LAMPIRAN

LAMPIRAN 1 Kode Program ANN dengan Standar *Duval Triangle*

```
import pandas as pd
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler,
LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,
classification_report, mean_squared_error

# Baca data dari file excel
data = pd.read_excel('Data_Skripsi.xlsx')

# Pisahkan data (parameter) dan target (fault)
X = data[['%CH4', '%C2H2', '%C2H4']]  # parameter ch4,
c2h2, dan c2h4
y = data['Fault']   # fault

# Normalisasi fitur gas
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Pisahkan data menjadi set pelatihan dan pengujian
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y, test_size=0.2,
random_state=42)

# Model ANN
model = MLPClassifier(hidden_layer_sizes=(100, 50),
activation='relu', solver='adam',
```

```
learning_rate_init=0.01, max_iter=1000,
random_state=42)

# Latih model dengan data latih
model.fit(X_train, y_train)

# Lakukan prediksi menggunakan data uji
y_pred_encoded = model.predict(X_test)

# Mapping dari angka ke nama label fault
label_mapping = {
    1: 'PD',
    2: 'D1',
    3: 'D2',
    4: 'T1',
    5: 'T2',
    6: 'T3',
    7: 'DT',
}

# Konversi prediksi dan y_test dari angka ke nama label
y_pred = [label_mapping[i] for i in y_pred_encoded]
y_test_labels = [label_mapping[i] for i in y_test]

# Menampilkan parameter model ANN
print("Parameter Model ANN:")
print("Jumlah Lapisan Tersembunyi:",
model.hidden_layer_sizes)
print("Fungsi Aktivasi:", model.activation)
print("Solver:", model.solver)
print("Jumlah Maksimum Iterasi:", model.max_iter)
print("Laju Pembelajaran:", model.learning_rate_init)
print("Toleransi:", model.tol)
```

```
# Evaluasi performa model
mse = mean_squared_error(y_test, y_pred_encoded)
print(f'Mean Squared Error (MSE): {mse:}')
mape = np.mean(np.abs((y_test - y_pred_encoded) /
y_test)) * 100
print(f'Mean Absolute Percentage Error (MAPE):
{mape:}%')
accuracy = accuracy_score(y_test, y_pred_encoded)
print(f'Akurasi: {accuracy * 100:.2f}%')
labels = ['PD', 'D1', 'D2', 'T1', 'T2', 'T3', 'DT']
print("Classification Report:")
print(classification_report(y_test_labels, y_pred,
labels=labels))

# Prediksi data dengan parameter tertentu
def predict_fault_duval_triangle(CH4, C2H2, C2H4):
    parameters = np.array([[CH4, C2H2, C2H4]])
    scaled_parameters = scaler.transform(parameters)
    pred_encoded= model.predict(scaled_parameters)[0]
    return label_mapping[pred_encoded]

# Memasukkan nilai untuk setiap variabel
CH4 = float(input("Masukkan Nilai CH4(%): "))
C2H2 = float(input("Masukkan Nilai C2H2(%): "))
C2H4 = float(input("Masukkan Nilai C2H4(%): "))

duval_triangle_prediksi =
predict_fault_duval_triangle(CH4, C2H2, C2H4)
print("\nPrediksi Fault:", duval_triangle_prediksi)
```

Hasil program dari ANN dengan standar *Duval Triangle*:

```
Parameter Model ANN:
Jumlah Lapisan Tersembunyi: (100, 50)
Fungsi Aktivasi: relu
Solver: adam
Jumlah Maksimum Iterasi: 1000
Laju Pembelajaran: 0.01
Toleransi: 0.0001
Mean Squared Error (MSE): 0.0625
Mean Absolute Percentage Error (MAPE): 1.5625%
Akurasi: 93.75%
Classification Report:
              precision    recall  f1-score   support

          PD       1.00      1.00      1.00         2
          D1       1.00      1.00      1.00         2
          D2       1.00      1.00      1.00         1
          T1       1.00      0.86      0.92         7
          T2       0.67      1.00      0.80         2
          T3       1.00      1.00      1.00         1
          DT       1.00      1.00      1.00         1

    accuracy                           0.94        16
   macro avg       0.95      0.98      0.96        16
weighted avg       0.96      0.94      0.94        16

Masukkan Nilai CH4(%): 83.87
Masukkan Nilai C2H2(%): 0
Masukkan Nilai C2H4(%): 16.13
Prediksi Fault: T1
```

LAMPIRAN 2 Kode Program ANN dengan Standar *Duval Pentagon*

```
import pandas as pd
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler,
LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,
classification_report, mean_squared_error

# Baca data dari file excel
data = pd.read_excel('Data_Skripsi_DPM.xlsx')
# Pisahkan data (parameter) dan target (fault)
X = data[['CX', 'CY']]  # Parameter Cx dan Cy
y = data['Fault']   # Fault

# Normalisasi fitur gas
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Pisahkan data menjadi set pelatihan dan pengujian
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y, test_size=0.2,
random_state=42)

# Model ANN
model = MLPClassifier(hidden_layer_sizes=(100, 50),
activation='relu', solver='adam',
learning_rate_init=0.01, max_iter=1500,
random_state=42)

# Latih model dengan data latih
```

```
model.fit(X_train, y_train)

# Lakukan prediksi menggunakan data uji
y_pred_encoded = model.predict(X_test)

# Mapping dari angka ke nama label fault
label_mapping = {
    1: 'PD',
    2: 'D1',
    3: 'D2',
    4: 'T1',
    5: 'T2',
    6: 'T3',
    7: 'S',
}

# Konversi prediksi dan y_test dari angka ke nama label
y_pred = [label_mapping[i] for i in y_pred_encoded]
y_test_labels = [label_mapping[i] for i in y_test]

# Menampilkan parameter model ANN
print("Parameter Model ANN:")
print("Jumlah Lapisan Tersembunyi:",
model.hidden_layer_sizes)
print("Fungsi Aktivasi:", model.activation)
print("Solver:", model.solver)
print("Jumlah Maksimum Iterasi:", model.max_iter)
print("Laju Pembelajaran:", model.learning_rate_init)
print("Toleransi:", model.tol)

# Evaluasi performa model
mse = mean_squared_error(y_test, y_pred_encoded)
print(f'Mean Squared Error (MSE): {mse:}')
```

```
mape = np.mean(np.abs((y_test - y_pred_encoded) /
y_test)) * 100
print(f'Mean Absolute Percentage Error (MAPE):
{mape:}%')
accuracy = accuracy_score(y_test, y_pred_encoded)
print(f'Akurasi: {accuracy * 100:.2f}%')
labels = ['PD', 'D1', 'D2', 'T1', 'T2', 'T3', 'S']
print("Classification Report:")
print(classification_report(y_test_labels, y_pred,
labels=labels))

# Prediksi data dengan parameter tertentu
def predict_fault_duval_pentagon(Cx, Cy):
    parameters = np.array([[Cx, Cy]])
    scaled_parameters = scaler.transform(parameters)
    pred_encoded =
model.predict(scaled_parameters)[0]
    return label_mapping[pred_encoded]

# Memasukkan nilai untuk setiap variabel
Cx = float(input("Masukkan Nilai Cx: "))
Cy = float(input("Masukkan Nilai Cy: "))

duval_pentagon_prediksi =
predict_fault_duval_pentagon(Cx, Cy)
print("Prediksi Fault:", duval_pentagon_prediksi)
```

Hasil dari program ANN dengan standar *Duval Pentagon*:

```
Parameter Model ANN:
Jumlah Lapisan Tersembunyi: (100, 50)
Fungsi Aktivasi: relu
Solver: adam
Jumlah Maksimum Iterasi: 1000
Laju Pembelajaran: 0.01
Toleransi: 0.0001
Mean Squared Error (MSE): 0.05555555555555555
Mean Absolute Percentage Error (MAPE): 1.1111111111111112%
Akurasi: 94.44%
Classification Report:
              precision    recall  f1-score   support

          PD       1.00      1.00      1.00         2
          D1       1.00      1.00      1.00         2
          D2       1.00      1.00      1.00         1
          T1       0.83      1.00      0.91         5
          T2       1.00      0.50      0.67         2
          T3       1.00      1.00      1.00         3
           S       1.00      1.00      1.00         3

    accuracy                           0.94        18
   macro avg       0.98      0.93      0.94        18
weighted avg       0.95      0.94      0.94        18

Masukkan Nilai Cx: -25.268
Masukkan Nilai Cy: -2.71
Prediksi Fault: T1
```