



Tecnologias e Sistemas de Informação

Sistemas Inteligentes

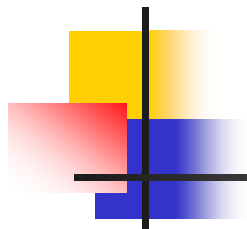
Aula 4 - Classificação e Regras de Classificação

Prof. José Artur Quilici-Gonzalez
Email: jose.gonzalez@ufabc.edu.br



Roteiro

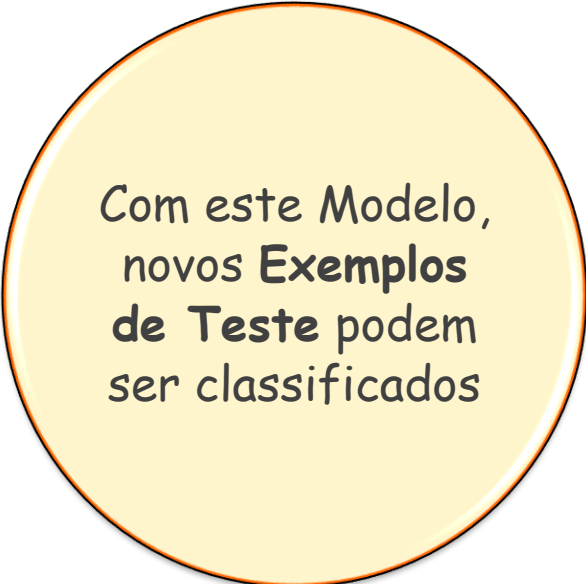
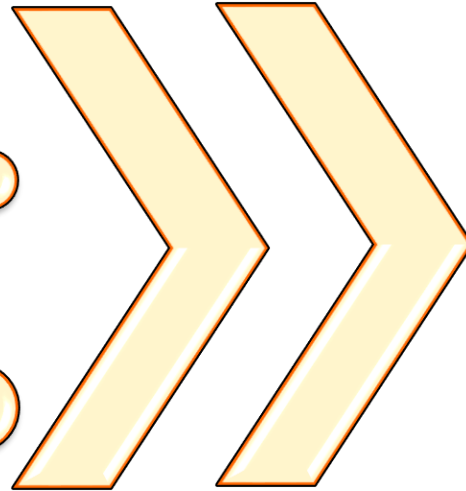
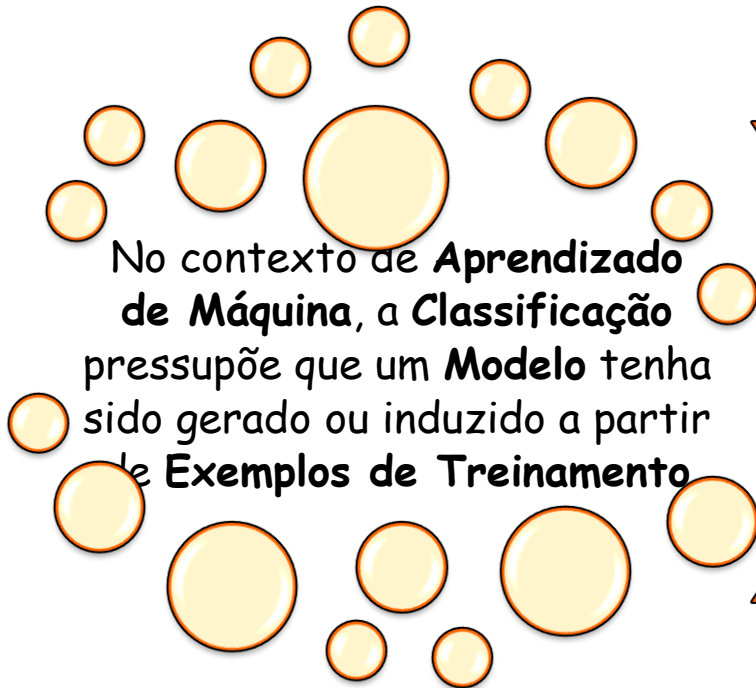
- Introdução
- Classificação
- Algoritmos de Aprendizado
 - Algoritmo oneR e PRISM
- Avaliação dos Resultados
- Avaliação do Desempenho do Classificador
 - Método da Ressubstituição
 - Método da Divisão da Amostra
 - Método da Validação Cruzada
 - Método Deixe-Um-De-Fora
- Considerações Finais
- Referência Bibliográfica



INTRODUÇÃO



Contexto

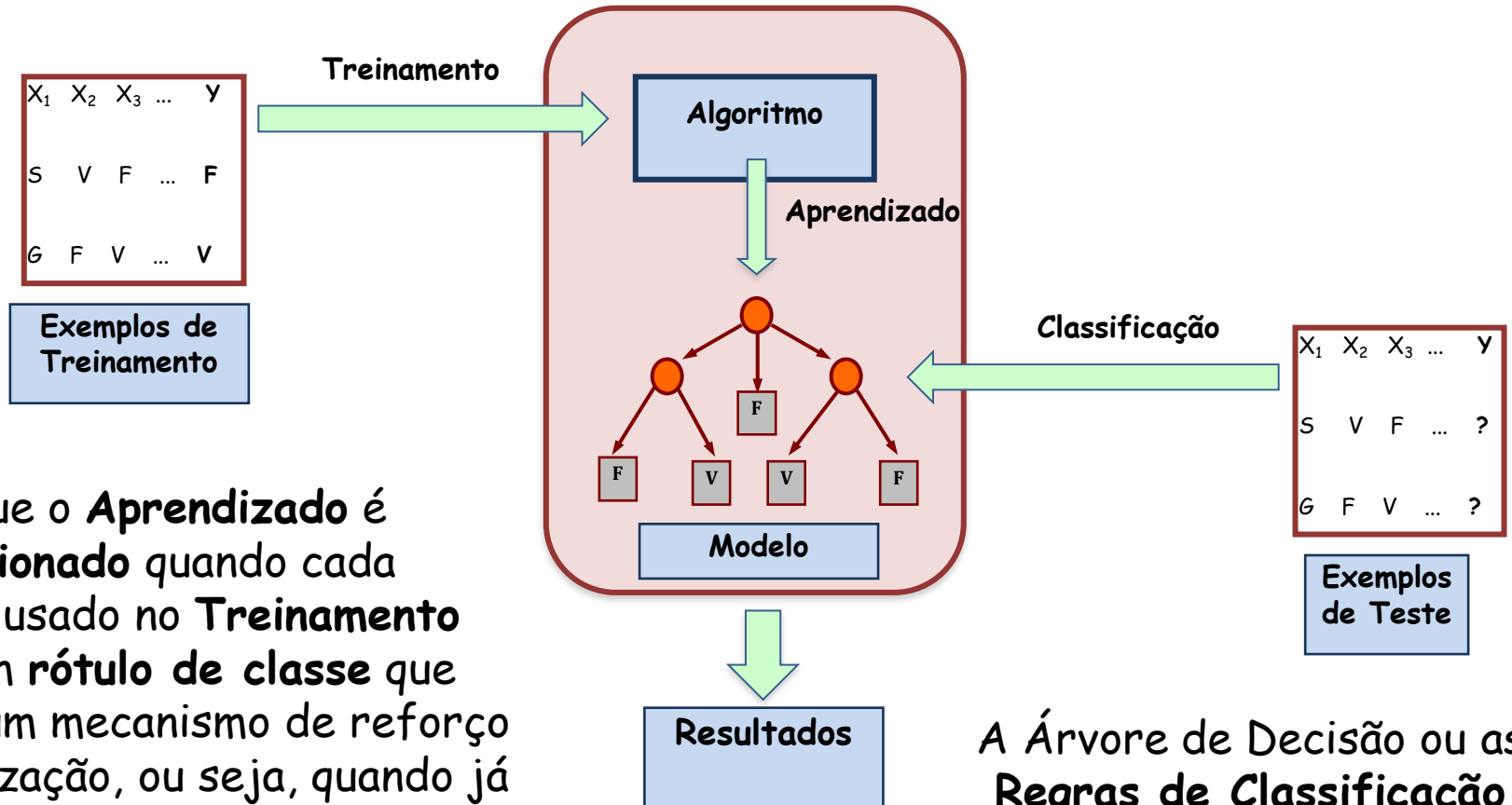


Com este Modelo, novos **Exemplos de Teste** podem ser classificados

Há várias maneiras de representar o conhecimento embutido num Modelo, sendo as mais comuns **Árvores de Decisão** e **Regras de Classificação**

Treinamento, Aprendizado e Classificação

Sistema Inteligente Simples



Diz-se que o **Aprendizado** é **Supervisionado** quando cada Exemplo usado no **Treinamento** possui um **rótulo de classe** que orienta um mecanismo de reforço ou penalização, ou seja, quando já se sabe antecipadamente a qual classe determinado elemento pertence

A **Árvore de Decisão** ou as **Regras de Classificação** resultante(s) representa(m) o **Modelo gerado** ou o **Conceito aprendido**



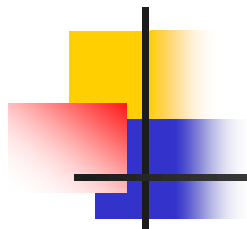
Forma das Regras

As Regras de Classificação assumem a forma genérica:

IF Condição **THEN** Valor

sendo "Valor" um resultado discreto, como sim/não, baixo/médio/alto verdadeiro/falso etc.

Quando os resultados esperados não pertencem a classes discretas, ou seja, quando "Valor" for uma variável real, a classificação recebe o nome de **Regressão**



Classificação

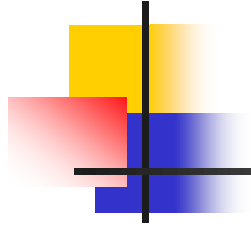


Exemplos de Treinamento

O **Aprendizado Supervisionado** se dá através de um **Algoritmo de Aprendizado**, cuja função é criar uma representação do conhecimento extraído de um conjunto de **Exemplos de Treinamento**

Os Exemplos de Treinamento, por sua vez, são uma forma conveniente de estruturar os dados de uma empresa ou de um certo domínio do saber

Todos os Exemplos de Treinamento têm o **mesmo número de atributos**, sendo a diferença entre eles representada pelos valores que cada atributo assume



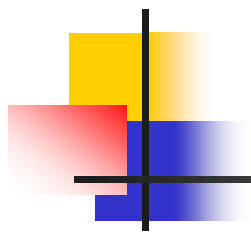
ALGORITMOS DE APRENDIZADO



Diferentes Algoritmos

Um **Algoritmo de Aprendizado** pode variar desde aqueles que

- simplesmente escolhem um dos atributos do Conjunto de Treinamento como a resposta possível a um teste, caso do algoritmo **oneR** (uma Regra),
- passando por aqueles cuja resposta a um novo teste é uma combinação linear dos valores dos atributos,
- até a utilização de complicados modelos não-lineares, como as **Redes Neurais**, ou
- o aprendizado estatístico das **Máquinas de Vetor de Suporte**, ou **Support Vector Machines, SVM**



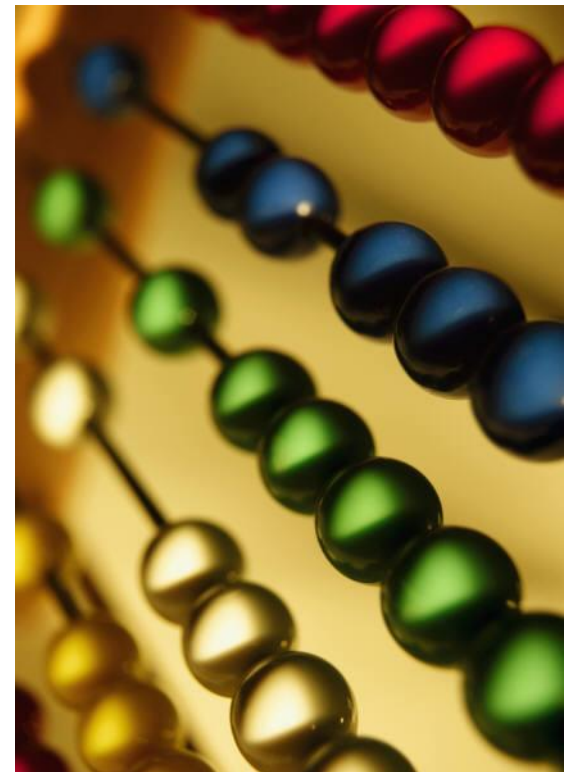
ALGORITMO oneR ou 1R

Algoritmo oneR

Possivelmente o **Algoritmo de Aprendizado para classificação** mais simples e, no entanto, seu desempenho pode ser surpreendentemente bom, dependendo da Base de Dados

Esse algoritmo aposta na hipótese de que basta consultar apenas um dos atributos para classificar corretamente os Exemplos de Teste

A tarefa então do algoritmo oneR é encontrar durante o treinamento o atributo que apresenta a menor taxa de erros de classificação



Algoritmo oneR Aplicado na Tabela do Tempo

Dia	Temperatura	Umidade	Vento	Partida
Ensolarado	Elevada	Alta	Falso	Não
Ensolarado	Elevada	Alta	Verdadeiro	Não
Nublado	Elevada	Alta	Falso	Sim
Chuvoso	Amena	Alta	Falso	Sim
Chuvoso	Baixa	Normal	Falso	Sim
Chuvoso	Baixa	Normal	Verdadeiro	Não
Nublado	Baixa	Normal	Verdadeiro	Sim
Ensolarado	Amena	Alta	Falso	Não
Ensolarado	Baixa	Normal	Falso	Sim
Chuvoso	Amena	Normal	Falso	Sim
Ensolarado	Amena	Normal	Verdadeiro	Sim
Nublado	Amena	Alta	Verdadeiro	Sim
Nublado	Elevada	Normal	Falso	Sim
Chuvoso	Amena	Alta	Verdadeiro	Não

Primeiramente isolamos um dos atributos, digamos "Dia", e verificamos qual a distribuição das classes "Sim" e "Não" no atributo de saída "Partida"

oneR para o Atributo "Dia"

Dia	Partida
Ensolarado	Sim
Ensolarado	Sim
Ensolarado	Não
Ensolarado	Não
Ensolarado	Não
Nublado	Sim
Nublado	Sim
Nublado	Sim
Chuvoso	Sim
Chuvoso	Sim
Chuvoso	Sim
Chuvoso	Não
Chuvoso	Não

Valor do Atributo	Exemplos com Partida=Não	Exemplos com Partida=Sim	Maioria	Erros
Ensolarado	3	2	Não	2/5
Nublado	0	4	Sim	0/4
Chuvoso	2	3	Sim	2/5
Total de Erros				4/14

Vamos considerar como "sucesso" a classe ("Sim" ou "Não") que aparecer com maior frequência, i.e., a maioria, para cada uma das opções possíveis ("Ensolarado", "Nublado", "Chuvoso") do atributo "Dia", e como "erro" a menos frequente

Podemos gerar algumas regras de classificação iniciais:

IF Dia=Ensolarado **THEN** Partida=Não
IF Dia=Nublado **THEN** Partida=Sim
IF Dia=Chuvoso **THEN** Partida=Sim

Taxa de Erros dos Atributos

Atributo	Regras	Erros	Total de Erros
Dia	Ensolarado → Não	2/5	4/14
	Nublado → Sim	0/4	
	Chuvoso → Sim	2/5	
Temperatura	Elevada → Não	2/4	5/14
	Amena → Sim	2/6	
	Baixa → Sim	1/4	
Umidade	Alta → Não	3/7	4/14
	Normal → Sim	1/7	
Vento	Falso → Sim	2/8	5/14
	Verdadeiro → Não	3/6	

Os atributos "Dia" e "Umidade" apresentam as menores taxas de erros



Regras do Algoritmo oneR

Adotando qualquer critério arbitrário de desempate, vamos ficar com o conjunto de erros gerados pelo atributo "Umidade" e gerar as seguintes Regras de Classificação:

IF Umidade=Alta **THEN** Partida=Não

IF Umidade=Normal **THEN** Partida=Sim

Portanto, quando o algoritmo **oneR** tiver que classificar um novo exemplo, somente o atributo "Umidade" será considerado, e o resultado será baseado nas duas Regras de Classificação mostradas acima



Treinamento X Teste

Se os **Exemplos de Treinamento** forem usados como **Exemplos de Teste**, e supondo que entre os 14 Exemplos não haja contradição entre si, o algoritmo **oneR** deve acertar 10 vezes e errar 4

Mas, e se o **Conjunto de Teste** for diferente do **Conjunto de Treinamento**?

Não será a estimativa de 10 acertos e 4 erros demasiadamente otimista ou ela se confirmará com os novos dados?

E se o número de **Exemplos de Treinamento** tivesse sido 140, em vez de 14, que implicações isso teria?



Algoritmo PRISM

Há outros algoritmos de classificação bem mais refinados que o **oneR** e que, na maioria dos casos, produzem resultados com taxa de sucesso mais elevada

Um dos algoritmos de classificação mais famosos é o **PRISM**, que utiliza o princípio de "cobertura", i.e., ele vai criando regras que se aplicam ao maior número possível de exemplos do Conjunto de Treinamento, até que toda a tabela esteja "coberta" pelas regras produzidas

Seu desenvolvimento foi inspirado nos "pontos fracos" do algoritmo de indução de Árvores de Decisão ID3, como a dificuldade de entender as árvores muito grandes e complexas geradas pelo algoritmo ID3



Desempenho de um Modelo

Não vamos aqui nos deter em particularidades do **PRISM** porque os resultados gerados pelo oneR são suficientemente representativos para os nossos propósitos, de abordar os métodos de avaliação dos resultados produzidos pelo modelo induzido

E ao avaliarmos os resultados, estamos de certa forma avaliando a capacidade de predição de um modelo para determinada Base de Dados

Abordar um modelo pelo seu desempenho é interessante porque há evidências empíricas de que nenhum algoritmo tem desempenho superior aos demais para qualquer Base de Dados

A estrutura interna do conjunto de dados desempenha um papel decisivo no desempenho do algoritmo e na qualidade dos resultados



Avaliação dos Resultados

Predição e Diagnóstico

Após o treinamento para gerar um Modelo através do Algoritmo de Aprendizado, é de grande importância fazer uma avaliação do desempenho do modelo

Interessa saber quão preditivo é o Modelo Aprendido

Há várias metodologias consagradas para este fim. Vamos iniciar nossa abordagem ao tema lembrando a **Acurácia** de um Classificador Binário





Combinações de Respostas

Se as respostas possíveis para um diagnóstico forem "Positivo" e "Negativo", quatro combinações de **resultados previstos** e **resultados reais** podem ocorrer

Se o paciente for portador da doença e o médico acertar no diagnóstico, dizemos que este caso é um **Verdadeiro Positivo** ou VP

Se o paciente não for portador da doença e o médico acertar no diagnóstico, dizemos que este caso é um **Verdadeiro Negativo** ou VN

Se o paciente for portador da doença, e o médico errar no diagnóstico afirmando que ele está são, dizemos que este caso é um **Falso Negativo** ou FN

Se o paciente não for portador da doença, e o médico errar no diagnóstico dizendo que ele está doente, dizemos que este caso é um **Falso Positivo** ou FP



Matriz de Confusão

As quatro combinações possíveis de resultados costumam ser representadas por uma matriz que recebe o nome de "Matriz de Confusão"

	Positivo Previsto	Negativo Previsto
Positivo Real	Verdadeiro Positivo (VP)	Falso Negativo (FN)
Negativo Real	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Os valores contidos numa Matriz de Confusão podem ser utilizados para avaliar o desempenho de uma Árvore de Decisão

O que se espera nos resultados é que os casos positivos sejam classificados como positivos e os negativos como negativos, ou seja, o desejável é que as taxas de sucesso para Verdadeiro Positivo e Verdadeiro Negativo sejam altas, e que as taxas de Falso Positivo e Falso Negativo sejam baixas



Precisão ou Acurácia

A Precisão ou Acurácia do Classificador se expressa pelo número de classificações corretas (VP+VN) divididas pelo número total de classificações (VP+VN+FP+FN)

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \times 100\%$$

Ocorre que em situações reais o custo de um **Falso Positivo** pode não ser igual ao de um **Falso Negativo**, e a Acurácia não consegue captar adequadamente essa situação de interesse



Falso Positivo e Falso Negativo

Suponha que um classificador usado para Detecção de Anomalia tenha que atribuir a cada um dos 100 testes um rótulo de "Situação=Normal" ou "Situação=Anormal"

Suponha ainda que a relação entre donos honestos de cartão de crédito e golpistas seja de 96 para 4 e o classificador tenha colocado 99 portadores de cartão na classe "Normal" e apenas um dos golpistas na classe "Anormal"

Neste caso a Acurácia do Classificador será de

$$Acurácia_{classif} = \frac{96 + 1}{96 + 1 + 3 + 0} \times 100\% = 97\%$$

A interpretação baseada apenas na Acurácia indicaria um excelente desempenho, mas na realidade este é um péssimo classificador para uma operadora de cartões de crédito, porque seu interesse em detectar os golpistas é bem maior do que os donos honestos de cartão!



Taxa de Verdadeiro Negativo

Para detectar estes casos de conjuntos não-balanceados de Falso Positivo e Falso Negativo, podemos definir a **Taxa de Verdadeiro Negativo**, também conhecida por **Especificidade**, como sendo o número de Verdadeiro Negativo (VN) dividido pelo número total de negativos, que é a soma de Verdadeiro Negativo (VN) mais Falso Positivo (FP)

$$Taxa_{VN} = \frac{VN}{VN + FP} \times 100\%$$



Cálculo de Taxa de VN

Se o indicador Taxa de Verdadeiro Negativo for utilizado para o caso citado dos cartões de crédito, teremos uma Taxa de Verdadeiro Negativo de

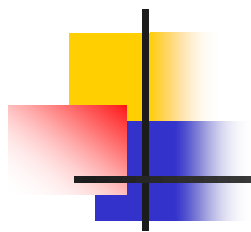
$$Taxa_{VN} = \frac{1}{1 + 3} \times 100\% = 25\%$$



Taxa de Verdadeiro Positivo

Para outras situações, pode ser mais conveniente utilizar a **Taxa de Verdadeiro Positivo**, também conhecida por **Sensibilidade**, como sendo o número de Verdadeiro Positivo (VP) dividido pelo número total de positivos, que é a soma de Verdadeiro Positivo (VP) mais Falso Negativo (FN)

$$Taxa_{VP} = \frac{VP}{VP + FN} \times 100\%$$



Avaliação de Desempenho do Classificador



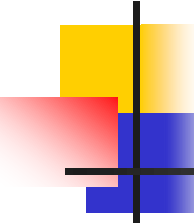
Métodos de Avaliação

Com estes indicadores em mente, suponha que se queira avaliar qual será o desempenho do Modelo gerado pelo Algoritmo de Aprendizado para determinada Base de Dados

Se utilizarmos o mesmo Conjunto de Treinamento como Conjunto de Teste, muito possivelmente a estimativa de desempenho resultará excessivamente otimista para testes reais, com novos Conjuntos de Teste

Outra alternativa é reservar parte do Conjunto de Treinamento para ser usada como Conjunto de Teste

Mas qual o tamanho ideal da partição do conjunto de Exemplos de Treinamento? E como escolher os elementos deste subconjunto do Conjunto de Treinamento que serão usados para teste? E se o Conjunto de Teste for muito pequeno?



Método da Ressubstituição ou "Use training set"

O **Conjunto de Treinamento** é também utilizado como **Conjunto de Teste**

Se o Conjunto de Treinamento for uma amostra representativa do universo do problema, suas estimativas de desempenho para um Conjunto de Teste composto por Exemplos não vistos anteriormente podem ser muito boas

Conjunto de Treinamento

Conjunto de Teste

Caso contrário, o modelo poderá apresentar muitos erros de generalização durante os testes, seja por problemas de excesso de complexidade do Modelo, que costuma causar **overfitting**, ou por **Poda** inadequada

O fato de o conjunto completo de treinamento ser usado para gerar o Modelo constitui uma vantagem sobre os métodos de reamostragem, principalmente se o número de Exemplos de Treinamento for pequeno



"Use training set" em Números

Na simulação Weka da Tabela do Tempo, com o algoritmo **oneR** usando o método "Use training set", o número de instâncias ou exemplos classificados corretamente foi 10 (71%), e 4 (29%) classificados incorretamente, enquanto que o **PRISM** classificou todos os Exemplos corretamente

A Matriz de Confusão para os 14 Exemplos é a seguinte:

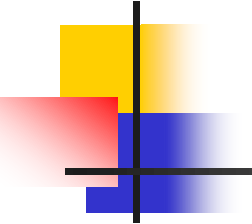
	Não Previsto	Sim Previsto
Não Real	3	2
Sim Real	2	7

oneR

	Não Previsto	Sim Previsto
Não Real	5	0
Sim Real	0	9

PRISM

Obs.: Tente reproduzir estes números no Weka



Método da Substituição da Amostra ou “Holdout” ou “Percentage Split”

Consiste na divisão dos Exemplos de Treinamento em **dois conjuntos disjuntos**, um para **Treinamento**, outro para **Teste**

O valor de divisão mais comum é 66% para Treinamento e 34% para Teste, embora não haja evidências empíricas que justifiquem essa escolha de 2/3 e 1/3

Sua vantagem é a simplicidade, mas dependendo da composição obtida, as classes dos Exemplos podem não estar igualmente representadas nos dois conjuntos

Conjunto de Treinamento	Conjunto de Teste
-------------------------	-------------------

Outra limitação desse método está no fato de que menos Exemplos são usados no Treinamento, podendo ter um impacto negativo no desempenho do Modelo induzido

"Percentage split" em Números

Para Conjuntos de Teste excessivamente pequenos, dividir o já escasso número de Exemplos de Teste pode ter um efeito desastroso ou na geração do Modelo ou na sua avaliação de desempenho

Na simulação Weka da Tabela do Tempo com o oneR usando o método da "Percentage Split", com o Conjunto de Treinamento correspondendo a 66%, o número de exemplos classificados corretamente foi 2 (40%), e 3 (60%) classificados incorretamente!

No PRISM com o método "Percentage split" e o Conjunto de Treinamento correspondendo a 66%, o número de exemplos classificados corretamente foi 4 (80%), e 1 (20%) classificado incorretamente

	Não Previsto	Sim Previsto
Não Real	0	2
Sim Real	1	2

oneR

	Não Previsto	Sim Previsto
Não Real	1	1
Sim Real	0	3

PRISM

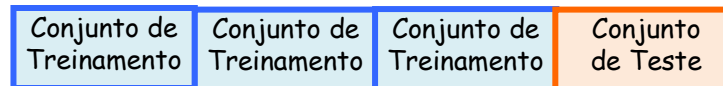


Método da Validação Cruzada ou "Cross Validation"

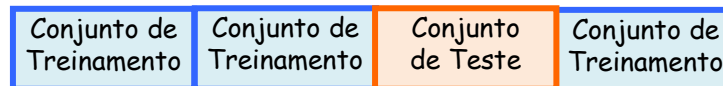
Os Exemplos de Teste são aleatoriamente divididos em k partições mutuamente exclusivas ou "*folds*", sendo k normalmente igual a 10

A cada iteração um desses *folds* será usado como Conjunto de Teste, enquanto que os outros serão usados para Treinamento

Iteração 1



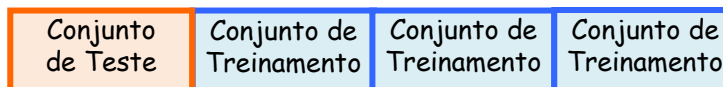
Iteração 2



Iteração 3



Iteração 4





"Cross validation" em Números

Na simulação Weka da Tabela do Tempo com o algoritmo *oneR* usando o método da "Cross-validation", com $k = 10$ folds, o número de exemplos classificados corretamente foi 5 (36%), e 9 (66%) classificados incorretamente!

Para o *PRISM* usando o método da "Cross-validation", com $k = 10$ folds, o número de instâncias ou exemplos classificados corretamente foi 12 (86%), e 0 (0%) classificados incorretamente!

	Não Previsto	Sim Previsto
Não Real	3	2
Sim Real	7	2

oneR

	Não Previsto	Sim Previsto
Não Real	5	0
Sim Real	0	7

PRISM

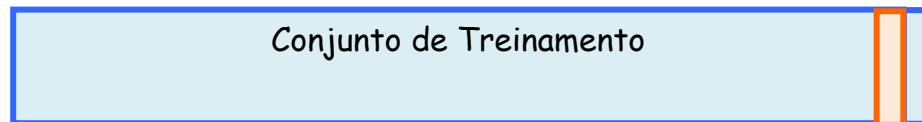
Método Deixe-Um-De-Fora ou "Leave One Out"

É um caso especial do Método da Validação Cruzada em que o número de partições k é igual ao número de Exemplos N , isto é, $k = N$, e cada partição é composta por apenas um Exemplo

Iteração 1



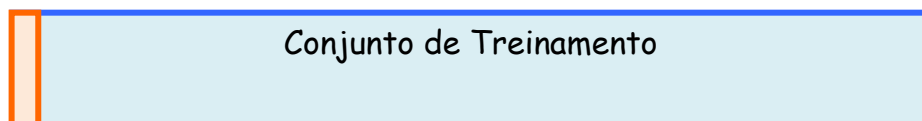
Iteração 2



Iteração H



Iteração N



A vantagem é que mais dados são usados para o Treinamento, mas a desvantagem é seu custo computacional para os casos em que N for muito grande

"Leave-One-Out" em Números

Na simulação Weka da Tabela do Tempo com o algoritmo oneR usando o método da "Cross-validation", com $k = 14$ folds, número idêntico ao de Exemplos, portanto $k = N$, o resultado da classificação foi idêntico ao da "Cross-validation", sendo o número de exemplos classificadas corretamente 5 (36%), e 9 (66%) classificados incorretamente!

Repetindo o mesmo Conjunto de Teste, porém com o Algoritmo PRISM, usando o método da "Cross-validation", com $k = 14$ folds, o resultado da classificação foi idêntico ao da "Cross-validation", sendo o número de exemplos classificadas corretamente 11 (79%), 0 (0%) classificados incorretamente e 3 exemplos não classificados

	Não Previsto	Sim Previsto
Não Real	3	2
Sim Real	7	2

oneR

	Não Previsto	Sim Previsto
Não Real	5	0
Sim Real	0	6

PRISM

Considerações Finais

Nesta unidade, vimos um algoritmo simples, conhecido como **oneR**, usado para gerar **Regras de Classificação**

Outros algoritmos mais refinados usam princípios mais complexos, como o de cobertura, para produzir **Regras de Classificação** (caso do PRISM)

Foram apresentados alguns indicadores que auxiliam o usuário a decidir se os resultados obtidos são satisfatórios ou não

Também foram apresentados alguns métodos para estimar o desempenho futuro do classificador em situação de testes reais



Referência Bibliográfica



Referência Bibliográfica

- CENDROWSKA, J. **PRISM: An Algorithm for Inducing Modular Rules**. International Journal of Man-Machine Studies. Vol. 27, pages 349-370, 1987. In <http://sci2s.ugr.es/keel/pdf/algorithm/articulo/1987-Cendrowska-IJMMS.pdf>. Acessado em 06.03.2013.
- QUINLAN, J. R. **Induction of Decision Trees**. Machine Learning, Vol. 1, No. 1, pp. 81-106. Boston: Kluwer Academic Publishers, 1986.
- REZENDE, S. O. (Organizadora). **Sistemas Inteligentes: Fundamentos e Aplicações**. Barueri: Editora Manole Ltda, 2005.
- ROCHA, M.; CORTEZ, P. & NEVES, J. M. **Análise Inteligente de Dados: Algoritmos e Implementação em Java**. Lisboa: Editora de Informática, 2008.
- TAN, P.N.; STEINBACH, M. & KUMAR, V. **Introdução ao Data Mining Mineração de Dados**. Rio de Janeiro: Editora Ciência Moderna Ltda., 2009.
- WITTEN, I. H. & FRANK, E. **Data Mining: Practical Machine Learning Tools and Techniques**. Second Edition. Amsterdam: Morgan Kaufmann Publishers, 2005.