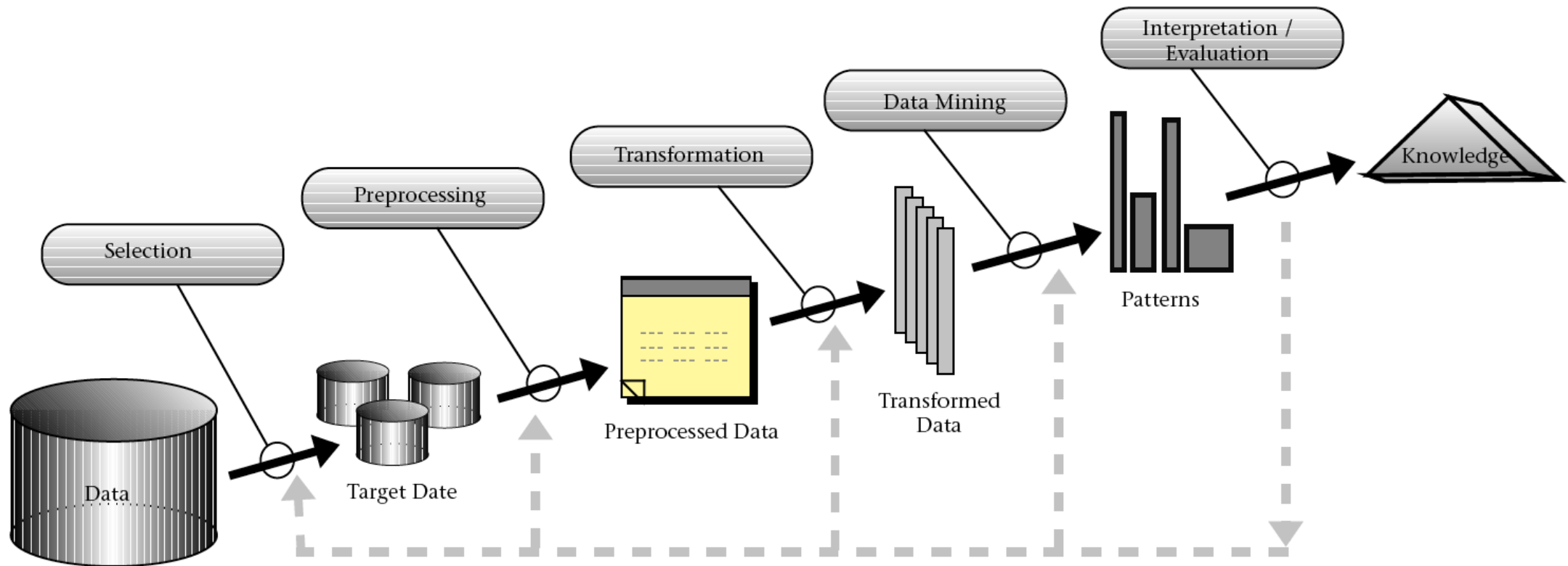
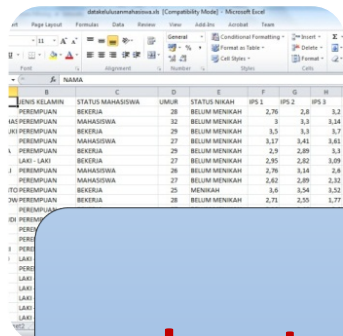


KDD



Tahapan Utama Data Mining

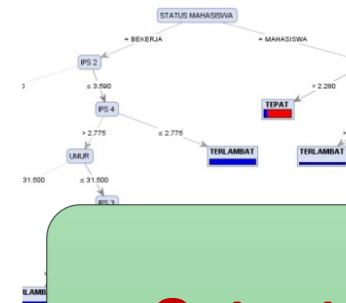


NAMA	STATUS MAHASISWA	UMUR	STATUS NIKAH	IPS 1	IPS 2	IPS 3
BENI KILAMIN	MAHASISWA	28	BELUM MENIKAH	2,76	2,8	3,2
PEREMPUAN	MAHASISWA	32	BELUM MENIKAH	3	3,3	3,14
MA PEREMPUAN	MAHASISWA	29	BELUM MENIKAH	3,3	3,3	3,7
PEREMPUAN	MAHASISWA	27	BELUM MENIKAH	3,17	3,41	3,81
PEREMPUAN	MAHASISWA	29	BELUM MENIKAH	2,9	2,89	3,3
LAJI-LAJI	MAHASISWA	27	BELUM MENIKAH	2,95	3,02	3,09
PEREMPUAN	MAHASISWA	26	BELUM MENIKAH	2,76	3,14	2,8
PEREMPUAN	MAHASISWA	27	BELUM MENIKAH	2,62	2,89	3,32
TO PEREMPUAN	MAHASISWA	25	MAHAJAWA	3,6	3,54	3,52
MA PEREMPUAN	MAHASISWA	28	BELUM MENIKAH	2,71	3,55	3,77

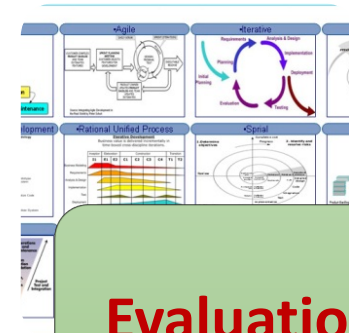
Input
(Data)

$$\forall x \exists y \text{ such that } |x - y| \leq \epsilon \iff \|f(x) - f(y)\|$$
$$\int_a^b f(x) dx = \lim_{n \rightarrow \infty} \frac{b-a}{n} \sum_{k=1}^n f\left(a + \frac{b-a}{n} \cdot k\right)$$
$$r(t) = m_2 r^2 \sin(\phi) \left[t = \frac{r^2}{4f} + r \left(\cos(\omega t) + \frac{r}{4f} \cos(2\omega t) \right) \right]$$
$$R_1 = \left(-\zeta + \sqrt{\zeta^2 - 1} \right) \exp t \quad \left(-\zeta - \sqrt{\zeta^2 - 1} \right) \exp t$$

Metode
(Algoritma Data Mining)



Output
(Pola/Model/
Knowledge)



Evaluation
(Akurasi, AUC,
RMSE, etc)

DATASET

Jenis dataset (**Private** dan **Public**)

- **Private Dataset**: data set dapat diambil dari organisasi yang kita jadikan obyek penelitian
 - Bank, Rumah Sakit, Industri, Pabrik, Perusahaan Jasa, etc
- **Public Dataset**: data set dapat diambil dari repositori publik yang disepakati oleh para peneliti data mining
 - **UCI Repository** (<http://www.ics.uci.edu/~mlearn/MLRepository.html>)

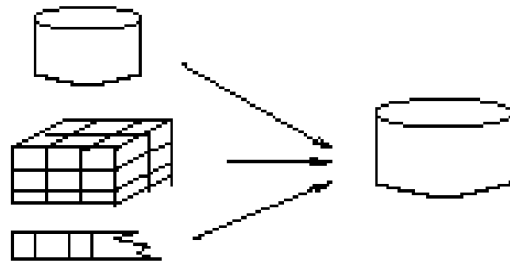
Tahap penelitian / eksperimen bisa dilakukan dengan menguji metode yang dikembangkan oleh peneliti dengan public dataset, penelitian dapat bersifat: **comparable, repeatable** dan **verifiable**.

BENTUK PEMROSESAN AWAL DATA

Pembersihan Data



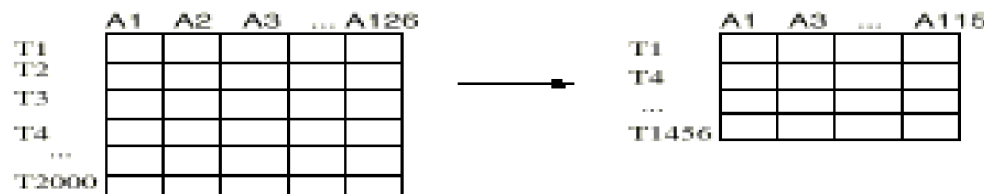
Integrasi Data



Transformasi Data

-2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Reduksi Data



Kenapa diperlukan?

- Data awal pada umumnya **"kotor"**.
- Data Baik menghasilkan output yang akurat.
- Berpengaruh pada Nilai presisi dan kinerja data mining .

Ekstraksi data, pembersihan, dan transformasi merupakan kerja utama dari pembuatan suatu data warehouse.

- Bill Inmon

PEMROSESAN AWAL DATA

- ***Pembersihan Data*** : Mengisi nilai-nilai yang hilang, menghaluskan noisy data, mengenali atau menghilangkan outlier, dan memecah ketidak konsistenan data.
- ***Integrasi Data*** : Integrasi banyak database, banyak kubus data, atau banyak file
- ***Transformasi Data*** : Normalisasi dan agregasi
- ***Reduksi Data*** : Mendapatkan representasi yang direduksi dalam volume tetapi menghasilkan hasil analitikal yang sama atau mirip
- ***Diskritisasi Data*** : Bagian dari reduksi data tetapi dengan kepentingan khusus, terutama data numerik.

DATA SAMPAH

- **Tidak-lengkap**: nilai-nilai atribut kurang, atribut tertentu yang dipentingkan tidak disertakan, atau hanya memuat data agregasi
 - Misal, pekerjaan=""
- **Noisy**: memuat error atau memuat outliers (data yang secara nyata berbeda dengan data-data yang lain)
 - Misal, Salary="-10"
- **Tidak-konsisten**: memuat perbedaan dalam kode atau nama
 - Misal, Age="42" Birthday="03/07/1997"
 - Misal, rating sebelumnya "1,2,3", sekarang rating "A, B, C"
 - Misal, perbedaan antara duplikasi record

NOISE

Information Sources

Attributes		Class
Att 1	Att 2	Class
0.25	red	positive
0.25	red	negative
0.99	green	negative
1.02	green	positive
2.05	?	negative
=	green	positive

Att. Noise Class Noise

Kinds of Noise

Class Noise

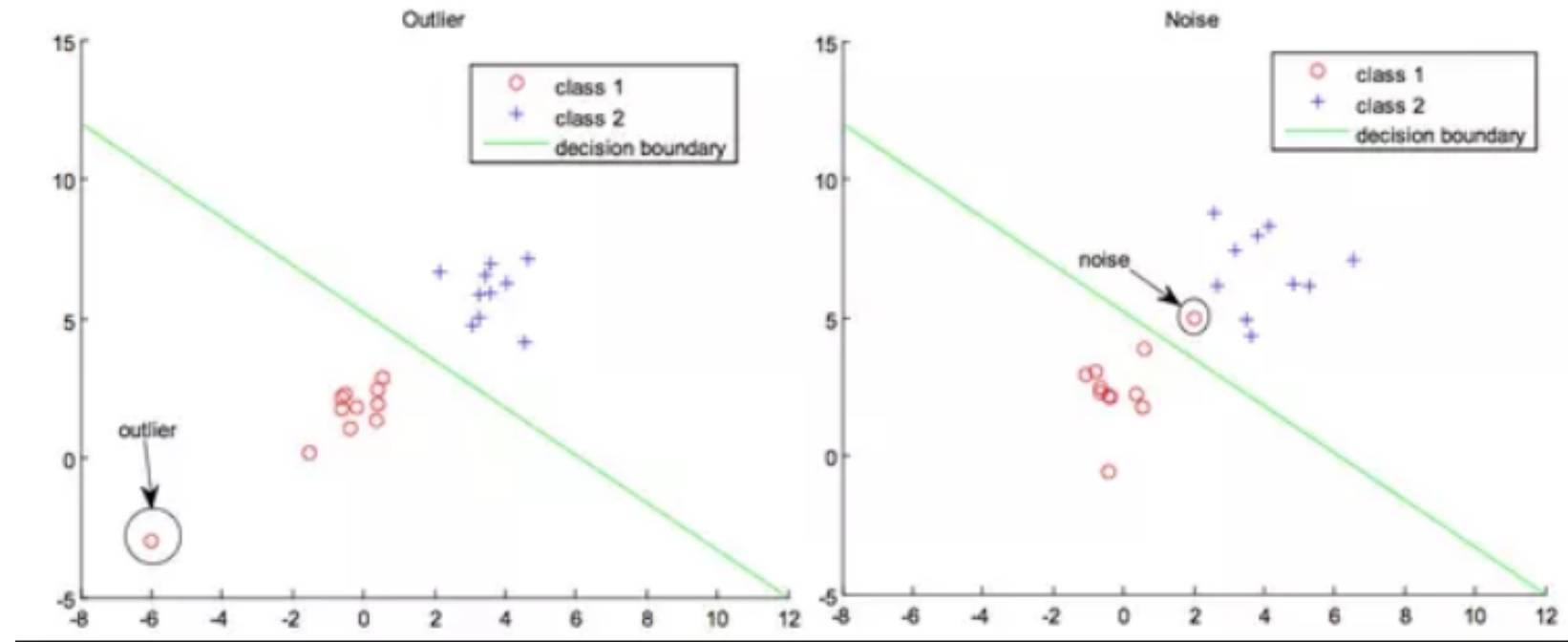
- Contradictory examples
- Mislabeled examples

Attribute Noise

- Erroneous values
- Missing values
- Don't care values

NOISE VS OUTLIER

- Noise adalah data yang tidak diinginkan / salah
- Outlier adalah nilai dari data diluar range / diluar nilai yang diharapkan
- Noise lebih sulit untuk dideteksi karena nilainya bisa menyerupai nilai yang diharapkan
- Baik noise maupun outlier sebaiknya dihilangkan pada saat proses preprocessing



outlier adalah data yang berada di luar jangkauan data yang diharapkan sedangkan **noise** adalah data yang tidak diinginkan dan salah

PENANGANAN MISSING VALUE (data tidak lengkap)

- Mengabaikan tuple atau record: mudah tetapi tidak efektif, dan merupakan metoda terakhir (*alternatif terakhir*)
- Mengisi nilai-nilai yang hilang secara manual (*effort besar*)
- Mengisi nilai-nilai yang hilang secara otomatis (*baik* dengan metode yang tepat), contoh :
 - Menggunakan konstanta global (“unknown”, “Null”, atau kelas baru)
 - Menggunakan nilai rata-rata atribut
 - Rata-rata atribut untuk seluruh sampel yang masuk kedalam kelas yang sama
 - Nilai yang paling mungkin (berbasis inferensi seperti regresi, rumus bayesian, atau pohon keputusan)
 - Dan lain-lain

Contoh Missing Value (Data Tidak Lengkap)

Case	Attributes			Decision
	Temperature	Headache	Nausea	Flu
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes

Beberapa record data memiliki atribut dengan nilai yang tidak sesuai / kosong.

Cara-1 : Mehapus record missing value

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	very_high	yes	yes	yes
2	high	yes	yes	yes
3	normal	yes	no	no
4	normal	no	yes	no

Cara-2 : Memberikan nilai paling dominan yang dimiliki atribut

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	high	no	no	no
4	high	yes	yes	yes
5	high	yes	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	high	yes	yes	yes

Cara-3 : Memberikan nilai berdasarkan pola nilai pada atribut lain

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	yes	no	yes
2	very_high	yes	yes	yes
3	normal	no	no	no
4	high	yes	yes	yes
5	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	high	yes	yes	yes

Cara-4 : Memberikan semua variasi nilai dari atribut

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1 ⁱ	high	yes	no	yes
1 ⁱⁱ	high	no	no	yes
2	very_high	yes	yes	yes
3 ⁱ	high	no	no	no
3 ⁱⁱ	very_high	no	no	no
3 ⁱⁱⁱ	normal	no	no	no
4	high	yes	yes	yes
5 ⁱ	high	yes	yes	no
5 ⁱⁱ	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8 ⁱ	high	yes	yes	yes
8 ⁱⁱ	high	yes	no	yes
8 ⁱⁱⁱ	very_high	yes	yes	yes
8 ^{iv}	very_high	yes	no	yes
8 ^v	normal	yes	yes	yes
8 ^{vi}	normal	yes	no	yes

Cara-5 : Memberikan semua variasi nilai berdasarkan pola nilai pada atribut lain

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	?	no	yes
2	very_high	yes	yes	yes
3	?	no	no	no
4	high	yes	yes	yes
5	high	?	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	high	yes	no	yes
2	very_high	yes	yes	yes
3 ⁱ	normal	no	no	no
3 ⁱⁱ	high	no	no	no
4	high	yes	yes	yes
5 ⁱ	high	yes	yes	no
5 ⁱⁱ	high	no	yes	no
6	normal	yes	no	no
7	normal	no	yes	no
8 ⁱ	high	yes	yes	yes
8 ⁱⁱ	high	yes	no	yes
8 ⁱⁱⁱ	very_high	yes	yes	yes
8 ^{iv}	very_high	yes	no	yes

Cara-6 : Memberikan nilai rata-rata atribut

Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	100.2	?	no	yes
2	102.6	yes	yes	yes
3	?	no	no	no
4	99.6	yes	yes	yes
5	99.8	?	yes	no
6	96.4	yes	no	no
7	96.6	no	yes	no
8	?	yes	?	yes



Case	Attributes			Decision
	Temperature	Headache	Nausea	
1	100.2	yes	no	yes
2	102.6	yes	yes	yes
3	99.2	no	no	no
4	99.6	yes	yes	yes
5	99.8	yes	yes	no
6	96.4	yes	no	no
7	96.6	no	yes	no
8	99.2	yes	yes	yes



Implementasi preprocessing dengan python

Data Preprocessing Template

Importing the libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

Importing the dataset

```
dataset = pd.read_csv('Data.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

Splitting the dataset into the Training set and Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)
```

Contoh Data Kasus

	Country	Age	Salary	Purchased
1	France	44	72000	No
2	Spain	27	48000	Yes
3	Germany	30	54000	No
4	Spain	38	61000	No
5	Germany	40		Yes
6	France	35	58000	Yes
7	Spain		52000	No
8	France	48	79000	Yes
9	Germany	50	83000	No
10	France	37	67000	Yes

Import library yang digunakan

```
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd
```

Library yang digunakan untuk preprocessing data diatas adalah library **numpy** dan **pandas**.

Import Dataset

```
dataset = pd.read_csv('Data.csv')  
X = dataset.iloc[:, :-1].values  
y = dataset.iloc[:, -1].values
```

```
print(X)
```

```
[['France' 44.0 72000.0]  
 ['Spain' 27.0 48000.0]  
 ['Germany' 30.0 54000.0]  
 ['Spain' 38.0 61000.0]  
 ['Germany' 40.0 nan]  
 ['France' 35.0 58000.0]  
 ['Spain' nan 52000.0]  
 ['France' 48.0 79000.0]  
 ['Germany' 50.0 83000.0]  
 ['France' 37.0 67000.0]]
```

```
print(y)
```

```
['No' 'Yes' 'No' 'No' 'Yes' 'Yes' 'No' 'Yes' 'No' 'Yes']
```

- Dataset tersimpan dengan nama file “**Data.csv**”, print(X) mencetak nilai atribut dan print(Y) mencetak nilai kelas.
- Perhatikan pada (X) masih terdapat **Noisy** nilai = **nan**
- [:, :-1]: pilih semua baris dalam dataset, serta semua kolom kecuali kolom terakhir (*Negative indexing* pada Python).
- [:, 0:3]: tanpa negative indexing (ada empat kolom pada dataset, kita memilih tiga kolom: indeks 0, 1, dan 2).
- [:, -1]: pilih semua baris, kolom terakhir.

Menghilangkan Missing Value (nan)

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
imputer.fit(X[:, 1:3])
X[:, 1:3] = imputer.transform(X[:, 1:3])
```

```
print(X)
```

```
[['France' 44.0 72000.0]
 ['Spain' 27.0 48000.0]
 ['Germany' 30.0 54000.0]
 ['Spain' 38.0 61000.0]
 ['Germany' 40.0 63777.77777777778]
 ['France' 35.0 58000.0]
 ['Spain' 38.77777777777778 52000.0]
 ['France' 48.0 79000.0]
 ['Germany' 50.0 83000.0]
 ['France' 37.0 67000.0]]
```

- Menambahkan library **sklearn**
- Class **SimpleImputer()** digunakan untuk mengganti nilai yang kosong dengan *mean* kolom.
 - **missing_values** : nilai data yang digunakan sebagai penanda bahwa nilai aslinya tidak ada (*missing*); dalam hal ini NaN (np.nan)
 - **strategy** : dalam hal ini rata-rata kolom (*'mean'*), bisa juga menggunakan *'median'*, *'most_frequent'* (modus), atau *'constant'*
- Selanjutnya obyek imputer harus di *fit* berdasarkan kolom yang bersangkutan menggunakan metode **fit()**

Encoding data kategori (Atribut)

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [0])], remainder='passthrough')
X = np.array(ct.fit_transform(X))
```

```
print(X)
```

```
[[1.0 0.0 0.0 44.0 72000.0]
 [0.0 0.0 1.0 27.0 48000.0]
 [0.0 1.0 0.0 30.0 54000.0]
 [0.0 0.0 1.0 38.0 61000.0]
 [0.0 1.0 0.0 40.0 63777.77777777778]
 [1.0 0.0 0.0 35.0 58000.0]
 [0.0 0.0 1.0 38.77777777777778 52000.0]
 [1.0 0.0 0.0 48.0 79000.0]
 [0.0 1.0 0.0 50.0 83000.0]
 [1.0 0.0 0.0 37.0 67000.0]]
```

- Matrix X yang terbentuk sebelumnya pada kolom **Country** bertipe string, jadi perlu diubah ke numerik (int atau float).
- Gunakan variable dummy **OneHotEncoder** dan **ColumnTransformer**

Encoding data kategori (Class / Label)

```
from sklearn.preprocessing import LabelEncoder  
le = LabelEncoder()  
y = le.fit_transform(y)
```

```
print(y)
```

```
[0 1 0 0 1 1 0 1 0 1]
```

- Matrix Y hanya akan diubah menjadi numerik (0, 1, dan seterusnya) dengan **LabelEncoder**

Membagi dataset ke dalam *training set* dan *test set*

```
from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 1)
```

```
print(X_train)
```

```
[[0.0 0.0 1.0 38.77777777777778 52000.0]  
 [0.0 1.0 0.0 40.0 63777.77777777778]  
 [1.0 0.0 0.0 44.0 72000.0]  
 [0.0 0.0 1.0 38.0 61000.0]  
 [0.0 0.0 1.0 27.0 48000.0]  
 [1.0 0.0 0.0 48.0 79000.0]  
 [0.0 1.0 0.0 50.0 83000.0]  
 [1.0 0.0 0.0 35.0 58000.0]]
```

```
print(X_test)
```

```
[[0.0 1.0 0.0 30.0 54000.0]  
 [1.0 0.0 0.0 37.0 67000.0]]
```

```
print(y_train)
```

```
[0 1 0 0 1 1 0 1]
```

```
print(y_test)
```

```
[0 1]
```

- **test_size** : proporsi test set, dalam hal ini 0.2.
- **train_size**: proporsi train size. Jika tidak di set, maka akan menyesuaikan dengan test size (dalam kasus ini 0.8). Berlaku kebalikannya.
- **random_state** : konstan ini akan membuat hasil splitting tetap sama antar runtime atau antar mesin. Nilai bebas.

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train[:, 3:] = sc.fit_transform(X_train[:, 3:])
X_test[:, 3:] = sc.transform(X_test[:, 3:])
```

```
print(X_train)
```

```
[[0.0 0.0 1.0 -0.19159184384578545 -1.0781259408412425]
 [0.0 1.0 0.0 -0.014117293757057777 -0.07013167641635372]
 [1.0 0.0 0.0 0.566708506533324 0.633562432710455]
 [0.0 0.0 1.0 -0.30453019390224867 -0.30786617274297867]
 [0.0 0.0 1.0 -1.9018011447007988 -1.420463615551582]
 [1.0 0.0 0.0 1.1475343068237058 1.232653363453549]
 [0.0 1.0 0.0 1.4379472069688968 1.5749910381638885]
 [1.0 0.0 0.0 -0.7401495441200351 -0.5646194287757332]]
```

```
print(X_test)
```

```
[[0.0 1.0 0.0 -1.4661817944830124 -0.9069571034860727]
 [1.0 0.0 0.0 -0.44973664397484414 0.2056403393225306]]
```

- Perlu dilakukan skala kolom-kolom yang dibutuhkan. Perbedaan skala dapat menyebabkan kendala dengan estimator.
- Ada tiga scaler di library scikit-learn yang sering digunakan: **StandardScaler**, **MinMaxScaler**, dan **RobustScaler**.
- **StandardScaler** menghilangkan mean (terpusat pada 0) dan menskalakan ke variansi (deviasi standar = 1), dengan asumsi data terdistribusi normal (gauss) untuk semua fitur.
- **MinMaxScaler** menskalakan nilai data ke dalam suatu *range*. Tidak masalah pada data non-gaussian.
- Sedangkan **RobustScaler** (sklearn.preprocessing.RobustScaler) mirip dengan Min-Max, hanya saja menggunakan range interkuartil. Scaler ini tahan terhadap outlier.