Nama : Khalishah

NIM : 1103213045

Heart Disease Week 11

```
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import DataLoader, TensorDataset
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
```

Pada tahap ini, berbagai pustaka diimpor untuk mendukung pembuatan dan evaluasi model machine learning. torch digunakan untuk membangun model deep learning, sklearn membantu dalam manipulasi data dan evaluasi, sedangkan pandas dan numpy digunakan untuk pengolahan data. Kombinasi pustaka ini memungkinkan pengembangan pipeline machine learning yang fleksibel dan kuat.

```
# Pemisahan data menjadi fitur (X) dan target (y)
X = df.drop('target', axis=1).values
y = df['target'].values
```

Data dipisahkan menjadi fitur (X) dan target (y). Fitur adalah variabel independen yang akan digunakan model untuk memprediksi target, sedangkan target adalah variabel dependen yang ingin kita prediksi. Dalam kode ini, kolom 'target' digunakan sebagai target, sementara kolom lainnya dijadikan fitur.

```
# Mengonversi data menjadi tensor untuk digunakan dalam PyTorch
X_train_tensor = torch.tensor(X_train, dtype=torch.float32)
y_train_tensor = torch.tensor(y_train, dtype=torch.long)
X_test_tensor = torch.tensor(X_test, dtype=torch.float32)
y_test_tensor = torch.tensor(y_test, dtype=torch.long)
```

Data pelatihan dan pengujian dikonversi menjadi tensor PyTorch. Proses ini penting karena PyTorch menggunakan tensor sebagai struktur data utama untuk operasinya. Jenis data (dtype) disesuaikan: fitur menggunakan float32, sedangkan target menggunakan long, agar sesuai dengan kebutuhan model neural network.

```python
# Menampilkan best dan worst hyperparameter berdasarkan akurasi
print("\nBest Hyperparameter Configuration:")
print(f"Hidden Layers: {best_result['Hyperparameters']['Hidden Layers']}")
print(f"Hidden Neurons: {best_result['Hyperparameters']['Hidden Neurons']}")
print(f"Activation Function: {best_result['Hyperparameters']['Activation Function']}")
print(f"Epochs: {best_result['Hyperparameters']['Epochs']}")
print(f"Learning Rate: {best_result['Hyperparameters']['Learning Rate']}")
print(f"Batch Size: {best_result['Hyperparameters']['Batch Size']}")
print(f"Test Accuracy: {best_result['Test Accuracy'] * 100:.2f}%")
```

```
Best Hyperparameter Configuration:
Hidden Layers: 2
Hidden Neurons: 16
Activation Function: Sigmoid
Epochs: 100
Learning Rate: 0.1
Batch Size: 128
Test Accuracy: 100.00%
```

Hasil terbaik dicapai dengan konfigurasi 2 hidden layers, 16 neuron, fungsi aktivasi Sigmoid, learning rate 0.1, batch size 128, dan 100 epoch. Dengan konfigurasi ini, model berhasil mencapai akurasi pengujian sebesar 100%, yang menunjukkan performa optimal untuk dataset ini.

```python
# Menampilkan hyperparameter terburuk berdasarkan akurasi terendah
print("\nWorst Hyperparameter Configuration:")
print(f"Hidden Layers: {worst_result['Hyperparameters']['Hidden Layers']}")
print(f"Hidden Neurons: {worst_result['Hyperparameters']['Hidden Neurons']}")
print(f"Activation Function: {worst_result['Hyperparameters']['Activation Function']}")
print(f"Epochs: {worst_result['Hyperparameters']['Epochs']}")
print(f"Learning Rate: {worst_result['Hyperparameters']['Learning Rate']}")
print(f"Batch Size: {worst_result['Hyperparameters']['Batch Size']}")
print(f"Test Accuracy: {worst_result['Test Accuracy'] * 100:.2f}%")
```

```
Worst Hyperparameter Configuration:
Hidden Layers: 1
Hidden Neurons: 8
Activation Function: linear
Epochs: 25
Learning Rate: 0.0001
Batch Size: 512
Test Accuracy: 22.08%
```

Sebaliknya, hasil terburuk dicapai dengan konfigurasi 1 hidden layer, 8 neuron, fungsi aktivasi Linear, learning rate 0.0001, batch size 512, dan 25 epoch. Akurasi pengujian hanya 22.08%, yang menunjukkan bahwa kombinasi ini tidak efektif dalam menangkap pola dari data.