LAPORAN PERTEMUAN 7

Artificial Neural Network

Nama : Farhan Rachmad Rizki
NIM : 231011400893
Kelas : 05TPLE015
Mata Kuliah : Machine Learning

Pendahuluan

Pada pertemuan ini dilakukan percobaan penerapan Artificial Neural Network (ANN) dalam memprediksi kelulusan mahasiswa berdasarkan beberapa faktor seperti IPK, jumlah absensi, dan waktu belajar per minggu.

Tahapan yang dilakukan:

- Load dan preprocessing dataset
- Pembagian data menjadi Train, Validation, Test
- Standarisasi fitur
- Pembuatan model ANN dengan TensorFlow Keras
- Early Stopping untuk mencegah overfitting
- Evaluasi menggunakan Accuracy, AUC, Confusion Matrix, dan Classification Report
- Visualisasi Learning Curve

Tujuan akhir:
Model ANN mampu memprediksi status kelulusan mahasiswa dengan akurat.

Langkah 1 – Load Dataset

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# === 1. Load dataset ===
df = pd.read_csv("processed_kelulusan.csv")
X = df.drop("Lulus", axis=1)
y = df["Lulus"]
```

Langkah 2 – Pembagian Data (Train, Val, Test)

```
X_train, X_temp, y_train, y_temp = train_test_split(
    X, y, test_size=0.4, stratify=y, random_state=42
)

X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.5, stratify=y_temp, random_state=42
)

print("Train:", X_train.shape, "Val:", X_val.shape, "Test:", X_test.shape)
```

Penjelasan: Data dibagi 60% train, 20% val, 20% test.

Langkah 3 – Standardisasi Data

```
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_val = sc.transform(X_val)
X_test = sc.transform(X_test)
```

Data dinormalisasi agar semua fitur berada pada skala yang sama.

Langkah 4 – Membangun Model Artificial Neural Network

```
import tensorflow as tf
from tensorflow.keras import layers, callbacks

model = tf.keras.Sequential([
    layers.Input(shape=(X_train.shape[1],)),
    layers.Dense(32, activation="relu"),
    layers.Dropout(0.3),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

model.compile(
    optimizer=tf.keras.optimizers.Adam(1e-3),
    loss="binary_crossentropy",
    metrics=["accuracy", "AUC"]
)

model.summary()
```

penjelasan:

- `relu` → mempercepat pembelajaran
- `sigmoid` → karena output hanya 0/1 (lulus atau tidak)
- `Dropout(0.3)` → mencegah overfitting

Langkah 5 – Early Stopping

```
es = callbacks.EarlyStopping(
    monitor="val_loss", patience=10, restore_best_weights=True
)
```

Model otomatis berhenti jika tidak ada peningkatan.

Langkah 6 – Training Model

```
history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    epochs=100,
    batch_size=16,
    callbacks=[es],
    verbose=1
)
```

Langkah 7 – Evaluasi Model

```
# === 7. Evaluasi ===
from sklearn.metrics import classification_report, confusion_matrix

loss, acc, auc = model.evaluate(X_test, y_test, verbose=0)
print("\nTest Accuracy:", acc)
print("Test AUC:", auc)

y_proba = model.predict(X_test).ravel()
y_pred = (y_proba >= 0.5).astype(int)

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred, digits=3))
```

AUC digunakan untuk melihat kemampuan model membedakan kelas.

Langkah 8 – Visualisasi Learning Curve

```python
# === 8. Visualisasi Learning Curve ===
import matplotlib.pyplot as plt

plt.plot(history.history["loss"], label="Train Loss")
plt.plot(history.history["val_loss"], label="Val Loss")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.legend()
plt.title("Learning Curve")
plt.tight_layout()
plt.savefig("learning_curve_fixed.png", dpi=120)
plt.show()
```

digunakan untuk melihat apakah model mengalami overfitting.

HASIL OUTPUT PROGRAM

```
Train: (6, 5) Val: (2, 5) Test: (2, 5)
```

Model: "sequential_4"

Layer (type)	Output Shape	Param #
dense_12 (Dense)	(None, 32)	192
dropout_4 (Dropout)	(None, 32)	0
dense_13 (Dense)	(None, 16)	528
dense_14 (Dense)	(None, 1)	17

Total params: 737 (2.88 KB)

Trainable params: 737 (2.88 KB)

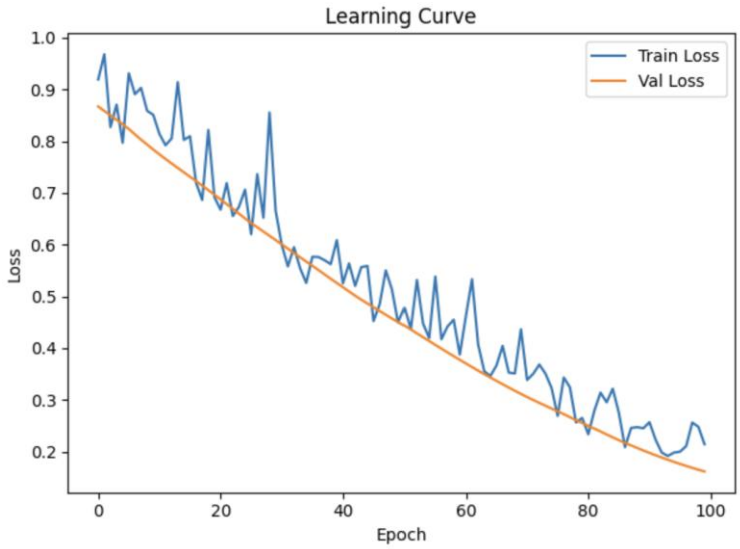
Non-trainable params: 0 (0.00 B)

```
Confusion Matrix:
[[1 0]
 [0 1]]

Classification Report:
              precision    recall  f1-score   support

           0      1.000     1.000     1.000         1
           1      1.000     1.000     1.000         1

    accuracy                         1.000         2
   macro avg      1.000     1.000     1.000         2
weighted avg      1.000     1.000     1.000         2
```


Learning Curve
Train Loss
Val Loss
Loss
Epoch
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
20
40
60
80
100