Pattern Recognition

Examples
Example 1 - Mahalanobis distance

- The Mahalanobis distance can be used to classify an unknown feature vector.
  - Assume you know the class centers (centroids) $z_i$, and their covariances $C_i$
  - We find the class that has the smallest distance from its center to the point in feature space

- The Mahalanobis distance of feature vector $x$ to the $i^{th}$ class is
  \[ d_i = \sqrt{(x - z_i)^T C_i^{-1} (x - z_i)} \]

- where $C_i$ is the covariance matrix of the feature vectors in the $i^{th}$ class
clear all
close all

% Load iris data. This loads in variables "meas" and "species".
load fisheriris

% Keep only the first two classes, which are 1..50 and 51..100.
X1 = meas(1:50,1:2); % Keep only 1st two measurement dimensions
X2 = meas(51:100,1:2); % Keep only 1st two measurement dimensions
plot(X1(:,1), X1(:,2), 'ro', X2(:,1), X2(:,2), 'go');

% Compute the mean of each class.
m1 = mean(X1);
m2 = mean(X2);
hold on
plot(m1(1), m1(2), 'r+');
plot(m2(1), m2(2), 'g+');

% Compute the covariance of each class.
C1 = cov(X1);
C2 = cov(X2);
disp('C1:'), disp(C1);
disp('C2:'), disp(C2);

% This bit of code displays the covariance as an ellipse.
th=0:2*pi/30:2*pi;
ptsCircle = [cos(th') sin(th')]; % Create points along a circle
n = size(ptsCircle,2);
p1 = chol(C1)'*ptsCircle + repmat(m1',1,n);
p2 = chol(C2)'*ptsCircle + repmat(m2',1,n);
plot(p1(1,:), p1(2,:), 'LineWidth', 2.0, 'Color', 'r');
plot(p2(1,:), p2(2,:), 'LineWidth', 2.0, 'Color', 'g');
• Let $X_{test} = [5.3, 3.1]$
• What is the Mahalanobis distance to class 1 (red) and class 2 (green)?
Xtest = [5.3, 3.1];
plot(Xtest(1), Xtest(2), 'b*');

% Find the Mahalanobis distance to each centroid
d1 = sqrt( (Xtest-m1) * inv(C1) * (Xtest-m1)' )
d2 = sqrt( (Xtest-m2) * inv(C2) * (Xtest-m2)' )
Example 2 – k-Nearest Neighbor Classifier

• Find the $k$ nearest neighbors to the query point, in the training data.

• Take the majority vote as the classification decision.

Example: using closest $k=5$ neighbors, majority vote is blue
Fisher Iris data

The 5 closest neighbors are classes:
class 1
class 1
class 1
class 1
class 1
Test point is classified as 1, true value is 1
clear all
close all

% Loads:
%   meas(150,4) - each row is a pattern (a 4-dimensional vector)
%   species{150} - each element is the name of a flower
load fisheriris

% Create a vector of class numbers. We know that the input data is grouped
% so that 1..50 is the 1st class, 51..100 is the 2nd class, 101..150 is the
% 3rd class.
y(1:50,1) = 1;    % class 'setosa'
y(51:100,1) = 2;    % class 'versico'
y(101:150,1) = 3;    % class 'virginica'

X = meas(:, 1:2);       % just use first 2 features (easier to visualize)

% We will just use the first 2 features, since it is easier to visualize.
% However, when we do that there is a chance that some points will be
% duplicated (since we are ignoring the other features). If so, just keep
% the first point.
indicesToKeep = true(size(X,1),1);
for i=1:size(X,1)
    % See if we already have the ith point.
    if any((X(i,1)==X(1:i-1,1)) & (X(i,2)==X(1:i-1,2)))
        indicesToKeep(i) = false;   % Skip this point
    end
end
X = X(indicesToKeep, :);
y = y(indicesToKeep);

% Let's take out a data point to use for testing. The rest we will
% use for training.
t = randi(size(X,1));       % Pick a point at random
xTest = X(t,:);
yTest = y(t);
% Remove that point from the training data. Note - if there are
% duplicates, remove those too.
indicesTraining = (X(:,1)~=xTest(1)) | (X(:,2)~=xTest(2));
X = X(indicesTraining, :);
y = y(indicesTraining);

% Plot the feature vectors.
figure
hold on
plot(X(y==1,1), X(y==1,2), '*r');
plot(X(y==2,1), X(y==2,2), '*g');
plot(X(y==3,1), X(y==3,2), '*b');
xlabel('Sepal length'), ylabel('Sepal width');

plot(xTest(1), xTest(2), 'ok');           % black is the test point
hold off

% Find distances from the test point to all other points.
distances = ((xTest(1)-X(:,1)).^2 + (xTest(2)-X(:,2)).^2).^0.5;

% Sort in ascending order, and save indices of sorted points.
[distSorted, indices] = sort(distances, 'ascend');

k = 5;
fprintf('The %d closest neighbors are classes:
', k);
hold on
for i=1:k
    fprintf(' class %d
', y(indices(i)));
    % Show point on the plot.
    xn = X(indices(i),:);
    plot(xn(1), xn(2), 'dk');
end
% Get majority vote (ie, the mode) among neighbors.
class = mode(y(indices(1:k)));
fprintf('Test point is classified as %d, true value is %d
', class, yTest);
Example 3 – Shape classification

• On the course web page are two binary images, “hearts.bmp” and “plus.bmp”. These are images of hand-drawn shapes of hearts and pluses, respectively.

• Extract connected components from these images, and compute feature properties for each image, using the Matlab function regionprops.
Example 3 (continued)

• Plot the feature “Area” vs “Solidity” for each of the two classes (using Matlab’s plot function). You will need to put those features into a 2xN array:

```matlab
for i=1:n1
    X(i,1) = props1(i).Area;
    X(i,2) = props1(i).Solidity;
    y(i,1) = 1;     % class label
end
for i=1:n2
    X(i+n1,1) = props2(i).Area;
    X(i+n1,2) = props2(i).Solidity;
    y(i+n1,1) = 2;  % class label
end
```
Example 3 (continued)

• Create a decision tree for these two classes, using the Matlab function `ClassificationTree.fit`.
  - Make sure that you set the input parameter “MinParent” to 1 (the default is 10).
  - Setting MinParent = 1 will cause the decision tree to split (make a new node) if there are any instances that are still not correctly labeled.

• Draw the resulting decision tree (using the function `view`):
  ```matlab
  view(ctree, 'mode', 'graph');
  ```

• Apply the tree to classify the new shape in the image “test.bmp”. Which class do you assign this shape to?