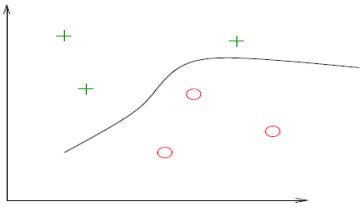
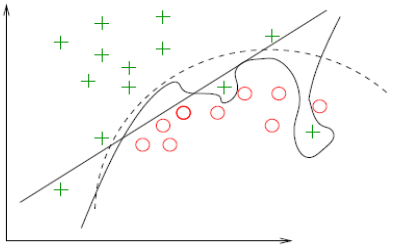
Classification supervisée
Aperçu de quelques méthodes avec le logiciel R

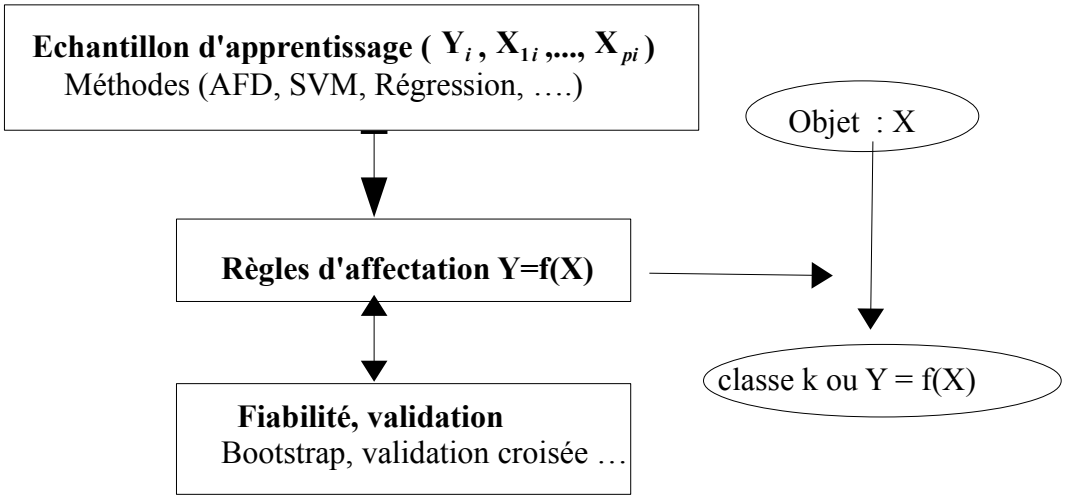
L'objectif de la classification supervisée est principalement de définir des **règles permettant de classer** des objets dans des classes à partir de variables qualitatives ou quantitatives caractérisant ces objets. Les méthodes s'étendent souvent à des variables Y quantitatives (régression).



On dispose au départ d'un **échantillon dit d'apprentissage** dont le classement est connu. Cet échantillon est utilisé pour l'apprentissage des règles de classement.



Il est nécessaire d'étudier la **fiabilité** de ces règles pour les comparer et les appliquer, évaluer les cas de sous apprentissage ou de sur apprentissage (complexité du modèle). On utilise souvent un deuxième échantillon indépendant, dit de validation ou de test.



On dispose de différentes **stratégies d'apprentissage** :

- **Règle majoritaire** : à tout objet, on associe la classe k telle que P(k) est maximale.
- **Règle du maximum de vraisemblance** : à tout objet on associe k telle que P(d/k) maximale.
- **Règle de Bayes** : à tout objet on associe k telle que P(k/d) maximale.

Ce document présente quelques méthodes en complément du cours sur l'analyse discriminante.

Table des matières

Méthode 1: Arbre de décision....4
Méthode 3 : Réseaux de neurone....7
Méthode 4: Support Vector Machine....11
Méthode 5 : Régression logistique....14
Méthode 6 : Réponse binaire - Courbe ROC....20
Méthode 7 : Autres méthodes....22
Méthode 8: Régression multiple, Ridge, Lasso, ACP, PLS....22

Quelques applications

- **Reconnaissance de formes**

Ex : Reconnaissance de chiffres manuscrits (codes postaux), Reconnaissance de visages
Entrées: image bidimensionnelle en couleur ou en niveaux de gris
Sortie: classe (chiffre, personne)

- **Catégorisation de textes**

Ex : Classification d'e-mails, Classification de pages web
Entrées: document (texte ou html)
Sortie: catégorie (thème, spam/non-spam)

- **Diagnostic médical**

Ex : Évaluation des risques de cancer, Détection d'arythmie cardiaque
Entrées: état du patient (sexe, age, bilan sanguin, génome...)
Sortie: classe (a risque ou non)

Exemples étudiés

Exemple 1: tennis

http://www.grappa.univ-lille3.fr/~ppreux/ensg/miashs/fouilleDeDonneesI/tp/arbres-de-decision/

```
> tennis <- read.table("http://www.grappa.univ-
lille3.fr/~ppreux/ensg/miashs/datasets/tennum.txt")
> tennis
     Ciel Temperature Humidite  Vent Jouer
1 Ensoleillé    27.5    85 Faible  Non
2 Ensoleillé    25.0    90  Fort  Non
3   Couvert    26.5    86 Faible  Oui
4    Pluie    20.0    96 Faible  Oui
5    Pluie    19.0    80 Faible  Oui
6    Pluie    17.5    70  Fort  Non
7   Couvert    17.0    65  Fort  Oui
8 Ensoleillé    21.0    95 Faible  Non
9 Ensoleillé    19.5    70 Faible  Oui
10   Pluie    22.5    80 Faible  Oui
11 Ensoleillé    22.5    70  Fort  Oui
12  Couvert    21.0    90  Fort  Oui
13  Couvert    25.5    75 Faible  Oui
14   Pluie    20.5    91  Fort  Non
```

Exemple 2 : iris

```
>data(iris)
> iris[1:2,]
 Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1     5.1     3.5     1.4     0.2 setosa
```

Références

Livres

- S. Tufféry, Data Mining et Statistique décisionnelle : l'intelligence des données, Technip, 2007.
- Michel Tenenhaus Collection: Gestion Sup, Dunod 2007 - 2ème édition - 696 pages -

web

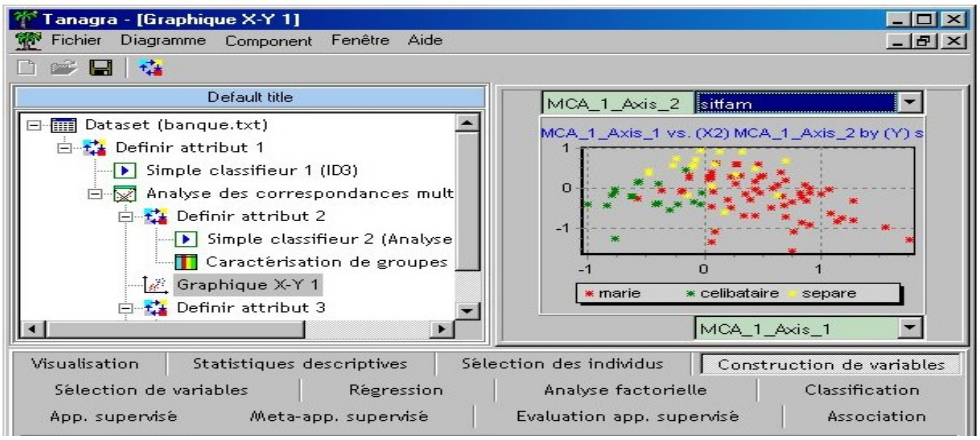
- http://www.math.univ-toulouse.fr/~besse/Wikistat/
- http://cedric.cnam.fr/~saporta/DM.pdf
- http://www.grappa.univ-lille3.fr/~ppreux/Documents/notes-de-cours-de-fouille-de-donnees.pdf
- http://www.google.fr/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5&ved=0CDUQFjAE&url=ftp%3A%2F%2Foceane.obs-vlfr.fr%2Fpub%2Fromagnan%2Fbiblio%2Fanalyses_stats%2FAppren_stat.pdf&ei=QsKcUMfyGceQhQff64GICw&usg=AFQjCNHCmbPD2jNhNhHAt87vVN3nlNIheA&sig2=fshJmLyEVue06A-olZK5XQ&cad=rja

Logiciels

Logiciels de références :

Rapidminer http://www.fil.univ-lille1.fr/~decomite/ue/MFFDD/tp1/rapidminer.pdf

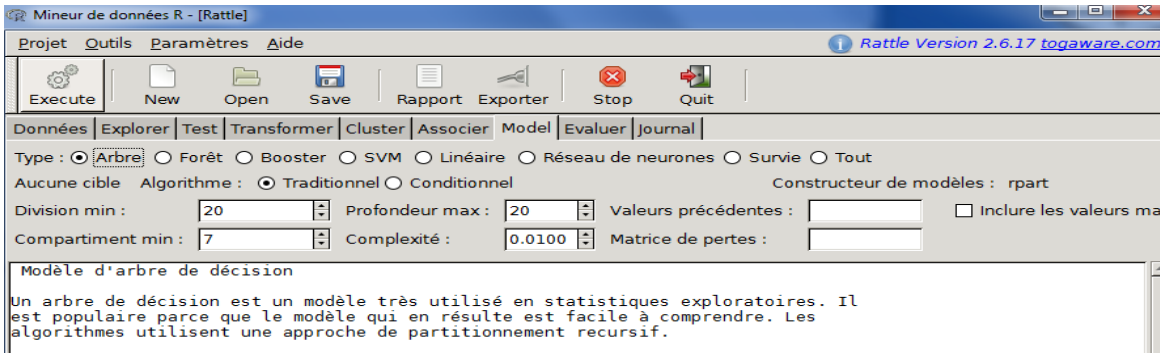
tanagra http://eric.univ-lyon2.fr/~ricco/tanagra/



Packages R : rattle, rpart, nnet

Il est possible dans certains cas d'utiliser l'interface rattle pour de la fouille de données.

```
> library(rattle)
> rattle()
```



Méthode 1: Arbre de décision

http://eric.univ-lyon2.fr/~ricco/doc/tutoriel_arbre_revue_modulad_33.pdf

1. Principe

L'apprentissage se fait par partitionnement récursif selon des règles sur les variables explicatives. Suivant les critères de partitionnement et les données, on dispose de différentes méthodes, dont CART, CHAID … Ces méthodes peuvent s'appliquer à une variable à expliquer qualitative ou quantitative. Deux types d'arbres de décision sont ainsi définis:

- **arbres de classification** : la variable expliquée est de type nominale (facteur). A chaque étape du partitionnement, on cherche à réduire l'impureté totale des deux nœuds fils par rapport au nœud père.
- **arbres de régression** : la variable expliquée est de type numérique et il s'agit de prédire une valeur la plus proche possible de la vraie valeur.

Construire un tel arbre consiste à définir une suite de nœud, chaque nœud permettant de faire une partition des objets en 2 groupes sur la base d'une des variables explicatives. Il convient donc :

- de définir un critère permettant de **sélectionner le meilleur nœud** possible à une étape donnée,
- de définir quand s'arrête le découpage, en définissant un **nœud terminal** (feuille),
- d'attribuer au nœud terminal la classe ou la valeur la plus probable,
- d'**élaguer l'arbre** quand le nombre de nœuds devient trop important en sélectionnant un sous arbre optimal à partir de l'arbre maximal,
- valider l'arbre à partir d'une validation croisée ou d'autres techniques

2. Critère de sélection d'un nœud

La construction d'un nœud doit réduire de façon optimale le désordre des objets.
Pour mesurer ce désordre, on définit l'entropie d'une variable qualitative Y à q modalités par :

$H(Y) = -\sum_{k=1}^{q} P(Y=k)\times\log(P(Y=k))$ avec la convention $0\log(0)=0$

$H(Y)$ est un critère d'incertitude de la variable Y.

On peut ensuite définir l'entropie de Y conditionnée par une variable qualitative X ayant q' modalités.

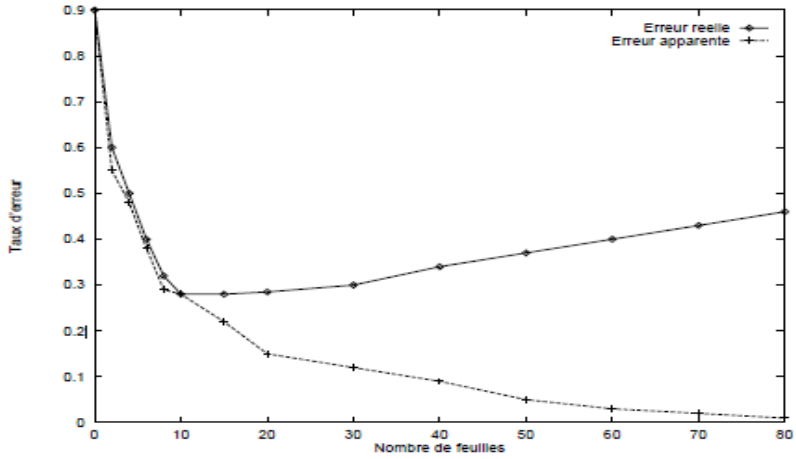
$$H(Y|X) = -\sum_{k}^{k'} P(Y=k, X=k')\times\log(P(Y=k|X=k')) = \sum_{k'} P(X=k')H(Y|X=k')$$

Dans l'exemple avec Y=Jouer et X=Vent: Y : 5-;9+] Y|X : [2-,6+] et [3-,3+]
H(Y)=-(5/14*log2(5/14)+9/14*log2(9/14))=0,94 [
H(Y|X)=-(2/14*log2(2/8)+6/14*log2(6/8)+3/14*log2(3/6)+3/14*log2(3/6))=0,89

Plus X nous renseigne sur Y plus l'entropie conditionnelle diminue. A l'extrême, $H(Y|Y)=0$. On choisira le nœud de façon à réduire au maximum le désordre.
Il existe d'autres critères :

- **critére de Gini** : G= 1 – $\sum_{k}^{k'} p_{kk'}$
- Pour un arbre de régression, on utilise le test de Fisher comme critère et dans l'algorithme **CHAID**, le test du χ^2.

Pour éviter le sur-apprentissage rendant l'arbre insuffisamment robuste, il faut souvent élaguer l'arbre maximal mais ces arbres sont souvent assez instable (règles peuvent changer facilement selon les données d'apprentissage).



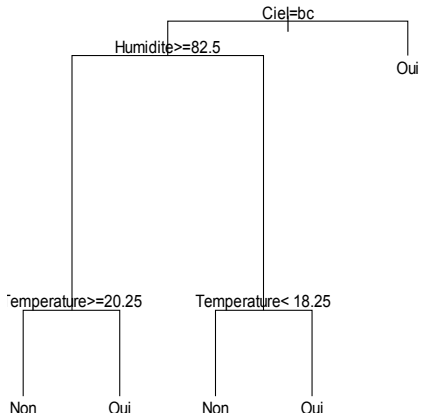
3. Exemple

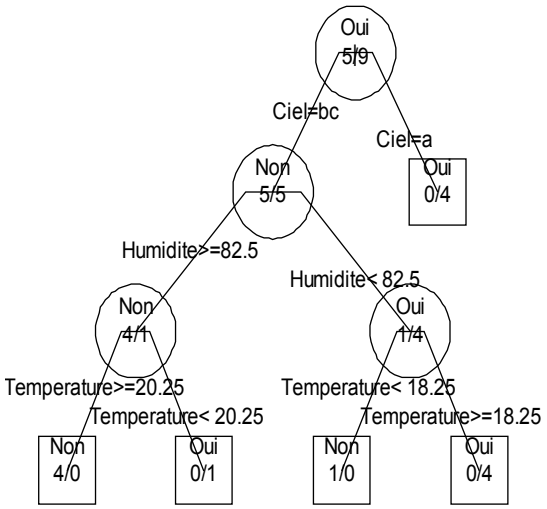
On utilise ici la library rpart.sur l'exemple :

```
library(rpart)
ad.tennis.cnt = rpart.control (minsplit = 1) # règle valable dès
un cas
ad.tennis=rpart(Jouer~.,tennis,control=ad.tennis.cnt)
ad.tennis
n= 14
node), split, n, loss, yval, (yprob)
        • denotes terminal node
 1) root 14 5 Oui (0.3571429 0.6428571)
  2) Ciel=Ensoleillé,Pluie 10 5 Non (0.5000000 0.5000000)
   4) Humidite>=82.5 5 1 Non (0.8000000 0.2000000)
    8) Temperature>=20.25 4 0 Non (1.0000000 0.0000000) *
    9) Temperature< 20.25 1 0 Oui (0.0000000 1.0000000) *
   5) Humidite< 82.5 5 1 Oui (0.2000000 0.8000000)
   10) Temperature< 18.25 1 0 Non (1.0000000 0.0000000) *
   11) Temperature>=18.25 4 0 Oui (0.0000000 1.0000000) *
  3) Ciel=Couvert 4 0 Oui (0.0000000 1.0000000) *

plot(ad.tennis)
text(ad.tennis)

plot (ad.tennis, branch=.2, uniform=T, compress=T, margin=.1)
text (ad.tennis, all=T, use.n=T, fancy=T)
```





summary(ad.tennis)

```
rpart(formula = Jouer ~ ., data = tennis, control = ad.tennis.cnt)
 n= 14

  CP nsplit rel error xerror   xstd
1 0.30   0    1.0  1.0 0.3585686
2 0.20   2    0.4  2.0 0.3380617
3 0.01   4    0.0  1.6 0.3703280

Node number 1: 14 observations,  complexity param=0.3
 predicted class=Oui expected loss=0.3571429
  class counts:   5   9
  probabilities: 0.357 0.643
 left son=2 (10 obs) right son=3 (4 obs)
 Primary splits:
   Ciel    splits as RLL,    improve=1.4285710, (0 missing)
   Humidite  < 82.5 to the right, improve=1.2857140, (0 missing)
   Temperature < 27  to the right, improve=0.8901099, (0 missing)
   Vent    splits as RL,    improve=0.4285714, (0 missing)
 Surrogate splits:
   Temperature < 25.25 to the left, agree=0.786, adj=0.25, (0 split)

Node number 2: 10 observations,  complexity param=0.3
```

…

> t(predict(ad.tennis, tennis))

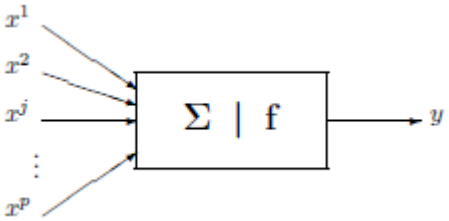
```
  1 2 3 4 5 6 7 8 9 10 11 12 13 14
Non 1 1 0 0 0 1 0 1 0 0 0 0 0 1
Oui 0 0 1 1 1 0 1 0 1 1 1 1 1 0
```

Méthode 2 : Réseaux de neurone

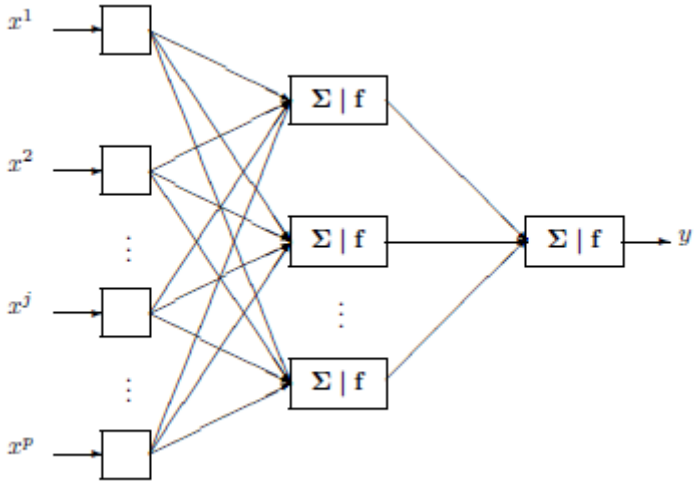
http://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-app-rn.pdfs :

1. Principe

Cette méthode repose sur la notion de neurone formel. Un neurone formel est un modèle caractérisé par des signaux d'entrée (les variables explicatives par exemple) , une fonction d'activation f, $f\left(\alpha_0+\sum_i \alpha_i \times x_i\right)$. f peut être linéaire, à seuil, stochastique et le plus souvent sigmoïde. Le calcul des paramètres se fait par apprentissage.



Les neurones sont ensuite associés en couche. Une couche d'entrée lit les signaux entrant, un neurone par entrée xj , une couche en sortie fournit la réponse du système. Une ou plusieurs couches cachées participent au transfert. Un neurone d'une couche cachée est connecté en entrée à chacun des neurones de la couche précédente et en sortie à chaque neurone de la couche suivante. De façon usuelle et en régression (Y quantitative), la dernière couche est constituée d'un seul neurone muni de la fonction d'activation identité tandis que les autres neurones (couche cachée) sont munis de la fonction sigmoïde.
En classification binaire, le neurone de sortie est muni également de la fonction sigmoïde tandis que dans le cas d'une discrimination à m classes (Y qualitative), ce sont m neurones avec fonction sigmoïde, un par classe, qui sont considérés en sortie..



2. Apprentissage (ajustement)

On minimise une fonction objective $Q(\alpha)$ (perte quadratique si Y est quantitative ou une fonction entropie en classification). A partir des gradients de cette fonction, on utilise un algorithme d'optimisation.

3. Paramètres de complexité

Le modèle dépend de plusieurs paramètres :

- l'architecture du réseau : nombre de couches cachées (une ou deux en général) et le nombre de neurones par couche,
- le nombre d'itération, l'erreur maximale tolérée et un terme de régularisation (decay).

Les paramètres de réglage sont difficiles à définir correctement. On peut utiliser library(e1071) par exemple pour rechercher les valeurs optimales

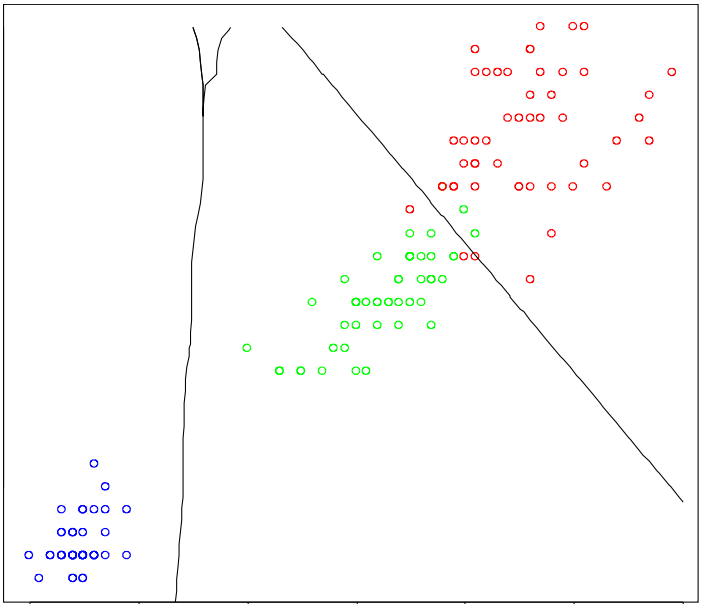
4. Exemple avec le package nnet

On souhaite prédire l'espèce d'iris en fonction des variables quantitatives `Petal.Length et Petal.Width`. La fonction `nnet` fonctionne avec des variables quantitatives pour prédire une variable qualitative. On se limite ici à deux variables quantitatives pour l'illustration graphique.

```
data(iris)
attach(iris)
H=class.ind(iris$Species) # tableau disjonctif complet : une
colonne par espèce et 1 si c'est la bonne espèce, 0 sinon
X=iris[,3:4] # On se limite à 2 variables quantitatives
colour=rep("blue",150)
colour[Species=="versicolor"]='green'
colour[Species=="virginica"]='red'
plot(X,col=colour)
library(nnet)
model=nnet(X,H,size=3,decay=0)
predX=predict(model,X) # probabilité proposée par le modèle
predX=apply(predX,1,which.is.max)
classX=apply(H,1,which.is.max)
table(classX,predX)
```

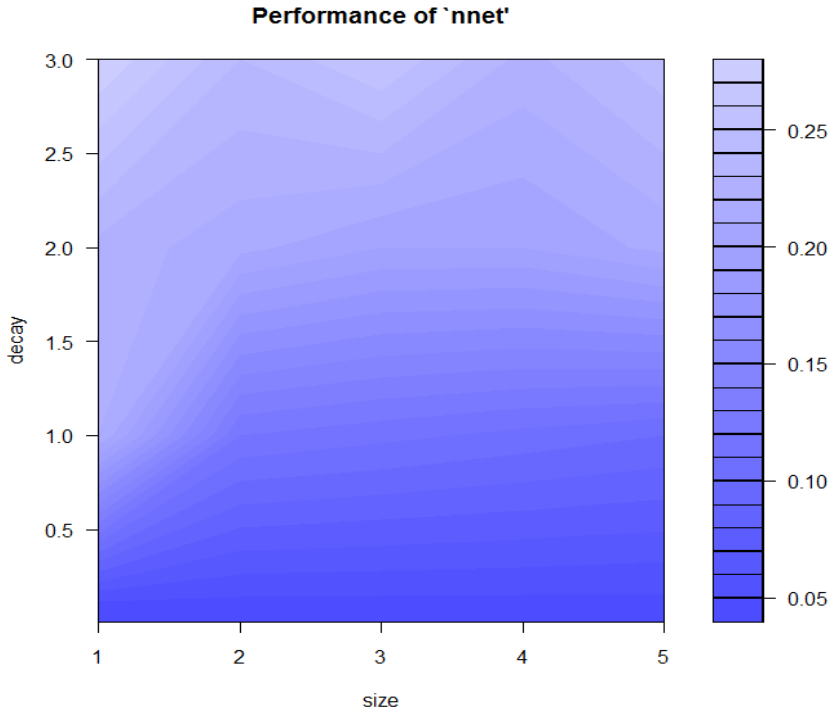
```
   predX
classX 1 2 3
   1 50 0 0
   2 0 47 3
   3 0 3 47
```

```
xp=seq(1,7,length=50)
yp=seq(0,2.5,length=50)
grille=expand.grid(z1=xp,z2=yp) # grille du plan
pred=predict(model,grille) # proba
zp=pred[,3]-pmax(pred[,1],pred[,2])bilités pour chaque point de la
grille
contour(xp,yp,matrix(zp,50),add=TRUE,levels=0,drawlabel=F)
zp=pred[,1]-pmax(pred[,3],pred[,2])
contour(xp,yp,matrix(zp,50),add=TRUE,levels=0,drawlabel=F)
```



Recherche de paramètres optimaux :

```
library(e1071)
plot(tune.nnet(Species~Petal.Length+Petal.Width,data=iris,size=1:5
,decay=c(0.01,0.1,1,2,3),maxit=100,linout=TRUE))
```

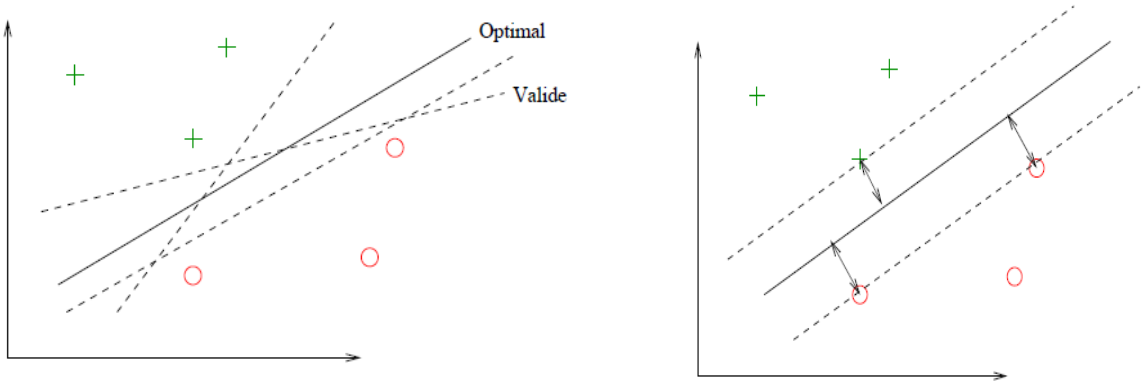


Méthode 3: Support Vector Machine

http://www.math.u-psud.fr/~blanchard/gtsvm/intro.pdf

1. Principe

Les entrées X sont transformées en un vecteur dans un espace de Hilbert F. Dans le cas d'un classement en 2 classes, on détermine un hyperplan dans cet espace F. La solution optimale repose sur la propriété que les objets sont les plus éloignés possibles de l'hyperplan, on maximise ainsi les marges.



Soit x le vecteur associé. On définit $f(x) = \omega x + \beta$ et l'hyperplan a pour équation $\omega x + \beta = 0$.
La distance d'un point au plan est donnée par

$$d(x) = | \omega x + \beta | / \| \omega \|$$

Le classement est correct si $yf(x) > 0$ ou à un coefficient près $yf(x) \geqslant 1$.

2. Ajustement

Maximiser la marge revient a minimiser ||w || ou ||w ||²/2 sous les contraintes $y_i \ f(x_i) \geqslant 1$.
On utilise la méthode des multiplicateurs de Lagrange en ne conservant que les vecteurs x_i les plus proches de l'hyperplan (vecteurs supports).
Lorsque tous les cas ne sont pas séparables, on introduit un terme d'erreur : $y_i \ f(x_i) \geqslant 1 - \xi_i$.

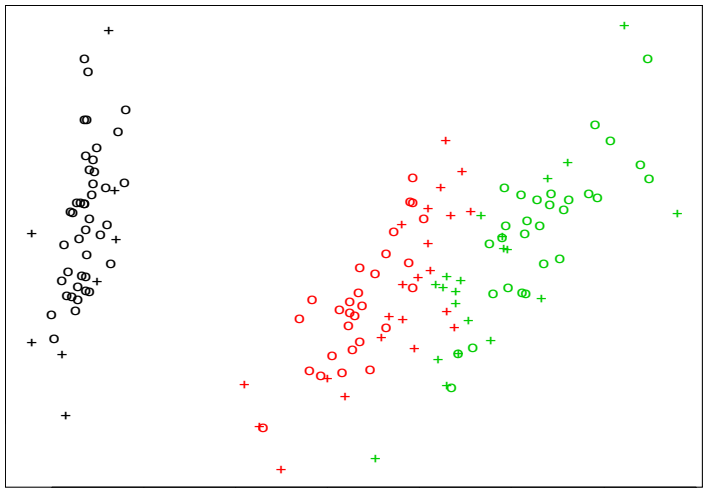
La transformation en vecteur ne fait intervenir que l'expression du produit scalaire dans F. On recherche en fait directement l'expression du produit scalaire à partir des coordonnées initiales à l'aide d'une fonction k appelée noyau. On distingue les noyaux linéaire, polynômiaux, gaussien … s'adaptant aux différentes problématiques rencontrées.

3. Etude d'un exemple

http://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/tp_ozone_svm.pdf

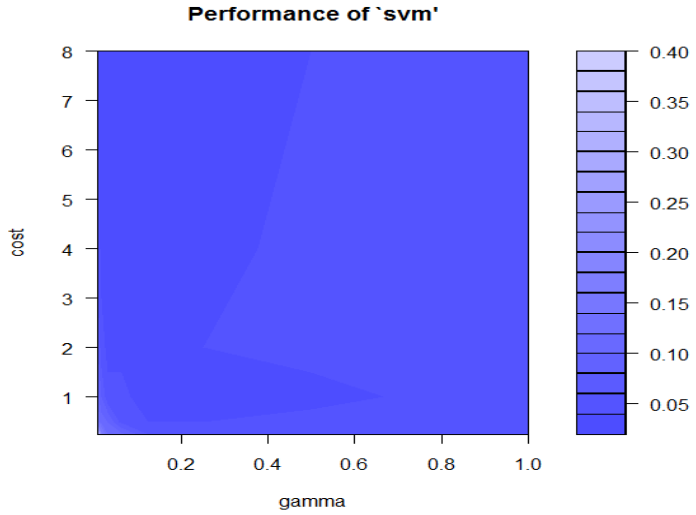
```
library(e1071)
# déclaration des données
data(iris)
# le modèle est calculé avec les valeurs
# par défaut des paramètres
# (noyau gaussien, pénalisation à 1, gamma=0,25)
```

```
model = svm(Species ~ ., data = iris)
print(model)
summary(model)
# prévision de l'échantillon d'apprentissage
pred = predict(model, iris[,1:4])
# Matrice de confusion pour l'échantillon
# d'apprentissage
table(pred, iris$Species)
# Visualisation des classes (couleurs)
# et des vecteurs supports ("+")
plot(cmdscale(dist(iris[,-5])),
col = as.integer(iris[,5]),
pch = c("o","+")[1:150 %in% model$index + 1])
```



optimisation

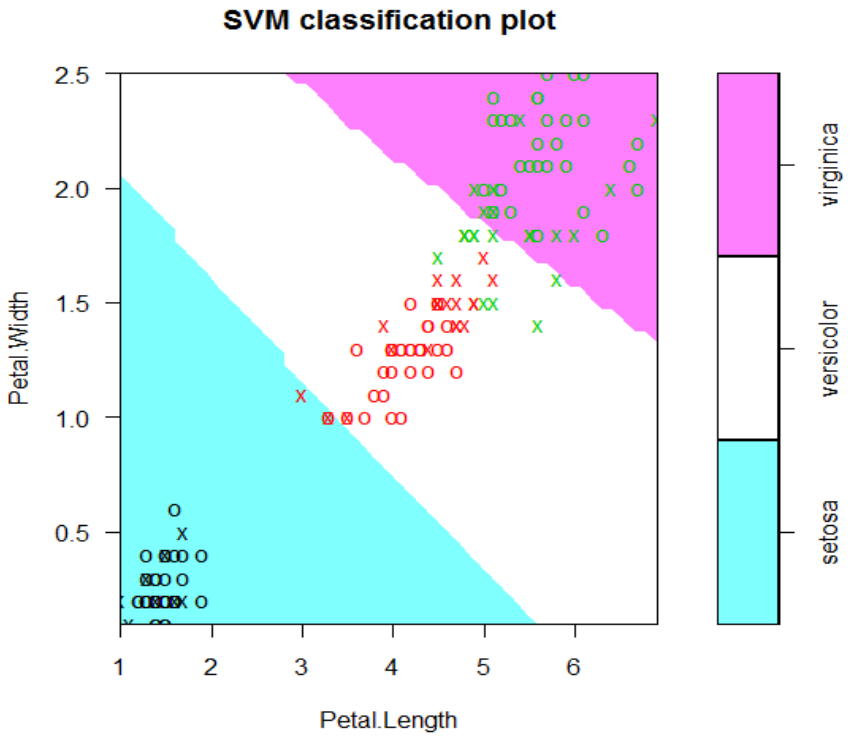
```
obj = tune.svm(Species~., data = iris,gamma = 2^(-7:0), cost = 2^(-2:3))
```




```
summary(obj)
plot(obj)
```

```
library(e1071)

plot(model, iris, Petal.Width ~ Petal.Length, slice = list(Sepal.Width = 3,
Sepal.Length = 4))
```



Méthode 4: Régression logistique

http://www.iumsp.ch/Unites/us/Alfio/polybiostat/ch22.pdf

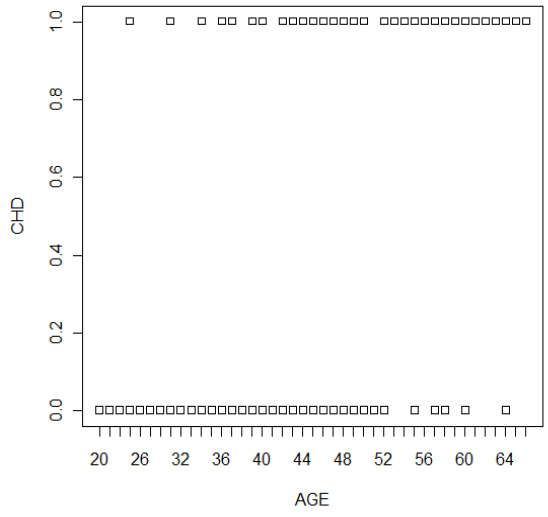
1. Principe

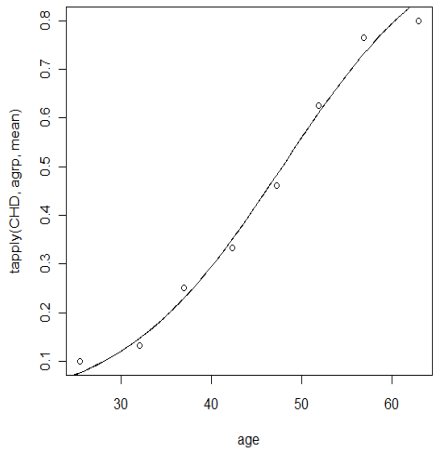
Nous étudions la présence chez l'homme de maladie cardiovasculaire en fonction de l'âge. La variable réponse est la maladie CHD (0 absence, 1 présence) et la variable explicative est l'âge ou la catégorie d'âge AGRP.

```
> cardio[1,]
AGRP AGE CHD
  1    20  0
```

On peut représenter les données par :

```
> stripchart(CHD~AGE,vertical=T,xlab="AGE")
>plot(1:8,tapply(CHD,agrp,mean))
> age=tapply(AGE,agrp,mean)
> age
    1        2        3        4        5        6        7        8
25.40000 32.00000 36.91667 42.33333 47.23077 51.87500 56.88235 63.00000
> plot(age,tapply(CHD,agrp,mean))
> x=seq(25,63,length=50)
> lines(x,exp(-5.31+ 0.111*x)/(1 + exp(-5.31 + 0.111*x)))
```





On note Y (1 ou 0) la réponse et X la variable explicative. On pose $\pi_x = P(Y=1|X=x)$. π_x ne peut pas être décrit par une fonction linéaire de x, nous allons pour cela utiliser **une fonction de lien**, ici la **fonction logit** $\log\left(\frac{y}{1-y}\right)$ qui elle s'exprime linéairement en fonction de x. On a alors :

$$\log \frac{\pi_x}{1-\pi_x} = \beta_0 + \beta_1 \ x$$

et inversement avec sa fonction réciproque (sigmoïde)

$$\pi_x = \exp(\beta_0 + \beta_1 x)/[1+\beta_0 + \beta_1 x]$$

$\frac{\pi_x}{1-\pi_x}$ définit un rapport de probabilité (odds). Si X est une variable binaire (Homme, Femme), ce rapport devient le rapport des probabilités d'avoir la maladie des hommes et femmes, soit le coefficient multiplicateur de la maladie si on est homme par rapport à une femme. Si odds est 2, on a deux fois plus de chance d'être malade en étant un homme.

En présence de plusieurs variables variables explicatives, on complexifie le modèle linéaire. On parle ici de **modèle linéaire généralisé GLMM**.

2. Ajustement

Prenons l'exemple d'une réponse binaire (0 ou 1). Pour chaque traitement i (combinaison des variables), on observe y_i succés pour n_i répétition. Y_i suit la loi binomiale de paramètres n_i et π_i. Donc :

$$\mathrm{P}\left(\mathrm{Y}_i = y_i\right) = \binom{n_i}{y_i} \left(\pi_i\right)^{y_i} \left(1-\pi_i\right)^{n_i - y_i}$$

On ajuste alors le modèle en maximisant la vraisemblance. Il n'y a pas de solution analytique et on utilise des méthodes itératives (Newton Raphson par exemple).

Avec les données de cardio, on obtient les coefficients estimés $\left(\hat{\beta}_0\right) = -5.310$ et $\left(\hat{\beta}_1\right) = 0.111$ (cf l'ajustement de la sigmoïde).

```
> cardio.glm=glm(CHD~AGE,family=binomial)
> summary(cardio.glm)
Deviance Residuals:
  Min    1Q  Median    3Q   Max
-1.9718 -0.8456 -0.4576  0.8253  2.2859
Coefficients:
      Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.30945  1.13365 -4.683 2.82e-06 ***
AGE     0.11092  0.02406  4.610 4.02e-06 ***
(Dispersion parameter for binomial family taken to be 1)
  Null deviance: 136.66 on 99 degrees of freedom
Residual deviance: 107.35 on 98 degrees of freedom
AIC: 111.35
```

Bilan sur l'échantillon d'apprentissage

```
> table(cardio.glm$fitted.values>0.5,CHD)
    CHD
        0 1
 FALSE 45 14
 TRUE  12 29
```

3. Test

Il existe trois principaux tests :

- **Test du maximum de vraisemblance**

On définit la déviance du modèle étudié par :

Deviance (modèle) = -2 log (vraisemblance du modèle/vraisemblance du modèle saturé)
= Null deviance – Residual Deviance
= 29,31

Le modèle saturé est le modèle ou les fréquences $\frac{y_i}{n_i}$ observées sont utilisées.

Cette déviance suit sous l'hypothèse nulle « $\beta_1 = 0$ » la loi du χ^2 à 99-98 ddl

> Anova(GLM.1, type="II", test="LR")

Analysis of Deviance Table (Type II tests)

Response: CHD

	LR Chisq	Df	Pr(>Chisq)	
AGE	29.31	1	6.168e-08	***

- **Test de Wald**

La statistique $T^2 = \left(\hat{\beta}^2\right) / \left(\left(\hat{\sigma_\beta}\right)^2\right)$ suit la loi du $\chi^2(1)$

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.30945    1.13365  -4.683 2.82e-06 ***
AGE          0.11092    0.02406   4.610 4.02e-06 ***
```

C'est la p-value de ce test que l'on iobtient pour les paramètres.

- **Test du score**

Test du score :

$$U = U'(\beta)_{\hat{\beta}_{H0}} \left[J(\hat{\beta}_{H0})\right]^{-1} U(\beta)_{\hat{\beta}_{H0}}$$

où $J(\hat{\beta}) = \left[-\frac{\partial^2 l(\beta)}{\partial \beta^2}\right]_{\beta=\hat{\beta}}$ et U le vecteur des dérivées partielles de la log-vraisemblance.
La statistique U suit une loi du $\chi^2(1)$.

La régression permet d'évaluer l'intérêt des différentes variables et interactions entre variables en recherchant un modèle optimal.

4. Retour sur l'exemple tennis

Modèle complet sans interaction

```
attach(tennis)
C=Temperature*0
C[Jouer=='Oui']=1
[1] 0 0 1 1 1 0 1 0 1 1 1 1 1 0
 tennis.gllm=glm(C~.,data=tennis[1:4],family='binomial')
summary(tennis.glm
Deviance Residuals:
   Min     1Q  Median     3Q    Max
-1.46009 -0.26286  0.00003  0.35316  1.57622
Coefficients:
          Estimate Std. Error z value Pr(>|z|)
(Intercept)     38.7955 4353.9192  0.009  0.993
Ciel[T.Ensoleillé] -21.5831 4353.8872 -0.005  0.996
Ciel[T.Pluie]    -20.8136 4353.8873 -0.005  0.996
Vent[T.Fort]     -3.7317   2.9052 -1.284  0.199
Temperature     -0.1551   0.3918 -0.396  0.692
Humidite       -0.1556   0.1215 -1.280  0.200
(Dispersion parameter for binomial family taken to be 1)
  Null deviance: 18.2492 on 13 degrees of freedom
Residual deviance: 8.4882 on 8 degrees of freedom
AIC: 20.488
Number of Fisher Scoring iterations: 18
```

Prédiction :

```
> R=C*0;R[predict(tennis.glm,tennis[,1:4],type="response")>0.5]=1;table(C,R)
  R
C  0 1
 0 4 1
 1 2 7
> predict(tennis.glm,tennis[,1:4],type="response")
     1     2     3     4     5     6     7
0.43058209 0.01211354 1.00000000 0.48553379 0.93000362 0.65559342 1.00000000
     8     9     10     11     12     13     14
0.30429080 0.96427698 0.88531270 0.28873658 0.99999998 1.00000000 0.04355647
```

Modèle complet avec interactions : on manque alors de données, le modèle est en sur apprentissage:

```
(C~.^2,data=tennis[1:4],family='binomial')->tennis.glm

> summary(tennis.glm)
Deviance Residuals:
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Coefficients: (1 not defined because of singularities)
                    Estimate   Std. Error z value Pr(>|z|)
(Intercept)              -3.27984 8954942.83251    0    1
Ciel[T.Ensoleillé]          77.18780 19134381.70509    0    1
Ciel[T.Pluie]            -103.43392 16524980.62931    0    1
Temperature              -1.05084  687229.45499    0    1
Humidite                2.85011  312632.12680    0    1
Vent[T.Fort]             -38.05549 10190688.41572    0    1
Ciel[T.Ensoleillé]:Temperature     5.09273 1228175.44486    0    1

....
```

Sous modèle avec interactions: nous allons utiliser seulement les variables **Vent et Humidite** et leurs interactions :

```
> glm(C~.^2,data=tennis[3:4],family='binomial')->tennis.glm
> summary(tennis.glm)
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)      13.50732  10.93514  1.235  0.217
Humidite       -0.14477  0.12433 -1.164  0.244
Vent[T.Fort]     -7.61191  12.69137 -0.600  0.549
Humidite:Vent[T.Fort] 0.07049  0.14801  0.476  0.634
  Null deviance: 18.249 on 13 degrees of freedom
Residual deviance: 14.626 on 10 degrees of freedom
AIC: 22.626

> predict(tennis.glm,tennis[3],type="response")
    1     2     3     4     5     6     7     8
0.7507914 0.2613214 0.7320061 0.5061735 0.8310241 0.7152758 0.8039570 0.5306386
    9    10    11    12    13    14
0.9291057 0.8310241 0.7152758 0.2613214 0.8892366 0.2428486
> R=C*0;R[predict(tennis.glm,tennis[3],type="response")>0.5]=1;table(C,R)
  R
C  0 1
 0 2 3
 1 1 8
```

Un des objectifs maintenant serait de déterminer le meilleur modèle possible avec les données dont nous disposons, en sélectionnant les variables d'intérêt (stepwise), mais cela sort de ce document. On utilise des paramètres comme AIC pour sélectionner un tel modèle.

Méthode 5 : Réponse binaire - Courbe ROC

On se place dans le cas d'une réponse binaire +/-. On obtient un résultat de prédiction de la forme

O\P	-	+	Total
-	TN	FP	N
+	FN	TP	P

avec

- TP (true positives) : les prédits positifs qui le sont vraiment.
- FP (false positives) : les prédits positifs qui sont en fait négatifs.
- TN (true negatives) : les prédits négatifs qui le sont vraiment.
- FN (false negatives) : les prédits négatifs qui sont en fait positifs.
- P (positives) : tous les positifs quelque soit l'état de leur prédiction. P = TP + FN.
- N (negatives) : tous les négatifs quelque soit l'état de leur prédiction. N = TN + FP.

On définit alors la spécificité et la sensibilité :

- la **sensibilité** est : TP / (TP + FN) = TP / P.
- la **spécificité** est : TN / (TN + FP) = TN / N.

Principe de la courbe ROC : Si le test donne un résultat numérique avec un seuil t tel que la prédiction est positive si x > t, et la prédiction est négative si x < t, alors au fur et à mesure que t augmente :

- la spécificité augmente.
- mais la sensibilité diminue.

La courbe ROC représente l'évolution de la sensibilité (taux de vrais positifs) en fonction de 1 - spécificité (taux de faux positifs) quand on fait varier le seuil t.

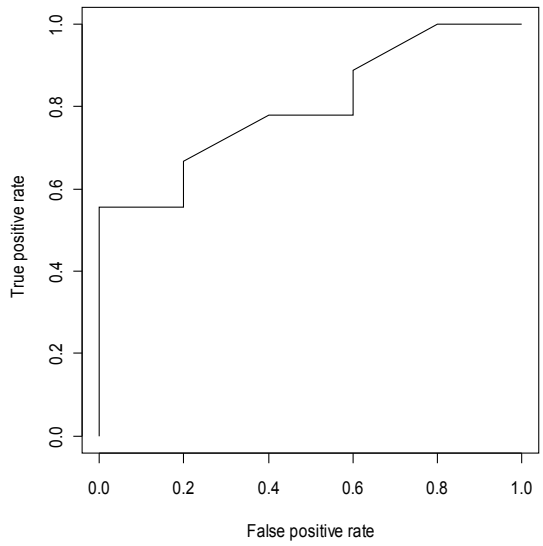
- C'est une courbe croissante entre le point (0,0) et le point (1, 1) et en principe au-dessus de la première bissectrice.
- Une prédiction random donnerait la première bissectrice.
- Meilleure est la prédiction, plus la courbe est au-dessus de la première bissectrice.
- Une prédiction idéale est l'horizontale y=1 sur]0,1] et le point (0,0).
- L'aire sous la courbe ROC (AUC, Area Under the Curve) donne un indicateur de la qualité de la prédiction (1 pour une prédiction idéale, 0.5 pour une prédiction random).

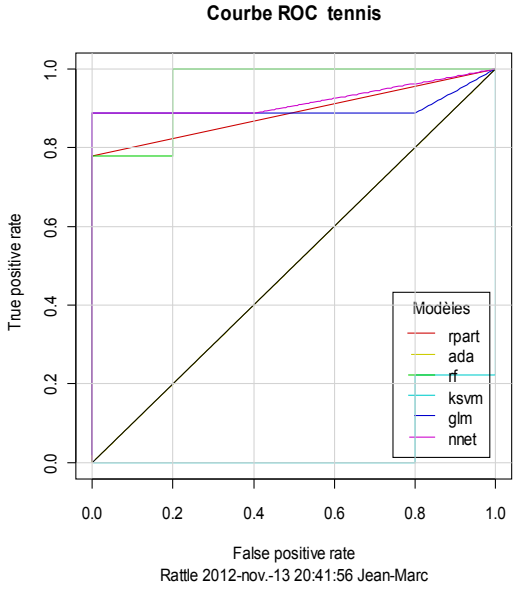
```
library(ROCR)
tennis.glm=glm(C~.,data=tennis[1:4],family='binomial')
> C
 [1] 0 0 1 1 1 0 1 0 1 1 1 1 1 0
> round(predict(tennis.glm,tennis[1:4],type="response"),1)
 1  2  3  4  5  6  7  8  9 10 11 12 13 14
 0.4 0.0 1.0 0.5 0.9 0.7 1.0 0.3 1.0 0.9 0.3 1.0 1.0 0.0
fr <- data.frame(score = predict(tennis.glm,tennis[1:4],type="response"),label
=C)
pred <- prediction(fr$score, fr$label)
perf <- performance(pred, "tpr", "fpr")
```

```
> fr[order(-fr$score),]
```

	score	label	TP	FP	TP/P	FP/N	
13	1.00000000	1	1	0	0,11	0	
7	1.00000000	1	2	0	0,22	0	
3	1.00000000	1	3	0	0,33	0	
12	0.99999998	1	4	0	0,44	0	
9	0.96427698	1	5	0	0,55	0	
5	0.93000362	1	6	0	0,66	0	
10	0.88531270	1	7	0	0,77	0	
6	0.65559342	0	7	1	0,77	0,2	
4	0.48553379	1	8	1	0,88	0,2	
1	0.43058209	0	8	2	0,88	0,4	
8	0.30429080	0	8	3	0,88	0,6	
11	0.28873658	1	9	3	0,99	0,6	
14	0.04355647	0	9	4	0,99	0,8	
2	0.01211354	0	9	5	0,99	1	

```
plot(perf)
```





Méthode 6 : Autres méthodes

Ce document est un simple descriptif de quelques méthodes parmi d'autres. Ce domaine est en constante évolution. Signalons par exemples comme méthodes non abordées : les forêts aléatoires, la méthode des plus proches voisins ...

Méthode 7 : Y quantitative : Régression multiple, Ridge, Lasso, ACP, PLS

Ces méthodes non présentées ici concernent le cas d'une ou plusieurs variables Y quantitatives à expliquer.

http://www.unitheque.com/Livre/springer/Regression_avec_R-37175.html
http://cedric.cnam.fr/~saporta/multicolinearite.pdf
http://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/tp_reg-penal_ukcomp.pdf
http://apiacoa.org/publications/teaching/data-mining/m2p6/supervised-slides.pdf