```
%==========================================================================
%VECTORIZED DEEP LEARNING
%==========================================================================
%==========================================================================
%COPYRIGHT CHARLES DAVI 2020

%UCI DATASETS

%==========================================================================
%LOADS UCI IONOSPHERE DATASET
%==========================================================================

%clears the memory and the command prompt
clear
clc
%--------------------------------------------------------------------

file_path = '/Users/charlesdavi/Desktop/Datasets/UCI/Ionosphere_full_dataset.txt';
dataset = csvread(file_path);
num_rows = size(dataset,1);
N = size(dataset,2) - 1;

training_percentage = .85;

num_training_rows = ceil(num_rows*training_percentage);
training_rows = randperm(num_rows,num_training_rows);
training_dataset = dataset(training_rows,:);

temp = dataset;
temp(training_rows,:) = [];
testing_dataset = temp;
num_testing_rows = size(testing_dataset,1);

%==========================================================================
%GENERATE NON-MUTUALLY EXCLUSIVE CLUSTERS
%==========================================================================

tic;[cluster_matrix final_delta] = fully_vectorized_delta_clustering(training_dataset, N);
toc

%==========================================================================
%GENERATES PREDICTIONS
%==========================================================================

[prediction_matrix] = fully_vectorized_delta_prediction(training_dataset, testing_dataset,
final_delta);
```

```
for i = 1 : num_testing_rows

  class = testing_dataset(i,N+1) %the class of the input vector
  matched_rows = find(prediction_matrix(i,:) == 1); %the training rows that matched with
the input vector
  class_vector = training_dataset(matched_rows,N+1) %their classes

  num_errors(i) = sum(class_vector != class);

endfor

accuracy = 1 - num_errors/num_training_rows

%==========================================================================
%LOADS UCI WINE DATASET
%==========================================================================

%clears the memory and the command prompt
clear
clc
%--------------------------------------------------------------

file_path = '/Users/charlesdavi/Desktop/Datasets/UCI/wine_full_dataset';
dataset = csvread(file_path);
num_rows = size(dataset,1);
N = size(dataset,2) - 1;

%moves the classifer to the last column--------------------------

temp = dataset(:,1);
dataset(:, 1) = dataset(:, N+1);
dataset(:, N+1) = temp;

%==========================================================================
%NORMALIZES DATASET
%==========================================================================

tic;

%normalizes the dataset
[digit_matrix s] = test_data_magnitude(dataset,N);

[final_weight_vector] = express_normalize_dataset(dataset,digit_matrix,N);

temp_matrix = dataset(:,1:N);
temp_matrix = temp_matrix./final_weight_vector;
temp_matrix(:,N+1) = dataset(:,N+1); %adds classifiers
```

```
dataset = temp_matrix;

toc

%==========================================================================
%GENEREATES TESTING / TRAINING DATASETS
%==========================================================================

training_percentage = .8;

num_training_rows = ceil(num_rows*training_percentage);
training_rows = randperm(num_rows,num_training_rows);
training_dataset = dataset(training_rows,:);

temp = dataset;
temp(training_rows,:) = [];
testing_dataset = temp;
num_testing_rows = size(testing_dataset,1);

%==========================================================================
%GENERATE NON-MUTUALLY EXCLUSIVE CLUSTERS
%==========================================================================

tic;[cluster_matrix final_delta] = fully_vectorized_delta_clustering(training_dataset, N);
toc

%==========================================================================
%GENERATES PREDICTIONS
%==========================================================================

[prediction_matrix] = fully_vectorized_delta_prediction(training_dataset, testing_dataset,
final_delta);

num_rejections = 0;

for i = 1 : num_testing_rows

  class = testing_dataset(i,N+1) %the class of the input vector
  matched_rows = find(prediction_matrix(i,:) == 1); %the training rows that matched with
the input vector
  class_vector = training_dataset(matched_rows,N+1) %their classes

  if(isempty(matched_rows))

    num_rejections = num_rejections + 1;

  endif
```

```
  num_errors(i) = sum(class_vector != class);
endfor
accuracy = 1 - num_errors/num_training_rows
```