```
function [max_matches final_abstraction final_category final_match_list] = MNIST_generate_abstraction(data_matrix, data_matrix_cat, final_delta, avg_num_points)

N = size(data_matrix,2);
num_scaled_images = size(data_matrix,3);

data_matrix(:,N+1,:) = -1; %this is a dummy classifier

max_matches = 0;

for k = 1 : num_scaled_images

base_object = data_matrix(:,:,k);

for i = 1 : avg_num_points

match_list = [];

current_point = base_object(i,:);

temp = zeros(1,N+1);

for j = 1 : num_scaled_images

current_object = data_matrix(:,:,j);

[predicted_vector predicted_class prediction_row] = find_NN(current_point, current_object,N);

current_object(prediction_row,:) = Inf;

temp = temp + predicted_vector;

endfor

avg = temp/num_scaled_images;
avg_points(i,:) = avg;

endfor

x = avg_points(:,1);
y = avg_points(:,2);

abstraction_object = [x' y'];
abstraction_object(2*avg_num_points + 1) = -1; %this is a dummy classifier


%-------------------------------------------------------------
%TESTS FOR THE ABSTRACTION THAT GENERATES THE MOST MATCHES
num_matches = 0;

for i = 1 : num_scaled_images

temp_object = data_matrix_cat(i,:);

if(norm(temp_object - abstraction_object) <= final_delta)

num_matches = num_matches + 1;
category(num_matches,:) = temp_object; %saves the object that is within delta of the abstraction
match_list(num_matches) = i;

endif

endfor

if(num_matches > max_matches)

max_matches = num_matches
final_abstraction = abstraction_object;
final_category = category;
```

```
69    final_match_list = match_list;
70
71    endif
72
73    endfor
74
75    endfunction
76
```