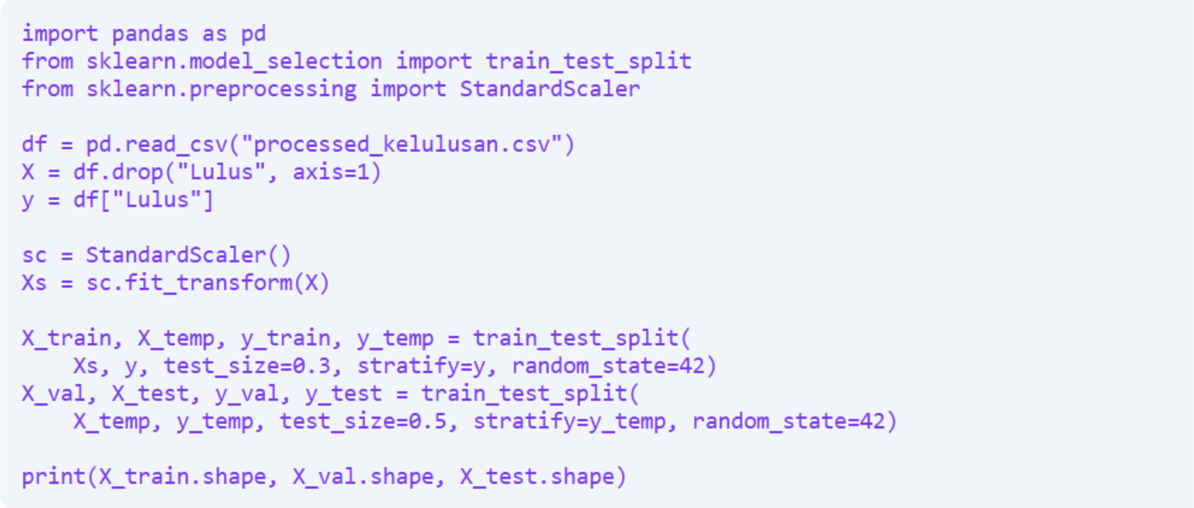
Laporan pengerjaan Pertemuan 7

Langkah 1 — Siapkan Data

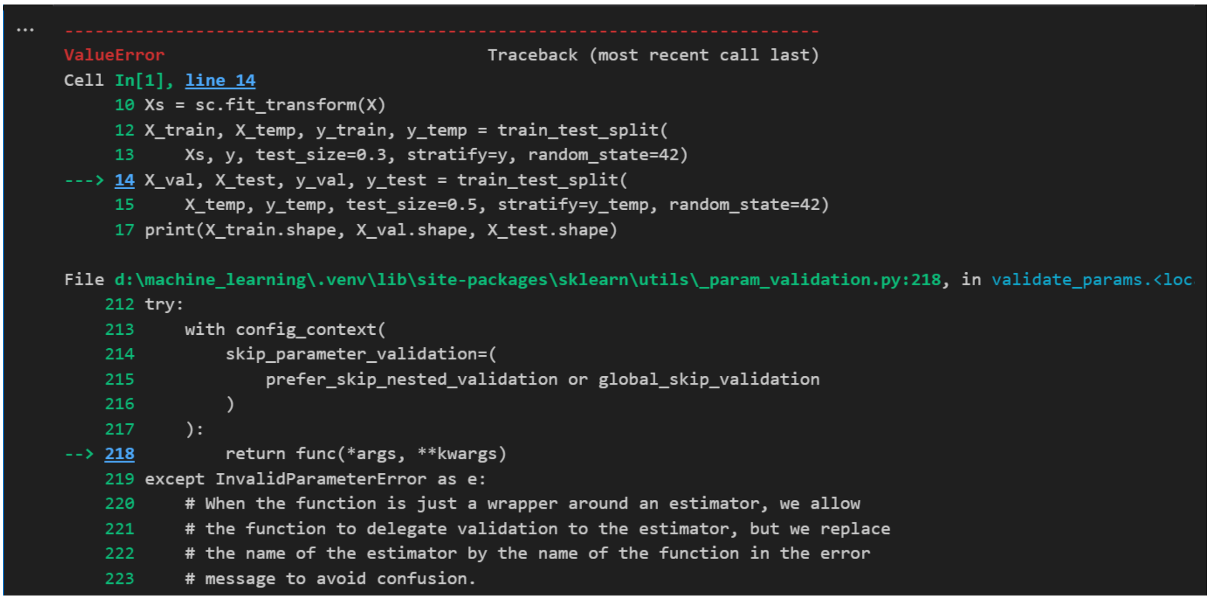
Gunakan processed_kelulusan.csv (hasil Pertemuan 4) atau dataset tabular sejenis.

1. Copy paste kode dari lembar kerja pertemuan 7 ke vscode



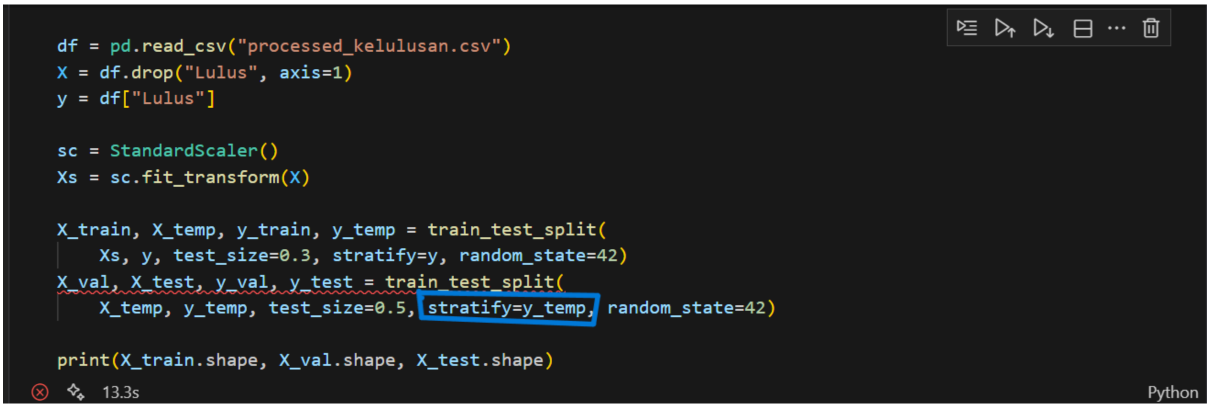
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

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

2. Setelah di ketik ada erornya seperti ini



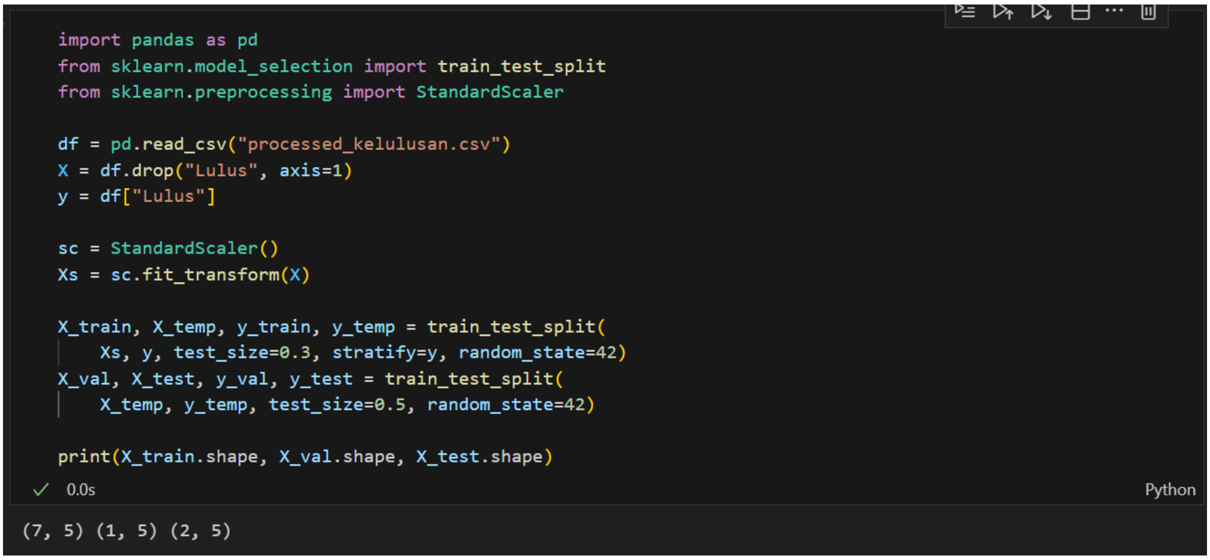
```
...  ---------------------------------------------------------------------------
     ValueError                                Traceback (most recent call last)
     Cell In[1], line 14
          10 Xs = sc.fit_transform(X)
          12 X_train, X_temp, y_train, y_temp = train_test_split(
          13     Xs, y, test_size=0.3, stratify=y, random_state=42)
     ---> 14 X_val, X_test, y_val, y_test = train_test_split(
          15     X_temp, y_temp, test_size=0.5, stratify=y_temp, random_state=42)
          17 print(X_train.shape, X_val.shape, X_test.shape)

     File d:\machine_learning\.venv\lib\site-packages\sklearn\utils\_param_validation.py:218, in validate_params.<loc
         212 try:
         213     with config_context(
         214         skip_parameter_validation=(
         215             prefer_skip_nested_validation or global_skip_validation
         216         )
         217     ):
     --> 218         return func(*args, **kwargs)
         219 except InvalidParameterError as e:
         220     # When the function is just a wrapper around an estimator, we allow
         221     # the function to delegate validation to the estimator, but we replace
         222     # the name of the estimator by the name of the function in the error
         223     # message to avoid confusion.
```

3. Hapus yang ditandai supaya tidak eror



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

4. Setelah itu dapat hasilnya seperti ini

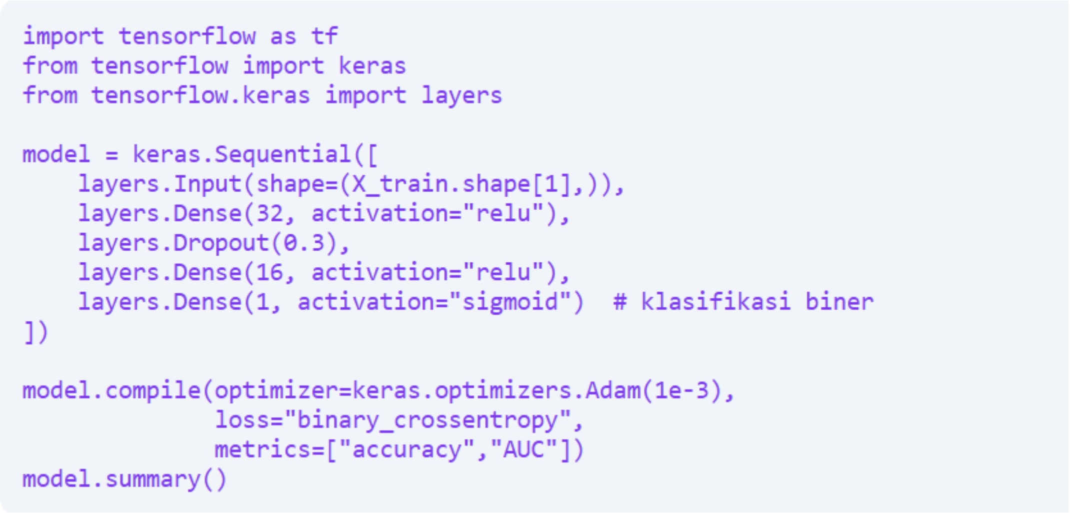


```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

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, stratify=y, 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)
```

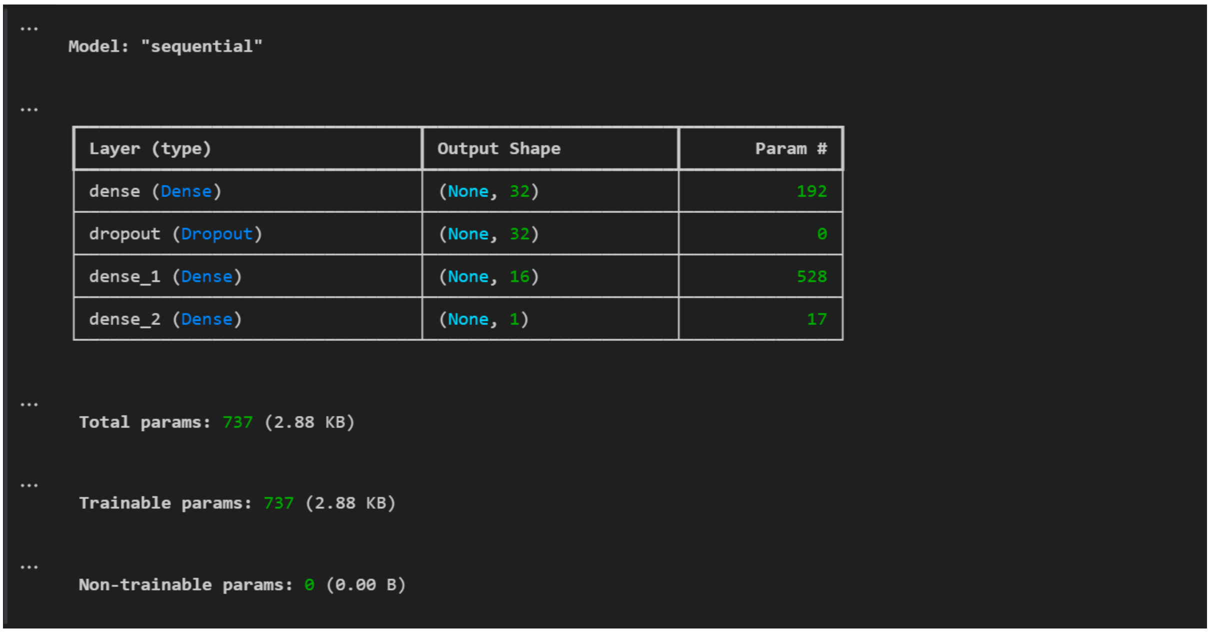
Langkah 2 — Bangun Model ANN

1. Copy paste dari lembar kerja ke vscode



```python
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()
```

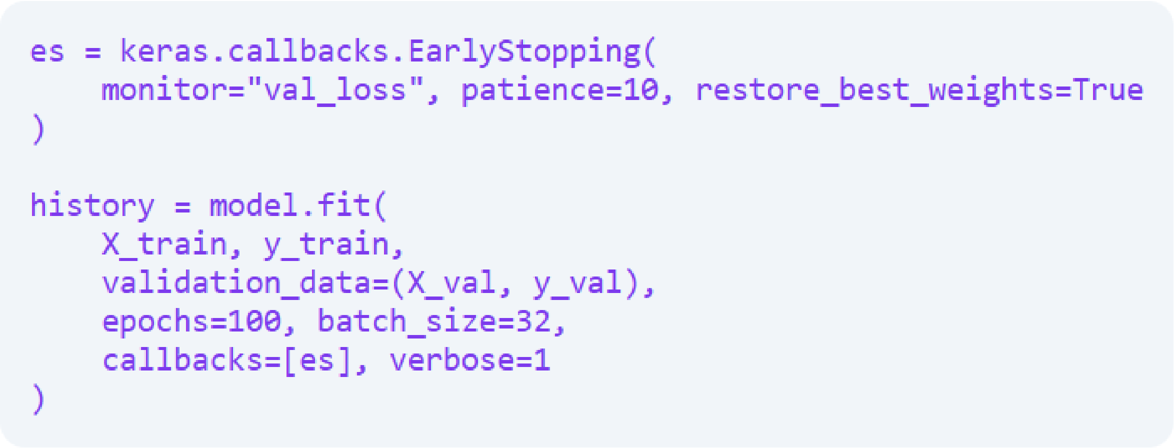
2. Setelah copy paste dapat hasilnya seperti ini :



```
...
   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)
```

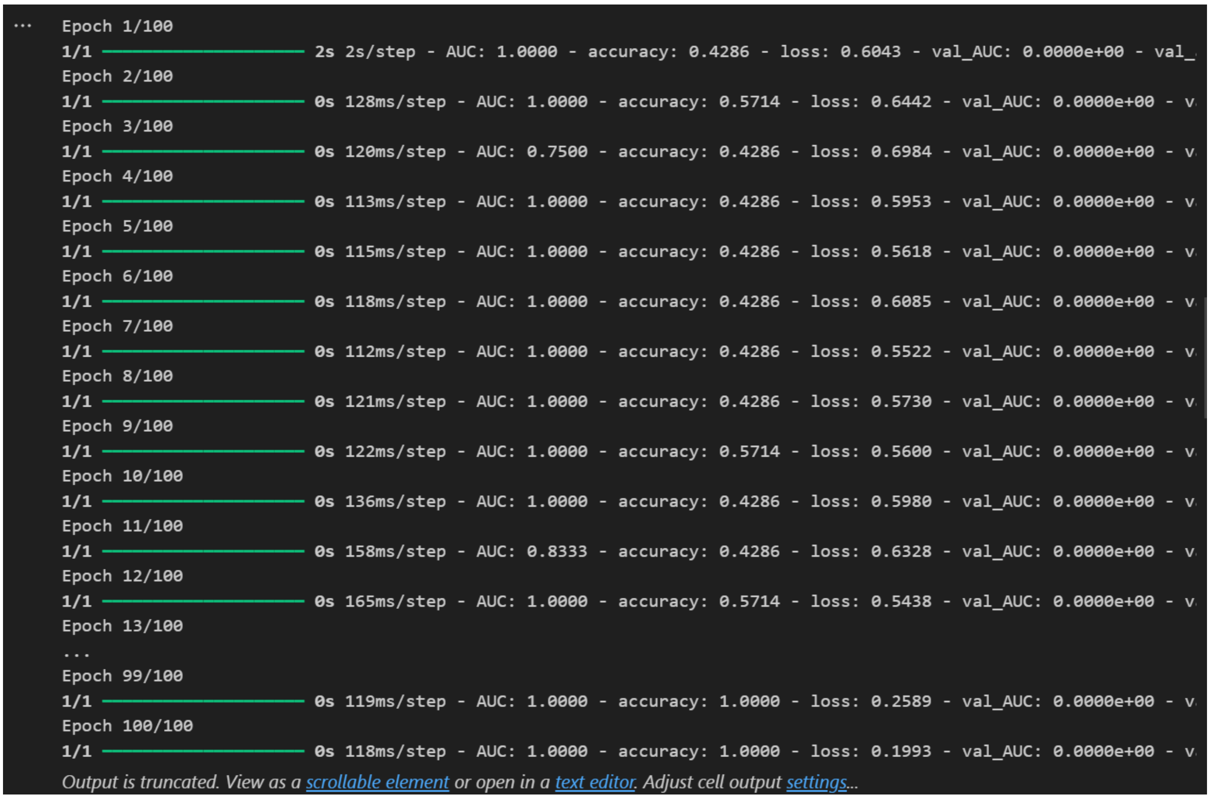
Langkah 3 — Training dengan Early Stopping

1. Lanjut copy paste lagi dari lembar kerja ke vscode



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

history = model.fit(
    X_train, y_train,
    validation_data=(X_val, y_val),
    epochs=100, batch_size=32,
    callbacks=[es], verbose=1
)
```

2. Hasilnya setelah copy paste



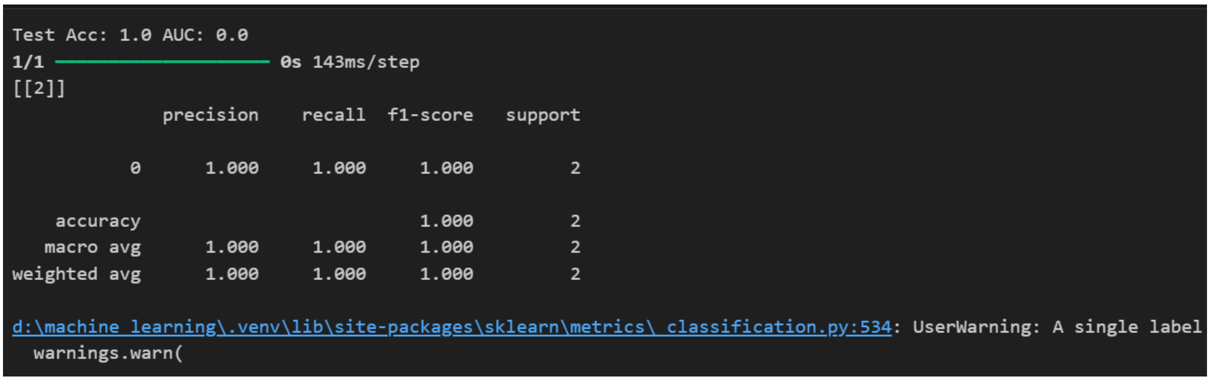
```
...  Epoch 1/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.6043 - val_AUC: 0.0000e+00 - val_
     Epoch 2/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 128ms/step - AUC: 1.0000 - accuracy: 0.5714 - loss: 0.6442 - val_AUC: 0.0000e+00 - v
     Epoch 3/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 120ms/step - AUC: 0.7500 - accuracy: 0.4286 - loss: 0.6984 - val_AUC: 0.0000e+00 - v
     Epoch 4/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 113ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.5953 - val_AUC: 0.0000e+00 - v
     Epoch 5/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 115ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.5618 - val_AUC: 0.0000e+00 - v
     Epoch 6/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 118ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.6085 - val_AUC: 0.0000e+00 - v
     Epoch 7/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 112ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.5522 - val_AUC: 0.0000e+00 - v
     Epoch 8/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 121ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.5730 - val_AUC: 0.0000e+00 - v
     Epoch 9/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 122ms/step - AUC: 1.0000 - accuracy: 0.5714 - loss: 0.5600 - val_AUC: 0.0000e+00 - v
     Epoch 10/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 136ms/step - AUC: 1.0000 - accuracy: 0.4286 - loss: 0.5980 - val_AUC: 0.0000e+00 - v
     Epoch 11/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 158ms/step - AUC: 0.8333 - accuracy: 0.4286 - loss: 0.6328 - val_AUC: 0.0000e+00 - v
     Epoch 12/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 165ms/step - AUC: 1.0000 - accuracy: 0.5714 - loss: 0.5438 - val_AUC: 0.0000e+00 - v
     Epoch 13/100
     ...
     Epoch 99/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 119ms/step - AUC: 1.0000 - accuracy: 1.0000 - loss: 0.2589 - val_AUC: 0.0000e+00 - v
     Epoch 100/100
     1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 118ms/step - AUC: 1.0000 - accuracy: 1.0000 - loss: 0.1993 - val_AUC: 0.0000e+00 - v
     Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

Langkah 4 — Evaluasi di Test Set

1. Lanjut copy paste dari lembar kerja ke vscode

```
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))
```

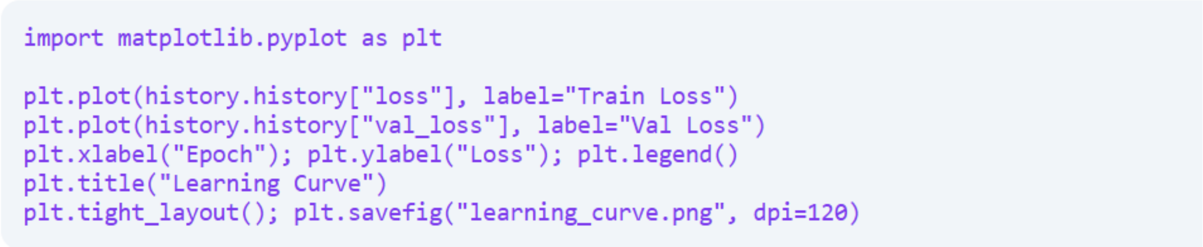
2. Hasilnya seperti ini



```
Test Acc: 1.0 AUC: 0.0
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 143ms/step
[[2]]
              precision    recall  f1-score   support

           0      1.000     1.000     1.000         2

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

d:\machine learning\.venv\lib\site-packages\sklearn\metrics\_classification.py:534: UserWarning: A single label
  warnings.warn(
```

Langkah 5 — Visualisasi Learning Curve

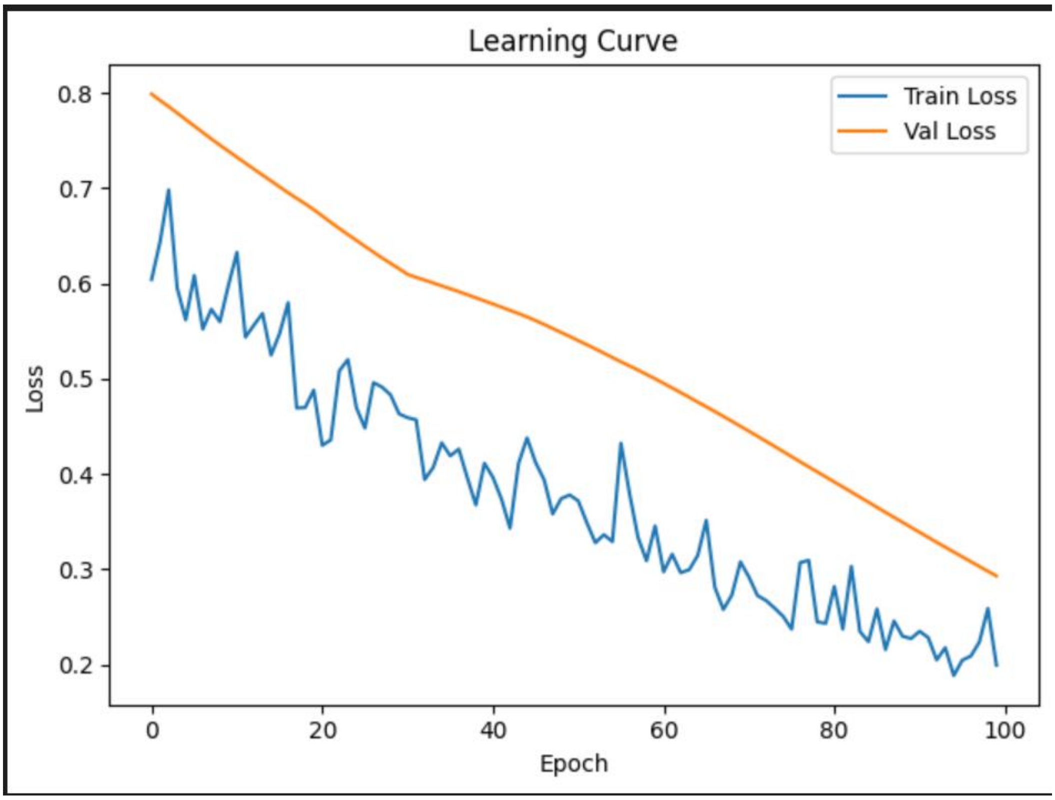
1. Lanjut copy paste dari lembar kerja ke vscode



```
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)
```

Kurva membantu mendeteksi overfitting/underfitting.

2. Hasilnya seperti ini


Learning Curve
Train Loss
Val Loss
Loss
Epoch