Explorer Pattern Tools: Uma Ferramenta para Mineração de Dados Utilizando Algoritmos Genéticos

Joilma S. Santos[1], **Daniela Barreiro Claro**[1]

[1]Laboratorio de Sistemas Distribuidos (LaSiD)
Departamento de Ciência da Computação
Universidade Federal da Bahia (UFBA)
Av. Adhemar de Barros, s/n, Ondina – Salvador – BA – Brasil

joilma@dcc.ufba.br, dclaro@ufba.br

Abstract. *The Data Mining enables the extraction of useful information from databases quickly through algorithms based on Artificial Intelligence, such as Genetic Algorithms. The use of Genetic Algorithms in Data Mining allows various combinations between attributes, thanks to their genetic operator of crossover, which provides a better interaction between the attributes found in algorithms based on greedy strategy and the evaluation, which selects relevant data to be mined. This paper describes the implementation of a Genetic Algorithm for Data Mining and its incorporation into the Explorer Pattern Tree, designed to establish standards for using Fuzzy Logic implementation of Data Mining, leading to Explorer Pattern Tools. Besides the description of the techniques used in the algorithm genetic implemented, this article also describes the results of tests performed on four databases.*

Resumo. *Mineração de Dados permite a extração de informação útil de bases de dados de forma rápida através de algoritmos baseados na Inteligência Artificial, como por exemplo, Algoritmos Genéticos. A utilização de Algoritmos Genéticos na Mineração de Dados permite várias combinações entre atributos, graças ao seus operador genético de crossover, que proporciona uma melhor interação entre atributos que a encontrada em algoritmos baseados na estratégia gulosa, e a função de avaliação, que seleciona informação relevante dos dados a serem minerados. Este artigo descreve a implementação de um Algoritmo Genético para Mineração de Dados e sua incorporação à ferramenta Explorer Pattern Tree, desenvolvida para determinar padrões utilizando Lógica Fuzzy para realização de Mineração de Dados, dando origem a ferramenta Explorer Pattern Tools. Além da descrição das técnicas utilizadas no algoritmo genético implementado, este artigo também descreve os resultados de testes realizados sobre quatro bases de dados.*

1. Introdução

O processo de Mineração de Dados ou Data Mining analisa e extrai informações úteis a partir de dados. A captação de conhecimento de alto nível através deste processo se dá a partir da procura de padrões consistentes e/ou relacionamentos sistemáticos entre instâncias de dados, classificando-os em subconjuntos de informações baseados em regras estatísticas e da teoria da informação.

Algoritmos de aprendizado de máquina são amplamente utilizados na tarefa de Mineração de Dados. Estes algoritmos são baseados na construção de árvores de decisão e utilizam como matéria-prima dados de treinamento. Através do percurso destas árvores, da raiz até um nó folha é possível inferir classes baseados em determinados valores para cada atributo.

O WEKA *(Waikato Environment for Knowledge Analysis)* [Witten and Frank 2005] é a principal ferramenta gratuita para Mineração de Dados, baseada em algoritmos de aprendizado de máquina. O WEKA permite a realização da mineração de dados utilizando diferentes abordagens e algoritmos, disponibilizando ferramentas como seleção de atributos, visualização de gráficos, dentre outras.

Para uma classificação com maior percentual de acerto (acurácia), métodos de aprendizado de máquina baseados em Lógica Fuzzy têm-se mostrado mais eficientes que métodos clássicos [Souza and Claro 2008]. Apesar da perceptível contribuição que a Lógica Fuzzy proporciona à acurácia das árvores de classificação utilizadas na Mineração de Dados o WEKA não implementa técnicas com algoritmos baseados na lógica nebulosa.

Para disponibilizar uma ferramenta gratuita que realizasse Mineração de Dados utilizando árvores de classificação fuzzy e testar a acurácia dessas árvores em relação a árvores de classificação clássicas, foi desenvolvido o EFT *(Explorer Fuzzy Tree)* [Souza and Claro 2008]. Para determinar a acurácia das árvores fuzzy implementadas pelo EFT, foram feitos testes utilizando cinco bases de dados.

Outra abordagem para realização da Mineração de Dados utiliza algoritmos de aprendizado de máquina baseados na Computação Evolutiva - os Algoritmos Genéticos. Estes algoritmos realizam buscas globais e podem trabalhar ao mesmo tempo com uma quantidade maior de atributos que os algoritmos baseados na estratégia gulosa que são geralmente utilizados para a tarefa de Mineração de Dados e por isso são extremamente úteis na procura de padrões relevantes em dados [Freitas 2002b] apud [Dhar et al. 2000]). Os Algoritmos Genéticos também não foram implementados no WEKA.

Este artigo apresenta a integração Algoritmos Genéticos(AG) à tarefa de Mineração de Dados e incorpora o algoritmo implementado à ferramenta Explorer Fuzzy Tree (EFT) dando origem a uma nova ferramenta denominada EPT (Explorer Pattern Tools). Experimentos foram realizados com o intuito de avaliar o algoritmo implementado através dos AG utilizando cinco bases de dados e os resultados foram comparados com os obtidos na utilização da lógica nebulosa.

Este artigo está dividido da seguinte maneira: na seção 2 é feita uma breve descrição sobre Mineração de Dados e o uso de Algoritmos Genéticos na descoberta de padrões; a seção 3 descreve os métodos utilizados no algoritmo implementado. A seção 4 descreve e discute os resultados obtidos através dos testes realizados. Finalmente, na seção 5 é feita uma conclusão e são discutidas novas direções para esta pesquisa, além de trabalhos futuros.

2. Algoritmos Genéticos aplicados à Mineração de Dados

A Mineração de Dados consiste na aplicação de algoritmos para a análise e descoberta de dados e na produção de padrões ou modelos a partir de grandes bases de informação [Fayyad et al. 1996]. Esta tarefa é uma etapa de um processo maior chamado de *Kno-*

wledge Data Discovery (KDD) que visa a obtenção de informação de alto nível a partir de dados brutos de uma base de dados. Algoritmos Evolucionários são basicamente algoritmos inspirados nos princípios da seleção natural e da genética natural [Freitas 2008]. Operadores genéticos de seleção natural como cruzamento ou recombinação *(crossover)*, elitismo, mutação *(mutation)* e o uso de uma função de adaptação *(fitness)* para gerar sucessivas gerações de soluções, são aplicados para se chegar à solução que, se não a ótima, é a solução próxima da ótima. Em um algoritmo genético, cada solução candidata é representada por um indivíduo (cromossomo), que é um ponto do espaço de busca dentre todas as soluções possíveis de um problema [Carvalho 2005]. A representação cromossomial deve representar adequadamente o problema em questão, traduzindo suas informações para uma maneira viável a ser tratada por um computador, já que a adaptação destas informações está fortemente ligada à qualidade das soluções obtidas [Linden 2006]. Os operadores genéticos são brevemente apresentados a seguir.

2.1. Operador de Seleção

Este operador seleciona cromossomos da população para reprodução e favorece os cromossomos com maior *fitness*, ou seja, quanto melhor a avaliação do cromossomo, feita pela a função de avaliação, mais vezes este cromossomo pode ser selecionado para se reproduzir [Mitchell 1996].

Este método simula o mecanismo de seleção natural, onde os pais mais capazes geram mais filhos e pais menos aptos podem gerar menos descendentes [Linden 2006]. É importante notar que os indivíduos menos aptos não podem ser totalmente descartados, a fim de evitar a convergência genética - onde os indivíduos se tornam cada vez mais semelhantes, o que destrói a diversidade da população, comprometendo a evolução.

2.2. Elitismo

A cada geração G de um Algoritmo Genético, os indivíduos resultantes da geração $G - 1$ são todos descartados, dando lugar a novos indivíduos resultantes da aplicação dos operadores de seleção, cruzamento e/ou mutação aos indivíduos da geração $G - 1$.

A estratégia elitista visa preservar indivíduos com altos valores de *fitness*, ou seja, as melhores soluções encontradas na geração corrente, por mais de uma geração, copiando-os para a geração seguinte [Pappa 2002]. Dada uma população com N indivíduos, são escolhidos N_{elit} indivíduos com os melhores *fitness*. Estes indivíduos são copiados integralmente para a população da próxima geração do AG. Os outros $N - N_{elit}$ indivíduos da nova população são gerados a partir da aplicação dos operadores genéticos citados acima. O número N_{elit} é chamado de fator de elitismo, e é um número, definido pelo usuário e geralmente pequeno [Carvalho 2005].

2.3. Cruzamento ou Recombinação *(Crossover)*

O processo de criação de um novo cromossomo através da combinação de um ou mais cromossomos de alto desempenho (cromossomo que obteve uma alta nota da função de avaliação) é conhecido como recombinação ou cruzamento [Cox 2005]. A principal função deste operador é assegurar a troca de material genético entre dois indivíduos chamados pais, combinando informações de maneira que haja uma probabilidade razoável que os indivíduos resultantes deste cruzamento sejam melhores que os pais [Pappa 2002].

Graças ao operador de recombinação e a função de avaliação, os algoritmos genéticos são classificados como uma busca dirigida, já que utiliza a seleção para determinar as áreas mais promissoras de pesquisa e a recombinação para combiná-las de modo a gerar soluções mais aptas para resolução do problema em questão [Linden 2006].

2.4. Operador de Mutação

Este operador troca randomicamente alguns genes de um cromossomo. Por exemplo, a seqüência 00000100 poderia ser modificada em sua segunda posição, transformando-se na seqüência 01000100. A mutação pode ocorrer em cada gene de um cromossomo com uma probabilidade informada pelo usuário, ou baseada em algum critério previamente definido [Mitchell 1996].

O propósito do operador de mutação é manter a diversidade da população e assegurar que o cromossomo sempre cobrirá uma parte suficientemente grande do espaço de busca [Pappa 2002], introduzindo material genético que não está presente em nenhum outro indivíduo da população, ao contrário do operador de *crossover* [Carvalho 2005].

A mutação diversifica a população e combate as regiões de mínimos e máximos locais, assegurando o surgimento de novas soluções potenciais, independentes dos cromossomos já existentes [Cox 2005].

3. Algoritmo Implementado

Neste trabalho, Algoritmos Genéticos foram utilizados para a análise dos dados e posterior produção de padrões gerando regras de classificação do tipo: $A \longrightarrow C$ (lê-se: Se A então C), onde C (a parte conseqüente da regra) representa o atributo objetivo (classe) e A (a parte antecedente da regra) representa um conjunto de valores tomados por atributos, geralmente representados por uma conjunção. Este formato foi escolhido por ser intuitivo e facilmente compreendido pelo usuário.

A utilização dos algoritmos genéticos para a tarefa de descoberta de regras de previsão se deve a busca global realizada por esta abordagem. Outro fator determinante para a obtenção de bons resultados na aplicação de algoritmos genéticos a esta tarefa é a boa interatividade com os atributos inerente aos processos de seleção e cruzamento providos pelos algoritmos genéticos, ou seja, a cada geração, novos indivíduos surgem cobrindo atributos diferentes (mutação) ou permitindo a união de diferentes atributos em um único indivíduo (cruzamento). Essa interatividade é maior que a interatividade oferecida pelos algoritmos freqüentemente utilizados para Mineração de Dados, geralmente baseados na estratégia gulosa [Freitas 2002b] apud [Dhar et al. 2000].

Para obtenção das regras de classificação, o algorimo divide a base de dados a ser analisada em 2 subconjuntos distintos: o conjunto de teste e o conjunto de treinamento. A partir do conjunto de treinamento, o algoritmo produz regras de classificação e utiliza o conjunto de teste para verificar o grau de acurácia das regras concebidas.

O algoritmo proposto é uma combinação dos diversos operadores e técnicas comumente utilizadas para a tarefa de Mineração de Dados. A escolha de determinados operadores e técnicas foi feita baseada em testes e pesquisas em busca de melhores taxas de acurácia.

A discretização dos valores contínuos foi feita utilizando o método de intervalos iguais (*equal-width discretization*). Este método divide o conjunto de valores reais de um atributo em N intervalos de tamanhos iguais onde X e Y são respectivamente os limites inferior e superior dos valores do atributo a ser discretizado. Assim, o tamanho deste intervalo é dado por $(Y - X)/N$, onde N é a quantidade de intervalos que devem ser gerados pela discretização. Na ferramenta EPT este valor é informado pelo usuário. Este método foi escolhido por ser um método simples de discretização.

O número N_{elit} utilizado para este algoritmo foi 1 (a cada geração o melhor indivíduo é copiado integralmente para a geração posterior). Foi aplicado o operador de mutação simples, que pode mudar cada um dos genes de um indivíduo com uma probabilidade de $1/L$ onde L é o tamanho da cadeia de genes do indivíduo.

3.1. Codificação do Indivíduo

Algoritmos genéticos para descoberta de regras podem ser divididos em duas grandes abordagens, que diferem entre si na maneira como elas representam as regras de um indivíduo da população de soluções: a abordagem de Michigan e a abordagem de Pittsburgh [Freitas 2002a].

Na abordagem de Michigan, cada indivíduo representa uma única regra, enquanto que para a abordagem de Pittsburgh, cada individuo representa um conjunto de regras [Freitas 2002b]. Cada abordagem tem sua aplicação em uma etapa diferente do processo de Mineração de Dados. A abordagem de Michigan é aplicada à descoberta de regras de previsão de uma maneira mais natural, pois nesta tarefa a avaliação de cada regra é mais interessante para o sucesso no alcance do objetivo. Se for avaliada cada regra individualmente, ao final do algoritmo genético as regras que possuem os melhores índices de previsão serão conhecidas. Para a tarefa de classificação de dados, a abordagem mais adequada dentre as duas é a de Pittsburgh, já que nesta abordagem são avaliados os conjuntos de regras. Os melhores conjuntos avaliados classificarão os dados de maneira mais satisfatória.

Para a Mineração de Dados a representação cromossomial é geralmente uma seqüência linear de condições de regras, onde usualmente cada condição é um par atributo-valor [Carvalho 2005]. Para o algoritmo implementado, foi escolhida a representação binária de tamanho fixo, onde cada indivíduo é formado por uma cadeia de bits que podem assumir os valores 0 ou 1. Esta representação foi escolhida pois além de ser uma representação compacta, facilita os operadores genéticos *crossover* e mutação [Freitas 2002b].

3.2. Operador de *Seeding*

Utilizado para definir a população inicial do algoritmo genético implementado, o operador de *seeding* introduz na população um novo indivíduo. O diferencial deste operador é que ele age como uma função e o novo indivíduo gerado cobre um exemplo pertencente a base de treinamento [Giordana and Neri 1995]. Este operador foi escolhido para garantir que os indivíduos da população do algoritmo implementado cubram pelo menos um exemplo do conjunto de treinamento.

3.3. Métodos para Manter a Diversidade de Regras

Para evitar a convergência primária em torno de um super indivíduo, foi introduzido no algoritmo implementado um método de nicho *(Niching)*. Neste método o processo seletivo natural é simulado e cada espécie evolui em nichos ecologicamente separados e consomem os recursos oferecidos neste nicho. Isso permite primeiramente que diversos conceitos sejam aprendidos simultaneamente, além de permitir que os recursos computacionais sejam efetivamente explorados, evitando replicações desnecessárias e redundâncias [Giordana and Neri 1995].

Esses nichos são implementados através do operador de seleção chamado Sufrágio Universal *(Universal Suffrage)*. Neste operador, cada geração do algoritmo genético é selecionado de maneira aleatória e com reposição um subconjunto de exemplos do conjunto de treinamento E. Para cada exemplo selecionado são escolhidos regras candidatas a receberem o voto deste exemplo. Os candidatos são armazenados no conjunto $B(t)$, e o indivíduo vencedor da eleição é escolhido utilizando um método de seleção que favorece indivíduos mais aptos (indivíduos com maiores avaliações).

Apenas os exemplos pertencentes ao conjunto $B(t)$ são selecionados para cruzamento, e é esse aspecto que garante que apenas fórmulas que cobrem o mesmo exemplo sejam competidoras entre si. Se algum exemplo escolhido não possuir candidatos, um novo indivíduo que o cobre é introduzido na população através de um operador de *Seeding*.

3.4. Função de Avaliação

A função de avaliação deve traduzir o problema a ser resolvido para que possa ser compreendido e tratado por um computador [Linden 2006]. Para a descoberta de regras de classificação, a função de avaliação deve ser capaz de medir as seguintes características: a acurácia preeditiva da regra, ou seja, o poder desta regra de prever comportamentos baseada nas informações cedidas pelo conjunto de treinamento; o quanto esta regra é compreensível ao usuário, pois regras que não são claras ou que são muito complexas serão sumariamente descartadas pelo usuário; e o quanto esta regra é interessante, sendo esta a característica mais difícil de ser medida, por ser extremamente subjetiva [Freitas 2002b].

A função de avaliação do algoritmo implementado utiliza a acurácia preditiva e a completude para avaliação das regras obtidas. Para realizar esta avaliação, o algoritmo utiliza como parâmetros basicamente quatro conceitos (VP, FP, VN, FN) que são observados ao utilizar uma regra para classificar um exemplo da base de dados a partir da classe que foi prevista pela regra e da classe efetiva deste exemplo, obtidas a partir dos dados provenientes da base de dados utilizada.

Verdadeiros Positivos (VP) : denota o número de exemplos que atendem a parte antecedente da regra (a parte A) e a parte conseqüente da regra (a parte representada por C);

Falsos Positivos (FP) : denota o número de exemplos que atendem a parte antecedente da regra (a parte A) e não atendem a parte conseqüente da regra (a parte representada por C);

Falsos Negativos (FN) : denota o número de exemplos que não atendem a parte antecedente da regra (a parte A), porém atendem a parte conseqüente da regra (a parte representada por C);

Verdadeiros Negativos (VN) : denota o número de exemplos que não atendem a parte antecedente da regra (a parte A) e não atendem a parte conseqüente da regra (a parte representada por C);

A acurácia preditiva de uma regra de classificação pode ser resumida a uma matriz 2 x 2 chamada de matriz de confusão. Seu conteúdo é representado a partir dos conceitos mostrados acima, e possui o formato descrito na (Figure 1).

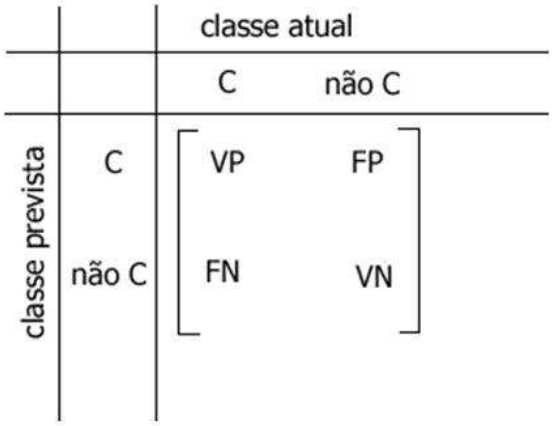

Figura 1. Matriz de Confusão para uma Regra de Classificação

A acurácia preditiva AC pode também ser obtida utilizando a seguinte formula:

$$AC = \frac{|VP|}{|VP+FP|}$$

Outro conceito importante para a avaliação de regras de classificação é o conceito da completude, representado por *Comp*, que representa o número de exemplos da base de dados que possuem o mesmo valor do atributo objetivo que a regra a ser avaliada e que são cobertos pela regra antecedente. A completude de uma regra pode ser avaliada através da fórmula:

$$Comp = \frac{|VP|}{|VP+FN|}$$

Baseado nos conceitos supracitados, a função de avaliação (*fitness*) utilizada neste trabalho foi:

$$Fitness = ACXComp$$

3.5. Operador de Cruzamento *Crossover*

O operador de cruzamento utilizado para este trabalho foi o *crossover* uniforme. Este tipo de *crossover* foi escolhido por ser capaz de combinar todo e qualquer esquema existente na população [Linden 2006]. O *crossover* uniforme é mais poderoso para algoritmos genéticos que utilizam a representação cromossomial binária, já que cria soluções que os outros tipos de *crossover* podem criar, além de criar combinações que seriam impossíveis em outros tipos de *crossover*.

4. Testes Realizados

Para avaliar o desempenho do algoritmo genético implementado, foram realizados experimentos em quatro bases de dados, onde se avaliou a acurácia das regras encontradas. Foram feitos também experimentos para a comparação do desempenho do algoritmo implementado, baseado na computação evolutiva, com os algoritmos implementados para a ferramenta EFT, baseados em árvores nebulosas e clássicas, e usados para descoberta de padrões em bases de dados. Os experimentos foram realizados utilizando o método de validação cruzada (*cross validation*) sobre quatro bases de dados.

4.1. Bases de Dados Utilizadas

- Iris (FISHER, 1988) - esta base de dados é utilizada para ilustrar análise discriminante [Souza and Claro 2008]. A base de dados Iris tem ao todo 150 exemplos, que descrevem três tipos de flores: Iris-setosa, Iris-versicolor e iris-virginica. Cada objeto desta base possui quatro atributos reais (numéricos) e um atributo nominal, o atributo *class* classifica cada objeto como um dos três tipos de flores descrito acima. Cada tipo de flor é a classe de 50 objetos cada.
- Segment-Challenge (GROUP, 1990a) - esta base de dados foi obtida a partir da seleção aleatória de objetos de uma base de dados de sete imagens de outdoors segmentadas para que seja possível a classificar todos os pixels. O tamanho desta base de dados é de 1500 objetos, e cada objeto possui 20 atributos, sendo 19 atributos reais (numéricos) e mais um atributo nominal, chamado de class e que possui o seguinte domínio: brickface, sky, foliage, cement, window, path, e grass.
- Segment-Test (GROUP, 1990a) - esta base de dados é um subconjunto da base de dados Segment-Challenge com 810 objetos.
- SPAMBASE (HOPKINS et al., 2001) - base codificada de e-mails construída por pesquisadores da HP [Souza and Claro 2008]. Possui 4601 e-mails pessoais cedidos pelo pesquisador George Forman, codificados em 58 atributos - 57 atributos de domínio real (numéricos) e um atributo binário, o atributo *spam* (o valor 1 denota que o objeto não é um *spam* o valor 0 denota que este objeto trata-se de um *spam*).

4.2. Procedimentos Experimentais

Foramm realizados dois experimentos: o primeiro experimento visa comparar os resultados obtidos pelo algoritmo com os valores obtidos nos testes realizados em [Souza and Claro 2008], com o objetivo de realizar uma comparação entre os índices de acurácia obtidos pelo algoritmo implementado com os índices obtidos pelos algoritmos baseados em árvores clássicas e nebulosas. No segundo experimento, o algoritmo genético foi executado utilizando diferentes combinações de tamanho de população, número de gerações executadas e tamanho de intervalos utilizados para classificação, com o objetivo de realizar um estudo da relação entre o número de gerações implementadas e as taxas de acurácia resultantes.

O primeiro experimento foi feito sobre todas as bases de dados mostradas anteriormente. Para este experimento foram utilizadas duas modalidades de teste: divisão por porcentagem *(percentage split)* e validação cruzada *(cross validation)*. Para a divisão por porcentagem, foram feitas 300 execuções do algoritmo implementado, nas quais foram utilizados 60% do conjunto de dados respectivo para treinar o algoritmo e os outros

40% restantes foram utilizados para validação das regras encontradas. Para a validação cruzada, as bases de dados foram divididas em 10 *folds*.

O segundo experimento executou o algoritmo 300 vezes utilizando a validação cruzada, dividindo a base de dados Spambase em 10 *folds*, obtendo e comparando os resultados.

4.3. Resultados e Discussões

A figura 2 apresenta a média aritmética para os resultados obtidos nos testes com o algoritmo genético, onde a coluna AG mostra os resultados obtidos pelo algoritmo genético implementado: V.C. representa os resultados obtidos para os testes que utilizaram a validação cruzada e D.P. representa os resultados obtidos pelos testes utilizando o método de divisão por porcentagem. As colunas seguintes representam os resultados obtidos em [Souza and Claro 2008].

Conjuntos de Dados	AG		FILMJ47	FILMJ48	FJ47	FJ48
	V.C.	D.P.				
SPAMBASE	0,520	0,622	0,853	0,857	0,614	0,645
Segment-challenge	0,820	0,832	0,874	0,874	0,838	0,841
Segment-test	0,830	0,845	0,847	0,835	0,790	0,798
Iris	0,726	0,784	0,869	0,907	0,803	0,837

Figura 2. Resultados....

O primeiro experimento mostrou taxas de acurácia maiores que 60% em todas as bases para o método de divisão por porcentagem, porém apenas para a base de dados Segment-test a taxa de acurácia foi superior às taxas obtidas pelos outros algoritmos. Este desempenho inferior é explicado pelo caráter heurístico dos algoritmos genéticos, além da precisão obtida pelos algoritmos de árvore clássicas e fuzzy ao trabalharem com atributos contínuos e que não é possível no algoritmo implementado já que este algoritmo utilizou discretizações simples para trabalhar com estes valores. Os valores de acurácia também poderiam ser modificados caso outros operadores fossem utilizados neste algoritmo ou caso fosse utilizado um método de discretização mais preciso.

Os resultados obtidos no segundo experimento podem ser mostrados através da figura 3. Este experimento mostrou que o índice de acurácia obtido pelo algoritmo implementado é aumentado à medida que o número de gerações também é aumentado.

5. Conclusão

Neste artigo foi descrita a implementação de uma algoritmo genético com o fim de realizar mineração de dados utilizando técnicas que, segundo a literatura pesquisada, maximizavam o desempenho de um algoritmo genético para encontrar regras de predição. O algoritmo implementado obteve acurácia superior a acurácia obtidas por técnicas de árvores clássicas apenas em uma base de dados. Este algoritmo obteve taxas de acurácia menores que todos os métodos de árvores fuzzy implementados em [Souza and Claro 2008]. Métodos de discretização mais eficientes, utilizando a interpolação linear, maximizariam o desempenho do algoritmo implementado, e podem ser agregados ao algoritmo como um trabalho futuro.

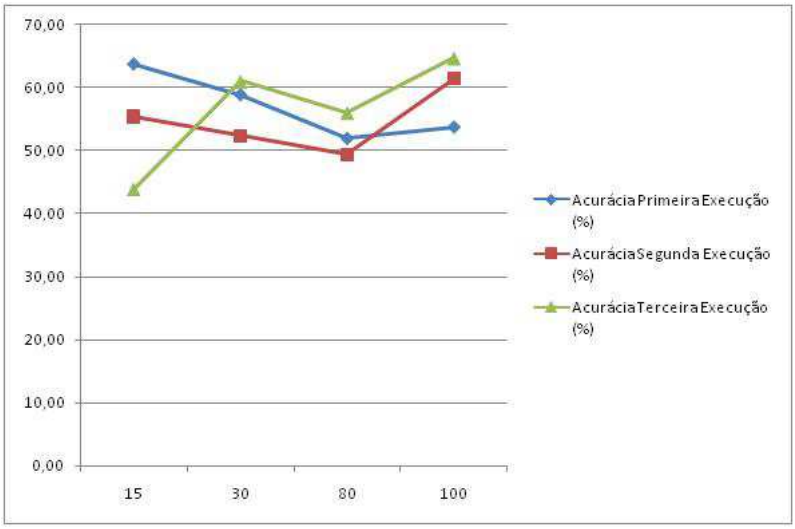

Figura 3. Acurácia X número de gerações

Outro trabalho futuro que pode ser implementado é a implementação de métodos de seleção de atributos para diminuir o tempo de execução dos algoritmos implementados além da implementação de um algoritmo genético que proponha uma função de avaliação, um modo de seleção de indivíduos, um modo de mutação diferentes do utilizado pelo algoritmo implementado neste trabalho.

Referências

Carvalho, D. R. (2005). *Árvore de Decisão / Algoritmo Genético para Tratar o Problema de Pequenos Disjuntos em Classificação de Dados*. PhD thesis, Programas de Pós-Graduação em Computação de Alto Desempenho/Sistemas Computacionais e Programa de Engenharia Civil da Universidade Federal do Rio de Janeiro.

Cox, E. (2005). *Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration*. Elsevier/Morgan Kaufmann.

Dhar, V., Chou, D., and Provost, F. (2000). Discovering interesting patterns for investment decision making with glower ◯-a genetic learner overlaid with entropy reduction. *Data Min. Knowl. Discov.*, 4(4):251–280.

Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54.

Freitas, A. A. (2002a). *Data Mining and Knowledge Discovery with Evolutionary Algorithms*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Freitas, A. A. (2002b). A survey of evolutionary algorithms for data mining and knowledge discovery. In *Advances in Evolutionary Computation*, pages 819–845. Springer-Verlag.

Freitas, A. A. (2008). A review of evolutionary algorithms for data mining. In *Soft Computing for Knowledge Discovery and Data Mining*, pages 79–111. Springer.

Giordana, A. and Neri, F. (1995). Search-intensive concept induction. *Evolutionary Computation, MIT Press*.

Linden, R. (2006). *Algoritmos Genéticos. Uma importante ferramenta da Inteligência Computacional*. Brasport, Rio de Janeiro, RJ, BR.

Mitchell, M. (1996). *An introduction to genetic algorithms*. MIT Press, Cambridge, MA, USA.

Pappa, G. L. (2002). Seleção de atributos utilizando algoritmos genéticos multiobjetivos. *Dissertação (Mestrado em Informática Aplicada) - Programa de Pós-Graduação em Informática Aplicada da Pontifícia Universidade Católica do Paraná*.

Souza, E. N. P. and Claro, D. B. (2008). Explorer fuzzy tree: uma ferramenta para experimentação de técnicas de classificação baseadas em árvores de decisão fuzzy. *ERBASE 2008 - WTICG-BASE (Workshop de Trabalhos de Iniciação Científica e Graduação Bahia, Alagoas e Sergipe)*.

Witten, I. H. and Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, CA, USA.