Question 4:

Log\_grad.m code:”

function G=log\_grad(y, X, B)

[n,d] = size(X);%n: number of samples, d: number of features

K = size(B,2) + 1; %Total number of classes

%compute gradient

XB = X \* B;

expXB = exp(XB);

prob = expXB ./ (1 + sum(expXB, 2));

prob = [prob, 1 - sum(prob, 2)];

G = zeros(d,K-1);

for k = 1:K-1

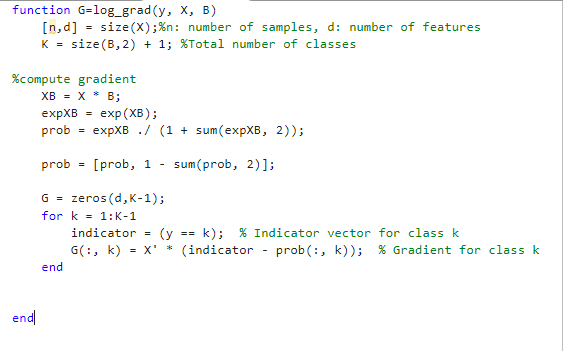
indicator = (y == k); % Indicator vector for class k

G(:, k) = X' \* (indicator - prob(:, k)); % Gradient for class k

end

end

“



Following this code and implementing the logistic\_classify .m and generate these graphs:

When lambda = 0

A graph with a blue line

Description automatically generatedA graph of a training curve

Description automatically generated

A graph with a blue line

Description automatically generated

When lambda = 1

A graph with a blue line

Description automatically generated A graph of a training curve

Description automatically generated

A graph with numbers and lines

Description automatically generated

When lambda = 10

A graph with a line

Description automatically generated A graph of a training curve

Description automatically generated

A graph with numbers and lines

Description automatically generated

When lambda =100

A graph with a line drawn on it

Description automatically generated A graph of a training accuracy

Description automatically generated

A graph with numbers and lines

Description automatically generated

When lambda =200

A graph with a line

Description automatically generated A graph of training accuracy

Description automatically generated

A graph with numbers and lines

Description automatically generated

From the above charts we recognize that:

* Objective value:
  + In general the obj value decreases over iterations, showing convergence for all lambda values.
  + While for smaller lambda values (lambda = 0), the convergence is faster and reach a lower final value.
  + And for larger values, the convergence is slower and the final obj is higher.
  + To wrap up: a moderate lambda balances fast convergence and regularization.
* Training Accuracy:
  + For smaller lambda values, training accuracy is higher as the model overfits to the training data. While for larger values, training accuracy decreases because the model becomes over regularized.
  + To wrap up, moderate lambda has in between better training accuracy.
* Testing Accuracy:
  + This improves with moderate values as the model geralization better to unseen data.
  + While for small values, testing accuracy is lower since the model overfits to the training sets.
  + And for larger values, it declines again due to underfitting since the model becomes too simplistic.