

Supervised Learning Bagian - I

16 April 2024



Materi Hari Ini

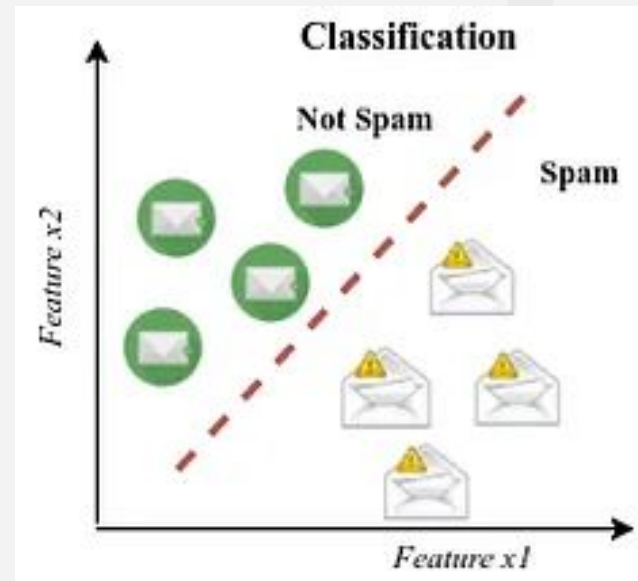
Kita akan mempelajari antara lain:

1. Pengantar Model Klasifikasi
2. K - Nearest Neighbour
3. Decision Tree (Pohon Keputusan)
4. Random Forest (Metode Ensemble)
5. Logistic Regression (Regresi Logistik)



Pengantar Model Klasifikasi

Apa dan Bagaimana



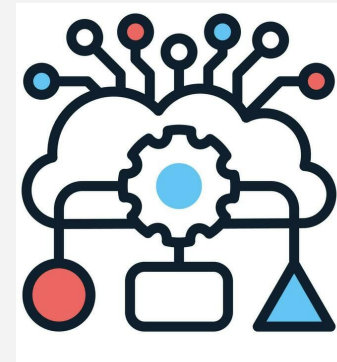
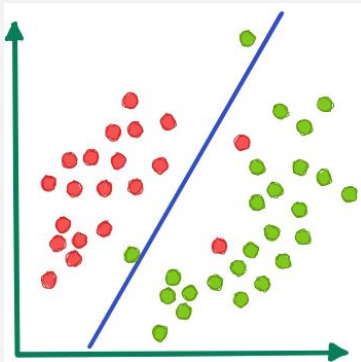
Model Klasifikasi

❖ Definisi:

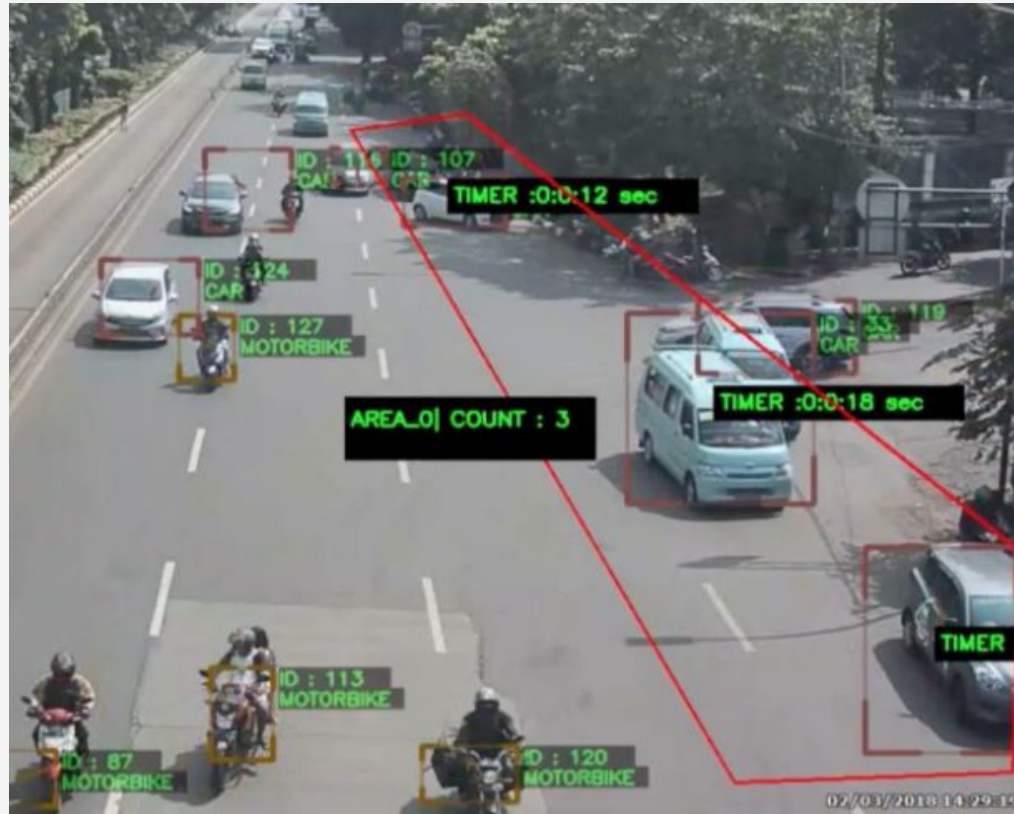
Algoritma atau metode statistik yang digunakan untuk memprediksi atau mengklasifikasikan data ke dalam beberapa kelas atau kategori yang telah ditentukan sebelumnya.

❖ Tujuan:

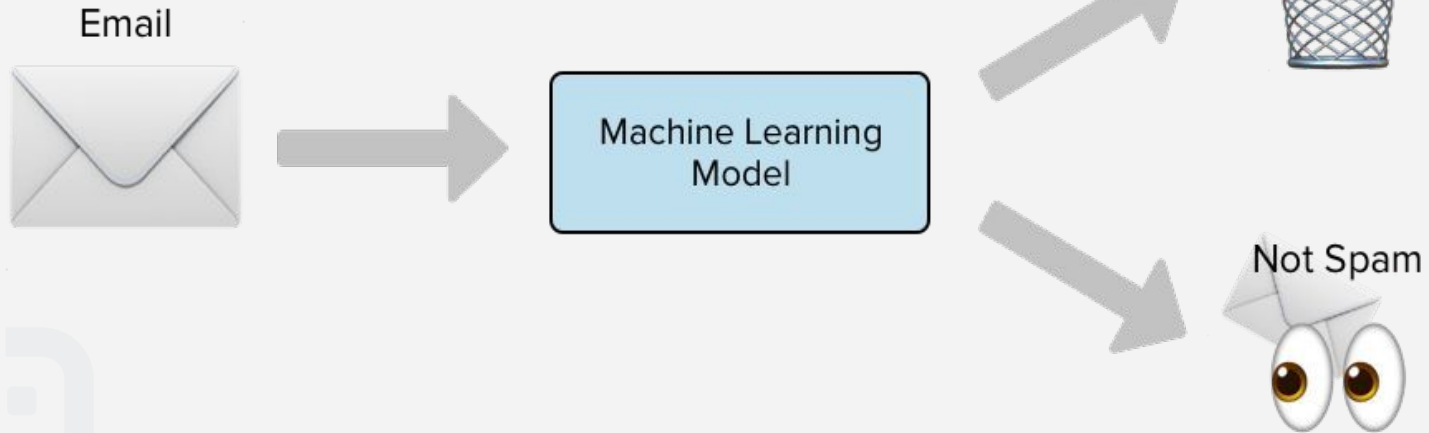
Untuk mengidentifikasi pola atau hubungan antara fitur-fitur input dengan kelas-kelas output, sehingga model dapat digunakan untuk memprediksi kelas dari data baru yang belum pernah dilihat sebelumnya.



Kasus Pengklasifikasian



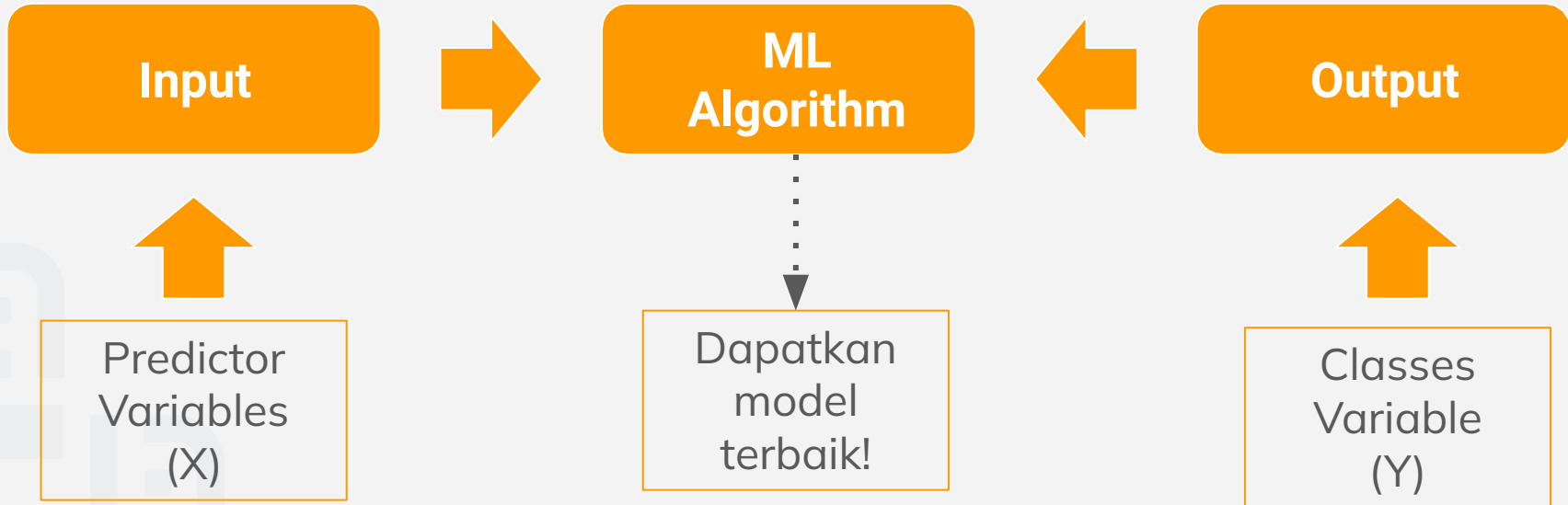
Kasus Pengklasifikasian



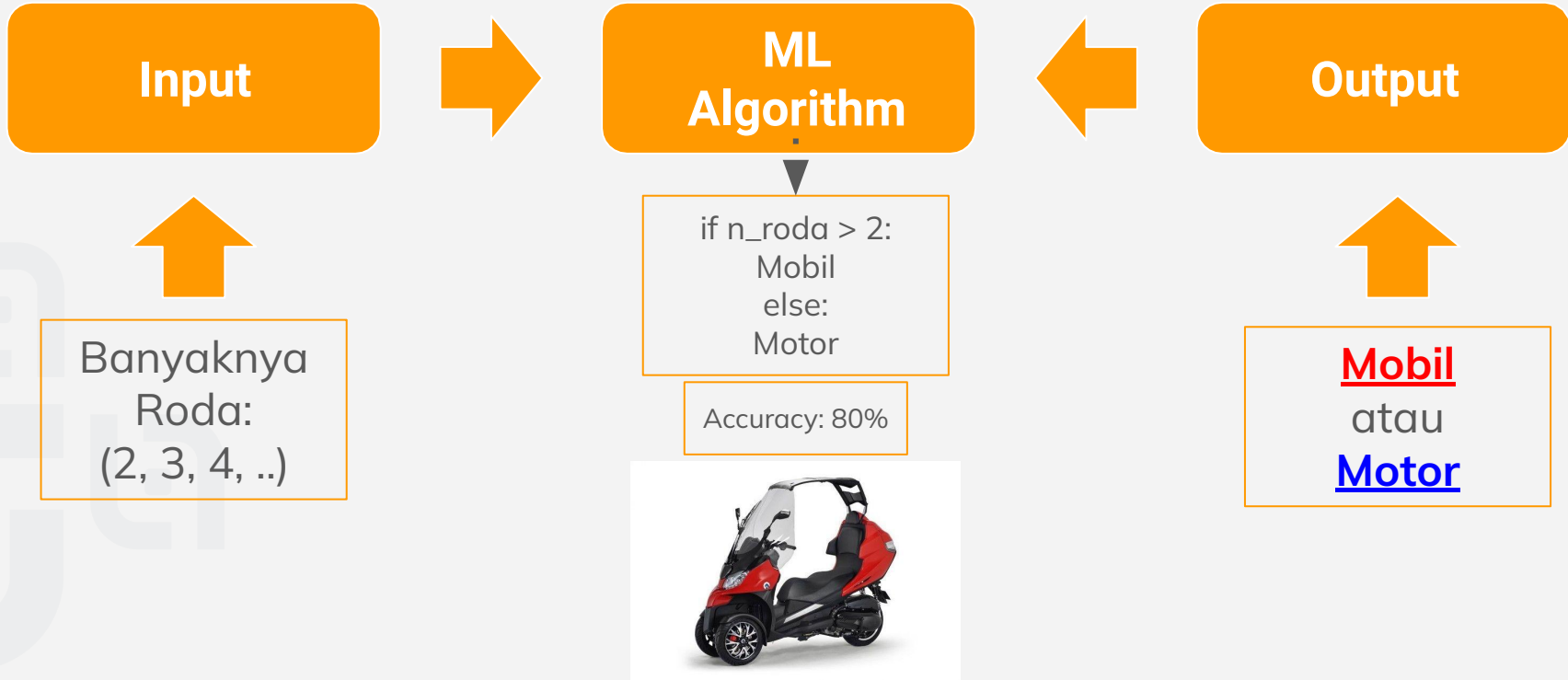
Kasus Pengklasifikasian



Cara Kerja Model Pengklasifikasian

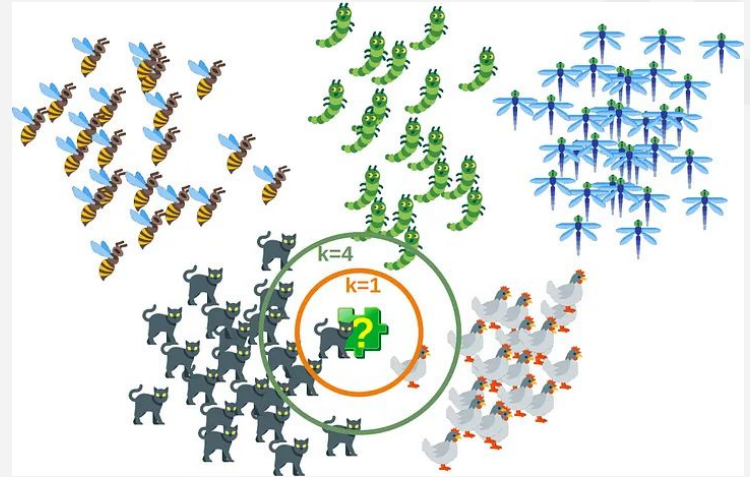


Cara Kerja Model Pengklasifikasian



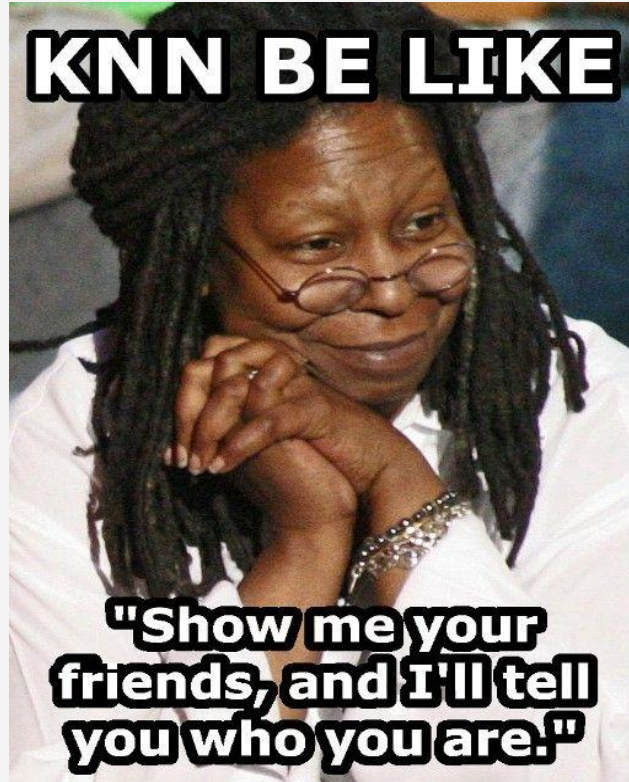
k - Nearest Neighbour (k-NN)

Apa dan Bagaimana



Source: [Link](#)

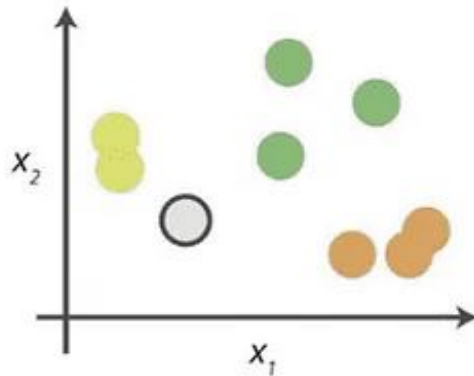
Ide Utama



Cara Kerja k-NN

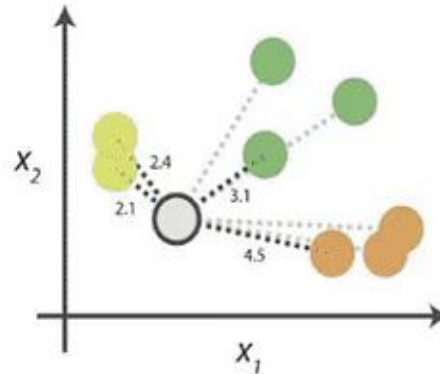


0. Look at the data



Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances











Start by calculating the distances between the grey point and all other points.

Cara Kerja k-NN









2. Find neighbours

Point Distance		
 ... 	2.1	→ 1st NN
 ... 	2.4	→ 2nd NN
 ... 	3.1	→ 3rd NN
 ... 	4.5	→ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

Class	# of votes	
	2	→ Class  wins the vote! Point  is therefore predicted to be of class  .
	1	
	1	

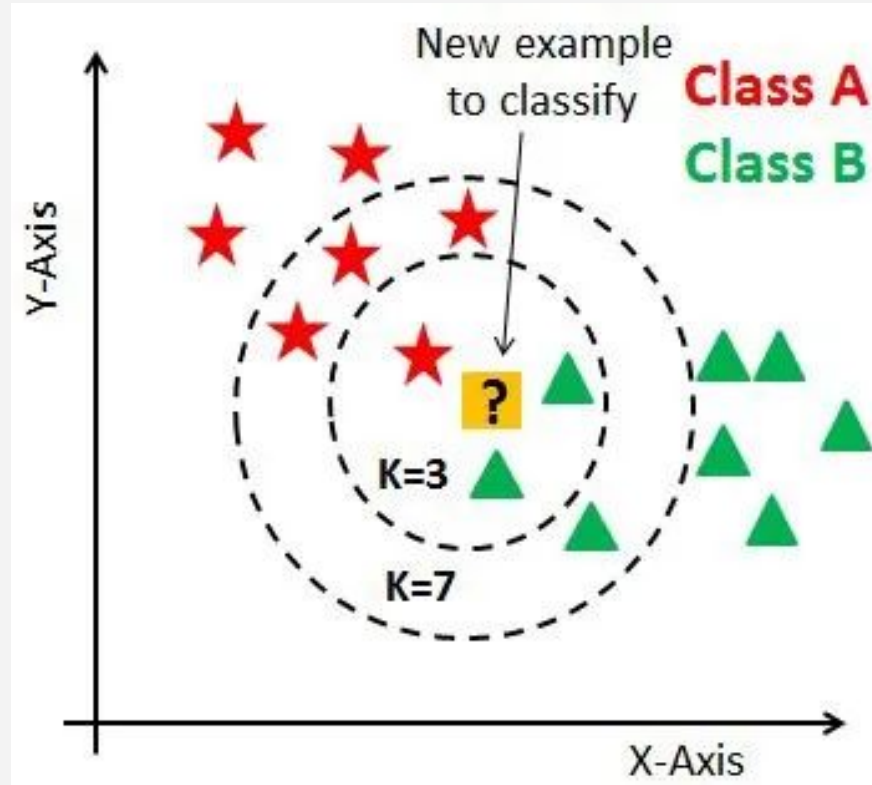
Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Algoritma k-NN



1. Langkah-1: Pilih jumlah K dari tetangga
2. Langkah-2: Hitung jarak Euclidean dari K jumlah tetangga
3. Langkah-3: Ambil K tetangga terdekat sesuai dengan jarak Euclidean yang dihitung.
4. Langkah-4: Di antara K tetangga ini, hitung jumlah titik data dalam setiap kategori.
5. Langkah-5: Tetapkan titik data baru ke kategori tersebut di mana jumlah tetangga maksimum.
6. Langkah-6: Model sudah siap.

Contoh Kasus 2D



Pengukuran Jarak (Distance): Kasus Data Continuous

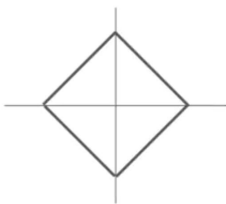


- ❑ Jarak Euclidean: didasarkan pada jarak garis lurus antara dua titik dan dihitung menggunakan teorema Pythagoras.
- ❑ Jarak Manhattan: didasarkan pada jarak yang diukur sepanjang sumbu pada sudut kanan.

K-Nearest Neighbors: Distance Metric

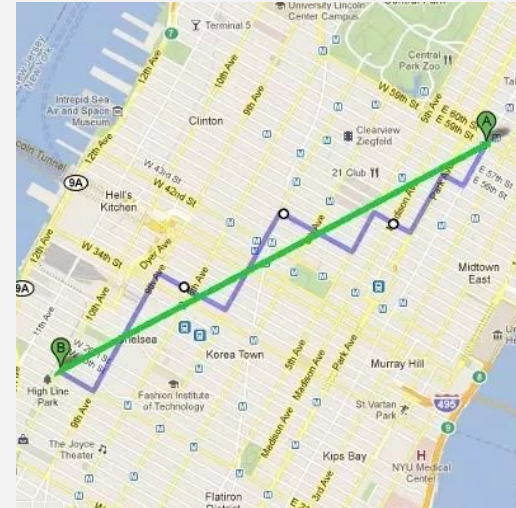
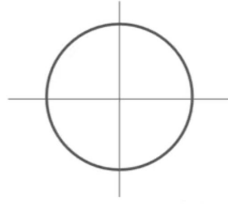
L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$



Pengukuran Jarak (Distance): Kasus Data Kategorik



- ❑ **Jarak Hamming:** Digunakan ketika fitur-fitur yang diukur adalah biner, seperti pada data kategorik atau data yang di-encode dalam bentuk biner.
- ❑ **Kelebihan/Keterbatasan:** Memiliki Kemampuannya untuk mengukur perbedaan pada data kategorik dengan baik. Keterbatasannya adalah tidak mampu menangani data numerik atau data dengan dimensi yang tinggi dengan baik karena hanya memperhitungkan perbedaan biner pada posisi yang sama.

Hamming Distance

$$D_H = \sum_{i=1}^k |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

Pentingnya Data Scaling Pada k-NN



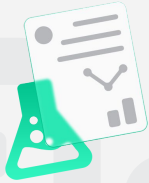
1. Algoritma KNN mengukur jarak antara titik data menggunakan metrik jarak seperti jarak Euclidean atau jarak Manhattan.
2. Fitur-fitur dengan rentang nilai yang besar akan memiliki pengaruh yang lebih besar dalam perhitungan jarak daripada fitur dengan rentang nilai yang lebih kecil.
3. Tanpa penskalaan, KNN dapat memberikan bobot yang tidak proporsional terhadap fitur-fitur dengan rentang nilai yang berbeda, sehingga mempengaruhi hasil prediksi.

Pentingnya Data Scaling Pada k-NN



4. Melakukan penskalaan memastikan bahwa setiap fitur memiliki rentang nilai yang serupa, memastikan bahwa jarak antara titik data dihitung dengan benar dan tidak didominasi oleh fitur-fitur tertentu yang memiliki skala yang besar.
5. Penskalaan membantu menjaga stabilitas dan konvergensi algoritma KNN, serta mencegah fitur-fitur dengan skala yang besar mendominasi perhitungan dan menghasilkan model yang tidak optimal.
6. Penskalaan (Scaling) adalah langkah penting dalam pra-pemrosesan data saat menggunakan KNN, membantu meningkatkan kinerja dan keandalan model.

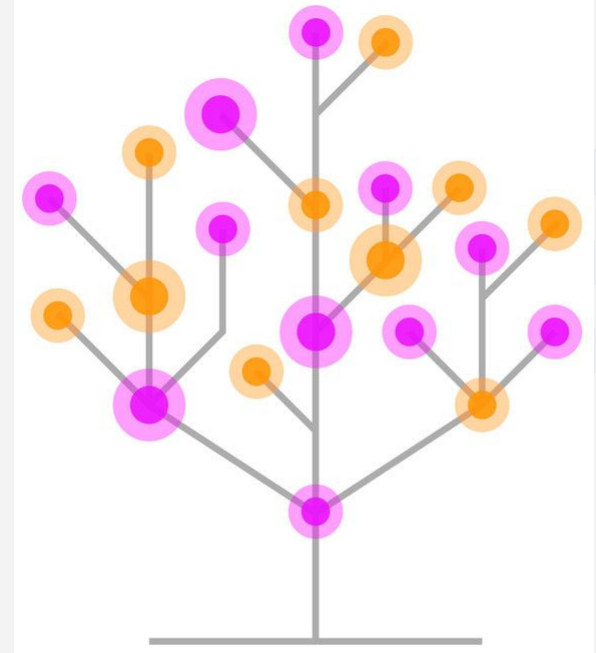
Rangkuman Metode k-NN



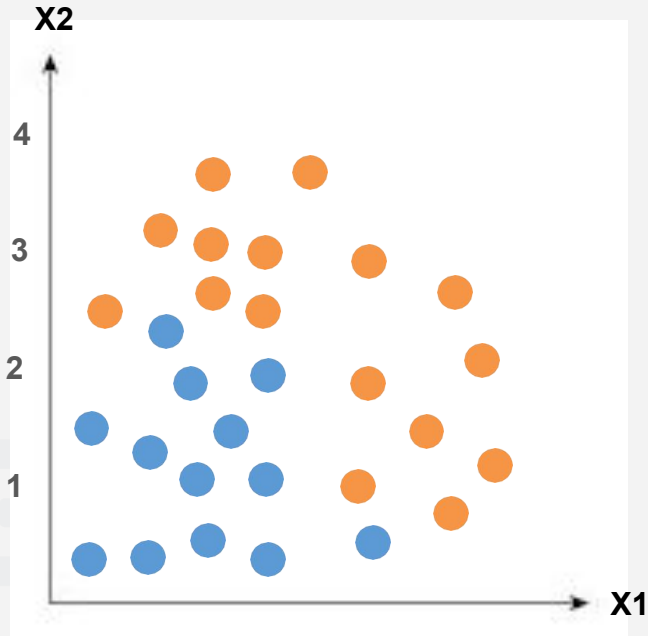
- ❑ “k” adalah jumlah tetangga yang akan kita tentukan
- ❑ Scaling sangat penting!
- ❑ “k”: harus ganjil
- ❑ Jika kita punya binary features kita bisa gunakan Hamming Distance
- ❑ Voting bisa diboboti berdasarkan jarak setiap tetangga
- ❑ Tidak bisa untuk dataset yang imbalance

Decision Tree (Pohon Keputusan)

Apa dan Bagaimana



Ide Dasar



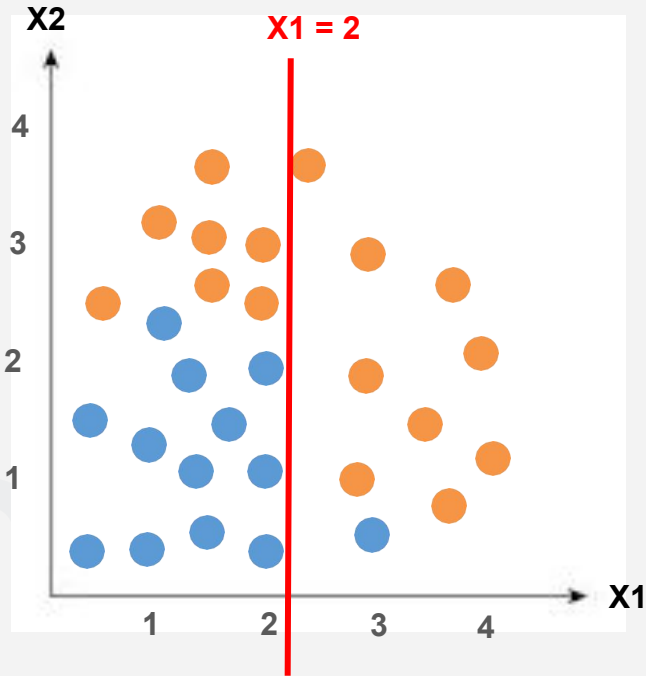
● 16 obs.

● 13 obs.

- ❖ Temukan pemisah terbaik antara ● & ●
- ❖ Pemisah terbaik adalah yang menghasilkan elemen paling homogen di setiap kelas.



Ide Dasar : Splitting



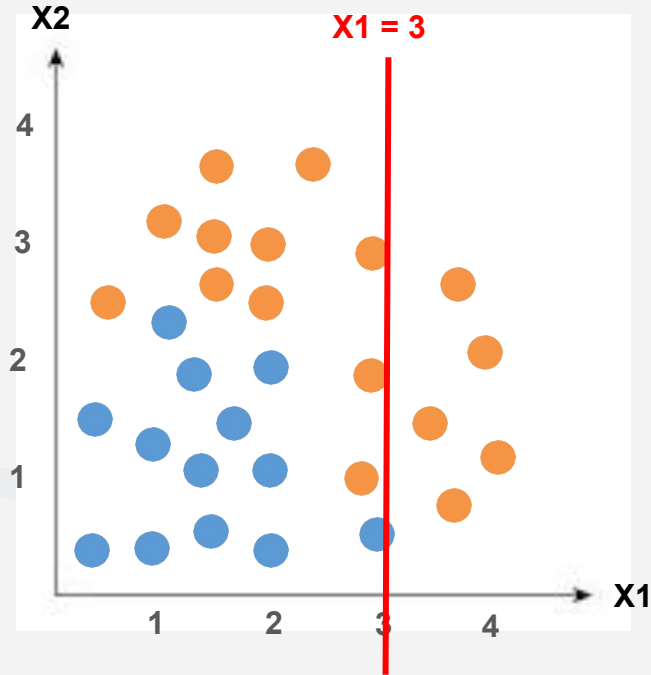
- ❖ Splitting at $X1 = 2$
- ❖ Split to 2 classes

Class $X1 < 2$ ● 7 obs.
 ● 12 obs.

Class $X1 > 2$ ● 9 obs.
 ● 1 obs.

Seberapa baik pemisahan (split) pada gambar disamping?

Ide Dasar : Splitting



- ❖ Splitting at $X1 = 3$
- ❖ Split to 2 classes

Class $X1 < 3$ ● 11 obs.
 ● 13 obs.

Class $X1 > 3$ ● 5 obs.
 ● 0 obs.

Apakah pemisahan (split) pada gambar lebih baik dari sebelumnya?

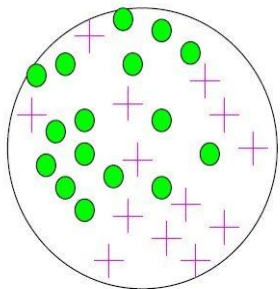
Entropy: Mengukur Seberapa Baik Pemisahan



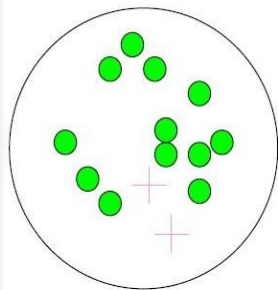
- ❑ Diberikan Dataset D , terdiri dari 2 kelas: YES dan NO
- ❑ Proporsi dari YES adalah p , maka proporsi dari NO adalah $(1-p)$

$$E(D) = -p \log_2(p) - (1-p) \log_2(1-p)$$

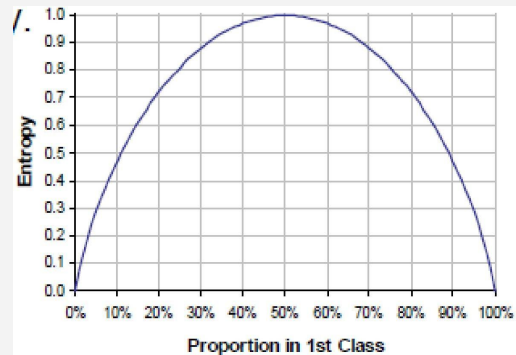
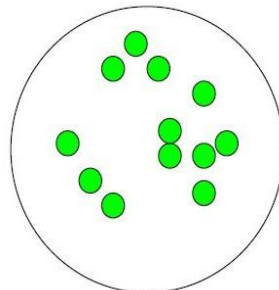
Very impure group



Less impure



Minimum impurity

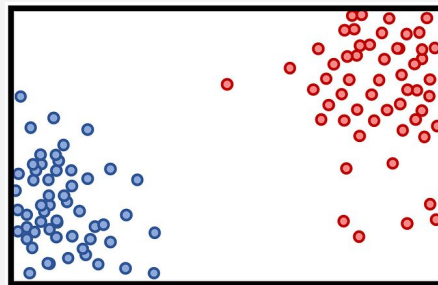


Entropy: Mengukur Seberapa Baik Pemisahan

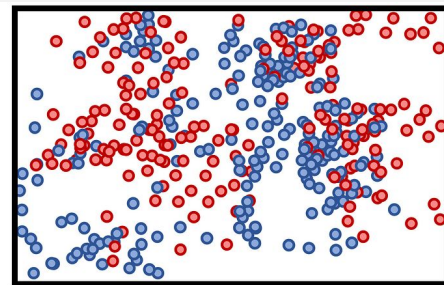


- ❑ **Konsep Dasar**: Entropy adalah ukuran ketidakpastian atau kekacauan dalam sebuah set data. Dalam decision tree, entropy digunakan untuk mengukur seberapa homogen atau seberapa campur aduk sebuah set data.
- ❑ **Interpretasi**: Entropy memiliki nilai antara 0 dan 1. Nilai 0 menunjukkan bahwa semua sampel dalam set data termasuk ke dalam satu kelas, sementara nilai 1 menunjukkan bahwa setiap kelas dalam set data memiliki proporsi yang sama.

$$H = - \sum_i p_i (\log_2 p_i)$$



Low Entropy



High Entropy

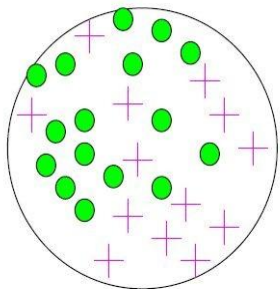
Entropy: Mengukur Seberapa Baik Pemisahan



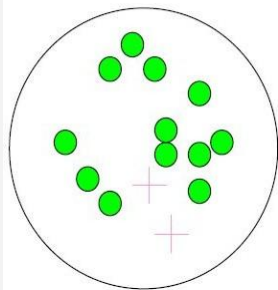
- ❑ Diberikan Dataset D , terdiri dari 2 kelas: YES dan NO
- ❑ Proporsi dari YES adalah p , maka proporsi dari NO adalah $(1-p)$

$$E(D) = -p \log_2(p) - (1-p) \log_2(1-p)$$

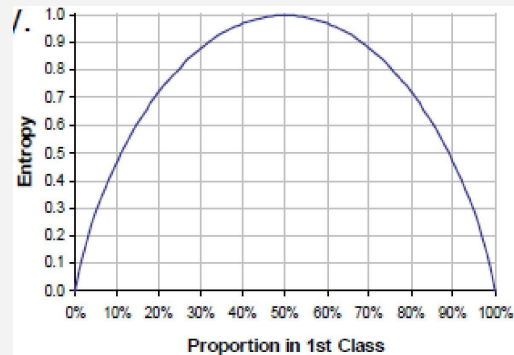
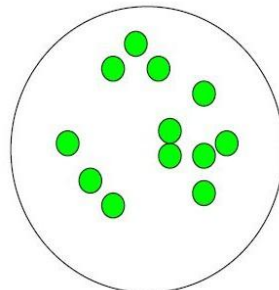
Very impure group



Less impure



Minimum impurity



Information Gain (IG) :

Evaluasi Kontribusi Fitur/Variabel



- ❑ Diberikan Dataset D , dipisahkan menjadi beberapa grup: D_1 , D_2 , ..., D_k , berdasarkan fitur prediktor V
- ❑ Untuk setiap D_i , dapat dihitung dengan formula berikut:

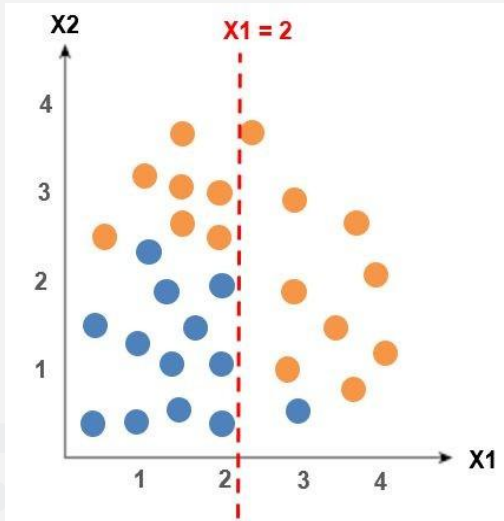
$$IG(D, V) = E(D) - \sum_{i=1}^k \frac{|D_i|}{|D|} E(D_i)$$



Ide Dasar : Better Splitting



Pemisahan yang lebih baik?

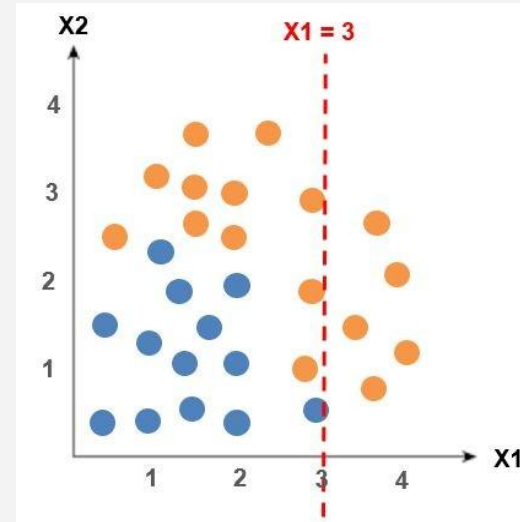


● 7 obs.

● 12 obs.

● 9 obs.

● 1 obs.



● 11 obs.

● 13 obs.

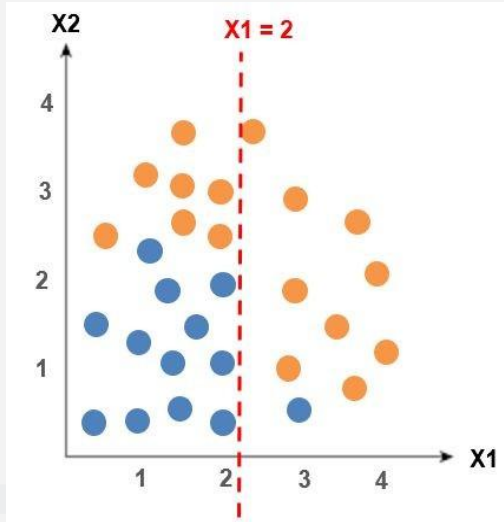
● 5 obs.

● 0 obs.



Ide Dasar : Better Splitting

Pemisahan yang lebih baik?



● 7 obs.

● 12 obs.

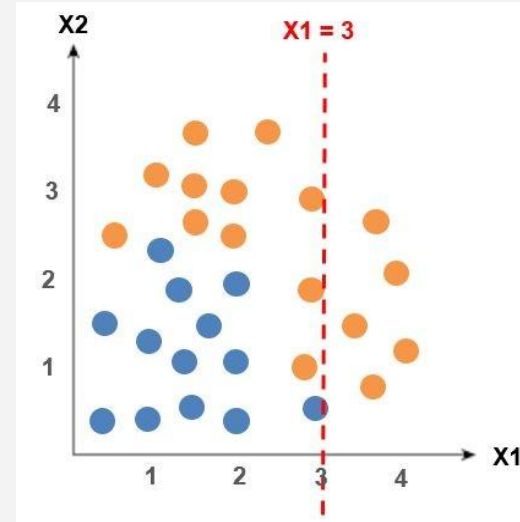
$E = 0.95$

$IG = 0.21$

● 9 obs.

● 1 obs.

$E = 0.47$



● 11 obs.

● 13 obs.

$E = 0.99$

$IG = 0.17$

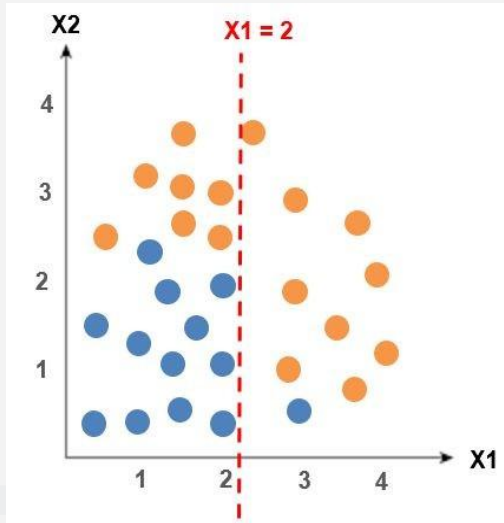
● 5 obs.

● 0 obs.

$E = 0$

Ide Dasar : Entropy & Information Gain

Pemisahan yang lebih baik?



● 7 obs.

● 9 obs.

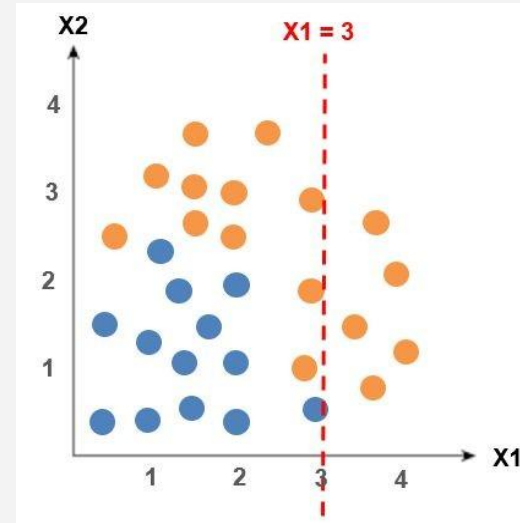
● 12 obs.

● 1 obs.

$E = 0.95$

$E = 0.47$

$IG = 0.21$



● 11 obs.

● 5 obs.

● 13 obs.

● 0 obs.

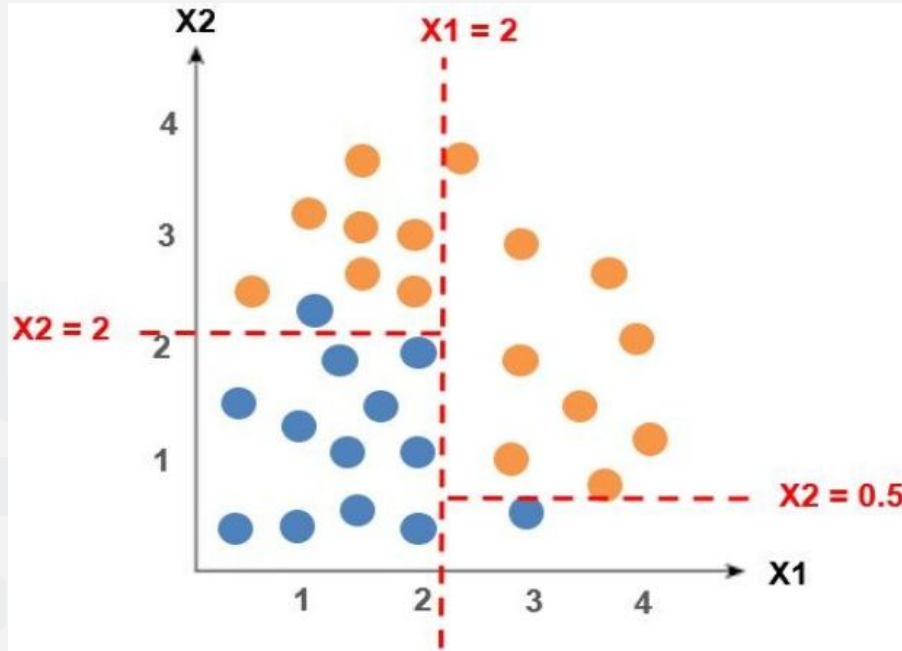
$E = 0.99$

$E = 0$

$IG = 0.17$



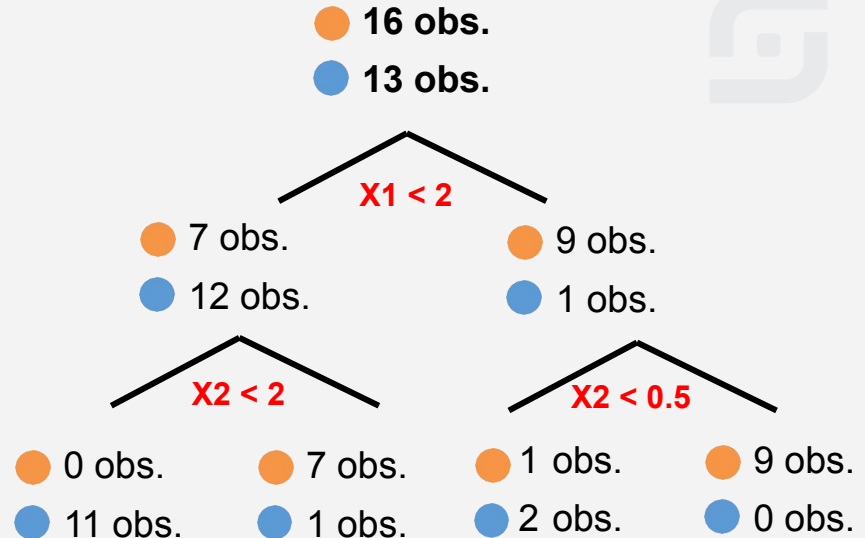
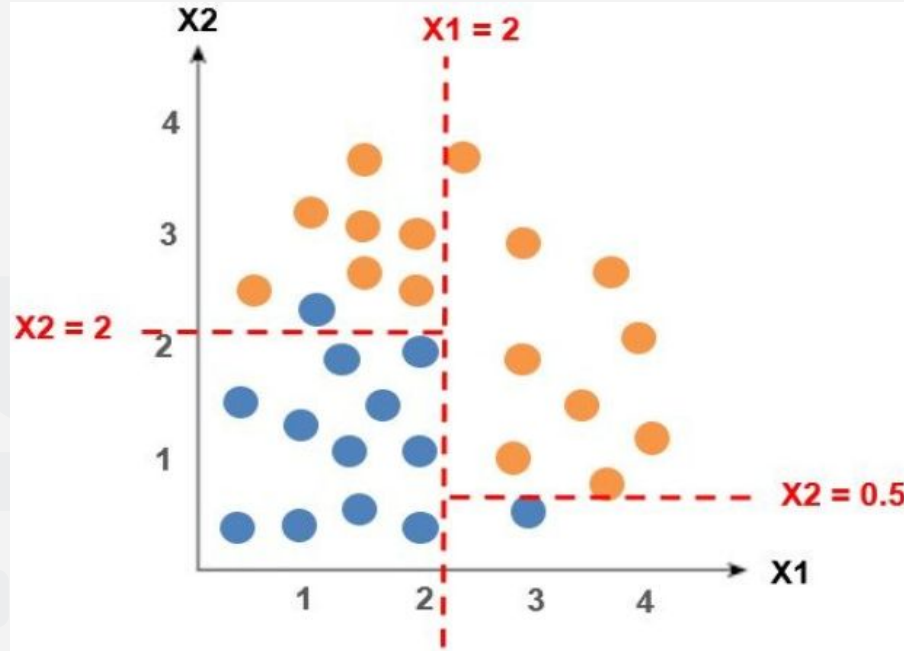
Ide Dasar : Continue Splitting



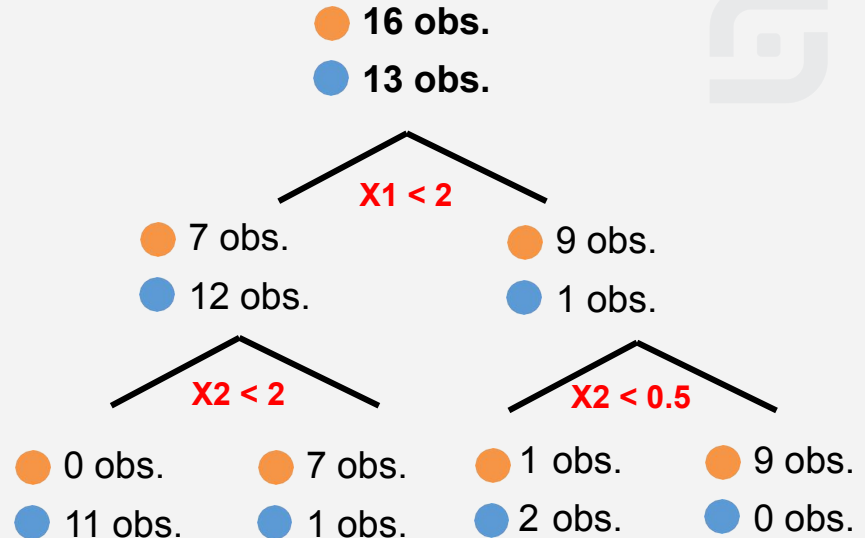
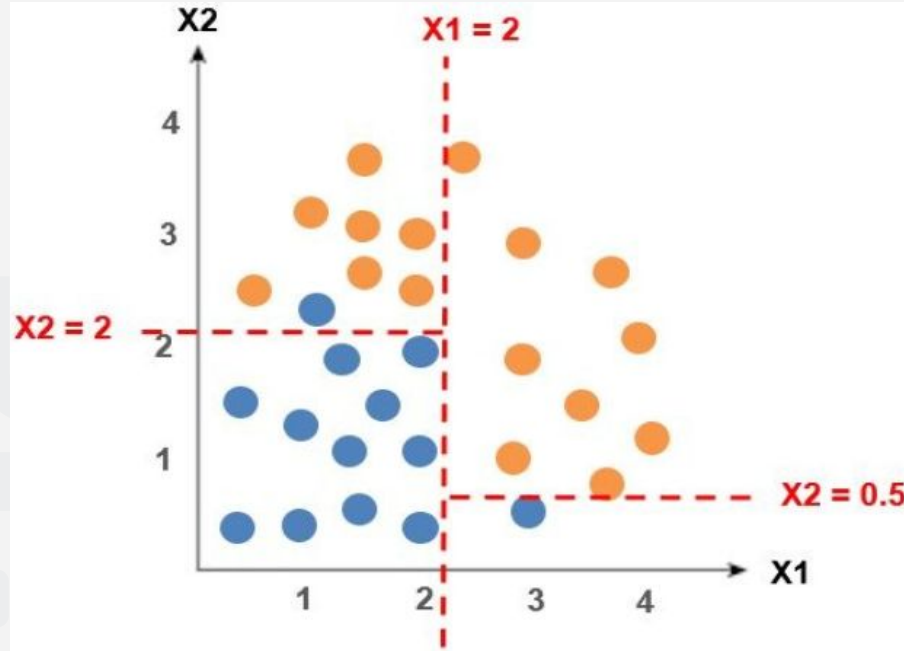
Continue splitting on each group.

- ❖ At $X_2 = 2$ for class $X_1 < 2$
- ❖ At $X_2 = 0.5$ for class $X_1 > 2$

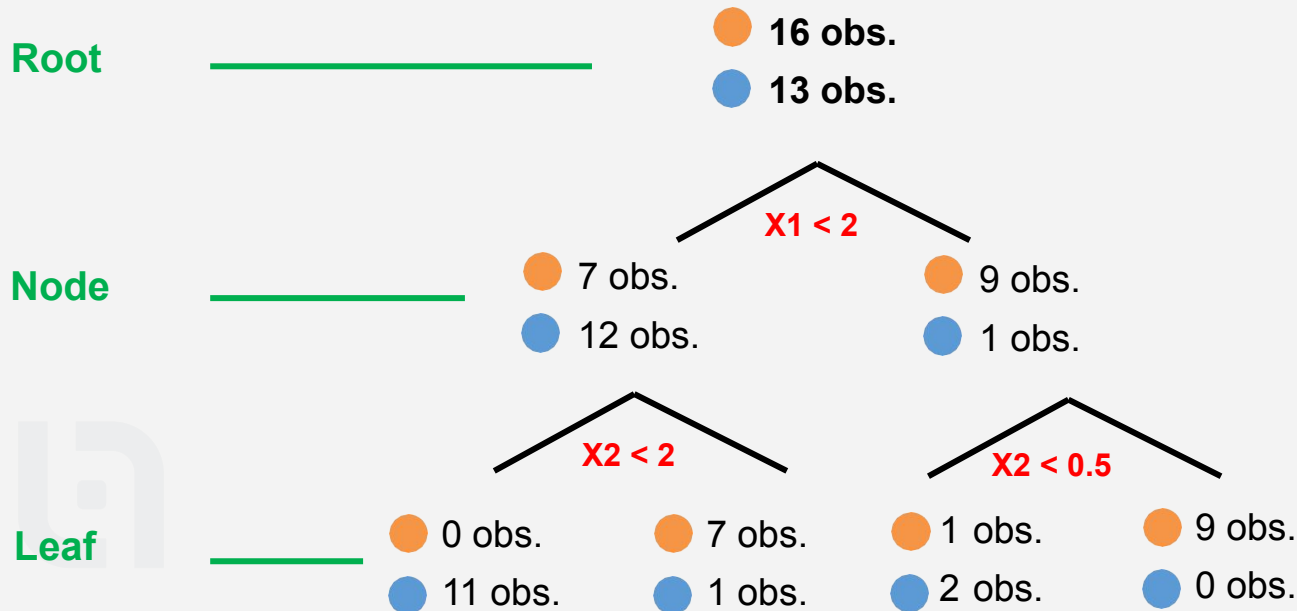
Ide Dasar : Continue Splitting



Ide Dasar : Continue Splitting



Decision Tree: Terminologi



Decision Tree: Algoritma



Lakukan 3 langkah berikut untuk setiap Node tunggal dan hasil pemisahannya:

1. Langkah-1: Temukan pemisah terbaik pada setiap variabel
2. Langkah-2: Pilih variabel terbaik untuk pemisahan
3. Langkah-3: Lakukan pemisahan berdasarkan hasil pada Langkah-2. Periksa apakah pemisahan harus dihentikan.

Decision Tree: Kriteria Stop-Splitting

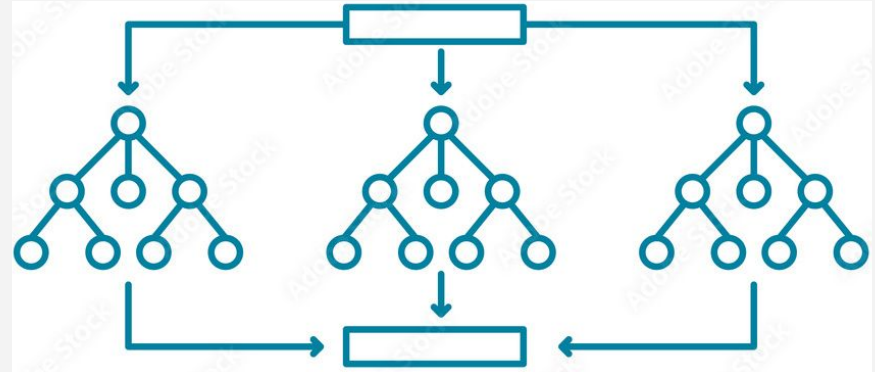


Pemisahan (Splitting) akan berhenti jika salah satu dari kondisi di bawah ini terpenuhi:

1. Node hanya berisi 1 kelas variabel target
2. Jumlah observasi dalam sebuah node sebelum pemisahan kurang dari jumlah yang telah ditentukan sebelumnya (min_split)
3. Jumlah observasi dalam sebuah node setelah pemisahan kurang dari jumlah yang telah ditentukan sebelumnya (min_bucket)
4. Kedalaman pohon telah mencapai maksimum yang ditentukan

Random Forest (Metode Ensemble)

Penggunaanya dalam
Kasus Klasifikasi



Source: [Link](#)

Decision Tree To Random Forest



1. Decision Tree (Pohon Keputusan)

Algoritma klasifikasi yang membagi ruang fitur berdasarkan aturan keputusan. Memisahkan data ke dalam bentuk struktur pohon.

2. Kelemahan Decision Tree

- a. Rentan terhadap overfitting pada data pelatihan.
- b. Sensitif terhadap variasi kecil dalam data.



Decision Tree To Random Forest



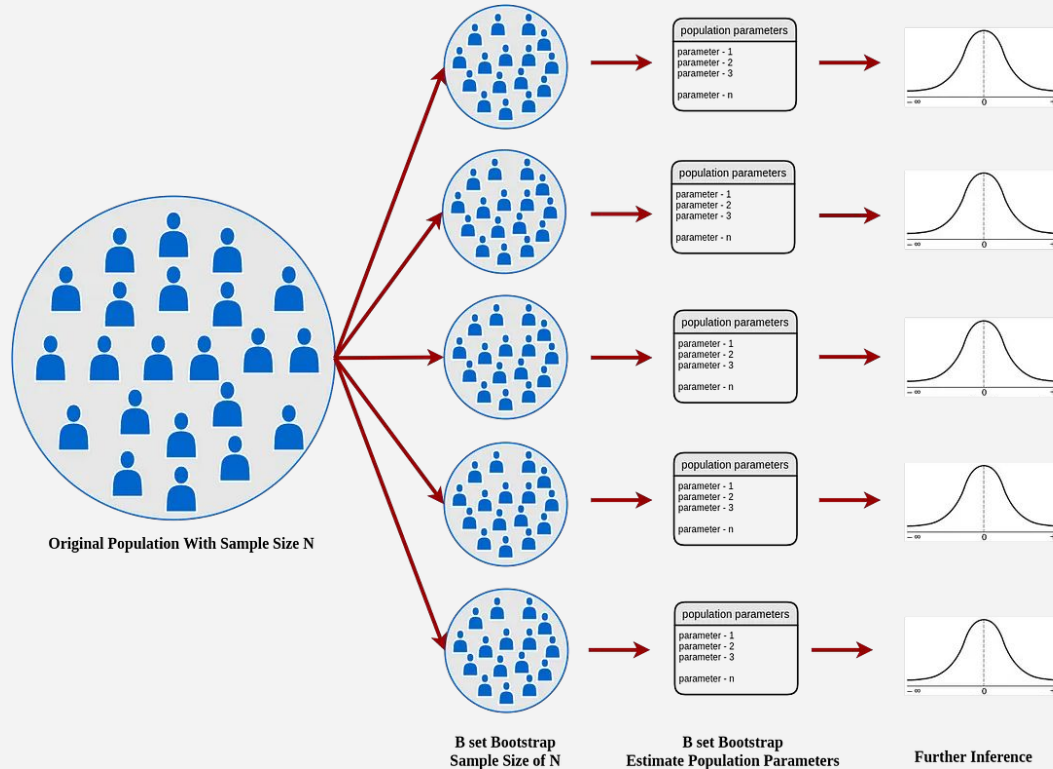
1. Random Forest (Hutan Acak)

Kombinasi dari banyak pohon keputusan yang dibangun secara acak. Setiap pohon keputusan dibangun pada subset acak dari data pelatihan.

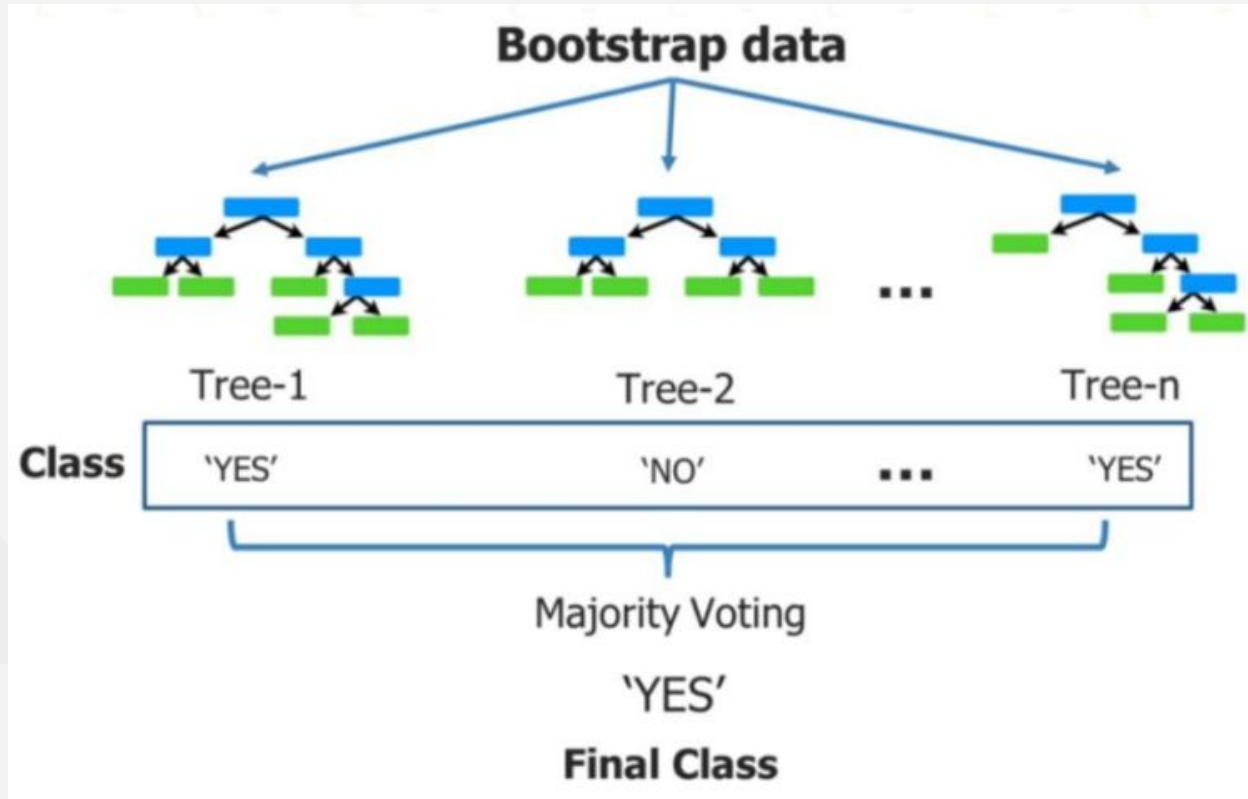
2. Mengapa Random Forest Efektif?

- a. Mengurangi overfitting dengan menggabungkan prediksi dari banyak pohon.
- b. Meningkatkan kinerja dengan mengurangi varian prediksi.

Random Forest: Sebagai Metode Bootstrap (Ensemble)



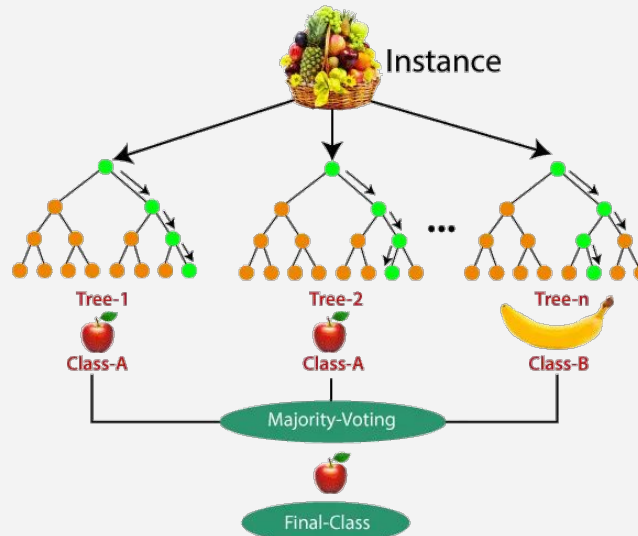
Random Forest: Sebagai Metode Bootstrap (Ensemble)



Random Forest: Sebagai Metode Bootstrap (Ensemble)

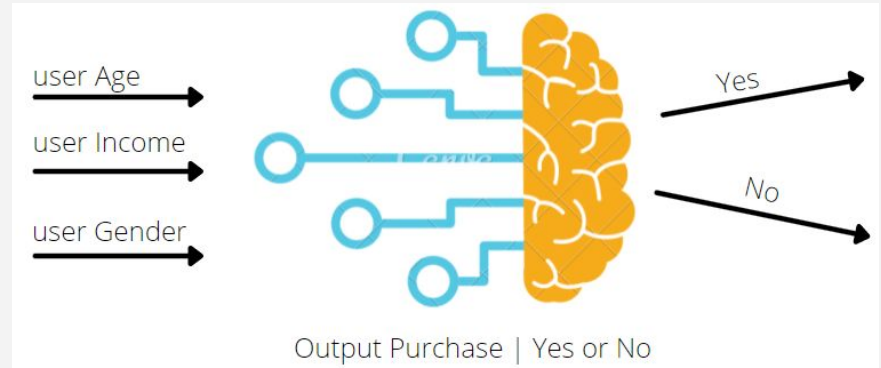
Kelebihan Random Forest

- a. Random Forest adalah pengembangan dari Decision Tree yang lebih kuat dan lebih stabil.
- b. Cocok untuk dataset besar dan kompleks.
- c. Dapat menangani keduanya masalah klasifikasi dan regresi.



Logistic Regression (Regresi Logistik)

Apa dan Bagaimana



Regresi Linier Untuk Klasifikasi?



$$y = \alpha + \beta x + \varepsilon$$

❑ **Regresi Linier Sederhana** dimana:

y : Variabel Target (Response) [continuous]

x : Variabel Predictor

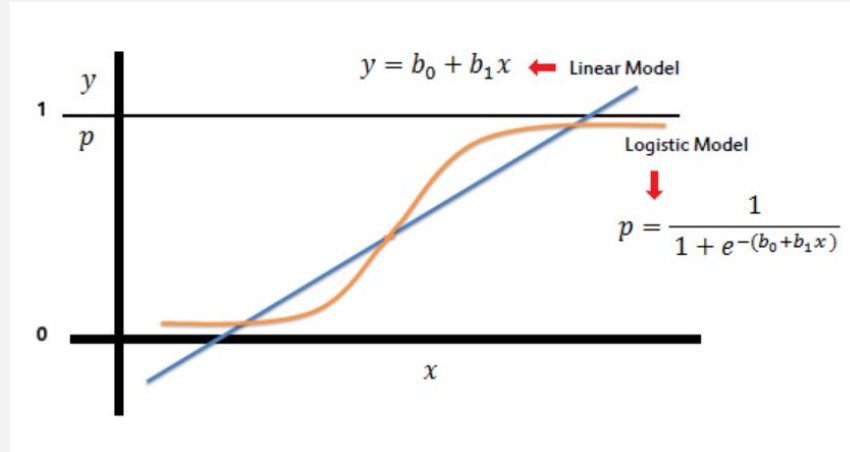
❑ **Kenapa Regresi Linier Sederhana tidak bisa digunakan untuk Variabel Target binary?**

- Model harus menghasilkan probabilitas prediksi yang berada di antara 0 dan 1.
 - Model linear menghasilkan respons yang bervariasi dari $-\infty$ hingga $+\infty$.
 - Target (y) tidak terdistribusi secara normal.
 - Variabilitas dari target (y) tidak konstan.

Logistic Regression: Pendekatan Matematis



- ❑ Merupakan algoritma klasifikasi ML yang digunakan untuk memprediksi probabilitas variabel target kategorik
- ❑ Variabel target kategorik adalah variabel biner yang berisi data yang dikodekan sebagai 1 (untuk kasus benar / ya / sukses) atau 0 (untuk kasus salah / tidak / gagal).
- ❑ Pikirkan sebagai kasus khusus dari regresi linear dimana targetnya adalah log dari odds.
- ❑ Model Regresi Logistik memprediksi $P(Y=1)$ dengan menyesuaikan data ke fungsi logit.



Logistic Regression: Model Matematis

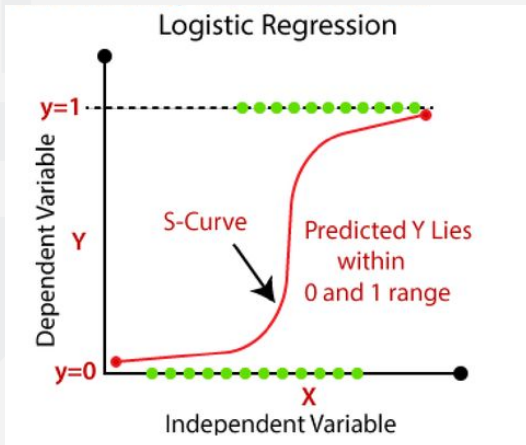
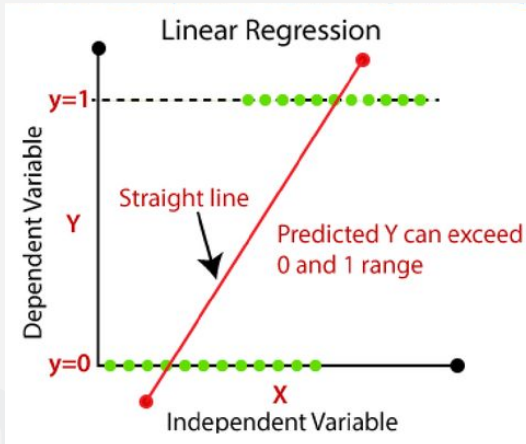


- ❑ Logit dalam Regresi Logistik adalah transformasi matematis yang digunakan untuk menghubungkan variabel independen (fitur) dengan probabilitas kejadian suatu peristiwa pada variabel target.
- ❑ Logit dari probabilitas $P(Y = 1)$ (di mana Y adalah variabel target) adalah logaritma natural dari odds dari kejadian $Y = 1$ yang dinyatakan sebagai:

$$\text{logit}(P(Y = 1)) = \log \left(\frac{P(Y=1)}{1-P(Y=1)} \right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

- ❑ Logit ini kemudian digunakan untuk memodelkan hubungan antara fitur-fitur dan probabilitas kejadian $Y = 1$ menggunakan teknik regresi. Dengan kata lain, Regresi Logistik mencoba menemukan garis (atau batas keputusan) yang memisahkan dua kelas (misalnya, kelas 1 dan kelas 0) dalam ruang fitur berdasarkan logit dari probabilitas.

Linear Regression vs. Logistic Regression



Linear Regression	Logistic Regression
Linear regression is used to predict the continuous dependent variable using a given set of independent variables.	Logistic Regression is used to predict the categorical dependent variable using a given set of independent variables.
Linear Regression is used for solving Regression problem.	Logistic regression is used for solving Classification problems.
In Linear regression, we predict the value of continuous variables.	In logistic Regression, we predict the values of categorical variables.
In linear regression, we find the best fit line, by which we can easily predict the output.	In Logistic Regression, we find the S-curve by which we can classify the samples.
Least square estimation method is used for estimation of accuracy.	Maximum likelihood estimation method is used for estimation of accuracy.
The output for Linear Regression must be a continuous value, such as price, age, etc.	The output of Logistic Regression must be a Categorical value such as 0 or 1, Yes or No, etc.
In Linear regression, it is required that relationship between dependent variable and independent variable must be linear.	In Logistic regression, it is not required to have the linear relationship between the dependent and independent variable.
In linear regression, there may be collinearity between the independent variables.	In logistic regression, there should not be collinearity between the independent variable.

Implementasi

- EDA, Preprocessing, API Testing
- Hands-on Coding in Google Colab

```
31 def __init__(self, settings):
32     self.file = None
33     self.fingerprints = set()
34     self.logdupes = True
35     self.debug = debug
36     self.logger = logging.getLogger(__name__)
37     if path:
38         self.file = open(os.path.join(path, "requests.log"),
39                         "a")
40         self.fingerprints.update({os.path.join(path, "requests.log")})
41
42 @classmethod
43 def from_settings(cls, settings):
44     debug = settings.getbool("DEBUG_LOGGING")
45     return cls(job_dir(settings), debug)
46
47 def request_seen(self, request):
48     fp = self.request_fingerprint(request)
49     if fp in self.fingerprints:
50         return True
51     self.fingerprints.add(fp)
52     if self.file:
53         self.file.write(fp + os.linesep)
54
55 def request_fingerprint(self, request):
56     return request_fingerprint(request)
```





Thank You

