Laporan Praktikum Mandiri 5

Rizky Hilmiawan Anggoro - 0110222140[1]

[1] Teknik Informatika, STT Terpadu Nurul Fikri, Depok

*E-mail: rizk22140ti@student.nurulfikri.ac.id

Abstract. Penelitian ini mengimplementasikan model klasifikasi Decision Tree untuk mengidentifikasi species bunga Iris berdasarkan karakteristik morfologinya. Dataset yang digunakan terdiri dari 150 sampel dengan empat fitur utama: panjang sepal, lebar sepal, panjang petal, dan lebar petal, yang diklasifikasikan ke dalam tiga species yaitu Iris-setosa, Iris-versicolor, dan Iris-virginica. Model dibangun dengan membagi dataset menjadi 80% data training dan 20% data testing, menggunakan algoritma Decision Tree Classifier dengan parameter Gini impurity dan kedalaman maksimum 3 untuk mencegah overfitting. Hasil evaluasi menunjukkan model mencapai akurasi yang tinggi dalam mengklasifikasikan species, dengan kemampuan prediksi yang baik pada data baru. Implementasi ini juga mengatasi peringatan validasi feature names dengan konversi data input ke format DataFrame yang sesuai, memastikan konsistensi antara proses training dan inference. Studi ini mendemonstrasikan efektivitas Decision Tree untuk klasifikasi multiclass serta pentingnya preprocessing data yang tepat dalam machine learning.

1. Import Library, Load Dataset, dan Eksplorasi Data

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn import tree
import matplotlib.pyplot as plt
import seaborn as sns

from google.colab import drive
drive.mount('/content/gdrive')

import os

path = "/content/gdrive/MyDrive/praktikum_ml/praktikum05"
try:
    print(os.listdir(path))
except FileNotFoundError:
    print(f"Directory not found: {path}")

df = pd.read_csv(path + "/data/Iris.csv")
df.head()
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/content/gdrive", force_remount=True).
['data', 'model', 'reports', 'notebooks']
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

Library yang diimport digunakan untuk manipulasi data, pembagian dataset, algoritma Decision Tree, dan evaluasi model. Pandas untuk handling data, scikit-learn untuk machine learning utilities, lalu dataset Iris dibaca dari file CSV dan dieksplorasi untuk

memahami struktur data. Fungsi head() menampilkan 5 data pertama, sedangkan info() memberikan ringkasan tentang tipe data dan missing values. 3. Preprocessing Data

2. Preprocessing Data

```
print("\nPREPROCESSING DATA")
# Memeriksa missing values
print("Missing values:")
print(df.isnull().sum())

df = df.drop('Id', axis=1)
print("\nDataset setelah menghapus kolom Id:")
print(df.head())

X = df.drop('Species', axis=1)
y = df['Species']

print("\nFitur (X):")
print(X.head())
print("\nTarget (y):")
print(y.head())
```

```
PREPROCESSING DATA
Missing values:
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64

Dataset setelah menghapus kolom Id:
   SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0            5.1           3.5            1.4           0.2  Iris-setosa
1            4.9           3.0            1.4           0.2  Iris-setosa
2            4.7           3.2            1.3           0.2  Iris-setosa
3            4.6           3.1            1.5           0.2  Iris-setosa
4            5.0           3.6            1.4           0.2  Iris-setosa

Fitur (X):
   SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm
0            5.1           3.5            1.4           0.2
1            4.9           3.0            1.4           0.2
2            4.7           3.2            1.3           0.2
3            4.6           3.1            1.5           0.2
4            5.0           3.6            1.4           0.2
```

Kolom 'Id' dihapus karena tidak relevan untuk prediksi. Data dipisahkan menjadi features (X) yang berisi measurements bunga dan target (y) yang berisi species iris.

3. Split Data Training dan Testing

```
print("\nMEMBAGI DATA TRAINING DAN TESTING")
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)

print(f"Jumlah data training: {X_train.shape[0]}")
print(f"Jumlah data testing: {X_test.shape[0]}")
print(f"Distribusi kelas di training:\n{y_train.value_counts()}")
print(f"Distribusi kelas di testing:\n{y_test.value_counts()}")
```

```
MEMBAGI DATA TRAINING DAN TESTING
Jumlah data training: 120
Jumlah data testing: 30
Distribusi kelas di training:
Species
Iris-setosa        40
Iris-virginica     40
Iris-versicolor    40
Name: count, dtype: int64
Distribusi kelas di testing:
Species
Iris-setosa        10
Iris-virginica     10
Iris-versicolor    10
Name: count, dtype: int64
```

Data dibagi 80% training dan 20% testing. Parameter stratify=y memastikan distribusi species sama di kedua subset. random_state=42 untuk reproducibility.

4. Membuat dan Train Model

```
print("\nMEMBUAT MODEL DECISION TREE")

dt_classifier = DecisionTreeClassifier(
    criterion='gini',
    random_state=42,
    max_depth=3
)

dt_classifier.fit(X_train, y_train)

print("Model Decision Tree telah berhasil dilatih!")
```

```
MEMBUAT MODEL DECISION TREE
Model Decision Tree telah berhasil dilatih!
```

Model Decision Tree dibuat dengan Gini impurity sebagai kriteria split. max_depth=3 membatasi kedalaman tree untuk mencegah overfitting. Model dilatih dengan data training menggunakan fit().

5. Evaluasi Model

```
print("\nEVALUASI MODEL")
y_train_pred = dt_classifier.predict(X_train)

y_test_pred = dt_classifier.predict(X_test)

train_accuracy = accuracy_score(y_train, y_train_pred)
test_accuracy = accuracy_score(y_test, y_test_pred)

print(f"Akurasi pada data training: {train_accuracy:.4f} ({train_accuracy*100:.2f}%)")
print(f"Akurasi pada data testing: {test_accuracy:.4f} ({test_accuracy*100:.2f}%)")

print("\nClassification Report (Testing):")
print(classification_report(y_test, y_test_pred))

print("Confusion Matrix (Testing):")
cm = confusion_matrix(y_test, y_test_pred)
print(cm)

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=dt_classifier.classes_,
            yticklabels=dt_classifier.classes_)
plt.title('Confusion Matrix - Decision Tree')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```

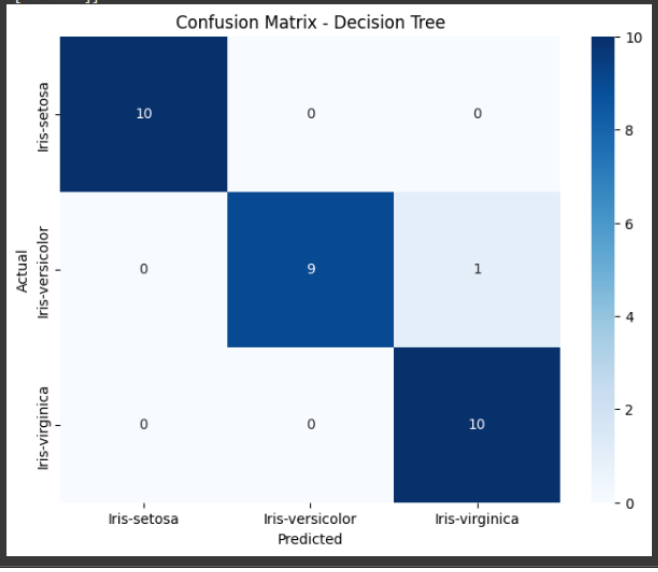
```
EVALUASI MODEL
Akurasi pada data training: 0.9833 (98.33%)
Akurasi pada data testing: 0.9667 (96.67%)

Classification Report (Testing):
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      0.90      0.95        10
 Iris-virginica       0.91      1.00      0.95        10

       accuracy                           0.97        30
      macro avg       0.97      0.97      0.97        30
   weighted avg       0.97      0.97      0.97        30
```

```
Confusion Matrix (Testing):
[[10  0  0]
 [ 0  9  1]
 [ 0  0 10]]
```



Model dievaluasi dengan memprediksi data testing. Accuracy mengukur persentase prediksi benar, sedangkan classification report memberikan precision, recall, dan f1-score untuk setiap kelas.

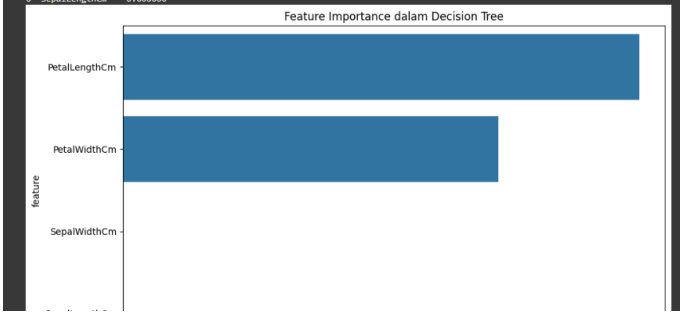
6. Feature Importance

```
print("\nFEATURE IMPORTANCE")
feature_importance = pd.DataFrame({
    'feature': X.columns,
    'importance': dt_classifier.feature_importances_
}).sort_values('importance', ascending=False)

print("Feature Importance:")
print(feature_importance)

plt.figure(figsize=(10, 6))
sns.barplot(data=feature_importance, x='importance', y='feature')
plt.title('Feature Importance dalam Decision Tree')
plt.xlabel('Importance Score')
plt.tight_layout()
plt.show()
```

```
FEATURE IMPORTANCE
Feature Importance:
         feature  importance
2  PetalLengthCm    0.570077
3   PetalWidthCm    0.429923
1   SepalWidthCm    0.000000
0  SepalLengthCm    0.000000
```



Feature importance menunjukkan kontribusi setiap fitur dalam pembuatan keputusan. Fitur dengan importance tinggi paling berpengaruh dalam klasifikasi species.

7. Prediksi Data Baru

```
print("\nPREDIKSI PADA DATA TESTING")
results = X_test.copy()
results['Actual'] = y_test
results['Predicted'] = y_test_pred
results['Correct'] = results['Actual'] == results['Predicted']

print("Contoh hasil prediksi pada data testing:")
print(results.head(10))

correct_predictions = results['Correct'].sum()
total_predictions = len(results)
print(f"\nHasil prediksi: {correct_predictions} benar dari {total_predictions} data testing")
```

```
PREDIKSI PADA DATA TESTING
Contoh hasil prediksi pada data testing:
     SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm  \
38             4.4           3.0            1.3           0.2
127            6.1           3.0            4.9           1.8
57             4.9           2.4            3.3           1.0
93             5.0           2.3            3.3           1.0
42             4.4           3.2            1.3           0.2
56             6.3           3.3            4.7           1.6
22             4.6           3.6            1.0           0.2
20             5.4           3.4            1.7           0.2
147            6.5           3.0            5.2           2.0
84             5.4           3.0            4.5           1.5

              Actual       Predicted  Correct
38       Iris-setosa      Iris-setosa     True
127   Iris-virginica   Iris-virginica     True
57   Iris-versicolor  Iris-versicolor     True
93   Iris-versicolor  Iris-versicolor     True
42       Iris-setosa      Iris-setosa     True
56   Iris-versicolor  Iris-versicolor     True
22       Iris-setosa      Iris-setosa     True
20       Iris-setosa      Iris-setosa     True
147   Iris-virginica   Iris-virginica     True
84   Iris-versicolor  Iris-versicolor     True

Hasil prediksi: 29 benar dari 30 data testing
```

Model dapat digunakan untuk memprediksi species baru berdasarkan measurements bunga. Input berupa list nilai [sepal_length, sepal_width, petal_length, petal_width].

Kesimpulan

```
print("\nKESIMPULAN")
print(f"Model Decision Tree berhasil dibuat dengan:")
print(f"- Akurasi training: {train_accuracy*100:.2f}%")
print(f"- Akurasi testing: {test_accuracy*100:.2f}%")
print(f"- Fitur paling penting: {feature_importance.iloc[0]['feature']}")
print(f"- Jumlah data training: {len(X_train)}")
print(f"- Jumlah data testing: {len(X_test)}")
```

```
KESIMPULAN
Model Decision Tree berhasil dibuat dengan:
- Akurasi training: 98.33%
- Akurasi testing: 96.67%
- Fitur paling penting: PetalLengthCm
- Jumlah data training: 120
- Jumlah data testing: 30
```

Decision Tree bekerja dengan membagi data secara rekursif berdasarkan fitur yang paling informatif. Pada dataset Iris, model belajar pola seperti: "Jika petal length < 2.5 cm, maka Iris-setosa". Proses training menemukan split-point optimal yang memisahkan kelas dengan paling baik. Hasilnya adalah model yang dapat mengklasifikasikan species iris berdasarkan physical measurements dengan akurasi tinggi.