Data Mining
Regras de Classificação, parte III

Prof. Dr. Joaquim Assunção

DEPARTAMENTO DE COMPUTAÇÃO APLICADA
CENTRO DE TECNOLOGIA
UFSM
2019

www.inf.ufsm.br/~joaquim

Fair user agreement

Este material foi criado para a disciplina de Mineração de Dados - Centro de Tecnologia da UFSM.

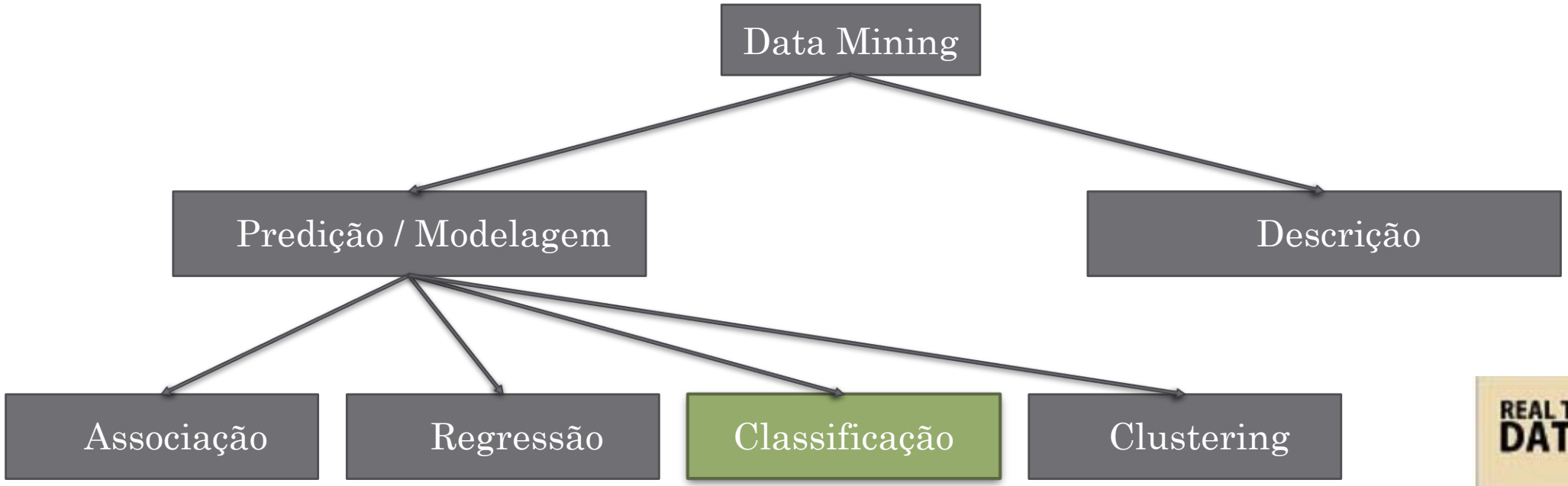
Você pode usar este material livremente*; porém, caso seja usado em outra instituição, **me envie um e-mail** avisando o nome da instituição e a disciplina.

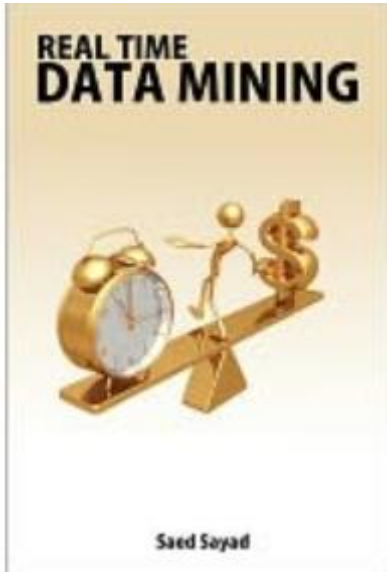
*Caso você queira usar algo desse material em alguma publicação, envie-me um e-mail com antecedência.

Prof. Dr. Joaquim Assunção.

joaquim@inf.ufsm.br

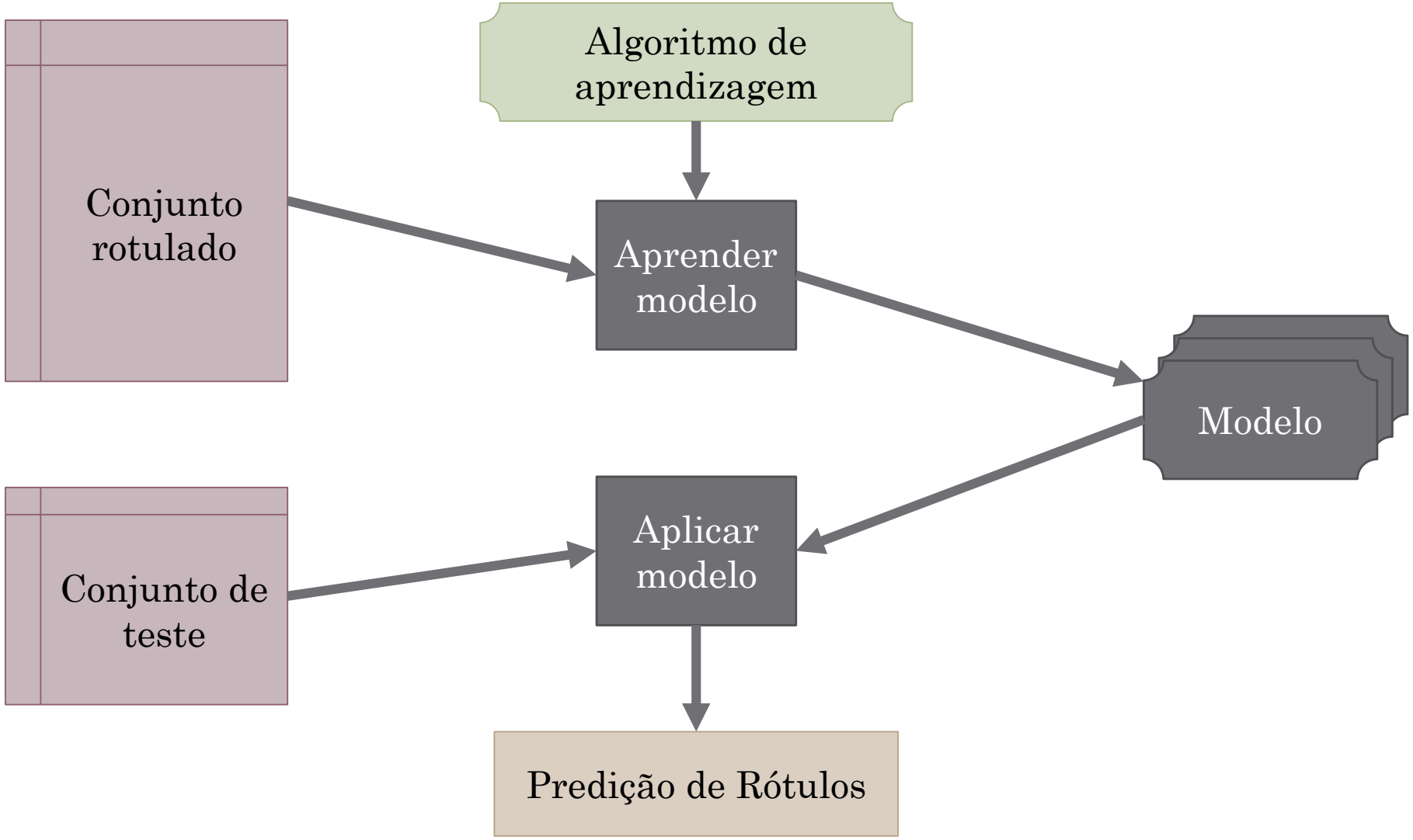
Mapa para Mineração de Dados*





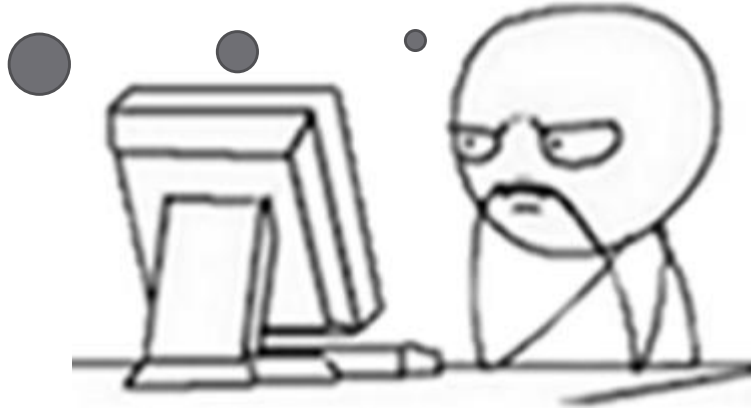
***http://www.saedsayad.com/data_mining_map.htm**

Abordagem para construção de um modelo.



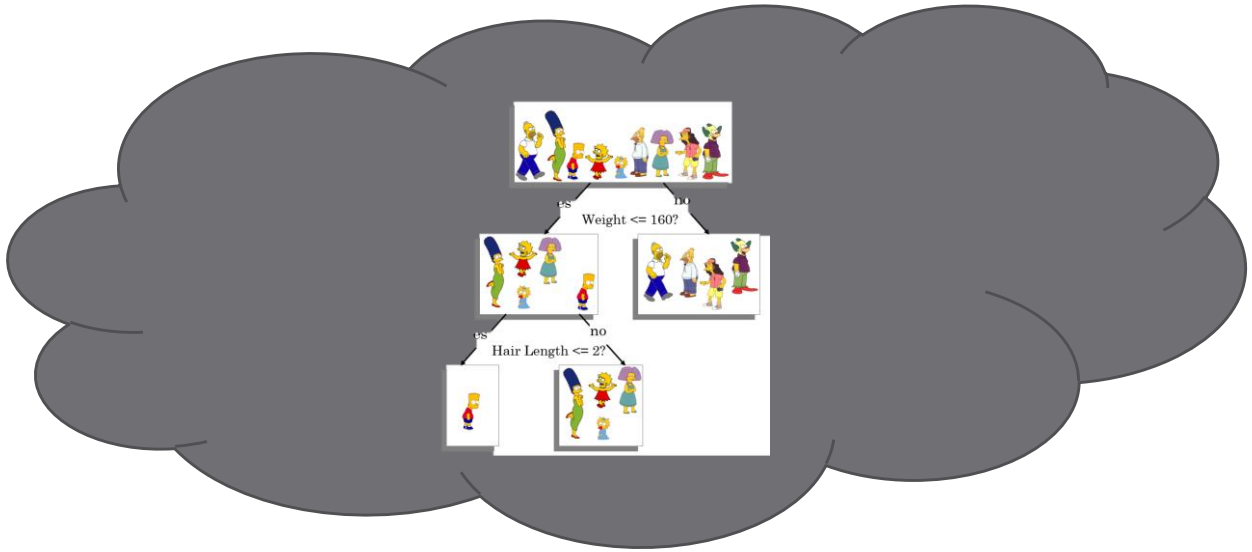
Conjunto rotulado (treino)

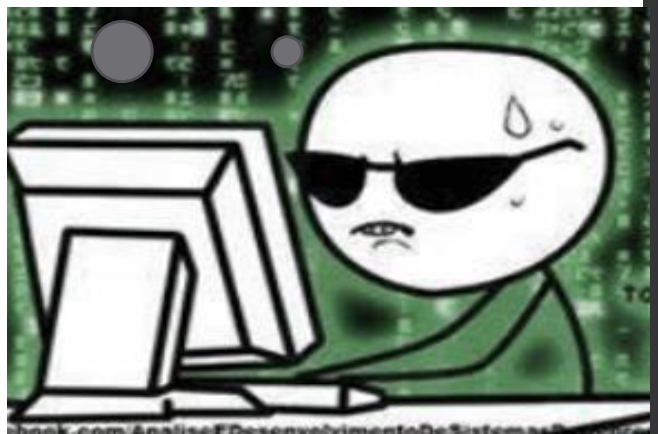
- Um conjunto cuja classe (objetivo) deve estar descrito de modo com que o algoritmo aprenda as melhores combinações para cada classe.
- Assim como em sala de aula, o treino é aquilo que é supervisionado (aulas e exercícios).



Conjunto de validação (teste)

- O conjunto rotulado gerou uma série de regras para classificar os dados. Agora vejamos quão bem esse classificador se sai.



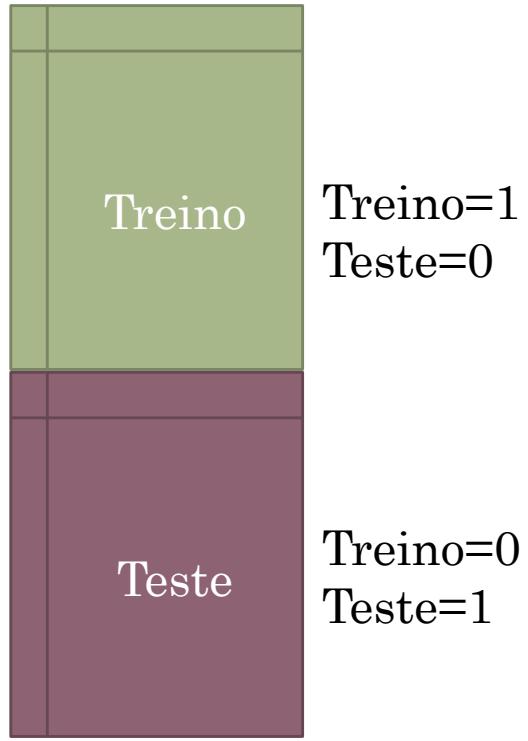


Holdout

- O método comum para validar um classificador (conjunto de treino e teste)
- O conjunto é dividido em partes, geralmente 70% e 30% ou 60% e 40% para, respectivamente, treino e teste.

Holdout → Validação cruzada

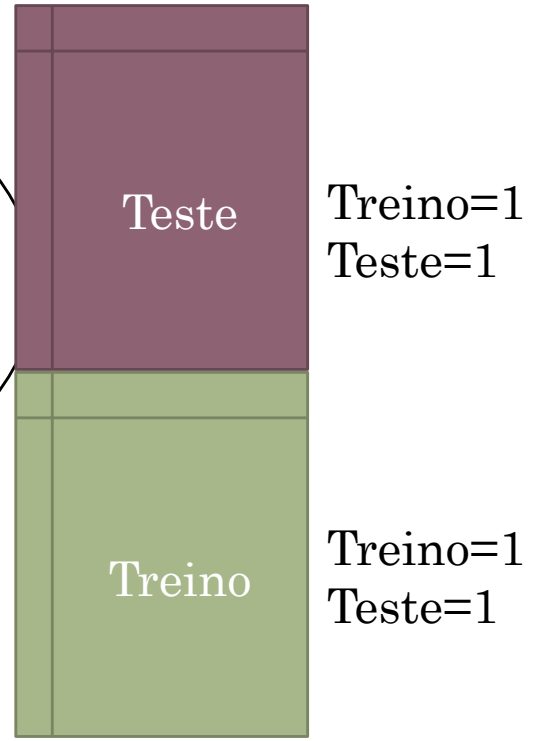
- Nesta abordagem cada registro, de n partições, é usado $n - 1$ vezes para treinamento e 1 vez para teste.
- A partição mais simples é a divisão por dois. 50% a 50% fará com que cada partição seja usada para treino e teste uma única vez.
  - Logo, cada um recebe um papel, depois troca.



Holdout → Validação cruzada

- Nesta abordagem cada registro, de n partições, é usado $n-1$ vezes para treinamento e 1 vez para teste.

Se $n = 3$ *Então*
$nTreino = 2;$
...
Teste sempre será 1

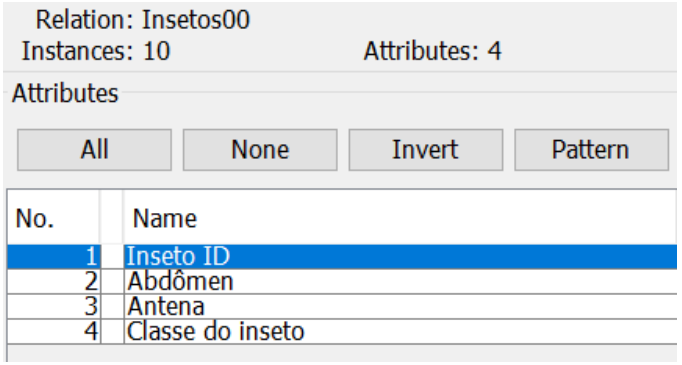


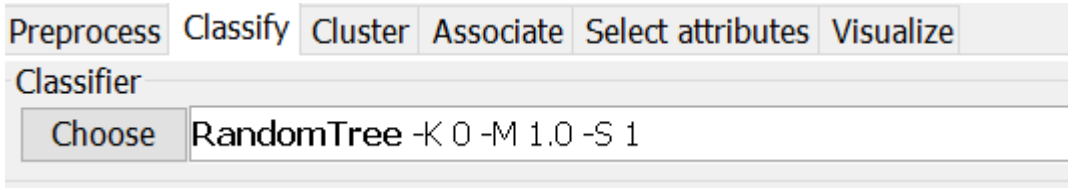
Check point

- No Weka, ao entrar na aba de classificação, uma das opções de teste é “Cross-validation”. Teoricamente, quantas vezes cada *fold* será usado como treino, e como teste, se um dataset de 140 registros forem submetidos a *folds=10 ?*

Hands on!

No Weka, carregue o arquivo ‘Insetos00.csv’.





Vá até a aba *classify* e selecione ‘*RandomTree*’ como classificador. Use validação cruzada, visualize a árvore, depois use divisão percentual. Qual obteve melhor resultado?

Boosting

- A ideia do *boosting* é coletar classificadores fracos e transformá-los em classificadores fortes.
- É como um grupo que decide por um indivíduo, uma espécie de democracia entre os classificadores.
- Em analogia a uma plateia, é como perguntar a um indivíduo pertencente a uma plateia, este pode ter pouca certeza de sua escolha, mas a plateia como um todo pode ter um palpite melhor, influenciados por indivíduos que sabem a resposta.

Boosting - uma analogia real

- Também podemos imaginar a seguinte situação: em uma corte, um rei solicita conselho para um grupo de plebeus, o grupo deve chegar a uma conclusão melhor que um único plebeu e possivelmente um único conselheiro que passa seus dias estudando o reino.

Isso se o conhecimento dos plebeus for parcialmente, ou todo (utópico), somado.

Boosting - uma analogia real

- Por mais estranha que pareça esta estratégia, estas são as bases para o Boosting. Escolha vários conselheiros pouco competentes, e aos poucos chegue em um filtro comum.
- ...

Boosting - uma analogia real

- Para que isso aconteça, cada conselheiro recebe uma devida atenção do rei. Alguns são mais creditados que outros, mas a combinação é que fará a decisão final.

Neste caso, o Rei é o algoritmo do Boosting, os conselheiros são os classificadores e os créditos atribuídos a cada um são chamados de pesos.

Boosting - uma analogia real

Ainda na analogia do rei, um conselheiro formal seria um classificador forte pois este conhece bem o reino e acerta a grande maioria das vezes.

Boosting - uma analogia real

Um plebeu seria um classificador fraco, pois este conhece bem sua realidade, mas desconhece eventos esternos e tramites burocráticos, táticas de guerra ou diplomacia externa.

Boosting - uma analogia real

Contudo, se juntarmos muitos plebeus, encontraremos soldados aposentados, viajantes e estrangeiros que conhecem bem outros assuntos em questão; logo, se o grupo for unido e escutar cada indivíduo, este grupo se torna forte.

Boosting - classificador forte e fraco

Matematicamente, um classificador **forte** é aquele que tem uma taxa de erro próxima a zero.

Um classificador **fraco** é aquele que tem uma taxa de erro perto de *1/2*.

Note que *1/2* é o que temos em um evento aleatório de duas possibilidades, como jogar uma moeda.

Leitura recomendada

A. Leães, P. Fernandes, L. Lopes, J. Assunção. **Classifying With AdaBoost.M1: The Training Error Threshold Myth.**

https://www.aaai.org/ocs/index.php/FLAIRS/FLAIRS17/paper/viewFile/15498/14966

Hands on!

No Weka, carregue o arquivo ‘Insetos00.csv’.

1) Escolha “*AdaboostM1*” como meta-learner e DecisionStump como classificador.

2) Compare o resultado de exercício anterior com esse resutado (use validação cruzada e divisão percentual).

CART

- Classification And Regression Trees (**CART**) é um termo introduzido por Breiman para se referir aos algoritmos de Árvore de Decisão que podem ser usados para problemas de modelagem preditiva de classificação ou regressão.

CART

- CART é a base para muitos algoritmos de classificação.
- Há vantagens em se usar o CART puro.
  - Do ponto de vista de *machine learning*, podemos citar a facilidade de entender, implementar e manipular o algoritmo.
  - Do ponto de vista de *data mining*
    - É um algoritmo rápido para rodar em grandes conjuntos de dados.
    - É relativamente fácil (configurações) de controlar a complexidade da árvore gerada.
    - Permite fácil visualização e análise dos dados.

CART

- A representação do CART é uma típica árvore binária.
  - Cada nó raiz representa uma variável de entrada única (X) e um ponto de divisão nessa variável (supondo que a variável seja numérica).
  - Os nós de folhas da árvore contêm uma variável de saída (Y) que é usada para fazer uma previsão.

Technical help

Use a biblioteca rpart → `install.packages("rpart")` … e a biblioteca para plot `install.packages("rpart.plot")`

Use `data(…)` para carregar um conjunto no R

Para treinar um modelo com o rpart use a função `rpart(variavel_classe ~ ., data= meuDataFrame, method=…)`

Use `type` e `digits` para controlar o plot de `rpart.plot`

Hands on!

- Carregue os dados da biblioteca do `ggplot2` → `msleep`.
- Crie um sub conjunto dos dados com as seguintes variáveis:

```
$ order
$ sleep_total
$ sleep_cycle
$ awake
$ brainwt
```

- Use a função rpart usando *sleep_total* como classe.
- Gere a árvore de decisão e analise a saída.