```
function [accuracy_inf accuracy_size num_predictions_inf num_predictions_size] = MASSTempFunction()
%==========================================================================
%==========================================================================
%BLACK TREE VECTORIZED DEEP LEARNING
%==========================================================================
%==========================================================================
%COPYRIGHT CHARLES DAVI

%==========================================================================
%RUNS MASSIVE SUPERVISED MODAL CLASSIFICATION
%==========================================================================

  clear all
  num_iterations = 100;

%==========================================================================
%LOADS DATASET
%==========================================================================

  dataset_file = "/Users/charlesdavi/Desktop/Datasets/UCI/credit.txt";

  dataset = csvread(dataset_file);
  num_rows = size(dataset,1);
  N = size(dataset,2) - 1;

  %normalizes dataset
  [weight_vector] = mean_normalization(dataset,N);
  dataset = dataset.*weight_vector;

  %sets the training percentage
  testing_percentage = .15;
  num_testing_rows = floor(testing_percentage*num_rows)

  %finds unique class labels
  class_vector = dataset(:,N+1);
  class_vector = unique(class_vector);
  num_classes = size(class_vector,1);

%==========================================================================
%RUNS ALGORITHM
%==========================================================================
  tic;
  %repeatedly applies the algorithm on random testing subsets
  for i = 1 : num_iterations
    i
    %runs algorithm
    [predicted_class_matrix(:,i) delta_vector prediction_vector_array modal_probability_matrix(:,i) sorted_dataset
cluster_size(:,i)] = MASS_Sup_Modal_BlackTree(dataset, num_testing_rows);

    %calculates knowledge, used for confidence
    [knowledge_matrix(:,i)] = MASS_calc_knowledge(prediction_vector_array, predicted_class_matrix(:,i), delta_vector,
num_classes, N);

    %stores testing_classifiers
    testing_rows = find(sorted_dataset(:,N+1) == -1);
    testing_dataset = sorted_dataset(testing_rows,:);
    actual_class_matrix(:,i) = testing_dataset(:,N+2);

  endfor
```

```
%==========================================================================
%CALCULATES CONFIDENCE (INFORMATION-BASED) AND ACCURACY
%==========================================================================

  %normalizes knowledge using the max
  max_knowledge = max(knowledge_matrix);
  norm_knowledge_matrix = knowledge_matrix / max_knowledge;

  %confidence is the minimum of modal probality and knowledge
  conf_matrix = min(modal_probability_matrix,norm_knowledge_matrix);

  %the set of unique confidence values
  interval_set = unique(conf_matrix);
  num_intervals = size(interval_set,1);

  %calculates accuracy as a function of confidence
  for i = 1 : num_intervals

    x = find(conf_matrix >= interval_set(i));
    num_predictions_inf(i) = size(x,1);
    num_errors = sum(predicted_class_matrix(x) != actual_class_matrix(x));

    accuracy_inf(i)  = 1 - num_errors/num_predictions_inf(i);

  endfor

  figure, plot(accuracy_inf)

%==========================================================================
%CALCULATES CONFIDENCE (SIZE-BASED) AND ACCURACY
%==========================================================================

  max_size = max(cluster_size(:));

  %calculates accuracy as a function of cluster size
  for i = 1 : max_size

    x = find(cluster_size >= i);
    num_predictions_size(i) = size(x,1);
    num_errors = sum(predicted_class_matrix(x) != actual_class_matrix(x));

    accuracy_size(i)  = 1 - num_errors/num_predictions_size(i);

  endfor

  figure, plot(accuracy_size)

  toc

endfunction
```