UNIVERSIDADE FEDERAL DO MARANHÃO
Curso de Ciência da Computação

Eduardo Roger Silva Nascimento

Aplicação de Algoritmos Genéticos Em Classificação de Dados

São Luís - MA
2021

Eduardo Roger Silva Nascimento

Aplicação de Algoritmos Genéticos Em Classificação de Dados

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Ivo José da Cunha Serra

São Luís - MA

2021

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Nascimento, Eduardo Roger Silva.
Aplicação de Algoritmos Genéticos Em Classificação de Dados / Eduardo Roger Silva Nascimento. - 2021.
50 p.

Orientador(a): Ivo José da Cunha Serra.
Monografia (Graduação) - Curso de Ciência da Computação, Universidade Federal do Maranhão, São Luís, 2021.

1. Algoritmos Genéticos. 2. Classificação de dados. 3. Regras de Associação. I. Serra, Ivo José da Cunha. II. Título.

Eduardo Roger Silva Nascimento

Aplicação de Algoritmos Genéticos Em Classificação de Dados

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Trabalho aprovado em: / /

Prof. Dr. Ivo José da Cunha Serra
Orientador
Universidade Federal do Maranhão

Prof. Msc. Carlos Eduardo Portela Serra de Castro
Membro da Banca Examinadora
Universidade Federal do Maranhão

Prof. Dr. Tiago Bonini Borchartt
Membro da Banca Examinadora
Universidade Federal do Maranhão

São Luís - MA
2021

À Deus e minha família por possibilitarem essa conquista.

Agradecimentos

Em primeiro lugar agradeço a Deus que possibilitou o término desse trabalho.

A minha família, principalmente minha mãe Edilene, minha vó Jacilene, minha madrinha Soraia, minha tia Márcia e minha tia Leia pela criação, por tornarem a pessoa que sou hoje e também por nunca terem medido esforços para me proporcionar um ensino de qualidade durante todo o meu período escolar e acadêmico.

Ao meu orientador, professor Ivo, pela confiança e pelo trabalho que conduziu com paciência, empenho e dedicação.

Ao professor Geraldo e a professora Simara que através de suas orientações, tutorias, ensinamentos, colaborações, conselhos e trabalhos dados que hoje pude entender a importância no meu crescimento pessoal e acadêmico e por isso eu os agradeço bastante.

Aos meus amigos Anderson, Wesley e Erik, a galera do "rolê"e meus amigos do "True Friends", especialmente Lucas, Sky e Diego pela cumplicidade, pelo apoio dado, pela amizade, pelo companheirismo, pela diversão e por compartilharem comigo momentos de dificuldades e alegrias.

Aos amigos do PETComp e do Codebuilder, pela ajuda nos trabalhos complicados, pelos momentos de diversão e que contribuíram bastante na minha evolução.

Aos professores do curso de Ciência da Computação pelos seus ensinamentos, sempre disponíveis a compartilharem seus conhecimentos e que ajudaram a concluir este trabalho direta e indiretamente.

A todos que contribuíram de alguma forma para conclusão deste trabalho.

"Like the sun, I rise again."
(Faceless Void, Defense of Ancients 2)

Resumo

Este trabalho tem como objetivo apresentar um modelo de aplicação dos algoritmos genéticos no contexto da tarefa de classificação de dados. Dados são gerados todos os dias em larga escala e têm como consequência enormes bases de dados, então se faz necessário buscar e utilizar métodos quer auxiliem na extração de conhecimento desses dados e que permita classificar e tomar decisões, por exemplo. Nessa circunstância, este trabalho apresenta conceitos de algoritmos genéticos e sobre como eles podem ser uma ótima ferramenta para extrair regras de associação com a finalidade de classificar dados. São realizados dois estudos de caso a fim de extrair regras simples de fácil assimilação humana e assim permitir uma classificação de maneira bem simples. Os resultados se mostraram satisfatórios para bases de dados independente do seu tamanho, mas onde a classe alvo possui uma quantidade grande de instâncias, o que leva a concluir que os algoritmos genéticos podem ser boas ferramentas no cenário da classificação de dados.

Palavras-chaves: Algoritmos Genéticos, Classificação de dados, Regras de Associação.

Abstract

This work aims to present a model of application of genetic algorithms in the context of the task of data classification. Data is generated every day on a large scale and has the consequence of huge databases, so it is necessary to search and use methods that help in extracting knowledge from that data and that allows classifying and making decisions, for example. In this circumstance, this work presents concepts of genetic algorithms and how they can be a great tool to extract association rules in order to classify data. Two case studies are carried out in order to extract simple rules of easy human assimilation and thus allow a very simple classification. The results were satisfactory for datasets regardless of their size, but where the target class has a large number of instances, which leads to the conclusion that genetic algorithms can be good tools in the scenario of data classification.

Keywords: Genetic Algorithms, Data Classification, Association Rules.

Lista de ilustrações

Figura 1 – Pseudocódigo de um algoritmo genético. 20
Figura 2 – As diferentes formas da operação de cruzamento 25
Figura 3 – Efeito da mutação em um cromossomo. 26
Figura 4 – Representação binária do cromossomo. 31
Figura 5 – Fluxograma do Algoritmo genético utilizado nos experimentos. 38
Figura 6 – Exemplo de conversão da matriz binaria para uma com os valores categóricos da regra. 40

Lista de tabelas

Tabela 1 – Exemplo de Transações de cestas de compras. 29
Tabela 2 – Distribuição das instâncias em número e porcentagem das classes da base de dados de classificação de automóveis. 33
Tabela 3 – Visão parcial da base de dados *Classificação de Automóveis* 34
Tabela 4 – Distribuição das instâncias por classe em número e porcentagem da base de dados *Golf-Weather* 34
Tabela 5 – Base de dados *Golf-Weather* 35
Tabela 6 – Visão parcial da base de dados da classificação de automóveis 36
Tabela 7 – Visão parcial da tabela gerada no final do estudo de caso na classe *muito bom* da base de dados *Classificação de Automóveis* ordenada pelo número de ocorrências. 42
Tabela 8 – Visão parcial da tabela gerada no final do estudo de caso na classe *muito bom* da base de dados *Classificação de automóveis* ordenada pelo grau de confiança. 43
Tabela 9 – Visão parcial da tabela gerada no final do estudo de caso na classe *inaceitável* da base de dados *Classificação de automóveis* ordenada pelo número de ocorrências. 44
Tabela 10 – Visão parcial da tabela gerada no final do estudo de caso na classe *inaceitável* da base de dados *Classificação de automóveis* ordenada pelo grau de confiança. 44
Tabela 11 – Tabela gerada no final do experimento da base de dados *Golf-Weather* ordenada pelo número de ocorrências. 45

Lista de abreviaturas e siglas

AG	Algoritmos Genéticos

Sumário

1 INTRODUÇÃO 14
1.1 Contextualização 14
1.2 Objetivos 15
1.3 Organização do Trabalho 15

2 ALGORITMOS GENÉTICOS 17
2.1 Teoria da Evolução 17
2.2 Computação Evolutiva 18
2.3 Algoritmos Genéticos 19
2.3.1 Representação do indivíduo 20
2.3.2 Inicialização da População 21
2.3.3 Função de aptidão (*Fitness*) 22
2.3.4 Operadores genéticos 22
2.3.4.1 Seleção 23
2.3.4.1.1 Método da roleta 23
2.3.4.1.2 Elitismo 23
2.3.4.1.3 Torneio 24
2.3.4.2 Cruzamento (*Crossover*) 24
2.3.4.2.1 *Crossover* de um ponto 24
2.3.4.2.2 *Crossover* de dois pontos 25
2.3.4.2.3 *Crossover* uniforme 25
2.3.4.3 Mutação 25
2.3.5 Critério de parada 26

3 ALGORITMOS GENÉTICOS E REGRAS DE ASSOCIAÇÃO NO CONTEXTO DA CLASSIFICAÇÃO DE DADOS 28
3.1 Classificação de Dados 28
3.2 Extração de regras de associação 29
3.2.1 Suporte e Confiança 29
3.2.2 Regras do tipo $SE \rightarrow ENTÃO$ 30
3.3 Um modelo de Algoritmo Genético para classificar dados 31

4 ESTUDOS DE CASO: CLASSIFICAÇÃO DE DADOS COM ALGORITMOS GENÉTICOS 33
4.1 As bases de dados utilizadas 33
4.1.1 Base de dados *Classificação de Automóveis* 33

4.1.2 Base de dados *Golf-Weather* 34
4.2 População e representação do indivíduo no estudo de caso **35**
4.3 Função de aptidão utilizada **36**
4.4 Operadores genéticos **37**
4.5 Ferramentas utilizadas para construção do algoritmo genético . . . **37**
4.6 Experimentos realizados **37**
4.6.1 Estudo de Caso I: *Classificação de Automóveis* 38
4.6.2 Estudo de Caso II: Base de dados *Golf-Weather* 41

5 RESULTADOS E DISCUSSÃO **42**

6 CONCLUSÃO **47**

REFERÊNCIAS **48**

1 Introdução

1.1 Contextualização

Os algoritmos genéticos (AG) são métodos de busca adaptativos muito eficientes para resolver problemas de busca e otimização encontrando soluções ótimas, ou aproximadamente ótimas em uma grande variedade de problemas (FERNANDES, 2003). Eles não impõem muitas das limitações encontradas nos métodos de busca tradicionais que exigem maior quantidade de memória requerida pelos algoritmos, maior espaço necessário para entradas e saídas e um maior tempo de execução para que eles encontrem a melhor resposta. Zuben (2000) enfatizam em seu trabalho que os algoritmos genéticos constituem uma classe de métodos de busca de propósito geral que apresentam um balanço notável entre aproveitamento de melhores soluções e exploração do espaço de busca.

A quantidade de informações disponíveis no meio digital superou a capacidade humana de entendimento. Cada vez mais dados são gerados seja na web, diagnósticos médicos, organizações e empresas. E surge a necessidade de extrair conhecimento desses dados a fim de encontrar padrões, organizá-los e classificá-los a fim de tomar decisões de forma automatizada. E uma das técnicas mais utilizadas para extrair informação de forma automática é a Mineração de Dados, definida por FAYYAD, PIATETSKY-SHAPIRO e SMYTH (1996) como um processo de Descoberta de Conhecimento que consiste na realização da análise dos dados e na aplicação de algoritmos de descoberta.

Dentro do contexto de mineração de dados, existe a classificação. A classificação de dados é o processo de organização de dados em categorias para o seu uso mais eficaz e eficiente, cujo objetivo é encontrar uma função que possa atribuir rótulos de classe a um determinado exemplo (AMARAL, 2016).

A classificação visa identificar a qual classe um determinado registro pertence. Geralmente, esta tarefa é aplicada por soluções que utilizam algoritmos de software com base em palavras-chave ou frases no conteúdo, onde um modelo analisa o conjunto de registros fornecidos, com cada registro já contendo a indicação à qual classe pertence, a fim de "aprender"como classificar um novo registro (CAMILO; SILVA, 2009).

Diante deste cenário, despertou o seguinte questionamento “como algoritmos genéticos podem ser aplicados no contexto da classificação de dados”. A partir dessa curiosidade o trabalho se propõe a buscar não apenas a resposta desse questionamento como também apresentar e aplicar um modelo de algoritmos genéticos na tarefa de classificação de dados.

Os principais trabalhos envolvendo aplicações de AG em classificação de dados que

vêm sendo descritos na literatura foram realizados por Tan et al. (2009), que descrevem em seu livro "Introdução ao Data Mining: Mineração de Dados" um modelo de algoritmos genéticos para minerar dados; por Han, Kamber e Pei (2006), onde propuseram um modelo de algoritmo genético capaz de classificar dados e por Morais, Domingues e Oliveira (2005), que apresentam e aplicam em seu trabalho algoritmos genéticos com a tarefa de classificar dados através de uma abordagem em que representa as instâncias de uma base de dados como indivíduos de uma população do algoritmo genético. E a partir destes indivíduos e aplicando os operadores genéticos obtiveram regras simples de classificação.

1.2 Objetivos

O objetivo principal deste trabalho é apresentar um modelo de aplicação dos algoritmos genéticos para a tarefa de classificação de dados e ilustrar sua aplicação por meios de dois estudos de caso.

Os objetivos específicos desse trabalho são:

- Apresentar um modelo de AG capaz de extrair regras simples de uma base de dados que possam classificar dados utilizando os algoritmos genéticos;
- Aplicar esse modelo de AG em duas bases de dados de tamanhos distintos a fim de verificar a viabilidade do modelo.
-

1.3 Organização do Trabalho

Este trabalho está subdividido em 6 capítulos. O presente capítulo contextualiza o problema, apresenta os trabalhos relacionados e por fim são evidenciados o objetivo geral e os específicos. Nos demais capítulos, o conteúdo está descrito a seguir de maneira sintetizada:

- Capítulo 2: Apresenta uma visão detalhada dos algoritmos genéticos.
- Capítulo 3: Apresenta uma visão geral da classificação de dados e é proposto um modelo de algoritmos genéticos para classificação.
- Capítulo 4: Aplica-se o modelo de algoritmo genético para classificação de dados em dois estudos de caso.
- Capítulo 5: Mostra o resultado obtido e é realizado uma discussão sobre este resultado.

- Capítulo 6: É apresentado a conclusão do trabalho e também sugestões para a realização de trabalhos futuros.

2 Algoritmos Genéticos

Neste capítulo são discutidos os principais fundamentos dos algoritmos genéticos para compreensão da discussão apresentada no decorrer do desenvolvimento do trabalho.

2.1 Teoria da Evolução

A teoria da evolução, publicada por Charles Darwin em 1859, afirma que é o ambiente que determina a importância da característica do indivíduo ou de suas variações e que os organismos mais bem adaptados a esse ambiente têm maiores chances de sobrevivência, deixando um número maior de descendentes (SILVA; PIGNATA,). Isso acontece devido a um "Poder predominante"entre as "causas da Mudança", como Darwin denomina de seleção natural (DARWIN, 1859 apud REGNER, 2002). Os organismos mais bem adaptados são selecionados (escolhidos) pelo ambiente e assim, ao longo das gerações, a atuação da seleção natural mantém ou melhora o grau de adaptação dos organismos, fixando suas características no ambiente(ZUBEN, 2000).

Para explicar o processo de seleção natural, Darwin definiu as seguintes hipóteses (DARWIN, 1859 apud ZUBEN, 2000) :

1. Os filhos tendem a ser em maior número que os pais;
2. O número de indivíduos de uma espécie permanece aproximadamente constante;
3. De (1) e (2), concluiu que vai haver uma luta pela sobrevivência;
4. Dentro de uma mesma espécie, os indivíduos apresentam pequenas diferenças, sendo que a maioria delas também está presente nos respectivos pais;

Zuben (2000) conclui que o princípio da seleção natural indica que os indivíduos cujas variações se adaptam melhor ao ambiente terão maior probabilidade de sobreviver e se reproduzir, porém, Darwin considerou estas hipóteses como suficientes para explicar a origem das espécies. E atualmente, elas são consideradas na explicação de processos adaptativos em condições e recursos limitados. Para transformar esta teoria em uma “teoria da origem das espécies”, três hipóteses vinculadas à genética foram adicionadas, conduzindo ao neodarwinismo:

1. Algum processo de variação continuada deve ser responsável pela introdução de novas informações junto à carga genética dos organismos;

2. Não há limite para a sucessão de variações que podem ocorrer;

3. A seleção natural é o mecanismo para preservação das novas informações que correspondam a uma maior adaptação.

Logo, a seleção natural é probabilística, e seu alvo primário é o indivíduo, embora seu efeito resultante vai se manifestar na espécie como um todo. A espécie é o beneficiário final do processo evolutivo (ZUBEN, 2000).

2.2 Computação Evolutiva

Nas décadas de 1950 e 1960, vários cientistas da computação estudaram de forma independente sistemas evolucionários com a ideia de que a evolução pode ser usada como uma ferramenta de otimização para problemas de engenharia. A ideia em todos esses sistemas era desenvolver uma população de soluções candidatas para um determinado problema, usando operadores inspirados por variação genética natural e seleção natural (MITCHELL, 2000).

Nesse contexto, surge a Computação Evolutiva, um ramo de pesquisa da Ciência da Computação que compreende um conjunto de técnicas de busca e otimização baseados nas teorias de Darwin. Mas por que usar a evolução como inspiração para resolver problemas computacionais? Isso se deve ao fato dos mecanismos de evolução parecem bem adequados para alguns dos problemas bem complexos, como problema de engenharia computacional de proteínas, em que um busca-se um algoritmo que pesquisará entre o vasto número de possíveis sequências de aminoácidos para uma proteína com propriedades especificadas ou a busca por um conjunto de regras ou equações que preverão os altos e quedas de um mercado financeiro, como o de moeda estrangeira (MITCHELL, 2000).

Mitchell (2000) argumenta que muitos problemas computacionais requerem a busca por um grande número de possibilidades de soluções. Esses problemas de pesquisa muitas vezes podem se beneficiar de um uso efetivo do paralelismo computacional, em que muitas possibilidades diferentes são exploradas simultaneamente em uma forma eficiente, então seria necessário utilizar uma estratégia inteligente para escolher qual próximo conjunto de possíveis soluções para serem avaliadas.

Outra justificativa apresentada por Mitchell (2000), é a capacidade de adaptação. Muitos problemas computacionais exigem que um programa de computador seja adaptativo para continuar a ter um bom desempenho em um ambiente em mudança.

Na computação evolutiva, as regras são normalmente seleção (“seleção natural”) com variação devido ao cruzamento e / ou mutação (conceitos que serão mais detalhados na seção 2.3.4) e o comportamento que se espera é o projeto de soluções de alta qualidade

para problemas difíceis e a capacidade de adaptar essas soluções em um ambiente de mudança.

As principais abordagens da computação evolutiva são algoritmos genéticos, estratégias evolutivas e programação evolutiva. Como o foco desse trabalho é aplicação dos AG, os outros não serão abordados.

2.3 Algoritmos Genéticos

Para Fernandes (2003), algoritmos genéticos são métodos adaptativos que podem ser usados para resolver problemas de busca e otimização, inspirados no processo genético e evolutivo dos organismos vivos.

Proposto por John Holland, os algoritmos genéticos empregam uma terminologia originada da teoria da evolução natural e da genética. Um indivíduo da população é representado por um único cromossomo, o qual contém a codificação (genótipo) de uma possível solução do problema (fenótipo). Cromossomos são usualmente implementados na forma de listas de atributos ou vetores, onde cada atributo é conhecido como gene. Os possíveis valores que um determinado gene pode assumir são denominados alelos (HOLLAND, 1975 apud ZUBEN, 2000).

Zuben (2000) enfatiza que os algoritmos genéticos constituem uma classe de métodos de busca de propósito geral que apresentam um balanço notável entre aproveitamento de melhores soluções e exploração do espaço de busca. Embora apresentem etapas não-determinísticas em seu desenvolvimento, ele argumenta que os algoritmos genéticos não são métodos de busca puramente aleatórios, pois possuem propriedades como a combinação de variações aleatórias com a seleção e pelos valores de aptidão (*fitness*) atribuído a cada indivíduo.

Outra propriedade importante dos algoritmos genéticos (assim como de todos os algoritmos evolutivos) destacada por Zuben (2000) é que os AG mantêm uma população de soluções candidatas. O processo de busca é realizado através da conservação de possíveis soluções candidatas e da troca de informação entre elas. A cada iteração, uma nova geração é criada as soluções e as mais aptas se reproduzem enquanto as consideradas ruins são eliminadas.

Entretanto, como saber quem é apto e quem não é? Utiliza-se uma função que calcula a aptidão de cada uma, simulando o papel da pressão exercida pelo ambiente sobre o indivíduo na natureza, denominada função de *fitness* (ZUBEN, 2000).

Um AG para ser executado em um problema particular deve ter os seguintes componentes (MICHALEWICZ, 1996):

- Uma representação genética para as potenciais soluções do problema;
- Uma forma de criar a população inicial das soluções;
- Uma função que avalia cada solução calculando a aptidão dela em relação a resolução do problema;
- Operadores genéticos que alteram a composição das soluções e que formam a nova população (composta pelos filhos das soluções mais bem avaliadas).

O pseudocódigo de um AG padrão é representado pela Figura 1:

Figura 1 – Pseudocódigo de um algoritmo genético.

```
Início do Algoritmo

    Obter uma população inicial de forma aleatória

Enquanto não (pare) faça

    Selecionar pais da população;
    Produzir filhos a partir dos pais selecionados;
    Mutação dos filhos;
    Estender a população adicionando os filhos;
    Reduzir a população estendida;

Fim-enquanto

Fim do Algoritmo
```

Fonte – Fernandes (2003)

Inicia-se gerando uma população inicial de forma aleatória que é composta por indivíduos conhecidos com soluções candidatas. Então, entra-se em um laço de repetição cuja condição de parada geralmente é um número fixo de iterações ou quando ele converge para a solução do problema. A cada iteração, selecionam-se indivíduos baseados em uma aptidão e são produzidos novos filhos a partir do cruzamento dos indivíduos selecionados. Os filhos podem sofrer uma mutação em seu código genético e no final são adicionados a população que substituirá a geração anterior.

2.3.1 Representação do indivíduo

Os Algoritmos Genéticos são capazes de criar soluções para os problemas do mundo real. A evolução de tais soluções, bem como a obtenção de valores ótimos para o problema, dependem, em parte, de uma representação (também denominada codificação/decodificação) adequada para elas. Os indivíduos, componentes da população,

podem ser representados como um conjunto de parâmetros (denominados "genes"), que formam juntos uma coleção de valores (cromossomos) ou *strings*. AG simples normalmente trabalham essa representação de um indivíduo como descrição de entrada por um vetor binário de tamanho fixo (ZUBEN, 2000).

Michalewicz (1996) exemplifica como é feita a representação na seguinte *string* binária de comprimento 22:

$$(1000101110110101000111)_2$$

Ele utiliza a equação 2.1 para converter da base 2 para base 10 resultando em 0.637197.

$$(b_2 1 b_2 0 ... b_0)_2 = (\sum_{i=0}^{21} b_i \cdot 2^i)_{10} = x' \tag{2.1}$$

E para encontrar o correspondente número real, a fórmula 2.2:

$$x = -1.0 + x' \cdot \frac{3}{2^2 2 - 1} \tag{2.2}$$

A partir da equação 2.1 tem como resultado:

$$x' = (1000101110110101000111)_2 = 2288967$$

E por fim o resultado da equação 2.1 é utilizado na equação 2.2:

$$x = -1.0 + 2288967 \cdot \frac{3}{4194303} = 0.637197$$

A etapa de representação é muito importante na inicialização de um algoritmo genético, pois uma codificação inadequada pode levar a uma convergência prematura ou que indivíduos modificados por operadores genéticos se tornem inválidos. Os inválidos são aqueles que (ZUBEN, 2000):

- Possui o tamanho do seu cromossomo maior que o tamanho máximo permitido para cromossomos
- A representação binária ou qualquer outra de um indivíduo não ser equivalente à do valor original

2.3.2 Inicialização da População

Geralmente, a inicialização da população é realizada de maneira aleatória (BENTO; KAGAN, 2008). Entretanto, se existe um conhecimento prévio do problema, pode-se utilizá-lo na geração da população inicial. Zuben (2000) exemplifica no caso da codificação

binária, se é sabido que a solução final vai apresentar mais 0's do que 1's, então esta informação pode ser utilizada, mesmo que não se saiba exatamente a proporção.

Em problemas bem restritivos, é necessário um cuidado na hora de gerar a população com indivíduos inválidos como abordado na seção 2.3.1.

2.3.3 Função de aptidão (*Fitness*)

A adaptação de um indivíduo leva em conta o seu cromossomo e o valor resultado de uma função de aptidão ou função de *fitness*. Essa função deve ser projetada de maneira específica para cada problema. Dado um cromossomo, a função associa um número real, que supõe refletir o nível de adaptação do indivíduo, representado pelo cromossomo, ao problema (FERNANDES, 2003).

Michalewicz (1996) explica a função de aptidão da seguinte forma, para um vetor binário v, tem-se uma função equivalente f, tal que:

$$eval(v) = f(x),$$

onde v é o cromossomo que representa o valor real de x (uma possível solução). Isto é, um vetor binário é decodificado em um valor real x, o qual é aplicado em uma função f que retorna a aptidão dessa solução x, ou seja, o *fitness*, valor de quão boa ela é para resolver o problema. Por exemplo, dados três indivíduos(MICHALEWICZ, 1996):

$$x_1 = 0.12, x_2 = 0.13, x_3 = 1.57.$$

Essas soluções candidatas são codificadas em vetores binários, respectivamente:

$$v_1 = (111010001100), v_2 = (000001110110), v_3 = (100101010101).$$

Quando se calcula o *fitness* de cada um, tem-se:

$$eval(v_1) = f(x_1) = 1.678289, eval(v_2) = f(x_2) = 0.989877, eval(v_3) = f(x_3) = 2.562323.$$

Concluindo assim que o indivíduo v_3 é o melhor da população já que a função de aptidão retorna o maior valor para ele.

2.3.4 Operadores genéticos

Os operadores genéticos têm como objetivos a seleção, criação e modificação das soluções a fim de garantir que os melhores indivíduos tenham uma maior probabilidade de gerar descendentes. Os mais frequentemente utilizados em algoritmos genéticos são a seleção, cruzamento e a mutação. Nesta seção, apresenta-se os principais aspectos relacionados a estes operadores.

2.3.4.1 Seleção

Na natureza, tantos os indivíduos mais aptos quanto os menos geram filhos. A seleção consiste escolher esses indivíduos na população que gerará descendentes para o próximo geração e quantos descendentes cada um criará.

O objetivo da seleção é enfatizar os indivíduos mais aptos na população na esperança de que seus filhos tenham, por sua vez, uma aptidão ainda maior. Entretanto, os indivíduos menos aptos não devem ser desprezados. Para Mitchell (2000), a seleção deve ser equilibrada com a variação de cruzamento e mutação. Uma seleção muito forte significa que indivíduos ótimos com alta aptidão assumirão o controle da população, reduzindo a diversidade necessária para mais mudança e progresso, e uma seleção muito fraca resultará em uma evolução muito lenta. Alguns dos métodos mais utilizados de seleção são método da roleta, elitismo e torneio.

2.3.4.1.1 Método da roleta

O algoritmo genético clássico utiliza um esquema de seleção chamado método da roleta (*roulette wheel*), onde é atribuído uma probabilidade de um indivíduo ser selecionado e passar para a próxima geração o seu material genético. Essa probabilidade é proporcional ao fitness dele, sendo medido na função de aptidão e em relação à somatória do fitness de todos os indivíduos da população (MICHALEWICZ, 1996). Desse modo, tem maior probabilidade de passar para a próxima geração indivíduos que apresentem maior fitness, o que pode ocasionar um problema, o melhor indivíduo de uma geração será perdido, já que pode ocorrer de ele não ser o sorteado para a próxima geração.

2.3.4.1.2 Elitismo

A forma básica dos algoritmos genéticos descarta a geração anterior e considera para a futura apenas os descendentes obtidos. Existem técnicas que visam contornar essa situação e colocar na próxima geração indivíduos mais bem avaliados da geração anterior. Essa técnica é denominada elitismo. O método Elitista é uma técnica adicional a seleção e consiste em reintroduzir o indivíduo melhor avaliado de uma geração para a seguinte, evitando a perda de informações importantes presentes em indivíduos de alta avaliação e que podem ser perdidas durante os processos de seleção e cruzamento. Algumas técnicas controlam o número de vezes que o indivíduo pode ser reintroduzindo, o que contribui para evitar convergência a máximos locais. E que outra forma de implementar o elitismo seria acrescentar indivíduos considerados mais aptos ao final das operações de cruzamento e mutação, considerando assim para a geração futura os melhores entre os pais e descendentes. Esta técnica, naturalmente reintroduz os melhores avaliados na geração seguinte, indefinidamente (BENTO; KAGAN, 2008).

2.3.4.1.3 Torneio

Um outro método é a seleção por Torneio, onde um número n de indivíduos da população é escolhido aleatoriamente para formar uma subpopulação temporária. Deste grupo, o indivíduo selecionado dependerá de uma probabilidade k definida previamente, ela dirá se o melhor ou o pior será o selecionado dessa subpopulação temporária. Dois indivíduos são escolhidos aleatoriamente na população. Um número aleatório r é então escolhido entre 0 e 1. Se $r < k$ (onde k é um parâmetro fixo, por exemplo 0.75), então o mais apto dos dois indivíduos é selecionado para ser um pai; caso contrário, o indivíduo menos apto é selecionado. Então, os dois são devolvidos à população original e podem ser selecionados novamente (MITCHELL, 2000).

2.3.4.2 Cruzamento (*Crossover*)

Crossover ou cruzamento tem como objetivo a reprodução de novos indivíduos. Há uma troca de material genético entre os elementos selecionados. Este operador realiza uma exploração global, uma vez que os indivíduos que são produzidos são radicalmente diferentes para ambos os pais em, pelo menos, alguns lugares do seu vetor binário. Espera-se que as boas partes de ambas as soluções quando combinadas juntas gerem uma solução ainda melhor (MARSLAND, 2015).

Existe uma probabilidade chamada taxa de cruzamento para que seja aplicada aos indivíduos selecionados.Fernandes (2003) recomenda que essa taxa esteja entre 0.5 e 1. E no caso em que o operador não se aplica (quando a probabilidade gerada for menor que a taxa de cruzamento), a descendência se obtém simplesmente duplicando os pais.

Existem várias maneiras de implementar o *crossover*, entre elas, tem-se o *crossover* de um ponto, o *crossover* de dois pontos e o *crossover* uniforme. Zuben (2000) argumenta que não há nenhum operador de cruzamento que claramente apresente um desempenho superior aos demais e que cada operador é particularmente eficiente para uma determinada classe de problemas e extremamente ineficiente para outras.

Segue a descrição de cada um dos tipos citados para casos de codificação binária. Os operadores de *crossover* que vão ser descritos também podem ser utilizados em cromossomos com codificação em ponto flutuante. Um exemplo é o chamado *crossover* aritmético (MICHALEWICZ, 1996). Este operador é definido como uma combinação linear de dois vetores (cromossomos).

2.3.4.2.1 *Crossover* de um ponto

É o operador de *crossover* mais comumente empregado por causa de sua simplicidade. Para a aplicação deste operador, são selecionados dois indivíduos (pais) e a partir de seus cromossomos um novo indivíduo é gerado. Seleciona-se um mesmo ponto de corte

aleatoriamente nos cromossomos dos pais, e os segmentos de cromossomo criados a partir do ponto de corte são trocados, como mostrado na Figura 2a. Apesar de a figura mostrar apenas um filho, essa operação resulta em dois filhos, pois o restante que não foi utilizado na criação de um filho é utilizado para gerar o outro.

2.3.4.2.2 *Crossover* de dois pontos

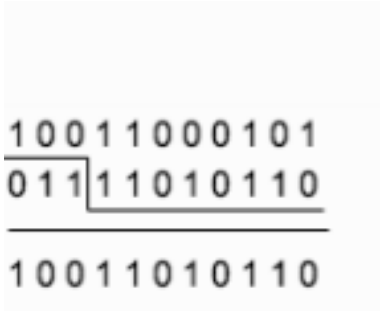
O *crossover* de dois pontos é realizado de maneira análoga ao *crossover* de um ponto. Difere apenas na quantidade de pontos sorteados, que são dois, e no número de partes geradas na divisão do cromossomo dos pais, que são três partes. E em seguida, são trocados para gerar dois filhos. A Figura 2b ilustra a geração de um filho por esta forma de cruzamento.

2.3.4.2.3 *Crossover* uniforme

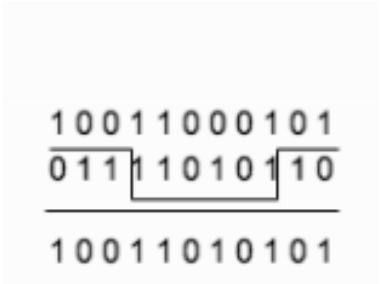
No *crossover* uniforme, para cada bit no primeiro filho é decidido (com alguma probabilidade fixa p) qual pai vai contribuir com seu valor para aquela posição. Como o *crossover* uniforme troca bits em vez de segmentos de bits (que aqui fazem o papel dos genes), ele pode combinar características independentemente da sua posição relativa no cromossomo (ZUBEN, 2000).

Se o valor sorteado for 1, o gene do primeiro pai é passado ao primeiro filho e o do segundo pai para o segundo filho. No caso contrário, o gene do primeiro pai é passado ao segundo filho e o do segundo pai para o primeiro filho. Vemos isso na Figura 2c.

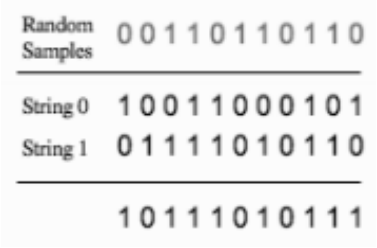
Figura 2 – As diferentes formas da operação de cruzamento



(a) *Crossover* de um ponto



(b) *Crossover* de dois pontos



(c) *Crossover* uniforme

Fonte – Marsland (2015)

2.3.4.3 Mutação

Embora a operação de cruzamento proporcionar uma exploração rápida no espaço de busca através da seleção dos melhores indivíduos e realizar o cruzamento entre eles,

tem-se o problema de ficar preso em um certo local no espaço de busca, gerando ótimas soluções locais, e no pior dos casos nunca convergir para solução global.

O problema de encorajar a convergência prematura é quando o algoritmo se estabelece em uma população constante que nunca muda, embora não tenha encontrado um ótimo. Isso ocorre porque os algoritmos genéticos favorecem membros mais aptos da população, que significa que uma solução que atinge um máximo local geralmente será favorecida e essa solução prevalecerá (MARSLAND, 2015).

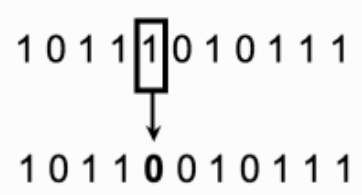
Para Marsland (2015), técnicas de seleção como torneios e elitismo encorajam isso, porque reduzem a quantidade de diversidade na população, permitindo que os mesmos indivíduos permaneçam por muitos gerações.

Isso significa que o aspecto de exploração dos Algoritmos Genéticos para de ocorrer, pois a exploração no espaço de busca será mínima, o que torna difícil escapar dessa região e sempre resulta na mesma solução local já que a maioria dos indivíduos têm pior aptidão e, portanto, serão repostos na população.

Para resolver essa questão, tem-se o operador de mutação. Ela visa dar diversidade a população, garantindo que nenhum ponto do espaço de busca tenha probabilidade zero para ser examinado, mas também que a busca não se torna essencialmente aleatória e assim assegurar a convergência do algoritmo genético (FERNANDES, 2003). A mutação garante que a população fique contra a fixação permanente em qualquer região particular do espaço de busca (MITCHELL, 2000).

Fernandes (2003) explica que o operador é aplicado a cada filho de maneira individual e consiste em uma alteração, de maneira aleatória (normalmente com probabilidade bem pequena), de cada gene componente do cromossomo. A Figura 3 mostra a mutação sendo aplicada no quinto gene do cromossomo de um indivíduo.

Figura 3 – Efeito da mutação em um cromossomo.



Fonte – Marsland (2015)

2.3.5 Critério de parada

O objetivo ideal seria os algoritmos genéticos encontrarem a solução correta para o problema. Entretanto, a não ser que se saiba previamente a solução ótima, não se pode ter

certeza se a solução apresentada é a melhor solução. Os parâmetros que mais são utilizados como critério de parada nos algoritmos genéticos são (OTERO, 2016):

- Número de gerações: representa o número máximo de ciclos de evolução de um Algorimo Genético, o qual é um dos critérios mais adotados. Valores grandes podem demandar um tempo maior no processamento, mas possibilitam uma busca maior no espaço de soluções. Um valor pequeno pode ocasionar uma convergência prematura para uma solução local.
- Convergência da Função de Aptidão: representa a ideia de estagnação do algoritmo, ou seja, quando não ocorre melhoria significativa na solução durante um dado número de gerações, o algoritmo de processamento é automaticamente interrompido. É útil para evitar que o processamento consuma tempos computacionais elevados.
- Número de rodadas: representa o número total de vezes que um AG será executado. O AG é uma técnica probabilística, sendo que, em alguns casos, pode ser interessante executá-lo mais de uma vez, a fim de se tentar obter melhores soluções. As novas rodadas não estão presas aos ótimos locais encontrados em rodadas anteriores. Dessa forma, serão avaliados outros pontos do espaço de busca, os quais poderão implicar na obtenção de melhores soluções.

No capítulo é apresentado um experimento que ilustra o critério de parada de número de rodadas, onde cada rodada é finalizada pelo número de gerações.

3 Algoritmos Genéticos e Regras de Associação no contexto da Classificação de Dados

Este capítulo aborda sobre como os algoritmos genéticos podem ser utilizados para descoberta de conhecimento no contexto da classificação. Inicia-se descrevendo as regras de associação, o qual é bastante útil para descobrir relações interessantes escondidas em grandes bases de dados e classificá-los. Estas regras devem ser fáceis para o entendimento humano . E por fim, relaciona as regras de associação e algoritmos genéticos, onde é apresentado um modelo de Algoritmos Genéticos capaz de extrair regras e a partir delas, induzir regras de classificação simples para os usuários.

3.1 Classificação de Dados

Para Amaral (2016), a classificação de dados é uma das tarefas de mineração de dados. Ele explica que a mineração de dados são processos para explorar e analisar grandes volumes de dados em busca de padrões, previsões, erros, associações a fim de classificar, agrupar e tomar decisões de forma automatizada.

Segundo SCHUNEIDER (2002), a tarefa de classificação tem como objetivo examinar as características de um objeto e enquadrá-las em conjuntos pré-definidos. Consiste na generalização e especialização de dados que servem para distinguir as classes de modo a predizer dados ou classes de registros não classificados automaticamente. Faz-se todo um processo analítico utilizando técnicas estatísticas e matemáticas para examinar a base de dados.

A classificação preveem com uma determinada precisão se algo pertence ou não a uma classe, porém, a maioria, senão todos, os algoritmos de classificação tem como limitação em explicar o porquê daquela classe é a prevista, pois não deixam evidente o motivo pelo qual uma instância é associada a uma determinada classe. E uma das formas de solucionar essa questão, é a utilização de algoritmos que possibilitam extrair regras de associação. A extração de regras de associação buscam relações entre atributos e isso permite mostrar que conjunto de valores de atributos causam aquela classe, e a partir disso se pode classificar dados.

3.2 Extração de regras de associação

A tarefa de extração de regras de associação tem o intuito de identificar associações entre registros de dados que, de alguma maneira, estão ou devem estar relacionados. Sua premissa básica é encontrar elementos que implicam na presença de outros em uma mesma transação SCHUNEIDER (2002).

Uma análise de associação descobre relacionamentos que podem ser expressos na forma regras de associação ou conjunto de itens frequentes. Uma regra de associação é uma expressão de implicação no formato $X \rightarrow Y$, onde X e Y são conjuntos disjuntos de itens, ou seja, $X \cap Y = \emptyset$ (TAN et al., 2009).

A Tabela 1 ilustra um conjunto de dados, onde cada linha corresponde uma transação com um identificador único e um conjunto de itens comprados por um determinado cliente e a partir dela uma regra pode ser extraída.

Tabela 1 – Exemplo de Transações de cestas de compras.

ID	**Itens**
1	{Pão, Leite}
2	{Pão, Fraldas, Cerveja, Ovos}
3	{Leite, Fraldas, Cerveja, Cola}
4	{Pão, Leite Fraldas, Cerveja}
5	{Pão, Leite Fraldas, Cola}

Fonte – Tan et al. (2009)

Tan et al. (2009) utilizam como exemplo a regra $\{Fraldas \rightarrow Cerveja\}$. Eles sugerem que essa regra pode ter um relacionamento forte entre a venda de fraldas e cerveja porque muitos clientes que compram fraldas também compram cerveja. Varejistas podem usar este tipo de regras para ajudar na identificação de novas oportunidades cruzando essas informações de vendas dos seus produtos para os clientes, como no caso organizar o estabelecimento onde as fraldas ficam próximo do local onde ficam as cervejas.

3.2.1 Suporte e Confiança

Extrair regras tem alto custo computacional e por isso algumas métricas são utilizadas. Dessas métricas as mais importantes são o suporte, a confiança e a força da regra como explica Amaral (2016), o suporte mede o número de transações que contém todos os itens da transação. Já a confiança, indica a proporção de vezes que uma transação contendo o elemento A também contém o elemento B. A força da regra é a soma do suporte mais a confiança. Na mineração de regras de associação, é preciso determinar um suporte e confiança mínimo para geração dessas regras.

As definições formais dessas duas métricas são dadas pelas equações (TAN et al., 2009):

$$Suporte, s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N} \tag{3.1}$$

$$Confiança, c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \tag{3.2}$$

Na equação 3.1, o suporte é dado pelo número de ocorrências que X e Y estão juntos em cada uma das transações ($\sigma(X \cup Y)$) dividido pelo número total de transações (N). Já na equação 3.2, define-se a confiança como a número de ocorrências que X e Y estão juntos em cada uma das transações ($\sigma(X \cup Y)$) dividido pela quantidade de vezes que X aparece em cada transação ($\sigma(X)$).

Utilizando a Tabela 1 e analisando a regra $\{Leite, Fralda\} \rightarrow \{Cerveja\}$, Tan et al. (2009) explicam como se dá o cálculo do grau de suporte e confiança. Tem-se que a transação $\{Leite, Fralda, Cerveja\}$ aparece em 2 das 5 transações, logo o seu grau de suporte é $2/5 = 0.4$ na base de dados, ou seja, para 40% das transações que contém leite e fralda a regra está correta. Já a confiança da regra utiliza o contador de suporte da transação $\{Leite, Fralda, Cerveja\}$ que é 2 e o contador de suporte de $\{Leite, Fralda\}$ que é 3 (Leite e Fralda aparecem juntos em 3 das 5 transações). O grau de confiança para esta regra é $2/3 = 0.67$. E por fim, a força da regra é $0.4 + 0.67 = 1.07$.

Segundo Tan et al. (2009) utilizar essas métricas nas regras são necessárias pois alguns dos padrões descobertos são potencialmente falsos já que podem acontecer simplesmente ao acaso. O suporte é útil, pois uma regra que tenha baixo suporte pode acontecer por coincidência e uma grande probabilidade de ser interessante a partir de uma perspectiva de negócio e por estas razões, é usado para eliminar regras sem interesse. Já a confiança mede o grau de confiança para uma determinada regra $X \rightarrow Y$. Quanto maior a confiança, maior a probabilidade de que Y esteja presente em transações que contenham Y. A confiança também fornece uma estimativa da probabilidade condicional de Y dado X.

3.2.2 Regras do tipo $SE \rightarrow ENTÃO$

Uma forma de representar o conhecimento nas regras de associação é da forma $SE \rightarrow ENTÃO$. O conhecimento descoberto é expresso como um conjunto de regras do tipo $SE \rightarrow ENTÃO$ que também são chamadas regras de produção e constituem uma forma de representação simbólica.

A parte SE é o antecedente da regra formado a partir de expressões condicionais envolvendo os atributos do domínio do banco de dados e a parte $ENTÂO$ é o consequente,

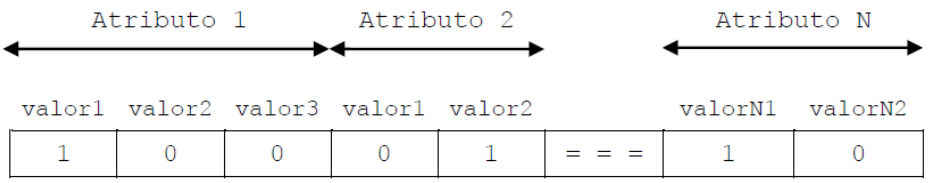
formado por expressões que indicam a previsão de algum valor para um atributo meta, obtido em função dos valores encontrados nos atributos que compõem o antecedente (MORAIS; DOMINGUES; OLIVEIRA, 2005).

3.3 Um modelo de Algoritmo Genético para classificar dados

Morais, Domingues e Oliveira (2005) defendem que modelos de Algoritmos Genéticos são ótimos para gerar regras de associação com o intuito de classificar, pois geram regras de fácil interpretação para o usuário e que exprimem uma realidade do domínio da aplicação. Eles argumentam que estas regras são modeladas na forma de regras de associação, do tipo $SE \rightarrow ENTÃO$, entre os atributos de predição (chamados por eles de antecedentes das regras) e o atributo objetivo ou atributo-classe (denominado consequente da regra). E com isso propuseram e aplicaram um algoritmo genético que a partir da base de dados pudesse encontrar relações entre os atributos e uma determinada classe.

As instâncias da base eram codificadas de forma binária a partir de uma técnica chamada de abordagem de Michigan, uma representação em que um cromossomo representa uma regra de associação da forma SE (A_1 e A_2 e ... A_n) então P. O conjunto (A_1 e A_2 e ... A_n) representa os atributos de predição e P, o valor da classe alvo (MORAIS; DOMINGUES; OLIVEIRA, 2005), como se pode ver na Figura 4.

Figura 4 – Representação binária do cromossomo.



Fonte – Morais, Domingues e Oliveira (2005)

Na execução do algoritmo, apenas instâncias de mesma classe eram executadas, enquanto as outras que não tinham o mesmo rótulo eram ignoradas. No final, eram extraídas regras com os atributos mais importantes que tinham uma maior relacionamento com a classe alvo e a partir disso classificar se uma instância nova é ou não pertencente a classe alvo com base nos atributos mais relevantes.

Um modelo de algoritmo genético proposto, foi por Han, Kamber e Pei (2006). Esse modelo é mais generalista e no contexto da Mineração de Dados, o qual a classificação faz parte como abordado na seção 3.1. Para eles, em geral, a aprendizagem genética, como denominaram, funcionaria da seguinte forma, uma população inicial é criada consistindo de regras geradas aleatoriamente e não a partir de uma base de dados. Cada regra pode ser representada por uma sequência de bits semelhantes a abordagem de Michigan e que

nesse modelo poderia considerar mais de duas classes distintas. Eles apresentam que se um atributo tiver k valores, onde $k > 2$, então k bits podem ser usados para codificar os valores do atributo. As classes podem ser codificadas de uma forma semelhante. Ou seja, dado uma instância que possui um atributo A que pode ter somente como possíveis valores A_1 e A_2 e sua classe C que pode ser C_1 ou C_2, poderia codificar essa amostra de forma binária na ordem $A|C$:

SE A tem como valor A_1 E NÃO A_2 ENTÃO C_2 => 10|01

SE A tem como valor A_2 E NÃO A_1 ENTÃO C_1 => 01|10

Com base na noção de sobrevivência do mais apto, uma nova população é formada para consistir nas regras mais aptas na população atual, bem como a descendência dessas regras. Normalmente, a adequação de uma regra é avaliada por sua precisão de classificação em um conjunto de amostras de treinamento através de uma função de aptidão.

O processo de geração de novas populações com base em populações anteriores de regras formam novas regras de associação e continua até que uma população, P, evolua, onde cada regra em P satisfaça uma aptidão especificada previamente por um limiar e a geração é criada pela aplicação de operadores genéticos, como *crossover* e mutação. No cruzamento, *substrings* de pares das regras são trocados para formar novos pares das regras. Na mutação, bits selecionados aleatoriamente em uma *string* de regra são invertidos (HAN; KAMBER; PEI, 2006).

Com base nesses modelos de algoritmos genéticos apresentados por Han, Kamber e Pei (2006) e por Morais, Domingues e Oliveira (2005), no próximo capítulo são realizados dois estudos de caso onde se aplicam algoritmos genéticos em duas base de dados e a partir delas são geradas regras de associação simples para classificação.

4 Estudos de caso: Classificação de dados com Algoritmos Genéticos

Este capítulo ilustra os conceitos de algoritmos genéticos aplicados em dois experimentos com o objetivo de extrair regras de associação com o intuito de classificar dados. A seção 4.1 mostra as bases de dados utilizadas. Na parte 4.2 aborda sobre como foi feita a modelagem do indivíduo e da população do algoritmo genético. A seção 4.3 aborda sobre a função de aptidão. A seção 4.4 descreve os operadores genéticos e como foram modelados para rseolução do problema. E por fim, na seção 4.6 descreve e discute os experimentos realizados.

4.1 As bases de dados utilizadas

4.1.1 Base de dados *Classificação de Automóveis*

Para a realização do primeiro experimento, foi utilizada a base de dados *Car Evaluation Data Set* (Conjunto de dados de classificação de automóveis), um banco de dados de classificação de automóveis que armazena registros para saber se um carro é *inaceitável*, *aceitável*, *bom* ou *muitobom* a vista do consumidor (DUA; GRAFF, 2017). A base foi obtida no repositório da UCI Machine Learning Repository (Repositório de aprendizado de máquina da Universidade da Califórnia, disponível em <http://archive.ics.uci.edu/ml>) e contém 1728 instâncias, 6 atributos e todos são categóricos. Os atributos de predição são: preço de compra (muito alto, alto, médio, baixo), preço de manutenção (muito alto, alto, médio, baixo), número de portas (2, 3, 4, 5 ou mais), quantidade de pessoas (2, 4, 5 ou mais), tamanho da bagageira (pequeno, médio, grande) e segurança (baixa, média, alta). A Tabela 2 mostra o número de instâncias por classe.

Tabela 2 – Distribuição das instâncias em número e porcentagem das classes da base de dados de classificação de automóveis.

Classe	**Quantidade**	**Porcentagem**
Inaceitável	1210	70.023
Aceitável	384	22.222
Bom	69	3.993
Muito Bom	65	3.762

Fonte – Produzido pelo autor

Já a Tabela 3, mostra onze das instâncias contidas na base de dados.

Tabela 3 – Visão parcial da base de dados *Classificação de Automóveis*

Atributos de predição						**Atributo alvo**
preco_compra	**preco_manutencao**	**qtd_portas**	**qtd_pessoas**	**tam_bagageira**	**seguranca**	**classe**
muito alto	muito alto	2	2	pequeno	baixo	inaceitável
muito alto	muito alto	3	4	grande	alto	inaceitável
baixo	médio	2	4	médio	alto	bom
alto	muito alto	2	4	grande	baixo	inaceitável
alto	alto	2	mais	grande	médio	aceitável
alto	alto	2	mais	grande	alto	aceitável
médio	baixo	5mais	mais	grande	baixo	inaceitável
médio	baixo	5mais	mais	grande	médio	bom
médio	baixo	5mais	mais	grande	alto	muito bom
médio	médio	2	4	grande	alto	muito bom
baixo	médio	2	4	médio	médio	aceitável

Fonte – Produzido pelo autor

4.1.2 Base de dados *Golf-Weather*

Para o segundo experimento foi utilizada a base de dados *Golf-Weather* (RAHMAN, 2020). O *dataset* (disponível em <https://gist.github.com/kudaliar032>) contém dados sobre as condições meteorológicas adequadas para jogar uma partida de golfe. Muito menor que a base de classificação de automóveis, com 14 instâncias, quatro atributos de predição estado do tempo (ensolarado, nublado, chuvoso), temperatura (quente, média, fria), umidade (alta, normal), está ventando? (falso, verdadeiro) e a classe joga (sim, não). A Tabela 4 mostra a distribuição das instâncias por classe em quantidade de ocorrências e porcentagem em relação a base. Já a Tabela 5 mostra toda a base de dados.

Tabela 4 – Distribuição das instâncias por classe em número e porcentagem da base de dados *Golf-Weather*

Classe	**Quantidade**	**Porcentagem**
Sim	9	64,286
Não	5	35,714

Fonte – Produzido pelo autor

Tabela 5 – Base de dados *Golf-Weather*

Atributos de predição				**Atributo alvo**
tempo	**temperatura**	**umidade**	**ventando**	**joga**
ensolarado	quente	alta	FALSO	não
ensolarado	quente	alta	VERDADEIRO	não
nublado	quente	alta	FALSO	sim
chuvoso	media	alta	FALSO	sim
chuvoso	fria	normal	FALSO	sim
chuvoso	fria	normal	VERDADEIRO	não
nublado	fria	normal	VERDADEIRO	sim
ensolarado	media	alta	FALSO	não
ensolarado	fria	normal	FALSO	sim
chuvoso	media	normal	FALSO	sim
ensolarado	media	normal	VERDADEIRO	sim
nublado	media	alta	VERDADEIRO	sim
nublado	quente	normal	FALSO	sim
chuvoso	media	alta	VERDADEIRO	não

Fonte – Produzido pelo autor

4.2 População e representação do indivíduo no estudo de caso

Neste trabalho, utilizou-se as próprias bases de dados para formar a população do algoritmo genético, onde cada instância é um indivíduo da população e codificados pela abordagem de Michigan como feito no trabalho de Morais, Domingues e Oliveira (2005). Cada alelo desse cromossomo recebeu o bit 1, caso tivesse aquele valor do atributo, e bit 0, caso não tivesse. Os atributos da base de dados são todos categóricos, se houvesse atributos contínuos, Morais, Domingues e Oliveira (2005) defendem que eles deveriam ser discretizados.

Considerando a parte da base de dados mostrada na Tabela 3 e a representação binária ilustrada na Figura 4, por exemplo, a parte da base de dados teve como resultado obtido a tabela 6.

Tabela 6 – Visão parcial da base de dados da classificação de automóveis

Atributos de predição																				**Atributo alvo**			
preco_compra				**preco_manutencao**				**qtd_portas**				**qtd_pessoas**			**tam_bagageira**			**seguranca**			**classe**			
muito alto	alto	médio	baixo	muito alto	alto	médio	baixo	duas	três	quatro	cinco ou mais	duas	quatro	mais	pequeno	médio	grande	baixo	médio	alto	inaceitável	aceitável	bom	muito bom
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0
1	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	1	0	0	0
0	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	1	0
0	1	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0
0	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	1	0	0	1	0	0	0
0	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	1	0	0	0	1	0
0	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0

Fonte – Produzido pelo autor

Nessa modelagem, os registros da base são interpretados como regras de associação do tipo $SE \rightarrow ENTÂO$, onde cada atributo representa uma condição da regra. A primeira instância da Tabela 6 pode ser lida da seguinte forma, por exemplo:

SE preço da compra é muito alto E preço de manutenção é muito alto E quantidade de portas são duas E quantidade de pessoas são duas E tamanho da bagageira é pequeno E segurança é baixa ENTÃO o carro é inaceitável.

Essa forma de representação mostra o quão complexo pode ser para o usuário interpretar a relação dos atributos e a classe, caso ele queira saber qual atributo teria maior ou menor relevância para que o carro tenha avaliação inaceitável. Por esse motivo, justifica a função de aptidão utilizada com o objetivo de minimizar o número de condições de forma a obter regras mais simples. Isso é explicado com mais detalhes na próxima seção.

4.3 Função de aptidão utilizada

Caso existisse uma regra na Tabela 6 de tal forma que nem todos os atributos estejam presentes como:

SE preço da compra é muito alto ENTÃO o carro é inaceitável

Temos uma regra com menos condições nos antecedentes. Essa forma seria bem útil para o usuário final interpretar e compreender. Segundo os resultados obtidos no trabalho de Romão (2002) apud Morais, Domingues e Oliveira (2005), foi constatado que regras com mais de três condições no antecedente tornam-se difíceis de serem interpretadas pelos usuários. Esse fato motivaram Morais, Domingues e Oliveira (2005) a modelarem em seu trabalho de AG com mineração de dados a utilizarem como função de aptidão minimizar o número de condições de forma a obter regras mais simples. Com isso, nesse trabalho utilizou a mesma função de aptidão. A base de dados de *classificação de automóveis* possui

6 atributos como se pode observar na Tabela 6 e o *dataset Golf-Weather*, 4 atributos. Como o objetivo é minimizar esses números, então a função de aptidão é a equação 4.1, dada pela somatória do número de atributos no antecedente da regra da base de dados.

$$f(x) = \sum_{i=0}^{n} i \tag{4.1}$$

4.4 Operadores genéticos

Na seleção, utilizou-se o método da roleta. Como a função de aptidão busca minimizar o número de atributos para extrair regras mais simples, todos os indivíduos iniciam com peso 0, ou seja, mesma probabilidade de serem selecionados.

Na etapa de cruzamento, optou-se pelo *crossover* de um ponto pela simplicidade, onde duas instâncias ou melhor conjunto de regras são selecionados e a partir de uma dada probabilidade, dois novos indivíduos são gerados para formar a nova população. Esses dois filhos são formados a partir de uma recombinação ou como cópias exatas dos pais.

A mutação visa dar diversidade a população através de um processo aleatório onde um gene é alterado para produzir um novo indivíduo. Foi colocada com uma taxa bem baixa como explicado na seção 2.3.4.3 e assim gerar novos conjuntos de regras da base, aumentando assim o espaço de busca.

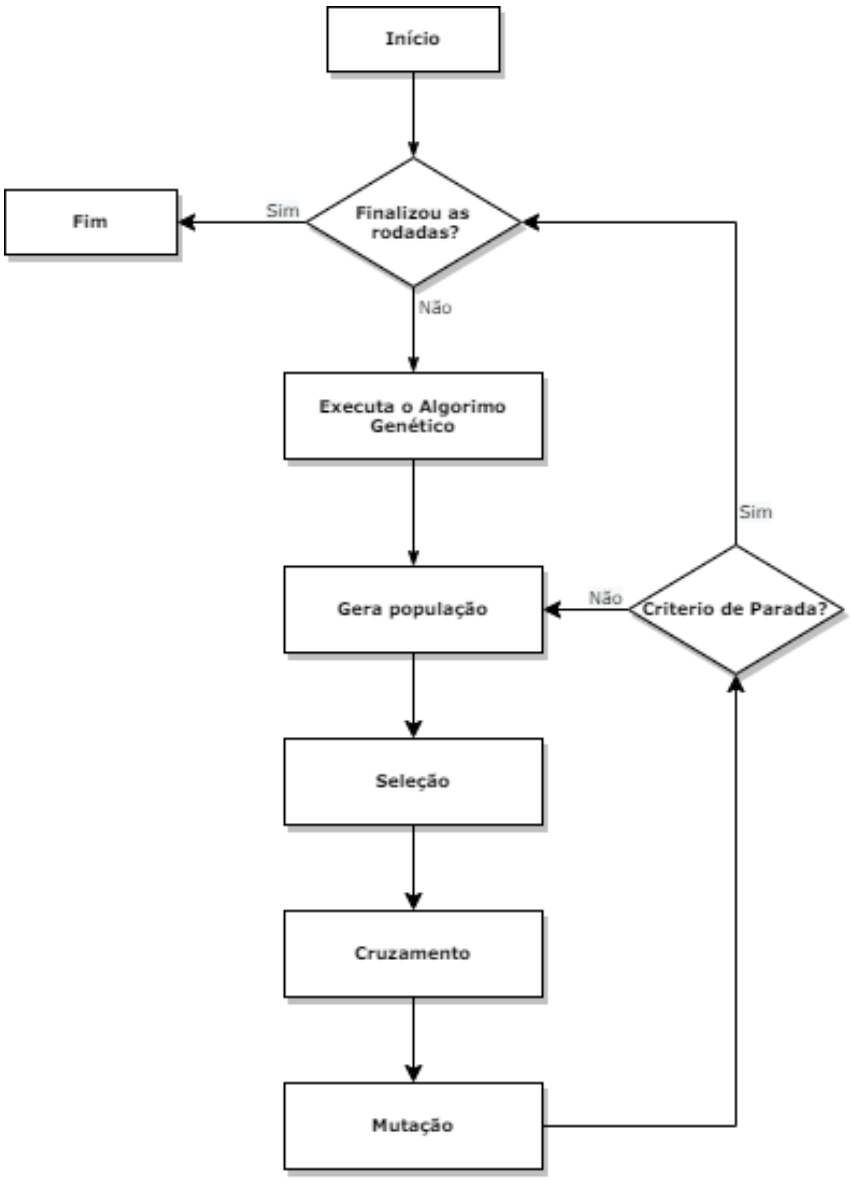
4.5 Ferramentas utilizadas para construção do algoritmo genético

Para a fase de programação do algoritmo genético, utilizou-se a linguagem de programação *Python*, na versão 3.8.5. Esta linguagem possui diversas bibliotecas que facilitaram o desenvolvimento da aplicação como *numpy* (versão 1.19.2), muito útil para escrever algoritmos de *Machine Learning* e realizar cálculos numéricos em *arrays*. Outra biblioteca importante foi o *pandas* (versão 1.1.3) para manipulação e análise de dados do *dataset*.

4.6 Experimentos realizados

O fluxograma na Figura 5 mostra como foi modelado o algoritmo genético para os experimentos em 4.6.1 e 4.6.2. Nessas seções são descritas mais detalhadamente o que é feito em cada parte do algoritmo.

Figura 5 – Fluxograma do Algoritmo genético utilizado nos experimentos.



Fonte – Produzido pelo autor

4.6.1 Estudo de Caso I: *Classificação de Automóveis*

Esse trabalho aplicou o AG na classe *muito bom* e *inaceitável*, pois, pensando do ponto de vista do consumidor, o ideal seria comprar os carros que fossem melhor avaliados e excluir os piores, e desse modo extrair regras de associação a fim de verificar quais atributos possui relação com a classe alvo, classificando o melhor ou pior carro. Os dados foram manipulados, separados e convertidos para uma representação binária. Teve-se um problema, pois atributos diferentes possuíam mesmo nome no valor, por exemplo, *preço de compra = médio* e *segurança = médio*, como o algoritmo ao executar poderia diferenciar o médio preço de compra do médio grau de segurança do automóvel ou de outros atributos? Como solução o nome dos valores foram renomeados:

- *preço de compra*: {*muito alto, alto, médio, baixo*} → {*pc_m_alto, pc_alto, pc_med, pc_baixo*};
- *preço de manutenção*: {*muito alto, alto, médio, baixo*} → {*pm_m_alto, pm_alto, pm_med, pm_baixo*};
- *quantidade de portas*: {*2, 3, 4, 5 ou mais*} → {*qport_2, qport_3, qport_4, qport_5mais*};
- *quantidade de pessoas*: {*2, 4, 5 ou mais*} → {*qpes_2, qpes_4, qpes_mais*};
- *tamanho da bagageira*: {*pequeno, médio, grande*} → {*tb_peq, tb_med, tb_grande*};
- *segurança*: {*baixa, média, alta*} → {*s_baixa, s_med, s_alta*};
- *classe*: {*inaceitável, aceitável, bom, muito bom*} → {*inac, aceit, bom, mbom*}.

A fim de formar a população, considerou-se todas as instâncias que possui como classe o valor *muito bom*, dando um total de 65 instâncias. Essas 65 passaram a formar a população e serem cada uma vista como uma transação. Para a classe *inaceitável* foi um total de 1210 indivíduos (ou instâncias).

O indivíduo era formado por um cromossomo com 21 bits. Na seleção da roleta para dar pesos maiores àqueles indivíduos que possuíam menos atributos, utilizou a fórmula 4.2:

$$P = 1 - \frac{K}{6} \tag{4.2}$$

Pela equação 4.2, que P é o peso e K representa o número de atributos que aquela instância possui. Foi dividido por 6, pois era o número máximo de atributos. Assim, no início da execução todas as instâncias tinham peso 0. Com o passar das gerações apareciam filhos com menos atributos. Por exemplo, sejam X_1 e X_2 dois indivíduos dentro de uma população, P_1 e P_2 seus respectivos pesos e K_1 e K_2 o número de seus atributos.

$$X_1 : 100000000000010000000$$

$$X_2 : 100001000000010000100$$

O indivíduo X_1 tem dois atributos, então $K_1 = 2$ e X_2 tem quatro atributos, logo $K_2 = 4$. Desse modo, calculamos os pesos de cada um, P_1 fica com $1 - (2/6) = 0.67$ e P_2 fica com $1 - (4/6) = 0.34$. Assim X_1 possui peso maior que X_2 e com isso tem probabilidade maior de ser selecionado.

Os selecionados foram submetidos a um teste, onde se gerava um número aleatório e o comparava com a taxa de cruzamento. Caso a probabilidade gerada aleatoriamente fosse maior que a taxa de cruzamento, era feito o cruzamento de um ponto com os selecionados

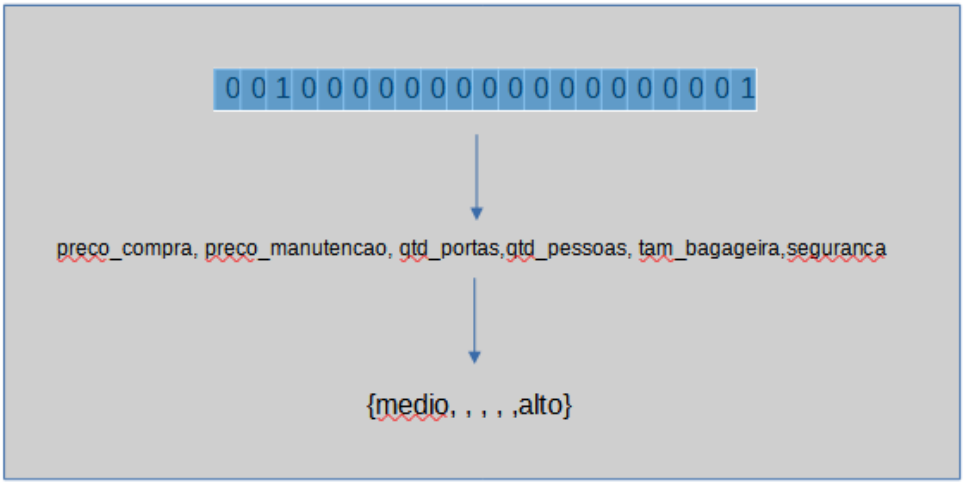
e geravam dois filhos, os quais formavam a nova população que passava para geração seguinte. Caso a probabilidade gerada não fosse maior, cópias idênticas aos selecionados iriam compor a nova população. A taxa de cruzamento nesse experimento foi de 60%.

A taxa de mutação foi de 10%. Caso a probabilidade gerada aleatoriamente fosse maior que a taxa de mutação, o escolhido para compor a nova população sofria mutação. Como sempre eram inseridos de dois em dois, esse teste foi feito duas vezes.

O critério de parada utilizado foi o número de rodadas com valor 100. No experimento foram realizados 100 rodadas e cada rodada executava 60 gerações do algoritmo genético. Assim que finalizava o número de gerações de cada rodada, o resultado gerado era uma matriz com valores binários, onde cada linha era uma transação. Regras repetidas foram desconsideradas. As vezes apareciam transações nulas e transações com o mesmo número de atributos do início. Para resolver esse problema, criou-se dois parâmetros, número de atributos máximos para o resultado, o qual foi setado com 3 e número de atributos mínimos, com valor 1. Essa matriz era ordenada, para que a regra que possuía menos atributos sempre fique no topo.

A matriz binária foi convertida em uma matriz com os valores dos atributos categóricos. A Figura 6 ilustra isso:

Figura 6 – Exemplo de conversão da matriz binaria para uma com os valores categóricos da regra.



Fonte – Produzido pelo autor

Essa matriz da Figura 6 mostra o resultado da conversão da matriz binária para uma regra no formato de *array* $\{medio, , , , alto\}$ que pode ser interpretada como "*Se o preço de compra é médio e a segurança é alta então o carro é muito bom*". Com essa regra extraída era calculada o grau de confiança e armazenada numa planilha.

O resultado final após as cem execuções foi uma planilha que continha todas as regras extraídas, a quantidade de vezes em que elas foram geradas e o grau de confiança

de cada uma delas mostrada na seção 5.

4.6.2 Estudo de Caso II: Base de dados *Golf-Weather*

Nessa base de dados, a classe alvo foi *sim*, então foram selecionados 9 instâncias. De maneira análoga ao experimento 4.6.1, os dados foram transformados para a forma binária e lançados no algoritmo genético. Os mesmos parâmetros de taxa de mutação, quantidade de rodadas, taxa de cruzamento, número mínimo e máximo de atributos de regras a serem extraídas do estudo de caso I na seção 4.6.1 foram utilizados, mas o número de gerações foi 30, pois a população era bem pequena. As 9 instâncias formaram a população e cada indivíduo foi formado por um cromossomo de 10 bits. Na seleção da roleta para dar pesos maiores foi utilizada a fórmula 4.3:

$$P = 1 - \frac{K}{4} \tag{4.3}$$

Então, pela equação 4.3, que P é o peso e K representa o número de atributos que aquela instância possui. Foi dividido por 4, pois era o número máximo de atributos. No final, as regras extraídas iam para uma planilha a quantidade de vezes em que elas foram geradas e o grau de confiança de cada uma delas mostrada na seção 5.

5 Resultados e discussão

No estudo de caso I foram avaliadas as regras da base de dados *Classificação de automóveis* nas classes *muito bom* e *inaceitável*. As regras induzidas de classificação simples continham no máximo 3 condições e no mínimo 1 e foram extraídas ao final de 100 rodadas com 30 gerações cada no algoritmo genético.

Para classe *muito bom* foram extraídas 329 regras no final da execução. A Tabela 7 mostra uma visão parcial das regras extraídas ordenadas pelo número de ocorrências que aparecem e com seus respectivos graus de confiança. Já a Tabela 8, mostra as regras extraídas ordenadas pelo grau de confiança.

Tabela 7 – Visão parcial da tabela gerada no final do estudo de caso na classe *muito bom* da base de dados *Classificação de Automóveis* ordenada pelo número de ocorrências.

	Nº de ocorrências	**Confiança %**
pm_med, mbom	28	6,02
qpes_2, mbom	28	0,00
qpes_mais, mbom	28	6,08
pc_alto, mbom	27	0,00
pc_med, mbom	27	6,02
pm_alto, mbom	27	3,01
s_med, mbom	27	0,00
qport_2, mbom	27	2,31
tb_peq, mbom	27	0,00
pm_m_alto, mbom	26	0,00
pc_baixo, mbom	25	9,03
qport_3, mbom	25	3,47

Fonte – Produzido pelo autor

As três regras mais simples encontradas com maior número de ocorrências e seus respectivos graus de confiança pela Tabela 7 e que permitem classificar um carro como *muito bom* foram:

SE o preço de manutenção é médio, ENTÃO o carro é muito bom (Nº de ocorrências = 28 e Confiança = 6,02%).

SE a quantidade de pessoas que cabem no carro são 2, ENTÃO o carro é muito bom (Nº de ocorrências = 28 e Confiança = 0%).

SE a quantidade de pessoas que cabem no carro são mais de 4, ENTÃO o carro é muito bom (Nº de ocorrências = 28 e Confiança = 6,07%).

Tabela 8 – Visão parcial da tabela gerada no final do estudo de caso na classe *muito bom* da base de dados *Classificação de automóveis* ordenada pelo grau de confiança.

	Nº de ocorrências	**Confiança %**
pc_baixo, tb_grande, s_alta, mbom	1	50,00
pc_baixo, pm_baixo, s_alta, mbom	1	36,11
pc_baixo, qport_4, s_alta, mbom	1	33,33
pm_med, qpes_mais, s_alta, mbom	1	29,17
pc_baixo, s_alta,mbom	2	27,08
pm_baixo, qpes_4, s_alta, mbom	1	25,00
pc_med, qport_4, s_alta, mbom	1	22,22
tb_grande, s_alta, mbom	2	20,83
pc_med, tb_med, s_alta, mbom	1	20,83
qport_5mais, qpes_4, s_alta, mbom	1	20,83
pc_med, pm_med, qpes_mais, mbom	1	19,44
qpes_mais, s_alta, mbom	2	18,23

Fonte – Produzido pelo autor

E pela Tabela 8, as três regras mais simples encontradas com maior graus de confiança e seus respectivos número de ocorrências foram:

SE o preço de compra do carro é baixo, o tamanho da bagageira é grande e a segurança é alta, ENTÃO o carro é muito bom (Confiança = 50% e Nº de ocorrências = 1).

SE o preço de compra do carro é baixo, o preço de manutenção é baixo e a segurança é alta, ENTÃO o carro é muito bom (Confiança = 36,11% e Nº de ocorrências = 1).

SE o preço de compra do carro é baixo, a quantidade de portas é 4 e a segurança é alta, ENTÃO o carro é muito bom (Confiança = 33,33% e Nº de ocorrências = 1).

Já para classe "inaceitável" foram extraídas 289 regras contendo no máximo 3 condições e no mínimo 1 ao final de 100 rodadas com 30 gerações cada no algoritmo genético. Das 289 regras, 70 regras com 100% no grau de confiança e 219 regras com grau de confiança acima de 50%. Ou seja, teve um resultado mais satisfatório que o da outra classe utilizada no experimento. A Tabela 9 mostra uma visão parcial das regras extraídas que melhor classificam os carros como *inaceitável* ordenadas pelo número de ocorrências que aparecem e com seus respectivos graus de confiança. A Tabela 10, lista as regras ordenadas pelo grau de confiança.

Tabela 9 – Visão parcial da tabela gerada no final do estudo de caso na classe *inaceitável* da base de dados *Classificação de automóveis* ordenada pelo número de ocorrências.

	Nº de ocorrências	**Confiança %**
pc_alto, inac	27	75,00
tb_peq, inac	26	78,12
s_med, inac	25	61,98
qport_2, inac	25	75,46
tb_grande, inac	25	63,89
s_alta, inac	25	48,09
s_baixa, inac	25	100,00
pm_baixo, inac	25	62,04
qpes_2, inac	24	54,17
pc_med, inac	24	62,04
qport_4, inac	23	67,59
pc_m_alto, inac	23	83,33

Fonte – Produzido pelo autor

Tabela 10 – Visão parcial da tabela gerada no final do estudo de caso na classe *inaceitável* da base de dados *Classificação de automóveis* ordenada pelo grau de confiança.

	Nº de ocorrências	**Confiança %**
s_baixa, inac	25	100,00
qpes_2, inac	20	100,00
pm_alto, qpes_2, inac	5	100,00
pm_alto, s_baixa, inac	5	100,00
qpes_mais, s_baixa, inac	5	100,00
pm_med, s_baixa, inac	5	100,00
pc_m_alto, s_baixa, inac	4	100,00
qport_5mais, s_baixa, inac	3	100,00
pc_m_alto, pm_m_alto, inac	3	100,00
qport_4, s_baixa, inac	3	100,00
high, s_baixa, inac	3	100,00
pc_m_alto, pm_alto, inac	3	100,00

Fonte – Produzido pelo autor

As quatro regras mais simples e que permitem classificar um carro como inaceitável pela Tabela 9 foram:

SE o preço do carro é alto, ENTÃO é inaceitável (Nº de ocorrências = 27 e Confiança = 75%)

SE o tamanho da bagageira é pequena, ENTÃO o carro é inaceitável (Nº de ocorrências = 26 e Confiança = 78,12%)

SE a segurança é média, ENTÃO o carro é inaceitável (Nº de ocorrências = 25 e Confiança = 61,98%).

SE a quantidade de portas são 2, ENTÃO o carro é inaceitável (Nº de ocorrências = 25 e Confiança = 75,46%).

E pela Tabela 10, mostra as regras extraídas ordenadas pelo grau de confiança, foram:

SE a segurança é baixa, ENTÃO o carro é inaceitável (Confiança = 100% e Nº de ocorrências = 25).

SE a quantidade de pessoas que cabem no carro são 2, ENTÃO o carro é inaceitável (Confiança = 100% e Nº de ocorrências = 20).

SE o preço de manutenção é alto E a quantidade de pessoas que cabem no carro são 2, ENTÃO o carro é inaceitável (Confiança = 100% e Nº de ocorrências = 5).

SE o preço de manutenção é alto E a segurança é baixa, ENTÃO o carro é muito bom (Confiança = 100% e Nº de ocorrências = 5).

Na base de dados *Golf-Weather* foram geradas 94 regras e a Tabela 11 mostra algumas das regras extraídas que contém no máximo 3 condições no antecedente para a classe *sim*, ordenadas pela quantidade de vezes em que apareceram e seus respectivos graus de confiança.

Tabela 11 – Tabela gerada no final do experimento da base de dados *Golf-Weather* ordenada pelo número de ocorrências.

	Nº de ocorrências	**Confiança %**
nublado,sim	21	100,00
normal,sim	14	85,71
nublado,falso,sim	11	100,00
falso,sim	11	75,00
quente,sim	9	50,00
nublado,quente,sim	8	100,00
normal,falso,sim	8	100,00
fria,sim	8	75,00
verdadeiro,sim	7	50,00
nublado,normal,sim	7	100,00
media,sim	6	66,67
chuvoso,quente,sim	6	0,00

Fonte – Produzido pelo autor

As três regras mais simples encontradas com maior quantidade de ocorrências e seus respectivos graus de confiança em uma execução com 100 rodadas e com cada rodada 30 gerações cuja classe desejada foi a “sim” foram:

SE tempo está nublado, ENTÃO joga sim (Confiança = 100%)

SE a umidade está normal, ENTÃO joga sim (Confiança = 85,71%)

SE tempo está nublado E não está ventando, ENTÃO joga sim (Confiança = 100%)

O algoritmo mostrou-se satisfatório para a base de dados *Golf-Weather*, que é bem pequeno, com melhores resultados e menor tempo de execução. Já para o *dataset* maior como no caso a de *Classificação dos automóveis*, os resultados não foram bons para classe *muito bom* e com sucesso para classe *inaceitável*. Ao executar o experimento da classe *muito bom*, muitas regras tiveram como grau de confiança 0, isso se deve ao fato delas terem como número mínimo de condições uma regra no antecedente, pois uma condição não era forte suficiente para ter um grau de confiança alto e também algumas dessas regras foram geradas ao acaso. As melhores tiveram valores muito baixos e a sua execução demandou um maior tempo e alto custo computacional.

Indagou-se sobre o que fez os estudos de caso na base *Golf-Weather* cuja classe alvo foi a *sim* e na base *Classificação de Automóveis* de classe *inaceitável* terem obtidos resultados satisfatórios. Ambas tem em comum o maior percentual de instâncias classificadas em seus conjuntos de dados, como se pode observar na Tabela 2 e na Tabela 4, sendo de classe *sim* aproximadamente 64% e classe *inaceitável*, 70%. E por haver um maior número de instâncias, afetou no cálculo da confiança da regra, levando a induzir que havia uma relação sobre a quantidade de elementos classificados na classe alvo afetava no desempenho do algoritmo, já que na classe *muito bom* correspondia a apenas 3.76% no domínio da base de dados *Classificação de Automóveis*.

6 Conclusão

Este trabalho buscou mostrar uma abordagem de classificar dados através de algoritmos genéticos a partir da extração de regras simples de associação. Os resultados se mostraram satisfatórios na maioria das situações estudadas com graus de confiança das regras bem altos.

Houve um pouco de dificuldade na forma de representar os registros das bases de dados na forma binária, pois gerava um custo computacional significativo em codificar e decodificar os atributos e seus valores categóricos, mas com muita persistência funcionou.

O AG foi implementado e não utilizado pronto de alguma biblioteca e isso foi vantajoso, pois possibilitou que fossem feitas modificações que permitissem alcançar os objetivos do trabalho, como a criação de uma fórmula para os pesos nas aptidões dos indivíduos e assim serem utilizados durante a seleção no método da roleta.

Descobrir padrões a partir de um conjunto grande de dados de transações com algoritmos genéticos pode ser computacionalmente custoso. Alguns dos padrões descobertos foram falsos porque podem acontecer simplesmente ao acaso e como soluções automáticas não entendem o contexto e são, portanto, sujeitos ao erro, acabou ocorrendo resultados falsos positivos que podem frustrar os usuários e impedir procedimentos para classificação de dados, bem como erros de falsos negativos que expõem à perda de dados importantes.

Como sugestões para trabalhos futuros, ficam:

- Realizar uma codificação que englobe todas as classes a fim de serem avaliadas todas ao mesmo tempo e não apenas uma.
- A utilização do elitismo para que as melhores regras possam continuar nas próximas gerações do algoritmo genético;
- Utilizar o grau de confiança da regra unido ao menor número de antecedentes da regra na função de aptidão, assim possam ter maior probabilidade de serem selecionadas as regras mais simples e com maior grau de confiança, entretanto, poderá demandar um alto custo computacional, o que leva a mais uma sugestão abaixo;
- A construção de um AG adequado para a questão de grande quantidade de dados e custo computacional;

Referências

AMARAL, F. *Aprenda Mineração de Dados: Teoria e Prática*. 1st. ed. [S.l.]: Editora Alta Books, 2016. 240p. ISBN 8576089882. Citado 3 vezes nas páginas 14, 28 e 29.

BENTO, E. P.; KAGAN, N. Algoritmos genéticos e variantes na solução de problemas de configuração de redes de distribuição. 2008. Revista Controle Automação/Vol.19, no.3/Julho, Agosto e Setembro. São Paulo, 2008. Citado 2 vezes nas páginas 21 e 23.

CAMILO, C. O.; SILVA, J. C. da. Mineração de dados: Conceitos, tarefas, métodos e ferramentas. 2009. Disponível em: <https://rozero.webcindario.com/disciplinas/fbmg/dm/RT-INF_001-09.pdf>. Acesso em: 02 de novembro de 2020. Citado na página 14.

DARWIN, C. *On the Origin of Species by Means of Natural Selection*. London: Murray, 1859. Or the Preservation of Favored Races in the Struggle for Life. Citado na página 17.

DUA, D.; GRAFF, C. *UCI Machine Learning Repository*. 2017. Disponível em: <http://archive.ics.uci.edu/ml>. Citado na página 33.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. 1996. American Association for Artificial Intelligence, 1996. Citado na página 14.

FERNANDES, A. M. da R. *Inteligência Artificial – Noções Gerais*. Rio de Janeiro: Visual Books, 2003. Citado 6 vezes nas páginas 14, 19, 20, 22, 24 e 26.

HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*. 2nd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006. ISBN 1558609016. Citado 3 vezes nas páginas 15, 31 e 32.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press, 1975. Second edition, 1992. Citado na página 19.

MARSLAND, S. *Machine Learning: An Algorithmic Perspective*. 2nd. ed. [S.l.]: Chapman and Hall/CR, 2015. Citado 3 vezes nas páginas 24, 25 e 26.

MICHALEWICZ, Z. *Genetic algorithms + Data Structures = Evolution Programs*. 3rd. ed. [S.l.]: Springer-Verlag, 1996. Citado 5 vezes nas páginas 19, 21, 22, 23 e 24.

MITCHELL, M. *An Introduction to Genetic Algorithms*. Rio de Janeiro: Visual Books, 2000. Citado 4 vezes nas páginas 18, 23, 24 e 26.

MORAIS, L.; DOMINGUES, M.; OLIVEIRA, R. L. de. Algoritmo genético para a descoberta de regras de predição. 2005. Universidade Federal do Pará (UFPA). Belém, PA, 2005 Disponível em: <https://www.researchgate.net/publication/262859019_Algoritmo_Genetico_para_a_Descoberta_de_Regras_de_Predicao>. Acesso em: 02 de novembro de 2020. Citado 5 vezes nas páginas 15, 31, 32, 35 e 36.

OTERO, J. Algoritmos genéticos aplicados à solução do problema inverso biomagnético. 2016. Tese (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio). Rio de Janeiro, p. 63-64. 2016. Citado na página 27.

RAHMAN, A. *golf-dataset.csv*. 2020. Disponível em: <https://gist.github.com/kudaliar032/b8cf65d84b73903257ed603f6c1a2508>. Acesso em: 27 de novembro de 2020. Citado na página 34.

REGNER, A. A teoria darwiniana da seleção natural sem a leitura de malthus. 2002. In: III Encontro de Filosofia e História da Ciência do Cone Sul, Águas de Lindóia, SP, 2002, p. 48-64. Citado na página 17.

SCHUNEIDER, L. *Mineração de Dados – Conceitos*. 2002. Citado 2 vezes nas páginas 28 e 29.

SILVA, R. F. da; PIGNATA, M. I. B. *Charles Darwin E A Teoria Da Evolução*. [S.l.: s.n.]. Disponível em: <https://files.cercomp.ufg.br/weby/up/80/o/TCEM2014-Biologia-RicardoFernandesSilva.pdf>. Acesso em: 02 de novembro de 2020. Citado na página 17.

TAN, P.-N. et al. *Introdução ao data mining: mineração de dados*. [S.l.]: Ciencia Moderna, 2009. 928p. ISBN 9788573937619. Citado 3 vezes nas páginas 15, 29 e 30.

ZUBEN, F. J. V. Computação evolutiva: Uma abordagem pragmática. 2000. In: Anais da I Jornada de Estudos em Computação de Piracicaba e Região (1a. JECOMP), Piracicaba, SP, 2000, p. 25–45. Citado 7 vezes nas páginas 14, 17, 18, 19, 21, 24 e 25.