STAT 161/261: Homework 2 Solutions
Bayes Decision Theory
Due Monday, April 18 in class or through CCLE

1. See MATLAB published files `docDatProc1.pdf` and `docDatProc2.pdf`.

    (a) Loading data – See MATLAB published file `docDatProc1.pdf`.

    (b) Histogram – See MATLAB published file `docDatProc1.pdf`.

    (c) Gaussian density – See MATLAB published file `docDatProc1.pdf`. We see that the Gaussian density surface looks similar in shape to the histogram. Later in the class, we will discuss formal measures for goodness of fit. But, for now, we can at least see that they qualitatively match.

    (d) See MATLAB published file `docDatProc2.pdf`. In this case, from the scatter plot and histogram, we see there is a clear second cluster of patients. This is not captured by the Gaussian distribution. This second cluster is possibly coming from the case where the second data has included teenagers along with infants.

    (e) For the second doctor, we could try to fit a mixture of two Gaussians. We will discuss how to do this mixture fitting later in the class.

2. See MATLAB published file `house.pdf`.

    (a) Plot data – see file

    (b) Linear fit – see file.

    (c) There is one outlier with a very low cost. We will see later in the class how to automatically identify these outliers.

    (d) Correlation – see file.

3. See MATLAB published file `expclass.pdf`.

    (a) The MAP classifier is

    $$\widehat{y} = \arg\max_{y=i} p(x|y=i)P(y=i).$$

    Since $P(y=0) = P(y=1)$, this is identical to the ML classifier.

    $$\widehat{y} = \arg\max_{y=i} p(x|y=i).$$

Hence, the classifier will select $\widehat{y} = 1$ when

$$\begin{aligned}
\widehat{y} = 1 &\iff p(x|y=1) \geq p(x|y=0) \iff \ln p(x|y=1) \geq \ln p(x|y=0) \\
&\iff -\ln(\lambda_1) - \frac{x}{\lambda_1} \geq -\ln(\lambda_0) - \frac{x}{\lambda_0} \\
&\iff x \geq t = \left[\frac{1}{\lambda_0} - \frac{1}{\lambda_1}\right]^{-1} \ln\left[\frac{\lambda_1}{\lambda_0}\right] = \frac{\lambda_1\lambda_0}{\lambda_1 - \lambda_0} \ln\left[\frac{\lambda_1}{\lambda_0}\right]
\end{aligned}$$

See the MATLAB published file for the generation of the data and running of the classifier.

(b) To derive the MLE for the parameters, first we write the data as

$$(\mathbf{x}, \mathbf{y}) = \{(x_i, y_i), i = 1, \ldots, N\},$$

which is the set of all the training samples. The unknown parameters are

$$\theta = (q_0, q_1, \lambda_0, \lambda_1),$$

where $q_j = P(y_i = j)$ is the class probability and λ_j is the mean of the exponential in each class. Now, we have the likelihood function is

$$p(\mathbf{x}, \mathbf{y}|\theta) \overset{(a)}{=} \prod_{i=1}^{N} p(x_i, y_i|\theta) \overset{(b)}{=} \prod_{i=1}^{N} p(x_i|y_i, \theta) P(y_i|\theta),$$

where (a) follows from the assumption that, given the parameters θ and the samples are independent; (b) is the conditional probability rule. So, the negative log likelihood is given by

$$L(\theta) = -\ln p(\mathbf{x}, \mathbf{y}|\theta) = -\sum_{i=1}^{N} \left[\ln p(x_i|y_i, \theta) + \ln P(y_i|\theta)\right] \tag{1}$$

We now do the following trick: Define I_0 and I_1 as

$$I_j = \{i \ : \ y_i = j\},$$

which are the samples on which $y_i = j$. We can then rewrite the sum in (1) as a double sum

$$L(\theta) = -\sum_{j=0,1} \sum_{i \in I_j} \left[\ln p(x_i|y_i = j, \theta) + \ln P(y_i = j|\theta)\right],$$

so that we first sum over the classes $j = 0, 1$ and then the samples $i \in I_j$. Next, observe that when $y_i = j$,

$$p(x_i|y_i = j, \theta) = \frac{1}{\lambda_j} e^{-x_i/\lambda_j}, \quad P(y_i = j) = q_j.$$

Hence,

$$\begin{aligned}
L(\theta) &= \sum_{j=0,1} \sum_{i \in I_j} \left[\ln(\lambda_j) + \frac{x_i}{\lambda_j} + \ln q_j\right] \\
&= \sum_{j=0,1} \left[N_j \ln(\lambda_i) + \frac{1}{\lambda_j} \sum_{i \in I_j} x_j + \ln(q_j)\right].
\end{aligned}$$

To find the MLE for λ_j, we take the derivative

$$\frac{\partial L(\theta)}{\partial \lambda_j} = 0 \Rightarrow \frac{N_j}{\lambda_j} - \frac{1}{\lambda_j^2}\sum_{i \in I_j} x_j = 0 \Rightarrow \lambda_j = \frac{1}{N_j}\sum_{i \in I_j} x_j.$$

Hence the MLE for λ_j is the sample mean of x_i within the samples where $y_i = j$.
For the MLE of q_j, first observe that

$$L(\theta) = N_0 \ln(q_0) + N_1 \ln(q_1) + \text{other terms},$$

where the other terms do not depend on the q_j. Now, since $q_0 + q_1 = 1$, we have $q_1 = 1 - q_0$ and hence

$$L(\theta) = N_0 \ln(q_0) + N_1 \ln(1 - q_0) + \text{other terms}.$$

Taking the derivative,

$$\frac{\partial L(\theta)}{\partial q_j} = 0 \Rightarrow \frac{N_0}{q_0} = \frac{N_1}{1 - q_0} \Rightarrow q_0 = \frac{N_0}{N_0 + N_1} = \frac{N_0}{N},$$

so the MLE of q_0 is simply the fraction of times that $y_i = 0$. The MATLAB to compute these are in the published MATLAB file.

4. (a) The linear model in this case is

$$\mathbf{y} = \mathbf{X}\beta_0, \quad \mathbf{X} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}.$$

The solution is

$$\widehat{\beta}_0 = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Now, since $\mathbf{X}$ is a vector of ones,

$$\mathbf{X}^T\mathbf{X} = N, \quad \mathbf{X}^T\mathbf{y} = \sum_{n=1}^{N} y_n \Rightarrow \widehat{\beta}_0 = \frac{1}{N}\sum_{n=1}^{N} y_n,$$

which is the sample mean.

(b) For a linear model of the form $\mathbf{y} = \mathbf{X}\beta$, the regression solution is

$$\widehat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

Now if $\mathbf{y} = \mathbf{X}\beta + \epsilon$,

$$\widehat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\beta + \epsilon) = \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon.$$

Therefore, using the fact that $\mathbb{E}(\epsilon) = 0$,

$$\mathbb{E}\left[\widehat{\beta}|\beta\right] = \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbb{E}(\epsilon) = \beta.$$

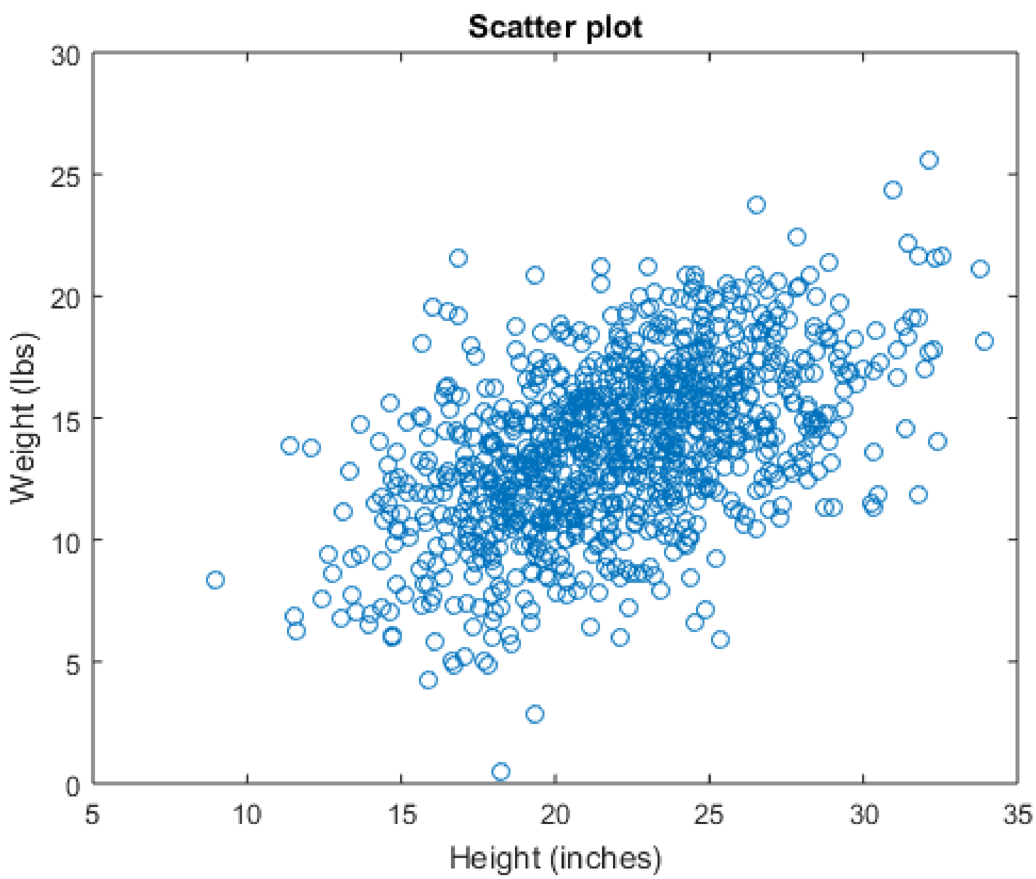
Hence, the estimate is unbiased.

docDatProc1.m: Processes doctor data

Contents

Load the data

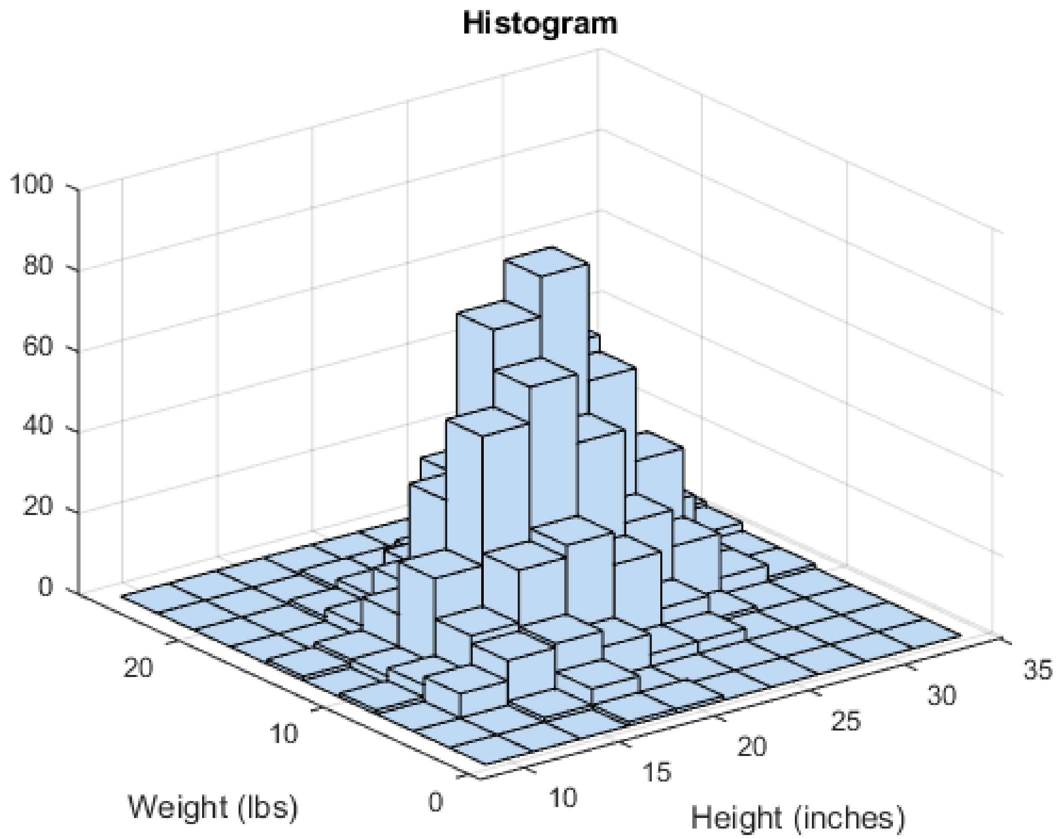
```
% Load the file using MATLAB's load command
dataSet = 1;    % 1 or 2 depending on which file to load
fn = sprintf('Doctor%ddata.txt', dataSet);
dat = load(fn);
n = size(dat,1);

% Create a scatter plot of the data.  This is not required, but gives a
% nice way to visualize the data
plot(dat(:,1),dat(:,2),'o');
title('Scatter plot');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```



Plot the histogram for the data

Plot the 2d histogram as a intensity map

```
nbins = 10;
hist3(dat,[nbins nbins]);
title('Histogram');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```

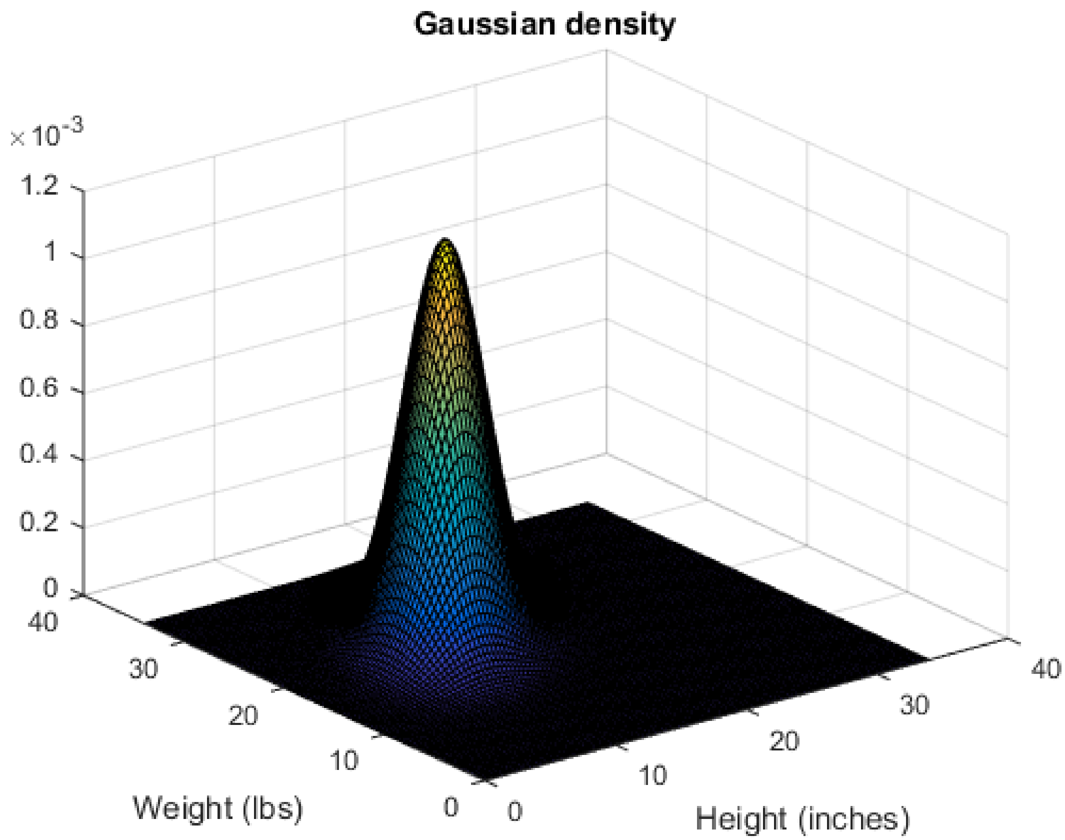


Estimate the Gaussian

```
% MLE of parameters
mhat = mean(dat)';                % Sample mean
dx = dat - repmat(mhat',n,1);
Phat = dx'*dx/n;                  % Covariance matrix
fprintf(1,'Mean height %12.4e weight %12.4e\n', mhat(1), mhat(2));

% Surface plot of density
npts = 100;             % Number of points in each axis
xp1 = linspace(0,max(dat(:,1)),npts)';  % Points to plot for x1
xp2 = linspace(0,max(dat(:,2)),npts)';  % Points to plot for x2

Qhat = inv(Phat);
phat = zeros(npts,npts);
for i1 = 1:npts
    for i2 = 1:npts
        xi =  [xp1(i1) xp1(i2)]'-mhat;
        phat(i1,i2) = exp(-0.5*xi'*Qhat*xi);
```

```
    end
end
phat = phat/(2*pi*det(Phat));
surf(xp1, xp1, phat);
title('Gaussian density');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```

```
Mean height   2.2070e+01 weight   1.3896e+01
```

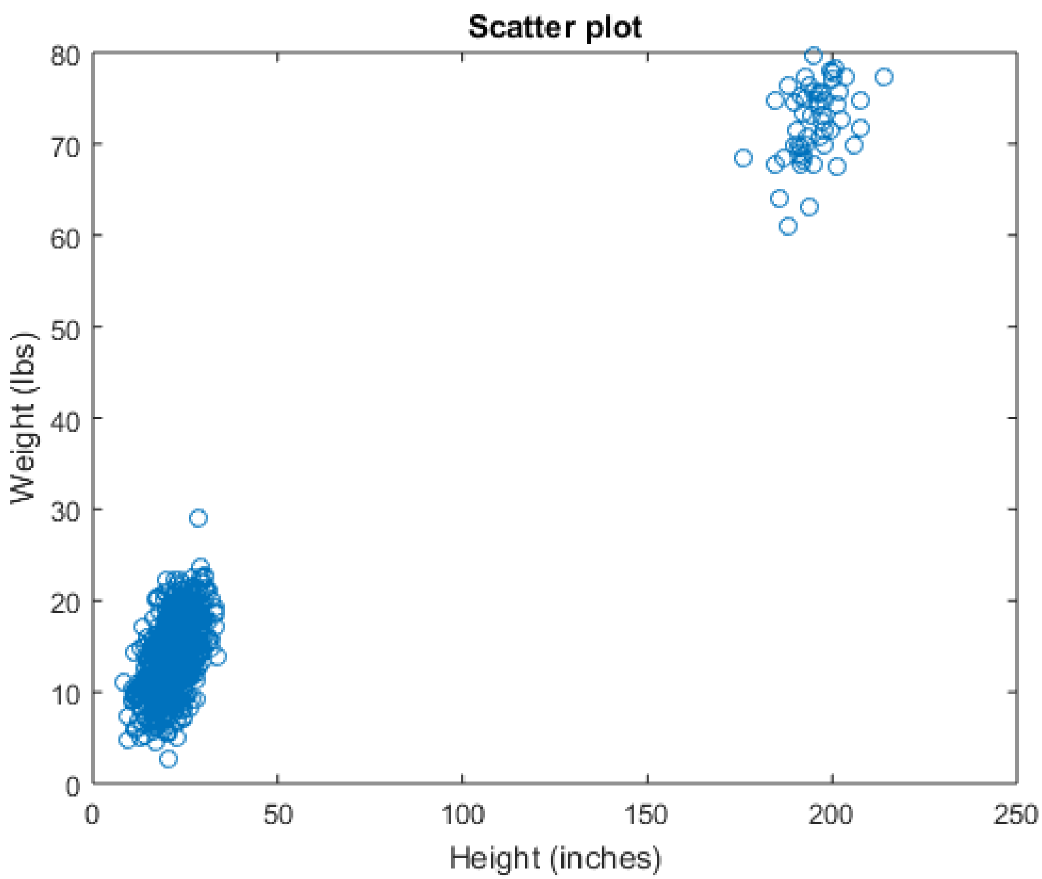


docDatProc2.m: Processes doctor data

Contents

Load the data

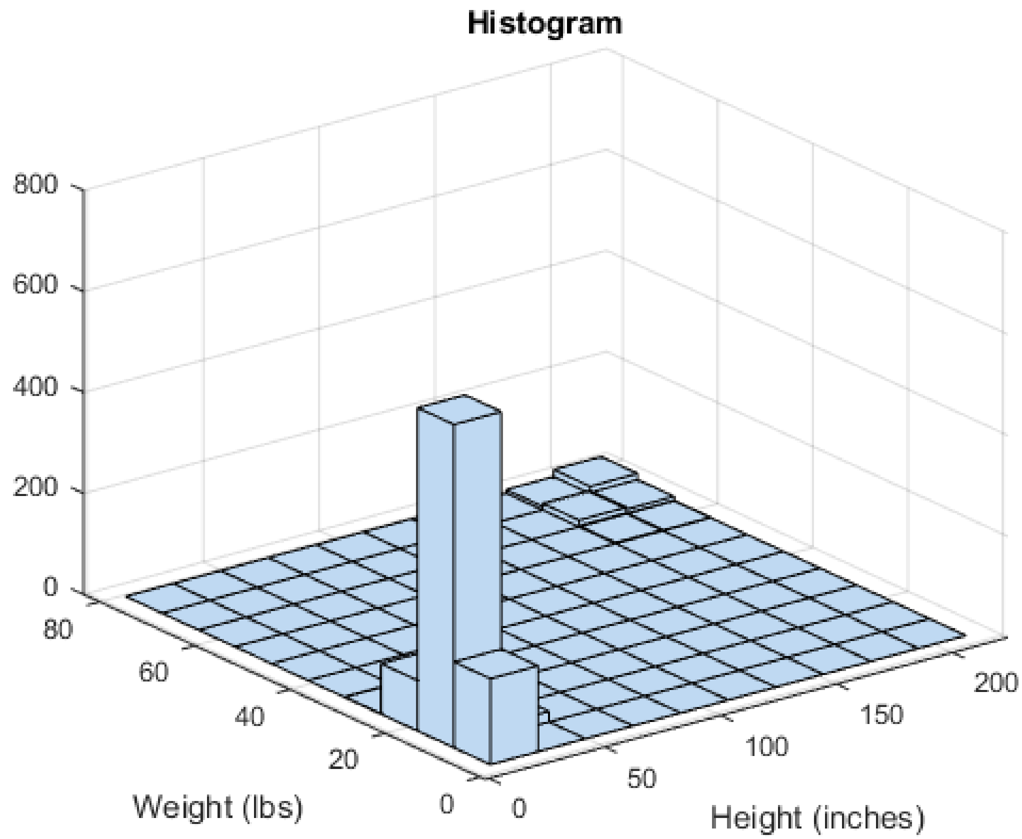
```
% Load the file using MATLAB's load command
dataSet = 2;    % 1 or 2 depending on which file to load
fn = sprintf('Doctor%ddata.txt', dataSet);
dat = load(fn);
n = size(dat,1);

% Create a scatter plot of the data.  This is not required, but gives a
% nice way to visualize the data
plot(dat(:,1),dat(:,2),'o');
title('Scatter plot');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```



Plot the histogram for the data

Plot the 2d histogram as a intensity map

```
nbins = 10;
hist3(dat,[nbins nbins]);
title('Histogram');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```

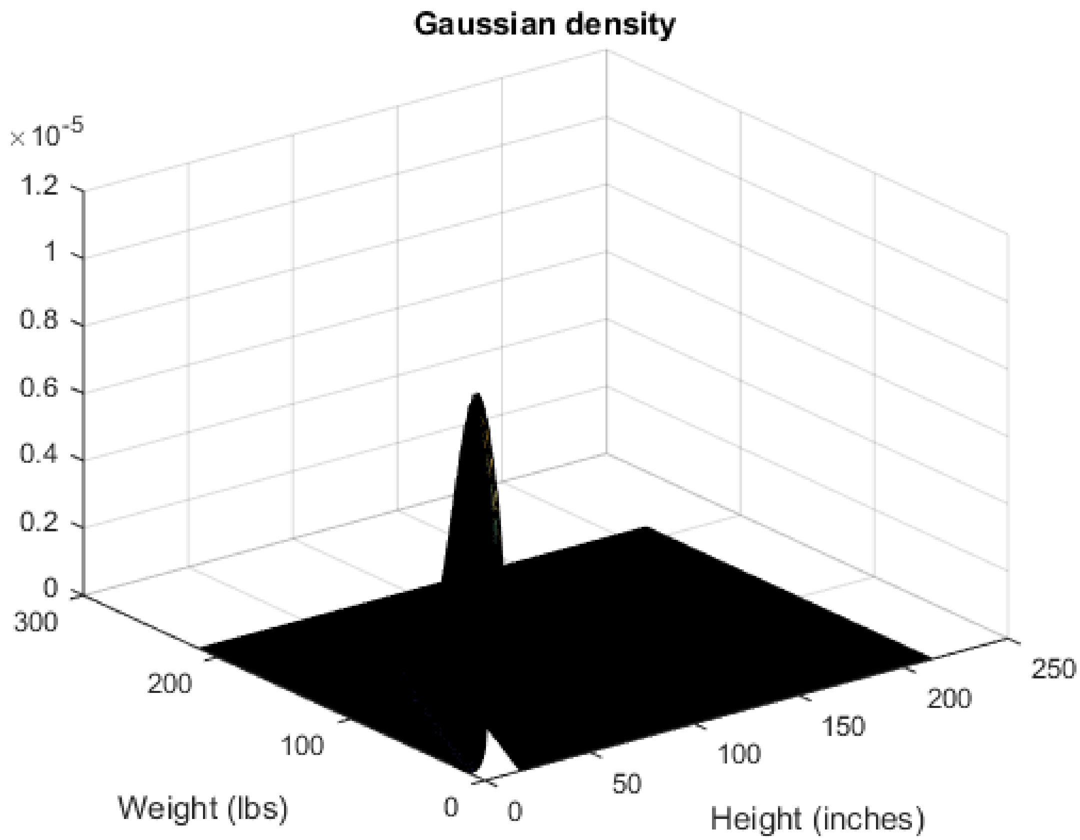


Estimate the Gaussian

```
% MLE of parameters
mhat = mean(dat)';                % Sample mean
dx = dat - repmat(mhat',n,1);
Phat = dx'*dx/n;                  % Covariance matrix
fprintf(1,'Mean height %12.4e weight %12.4e\n', mhat(1), mhat(2));

% Surface plot of density
npts = 100;             % Number of points in each axis
xp1 = linspace(0,max(dat(:,1)),npts)';  % Points to plot for x1
xp2 = linspace(0,max(dat(:,2)),npts)';  % Points to plot for x2

Qhat = inv(Phat);
phat = zeros(npts,npts);
for i1 = 1:npts
    for i2 = 1:npts
        xi =  [xp1(i1) xp1(i2)]'-mhat;
        phat(i1,i2) = exp(-0.5*xi'*Qhat*xi);
```

```
    end
end
phat = phat/(2*pi*det(Phat));
surf(xp1, xp1, phat);
title('Gaussian density');
xlabel('Height (inches)');
ylabel('Weight (lbs)');
```

```
Mean height   3.1058e+01 weight   1.7002e+01
```



Published with MATLAB® R2015a

expclass.m: Classifier example with exponential distributions

Contents

MAP classifier

Since the classes are equiprobable, the MAP = ML. The MAP selects yhat = 1 when ln p(x|y=1) >= ln p(x|y=0) ln(lam1) + x/lam1 <= ln(lam0) + x/lam0 x >= (1/lam0-1/lam1)^{-1}*ln(lam1/lam0)

```
% Parameters
lam1 = 10;
lam0 = 1;
p1 = 0.5;   % P(y=1)
t = lam1*lam0/(lam1-lam0)*log(lam1/lam0);

% Generate training data
ntr = 1000;                     % num samples
ytr = (rand(ntr,1) < p1);       % class label
lam = lam1*ytr + lam0*(1-ytr);  % lambda value
xtr = exprnd(lam,ntr,1);        % data

% Run classifier
yhat = (xtr > t);
perr0 = mean(yhat ~= ytr);
fprintf(1,'Error with opt classifier: %12.4e\n', perr0);
```

```
Error with opt classifier:   1.5800e-01
```

Classifier with learning

```
% MLE of lambda
lam0hat = sum(xtr.*(ytr==0))/sum(ytr==0);
lam1hat = sum(xtr.*(ytr==1))/sum(ytr==1);
t = lam1hat*lam0hat/(lam1hat-lam0hat)*log(lam1hat/lam0hat);

% Generate test data
nts = 1000;                     % num samples
yts = (rand(nts,1) < p1);       % class label
lam = lam1*yts + lam0*(1-yts);  % lambda value
xts = exprnd(lam,nts,1);        % data

% Run classifier on the test data
yhat = (xts > t);
perr1 = mean(yhat ~= yts);
fprintf(1,'Error with learned classifier: %12.4e\n', perr1);
```

```
Error with learned classifier:   1.6900e-01
```

The learned classifier does roughly as well as the optimal classifier. In fact, since the number of samples is not that large, the learned classiier may outperform the "optimal" classifier due to random variations.

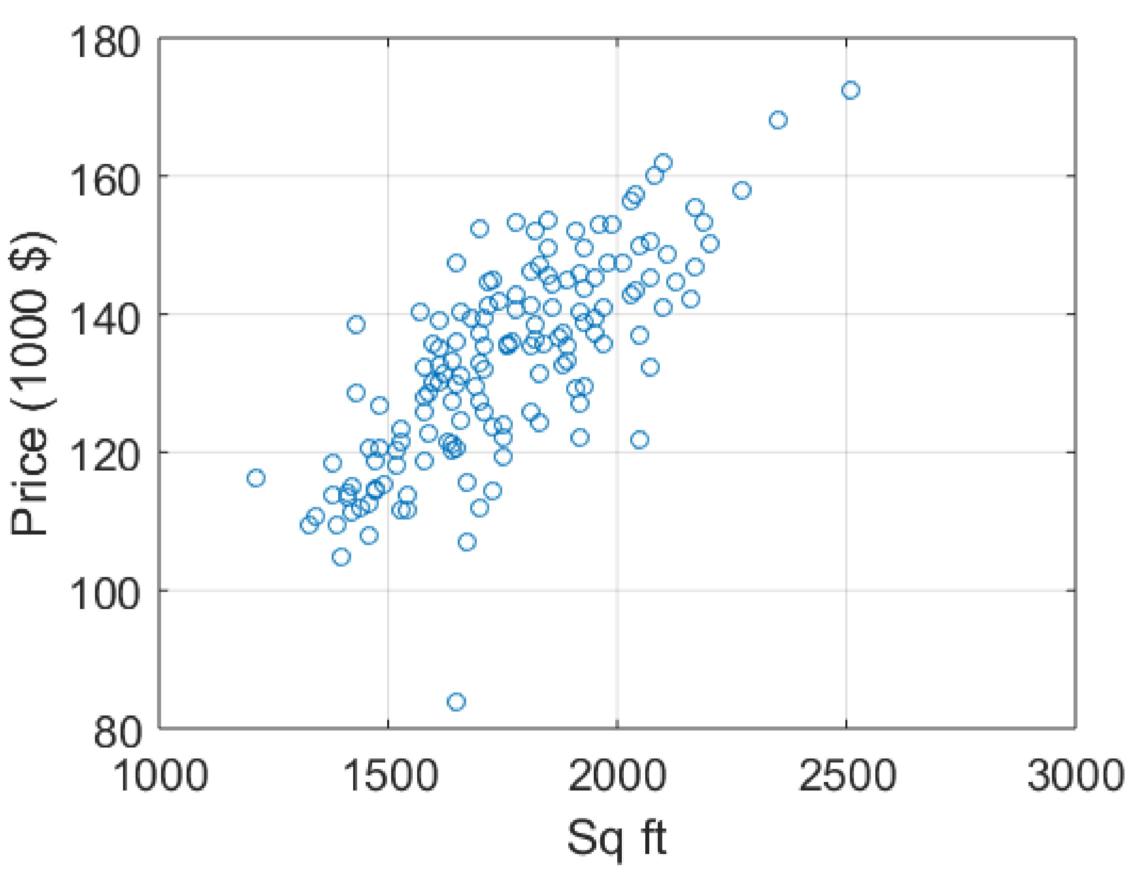
Published with MATLAB® R2015a

house.m: Housing data analysis

Contents

Load and plot the data.

```
% Load the data:  The original file has a header, which was stripped.
dat = load('housePrM_nohdr.txt');
price = dat(:,2);       % Price is in 1000's of dollars
sqft = dat(:,3)*100;    % Data in file is scaled by 100

% Plot the data
plot(sqft,price,'o');
grid on;
set(gca,'Fontsize',16);
xlabel('Sq ft');
ylabel('Price (1000 $)');
```

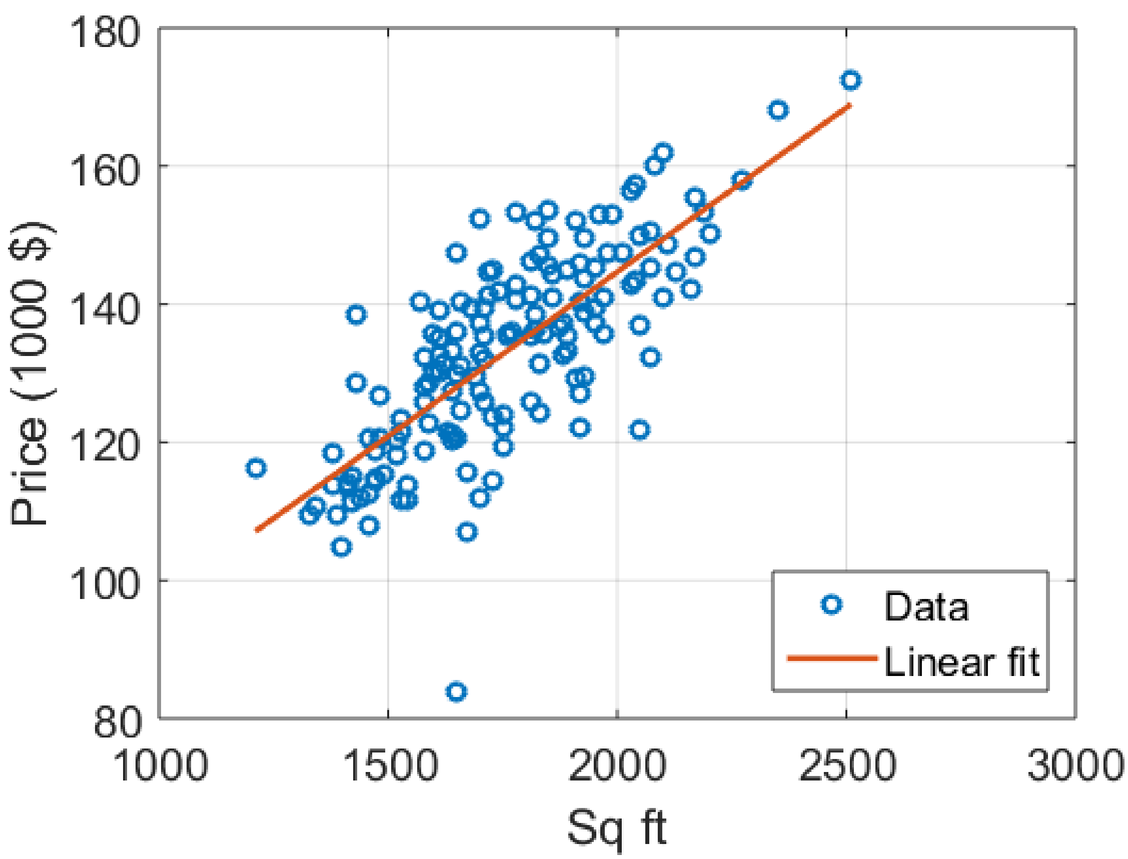


Linear fit

```
n = size(sqft,1);           % number samples
```

```
X = [ones(n,1) sqft];   % data matrix
beta = X \ price;       % coefficients with the least square soln
pricehat = X*beta;      % estimate price

plot(sqft,price,'o', sqft,pricehat,'-','LineWidth',2);
grid on;
xlabel('Sq ft');
set(gca,'Fontsize',16);
ylabel('Price (1000 $)');
legend('Data', 'Linear fit','Location','SouthEast');

fprintf(1,'Dollars per sq ft: %f\n', beta(2)*1000);
```

```
Dollars per sq ft: 47.591749
```



Compute the correlation

```
dsqft = sqft-mean(sqft);
dprice = price-mean(price);
R = dsqft'*dprice/norm(dsqft)/norm(dprice);
fprintf(1,'Correlation = %f\n', R);
```

```
Correlation = 0.762435
```

Published with MATLAB® R2015a