Aura Indri Aprilia
231011403707
Machine Learning

PERTEMUAN 7

Selanjutnya saya akan masuk ke lembar kerja pertemuan ke-7 Artificial Neural Network (ANN) untuk Klasifikasi

Langkah pertama yaitu Siapkan Data terlebih dahulu disini saya menggunakan processed_kelulusan.csv (hasil Pertemuan 4) atau dataset tabular sejenis.

```
[1]
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

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

sc = StandardScaler()
Xs = sc.fit_transform(X)

X_train, X_temp, y_train, y_temp = train_test_split(
    Xs, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.5,  random_state=42)

print(X_train.shape, X_val.shape, X_test.shape)
```

```
... (7, 5) (1, 5) (2, 5)
```

- X_train memiliki 7 sampel dan 5 fitur → digunakan untuk melatih model.
- X_val memiliki 1 sampel dan 5 fitur → digunakan untuk validasi model selama tuning.
- X_test memiliki 2 sampel dan 5 fitur → digunakan untuk evaluasi akhir model.

Data awal di-scaling menggunakan StandardScaler, sehingga seluruh fitur memiliki distribusi standar (mean = 0, std = 1).

Pembagian data dilakukan dalam dua tahap:

- Pertama: 70% data untuk pelatihan, 30% sisanya untuk validasi dan pengujian.
- Kedua: 30% sisa dibagi lagi menjadi 50% untuk validasi dan 50% untuk pengujian.

Ukuran dataset yang kecil (total 10 sampel) membuat evaluasi model perlu dilakukan dengan hati-hati agar tidak overfitting atau menghasilkan kesimpulan yang terlalu optimis.

Sebelumnya saya belum memiliki library TensorFlow di lingkungan Python saya, sehingga saya menjalankan perintah pip install tensorflow untuk mengunduh dan menginstalnya beserta semua dependensi yang diperlukan.

```
[5]  !pip install tensorflow

...  Collecting tensorflow
       Using cached tensorflow-2.20.0-cp310-cp310-win_amd64.whl.metadata (4.6 kB)
     Collecting absl-py>=1.0.0 (from tensorflow)
       Using cached absl_py-2.3.1-py3-none-any.whl.metadata (3.3 kB)
     Collecting astunparse>=1.6.0 (from tensorflow)
       Using cached astunparse-1.6.3-py2.py3-none-any.whl.metadata (4.4 kB)
     Collecting flatbuffers>=24.3.25 (from tensorflow)
       Using cached flatbuffers-25.9.23-py2.py3-none-any.whl.metadata (875 bytes)
     Collecting gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 (from tensorflow)
       Using cached gast-0.6.0-py3-none-any.whl.metadata (1.3 kB)
```

Lanjut ke Langkah ke 2 yaitu bangun model ANN, saya menggunakan kode tersebut

```
[2]  import tensorflow as tf
     from tensorflow import keras
     from tensorflow.keras import layers

     model = 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")  # klasifikasi biner
     ])

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

Dan menghasilkan output sebagai berikut

```
Model: "sequential"
```

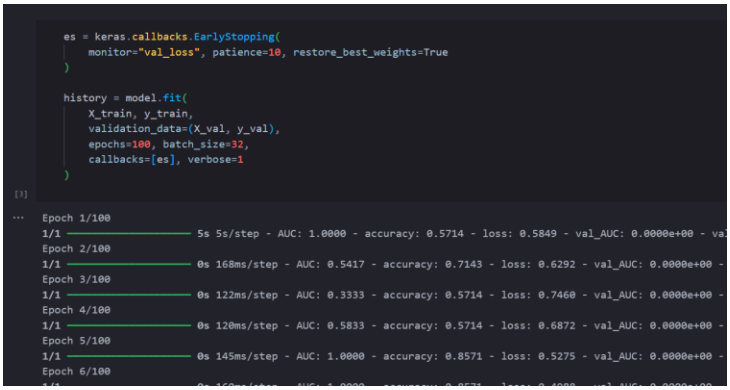
Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	192
dropout (Dropout)	(None, 32)	0
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 1)	17

```
Total params: 737 (2.88 KB)

Trainable params: 737 (2.88 KB)

Non-trainable params: 0 (0.00 B)
```

Lanjut ke Langkah selanjutnya yaitu pertemuan ke Training dengan Early Stopping



Langkah ke 4 yaitu Evaluasi di Test Set

```
[4]
from sklearn.metrics import classification_report, confusion_matrix

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

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

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred, digits=3))
```

```
Test Acc: 1.0 AUC: 1.0
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 200ms/step
[[1 0]
 [0 1]]
              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
```

Dan lanjut ke Langkah yang terakhir yaitu Visualisasi Learning Curve

```
[5]
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.png", dpi=120)
```

Yang menghasilkan output seperti berikut