Arbre de décision et forêts aléatoires

Julien JACQUES

26/09/2018

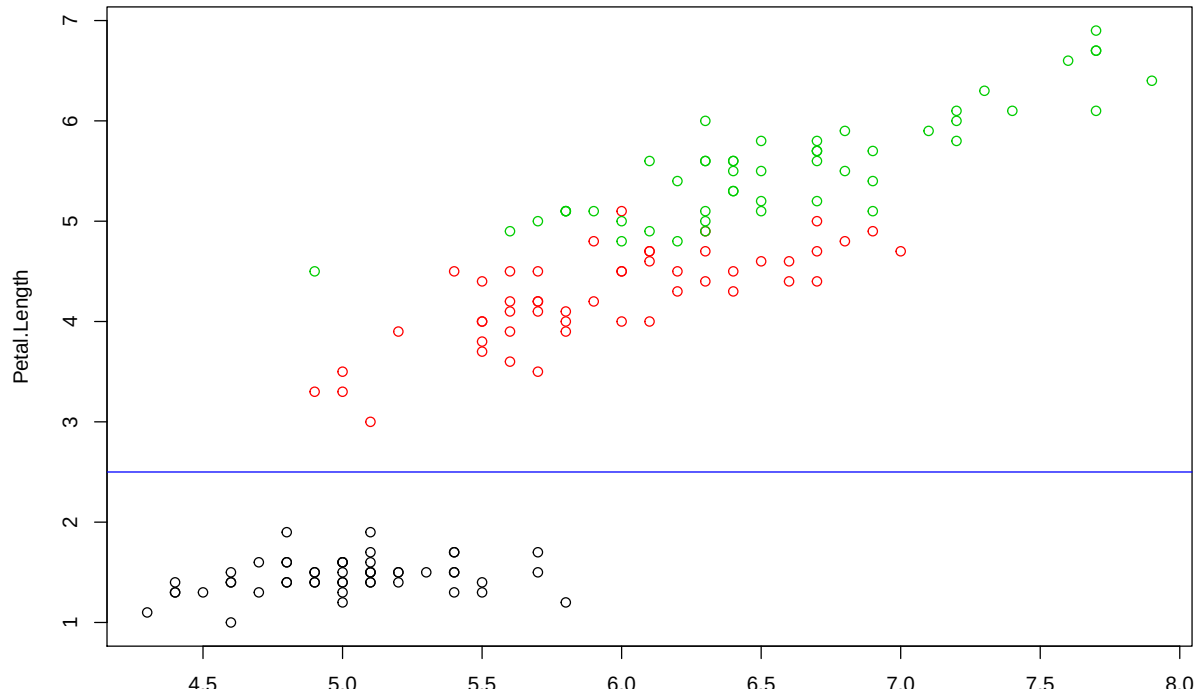
Arbre de décision et forêts aléatoires

Arbre binaire de classification

Principe :

- on veut construire des sous-groupes d'individus les plus homgènes possibles
- en définissant des endroits de coupures sur les variables explicatives

```
plot(iris[,c(1,3)],col=iris[,5])
abline(h=2.5,col=4)
```



Arbre binaire de classification

Les enjeux :

- trouver les variables et les endroits de coupure permettant de découper en groupes les plus homogènes possibles (critère de qualité ?)
- savoir jusqu'à quelle profondeur aller dans l'arbre (pour éviter le sur-apprentissage)

Principe

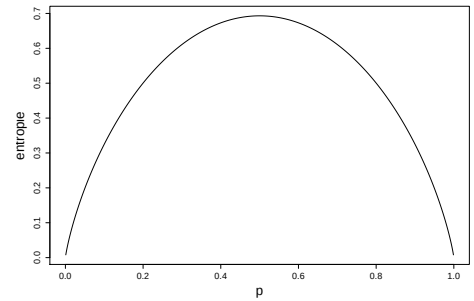
- arbre binaire de classification : succession de noeud
- noeud : défini par le choix d'une variable et d'une division $\Rightarrow$ partition en 2 classes
- division : choix d'un seuil (variable explicative quanti.) ou d'un groupe de modalités (variable explicative quali.)
- racine (noeud initial) : ensemble de l'échantillon ; la procédure est itérée pour chacun des 2 fils du noeud init. Et ainsi de suite.

Arbre binaire de classification

Critère de division

- D : critère d'hétérogénéité d'un noeud
  - critère d'entropie

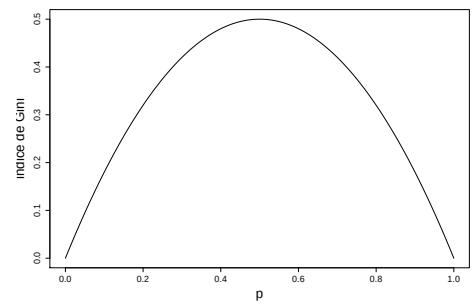
$$D = -2n \sum_{k=1}^{K} \frac{n_k}{n} \log \frac{n_k}{n}$$



où n est le nombre d'éléments du noeud, n_k celui de la classe k, et avec la convention $0 \log 0 = 0$

  - indice de Gini

$$D = -\sum_{k=1}^{K} \frac{n_k}{n} \left(1 - \frac{n_k}{n}\right)$$



- on retient la division qui maximise

$$D_{\text{noeud}} - (D_{\text{fils gauche}} + D_{\text{fils droit}})$$

Arbre binaire de classification

Règle définissant un noeud terminal

- si un des noeuds fils est vide
- si le nombre d'observation est inférieur à un seuil (entre 1 et 5)

Règle d'affectation d'un noeud à une classe

- la classe majoritaire
- la classe a posteriori la plus probable (si des probabilités a priori sont connues)
- la classe la moins coûteuse (si des coûts de mauvais classements sont connus)

Arbre binaire de classification

Construction d'un arbre optimal

- les arbres ainsi créés sont souvent très raffinés et donc instable (sur-ajustement)
- on va recherche des arbres plus parcimonieux :

Méthode CART {(Breinman et al. 1984)}

1. construction de l'arbre maximal
2. construction d'une séquence d'arbre emboités
   - on compare tous les arbres obtenus en supprimant une des dernières divisions (et donc une feuille) suivant un critère d'erreur de classement, et on retient le meilleur
   - ainsi de suite en remontant jusqu'à la racine
3. comparer les arbres obtenus via un taux de mauvais classement estimé sur échantillon test ou par validation croisée

Arbre binaire de classification

Avantages

- règle d'affectation lisible
- aucune hypothèse paramétrique
- robuste vis-à-vis des données atypiques
- traitement indifférencié des variables continues et catégorielles

Inconvenients

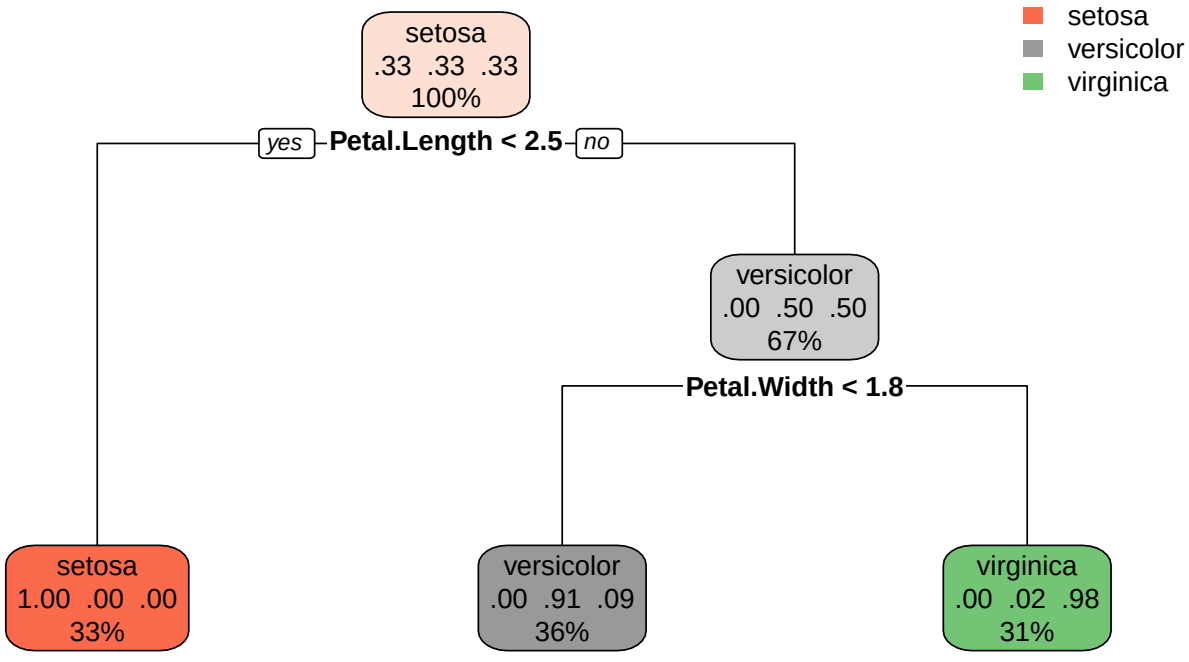
- demande des tailles d'échantillons grandes
- coûteux en temps de calcul
- parfois instable

Packages R

```
library('rpart')
library('rpart.plot')
```

Arbre classification sous R

```
arbre=rpart(Species~.,data=iris, cp = .02)
rpart.plot(arbre)
```



Random Forest

Compromis biais variance

- un modèle trop simple risque d'être trop biaisé, trop loin du *vrai* modèle
- un modèle trop complexe risque d'avoir une grande variance, c'est-à-dire d'être trop sensible aux fluctuations d'échantillonnage (sur-apprentissage)

Le meilleur modèle sera celui qui réalisera le meilleur compromis entre biais et variance.

Bagging

L'idée du **bagging** (Boostrap Aggregating) est de faire coopérer plusieurs arbres de classification construits sur des échantillons bootstrap (tirage avec remise) :

- on tire B échantillons de taille n avec remise
- pour chaque modèle on estime un arbre de classification
- pour un nouvel individu à classer, on le classe avec chaque arbre construit, et on retient la classe majoritaire

Bagging

Avantage :

- le bagging permet de réduire la variance par rapport à un unique arbre de classification
- même si le bagging peut être utiliser pour tous types de modèle, il est particulièrment adapté pour les arbres de classification non élagués :
  - les arbres non élagués auront des biais faibles
  - l'aggrégation de ces arbres permet de réduire la variance

Bagging sous R

Exemple du'itlisation du bagging avec les arbres de décisions sur les données iris

```
library(adabag)
sub=sample(1:nrow(iris),2*nrow(iris)/3)
model <- bagging(Species~.,data=iris[sub,],mfinal=10)
pred <- predict.bagging(model,newdata=iris[-sub, ])
pred$confusion
```

```
##                Observed Class
## Predicted Class setosa versicolor virginica
##      setosa         15          0         0
##      versicolor      0         21         1
##      virginica       0          2        11
```

```
model.bagging.2 <- bagging(Species ~ ., data = iris, mfinal
```

Bagging sous R

Même exercice mais en créant des arbres individuels plus profonds

```
model2 <- bagging(Species ~ ., data = iris[sub,], mfinal=10
                  control=list(cp=0,minsplit=2,minbucket=1))
pred <- predict.bagging(model2,newdata=iris[-sub,])
pred$confusion
```

```
##                  Observed Class
## Predicted Class setosa versicolor virginica
##      setosa         15          0         0
##      versicolor      0         20         1
##      virginica       0          3        11
```

Bagging sous R

L'importance relative des variables peut être obtenue par :

```
model2$importance
```

```
## Petal.Length  Petal.Width Sepal.Length  Sepal.Width
##  78.65557620  19.62946145   1.64675362   0.06820873
```

Cet indicateur prendre en compte le gain d'index de Gini obtenu par une variable dans chaque arbre.

Random Forest

Pour que le bagging soit efficace, il faut que :

- les arbres soient profonds (taille minimal des feuilles $= 1$)
- les arbres soient différents les uns des autres

Les **forêts aléatoires** considérent des arbres profonds et **introduit une perturbation dans le choix des variables** à chaque segmentation de noeux: plutôt que de tester toutes les p variables, on en teste que m choisies aléatoirement parmi p.

Random Forest sous R

On peut utiliser le package *randomForest* :

```
library(randomForest)
model <- randomForest(Species~.,data=iris[sub,],ntree=200)
```

On peut afficher la matrice de confusion sur l'échantillon Out-Of-Bag (calculées sur données non choisies dans chaque échantillon bootstrap) :

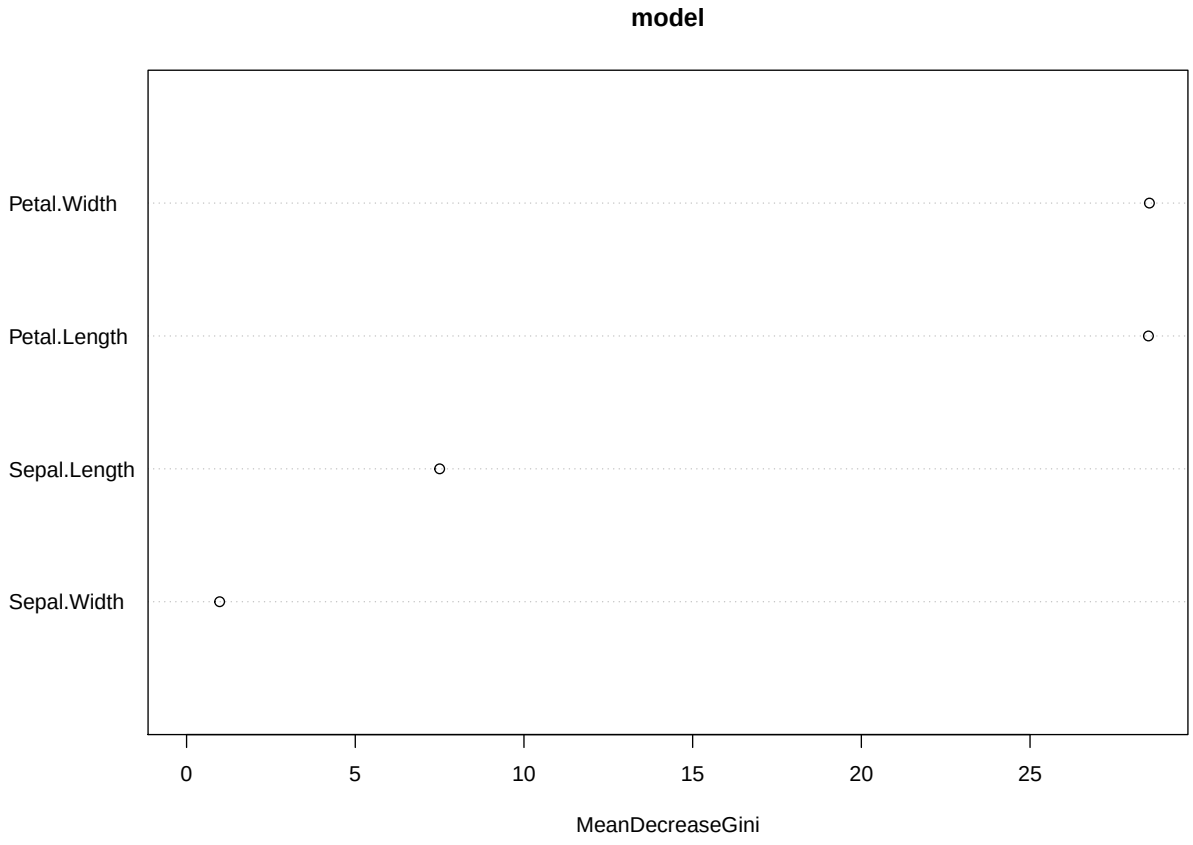
```
model$confusion
```

```
##            setosa versicolor virginica class.error
## setosa         35          0         0  0.00000000
## versicolor      0         25         2  0.07407407
## virginica       0          3        35  0.07894737
```

Random Forest sous R

L'importance des variables est donnée par :

```
varImpPlot(model)
```



Random Forest sous R

L'erreur sur l'échantillon test est :

```
pred<-predict(model,newdata=iris[-sub,],type="class")
table(pred,iris[-sub,5])
```

```
##
## pred          setosa versicolor virginica
##   setosa          15          0         0
##   versicolor       0         21         1
##   virginica        0          2        11
```

Bilan sur Random Forest

Avantages :

- bonnes performances en prédiction
- pas de problème d'overfitting (on peut augmenter B)
- mesure de l'importance des variables
- évaluation de l'erreur intégrée (OOB)

Inconvénient :

- problème si le nombre de variables pertinentes est faible