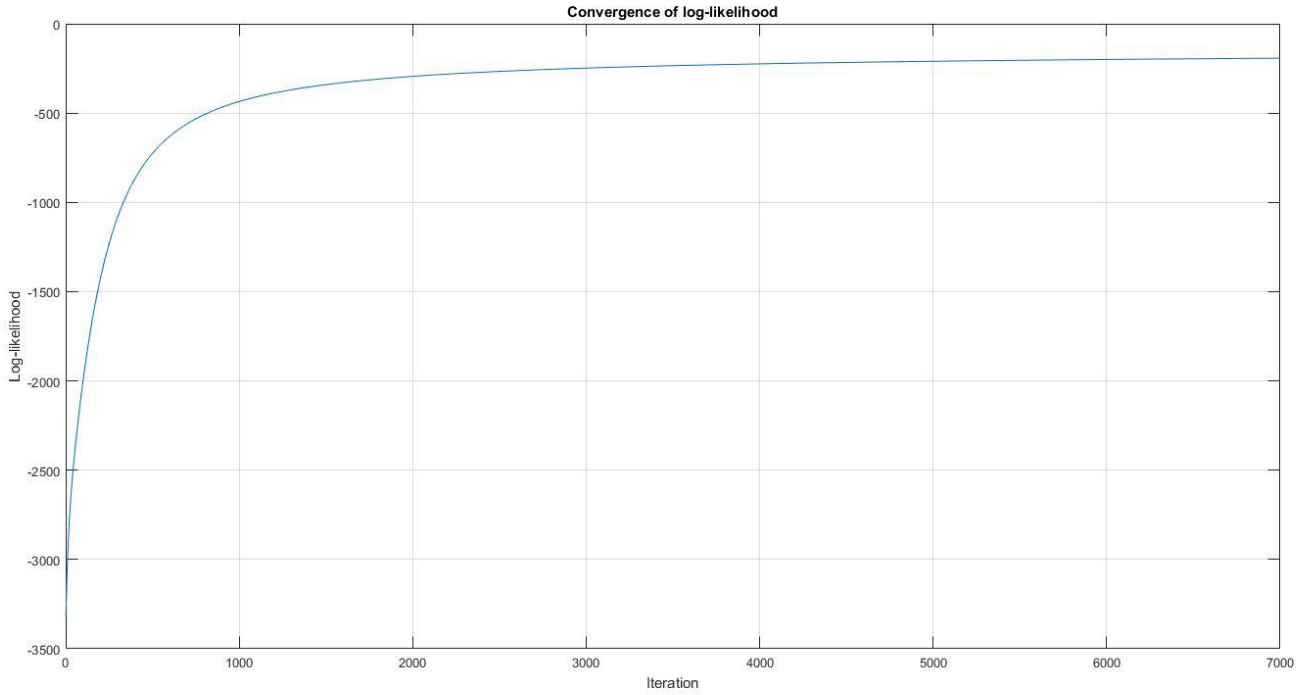
5.5) Handwritten digit classification:

(a) Training –

I used Gradient ascent, to perform logistic regression on the training image files. The evidence for convergence of the log-likelihood value (the algorithm took nearly 20000 iterations to converge within the error bounds, but only the first 7000 are shown below for clarity, and to prove convergence.)




Error on training data:
Tested at 3 iteration levels; where log-likelihood seems to converge, but error keeps decreasing

| | Error rate (%) | | |
|---|---|---|---|
| File | @ 20000 iterations | @ 100000 iterations | @ 300000 iterations |
| Train3.txt | 4 | 3.8571 | 3.7143 |
| Train5.txt | 4.8571 | 3.7143 | 3.7143 |
| Overall | 4.4286 | 3.7857 | 3.7143 |

The optimal 64 x 1 weight vector (obtained by gradient ascent) is shown below as a 8x8 matrix :

w_opt =

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -1.5356 | 0.4585 | 0.9769 | 1.8330 | 0.1012 | 0.9971 | 0.6093 | -0.1493 |
| -0.5768 | -0.1749 | 1.4378 | 0.7288 | 0.2876 | -0.5358 | 0.0119 | -0.0290 |
| -0.2523 | 0.1202 | 0.5985 | 0.9229 | -0.0970 | 0.4739 | -0.1137 | 0.7506 |
| -1.3538 | 0.0783 | 0.6740 | 0.0826 | -0.6201 | -0.1191 | 0.6606 | 1.7053 |
| -0.8064 | 0.3199 | -0.2567 | -1.0011 | -0.1510 | 0.9620 | -0.1387 | -0.9088 |
| -0.5828 | 0.3013 | -1.3256 | -0.3257 | -0.9669 | 0.0445 | 0.1652 | 1.3659 |
| 0.7807 | -0.6237 | -2.5072 | 0.3535 | -0.7027 | 0.5626 | -0.1364 | 0.3411 |
| 1.7000 | -0.5701 | -1.6712 | -1.0595 | 0.0839 | -1.3057 | -0.9689 | -0.1530 |

Error on testing data :

| | Error rate (%) | | |
|---|---|---|---|
| File | @ 20000 iterations | @ 100000 iterations | @ 300000 iterations |
| Test3.txt | 5.25 | 5.75 | 5.5 |
| Test5.txt | 4.75 | 4.75 | 5 |
| Overall | 5 | 5.25 | 5.25 |

We can see that the test error increases with number of iterations, indicating that the model might be overfitting.

Source code:

Utilities:

```
function g = sigmoid(z)
    g= 1./(1+exp(-z));
end

function pError = errorRate(x,y,w)
    pred = sigmoid(x*w)>0.5;
    incorrectlyClassified = length(find(pred~=y));
    pError = (incorrectlyClassified/size(x,1))*100;
end
```

Code:

```
close all;
clear all;

dd = 64;

temp3 = textread('../Data/train3.txt','%d');
for t = 1:length(temp3)/(dd)
    train3(t,:) = temp3(dd*(t-1)+1:dd*t);
end

temp5 = textread('../Data/train5.txt','%d');
for t = 1:length(temp5)/(dd)
    train5(t,:) = temp5(dd*(t-1)+1:dd*t);
end

x = [train3; train5];
y= [zeros(size(train3,1),1); ones(size(train5,1),1)];
T=length(y);

clear temp3 temp5 t

%% Gradient ascent

w = randn(dd,1);
iter = 7000;
eta = 0.02/T;

for i = 1:iter
    p=sigmoid(x*w);
```

```
    grad = x'*(y-p);
    w=w+(eta*grad);
    L(i) = sum((y.*log(p))+((1-y).*log(1-p)));
end

w_opt = reshape(w,[8 8]);

figure;
set(gcf,'color','w');
plot(1:iter,L);
grid on;
xlabel('Iteration');
ylabel('Log-likelihood');
title('Convergence of log-likelihood');

clear eta grad i iter L p T

%% Testing

temp3 = textread('../Data/test3.txt','%d');
for t = 1:length(temp3)/(dd)
    test3(t,:) = temp3(dd*(t-1)+1:dd*t);
end

temp5 = textread('../Data/test5.txt','%d');
for t = 1:length(temp5)/(dd)
    test5(t,:) = temp5(dd*(t-1)+1:dd*t);
end

xt = [test3;test5];
yt = [zeros(size(test3,1),1); ones(size(test5,1),1)];
clear temp3 temp5 t

%% Error rates

% Train errors
trainError3 = errorRate(train3, zeros(size(train3, 1), 1), w);
trainError5 = errorRate(train5, ones(size(train5, 1), 1), w);
overallTrainError = errorRate(x, y, w);

% Test errors
testError3 = errorRate(test3, zeros(size(test3, 1), 1), w);
testError5 = errorRate(test5, ones(size(test5, 1), 1), w);
overallTestError = errorRate(xt, yt, w);
```