

# Klasifikasi dengan Decision Tree

PERTEMUAN 10



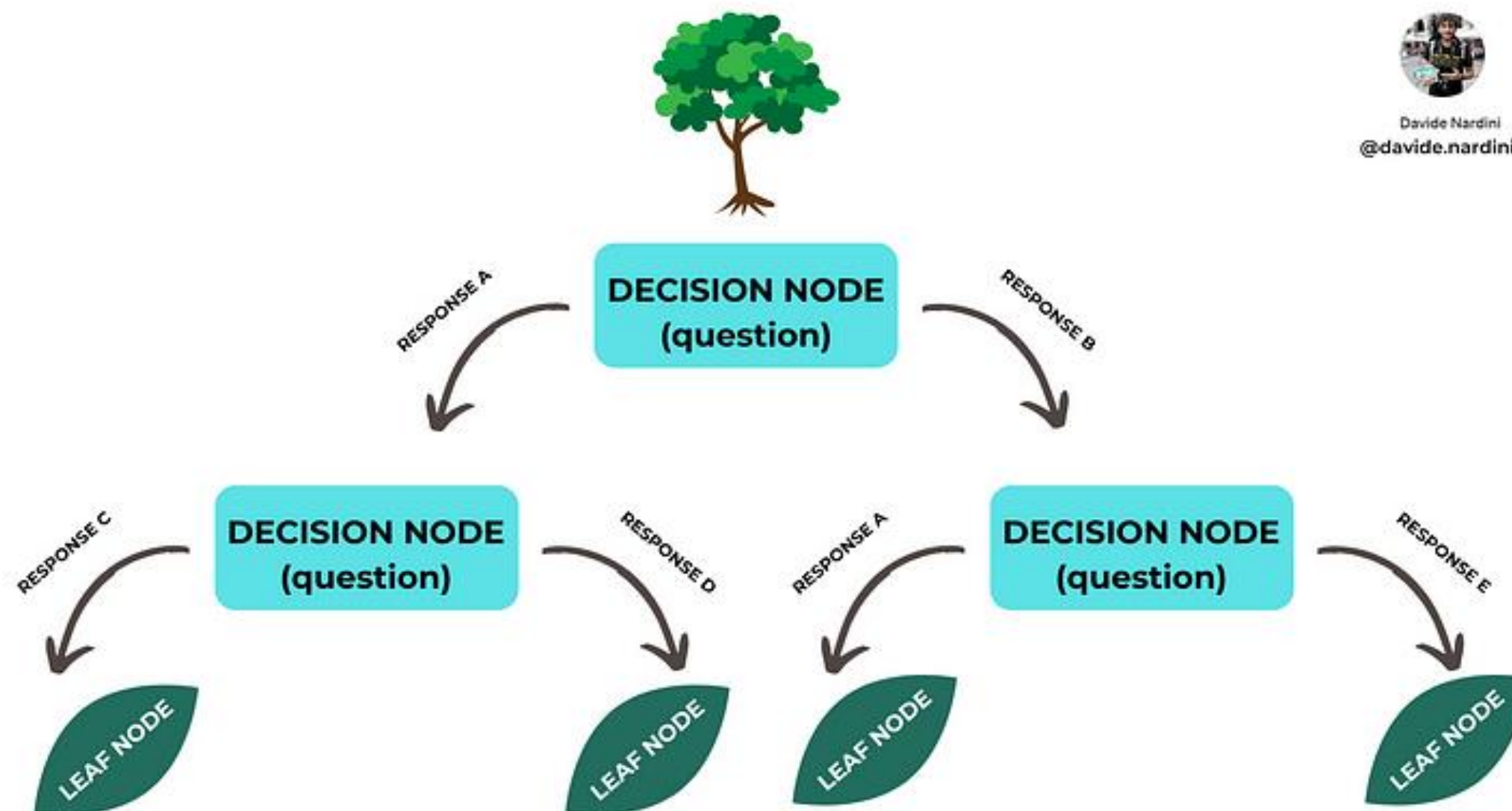
# Apa itu Decision Tree dan cara kerjanya?

Decision Tree (pohon keputusan) berfungsi sebagai algoritma pembelajaran mesin terbimbing (*supervised learning*) yang terbukti bermanfaat untuk tugas klasifikasi dan regresi .

Memahami istilah “keputusan” dan “pohon” sangat penting dalam memahami algoritma ini: pada dasarnya, pohon keputusan membuat keputusan dengan menganalisis data dan membangun struktur mirip pohon untuk memfasilitasi proses ini.

Anggap saja ini sebagai **konstruksi “jika-maka-lainnya” yang canggih** , terutama dengan respons biner. Kami mengajukan **pertanyaan berurutan** , dengan setiap pertanyaan mewakili sebuah **simpul** . Secara bersamaan, setiap respons potensial menandakan sebuah **cabang** di dalam pohon. Pohon Keputusan menghasilkan **prediksi** berdasarkan **jalur dalam arsitektur pohon** , yang pada akhirnya mengarah ke **daun** , yang merupakan bagian akhir model.

# Apa itu Decision Tree dan cara kerjanya?

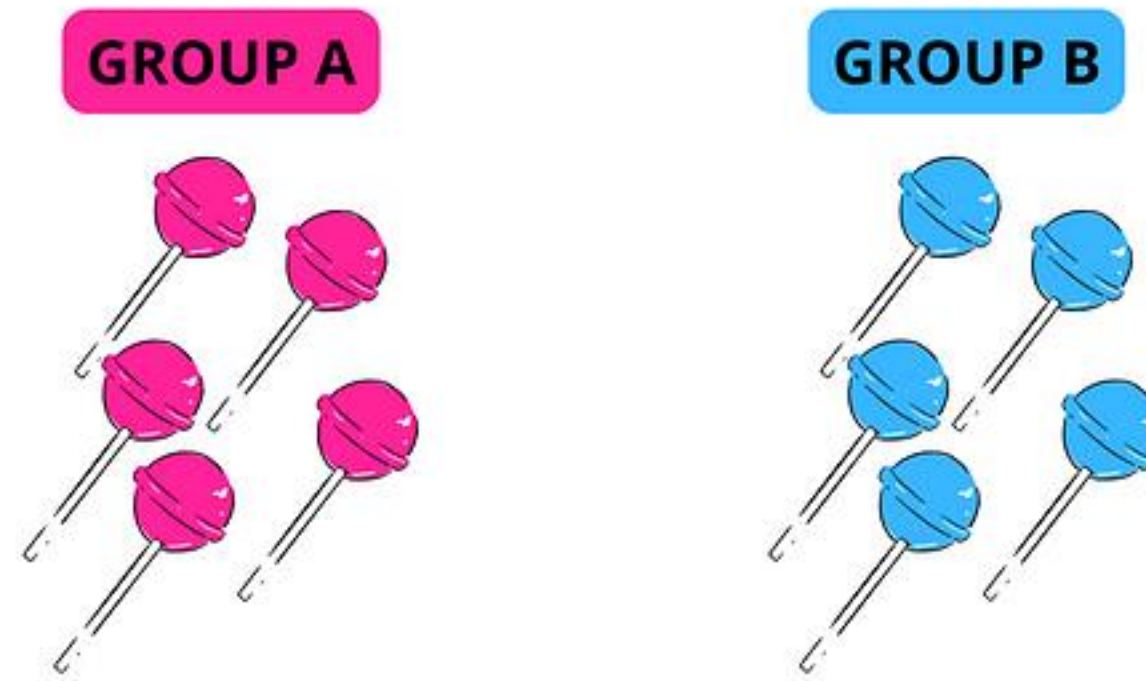


  
Davide Nardini  
@davide.nardini16

Di Node Keputusan, kita menemukan pertanyaan, sedangkan respons potensial diwakili oleh cabang-cabangnya. Namun... bagaimana Pohon Keputusan menentukan pertanyaan dan tanggapan terkait?

# Apa itu Decision Tree dan cara kerjanya?

Perhatikan gambar ini

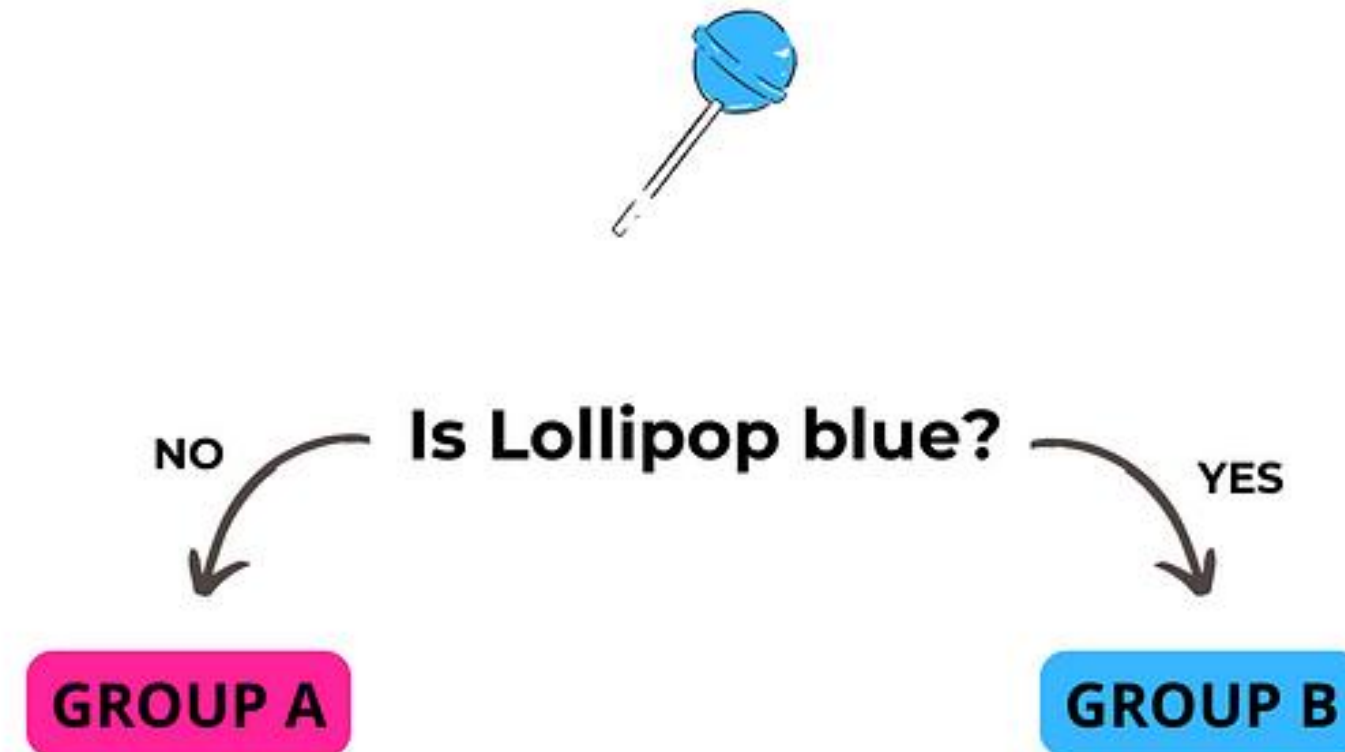


Apa perbedaan utama antara grup A dan grup B?



# Apa itu Decision Tree dan cara kerjanya?

Pohon Keputusan yang mewakili skenario dengan lolipop akan muncul sebagai berikut:

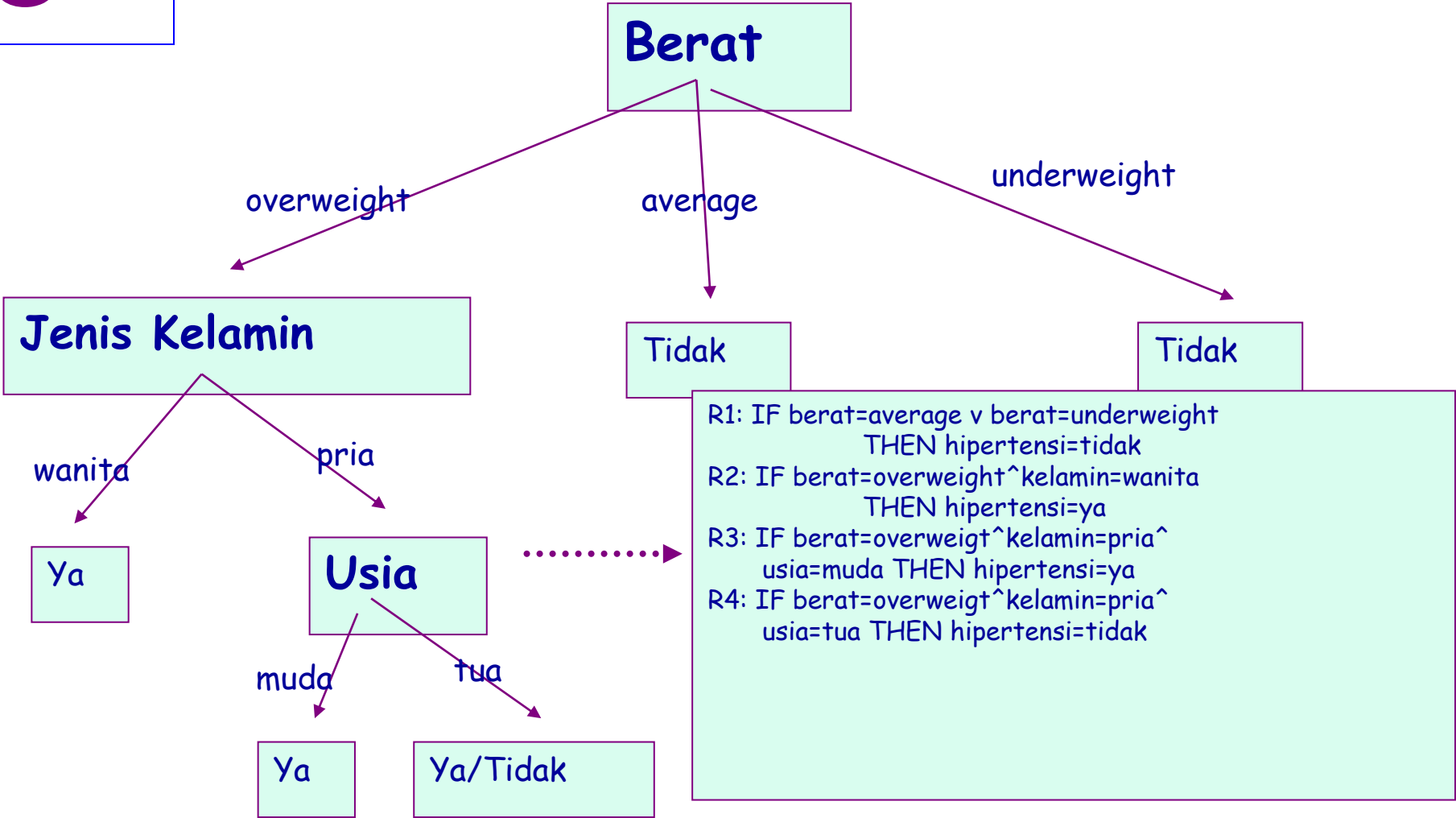


# Konsep Decision Tree

Mengubah data menjadi pohon keputusan (*decision tree*) dan aturan-aturan keputusan (*rule*)



Nama	Usia	Berat	Kelamin	Hipertensi
Ali	muda	overweight	pria	ya
Edi	muda	underweight	pria	tidak
Annie	muda	average	wanita	tidak
Budiman	tua	overweight	pria	tidak
Herman	tua	overweight	pria	ya
Didi	muda	underweight	pria	tidak
Rina	tua	overweight	wanita	ya
Gatot	tua	average	pria	tidak

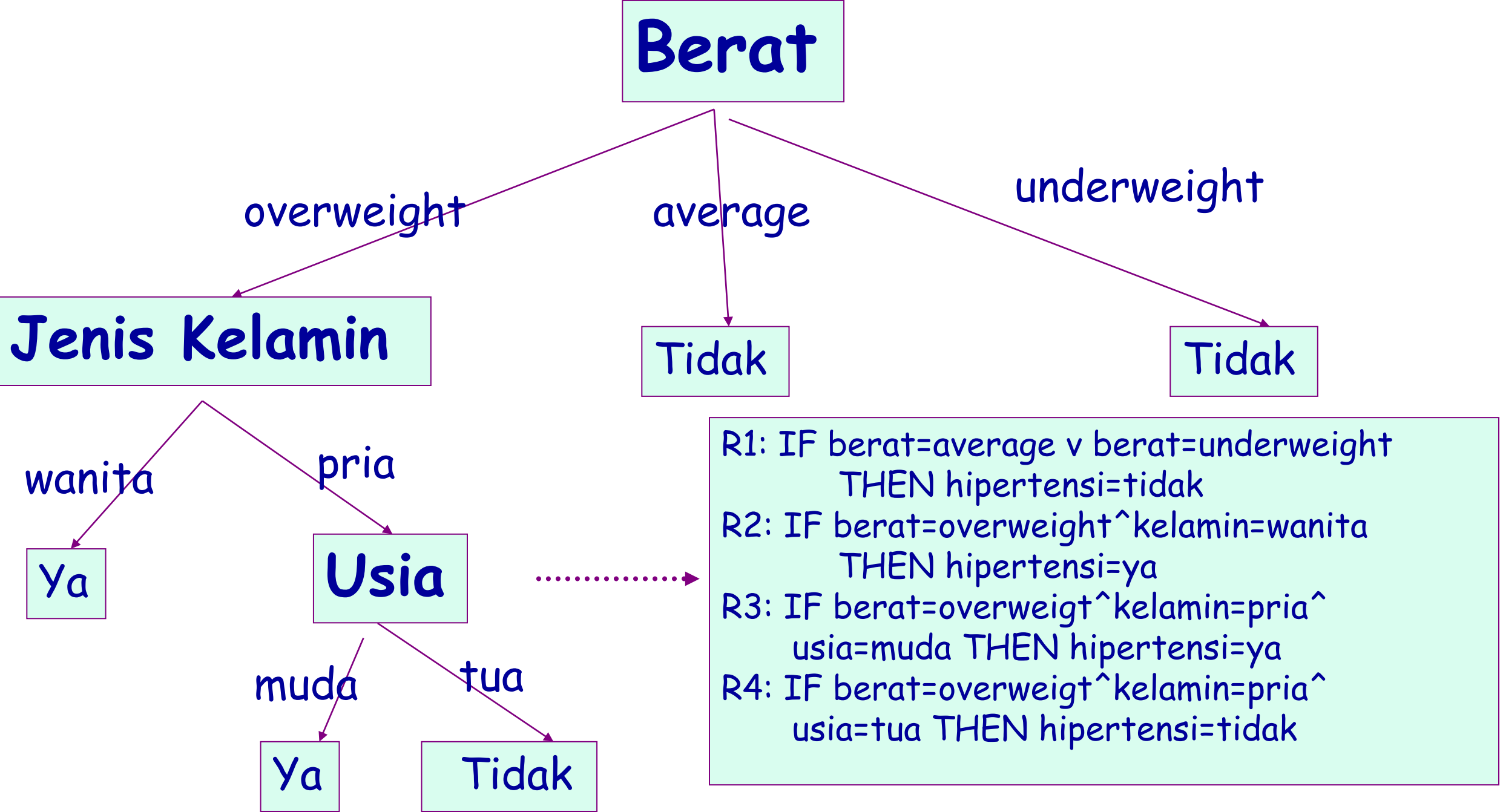


# Gambaran Pemakaian Decision Tree

Membuat aturan (**rule**) yang dapat digunakan untuk menentukan apakah seseorang mempunyai potensi untuk menderita hipertensi atau tidak berdasarkan data usia, berat badan dan jenis kelamin.

#	Usia	Berat Badan	Kelamin	Hipertensi
1	muda	overweight	pria	ya
2	muda	underweight	pria	tidak
3	muda	average	wanita	tidak
4	tua	overweight	pria	tidak
5	tua	overweight	pria	ya
6	muda	underweight	pria	tidak
7	tua	overweight	wanita	ya
8	tua	average	pria	tidak

# Gambaran Pemakaian Decision Tree



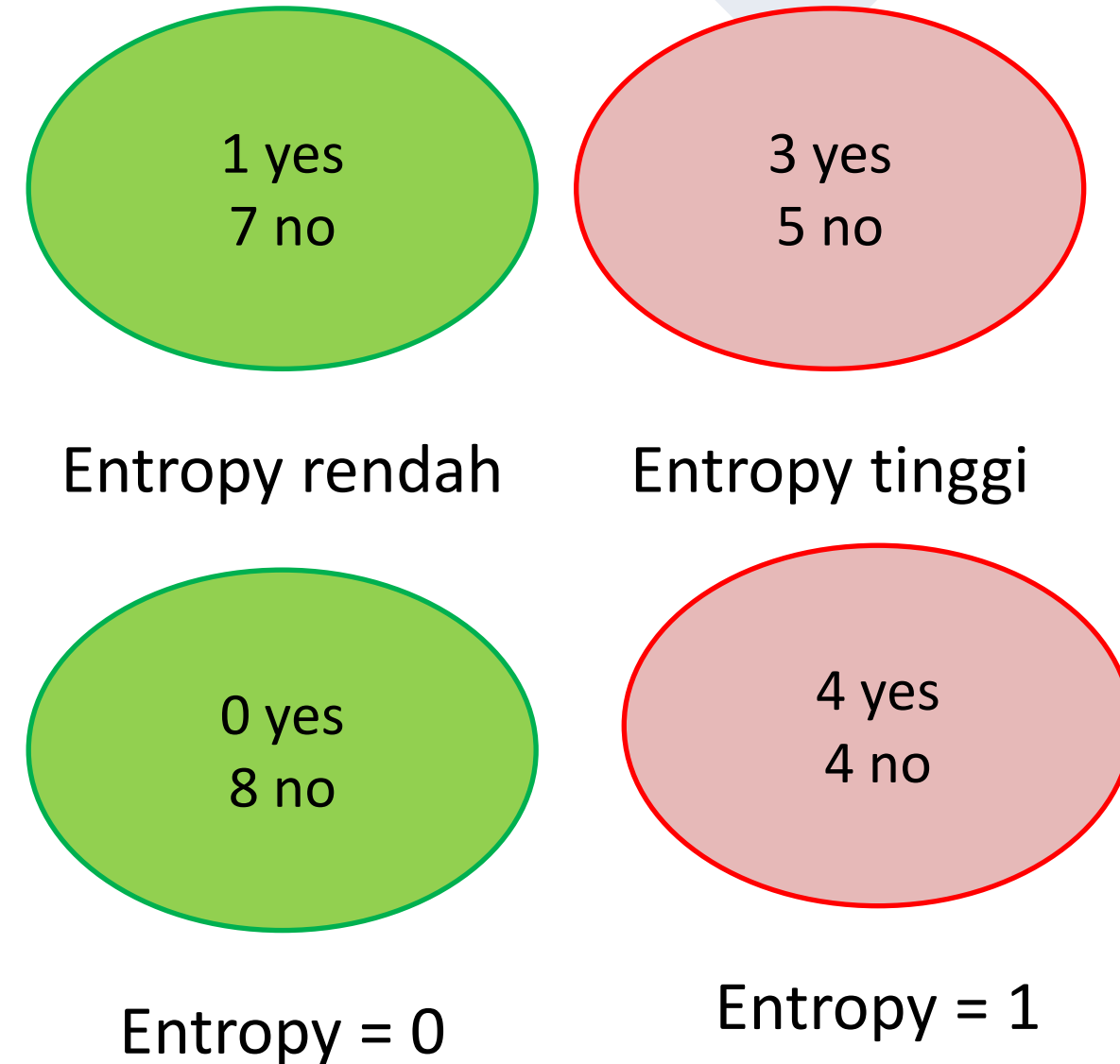


# ENTROPY

- Ukuran kemurnian, keberagaman, randomness atau uncertainty
- Nilai entropy semakin kecil, distribusi semakin homogen, node semakin murni (pure)

$$Info(D) = -\sum_{i=1}^m p_i \log_2(p_i)$$

\*Info(D) = Entropy (D)



# Information Gain

- Information gain (sering disebut “gain” saja) merupakan teknik yang digunakan dalam algoritma decision tree untuk menentukan atribut yang paling tepat digunakan sebagai kriteria dalam membagi bagian data menjadi beberapa bagian yang lebih kecil.
- Information gain menggambarkan seberapa besar entropy berkurang akibat suatu atribut. Semakin nilai information gain besar, semakin bagus.

$$\text{InformationGain}(Y, X) = E(Y) - E(Y|X)$$

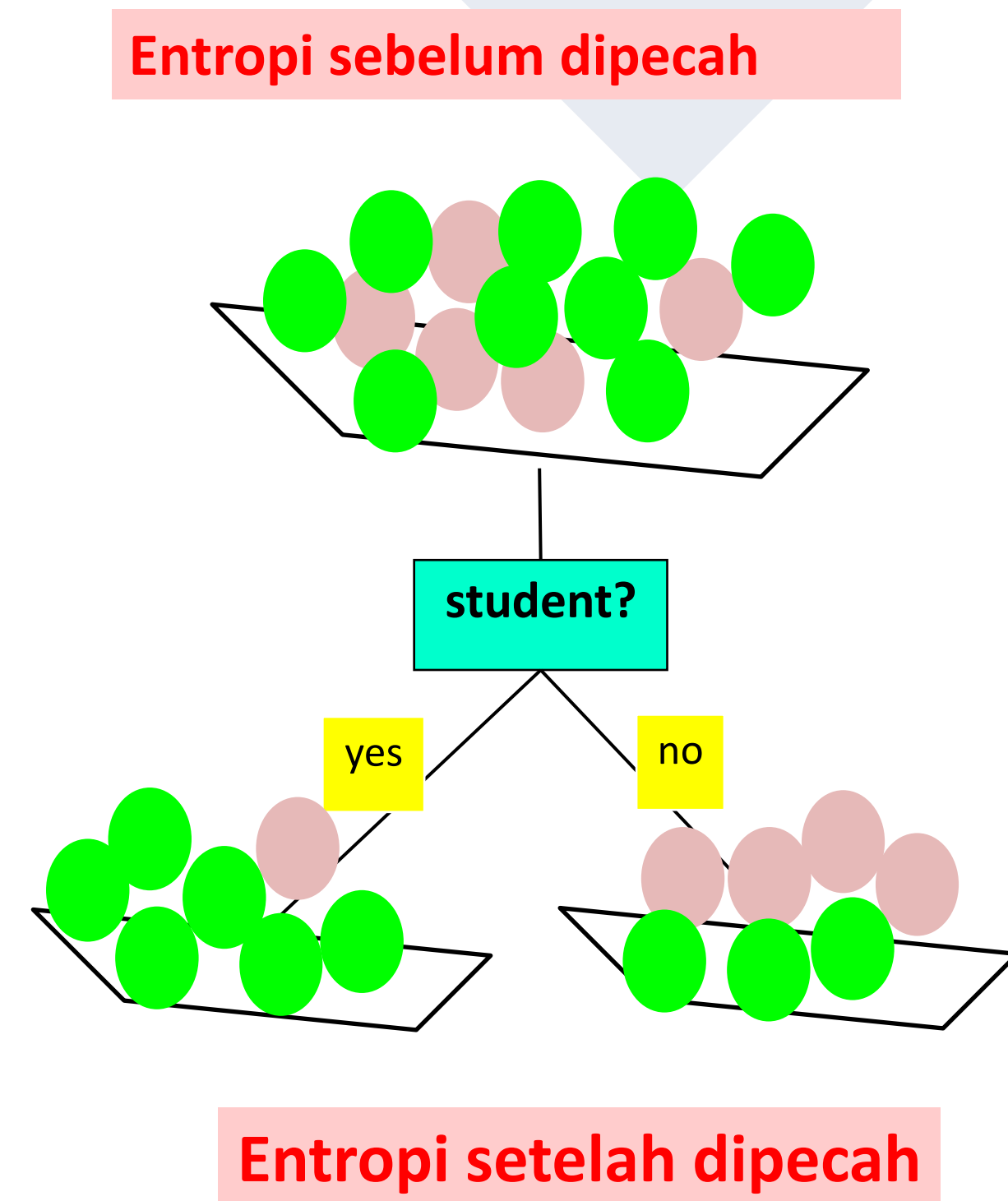
Ket:

$E(Y)$  : Entropi dari fitur target

$E(Y|X)$  : Entropi rata-rata dari fitur predictor terhadap fitur target

$X$ : Fitur prediktor

$Y$ : Fitur target



# Mengapa Decision Tree?

- Mudah diimplementasikan
- Hipotesis yang dihasilkan mudah dipahami
- Efisien

## Decision Tree Cocok untuk Masalah:

- Data dalam bentuk atribut-nilai. Kondisi ideal adalah jika isi nilai jumlahnya sedikit. Misalnya: “panas”, “sedang”, “dingin”.
- Output diskrit.
- Training data dapat tidak lengkap

# Kelebihan dan Kekurangan Decision Tree

- Kelebihan:
  - Mudah diimplementasikan
  - Hipotesis yang dihasilkan mudah dipahami
  - Efisien
- Kekurangan:
  - Overfitting: terlalu mengikuti training data
    - Terlalu banyak cabang, merefleksikan anomali akibat noise atau outlier.
    - Akurasi rendah untuk data baru
  - Proses pembangunan pohon keputusan pada data numerik menjadi lebih rumit dan memungkinkan terdapat informasi yang hilang
  - Pohon keputusan dapat tumbuh menjadi sangat kompleks pada data yang rumit.

# Pengembangan Decision Tree

- Mengatasi overfitting
  - **Pre-pruning:** Hentikan pembuatan tree di awal. Tidak mensplit node jika goodness measure dibawah threshold.
  - **Post-pruning:** Buang cabang setelah tree jadi
- Pengembangan Metode:
  - C4.5, C5.0
  - Conditional Decision Tree
  - Gradient-boosted Trees
  - Random Forest

# Studi Kasus

Jenis Kelamin	Asal SMA	Nikah	Program	Kelulusan
Laki-laki	SMA	Belum	Reguler	Tepat
Laki-laki	SMA	Belum	Reguler	Tepat
Laki-laki	SMK	Belum	Ekstensi	Terlambat
Perempuan	SMA	Belum	Reguler	Tepat
Laki-laki	SMA	Belum	Reguler	Tepat
Laki-laki	SMA	Belum	Reguler	Tepat
Laki-laki	SMK	Sudah	Ekstensi	Terlambat
Laki-laki	SMK	Belum	Ekstensi	Terlambat
Laki-laki	SMK	Belum	Reguler	Tepat
Perempuan	SMK	Belum	Reguler	Terlambat

# Studi Kasus

## Langkah 1

Menghitung entropi dan information gain pada setiap fitur yang ada pada dataset. Salah satu cara untuk mempermudah perhitungan ini adalah dengan membuat tabel kontingensi untuk setiap fitur pada dataset.

Jenis Kelamin	Kelulusan		
	Tepat	Terlambat	Total
Laki-laki	5	3	8
Perempuan	1	1	2
Total	6	4	10

nikah	Kelulusan		
	Tepat	Terlambat	Total
Belum	6	3	9
Sudah	0	1	1
Total	6	4	10

Asal SMA	Kelulusan		
	Tepat	Terlambat	Total
SMA	5	0	5
SMK	1	4	5
Total	6	4	10

Program	Kelulusan		
	Tepat	Terlambat	Total
Reguler	6	1	7
Ekstensi	0	3	3
Total	6	4	10

# Studi Kasus

Menghitung nilai entropi dari keseluruhan dataset. Entropi ini menggambarkan tingkat keacakan atau ketidakpastian dalam data. Semakin tinggi nilai entropi, semakin banyak variasi dalam data.

$$E(Kelulusan) = \left( -\frac{6}{10} \times \log_2 \left( \frac{6}{10} \right) \right) + \left( -\frac{4}{10} \times \log_2 \left( \frac{4}{10} \right) \right) \approx 0.97$$

Menghitung entropi pada fitur jenis kelamin:

$$E(Kelulusan|Jenis kelamin = laki - laki) = \left( -\frac{5}{8} \times \log_2 \left( \frac{5}{8} \right) \right) + \left( -\frac{3}{8} \times \log_2 \left( \frac{3}{8} \right) \right) \approx 0.9544$$

$$E(Kelulusan|Jenis kelamin = perempuan) = \left( -\frac{1}{2} \times \log_2 \left( \frac{1}{2} \right) \right) + \left( -\frac{1}{2} \times \log_2 \left( \frac{1}{2} \right) \right) \approx 1$$

$$E(Kelulusan|Jenis kelamin) = \left( \frac{6}{10} \times 0.9544 \right) + \left( \frac{4}{10} \times 1 \right) \approx 0.9635$$

Menghitung entropi pada fitur Asal SMA

$$E(Kelulusan|Asal SMA = SMA) = \left( -\frac{5}{5} \times \log_2 \left( \frac{5}{5} \right) \right) + \left( -\frac{0}{5} \times \log_2 \left( \frac{0}{5} \right) \right) \approx 0$$

$$E(Kelulusan|Asal SMA = SMK) = \left( -\frac{1}{5} \times \log_2 \left( \frac{1}{5} \right) \right) + \left( -\frac{4}{5} \times \log_2 \left( \frac{4}{5} \right) \right) \approx 0.7219$$

$$E(Kelulusan|Asal SMA) = \left( \frac{6}{10} \times 0 \right) + \left( \frac{4}{10} \times 0.7219 \right) \approx 0.3610$$



# Studi Kasus

Menghitung entropi pada fitur Nikah

$$E(Kelulusan|Nikah = Belum) = \left(-\frac{6}{9} \times \log_2 \left(\frac{6}{9}\right)\right) + \left(-\frac{3}{9} \times \log_2 \left(\frac{3}{9}\right)\right) \approx 0.9183$$

$$E(Kelulusan|Nikah = Sudah) = \left(-\frac{0}{1} \times \log_2 \left(\frac{0}{1}\right)\right) + \left(-\frac{1}{1} \times \log_2 \left(\frac{1}{1}\right)\right) \approx 0$$

$$E(Kelulusan|Nikah) = \left(\frac{6}{10} \times 0.9183\right) + \left(\frac{4}{10} \times 0\right) \approx 0.8265$$

Menghitung entropi pada fitur Asal Program

$$E(Kelulusan|Program = Reguler) = \left(-\frac{6}{7} \times \log_2 \left(\frac{6}{7}\right)\right) + \left(-\frac{1}{7} \times \log_2 \left(\frac{1}{7}\right)\right) \approx 0.5917$$

$$E(Kelulusan|Program = Ekstensi) = \left(-\frac{0}{3} \times \log_2 \left(\frac{0}{3}\right)\right) + \left(-\frac{3}{3} \times \log_2 \left(\frac{3}{3}\right)\right) \approx 0$$

$$E(Kelulusan|Program) = \left(\frac{6}{10} \times 0.5917\right) + \left(\frac{4}{10} \times 0\right) \approx 0.4142$$

# Studi Kasus

Langkah selanjutnya adalah menghitung nilai information gain untuk masing-masing fitur.

**Information gain untuk fitur jenis kelamin.**

$$IG(Kelulusan, Jenis Kelamin) = E(Kelulusan) - E(Kelulusan|Jenis kelamin) = 0.97 - 0.9635 = 0.0074$$

**Information gain untuk fitur asal SMA.**

$$IG(Kelulusan, Asal SMA) = E(Kelulusan) - E(Kelulusan|Asal SMA) = 0.97 - 0.3610 = 0.6100$$

**Information gain untuk fitur nikah.**

$$IG(Kelulusan, Nikah) = E(Kelulusan) - E(Kelulusan|Nikah) = 0.97 - 0.8265 = 0.1445$$

**Information gain untuk fitur program.**

$$IG(Kelulusan, Program) = E(Kelulusan) - E(Kelulusan|Program) = 0.97 - 0.4142 = 0.5568$$

$$InformationGain(Y, X) = E(Y) - E(Y|X)$$

Ket:

$E(Y)$  : Entropi dari fitur target

$E(Y|X)$  : Entropi rata-rata dari fitur predictor terhadap fitur target

$X$ : Fitur prediktor

$Y$ : Fitur target

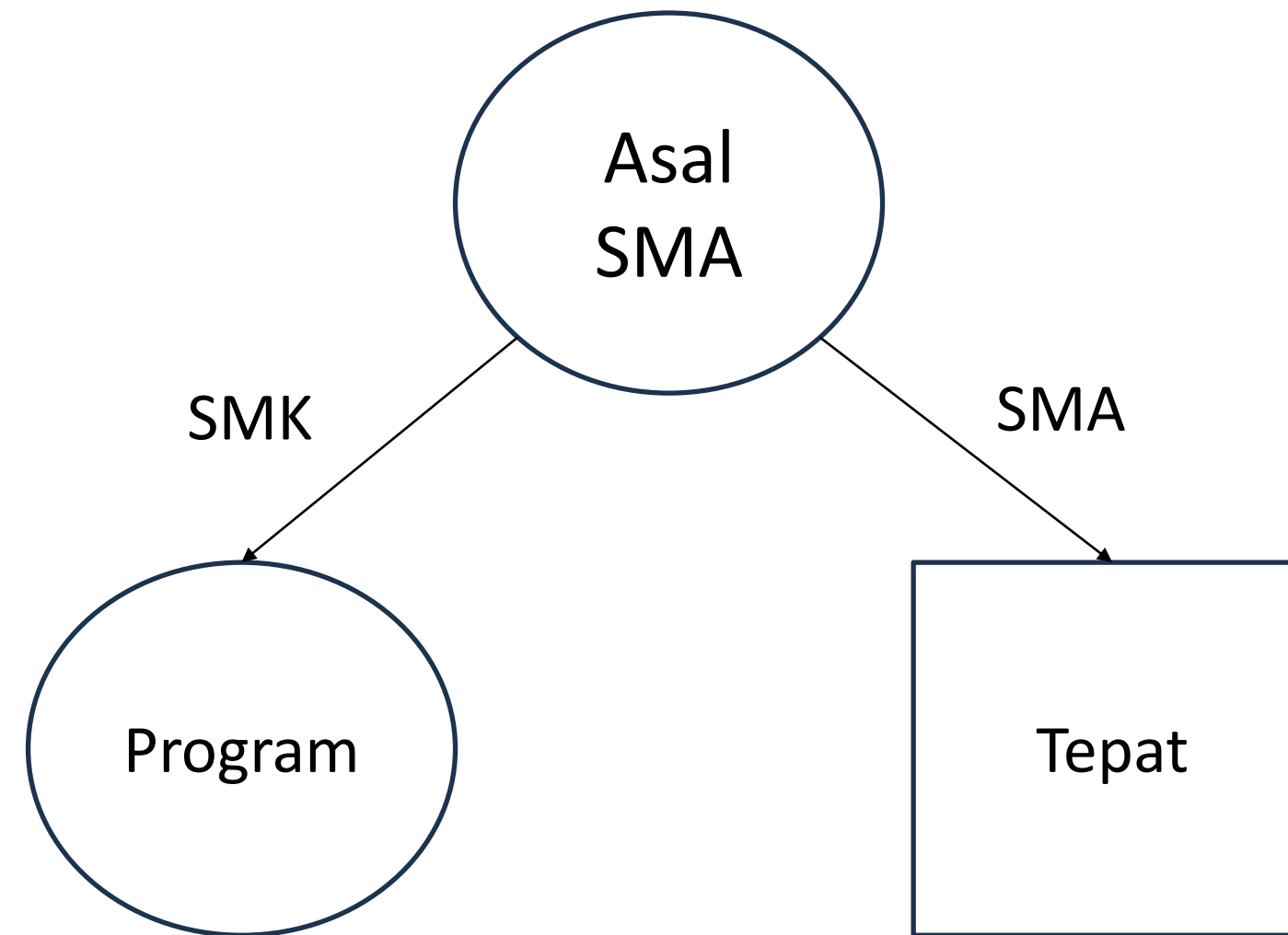
# Studi Kasus

Langkah selanjutnya adalah mengumpulkan hasilnya dalam bentuk seperti dibawah ini.

Fitur	Kelulusan			Entropi	Entropi Per Fitur	Information Gain
	Tepat	Terlambat	Total			
Jenis Kelamin						
Laki-laki	5	3	8	0,9544	0.9635	0.0074
Perempuan	1	1	2	1		
Asal SMA						
SMA	5	0	5	0	0.3610	0.6100
SMK	1	4	5	0.7219		
Nikah						
Belum	6	3	9	0.9183	0.8265	0.1445
Sudah	0	1	1	0		
Program						
Reguler	6	1	7	0.5917	0.4142	0.5568
Ekstensi	0	3	3	0		

# Studi Kasus

Untuk melanjutkan pembentukan decision tree, kita dapat memilih fitur dengan information gain paling tinggi sebagai akar pohon.



Untuk menentukan cabang dan daun dari decision tree, langkah pertama yang dilakukan adalah mencari entropi dan information gain untuk setiap fitur seperti sebelumnya. Namun, pada tahap ini dilakukan filterisasi pada fitur asal SMA untuk SMK, karena cabang akan muncul dari kelas ini. Setelah proses filterisasi, akan dihasilkan tabel baru yang berisi informasi yang relevan dengan kelas SMK. Selanjutnya, kita dapat menentukan cabang dan daun dengan mencari nilai information gain tertinggi dari setiap fitur pada tabel baru tsb.

# Studi Kasus

Jenis Kelamin	Asal SMA	Nikah	Program	Kelulusan
Laki-laki	SMK	Belum	Ekstensi	Terlambat
Laki-laki	SMK	Sudah	Ekstensi	Terlambat
Laki-laki	SMK	Belum	Ekstensi	Terlambat
Laki-laki	SMK	Belum	Reguler	Tepat
Perempuan	SMK	Belum	Reguler	Terlambat

Berikutnya menghitung nilai entropi keseluruhan dari data yang telah dilakukan filterisasi.

$$E(Kelulusan) = \left(-\frac{1}{5} \times \log_2 \left(\frac{1}{5}\right)\right) + \left(-\frac{4}{5} \times \log_2 \left(\frac{4}{5}\right)\right) \approx 0.72$$

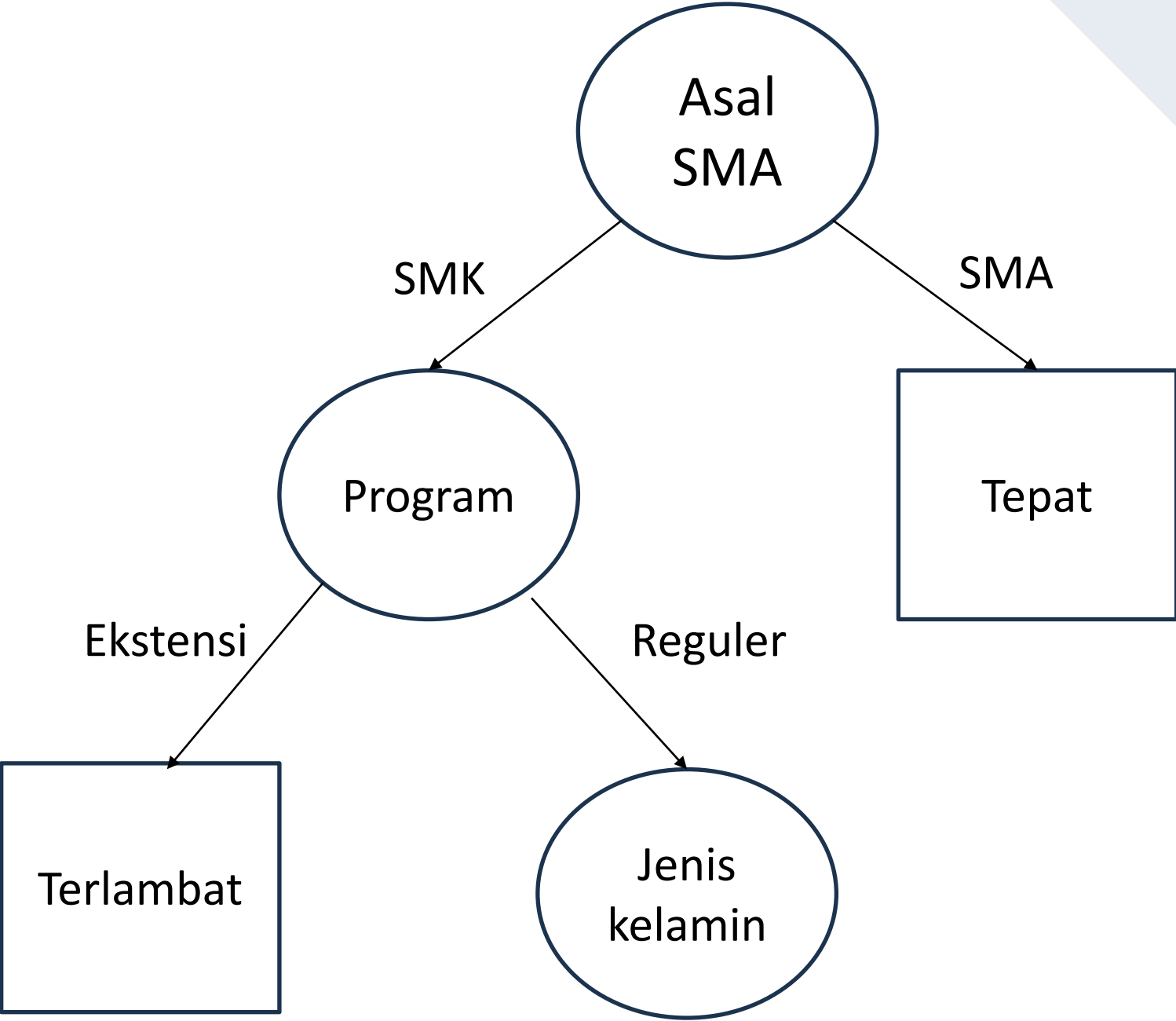
Langkah selanjutnya adalah menghitung nilai entropi dan nilai information gain untuk seluruh fitur dan kelas setelah dilakukan filterisasi pada fitur **asal SMA** untuk kelas SMK. Proses perhitungan dilakukan menggunakan rumus yang sama pada proses perhitungan sebelumnya.

# Studi Kasus

Fitur	Kelulusan			Entropi	Entropi Per Fitur	Information Gain
	Tepat	Terlambat	Total			
Jenis Kelamin						
Laki-laki	1	3	4	0.8113	0.6490	0.0729
Perempuan	0	1	1	0		
Asal SMA						
SMK	1	4	5	0.7219	0.7219	0
Nikah						
Belum	1	3	4	0.8113	0.6490	0.0729
Sudah	0	1	1	0		
Program						
Reguler	1	1	2	1	0.3333	0.3886
Ekstensi	0	4	4	0		

Dari tabel tsb, dapat dilihat bahwa fitur program memiliki nilai information gain tertinggi, sehingga akan menjadi cabang selanjutnya pada pohon keputusan. Daun untuk kelas **reguler** akan bernilai tepat, sedangkan daun untuk kelas **ekstensi** akan bernilai terlambat.

# Studi Kasus



Lakukan langkah yang sama seperti sebelumnya untuk memfilter kelas reguler pada fitur program. Hasilnya sbb:

Jenis Kelamin	Asal SMA	Nikah	Program	Kelulusan
Laki-laki	SMK	Belum	Reguler	Tepat
Perempuan	SMK	Belum	Reguler	Terlambat

# Studi Kasus

Berikutnya menghitung nilai entropi keseluruhan dari data yang telah dilakukan filterisasi.

$$E(Kelulusan) = \left(-\frac{1}{2} \times \log_2 \left(\frac{1}{2}\right)\right) + \left(-\frac{1}{2} \times \log_2 \left(\frac{1}{2}\right)\right) \approx 1$$

Kemudian, menghitung nilai entropi dan nilai information gain untuk seluruh fitur yang telah difilter sehingga diperoleh hasil sbb:

Fitur	Kelulusan			Entropi	Entropi Per Fitur	Information Gain
	Tepat	Terlambat	Total			
Jenis Kelamin						
Laki-laki	1	0	1	1	0	0,398
Perempuan	0	1	1	0		
Asal SMA						
SMK	1	1	2	1	1	0
Nikah						
Belum	1	1	2	1	1	0,5
Program						
Reguler	1	1	2	1	1	0



# Studi Kasus

Menghitung entropi pada fitur Nikah

$$E(Kelulusan|Nikah = Belum) = \left(-\frac{1}{2} \times \log_2 \left(\frac{1}{2}\right)\right) + \left(-\frac{1}{2} \times \log_2 \left(\frac{1}{2}\right)\right) \approx 1$$

$$E(Kelulusan|Nikah = Sudah) = \left(-\frac{0}{0} \times \log_2 \left(\frac{0}{0}\right)\right) + \left(-\frac{0}{0} \times \log_2 \left(\frac{0}{0}\right)\right) \approx 0$$

$$E(Kelulusan|Nikah) = \left(\frac{1}{2} \times 1\right) + \left(\frac{1}{2} \times 0\right) \approx 0,5$$

Menghitung entropi pada fitur jenis kelamin:

$$E(Kelulusan|Jenis kelamin = laki - laki) = \left(-\frac{1}{1} \times \log_2 \left(\frac{1}{1}\right)\right) + \left(-\frac{0}{1} \times \log_2 \left(\frac{0}{1}\right)\right) \approx 0,301$$

$$E(Kelulusan|Jenis kelamin = perempuan) = \left(-\frac{0}{1} \times \log_2 \left(\frac{0}{1}\right)\right) + \left(-\frac{1}{1} \times \log_2 \left(\frac{1}{1}\right)\right) \approx 0,301$$

$$E(Kelulusan|Jenis kelamin) = \left(\frac{1}{2} \times 0,301\right) + \left(\frac{1}{2} \times 0,301\right) \approx 0,602$$

**Information gain untuk fitur nikah.**

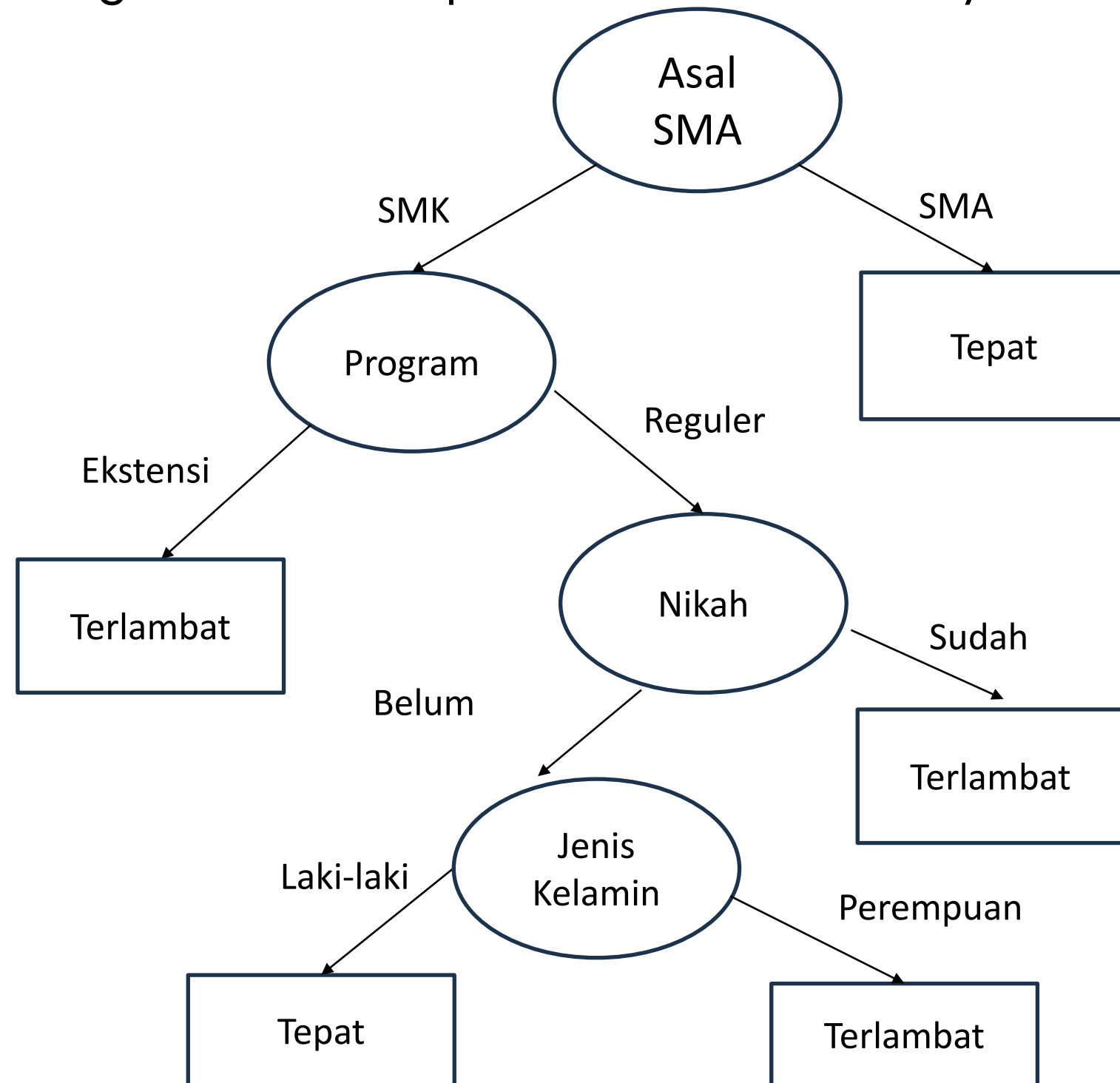
$$IG(Kelulusan, Nikah) = E(Kelulusan) - E(Kelulusan|Nikah) = 1 - 0,5 = 0,5$$

**Information gain untuk fitur jenis kelamin.**

$$IG(Kelulusan, Jenis Kelamin) = E(Kelulusan) - E(Kelulusan|Jenis Kelamin) = 1 - 0,602 = 0,398$$

# Studi Kasus

Dari tabel hasil perhitungan di atas dapat dilihat bahwa fitur **jenis kelamin** memiliki nilai information gain tertinggi, sehingga akan menjadi cabang selanjutnya pada pohon keputusan. Daun untuk kelas **jenis kelamin** (laki-laki) akan bernilai tepat, sedangkan daun untuk kelas perempuan akan bernilai terlambat. Dengan demikian, pohon decision tree nya akan berubah menjadi berikut:



Gambar disamping merupakan hasil akhir dari perhitungan decision tree yang telah dilakukan. Aturan IF-ELSE dapat dihasilkan dari pohon keputusan yang dihasilkan menggunakan algoritma decision tree. Aturan tersebut dapat dijadikan sebagai pedoman untuk mengklasifikasikan data baru.

## RULES:

R1: IF Asal SMA = SMA, THEN Kelulusan = Tepat

R2: IF Asal SMA = SMK dan Program = Ekstensi, THEN Kelulusan = Terlambat

R3: IF Asal SMA = SMK dan Program = Reguler dan Nikah = Belum dan Jenis Kelamin = Laki-laki, THEN Kelulusan = Tepat

R4: IF Asal SMA = SMK dan Program = Reguler dan Nikah = Belum dan Jenis Kelamin = Perempuan, THEN Kelulusan = Terlambat.

# Program Python

**Dataset** : Car Evaluation

**Sumber** : <https://www.kaggle.com/datasets/elikplim/car-evaluation-dataset?resource=download>

## Import Library

```
In [32]: import pandas as pd
import numpy as np
from sklearn.tree import DecisionTreeClassifier
import sklearn.model_selection as ms
```

```
In [8]: df = pd.read_csv('car_evaluation.csv')
df
```

Out[8]:

	vhigh	vhigh.1	2	2.1	small	low	unacc
0	vhigh	vhigh	2	2	small	med	unacc
1	vhigh	vhigh	2	2	small	high	unacc
2	vhigh	vhigh	2	2	med	low	unacc
3	vhigh	vhigh	2	2	med	med	unacc
4	vhigh	vhigh	2	2	med	high	unacc
...	...	...	...	...	...	...	...
1722	low	low	5more	more	med	med	good
1723	low	low	5more	more	med	high	vgood
1724	low	low	5more	more	big	low	unacc
1725	low	low	5more	more	big	med	good
1726	low	low	5more	more	big	high	vgood

1727 rows × 7 columns

Dataset terdiri dari 1727 record dengan 6 atribut independen dan 1 target.

Tujuan : untuk mengklasifikasikan mobil berdasarkan beberapa kriteria seperti harga, jumlah pintu, kapasitas penumpang dan jenis bodi mobil.

Dengan menggunakan algoritma Decision Tree, kita akan membangun model yang dapat memprediksi kelas mobil (unacc, acc) berdasarkan kriteria yang ada,

# Program Python

## Eksplorasi Data Analysis

```
In [47]: df.shape
```

```
Out[47]: (1727, 7)
```

```
In [34]: col_names = ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']

df.columns = col_names

col_names
```

```
Out[34]: ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']
```

```
In [48]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1727 entries, 0 to 1726
Data columns (total 7 columns):
#   Column      Non-Null Count  Dtype
---  -
0   buying      1727 non-null   object
1   maint       1727 non-null   object
2   doors       1727 non-null   object
3   persons     1727 non-null   object
4   lug_boot    1727 non-null   object
5   safety      1727 non-null   object
6   class       1727 non-null   object
dtypes: object(7)
memory usage: 94.6+ KB
```

```
In [49]: # check missing values in variables
```

```
df.isnull().sum()
```

```
Out[49]: buying      0
maint      0
doors      0
persons    0
lug_boot   0
safety     0
class      0
dtype: int64
```

# Program Python

## Eksplorasi Data Analysis

```
In [35]: # Let's again preview the dataset  
df.head()
```

Out[35]:

	buying	maint	doors	persons	lug_boot	safety	class
0	vhigh	vhigh	2	2	small	med	unacc
1	vhigh	vhigh	2	2	small	high	unacc
2	vhigh	vhigh	2	2	med	low	unacc
3	vhigh	vhigh	2	2	med	med	unacc
4	vhigh	vhigh	2	2	med	high	unacc

**Output (Target)/class : digunakan untuk memprediksi kelas mobil yang terdiri dari:**

Unacc = Tidak dapat diterima

Acc = Diterima

### Atribut:

- Buying : atribut ini mengindikasikan harga pembelian mobil. Terdapat 4 kategori pada atribut ini. Yaitu: vhigh (very high), high, medium, dan low. Semakin tinggi kategori, semakin tinggi pula harga mobil yang dibeli.
- Maint : atribut ini mengindikasikan biaya perawatan mobil
- Doors : atribut ini mengindikasikan jumlah pintu pada mobil.
- Persons : atribut ini mengindikasikan kapasitas mobil, yaitu banyak penumpang yang dapat diangkut.
- Hug\_boot : atribut ini mengindikasikan kapasitas bagasi mobil.
- Safety : atribut ini mengindikasikan tingkat keamanan mobil.

# Program Python

## Menentukan fitur dan variabel target

```
In [37]: X = df.drop(['class'], axis=1)
        y = df['class']
```

## Membagi data menjadi set pelatihan dan set pengujian

```
In [38]: # split X and y into training and testing sets

        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)
```

## Feature Engineering

```
In [53]: # check data types in X_train

        X_train.dtypes
```

```
Out[53]: buying      object
        maint      object
        doors      object
        persons     object
        lug_boot    object
        safety      object
        dtype: object
```

Kita dapat melihat bahwa semua variabel adalah tipe data kategorikal ordinal.

# Program Python

## Enkode variabel kategori

```
In [54]: X_train.head()
```

Out[54]:

	buying	maint	doors	persons	lug_boot	safety
1648	low	low	3	2	small	high
1465	low	high	4	2	big	high
361	vhigh	low	3	4	small	high
315	vhigh	med	5more	more	small	med
1372	low	vhigh	4	more	med	high

```
In [40]: # import category encoders
```

```
import category_encoders as ce
```

```
In [41]: # encode variables with ordinal encoding
```

```
encoder = ce.OrdinalEncoder(cols=['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety'])
```

```
X_train = encoder.fit_transform(X_train)
```

```
X_test = encoder.transform(X_test)
```

```
In [42]: X_train.head()
```

Out[42]:

	buying	maint	doors	persons	lug_boot	safety
1648	1	1	1	1	1	1
1465	1	2	2	1	2	1
361	2	1	1	2	1	1
315	2	3	3	3	1	2
1372	1	4	2	3	3	1

# Program Python

## Decision Tree Classifier with entropy

```
In [72]: clf_en = DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=0)

# fit the model
clf_en.fit(X_train, y_train)
```

```
Out[72]:
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=0)
```

## Memprediksi data testing dengan entropy

```
In [44]: y_pred_en = clf_en.predict(X_test)
```

## Memeriksa skor akurasi dengan entropy

```
In [76]: from sklearn.metrics import accuracy_score

print('Model accuracy score with criterion entropy: {0:0.4f}'.format(accuracy_score(y_test, y_pred_en)))

Model accuracy score with criterion entropy: 0.7890
```



# Program Python

## Membandingkan akurasi data training dan data testing

```
In [82]: y_pred_train_en = clf_en.predict(X_train)

y_pred_train_en
```

```
Out[82]: array(['unacc', 'unacc', 'acc', ..., 'unacc', 'acc', 'acc'], dtype=object)
```

```
In [83]: print('Training-set accuracy score: {0:0.4f}'.format(accuracy_score(y_train, y_pred_train_en)))

Training-set accuracy score: 0.8096
```

## Memeriksa overfitting and underfitting

```
In [84]: # print the scores on training and test set

print('Training set score: {:.4f}'.format(clf_en.score(X_train, y_train)))

print('Test set score: {:.4f}'.format(clf_en.score(X_test, y_test)))

Training set score: 0.8096
Test set score: 0.7890
```

Kita dapat melihat bahwa skor set pelatihan dan skor set tes sama seperti di atas. Skor akurasi set pelatihan sebesar 0,8096 sedangkan akurasi set tes sebesar 0,7890. Kedua nilai ini cukup sebanding. Jadi, tidak ada tanda-tanda overfitting.

# Program Python

## Confusion matrix

```
In [78]: # Print the Confusion Matrix and slice it into four pieces
```

```
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred_en)

print('Confusion matrix\n\n', cm)
```

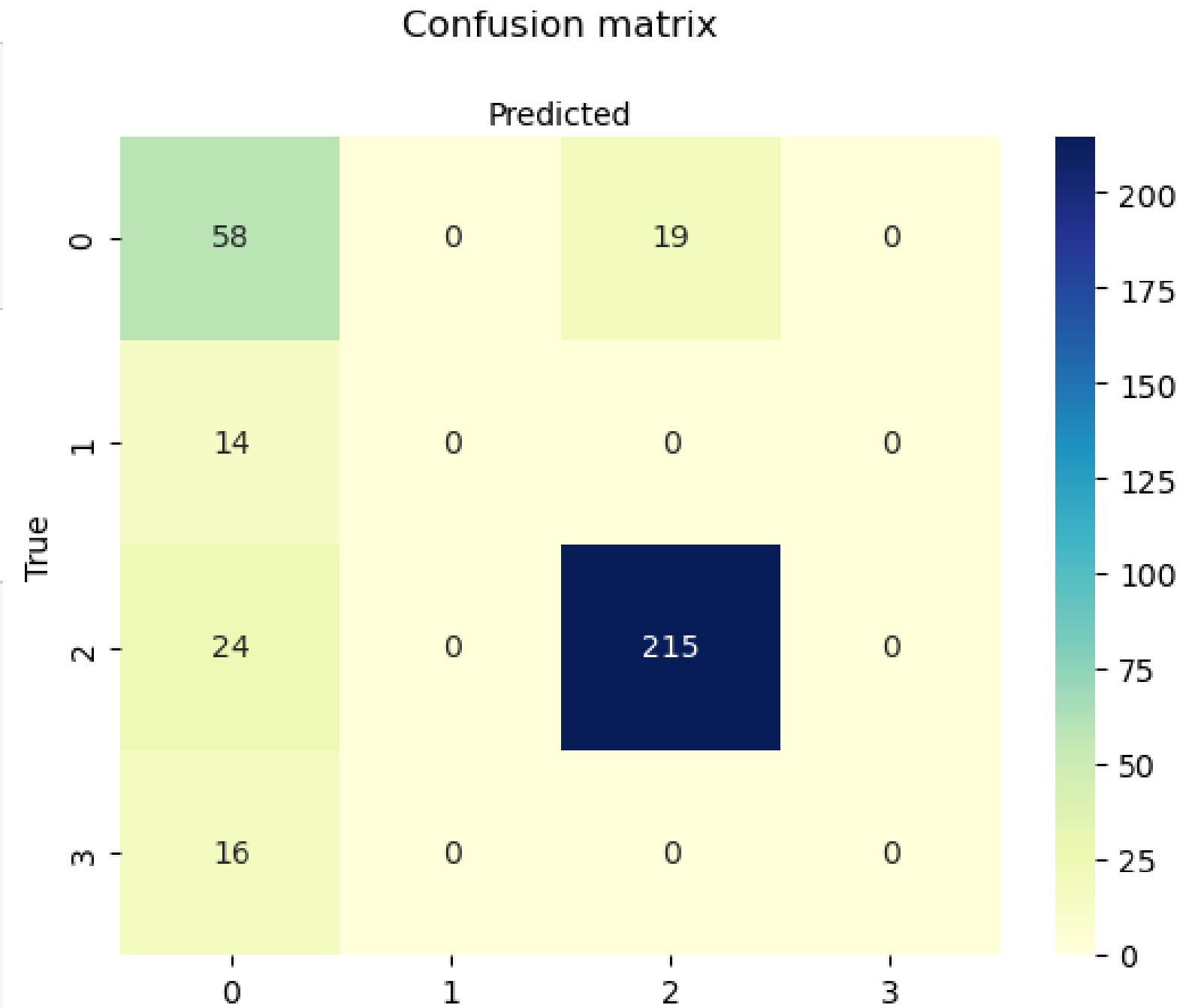
Confusion matrix

```
[[ 58   0  19   0]
 [ 14   0   0   0]
 [ 24   0 215   0]
 [ 16   0   0   0]]
```

```
In [80]: import matplotlib.pyplot as plt
import seaborn as sns
```

```
labels = [0, 1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(labels))
plt.xticks(tick_marks, labels)
plt.yticks(tick_marks, labels)
# create heatmap
sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu", fmt='g')
ax.xaxis.set_label_position("top")
plt.title('Confusion matrix', y=1.1)
plt.ylabel('True')
plt.xlabel('Predicted')
```

```
Out[80]: Text(0.5, 23.52222222222222, 'Predicted')
```



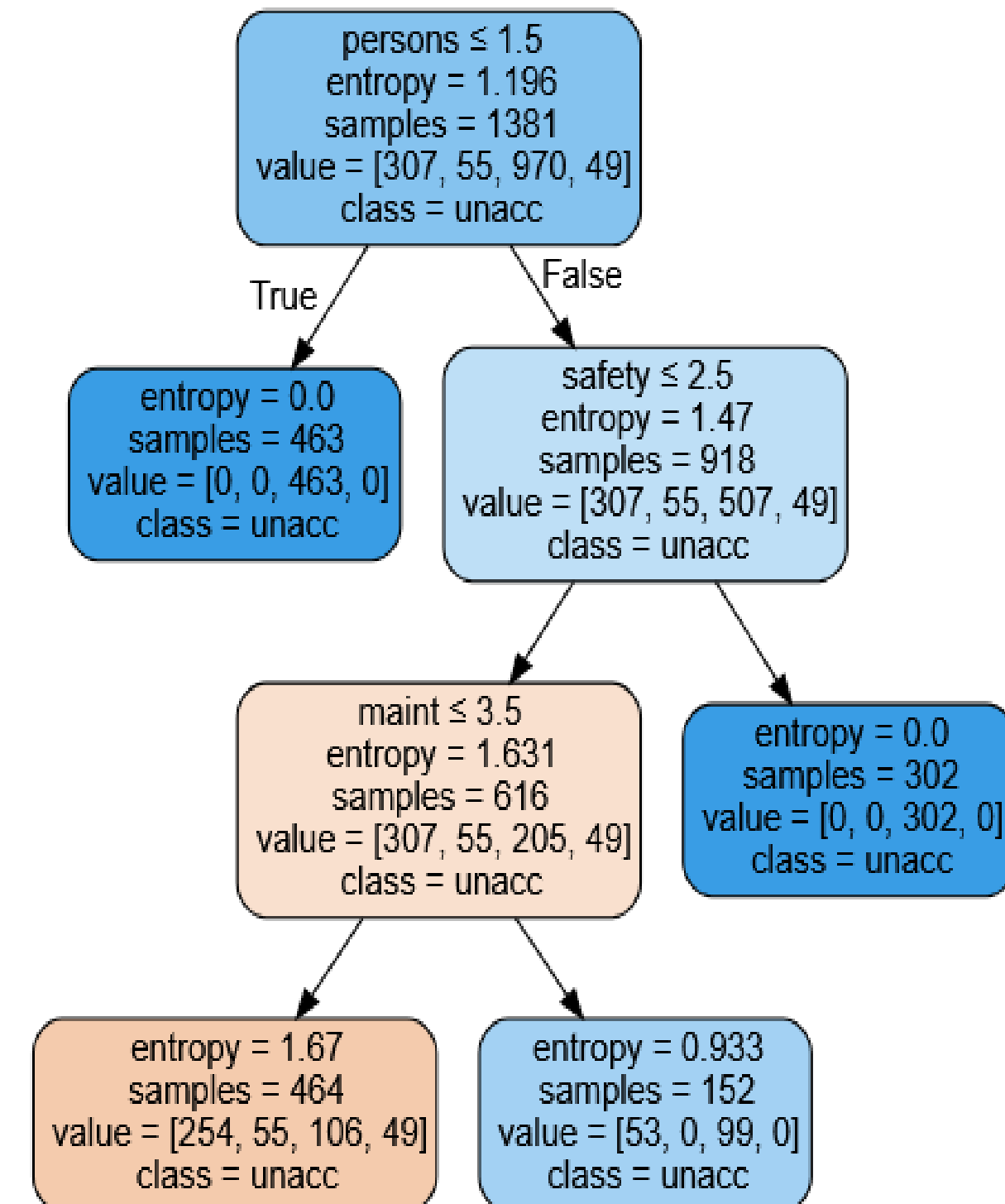
# Program Python

## Visualisasi decision-trees

```
In [77]: from sklearn import tree
import graphviz
dot_data = tree.export_graphviz(clf_en, out_file=None,
                                feature_names=X_train.columns,
                                class_names=y_train,
                                filled=True, rounded=True,
                                special_characters=True)

graph = graphviz.Source(dot_data)
graph
```


Out[77]:



Berdasarkan analisis di atas, kita dapat menyimpulkan bahwa akurasi model klasifikasi kita cukup baik. Model memprediksi label kelas.

## Latihan: Buatlah Decision Tree untuk data berikut ini

No	Kelas	Kulit Buah	Warna	Ukuran	Bau
1	Aman	Kasar	Coklat	Besar	keras
2	Aman	Kasar	Hijau	Besar	keras
3	Berbahaya	Halus	Merah	Besar	Lunak
4	Aman	Kasar	Hijau	Besar	Lunak
5	Aman	Kasar	Merah	Kecil	Keras
6	Aman	Halus	Merah	Kecil	Keras
7	Aman	Halus	Coklat	Kecil	Keras
8	Berbahaya	Kasar	Hijau	Kecil	Lunak
9	Berbahaya	Halus	Hijau	Kecil	Keras
10	Aman	Kasar	Merah	Besar	Keras
11	Aman	Halus	Coklat	Besar	Lunak
12	Berbahaya	Halus	Hijau	Kecil	Keras
13	Aman	Kasar	Merah	Kecil	Lunak
14	Berbahaya	Halus	Merah	Besar	Keras
15	Aman	Halus	Merah	Kecil	Keras
16	Berbahaya	Kasar	Hijau	Kecil	Keras



**TERIMA KASIH**