

Machine Learning Lanjutan



Outline Kelas

- Data preprocessing
 - Handling missing values
 - Handling duplicate rows
 - Feature Encoding
- Membangun Model Klasifikasi:
 - Decision tree
 - Random forest

Missing Value

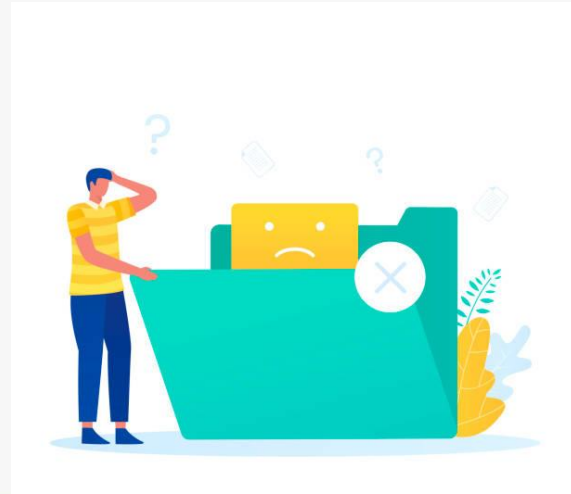
Missing Value

Pengertian : *Missing value* adalah peristiwa **hilang atau tidak terbacanya data**. Biasanya, data-data yang tidak dapat terdeteksi akan disimbolkan dengan "**NaN**", NULL atau bisa juga cell kosong

Tujuan : *Missing value* yang tidak diatasi dapat menyebabkan kesalahan hasil analisis dan kesimpulan yang tidak sesuai dengan yang diharapkan. Selain itu, ada **beberapa mesin algoritma yang tidak memperbolehkan adanya *missing value*** dalam sebuah data.

Cara menangani *missing value* yang biasa dilakukan adalah sebagai berikut :

- *Leave as it is* (biarkan saja)
- *Drop Row* yang mengandung *missing value*
- *Imputation Process*



Memeriksa Missing Value

Untuk memeriksa missing value pada sebuah data dapat menggunakan method `.isnull()` atau `.isna()`

```
# Import library yang dibutuhkan
import pandas as pd

# Proses ekstraksi data dari sumber berformat excel
data_transaction = pd.read_excel('/content/Transaction Customer Telco.xlsx')

# Periksa banyaknya missing value pada tiap kolom
check_null_sum = data_transaction.isnull().sum()

# Tampilkan hasilnya
display(check_null_sum)
```

output :

Customer ID	5
Offer	0
Phone Service	0
Multiple Lines	0
Internet Service	0
Internet Type	0
Online Security	0
Online Backup	0
Device Protection Plan	0
Premium Tech Support	0
Streaming TV	0
Streaming Movies	0
Streaming Music	0
Unlimited Data	0
Contract	0
Paperless Billing	0
Payment Method	0
Monthly Charge	6
Total Charges	0
Total Refunds	0
Total Extra Data Charges	0
Total Long Distance Charges	0
Total Revenue	0
Updated At	0

dtype: int64

```
# Import library yang dibutuhkan
import pandas as pd

# Proses ekstraksi data dari sumber berformat excel
data_transaction = pd.read_excel('/content/Transaction Customer Telco.xlsx')

# Periksa kondisi missing value pada data
check_null = data_transaction.isnull().any()

# Tampilkan hasil
display(check_null)
```

output :

Customer ID	True
Offer	False
Phone Service	False
Multiple Lines	False
Internet Service	False
Internet Type	False
Online Security	False
Online Backup	False
Device Protection Plan	False
Premium Tech Support	False
Streaming TV	False
Streaming Movies	False
Streaming Music	False
Unlimited Data	False
Contract	False
Paperless Billing	False
Payment Method	False
Monthly Charge	True
Total Charges	False
Total Refunds	False
Total Extra Data Charges	False
Total Long Distance Charges	False
Total Revenue	False
Updated At	False

dtype: bool

#1 Handling Missing Value

1. Leave As It Is

Membiarkan saja missing value pada data juga mungkin dilakukan. Biasanya data yang kosong **pada kolom yang tidak begitu dibutuhkan analisa lebih lanjut** akan dibiarkan kosong begitu saja.

Contoh : Missing value pada kolom alamat customer, deskripsi produk atau kolom kolom lain yang sifatnya kolom tambahan. Ignore saja kolomnya altogether.

2. Drop Missing Value

Menghapus satu atau lebih row jika terdapat missing value juga mungkin dilakukan. Biasanya saat kolom yang sebagai *mandatory* proses bisnis bernilai NULL atau kosong.

Contoh : Kolom transaction_id pada tabel transaksi merupakan kolom wajib yang biasanya di generate otomatis oleh system. Namun jika ada data pada kolom ini hilang maka biasanya satu atau lebih row perlu dihapus untuk mempertahankan kevalidan data (dianggap *data corrupt*)

2. Drop Missing Value

Pada saat jumlah missing values **kecil/tidak signifikan**, solusi terbaik adalah menghapus baris data yang mengandung missing values.

Pada pandas drop missing value dapat menggunakan sintaks berikut :

```
df = df.dropna()
```

#2 Handling Missing Value

3. Imputation

Imputation atau imputasi adalah proses **mengisi missing value dengan nilai tertentu**. Biasanya diisi dengan rata-rata populasi data atau median (kolom numerik), dan dengan modus (kolom kategorik)

Perhatikan contoh berikut

X	
4	Hitung rata-rata data $\bar{X} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{1}{5} (4 + 4 + 6 + 7 + 9)$ $\bar{X} = \frac{1}{5} (30) = 6$
4	
6	
7	
9	

Data dengan *missing value*

X	
4	Hitung Kembali rata-rata data $\bar{X} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{1}{7} (4 + 4 + 6 + 6 + 7 + 9 + 6)$ $\bar{X} = \frac{1}{7} (42) = 6$
4	
6	
6	
7	
9	
6	

Data setelah proses imputasi

Setelah dilakukan proses imputasi maka metrics rata-rata data tidak berubah

3. Imputation

Pada pandas, missing value dapat diisikan oleh perhitungan berikut

- Rataan : Gunakan sintaks `value = Series.mean()`
- Median : Gunakan sintaks `value = Series.median()`
- Modus : Gunakan sintaks `value = Series.mode()`

Setelah dilakukan perhitungan value barulah proses imputasi dilakukan dengan method `.fillna(value)`

Contoh:

```
df['gender'] = df['gender'].fillna(df['gender'].mode)
```

Redudansi Data

Redudansi Data

Pengertian : Redudansi data adalah duplikasi atau penyimpanan **data yang sama secara berulang** dalam satu atau lebih tabel, sehingga data yang sama di simpan di dalam lebih dari 1 baris.

Data bisa terjadi duplikasi karena kesalahan manusia (*Human Error*) atau bisa jadi karena kesalahan sistem.

Cara menangani data duplikat yang biasa dilakukan adalah sebagai berikut :

- *Drop Row* yang mengandung duplikasi



Handling Duplicate Data

Untuk **memeriksa** status duplikasi pada sebuah dataframe dapat menggunakan sintaks berikut :

```
DataFrame.duplicated().sum()
```

Jika output dari sintaks tersebut >0, maka terdapat duplikasi/redundansi baris pada data.

Jika demikian, kita dapat menghapus baris yang duplikat tersebut dengan sintaks sebagai berikut

```
DataFrame.drop_duplicates()
```

Mengenal

Feature Encoding

Apa itu **Feature Encoding**?

Feature Encoding adalah proses mengubah feature categorical menjadi feature numeric.

Mengapa kita perlu *feature encoding*?
Tak semua model/algorithm ML dapat menggunakan feature categorical.

Data Besaran Penghasilan dari Survei Abhal²

Gender	Pendidikan	Pekerjaan	Penghasilan (juta)
Laki-laki	S1	SWASTA	7
Laki-laki	SMA	PNS	13
Perempuan	S1	PNS	15
Laki-laki	S2	FREELANCE	24
Perempuan	S3	PNS	17
Perempuan	S1	SWASTA	23
Perempuan	SMA	FREELANCE	12


Pertanyaan :

Bagaimana cara menulis rumus dari Penghasilan **secara matematis**?

Teknik #1: Label Encoding

Label Encoding adalah perubahan feature categorical menjadi numeric dengan memberikan angka yang berbeda bagi masing-masing nilai unik

```
mapping_gender = {  
    'Laki-laki': 0,  
    'Perempuan': 1  
}  
df['gender'] = df['gender'].map(mapping_gender)  
  
mapping_pendidikan = {  
    'SMA': 0,  
    'S1': 1,  
    'S2': 2,  
    'S3': 3  
}  
df['pendidikan'] = df['pendidikan'].map(mapping_pendidikan)
```



	gender	pendidikan	pekerjaan	penghasilan
0	0	1	SWASTA	7
1	0	0	PNS	13
2	1	1	PNS	15
3	0	2	FREELANCE	24
4	1	3	PNS	17
5	1	1	SWASTA	23
6	1	0	FREELANCE	12

**Lalu bagaimana
dengan kolom
'pekerjaan'?**

Teknik #2: One-hot Encoding

One-hot encoding adalah perubahan feature categorical menjadi numeric dengan menjadikan masing-masing nilai unik feature tersendiri

```
1 pd.get_dummies(df['pekerjaan'], prefix='kerja')
```

Kode di atas menunjukkan cara melakukan one-hot encoding pada kolom pekerjaan menggunakan `get_dummies()`. Berikut penjelasannya:

1. Parameter pertama adalah kolom yang ingin di one-hot encoding (pekerjaan)
2. Parameter prefix diisi dengan nama awalan dari kolom-kolom baru yang akan dihasilkan
3. Fungsi ini akan mengembalikan dataframe baru yang berisi feature-feature numerik

Teknik #2: One-hot Encoding (lanjutan)

	kerja_FREELANCE	kerja_PNS	kerja_SWASTA
0	0	0	1
1	0	1	0
2	0	1	0
3	1	0	0
4	0	1	0
5	0	0	1
6	1	0	0

Ketika kita tampilkan dataframe yang dihasilkan terlihat bahwa setiap nilai unik berubah menjadi kolom baru.

Awalan nama kolom-kolom baru ini sesuai dengan isi parameter `prefix`.

Data Besaran Penghasilan dari Survei Abhal² (encoded)

Gender	Pendidikan	Freelance	PNS	Swasta	Penghasilan
0	1	0	0	1	7
0	0	0	1	0	13
1	1	0	1	0	15
0	2	1	0	0	24
1	3	0	1	0	17
1	1	0	0	1	23
1	0	1	0	0	12

Rumus nilai penghasilan:

- $A * \text{gender} + B * \text{pendidikan} + C * \text{freelance} + D * \text{PNS} + E * \text{swasta}$
- Dimana A, B, C, D, E didapat dari melatih model machine learning (regresi)

Recap :

Label Encoding
atau OHE?

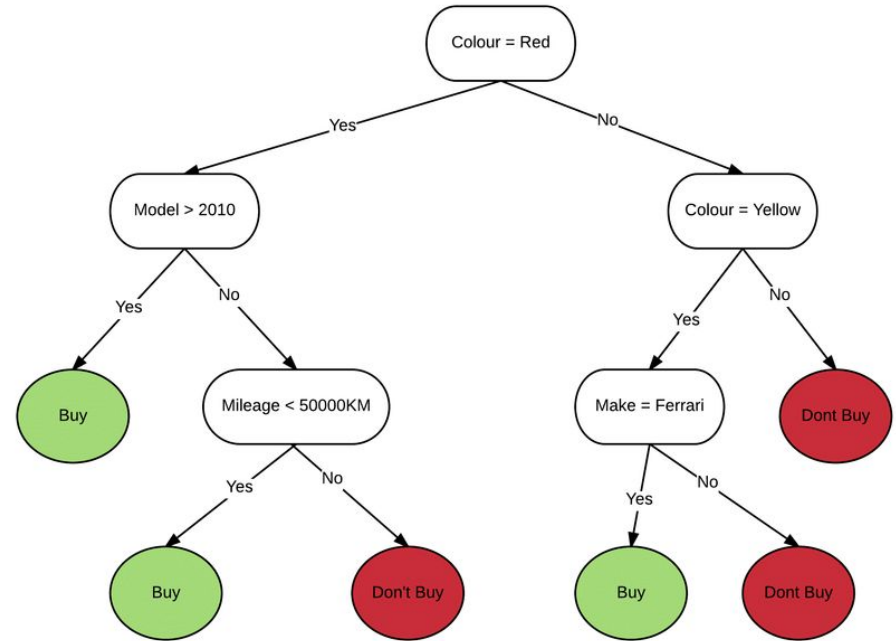
- **Gunakan Label Encoding pada:**
 - Kolom kategorikal dengan jumlah distinct values = 2. E.g. Gender, respon ya/tidak, etc
 - Kolom kategorikal dengan tipe ordinal (punya urutan). E.g. tingkat pendidikan, intensitas (rendah/medium/tinggi), socio-economic status (A/B/C/D), etc
- **Selainnya, gunakan One Hot Encoding (OHE)**

Pemodelan Klasifikasi: Decision Tree

Machine Learning - Classification - Tree

Decision tree adalah salah satu dari sekian banyak model yang dapat digunakan untuk klasifikasi.

Model ini berbentuk pohon keputusan, di mana tiap data "ditanyai" pertanyaan binary (if-else) hingga didapatkan kelas yang tepat

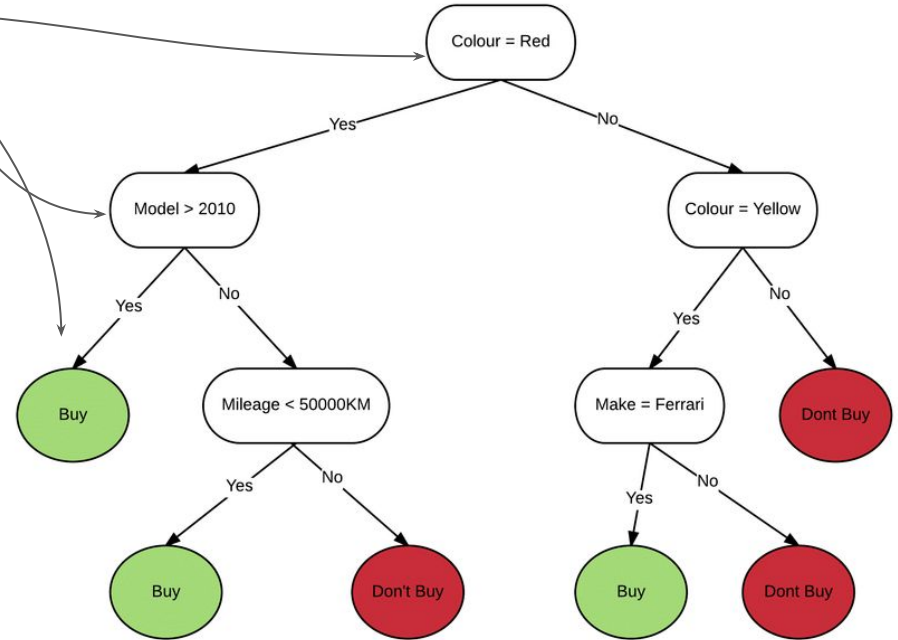


Decision Tree

Node

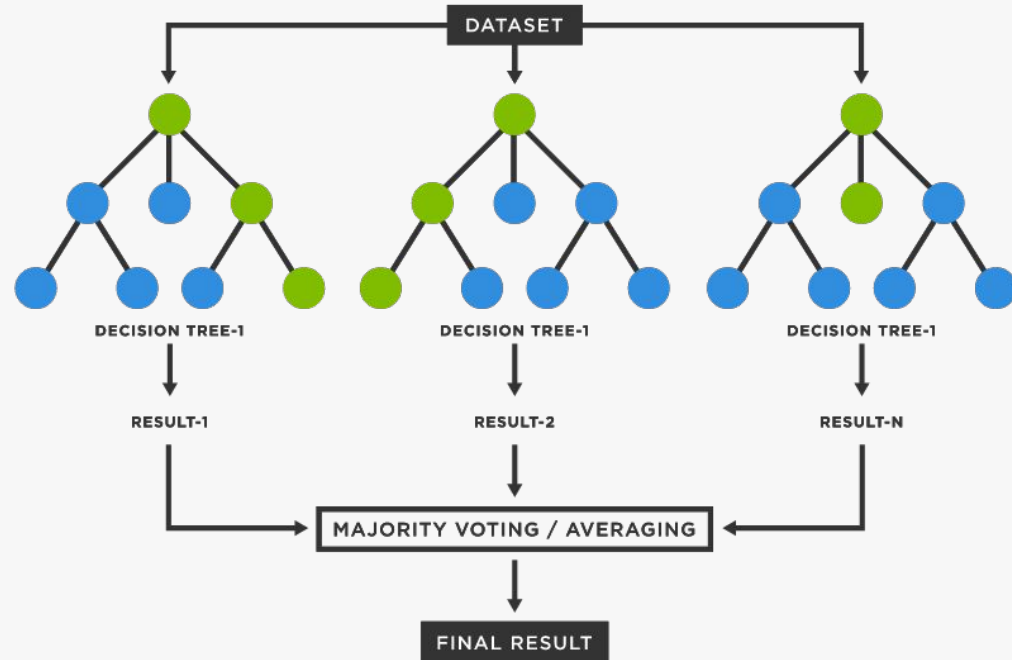
Node tanpa "orang tua" = *root*
Node tanpa "anak" = *leaf* (*hasil prediksi label*)

Aturan pada node diperoleh dari proses model training.



Random Forest

- Random forest adalah model klasifikasi yang dibentuk dari kumpulan decision tree
- Mekanisme prediksi random forest serupa dengan voting
 - Label dengan suara terbanyak akan menjadi hasil prediksi keseluruhan



Evaluasi performa model

1. Setelah model selesai di-train, kita perlu mengevaluasi performa model pada test data
2. Metrik yang dapat digunakan adalah akurasi
 - a. I.e. persentase jumlah data poin yang diprediksi secara tepat dibanding dengan jumlah keseluruhan data poin yang ada

		Actual Class	
		Positive	Negative
Predicted Class	Positive	True Positive 44	False Positive 15
	Negative	False Negative 4	True Negative 37

Figure 2

maka akurasi model tersebut adalah $(44+37) \div (44+15+4+37) = 81/100 = 81\%$.

Data yang dipakai

- Kita akan memakai churn.csv
- Data tsb tentang memprediksi apakah seseorang akan melakukan churn (berhenti berlangganan)

	customer_id	gender	senior_citizen	partner	dependent	tenure	contract	payment_method	monthly_charges	total_charges	churn
0	7590-VHVEG	Female	0.0	NaN	No	1.0	monthly	Electronic check	29.85	29.85	No
1	5575-GNVDE	Male	0.0	No	No	34.0	bimonthly	Mailed check	56.95	1889.5	No
2	3668-QPYBK	Male	0.0	No	No	2.0	monthly	Mailed check	53.85	108.15	Yes
3	7795-CFOCW	Male	0.0	No	No	45.0	bimonthly	Bank transfer (automatic)	42.30	1840.75	No
4	9237-HQITU	Female	0.0	No	No	2.0	monthly	Electronic check	70.70	151.65	Yes

Langkah pemodelan

Prasyarat: data yang sudah dipreproses

1. Split data menjadi dua bagian
 - a. Training data (80%)
 - b. Test data (20%)
2. Convert data menjadi numpy array
3. Fit model
 - a. Decision tree
 - b. Random forest
4. Evaluasi performa model pada test data
 - a. Pilih model yang performanya terbaik di test data

Hands-On

Thanks!

