CS74/174 Homework #1

Machine Learning and Statistical Data Analysis: Winter 2016

Due: Jan 23, 2016

This may be your first homework using MATLAB ; please see Canvas and the course webpage for links to tutorials to help you start using it. In matlab, "help <functionname>" is often a good idea; e.g., "help plot", "help find"...

The homework should be submitted electronically via Canvas. It should be a ZIP file which includes the main report in a PDF format, and a folder named as "code" that includes all the MATLAB functions and code. Please make sure your code is directly implementable, or you will not receive any credit otherwise. PDF files can be created using Word, OpenOffice, or LaTeX or any other way you prefer. If you want to use LaTeX, there is a template under this folder (`homework_latex_template.pdf`) that you can use. It is important that you include enough detail that we know how you solved the problem, since otherwise we will be unable to grade it.

Please include all the figures in the main PDF document with enough comments; (if you write your PDF using Latex, please **do not** upload your tex file and individual figures; just the PDF is enough). In Windows or Mac, you can import MATLAB figures into your document directly using copy/paste operations, or you can export a figure from MATLAB directly to pdf using e.g. "print -dpdf <filename>". You can also use this to create and import JPEG or PNG files, but if you do so, please ensure that they are of sufficient resolution to be clear in the resulting document ("-r<value>" controls the resolution).

Sometimes, you may find it useful to set the random number seed at the beginning, e.g., `rand('state',0); randn('state',0);` to ensure consistent behavior each time. Alternatively, if you merely want to remember what commands you executed, the `diary` function may be helpful.

There will be **undergraduate only** problems and **graduate only** problems. A graduate student will not get credit by solving **undergraduate only** problems, while a undergraduate student can get extra credit if he/she solves the **graduate only** problems.

Problem 0: Getting Connected

Please join our class forum on Piazza,
`https://piazza.com/dartmouth/winter2016/cosc07401cosc17401wi16/home`
Please post your questions and discussion points on Piazza, rather than by email to me or the TA, since chances are that other students have the same or similar questions, and will be helped by seeing the discussion.

Problem 1: Matlab & Data Exploration

In this problem, we will explore some basic statistics and visualizations of an example data set. First, download and load the "Fisher iris" data set into Matlab (or Octave):

```
iris=load('data/iris.txt');     % load the text file
y = iris(:,end);           % target value is last column
X = iris(:,1:end-1);       % features are other columns
whos                       % show current variables in memory and sizes
size(X,2), size(X,1)    % get the number of features, and the number of data points.
```

The Iris data consist of four real-valued features used to predict which of three types of iris flower was measured (a three-class classification problem).

(a) **[2 points]** For each feature, plot a histogram ("`hist`") of the data values

(b) **[2 points]** Compute the mean of the data points for each feature (`mean`)

(c) **[2 points]** Compute the variance and standard deviation of the data points for each feature

(d) **[2 points]** "Normalize" the data by subtracting the mean value from each feature, and dividing by its standard deviation. (This will make the data zero-mean and unit variance.) Note: you can do this with a for-loop (easy, but slow in Matlab), or in a "vectorized" form using `repmat` or `bsxfun` (faster, but harder to read). Please show your code in your PDF report.

(e) **[2 points]** For each pair of features (1,2), (1,3), and (1,4), plot a scatterplot (see '`plot`" or "`scatter`") of the feature values, colored according to their target value (class). (For example, plot all data points with $y = 0$ as blue, $y = 1$ as green, etc.) Note: if you wish to overlay several plot commands, use "`hold on`" before subsequent plots; to stop this behavior, use "`hold off`".

Problem 2: kNN predictions

In this problem, you will continue to use the Iris data and explore a KNN classifier using MATLAB.

First, we will shuffle and split the data into training and test subsets:

```
iris=load('data/iris.txt'); y=iris(:,end); X=iris(:,1:end-1);
% Note: indexing with ":" indicates all values (in this case, all rows);
%  indexing with a value ("1", "end", etc.) extracts only that one value (here, columns);
%  indexing rows/columns with a range ("1:end-1") extracts any row/column in that range.

n = size(X,1); d = size(X,2);  % n: # of  instances; d: # of features

dxtrain = 1:round(0.75*n); dxtest = setdiff(1:n, dxtrain);
Xtrain = X(dxtrain, :); Ytrain = y(dxtrain,:);
Xtest = X(dxtest, :); Ytest = y(dxtest,:);
% Split data into 75/25 train/test
```

(a) **[8 points]** This problem involves implementing KNN classifier. Please check `KNN.m` in which I have implemented the case when $K = 1$:

```
Ytest_hat = KNN(Xtest, 1, Xtrain, Ytrain);
% KNN prediction with K=1;

Error_test = mean(Ytest_hat ~= Ytest);
% Calculate testing error;

Ytrain_hat = KNN(Xtrain, 1, Xtrain, Ytrain);
Error_train = mean(Ytrain_hat ~= Ytrain);
% Calculate training error (which is zero in this case (since K = 1));
```

Please complete the code for the general case when $K > 1$, and plot the training and testing errors as $K = [1, 3 \ldots, 9]$ (x-axis is K, and y-axis is the training and testing error). We set K to be odd numbers to avoid ties in majority voting. Pick the optimal K based on your plot.

(b) **[8 points]** The standard KNN is based on Euclidean distance $d(x^{(1)}, x^{(2)}) = \sqrt{\sum_i (x_i^{(1)} - x_i^{(2)})^2}$. We can generalize KNN to base on more general notion of distance. One example of this the weighted Euclidean distance:

$$d_w(x^{(1)}, x^{(2)}) = \sqrt{\sum_i w_i (x_i^{(1)} - x_i^{(2)})^2}$$

where each feature x_i is assigned with an importance weight w_i. Please complete `WeightedKNN.m` to implement this weighted KNN classifier. Try your code with the first two features of Iris data:

```
w = [2,1]; K = 2;
Ytest_hat = WeightedKNN(Xtest(:,1:2), K, w, Xtrain(:,1:2), Ytrain)
 % KNN prediction;

 Error_test = mean(Ytest_hat ~= Ytest);
 % Calculate testing error;

 Ytrain_hat = WeightedKNN(Xtrain(:,1:2), K, Xtrain(:,1:2), Ytrain);
 Error_train = mean(Ytrain_hat ~= Ytrain);
 % Calculate training error
```

Now, fix $w(2) = 1$ and try your code for the cases when $w(1) = [0.01, 0.1, 1, 10, 100]$. Show the plot of the training and testing errors with different values of $w(1)$ and pick the best $w(1)$ based on your plot.

Problem 3: Maximum Likelihood Estimation on Discrete Variables

Assume X is a discrete random variable that takes values in $\{1, 2, 3\}$, with probability $\Pr[X = 1] = \theta_1$, $\Pr[X = 2] = 2\theta_1$ and $\Pr[X = 3] = \theta_2$ where $\theta = [\theta_1, \theta_2]$ is an unknown parameter to be estimated.

Now assume we observe a sequence $D = \{x_1, x_2, \ldots, x_n\}$ that is drawn *i.i.d.* from X; we assume the number of $1, 2, 3$ in D are n_1, n_2, n_3, respectively.

(a) **[2 points]** To ensure that $\Pr[X = i]$ is a valid probability mass function, what constraint we should put on $\theta = [\theta_1,\ \theta_2]$?

(b) **[2 points]** Write down the joint probability $\Pr[D\ ;\ \theta]$ and the log probability $\Pr[D\ ;\ \theta]$.

(c) **[5 points]** Calculate the maximum likelihood estimation $\hat{\theta}$ based on D.

Problem 4: Maximum Likelihood Estimation on Gaussian Variables

Consider the bivariate Gaussian random variable $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim \mathcal{N}(\mu, \Sigma)$ with mean $\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}$ and covariance matrix $\Sigma = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & \sigma_{22} \end{bmatrix}$; its probability density function is defined as

$$p(x) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \exp(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu)),$$

where $\det(\Sigma)$ denotes the determinant of matrix Σ, and "$\top$" represents the transpose, $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top = [x_1, x_2]$.

(a) [**5 points**] Complete the MATLAB function `pdfGaussian2D.m` to calculate the probability density function. After you finish it, use it to visualize the curve and its contour in MATLAB:

```
mu = [1; -1];  Sigma = [3,1; 1,3]; % define parameters

[X1,X2] = meshgrid([-6:1:6], [-6:1:6]); % specify points in a grid

% calculate the likelihood
P = zeros(size(X1));
for i = 1:size(X1,1)
    for j = 1:size(X2, 2)
        P(i,j)=pdfGaussian2D([X1(i,j), X2(i,j)]', mu, Sigma);
    end
end

figure;
subplot(1,2,1); contour(X1,X2,P, 'color', 'red'); title('Contour');
subplot(1,2,2); surf(X1, X2, P); title('Surface');
```

(b) [**5 points; Undergraduate Only**] Assume $\sigma_{12} = 0$, show that the maximum likelihood estimator of $\mu_1, \mu_2, \sigma_{11}, \sigma_{22}$ are the empirical means and variances, respectively.

(c) [**5 points; Graduate Only**] Show that the maximum likelihood estimator $\hat{\mu}$ and $\hat{\Sigma}$ are the empirical mean and covariance, respectively. (hint: you can let $Q = \Sigma^{-1}$, and optimize Q instead of Σ. Useful fact: $\frac{\partial \log \det(\Sigma)}{\partial \Sigma} = \Sigma^{-1}$).

Problem 5: Bayes Classifiers

In order to reduce my email load, I decide to implement a machine learning algorithm to decide whether or not I should read an email, or simply file it away instead. To train my model, I obtain the following data set of binary-valued features about each email, including whether I know the author or not, whether the email is long or short, and whether it has any of several key words, along with my final decision about whether to read it ($y = +1$ for "read", $y = -1$ for "discard").

x_1 know author?	x_2 is long?	x_3 has 'research'	x_4 has 'grade'	x_5 has 'lottery'	y $\Rightarrow$ read?
0	0	1	1	0	-1
1	1	0	1	0	-1
0	1	1	1	1	-1
1	1	1	1	0	-1
0	1	0	0	0	-1
1	0	1	1	1	1
0	0	1	0	0	1
1	0	0	0	0	1
1	0	1	1	0	1
1	1	1	1	1	-1

In the case of any ties, we will prefer to predict class +1.

I decide to try a Bayes classifier to make my decisions and compute my uncertainty.

(a) [**5 points**] Compute all the probabilities necessary for a naïve Bayes classifier, i.e., the class

probability $p(y)$ and all the individual feature probabilities $p(x_i|y)$, for each class y and feature x_i

(b) **[2 points]** Which class would be predicted for $\underline{x} = (0\ 0\ 0\ 0\ 0)$? What about for $\underline{x} = (1\ 1\ 0\ 1\ 0)$?

(c) **[2 points]** Compute the posterior probability that $y = +1$ given the observation $\underline{x} = (1\ 1\ 0\ 1\ 0)$.

(d) **[2 points]** Why should we probably not use a Bayes classifier (using the joint probability of the features x, as opposed to a naïve Bayes classifier) for these data?

Problem 6: Gaussian Bayes Classifiers

Now, using the Iris data, we will explore a classifier based on Gaussian distribution and Bayes rule. We'll use only the first two features of Iris (that is, $X(:, 1:2)$ in MATLAB), and split in to training and test sets as Problem 2.

(a) **[2 points]** Estimate the prior $\hat{p}(y)$ of the class labels.

(b) **[2 points]** Splitting your training data by class (the value of y), compute the empirical mean vector $\hat{\mu}^{(k)}$ and covariance matrix $\hat{\Sigma}^{(k)}$ of the data in each class $y = k$. (You can use `mean` and `cov` and `find` in MATLAB for this)

(c) **[5 points]** Plot a scatterplot of the data, coloring each data point by its class. Use your code in Problem 4 to plot contours on your scatterplot for each class, i.e., plot a Gaussian contour for each class using its empirical parameters, in the same color you used for those data points.

(d) **[5 points]** The Gaussian Bayesian classifier predicts the label y of a given x by maximizing the posterior probability $p(y|x)$, that is,

$$\hat{y}(x) = \arg\max_k p(y = k|x),$$

where "arg max" (the argument of the maximum) represents the point that maximizes the given function. Please estimate $p(y|x)$ using $\hat{\mu}^{(k)}$, $\hat{\Sigma}^{(k)}$ and $\hat{p}(y)$ calculated above (using Bayesian rule). This problem involves Implementing this classifier in MATLAB, and use it compute the empirical error rate (number of misclassified points) on the training and test data. Please complete the code in `GaussianBayesPredict.m`, so that it works in the following code:

```
Ytest_hat = GaussianBayesPredict.m(Xtest(:,1:2), Xtrain(:,1:2), Ytrain);
Error_test = mean(Ytest_hat ~= Ytest);
% Calculate testing error;

Ytrain_hat = GaussianBayesPredict.m(Xtrain(:,1:2), Xtrain(:,1:2), Ytrain);
Error_train = mean(Ytrain_hat ~= Ytrain);
% Calculate training error
```

Please report the value of `Error_test` and `Error_test` you obtained.