Linear/Quadratic DiscriminantAnalysis

Jai Vrat Singh

R Markdown

We try to test linear discriminant analysis using our own generated data. We will have two data sets with labels A and B with different means and common variance.

First generate two Multivariate normals with different means and variances.

```
getTransformMatrix <- function(covMat)
{
  svdRes <- svd(covMat)

  U <- svdRes$u
  D <- diag(svdRes$d)
  V <- svdRes$v
  A = U %*% (D^0.5)
  A
}


#--------------------  generate data -------------------------------#

vars1      <- c(10, 5)
vars2      <- c(4, 4)
#correlMat
correlMat1 <- rbind(c(1, 0.6),
                    c(0.6, 1))

correlMat2 <- rbind(c(1, -0.6),
                    c(-0.6, 1))


covMat1   <- diag(vars1) %*% correlMat1 %*% diag(vars1)
covMat2   <-  diag(vars2) %*% correlMat2 %*% diag(vars2)

MU1       <- matrix(c(1,3), ncol=1)
MU2       <- matrix(c(10,10), ncol=1)


N1 <- 300
N2 <- 300

getData <- function(N, MU, covMat)
{
  e <- matrix(rnorm(2*N), nrow = 2)
  matrix(MU, ncol = N, nrow=2) + getTransformMatrix(covMat) %*% matrix(rnorm(2*N), nrow = 2)
}

#generate data points  for each distribution
#1. Random Normals
```

```r
e <- matrix(rnorm(2*N1), nrow = 2)
#Data for fist set
D1 <- getData(N = N1, MU = MU1, covMat = covMat1)
D2 <- getData(N = N2, MU = MU2, covMat = covMat2)

dat_1 <- data.frame(x = D1[1,], y = D1[2,], "type" = "1")
dat_2 <- data.frame(x = D2[1,], y = D2[2,], "type" = "2")

bindedDat <- rbind(dat_1, dat_2)

bindedDat[["categ"]] = bindedDat$type == 1


#--------------------  generate data --------------------------------#
```

Data had been generated and stored in data.frame bindedDat.

```r
head(bindedDat)
```

```
##             x         y type categ
## 1   0.8486749 0.5449125    1  TRUE
## 2   8.7097415 7.2393959    1  TRUE
## 3   7.5472881 9.5400861    1  TRUE
## 4  15.2770263 7.8712350    1  TRUE
## 5  28.0762491 2.9219425    1  TRUE
## 6 -14.2930815 5.4210593    1  TRUE
```

```r
tail(bindedDat)
```

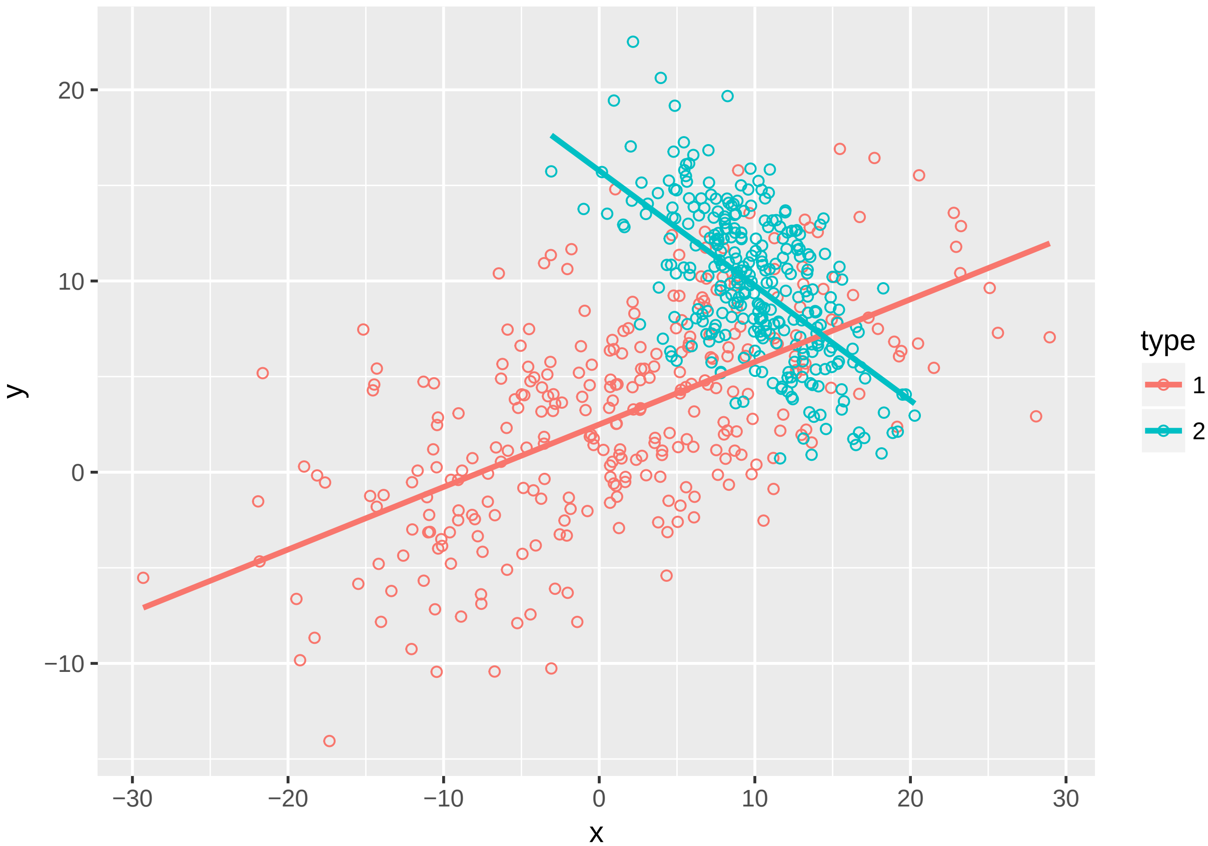
```
##              x         y type categ
## 595  7.7990152  9.511010    2 FALSE
## 596  9.3243441  9.423745    2 FALSE
## 597  8.7952496 11.171115    2 FALSE
## 598 13.7939466  2.899802    2 FALSE
## 599 10.4805477  8.063693    2 FALSE
## 600  0.9429654 19.436029    2 FALSE
```

```r
dim(bindedDat)
```

```
## [1] 600   4
```

The data looks like:

```r
library(ggplot2)
ggplot(bindedDat, aes(x=x, y=y, color=type)) + geom_point(shape=1) +
  geom_smooth(method=lm,   # Add linear regression lines
              se=FALSE)    # Don't add shaded confidence region
```



We try fitting in different models, where dependent variable is the **type** and expanatory variables are ** x ** and ** y **.

1. Logistic Regression

```
library(caTools)
set.seed(101)
sample = sample.split(bindedDat$categ, SplitRatio = .75)
train = subset(bindedDat, sample == TRUE)
test  = subset(bindedDat, sample == FALSE)

table(train$categ)
```

```
##
## FALSE  TRUE
##   225   225
```

```
table(test$categ)
```

```
##
## FALSE  TRUE
##    75    75
```

```
logistic.fit = glm(categ ~ x + y, data = train, family = binomial)
summary(logistic.fit)
```

```
##
```

```
## Call:
## glm(formula = categ ~ x + y, family = binomial, data = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.63538  -0.76653  -0.09399   0.56783   2.76932
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.80026    0.33036   8.476  < 2e-16 ***
## x           -0.11088    0.02080  -5.332 9.72e-08 ***
## y           -0.27910    0.03225  -8.655  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 623.83  on 449  degrees of freedom
## Residual deviance: 403.84  on 447  degrees of freedom
## AIC: 409.84
##
## Number of Fisher Scoring iterations: 5
```

```
fitted.results <- predict(logistic.fit, newdata=test, type='response')
#fitted.results

t <- table(fitted.results > 0.5, test$categ == 1)
#Accuracy
sum(diag(t))/sum(t)
```

```
## [1] 0.8266667
```

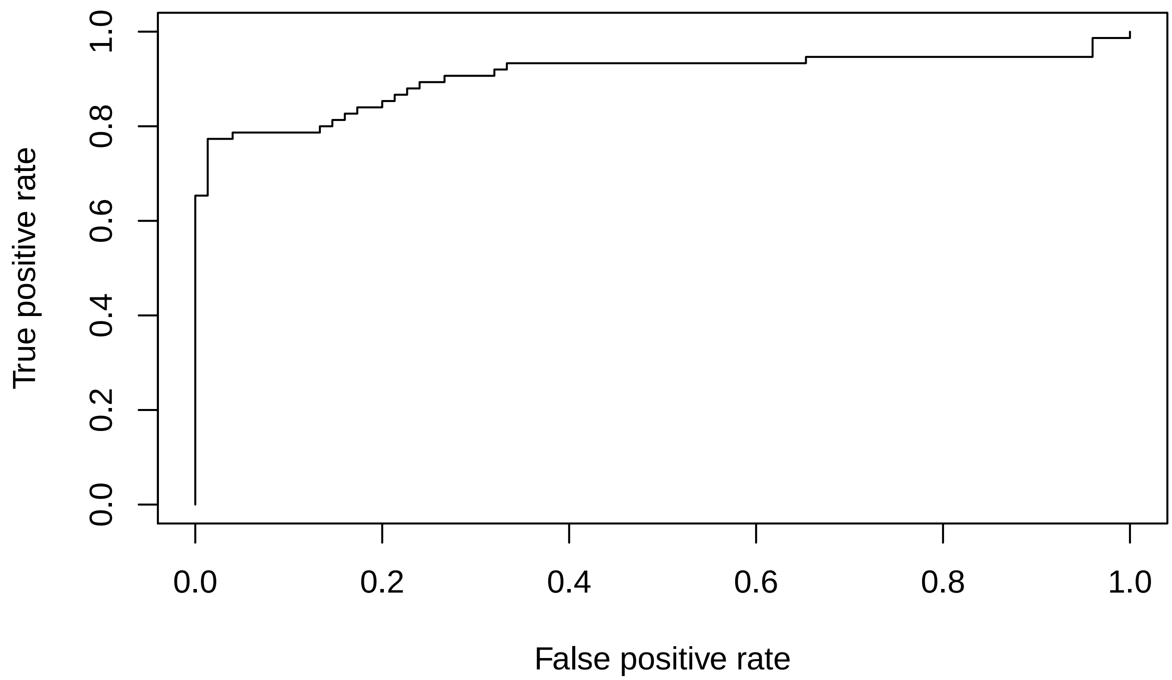
```
library(ROCR)
```

```
## Loading required package: gplots
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
p   <- predict(logistic.fit, newdata=test, type="response")
pr  <- prediction(p, test$categ)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```




```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.9052444
```

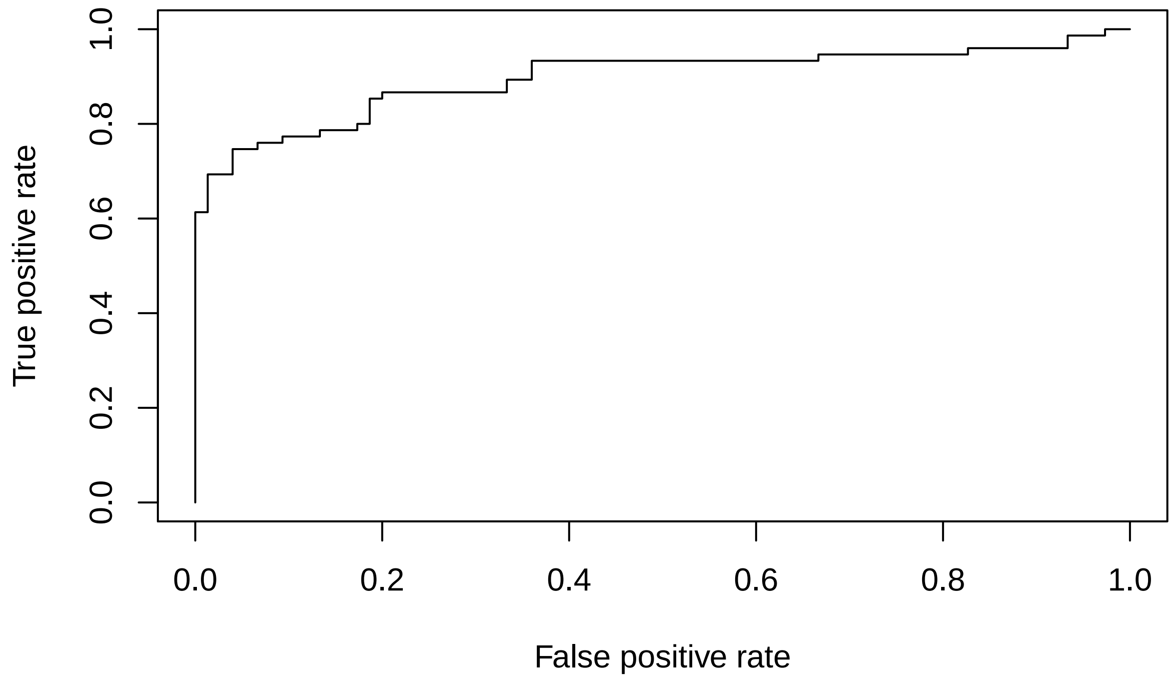
2. Linear Discriminant Analysis

```
library(MASS)
lda.fit = lda(categ ~ x + y ,data=train)
lda.fit
```

```
## Call:
## lda(categ ~ x + y, data = train)
##
## Prior probabilities of groups:
## FALSE  TRUE
##   0.5   0.5
##
## Group means:
##              x        y
## FALSE 9.734206 9.924138
## TRUE  1.607912 3.299335
##
## Coefficients of linear discriminants:
##           LD1
## x -0.05076938
## y -0.16966961
```

```
lda.pred=predict (lda.fit , test)
pr  <- prediction(lda.pred$posterior[,"TRUE"], test$categ)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
```

```r
plot(prf)
```

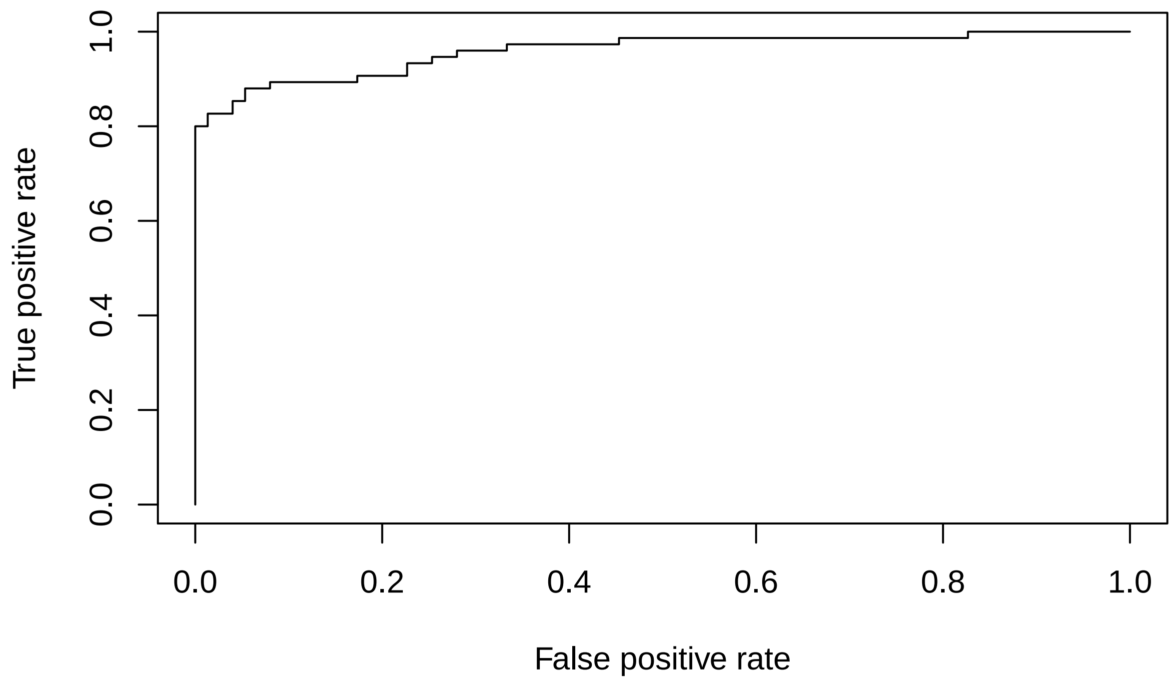



```r
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.8968889
```

3. Quadratic Discriminant Analysis

```r
qda.fit = qda(categ ~ x + y , data=train)
qda.pred=predict (qda.fit , test)
pr  <- prediction(qda.pred$posterior[,"TRUE"], test$categ)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```



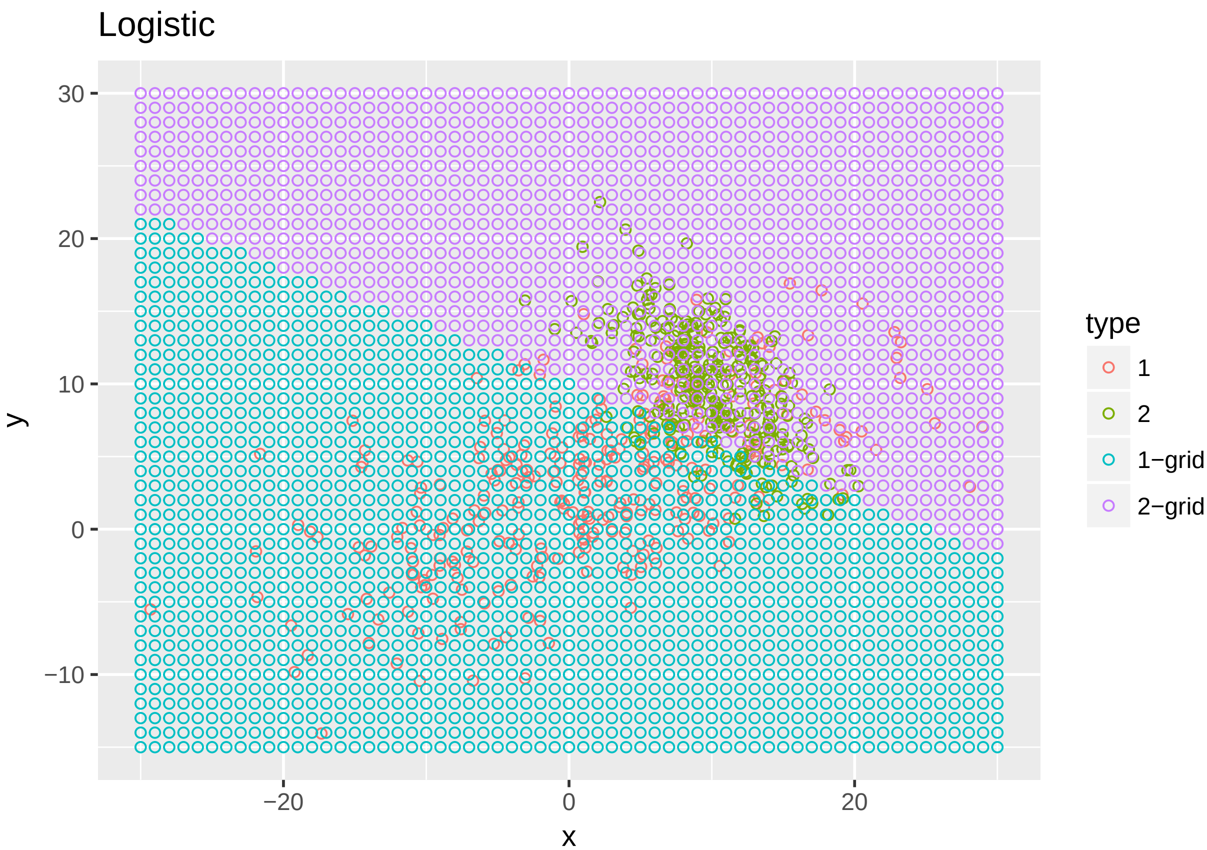

```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.9591111
```

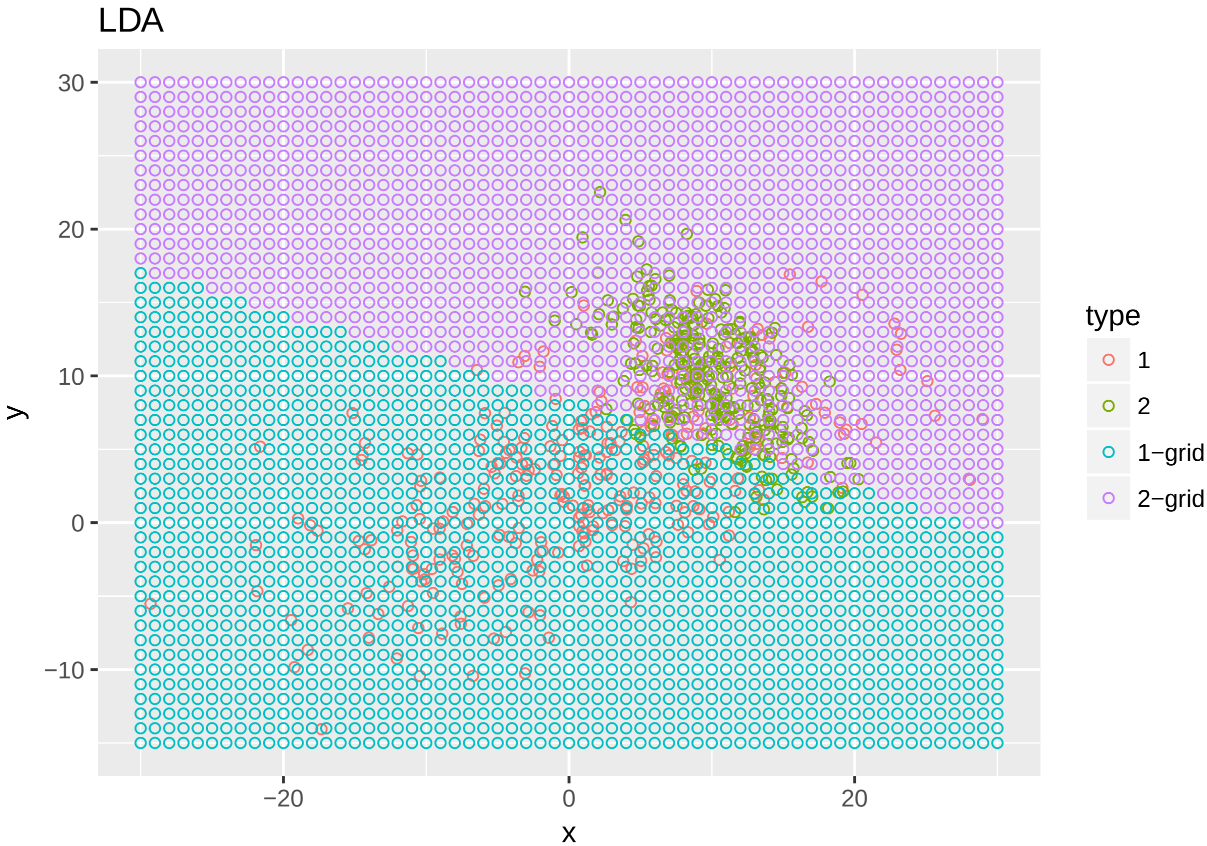
Boundaries : Logistic, LDA, QDA

```
x <- seq(from = -30, to = +30, by = 1)
y <- seq(from = -15, to = 30,  by = 1)

vals           <- expand.grid(x,y)
colnames(vals) = c("x", "y")
#head(vals)


#QDA
vals_qda <- vals;
qda.pred.grid = predict (qda.fit , vals_qda)
vals_qda[["type"]] = ifelse(qda.pred.grid$posterior[,2] >= 0.5, "1-grid", "2-grid")

#LDA
vals_lda <- vals
qda.pred.grid = predict (lda.fit , vals_lda)
vals_lda[["type"]] = ifelse(qda.pred.grid$posterior[,2] >= 0.5, "1-grid", "2-grid")

#Logistic
vals_logistic <- vals
logistic.pred.grid = predict(logistic.fit, newdata=vals_logistic, type='response')
vals_logistic[["type"]] = ifelse(logistic.pred.grid  >= 0.5, "1-grid", "2-grid")
```

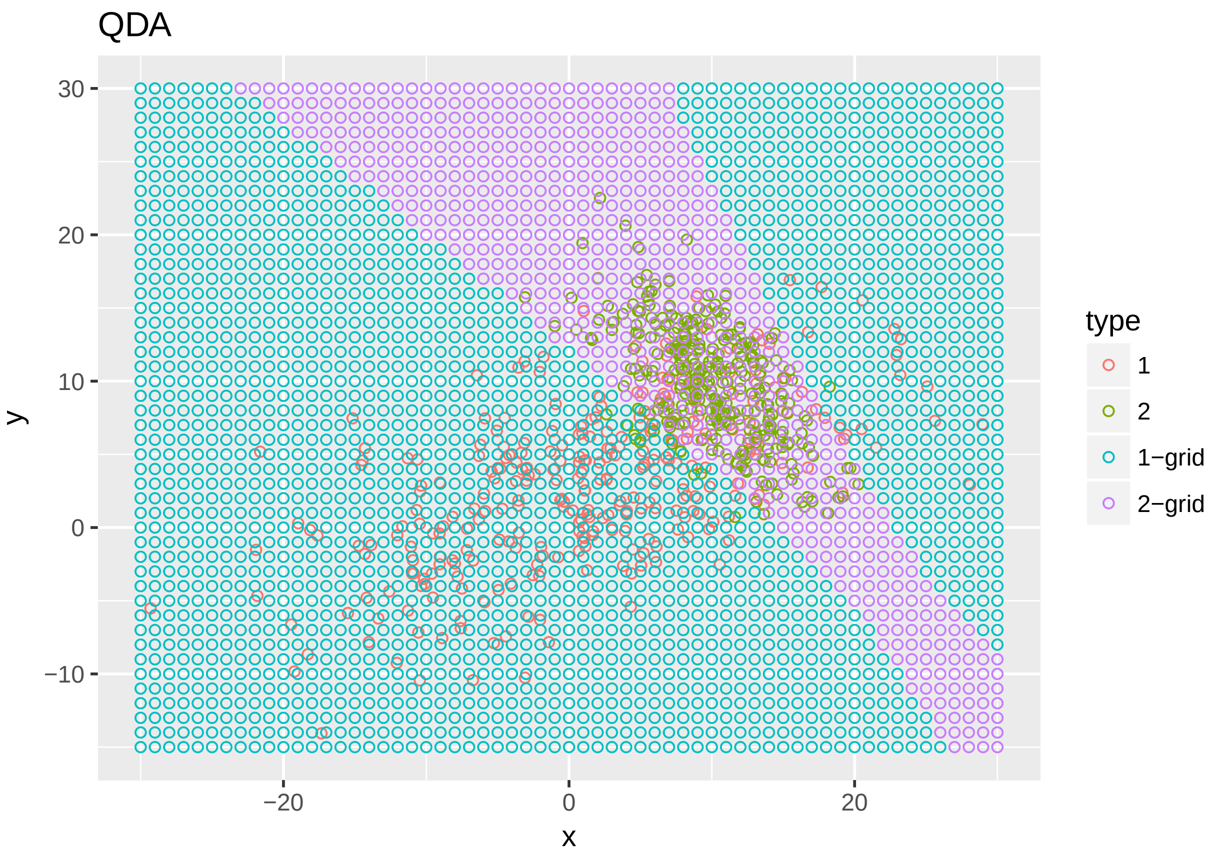
```
ggplot( rbind( bindedDat[,c("x", "y", "type")],
              vals_logistic), aes(x=x, y=y, color=type)) + geom_point(shape=1)  + ggtitle("Logistic")
```




```
ggplot( rbind( bindedDat[,c("x", "y", "type")],
              vals_lda), aes(x=x, y=y, color=type)) + geom_point(shape=1)  + ggtitle("LDA")
```




```
ggplot( rbind( bindedDat[,c("x", "y", "type")],
               vals_qda), aes(x=x, y=y, color=type)) + geom_point(shape=1)  + ggtitle("QDA")
```


QDA
30
20
10
0
−10
y
−20
0
20
x
type
1
2
1−grid
2−grid