Extracção de Conhecimento

João Gama
LIAAD, FEP – Universidade do Porto
jgama@liaad.up.pt

Road Map

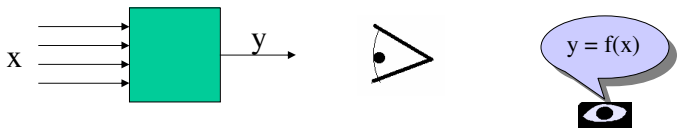
1. Aprendizagem Bayesiana
   1. Naive Bayes
   2. Funções Discriminantes
   3. SVM
2. Arvores de Decisão
   1. Construção
   2. Regularização
3. Modelos Multiplos
4. Fluxos de Dados
   1. Árvores de decisão para *streams*
   2. Detecção de Mudança

Aprendizagem Automática

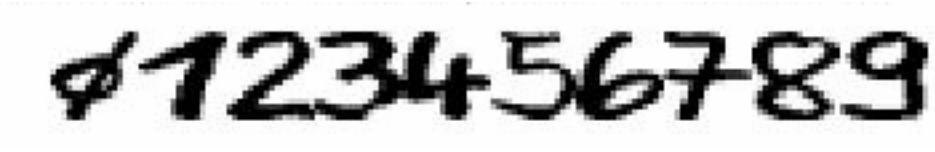
- Áreas disciplinares
  - Estatística
    - Inferência estatística
  - Computação
    - Inteligência Artificial
      - Aprendizagem Automática
  - Bases de dados
    - Bases de Dados Multidimensionais
- Definições:
  - "*Self-constructing or self-modifying representations of what is being experienced for possible future use*" Michalski, 1990
  - "*Analysis of observational data to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful for the data owner*" Hand, Mannila, Smyth, 2001
  - Obter representações em compreensão a partir de representações em extensão.

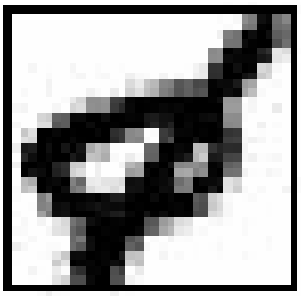
Aplicações

- Códigos Postais
- Predição do uso da terra
- Aprender a conduzir veículos autónomos
- Web sites Adaptativos.



Códigos Postais





Predição do Uso da Terra

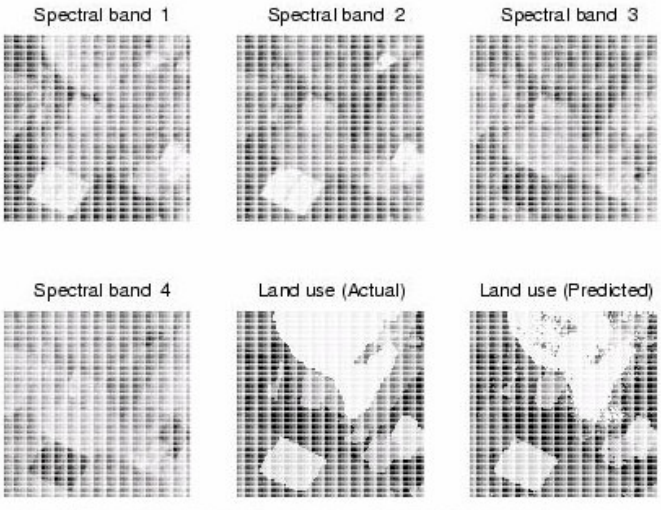
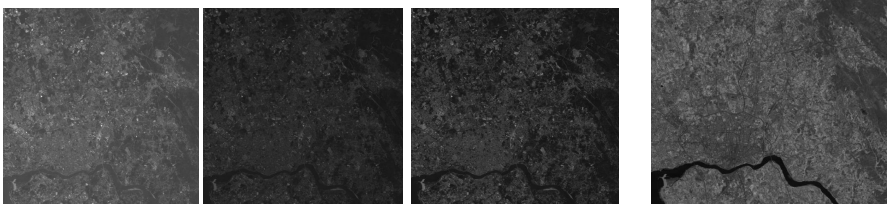
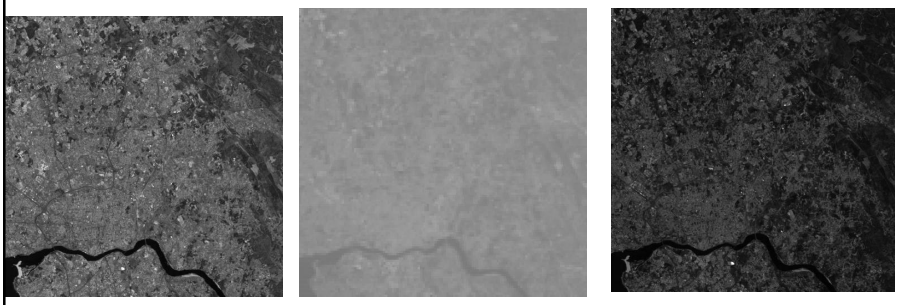


Fig. 9.3: Satellite image dataset. Spectral band intensities as seen from a satellite for a small (8.2*6.6 km) region of Australia. Also given are the actual land use as determined by on-site visit and the estimated classes as given by linear discriminants.

LandSat - Porto



Verde-azul vermelho vermelho



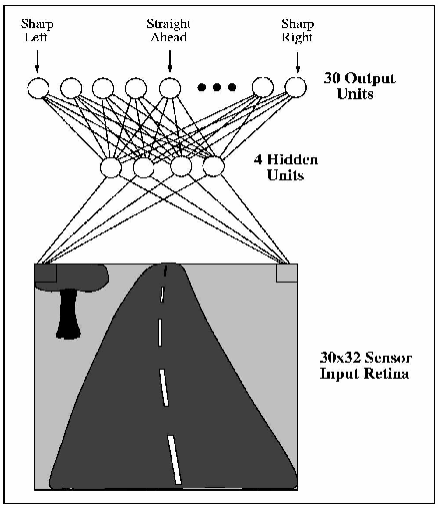
(Infravermelhos) Pixel:30mx30m

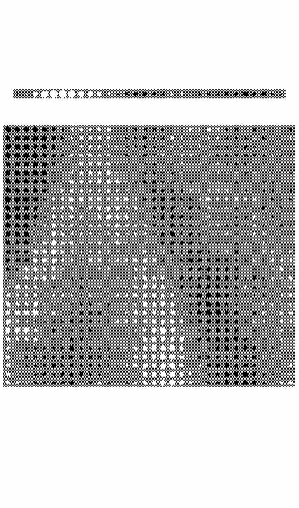
J. Gama 7

Veículos Autónomos

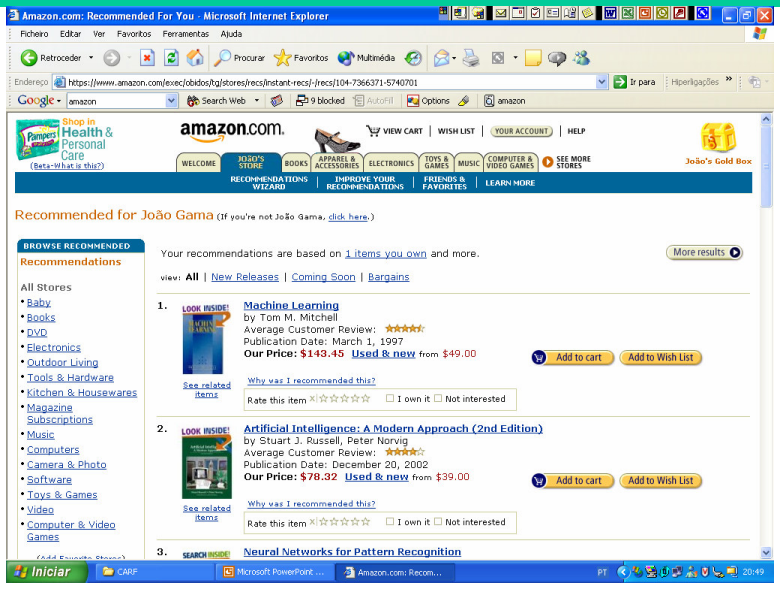
J. Gama 8

Veículos Autónomos





Web sites adaptativos

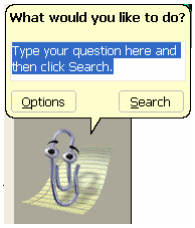


Help Systems!

The “Clip” is an interface for a Bayesian Network

“ERIC HORVITZ, a researcher at Microsoft and a guru in the field of Bayesian statistics, feels bad about the paperclip-but he hopes his latest creation will make up for it. The paperclip in question, as even casual users of Microsoft's Office software will be aware, is a cheery character who pops up on the screen to offer advice on writing a letter or formatting a spreadsheet. That was the idea, anyway. But many people regard the paperclip as annoyingly over-enthusiastic, since it appears without warning and gets in the way.”

“Mobile Manager evaluates incoming e-mails on a user's PC and decides which are important enough to forward to a pager, mobile phone or other e-mail address. Its Bayesian innards give it an almost telepathic ability to distinguish junk mail from genuinely important messages. “



Aprendizagem Bayesiana
Introdução

João Gama
jgama@liaad.up.pt
http://www.liaad.up.pt/~jgama

Sumário

- O teorema de Bayes
  - Motivação
  - O Bayes-óptimo
  - O erro de Bayes
- Algoritmos de Classificação derivados do teorema de Bayes
  - Naive- Bayes
  - Tree aumengted naive-Bayes
  - K-dependence Bayesian Classifiers
- Desenvolvimentos

O teorema de Bayes - Introdução

- Considere um problema de diagnóstico médico:
  - Duas alternativas (*exclusivas*)
    - O doente tem um determinado tipo de cancro
    - O doente **não** tem um determinado tipo de cancro
- É sabido que a probabilidade de observar uma pessoa com este tipo de cancro é 0.008.
- Existe um teste de laboratório que dá apenas um indicação imperfeita sobre a presença (ausência) do cancro.
  - O teste foi negativo em 97% de casos em que o doente não tinha cancro.
  - O teste foi positivo em 98% de casos em que o doente tinha cancro.

P(sim) = 0.008	P(não) = 0.992
P(+ | sim) = 0.98 P(- | sim) = 0.02	P(+ | não) = 0.03 P(- | não) = 0.97

- Para um novo doente o teste é positivo. Qual deverá ser o diagnóstico?
  - P(sim | +)
  - P(não | +)

O teorema de Bayes

- O teorema de Bayes responde a esta questão:

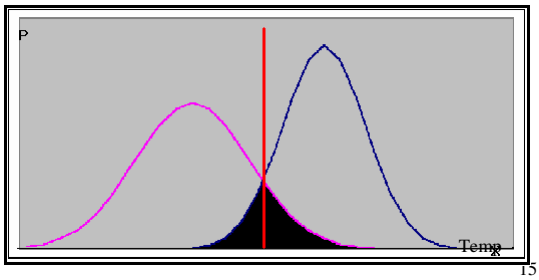
$$p(Decisão_i \mid x) = \frac{p(x \mid Decisão_i)\, p(Decisão_i)}{p(x)}$$

  – A regra de Bayes mostra como alterar as probabilidades *a priori* tendo em conta novas evidências de forma a obter probabilidades *a posteriori*.

- Sendo conhecidas as probabilidades *a priori* e as probabilidades condicionais, a regra de decisão é:

$$\text{argmax } p(\text{Decisão}_i|x) = \text{argmax } [p(\text{Decisão}_i) * p(x|\text{Decisão}_i)]$$

P(sim | +) α P(sim)*P(+ | sim)	0.008* 0.98 = 0.0078
P(não | +) α P(não)*P(+ | não)	0.992 * 0.03 = 0.0298



O erro de Bayes

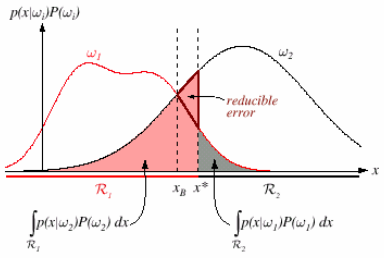


FIGURE 2.17. Components of the probability of error for equal priors and (nonoptimal) decision point x^*. The pink area corresponds to the probability of errors for deciding ω_1 when the state of nature is in fact ω_2; the gray area represents the converse, as given in Eq. 70. If the decision boundary is instead at the point of equal posterior probabilities, x_B, then this reducible error is eliminated and the total shaded area is the minimum possible; this is the Bayes decision and gives the Bayes error rate. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

O teorema de Bayes

- A aplicação do teorema de Bayes como classificador requer:
  - Conhecer as probabilidades *a priori* p(decisão$_i$)
  - As probabilidades condicionais p(x|decisão$_i$)
- Este classificador é óptimo no sentido em que, em média, nenhum outro classificador pode obter melhores resultados usando a mesma informação.
- O erro deste classificador estabelece um mínimo teórico à capacidade de generalização de qualquer classificador: o *erro do Bayes óptimo*.
  - È proporcional à área da superfície a negro.
  - Possibilidade de gerar conjuntos de dados onde é conhecido o erro mínimo.
- Na pratica estas probabilidades são desconhecidas.
  - Estimativas fiáveis destas probabilidades a partir de um conjunto de exemplos, requer um numero infinito de exemplos.
    - $O(k^p)$ sendo p o nr.de variáveis e k o nr. de valores das variáveis.

O teorema de Bayes

- Como ultrapassar o problema?
  - Assumindo simplificações no calculo de p(x|decisão).
- Dependente das assumpções, são obtidos diferentes classificadores:
  - Assumindo que os atributos são independentes dada a decisão.
    - Naive Bayes
  - Assumindo que p(x|decisão) segue uma determinada função densidade de probabilidade.
    - Funções Discriminantes.

O naive Bayes

- Assumindo que o valor dos atributos são condicionalmente independentes dada a classe: $P(\vec{x} \mid C_i) = \prod P(x_j \mid C_i)$
  - Aplicando o teorema de Bayes:
    $$P(C_i \mid \vec{x}) = \frac{P(C_i)}{P(\vec{x})} \prod P(x_j \mid C_i)$$
  - O termo P(x) pode ser ignorado já que não depende da classe.
    $$P(C_i \mid \vec{x}) \propto P(C_i) \prod P(x_j \mid C_i)$$
  - Para cada classe é calculado um valor proporcional a $P(C_i|x)$.

O naive Bayes

- Suponha um problema com $\boldsymbol{p}$ variáveis.
  - Cada variável pode assumir k valores.
  - A estimativa da probabilidade conjunta das p variáveis requer estimar k^p probabilidades.
  - Assumindo que as variáveis são condicionalmente independentes dada a classe, requer estimar kp probabilidades.
- O modelo do naive Bayes pode ser expresso de forma aditiva.
  - Aplicando logaritmos $P(C_i \mid \vec{x}) \propto \log(P(C_i)) + \sum_j \log(P(x_j \mid C_i))$
    - Salienta a contribuição de cada uma das variáveis para a tomada de decisão
  - Considerando apenas duas classes

$$\log \frac{p(c1 \mid x)}{p(c2 \mid x)} = \log \frac{p(c1)}{p(c2)} + \sum \log \frac{p(xj \mid c1)}{p(xj \mid c2)}$$

Exemplo

Tempo	Temperatu.	Humidade	vento	Joga
Sol	85	85	Não	Não
Sol	80	90	Sim	Não
Nublado	83	86	Não	Sim
Chuva	70	96	Não	Sim
Chuva	68	80	Não	Sim
Chuva	65	70	Sim	Não
Nublado	64	65	Sim	Sim
Sol	72	95	Não	Não
Sol	69	70	Não	Sim
Chuva	75	80	Não	Sim
Sol	75	70	Sim	Sim
Nublado	72	90	Sim	Sim
Nublado	81	75	Não	Sim
Chuva	71	91	Sim	Não

- Dado um exemplo para classificar:

Tempo	Temperatu.	Humidade	vento	Joga
Sol	66	90	Sim	?

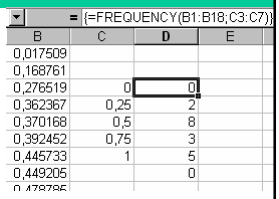
  - P(joga = sim | x) = p(joga = sim) * p(tempo=sol|joga=sim) * p(temperatura = 66|joga=sim)* p(humidade = 90| joga=sim) * p(vento=sim | joga=sim)
  - P(joga = nao | x) = p(joga = nao) * p(tempo=sol|joga=nao) * p(temperatura = 66|joga=nao)* p(humidade = 90| joga=nao) * p(vento=sim | joga=nao)
- Como calcular, a partir do conjunto de treino, estas probabilidades ?

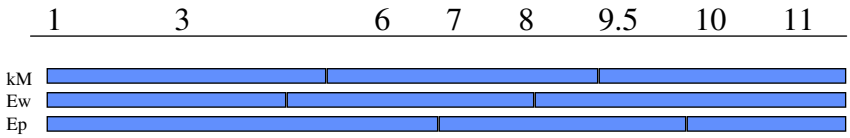
Naive Bayes - Implementação

- Todas as probabilidades condicionais são estimadas a partir do conjunto de treino.
  - Para estimar $P(C_i)$ é necessário contar o número de exemplos para cada classe.
  - Para estimar $P(x_j|C_i)$ é necessário distinguir duas situações:
    - O domínio do atributo é um conjunto finito (nominal).
      - Contar o número de exemplos em que são observados simultaneamente o valor x_j e a class C_i.
    - O domínio do atributo é continuo (toma valores num subconjunto dos números reais). Duas alternativas:
      - É assumido uma distribuição para os valores do atributo.
        - » Usualmente a distribuição normal.
      - O atributo é discretizado e tratado como um atributo nominal.

Atributos contínuos - discretização

- Discretização
  - Quantos intervalos?
    - Nr. de intervalos = min(10, nr.de valores diferentes) (Domingos & Pazzani)
  - Qual a amplitude de cada intervalo?
    - Intervalos com a mesma amplitude
    - Intervalos com a mesma frequência de valores observados
    - K-means
      - K intervalos que minimizam a soma das distancias ao centro de gravidade de cada intervalo
    - Método de Fisher

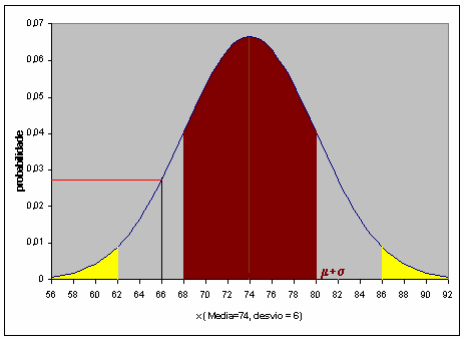




Atributos contínuos – distribuição normal

- A função densidade de probabilidade normal:
  - Requer conhecimento da média e do desvio padrão da variável aleatória.
  - A curva é simétrica em torno da média e amplitude dada pelo desvio padrão.
  - Para uma variável aleatória **x** de μ=74 e σ=6 a probabilidade de observar x=66 é 0.0273

$$f(x) = \frac{1}{\sqrt{2\Pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



Naive Bayes - Exemplo

Tempo	Temperatu.	Humidade	vento	Joga
Sol	85	85	Não	Não
Sol	80	90	Sim	Não
Nublado	83	86	Não	Sim
Chuva	70	96	Não	Sim
Chuva	68	80	Não	Sim
Chuva	65	70	Sim	Não
Nublado	64	65	Sim	Sim
Sol	72	95	Não	Não
Sol	69	70	Não	Sim
Chuva	75	80	Não	Sim
Sol	75	70	Sim	Sim
Nublado	72	90	Sim	Sim
Nublado	81	75	Não	Sim
Chuva	71	91	Sim	Não

Naive Bayes - Exemplo

- Tabelas de distribuição
  - Nr. de Exemplos: 14
  - Nr. Exemplos por classe:
    - Sim: 9
    - Não: 5

Tempo			Temperatura			Humidade			Vento		
	Sim	**Não**		**Sim**	**Não**		**Sim**	**Não**		**Sim**	**Não**
Sol	2	3	**Média**	73	74.6	**Média**	79.1	86.2	**Falso**	6	2
Nublado	4	0	**Desv.**	6.2	7.9	**Desv.**	10.2	9.7	**Verda.**	3	3
Chuva	3	2									

Naive Bayes - Exemplo

- Dado um exemplo para classificar:

Tempo	Temperatu.	Humidade	vento	Joga
Sol	66	90	Sim	?

  - P(joga = sim | x) = p(joga = sim) * p(tempo=sol|joga=sim) * p(temperatura = 66|joga=sim)*p(humidade = 90| joga=sim) *p(vento=sim | joga=sim)
  - P(joga = nao | x) = p(joga = nao) * p(tempo=sol|joga=nao) * p(temperatura = 66|joga=nao)*p(humidade = 90| joga=nao) *p(vento=sim | joga=nao)

Analise

- Em domínios onde todos os atributos são nominais
  - O número de possíveis estados do naive Bayes é finito.
    - d ^ (#classes * (#atributos * #valores+1))
      - d é o nr. de diferente números representáveis na máquina
        - » Numa maquina de 16 bits, d é 2^16=65536
  - Um naive Bayes é equivalente a uma *máquina linear* (combinação linear dos atributos).
    - Definindo um atributo booleano para cada atributo = valor
      - O novo atributo indica a presença (ou não) do valor do atributo num determinado exemplo.

$$P(C_i \mid \vec{x}) \propto \log(P(C_i)) + \sum_{j,k} \log(P(x_j = v_{j,k} \mid C_i)) * b_{j,k}$$

Naive Bayes

- O naive Bayes sumariza a variabilidade de um conjunto de dados em tabelas de probabilidades condicionais.
- A dimensão do modelo é independente do numero de exemplos
  - Estável em relação a perturbações do conjunto de treino
- Em problemas práticos tem boa performance mesmo em situações onde há claras dependências entre atributos.
  - Em problemas de classificação (funções de custo 0-1) o que é importante é a ordenação de $p(Cl_i|x)$.
- É robusto ao ruído e atributos irrelevantes.
- Todas as quantidades requeridas para construir o classificador podem ser calculadas numa única passagem pelo conjunto de treino.
  - Algoritmo On-line, Incremental

O Discriminante Linear

- A regra de Bayes atribui a um exemplo a classe mais provável: $\text{argmax}_i P(Cl_i)*P(x|Cl_i)$
- Assumindo que:
  - Para cada classe os exemplos são independentes.
  - O vector dos atributos segue uma distribuição multivariada *normal*
    - Os vectores das médias são diferentes para cada classe.
    - Mas têm a mesma matriz de co-variância.
- A função de densidade de probabilidade de uma distribuição multivariada normal de parâmetros μ (vector média) e $\sum$ (matriz de co-variâncias) é:

$$N(\mu,\sum) = \frac{1}{\sqrt{2\pi|\sum|}}\exp(-\frac{1}{2}(x-\mu)^T \sum^{-1}(x-\mu)$$

O Discriminante Linear

- Supondo que:
  - A probabilidade "a priori" da classe C_i é $P(C_i)$
  - A função de densidade de probabilidade relativa á classe C_i é f_i
- $P(C_i \mid x) = P(C_i) * f_i(x)$
- O logaritmo desta probabilidade é:

$$\log(P(Cl_i)) - \frac{1}{2}(x-\mu_i)^T \Sigma^{-1}(x-\mu_i)$$

$$\log(P(Cl_i)) - \frac{1}{2}x^T \Sigma^{-1} x + \frac{1}{2}x^T \Sigma^{-1}\mu_i + \frac{1}{2}\mu^T{}_i \Sigma^{-1} x - \frac{1}{2}\mu'{}_i \Sigma^{-1}\mu_i$$

$$\log(P(Cl_i)) - \frac{1}{2}\mu^T{}_i \Sigma^{-1}\mu_i + x^T \Sigma^{-1}\mu_i$$

  - Tendo em conta a simetria de Σ e que o termo $x^T\Sigma^{-1}x$ é independente da classe.

O Discriminante Linear

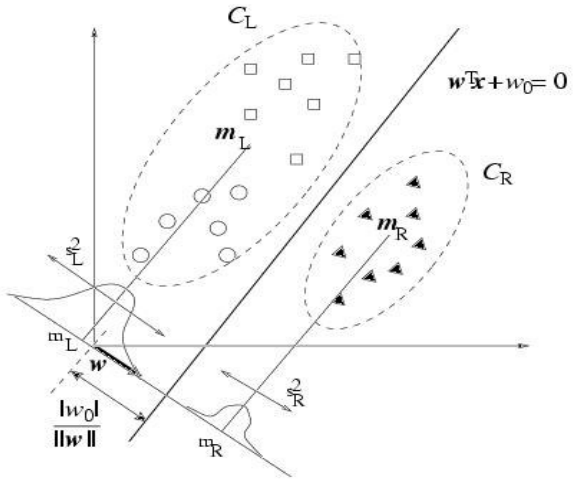
- Para cada classe, um discriminante é um hiper-plano.
  - O hiperplano é uma combinação linear dos atributos:
    - $Hi = w_0+w_1x_1+...+w_nx_n$
      - $H_i = \alpha_i+x^T\beta_i$
      - $\beta_i = \Sigma^{-1}\mu_i$ e $\alpha_i = \log(p(C_i)) - 1/2\mu_i^T \Sigma^{-1}\mu_i$
- Como classificar um exemplo de teste x ?
  - Calculo das probabilidades "a posteriori" $P(C_i|x)$:

$$P(Cl_i \mid x) = \exp[\log(P(Cl_i)) - \frac{1}{2}\mu_i^T \Sigma^{-1}\mu_i + x^T \Sigma^{-1}\mu_i]$$

  - O exemplo é classificado na classe que maximiza $P(C_i|x)$

Discriminante Linear

- O caso de duas classes:



O discriminante logistico

- Maximiza uma função de verosimilhança:

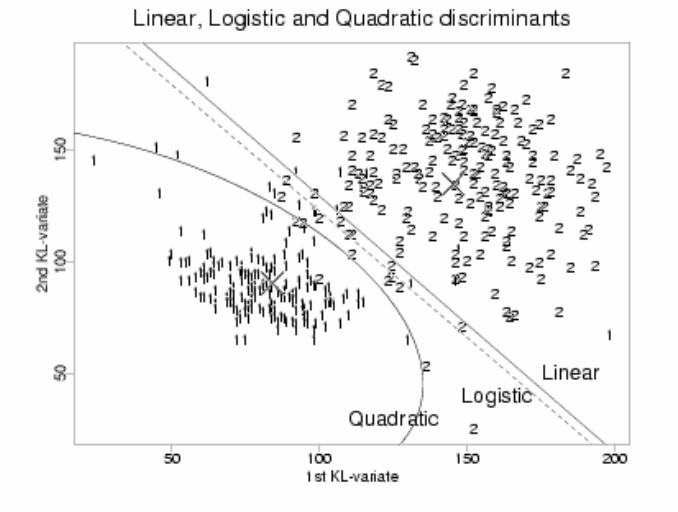
$$L(\beta_1,...,\beta_{n-1}) = \prod_1 P(C_1 \mid x)....\prod_n P(C_n \mid x)$$

- Modela o quociente dos logaritmos das funções densidade de probabilidade:
  - Dados um exemplo x e $n-1$ vectores de coeficientes β, a probabilidade de x pertencer a uma classe i é:

$$P(C_i \mid x) = \frac{\exp(\beta_i x)}{\sum_{j=1...n} \exp(\beta_j \mid x)}$$

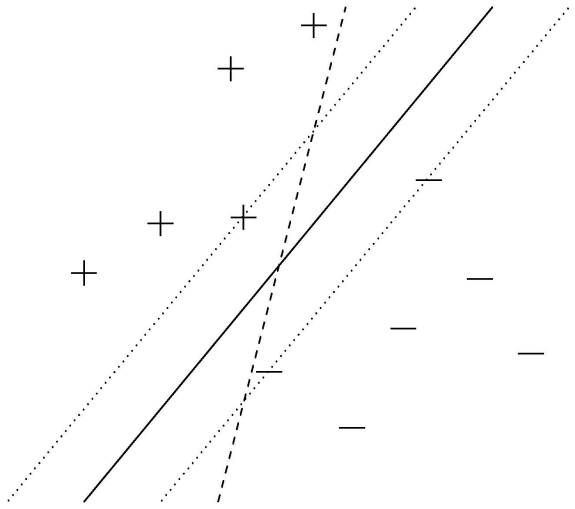
$$P(C_n \mid x) = \frac{1}{\sum_{j=1...n} \exp(\beta_j \mid x)}$$

Superfícies de decisão



J. Gama 35

Support Vector Machines



Support Vector Machines

- Many different hyperplanes can separate positive and negative examples
- Choose hyperplane with maximum margin
- **Margin**: Min. distance between plane and example
- Bound on VC dimension decreases with margin
- **Support vectors:** Examples that determine the plane
- $E[error_{\mathcal{D}}(h)] \leq \frac{E[\#support\ vectors]}{\#training\ vectors - 1}$
- Noisy data: use slack variables
- Avoids overfitting even in very high-dimensional spaces (e.g., text)
- Non-linear: augment data with derived features

Bibliografia adicional

- Richard Duda, Peter Hart; (1973) "Pattern Classification and Scene Analysis", J.Wiley & Sons
- Tom Mitchell, Machine Learning (cap. 6) McGraw-Hill, 1997
- J.Pearl, Probabilistc Reasoning in Intelligent Systems: Networks of plausible Inference, Morgan Kaufman, 1988
- Pedro Domingos, M.Pazzani (1997) "On the optimality of the Simple Bayes Classifier under zero-one loss", Machine Learning, 29

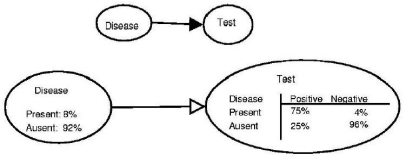
- KDDCup 1998 – Boosting naive Bayes – winner
- Coil 1999 – Simple naive Bayes – winner
- The most used classifier in Text Mining

Going behind the independence assumption

(In)dependence

- A patient takes a lab test and the result comes back positive. The test returns:
  - a correct positive result in only 75% of the cases in which the disease is actually present, and
  - a correct negative result in only 96% of the cases in which the disease is not present.
  - Furthermore, 8% of the entire population have this cancer.
- How to represent that information?

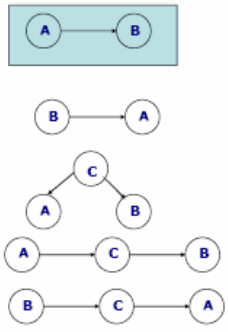
(In)dependence



- It is useful to represent this information in a graph.
  - The graphical information is qualitative
    - The nodes represent variables.
  - Arcs specify the (in)dependence between variables.
    - Direct arcs represent influence between variables.
    - The direction of the arc tell us that the value of the variable disease influences the value of the variable test.

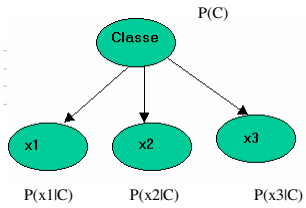
The semantics of Arrows

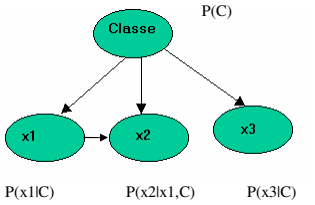
- Direction of Arrow indicates Influence not causality.
  - The ground truth might be different!



Improving naïve Bayes

- Estimar a probabilidade conjunta de um conjunto de variáveis
  - Modelo qualitativo
    - Grafo (DAG) das dependências causais das variáveis
  - Modelo quantitativo
    - Tabelas de distribuição
- Naive Bayes:
  - $P(C|x_1,x_2,x_3)=P(C).P(x_1|C).P(x_2|C).P(x_3|C)$
- Redes Bayesians
  - $P(C|x_1,x_2,x_3)= P(C).P(x_1|C).P(x2|x1,C).P(x_3|C)$





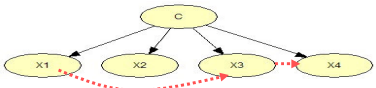
P(C)	
C=0	C=1

P(X | C)		
C \ X	0	1
0		
1		

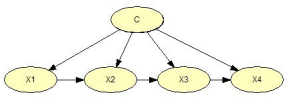
P(X1 | X2, C)				
	X1=0		X1=1	
C	X2=0	X2=1	X2=0	X2=1
0				
1				

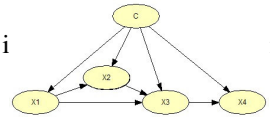
Improving naïve Bayes

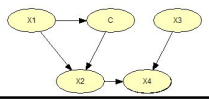
- We can improve the performance of Naïve Bayes:
  - by relaxing the independence assumption



- One natural extension: Bayesian Network Classifiers (BNCs)
- Restricted approach: network structures based on the NB structure: the class node is parent of all the attributes
- Ur class node i node

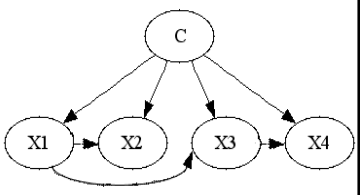






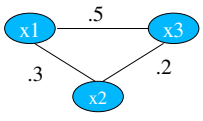
Tree aumented naive Bayes (TAN)

- Apresentado por Friedman e Goldszmidt[97]
  - possibilita representar dependências entre pares de atributos.
- O TAN é definida pelas seguintes condições:
  - cada atributo depende condicionalmente da classe (tal como ocorre no naive Bayes);
  - Cada atributo pode depender condicionalmente de um outro atributo.



Tree augmented naive Bayes

	x_1	x_2	x_3
x1	-	.3	.5
x2	-	-	.2
x3	-	-	-



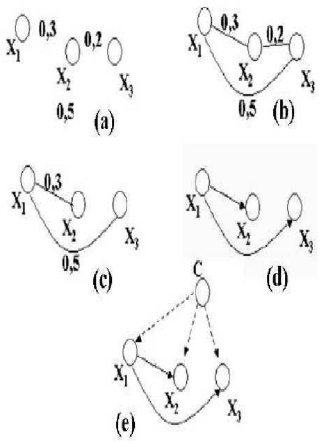
1-Compute the Mutual Information between all pairs of variables given the Class:

$$I(X,Y \mid C) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{r=1}^{w} p(x_i, y_j, c_r) \log \frac{p(x_i, y_j \mid c_r)}{p(x_i \mid c_r) p(y_j \mid c_r)}$$

2-Construct a spanning tree maximizing MI.

3-All the variables depends on the class.

Tree augmented naive Bayes



- Construct a complete undirected graph between all variables (except the class)
- Assign to each edge the corresponding $I(X_i,X_j|C)$
- Construct a spanning tree maximizing MI.
  - The minimal set of edges that connect all vertices.
- Direct the tree, starting from the edge with highest I(Xi,Xj|C)
- All the variables depends on the class.

Bases of Information Theory

- Uncertainty of a random Variable (Entropy):
  - $H(X) = -\Sigma P(x) \log_2(P(x))$
    - The expected number of bits needed to store the values of X
    - High values of H(.) means high randomness (less predictable)
- Uncertainty about X after knowing Y (Conditional Entropy):
  - $H(X|Y) = -\Sigma_y P(y) \Sigma_x P(x|y) \log_2 P(x|Y) = H(X,Y)-H(X)$
    - The entropy of X given the values of Y
    - Low values of H(X|Y):
      - the more we can predict the value of X knowing the value of Y
    - $H(X \mid Y) = 0$ if and only if the value of X is completely determined by the value of Y

Bases of Information Theory

- Mutual Information: I(X,Y)
  - Reduction in the uncertainty of X when Y is known
  - I(X,Y) = H(X|Y) - H(X)
  - $I(X,Y) = \Sigma_i \Sigma_j P(x_i,y_j) \log_2 (P(x_i,y_j)/P(x_i)p(x_j))$
- Measures the degree of dependence between X and Y
  - I(X,Y)=0 means that X and Y are independent
  - I(X,Y) increases with the increase of the degree of dependence between X and Y

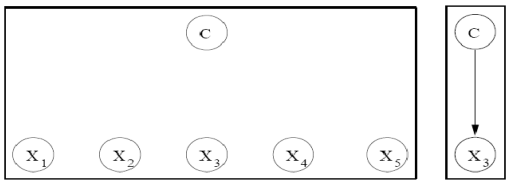
k- Dependence Byesian Classifiers

- k-DBCs represent in one single class a full spectrum of allowable dependences
  - Sahami, M, 1996
  - NB at the most restrictive end
  - the full BN at the most general extreme
- A k-DBC is a BN which:
  - contains the structure of NB
  - allows each attribute X_i to have a maximum of k attribute nodes as parents

k- Dependence Byesian Classifiers

- Start:
  - Fix **k**: the maximum number of allowable dependences.
- Begin
  - Compute $I(X_i,C)$ and $I(X_i,X_j|C)$ for all pairs of variables
  - Iterate
    - Choose the variable X_{max} not yet in the model that maximizes $I(X_i|C)$
    - Choose the **k** parents of X_{max}:
      - those with greater $I(X_j,X_{max}|C)$

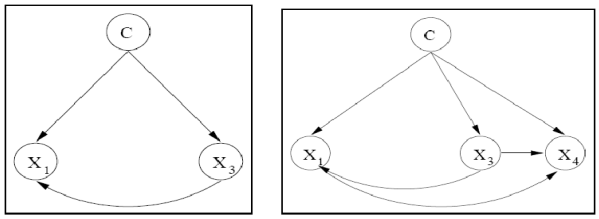
k- Dependence Byesian Classifiers



kDB $k = 2$.

$$I(X_3,C) > I(X_1,C) > I(X_4,C) > I(X_5,C) > I(X_2,C)$$
$$I(X_3,X_4|C) > I(X_2,X_5|C) > I(X_1,X_3|C) > I(X_1,X_2|C) > I(X_2,X_4|C) >$$
$$I(X_2,X_3|C) > I(X_1,X_4|C) > I(X_4,X_5|C) > I(X_1,X_5|C) > I(X_3,X_5|C)$$

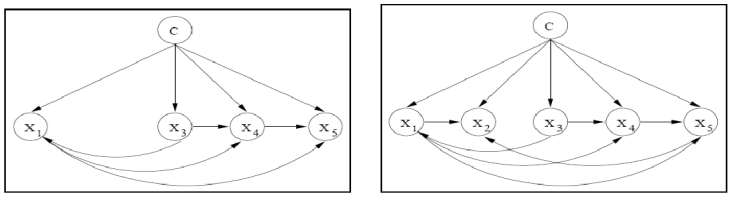
k- Dependence Byesian Classifiers



kDB $k = 2$.

$$I(X_3, C) > I(X_1, C) > I(X_4, C) > I(X_5, C) > I(X_2, C)$$
$$I(X_3, X_4|C) > I(X_2, X_5|C) > I(X_1, X_3|C) > I(X_1, X_2|C) > I(X_2, X_4|C) >$$
$$I(X_2, X_3|C) > I(X_1, X_4|C) > I(X_4, X_5|C) > I(X_1, X_5|C) > I(X_3, X_5|C)$$
$$p(c|x_1, x_2, x_3, x_4, x_5)$$

k- Dependence Byesian Classifiers



kDB $k = 2$.

$$I(X_3, C) > I(X_1, C) > I(X_4, C) > I(X_5, C) > I(X_2, C)$$
$$I(X_3, X_4|C) > I(X_2, X_5|C) > I(X_1, X_3|C) > I(X_1, X_2|C) > I(X_2, X_4|C) >$$
$$I(X_2, X_3|C) > I(X_1, X_4|C) > I(X_4, X_5|C) > I(X_1, X_5|C) > I(X_3, X_5|C)$$
$$p(c|x_1, x_2, x_3, x_4, x_5) \propto$$
$$p(c)p(x_1|x_3, c)p(x_2|x_1, x_5, c)p(x_3|c)p(x_4|x_1, x_3, c)p(x_5|x_1, x_4, c)$$

k- Dependence Byesian Classifiers

- General Model:
  - –All the attributes depends on the class
  - –Any attribute depends on *k* other attributes, at most.
- Restricted Bayesian Networks
- Decision Models of Increase Complexity
- k-Dependency Bayesian Networks:
  - a common framework for classifiers with increase (smooth) complexity

Software Available

- Weka – Bayesian Classifiers
- Elvira – Spanish Consortium
- Open source package for the technical computing language R, developed by Aalborg University (http://www.math.auc.dk/novo/deal)
- Kevin Murphy's MATLAB toolbox –
  - supports dynamic BNs, decision networks, many exact and approximate inference algorithms, parameter estimation, and structure learning (http://www.ai.mit.edu/~murphyk/Software/BNT/bnt.html)
- Free Windows software for creation, assessment and evaluation of belief networks (http://www.research.microsoft.com/dtas/msbn/default.htm)
- http://www.snn.ru.nl/nijmegen

Bibliografia

- Tom Mitchell, *Machine Learning*, (chapter 6), McGraw-Hill,1997
- R. Duda, P. Hart, D. Stork; *Pattern Classification*, J. Willey & Sons, 2000
- J.Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988
- P. Domingos, M.Pazzani; *On the Optimality of the Simple Bayes Classifier under zero-one loss*, Machine Learning, 29
- R. Neapolitan, *Learning Bayesian Networks*, Prentice Hall, 2004

Árvores de Decisão

João Gama

Jgama@liacc.up.pt

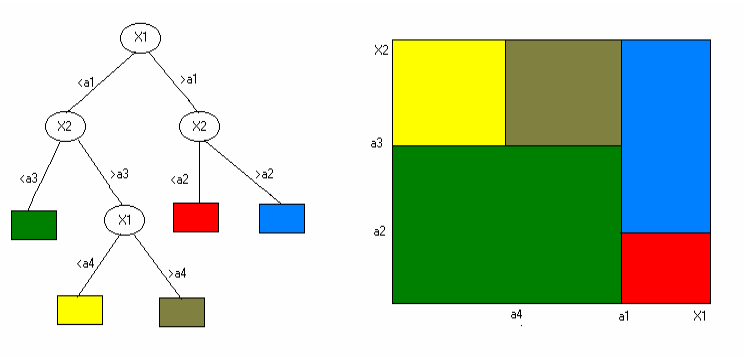
Sumario

- Árvores de decisão
  - Motivação
  - Construção de uma árvore de decisão
    - Critérios para seleccionar atributos
      - Entropia
  - Podar a árvore
    - Estimativas de erro
  - Extensões
    - Árvores multivariadas

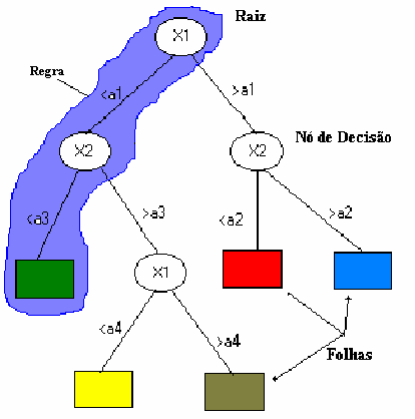
Árvores de Decisão

- Uma árvore de decisão utiliza uma estratégia de *dividir-para-conquistar*:
  - Um problema complexo é decomposto em sub-problemas mais simples.
  - Recursivamente a mesma estratégia é aplicada a cada sub-problema.
- A capacidade de discriminação de uma árvore vem da:
  - Divisão do espaço definido pelos atributos em sub-espaços.
  - A cada sub-espaço é associada uma classe.
- Crescente interesse
  - CART (Breiman, Friedman, et.al.)
  - C4.5 (Quinlan)
  - S_{plus}, Statistica, SPSS , R

Árvores de decisão – Exemplo da partição do espaço dos atributos



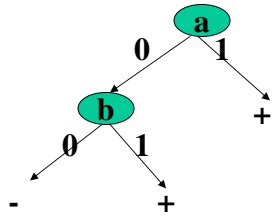
O que é uma Arvore de Decisão?



- Representação por árvores de decisão:
  - Cada nó de decisão contem um teste num atributo.
  - Cada ramo descendente corresponde a um possível valor deste atributo.
  - Cada Folha está associada a uma classe.
  - Cada percurso na arvore (da raiz à folha) corresponde a uma regra de classificação.
- No espaço definido pelos atributos:
  - Cada folha corresponde a uma região
    - Hiper-rectângulo
  - A intersecção dos hiper-rectângulos é vazio
  - A união dos hiper-rectângulos é o espaço completa.

Representação

- Uma árvore de decisão representa a disjunção de conjunções de restrições nos valores dos atributos
  - Cada ramo na árvore é uma conjunção de condições
  - O conjunto de ramos na árvore são disjuntos
    - DNF (disjuntive normal form)
  - Qualquer função lógica pode ser representada por um árvore de decisão.
    - Exemplo **a** or **b**



Construção de uma árvore de decisão

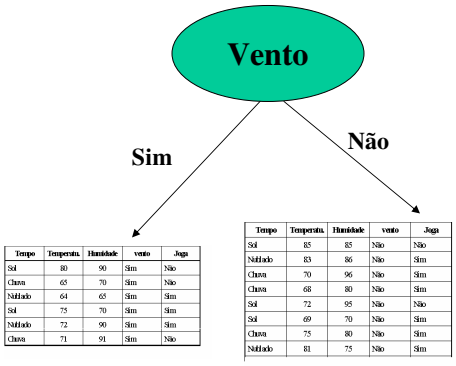
- A ideia *base*:
  1. Escolher um atributo.
  2. Estender a árvore adicionando um ramo para cada valor do atributo.
  3. Passar os exemplos para as folhas (tendo em conta o valor do atributo escolhido)
  4. Para cada folha
     1. Se todos os exemplos são da mesma classe, associar essa classe à folha
     2. Senão repetir os passos 1 a 4

Exemplos:

O conjunto de dados original:

Tempo	Temperatu.	Humidade	vento	Joga
Sol	85	85	Não	Não
Sol	80	90	Sim	Não
Nublado	83	86	Não	Sim
Chuva	70	96	Não	Sim
Chuva	68	80	Não	Sim
Chuva	65	70	Sim	Não
Nublado	64	65	Sim	Sim
Sol	72	95	Não	Não
Sol	69	70	Não	Sim
Chuva	75	80	Não	Sim
Sol	75	70	Sim	Sim
Nublado	72	90	Sim	Sim
Nublado	81	75	Não	Sim
Chuva	71	91	Sim	Não

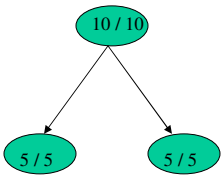
Selecciona um atributo:

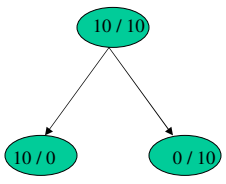


Qual o 'melhor' atributo?

Critérios para Escolha do Atributo

- Como medir a *habilidade* de um dado atributo discriminar as classes?
- Existem muitas medidas.
  Todas concordam em dois pontos:
  - Uma divisão que mantêm as proporções de classes em todas as partições é inútil.
  - Uma divisão onde em cada partição todos os exemplos são da mesma classe tem utilidade máxima.





Caracterização das medidas de partição

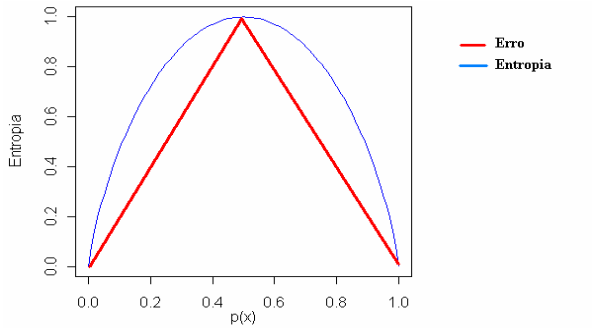
- Medida da diferença dada por uma função baseada nas proporções das classes entre o nó corrente e os nós descendentes.
  - Valoriza a pureza das partições.
  - Gini, entropia
- Medida da diferença dada por uma função baseada nas proporções das classes entre os nós descendentes.
  - Valoriza a disparidade entre as partições.
  - Lopez de Mantaras
- Medida de independência
  - Medida do grau de associação entre os atributos e a classe.

Entropia

- Entropia é uma medida da aleatoridade de uma variável.
- A entropia de uma variável nominal *X* que pode tomar *i* valores:

$$entropia\ (X) = -\sum_i p_i * \log_2 p_i$$

  - A entropia tem máximo ($\log_2$ i) se $p_i = p_j$ para qualquer i <> j
  - A entropia(x) = 0 se existe um *i* tal que $p_i = 1$
    - É assumido que $0 * \log_2 0 = 0$



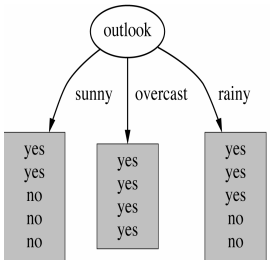
Ganho de Informação

- No contexto das árvores de decisão a entropia é usada para estimar a aleatoridade da variável a prever: classe.
- Dado um conjunto de exemplos, que atributo escolher para teste?
  - Os valores de um atributo definem partições do conjunto de exemplos.
  - O ganho de informação mede a redução da entropia causada pela partição dos exemplos de acordo com os valores do atributo.

$$ganho\ (Exs, Atri) = entropia\ (Exs) - \sum \frac{\#\ Exs_v}{\#\ Exs}\ entropia\ (Exs_v)$$

- A construção de uma árvore de decisão é guiada pelo objectivo de diminuir a entropia ou seja a aleatoridade -dificuldade de previsão- da variável objectivo.

Calculo do Ganho de Informação de um atributo nominal



	Sol	Nublado	Chuva
Sim	2	4	3
Não	3	0	2

- Informação da Classe:
  - p(sim) = 9/14
  - p(não) = 5/14
  - Info(joga) =
    - = - 9/14 $\log_2$ 9/14 – 5/14 $\log_2$ 5/14 = 0.940 bits
- Informação nas partições:
  - p(sim|tempo=sol) = 2/5
  - p(não|tempo=sol) = 3/5
  - Info(jogaltempo=sol)
    - = -2/5$\log_2$2/5 –3/5$\log_2$3/5 = 0.971 bits
  - Info(jogaltempo=nublado) = 0.0 bits
  - Info(jogaltempo=chuva) = 0.971 bits
  - **Info(tempo)**
    - **= 5/14*0.971+4/14*0 + 5/14*0.971 = 0.693 bits**
- Ganho de Informação obtida neste atributo:
  - Ganho(tempo) = 0.940 – 0.693 = 0.247 bits

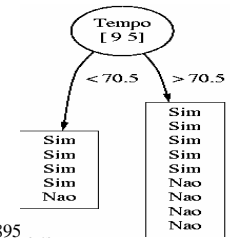
Calculo do Ganho para Atributos numéricos

- Um teste num atributo numérico produz uma partição binária do conjunto de exemplos:
  - Exemplos onde valor_do_atributo < ponto_referência
  - Exemplos onde valor_do_atributo >= ponto_referência
- Escolha do ponto de referência:
  - Ordenar os exemplos por ordem crescente dos valores do atributo numérico.
  - Qualquer ponto intermédio entre dois valores diferentes e consecutivos dos valores observados no conjunto de treino pode ser utilizado como possível ponto de referência.
    - É usual considerar o valor médio entre dois valores diferentes e consecutivos.
  - Fayyard e Irani (1993) mostram que de todos os possíveis pontos de referência aqueles que maximizam o ganho de informação separam dois exemplos de classes diferentes.

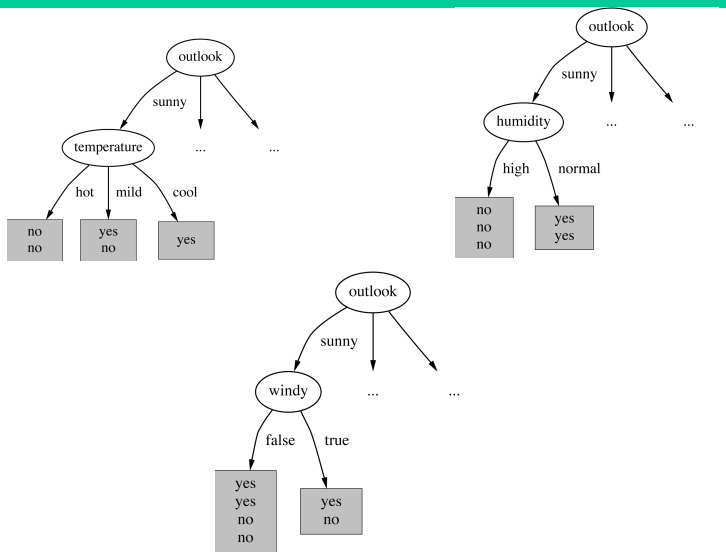
Calculo do Ganho para Atributos numéricos

Temperatu.	Joga
64	Sim
65	Não
68	Sim
69	Sim
70	Sim
71	Não
72	Não
72	Sim
75	Sim
75	Sim
80	Não
81	Sim
83	Sim
85	Não

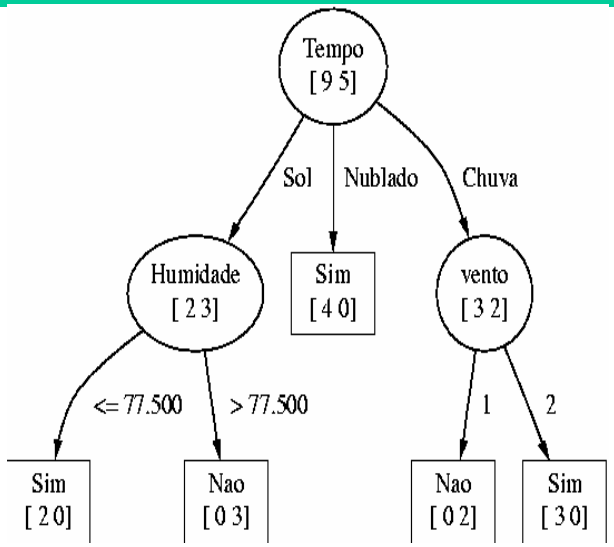
- Considere o ponto de referência temperatura = 70.5
- Um teste usando este ponto de referência divide os exemplos em duas partições:
  - Exemplos onde temperatura < 70.5
  - Exemplos onde temperatura > 70.5
- Como medir o ganho de informação desta partição?
- Informação nas partições
  - p(sim | temperatura<70.5)=4/5
  - p(não | temperatura<70.5)=1/5
  - p(sim | temperatura>70.5)=5/9
  - p(não | temperatura>70.5)=4/9
  - Info(joga | temperatur<70.5) =
    - $-4/5\log_2 4/5 - 1/5 \log_2 1/5 = 0.721$ bits
  - Info(joga | temperatura >70.5) =
    - $-5/9 \log_2 5/9 - 4/9 \log_2 4/9 = 0.991$ bits
  - Info(temperatura)=5/14*0.721+9/14*0.991 = 0.895
  - Ganho(temperatura) = 0.940 – 0.895 = 0.045 bits



Repetir o processo



Arvore de decisão final



Critérios de paragem

- Quando parar a divisão dos exemplos?
  - Todos os exemplos pertencem á mesma classe.
  - Todos os exemplos têm os mesmos valores dos atributos (mas diferentes classes).
  - ✓ O número de exemplos é inferior a um certo limite.
  - ❖ (?) O mérito de todos os possíveis testes de partição dos exemplos é muito baixo.

Construção de uma Árvore de Decisão

- Input: Um conjunto exemplos
- Output: Uma árvore de decisão
- Função GeraArvore(Exs)
  - Se criterio_paragem(Exs) = TRUE
    - retorna Folha
  - Escolhe o atributo que maximiza o critério_divisão(Exs)
  - Para cada partição *i* dos exemplos baseada no atributo escolhido
    - $Arvore_i = GeraArvore(Exs_i)$
  - Retorna um nó de decisão baseado no atributo escolhido e com descendentes $Arvore_i$.
- Fim

Construção de uma Arvore de Decisão

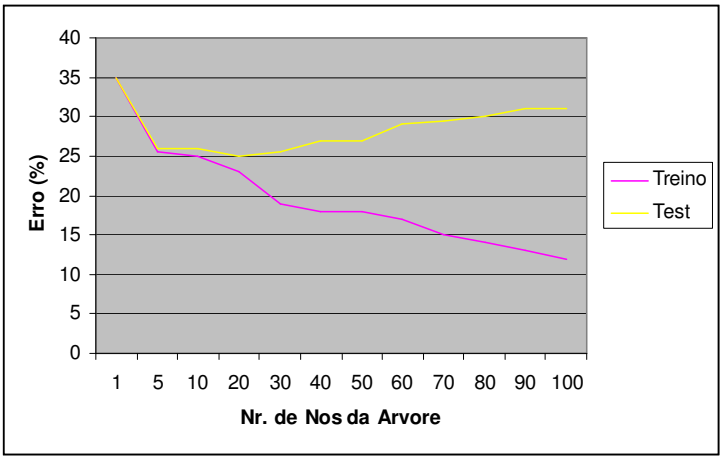
- O problema de construir uma árvore de decisão:
  - Consistente com um conjunto de exemplos
  - Com o menor numero de nós
  - É um problema *NP* completo.
- Dois problemas:
  - Que atributo seleccionar para teste num nó?
  - Quando parar a divisão dos exemplos ?
- Os algoritmos mais divulgados:
  - Utilizam heurísticas que tomam decisões olhando para a frente um passo.
  - Não reconsideram as opções tomadas.

Regularização em Árvores de Decisão

Sobre-ajustamento

- O algoritmo de partição recursiva do conjunto de dados gera estruturas que podem obter um ajuste aos exemplos de treino perfeito.
  - Em domínios sem ruído o nr. de erros no conjunto de treino pode ser 0.
- Em problemas com *ruído* esta capacidade é problemática:
  - A partir de uma certa profundidade as decisões tomadas são baseadas em pequenos conjuntos de exemplos.
  - A capacidade de generalização para exemplos não utilizados no crescimento da arvore diminui.

Variação do erro com o nr. de nós



Sobre-ajustamento ("*overfitting*")

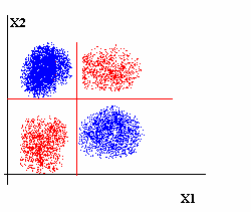
- Definição:
  - Uma arvore de decisão *d* faz sobre-ajustamento aos dados se existir uma arvore *d´* tal que:
    - *d* tem menor erro que d´ no conjunto de treino
    - mas *d´* tem menor erro na população.
- Como pode acontecer:
  - Ruído nos dados
  - Excesso de procura
- O numero de parâmetros de uma árvore de decisão cresce linearmente com o numero de exemplos.
  - Uma árvore de decisão pode obter um ajuste perfeito aos dados de treino.

Sobre-ajustamento

- Occam's razor: preferência pela hipótese mais simples.
  - Existem menos hipóteses simples do que complexas.
  - Se uma hipótese simples explica os dados é pouco provável que seja uma coincidência.
  - Uma hipótese complexa pode explicar os dados apenas por coincidência.
- A avaliação de uma hipótese deve ter em conta o processo de construção da hipótese.

Simplificar a arvore

- Duas possibilidades:
  - Parar o crescimento da arvore mais cedo (pre-pruning).
  - ➢ Crescer uma arvore completa e podar a arvore (pos-pruning).
    - "*Growing and pruning is slower but more reliable*"
      - Quinlan, 1988
  - O problema do "Xor"
    - Requer olhar em frente mais que um nível.



Critérios

- Critérios:
  - ➢ Obter estimativas fiáveis do erro a partir do conjunto de treino.
  - Optimizar o erro num conjunto de validação independente do utilizado para construir a arvore.
  - Minimizar:
    - *erro no treino + dimensão da arvore*
      - *Cost Complexity pruning (Cart)*
    - dimensão da arvore + dimensão dos exemplos mal classificados
      - MDL pruning (Quinlan)

Estimativas de Erro

- O problema fundamental do algoritmo de poda é a estimativa de erro num determinado nó.
  - O erro estimado a partir do conjunto de treino não é um estimador fiável.
- O "*reduced error pruning*"
  - consiste em obter estimativas de erro a partir de um conjunto de validação independente do conjunto de treino.
  - Reduz o volume de informação disponível para crescer a arvore.
- O "*Cost complexity pruning*", Breiman, 1984
  - Podar com base na estimativa do erro e complexidade da arvore.
    - Cart (Breiman et al.)
- ➢ O "*Error based pruning*",
  - Podar com base numa estimativa do erro no conjunto de treino.
    - Assume uma distribuição Binomial para os exemplos de um nó.
      - Usado no C5.0

Valores Desconhecidos

- Pré-Processados
  - Substituir o valor desconhecido pelo valor mais provável
    - Atributos numéricos: média.
    - Atributos nominais: moda.
- Na construção do modelo
  - Assumir que um atributo tem como possível valor o valor desconhecido.
  - Atribuir um peso a cada exemplo.
    - Nos exemplos em que o atributo de teste toma um valor desconhecido, o exemplo é passado para todos os nós descendentes com um peso proporcional á probabilidade de um exemplo seguir o ramo.

Extensões

Algumas Ideias:

- Árvores com funções nas folhas
  - NBTree – usa Naïve Bayes nas Folhas
- Árvores com funções nos nós
  - Multivariate Trees
- Árvores com funções nos nós e nas folhas
  - Functional Trees
- Árvores Incrementais
  - Hoeffding Trees
- Florestas de árvores
- ….

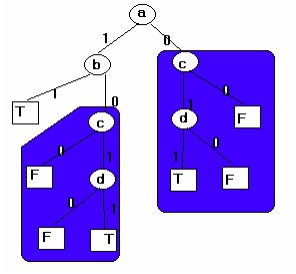
Vantagens das Arvores de decisão

- Método não-paramétrico
  - Não assume nenhuma distribuição particular para os dados.
  - Pode construir modelos para qualquer função desde que o numero de exemplos de treino seja suficiente.
- A estrutura da árvore de decisão é independente da escala das variáveis.
  - Transformações monótonas das variáveis (log x, 2*x, ...) não alteram a estrutura da arvore.
- Elevado grau de interpretabilidade
  - Uma decisão complexa (prever o valor da classe) é decomposto numa sucessão de decisões elementares.
- É eficiente na construção de modelos:
  - Complexidade média O(n log n)
- Robusto á presença de pontos extremos e atributos redundantes ou irrelevantes.
  - Mecanismo de selecção de atributos.

Inconvenientes das Árvores de decisão

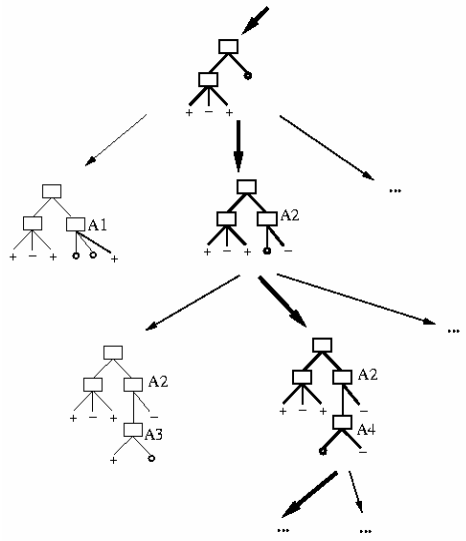
- Instabilidade
  - Pequenas perturbações do conjunto de treino podem provocar grandes alterações no modelo aprendido.
- Presença de valores desconhecidos
- Fragmentação de conceitos
  - Replicação de sub-arvores

$$(a \wedge b) \vee (c \wedge d)$$



O espaço de Hipóteses

- O espaço de hipóteses é completo
  - Qualquer função pode ser representada por uma árvore de decisão.
- Não reconsidera opções tomadas
  - Mínimos locais
- Escolhas com suporte estatístico
  - Robusto ao ruído
- Preferência por árvores mais pequenas



Bibliografia Adicional

- Online:
  - http://www.Recursive-Partitioning.com/
- Tom Mitchell
  - Machine Learning (chap.3)
    - MacGrawHill, 1997
- Quinlan, R.
  - "C4.5 Programs for Machine Learning"
    - Morgan Kaufmann Publishers, 1993
- L.Breiman, J.Friedman, R.Olshen, C.Stone
  - "Classification and Regression Trees"
    - Wadsworth, 1984

Decomposição do Erro

- O erro esperado de um classificador pode ser decomposto em:

$$E = \sum(\sigma_x^2 + bias_x^2 + \text{var}iance_x)$$

  - Ruído no conjunto de dados
    - Erro do Bayes Óptimo
  - Viés (Bias)
    - Mede os erros sistemáticos.
    - Estimativa da capacidade de adaptação da linguagem de representação utilizada pelo algoritmo ao problema.
  - Variância
    - Mede a variabilidade das predições
    - Estimativa da dependência do modelo gerado ao conjunto de treino.

Decomposição para MSE

Erro estimado para diferentes conjuntos de treino.

$$(o-y)^2 = (o-\overline{y}+\overline{y}-y)^2$$
$$= (o-\overline{y})^2 + (\overline{y}-y)^2 + 2(o-\overline{y})(\overline{y}-y)$$
$$E[(o-y)^2] = (o-\overline{y})^2 + E[(\overline{y}-y)^2]$$

$\overline{y}$ – valor médio das previsões

y - previsão

o - valor observado

Estimativa das componentes do Erro

- Dados um conjunto de dados e um algoritmo.
- Dividir o conjunto de dados em:
  - Conjunto de treino e conjunto de Validação
  - Obter *n* conjuntos de treino por amostragem uniforme
  - Gerar *n* modelos utilizando o algoritmo dado.
  - Cada modelo classifica os exemplos do conjunto de validação.
- Para cada exemplo do conjunto de validação estimar:
  - $P(Cl_i|x)$ por votação uniforme
- Usando a definição de Kohavi & Wolpert 96
  - $Variance_x = 1/2(1- Sum_i(P(Cl_i|x)^2))$
  - $Bias_x = 1/2(Sum_i(P(i=y)-P(Cl_i|x))^2)$
    - P(i=y) = 1 se e só se i = y

$$Variancia_x = \frac{1}{2}(1 - \sum_i P(Cl_i \mid x)^2)$$

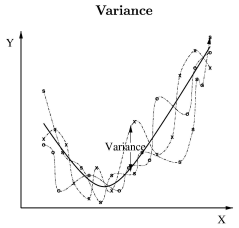
$$Bias_x = \frac{1}{2}(\sum_i (I(i = f(x)) - P(Cl_i \mid x))^2)$$

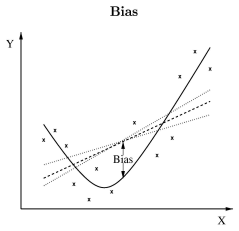
Exemplo

Observado	Cl1	Cl2	Cl3	Cl4	Cl5		P0	P1	Variancia	BIAS	Bias+Variance
0	1	1	0	0	0	2	0,6	0,4	0,24	0,16	0,4
1	0	0	1	1	1	3	0,4	0,6	0,24	0,16	0,4
0	0	0	0	1	1	2	0,6	0,4	0,24	0,16	0,4
1	0	0	0	0	0	0	1	0	0	1	1
1	1	1	1	1	1	5	0	1	0	0	0
									0,144	0,296	0,44
	Erro										
	1	1	0	0	0						
	1	1	0	0	0						
	0	0	0	1	1						
	1	1	1	1	1						
	0	0	0	0	0						
	0,6	0,6	0,2	0,4	0,4						
	erro médio	0,44									

O Compromisso Bias-Variance

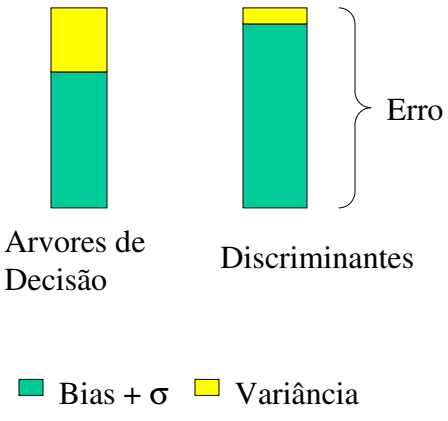
- Aumentando o número de graus de liberdade de um modelo:
  - Diminuição da componente do "Bias"
  - Aumento da variância.
- Minimizar o erro esperado requer um compromisso entre as duas componentes.





Decomposição em "Bias-Variance"

- Funções Discriminantes
  - Variância reduzida
  - *Bias* elevado
- Arvores de decisão
  - Variância elevada
  - *Bias* reduzido



Sumario

- Avaliação de classificadores
  - Como estimar o erro do classificador num conjunto de dados?
  - Qual o melhor algoritmo para um problema?
- Amostragem
  - Validação cruzada
  - Amostragem com reposição
- Teste de Hipóteses
- Decomposição do erro em *bias* e *variance*

Modelos Múltiplos

João Gama
jgama@liacc.up.pt

Modelos Múltiplos

- Diferentes algoritmos de aprendizagem exploram:
  - Diferentes linguagens de representação.
  - Diferentes espaços de procura.
  - Diferentes funções de avaliação de hipóteses.
- Como poderemos explorar estas diferenças ?
  - Será possível obter um conjunto de classificadores cuja performance é melhor que a performance de cada classificador individual ?
- Observação:
  - Não existe um algoritmo que seja o melhor para todos os problemas
    - Resultados experimentais: Projecto Statlog
    - Resultados Teóricos: "No free lunch"

Erro Correlacionado

- Uma condição necessária:
  - *Um conjunto melhora sobre os classificadores individuais se estes discordam entre si.* Hansen & Salamon - 1990
- O *erro correlacionado* é uma métrica da diversidade entre as predições de dois algoritmos.
- *Erro correlacionado*:
  - Probabilidade de dois classificadores cometerem o mesmo erro dado que um deles comete um erro.

$$\phi_{i,j} = p(\hat{f}_i(x) = \hat{f}_j(x) \mid \hat{f}_i(x) \neq f(x) \vee \hat{f}_j(x) \neq f(x))$$

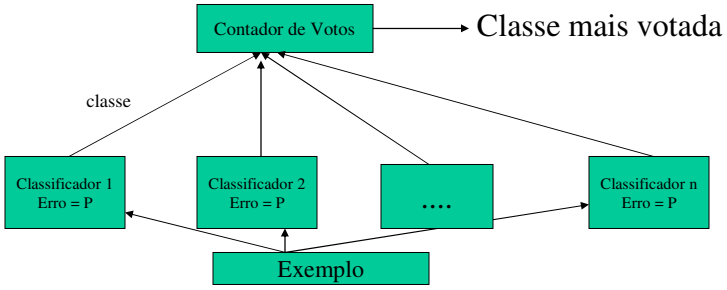
Observado	0	0	0	1	0	1	1	0
Algoritmo A	1	1	1	1	1	1	0	1
Algoritmo B	0	1	1	0	1	1	0	0

$$\phi_{A,B} = 4/7 = 0.57$$

- Será uma condição suficiente?

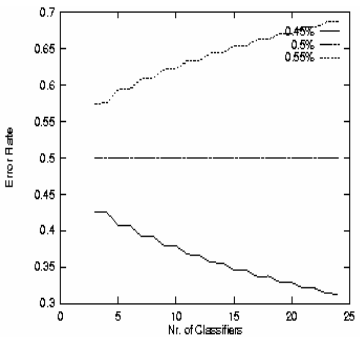
Modelos Múltiplos

- Um estudo em simulação:
  - Considere um problema de duas classes equi-prováveis:
    - $P(Classe_1) = P(Classe_2)$
  - Numero de classificadores:[3..25]
    - Com a mesma probabilidade de cometer erros.
    - $P\_erro(Classificador_i) = \{0.45;0.5;0.55\}$
  - O modelo múltiplo é obtido por agregação dos vários classificadores
    - As predições dos classificadores são agregadas por votação uniforme.



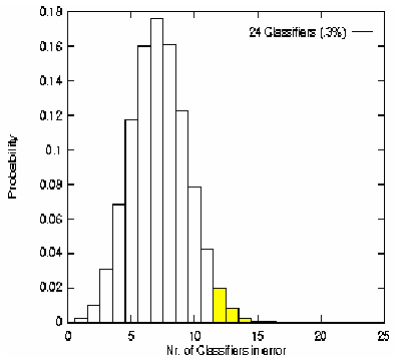
Modelos Múltiplos – Uma Simulação

- Estudo da variação do erro de um conjunto de classificadores variando o numero de classificadores agregados:
  - A probabilidade de erro de cada classificador é:
    - P = 0.5 (escolha aleatória de uma das classes)
      - A probabilidade de erro do conjunto é constante: 0.5
    - P > 0.5
      - A probabilidade de erro do conjunto cresce linearmente com o numero de classificadores
    - P < 0.5
      - A probabilidade de erro do conjunto diminui linearmente com o numero de classificadores
- Uma condição necessária:
  - A taxa de erro de um conjunto de classificadores diminui em relação à taxa de erro dos classificadores individuais se:
    - Cada classificador individual do conjunto tiver uma performance melhor que uma escolha aleatória.



Modelos Múltiplos – Uma Simulação

- Considere um modelo múltiplo obtido agregando por votação uniforme:
  - 23 classificadores.
  - A probabilidade de erro de cada classificador é 30%.
- Dado exemplo a classificar
  - O modelo múltiplo classifica o exemplo incorrectamente se e só se:
    - 12 ou mais classificadores classificam o exemplo incorrectamente.
  - A probabilidade do modelo múltiplo errar é dada pela área sob a curva da distribuição binomial.
    - No caso em estudo a área é de 0.026.
    - Muito menor que o erro de cada classificador.



Condições Necessárias

- "To achieve higher accuracy the models should be diverse and each model must be quite accurate"

  Ali & Pazzani 96

- Condições Necessárias
  - Os classificadores devem ter uma performance melhor que uma escolha aleatória (*random guess*)
  - Os classificadores devem cometer erros não correlacionados.
    - Diferentes tipos de erros.
    - Erros em diferentes regiões do espaço.

Podem os Modelos Múltiplos funcionar na pratica?

- Um algoritmo de aprendizagem efectua uma procura num espaço de hipóteses H.
- A escolha de um único modelo tem vários problemas:
  - Estatísticos
    - O volume de dados é pequeno em relação ao espaço das hipóteses.
    - Decisões sem suporte estatístico.
  - Computacionais
    - Procura heurística.
    - Máximos Locais
  - Representação
    - A função que governa o fenómeno não está em H.
- A utilização de modelos múltiplos pode minimizar qualquer um destes problemas.

Modelos Múltiplos

- Combinação de predições
  - Votação Uniforme
  - Votação Pesada
  - Soma de distribuições
- Gerar Modelos
  - Modelos Homogéneos
    - Bagging
    - Boosting
  - Modelos Heterogéneos
    - Stacking
- Modelos Híbridos
  - *Model Class Selection*

Geração de Modelos

- Métodos para gerar modelos diferentes
  - Geração de classificadores homogéneos
    - *Bagging*, Breiman 92
    - *Boosting*, Schapire & Freund, 94
    - *Option Trees*, Buntine 92
  - Geração de classificadores heterogéneos
    - Stacked Generalization, Wolpert, 92
    - Cascade Generalization, Gama 98
  - Gerar classificadores usando diferentes atributos
- Modelos Híbridos
  - MCS (Brodley, 95)

Combinação de Modelos Homogéneos

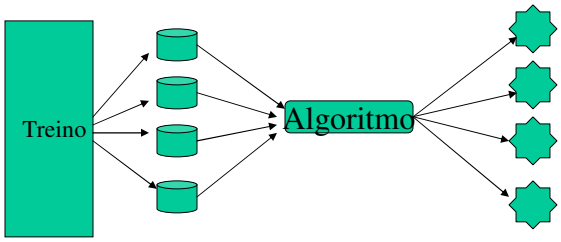
1. Bagging (Boostrap Aggregation)
2. Ada-Boosting (Adaptive boosting)

Bagging

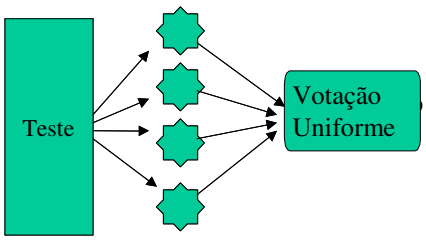
- **Aprendizagem:**
  - Obter *N* réplicas, com reposição, do conjunto de treino.
    - As amostras têm o mesmo numero de exemplos do conjunto de treino.
    - É usual usar 25 amostras.
  - Para cada amostra gerar um classificador.
- **Aplicação**
  - Para cada exemplo de teste
    - Determina a classe predita por cada classificador.
    - As predições são agregadas por voto uniforme.
      - O exemplo é classificado na classe mais votada.

Bagging

- **Aprendizagem:**

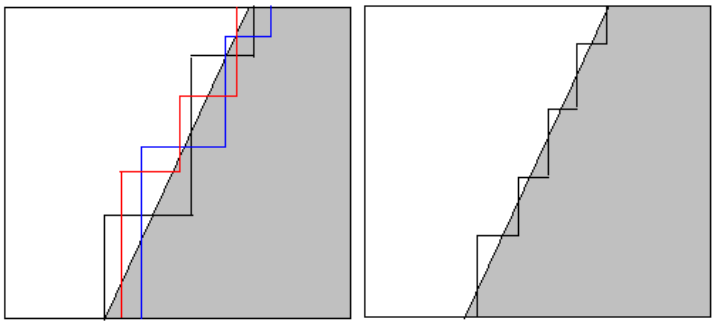


- **Aplicação**



Porque Funciona ?

- Escolhendo o voto maioritário sobre muitos modelos, reduz a variabilidade aleatória dos modelos individuais.
  - Por exemplo, em arvores de decisão
    - A escolha do atributo de teste para um nó
    - A escolha dos pontos de referência nos atributos reais.



Bagging

- Características
  - Requer Algoritmos instáveis.
    - Algoritmos sensíveis a pequenas variações do conjunto de treino.
    - Árvores de Decisão, Redes Neuronais
  - Fácil de implementar com qualquer algoritmo.
  - Fácil de implementar em ambientes paralelos.
- A redução de erro observada é devida á redução na componente da variância. (Breiman 92)

Boosting

- O problema
  - Existirá um algoritmo tal que:
    - Dados:
      - Um nível de confiança: δ ($0 < \delta < 0.5$) e
      - Um limite para o erro: ε ($0 < \varepsilon < 0.5$)
    - O algoritmo gere uma hipótese *h* tal que
      - Com probabilidade 1-δ
      - O $erro_D(h) < \varepsilon$
      - Para qualquer distribuição D dos exemplos?
- *Boosting* é um algoritmo que satisfaz estas condições.

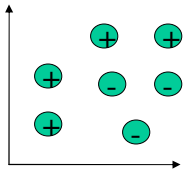
Boosting

- Aprendizagem
  - É um algoritmo iterativo.
  - Associa um peso a cada exemplo.
- Algoritmo:
  - Inicializa o peso de cada exemplo de forma uniforme
  - Iterativamente
    - Gera um classificador usando a actual distribuição dos exemplos.
      - A distribuição é dada pelos pesos
    - Os pesos dos exemplos incorrectamente classificados são incrementados para a iteração seguinte.
  - Os classificadores gerados são agregados por votação pesada.
- Aplicável a qualquer algoritmo de aprendizagem,
  - O algoritmo deverá ser capaz de gerar hipóteses ligeiramente melhores que uma escolha aleatória (*weak learner*).

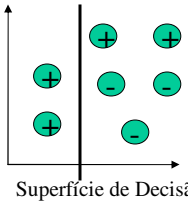
Boosting – Um Exemplo

Weak learner – gera um hiper-plano perpendicular a um dos eixos.

1ª Iteração

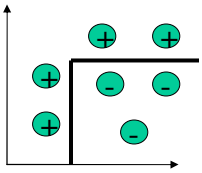


Conjunto de Treino

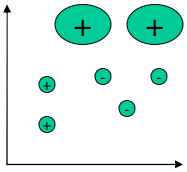


Superfície de Decisão

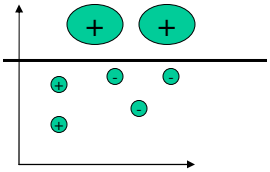
Composição dos 2 classificadores



2ª Iteração



Conjunto de Treino



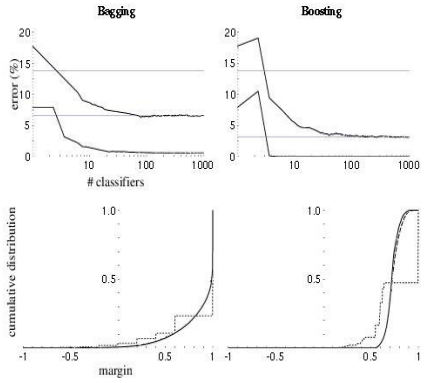
Superfície de Decisão

AdaBoosting

- Input:
  - Conjunto de Dados D,
  - Algoritmo Alg,
  - Nr. de Iterações L_{max}
- Inicializa $w_i = 1/m$ (*i* exemplo, *m* nr. de exemplos)
- Para L=1 até L_{max}
  - $h_L = Alg(D_w)$
  - E_L = Erro de h_L
  - Se $E_L > 0.5$ Ignora h_L
  - Senão
    - $B_L = E_L/(1-E_L)$
    - Para cada i
      - $w_{L+1}(i) = w_L(i)B_L^{1-[h_L(xi) <> yi]}$
- Output
  - $H_f(x) = \text{argmax}_y \text{Sum}_L(\log(1/B_L)[h_L(x)=y]$

Comparação entre *Bagging* e *Boosting*

- *Bagging*
  - Redução do erro devida á variância.
  - Efectivo com classificadores instáveis
    - Não são reportados exemplos de degradação do erro.
- *Boosting*
  - Redução do erro quer na variância quer no *bias.*
  - Em problemas com ruído pode haver degradação da taxa de erro.



Data Streams

João Gama
jgama@liaad.up.pt
University of Porto

The Data Stream Phenomenon

- Highly detailed, automatic, rapid data feeds.
  - Radar: meteorological observations.
  - Satellite: geodetics, radiation,.
  - Astronomical surveys: optical, radio,.
  - Internet: traffic logs, user queries, email, financial,
  - Sensor networks: many more observation points ...
- Most of these data will never be seen by a human!
- Need for near-real time analysis of data feeds.
  - Monitoring, intrusion, anomalous activity Classification,Prediction, Complex correlations, Detect outliers, extreme events, etc

Data Streams

- Continuous flow of data generated at high-speed in Dynamic, Time-changing environments.
- The usual approaches for querying, clustering and prediction use batch procedures cannot cope with this streaming setting.
- Machine Learning algorithms assume:
  - Instances are independent and generated at random according to some probability distribution D.
  - It is required that D is stationary
- Practice: finite training sets, static models.

Data Streams

- We need to maintain Decision models in real time.
- Decision Models must be capable of:
  - incorporating new information at the speed data arrives;
  - detecting changes and adapting the decision models to the most recent information.
  - forgetting outdated information;
- Unbounded traning sets, dynamic models.

Decision Trees from Data Sreams

- Desirable properties:
  - Processing each example
    - Small constant time
    - Fixed amount of main memory
  - Single scan of the data
    - Without (or reduced) revisit old records.
      - Eventually using a sliding window of more recent examples
    - Processing examples at the speed they arrive
  - Classifiers at *anytime*
    - Ideally, produce a model equivalent to the one that would be obtained by a batch data-mining algorithm
  - Ability to detect and react to concept drift

Incremental Decision Trees

- Algorithms using tree re-structuring operators
  - When new information is available splitting-tests are re-evaluated
    - Incremental Induction of Topologically Minimal Trees
      - Walter Van de Velde, 1990
    - Sequential Inductive Learning
      - J.Gratch, 1996
    - Incremental Tree Induction
      - P.Utgoff, 1997
    - Efficient Incremental Induction of Decision Trees
      - D.Kalles, 1995
- Algorithms that do not re-consider splitting-test changes
  - Install a splitting test only when there is evidence enough in favor to that test
    - Very Fast Decision Tree
      - P.Domingos, 2000
    - UFFT (Gama, Medas, SAC04)

Decision Trees from Data Sreams

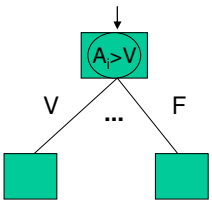
- Algorithms that do not re-consider splitting-test changes
  - Very Fast Decision Trees for Mining High-Speed Data Streams
    - Expand a node only there is evidence enough in favor to a splitting-test
    - Based on Hoeffding bound
    - P. Domingos, G. Hulten; KDD 2000
  - Accurate Decision Trees for Data Streams
    - Extentions to VFDT
      - Continuous attributes
      - Naive Bayes in leaves
      - J.Gama, R. Rocha; KDD 2003

VFDT - Main Algorithm

- *The base Idea:*
  - A small sample can often be enough to choose the optimal splitting attribute
  - Collect sufficient statistics from a small set of examples
  - Estimate the merit of each attribute
  - Use Hoeffding bound to guarantee that the best attribute is really the "best"
    - Statistical evidence that it is better than the second best

VFDT - Main Algorithm

- Input:
  - δ desired probability level.
- Init: Decision Tree = Leaf
- While (TRUE)
  - Read next Example
  - Propagate Example through the Tree
  - Update Sufficient Statistics
  - If leaf(#examples) > N_{min}
    - Evaluate the merit of each attribute
    - If G(A1)-G(A2) > ε
      - Install a splitting test based on A1
      - Expand the tree with two descendent leaves



The Hoeffding bound

- Suppose we have made **n** independent observations of a random variable **r** whose range is **R.**
- The Hoeffding bound states that:
  - With probability 1-δ
  - The true mean of **r** is in the range $\bar{r} \pm \varepsilon$ where

$$\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

  - Independent of the probability distribution generating the examples.

The Hoeffding bound

- The heuristic used to choose test attributes is the information gain G(.)
  - Select the attribute that maximizes the information gain.
  - The range of information gain is log (#classes)
- Suppose that after seeing **n** examples, $G(X_a) > G(X_b) > \ldots > G(X_c)$
- Given a desired δ, the Hoeffding bound ensures that Xa is the correct choice if G(Xa)-G(Xb) > ε.
  - with probability 1- δ

Classifying Test Examples

- To classify a test example
  - The example traverse the tree from the root to a leaf
  - It is classified using the information stored at this leaf.
    - The original VFDT classifies the test example using the majority class.
- VFDT like algorithms store in leaves much more information:
    - The distribution of attribute values per class.
      - Required by the splitting criteria
    - Information collected from hundred's (or thousand's) of examples!
- Can we use this information?
  - Functional Leaves

Functional Leaves

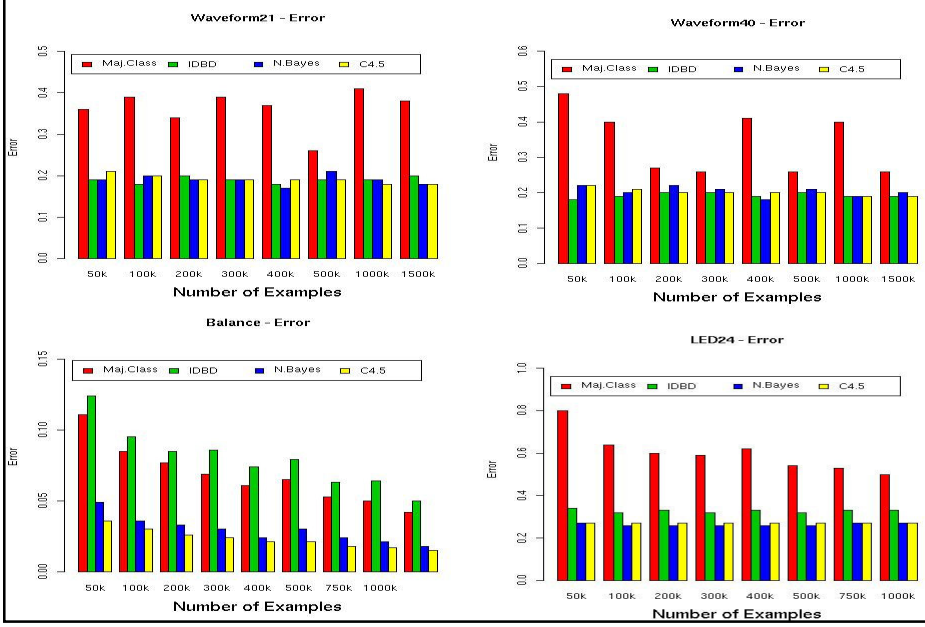
- CART book (Breiman, Freadman, et al)
  - "*grow a small tree using only the most significant splits. Then do multiple regression in each of the terminal nodes*". (pag. 248)
- Perceptron trees
  - P. Utgoff, 1988
- NBTree
  - R. Kohavi, 1996
- Hybrid decision tree learners
  - A. Seewald, 2001
- ...

Why Naive Bayes?

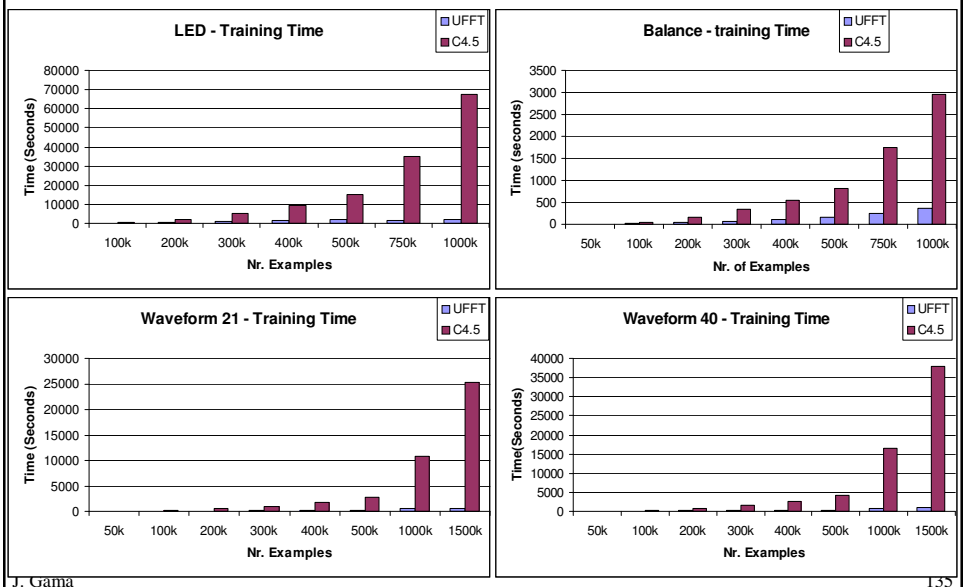
- VFDTc classifies test examples using a naive Bayes algorithm
- Why Naive Bayes?
  - NB can use all the information available at leaves: $P(C_i)$; $P(x_j|C_i)$
    - Is Incremental by nature.
    - Process heterogeneous data, missing values, etc.
  - Assume attributes are independent
    - Can use the splitting criteria sufficient statistics
  - NB is very competitive for small data sets.

$$y = \arg\max_{k \in K} \; P(k) \times \prod_{j} P(a_j = i \mid k)$$

Learning Curves: Error Rate vs. Nr of Examples



Training Time vs. Nr of Examples



VFDT - Analysis

- Low variance models
- Low overfiting
- VFDT becomes asymptotically close to that of a batch learner.
  - The expected disagreement is δ/p; where p is the probability that an example fall into a leaf.

Change Detection

J.Gama, G.Castillo
LIAAD-University of Porto

Motivation

- The Problem:
  - In most challenge applications of machine learning
    - Data flows continuously over time
    - Dynamic Environments
      - Some characteristic properties of the problem can change over time
  - Examples:
    - *e-commerce, user modelling*
    - *Fraud Detection, Intrusion detection*
    - *Monitoring in biomedicine and industrial processes*
- Machine Learning algorithms assume:
  - instances are generated at random according to some probability distribution *D*.
  - Instances are independent and identically distributed
  - It is required that D is stationary

Basic Concepts

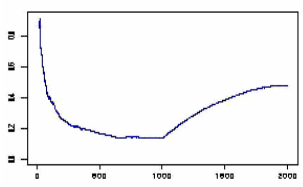
- Concepts are not static, can change over time
- Example:
  - **User Modeling Systems**:
    - to help users to find information
    - to recommend products
    - to adapt an interface, etc.

Can change over time:
- User's needs
- User's preferences
- Characteristics of the environment
- Sensor environment

- We are talking about learning systems
  - **Incremental**
  - **Real Time**
  - **Monitoring**
  - That learn from data streams in dynamic environments
    - instances are generated at random according to some non-stationary probability distribution
  - Hidden Contexts

Concept Drift

- Concept drift means that the concept about which data is obtained may shift from time to time, each time after some minimum permanence.
  - Any change in the distribution underlying the data
  - **Context:** a set of examples from the data stream where the underlying distribution is stationary



The Nature of Change

- The **causes** of change
  - Changes due to modifications in the context of learning due to changes in hidden variables
  - Changes in the characteristic properties of the observed variables.

Detecting Drift

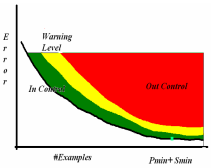
- The Basic Idea:
  - When there is a change in the class-distribution of the examples:
    - The actual model does not correspond any more to the actual distribution.
    - The error-rate increases
- Main Problems:
  - Detect when the actual model is out-date
    - Trace of the error rate
  - React to drift
    - Re-learn the decision model using the most recent examples
      - Dynamic Window
        - » *Short Term Memory*

Detecting Drift

- Suppose a sequence of examples in the form $<x_i,y_i>$
- The actual decision model classifies each example in the sequence
  - In the 0-1 loss function, predictions are either True or False
    - The predictions of the learning algorithm are:
      - T,F,T,F,T,F,T,T,T,F,….
    - A random variable from Bernoulli trials
- The Binomial distribution gives the general form of the probability of observing a F
    - $p_i = (\#F/i)$
    - $S_i = \sqrt{p_i(1-p_i)/i}$
      - *Where i is the number of trials*

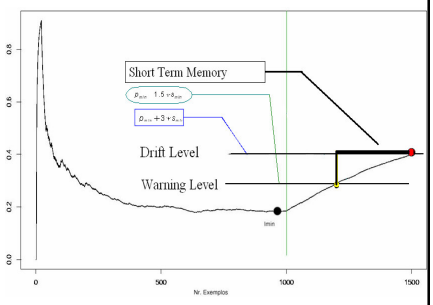
Detect Drift

- The algorithm maintains two registers
  - P_{min} and S_{min} such that $P_{min}+S_{min} = \min(p_i+s_i)$
    - Minimum of the Error rate taking the variance of the estimator into account.
- At example *j*
  - The error of the learning algorithm will be
    - **Out-control** if $p_j+s_j > p_{min}+ \alpha * s_{min}$
    - **In-control** if $p_j+s_j < p_{min}+ \beta * s_{min}$
    - Warning if $p_{min}+ \alpha * s_{min} > p_j+s_j > p_{min}+ \beta * s_{min}$
      - » The constants α and β depend on the confidence level
      - » In our experiments $\beta=2$ and $\alpha = 3$



The Algorithm

- At example *j* the actual model classifies the example
  - Compute the error and variance: p_j and s_j
  - If the error is
    - In-control the actual model is updated
      - Incorporate the example in the decision model
    - Warning zone:
      - Maintain the actual model
      - First Time:
        - » the lower limit of the window is: $L_{warning} = j$
    - Out-Control
      - Re-learn a new model using as training set the set of examples $[L_{warning}, j]$



Data streams: sequences of contexts

- Whenever a new context is detect
  - The actual decision is forgotten
  - A new decision model is learn
    - Using the examples is the short term memory.
      - The most recent examples
- The process of monitoring the error is re-initialized

Open Issues in Learning from Data Streams

- Continuous flow of data ! Models evolve over time;
  - Online, Anytime, and Real-Time Learning;
  - Incremental and Decremental Issues
  - Cost-Performance Management
- Incorporate Change Detection Mechanisms into the Learning Algorithms;
- Online Feature Selection and Pre-processing;
- Evolving Feature Spaces;
- Dynamic models: Which Evaluation Methods and Metrics