

Aprendizagem de Dados Simbólicos e/ou Numéricos

Francisco de A.T. de Carvalho

Regras de Classificação 1/

✉ Algoritmos de Cobertura

✉ Uma árvore de decisão pode ser convertida em um conjunto de regras

✉ conversão direta: conjunto de regras complexo

✉ conversão mais elaborada não é trivial

Regras de Classificação 2/

✉ Algoritmos de Cobertura

✉ Abordagem para gerar diretamente um conjunto de regras:

✉ considere uma classe de cada vez e encontre o conjunto de regras que cobrem todas as suas instâncias

✉ excluindo as instancias das outras classes

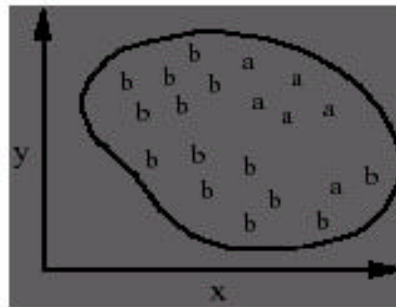
Regras de Classificação 3/

✉ Algoritmos de Cobertura

✉ Essa abordagem se chama “cobertura”

✉ em cada estágio identifica-se uma regra que “cobre” parte das instancias da classe considerada

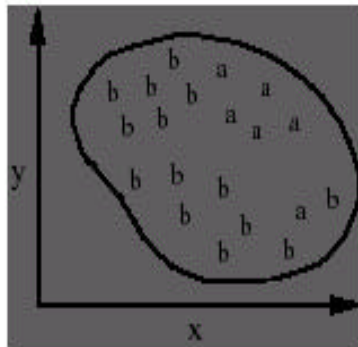
✉ Exemplo



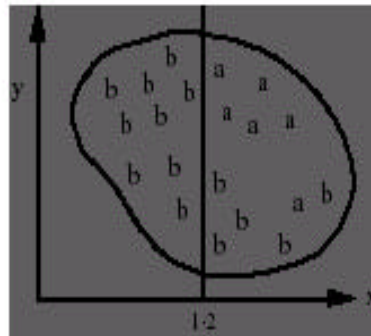
Regras de Classificação 4/

✉ Algoritmos de Cobertura

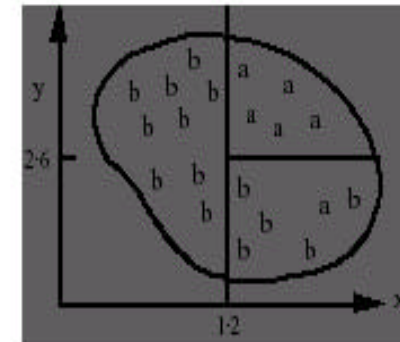
✉ Exemplo: gerando uma regra para a classe A



If true then class = a



If $x > 1.2$ then class = a

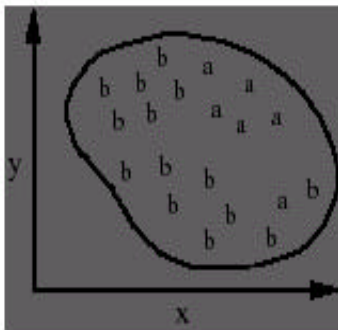


If $x > 1.2$ and $y > 2.6$ then class = a

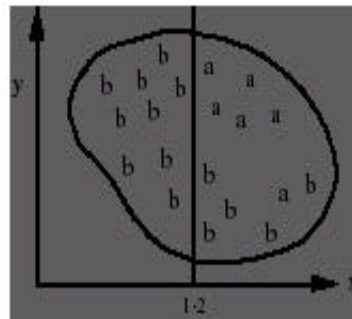
Regras de Classificação 5/

✉ Algoritmos de Cobertura

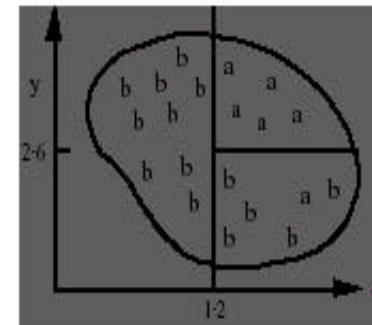
✉ Exemplo: gerando uma regra para a classe B



If true then classe = b



If age ≤ 30 then classe = b

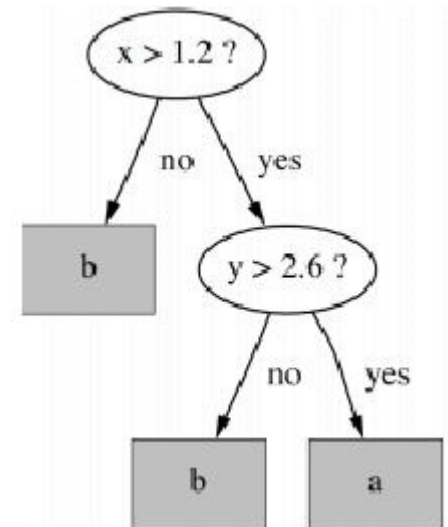


If age > 30 and height ≤ 170
then classe = b

Regras de Classificação 6/

✉ Regras × árvores

- ✉ Aprendizagem de árvore: leva em conta todas as classes ao mesmo tempo;
aprendizagem de regras: concentra-se em uma classe de cada vez
- ✉ Conjunto de regras pode ser mais claro quando árvore tem réplica de sub-árvore
- ✉ Conjunto de regras: o teste é selecionado maximizando a cobertura de uma classe;
Árvore: o teste é selecionado maximizando o ganho de informação de um atributo

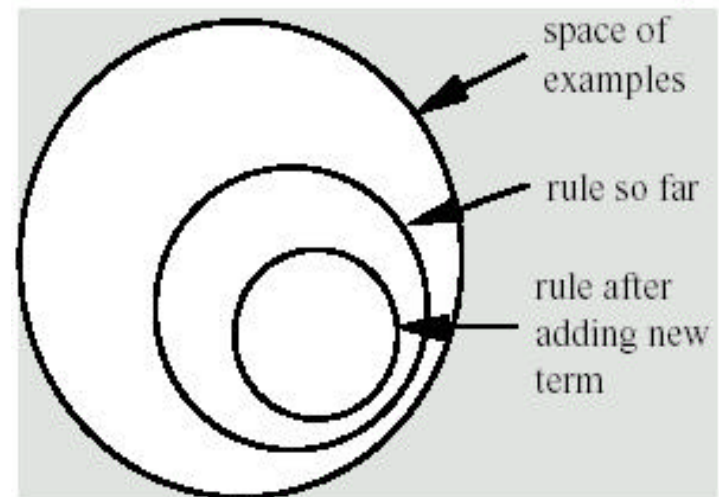


Regras de Classificação 7/

✉ Algoritmos de Cobertura: algoritmo básico

✉ Para cada classe C gera uma regra adicionando testes que maximizem a precisão da regra: capacidade de cobertura

✉ cada novo teste reduz a cobertura da regra



Regras de Classificação 8/

✉ Seleção de um teste

✉ Objetivo: maximizar a precisão

✉ t: número total de exemplos cobertos pela regra

✉ p: exemplos positivos da classe cobertos pela regra

✉ t-p: número de erros realizados pela regra

✉ Seleção: escolher o teste que maximiza a razão p/t

✉ Critério de parada: quando $p/t = 1$ ou não existem mais exemplos

Regras de Classificação 9/

✉ Exemplo: dados de tempo

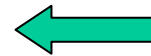
outlook	temperatura	umidade	ventando	joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
nublado	quente	alta	falso	sim
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
nublado	fria	normal	verdadeiro	sim
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim
nublado	moderada	alta	verdadeiro	sim
nublado	quente	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe não

Se ? Então resposta = não

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	3/5
	nublado	2/4
	chuvoso	0/5
Temperatura	quente	2/4
	moderada	2/6
	fria	1/4
Umidade	alta	4/7
	normal	1/7
Ventando	falso	2/8
	verdadeiro	3/6



Conjunto de Regras : classe não

Regra modificada

Se Outlook = ensolarado Então resposta = não

Exemplos cobertos pela regra modificada


outlook	temperatura	umidade	Ventando	Joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim

Conjunto de Regras : classe não

Se Outlook = ensolarado e ? Então resposta = não

Possíveis Testes

Atributo	Valor	
Temperatura	quente	2/2
	moderada	1/2
	fria	0/1
Umidade	alta	3/3
	normal	0/2
Ventando	falso	2/3
	verdadeiro	1/2



Conjunto de Regras : classe não

Regra Final

Se Outlook = ensolarado e Umidade = alta Então resposta = não

Próxima regra

Se ? Então resposta = não

- ❑ Excluir do conjunto total os exemplos que satisfazem a regra construída previamente.
- ❑ Repetir a etapa anterior até que não exista mais exemplos da classe não

Conjunto de Regras : classe não

Conjunto Reduzido

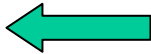
outlook	temperatura	umidade	ventando	joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
nublado	quente	alta	falso	sim
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
nublado	fria	normal	verdadeiro	sim
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim
nublado	moderada	alta	verdadeiro	sim
nublado	quente	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe não

Se ? Então resposta = não

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	0/2
	nublado	0/4
	chuvoso	2/5
Temperatura	quente	0/2
	moderada	1/5
	fria	1/4
Umidade	alta	1/4
	normal	1/7
Ventando	falso	0/6
	verdadeiro	2/5



Conjunto de Regras : classe não

Regra modificada

Se Outlook = chuvoso Então resposta = não

Exemplos cobertos pela regra modificada


outlook	temperatura	umidade	ventando	joga
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
chuvoso	moderada	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe não

Se Outlook = chuvoso e ? Então resposta = não

Possíveis Testes

Atributo	Valor	
Temperatura	moderada	1/3
	fria	1/2
Umidade	alta	1/2
	normal	1/3
Ventando	falso	0/3
	verdadeiro	2/2



Conjunto de Regras : classe não

Regra Final

Se Outlook = chuvoso e Ventando = verdadeiro Então
resposta = não

- ❑ Não existem mais exemplos da classe não e o processo para.
- ❑ O conjunto de regras da classe não é dado por:

Se Outlook = ensolarado e Umidade = alta
Então resposta = não

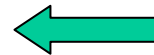
Se Outlook = chuvoso e Ventando = verdadeiro
Então resposta = não

Conjunto de Regras : classe sim

Se ? Então resposta = sim

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	2/5
	nublado	4/4
	chuvoso	3/5
Temperatura	quente	2/4
	moderada	4/6
	fria	3/4
Umidade	alta	3/7
	normal	6/7
Ventando	falso	6/8
	verdadeiro	3/6



Conjunto de Regras : classe sim

Regra Final

Se Outlook = nublado Então resposta = sim

Próxima regra

Se ? Então resposta = sim

- ❑ Excluir do conjunto total os exemplos que satisfazem a regra construída previamente.
- ❑ Repetir a etapa anterior até que não exista mais exemplos da classe não

Conjunto de Regras : classe sim

Conjunto Reduzido

outlook	temperatura	umidade	vento	joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
nublado	quente	alta	falso	sim
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
nublado	fria	normal	verdadeiro	sim
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim
nublado	moderada	alta	verdadeiro	sim
nublado	quente	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe sim

Se ? Então resposta = sim

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	3/5
	chuvoso	2/5
Temperatura	quente	0/2
	moderada	3/5
	fria	2/3
Umidade	alta	1/5
	normal	4/5
Ventando	falso	4/6
	verdadeiro	1/4



Conjunto de Regras : classe sim

Regra modificada

Se **Umidade = normal** Então resposta = sim

Exemplos cobertos pela regra modificada

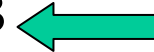
outlook	temperatura	umidade	ventando	joga
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim

Conjunto de Regras : classe sim

Se **Umidade** = **normal** ? Então
resposta = sim

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	2/2
	chuvoso	2/3
Temperatura	moderada	2/2
	fria	2/3
Ventando	falso	3/3
	verdadeiro	1/2



Conjunto de Regras : classe sim

Regra Final

Se **Umidade= normal** e **Ventando= falso**
Então resposta = sim

Próxima regra

Se ? Então resposta = sim

- ❑ Excluir do conjunto total os exemplos que satisfazem a regra construída previamente.
- ❑ Repetir a etapa anterior até que não exista mais exemplos da classe não

Conjunto de Regras : classe sim

Conjunto Reduzido


outlook	temperatura	umidade	ventando	joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
nublado	quente	alta	falso	sim
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
nublado	fria	normal	verdadeiro	sim
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim
nublado	moderada	alta	verdadeiro	sim
nublado	quente	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe sim

Se ? Então resposta = sim

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	1 / 4
	chuvoso	1 / 3
Temperatura	quente	0 / 2
	moderada	2 / 4
	fria	0 / 1
Umidade	alta	1 / 5
	normal	1 / 2
Ventando	falso	1 / 3
	verdadeiro	1 / 4



Conjunto de Regras : classe sim

Regra modificada

Se **Temperatura = moderada** Então resposta = sim


Exemplos cobertos pela regra modificada

outlook	temperatura	umidade	ventando	joga
chuvoso	moderada	alta	falso	sim
ensolarado	moderada	alta	falso	não
ensolarado	moderada	normal	verdadeiro	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe sim

Se Temperatura = moderada ? Então resposta = sim

Possíveis Testes

Atributo	Valor	
Outlook	ensolarado	1/2
	chuvoso	1/2
Umidade	alta	1/3
	normal	1/1 
Ventando	falso	1/2
	verdadeiro	1/2

Conjunto de Regras : classe sim

Regra Final

Se Temperatura = moderada e Umidade = normal
Então resposta = sim

Próxima regra

Se ? Então resposta = sim

- ❑ Excluir do conjunto total os exemplos que satisfazem a regra construída previamente.
- ❑ Repetir a etapa anterior até que não exista mais exemplos da classe não

Conjunto de Regras : classe sim

Conjunto Reduzido

outlook	temperatura	umidade	ventando	joga
ensolarado	quente	alta	falso	não
ensolarado	quente	alta	verdadeiro	não
nublado	quente	alta	falso	sim
chuvoso	moderada	alta	falso	sim
chuvoso	fria	normal	falso	sim
chuvoso	fria	normal	verdadeiro	não
nublado	fria	normal	verdadeiro	sim
ensolarado	moderada	alta	falso	não
ensolarado	fria	normal	falso	sim
chuvoso	moderada	normal	falso	sim
ensolarado	moderada	normal	verdadeiro	sim
nublado	moderada	alta	verdadeiro	sim
nublado	quente	normal	falso	sim
chuvoso	moderada	alta	verdadeiro	não

Conjunto de Regras : classe sim

Se Outlook = nublado Então resposta = sim

Se Umidade = normal e Ventando = falso Então
resposta = sim

Se Temperatura = moderada e Umidade = normal
Então resposta = sim

Se Temperatura = moderada Umidade = alta Ventando=falso
Outlook =chuvoso Então resposta = sim

Regras de Classificação 33/

✉ Método Prism: pseudo código

```
For each class C
  Initialize E to the instance set
  While E contains instances in class C
    Create a rule R with an empty left-hand side that predicts class C
    Until R is perfect (or there are no more attributes to use) do
      For each attribute A not mentioned in R, and each value v,
        Consider adding the condition A = v to the left-hand side of R
        Select A and v to maximize the accuracy p/t
        (break ties by choosing the condition with the largest p)
      Add A = v to R
    Remove the instances covered by R from E
```

Regras de Classificação 34/

✉ Método Prism

- ✉ O método Prism gera uma lista de descrição por classe
 - ✉ As regras subsequentes são construídas tal que não são cobertas por regras prévias
 - ✉ A ordem das regras não importa porque todas dizem respeito a mesma classe
- ✉ Problemas: intersecção de regras; regra default requerida

Regras de Classificação 35/

✉ Método Prism

- ✉ Métodos do tipo Prism tem como estratégia “separar para conquistar”
 - ✉ Primeiro é identificada uma regra
 - ✉ Depois, todos os exemplos cobertos por essa regra são desconsiderados
 - ✉ Enfim, os exemplos restantes são “conquistados”

Regras de Classificação 36/

✉ Método Prism

- ✉ Diferença em relação aos métodos de estratégia “dividir para conquistar”
 - ✉ o subconjunto coberto pela regra não necessita ser mais levado em consideração posteriormente

Regras de Classificação 37/

✉ Poda de regras

✉ Duas estratégias principais

- ✉ poda incremental

- ✉ poda global

✉ Outra diferença: critério de poda

- ✉ Erro no conjunto hold out (reduced error pruning)

- ✉ significância estatística

- ✉ principio MLD (minimum description lenght)

- ✉ também: pós-poda versus pré-poda

Regras de Classificação 38/

✉ INDUCT: realiza poda incremental

Initialize E to the instance set

Until E is empty do

For each class C for which E contains an instance

Use basic covering algorithm to create best perfect rule for C

Calculate significance $m(R)$ for rule and significance $m(R-)$ for rule with final condition omitted

If $(m(R-) < m(R))$, prune rule and repeat previous step

From the rules for the different classes, select the most significant one (i.e. the one with smallest $m(R)$)

Print the rule

Remove the instances covered by rule from E

Continue

Regras de Classificação 39/

✉ INDUCT: cálculo da significância

✉ medida de significância no INDUCT

✉ probabilidade de que uma regra aleatória com a mesma cobertura tenha desempenho pelo menos tão bom quanto a regra em questão

✉ A regra aleatória R seleciona aleatoriamente t casos

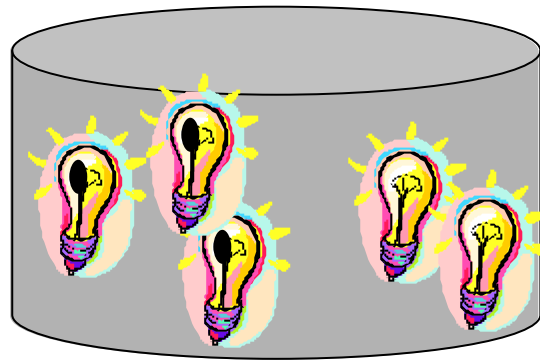
✉ qual a chance de que p desses pertençam a classe em questão?

✉ Essa probabilidade é fornecida pela regra hipergeométrica

Distribuição Binomial

Contexto Estatístico

Ensaio de Bernoulli



Uma lâmpada é escolhida ao acaso

A- a lâmpada é defeituosa

$A = 0$ se a lâmpada não é defeituosa

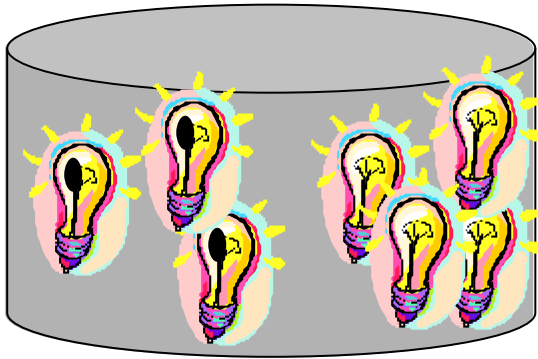
$A = 1$ se a lâmpada é defeituosa

$$P(A=0) = 2/5$$

$$P(A=1) = 3/5$$

Distribuição Binomial

Contexto Estatístico



3 Ensaio de Bernoulli, $t = 3$

$P(\text{defeituosa}) = 3/7$

$P(\text{não defeituosa}) = 4/7$

Seja X o número de defeituosas

$S = \{111, 110, 101, 011, 001, 010, 100, 000\}$

$X = 0 - \{000\}$

$X = 1 - \{001, 010, 100\}$

$X = 2 - \{110, 101, 011\}$

$X = 2 - \{110, 101, 011\}$

$X = 3 - \{111\}$

$P(X=1)=?$

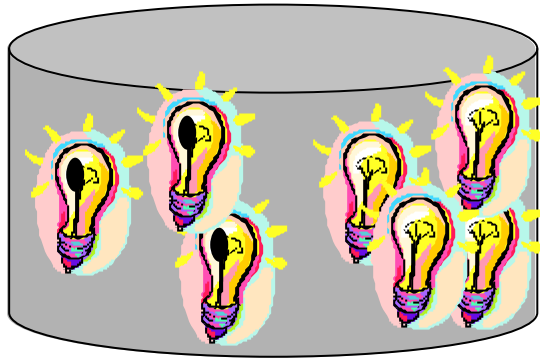
$$P(001) = 4/7 \times 4/7 \times 3/7$$

$$P(X = 1) = \binom{3}{1} \times \frac{3}{7} \times \left(\frac{4}{7}\right)^2$$

$$P(X = i) = \binom{t}{i} \times p^i \times (1-p)^{t-i}$$

Distribuição Hipergeométrica

Contexto Estatístico



Experimento: selecionar três lâmpadas ao acaso sem reposição.

N- número total de lâmpadas

r- número total de lâmpadas defeituosas

Seja X o número de defeituosas.

$S = \{111, 110, 101, 011, 001, 010, 100, 000\}$

X = 0 - {000}

X = 1 - {001, 010, 100}

X = 2 - {110, 101, 011}

X = 2 - {110, 101, 011}

X = 3 - {111}

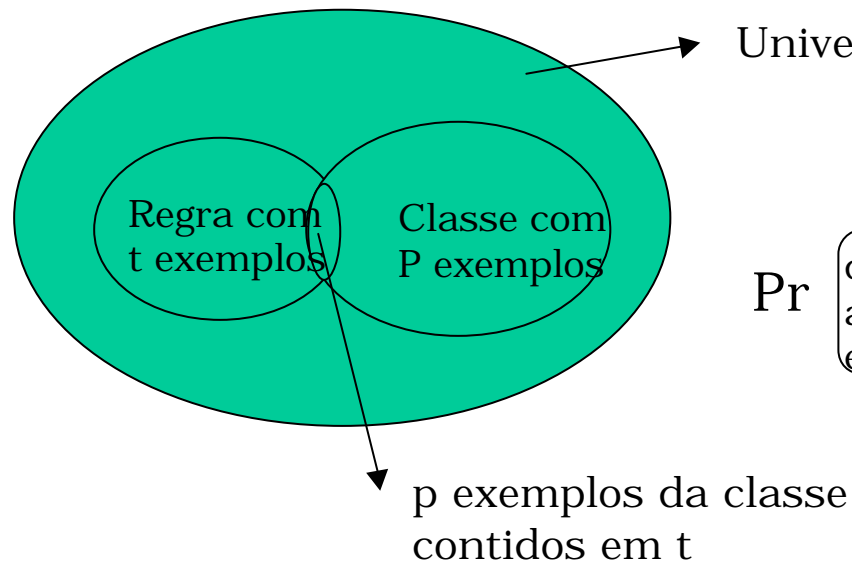
X= k se e somente são selecionadas k peças dentre 3 e 3-k peças não defeituosas dentre 4

$$P(X = 1) = \frac{\binom{3}{1} \binom{4}{2}}{\binom{7}{3}}$$

$$P(X = i) = \frac{\binom{P}{i} \binom{T-P}{t-i}}{\binom{T}{t}}$$

Critério Probabilístico para avaliar uma regra

Qual a probabilidade de uma regra selecionada aleatoriamente seja tão boa ou melhor que regra que esta sendo avaliada ?



$\Pr \left[\begin{array}{l} \text{de } t \text{ casos selecionado} \\ \text{aleatoriamente } i \\ \text{estão na classe } C \end{array} \right]$

$$= \frac{\binom{P}{i} \binom{T-P}{t-i}}{\binom{T}{t}}$$

$$i = 0, 1, \dots, \min(t, P)$$

Critério Probabilístico para avaliar uma regra

Seja X o número de exemplos que estão na classe C . X é uma variável aleatória com distribuição Hipergeométrica e $x = 0, 1, \dots, \min(t, P)$.

A probabilidade de uma regra R é dada por

$$m(R) = \sum_{i=p}^{\min(t, P)} \Pr(\text{de } t \text{ exemplo } i \text{ estão em } C)$$

$m(R)$ é a probabilidade de t conter pelo menos p exemplos da classe C

Aproximação usando distribuição Binomial

Se n é grande ($n > 30$)

\Pr é aproximado pela probabilidade usando uma Distribuição binomial para X = número de exemplos da classe C que estão t .

$$\Pr \begin{array}{l} \text{de } t \text{ casos selecionado} \\ \text{aleatoriamente } i \\ \text{estão na classe } C \end{array} = \binom{t}{i} \left(\frac{P}{T} \right)^i \left(1 - \frac{P}{T} \right)^{t-i}$$

$$i = 0, 1, \dots, \min(T, P)$$

Critério Probabilístico para avaliar uma regra

Regra : Se Outlook = ensolarado Então resposta = não

$$P/T = 5/14 \longrightarrow p/t = 3/5$$

$$m(R) = \sum_{i=3}^5 \Pr(\text{de 5 exemplos, } i \text{ estão em não})$$

$$m(R) = \frac{\binom{5}{3} \binom{9}{2}}{\binom{14}{5}} + \frac{\binom{5}{4} \binom{9}{1}}{\binom{14}{5}} + \frac{\binom{5}{5} \binom{9}{0}}{\binom{14}{5}} = 0.20$$

A regra é considerada de baixa qualidade.

Critério Probabilístico para avaliar uma regra

Regra: Se Outlook = ensolarado e Temperatura = quente
Então resposta = não

$$P/T = 5/14 \quad \longrightarrow \quad p/t = 2/2$$

$$m(R) = \sum_{i=2}^2 \Pr(\text{de 2 exemplos, } i \text{ estão em não})$$

$$m(R) = \frac{\binom{5}{2} \binom{9}{0}}{\binom{14}{5}} = 0.04$$

A regra é considerada de boa qualidade.

Critério Probabilístico para avaliar uma regra

Regra: Se Outlook = chuvoso Então resposta = sim

$$P/T = 9/14 \longrightarrow p/t = 3/5$$

$$m(R) = \sum_{i=3}^5 Pr(\text{de 5 exemplos, } i \text{ estão em sim})$$

$$m(R) = 0.80$$

A regra é considerada ruim.

Gerando regras usando critério de probabilidade para evitar *overfitting*

Para uma classe C encontre a regra perfeita

R: Se astigmatismo = sim, taxa = normal e idade = jovem então classe A

$$p/t = 2/2 \quad Pr = 2.2\%$$

Elimine o último termo de R

R: Se astigmatismo = sim, taxa = normal então classe A

$$p/t = 4/6 \quad Pr = 0.14\%$$

Elimine o último termo de R

R: Se astigmatismo = sim, então classe A

$$p/t = 4/12 \quad Pr = 4.7\%$$

Regra Final **R: Se astigmatismo = sim, taxa = normal então classe A**

Gerando regras usando critério de probabilidade para evitar *overfitting*

O método combina medida de precisão (para crescimento das regras) e o critério de probabilidade (para podar as regras).

A regra final pode não ser a melhor regra:

- ❑ O algoritmo de cobertura não necessariamente gera a melhor regra candidata para o *pruning*
- ❑ O *pruning* inicia pelo último teste adicionado; isto não significa que é o melhor teste para o prune
- ❑ O *pruning* para quando a probabilidade cresce; continuando com este último ponto pode ser que seja encontrado uma melhor regra para podar

Gerando regras usando conjunto de teste para evitar *overfitting*

- ❑ Estratégia: Usar um conjunto para avaliar as regras: *pruning*.

Conjunto que gera as regras deve ter $2/3$ do conjunto de exemplos E

Conjunto para *pruning* tem $1/3$

- ❑ Problema de representação do conjunto de *pruning*

Solução: Bootstrapping

- ❑ A idéia é chamada de *reduced-error pruning*

Gerando regras usando conjunto de teste para evitar *overfitting*

Critério de avaliação de uma regra R

$$W(R) = (p + (N - n)) / T$$

onde

$N = T - P$ número de exemplos negativos

$n = t - p$ número de exemplos negativos cobertos por l

A medida avalia os exemplos negativos não cobertos e os exemplos positivos cobertos pela regra

Gerando regras usando conjunto de teste para evitar *overfitting*

Exemplo

Uma regra R_1 tem $p = 2000$ positivos e $n = 1000$ negativos $t = 3000$

$$p/t = 2000/3000 \quad W(R_1) = (1000+N)/T$$

Uma regra R_2 tem $p = 1000$ positivos e $n=1$ negativo $t = 1001$

$$p/t = 1000/1001 \quad W(R_2) = (900+N)/T$$

Gerando regras usando conjunto de teste para evitar *overfitting*

Exemplo

Uma regra R_1 tem $p = 1$ positivo e $n = 0$ negativo então $t = 1$

$$p/t = 1/1 \quad W(R_1) = (1+N)/T$$

Uma regra R_2 tem $p = 1000$ positivos e $n=1$ negativo então $t = 1001$

$$p/t = 1000/1001 \quad W(R_2) = (900+N)/T$$

Incremental reduced-error pruning

```
Initialize E to the instance set
Until E into Grow and Prune in the ratio 2:1
  For each class C for which Grow and Prune both contain an instance
    Use basic covering algorithm to create best perfect rule for C
    Calculate worth  $w(R)$  for rule on Prune and worth  $w(R_-)$  for
      rule with final condition omitted
    If  $(w(R_-) < w(R))$ , prune rule and repeat previous step
  From the rules for the different classes, select the one that's
    worth most (i.e. the one with the largest  $w(R)$ )
  Print the rule
  Remove the instances covered by rule from E
Continue
```


Gerando regras usando conjunto de teste para evitar *overfitting*

- ❑ O método é rápido e em geral apresenta um bom desempenho para grandes bases de dados
- ❑ Acelera o processo se gera regras quando as classes são ordenadas
- ❑ Critério de parada evita gastar tempo gerando um conjunto de regras com pouca cobertura