

# » Introdução ao *Data Mining*

Pedro Campos  
João Lopes  
Conceição Ferreira

DMSI / ME



3 a 6 de julho de 2017

»

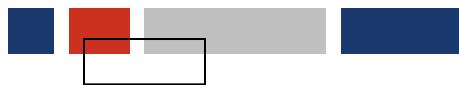


# Índice



- Introdução
- Preparação dos Dados
- Métodos de avaliação
- Modelos Exploratórios
- Modelos Preditivos
- Computação Natural





# Sumário



- 1. Introdução**
- 2. Preparação dos Dados**
- 3. Métodos de avaliação**
- 4. Modelos Exploratórios**

*Clustering*

Regras de Associação

- 5. Modelos Preditivos**

*K-Nearest Neighbour (K-NN)*

*Naive Bayes*

*Árvores de Decisão*

*Árvores de Regressão*

*Redes Neuronais Artificiais*

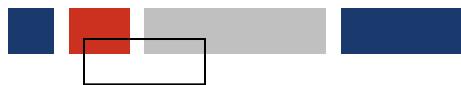
*Support Vector Machines (SVM)*

- 6. Computação Natural**

*Algoritmos Genéticos*

*Particle Swarm Optimization*



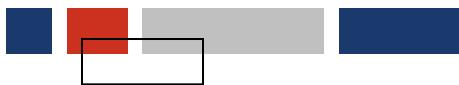


# Introdução



- » Enquadramento
- » O que é *Data Mining*?
- » Etapas do Processo de Extração de Conhecimento
- » Os algoritmos de *Data Mining*
- » Aplicações
- » *Data Mining* e outras áreas
- » *Data Mining* Vs. Estatística
- » Modelo CRISP-DM
- » Principais Tarefas do *Data Mining*
- » Aprendizagem
- » Software





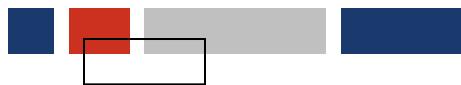
# Introdução



## » Enquadramento

- Numa economia baseada em conhecimento, a quantidade de informação gerada em cada momento levou ao aparecimento de procedimentos capazes de tratar a informação de forma rápida e, muitas vezes em contínuo.
- Este novo paradigma de geração e tratamento de informação conduziu ao aparecimento do conceito de *Big Data*, que envolve **velocidade, volume, variedade, veracidade e valor** de informação.





# Introdução

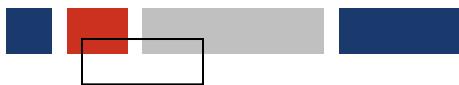


## » Enquadramento

O *Data Mining* (DM) surgiu devido a:

- Grande acumulação de dados (sempre a aumentar);
- Omnipresença dos computadores;
- Baixos preços da computação;
- Dados digitalizados (cartões de crédito, via verde, internet, telecomunicações, energia, etc.)

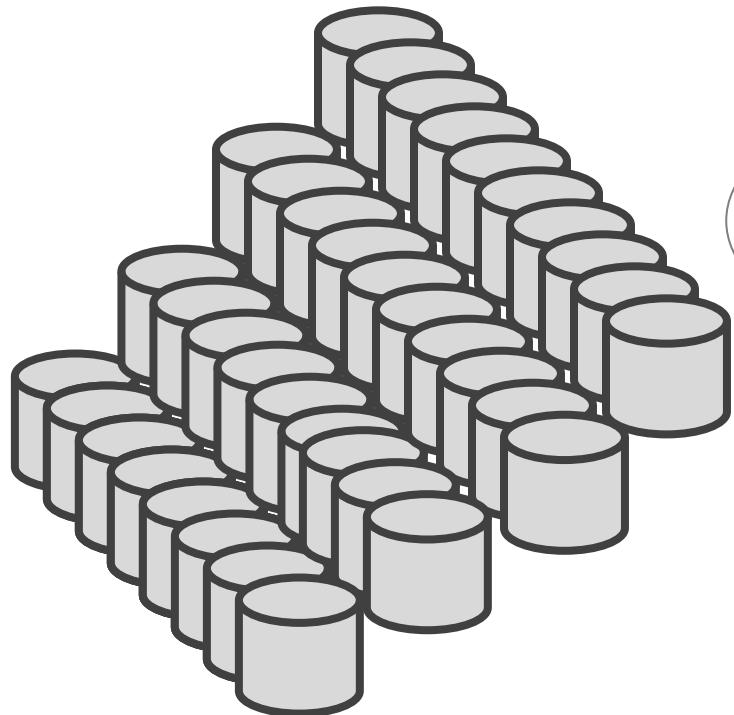
**Muita informação mas pouco conhecimento!**



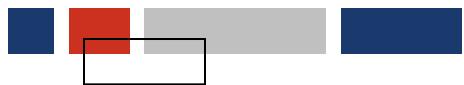
# Introdução



## » Enquadramento



**Como podemos  
analisar os dados?**

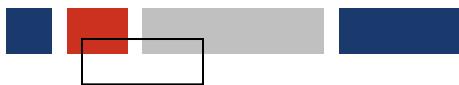


# Introdução



## » O que é *Data Mining*?

- Processo de descoberta de padrões nos dados;
- Processo automático ou semiautomático (com recurso a computadores e algoritmos);
- Os padrões encontrados devem fazer sentido;
- Os dados estão sempre presentes e em grandes quantidades (grandes bases de dados)
- Dados: exemplos ou instâncias

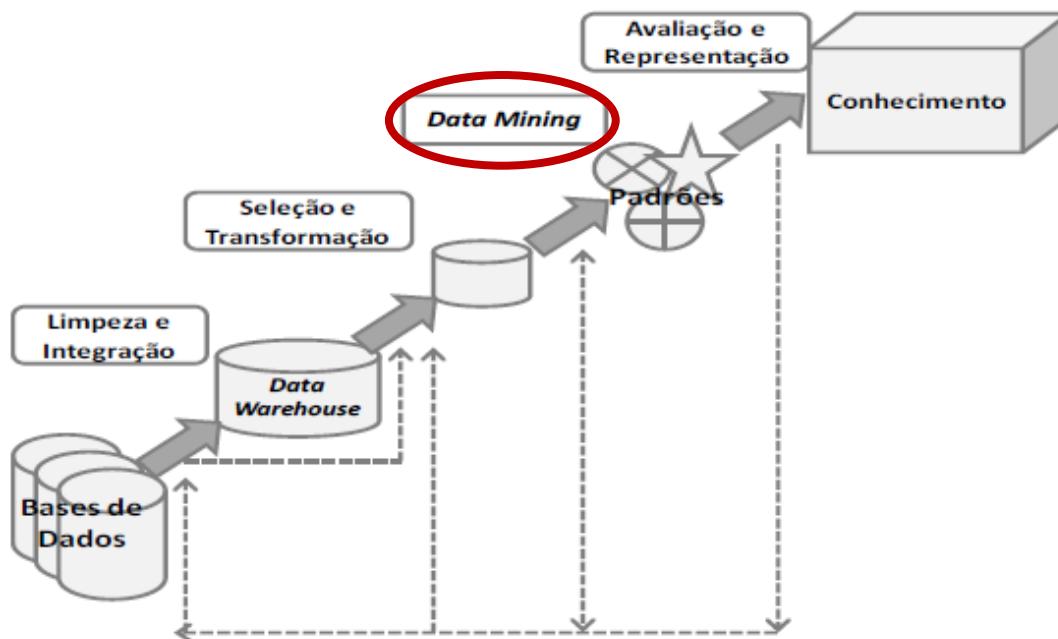


# Introdução



## » Data Mining

- A metodologia DM é vista como um passo essencial no processo de descoberta de conhecimento a partir de dados (*Knowledge Discovery from Data - KDD*) (Han e Kamber, 2006).



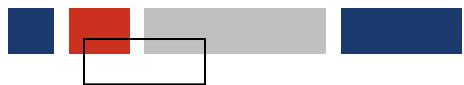


# Introdução



## » Etapas do Processo de Extração de Conhecimento

- Seleção
- Pré-processamento
- Transformação
- *Data mining*
- Interpretação e Avaliação



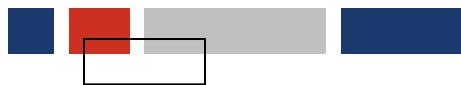
# Introdução



## » Os algoritmos de *Data Mining*

- Os algoritmos de DM que tratam grandes quantidades de informação, podem hoje gerir informação dispersa e diversa com milhões de instâncias de forma rápida e funcionar de forma contínua.
- Os algoritmos aprendem a induzir uma hipótese com a experiência passada.





# Introdução



## » Aplicações

- Banca

- Determinar as situações em que se deve conceder crédito bancário
- Determinar casos de fraude em cartões de crédito

- Economia

- Compreender os hábitos de consumo numa cadeia de supermercados

- Transportes

- Determinar a distribuição dos horários entre os vários percursos

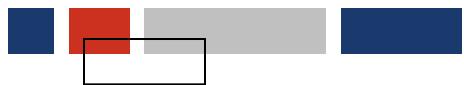
- Medicina

- Identificar terapias de sucesso em diferentes doenças

- Seguros

- Prever quais os consumidores que comprarão novas apólices





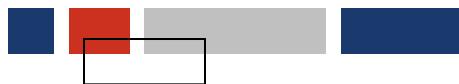
# Introdução



## » *Data Mining e outras áreas*

- *KDD (Knwoledge Discovery from Databases)*: descoberta de conhecimento a partir de bases de dados.
- *Machine learning*: muita da descoberta de padrões estruturais nos dados é feita através de algoritmos de aprendizagem “automática” (*machine learning*).





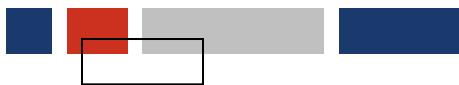
# Introdução



## » *Data Mining Vs. Estatística*

- *Data Mining* requer compreensão do problema e do “negócio”;
- Menos baseada na “mecânica da técnica”;
- Não requer muitos pressupostos acerca dos dados;
- Consegue encontrar padrões em grandes quantidades de dados.
- Os modelos estatísticos precisam de pressupostos sobre os dados;
- Tende a basear-se nos processo de amostragem;
- As técnicas não se encontram otimizadas para grandes quantidades de dados;
- Necessita de grandes competências em conhecimentos de estatística.



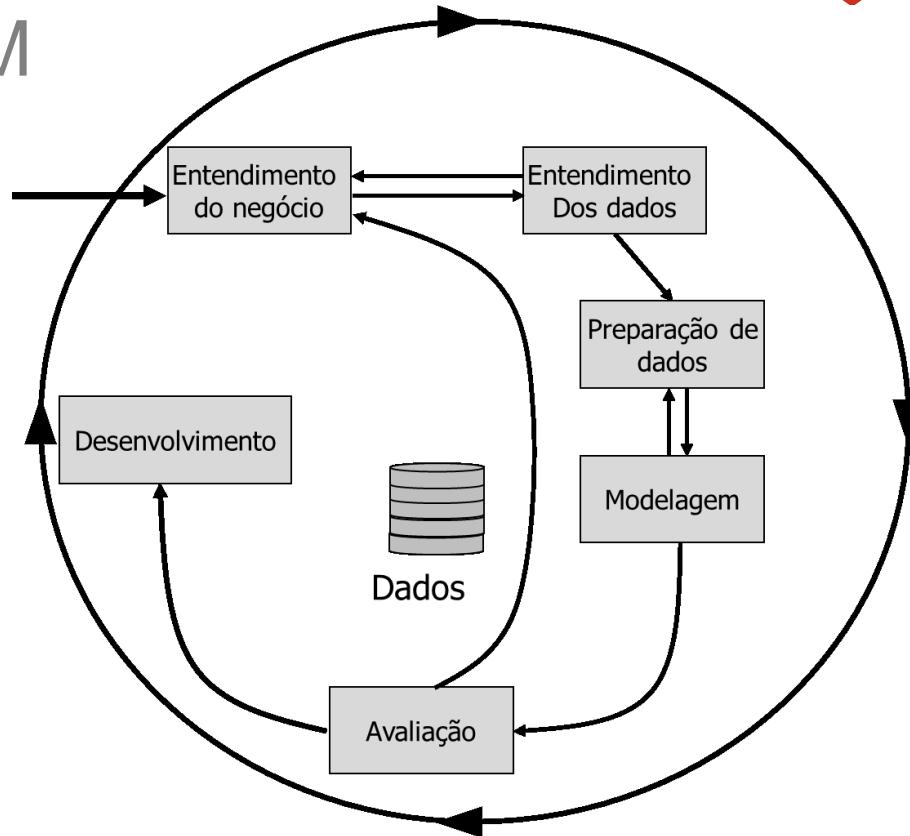


# Introdução

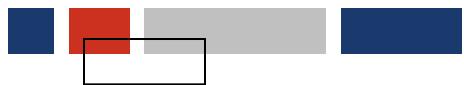


## » Modelo CRISP-DM

*Cross Industry Standard  
Process for Data Mining*  
(Chapman et al, 2000)



In Augusto et al., Marketing Intelligence,  
adaptado de Chapman et al. (2000)



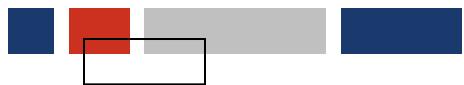
# Introdução



## » Modelo CRISP-DM

- Este modelo ajuda na implementação de um projeto de *data mining* sendo independente do tipo de aplicação (indústria, serviços, etc)
- Trata-se de uma metodologia-tipo que é implementável em várias ferramentas informáticas
- Trata-se de uma metodologia flexível (as fases não têm de ser todas seguidas rigorosamente, sendo possível a introdução de novas fases)





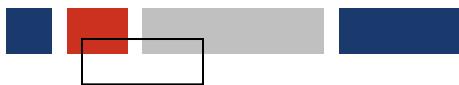
# Introdução



## » Principais Tarefas do *Data Mining* (exemplos)

- Descoberta de Padrões:** encontrar padrões no comportamento dos clientes de forma a classificá-los como compradores ou curiosos.
- Descoberta de Associações:** nas compras de supermercados, encontrar associações de produtos que é habitual os consumidores combinarem no cesto (*market basket analysis*)
- Classificação:** como podemos caracterizar os melhores dias para jogar ténis?
- Agrupamento (*Clustering*):** que grupos podemos formar com as empresas nossas clientes (segmentação)
- Previsão:** como vão evoluir as vendas nos próximos 2 trimestres?





# Introdução

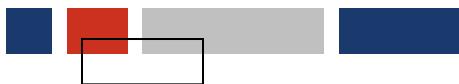


## » Os dados (exemplo)

		Atributos		
		A	B	C
O b j e t o s	1	F	69	70
	2	V	89	90
	...	...	...	...
	n	V	75	70

Exemplos:

- IES
- Censos
- Conjuntura
- IDEF
- etc

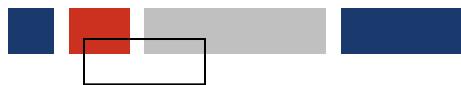


# Introdução



## » Os dados (exemplo)

Objetos	Atributos Preditivos			Atributo Alvo	Exemplos:
	A	B	C		
	Categoria				
1	F	69	70	Sim	- IE (desemprego)
2	V	89	90	Não	- Insolvências
...	...	...	...	...	
n	V	75	70	?	



# Introdução



## » Aprendizagem

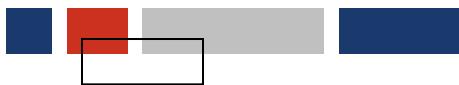
### ▪ **Supervisionada**

- Aprende com exemplos pré-classificados;
- Durante o processo de treino é dado ao sistema a resposta correta.

### ▪ **Não Supervisionada**

- É dado um conjunto de exemplos com o objetivo de estabelecer a existência de categorias, classes ou clusters nos dados.

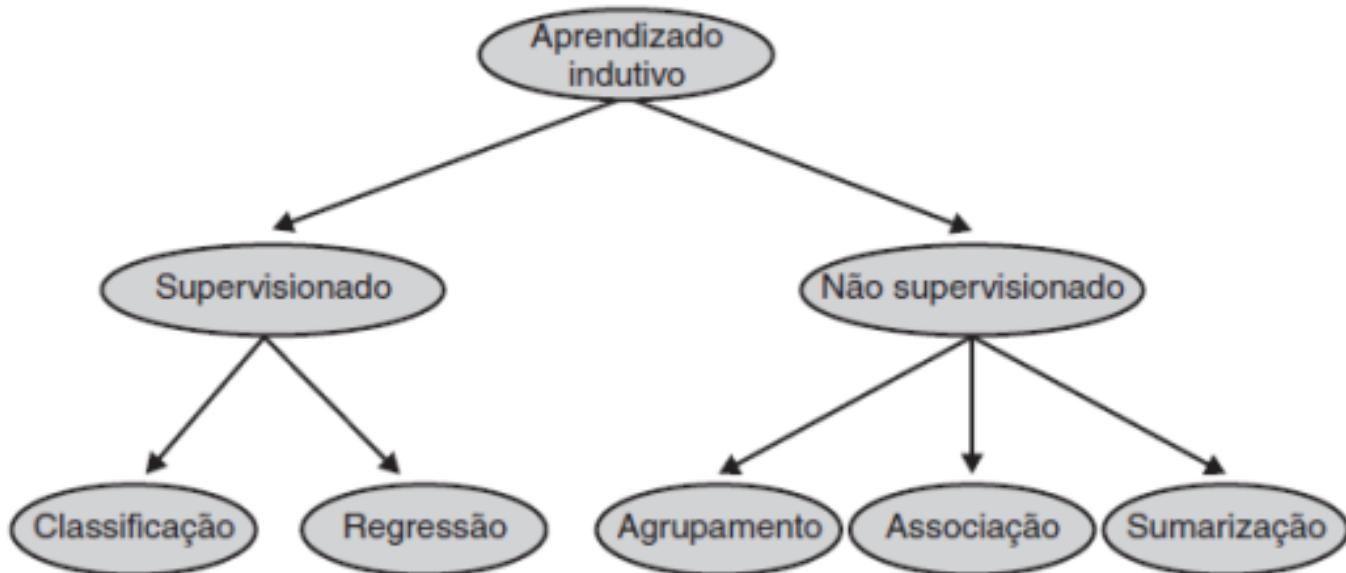




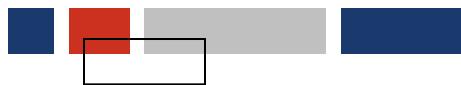
# Introdução



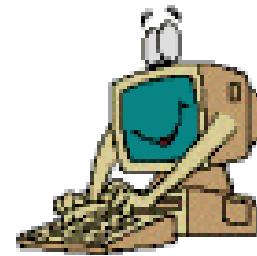
## » Aprendizagem



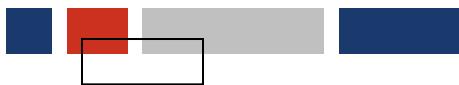
Adaptado de Gama *et al.* (2012)



# Exercício



» Exercício 1.1. e 1.2



# Introdução

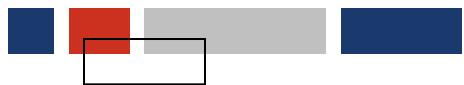


## » O software R\*



- O R é uma linguagem e ambiente de computação estatística e construção de gráficos;
- Dispõe de um conjunto de funcionalidades para manipulação de dados, cálculos e apresentação gráfica.
- O R surge como um importante e poderoso veículo de análise interactiva de dados. Rapidamente se estende a partir de um crescente conjunto de livrarias (packages).
- Os *packages* estão disponíveis por CRAN (Comprehensive R Archive Network, <http://CRAN.R-project.org>) e recomenda-se a sua instalação para explorar áreas mais específicas da análise estatística.

\* slides adaptados do curso de R- nível básico de Rita Sousa e Pedro Campos, 2010



# Introdução



## » O software *R* - Vantagens

- De distribuição gratuita, muito voltado para a análise e tratamento estatístico de dados;
- Compatível com diversas plataformas: UNIX, Windows e MacOS;
- Permite a ligação a interfaces de diferentes formatos: Excel, Access, SPSS, SAS, SQL Server...
- É *Open Source*, permitindo ao utilizador aceder ou alterar funcionalidades existentes, bem como criar novas.





# Introdução



## » O software *R* - Desvantagens

- Interação, com o utilizador, baseada numa janela de comandos;
- Exige o recurso a programação;
- Apesar de existirem muitas facilidades de entreajuda na comunidade de utilizadores do R, esta linguagem não tem suporte técnico assegurado.

```
R version 2.4.1 (2006-12-18)
Copyright (C) 2006 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> █
```

# Introdução

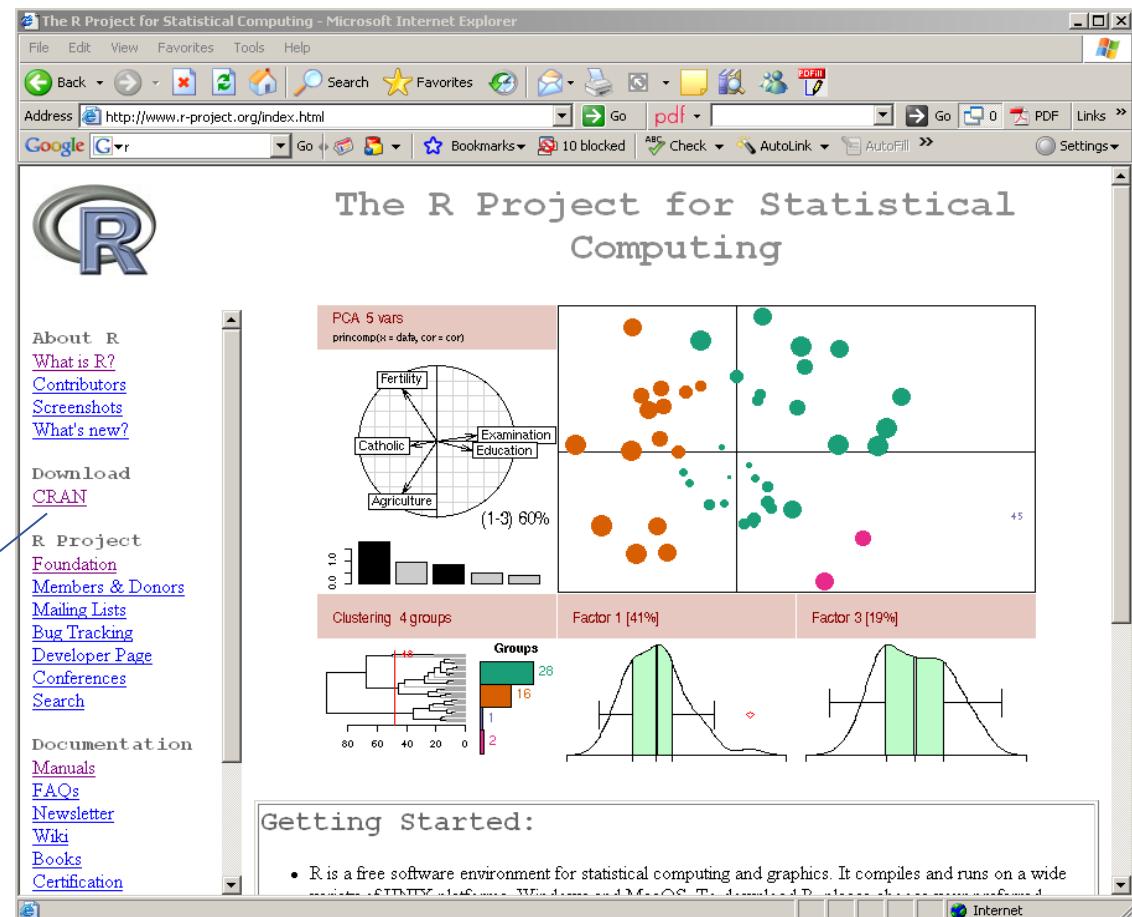


## » O software R

- <http://www.r-project.org/>

Página principal, onde pode aceder a todos os recursos disponíveis do R.

Permite fazer o *download* da aplicação (com as funcionalidades base) e dos vários *packages* adicionais.





# Introdução



## » O software R

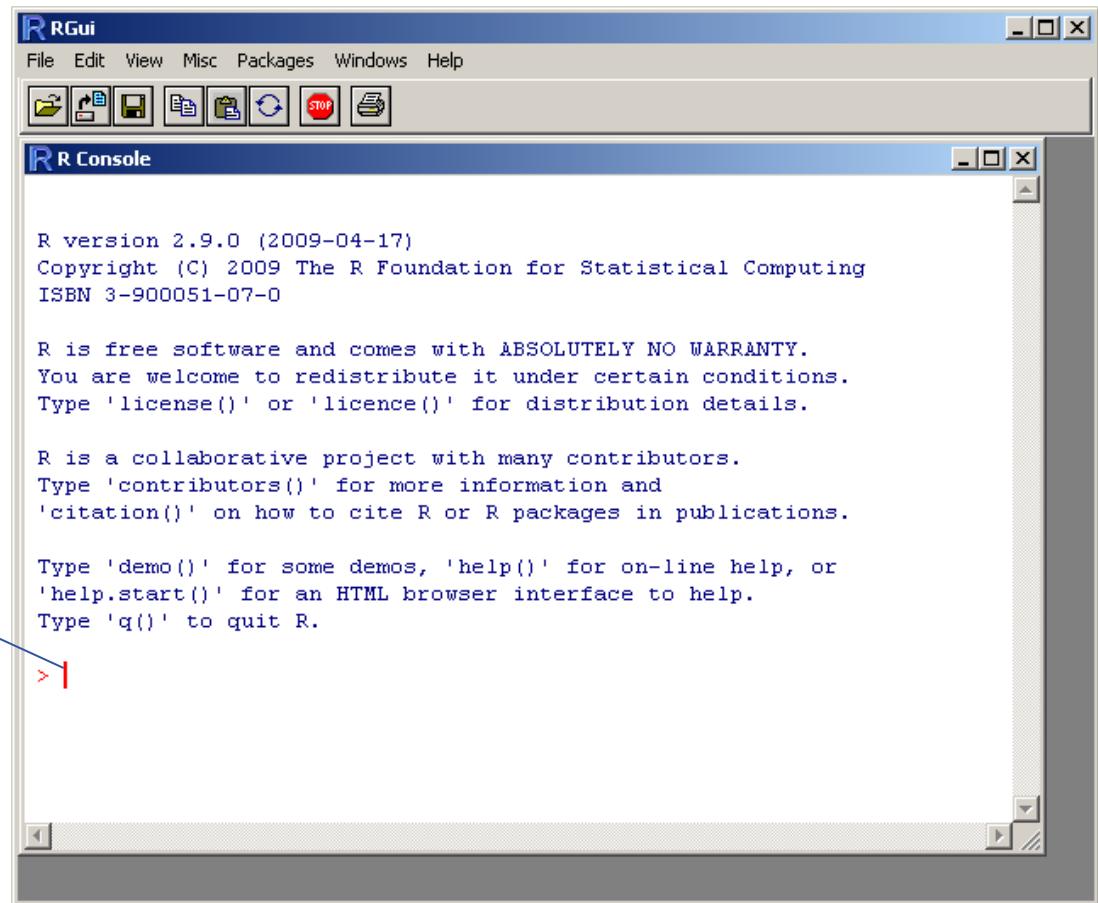


### Primeiro écran



A última versão é a 3.0.3  
(Março 2014)

Ao iniciar o R dá acesso à janela onde podem ser usados os comandos de R, logo após o cursor de *prompt >*.



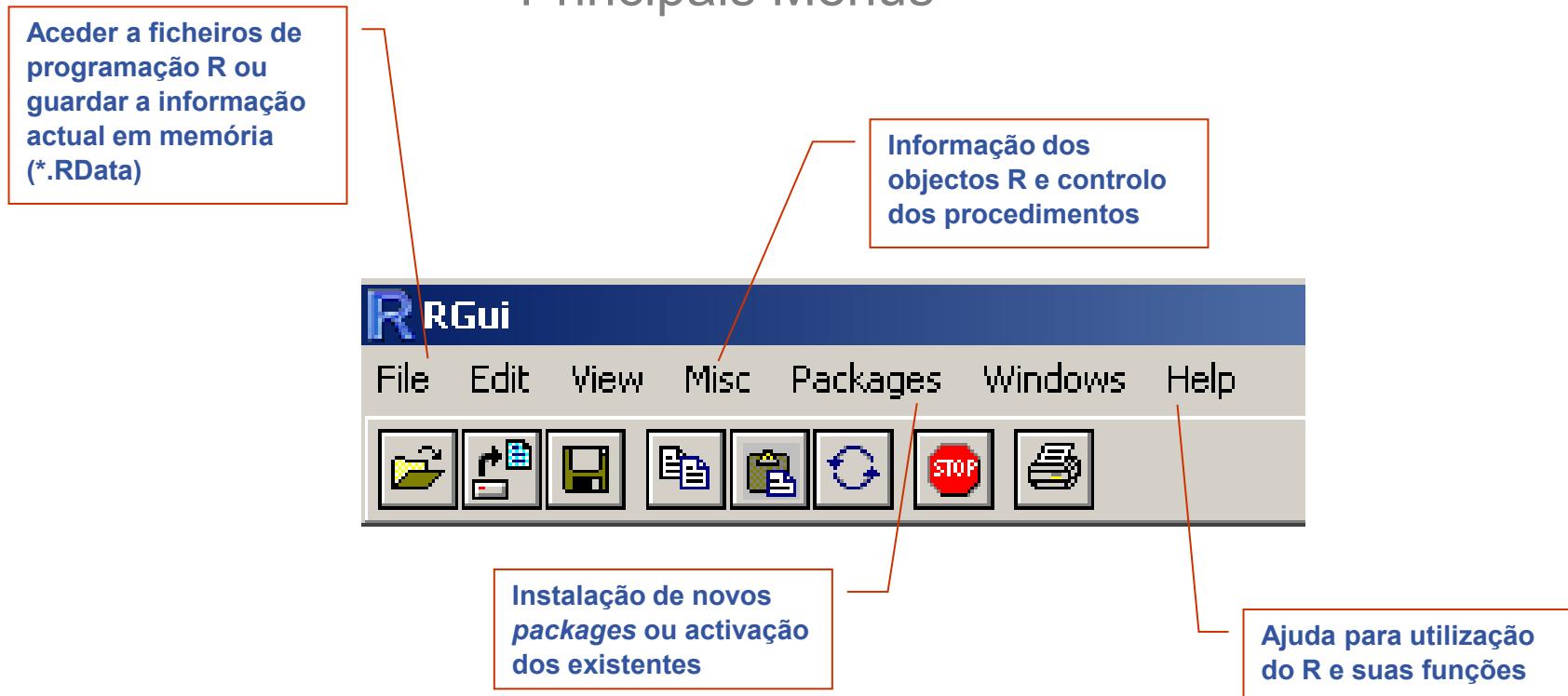


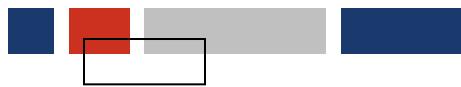
# Introdução



## » O software R

### Principais Menus





# Introdução



## » O software R



### Comandos Principais

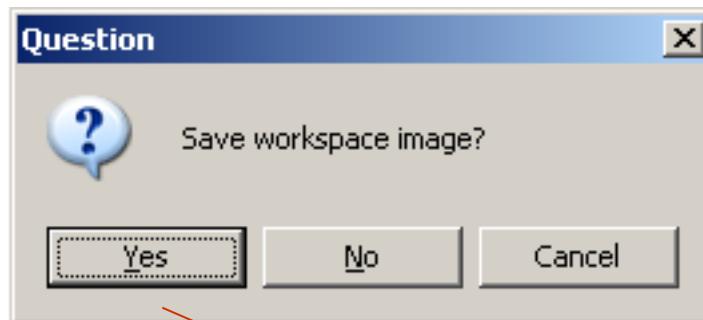
- Informação da versão atual do R

> `R.version`

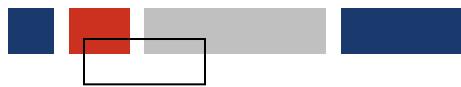
```
platform      i386-pc-mingw32
arch          i386
:
language      R
version.string R version 2.6.2 (2008-02-08)
```

- Sair do R

> `q( )`



Ao sair é possível guardar a informação actual, que é lida automaticamente na sessão seguinte



# Introdução



## » O software R

### Comandos Principais

- Os ficheiros de sessão são sempre guardados na diretoria atual de trabalho, que pode ser consultada:

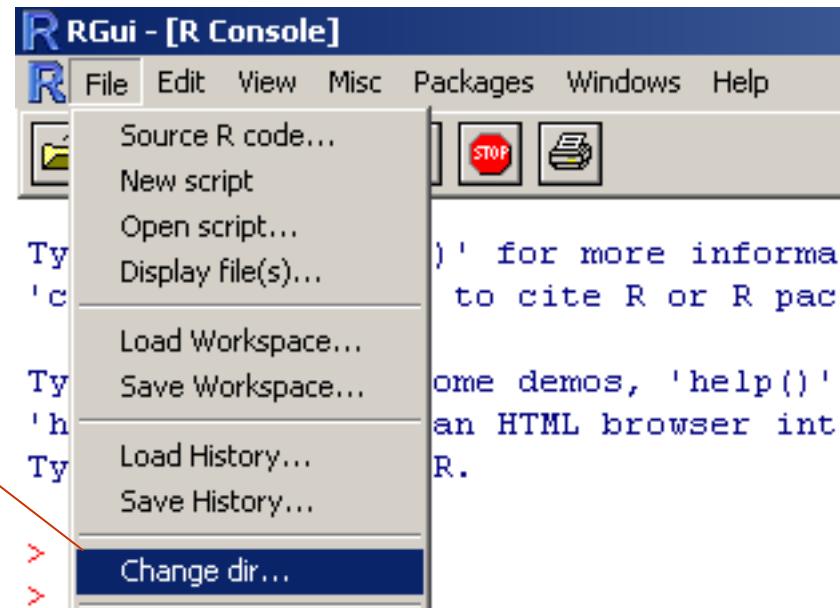
```
> getwd() [1] "C:/Documents and Settings/.../My Documents"
```

- Alterar a diretoria de trabalho

```
> setwd("C:\\Formacao\\CursoR")
```

```
> setwd("C:/Formacao/CursoR")
```

Opção alternativa





## » O software R



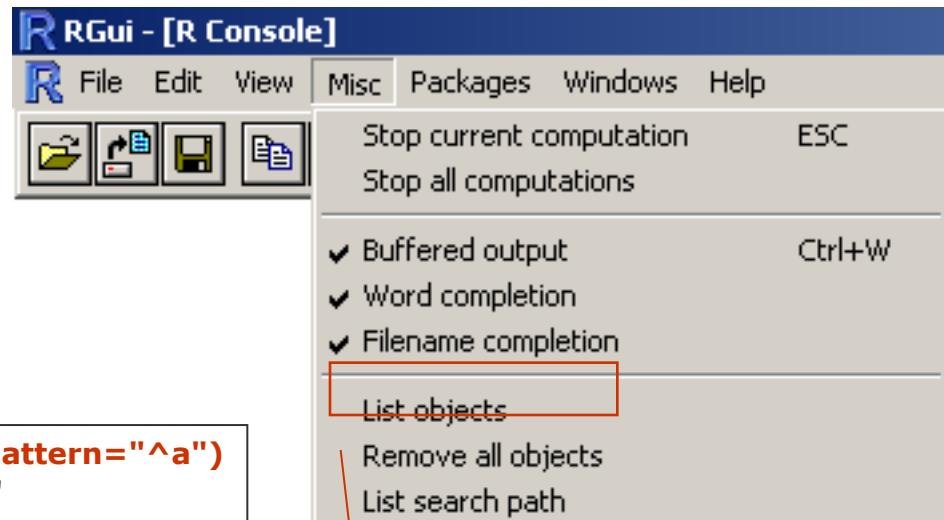
### Comandos Principais

- Listagem de todos os objetos criados no R\*

```
> ls()
```

```
> a = 1  
> b = 2  
> ls()  
[1] "a" "b"
```

```
> objects()
```



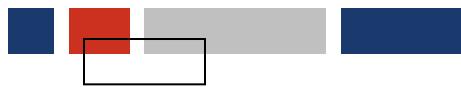
- Objetos que começam por “a”

```
> ls(pattern="^a")
```

```
> ls(pattern="^a")  
[1] "a"
```

```
> objects(pattern="^a")
```

\* Opção alternativa



# Introdução



## » O software R



### Comandos Principais

- Remover um objeto específico

> rm(a)

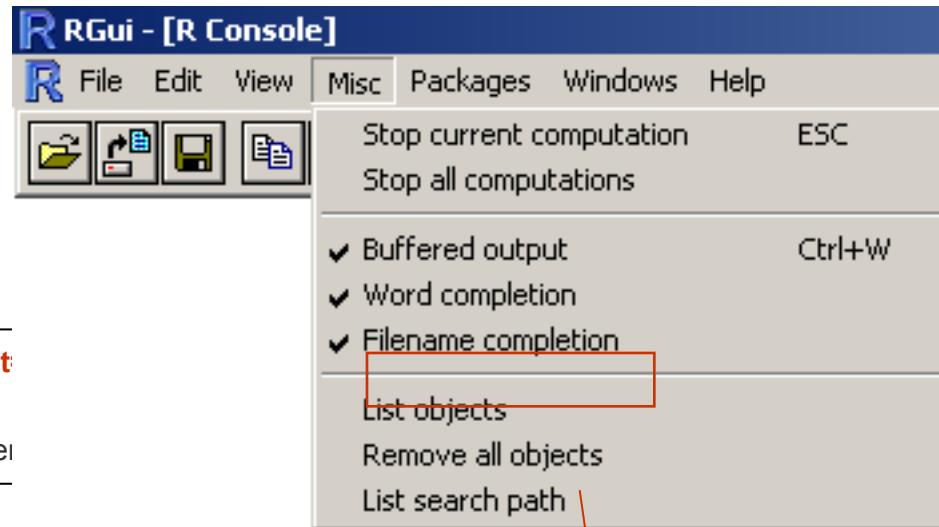
```
> rm(a)
> ls()
[1] "b"
```

- Remover todos os objectos\*

> rm(list=ls( ))

> rm(list=objects( ))

```
> rm(list
> ls()
character)
```



\* Opção alternativa



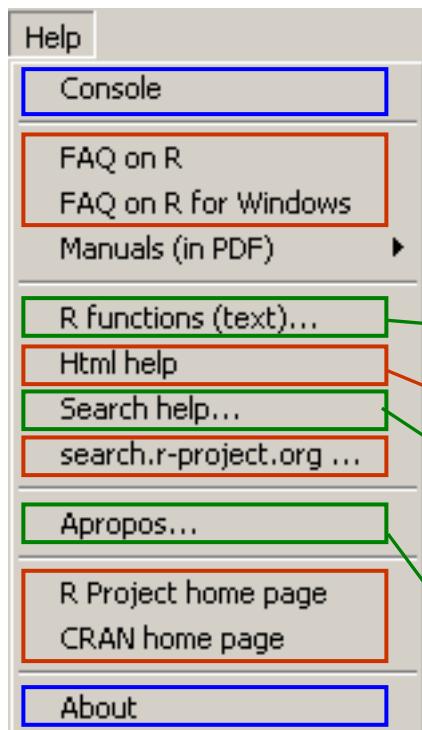
# Introdução



## » O software R



### Opções de Ajuda



- Opções de ajuda sobre a aplicação
- Opções de ajuda com acessos a partir do browser
- Opções de ajuda na janela de comandos do R

`> help("function")`

Apresenta a ajuda relativa à função especificada

`> help.start()`

Dá acesso a informação auxiliar a partir do browser

`> help.search("text")`

Procura as funções cujo nome, detalhes ou descrição contenha o texto indicado

`> apropos("text")`

Procura as funções cujo nome contenha o texto indicado



# Introdução

## » O software R



### Comandos de Ajuda



- Mostra um exemplo de aplicação de uma dada função

> `example(function)`

```
> example(rm)
> tmp <- 1:4
> ## work with tmp and cleanup
> rm(tmp)
```

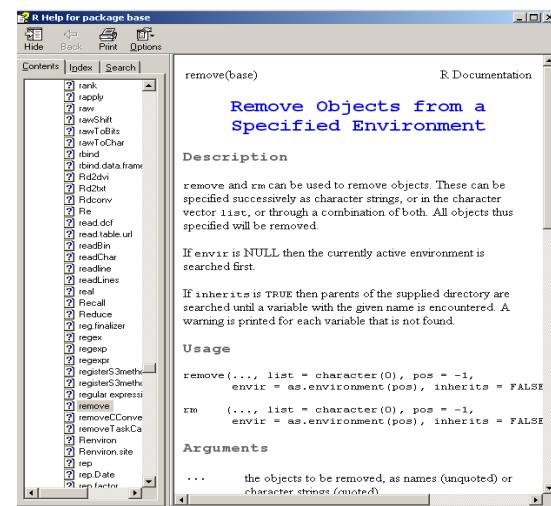
❖ Sinal de atribuição,  
equivalente ao =  
<-

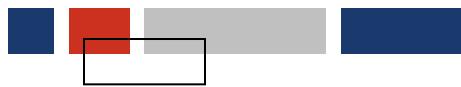
- Mostra os detalhes de uma dada função

> `help(function)`

> `?function`

> **help(rm)**
> ?rm





# Introdução



## » O software R



### Principais Operadores

'+'	Soma
' - '	Subtração
' * '	Multiplicação
' %*% '	Multiplicação de matrizes
' / '	Divisão
' ^ '	Exponenciação
' %/ % '	Divisão Inteira
' %% '	Resto da divisão inteira
' : '	Criação de sequências

```
> 8+3  
[1] 11  
> 8-3  
[1] 5  
> 8*3  
[1] 24  
> 8/3  
[1] 2.666667  
> 8^3  
[1] 512  
> 8%/%3  
[1] 2  
> 8%%3  
[1] 2  
> 8:3  
[1] 8 7 6 5 4 3  
> 3:8  
[1] 3 4 5 6 7 8
```



# Introdução

## » O software *R*



### Regras de Sintaxe



- O R, como a maioria das linguagens que nasce em ambientes UNIX, é **case sensitive**, pelo que por exemplo as letras ‘a’ e ‘A’ podem corresponder a diferentes variáveis.
- O R **ignora espaços**, ou seja, ‘8+3’ e ‘8+ 3’ dão origem exatamente ao mesmo resultado.
- Os **comandos** devem ser **separados** por ‘;’ ou por uma **nova linha**.
- Podemos **agrupar comandos**, para serem executados em simultâneo, se estiverem entre chavetas ‘{ }’.
- O ‘#’ é utilizado para **comentários**.
- Quando um **comando não** está **completo**, o R coloca o sinal de ‘+’ na linha seguinte, permitindo que este seja completado.





# Introdução

## » O software R



### Alguns exercícios



```
> x <- c (3.5,1.4,5,2.6,7,4.8)
> x
[1] 3.5 1.4 5.0 2.6 7.0 4.8
```

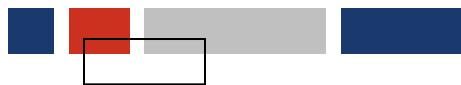
```
> x
[1] 3.5 1.4 5.0 2.6 7.0 4.8
> y <- 2*x + 5
> y
[1] 12.0 7.8 15.0 10.2 19.0 14.6
```

```
> x <- c (3.5,1.4,5,2.6,7,4.8)
> x
[1] 3.5 1.4 5.0 2.6 7.0 4.8
> x > 3
[1] TRUE FALSE TRUE FALSE TRUE TRUE
```

```
> a <- c(1,2.5,4,NA)
> a
[1] 1.0 2.5 4.0 NA
> a+2
[1] 3.0 4.5 6.0 NA
> is.na(a)
[1] FALSE FALSE FALSE TRUE
```

```
> b <- a[2:4]
> b
[1] 2.5 4.0 NA
> b[3] <- 1.7
> b
[1] 2.5 4.0 1.7
> a[-c(3,4)]
[1] 1.0 2.5
```





# Introdução

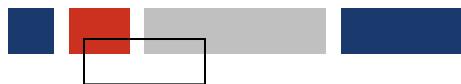


## » O software *Rapid Miner*



<http://rapid-i.com/content/view/181/190/>

Software open source para *Data Mining*

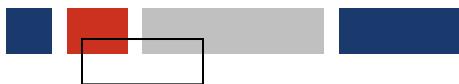


# Introdução



## » O software *Rapid Miner*

- Software que funciona com base em fluxograma
- Baseado em processos
- Contém **operadores** que podem ser parametrizados
- Outros softwares
  - Python
  - Knime
  - Weka
  - Etc.

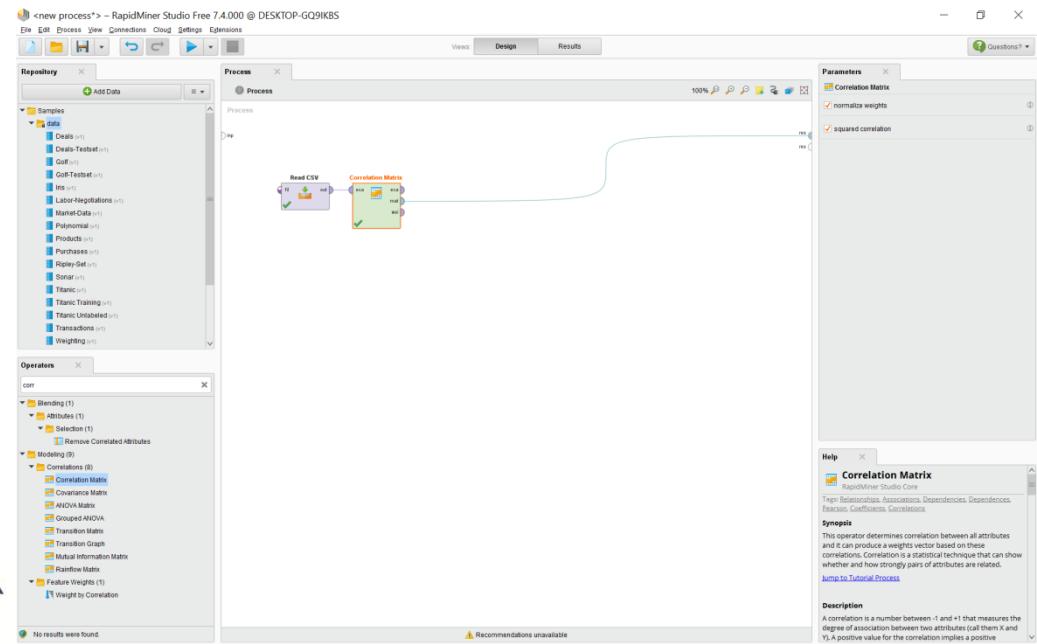


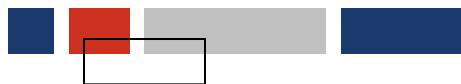
# Introdução



## » O software *Rapid Miner*

- Exemplo
  - Operador Read csv (dados fogos.csv)
  - Operador correlation matrix





# Preparação dos Dados



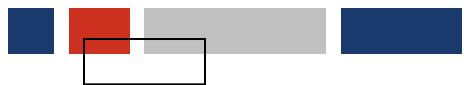
## » Enquadramento

## » Análise de Dados

- Dados
- Tipo de atributos e escala
- Tratamento de Dados

## » Técnicas de Pré-processamento

- Limpeza dos Dados (*Data Cleaning*)
- Redução do Número de Atributos
- Integração de Dados
- Transformação de Dados
- Amostragem
- Redução da Dimensão



# Preparação dos Dados



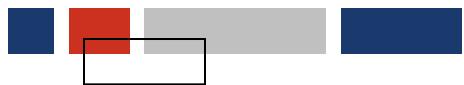
## » Enquadramento

A cada dia é gerada uma enorme quantidade de dados. Os algoritmos de Machine Learning (ML) ou Aprendizagem Máquina (AM) nem sempre são possíveis de utilizar diretamente nesses dados.

Antes de aplicar qualquer algoritmo aos dados é importante fazer uma **análise preliminar dos dados** para melhor compreender e estruturar o problema. Para adaptar os conjuntos de dados à utilização de algoritmos de AM são frequentemente utilizadas técnicas de **pré-processamento**.

A preparação dos dados, que inclui esta análise preliminar e de pré-processamento ocupa cerca de 90% do tempo de resolução do problema.





# Preparação dos Dados



## » Análise preliminar de Dados

O estudo de uma base de dados é iniciado por técnicas estatísticas destinadas a descrever, explorar e encontrar padrões na coleção de dados em estudo. Esta fase da análise de dados onde estes são organizados em tabelas e gráficos e onde se calculam algumas características como a moda, a mediana, a média, o desvio padrão, entre outras é denominada por **análise exploratória ou estatística descritiva**.



# Preparação dos Dados

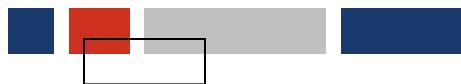


## » Dados

Os **dados estruturados** são geralmente a fonte de um processo de *data mining*. A representação de dados estruturados é, normalmente, na forma de um quadro, em que as colunas são características de objetos armazenados numa tabela e as linhas são valores dessas características (Kantardzic, 2003).

Na literatura é frequente usarem-se os termos objetos, amostras, instâncias, exemplos ou casos para as **linhas** e atributos, variáveis ou características para as **colunas**.

- Objeto: corresponde a uma ocorrência dos dados.
- Atributo: está associado a uma propriedade do objeto.

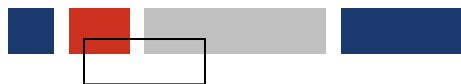


# Dados



- » Os dados podem ser **rotulados** ou **não rotulados**.
- » No caso de **dados rotulados**, além dos valores de atributos de entrada (também denominados atributos preditivos), é apresentado um **atributo especial** denominado atributo alvo, que representa a característica de interesse sobre a qual se deseja fazer previsões.

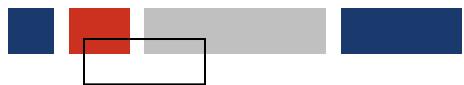
Atributos Preditivos				Atributo Alvo	
	A	B	C	Categoria	
O b j e t o s	1	F	69	70	Sim
	2	V	89	90	Não
	...	...	...	...	...
	$n$	V	75	70	?



# Dados



- » Na presença de dados rotulados o objetivo consiste em utilizar os dados fornecidos para predizer o valor do atributo especial para novas instâncias. Este tipo de aprendizagem é conhecido como **aprendizagem supervisionada**.
  
- » No caso de dados não rotulados o objetivo é simplesmente extrair o máximo de informação a partir dos dados disponíveis. A utilização deste tipo de dados é conhecida como **aprendizagem não supervisionada** (Bramer, 2007).



# Tipo de Atributos



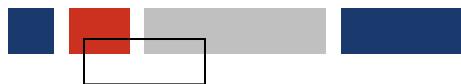
## » Atributos Qualitativos

Representam a informação de características que não se conseguem medir, ou seja, descrevem características não numéricas da população.

## » Atributos Quantitativos

Representam a informação resultante de características que se conseguem medir, ou seja, descrevem características numéricas da população.

- Discretos: assumem um número finito ou infinito numerável de observações ;
- Contínuos: assumem valores num intervalo ou sub-intervalo real, tomando uma infinidade de valores.



# Escala



## » Qualitativos

- Escala nominal

Ex: NUTS, códigos postais, estado civil, sexo, etc.

- Escala ordinal

Ex: Escalão do número de pessoas ao serviço, escalão do volume de negócios, etc.

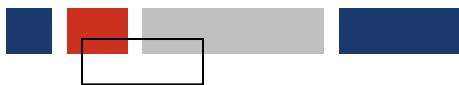
## » Quantitativos

- Escala Intervalar

Ex: escalas de temperatura Celsius e Fahrenheit.

- Escala de Razão

Ex: número de pessoas ao serviço, volume de negócios, idade, salário, preço, etc.



# Tratamento de Dados



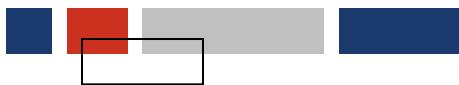
## » Tratamento de dados **qualitativos**

- A partir dum conjunto de dados, identificar o atributo a estudar, bem como o conjunto de valores (categorias) que o atributo pode assumir.
- Frequência absoluta: número de vezes que cada uma das categorias do atributo foi observada.
- Frequência relativa: frequência absoluta/nº total de observações.

## » Representação gráfica de dados **qualitativos**

- Gráfico de barras
- Gráfico circular





# Tratamento de Dados

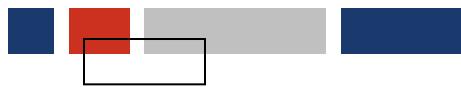


## » Tratamento de dados **quantitativos**

- Tabela de Frequências (no caso de dados quantitativos contínuos os dados terão que ser agrupados)
- Medidas de localização
- Medidas de dispersão
- Medidas de assimetria e achatamento

## » Representação gráfica de dados **quantitativos**

- Gráfico de barras (dados quantitativos discretos)
- Histograma (dados quantitativos contínuos)
- Polígono de frequências (dados quantitativos contínuos)
- Diagrama de extremos e quartis



# Dados Univariados



## » Medidas de localização

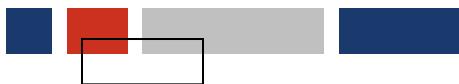
- Média;
- Mediana;
- Moda;
- Quartis, Decis e Percentis.

## » Medidas de dispersão

- Desvio padrão;
- Variância;
- Coeficiente de variação;
- Amplitude;
- Amplitude interquartil.

## » Medidas de assimetria e achatamento

- Coeficiente de simetria;
- Coeficiente de achatamento.



# Dados Bivariados



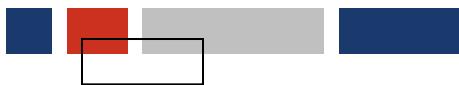
## » Relacionamento entre dois atributos

- Covariância;
- Correlação.

A medida de covariância é afetada pela dimensão dos atributos avaliados. Dois atributos com dimensão elevada podem apresentar um valor de covariância maior que dois atributos mais semelhantes entre si, mas de menor dimensão.

A medida de correlação elimina esse problema retirando a influência da dimensão.

A correlação é mais utilizada para explorar dados multivariados que a covariância.



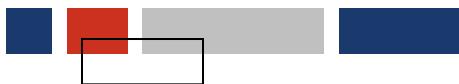
# Pré-Processamento



## » Técnicas de Pré-processamento

- Limpeza dos Dados (*Data Cleaning*)
- Redução do Número de Atributos
- Integração de Dados
- Transformação de Dados
- Amostragem
- Redução da Dimensão



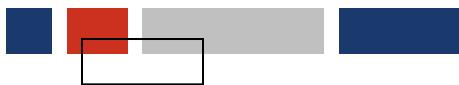


# Pré-Processamento



## » Limpeza dos Dados (*Data Cleaning*)

- ruidosos: possuem erros ou valores que são diferentes do esperado;
- inconsistentes: não combinam ou contradizem valores de outros atributos do mesmo objeto;
- redundantes: quando dois ou mais objetos têm os mesmos valores para todos os atributos ou dois ou mais atributos têm os mesmos valores para dois ou mais objetos;
- incompletos: com valores ausentes em alguns dos atributos.



# Limpeza dos Dados



## » Limpeza de dados ruidosos

- Eliminar os objetos com valores *outliers*;
- A presença destes dados pode resultar em estatísticas e análises incorretas.
- A tabela seguinte apresenta um exemplo de dados ruidosos.

A	B	C	Categoria
F	69	70	Sim
V	89	90	Não
F	<b>500</b>	68	Não
V	75	70	Sim



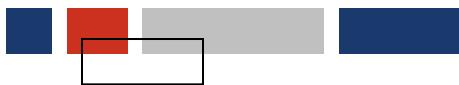
# Limpeza dos Dados



## » Limpeza de dados inconsistentes

- Algoritmos simples podem verificar automaticamente se relacionamentos existentes entre atributos são violados.
- No caso do conjunto de dados não ser muito grande, os dados inconsistentes podem ser removidos manualmente.
- A tabela seguinte apresenta um exemplo de dados inconsistentes.

A	B	C	Categoria
F	69	70	Sim
V	89	90	Não
F	69	70	Não
V	75	70	Sim



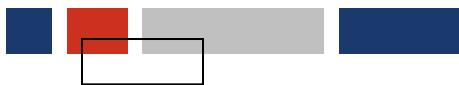
# Limpeza dos Dados



## » Limpeza de dados redundantes

- Um objeto é redundante quando existe semelhança com outro objeto do mesmo conjunto de dados.
- Um atributo é redundante quando o valor para todos os objetos pode ser deduzido a partir do valor de um ou mais atributos.
- Identificar e eliminar os dados redundantes.
- A tabela seguinte apresenta um exemplo de dados redundantes.

A	B	C	Categoria
F	69	70	Sim
V	89	90	Não
F	69	70	Sim
V	75	70	Sim

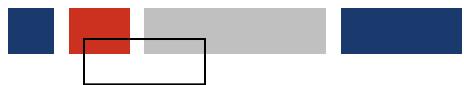


# Limpeza dos Dados



## » Limpeza de dados incompletos

- Eliminar os objetos com valores ausentes;
- Definir e preencher manualmente valores para os atributos com valores ausentes;
- Utilizar algum método ou heurística para definir automaticamente os valores para atributos com valores ausentes;
- Utilizar algoritmos de DM que lidam internamente com valores ausentes.



# Eliminação de Atributos



## » Redução do Número de Atributos

Nem todos os atributos do conjunto original são necessários para prever a característica de interesse. Por exemplo, se os valores dos atributos como ID e Nome não são úteis para prever o atributo alvo, são eliminados.

Quando um atributo claramente não contribui para a estimativa do valor do atributo alvo, ele é considerado irrelevante.

ID	Nome	A	B	C	Atributo alvo
1	Carlos	F	69	70	Sim
2	José	V	89	90	Não
3	Maria	V	75	70	Sim



# Integração de Dados



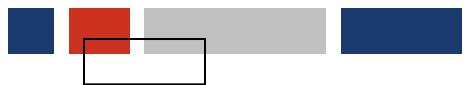
## » Integração

Podem existir dados em diferentes bases de dados para serem utilizados no mesmo problema de DM. Esses dados, distribuídos em diferentes bases de dados, devem ser integrados antes de iniciar o uso de técnicas de DM.

Na integração, é necessário identificar quais os objetos que estão presentes nos diferentes conjuntos a serem combinados.

O processo de integração origina um arquivo de dados (*data warehouse*), que funciona como uma base de dados centralizada.





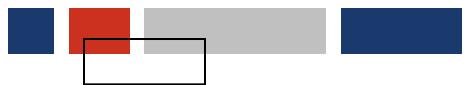
# Transformação de Dados



## » Transformação

Algumas técnicas de DM usam apenas valores numéricos ou valores simbólicos ou têm o seu desempenho influenciado pelo intervalo de variação dos valores numéricos dados.

Podem ser utilizados métodos para converter valores simbólicos em valores numéricos, converter valores numéricos em valores simbólicos e mudar a escala ou intervalo de valores de atributos com valores numéricos.



# Transformação de Dados



## » Converter valores simbólicos em valores numéricos

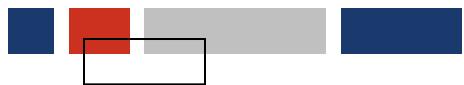
Quando o tipo do atributo é **simbólico** e assume apenas **dois valores**:

- Se os valores indicam a presença ou ausência de uma característica, o valor **0** indica a ausência e o valor **1** a presença.

		A	A'
Falso	0		
Verdadeiro	1		
Falso	0		

		B	B'
Bom	1		
Mau	0		
Mau	0		



# Transformação de Dados



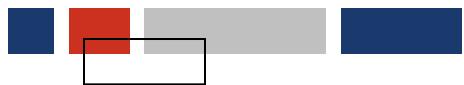
» Converter valores simbólicos em valores numéricos

Quando o tipo do atributo é **simbólico** e assume **mais de dois valores**:

-Se o atributo é nominal, a diferença entre quaisquer dois valores numéricos deve ser a mesma (codificação canónica).

C	C'
Sim	100
Não	010
Talvez	001

D	D'
Mau	0
Suficiente	1
Bom	2



# Transformação de Dados

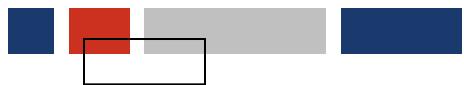


## » Converter valores numéricos em valores simbólicos

Métodos de discretização permitem transformar atributos quantitativos em qualitativos.

Os métodos de discretização podem ser supervisionados ou não supervisionados.

Quando um atributo quantitativo é discretizado, o conjunto de valores possíveis é dividido em intervalos.



# Transformação de Dados



## » Métodos de Discretização Não Supervisionados

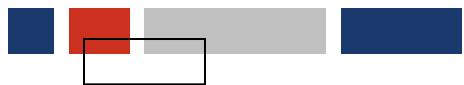
### Discretização com Intervalos Iguais

- Cortar um atributo contínuo em intervalos

### Discretização com Intervalos com Frequências Iguais

- Cada intervalo deve conter um número semelhante de observações.

(Brazdil, 2009a)



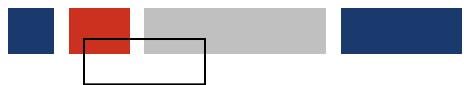
# Transformação de Dados



## » Métodos de Discretização Não Supervisionados

Considerem-se os seguintes dados:

Tempo	Temperatura	Humidade	Vento	Jogar
Chuvoso	71	91	sim	não
Soalheiro	69	70	não	sim
Soalheiro	80	90	sim	não
Nublado	83	86	não	sim
Chuvoso	70	96	não	sim
Chuvoso	65	70	sim	não
Nublado	64	65	sim	sim
Nublado	72	90	sim	sim
Soalheiro	75	70	sim	sim
Chuvoso	68	80	não	sim
Nublado	81	75	não	sim
Soalheiro	85	85	não	não
Soalheiro	72	95	não	não
Chuvoso	75	80	não	sim



# Transformação de Dados



## » Métodos de Discretização Não Supervisionados

### Discretização com Intervalos Iguais

Discretizar o atributo Temperatura com 4 intervalos:

71, 69, 80, 83, 70, 65, 64, 72, 75, 68, 81, 85, 72, 75

# Transformação de Dados



## » No RapidMiner - Discretização com Intervalos Iguais

1- Ler o ficheiro  
com os dados:  
'weather.numeric'

The screenshot shows the RapidMiner interface. On the left, the Operators palette is open, with the 'Read' operator category highlighted by a red circle. A red arrow points from this circle to the 'Read CSV' operator icon in the center workspace. In the center workspace, there is a single 'Read CSV' operator with a yellow warning icon. To the right, the 'Parameters' palette shows the configuration for reading a CSV file, with a red arrow pointing to the 'Import Configuration Wizard...' button. A modal window titled 'Data import wizard - Step 1 of 4' is displayed, showing the file 'weather.numeric.csv' selected for import.



# Transformação de Dados

## » No RapidMiner - Discretização com Intervalos Iguais

The screenshot shows the Data import wizard in RapidMiner Studio across three steps:

- Step 2 of 4:** File Encoding is set to "windows-1252". Column Separation is set to "Comma",". The data preview shows a table with columns: Tempo, Temperatura, Humidade, Vento, and Jogar. The "Use Quotes" checkbox is checked.
- Step 3 of 4:** Annotations are being added to the columns: Name (Tempo), att1 (Temperatura), att2 (Humidade), att3 (Vento), att4 (Jogar), and att5 (empty). The data preview shows the same table with the newly annotated columns.
- Step 4 of 4:** Data types are being assigned: Tempo is polynom., Temperatura is integer, Humidade is integer, Vento is polynom., and Jogar is polynom. The data preview shows the table with these typed attributes.

Red arrows indicate the progression from Step 2 to Step 3 and from Step 3 to Step 4.

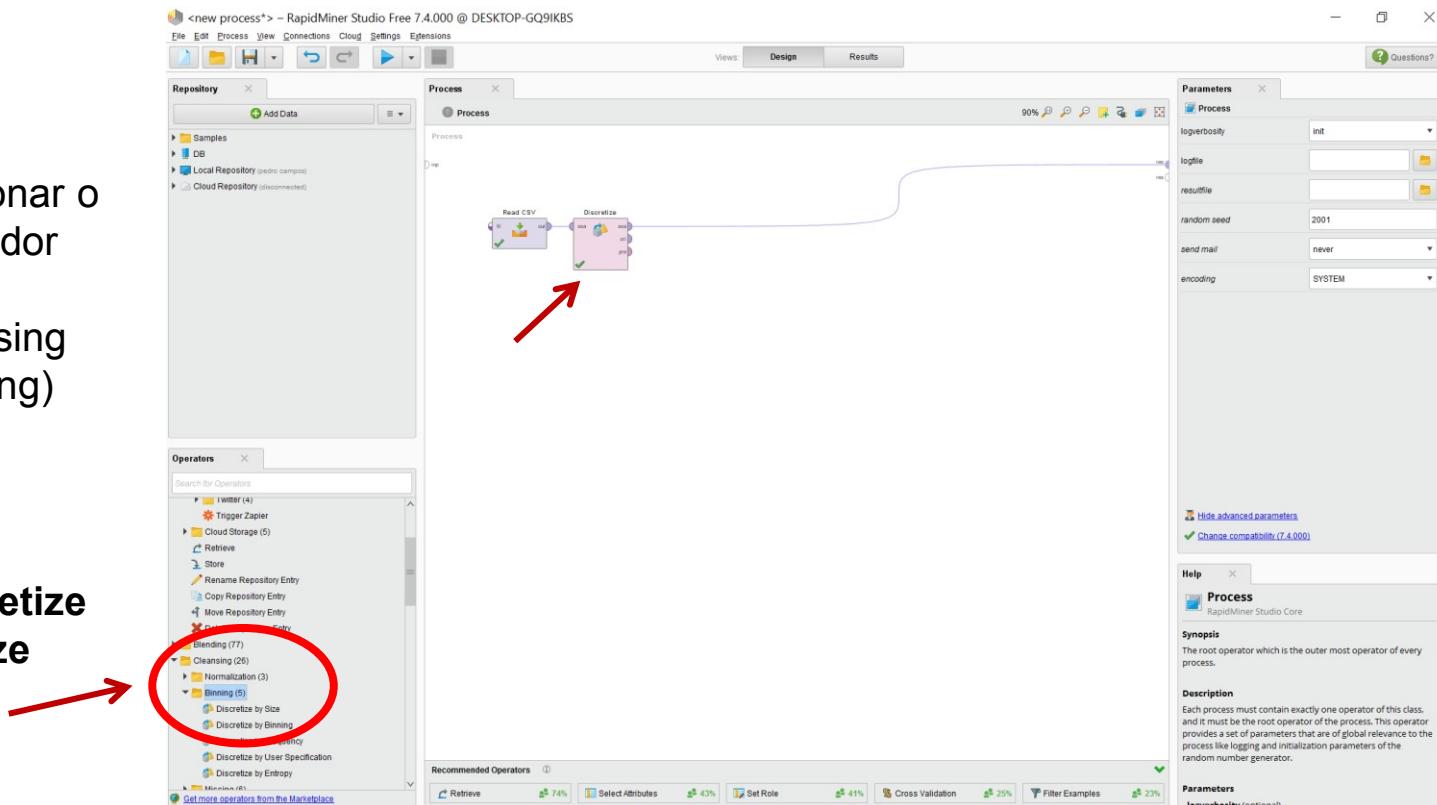
1- Ler o ficheiro  
com os dados:  
'weather.numeric'



# Transformação de Dados

## » No RapidMiner - Discretização com Intervalos Iguais

2-  
Adicionar o  
operador  
de  
cleansing  
(binning)



Discretize  
by size



# Transformação de Dados



## » No R - Discretização com Intervalos Iguais

#Ler o ficheiro com os dados

```
> data<- read.csv("weather.numeric.csv", header=T)
```

```
> attach(data)
```

# Cortar o atributo em  $k$  intervalos

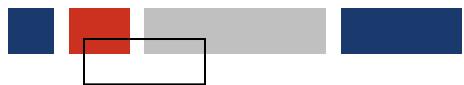
```
> atributo_disc <- cut (atributo, k)
```

# Transformar as categorias observadas em números inteiros

```
> atributo_disc <- cut (atributo, k, labels=FALSE)
```

# Obter a distribuição de frequências das várias categorias.

```
> table(atributo_disc)
```



# Transformação de Dados



## » Métodos de Discretização Não Supervisionados

### Discretização com Intervalos com Frequências Iguais

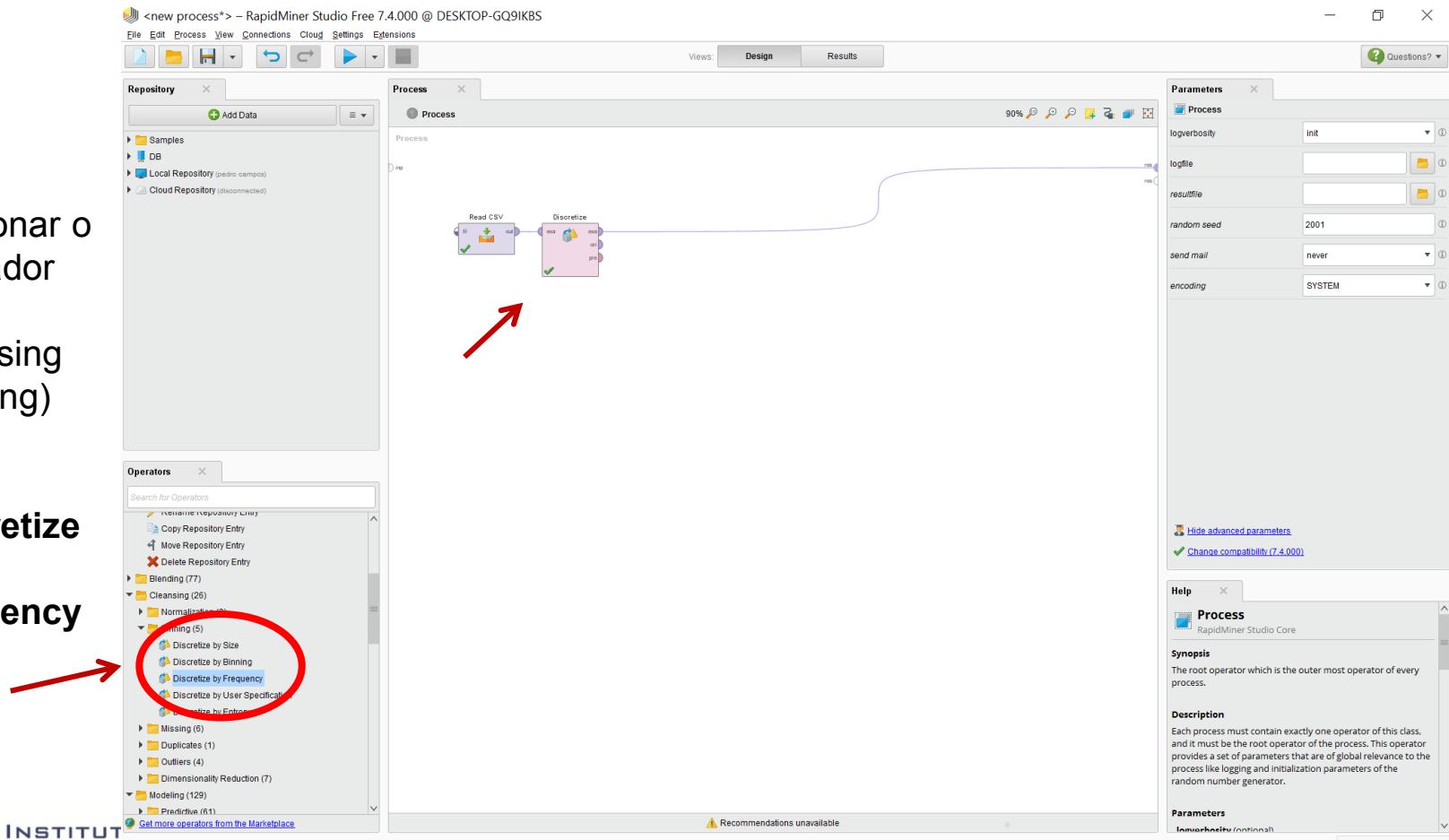
Discretizar o atributo Temperatura:

71, 69, 80, 83, 70, 65, 64, 72, 75, 68, 81, 85, 72, 75

# Transformação de Dados

## » No RapidMiner - Discretização com Intervalos Iguais

2-  
Adicionar o  
operador  
de  
cleansing  
(binning)



**Discretize  
by  
frequency**



# Transformação de Dados



## » No R - Intervalos com Frequências Iguais

```
#Ler o ficheiro com os dados
```

```
> dados<- read.csv("ficheiro.csv", header=T)  
> attach(dados)
```

```
# Determinar vector que possui os pontos de corte.
```

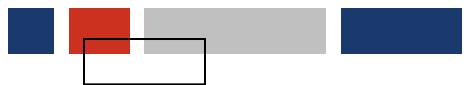
```
> v_pc <- quantile(atributo, seq(from=0, to=1, by=1/k), names=FALSE)
```

```
# Cortar o atributo usando os pontos de corte
```

```
> atributo_disc <- cut(atributo, v_pc, include.lowest=TRUE)
```

```
# Obter a distribuição de frequências das várias categorias.
```

```
> table(atributo_disc)
```



# Transformação de Dados



## » No R – Dados discretizados

```
# Obter um novo conjunto de dados (dados_disc) com os atributos  
discretizados:
```

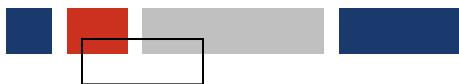
```
> dados_disc <-dados
```

```
# Substituir os valores originais dos atributos pelos discretizados
```

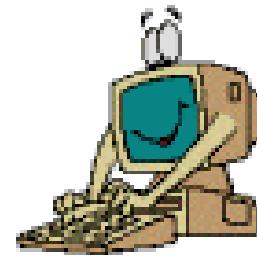
```
> dados_disc$atributo1<- atributo_disc1
```

```
> dados_disc$atributo2 <- atributo_disc2
```

```
> dados_disc
```



# Exercício



## » Exercício 2.1.



# Transformação de Dados



## » Método de Discretização **Supervisionado**

### Discretização baseada na entropia do atributo

- Utilizar o ganho de informação:

$$GI(\text{atributo}) = H(\text{classe}) - H(\text{atributo})$$

O ganho de informação mede o número de bits necessários para definir a classe.

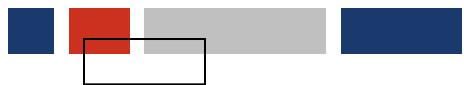
Considere  $X$  uma variável aleatória cujo domínio é  $\{x_1, x_2, \dots, x_i\}$ .

Sendo  $p_1, p_2, \dots, p_i$  as probabilidades de observar cada um dos valores,

a entropia de  $X$  é dada por: 
$$H(X) = -\sum_i p_i * \log_2(p_i)$$

A entropia mede a incerteza associada ao valor de  $X$ .





# Transformação de Dados



## » Método de Discretização **Supervisionado**

Processo:

- 1 – Selecionar o atributo e a classe
- 2 – Obter os pontos de corte
- 3 – Calcular as frequências
- 4 – Calcular a informação para todos os pontos de corte
- 5 – Escolher o melhor ponto de corte



# Transformação de Dados



## » Método de Discretização Supervisionado

Calcular o **ganho de informação** para um atributo **quantitativo**:

- atributo ‘Temperatura’

- 1- Ordenar os valores por ordem crescente
- 2- Cada ponto médio entre duas observações consecutivas é um potencial ponto de corte.
- 3- Calcular o ganho de informação para cada potencial ponto de corte.

Possíveis pontos de corte:

64.5 66.5 68.5 69.5 70.5 71.5 73.5 77.5 80.5 82.0 84.0

Considerar, por exemplo, o ponto de corte:

Temperatura = 80.5

Temperatura	Jogar
64	sim
65	não
68	sim
69	sim
70	sim
71	não
72	sim
72	não
75	sim
75	sim
80	não
81	sim
83	sim
85	não



# Transformação de Dados



## » Método de Discretização Supervisionado

A entropia da classe é:

$$\begin{aligned} H(\text{Jogar}) &= -\sum_i p_i * \log_2(p_i) \\ &= -p_1 * \log_2(p_1) - p_2 * \log_2(p_2) \end{aligned}$$

$$p_1 = p(\text{Jogar} = 'sim') = \frac{9}{14}$$

$$p_2 = p(\text{Jogar} = 'não') = \frac{5}{14}$$

$$H(\text{Jogar}) = -\frac{9}{14} * \log_2\left(\frac{9}{14}\right) - \frac{5}{14} * \log_2\left(\frac{5}{14}\right) = 0.940 bits$$



# Transformação de Dados



## » Método de Discretização Supervisionado

ponto de corte Temperatura = 80.5

$$p(\text{Jogar} = \text{'sim'} | \text{Temperatura} \leq 80.5) = \frac{7}{11}$$

$$p(\text{Jogar} = \text{'não'} | \text{Temperatura} \leq 80.5) = \frac{4}{11}$$

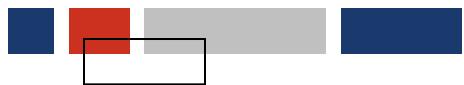
$$H(\text{Jogar} | \text{Temperatura} \leq 80.5) = -\frac{7}{11} * \log_2\left(\frac{7}{11}\right) - \frac{4}{11} * \log_2\left(\frac{4}{11}\right) = 0.926 \text{ bits}$$

$$p(\text{Jogar} = \text{'sim'} | \text{Temperatura} \geq 80.5) = \frac{2}{3}$$

$$p(\text{Jogar} = \text{'não'} | \text{Temperatura} \geq 80.5) = \frac{1}{3}$$

$$H(\text{Jogar} | \text{Temperatura} \geq 80.5) = -\frac{2}{3} * \log_2\left(\frac{2}{3}\right) - \frac{1}{3} * \log_2\left(\frac{1}{3}\right) = 0.918 \text{ bits}$$





# Transformação de Dados



## » Método de Discretização Supervisionado

A **entropia** para o atributo ‘Temperatura’:

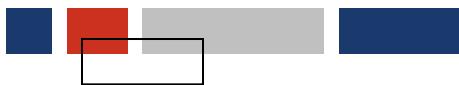
$$p(\text{Temperatura} \leq 80.5) = \frac{11}{14}$$

$$p(\text{Temperatura} \geq 80.5) = \frac{3}{14}$$

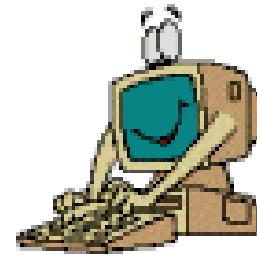
$$H(\text{Temperatura}) = \frac{11}{14} * 0.926 + \frac{3}{14} * 0.918 = 0.926 \text{ bits}$$

O **ganho de informação** do atributo ‘Temperatura’:

$$GI(\text{Temperatura}) = H(\text{classe}) - H(\text{Temperatura}) = 0.940 - 0.926 = 0.014 \text{ bits}$$



# Exercício



## » Exercício 2.2.



# Transformação de Dados



## » No R

```
# Atributo a discretizar
```

```
>atributo<-dados$atributo
```

```
# Capturar a variável objectivo (assumindo que é a última)
```

```
>classe<-names(dados)[length(names(dados))]
```

```
# Ou considerar diretamente a classe:
```

```
>classe<-"atributo_alvo"
```

```
# Eliminar os valores duplicados e ordenar
```

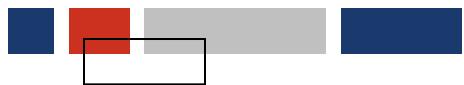
```
>ppcl<-sort(as.numeric(levels(as.factor(atributo))))
```

```
# Obter os possíveis pontos de corte (valor médio entre dois pontos consecutivos)
```

```
>ppc<-numeric(0)
```

```
>for (i in 1:(length(ppcl)-1)) ppc[i]<-((ppcl[i]+ppcl[i+1])/2)
```

```
>ppc
```



# Transformação de Dados



## » No R

```
# Função auxiliar para calcular as frequências
info<-function (freqs) {
  output<-0
  for (i in freqs) output <- output - (i/sum(freqs))*log2(i/sum(freqs))
  if (is.nan(output)) output<-0
  output
}
```

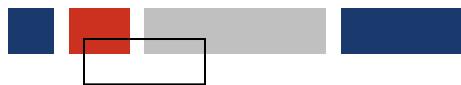


# Transformação de Dados



## » No R

```
# Obter o ganho de informação de cada possível ponto de corte
info.ppc<-numeric(0)
for (i in 1:length(ppc)) {
  distr.anterio_pc <- table( get(classe)[atributo<ppc[i]] )
  distr.depois_pc <- table( get(classe)[atributo>ppc[i]] )
  info1<- sum(distr.anterio_pc) * info(distr.anterio_pc)
  info2<- sum(distr.depois_pc) * info(distr.depois_pc)
  info.ppc[i] <- (info1+info2) / (length(atributo))
}
# Em que posição se encontra o melhor ponto de corte
ix.min.info<-which.min(info.ppc)
# Qual o melhor ponto de corte
pc.melhor <- ppc[ix.min.info]
```



# Transformação de Dados



## » No R

```
#Publicar os resultados
```

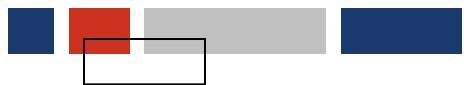
```
cat("Pontos de corte possíveis:",ppc,'\n')
cat("Info de pontos de corte:",round(info.ppc,3),'\n')
cat("Índice de melhor info:", ix.min.info,'\n')
cat("Melhor ponto de corte:", pc.melhor,'\n')
```

```
# Obter o atributo discretizado
```

```
pc<- c(min(atributo), pc.melhor, max(atributo))
atributo_disc<- cut(atributo,pc,include.lowest=TRUE,labels=FALSE)
atributo_disc
```

```
# Substituir os valores originais pelos discretizados
```

```
dados_disc$Atributo_disc<- atributo_disc
dados_disc
```

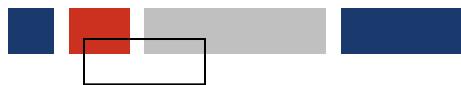


# Transformação de Dados



## » Transformação de Atributos Numéricos

- Quando os valores mínimo e máximo dos valores observados dos atributos numéricos variam muito ou se alguns atributos estão em escalas diferentes, é realizada uma transformação dos dados para evitar que um atributo prevaleça sobre outro.
- No entanto, também podem existir situações em que, por ser importante para a indução de um bom modelo, essa variação deve ser mantida.

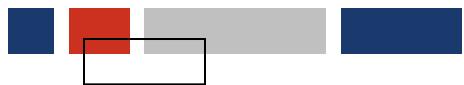


# Amostragem de Dados



## » Amostragem

- Alguns algoritmos de DM podem ter dificuldades em lidar com um elevado número de objetos.
- Selecionar uma amostra que seja representativa do conjunto de dados original.
- Podem ser gerados diferentes modelos de uma mesma população com amostras diferentes se estas não forem representativas.



# Redução da Dimensão

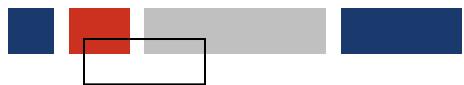


## » Seleção de Atributos

- Permite que sejam considerados os atributos mais relevantes na criação do modelo.

Com a utilização do Weka esta tarefa é facilitada com utilização de alguns algoritmos. Por exemplo:

- *InfoGainAttributeEval*: Avalia o valor de um atributo, medindo o ganho de informação com respeito à classe.
- *Correlation Feature Selection (CFS)*: é uma técnica de seleção de um subconjunto de atributos relevantes para a construção robusta de modelos de aprendizagem. Um bom subconjunto de atributos tem atributos pouco correlacionados entre si e muito correlacionados com o atributo alvo.

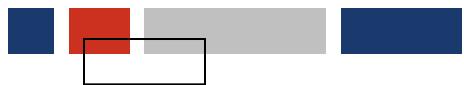


# Modelos Exploratórios



## » Enquadramento

- Modelos não supervisionados.
- Modelos que trabalham sobre exemplos não rotulados.
- Permitem descobrir associações nos dados e encontrar agrupamentos naturais nos dados.



# Modelos Preditivos



## » Enquadramento

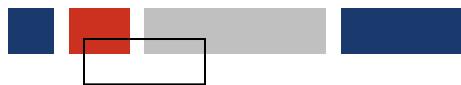
- Modelos supervisionados.
- Modelos que trabalham sobre exemplos rotulados.
- Permitem classificar os dados e “prever” valores dos rótulos (classes) para novos exemplos.
- Rótulos qualitativos: problema de classificação.

Estimadores = classificadores

- Rótulos quantitativos: problema de regressão.

Estimadores = regressores





# Exemplos de Modelos



## » Exploratórios

- Regras de Associação: descobrir associações nos dados
- Agrupamento (*Clustering*): encontrar agrupamentos naturais nos dados
- Outros (Análise Fatorial, etc.)

## » Preditivos

- Procura
  - Árvores de Decisão
  - Árvores de Regressão
- Distâncias
  - *k - Nearest Neighbour (k-NN)*
- Probabilísticos
  - Redes *Bayesianas*
- Otimização
  - Redes Neuronais
  - *Support Vector Machine (SVM)*

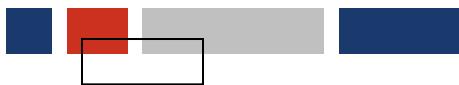


# Avaliação de Modelos Preditivos



## » Enquadramento

- Em diferentes problemas reais, diversos algoritmos de DM podem ser candidatos à resolução do mesmo.
- Contudo, apenas um único algoritmo será escolhido.
- A realização de experiências é necessária para demonstrar a sua eficiência.
- A avaliação do algoritmo considera várias medidas relacionadas com o desempenho obtido nas previsões efetuadas.

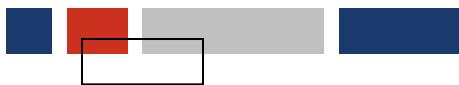


# Métodos de Avaliação



## » Aferir a qualidade do modelo – Modelos Preditivos

- Que confiança podemos ter no modelo de decisão obtido a partir do registo histórico de decisões?
- Podemos tentar responder a esta questão usando o modelo para tomar decisões e verificar que percentagem destas é acertada.
- Para isso vamos proceder do seguinte modo:
  - A nossa amostra de decisões (para as quais sabemos a decisão correta) é dividida em duas partes:
    - ✓ Uma para construir o modelo de decisão (**conjunto de treino**).
    - ✓ Outra para o testar o modelo, comparando as previsões do modelo com as decisões dos peritos (**conjunto de teste**).



# Métodos de Avaliação



## » Aferir a qualidade do modelo

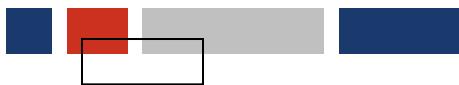
São utilizados processos que combinam amostragem e métricas:

- **Amostragem**

- ✓ Holdout
- ✓ Validação cruzada
- ✓ Bootstrap

- **Métricas**

- ✓ Métricas para classificação
- ✓ Métricas para regressão



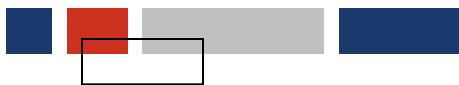
# Métodos de Avaliação



## » Amostragem

Seleção do conjunto de treino e do conjunto de teste através de um dos seguintes métodos:

- ✓ Holdout
- ✓ Amostragem Aleatória
- ✓ Validação cruzada
- ✓ Bootstrap

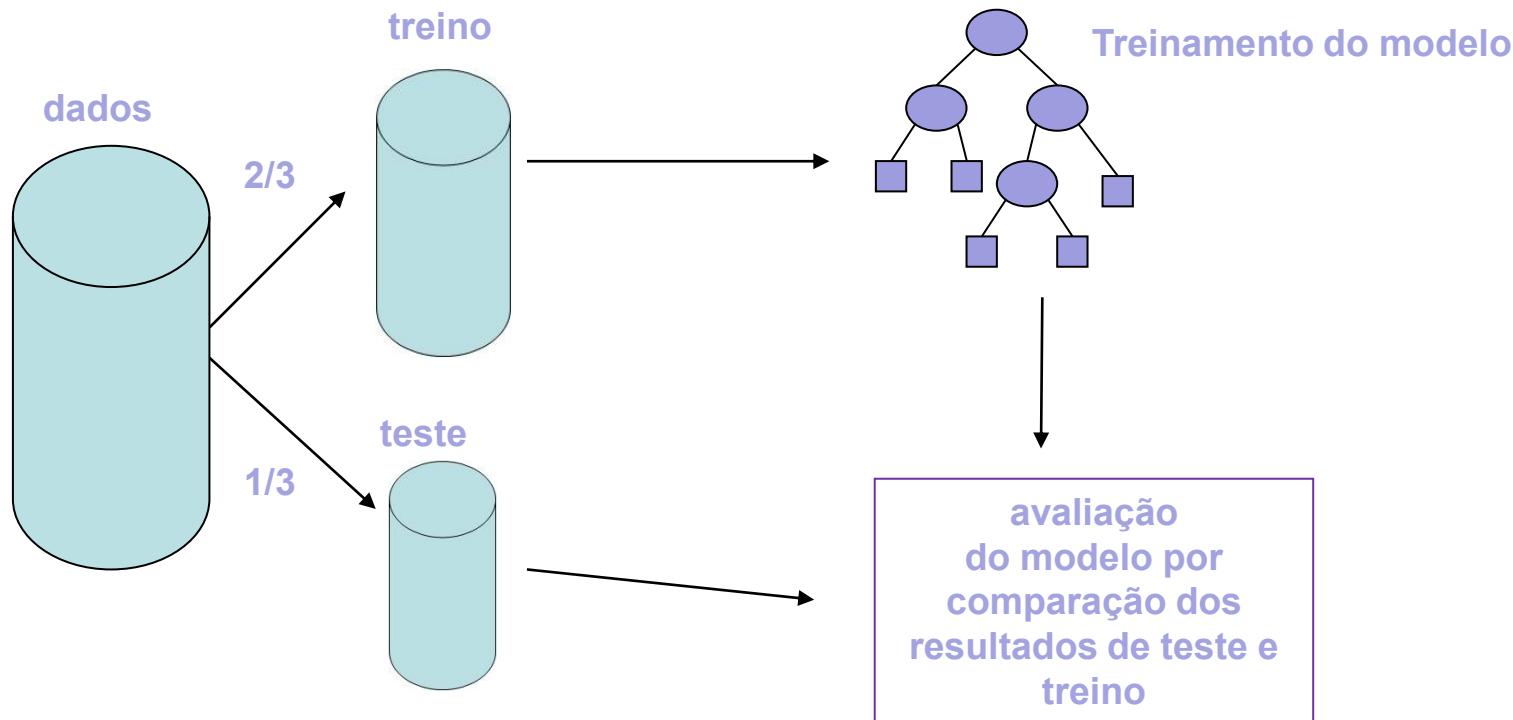


# Métodos de Avaliação



## » Holdout

(adequado e suficiente para amostras muito grandes)

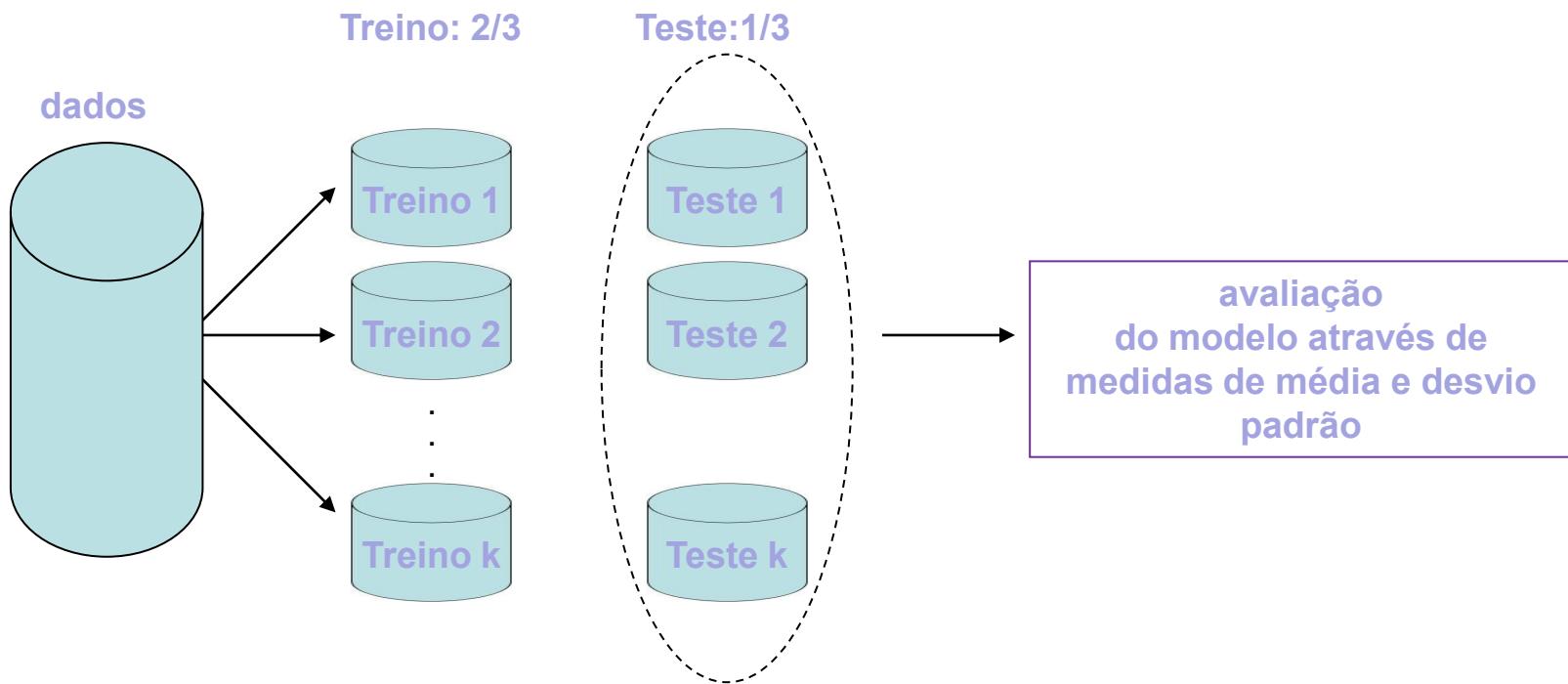




# Métodos de Avaliação



## » Amostragem Aleatória

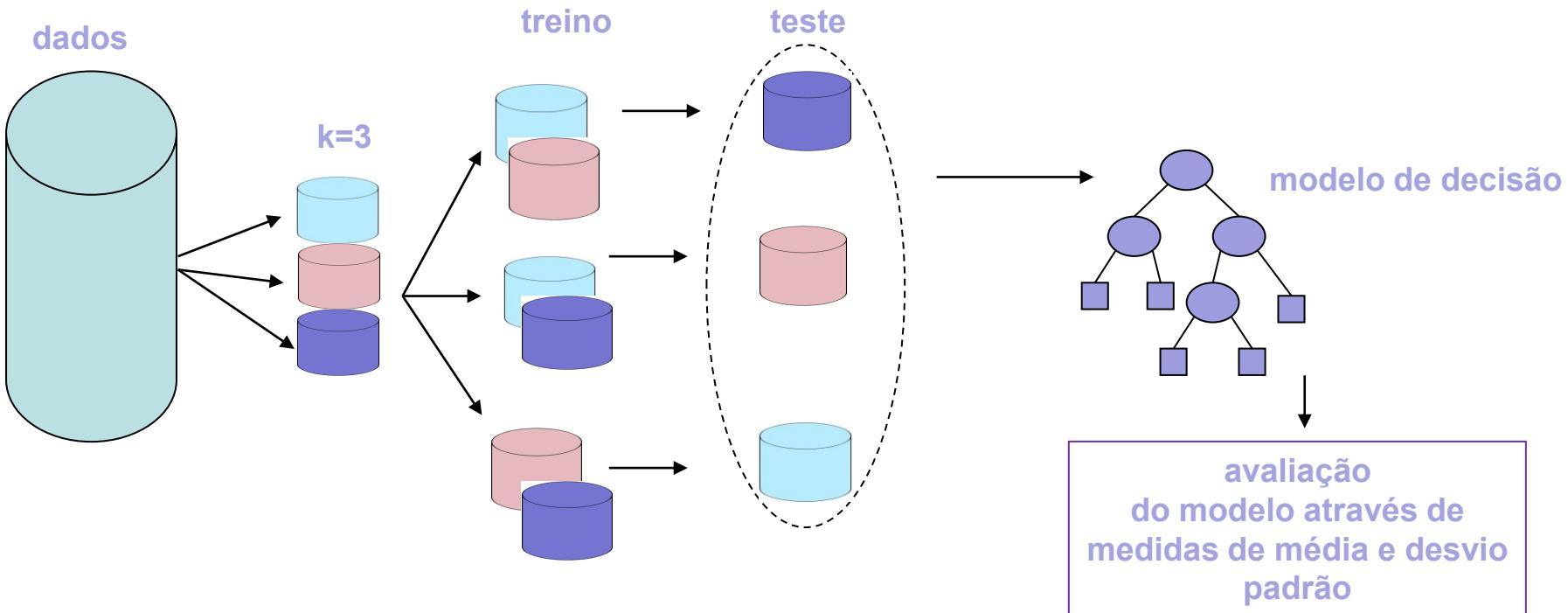


# Métodos de Avaliação



## » Validação Cruzada

(o mais usado, geralmente com  $k=10$ )

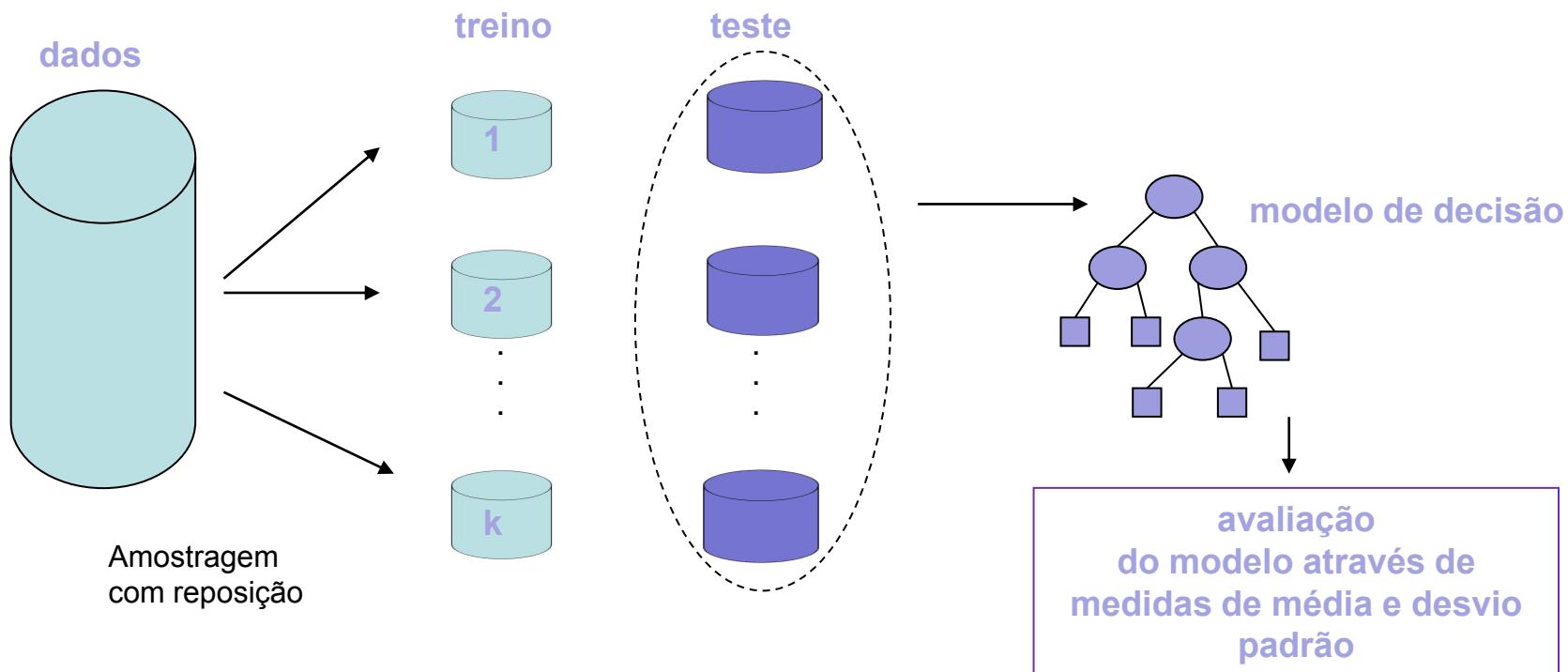


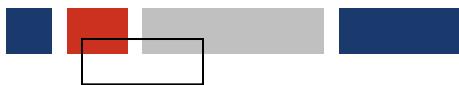
# Métodos de Avaliação



## » *Bootstrap*

(mais adequado para amostras pequenas)





# Métodos de Avaliação



## » Métricas

A avaliação do desempenho de um algoritmo supervisionado é realizada através de diferentes medidas disponíveis para os diferentes tipos de problemas:

- ✓ Classificação
- ✓ Regressão



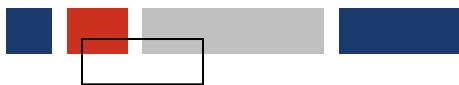
# Métodos de Avaliação



## » Avaliação de algoritmos de classificação

A avaliação do desempenho de um classificador baseia-se nas contagens de exemplos de teste corretamente e incorretamente previstos pelo classificador (Tan *et al.*, 2006).

A matriz de confusão facilita a visualização do número de classificações corretas e do número de classificações preditas para cada classe, de um determinado conjunto de exemplos, segundo o classificador em análise.



# Métodos de Avaliação



## » Avaliação de algoritmos de classificação

Num problema de classificação em que o conjunto de dados tem apenas duas categorias, é muitas vezes considerada uma como ‘positiva’ e a outra como ‘negativa’ (Bramer, 2007).

As entradas da matriz de confusão são referidas como:

- TP - verdadeiros positivos (*true positives*),
- FP - falsos positivos (*false positives*),
- FN - falsos negativos (*false negatives*),
- TN - verdadeiros negativos (*true negatives*)



# Métodos de Avaliação



## » Avaliação de algoritmos de classificação

A tabela seguinte apresenta a matriz de confusão para um problema de classificação binária.

Categoria	Prevista $C^+$	Prevista $C^-$
Verdadeira $C^+$	TP	FN
Verdadeira $C^-$	FP	TN

- TP – nº de exemplos da categoria ‘positiva’ corretamente previstos
- FP – nº de exemplos da categoria ‘positiva’ incorretamente previstos
- FN – nº de exemplos da categoria ‘negativa’ incorretamente previstos
- TN – nº de exemplos da categoria ‘negativa’ corretamente previstos



# Métodos de Avaliação

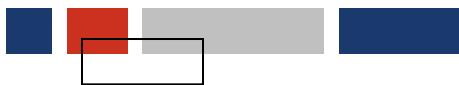


## » Métrica para classificação

- **Taxa de Erro**

$$err(\hat{f}) = \frac{1}{n} \sum_{i=1}^n I(y_i \neq \hat{f}(x_i)) \quad \text{ou} \quad err(\hat{f}) = \frac{FP_c + FN_c}{n}$$

Dado um conjunto de dados com  $n$  objetos, sobre o qual a avaliação será realizada, essa taxa corresponde à proporção de exemplos desse conjunto classificados incorretamente, obtidos por comparação entre a classe conhecida de  $x_i$ , ou seja  $y_i$ , com a classe prevista  $\hat{f}(x_i)$ .



# Métodos de Avaliação



## » Métrica para classificação

- **Precisão (*precision*):**

A precisão para a categoria  $c$  mede a percentagem de atribuições corretas entre todos os exemplos atribuídos a  $c$ .

$$\text{Precisão}_c = \frac{TP_c}{TP_c + FP_c}$$



# Métodos de Avaliação



## » Métrica para classificação

- **Sensibilidade (*recall*):**

A medida *recall* para a categoria c indica a percentagem de atribuições corretas em c entre todos os exemplos que deviam ser atribuídos a c.

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c}$$



# Métodos de Avaliação



## » Métrica para classificação

Faceli *et al.* (2011) consideram a precisão uma medida de exatidão do classificador e a *recall* uma medida da sua completude.

A análise destas duas medidas em separado normalmente não é discutida. No entanto, a média harmónica das duas medidas dá origem à medida-F1.

- **Medida-F1:**

$$F_{1c} = \frac{2 * \text{Precisão}_c * \text{Recall}_c}{\text{Precisão}_c + \text{Recall}_c}$$



# Métodos de Avaliação



## » Métricas de erro para regressão

- **Mean Square Error**

$$MSE(\hat{f}) = \frac{1}{n} \sum_{i=1}^n I(y_i - \hat{f}(x_i))^2$$

- **Mean Absolute Error**

$$MAE(\hat{f}) = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{f}(x_i)|$$

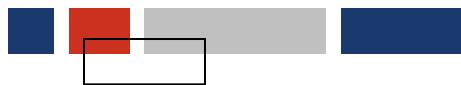
$n$ : corresponde ao número de objetos sobre o qual a avaliação será realizada

$x_i$  : objeto  $i$

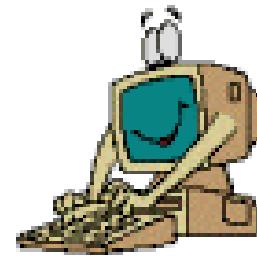
$y_i$  : classe conhecida de  $x_i$

$\hat{f}(x_i)$  : classe prevista para  $x_i$

➤ *Valores mais baixos correspondem a melhor capacidade preditiva.*



# Exercício



## » Exercício 3.1.



# Modelos Exploratórios e Modelos Preditivos



- *Clustering*
- Regras de Associação
- *k - Nearest Neighbour (k-NN)*
- Redes *Bayesianas*
- Redes Neuronais
- *Support Vector Machine (SVM)*
- Árvores de Decisão
- Árvores de Regressão



# Clustering



## » *Clustering, Análise de Clusters ou Agrupamentos*

- A Análise de Clusters é um conjunto de técnicas estatísticas que são aplicadas com o objetivo de encontrar **agrupamentos naturais ou segmentos** nas observações de um conjunto de dados.
- A Análise de Clusters também é conhecida nos domínios da Estatística e Análise de Dados por Análise Classificatória.



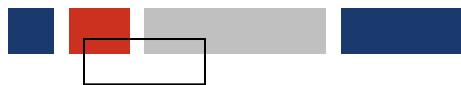
# Clustering



## » *Clustering, Análise de Clusters ou Agrupamentos*

Utilizada em vários domínios, como por exemplo:

- **No Marketing** devido à necessidade de segmentação que consiste em dividir o mercado em diferentes subconjuntos (segmentos, ou nichos) que se comportam de forma da mesma forma ou que apresentam necessidades semelhantes.
- **Clima** : identificação de padrões de incêndios em municípios
- **Medicina**: analisar os diferentes tipos de depressão e classificá-los de acordo com os sintomas
- **Demografia**: identificar grupos de municípios de acordo com o perfil de envelhecimento



# Clustering

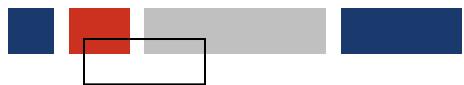


## » Após formação dos grupos

- Os grupos devem ser homogéneos, ou seja, os objetos dentro de cada grupo devem ser semelhantes.
- Os objetos pertencentes a grupos distintos devem ser diferentes.

Segmentos=grupos=classes



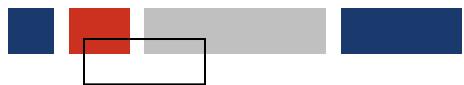


# Clustering



## » Metodologias de Análise de Clusters

- **Hierárquica** – é feito um agrupamento ascendente dos  $n$  indivíduos ou variáveis, começando por  $n$  clusters até ao número final de grupos.
- **Não hierárquica** – é feito um agrupamento com base num número definido de *clusters* e os novos casos são classificados de acordo com a proximidade dos centros dos grupos já existentes.
- **Fuzzy** – ao contrário dos casos anteriores, em que não pode haver sobreposição dos grupos, no *fuzzy clustering* essa sobreposição pode existir.



# Clustering



## » Metodologias de Análise de Clusters

### Outra classificação

- **Métodos baseados em distâncias:**

- Métodos de Partição / Métodos Hierárquicos

- **Outros métodos:**

- CobWeb: algoritmo de clustering incremental em que se trabalha com os dados de forma incremental da raiz para as folhas (Witten and Frank, p. 212)

- EM (expectation – maximization): algoritmo que se baseia na distribuição dos dados (Witten and Frank, p. 221)

- *Self Organization Maps (SOM)*: tipo de rede neural artificial (RNA) que é treinada para aprender em dimensões baixas (tipicamente bidimensional). Ao contrário das RNA, os SOM são diferentes no sentido de que usam uma função de vizinhança para preservar as propriedades topológicas do espaço de entrada.

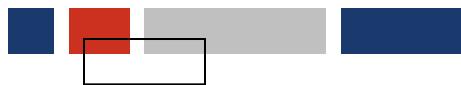


# Clustering



## » Análise Hierárquica

- Na análise hierárquica é feito um agrupamento ascendente dos  $n$  indivíduos ou variáveis, começando por  $n$  clusters até ao número final de grupos.
- Baseia-se na construção de uma matriz de semelhança ou diferenças, em que cada elemento da matriz descreve o grau de semelhança ou diferença entre cada dois casos, com base nas variáveis escolhidas.



# Clustering



## » Análise Hierárquica

Dado um conjunto de dados, uma hierarquia é um conjunto de subconjuntos (*clusters*) tais que :

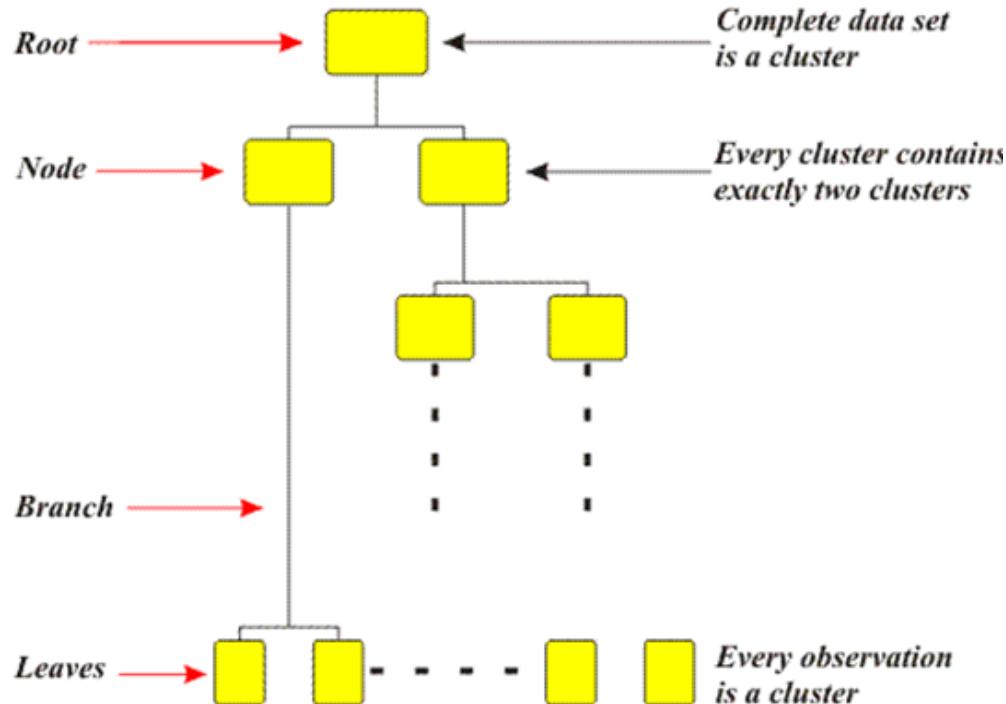
- O conjunto completo é um *cluster*;
- Cada observação isolada é um *cluster* (*singleton*);
- Dados dois *clusters* na hierarquia:
  - ou não têm observações em comum
  - ou então um *cluster* está incluído no outro (não existe sobreposição)
- É comum por razões práticas, que a agregação se faça aos pares.
- A representação associada designa-se por **dendrograma**.



# Clustering

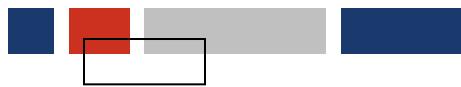


## » Análise Hierárquica (Dendrograma)



### Hierarchy

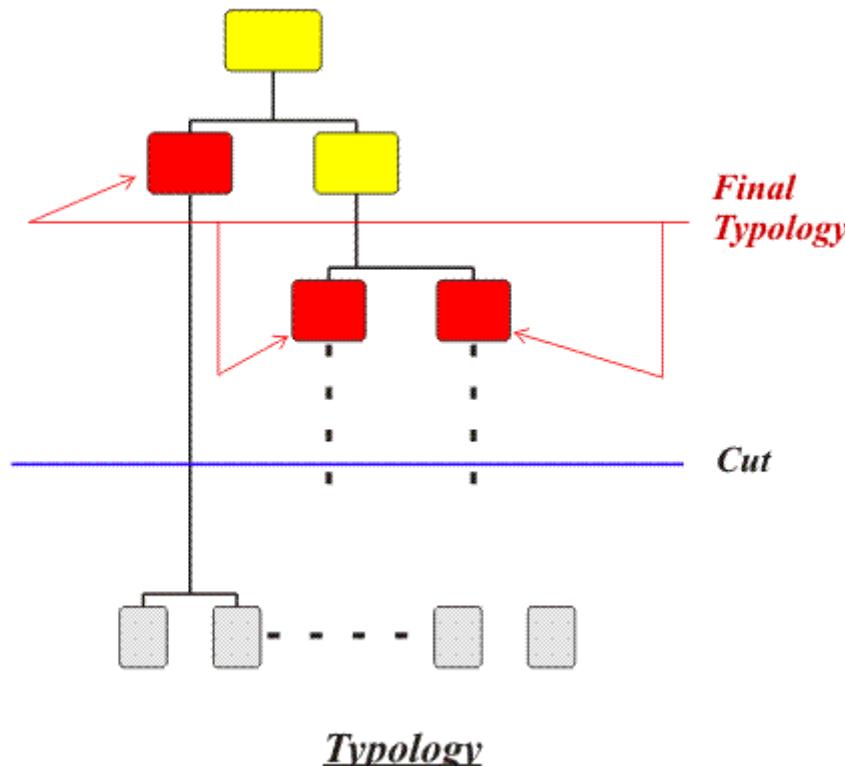
In In [http://www.aiaccess.net/English/Glossaries/GlosMod/e\\_gm\\_k\\_means.htm](http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_k_means.htm)



# Clustering

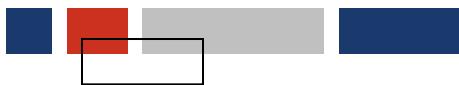


## » Análise Hierárquica (Dendrograma)



Typology

In In [http://www.aiaccess.net/English/Glossaries/GlosMod/e\\_gm\\_k\\_means.htm](http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_k_means.htm)

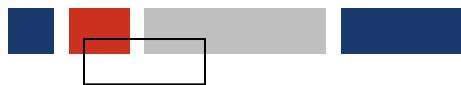


# Clustering



## » Análise Hierárquica

- No processo de formação de grupos através e análise de clusters hierárquica é importante definir:
  - uma medida de semelhança para agrupar indivíduos semelhantes
  - um critério de agregação de indivíduos



# Clustering



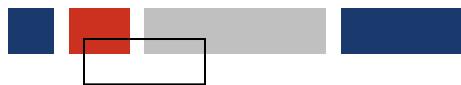
## » Análise Hierárquica

Para realizar uma Análise de *Clusters* é necessário um conjunto de  $n$  indivíduos e  $p$  variáveis.

O método agrupa os indivíduos semelhantes, decorrendo em quatro etapas:

1. Seleccionar os indivíduos e as variáveis;
2. Definir a medida de semelhança ou distância entre indivíduos;
3. Definir o critério de agregação;
4. Validar os resultados.





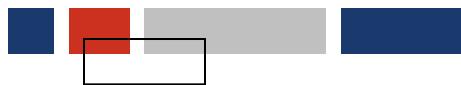
# Clustering



## » Análise Hierárquica

1. Seleccionar os indivíduos e as variáveis;

- O investigador deverá ter em conta o seu conhecimento da informação para fazer a selecção adequada da base de dados (indivíduos e variáveis);
- As variáveis têm de ser do mesmo tipo, para se poder usar a mesma medida de semelhança entre indivíduos.



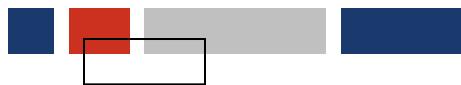
# Clustering



## » Análise Hierárquica

2. Definir a medida de semelhança ou distância entre indivíduos;

- Distância euclidiana, coseno, correlação de Pearson, etc.  
(variáveis quantitativas)
- Quiquadrado, Phiquadrado, etc.(variáveis qualitativas)



# Clustering

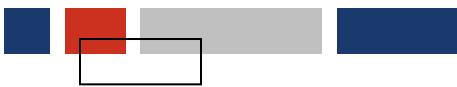


## » Análise Hierárquica

3. Definir o critério de agregação (**cluster method**);

Exemplos de métodos de agregação:

- **Vizinho mais próximo (Nearest Neighbour ou single linkage):** Em cada iteração é escolhido o emparelhamento entre os indivíduos com menor distância. Depois é escolhida a menor distância entre os indivíduos e os novos clusters.
- **Vizinho mais afastado (Furthest Neighbour ou complete linkage):** Em cada iteração é escolhido o emparelhamento entre os indivíduos com menor distância. Depois é escolhida a maior distância entre os indivíduos e os novos clusters.
- **Ward:** baseia-se na perda de informação resultante do agrupamento dos indivíduos, medida através da soma dos quadrados dos desvios das observações individuais relativamente às médias dos grupos em que são classificadas.



# Clustering

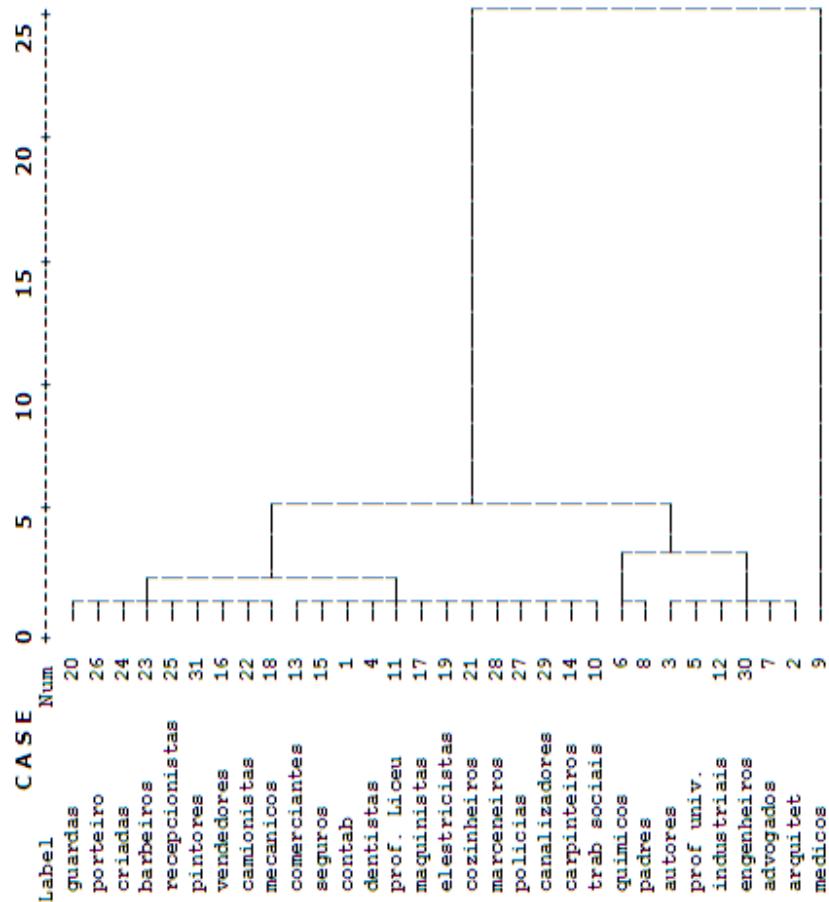


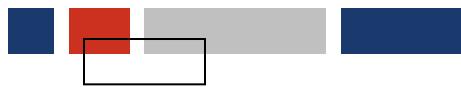
## » Análise Hierárquica

4. Validar os resultados.

Dendrograma.

Qual o número ideal de classes?





# Clustering



## » Análise Hierárquica

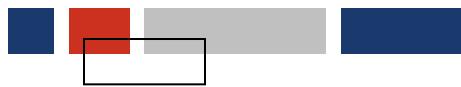
Exemplo ilustrativo:

### 1. Matriz de dados

	v1	v2
a	1	3
b	5	8
c	1	2
d	6	5

### 2. Matriz de distâncias entre 4 objetos: a,b,c,d

	a	b	c	d
a	0	5	3	9
b	5	0	7	4
c	3	7	0	6
d	9	4	6	0



# Clustering



## » Análise Hierárquica

Exemplo ilustrativo:

3. Definir o critério de agregação: Nearest Neighbour (Single Linkage)

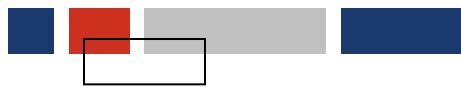
				Min d= d(a,c)=d(c,a)=3				Min [d(a,b), d(c,b))]= d(a,b)=5				Min d= d(b,d)=d(d,b)=4					
				a	b	c	d	a	b	c	d	a+c	b	d	a+c	b	d
a	0	5	3	9	a	0	5	3	9	a+c	0	5	6	a+c	0	5	6
b	5	0	7	4	b	5	0	7	4	b	5	0	4	b	5	0	4
c	3	7	0	6	c	3	7	0	6	d	6	4	0	d	6	4	0
d	9	4	6	0	d	9	4	6	0								

Matriz de distâncias  
entre 4 objetos: a,b,c,d

Min [d(a+c,b),  
d(a+c,d))]= d(a+c,b)=5

	a+c	b+d
a+c	0	5
b+d	5	0





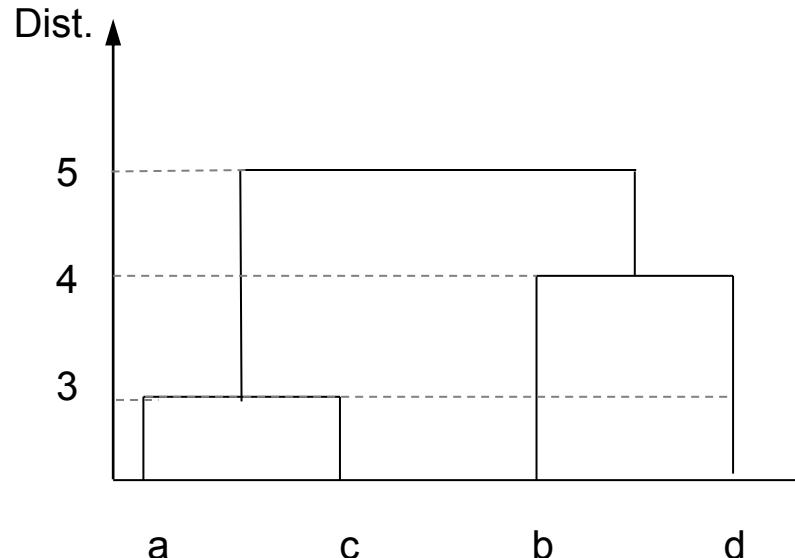
# Clustering

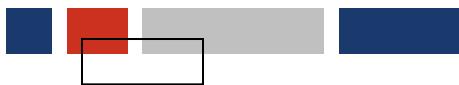


## » Análise Hierárquica

Exemplo ilustrativo :

### 4. Dendrograma





# Clustering



## » Análise Hierárquica

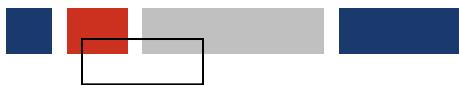
Exemplo ilustrativo:

3. Definir outro critério  
de agregação: ward

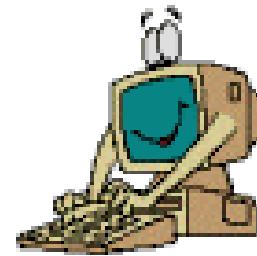
Passos:

- Cálculo das médias das variáveis para cada grupo;
- Cálculo do quadrado da distância Euclidiana entre essas médias e os valores das variáveis para cada indivíduo
- $$D_{ij}^2 = \sum_{k=1}^p \sum_{i=1}^{n_i} (x_{ik} - \bar{x}_k)^2$$
- Soma das distâncias para todos os indivíduos (soma dos quadrados das diferenças ou erros) - Minimiza a variância dentro dos grupos minimizando a soma dos quadrados dos erros (ESS).

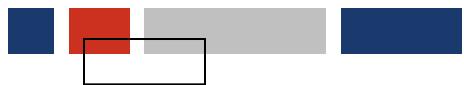
	Membros no cluster			
	1	2	3	ESS
1	a, b	c	d	20,5
2	a, c	b	d	0,5
3	a,d	b	c	14,5
4	b,c	a	d	26,0
5	b,d	a	c	5,0
6	c,d	a	b	17,0
1	a,b,c	d		31,3
2	a,b,d	c		26,7
3	a,c,d	b		21,3
4	b,c,d	a		32,0
1	a,b	c,d		37,5
2	a,c	d,b		5,5
3	a,d	b,c		40,5



# Exercício



» Exercício 4.1. e 4.2.

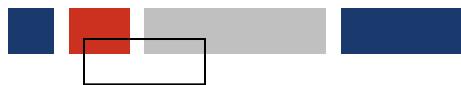


# Clustering



## » Análise Hierárquica

1. Dados: incêndios
2. Definir a medida de semelhança ou distância entre indivíduos:
  - Quadrado da distância euclidiana
3. Definir o critério de agregação;
  - Nearest Neighbour, Furthest Neighbour, Ward
4. Validar os resultados.



# Clustering



## » Análise Hierárquica

Exemplo:

1. Dados: fonte: ICNF - Instituto da Conservação da Natureza e das Florestas)

<http://www.icnf.pt/portal>

Variáveis :

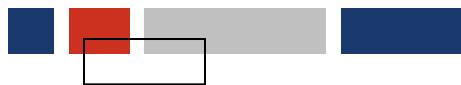
V1: Dimensão média;

V2: Ocorrências/área;

V3: % área ardida;

V4: povoamentos /área ardida





# Clustering



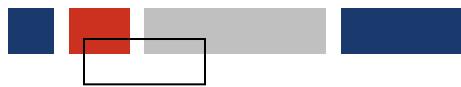
## » Análise Hierárquica

Exemplo:

2. Definir a medida de semelhança ou distância entre indivíduos:

- Quadrado da distância euclidiana entre dois pontos i,j

$$D_{ij}^2 = \sum_{k=1}^p (x_{ik} - x_{kj})^2$$



# Clustering

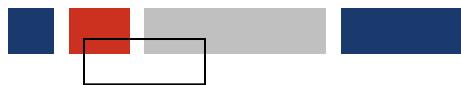


## » Análise Hierárquica

Exemplo:

3. Definir o critério de agregação;

- Nearest Neighbour (Single Linkage)
- Furthest Neighbour (Complete Linkage)
- Critério de Ward



# Clustering



## » Análise Hierárquica

Exemplo:

### 4. Validar os resultados.

- Verificar qual o número ideal de clusters
- Comparar vários critérios de agregação / estabilidade da solução?
- Fazer análise descritiva das classes encontradas. Os grupos são diferentes ? (recorrer a testes paramétricos ou não paramétricos para confirmar estas diferenças)





# Clustering

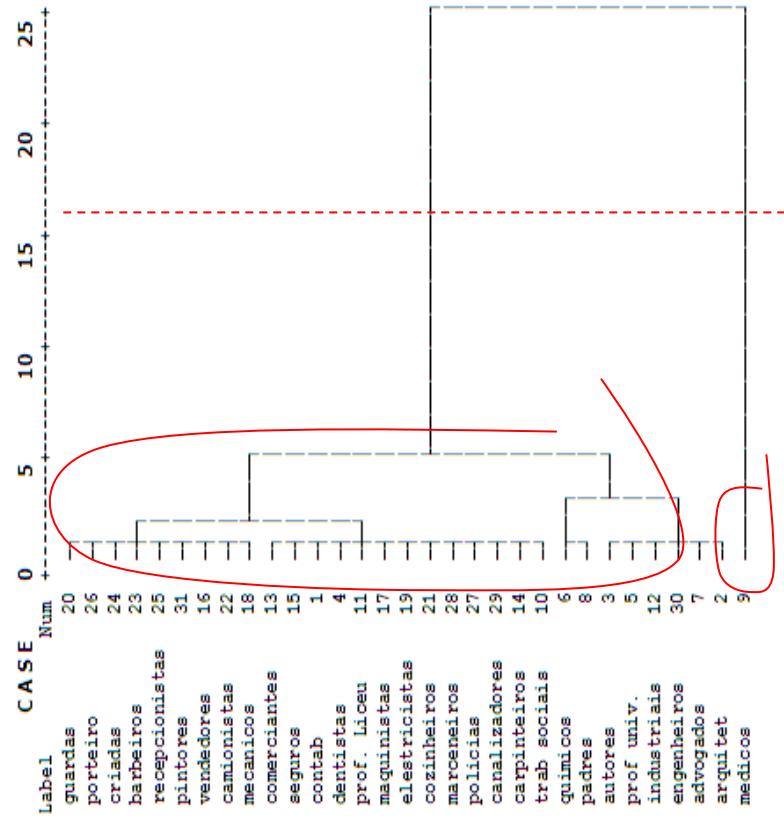


## » Análise Hierárquica

Exemplo:

### 4. Validar os resultados.

- Verificar qual o número ideal de clusters
  - Baseado no dendrograma
  - Embora não exista uma regra universal, o corte deve-se fazer no local onde existe o maior salto de distância





# Clustering

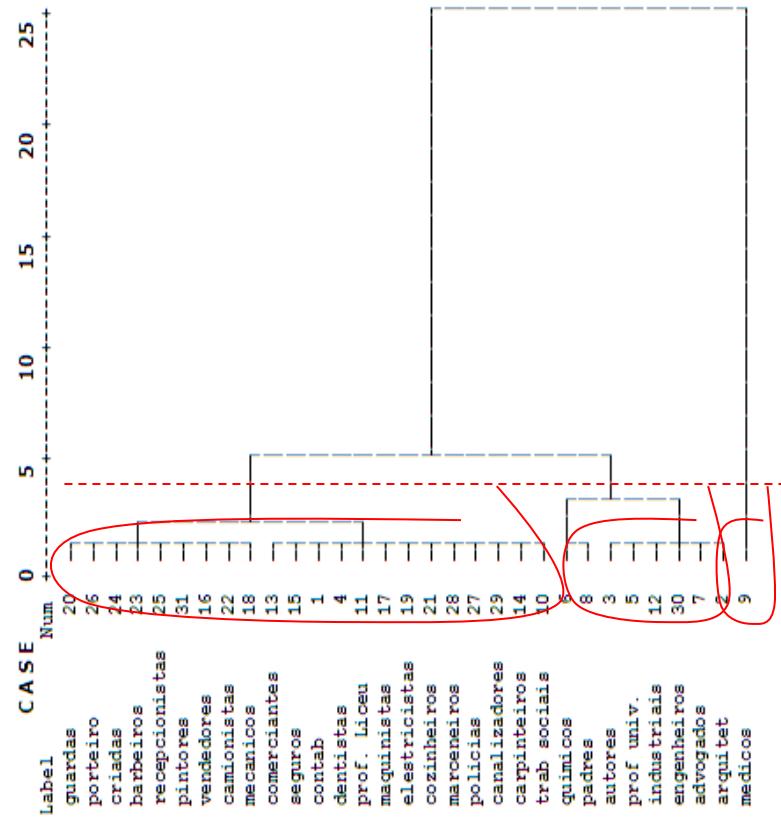


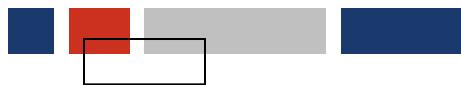
## » Análise Hierárquica

Exemplo:

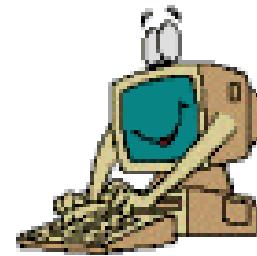
### 4. Validar os resultados.

- Verificar qual o número ideal de clusters
  - Baseado no dendrograma
  - Embora não exista uma regra universal, o corte deve-se fazer no local onde existe o maior salto de distância
  - O número de clusters em causa é também determinante para a solução

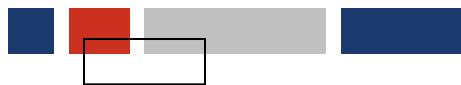




# Exercício



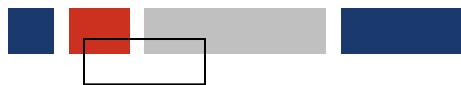
» Exercício 4.3. e 4.4.



## » Análise Não Hierárquica

### ■ KMeans

- É um método que visa a partição de n observações em k clusters, em que cada observação pertence ao cluster com o centro mais próximo da média.
- O objetivo do k-Means é minimizar a soma do erro quadrático dos vários grupos gerados.
- Pelo seu funcionamento o k-Means apenas possibilita a convergência para um mínimo local designado de centróide, dado um determinado número de grupos.
- É uma técnica de clustering baseada na minimização de um critério.
- Baseia-se numa escolha prévia do número de pontos inicial (centróides)



# Clustering

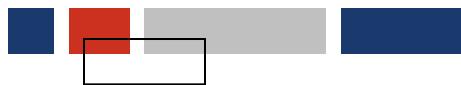


## » Análise Não Hierárquica

### ■ KMeans

#### Algoritmo:

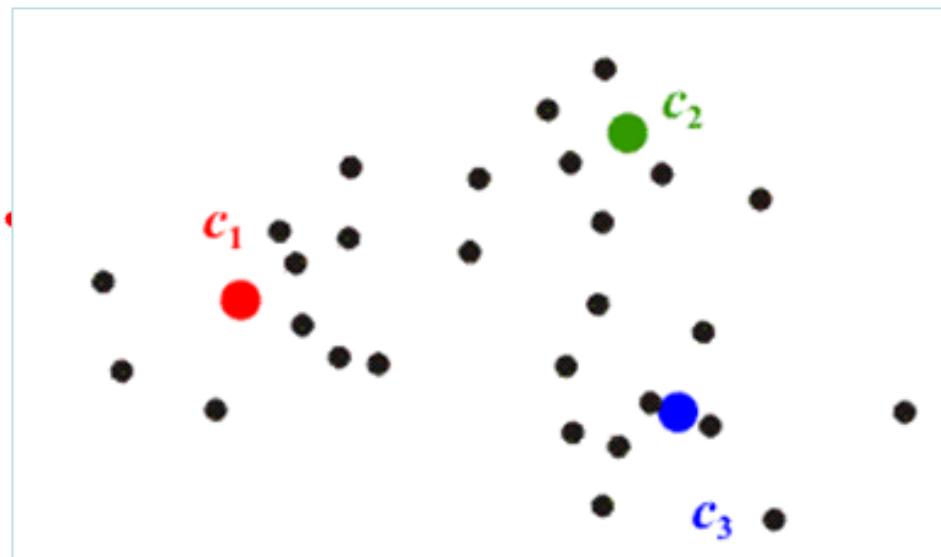
1. Selecionar K pontos como sendo os centróides iniciais
2. Repetir até se atingir o mínimo do critério estabelecido:
  - 2.1. Formar K clusters, associando cada novo ponto (registo de uma base de dados) ao centróide mais próximo
  - 2.1. Recalcular o centróide de cada cluster



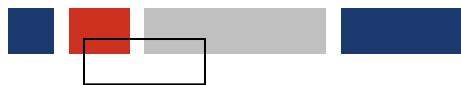
# Clustering



## » Análise Não Hierárquica (ilustração do Kmeans)



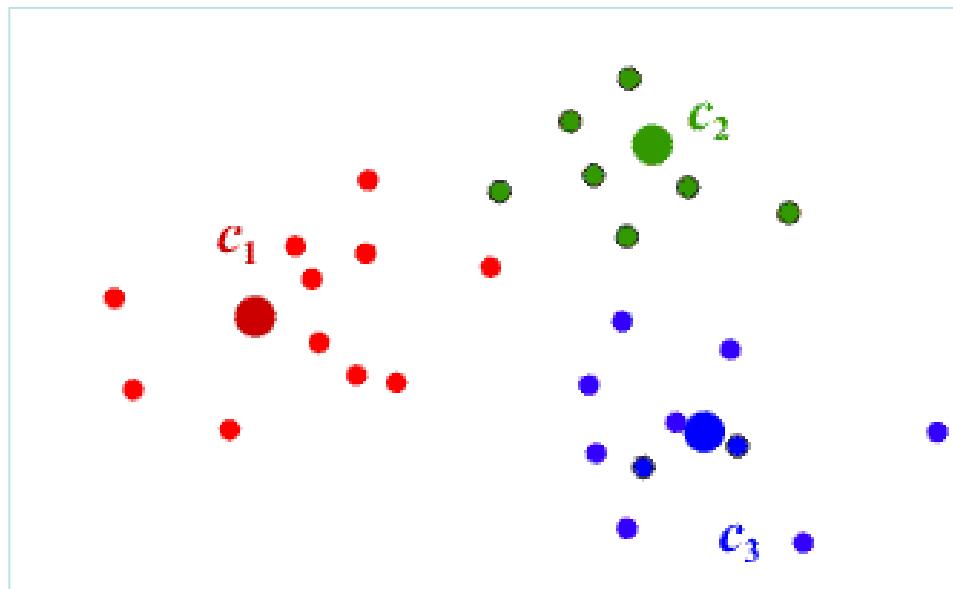
In In [http://www.aiaccess.net/English/Glossaries/GlosMod/e\\_gm\\_k\\_means.htm](http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_k_means.htm)



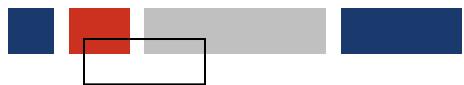
# Clustering



## » Análise Não Hierárquica (ilustração do Kmeans)



In In [http://www.aiaccess.net/English/Glossaries/GlosMod/e\\_gm\\_k\\_means.htm](http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_k_means.htm)



# Clustering



## » Análise Não Hierárquica

**Kmeans:** vantagens e desvantagens face à análise hierárquica

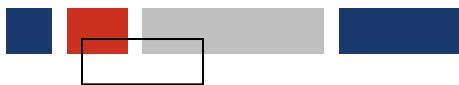
### Vantagens

- Trata-se de um método conceitualmente mais simples e mais rápido.

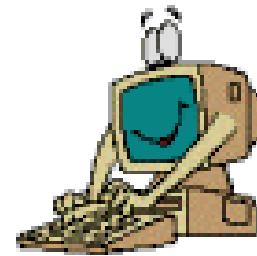
### Desvantagens:

- Utilizador tem de escolher o valor  $K$  (número de clusters) (enquanto no caso da análise hierárquica, essa escolha pode ser “natural”).
- Dado um conjunto  $K$  de clusters, estes grupos vão depender muito da configuração inicial de centróides o que leva a uma interpretação dos clusters mais incerta.

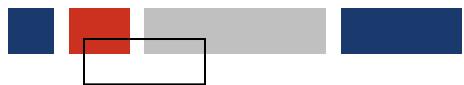




# Exercício



» Exercício 4.5. e 4.6.



# Regras de Associação



## » Regras de Associação

- Modelo exploratório.
- Descoberta de associações que ocorrem significativamente numa BD.
- As regras são extraídas a partir de bases de dados que contêm transações formadas por conjuntos de itens do domínio da aplicação.
- A tarefa ‘associação’ pretende encontrar relacionamentos ou padrões frequentes entre conjuntos de dados.



# Regras de Associação



## » Regras de Associação

Considere-se:

- Conjunto de itens:  $I = \{i_1, i_2, \dots, i_n\}$
- Conjunto de transações:  $T = \{t_1, t_2, \dots, t_n\}$

Uma regra de associação é uma implicação da forma  $A \Rightarrow B$ ,  
onde  $A \subset I$ ,  $B \subset I$  e  $A \cap B = \emptyset$ .



# Regras de Associação



## » Suporte de uma regra

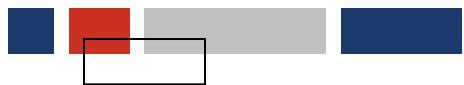
- O **suporte** da regra  $A \Rightarrow B$  é a probabilidade do conjunto de itens  $\{A, B\}$ .

$$\text{suporte}(A \Rightarrow B) = p(\{A, B\})$$

## » Confiança de uma regra

- A **confiança** da regra  $A \Rightarrow B$  é a probabilidade condicional de B dado A.

$$\text{confiança}(A \Rightarrow B) = p(B | A) = \frac{\text{suporte}(\{A, B\})}{\text{suporte}(\{A\})}$$



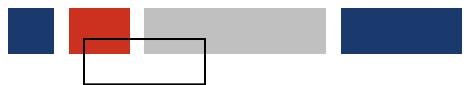
# Regras de Associação



## » *Lift* de uma regra

- Um valor de ***lift igual a 1***, indica que A e B são independentes.
- Um valor de ***lift igual < 1***, indica que A e B são negativamente correlacionados.
- Um valor de ***lift igual > 1***, indica que A e B são positivamente correlacionados.

$$Lift(A \Rightarrow B) = \frac{\text{confiança}(A \Rightarrow B)}{\text{suporte}(\{B\})}$$



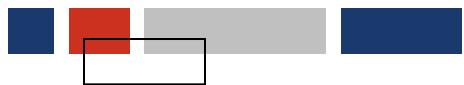
# Regras de Associação



## » Algoritmo *Apriori*

O algoritmo percorre a BD em profundidade e gera *itemsets* candidatos de  $k$  elementos a partir de conjuntos de itens de  $k-1$  elementos.

A partir dos  $k$ -*itemsets* candidatos é percorrida a BD para determinar se os mesmos são frequentes, identificando assim todos os  $k$ -*itemsets* frequentes.



# Regras de Associação



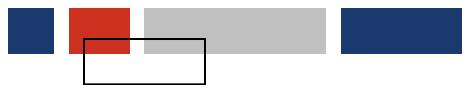
## » Algoritmo *Apriori*

### O princípio “Apriori”:

- Qualquer subconjunto de itens frequentes deve ser frequente;
- Qualquer sobreconjunto de conjuntos de itens frequentes deve ser frequente.

### Extrair conjuntos de itens frequentes:

- Encontrar os conjuntos de itens frequentes que tenham pelo menos o suporte mínimo dado;
- Encontrar iterativamente os conjuntos de itens;
- Usar os itens frequentes para gerar regras de associação.



# Regras de Associação



## » Exemplo:

Encontrar os conjuntos de itens frequentes

Encontrar as regras com confiança > 80%

Conjunto de itens:  $I = \{i_1, i_2, i_3, i_4, i_5\}$

Conjunto de transações:  $T = \{t_1, t_2, t_3, t_4\}$

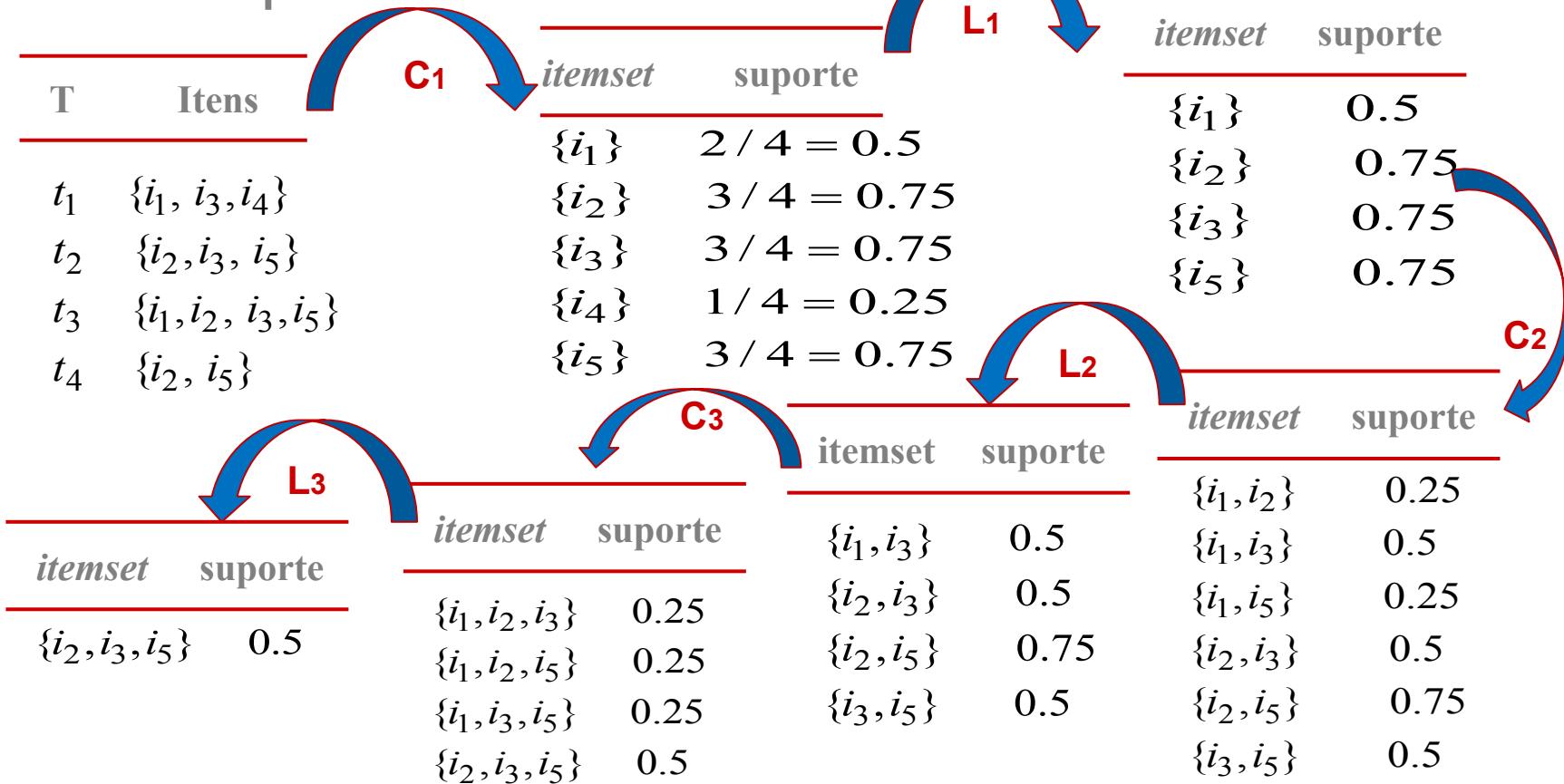
Suporte mínimo = 50%

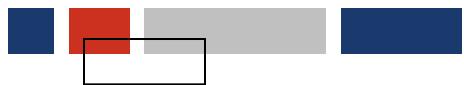
$C_k$ : *itemsets* candidatos de  $k$  elementos

$L_k$ : *itemsets* frequentes de  $k$  elementos

# Regras de Associação

## » Exemplo



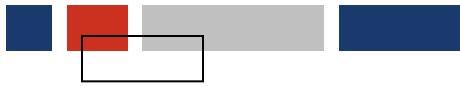


# Regras de Associação



## » Conjuntos de itens frequentes

0 itens	1 item	2 itens	3 itens
$\{\}$	$\{i_1\}$	$\{i_1, i_3\}$	$\{i_2, i_3, i_5\}$
	$\{i_2\}$	$\{i_2, i_3\}$	
	$\{i_3\}$	$\{i_2, i_5\}$	
	$\{i_5\}$	$\{i_3, i_5\}$	

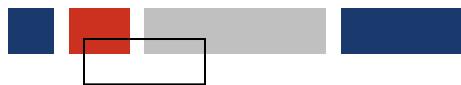


# Regras de Associação

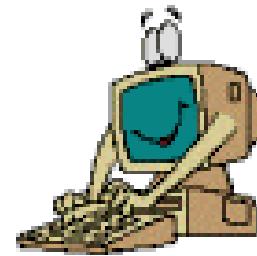


» Encontrar as regras com confiança > 80%

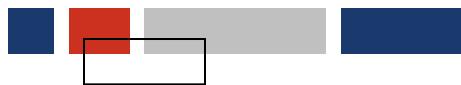
Regra	Suporte	Confiança
$\{i_2\} \rightarrow \{i_3\}$	0.5	$0.5 / 0.75 = 66.67\%$
$\{i_3\} \rightarrow \{i_2\}$	0.5	$0.5 / 0.75 = 66.67\%$
→ $\{i_2\} \rightarrow \{i_5\}$	0.75	$0.75 / 0.75 = 100\%$
→ $\{i_5\} \rightarrow \{i_2\}$	0.75	$0.75 / 0.75 = 100\%$
$\{i_3\} \rightarrow \{i_5\}$	0.5	$0.5 / 0.75 = 66.67\%$
$\{i_5\} \rightarrow \{i_3\}$	0.5	$0.5 / 0.75 = 66.67\%$
→ $\{i_2, i_3\} \rightarrow \{i_5\}$	0.5	$0.5 / 0.5 = 100\%$
$\{i_5\} \rightarrow \{i_2, i_3\}$	0.5	$0.5 / 0.75 = 66.67\%$
$\{i_2, i_5\} \rightarrow \{i_3\}$	0.5	$0.5 / 0.75 = 66.67\%$
$\{i_3\} \rightarrow \{i_2, i_5\}$	0.5	$0.5 / 0.75 = 66.67\%$
→ $\{i_3, i_5\} \rightarrow \{i_2\}$	0.5	$0.5 / 0.5 = 100\%$
$\{i_2\} \rightarrow \{i_3, i_5\}$	0.5	$0.5 / 0.75 = 66.67\%$



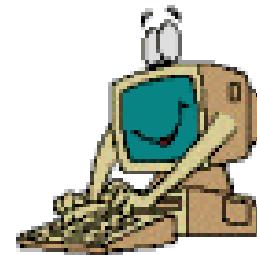
# Exercício



## » Exercício 4.7.



# Exercício

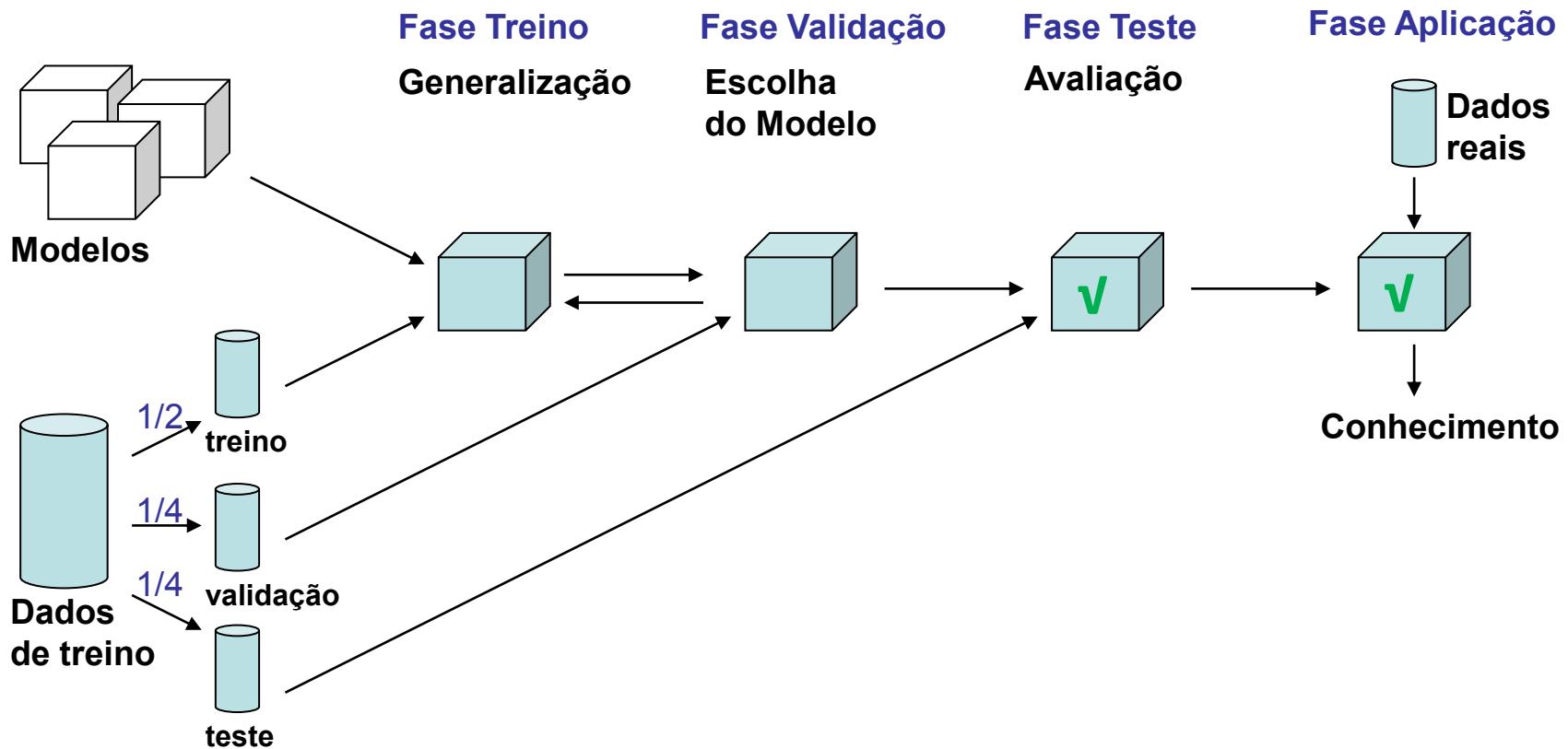


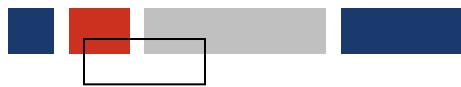
## » Exercício 4.8.

# Modelos preditivos



## » Visão Geral





# Modelos preditivos



## » Conceitos

- atributo-resposta (i.e. atributo-alvo) e atributos-preditivos;
- problemas no conjunto de treino (e.g. ruído, dados em falta, ...);
- *curse of dimensionality*;
- generalização (*lazy learners* vs. *eager learners*);
- precisão e exactidão;
- sub-ajustamento vs. sobre-ajustamento;
- critérios de paragem de um algoritmo;
- máximos locais e máximo global.





# Modelos preditivos



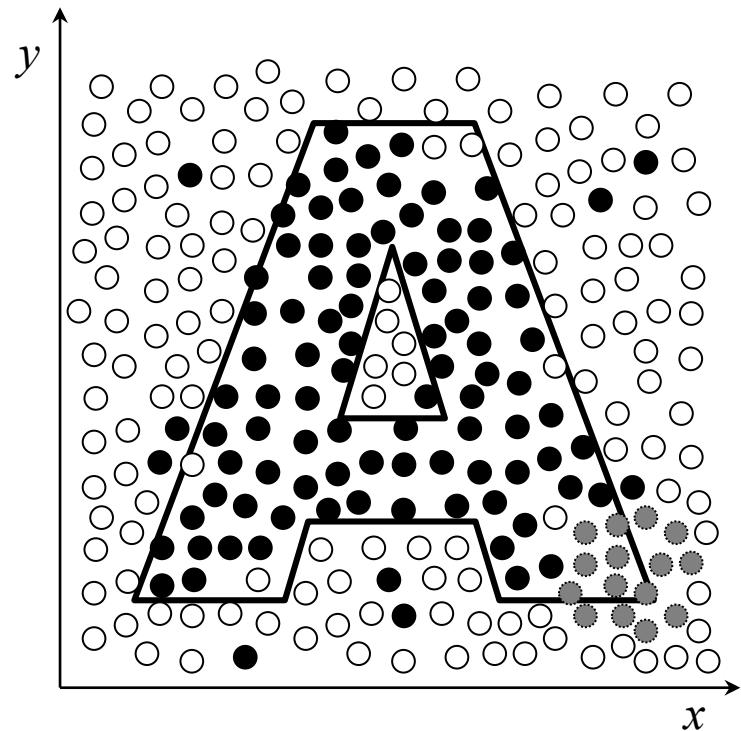
## » Conceitos: conjunto de treino

e.g.

atributo-resposta (atributo-alvo)

2 atributos-preditivos

- ruído nos dados (*noisy data*);
- dados em falta (*missing data*);
- exemplos fronteiriços (*borderline*);
- dados desequilibrados (*imbalanced data*).



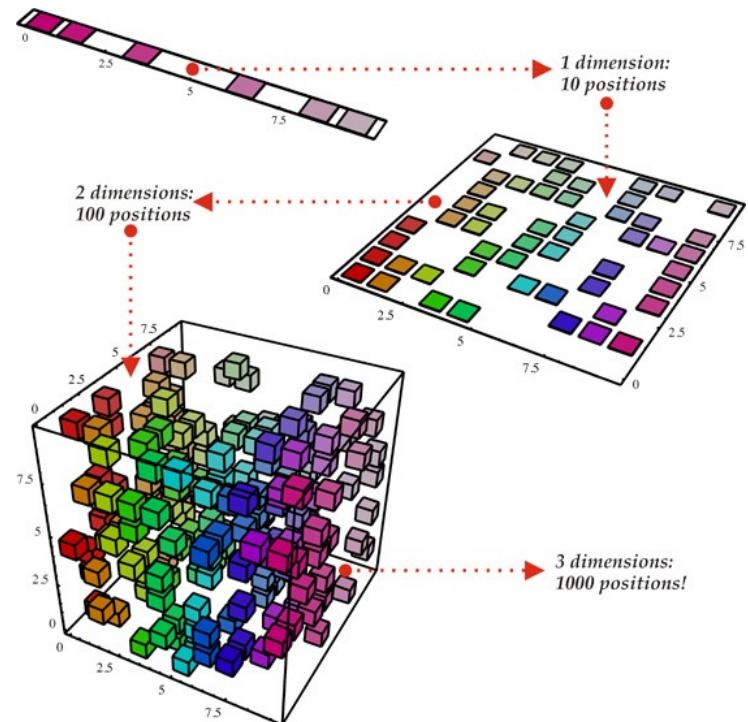


# Modelos preditivos



## » Conceitos: *curse of dimensionality*

- Quanto mais atributos houver, mais informação estará disponível.
- Mas também maior será o espaço de soluções a explorar.



Y. Bengio (2008) "CurseDimensionality.jpg"

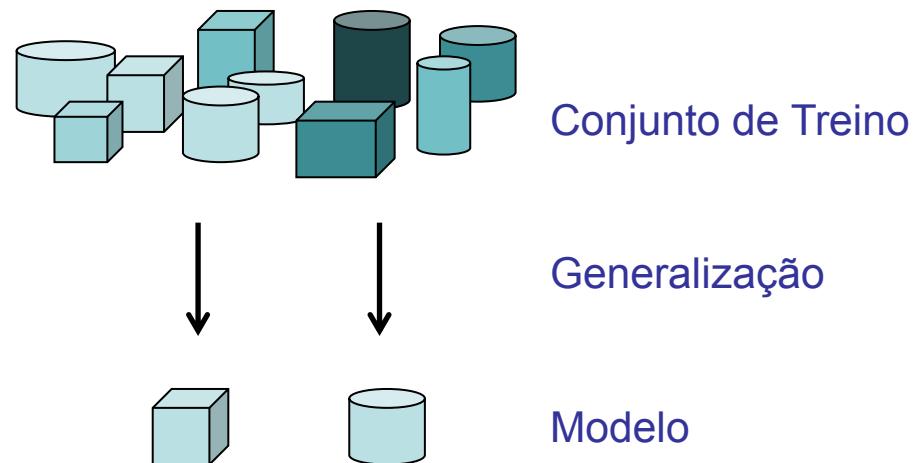


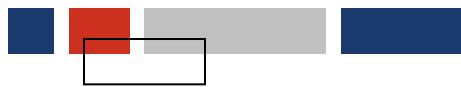
# Modelos preditivos



## » Conceitos: generalização

- *lazy learner*: as *queries* são feitas ao conjunto de treino;
- *eager learner*: as *queries* são feitas ao modelo após a generalização.



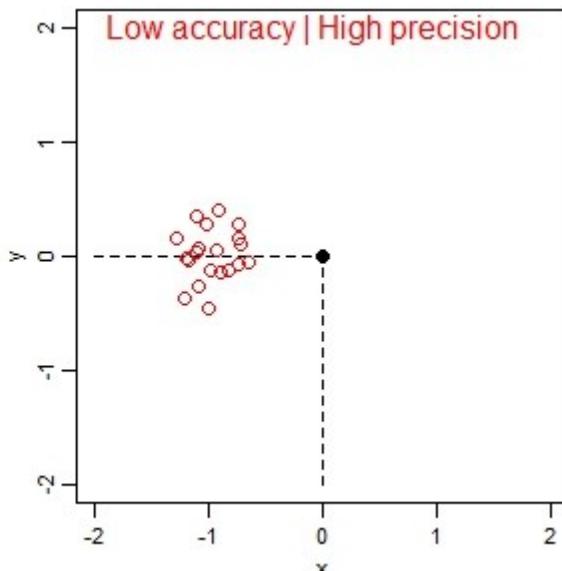
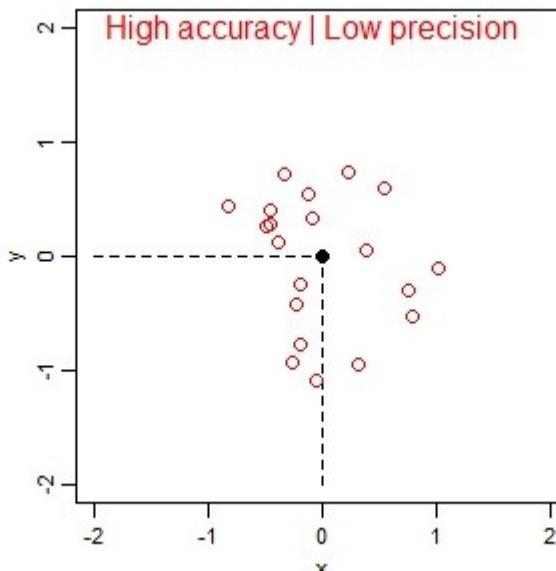
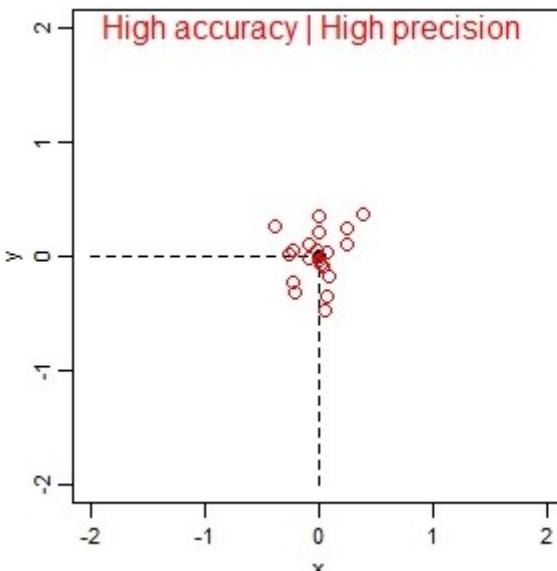


# Modelos preditivos



## » Conceitos: precisão e exactidão

- Os conceitos **precisão** e **exactidão** não são antagónicos;
- Mas é comum que para obter uma maior **precisão** se tenha que “sacrificar” a **exactidão** (e.g. k-NN, Árvores de decisão).



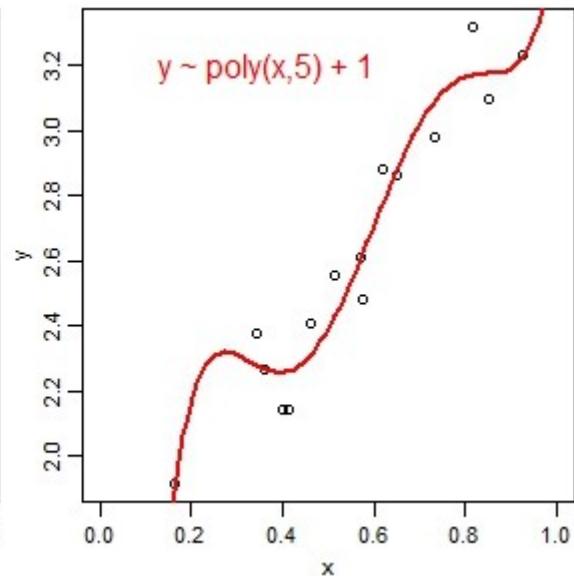
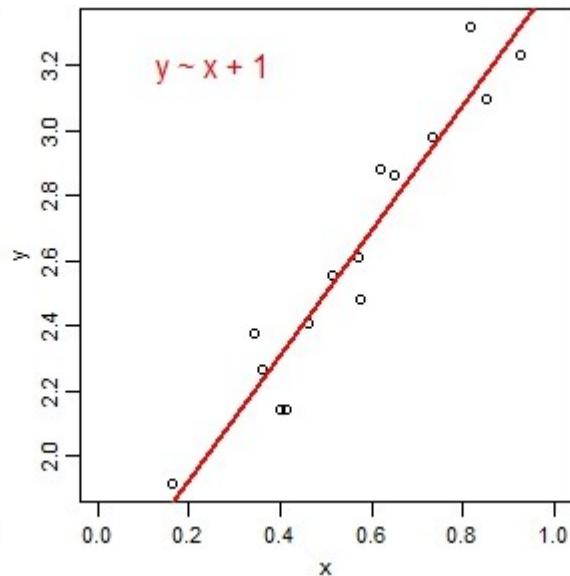
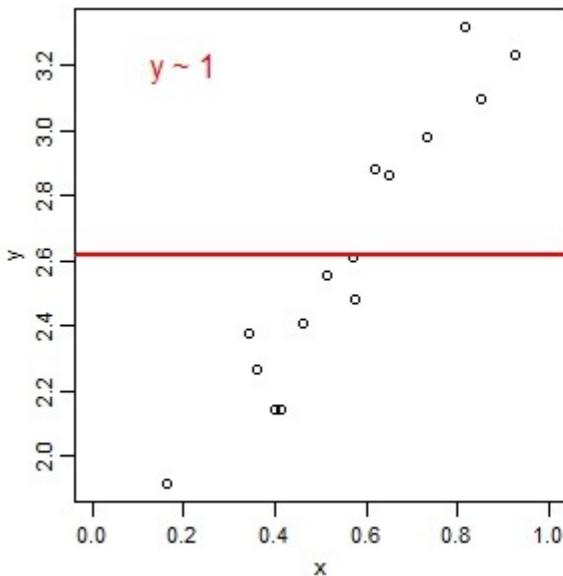


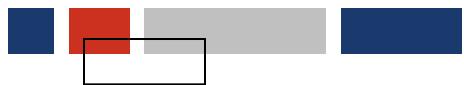
# Modelos preditivos



## » Conceitos: sub- e sobre-ajustamento

- sub-ajustamento: extrai pouco conhecimento dos dados;
- sobre-ajustamento: não constrói **generalizações** dos dados.



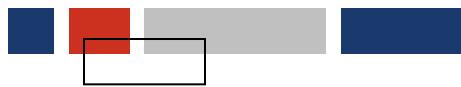


# Modelos preditivos



## » Conceitos: critérios de paragem

- A maior parte dos algoritmos dos modelos preditivos são iterativos;
- Em cada iteração estes algoritmos tentam ajustar melhor o modelo ao conjunto de treino;
- Este ajustamento pára quando se atinge determinado critério:
  - i) número de **iterações máximo** é atingido;
  - ii) erro reduz-se abaixo de um **limiar de erro** (e.g.  $\varepsilon = 0$ );
  - iii) **não há alteração** do modelo.

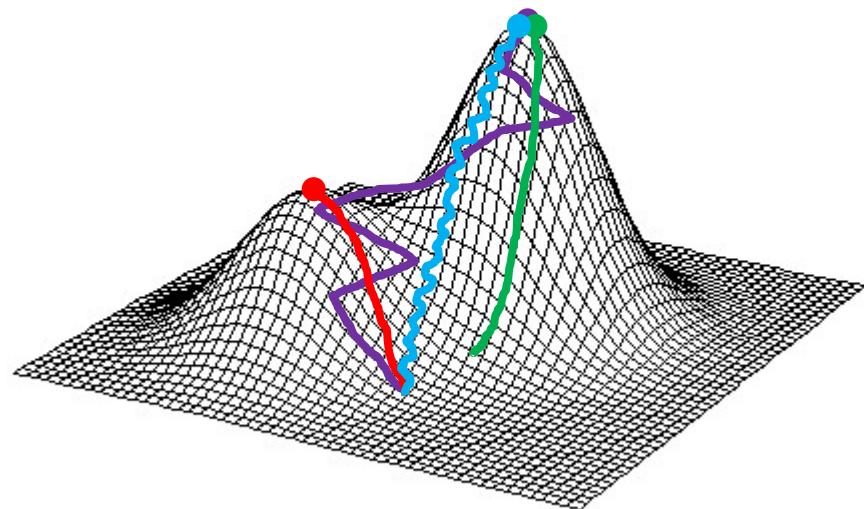


# Modelos preditivos

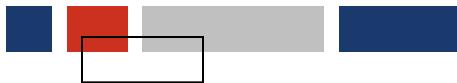


## » Conceitos: máximos locais e globais

- *hill climbing* (algoritmo greedy);
- *stochastic hill climbing*;
- *simulated annealing*.

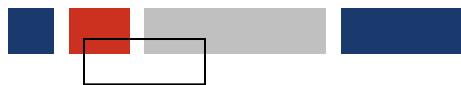


adaptado de Headlessplatter (2007) "Local\_maximum.png"



## » Enquadramento

- O k-NN, ou algoritmo dos vizinhos mais próximos, é o mais simples de todos os algoritmos de Data Mining e é baseado em **distâncias**;
- Pode ser usado em **classificação** ou regressão (i.e. **estimação**);
- O exemplo a avaliar é comparado aos exemplos mais próximos do conjunto de treino;
- É considerado ***lazy learner*** (cf. ***eager learning***), porque não constrói generalizações a partir dos exemplos de treino, estando mais suscetível a **ruído** e a **dados em falta**.



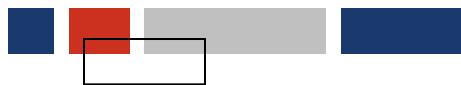
## » k - Nearest Neighbour (k-NN)

- O algoritmo k-NN é um método baseado em distâncias;
- Atributos quantitativos: a distância euclidiana é a mais usual

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

- Atributos qualitativos: e.g., distância de *Hamming*

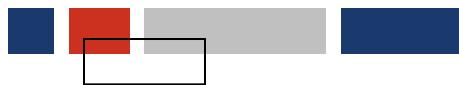
$$d(x_i, x_j) = \begin{cases} 0 & \text{sse } x_i = x_j \\ 1 & \text{sse } x_i \neq x_j \end{cases}$$



## » Algoritmo para Classificação

Dado um ponto  $x = \{?\}$  a classificar:

- 1) Calcular a distância desse ponto a todos os pontos do conjunto de treino.
- 2) Observar os  $k$  vizinhos mais próximos, cada vizinho vota numa classe;
- 3) Escolhe-se a **classe mais votada**;
- 4) O ponto a classificar é classificado pela classe mais votada.



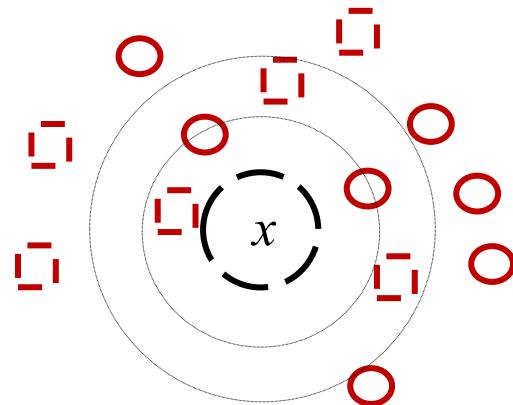
## » Algoritmo para Regressão

Dado um ponto  $x = \{?\}$  a estimar:

- 1) Calcular a distância desse ponto a todos os pontos do conjunto de treino.
- 2) Observar os  $k$  vizinhos mais próximos;
- 3) Calcula-se uma média ponderada pelas distâncias;
- 4) A estimação é dada por essa média.



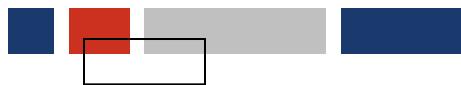
## » Exemplo



Para 1 vizinho mais próximo,  $x$  é classificado como □

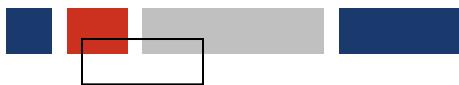
Para 3 vizinhos mais próximos,  $x$  é classificado como ○

Para 5 vizinhos mais próximos,  $x$  é classificado como □

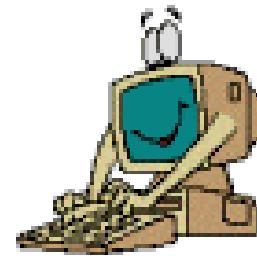


## » Vantagens e Desvantagens

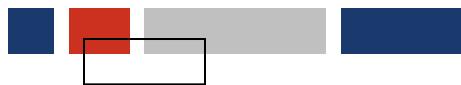
- Modelos são simples de usar e interpretar.
- Não faz generalizações e é susceptível a ruído e a dados em falta;
- Podem-se tornar computacionalmente intensos com números elevados de atributos (i.e. *curse of dimensionality*);
- Escolha da métrica de distância e do parâmetro  $k$  pode ser complexa (mas existem algoritmos desenvolvidos para o efeito);



# Exercício



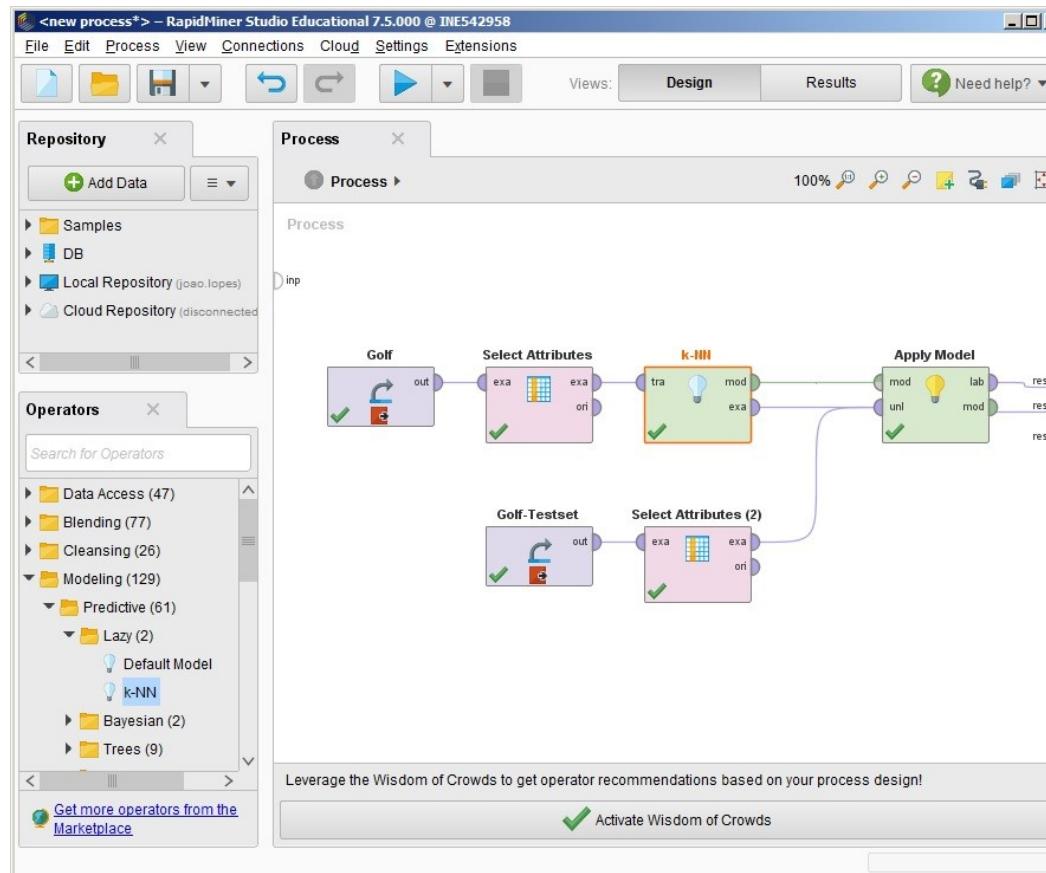
## » Exercício 4.9.

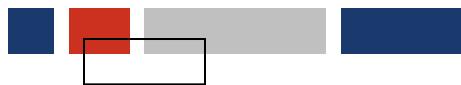


# k-NN



## » No RapidMiner





# k-NN



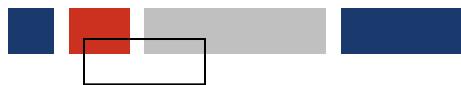
## » No R

```
library("class")

trainset <- read.csv("../data/golf.csv", header=TRUE)
testset <- read.csv("../data/golf-testset.csv", header=TRUE)

res <- knn(trainset[,c("Temperature", "Humidity")], #training set (attr.)
           testset[,c("Temperature", "Humidity")], #testing set
           trainset[, "Play"],                      #training set (cl.)
           k=1)                                     #k-nearest neighbour

#Confusion Matrix
conf_mat <- table(testset[,c("Play")], res)
conf_mat
```



# k-NN



```
#Error rate  
  
err_rt <- 100*(sum(conf_mat) - sum(diag(conf_mat)))/sum(conf_mat)  
  
err_rt
```

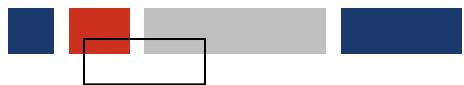


# Naive Bayes



## » Enquadramento

- Este algoritmo, baseado em **probabilidades**, tem sido estudado desde os anos 60, mas mantém-se competitivo com métodos recentes mais avançados (nomeadamente na **Categorização de Texto**);
- Pressupõe ingenuamente (*naive*) a **independência** entre atributos, mas é robusto a ligeiras violações deste pressuposto;
- Apesar de poder estimar as probabilidades com **pouca exactidão**, tipicamente consegue classificar de forma **exacta**;
- O algoritmo requer um número de parâmetros linear com o número de atributos (i.e. não sofre de **curse of dimensionality**).



# Naive Bayes



## » Teorema de Bayes

- O algoritmo *Naive Bayes* é um **método probabilístico** baseado no **teorema de Bayes**.

$$P(\theta | D) = \frac{P(\theta) \cdot P(D | \theta)}{P(D)}$$
$$P(\theta, D) = P(\theta) \cdot P(D | \theta)$$
$$P(\theta, D) = P(D) \cdot P(\theta | D)$$

$P(\theta)$  - Prob. dos parâmetros  $\theta$  (**Probabilidade a priori**);

$P(D)$  - Prob. dos dados  $D$  (**Probabilidade dos dados**);

$P(\theta | D)$  - Prob. dos parâmetros  $\theta$  dado os dados  $D$  (**Probabilidade posterior**).

$P(D | \theta)$  - Prob. dos dados  $D$  dados os parâmetros  $\theta$  (**Verosimilhança**)

$P(T, \theta)$  - Prob. dos dados  $D$  e dos parâmetros  $\theta$  (**Probabilidade conjunta**)



# Naive Bayes



## » Naive Bayes

- Probabilidade de cada classe  $c$  para um conjunto de teste  $T$  é

$$P(c | T) = \frac{P(c) \cdot P(T | c)}{P(T)}$$

onde,

$P(c)$  é a probabilidade *a priori* de cada classe  $c$  (e.g. proporcional à frequência de cada classe; uniforme; ...);

$P(T)$  é a probabilidade de observar o conjunto de teste  $T$ ;

$P(T | c)$  é a *verosimilhança*, ou seja a probabilidade de observar  $T$  para cada classe  $c$ .



# Naive Bayes



## » Naive Bayes

- Sendo  $X_1, \dots, X_p$  valores de  $p$  atributos do conjunto de teste  $T$ ,

$$P(c | X_1, \dots, X_p) = \frac{P(c) \cdot P(X_1, \dots, X_p | c)}{P(X_1, \dots, X_p)}$$

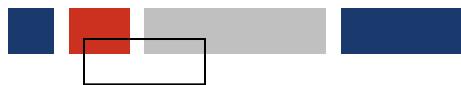
- $P(X_1, \dots, X_p)$  é constante para todas as classes. Para comparar classes só precisamos do numerador,

$$P(c | X_1, \dots, X_p) \propto P(c) \cdot P(X_1, \dots, X_p | c)$$

- Assumindo independência entre atributos,

$$P(c | X_1, \dots, X_p) = P(c) \prod_{i=1}^p P(X_i | c)$$





# Naive Bayes

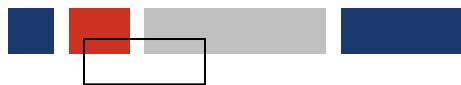


## » Exemplo Naive Bayes

Conjunto de Teste (1000 elementos):

Classe	Tamanho		Doçura		Cor		Total
	Longa	Curta	Doce	Amarga	Amarela	Outra	
Banana	400	100	350	150	450	50	500
Laranja	0	300	150	150	300	0	300
Outra	100	100	150	50	50	150	200
Total	500	500	650	350	800	200	1000

Exemplo a testar  $x = \{\text{Longa}; \text{Doce}; \text{Amarela}\}$



# Naive Bayes



## » Exemplo Naive Bayes

Probabilidades a priori:

$$P(\text{Banana}) = 0.5$$

$$P(\text{Laranja}) = 0.3$$

$$P(\text{Outra}) = 0.2$$

Probabilidade dos dados:

$$P(\text{Longa}) = 0.5$$

$$P(\text{Doce}) = 0.65$$

$$P(\text{Amarela}) = 0.8$$

Verosimilhança:

$$P(\text{Longa} \mid \text{Banana}) = 0.8$$

$$P(\text{Longa} \mid \text{Laranja}) = 0.0$$

$$P(\text{Longa} \mid \text{Outra}) = 0.5$$

$$P(\text{Doce} \mid \text{Banana}) = 0.7$$

$$P(\text{Doce} \mid \text{Laranja}) = 0.5$$

$$P(\text{Doce} \mid \text{Outra}) = 0.75$$

$$P(\text{Amarela} \mid \text{Banana}) = 0.9$$

$$P(\text{Amarela} \mid \text{Laranja}) = 1.0$$

$$P(\text{Amarela} \mid \text{Outra}) = 0.25$$





# Naive Bayes

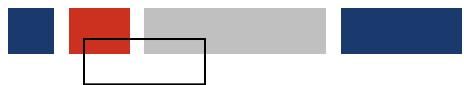


## » Exemplo Naive Bayes

$$\begin{aligned} P(\text{Banana} \mid \text{Longa, Doce, Amarela}) &= \frac{P(\text{Banana})P(\text{Longa, Doce, Amarela} \mid \text{Banana})}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \\ &= \frac{0.252}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \end{aligned}$$

$$\begin{aligned} P(\text{Laranja} \mid \text{Longa, Doce, Amarela}) &= \frac{P(\text{Laranja})P(\text{Longa, Doce, Amarela} \mid \text{Laranja})}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \\ &= \frac{0}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \end{aligned}$$

$$\begin{aligned} P(\text{Outra} \mid \text{Longa, Doce, Amarela}) &= \frac{P(\text{Outra})P(\text{Longa, Doce, Amarela} \mid \text{Outra})}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \\ &= \frac{0.01875}{P(\text{Longa}) \cdot P(\text{Doce}) \cdot P(\text{Amarela})} \end{aligned}$$



# Naive Bayes



## » Exemplo Naive Bayes

$$\frac{P(\text{Banana} \mid \text{Longa, Doce, Amarela})}{P(\text{Laranja} \mid \text{Longa, Doce, Amarela})} = \frac{0.252}{0} = +\infty$$

$$\frac{P(\text{Banana} \mid \text{Longa, Doce, Amarela})}{P(\text{Outra} \mid \text{Longa, Doce, Amarela})} = \frac{0.252}{0.01875} = 13.44$$



# Naive Bayes



## » Naive Bayes com atributos contínuos

- Atributos distribuídos de acordo com uma **distribuição normal**. Assim a probabilidade do atributo  $x = v$  é dada por,

$$P(x = v | c) = \text{Normal}(x = v | \mu_c, \sigma_c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v-\mu_c)^2}{2\sigma_c^2}}$$

onde  $\mu_c$  e  $\sigma_c$  são a média e o desvio padrão dos valores de  $x$  da classe  $c$ ;

ou

- Transformação dos valores contínuos em valores discretos.

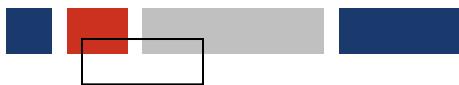


# Naive Bayes

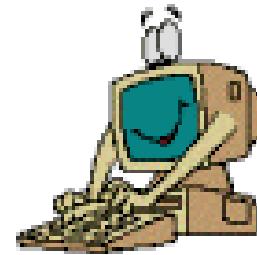


## » Vantagens e Desvantagens

- Modelos são simples de usar e interpretar;
  - Permitem fazer generalizações;
  - Podem usar facilmente grandes quantidades de dados (i.e. não sofre de *curse of dimensionality*).
- 
- As probabilidades obtidas podem ser pouco exactas (sobretudo para conjuntos de teste pequenos);
  - O pressuposto de independência entre atributos raramente se verifica (apesar de alguma robustez à sua violação).



# Exercício

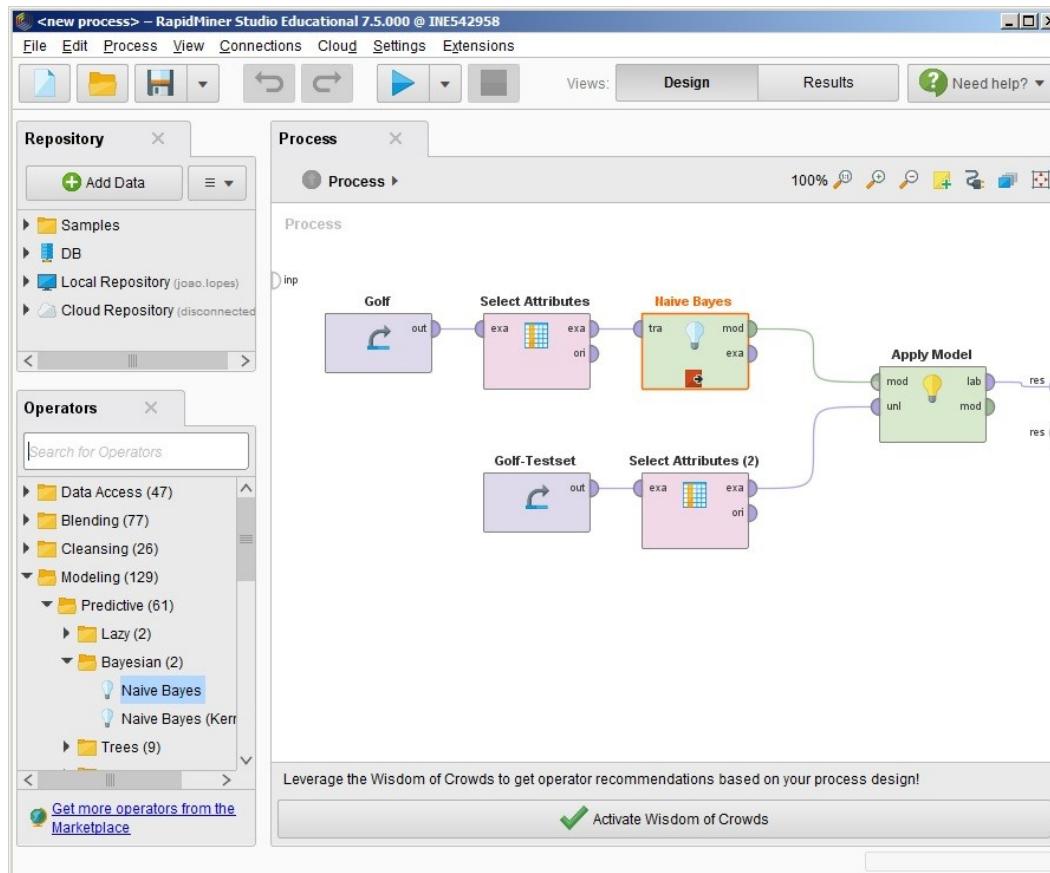


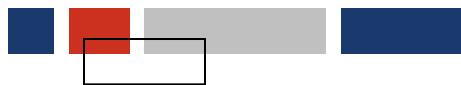
» Exercício 4.10 e 4.11.

# Naive Bayes



## » No RapidMiner





# Naive Bayes



## » No R

```
library("e1071")

trainset <- read.csv("../data/golf.csv", header=TRUE)
testset <- read.csv("../data/golf-testset.csv", header=TRUE)

mod <- naiveBayes(Play ~ Outlook + Wind,           #model design
                  trainset,                      #training set
                  laplace=0)                   #Laplace correction
res <- predict(mod,                            #model
               testset[,c("Outlook", "Wind")]) #testing set

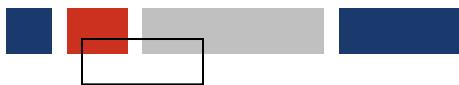
#Confusion Matrix
conf_mat <- table(testset[,c("Play")], res)
conf_mat
```



# Naive Bayes



```
#Error rate  
  
err_rt <- 100*(sum(conf_mat) - sum(diag(conf_mat)))/sum(conf_mat)  
err_rt
```



# Árvores de Decisão



## » Árvores de Decisão

- Algoritmos baseados em **procura** que produzem uma **estrutura de árvore** com base no conjunto de treino;
- Produzem **modelos facilmente interpretáveis** e são bastante úteis para **descrever os dados**;
- Podem ser usados em **classificação** ou regressão (i.e. **estimação**);
- Para variáveis-alvo qualitativas usa-se **árvores de classificação**, para variáveis quantitativas usa-se **árvores de regressão**.



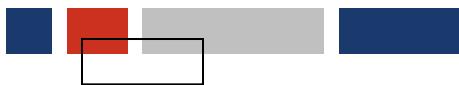


# Árvores de Decisão



## » Estrutura de Árvore

- A estrutura de árvore é um **gráfico de fluxo** em forma de árvore.
- A árvore é composta pelos seguintes elementos: **nó-raiz**, **nós-internos**, **ramos** e **nós-folha**.
- O **nó-raiz** é o nó que inicia a árvore, os **nós-internos** representam um teste a um atributo, os **ramos** representam o resultado do teste, e os **nós-folha** são portadores da informação sobre a variável-alvo.



# Árvores de Decisão



## » Exemplo\*

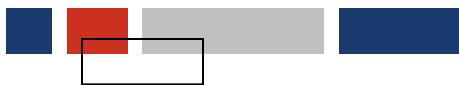
Processo de decisão na concessão de crédito bancário

Conjunto de teste: 99 exemplos, 5 atributos (incl. atributo-alvo)

Exemplo de dados:

Montante	Idade	Salário	Conta	Decisão
100000	25	18000	sim	concedido
150000	20	10000	não	não concedido
300000	60	40000	sim	concedido
50000	30	17000	sim	concedido
250000	59	16500	não	não concedido
...				

\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



# Árvores de Decisão



## » Exemplo\*

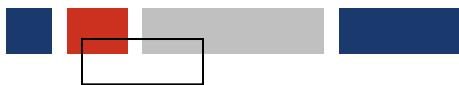
Processo de decisão na concessão de crédito bancário

Conjunto de teste: 99 exemplos, 5 atributos (incl. atributo-alvo)

Exemplo de dados:

Montante	Idade	Salário	Conta	Decisão
médio	júnior	médio	sim	concedido
médio	júnior	baixo	não	não concedido
alto	sénior	alto	sim	concedido
baixo	júnior	médio	sim	concedido
alto	sénior	médio	não	não concedido
...				

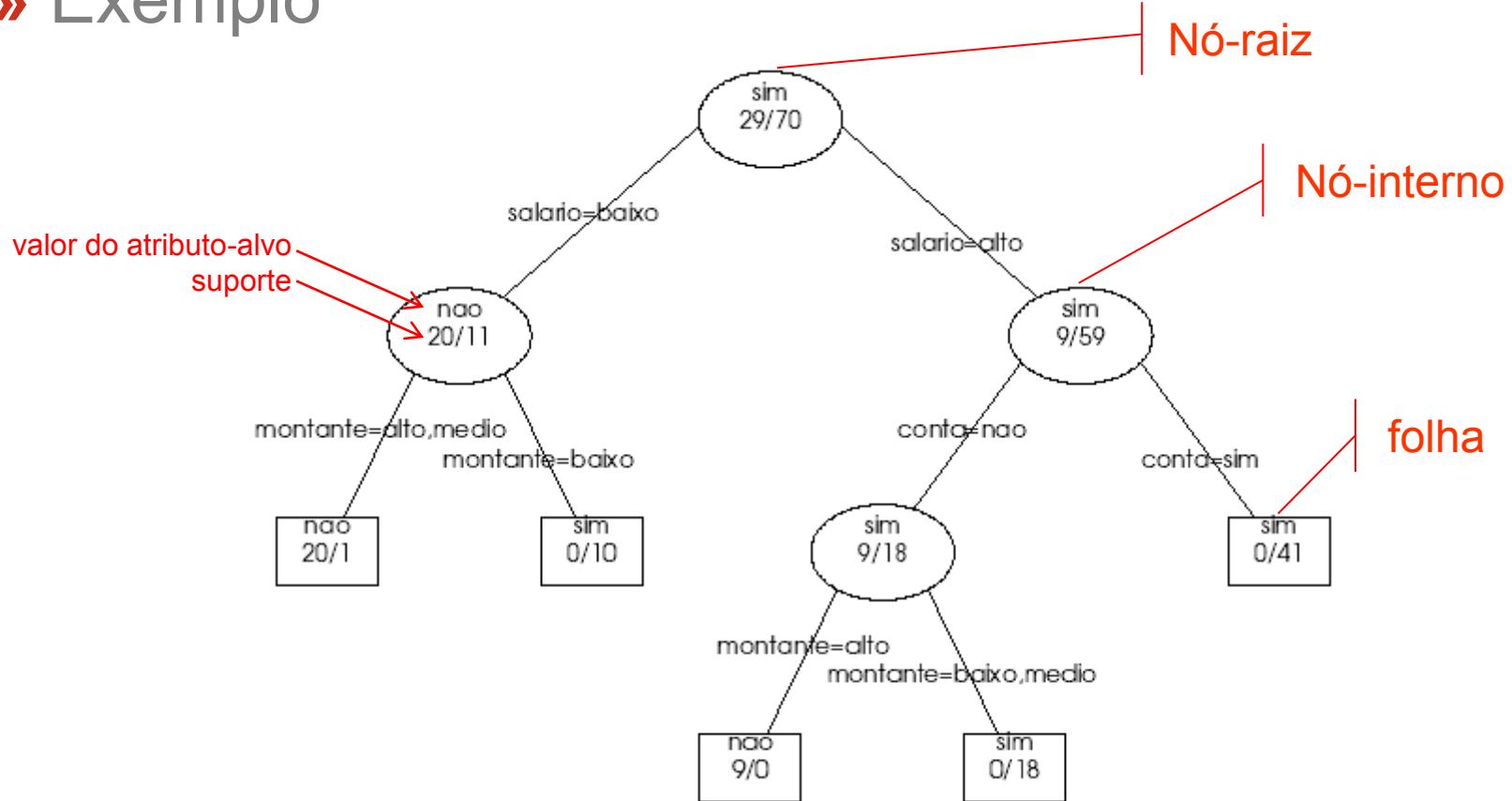
\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



# Árvores de Decisão



## » Exemplo\*



\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



# Árvores de Decisão



## » Algoritmo

- Como escolher o nó-raiz (e os nós-internos subsequentes)?
- Como dividir e (subdividir) o conjunto de teste?
- Quando parar de acrescentar nós-internos e terminar numa folha?
- Que valor dar a cada folha?



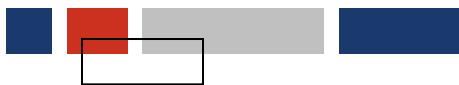
# Árvores de Decisão



## » Algoritmo básico para atributos binários

- Escolher a ordem dos atributos a usar aleatoriamente (e.g. **ordem alfabética**);
- Para **atributos binários** (e.g.  $x = \{0,1\}$ ) a divisão nos seus dois valores é natural;
- Criar folhas só quando obtiver **folhas “puras”** (i.e. subconjunto de teste com apenas um dos valores da variável-alvo);
- O valor da folha “pura” é o **valor único** da variável-alvo.





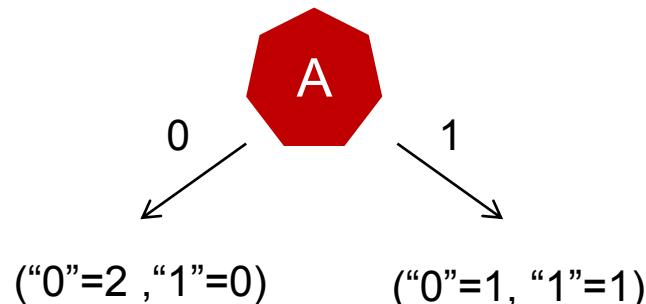
# Árvores de Decisão



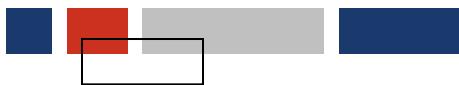
## » Exemplo de algoritmo básico

Diagrama em árvore da conjunção (i.e. AND)\*

A	B	$\wedge$
0	0	0
0	1	0
1	0	0
1	1	1



\* Adaptado de Gama, J. Faculdade de Economia do porto, EDC1, 2000/2001



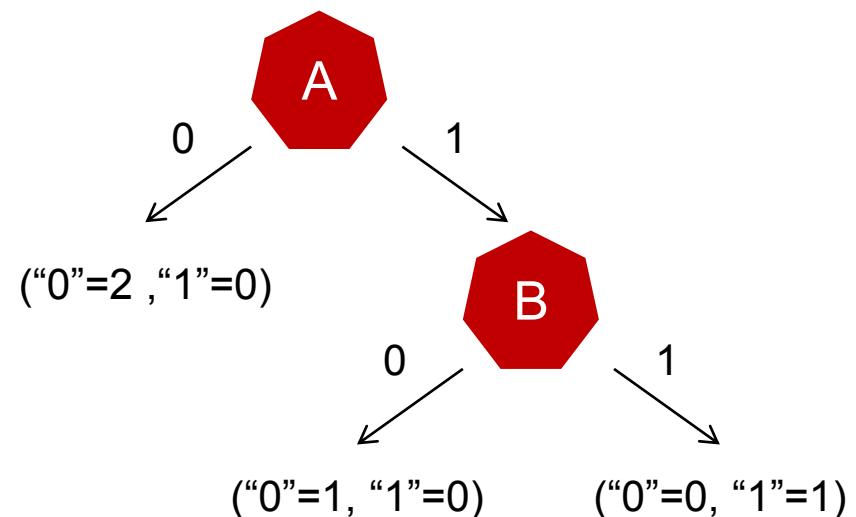
# Árvores de Decisão



## » Exemplo de algoritmo básico

Diagrama em árvore da conjunção (i.e. AND)\*

A	B	$\wedge$
0	0	0
0	1	0
1	0	0
1	1	1



\* Adaptado de Gama, J. Faculdade de Economia do porto, EDC1, 2000/2001



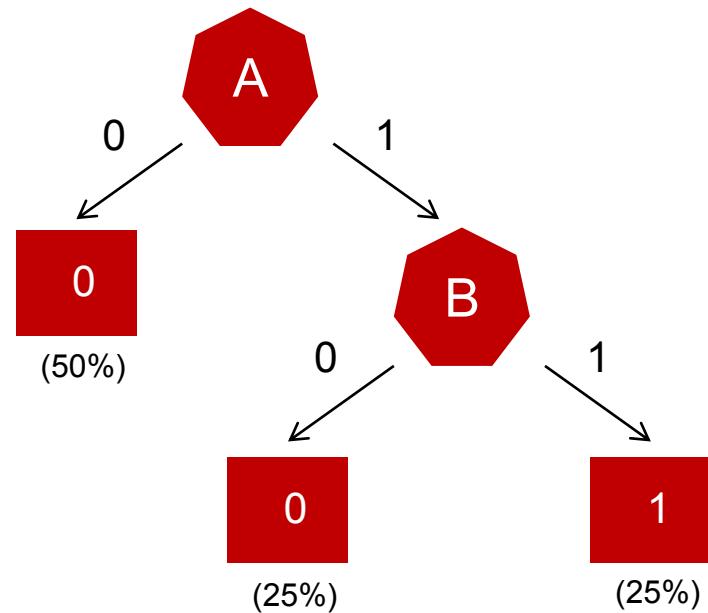
# Árvores de Decisão



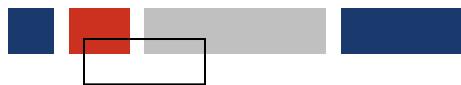
## » Exemplo de algoritmo básico

Diagrama em árvore da conjunção (i.e. AND)\*

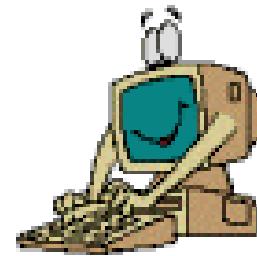
A	B	$\wedge$
0	0	0
0	1	0
1	0	0
1	1	1



\* Adaptado de Gama, J. Faculdade de Economia do porto, EDC1, 2000/2001



# Exercício



## » Exercício 4.12.

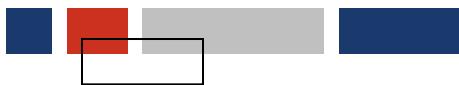


# Árvores de Decisão



## » Algoritmos de “particionamento recursivo”

- As árvores são criadas **dividindo recursivamente** o conjunto de teste em subconjuntos (i.e. os subconjuntos formados são eles próprios divididos em subconjuntos mais pequenos);
- A escolha do atributo para o nó-raiz (e nós-internos subsequentes) e como realizar a sua divisão é feito de modo a obter a “**melhor**” divisão dos dados após uma **procura exaustiva**;
- Esta avaliação depende da **métrica** escolhida (tipicamente relacionadas com a **homogeneidade** dos subconjuntos formados);
- Esta divisão termina quando é atingido determinado **critério de paragem** (e.g. obtém-se uma **folha “pura”**; o **suporte** da folha não reduz abaixo de determinado limiar; ...).



# Árvores de Decisão



## » Divisão recursiva

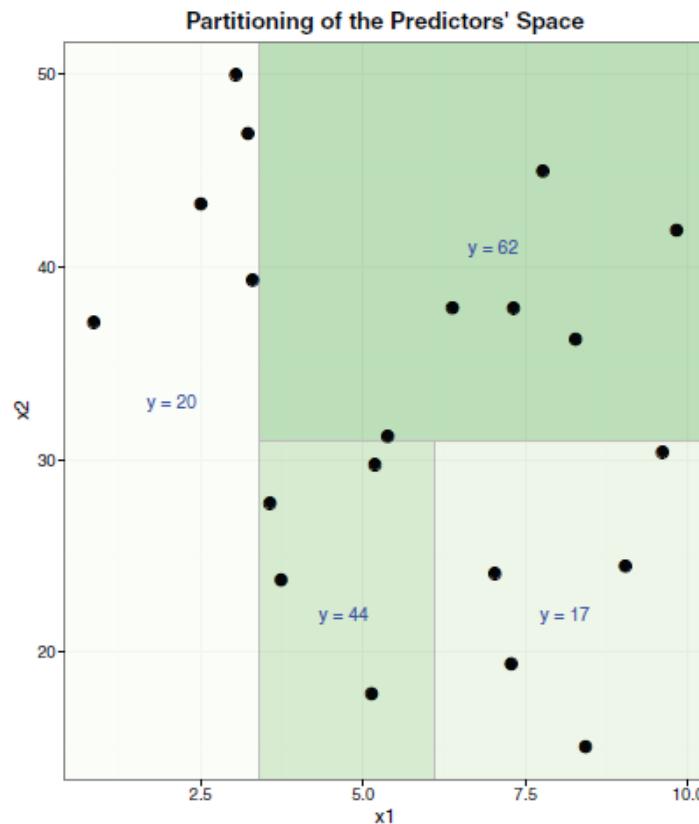
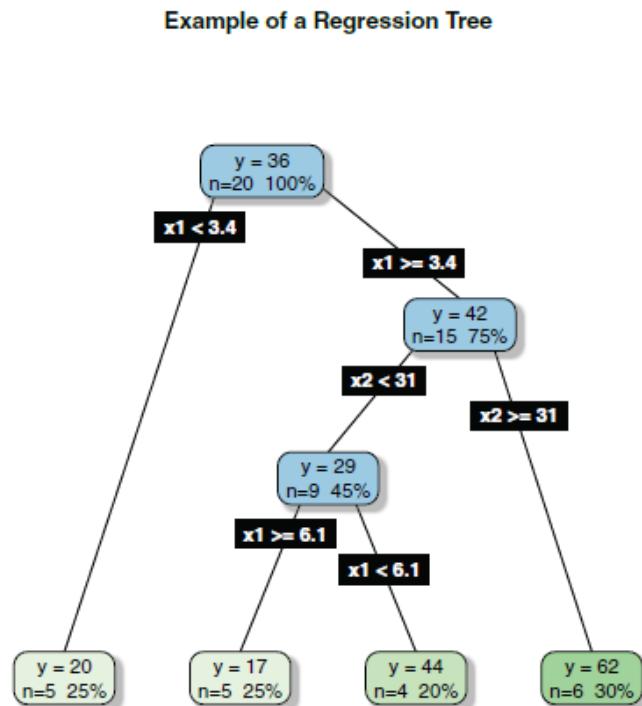
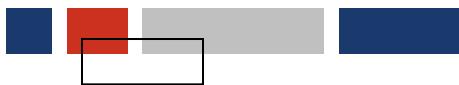


Figura 3.27 de Torga L. “Data Minig with R” (2017)



# Árvores de Decisão



## » Algoritmos (Métricas)

C4.5

métrica *information gain*

divisões multiplas

**CART** (Classification and Regression Trees)

métricas *Gini impurity* (categórica) e Redução de Variância (contínua)

divisões binárias



# Árvores de Decisão



## » Critério de paragem e Poda (*pruning*)

- **Árvores grandes de folhas pequenas:** resultados **demasiado específicos** e com **muito ruído** (i.e. sobre-ajustamento);
- **Árvores pequenas de folhas grandes:** resultados extraem **pouco conhecimento** dos dados (i.e. sub-ajustamento);

**Poda** (substituir nós-internos por folhas):

- 1) parar a construção da árvore quando se atinge determinado critério (i.e. **pré-poda**)
- 2) criar árvores muito grandes, seguido de eliminação de ramos estatisticamente pouco significantes (i.e. **pós-poda**).





# Árvores de Decisão



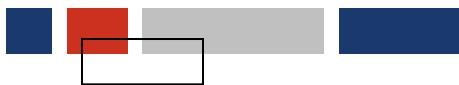
## » Valores das folhas

### **Variável-alvo categórica**

valor **mais frequente** do subconjunto de teste da folha

### **Variáveis-alvo contínua**

valor **médio** do subconjunto de teste da folha



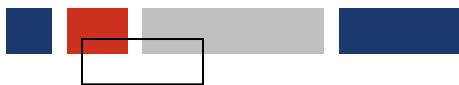
# Árvores de Decisão



## » Regras de Decisão

- As Árvores de Decisão podem ser decompostas em Regras de Decisão;
- Estas Regras de Decisão podem ser obtidas simplesmente por atravessar as árvores desde o nó-raiz até às folhas;
- As Regras de Decisão são da forma **IF A THEN B** (e.g. IF salário= baixo & montante = baixo **THEN** conceder = sim)
- O conjunto completo das Regras de Decisão é equivalente à Árvore de Decisão;
- Regras podem ser interpretadas isoladamente aumentando a interpretabilidade da Árvore

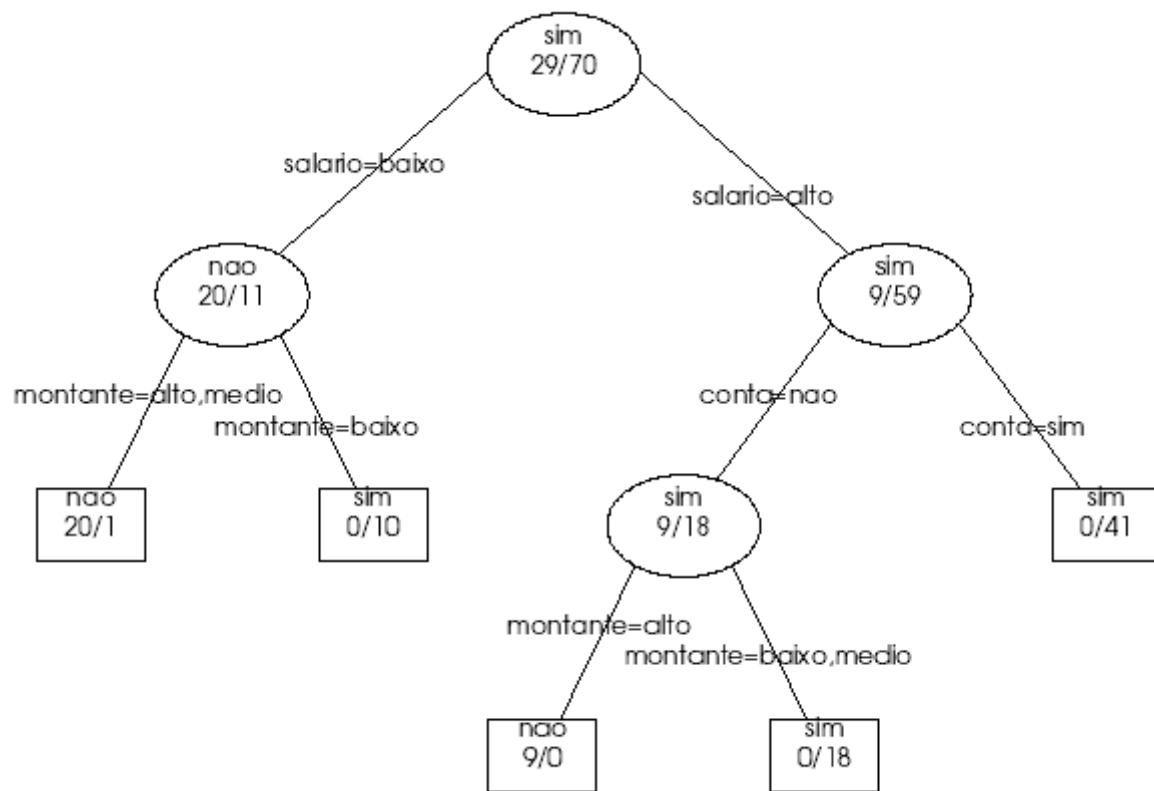




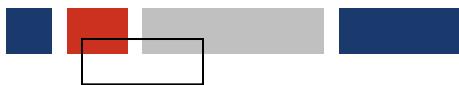
# Árvores de Decisão



## » Exemplo\*



\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



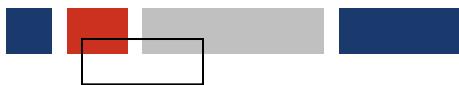
# Árvores de Decisão



## » Exemplo\*

- 1) **IF** salário=baixo & montante={alto,médio} **THEN** não conceder
- 2) **IF** salário=baixo & montante=baixo **THEN** conceder
- 3) **IF** salário=alto & conta=não & montante=alto **THEN** não conceder
- 4) **IF** salário=alto & conta=não & montante={baixo,médio} **THEN** conceder
- 5) **IF** salário=alto&conta=sim **THEN** conceder

\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



# Árvores de Decisão



## » Exemplo\*

**IF** salário=baixo

**IF** montante={alto,médio} **THEN** não conceder

**ELSE** conceder

**ELSE**

**IF** conta=não

**IF** montante=alto **THEN** não conceder

**ELSE** conceder

**ELSE** conceder

\*Faculdade de Economia do Porto, Apontamentos de Informática Aplicada, 2011



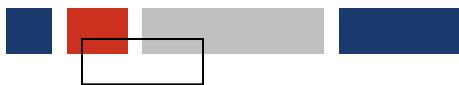
# Árvores de Decisão



## » Regras de Decisão

- As Regras de Decisão podem ser obtidas **directamente dos dados**;
- Algoritmos para extração de regras variam em complexidade:
  - One R** (One Rule): regras geradas com base num atributo
  - Top Down**: parte de uma regra mais geral e vai acrescentando condições (de atributos presentes ou de outros)
  - Bottom up**: parte de uma regras mais específica e vai retirando condições (eliminando atributos ou não)





# Árvores de Decisão



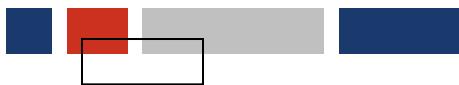
## » Regras de Decisão: One R

- 1) Para cada atributo  $A$  constrói-se uma regra  $A = v_i \rightarrow C$ ,  
onde  $v_i$  = valores de  $A$  e  $C$  = classe mais frequente da variável;
- 2) Seleciona-se o atributo que possui menor erro.

- Os **erros** são calculados como:

$$\text{Erro atributo}_i = \frac{erro_i}{total_i}$$

$$\text{Erro atributo} = \frac{\sum_i erro_i}{\sum_i total_i}$$



# Árvores de Decisão



## » Regras de Decisão: Qualidade

- A qualidade de uma regra pode ser medida por:

$$\text{Taxa cobertura} = \frac{\text{número exemplos da regra}}{\text{total}}$$

$$\text{Taxa acerto} = \frac{\text{número exemplos bem classificados}}{\text{número exemplos da regra}}$$

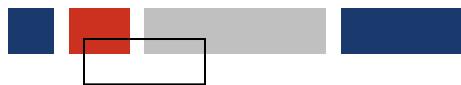


# Árvores de Decisão

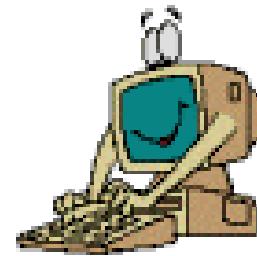


## » Vantagens e Desvantagens

- Modelos são **simples de usar e interpretar**;
- Mimetizam bem o **processo de decisão humana**.
  
- São **pouco exactos** relativamente a outros métodos (problemas de **enviesamento**);
- São pouco robustos a **dados em falta** e **ruído dos dados** (problemas de **generalização**);
- Algoritmos de procura podem não capturar facilmente alguns conceitos de decisão (e.g. XOR).



# Exercício

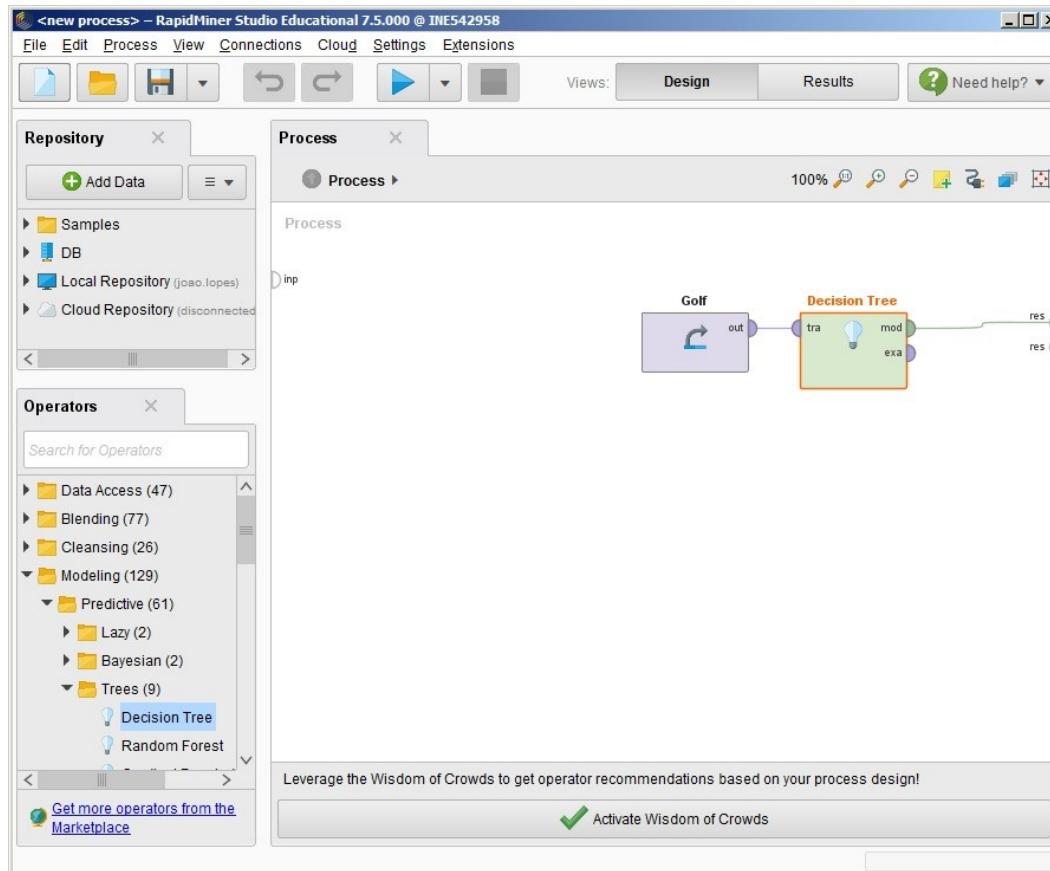


## » Exercício 4.13.

# Árvores de Decisão



## » No RapidMiner



# Árvores de Decisão



## » No RapidMiner

The screenshot shows the RapidMiner Studio interface. On the left, there are several panels: 'Tutorials' (with a section titled 'Build a survival prediction model'), 'Repository' (listing 'Samples', 'DB', 'Local Repository', and 'Cloud Repository'), 'Operators' (listing categories like 'Data Access', 'Blending', 'Cleansing', 'Modeling', 'Scoring', and 'Validation'), and 'Parameters' (showing settings for 'Process' like 'logverbosity', 'logfile', 'resultfile', 'random se...', 'send mail', and a note about compatibility). The central area is the 'Process' canvas, which contains a flow: 'Retrieve Titanic' → 'Set Role' → 'Select Attributes' → 'Decision Tree'. The 'Help' panel on the right provides a synopsis of the 'Process' operator. A message at the bottom encourages activating the 'Wisdom of Crowds' feature.

```
graph LR; Retrieve[Retrieve Titanic] --> SetRole[Set Role]; SetRole --> SelectAttributes[Select Attributes]; SelectAttributes --> DecisionTree[Decision Tree]
```



# Árvores de Decisão



## » No R

```
library("rpart")

trainset <- read.csv("../data/golf.csv", header=TRUE)
testset <- read.csv("../data/golf-testset.csv", header=TRUE)

mod <- rpart(Play ~.,
             trainset,
             method="class",
             parms=list(split="gini"), #Metric "Gini Impurity"
             control=rpart.control(
               minsplit=4, #minimal size for split
               minbucket=2, #minimal leaf size
               cp=0.1, #minimal gain
               maxdepth=20) #maximal depth
           )
```



# Árvores de Decisão



## » No R

```
plot(mod,uniform=TRUE,branch=0.5,margin=0.05,main="Classification Tree")
text(mod,use.n=TRUE,all=TRUE,cex=0.8,fancy=TRUE,pretty=0)
res <- predict(mod,
                 testset,
                 type="class")           #model
                           #testing set
                           #can be "prob" or "class"

#Confusion Matrix
conf_mat <- table(testset[,c("Play")],res)
conf_mat

#Error rate
err_rt <- 100*(sum(conf_mat) - sum(diag(conf_mat)))/sum(conf_mat)
err_rt
```



# Redes Neuronais Artificiais



## » Redes Neuronais Artificiais

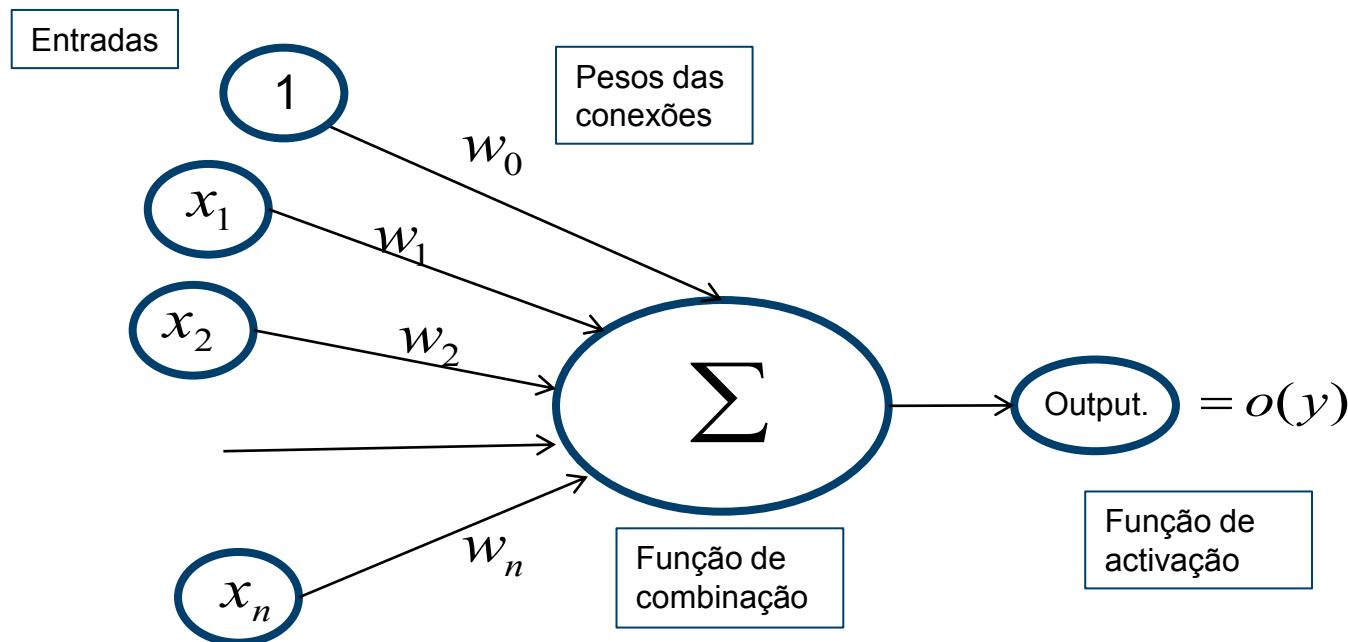
- Algoritmos de **optimização** inspirados nas redes neuronais do cérebro, onde redes densas de simples neurónios interligados realizam processos de aprendizagem muito complexos;
- Os neurónios recebem um **input** de estímulos, estes estímulos juntam-se a partir de uma **função de combinação** e servem de input a uma **função de activação** que produz um **output** de resposta;
- Podem ser usados em **classificação** ou regressão (i.e. **estimação**);
- Os modelos mais simples tem um só neurónio (i.e. **perceptron**), os mais complexos são compostas por **multi-camadas** de neurónios.



# Redes Neuronais Artificiais



## » Single-layer perceptron (SLP)





# Redes Neuronais Artificiais



## » Single-layer perceptron (SLP)

Função de combinação

$$y = w_0 + w_1 x_1 + w_2 x_2 + \dots + w_n x_n$$

$$= \sum_{i=0}^n w_i x_i$$

Função de activação

$$o(y) = \begin{cases} 1 & \text{sse } y > 0 \\ 0 & \text{c.c.} \end{cases}$$

$$o(y) = \frac{1}{1 + e^{-y}}$$

$$o(y) = y$$

- Classificador Linear
- Regressão Múltipla Linear





# Redes Neuronais Artificiais



## » Algoritmo para Classificador Linear

- 1) Inicializa os pesos aleatoriamente;
- 2) Sequencialmente percorre todos os exemplos de treino, onde para cada exemplo:
  - Calcula o resultado ([função de activação](#)) do classificador linear;
  - De acordo com o valor do erro, [actualiza os pesos](#);
- 3) Volta ao 2) quando um exemplo está incorretamente classificado;
- 4) Retorna os pesos atualizados.

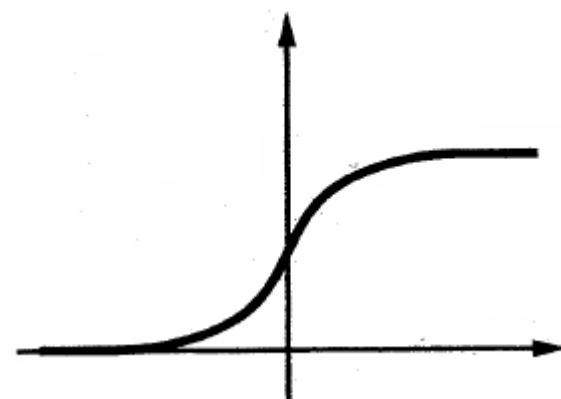
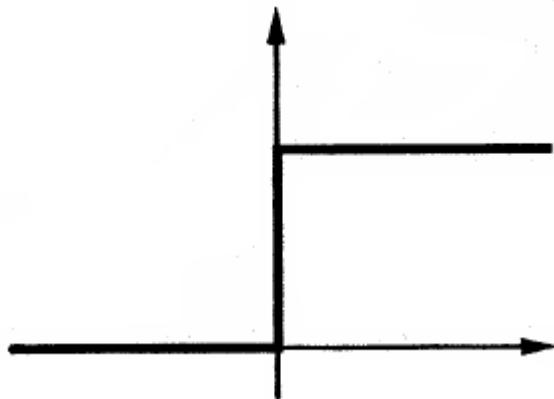


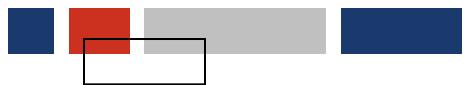
# Redes Neuronais Artificiais



## » Função de activação

- Função não-linear geralmente com output entre 0 e 1;
- Tipicamente, funções degrau ou funções sigmóide



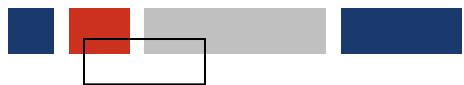


# Redes Neuronais Artificiais



## » Actualização dos pesos (*Backpropagation*)

- Algoritmos de **optimização** constituídos por duas fases: **propagação** e **actualização dos pesos**;
- os inputs **propagam-se para a frente** pelas camadas da rede até à camada de output;
- o output observado é comparado com o esperado usando uma **função de custo** e obtendo o erro;
- o erro **propaga-se para trás** pelas camadas da rede desde o output até que todos os neurónios têm um erro associado;
- esse erro **actualiza os pesos** de cada neurónio minimizando a **função de custo**;



# Redes Neuronais Artificiais



## » Actualização dos pesos (*Backpropagation*)

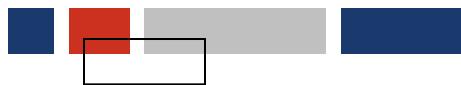
$$w(t+1) = f(x, w(t), \boxed{Err(o_{real}, o(y))})$$

função-custo

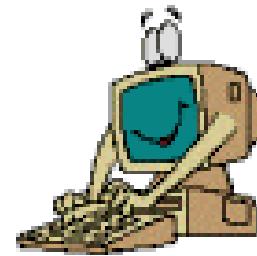
$$w_i(t+1) = w_i(t) + \eta \cdot (\text{Esperado} - \text{Observado}) \cdot x_i$$

**taxa de aprendizagem**

- Define a magnitude do ajuste feito no valor de cada peso;
- A magnitude define a velocidade de convergência da rede.



# Exercício



## » Exercício 4.14

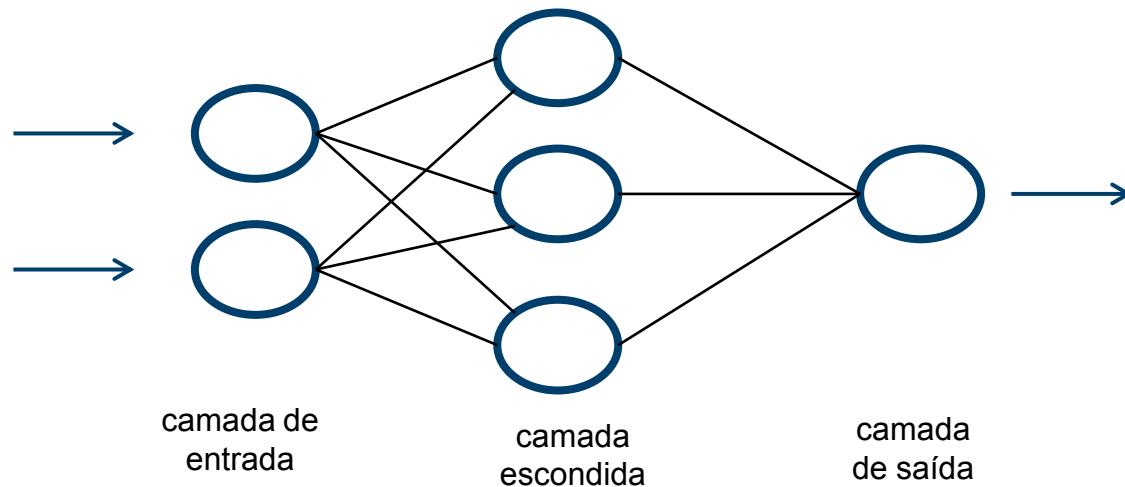


# Redes Neuronais Artificiais



## » Multi-layer Perceptron (MLP)

- As redes podem ser constituídas por diferentes camadas dispostas paralelamente. A primeira designa-se por **camada de entrada**, as camadas intermédias são designadas por **camadas escondidas** e a última camada é designada por **camada de saída**.





# Redes Neuronais Artificiais



## » Dificuldades

1. Inputs e outputs devem ter **valores entre 0 e 1**;
2. Número de neurónios das camadas escondidas influenciam o ajustamento do modelo aos dados:
  - número reduzido pode levar a um **sub-ajustamento**;
  - número elevado pode levar a um **sobre-ajustamento**;
3. Taxa de aprendizagem influencia a **convergência** da optimização:
  - taxa reduzida pode levar a longos tempos de aprendizagem;
  - taxa elevada pode levar a perda de precisão;
4. Pesos iniciais influenciam o **local de convergência** da optimização.



# Redes Neuronais Artificiais



## » Estratégias

1. Transformação dos dados (e.g. minmax para atributos contínuos, variáveis indicadores (*dummy*) para atributos categóricos);
2. Utilizar avaliação de modelos para definir a arquitectura da rede;
3. Taxa de aprendizagem dinâmica (e.g. começar elevada e reduzir progressivamente);
4. Treinar a rede várias vezes utilizando diferentes valores iniciais para os pesos.



# Redes Neuronais Artificiais



## » Vantagens e Desvantagens

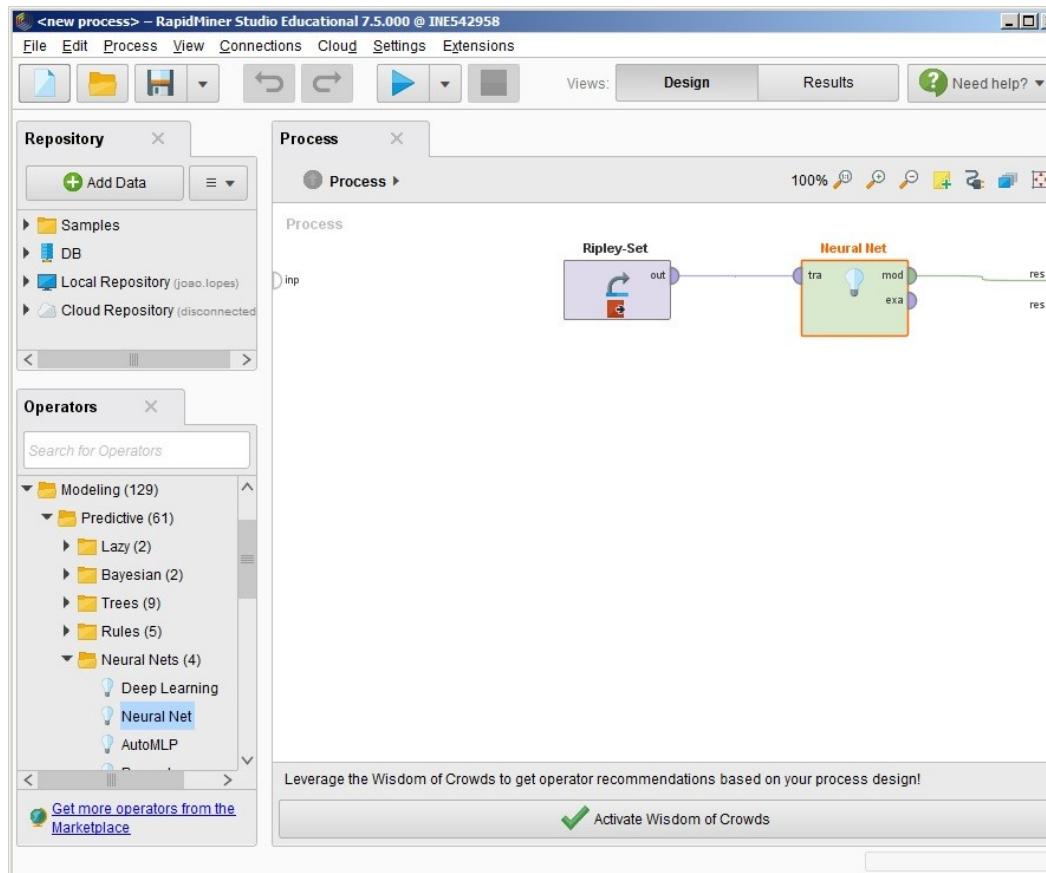
- Grande capacidade de formar **generalizações**, e particularmente tolerante a **ruído** e a **dados em falta**;
- Ajusta **modelos complexos** aos dados sem necessidade da sua discriminação prévia.
- Os parâmetros ajustados dos modelos são de **difícil interpretação** (**blackbox**);
- Os resultados podem ser **pouco robustos** porque dependem da arquitectura da rede e dos pesos iniciais das conexões.



# Redes Neuronais Artificiais



## » No RapidMiner





# Redes Neuronais Artificiais



## » No R

```
library("nnet")
library("caret")
library("NeuralNetTools")

trainset <- read.csv("../data/ripley-set.csv",header=TRUE)
trainset[, "label"] = factor(trainset[, "label"], levels=c("0", "1"))

#Create testing set by sampling with replacement
n <- nrow(trainset)
bootindex <- sample(1:n, size=n, replace=TRUE)
testset <- trainset[bootindex,]

#Select settings for artificial neural network
nn_decay <- 0.3 #weight decay (improved version of learning rate)
nn_maxit <- 500 #maximum number of iterations
```



# Redes Neuronais Artificiais



```
grid1 <- expand.grid(size=1:5,decay=nn_decay)
train1 <- train(label ~.,
                 trainset,
                 maxit=nn_maxit,
                 tuneGrid=grid1,
                 metric="Accuracy",
                 method="nnet",trace=FALSE)

nn_size <- train1$bestTune[1,1] #number of nodes in hidden layer

#Perform Neural Network fit
mod <- nnet(label ~.,
            trainset,
            size=nn_size,decay=nn_decay,maxit=nn_maxit)
print(mod)
summary(mod)
plotnet(mod,cex_val=0.7,circle_cex=3)
olden(mod,bar_plot=TRUE)
```



# Redes Neuronais Artificiais



```
res <- predict(mod,           #model
                testset,       #testing set
                type="class") #can be "raw" or "class"

#Confusion Matrix
conf_mat <- table(testset[,c("label")],res)
conf_mat

#Error rate
err_rt <- 100*(sum(conf_mat) - sum(diag(conf_mat)))/sum(conf_mat)
err_rt
```



# Support Vector Machine



## » Support Vector Machine (SVM)

- Algoritmos de **optimização** que se baseiam na construção de um **hiperplano** óptimo (ou **conjunto de hiperplanos**) com o objetivo de separar as classes de dados com a maior **margem** possível;
- É utilizado para **categorizações lineares**, mas também permite **classificações não-lineares** (utilizando **funções kernel**);
- Tanto permite modelos **supervisionados** (usando conjuntos de testes) como **não-supervisionados** (formando *clusters* naturais);
- Pode ser usado em **classificação** ou regressão (i.e. **estimação**).



# Support Vector Machine



## » Linear SVM

- Em termos geométricos pode ser visto como a procura de um hiperplano que separa um atributo **categórico binário**;
- A separação completa (**hard-margin**) só é possível se o conjunto de treino for **linearmente separável**;
- Escolhe, de entre um número infinito de hiperplanos, aquele que faz a **separação máxima** das duas classes;
- No caso de não haver separação linear pode-se usar uma **função de custo** e ainda assim se encontrar a separação incompleta máxima (**soft-margin**);
- Tipicamente existe um único **máximo global** bem definido.



# Support Vector Machine



## » Linear SVM

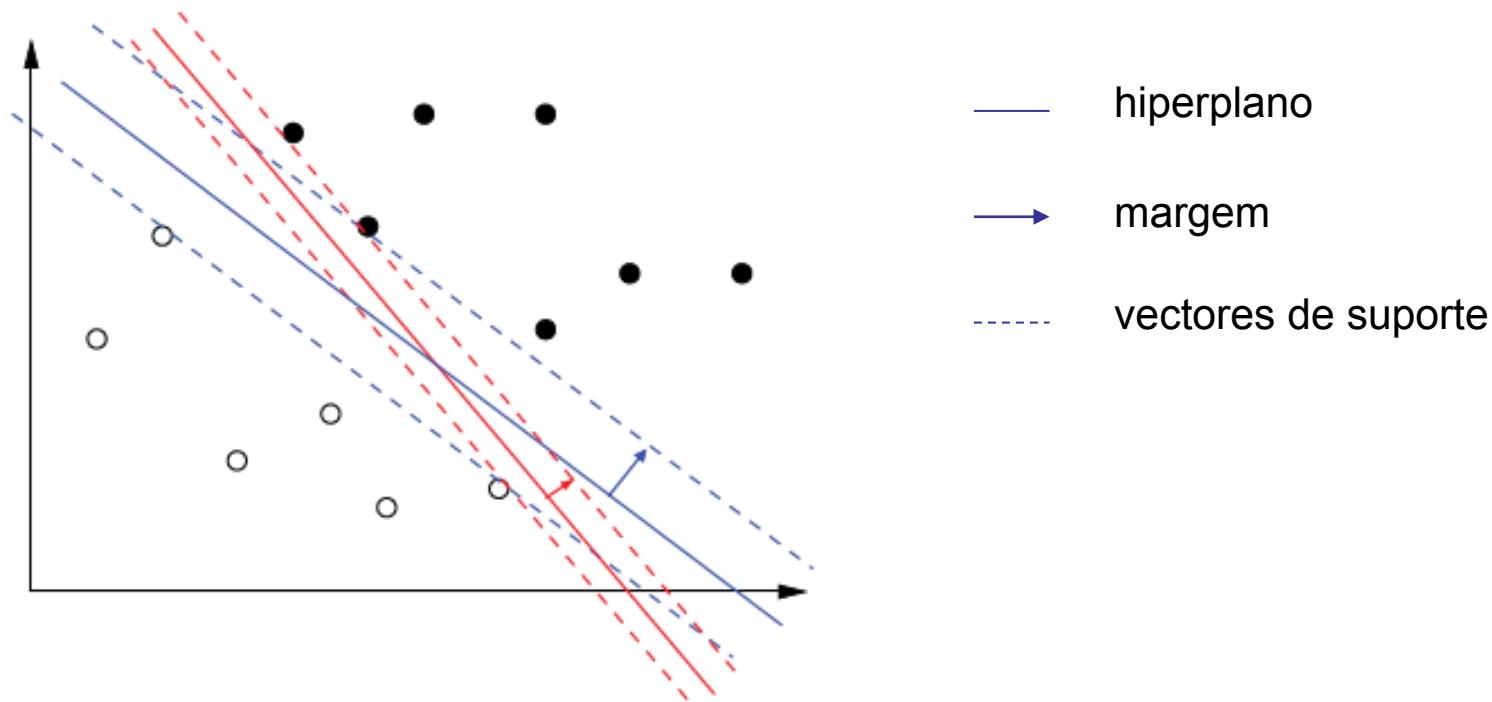
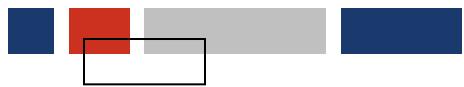


Figura 3.31 de Torga L. "Data Minig with R" (2017)

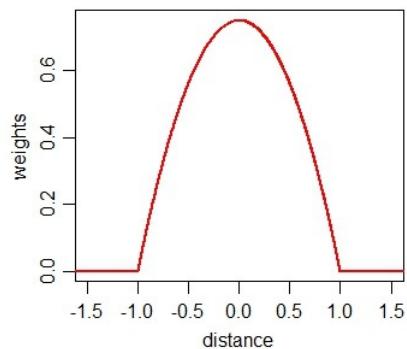


# Support Vector Machine

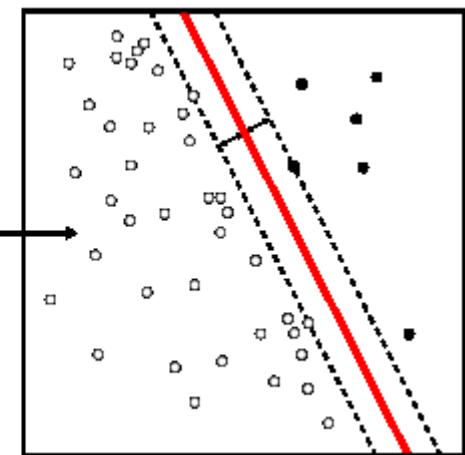
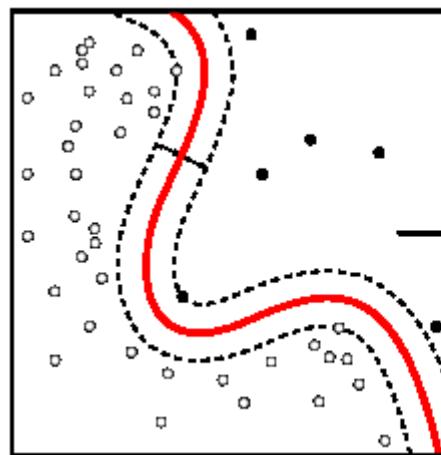


## » Nonlinear SVM

- Na maioria dos problemas reais não é possível separar as classes linearmente;
- Usa-se funções de *kernel* para definir fronteiras locais (e.g. Gaussiano, Epanechnikov, Polinomial, Radial)



epanechnikov kernel



Alisneaky (2011) "Kernel Machine.png"



# Support Vector Machine



## » SVM for Regression

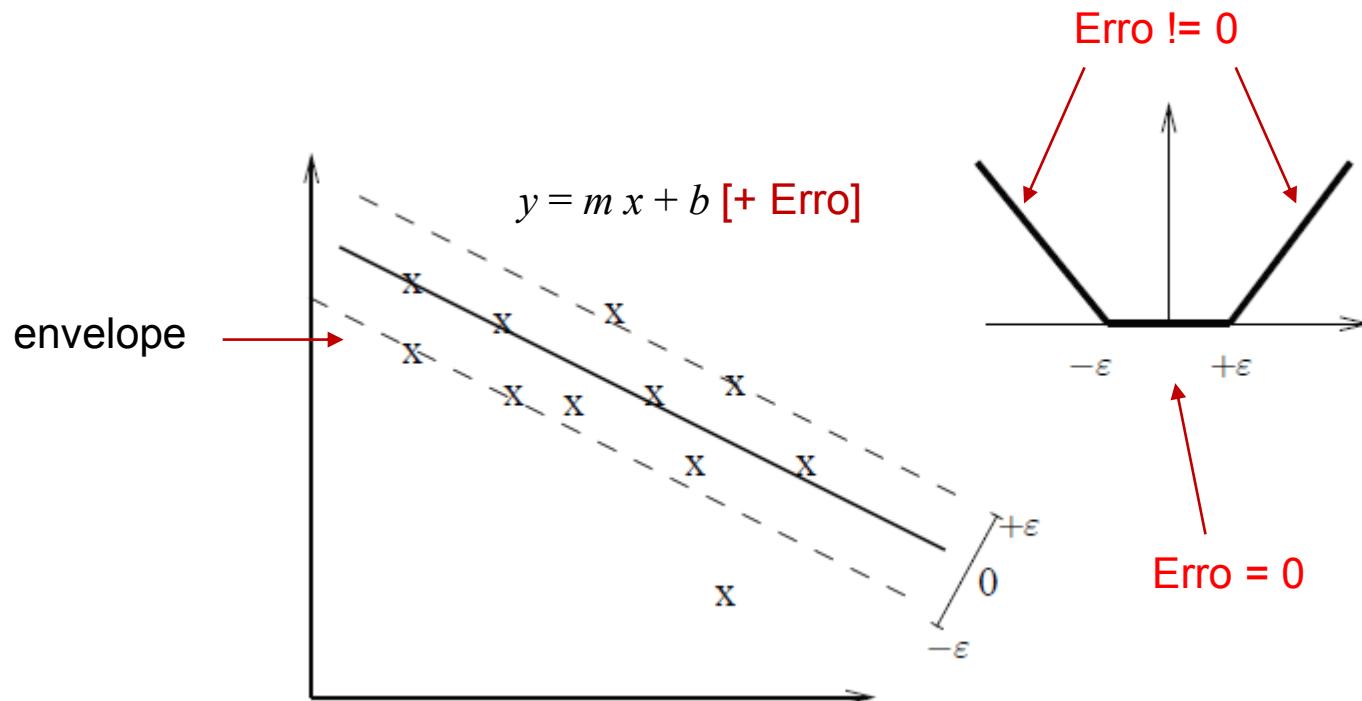


Figura 3.33 de Torga L. “Data Minig with R” (2017)



# Support Vector Machine



## » Algoritmos

### Variável-alvo categórica

e.g. *C-classification* onde o  $C$  é o custo da classificação errada (soft-margin).

### Variável-alvo contínua

e.g.  *$\epsilon$ -regression* onde  $\epsilon$  é o tamanho do “envelope” que define a “linha de regressão”.



# Support Vector Machine



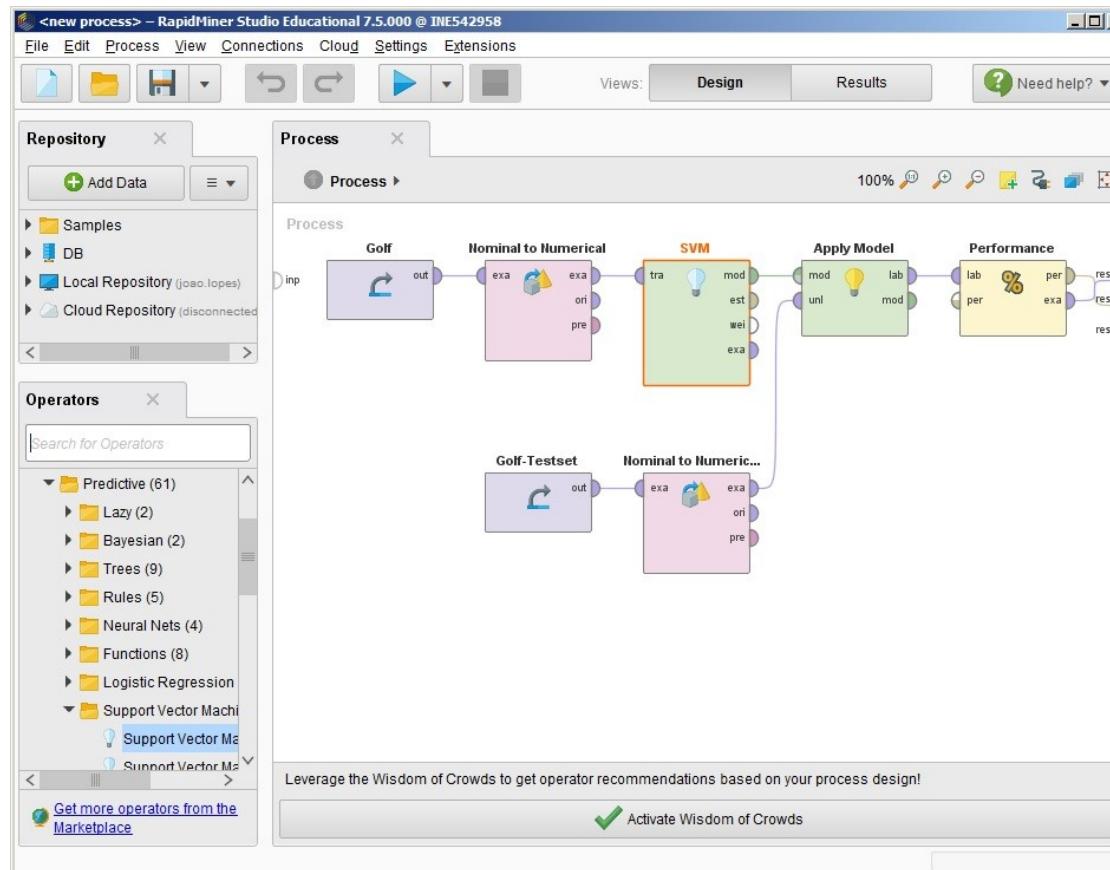
## » Vantagens e Desvantagens

- Grande capacidade de generalização.
- Ajusta modelos complexos aos dados sem necessidade da sua discriminação prévia;
- Tipicamente não tem problemas de convergência local, garantindo a procura pelo máximo global.
- Os parâmetros ajustados dos modelos são de difícil interpretação (blackbox);
- Falta de objectividade na escolha dos parâmetros da função kernell.

# Support Vector Machine



## » No RapidMiner





# Support Vector Machine



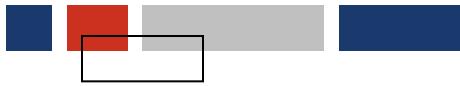
## » No R

```
library("e1071")

trainset <- read.csv("../data/golf.csv",header=TRUE)
testset <- read.csv("../data/golf-testset.csv",header=TRUE)

#Recoding dummy variables
trainset[,"Outlook"] <- factor(trainset[,"Outlook"])
trainset[,"Wind"] <- factor(trainset[,"Wind"])
tmp1 = model.matrix(~trainset[,"Outlook"]-1)
tmp2 = model.matrix(~trainset[,"Wind"]-1)
colnames(tmp1) = paste("Outlook",levels(trainset[,"Outlook"]),sep="_")
colnames(tmp2) = paste("Wind",levels(trainset[,"Wind"]),sep="_")
trainset = cbind(tmp1,tmp2,trainset[,c("Temperature","Humidity","Play")])

testset[,"Outlook"] <- factor(testset[,"Outlook"])
```



# Support Vector Machine



```
testset[, "Wind"] <- factor(testset[, "Wind"])

tmp1 = model.matrix(~testset[, "Outlook"]-1)

tmp2 = model.matrix(~testset[, "Wind"]-1)

colnames(tmp1) = paste("Outlook", levels(testset[, "Outlook"]), sep="_")
colnames(tmp2) = paste("Wind", levels(testset[, "Wind"]), sep="_")

testset = cbind(tmp1, tmp2, testset[, c("Temperature", "Humidity", "Play")])

#Perform SVM

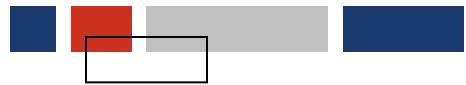
mod <- svm(Play ~ .,
            trainset,                                #model design
            scale=TRUE,                                #training set
            type="C-classification", #SVM algorithm
            kernel="radial",                           #kernel type
            gamma=1.0,                                 #kernel gamma
            cost=1.0,                                  #C-constant (cost)
            tolerance=0.001,                            #convergence epsilon
            epsilon=0.0)                               #loss-function epsilon
```



# Support Vector Machine



```
res <- predict(mod,  
                testset)                      #model  
                                         #testing set  
  
#Confusion Matrix  
conf_mat <- table(testset[,c("Play")],res)  
conf_mat  
  
#Error rate  
err_rt <- 100*(sum(conf_mat) - sum(diag(conf_mat)))/sum(conf_mat)  
err_rt
```



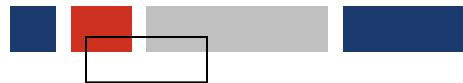
# Computação Natural



## » Enquadramento

- A computação natural inclui algoritmos que se inspiram em processos que ocorrem na **Natureza**;
- Usados quando os **métodos matemáticos tradicionais** não podem ser usados ou são demasiado “dispendiosos”;
- Algoritmos desenvolvidos para **procura e otimização** (i.e. seleção do melhor elemento de um conjunto);
- São normalmente constituídos por **heurísticas de procura** acopladas a funções de custo (**minimização**) ou de fitness (**maximização**);

e.g. **redes neurais artificiais**, **computação evolutiva** e **inteligência de enxames**.



# Computação Natural



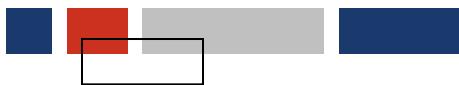
## » Computação Natural

### **Computação Evolutiva (e.g. Algoritmos Genéticos)**

Baseada na evolução natural dos **sistemas biológicos**, em que uma **população de agentes** em evolução (mutação) é selecionada (**seleção**) e troca informação entre si (**crossover**).

### **Inteligência de Enxames (e.g. *Particle Swarm Optimization*)**

Baseada na **inteligência colectiva** de uma **população de agentes** que interagem a nível local. Os agentes são caracterizados pela sua **posição** e **velocidade**, e são influenciados pelo **seu conhecimento** e pelo conhecimento da sua **vizinhança**.



# Algoritmos Genéticos



## » Enquadramento

- Tornados popular no início dos anos 70 por John Holland;
- Baseados na evolução natural de sistemas biológicos;
- Utilizam uma população de agentes, cada um caracterizado por uma sequência de valores dos atributos (cromossomas), que geram iterativamente novas gerações de agentes;
- Os cromossomas são avaliados por uma função de *fitness* que deve ser maximizada.
- A população evolui ao longo do tempo através de mecanismos evolutivos ao selecionar os “melhores” agentes para as gerações seguintes (seleção), por troca de informação entre agentes (crossover) e por transferência imperfeita de informação (mutação).



# Algoritmos Genéticos



## » Elementos dos Algoritmos Genéticos

- Cada **população** é formada por vários agentes ou **indivíduos**, e esses indivíduos são representados por **cromossomas**;
- A população de indivíduos gera novas populações ao longo do tempo (**gerações**) caracterizadas por diferentes conjuntos de cromossomas;
- Os cromossomas são uma possível solução para o problema (i.e. **hipótese**) e são constituídos por um conjunto de valores dos atributos (**genes**);
- A **codificação binária** dos atributos é comum representando presença ou ausência do atributo (mas existem outras alternativas);
- Os **mecanismos evolutivos** actuam sobre os cromossomas.



# Algoritmos Genéticos



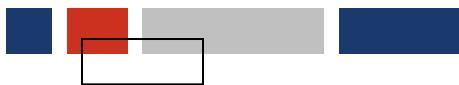
## » Mecanismos evolutivos

**Seleção** - os cromossomas com maior fitness são passados para a geração seguinte. Esta seleção pode ser **determinística** (passam os melhores) ou **estocástica** (existe uma probabilidade associada).

**Crossover** - os cromossomas “misturam-se”, trocando valores dos atributos entre si. O crossover pode ser feito por divisões de **um ponto** ou por **múltiplos pontos**;

**Mutação** - a informação dos cromossomas é alterada. Esta alteração pode ser imposta a **subconjuntos de atributos** ou a **um só atributo**;

**Outros mecanismos (e.g. migração)** - pode haver fenómenos de migração de **cromossomas novos** na população.



# Algoritmos Genéticos



## » Algoritmo

- Cada indivíduo é caracterizada por:

$x_i$ , vector de valores dos atributos (cromossoma)

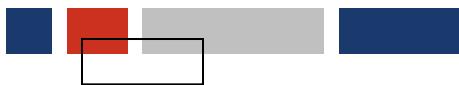
- Parâmetros do modelo evolutivo:

$m$ , taxa de mutação

$r$ , taxa de crossover

**Outros parâmetros:**

$f(x)$ , função *fitness*

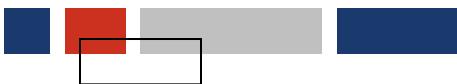


# Algoritmos Genéticos



## » Algoritmo

```
Gerar uma população inicial de indivíduos;  
WHILE critério de paragem não é atingido DO  
    Avaliar fitness de cada indivíduo;  
    Selecionar pais através da sua fitness;  
    Aplicar crossover entre pais;  
    Aplicar mutações aos pais;  
END
```



# Algoritmos Genéticos



## » Exemplo\*

- Uma pessoa vai passar um mês na selva;
- Leva consigo uma mochila com capacidade para 20Kg;
- Pode levar um conjunto de itens correspondendo a determinados **pontos de sobrevivência** e a um peso;
- Qual a melhor solução?

ITEM	SURVIVAL POINTS	WEIGHT
pocketknife	10.00	1.00
beans	20.00	5.00
potatoes	15.00	10.00
unions	2.00	1.00
sleeping bag	30.00	7.00
rope	10.00	5.00
compass	30.00	1.00

\*Knapsack problem tem mais de um século e é um problema de otimização combinatória. Adaptado de Marek Obitko. <http://www.r-bloggers.com/genetic-algorithms-a-simple-r-example/>



# Algoritmos Genéticos



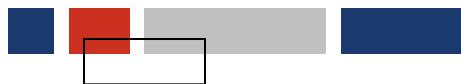
## » Exemplo\*

- O número possível de instâncias é  $2^7=128$ ;
- Maximizar pontos de sobrevivência, mas mantendo peso < 20Kg.

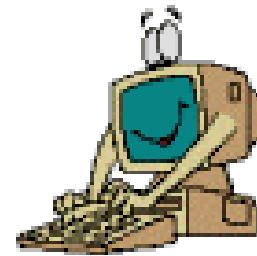
Exemplos de instâncias:

pocket-knife	beans	potatoes	unions	sleeping bag	rope	compass	Fitness
0	1	0	1	1	0	0	52
1	0	0	0	1	1	1	80
0	0	1	0	1	1	0	0

\*Knapsack problem tem mais de um século e é um problema de otimização combinatória. Adaptado de Marek Obitko. <http://www.r-bloggers.com/genetic-algorithms-a-simple-r-example/>



# Exercício



## » Exercícios 5.1 e 5.2



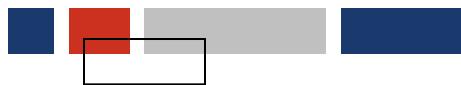
# Algoritmos Genéticos



## » Vantagens e Desvantagens

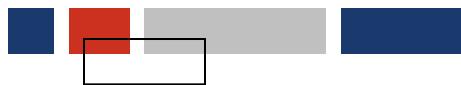
- Algoritmo é facilmente interpretável;
  - Processo de optimização robusto a ruído e a máximos locais;
  - Facilmente paralelizável, reduzindo grandemente o tempo de computação.
- 
- Alguma dificuldade em selecionar os parâmetros dos modelos (mutação, seleção, crossover) e critérios de paragem;
  - A construção da função de fitness requer algum cuidado;
  - Podem-se tornar computacionalmente intensas com números elevados de atributos (i.e. *curse of dimensionality*).





## » Enquadramento

- Proposto em 1995 por Everhart e Kennedy;
- Baseado no **comportamento de grupo de animais** (e.g. bandos de aves, cardumes de peixes);
- Utiliza **uma populações de agentes** (tal como os AGs) para **explorar iterativamente o espaço** de atributos até atingir determinada condição de paragem;
- Cada agente (ou partícula) interage com a sua vizinhança influenciando a sua **trajectória**;
- As posições no espaço de atributos é caracterizada por uma **função de custo** que deve ser minimizada.



## » Algoritmo

- Cada partícula  $i$  é caracterizada por:

$x_i$ , posição actual

$p_i$ , melhor posição histórica

$I_i$ , melhor posição histórica da vizinhança

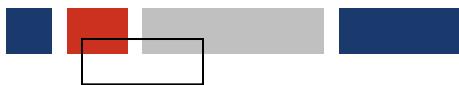
$v_i$ , velocidade

- Outros parâmetros:

$w(t)$ , inércia dinâmica (facilita procura)

$\varphi_1 \varphi_2$ , coeficientes de aceleração (impõe o peso dos componentes)

$f(x)$ , função custo



# » Algoritmo

## Atualização da velocidade:

$$v_{t+1} = w_t v_t + \varphi_1 \cdot (p_t - x_t) + \varphi_2 \cdot (I_t - x_t)$$

↑

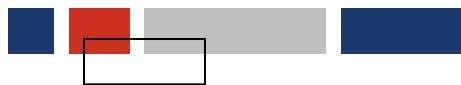
componente cognitiva

inércia

componente social

## Atualização da velocidade:

$$x_{t+1} = x_t + v_t$$



## » Algoritmo

**Gerar uma população inicial de partículas;**

**Gerar uma velocidade inicial das partículas;**

**WHILE** critério de paragem não é atingido **DO**

**FOR** cada partícula  $i$  **DO**

**IF**  $f(x_i) < f(p_i)$  **THEN**  $p_i = x_i$

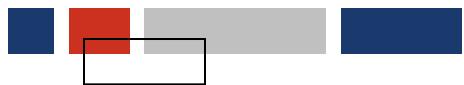
$I_i = \min(p_{vizinhança})$

        Actualização\_da\_velocidade( )

        Actualização\_da\_posição( )

**END**

**END**



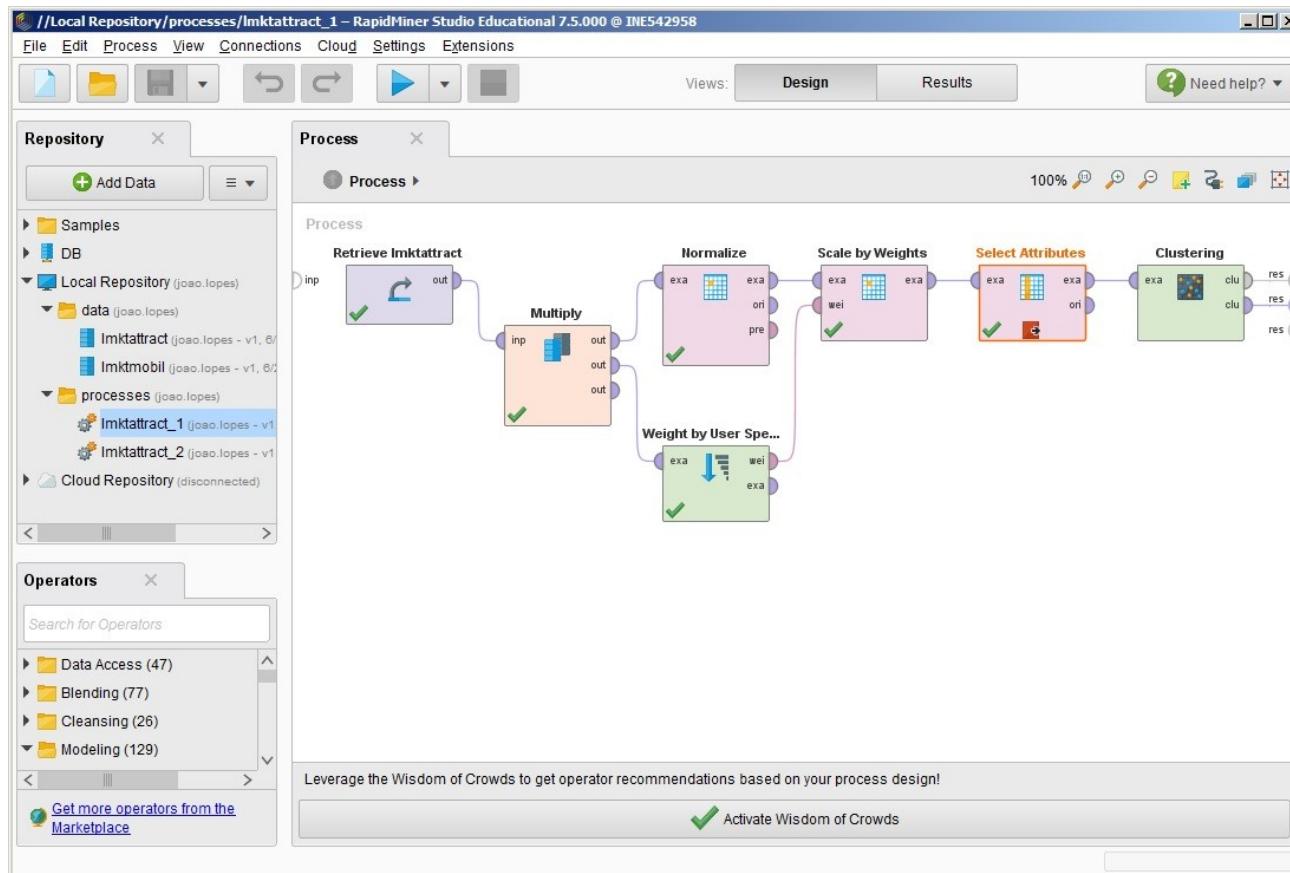
## » Vantagens e Desvantagens

- Algoritmo é tendencialmente mais eficiente que os AG;
- Processo de optimização robusto a ruído e a máximos locais.
- Alguma dificuldade em selecionar os parâmetros dos modelos (que influenciam posição e velocidade) e critérios de paragem;
- A construção da função de custo requer algum cuidado;
- Podem-se tornar computacionalmente intensas com números elevados de atributos (i.e. *curse of dimensionality*).

# Data Mining: Exemplo 1



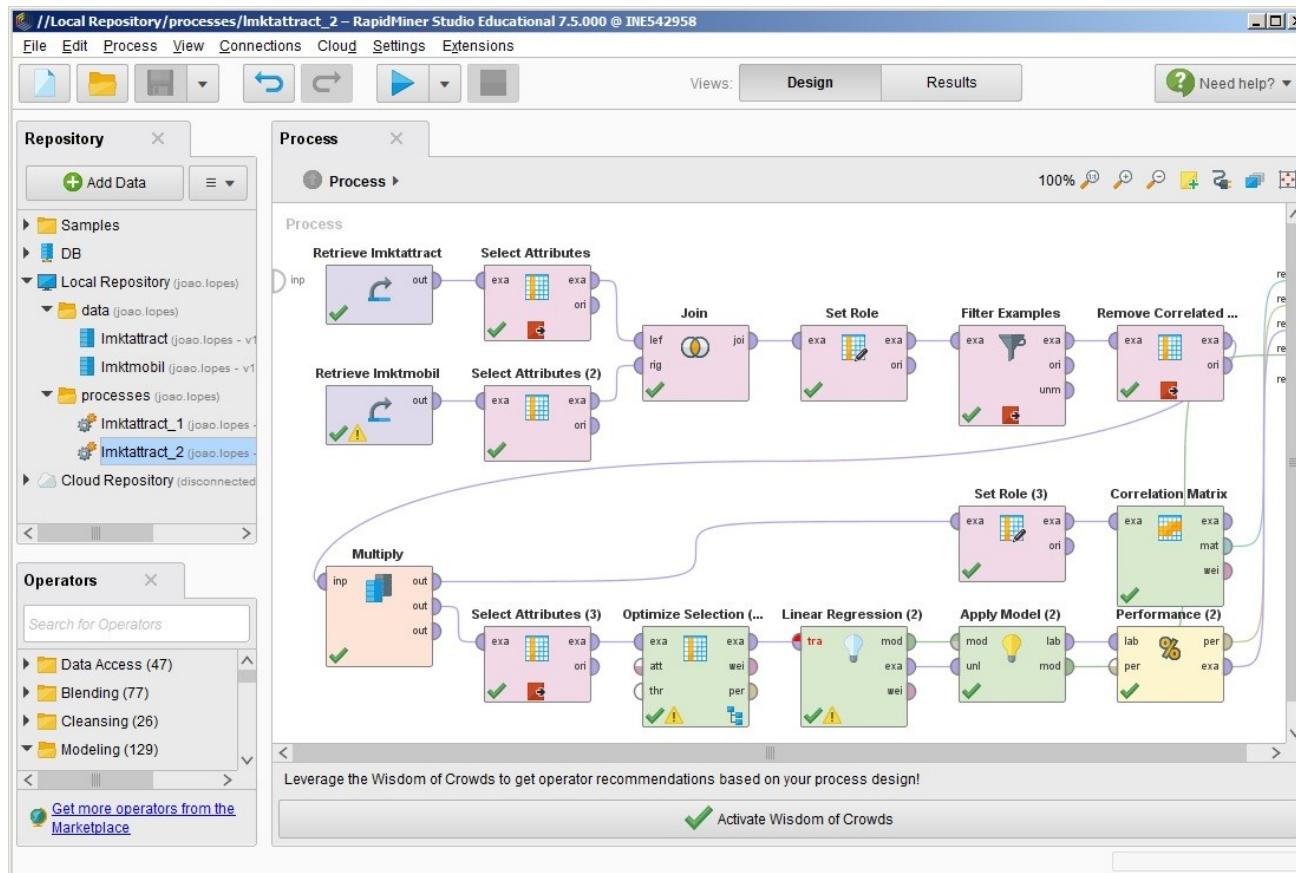
## » No RapidMiner



# Data Mining: Exemplo 2



## » No RapidMiner



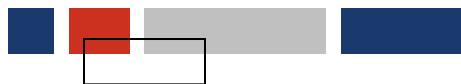


# Data Mining: Exemplos



## » No R

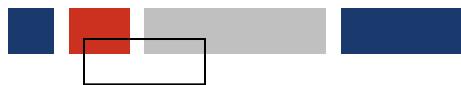
[ver ficheiros “bin/lmktattract\_1.r” e “bin/lmktattract\_2.r”]



# Bibliografia



- » Azzalini, A. E Scarpa, B., (2012). Data Analysis and Data Mining – Na Introduction. Oxford University Press, New York.
- » Bramer, M. (2007). Principles of Data Mining. Springer. New York.
- » Brazdil, P. (2009,a). Unsupervised Discretization in R. LIAAD – INESC Porto L.A./FEP.
- » Brazdil, P. (2009,b). Supervised Discretization in R. LIAAD – INESC Porto L.A./FEP.
- » Due, KL e Swamy M.N.S., (2016). Search and Optimization by Metaheuristics - Techniques and Algorithms Inspired by Nature. Springer, Switzerland.
- » Dumitrescu, D., B. Lazzerini, L. C. Jain, and A. Dumitrescu (2000), Evolutionary Computation, USA, CRC Press
- » Gama, J., Carvalho, A.P.L., Faceli, K., Lorena, A.C. e Oliveira, M., (2012). Extração de Conhecimento de Dados. *Data Mining*. Edições Sílabo. Lisboa.



# Bibliografia



- » Han, J. e Kamber, M. (2006). *The Data Mining: Concepts and Techniques*. Second Edition, Morgan Kaufmann Publishers. San Francisco.
- » Larose, D.T. e Larose C.D., (2014). *Discovering Knowledge in Data – An Introduction to Data Mining*. John Wiley & Sons, New Jersey.
- » Mihaescu, C. (2011). *Naive-Bayes Classification Algorithm*. Universitatea din Craiova, Departamentul de Inginerie Software.
- » Mitchell, T., (1997), *Machine Learning*, McGraw-Hill
- » Tan, P.-N., Steinbach, M. e Kumar, V. (2006). *Introduction to Data Mining*. Pearson Addison-Wesley.
- » Torgo, L., (2017). *Data Mining with R – Learning with Case Studies*. Taylor & Francis Group, New York.
- » Witten, I., Frank, E., 2000, *Data Mining: practical machine learning tools and techniques with Java implementations*, Morgan and Kaufman