Leave-One-Out-Cross-Validation Code

%%%%%%%%%%%%%%%%%%%%%% FACE FOUR DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function FaceFour\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('FaceFour\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)

TRAIN = load('FaceFour\_TRAIN');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)+1), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% BEEF DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Beef\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('Beef\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)

TRAIN = load('Beef\_TRAIN');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)+1), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% WAFER DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function wafer\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('wafer\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)

TRAIN = load('wafer\_TRAIN');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)+1), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% SHAPES ALL DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function ShapesAll\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('ShapesAll\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)

TRAIN = load('ShapesAll\_TRAIN');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)+1), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Holdout Classification Code

%%%%%%%%%%%%%%%%%%%%%% FACE FOUR DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Beef\_holdout %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('FaceFour\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

TEST = load('FaceFour\_TEST' ); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

correct = 0; % Initialize the number we got correct

for i = 1 : length(TEST\_class\_labels) % Loop over every instance in the test set

classify\_this\_object = TEST(i,:);

this\_objects\_actual\_class = TEST\_class\_labels(i);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(i), ' out of ', int2str(length(TEST\_class\_labels)), ' done']) % Report progress

end;

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((length(TEST\_class\_labels)-correct )/length(TEST\_class\_labels))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% BEEF DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Beef\_holdout %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('Beef\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

TEST = load('Beef\_TEST' ); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

correct = 0; % Initialize the number we got correct

for i = 1 : length(TEST\_class\_labels) % Loop over every instance in the test set

classify\_this\_object = TEST(i,:);

this\_objects\_actual\_class = TEST\_class\_labels(i);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(i), ' out of ', int2str(length(TEST\_class\_labels)), ' done']) % Report progress

end;

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((length(TEST\_class\_labels)-correct )/length(TEST\_class\_labels))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% WAFER DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function wafer\_holdout %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('wafer\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

TEST = load('wafer\_TEST' ); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

correct = 0; % Initialize the number we got correct

for i = 1 : length(TEST\_class\_labels) % Loop over every instance in the test set

classify\_this\_object = TEST(i,:);

this\_objects\_actual\_class = TEST\_class\_labels(i);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(i), ' out of ', int2str(length(TEST\_class\_labels)), ' done']) % Report progress

end;

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((length(TEST\_class\_labels)-correct )/length(TEST\_class\_labels))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

%%%%%%%%%%%%%%%%%%%%%% SHAPES ALL DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function ShapesAll\_holdout %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('ShapesAll\_TRAIN'); % Only these two lines need to be changed to test a different dataset. %

TEST = load('ShapesAll\_TEST' ); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

correct = 0; % Initialize the number we got correct

for i = 1 : length(TEST\_class\_labels) % Loop over every instance in the test set

classify\_this\_object = TEST(i,:);

this\_objects\_actual\_class = TEST\_class\_labels(i);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(i), ' out of ', int2str(length(TEST\_class\_labels)), ' done']) % Report progress

end;

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((length(TEST\_class\_labels)-correct )/length(TEST\_class\_labels))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Table of Data Set Information

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Num Classes | Size of Training Data | Size of Test Data | Num Features (time series length) | Default Rate | Leave-one-out Error-rate | Hold-Out Error-Rate |
| FaceFour | 4 | 24 | 88 | 350 | 0.286 | 0.30435 | 0.21591 |
| Beef | 5 | 30 | 30 | 470 | 0.333 | 0.048276 | 0.33333 |
| Wafer | 2 | 1000 | 6164 | 152 | 0.005 | 0.999 | 0.0045425 |
| ShapesAll | 60 | 600 | 600 | 512 | 0.248 | 0.2621 | 0.24833 |

I chose the datasets FaceFour, Beef, Wafer, and ShapesAll. The first 6 columns of the table were taken from the database, and confirmed by the reports after running the code. The leave-one-out and hold-out error rates were the results of their respective code. We observe that the hold-out error rates are typically closer to the default values than the leave-one-out rates, except for the FaceFour dataset. We see that the Wafer database has an unreliably high leave-one-out error rate. This may be due to the large size of training data in the set. Conversely, the Beef database has a very small leave-one-out error rate, making it the most reliable estimate.

Leaf Data Set

Leaf Data Leave-One-Out Code

* Note removal of column 2 of data since it is a field that does not affect classification of the leaves.
* The result for error rate is 0.4078. Normalizing the data may help lower this error rate.

%%%%%%%%%%%%%%%%%%%%%% LEAF DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Leaf\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('leaf.csv'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('leaf.csv');

TRAIN(:,2) = []; % remove column 2 (specimen #) as it does not help with classification

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Leaf Data Leave-One-Out Code with Zero-One Normalization

* Note adding of zero-one normalization by subtracting minimum value from each column and dividing each column by its maximum value.
* The result for error rate is 0.93215. This is even higher. Perhaps another method of normalization is better.

%%%%%%%%%%%%%%%%%%%%%% LEAF DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Leaf\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('leaf.csv'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('leaf.csv');

TRAIN(:,2) = []; % remove column 2 (specimen #) as it does not help with classification

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

for i = 1 : size(TRAIN,2) % zero-one normalization

for k = 1 : size(TRAIN,1)

TRAIN(k,i) = TRAIN(k,i)-min(TRAIN(:,i)); % subtract each column by the smallest value in the column

TRAIN(k,i) = TRAIN(k,i)/max(TRAIN(:,i)); % divide each column by the largest value in the column

end

end

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hist(TRAIN(:,3));

hist(TRAIN(:,5));

hist(TRAIN(:,6));

hist(TRAIN(:,8));

hist(TRAIN(:,10));

hist(TRAIN(:,11));

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Leaf Data Leave-One-Out Code with Z Normalization

* Note adding of zero-one normalization by subtracting mean value from each column and dividing each column by its standard deviation.
* The result for error rate is 0.9115. This is better than the zero-one normalization, but still worse than without normalization.

%%%%%%%%%%%%%%%%%%%%%% LEAF DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function Leaf\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('leaf.csv'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('leaf.csv');

TRAIN(:,2) = []; % remove column 2 (specimen #) as it does not help with classification

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

TRAIN\_class\_labels = TRAIN(:,1); % Pull out the class labels.

TRAIN(:,1) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,1); % Pull out the class labels.

TEST(:,1) = []; % Remove class labels from testing set.

for i = 1 : size(TRAIN,2) % zero-one normalization

for k = 1 : size(TRAIN,1)

TRAIN(k,i) = (TRAIN(k,i)-mean(TRAIN(:,i)))/std(TRAIN(:,i)); % subtract each column by the column mean and divide by the column standard deviation

end

end

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Leaf Leave-One-Out Error-Rate with and without normalization

Without normalization, we reported an error-rate of 0.4078, which is not terrible, but we would like to improve on it. Generally, normalization is a great way to do this. However, it appears the two efforts at normalization that we tried were unsuccessful. The zero-one normalization produced an error-rate of 0.93215, which is a drastic downgrade from the standard data set. The Z normalization error-rate was slightly better at 0.9115, but still a disappointment. It appears the data set in its standard format is our best shot at attempting to classify these leaves between the algorithms that we tested.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Num Classes | Size of Training Data | Num Features (time series length) | Leave-One-Out Error-Rate |
| Standard data set | 36 | 340 | 14 | 0.4078 |
| Zero-One Normalization | 36 | 340 | 14 | 0.93215 |
| Z Normzalization | 36 | 340 | 14 | 0.9115 |

Banknote Authentication Data Set

Banknote Authentication Leave-One-Out Code

* The banknote authentication data set is simple to use and required minimal adjustment. The classes are in column 5 in this data set, which we needed to change our code to reflect. The rest of the fields are all fields of continuous numeric data which can contribute to classification. It remains to be seen if normalization will help improve the classification
* The error-rate is 0.00072939. That is extremely accurate.

%%%%%%%%%%%%%%%%%%%%%% BANKNOTE AUTHENTICATION DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function banknote\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('data\_banknote\_authentication.txt'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('data\_banknote\_authentication.txt');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

% note that class labels are in column 5 for this data set

TRAIN\_class\_labels = TRAIN(:,5); % Pull out the class labels.

TRAIN(:,5) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,5); % Pull out the class labels.

TEST(:,5) = []; % Remove class labels from testing set.

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Banknote Authentication Leave-One-Out Code with Zero-One Normalization

* Note adding of zero-one normalization by subtracting minimum value from each column and dividing each column by its maximum value.
* The result for error rate is 0.50109. This is higher.

%%%%%%%%%%%%%%%%%%%%%% BANKNOTE AUTHENTICATION DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function banknote\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('data\_banknote\_authentication.txt'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('data\_banknote\_authentication.txt');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

% note that class labels are in column 5 for this data set

TRAIN\_class\_labels = TRAIN(:,5); % Pull out the class labels.

TRAIN(:,5) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,5); % Pull out the class labels.

TEST(:,5) = []; % Remove class labels from testing set.

for i = 1 : size(TRAIN,2) % zero-one normalization

for k = 1 : size(TRAIN,1)

TRAIN(k,i) = TRAIN(k,i)-min(TRAIN(:,i)); % subtract each column by the smallest value in the column

TRAIN(k,i) = TRAIN(k,i)/max(TRAIN(:,i)); % divide each column by the largest value in the column

end

end

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

hist(TRAIN(:,3));

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Banknote Authentication Leave-One-Out Code with Z Normalization

* Note adding of zero-one normalization by subtracting mean value from each column and dividing each column by its standard deviation.
* The result for error rate is 0.19183. This is better than the zero-one normalization, but still worse than without normalization.

%%%%%%%%%%%%%%%%%%%%%% BANKNOTE AUTHENTICATION DATA SET %%%%%%%%%%%%%%%%%%%%%%%%%%%

function banknote\_LOOCV %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% (C) Eamonn Keogh %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

TRAIN = load('data\_banknote\_authentication.txt'); % Only these two lines need to be changed to test a different dataset. %

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

correct = 0; % Initialize the number we got correct

for j = 1 : size(TRAIN,1)-1

TRAIN = load('data\_banknote\_authentication.txt');

TEST = TRAIN(j,:); % identify the test example

TRAIN(j,:) = []; % remove test example from training set

% note that class labels are in column 5 for this data set

TRAIN\_class\_labels = TRAIN(:,5); % Pull out the class labels.

TRAIN(:,5) = []; % Remove class labels from training set.

TEST\_class\_labels = TEST(:,5); % Pull out the class labels.

TEST(:,5) = []; % Remove class labels from testing set.

for i = 1 : size(TRAIN,2) % zero-one normalization

for k = 1 : size(TRAIN,1)

TRAIN(k,i) = (TRAIN(k,i)-mean(TRAIN(:,i)))/std(TRAIN(:,i)); % subtract each column by the column mean and divide by the column standard deviation

end

end

classify\_this\_object = TEST(1,:);

this\_objects\_actual\_class = TEST\_class\_labels(1);

predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels, classify\_this\_object);

if predicted\_class == this\_objects\_actual\_class

correct = correct + 1; % we got one more correct

end;

disp([int2str(j), ' out of ', int2str(size(TRAIN,1)), ' done']) % Report progress

end

%%%%%%%%%%%%%%%%% Create Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

disp(['The dataset you tested has ', int2str(length(unique(TRAIN\_class\_labels))), ' classes'])

disp(['The training set is of size ', int2str(size(TRAIN,1)),', and the test set is of size ',int2str(size(TEST,1)),'.'])

disp(['The time series are of length ', int2str(size(TRAIN,2))])

disp(['The error rate was ',num2str((size(TRAIN,1)-correct )/size(TRAIN,1))])

%%%%%%%%%%%%%%%%% End Report %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function predicted\_class = Classification\_Algorithm(TRAIN,TRAIN\_class\_labels,unknown\_object)

best\_so\_far = inf;

for i = 1 : length(TRAIN\_class\_labels)

compare\_to\_this\_object = TRAIN(i,:);

distance = sqrt(sum((compare\_to\_this\_object - unknown\_object).^2)); % Euclidean distance

if distance < best\_so\_far

predicted\_class = TRAIN\_class\_labels(i);

best\_so\_far = distance;

end

end;

Banknote Authentication Leave-One-Out Error-Rate with and without normalization

Without normalization, we reported an error-rate of 0.00072939, which is fantastic. With a rate like that, we wouldn’t count on normalization to improve on it. It seems as if the data is distributed in a normal fashion. It appears our two efforts at normalization that we tried were unsuccessful again on this data set, but in this case it is not disappointing given our result for the standard data set. The zero-one normalization produced an error-rate of 0.50109, which is a drastic downgrade from the standard data set. The Z normalization error-rate was better at 0.19183, but we still would not pick that normalized data set over the untouched one.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Num Classes | Size of Training Data | Num Features | Leave-One-Out Error-Rate |
| Standard data set | 2 | 1371 | 4 | 0.00072939 |
| Zero-One Normalization | 2 | 1371 | 4 | 0.50109 |
| Z Normzalization | 2 | 1371 | 4 | 0.19183 |