

# Laporan Praktikum week 2

Nama : Muhamad Debi Priantoro

NIM : 224308014

Kelas : TKA-6A

Akun Github (Tautan) : <https://github.com/muhammaddebi12>

Student Lab Assistant : Muhammad Mahirul Faiq (214308043)

## 1. Judul Percobaan

Object Classification with Scikit-learn

## 2. Tujuan Percobaan

Tujuan dari praktikum ini yaitu :

1. Memahami dasar-dasar *Machine Learning* dalam sistem kendali.
2. Mengimplementasikan model ML sederhana untuk klasifikasi objek.
3. Menggunakan Scikit-learn untuk membuat model ML dasar.
4. Mengintegrasikan model ML dengan Computer Vision untuk deteksi objek.
5. Mengelola dataset dan melakukan pelatihan model sederhana.

## 3. Landasan Teori

*Machine Learning* adalah salah satu bidang cabang ilmu komputer yang memberikan kemampuan kepada komputer untuk dapat belajar tanpa diprogram secara eksplisit (Daqiqil, 2021). Untuk bisa mengaplikasikan teknik-teknik machine learning maka harus ada data. Tanpa data maka algoritma machine learning tidak dapat bekerja. Data yang ada biasanya dibagi menjadi dua, yaitu data training dan data testing. Data training digunakan untuk melatih algoritma, sedangkan data *testing* digunakan untuk mengetahui performa algoritma yang telah dilatih sebelumnya ketika menemukan data baru yang belum pernah dilihat. *Machine Learning* sendiri terbagi menjadi beberapa algoritma yang berbeda-beda dan setiap dari algoritma tersebut memiliki fungsi dan tujuannya masing-masing, seperti *supervised learning*, *unsupervised learning*, dan *reinforcement learning* (Karimah, 2023).

*Supervised Learning* adalah algoritma yang menyimpulkan fungsi dari data pelatihan berlabel yang terdiri dari sekumpulan contoh pelatihan, misalnya seperti *logistic regression* dan *K-Nearest Neighbors* (KNN). *Unsupervised Learning* adalah algoritma yang bertujuan untuk meningkatkan kinerja tugas pembelajaran yang diawasi dimana kita tidak memiliki banyak data, contohnya seperti *k-means*, *Apriori*, *DBSCAN*, dan *clustering*. *Reinforcement Learning* adalah algoritma yang mengumpulkan pengetahuan (sinyal penguatan) untuk memilih tindakan yang mengarah ke hasil tertinggi yang diharapkan. Dibandingkan dengan metodologi lain dalam *Machine Learning*, algoritma *Reinforcement Learning* memiliki kelebihan yang berbeda yaitu kemampuan untuk secara mandiri menjelajahi lingkungan yang sangat dinamis dan stokastik dan mengembangkan, dengan mengumpulkan umpan balik evaluatif dari lingkungan, kebijakan kontrol yang optimal (Fathurohman, 2021). *Machine Learning* memungkinkan sistem kendali untuk belajar dari data dan meningkatkan performanya seiring waktu. Beberapa contoh penerapannya yaitu *Predictive Maintenance* (mendeteksi potensi kegagalan sistem sebelum terjadi), *Adaptive Control* (sistem kendali yang dapat menyesuaikan parameter secara otomatis), dan *Anomaly Detection* (mendeteksi perilaku tidak normal dalam sistem industri) (Roihan dkk., 2020).

## 4. Analisis dan Diskusi

### Analisis:

Performa model dalam mendeteksi warna bergantung pada kualitas dan variasi data pada dataset (**color.csv**). Model ***K-Nearest Neighbors*** (KNN) pada kode tersebut menggunakan fitur RGB untuk melakukan klasifikasi warna. Jika dataset mencakup berbagai warna secara merata, model akan memiliki akurasi yang baik dalam mendeteksi warna. Namun, jika warna tidak merata maka akan sebaliknya. Oleh karena itu, untuk memiliki performa yang baik secara objektif, dapat menggunakan metrik seperti akurasi, *precision*, *recall*, dan F1-score dengan membandingkan hasil dengan label asli pada data. Perbedaan akurasi jika menambah jumlah dataset adalah akan meningkatkan akurasi model, terutama jika data tambahan mencakup warna yang lebih luas, lebih banyak, dan detail, karena model mempelajari pola warna yang berbeda. Namun, jika data tambahan memiliki *noise*, maka performa model dan akurasi bisa menurun.

### Diskusi:

1. Keuntungan *Machine Learning* (ML) dibandingkan metode berbasis *rule-based* yaitu ML memiliki fleksibilitas yang lebih tinggi dan tidak memerlukan aturan yang ditulis secara manual, kemudian ML dapat belajar secara otomatis dari data dan menangkap pola yang sulit. Selain itu, model ML dapat beradaptasi dengan mudah terhadap data baru dan meningkatnya akurasi seiring dengan waktu jika dibandingkan dengan *rule-based*.
2. ML dapat diintegrasikan lebih lanjut dan jauh dalam sistem kendali untuk meningkatkan efisiensi dan adaptasi. Beberapa contohnya yaitu dalam sistem otomatisasi, model ML dapat digunakan untuk deteksi objek secara *real-time* menggunakan kamera, kemudian dihubungkan dengan aktuator untuk pengambilan tindakan secara otomatis. Selain itu, ML digunakan untuk memprediksi sistem dan optimasi, seperti mengatur parameter kontrol berdasarkan kondisi lingkungan yang berubah-ubah.
3. Tantangan penerapan ML dalam sistem *real-time* yaitu:
  - Keterlambatan dalam pengambilan keputusan.
  - Keterbatasan CPU, GPU, dan memori pada perangkat *embedded*.
  - Keandalan dan akurasi dalam kondisi lingkungan yang *noise* dan dinamis dapat mempengaruhi input sensor.
  - Model ML dapat *overfitting* jika tidak diperbaharui secara berkala dan tidak dilatih dengan dataset yang beragam.

## 5. Assignment

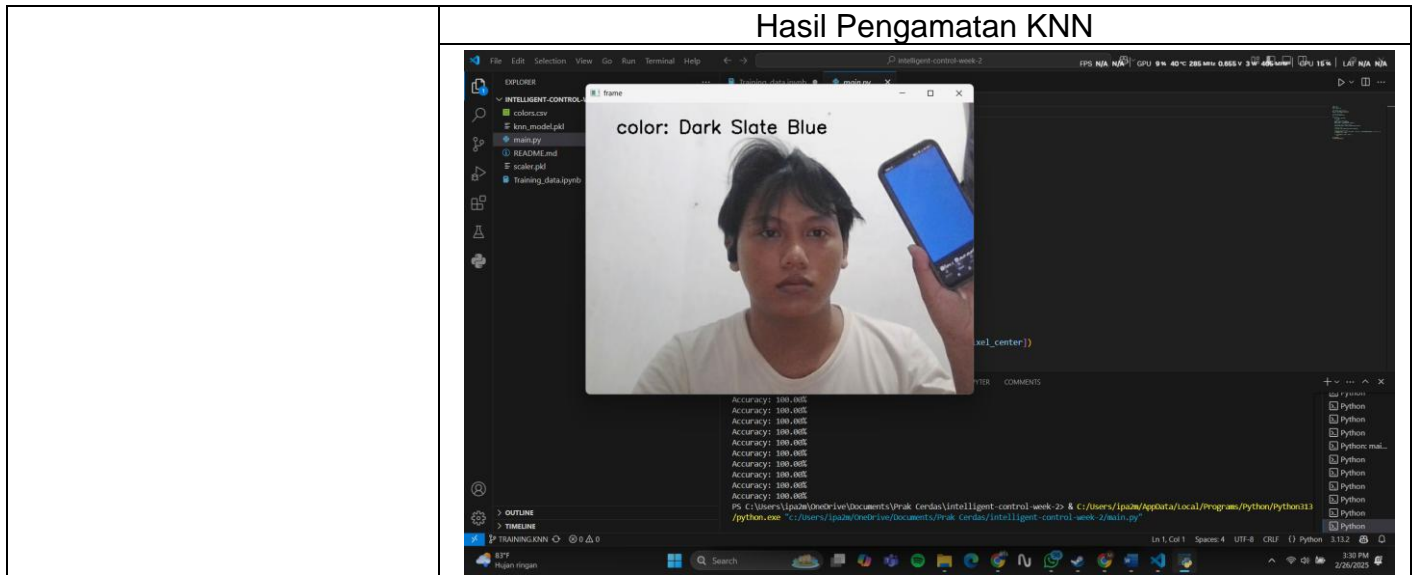
Program ini melakukan sejumlah tugas (penugasan variabel) untuk menyimpan data dan konfigurasi penting yang diperlukan dalam deteksi warna. Pertama, Dataset warna dikumpulkan dalam bentuk nilai RGB atau HSV, Agar model bekerja optimal, fitur dalam dataset distandarisasi menggunakan **StandardScaler()**, ini membantu model untuk berkonvergensi lebih cepat dan meningkatkan akurasi prediksi. Setiap data warna diberi label sesuai dengan klasifikasinya ('R', 'G', 'B'). Model *Decision Tree* dilatih menggunakan dataset yang telah diproses. Model disimpan dalam file **dtree\_model.pkl** untuk digunakan dalam deteksi real-time. Selanjutnya membuat model dan scaler **dtree = joblib.load('dtree\_model.pkl')** **scaler = joblib.load('scaler.pkl')** Model *Decision Tree* yang telah dilatih sebelumnya dimuat dari file. **StandardScaler** juga dimuat untuk memastikan data input memiliki skala yang sama. Selanjutnya inisiasi kamera **cap = cv2.VideoCapture(0)** kamera diaktifkan untuk menangkap frame secara real-time. Pada setiap frame mendeteksi warna dari kamera dengan konversi warna **hsv\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)** gambar diubah dari format BGR ke HSV untuk meningkatkan akurasi deteksi warna, dengan Gaussian blur dapat diterapkan untuk mengurangi noise pada gambar sehingga warna mudah untuk dikenali **blurred = cv2.GaussianBlur(hsv\_frame, (15, 15), 0)**. Menampilkan warna yang terdeteksi dengan kotak yang dibuat disekitar area warna yang terdeteksi, dan nama warna ditampilkan pada layer. Terdapat akurasi di layar, dihitung berdasarkan jumlah deteksi warna yang sesuai dengan ekspektasi

Hasil prediksi serta akurasi ditampilkan di layar kamera dengan informasi yang diperbarui secara dinamis berdasarkan *running accuracy* dari 50 prediksi terakhir. Penugasan variabel yang terstruktur dengan baik dalam program ini memungkinkan aliran data yang lancar dari *preprocessing*, *training model*, hingga prediksi *real-time*.

## 6. Data dan Output Hasil Pengamatan

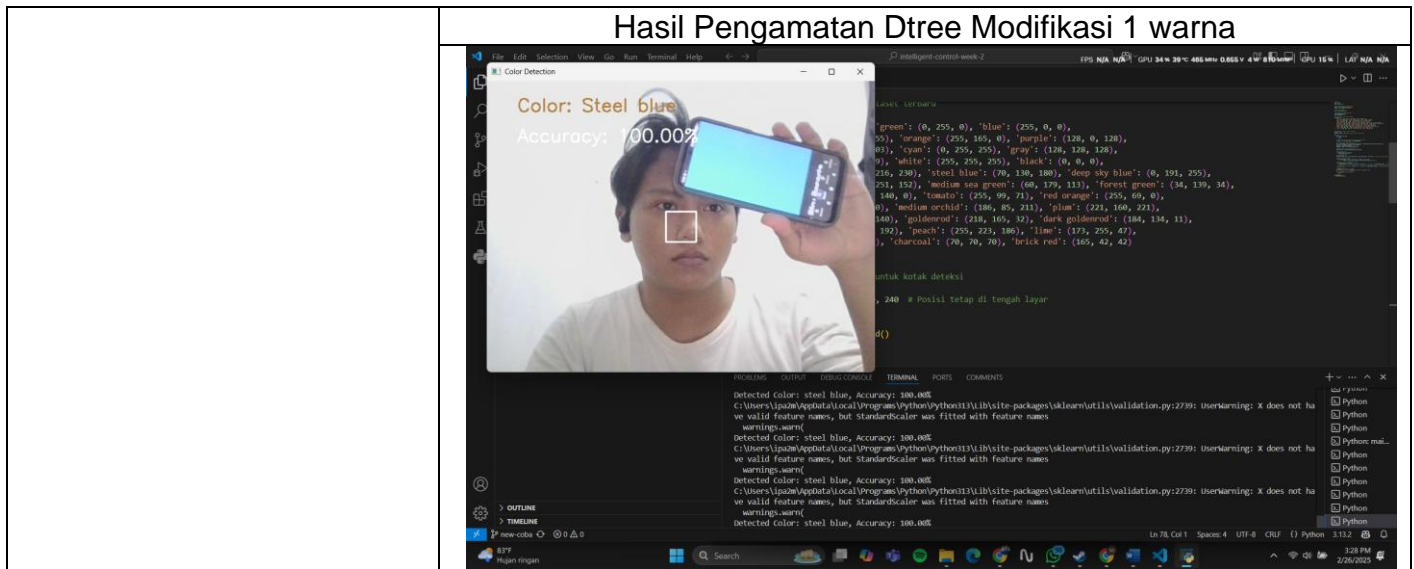
Hasil Pengamatan Machine Learning Model KNN & Model Decision Tree

### Hasil Pengamatan KNN



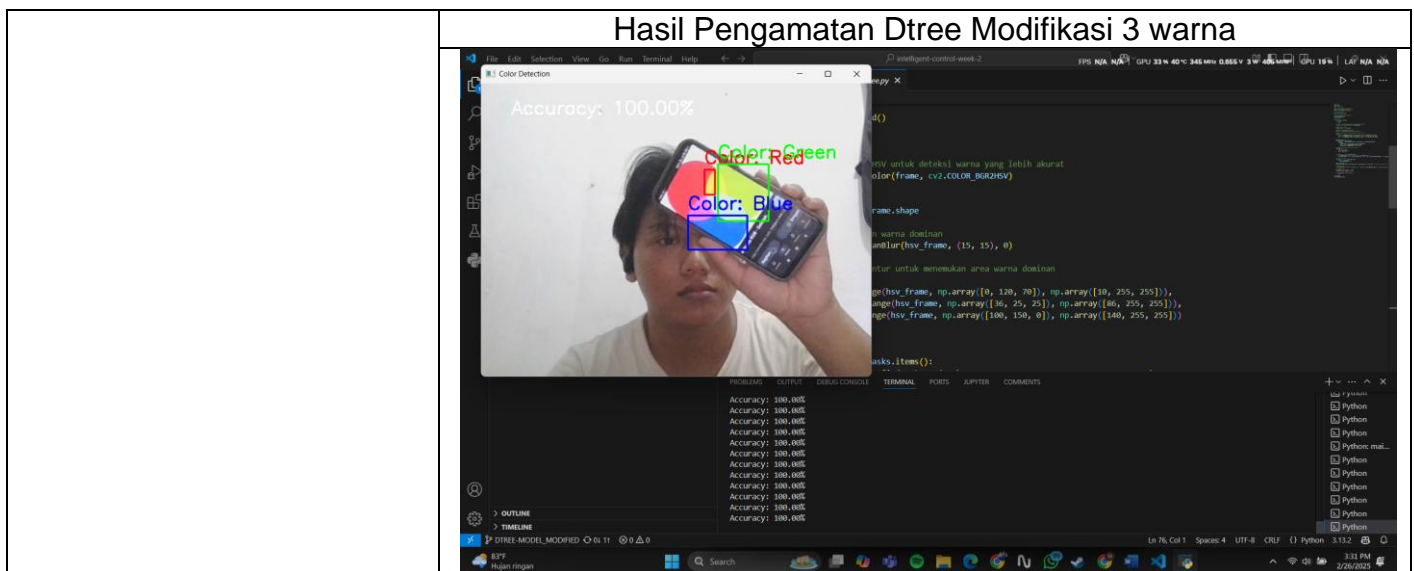
The screenshot shows a KNN model's output. A person is holding a smartphone with a blue screen. The text "color: Dark Slate Blue" is displayed above the phone. The background is a dark IDE interface with a file explorer on the left and a terminal at the bottom.

### Hasil Pengamatan Dtree Modifikasi 1 warna



The screenshot shows a Decision Tree model's output. A person is holding a smartphone with a blue screen. The text "Color: Steel blue" and "Accuracy: 100.00%" is displayed above the phone. The background is a dark IDE interface with a file explorer on the left and a terminal at the bottom.

### Hasil Pengamatan Dtree Modifikasi 3 warna



The screenshot shows a Decision Tree model's output. A person is holding a smartphone with a blue screen. The text "Color: Green", "Color: Red", and "Color: Blue" is displayed above the phone. The background is a dark IDE interface with a file explorer on the left and a terminal at the bottom.

## 7. Kesimpulan

Dari percobaan yang telah dilakukan dapat disimpulkan bahwa:

1. Berdasarkan praktikum dan analisis yang telah dilakukan, maka dapat diambil kesimpulan yaitu:
2. Program ini untuk mendeteksi warna dengan menggunakan *Decision Tree*.
3. Menggunakan *StandardScaler* membantu model untuk berkonvergensi lebih cepat dan meningkatkan akurasi prediksi.
4. *Running accuracy* memberikan evaluasi hasil deteksi model yang lebih akurat dan dinamis.
5. Hasil prediksi dan keakuratan objek warna bergantung pada pencahayaan dan kualitas kamera.

## 8. Saran

Untuk meningkatkan akurasi yang tepat, terdapat beberapa hal yang dapat dilakukan yaitu di antaranya, menggunakan *color correction* pada tahap *processing*. Kemudian, agar agar model lebih adaptif terhadap perubahan pencahayaan dengan menambahkan *adaptive thresholding*. Selain itu, perluas dataset warna dengan lebih banyak variasi untuk meningkatkan kemampuan generalisasi model dalam mengenali warna lebih akurat.

## 9. Daftar Pustaka

- Daqiqil, I. (2021). *Machine Learning: Teori, Studi Kasus, dan Implementasi Menggunakan Python*. Riau: UR PRESS.
- Fathurohman, A. (2021). Machine Learning untuk Pendidikan: Mengapa dan Bagaimana. 1(3). *Karimah Tauhid, Volume 2 Nomor 1 (2023), e-ISSN 2963-590X*. (2023). 2.
- Roihan, A., Sunarya, P. A., & Rafika, A. S. (2020). Pemanfaatan Machine Learning dalam Berbagai Bidang: Review paper. *IJCIT (Indonesian Journal on Computer and Information Technology)*, 5(1). <https://doi.org/10.31294/ijcit.v5i1.7951>
-