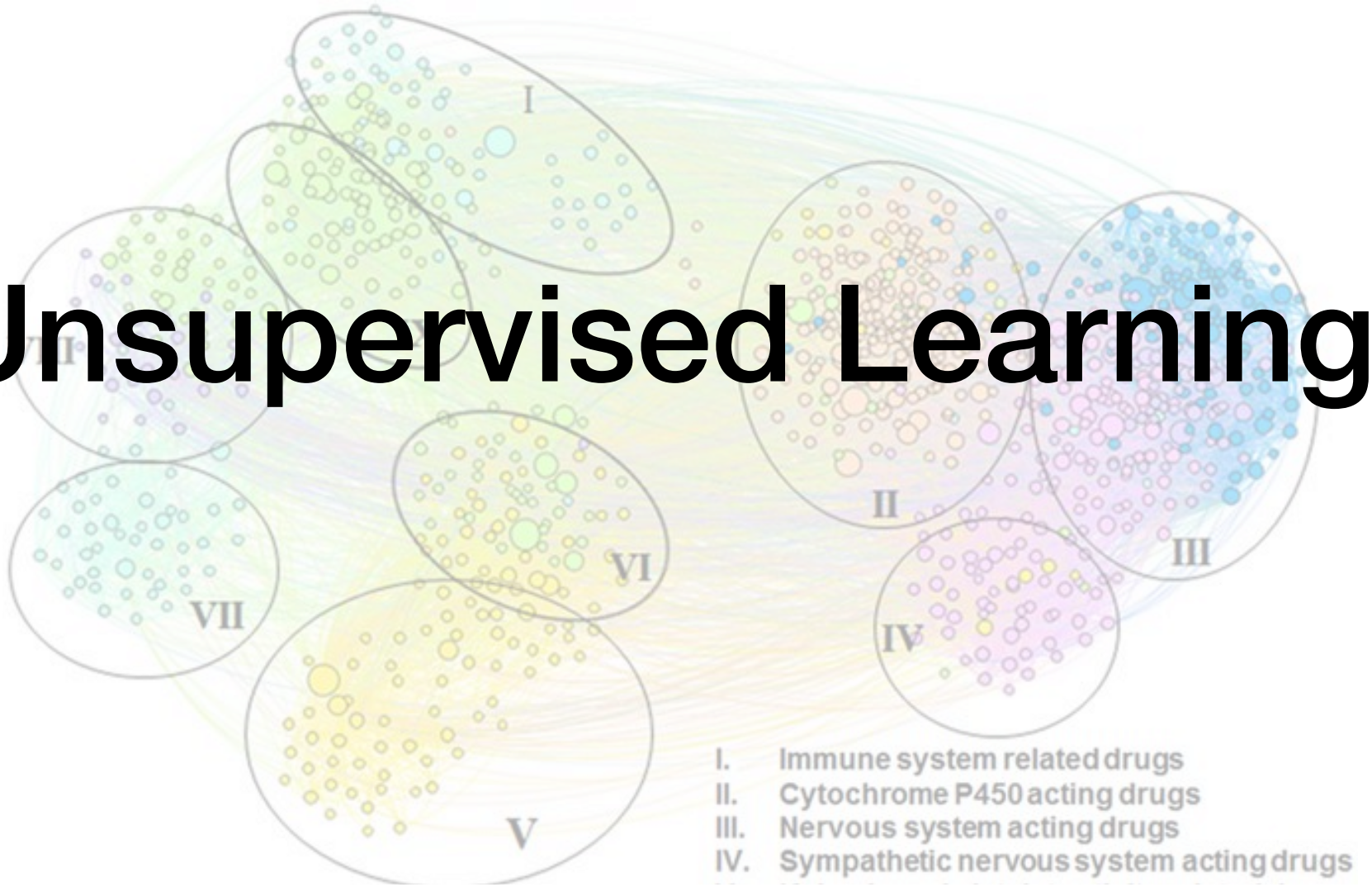
Unsupervised Learning



Dr. Fabien Plisson

Chemoinformatics in Drug Discovery

LANGEBIO, UGA CINVESTAV

October 15-18, 2019 - Irapuato, Mexico

Program

Introduction to Unsupervised Learning

Clustering methods

Dimension(ality) Reduction methods

Program

Introduction to Unsupervised Learning

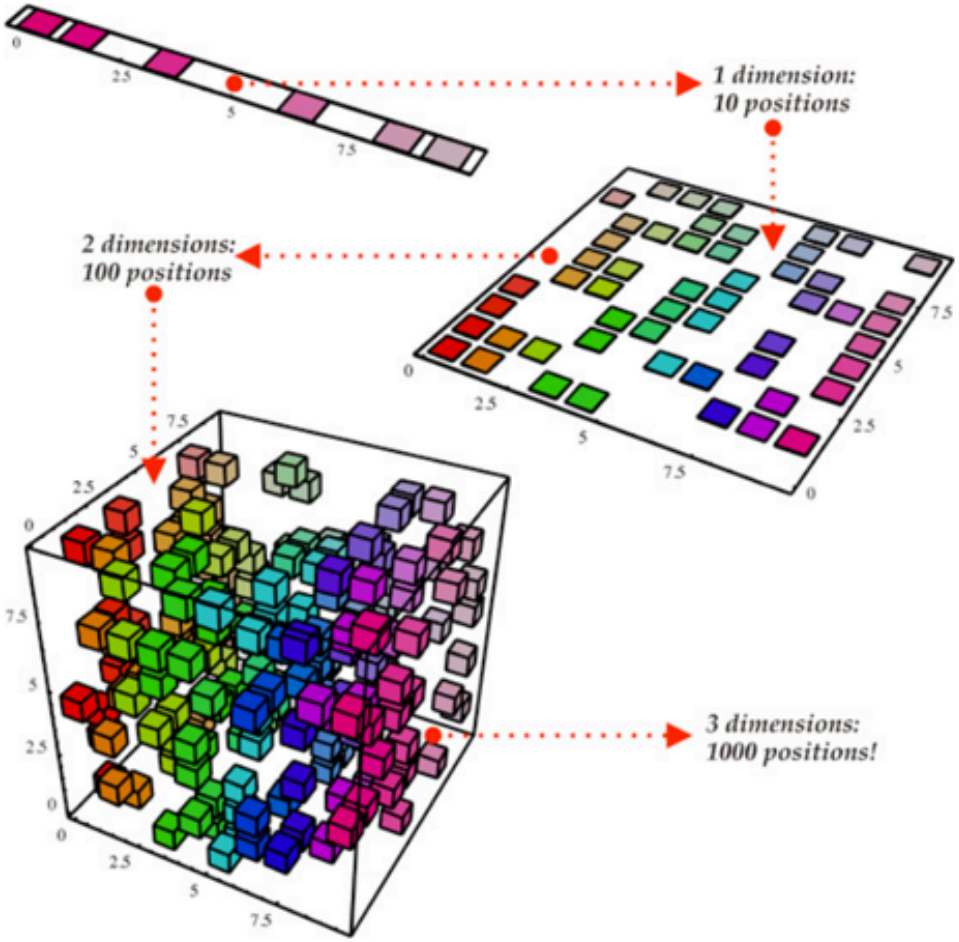
Clustering methods

Dimension(ality) Reduction methods


The data deluge
Brett Ryder - http://www.economist.com/node/15579

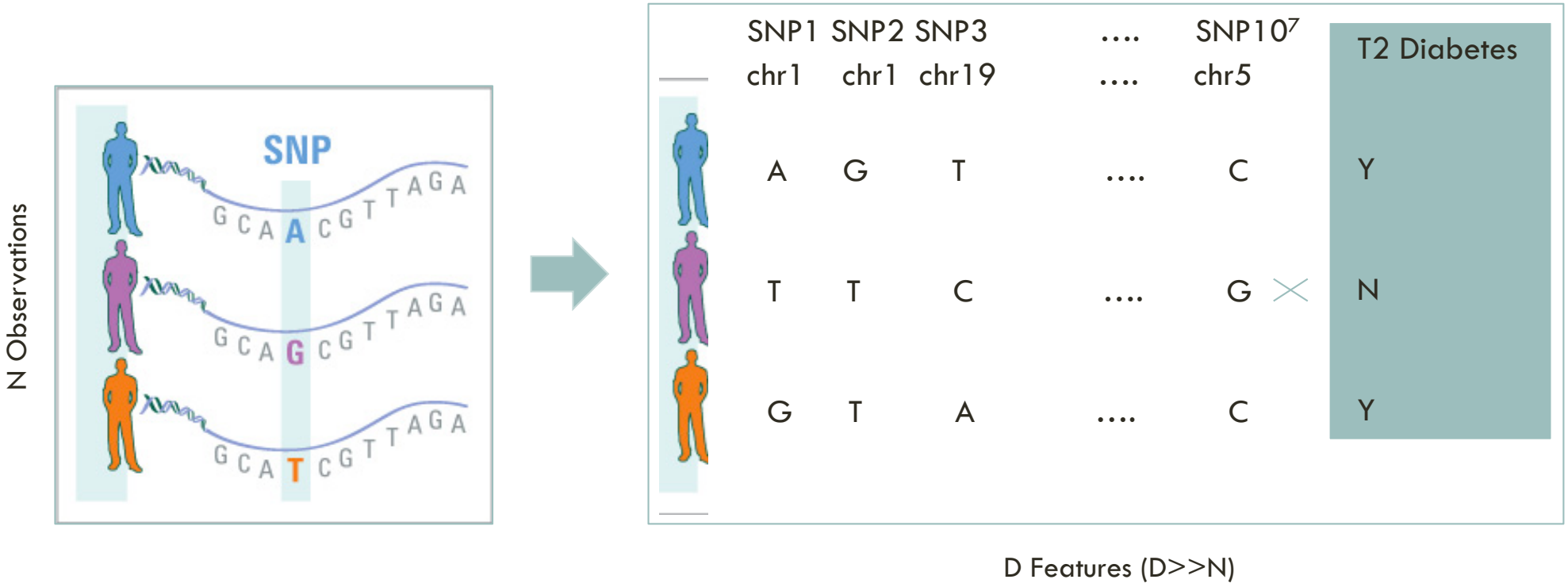
Curse of Dimensionality

Algorithms like clustering are not well adapted to deal with datasets with larger number of **features (dimensions)** than **observations.**

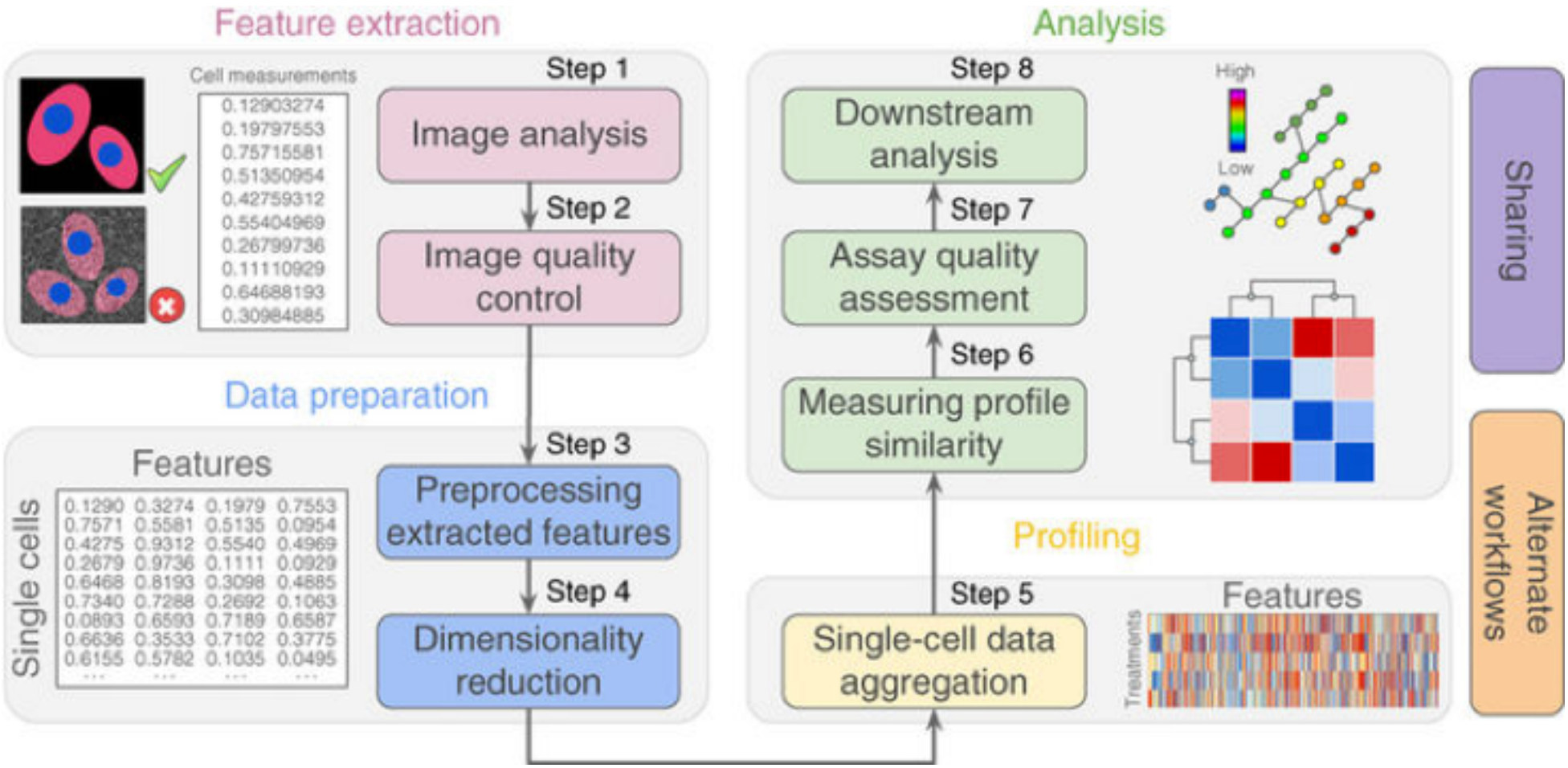


Case in Genome-Wide Association Study (GWAS)

Single Nucleotide Polymorphism variants - disease relationships

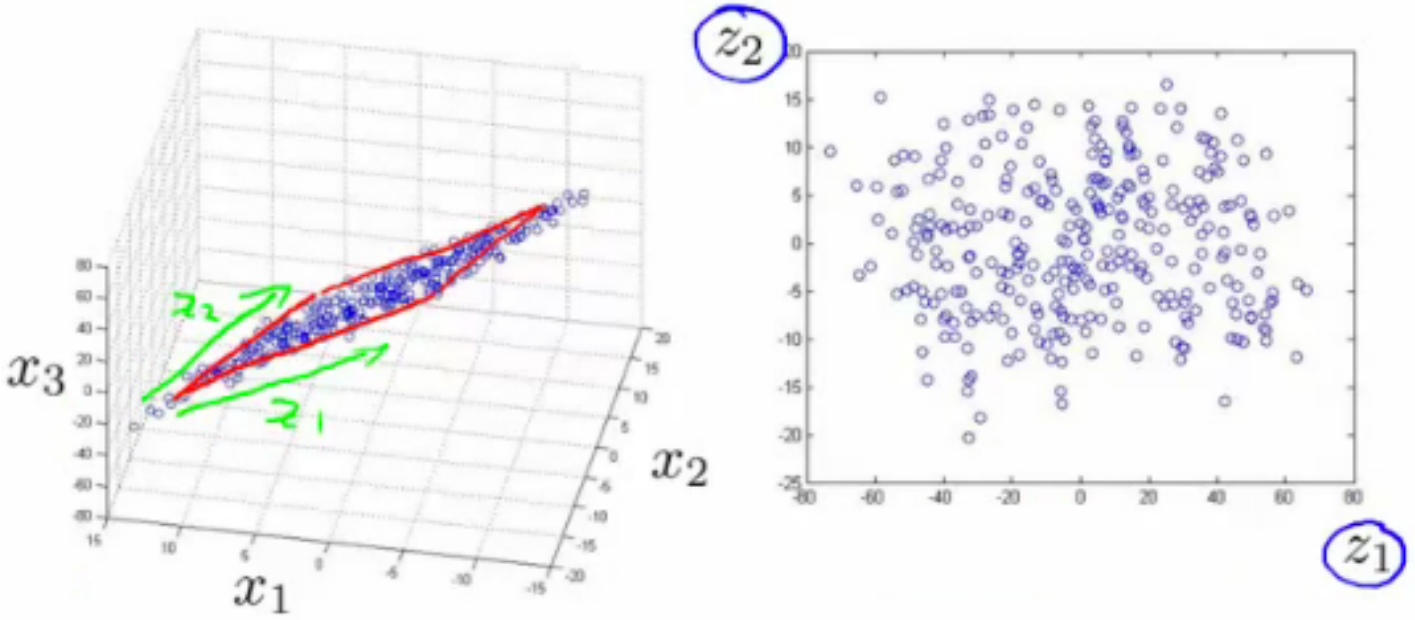


Case in Cell Segmentation / Image Pre-processing



Nature Methods (2017), 14, 849-863

Dimension(ality) Reduction is the process of converting a set of data having vast dimensions into data with lesser dimensions ensuring that it conveys similar information concisely.



D Features ($x1, x2 \ldots x_D$)

d Features ($z1, z2 \ldots z_d$)

D >>> d

Pros and Cons

Simplify the pattern representation and (extrinsic) classification

Less memory consumption

Alleviate **curse of dimensionality** with limited sample data

Loss of information

Increased error

Dealing with Variables / Features

Variable Selection

Pick a subset of the original variables: remove irrelevant or redundant variables

$$X_1\ X_2\ X_3\ X_4\ \ldots\ X_D$$

Goal: minimizing classification error

Variable Extraction

Construct a new set of features $Z_1\ Z_2\ \ldots\ Z_d$

$$Z_i = f(X_1\ X_2\ X_3\ X_4\ \ldots\ X_D)$$

Based on (linear) combinations of original variables
$X_1\ X_2\ X_3\ X_4\ \ldots\ X_D$

Goal: reducing dimension while preserving information

Unsupervised Methods

Variable Selection

Variance Thresholds

Multi-collinearity

Factor Analysis

Variable Extraction

Principal Component Analysis (PCA)

Kernel PCA

Independent Components Analysis (ICA)

Multidimensional Scaling (MDS)

IsoMap

Locally-Linear Embedding (LLE)

Supervised Methods

Variable Selection

Genetic Algorithms

Least Absolute Shrinkage and Selection Operator (LASSO)

Stepwise regressions:
- Forward selection
- Backward selection
- Stepwise selection

Variable Extraction

Linear Discriminant Analysis (LDA)

Quadratic Discriminant Analysis (QDA)

Generalized Discriminant Analysis

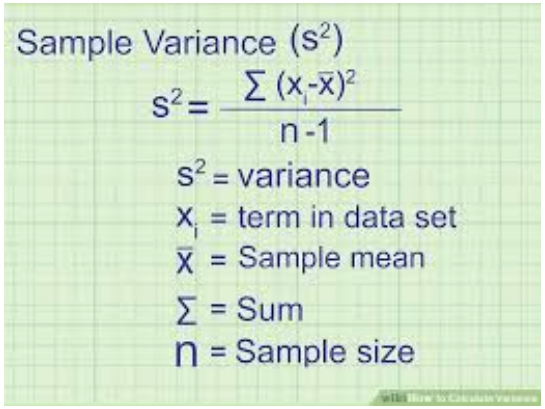
t-Distributed Stochastic Neighbor Embedding (t-SNE)

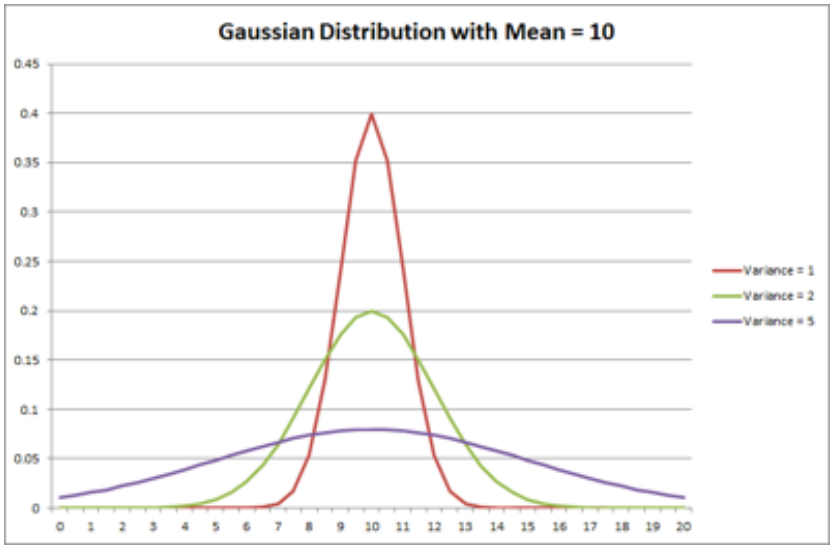
Autoencoders

Variance thresholds

What is the variance threshold?

Remove features whose values don't change much from observation to observation (i.e. their **variance** falls below a threshold). These features provide little value.





Age	Gender
32	F
35	M
34	F
32	F
35	F
26	F
21	F
35	F

Pros and Cons

Based on solid intuition: **features that don't change much also don't add much information**

Safe way to reduce dimensionality at the start of modeling process

Rarely sufficient

Manual tuning of variance threshold could be tricky: **recommended lower value**

Scikit-Learn Function .VarianceThreshold( )

sklearn.feature_selection.VarianceThreshold

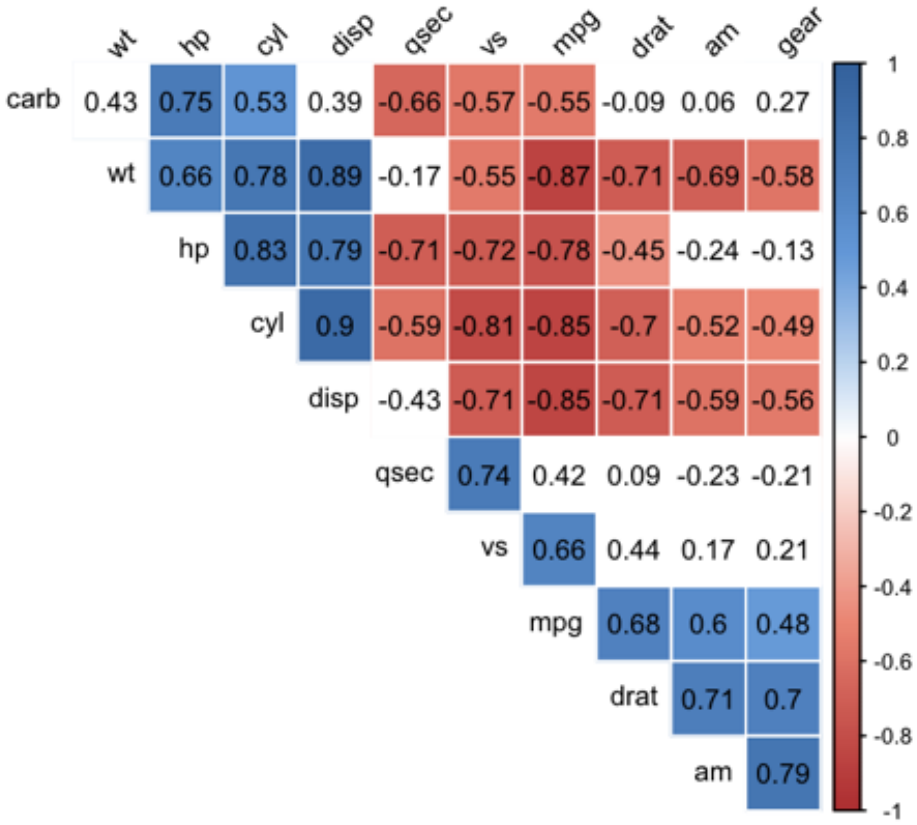
class `sklearn.feature_selection.` **`VarianceThreshold`** (*threshold=0.0*)

Parameters:	**threshold : *float, optional*** Features with a training-set variance lower than this threshold will be removed. The default is to keep all features with non-zero variance, i.e. remove the features that have the same value in all samples.
Attributes:	**variances_ : *array, shape (n_features,)*** Variances of individual features.

The following dataset has integer features, two of which are the same in every sample. These are removed with the default setting for threshold:

```
>>> X = [[0, 2, 0, 3], [0, 1, 4, 3], [0, 1, 1, 3]]
>>> selector = VarianceThreshold()
>>> selector.fit_transform(X)
array([[2, 0],
       [1, 4],
       [1, 1]])
```

Multi-collinearity



What is it?

Remove features that are **highly correlated** with others (i.e. its values change very similarly to another's). These features provide **redundant information**.

How?

- Calculate all pair-wise correlations.
- If the correlation between a pair of features is above a given threshold e.g. 0.95, remove the one that has larger mean absolute correlation with other features.

Correlation Ranks / Coefficients

Parametric

Pearson product-moment r

Linear relationship between continuous variables

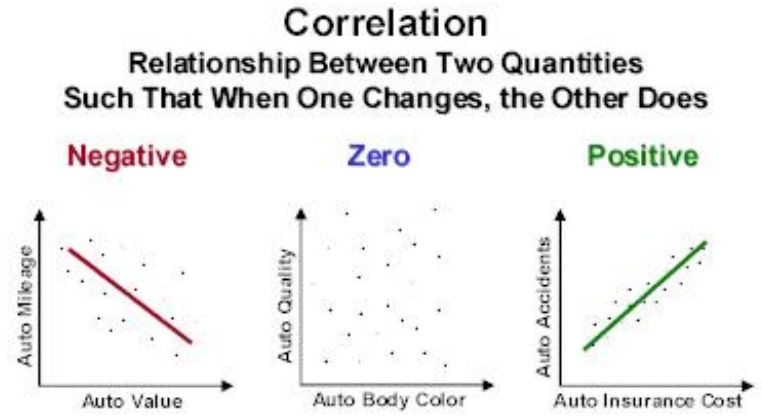
Non-parametric:

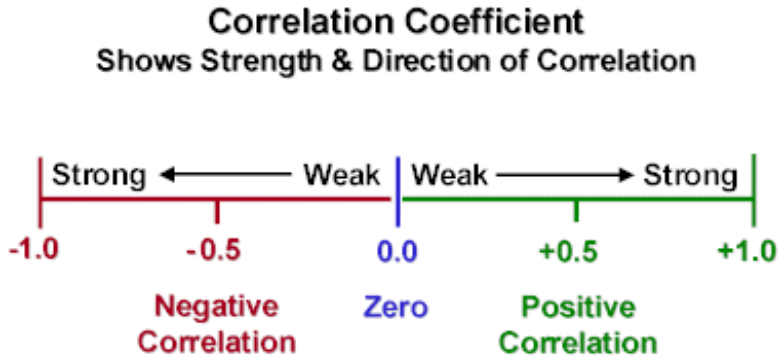
Kendall's tau rank correlation

Linear or not relationship between ordinal variables

Spearman's order-rank correlation rho

Linear or not relationship between continuous and discrete variables





Pros and Cons

Based on solid intuition: **similar features provide redundant information.**

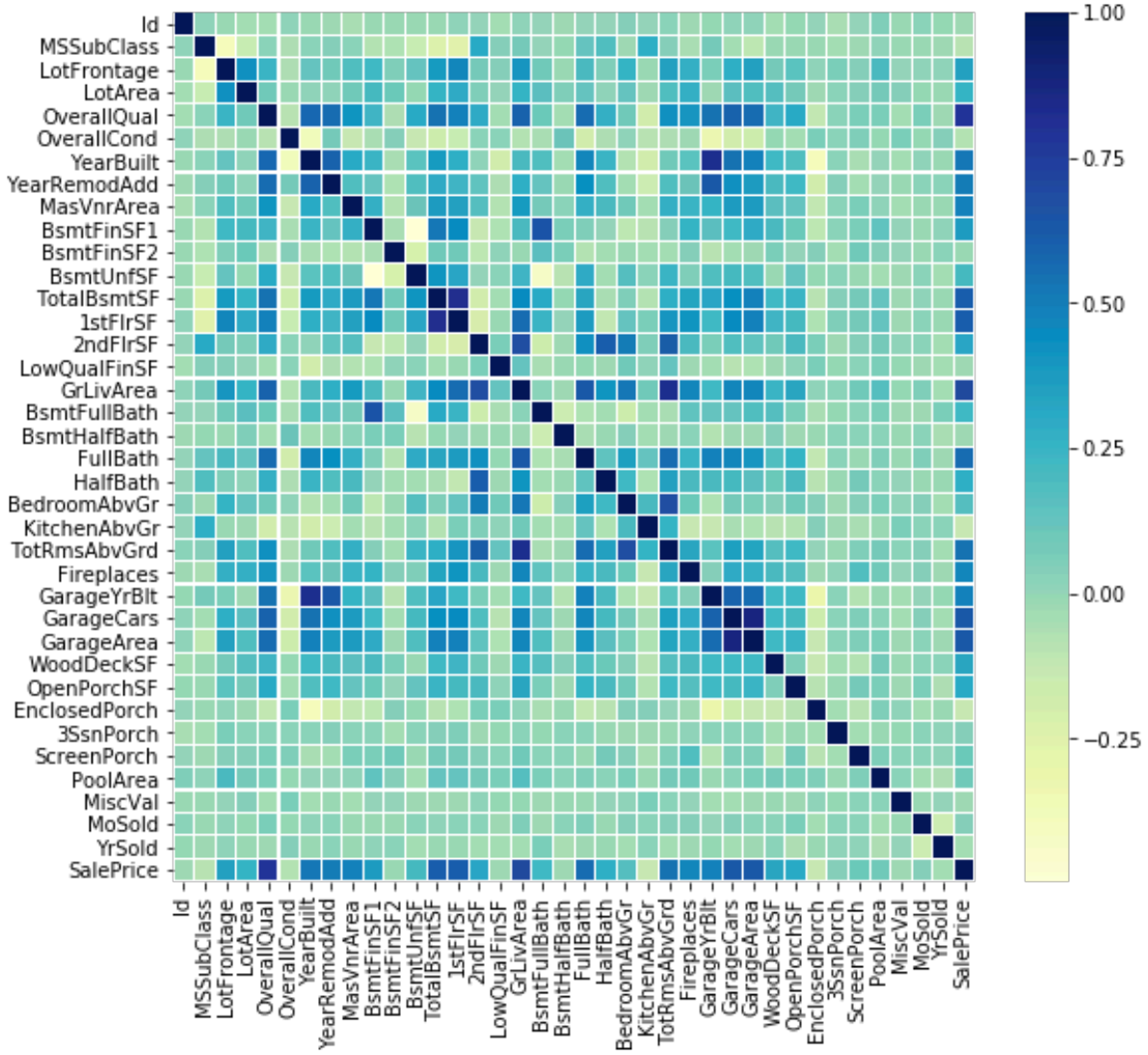
Some algorithms are not robust to correlated features, so removing them can boost performance.

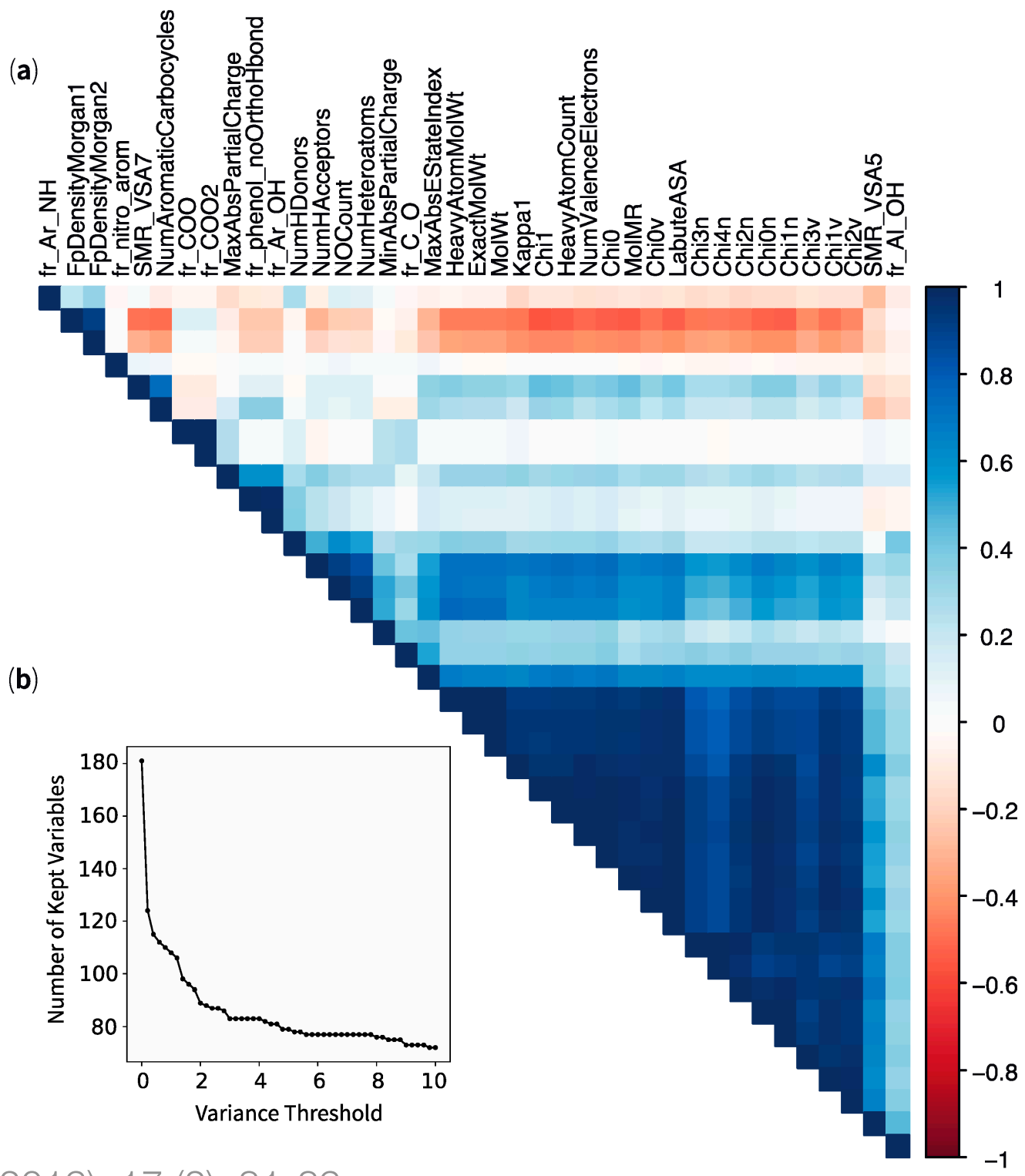
Manual tuning of correlation threshold could be tricky: **recommended algorithms with build-in feature selection.**

Principal Component Analysis preferred**.**

Pandas Function .corr( )

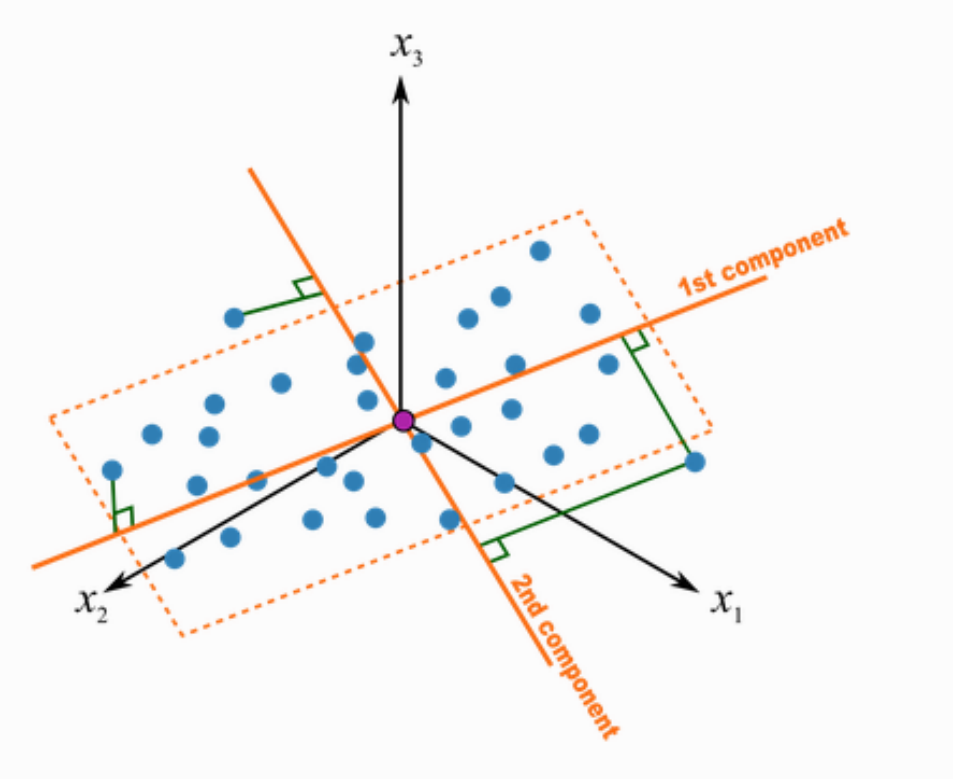
1. Import libraries (numpy, pandas, seaborn, …)
2. Get data frame D
3. Call D.corr( ) ‘matrix’
4. Display matrix as heatmap using seaborn





Plisson et al. Marine Drugs (2019), 17 (2), 81-89

Principal Components Analysis



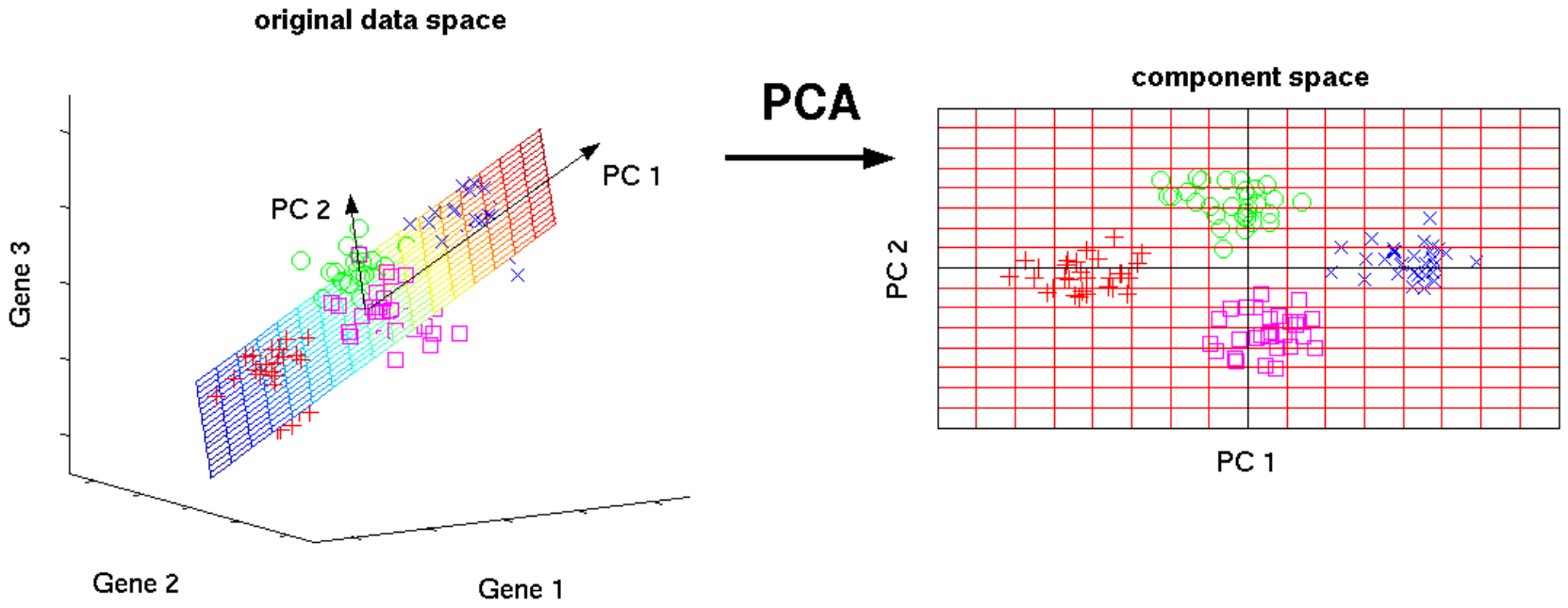
Definition

PCA creates linear combinations of the original features (dimensions). Identify the directions that maximize the variance of this subspace. These are called **principal components** (they are orthogonal = not correlated).

The first principal component or PC1 explained the most variance (**eigenvalue**) in the dataset, PC2 explained the second-most variance …

PCA maximizes the variance = search for the directions (main components) / **loadings** that contain the greatest dispersion of data (variability, greater amount of information) using fewer features

Linear Principal Components



Loadings

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + \ldots + \phi_{p1}X_p \qquad (1)$$

$$Z_2 = \phi_{12}X_1 + \phi_{22}X_2 + \ldots + \phi_{p2}X_p \qquad (2)$$

Pros and Cons

Fast and simple to implement to boost performance.

PCA offers **several variations and extensions** (i.e. kernel PCA, sparse PCA etc.) to tackle specific roadblocks.

Should normalise your data prior extracting feature(s).

The new principal components are hard to interpret.

Manual tuning of a threshold for cumulative explained variance.

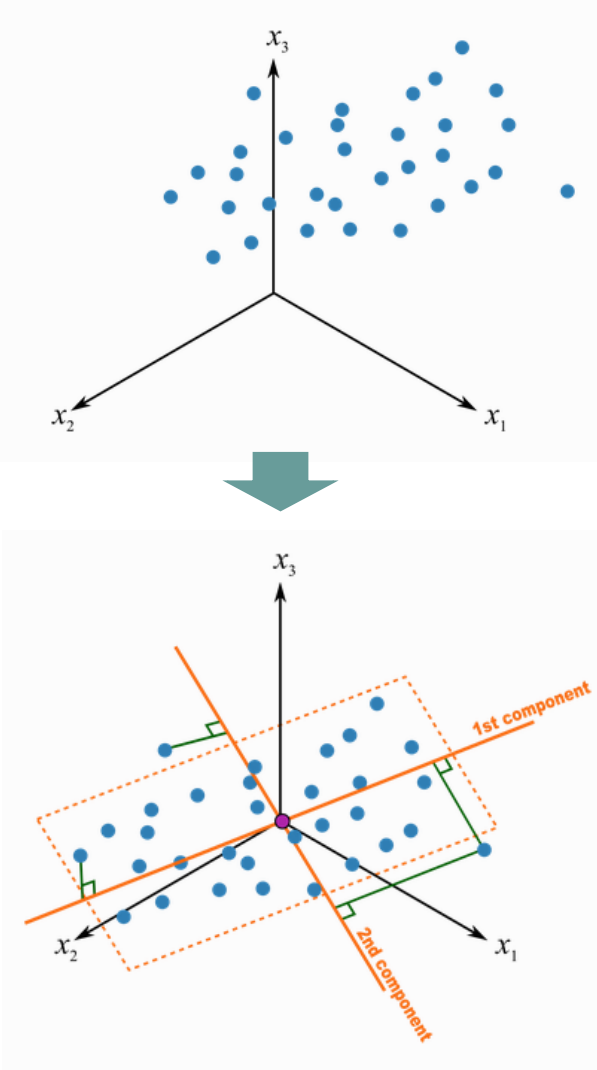
Reminder: Feature Extraction

Construct a new set of features Z_1 Z_2 … Z_d

$$Z_i = f(X_1\ X_2\ X_3\ X_4\ \ldots\ X_D)$$

Based on (linear) combinations of original features X_1 X_2 X_3 X_4 … X_D

Reducing dimensionality while preserving information



Manual PCA Computation with Scikit-learn

1. Standardize dataset `from sklearn.preprocessing import StandardScaler`

	sepal length	sepal width	petal length	petal width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

Standardization →

	sepal length	sepal width	petal length	petal width
0	-0.900681	1.032057	-1.341272	-1.312977
1	-1.143017	-0.124958	-1.341272	-1.312977
2	-1.385353	0.337848	-1.398138	-1.312977
3	-1.506521	0.106445	-1.284407	-1.312977
4	-1.021849	1.263460	-1.341272	-1.312977

The array **x** (visualized by a pandas dataframe) before and after standardization

2. Calculate covariance matrix to get eigen values (~% variance) and eigenvectors / loadings (directions) then calculate PCs scores.

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)

principalComponents = pca.fit_transform(x)
```

PCA
(2 components) →

	principal component 1	princial component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

Manual PCA Computation with Scikit-learn

3. Concatenate new data frame with labels

	principal component 1	principal component 2
0	-2.264542	0.505704
1	-2.086426	-0.655405
2	-2.367950	-0.318477
3	-2.304197	-0.575368
4	-2.388777	0.674767

principalDf

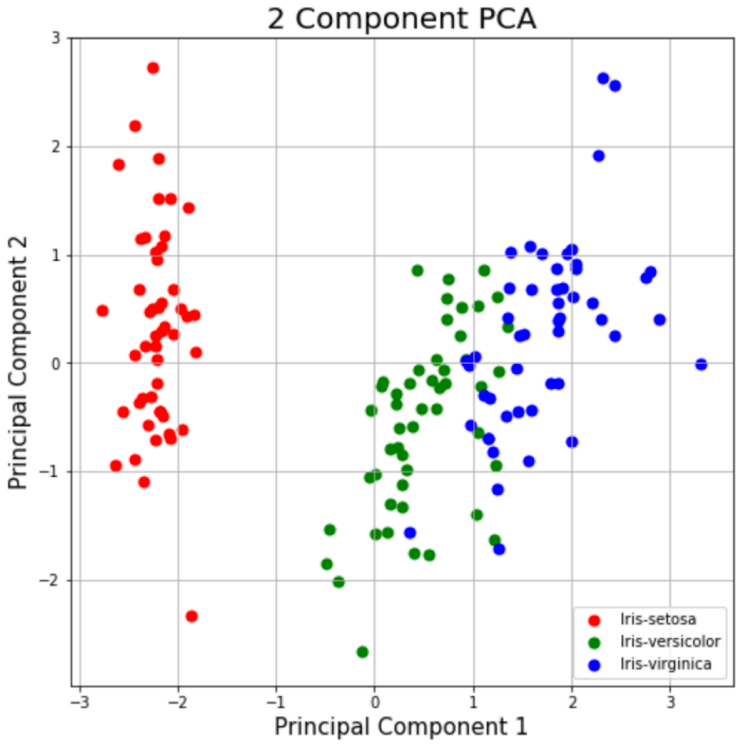
	target
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa

df[['target']]

pd.concat(axis = 1)

	principal component 1	principal component 2	target
0	-2.264542	0.505704	Iris-setosa
1	-2.086426	-0.655405	Iris-setosa
2	-2.367950	-0.318477	Iris-setosa
3	-2.304197	-0.575368	Iris-setosa
4	-2.388777	0.674767	Iris-setosa

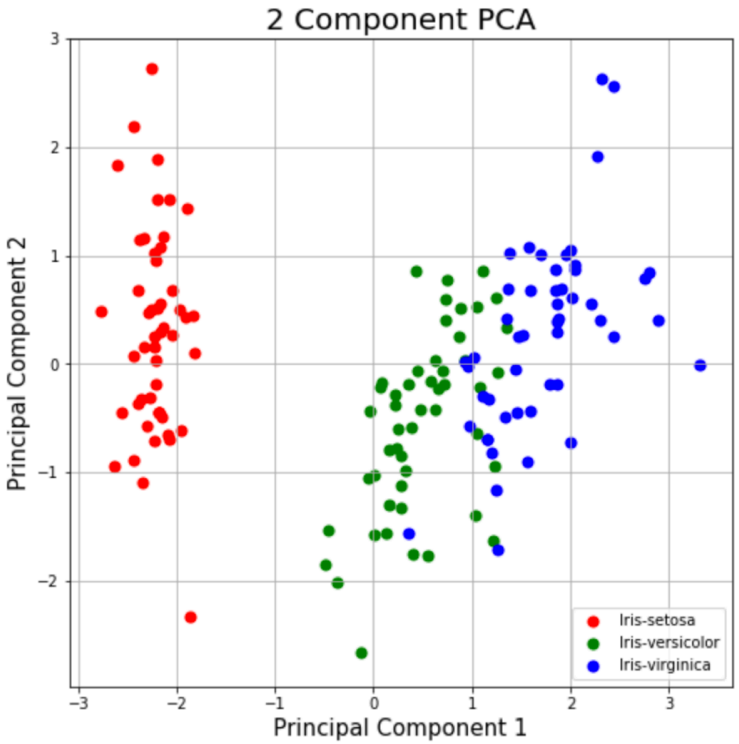
finalDf



Proportion of Variance Explained

By performing some algebra, the proportion of variance explained (PVE) by the mth principal component is calculated using the equation:

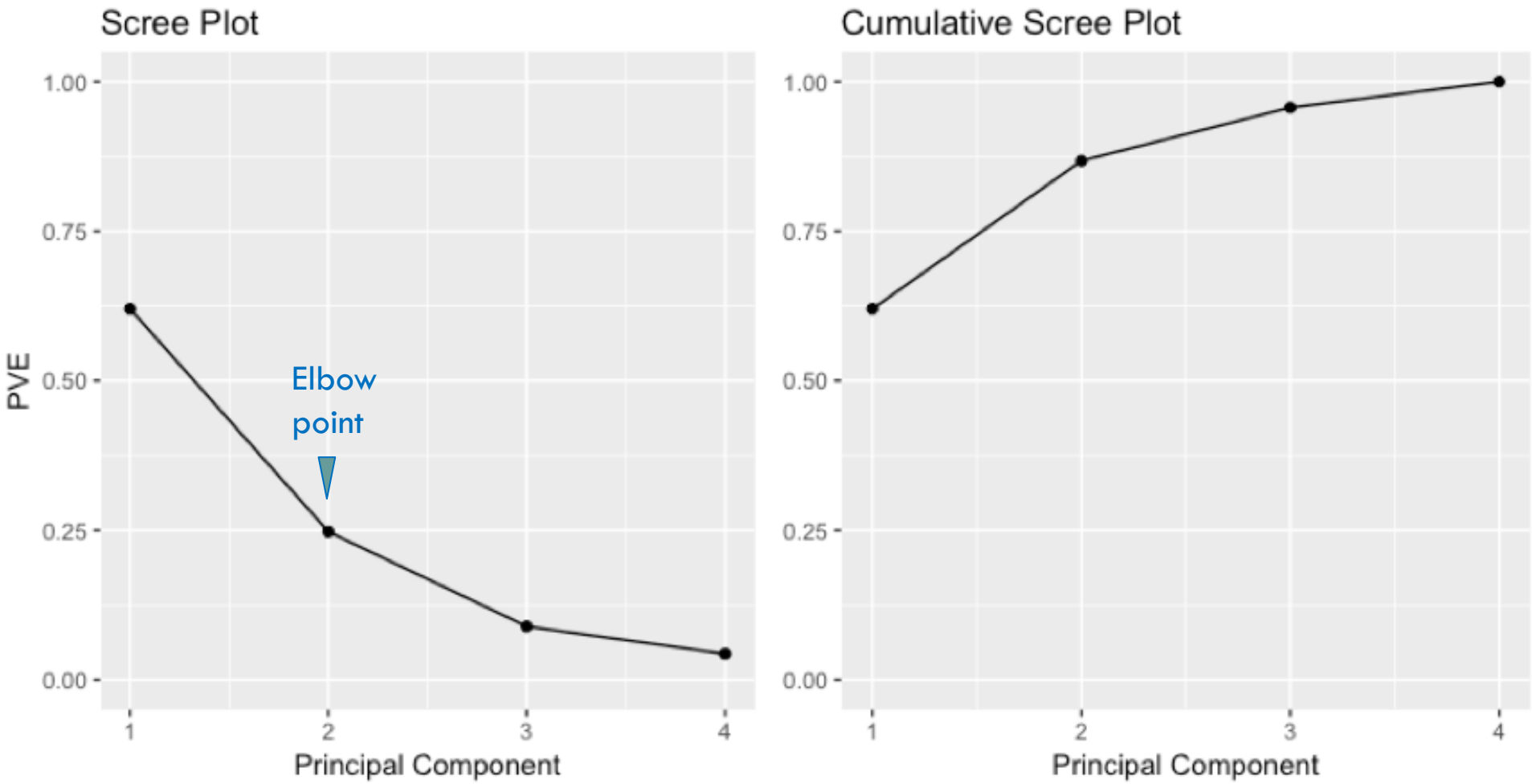
$$PVE = \frac{\sum_{i=1}^{n}(\sum_{j=1}^{p}\phi_{jm}x_{ij})^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2} \quad (3)$$




```
pca.explained_variance_ratio_
```

PC1 72.77%
PC2 23.03%

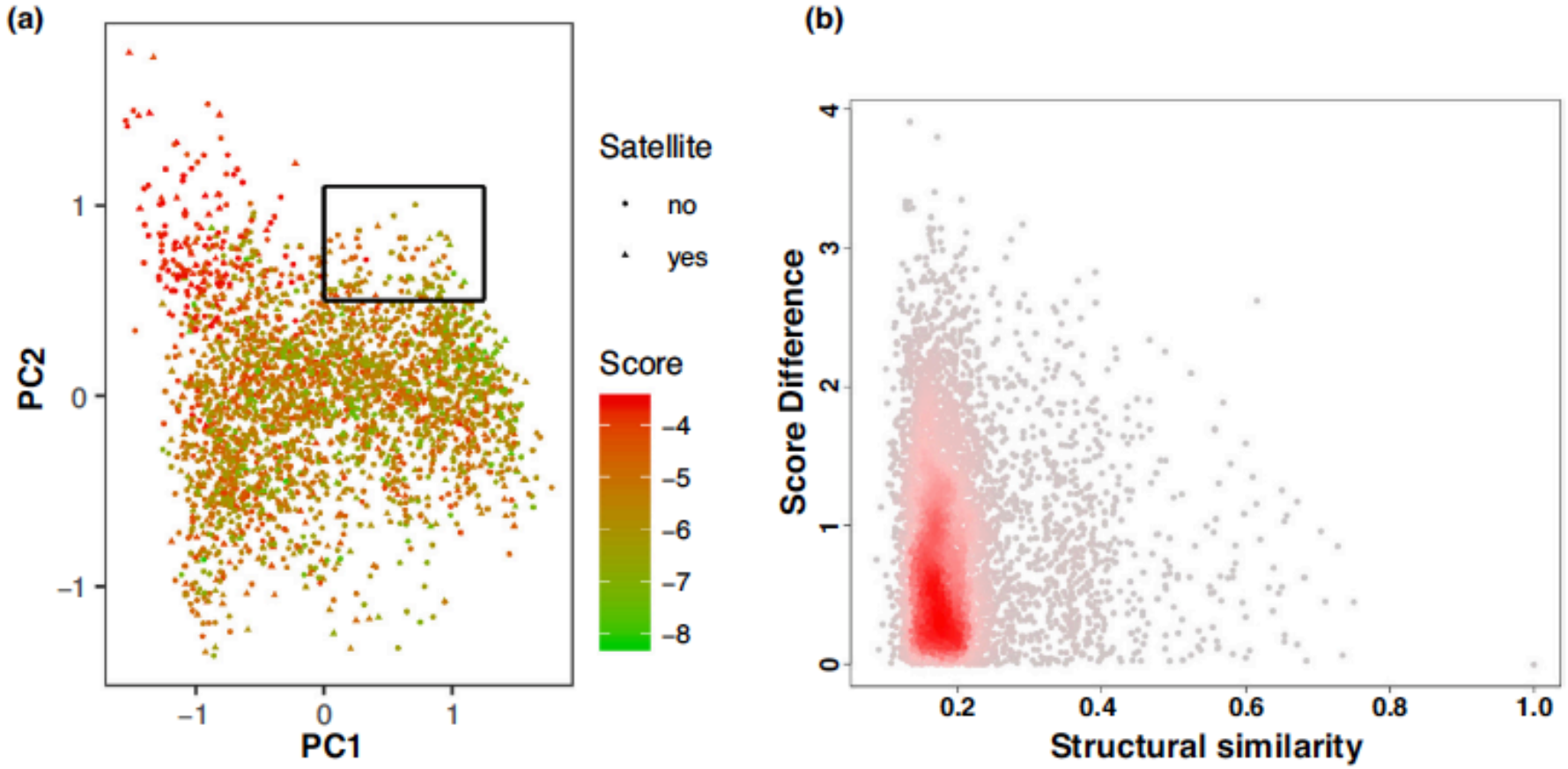
Together 95.80% PVE


Scree Plot
PVE
1.00
0.75
0.50
0.25
0.00
Elbow
point
1
2
3
4
Principal Component
Cumulative Scree Plot
1.00
0.75
0.50
0.25
0.00
1
2
3
4
Principal Component

Visualising results from virtual screening

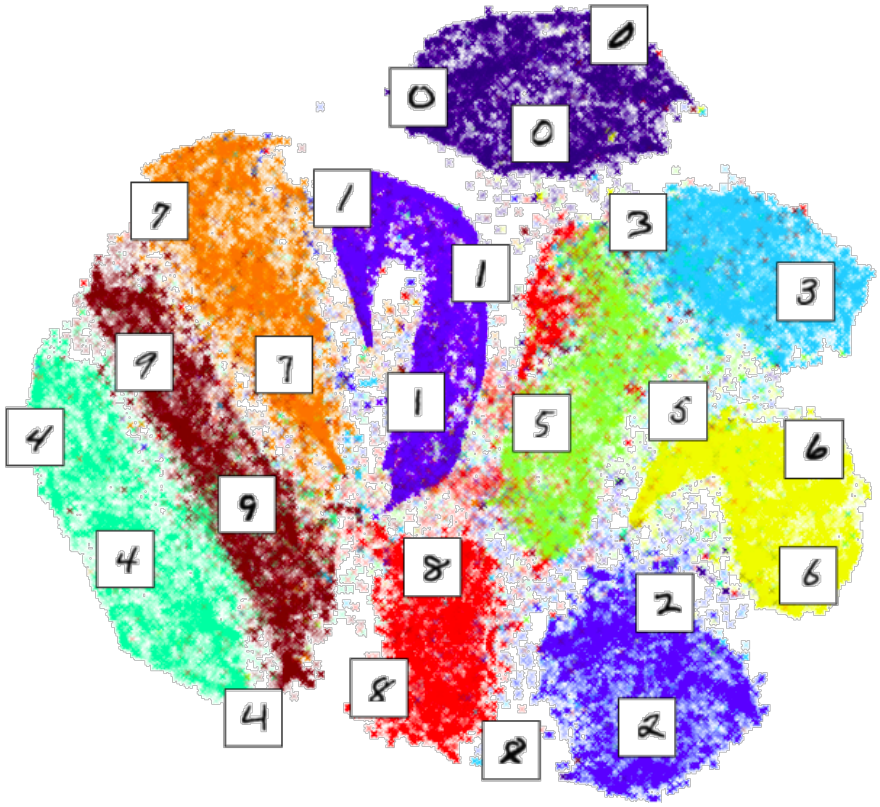
(a) 2789 virtual molecules - DNMT inhibitors

(b) An adapted density structure-activity similarity (SAS) map



Medina-Franco et al. Drug Discovery (2019) - Hot off the press

t-Distributed Stochastic
Neighbor Embedding (t-SNE)



Definition

t-SNE is a **nonlinear dimensionality reduction** technique for embedding high-dimensional data for visualisation in a low-dimensional space of two or three dimensions.

1. t-SNE constructs a probability distribution over pairs of high-dimensional observations in such a way that **similar objects have a high probability of being picked** while dissimilar points have an extremely small probability of being picked.

2. t-SNE defines a similar probability distribution over the points in the low-dimensional map, and it minimizes the sum of **Kullback–Leibler divergence** of overall data points using a gradient descent method.

Note: Kullback-Leibler (KL) divergence is a measure of how one probability distribution diverges from a second probability distribution.

Applications

MNIST dataset (in 2D)
MNIST dataset (in 3D)
Olivetti faces dataset (in 2D)
COIL-20 dataset (in 2D)
Netflix dataset (in 3D) on Russ's RBM features
Words dataset (in 3D) on Andriy's semantic features
20 Newsgroups dataset (in 2D) on Simon's discLDA features
Reuters dataset (in 2D) landmark t-SNE using semantic hashing
NIPS dataset (in 2D) on co-authorship data (1988-2003)
NORB dataset (in 2D) by Vinod
Words (in 2D) by Joseph on features learned by Ronan and Jason
CalTech-101 on SIFT bag-of-words features
S&P 500 by Steve on information about daily returns on company stock
Interactive map of scientific journals on data by Nees-Jan and Ludo, using VOSviewer
Relation between World Economic Forum councils
ImageNet by Andrej on Caffe convolutional net features
Multiple maps visualizations
Allen Brain data

https://lvdmaaten.github.io/tsne

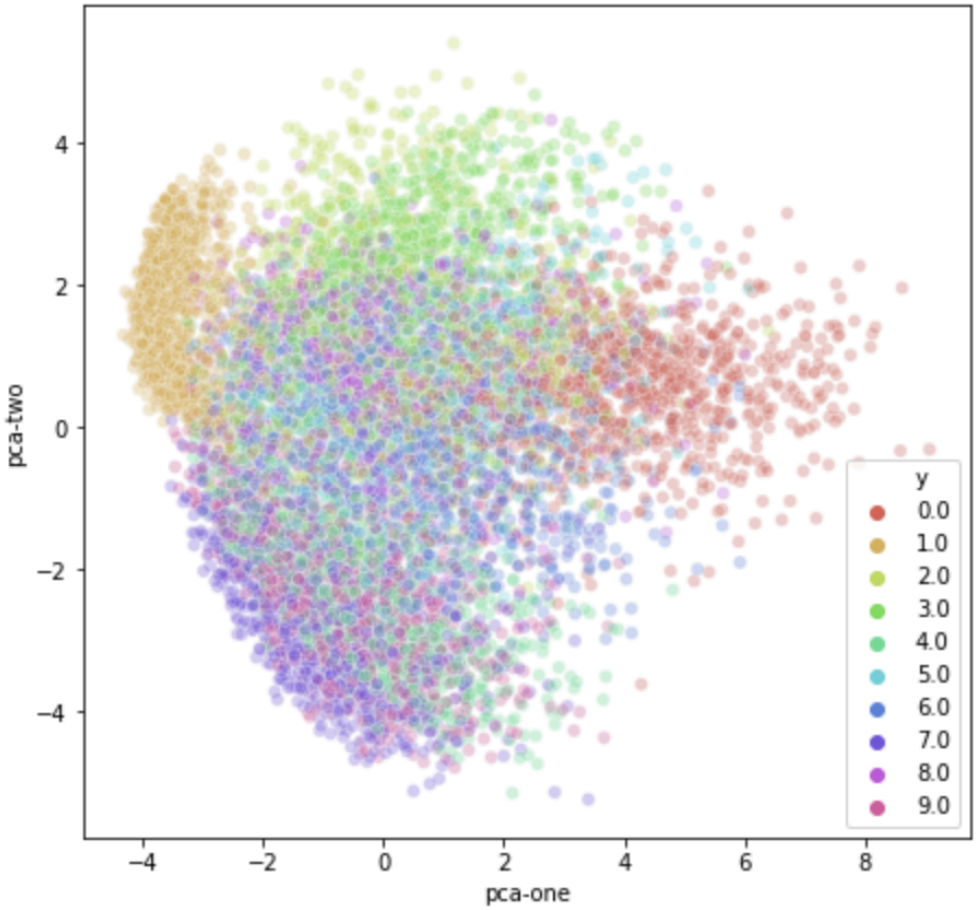
Pros and Cons

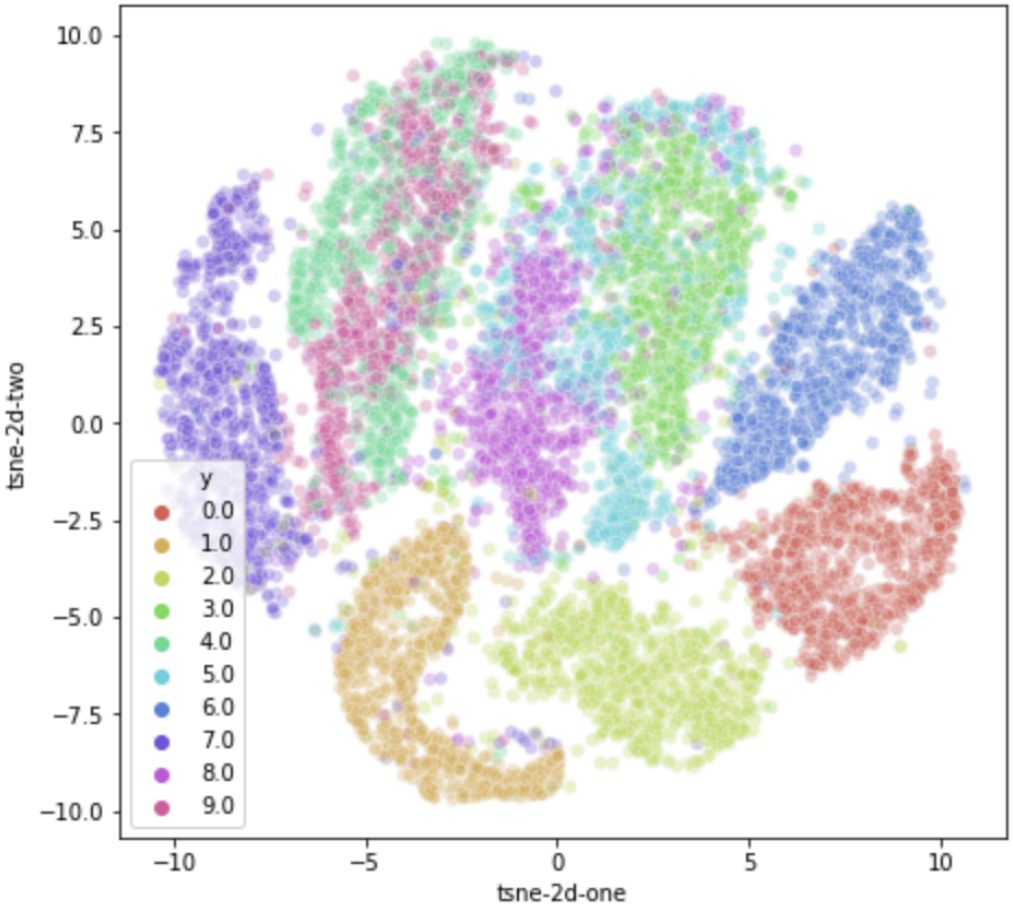
t-SNE maps the multi-dimensional data **to a lower dimensional space and attempts to find patterns** in the data by identifying observed clusters based on similarity of data points with multiple features

However, after this process, the input features are no longer identifiable, and **you cannot make any inference based only on the output of t-SNE.**

It is mainly a data exploration and visualization technique.

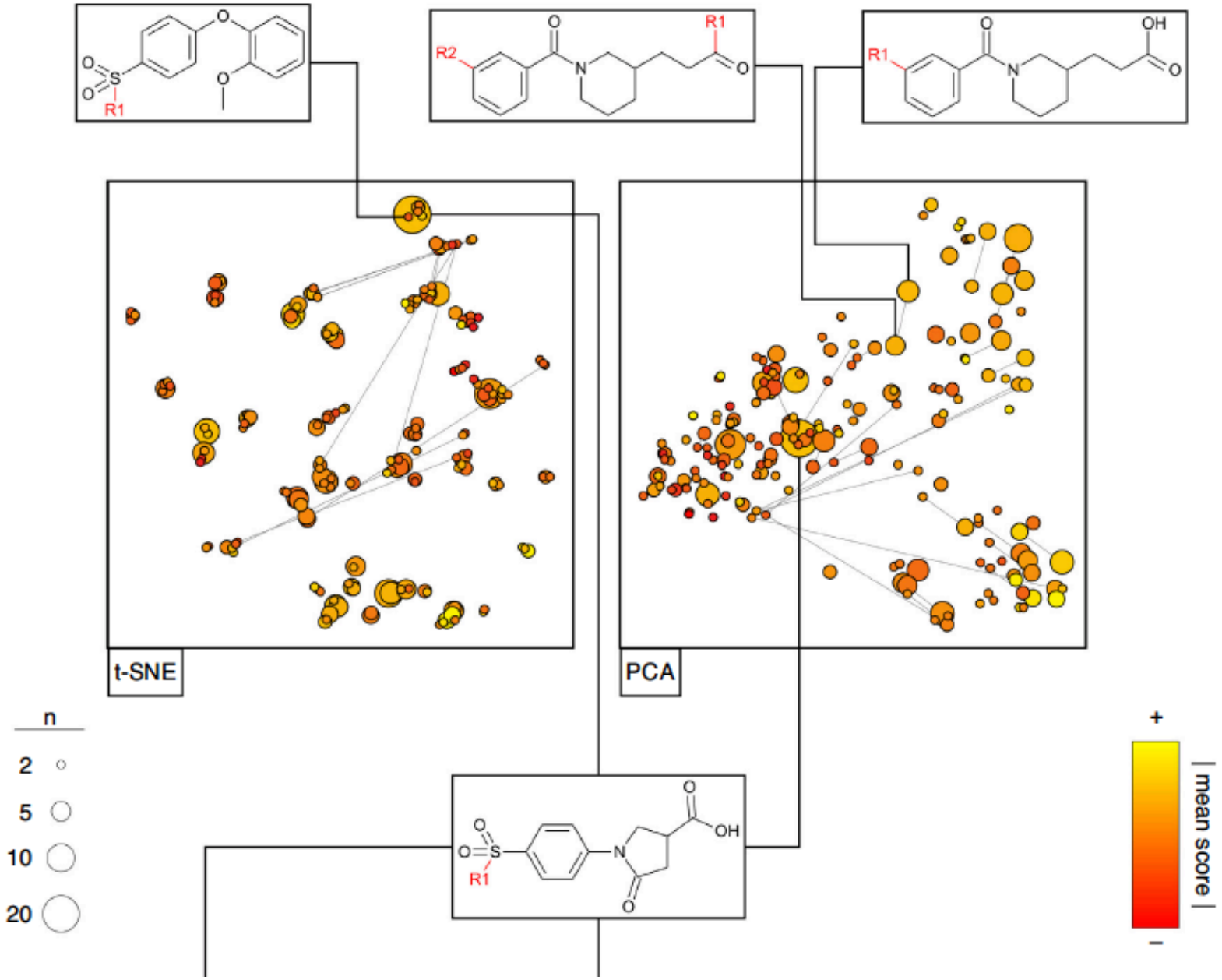
MNIST data on dimensionality reduction methods





Mapping Structure-Activity Data (Chemical Space)

Constellation plots showing the 188 cores found for a data set of 472 molecules in 153 analog series - DNMT inhibitors



Medina-Franco et al. Drug Discovery (2019) - Hot off the press

References

1. Clustering algorithms Data Science for Business
Foster Provost & Tom Fawcett (2013) O'REILLY

2. Chapter 17 – Distances and Similarities in Data Analysis
Dictionary of distances (ISBN: 978-0-444-52087) by E. Deza & M.-M. Deza

3. Applied Predictive Modeling (2013) Kuhn, Max, Johnson, Kjell

4. David Sheehan https://dashee87.github.io

5. Laurens van der Maaten https://lvdmaaten.github.io/tsne

https://github.com/BarbaraDiazE/CABANA_CHEMOINFORMATICS