IOT607U Data Mining

Week 4: Data preprocessing

Dr Lin Wang

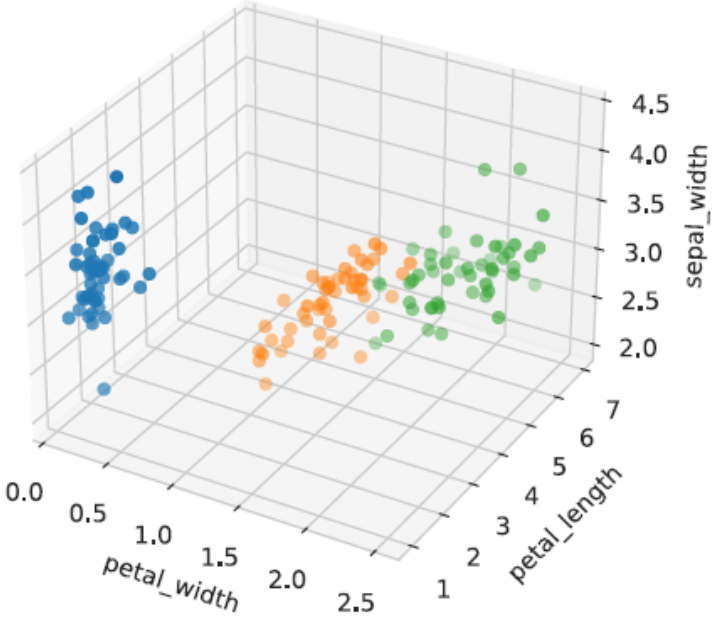
School of EECS, Queen Mary University of London

Last week

1. Data Exploration

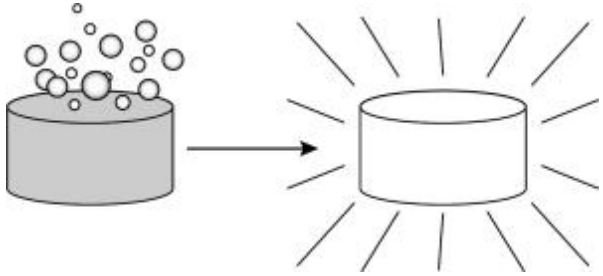
2. Data Summarization

3. Data Visualization



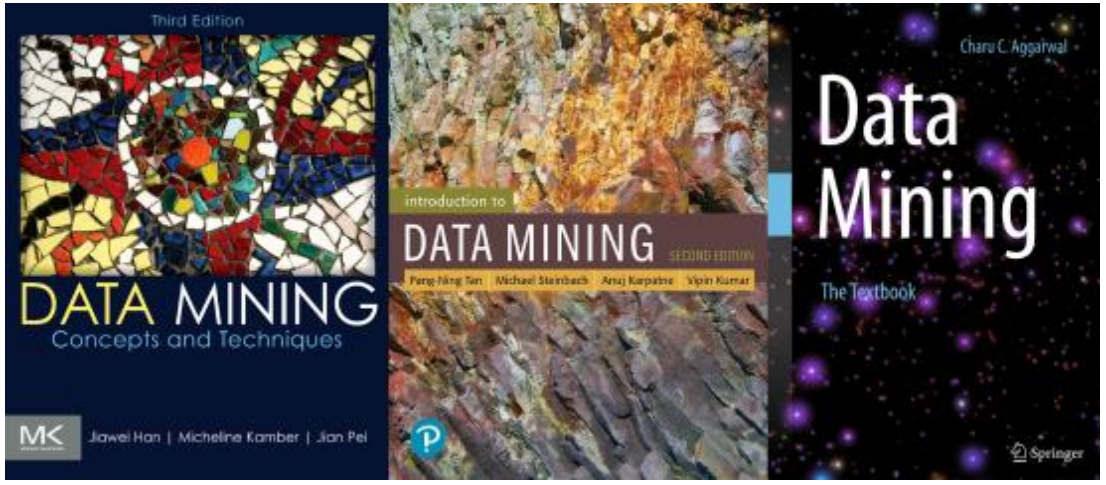
This week's contents

1. Data Preprocessing: An Overview
2. Data Cleaning
3. Data Normalziation
4. Dimensionality Reduction



Reading

- **Chapter 3 of J. Han, M. Kamber, J. Pei, "Data Mining: Concepts and Techniques", 3rd edition, Elsevier/Morgan Kaufmann, 2012**
- Section 2.3 of P.-N. Tan, M. Steinbach, A. Karpatne, V. Kumar, "Introduction to Data Mining", 2nd edition, Pearson, 2019
- Chapter 2 of C. C. Aggarwal, "Data Mining: The Textbook", Springer, 2015



Data Preprocessing: An Overview

Why Preprocess the Data?

Data have quality if they satisfy the requirements of the intended use. There are many factors comprising data quality:

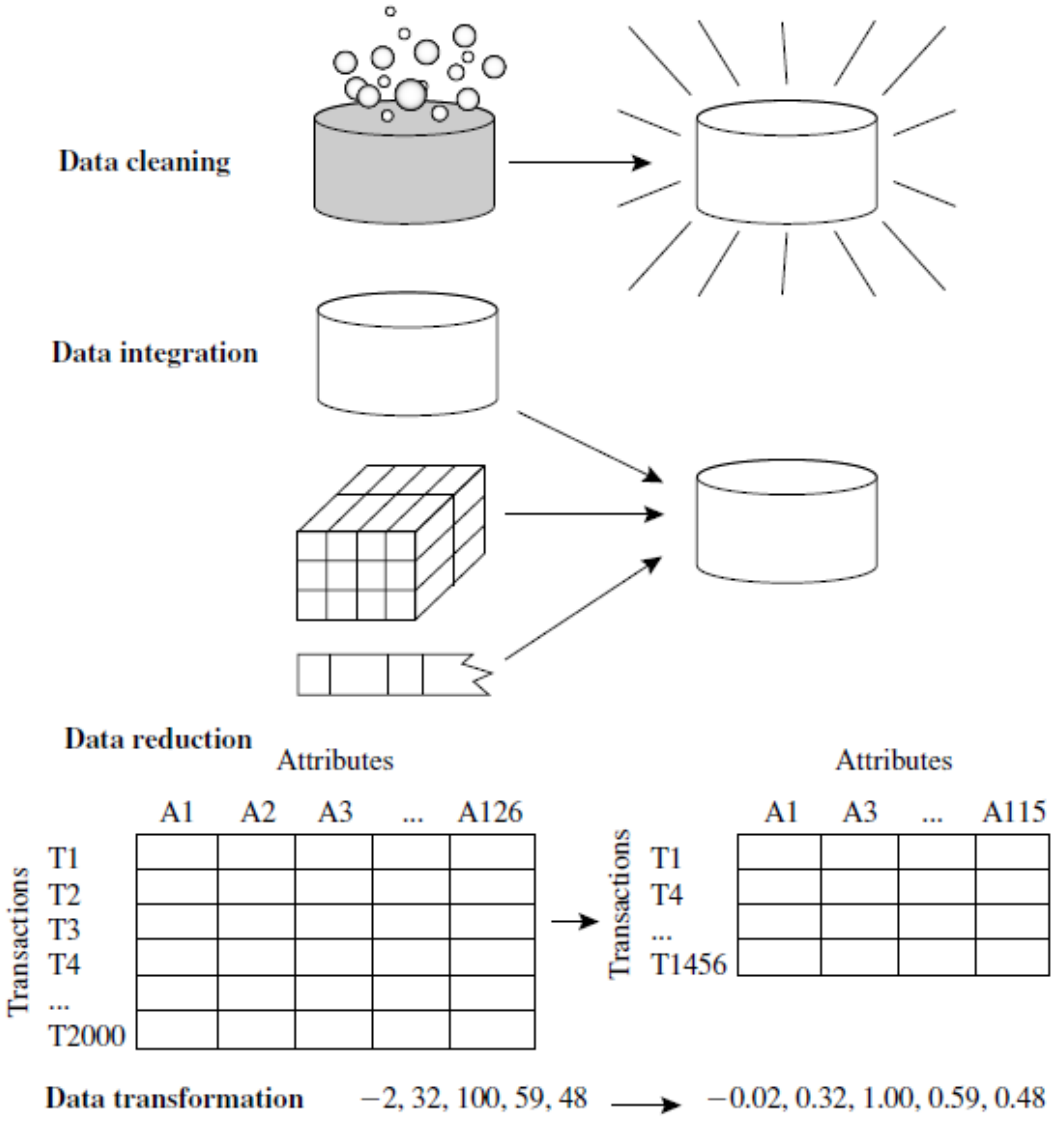
- Accuracy: incorrect attribute values
- Completeness: not recorded, unavailable
- Consistency: data not consistent with other recorded data

- Timeliness: data not updated in a timely fashion
- Believability: how much the data is trusted by users
- Interpretability: how easily the data can be understood

What is Data Preprocessing? — Major Tasks

- **Data cleaning**: handle missing data, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- **Data integration**: Integration of multiple databases
- **Data reduction**:
  - Subset selection
  - Dimensionality reduction
- **Data transformation**:
  - Normalization

What is Data Preprocessing? — Major Tasks



Forms of data preprocessing.

Data Cleaning

DATA CLEANING

> *80 percent of a data scientist's valuable time is spent simply finding, cleansing, and organizing data, leaving only 20 percent to actually perform analysis...*
>
> IBM Data Analytics

- Filling in missing values
- Smoothing noisy data
- Removing outlier
- Correcting inconsistencies

DATA CLEANING

Data in the real world is incomplete, noisy, and inconsistent.

- Incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
  e.g. *Occupation* = " " (missing data)
- Noisy: containing noise, errors, or outliers
  e.g. *Salary* = "*-10*" (an error)
- Inconsistent: containing discrepancies in records
  e.g. *Age* = "*42*", *Birthday* = "*03/07/2010*"
  e.g. Rating was "*1, 2, 3*", now rating is "*A, B, C*"
- Intentional (disguised missing data)
  e.g. *Jan. 1 as everyone's birthday*

MISSING DATA

Data is not always available, and many entries might have no recorded value for several attributes.

SOURCES OF MISSING DATA

- User forgot to fill in a field.
- Data was lost while transferring manually from a legacy database.
- There was a programming error.
- Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.

HANDLING MISSING DATA

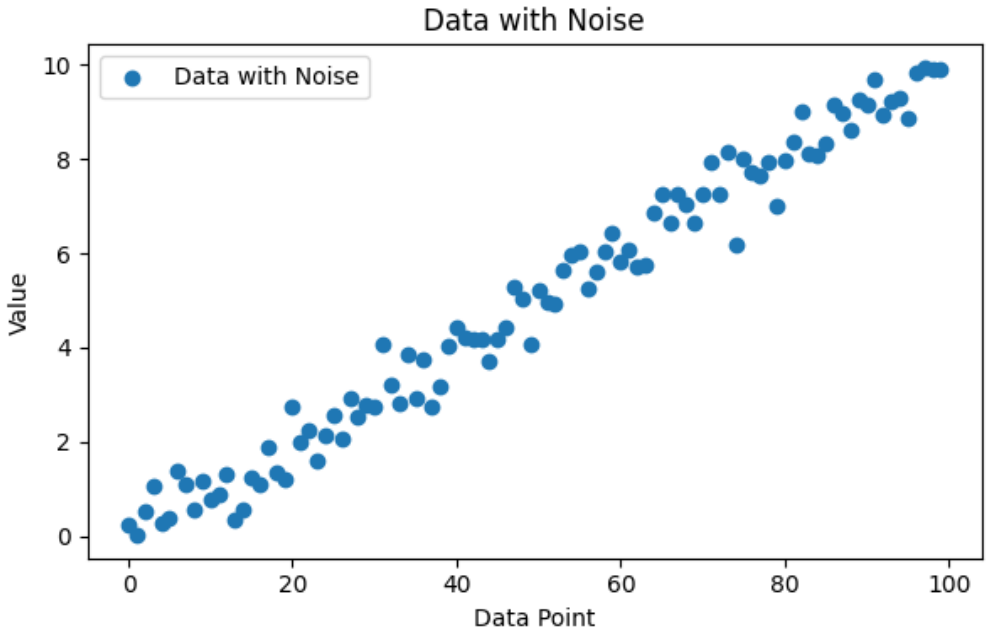
- **Ignore:** not effective when the % of missing values per attribute varies considerably
- **Fill in the missing value manually**: tedious + infeasible
- **Fill in the missing value automatically** with:
    - A global constant
    - The attribute mean/median
    - The most probable value: inference using interpolation/regression

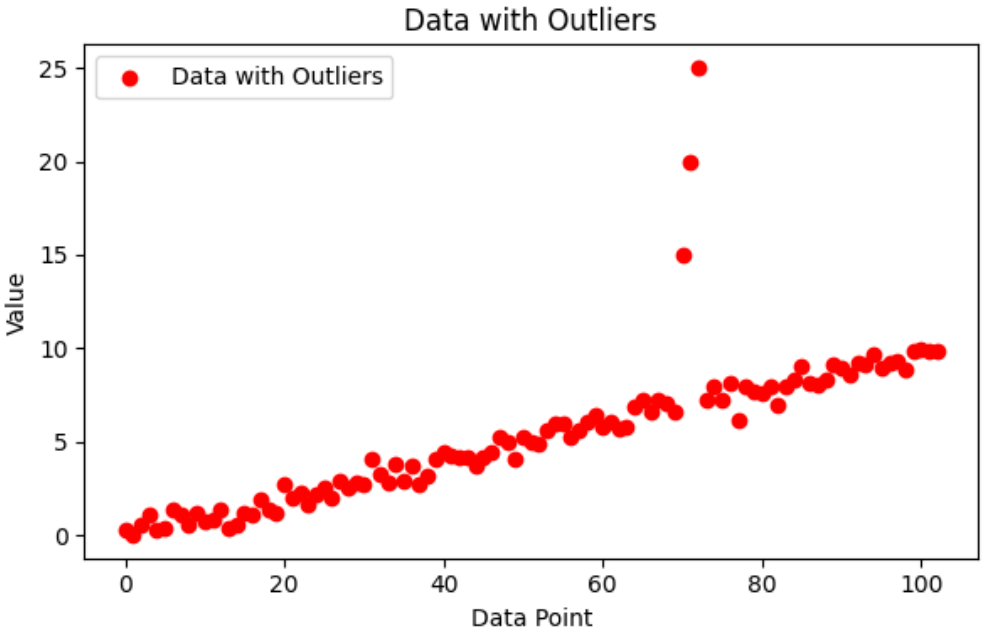
- Useful Python functions
    - isnull(): detecting missing/null values
    - dropna(): removing rows/columns containing null values
    - fillna(): filling null values using a specified method

Noisy Data

Noise: random error or variance in a measured variable

Outlier: data points that significantly differ from the rest of the data set



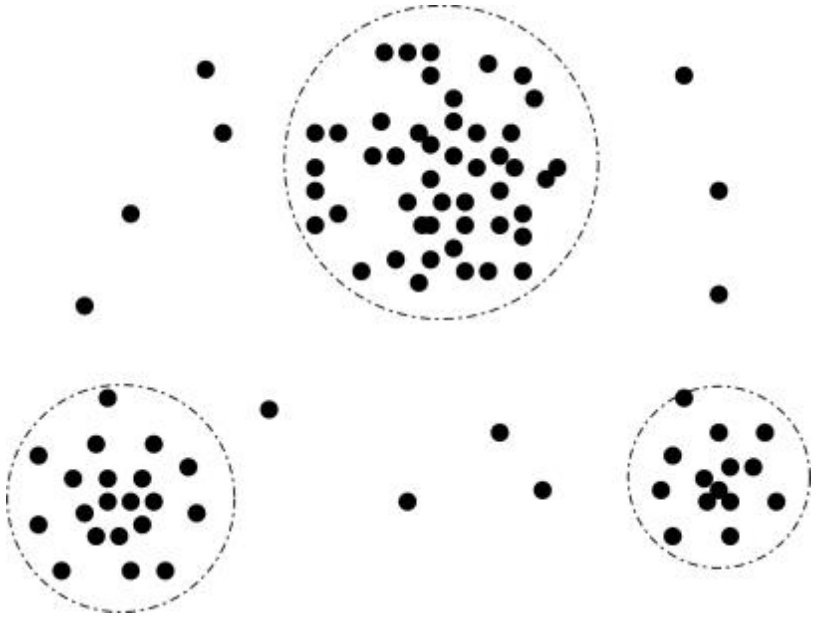


How to Handle Noisy Data?

Smoothing: moving average, exponential smoothing

Regression: smooth by fitting the data into regression functions.

Outlier Analysis: detect and remove outliers.



Semi-supervised: combined computer and human inspection

Data Transformation by Normalization

Data Transformation

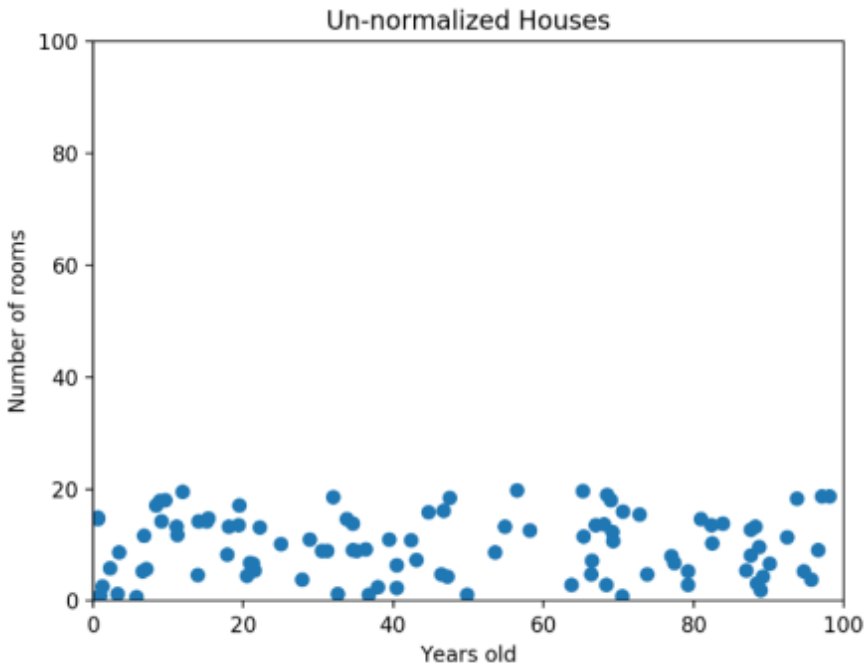
Data Transformation

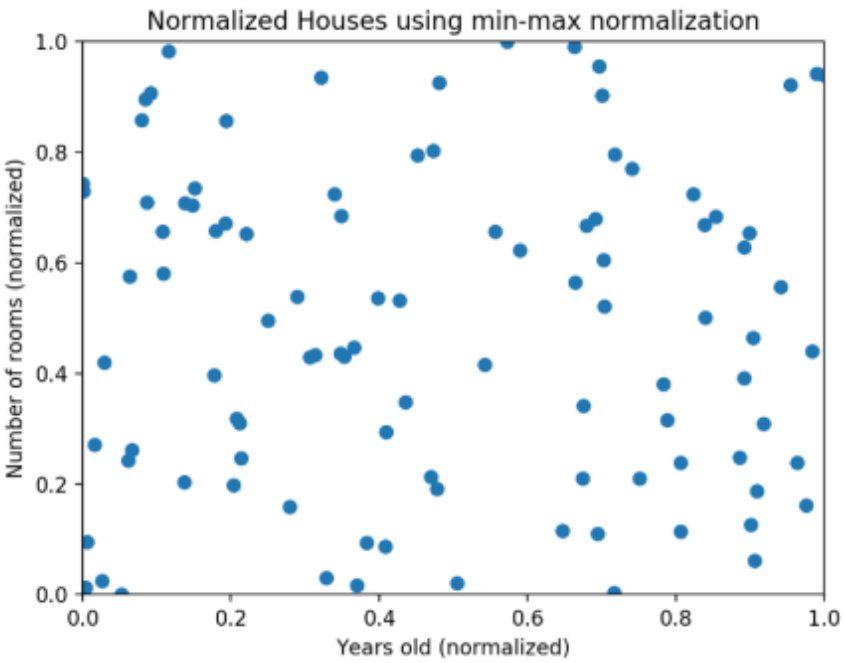
Data are transformed or consolidated so that the resulting mining process may be more efficient, and the patterns found may be easier to understand.

Normalisation: Scaled to fall within a smaller, specified range

- min-max normalisation
- z-score normalisation
- normalisation by decimal scaling

Normalisation





Normalisation

The measurement unit used in an attribute can affect data analysis. In normalisation, attributes are scaled to fall within a smaller, specified range, such that all attributes have an equal weight in the analysis.

Min-max normalisation: maps a value v of A to v' in the range $[new\_min_A, new\_max_A]$.

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

Example: Normalise the income range 212,000 - 298,000 to [0.0, 1.0]. What is the normalised value for 273,600?

Normalisation

z-score normalisation: the values for an attribute *A* are normalised based on the mean and standard deviation of *A*.

$$v' = \frac{v - \mu_A}{\sigma_A}$$

Useful when the actual minimum and maximum of attribute *A* are unknown, or when there are outliers that dominate the min-max normalisation.

Example: Suppose that the mean and standard deviation of the values for the attribute income are 254,000 and 216,000, respectively. What is the z-score normalised value for 273,600?

Normalisation

Normalisation by decimal scaling: normalises by moving the decimal point of values of attribute A.

$$v' = \frac{v}{10^j}$$

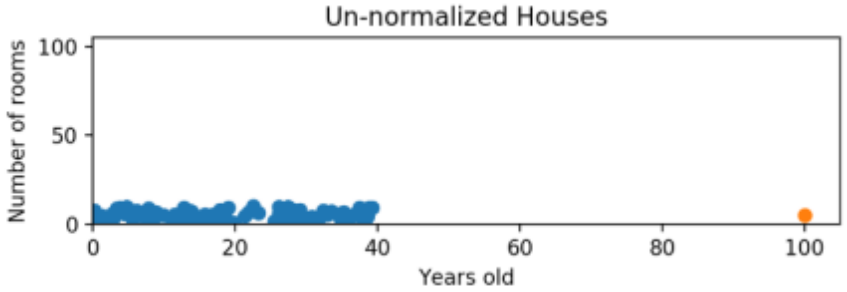
where j is the smallest integer such that $max(|v'|) < 1$.

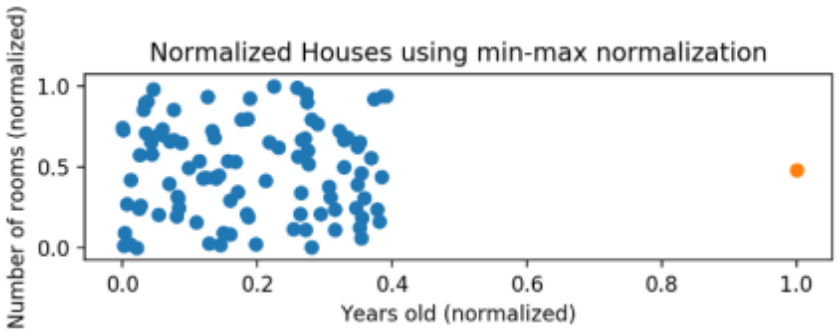
Example: Suppose that the recorded values of A range from -986 to 917. The maximum absolute value of A is 986. To normalise by decimal scaling, we therefore divide each value by 1000 ($j = 3$) so that -986 normalizes to -0.986 and 917 normalises to 0.917.

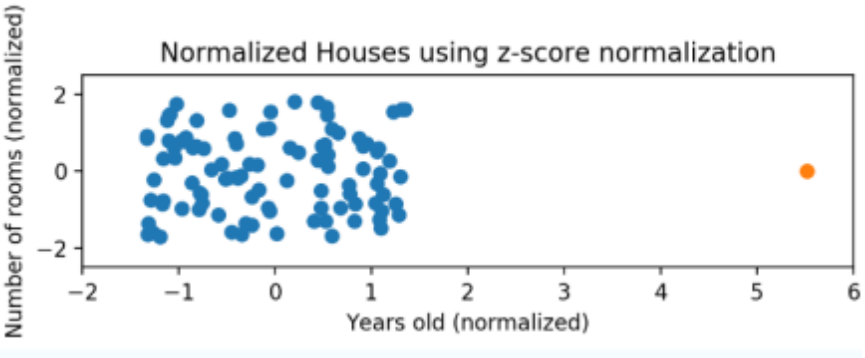
Note: it is necessary to save the normalisation parameters (e.g. the mean and standard deviation if using z-score normalisation) so that future data can be normalized in a uniform manner.

MIN-MAX vs Z-SCORE

- Min-max normalization: Guarantees all features will have the exact same scale but does not handle outliers well.
- Z-score normalization: Handles outliers, but does not produce Normalized data with the exact same scale.







Dimensionality Reduction

Data Reduction

Data Reduction
Obtain a reduced representation of the data set.

- Much smaller in volume but yet produces almost the same analytical results
- **Why data reduction?** Complex analysis may take a very long time to run on a complete data set.
- **Methods**
  - Attribute subsect selection
  - Dimensionality reduction: principal component analysis (PCA)
    multidimensional scaling (MDS)
    t-SNE

Dimensionality Reduction

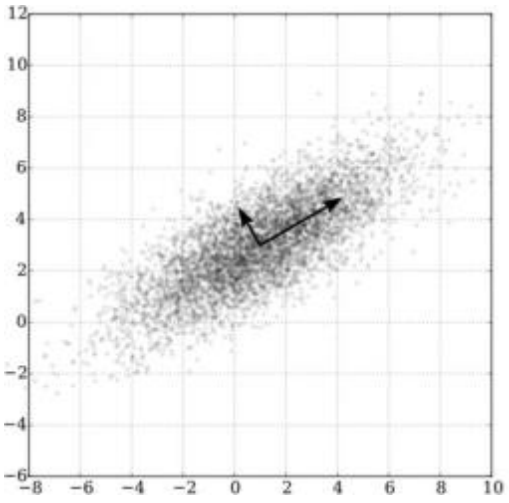
Principal component analysis (PCA)

Principal Component Analysis

Principal Component Analysis (PCA)

A statistical procedure that uses a transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components.

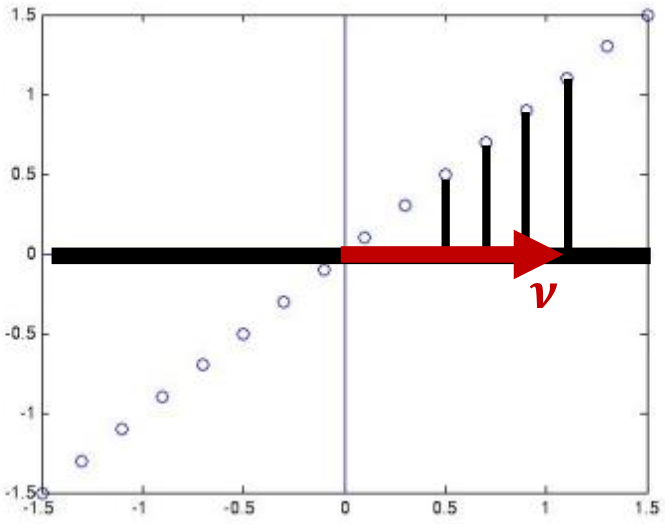
- The original data is projected onto a much smaller space, resulting in dimensionality reduction.
- **Method**: find the eigenvectors of the covariance matrix, and these eigenvectors define the new space.



Projection

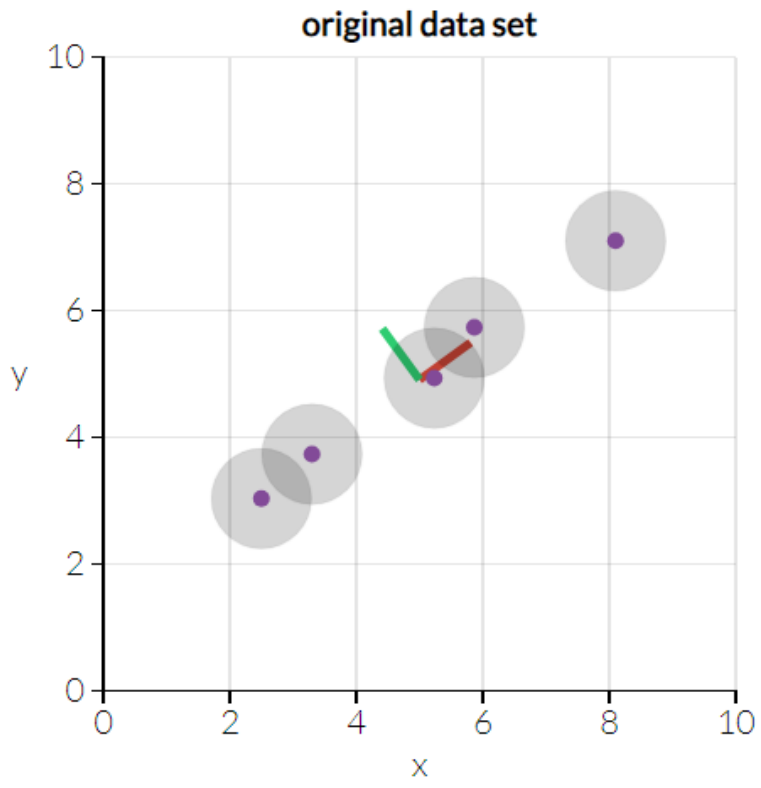
- Data points: $\boldsymbol{x}_i, i = 1, \cdots, n$
- Direction defined by $\boldsymbol{v}$
- Projecting $\boldsymbol{x}_i$ towards the direction, the new value is

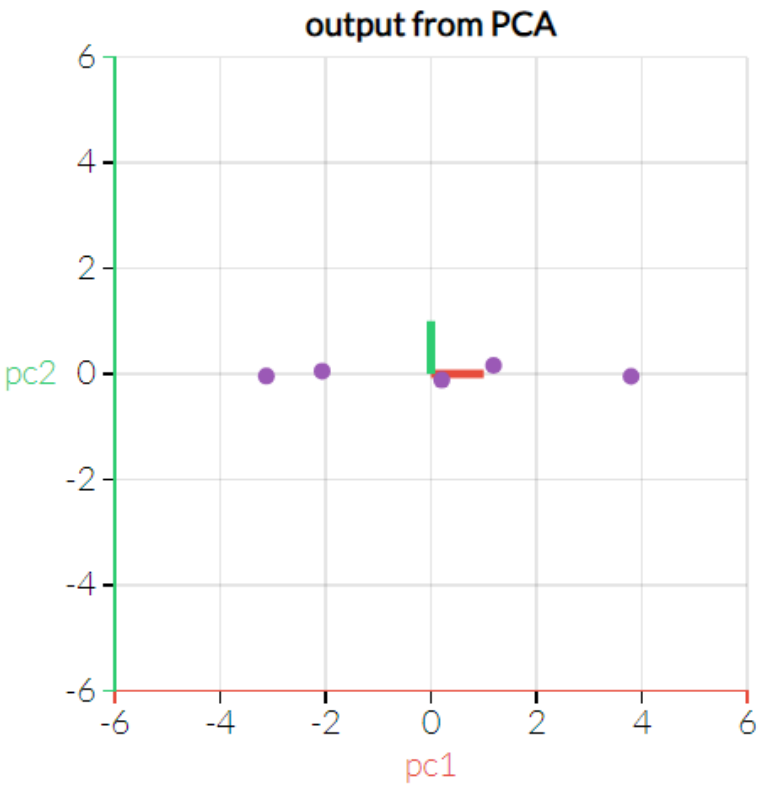
$$\boxed{y_i = \boldsymbol{v}^{\mathrm{T}} \boldsymbol{x}_i}$$

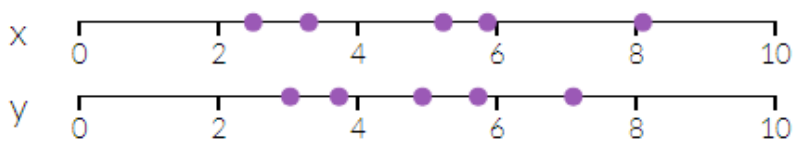


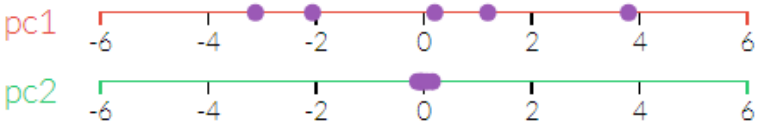
$$\boldsymbol{v} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

2D Example

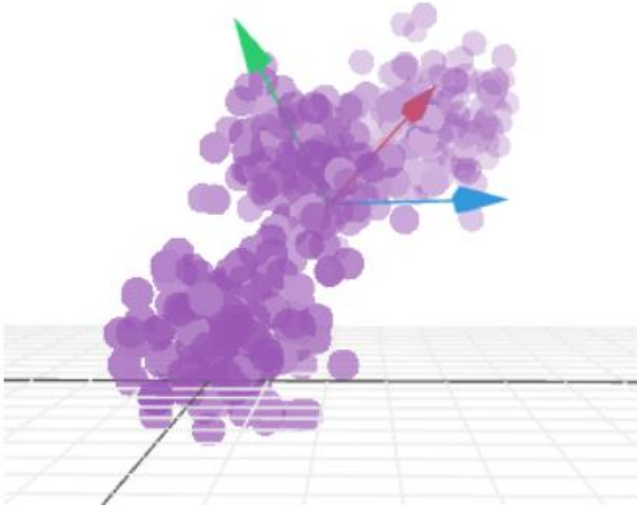

original data set
10
8
6
4
2
0
y
0
2
4
6
8
10
x

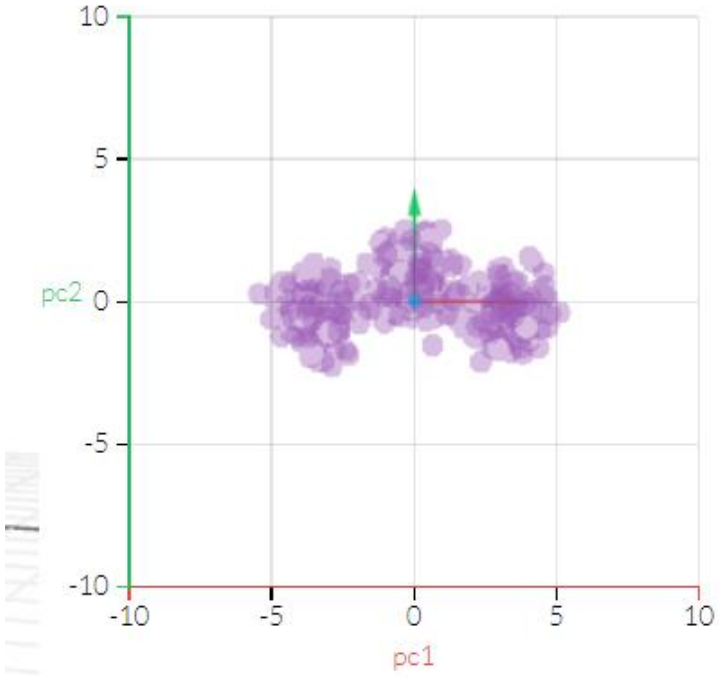

output from PCA
6
4
2
0
-2
-4
-6
pc2
-6
-4
-2
0
2
4
6
pc1

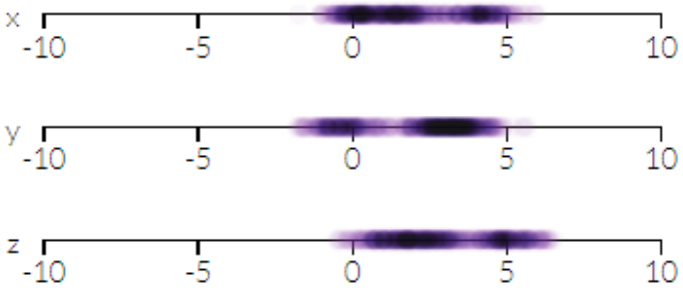

x
y
0
2
4
6
8
10

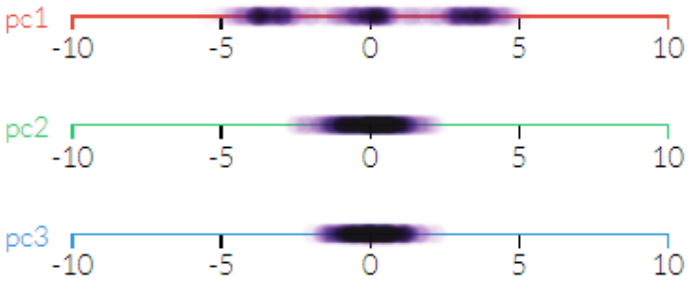

pc1
pc2
-6
-4
-2
0
2
4
6

3D Example




10
5
pc2 0
-5
-10
-10
-5
0
5
10
pc1


x
-10
-5
0
5
10
y
-10
-5
0
5
10
z
-10
-5
0
5
10


pc1
-10
-5
0
5
10
pc2
-10
-5
0
5
10
pc3
-10
-5
0
5
10

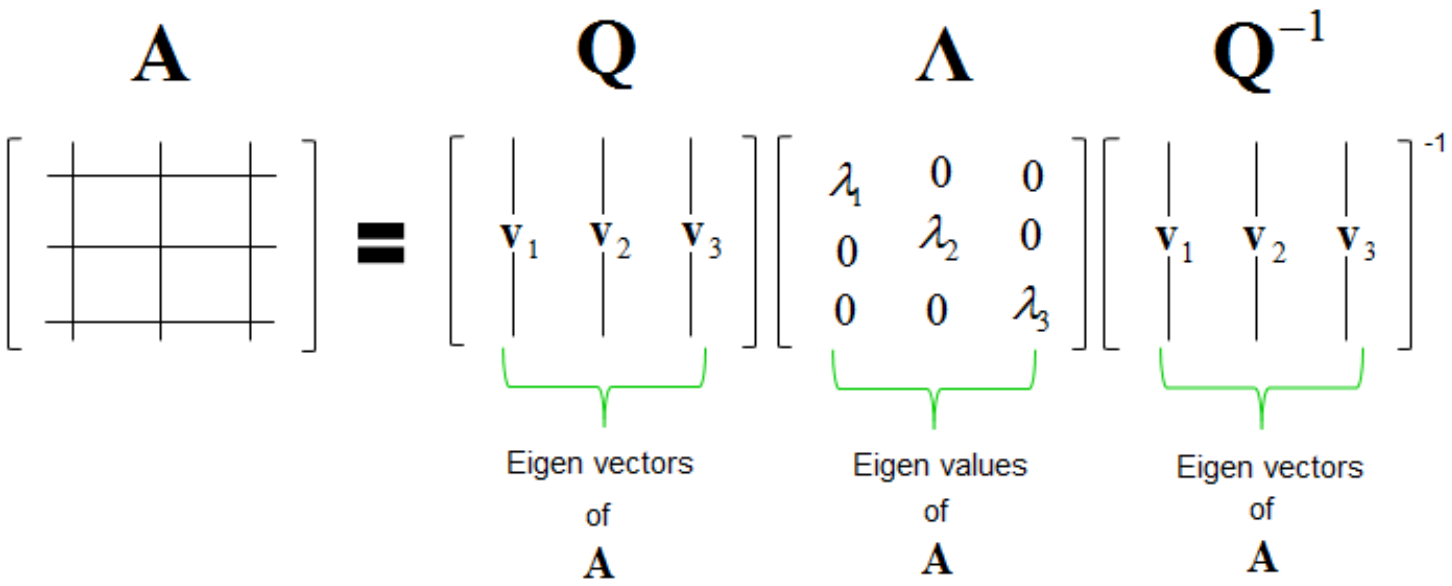
Principal Component Analysis

- To find the principal components (projections)
  - The projection should be **orthogonal**
  - After projection, the data should be mostly spread out, i.e. with the highest **variance**
- PCA based on eigenvalue decomposition.
- Given samples $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_D\}$, $\boldsymbol{x}_i \in R^{M\times 1}$
  - compute sample mean: $\boldsymbol{\mu} = \frac{\mathbf{1}}{\boldsymbol{D}} \sum_{\boldsymbol{i}} \boldsymbol{x}_{\boldsymbol{i}}$
  - compute sample variance: $\boldsymbol{\Sigma} = \frac{\mathbf{1}}{\boldsymbol{D}} \sum_{\boldsymbol{i}} (\boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{\mu})(\boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{\mu})^T$
  - compute eigenvalue and eigenvector of $\boldsymbol{\Sigma} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^{-\mathbf{1}}$:
    - $\boldsymbol{\Phi} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_M]$ – projection
    - $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_M)$ - the variance after projection
  - order eigen values $\lambda_1 > \cdots > \lambda_M$ and remain the first K eigenvalues
  - the eigenvectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{\boldsymbol{K}}$ would be the K principal components

  (projections) $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{v}_{\boldsymbol{K}}^{\mathrm{T}} \end{bmatrix} \rightarrow \boldsymbol{Y} = \boldsymbol{V}\boldsymbol{X}$

Eigenvalue decomposition - review

- Eigenvalue decomposition of an $M \times M$ matrix $\boldsymbol{A}$:
  - $\boldsymbol{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1}$, where $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_M)$, $\mathbf{Q} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_M]$
- $\boldsymbol{A}\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i$
  - $\boldsymbol{\lambda}_i$ is the eigenvalue (scalar), $i = 1, \cdots, M$
  - $\boldsymbol{v}_i$ is the eigenvector ($\boldsymbol{v}_i \in R^{M \times 1}$), $i = 1, \cdots, M$
  - $\boldsymbol{v}_i$ **and** $\boldsymbol{v}_j$ are orthogonal to each other, $i \neq j$



Eigenvalue decomposition - review

- Generalized eigenvalue decomposition for a pair of matrix $\boldsymbol{A}$ and $\boldsymbol{B}$: $\boldsymbol{A}\boldsymbol{v}_i = \lambda_i \boldsymbol{B}\boldsymbol{v}_i$
  - $\boldsymbol{\lambda}_i$ is the generalized eigenvalue, $i = 1, \cdots, M$
  - $\boldsymbol{v}_i$ is the generalized eigenvector, $i = 1, \cdots, M$;
  - $\boldsymbol{v}_i$ **and** $\boldsymbol{v}_j$ are orthogonal to each other

- Rayleigh quotient

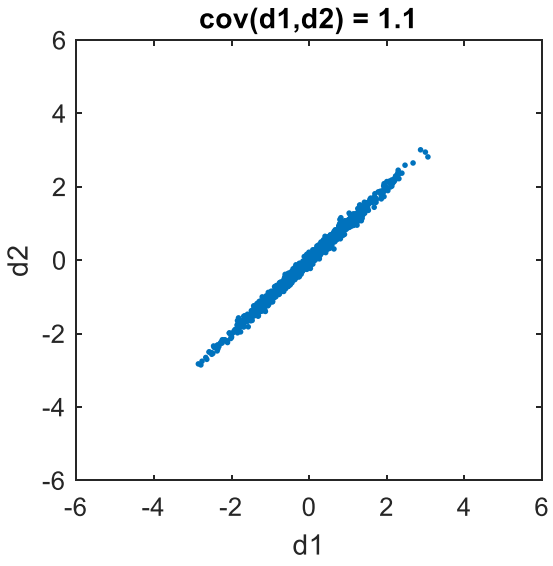
$$\boldsymbol{\lambda}_i = \frac{\boldsymbol{v}_i^T \boldsymbol{A}\boldsymbol{v}_i}{\boldsymbol{v}_i^T \boldsymbol{B}\boldsymbol{v}_i}$$

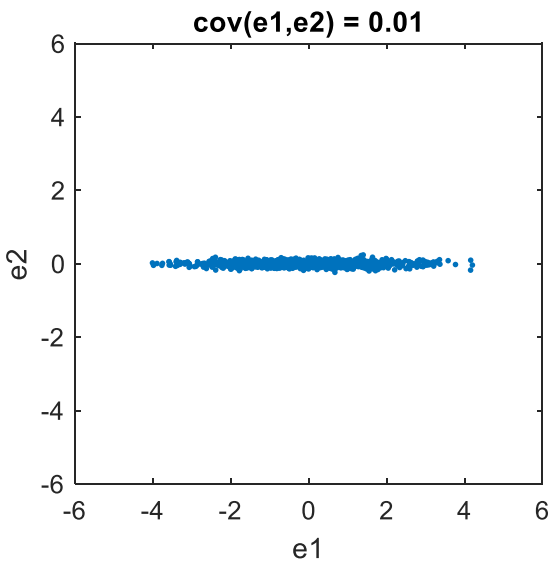
When $\boldsymbol{B} = \boldsymbol{I}$, $\lambda_i = \frac{\boldsymbol{v}_i^T \boldsymbol{A}\boldsymbol{v}_i}{\boldsymbol{v}_i^T \boldsymbol{v}_i}$

Covariance

- n data points: $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^{d_1} \\ \boldsymbol{x}^{d_2} \end{bmatrix} = \begin{bmatrix} x_1^{d_1} & \cdots & x_n^{d_1} \\ x_1^{d_2} & \cdots & x_n^{d_2} \end{bmatrix}_{2\times n}$
- The covariance between two directions (d1, d2) is

$$cov(d_1, d_2) = \frac{\sum_{i=1}^{n}(x_i^{d_1} - \bar{x}^{d1})(x_i^{d_2} - \bar{x}^{d_2})}{n}$$



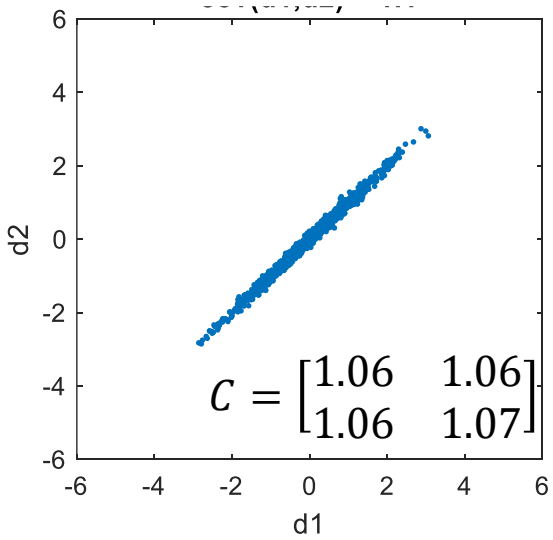


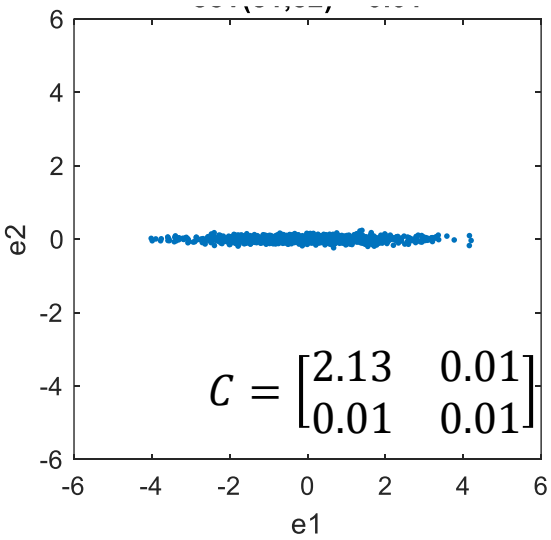
Covariance matrix

- The covariance matrix of two directions d1, d2 is

$$\mathrm{C}(d1, d2) = \begin{bmatrix} var(d_1) & cov(d_1, d_2) \\ cov(d_2, d_1) & var(d_2) \end{bmatrix}$$

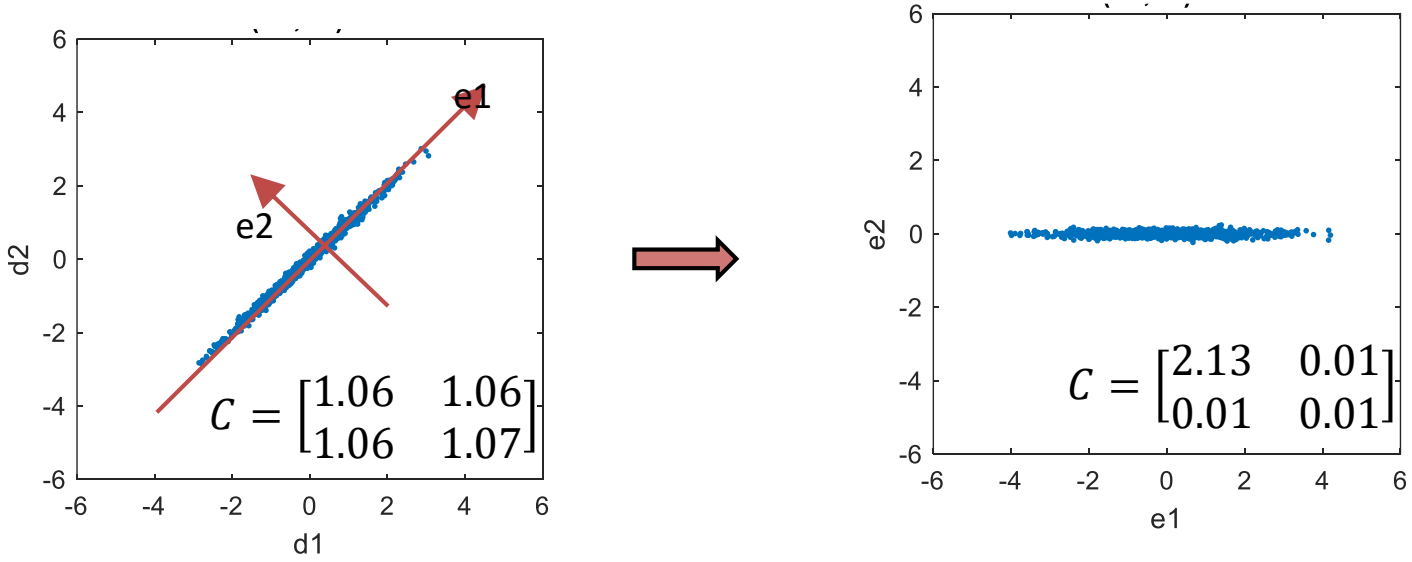
- The diagonal element (variance) indicates the spread along one direction
- The off-diagonal element (covariance) indicates the correlation (redundancy) between two directions
- Matrix form: $\boldsymbol{C}_X = \boldsymbol{X}\boldsymbol{X}^T$ (after removing the mean





Principal Component Analysis

- Principal component analysis, or PCA, is a dimensionality reduction method that is often used to reduce the redundancy in the data set
- To find some directions that preserve most information (variation/energy) of the data points after projection
- To diagonalize the covariance matrix
  - Minimize redundancy (covariance)
  - Maximize signal energy (variance)



Principal Component Analysis

- n data points: $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}^{d_1} \\ \boldsymbol{x}^{d_2} \end{bmatrix} = \begin{bmatrix} x_1^{d_1} & \cdots & x_n^{d_1} \\ x_1^{d_2} & \cdots & x_n^{d_2} \end{bmatrix}_{2\times n}$
- covariance matrix: $\boldsymbol{C}_{\mathbf{X}} = \boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$
- one projection direction: $\boldsymbol{v_i} = \begin{bmatrix} v_{i1} \\ v_{i2} \end{bmatrix}_{2\times 1}$
- n data points after projection: $\mathbf{y}^{\boldsymbol{v}_i} = \boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{X} = [y_1^{v_i}, \cdots, y_n^{v_i}]$
- variance along $\boldsymbol{v}_i$: $var(\boldsymbol{v}_i) = \mathbf{y}^{\boldsymbol{v}_i}(\mathbf{y}^{\boldsymbol{v}_i})^T = \boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{X}\boldsymbol{X}^{\mathbf{T}}\boldsymbol{v}_i = \boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{C}_{\mathbf{X}}\boldsymbol{v}_i$

$$\boldsymbol{C}_Y = \begin{bmatrix} \boldsymbol{v}_1^T\boldsymbol{C}_X\boldsymbol{v}_1 & \boldsymbol{v}_1^T\boldsymbol{C}_X\boldsymbol{v}_2 \\ \boldsymbol{v}_2^T\boldsymbol{C}_X\boldsymbol{v}_1 & \boldsymbol{v}_2^T\boldsymbol{C}_X\boldsymbol{v}_2 \end{bmatrix}$$

Principal Component Analysis

$$\boldsymbol{C}_Y = \begin{bmatrix} \boldsymbol{v}_1^T \boldsymbol{C}_X \boldsymbol{v}_1 & \boldsymbol{v}_1^T \boldsymbol{C}_X \boldsymbol{v}_2 \\ \boldsymbol{v}_2^T \boldsymbol{C}_X \boldsymbol{v}_1 & \boldsymbol{v}_2^T \boldsymbol{C}_X \boldsymbol{v}_2 \end{bmatrix}$$

- Eigenvalue decomposition: $\boldsymbol{C}_{\mathbf{X}} \boldsymbol{v} = \lambda \boldsymbol{v}$
- eigenvalues: $\{\lambda_1, \lambda_2\}$, eigenvectors: $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$

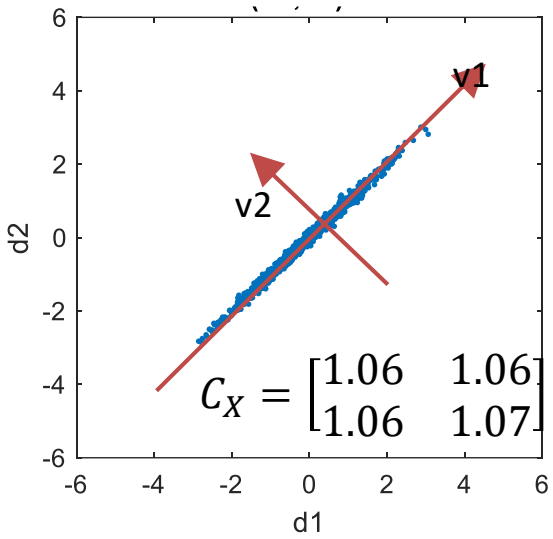
- Rayleigh quotient: $\lambda = \frac{\boldsymbol{v}^T \boldsymbol{C}_{\mathbf{X}} \boldsymbol{v}}{\boldsymbol{v}^T \boldsymbol{v}} \rightarrow \boldsymbol{\lambda} = \boldsymbol{v}^T \boldsymbol{C}_{\mathbf{X}} \boldsymbol{v}$ s.t. $\boldsymbol{v}^T \boldsymbol{v} = \mathbf{1}$
- To maximize the variance after projection (the diagonal item of the covariance matrix)
  - first direction: $\boldsymbol{v}_1$ is the eigenvector corresponding to the largest eigenvalue
  - second direction: $\boldsymbol{v}_2$ is the eigenvector corresponding to the second largest eigenvalue
  - …

- The off-diagonal item of the covariance matrix
  - $\boldsymbol{y}^{v_1} = \boldsymbol{v}_1^T \boldsymbol{X}, \boldsymbol{y}^{v_2} = \boldsymbol{v}_2^T \boldsymbol{X}$
  - $cov(\boldsymbol{v}_1, \boldsymbol{v}_2) = \boldsymbol{y}^{v_1} (\boldsymbol{y}^{v_2})^{\mathrm{T}} = \boldsymbol{v}_1^{\mathrm{T}} \boldsymbol{C}_X \boldsymbol{v}_2 = \boldsymbol{\lambda}_2 \boldsymbol{v}_1^T \boldsymbol{v}_2 = \mathbf{0}$

- $\boldsymbol{C}_Y = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$

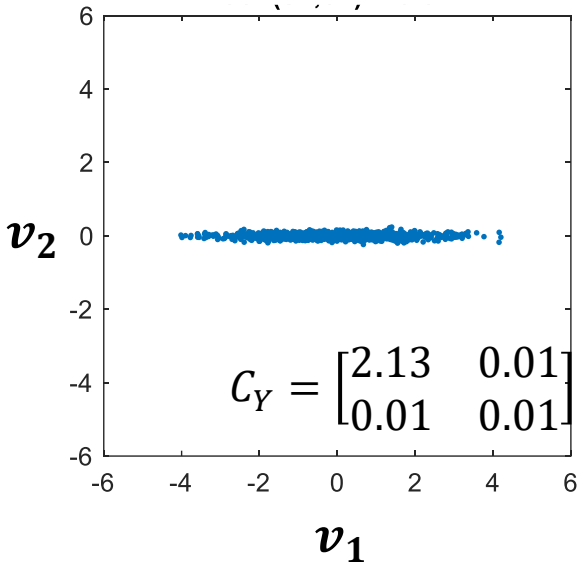
Principal Component Analysis

$$\boldsymbol{C_Y} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$



$$\boldsymbol{v_1} = \begin{bmatrix} 0.71 \\ 0.71 \end{bmatrix}, \lambda_1 = 2.13$$

$$\boldsymbol{v_2} = \begin{bmatrix} -0.71 \\ 0.71 \end{bmatrix}, \lambda_2 = 0.01$$



Principal Component Analysis

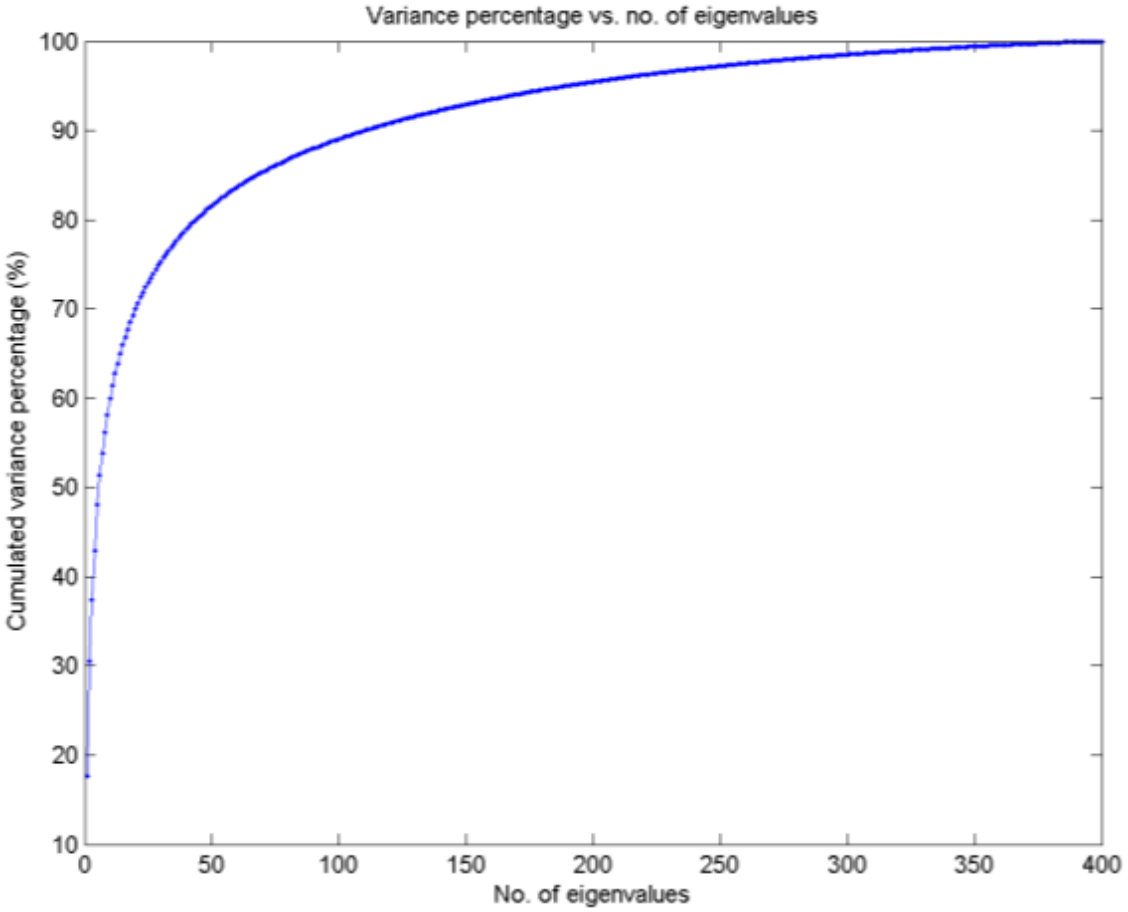
- To find the principal components (projections)
  - The projection should be **orthogonal**
  - After projection, the data should be mostly spread out, i.e. with the highest **variance**
- PCA based on eigenvalue decomposition.
- Given samples $\boldsymbol{X} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_D\}$, $\boldsymbol{x}_i \in R^{M\times 1}$
  - compute sample mean: $\boldsymbol{\mu} = \frac{\mathbf{1}}{\boldsymbol{D}} \sum_{\boldsymbol{i}} \boldsymbol{x}_{\boldsymbol{i}}$
  - compute sample variance: $\boldsymbol{\Sigma} = \frac{\mathbf{1}}{\boldsymbol{D}} \sum_{\boldsymbol{i}} (\boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{\mu})(\boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{\mu})^T$
  - compute eigenvalue and eigenvector of $\boldsymbol{\Sigma} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^{-\mathbf{1}}$:
    - $\boldsymbol{\Phi} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_M]$ – projection
    - $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_M)$ - the variance after projection
  - order eigen values $\lambda_1 > \cdots > \lambda_M$ and remain the first K eigenvalues
  - the eigenvectors $\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{\boldsymbol{K}}$ would be the K principal components

  (projections) $\boldsymbol{V} = \begin{bmatrix} \boldsymbol{v}_1^{\mathrm{T}} \\ \vdots \\ \boldsymbol{v}_{\boldsymbol{K}}^{\mathrm{T}} \end{bmatrix} \rightarrow \boldsymbol{Y} = \boldsymbol{V}\boldsymbol{X}$

Principal Component Analysis

Determining K: cumulative variance proportion explained by the first *n* principal components

$$p(n) = \frac{\sum_{i=1}^{n} \lambda_i}{\sum_{i=1}^{N^2} \lambda_i}$$



Principal Component Analysis

- Application of PCA
  - Pattern extraction and visualization
  - Dimensionality reduction for classification
  - Data compression

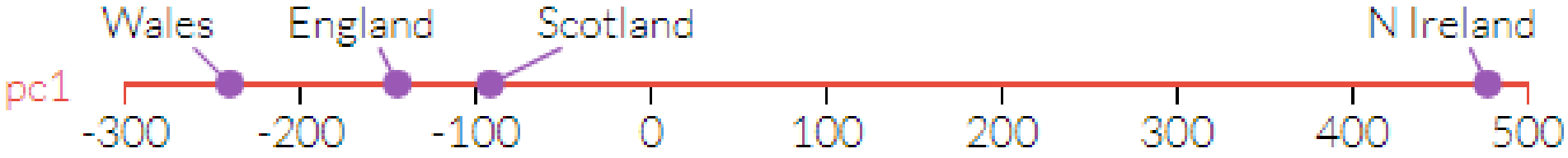
- Advantages:
  - It helps in data compression and removes correlated features.
  - It helps in Speeding up other Data Mining Algorithms.
  - It converts high-dimensional data into low-dimensional data which improves and make visualization easy.

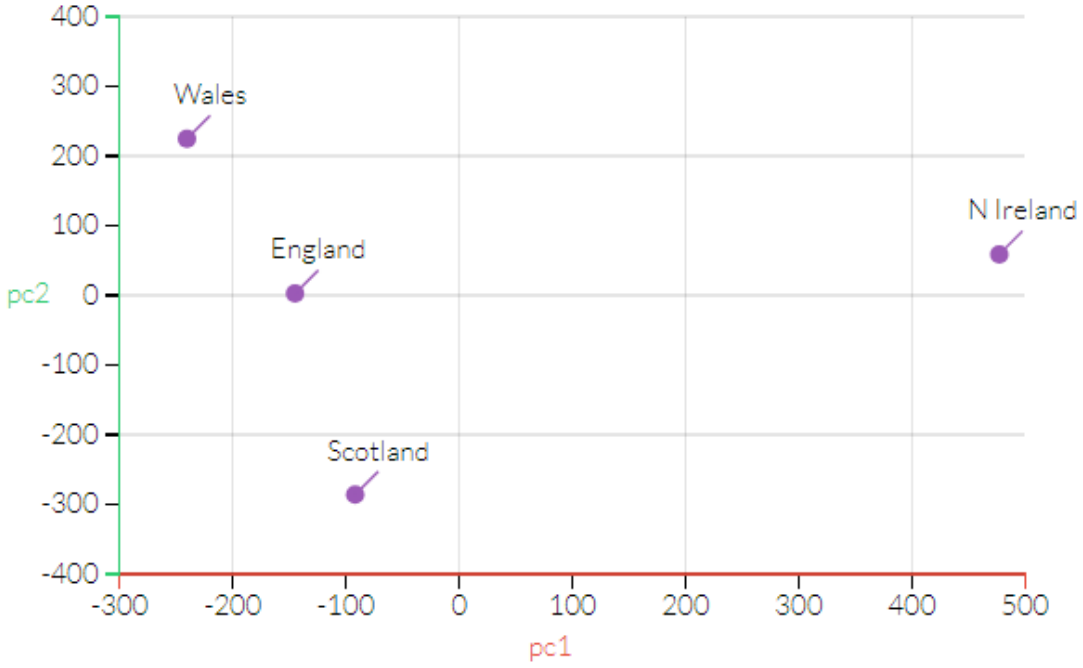
- Disadvantages:
  - It may lead to some amount of data loss.
  - It does not consider class separability and thus might not be optimal for classification problem.

Principal Component Analysis

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

Principal Component Analysis





Dimensionality Reduction

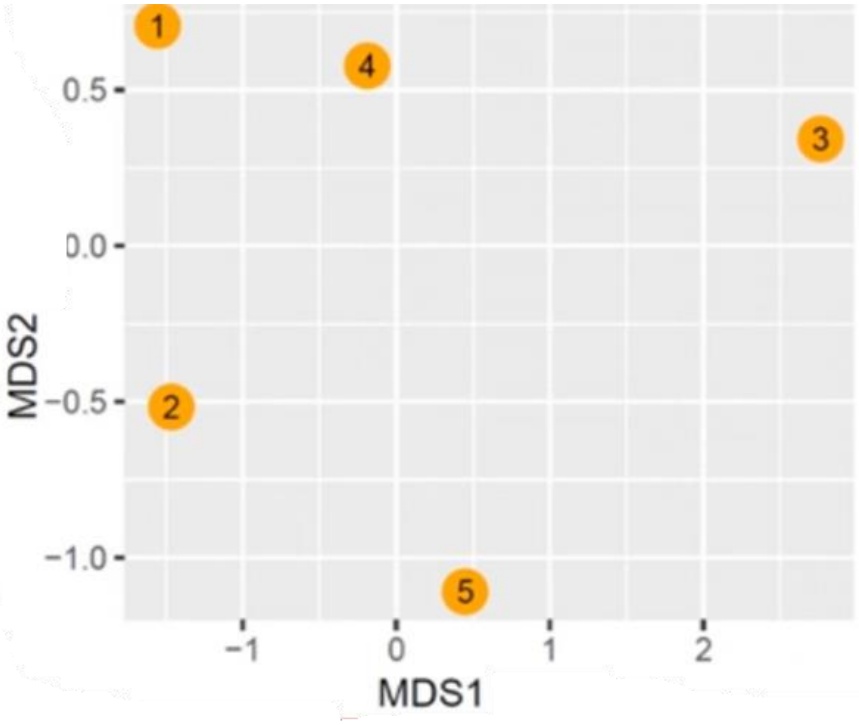
Multidimensional scaling (MDS)

Multidimensional scaling (MDS)

- Multidimensional scaling (MDS) is an unsupervised dimension reduction technique for data visualization
- Mapping data (observations) from a higher dimension down to a lower dimension, in such a way that the similarity (distance) between observations is preserved
- E.g. if observations are “similar” in 5D, they should be “similar” when mapped down to 2D with MDS
- MDS is commonly used to visualize complex, high-dimensional data, and to identify patterns and relationships that may not be apparent in the original space.

Multidimensional scaling (MDS)

ID	DBP	SBP	BMI	Chol
1	82	132	18	192
2	86	133	20	240
3	98	145	35	140
4	85	139	22	170
5	87	145	28	218



Distance matrix

	1	2	3	4	5
1	0	1.426	4.356	1.463	2.741
2	1.426	0	4.327	2.051	2.314
3	4.356	4.327	0	3.093	2.866
4	1.463	2.051	3.093	0	1.809
5	2.741	2.314	2.866	1.809	0

Multidimensional scaling (MDS)

- Classical multidimensional scaling (MDS) is also called Principal Coordinates Analysis (PCoA)
- General steps (very similar to PCA)
  - Compute distance matrix (e.g. Euclidean distance)
  - Eigenvalue decomposition
  - Retain eigenvectors with largest eigenvalues
  - Apply projection

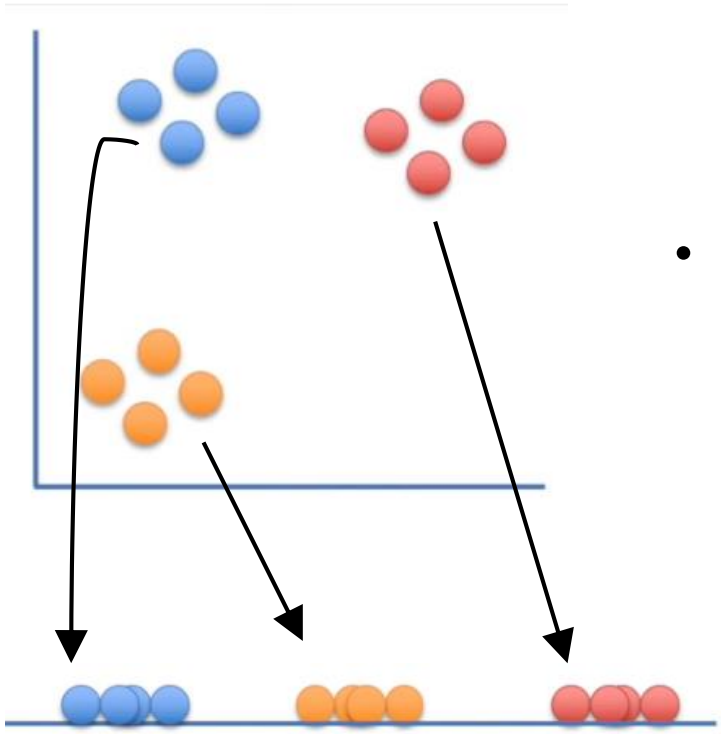
Dimensionality Reduction

t-SNE

t-SNE

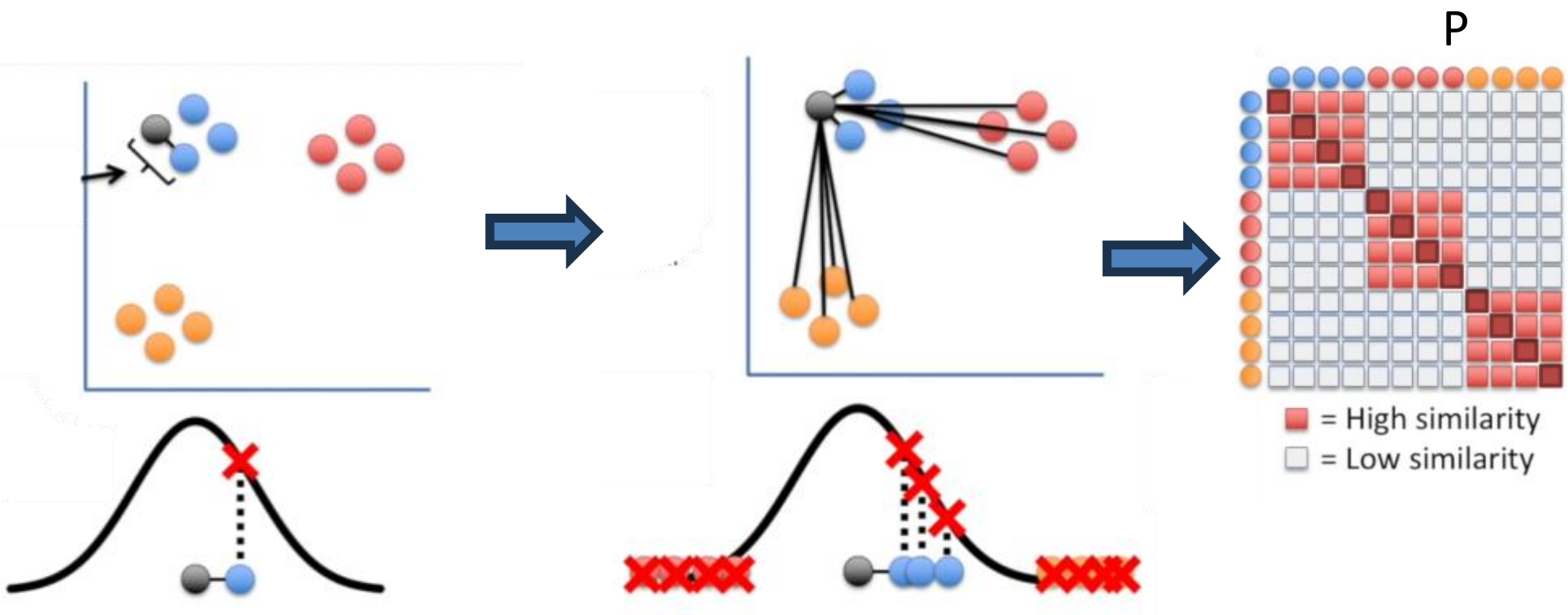
- t-SNE (t-distributed Stochastic Neighbor Embedding) is an unsupervised non-linear dimensionality reduction technique
- t-SNE is often used to visualize complex datasets into two and three dimensions, allowing us to understand more about underlying patterns and relationships in the data.
- t-SNE focuses on preserving the pairwise similarities (distance) between data points in a lower-dimensional space.
- It focuses on preserving small pairwise distances, e.g. the clustering of the data points.

t-SNE



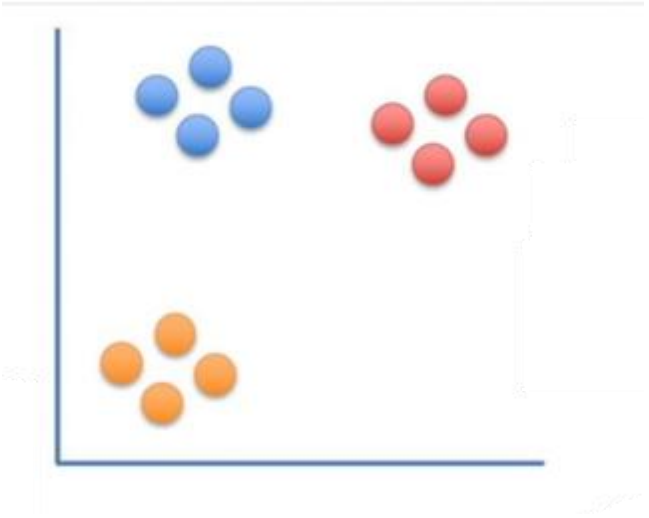
- What t-SNE does is to find way to project into a lower dimensional space so that the clustering in the high dimensional space is preserved.
- It focuses on preserving small pairwise distances, e.g. the data points within each cluster.

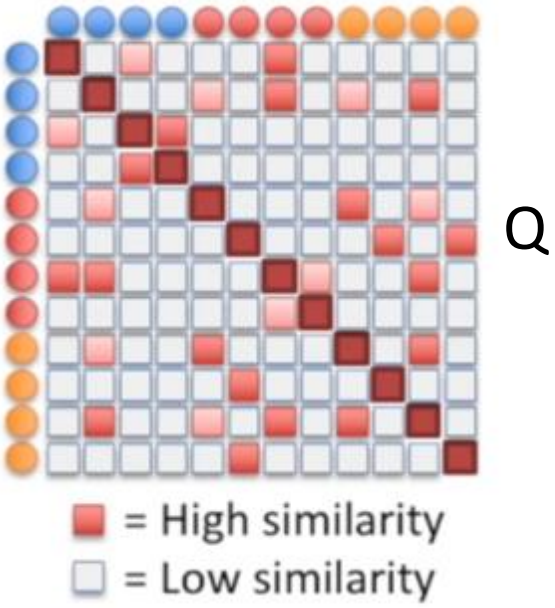
t-SNE

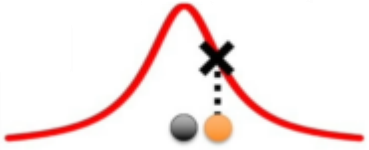


$$p_{ij} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{d_{ij}^2}{\sigma^2}}$$

t-SNE

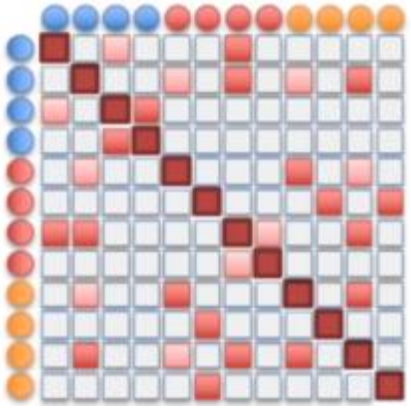




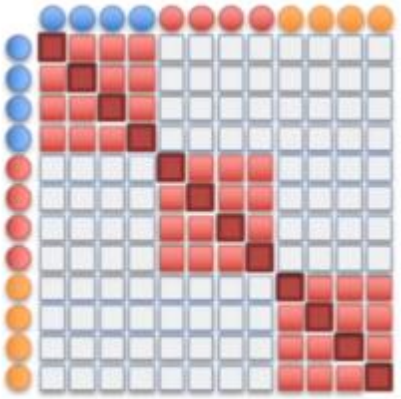


t-SNE

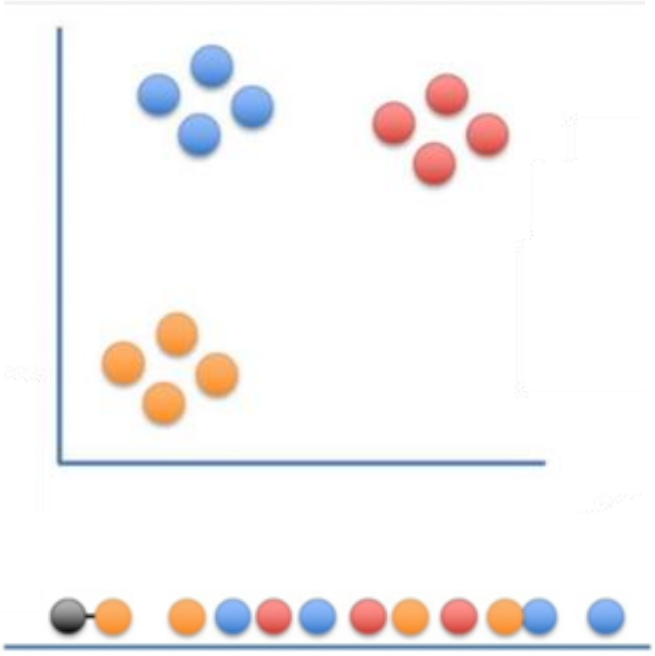
Q

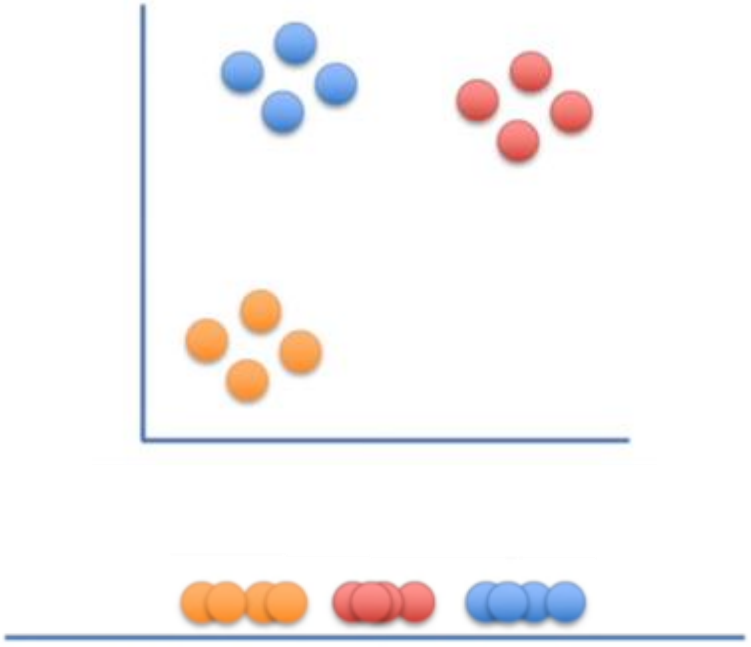


- Iterative optimization
- Minimize the difference between P (similarity matrix in high dimension) and Q (similarity matrix in low dimension), which is called KL divergency



P



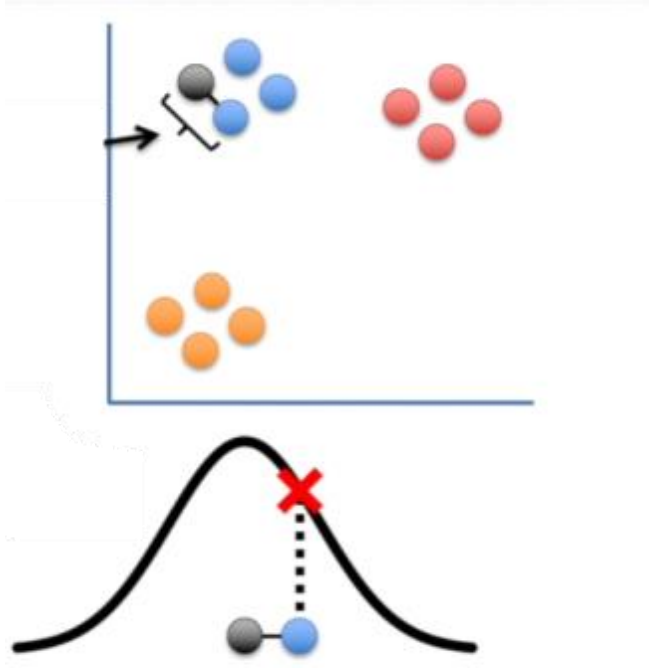


t-SNE

1. Initialize points randomly / with PCA
2. Compute pairwise distances
3. Convert distances to similarity measures
4. Optimize KL divergence via gradient descent

t-SNE

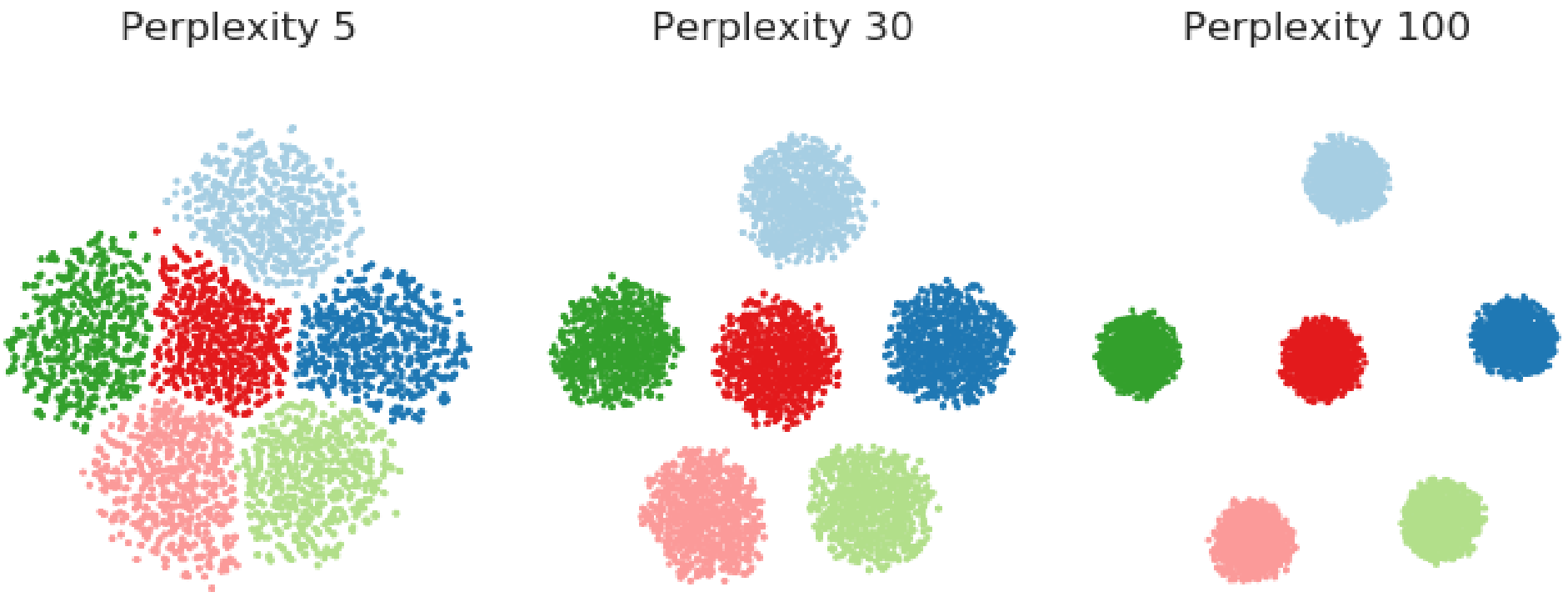
- Perplexity is a hyperparameter that measures how many neighbors we want to include for each point.
- It simply controls the variance of Gaussian kernel.
- A high perplexity means more data points are included in the neighborhood. It indicates a larger variance and a flat Gaussian distribution.
- A low perplexity means less points are included in the neighborhood. It indicates a peaked Gaussian distribution with small variance.
- Typical range [5, 50]



$$p_{ij} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{d_{ij}^2}{\sigma^2}}$$

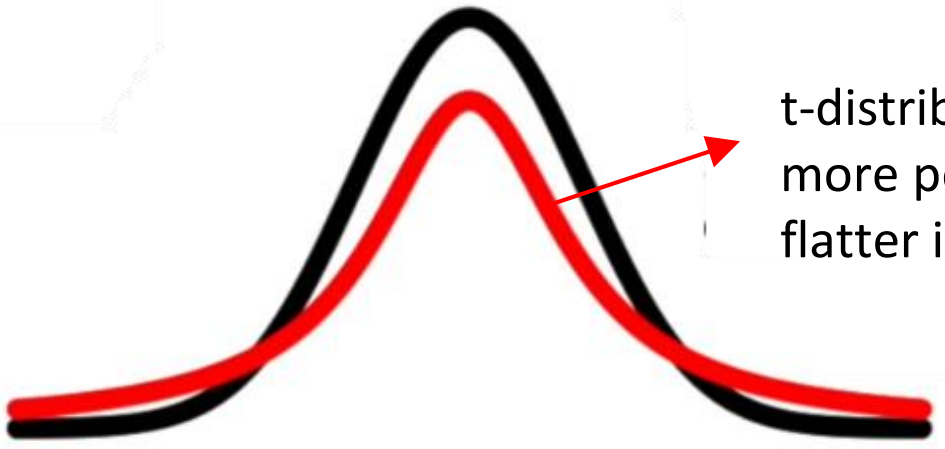
t-SNE

- Original data: six 10-dimensional Gaussian balls (n=1000 in each ball) far from each other
- Large perplexity tends to shrink into denser clusters



t-SNE

- Similarity measure: t-distribution in stead of Gaussian distribution



t-distribution:
more peaky in the middle and flatter in the tail

The “t-distribution” is the “t” in t-SNE.

PCA vs MDS vs t-SNE

- PCA: linear projection, preserve the variance in the data
  - Application: data compression, data visualization, feature extraction,
  - Method: eigen-value decomposition on the covariance matrix
- MDS: linear projection, preserve the pair-wise distance of all the data
  - Application: data visualization,
  - Method: eigen-value decomposition on the distance matrix
- t-SNE: nonlinear projection, preserve the pair-wise similarity in local neighborhoods,
  - Application: data visualization
  - Method: iterative optimization that minimize the KL divergency between similarity matrix

Summary

Data cleaning routines attempt to fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data.

Data transformation routines convert the data into appropriate forms for mining.

Data reduction techniques obtain a reduced representation of the data while minimizing the loss of information content. Methods discussed include PCA, MDS and t-SNE.

Questions?

also please use the forum on QM+