

# Machine Learning

## Data Preprocessing

Created by: ADF dan SSD





# Data

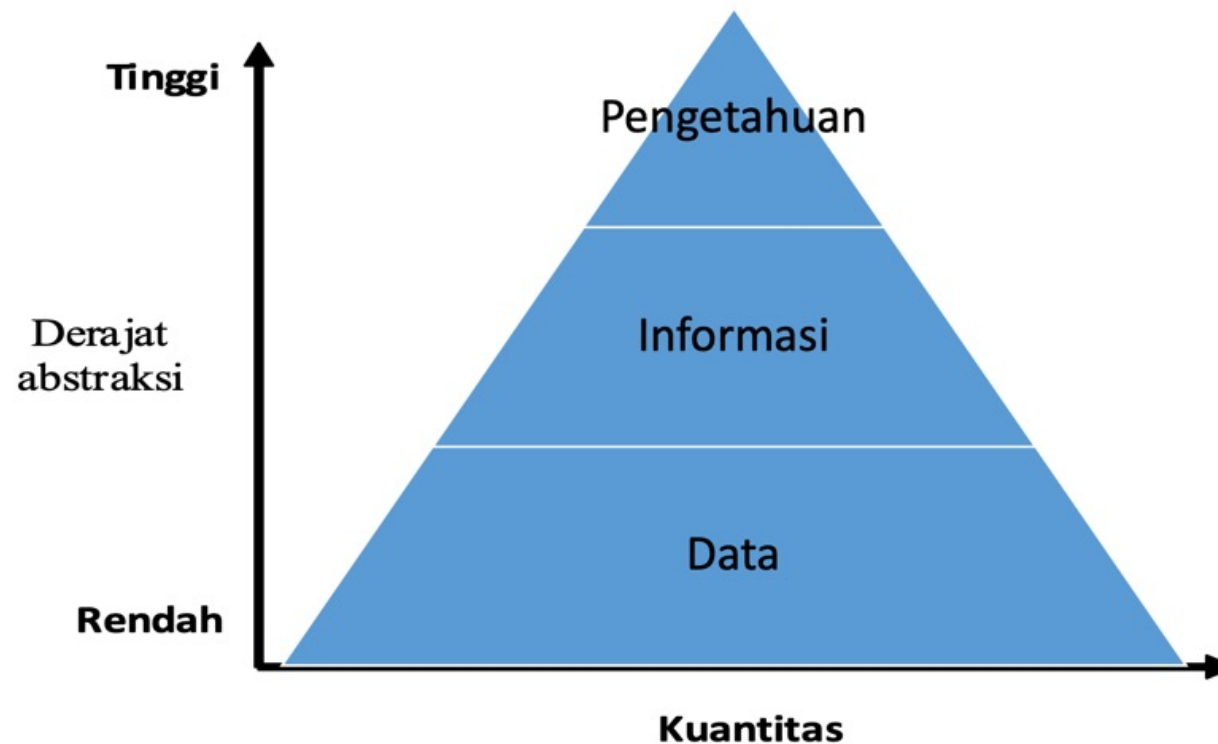


## Data dan Informasi

- ▶ **Data** adalah deskripsi tentang benda, kejadian, aktivitas, dan transaksi, yang **tidak mempunyai makna** atau tidak berpengaruh secara langsung kepada pemakai.
- ▶ **Informasi** adalah data yang telah **diproses** dan **diformat** menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam **pengambilan keputusan** saat ini atau saat mendatang (Walton, 2016).



# Hirarki Data, Informasi, dan Knowledge





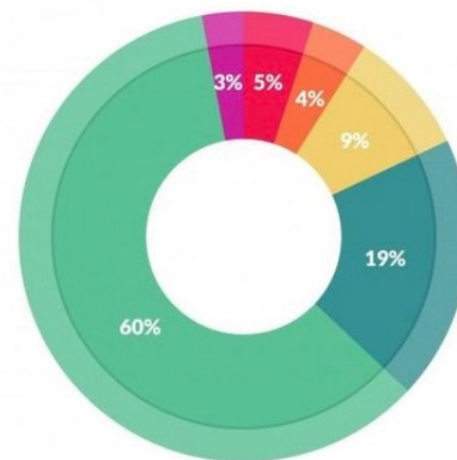
# Data Importance

- ▶ Before we dive in too deeply in the algorithms, let's talk about the **data** first
- ▶ If your data is bad, even the smartest algorithm will break
- ▶ Example:
  - Error, noise, outliers, duplicate data, missing value, inconsistent data, timeliness, relevance, etc.



# Data Importance

- ▶ If your data is bad, even the smartest algorithm will break
- ▶ Example:
  - Error, noise, outliers, duplicate data, missing value, inconsistent data, timeliness, relevance, etc.
- ▶ Data preparation accounts for around 80% of data scientists work.



What data scientists spend the most time doing

- Building training sets: 3%
- Cleaning and organizing data: 60%
- Collecting data sets: 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%



# Kualitas Data

- Apa yang menjadi permasalahan dengan data?
- Bagaimana mendeteksinya?
- Apa yang bisa dilakukan?
- Contoh **masalah** kualitas data:





# Dataset

- ▶ Database
- ▶ IoT Sensors
- ▶ Text
- ▶ Audio
- ▶ Speech
- ▶ Image
- ▶ Video





# Dataset

- Apakah dataset sudah valid?
- Apakah dataset terlalu kecil untuk kasus ini?
- Berapa jumlah data minimal untuk menghasilkan model machine learning yang baik?
- Apakah distribusi data sudah baik dan mudah dipahami?
- Apakah atribut (dimensi) data kurang atau malah terlalu banyak?



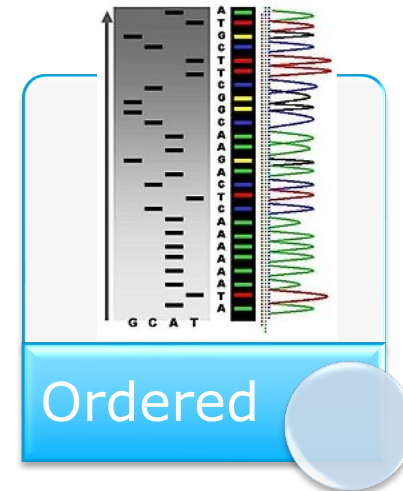
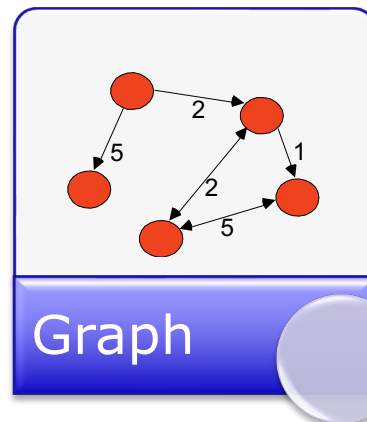
**Fakultas Informatika**  
School of Computing  
Telkom University

# Tipe Dataset

## ► Tipe Umum:

TID	Item
1	Susu, Coklat, Roti
2	Roti, Selai
3	Selai, Roti, Coklat, Susu
4	Roti, Coklat, Susu
5	Roti, Coklat, Susu, Biskuit

Record



## ► Karakteristik Umum:

- Dimensionality : jumlah atribut
- Sparsity : tingkat kepadatan data
- Resolution : pola tergantung pada skala data
- Size : tipe analisis tergantung pada ukuran data



# Data Quality

- ▶ High quantity of data prone to **measurement error ('noise')** and **Outliers** i.e. 'anomalous' objects
- ▶ How to check?
- ▶ **Variance and Covariance**



# Data Quality

## ► Variance

$$\sigma^2 = \frac{\sum (X - \bar{X})^2}{N} \quad (0a)$$

- Refers to the spread of the data set
- How far apart the numbers are in relation to the mean

## ► Covariance

$$\text{cov}(X, Y) = \sum_i^N \frac{(x_i - \bar{X})(y_i - \bar{Y})}{N - 1} \quad (0b)$$

- Refers to the measure of how two random variables will change together
- Used to calculate the correlation between variable



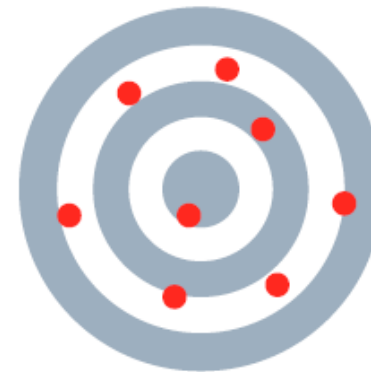
# Data Quality

## ► Precision and Bias

- Precision: closeness of repeated measurements to each other
- Bias: systematic deviation from the true underlying value
- Variance: reciprocal of Precision



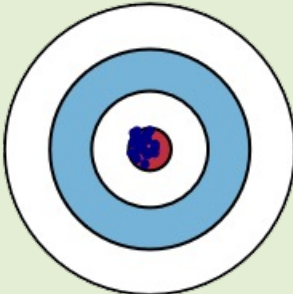
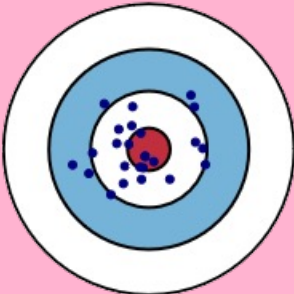


high precision  
high bias.



low precision  
low bias



# Data Quality

	Low Variance	High Variance	
Low Bias			
High Bias			
	High Precision	Low Precision	



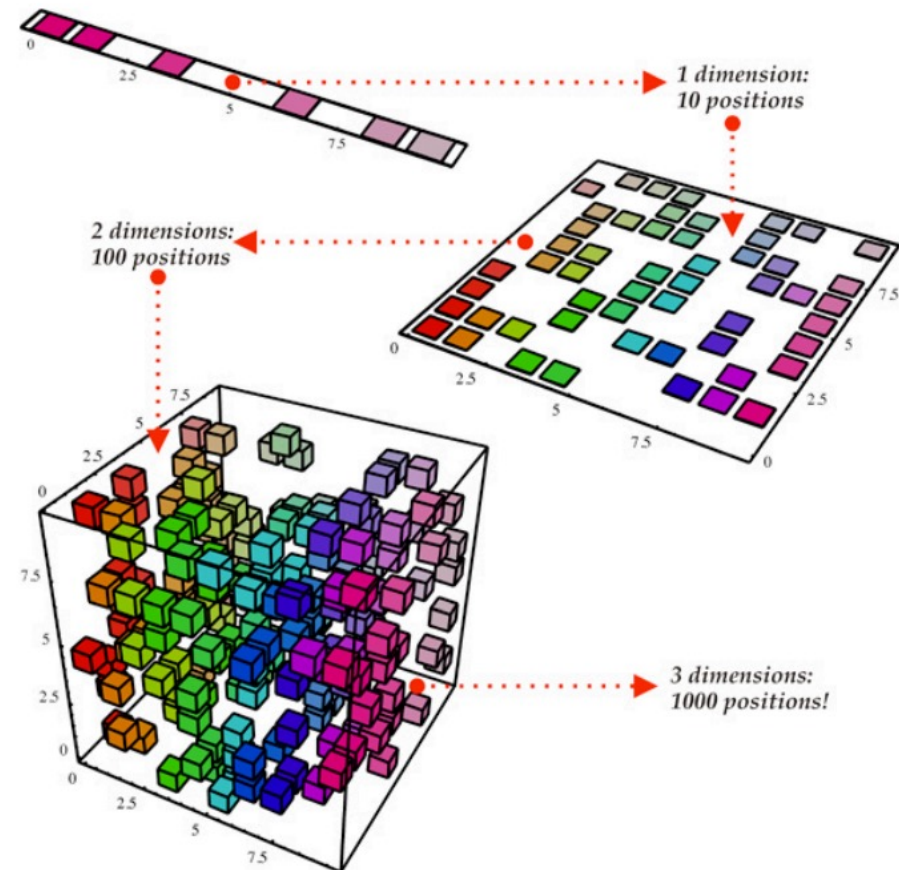
# Data Points vs Dimensionality

- ▶ Dimensionality: Number of attributes
- ▶ Most traditional data analysis methods assume (many) more data points than dimensions
- ▶ Many interesting datasets have extremely high dimensions and few data objects



# The Curse of Dimensionality

- ▶ As the dimensionality grows, the data becomes increasingly sparse in the space
- ▶ (e.g.  $n$  dimensions of binary variables has  $2^n$  joint states)





# The Curse of Dimensionality

- ▶ Less (dimension) is More
- ▶ Learning from a high-dimensional feature space requires an enormous amount of training to ensure that there are several samples with each combination of values.
- ▶ With a fixed number of training instances, the predictive power reduces as the dimensionality increases.



# Data Preprocessing



# Data Preprocessing

- ▶ Technique that **involves transforming raw data** into an **understandable format**.
- ▶ Resolve data that is **incomplete, inconsistent, and/or lacking** in certain behaviors or trends,
- ▶ Resolve data that is **likely to contain many errors**.
- ▶ Reduce the **dimensionality**



# Kategori Data Preprocessing

- ▶ Bisa dibedakan menjadi 2:
  - Pemilihan berdasarkan **objek data** (*record*) untuk menganalisis atau creating/changing atribut
    - Contoh : Agregasi, sampling
  - Pemilihan **atribut** untuk menganalisis atau creating/changing atribut
    - Contoh: Pengurangan Dimensi, feature subset selection



## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



# Agregasi

- **Mengkombinasikan dua atau lebih atribut (atau objek) menjadi satu atribut (atau objek)**
- **Tujuannya:**
  - Pengurangan data baik secara jumlah atribut atau objek
  - Merubah skala misalkan penggabungan atribut kota dengan atribut propinsi dan negara
  - Mendapatkan data yang lebih “stabil” karena bisa didapatkan data dengan variabilitas yang kecil



## Pokok Bahasan

### ▶ Apa Praproses Data

- Agregasi
- Sampling
- Pengurangan dimensi
- Feature subset selection
- Feature creation
- Diskretisasi dan Binerisasi
- Transformasi atribut

### ▶ Similaritas & Disimilaritas

- Euclidean distance
- Minkowski distance
- Mahalanobis Distance
- Simple Matching
- Jaccard Coefficients
- Cosine
- Tanimoto
- Korelasi





# Sampling

- Merupakan teknik utama untuk memilih data dan biasanya digunakan untuk **investigasi** data dan **analisis data akhir**
- Sampling di statistik  $\neq$  sampling di data mining
  - Jika di statistik berkaitan dengan mahalanya atau lamanya pengumpulan keseluruhan data, jika di data mining data keseluruhan ada namun untuk memproses keseluruhan akan terlalu lama
- Prinsip utama sampling yang **efektif** adalah:
  - Output dengan penggunaan sampel sama bagusnya dengan penggunaan data keseluruhan  $\square$  berarti data sampel sudah representatif
  - Sampel sudah representatif jika memiliki properti yang mirip (dari segi interest) seperti data asli



# Tipe-tipe Sampling

## ➤ **Simple Random Sampling**

- Setiap item memiliki probabilitas yang sama untuk dipilih

## ➤ **Sampling without Replacement**

- Setiap item yang terpilih akan dikeluarkan dari populasi

## ➤ **Sampling with Replacement**

- Setiap item yang terpilih tidak dikeluarkan dari populasi, bisa saja terpilih lebih dari satu kali

## ➤ **Stratified Sampling**

- Data displit menjadi beberapa bagian; lalu diambil sampel secara acak dari tiap bagian

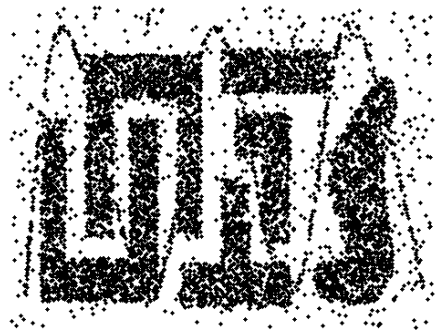


## Ukuran Sampel

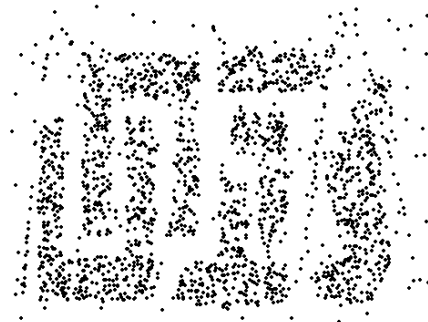
- **Ukuran sampel penting untuk ditentukan**
- Jika **semakin besar jumlah sampel** maka peluang sampel tersebut **representatif akan besar**, namun keuntungan sampling tidak didapat secara optimal
- Jika **semakin kecil sampel** kemungkinan **pola tidak didapatkan** atau walaupun didapat pola tsb salah
- Solusinya??
  - **Adaptive/ progressive sampling**
  - Dimulai dari sampel kecil sampai sejumlah sampel yang sudah memadai



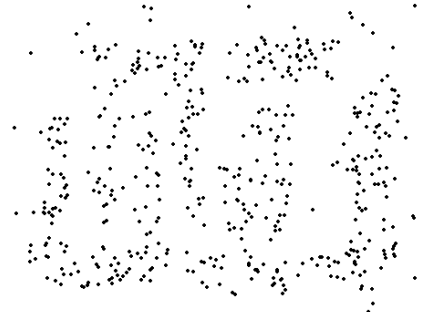
# Ukuran Sampel



8000 points



2000 Points



500 Points



## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



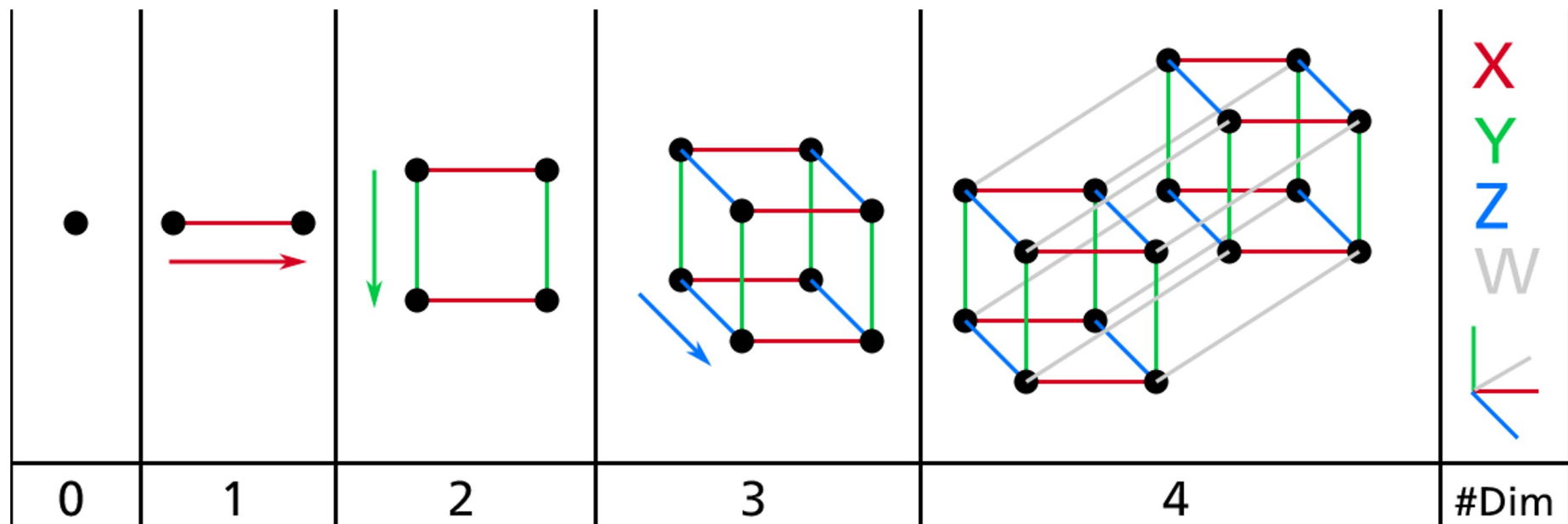
# Reduksi Dimensi

## ► Kenapa harus?

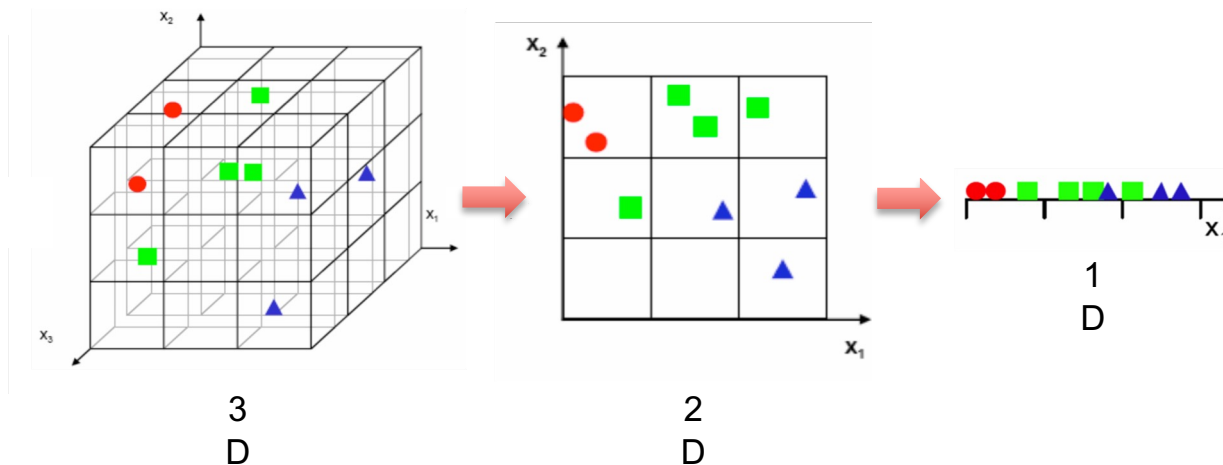
- Karena data set bisa saja memiliki jumlah features yang sangat banyak (contohnya: data dokumen dengan term sebagai vektor feature-nya)
- Menghindari Curse of Dimensionality (yakni fenomena di mana analisis data menjadi sangat sulit disebabkan penambahan dimensi data, data menjadi tersebar /sparse)
- Mengurangi penggunaan memori dan waktu yang dibutuhkan oleh algoritma data mining
- Memudahkan visualisasi data
- Membantu eliminasi data yang tidak relevan atau noise



# Dimensi Data



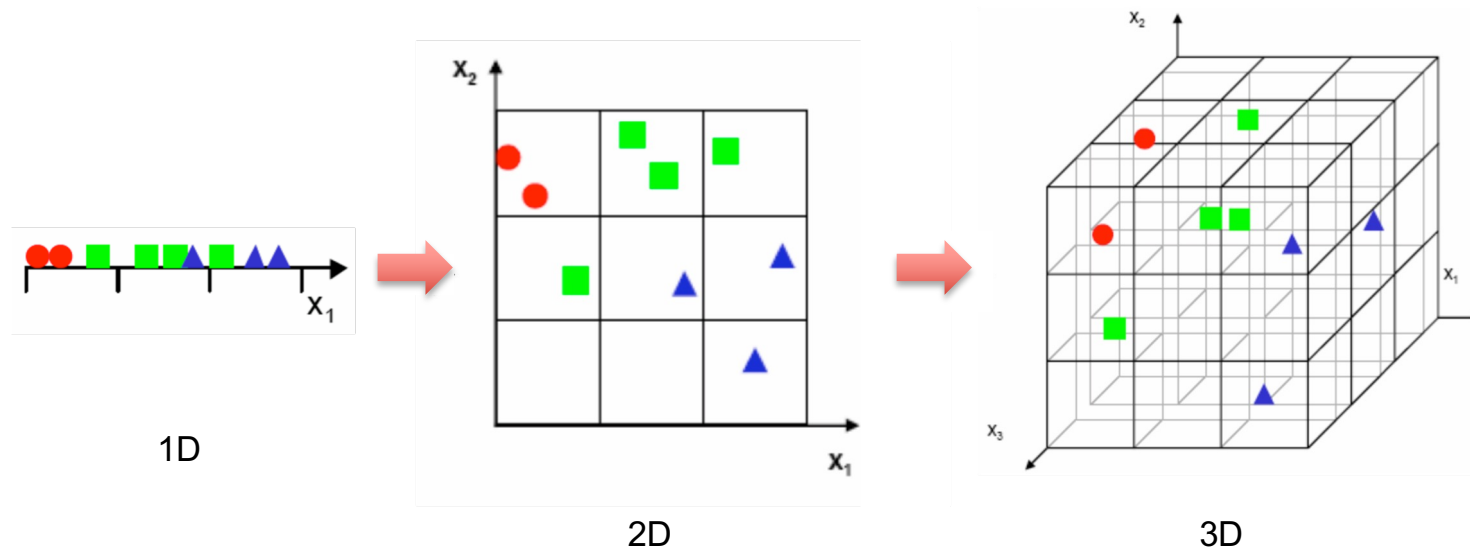
# Reduksi Dimensi



- Semakin **rendah dimensi** semakin **padat (dense)** struktur geometrinya
  - Proyeksi mengakibatkan hilangnya beberapa informasi mengenai data
  - Data tumpang tindih, campur-aduk, sulit dipisahkan



# Ekspansi Dimensi

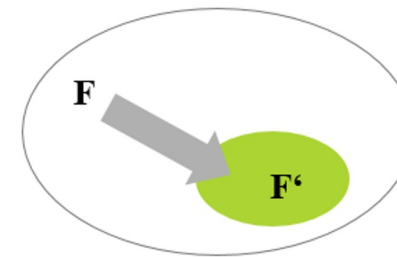


- ▶ Semakin **tinggi dimensi** data semakin **jarang (sparse)** struktur geometrinya
  - Membutuhkan memori dan komputasi yang besar
  - Geometric-structure data semakin buruk

# Reduksi Dimensi

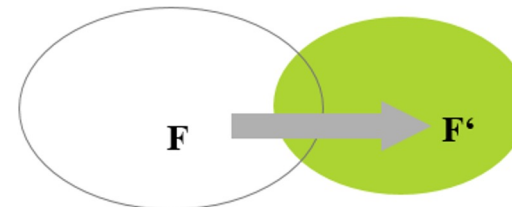
- Seleksi Fitur
- Ekstraksi Fitur

Feature Selection:



$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f. \text{selection}} \{f_{i_1}, \dots, f_{i_j}, \dots, f_{i_m}\} \quad \begin{matrix} i_j \in \{1, \dots, n\}; j = 1, \dots, m \\ i_a = i_b \Rightarrow a = b; a, b \in \{1, \dots, m\} \end{matrix}$$

Feature Extraction/Creation



$$\{f_1, \dots, f_i, \dots, f_n\} \xrightarrow{f. \text{extraction}} \{g_1(f_1, \dots, f_n), \dots, g_j(f_1, \dots, f_n), \dots, g_m(f_1, \dots, f_n)\}$$



## Reduksi Dimensi

Information Gain rendah, tidak membedakan kelas

Bisa direduksi? Ya, bisa.

Objek	Panjang	Lebar	Tinggi	Kelas
Objek 1	2,1	1,5	0,8	Meja
Objek 2	2,3	1,7	0,8	Meja
Objek 3	2,1	1,3	0,8	Kursi
Objek 4	1,6	1,5	0,8	Kursi
Objek 5	2,5	1,9	0,8	Meja



# Dimensionality Reduction

- ▶ As a counter-measure, many dimensionality reduction techniques have been proposed, and it has been shown that when done properly, the properties or structures of the objects can be well preserved even in the lower dimensions.
- ▶ Feature subset selection, Feature extraction, Reduction by Transformation, Discretization, Binarization, etc.



# Dimensionality Reduction

## ► Binarization

- Some algorithms require binary attributes
- How **NOT** to represent nominal variables:

Categorical value	Integer value	$x_1$	$x_2$	$x_3$
Toyota	0	0	0	0
Volkswagen	1	0	0	1
Chevrolet	2	0	1	0
Saab	3	0	1	1
Volvo	4	1	0	0



# Dimensionality Reduction

## ► Binarization

- Some algorithms require binary attributes
- How **better** to represent nominal variables

Categorical value	Integer value	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$
Toyota	0	1	0	0	0	0
Volkswagen	1	0	1	0	0	0
Chevrolet	2	0	0	1	0	0
Saab	3	0	0	0	1	0
Volvo	4	0	0	0	0	1

- Useful even when binary variables are not strictly required!

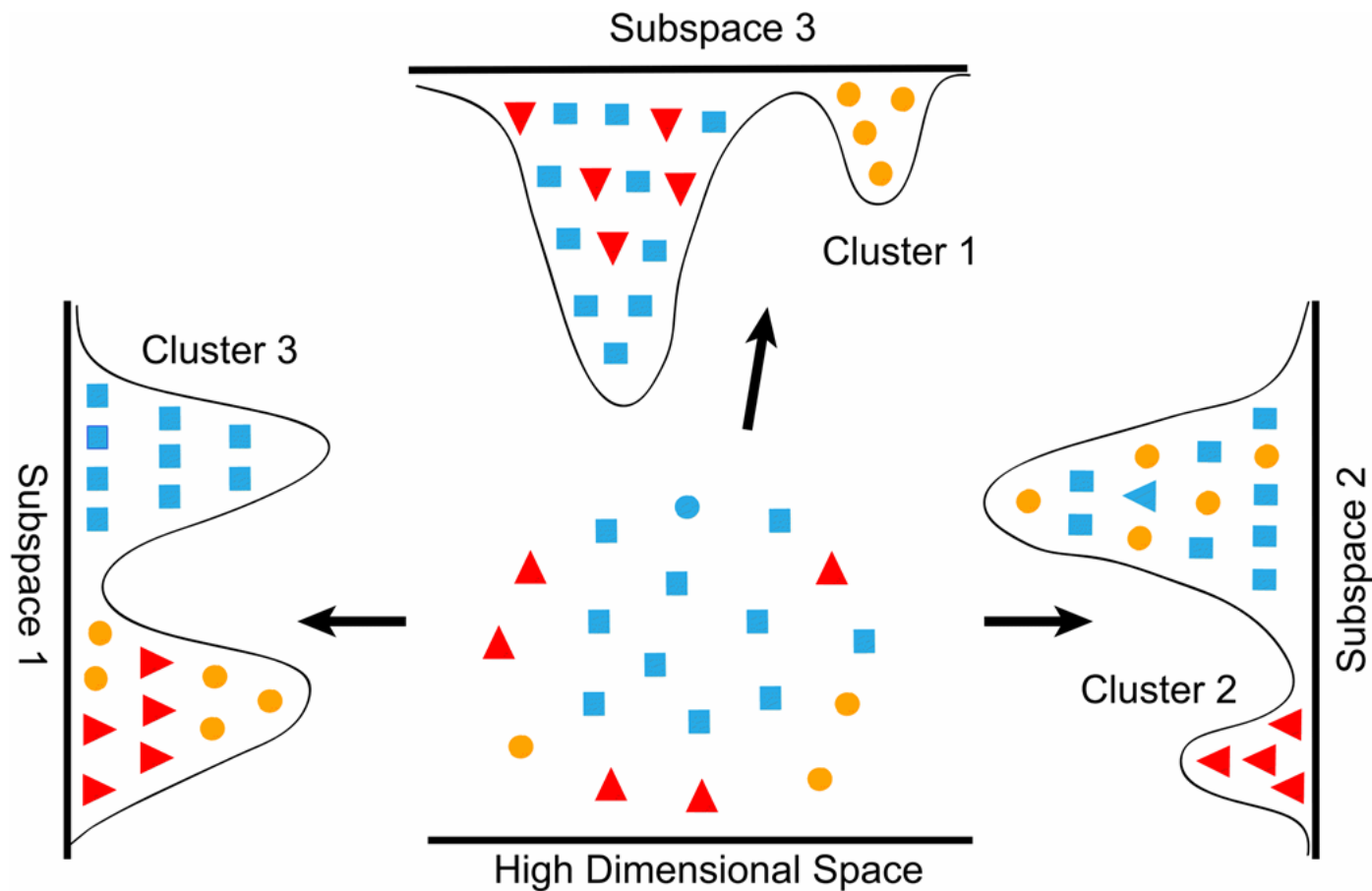


## Dimensionality Reduction

- ▶ Nevertheless, naively applying dimensionality reduction can lead to pathological results.
- ▶ While dimensionality reduction is an important tool in machine learning/data mining, we must always be aware that it can distort the data in misleading ways.
- ▶ Next is a two dimensional projection of an intrinsically three dimensional world.



# Dimensionality Reduction







## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



## Feature Subset Selection

- Untuk pengurangan dimensi data
- Redundant features
  - Duplikasi sebagian besar atau seluruh informasi dalam satu atau lebih atribut lainnya
  - Contoh: harga pembelian produk dan jumlah pajak penjualan yang dibayar
- Irrelevant features
  - Tidak mengandung informasi yang “berguna”
  - Contoh: ID mahasiswa seringkali tidak relevan dengan task untuk memprediksi IPK mahasiswa



# Teknik-Teknik Feature Subset Selection

- ▶ Pendekatan **Brute-force**
  - Mencari semua kemungkinan subsets feature sebagai input algoritma data mining
- ▶ Pendekatan **Embedded**
  - Feature selection dilakukan sebagai bagian dari algoritma data mining
- ▶ Pendekatan **Filter**
  - Feature dipilih sebelum algoritma data mining dijalankan
- ▶ Pendekatan **Wrapper**
  - Penggunaan algoritma data mining sebagai black box untuk menemukan best subset dari atribut



## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



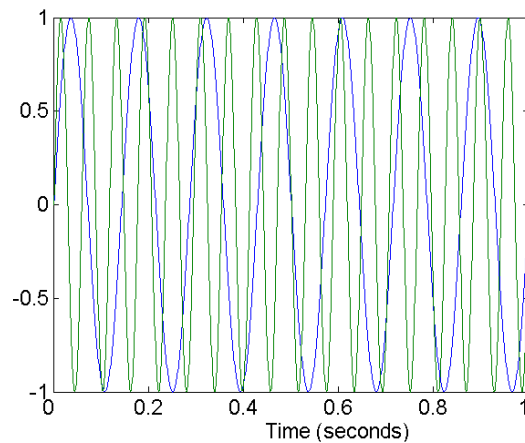
## Feature Creation

- ▶ Pembuatan atribut baru yang menggambarkan informasi penting pada dataset secara lebih efisien dibanding atribut asal
- ▶ Ada 3 metodologi umum :
  - Ekstraksi fitur
    - Domain-specific
  - Mapping Data ke New Space
  - Konstruksi Feature
    - Kombinasi features

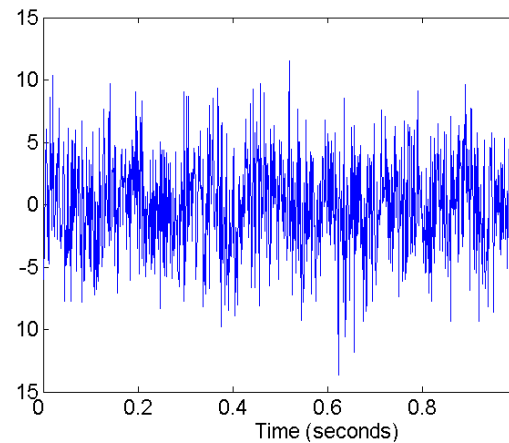


# Mapping Data ke New Space

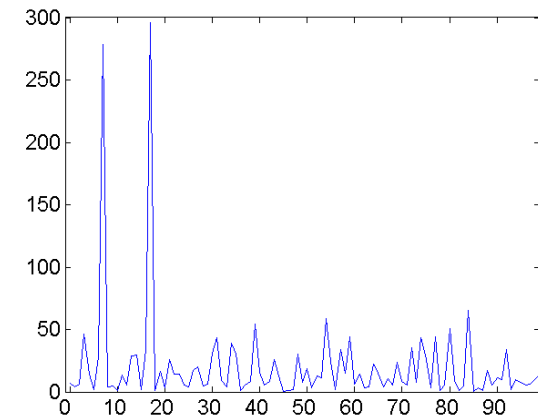
- ▶ Transformasi Fourier
- ▶ Transformasi Wavelet



Two Sine  
Waves



Two Sine Waves + Noise



Frequency



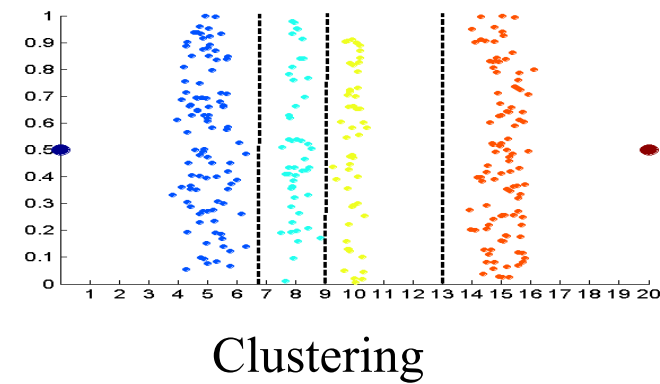
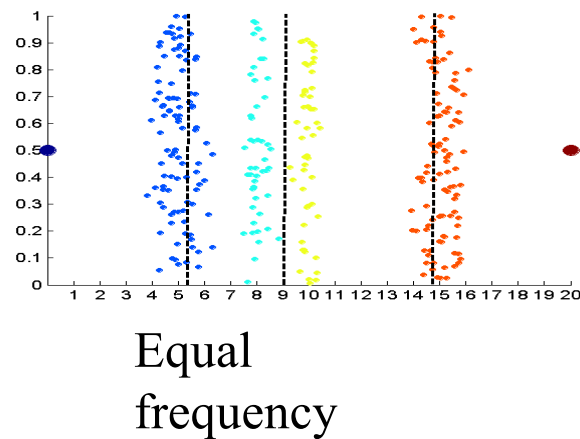
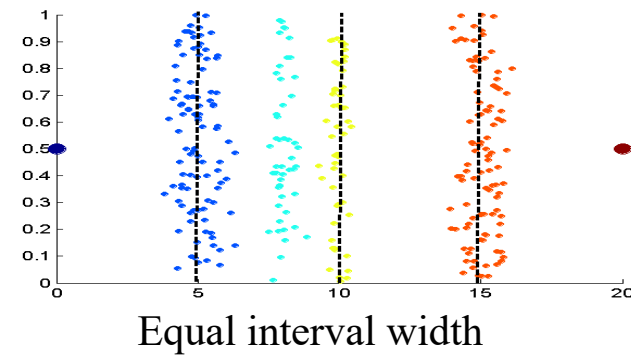
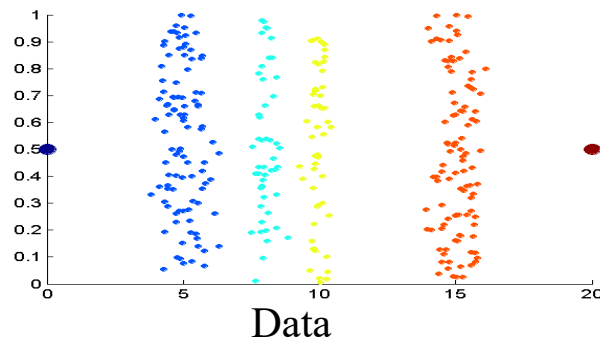
## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



# Diskretisasi

- ▶ Beberapa teknik tidak menggunakan label kelas

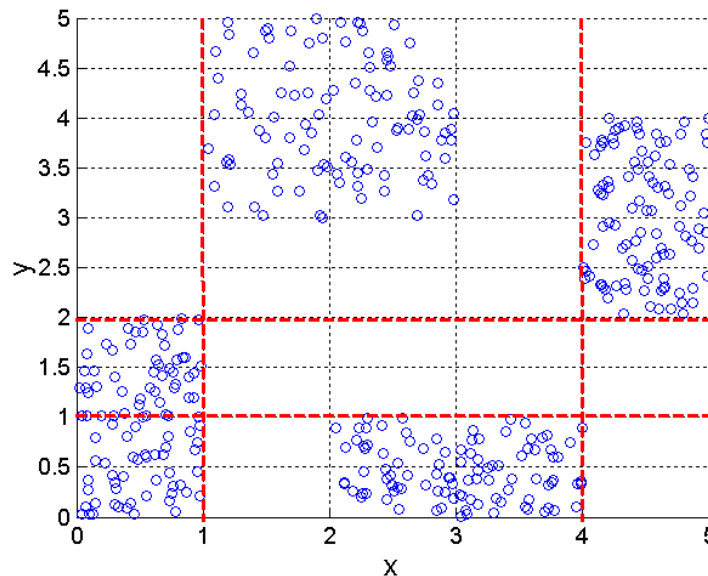




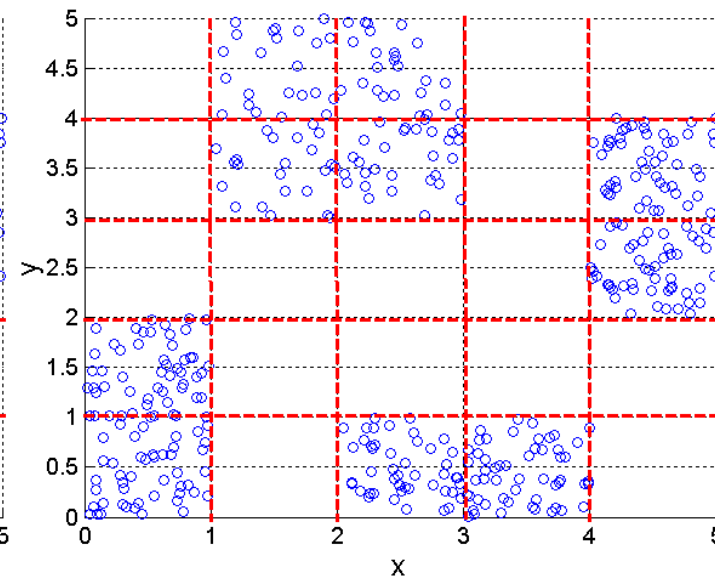


# Diskretisasi

- ▶ Beberapa teknik menggunakan label kelas
- ▶ Entropy based approach



*3 categories for both  $x$  and  $y$*



*5 categories for both  $x$  and  $y$*



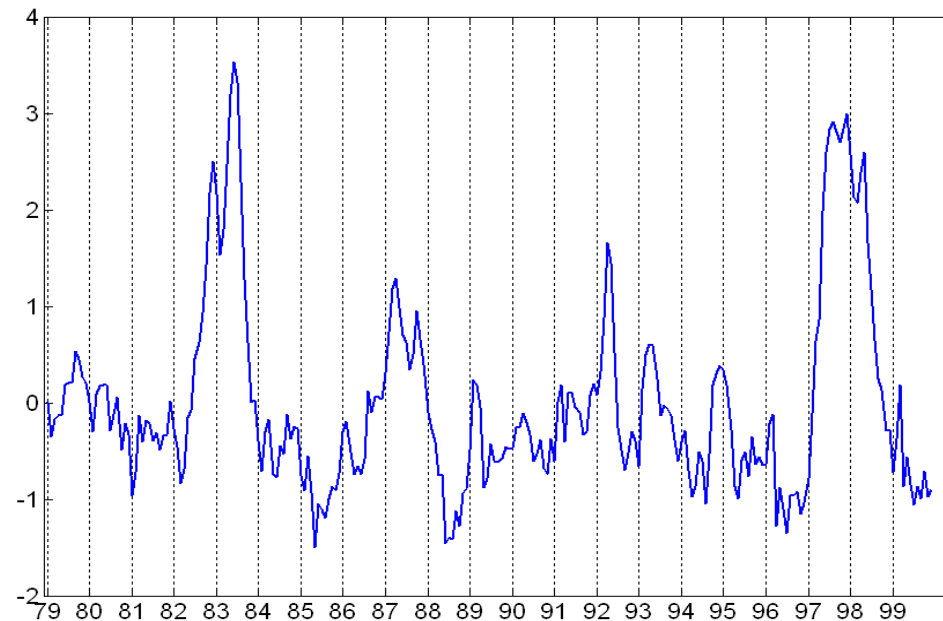
## Pokok Bahasan

- ▶ Apa Praproses Data
  - Agregasi
  - Sampling
  - Pengurangan dimensi
  - Feature subset selection
  - Feature creation
  - Diskretisasi dan Binerisasi
  - Transformasi atribut
- ▶ Similaritas & Disimilaritas
  - Euclidean distance
  - Minkowski distance
  - Mahalanobis Distance
  - Simple Matching
  - Jaccard Coefficients
  - Cosine
  - Tanimoto
  - Korelasi



# Transformasi Atribut

- Merupakan fungsi yang memetakan keseluruhan nilai atribut ke nilai baru dan setiap nilai lama dapat diidentifikasi dengan satu nilai baru
  - Fungsi sederhana:  $x_k$ ,  $\log(x)$ ,  $e^x$ ,  $|x|$
  - Standarisasi dan Normalisasi





## Pokok Bahasan

### ► Apa Praproses Data

- Agregasi
- Sampling
- Pengurangan dimensi
- Feature subset selection
- Feature creation
- Diskretisasi dan Binerisasi
- Transformasi atribut

### ► Similaritas & Disimilaritas

- Euclidean distance
- Minkowski distance
- Mahalanobis Distance
- Simple Matching
- Jaccard Coefficients
- Cosine
- Tanimoto
- Korelasi



# Similaritas dan Disimilaritas

- Similaritas
  - Pengukuran numerik untuk kemiripan dua objek
  - Semakin tinggi semakin mirip
  - Range antara  $[0,1]$
- Disimilaritas
  - Pengukuran numerik untuk perbedaan dua objek
  - Semakin tinggi semakin berbeda
  - Minimum dissimilaritas = 0
  - Upper limit varies
- Untuk ukuran similaritas & dissimilaritas bisa menggunakan jarak (distance)



# Similaritas/Disimilaritas untuk Atribut Sederhana

- ▶ Misalkan  $p$  dan  $q$  adalah nilai atribut untuk 2 objek data.

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$
Ordinal	$d = \frac{ p-q }{n-1}$ (values mapped to integers 0 to $n-1$ , where $n$ is the number of values)	$s = 1 - \frac{ p-q }{n-1}$
Interval or Ratio	$d =  p - q $	$s = -d, s = \frac{1}{1+d}$ or $s = 1 - \frac{d - \min\_d}{\max\_d - \min\_d}$

**Table 5.1.** Similarity and dissimilarity for simple attributes



# Teknik-Teknik Pengukuran Jarak

- ▶ Euclidean Distance

$$dist = \sqrt{\sum_{k=1}^n (p_k - q_k)^2}$$

- ▶ Minkowski distance

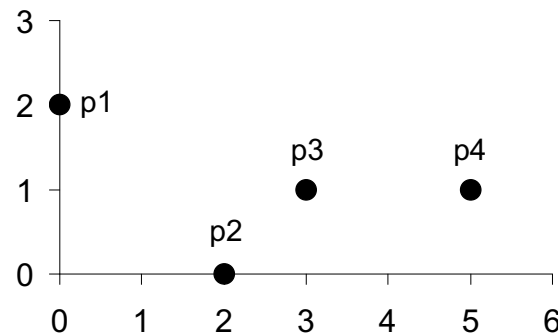
$$dist = \left( \sum_{k=1}^n |p_k - q_k|^r \right)^{\frac{1}{r}}$$

- ▶ Mahalanobis Distance

$$mahalanobis(p, q) = (p - q) \Sigma^{-1} (p - q)^T$$



## Contoh Perhitungan Euclidean Distance



$$dist = \sqrt{\sum_{k=1}^n (p_k - q_k)^2}$$

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix





## Pokok Bahasan

### ► Apa Praproses Data

- Agregasi
- Sampling
- Pengurangan dimensi
- Feature subset selection
- Feature creation
- Diskretisasi dan binerisasi
- Transformasi atribut
- Scaling

### ► Similaritas & Disimilaritas

- Euclidean distance
- Minkowski distance
- Mahalanobis Distance
- Simple Matching
- Jaccard Coefficients
- Cosine
- Tanimoto
- Korelasi



# Scaling Method

- ▶ Min-Max Normalization
- ▶ Mean Normalization
- ▶ Standardization (Z-score Normalization)
- ▶ Scaling to Unit Length



## Scaling: Min-Max Normalization

- ▶ The general formula to scale to range  $[a,b]$ .
- ▶ 
$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$
- ▶ If we want to scale to range  $[0,1]$ , then the formula becomes:
- ▶ 
$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$



## Scaling: Min-Max Normalization

- ▶ For example we have feature Height of 3 students = 170, 160, 180, and we want to rescale to [0,1]
- ▶ Then after-scale values are:
- ▶  $\frac{170-160}{180-160} = 0.5$ ,  $\frac{160-160}{180-160} = 0$ ,  $\frac{180-160}{180-160} = 1$
- ▶ 170, 160, 180  $\rightarrow$  0.5, 0, 1



## Scaling: Mean Normalization

- ▶ The formula:
- ▶  $x' = \frac{x - \text{average}(x)}{\max(x) - \min(x)}$
- ▶ From previous example: 170, 160, 180, the average is 170
- ▶ Then,  $\frac{170-170}{180-160} = 0$ ,  $\frac{160-170}{180-160} = -0.5$ ,  $\frac{180-170}{180-160} = 0.5$



## Scaling: Standardization (Z-score Normalization)

- ▶  $x' = \frac{x - \text{average}(x)}{\text{stddev}(x)}$
- ▶ This rescaling will make sure the feature will have zero-mean, and unit-variance.
- ▶ From previous example: 170, 160, 180, the average is 170, and the standard deviation is 8.16
- ▶ Then,  $\frac{170-170}{8.16} = 0$ ,  $\frac{160-170}{8.16} = -1.22$ ,  $\frac{180-170}{8.16} = 1.22$
- ▶ If we compute variance of 0, -1.22, 1.22, it will very close to 1.



## Scaling: Unit Length

- ▶ The formula:
- ▶  $x' = \frac{x}{\|x\|}$ , where  $\|x\|$  is the length of feature vector.

- ▶ Length of vector 170, 160, 180 is 294.79

- ▶ So the new vector is :

$$\frac{170}{294.79} = 0.58, \quad \frac{160}{294.79} = 0.54, \quad \frac{180}{294.79} = 0.61$$

- ▶ If we compute the length of the new vector 0.58, 0.54, 0.61 it will be around 1.

## Scaling Effect

- For example we scale Feature\_2 with min-max normalization, the new range is [0-1].

	F1	F2
#1	0.1	1000
#2	0.1	800
#3	0.9	800



	F1	F2
#1	0.1	1
#2	0.1	0.8
#3	0.9	0.8

- Now, Euclidean Distance from Record-1 to Record-2 is 0.2
- While, from Record-2 to Record-3 is 0.8
- Now, Record-2 is closer to Record-1 than to Record-3, totally different from pre-scaling.





## Binning

- ▶ Binning is grouping values into new category.
- ▶ For example, instead of using exact value of salary, we can bin the values into 3 categories:
  - $< \text{Rp. 5 million}$  → Low
  - $\text{Rp. 5 million} - \text{Rp. 10 million}$  → Mid
  - $> \text{Rp. 10 million}$  → High
- ▶ Binning can help our model more robust to the slight changes on the values, but on the other hand our model also loose some precise information.



# Data Preprocessing for Text Data



# Data Cleaning And Text Preprocessing

- ▶ Preprocessing the raw text involves the following:
  - I. Removing URL
  - II. Removing all irrelevant characters (Numbers and Punctuation)
  - III. Convert all characters into lowercase
  - IV. Tokenization
  - V. Removing Stopwords
  - VI. Stemming and Lemmatization
  - VII. Remove the words having length  $\leq 2$
  - VIII. Convert the list of tokens into back to the string



## Example

```
dataset['Reviews'].iloc[2]
```

"Hello e-v-e-r-y-o-n-e!!!@@@!!!!!! 😊 @DONT BUY THIS PHONE at all-first of al  
l that say the phone in new , i took it to the lab after 6 month the phone is d  
ead dead ,you can save itthay open the phone in the lab and say!!!!the phone is  
renew ,and its cheepe components.I payed 400\$ for only 6 month ,now i need to  
buy new one this LG G4 is dead .not nice 'people say to me dont buy from [http  
s://www.amazon.com/gp/aw/d/B01GYUDMFY/ref=ya\\_aw\\_od\\_pi?ie=UTF8&psc=1](http://www.amazon.com/gp/aw/d/B01GYUDMFY/ref=ya_aw_od_pi?ie=UTF8&psc=1) at al!!!"

Example



# I. Removing URL

- Remove by using below line of code:

```
# Importing the library 're' to remove non url, Numbers and punctuation
import re

def clean_url(review_text):
    return re.sub(r'http\S+', '', review_text)

dataset['CleanReview'] = dataset['Reviews'].apply(clean_url)
```

- Result: The url (highlighted by green color) has been removed.

"Hello e-v-e-r-y-o-n-e!!!!@!!!!!! ?? @DONT BUY THIS PHONE at all-first o  
f all that says the phone in new , i took it to the lab after 6 month the  
phone is dead dead ,you can save it,they open the phone in the lab and say  
s!!!! the phone is renew ,and its cheapest components.I payed 400\$ for on  
ly 6 month ,now i need to buy new one this LG G4 is dead .not a best thing  
'people are saying to me dont buy from [https://www.amazon.com/gp/aw/d/BU0000FY/REF\\_ya\\_av\\_sd\\_pi/1e-UTR8spac-1](https://www.amazon.com/gp/aw/d/BU0000FY/REF_ya_av_sd_pi/1e-UTR8spac-1) at all, it's troubling!!!"

Removing url

"Hello e-v-e-r-y-o-n-e!!!!@!!!!!! ?? @DONT BUY THIS PHONE at all-first o  
f all that says the phone in new , i took it to the lab after 6 month the  
phone is dead dead ,you can save it,they open the phone in the lab and say  
s!!!! the phone is renew ,and its cheapest components.I payed 400\$ for on  
ly 6 month ,now i need to buy new one this LG G4 is dead .not a best thing  
'people are saying to me dont buy from at all, it's troubling!!!"



## II. Removing All Irrelevant Characters (Numbers and Punctuation)

- Remove numbers if they are not relevant to your analyses (0–9). And punctuation also will be remove. Punctuation is basically the set of symbols [!"#\$%&'()\*+,-./:;<=>?@[\\]^\_`{|}~]:

```
def clean_non_alphanumeric(review_text):
    return re.sub('[^a-zA-Z]', ' ', review_text)

dataset['CleanReview'] = dataset['CleanReview'].apply(clean_non_alphanumeric)
```

- Result: All numeric and punctuation has been replaced with space ' '.

"Hello e-v-e-r-y-o-n-e!!! @@@@!!!!!! 22 @DONT BUY THIS PHONE at all, first o  
f all that says the phone in new, i took it to the lab after 6 month the  
phone is dead dead, you can save it, they open the phone in the lab and say  
s!!!! the phone is renew, and its cheapest components. I paid 400\$ for on  
ly 6 month, now i need to buy new one this LG G4 is dead .not a best thing  
people are saying to me dont buy from at all, its troubling!!!"

### Removing irrelevant characters

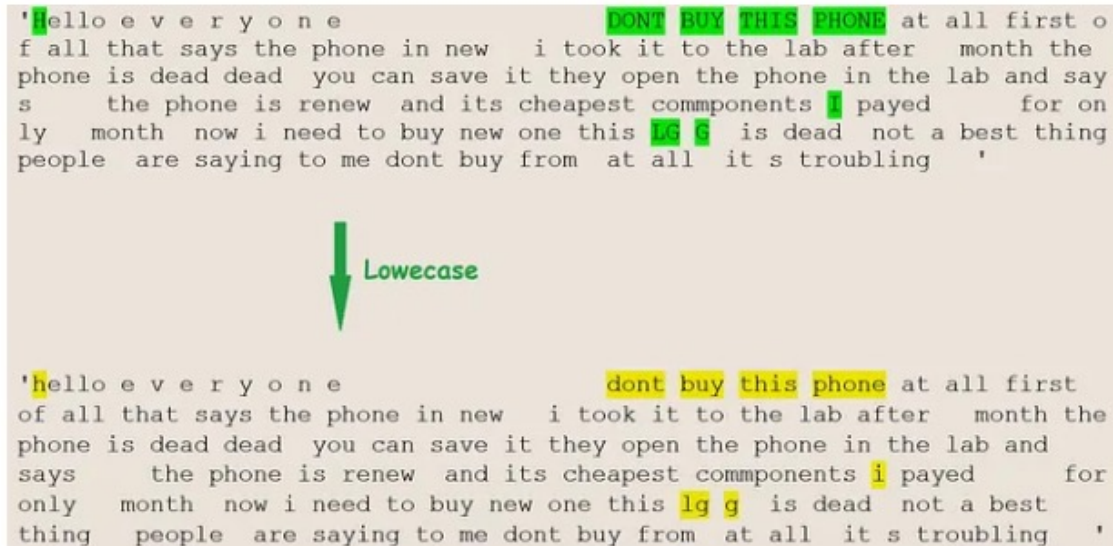
'Hello everyone [REDACTED] DONT BUY THIS PHONE at all first of all that says the phone is new i took it to the lab after month the phone is dead dead you can save it they open the phone in the lab and say s the phone is renew and its cheapest components I paid for on ly month now i need to buy new one this LG G is dead not a best thing people are saying to me dont buy from at all its troubling '

### III. Convert All Characters into Lowercase

- All words changes into lower case or uppercase to avoid the duplication. Because "Phone" and "phone" will be considered as 2 separate words if this step is not done.

```
def clean_lowercase(review_text):  
    return str(review_text).lower()  
dataset['CleanReview'] = dataset['CleanReview'].apply(clean_lowercase)
```

- Result: All upper case that are highlighted by green color has been replaced with lower case (highlighted by yellow color).



'Hello everyone DONT BUY THIS PHONE at all first o  
f all that says the phone in new i took it to the lab after month the  
phone is dead dead you can save it they open the phone in the lab and say  
s the phone is renew and its cheapest components i payed for on  
ly month now i need to buy new one this LG G is dead not a best thing  
people are saying to me dont buy from at all its troubling '

↓ Lowecase

'hello everyone dont buy this phone at all first  
of all that says the phone in new i took it to the lab after month the  
phone is dead dead you can save it they open the phone in the lab and  
says the phone is renew and its cheapest components i payed for  
only month now i need to buy new one this lg g is dead not a best  
thing people are saying to me dont buy from at all its troubling '



## IV. Tokenization

- Tokenization is the process of splitting the given text into smaller pieces called tokens. Words, numbers, punctuation marks, and others can be considered as tokens. We will use Natural language tool kit (nltk) library for tokenization.

```
# Using nltk (Natural Language tool kit) Library for tokenization
import nltk
from nltk.tokenize import word_tokenize

def clean_tokenization(review_text):
    return word_tokenize(review_text)

dataset['CleanReview'] = dataset['CleanReview'].apply(clean_tokenization)
```



## IV. Tokenization

- Result: The string has been changed into tokens, that has been stored in the form of 'list of string'.

```
'hello e v e r y o n e          dont buy this phone at all first o
f all that says the phone in new  i took it to the lab after  month the
phone is dead dead  you can save it they open the phone in the lab and say
s      the phone is renew  and its cheapest components i payed      for on
ly  month now i need to buy new one this lg g  is dead  not a best thing
people  are saying to me dont buy from  at all  it s troubling  '
```



```
['hello', 'e', 'v', 'e', 'r', 'y', 'o', 'n', 'e', 'dont', 'buy', 'this',
'phone', 'at', 'all', 'first', 'of', 'all', 'that', 'says', 'the',
'phone', 'in', 'new', 'i', 'took', 'it', 'to', 'the', 'lab', 'after',
'month', 'the', 'phone', 'is', 'dead', 'dead', 'you', 'can', 'save', 'it',
'they', 'open', 'the', 'phone', 'in', 'the', 'lab', 'and', 'says', 'the',
'phone', 'is', 'renew', 'and', 'its', 'cheapest', 'components', 'i',
'payed', 'for', 'only', 'month', 'now', 'i', 'need', 'to', 'buy', 'new',
'one', 'this', 'lg', 'g', 'is', 'dead', 'not', 'a', 'best', 'thing',
'people', 'are', 'saying', 'to', 'me', 'dont', 'buy', 'from', 'at', 'all',
'it', 's', 'troubling']
```

dataset.head(5)

	Rating	Reviews	Label	CleanReview
0	5	I feel so LUCKY to have found this used (phone...	Positive	[i, feel, so, lucky, to, have, found, this, us...
1	4	nice phone, nice up grade from my pantach revu...	Positive	[nice, phone, nice, up, grade, from, my, panta...
2	1	Hello e-v-e-r-y-o-n-e!!!! @@@@!!!!!! 😞 @DONT BUY...	Negative	[hello, e, v, e, r, y, o, n, e, dont, buy, thi...
3	5	Very pleased	Positive	[very, pleased]
4	4	It works good but it goes slow sometimes but i...	Positive	[it, works, good, but, it, goes, slow, sometim...



## V. Removing Stopwords

- “Stopwords” are the most common words in a language like “the”, “a”, “me”, “is”, “to”, “all”,. These words do not carry important meaning and are usually removed from texts. It is possible to remove stopwords using Natural Language Toolkit (nltk).

```
# importing the libraries required to remove the stopwords and punctuation
from nltk.corpus import stopwords
# Let's look at the stopwords in english
stopwords.words('english')
```

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
 'his',
```

- these are the stopwords which need to remove, remove the stopwords.



## V. Removing Stopwords

```
stop_words = set(stopwords.words('english'))  
def clean_stopwords(token):  
    return [item for item in token if item not in stop_words]  
  
dataset['CleanReview'] = dataset['CleanReview'].apply(clean_stopwords)
```

- Result: All highlighted tokens has been removed from the corpus after apply the function clean\_stopwords().

['hello', 'e', 'v', 'e', 'r', 'y', 'to', 'n', 'e', 'dont', 'buy', 'this',  
'phone', 'at', 'all', 'first', 'of', 'all', 'that', 'says', 'the',  
'phone', 'in', 'new', 'i', 'took', 'it', 'to', 'the', 'lab', 'after',  
'month', 'the', 'phone', 'is', 'dead', 'dead', 'you', 'can', 'save', 'it',  
'they', 'open', 'the', 'phone', 'in', 'the', 'lab', 'and', 'says', 'the',  
'phone', 'is', 'renew', 'and', 'its', 'cheapest', 'components', 'i',  
'payed', 'for', 'only', 'month', 'now', 'i', 'need', 'to', 'buy', 'new',  
'one', 'this', 'lg', 'g', 'is', 'dead', 'not', 'a', 'best', 'thing',  
'people', 'are', 'saying', 'to', 'me', 'dont', 'buy', 'from', 'at', 'all',  
'it', 's', 'troubling']



Removing Stopwords

['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',  
'says', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead',  
'save', 'open', 'phone', 'lab', 'says', 'phone', 'renew', 'cheapest',  
'components', 'payed', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',  
'dead', 'best', 'thing', 'people', 'saying', 'dont', 'buy', 'troubling']



## VI. Stemming and Lemmatization

- ▶ Stemming usually refers to a crude process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational units (the obtained element is known as the stem).
- ▶ On the other hand, lemmatization consists in doing things properly with the use of a vocabulary and morphological analysis of words, to return the base or dictionary form of a word, which is known as the lemma.

I saw an amazing thing  $\xrightarrow{\text{stem}}$  I s an amazing thing

I saw an amazing thing  $\xrightarrow{\text{lemma}}$  I see an amazing thing





# Stemming

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()
def clean_stem(token):
    return [stemmer.stem(i) for i in token]

dataset['CleanReview'] = dataset['CleanReview'].apply(clean_stem)
```

- Result: As observe the output, there is some words has been stem like 'commponents' to 'commpon', 'says' to 'say', 'people' to 'peopl' and 'troubling' to 'troubl'.

Before stemming:

['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first', 'says', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead', 'save', 'open', 'phone', 'lab', 'says', 'phone', 'renew', 'cheapest', 'components', 'payed', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g', 'dead', 'best', 'thing', 'people', 'saying', 'dont', 'buy', 'troubling']

↓ Stemming

After stemming:

['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first', 'say', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead', 'save', 'open', 'phone', 'lab', 'say', 'phone', 'renew', 'cheapest', 'commpon', 'pay', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g', 'dead', 'best', 'thing', 'peopl', 'say', 'dont', 'buy', 'troubl']

# Lemmatization

```
from nltk.stem import WordNetLemmatizer
lemma=WordNetLemmatizer()

def clean_lemmatization(token):
    return [lemma.lemmatize(word=w,pos='v') for w in token]

dataset['CleanReview'] = dataset['CleanReview'].apply(clean_lemmatization)
```

- Result: Now here we can see it finds the root word, like 'troubling' to 'trouble', 'took' to 'take' and 'payed' to 'pay'. So, As opposed to stemming, lemmatization does not simply chop off inflections. Instead it uses lexical knowledge bases to get the correct base forms of words.

['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',  
'says', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead',  
'save', 'open', 'phone', 'lab', 'says', 'phone', 'renew', 'cheapest',  
'components', 'payed', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',  
'dead', 'best', 'thing', 'people', 'saying', 'dont', 'buy', 'troubling']

↓ Lemmatization

['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',  
'say', 'phone', 'new', 'take', 'lab', 'month', 'phone', 'dead', 'dead',  
'save', 'open', 'phone', 'lab', 'say', 'phone', 'renew', 'cheapest',  
'components', 'pay', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',  
'dead', 'best', 'thing', 'people', 'say', 'dont', 'buy', 'trouble']

## VII. Remove the Words Having Length $\leq 2$

- Removing the words which have very short length.

```
def Clean_length(token):  
    return [i for i in token if len(i) > 2]  
  
dataset['CleanReview'] = dataset['CleanReview'].apply(Clean_length)
```

- Result: Removing the words those have length less than or equal to 2.

```
['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',  
'say', 'phone', 'new', 'take', 'lab', 'month', 'phone', 'dead', 'dead',  
'save', 'open', 'phone', 'lab', 'say', 'phone', 'renew', 'cheapest',  
'components', 'pay', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',  
'dead', 'best', 'thing', 'people', 'say', 'dont', 'buy', 'trouble']
```

↓ Removing word having less length

```
['hello', 'dont', 'buy', 'phone', 'first', 'say', 'phone', 'new', 'take',  
'lab', 'month', 'phone', 'dead', 'dead', 'save', 'open', 'phone', 'lab',  
'say', 'phone', 'renew', 'cheapest', 'components', 'pay', 'month', 'need',  
'buy', 'new', 'one', 'dead', 'best', 'thing', 'people', 'say', 'dont',  
'buy', 'trouble']
```



## VII. Convert the List of Tokens into Back to the String

```
def convert_to_string(listReview):  
    return ' '.join(listReview)  
  
dataset['CleanReview'] = dataset['CleanReview'].apply(convert_to_string)
```

- Result: After performing the above code, getting a string of input list.

```
['hello', 'dont', 'buy', 'phone', 'first', 'say', 'phone', 'new', 'take',  
'lab', 'month', 'phone', 'dead', 'dead', 'save', 'open', 'phone', 'lab',  
'say', 'phone', 'renew', 'cheapest', 'components', 'pay', 'month', 'need',  
'buy', 'new', 'one', 'dead', 'best', 'thing', 'people', 'say', 'dont',  
'buy', 'trouble']
```



Converting into string

```
'hello dont buy phone first say phone new take lab month phone dead dead s  
ave open phone lab say phone renew cheapest components pay month need buy  
new one dead best thing people say dont buy trouble'
```





## VII. Convert the List of Tokens into Back to the String

- Dataset after performing text preprocessing.

```
dataset.head()
```

	Rating	Reviews	Label	CleanReview
0	5	I feel so LUCKY to have found this used (phone...	Positive	feel lucky find use phone use hard phone line ...
1	4	nice phone, nice up grade from my pantach revu...	Positive	nice phone nice grade pantach revue clean set ...
2	1	Hello e-v-e-r-y-o-n-e!!!@@@@!!!!!! ?? @DONT BU...	Negative	hello dont buy phone first say phone new take ...
3	5	Very pleased	Positive	please
4	4	It works good but it goes slow sometimes but i...	Positive	work good slow sometimes good phone love



# Single Function for Text Preprocessing

```
import re
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

stop_words = set(stopwords.words('english'))
lemma=WordNetLemmatizer()

def clean_Review(review_text):
    review_text = re.sub(r'http\S+', '', review_text)           #Removing the url
    review_text = re.sub('[^a-zA-Z]', '', review_text)          #Removing Numbers and Punctuation
    review_text = str(review_text).lower()                       #Convert all characters into lowercase
    review_text = word_tokenize(review_text)                     #Tokenization
    review_text = [item for item in review_text if item not in stop_words] #Removing Stop Words
    review_text = [lemma.lemmatize(word=w,pos='v') for w in review_text] #Lemmatization
    review_text = [i for i in review_text if len(i) > 2]        #Remove the words having length <= 2
    review_text = ' '.join(review_text)                         #Converting list to string
    return review_text

dataset['CleanReview'] = dataset['Reviews'].apply(clean_Review)
```



# Data Preprocessing for Image Data



# Type of Image Preparation

- ▶ Crop the image



- ▶ Resize the image without the same ratio







## Type of Image Preparation

- ▶ Resize the image with the same aspect ratio



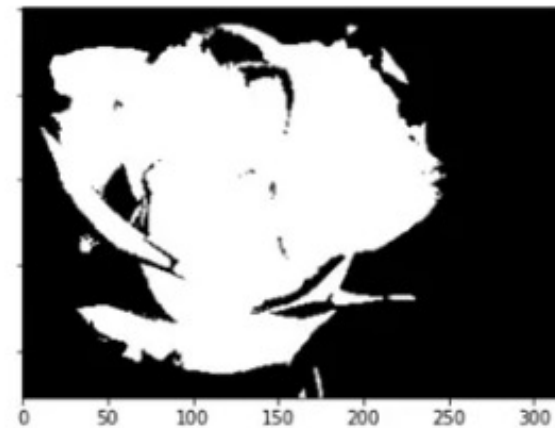
- ▶ Randomly crop





# Type of Image Preparation

- Image Scaling
- Dimensionality Reduction
- Morphological Transformations
  - Thresholding





## Type of Image Preparation

- Erosion, Dilation, Opening & Closing
  - Erosion: Shrinks bright regions and enlarges dark regions.
  - Dilation: Shrinks dark regions and enlarges the bright regions.
  - Opening: Erosion followed by dilation. Opening can remove small bright spots (i.e. “salt”) and connect small dark cracks. This tends to “open” up (dark) gaps between (bright) features.
  - Closing: Dilation followed by erosion. Closing can remove small dark spots (i.e. “pepper”) and connect small bright cracks. This tends to “close” up (dark) gaps between (bright) features.



# Data Preprocessing for Audio Data



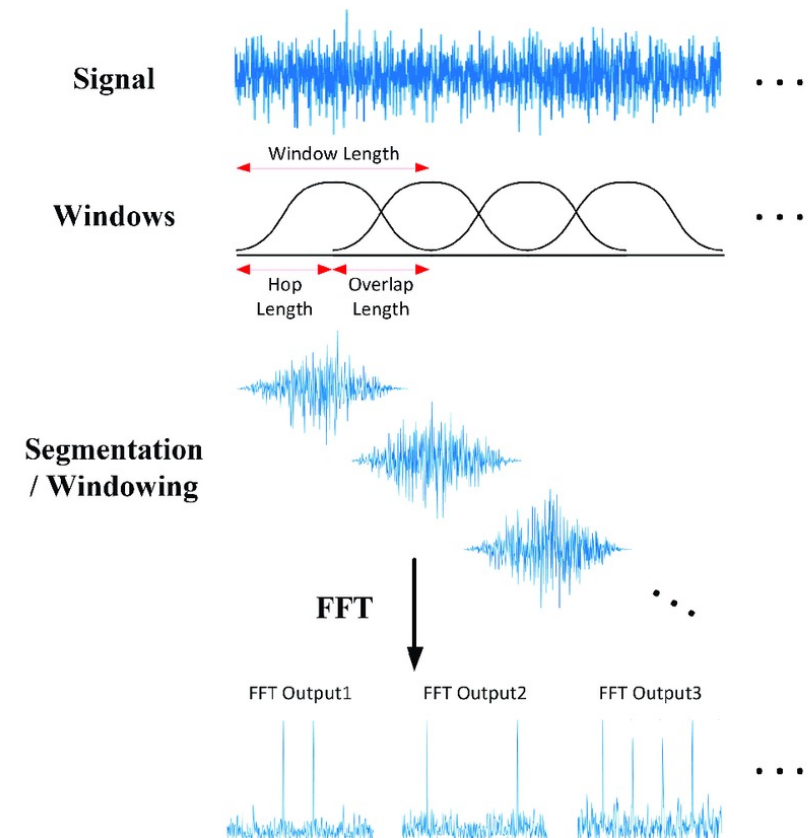


# Data Preprocessing for Audio Data

- ▶ Common audio features for preprocessing:
  - Short-time Fourier Transform (STFT)
  - Mel-Frequency Cepstral Coefficients (MFCCs)
  - Zero-crossing rate
- ▶ Noise Reduction and Enhancement
  - Filtering methods (e.g., low-pass, high-pass)
  - Spectral subtraction
  - Noise cancellation algorithms

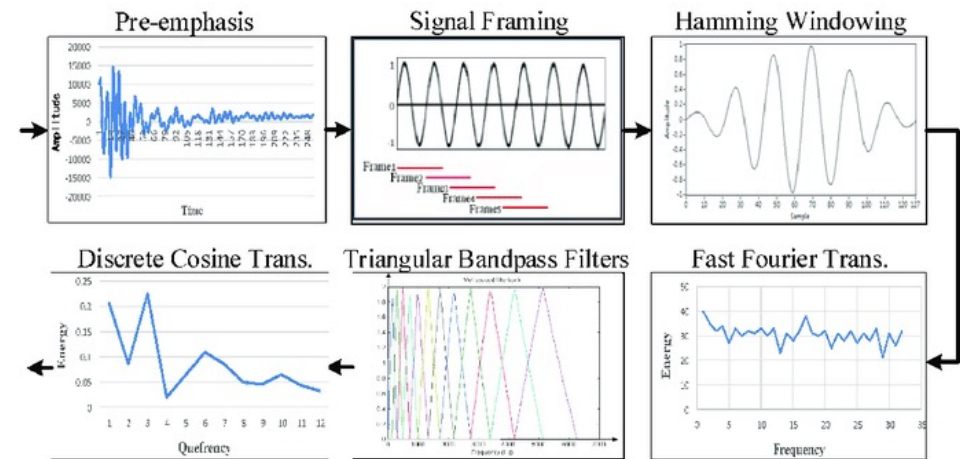
# Audio Features for Preprocessing

- ▶ Short-time Fourier Transform (STFT)
  - A mathematical technique used in signal processing to analyze and represent the time-varying frequency content of a signal, such as an audio waveform.
  - It's a powerful tool for extracting information about how the frequency components of a signal change over time.



# Audio Features for Preprocessing

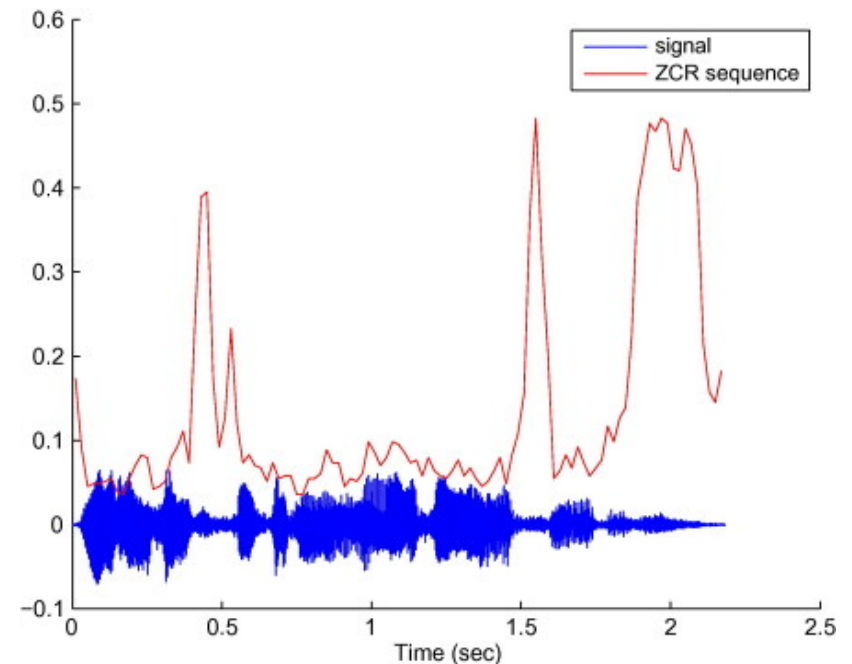
- ▶ Mel-Frequency Cepstral Coefficients (MFCCs)
  - A widely used feature extraction technique in audio and speech processing.
  - They are especially popular for tasks like speech recognition and audio classification.
  - MFCCs are derived from the Mel-frequency scale and the cepstral analysis of the power spectrum of an audio signal.



# Audio Features for Preprocessing

## ► Zero-crossing Rate

- A simple yet informative audio feature used in various signal processing and audio analysis tasks.
- It measures how often the signal crosses the zero amplitude threshold within a given time frame.





# Noise Reduction

## ► Filtering Methods

- Techniques used to remove or reduce unwanted noise from audio signals or other types of data.
- These methods involve applying filters that emphasize or attenuate specific frequency components in the signal to achieve noise reduction.
- Low-Pass Filtering
  - Allow low-frequency components (e.g., speech or desired signal) to pass while attenuating high-frequency components (e.g., noise).
- High-Pass Filtering:
  - They allow high-frequency components to pass while attenuating low-frequency components.



# Noise Reduction

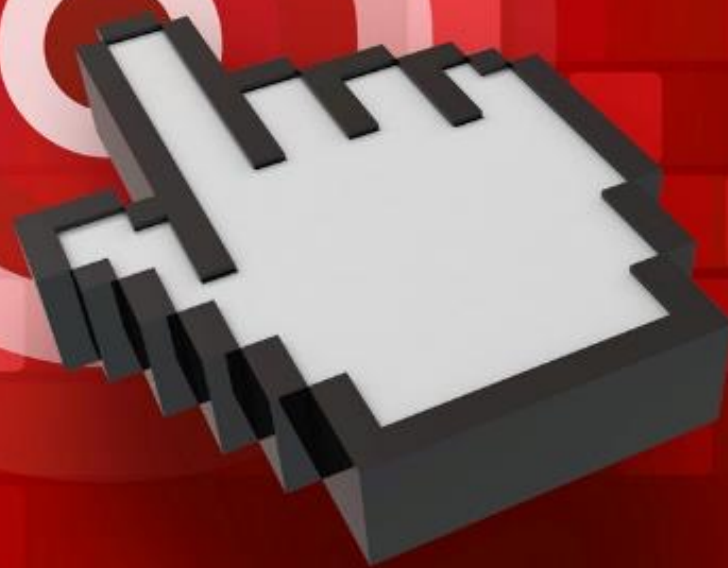
## ▶ Noise Cancellation Algorithms

- Known as active noise cancellation (ANC), is a technology used to reduce or eliminate unwanted background noise from an audio signal, such as the noise from a busy street or the hum of an airplane engine.
- Unlike passive noise reduction methods (e.g., insulation or soundproofing), noise cancellation actively generates anti-noise to counteract the incoming noise





Fakultas Informatika  
School of Computing  
Telkom University



*THANK YOU*