# Arbori de decizie
# Decision Trees (DT)

Ruxandra Stoean
rstoean@inf.ucv.ro
http://inf.ucv.ro/~rstoean

# Bibliografie


- Leo Breiman, Jerome Friedman, Charles J. Stone, R.A. Olshen, Classification and Regression Trees, Chapman and Hall/CRC; 1 edition (January 1, 1984)

- Florin Gorunescu, Data Mining: Concepts, Models and Techniques, Intelligent Systems Reference Library, Volume 12, Springer, 2011

- Dianne Cook, Deborah F. Swayne, Graphics for Data Analysis. Interactive and Dynamic With R and Ggobi, Springer, 2007

# Introducere

- Tehnica foarte apreciata in ML:
  - Posibilitate de vizualizare a modelului de clasificare a datelor
  - Invatare white box, spre deosebire de masini cu suport vectorial, retele neuronale (black box)
  - Utilizarea datelor de orice tip

- Procedeul de discriminare a claselor problemei:
  - Se partitioneaza recursiv datele de antrenament
  - Procesul se opreste cand se obtin noduri terminale cu aceeasi categorie

# Inductia DT

- Constructia DT este un proces inductiv (decision tree induction)

- Numerosi algoritmi:
  - Hunt
  - CART
  - ID3
  - C4.5
  - etc.

# Structura DT

- Fiecare nod intern exprima un test relativ la un atribut al problemei care partitioneaza obiectele
- Fiecare arc semnifica partitia rezultata in urma aplicarii testului ce corespunde acelui nod
- Nodurile terminale reprezinta clasa (dominanta intre clasele problemei) ce a rezultat pe acea ramura.

# Alegerea testului pentru un nod

- Masura eficientei partitionarii datelor in nodul curent pe baza unei perechi (atribut, valoare) este data de trei criterii clasice:
  - Indexul Gini (Gini index) de partitionare care trebuie sa aiba valoare minima
  - Castigul de informatie (information gain) maxim
    - Sau entropie minima
  - Masura de clasificare gresita
    - Eroarea minima a partionarii generate de nod

# Alte aspecte

- Probabilitati prealabile (prior probabilities):
  - Se poate specifica probabilitatea apartenentei unui obiect la o clasa
  - De obicei, acest parametru este proportional cu numarul de obiecte din fiecare clasa
- Costuri de clasificare gresita (misclassification costs):
  - Se poate specifica daca o clasa este mai important a se determina mai precis decat alta
  - De obicei, acest parametru este egal cu 1 – costuri egale pentru toate clasele

# Extragerea regulilor din DT

- Reguli de tip DACA-ATUNCI (IF-THEN)
- Are loc o coborare pe arce
- Se realizeaza o conjunctie intre valorile (testele) nodurilor intermediare pana la frunze pentru partea conditionala a unei reguli
- Concluzia unei reguli este data de frunza atinsa in coborare

# Pachetul rpart

- Implementeaza arbori de clasificare si regresie
- Pentru inductie utilizeaza algoritmul CART (Breiman, 1984)
- Criteriul de partionare bazat pe indexul Gini sau pe information gain.

# Exemplu

```
library(rpart)
library(datasets)

# 150 de date, cate 50 din fiecare clasa: setosa, versicolor, virginica
data(iris)
dat <- iris

# partionare dupa criteriul Gini (implicit)
rpart.model <- rpart(Species~., dat, method="class")
rpart.model

plot(rpart.model, margin = 0.05); text(rpart.model, use.n = TRUE,
cex = 0.8)
```

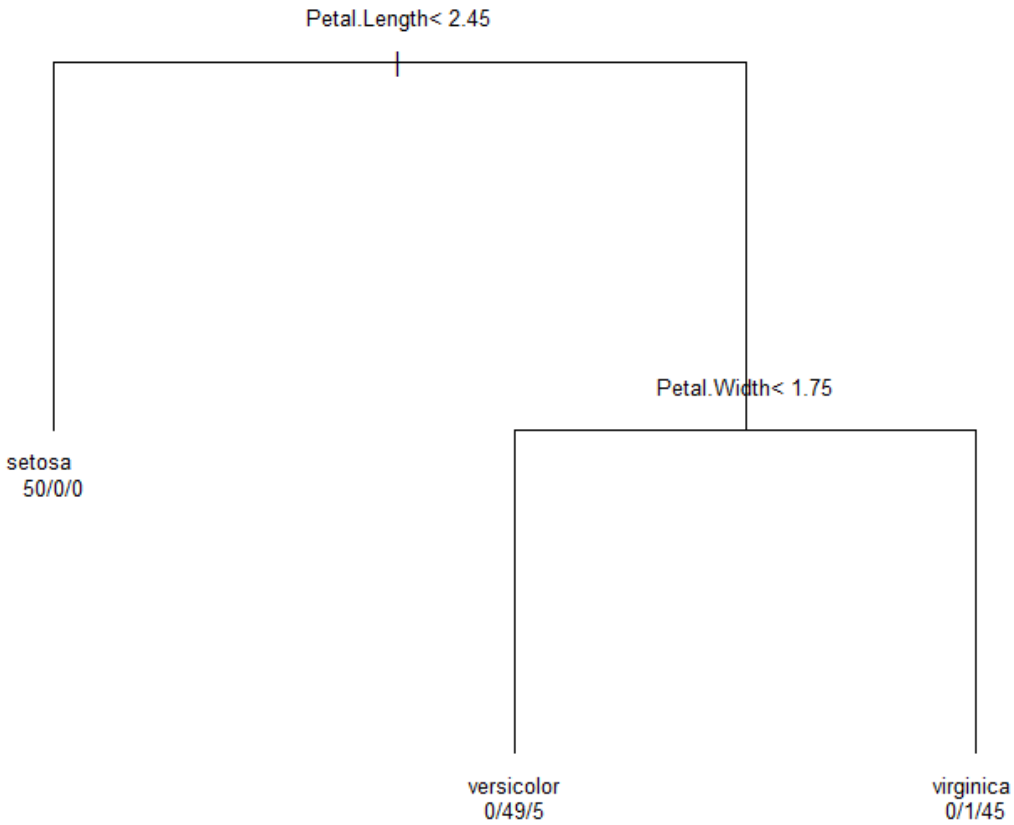
# Modelul construit

```
> library(rpart)
> library(datasets)
>
> data(iris)
> dat <- iris
>
> # partionare dupa criteriul Gini (implicit)
> rpart.model <- rpart(Species~., dat, method="class")
> rpart.model
n= 150

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
  2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.000000
00) *
  3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.5
0000000)
    6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.0
9259259) *
    7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97
826087) *
>
> plot(rpart.model, margin = 0.05); text(rpart.model, use.n = TRUE, c
ex = 0.8)
```

# Arborele rezultat



# Predictie 1/2

```
# clasa reala: setosa
nou1 <- data.frame(Sepal.Length = 5.1,  Sepal.Width = 3.5,  Petal.Length = 1.4, Petal.Width =          0.2)
predict(rpart.model, nou1, type = c("class"))

# clasa reala: versicolor
nou2 <- data.frame(Sepal.Length = 6.3,  Sepal.Width = 2.5,  Petal.Length = 4.9, Petal.Width =  1.5)
predict(rpart.model, nou2, type = c("class"))

# clasa reala: virginica
nou3 <- data.frame(Sepal.Length = 6.5,  Sepal.Width = 3.0,  Petal.Length = 5.5, Petal.Width =  1.8)
predict(rpart.model, nou3, type = c("class"))
```

# Predictie 2/2

```
> # clasa reala: setosa
> nou1 <- data.frame(Sepal.Length = 5.1,  Sepal.Width = 3.5,  Petal.L
ength = 1.4, Petal.Width = 0.2)
> predict(rpart.model, nou1, type = c("class"))
     1
setosa
Levels: setosa versicolor virginica
>
> # clasa reala: versicolor
> nou2 <- data.frame(Sepal.Length = 6.3,  Sepal.Width = 2.5,  Petal.L
ength = 4.9, Petal.Width =  1.5)
> predict(rpart.model, nou2, type = c("class"))
          1
versicolor
Levels: setosa versicolor virginica
>
> # clasa reala: virginica
> nou3 <- data.frame(Sepal.Length = 6.5,  Sepal.Width = 3.0,  Petal.L
ength = 5.5, Petal.Width =  1.8)
> predict(rpart.model, nou3, type = c("class"))
         1
virginica
Levels: setosa versicolor virginica
```

# Exercitii

- Implementati in R un model de arbore de decizie pentru problema diagnozei diabetului (Pima Indians Diabetes) [1].

[1] https://archive.ics.uci.edu/ml/datasets/Pima+Indians+Diabetes