```
%FIRST PART
% EXPERIMENTAL DATA
load('chal_64_100000_bi.mat');
% 100000 * 64
lol = C;
challenge_set = C;
challenge_set(1:100000,65) = ones(100000,1);
% making bipolar
for i = 1:100000
    for j = 1:64
        if lol(i,j) == 0
            lol(i,j) = 1;
        else
            lol(i,j) = -1;
        end
    end
end
%  calculating parity vectors
for i = 1:100000
    for j = 1:64
        hi = 1;
%         if j == 64
%             challenge_set(i,j) = 1;
%         else
            for k = j:64
                hi = hi * lol(i,k);
            end
            challenge_set(i,j) = hi;
%         end
    end
end
clear C;
clear lol;
% %
load('respGolden_10_APUF_64_100000_Br_5.mat');
response_set = G;
for i = 1:100000
    for j = 1:10
        if response_set(i,j) == 0
            response_set(i,j) = 1;
        else
            response_set(i,j) = -1;
        end
    end
end
clear G;

% setting training and classifying data
training_set_features = challenge_set(1:70000,:);
```

```
training_set_groups = response_set(1:70000,:);

classify_set_features = challenge_set(70001:100000,:);
classify_set_groups = response_set(70001:100000,:);

diary('first_part_exp.txt');
diary on;

% % radial kernel
% % training_set_features from first
S=sprintf('<========================Radial
Kernel=============================>');
disp(S);
for n = 1:10
   training_set_groups_1 = training_set_groups(1:70000,n);     % actual training data
   model = svmtrain(training_set_groups_1, training_set_features, '-t 2'); % radial

   S=sprintf('Accuracy for %d APUF ===>',n);
   disp(S);

   classify_set_groups_1 = classify_set_groups(1:30000,n);      % actual classifying data
   [predicted_label, accuracy, decision_values] = svmpredict(classify_set_groups_1,
classify_set_features, model);
end
diary off;

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

% SECOND QUESTION
% % SIMULATED DATA
load('chal_APUF_64_100000_bi.mat');
% 100000 * 64
lol = C;
challenge_set = C;
challenge_set(1:100000,65) = ones(100000,1);
% making bipolar
for i = 1:100000
   for j = 1:64
      if lol(i,j) == 0
         lol(i,j) = 1;
      else
         lol(i,j) = -1;
      end
   end
end
%  calculating parity vectors
for i = 1:100000
```

```
    for j = 1:64
        hi = 1;
%         if j == 64
%             challenge_set(i,j) = 1;
%         else
            for k = j:64
                hi = hi * lol(i,k);
            end
            challenge_set(i,j) = hi;
%         end
    end
end
clear C;
clear lol;
% %
load('resp_APUF_64_100000_10_inst_.mat');
response_set = R;
for i = 1:100000
    for j = 1:10
        if response_set(i,j) == 0
            response_set(i,j) = 1;
        else
            response_set(i,j) = -1;
        end
    end
end
clear R;


% setting training and classifying data
training_set_features = challenge_set(1:70000,:);
training_set_groups = response_set(1:70000,:);

classify_set_features = challenge_set(70001:100000,:);
classify_set_groups = response_set(70001:100000,:);

diary('second_part_sim.txt');
diary on;




% % radial kernel
% % training_set_features from first
S=sprintf('<========================Radial
Kernel==============================>');
disp(S);
for n = 1:10
    training_set_groups_1 = training_set_groups(1:70000,n);     % actual training data
    model = svmtrain(training_set_groups_1, training_set_features, '-t 2'); % radial
    S=sprintf('Accuracy for %d APUF ===>',n);
    disp(S);

    classify_set_groups_1 = classify_set_groups(1:30000,n);      % actual classifying data
    [predicted_label, accuracy, decision_values] = svmpredict(classify_set_groups_1,
classify_set_features, model);
end
diary off;
```