Pareto Front Elite

Celso Yoshikazu Ishida[1], **Aurora T. R. Pozo**[2]

[1]PPGMNE – Universidade Federal do Paraná (UFPR)
Caixa Postal 19011 – 81.531-990 – Curitiba – PR – Brazil

[2]Departamento de Informática – UFPR – Curitiba – PR – Brazil

celsoishida@gmail.com, aurora@inf.ufpr.br

Abstract. *ROC analysis has been used for a fair comparison of machine learning algorithms since the end of the last decade. The area under the ROC curve (AUC) is considered a relevant criterion to deal with imbalanced data, misclassification costs and noisy data. The greater value of AUC indicates the classifier with the best average performance. Based on this preference, we introduce an algorithm for rule learning. This algorithm builds a Pareto Front using the Sensitivity and Specificity criteria selecting rules from a larges set of rules. We compare our results from others rule induction algorithms and the results shows that the new algorithm obtains a set of rules with great values of AUC.*

Resumo. *Desde a última década, a análise ROC tem sido utilizada para comparações de algoritmos de aprendizado de máquina. A área abaixo da curva ROC (AUC) é considerada um critério relevante para lidar com dados não balanceados, custos de erros de classificação e ruídos. Um classificador com o maior valor de AUC indica que possui o melhor desempenho médio. Baseado nestas preferências, introduzimos um algoritmo de aprendizado de regras. A partir de um grande conjunto de regras, o algoritmo constrói uma Fronteira de Pareto utilizando os critérios de Sensitividade e Especificidade. Comparamos os resultados com outros algoritmos de indução de regras evidenciando que o novo algoritmo obtém um conjunto de regras com altos valores de AUC.*

1. Introdução

Trabalhos recentes têm discutido sobre medidas de desempenho de um classificador. Em geral um classificador comete dois tipos de erros: Falsos Positivos (FP) e os Falsos Negativos (FN). Por exemplo, num problema de classificação de pessoas que estão ou não com uma determinada doença, os FP seriam as pessoas sãs que foram diagnosticadas, ou classificadas, como doentes. Já os FN são as pessoas doentes que foram classificadas como sadias.

Complementares ao FP e FN, respectivamente, citam-se os Verdadeiros Positivos (VP) e Verdadeiros Negativos (VN). As duas medidas indicam a classificação correta. O VP, por exemplo, é o número de pacientes que possuem a doença e foram diagnosticadas corretamente. Já VN é o número de pacientes sãos que voltaram para casa. Das medidas básicas, várias medidas podem ser derivadas, tais como a Taxa de VP, Taxa de FP e especificidade. A Taxa de VP, também chamada de *recall* ou sensitividade, é a divisão entre VP e o total de exemplos positivos. A Taxa de FP, também chamada de Taxa de Alarme Falso, é a porcentagem de FP em relação aos exemplos negativos ($\frac{FP}{FP+VN}$). O

seu complemento, a especificidade, indica a taxa de precisão entre os exemplos negativos ($especificidade = 1 - taxa\ FP = 1 - \frac{FP}{FP+VN} = \frac{(FP+VN)-FP}{FP+VN} = \frac{VN}{FP+VN}$).

Uma medida de desempenho frequentemente utilizada é a taxa de erro de classificação que mede a porcentagem de exemplos que foram classificados em uma classe diferente de sua original, o seu complemento, a precisão ($erro = 1 - precisão$). Embora a precisão (accuracy) seja uma medida aceita, a sua maximização não é um objetivo apropriado para muitas tarefas do mundo real [Provost et al. 1998]. A precisão assume que existe um mesmo custo para os erros de classificação dos FP e FN. Diferentemente do que ocorre no mundo real, onde um erro tem um custo maior do que o outro [Fawcett and Provost 1997]. Ou seja, o erro FN de orientar um paciente doente para ir para sua casa, dizendo que ele não precisa de tratamento, é um problema maior do que o erro FP.

Devido a estes comportamentos, muitos têm preferido a análise "Receiver Operating Characteristic" (ROC). A curva ROC tem sido utilizada em teoria de detecção de sinais para representar a relação entre Taxa de VP e Taxa de FP [Egan 1975]. Mas foi introduzida na área de aprendizado de máquina no final dos anos 90 [Provost and Fawcett 1997].

O gráfico ROC possui duas dimensões, a Taxa de VP (ou $sensitividade$) no eixo Y e a Taxa de FP (ou $1 - especificidade$) no eixo X. O gráfico mostra a relação das diferenças entre os benefícios (indicados pela Taxa de VP) e o custo (Taxa de FP). No gráfico ROC 1, adaptado de [Fawcett 2003], estão cinco classificadores de uma mesma classe de exemplos, nomeados de A até E. Cada classificador produz um par (Taxa de FP, Taxa de VP), que corresponde a um ponto no espaço ROC. Note que os pares (Taxa de FP, Taxa de VP) e (1-Especificidade, Sensitividade) representam o mesmo ponto.

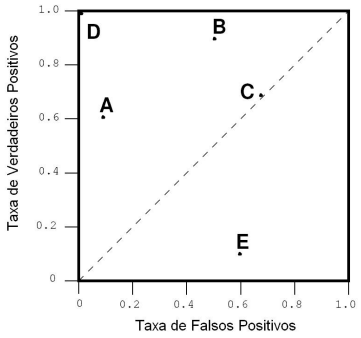
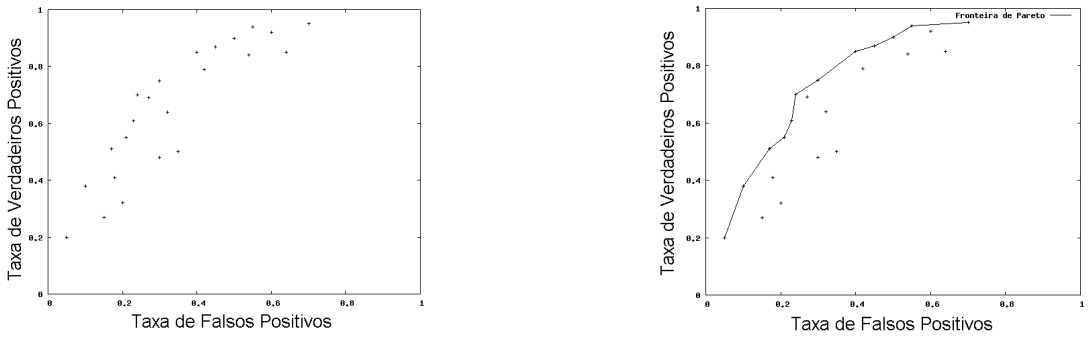


Figure 1. Gráfico ROC com cinco classificadores, adaptado de [Fawcett 2003]

É importante observar alguns pontos no espaço ROC. O ponto (0, 0) representa um classificador que não classifica nenhum exemplo como positivo, o classificador ligado a este ponto não comete nenhum erro (Taxa de FP=0), mas também nenhum acerto (Taxa de VP=0). O classificador do ponto extremo (1,1) associa todos os exemplos como sendo positivos. Já o ponto (0, 1) representa a classificação perfeita, o ponto D da Figura 1 é um classificador perfeito com Taxa de FP igual a zero (especificidade=1) e sensitividade igual a 1.

No espaço ROC, quanto mais a noroeste melhor é o ponto, ou seja, se um classificador possuir o maior valor de Taxa de VP e o menor valor de Taxa de FP será o melhor

classificador. Ou seja, o melhor classificador tem valor de sensitividade e especificidade próximo de 1. Na figura 1, o ponto B indica que o seu classificador associado é melhor do que o do ponto C, neste caso podemos afirmar que o classificador B domina o classificador C. O mesmo não podemos afirmar para o ponto A em relação ao ponto C, pois o ponto A, apesar de possuir menor valor de Taxa de FP, não possui maior valor de Taxa de VP em relação ao ponto C. Neste caso, o ponto A não é dominado pelo ponto C. O conceito de pontos não dominados nos leva a separar as melhores regras de um determinado conjunto. Dizemos que as regras não dominadas formam a Fronteira de Pareto e nenhum outro elemento do conjunto possui valores de medida melhores do que as regras não dominadas. Na Figura 2(a) é possível visualizar um conjunto de regras e a Figura 2(b) possui o mesmo conjunto com uma linha ligando os pontos da Fronteira.



(a) (b)

Figure 2. (a) Conjunto de Regras no Espaço ROC. (b) Regras e sua Fronteira de Pareto.

A curva ROC permite uma boa visualização do desempenho de um classificador [Bradley 1997]. Entretando, frequentemente, deseja-se uma medida única para a desempenho do classificador. Para tal foi proposto que a área abaixo da curva ROC (AUC, *Area Under the Curve*) fosse utilizada para a comparação de algoritmos de aprendizado [Provost and Domingos 2003] e para a construção e otimização de modelos de aprendizado [Ferri et al. 2002] [Rakotomamonjy 2004] [Sebag et al. 2003a].

A curva ROC pode ser construída para um único classificador baseado na variação de um limiar (*threshold*). O classificador atribui um valor (*score*) a cada exemplo e ordena os exemplos por este valor. Através de um limiar, os exemplos com valor acima do limiar são classificados como positivos, os demais são classificados como negativos. Variando este limiar de $-\infty$ a $+\infty$, obtêm-se vários pontos no espaço ROC, um para cada limiar. A área abaixo desta curva indica o desempenho do classificador [Fawcett 2003].

Um conjunto de regras não-ordenadas formam um classificador e pode ser avaliado pela curva ROC. Uma estratégia para a definição do *score* é a Votação por Pesos (*Weighted Voting* [Fawcett 2001]). As regras votam em todos os exemplos selecionados pela regras utilizando a confidência. Numa classificação binária, o *score* associado a cada exemplo é a soma da confidência das regras positivas e a diminuição da confidência das regras negativas. Após este cálculo, os exemplos são ordenados pelo *score* e com a variação do limiar de $-\infty$ a $+\infty$ a curva ROC é construída. Note que, as regras que classificam os exemplos corretamente são preferidas para uma boa ordenação. E, quanto melhor a ordenação melhor será a curva ROC construída. Portanto, a presença de regras

no conjunto resultado com bons valores de sensitividade e especificidade é preferida para a maximização da AUC.

Este trabalho apresenta um novo algoritmo para seleção de um conjunto de regras de classificação e tem como objetivo maximizar AUC. Para tal, dado um grande conjunto de regras seleciona-se um conjunto limitado de regras. A seleção para a formação da Fronteira de Pareto é feita de acordo com os critérios de Sensitividade e Especificidade.

O artigo esta organizado da seguinte forma. Na próxima seção serão denominados os trabalhos relacionados: ROGER, AprioriC e ROCCER. Na seção 3 apresentamos o algoritmo Pareto Front Elite, seguido pelos experimentos realizados para sua comparação com outros algoritmos (seção 4). E finalmente a seção 5 conclui o trabalho.

2. Trabalhos Relacionados

Vários trabalhos têm sido propostos para encontrar um conjunto de regras utilizadas para a tarefa de classificação. Mas apenas nos últimos anos têm se dado grande importância à critérios diferentes ou complementares à precisão.

Baseado na Estratégia Evolucionária [Bäck and Schwefel 1993], um algoritmo chamado ROGER (ROC-based Genetic Learning, algoritmo de aprendizado genético baseado na curva ROC) foi criado para optimização do critério AUC [Sebag et al. 2003a]. O trabalho é parte de uma solução de um problema prático da área médica, e o seu sistema apresentou resultados significantes para o diagnóstico da Arteriosclerose [Sebag et al. 2003b]. O nosso trabalho tenta atingir o mesmo objetivo de maximizar a área AUC, mas a diferença esta no modelo de classificação. Enquanto o modelo do ROGER é uma *perceptron*, o nosso é um conjunto de regras.

Também para a otimização da AUC, alguns trabalhos para indução de regras de classificação têm sido apresentados utilizando-se de algoritmos de regras de associação para geração de regras citados em [Prati and Flach 2005]. E, devido ao grande número de regras geradas, alguns algoritmos como AprioriC propuseram a remoção de algumas regras redundantes [Jovanoski and Lavrac 2001]. Contudo, o AprioriC ainda tende a gerar um grande conjunto de regras. A nossa idéia é não produzir um grande número de regras sem se importar com as regras redundantes.

ROCCER [Prati and Flach 2005], um trabalho recente, contrói um convex hull no espaço ROC. Baseado no algoritmo Apriori, o ROCCER obtém um conjunto de regras com valores AUC compatíveis com os melhores preditores probabilísiticos tais como árvore de decisão de poda, e, também, encontra poucas regras como resultado. As regras encontradas pelo algoritmo Apriori são incorporadas ao ROCCER de acordo com sua contribuição à AUC e formam um conjunto de regras ordenadas. Porém, a inclusão de uma regra leva à uma reavaliação da contribuição das regras, o que causa um grande problema: a complexidade $O(n^2)$ (onde n é o número de regras geradas pelo Apriori), no pior caso. O nosso objetivo é criar um algoritmo que encontra regras não-ordenadas através de um método mais simples. Apesar do ROCCER obter bons resultados, também é nosso objetivo melhorar os seus resultados.

Alguns autores propõem a utilização do nível de confiança para seleção do conjunto de regras [Gamberger and Lavrac 2000]. Nesta estratégia, os exemplos não cobertos aparecem como não classificados. [Ferri et al. 2004] extende a idéia criando novos clas-

sificadores para estes exemplos. O nosso trabalho propõe a utilização das medidas de Sensitividade e Especificidade para seleção de uma Fronteira de Pareto. E os exemplos não cobertos são classificados para a classe majoritária.

3. Pareto Front Elite

O algoritmo Pareto Front Elite (Fronteira de Pareto a partir das regras Elites) tem como objetivo encontrar um conjunto de regras não-ordenadas para maximizar AUC. O algoritmo seleciona as melhores regras geradas a partir de um algoritmo de associação de regras (tipo Apriori). As melhores regras, ou regras Elite, passam por uma seleção para a formação da Fronteira de Pareto.

Os passos gerais, detalhados no Algoritmo 1, são: execução do algoritmo de aprendizado de regras para geração das regras Elite ($RegrasElite$) e seleção da Fronteira de Pareto ($paretoFront$). Estes passos são executados duas vezes, uma para cada *head*, ou seja, uma para selecionar um conjunto de regras para exemplos positivos e outro conjunto para exemplos negativos.

Um dos grandes problemas do Apriori esta associado é a geração de um grande número de regras. Geralmente contornado nos passos: estabelecimento, pelo usuário, dos parâmetros mínimos de Suporte e Confidência; e a execução do algoritmo gerando apenas as regras com Suporte e Confidência maiores do que os mínimos. Quanto maiores os mínimos escolhidos para a execução do Apriori, menor será o número de regras geradas.

Propomos a execução do Apriori para geração de regras com Suporte ou Confidência maior do que os mínimos estabelecidos. As regras geradas pelo Apriori são chamadas de regras Elites ($RegrasElite$). Para que o parâmetro seja independente do usuário, o algoritmo estabelece o Suporte Mínimo como: $SuporteMedio + SupDesvioPadrao$, onde o $SuporteMedio$ é a média do suporte de todas as regras com um único atributo e e $SupDesvioPadrao$ é o desvio padrão relacionado. A Confidência Mínima é definida como $ConfidenciaMedia + ConfDesvioPadrao$, onde o $ConfidenciaMedia$ é a média da confidência de todas as regras com um único atributo e $ConfDesvioPadrao$ é o desvio padrão relacionado. Se os suportes e confidências estivessem numa distribuição normal, seriam criados apenas as 15.8% melhores regras. O fato dos mínimos serem calculados automaticamente, faz do Pareto Front Elite um algoritmo sem parâmetros estabelecidos pelo usuário.

A partir das regras Elite geradas, cria-se a Fronteira de Pareto ($paretoFront$) com as regras não-dominadas. Ou seja, uma regra fará parte do conjunto $paretoFront$ se, e somente se, não existir nenhuma outra regra com valores maiores de Sensitividade e Especificidade. Como a fronteira de Pareto pode conter regras com valores baixos de sensitividade e especificidade, portanto, é proposto um valor de corte para a remoção destas regras.

Para o cálculo da AUC foi utilizada a votação por pesos para atribuir os *scores* aos exemplos. Para cada exemplo da base de teste, todas as regras que o cobre votam utilizando a confidência. O *score* associado a cada exemplo é a soma da confidência das regras positivas e a diminuição da confidência das demais regras. Após este cálculo, os exemplos são ordenados pelo *score* e com a variação do limiar de $-\infty$ a $+\infty$ a curva ROC é construída.

Algorithm 1 Pareto Front Elite

Entrada:
$head$ - Cabeçalhos das regras (positivo / negativo)

Saída:
$paretoFront$ - Fronteira de Pareto

Passos Gerais:

$RegrasElite$ = Apriori($head$, $SuporteMinimo$, $ConfidenciaMinima$)

$paretoFront = \{rule_j \in RegrasElite \ / \ \neg\exists rule_l,$
$Sensitividade(rule_l) > Sensitividade(rule_j)$ and $Especificidade(rule_l) > Especificidade(rule_j)\}$

Table 1. Bases de Experimentos

#	DB	# Atrb	# Exemplos	#	DB	# Atrb	# Exemplos
1	breast	10	683	9	ionosphere	34	351
2	bupa	7	345	10	kr-vs-kp	37	3196
3	ecoli	8	336	11	lettera	17	20000
4	flag	29	174	12	new-thyroid	6	215
5	german	21	1000	13	nursery	9	12960
6	glass	10	214	14	pima	9	768
7	haberman	4	306	15	satimage	37	6435
8	heart	14	270	16	vehicle	19	846

Quanto à complexidade do nosso algoritmo, o maior problema esta na complexidade da geração das regras feita pelo Apriori. A seleção dos elementos não dominantes é a parte menos onerosa do algoritmo, cuja complexidade é de $O(n!)$, onde n é o número de regras do conjunto Elite.

4. Experimentos

Nosso objetivo de avaliação é melhorar o desempenho apresentado pelo ROCCER e outros algoritmos descritos no artigo [Prati and Flach 2005]. Portanto, os resultados foram obtidos do trabalho citado e os experimentos com o algoritmo Pareto Front Elite foram feitos com a mesma metodologia e nas mesmas bases de dados (Tabela 1, originalmente obtidas do *UCL Machine Learning Repository* [D.J. Newman and Merz 1998]). Os experimentos pareados foram executados utilizando a validação cruzada (*10-fold stratified cross-validation*). Pareados no sentido de todas as execuções serem feitas com os mesmos arquivos de treinamento e de teste. Todas as comparações entre os algoritmos foram feitas utilizando o teste T com nível de confiança de 95%. O algoritmo Pareto Front Elite foi executado como o valor de corte igual a $0,08$ para Sensitividade e Especificidade na seleção final das regras. Este foi o melhor valor encontrado empiricamente.

As comparações foram feitas com os resultados das execuções dos algoritmos

Table 2. AUC médio

#	PF Elite	ROCCER	C45	C45NP	CN2	CN2OR	Ripper	Slipper	Todas
1	99,38(1,00)	98,63 (1,88)	97,76 (1,51)	98,39 (1,3)	99,26 (0,81)	99,13 (0,92)	98,72 (1,38)	99,24 (0,57)	99,07 (0,87)
2	66,76(11,79)	65,3 (7,93)	62,14 (9,91)	57,44 (11,92)	62,74 (8,85)	62,21 (8,11)	69,1 (7,78)	59,84 (6,44)	65,38 (10,63)
3	91,57(5,11)	90,31 (11,56)	50 (0)	90,06 (7,75)	90,17 (6,9)	85,15 (11,38)	61,86 (25,49)	74,78 (15,94)	16,5 (10,43)
5	73,42(6,40)	72,08 (6,02)	71,43 (5,89)	67,71 (4,12)	75,25 (5,38)	70,9 (4,7)	64,02 (13,62)	71,32 (6,2)	73,37 (4,84)
6	77,60(13,46)	79,45 (12,98)	50 (0)	81,5 (12,65)	73,74 (15,4)	79,64 (13,24)	49,75 (0,79)	50 (2,36)	35,62 (18,93)
7	67,43(5,86)	66,41 (11,54)	55,84 (6,14)	64,33 (13,58)	59,83 (9,87)	59,28 (10,13)	57,45 (3,85)	50,4 (11,14)	66,52 (5,94)
8	88,00(7,13)	85,78 (8,43)	84,81 (6,57)	81,11 (7,91)	83,61 (6,89)	82,25 (6,59)	84,89 (7,68)	84,03 (6,36)	90,72 (6,28)
12	97,40(4,00)	98,4 (1,7)	87,85 (10,43)	97,5 (3,39)	99,14 (1,19)	98,43 (2,58)	94,95 (9,94)	99,12 (1,25)	89,97 (7,75)
13	97,24(0,86)	97,85 (0,44)	99,42 (0,14)	99,74 (0,13)	100 (0)	99,99 (0,01)	99,43 (0,26)	94,4 (1,59)	97,79 (0,65)
14	70,52(4,28)	70,68 (5,09)	72,07 (4,42)	72,6 (6,5)	70,96 (4,62)	71,97 (5,44)	68,07 (9,46)	70,02 (5,97)	70,37 (5,01)
16	93,73(2,79)	96,42 (1,47)	94,76 (3)	96,99 (1,44)	97,38 (2,05)	96,49 (2,41)	95,01 (2,22)	93,99 (3,13)	93,37 (3,05)
Média	83,91	83,76	75,10	82,49	82,92	82,31	76,66	77,01	72,61

ROCCER, C4.5, CN2, Ripper e Slipper. O algoritmo ROCCER foi executado com 50% de Confiança e Suporte Mínimo igual a $1/3$ da classe minoritária. O C4.5 [Quinlan 1993] utiliza a ganho de informação como uma medida para construir árvores de decisão e um passo de poda baseado na redução de erro. Foram utilizadas as versões com poda (C45) e sem poda (C45NP). Na primeira versão do CN2 [Clark and Niblett 1989], uma lista de decisão é induzida usando a entropia como a heurística de busca. O algoritmo CN2 foi executado na versão ordenada (CN2OR) e não ordenada (CN2). Ripper é um algoritmo de aprendizado de lista de decisão [Cohen 1995]. Tem características de poda baseado no erro e uma heurística baseada em MDL para determinar quantas regras devem ser aprendidas. O Slipper [Cohen and Singer 1999] melhora o algoritmo Ripper uma vez que utiliza a técnica de *weighted set-covering*.

Os valores médios de AUC, que foram calculados pela regra do trapézio, e seus desvios padrões estão na Tabela 2. Os resultados do algoritmo Pareto Front Elite estão na primeira coluna. As células que estão em cinza claro indicam valores que são estatisticamente piores do que o algoritmo Pareto Front Elite. Em cinza escuro são os valores estatisticamente melhores do que o nosso algoritmo. Por uma limitação de tempo computacional e memória, os resultados das bases de dados 4, 9, 10, 11 e 15 não foram apresentados.

Em termos de diferenças estatísticas, de 11 bases de dados, o algorithmo Pareto Front Elite é pior do que o ROCCER em 2 bases de dados. Nas outras 9 bases de dados, não existe diferenças entre os algoritmos, mesmo que o nosso algoritmo tenha os valores maiores em 6. Verificando a média das 11 execuções apresentadas, nota-se uma certa semelhança do nosso algoritmo com o ROCCER. Comparando com outros algoritmos, a maior vantagem é contra o C45, o Pareto Front Elite ganha em 5 base de dados. Contra os algoritmos Ripper e Slipper, ganha-se 4 vezes dos dois algoritmos. Contra o C45NP, o nosso algoritmo ganha em 3 bases de dados. Pode-se notar que em duas bases de dados, o nosso algoritmo não obteve bons resultados, perdendo para quase todos os algoritmos.

Acreditamos que os bons resultados com relação à AUC se devem à boa qualidade das regras da Fronteira de Pareto. Avaliando as regras da Fronteira de Pareto, avalia-se que o suporte médio é de $16,09$ e o desvio padrão de $4,17$. Se compararmos com o algoritmo com maior valor neste critério, o algoritmo ROCCER com $13,67$ de média e desvio de $13,89$, nota-se que os valores são semelhantes. Em termos de Precisão Relativa Ponderada (*Weighted Relative Accuracy* - WRAcc), que indica a importância da regra, o nosso algoritmo alcança um valor ($0,0300$ e desvio padrão de $0,0210$) muito semelhante ao ROCCER ($0,0355$ e desvio padrão de $0,018$), o que indica que as regras são diferentes mas igualmente importantes.

Table 3. Número Médio de Regras

#	PF Elite	ROCCER	C4.5	C4.5NP	CN2	CN2OR	Ripper	Slipper	Todas
1	284(11,08)	48,4(2,32)	37,8(12,62)	104,2(9,58)	32,8(2,74)	24,3(2,71)	16,9(1,1)	36,7(10,78)	502,1(8,96)
2	10,4(1,35)	3,9(0,99)	15(10,53)	143,3(13,12)	100,7(3,77)	83,6(5,87)	7,5(1,84)	29,3(8,67)	292,8(21,57)
3	24,9(3,14)	6,7(0,82)	0(0)	62(10,46)	35,3(2,83)	33,6(3,13)	4,9(4,33)	15,2(6,73)	27,5(6,05)
5	91,9(7,50)	23,7(6,75)	78,2(18,5)	388,6(19,07)	143,9(6,98)	106,8(4,37)	8,9(3,25)	37,4(12,42)	2886,1(577,3)
6	50,6(3,63)	2,4(0,52)	0(0)	32,1(5,99)	21,6(1,26)	21,9(3,18)	0,2(0,42)	2(3,43)	26,7(12,1)
7	7,2(0,92)	0,8(0,42)	5,6(5,72)	71,2(9,39)	75,9(3,9)	57(5,98)	3,5(0,97)	11,8(13,21)	21,2(1,4)
8	72,4(20,28)	68,2(4,42)	13,2(4,49)	97,4(15,04)	42,8(2,49)	36,1(1,79)	7,7(1,25)	27,2(8,73)	1875,6(91,9)
12	30(2,16)	8,5(0,53)	4(0)	35,4(4,2)	18,7(0,67)	15,3(0,82)	10,9(3,21)	20,7(5,48)	39,1(4,72)
13	38,3(1,89)	18,9(1,66)	114,6(3,1)	227,1(3,57)	112,4(2,27)	17,6(0,52)	44,1(7,32)	57,6(2,37)	100(4,29)
14	27,4(3,72)	4(0,82)	49,4(20,27)	347,4(10)	169,7(10,08)	168,4(7,9)	8,7(2,21)	30,5(10,82)	10,7(3,62)
16	145,9(7,27)	48,4(4,48)	89,5(12,12)	188,9(9,42)	49,8(3,29)	41,2(2,82)	23,7(4,06)	75,7(12,11)	431,4(53,85)
Méd.	71,12	35,54	73,43	272,92	73,49	61,89	18,01	38,33	4674,23

A Tabela 3 apresenta o número médio de regras do conjunto final para cada algoritmo. Na maioria das bases de dados, o algoritmo Pareto Front Elite gera conjuntos estatisticamente maiores do que os algoritmos ROCCER, RIPPER e Slipper, com excessão de algumas bases. O algoritmo apenas possui bom desempenho se comparado com o algoritmo C45NP. E se for analisado a média dos algoritmos, pode-se dizer que o Pareto Front Elite tem um desempenho semelhante ao C45.

5. Conclusão

Apresentamos um algoritmo de seleção de regras que maximiza a AUC. A partir de regras Elites, o Pareto Front Elite seleciona as regras que farão parte da Fronteira de Pareto. Como uma forma de automatizar o processo, o Suporte Mínimo e a Confidência Mínima são calculados com base nos valores médios de todas as regras possíveis com um único atributo. Após as regras induzidas, ou regras Elite, são selecionadas de acordo com os critérios de Sensitividade e Especificidade para formar a Fronteira de Pareto. A Fronteira de Pareto é formada pelas regras não-dominadas, ou seja, as melhores regras em termos de Sensitividade e Especificidade compõem o conjunto resultado. Os experimentos confirmam que os conjuntos obtidos possuem regras com bons valores de Suporte e WRAccuracy. Apesar de uma pequena melhora em algumas bases de dados, os AUC médios encontrados são muito semelhantes aos apresentados pelo melhor algoritmo relacionado: ROCCER. Um ponto negativo do algoritmo é a geração de muitas regras, mas a simplicidade do algoritmo Pareto Front Elite justifica a sua utilização para a maximização da AUC.

Agradecimentos

Os nossos agradecimentos especiais ao Ronaldo Prati pelas bases de dados utilizadas nos experimentos. E, também, os agradecimentos pelo apoio financeiro recebido pelo programa CAPES/COFECUB.

References

Bäck, T. and Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Clark, P. and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3:261–283.

Cohen, W. W. (1995). Fast effective rule induction. In *ICML*, pages 115–123.

Cohen, W. W. and Singer, Y. (1999). A simple, fast, and effective rule learner. In *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI-99); Proceedings of the 11th Conference on Innovative Applications of Artificial Intelligence*, pages 335–342, Menlo Park, Cal. AAAI/MIT Press.

D.J. Newman, S. Hettich, C. B. and Merz, C. (1998). UCI repository of machine learning databases.

Egan, J. (1975). *Signal detection theory and ROC analysis*. Academic Press New York.

Fawcett, T. (2001). Using rule sets to maximize roc performance. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 131–138, Washington, DC, USA. IEEE Computer Society.

Fawcett, T. (2003). ROC graphs: Notes and practical considerations for data mining researchers. Technical Report HPL-2003-4, Hewlett Packard Laboratories.

Fawcett, T. and Provost, F. J. (1997). Adaptive fraud detection. *Data Mining and Knowledge Discovery*, 1(3):291–316.

Ferri, C., Flach, P., and Hernandez-Orallo, J. (2002). Learning decision trees using the area under the ROC curve. pages 139–146.

Ferri, C., Flach, P., and Hernandez-Orallo, J. (2004). Delegating classifiers. In Greineer, R. and Schuurmans, D., editors, *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*. ACM.

Gamberger, D. and Lavrac, N. (2000). Confirmation rule sets. In Zighed, D. A., Komorowski, H. J., and Zytkow, J. M., editors, *PKDD*, volume 1910 of *Lecture Notes in Computer Science*, pages 34–43. Springer.

Jovanoski, V. and Lavrac, N. (2001). Classification rule learning with APRIORI-C. In Brazdil, P. and Jorge, A., editors, *EPIA*, volume 2258 of *Lecture Notes in Computer Science*, pages 44–51. Springer.

Prati, R. C. and Flach, P. A. (2005). ROCCER: An algorithm for rule learning based on ROC analysis. In Kaelbling, L. P. and Saffiotti, A., editors, *IJCAI*, pages 823–828. Professional Book Center.

Provost, F. and Domingos, P. (2003). Tree induction for probability based ranking. *Machine Learning*, 52(3):199–215.

Provost, F., Fawcett, T., and Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *Proceedings 15th International Conference on Machine Learning*, pages 445–453. Morgan Kaufmann, San Francisco, CA.

Provost, F. J. and Fawcett, T. (1997). Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *KDD*, pages 43–48.

Quinlan, J. (1993). *C4. 5: Programs for Machine Learning*. Morgan Kaufmann.

Rakotomamonjy, A. (2004). Optimizing area under roc curve with SVMs. In Hernández-Orallo, J., Ferri, C., Lachiche, N., and Flach, P. A., editors, *ROCAI*, pages 71–80.

Sebag, Aze, and Lucas (2003a). ROC-based evolutionary learning: Application to medical data mining. In *International Conference on Artificial Evolution, Evolution Artificielle, LNCS*, volume 6.

Sebag, M., Azé, J., and Lucas, N. (2003b). Impact studies and sensitivity analysis in medical data mining with ROC-based genetic learning. In *ICDM*, pages 637–640. IEEE Computer Society.