CS273a Homework #1 Solution

Introduction to Machine Learning: Winter 2015

Problem 0: Getting connected

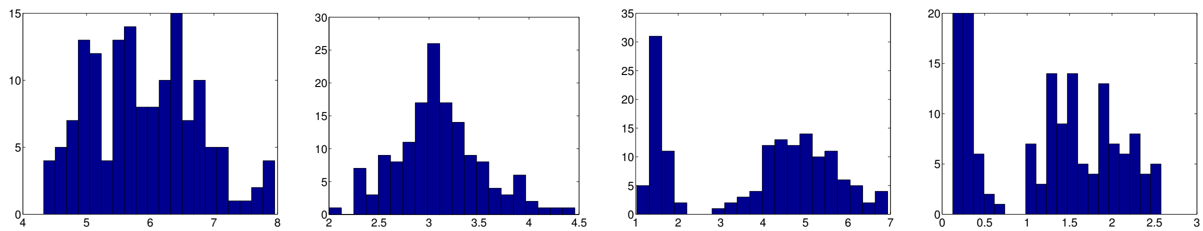
Hopefully you did this.

Problem 1: Data Exploration

```
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
```

```
% 1(a) : Use "size":
 size(X),
 % ans =
 %    148     4
 % => 148 data points, in 4 dimensions
```

```
% 1(b) : For each feature plot a histogram of the data values
% See "hist" function for more information
for i=1:4,
  figure(i);
  hist(X(:,i), 20);
end;
```



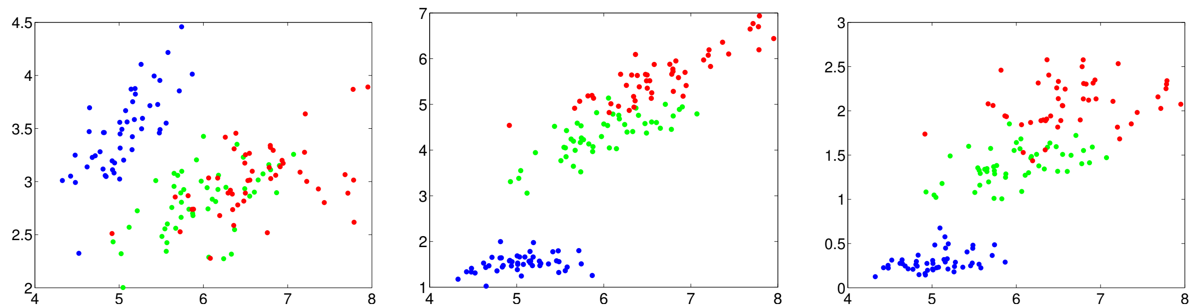
Feature 1 Feature 2 Feature 3 Feature 4

```
% 1(c),(d) : For each feature, compute the mean, variance, std-dev of the data values
% See built-in "mean" and "var" functions to understand how they operate:
mean(X)
%ans =
%    5.9001    3.0989    3.8196    1.2526
var(X)
%ans =
%    0.6993    0.1916    3.0976    0.5797
std(X)
%ans =
%    0.8362    0.4378    1.7600    0.7613
```

```
% 1(e) : normalize the data
Xn = X - repmat(mean(X),[148,1]);
% repmat "tiles" a matrix, so the 1x4 mean vector will be repeated to make it
%  the same size as our [148 x 4] data matrix.  Similarly,
```

```
Xn = Xn ./ repmat(std(X), [148,1]);
% if you check, Xn will be zero mean, unit variance
```

```
% 1(f) : For each feature pair (1,2),(1,3),(1,4) scatterplot the data values
% (I did this with the unnormalized data, X; Xn will look only slightly different)
% See "find", "plot" and "hold" functions for more information
i=1; for j=2:4,
  figure(j);
  ids=find(y==0); plot(X(ids,i),X(ids,j),'b.','markersize',20); hold on;
  ids=find(y==1); plot(X(ids,i),X(ids,j),'g.','markersize',20);
  ids=find(y==2); plot(X(ids,i),X(ids,j),'r.','markersize',20);
end;
```



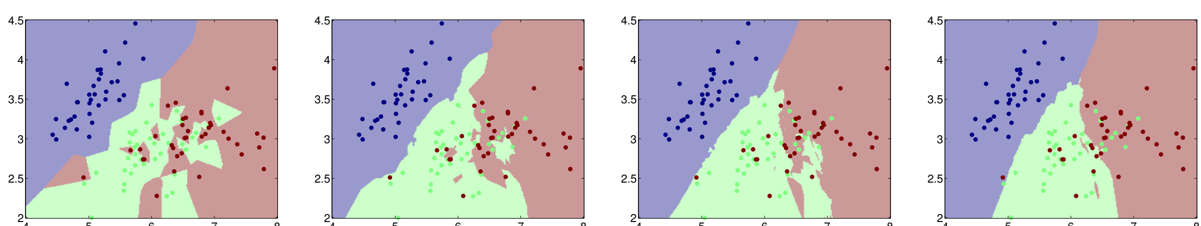
Features (1,2) Features (1,3) Features (1,4)

Problem 2: kNN predictions

```
% Start by loading the data, reordering it, and splitting it into training and validation:
iris=load('data/iris.txt'); y=iris(:,end); X=iris(:,1:end-1);
[X y] = shuffleData(X,y);
[Xtr Xva Ytr Yva] = splitData(X,y, .75); % split data into 75/25 train/test
```

Now, let's plot the k nearest neighbor classification boundary using the first two features:

```
for k=[1 5 10 20]
  knn = knnClassify( Xtr(:,1:2),Ytr, k);
  plotClassify2D( knn, Xtr(:,1:2), Ytr);  % plot data and decision boundary
  fname = sprintf('hw1_4a_%d.eps',k);
  set(gca,'fontsize',20);
  print(fname,'-depsc2');  system(['epstopdf ' fname]); system(['rm ' fname]);
end;
```

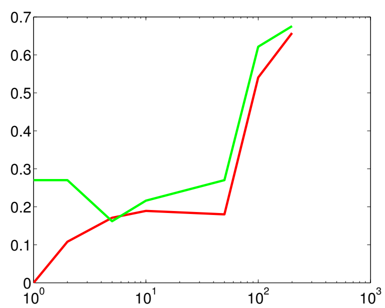


$k = 1$ $k = 5$ $k = 10$ $k = 20$

Figure 1: Classiification boundaries at various values of k.

Now, let's compute the error rates:

```
K=[1,2,5,10,50,100,200];
for k=1:length(K)
  learner = knnClassify( Xtr(:,1:2),Ytr, K(k) );
  Yhat = predict( learner, Xtr(:,1:2) );
  etrain(k) = mean( Yhat ~= Ytr );
  Yhat = predict( learner, Xva(:,1:2) );
  evalid(k) = mean( Yhat ~= Yva );
end;
figure; semilogx(K,etrain,'r-',K,evalid,'g-','linewidth',3);
set(gca,'fontsize',20);
print -depsc2 hw1_4b.eps;
!epstopdf hw1_4b.eps
!rm hw1_4b.eps
```



Based on this plot, $k = 5$ has the lowest validation error, so I would most likely choose that. You can also see evidence of overfitting ($k = 1$ and 2; low training error but high validation error) and of underfitting ($k = 100$ or more; similar, high training and validation errors).

Problem 3: Bayes Classifiers

(a) You can most easily do this by hand, but since I have to type it I will put it in Matlab format:

```
p_y = 4/10;   % p(y) = 4/10

%p(xi | y=-1)
p_x1_y0 = 3/6; p_x2_y0 = 5/6;
p_x3_y0 = 4/6; p_x4_y0 = 5/6; p_x5_y0 = 2/6;

%p(xi | y=+1)
p_x1_y1 = 3/4; p_x2_y1 = 0/4;
p_x3_y1 = 3/4; p_x4_y1 = 2/4; p_x5_y1 = 1/4;
```

(b)

```
f_y1_00000 = p_y*(1-p_x1_y1)*(1-p_x2_y1)*(1-p_x3_y1)*(1-p_x4_y1)*(1-p_x5_y1),
         % = 0.0094
f_y0_00000 = (1-p_y)*(1-p_x1_y0)*(1-p_x2_y0)*(1-p_x3_y0)*(1-p_x4_y0)*(1-p_x5_y0),
         % = 0.0019
% => Predict Class +1

f_y1_11010 = p_y*(p_x1_y1)*(p_x2_y1)*(1-p_x3_y1)*(p_x4_y1)*(1-p_x5_y1),
         % = 0
f_y0_11010 = (1-p_y)*(p_x1_y0)*(p_x2_y0)*(1-p_x3_y0)*(p_x4_y0)*(1-p_x5_y0),
```

```
        % = 0.0463
% => Predict Class -1
```

(c)

```
% p(y1|11010) =
    f_y1_11010 / (f_y1_11010 + f_y0_11010),
%   = 0

% For the other pattern (not required), p(y1|00000) =
    f_y1_00000 / (f_y1_00000 + f_y0_00000),
%   = .8351
```

(d) A Bayes classifier using a joint distribution model for $p(x|y = c)$ would have $2^5 - 1 = 31$ degrees of freedom (independent probabilities) to estimate; here we have only 6 and 4 data points respectively. So such a model would be extremely unlikely to generalize well to new data.

Problem 4: Gaussian Bayes Classifiers

```
 rand('state',0); randn('state',0);
 iris=load('data/iris.txt'); y=iris(:,end); X=iris(:,1:2); % take only 2 features
 [X y] = shuffleData(X,y); % shuffle data randomly to avoid pathological orders
 [Xtr Xte Ytr Yte] = splitData(X,y, .75); % split data into 75/25 train/test

% (a) Split your data by class and compute the empirical mean and covariance:
 mu0 = mean( Xtr(Ytr==0,:) ), cov0 = cov( Xtr(Ytr==0,:) ),
 % [5.0674    3.5092]  and  [ 0.1409    0.1062 ; 0.1062    0.1361 ]
 mu1 = mean( Xtr(Ytr==1,:) ), cov1 = cov( Xtr(Ytr==1,:) ),
 % [5.9243    2.7871]  and  [ 0.2629    0.0913 ; 0.0913    0.1007 ]
 mu2 = mean( Xtr(Ytr==2,:) ), cov2 = cov( Xtr(Ytr==2,:) ),
 % [6.5840    2.9571]  and  [ 0.4014    0.0554 ; 0.0554    0.0900 ]

% (b) Plot a scatterplot of the data, colored by class and
% (c) Plot the Gaussian distributions on top

 figure; plotClassify2D([],Xtr,Ytr); hold on;
 plotGauss2D(mu0,cov0,'b','linewidth',3);
 plotGauss2D(mu1,cov1,'g','linewidth',3);
 plotGauss2D(mu2,cov2,'r','linewidth',3);

% (d) Build and visualize a Gaussian Bayes classifier:
 bc = gaussBayesClassify( Xtr,Ytr );
 figure; plotClassify2D(bc,Xtr,Ytr);
```

```
% (e) Compute the empirical error rates of training & validation data:
 % NOTE: and your error rates may be off (higher) as well
 err(bc,Xtr,Ytr),  % training empirical error
 % =  0.1892
 err(bc,Xte,Yte),  % validation/test empirical error
 % =  0.2432
```

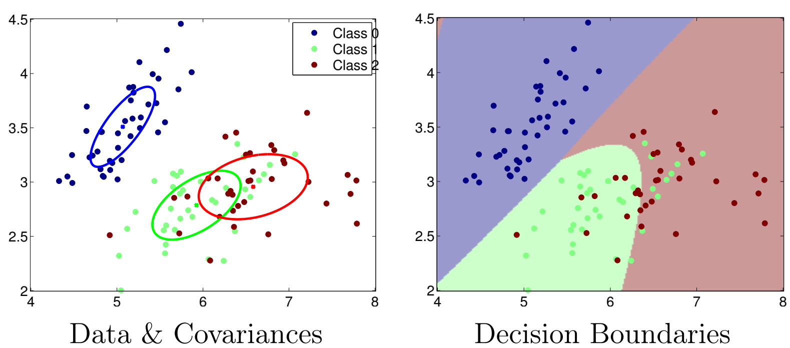


Figure 2: Data, Gaussian statistics, and decision boundaries for the Gaussian Bayes Classifier.