Instituto de Ciências Matemáticas e de Computação

ISSN - 0103-2569

Comparação Empírica dos Sistemas de Programação Lógica Indutiva Foil e Progol

Chandler Wellington Caulkins

Maria Carolina Monard/ILTC

Nº 121

RELATÓRIOS TÉCNICOS DO ICMC

São Carlos

Novembro/2000

Comparação Empírica dos Sistemas de Programação Lógica Indutiva Foil e Progol *

Chandler Wellington Caulkins
Maria Carolina Monard/ILTC

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departmento de Ciências de Computação e Estatística
Laboratório de Inteligência Computacional
Cx. Postal 668, 13560-970 - São Carlos, SP
e-mail: {chandler, mcmonard}@icmc.sc.usp.br

Resumo Em princípio, qualquer conjunto de exemplos representados através do formato atributo-valor, utilizados por algoritmos de Aprendizado de Máquina — AM — proposicional pode ser utilizado por algoritmos de AM relacional, ou Programação Lógica Indutiva — PLI. Para tanto, os dados devem ser representados na forma de predicados. Neste trabalho, é avaliado empiricamente a utilidade de duas possíveis representações usando-as como entrada para dois sistemas de PLI bastante conhecidos, Foil e Progol. Experimentos são realizados utilizando alguns conjuntos de dados de problemas do mundo real, tirados do Repositório de Aprendizado de Máquina da UCI, e algumas comparações são feitas através dos resultados obtidos.

Palavras Chaves: Programação Lógica Indutiva, Avaliação Empírica, Foil, Progol

Este documento encontra-se no site da Biblioteca do ICMC

ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_121.ps.zip

Novembro 2000

*Trabalho realizado com auxílio parcial de CAPES e FINEP.

Sumário

1 Introdução **1**

2 Sistemas de PLI Utilizados **3**

2.1 Foil 3

2.2 Progol 3

3 Conjuntos de Dados **4**

3.1 Descrição Geral dos Conjuntos de Dados 4

3.2 Características dos Conjuntos de Dados 5

4 Descrição dos Experimentos **5**

5 Medidas Consideradas **7**

6 Resultados Experimentais **10**

6.1 Conjunto de Dados Bupa 11

6.2 Conjunto de Dados Pima 17

6.3 Conjunto de Dados Breast 26

6.4 Conjunto de Dados Hungaria 32

6.5 Conjunto de Dados Credit 37

6.6 Conjunto de Dados Hepatitis 42

7 Comparação dos Resultados **46**

8 Considerações Finais **47**

Lista de Figuras

3.2.1 Dimensionalidade dos Conjuntos de Dados 6

5.1 Verdadeiros Positivos, Falsos Positivos e Falsos Negativos 8

6.1.1 Regras Geradas por FOIL: Conjunto de Dados Bupa e Método Prop, com C1 = um 13

6.1.2 Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método Prop, com C1 = um 14

6.1.3 Regras Geradas por FOIL: Conjunto de Dados Bupa e Método MB, com C1 = um 14

6.1.4 Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método MB, com C1 = um 14

6.1.5 Regras Geradas por FOIL: Conjunto de Dados Bupa e Método Prop, com C2 = dois 15

6.1.6 Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método Prop, com C2 = dois 15

6.1.7 Regras Geradas por FOIL: Conjunto de Dados Bupa e Método MB, com C2 = dois 16

6.1.8 Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método MB, com C2 = dois 16

6.2.1 Regras Geradas por FOIL: Conjunto de Dados Pima e Método Prop, com C1 = zero 20

6.2.2 Regras Geradas por PROGOL: Conjunto de Dados Pima e Método Prop, com C1 = zero 21

6.2.3 Regras Geradas por FOIL: Conjunto de Dados Pima e Método MB, com C1 = zero 22

6.2.4 Regras Geradas por PROGOL: Conjunto de Dados Pima e Método MB, com C1 = zero 23

6.2.5 Regras Geradas por FOIL: Conjunto de Dados Pima e Método Prop, com C2 = um 23

6.2.6 Regras Geradas por PROGOL: Conjunto de Dados Pima e Método Prop, com C2 = um 24

6.2.7 Regras Geradas por FOIL: Conjunto de Dados Pima e Método MB, com C2 = um 24

6.2.8 Regras Geradas por PROGOL: Conjunto de Dados Pima e Método MB, com C2 = um 25

6.3.1 Regras Geradas por FOIL: Conjunto de Dados Breast e Método Prop, com C1 = recurrence 28

6.3.2 Regras Geradas por PROGOL: Conjunto de Dados Breast e Método Prop, com C1 = recurrence 29

6.3.3 Regras Geradas por FOIL: Conjunto de Dados Breast e Método MB, com C1 = recurrence 29

6.3.4 Regras Geradas por PROGOL: Conjunto de Dados Breast e Método MB, com C1 = recurrence 29

6.3.5 Regras Geradas por FOIL: Conjunto de Dados Breast e Método Prop, com C2 = no_recurrence 30

6.3.6 Regras Geradas por PROGOL: Conjunto de Dados Breast e Método Prop, com C2 = no_recurrence 30

6.3.7 Regras Geradas por FOIL: Conjunto de Dados Breast e Método MB, com C2 = no_recurrence 31

6.3.8 Regras Geradas por PROGOL: Conjunto de Dados Breast e Método MB, com C2 = no_recurrence 31

6.4.1 Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método Prop, com C1 = presence 34

6.4.2 Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método Prop, com C1 = presence 34

6.4.3 Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método MB, com C1 = presence 35

6.4.4 Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método MB, com C1 = presence 35

6.4.5 Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método Prop, com C2 = absence 35

6.4.6 Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método Prop, com C2 = absence 36

6.4.7 Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método MB, com C2 = absence 36

6.4.8 Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método MB, com C2 = absence 36

6.5.1 Regras Geradas por FOIL: Conjunto de Dados Credit e Método Prop, com C1 = positive 39

6.5.2 Regras Geradas por PROGOL: Conjunto de Dados Credit e Método Prop, com C1 = positive 40

6.5.3 Regras Geradas por FOIL: Conjunto de Dados Credit e Método MB, com C1 = positive 40

6.5.4 Regras Geradas por PROGOL: Conjunto de Dados Credit e Método MB, com C1 = positive 40

6.5.5 Regras Geradas por PROGOL: Conjunto de Dados Credit e Método MB, com C2 = negative 40

6.5.6 Regras Geradas por FOIL: Conjunto de Dados Credit e Método Prop, com C2 = negative . . . 41

6.5.7 Regras Geradas por PROGOL: Conjunto de Dados Credit e Método Prop, com C2 = negative . . . 41

6.5.8 Regras Geradas por FOIL: Conjunto de Dados Credit e Método MB, com C2 = negative . . . 41

6.6.1 Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método Prop, com C1 = die . . . 44

6.6.2 Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método Prop, com C1 = die . . . 44

6.6.3 Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método MB, com C1 = die . . . 44

6.6.4 Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método MB, com C1 = die . . . 44

6.6.5 Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método Prop, com C2 = live . . . 45

6.6.6 Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método Prop, com C2 = live . . . 45

6.6.7 Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método MB, com C2 = live . . . 45

6.6.8 Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método MB, com C2 = live . . . 45

Lista de Tabelas

3.2.1 Resumo de Características dos Conjuntos de Dados . . . 5

5.1 Desempenho de Classificação para Duas Classes . . . 7

6.1.1 Bupa – Descrição dos Atributos . . . 11

6.1.2 Médias para o Conjunto de Dados Bupa e Método Prop, (C1 = um e C2 = dois) . 11

6.1.3 Médias para o Conjunto de Dados Bupa e Método MB, (C1 = um e C2 = dois) . . 12

6.1.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 12

6.1.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 12

6.1.6 Erro Aparente para o Conjunto de Dados Bupa (C1 = um e C2 = dois) . . . 13

6.1.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Bupa (C1 = um e C2 = dois) . . . 13

6.2.1 Pima – Descrição dos Atributos 17

6.2.2 Médias para o Conjunto de Dados Pima e Método Prop, (C1 = zero e C2 = um) . 17

6.2.3 Médias para o Conjunto de Dados Pima e Método MB, (C1 = zero e C2 = um) . . 18

6.2.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 18

6.2.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 18

6.2.6 Erro Aparente para o Conjunto de Dados Pima (C1 = zero e C2 = um) 19

6.2.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Pima (C1 = zero e C2 = um) 19

6.3.1 Breast – Descrição dos Atributos 26

6.3.2 Médias para o Conjunto de Dados Breast e Método Prop, (C1 = recurrence e C2 = no_recurrence) 26

6.3.3 Médias para o Conjunto de Dados Breast e Método MB, (C1 = recurrence e C2 = no_recurrence) 27

6.3.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 27

6.3.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 27

6.3.6 Erro Aparente para o Conj. de Dados Breast (C1 = recurrence e C2 = no_recurrence) 28

6.3.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conj. de Dados Breast (C1 = recurrence e C2 = no_recurrence) 28

6.4.1 Hungaria – Descrição dos Atributos 32

6.4.2 Médias para o Conjunto de Dados Hungaria e Método Prop, (C1 = presence e C2 = absence) 32

6.4.3 Médias para o Conjunto de Dados Hungaria e Método MB, (C1 = presence e C2 = absence) 33

6.4.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 33

6.4.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 33

6.4.6 Erro Aparente para o Conjunto de Dados Hungaria (C1 = presence e C2 = absence) 34

6.4.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Hungaria (C1 = presence e C2 = absence) 34

6.5.1 Credit – Descrição dos Atributos 37

6.5.2 Médias para o Conjunto de Dados Credit e Método Prop, (C1 = positive e C2 = negative) 37

6.5.3 Médias para o Conjunto de Dados Credit e Método MB, (C1 = positive e C2 = negative) 38

6.5.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 38

6.5.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 38

6.5.6 Erro Aparente para o Conjunto de Dados Credit (C1 = positive e C2 = negative) . 39

6.5.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Credit (C1 = positive e C2 = negative) . 39

6.6.1 Hepatitis – Descrição dos Atributos 42

6.6.2 Médias para o Conjunto de Dados Hepatitis e Método Prop, (C1 = die e C2 = live) 42

6.6.3 Médias para o Conjunto de Dados Hepatitis e Método MB, (C1 = die e C2 = live) 43

6.6.4 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop . 43

6.6.5 Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB . . 43

6.6.6 Erro Aparente para o Conjunto de Dados Hepatitis (C1 = die e C2 = live) 44

6.6.7 Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Hepatitis (C1 = die e C2 = live) 44

7.1 Taxas Médias de Erro e Número Médio de Cláusulas e Regras 46

7.2 Diferenças em Desvios-Padrão dos Erros entre FOIL e PROGOL 47

1 Introdução

O objetivo de Aprendizado de Máquina – AM – é desenvolver métodos, técnicas e ferramentas para construir *máquinas inteligentes*, que se modificam para realizar cada vez melhor sua(s) tarefa(s). 'Melhor' nesse sentido pode ser trabalhar com mais eficiência ou precisão, ou ainda de forma a abranger uma classe maior de problemas. Para tanto, essas máquinas devem ser capazes de aprender.

Em AM indutivo supervisionado, apresenta-se um conjunto de instâncias (exemplos) de treinamento a um algoritmo de indução, onde cada instância é descrita por um vetor de atributo-valores e um rótulo de classe, ou seja, através de um formato atributo-valor. A tarefa do algoritmo indutor consiste em induzir um classificador que pode ser útil para classificar novas instâncias.

Vários formalismos lógicos têm sido utilizados em sistemas de aprendizado indutivo para a representação de exemplos e conceitos. Em geral, dividem-se em dois tipos de descrição: *descrição baseada em atributos ou proposicional* e *descrição relacional*.

No primeiro caso, as linguagens de descrição de conceitos utilizadas pelos algoritmos para expressar as hipóteses induzidas, por exemplo árvores de decisão e regras de produção, podem ser tratadas como variantes de linguagens baseadas em atributos, *i.e.* linguagens proposicionais. No entanto, o aprendizado baseado em atributos é limitado a descrições não-relacionais pois as descrições aprendidas *não* especificam *relações* entre as partes dos objetos. Com isso, a inclusão de conhecimento prévio do domínio é limitada e a própria representação se torna inapropriada para domínios com relações intrínsecas.

Linguagens de descrição relacional, ou baseadas em lógica de primeira ordem, são usadas há mais de 30 anos [Sammut, 1993]. No entanto, aprendizado empírico de primeira ordem só começou a atrair atenção maciça no início dos anos 90. Programação Lógica Indutiva – PLI – é uma abordagem relativamente recente de AM indutivo, que busca contornar algumas das limitações das abordagens anteriores de AM através do uso de teoria do domínio (ou conhecimento prévio) estruturada e uma linguagem de descrição de exemplos e conceitos baseada em lógica de primeira ordem. A adoção por sistemas de aprendizado de linguagens lógicas de primeira ordem como linguagens de representação permite que relações e/ou estruturas possam ser induzidas.

Sistemas de aprendizado de primeira ordem (ou de PLI), como Foil [Quinlan, 1990] e Progol [Muggleton, 1995], já foram demonstrados diversas vezes em problemas de aprendizado nos quais definições recursivas de relações são procuradas a partir de listas de constantes discretas (nominais) [Quinlan e Cameron-Jones, 1993], [Muggleton, 1997]. Problemas desse tipo parecem muito distantes do tipo de problema de aprendizado atributo-valor tratado por sistemas de zero ordem ou de aprendizado proposicional, como $\mathcal{CN}2$ [Clark e Niblett, 1989] e $\mathcal{C}4.5$ [Quinlan, 1993], especialmente quando tratam de valores contínuos. Com algumas mudanças na especificação original

de sistemas de aprendizado relacional, tornou-se possível definir variáveis contínuas que podem ser comparadas com um limite numérico. Através desse tratamento de valores numéricos, sistemas de PLI podem ser aplicados não só na busca de relações entre atributos, mas também a problemas de classificação atributo-valor convencionais. Assim, é interessante verificar o comportamento de sistemas de PLI nesse tipo de problema.

Em princípio, qualquer problema que possa ser resolvido utilizando AM proposicional também pode ser resolvido com AM relacional. Para tanto, cada exemplo no formato atributo-valor pode ser representado utilizando predicado(s) cujos argumentos são os valores dos atributos para aquele exemplo. Assim, um primeiro passo na conversão do formato atributo-valor para o formato relacional seria transformar os exemplos em fatos utilizando a notação relacional. No entanto, isso também força o sistema a considerar todos os atributos que estão contidos nos predicados, embora pode ser que nem todos sejam necessários.

Em aprendizado relacional, necessita-se de conhecimento de fundo sobre um domínio de aplicação para formar a hipótese sobre o conceito desejado. Uma forma de fazer isso é numerar os exemplos, e apresentar os valores dos atributos como sendo o conhecimento de fundo sobre o exemplo. Ou seja, cada atributo pode se relacionar com o exemplo através de um predicado [Morik e Brockhausen, 1997]. Assim os dados assumem um formato ainda mais "relacional", o qual é denominado neste trabalho de formato MB. Com isso, o sistema pode considerar somente aqueles predicados que são interessantes para a definição do conceito.

Neste trabalho, é descrita uma comparação do aprendizado de conceitos a partir de seis conjuntos de dados do *UCI Machine Learning Repository* (Repositório de Aprendizado de Máquina da Universidade de Califórnia-Irvine - [Blake et al., 1998]), utilizando os sistemas de PLI Foil e Progol. Inicialmente esses conjuntos estão no formato atributo-valor. Ferramentas, descritas em [Caulkins e Monard, 2000], são utilizadas para converter os dados no formato atributo-valor para os formatos "proposicional" e MB. Os dados são apresentados a Foil e Progol tanto no formato "proposicional" quanto no formato MB, e o desempenho de ambos sistemas é medido em cada caso.

Depois de realizado o aprendizado, os resultados obtidos com cada conjunto de dados são analisados utilizando uma matriz de confusão. Além disso, várias medidas são aplicadas ao conjunto de regras obtido de modo a avaliar seu desempenho.

O trabalho está organizado da seguinte forma: Na seção 2 são descritos os sistemas de aprendizado relacional utilizados neste trabalho. Uma descrição dos conjuntos de dados da UCI é esboçada na seção 3, explicando de onde cada um foi retirado e mostrando em linhas gerais os atributos que os compõem. Na seção 4, é descrita a preparação dos experimentos. As medidas utilizadas para avaliar os resultados obtidos são explicadas na seção 5. Na seção 6 são apresentados os resultados experimentais obtidos para cada conjunto de dados. Na seção 7, os resultados obtidos com Foil

e com PROGOL são comparados, procurando salientar algumas vantagens e desvantagens de cada sistema no tratamento desses conjuntos de dados. E, por fim, algumas considerações finais são elaboradas na seção 8.

2 Sistemas de PLI Utilizados

Os sistemas de PLI contemplados nesse trabalho, FOIL e PROGOL, possuem algumas características em comum, como o fato de fazerem uma busca "top-down" no espaço de hipóteses, ou seja, do geral para o específico. Diferem principalmente quanto à forma de inclusão de conhecimento prévio do domínio. Ambos têm sido aplicados em diversos domínios de aplicação.

2.1 Foil

É um sistema que lê especificações extensionais de um conjunto de relações e produz hipóteses compostas de cláusulas de Horn para uma ou mais delas [Quinlan, 1990]. O sistema aprende um predicado por vez, de uma forma não-interativa e não-incremental, fazendo uma busca "top-down" (do geral para o específico) no espaço de hipóteses. Seu conhecimento prévio deve ser sempre em forma de listas de fatos.

FOIL aproveita algumas idéias do paradigma de aprendizado baseado em atributo-valor, utilizando um método de cobertura semelhante ao de AQ [Michalski, 1983] e uma heurística baseada em informação semelhante ao de ID3 [Quinlan, 1986]. Além disso, herda a idéia de busca "top-down" em grafos de refinamento do MIS [Shapiro, 1983]. Seu critério de aceitação de uma hipótese baseia-se no Ganho de Informação obtido, procurando encontrar cláusulas com comprimento mínimo de descrição, que possam substituir os exemplos sendo considerados, obtendo assim uma representação mais compacta do conjunto de dados.

2.2 Progol

PROGOL aprende múltiplos predicados, de forma não-interativa e empírica. Permite a inclusão de conhecimento prévio descrito de forma intensional. Utiliza uma estratégia mista de busca no espaço de hipóteses. Isto é, além de fazer uma busca do geral para o específico, também faz uma busca no sentido inverso ("bottom-up"). Inicialmente procura uma hipótese menos geral para servir de limite inferior ao reticulado de hipóteses. Faz uma busca do geral ao específico através de um grafo de refinamento das cláusulas, ou seja, de uma forma "top-down" [Muggleton, 1995]. PROGOL também pode ser usado para aprender somente com exemplos positivos. Seu critério de aceitação se baseia na compressão dos exemplos obtida por cada cláusula.

3 Conjuntos de Dados

Os experimentos aqui descritos foram realizados com dados de vários domínios diferentes. Todos os conjuntos de dados são do UCI ML Repository [Blake et al., 1998], um repositório de dados, de domínio público, para aprendizado de máquina.

Para tornar as comparações mais interessantes, os conjuntos de dados foram escolhidos de modo que contivessem tipos de atributos bastante variados. Alguns conjuntos contem somente valores contínuos, enquanto outros combinam atributos discretos com atributos contínuos. Há ainda conjuntos contendo valores desconhecidos. Para melhor entendimento, uma descrição do conteúdo de cada conjunto, bem como as características gerais dos conjuntos, são apresentadas a seguir.

3.1 Descrição Geral dos Conjuntos de Dados

Bupa Contribuição de R. S. Forsyth à base UCI, consiste em dados que descrevem pacientes masculinos, através de vários exames de sangue e o volume de álcool consumido. Os pacientes dividem-se em "com problemas de fígado" — classe *um* — e "sem indícios de problemas" — classe *dois*.

Pima Doado por V. Sigillito do Laboratório de Física Aplicada da Universidade Johns Hopkins (EUA), trata-se de um subconjunto de uma base de dados maior mantida pelo Instituto Nacional de Diabete e Doenças Digestivas e de Rim, que descreve mulheres com pelo menos 21 anos, descendentes da tribo indígena Pima. O problema consiste em predizer se uma paciente irá testar como positiva — classe *um* — ou negativa — classe *zero* — em relação à diabete de acordo com os critérios da Organização Mundial da Saúde, a partir de resultados de exames médicos e medidas fisiológicas.

Breast Um de dois conjuntos de dados no repositório da UCI que tratam de câncer de mama. Divide-se em casos de recorrência — classe *recurrence* — de câncer e outros em que não houve recorrência — classe *no_recurrence*.

Hungaria Trata de doenças cardíacas em casos reunidos pelo *Hungarian Institute of Cardiology*. Divide-se em casos nos quais o paciente mostra a presença de uma doença — classe *presence* — ou a ausência da mesma — classe *absence*.

Credit Utilizado para aprovação de crédito a clientes, divide-se em casos de aceitação — classe *positive* — e rejeição — classe *negative* — de crédito.

Hepatitis Reúne casos com características de pacientes com hepatite. Divide-se em casos de pacientes com esperança de vida — classe *live* — e sem esperança de vida — classe *die*.

3.2 Características dos Conjuntos de Dados

A Tabela 3.2.1 mostra um resumo das características dos conjuntos de dados utilizados para realizar as comparações entre FOIL e PROGOL. Mostra-se, para cada conjunto, o número de instâncias (#Inst.), número e porcentagem de instâncias duplicadas (que aparecem mais de uma vez) e/ou conflitantes (mesmos valores dos atributos mas com classificações diferentes), número de atributos (#Atrib.) contínuos e nominais, distribuição das classes, erro majoritário e se o conjunto de dados tem pelo menos um valor desconhecido (Valores Desconh.)[1].

Conj. de Dados	#Inst.	#Duplic. ou Conflit. (%)	#Atrib. (cont.,nom.)	Classe	% Classe	Erro Majoritário	Valores Desconh.
bupa	345	4 (1.16%)	6 (6,0)	"1" "2"	42.03% 57.97%	42.03% em "2"	N
pima	769	1 (0.13%)	8 (8,0)	"0" "1"	65.02% 34.98%	34.98% em "0"	N
breast	285	2 (0.7%)	9 (4,5)	"recurrence" "no_recurrence"	29.47% 70.53%	29.47% em "no_recurrence"	Y
hungaria	294	1 (0.34%)	13 (13,0)	"presence" "absence"	36.05% 63.95%	36.05% em "absence"	Y
credit	490	0 (0%)	15 (6,9)	"positive" "negative"	44.29% 55.71%	44.29% em "negative"	Y
hepatitis	155	0 (0%)	19 (6,13)	"die" "live"	20.65% 79.35%	20.65% em "live"	Y

Tabela 3.2.1: Resumo de Características dos Conjuntos de Dados

Os conjuntos são apresentados em ordem crescente de acordo com o número de atributos de cada um [Lee et al., 1999], ordenação esta que será mantida na descrição dos resultados. A Figura 3.2.1 mostra a dimensionalidade dos conjuntos de dados, isto é, o número de atributos e instâncias em cada conjunto. Deve ser observado que devido à grande variação, o número de instâncias é representado na figura como $\log_{10}$(#Instâncias).

4 Descrição dos Experimentos

Os sistemas de PLI aqui tratados aprendem um conceito que diferencia entre duas classes, induzindo uma hipótese que cobre os exemplos de uma das classes — classe positiva — de modo que qualquer exemplo pertencente à outra classe — classe negativa — não seja coberto. Para que os sistemas aprendam tanto o conceito referente à primeira classe (X) quanto o conceito referente à segunda classe (Y), é necessário executá-los duas vezes. Inicialmente, os exemplos da classe X são fornecidos como exemplos positivos e os exemplos da classe Y são fornecidos como exemplos

[1] Estas informações foram obtidas utilizando a ferramenta *info* contida no $\mathcal{MLC}$++.

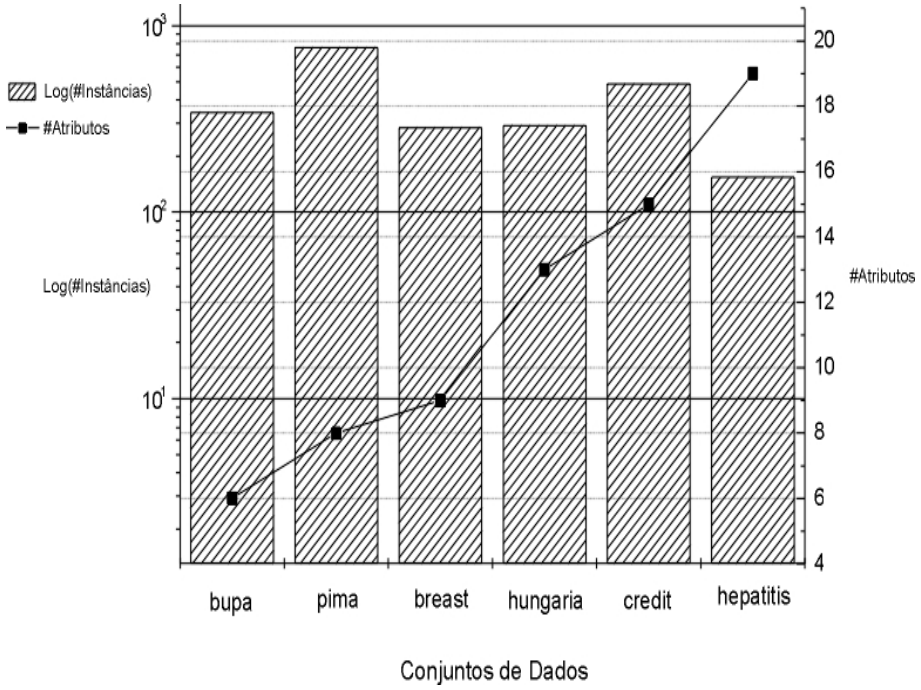

Figura 3.2.1: Dimensionalidade dos Conjuntos de Dados

negativos desse conceito. Na execução seguinte é feito o processo contrário, fornecendo os exemplos da classe Y como exemplos positivos e os exemplos da classe X como exemplos negativos. Neste trabalho ambos conceitos foram induzidos.

Os sistemas de aprendizado FOIL e PROGOL foram executados com os parâmetros padrões. A única alteração feita foi no sentido de alterar o parâmetro de erro permitido no PROGOL para um máximo de 20% de erro nas regras geradas. Isto foi feito com a finalidade de equipará-las com as regras geradas por FOIL, já que a taxa de erro padrão para as regras deste último é de 20%. Vale ressaltar ainda que todos os atributos numéricos (contínuos) foram considerados como sendo do mesmo tipo "real", e todos os atributos booleanos, ou seja, que tivessem valores "yes" ou "no", foram marcados como sendo do tipo "boolean".

Para estimar a taxa de erro verdadeira, foi escolhido o método de "resampling" chamado "10-fold cross validation". Este método é aconselhável para conjuntos de exemplos de tamanho médio como os utilizados neste trabalho, pois é capaz de combinar eficiência computacional e boa precisão para estimar a taxa de erro verdadeira.

Os conjuntos de exemplos foram embaralhados e divididos em 10 partições mutuamente exclusivas, utilizando as ferramentas para particionamento de exemplos presentes no ambiente *AMPSAM* [Batista, 1997]. Posteriormente, foram construídos os conjuntos de treinamento e teste segundo

o método "10-fold cross validation". Um aspecto importante a ser destacado nos experimentos realizados, é que exatamente as mesmas partições foram utilizadas para medir ambos algoritmos. Ou seja, para cada conjunto de exemplos, os sistemas foram medidos utilizando os mesmos 10 conjuntos de treinamento e teste. Este procedimento pode, eventualmente, auxiliar em uma análise mais minuciosa para entender as possíveis diferenças de performance entre os dois sistemas de aprendizado. Após a construção dos conjuntos de treinamento e teste, foram aplicados os conversores sintáticos, descritos em [Caulkins e Monard, 2000], para o formato de FOIL e PROGOL, colocando os dados no formato atributo-valor, ou "proposicional", e também no formato relacional através do método MB.

5 Medidas Consideradas

A *matriz de confusão* oferece uma medida da eficiência do modelo de classificação, *i.e.* a hipótese h induzida, mostrando o número de classificações corretas versus as classificações preditas para cada classe. Para exemplificar, considera-se um problema de classificação com duas classes, rotuladas como "+" e "−". Os dois tipos de erro possíveis são chamados de *falso positivo* e *falso negativo* [Baranauskas e Monard, 2000b]. A Tabela 5.1 ilustra a matriz de confusão correspondente onde T_P é o número de exemplos positivos classificados corretamente, ou *verdadeiros positivos*, F_N é o número de instâncias positivas classificadas de forma errada, ou *falsos negativos*, F_P é o número de exemplos negativos classificados de forma errada, ou *falsos positivos*, e T_N é o número de instâncias negativas classificadas corretamente, ou *verdadeiros negativos*, em um total de $n = (T_P + F_N + F_P + T_N)$ instâncias.

Rótulo da classe	predito como C_+	predito como C_-	Taxa de Erro da Classe	Taxa de erro total
	Verd. positivos	Falsos negativos		
verd. C_+	T_P	F_N	$\frac{F_N}{T_P+F_N}$	$\frac{F_P+F_N}{n}$
	Falsos positivos	Verd. negativos		
verd. C_-	F_P	T_N	$\frac{F_P}{F_P+T_N}$	

Tabela 5.1: Desempenho de Classificação para Duas Classes

Essas quatro situações são ilustradas na Figura 5.1 para a <classe = C_+>, isto é:

1. a instância pertence à classe C_+ e é predito pelo classificador como sendo da classe C_+. Nesse caso, essa instância é um *verdadeiro positivo.*

2. a instância pertence à classe C_- e é predito pelo classificador como sendo da classe C_-. Nesse caso, essa instância é um *verdadeiro negativo.*

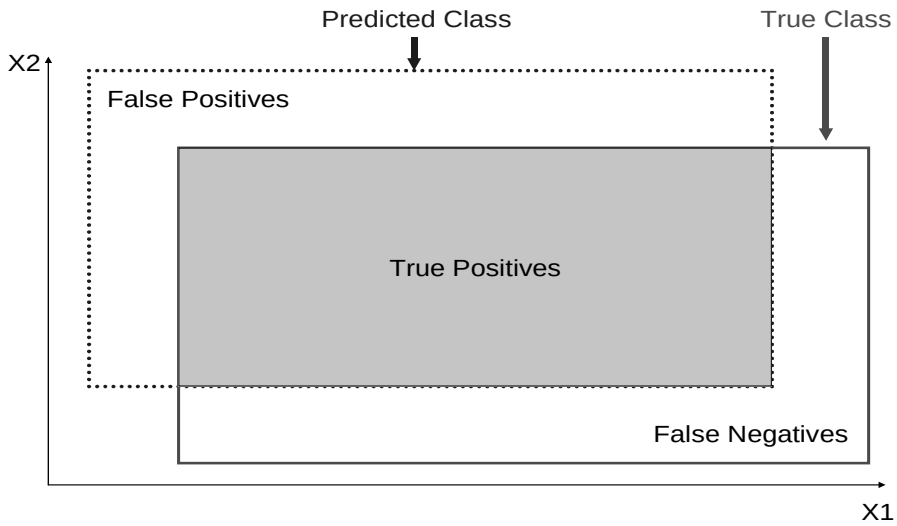

Figura 5.1: Verdadeiros Positivos, Falsos Positivos e Falsos Negativos

3. a instância pertence à classe C_- e é predito pelo classificador como sendo da classe C_+. Nesse caso, essa instância é um *falso positivo*.

4. a instância pertence à classe C_+ e é predito pelo classificador como sendo da classe C_-. Nesse caso, essa instância é um *falso negativo*.

A *taxa de erro* é uma métrica bastante utilizada para avaliar o desempenho de sistemas de aprendizado. Dentre as possíveis taxas, as utilizadas neste trabalho são:

1. a taxa de erro relativo à classe positiva — Equação 1;

2. a taxa de erro relativo à classe negativa — Equação 2;

3. o erro total, onde $n = (T_P + F_N + F_P + T_N)$ é o número total de instâncias — Equação 3; e

4. a média G, uma variação da média geométrica a qual é apropriada para situações nas quais o número de exemplos em cada uma das classes X e Y está muito desbalanceado — Equação 4.

$$\text{Erro em } C_+ = \frac{F_N}{T_P + F_N} \tag{1}$$

$$\text{Erro em } C_- = \frac{F_P}{F_P + T_N} \tag{2}$$

$$\text{Erro total} = \frac{F_P + F_N}{n} \tag{3}$$

$$\text{Média G} = \sqrt{\frac{F_N}{T_P + F_N}\frac{F_P}{F_P + T_N}} \tag{4}$$

Outra medida utilizada é a *diferença em desvios-padrão* do erro. Suponha que se queira comparar dois algoritmos com taxas de erro iguais a $9,0 \pm 1,0$ e $7,5 \pm 0,8$. Será que um deles é melhor que o outro? *i.e.*, que existe uma diferença significativa de um para o outro com relação às taxas de erro apresentadas por eles, com nível de confiança de 95%? Para responder essa pergunta, assume-se o caso geral para determinar se a diferença de um algoritmo para outro — por exemplo A_1 e A_2 — é significativa ou não. Para tanto, a média e o desvio padrão combinados são calculados de acordo com as Equações 5 e 6, respectivamente. A diferença absoluta em desvios-padrão é calculada através da Equação 7 [Weiss e Indurkhya, 1998].

$$\mathsf{mean}(A_2 - A_1) = \mathsf{mean}(A_2) - \mathsf{mean}(A_1) \tag{5}$$

$$\mathsf{sd}(A_2 - A_1) = \sqrt{\frac{\mathsf{sd}(A_2)^2 + \mathsf{sd}(A_1)^2}{2}} \tag{6}$$

$$\mathsf{ad}(A_2 - A_1) = \frac{\mathsf{mean}(A_2 - A_1)}{\mathsf{sd}(A_2 - A_1)} \tag{7}$$

Se $\mathsf{ad}(A_2 - A_1) > 0$, pode-se dizer que A_1 tem um desempenho melhor que A_2. Se $\mathsf{ad}(A_2 - A_1) \geq 2$ desvios-padrão, pode-se dizer que A_1 tem melhor desempenho que A_2 com nível de confiança de 95%. Por outro lado, se $\mathsf{ad}(A_2 - A_1) < 0$ diz-se que A_2 tem um desempenho melhor que A_1 e se $\mathsf{ad}(A_2 - A_1) \leq 2$ desvios-padrão então A_2 tem um desempenho melhor que A_1 com nível de confiança de 95%.

6 Resultados Experimentais

Para cada conjunto de dados foram construídas várias tabelas que procuram detalhar os resultados experimentais obtidos. Nas seções que se seguem, serão apresentadas essas tabelas.

- Na primeira tabela é descrito cada atributo do conjunto de dados, através de um número, o nome do atributo e o tipo (discreto ou contínuo). Para atributos discretos, o número máximo de valores possíveis e o número de valores realmente atribuídos ao atributo no conjunto, ambos calculados pela ferramenta *info* do $\mathcal{MLC}$++, são mostrados. Deve ser observado que um número maior de valores encontrados em relação ao de valores possíveis indica que há valores faltando para aquele atributo, porém o contrário não é verdade.

- A segunda tabela consiste de duas partes, sendo que os valores foram calculados para os experimentos realizados utilizando o método "proposicional" e "10-fold cross-validation". Na parte (a) são mostrados o valor médio e desvio-padrão, para as taxas de erro relativo à classe positiva e à classe negativa, o erro total e o erro baseado na média G. Também são indicados os valores máximo e mínimo de cada taxa. Além disso, são mostradas matrizes de confusão onde cada posição contém o valor médio para os 10 "folds".

  Na parte (b) dessa tabela são mostrados o número médio de regras geradas (cláusulas com corpo) e o número médio do total de cláusulas (com corpo e sem) para se ter uma idéia do tamanho das hipóteses induzidas e da cobertura das regras. Na quarta coluna está indicado o tempo médio de execução e seu desvio-padrão. O tempo de CPU nas tabelas refere-se à execução de Progol em um Pentium II com clock de 233 MHz, rodando Linux como sistema operacional, e à execução de Foil em uma estação de trabalho Sun Ultra1 com clock de 140 MHz, rodando o Solaris como sistema operacional.

- Na terceira tabela são mostrados os mesmos números que na segunda tabela mas para os experimentos realizados utilizando o método MB para representar os dados.

- Na quarta tabela é mostrada a diferença em desvios-padrão de Foil para Progol quando o método "proposicional" foi utilizado.

- Na quinta tabela é mostrada a diferença em desvios-padrão de Foil para Progol quando o método MB foi utilizado.

- Na sexta tabela são mostradas as taxas de erro aparente relativo à classe positiva e à classe negativa, e o erro total, obtidos quando regras são induzidas sobre todo o conjunto de dados. Da segunda à quinta coluna os resultados utilizando a representação "proposicional" são indicados e nas quatro colunas seguintes são mostrados resultados obtidos com a representação MB.

- Na última tabela é mostrado o número total de cláusulas geradas quando regras são induzidas utilizando todo o conjunto de dados, o número de regras (cláusulas com corpo) induzidas, e o tempo de execução necessário para fazer a indução, através da representação "proposicional" e da representação MB. Cada linha mostra o resultado de um dos sistemas, levando em consideração cada uma das duas classes como "positiva".

  Deve ser observado que a diferença entre o número total de cláusulas e regras fornece o número de exemplos positivos não cobertos pelas regras induzidas, mas que são cobertos por uma cláusula sem corpo que simplesmente nomeia o exemplo.

- Finalmente, são mostradas oito figuras com as regras geradas em cada um dos casos descritos na última tabela.

6.1 Conjunto de Dados Bupa

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	mcv	-	26	contínuo
#1	alkphos	-	78	contínuo
#2	sgpt	-	67	contínuo
#3	sgot	-	47	contínuo
#4	gammagt	-	94	contínuo
#5	drinks	-	16	contínuo

Tabela 6.1.1: Bupa – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	8.5	6.0	41.6 ± 10.7	29.6 ± 8.9	34.0 ± 4.3	33.8 ± 4.4
$[max, min]$	5.8	14.2	[60.0, 25.0]	[46.7, 13.6]	[42.9, 26.5]	[42.4, 26.1]
Progol(C1)	7.0	7.5	51.8 ± 11.5	24.4 ± 11.5	35.8 ± 6.7	33.6 ± 9.4
$[max, min]$	4.9	15.1	[75.0, 35.7]	[40.9, 5.0]	[42.9, 20.6]	[43.2, 14.7]
Foil(C2)	14.1	5.9	29.6 ± 10.5	38.1 ± 8.4	32.9 ± 6.5	32.5 ± 7.3
$[max, min]$	5.5	9.0	[45.5, 10.0]	[50.0, 21.4]	[47.1, 22.9]	[47.7, 20.0]
Progol(C2)	12.5	7.5	37.6 ± 9.6	39.9 ± 13.7	38.1 ± 6.6	37.6 ± 7.7
$[max, min]$	5.7	8.8	[50.0, 22.2]	[66.7, 21.4]	[55.9, 31.4]	[57.8, 29.0]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	26.5 ± 2.6	20.4 ± 4.4	0.6 ± 4.4
Progol(C1)	51.5 ± 0.5	15.1 ± 1.3	303.2 ± 1.6
Foil(C2)	27.3 ± 3.9	23.7 ± 4.3	0.6 ± 4.3
Progol(C2)	44.1 ± 0.1	15.2 ± 1.3	222.9 ± 14.6

(b)

Tabela 6.1.2: Médias para o Conjunto de Dados Bupa e Método Prop, (C1 = um e C2 = dois)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	7.7	6.8	46.4 ± 11.7	25.1 ± 11.8	34.1 ± 7.1	32.7 ± 7.0
$[max, min]$	5.0	15.0	[66.7, 28.6]	[45.5, 10.0]	[45.7, 20.6]	[45.8, 20.7]
Progol(C1)	5.5	9.0	62.3 ± 13.3	13.7 ± 8.1	33.7 ± 6.8	27.2 ± 10.9
$[max, min]$	2.7	17.3	[91.7, 47.4]	[26.7, 0.0]	[44.1, 23.5]	[40.9, 7.6]
Foil(C2)	13.4	6.6	32.8 ± 9.5	29.0 ± 13.0	31.5 ± 7.9	29.5 ± 8.5
$[max, min]$	4.3	10.2	[50.0, 20.0]	[52.6, 7.1]	[50.0, 22.9]	[50.0, 17.9]
Progol(C2)	12.1	7.9	40.0 ± 11.6	36.6 ± 11.4	38.2 ± 5.3	36.9 ± 5.7
$[max, min]$	5.3	9.2	[55.0, 23.8]	[58.3, 14.3]	[42.9, 28.6]	[43.0, 28.1]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	57.1 ± 12.8	35.3 ± 13.9	95.3 ± 28.5
Progol(C1)	66.9 ± 1.2	11.3 ± 1.8	301.1 ± 2.6
Foil(C2)	49.2 ± 3.4	32.5 ± 4.3	82.7 ± 29.6
Progol(C2)	57.1 ± 4.3	14.1 ± 4.4	227.1 ± 21.1

(b)

Tabela 6.1.3: Médias para o Conjunto de Dados Bupa e Método MB, (C1 = um e C2 = dois)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−0.9	+0.5	−0.3	0.0
C2	−0.8	−0.2	−0.8	−0.7

Tabela 6.1.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−1.3	+1.1	+0.1	+0.6
C2	−0.7	−0.6	−1.0	−1.0

Tabela 6.1.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop					Método MB				
Algoritmo	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
FOIL (C1)	137	8	0.06	0.01	0.03	145	0	0.00	0.00	0.00
	1	199				0	200			
PROGOL (C1)	145	0	0.00	0.06	0.03	145	0	0.00	0.05	0.03
	12	188				10	190			
FOIL (C2)	189	11	0.06	0.00	0.03	200	0	0.00	0.00	0.00
	0	145				0	145			
PROGOL (C2)	200	0	0.00	0.21	0.09	200	0	0.00	0.18	0.08
	31	114				26	119			

Tabela 6.1.6: Erro Aparente para o Conjunto de Dados Bupa (C1 = um e C2 = dois)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
FOIL (C1)	30	22	0.8	73	43	95.6
PROGOL (C1)	59	14	405.19	76	10	350.23
FOIL (C2)	34	23	0.7	46	33	61.4
PROGOL (C2)	52	14	269.29	71	11	279.67

Tabela 6.1.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Bupa (C1 = um e C2 = dois)

```
is_um(A,B,C,D,E,F) :- D<=21, C>21, E<=17.
is_um(A,B,C,D,E,F) :- C>E, B>61, A>℘7, B<=71, F>2.
is_um(A,B,C,D,E,F) :- C>D, E<=35, B>65, B<=℘2, C>E, C>24.
is_um(A,B,C,D,E,F) :- B>76, E<=20, A>℘℘, B<=94.
is_um(A,B,C,D,E,F) :- F>5, E<=32, C>25.
is_um(A,B,C,D,E,F) :- D<=17, F<=2, A>℘6, C>1℘.
is_um(A,B,C,D,E,F) :- C>32, E>69, E<=120, A>92.
is_um(A,B,C,D,E,F) :- F<=0.5, B>101, B<=119.
is_um(A,B,C,D,E,F) :- A>℘9, D<=23, C>27, A<=92, D>1℘.
is_um(A,B,C,D,E,F) :- E<=7.
is_um(A,B,C,D,E,F) :- D<=35, C>59.
is_um(A,B,C,D,E,F) :- D<=22, A>℘9, C>27, D>20.
is_um(A,B,C,D,E,F) :- B<=35.
is_um(A,B,C,D,E,F) :- D<=17, C>15, B>℘4.
is_um(A,B,C,D,E,F) :- C>42, B>7℘, A<=℘5, C<=47.
is_um(A,B,C,D,E,F) :- F>5, B<=E, A<=93, A>90, B<=93.
is_um(A,B,C,D,E,F) :- F<=3, D<=21, C>16, E<=22, A>℘5, A<=96, D>14.
is_um(A,B,C,D,E,F) :- A<=℘2, D<=1℘.
is_um(A,B,C,D,E,F) :- F>5, A>97, A<=99, B>57.
is_um(A,B,C,D,E,F) :- B<=54, B>49, C<=25, A>℘7, E>12, D<=24.
is_um(A,B,C,D,E,F) :- B>60, D<=℘.
```

Figura 6.1.1: Regras Geradas por FOIL: Conjunto de Dados Bupa e Método Prop, com C1 = um

```
example(A) :- is_um(A,B,C,D,E,F,G), 0.500=<G, 34=<D, F=<27.
example(A) :- is_um(A,B,C,D,E,F,G), 26=<D, E=<20, F=<19.
example(A) :- is_um(A,B,C,D,E,F,G), 23=<D, ℘℘=<B, F=<15.
example(A) :- is_um(A,91,B,C,D,E,F), ℘4=<B, F=<0.500.
example(A) :- is_um(A,B,C,D,E,F,G), ℘0=<C, 90=<B, D=<19.
example(A) :- is_um(A,B,C,70,D,E,F).
example(A) :- is_um(A,B,C,D,D,E,F), E=<16.
example(A) :- is_um(A,B,C,22,D,E,F), ℘9=<B.
example(A) :- is_um(A,B,C,D,22,E,F), 25=<D, 54=<C.
example(A) :- is_um(A,B,66,C,D,E,F).
example(A) :- is_um(A,B,C,D,E,F,G), 62=<C, E=<13.
example(A) :- is_um(A,B,77,C,D,E,F).
example(A) :- is_um(A,B,C,D,E,F,6.000), 92=<B, F=<36.
example(A) :- is_um(A,B,C,D,E,F,G), ℘.000=<G, 2℘=<D, E=<24.
```

Figura 6.1.2: Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método Prop, com C1 = um

```
is_um(A) :- a2(A,B), a3(A,C), a4(A,D), C<=21, B>21, D<=17.
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a4(A,E), a5(A,F), D>E, C>61, B>℘7, a4(G,C), F>2.
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), a4(A,F), D>E, F<=35, C>65, a4(G,C), B>℘6.
is_um(A) :- a2(A,B), a4(A,C), a5(D,C), a0(D,E), B>15, E<=℘7.
is_um(A) :- a1(A,B), a2(A,C), a3(A,D), a4(A,E), B>76, B<=℘0, C>E, C>D.
is_um(A) :- a5(A,B), B>12.
is_um(A) :- a0(A,B), a4(A,C), a5(A,D), D>5, a5(E,C), B>℘℘.
is_um(A) :- a2(A,B), a4(A,C), a5(A,D), D>5, B>30, C<=32.
is_um(A) :- a0(A,B), a2(A,C), a3(A,D), a4(A,E), B>℘9, D<=22, C>31, a3(F,E).
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), E<=2, C>7℘, D<=20, B>90.
is_um(A) :- a0(A,B), a2(A,C), a4(A,D), C>25, D<=21, a4(E,B), C<=30.
is_um(A) :- a0(A,B), a2(A,C), a4(A,D), a1(E,C), a0(E,F), D>62, B>91, F>92.
is_um(A) :- a0(A,B), a2(A,C), a4(A,D), a5(A,E), E<=2, B<=91, D>30, B>℘9, a4(F,C), E>0.
is_um(A) :- a1(A,B), a2(A,C), a3(A,D), a4(A,E), C>D, B<=71, B>65, E<=64.
is_um(A) :- a2(A,B), a3(A,C), C<=17, a1(D,B).
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), C<=44, a2(E,B), a5(E,F), a2(G,F), D>21.
```

Figura 6.1.3: Regras Geradas por FOIL: Conjunto de Dados Bupa e Método MB, com C1 = um

```
example(A) :- a2(A,B), a4(A,C), 26=<B, C=<19.
example(A) :- a1(A,66).
example(A) :- a4(A,B), a5(A,0.500), B=<10.
example(A) :- a0(A,90), a5(A,2.000).
example(A) :- a2(A,70).
example(A) :- a4(A,B), a5(A,3.000), B=<16.
example(A) :- a4(A,20).
example(A) :- a1(A,77).
example(A) :- a2(A,B), a5(A,℘.000), 29=<B.
example(A) :- a2(A,25), a4(A,B), B=<15.
```

Figura 6.1.4: Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método MB, com C1 = um

```
is_dois(A,B,C,D,E,F) :- C<=E, A<=℘9, F>1.
is_dois(A,B,C,D,E,F) :- C<=D, E>20, B<=92, D>19, C<=26.
is_dois(A,B,C,D,E,F) :- E>35, F<=5, F>2.
is_dois(A,B,C,D,E,F) :- C<=E, F<=4, B<=79, B>70.
is_dois(A,B,C,D,E,F) :- B<=65, E>21, F<=1, C<=30.
is_dois(A,B,C,D,E,F) :- C<=D, A<=℘℘, F>3.
is_dois(A,B,C,D,E,F) :- B<=65, C<=19, D>16, A>℘9.
is_dois(A,B,C,D,E,F) :- D>21, B<=60, A<=℘℘, D<=26.
is_dois(A,B,C,D,E,F) :- E>36, E<=62, D>32.
is_dois(A,B,C,D,E,F) :- B<=60, B>53, E>42.
is_dois(A,B,C,D,E,F) :- C<=21, A<=90, F>3, E>13.
is_dois(A,B,C,D,E,F) :- D>21, F<=2, C<=26, A<=℘7.
is_dois(A,B,C,D,E,F) :- E>20, D>23, C<=3℘, C>33.
is_dois(A,B,C,D,E,F) :- D>19, E>20, C<=30, B>71, A>B, F<=℘.
is_dois(A,B,C,D,E,F) :- C>40, F<=1, C<=41.
is_dois(A,B,C,D,E,F) :- B<=50, B>44, D>20.
is_dois(A,B,C,D,E,F) :- C<=19, A<=℘9, C>17.
is_dois(A,B,C,D,E,F) :- A<=92, D>41.
is_dois(A,B,C,D,E,F) :- C<=14, D<=20, F<=4, D>12.
is_dois(A,B,C,D,E,F) :- A<=℘5, E>17, E<=23.
is_dois(A,B,C,D,E,F) :- A>90, C<=24, B<=64, F<=5, E>7, D>15.
is_dois(A,B,C,D,E,F) :- A>99.
```

Figura 6.1.5: Regras Geradas por FOIL: Conjunto de Dados Bupa e Método Prop, com C2 = dois

```
example(A) :- is_dois(A,B,C,D,E,F,G), 25=<F, 94=<B, D=<30.
example(A) :- is_dois(A,B,C,D,E,F,4.000).
example(A) :- is_dois(A,B,C,D,E,F,G), 33=<E, B=<92, F=<94.
example(A) :- is_dois(A,B,C,D,E,E,F), 23=<E.
example(A) :- is_dois(A,B,C,D,E,F,G), 13=<F, C=<75, D=<19.
example(A) :- is_dois(A,B,C,D,E,F,5.000).
example(A) :- is_dois(A,B,C,D,25,E,F), 9=<E.
example(A) :- is_dois(A,B,B,C,D,E,F).
example(A) :- is_dois(A,B,C,D,E,4℘,F).
example(A) :- is_dois(A,B,C,D,22,E,F), B=<90.
example(A) :- is_dois(A,B,C,D,E,F,G), 42=<F, B=<℘℘.
example(A) :- is_dois(A,B,C,14,D,E,F).
example(A) :- is_dois(A,B,C,D,E,F,G), 64=<E.
example(A) :- is_dois(A,℘5,B,C,D,E,F), B=<65.
```

Figura 6.1.6: Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método Prop, com C2 = dois

```
is_dois(A) :- a0(A,B), a2(A,C), a4(A,D), a5(A,E), C<=D, B<=℘9, E>1.
is_dois(A) :- a1(A,B), a2(A,C), a3(A,D), a4(A,E), C<=D, E>20, B<=92, D>19, C<=26.
is_dois(A) :- a1(A,B), a2(A,C), a4(A,D), B<=59, C<=21, a2(E,D), a4(E,F), F<=10.
is_dois(A) :- a4(A,B), a5(A,C), B>35, C<=5, C>2.
is_dois(A) :- a0(A,B), a5(A,C), B<=℘7, a2(D,C).
is_dois(A) :- a2(A,B), a3(A,C), a4(A,D), B<=D, a5(E,C), D<=31.
is_dois(A) :- a1(A,B), a3(A,C), a4(A,D), a1(E,D), D<=62, C>26, B>54.
is_dois(A) :- a1(A,B), a2(A,C), a4(A,D), a5(A,E), B<=65, D>20, E<=1, C<=30.
is_dois(A) :- a1(A,B), a3(A,C), a4(A,D), C>21, a3(E,B), a2(E,F), D>1℘, F<=℘3.
is_dois(A) :- a0(A,B), a2(A,C), a3(A,D), C<=D, B<=℘℘, C>20.
is_dois(A) :- a0(A,B), a1(A,C), a2(A,D), D<=19, C<=75, B<=90, C>62.
is_dois(A) :- a2(A,B), a4(A,C), C>42, B<=34, B>29.
is_dois(A) :- a0(A,B), a3(A,C), C<=25, C>23, B>90, B<=96.
is_dois(A) :- a1(A,B), a2(A,C), a4(A,D), D>52, C>℘1, B>66.
is_dois(A) :- a1(A,B), a2(A,C), a3(A,D), a4(A,E), B<=64, C<=19, E<=13, D>13.
is_dois(A) :- a0(A,B), B>99.
is_dois(A) :- a1(A,B), a2(A,C), a5(A,D), D<=4, C>40, B>92.
is_dois(A) :- a0(A,B), a1(A,C), C<=65, B<=℘5, B>℘3.
is_dois(A) :- a1(A,B), a3(A,C), B<=50, B>44, C>20.
is_dois(A) :- a0(A,B), a2(A,C), C<=24, B>94, C>1℘.
is_dois(A) :- a0(A,B), a1(A,C), a2(A,D), D<=1℘, B<=℘9, a0(E,C), C>79.
```

Figura 6.1.7: Regras Geradas por FOIL: Conjunto de Dados Bupa e Método MB, com C2 = dois

```
example(A) :- a0(A,94), a4(A,B), 25=<B.
example(A) :- a5(A,4.000).
example(A) :- a2(A,B), a3(A,C), B=<16, C=<23.
example(A) :- a5(A,5.000).
example(A) :- a3(A,25).
example(A) :- a0(A,B), a3(A,C), 36=<C, B=<92.
example(A) :- a0(A,℘5), a2(A,B), B=<25.
example(A) :- a4(A,4℘).
example(A) :- a0(A,B), 100=<B.
example(A) :- a0(A,B), a4(A,C), 42=<C, B=<℘℘.
example(A) :- a2(A,21), a3(A,B), 17=<B.
```

Figura 6.1.8: Regras Geradas por PROGOL: Conjunto de Dados Bupa e Método MB, com C2 = dois

6.2 Conjunto de Dados Pima

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	Number	-	17	contínuo
#1	Plasma	-	136	contínuo
#2	Diastolic	-	47	contínuo
#3	Triceps	-	51	contínuo
#4	Two	-	186	contínuo
#5	Body	-	248	contínuo
#6	Diabetes	-	517	contínuo
#7	Age	-	52	contínuo

Tabela 6.2.1: Pima – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	37.8	12.2	24.4 ± 4.7	47.1 ± 7.0	32.4 ± 5.0	34.0 ± 4.2
$[max, min]$	12.8	14.1	$[35.7, 18.3]$	$[56.7, 28.6]$	$[39.0, 22.4]$	$[39.1, 23.9]$
Progol(C1)	35.6	14.4	28.7 ± 3.4	33.6 ± 8.3	30.6 ± 3.9	31.0 ± 4.8
$[max, min]$	9.1	17.8	$[35.2, 24.5]$	$[50.0, 17.6]$	$[39.0, 24.7]$	$[40.9, 21.7]$
Foil(C2)	14.4	12.5	45.9 ± 12.5	20.9 ± 7.0	30.1 ± 5.8	30.4 ± 6.3
$[max, min]$	10.6	39.4	$[75.0, 28.6]$	$[31.9, 11.4]$	$[37.7, 17.1]$	$[39.5, 19.1]$
Progol(C2)	12.4	14.5	53.1 ± 7.8	11.9 ± 4.6	26.4 ± 5.9	24.6 ± 4.7
$[max, min]$	5.8	44.2	$[64.3, 38.5]$	$[19.6, 6.7]$	$[35.1, 14.3]$	$[32.1, 16.6]$

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	42.8 ± 6.5	37.0 ± 8.4	3.0 ± 8.4
Progol(C1)	92.2 ± 17.1	24.6 ± 17.1	1163.9 ± 343.9
Foil(C2)	48.0 ± 6.3	30.6 ± 7.4	3.5 ± 7.4
Progol(C2)	103.2 ± 4.4	19.6 ± 4.7	1255.7 ± 58.4

(b)

Tabela 6.2.2: Médias para o Conjunto de Dados Pima e Método Prop, (C1 = zero e C2 = um)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	37.5	12.5	24.7 ± 4.8	38.7 ± 8.6	30.0 ± 3.0	30.6 ± 3.5
$[max, min]$	10.6	16.3	$[31.4, 15.6]$	$[51.5, 26.9]$	$[35.1, 24.7]$	$[36.2, 25.2]$
Progol(C1)	33.4	16.6	33.3 ± 5.0	29.5 ± 8.2	32.0 ± 4.4	31.1 ± 5.7
$[max, min]$	8.1	18.8	$[40.8, 26.4]$	$[39.3, 11.8]$	$[40.3, 27.3]$	$[40.0, 19.3]$
Foil(C2)	12.4	14.5	52.6 ± 11.3	13.9 ± 4.4	27.9 ± 5.8	26.4 ± 4.3
$[max, min]$	6.9	43.1	$[72.7, 35.3]$	$[21.4, 6.8]$	$[35.1, 16.9]$	$[31.3, 19.4]$
Progol(C2)	11.5	15.4	56.5 ± 11.1	9.3 ± 3.1	26.1 ± 6.4	22.8 ± 4.8
$[max, min]$	4.7	45.3	$[79.2, 41.2]$	$[15.6, 4.1]$	$[35.1, 16.9]$	$[31.2, 13.8]$

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	95.3 ± 3.9	60.3 ± 4.3	8832.7 ± 2090.0
Progol(C1)	119.9 ± 31.5	22.3 ± 33.3	1013.0 ± 503.1
Foil(C2)	159.7 ± 18.7	87.1 ± 20.1	48710.1 ± 40009.2
Progol(C2)	128.7 ± 2.4	10.3 ± 3.0	1144.9 ± 97.6

(b)

Tabela 6.2.3: Médias para o Conjunto de Dados Pima e Método MB, (C1 = zero e C2 = um)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−1.0	+1.8	+0.4	+0.7
C2	−0.7	+1.5	+0.6	+1.0

Tabela 6.2.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−1.8	+1.1	−0.5	−0.1
C2	−0.3	+1.2	+0.3	+0.8

Tabela 6.2.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop				Método MB			
Algoritmo	Matriz de Conf.	$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.	$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
Foil (C1)	490 10 1 268	0.02	0.00	0.01	500 0 0 269	0.00	0.00	0.00
Progol (C1)	500 0 47 222	0.00	0.17	0.06	500 0 58 211	0.00	0.22	0.08
Foil (C2)	237 32 2 498	0.12	0.00	0.04	269 0 0 500	0.00	0.00	0.00
Progol (C2)	167 102 24 476	0.38	0.05	0.16	269 0 24 476	0.00	0.05	0.03

Tabela 6.2.6: Erro Aparente para o Conjunto de Dados Pima (C1 = zero e C2 = um)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
Foil (C1)	46	36	3.7	125	75	7275.0
Progol (C1)	92	26	1100.04	107	12	746.15
Foil (C2)	61	29	3.9	175	96	57155.4
Progol (C2)	119	17	1596.21	143	14	1471.98

Tabela 6.2.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Pima (C1 = zero e C2 = um)

```
is_zero(A,B,C,D,E,F,G,H) :- B<=123, H<=2℘, F<=30.9, A<=2.
is_zero(A,B,C,D,E,F,G,H) :- B<=127, H<=2℘, G<=0.4℘3, G>0.27℘.
is_zero(A,B,C,D,E,F,G,H) :- B<=127, F>22.9, F<=26.3.
is_zero(A,B,C,D,E,F,G,H) :- B<=106, G<=0.292, C<=℘0, C>64.
is_zero(A,B,C,D,E,F,G,H) :- B<=143, H<=2℘, F<=2℘.7, A<=2.
is_zero(A,B,C,D,E,F,G,H) :- B<=94, D<=30, G<=0.767.
is_zero(A,B,C,D,E,F,G,H) :- B<=154, H<=30, C>72, E>70, F<=45.3, F>33.7.
is_zero(A,B,C,D,E,F,G,H) :- B<=123, D>H, H>31.
is_zero(A,B,C,D,E,F,G,H) :- H<=30, G<=0.191, G>0.15.
is_zero(A,B,C,D,E,F,G,H) :- B<=123, B>109, E>0, H<=2℘.
is_zero(A,B,C,D,E,F,G,H) :- B<=154, F<=27.℘, H>54, C>72.
is_zero(A,B,C,D,E,F,G,H) :- A<=6, B<=114, C<=60, C>50.
is_zero(A,B,C,D,E,F,G,H) :- B<=154, C>E, H<=30, G>0.543, H>23, B>℘0.
is_zero(A,B,C,D,E,F,G,H) :- B<=154, G<=0.527, F<=40.6, C>℘2, E<=0, F>2℘.4.
is_zero(A,B,C,D,E,F,G,H) :- F<=23.1, F>0, B>107.
is_zero(A,B,C,D,E,F,G,H) :- G<=0.527, G>0.403, F<=30.7, C>74.
is_zero(A,B,C,D,E,F,G,H) :- B<=159, C<=75, E>115, G<=0.33℘, H>33.
is_zero(A,B,C,D,E,F,G,H) :- B<=166, H<=24, E>250.
is_zero(A,B,C,D,E,F,G,H) :- B<=157, A<=7, G<=0.151, F<=2℘.9.
is_zero(A,B,C,D,E,F,G,H) :- D<=31, F>32.9, A>1, B<=129, C<=E, A<=10.
is_zero(A,B,C,D,E,F,G,H) :- C>E, H>53, C<=75, A>4.
is_zero(A,B,C,D,E,F,G,H) :- B<=157, A<=7, G>0.565, B>131, A>4, E>126.
is_zero(A,B,C,D,E,F,G,H) :- E<=120, F<=40.6, F>39.9.
is_zero(A,B,C,D,E,F,G,H) :- B<=14℘, H<=24, D>E, D<=32.
is_zero(A,B,C,D,E,F,G,H) :- B<=111, C>℘6, D<=E.
is_zero(A,B,C,D,E,F,G,H) :- F<=35.5, D<=1℘, F>32.9, A<=5, A>3.
is_zero(A,B,C,D,E,F,G,H) :- F<=32.4, C<=6℘, B<=125, C>66.
is_zero(A,B,C,D,E,F,G,H) :- F<=2℘.6, B>125, F>25.℘, C>E, G<=0.6℘7.
is_zero(A,B,C,D,E,F,G,H) :- G>0.371, B<=139, H<=42, C>62, G<=0.457, C<=℘0.
is_zero(A,B,C,D,E,F,G,H) :- G<=0.3℘, D>14, H>47, H<=50.
is_zero(A,B,C,D,E,F,G,H) :- G<=0.2, G>0.163, F>30.℘, D<=H.
is_zero(A,B,C,D,E,F,G,H) :- E>99, D<=16, D>14.
is_zero(A,B,C,D,E,F,G,H) :- H<=31, D>29, C<=74, C>6℘, E>135, F>H.
is_zero(A,B,C,D,D,F,G,H) :- H>41, C<=52.
is_zero(A,B,C,D,E,F,G,H) :- F>39.4, G>1.136, H>33.
```

Figura 6.2.1: Regras Geradas por FOIL: Conjunto de Dados Pima e Método Prop, com C1 = zero

```
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), B=<6, C=<123, G=<33.600.
example(A) :- is_zero(A,B,C,D,E,E,F,G,H), C=<15℘, F=<29.℘00,
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), C=<143, G=<26.200.
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), 35.300=<G, 60=<D, C=<93.
example(A) :- is_zero(A,0,B,C,D,E,F,G,H), 33.℘00=<F, B=<117.
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), 5℘=<I, C=<173, D=<7℘.
example(A) :- is_zero(A,B,C,D,E,105,F,G,H).
example(A) :- is_zero(A,1,B,C,D,E,F,G,H), H=<25.
example(A) :- is_zero(A,B,C,D,E,F,G,H,30), 74=<D.
example(A) :- is_zero(A,B,C,D,E,E,F,G,H), 10=<B, G=<0.200.
example(A) :- is_zero(A,B,C,70,D,E,F,G,H), 35.400=<F, C=<124.
example(A) :- is_zero(A,B,C,D,E,140,F,G,H).
example(A) :- is_zero(A,B,C,6℘,D,D,E,F,G).
example(A) :- is_zero(A,B,C,7℘,D,E,F,G,H), C=<96.
example(A) :- is_zero(A,B,C,D,E,E,F,G,H), 34.100=<F, C=<117,
example(A) :- is_zero(A,B,127,C,D,E,F,G,H).
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), 33.℘00=<G, C=<117,
example(A) :- is_zero(A,B,126,C,D,E,F,G,H), 27=<D.
example(A) :- is_zero(A,B,C,D,E,F,33.200,G,H).
example(A) :- is_zero(A,2,B,C,D,E,F,G,26).
example(A) :- is_zero(A,B,C,D,E,F,G,H,2℘), 33.100=<G, F=<166.
example(A) :- is_zero(A,B,129,C,D,D,E,F,G).
example(A) :- is_zero(A,B,C,D,E,E,F,G,H), C=<7℘.
example(A) :- is_zero(A,B,154,C,D,E,F,G,H).
example(A) :- is_zero(A,B,C,D,E,E,F,G,H), 1.441=<G.
example(A) :- is_zero(A,B,C,D,E,F,G,H,I), 37=<E, F=<0, H=<0.305.
```

Figura 6.2.2: Regras Geradas por PROGOL: Conjunto de Dados Pima e Método Prop, com C1 = zero

```
is_zero(A) :- a1(A,B), a5(A,C), a7(A,D), B<=123, D<=2°, a4(E,B), C<=2°.1.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), a7(A,F), C<=127, F<=2°, E<=30.9, a4(G,D), B<=6.
is_zero(A) :- a1(A,B), a2(A,C), B<=99, a4(D,C), a2(D,E), a4(e50,E).
is_zero(A) :- a1(A,B), a3(A,C), a4(D,C), B<=94, C<=1°.
is_zero(A) :- a1(A,B), a3(A,C), a7(A,D), C>D, a4(E,B), D<=23.
is_zero(A) :- a1(A,B), a2(A,C), a5(A,D), a7(A,E), B<=143, D<=27.3, C>64, E<=51.
is_zero(A) :- a0(A,B), a1(A,C), a4(A,D), a6(A,E), a7(A,F), E<=0.527, C<=12°,
              a5(G,F), D>36, C>D, B<=9.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), a6(A,F), F<=0.21°, C<=110, D<=°0, B<=E.
is_zero(A) :- a1(A,B), a4(A,C), a6(A,D), a7(A,E), E<=2°, C>135, D<=0.52°, B<=134.
is_zero(A) :- a1(A,B), a2(A,C), a5(A,D), a6(A,E), B<=154, D<=25.4, a4(F,C), E>0.277.
is_zero(A) :- a1(A,B), a2(A,C), a5(A,D), a7(A,E), B<=111,
              a4(F,C), a3(F,G), D>35.6, a5(H,E), a2(H,I), G>I.
is_zero(A) :- a1(A,B), a2(A,C), a5(A,D), a6(A,E), E<=0.4°°, C>72, D<=40.6, a2(F,B), a0(F,G), G>°.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), a7(A,F),
              C<=154, B<=6, D>72, F<=30, C>119, E<=41.5, a4(G,D).
is_zero(A) :- a0(A,B), a2(A,C), a4(A,D), a5(A,E), a6(A,F), C>D, B<=5, E<=31, F<=0.244.
is_zero(A) :- a0(A,B), a1(A,C), a3(A,D), C<=101, a3(E,B), a4(E,D).
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), a4(A,F), a5(A,G),
              C<=154, F<=140, B<=5, B>1, D<=72, E<=G, E>20.
is_zero(A) :- a0(A,B), a1(A,C), a5(A,D), a7(A,E), C<=154, E>54, D<=2°.°, B>4.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), B<=7,
              a1(F,D), a3(F,G), B>2, D>°2, C<=133, a5(H,G), E<=37.
is_zero(A) :- a0(A,B), a1(A,C), a3(A,D), C<=157, B<=6, C>149, D>27, B>0.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), a5(A,F), C<=129, F>31.1,
              a4(G,D), B>E, a2(G,H), a3(G,I), a4(J,H), I<=13.
is_zero(A) :- a1(A,B), a4(A,C), a7(A,D), B<=129, a7(E,C), D>33.
is_zero(A) :- a0(A,B), a1(A,C), a2(A,D), a3(A,E), a4(A,F), C<=143, B<=4, E>F, D>74, D<=°2.
is_zero(A) :- a0(A,B), a2(A,C), a3(A,D), a7(E,C), D<=21, a3(F,C), B<=D.
is_zero(A) :- a1(A,B), a3(A,C), a6(A,D), a7(A,E), C>E, D>0.°0°, a4(F,B), B<=122.
is_zero(A) :- a4(A,B), a6(A,C), C<=0.305, C>0.27°, B>°°.
is_zero(A) :- a3(A,B), a5(A,C), a7(A,D), a4(E,B), D>44, C>3°.5, B<=32.
is_zero(A) :- a1(A,B), a5(A,C), a6(A,D), a7(A,E), C<=36.1, D<=0.73°, D>0.37°,
              a2(F,B), a1(F,G), E>3°, G<=12°.
is_zero(A) :- a2(A,B), a3(A,C), a4(A,D), a7(A,E), E<=24, E>23, B>D, a4(F,C), a3(F,G), a5(H,G).
is_zero(A) :- a1(A,B), a5(A,C), a6(A,D), a7(A,E), C<=29.9, B>134, a2(F,E), D>0.17°.
is_zero(A) :- a1(A,B), a6(A,C), a7(A,D), C>1.394, D>25, B>137.
is_zero(A) :- a1(A,B), a5(A,C), a7(A,D), C<=26.1, C>D, B>139.
```

Figura 6.2.3: Regras Geradas por FOIL: Conjunto de Dados Pima e Método MB, com C1 = zero

```
example(A) :- a1(A,B), a7(A,C), B=<123, C=<34.
example(A) :- a7(A,B), 63=<B.
example(A) :- a1(A,B), a5(A,C), B=<143, C=<26.200.
example(A) :- a5(A,33.200).
example(A) :- a1(A,B), a4(A,C), B=<℘℘, C=<54.
example(A) :- a0(A,B), a5(A,C), B=<1, C=<2℘.700.
example(A) :- a4(A,140).
example(A) :- a1(A,106).
example(A) :- a1(A,127).
example(A) :- a1(A,B), a3(A,33), B=<142.
example(A) :- a5(A,B), a7(A,37), B=<32.400.
example(A) :- a1(A,129), a3(A,B), B=<20.
```

Figura 6.2.4: Regras Geradas por PROGOL: Conjunto de Dados Pima e Método MB, com C1 = zero

```
is_um(A,B,C,D,E,F,G,H) :- B>154, F>29.℘, E<=249, G>0.341, H<=43.
is_um(A,B,C,D,E,F,G,H) :- B>123, F>29.9, G>0.42, A>7, G<=1.001.
is_um(A,B,C,D,E,F,G,H) :- B>166, E>130, G<=0.5℘6.
is_um(A,B,C,D,E,F,G,H) :- F>29.9, B>155, A>E, B<=17℘.
is_um(A,B,C,D,E,F,G,H) :- B>114, H>30, F>40.℘, F>H.
is_um(A,B,C,D,E,F,G,H) :- B>111, F>27.℘, H>41, G>0.21℘, D<=29, F<=33.3.
is_um(A,B,C,D,E,F,G,H) :- F>29.9, C<=72, A<=1, H>23, B>119.
is_um(A,B,C,D,E,F,G,H) :- B>99, G>0.252, F>26.7, B<=119, A>6, H<=3℘.
is_um(A,B,C,D,E,F,G,H) :- B>123, H>24, C<=72, C>6℘, G>0.255, D<=25.
is_um(A,B,C,D,E,F,G,H) :- B>114, H>31, F>33.2, F<=37.1, A<=3, C>0.
is_um(A,B,C,D,E,F,G,H) :- B>123, F<=H, H<=54, F<=2℘.4, G>0.207, C>74.
is_um(A,B,C,D,E,F,G,H) :- F>30.℘, G>0.619, A>6, H>42.
is_um(A,B,C,D,E,F,G,H) :- B>12℘, F>34.2, F<=40, H>42.
is_um(A,B,C,D,E,F,G,F) :- B>103.
is_um(A,B,C,D,E,F,G,H) :- G>0.705, A>2, C<=℘2, F>36.7, F<=43.6.
is_um(A,B,C,D,E,F,G,H) :- B>103, F>29.℘, F<=34, G>0.493, D<=H, A>0, B<=14℘.
is_um(A,B,C,D,E,F,G,H) :- B>166, D>H, B<=193.
is_um(A,B,C,D,E,F,G,H) :- B>101, E<=5℘, F>31.2, F<=34, B<=131, D>2℘.
is_um(A,B,C,D,D,F,G,H) :- B>101, F>27.℘, H<=23, H>21.
is_um(A,B,C,D,E,F,G,H) :- H>24, B>141, F<=2℘.4, H<=36.
is_um(A,B,C,D,E,F,G,H) :- F>29.6, C<=66, A>3, B>106, H>29, D<=33.
is_um(A,B,C,D,E,F,G,H) :- F>45.3, H<=29, E<=130.
is_um(A,B,C,D,E,F,G,H) :- H>2℘, D<=27, D>21, E>112, B<=125.
is_um(A,B,C,D,E,F,G,H) :- B>137, G<=0.17℘, A>2, D<=27.
is_um(A,B,C,D,E,F,G,H) :- F>30.9, C<=66, D>F, F<=34.℘, A<=4, C>5℘.
is_um(A,B,C,D,B,F,G,H) :- H>24.
is_um(A,B,C,D,E,F,G,H) :- H>24, F>25.3, G>0.709, D<=22, F<=32.9, G<=1.224.
is_um(A,B,C,D,D,F,G,H) :- G<=0.1℘, A<=6, F>31.6.
```

Figura 6.2.5: Regras Geradas por FOIL: Conjunto de Dados Pima e Método Prop, com C2 = um

```
example(A) :- is_um(A,B,C,D,E,F,G,H,I), 0.371=<H, 151=<C, D=<90.
example(A) :- is_um(A,B,C,D,E,F,G,H,I), 37=<E, 46.200=<G, ℘0=<D.
example(A) :- is_um(A,4,B,C,D,D,E,F,G), 0.277=<F, 72=<C.
example(A) :- is_um(A,B,C,D,E,F,G,H,I), 7=<B, 43=<I, 125=<F.
example(A) :- is_um(A,B,C,D,E,E,F,G,H), 32.400=<F, 10℘=<C, D=<66.
example(A) :- is_um(A,B,C,D,E,E,F,G,H), 164=<C, B=<9, D=<7℘.
example(A) :- is_um(A,9,B,C,D,D,E,F,G), 70=<C.
example(A) :- is_um(A,B,C,D,E,F,G,H,I), 29.900=<G, 169=<C, H=<0.26℘.
example(A) :- is_um(A,B,C,D,E,F,G,H,37), 36.500=<G.
example(A) :- is_um(A,B,C,D,E,F,31.600,G,H), 25=<H, 112=<C.
example(A) :- is_um(A,B,C,D,E,F,G,H,I), 32.000=<G, 9℘=<D, 12℘=<C.
example(A) :- is_um(A,B,C,D,36,E,F,G,H), 41.200=<F.
example(A) :- is_um(A,B,109,C,D,E,F,G,H), 31=<D.
example(A) :- is_um(A,B,C,D,D,D,30.000,E,F).
example(A) :- is_um(A,B,125,C,D,E,F,G,H), 41=<H.
example(A) :- is_um(A,B,C,D,32,E,F,G,H), 7=<B.
example(A) :- is_um(A,B,C,D,E,F,32.900,G,H).
```

Figura 6.2.6: Regras Geradas por PROGOL: Conjunto de Dados Pima e Método Prop, com C2 = um

```
is_um(A) :- a1(A,B), a2(A,C), a4(A,C), B>130.
is_um(A) :- a1(A,B), a3(A,C), a5(A,D), B>154, D>29.℘, a4(E,C), a7(E,F), B<=17℘, F>27.
is_um(A) :- a1(A,B), a4(A,C), a6(A,D), a7(A,E), B>154, D>0.341, C<=32℘, E<=36.
is_um(A) :- a1(A,B), a5(A,C), a7(A,D), D>30, a4(E,B), a5(E,F), C>3℘.9, F<=25.1.
is_um(A) :- a1(A,B), a5(A,C), a6(A,D), a7(A,E), B>123, E>30, C>30, D>0.391, B<=132.
is_um(A) :- a1(A,B), a5(A,C), a7(A,D), C>29.9, D>37, B>1℘3.
is_um(A) :- a0(A,B), a4(A,C), a6(A,D), C>112, B>7, D>0.2℘2, B<=12.
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), a7(A,F), C>111, F>2℘, a7(G,D), E>30, B<=1.
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), a6(A,F), a7(A,G),
            C>106, B>2, E>30.1, G>40, F>0.251, D>6℘, E<=35.9.
is_um(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), C>106, B>2, E>30.1, a7(F,D), a3(G,D).
is_um(A) :- a0(A,B), a1(A,C), a6(A,D), B>2, C>159, D<=0.26℘, B<=6.
is_um(A) :- a1(A,B), a2(A,C), a4(A,D), a5(A,E), a7(A,F), B>106, E>2℘.℘, a2(G,D), a7(H,C),
            a0(H,I), E<=F, I<=2.
is_um(A) :- a1(A,B), a3(A,C), a5(A,D), a7(A,E), B>101, E>2℘, a7(F,C), C<=24, D>26.2.
is_um(A) :- a1(A,B), a5(A,C), B>123, C>41.5, B<=142.
is_um(A) :- a1(A,B), a2(A,C), a5(A,D), a7(A,E), B>101, E>27, D>43.3, C>7℘.
is_um(A) :- a1(A,B), a6(A,C), a7(A,D), B>123, D>24, B<=125, C>0.464.
is_um(A) :- a1(A,B), a2(A,C), a5(A,D), D>2℘.℘, a4(E,C), a1(E,B), a2(G,D).
is_um(A) :- a1(A,B), a7(A,C), B>101, C<=43, C>42.
is_um(A) :- a5(A,B), a7(A,B), B>2℘.℘.
is_um(A) :- a2(A,B), a4(A,C), a7(A,D), B>6℘, C>112, D>40, a1(E,C), a6(E,F), F<=0.26℘.
```

Figura 6.2.7: Regras Geradas por FOIL: Conjunto de Dados Pima e Método MB, com C2 = um

```
example(A) :- a1(A,B), a6(A,C), 0.371=<C, 151=<B.
example(A) :- a7(A,43).
example(A) :- a7(A,52).
example(A) :- a1(A,B), a6(A,C), 190=<B, C=<0.27℘.
example(A) :- a1(A,129), a3(A,B), 29=<B.
example(A) :- a5(A,42.300).
example(A) :- a1(A,B), a6(A,C), 169=<B, C=<0.26℘.
example(A) :- a1(A,145).
example(A) :- a1(A,12℘), a6(A,B), 0.549=<B.
example(A) :- a2(A,B), a5(A,C), 49.300=<C, B=<94.
example(A) :- a5(A,32.900).
example(A) :- a0(A,9), a2(A,B), ℘0=<B.
example(A) :- a4(A,B), a5(A,C), 45.600=<C, B=<0.
example(A) :- a1(A,B), a2(A,C), 137=<B, C=<40.
```

Figura 6.2.8: Regras Geradas por PROGOL: Conjunto de Dados Pima e Método MB, com C2 = um

6.3 Conjunto de Dados Breast

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	Age	-	44	contínuo
#1	Age-at-meno	-	3	discreto
#2	Tumor-size	-	23	contínuo
#3	Involved-nodes	-	18	contínuo
#4	Node-capsule	3	3	discreto
#5	Degree-of-malig	-	3	contínuo
#6	Breast	-	2	discreto
#7	Breast-Quadrant	6	6	discreto
#8	Irradiation	-	2	discreto

Tabela 6.3.1: Breast – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	4.0	4.4	52.1 ± 9.0	21.0 ± 9.3	30.2 ± 7.4	32.1 ± 8.6
$[max, min]$	4.2	15.9	[71.4, 42.9]	[38.9, 4.8]	[46.4, 17.2]	[48.3, 15.5]
Progol(C1)	2.1	6.3	75.9 ± 13.9	11.7 ± 2.6	30.7 ± 4.8	30.0 ± 4.5
$[max, min]$	2.4	17.7	[100.0, 50.0]	[16.7, 9.1]	[37.9, 24.1]	[37.8, 22.9]
Foil(C2)	16.4	3.7	18.3 ± 7.8	49.1 ± 19.2	27.3 ± 10.3	29.7 ± 11.7
$[max, min]$	4.1	4.3	[31.8, 9.5]	[85.7, 25.0]	[46.4, 13.8]	[52.2, 15.4]
Progol(C2)	16.1	4.0	19.9 ± 10.9	54.5 ± 19.6	30.0 ± 8.7	31.0 ± 9.3
$[max, min]$	4.5	3.9	[40.9, 4.8]	[85.7, 25.0]	[44.8, 13.8]	[48.4, 13.4]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	26.8 ± 3.4	16.5 ± 4.3	0.7 ± 4.3
Progol(C1)	44.3 ± 3.5	6.4 ± 3.8	106.7 ± 15.5
Foil(C2)	28.0 ± 4.2	24.4 ± 4.2	0.7 ± 4.2
Progol(C2)	29.3 ± 2.8	8.3 ± 3.4	52.9 ± 3.4

(b)

Tabela 6.3.2: Médias para o Conjunto de Dados Breast e Método Prop, (C1 = recurrence e C2 = no_recurrence)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	3.0	5.4	65.2 ± 11.7	14.8 ± 9.8	29.5 ± 7.8	29.4 ± 11.8
$[max, min]$	3.0	17.1	$[87.5, 55.6]$	$[33.3, 0.0]$	$[42.9, 17.2]$	$[44.7, 7.9]$
Progol(C1)	1.2	7.2	85.8 ± 9.2	7.8 ± 3.8	30.9 ± 5.2	25.6 ± 7.6
$[max, min]$	1.6	18.5	$[100.0, 71.4]$	$[14.3, 0.0]$	$[37.9, 21.4]$	$[37.8, 9.3]$
Foil(C2)	14.7	5.4	26.7 ± 8.5	41.7 ± 19.5	31.6 ± 9.8	32.6 ± 10.4
$[max, min]$	3.6	4.8	$[40.9, 16.7]$	$[77.8, 14.3]$	$[50.0, 17.9]$	$[53.5, 16.5]$
Progol(C2)	14.4	5.7	28.2 ± 8.5	54.1 ± 21.9	35.8 ± 6.1	37.1 ± 8.3
$[max, min]$	4.5	3.9	$[38.1, 9.5]$	$[85.7, 22.2]$	$[44.8, 24.1]$	$[51.0, 24.4]$

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	60.8 ± 0.8	35.8 ± 1.2	50.9 ± 6.8
Progol(C1)	55.5 ± 7.9	4.4 ± 8.0	156.1 ± 34.0
Foil(C2)	55.2 ± 0.2	33.9 ± 1.2	90.8 ± 38.1
Progol(C2)	38.2 ± 3.0	7.3 ± 3.8	71.8 ± 3.9

(b)

Tabela 6.3.3: Médias para o Conjunto de Dados Breast e Método MB, (C1 = recurrence e C2 = no_recurrence)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−2.0	+1.4	−0.1	+0.3
C2	−0.2	−0.3	−0.3	−0.1

Tabela 6.3.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−2.0	+0.9	−0.2	+0.4
C2	−0.2	−0.6	−0.5	−0.5

Tabela 6.3.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop					Método MB				
Algoritmo	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
FOIL (C1)	76	8	0.10	0.00	0.03	84	0	0.00	0.00	0.00
	0	201				0	201			
PROGOL (C1)	84	0	0.00	0.02	0.01	84	0	0.00	0.02	0.02
	4	197				5	196			
FOIL (C2)	201	0	0.00	0.00	0.00	201	0	0.00	0.00	0.00
	0	84				0	84			
PROGOL (C2)	201	0	0.00	0.24	0.07	201	0	0.00	0.00	0.00
	20	64				0	84			

Tabela 6.3.6: Erro Aparente para o Conj. de Dados Breast (C1 = recurrence e C2 = no_recurrence)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
FOIL (C1)	27	19	0.7	52	33	35.3
PROGOL (C1)	48	7	122.57	55	6	167.86
FOIL (C2)	31	31	0.9	64	39	63.9
PROGOL (C2)	34	11	63.97	123	13	403.88

Tabela 6.3.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conj. de Dados Breast (C1 = recurrence e C2 = no_recurrence)

```
is_recurrence(A,B,C,D,E,F,brleft,H,I) :- D>0, F>2, A<=52, A>33.
is_recurrence(A,B,C,D,E,F,brleft,H,E) :- D>0, F>2, A>55.
is_recurrence(A,B,C,D,E,D,G,bqleft_lower,I) :- C>15, D>1.
is_recurrence(A,B,C,D,E,F,brright,H,I) :- F>2, A>65.
is_recurrence(A,ampremenopaus,C,D,E,F,G,H,booleanno) :- D>0, F>2, C>20.
is_recurrence(A,B,C,D,E,F,G,H,I) :- D>0, A>52, A<=53.
is_recurrence(A,amleq40,C,D,E,F,brright,bqright_upper,I) :- D>0, D<=4.
is_recurrence(A,ampremenopaus,C,D,booleanno,F,G,bqleft_upper,I) :- A<=36, D>0.
is_recurrence(A,B,C,D,E,F,brleft,H,I) :- C<=25, D>3.
is_recurrence(A,ampremenopaus,C,D,E,F,G,H,I) :- C>27, A>46, D>3.
is_recurrence(A,ampremenopaus,C,D,E,F,brleft,bqright_upper,I) :- F>1.
is_recurrence(A,B,C,D,E,F,G,bqcentral,E) :- A>56, C>15.
is_recurrence(A,B,C,D,E,F,G,H,I) :- C>30, A>54, C<=35.
is_recurrence(A,B,C,D,E,F,G,H,I) :- A<=31, A>29.
is_recurrence(A,ampremenopaus,C,D,E,F,G,H,I) :- A>43, D<=F, C>30, F<=1.
is_recurrence(A,B,C,D,E,F,G,H,I) :- F>2, A>55, A<=60.
is_recurrence(A,B,C,D,booleanno,F,brright,bqleft_upper,I) :- D<=1, D>0, C>15.
is_recurrence(A,B,C,D,E,F,G,H,booleanyes) :- C<=22, C>15, A>44.
```

Figura 6.3.1: Regras Geradas por FOIL: Conjunto de Dados Breast e Método Prop, com C1 = recurrence

```
example(A) :- is_recurrence(A,B,C,D,E,F,E,G,bqleft_lower,H), 2=<E.
example(A) :- is_recurrence(A,B,C,D,E,booleanyes,F,brleft,G,H), 3=<F.
example(A) :- is_recurrence(A,B,C,D,E,F,G,H,I,J), 3=<E, 66=<B.
example(A) :- is_recurrence(A,B,C,D,E,F,G,H,I,F), 1=<E, 2º=<D, B=<43.
example(A) :- is_recurrence(A,B,C,D,4,E,F,G,H,I).
example(A) :- is_recurrence(A,B,C,20,D,E,D,F,G,H).
example(A) :- is_recurrence(A,31,B,C,D,E,F,G,H,I).
```

Figura 6.3.2: Regras Geradas por PROGOL: Conjunto de Dados Breast e Método Prop, com C1 = recurrence

```
is_recurrence(A) :- a6(A,brleft), a0(A,C), a3(A,D), a5(A,E), D>0, E>2, C<=52, C>33.
is_recurrence(A) :- a4(A,B), a6(A,brleft), aº(A,B), a0(A,D), a3(A,E), a5(A,F), E>0,
F>2, D>55.
is_recurrence(A) :- a7(A,bqleft_lower), a2(A,C), a3(A,D), a5(A,D), C>15, D>1.
is_recurrence(A) :- a6(A,brright), a0(A,C), a5(A,D), D>2, C>65.
is_recurrence(A) :- aº(A,booleanno), a1(A,ampremenopaus), a2(A,D), a3(A,E), a5(A,F),
E>0, F>2, D>20.
is_recurrence(A) :- a0(A,B), a3(A,C), C>0, B>52, B<=53.
is_recurrence(A) :- a6(A,brright), a1(A,amleq40), a7(A,bqright_upper), a3(A,E),
E>0, E<=4.
is_recurrence(A) :- a4(A,booleanno), a1(A,ampremenopaus), a7(A,bqleft_upper),
a0(A,E), a3(A,F), E<=36, F>0.
is_recurrence(A) :- a6(A,brleft), a2(A,C), a3(A,D), C<=25, D>3.
is_recurrence(A) :- a0(A,B), a5(A,C), C>2, B>5º, B<=60.
is_recurrence(A) :- a6(A,brleft), a1(A,ampremenopaus), a7(A,bqright_upper),
a5(A,E), E>1.
is_recurrence(A) :- a0(A,B), a2(A,C), a5(A,D), B>56, B<=5º, D<=2, C>15, D>1.
is_recurrence(A) :- a0(A,B), B<=31, B>29.
```

Figura 6.3.3: Regras Geradas por FOIL: Conjunto de Dados Breast e Método MB, com C1 = recurrence

```
example(A) :- a0(A,B), a3(A,2), B=<43.
example(A) :- a3(A,6).
example(A) :- a3(A,B), a6(A,brleft), 11=<B.
example(A) :- a3(A,B), a5(A,B), 3=<B.
example(A) :- a3(A,4).
example(A) :- a0(A,31).
```

Figura 6.3.4: Regras Geradas por PROGOL: Conjunto de Dados Breast e Método MB, com C1 = recurrence

```
is_no_recurrence(A,B,C,D,booleanno,F,G,H,I) :- A>46, C<=1℘.
is_no_recurrence(A,B,C,D,E,F,G,H,I) :- D<=0, F<=1, A>51.
is_no_recurrence(A,B,C,D,E,F,G,H,I) :- D<=0, A<=43, A>31, F<=2.
is_no_recurrence(A,B,C,D,E,F,G,H,I) :- F<=2, A>44, A<=56, F>1, D<=2, C>25.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_upper,I) :- D<=0, F>2.
is_no_recurrence(A,amleq40,C,D,E,F,G,H,I) :- F<=2, A<=56, A>46.
is_no_recurrence(A,B,C,D,E,F,G,H,I) :- F<=2, A>36, A<=41.
is_no_recurrence(A,ampremenopaus,C,D,E,F,G,H,E) :- A>44, A<=50, C<=25, C>22.
is_no_recurrence(A,ampremenopaus,C,D,booleanno,F,G,H,I) :- A>44, C<=20.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_lower,I) :- D<=0, F>2.
is_no_recurrence(A,B,C,D,E,F,G,H,I) :- F<=2, A>5℘, D<=0, C<=30.
is_no_recurrence(A,amleq40,C,D,E,F,G,H,I) :- F<=2, D>F, A<=63.
is_no_recurrence(A,B,C,D,booleanyes,F,brright,H,booleanyes) :- A<=46.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_lower,I) :- A>43, C>24, F<=2, C<=30.
is_no_recurrence(A,B,C,D,booleanno,F,G,H,I) :- A<=55, A>53.
is_no_recurrence(A,B,C,D,E,F,brright,H,booleanyes) :- A<=46, C<=27.
is_no_recurrence(A,B,C,D,E,F,G,H,booleanno) :- D>1, A>49, D<=3.
is_no_recurrence(A,B,C,D,booleanno,F,brleft,H,booleanno) :- C<=35, F<=1.
is_no_recurrence(A,B,C,D,E,F,G,H,booleanno) :- D>6, A>54.
is_no_recurrence(A,B,C,D,E,F,G,bqright_upper,I) :- A<=35.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_lower,booleanno) :- C>40.
is_no_recurrence(A,B,C,D,E,F,G,bqright_upper,I) :- A<=51, F>2, A>47.
is_no_recurrence(A,amleq40,C,D,E,F,G,bqleft_upper,I) :- D>4, C>13.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_lower,I) :- C<=10.
is_no_recurrence(A,ampremenopaus,C,D,booleanno,F,G,H,booleanno) :- C>2℘, A>40, A<=46.
is_no_recurrence(A,B,C,D,E,F,G,bqleft_lower,I) :- A>56, A<=63, C<=30.
```

Figura 6.3.5: Regras Geradas por FOIL: Conjunto de Dados Breast e Método Prop, com C2 = no_recurrence

```
example(A) :- is_no_recurrence(A,B,C,D,E,F,G,H,I,F), D=<10.
example(A) :- is_no_recurrence(A,B,C,D,E,F,G,H,I,F), E=<0, G=<2.
example(A) :- is_no_recurrence(A,B,C,D,1,E,2,F,G,H).
example(A) :- is_no_recurrence(A,B,amleq40,C,D,E,F,G,H,E), 3=<F, D=<0.
example(A) :- is_no_recurrence(A,4℘,B,C,D,E,F,G,H,E).
example(A) :- is_no_recurrence(A,B,C,D,2,E,3,F,G,E), 33=<B.
example(A) :- is_no_recurrence(A,B,C,D,E,booleanyes,F,G,bqleft_upper,H), F=<2.
example(A) :- is_no_recurrence(A,55,B,C,D,E,F,G,H,I).
example(A) :- is_no_recurrence(A,B,C,D,3,E,2,F,G,H), B=<62.
example(A) :- is_no_recurrence(A,B,C,D,E,F,G,H,I,J), 2=<E, G=<1.
example(A) :- is_no_recurrence(A,B,C,D,7,E,F,G,H,I).
```

Figura 6.3.6: Regras Geradas por PROGOL: Conjunto de Dados Breast e Método Prop, com C2 = no_recurrence

```
is_no_recurrence(A) :- a4(A,booleanno), a0(A,C), a2(A,D), C>46, D<=1ª.
is_no_recurrence(A) :- a0(A,B), a3(A,C), a5(A,D), C<=0, D<=1, B>51.
is_no_recurrence(A) :- a0(A,B), a3(A,C), a5(A,D), C<=0, B<=43, B>31, D<=2.
is_no_recurrence(A) :- a4(A,booleanno), a0(A,C), a3(A,D), a5(A,E), E<=2, C>44,
C<=56, a5(F,D).
is_no_recurrence(A) :- a7(A,bqleft_upper), a3(A,C), a5(A,D), C<=0, D>2.
is_no_recurrence(A) :- a0(A,B), a2(A,C), a3(A,D), a5(A,E), E<=2, E>1, D<=1, C>22, B<=50.
is_no_recurrence(A) :- a6(A,brright), a7(A,bqleft_upper), a0(A,D), a3(A,E), a5(A,F),
F<=2, F>1, D>47, E<=5.
is_no_recurrence(A) :- a7(A,bqleft_lower), a0(A,C), a2(A,D), a3(A,E), E<=0, C>47, D<=30.
is_no_recurrence(A) :- a4(A,booleanyes), a2(A,C), a5(A,D), D<=2, a3(E,C).
is_no_recurrence(A) :- a6(A,brright), a0(A,C), a3(A,D), a5(A,E), E<=2, D>2, C<=46, C>35.
is_no_recurrence(A) :- a4(A,booleanno), aª(A,booleanno), a7(A,bqright_lower).
is_no_recurrence(A) :- a4(A,B), aª(A,B), a0(A,C), a2(A,D), a5(A,E), C>43, C<=51,
E>1, D>30.
is_no_recurrence(A) :- a7(A,bqleft_lower), a0(A,C), a3(A,D), a5(A,E), E<=2, C>45,
C<=63, D>1.
is_no_recurrence(A) :- a4(A,booleanno), a1(A,ampremenopaus), a0(A,D), a2(A,E), E<=20,
D>37, E>15.
is_no_recurrence(A) :- a4(A,booleanno), a2(A,C), a3(A,D), C<=15, D>0.
is_no_recurrence(A) :- aª(A,booleanno), a1(A,amleq40), a0(A,D), a3(A,E), E>6, D>52.
is_no_recurrence(A) :- a0(A,B), a2(A,C), a3(A,D), D<=0, C>25, B<=46.
is_no_recurrence(A) :- a6(A,brleft), a0(A,C), a3(A,D), D<=0, C>59.
```

Figura 6.3.7: Regras Geradas por FOIL: Conjunto de Dados Breast e Método MB, com C2 = no_recurrence

```
example(A) :- a2(A,10).
example(A) :- a0(A,4ª).
example(A) :- a0(A,33).
example(A) :- a0(A,55).
example(A) :- a2(A,12).
example(A) :- a0(A,50).
example(A) :- a5(A,B), a7(A,bqcentral), B=<1.
example(A) :- a5(A,B), a7(A,bqright_lower), B=<1.
example(A) :- a0(A,62), a3(A,B), B=<0.
example(A) :- a3(A,B), a7(A,bqright_lower), B=<0.
example(A) :- a0(A,54), a3(A,B), B=<1.
example(A) :- a0(A,39).
example(A) :- a0(A,45), a2(A,B), 35=<B.
```

Figura 6.3.8: Regras Geradas por PROGOL: Conjunto de Dados Breast e Método MB, com C2 = no_recurrence

6.4 Conjunto de Dados Hungaria

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	age	-	38	contínuo
#1	sex	-	2	contínuo
#2	cp	-	4	contínuo
#3	trestbps	-	31	contínuo
#4	chol	-	153	contínuo
#5	fbs	-	2	contínuo
#6	restecg	-	3	contínuo
#7	thalach	-	71	contínuo
#8	exang	-	2	contínuo
#9	oldpeak	-	10	contínuo
#10	slope	-	3	contínuo
#11	ca	-	2	contínuo
#12	thal	-	3	contínuo

Tabela 6.4.1: Hungaria – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	7.2	3.4	34.7 ± 23.1	10.9 ± 7.9	19.3 ± 9.9	17.7 ± 12.2
$[max, min]$	2.2	16.6	[75.0, 0.0]	[32.0, 5.0]	[37.9, 10.0]	[49.0, 4.2]
Progol(C1)	5.6	5.0	48.9 ± 14.7	4.3 ± 4.9	19.8 ± 6.9	13.8 ± 7.3
$[max, min]$	0.8	18.0	[75.0, 33.3]	[16.7, 0.0]	[31.0, 10.3]	[30.2, 5.8]
Foil(C2)	15.1	3.7	20.0 ± 12.4	33.5 ± 16.3	24.8 ± 9.5	24.1 ± 9.3
$[max, min]$	3.6	7.0	[47.1, 5.0]	[54.5, 11.1]	[36.7, 6.9]	[37.8, 7.5]
Progol(C2)	12.8	6.0	31.6 ± 11.4	22.2 ± 12.3	28.4 ± 9.6	25.4 ± 9.9
$[max, min]$	2.3	8.3	[47.4, 13.3]	[45.5, 6.7]	[44.8, 10.0]	[44.9, 9.4]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	20.9 ± 6.2	13.5 ± 6.4	1.2 ± 6.4
Progol(C1)	46.4 ± 3.8	4.0 ± 4.3	57.5 ± 16.0
Foil(C2)	19.5 ± 1.6	17.1 ± 4.4	1.4 ± 4.4
Progol(C2)	45.5 ± 3.7	13.2 ± 3.8	20.6 ± 4.2

(b)

Tabela 6.4.2: Médias para o Conjunto de Dados Hungaria e Método Prop, (C1 = presence e C2 = absence)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	6.3	4.3	40.0 ± 18.0	11.5 ± 9.4	22.2 ± 6.5	18.0 ± 6.0
$[max, min]$	2.2	16.6	[66.7, 7.7]	[35.3, 0.0]	[33.3, 13.3]	[28.3, 8.2]
Progol(C1)	5.8	4.8	45.9 ± 16.7	8.0 ± 5.4	21.6 ± 7.4	17.9 ± 7.4
$[max, min]$	1.5	17.3	[75.0, 22.2]	[16.7, 0.0]	[31.0, 6.9]	[30.2, 4.7]
Foil(C2)	15.3	3.5	18.1 ± 8.9	24.0 ± 14.5	20.1 ± 6.3	19.0 ± 8.8
$[max, min]$	2.4	8.2	[29.4, 0.0]	[50.0, 7.7]	[31.0, 10.0]	[34.6, 5.2]
Progol(C2)	13.3	5.5	29.5 ± 10.1	18.5 ± 13.6	25.5 ± 6.0	21.1 ± 7.8
$[max, min]$	2.0	8.6	[46.7, 16.7]	[45.5, 0.0]	[34.5, 13.3]	[33.0, 4.2]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	44.3 ± 7.1	26.5 ± 7.5	235.6 ± 99.4
Progol(C1)	37.9 ± 9.6	2.5 ± 10.3	84.2 ± 43.4
Foil(C2)	41.9 ± 0.1	26.6 ± 0.4	9927.6 ± 8469.8
Progol(C2)	44.3 ± 12.3	4.5 ± 12.3	69.9 ± 39.9

(b)

Tabela 6.4.3: Médias para o Conjunto de Dados Hungaria e Método MB, (C1 = presence e C2 = absence)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	–0.7	+1.0	–0.1	+0.4
C2	–1.0	+0.8	–0.4	–0.1

Tabela 6.4.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	–0.3	+0.5	+0.1	0.0
C2	–1.2	+0.4	–0.9	–0.3

Tabela 6.4.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop					Método MB				
Algoritmo	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
Foil (C1)	104	2	0.02	0.01	0.01	106	0	0.00	0.00	0.00
	1	187				0	188			
Progol (C1)	106	0	0.00	0.04	0.03	106	0	0.00	0.06	0.04
	8	180				11	177			
Foil (C2)	184	4	0.02	0.00	0.01	188	0	0.00	0.00	0.00
	0	106				0	106			
Progol (C2)	188	0	0.00	0.11	0.04	188	0	0.00	0.09	0.03
	12	94				10	96			

Tabela 6.4.6: Erro Aparente para o Conjunto de Dados Hungaria (C1 = presence e C2 = absence)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
Foil (C1)	20	18	2.1	46	27	433.9
Progol (C1)	52	6	74.08	42	3	105.52
Foil (C2)	22	18	2.0	41	27	2399.0
Progol (C2)	50	16	24.68	63	4	103.06

Tabela 6.4.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Hungaria (C1 = presence e C2 = absence)

```
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- B>0, D>124, J>1.
is_presence(A,B,C,D,E,F,G,H,B,B,K,L,M) :- C>3, B>0, D>110.
is_presence(A,B,C,D,E,F,F,H,I,J,K,L,M) :- C>3, D<=12ª, I>0, A<=49.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- C>3, B>G, D<=125, H<=130.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- M>3, E>276.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- C>2, A>46, H>140, D>140.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- C>2, A>53, F>0.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- C>2, A>46, H>140, A<=49, E>1ª7.
is_presence(A,B,C,D,E,F,F,H,I,F,K,L,M) :- C>3, H<=125, H>110.
is_presence(A,B,B,D,E,F,G,H,I,J,K,L,M) :- D>120.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- K>1, H<=10ª, C>1.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- E>46ª.
is_presence(A,B,C,D,E,F,G,H,I,J,C,L,M) :- E>216, A>54.
is_presence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- D<=125, B>F, A<=3ª, H<=170, A>34.
is_presence(A,B,C,D,E,F,B,H,I,J,K,L,M) :- M>3.
is_presence(A,B,C,D,E,F,F,H,B,J,K,L,M) :- M>6.
```

Figura 6.4.1: Regras Geradas por Foil: Conjunto de Dados Hungarian e Método Prop, com C1 = presence

```
example(A) :- is_presence(A,B,C,D,E,F,G,G,H,G,I,I,J,J), 1=<C.
example(A) :- is_presence(A,B,C,D,E,F,G,G,H,C,C,I,J,J), 1=<C.
example(A) :- is_presence(A,B,C,D,E,F,G,G,H,C,I,J,K,K), 1=<C,
example(A) :- is_presence(A,B,C,D,E,F,G,G,H,C,I,I,J,K), 1=<C.
example(A) :- is_presence(A,B,C,D,E,F,C,G,H,C,I,J,K,K), 1=<C.
example(A) :- is_presence(A,B,C,D,E,F,G,C,H,C,I,J,K,K), 1=<C.
```

Figura 6.4.2: Regras Geradas por Progol: Conjunto de Dados Hungarian e Método Prop, com C1 = presence

```
is_presence(A) :- a0(A,B), a1(A,C), a3(A,D), a9(A,E), a10(A,F), F>1, a2(G,E), C>0, D>124, B<=5℘.
is_presence(A) :- a0(A,B), a1(A,C), a2(A,D), a5(A,E), a6(A,F), a9(A,G), D>3, G>0.℘, C>F, B>E.
is_presence(A) :- a2(A,B), a7(A,C), a℘(A,D), B>3, D>0, C>140.
is_presence(A) :- a7(A,B), a9(A,C), B<=130, C>1, C<=2.5.
is_presence(A) :- a2(A,B), a3(A,C), a6(A,D), a7(A,E), a9(A,D), B>3, E<=130, C<=120.
is_presence(A) :- a0(A,B), a1(A,C), a2(A,D), a4(A,E), a7(A,F), D>3, C>0, F>14℘, a3(G,F),
                  E<=2℘9, B<=52.
is_presence(A) :- a3(A,B), a7(A,C), a℘(A,D), a9(A,E), D>0, C<=119, B>125, a1(F,E).
is_presence(A) :- a2(A,B), a3(A,C), a4(A,D), a7(A,C), B>2, D>147.
is_presence(A) :- a1(A,B), a2(A,B), a3(A,D), D>120.
```

Figura 6.4.3: Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método MB, com C1 = presence

```
example(A) :- a9(A,B), a10(A,B), 2=<B.
example(A) :- a1(A,B), a℘(A,B), a10(A,C), 2=<C.
example(A) :- a4(A,B), 491=<B.
```

Figura 6.4.4: Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método MB, com C1 = presence

```
is_absence(A,B,C,D,E,F,G,H,I,G,K,L,M) :- C<=3, E<=246, C>1.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- I<=0, C<=2, C>1, D>120.
is_absence(A,B,C,D,E,F,G,H,B,F,K,L,M) :- B<=0.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- I<=0, C<=3, D<=130, H<=13℘.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- J<=0.℘, H>119, D>135, C>3, H<=144.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- I<=0, H>130, E>295, E<=412.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- K<=1.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- J<=0.5, A>39, C<=2, C>G.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- M<=3.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- A>50, H>150.
is_absence(A,B,C,D,E,F,G,H,I,G,K,L,M) :- H<=14℘, E<=1℘2.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- C<=3, A<=45, C>2.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- B<=0, D<=130, H>100, A<=54, A>33.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- B>J, A>53, G<=I, E>223.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- L>0.
is_absence(A,B,C,D,E,F,G,H,F,J,K,L,M) :- H<=142, H>130, D<=120, A>35.
is_absence(A,B,C,D,E,F,G,H,I,J,K,L,M) :- A>52, A<=53, J>1.5.
```

Figura 6.4.5: Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método Prop, com C2 = absence

```
example(A) :- is_absence(A,B,C,2,D,E,F,F,G,F,F,H,H,H).
example(A) :- is_absence(A,B,C,D,120,E,F,F,G,F,F,H,H,H).
example(A) :- is_absence(A,B,C,D,E,F,G,G,H,G,C,I,J,D).
example(A) :- is_absence(A,B,C,D,E,F,G,G,H,G,G,F,F,F).
example(A) :- is_absence(A,B,C,3,D,E,F,F,G,F,F,H,H,H).
example(A) :- is_absence(A,B,C,D,E,F,G,C,H,G,G,I,I,I).
example(A) :- is_absence(A,B,C,D,E,F,C,G,H,C,C,I,I,I).
example(A) :- is_absence(A,B,C,D,E,F,G,H,I,H,C,J,G,G).
example(A) :- is_absence(A,B,C,D,E,F,G,G,H,C,I,J,K,K), D=<3,
example(A) :- is_absence(A,B,C,D,E,F,G,C,H,G,C,C,I,I).
example(A) :- is_absence(A,B,C,D,E,F,G,H,I,C,C,G,G,J).
example(A) :- is_absence(A,B,C,D,E,F,C,C,G,H,I,J,K,K), 1.500=<I,
example(A) :- is_absence(A,B,C,D,E,F,C,G,H,G,G,I,I,J).
example(A) :- is_absence(A,B,C,D,140,E,F,F,G,F,F,H,H,I).
example(A) :- is_absence(A,B,C,D,E,F,G,H,I,G,G,J,J,J), 2=<H.
example(A) :- is_absence(A,B,C,D,E,F,G,G,H,G,G,I,G,I).
```

Figura 6.4.6: Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método Prop, com C2 = absence

```
is_absence(A) :- a0(A,B), a2(A,C), a9(A,D), a10(E,C), a0(E,B), D<=0.5.
is_absence(A) :- a0(A,B), a2(A,C), a3(A,D), a4(A,E), aº(A,F), F<=0, a10(G,C), E<=254,
                 D<=130, B<=56.
is_absence(A) :- a0(A,B), a2(A,C), a3(A,D), aº(A,E), a9(A,E), a10(G,C), D>120, C>1, B<=46.
is_absence(A) :- a1(A,B), a5(A,C), a6(A,D), aº(A,C), a9(A,B), B<=D.
is_absence(A) :- a0(A,B), a3(A,C), a7(A,D), aº(A,E), E<=0, D>130, D<=14º, C<=120, B>35.
is_absence(A) :- a10(A,B), B<=1.
is_absence(A) :- a12(A,B), B<=3.
is_absence(A) :- a3(A,B), a4(A,C), a5(A,D), a7(A,E), aº(A,D), B>12º, E>136, C>213, B<=140.
is_absence(A) :- a1(A,B), a2(A,C), a3(A,D), a7(A,E), a9(A,F), B>F, D>130, a3(G,E), a2(G,B), C>3.
is_absence(A) :- a2(A,B), a3(A,C), a4(A,D), B<=3, C<=130, D>294.
is_absence(A) :- a0(A,B), a1(A,C), a4(A,D), C<=0, B>51, D<=273.
is_absence(A) :- a0(A,B), a2(A,C), a5(A,D), a6(A,D), aº(A,D), a12(G,C), B>53.
```

Figura 6.4.7: Regras Geradas por FOIL: Conjunto de Dados Hungarian e Método MB, com C2 = absence

```
example(A) :- a2(A,2).
example(A) :- a1(A,B), a5(A,B), aº(A,B), B=<0.
example(A) :- a12(A,B), B=<3.
example(A) :- a1(A,B), a10(A,B).
```

Figura 6.4.8: Regras Geradas por PROGOL: Conjunto de Dados Hungarian e Método MB, com C2 = absence

6.5 Conjunto de Dados Credit

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	aa	2	3	contínuo
#1	ab	-	290	discreto
#2	ac	-	178	contínuo
#3	ad	4	4	contínuo
#4	ae	3	4	discreto
#5	af	14	15	contínuo
#6	ag	9	10	discreto
#7	ah	-	120	discreto
#8	ai	2	2	discreto
#9	aj	2	2	discreto
#10	ak	-	22	discreto
#11	al	2	2	discreto
#12	am	3	3	discreto
#13	an	-	140	discreto
#14	ao	-	177	discreto

Tabela 6.5.1: Credit – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	16.8	4.9	22.1 ± 10.2	16.5 ± 7.0	19.0 ± 6.4	18.2 ± 6.3
$[max, min]$	4.5	22.8	[42.9, 11.8]	[28.6, 6.7]	[34.7, 12.2]	[35.1, 12.3]
Progol(C1)	15.8	5.9	26.6 ± 12.2	8.7 ± 3.4	16.7 ± 5.7	14.1 ± 4.9
$[max, min]$	2.4	24.9	[45.8, 5.3]	[12.5, 0.0]	[28.6, 8.2]	[23.5, 6.4]
Foil(C2)	22.6	4.7	16.8 ± 8.6	26.1 ± 11.4	20.9 ± 3.9	18.6 ± 6.1
$[max, min]$	5.7	16.0	[29.2, 0.0]	[47.8, 5.9]	[26.5, 14.3]	[26.5, 5.1]
Progol(C2)	23.2	4.1	14.4 ± 8.2	15.6 ± 7.3	14.8 ± 5.2	13.7 ± 5.6
$[max, min]$	3.3	18.4	[31.2, 0.0]	[26.3, 4.0]	[22.4, 8.2]	[22.6, 3.9]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	21.8 ± 0.8	19.5 ± 1.0	1.5 ± 1.0
Progol(C1)	47.0 ± 2.1	10.1 ± 2.3	163.5 ± 10.1
Foil(C2)	23.0 ± 1.1	20.9 ± 1.6	1.6 ± 1.6
Progol(C2)	38.9 ± 1.2	11.9 ± 1.5	99.1 ± 7.8

(b)

Tabela 6.5.2: Médias para o Conjunto de Dados Credit e Método Prop, (C1 = positive e C2 = negative)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	17.2	4.5	20.7 ± 6.6	9.9 ± 3.9	14.4 ± 4.1	13.9 ± 3.6
$[max, min]$	2.7	24.6	$[33.3, 12.0]$	$[18.7, 3.3]$	$[20.4, 8.2]$	$[18.9, 7.2]$
Progol(C1)	14.5	7.2	32.4 ± 7.1	7.2 ± 3.6	18.5 ± 4.5	14.8 ± 3.4
$[max, min]$	2.0	25.3	$[44.4, 21.1]$	$[14.3, 3.3]$	$[26.5, 10.2]$	$[21.8, 8.3]$
Foil(C2)	22.2	5.1	17.8 ± 8.7	11.0 ± 6.9	15.0 ± 6.2	12.8 ± 5.8
$[max, min]$	2.4	19.3	$[30.0, 4.5]$	$[21.7, 0.0]$	$[24.5, 2.0]$	$[21.8, 2.1]$
Progol(C2)	22.9	4.4	15.8 ± 7.2	10.8 ± 8.6	13.2 ± 4.6	11.7 ± 4.4
$[max, min]$	2.2	19.5	$[25.0, 4.5]$	$[31.2, 3.7]$	$[20.4, 4.1]$	$[19.4, 4.1]$

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	52.8 ± 6.1	30.8 ± 6.4	589.7 ± 189.5
Progol(C1)	63.6 ± 22.6	5.0 ± 22.7	231.5 ± 122.7
Foil(C2)	62.1 ± 5.2	35.5 ± 5.8	8252.5 ± 2279.7
Progol(C2)	40.5 ± 2.6	2.8 ± 2.6	110.9 ± 7.6

(b)

Tabela 6.5.3: Médias para o Conjunto de Dados Credit e Método MB, (C1 = positive e C2 = negative)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−0.4	+1.4	+0.4	+0.7
C2	+0.3	+1.1	+1.3	+0.8

Tabela 6.5.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−1.7	+0.7	−1.0	−0.3
C2	+0.3	0.0	+0.3	+0.2

Tabela 6.5.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop				Método MB			
Algoritmo	Matriz de Conf.	$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.	$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
FOIL (C1)	216 1 0 273	0.00	0.00	0.00	217 0 0 273	0.00	0.00	0.00
PROGOL (C1)	217 0 13 260	0.00	0.05	0.03	217 0 11 262	0.00	0.04	0.02
FOIL (C2)	265 8 0 217	0.03	0.00	0.02	273 0 0 217	0.00	0.00	0.00
PROGOL (C2)	273 0 14 203	0.00	0.06	0.03	273 0 16 201	0.00	0.07	0.03

Tabela 6.5.6: Erro Aparente para o Conjunto de Dados Credit (C1 = positive e C2 = negative)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
FOIL (C1)	23	22	1.9	56	32	3236.1
PROGOL (C1)	45	12	161.92	76	8	321.85
FOIL (C2)	26	18	1.6	84	46	13019.5
PROGOL (C2)	51	11	145.19	46	3	135.92

Tabela 6.5.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Credit (C1 = positive e C2 = negative)

```
is_positive(A,B,C,D,E,F,G,H,Ft,J,K,L,M,N,O) :- O>500.
is_positive(A,B,C,D,Cg,F,G,H,Ft,J,K,L,M,N,O) :- N<=100, H>0.375.
is_positive(A,B,C,D,E,F,G,H,I,J,K,L,Gg,N,O) :- K>3, O<=0.
is_positive(A,B,C,D,E,F,G,H,Ft,J,K,L,M,N,O) :- O>5, C>1.75, H>0.375.
is_positive(A,B,C,D,E,F,Eh,H,Ft,J,K,L,M,N,O) :- H<=4.165, B>20.67.
is_positive(A,B,C,D,E,F,G,H,Ft,J,K,L,M,N,O) :- C<=0.25.
is_positive(A,B,C,D,E,F,Ev,H,Ft,J,K,L,M,N,O) :- C>1.5, C<=4.25, H<=2.5.
is_positive(A,B,C,D,E,F,G,H,Ft,J,K,Ff,M,N,O) :- O>225.
is_positive(Ab,B,C,D,E,F,Ev,H,I,J,K,L,M,N,O) :- H>1.375, B<=26.67, C<=10, C>2.21.
is_positive(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O) :- C<=H, N>212, B>33, H<=3.5.
is_positive(A,B,C,D,E,Dq,G,H,Ft,J,K,Ft,M,N,O) :- C<=6.5, O<=400.
is_positive(A,B,C,D,E,F,G,H,I,I,K,L,M,N,O) :- C<=H, N>292, B<=2?.25.
is_positive(A,B,C,D,E,F,G,H,I,J,K,L,Gp,N,O) :- B>34.5?.
is_positive(A,B,C,D,E,F,G,H,Ft,J,K,L,M,N,O) :- B<=23.92, B>23.
is_positive(A,B,C,D,Cg,F,G,H,Ft,J,K,L,M,N,O) :- B>44.25, H>1.625.
is_positive(A,B,C,D,E,Dc,G,H,I,J,K,L,Gg,N,K) :- N<=12?, C<=0.?3.
```

Figura 6.5.1: Regras Geradas por FOIL: Conjunto de Dados Credit e Método Prop, com C1 = positive

```
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,J,J,K,L,M,N,O), 0.625=<I,
example(A) :- is_positive(A,B,C,D,bu,E,F,G,H,ft,ft,I,J,K,L,M),
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,ft,J,K,J,L,M,K),
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,ft,J,K,ft,L,M,N),
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,J,J,K,L,M,N,O), 4=<K.
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,ft,J,K,J,L,M,N),
example(A) :- is_positive(A,B,C,D,E,F,G,eh,H,ft,I,J,ft,K,L,J),
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,J,J,K,L,M,N,O), 1=<K,
example(A) :- is_positive(A,B,C,D,E,F,G,H,I,ft,J,K,ft,L,M,K),
example(A) :- is_positive(A,B,C,D,bu,E,F,G,H,ft,I,J,ft,K,L,M),
example(A) :- is_positive(A,B,C,D,E,F,doldw,G,H,ft,ff,I,ft,J,
example(A) :- is_positive(A,B,C,D,E,F,G,ev,H,I,I,J,K,L,M,N),
```

Figura 6.5.2: Regras Geradas por PROGOL: Conjunto de Dados Credit e Método Prop, com C1 = positive

```
is_positive(A) :- a°(A,ft), a14(A,C), C>500.
is_positive(A) :- a°(A,ft), a4(A,cg), a1(A,D), a13(A,E), D>19.75, E<=°0.
is_positive(A) :- a°(A,ft), a10(A,C), a14(A,D), C>3, a13(E,D).
is_positive(A) :- a°(A,ft), a2(A,C), a7(A,D), a14(A,E), E>5, C>1.75, D>0.375.
is_positive(A) :- a°(A,ft), a1(A,C), a13(A,D), C<=2°.5, D<=100, D>75.
is_positive(A) :- a°(A,ft), a12(A,gg), a2(A,D), a13(A,E), D<=3.165, E>212.
is_positive(A) :- a2(A,B), a7(A,C), a13(A,D), C>1.165, B<=0.°3, D<=300, D>160.
is_positive(A) :- a°(A,ft), a7(A,C), a13(A,D), a14(A,E), C>1.29, E<=0, C<=2.54, D<=360.
is_positive(A) :- a0(A,ab), a°(A,ff), a2(A,D), a7(A,E), a13(A,F), D<=E, F>420.
```

Figura 6.5.3: Regras Geradas por FOIL: Conjunto de Dados Credit e Método MB, com C1 = positive

```
example(A) :- a°(A,B), a9(A,B), a10(A,C), 7=<C.
example(A) :- a10(A,6).
example(A) :- a1(A,33.170).
example(A) :- a°(A,ft), a10(A,B), 4=<B.
example(A) :- a14(A,B), 5000=<B.
example(A) :- a°(A,ft), a10(A,1).
example(A) :- a14(A,300).
example(A) :- a°(A,ft), a14(A,B), 444=<B.
```

Figura 6.5.4: Regras Geradas por PROGOL: Conjunto de Dados Credit e Método MB, com C1 = positive

```
example(A) :- a°(A,ff).
example(A) :- a14(A,5).
example(A) :- a1(A,?).
```

Figura 6.5.5: Regras Geradas por PROGOL: Conjunto de Dados Credit e Método MB, com C2 = negative

```
is_negative(A,B,C,D,E,F,G,H,Ff,J,K,L,M,N,O) :- C>H, C<=6, N>100.
is_negative(A,B,C,D,E,F,G,H,Ff,J,K,L,M,N,O) :- N<=120, H>0.165.
is_negative(A,B,C,D,E,F,G,H,Ff,J,K,L,M,N,O) :- C>H, O<=1110, H<=O.
is_negative(A,B,C,D,E,F,G,H,Ff,J,K,L,M,N,O) :- B<=33.5º, B>22.5, C>0.665.
is_negative(A,B,C,D,E,F,G,H,I,J,K,L,M,N,K) :- N>100, N<=200, B>23.25.
is_negative(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O) :- N>O, C<=1.375, H<=1, N<=2ºº, H>0.21.
is_negative(A,B,C,D,E,F,G,H,I,J,K,L,M,N,K) :- N>12º, B<=23, H<=2.
is_negative(A,B,C,D,Cp,F,G,H,I,J,K,I,M,N,O) :- O<=225, C<=1.5.
is_negative(Ab,B,C,D,E,F,G,H,Ft,J,K,L,M,N,O) :- O<=3º, H<=O, B<=25.
is_negative(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O) :- K<=2, O<=1, C>13.5º5.
is_negative(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O) :- K<=2, H<=1.29, H>0.º35, N>60, C>0.5.
is_negative(A,B,C,D,Cg,F,Ev,H,I,J,K,L,M,N,K) :- N>360.
is_negative(A,B,C,D,E,F,G,H,Ff,J,K,Ff,M,N,O) :- H<=0.0º5, B<=36.17.
is_negative(A,B,C,D,E,F,Ev,H,I,J,K,L,M,N,O) :- C>O, H>2.54, N>230, C>3.165.
```

Figura 6.5.6: Regras Geradas por FOIL: Conjunto de Dados Credit e Método Prop, com C2 = negative

```
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,ff,J,K,J,L,M,N),
example(A) :- is_negative(A,B,C,D,E,F,dm,G,H,I,J,K,J,L,M,K).
example(A) :- is_negative(A,B,C,D,by,E,F,G,H,I,J,K,J,L,M,K),
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,J,K,I,J,L,M,I), 10=<D.
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,J,J,K,ft,L,M,N),
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,J,J,K,ft,L,M,K).
example(A) :- is_negative(A,B,C,D,E,F,dff,G,H,I,I,J,K,L,M,N),
example(A) :- is_negative(A,B,?,C,D,E,F,G,H,I,J,K,I,L,M,K).
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,ff,ft,J,ff,K,L,M).
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,J,J,K,ft,L,M,N),
example(A) :- is_negative(A,B,C,D,E,F,G,H,I,J,J,K,J,L,M,N), 2.500=<D,
```

Figura 6.5.7: Regras Geradas por PROGOL: Conjunto de Dados Credit e Método Prop, com C2 = negative

```
is_negative(A) :- aº(A,ff), a2(A,C), a7(A,D), a13(A,E), C>D, a7(F,C), E>100.
is_negative(A) :- aº(A,ff), a7(A,C), a13(A,D), D<=144, a2(E,C), a4(E,cp).
is_negative(A) :- aº(A,ff), a2(A,C), a7(A,D), a14(A,E), C>1.54, E<=1000, a2(F,D).
is_negative(A) :- a9(A,ff), a6(A,ev), a7(A,D), a13(A,E), E>102, E<=200, D<=0.795.
is_negative(A) :- a1(A,B), a13(A,C), a14(A,D), C>D, C>239, C<=292, B<=24.5º.
is_negative(A) :- a9(A,ff), a7(A,C), a13(A,D), a14(A,E), E<=112, D>100, C>3.5, D<=292.
is_negative(A) :- a6(A,ev), a1(A,C), a2(A,D), a13(A,E), a14(A,F), D<=1.5, E>F, C>37.5.
is_negative(A) :- aº(A,ff), a9(A,ft).
is_negative(A) :- a13(A,B), B>711.
is_negative(A) :- a1(A,B), a7(A,C), a13(A,D), a14(A,E), a14(F,C), D>E, B>4º.5º, D>22.
is_negative(A) :- a5(A,di), a1(A,C), a10(A,D), a13(A,E), a14(A,F), E>F, C<=33.5º, D<=2.
```

Figura 6.5.8: Regras Geradas por FOIL: Conjunto de Dados Credit e Método MB, com C2 = negative

6.6 Conjunto de Dados Hepatitis

Número Atrib.	Nome Atrib.	#Valores Distintos possíveis	reais	tipo
#0	age	-	49	contínuo
#1	female	2	2	discreto
#2	steroid	2	3	discreto
#3	antivirals	2	2	discreto
#4	fatigue	2	3	discreto
#5	malaise	2	3	discreto
#6	anorexia	2	3	discreto
#7	liver-big	2	3	discreto
#8	liver-firm	2	3	discreto
#9	spleen-palpable	2	3	discreto
#10	spiders	2	3	discreto
#11	ascites	2	3	discreto
#12	varices	2	3	discreto
#13	bilirubin	-	34	contínuo
#14	alk-phosphate	-	83	contínuo
#15	sgot	-	84	contínuo
#16	albumin	-	29	contínuo
#17	protime	-	44	contínuo
#18	histology	2	2	discreto

Tabela 6.6.1: Hepatitis – Descrição dos Atributos

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	1.1	2.1	65.5 ± 30.5	14.1 ± 7.6	25.1 ± 10.5	29.2 ± 14.7
$[max, min]$	1.8	10.5	[100.0, 0.0]	[28.6, 0.0]	[43.7, 6.7]	[53.5, 2.8]
Progol(C1)	0.0	3.2	100.0 ± 0.0	0.0 ± 0.0	20.4 ± 10.5	10.0 ± 0.0
$[max, min]$	0.0	12.3	[100.0, 100.0]	[0.0, 0.0]	[33.3, 6.2]	[10.0, 10.0]
Foil(C2)	10.6	1.7	13.1 ± 12.4	55.5 ± 37.3	23.2 ± 13.1	23.1 ± 18.9
$[max, min]$	1.9	1.3	[46.2, 0.0]	[100.0, 0.0]	[53.3, 6.7]	[68.0, 2.8]
Não obteve resultados para Progol(C2) prop						

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	9.5 ± 0.5	6.4 ± 0.7	0.4 ± 0.7
Progol(C1)	28.8 ± 1.9	0.0 ± 1.9	5296.5 ± 257.3
Foil(C2)	9.6 ± 2.7	8.8 ± 3.3	0.3 ± 3.3
Não obteve resultados para Progol(C2) prop			

(b)

Tabela 6.6.2: Médias para o Conjunto de Dados Hepatitis e Método Prop, (C1 = die e C2 = live)

Algoritmo	Matriz		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	$\sqrt{\frac{F_N}{T_P+F_N}\frac{F_P}{F_P+T_N}}$
Foil(C1)	1.0	2.2	67.5 ± 29.5	6.9 ± 6.7	19.9 ± 7.4	17.6 ± 10.7
$[max, min]$	0.9	11.4	[100.0, 0.0]	[18.2, 0.0]	[33.3, 12.5]	[37.0, 3.8]
Progol(C1)	0.6	2.6	79.0 ± 30.5	5.4 ± 6.6	21.1 ± 9.0	15.6 ± 8.8
$[max, min]$	0.7	11.6	[100.0, 0.0]	[21.4, 0.0]	[33.3, 6.2]	[30.2, 4.6]
Foil(C2)	10.4	1.9	15.1 ± 12.0	51.0 ± 33.3	23.9 ± 12.3	23.4 ± 14.2
$[max, min]$	1.8	1.4	[46.2, 0.0]	[100.0, 0.0]	[46.7, 6.7]	[48.1, 2.8]
Progol(C2)	9.9	2.4	19.1 ± 12.4	44.5 ± 28.5	26.3 ± 8.6	22.3 ± 14.7
$[max, min]$	1.7	1.5	[35.7, 0.0]	[80.0, 0.0]	[40.0, 12.5]	[45.3, 3.6]

(a)

Algoritmo	# Cláusulas	# Regras	Tempo Médio
Foil(C1)	21.2 ± 2.3	11.9 ± 2.5	16.2 ± 3.1
Progol(C1)	20.1 ± 1.2	2.4 ± 1.2	31.8 ± 1.7
Foil(C2)	14.7 ± 5.0	9.9 ± 5.8	15.4 ± 6.3
Progol(C2)	15.5 ± 2.6	4.0 ± 3.4	13.3 ± 3.4

(b)

Tabela 6.6.3: Médias para o Conjunto de Dados Hepatitis e Método MB, (C1 = die e C2 = live)

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−1.6	+2.6	+0.4	+1.8
Só FOIL conseguiu rodar para Hepatitis – C2				

Tabela 6.6.4: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método Prop

Conceito	Erro Neg.	Erro Pos.	Erro Total	Erro G
C1	−0.4	+0.2	−0.1	+0.2
C2	−0.3	+0.2	−0.2	+0.1

Tabela 6.6.5: Diferença em Desvios-Padrão dos Erros entre FOIL e PROGOL com Método MB

	Método Prop					Método MB				
Algoritmo	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$	Matriz de Conf.		$\frac{F_N}{T_P+F_N}$	$\frac{F_P}{F_P+T_N}$	$\frac{F_P+F_N}{n}$
FOIL (C1)	29	3	0.09	0.00	0.02	32	0	0.00	0.00	0.00
	0	123				0	123			
PROGOL (C1)	273	0	0.09	0.00	0.02	32	0	0.00	0.02	0.01
	14	203				2	121			
FOIL (C2)	122	1	0.01	0.00	0.01	123	0	0.00	0.00	0.00
	0	32				0	32			
PROGOL (C2)	273	0	0.01	0.00	0.01	123	0	0.00	0.34	0.07
	14	203				11	21			

Tabela 6.6.6: Erro Aparente para o Conjunto de Dados Hepatitis (C1 = die e C2 = live)

	Método Prop			Método MB		
Algoritmo	Total Cl.	Regras	Tempo de Exec.	Total Cl.	Regras	Tempo de Exec.
FOIL (C1)	10	7	0.4	21	13	20.3
PROGOL (C1)	30	0	3813.86	19	3	30.68
FOIL (C2)	11	10	0.4	16	10	18.4
PROGOL (C2)	Não obteve resultados			18	5	13.46

Tabela 6.6.7: Número Total de Cláusulas, Número de Regras e Tempo de Execução em Relação ao Erro Aparente do Conjunto de Dados Hepatitis (C1 = die e C2 = live)

```
is_die(A,B,C,D,E,F,G,H,I,J,K,E,M,N,O,P,Q,R,booleanyes) :- N>1.6.
is_die(A,booleanno,C,D,booleanno,F,booleanyes,H,I,booleanno,booleanno,L,M,N,O,P,Q,R,S) :- O<=167.
is_die(A,B,C,D,E,F,G,H,I,J,K,K,M,N,O,P,Q,R,S) :- Q<=2.℘.
is_die(A,booleanno,C,D,booleanno,F,G,H,I,J,K,L,M,N,O,P,Q,R,S) :- R<=41, A>32.
is_die(A,B,C,D,E,F,G,booleanno,I,J,K,L,M,N,O,P,Q,R,S) :- N>2.3, N<=2.9.
```

Figura 6.6.1: Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método Prop, com C1 = die

Figura 6.6.2: Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método Prop, com C1 = die

```
is_die(A) :- a4(A,B), a11(A,B), a1℘(A,booleanyes), a13(A,D), D>1.6.
is_die(A) :- a1(A,booleanno), a4(A,booleanno), a6(A,booleanyes), a9(A,booleanno),
             a10(A,booleanno), a14(A,G), G<=167.
is_die(A) :- a1(A,booleanno), a7(A,booleanyes), a16(A,D), D<=2.℘.
is_die(A) :- a1(A,booleanno), a4(A,booleanno), a0(A,D), a17(A,E), E<=41, D>32.
```

Figura 6.6.3: Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método MB, com C1 = die

```
example(A) :- a13(A,B), 3.900=<B.
example(A) :- a16(A,B), B=<2.600.
example(A) :- a16(A,2.℘00).
```

Figura 6.6.4: Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método MB, com C1 = die

```
is_live(A,B,C,D,E,F,G,H,I,J,J,L,M,N,O,P,Q,R,S) :- Q>3.℘.
is_live(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,booleanno) :- N<=1.6, O>67.
is_live(A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S) :- A<=32, Q>2.℘.
is_live(A,B,C,D,E,C,G,H,booleanyes,booleanyes,K,L,M,N,O,P,Q,R,S) :- P>℘0.
is_live(A,B,C,D,E,F,G,H,booleanno,J,K,L,booleanno,N,O,P,Q,R,S) :- R>50.
```

Figura 6.6.5: Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método Prop, com C2 = live

Figura 6.6.6: Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método Prop, com C2 = live

```
is_live(A) :- a9(A,B), a10(A,B), a16(A,C), C>3.℘.
is_live(A) :- a1℘(A,booleanno), a13(A,C), a14(A,D), C<=1.6, D>67.
is_live(A) :- a4(A,B), a7(A,B), a℘(A,C), a12(A,C).
is_live(A) :- a2(A,booleanyes), a℘(A,booleanno), a12(A,booleanyes), a1℘(A,booleanyes).
is_live(A) :- a0(A,B), a16(A,C), B<=32, C>2.℘.
is_live(A) :- a6(A,booleanyes), a7(A,booleanno), a1℘(A,booleanyes).
is_live(A) :- a1(A,B), a2(A,B), a℘(A,booleanyes), a15(A,D), a9(A,booleanyes), D>℘0.
```

Figura 6.6.7: Regras Geradas por FOIL: Conjunto de Dados Hepatitis e Método MB, com C2 = live

```
example(A) :- a3(A,B), a4(A,B), a12(A,B).
example(A) :- a4(A,booleanno), a1℘(A,booleanno).
example(A) :- a4(A,B), a7(A,B), a℘(A,B).
example(A) :- a7(A,B), a11(A,B), a15(A,C), 173=<C.
example(A) :- a3(A,booleanno), a℘(A,booleanno).
```

Figura 6.6.8: Regras Geradas por PROGOL: Conjunto de Dados Hepatitis e Método MB, com C2 = live

7 Comparação dos Resultados

Com relação ao tempo de execução, deve ser observado que PROGOL mostrou-se muito lento quando comparado com FOIL. Como já mencionado, nestes experimentos FOIL foi executado em um computador mais lento que aquele onde foi executado PROGOL. Ainda assim, o tempo de execução de PROGOL é, na grande maioria dos casos, muito maior que o tempo de execução de FOIL.

A Tabela 7.1 mostra, para todos os conjuntos de dados, o erro total[2] e o número de cláusulas (com corpo e sem) e regras que participam da hipótese induzida. Com relação ao número médio de cláusulas, pode-se observar que, no caso de FOIL, esse número é sempre maior quando são utilizados os dados no formato MB. O mesmo acontece com o número médio de regras. Porém, isso não se verifica no caso de PROGOL. Considerando o número médio de cláusulas, algumas vezes esse valor é inferior quando o formato MB é utilizado, por exemplo para o conjunto de dados Hepatitis.

Conjunto de Dados	Algoritmo	$\frac{F_P+F_N}{n}$		# Cláusulas		# Regras	
		Prop	MB	Prop	MB	Prop	MB
Bupa	Foil(C1)	30.2 ± 7.4	29.5 ± 7.8	26.8 ± 3.4	60.8 ± 0.8	16.5 ± 4.3	35.8 ± 1.2
	Progol(C1)	30.7 ± 4.8	30.9 ± 5.2	44.3 ± 3.5	55.5 ± 7.9	6.4 ± 3.8	4.4 ± 8.0
	Foil(C2)	27.3 ± 10.3	31.6 ± 9.8	28.0 ± 4.2	55.2 ± 0.2	24.4 ± 4.2	33.9 ± 1.2
	Progol(C2)	30.0 ± 8.7	35.8 ± 6.1	29.3 ± 2.8	38.2 ± 3.0	8.3 ± 3.4	7.3 ± 3.8
Pima	Foil(C1)	32.4 ± 5.0	30.0 ± 3.0	42.8 ± 6.5	95.3 ± 3.9	37.0 ± 8.4	60.3 ± 4.3
	Progol(C1)	30.6 ± 3.9	32.0 ± 4.4	92.2 ± 17.1	119.9 ± 31.5	24.6 ± 17.1	22.3 ± 33.3
	Foil(C2)	30.1 ± 5.8	27.9 ± 5.8	48.0 ± 6.3	159.7 ± 18.7	30.6 ± 7.4	87.1 ± 20.1
	Progol(C2)	26.4 ± 5.9	26.1 ± 6.4	103.2 ± 4.4	128.7 ± 2.4	19.6 ± 4.7	10.3 ± 3.0
Breast	Foil(C1)	19.3 ± 9.9	22.2 ± 6.5	20.9 ± 6.2	44.3 ± 7.1	13.5 ± 6.4	26.5 ± 7.5
	Progol(C1)	19.8 ± 6.9	21.6 ± 7.4	46.4 ± 3.8	37.9 ± 9.6	4.0 ± 4.3	2.5 ± 10.3
	Foil(C2)	24.8 ± 9.5	20.1 ± 6.3	19.5 ± 1.6	41.9 ± 0.1	17.1 ± 4.4	26.6 ± 0.4
	Progol(C2)	28.4 ± 9.6	25.5 ± 6.0	45.5 ± 3.7	44.3 ± 12.3	13.2 ± 3.8	4.5 ± 12.3
Hungaria	Foil(C1)	19.3 ± 9.9	22.2 ± 6.5	20.9 ± 6.2	44.3 ± 7.1	13.5 ± 6.4	26.5 ± 7.5
	Progol(C1)	19.8 ± 6.9	21.6 ± 7.4	46.4 ± 3.8	37.9 ± 9.6	4.0 ± 4.3	2.5 ± 10.3
	Foil(C2)	24.8 ± 9.5	20.1 ± 6.3	19.5 ± 1.6	41.9 ± 0.1	17.1 ± 4.4	26.6 ± 0.4
	Progol(C2)	28.4 ± 9.6	25.5 ± 6.0	45.5 ± 3.7	44.3 ± 12.3	13.2 ± 3.8	4.5 ± 12.3
Credit	Foil(C1)	19.0 ± 6.4	14.4 ± 4.1	21.8 ± 0.8	52.8 ± 6.1	19.5 ± 1.0	30.8 ± 6.4
	Progol(C1)	16.7 ± 5.7	18.5 ± 4.5	47.0 ± 2.1	63.6 ± 22.6	10.1 ± 2.3	5.0 ± 22.7
	Foil(C2)	20.9 ± 3.9	15.0 ± 6.2	23.0 ± 1.1	62.1 ± 5.2	20.9 ± 1.6	35.5 ± 5.8
	Progol(C2)	14.8 ± 5.2	13.2 ± 4.6	38.9 ± 1.2	40.5 ± 2.6	11.9 ± 1.5	2.8 ± 2.6
Hepatitis	Foil(C1)	25.1 ± 10.5	19.9 ± 7.4	9.5 ± 0.5	21.2 ± 2.3	6.4 ± 0.7	11.9 ± 2.5
	Progol(C1)	20.4 ± 10.5	21.1 ± 9.0	28.8 ± 1.9	20.1 ± 1.2	0.0 ± 1.9	2.4 ± 1.2
	Foil(C2)	23.2 ± 13.1	23.9 ± 12.3	9.6 ± 2.7	14.7 ± 5.0	8.8 ± 3.3	9.9 ± 5.8
	Progol(C2)	–	26.3 ± 8.6	–	15.5 ± 2.6	–	4.0 ± 3.4

Tabela 7.1: Taxas Médias de Erro e Número Médio de Cláusulas e Regras

Com relação ao número médio de regras, FOIL também gera um número maior quando os dados estão no formato MB, exceto para o conjunto de dados Hepatitis. Para esse conjunto de dados não é possível tirar alguma conclusão, devido ao elevado desvio padrão. Entretanto, isso não

[2]Não foi possível executar PROGOL (C2) com o conjunto de dados Hepatitis e formato "proposicional", pois PROGOL entrou em "loop".

acontece com PROGOL que mostra uma clara tendência a gerar um pequeno número de regras utilizando esse formato. Na realidade, os dados no formato MB mostram-se mais apropriados para serem usados com PROGOL.

A Tabela 7.2 mostra a diferença em desvios-padrão de FOIL para PROGOL para o método "proposicional" e MB. Ou seja, valores negativos indicam que o erro cometido por FOIL é menor que o cometido por PROGOL, enquanto que valores positivos indicam que o erro cometido por PROGOL é menor que o cometido por FOIL. Como pode ser observado, em nenhum dos casos considerados esse valor é maior que 2 ou menor que -2. Portanto, nenhum dos sistemas mostrou, para esses conjuntos de dados, um desempenho melhor que o outro com nível de confiança de 95%.

Conjunto de Dados	Conceito	Erro Total	
		Prop	MB
Bupa	C1 = um	-0,3	+0,1
	C2 = dois	-0,8	-1,0
Pima	C1 = zero	+0,4	-0,5
	C2 = um	+0,6	+0,3
Breast	C1 = recurrence	-0,1	-0,2
	C2 = no_recurrence	-0,3	-0,5
Hungaria	C1 = presence	-0,1	+0,1
	C2 = absence	-0,4	-0,9
Credit	C1 = negative	+0,4	-1,0
	C2 = positive	+1,3	+0,3
Hepatitis	C1 = die	+0,4	-0,1
	C2 = live	—	-0,2

Tabela 7.2: Diferenças em Desvios-Padrão dos Erros entre FOIL e PROGOL

8 Considerações Finais

Neste trabalho foram apresentados resultados de uma comparação empírica entre os sistemas de Programação Lógica Indutiva FOIL e PROGOL. Ambos sistemas foram executados com seus parâmetros default utilizando seis conjuntos de dados de problemas do mundo real tirados do Repositório de Aprendizado de Máquina da UCI.

Todos os dados estavam inicialmente no formato atributo-valor, formato bastante utilizado nos sistemas de AM proposicional. Esses dados representados na forma relacional também podem ser utilizados na execução de experimentos com sistemas de PLI. Neste trabalho foram utilizadas duas possíveis representações relacionais:

1. cada exemplo é representado utilizando um único predicado cuja aridade é determinada pelo número de atributos;

2. cada exemplo é representado por predicados diferentes, de aridade 2, associados a cada atributo, onde o primeiro argumento identifica o exemplo e o segundo argumento o valor

do atributo correspondente.

Esse último formato é denominado neste trabalho de MB [Caulkins, 2000].

A preparação dos dados foi feita utilizando uma série de ferramentas escritas na linguagem Perl, descritas em [Caulkins e Monard, 2000]. A partir dos dados preparados, foram realizados experimentos para avaliar o desempenho dos sistemas FOIL e PROGOL em relação a cada formato. Através de "10-fold cross-validation" estimou-se o erro de classificação para ambos sistemas. Além disso, regras finais foram induzidas utilizando todo o conjunto de dados de cada problema de forma a criar um único modelo para cada conjunto de dados. Esses modelos serão melhor analisados em trabalhos futuros.

Observando os resultados, percebe-se que o formato MB é mais apropriado para PROGOL, enquanto que o "proposicional" se mostra mais apropriado para FOIL. Uma possível explicação é o fato de FOIL ter sido desenvolvido como uma versão "relacional" de $\mathcal{ID}3$ e $\mathcal{C}4.5$, sistemas de AM proposicional criados pelo Prof. J. Ross Quinlan [Baranauskas e Monard, 2000a], enquanto PROGOL foi desenvolvido a partir de uma teoria relacional, baseada em lógica de primeira ordem.

Deve ser ressaltado que o tempo de processamento de PROGOL de um modo geral é muito grande. Dentro de uma proposta de utilizar sistemas de aprendizado de máquina de primeira ordem para realizar Data Mining, o sistema PROGOL necessita ainda ser trabalhado no sentido de reduzir seu tempo de execução para que se possa trabalhar com conjuntos de dados considerados grandes.

Referências

[Baranauskas e Monard, 2000a] Baranauskas, J. A. e Monard, M. C. (2000a). An Unified Overview of Six Supervised Symbolic Machine Learning Inducers. Relatório Técnico 103, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_103.ps.zip.

[Baranauskas e Monard, 2000b] Baranauskas, J. A. e Monard, M. C. (2000b). Reviewing some Machine Learning Concepts and Methods. Relatório Técnico 102, ICMC – USP, São Carlos, SP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_102.ps.

[Batista, 1997] Batista, G. E. A. P. A. (1997). Um Ambiente de Avaliação de Algoritmos de Aprendizado de Máquina utilizando Exemplos. Dissertação de Mestrado, ICMC-USP.

[Blake et al., 1998] Blake, C., Keogh, E., e Merz, C. J. (1998). *UCI Repository of Machine Learning Datasets*. University of California, Department of Information and Computer Science, Irvine, CA. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[Caulkins, 2000] Caulkins, C. W. (2000). Aquisição de Conhecimento Utilizando Aprendizado de Máquina Relacional. Dissertação de Mestrado, ICMC-USP.

[Caulkins e Monard, 2000] Caulkins, C. W. e Monard, M. C. (2000). Ferramentas para Conversão de Dados de Entrada dos Algoritmos de Aprendizado de Máquina Proposicional CN2 e C4.5 para os Algoritmos de Aprendizado de Máquina Relacional Foil e Progol. Relatório Técnico 116, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_116.ps.zip.

[Clark e Niblett, 1989] Clark, P. e Niblett, T. (1989). The CN2 Induction Algorithm. *Machine Learning*, 9(1):57–94.

[Lee et al., 1999] Lee, H. D., Monard, M. C., e Baranauskas, J. A. (1999). Empirical Comparison of Wrapper and Filter Approaches for Feature Subset Selection. Relatório Técnico 94, ICMC-USP. ftp://ftp.icmc.sc.usp.br/pub/BIBLIOTECA/rel_tec/Rt_94.ps.zip.

[Michalski, 1983] Michalski, R. S. (1983). A Theory and Methodology of Inductive Learning. In Michalski, R. S., Carbonell, J. G., e Mitchell, T. M., editores, *Machine Learning: An Artificial Intelligence Approach*, volume I, pages 83–184, Palo Alto, CA. Tioga.

[Morik e Brockhausen, 1997] Morik, K. e Brockhausen, P. (1997). A Multistrategy Approach to Relational Knowledge Discovery in Databases. *Machine Learning*, 27:287–312.

[Muggleton, 1995] Muggleton, S. (1995). Inverse Entailment and Progol. *New Generation Computing*, 13:245–286.

[Muggleton, 1997] Muggleton, S. (1997). Learning from Positive Data. In *Proceedings of the Sixth International Workshop on Inductive Logic Programming*, volume LNAI 1314. Springer Verlag.

[Quinlan, 1986] Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*, 1:81–106.

[Quinlan, 1990] Quinlan, J. R. (1990). Learning Logical Definitions from Relations. *Machine Learning*, 5(3):239–266.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5 Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, California.

[Quinlan e Cameron-Jones, 1993] Quinlan, J. R. e Cameron-Jones, R. M. (1993). FOIL: a Midterm Report. In *Proceedings European Conference on Machine Learning*, pages 3–20, Vienna, Berlin. Springer Verlag.

[Sammut, 1993] Sammut, C. A. (1993). The Origins of Inductive Logic Programming: A Prehistoric Tale. In *Proceedings Third International Workshop on Inductive Logic Programming*, pages 127–147, Bled, Slovenia. Springer Verlag.

[Shapiro, 1983] Shapiro, E. Y. (1983). *Algorithmic Program Debugging*. The MIT Press.

[Weiss e Indurkhya, 1998] Weiss, S. M. e Indurkhya, N. (1998). *Predictive Data Mining: A Practical Guide*. Morgan Kaufmann Publishers, San Francisco, CA.