```
function [data_categories_array category_vec anchor_array H_final delta] =
optimize_categories_N(orig_data_array,N)

  %takes in an array of N-dimensional vectors and returns a categorization of those
vectors.
  %All dimensions above N are ignored.

  [x num_items] = size(orig_data_array);


%-------------------------------------------------------------------------------
-------
  %initializes the data_categories_array to the fully partitioned data_set
  %and calculates the norm of each vector, ignoring the category number

  for i = 1 : num_items

    data_categories_array{i} = orig_data_array{i};
    temp_vec = orig_data_array{i};

    temp_vec = temp_vec(1:N); %this is to ignore dimensions above N;

    norm_vector(i) = norm(temp_vec);

  endfor


%-------------------------------------------------------------------------------
-------
  s = 1.23*std(norm_vector(:));

  %this is the main section of the algorithm
  divisor = 25;
  H_max = log2(num_items);
  min_entropy = .47*H_max + .53*((divisor - 25)/divisor)*H_max;
  max_cnt = 25;
  increment = (s - (s/divisor))/max_cnt;

  cnt = 0;


%-------------------------------------------------------------------------------
------------------------
  %this initializes the return values to the case where the greatest change in
entropy occurs for delta = 0

    max_delta = 0;
    [data_categories_array_1 anchor_array_1] =
```

```
generate_categories_N(orig_data_array,max_delta,N);

    max_delta = s/divisor;
    [data_categories_array_2 anchor_array_2] =
generate_categories_N(orig_data_array,max_delta,N);

    [x num_categories] = size(data_categories_array_1);

    for i = 1 : num_categories

      [x temp] = size(data_categories_array_1{i}); %loads the number of items in
each category
      category_vec_1(i) = temp;

    endfor

    [x num_categories] = size(data_categories_array_2);

    for i = 1 : num_categories

      [x temp] = size(data_categories_array_2{i});  %loads the number of items in
each category
      category_vec_2(i) = temp;

    endfor

    category_vec_1 = category_vec_1/num_items;
    category_vec_2 = category_vec_2/num_items;

    H_1 = vector_entropy(category_vec_1);
    H_2 = vector_entropy(category_vec_2);


    max_ent_change = abs(H_1 - H_2)/increment;
    data_categories_array = data_categories_array_2;
    category_vec = category_vec_2;
    anchor_array = anchor_array_2;
    delta = max_delta;
    H_final = H_2;


%-------------------------------------------------------------------------------
----------

  while(cnt <= max_cnt && (H_2 >= min_entropy))

    category_vec_1 = category_vec_2;
    clear category_vec_2;
```

```
    data_categories_array_1 = data_categories_array_2;
    anchor_array_1 = anchor_array_2;

    H_1 = H_2;

    max_delta = s/divisor + (cnt + 1)*increment;
    [data_categories_array_2 anchor_array_2] =
generate_categories_N(orig_data_array,max_delta,N);

    [x num_categories] = size(data_categories_array_2);

    for i = 1 : num_categories

      [x temp] = size(data_categories_array_2{i});  %loads the number of items in
each category
      category_vec_2(i) = temp;

    endfor

    category_vec_2 = category_vec_2/num_items;

    H_2 = vector_entropy(category_vec_2);

    if(abs(H_1 - H_2)/increment > max_ent_change)

      max_ent_change = abs(H_1 - H_2)/increment;
      data_categories_array = data_categories_array_2;
      category_vec = category_vec_2;
      anchor_array = anchor_array_2;
      delta = max_delta;
      H_final = H_2;

    endif

    cnt = cnt + 1;

  endwhile

endfunction
```