

PRAKTIKUM
MMAI1004 – PEMBELAJARAN MESIN

PERTEMUAN III

DOSEN PENGASUH:

Prof. Dr. Taufik Fuadi Abidin, S.Si., M.Tech

Dr. Rumaisa Kruba, S.Si

Prof. Dr. Ir. Hammam Riza, M.Sc. IPU

Praktisi Komdigi (Dr. Said Mirza Pahlevi)

ASISTEN PRAKTIKUM:

Muhammad Chaidir, S.Kom, M.Kom

Hands on Decision Tree and Random Forest

1. Introduction

In this tutorial, we will explore the Decision Tree Classifier using the Bank Marketing Dataset from UCI. This dataset contains client information and whether they subscribed to a term deposit.

The objectives of this session include:

- ✓ Understanding Decision Trees for classification tasks.
- ✓ Preprocessing categorical and numerical data.
- ✓ Evaluating model performance using accuracy, confusion matrix, and classification report.
- ✓ Visualizing the Decision Tree and Feature Importance.
- ✓ Analyze and compare with Random Forest Classifier

2. Dataset: Bank Marketing Dataset

The dataset can be downloaded from dataset page:

<https://archive.ics.uci.edu/dataset/222/bank+marketing>

Features:

- ✓ Categorical: Job, marital status, education, default, housing, loan, contact, month, day_of_week, etc.
- ✓ Numerical: Age, duration, campaign, pdays, previous, etc.
- ✓ Target variable (y): Whether a client subscribed (1: Yes, 0: No)

3. Steps to Implement Decision Tree Classifier

Step 1: Load Dataset

```
import pandas as pd

# Load dataset from UCI repository
path = "../../datasets/bank-additional-full.csv"
data = pd.read_csv(path, sep=';')
```

Step 2: Data Preprocessing

Handling Missing Values

```
# Remove rows with missing values
data.dropna(inplace=True)
```

Encoding Categorical Features

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
categorical_cols = data.select_dtypes(include=['object']).columns

for col in categorical_cols:
    data[col] = label_encoder.fit_transform(data[col])
```

Splitting Features and Target Variable

```
X = data.drop(columns=['y']) # Features
y = data['y'] # Target Variable
```

Normalizing Numerical Features

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Step 3: Train-Test Split

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,
                                                    random_state=42)
```

Step 4: Train Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier(max_depth=4, random_state=42)
clf.fit(X_train, y_train)
```

Step 5: Model Evaluation

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

y_pred = clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)
```

Step 6: Visualizing the Decision Tree

```
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

plt.figure(figsize=(12, 8))
plot_tree(clf, feature_names=X.columns, class_names=['No', 'Yes'], filled=True)
plt.show()
```

Step 7: Feature Importance Analysis

```
import seaborn as sns

feature_importance = pd.DataFrame({'Feature': X.columns, 'Importance':
clf.feature_importances_})
feature_importance = feature_importance.sort_values(by='Importance',
ascending=False)

plt.figure(figsize=(10, 5))
sns.barplot(x='Importance', y='Feature', data=feature_importance)
plt.title('Feature Importance in Decision Tree')
plt.show()
```

4. Conclusion

- ✓ We trained a Decision Tree Classifier on the Bank Marketing Dataset.
- ✓ The model's performance was evaluated using Accuracy, Confusion Matrix, and Classification Report.
- ✓ We visualized the Decision Tree and analyzed the Feature Importance.
- ✓ This approach helps in understanding which factors influence client subscription decisions.

5. Further Exploration

- ✓ Modify `max_depth` and observe the model's performance.
- ✓ Experiment with Random Forest. Hint:
`clf = DecisionTreeClassifier(max_depth=4, random_state=42)`
- ✓ Analyze and compare the result between Decision Tree and Random Forest.