STAT 441: Lecture 15
Linear discriminant analysis

Once we know the probability density of the groups
we can write the optimal classification rule
Thus the only missing step is to estimate the density
Let us do it in the parametric way now: fit normal distribution

Assume that the distributions of the groups are multivariate normal

$f(x) = Ce^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$ $\qquad C = 1/\sqrt{(2\pi)^p \det(\boldsymbol{\Sigma})}$

We invoke the optimal rule for classifying with minimum expected cost of missclassification:

classify to k-th class when $\sum_{i \neq k} p_i f_i(\mathbf{x}_0) c(k|i)$ is the smallest

For equal misclassification costs, this rule is equivalent to:

classify to k-th class when $p_k f_k(\mathbf{x}_0)$ is the largest

When f_k is multivariate normal, then the classification scores equivalent to $p_k f_k(\mathbf{x}_0)$ are their logs:

$$\log p_k - \frac{1}{2}\log\det(\boldsymbol{\Sigma}_k) - \frac{1}{2}(\mathbf{x}_0 - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_0 - \boldsymbol{\mu}_k).$$

If the $\boldsymbol{\Sigma}_k$ are all equal to $\boldsymbol{\Sigma}$, then the scores simplify to

$$\log p_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\top \boldsymbol{\Sigma}^{-1}\mathbf{x}_0$$

- which makes the boundaries linear

Well, and yes: $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are not known

Not known? Replace by estimates!

In the actual scores, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are replaced by their estimates - sample versions $\bar{\mathbf{x}}_k$ and $\mathbf{S}_k$. When $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$, then $\boldsymbol{\Sigma}$ is replaced by the pooled estimate

$$\mathbf{S}_{\text{pooled}} = \frac{\sum_k (n_k - 1)\mathbf{S}_k}{\sum_k (n_k - 1)}.$$

There are also other ways to estimate $\boldsymbol{\mu}_k$'s and $\boldsymbol{\Sigma}$, but this is the most straightforward, and often satisfactory. It is called **plug-in rule** or prediction. A true Bayesian would not do that, but use instead a **predictive** rule (or prediction), which takes a linear combination of the normal distribution for various $\boldsymbol{\mu}_k$, each weighted by its posterior probability.

The resulting techniques are

Linear Discriminant Analysis (LDA) - if all $\boldsymbol{\Sigma}_k$ assumed equal

Quadratic Discriminant Analysis (QDA) - if not (assumed equal)

Another way of looking at LDA: Fisher's linear discriminants

"Principal components for k groups"

We look for $\mathbf{a}$('s) maximizing

$$\frac{\mathbf{a}^\top \left(\sum_k (\boldsymbol{\mu}_k - \bar{\boldsymbol{\mu}})(\boldsymbol{\mu}_k - \bar{\boldsymbol{\mu}})^\top\right)\mathbf{a}}{\mathbf{a}^\top \boldsymbol{\Sigma} \mathbf{a}}.$$

In the sample version:

$$\frac{\mathbf{a}^\top \left(\sum_k n_k (\bar{\mathbf{x}}_k - \bar{\mathbf{x}})(\bar{\mathbf{x}}_k - \bar{\mathbf{x}})^\top\right)\mathbf{a}}{\mathbf{a}^\top \left(\sum_k \sum_i^{n_k} (\mathbf{x}_{ki} - \bar{\mathbf{x}}_k)(\mathbf{x}_{ki} - \bar{\mathbf{x}}_k)^\top\right)\mathbf{a}} = \frac{\mathbf{a}^\top \mathbf{B} \mathbf{a}}{\mathbf{a}^\top \mathbf{W} \mathbf{a}},$$

where $\bar{\mathbf{x}}_k$ is the sample mean of the i-th class
and $\bar{\mathbf{x}}$ the sample mean of all data.

Similarly to principal component analysis, we are looking for maximimizing $\mathbf{a}_1$; then for maximizing $\mathbf{a}_2$ orthogonal to $\mathbf{a}_1$; and so forth. Note that the maximized function is same for $c\mathbf{a}$ as for $\mathbf{a}$; hence we may set $\mathbf{a}^\top\mathbf{a} = 1$, for all $\mathbf{a}_j$.

Fisher's linear discriminants

Fisher's linear discriminants: projections of the datapoints $\mathbf{a}_j^\top \mathbf{x}_i$, $i = 1, \ldots, n$. Concerning j, there are at most $\min\{p, g-1\}$ linear discriminants, where p is the number of features and g the number of classes - because the rank of $\mathbf{B}$ is at most p and at most $g-1$.

The classification rule using first r linear discriminants is:

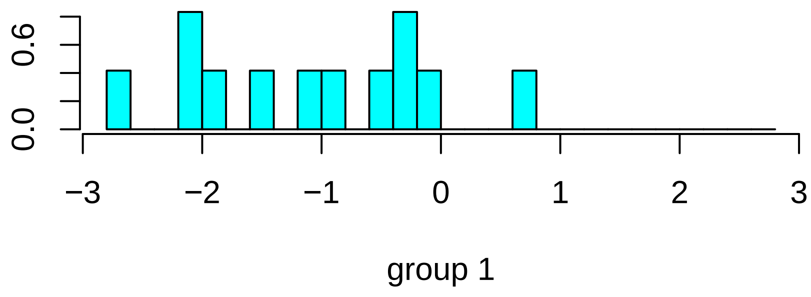
classify $\mathbf{x}_0$ to k-th class if

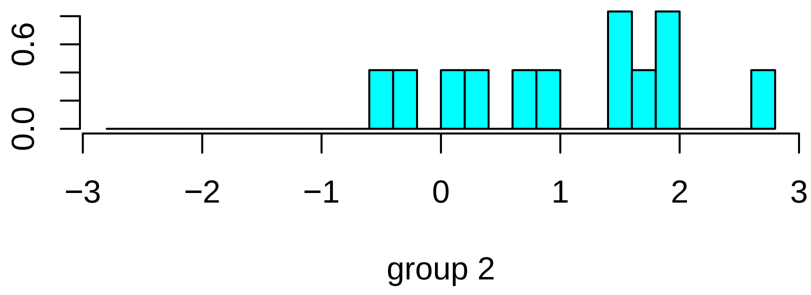
$$\sum_{j=1}^{r} (\mathbf{a}_j^\top (\mathbf{x}_0 - \bar{\mathbf{x}}_k))^2 \leqslant \sum_{j=1}^{r} (\mathbf{a}_j^\top (\mathbf{x}_0 - \bar{\mathbf{x}}_i))^2 \text{ for all } i \neq k$$

It is equivalent to optimal classification, if all prior probabilities are equal and $r = g$, the total number of discriminants.

Plotting

Linear discriminants (first one or two) are often plotted (like principal components)





QDA and other aspects

If we don't assume variance matrices equal, and fit them separately, then we obtain quadratic boundaries: quadratic discriminant analysis (QDA). Its results "may be strange".

We don't have to scale the variables - LDA is not vulnerable to it. In what sense: if we transform features linearly, the results of LDA transform accordingly.

Other aspects:

- we can also use transformed features as classifiers
- variable selection (selection of classifiers)

Implementation: functions `lda()` and `qda()` in `library(MASS)`. They use model formula notation: response is the class indicator, predictors are classifiers.

Regression interpretation

The connection to least squares

If there are only two classes (coded -1 and 1), and *if there is the same number of items in these classes*, then, if we run a linear regression of these classes (with the codes above) on the classifiers, the intersection of the fitted plane with level 0 (now we see why we need fixed codes) gives the separating plane for the linear discriminant analysis with equal prior probabilities and misclassification costs.

Even if the number of the points in the classes is not the same: the level zero set is parallel

The connection to logistic regression

That will be thoroughly discussed later; note now that the estimates "logarithmic scores" are in fact estimates of the logs of posterior probabilities (or proportional to those)

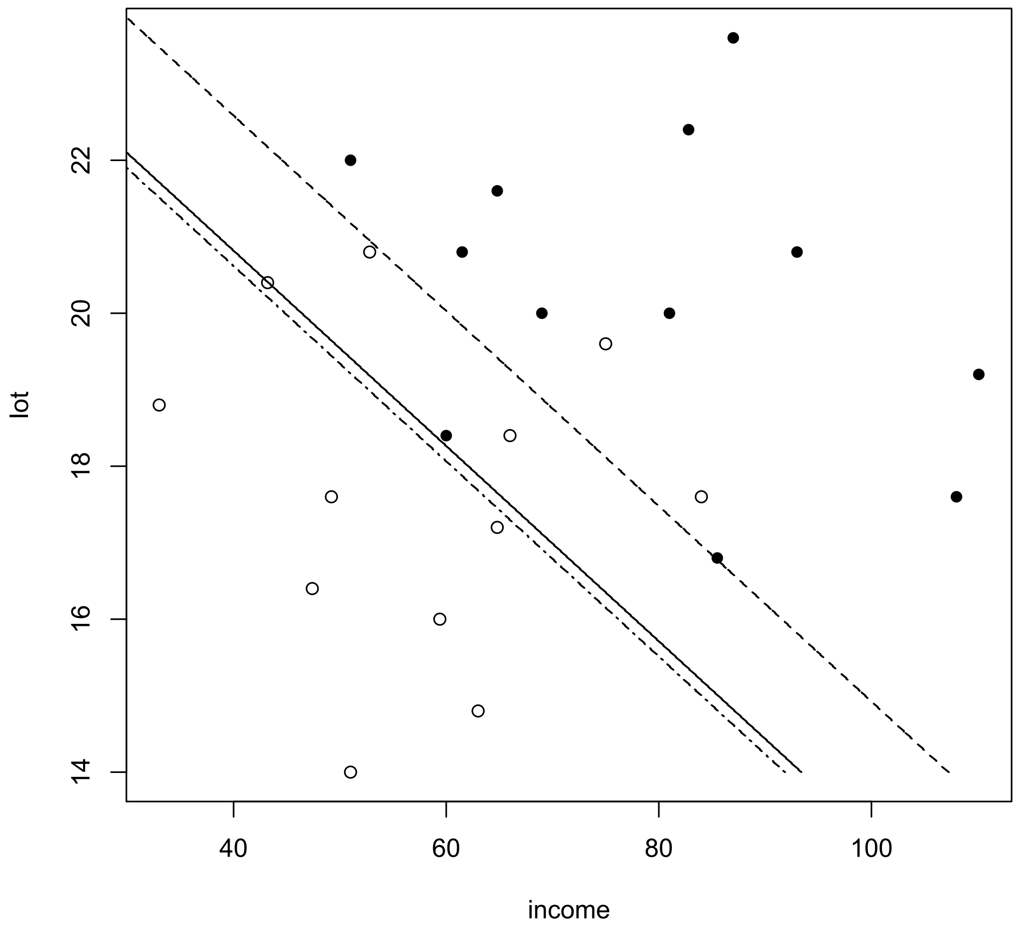
Also, R implementation draws on regression interpretation: formula notation, response, etc.

Mowers: LDA

Dashed line: prior estimated from the data (plug-in and predictive predictions equal)

Solid line: prior set to (0.2,0.8), plug-in prediction

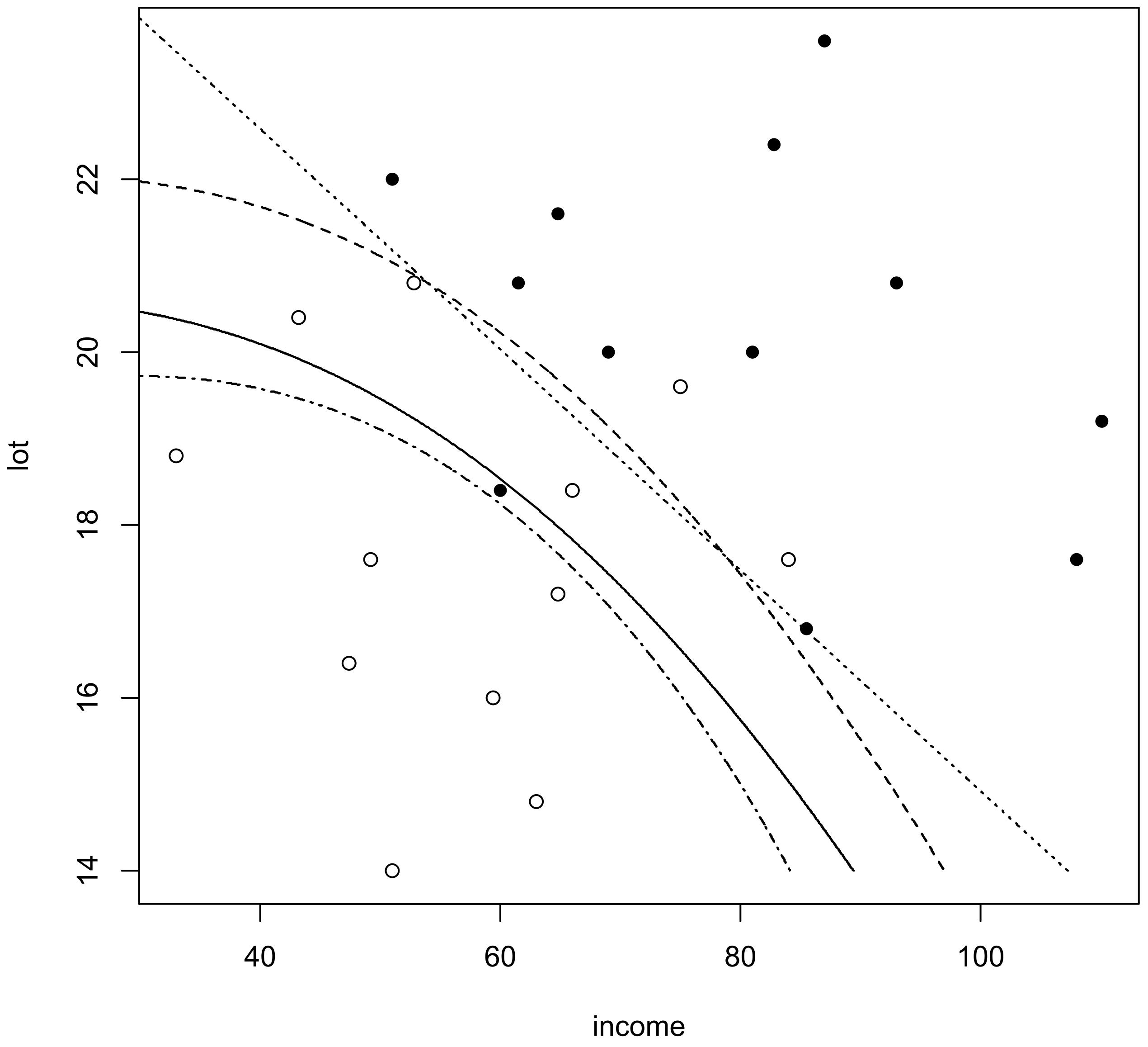
Dash-dotted line: prior set to (0.2,0.8), predictive prediction



Some code to try

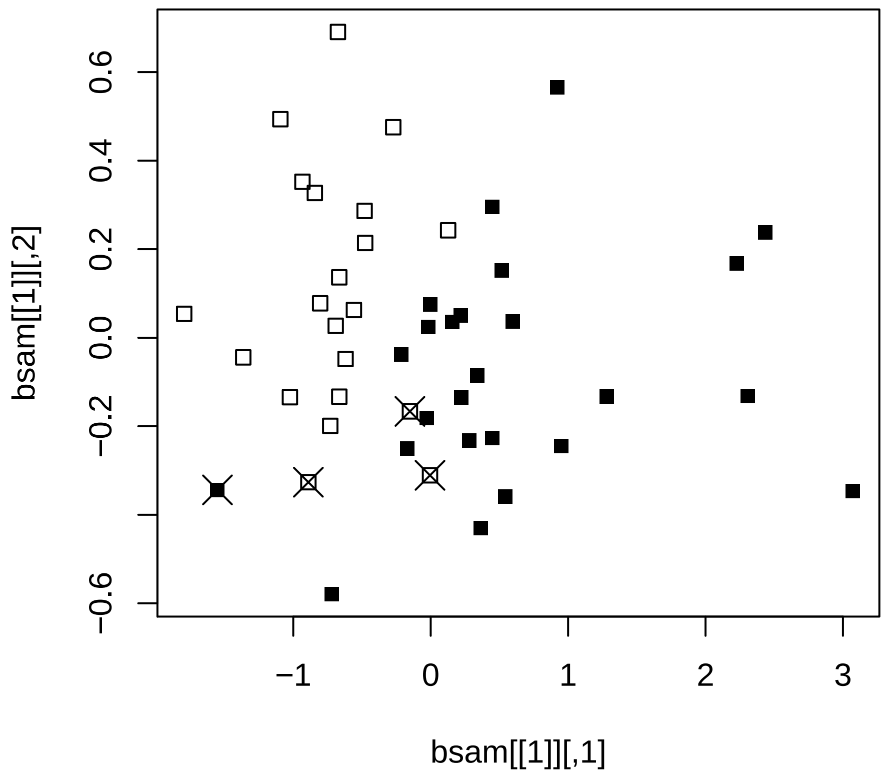
```
> library(MASS)
> ldob=lda(riding~income+lot,data=mowers)
> ldob
...
> predict(ldob)
...
> predict(ldob)$class
 [1] 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0
Levels: 0 1
> predict(ldob)$posterior
            0           1
1  0.78203155 0.217968446
2  0.49449211 0.505507885
3  0.15236751 0.847632493
4  0.31924493 0.680755073
5  0.00402325 0.995976750
...
> ?lda
```

Mowers: the same for QDA



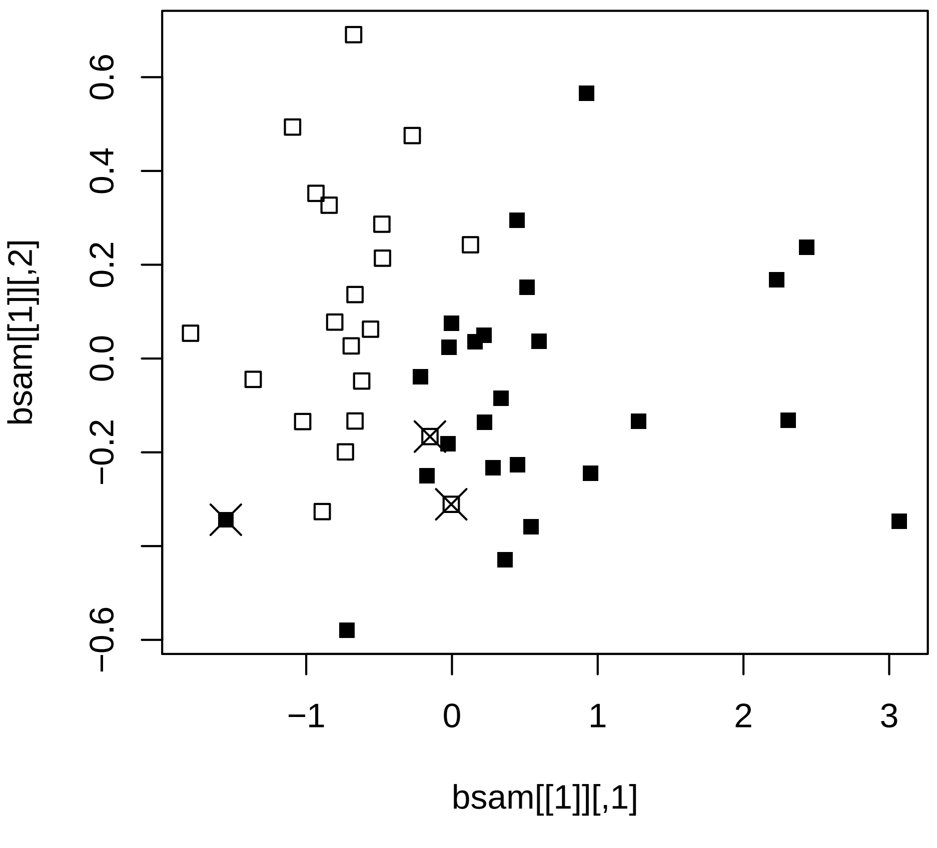
Bank data: LDA

```
> library(MASS)
> bsam=sammon(dist(Bank[,1:4]))
> plot(bsam[[1]],pch=15*Bank[,5])
> banlda=lda(k~.,data=Bank)
> wrong=Bank[,5] != predict(banlda)$class
> points(bsam[[1]][wrong,],pch=4,cex=2)
```

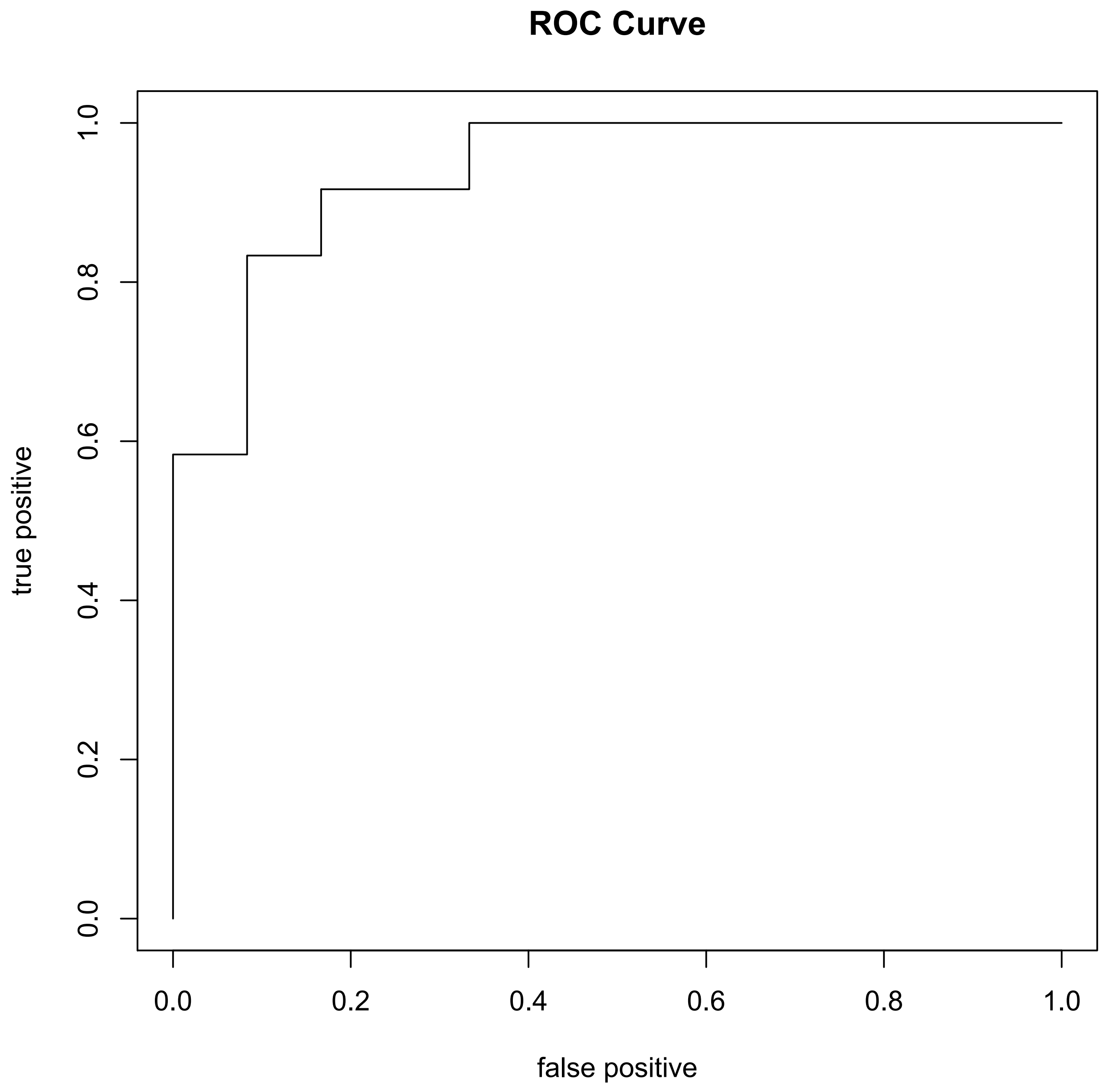


Bank data: QDA

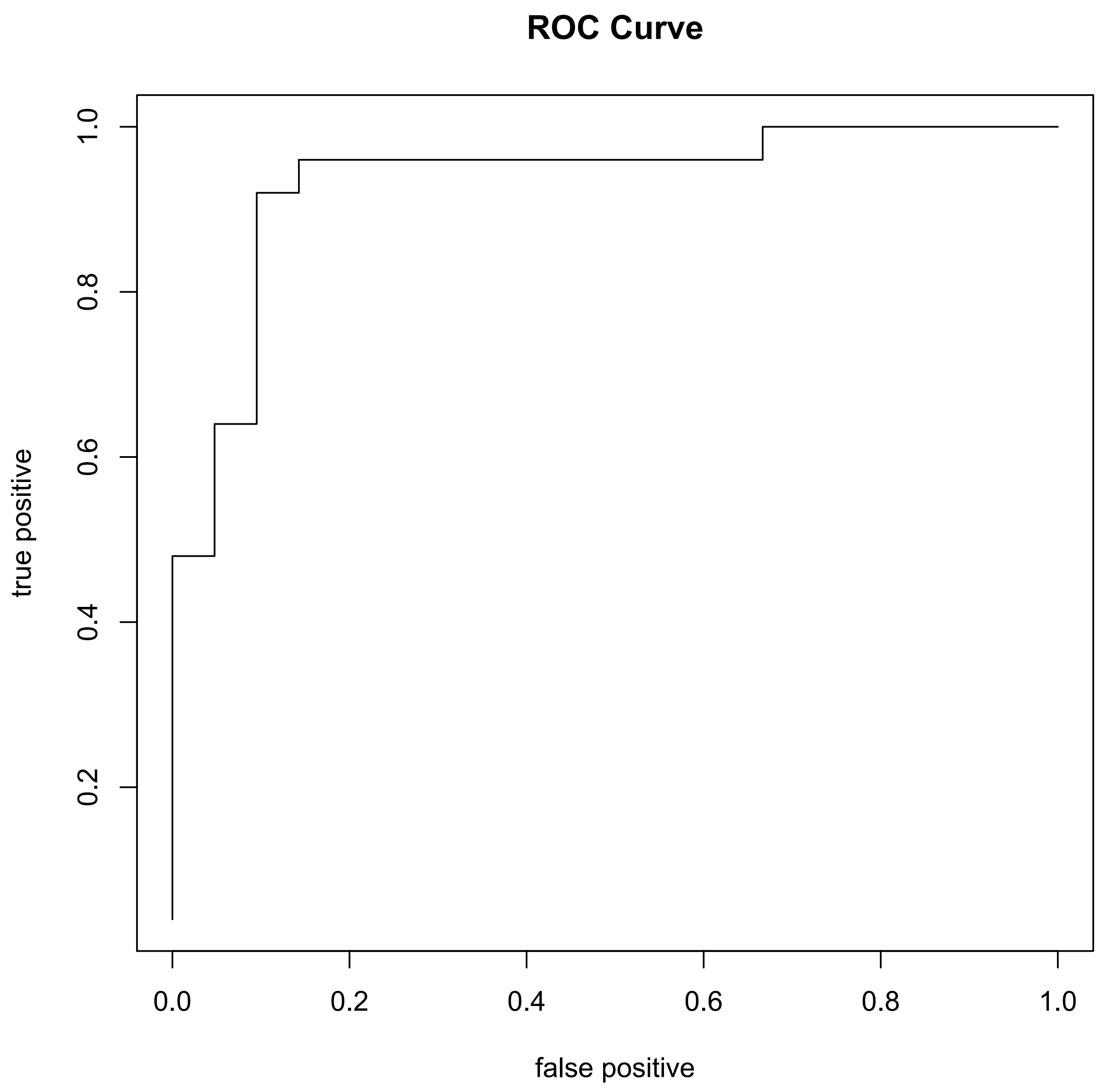
```
> bsam=sammon(dist(Bank[,1:4]))
> plot(bsam[[1]],pch=15*Bank[,5])
> banqda=qda(k~.,data=Bank)
> wrong=Bank[,5] != predict(banqda)$class
> points(bsam[[1]][wrong,],pch=4,cex=2)
```



Finally, promised ROC curves: mowers



And bank data


ROC Curve
true positive
false positive