

Algoritmos de Classificação

K-Vizinhos mais Próximos

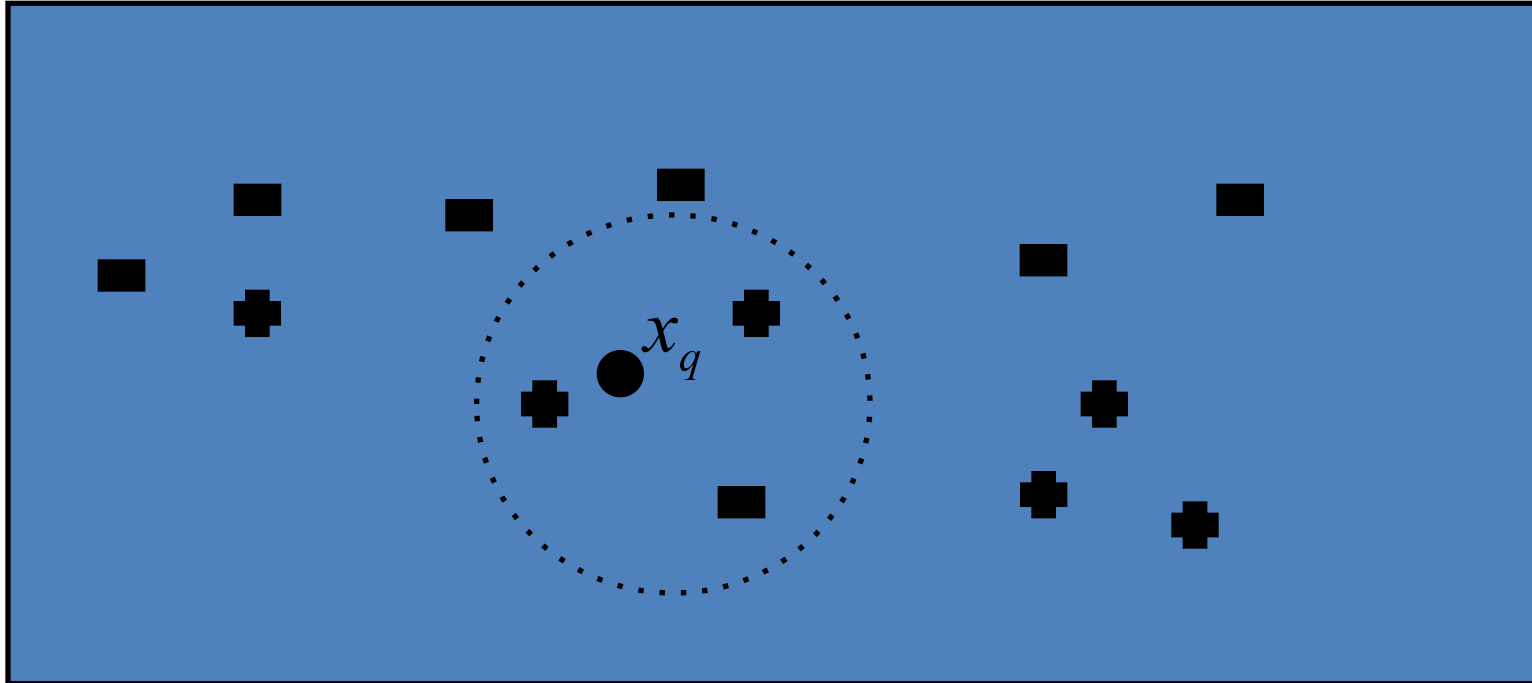
Algoritmo k-NN

- Todas as instâncias correspondem a *pontos* em um espaço n-dimensional
- Vizinhança definida por uma função de *distância*, ou por uma função de *similaridade*
 - Menor distância = maior similaridade
- Classe de um novo exemplo é definida a partir dos *vizinhos mais próximos*

Algoritmo k-NN

- Exemplo

Espaço de instâncias



- Exemplos da classe negativa
- ✚ Exemplos da classe positiva
- Exemplos a ser classificado

- Com $k = 3$, exemplo x_q recebe classe positiva

Algoritmo k-NN

- Algoritmo k-NN usa comumente a *Distância Euclidiana* para definição de vizinhança

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^n (a_r(x_i) - a_r(x_j))^2}$$

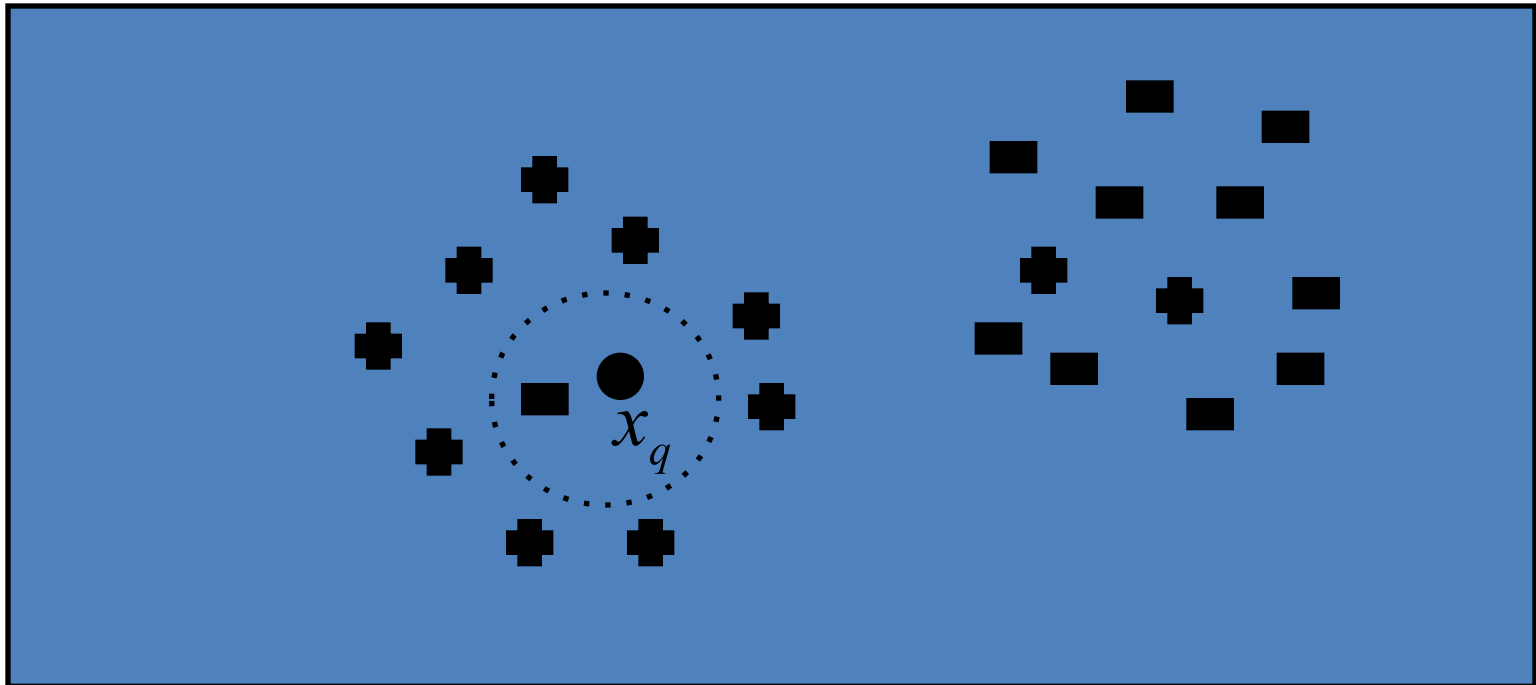
Algoritmo k-NN

- Atributos de maior *escala numérica* podem dominar função de distância
 - Usualmente, os atributos são normalizados para intervalo entre 0 e 1

$$a_{NORM}(x) = \frac{a(x) - \min_i(a(x_i))}{\max_i a(x_i) - \min_i(a(x_i))}$$

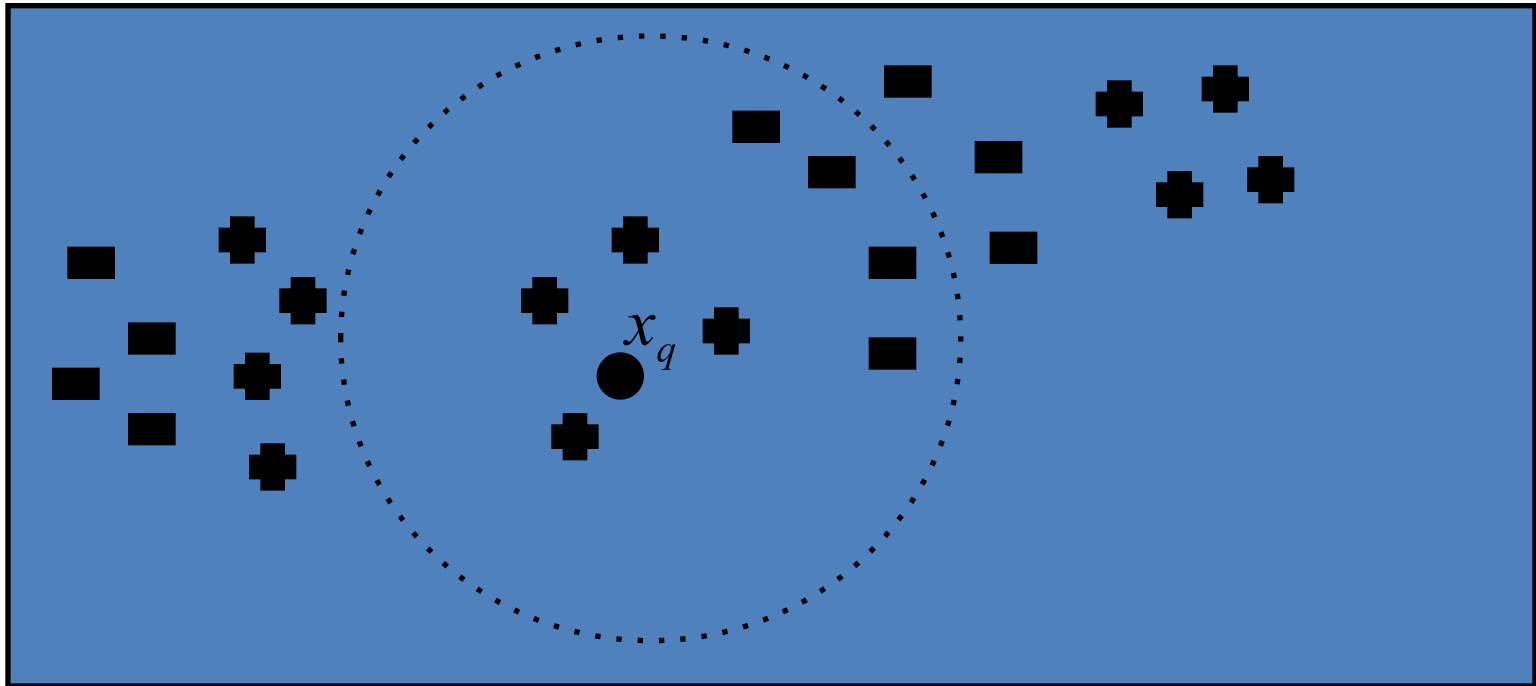
Algoritmo k-NN

- O *dilema* da escolha do parâmetro k
 - Valores muito baixos podem aumentar a contribuição de *exemplos ruidosos*



Algoritmo k-NN

- O *dilema* da escolha do parâmetro k
 - Valores muito altos podem aumentar a contribuição de exemplos *pouco similares*, e assim, *menos relevantes*



Algoritmo k-NN

- O valor do parâmetro k é escolhido comumente através de *tentativa-e-erro*
 - Avaliação empírica com diferentes valores de k
 - *Validação cruzada*

Algoritmo k-NN

- Discussão

- K-NN é um *método lazy*
 - I.e., não gera um modelo durante o treinamento
 - *Métodos eager*, como as árvores de decisão, geram modelos de dados
 - Consequências para o k-NN:
 - Treinamento rápido
 - Resposta lenta durante uso

Algoritmo k-NN

- Discussão

- Vantagens
 - É capaz de gerar boas respostas mesmo com poucos exemplos de treinamento
 - Algoritmos, como árvores de decisão, precisam de mais dados para gerar um bom modelo
 - Fácil de implementar

Algoritmo k-NN

- Discussão

- Desvantagens

- É muito sensível a presença de atributos irrelevantes e/ou redundantes

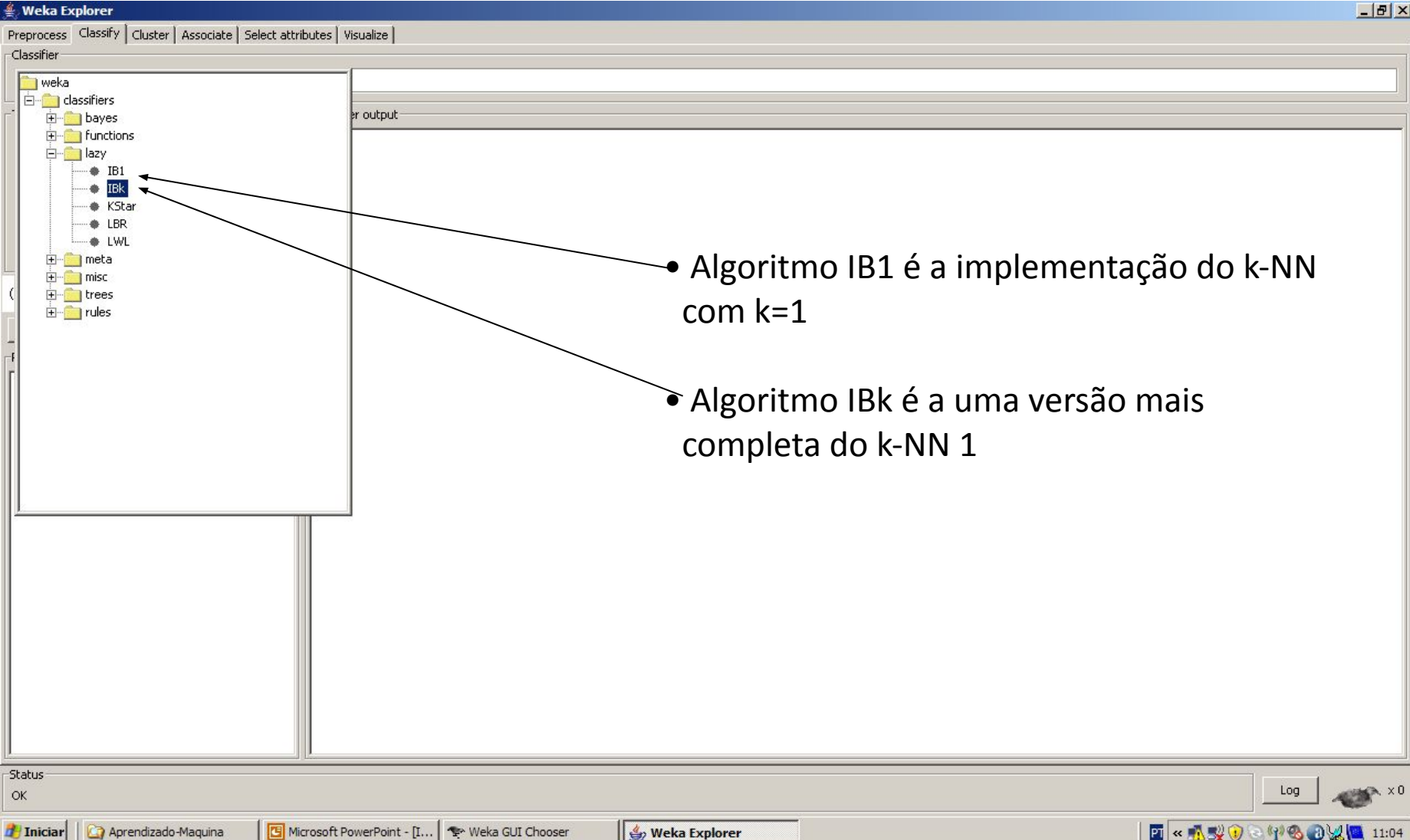
- *Curse of Dimensionality*

- Tempo de resposta em alguns contextos é impraticável

- Reduzir o número de exemplos de treinamento pode amenizar esse problema

- *Algoritmos baseados em protótipos* também podem ajudar

Algoritmo k-NN no WEKA



Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier

- weka
 - classifiers
 - bayes
 - functions
 - lazy
 - IB1
 - IBk
 - KStar
 - LBR
 - LWL
 - meta
 - misc
 - trees
 - rules

- Algoritmo IB1 é a implementação do k-NN com $k=1$
- Algoritmo IBk é a uma versão mais completa do k-NN 1

Status
OK

Log

Weka Explorer

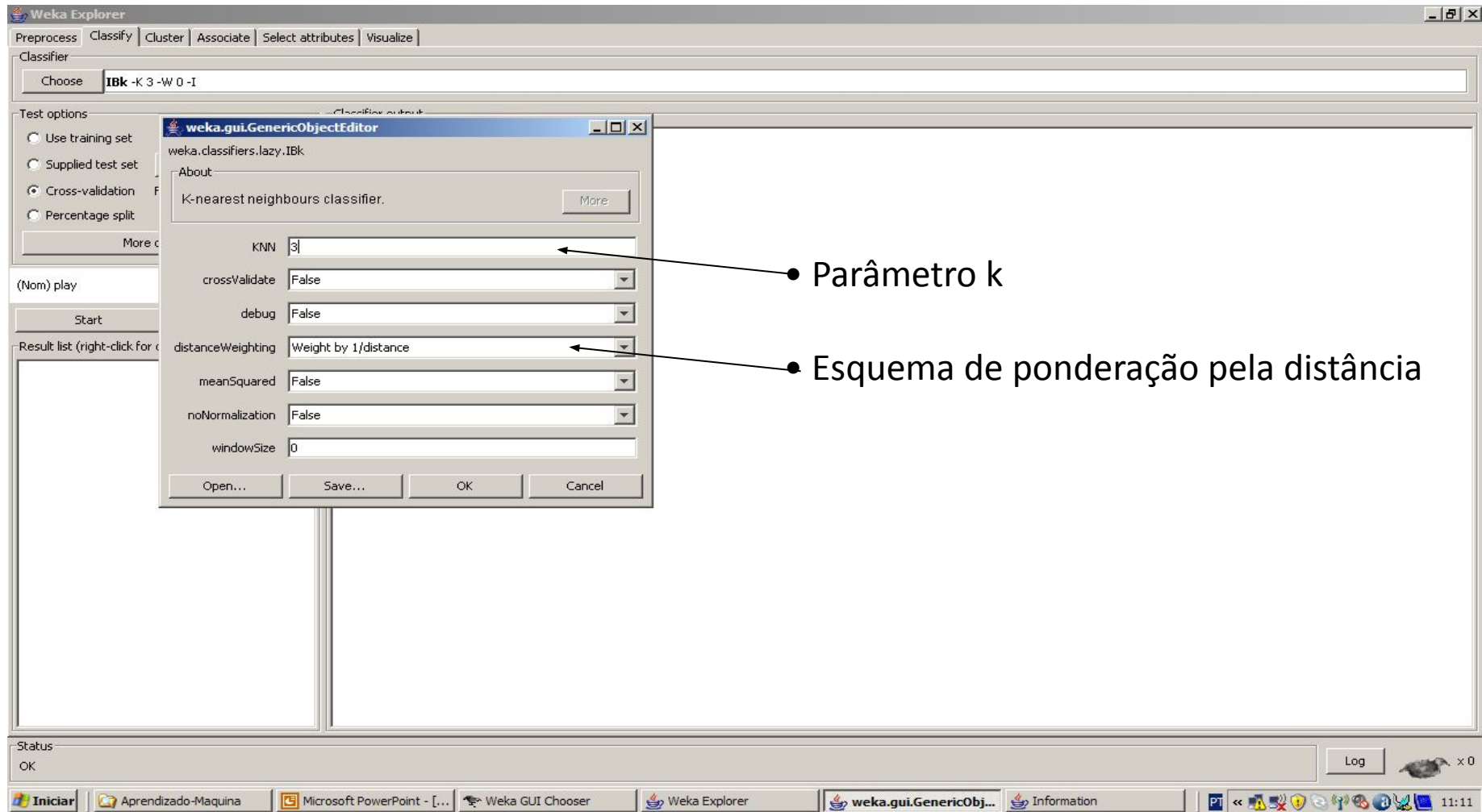
Microsoft PowerPoint - [I...]

Aprendizado-Maquina

Iniciar

11:04

Algoritmo k-NN no WEKA



Algoritmos de Classificação

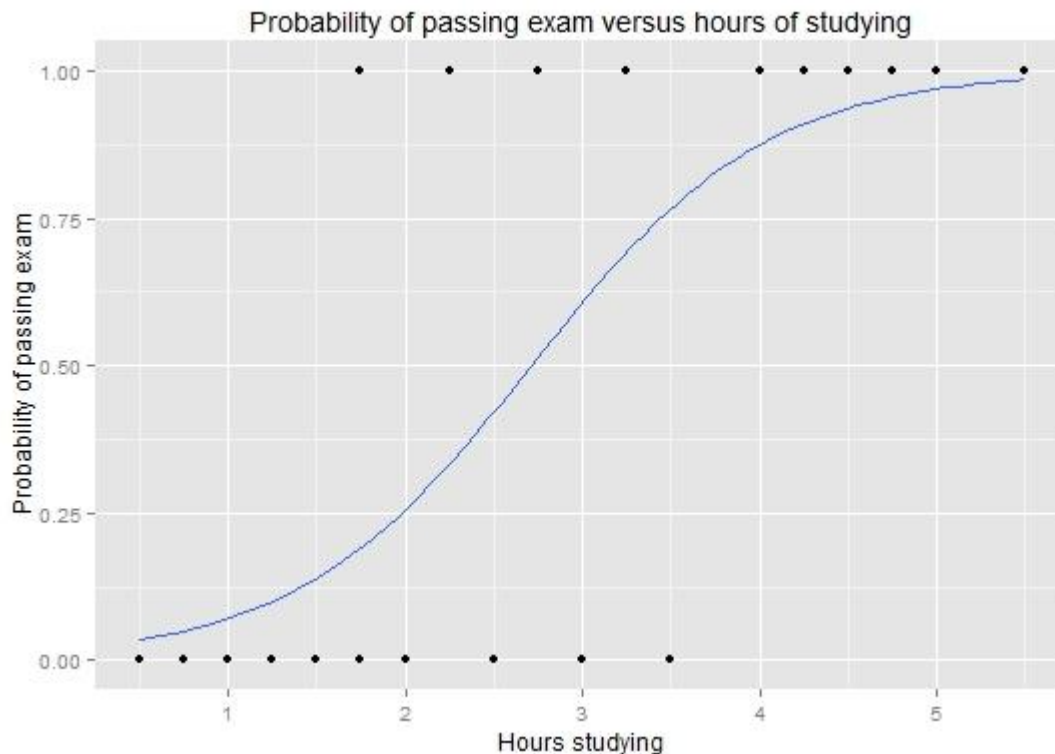
Regressão Logística

Regressão Logística

- Retorna a **probabilidade** de classe dado o conjunto de **variáveis dependentes**
- Retorna um modelo interpretável onde cada atributo preditor é associado a um **peso numérico (importância do atributo)**

Regressão Logística

- Exemplo: https://en.wikipedia.org/wiki/Logistic_regression



$$P(\text{Pass} \mid \text{Hours}) = \frac{1}{1 + \exp(-(-4.07 + 1.50\text{Hours}))}$$

Regressão Logística - Modelo

$$P(Y = 1 | X) = \frac{1}{1 + \exp(-(\beta + \alpha X))}$$

$$P(Y = 1 | x_1, \dots, x_p) = \frac{1}{1 + \exp(-(\beta + \alpha_1 x_1 + \dots + \alpha_p x_p))}$$



Importância da variável

Regressão Logística - WEKA



The screenshot displays the Weka Explorer application window. The 'Classify' tab is active, and the 'Logistic -R 1.0E-8 -M -1' classifier is selected. The 'Test options' section on the left shows 'Percentage split' at 66%. The 'Classifier output' pane on the right displays the results of the classification, including the classifier model and a list of coefficients for various features.

Classifier output

=== Classifier model (full training set) ===

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...

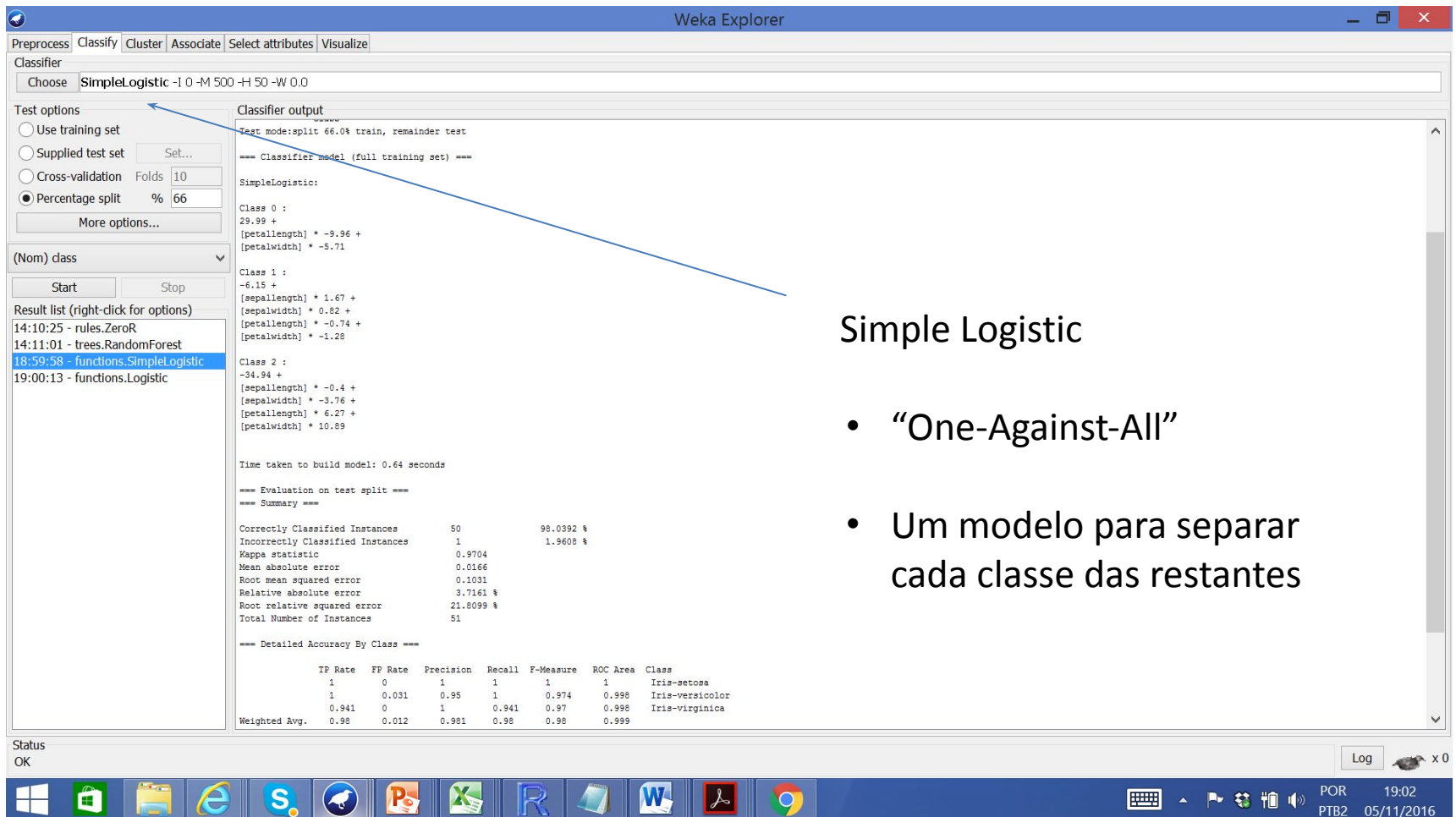
Variable	Class
checking_status=<0	-0.778
checking_status=0<=X<200	-0.4032
checking_status=>=200	0.1877
checking_status=no checking	0.9338
duration	-0.0279
credit_history=no credits/all paid	-0.0129
credit_history=all paid	-0.9562
credit_history=existing paid	-0.2268
credit_history=delayed previously	0.0403
credit_history=critical/other existing credit	0.6229
purpose=new car	-0.692
purpose=used car	0.9744
purpose=furniture/equipment	0.0996
purpose=radio/tv	0.1996
purpose=domestic appliance	-0.1692
purpose=repairs	-0.4756
purpose=education	-0.7283
purpose=vacation	0
purpose=retraining	1.3674
purpose=business	0.0481
purpose=other	0.7968
credit_amount	-0.0001
savings_status=<100	-0.4402
savings_status=100<=X<500	-0.0825
savings_status=500<=X<1000	-0.0641
savings_status=>=1000	0.8989
savings_status=no known savings	0.5065
employment=unemployed	-0.2934
employment=<1	-0.2265
employment=1<=X<4	-0.1106
employment=4<=X<7	0.5376
employment=>=7	-0.0168
installment_commitment	-0.3301
personal_status=male div/sep	-0.4923
personal_status=female div/dep/mar	-0.2168
personal_status=male single	0.3238
personal_status=male mar/wid	-0.1252
personal_status=female single	0
other_parties=none	-0.1798

Status: OK

Log x 0

POR 19:15
PTB 05/11/2016

Regressão Logística – WEKA - Problemas Multi-Classe



The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The 'Simple Logistic' classifier is chosen, and the 'Percentage split' is set to 66%. The 'Classifier output' pane displays the results for three classes: Class 0, Class 1, and Class 2. The output includes the model equations, evaluation statistics, and a detailed accuracy table.

Classifier output

Test mode: split 66.0% train, remainder test

=== Classifier model (full training set) ===

SimpleLogistic:

Class 0 :
29.99 +
[petallength] * -9.96 +
[petalwidth] * -5.71

Class 1 :
-6.15 +
[sepalwidth] * 1.67 +
[sepalwidth] * 0.82 +
[petallength] * -0.74 +
[petalwidth] * -1.28

Class 2 :
-34.94 +
[sepalwidth] * -0.4 +
[sepalwidth] * -3.76 +
[petallength] * 6.27 +
[petalwidth] * 10.89

Time taken to build model: 0.64 seconds

=== Evaluation on test split ===

=== Summary ===

Correctly Classified Instances	50	98.0392 %
Incorrectly Classified Instances	1	1.9608 %
Kappa statistic	0.9704	
Mean absolute error	0.0166	
Root mean squared error	0.1031	
Relative absolute error	3.7161 %	
Root relative squared error	21.8099 %	
Total Number of Instances	51	

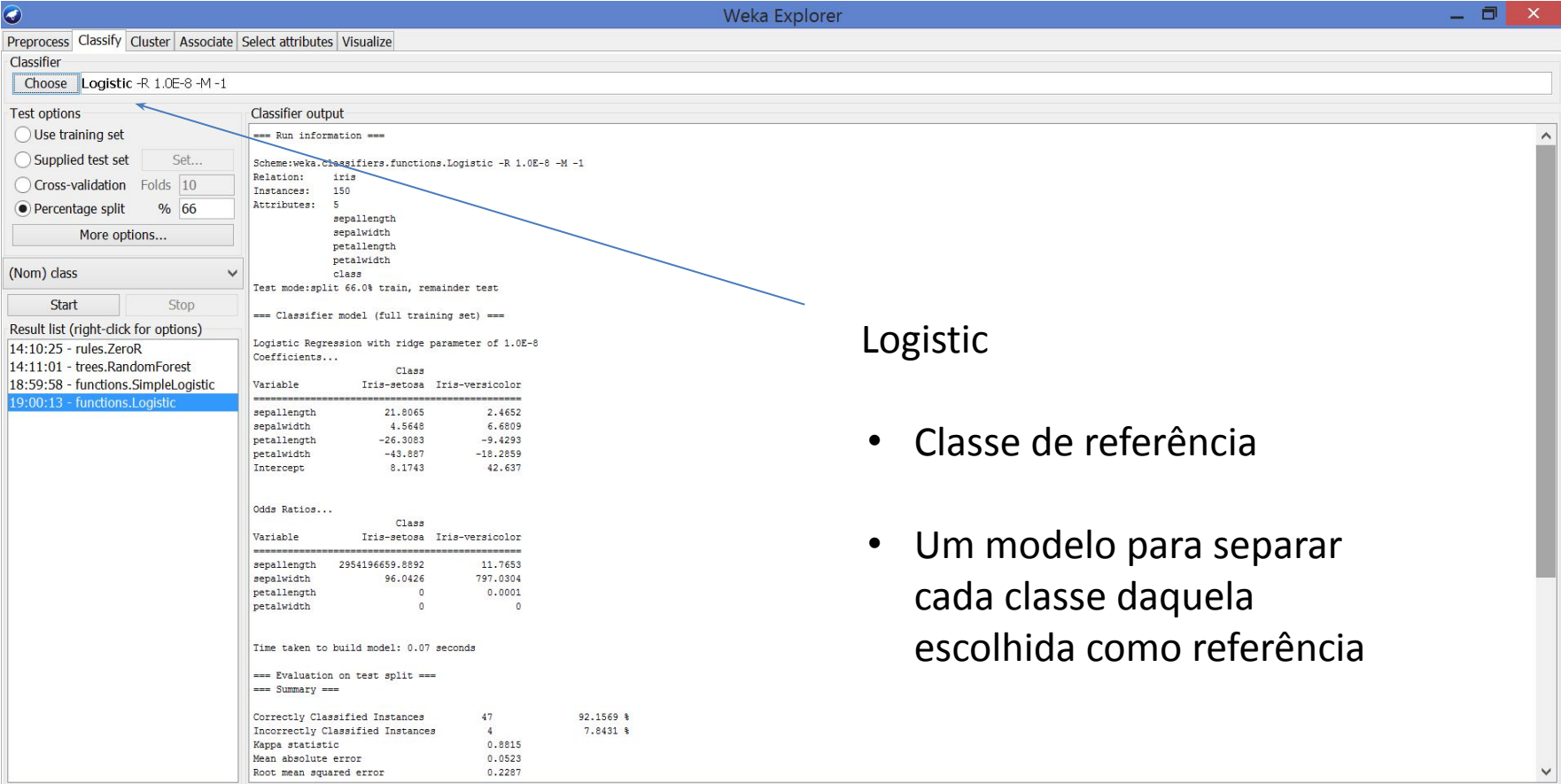
=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
1	0	1	1	1	1	1	Iris-setosa
1	0.031	0.95	1	0.974	0.998		Iris-versicolor
0.941	0	1	0.941	0.97	0.998		Iris-virginica
Weighted Avg.	0.98	0.012	0.981	0.98	0.98	0.999	

Simple Logistic

- “One-Against-All”
- Um modelo para separar cada classe das restantes

Regressão Logística – WEKA - Problemas Multi-Classe



The screenshot shows the Weka Explorer window with the 'Classify' tab selected. The 'Classifier' dropdown is set to 'Logistic -R 1.0E-8 -M -1'. The 'Test options' section shows 'Percentage split' at 66%. The 'Classifier output' pane displays the following information:

==== Run information ====

Scheme: weka.classifiers.functions.Logistic -R 1.0E-8 -M -1
Relation: iris
Instances: 150
Attributes: 5
sepalwidth
sepalwidth
petalwidth
petalwidth
class

Test mode: split 66.0% train, remainder test

==== Classifier model (full training set) ====

Logistic Regression with ridge parameter of 1.0E-8

Coefficients...

Variable	Iris-setosa	Iris-versicolor
sepalwidth	21.8065	2.4652
sepalwidth	4.5648	6.6809
petalwidth	-26.3083	-9.4293
petalwidth	-43.887	-18.2859
Intercept	8.1743	42.637

Odds Ratios...

Variable	Iris-setosa	Iris-versicolor
sepalwidth	2954196659.8892	11.7653
sepalwidth	96.0426	797.0304
petalwidth	0	0.0001
petalwidth	0	0

Time taken to build model: 0.07 seconds

==== Evaluation on test split ====

==== Summary ====

Correctly Classified Instances	47	92.1569 %
Incorrectly Classified Instances	4	7.8431 %
Kappa statistic	0.8815	
Mean absolute error	0.0523	
Root mean squared error	0.2287	

The 'Result list' on the left shows the following entries:

- 14:10:25 - rules.ZeroR
- 14:11:01 - trees.RandomForest
- 18:59:58 - functions.SimpleLogistic
- 19:00:13 - functions.Logistic

The 'Logistic' entry is selected.

Logistic

- Classe de referência
- Um modelo para separar cada classe daquela escolhida como referência

Status: OK

Log x 0

POR PTB2 19:00 05/11/2016

Algoritmos de Classificação

Naive Bayes

Aprendizagem Bayesiana

- Fornece probabilidades para suas respostas
- Permite combinar facilmente conhecimento a priori com dados de treinamento
- Métodos práticos e bem sucedidos para aprendizagem
 - Aprendizagem Bayesiana Ingênua
 - Aprendizagem de Redes Bayesianas

Teoria da Probabilidade

- Associa às sentenças um grau de crença numérico entre 0 e 1
 - Contudo, cada sentença ou é **verdadeira** ou é **falsa**
- Grau de crença (probabilidade):
 - a priori (incondicional): calculado antes do agente receber percepções
 - ◇ Ex. $P(\text{cárie} = \text{true}) = P(\text{cárie}) = 0.5$
 - condicional: calculado de acordo com as **evidências** disponíveis (permite a **inferência**)
 - ◇ **evidências**: percepções que o agente recebeu até agora
 - ◇ Ex: $P(\text{cárie} | \text{dor de dente}) = 0.8$
 $P(\text{cárie} | \sim \text{dor de dente}) = 0.3$

Probabilidade Condicional

- Regra do produto:
 - $P(A|B) = \frac{P(A \cap B)}{P(B)}$, quando $P(B) > 0$.

- Regra de Bayes
 - $$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

considerando que $P(B|A)$
 $= \frac{P(A \cap B)}{P(A)}$

Aplicação da Regra de Bayes: Diagnóstico Médico



•Seja

M=doença
meningite

S= rigidez no
pescoço

•Um Doutor sabe:

$$P(S/M)=0.5$$

$$P(M)=1/50000$$

$$P(S)=1/20$$



$$P(M/S)=\frac{P(S/M)P(M)}{P(S)}$$

$$P(S)$$

$$=\frac{0,5*(1/50000)}{1/20}=0,002$$

•A probabilidade de uma pessoa ter meningite dado que ela está com rigidez no pescoço é 0,02% ou ainda 1 em 5000.

Classificador Bayesiano Ingênuo

- Suponha uma função de classificação $f: X \rightarrow V$, onde cada instância x é descrita pelos atributos $\{a_1, \dots, a_n\}$
- O valor mais provável de $f(x)$ é

$$V_{NB} = \mathbf{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i / v_j)$$

Classificador Bayesiano Ingênuo: Exemplo

• Dia	Tempo	Temp.	Humid.	Vento	Jogar
• D1	Sol	Quente	Alta	Fraco	Não
• D2	Sol	Quente	Alta	Forte	Não
• D3	Coberto	Quente	Alta	Fraco	Sim
• D4	Chuva	Normal	Alta	Fraco	Sim
• D5	Chuva	Frio	Normal	Fraco	Não
• D6	Chuva	Frio	Normal	Forte	Não
• D7	Coberto	Frio	Normal	Forte	Sim
• D8	Sol	Normal	Alta	Fraco	Não
• D9	Sol	Frio	Normal	Fraco	Sim
• D10	Chuva	Normal	Normal	Fraco	Sim
• D11	Sol	Frio	Alta	Forte	?

$$P(\text{Sim}) = 5/10 = 0.5$$

$$P(\text{Não}) = 5/10 = 0.5$$

$$P(\text{Sol/Sim}) = 1/5 = 0.2$$

$$P(\text{Sol/Não}) = 3/5 = 0.6$$

$$P(\text{Frio/Sim}) = 2/5 = 0.4$$

$$P(\text{Frio/Não}) = 2/5 = 0.4$$

$$P(\text{Alta/Sim}) = 2/5 = 0.4$$

$$P(\text{Alta/Não}) = 3/5 = 0.6$$

$$P(\text{Forte/Sim}) = 1/5 = 0.2$$

$$P(\text{Forte/Não}) = 2/5 = 0.4$$

$$P(\text{Sim})P(\text{Sol/Sim})P(\text{Frio/Sim})P(\text{Alta/Sim})P(\text{Forte/Sim}) = 0.0032$$

$$P(\text{Não})P(\text{Sol/Não})P(\text{Frio/Não})P(\text{Alta/Não})P(\text{Forte/Não}) = 0.0288$$

⇒ Jogar_Tenis (D11) = Não

Material de Estudo

- I. Witten, E. Frank, 2000. *Data Mining – Practical Machine Learning Tools and Techniques with Java Implementations*.
- T. Mitchell, 1997. *Machine Learning*.
- D. Aha, D. Kibler, M. Albert, ,1991. Instance-based learning algorithms. *Machine Learning*, 6:37--66.
- C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition.
- S. Gunn, Support Vector Machines for Classification and Regression.