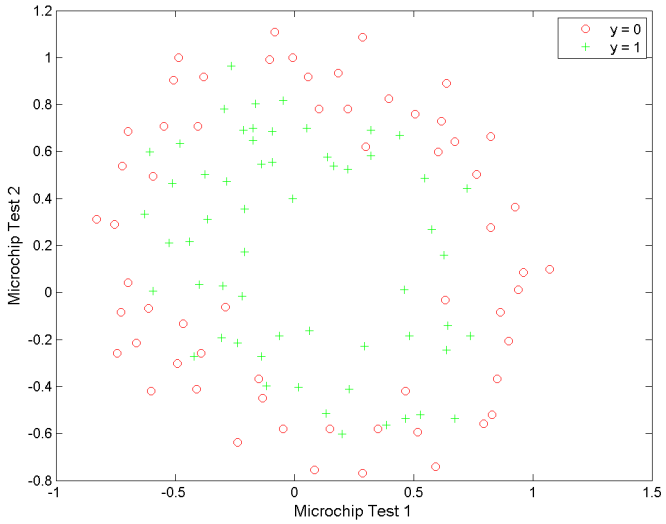
ROCHESTER INSTITUTE OF TECHNOLOGY
DEPARTMENT OF COMPUTER ENGINEERING
CMPE 677 Machine Intelligence
HW #4

Name:_____

Due 11:55pm, 09/26/2018, submit via Dropbox. Work alone. All questions pertain to Matlab/Octav.

1. (15 pts) Download (ex2data2.txt, costFunctionLogisticRegression_slow.m, sigmoid.m, mapFeature.m, plotDecisionBoundary.m) from myCourses in the hwk tab. Starting with the code:

```
clear ; close all; clc

% Load Data  (from Andrew Ng Machine Learning online MOOC)
%  The first two columns contains the X values and the third column
%  contains the label (y).

data = load('ex2data2.txt');  %data is 118x3
X = data(:, [1, 2]); y = data(:, 3);

index0 = find(y == 0);
index1 = find(y == 1);

hold off; plot(X(index0,1),X(index0,2),'ro'); hold on
plot(X(index1,1),X(index1,2),'g+');

% Labels and Legend
xlabel('Microchip Test 1','fontsize',12)
ylabel('Microchip Test 2','fontsize',12)
legend('y = 0', 'y = 1')
```




```
% The data points that are not
```

```
%  linearly separable. However, you would still like to use logistic
%  regression to classify the data points.
%
%  To do so, you introduce more features to use -- in particular, you add
%  polynomial features to our data matrix (similar to polynomial
%  regression).
% Note that mapFeature also adds a column of ones for us, so the intercept
% term is handled
degree=6;  %degree of polynomial allowed
Xdata = mapFeature(X(:,1), X(:,2),degree);

% Initialize fitting parameters
initial_theta = zeros(size(Xdata, 2), 1);

% Set regularization parameter lambda to 1
lambda = 1;

% Compute and display initial cost and gradient for regularized logistic
% regression
[cost, grad] = costFunctionLogisticRegression_slow(initial_theta, Xdata, y, lambda);
fprintf('Cost at initial theta (zeros): %f\n', cost);  %should be about 0.693
```

Copy costFunctionLogisticRegression_slow.m to costFunctionLotisticRegression.m. Modify costFunctionLogisticRegression.m so that there are no for loops. When you execute the code:

```
[cost, grad] = costFunctionLogisticRegression(initial_theta, Xdata, y, lambda);
fprintf('Cost at initial theta (zeros): %f\n', cost);  %should be about 0.693
```

your answer should be identical to costFunctionLogisticRegression_slow.

Ans (Show code for costFunctionLogisticRegression.m):

2. (15 pts) Using the following code and Xdata and y from problem 1:

```
% Initialize fitting parameters
initial_theta = zeros(size(Xdata, 2), 1);

% Set regularization parameter lambda to 1 (you should vary this)
lambda = 1;

% Set Options
options = optimset('GradObj', 'on', 'MaxIter', 400);

% Optimize
% Specifying function with the @(t) allows fminunc to call our costFunction
% The t is an input argument, in this case initial_theta
[theta, J, exit_flag] = ...
   fminunc(@(t)(costFunctionLogisticRegression(t, Xdata, y, lambda)), initial_theta, options);
```

Create four plots using plotDecisionBoundary.m (with degree=6) of lambda=0, 1, 10, and 100.
Write your own code to compute the classification accuracy, noting that if sigmoid(Xdata*theta) >= 0.5 we are in class '1', otherwise we are in class '0'. The title of each plot should be of the form:
λ=<value>, Accuracy=<value>%

Ans: Show your four plots at 50% size and code to generate plots and compute Accuracy

3. (15 pts) In the previous problem, no regularization is favored as we are over fitting to the training set. To get around this, we use cross validation. In cross validation, we split the training data into partitions: use all but one partition at a time for training, and the left out partition for testing; and rotate through all partitions. Unless stated otherwise, the following code should be used to implement cross validation for all future project in this class. In this case, you will modify this code for logistic regression (once again, please use Xdata and y from problem 1):

```
% the matlab functions you want to use are crossvalind.m and confusionmat.m_
% Xdata- A vector of feature, nxD, one set of attributes for each dataset sample
% y- A vector of ground truth labels, nx1 (each class has a unique integer value), one label for
each dataset sample
% numberOfFolds- the number of folds for k-fold cross validation
numberOfFolds=5;
rng(2000);  %random number generator seed
CVindex = crossvalind('Kfold',y, numberOfFolds);

method='LogisticRegression'

lambda=1
   for i = 1:numberOfFolds
      TestIndex = find(CVindex == i);
      TrainIndex = find(CVindex ~= i);

      TrainDataCV = Xdata(TrainIndex,:);
      TrainDataGT =y(TrainIndex);

      TestDataCV = Xdata(TestIndex,:);
      TestDataGT = y(TestIndex);

      %
      %build the model using TrainDataCV and TrainDataGT
      %test the built model using TestDataCV
      %
      switch method
         case 'LogisticRegression'
            % for Logistic Regression, we need to solve for theta
            % Insert code here to solve for theta...


            % Using TestDataCV, compute testing set prediction using
            % the model created
            % for Logistic Regression, the model is theta
            % Insert code here to see how well theta works...
```

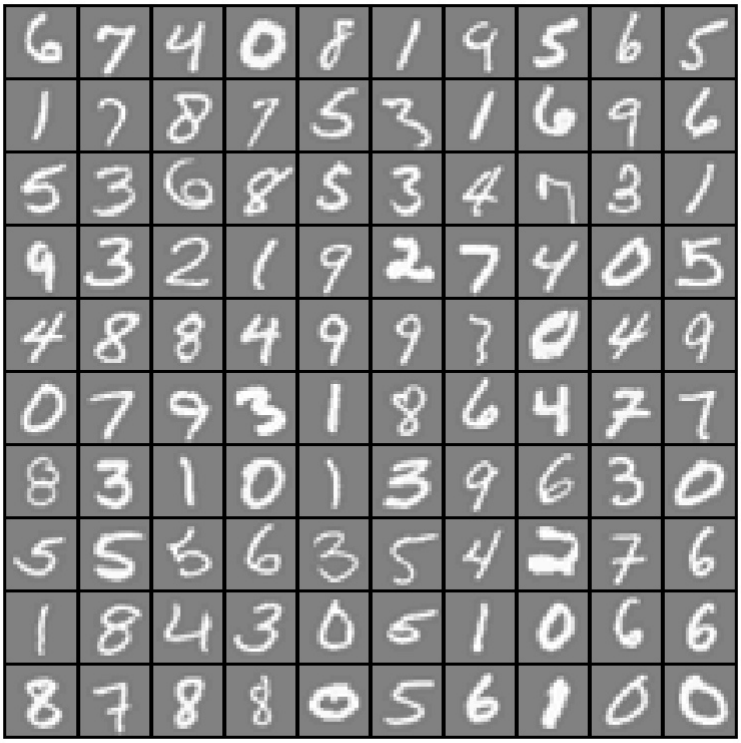
```
            TestDataPred=

        case 'KNN'
            disp('KNN not implemented yet')
        otherwise
            error('Unknown classification method')
    end

    predictionLabels(TestIndex,:) =double(TestDataPred);
end

confusionMatrix = confusionmat(y,predictionLabels);
accuracy = sum(diag(confusionMatrix))/sum(sum(confusionMatrix));

fprintf(sprintf('%s: Lambda = %d, Accuracy = %6.2f%%%% \n',method,
lambda,accuracy*100));
fprintf('Confusion Matrix:\n');
[r c] = size(confusionMatrix);
for i=1:r
    for j=1:r
        fprintf('%6d ',confusionMatrix(i,j));
    end
    fprintf('\n');
end
```

Run this code with lambda = 0, then 1, then 10, then 100. Show the four confusion matrices and accuracies.

Ans: Show code, 4 confusion matrices, and 4 accuracies

4. (10 pts) This problem extends binary classification techniques to multi-class support. Download (ex3data1.mat, displayData.m, fmincg.m) from myCourses in the hwk tab. Starting with the code:

```
clear ; close all; clc

% Load Training Data- Andrew Ng Machine Learning MOOC
load('ex3data1.mat'); % training data stored in arrays X, y
n = size(X, 1);
num_labels =  length(unique(y));          % 10 labels, from 1 to 10   (note  "0" is mapped to label
10)

% Randomly select 100 data points to display
rng(2000);  %random number generator seed
rand_indices = randperm(n);
sel = X(rand_indices(1:100), :);

displayData(sel);
%print -djpeg95 hwk4_4.jpg
```



For multiclass classification with C classes, we build c=1..C binary classification models. For each model, we are either class c or not class c. Our code does its training with the variables TrainDataCV and TrainDataGT. For multi-class classification, TrainDataGT is a $n\times1$ vector, where each entry contains the value 1 through 10 corresponding to digits 1:9,0. To convert this to a binary vector, where each entry is a 1 or a 0, we can do a loop of the sort:

```
for c=1:num_labels
    TrainDataGT_binary = (TrainDataGT == c);  % form vector of 0 and 1's
    % then solve for theta using TrainDataCV and TrainDataGT_binary
    %keep track of c separate theta values
end
```

The only complication here is that we now need to keep track of C variants of theta! To see which class each test point belongs to, we apply logistic regression C times, and the class with the highest predicted value is the class we assign our test sample to. We have:

```
for c=1:num_labels
    TrainDataGT_binary = (TrainDataGT == c);  % form vector of 0 and 1's
    % then solve for theta using TrainDataCV and TrainDataGT_binary

    theta = fmincg…
    %keep track of c separate theta values
    pred(c) = sigmoid(TestDataCV * theta);
end
[maxval,TestDataPred]=max(pred);  %Note: the higher pred, the more certain we are with class c
```

Use the following code to perform multiclass prediction on our dataset, then show the final confusion matrix and accuracy. Note- you are doing 5 folds × 10 classes or solving for theta 50 times, so this code will take a few minutes to complete.

```
clear ; close all; clc

% Load Training Data- Andrew Ng Machine Learning MOOC
load('ex3data1.mat'); % training data stored in arrays X, y
n = size(X, 1);
num_labels =  length(unique(y));          % 10 labels, from 1 to 10   (note  "0" is mapped to label
10)

% Randomly select 100 data points to display
rng(2000);  %random number generator seed
rand_indices = randperm(n);
sel = X(rand_indices(1:100), :);

Xdata = [ones(n, 1) X];
% the matlab functions you want to use are crossvalind.m and confusionmat.m_
% Xdata- A vector of feature, nxD, one set of attributes for each dataset sample
% y- A vector of ground truth labels, nx1 (each class has a unique integer value), one label for
each dataset sample
% numberOfFolds- the number of folds for k-fold cross validation
numberOfFolds=5;
rng(2000);  %random number generator seed
CVindex = crossvalind('Kfold',y, numberOfFolds);

method='LogisticRegression'

lambda = 0.1;
for i = 1:numberOfFolds
    TestIndex = find(CVindex == i);
    TrainIndex = find(CVindex ~= i);

    TrainDataCV = Xdata(TrainIndex,:);
    TrainDataGT =y(TrainIndex);

    TestDataCV = Xdata(TestIndex,:);
    TestDataGT = y(TestIndex);

    %
    %build the model using TrainDataCV and TrainDataGT
    %test the built model using TestDataCV
    %
    switch method
        case 'LogisticRegression'
            % for Logistic Regression, we need to solve for theta
            % Initialize fitting parameters
            all_theta = zeros(num_labels, size(Xdata, 2));

            for c=1:num_labels
                % Set Initial theta
                initial_theta = zeros(size(Xdata, 2), 1);
                % Set options for fminunc
```

```
            options = optimset('GradObj', 'on', 'MaxIter', 50);

            % Run fmincg to obtain the optimal theta
            % This function will return theta and the cost
            [theta] = ...
               fmincg (@(t)(costFunctionLogisticRegression(t, TrainDataCV, (TrainDataGT == c),
lambda)), ...
               initial_theta, options);

            all_theta(c,:) = theta;
         end


         % Using TestDataCV, compute testing set prediction using
         % the model created
         % for Logistic Regression, the model is theta
         % Insert code here to see how well theta works...
         all_pred = sigmoid(TestDataCV*all_theta');
         [maxVal,maxIndex] = max(all_pred,[],2);
         TestDataPred=maxIndex;

      case 'KNN'
         disp('KNN not implemented yet')
      otherwise
         error('Unknown classification method')
   end

   predictionLabels(TestIndex,:) =double(TestDataPred);
end

confusionMatrix = confusionmat(y,predictionLabels);
accuracy = sum(diag(confusionMatrix))/sum(sum(confusionMatrix));

fprintf(sprintf('%s: Lambda = %d, Accuracy = %6.2f%%%% \n',method, lambda,accuracy*100));
fprintf('Confusion Matrix:\n');
[r c] = size(confusionMatrix);
for i=1:r
   for j=1:r
      fprintf('%6d ',confusionMatrix(i,j));
   end
   fprintf('\n');
end
```

Ans: Show confusion matrix and accuracy

5. (15 pts) Modify the code in the previous example for the case method ='KNN'. Use the built in function, knnsearch.m, with k=3. To solve for the final prediction, take the mode of the three closest neighbors.

Ans: show accuracy, confusion matrix, and code