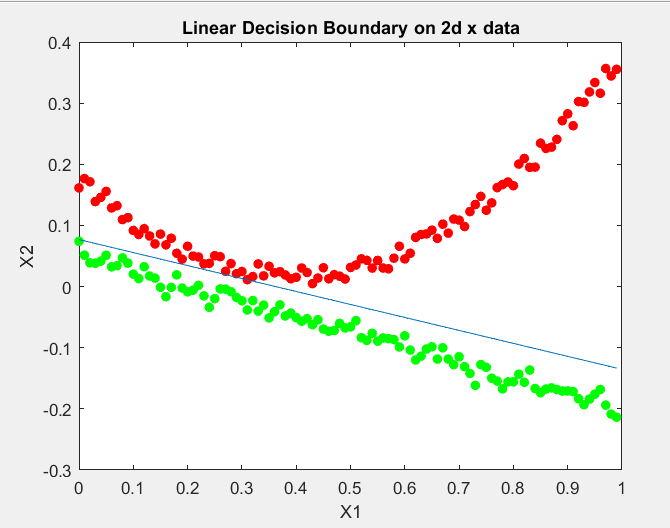
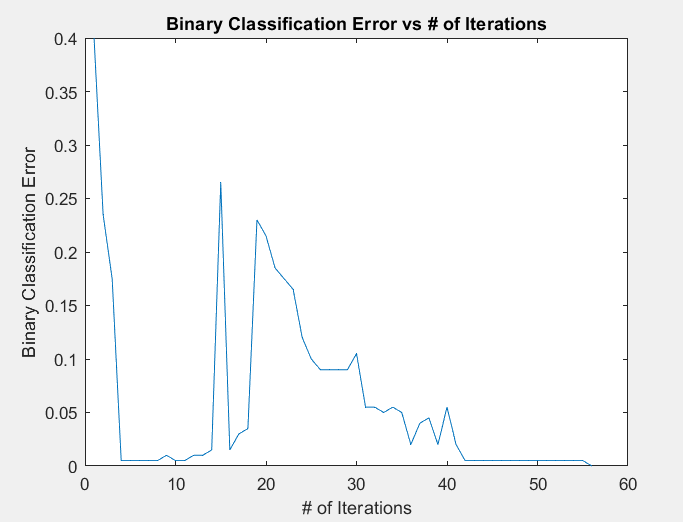
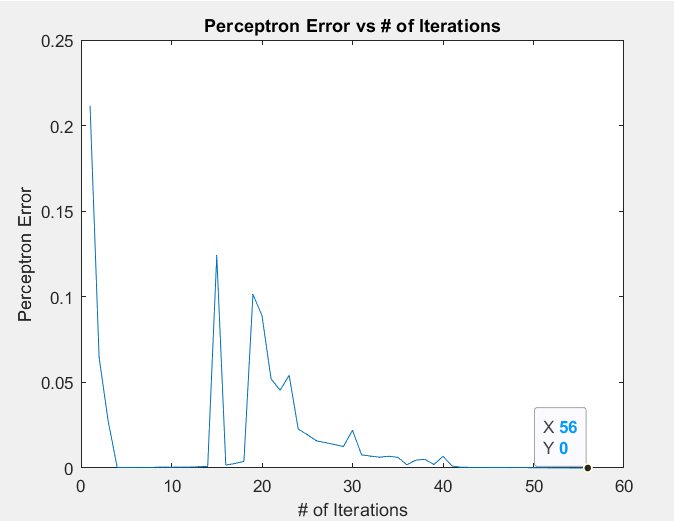
**ML Homework 2: vb2182@nyu.edu**

**Q1:**  
Using Stochastic Gradient Descent (SGD)  
Setting eta (Learning rate or step size) to 1 in the case of SGD, without loss of generality, plotted the following:  
As expected, binary classification error converges to zero comparably slowly than perceptron error (convergence to zero at the 56th iteration).  
Plots are below.  
  






**Code in Matlab:**  
clc; %clear work command window

clear variables;

clf;

path="C:\Users\vxbha\Desktop\VB2182\_ML\_HW2\";

load("data3.mat");

%Initialization of Data

N=length(data);

X=[data(:,1:2),ones(N,1)]; %need to add array of all ones to accomodate bias

Y=data(:,3);

theta=randn(3,1); %3-by-1 vector of random numbers

missed\_values=1;

current\_iteration=1;

binary\_classification\_error=[];

perceptron\_error=[];

%Employing SGD

while(missed\_values~=0)

for i=1:N

Loss=Y(i)\*(X(i,:)\*theta);

if Loss<=0

%Update

theta = theta+Y(i)\*X(i,:)';

y1=classification\_func(theta,X,Y);

y2 = perceptron\_func(theta,X,Y);

binary\_classification\_error(current\_iteration)=mean(y1);

perceptron\_error(current\_iteration)=mean(y2);

end

end

missed\_values=mean(y2);

current\_iteration=current\_iteration+1;

end

figure(1)

for i=1:N

if Y(i)==1

plot(X(i,1),X(i,2),'.','MarkerSize',20,'MarkerEdgeColor','g');

elseif Y(i)==-1

plot(X(i,1),X(i,2),'.','MarkerSize',20,'MarkerEdgeColor','r');

end

hold on

end

xlabel('X1');

ylabel('X2');

x = [min(X(:,1)):1/200:max(X(:,1))];

x2 = (-theta(3)\*ones(N,1)-theta(1)\*x)/theta(2);

plot(x,x2);

title("Linear Decision Boundary on 2d x data");

plotoutput=("Linear Decision Boundary on 2d x data.png");

print(path+plotoutput,"-dpng");

figure(2)

plot(binary\_classification\_error);

xlabel('# of Iterations');

ylabel('Binary Classification Error');

title("Binary Classification Error vs # of Iterations");

plotoutput=("Binary Classification Error vs # of Iterations.png");

print(path+plotoutput,"-dpng");

figure(3)

plot(perceptron\_error);

xlabel('# of Iterations');

ylabel('Perceptron Error');

title("Perceptron Error vs # of Iterations");

plotoutput=("Perceptron Error vs # of Iterations.png");

print(path+plotoutput,"-dpng");

function f = classification\_func(theta,x,Y)

y = x\*theta;

f = ones(length(x),1);

for i=1:length(x)

if y(i)\*Y(i)<0

f(i)=1;

elseif y(i)\*Y(i)>=0

f(i)=0; %instead of -1, I have equated it to zero so the classification error converges to 0.

end

end

end

function f = perceptron\_func(theta,x,Y)

y=x\*theta;

f=zeros(length(x),1);

for i=1:length(x)

if Y(i)\*y(i)<0

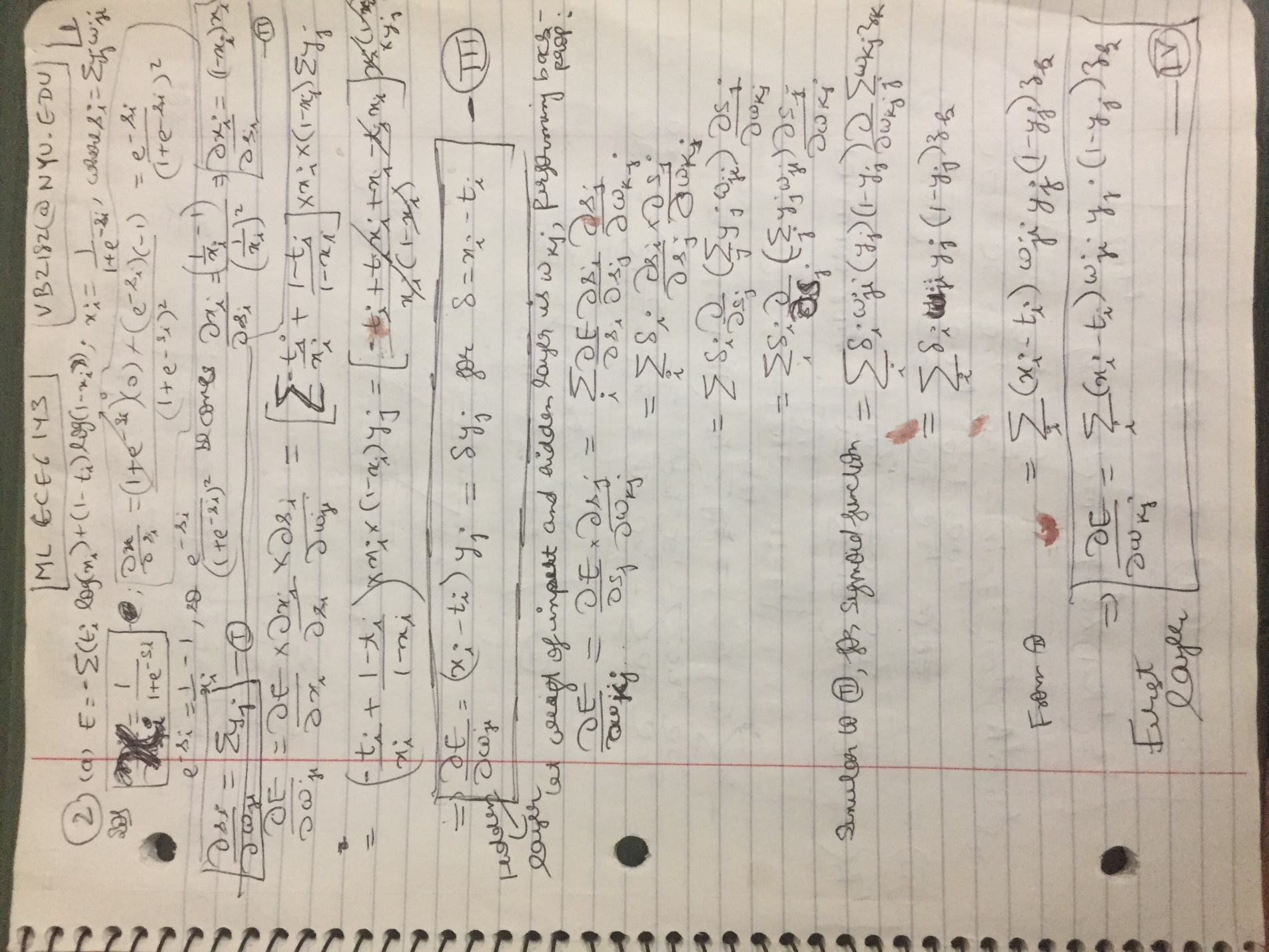
f(i)=-Y(i)\*y(i);

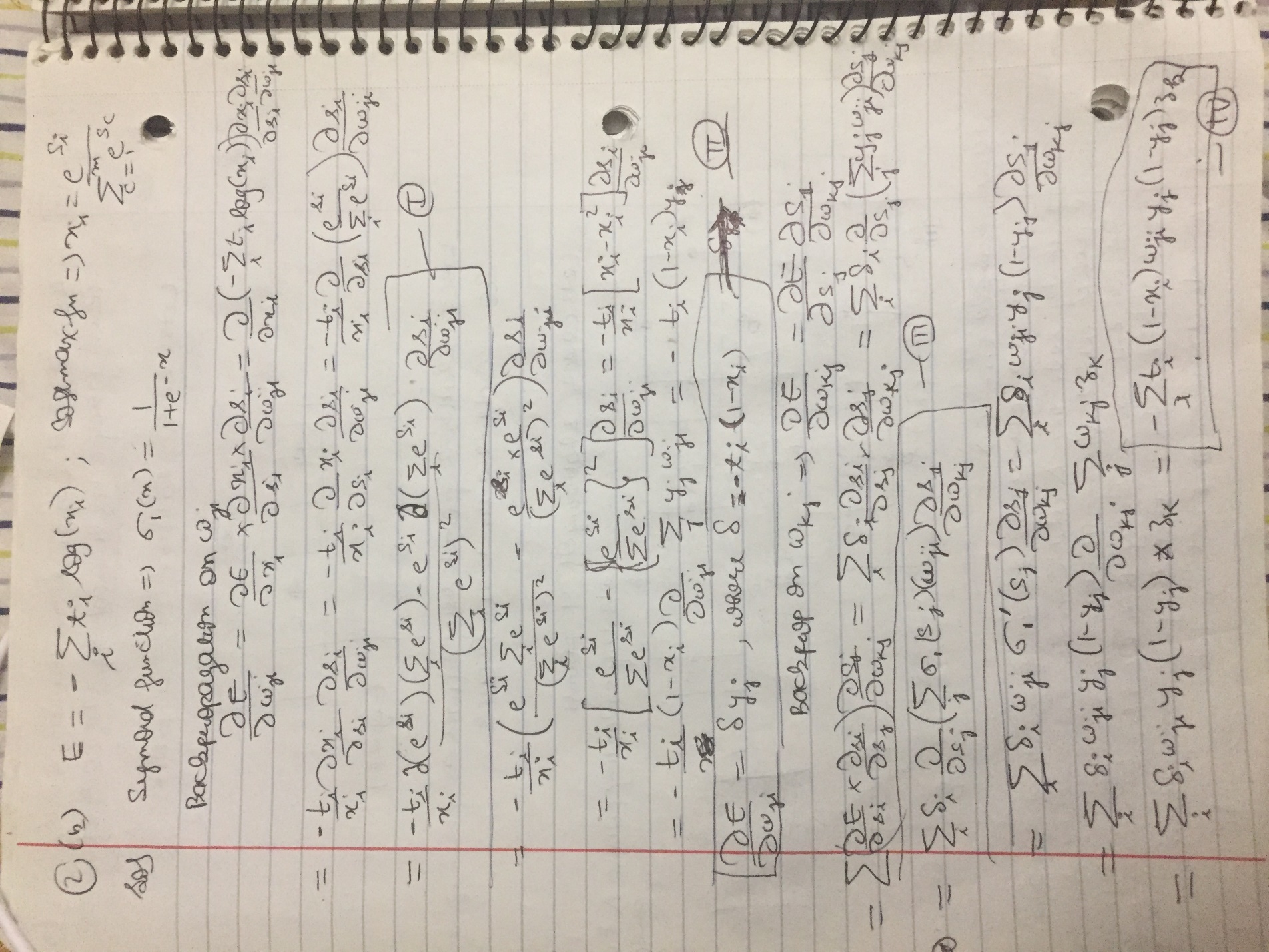
end

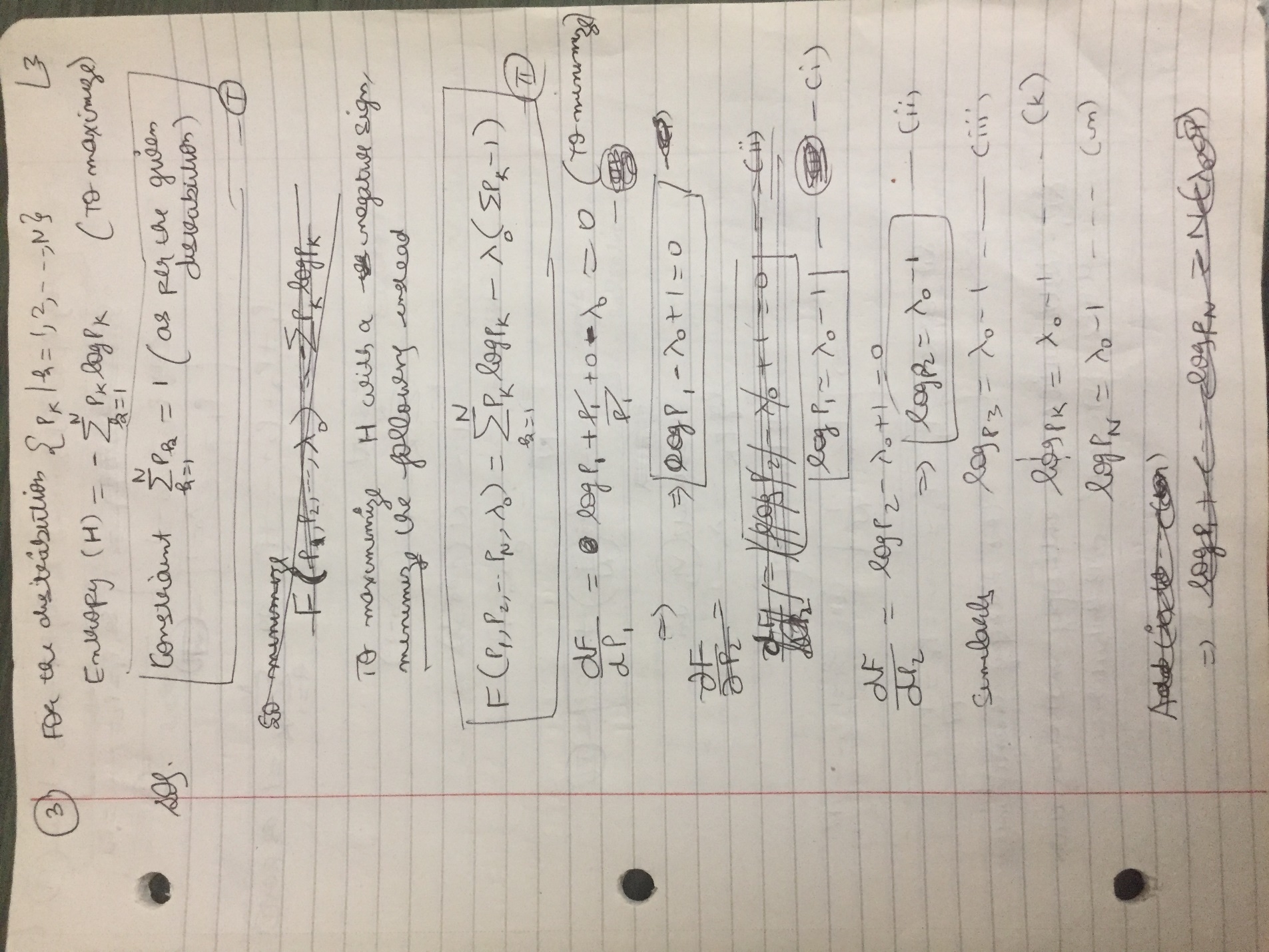
end

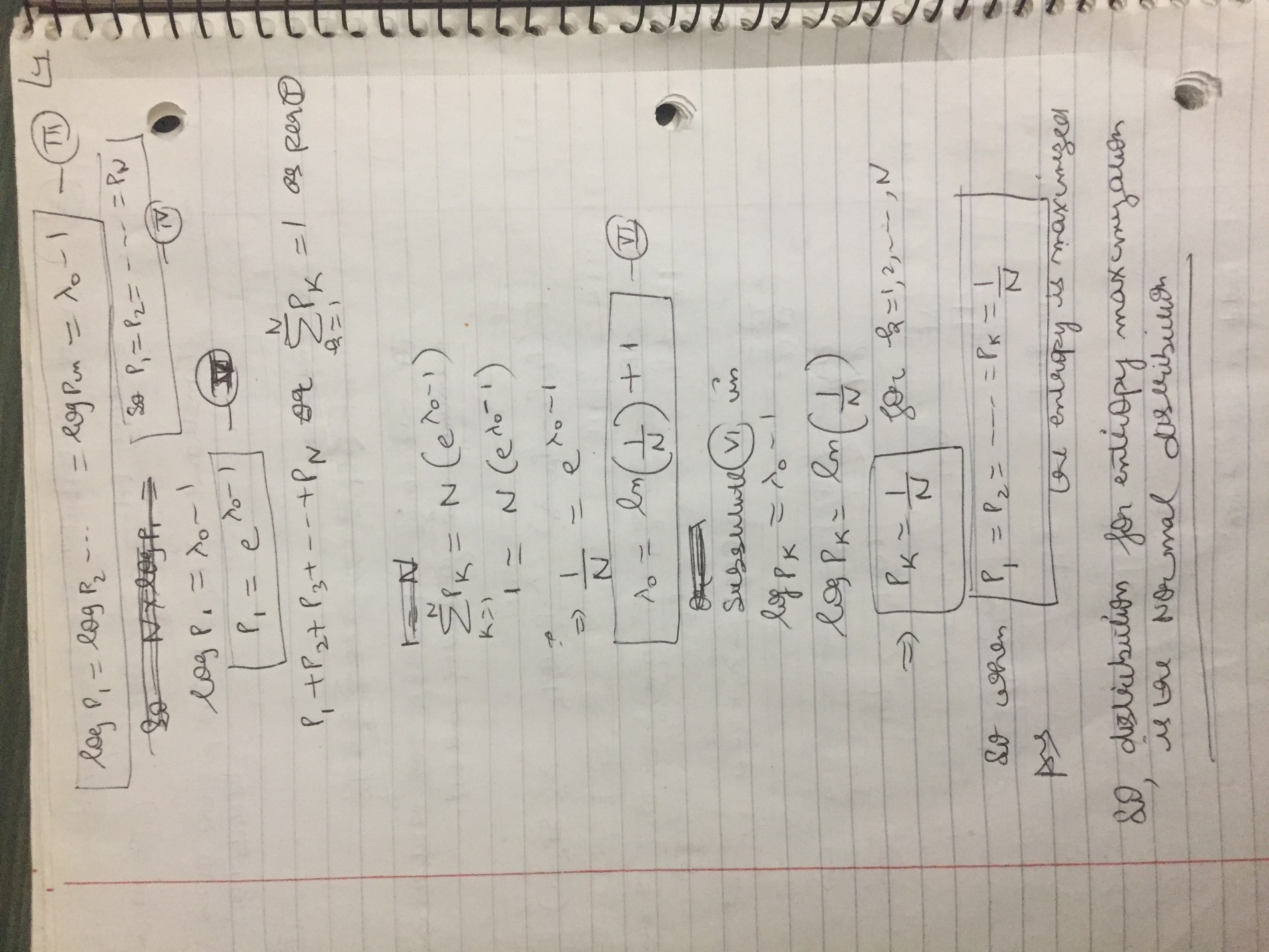
end

**Q2 (a)**



**2 (b)**   


**Q3**  




**Q4**

