Lampiran 1. *Source Code SVM*

```
from sklearn.model_selection import KFold
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay, roc_curve, auc
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
from sklearn.metrics import roc_curve, auc, accuracy_score

# Model initialization
from sklearn.calibration import CalibratedClassifierCV
from sklearn.svm import SVC
clf_svm = SVC(kernel='kernel name')
model_svm = CalibratedClassifierCV(clf_svm)


# Making object
scaler = MinMaxScaler()
accuracies = []
confusion_matrices = []
probabilities = []
labels = []
trained_models = []


#K-FOLD
kf = KFold(n_splits=number of fold, shuffle = True,
random_state = 40)

for i, (train_index, validation_index) in
enumerate(kf.split(X, y), 1):
    X_train, X_val = X_numpy[train_index],
X_numpy[validation_index]
    y_train, y_val = y_numpy[train_index],
y_numpy[validation_index]

    # Scaling dataset
    X_train = scaler.fit_transform(X_train)
    X_val = scaler.transform(X_val)

    # Training
    model_svm.fit(X_train, y_train)
```

```
# Predict probabilities for the test data
y_proba = model_svm.predict_proba(X_val)[::,1]
y_pred = model_svm.predict(X_val)

# Append the trained model to the list
trained_models.append(model_svm)

probabilities.append(y_proba)
labels.append(y_val)

# Compute the confusion matrix for the current fold
cm = confusion_matrix(y_val, y_pred)
confusion_matrices.append(cm)

# Calculate and print accuracy for the current fold
accuracy = metrics.accuracy_score(y_val, y_pred)
accuracies.append(accuracy)

print(f"Fold {i}:")
print("Accuracy:", accuracy)
```

Lampiran 2. *Source Code Backpropagation Menggunakan Scikit-Learn*

```
from sklearn.model_selection import KFold
from sklearn.metrics import confusion_matrix,
ConfusionMatrixDisplay, roc_curve, auc
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import roc_curve, auc, accuracy_score

from sklearn.neural_network import MLPClassifier
clf = MLPClassifier(solver='lbfgs', alpha=1e-
4,hidden_layer_sizes=(25, 10), random_state = None,
activation='activation function's name')

# Making object
scaler = MinMaxScaler()
accuracies = []
confusion_matrices = []
probabilities = []
labels = []
trained_models = []

#K-FOLD
kf = KFold(n_splits=number of fold, shuffle = True,
random_state = 40)

for i, (train_index, validation_index) in
enumerate(kf.split(X, y), 1):
    X_train, X_val = X_numpy[train_index],
X_numpy[validation_index]
    y_train, y_val = y_numpy[train_index],
y_numpy[validation_index]

    # Scaling dataset
    X_train = scaler.fit_transform(X_train)
    X_val = scaler.transform(X_val)

    # Training
    clf.fit(X_train, y_train)


    # Predict probabilities for the test data
    y_proba = clf.predict_proba(X_val)[::,1]
    y_pred = clf.predict(X_val)
```

```
    # Append the trained model to the list
    trained_models.append(clf)

    probabilities.append(y_proba)
    labels.append(y_test)

    # Compute the confusion matrix for the current fold
    cm = confusion_matrix(y_test, y_pred)
    confusion_matrices.append(cm)

    # Calculate and print accuracy
    accuracy = metrics.accuracy_score(y_test, y_pred)

    print(f"Fold {i}:")
    print("Accuracy:", accuracy)
    print()

    accuracy = accuracy_score(y_test, y_pred)
    accuracies.append(accuracy)
```

Lampiran 3. *Source Code Backpropagation Menggunakan Tensorflow*

```
import tensorflow as tf

# Create a model with custom activation functions
input_dim = 19
def create_model():
  model = tf.keras.Sequential([
      tf.keras.layers.Dense(25, activation='tanh',
input_shape=(input_dim,)),
      tf.keras.layers.Dense(10, activation='tanh',
input_shape=(input_dim,)),
      tf.keras.layers.Dense(1, activation='tanh')
  ])

from sklearn.model_selection import KFold,cross_val_score
from sklearn.metrics import
precision_recall_fscore_support, confusion_matrix,
ConfusionMatrixDisplay, roc_curve, auc, accuracy_score
from sklearn.preprocessing import MinMaxScaler, scale
from sklearn import metrics

#making an object for scaller
scaler = MinMaxScaler()

#number of fold
kf = KFold(n_splits= number of fold,shuffle=True,
random_state=40)

# Making object
compiled_result = []
confusion_matrices = []
probabilities = []
labels = []
accuracies = []
trained_models = []

for i, (train_index, validation_index) in
enumerate(kf.split(X, y), 1):
    X_train, X_val = X_numpy[train_index],
X_numpy[validation_index]
    y_train, y_val = y_numpy[train_index],
y_numpy[validation_index]

    # Scaling dataset
    X_train = scaler.fit_transform(X_train)
    X_val = scaler.transform(X_val)
```

```
  # Create the model
  model = create_model()

  # Train the model with backpropagation
  model.fit(X_train, y_train, epochs=10)

  #testing process
  y_pred = model.predict(X_val)


   # Append the trained model to the list
  trained_models.append(model)

  import numpy as np

  # Set a threshold value
  threshold = 0.0
  # Convert continuous values to binary labels
  y_pred_binary = np.where(y_pred >= threshold, 1,-1)

  # Predict probabilities for the test data
  y_proba = y_pred_binary.flatten()

  probabilities.append(y_proba)
  labels.append(y_test)

  # Compute the confusion matrix for the current fold
  cm = confusion_matrix(y_test, y_pred_binary)
  confusion_matrices.append(cm)


  # Calculate and print accuracy, precision, recall, and
F1 score for the current fold
  accuracy = metrics.accuracy_score(y_test, y_pred_binary)


  print(f"Fold {i}:")
  print("Accuracy:", accuracy)
  print()


  accuracy = accuracy_score(y_test, y_pred_binary)
  accuracies.append(accuracy)
```

RIWAYAT HIDUP

Dwi Prima Handayani Putri lahir di Singaraja pada tanggal 1 Desember 2001. Penulis lahir dari pasangan suami istri Bapak I Made Sukarma, S.Sos dan Ketut Parni. Penulis berkebangsaan Indonesia. Penulis beralamt di Jl. Sri Rama, No.55 Baktiseraga, Buleleng, Bali. Penulis mengenyam pendidikan dasar di SD 1 Baktiseraga dan lulus pada tahun 2013. Penulis melanjutkan Pendidikan di SMP Negeri 6 Singaraja dan lulus pada tahun 2016. Pada tahun 2019, penulis lulus dari SMA Negeri 1 Singaraja. Saat ini, penulis merupakan seorang mahasiswa aktif di Program Studi S1 Ilmu Komputer, Universitas Pendidikan Ganesha.