inputTable = trainingData;

predictorNames = {'Feature1', 'Feature2', 'Feature3', 'Feature4', 'Feature5', 'Feature6', 'Feature7', 'Feature8', 'Feature9', 'Feature10', 'Feature11', 'Feature12', 'Feature13', 'TARGET'};

predictors = inputTable(:, predictorNames);

response = inputTable.Subjects;

isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false, false, false, false, false];

% Train a classifier

% This code specifies all the classifier options and trains the classifier.

template = templateSVM(...

'KernelFunction', 'gaussian', ...

'PolynomialOrder', [], ...

'KernelScale', 0.9399999999999999, ...

'BoxConstraint', 1, ...

'Standardize', true);

classificationSVM = fitcecoc(...

predictors, ...

response, ...

'Learners', template, ...

'Coding', 'onevsone', ...

'ClassNames', categorical({'ALS'; 'HEALTHY'; 'HUNT'; 'PARKINSON'}));

% Create the result struct with predict function

predictorExtractionFcn = @(t) t(:, predictorNames);

svmPredictFcn = @(x) predict(classificationSVM, x);

trainedClassifier.predictFcn = @(x) svmPredictFcn(predictorExtractionFcn(x));

% Add additional fields to the result struct

trainedClassifier.RequiredVariables = {'Feature1', 'Feature10', 'Feature11', 'Feature12', 'Feature13', 'Feature2', 'Feature3', 'Feature4', 'Feature5', 'Feature6', 'Feature7', 'Feature8', 'Feature9', 'TARGET'};

trainedClassifier.ClassificationSVM = classificationSVM;

trainedClassifier.About = 'This struct is a trained model exported from Classification Learner R2022a.';

trainedClassifier.HowToPredict = sprintf('To make predictions on a new table, T, use: \n yfit = c.predictFcn(T) \nreplacing ''c'' with the name of the variable that is this struct, e.g. ''trainedModel''. \n \nThe table, T, must contain the variables returned by: \n c.RequiredVariables \nVariable formats (e.g. matrix/vector, datatype) must match the original training data. \nAdditional variables are ignored. \n \nFor more information, see <a href="matlab:helpview(fullfile(docroot, ''stats'', ''stats.map''), ''appclassification\_exportmodeltoworkspace'')">How to predict using an exported model</a>.');

% Extract predictors and response

% This code processes the data into the right shape for training the

% model.

inputTable = trainingData;

predictorNames = {'Feature1', 'Feature2', 'Feature3', 'Feature4', 'Feature5', 'Feature6', 'Feature7', 'Feature8', 'Feature9', 'Feature10', 'Feature11', 'Feature12', 'Feature13', 'TARGET'};

predictors = inputTable(:, predictorNames);

response = inputTable.Subjects;

isCategoricalPredictor = [false, false, false, false, false, false, false, false, false, false, false, false, false, false];

% Perform cross-validation

partitionedModel = crossval(trainedClassifier.ClassificationSVM, 'KFold', 5);

% Compute validation predictions

[validationPredictions, validationScores] = kfoldPredict(partitionedModel);

% Compute validation accuracy

validationAccuracy = 1 - kfoldLoss(partitionedModel, 'LossFun', 'ClassifError');