



UNIVERSITAS
SINGAPERBANGSA
KARAWANG

FAKULTAS ILMU KOMPUTER
SISTEM INFORMASI

UJIAN AKHIR SEMESTER (UAS)

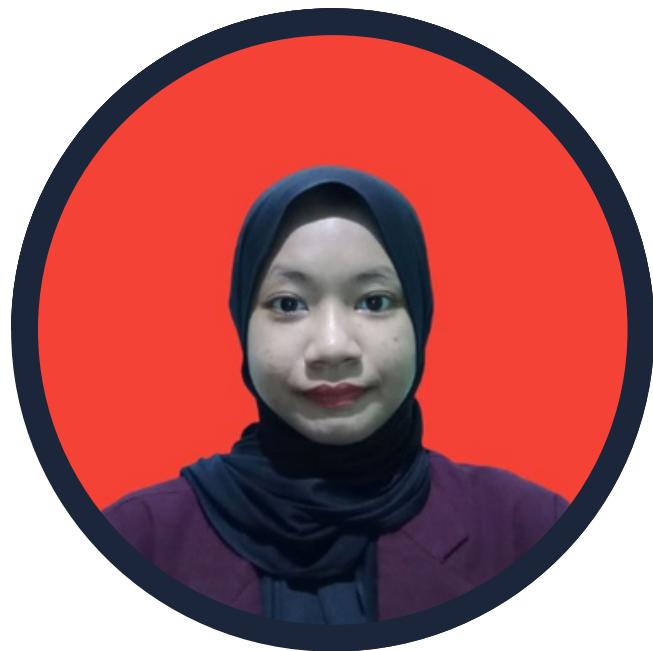
SISTEM CERDAS

DISUSUN OLEH
KELOMPOK 2

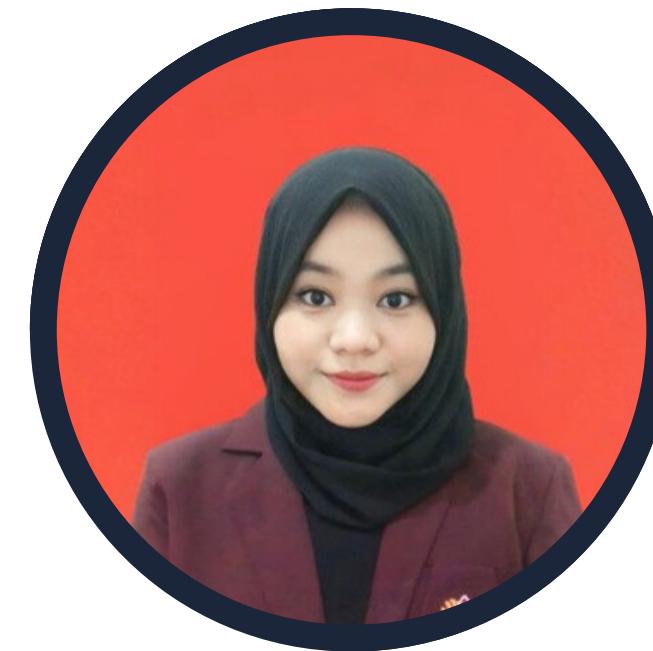
Pembimbing **Bpk. Taufik Ridwan**

13/06/2025

OUR TEAM



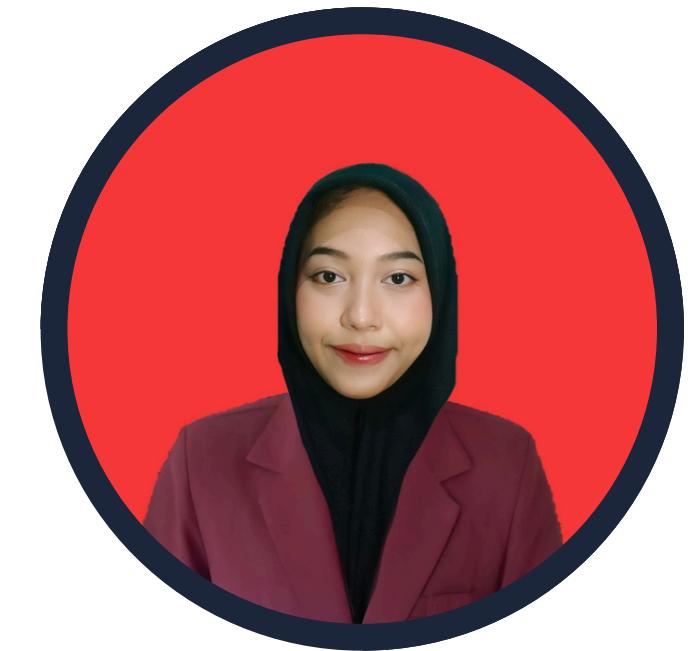
AZIZAH PAUZIAH
2210631250006



**NASYWA SYIFA
HANIFAH**
2210631250023



**ADITYA ALFAREZY
DAMANIK**
2210631250037



**VERNANDA AYU
PRASTIKA**
2210631250102

DAFTAR ISI



TUGAS 7

Algoritma Clustering Menggunakan Naive Bayes



TUGAS 9

Implementasi KMeans Clustering



TUGAS 10

Proyek Deteksi Objek Alat Pelindung Diri

TUGAS 7

“PENERAPAN ALGORITMA NAÏVE BAYES
UNTUK MEMPREDIKSI PERCERAIAN
PASANGAN DI TURKI BERDASARKAN METODE
TERAPI PASANGAN GOTTMAN”

STUDI KASUS

Perceraian adalah masalah sosial yang berdampak besar dan terus meningkat di berbagai negara, termasuk Turki. Untuk memahami faktor-faktor yang memicunya, pendekatan psikologis seperti terapi pasangan Gottman digunakan untuk menilai aspek komunikasi dan konflik dalam hubungan. Seiring perkembangan teknologi, algoritma machine learning seperti Naïve Bayes dimanfaatkan untuk memprediksi kemungkinan perceraian berdasarkan data survei.

Dalam tugas ini, algoritma tersebut diterapkan pada data dari 150 pasangan di Turki untuk mengidentifikasi pola hubungan yang berisiko tinggi terhadap perceraian serta fitur-fitur yang paling berpengaruh dalam prediksi tersebut.



PENGUMPULAN DATA



Dataset yang digunakan berasal dari Kaggle dengan nama “Divorce Prediction” yang diunggah oleh pengguna LARXEL sekitar lima tahun lalu. Dataset ini terdiri dari 170 baris dan 55 kolom, dengan 54 kolom sebagai fitur dan 1 kolom sebagai label. Sebanyak 80% data digunakan untuk training dan 20% untuk pengujian. Kaggle memberi skor Usability 10.00, yang menunjukkan bahwa dataset ini lengkap, terpercaya, dan kompatibel.

IMPORT LIBRARY



```
import os
import zipfile
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score,
import seaborn as sns
import matplotlib.pyplot as plt
```

PERSIAPAN DATASET

```
# 1. API Kaggle
os.environ['KAGGLE_USERNAME'] = 'kyozerakaramine'
os.environ['KAGGLE_KEY'] = 'fcc960787bece7f84018078f4cbaf4b6'

# 2. Download dataset dari Kaggle
!kaggle datasets download -d andrewmvd/divorce-prediction

# 3. Mendapatkan ZIP dan Ekstrak file ZIP tersebut
with zipfile.ZipFile('divorce-prediction.zip', 'r') as zip_ref:
    print("Isi ZIP:")
    zip_ref.printdir()
```

PEMBACAAN DAN TRAINING MODEL

```
● ● ●

# 4. Baca dataset
df = pd.read_csv('divorce_data/divorce_data.csv', delimiter=';')

# 5. Pisahkan fitur dan label
X = df.drop('Divorce', axis=1)
y = df['Divorce']

# 6. Split data menjadi training dan test yang dimana untuk test 80% dan uji 20%
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# 7. Latih model menggunakan algoritma Naive Bayes
model = GaussianNB()
model.fit(X_train, y_train)
```

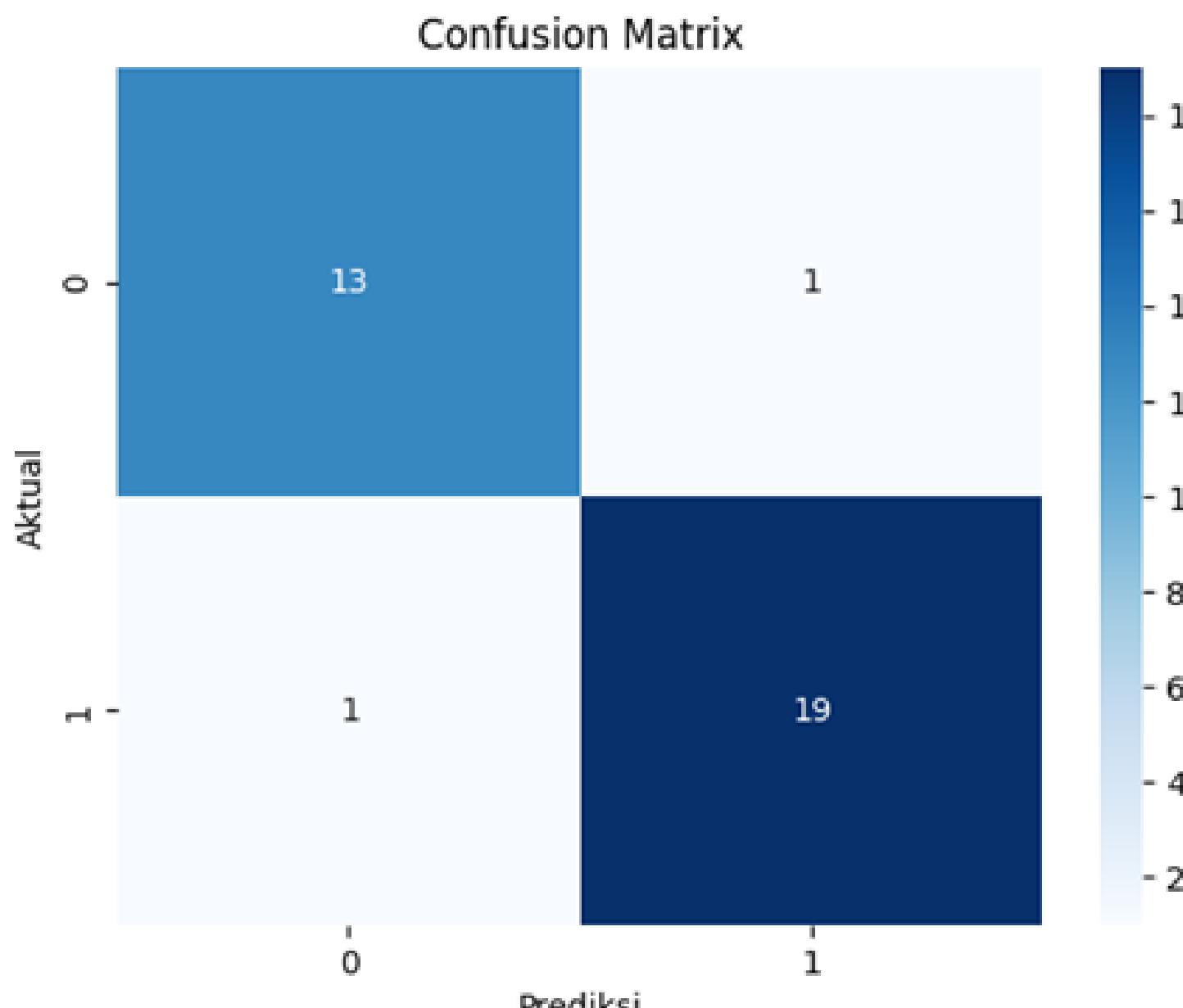
EVALUASI DAN CONFUSION MATRIX

```
● ● ●

# 8. Prediksi dan evaluasi
y_pred = model.predict(X_test)
print(f"Akurasi: {accuracy_score(y_test, y_pred):.2f}")
print("\nLaporan Klasifikasi:")
print(classification_report(y_test, y_pred))

# 9. Confusion matrix
cm = pd.crosstab(y_test, y_pred, rownames=['Aktual'], colnames=['Prediksi'])
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.show()
```

GAMBAR CONFUSION MATRIX



HASIL EVALUASI MODEL

Akurasi: 0.94

Laporan Klasifikasi:

	precision	recall	f1-score	support
0	0.93	0.93	0.93	14
1	0.95	0.95	0.95	20
accuracy			0.94	34
macro avg	0.94	0.94	0.94	34
weighted avg	0.94	0.94	0.94	34

PENGUJIAN HASIL CONFUSION MATRIX

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{19 + 13}{19 + 13 + 1 + 1} = \frac{32}{34} \approx 94.1\%$$

$$\text{Precision}_{\text{kelas 1 (bercerai)}} = \frac{TP}{TP + FP} = \frac{19}{19 + 1} = \frac{19}{20} = 95\%$$

$$\text{Recall}_{\text{kelas 1 (bercerai)}} = \frac{TP}{TP + FN} = \frac{19}{19 + 1} = \frac{19}{20} = 95\%$$

$$\text{Precision}_{\text{kelas 0 (tidak bercerai)}} = \frac{TN}{TN + FN} = \frac{13}{13 + 1} = \frac{13}{14} \approx 92.86\%$$

$$\text{Recall}_{\text{kelas 0 (tidak bercerai)}} = \frac{TN}{TN + FP} = \frac{13}{13 + 1} = \frac{13}{14} \approx 92.86\%$$

INFERENSI MODEL

```
import numpy as np

# Nilai dari Q1 hingga Q54 untuk mengisi fitur
data_baru = [
    4, 3, 4, 1, 2,
    3, 4, 3, 2, 1,
    4, 3, 4, 4, 3,
    2, 1, 3, 3, 2,
    4, 4, 2, 2, 3,
    4, 3, 4, 2, 1,
    3, 3, 4, 3, 2,
    4, 4, 4, 2, 1,
    3, 3, 2, 4, 4,
    3, 2, 1, 4, 4,
    4, 3, 2, 4
]

# Ubah data menjadi array dan bentuk ulang agar sesuai input model
data_baru_df = pd.DataFrame([data_baru], columns=X_train.columns)

# Lakukan prediksi
hasil_prediksi = model.predict(data_baru_df)

# Interpretasi hasil
if hasil_prediksi[0] == 1:
    print("Prediksi: Akan Bercerai")
else:
    print("Prediksi: Tidak Bercerai")
```

Prediksi: Akan Bercerai

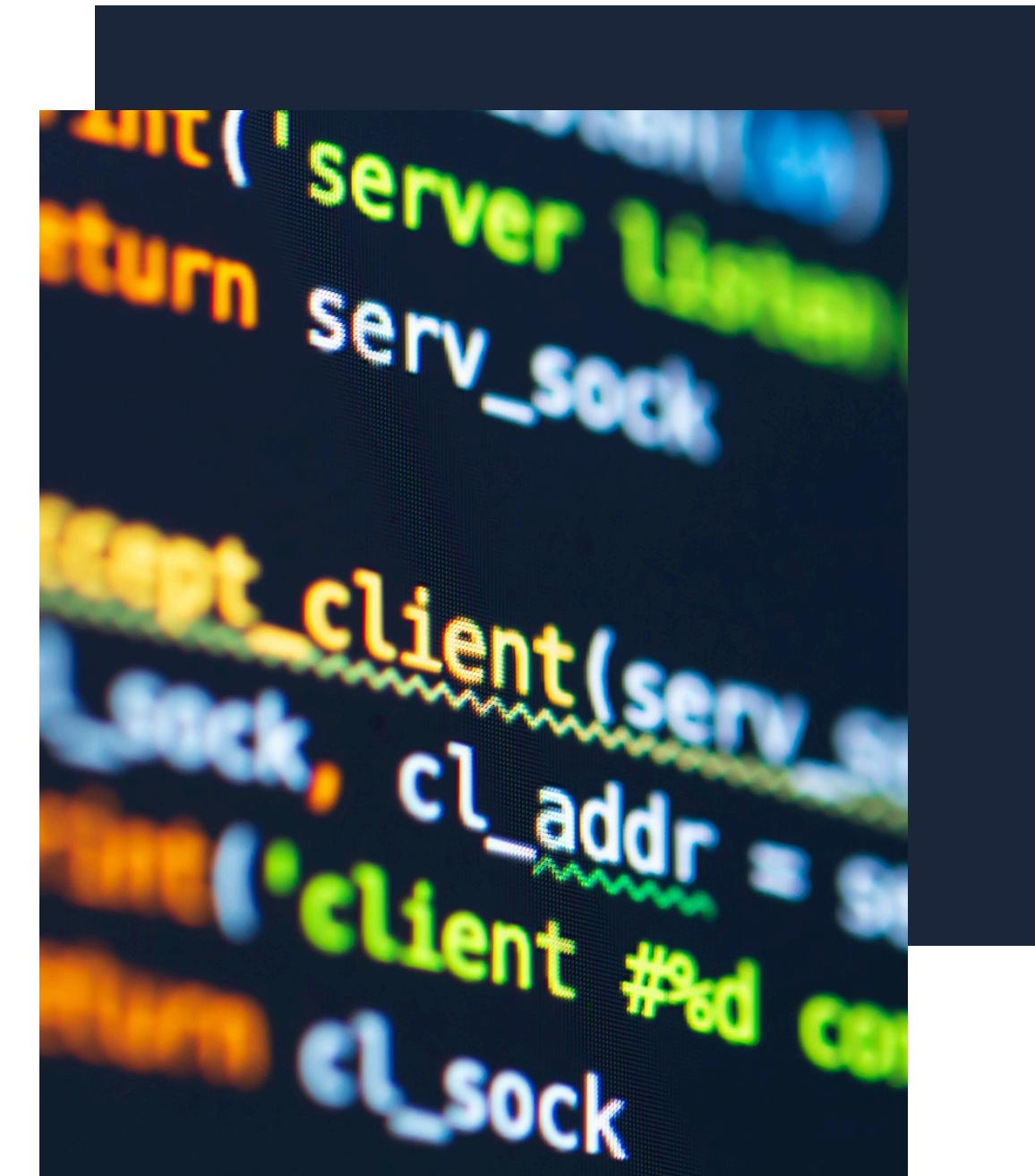
TUGAS 9

“PROGRAM CLUSTERING SEDERHANA
MENGGUNAKAN
BAHASA PEMROGRAMAN PYTHON”

STUDI KASUS

Clustering merupakan salah satu metode penting dalam bidang data mining dan machine learning, khususnya dalam pendekatan unsupervised learning. Clustering berperan dalam mengelompokkan data ke dalam beberapa bagian atau klaster berdasarkan kemiripan fitur yang dimiliki oleh masing-masing data, tanpa perlu adanya label atau kategori yang sudah ditentukan sebelumnya.

Dalam tugas ini, diimplementasikan metode clustering sederhana menggunakan algoritma K-Means, salah satu teknik clustering yang paling umum dan efisien. Implementasi dilakukan dalam lingkungan Google Colab dengan memanfaatkan bahasa pemrograman Python. Program dirancang untuk menerima input dataset berformat CSV dan memungkinkan pengguna menentukan jumlah klaster yang diinginkan. Selain itu, hasil clustering divisualisasikan dalam bentuk grafik dua dimensi untuk memudahkan interpretasi terhadap pembagian klaster yang dihasilkan.



UPLOAD DATASET

```
●●●  
# Langkah 1: Upload file csv  
from google.colab import files  
import pandas as pd  
import matplotlib.pyplot as plt  
from sklearn.cluster import KMeans  
from sklearn.preprocessing import StandardScaler  
import ipywidgets as widgets  
from IPython.display import display  
import numpy as np  
  
# Upload file  
print("Upload file CSV Anda:")  
uploaded = files.upload()  
  
# Baca file yang diunggah  
filename = list(uploaded.keys())[0]  
df = pd.read_csv(filename)  
print("\nDataset berhasil dimuat:")  
display(df.head())
```

Program dimulai dengan mengimpor berbagai library penting. Library `google.colab.files` digunakan untuk proses unggah file CSV dari pengguna, sedangkan `pandas` dipakai untuk membaca dan mengelola data dalam bentuk `dataframe`. `matplotlib.pyplot` disiapkan untuk visualisasi hasil klaster, sementara `KMeans` dan `StandardScaler` dari `sklearn` digunakan dalam proses clustering dan normalisasi data. Untuk mendukung antarmuka interaktif, digunakan `ipywidgets` dan `IPython.display`, misalnya `slider` untuk memilih jumlah klaster.

Setelah library dimuat, program meminta pengguna mengunggah file CSV. Saat cell dijalankan, muncul tombol “Choose Files” yang memungkinkan pengguna memilih file. File dibaca dengan `pd.read_csv()` dan diakses dari dictionary `uploaded`. Setelah berhasil dimuat, program mencetak pesan konfirmasi dan menampilkan beberapa baris pertama data menggunakan `df.head()`, untuk memastikan format data sudah benar sebelum proses clustering dilanjutkan.

PEMILIHAN KOLOM

```
● ● ●

print("\nKolom yang tersedia:")
print(list(df.columns))

selected_columns = input("\nPilih kolom untuk clustering (pisahkan dengan koma): ")
selected_columns = [col.strip() for col in selected_columns.split(",")]
print(selected_columns)
data = df[selected_columns].select_dtypes(include=['float64', 'int64'])

if data.empty:
    raise Exception("Error: Tidak ada kolom numerik terpilih!")
```

INPUT K CLUSTER

Untuk memudahkan pengguna memilih jumlah klaster (k) dalam algoritma K-Means, program menggunakan slider interaktif dari pustaka ipywidgets. Slider ini menggantikan input manual agar lebih user-friendly, terutama bagi pengguna yang tidak terbiasa mengetik kode.

Slider dibuat menggunakan IntSlider dengan rentang 1–10, nilai default 3, dan langkah naik 1. Label "Jumlah cluster (k):" ditampilkan di samping slider agar jelas fungsinya, sementara pengaturan style={'description_width': 'initial'} memastikan tampilannya rapi. Slider kemudian ditampilkan dengan display(k_slider), dan nilai yang dipilih akan digunakan sebagai parameter k dalam proses clustering. Pendekatan ini membuat interaksi lebih intuitif dan ramah bagi pengguna.

```
● ● ●  
k_slider = widgets.IntSlider(  
    value=3,  
    min=1,  
    max=10,  
    step=1,  
    description='Jumlah cluster (k):',  
    style={'description_width': 'initial'},  
)  
  
display(k_slider)
```



```
● ● ●  
# Ambil nilai k dari slider  
k = k_slider.value  
  
# Pilih hanya dua kolom numerik  
pertama untuk visualisasi 2D  
  
# Standarisasi data  
scaler = StandardScaler()  
scaled_data =  
scaler.fit_transform(data)
```

PLOT ELBOW

```
● ● ●  
max_k = min(10, scaled_data.shape[0])  
  
def plot_elbow(data, max_k=10):  
    inertia = []  
    K_range = range(1, max_k + 1)  
    for k in K_range:  
        kmeans = KMeans(n_clusters=k, random_state=42)  
        kmeans.fit(data)  
        inertia.append(kmeans.inertia_)  
    plt.figure(figsize=(8, 5))  
    plt.plot(K_range, inertia, 'bo-', markersize=8)  
    plt.xlabel('Jumlah Cluster (k)')  
    plt.ylabel('Inertia')  
    plt.title('Metode Elbow untuk Menentukan Jumlah Cluster')  
    plt.grid(True)  
    plt.show()  
  
plot_elbow(scaled_data, max_k=max_k)
```

PLOT ELBOW

```
● ● ●

from sklearn.metrics import silhouette_score

def find_best_k(data, max_k=10):
    silhouette_scores = []
    K_range = range(2, max_k + 1)
    for k in K_range:
        kmeans = KMeans(n_clusters=k, random_state=42)
        labels = kmeans.fit_predict(data)
        score = silhouette_score(data, labels)
        silhouette_scores.append(score)
    best_k = K_range[silhouette_scores.index(max(silhouette_scores))]
    print(f"\nRekomendasi jumlah cluster terbaik berdasarkan Silhouette Score: {best_k}")
    plt.plot(K_range, silhouette_scores, 'bo-', markersize=8)
    plt.xlabel('Jumlah Cluster (k)')
    plt.ylabel('Silhouette Score')
    plt.title('Silhouette Score untuk Menentukan k Terbaik')
    plt.grid(True)
    plt.show()
    return best_k

# Cari k terbaik (opsional)
best_k = find_best_k(scaled_data, max_k=max_k)
```

VISUALISASI DAN IMPLEMENTASI K-MEANS

```
● ● ●

# Jalankan K-Means
model = KMeans(n_clusters=k, random_state=42)
labels = model.fit_predict(scaled_data)

# Jarak ke pusat cluster
distances = model.transform(scaled_data)
min_distances = distances[np.arange(len(distances)), labels]
threshold = np.percentile(min_distances, 95) # ambil 5% teratas
outliers = df[min_distances > threshold]
print("\n Deteksi Outlier (Top 5%):")
display(outliers)
```

VISUALISASI DAN IMPLEMENTASI K-MEANS

```
● ● ●

from sklearn.decomposition import PCA

print("\nHasil clustering:")
print(labels, selected_columns)
if len(selected_columns) == 2:
    plt.scatter(df[selected_columns[0]], df[selected_columns[1]], c=labels,
    cmap='viridis')
    plt.xlabel(selected_columns[0])
    plt.ylabel(selected_columns[1])
    plt.title('K-Means Clustering')
    plt.colorbar()
    plt.show()
else:
    pca = PCA(n_components=2)
    pca_components = pca.fit_transform(scaled_data)
    plt.scatter(pca_components[:, 0], pca_components[:, 1], c=labels, cmap='viridis')
    plt.xlabel('PCA 1')
    plt.ylabel('PCA 2')
    plt.title(f'Visualisasi PCA + KMeans Clustering (k={k})')
    plt.colorbar(label='Cluster')
    plt.show()
```

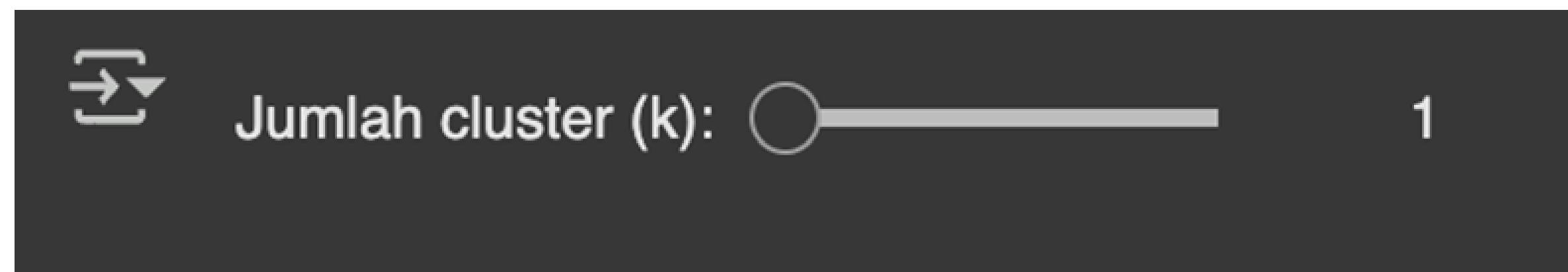
OUTPUT

1. Upload CSV

```
CSV Upload file CSV Anda:  
Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.  
Saving Book1.csv to Book1.csv  
  
CSV Dataset berhasil dimuat:  
X Y  
0 1 2  
1 2 1  
2 3 1  
3 6 7  
4 7 8
```

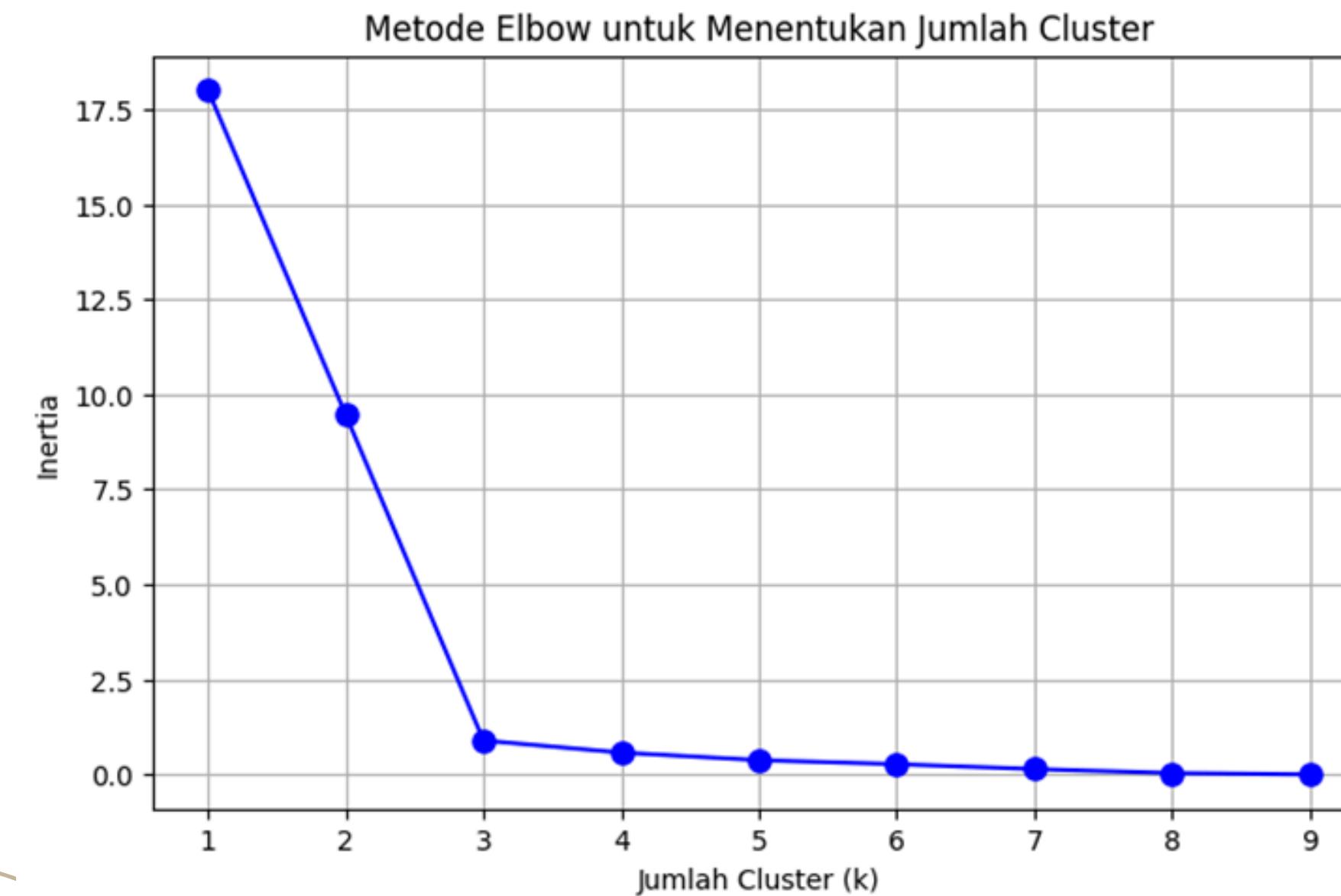
OUTPUT

2. Input Jumlah Cluster



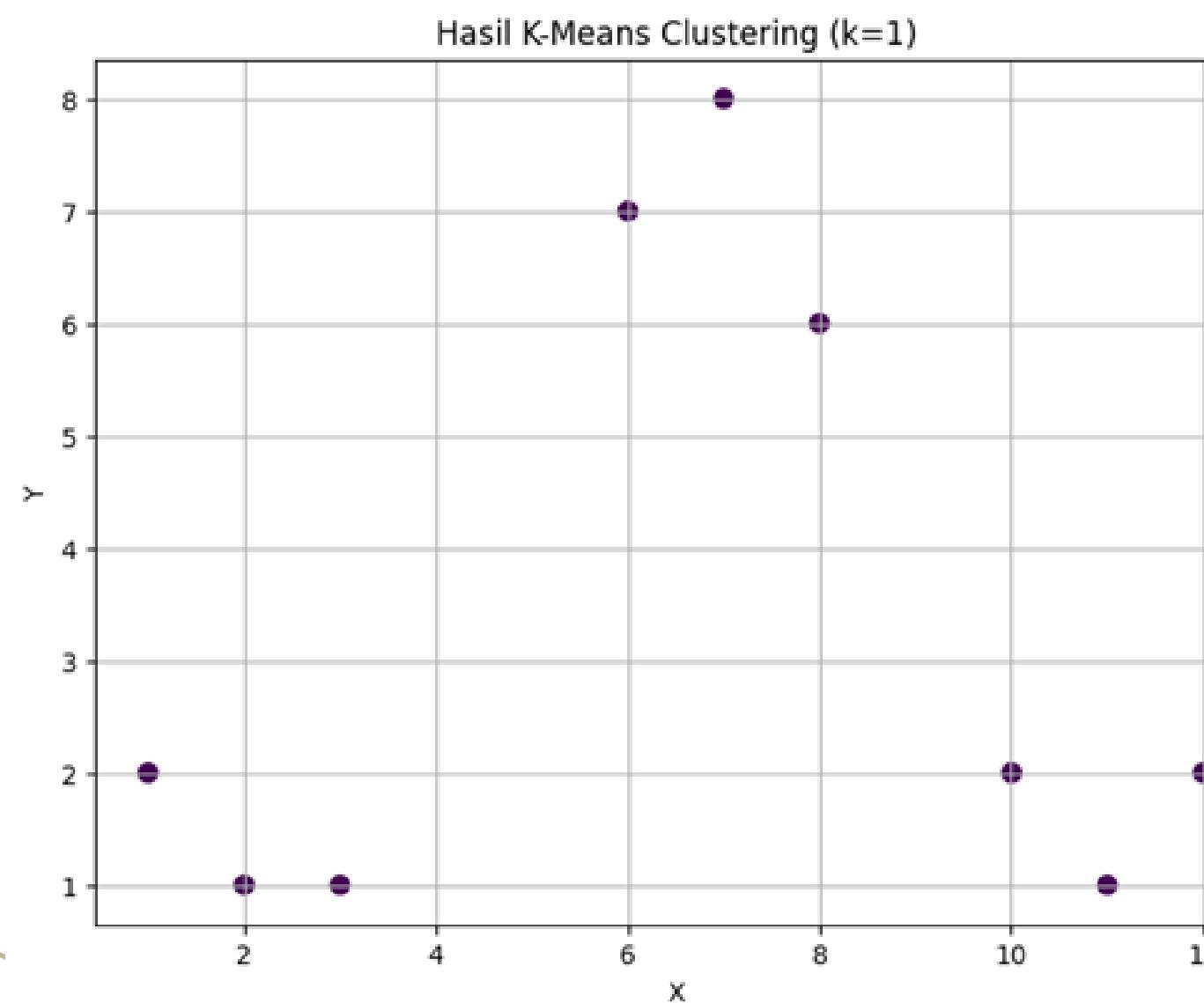
OUTPUT

3. Plot Elbow



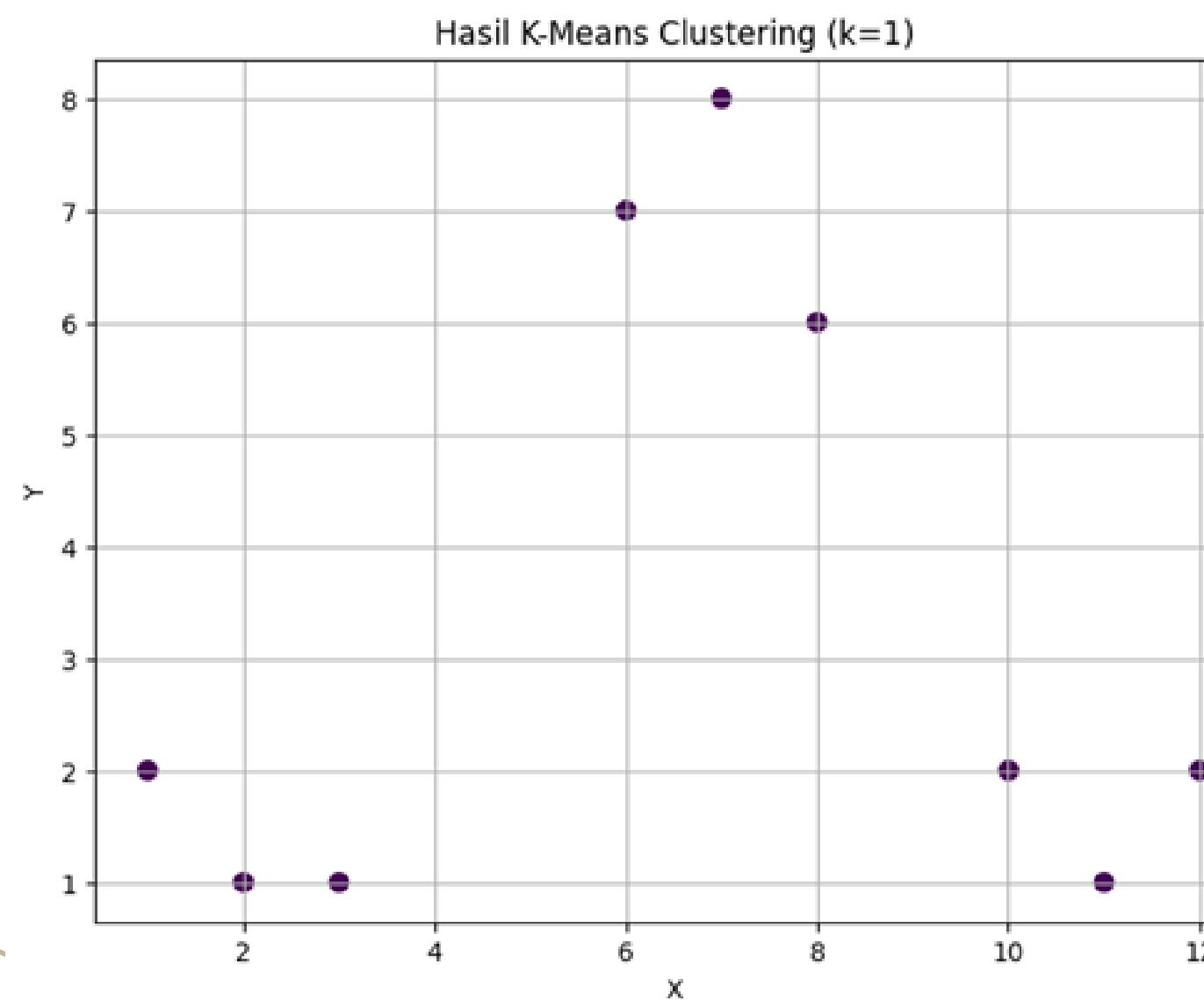
OUTPUT

4. Visualisasi Cluster



OUTPUT

4. Visualisasi Cluster



TUGAS 10

**“PENGEMBANGAN MODEL DETEKSI OBJEK
ALAT PELINDUNG DIRI DALAM RUANG
LINGKUP PERUSAHAAN MANUFAKTUR”**

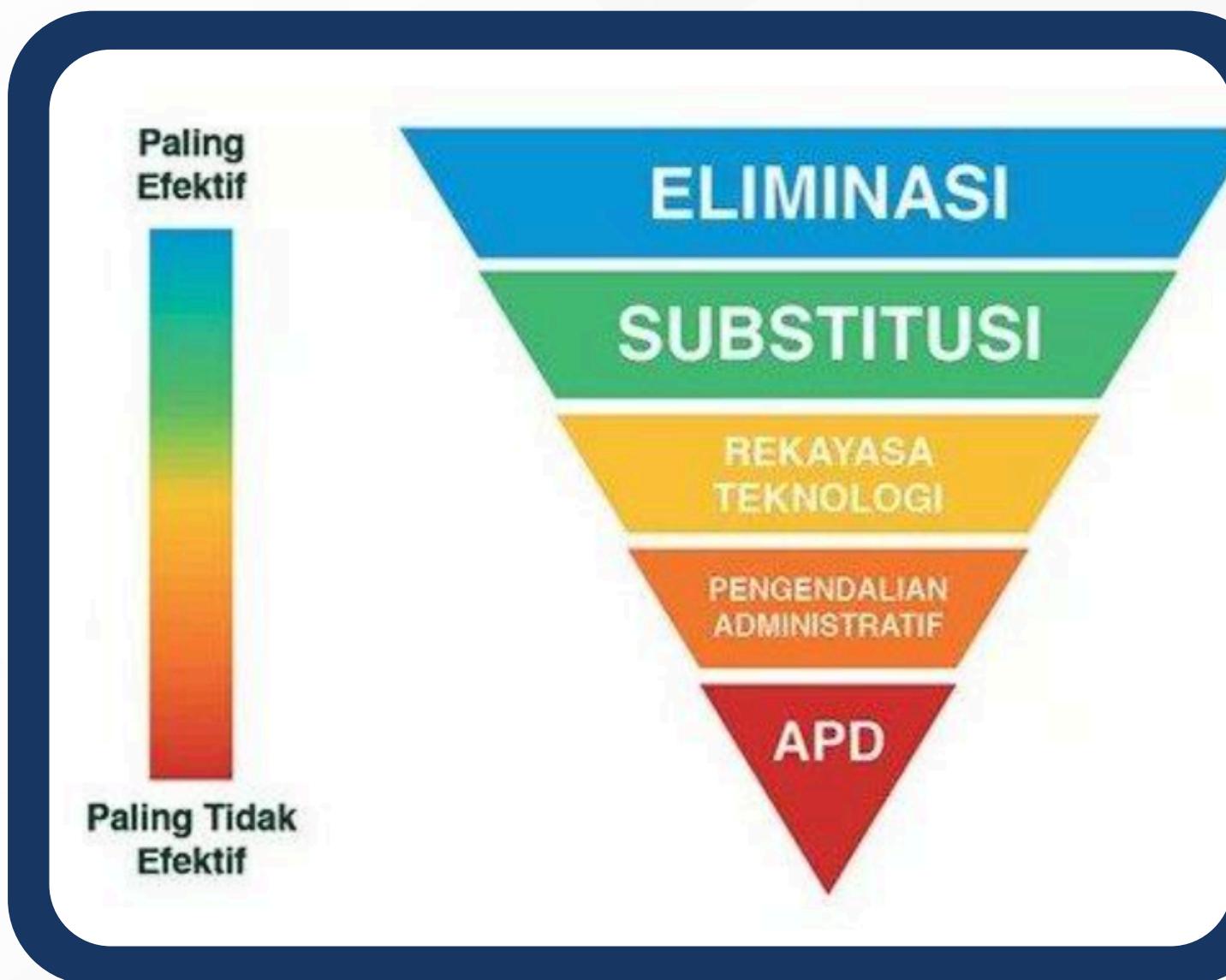
LATAR BELAKANG

Perusahaan manufaktur yang memproduksi bahan jadi (finished goods) tentunya terlibat dalam serangkaian proses produksi yang melibatkan mesin dan peralatan berat yang dapat menimbulkan potensi bahaya bagi para pekerjanya.

Informasi yang diperoleh dari DataIndonesia.id mengungkapkan bahwa Badan Penyelenggara Jaminan Sosial (BPJS) Ketenagakerjaan mencatat peningkatan yang signifikan dalam jumlah kecelakaan kerja di Indonesia.



HIERARKI PENGENDALIAN BAHAYA



Dalam perusahaan manufaktur, penggunaan alat pelindung diri sebenarnya dianggap sebagai opsi paling tidak efektif dalam pengendalian bahaya. Hal tersebut dikarenakan:

1. APD hanyalah melindung pekerja dari dampak bahaya bukan menghilangkan bahaya itu sendiri
2. Efektivitas APD sangat bergantung pada perilaku kerja.

TUJUAN MULIA

Berdasarkan latar belakang yang dipaparkan, maka pembuatan model bertujuan:

Mengenali apakah seseorang menggunakan APD rompi atau tidak

Faktor Psikologis dalam Keputusan Pembelian Media Sosial (Jurnal B, 2020).

Peran Visual dalam Meningkatkan Produk di Media Sosial (Jurnal C, 2022).

PENDAHULUAN

Gambar dalam komputer direpresentasikan sebagai array numerik berdimensi tinggi. Misalnya:

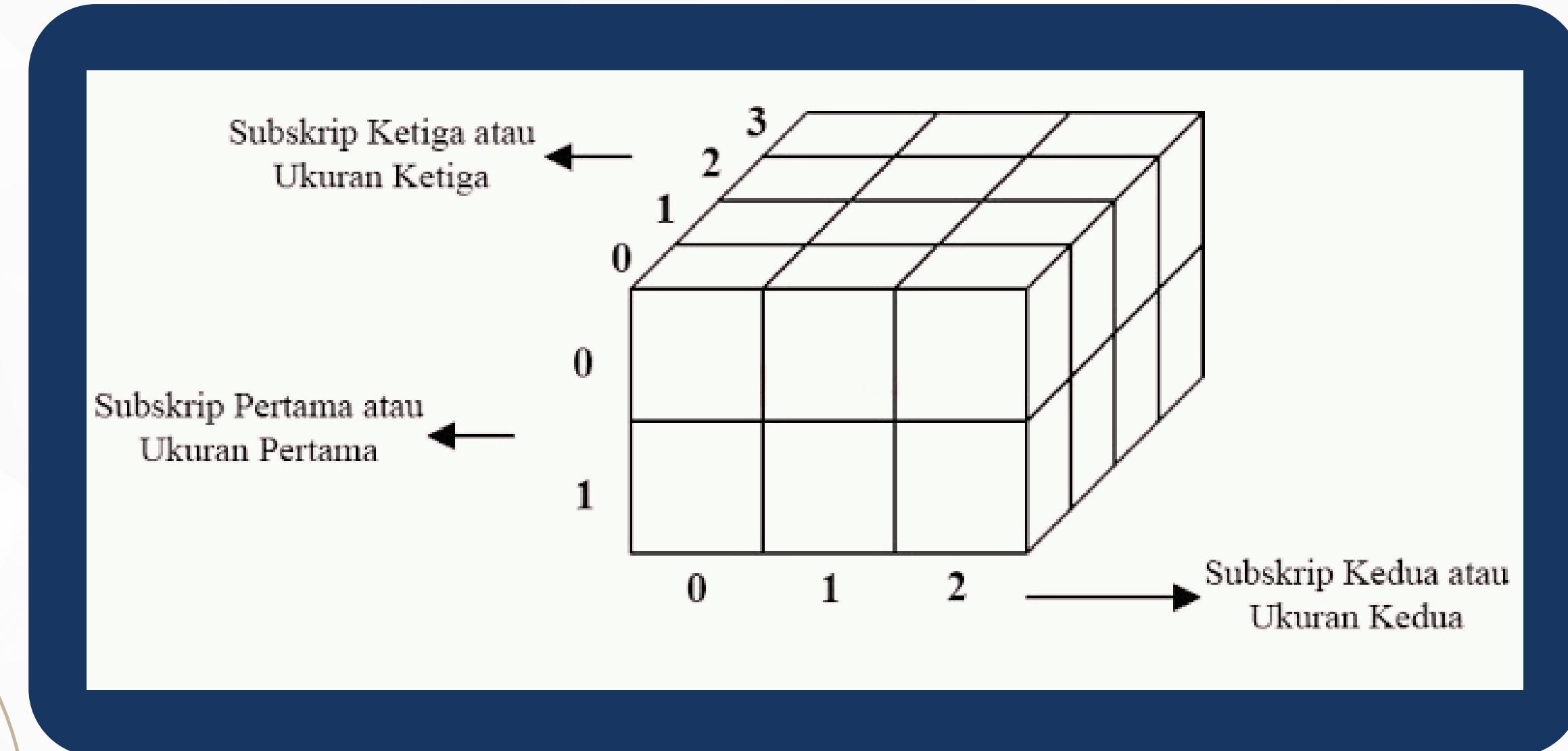
- Gambar berwarna ukuran 224x224 piksel disimpan sebagai matriks 3 dimensi: 224x224x3, di mana 3 mewakili kanal warna RGB, sementara 224 adalah tinggi dan lebar.
- Setiap piksel memiliki nilai intensitas (0–255) untuk masing-masing warna.

Sehingga bentuk dari arraynya dapat dikatakan seperti berikut:

$\text{image}[x,y] = [\text{red}, \text{green}, \text{blue}]$

$\text{image}[100, 150] = [200, 100, 50]$

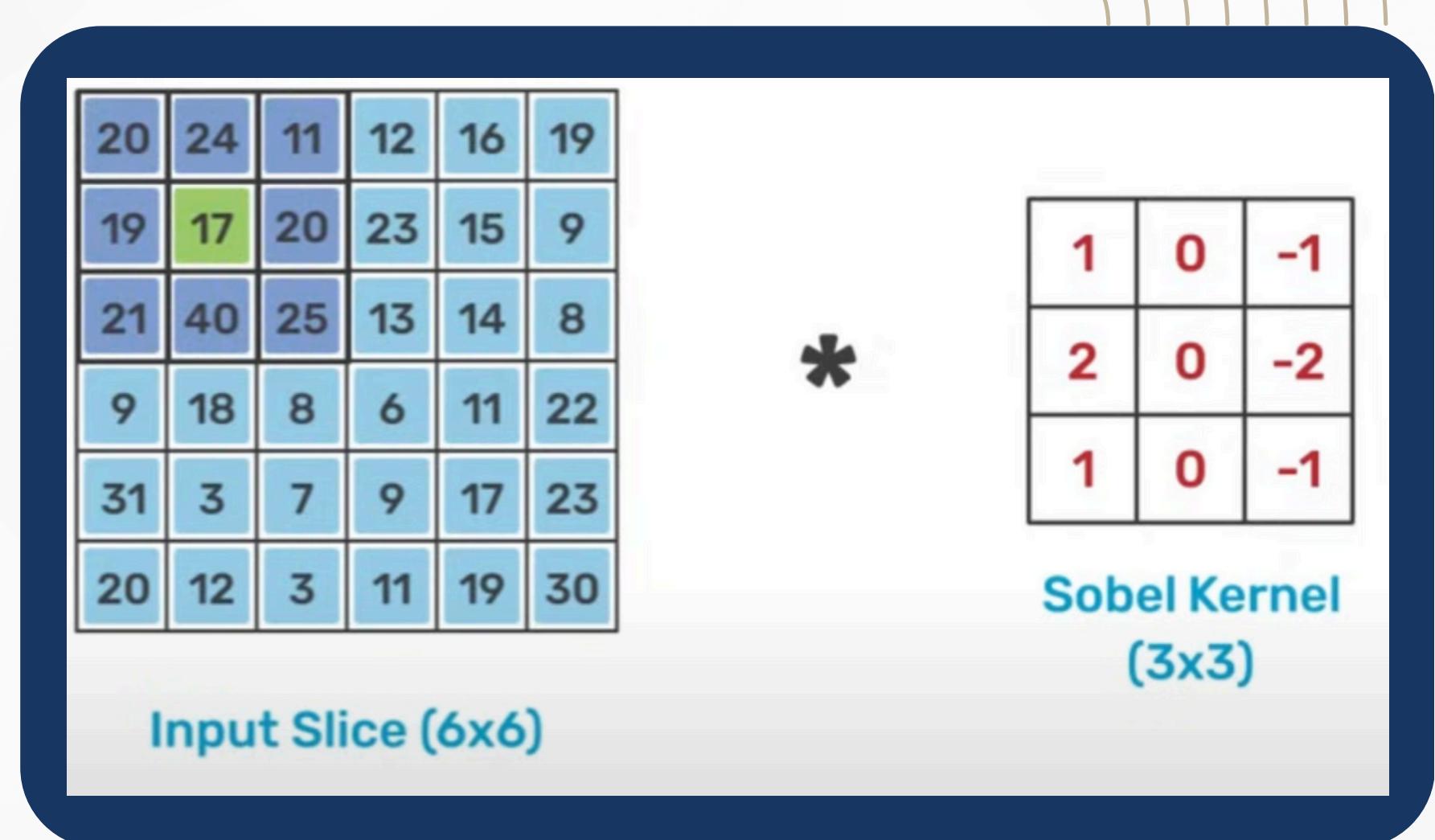




Dalam matriks 3 dimensi di atas, terdapat 3 layer index. Setiap index akan berisi nilai RGB seperti [R, G, B]. Pada layer pertama indexnya mempresentasikan koordinat Y, layer kedua koordinat X dan layer terakhir berisi RGB.

DASAR PENGENALAN

Pada dasarnya komputer tidak dapat mengenali atau melihat gambar secara langsung. Komputer hanya melakukan pemrosesan terhadap matriks 3D yang telah dikenali. Komputer mengenali pola-pola tersebut dalam bentuk angka tertentu. Untuk mengenali pola, komputer menggunakan filter (kernel) yang berbentuk matrix 3x3 atau 5x5 untuk mengenali garis horizontal, vertikal dan lainnya.



CONVOLUTIONAL

Operasi convolutional dimulai dengan menggeser filter (kernel) atau disebut dengan sliding dengan digeser dari kiri atas ke kanan bawah gambar, satu piksel sekali. Selanjutnya area yang disoroti filter atau disebut receptive field dilakukan dot product dengan kernel sehingga menghasilkan sebuah output yang disebut feature map.

Setiap nilai pada feature map, terhubung ke center dari receptive field. Dapat diperhatikan spasial dari output akan selalu lebih rendah dari input karena perbedaan ukuran dari input dengan kernel. Namun ada cara agar tetap sama yaitu dengan menggunakan padding atau pinggiran palsu. Output juga bisa mengecil jika stride kernel membesar.

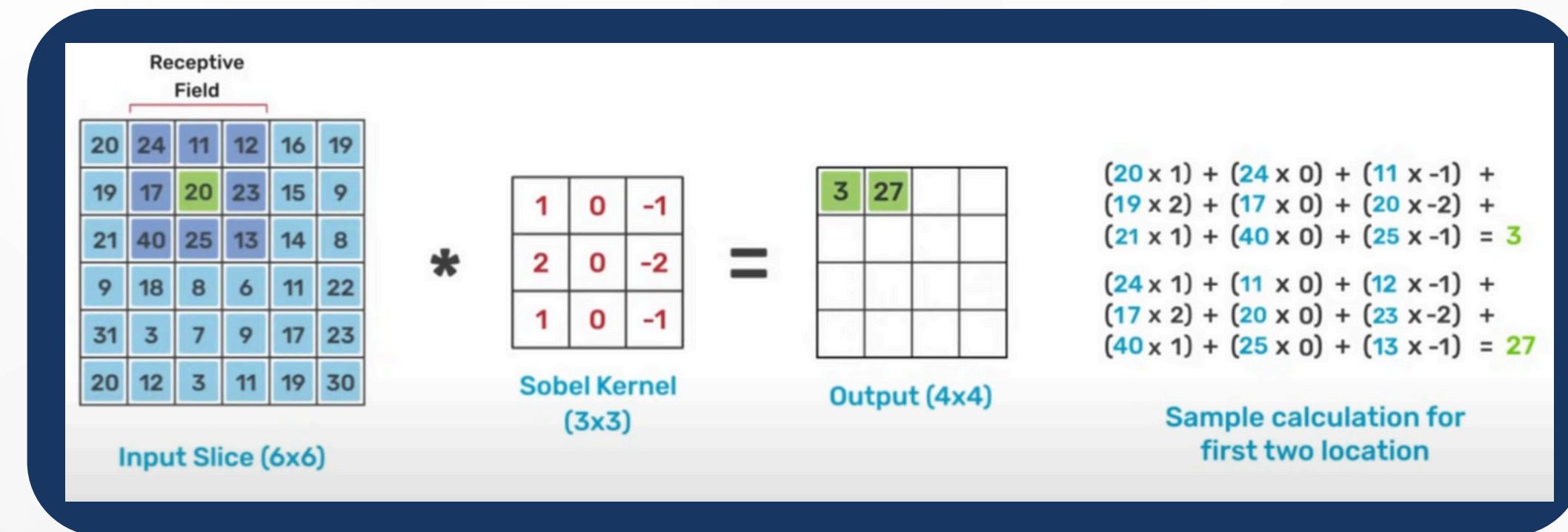
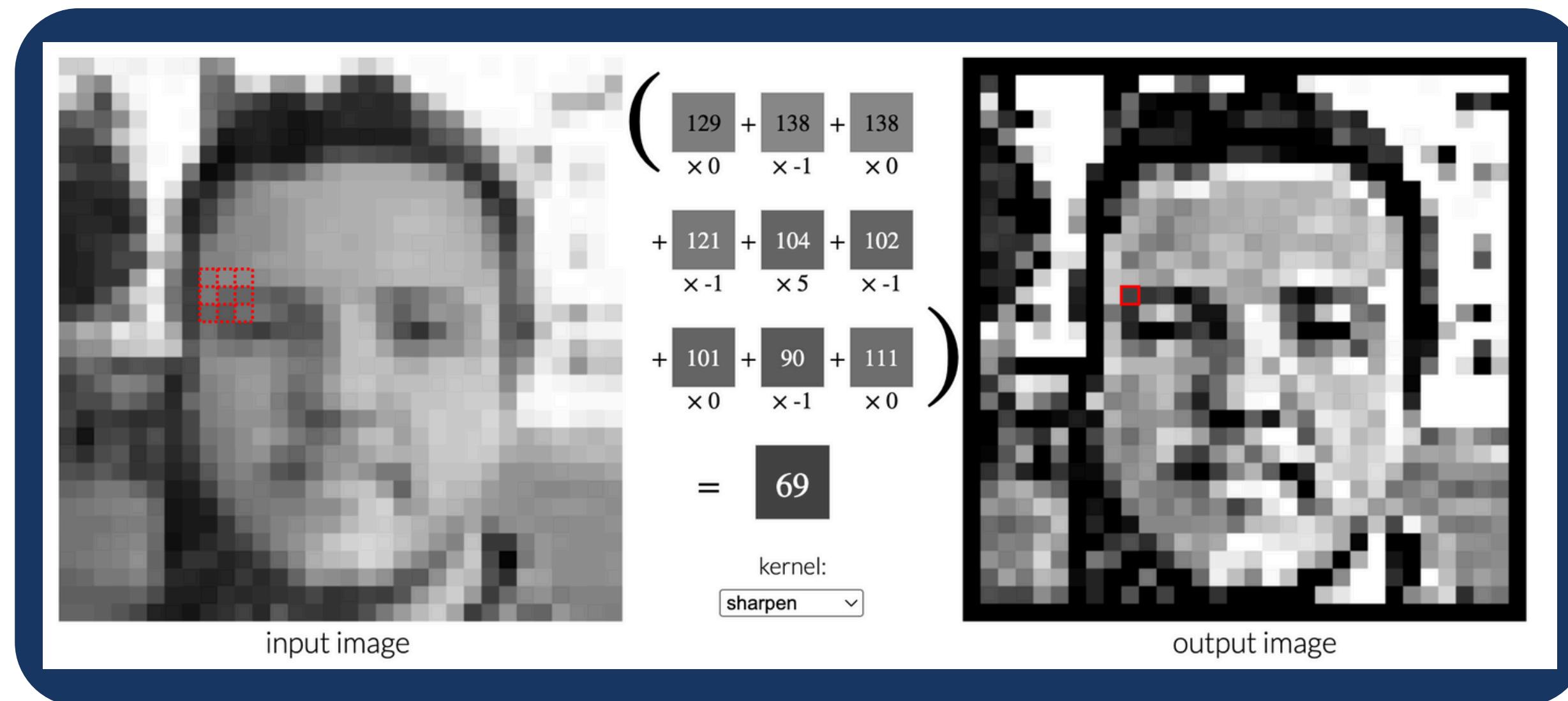


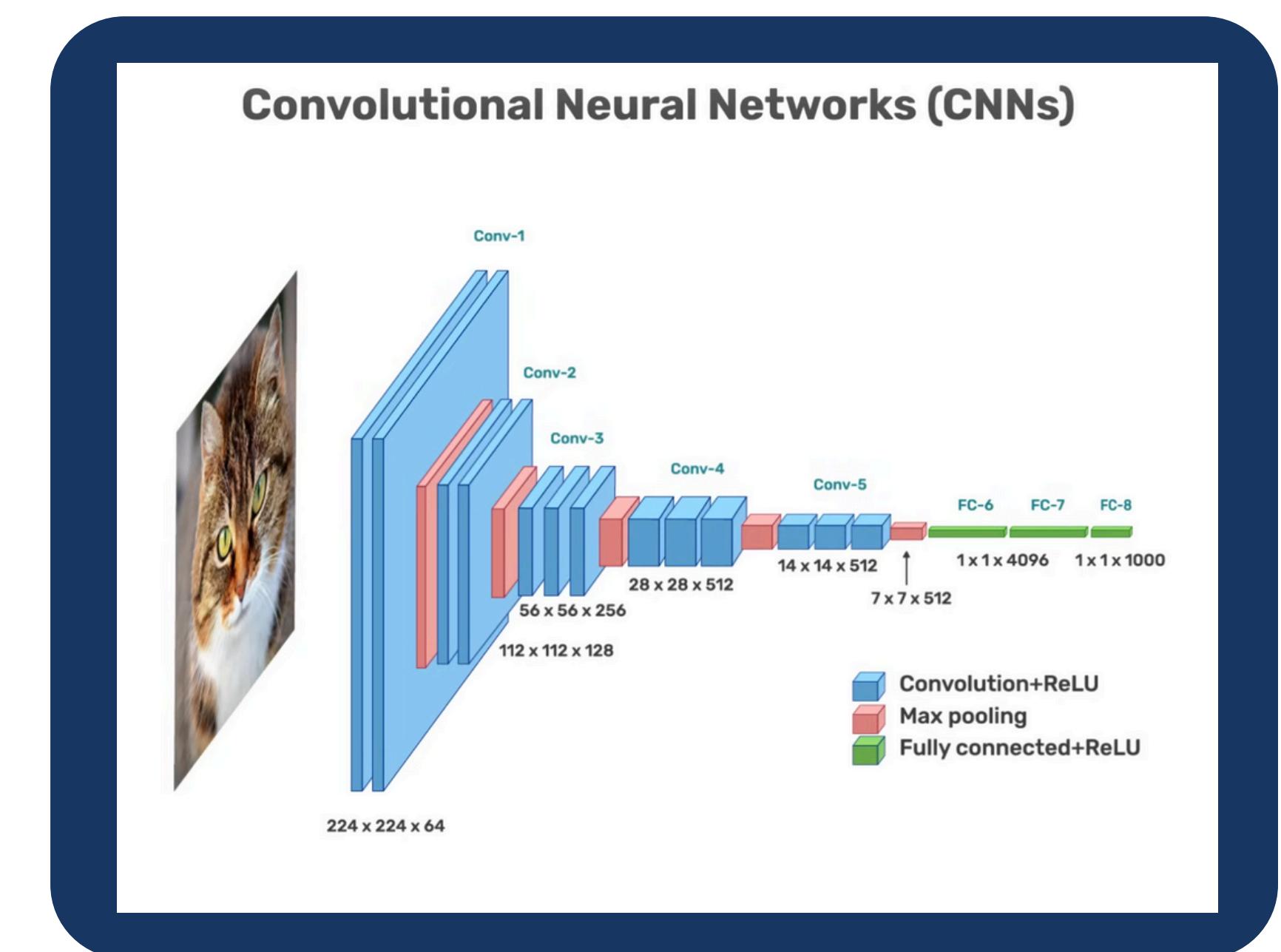
IMAGE KERNELS EXPLAINED VISUALLY



Terdapat website yang dapat memvisualisasikan bagaimana kernel bekerja, yakni: <https://setosa.io/ev/image-kernels/>

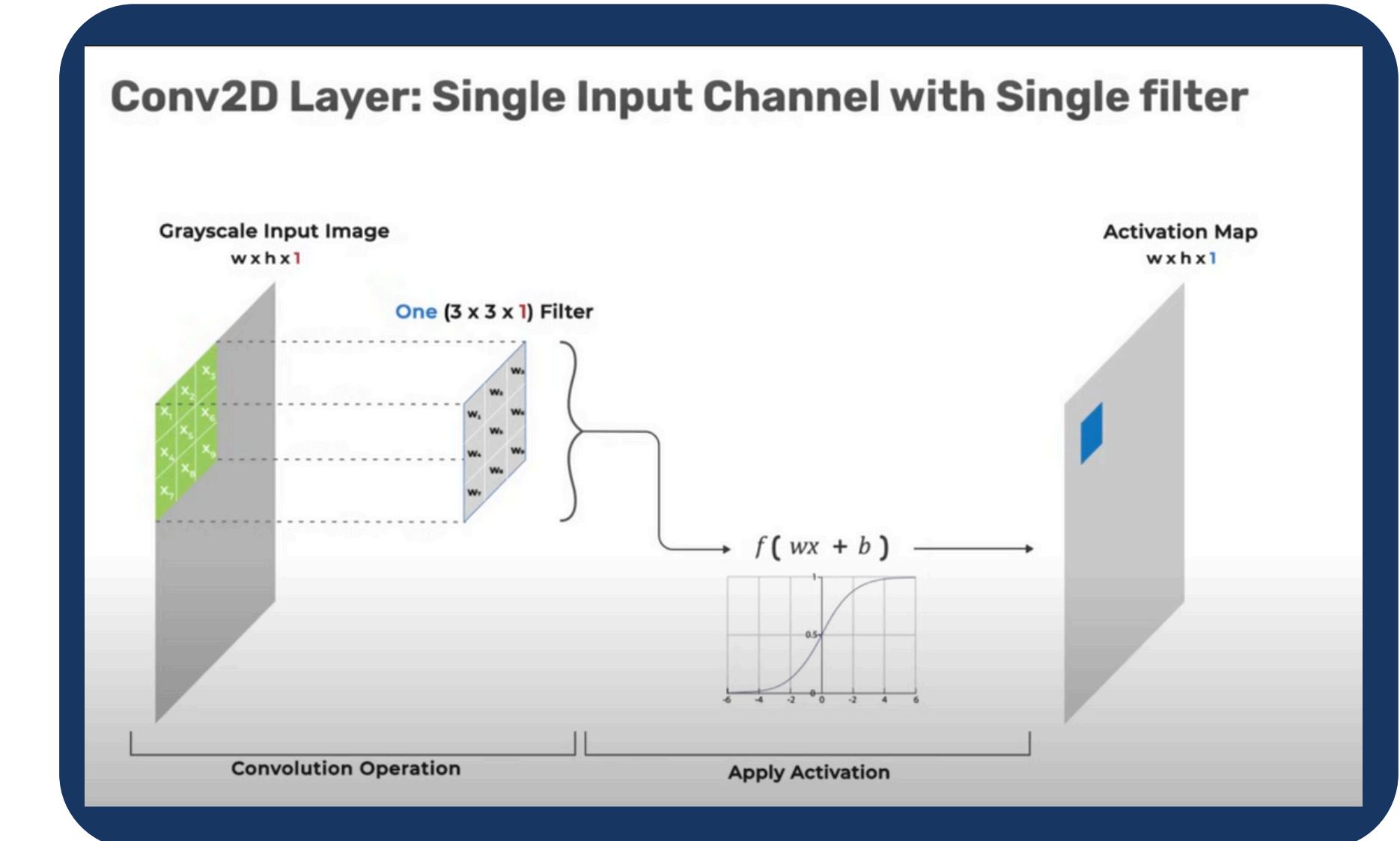
CONVOLUTIONAL NEURAL NETWORK

Dalam pengolahan citra (image) yang sangat kompleks, penggunaan deep learning akan sangat tepat karena memanfaatkan neural network yang memiliki akurasi tinggi untuk tugas kompleks. Salah satu arsitektur dalam deep learning adalah CNN (Convolutional Neural Network) yang salah satu perbedaan besarnya dibanding Neural Netwok biasa adalah dapat mengetahui kernel yang tepat ketika proses training secara otomatis.



BEHIND THE SCENE

Ketika CNN bekerja, terdapat sebuah function aktivasi yang dapat mengubah nilai dari kernel secara otomatis, function tersebut bernama Apply Activation.



YOLO DATASET

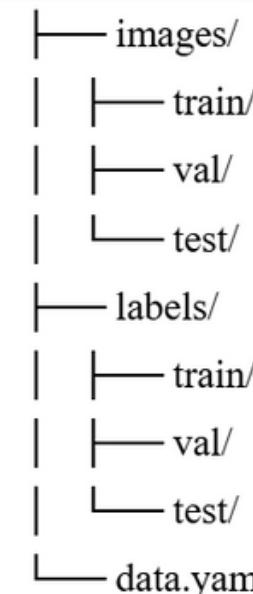
YOLO menggunakan format anotasi bounding box yang efisien dan sederhana. Setiap anotasi disimpan dalam file .txt dengan format:

class_id x_center y_center width height

Semua nilai posisi dan ukuran bersifat relatif terhadap ukuran gambar.

- class_id: Indeks kelas (mulai dari 0)
- x_center, y_center: Titik tengah bounding box (dalam rasio)
- width, height: Ukuran bounding box (dalam rasio)

Dataset YOLO memiliki struktur standar seperti berikut:



- images/: Menyimpan gambar (.jpg/.png)
- labels/: Menyimpan file anotasi .txt dengan format bounding box
- data.yaml: File konfigurasi dataset, berisi path dan daftar kelas objek

File data.yaml sangat penting karena digunakan oleh framework seperti Ultralytics YOLO untuk membaca lokasi data dan mengenali daftar kelas selama pelatihan dan validasi.

LAYER CNN

Berikut adalah layer CNN yang digunakan dalam pengembangan

Conv2D

Melakukan operasi konvolusi pada input gambar. Setiap filter (kernel) bekerja untuk mengenali fitur sederhana seperti tepi vertikal, horizontal, sudut, atau tekstur dasar.

BatchNormalization

Menormalkan nilai output dari layer sebelumnya (mean mendekati 0, standar deviasi mendekati 1) dengan itu akan mengurangi internal covariate shift – perubahan distribusi input antar layer.

MaxPooling2D

Mengambil nilai maksimum dari area 2x2 dan mengurangi dimensi (spasial) dari citra dan menyimpan hanya fitur dominan.

LAYER CNN

Dropout

Mematikan neuron secara acak saat training, sesuai dengan persentase yang ditentukan.

GlobalAveragePooling2D

Mengambil rata-rata dari tiap feature map (bukan flatten) yang dimana hasilnya adalah vektor 1D yang merangkum semua informasi fitur visual.

Dense

Menyambungkan seluruh fitur yang sudah dipelajari menjadi representasi akhir untuk klasifikasi dan menggunakan Aktivasi ReLU menjaga non-linearitas.

Dense with Activation

Mengeluarkan probabilitas prediksi untuk tiap kelas (misalnya: vest, no-vest). Dengan softmax memastikan total output berjumlah 1.

CALLBACK

Berikut adalah callback yang digunakan dalam pengembangan

EarlyStopping

Callback ini akan menghentikan training lebih awal jika metrik yang dimonitor (dalam pengembangan ini val_loss) tidak membaik dalam beberapa epoch berturut-turut.

ReduceLROnPlateau

Callback ini akan mengurangi learning rate saat model terjebak dan tidak mengalami perbaikan pada metrik yang dipantau.

ModelCheckpoint

Callback ini akan menyimpan model ke file setiap kali model mencapai performa terbaik (berdasarkan metrik tertentu).

MODEL AT COMPILE

Ketika melakukan compile (proses untuk mengkonfigurasi model), terdapat tiga komponen yang penting, yakni:

Optimizer

Algoritma yang digunakan untuk mengatur dan memperbarui bobot (weights) model selama proses training berdasarkan hasil dari loss

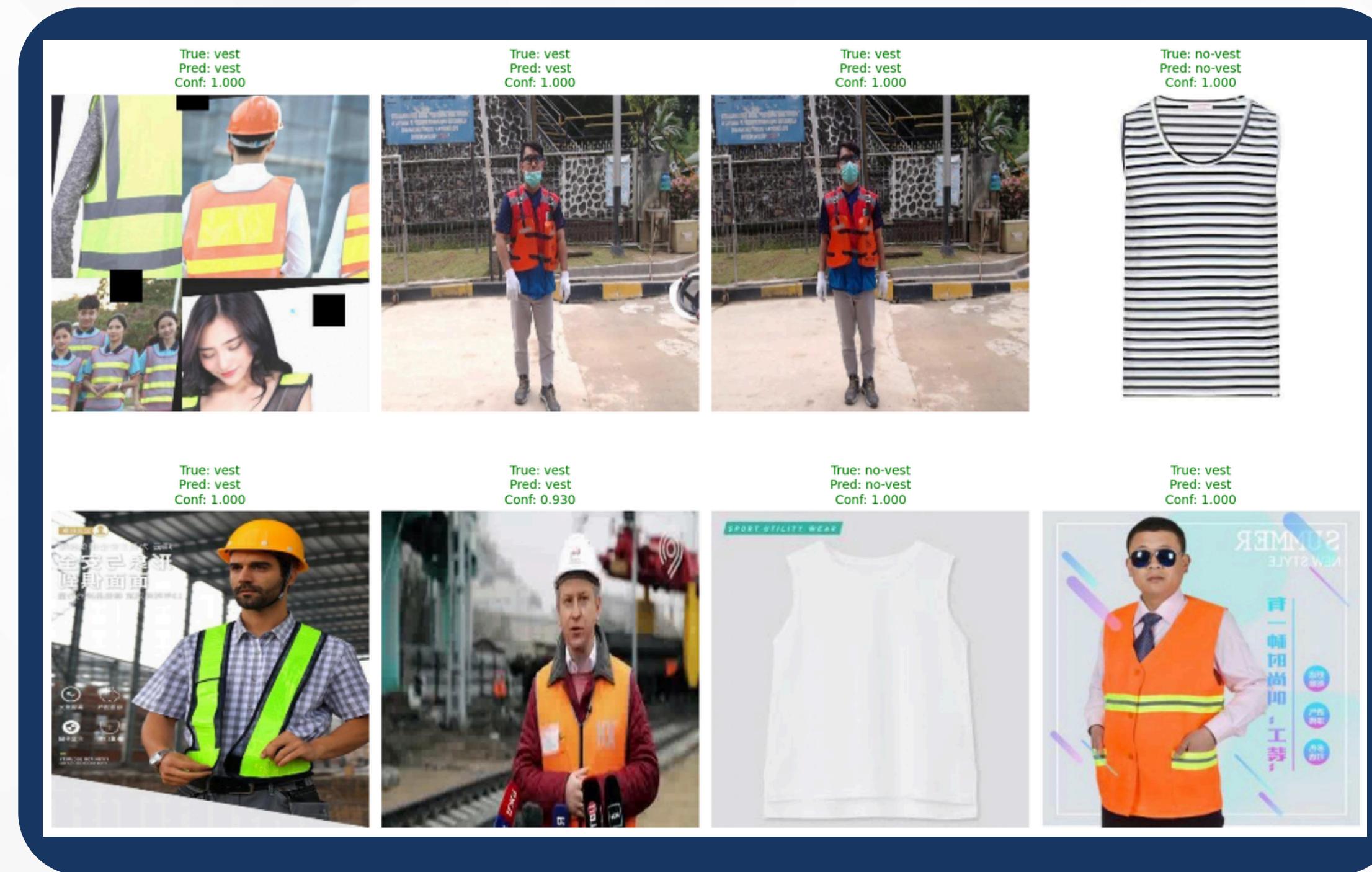
Loss Function

Mengukur seberapa buruk prediksi model dibanding label yang benar. Tujuannya adalah meminimalkan nilai loss ini selama training.

Metrics

Memilih hal apa yang dinilai untuk merepresentasikan performa model selama training dan evaluasi.

INFERENSI DENGAN DATA TEST





DISUSUN OLEH
KELOMPOK 2

TERIMA KASIH

Terima kasih atas kesempatan ini. Kami berharap kita dapat melanjutkan diskusi ini di lain waktu. Sampai jumpa!

Visualisasi akurasi dan loss per epoch.

◆ 8. Evaluasi Akhir

Evaluasi performa model terhadap test set.

Interpretasi akurasi, loss, classification report.

Confusion matrix dan makna tiap sel.

Visualisasi prediksi vs label asli (dengan confidence).