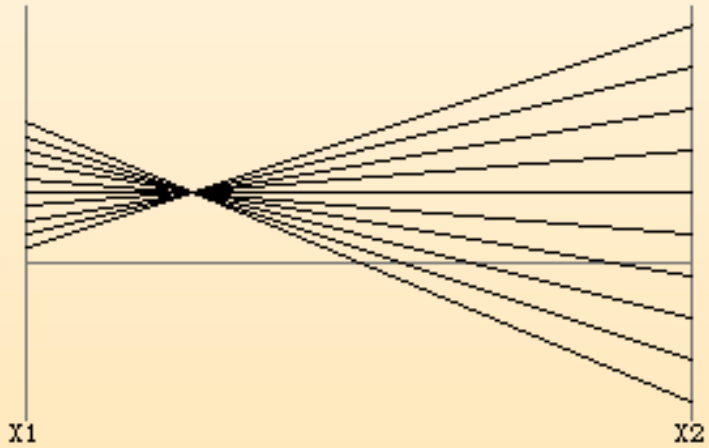
Invented by Alfred Inselberg while at IBM, 1985

Parallel Coordinates

- Encode variables along a horizontal row
- Vertical line specifies values



Dataset in a Cartesian coordinates



Same dataset in parallel coordinates

Invented by Alfred Inselberg while at IBM, 1985

The parallel coordinate plot:

- The parallel coordinate plot, described by Al Inselberg (1985), represents multidimensional data using lines.
- Whereas in traditional Cartesian coordinates all axes are mutually perpendicular, in parallel coordinate plots, all axes are parallel to one another and equally spaced.
- In this approach, a point in *m*-dimensional space is represented as a series of *m*-1 line segments in 2-dimensional space. Thus, if the original data observation is written as $(x_1, x_2, \ldots x_m,)$, then its parallel coordinate representation is the *m*-1 line segments connecting points $(1,x_1)$, $(2,x_2)$, . . . (m,x_m).
- Typically, features will be standardized before a parallel coordinate plot is drawn.

Example: Visualizing Iris Data

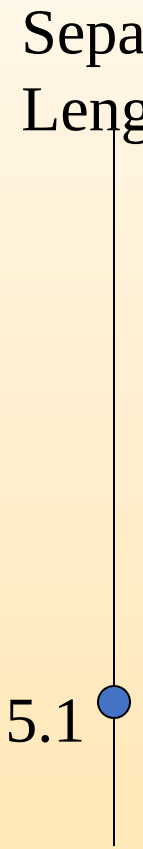
Iris setosa

sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2
4.9	3	1.4	0.2
…	…	…	…
5.9	3	5.1	1.8

Iris versicolor

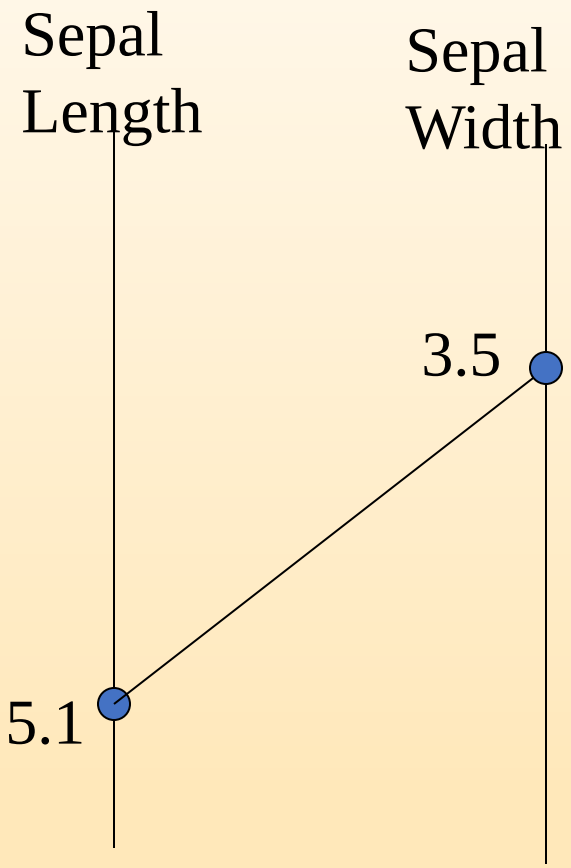
Iris virginica

Parallel Coordinates



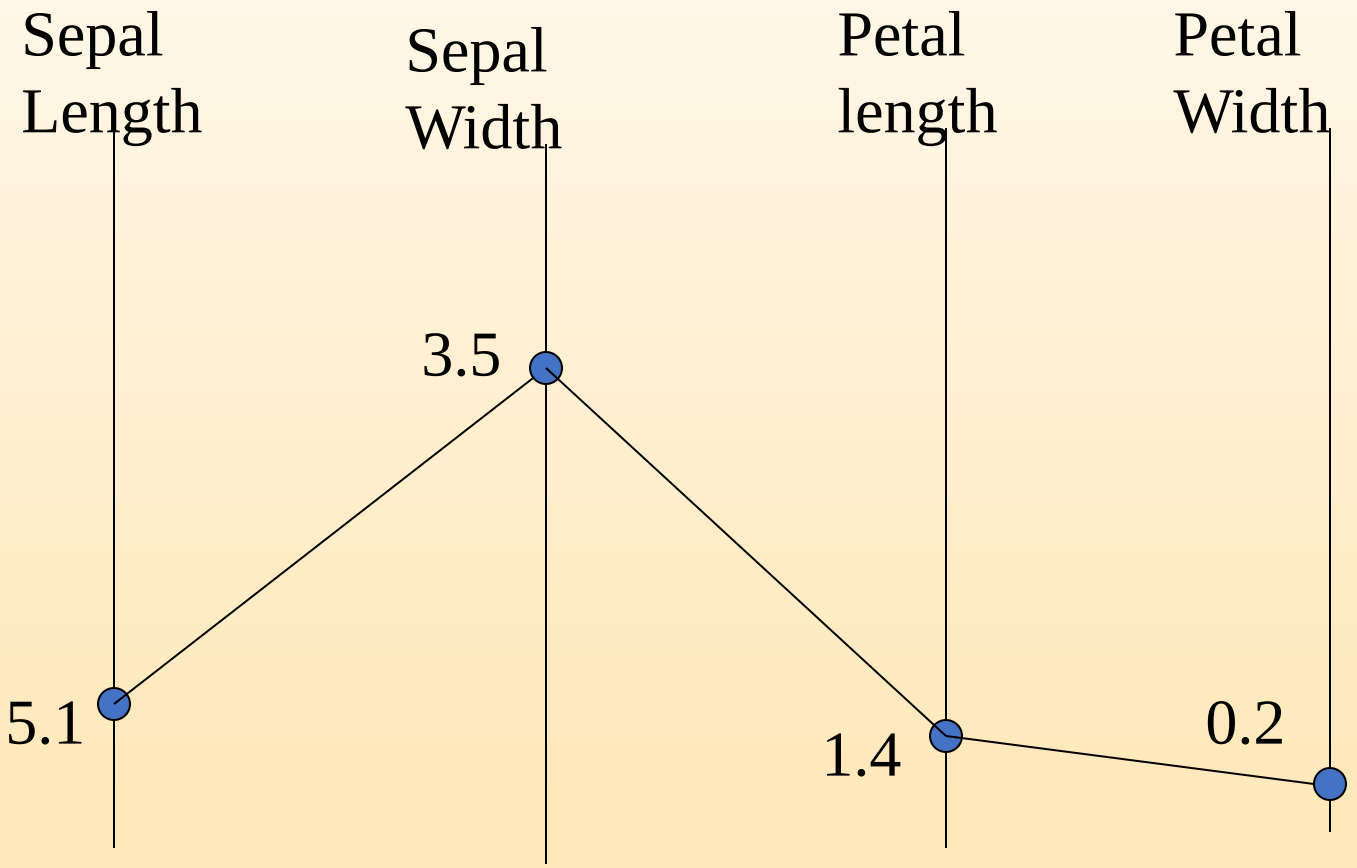
sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2

Parallel Coordinates: 2 D



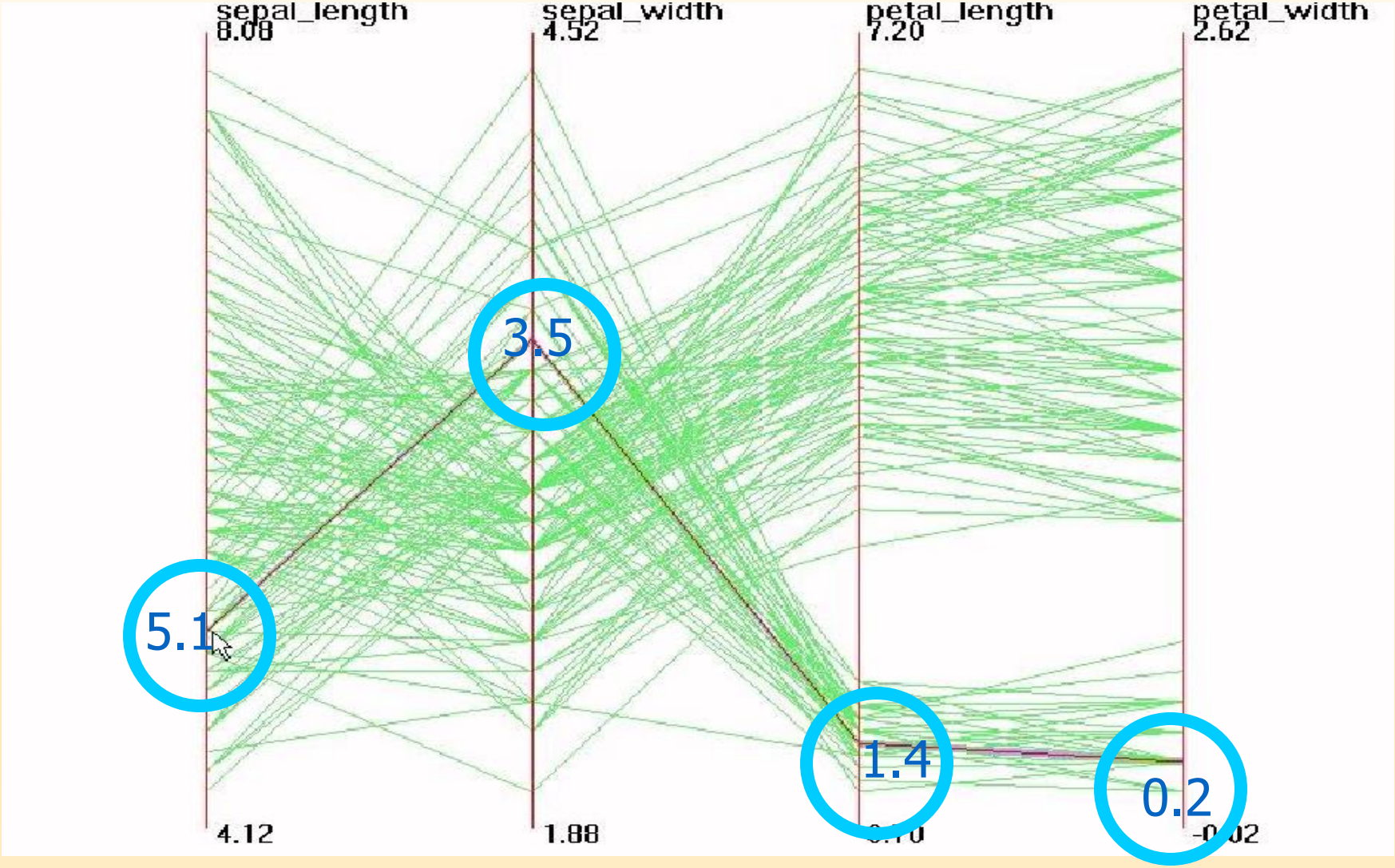
sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2

Parallel Coordinates: 4 D



sepal length	sepal width	petal length	petal width
5.1	3.5	1.4	0.2

Parallel Visualization of Iris data



Parallelplot (cont)

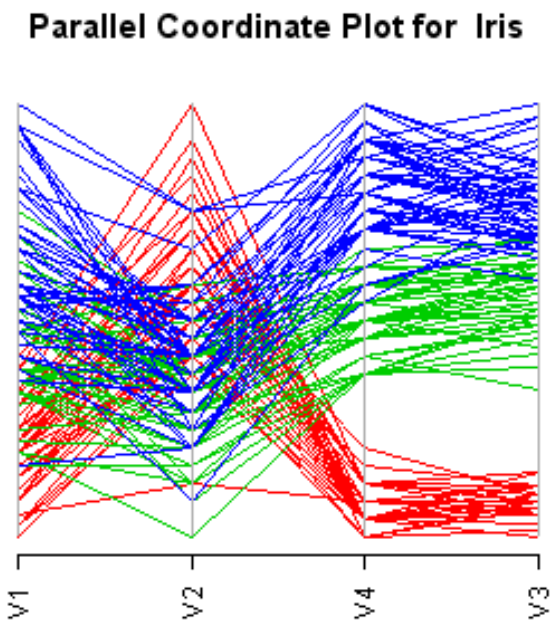
- Pairwise comparison is limited to those axis that are adjacent.
- For a dataset with p attributes there are p! permutations of the attributes so each of them is adjacent to every attribute in some permutation.
- Wegman (1990) determined that only [(p+1)/2] permutations are needed.([.] is the greatest integer function).

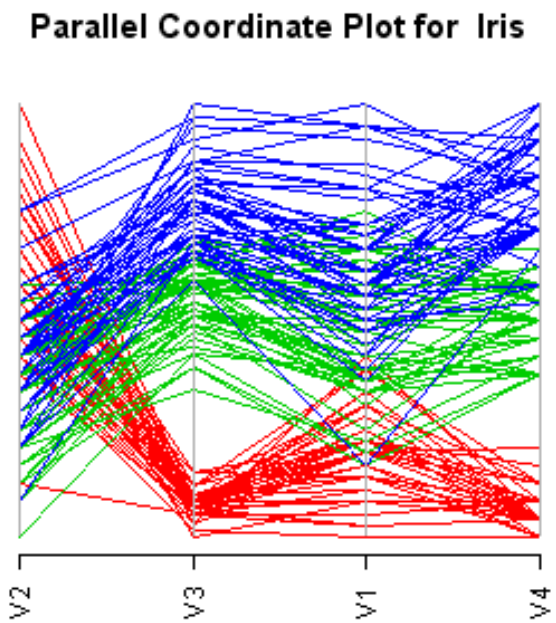
The parallel coordinate plot

parallelplot(dataset: matrix , name: string, class: integer,
comb: integer, obs: list of integer)

Iris dataset:

- **Data on the flowers.**
- **4 attributes (sepal length, sepal width petal length, and petal width,)**
- **150 instances**
- **3 classes (Setosa, Versicolor, Virginica)**
- **No missing values.**





Interpretation:

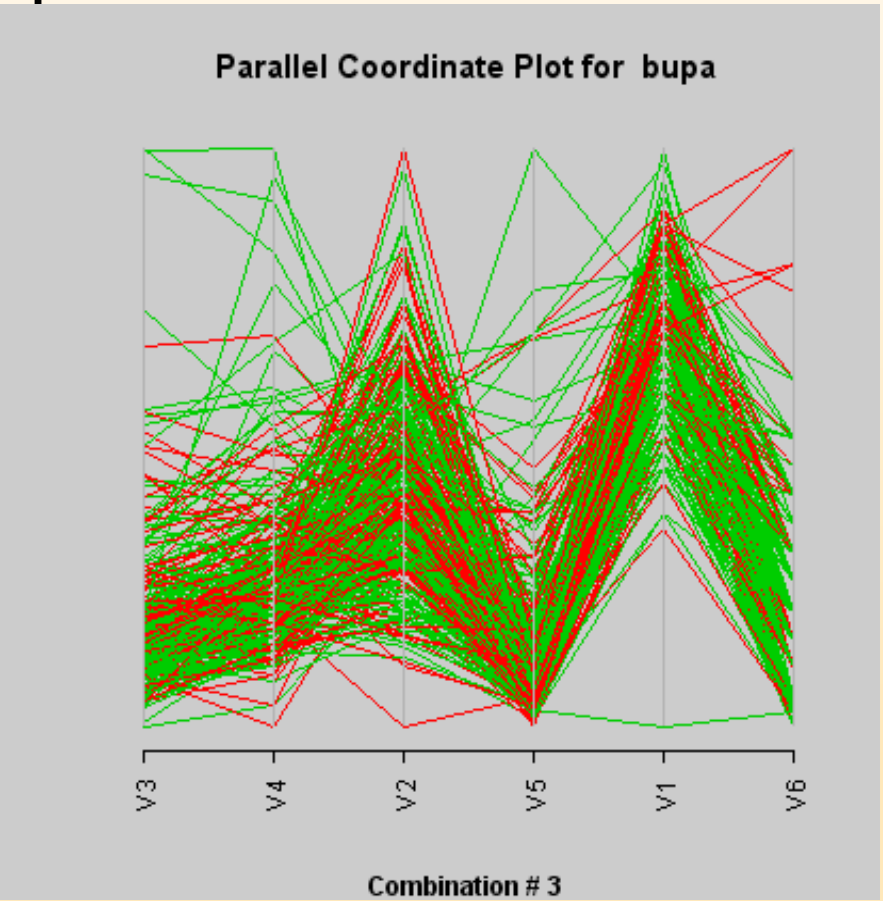
- **Each different color represents a different class.**
- **If two attributes are highly positively correlated, lines passing from one feature to another tend not to intersect between the parallel coordinate axes.**

The parallel coordinate plot

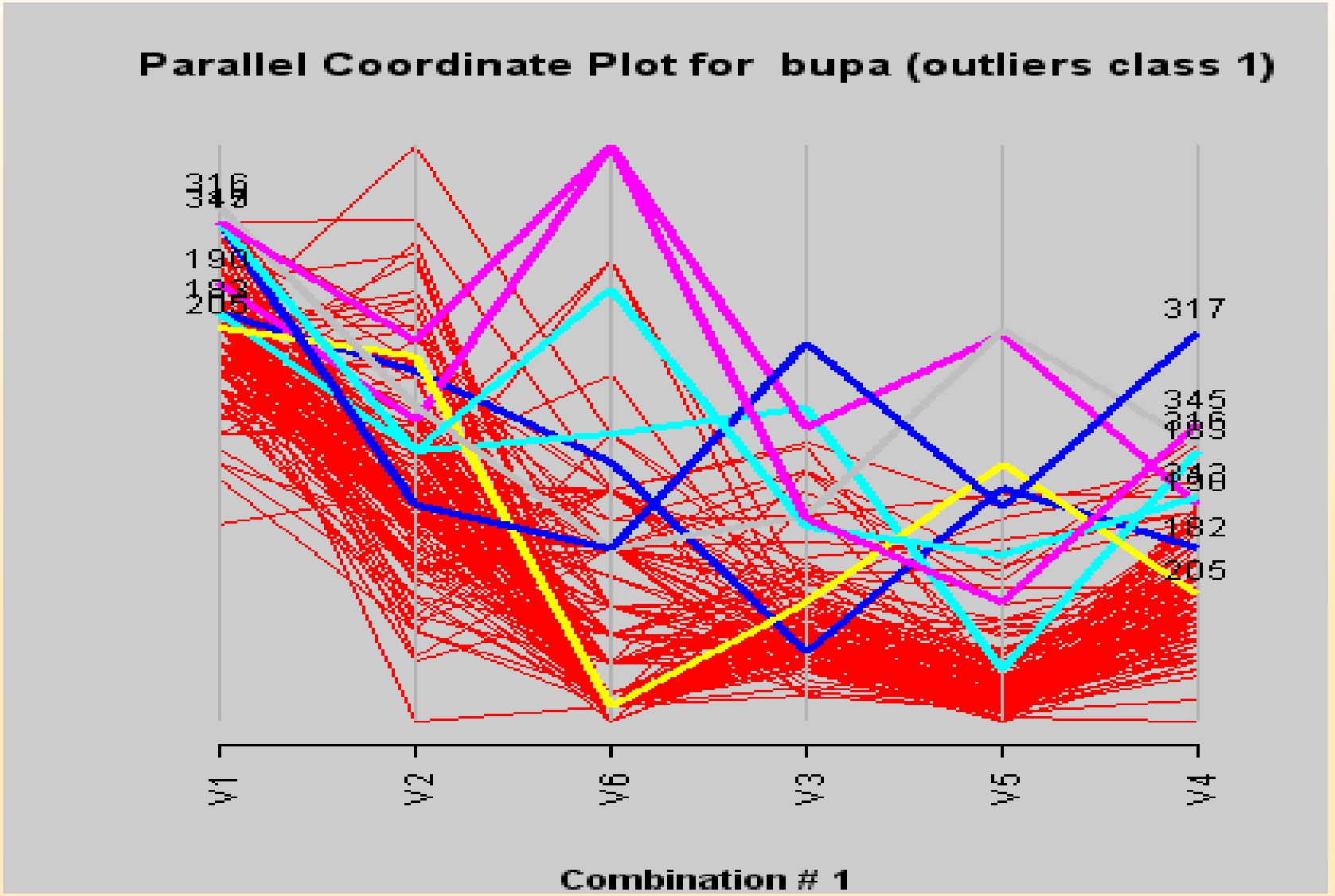
- **For highly negatively correlated attributes, the line segments tend to cross near a single point between the two parallel coordinate axes.**
- **The presences of outliers is suggested by poly-lines that do not follow the pattern for their class.**

Some discrimination can be observed for several features.

One limitation of this displays is the loss of the information that is encoded into the lines between the axes for discrete, heterogeneous data attributes.



Parallelplot as a tool to detect outliers



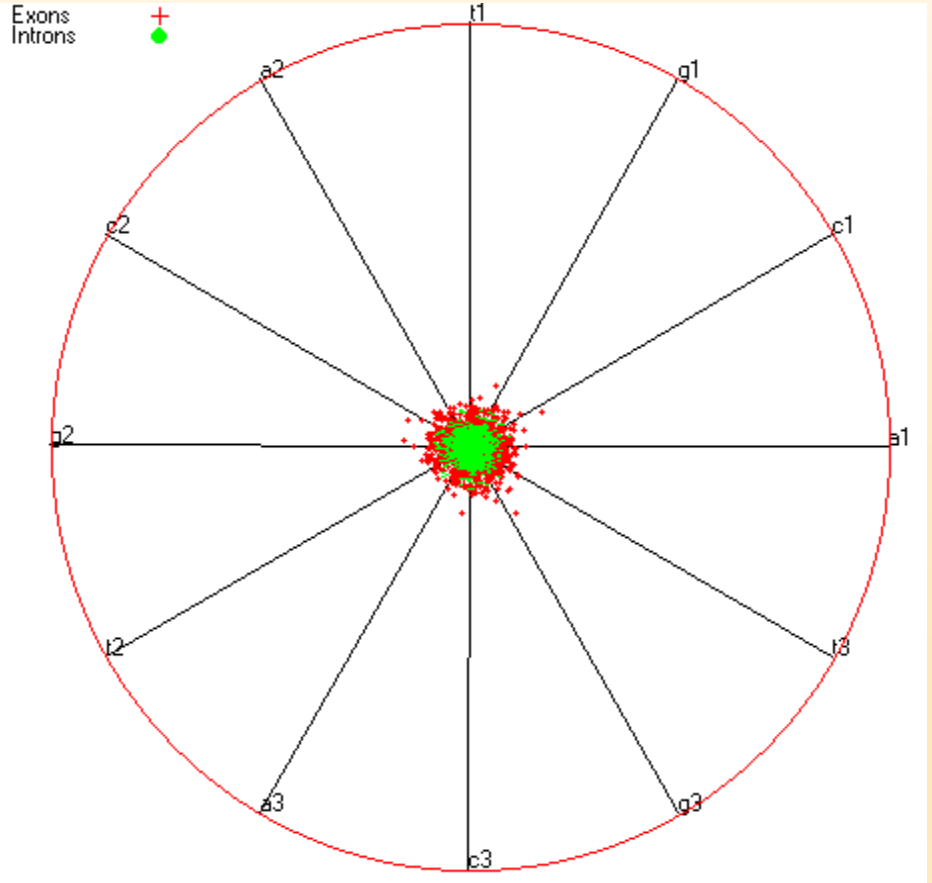
Parallel Visualization Summary

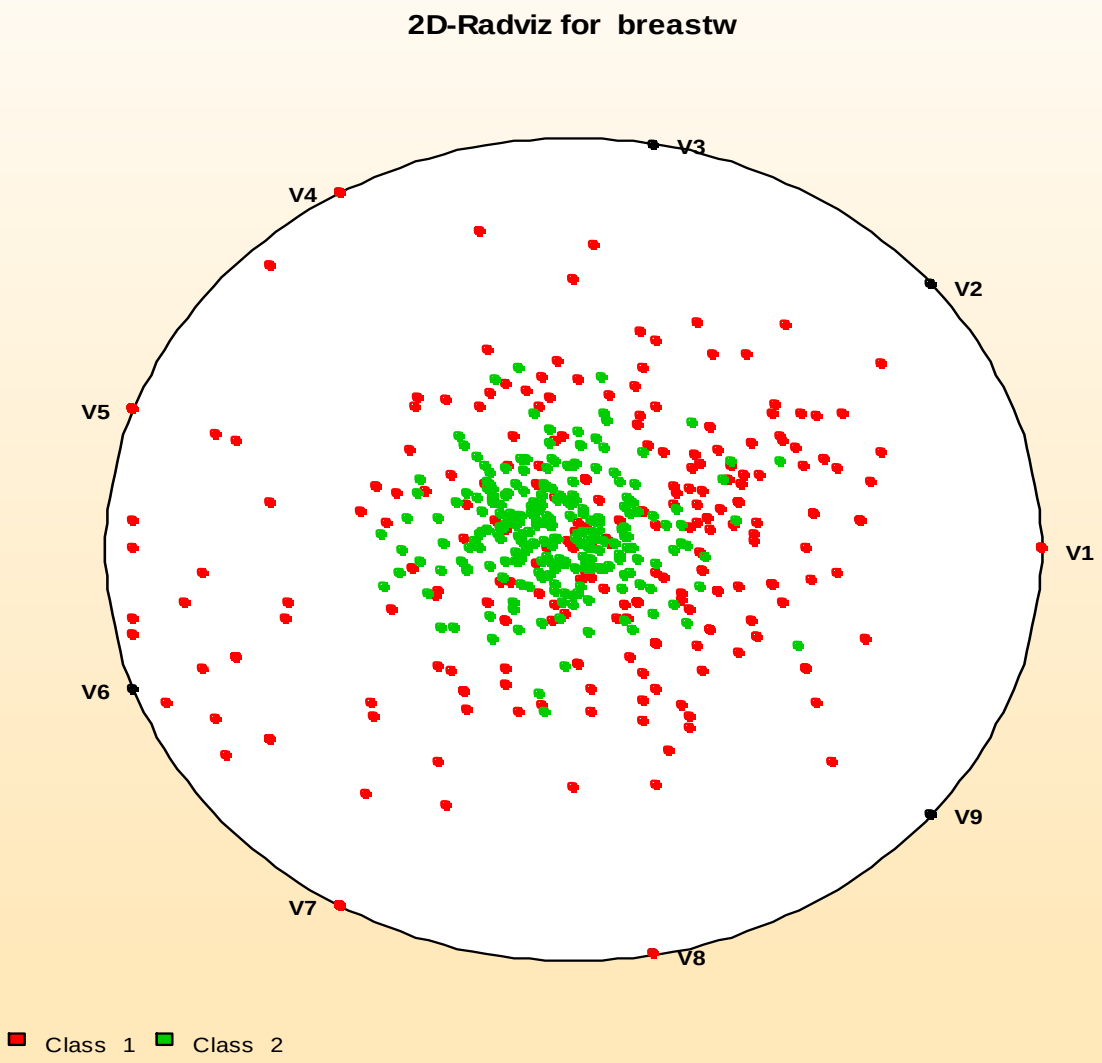
- Each data point is a line
- Similar points correspond to similar lines
- Lines crossing over correspond to negatively correlated attributes
- Interactive exploration and clustering

- Problems: order of axes, limit to ~20 dimensions
- Esta disponible en Pandas, Plotly y Yellowbrick.

RadViz (Ankerst, et al., 1996)

- a radial visualization
- One spring for each feature .
- One end attached to perimeter point where the feature position is located. The other end attached to a data point.
- Each data point is displayed inside the circle where the sum of the spring forces equals 0.
- Good for outlier detection
- Esta disponible en Pandas, Plotly y en Orange,Yellowbrick



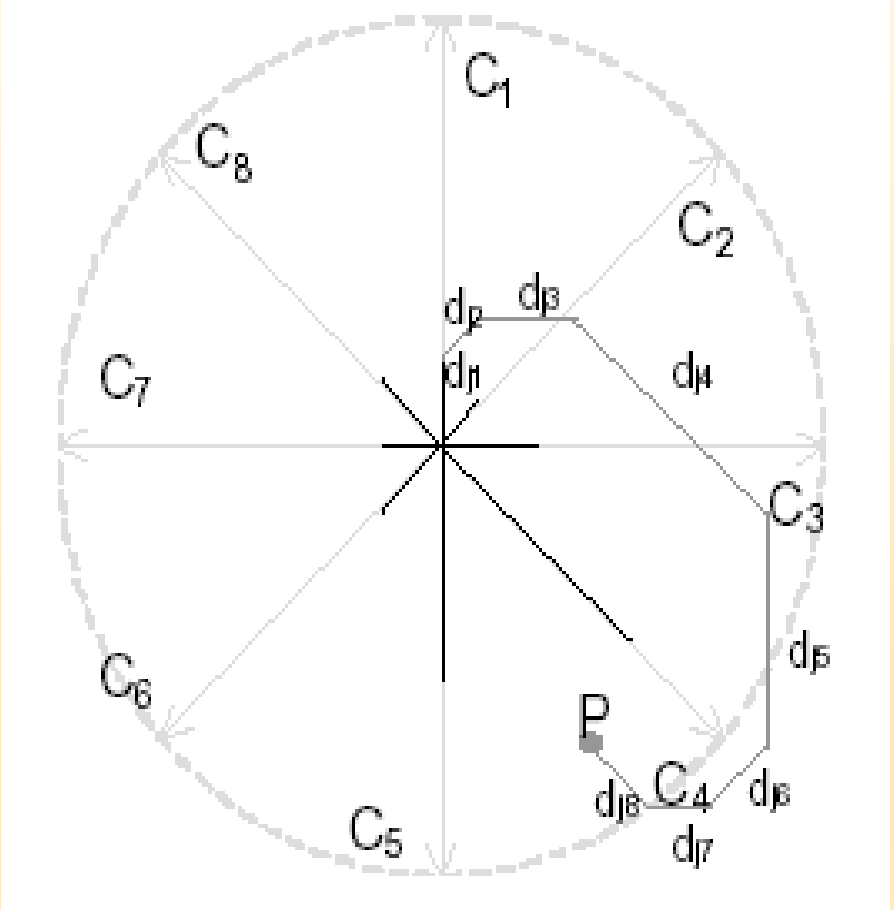

2D-Radviz for breastw
V3
V4
V2
V5
V1
V6
V9
V7
V8
Class 1
Class 2

Star Coordinates (Kandogan, 2001)

- Each dimension shown as an axis
- Data value in each dimension is represented as a vector.
- Data points are scaled to the length of the axis

  - min mapping to origin

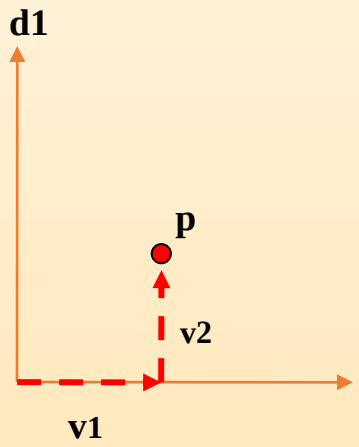
  - max mapping to the end



Star Coordinates Contd

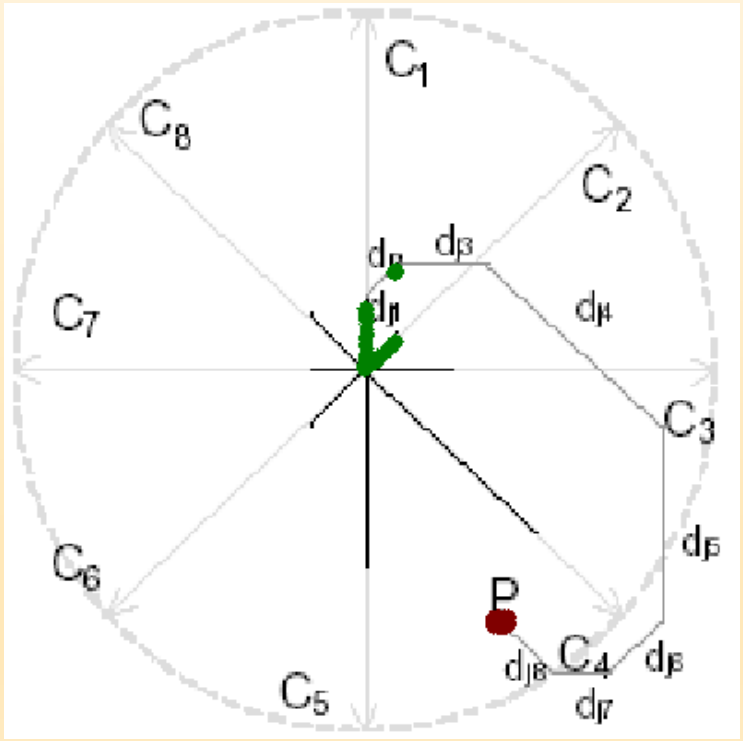
Cartesian

P=(v1, v2)



Star Coordinates

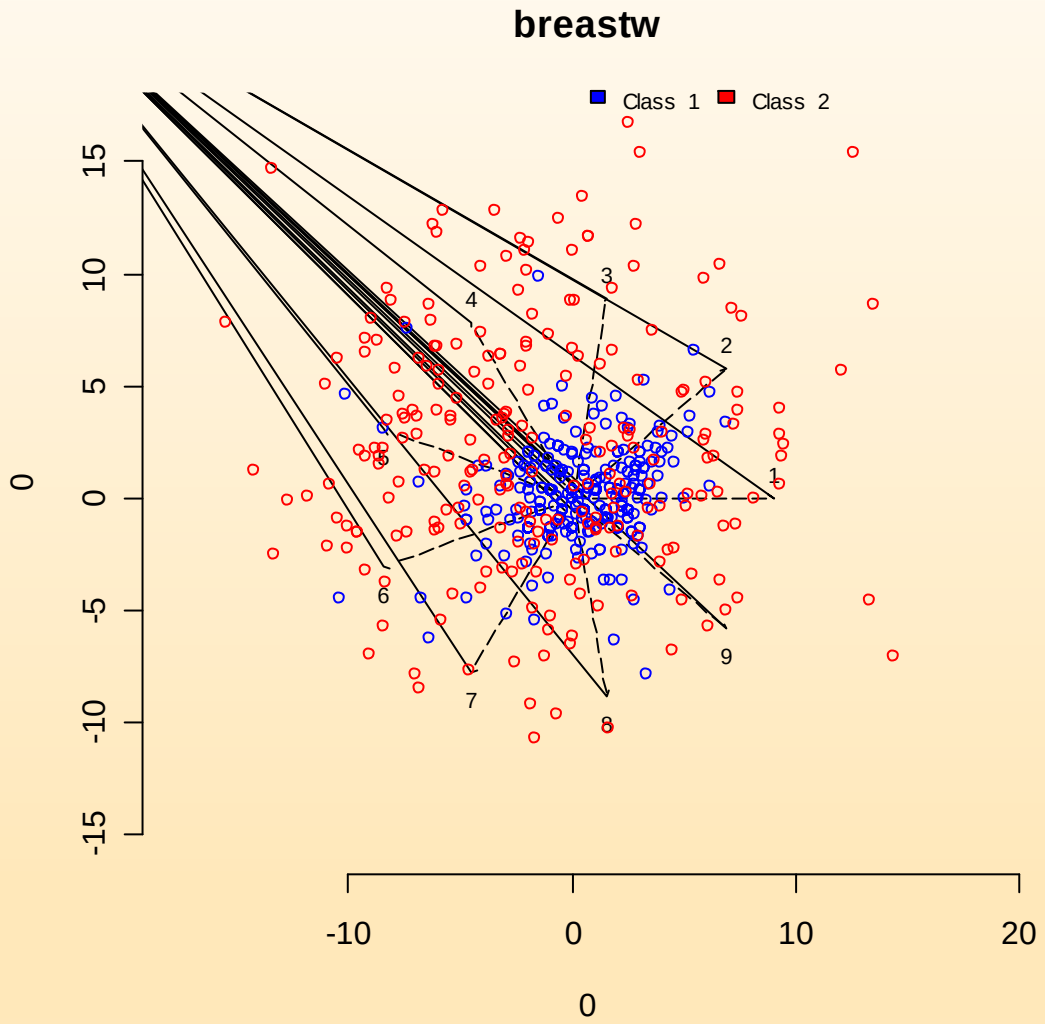
P=(v1,v2,v3,v4,v5,v6,v7,v8)



Mapping:

- Items → dots
- Σ attribute vectors → position

starcoord(breastw,main="breastw",class=T)



Visualization software

Free and Open-source

- Ggobi (before was xgobi). Built using Gtk. Interface with databases systems. Runs on Windows and Linux. http://www.ggobi.org/
- XmdvTool. The multivariate data visualization tool. Available for Linux and Windows. Built using OpenGL and Tcl/Tk. See http://davis.wpi.edu/~xmdv/

- Many more - see www.kdnuggets.com/software/visualization.html
- Tensorboard: Interface for visualization for Tensorflow.

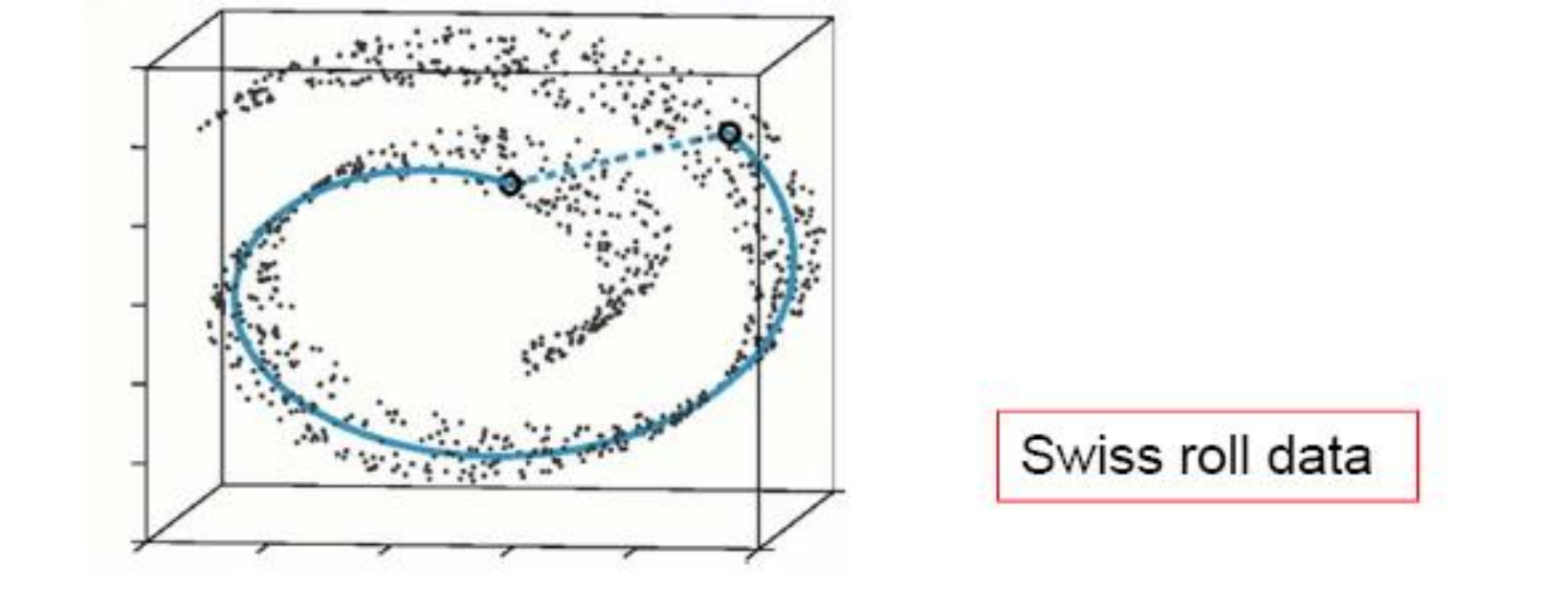
TensorFlow is an end-to-end platform that makes it easy for you to build and deploy ML models. Developed by Google since 2015.

PCA: Pros and cons

- Pros:
- Helps reduce computational complexity.
- Can help supervised learning.
- PCA can also be seen as noise reduction.
- Cons:
  - Fails when data consists of multiple separate clusters.
  - Directions of greatest variance may not be most informative.
- Practical issue: covariance matrix is n x n.
  - E.g. for image data Σ = 32768 x 32768.
  - Finding eigenvectors of such a matrix is slow. Use SVD.

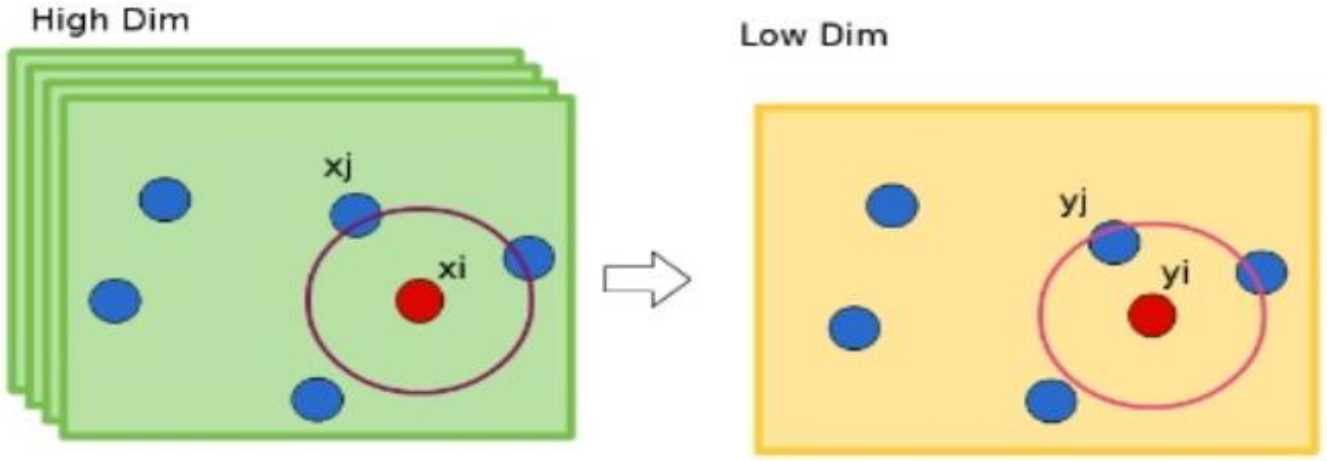
Nonlinear dimensionality reduction

- Data often lies on or near a nonlinear low-dimensional surface
- Such low-dimensional surfaces are called *manifolds*.



t-Stochastic neighbor embedding (t-SNE)

Measure pairwise similarities between high-dimensional and low-dimensonal objects



$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$$

Figure taken from K. Zhao's presentation (10/2014)

t-Stochastic neighbor embedding (t-SNE)

Converting the high-dimensional Euclidean distances into conditional probabilities that represent similarities

- Similarity of datapoints in High Dimension

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_k||^2/2\sigma_i^2)}$$

- Similarity of datapoints in Low Dimension

$$q_{j|i} = \frac{exp(-||y_i - y_j||^2)}{\sum_{k \neq i} exp(-||y_i - y_k||^2)}$$

- Cost function

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

Minimize the cost function using gradient descent

t-Stochastic neighbor embedding (t-SNE)

The t distribution isused to approximate probabilities $q_{i/}$

An important feature of t-SNE is a tuneable parameter, “perplexity,” which says (loosely) how to balance attention between local and global aspects of your data. The parameter is, in a sense, a guess about the number of close neighbors each point has. The perplexity value has a complex effect on the resulting pictures.

In the original paper (van der Maaten and Hinton, JMLR, 2008) is stated that *“The performance of t-SNE is fairly robust to changes in the perplexity, and typical values are between 5 and 50.”*

The t-SNE algorithm doesn’t always produce similar output on successive runs. There are additional hyperparameters related to the optimization process.

j

The bees_2p dataset

The activity (number of enter-exit from the hive) of 382 bees during 10 days is registered. The number of activities during morning and afternoon is counted per each day.

The MNIST dataset

It was used by LeCun, Cortes and Burges (1998) for handwritten recognition. The training dataset consists of 60k images of the digits 0 to 9 and the test dataset consists of 10k images.
Each row of the datasets has 785 entries containing 784 pixels (28x28) and the first entry is the digit label.
The training dataset is 104MB and the test is 17MB.

Visualization of classes in MNIST data using PCA

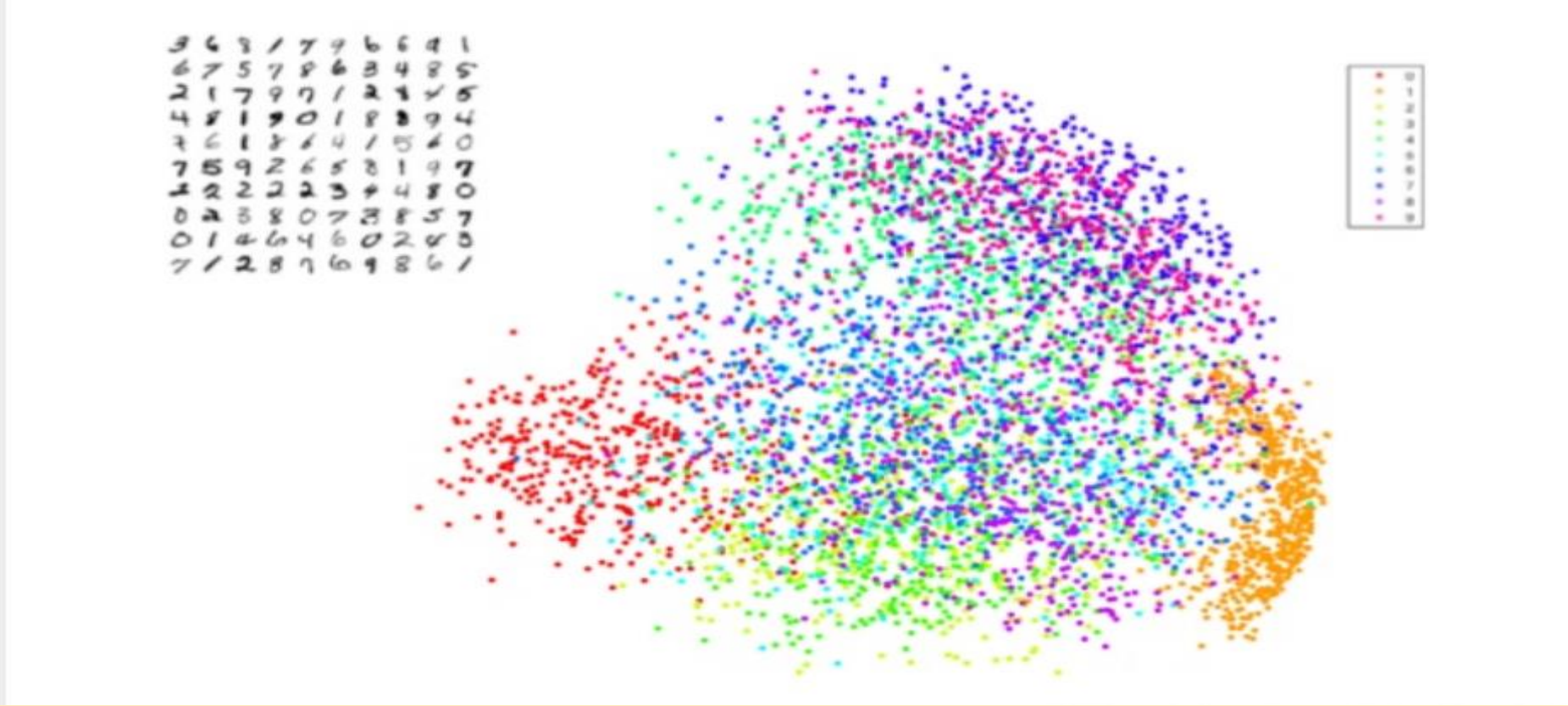
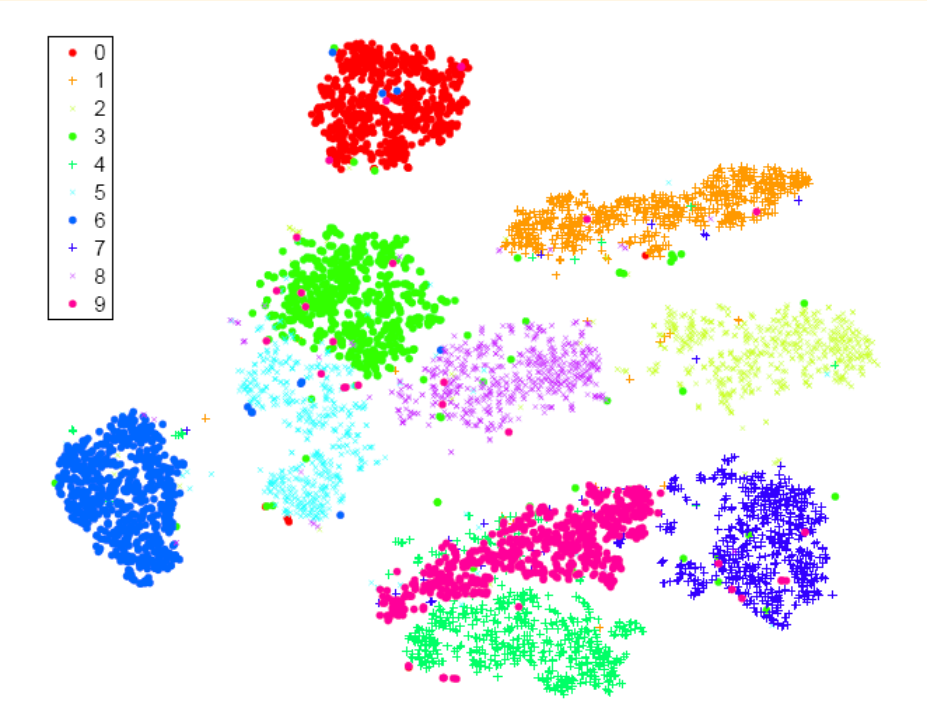
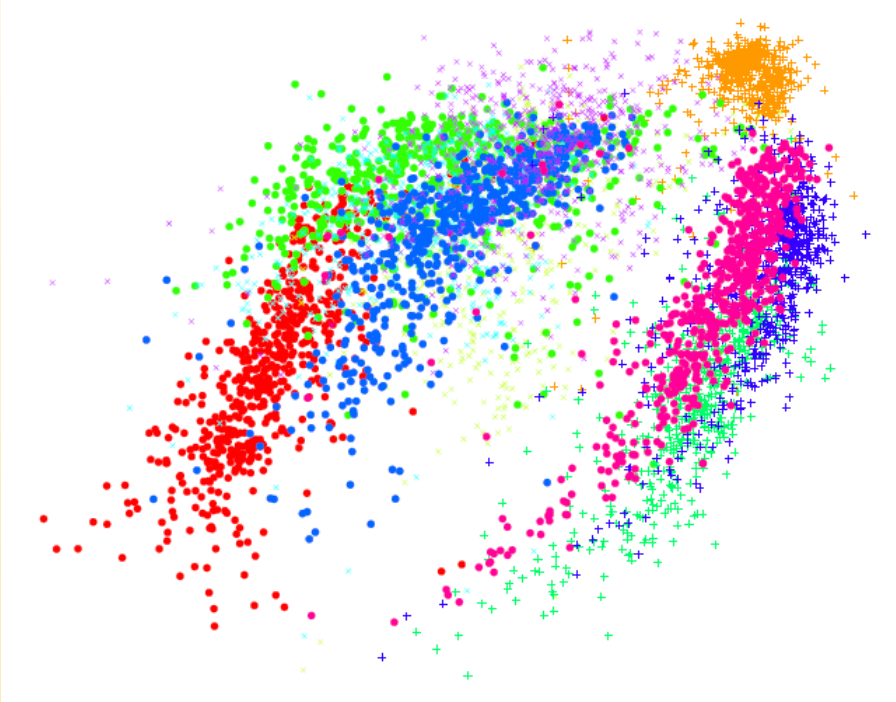


Figure taken from K. Zhao's presentation (10/2014)

Visualization of classes in MNIST data

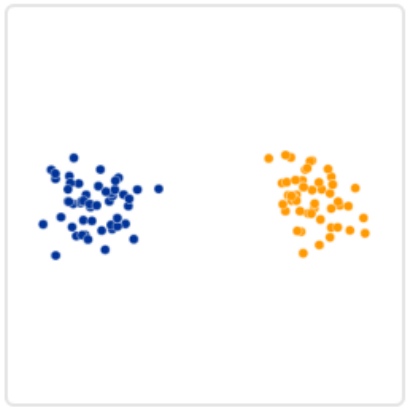


t-SNE

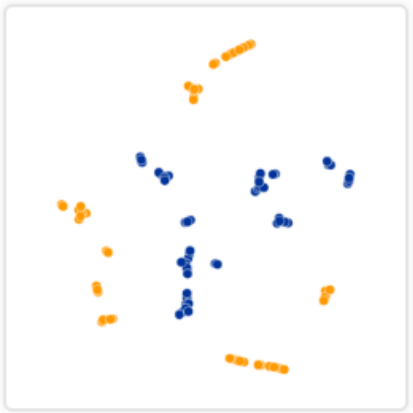


ISOMAP

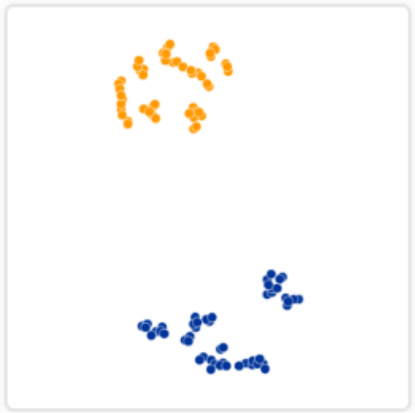
Effect of the perplexity parameter



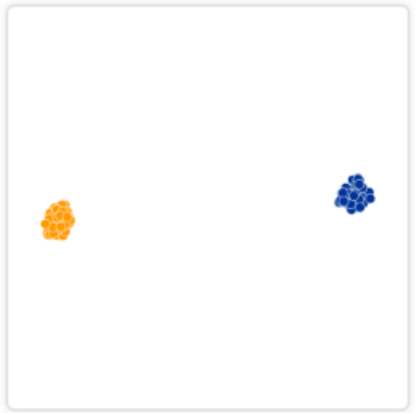
Original



Perplexity: 2
Step: 5,000



Perplexity: 5
Step: 5,000



Perplexity: 30
Step: 5,000

Perplexity: 50
Step: 5,000

t-Stochastic neighbor embedding (t-SNE)

t-SNE has a quadratic time and space complexity in the number of data points. This makes it particularly slow and resource draining while applying it to data sets comprising of more than 10,000 observations.

In order to reduce the time complexity the creators of t-SNE recommend to apply first PCA to reduce the number of dimension, say to 50, and after that apply t-SNE.

t-SNE of a data can be obtained either using the submodule Manifold of sklearn or using Tensorboard.

t-SNE using tensorboard

1-In your Jupyter notebook include the following two lines:

```
import tensorflow as tf
from tensorflow.contrib.tensorboard.plugins import projector
```

2-Create a log directory, say mylog, in your main directory (its name can be obtained using os.getcwd()). If you have supervised data, put the labels in a file called metadata.tsv. This file has to be in mylog.

3-Run the jupyter notebook

4-Activate your tensorboard environment

5-Execute the command tensorboard --logdir=c:/Users/eacun/mylog

The path is red must be changed according to the value of os.getcwd() .

t-SNE using tensorboard

(tensorflow_env) C:\Users\eacun>tensorboard --logdir=c:/Users/eacun/logs3
Serving TensorBoard on localhost; to expose to the network, use a proxy or pass --bind_all
TensorBoard 2.1.0 at http://localhost:6006/ (Press CTRL+C to quit)
Open in your browser http://localhost:6006/

Sometimes you have to try more than one time in order to open the browser

t-SNE using tensorboard