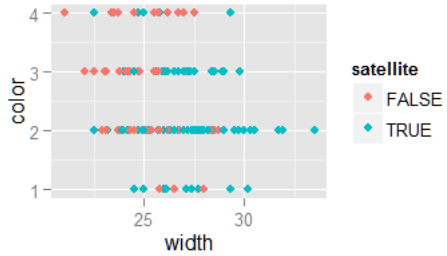
Categorical Data Analysis:

```
> ## Linear Discriminant Analysis:
> head(crab)
  color spine width satellite weight
1     2     3  28.3      TRUE   3050
2     3     3  22.5     FALSE   1550
3     1     1  26.0      TRUE   2300
4     3     3  24.8     FALSE   2100
5     3     3  26.0      TRUE   2600
6     2     3  23.8     FALSE   2100
```




```
> ## Try to come up with a rule whether or not a crab has
> ## a sattelite based on its width and color:
> require("ggplot2")
> ggplot(data=crab, aes(x=width, y=color, color=satellite)) + geom_point()
> require("MASS")
> lda(satellite ~ width + color, data=crab)
Call:
lda(satellite ~ width + color, data = crab)

Prior probabilities of groups:
FALSE  TRUE
0.358 0.642

Group means:
      width color
FALSE  25.2  2.73
TRUE   26.9  2.28

Coefficients of linear discriminants:
         LD1
width  0.429
color -0.552

> ## The linear discriminant function equals:
> ## 0.429*width + (-0.552)*color

> ## For given (width,color), when this is larger than a constant that
> ## depends on the sample means, covariance matrix and estimate of prior
> ## probability pi, then classify as Success, otherwise as Failure
> ## One can get the classified values for the input dataset directly (see
below how exactly this is done):
> fitted = lda(satellite ~ width + color, data=crab, CV=TRUE)$class
> fitted
  [1] TRUE  FALSE TRUE  FALSE TRUE  FALSE TRUE  FALSE
  [9] FALSE TRUE  FALSE TRUE  TRUE  FALSE TRUE  TRUE
...
> ## Prediction Matrix:
> table(crab$satellite, fitted)
       fitted
        FALSE TRUE
  FALSE    28   34
  TRUE     14   97
> ## Classifier has a hard time predicting FALSE correctly
```

Do it “by hand”:

```
> crab0=crab[!crab$satellite,] #just those with y=0
> crab1=crab[crab$satellite,]  #just those with y=1
> head(crab0)
  color spine width satellite weight
2     3     3  22.5     FALSE   1550
4     3     3  24.8     FALSE   2100
6     2     3  23.8     FALSE   2100
7     1     1  26.5     FALSE   2350
8     3     2  24.7     FALSE   1900
9     2     1  23.7     FALSE   1950
> head(crab1)
   color spine width satellite weight
1      2     3  28.3      TRUE   3050
3      1     1  26.0      TRUE   2300
5      3     3  26.0      TRUE   2600
13     2     3  28.2      TRUE   3050
15     2     1  26.0      TRUE   2300
16     1     1  27.1      TRUE   2950
> crab0x=crab0[,c(1,3)]
> crab1x=crab1[,c(1,3)]
> xbar0=colMeans(crab0x) #sample mean in group y=0
> xbar1=colMeans(crab1x) #sample mean in group y=1
> xbar0
color width
 2.73 25.17
> xbar1
color width
 2.28 26.93
> S0=cov(crab0x) #sample variance-covariance matrix in group y=0
> S1=cov(crab1x) #sample variance-covariance matrix in group y=1
> S0
       color  width
color  0.792 -0.341
width -0.341  2.802
> S1
       color  width
color  0.494 -0.226
width -0.226  4.280
> n0=dim(crab0x)[1]
> n1=dim(crab1x)[1]
> S = ((n0-1)*S0 + (n1-1)*S1)/(n0+n1-2) #pooled var-covariance matrix
> S #pooled estimate
       color  width
color  0.600 -0.267
width -0.267  3.753
> (xbar1-xbar0)*solve(S) #linear discriminant function
      color width
[1,] -0.553  0.43


> p0=mean(crab$satellite)
> p0
[1] 0.642
> tresh=0.5*(xbar0+xbar1)*solve(S)*(xbar1-xbar0) - log(p0/(1-p0))
```

```
> tresh
     [,1]
[1,] 9.23
> ## So, whenever the linear discriminant function is larger than 9.23,
> ## we classify the x-vector as belonging to y=1, otherwise to y=0.
> ## We could apply this to the observed data:
> disc.funct = (xbar1-xbar0)*solve(S)*t(crab[,c(1,3)])
> disc.funct
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
[1,] 11.1 8.01 10.6    9 9.52 9.12 10.8 8.96 9.08  9.34  8.78  9.98    11
...
     [,164] [,165] [,166] [,167] [,168] [,169] [,170] [,171] [,172] [,173]
[1,]   8.83   10.3   9.98    8.7    9.6   9.56   10.8   11.5   9.39   9.42

> disc.funct>9.23
     [,1]  [,2] [,3]  [,4] [,5]  [,6] [,7]  [,8]  [,9] [,10] [,11] [,12] [,13]
[1,] TRUE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE
...
     [,163] [,164] [,165] [,166] [,167] [,168] [,169] [,170] [,171] [,172] [,173]
[1,]   TRUE  FALSE   TRUE   TRUE  FALSE   TRUE   TRUE   TRUE   TRUE   TRUE   TRUE

> ## Better to use leave-one-out classification (called cross-validation),
where the classification of a crab is based on the dataset with that crab
deleted: For that dataset, find the linear discriminant fct and used it on
the x-vector from the left-out crab to get its classification. Do this for
all crabs. You can get this with lda by specifying CV=TRUE (for output, see
above):
> fitted = lda(satellite ~ width + color, data=crab, CV=TRUE)$class
```