Project name: [Klasifikasi]_Submission_Akhir_BMLP_Ilham_Pratama

Project Summary

- Struktur Alur Analisis: Mulai dari persiapan data (penghapusan kolom tertentu), *split* data, pelatihan model, hingga evaluasi dengan metrik yang lengkap (*accuracy, precision, recall, f1-score*) dan *confusion matrix*.
- Multi-Model Approach: Menggunakan berbagai model klasifikasi: KNN, Decision Tree, Random Forest, SVM, dan Naïve Bayes. Ini memberi perbandingan kinerja yang luas.
- Visualisasi Confusion Matrix: Penggunaan *heatmap* untuk memudahkan interpretasi hasil klasifikasi masing-masing model.
- Saya juga menambahkan Hyperparameter Tuning dengan GridSearchCV: Melakukan pencarian parameter terbaik untuk Random Forest dan KNN, sehingga dapat melihat *trade-off* performa model sebelum dan sesudah *tuning*.
- Modularisasi Kode dan *Code Efficiency*: Kode bisa dibuat lebih ringkas dengan membuat fungsi-fungsi terpisah untuk training, evaluasi, dan visualisasi, sehingga menghindari duplikasi baris.
- Analisis Kegunaan Fitur yang Dihapus: Kolom *species*, *w_l_ratio*, *length*, dan *weight* dihapus tanpa analisis lebih lanjut mengenai korelasinya dengan *target*. Ada kemungkinan beberapa fitur justru relevan atau berguna bagi performa model.

Error Notes

Pada project yang diperiksa terjadi sebuah error Karna data berpotensi terjadi data leakage dan saya mengatasinya dengan cara memiisahkan data train/test secara lebih teliti dan lakukan cross-validation agar hasil evaluasi lebih obyektif. Dan juga Kolom *species*, *length*, *weight*, dan *w_l_ratio* langsung dihapus tanpa melihat seberapa besar korelasinya dengan *cluster*.

Code Review

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler, MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
```

Pertimbangkan menambahkan tqdm atau *logging* jika ingin memantau proses pada dataset yang lebih besar.

```
# Baca file CSV dari URL
data = pd.read_csv('/content/datasetdalamnotebookklasifikasi123.csv')

data.head(50)
```

```
data.info()
data.describe()

data = data.drop(columns=['species','w_l_ratio','length','weight'])
```

Pada kode tersebut sudah menampilkan *head*, *info*, dan *describe* untuk *overview* dataset.
Namun akan lebih baik untuk memastikan memeriksa *missing values* dan *duplicates*.
dapat digunakan sebagai berikut :
“data.isnull().sum()”
“data.duplicated().sum()”

```
knn = KNeighborsClassifier().fit(X_train, y_train) dt = DecisionTreeClassifier().fit(X_train, y_train) rf = RandomForestClassifier().fit(X_train, y_train) svm = SVC().fit(X_train, y_train) nb = GaussianNB().fit(X_train, y_train)
```

Tidak ada langkah normalisasi/penskalaan data sebelum training. Untuk model KNN dan SVM. Karna itu saya sangat menyarankan untuk menerapkan *scaling*.
Lakukan *cross validation* jika dataset cukup besar, agar hasil evaluasi tidak tergantung pada satu kali *split* data.

```
plt.figure(figsize=(5, 4)) sns.heatmap(cm_knn, annot=True, fmt='d', cmap='Blues', cbar=False) plt.title('KNN Confusion Matrix') plt.xlabel('Predicted') plt.ylabel('Actual') plt.show()
```

Anda bisa menambahkan label nama kelas jika ada label yang jelas, atau menambahkan *normalisasi* agar matrix lebih intuitif saat jumlah data antar kelas tidak seimbang.
Classification report (dari sklearn.metrics.classification_report) dapat membantu merinci metrik setiap kelas (Precision, Recall, F1) dalam satu tampilan.

Rekomendasi:

- Terapkan Normalisasi/Standarisasi: Sebelum melatih model KNN atau SVM, gunakan *MinMaxScaler* atau *StandardScaler*.
- Lakukan Cross-Validation: Untuk mengecek kestabilan performa model secara keseluruhan di berbagai *split* data.
- Cek Kebutuhan Fitur yang Dihapus: Gunakan *feature importance* atau analisis korelasi sebelum menghapus kolom *species, length, weight, w_l_ratio*. Bisa jadi beberapa masih relevan.
- Modularisasi Kode: Buat fungsi-fungsi terpisah (misalnya train_model(), evaluate_model()) agar kode lebih rapi dan reusable.
- Eksplorasi Metode Lain: Seperti XGBoost atau LightGBM, terutama jika dataset bertambah besar atau kompleks.

Kesimpulan Akhir:
Proyek ini sudah memenuhi tahapan penting dalam proses *machine learning classification*, mulai dari pemisahan *feature* dan *target*, penggunaan beberapa algoritma, hingga evaluasi model yang cukup mendalam dengan *confusion matrix* dan metrik akurasi, presisi, *recall*, serta F1-Score.