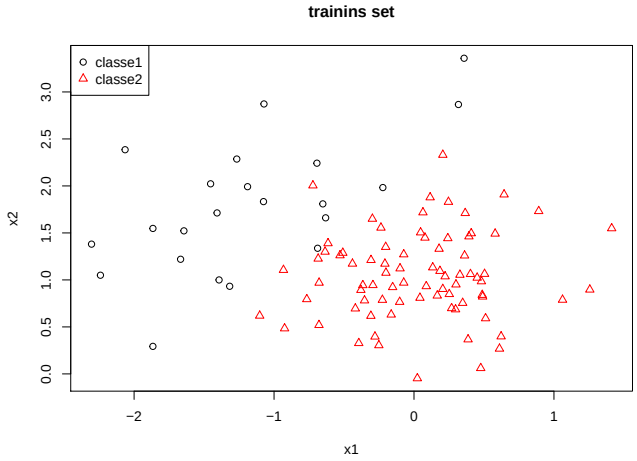
Master 2 MMS
Apprentissage automatique

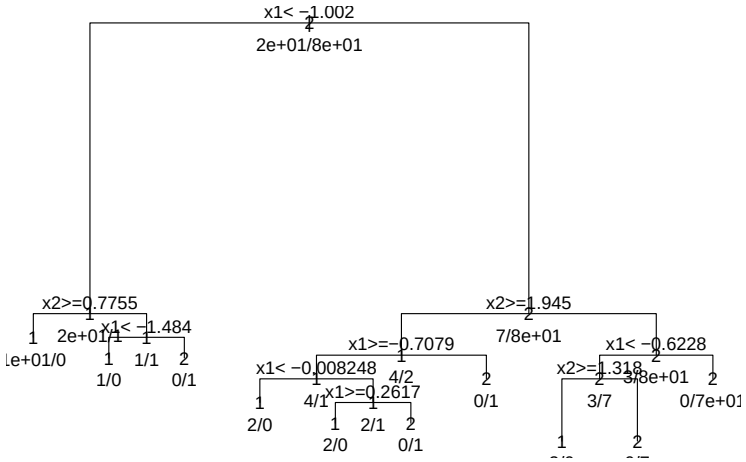
TP5 : arbres de classification

Exercice 1. Les objectifs :
- Constuire un arbre de classification sur un exemple simple.
- Découvrir la fonction `rpart`.

Reprendre les jeux de données synth_train.txt et synth_test.txt. On a $Y \in \{1, 2\}$ et $X \in \mathbb{R}^2$. On dispose de 100 données d'apprentissage et 200 données test.



1. Consulter l'aide des fonctions `rpart` et `rpart.control` du package `rpart` (**rpart** = **r**ecursive **part**itioning) :
   - quel algorithme est appliqué lorsque la variable à expliquer est qualitative ?
   - en laissant les paramètres par défaut, quelle version de cet algorithme est utilisée (priors, coûts, fonction d'impureté, critère d'arrêt, paramètre de complexité) ?
   - quelle est la différence entre un **competitor** et un **surrogate** ?
   - comment sont gérées les données manquantes dans cet algorithme ?
2. Construire un objet `tree` à l'aide de la fonction `rpart` en laissant tous les paramètres par défaut. Afficher `tree`, avec la fonction `print` et expliquez toutes les sorties.
3. Pour avoir plus de détails sur le déroulement de l'algorithme, afficher l'arbre avec la fonction `summary`. Expliquez toutes les sorties. Pourquoi l'algorithme s'arrête ici après une seule division ?
4. Tracer enfin l'arbre à l'aide des méthodes `plot` puis `text`.
5. Charger le jeu de données test puis representer la coupure $x_1 < -1.002$ sur le plot des données d'apprentissage puis des données test.
6. Calculer le taux d'erreur d'apprentissage et le taux d'erreur test.
7. Calculer à la main les probabilités à posteriori de la première et de la 10ème donnée test. Calculer ensuite avec la méthode `predict` les probabilités à posteriori de toutes les données test.
8. Modifier les paramètres de la fonction `rpart` pour construire l'arbre de longeur maximal T_{max} avec comme seul critère d'arrêt le nombre minimum d'observation dans le noeud $nmin = 2$. Tester ensuite différentes options des méthodes `plot` et `text` données dans la vignette **logintro.pdf**.



9. Afficher cet arbre avec la fonction `printcp`. Expliquez les sorties.
10. Afficher cet arbre avec la fonction `plotcp`. Expliquez le graphique.
11. A partir de ce graphique, choisir la valeur du paramètre de complexité *cp* avec la règle dite du "1-SE" et élaguer l'arbre T_{max} avec la fonction `prune`.
12. Vérifiez que vous comprenez le code et les sorties ci-dessous qui permettent d'introduire une matrice de coûts.

```
parms=list(loss=matrix(c(0,2,1,0), byrow=TRUE, nrow=2))
tree <- rpart(y~., data=train, method="class",parms=parms)
print(tree)
```

```
## n= 100
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 100 44 2 (0.22000 0.78000)
##   2) x1< -0.6228 28  9 1 (0.67857 0.32143) *
##   3) x1>=-0.6228 72  6 2 (0.04167 0.95833)
##     6) x2>=1.781 7  4 1 (0.42857 0.57143) *
##     7) x2< 1.781 65  0 2 (0.00000 1.00000) *
```

```
#summary(tree)
#cp(t1)= (44-9-6)/44
```

Exercice 2. On reprend les données concernant $n = 1260$ exploitations agricoles. Les variables explicatives sont $p = 30$ critères économiques et financiers et la variable qualitative à expliquer est la variable difficulté de paiement (0=saine et 1=défaillant).

1. Charger le jeu de données Desbois_complet.rda dans R.
2. Créez un découpage aléatoire des données en 945 observations d'apprentissage et 315 observations test.

3. Constuire l'arbre de classification CART avec les données d'apprentissage en laissant les options par défaut. Visualisez l'arbre en utilisant le package `rpart.plot` pour améliorer le graphique.
4. Visualiser ensuite la décroissante de l'erreur relative de validation croisée avec la fonction `plotcp`. Recommencez plusieurs fois. Que constatez-vous ?
5. Quel valeur du paramètre de complexité choisiriez-vous pour minimiser l'erreur relative de validation croisée ? Même question en utilisant la règle dite du "1-SE". Elaguez alors l'arbre en fonction de cette valeur.
6. Diminuez maintenant la valeur de l'argument `cp` dans la fonction `rpart` pour constuire l'arbre maximal et élaguer cet arbre en choisissant (automatiquement) la valeur du paramètre de complexité qui minimise l'erreur de validation croisée.
7. Elaguer maintenant l'arbre maximal en choisissant (automatiquement) la valeur du paramètre de complexité avec la règle dite du "1-SE".
8. Sur les données test calculer le taux d'erreur, tracer la courbe ROC et calculer le AUC.
9. Comparer alors pour plusieurs découpages les performances de la LDA, la régression logistique, de CART avec le paramétrage par défaut de `rpart`, CART avec l'élaguage selon la valeur du *cp* qui minimise l'erreur relative de validation croisée, CART avec l'élaguage selon la valeur du *cp* choisie avec la règle du 1-SE.