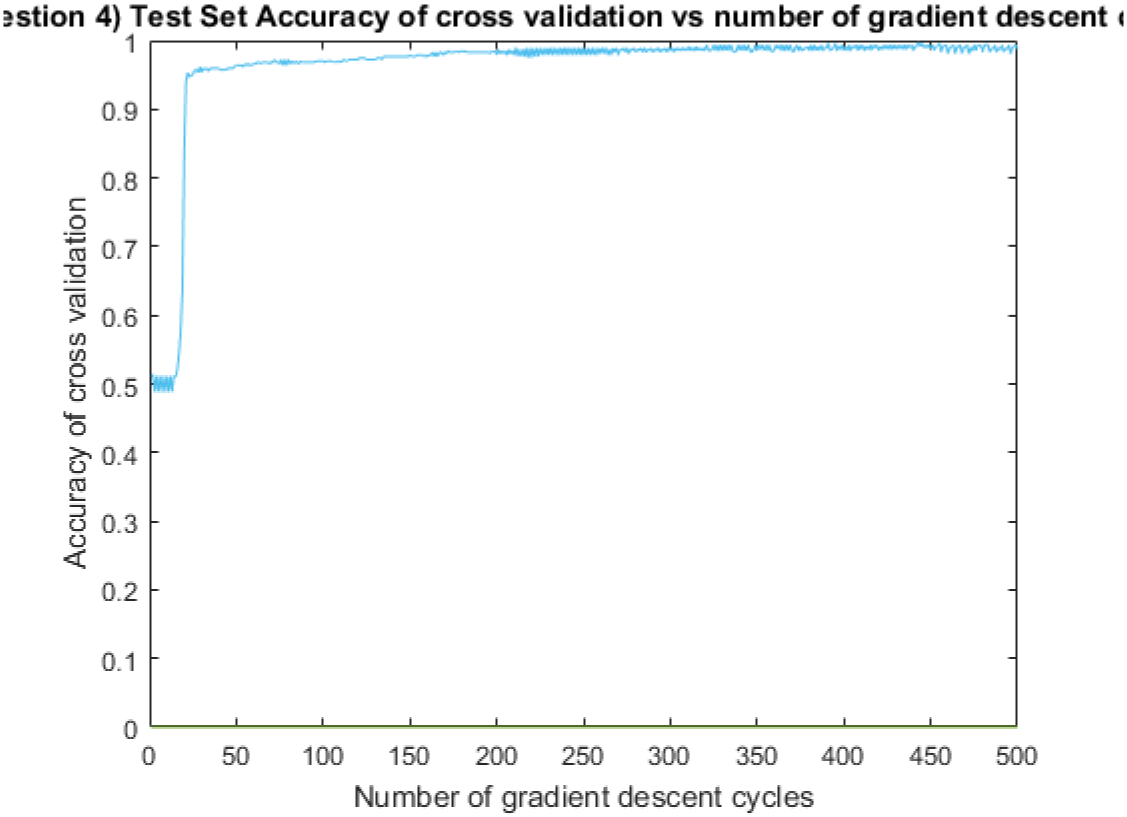
```
% *****  QUESTION 4 Logistic Regression ***********************

function main(path)

clear variables

load('testLabels.mat')
load('trainLabels.mat')
load('testSet.mat')
load('trainSet.mat')

% Learning with LOOCV (leave one out cross validation)
learning = [10^-5 10^-4 10^-3 10^-1 10 1]; % constant learning rates

% Accuracy of cross validation will be stored in this matrix
Accuracy = zeros(size(learning,2),size(trainSet,1));

gradDescentCycles = 20; % Number of cycles for gradient descent
 optimization

omega = zeros(1,size(trainSet,2));
w0 = 0;

%Going over different learning values
for i = 1:size(learning,2)

    % Excluding data points from the set for LOOCV
    for  j = 1:size(trainSet,1)

        trainSetCurrent = trainSet;
        trainLabelsCurrent=trainLabels;

        trainSetCurrent(j,:) = [];
        trainLabelsCurrent(j) =[];

        % Using gradient descent for optimization
            for k = 1:gradDescentCycles

                omegaSum = sum(omega.*trainSet,2);
                exponential = exp((w0 + omegaSum)./(1 + exp(w0 +
 omegaSum)));

                Gradient1 = sum(trainLabels - exponential);
                Gradient2 = sum(trainSet.*(trainLabels -(exp((w0 +
 omegaSum)./(1 + exp(w0 + omegaSum))))),1);

                omega(:) = omega(:) + learning(i)*Gradient2(:);

                w0 = w0 + learning(i)*Gradient1;

            end
        prediction = w0 + sum(omega.*trainSet,2);
        prediction(prediction >= 0) = 1;
        prediction(prediction < 0) = 0;
        Acc1 = (prediction - trainLabels == 0);
        Acc1 = sum(Acc1)/size(trainLabels,1);
        Accuracy(i,j) = Acc1;

    end

end

Accuracy = sum(Accuracy,2)/size(trainSet,1)';
disp('Accuracy of cross validation for the 5 Learning rates: (in
 order)');
display(Accuracy);

% Best learning rate is found
bestLearning = learning(find(max(Accuracy) == Accuracy));
display('Best learning rate L = ')
display(bestLearning);

% Now model will be trained with L = 10 learning rate and will be
% tested on the test set

omega = zeros(1,size(trainSet,2));
w0 = 0;

gradDescentCycles = 500;

% Using gradient descent for optimization, using the best learning
 rate
% found
for i = 1:gradDescentCycles

    omegaSum = sum(omega.*trainSet,2);
    exponential = exp((w0 + omegaSum)./(1 + exp(w0 + omegaSum)));

    Gradient1 = sum(trainLabels - exponential);
    Gradient2 = sum(trainSet.*(trainLabels -(exp((w0 + omegaSum)./(1 +
 exp(w0 + omegaSum))))),1);

    omega(:) = omega(:) + bestLearning*Gradient2(:);

    w0 = w0 + bestLearning*Gradient1;

    prediction = w0 + sum(omega.*testSet,2);
    prediction(prediction >= 0) = 1;
    prediction(prediction < 0) = 0;
    Acc2 = (prediction - testLabels == 0);
    Acc2 = sum(Acc2)/size(testLabels,1);
    TestAccuracy(i,j) = Acc2;

end
```

```
disp('Test Accuracy:');
display(TestAccuracy(end));

figure()
plot(TestAccuracy)
title('Question 4) Test Set Accuracy of cross validation vs number of
 gradient descent cycles')
xlabel('Number of gradient descent cycles');
ylabel('Accuracy of cross validation')


end
```

```
Accuracy of cross validation for the 5 Learning rates: (in order)

Accuracy =

    0.7284
    0.8868
    0.9885
    0.9890
    0.9894
    0.9905

Best learning rate L =

bestLearning =

     1

Test Accuracy:
    0.9873
```



Published with MATLAB® R2016b