




Nama Dosen : Teguh Iman Hermanto, M.Kom  
Mata Kuliah : Machine Learning 2  
Pembahasan : ANN Regresi  
Pokok Pemb : Membuat model regresi sederhana menggunakan ANN



```
1 import numpy as np
2 import tensorflow as tf
3 from tensorflow.keras.models import Sequential
4 from tensorflow.keras.layers import Dense
5 from tensorflow.keras.utils import plot_model
6 from sklearn.model_selection import train_test_split
7 from sklearn.preprocessing import StandardScaler
8 from sklearn.metrics import r2_score # Untuk menghitung akurasi
```



```
1 # Dataset
2 kalori = np.array([2000, 2200, 2400, 2600, 2800,
3                    3000, 3200, 3400, 3600, 3800], dtype=float)
4
5 langkah = np.array([10000, 11000, 12000, 13000, 14000,
6                    15000, 16000, 17000, 18000, 19000], dtype=float)
```



```
1 # Normalisasi data
2 scaler_kalori = StandardScaler()
3 scaler_langkah = StandardScaler()
4 kalori = scaler_kalori.fit_transform(kalori.reshape(-1, 1))
5 langkah = scaler_langkah.fit_transform(langkah.reshape(-1, 1))
```

[illegible]



```
1 # Membangun model ANN
2 model = Sequential()
3 model.add(Dense(10, input_shape=(1,), activation='relu'))
4 model.add(Dense(10, activation='relu'))
5 model.add(Dense(1)) # Output layer
```



```
1 # Compile model
2 model.compile(optimizer='adam', loss='mean_squared_error')
```



```
1 # Plot arsitektur model
2 plot_model(model, show_shapes=True, show_layer_names=True)
```



```
1 # Training model
2 history = model.fit(X_train, y_train, epochs=100, batch_size=2,
3                     verbose=1, validation_split=0.2)
```



```
1 # Evaluasi model
2 y_pred = model.predict(X_test)
3 y_pred = scaler_langkah.inverse_transform(y_pred) # Kembalikan ke skala asli
4 y_test = scaler_langkah.inverse_transform(y_test) # Kembalikan ke skala asli
```



```
1 # Hitung akurasi menggunakan R-squared
2 r2 = r2_score(y_test, y_pred)
3 print(f'Akurasi (R-squared): {r2 * 100:.2f}%')
```



```
1 # Prediksi jumlah langkah untuk kalori tertentu
2 kalori_input = np.array([3000], dtype=float)
3 kalori_scale = scaler_kalori.transform(kalori_input.reshape(-1, 1))
4 prediksi_langkah = model.predict(kalori_scale)
5 prediksi_langkah = scaler_langkah.inverse_transform(prediksi_langkah) # Kembalikan ke skala asli
6 print(f'Prediksi jumlah langkah untuk {kalori_input} kalori adalah {prediksi_langkah} langkah')
```