

» *Data Analytics com Aplicações*

Pedro Campos
João Lopes

DMSI / ME



Dezembro de 2019



- Introdução
- Preparação dos Dados
- Métodos de avaliação
- Modelos Exploratórios
- **Modelos Preditivos**
- **Computação Natural**
- **Exemplos de *Data Analytics***





- Modelos Preditivos
- k-NN
- *Naive Bayes*
- Árvore de Decisão
- Regras de Decisão
- Redes Neuronais
- Support Vector Machine
- Computação Natural
- Exemplos *Data Analytics*

(0h30)]	
(1h30)]	dia1 am
(1h00)]	
(2h00)]	dia1 pm
(1h00)]	
(2h00)]	dia2 am
(1h00)]	
(2h00)]	dia2 pm
(1h00)]	





Modelos

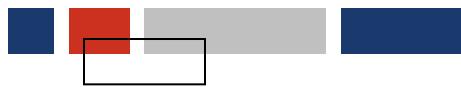


» Modelos exploratórios

- modelos **não-supervisionados** (não têm variável-resposta);
- **associações e agrupamentos.**

» Modelos preditivos

- modelos **supervisionados** (têm variável-resposta);
- relações entre variáveis que permitem fazer **previsões.**



Modelos



» Modelos exploratórios

- Regras de Associação
- Agrupamento (e.g. Hierárquico, K-Means)

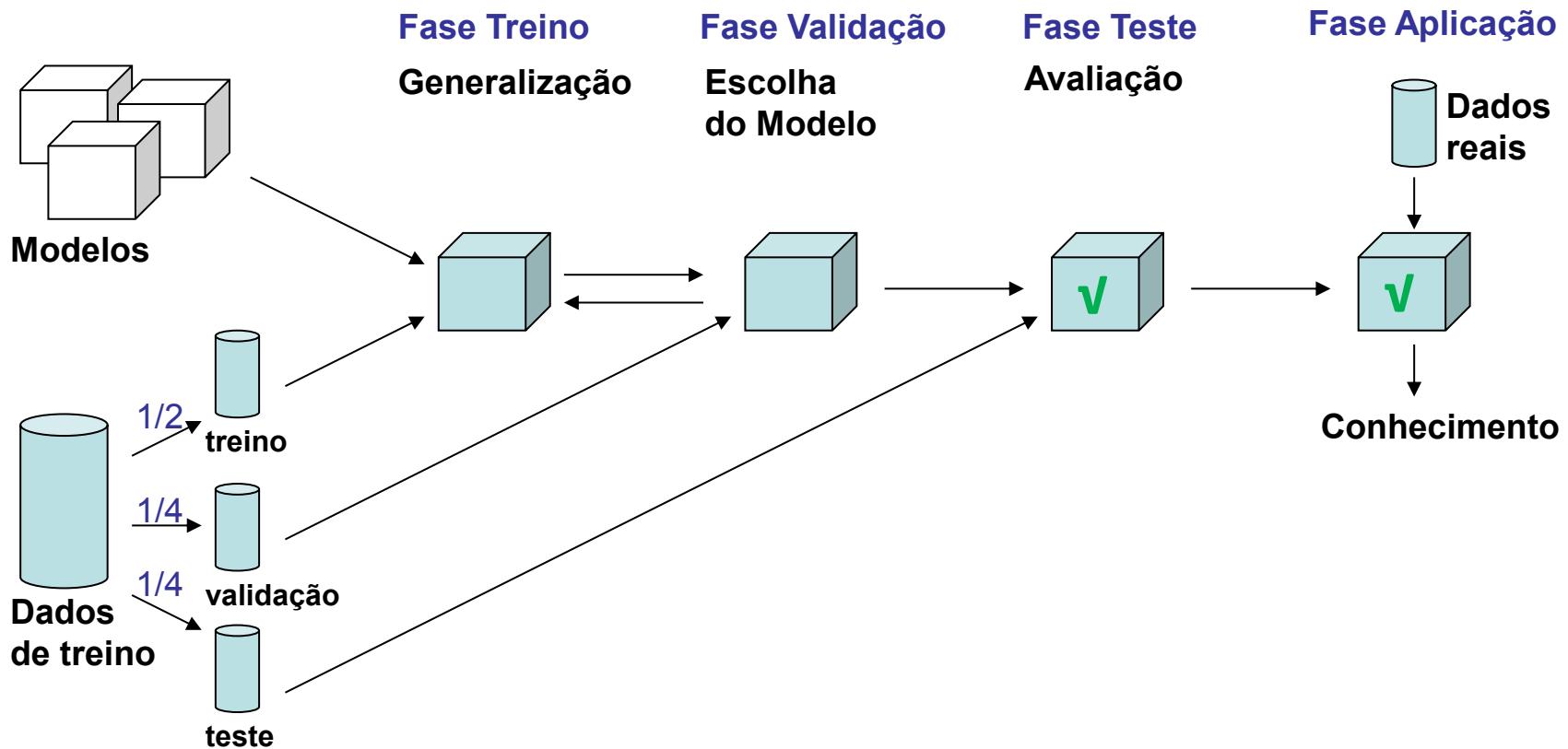
» Modelos preditivos

- Distâncias (e.g. *k-Nearest Neighbour*)
- Probabilísticos (e.g. Redes Bayesianas)
- Procura (e.g. Árvores de Decisão e Regras de Decisão)
- Otimização (e.g. Redes Neuronais, *Support Vector Machine*)

Modelos preditivos



» Visão Geral



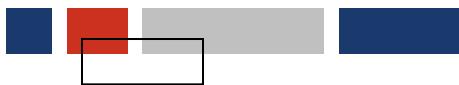


Modelos preditivos



» Conceitos

- variável-resposta e variáveis-preditivas;
- 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

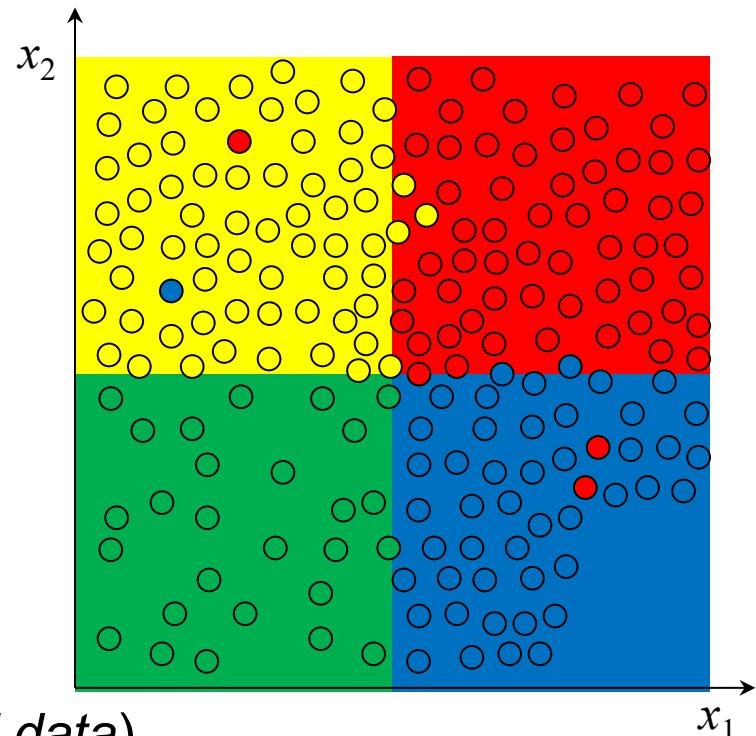


» Conceitos: conjunto de treino

e.g.

- variável-resposta (qualitativa)
- 2 variáveis-preditivas (x_1 e x_2)

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

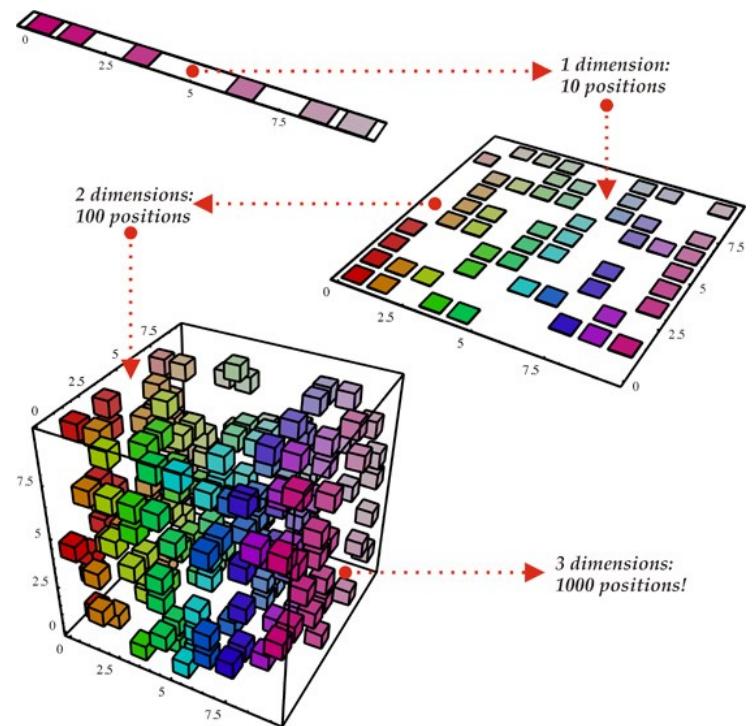


Modelos preditivos

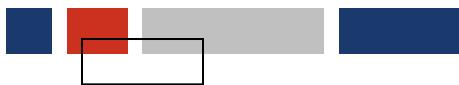


» Conceitos: *curse of dimensionality*

+ variáveis => + informação
=> + espaço a explorar



Y. Bengio (2008) "CurseDimensionality.jpg"

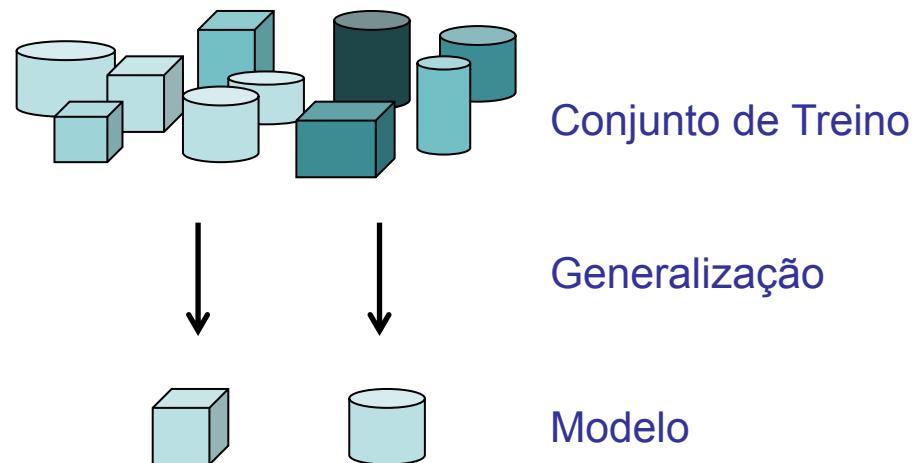


Modelos preditivos



» Conceitos: generalização

- *lazy learner*: previsões usam conjunto de treino;
- *eager learner*: previsões usam generalizações.

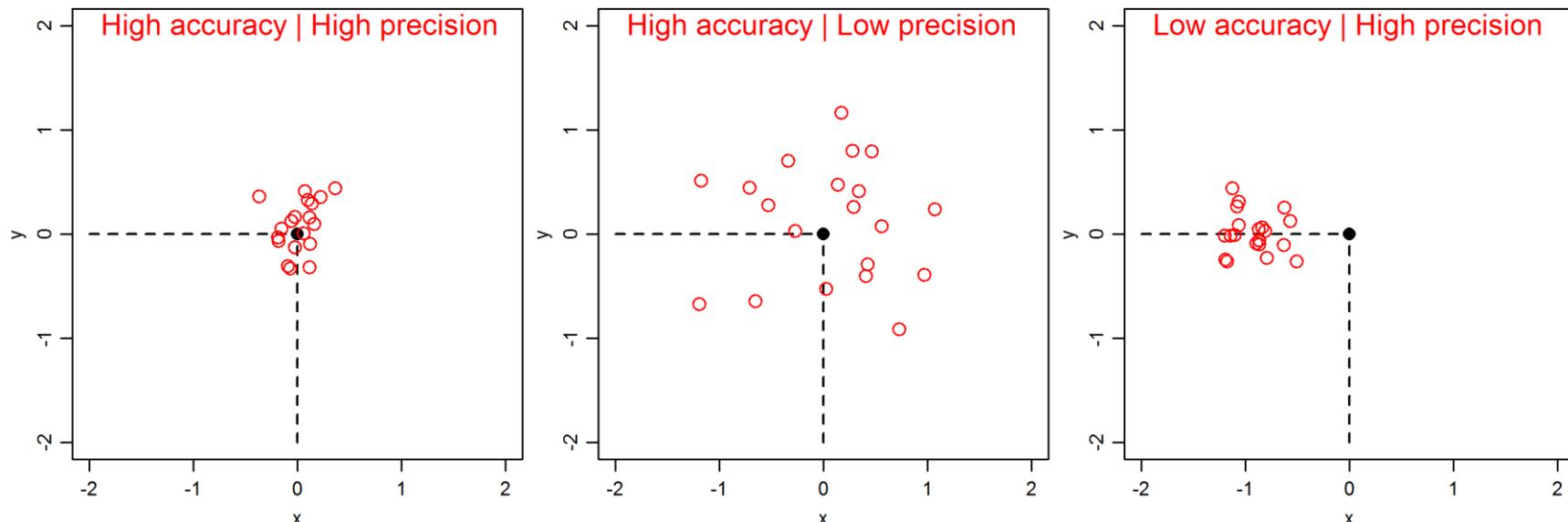




Modelos preditivos



» Conceitos: precisão e exactidão



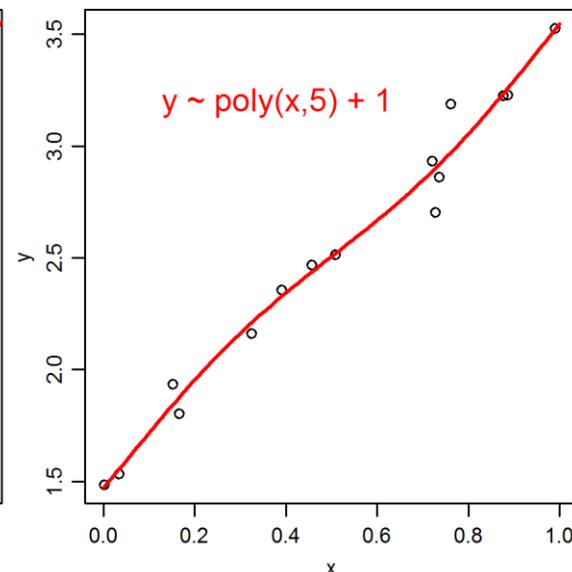
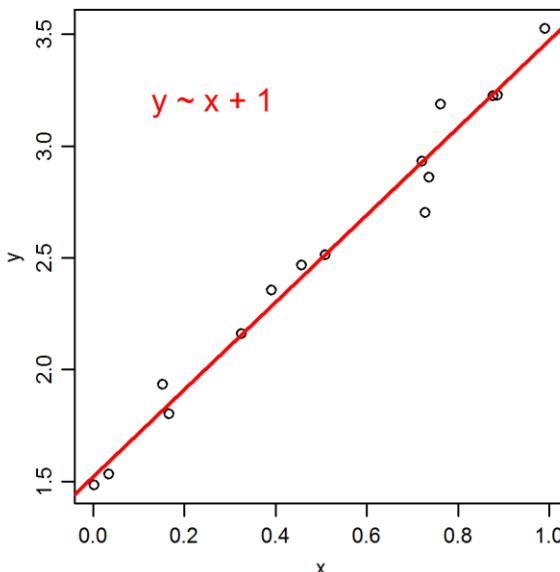
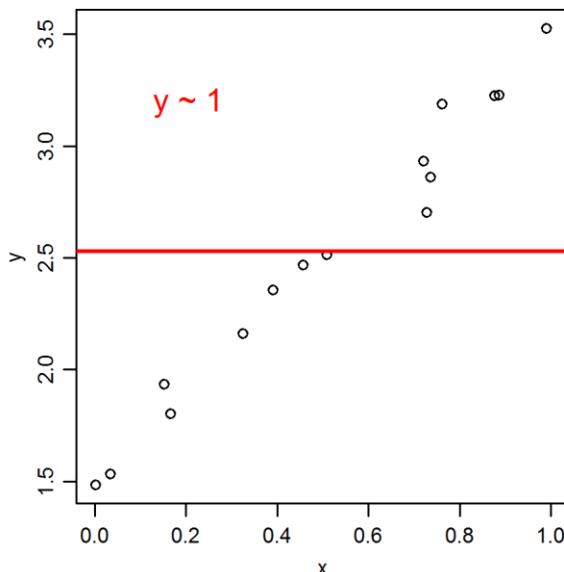
- Os conceitos **precisão** e **exactidão** não são antagónicos;
- Mas é comum que para obter uma maior **precisão** se tenha que “sacrificar” a **exactidão** (e.g. k-NN, Árvores de decisão).



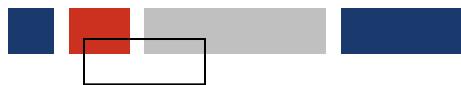
Modelos preditivos



» Conceitos: sub- e sobre-ajustamento



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

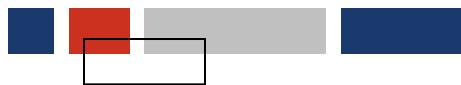


Modelos preditivos



» Conceitos: critérios de paragem

- Modelos preditivos são tipicamente iterativos;
- Cada iteração ajusta o modelo ao conjunto de treino;
- Critérios de paragem do ajustamento:
 - i) número de **iterações máximo** atingido;
 - ii) erro reduz-se abaixo de **limiar de erro**;
 - iii) modelo **não é alterado**.

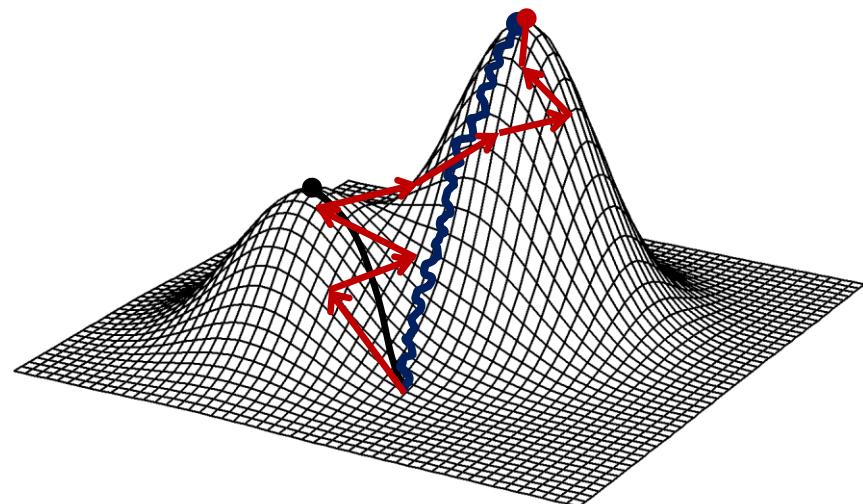


Modelos preditivos

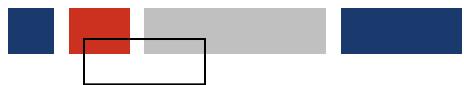


» Conceitos: máximos locais e globais

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

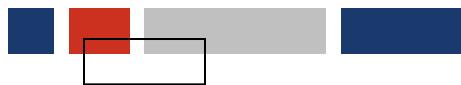


adaptado de Headlessplatter (2007) "Local_maximum.png"



» Enquadramento

- k-NN ou *k-Nearest Neighbours*, é baseado em *distâncias*, e é dos modelos mais simples;
- O exemplo a avaliar é comparado aos exemplos mais próximos do conjunto de treino;
- É considerado *lazy learner* (cf. *eager learning*), estando muito suscetível a *ruído* e a dados em *falta*;
- Pode ser usado em *classificação* ou *regressão*.



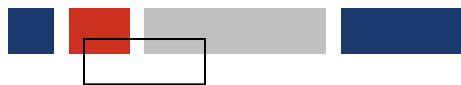
» Metodologia

- O algoritmo envolve cálculo de 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*

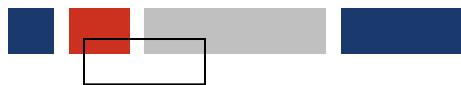
$$d(x_i, x_j) = \begin{cases} 0 & \text{sse } x_i = x_j \\ 1 & \text{sse } x_i \neq x_j \end{cases}$$



» Algoritmo para Classificação

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

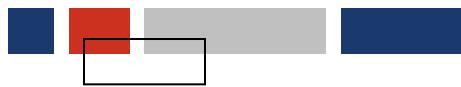
- 1) Calcular a distância desse ponto a todos os pontos do conjunto de treino.
- 2) Observar os k vizinhos mais próximos;
- 3) Conta-se o **número de ocorrência de cada classe**;
- 4) O ponto a classificar é classificado pela classe mais frequente.



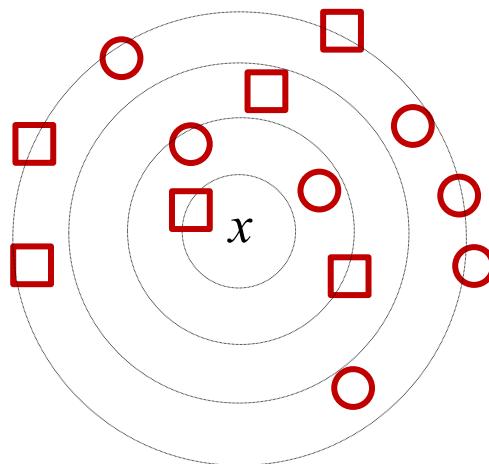
» Algoritmo para Regressão

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

- 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 previsão é dada por essa média.



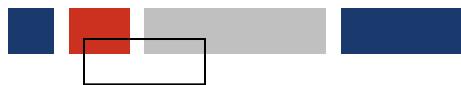
» Exemplo



Para 1 vizinho mais próximo, x é classificado como □

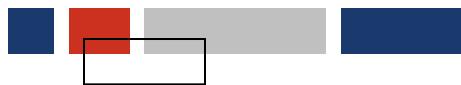
Para 3 vizinhos mais próximos, x é classificado como ○

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);



» Exercícios

1. Considere o seguinte conjunto de treino:

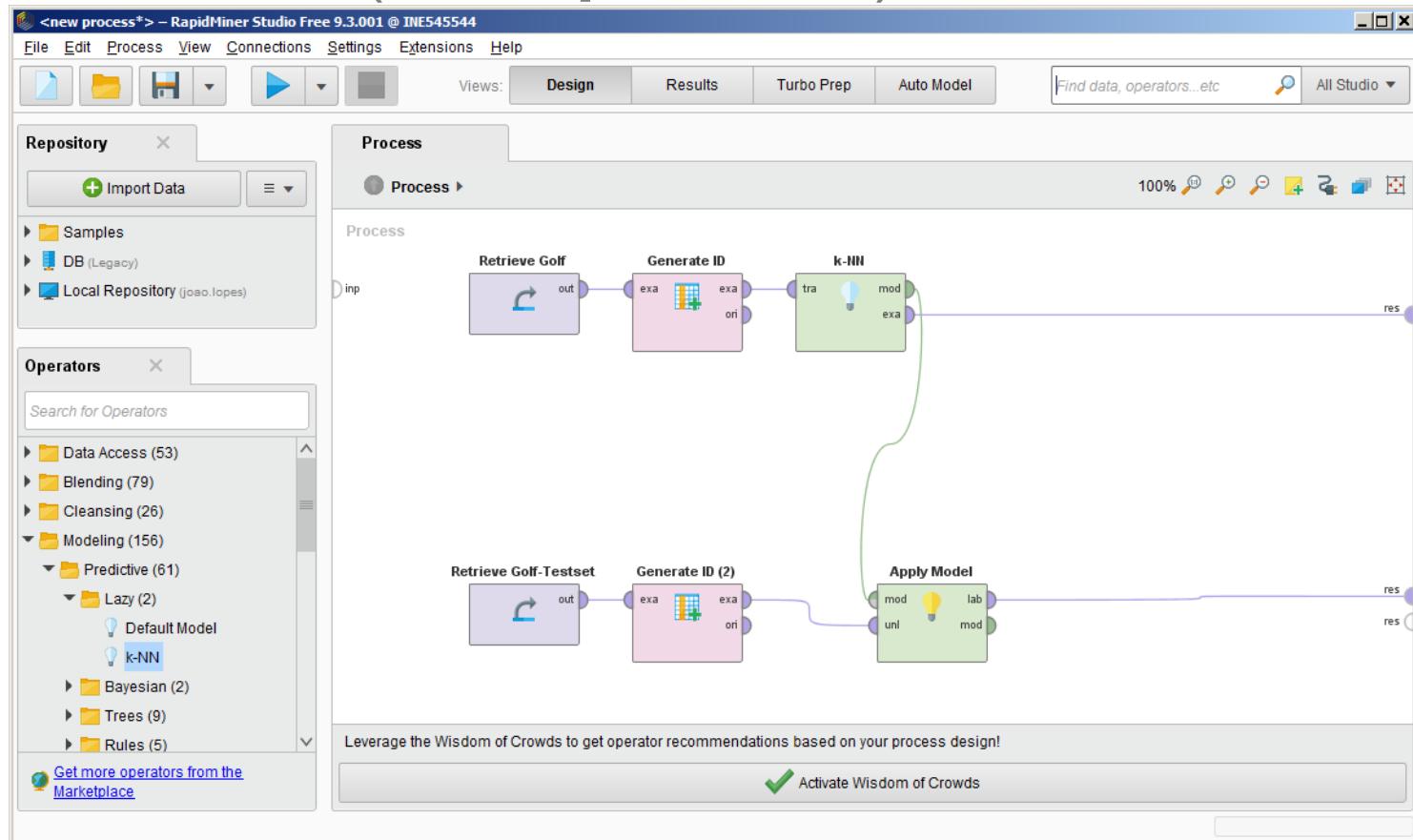
ID	a1	a2	a3	classe
1	1	1	0	X
2	0	1	1	Z
3	0	0	1	X
4	1	1	1	X
5	0	1	0	X
6	1	0	1	Z

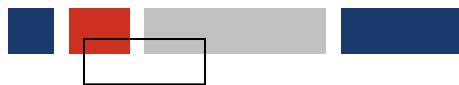
Classifique o exemplo $x = \{a_1 = 1, a_2 = 0, a_3 = 1\}$ usando:

- a) classificador 1-NN;
- b) classificador 5-NN.



» Exercícios (no RapidMiner)





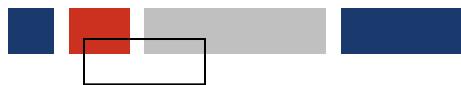
» Exercícios (no R)

```
library("class")

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

#Test model
res <- knn(trainset[,c("Temperature","Humidity")], #training set (attr.)
           testset[,c("Temperature","Humidity")], #testing set
           trainset[, "Play"],                  #training set (cl.)
           k=3)                                #k-Nearest Neighbour
           prob=TRUE)                          #proportion of winning class

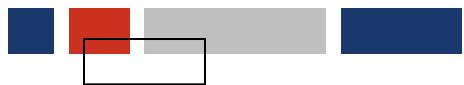
#Confusion Matrix
conf_mat <- table(testset[, "Play"], res)
conf_mat
```



k-NN



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

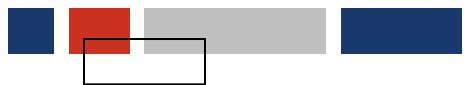


Naive Bayes



» Enquadramento

- Naive Bayes é baseado em **probabilidades**, e apesar de ter sido desenvolvido nos anos 60, mantém-se competitivo em ***Text Mining***;
- Pressupõe ingenuamente (*naive*) **independência** das variáveis-preditivas, mas é robusto a ligeiras violações;
- A classificação é tipicamente **exacta**, apesar da estimativa das probabilidades poder ter **pouca exactidão**;
- O algoritmo requer um número de parâmetros linear com o número de atributos (i.e. não sofre de ***curse of dimensionality***);
- Usado em **classificação**.



Naive Bayes



» Teorema de Bayes

- Naive Bayes é baseado no teorema de Bayes.

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

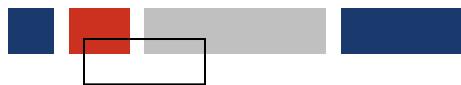
$P(\theta)$ - Prob. dos parâmetros θ (Probabilidade *a priori*);

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

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

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

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



Naive Bayes



» Metodologia

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

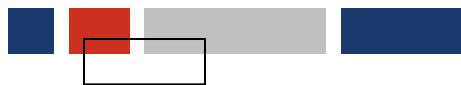
$$P(c | T) = \frac{P(c) \cdot P(T | c)}{P(T)}$$

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



» Metodologia

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

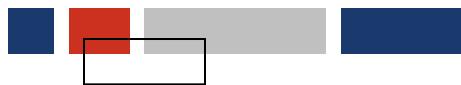
$$P(c | X_1, \dots, X_p) = \frac{P(c) \cdot P(X_1, \dots, X_p | c)}{P(X_1, \dots, X_p)}$$

- $P(X_1, \dots, X_p)$ 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

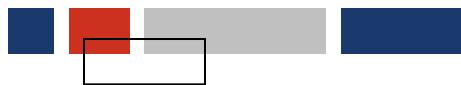


» Exemplo

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

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

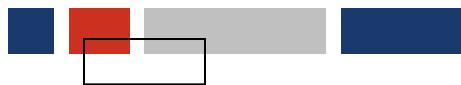


» Exemplo

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

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

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



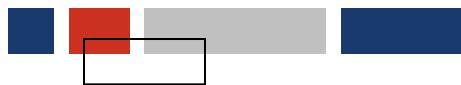
Naive Bayes



» Exemplo

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

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



Naive Bayes



» Para variáveis-preditoras contínuas

1. Assumindo **distribuição normal**, probabilidade do atributo $x = v$ é,

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

onde μ_c e σ_c são a média e o desvio padrão de x para a classe c ;

ou

2. Transformação dos valores contínuos em valores discretos.

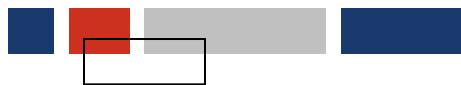


Naive Bayes



» 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).



Naive Bayes



» Exercícios

1. Considere o seguinte conjunto de treino:

ID	a1	a2	a3	classe
1	1	1	0	X
2	0	1	1	Z
3	0	0	1	X
4	1	1	1	X
5	0	1	0	X
6	1	0	1	Z

Classifique o exemplo $x = \{a_1 = 0, a_2 = 1, a_3 = 1\}$ usando Naive Bayes.



Naive Bayes



» Exercícios

2. Considere os dados do ficheiro “golf.csv”. Classifique o exemplo

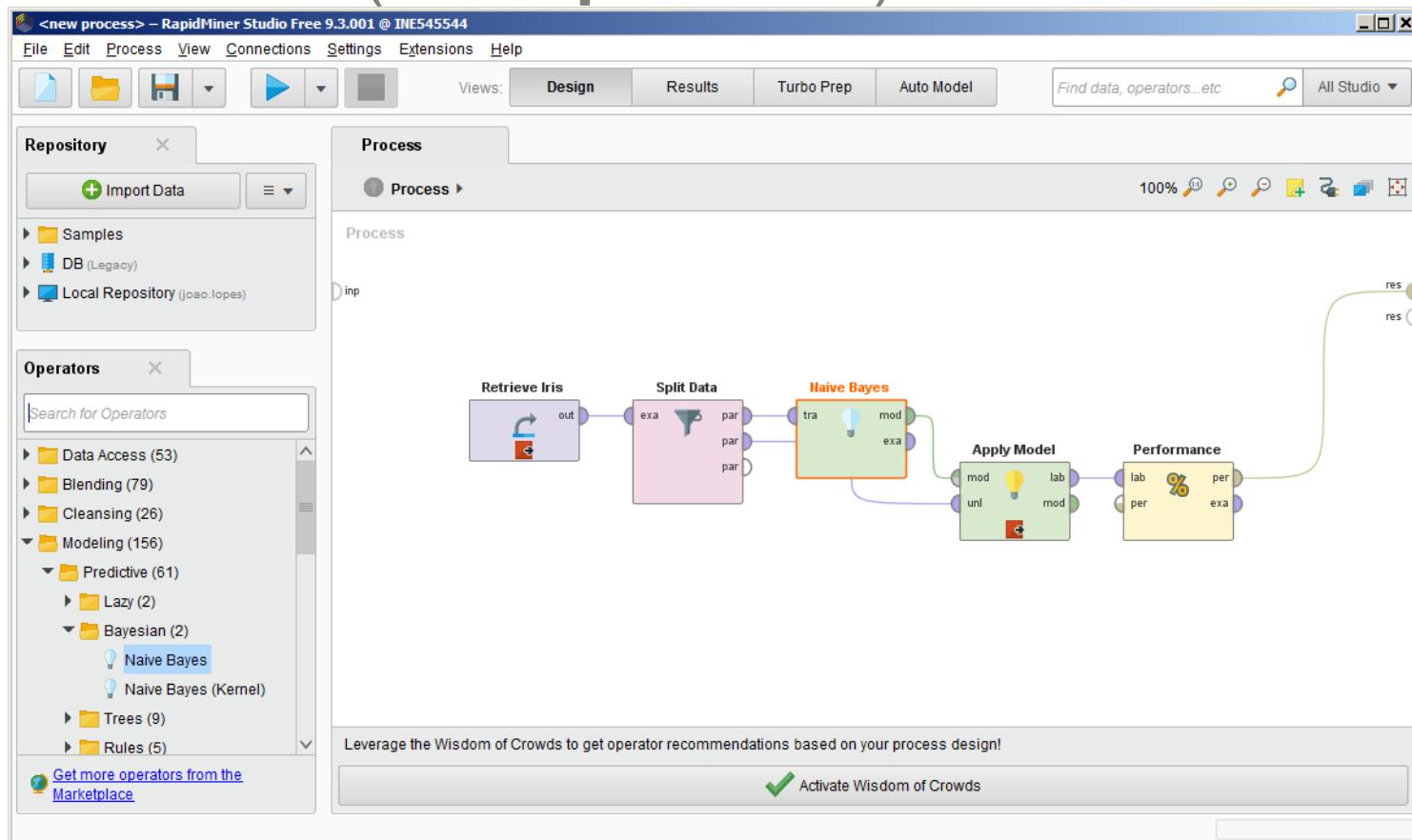
$x = \{\text{Outlook} = \text{"sunny"}, \text{Temperature} = 66, \text{Humidity} = 90, \text{Wind} = \text{TRUE}\}$

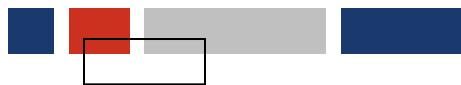
usando *Naive Bayes*.

Naive Bayes



» Exercícios (no RapidMiner)



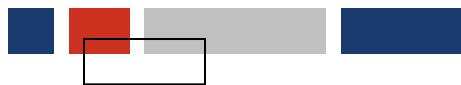


» Exercícios (no R)

```
library("e1071")

#Create testing set by hold out
set.seed(12345)
n <- nrow(iris)
n_test <- floor(n/3)
sampindex <- sample(1:n, size=n_test, replace=FALSE)
testset <- trainset[sampindex,]
trainset <- trainset[-sampindex,]

#Perform Naive Bayes fit
mod <- naiveBayes(Species ~ .,
                    trainset,           #model design
                    laplace=1)          #training set
                           #Laplace correction
```



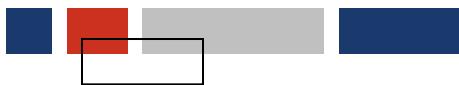
Naive Bayes



```
#Test model
res <- predict(mod,                                #model
                testset)                         #testing set

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

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

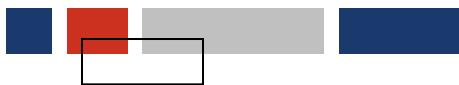


Árvores de Decisão



» Enquadramento

- Árvores de Decisão são 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** (i.e. árvores de classificação) ou **regressão** (i.e. árvores de regressão).

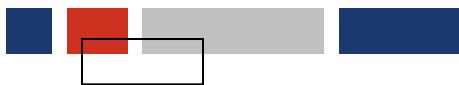


Árvores de Decisão



» Metodologia

- 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*

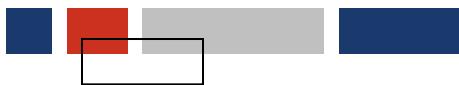
Processo de decisão na concessão de crédito bancário

Conjunto de teste: 99 exemplos, 5 atributos (incl. variável-resposta)

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. variável-resposta)

Exemplo de dados:

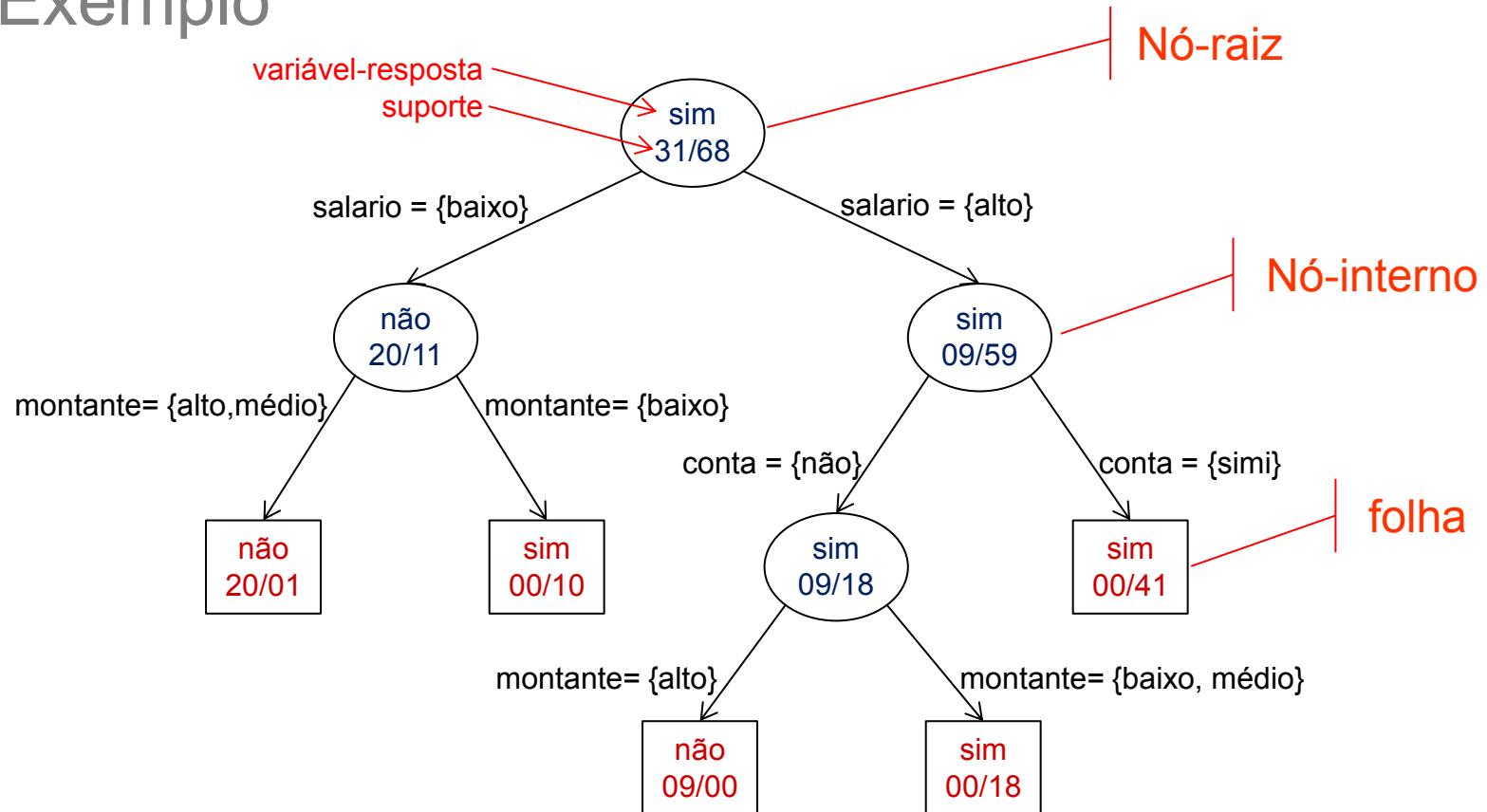
Montante	Idade	Salário	Conta	Decisão
médio	júnior	médio	sim	concedido
médio	júnior	baixo	não	não concedido
alto	sénior	alto	sim	concedido
baixo	júnior	médio	sim	concedido
alto	sénior	médio	não	não concedido
...				

*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 para atributos binários

- 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 para atributos binários

- Como escolher o nó-raiz (e os nós-internos subsequentes)?
Escolher aleatoriamente (e.g. **ordem alfabética**).
- Como dividir e (subdividir) o conjunto de teste?
Para **atributos binários** a divisão nos seus dois valores é natural.
- Quando parar de acrescentar nós-internos e terminar numa folha?
Criar folhas quando obtiver **folhas “puras”**.
- Que valor dar a cada folha?
O valor da folha “pura” é o **valor único** da variável-alvo.





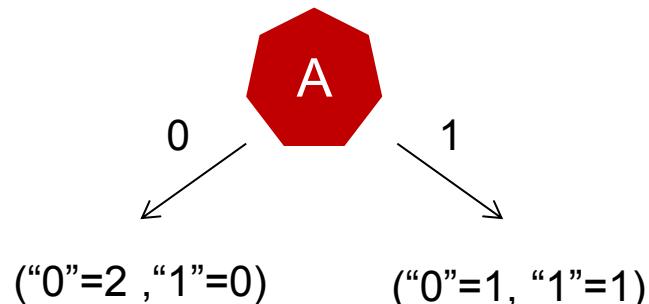
Árvores de Decisão



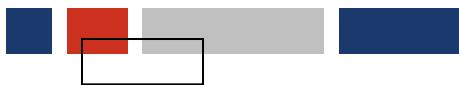
» Exemplo*

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



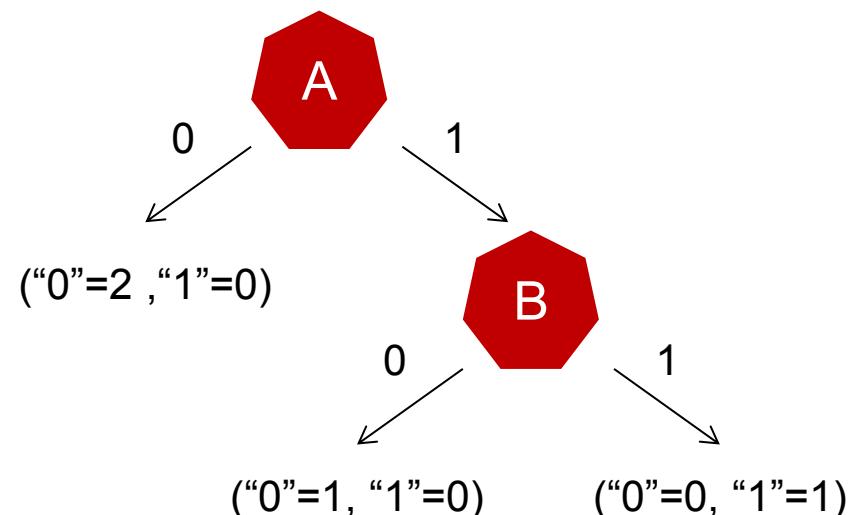
Árvores de Decisão



» Exemplo*

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



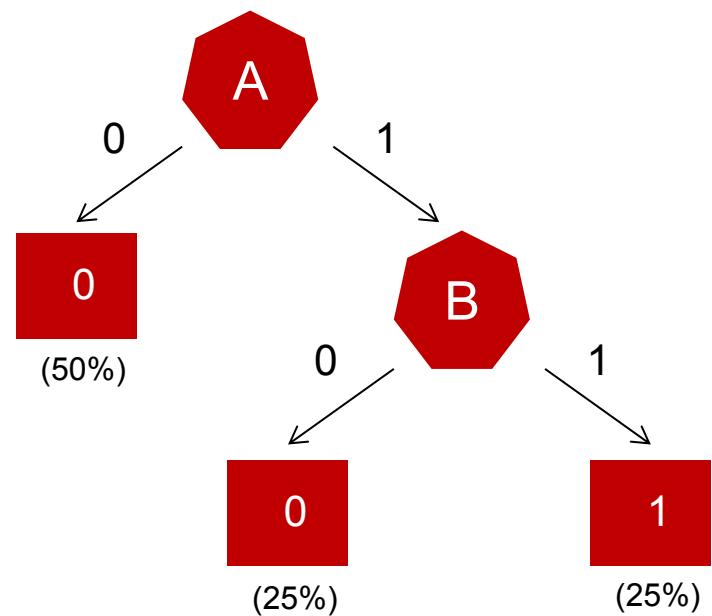
Árvores de Decisão



» Exemplo*

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



» Algoritmo “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



» Algoritmo “Particionamento recursivo”

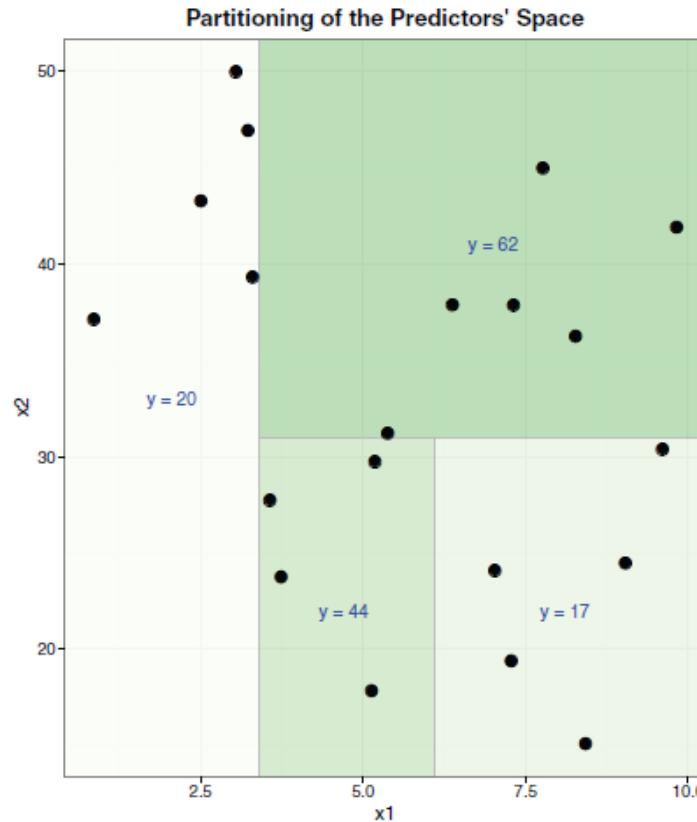
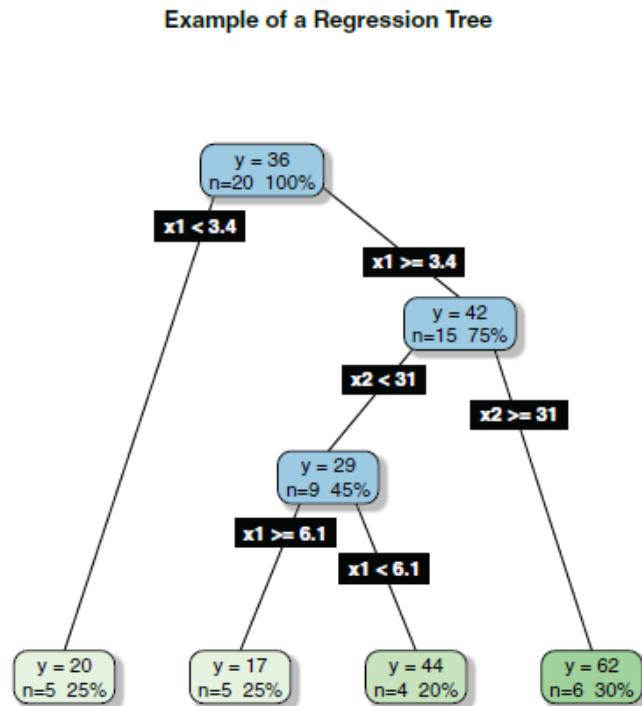
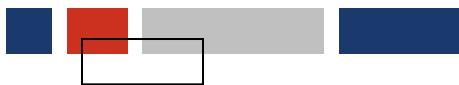


Figura 3.27 de Torga L. “Data Minig with R” (2017)



Árvores de Decisão



» Algoritmo “Particionamento recursivo”

Métrica C4.5

métrica *information gain*

divisões multiplas

Métrica 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



» 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.



Árvores de Decisão



» Exercícios

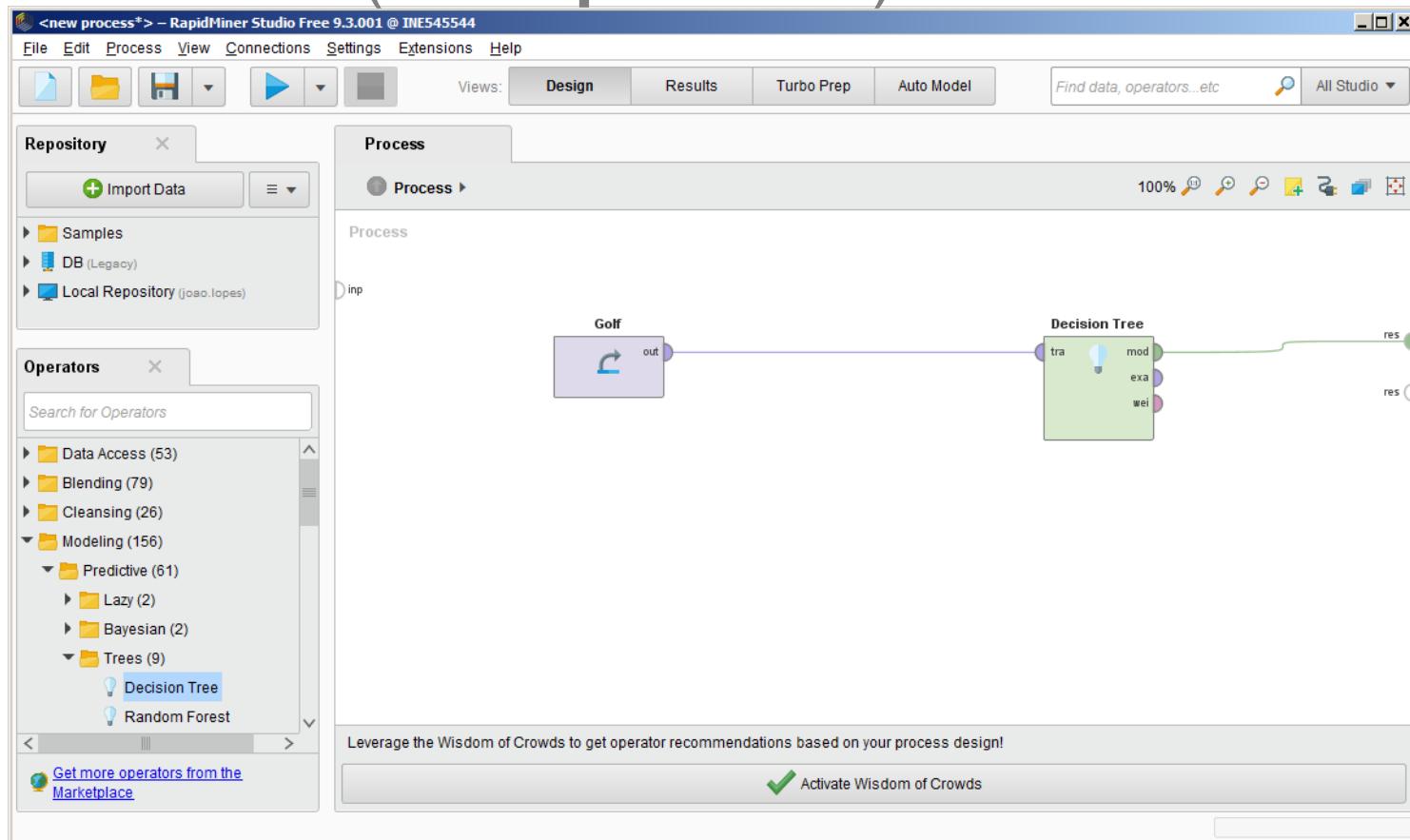
1. Faça o diagrama em árvore da disjunção (i.e. OR).

A	B	V
0	0	0
0	1	1
1	0	1
1	1	1

Árvores de Decisão



» Exercícios (no RapidMiner)



Árvores de Decisão

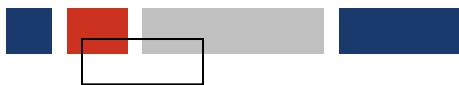


» Exercícios (no RapidMiner)

The screenshot shows the RapidMiner Studio Free interface version 9.3.001. The main window is titled '<new process> – RapidMiner Studio Free 9.3.001 @ INE545544'. The top menu bar includes File, Edit, Process, View, Connections, Settings, Extensions, and Help. The toolbar contains icons for New, Open, Save, Run, and Stop. The Views tab is set to 'Design'. The interface is divided into several panels:

- Tutorials**: A panel on the left showing a 'Modeling' tutorial titled '1/5 Let's do some data science.' It describes importing a dataset from a local repository and running a process to find survivors. Buttons for 'View All' and navigation arrows ('Back' and 'Next') are present.
- Repository**: A panel showing 'Samples', 'DB (Legacy)', and 'Local Repository (joao.lopes)'.
- Operators**: A panel listing categories: Data Access (53), Blending (79), Cleansing (26), Modeling (156), Scoring (12), Validation (29), and Utility (86). The 'Utility' category is currently selected.
- Process**: The central workspace where a process is being built. The process flow is: 'Retrieve Titanic Training' → 'Decision Tree'. The status bar indicates '100%' completion.
- Bottom Panel**: A message encourages using the 'Wisdom of Crowds' feature to get operator recommendations. A button labeled 'Activate Wisdom of Crowds' is shown.





Árvores de Decisão



» Exercícios (no R)

```
library("rpart")

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

#Perform Decision Trees fit
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



```
#Plot model
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)

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

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

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



Regras de Decisão



» Enquadramento

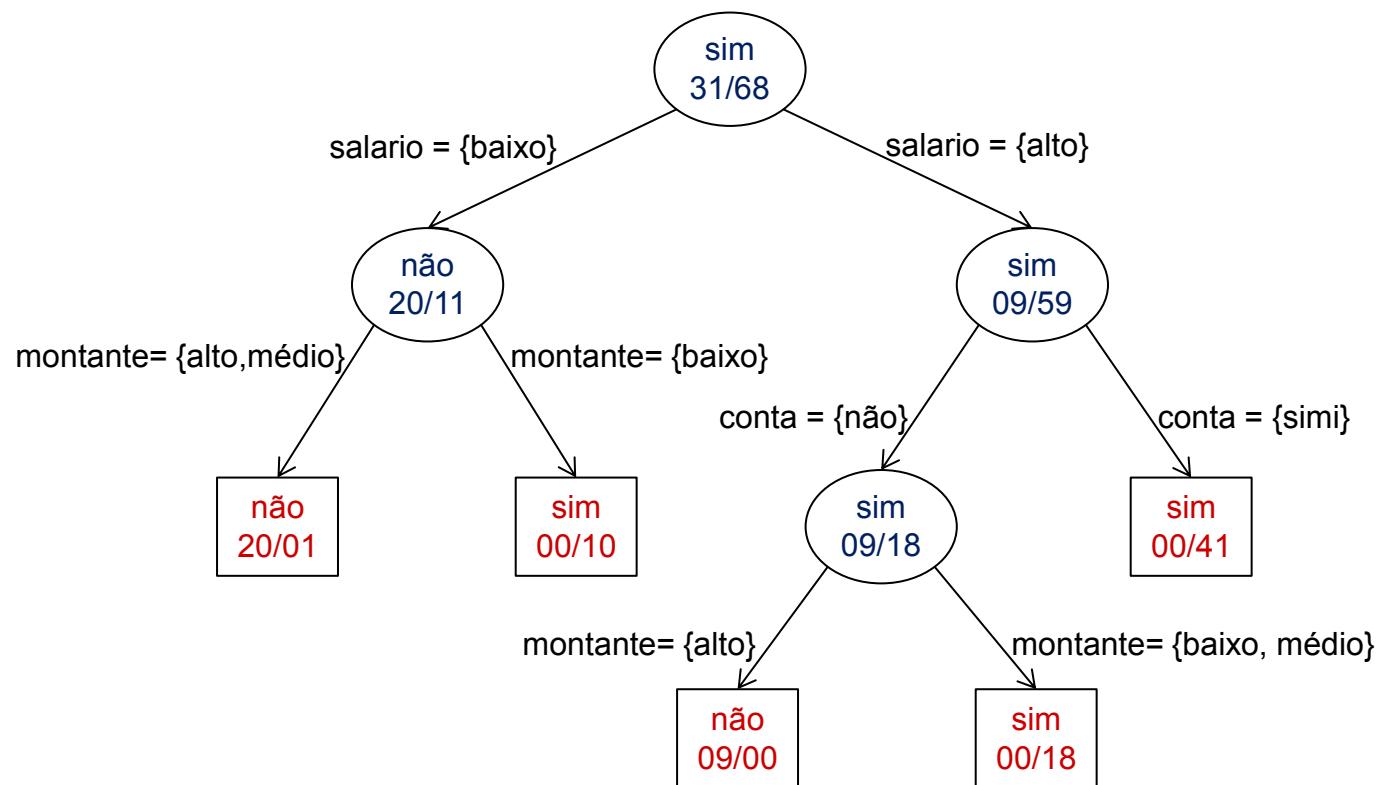
- Árvores de Decisão podem decompor-se em Regras de Decisão;
- Regras de Decisão podem obter-se simplesmente atravessando as árvores desde o nó-raiz até às folhas;
- Regras de Decisão são da forma **IF A THEN B** (e.g. IF salário=baixo & montante = baixo **THEN** conceder = sim)
- Conjunto completo das Regras de Decisão é equivalente à Árvore de Decisão;
- Regras aumentam a interpretabilidade da Árvore.



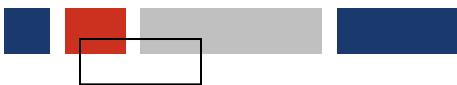
Regras de Decisão



» Exemplo*



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



Regras 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



Regras 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

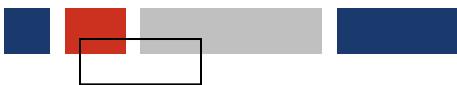


Regras de Decisão



» Algoritmos

- **One R (One Rule)**: regra criada directamente dos dados, gerada apenas com base numa variável. Cria uma árvore só com folhas.
- **Top Down**: parte de uma regra mais geral e vai acrescentando condições (de variáveis presentes ou de outras).
- **Bottom up**: parte de uma regra mais específica e vai-se retirando condições (eliminando variáveis ou não).



Regras de Decisão



» Algoritmo “One R”

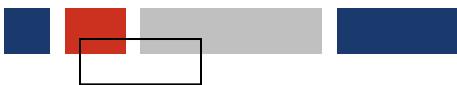
- 1) Para cada atributo A constrói-se uma regra $A = v_i \rightarrow C$,
onde v_i = valores de A e C = classe mais frequente da variável;
- 2) Seleciona-se o atributo que possui menor erro.

- Os **erros** são calculados como:

$$erro_A = \frac{\sum_{v_i} erro_{v_i}}{\sum_{v_i} total_{v_i}}$$

onde $erro_{v_i}$ é a proporção de exemplos mal classificados quando o atributo $A = v_i$.





Regras de Decisão



» Algoritmo “*Top down*”

- 1) Começa com uma regra principal (*Top*);
- 2) Remove os pontos cobertos por essa regra;
- 2) Enquanto não se verifica a **condição de paragem**:
 - 2.1) Adiciona uma nova regra
 - 2.2) Remove os pontos cobertos por essa regra

Condição de paragem:

- **cobertura** da regra atingir determinado limiar;
- **qualidade** da regra baixar de determinado limiar.



Regras de Decisão

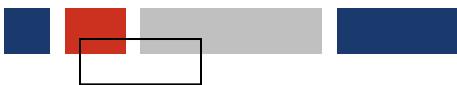


» Algoritmo “Bottom up”

- 1) Começa com uma regra complexa (*Bottom*);
- 2) Enquanto não se verifica a **condição de paragem**:
 - 2.1) Remove uma regra

Condição de paragem:

- **cobertura** da regra atingir determinado limiar;
- **qualidade** da regra baixar de determinado limiar.



Regras de Decisão

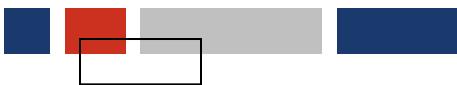


» Avaliar qualidade da regra

- A qualidade de uma regra pode ser medida por:

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

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



Regras de Decisão



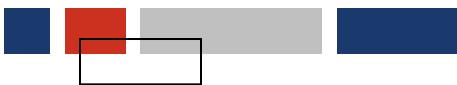
» Exercícios

2. Considere o conjunto de dados:

DMC	FUM	CT	Ab
alta	alta	sim	sim
alta	alta	não	não
baixa	alta	não	não
baixa	alta	sim	sim
baixa	baixa	sim	não

onde **DMC** = duração média de chamada, **FUM** = facturação último mês, **CT** = contrato terminou, e **Ab** = abandonou (variável-resposta).

Extraia a Regra de Decisão utilizando o algoritmo oneR.



Regras de Decisão



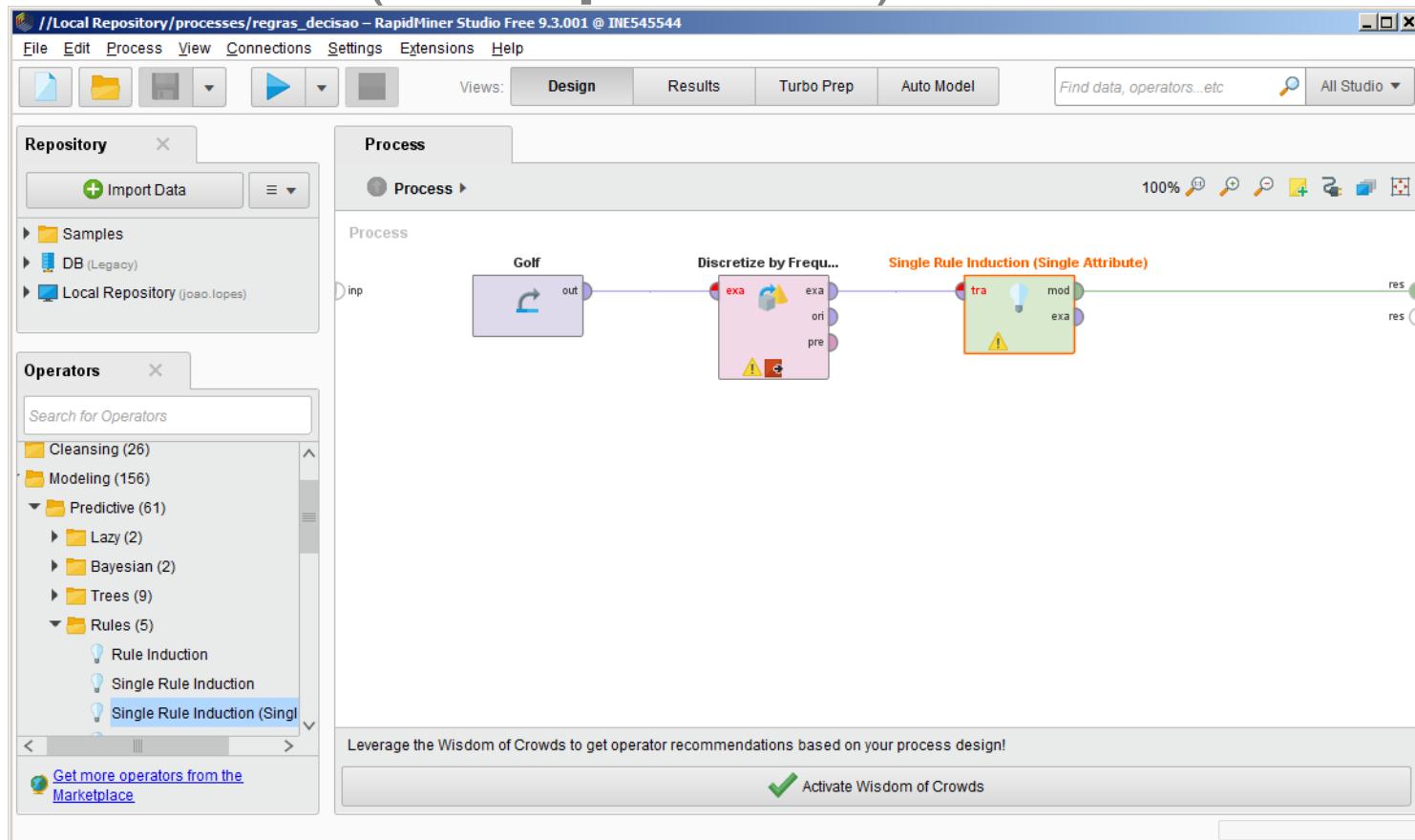
» Exercícios

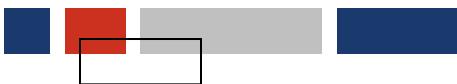
3. Considere a árvore criada anteriormente. Expanda a folha “impura” utilizando o oneR, e considerando apenas as variáveis **DMC** e **FUM**.
4. Da árvore criada, extraia um conjunto de regras com o algoritmo *top-down* (STOP: cobertura = 80%, qualidade = 100%).
5. Usando o algoritmo *bottom-up*, e a partir da regra “**IF** CT = sim & FUM = alto **THEN** Ab = sim”, extraia uma regra com cobertura $\geq 60\%$ e qualidade $\geq 60\%$.

Regras de Decisão



» Exercícios (no RapidMiner)





Regras de Decisão

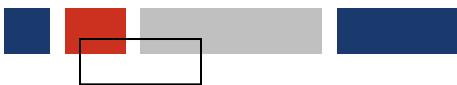


» Exercícios (no RapidMiner)

The screenshot shows a GitHub repository page for the 'rapidminer / rapidminer-studio' project. The repository has 55 stars and 182 forks. The 'Code' tab is selected, showing the file 'SingleRuleLearner.java' located at `rapidminer-studio / src / main / java / com / rapidminer / operator / learner / rules / SingleRuleLearner.java`. The file was last updated by Andreas Timm on Feb 6, 2019, with commit hash 3a2da3e. The code itself is a Java class with a detailed copyright notice:

```
1 /**
2 * Copyright (C) 2001-2019 by RapidMiner and the contributors
3 *
4 * Complete list of developers available at our web site:
5 *
6 * http://rapidminer.com
7 *
8 * This program is free software: you can redistribute it and/or modify it under the terms of the
9 * GNU Affero General Public License as published by the Free Software Foundation, either version 3
10 * of the License, or (at your option) any later version.
11 *
12 * This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without
13 * even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
14 * Affero General Public License for more details.
```





Regras de Decisão



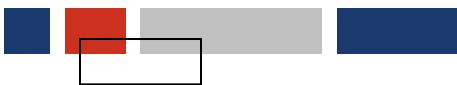
» Exercícios (no R)

```
library("OneR")

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

#Discretize numerical variables
trainset <- bin(trainset,
                 nbins=2)                                #training set
                                                #number of bins
testset <- bin(testset,
               nbins=2)                                #test set
                                                #number of bins

#Perform Decision Rule fit
mod <- OneR(Play ~.,
            trainset)                               #model design
                                                #training set
```



Regras de Decisão



```
#Summarize model
summary(mod)

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

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

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



Redes Neuronais Artificiais



» Enquadramento

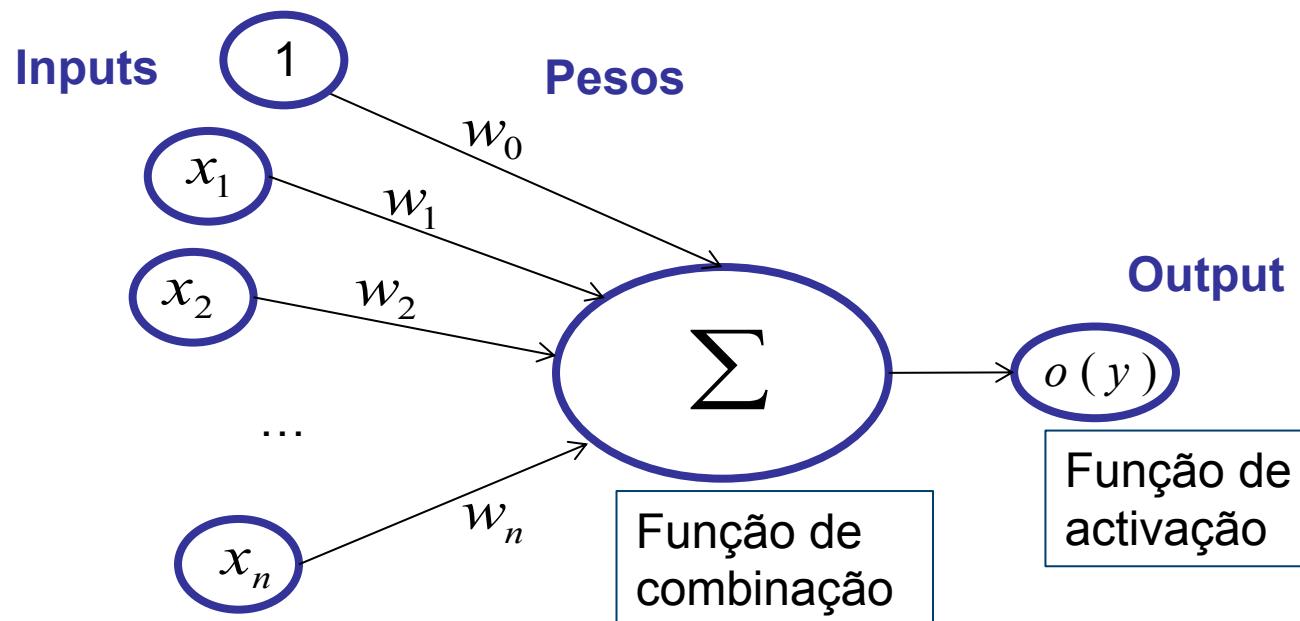
- Algoritmos de **optimização** inspirados nas redes neuronais do cérebro, onde redes densas de neurónios realizam aprendizagens complexas;
- Os neurónios recebem um **input de estímulos** que se combinam na **função de combinação** e produzem, através da **função de activação**, um **output de resposta**;
- Modelos simples: um neurónio (i.e. *single-layer perceptron*);
- Modelos complexos: multi-camadas (i.e. *multi-layer perceptron*).



Redes Neuronais Artificiais



» Single-layer perceptron





Redes Neuronais Artificiais



» *Single-layer perceptron*

Função de combinação

$$\begin{aligned}y &= w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n \\&= \sum_{i=0}^n w_i x_i\end{aligned}$$

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

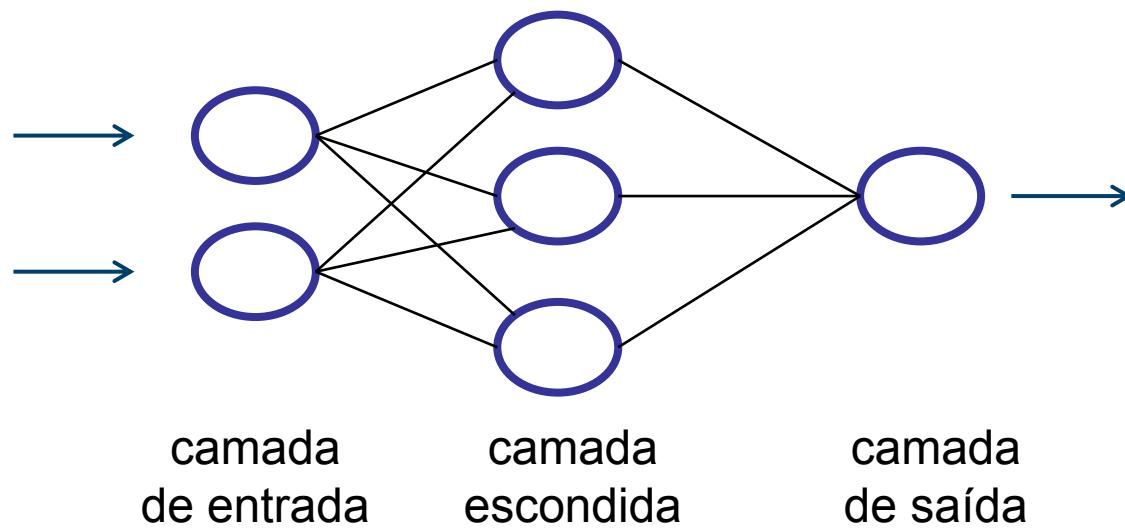


Redes Neuronais Artificiais



» Multi-layer Perceptron

- Camadas dispostas paralelamente: camada de entrada, camadas intermédias (i.e. **camadas escondidas**) e camada de saída.





Redes Neuronais Artificiais



» Algoritmo *Backpropagation*

- Constituído por duas fases: **propagação e actualização dos pesos**;
- Inputs **propagam-se para a frente** pelas camadas da rede até à camada de output;
- Output observado é comparado com o esperado usando uma **função de custo** e obtendo o erro;
- 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



» Algoritmo *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.



Redes Neuronais Artificiais



» Algoritmo *Backpropagation*: 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



» Algoritmo *Backpropagation*: 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 para **generalizar**, tolerante a **ruído** e a **dados em falta**;
- Ajusta **modelos complexos** sem tratamento analítico.
- Parâmetros do modelo de difícil interpretação (*blackbox*);
- Resultados dependem da arquitectura da rede (i.e. **pouco robustos**);
- Número de neurónios pode levar a **sobre-ajustamento** do modelo.



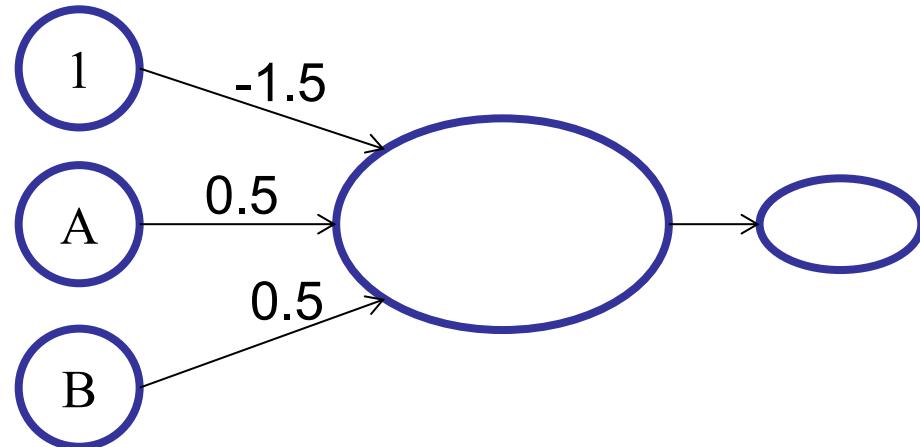
Redes Neuronais Artificiais



» Exercícios

1. Considere o seguinte conjunto de dados e o *perceptron*:

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



Utilizando a taxa de aprendizagem $\eta = 0.25$ e a função de activação,

$$f(x_1, x_2, \dots, x_n) = \begin{cases} 1, & \text{sse } \sum x_i w_i > 1 \\ 0, & \text{c.c.} \end{cases}$$

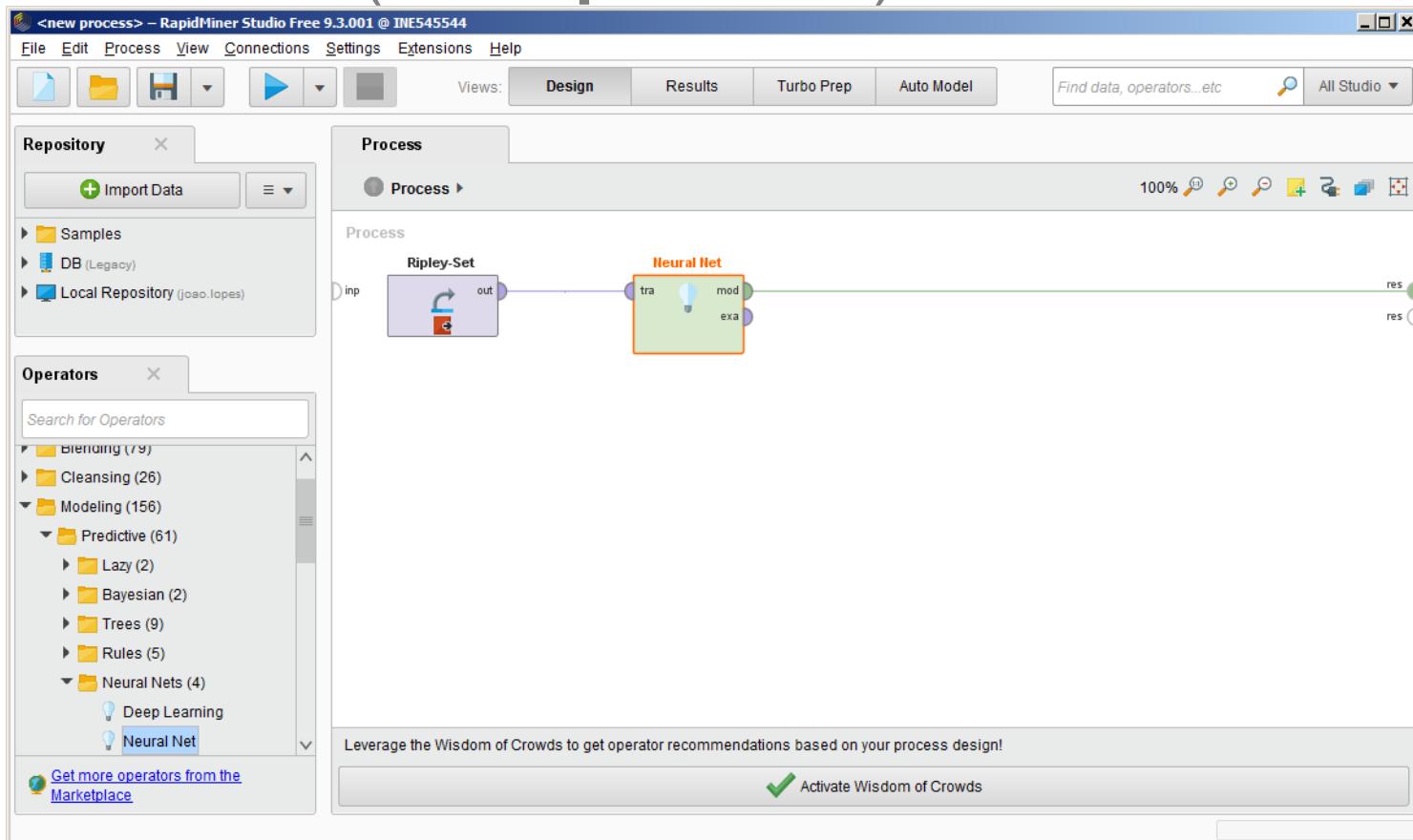
descreva uma iteração do algoritmo. O algoritmo convergiu?



Redes Neuronais Artificiais



» Exercícios (no RapidMiner)





Redes Neuronais Artificiais



» Exercícios (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 hold out
set.seed(12345)
n <- nrow(trainset)
n_test <- floor(n/3)
sampindex <- sample(1:n, size=n_test, replace=FALSE)
testset <- trainset[sampindex,]
trainset <- trainset[-sampindex,]
```



Redes Neuronais Artificiais



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



Redes Neuronais Artificiais



```
#Plot model
summary(mod)
plotnet(mod,cex_val=0.7,circle_cex=3)
olden(mod,bar_plot=TRUE)

#Test model
res <- predict(mod,
                 testset,
                 type="class") #model #testing set #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



» Enquadramento

- 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**.



Support Vector Machine



» Linear SVM Classifier

- 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 Classifier

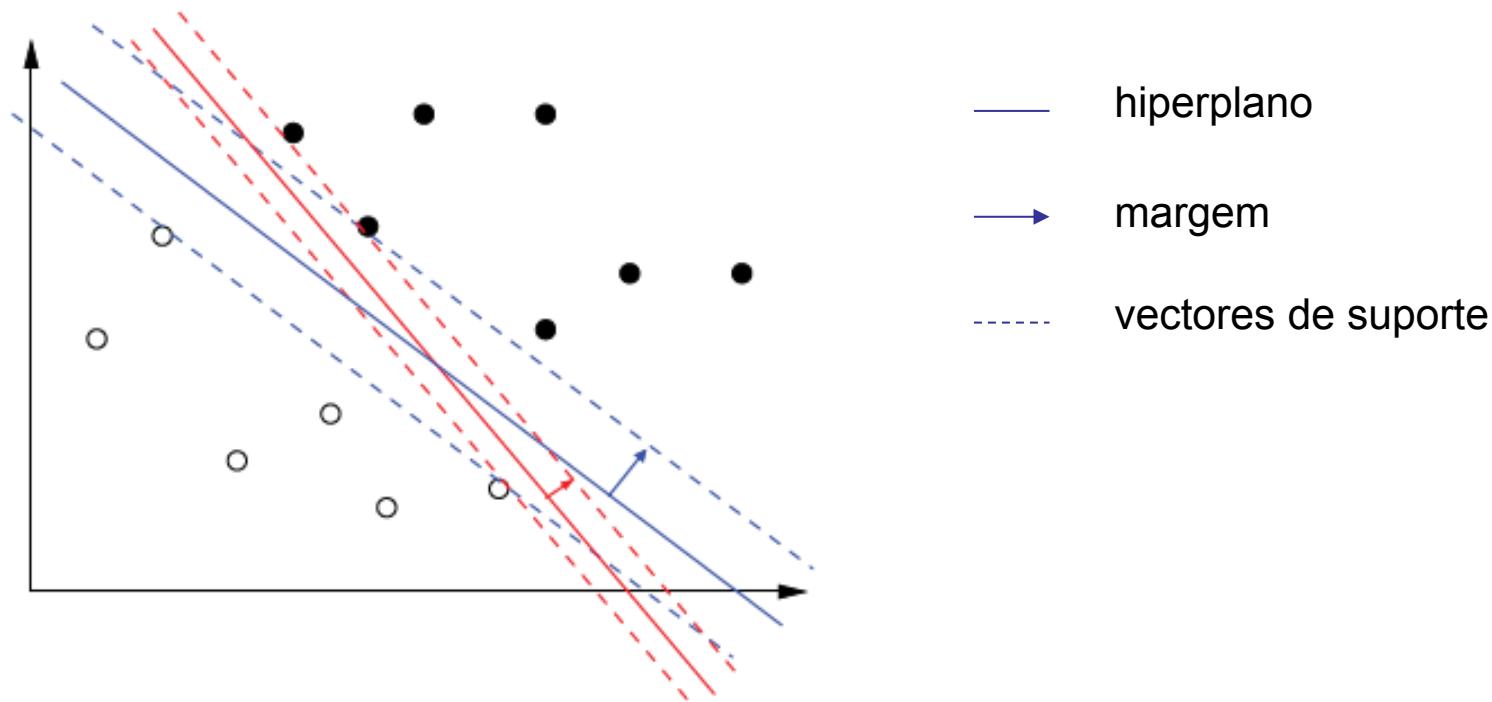


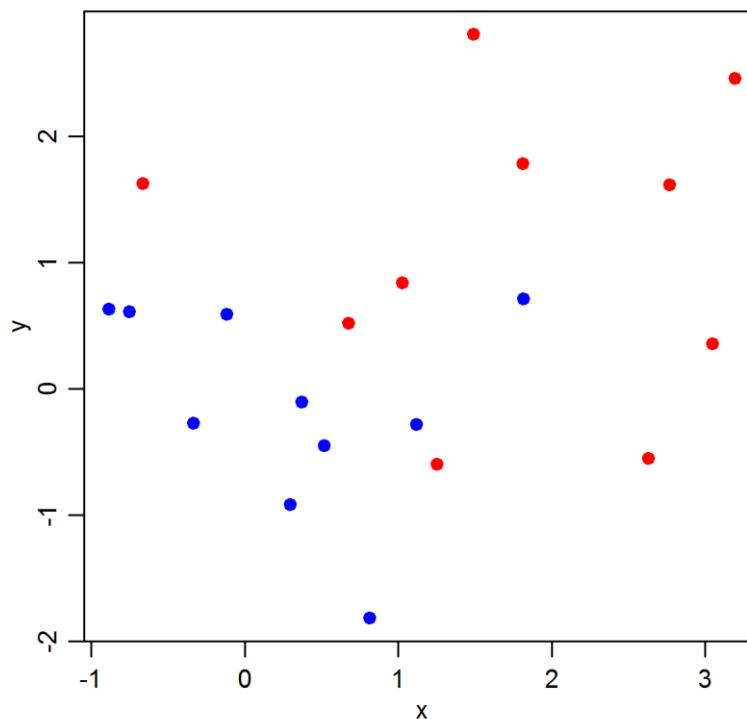
Figura 3.31 de Torga L. "Data Minig with R" (2017)



Support Vector Machine



» Exemplo Linear SVM Classifier*



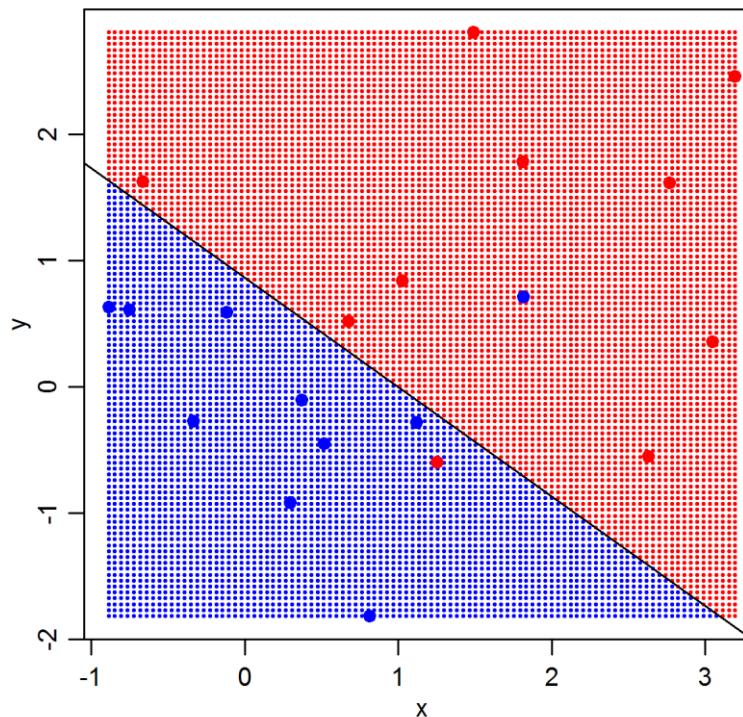
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Linear SVM Classifier*



*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)

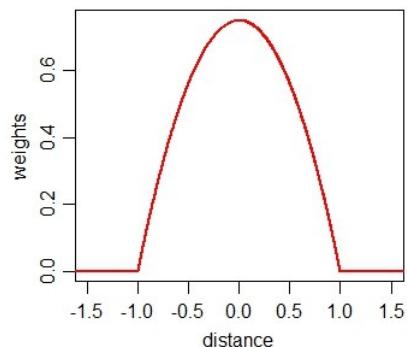


Support Vector Machine

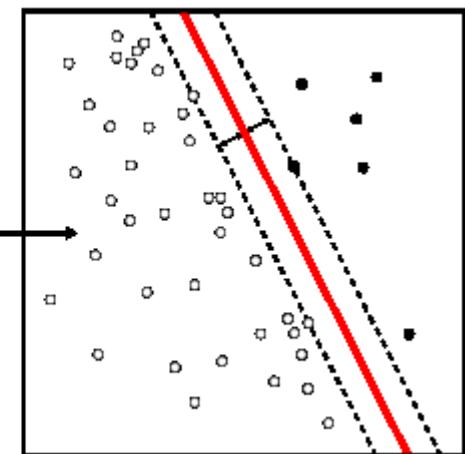
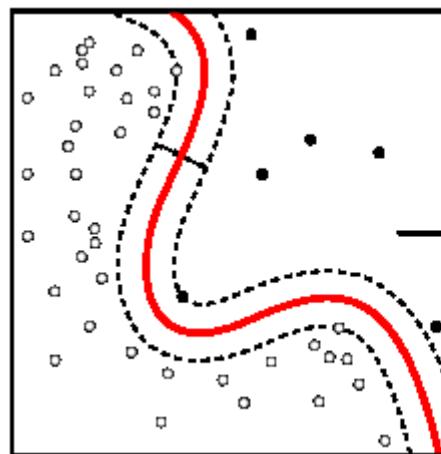


» Non-Linear SVM Classifier

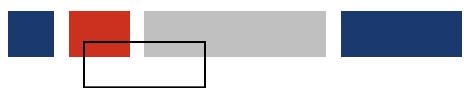
- Na maioria dos problemas reais não é possível separar as classes linearmente;
- Usa-se funções de *kernel* para definir fronteiras locais (e.g. Gaussiano, Epanechnikov, Polinomial, Radial)



epanechnikov kernel



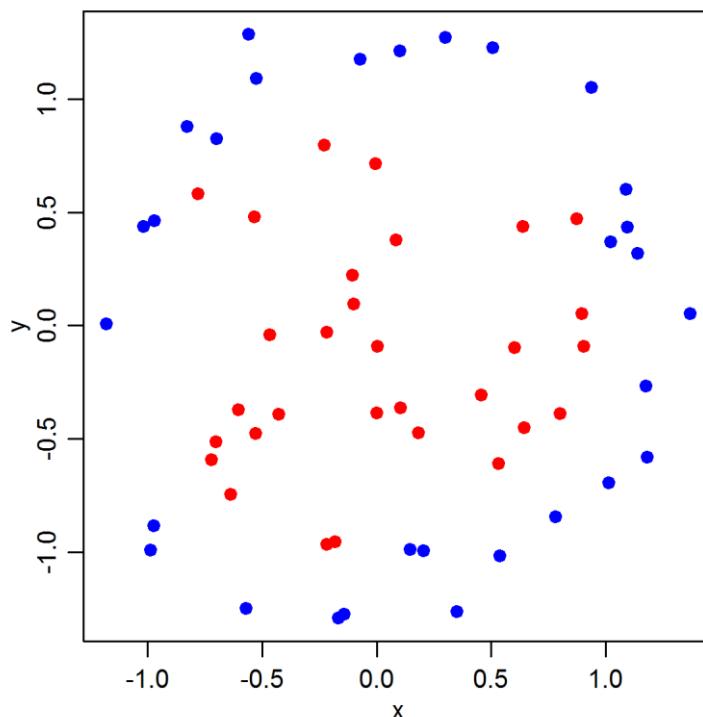
Alisneaky (2011) "Kernel Machine.png"



Support Vector Machine



» Exemplo Non-linear SVM Classifier*



hiper-plano:

$$z = x_2 + y_2$$

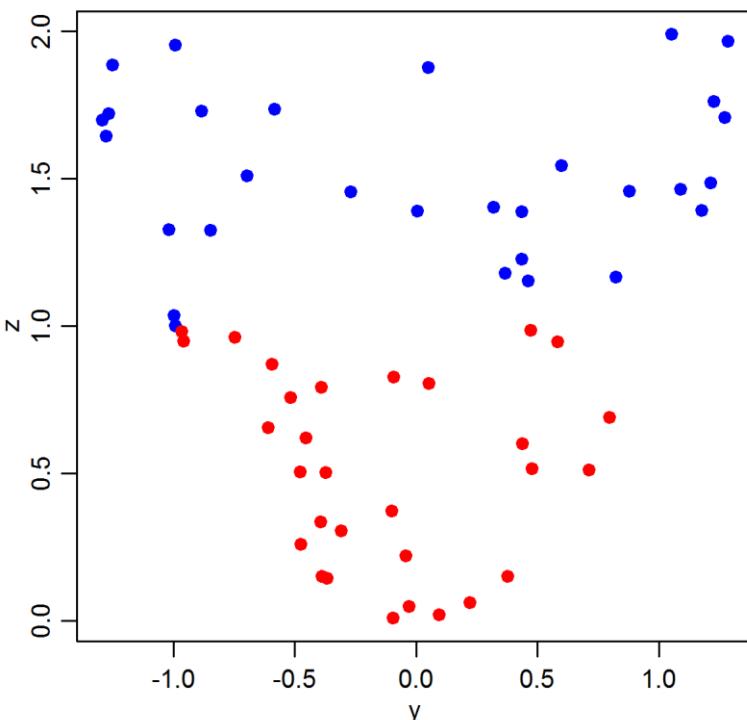
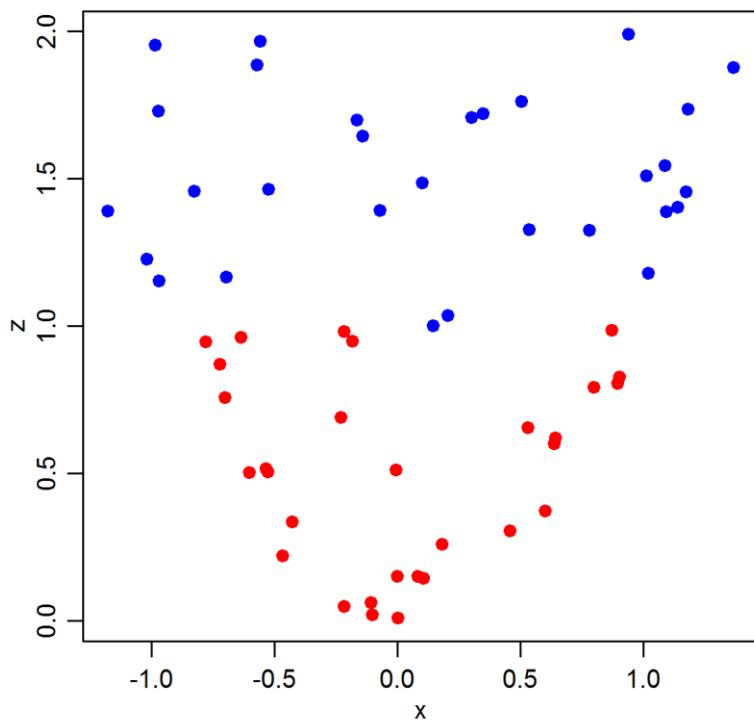
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Non-linear SVM Classifier*



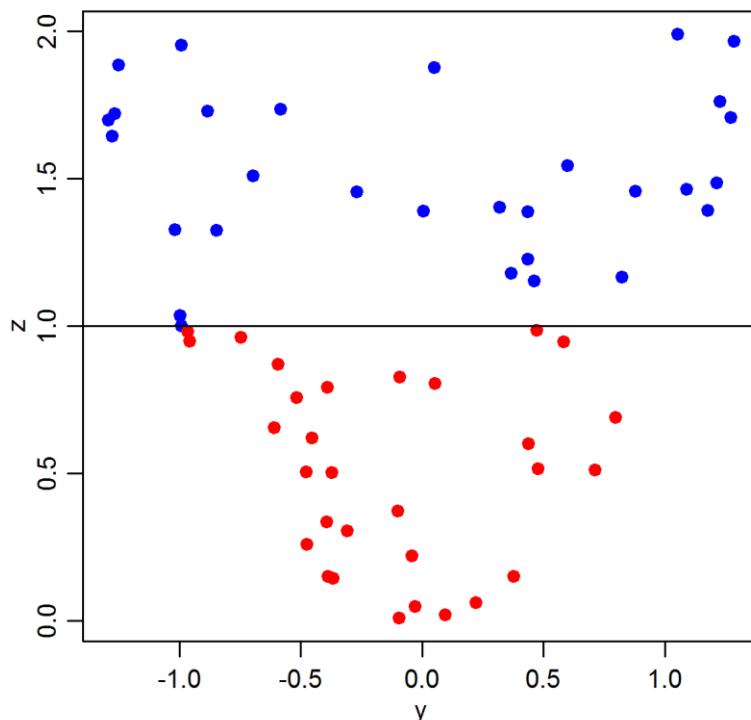
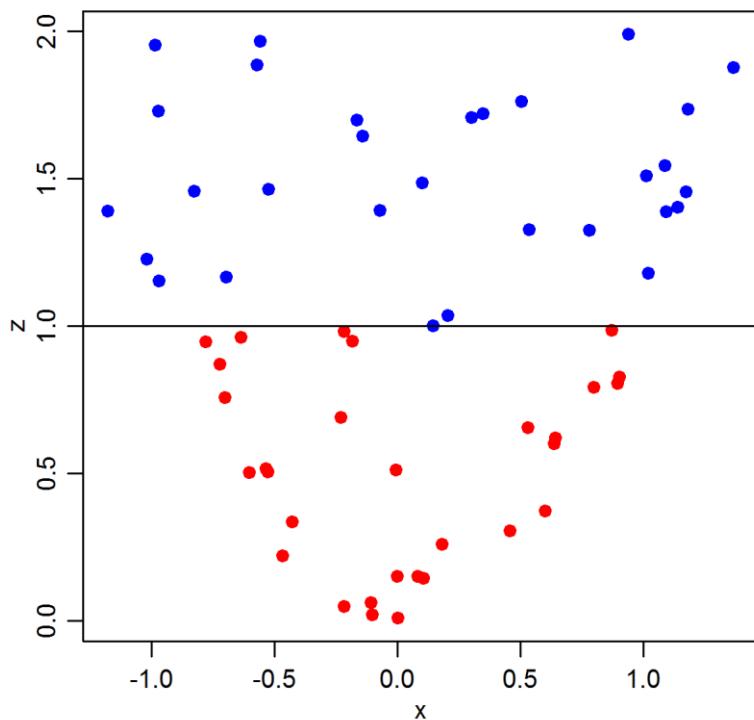
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Non-linear SVM Classifier*



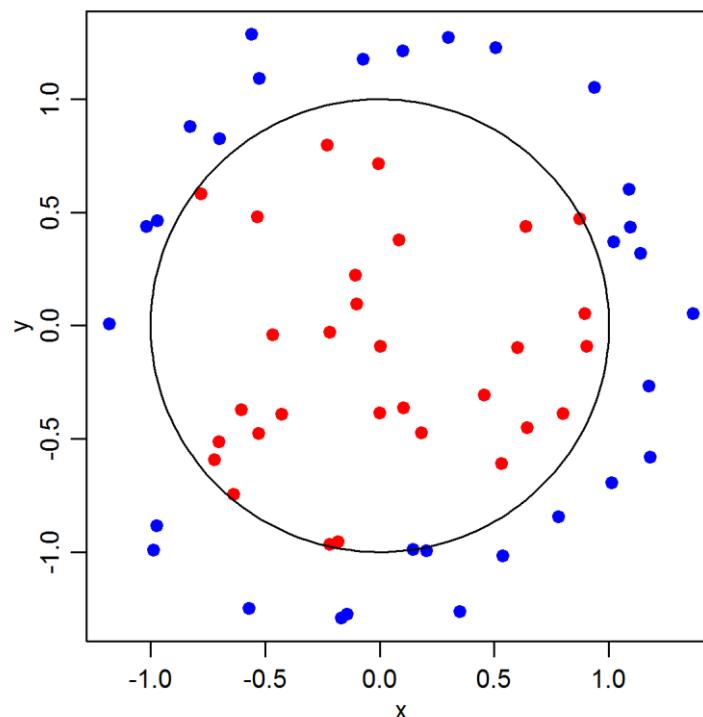
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Non-linear SVM Classifier*



hiper-plano:

$$z = x_2 + y_2$$

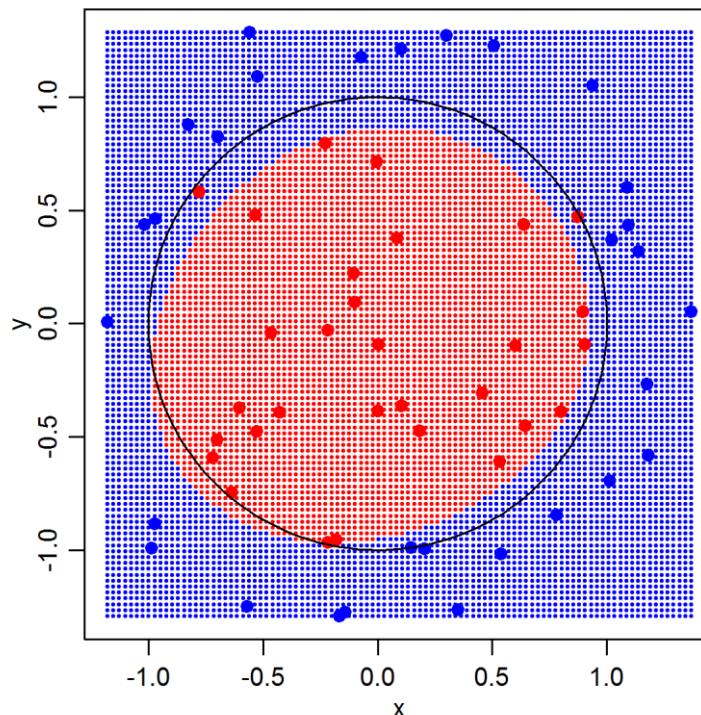
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Non-linear SVM Classifier*



*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Linear SVM Regression

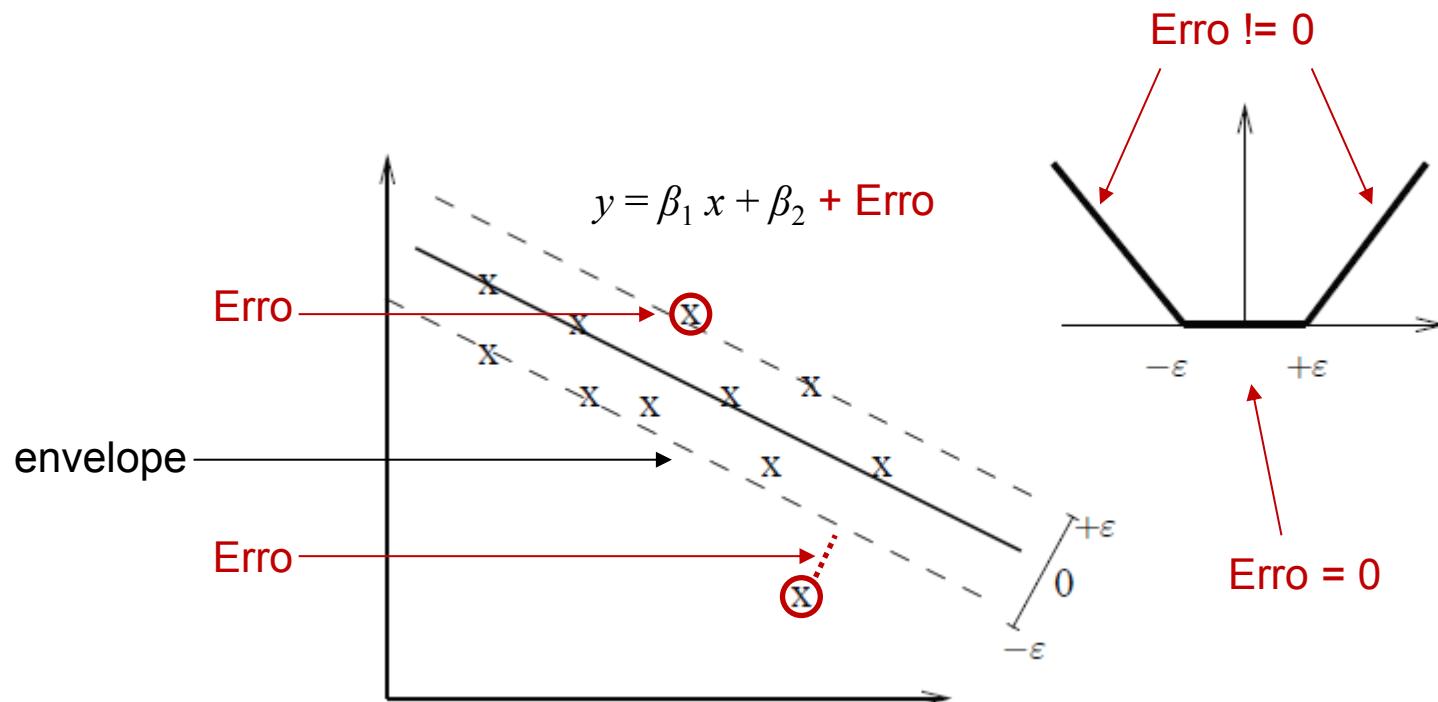


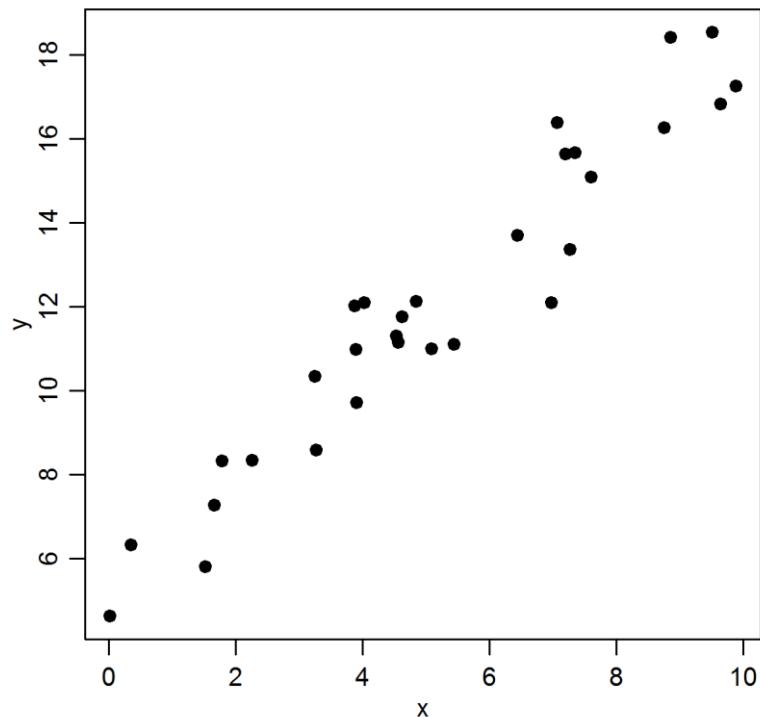
Figura 3.33 de Torga L. “Data Minig with R” (2017)



Support Vector Machine



» Exemplo Linear SVM Regressor*



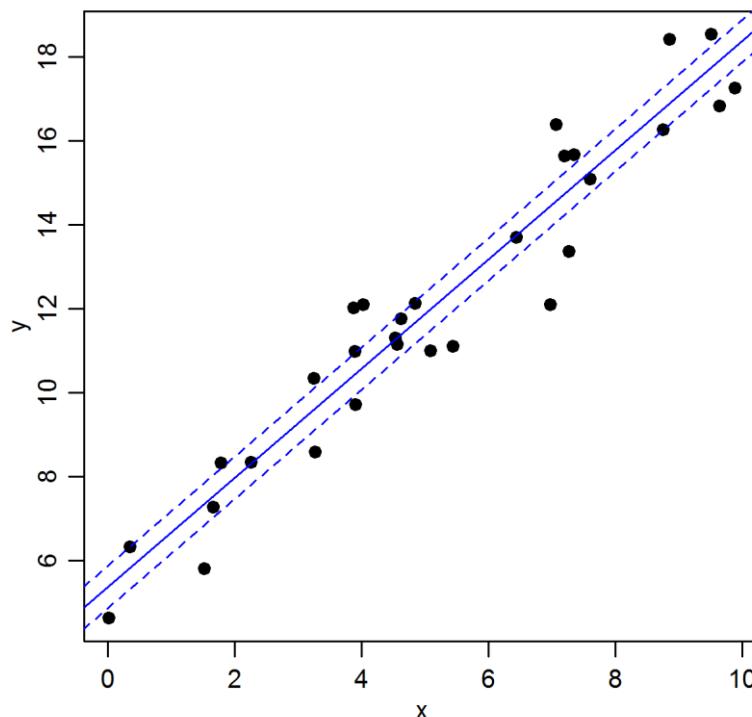
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Linear SVM Regressor*



Linear SVM

$\epsilon = 0.5$

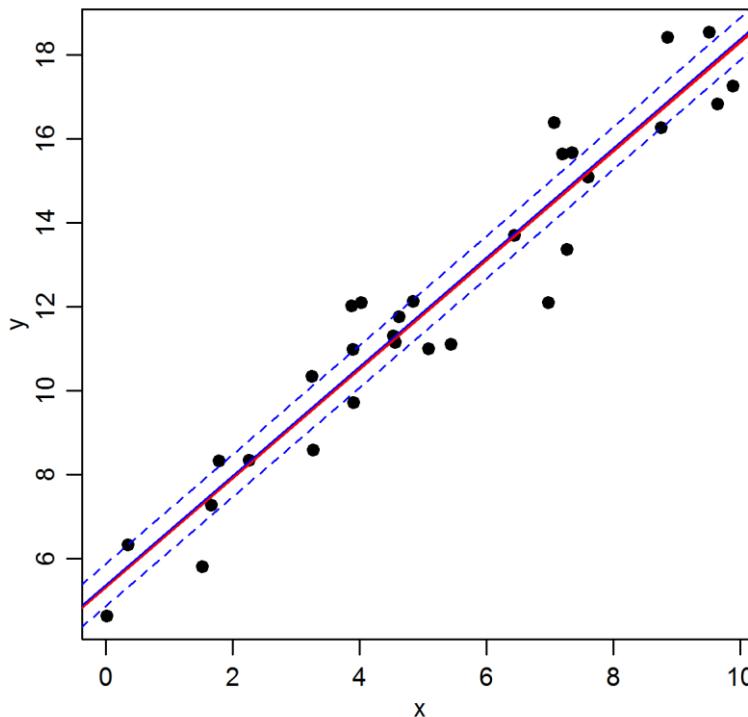
*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



Support Vector Machine



» Exemplo Linear SVM Regressor*



Linear SVM

$\varepsilon = 0.5$

Least-Squares

*<https://www.datacamp.com/community/tutorials/support-vector-machines-r> (acedido em 04-12-2019)



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. *ϵ -regression* onde ϵ 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 não-lineares são de difícil interpretação (blackbox);
- Falta de objectividade na escolha dos parâmetros da função kernell.



Support Vector Machine



» Exercícios

1. Considere os seguintes pontos:

x	y	classe
2	5	•
5	6	•
3	7	•
6	8	•

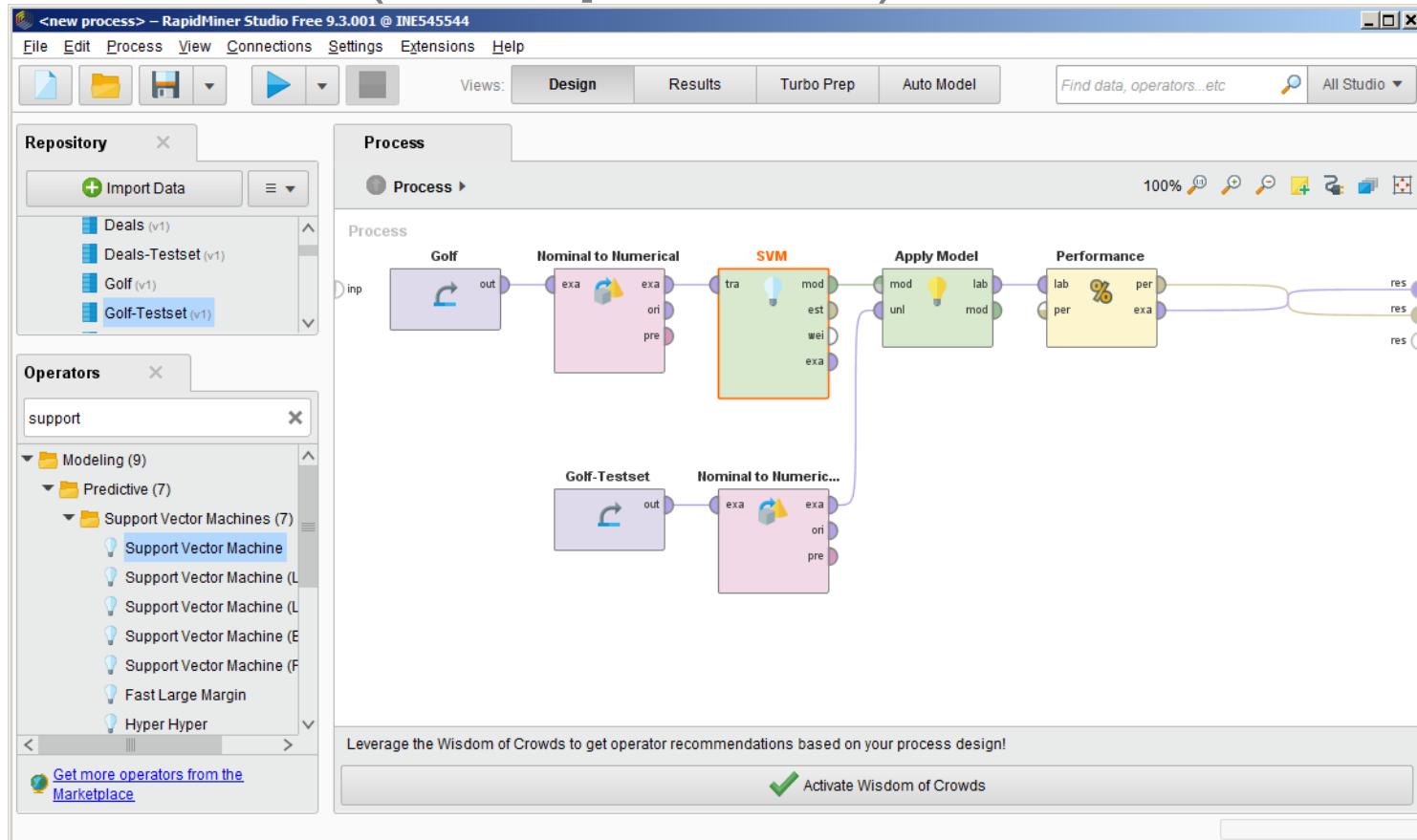
x	y	classe
1	1	■
2	3	■
4	2	■
5	4	■

- 1.1. Qual é o hiperplano paralelo ao eixo do x com a maior margem?
- 1.2. Este será o melhor hiperplano?

Support Vector Machine



» Exercícios (no RapidMiner)





Support Vector Machine



» Exercícios (no R)

```
library("e1071")

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

#Recoding dummy variables
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[, "Wind"] <- factor(testset[, "Wind"])
tmp1 <- model.matrix(~testset[, "Outlook"] - 1)
tmp2 <- model.matrix(~testset[, "Wind"] - 1)
```



Support Vector Machine



```
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 fit
set.seed(12345)
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
```



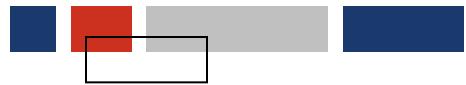
Support Vector Machine



```
#Test model
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
```



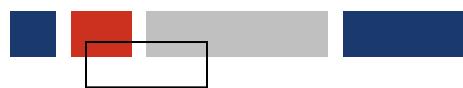
Computação Natural



» Enquadramento

- 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. **inteligência de enxames** e **computação evolutiva**.



Computação Natural



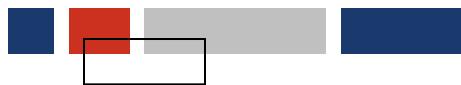
» Algoritmos

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**.

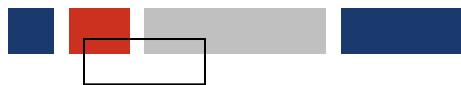
Computação Evolutiva (e.g. Algoritmos Genéticos)

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**).



» 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

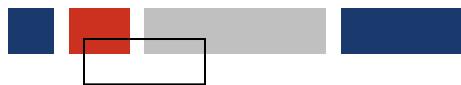
v_i , velocidade

- 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:

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

↑

inércia

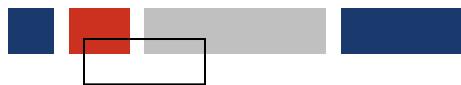
componente social

componente cognitiva

Atualização da posição:

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

- A função custo $f(x)$ actua em p_i e em I_i .



» 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** $p_i = x_i$

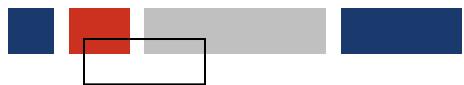
$I_i = \min(p_{vizinhança})$

 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*).

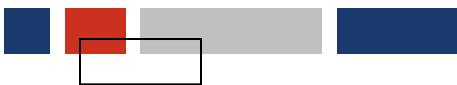


Algoritmos Genéticos



» Enquadramento

- Algoritmos de **optimização** inspirados na **evolução natural** de sistemas biológicos (popularizados por John Holland nos anos 70);
- Utilizam **população de soluções** (i.e. **agentes**), cada uma constituída por sequências de variáveis (i.e. **cromossomas**). Cada solução tem valores diferentes para cada variável (i.e. **genes**);
- As populações de soluções são avaliadas por **função de fitness**, e geram **novas gerações** de soluções;
- A população evolui ao longo das gerações através de **mecanismos evolutivos** (**seleção, recombinação, mutação, ...**).



Algoritmos Genéticos



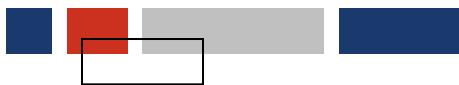
» Mecanismos evolutivos

Seleção: as “melhores” soluções passam para a geração seguinte;

Recombinação: troca de informação entre soluções;

Mutação: a transferência de informação entre gerações é imperfeita;

outros (e.g. Migração): entrada de soluções novas na população.



Algoritmos Genéticos



» Algoritmo

- Cada indivíduo é caracterizada por:

x_i , vector de valores dos atributos (cromossoma)

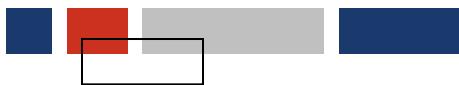
- Parâmetros do modelo evolutivo:

m , taxa de mutação

r , taxa de crossover

Outros parâmetros:

$f(x)$, função *fitness*

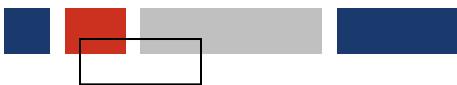


Algoritmos Genéticos



» 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

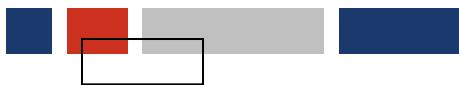


» Exemplo*

- Um explorador vai para a selva;
- Mochila com capacidade de 20Kg;
- Conjunto de itens caracterizados por pontos de sobrevivência e peso;
- Qual o melhor conjunto de itens?

ITEM	SURV. PTS.	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 é um problema de otimização clássico do século XIX. Adaptado de Marek Obitko. <http://www.r-bloggers.com/genetic-algorithms-a-simple-r-example/>



Algoritmos Genéticos



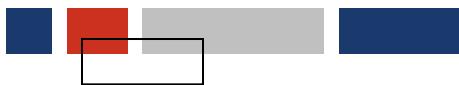
» Exemplo*

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

Exemplos de instâncias:

pkt-knife	beans	potatoes	unions	sleep 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 é um problema de otimização clássico do século XIX. Adaptado de Marek Obitko. <http://www.r-bloggers.com/genetic-algorithms-a-simple-r-example/>

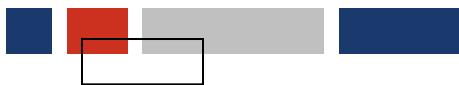


Algoritmos Genéticos



» Vantagens e Desvantagens

- Algoritmo é facilmente interpretável;
 - Processo de optimização robusto a ruído e a máximos locais;
 - Facilmente paralelizável, (i.e. redução de tempo de computação).
-
- Rapidez de convergência depende da escolha dos parâmetros do modelo e dos critérios de paragem;
 - A função de *fitness* requer tratamento analítico;
 - Número de variáveis utilizado influencia grandemente a intensidade computacional (i.e. *curse of dimensionality*).

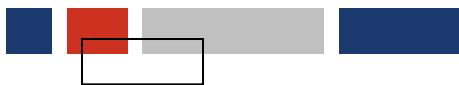


Algoritmos Genéticos



» Exercícios

1. Crie 10 instâncias para o “problema no *Knapsack*”. Considere uma taxa de seleção de 50% (as melhores 5 instâncias contribuem com 2 cópias cada). Como *crossover*, escolha 3 pares diferentes das instâncias selecionadas e troque os últimos 3 genes. Finalmente, insira uma mutação aleatória em cada instância. Repita o processo 2 vezes e indique a solução final.



Algoritmos Genéticos



» Exercícios (no R)

```
library("GA")

#Define data
n <- c("pocketknife","beans","potatoes","unions","sleepingbag","rope","compass")
p <- c(10,20,15,2,30,10,30)                      #profits
w <- c(1,5,10,1,7,5,1)                          #weights
W <- 20                                         #knapsack capacity

#Define fitness function
knapsack <- function(x){
  if(sum(x*w) > W)
    return(0)
  else
    return(sum(x*p))
}
```



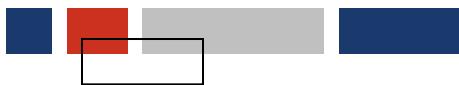
Algoritmos Genéticos



```
#Run SGA

SGA <- ga(type="binary",
fitness=knapsack,                                #fitness function
nBits=length(n),                                 #chromosome length
popSize=100,                                      #population size
pcrossover=0.8,                                    #crossover rate
pmutation=0.1,                                     #mutation rate
elitism=5,                                         #number of best individuals sure to be selected
maxiter=100,                                       #number of generations
names=n,                                           #name of "genes"
seed=12345)                                        #random seed

res <- SGA@solution
print(res)                                         #best solution
sum(res)                                           #total number of selected items
sum(res*p)                                         #total profit of selected items
sum(res*w)                                         #total weight of selected items
```



Algoritmos Genéticos



```
#Plot model  
  
plot(SGA@summary[, "mean"], type="l", ylab="mean fitness", xlab="iteration", lwd=2)  
  
barplot(SGA@fitness, ylab="fitness", xlab="", col="blue")
```



Exemplos *Data Analytics*



» EU BD Hackathon 2017

- **Objectivo:** ajudar a definir políticas sobre o mercado laboral
- **Organização:** Eurostat and CEDEFOP
- **Projecto:**
 - Caracterização do Mercado Laboral.
 - Estabelecer associações entre indicadores relevantes (e.g. Mobilidade Laboral na UE) e características do mercado laboral.

<https://github.com/jsollari/EUhackathon2017>

<https://ec.europa.eu/eurostat/web/products-statistical-working-papers/-/KS-TC-18-002>



Exemplos *Data Analytics*



» Dados: características do Mercado Laboral

- “reg_dem” – informação demográfica
- “earn” – estrutura de ganhos
- “educ_uee_fin” – gastos públicos em educação
- “ilc” – rendimento e condições de vida
- “employ” – informação sobre o emprego
- “nama10” – contas nacionais
- “educ_part” – informação sobre educação

7 datasets, 17 main variables



Exemplos *Data Analytics*



» Dados: características do Mercado Laboral

- “reg_dem” by **age**
- “earn” by **occupation** and **economic activity**
- “educ_uee_fin”
- “ilc”
- “employ” by **age**, **education level**, **economic activity**
- “nama10”
- “educ_part”

7 datasets, 17 main variables, 76 variables



Exemplos *Data Analytics*



» Dados: Mobilidade Laboral na UE

- “Ifso_14leeow” – informação sobre a força laboral

1 dataset, 1 main variable

subjects: **25 (NUTS0)**

Exemplos Data Analytics



» Clustering de países EU (no RapidMiner)

The screenshot shows the RapidMiner Studio Free 9.3.001 interface. The window title is '<new process> - RapidMiner Studio Free 9.3.001 @ INE545544'. The menu bar includes File, Edit, Process, View, Connections, Settings, Extensions, and Help. The 'File' menu is open, showing options like New Process, Open Process..., Recent Processes, Save Process, Save Process as..., Import Data (which is selected), Import into a repository, Export Process..., Print/Export Image, and Exit. The central workspace is titled 'Process' and displays the message 'Your process looks empty. Add some data first. Drag data or operators here.' Below the workspace, there's a sidebar titled 'Operators' with a search bar and a list of categories: Data Access (53), Blending (79), Cleansing (26), Modeling (156), and Scoring (12). At the bottom of the sidebar, there's a link 'Get more operators from the Marketplace'. A status bar at the bottom of the window says 'Leverage the Wisdom of Crowds to get operator recommendations based on your process design!' and has a button 'Activate Wisdom of Crowds'.

Exemplos Data Analytics



» Clustering de países EU (no RapidMiner)

Import Data - Format your columns.

Format your columns.

Date format: Enter value... ▾ Replace errors with missing values ⓘ

	id polynomial	pop_Total real	pop_Y0-14 real	pop_Y15-24 real	pop_Y25-64 real	pop_Y65-74 real	pop_YGE75 real	ARPR real
1	AT	14.328	11.900	55.473	10.155	8.144	14.100	
2	BE	Rename role [Opens a dialog to change the role.]	11.904	53.264	8.948	8.888	15.500	
3	BG	Exclude column	13.748	10.377	56.309	11.147	8.419	21.800
4	CY	858000	16.270	14.254	55.605	8.108	5.763	14.400
5	CZ	10512419	15.006	10.701	56.928	10.559	6.806	9.700
6	DE	80767463	13.177	10.775	55.183	10.634	10.232	16.700
7	DK	5627235	17.214	12.884	51.656	10.870	7.376	12.100
8	EE	1315819	15.810	11.169	54.646	9.526	8.849	21.800
9	EL	10926807	14.644	10.405	54.465	9.965	10.521	22.100
10	ES	46512199	15.194	9.708	56.953	8.895	9.251	22.200
11	FI	5451270	16.419	12.005	52.195	10.863	8.519	12.800
12	FR	65942093	18.543	11.894	51.548	8.923	9.092	13.300
13	HR	4246809	14.779	11.661	55.130	9.783	8.647	19.400
14	HU	9877365	14.435	11.873	56.158	10.024	7.510	15.000
15	IE	4605501	22.003	11.414	53.974	7.303	5.306	16.400
16	IT	60782668	13.899	9.834	54.855	10.681	10.732	19.400
17	LT	2943472	14.612	13.258	53.691	9.389	9.050	19.100
18	LU	549680	16.838	12.037	57.049	7.414	6.662	16.400

< ||| > no problems.

Previous Next Cancel

Exemplos Data Analytics



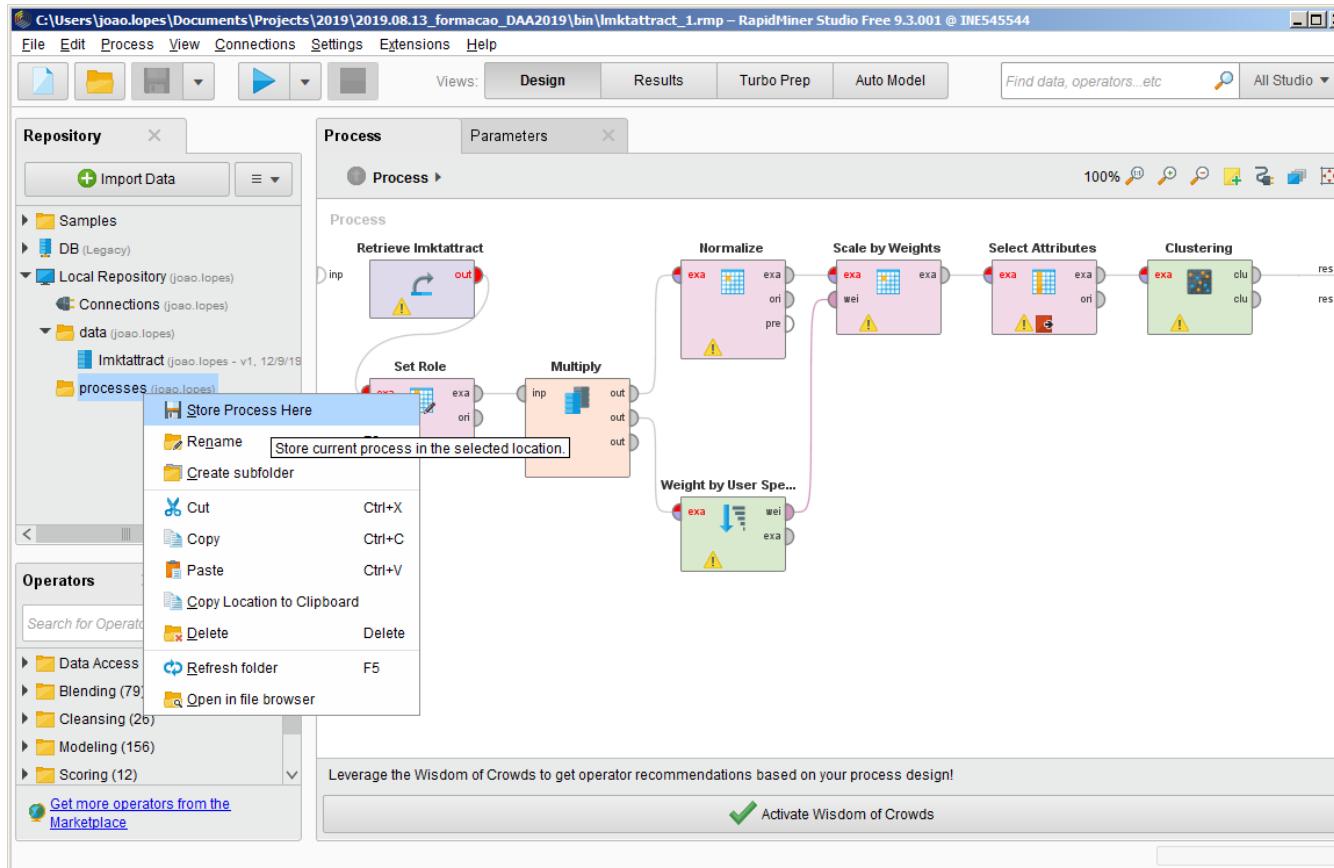
» Clustering de países EU (no RapidMiner)

The screenshot shows the RapidMiner Studio Free 9.3.001 interface. The window title is '<new process> - RapidMiner Studio Free 9.3.001 @ INE545544'. The menu bar includes File, Edit, Process, View, Connections, Settings, Extensions, and Help. The 'File' menu is open, showing options like New Process, Open Process..., Save Process, Import Data, Import Process... (which is selected), Export Process, Print/Export Image, and Exit. A file named 'processes (joao.lopes)' is listed under the Exit option. The main workspace is titled 'Process' and contains the message 'Your process looks empty. Add some data first. Drag data or operators here.' Below this, there's a section for operator recommendations with the text 'Leverage the Wisdom of Crowds to get operator recommendations based on your process design!' and a checkbox labeled 'Activate Wisdom of Crowds'. On the left, there's an 'Operators' panel with a search bar and a list of categories: Data Access (53), Blending (79), Cleansing (26), Modeling (156), and Scoring (12). At the bottom of the operators panel, there's a link 'Get more operators from the Marketplace'.

Exemplos Data Analytics



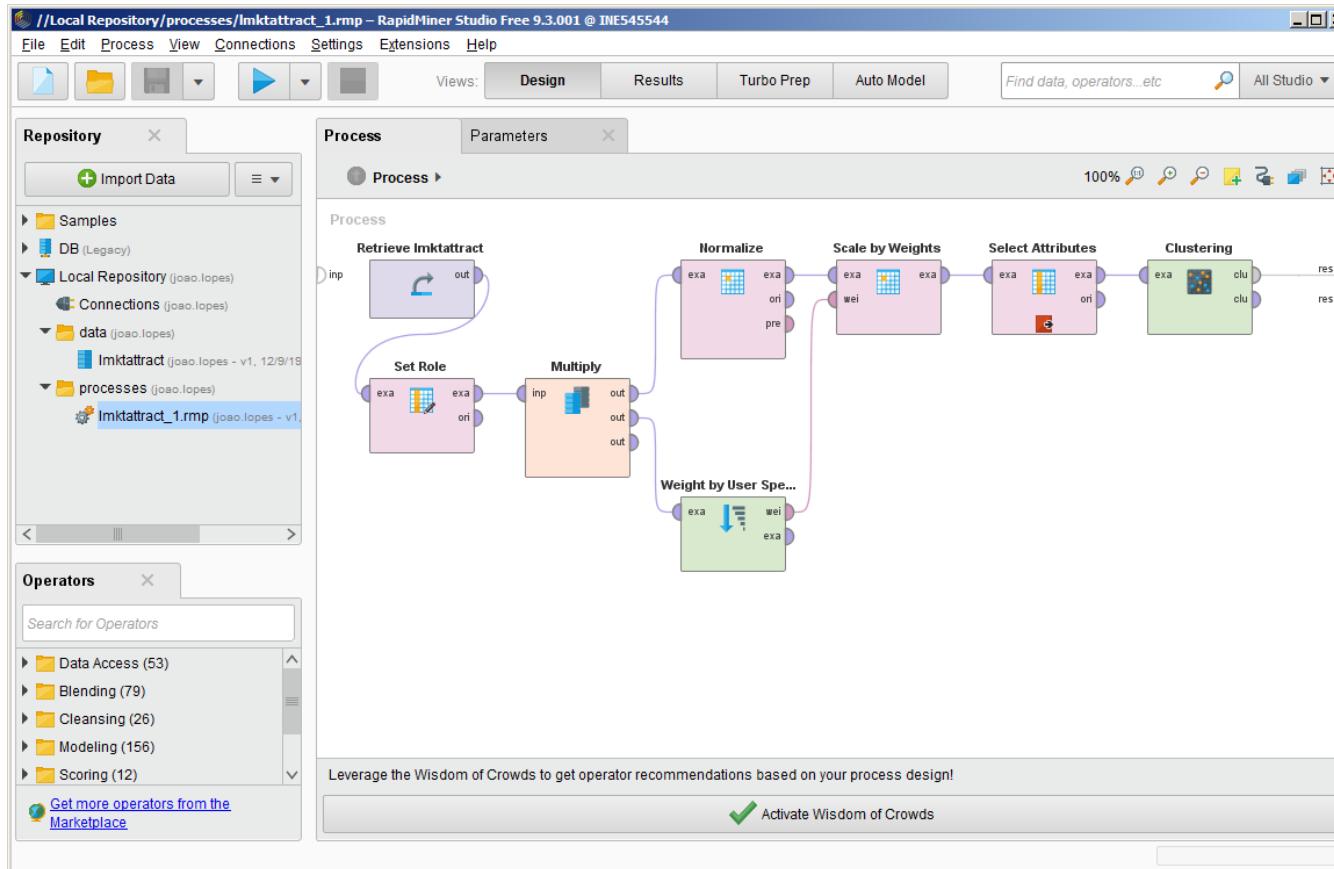
» Clustering de países EU (no RapidMiner)



Exemplos Data Analytics



» Clustering de países EU (no RapidMiner)





Exemplos *Data Analytics*



» Clustering de países EU (no R)

```
#1. FUNCTIONS
source("misc_v2.3.r")

#2. READ DATA
f1 <- "../data/lmktattract.csv"
x_all <- read.table(file=f1,header=TRUE,sep=",",dec=".",
                     row.names=1)

# 3. ANALYSE DATA
set.seed(12345)
main1 <- "Labour market attractiveness" #title for plots
col1 <- c(rbind(hsv(h=seq(0,1,len=8),s=1/3,v=5/5)[1:7], #very light
                 hsv(h=seq(0,1,len=8),s=2/3,v=5/5)[1:7], #light
                 hsv(h=seq(0,1,len=8),s=3/3,v=5/5)[1:7], #normal
                 hsv(h=seq(0,1,len=8),s=3/3,v=4/5)[1:7])) #dark
names(col1) = rownames(x_all) #colours for EU28
```



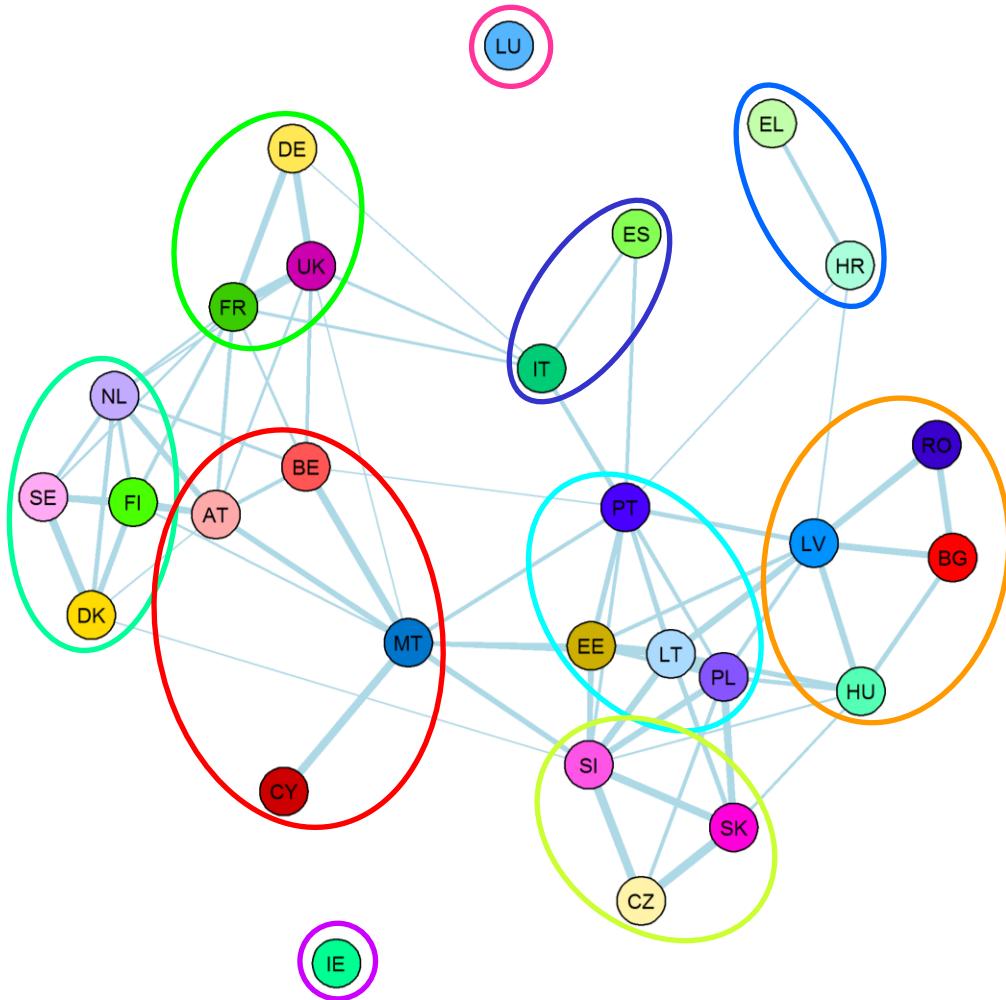
Exemplos *Data Analytics*

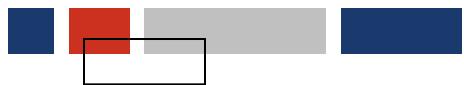


```
#normalize and scale data  
  
n_all <- c(1,5,1,1,1,1,1,6,2,3,20,2,1,1,1,1,27,1)  
w_all <- rep(1/n_all,times=n_all)  
x_norm <- t(apply(scale(x_all),1,function(x){x*w_all}))  
x_dist <- dist(x_norm,method="euclidean") #calculate distances  
  
#perform Social Network Analysis  
set.seed(12345)  
f1 <- "../results/lmktattract_1/net"  
plot_sna_v2(x_dist,col1=col1,f1=f1,main=main1)  
  
#perform Partition Around Medoids analysis  
f1 <- "../results/lmktattract_1/pam"  
plot_pam_v2(x_dist,col1=col1,f1=f1,main=main1)  
kbest <- 10  
f1 <- paste("../results/lmktattract_1/pam_grps",kbest,sep="")  
plot_pam_v2(x_dist,kbest=kbest,col1=col1,f1=f1,main=main1)
```

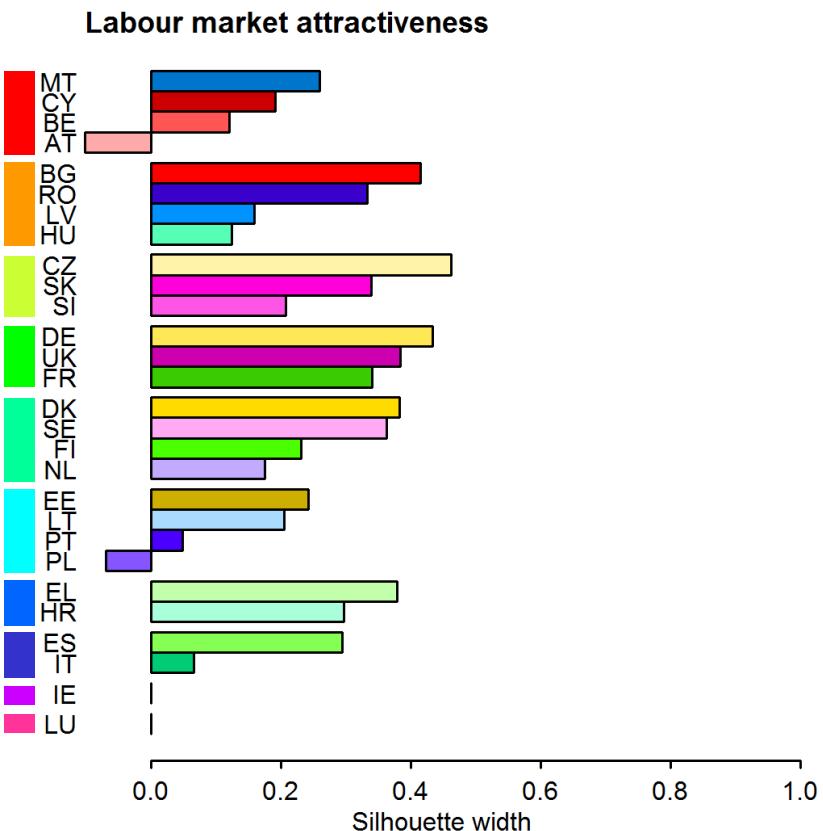
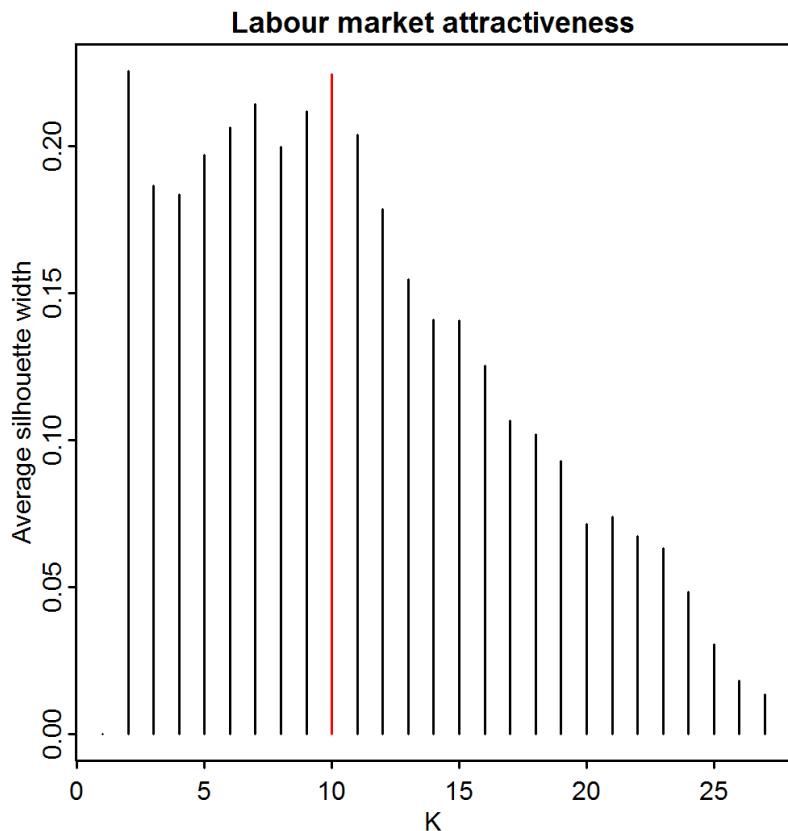


Exemplos Data Analytics





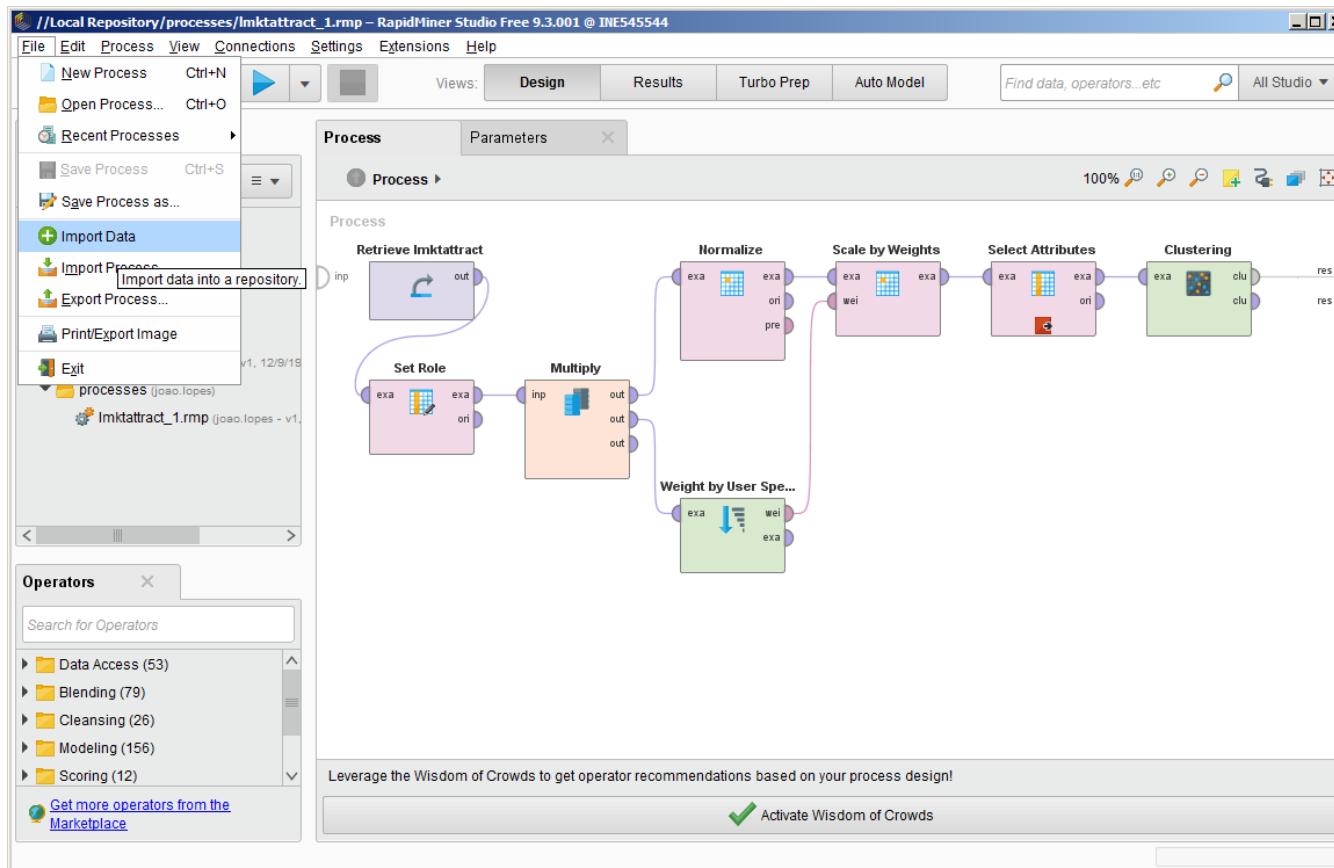
Exemplos Data Analytics



Exemplos Data Analytics



» Relação entre indicadores (no RapidMiner)



Exemplos Data Analytics



» Relação entre indicadores (no RapidMiner)

Import Data - Format your columns.

Format your columns.

Date format: Enter value... ▾ Replace errors with missing values ⓘ

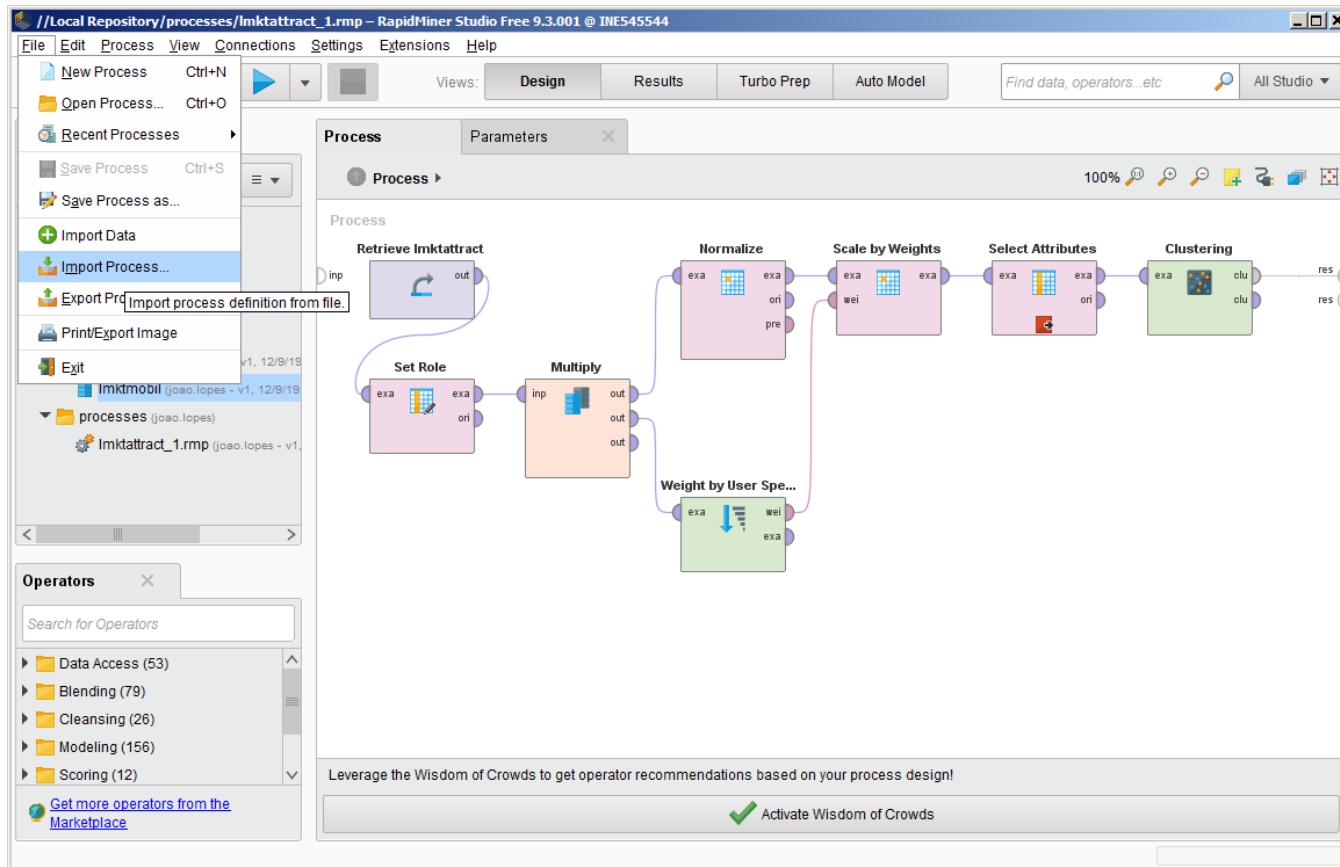
	id polynomial	lmktm_OC0 real	lmktm_OC1 real	lmktm_OC2 real	lmktm_OC3 real	lmktm_OC4 real	lmktm_OC5 real	lmktm_OC6 real
1	AT	Change Type ▾	22.351	24.369	19.152	19.740	30.740	20.492
2	BE	Change Role	Rename <small>Opens a dialog to change the role.</small>	18.446	17.850	19.289	26.516	?
3	BG	Exclude column	?	?	?	?	?	?
4	CY	20.930	28.283	18.929	15.776	13.947	25.836	?
5	CZ	10.526	8.224	7.052	5.177	5.339	7.351	8.914
6	DE	?	9.603	13.484	13.451	13.112	22.374	?
7	DK	?	?	?	?	?	?	?
8	EE	?	24.353	26.176	30.357	30.707	35.668	?
9	EL	?	?	2.959	4.962	3.851	13.492	53.503
10	ES	6.041	8.631	8.865	5.642	7.502	18.815	28.862
11	FI	?	5.589	5.936	4.761	4.035	8.239	10.638
12	FR	?	27.757	27.383	22.768	23.882	27.705	?
13	HR	?	?	14.853	16.271	15.949	17.275	?
14	HU	?	4.100	4.196	3.013	3.421	3.691	?
15	IE	?	?	?	?	?	?	?
16	IT	5.011	8.151	5.483	7.753	6.819	22.380	37.076
17	LT	?	7.261	8.183	9.382	?	9.145	?
18	LU	?	86.275	71.524	57.209	55.676	68.085	?

no problems.

Exemplos Data Analytics



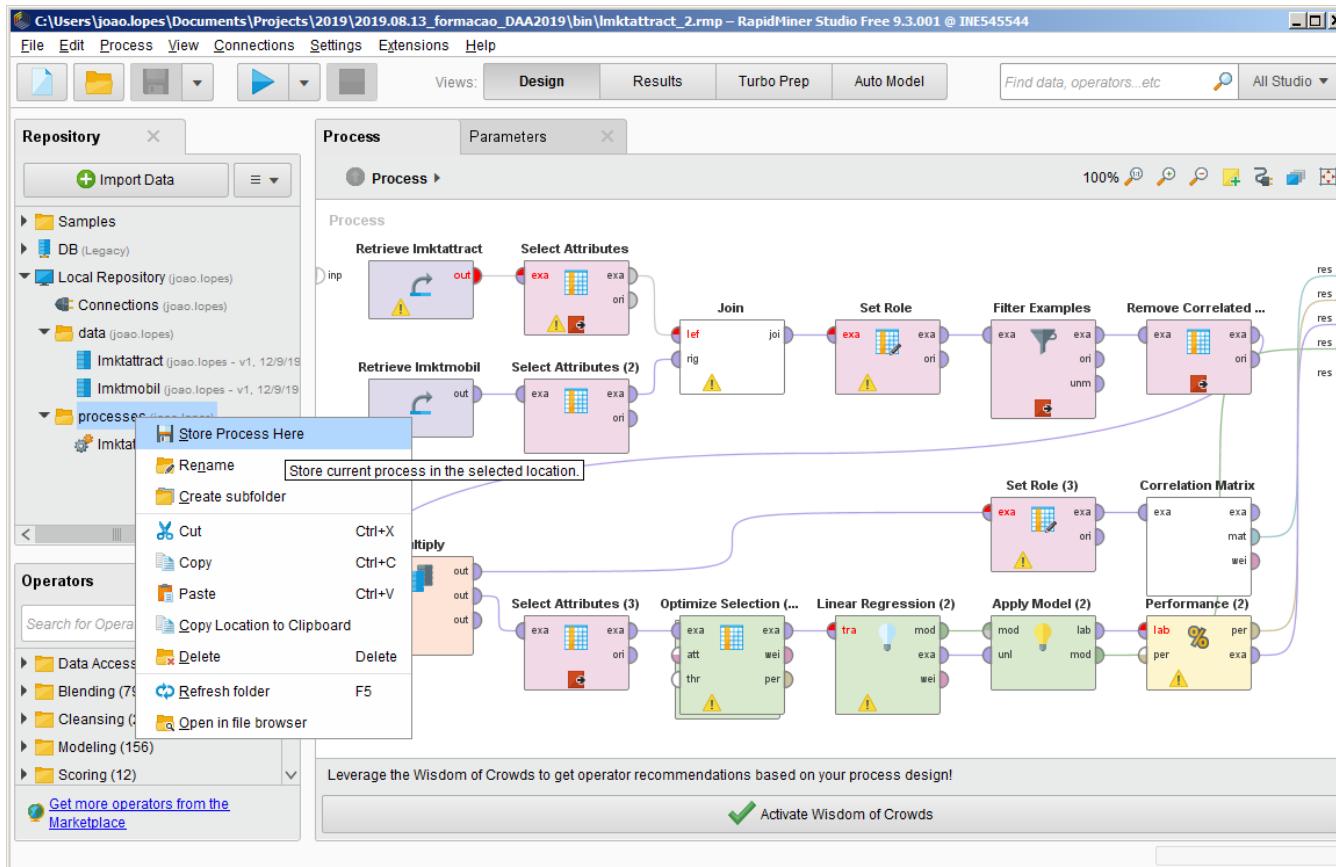
» Relação entre indicadores (no RapidMiner)



Exemplos Data Analytics



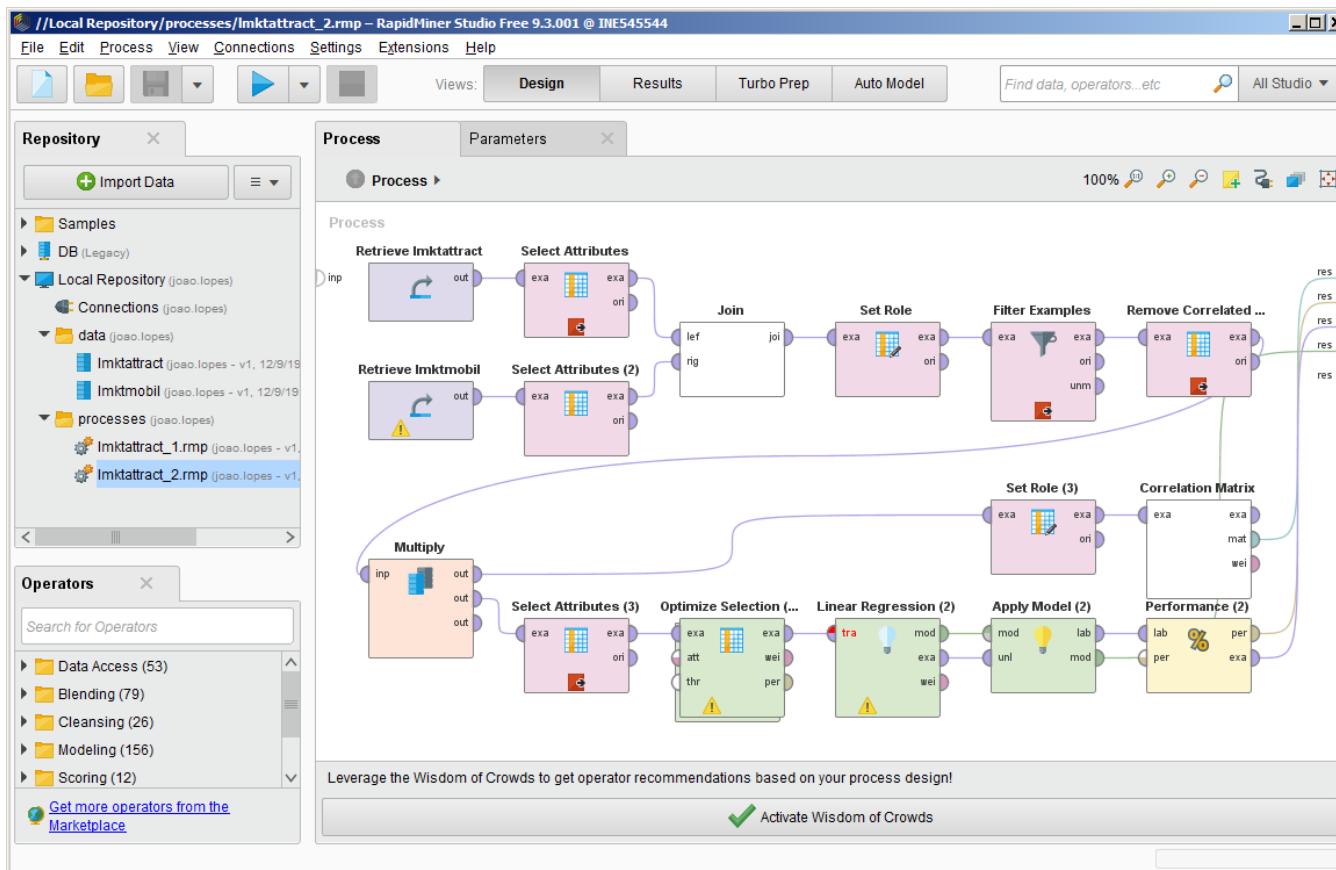
» Relação entre indicadores (no RapidMiner)



Exemplos Data Analytics



» Relação entre indicadores (no RapidMiner)





Exemplos *Data Analytics*



» Relação entre indicadores (no R)

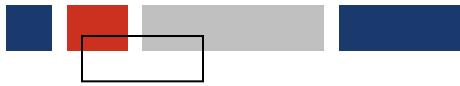
```
#1. FUNCTIONS  
  
source("misc_v2.3.r")  
  
#2. READ DATA  
  
f1 <- ".../data/lmktattract.csv"  
f2 <- ".../data/lmktmobil.csv"  
x_all <- read.table(file=f1,header=TRUE,sep=",",dec=". ",row.names=1)  
y_all <- read.table(file=f2,header=TRUE,sep=",",dec=". ",row.names=1)  
  
# 3. ANALYSE DATA  
  
set.seed(12345)  
  
#remove attributes with missing data  
ina <- apply(x_all,2,function(x){any(is.na(x))})  
x <- x_all[ ,!ina]
```



Exemplos *Data Analytics*



```
#remove data entries with missing data  
  
y <- y_all[, "lmktm_Total", drop=FALSE]  
ina <- is.na(y[, 1])  
x <- x[!ina, ]  
y <- y[!ina, , drop=FALSE]  
  
  
#reduce number of attributes  
  
f1 <- "../results/lmktattract_2/datred.log"  
x <- reduce_predictors_v2(x, y[, 1],  
  thr1=0.90,           #upper threshold for correlation between predictors  
  method="pearson",   #type of correlation  
  thr2=NULL,          #lower threshold for correlation between predictors and response  
  thr3=0.00,          #lower threshold for coefficient of variation  
  thr4=Inf,           #upper threshold for Variance Inflation Factor (VIF)  
  maxsize=30,         #maximum number of predictors  
  f1=f1)
```



Exemplos *Data Analytics*



```
#perform GA

f1 <- ".../results/lmktattract_2/modsel"

mod1 <- mod_select_lm_v2(x,y,
  maxs1=10,           #maximum number of attributes
  popsize = 100,      #population size
  mutrate = 0.001,    #per locus (i.e. per term) mutation rate, [0,1]
  sexrate = 0.1,      #sexual reproduction rate, [0,1]
  imm = 0.0,          #immigration rate, [0,1]
  deltaM=1e-6,        #Stop Rule: change in mean IC
  deltaB=1e-6,        #Stop Rule: change in best IC
  conseq = 5,          #Stop Rule: times with no improvement
  nreps = 4,           #number of repeats
  f1=f1)

#fit best model

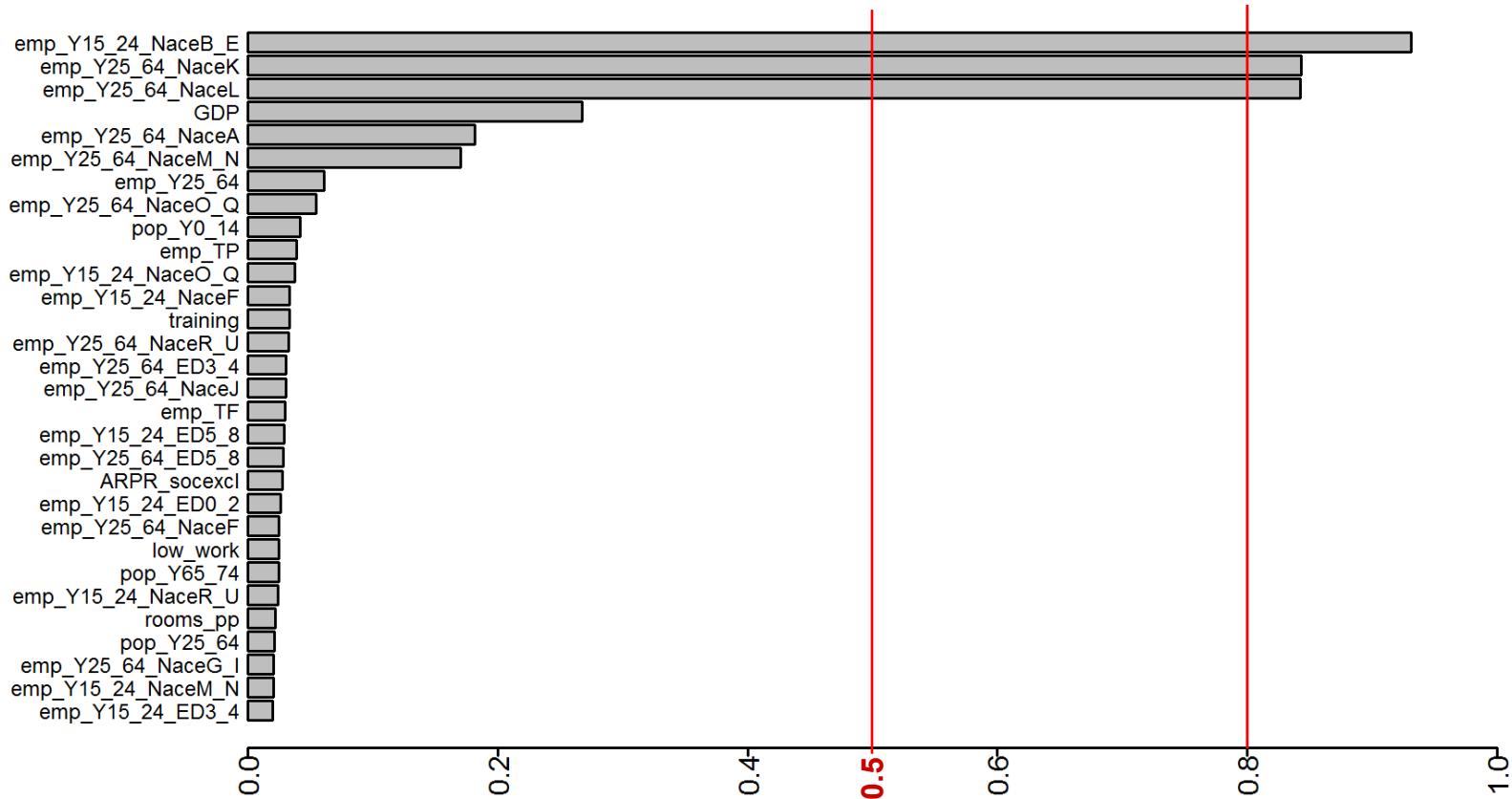
f1 <- ".../results/lmktattract_2/fit"
fit_lm(mod1$formula,mod1$data,f1=f1,main="LMkt Mobility")
```



Exemplos *Data Analytics*

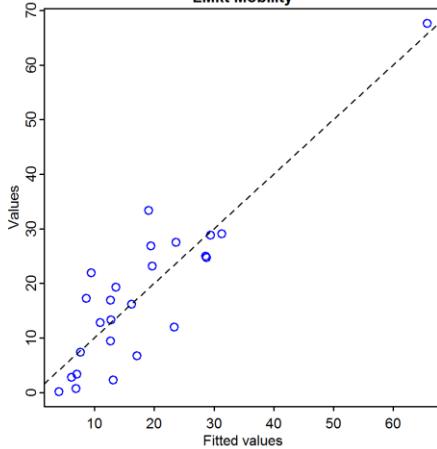
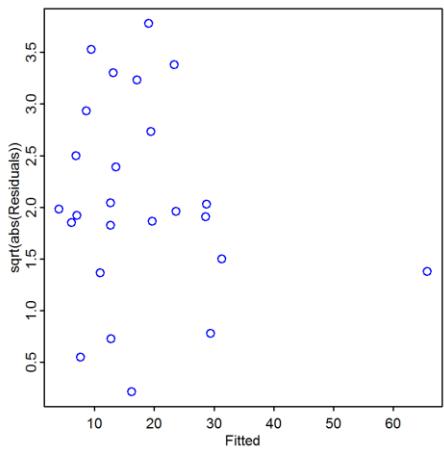
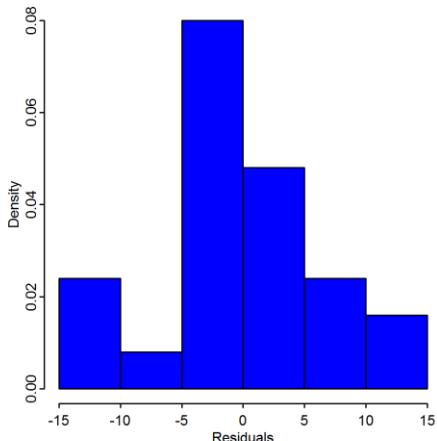
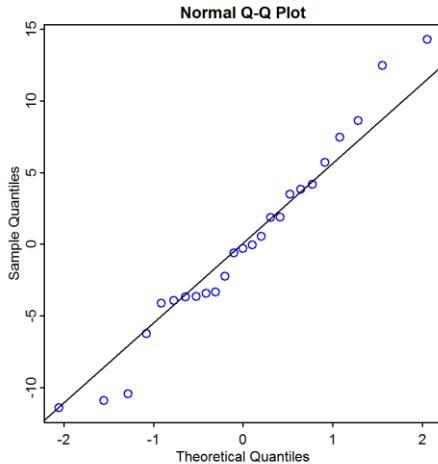


Model-averaged importance of terms





Exemplos Data Analytics



labels	value	st. err.	t value	Pr(> t)
(Intercept)	1.82	6.196	0.294	0.772
emp_Y15-24_NaceB-E	-0.38	0.198	-1.929	0.067
emp_Y25-64_NaceK	4.61	0.711	6.474	<0.001
emp_Y25-64_NaceL	10.04	2.832	3.474	<0.001

Adjusted-R² = 0.76; F(3, 21) = 25.85; p-value < 0.001.

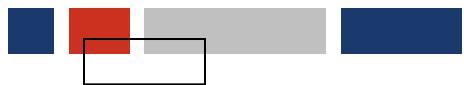


Exemplos *Data Analytics*



» Relação não-linear entre indicadores (no R)

```
#1. FUNCTIONS  
  
source("misc_v2.3.r")  
  
#2. ANALYSE DATA  
  
set.seed(12345)  
  
decay <- 1e-4                      #parameter for weight decay  
maxit <- 1000                      #maximum number of iterations  
abstol <- 1e-6                      #absolute fit criterion  
reltol <- 1e-12                     #relative fit criterion  
  
#fit best model (nn size = 1)  
size <- 1                            #number of units in the hidden layer  
f1 <- "../results/lmktattract_3/fit_nnet1"  
fit_nnet_v2(mod1$formula,mod1$data,nn_size=size,nn_decay=decay,nn_maxit=maxit,  
nn_abstol=abstol,nn_reltol=reltol,f1=f1,main="LMkt Mobility")
```



Exemplos *Data Analytics*

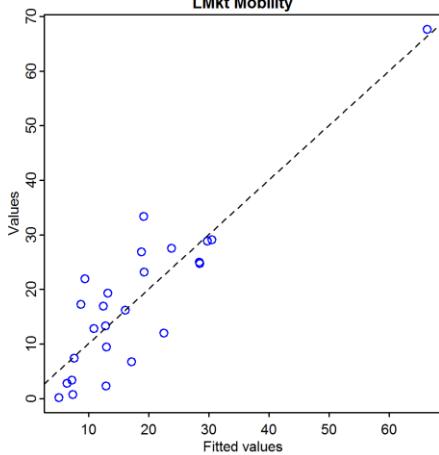
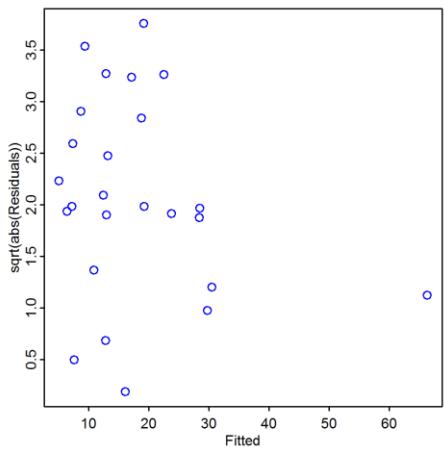
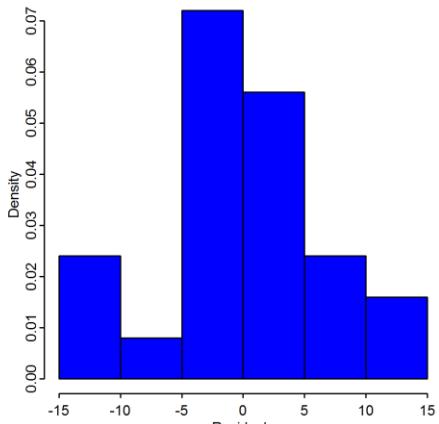
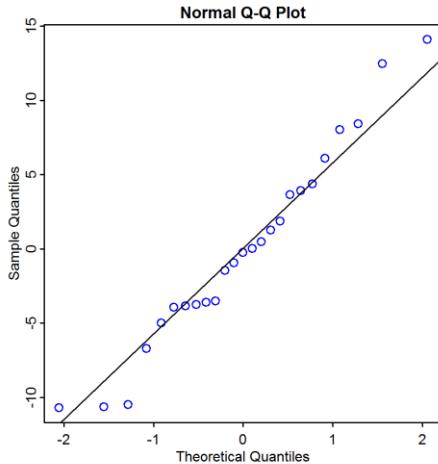


```
#fit best model (nn size = 3)
size <- 3                                #number of units in the hidden layer
f1 <- "../results/lmktattract_3/fit_nnet3"
fit_nnet_v2(mod1$formula,mod1$data,nn_size=size,nn_decay=decay,nn_maxit=maxit,
nn_abstol=abstol,nn_reltol=reltol,f1=f1,main="LMkt Mobility")

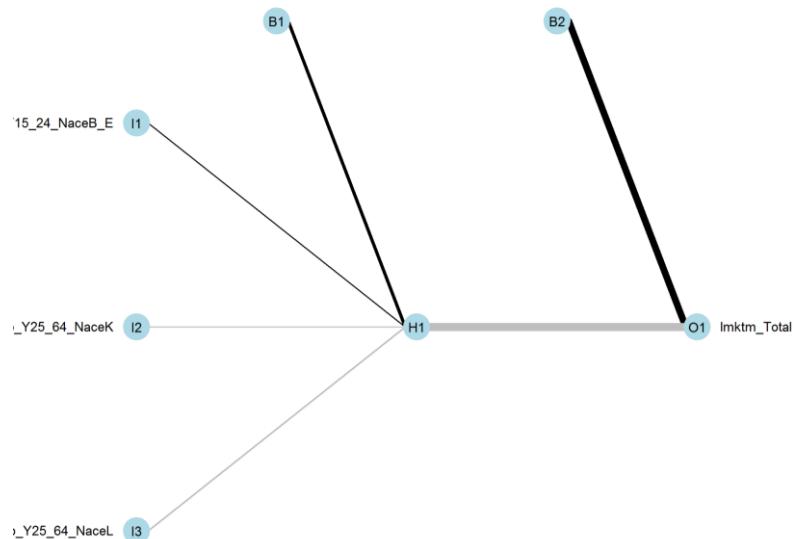
#fit best model (nn size = 10)
size <- 10                               #number of units in the hidden layer
f1 <- "../results/lmktattract_3/fit_nnet10"
fit_nnet_v2(mod1$formula,mod1$data,nn_size=size,nn_decay=decay,nn_maxit=maxit,
nn_abstol=abstol,nn_reltol=reltol,f1=f1,main="LMkt Mobility")
```



Exemplos Data Analytics

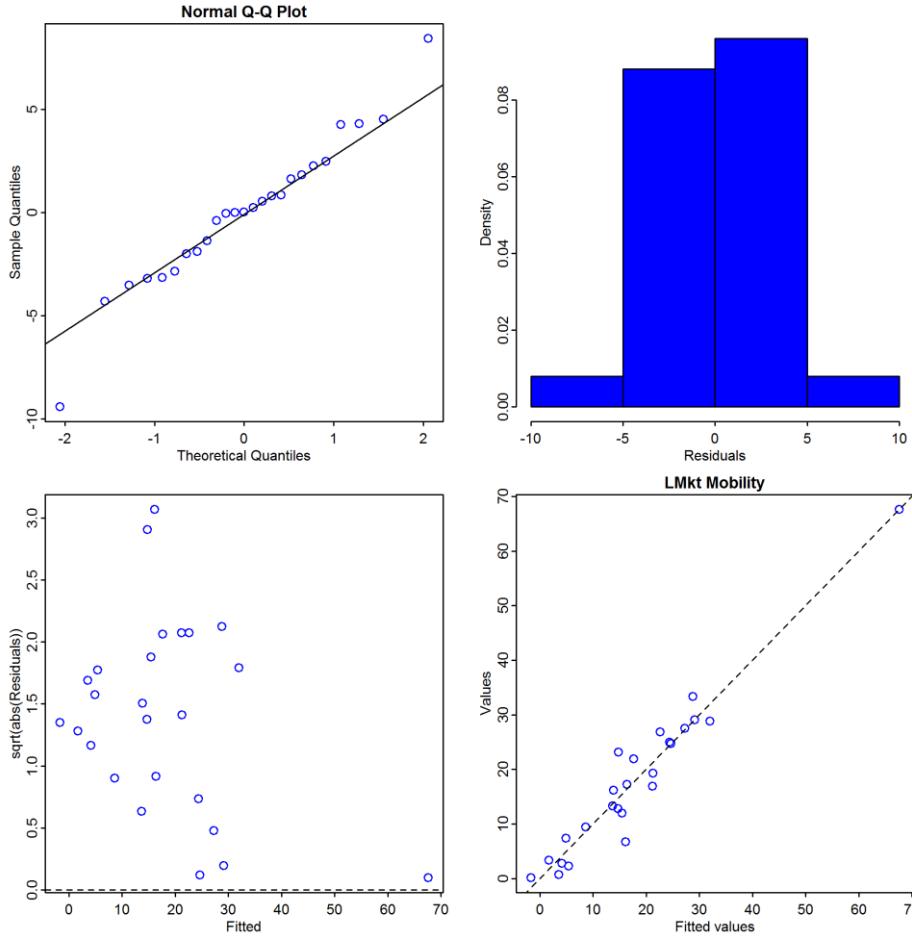


labels	Importance (Olsen, 2004)
emp_Y15-24_NaceB-E	-0.027
emp_Y25-64_NaceK	0.287
emp_Y25-64_NaceL	0.677

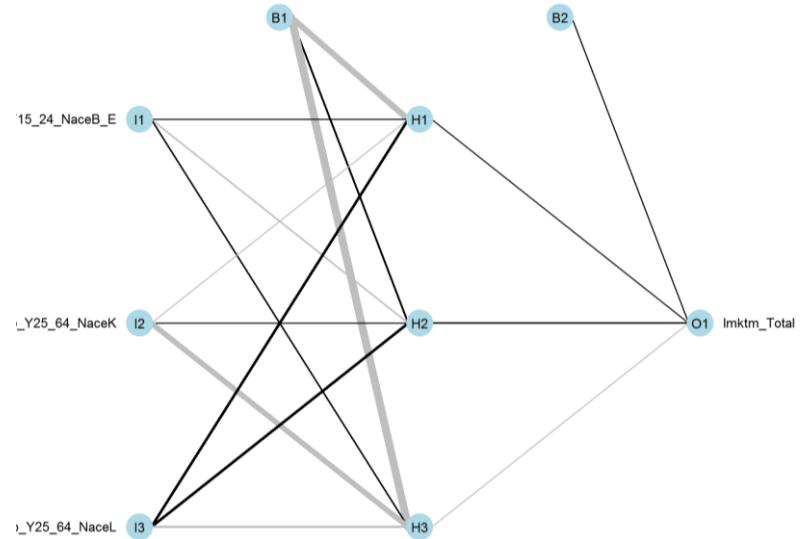




Exemplos Data Analytics

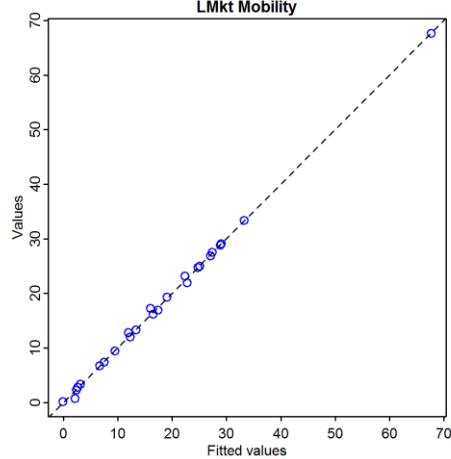
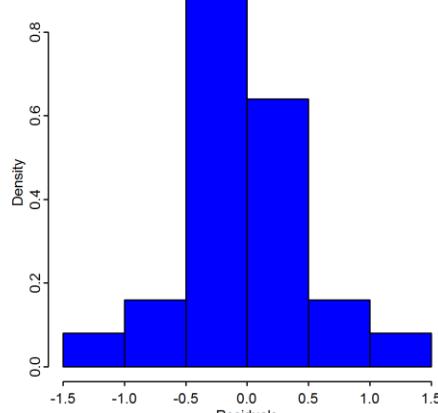
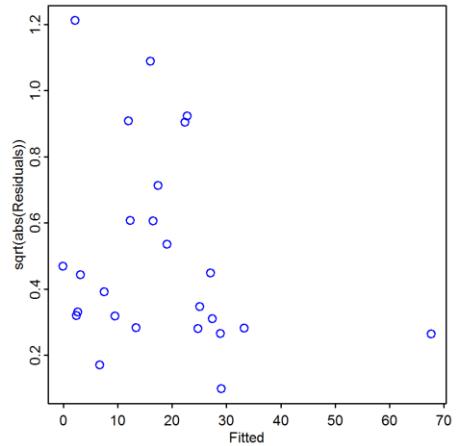
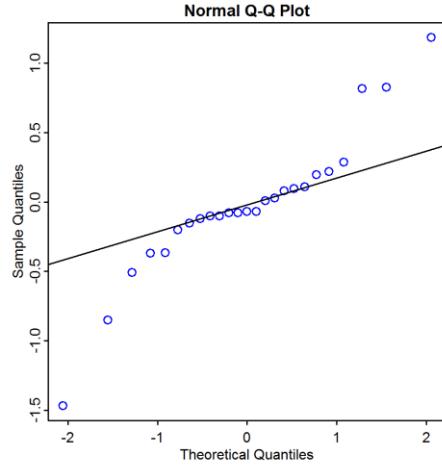


labels	Importance (Olsen, 2004)
emp_Y15-24_NaceB-E	-1.072
emp_Y25-64_NaceK	2.579
emp_Y25-64_NaceL	3.283

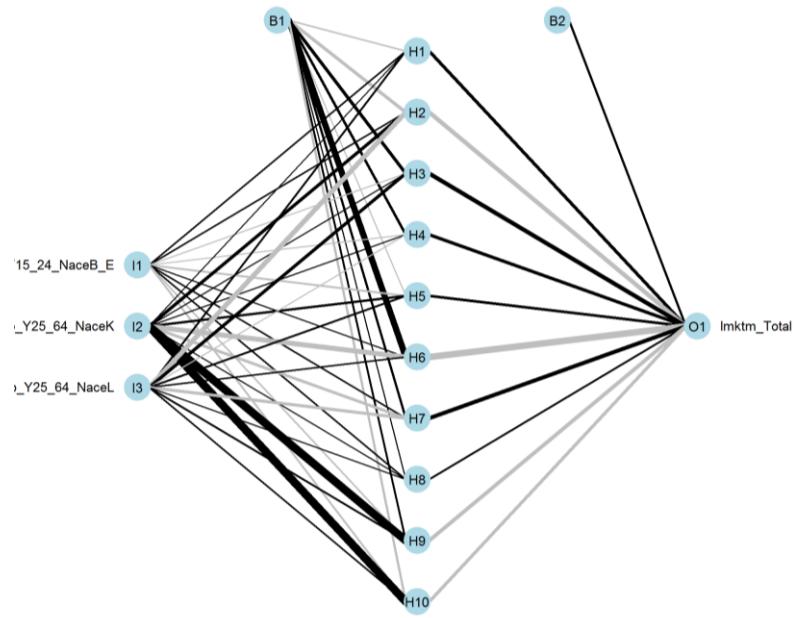




Exemplos Data Analytics



labels	Importance (Olsen, 2004)
emp_Y15-24_NaceB-E	0.440
emp_Y25-64_NaceK	-5.776
emp_Y25-64_NaceL	2.041





Gestão de projectos



» Boas práticas

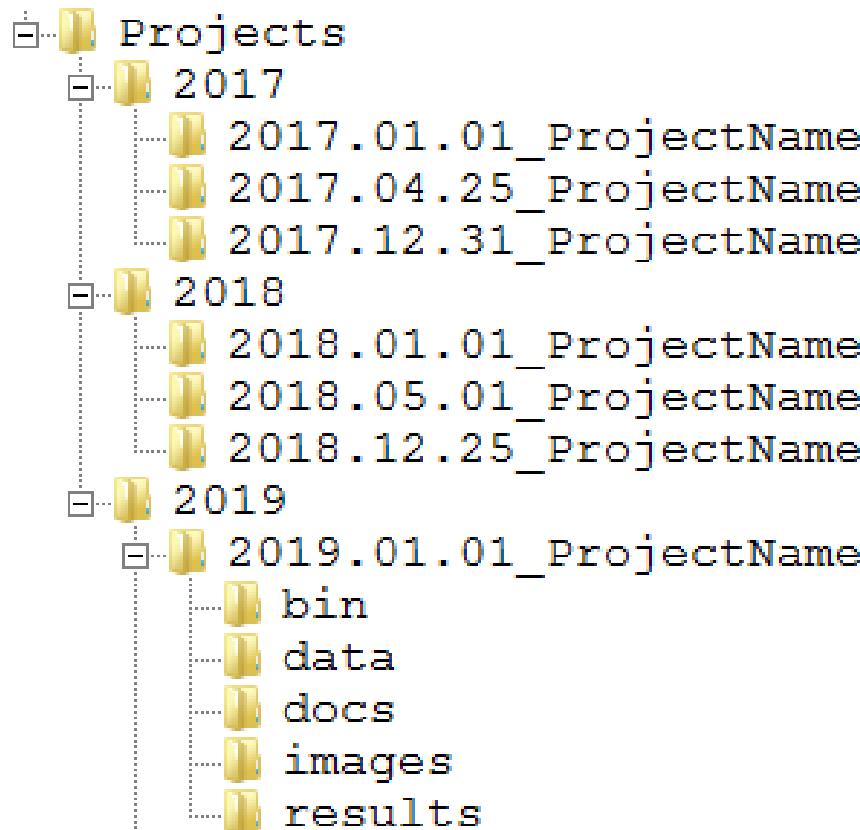
- Estrutura de pastas;
- Ficheiro README (e nomeação de ficheiros).

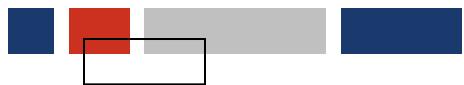


Gestão de projectos



» Estrutura de pastas





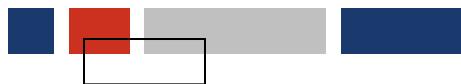
Gestão de projectos



» Ficheiro README

```
#autor:          Joao Sollari Lopes
#local:         INE, Lisboa
#criado:        03.12.2019
#modificado:   09.12.2019

+bin
  |decision_rule.r           #aplicar regras de decisao
  |decision_tree.r           #aplicar arvores de decisao
  |...
+data
  |golf.csv                  #dados "Golf" do RapidMiner
  |...
+docs
  |DAA2019_JL_practicals.pdf #exercicios [versao final]
  |...
+imagens
+results
  +lmktattract_1             #resultados da aplicacao pratica 1
  +lmktattract_2             #resultados da aplicacao pratica 2
  +lmktattract_3             #resultados da aplicacao pratica 3
README.txt                   #Este ficheiro
```



Bibliografia



- » Azzalini A & Scarpa B (2012) Data Analysis and Data Mining - An Introduction. Oxford University Press, New York.
- » Due KL & Swamy MNS (2016) Search and Optimization by Metaheuristics - Techniques and Algorithms Inspired by Nature. Springer, Switzerland.
- » Gama J, Carvalho APL, Faceli K, Lorena AC e Oliveira M (2017) Extração de Conhecimento de Dados. *Data Mining*. Edições Sílabo. Lisboa.
- » Larose DT & Larose CD (2014) Discovering Knowledge in Data – An Introduction to Data Mining. John Wiley & Sons, New Jersey.
- » Torgo L, (2017) Data Mining with R – Learning with Case Studies. Taylor & Francis Group, New York.
- » Wickham H & Grolemund G (2017) R for Data Science. O'Reilly Media Inc., Sevastopol.