Classification II

Yifei Sun

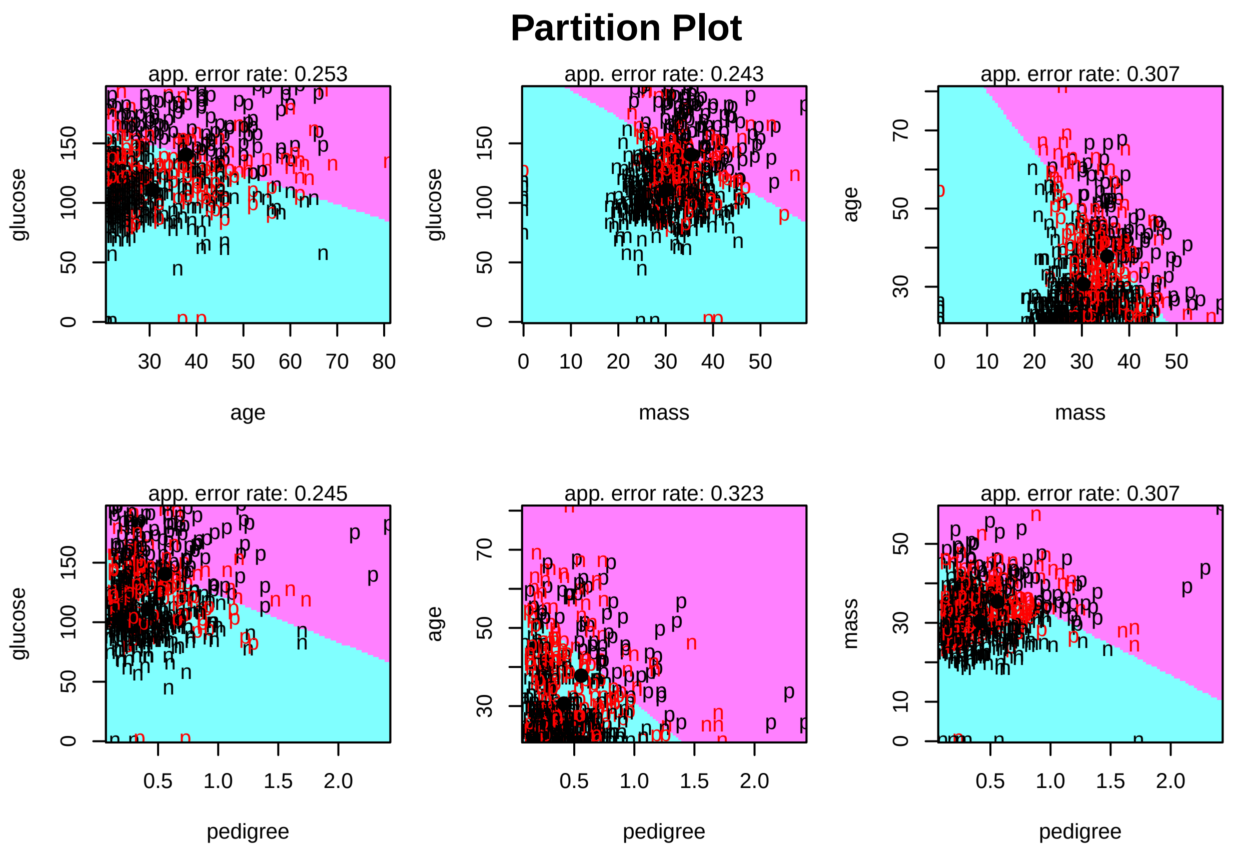
Contents

```
library(caret)
library(MASS)
library(mlbench)
library(pROC)
library(klaR)
```

Diabetes data

We use the Pima Indians Diabetes Database for illustration. The data contain 768 observations and 9 variables. The outcome is a binary variable `diabetes`. We start from some simple visualization of the data.

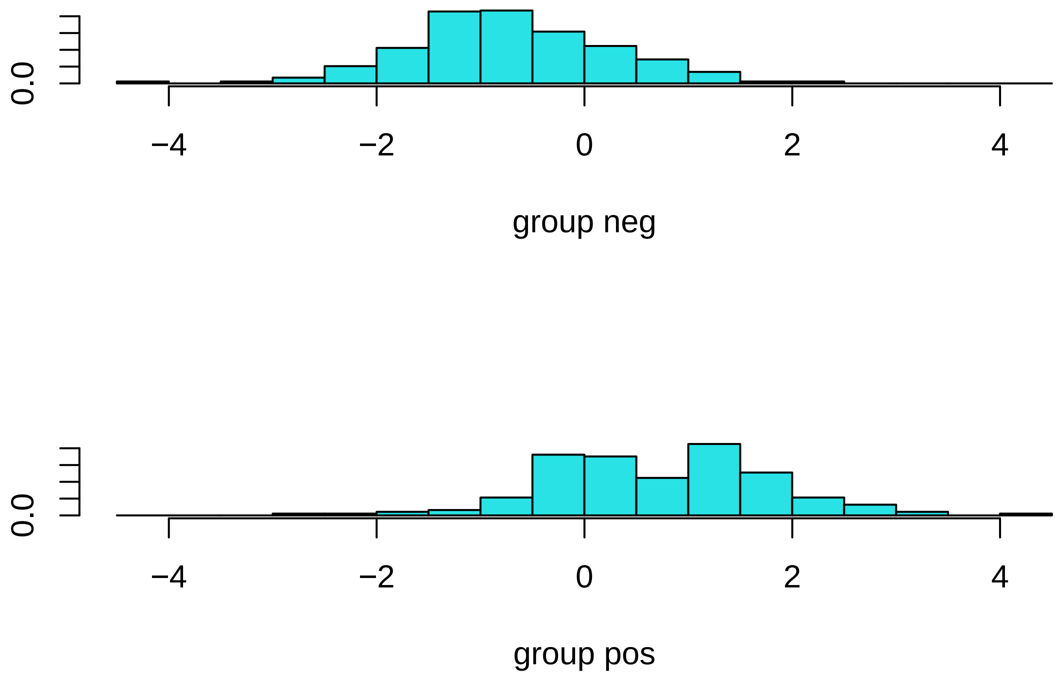
```
data(PimaIndiansDiabetes)
dat <- PimaIndiansDiabetes

set.seed(1)
rowTrain <- createDataPartition(y = dat$diabetes,
                                p = 0.7,
                                list = FALSE)

# Exploratory analysis: LDA based on every combination of two variables
partimat(diabetes ~ glucose + age + mass + pedigree,
         data = dat, subset = rowTrain, method = "lda")
```



LDA

We use the function `lda` in library `MASS` to conduct LDA.

```
lda.fit <- lda(diabetes~., data = dat,
               subset = rowTrain)
plot(lda.fit)
```



```
lda.fit$scaling
```

```
##                   LD1
## pregnant  0.085759790
## glucose   0.023406444
## pressure -0.008899919
## triceps  -0.006460031
## insulin  -0.000180940
## mass      0.066022727
## pedigree  1.053676313
## age       0.023618745
```

```
head(predict(lda.fit)$x)
```

```
##          LD1
## 1  1.384569
## 2 -1.628407
## 3  1.614858
## 4 -1.844059
## 5  2.842792
## 6 -0.691319
```

```
mean(predict(lda.fit)$x)
```

```
## [1] 3.846616e-16
```

```
dat_t <- dat[rowTrain,]
x_n_tr <- dat_t[dat_t$diabetes == "neg", 1:8]
x_p_tr <- dat_t[dat_t$diabetes == "pos", 1:8]
cov.neg <- cov(x_n_tr)
cov.pos <- cov(x_p_tr)
n.neg <- nrow(x_n_tr)
n.pos <- nrow(x_p_tr)
n <- n.neg + n.pos
K <- 2
W <- 1/(n - K) * (cov.neg * (n.neg - 1) + cov.pos * (n.pos - 1))
t(lda.fit$scaling) %*% W %*% lda.fit$scaling
```

```
##     LD1
## LD1   1
```

```
# head(as.matrix(dat[rowTrain,1:8]) %*% lda.fit$scaling -        mean(as.matrix(dat[rowTrain,1:8]) %*%
```

`(lda.fit$scaling) %*% W %*% lda.fit$scaling` equals to 1! Same as $a^T W a = 1$ sum of the posterior is one(Q: how to calculate posteriors by using linear discriminant?)

```
lda.pred <- predict(lda.fit, newdata = dat[-rowTrain,])
head(lda.pred$posterior)
```

```
##           neg       pos
## 11 0.79759173 0.2024083
## 13 0.08647245 0.9135276
## 18 0.81811921 0.1818808
## 19 0.71537570 0.2846243
## 20 0.78566312 0.2143369
## 21 0.62236228 0.3776377
```

Using caret:

```
ctrl <- trainControl(method = "repeatedcv", repeats = 5,
                     summaryFunction = twoClassSummary,
                     classProbs = TRUE)

set.seed(11)
model.lda <- train(x = dat[rowTrain,1:8],
                   y = dat$diabetes[rowTrain],
                   method = "lda",
                   metric = "ROC",
                   trControl = ctrl)
```

NO tuning parameters reduced rank LDA see `lda2()`

QDA

```
qda.fit <- qda(diabetes~., data = dat,
               subset = rowTrain)

qda.pred <- predict(qda.fit, newdata = dat[-rowTrain,])
head(qda.pred$posterior)
```

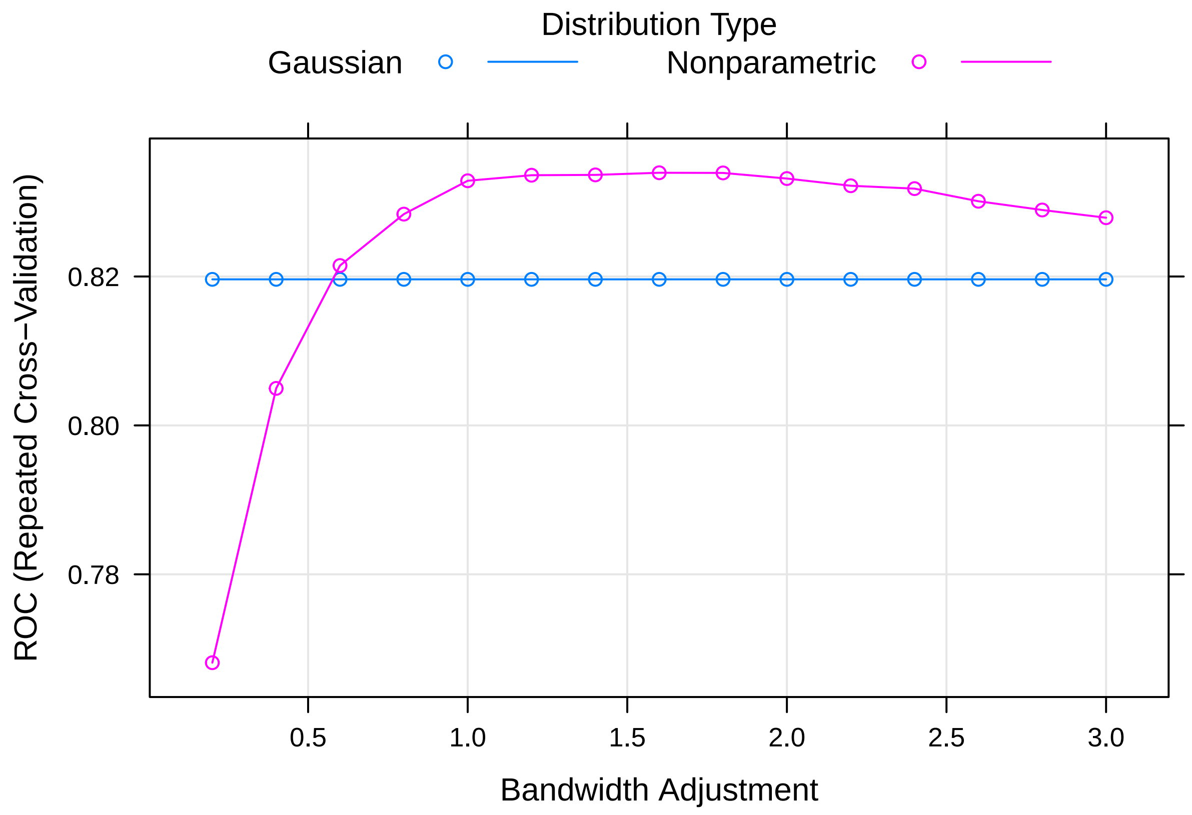
```
##           neg        pos
## 11 0.81996173 0.18003827
## 13 0.03707866 0.96292134
## 18 0.90103100 0.09896900
## 19 0.44435849 0.55564151
## 20 0.93494391 0.06505609
## 21 0.80361924 0.19638076
```

```
set.seed(11)
model.qda <- train(x = dat[rowTrain,1:8],
                   y = dat$diabetes[rowTrain],
                   method = "qda",
                   metric = "ROC",
                   trControl = ctrl)
```

Naive Bayes (NB)

There is one practical issue with the NB classifier when nonparametric estimators are used. When a new data point includes a feature value that never occurs for some response class, the posterior probability can become zero. To avoid this, we increase the count of the value with a zero occurrence to a small value, so that the overall probability doesn't become zero. In practice, a value of one or two is a common choice. This correction is called "Laplace Correction," and is implemented via the parameter `fL`. The parameter `adjust` adjusts the bandwidths of the kernel density estimates, and a larger value means a more flexible estimate.

```
nbGrid <- expand.grid(usekernel = c(FALSE,TRUE),
                      fL = 1,
                      adjust = seq(.2, 3, by = .2))

set.seed(11)
model.nb <- train(x = dat[rowTrain,1:8],
                  y = dat$diabetes[rowTrain],
                  method = "nb",
                  tuneGrid = nbGrid,
                  metric = "ROC",
                  trControl = ctrl)

plot(model.nb)
```

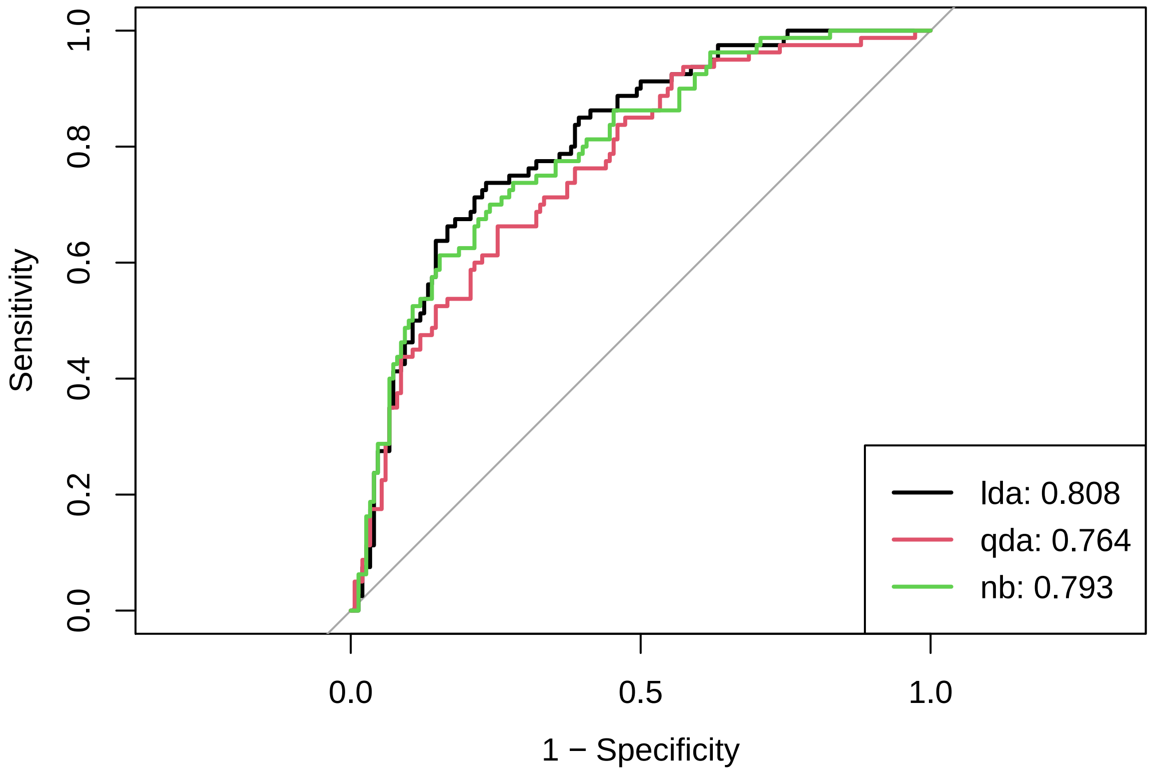


```
res <- resamples(list(LDA = model.lda, QDA = model.qda, NB = model.nb))
summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: LDA, QDA, NB
## Number of resamples: 50
##
## ROC
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.6601504 0.8056391 0.8383459 0.8371546 0.8796157 0.9488722    0
## QDA 0.6421053 0.7931704 0.8203008 0.8189791 0.8597327 0.9172932    0
## NB  0.6736842 0.8048872 0.8390977 0.8339332 0.8714286 0.9278195    0
##
## Sens
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.6857143 0.8357143 0.8714286 0.8691429 0.9142857 0.9714286    0
## QDA 0.6571429 0.8071429 0.8571429 0.8445714 0.8857143 0.9428571    0
## NB  0.7428571 0.8285714 0.8571429 0.8600000 0.8857143 0.9428571    0
##
## Spec
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.3157895 0.4802632 0.5409357 0.5642690 0.6315789 0.7894737    0
## QDA 0.3157895 0.5263158 0.5789474 0.5799415 0.6315789 0.7777778    0
## NB  0.2631579 0.5263158 0.5789474 0.5788889 0.6315789 0.8421053    0
```

Now let's look at the test set performance.

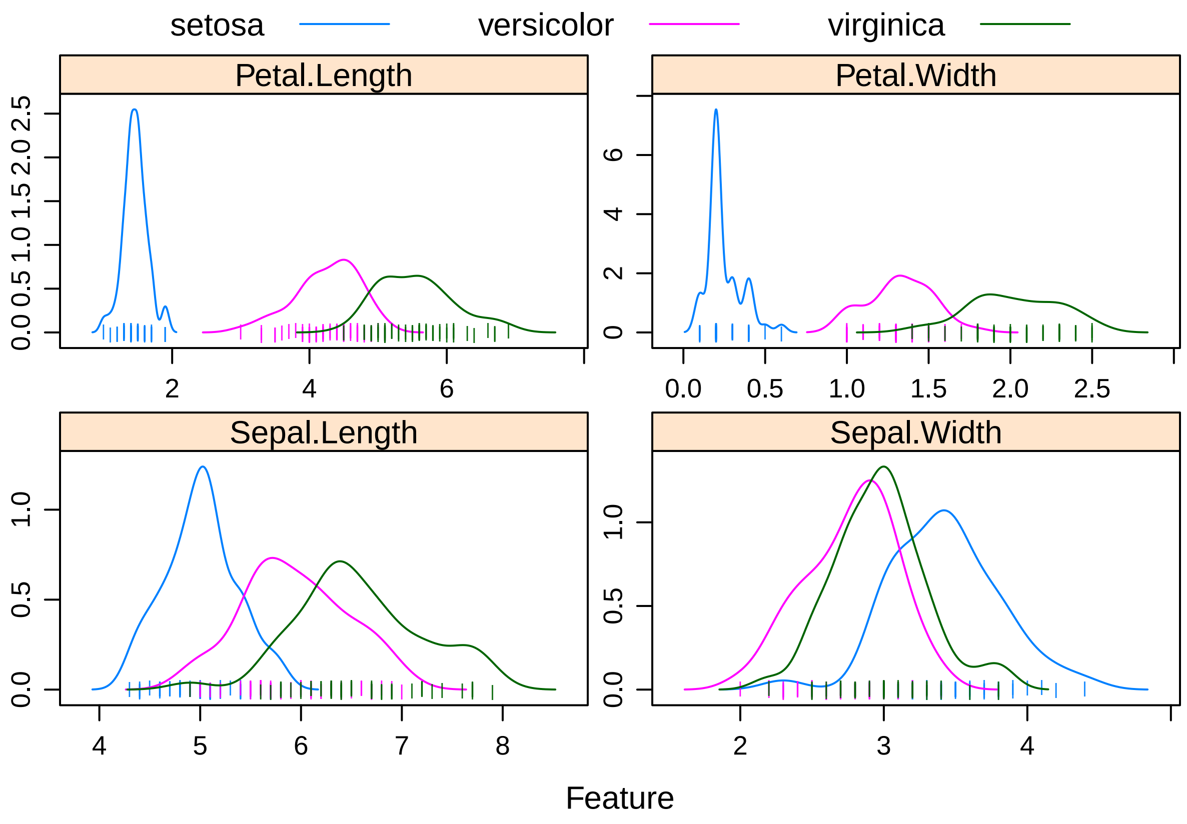
```
lda.pred <- predict(model.lda, newdata = dat[-rowTrain,], type = "prob")[,2]
nb.pred <- predict(model.nb, newdata = dat[-rowTrain,], type = "prob")[,2]
qda.pred <- predict(model.qda, newdata = dat[-rowTrain,], type = "prob")[,2]


roc.lda <- roc(dat$diabetes[-rowTrain], lda.pred)
roc.nb <- roc(dat$diabetes[-rowTrain], nb.pred)
roc.qda <- roc(dat$diabetes[-rowTrain], qda.pred)


auc <- c(roc.lda$auc[1], roc.qda$auc[1], roc.nb$auc[1])

plot(roc.lda, legacy.axes = TRUE)
plot(roc.qda, col = 2, add = TRUE)
plot(roc.nb, col = 3, add = TRUE)

modelNames <- c("lda","qda","nb")
legend("bottomright", legend = paste0(modelNames, ": ", round(auc,3)),
       col = 1:3, lwd = 2)
```



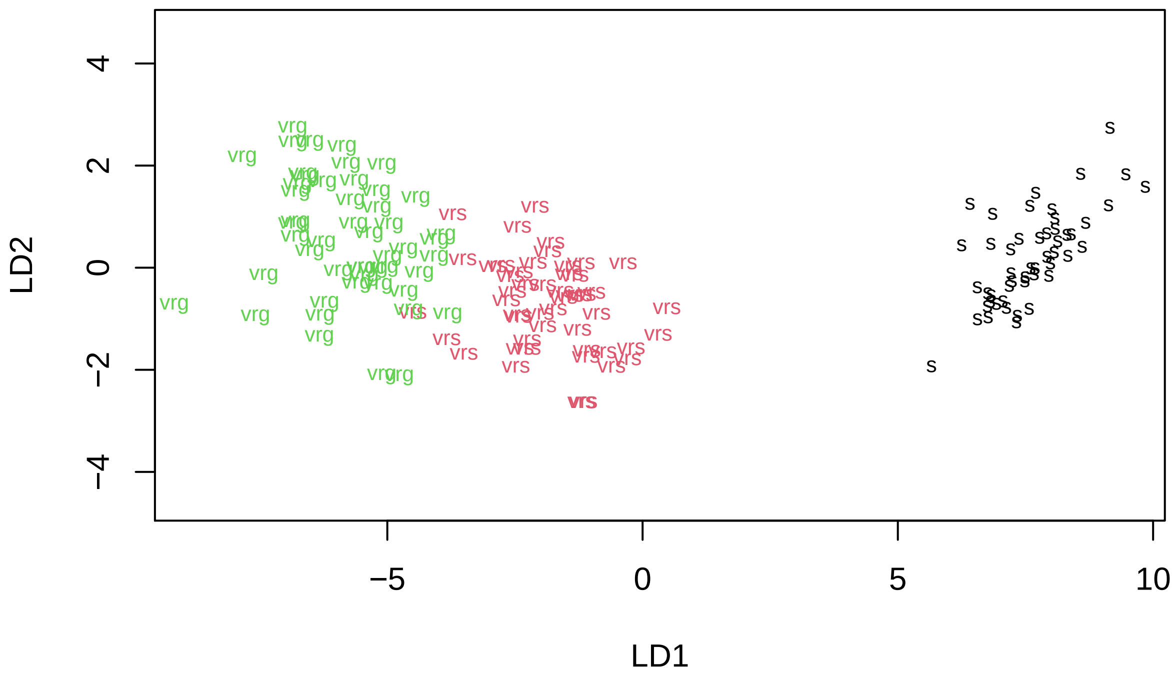
Iris data (K = 3)

The famous iris data!

```
data(iris)
dat2 <- iris

featurePlot(x = dat2[, 1:4],
            y = dat2$Species,
            scales = list(x=list(relation="free"),
                          y=list(relation="free")),
            plot = "density", pch = "|",
            auto.key = list(columns = 3))
```



```
lda.fit2 <- lda(Species~., data = dat2)
plot(lda.fit2, col = as.numeric(dat2$Species), abbrev = TRUE)
```



```r
ctrl2 <- trainControl(method = "cv") #try metirc = "Kappa"

set.seed(1)
model.lda2 <- train(x = dat2[,1:4],
                    y = dat2$Species,
                    method = "lda",
                    trControl = ctrl2)

set.seed(1)
model.qda2 <- train(x = dat2[,1:4],
                    y = dat2$Species,
                    method = "qda",
                    trControl = ctrl2)



res2 <- resamples(list(LDA = model.lda2,
                       QDA = model.qda2))
summary(res2)
```

```
##
## Call:
## summary.resamples(object = res2)
##
## Models: LDA, QDA
## Number of resamples: 10
##
## Accuracy
##          Min.   1st Qu. Median      Mean 3rd Qu. Max. NA's
## LDA 0.9333333 0.9500000      1 0.9800000       1    1    0
## QDA 0.9333333 0.9333333      1 0.9733333       1    1    0
##
## Kappa
##     Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
## LDA  0.9   0.925      1 0.97       1    1    0
## QDA  0.9   0.900      1 0.96       1    1    0
```