In the Name of God
Project-1
LEAST SQUARES LINEAR DISCRIMINANTS
Department of Electrical Engineering - Shiraz University

Instructor: Dr. Hamed Masnadi-Shirazi　　　　Spring 2015

1 Introduction

The object of this project is to build a linear classifier using the least squares criteria. A linear classification function can be written as $g(\mathbf{x}) = \mathbf{w^T x} + w_0$ where $\mathbf{x}$ is a sample data point, $\mathbf{w}$ is the weight vector or the normal of the separating hyperplane and w_0 is the threshold or bias. Given the explanation provided in class, the data point $\mathbf{x}$ is classified as belonging to the positive class if $g(\mathbf{x}) > 0$ and is said to belong to the negative class if $g(\mathbf{x}) < 0$. We know that this corresponds to the data point being on one side or the other of the hyperplane. We also saw that $g(\mathbf{x})$ gives an algebraic measure of the distance from point $\mathbf{x}$ to the hyperplane, specifically we have shown that $g(\mathbf{x}) = r||w||$ where r is the distance from $\mathbf{x}$ to the hyperplane.

Assuming that we have reason to believe that all the data points can be correctly classified by a hyperplane, then $\mathbf{w}$ and w_0 could be found such that $g(\mathbf{x}) > 0$ for all $\mathbf{x}$ in the positive class and $g(\mathbf{x}) < 0$ for all $\mathbf{x}$ in the negative class. We can simplify the problem and replace inequalities with equalities if we try to find $\mathbf{w}$ and w_0 such that $g(\mathbf{x}) = b$ for all $\mathbf{x}$ in the positive class where b is an arbitrary positive number, and $g(\mathbf{x}) = b$ for all $\mathbf{x}$ in the negative class where b is an arbitrary negative number. Assuming that we have n_1 positive data points of dimension d and n_2 negative data points of dimension d, the problem of finding $\mathbf{w}$ and w_0 can be written as a linear inverse problem in the form of

$$\begin{bmatrix} 1 & \mathrm{x}_1^{(1)} & .... & \mathrm{x}_d^{(1)} \\ . & . & .... & . \\ . & . & .... & . \\ . & . & .... & . \\ 1 & \mathrm{x}_1^{(n_1)} & .... & \mathrm{x}_d^{(n_1)} \\ 1 & \mathrm{x}_1^{(n_1+1)} & .... & \mathrm{x}_d^{(n_1+1)} \\ . & . & .... & . \\ . & . & .... & . \\ . & . & .... & . \\ 1 & \mathrm{x}_1^{(n_1+n_2)} & .... & \mathrm{x}_d^{(n_1+n_2)} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ . \\ . \\ . \\ w_d \end{bmatrix} = \begin{bmatrix} b^{(1)} \\ . \\ . \\ . \\ b^{(n_1)} \\ b^{(n_1+1)} \\ . \\ . \\ . \\ b^{(n_1+n_2)} \end{bmatrix}$$

or

$$\mathbf{Xw} = \mathbf{b} \tag{1}$$

where $b^1, ..., b^{n_1} > 0$ and $b^{n_1+1}, ..., b^{n_1+n_2} < 0$.

If $n_1 + n_2 >> d$ then it is safe to assume that the tall matrix $\mathbf{X}$ is full column rank and the unique minimum norm least squares solution is

$$\mathbf{w} = (\mathbf{X^T X})^{-1}\mathbf{X^T b}. \tag{2}$$

It can be shown that if the vector $\mathbf{b}$ is chosen as

$$\begin{bmatrix} \frac{n_1+n_2}{n_1} \\ \cdot \\ \cdot \\ \cdot \\ \frac{n_1+n_2}{n_1} \\ \frac{n_1+n_2}{n_2} \\ \cdot \\ \cdot \\ \cdot \\ \frac{n_1+n_2}{n_2} \end{bmatrix} \tag{3}$$

then the minimum norm least squares solution for this special choice of $\mathbf{b}$ is equivalent to Fisher's linear discriminant.

2 General Assignment

In this project you will be given sets of positive and negative data (training data) and you are to build a linear classifier for the data by solving the minimum norm least squares problem for different choices of the $\mathbf{b}$ vector. The classifier that you have trained will then be tested or evaluated on the test data sets. The true test data labels will be provided from which you will report test error rates for each data set.

3 Matlab details

STEP 1. We will first consider a simple example. Consider two dimensional data points $(1, 2)$ and $(2, 0)$ for the positive class and two dimensional data points $(3, 1)$ and $(2, 3)$ for the negative class. Plot these four points using different colors for each class. The least squares problem can be setup as

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & 2 & 0 \\ 1 & 3 & 1 \\ 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} \frac{4}{2} \\ \frac{4}{2} \\ -\frac{4}{2} \\ -\frac{4}{2} \end{bmatrix}$$

Solve this problem for $\mathbf{w}$ using the Matlab command

```
w=X\b;
```

Now, plot the line defined by the vector $\mathbf{w}$ in the same plot where you plotted the data points using the Matlab commands below

```
hold on,
x=1:0.1:3;
y=(w(1)+w(2)*x)./(-w(3));
plot(x,y)
```

How many of the data points were classified correctly? Now write Matlab code that will check all four data points to see if they are classified correctly. Explain your code in your report. Repeat the above steps for when you set $\mathbf{b} = [1 \;\; 1 \;\; -1 \;\; -1]^T$. Do the results change?

STEP 2. Now that you are sure your code works correctly, you can try your code on some real data. Use the Matlab code below to load and plot the Iris flower data set where we are using the length and width of the flowers to classify two different species of Iris flowers. We call this the training data because we use it to train or learn the classifier.

```
load fisheriris
xdata = meas(51:end,3:4);
N=100;
figure,
plot(xdata(1:50,1),xdata(1:50,2),'b*')  % the first 50 are class=1 and the last 50 are class=2
hold on
plot(xdata(51:end,1),xdata(51:end,2),'ro')
```

Now, setup the least squares problem and solve it like you did in STEP.1 where you choose $\mathbf{b}$ from equation (3). Plot the line defined by $\mathbf{w}$ within the data plot. Use your Matlab code to report the number of errors. The accuracy of your classifier is defined as

$$Accuracy = (1 - \frac{\#Errors}{\#DataPoints}) \times 100. \tag{4}$$

What is the accuracy of your classifier on the training data?

STEP 3. In this step we want to study the effect of adding more data to our training set. We suspect that adding more training data will generally improve our classifier. To do this we need a way to simulate as many data samples as we want. We will make our data sets by randomly sampling them from two different Gaussian distributions. The Matlab code below will generate 1000 negative points sampled from a Gaussian with mean $mu0$ and covariance $Sigma$ and 1000 positive points sampled from a Gaussian with mean $mu1$ and covariance $Sigma$. It will then plot these data points.

```
TrainSetSize=1000;
mu0 = [0.7416 0.7416];  % negative class mean
Sigma = [1 0; 0 1]; % negative class Covariance
R = chol(Sigma);
NTrain = repmat(mu0,TrainSetSize,1) + randn(TrainSetSize,2)*R; % negative set
mu1 = [0 0];                % positive class mean
Sigma = [1 0; 0 1];  % positive class Covariance
R = chol(Sigma);
PTrain = repmat(mu1,TrainSetSize,1) + randn(TrainSetSize,2)*R; % positive set
%%%%plot
figure,
plot(NTrain(:,1),NTrain(:,2),'ro')
hold on
plot(PTrain(:,1),PTrain(:,2),'b*')
```

It can be shown that the ideal classifier for this data is a line and that the best possible accuracy is 70%. Setup the least squares problem and solve it for $\mathbf{w}$. Plot the line defined by $\mathbf{w}$ on the train data plot. Now make a new test set to test the accuracy on the random test points. Use the code above and generate $10,00$ (ten thausand) random test data points. What is the accuracy of your classifier on the ten thousand random test points?

Repeat the whole process of making the random train data, solving the least squares problem and finding the accuracy on ten thousand random test points, five times. Report the accuracy for each time. Also report the average accuracy.

Now repeat the whole process five times but with $TrainSetSize = 5$;. Report the accuracy for each time. Also report the average accuracy. Does using more training data lead to better accuracy?

STEP 4. We have only dealt with two dimensional data so far. Using data with more dimensions is no different. We will build a classifier for breast cancer tumor diagnostics. The data consists of different measurements of a breast tumor. We will build a classifier to decide if the tumor is cancerous or not. Use the Matalb code below to load the data.

```
load(['C:\BREASTDIAG_UCI_5foldValidationData_NEWFIXEDcorrupt1.mat']);
size(RareTrain1)
size(MajorTrain1)
size(RareTest1)
size(MajorTest1)
```

The variable RareTrain1 should be used as the positive training set, the variable MajorTrain1 should be used as the negative training set, the variable RareTest1 should be used as the positive test set and the variable MajorTest1 should be used as the negative test set. What is the dimensions of the data? Setup the least squares problem like before and report the accuracy on the test data.

STEP 5. An $m \times n$ pixel image can be thought of as a vector of dimensions $m \times n$. We will use this to classify images of hand written digits. Use the Matlab code below to load the training and testing data for each digit. For example the code below will load the data corresponding to the digit 1 and show the first sample.

```
Current=1; % the digit number to load
%%%% Load the Current train set
A=open(['C:\digit' num2str(Current) '.mat']);
names = fieldnames(A);
size(A.D);
i=1;
I=A.D(i,:);
I=reshape(I,28,28);
I=I';
imshow(I,[])
PTrain=A.D;
```

The Matlab code below will load the test data corresponding to digit 1 and show the first sample in this set.

```
Current=1; % the digit number to load
%%%% Load the Current test set
A=open(['C:\test' num2str(Current) '.mat']);
names = fieldnames(A);
size(A.D);
i=1;
I=A.D(i,:);
I=reshape(I,28,28);
I=I';
imshow(I,[])
PTrain=A.D;
```

Explain what the code does in your report. What are the dimensions of the data and the images?

We want to see how well we can classify the digit 1 compared to the other digits $0, 2, ..., 8, 9$. Setup and solve the least squares problem for each digit pair: digit 1 compared to digit 0, digit 1 compared to digit 2, etc. Report the accuracy on each pair. Which pairs have the smallest accuracy? Which pairs have the best accuracy? For each pair, include in your report the images of at least two samples that are not 1 but have been incorrectly classified as 1 and the images of at least two samples that are 1 but have been incorrectly classified. What can you say about these images and why they were incorrectly classified?

4 What to turn in

You should turn in a CD which includes (1) A report on your project (2) and a Matlab .m file of your program. Your report should include your results for each STEP. You should clearly divide your report into STEPs. Failing to do so will get you a zero grade! You Matlab program should have a comment for each line and each segment of code explaining what that line or segment of code does. If you fail to comment your code, you will get a zero grade! I should be able to run your code directly from the CD and see the plots and results. If your code has errors when I run it or does not work when I run it from the CD, you will get a zero grade!

5 CHEATING :-(

Copying a portion of someones report or Matlab code is considered cheating. If even a portion of your report or Matlab code is similar to someone else you will both get a zero on the project, be introduced to the university disciplinary committee (Komite Enzebaty) and possibly fail the course. SO DO NOT EVEN SHOW YOUR CODE OR REPORT TO ANYONE ELSE!