Exam

Bishal sarker

2022-11-19

Contents

Model slection

```
library(leaps)
```

```
Warning: package 'leaps' was built under R version 4.2.2
```

```
# best subset
model  = regsubsets(mpg~., data = mtcars)
model
```

```
Subset selection object
Call: regsubsets.formula(mpg ~ ., data = mtcars)
10 Variables  (and intercept)
     Forced in Forced out
cyl      FALSE      FALSE
disp     FALSE      FALSE
hp       FALSE      FALSE
drat     FALSE      FALSE
wt       FALSE      FALSE
qsec     FALSE      FALSE
vs       FALSE      FALSE
am       FALSE      FALSE
gear     FALSE      FALSE
carb     FALSE      FALSE
```

```
1 subsets of each size up to 8
Selection Algorithm: exhaustive
```

```
d=summary(model)
which.max(d$rsq)
```

```
[1] 8
```

```
d$rsq
```

```
[1] 0.7528328 0.8302274 0.8496636 0.8578510 0.8637377 0.8667078 0.8680976
[8] 0.8687064
```

```
# forward selection
all = lm(mpg ~1 ,data = mtcars)
av  = lm(mpg ~., data = mtcars)
model_step = step(all, direction = 'forward', scope = formula(av))
```

```
Start:  AIC=115.94
mpg ~ 1

       Df Sum of Sq     RSS     AIC
+ wt    1    847.73  278.32  73.217
+ cyl   1    817.71  308.33  76.494
+ disp  1    808.89  317.16  77.397
+ hp    1    678.37  447.67  88.427
+ drat  1    522.48  603.57  97.988
+ vs    1    496.53  629.52  99.335
+ am    1    405.15  720.90 103.672
+ carb  1    341.78  784.27 106.369
+ gear  1    259.75  866.30 109.552
+ qsec  1    197.39  928.66 111.776
<none>              1126.05 115.943

Step:  AIC=73.22
mpg ~ wt

       Df Sum of Sq    RSS    AIC
+ cyl   1    87.150 191.17 63.198
+ hp    1    83.274 195.05 63.840
+ qsec  1    82.858 195.46 63.908
+ vs    1    54.228 224.09 68.283
+ carb  1    44.602 233.72 69.628
+ disp  1    31.639 246.68 71.356
<none>              278.32 73.217
+ drat  1     9.081 269.24 74.156
+ gear  1     1.137 277.19 75.086
+ am    1     0.002 278.32 75.217

Step:  AIC=63.2
mpg ~ wt + cyl

       Df Sum of Sq    RSS    AIC
+ hp    1   14.5514 176.62 62.665
+ carb  1   13.7724 177.40 62.805
<none>              191.17 63.198
```

```
+ qsec  1   10.5674 180.60 63.378
+ gear  1    3.0281 188.14 64.687
+ disp  1    2.6796 188.49 64.746
+ vs    1    0.7059 190.47 65.080
+ am    1    0.1249 191.05 65.177
+ drat  1    0.0010 191.17 65.198

Step:  AIC=62.66
mpg ~ wt + cyl + hp

       Df Sum of Sq    RSS    AIC
<none>              176.62 62.665
+ am    1    6.6228 170.00 63.442
+ disp  1    6.1762 170.44 63.526
+ carb  1    2.5187 174.10 64.205
+ drat  1    2.2453 174.38 64.255
+ qsec  1    1.4010 175.22 64.410
+ gear  1    0.8558 175.76 64.509
+ vs    1    0.0599 176.56 64.654
```

```
model_step$anova
```

```
   Step Df  Deviance Resid. Df Resid. Dev       AIC
1       NA        NA        31  1126.0472 115.94345
2  + wt -1 847.72525        30   278.3219  73.21736
3 + cyl -1  87.14997        29   191.1720  63.19800
4  + hp -1  14.55145        28   176.6205  62.66456
```

```
#perform backward stepwise regression

backward <- step(all, direction='backward', scope=formula(all))
```

```
Start:  AIC=115.94
mpg ~ 1
```

```
backward$anova
```

```
  Step Df Deviance Resid. Df Resid. Dev      AIC
1      NA       NA        31   1126.047 115.9434
```

```
backward$coefficients
```

```
(Intercept)
   20.09062
```

```
## Both

intercept_only <- lm(mpg ~ 1, data=mtcars)
#define model with all predictors
all <- lm(mpg ~ ., data=mtcars)
#perform backward stepwise regression
both <- step(intercept_only, direction='both', scope=formula(all), trace=0)
#view results of backward stepwise regression
both$anova
```

```
   Step Df  Deviance Resid. Df Resid. Dev       AIC
1       NA        NA        31  1126.0472 115.94345
2  + wt -1 847.72525        30   278.3219  73.21736
```

```
3 + cyl -1  87.14997        29   191.1720  63.19800
4  + hp -1  14.55145        28   176.6205  62.66456
```

K fold cross validation

```
library(caret)
```

```
Loading required package: ggplot2
```

```
Loading required package: lattice
```

```
model  = train(
  mpg ~ drat, mtcars, method = "lm",
trControl=  trainControl("cv")
)
model
```

```
Linear Regression

32 samples
 1 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 29, 28, 29, 30, 29, 28, ...
Resampling results:

  RMSE      Rsquared   MAE
  4.309608  0.7433229  3.698327

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Lda vs KNN

```
library(caTools)
```

```
Warning: package 'caTools' was built under R version 4.2.2
```

```
train = sample.split(iris$Species, SplitRatio = .75)
trin = iris[train ==TRUE,]

x = trin[,-5]
y = trin[,5]
tst = iris[train==FALSE,][,-5]
yt = iris[train==FALSE,][,5]
## KNN
library(class)
knnMo = knn(x,tst,y, k =1)
knnMo
```

```
 [1] setosa     setosa     setosa     setosa     setosa     setosa
 [7] setosa     setosa     setosa     setosa     setosa     setosa
[13] versicolor versicolor versicolor versicolor versicolor virginica
[19] versicolor versicolor versicolor versicolor versicolor versicolor
[25] virginica  virginica  virginica  versicolor virginica  virginica
```

```
[31] virginica  virginica  virginica  virginica  virginica  virginica
Levels: setosa versicolor virginica
```

```
table(yt, knnMo)
```

```
            knnMo
yt           setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         11         1
  virginica       0          1        11
```

```
mean(knnMo !=yt)
```

```
[1] 0.05555556
```

```
# lDA
library(MASS)
lid = lda(Species ~ . , trin)
pr = predict(lid,tst)
table(pr$class, yt)
```

```
            yt
             setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         12         0
  virginica       0          0        12
```

```
mean(pr$class != yt)
```

```
[1] 0
```

```
# QDA

liq = qda(Species ~ . , trin)
prq = predict(liq,tst)
table(prq$class, yt)
```

```
            yt
             setosa versicolor virginica
  setosa         12          0         0
  versicolor      0         11         0
  virginica       0          1        12
```

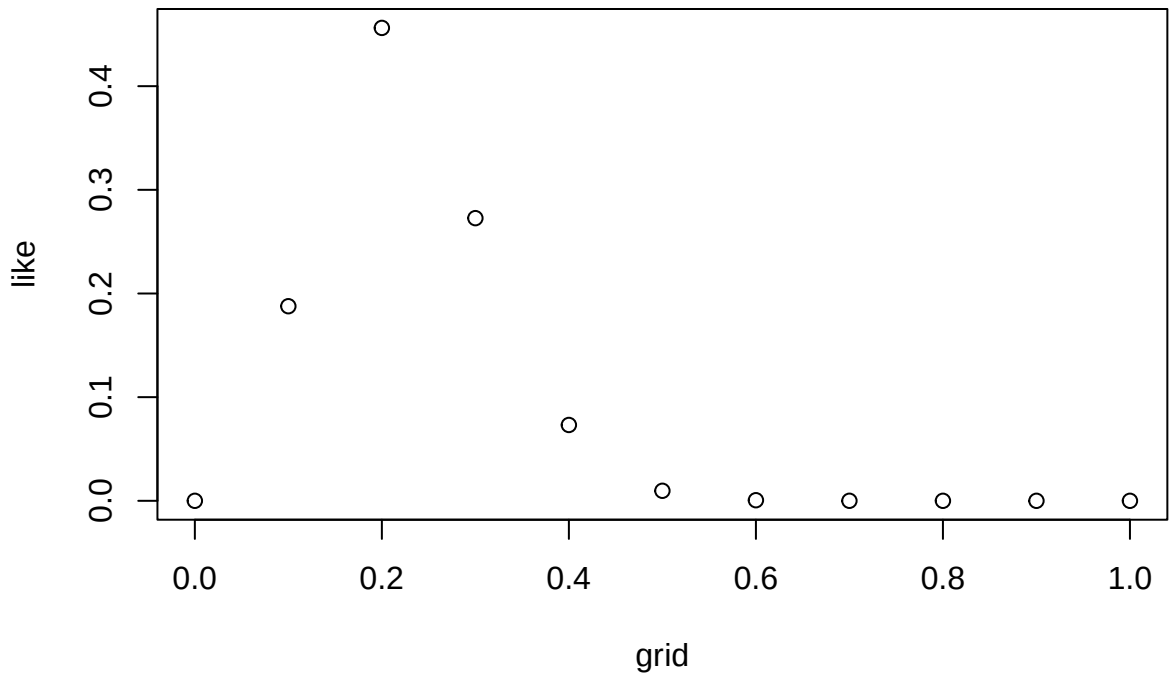
```
 error = mean(prq$class != yt)
```

Bayesian posterior

Beta binomial mode

```
n = 20
Y = 4
a = 3
b =1

grid = seq(0,1,.1)

prior = dbeta(grid, a,b)
```

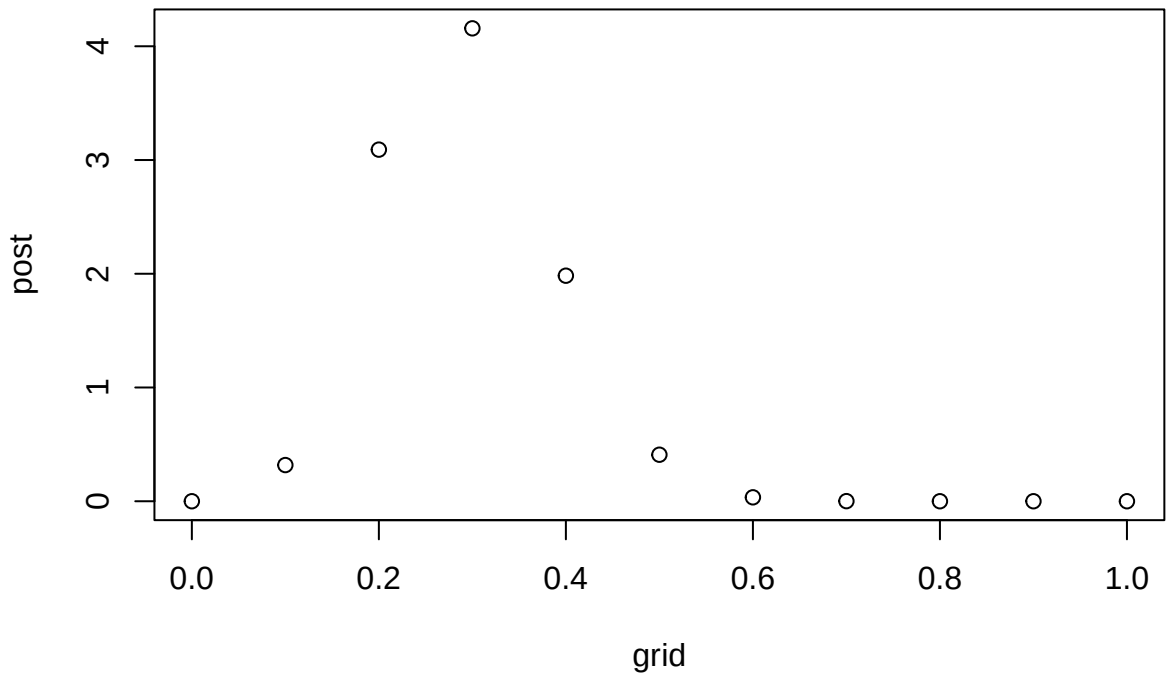
```r
prior = prior/ sum(prior) # standarized

like = dbinom(4, 20, grid)
like = like/sum(like)

post = like*prior
post = post/sum(post)

plot(grid, like)
```




```r
# Exact calculation

post = dbeta(grid, (Y+a), (n+b-Y) )
post
```

```
 [1] 0.000000e+00 3.179966e-01 3.091449e+00 4.157560e+00 1.982998e+00
 [6] 4.091499e-01 3.438822e-02 8.691018e-04 2.948236e-06 9.120054e-11
[11] 0.000000e+00
```

```r
plot(grid, post)
```

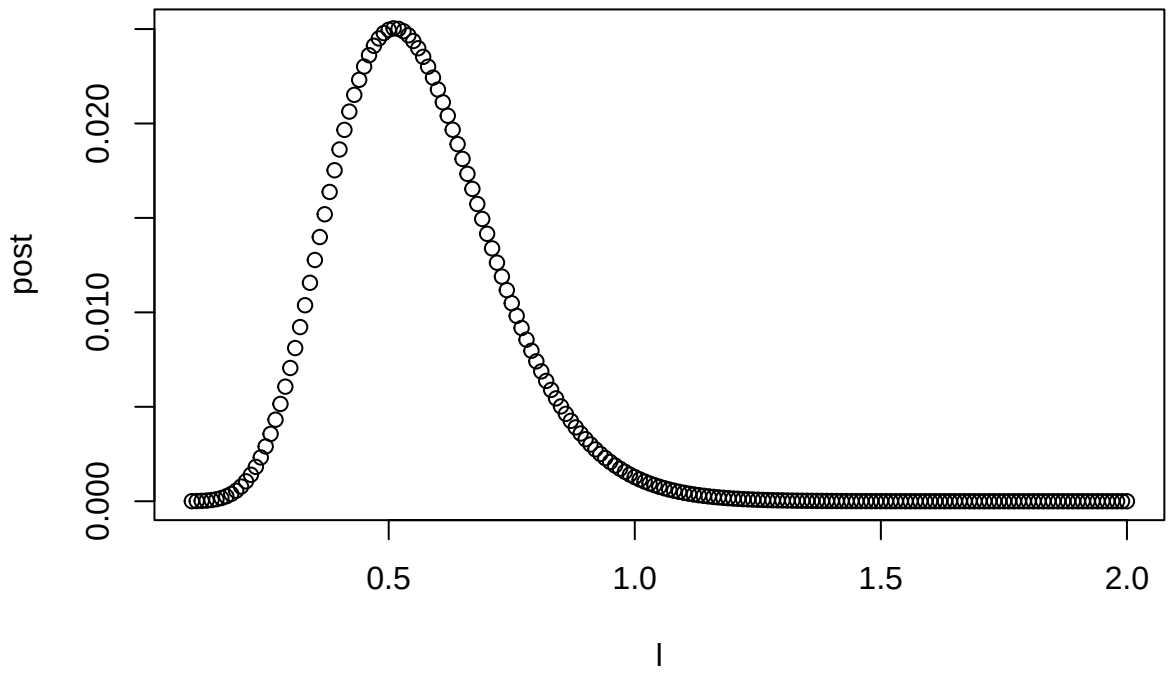



```
man = (Y +a)/(n- Y +b)

va = (Y + a)/(n - Y+b)^2

qbeta(c(.05,.95),Y+a, n-Y+b)
```

```
[1] 0.1524797 0.4509754
```

Gamma poisson

```
N = 20
y =11
a = .5
b = .5
l = seq(.1,2,.01)
like = dpois(y, N*l)
like = like/sum(like)

prior = dgamma(l, .5,.5)
prior = prior/sum(prior)

post = like * prior
post = post / sum(post)

plot(l, post)
```




```
li = ( y + a)/(N + b)
sq = (y + a)/ (N+b)^2
sqrt(sq)
```

```
[1] 0.1654227
```

Normal normal]

```
y = .1
sigma = .005

m = .05
s = .025
gri = seq(0,.15, .001)
like = dnorm(y, gri, sigma)
like = like /sum(like)

pr = dnorm(gri, m, s)
pr =  post/sum(post)

post = pr * like
```

```
Warning in pr * like: longer object length is not a multiple of shorter object
length
```

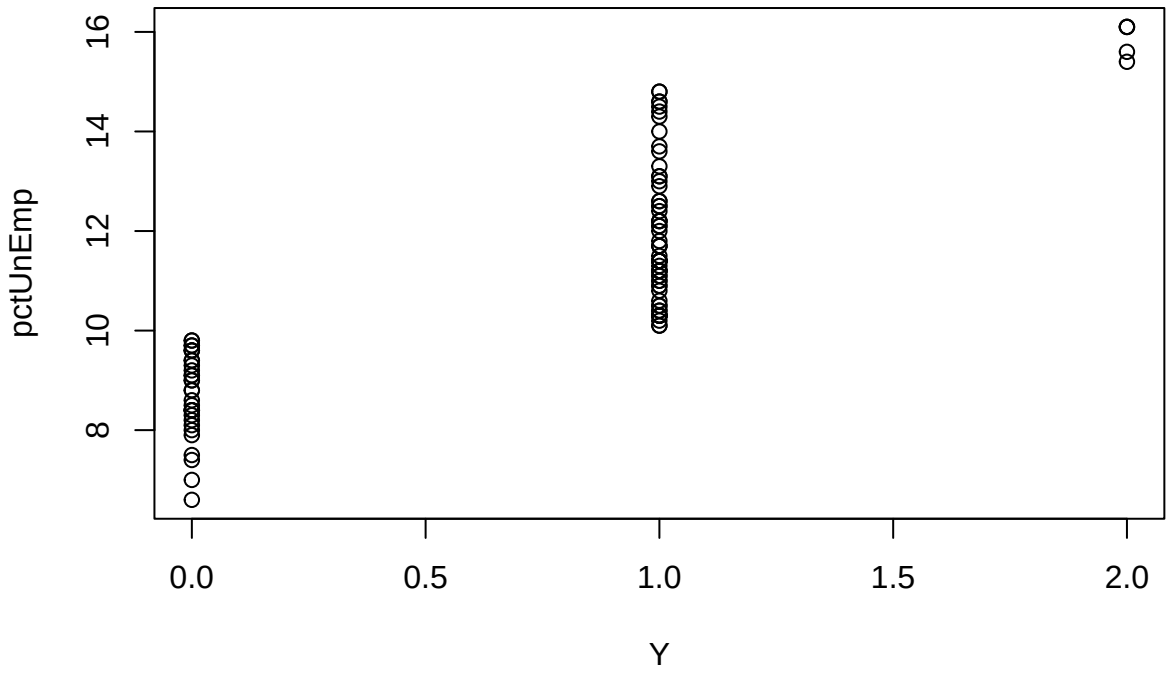
```
post = post/sum(post)
```

```
su  = 1/sigma^2 + 1/s^2
ms = (y/sigma^2 + m/s^2 )
mu = ms/su
mu
```

```
[1] 0.09807692
```

Joint distribution

```
dat      <- read.csv("http://www4.stat.ncsu.edu/~reich/ST590/assignments/Obama2012.csv")
pctObama <- 100*dat[,2]
pctUnEmp <- dat[,18]

##############################################
# Convert to discrete variables
##############################################

X        <- ifelse(pctObama>50,1,0)
Y        <- ifelse(pctUnEmp>10,1,0)+
  ifelse(pctUnEmp>15,1,0)
plot(Y,pctUnEmp)
```




```
# Compute the sample joint distribution

table(X,Y)/100
```

```
   Y
X      0    1    2
  0 0.20 0.48 0.03
  1 0.12 0.15 0.02
```

```
# Compute the sample marginal distributions

table(X)/100
```

```
X
   0    1
0.71 0.29
```

```
table(Y)/100
```

```
Y
   0    1    2
0.32 0.63 0.05
```

```
# Compute the conditional probabilities

mean(X[Y==0])
```

```
[1] 0.375
```

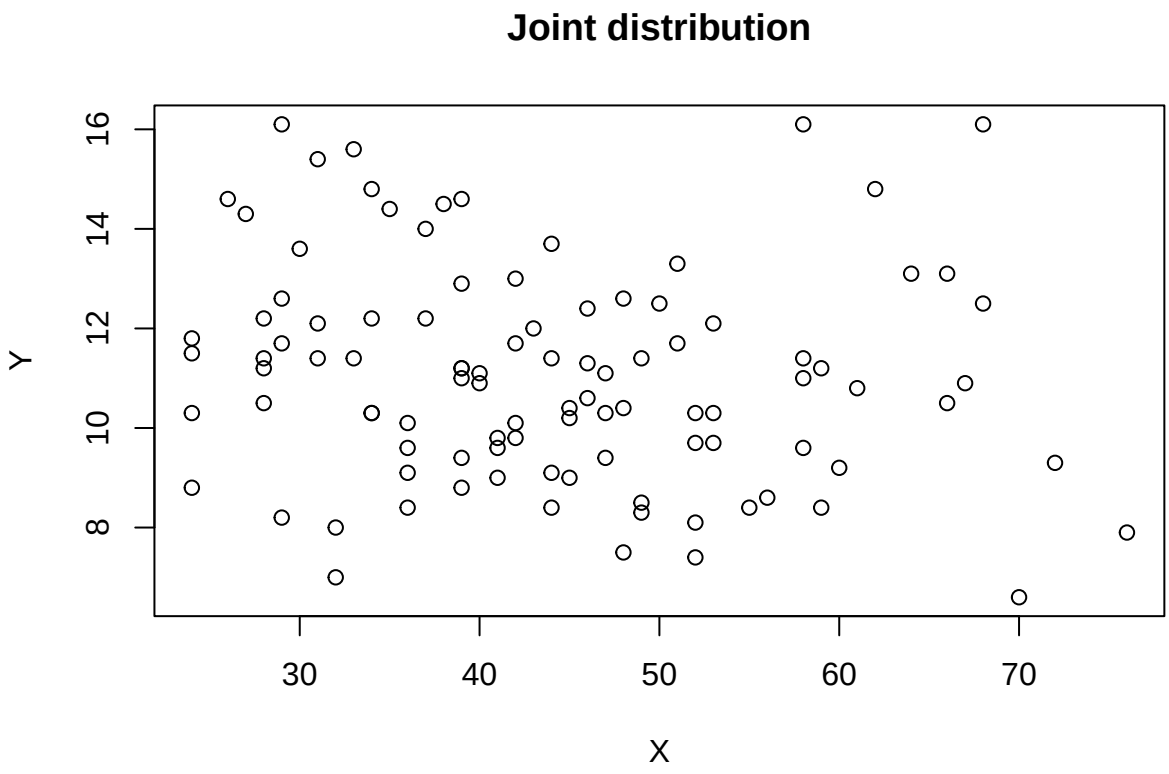
```
mean(X[Y==1])
```

```
[1] 0.2380952
```

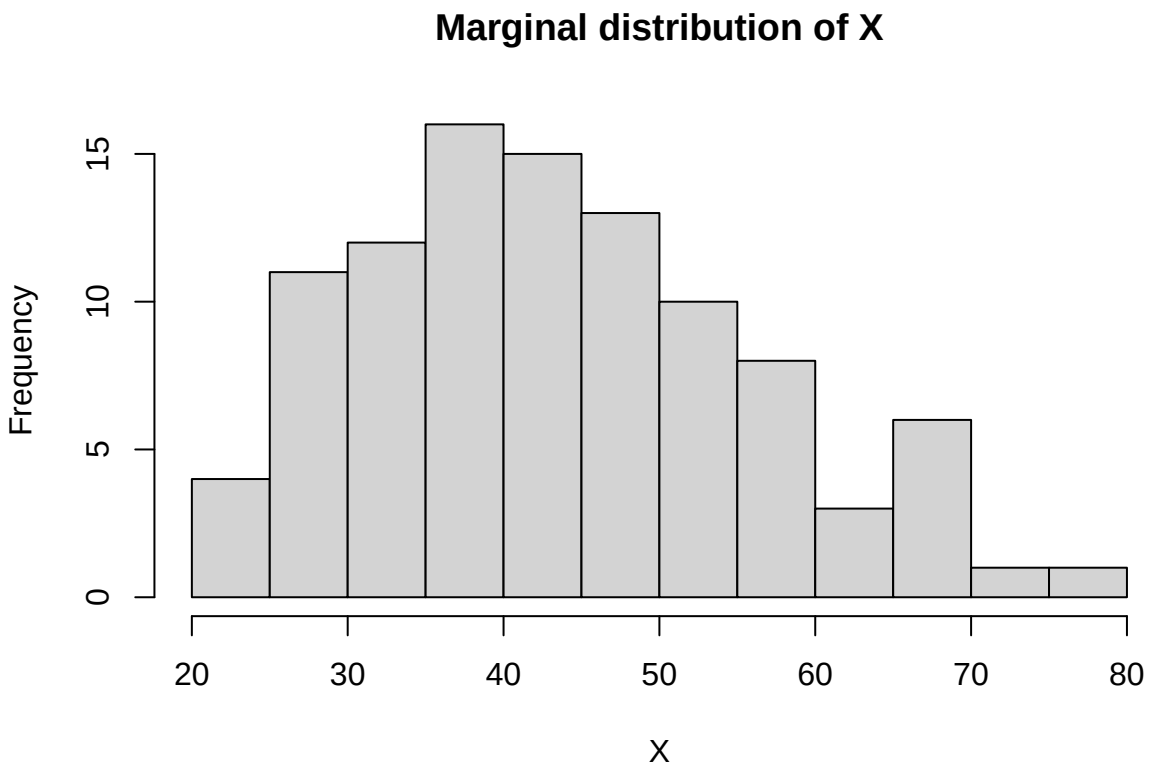
```
mean(X[Y==2])
```

```
[1] 0.4
```

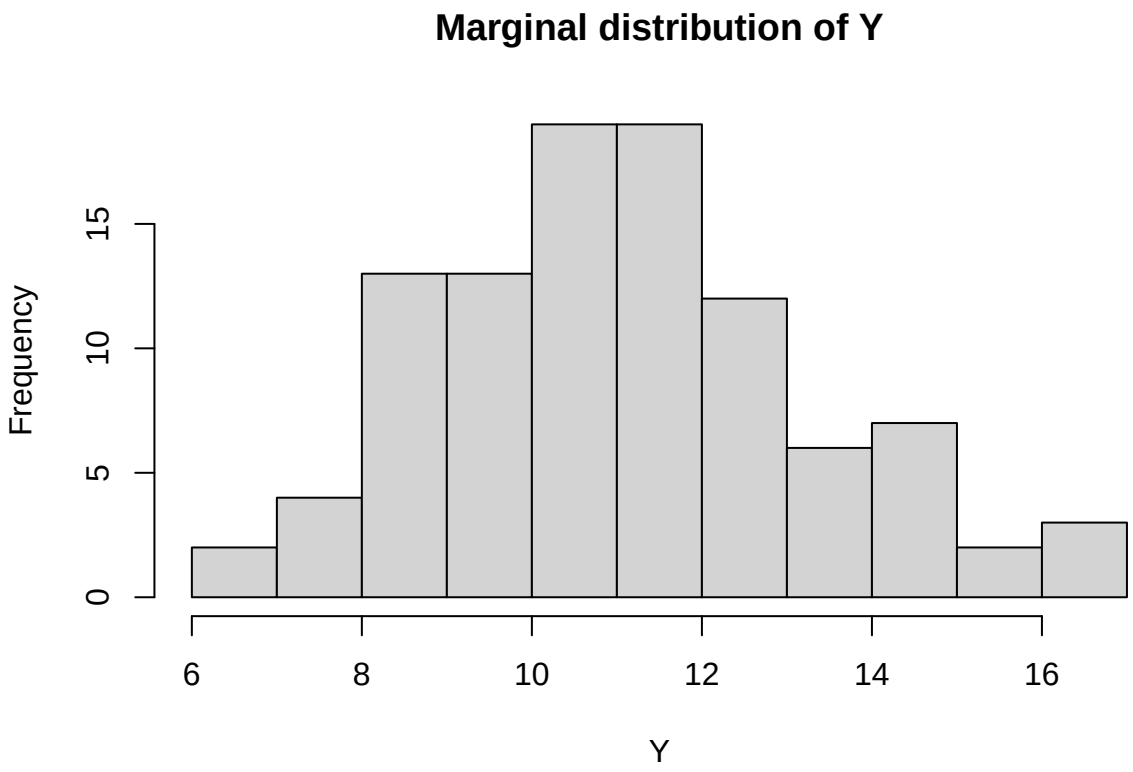
```
##############################################
# Plot for continuous variables
##############################################

X <- pctObama
Y <- pctUnEmp

#Joint

plot(X,Y,main="Joint distribution")
```



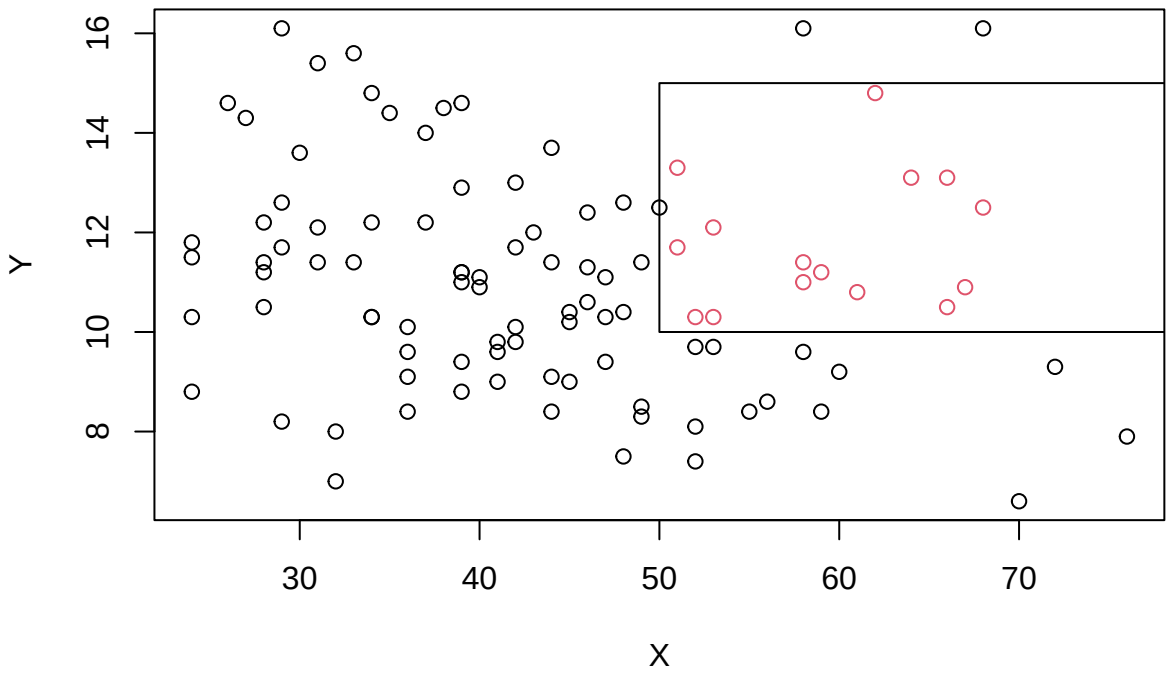

```
# Marginals

hist(X,main="Marginal distribution of X")
```



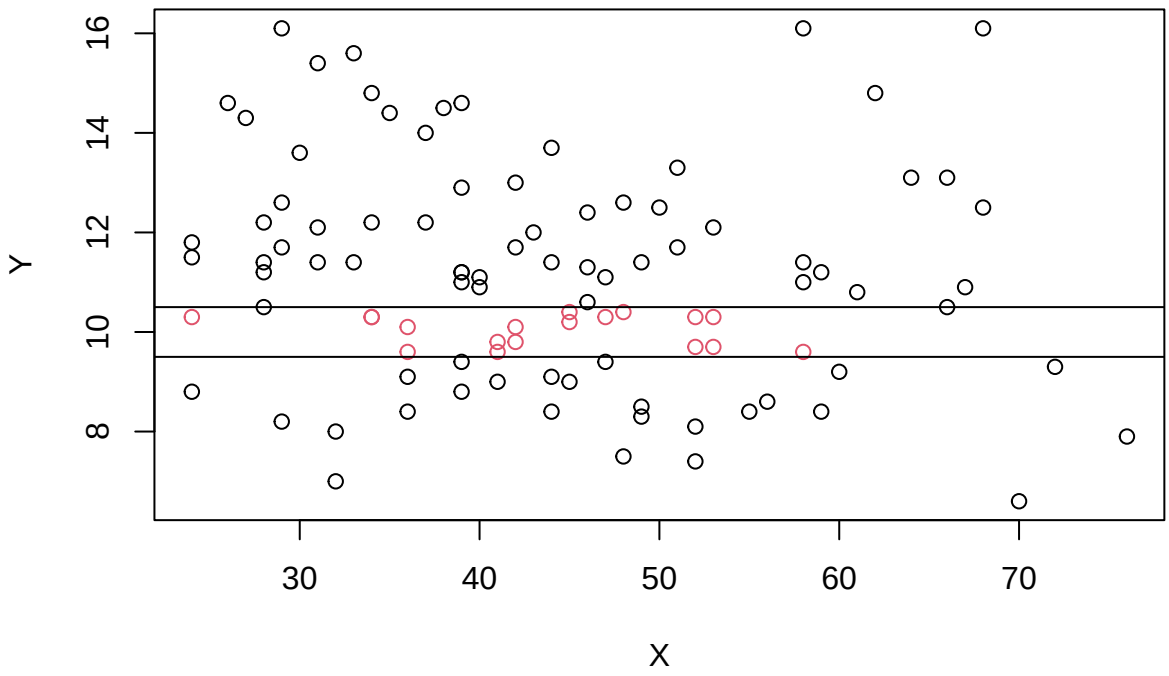

```
hist(Y,main="Marginal distribution of Y")
```



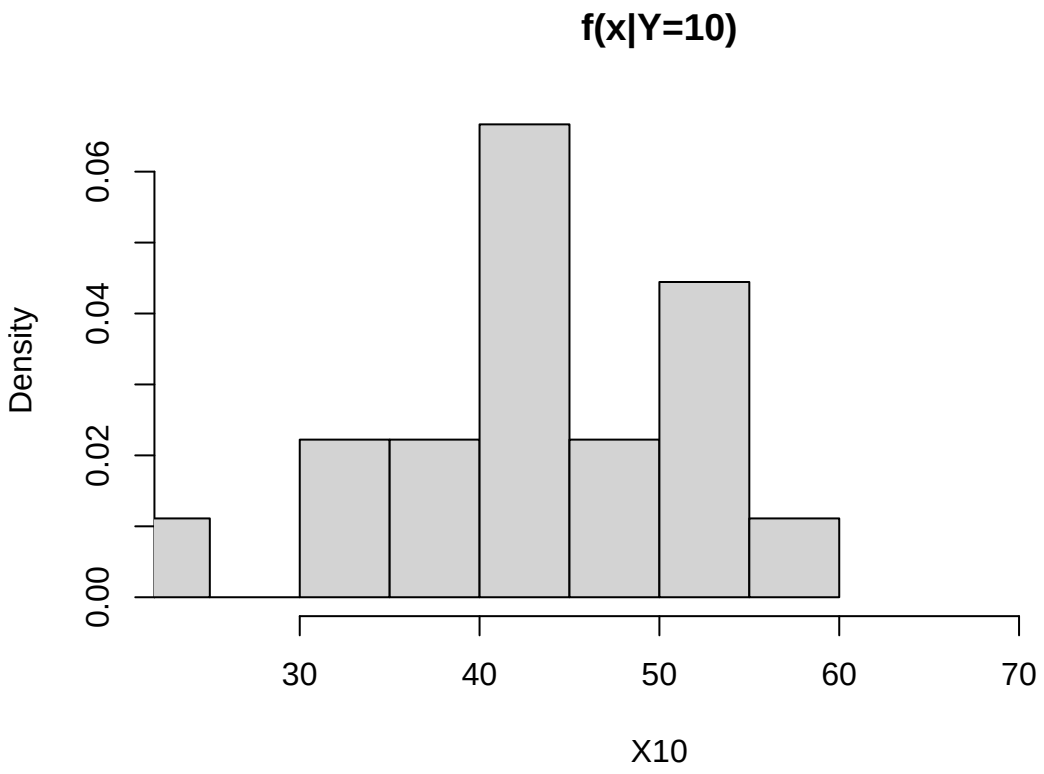

```
# Probability in a set

inA <- (X>50) & (Y>10) & (Y<15)
plot(X,Y,col=ifelse(inA,2,1),main="Prob in set A")
polygon(c(50,50,100,100,50),c(10,15,15,10,10))
```




```
# Approximate conditional pdf

Y10 <- Y>9.5 & Y<10.5
plot(X,Y,col=ifelse(Y10,2,1))
abline(9.5,0)
abline(10.5,0)
```

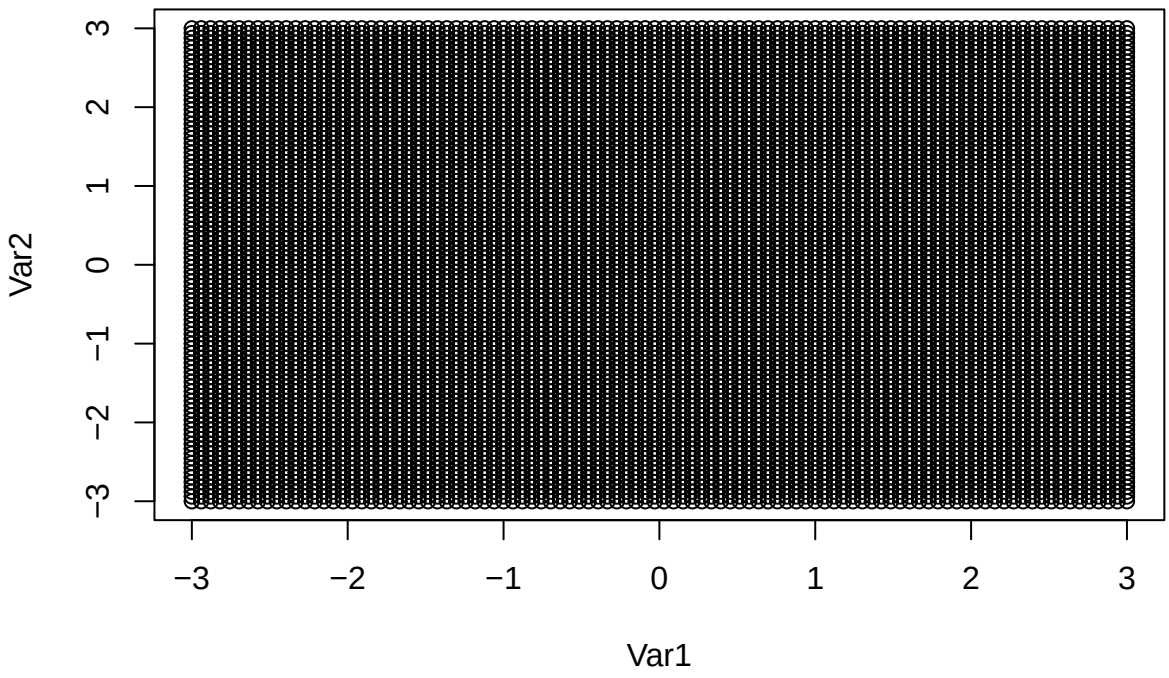



```
X10 <- X[Y10]
hist(X10,main="f(x|Y=10)",xlim=range(X),prob=TRUE)
```



Bivariate normal

```
binorm<-function(x,y,muX=0,muY=0,sigmaX=1,sigmaY=1,rho=0){

  c    <- sigmaX*sigmaY*sqrt(1-rho^2)*2*pi
  d    <- 1/(1-rho^2)
  z_x  <- (x-muX)/sigmaX
  z_y  <- (y-muY)/sigmaY

  pdf <- (1/c)*exp(-0.5*d*(z_x^2+z_y^2-2*rho*z_x*z_y))

return(pdf)}


 m    <- 100
 pts  <- seq(-3,3,length=m)
 grid <- expand.grid(pts,pts)
 plot(grid)
```




```
muX    <- 0
muY    <- 0
sigmaX <- 1
sigmaY <- 1
rho    <- 0.9

pdf    <- binorm(grid[,1],grid[,2],muX,muY,sigmaX,sigmaY,rho)
pdf    <- matrix(pdf,m,m)

library(fields)
```

```
Warning: package 'fields' was built under R version 4.2.2
```

```
Loading required package: spam
```

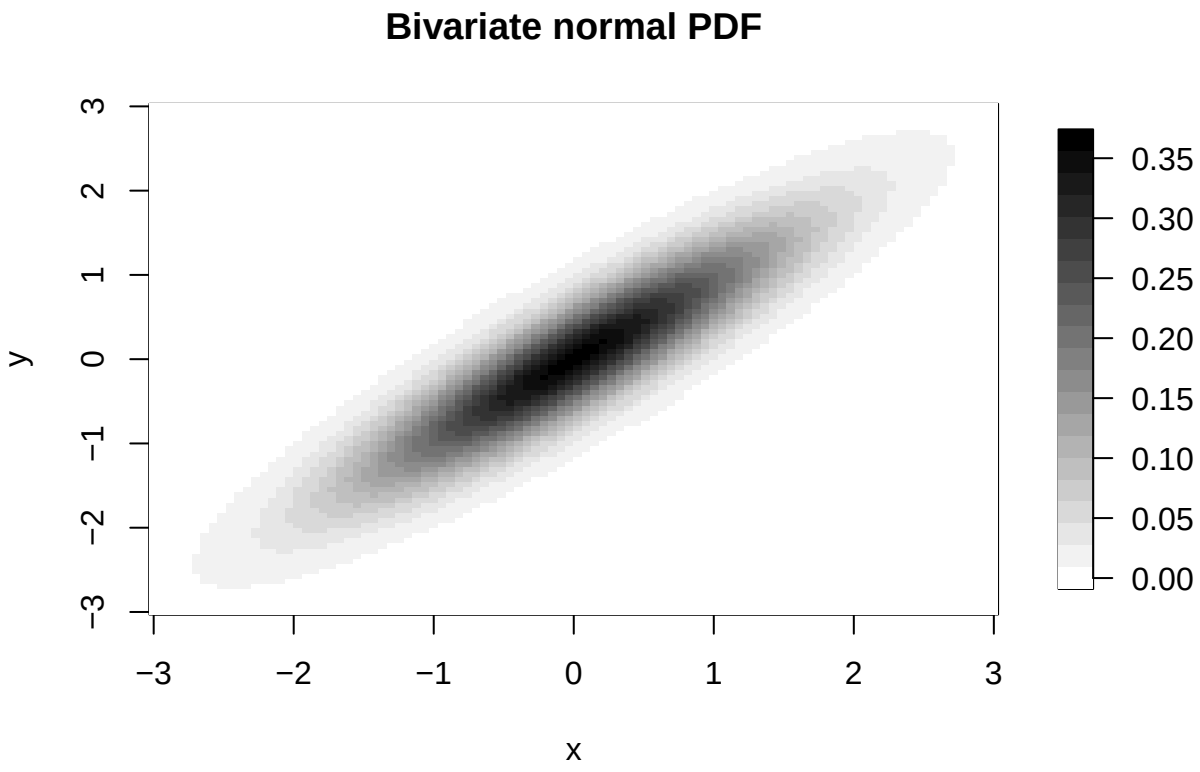
```
Warning: package 'spam' was built under R version 4.2.2
```

```
Spam version 2.9-1 (2022-08-07) is loaded.
Type 'help( Spam)' or 'demo( spam)' for a short introduction
and overview of this package.
Help for individual functions is also obtained by adding the
suffix '.spam' to the function name, e.g. 'help( chol.spam)'.


Attaching package: 'spam'
```

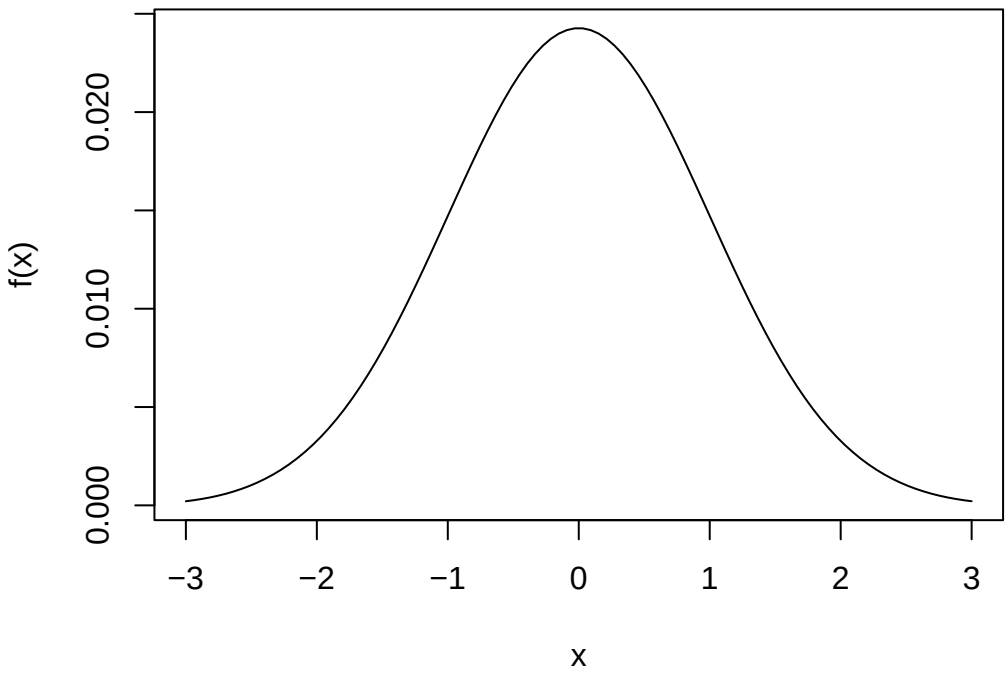
```
The following objects are masked from 'package:base':

    backsolve, forwardsolve
```

```
Loading required package: viridis
Warning: package 'viridis' was built under R version 4.2.2
Loading required package: viridisLite

Try help(fields) to get started.
```

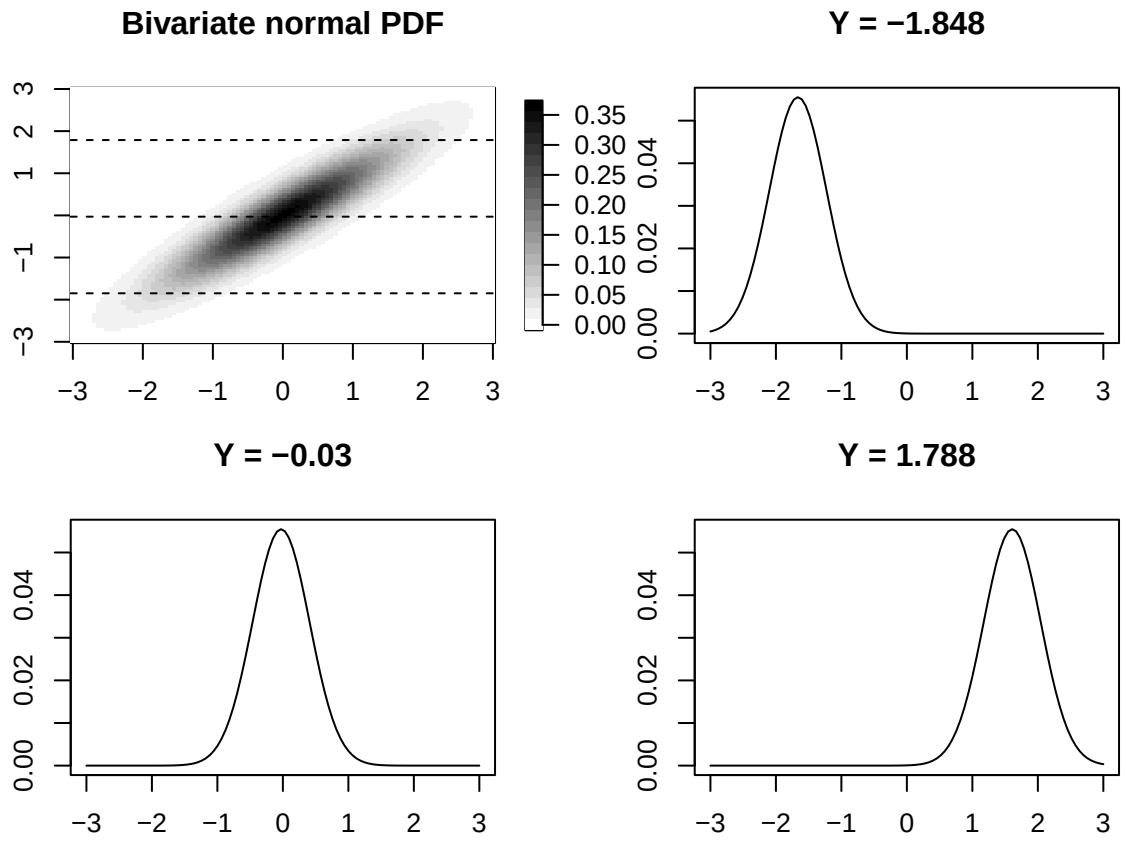
```r
image.plot(pts,pts,pdf,
           xlab="x",ylab="y",
           main="Bivariate normal PDF",
           col=gray(1-seq(0,1,.05)))
```




```r
## MARGINAL
fx <- colSums(pdf)
fx <- fx/sum(fx)

plot(pts,fx,type="l",xlab="x",ylab="f(x)")
```

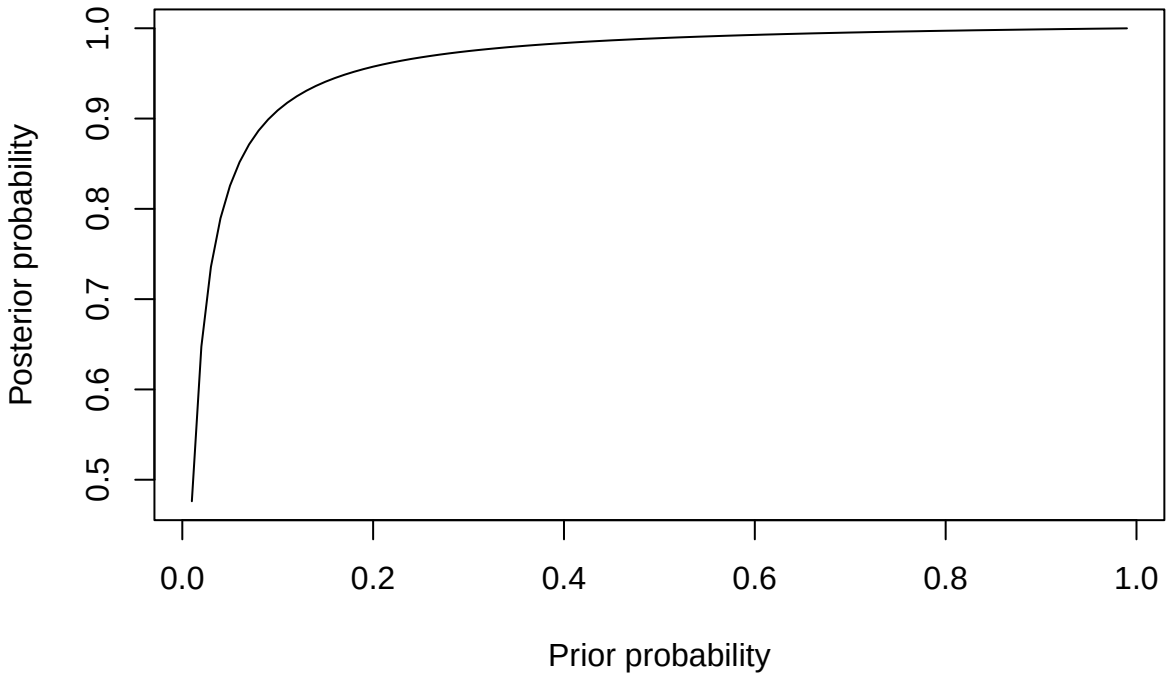



```
##Conditional distribution

par(mfrow=c(2,2))
 image.plot(pts,pts,pdf,
            xlab="x",ylab="y",
            main="Bivariate normal PDF",
            col=gray(1-seq(0,1,.05)))

 abline(pts[80],0,lty=2)
 abline(pts[50],0,lty=2)
 abline(pts[20],0,lty=2)

 cond20 <- pdf[20,]
 cond20 <- cond20/sum(cond20)
 plot(pts,cond20,type="l",xlab="x",ylab="f(x|y)",
      main=paste("Y =",round(pts[20],3)))

 cond50 <- pdf[50,]
 cond50 <- cond50/sum(cond50)
 plot(pts,cond50,type="l",xlab="x",ylab="f(x|y)",
      main=paste("Y =",round(pts[50],3)))

 cond80 <- pdf[80,]
 cond80 <- cond80/sum(cond80)
 plot(pts,cond80,type="l",xlab="x",ylab="f(x|y)",
      main=paste("Y =",round(pts[80],3)))
```



HIV data

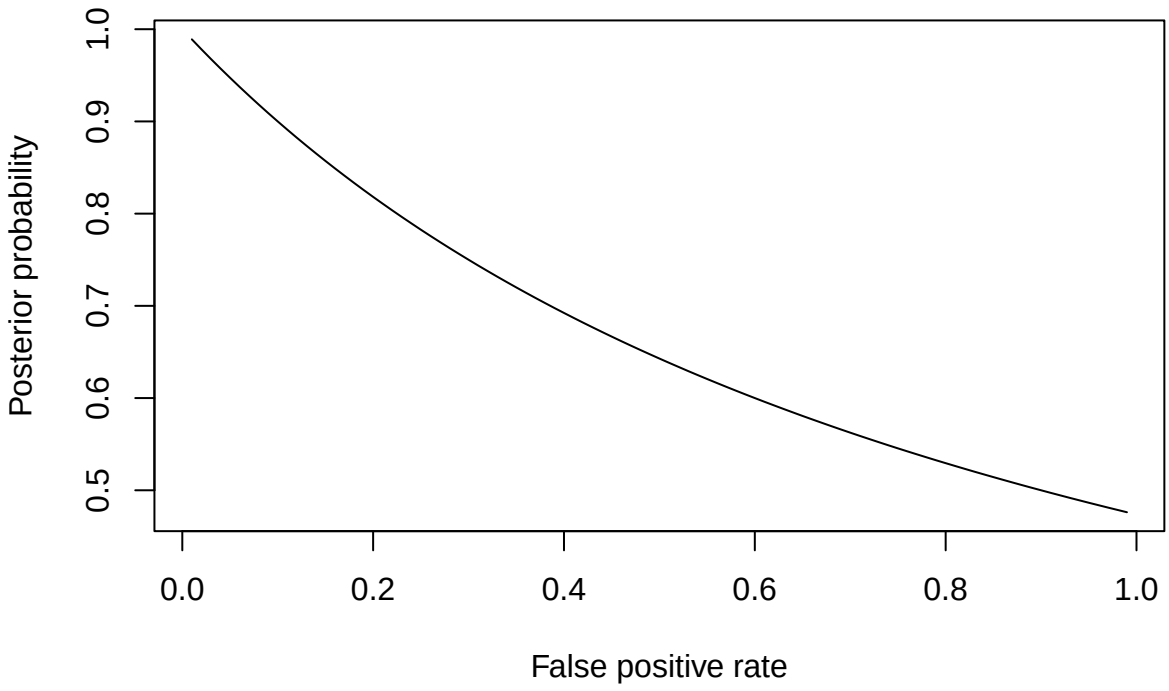
```
## Compute the posterior probability the patient has HIV given a positive test

post_prob<-function(p,q0,q1){
  p*q1/(p*q1+(1-p)*q0)
}

## Base case

p  <- 0.50   # Prior probability
q0 <- 0.01   # False positive probability
q1 <- 0.90   # True positive probability


## Effect of the prior
grid  <- seq(0.01,0.99,.01)

plot(grid,post_prob(grid,q0,q1),
     type="l",
     xlab="Prior probability",
     ylab="Posterior probability")
```



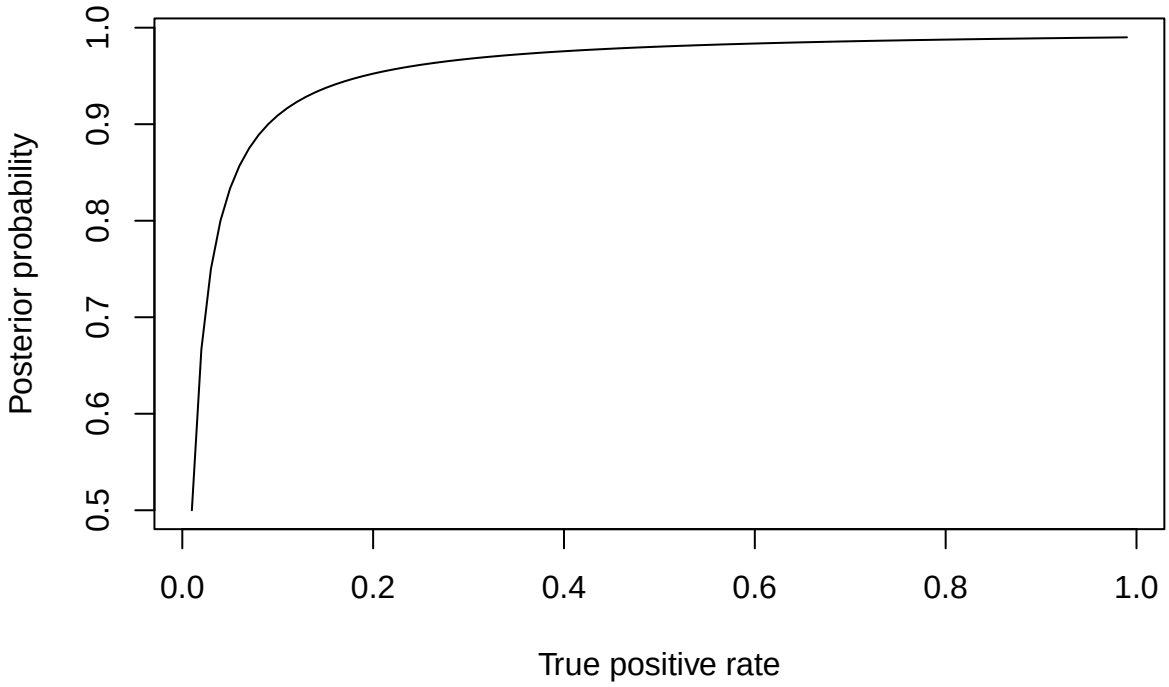

```
post_prob(p,q0,q1)
```

```
[1] 0.989011
```

```
## Effect of the likelihood - false positive rate

plot(grid,post_prob(p,grid,q1),
     type="l",
     xlab="False positive rate",
     ylab="Posterior probability")
```




```
## Effect of the likelihood - true positive rate
plot(grid,post_prob(p,q0,grid),
     type="l",
     xlab="True positive rate",
     ylab="Posterior probability")
```




```
#Monte Carlo approximation:

n     <- 10000
theta <- NULL
Y     <- NULL

#start sampling
for(i in 1:n){
   theta[i] <- rbinom(1,1,p)
   prob     <- ifelse(theta[i]==1,q1,q0)
   Y[i]     <- rbinom(1,1,prob)
}

table(Y,theta)/n
```

```
   theta
Y        0      1
  0 0.4887 0.0490
  1 0.0052 0.4571
```