Damien Druel

TP 6 : Analyse discriminante

Introduction

Dans ce TP, nous allons voir deux méthodes pour classifier les éléments d’une image en plusieurs classes, l’analyse linéaire discriminante et l’analyse quadratique discriminante. Nous allons essayer ses deux méthodes sur des échantillons, voir leur efficacité et les comparer entre elles.

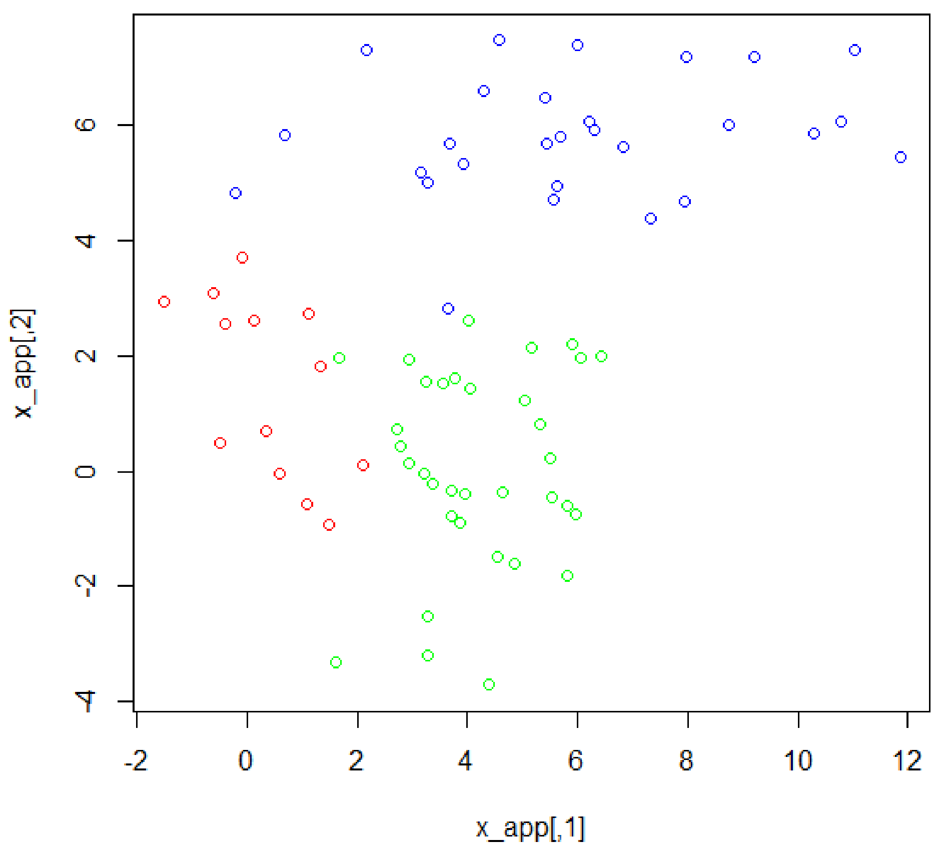
Damien Druel

1. Classication de donnees gaussiennes

1.1 Q1 : Achage des observations

Apprentissage

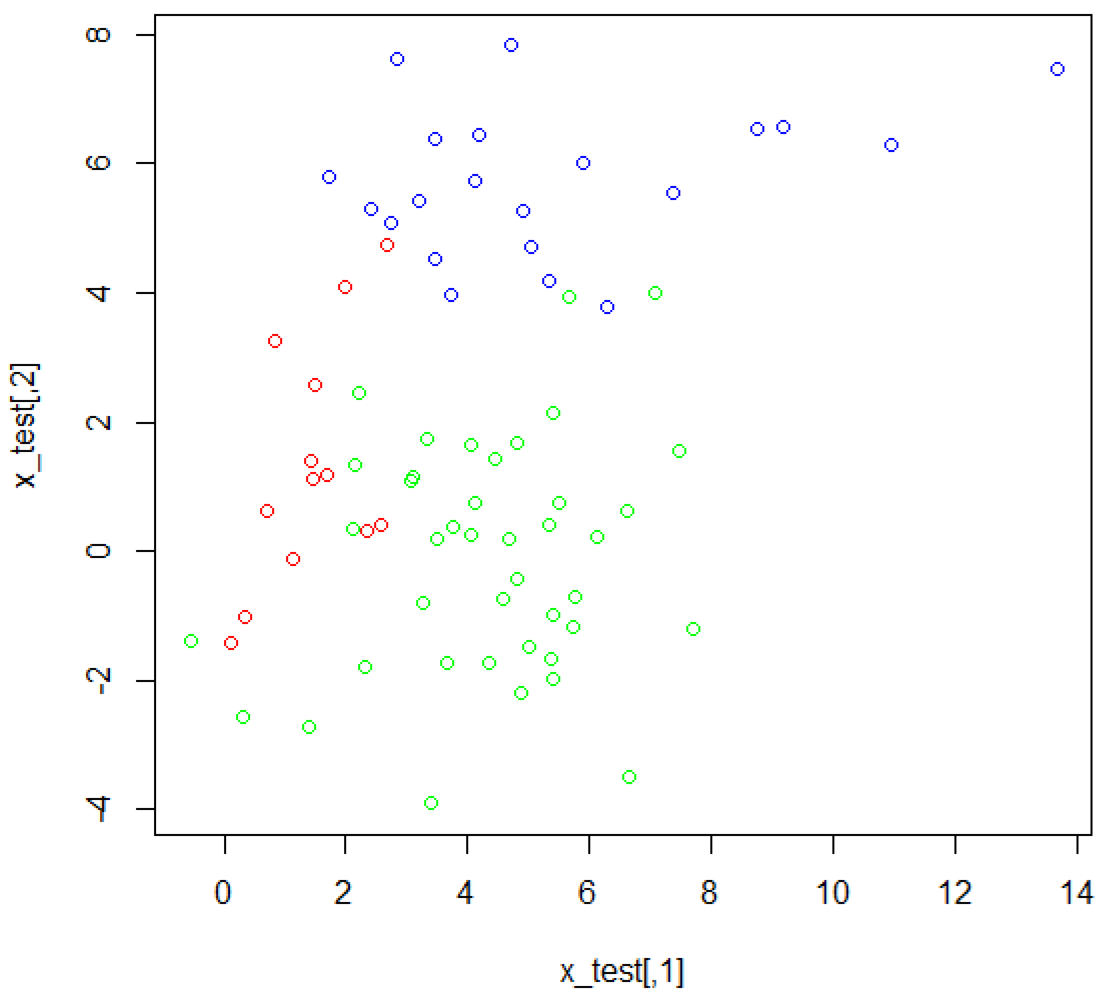
```
#app
couleur<-rep('red',n_app);  #classe 1 rouge
couleur[classe_app==2]='blue'; #classe 2 bleue
couleur[classe_app==3]='green'; #classe 3 verte
plot(x_app, col = couleur); #affichage
```



Damien Druel

Test

```
#test
couleur<-rep('red',n_test);  #classe 1 rouge
couleur[classe_test==2]='blue'; #classe 2 bleue
couleur[classe_test==3]='green'; #classe 3 verte
plot(x_test, col = couleur); #affichage
```



1.2 Q2: Estimation des moyennes et co-variance

1.2.1 Moyennes

```
M1[1] = mean(x_app[classe_app==1,1])
M1[2] = mean(x_app[classe_app==1,2])
M2[1] = mean(x_app[classe_app==2,1])
M2[2] = mean(x_app[classe_app==2,2])
M3[1] = mean(x_app[classe_app==3,1])
M3[2] = mean(x_app[classe_app==3,2])
```

resultat

	[1]	[2]
M1	0.3901397	1.483721
M2	5.979027	5.815149
M3	4.192731	0.05827418

m

	[1]	[2]
m1	1	2
m2	6	6
m3	4	0

Les valeurs sont très proches, il y a juste un petit écart de précision à cause des potentielles erreurs.

1.2.1 Co-Variance

```
Sigma1[1] = sqrt(cov(as.vector(x_app[classe_app==1,1]),as.vector(x_app[classe_app==1,1])))
Sigma1[2] = sqrt(cov(as.vector(x_app[classe_app==1,2]),as.vector(x_app[classe_app==1,2])))
Sigma2[1] = sqrt(cov(as.vector(x_app[classe_app==2,1]),as.vector(x_app[classe_app==2,1])))
Sigma2[2] = sqrt(cov(as.vector(x_app[classe_app==2,2]),as.vector(x_app[classe_app==2,2])))
Sigma3[1] = sqrt(cov(as.vector(x_app[classe_app==3,1]),as.vector(x_app[classe_app==3,1])))
Sigma3[2] = sqrt(cov(as.vector(x_app[classe_app==3,2]),as.vector(x_app[classe_app==3,2])))
```

Resultat

	[1]	[2]
Sigma1	1.017921	1.569878
Sigma2	3.025857	1.078742
Sigma3	1.262385	1.708196

S

	[1]	[2]
s1	1	2
s2	3	1
s3	1.5	2

Les valeurs sont très proches, il y a juste un petit écart de précision à cause des potentielles erreurs.

1.3 Q3 : Analyse lineaire discriminante

```
##### Definition de la grille
# Grille d'estimation de la densité de probabilité en 50 intervalles selon 1er attribut
xp1<-seq(min(x_test[,1]),max(x_test[,1]),length=50)
# Grille d'estimation de la densité de probabilité en 50 intervalles selon 2eme attribut
xp2<-seq(min(x_test[,2]),max(x_test[,2]),length=50)
grille<-expand.grid(x1=xp1,x2=xp2)
```

```
x_app.qda<- qda(x_app,classe_app)  ##### Algo discriminant ######
Zp<-predict(x_app.qda, grille)
assigne_test<-predict(x_app.qda, newdata=x_test)
# Estimation des taux de bonnes classifications
table_classification_test <-table(classe_test, assigne_test$class)
# table of correct class vs. classification
diag(prop.table(table_classification_test, 1))
# total percent correct
taux_bonne_classif_test <-sum(diag(prop.table(table_classification_test)))
```

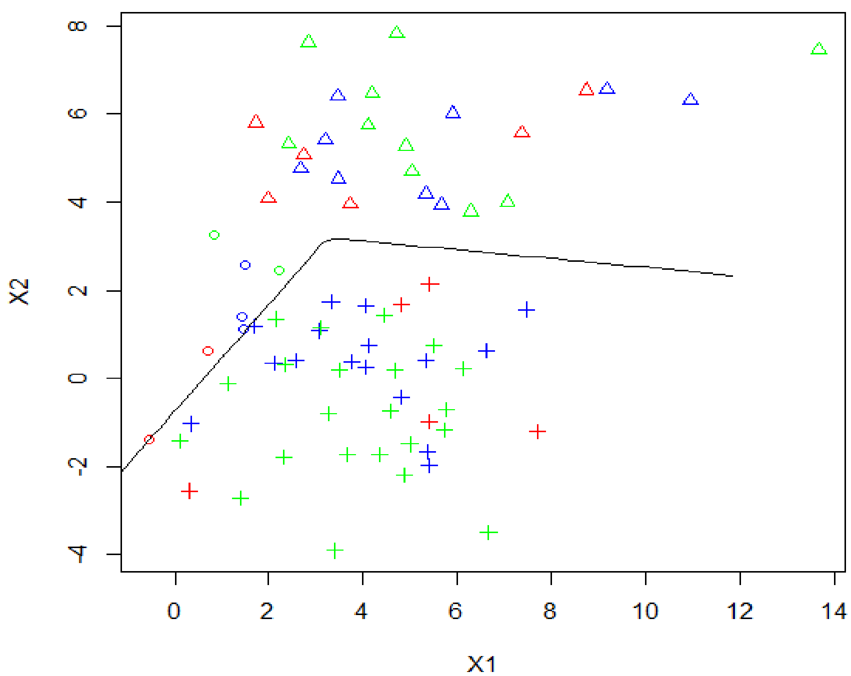
```r
##### Formes pour classe test
# forme de la classe test 1
shape<-rep(1,n_test) ;
# forme de la classe test 2
shape[assigne_test$class==2]=2 ;
# forme de la classe test 3
shape[assigne_test$class==3]=3 ;
```

```r
##### Couleur pour classe app
# couleur de la classe app 1
couleur<-rep('red',n_app);
# couleur de la classe app 2
couleur[classe_app==2]='blue';
# couleur de la classe app 3
couleur[classe_app==3]='green';
```

```r
##### affichage du resultat
plot(x_test,col=couleur,pch=shape,xlab = "X1", ylab = "X2")
```

taux_bonne_classif_test = 0.84

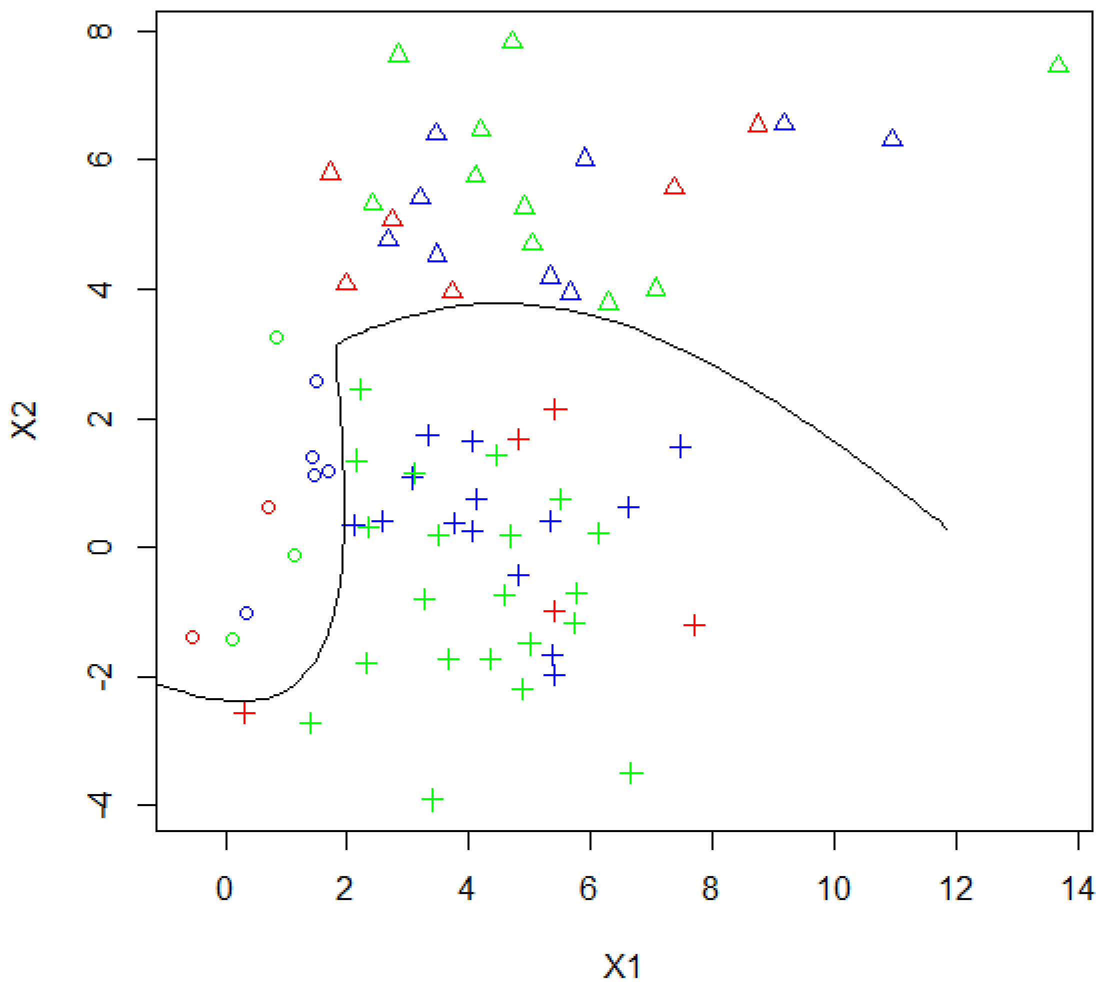
~~Tracer la fonction de decision gk~~



Damien Druel

1.4 Q4 : Analyse quadratique discriminante

taux_bonne_classif_test = 0.9066667

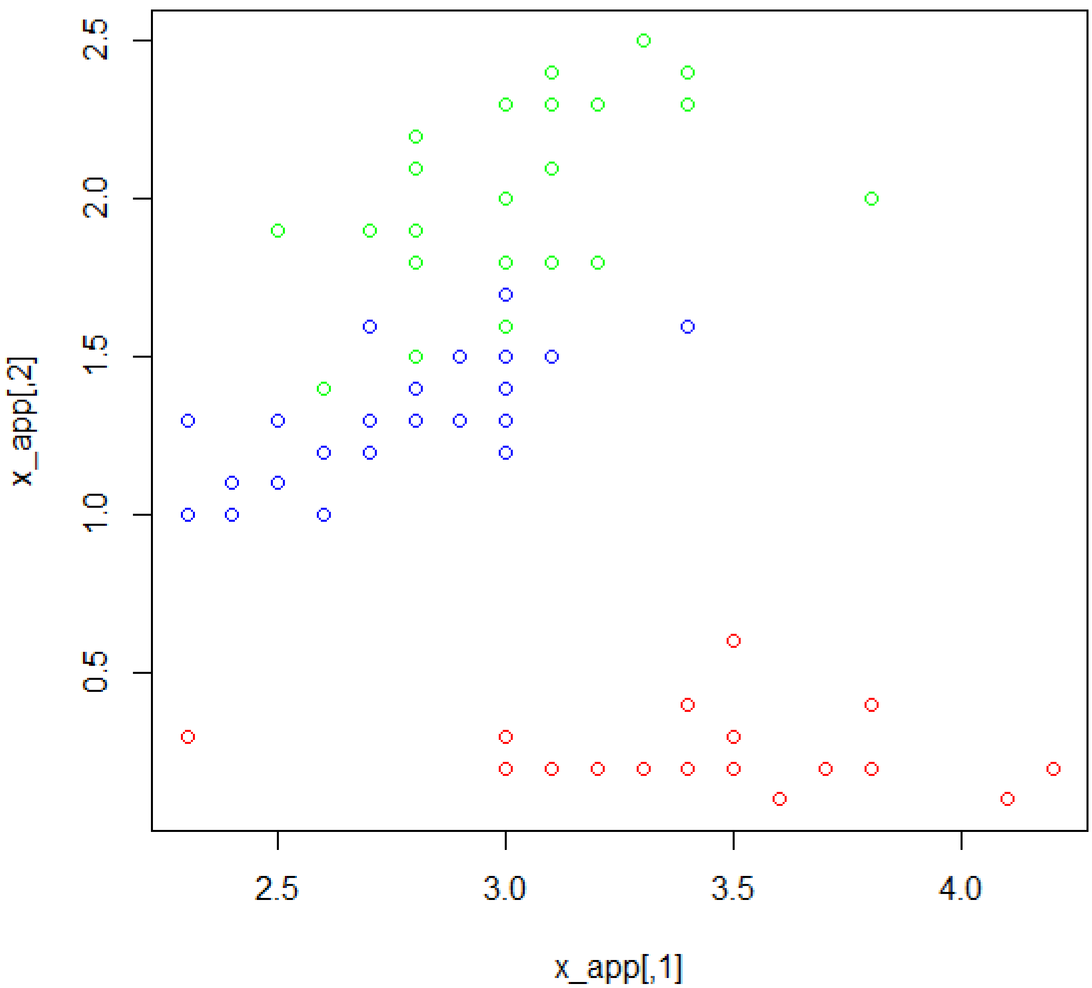


Le taux de bonne classification de l'**analyse quadratique est meilleur que** celui de l'**analyse linéaire** (0.84<0.90) donc il y a un peu plus d'erreurs dans l'analyse quadratique. La fonction de décision n'a pas le même tracé mais est semblable.

Damien Druel

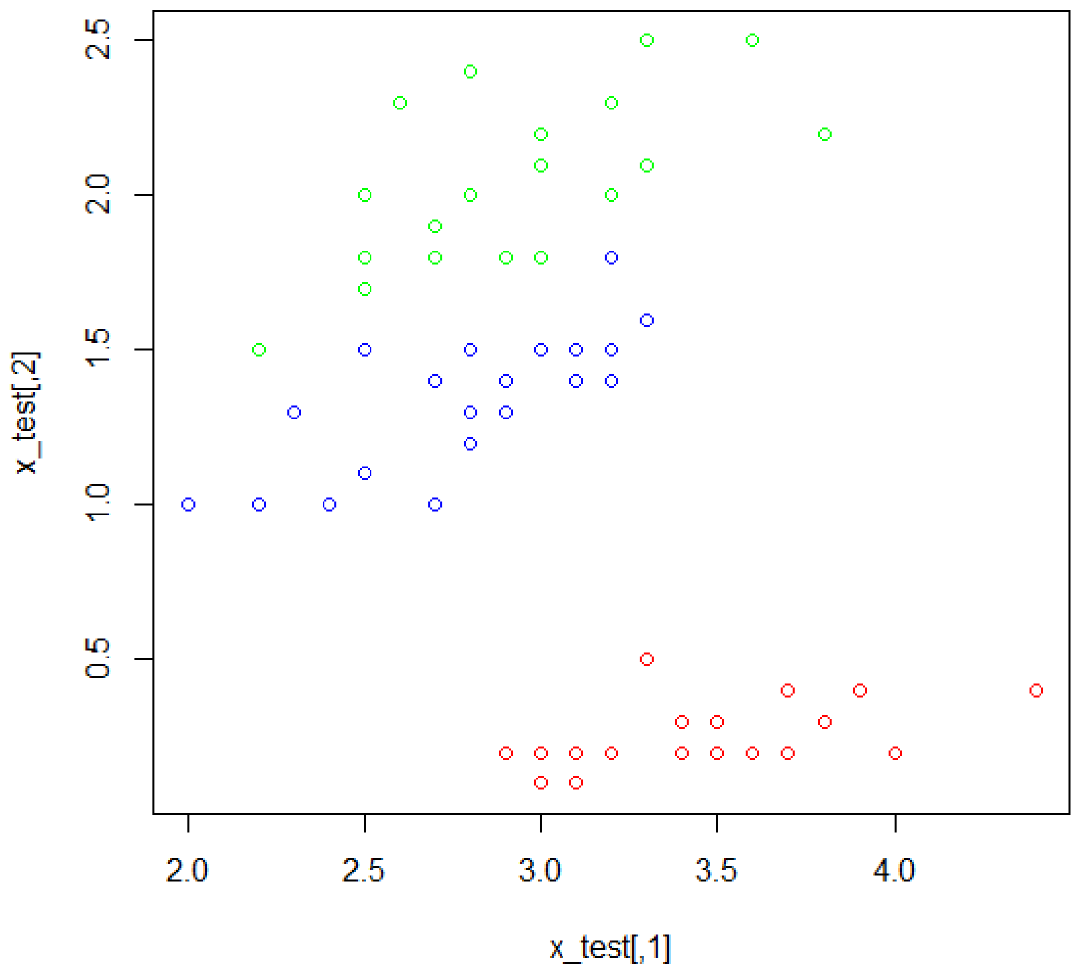
2 Classication de donnees reelles : Iris

Apprentissage



Damien Druel

<u>Test</u>

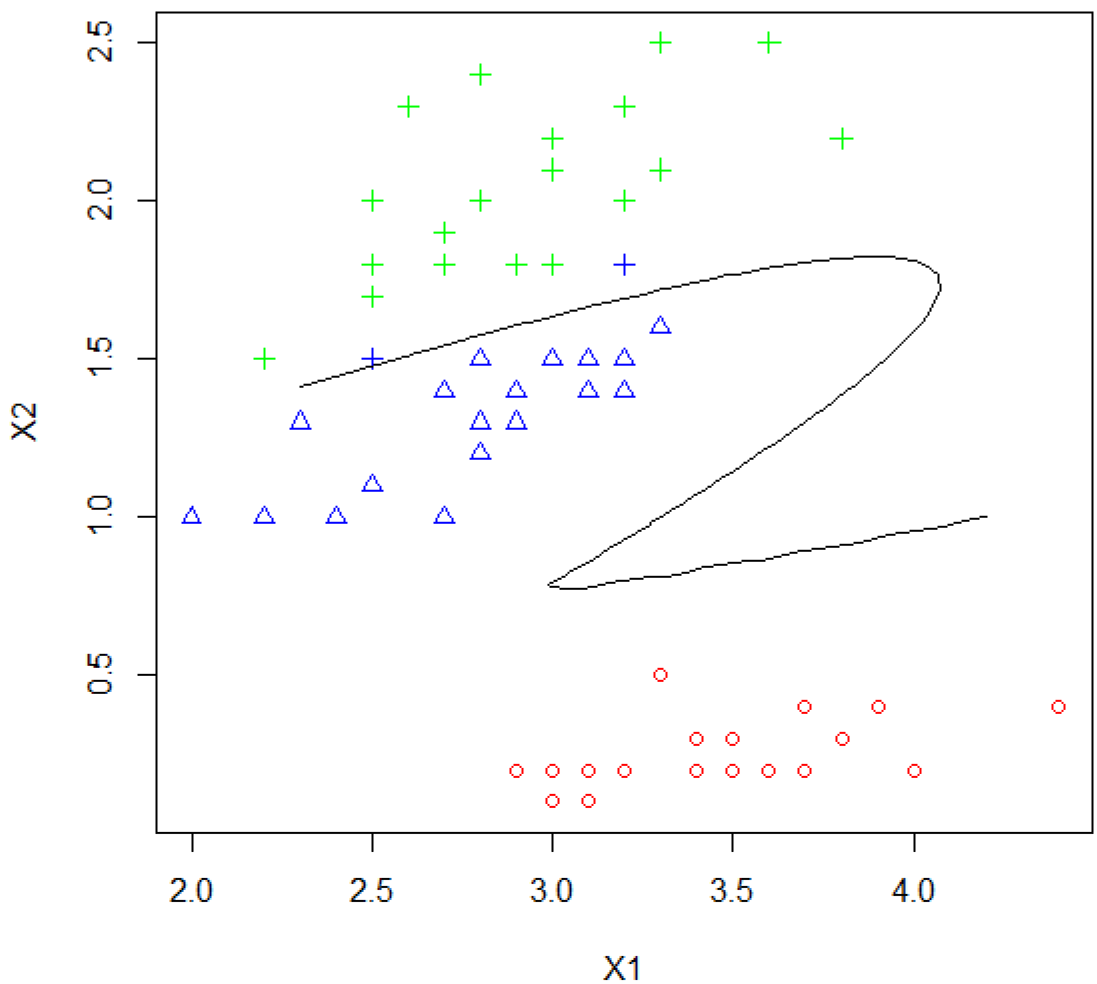


Damien Druel

2.1 Q5 : Analyse discriminante

<u>Quadratique (qda)</u>

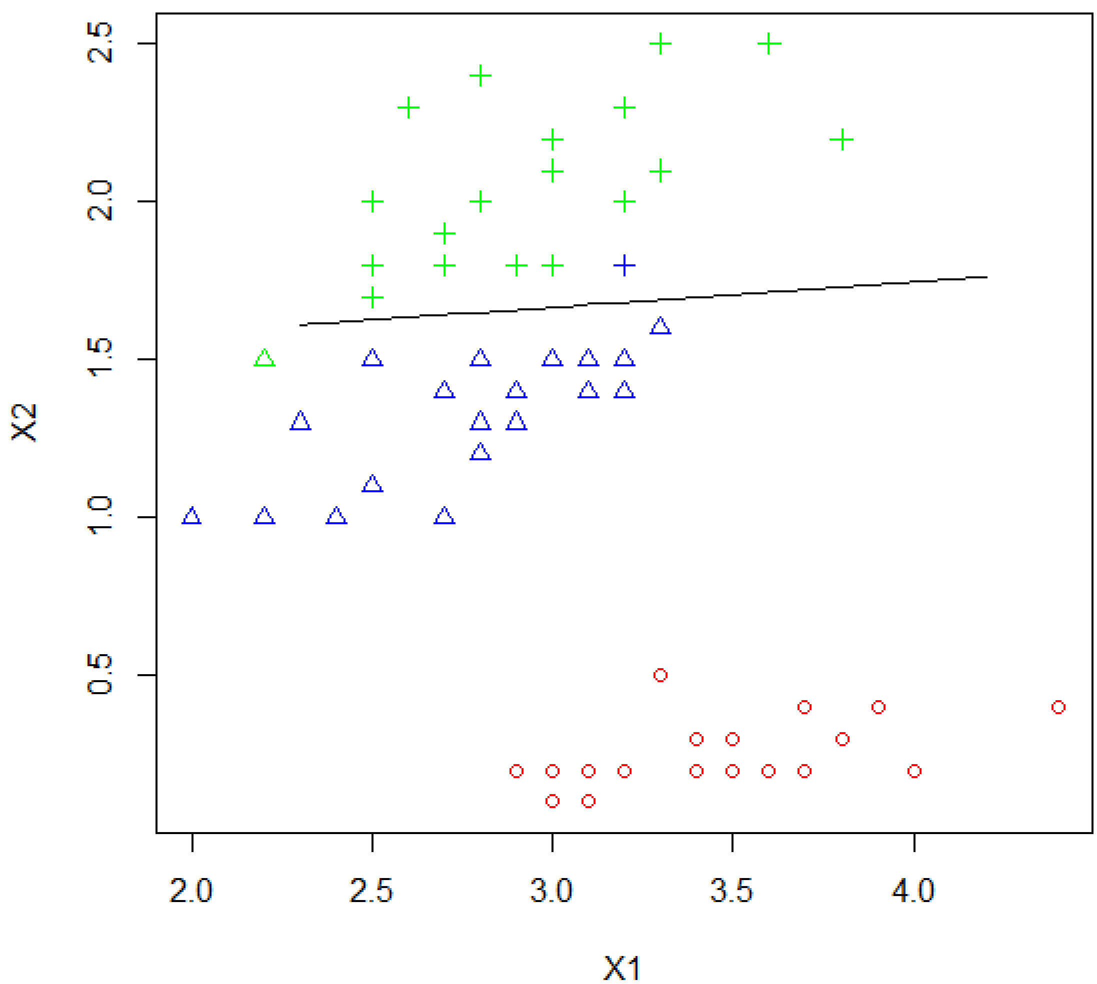
taux_bonne_classif_test = 0.96



Damien Druel

Linéaire (lda)

taux_bonne_classif_test = 0.9733333



Le taux de bonne classification de l'**analyse Linéaire est meilleur que** celui de l'**analyse Quadratique**. La fonction de décision quand a elle a un résultat inattendu (ell ne sépare pas la dernière classe peut être parce que la séparation est trop notable à l'oeil nu)

Damien Druel

2.2 Q6 : Echange des donnees test et apprentissage

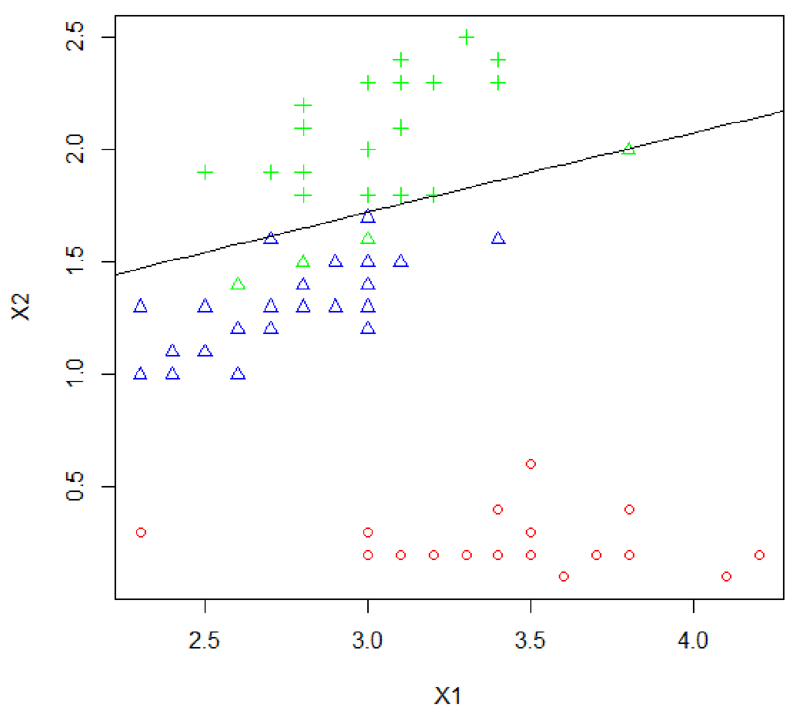
```
##### echange des donnees
classe_tmp <- classe_app;
classe_app <- classe_test;
classe_test <- classe_tmp;

n_tmp <- n_app;
n_app <- n_test;
n_test <- n_tmp;

x_tmp <- x_app;
x_app <- x_test;
x_test <- x_tmp;
```

<u>Quadratique (qda)</u>

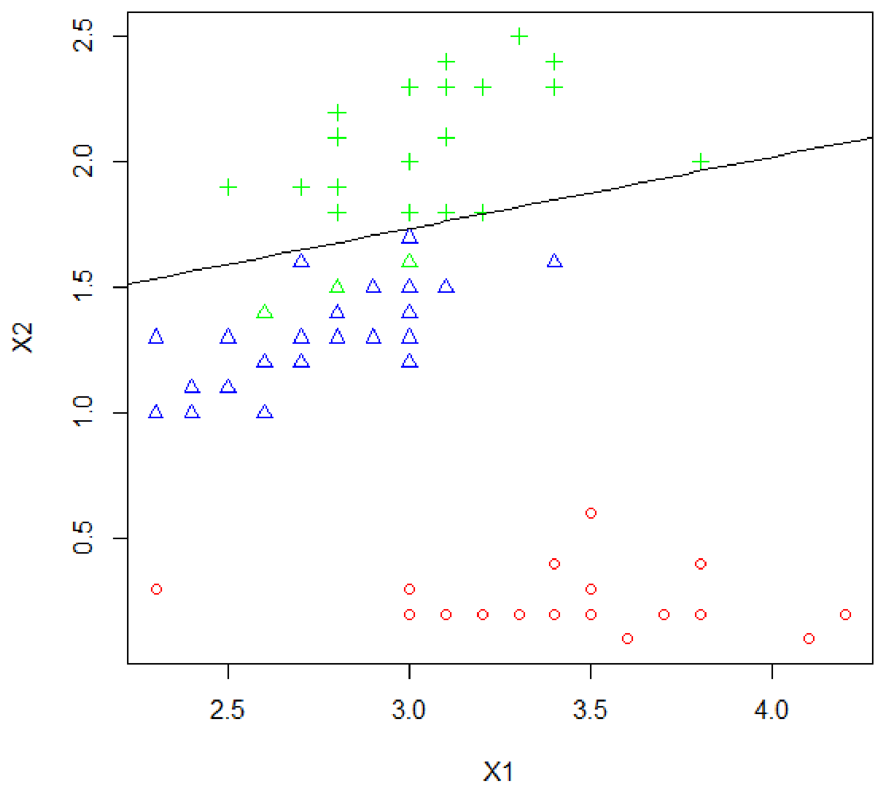
taux_bonne_classif_test = 0.9466667



Damien Druel

Linéaire (lda)

taux_bonne_classif_test = 0.96



En changeant les données test et app, le modèle obtient une base d'apprentissage différente. Que l'on utilise la fonction linéaire ou quadratique n'a donc plus de réelles conséquences sur le taux de classification puisque la classification effectuée par la méthode d'apprentissage n'est pas correcte.

Damien Druel

Conclusion

En utilisant les deux analyses (linéaire et quadratique) nous avons pu remarquer que pour des données déjà clairement séparées, la fonction Linéaire obtient un taux de classification meilleur que la fonction quadratique. Sur l'image de la fonction quadratique, cette chute du taux de classification est visible car la courbe ne considère pas 2 points (visible à X1=2.5 et X2=1.5). A l’inverse, pour les autres données un peu farfelues l’analyse quadratique s’est avérée plus efficace.