MATLAB CODE

1. Perceptron

```
w = zeros(1,2);
T = 1000;
x = zeros(2,1);
u = [1/sqrt(2) -1/sqrt(2)];
loss = 0;
c_loss = zeros(2,1000);
y_pred = zeros(2,1000);
ubound = zeros(2,1000);
for d = 1:2
    loss = 0;
    R = -inf;
    gamma = inf;
    w = zeros(1,2);

    for t = 1 : T
        x = get_instance(d,t);
        y_pred(d,t) = sign(w*x');
        y_true = get_label(d,t);
        R = max(R,x*x') ;
        gamma = min(gamma,y_true*(u*x'))
%         ubound(d,t) = (log10(R*R*(u*u')/(gamma*gamma)))/t;
        ubound(d,t) = (R*R*(u*u')/(t*gamma*gamma));
        if y_pred(d,t) ~= y_true
            w = w + y_true*x;
            loss = loss + 1 ;
        end
        c_loss(d,t) = loss/t;
    end

end

    hold on
    plot(1:1000,c_loss(1,:),'r');
    plot(1:1000,c_loss(2,:),'b');
    h1 = figure();
    plot(1:1000,ubound(1,:),'r');
    h2 = figure();
    plot(1:1000,ubound(2,:),'b');
```

2. Perceptron_b

```
T = 1000;
x = zeros(2,1);
u = [1/sqrt(2) -1/sqrt(2)];
y_pred = zeros(2,1000);
error = zeros(2,1000);
for d = 1:2
w = zeros(1,2);

   for t = 1 : T
      x = get_instance(d,t);
      y_pred(d,t) = sign(w*x');
      y_true = get_label(d,t);
%        ubound(d,t) = (log10(R*R*(u*u')/(gamma*gamma)))/t;
      if y_pred(d,t) ~= y_true
         w = w + y_true*x;
      end
      error(d,t) = error_uniform(u,w);
   end

end

   hold on
   plot(1:1000,error(1,:),'r');
   %plot(1:1000,error(2,:),'b');
%   h1 = figure();
%   plot(1:1000,ubound(1,:),'r');
%   h2 = figure();
%   plot(1:1000,ubound(2,:),'b');
```

3. Active_Perceptron

```
function[c_loss] = active_perceptron(b)
req = zeros(2,1000);
w = zeros(1,2);
T = 1000;
x = zeros(2,1);
u = [1/sqrt(2) -1/sqrt(2)];
k = 1;
loss = 0;
c_loss = zeros(2,1000);
y_pred = zeros(2,1000);
ubound = zeros(2,1000);
for d = 1:2
   loss = 0;
   R = -inf;
```

```
    gamma = inf;
    w = zeros(1,2);

    for t = 1 : T
        x = get_instance(d,t);
        x_cap = (1/sqrt(x*x'))*x ;
        r = w*x_cap' ;
        y_pred(d,t) = sign(r);
        par = b/(b + abs(r)) ;
        z = sign(rand(1) - (1-par)) ;
        R = max(R,x*x');
        if z == 1
            if t==1
                req(d,t) = 1
            else
                req(d,t) = req(d,t-1) + 1;
            end
            y_true = get_label(d,t);
%               ubound(d,t) = (log10(R*R*(u*u')/(gamma*gamma)))/t;
                gamma = min(gamma,y_true*(u*x'));
                ubound(d,t) = (R*R*(u*u')/(t*gamma*gamma));

                if y_pred(d,t) ~= y_true
                    w = w + y_true*x;
                    loss = loss + 1 ;
                end
        else
            if t==1
                ubound(d,t) = 0;
                req(d,t) = 0;
            else
                ubound(d,t) = ubound(d,t-1);
                req(d,t) = req(d,t-1);
            end
        end
        c_loss(d,t) = loss/t;
    end

end

    hold on
    plot(1:1000,c_loss(1,:),'r');
    plot(1:1000,c_loss(2,:),'b');
%   h1 = figure();
%   plot(1:1000,ubound(1,:),'r');
%     h2 = figure();
%     plot(1:1000,ubound(2,:),'b');
   h3 = figure();
   hold on
   plot(1:1000,req(1,:),'r');
   plot(1:1000,req(2,:),'b');
end
```

4. Active_Perceptron_b

```
function[req] = active_perceptron_b(b)
req = zeros(2,1000);

w = zeros(1,2);
T = 1000;
x = zeros(2,1);
u = [1/sqrt(2) -1/sqrt(2)];
y_pred = zeros(2,1000);
error = zeros(2,1000);
for d = 1:2

   for t = 1 : T
      x = get_instance(d,t);

      x_cap = (1/sqrt(x*x'))*x ;
      r = w*x_cap' ;
      y_pred(d,t) = sign(r);
      par = b/(b + abs(r)) ;
      z = sign(rand(1) - (1-par)) ;

      if z == 1
         if t==1
            req(d,t) = 1;
         else
            req(d,t) = req(d,t-1) + 1;
         end

         %y_pred(d,t) = sign(w*x');
         y_true = get_label(d,t);
%           ubound(d,t) = (log10(R*R*(u*u')/(gamma*gamma)))/t;
         if y_pred(d,t) ~= y_true
            w = w + y_true*x;
         end
      else
         if t==1
            req(d,t) = 0;
            else
                req(d,t) = req(d,t-1);
            end
        end
            error(d,t) = error_uniform(u,w);
    end

end

    hold on
    plot(req(1,:),error(1,:),'r');
    %plot(req(2,:),error(2,:));
    %plot(1:1000,error(1,:),'r');
    %plot(1:1000,error(2,:),'b');
%   h1 = figure();
%   plot(1:1000,ubound(1,:),'r');
%   h2 = figure();
%   plot(1:1000,ubound(2,:),'b');

end
```

5. Error_uniform

```
function[theta] = error_uniform(u,w)
norm_u = sqrt(u*u');
u = (1/norm_u)*u ;
norm_w = sqrt(w*w');
w = (1/norm_w)*w ;
theta = acos(u*w')/pi ; % this is the probability of misclassification

end
```